COMPUTER SCIENCE H446 PROJECT

Candidate Name: **Katie Day**

Candidate Number: **XXXX**

Centre Name: **XXXX**

Centre Number: **XXXX**

Contents

Analysis

Project Brief

To make a typing game software on a computer suitable for people without an internet connection to actively learn and improve their touch-typing skills in my project. My project will be ideal for multiple players to use the same computer, where their times and results will be stored in a database to analyse later and compete with each other. There will be a level of customisation available to the user to tailor their experience to their needs.

Existing Solutions to the Project

- *typeracer* – Website Based - https://play.typeracer.com/

The one the main solutions I have found is 'typeracer'. It is a free browser-based typing game that in addition to practicing by yourself also has a multiplayer aspect, which has people typing the same extract and whoever types it the fastest wins. This can be random people, or you can race against your friends specifically. Having both a single player mode and a multiplayer makes the typing game more versatile and interesting for first time users.

Another aspect is this solution tries to match people with the same skill level as you. This is done by having the ability of associating your stats with a username in a log in system. However, it should be noted that the log in is not required for you to use the typing game. This helps races be more competitive and the user more likely to keep on playing. Rather than the user becoming discouraged and leaving the website.

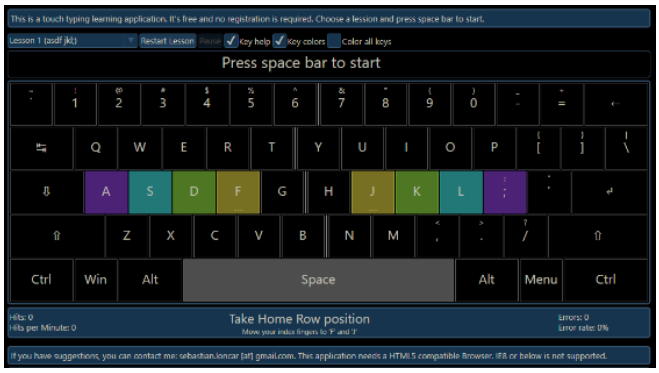
You can change the language of the extract, and an instant death mode toggle enabled. However, there is no way to eliminate punctuation or capital letters. The lack of customisation is notable, especially for beginners who might not feel comfortable practising letters that are not lower-case. But there is some customisation when choosing the type of extract, you want to practise against. For example, you can choose to practise one repeated word, have extracts based on themes (e.g., Anime) or have kid friendly extracts. This helps it become more mainstream and marketable to users of any ages however, I do think that having options to exclude punctuation and upper-case letter is a necessity. Overall, this website and its features is functional for users of any ages group and skill level. The choice of Instant Death can help users be more aware of their errors when typing and actively refrain from making mistakes and therefore promoting a better technique because of it instead of focusing on WPM speed exclusively.

As previously mentioned, users can choose to sign up to typeracer and can have their results logged each session and extract that they do. Typeracer also has a leaderboard for the 'Latest High Scores' that automatically refreshes every minute and a leaderboard of the fastest WPM on each extract. This helps develop user interactivity with the website and additionally adds extra competition and reflection to further better the user's typing skills. This encourages users to come back to the website regularly and promote users to improve their skills.

When typing a word on the website, if the character is correct it is highlighted green from a default grey colour, if a wrong the letter, it goes to a red forecolour. This is incredibly helpful for the user to quickly know their progress on that extract and when and where they have made an error and they can correct it quickly.

The UI is easy navigate and user friendly for new time users and there is a dark mode for night-time users. This helps the website be more accessible to users at night when it is dark, so it is less jarring for the eyes.

- *touchtyper.net* – Website Based - http://touchtyper.net/en/



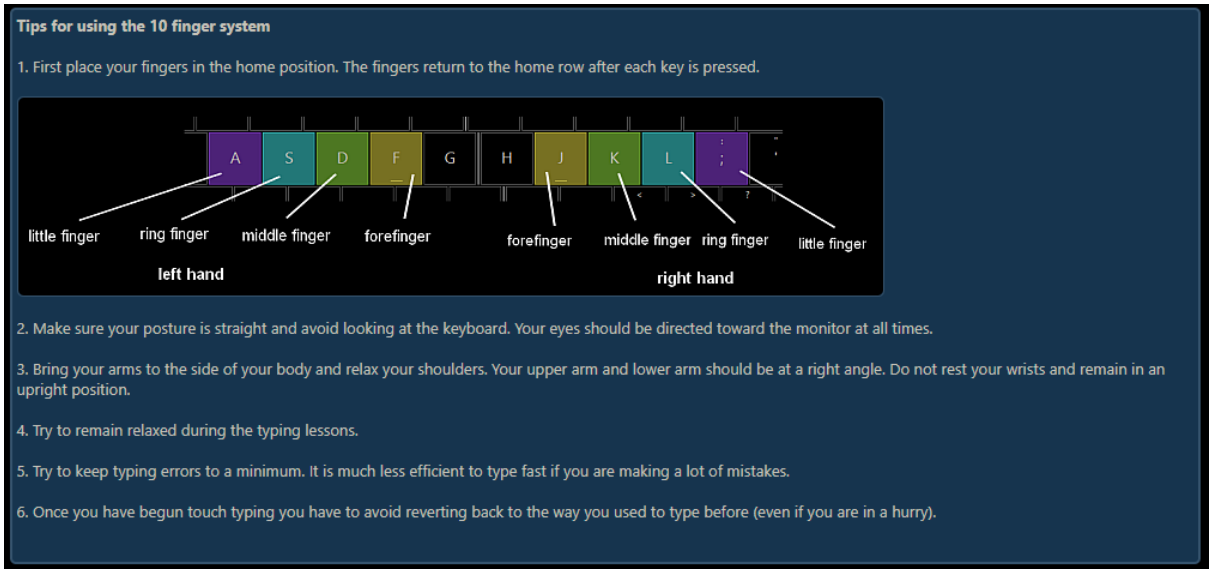
When I first started looking for solutions, one of the first to come up was 'touchtyper.net'. It is a website-based touch-typing service aimed at beginners to teach the basics of touch-typing to the user.

The site's strength is its UI that explicitly displays a keyboard which a key pressed by the user is shown on the keyboard. In this picture, the space bar has been pressed. Below the keyboard, there is informative text that will tell you the following action. So, if the following letter that should be pressed is an 's', then it will say "Left ring finger'. This encourages you not to look down at the keyboard and not get into bad habits. The user can customise the keyboard colouring and the presence of the informative text to their liking. What is also noting is that the colour of each key relates to the finger pressing on it. This is helpful for a beginner user to know what finger should press what key.

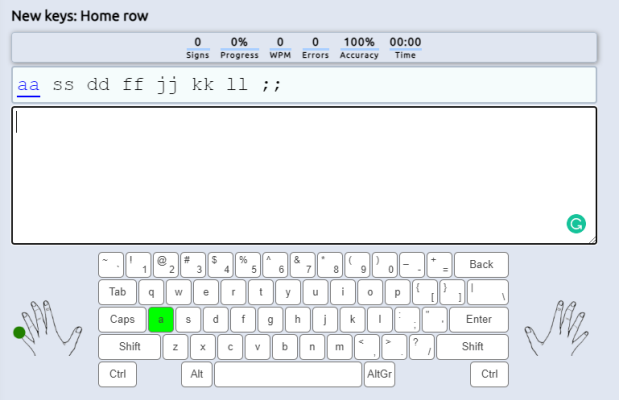
The extracts are comprised of 20 lessons that focus on specific letters, rows and numbers. This is very friendly for beginners, as they can first practise specific fingers and characters and create good habits and techniques. However, due to their only being 20 lessons and nothing else, there is no long-term use of the website. A person who has developed their touch typing skills beyond those 20 lessons will have no other use for this and will move on to another solution.

This solution also has colours in relation to the user pressing a key correctly or incorrectly. The justifications of this are explained in the previous solution.

Another feature that should be noted is the 'Tips' section of the website. Where it goes into detail about the correct technique, posture and finger placement that should be adopted, this is beneficial information for beginners who would not know this and it being an accessible part of the website means that it will be easy to access and use in their skills.



- *Touch Typing Study* – Website Based - https://www.typingstudy.com/



Like the previous solution, 'Touch Typing Study' aims at beginners who use lessons to help teach users. Additionally, it has a keyboard diagram that highlights the key that needs to be pressed next and diagrams of the right and left hand, highlighting the finger required to press the following letter.

I like the diagram in this solution more than the previous because the diagrams are more user friendly and help users know what finger they need to use more efficiently. Specifically, the hands highlighting a specific finger is particularly helpful in aiding the user to not 'look down' physically to their keyboard.

There are 15 separate lessons, each with multiple drills of different letters or words used. This helps users learn and hone in on the basics of touch typing. However, unlike the previous, 'Speed Test' and 'Typing Test' are what users can do after utilising the lessons. These entail multiple extracts where the user can practise touch typing repeatedly. This is without the aid of the keyboard and hand diagrams which cannot be toggled off or on.

Falling Blocks Game – Keys fall from the top and the user has to press the key shown beofre it hits the ground. If it hits the ground or a mistype, then a life out of 3 is taken

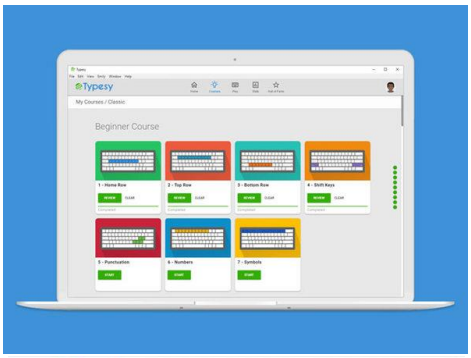
A negative with the solution is that when typing out a word and travelling to the next word, there is no way to go back and correct mistakes made on the previous word. The solution notes this as a wrong word but, the lack of retraction is unfavourable as the user cannot learn to quickly correct mistakes and redo words. It promotes inaccurate typing and bad muscle memory techniques. Additionally, there is little customisation of the experience regarding punctuation and capitals where seen previously. This would be helpful for users who would develop certain aspects of touch typing at different paces than the norm.

There are also four games about touch typing available to play in this solution. This help gives variety to the solution and provides the user with more activities to do. This makes it less likely they will get bored and leave the website.

However, this solution supports multiple languages and keyboard layouts. This would help more people be able to use these solutions and make them more mainstream.

- *Typesy* – Software Based

Typesy is a software-based touch typer that includes a paywall to access the software. It has multiple types, each with a different target market; for 'Homeschool', 'Education' and 'Individuals'. It can be accessed on various platforms and OS, most notably an App version for an iPad and supported on Windows, Mac, and Chromebook. This makes it accessible to anyone looking to learn typing and further widening your grasp of the market.



The solution consists of 4385 different lessons in total. It is divided into nine main courses that focus on various aspects of touch typing. The first course is shown on the left and is the 'Beginner' course, where each lesson focuses on a specific row or character subset. This method of teaching is comprehensive, which exposes the user to each row gradually and deliberately. This is especially useful for beginners who have no experience with typing at all as it encourages the user to adopt a good technique. Additionally, the significant number of lessons will help the user with muscle memory with a good technique that will help them in the future.

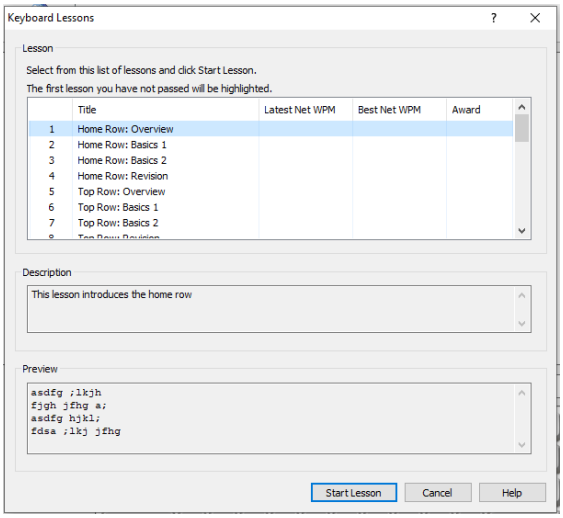
The following figure shows the user in a lesson that displays a speedometer in the top left to represent the current 'Words per Minute' that the user is currently typing at. This graphic and the software as a whole has a clean and simplistic aesthetic that is accessible to anyone. The WPM speedometer specifically promotes self-reflection and is very informative when the user is completing a lesson at that moment.

This solution likewise has a keyboard with hands graphics that changes depending on what character needs to be typed next. As said previously, this inhibits looking down at the keyboard when the user is stuck, but instead, you still look up and figure it out by just using coordination and still looking at the screen.

Other activities on 'Typesy' are games, each focusing on a specific weakness, e.g. Accuracy.

Another feature is that each user requires a login, where results from lessons are then associated with them and can then be transferred to different platforms or devices. This is very useful as it allows the user not to be limited to one specific device. Additionally, 'Typesy' provides a graphical representation of the user's statistics showing graphs of their progress development and their current level of WPM and Accuracy. This helps the user reflect, learn and become motivated with their progress and have a higher chance of using this service. However, a limitation is that a cloud-based database requires a stable, consistent connection to the internet. Which hinders the advantage of the solution being software-based if an internet connection is needed as a web browser-based solution would too.

- *Keyblaze* – Software Based



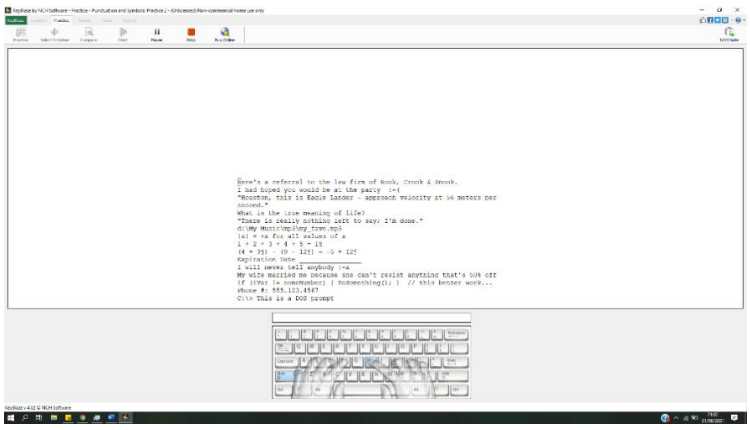
Keyblaze has a free version of their software, whose target market is any age group at all skill levels. When booting the program for the first time, it asks you for your age and skill level. Next, it gives you the option to perform an 'Initial Speed Test' where the program then asks you to set a WPM goal you would like to achieve. This feature helps the solution grasp the level of that user and what the user would like to achieve and cater for them accordingly.

The solution is comprised of 28 lessons which each have a focus on character subset or rows. When choosing the lesson, each has a description explaining it in more detail and a preview. Information like this is helpful for the user and ultimately boosts user experience. There is also a 'Practice' mode with different extracts divided into 'Prose', 'Poetry', 'Drills', 'Revision', 'Professional' and 'Custom'. You can pick specific extracts, and the 'Revision' section comprises extracts called 'Problem characters' and 'Problem words'. This changes according to the user's results. This level of customisation with the user is very beneficial for the user to focus on their weaknesses. Furthermore, there are three separate games called 'Key Blizzard', '30 Seconds to Type', and 'Typing Hero'. The variety of activities boost user experience and lengthen the time spent in the software.

This solution also has a graphical representation of the keyboard and a hand that changes to typing a specific key depending on the character needed to be typed next. This is very helpful for encouraging users not to look down when typing and use hand-eye coordination instead.

Keyblaze has a free version of their software, whose target market is any age group at all skill levels. When booting the program for the first time it asks you for your age and level. Next, it gives you the option to perform an 'Initial Speed Test' where the program then asks you to set you set a WPM goal you would like to achieve. This feature helps the solution grasp the level of that user and what the user would like to achieve and cater for them accordingly.

The solution is comprised of 28 lessons which each have a focus on character subset or rows. When choosing the lesson, each has a description explaining it in more details and a preview. Information like this is helpful for the user and ultimately boosts user experience. There is also a 'Practice' mode that has different extracts divided into 'Prose', 'Poetry', 'Drills', 'Revision', 'Professional' and 'Custom'. You have the ability to pick specific extracts and the 'Revision' section comprises of extracts called 'Problem characters' and 'Problem words'. This changes in according to the users results. This level of customisation with the user is very beneficial for the user to focus on their weaknesses.

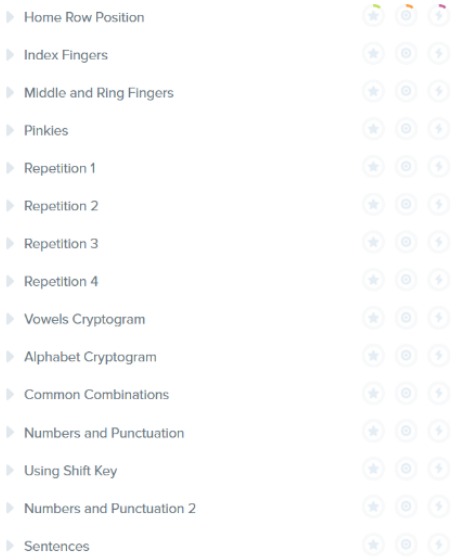


Furthermore, there are three separate games called ‘Key Blizzard’, ’30 Seconds to Type’, and ‘Typing Hero’. The variety of activities boost user experience and lengthen time spent in the software.

This solution also has a graphical representation of the keyboard and a hand, that changes to typing a specific key depending on the character needed to be typed next. This is very helpful for encouraging users to not look down when typing and to use hand-eye coordination instead.

- *RataType* – Website Based - https://www.ratatype.com/

This solution is an extensive touch-typing learner aimed towards children. It is divided into two main sections. ‘Typing tutor’ is comprised of different courses and several lessons within them. This is especially useful for first-time learners who have no experience with touch typing. All of the courses are shown on the right. In the lesson, the design is very clean and simplistic, with a colourful graphical representation of a keyboard to encourage users not to look down. The use of colours in specific keys allocated to each finger is a very nice feature that helps the user know what finger is for which key. This is very useful for beginners who may not understand which fingers are meant to press which keys, and a simple graphic like this will help them correct and have a good technique moving forward.

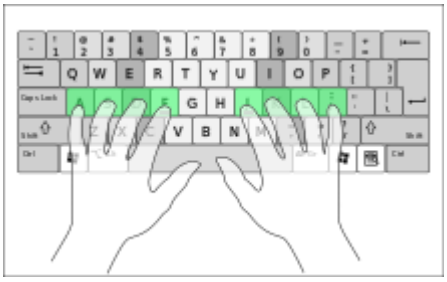


The following section is ‘Typing Test’, which has a random paragraph to type to the end. When typing, the accuracy and WPM is shown in real-time, changing depending on the user’s performance. This encourages the user not to make mistakes and not promoting any bad techniques. However, a disadvantage of this way is that there is no way to ‘backspace’ and correct your error. A small aspect of this solution is that when the user mistypes, there is no method of continuing, which is more akin to real-world typing, where mistakes are inevitable.

Another section is called the ‘Learn’ section, a web form of pointers about your posture, where to place your fingers on the home row, and other suggestions. This is inclined to beginners who will not have much experience or knowledge of good technique. However, I feel that this section should be shown at the beginning for new users, as this section can be easily missed, and the user can go straight into typing practise without any knowledge of the proper technique that should be used.

With a login system tied to your user results and experience, the user can add friends with a login in the solution to compete. This is great for the solutions own users to encourage friends and family to use the answer too. This led into the last main section, which is called ‘High Scores’. This is an assortment of leaderboards that are against friends or against everyone else using the results for today, this month and all time. This section encourages competitiveness and interactivity with the user and their friends, making the likelihood of the user continuing to use the solution more likely.

Target Market



1 - ©Cy21, Wikimedia Commons

My project will be primarily aimed at person(s) who already know the basics of touch typing and are at a beginner to intermediate level. I define this as people who have already learned the correct technique of positioning their finger on keys, as shown on the right. And want to practice their skills of getting a faster WPM (Words per Minute) and accuracy. Seeing their statistics of their time in the program can help them reflect and correct any inaccuracies and make them a better typer. To progress their typing skills, they would need to continually practise paragraphs and words instead of specific characters in a lesson-based way. This will help them become more fluid at typing words naturally and expose them to multiple words in the dictionary.

Likewise, touch typing services are mainly website based that is dependent on a connection to the internet. This is project will also target people who may have little to no internet connection at all; therefore, they would usually have no way of using a touch-typing service. In addition, people would still have some communal interaction of comparing results with other players using the program and can see their results even when they are offline.

I have chosen to make this project accessible to any age because the utilisation of personal computers in our day to day lives is already commonplace. And the skill of being able to type easily will aid people in using computers more comfortably and adequately. Especially for the elderly and young children, using a computer for the first time can be confusing and frustrating due to the unfamiliar feeling and dexterity issues that may hinder them when using a computer. This project can assist people in improving their fine hand movements as touch-typing aims to improve the hand-eye coordination of slight finger movement on specific keys, which can then be gradually improved from faster typing skills.

For young children, primary schools already teach some touch-typing lessons in ICT. However, parents will want to have an easy way of overseeing their progress when practising at home. They will see their child's advancement and see in detail when and what each entry they get is.

This will also be targeted at people with Dyslexia, and there will be font types, styles and sizes which can be tailored to the user's experience. This is very beneficial to people who find it hard to read text on a screen due to the poor, inaccessible UI. And when learning a new skill, the fewer hurdles there are to improvement means that it is more likely the user will continue to improve and consistently use the solution for a prolonged time.

Additionally, a simple program being launched off a desktop with no need for the internet is much more accessible than opening a web browser and finding the website online.

Essential Features of this Project

The main aim of this project is to teach and improve the touch-typing skills of the user. This entails encouraging good form (posture), technique and building up muscle memory in the fingers through continual and fun drills. The focus on making the typing fun and motivating for the user is of paramount importance. This is because if the user experience is adequate, then the likelihood of the solution being of repeated use for the user is better. I have planned for this by having three main

sections of the program to develop the user's skills, Typing Test, WordFade and Statistics. Having a variety of activities for the user to carry out will be less monotonous and increase user enjoyment.

The first and most important feature would be the Typing Test. This would be the user typing out an extract in the program. When the word inputted by the user is correct and corresponds to the extract, the program moves on to the next word. The test will not continue unless the word typed is correct and no mistakes are in the input. The solutions I have researched vary on how errors made by the user are tackled. However, I believe in letting the test progress when the user has made errors without correcting them. It encourages a lousy typing technique and is not akin to typing in real life, where errors in work are opposed.

Additionally, I will have the extract characters change from the default colour to Green when typed correctly or to Red if incorrect. When the user backspaces to correct, the character goes back to the default colour. This helps users know when they are typing correctly or incorrectly more efficiently and is an essential feature in the solutions I have researched.

WordFade is the same as Typing Test. However, there are only three words shown on the screen for the user to type. This is to help the user quickly type a word and not to see it in advance. This is more akin to a real-life situation of typing as a person forms and structures sentences in their minds.

An essential feature is a level of customisation of the user's experience with the 'Typing Test'. This will include options to allow Capital Letters and Punctuation in the extracts. Another option will be to change the type of extracts shown to the user. These options will include Paragraphs, Random Words, and the Alphabet. This will help the user independently tailor their experience to their needs and skill level. For example, a beginner might not be as confident with punctuation, so then can eliminate it.

Another feature that I would like to implement is a Login System. Each user will have a login and access the program using a username and password. New users will have the option to 'Add New User', which will open a new form to input their credentials. These credentials will be stored in a database. When logging into the program, the connected database will check the username and password and let them in if correct. This feature aims to keep track of who has connected to the program and to tailor and record the results of each session accordingly, which leads into my next feature. I will add this feature, so user's can track the entries of specific people and show and inform them of their progress. This will help them reflect and improve their touch-typing skills by seeing their weaker areas and then focusing on them.

I will also implement a guest mode where there is an option for a user to use the solution without the need for a login system and will not input results to the database. I have added this feature as not every person who will use my solution will want to have an account. Also, if the database connection cannot be initialised properly, users will not log into their accounts. The guest mode will be a solution so that people will still be able to use the solution.

This feature will be a way for each user to access their statistics. After each Typing Test or WordFade test is completed, the result, including WPM (Words Per Minute) and Accuracy, is to be inputted into the database concerning who the user was. They will see leaderboards of other users in the database of 'Highest WPM, 'Highest Accuracy', 'Average Highest WPM', 'Average Highest Accuracy' and 'Most Entries'. The user will also see a graph plotting their WPM/Date and Accuracy/Date. They will only be able to see their statistics. The user in this section will be able to edit their credentials and have the power to delete their account. Implementing this feature is to let each user see their results and reflect on their progress. This will help them become motivated to continue practising and correct any issues that they can identify. The leader-board will be a method to develop a competitive atmosphere

against other users and strive to become better. This will additionally maintain users using my solution for a longer duration as this feature will alleviate some of the monotonous feelings from constantly practising. Letting users edit and delete their login will let them have limited access to the database and aid them in changing any misinformation in the database. And free up storage if that login is not in use anymore.

A graphical representation of a keyboard is an essential part of the solution. In the existing solutions that I have researched, this is a crucial feature that aids users that are still very new to touch typing. This is because it encourages the user to practise good techniques of not looking down at the keyboard every time there is an error or if they are. However, if the user feels that they do not need the keys highlighted or the keyboard at all, there will be an option to turn it off. I will add this because some users may find it annoying and might not use it at all. I will also highlight the key that should be pressed next to help users know where it is on the keyboard. This will add accessibility to the solution.

Additionally, there will be users that will have Admin Privileges, where we have the power to see the individual entries of each user and be able to modify their credentials. This includes giving a user Admin Privileges too and the authority to delete users. This feature is essential as it lets the user have a clearer interaction with the database and see each entry.

Another essential feature would be a simple, easy to navigate interface with the user. This will help my solution be a more accessible and more coherent experience for any user at any age or impairment.

Limitations of my Solution

A limitation that I have identified is using the .NET framework to link the program with a Microsoft Access database. The solution is confined to Windows OS users. Eliminating macOS and Linux users who would equally want to use the solution as they need to use a keyboard daily too. This limitation is inevitable. However, I believe that this will not affect the user scope in a significant way, as Windows reportedly has a 76.12% market share of Desktop computers worldwide. Having my solution available to over three-quarters of users is still noteworthy for continuing my solution in the future.

Due to my program having no internet connection, I will not have a server to store paragraphs and words. Therefore, they will need to be already stored in the program on the user's computer. This means that there will be a limited number of paragraphs and words. This quantity will also be hindered by the loading times of the solution as these text files will need to be loaded into the main memory repeatedly. Modern computers are relatively fast, and this limitation should not be critical to the user's experience. However, it should be noted that I will not be able to store 1000s of paragraphs. Unfortunately, this means that a user may come across an extract that they have already typed before, weakening their experience. As stated earlier, I have decided for my solution to be offline because of the flexibility of using the program if the internet is not available compared to the majority of existing solutions being web browser based. Therefore, they are constrained to having a connection to the internet.

Additionally, a lack of connectivity to the internet means that features such as leaderboards that encourage competitiveness are diminished. Users can only compete with other users practising on the same device and cannot compete globally against others or friends on separate devices. This links with limits on social features, e.g. people can be friends with specific users across a network on a separate device. The justification of the decision with these limitations are discussed in the previous paragraph.

Another limitation is the vulnerability of using a Microsoft Access database. The data can be accessed easily by going into the program files and edited maliciously or accidentally, which may cause errors and exceptions in my solution or corrupt the database. This can somewhat be amended by separating the data management and application logic from each other. This splits the database into two files, one for the front-end (logic) and the other for the back-end (data). However, this is only useful when the back-end file is in a shared location like a server, and each user has access to the front-end. My program does not use a server or a method of sharing information over a network, and there is no sensitive information linking a user to a person in real life apart from their first name. I believe that a security breach will not be a significant hindrance or danger to users.

As I'm using a Microsoft Access database, if the user using the program does not have Microsoft Access, they will not have the ability to add users to the database. However, they will still have the ability to use Guest Mode but none of the features. I will include a hyperlink in the 'Options' form to the Microsoft Access runtime software, allowing Access databases to be interacted with without Access installed on the computer.

As a student creating the solution in a solo development process, there will be a limit to developing and implementing the number of features. Examples would include an online component where users can communicate with others on other devices with online leaderboards. However, I believe that focusing on a few core features and executing them to the best of my ability is a better path to success. As in the future, either me with more time or a bigger team for development can maintain and evolve the solution with already established fundamentals.

Why Computational Methods are Suitable for this Project

The purpose of this solution is to develop the Typing Skills of the user. This skill involves using a computer to type on and will benefit the user's handling and efficiency when on the computer. Without a computer and keyboard, a user would not be able to type in normal, everyday activities. So, it makes sense that this solution should be solved using computational methods.

In more depth, the ease of use by having the program time and regulating the user's errors is very significant in accurately returning a WPM and Accuracy result to the user. This can be calculated rapidly and can be shown for the user's benefit for reflection and information. Rather than the user having to individually calculate each result themselves. This would be a slow monotonous process of counting characters the user has typed correctly. Additionally, this would take time away on the main purpose of the solution being to help develop the users touch typing skill.

A login system will need to have the credentials of each user stored securely and accessed immediately for the user's entry. This can be done computationally, with an external database file that can be secure and encrypted, rather than writing down credentials physically on a piece of paper. A piece of paper would not be as safe as a database as it involves a user reading all the login details of everyone to then compare against. Meaning they would see everyone else's details. This is not a security safe method. Therefore it is correct that this aspect would be solved with computational methods.

Additionally, computational methods can enter and manipulate the data within the database for the user's interests quickly and easily. Especially with leaderboards, noting each entry one by one after each attempt at touch typing would be mundane and restrictive for the user. Also, then to sift through a large number of entries to then find the 'Max WPM', for example, would be slow and uninteresting.

Leaving users unmotivated to enquire about their results, and then they would not reflect and be informed on their progress or others.

Computational methods can solve the number of extracts needed for the user to practise randomly by storing them on external text files where the program can randomly choose one. If computational methods did not solve this, the user would have to have several extracts written down. Having the user select an extract for practice would not be as helpful for the learner to try a new extract blindly. Another point is that it would take significantly longer for a user to sift through hundreds of extracts to choose from rather than a computer loading one onto the screen in a split second.

Due to the culmination of reasons for using computational thinking, I believe a faster alternative than paper copy and the reality that this solution cannot be solved without a computational method. There is no other viable way to answer this solution.

System Requirements

For me to develop this project, I will need a laptop and an IDE. I have chosen to have Visual Basic installed because I can specify Workloads, Components and Language Packs to accommodate my needs. Due to this, Visual Studio has downloaded Visual Basic for me. Additionally, the support of linking the program to a Microsoft Access database is advantageous in comparison to other alternatives. It will save me time to then focus on different aspects of my project. I will need Microsoft Office to be able to create and develop Microsoft Access databases.

Hardware Requirements

- Operating System: Windows 10 & 32 or 64-bit version
- Peripherals: Monitor, Keyboard & Mouse
- CPU: Minimal requirements needed therefore an Intel Core i3 or AMD Ryzen 3
- RAM: At least 1GB
- Storage Space: HDD or SSD (Size: Around 5mb)
- Motherboard

Language Comparisons

Here below was my thought process when choosing a language to develop in.

Language	Features	Using?	Reason Why?
HTML	● Used with CSS and JavaScript to create and develop dynamics webforms ● Easy to maintain the code ● Supported on every mainstream web browser, therefore is very stable and compatible	No	Only able to create web forms and websites used on the internet. This assumes that my project will need an internet connection, which is not the path I wanted to take with my offline-based project.
C#	● Is an object orientated language, this helps it have a high level of abstraction ● Ability of being cross platform (Windows/Macs/Linux) ● Automatically manages memory leaks using a garbage collector	No	Even though I can create Windows Forms with as much ease as other languages, I do not have a lot of knowledge and experience with this language. It would take time to learn this language and would risk missing deadlines for my project.

	(unlike C++) ● The ability of linking Microsoft Access with the program and manipulating data		
Python	● Is a functional programming language ● Interpreted language, therefore can be debugged easily and more quickly ● Ability of linking to SQL databases and can manipulate data easier	No	Hard to develop a graphical interface with a user. Instead of focusing on the features, I would spend too much time developing the GUI.
VB.NET	●Easy to develop GUI interfaces ● Is an Event Driven program ● Can be used cross-platform depending on framework ● The ability of linking Microsoft Access with the program and manipulating data	Yes	I am the most familiar with this language, so I do not need to allocate time to learn the language. The easy GUI creation will be suitable for making my program and the linkage with Microsoft Access database

Success Criteria

● ① **User can successfully login with a specified username and password? Does the program show an error if there is unsuccessful login attempt? Does the program exit if attempts exceed 3 fails?**

The user needs to be able to log in to the program to access the program then. The credentials that the user input is checked against are stored in an external database. They need to be the same and case-sensitive. It is case-sensitive as the set of possible character passwords increases tremendously, making it harder for malicious users to access an account illegally.

If there is an unsuccessful login attempt, then show a prompt to the user. This is because a random person should not be able to access another user's login. When the user has an unsuccessful login, a prompt will show how many more attempts the program allows. If not allowed anymore, then the program closes. This is to stop a Brute-Force attack on users' accounts, where there would be a continuous stream of login attempts of thousands of different password combinations.

If there is an error in the database connection, a prompt will come up to inform the user. This is so that the user is informed on the problem and how they can fix it.

● ② **A new user can input their credentials, and it is successfully added to the database? Does the program validation the inputs e.g., no input in a textbox? Does the program show an error when a username is taken?**

A new user needs to have the ability to add themselves to the database for future use.

They will note down their Name, Username and Password. The password will be entered twice for validation purposes and prompt the user if the passwords do not match. This is to make sure that the user has not accidentally inputted their password incorrectly and to make sure they are happy that the credentials they are putting in are correct.

The whitespace will be stripped from the variable to avoid errors when adding to the database.

All input textboxes will be checked for text and not just whitespace, so credentials are inputted into the database correctly.

If an error occurs when adding the user to the database, a prompt will inform the user. This is so that the user is informed on the problem and how they can fix it.

- ③ **Is there a guest mode for the user which doesn't add entries to the database?**

If a user does not want to use or have a login but still wants to use the program, so that they could do that. Without affecting other user's entries. This is focused towards one-time users of the program on a specific device. It is also in case of the database connection to the Access file not being present, as users' will not be allowed to log in. This will give users a method of still being able to access the main part of the program.

- ④ **After a successful login, does the main window open as the first form?**

The main window should be the Typing Test, as it is the primary purpose of the program. Otherwise, it might be confusing or disorientating for first-time users.

- ⑤ **Can the user navigate to the different forms within the program with ease? And these forms show up on the users request?**

So that the user can utilise every feature of the program with ease. Otherwise, it can be frustrating and confusing for users to try and navigate the program with a poor interface.

- ⑥ **Can the user choose the type of extract they will be tested on e.g., Paragraphs, Random Words & Alphabet (only for Word Fade)?**

The user should have the ability to choose what type of extract they would like to practise. This is because they can tailor their experience to their own skill level and desire on how they would like to use the program that session.

- ⑦ **Can the user choose to eliminate punctuation and/or capitals from the extract?**

This has a similar justification to the previous criterion. Users with a lower skill level might not be at a point where they would like to practise with Capital letters or Punctuation and just would like to focus on lower-case letters at that moment. This focused towards people who would like to tailor the experience to their needs in a more specific way.

- ⑧ **Can the user choose the time duration of the typing test, and whether it is a Countdown or Count Up?**

Depending on skill-level or preference, users might want to change the duration of each test. This could be down to the users' stamina in typing skills and how long they can type as fast as possible.

Countdown and count up is a preference for the user, as some users might find it easier to judge the amount of time left when going up or down. This added customisation can help aid the user is having the most comfortable experience possible.

- ⑨ **When a new typing test has started, a new randomised extract is loaded into the program depending on the options chosen above?**

A new extract must be loaded into the program for each new Typing Test. Whilst also taking into account the options of the type of extract chosen by the user. The user can find it tedious to choose each extract they would like to do individually. It leads to distortion of results (WPM and Accuracy) if the user has already seen the extract before.

This level of automation makes the user experience straightforward and less mundane.

- ⑩ **After each typing test, is the WPM and Accuracy calculated shown to the user and inserted into the database as a new entry?**

The program will calculate the WPM, and Accuracy will be calculated after each test. This will be shown to the user and will be inputted as an entry into the database,

Having the results quickly calculated and shown helps the user be informed on their current progress and what they need to work on.

- ⑪ **Can the user choose to do a Typing Test or WordFade?**

Several activities can give the user more entertainment and ways to practise touch typing in a fun and innovative way. The user can choose the activity within the program they would like to do at that current moment. This increases the chances of the user to carry on practising touch typing and getting better at it.

- ⑫ **Is there an option for Sudden Death mode?**

There will be a toggle option when doing a Typing Test where if Sudden Death is on, the Typing Test will be stopped if a mistake is made. This accommodates higher-skilled users at touch typing who are more focused on practising their accuracy, so they have more avenues to practise their skills to the highest level possible.

- ⑬ **Is there a Leaderboard that the user can access and see the top scores in order from other users in the database, and the user can choose from?**

There will be different leaderboards that the user can access, which compares all the users against each other. They are then ranked depending on the criteria set and shown to the user.

The leaderboards will be ‘Max WPM’, ‘Max Accuracy’, ‘Average WPM’, ‘Average Accuracy’.

The leaderboard shown will be chosen by the user. This will encourage a competitive aspect for users and give the user another feature to keep practising touch typing. Additionally, it will help motivate them to get better at this skill and be better than other users.

- ⑭ **Is there a form where the user can see each users' entries, have the ability to search through those entries? Can this form only be accessed through Admin Privileges where the program checks for these privileges?**

For each login, the user can see a table of all the entries associated with that user and search through the entries based on the date specified by the user. This is to help see the user's progress in further detail and check for any errors in the database.

Users only with Admin Privileges will be able to access this section of the program. This is because the sensitive information is shown on each user, and without an Admin Privileges feature, any user can access information on others.

- ⑮ **Can users change the font type, style and size used in the Typing Test/Word? Do they have the option for the font of the program to be in the form of 'OpenDyslexic'?**

For users who might certain fonts/styles hard to read or too small to read. They will have the option to modify it.

For users who have Dyslexia but still would like to practise Touch typing, there will be an option to change the program's font to 'OpenDyslexic'. This font is shown to be more accommodating for users with Dyslexia. I have chosen to add this feature as I have not seen it implemented in the existing solutions I have researched. And I want to have everyone be accommodated and welcome when using my program.

- ⑯ **Can the user see a graph of WPM / Date and Accuracy / Date and see their progress throughout their time using the program?**

The user should see a graph of WPM against Date to show the progression of their skill in the program. The user will be able to choose whether the axis is WPM or Accuracy against Time. This will help inform the user on their progress and their weaknesses and help them reflect and improve.

- ⑰ **Can the user see a keyboard layout on the screen, with a specific key highlighted when that character needs to be pressed next? Can the user turn this off?**

The user should see the QWERTY keyboard layout beneath the extract, and it will highlight the key that is needed to be pressed next. This will help the user not develop bad habits of looking down when stuck on a character/word.

To accommodate higher skilled players, there will be an option to turn off the character's highlighting and hide the keyboard graphic entirely. This is because it can be distracting and not useful for some people or that they want to challenge themselves.

- ⑱ **Do users have the ability to modify and delete their own accounts?**

They will have the ability to modify their credentials. This is for users who might need to change their own password or other credentials.

This is for users who know they will not need an account anymore. They will have the ability to delete their own account. This is also useful for database clean-up as the program will not have useless data in it anymore.

However, before the user deletes the login, a prompt will give the user the last chance to save the account. This is for users who might have accidentally pressed the 'Delete' button by accident.

- ⑲ **Can users with Admin Privileges edit and delete any other user? Can they give other user Admin Privileges too?**

They will have the power to modify any account. This is so that if any user is abusing the program in any way. There is a method of deleting the user's account. Additionally, the ability of giving other users Admin Privileges lets the primary Admin have the ability to delete their account without having the admin section of the program inaccessible to anyone.

- ⑳ **Do the users have a way to download Microsoft Access Runtime to then have access to the database?**

This is for people who do not have Microsoft Access installed on their computer but still want to use the features related to the database. I have included this as some users will not know to download it.

The Method of Accomplishing the Success Criteria

No.	How will this be achieved?
①	Create a login form. The login credentials will be stored on an external database (Microsoft Access) and will be checked from. When checking the input, the input will be trimmed of Whitespaces so that there are not any errors from spaces. The form will count the amount unsuccessful login entries. Validation checks to see whether the login credentials are correct, if correct then show the main program.
②	Create a Add User Form. Create a button on the login form that links with it. Validation that the username is not already taken.
③	Have the main program load up, without the use of Statistics or Admin modes available. Entries from typing tests will not be added to the database.
④	Create and link the main form 'Typing Test' with the login form.
⑤	Add a menu bar at the top of the form with buttons for each form and link them with it.
⑥	Create an Options form and add a ComboBox of the different choices of extracts. Depending on the choice, a variable declared in the login form will be used to pass the value into the typing form. When starting the typing test the form will check the value of the variable and act accordingly.
⑦	In the Options form, add a Checkbox for the punctuation and Capital Letters and depending on the checkbox being ticked. Two different variables declared in the login form will be used to pass the choices through to the Typing Test. The punctuation will be stripped of if True, and the extract will be change to all lower case if 'Capital Letters?' is False

⑧	In the Options form, add a ComboBox of the different durations. Depending on the choice, a variable declared in the login form will be used to pass the duration into the Typing form
⑨	The Typing Form will check the variables passed from the Login form and will act accordingly. Depending on the extract choice, the extracts will be stored on an external text file and it will be loaded into the program and an extract will be randomly chosen from it.
⑩	The Typing Test will count the amount of characters typed overall and calculate WPM by ((countCharacters/5) / (TimeDuration/60)). It will count the overall faults/ errors and calculated accuracy by (totalFaults/countCharacters)
⑪	There will be different forms and buttons for Typing Test and Word fade in the navigation bar.
⑫	It will be a checkbox in the Options form. Depending on the choice, a variable declared in the login form will be used to pass the results into the Typing form. If true then a fault will end the test.
⑬	Create a form for statistics, add a DataGridVeiwer. Load the database in a dataset and manipulate the dataset with SQL to show the different leaderboards. The choice for leaderboards will be via a ComboBox.
⑭	Create a form for Admin mode, add a DataGridVeiwer. Load the database in a dataset and manipulate the dataset with SQL to show the different user entries. Add two DateTimePickers, for start and end date and manipulate the dataset with SQL accordingly
⑮	In the Options form, add a font dialog where the user can modify the font type/style/size
⑯	Add two Chart controls in the Statistics form, where on loading of the form, the charts are linked to a dataset containing the data which are then present of line Graphs
⑰	The keyboard will be made up of keys that are individual panels. The program will check the character needed to be typed next and will highlight the corresponding panel (that represents a key on a keyboard), Additionally, there will be a ComboBox for the different options in the Options form for 'Shown and Highlighted', 'Shown and Not Highlighted' and 'Not Shown'.
⑱	The credentials will be shown to the user. And they will have the ability to edit it. To save the edits, there will be a button to confirm the edit and change it in the database. There will be a button to delete.
⑲	The credentials will be shown to the user. To change the user being shown, there will be a button to go to the next user. And they will have the ability to edit it. To save the edits, there will be a button to confirm the edit and change it in the database. There will be a button to delete.
⑳	In the 'Options' form, add a LinkLabel where when the user clicks on it. The Microsoft webform for downloading Microsoft Access Runtime will come up. (this link)

Design

Systemic Breakdown of my Solution

I have broken up my project into smaller sections to ensure my project's successful and organised design. This will be broken down into forms, where I will display a brief diagram of my vision of the form's layout, its purpose, any validation needed for the program.

Validation for All Forms apart from Login and Add User Forms

When navigating to the Admin Section of the program via the Menu bar, the program will check whether the user has admin privileges. If not, a prompt will come up saying, 'You do not have access to this'. This is needed as a random user should not be allowed access to actions that only an admin can do.

When navigating to the Statistics form, the program will check how many entries that user has. If the user does not have at least one entry, then the user will not be allowed to access this form. This is because the Statistics form would rely on the existence of entries to then interpret the data for the user. Without entries, the form is useless. Because of this, the Statistics form will also be unavailable to Guest Users.

Login Form

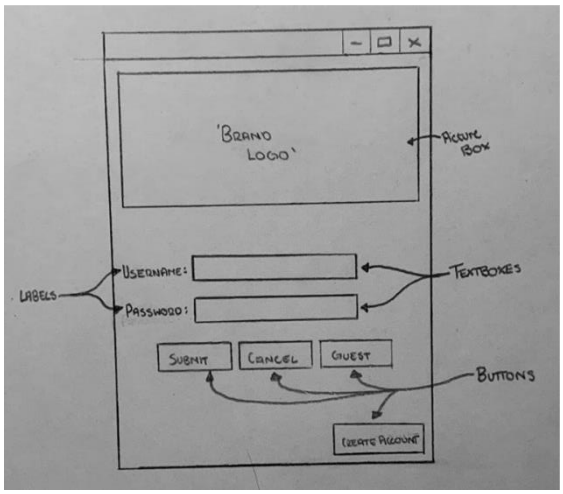
The purpose of a Login Form is to check the users' credentials who want to use this program and act accordingly.

The 'Submit' button is to enter the inputted login details and check against the database.

The 'Cancel' button is to let the user exit the program.

The 'Guest' button lets the user access the main program without a login and will not be tied to a specific login.

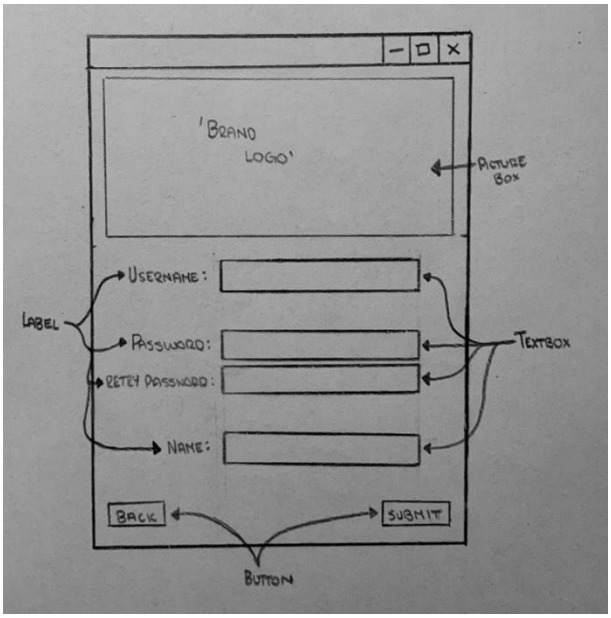
The 'Create Account' button is to open the 'Add User Form' so the user can add login details for them.



Validation

When the 'Submit' button is pressed the program checks that there is an input in both Textboxes. If not, an error message is displayed. Additionally, the input is trimmed of whitespace, so spaces in the text are not passed into the check of the database credentials. The Textboxes are limited to a maximum of 20 characters, so the user does not forget their details. And to a minimum of at least 5 characters so the user can have a strong password. There will only be 3 login attempts allowed before the login form closes so that it cannot be brute forced. There will also be prompts if the program cannot connect to the database or the SQL query of fetching the login details cannot be done.

Add User Form



The purpose of an Add User Form is to give new users an opportunity to have a login account so they can access statistics.

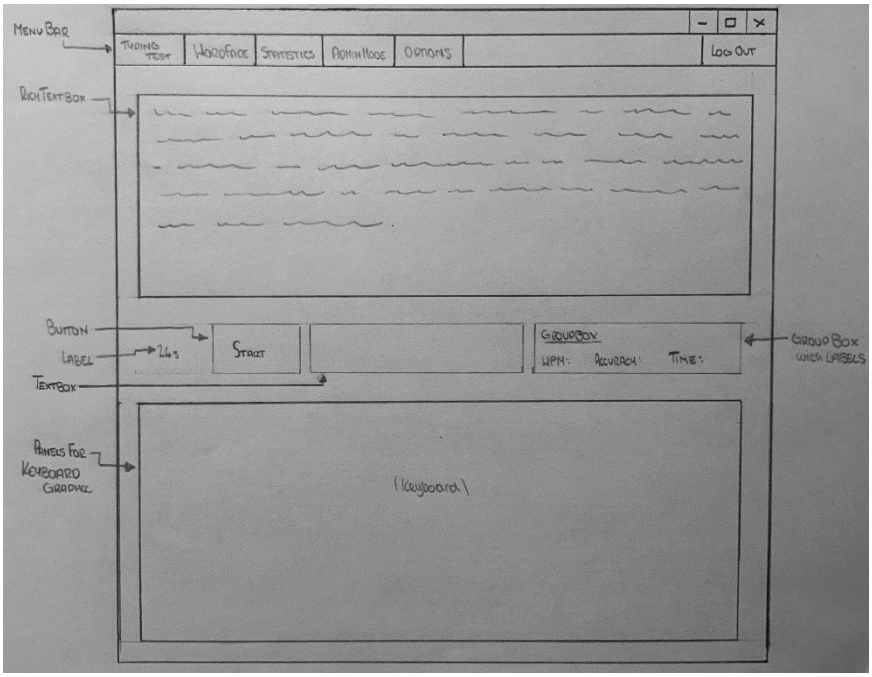
The 'Submit' button is to add the inputted credentials from the textboxes into the database.

The 'Back' button is so that a user who may have accidentally pressed the 'Create Account' button from the previous form can then go back.

Validation

The Textboxes are limited to a maximum of 20 characters so that there is an input in both Textboxes. If not, an error message is displayed. Additionally, the input is trimmed of whitespace so there are no spaces in the text. The user needs to input their password twice and will be checked so that the user does not accidentally put their password incorrectly. Then a prompt will come up saying, "Are you sure?" If yes, credentials are added. There will also be prompts if the program cannot connect to the database or the SQL query of inserting the login details cannot be done.

Typing Test Form



The purpose of this form is to test the User on a random paragraph and calculate their WPM and Accuracy from that test.

The buttons on the Menu Bar are for navigation to the different forms. The 'Log Out' button, returns the user back to the starting form and logs the user out.

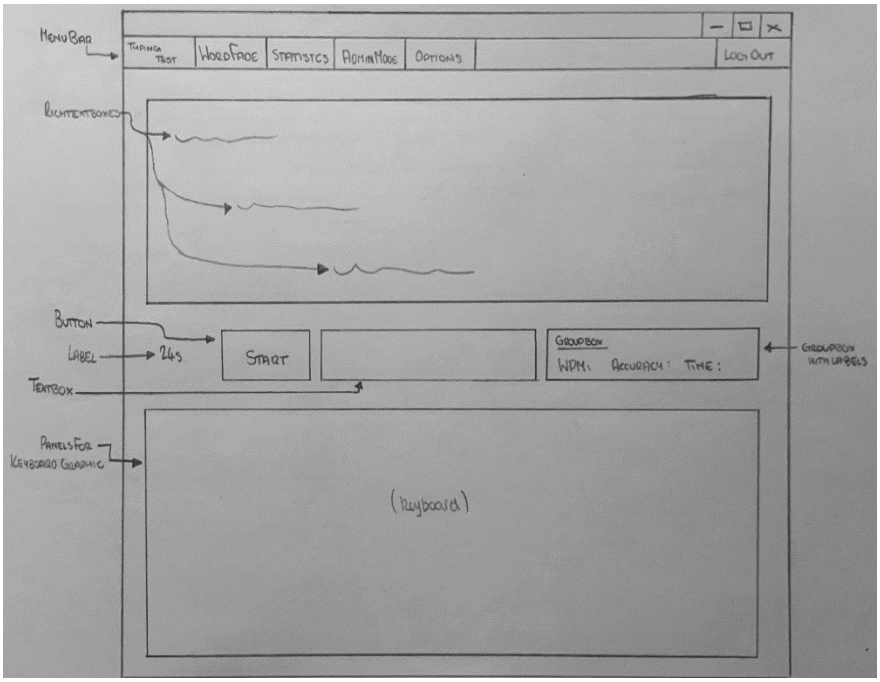
When the 'Start' button is pressed, an extract will be loaded into the form and the timer will countdown for 3 seconds and then start the Typing Test. The button's text will now be 'Stop' where if pressed in the middle of the Typing Test, the test will be stopped. If the user finishes typing the extract or the timer has been reached, WPM and Accuracy is calculated, and an entry is inserted into the database if not in Guest Mode.

When a Typing Test is in progress, depending on the user's preference, the keyboard will highlight the next character to be pressed, not highlight the next character needed to be pressed but the keyboard is still visible, or the keyboard is not visible at all.

Validation

There will also be prompts if the program cannot connect to the database or the SQL query of inserting the entry details cannot be done. Additionally, an entry will not be made if the user is in Guest Mode and not logged into an account.

Word Fade Form



The purpose of this form is to test the User on a random paragraph and calculate their WPM and Accuracy from that test.

The buttons on the Menu Bar are for navigation to the different forms. The 'Log Out' button, returns the user back to the starting form and logs the user out.

When the 'Start' button is pressed, an extract and the first three words will be loaded into the form and the timer will countdown for 3 seconds and then start the WordFade test. The button's text will now be 'Stop' where if pressed in the middle of the WordFade test, the test will be stopped. When the user types the first word, the word order shifts to add the next third word at the top and the tore two shift down into the main view. If the user finishes typing the extract or the timer has been reached, WPM and Accuracy is calculated, and an entry is inserted into the database if not in Guest Mode and if the extract is not Alphabet.

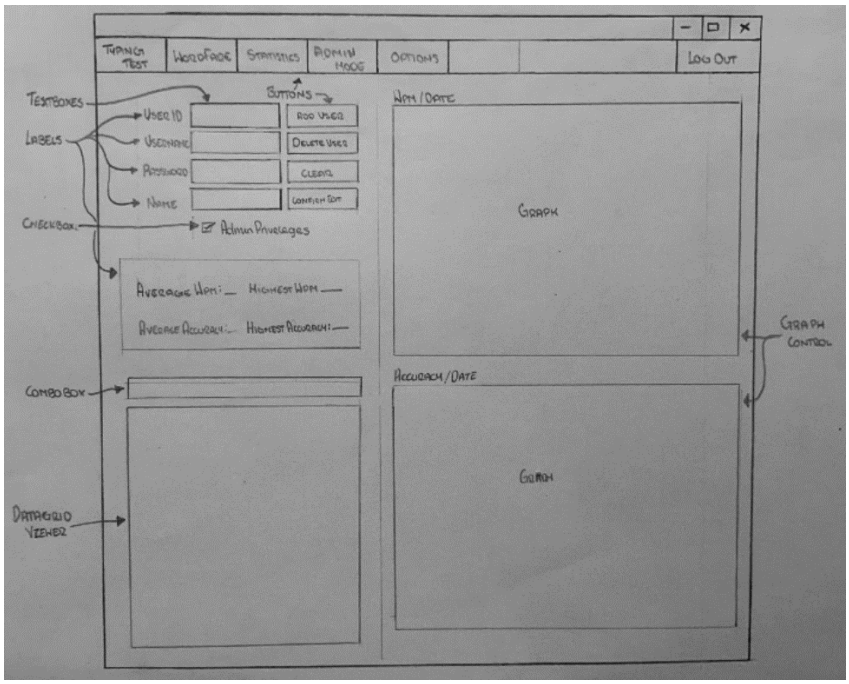
When a WordFade test is in progress, depending on the user's preference, the keyboard will highlight the next character to be pressed, not highlight the next character needed to be pressed but the keyboard is still visible, or the keyboard is not visible at all.

Validation

There will also be prompts if the program cannot connect to the database or the SQL query of inserting the entry details cannot be done. Additionally, an entry will not be made if the user is in Guest Mode and not logged into an account.

Statistics Form

The purpose of the 'Statistics' form is to inform users of their progress and the ability to edit their login details.



The buttons on the Menu Bar are for navigation to the different forms. The 'Log Out' button, returns the user back to the starting form and logs the user out.

When the 'Statistics' form loads, the login details for that user are loaded into the text boxes on the top left. Depending on whether the user has Admin Privileges, the Checkbox will either be ticked or not. The user can edit the text boxes and save the changes using the 'Confirm' button. However, they will not be able to edit the User ID (as it's used as a Primary Key) and whether they have Admin Privileges.

The 'Clear' button clears everything in the textboxes, apart from userIDtextbox and adminCheckbox for easier editing of their credentials.

The 'Add user' button, changes the Textbox use to now be able to input credentials for a new user. All textboxes will be cleared and inputs for a new user can be made and apart from UserID and Admin Privileges. When the 'Add User' button is pressed, it will change to 'Cancel Add User' where the user can resort back to seeing their own credentials. Instead of that if the user presses the 'Confirm' button when in Add User mode. The user will be added to the database.

The 4 located on the middle left, will be used to inform the user of their 'Highest WPM, 'Highest Accuracy', 'Average Accuracy', 'Average WPM'. This will be calculated by entries from the database connected to the program.

The DataGridViewer will show a leaderboard of every user. The types of leaderboard will include 'Most Entries', 'Highest WPM', 'Highest Accuracy', 'Average WPM', 'Average Accuracy'. This is to add competitiveness between the users. The type of leaderboard shown will be chosen using the ComboBox above it.

The two Graph controls situated on the right of the form will each show data from the entries made by the user logged in. It will take the average entry of each day and plot a line graph for Average WPM/Date and Average Accuracy/Date. This is to visualise the progression of the users time with the solution and their progress.

Validation

When editing your own credentials or making a new account, all whitespaces will be trimmed. Additionally, if textbox has no input and the user presses 'Confirm' to save it. This will display a

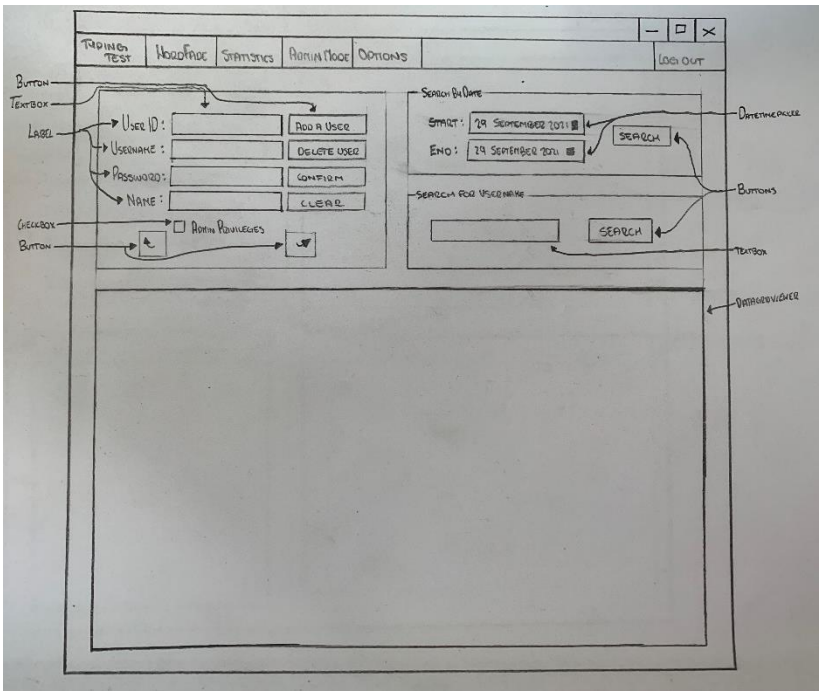
MessageBox error and will not save into the database. If the connection to the database does not work then an error will show.

Admin Form

The purpose of the 'Admin' form is to inform users with admin privileges of specific user's entries and the ability to edit any user's login details.

The buttons on the Menu Bar are for navigation to the different forms. The 'Log Out' button, returns the user back to the starting form and logs the user out.

When the 'Admin' form loads, the login details for the first user in the database is loaded into the text boxes on the top left. Depending on whether the user has Admin Privileges, the Checkbox will either be ticked or not. The user can edit the text boxes and save the changes using the 'Confirm' button. However, they will not be able to edit the User ID (as it's used as a Primary Key) but can determine whether they have Admin Privileges.



To cycle between different user, there is a '→' and a '←' button located under the login credentials.

The 'Clear' button clears everything in the textboxes, apart from userIDtextbox and adminCheckbox for easier editing of their credentials.

The 'Add user' button, changes the Textbox use to now be able to input credentials for a new user, they can also determine whether the new user will have Admin Privileges. All textboxes will be cleared and inputs for a new user can be made and apart from UserID. When the 'Add User' button is pressed, it will change to 'Cancel Add User' where the user can resort back to seeing their own credentials. Instead of that if the user presses the 'Confirm' button when in Add User mode. The user will be added to the database.

The DataGridViewer will show each individual entry for the user loaded into the textboxes. They will not be able to edit/delete any of them. This is for a more informative look into the database for each user and to identify any errors made in the database.

There is a 'Search' button in the 'Search by Date' GroupBox where the user can specify the entries made between specific time periods using two DateTimePickers.

There is a 'Search button in the 'Search by Username' GroupBox, where the user can search for a username for that to be loaded into the form and DataGridVeiwer. The input for the username search is via the Textbox to the left of the button.

Validation

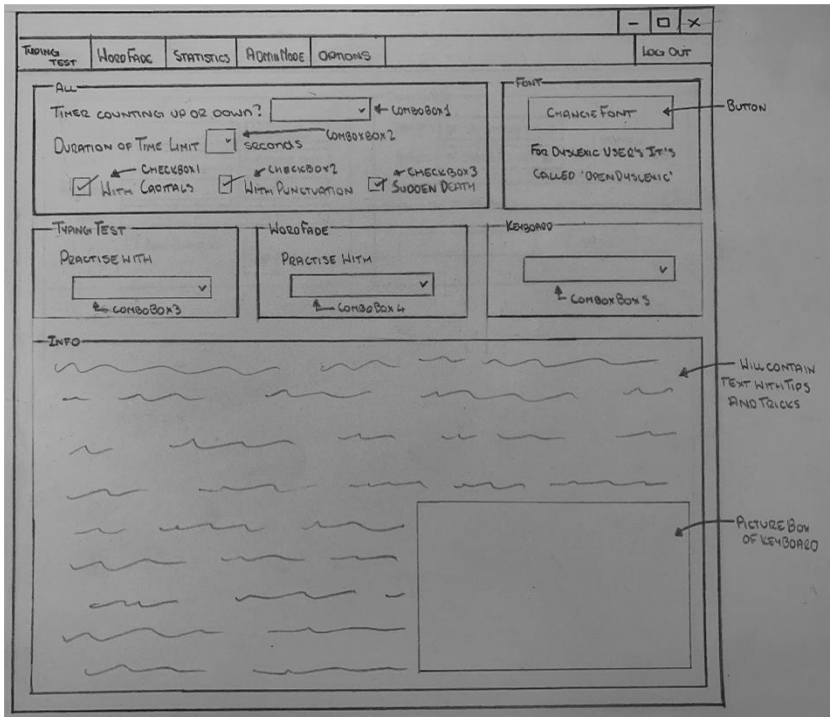
When interacting with the database, if the connection to the database does not work then an error will show. When editing your own credentials or making a new account, all whitespaces will be trimmed. Additionally, if textbox has no input and the user presses 'Confirm' to save it. This will display a MessageBox error and will not save into the database. If the search input for a username does not find anything, a message box shows saying that a user by this name was not found.

Options Form

The purpose of the 'Options' form is let the user tailor the experience of the Typing Test and WordFade to their own needs. This is because people learned and improve on a skill in different ways at different speeds.

The buttons on the Menu Bar are for navigation to the different forms. The 'Log Out' button, returns the user back to the starting form and logs the user out.

The first ComboBox is used to determine whether the timer counts down or counts up to the time limit.



The second ComboBox is used to determine the time duration a Typing Test or WordFade goes on for. The choices are 15, 30, 45 and 60 seconds

The third ComboBox is used to specify the type of extract used in Typing Test's. The choices are either full paragraphs or random words in a random order.

The fourth ComboBox is used to specify the type of extract used in WordFade. The choices are full paragraphs, random words or the alphabet.

The fifth ComboBox is used to specify whether the keyboard in Typing Test or WordFade is shown and highlights the next key, is shown and does not highlight the next key, or does not show the keyboard at all.

The first CheckBox is used to specify if the user wants capitals in the extracts they are tested against.

The second CheckBox is used to specify if the user wants the Typing Test or WordFade to stop after one mistake.

The third CheckBox is used to specify if the user wants punctuation in the extracts they are tested against.

The 'Change Font' button when clicked opens a FontDialog where the user can choose the type, style and size of font used in the Typing Test and WordFade.

The PictureBox will have an image of a keyboard.

The LinkLabel will be used for a hyperlink to access the webform to download Microsoft Access Runtime if they need to.

The rest of the text comprised as labels in the 'Information' GroupBox is used to give tips for the user

Validation
There is no validation needed in this section.

Structure of Development of my Project

I will start developing my project in the order of the forms shown in the previous section. I will prioritise the basic functionality in each form and make sure that a database can be linked with the form correctly. Additionally, I will make sure that I can load external text files into the program before adding anything further to the project.

Through each iteration of each form in my project, I will use a modular approach to create a new version of a form with an increase in functionality. This will mean systemically breaking the features down into smaller aspects, focusing on a specific aspect and solving how I will solve it computationally. This will help me be organised, focused and not overwhelmed by the scope of my project.

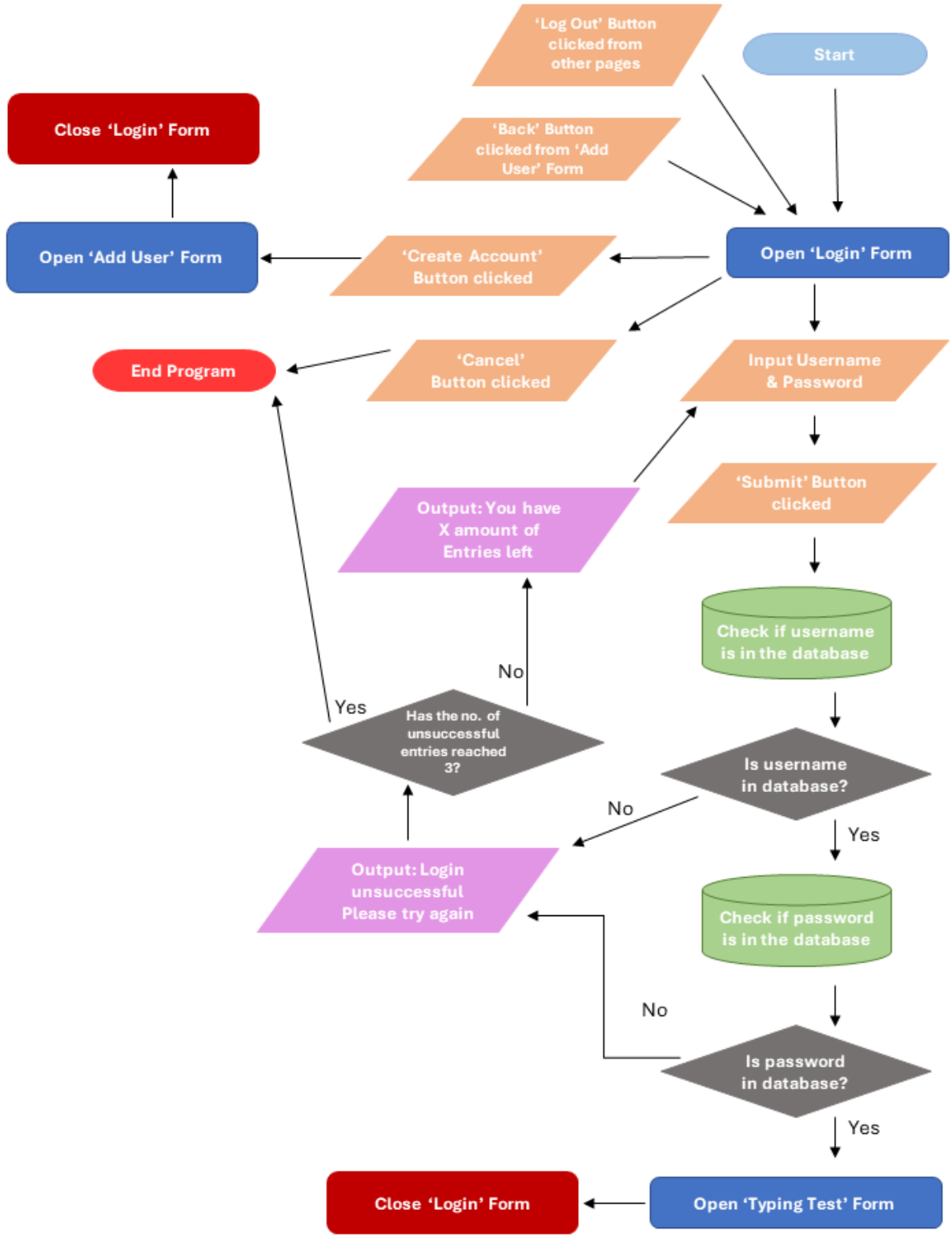
I will test each iteration thoroughly using the Test Data defined in the 'Test Data' section subsequent to this part. When I have an iteration of the project that meets all of the Test Data, I will move on to Success Criteria testing. At the end, I will do Post Development Testing on the final version of my project. By testing each iteration of my project, I will have an increased understanding of what I should focus on next to enhance the functionality of the subsequent iteration.

Additionally, I will focus on making my code as efficient and easy to read as possible. This will be reflected on in each iteration, where I will see where I can have my logic be more economical. I will also be aware of making the code as coherent as possible. This is because this code needs to be easy to maintain and will need to include comments explaining each branch of code for the developer to maintain the project. I will comment throughout the development of my project. However, I am aware of the time constraints that this project is under. So, some difficult parts of the project will inevitably be hard-coded, and I will not have the time to figure out how to make it more efficient. In the future maintenance of the project, the code will be revisited so that any mistakes and inefficient parts of my code will be addressed.

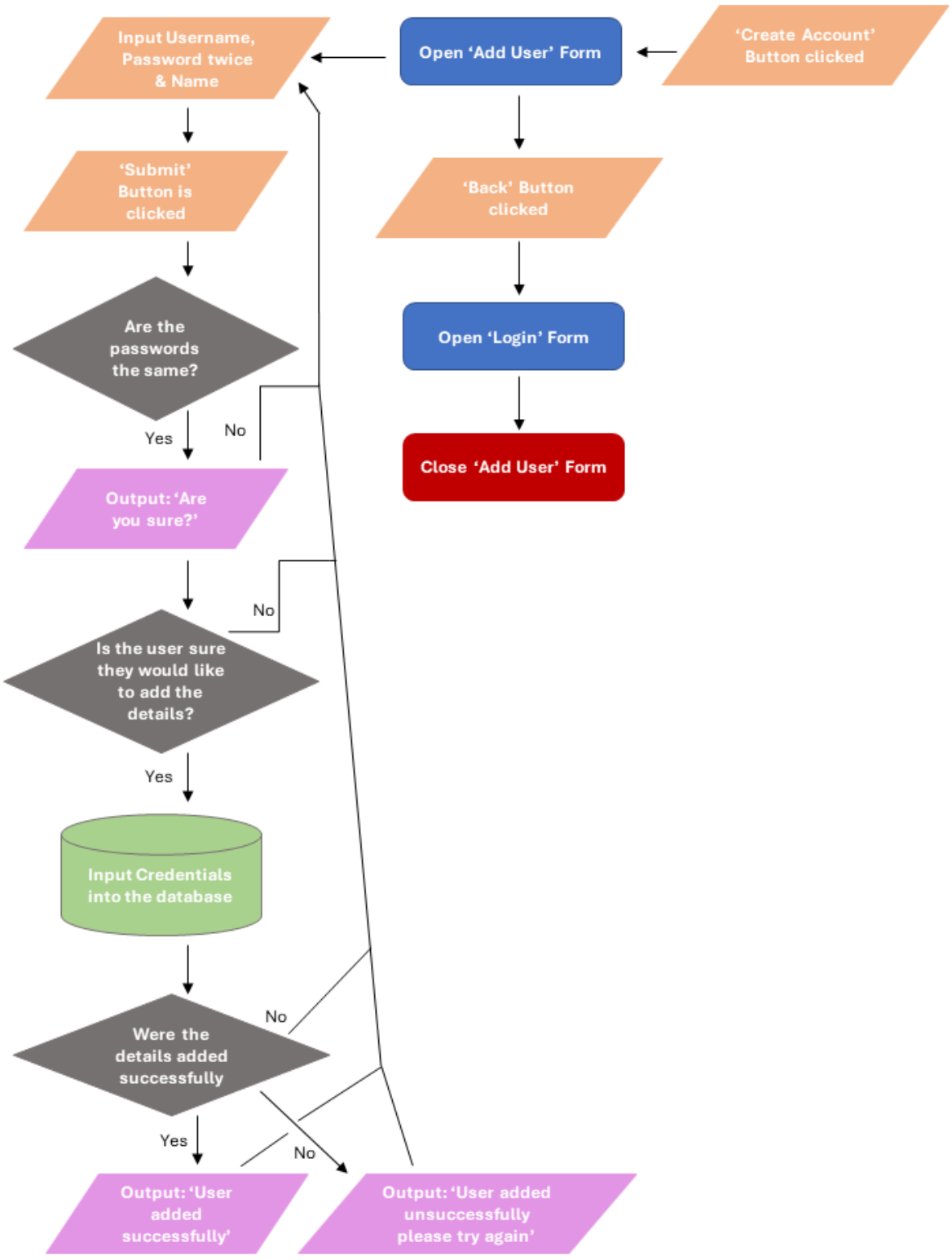
Algorithms

I have decided to break down the main program into the different forms and to present the algorithm as flow charts. The processes, features and validations have already been justified in the previous section.

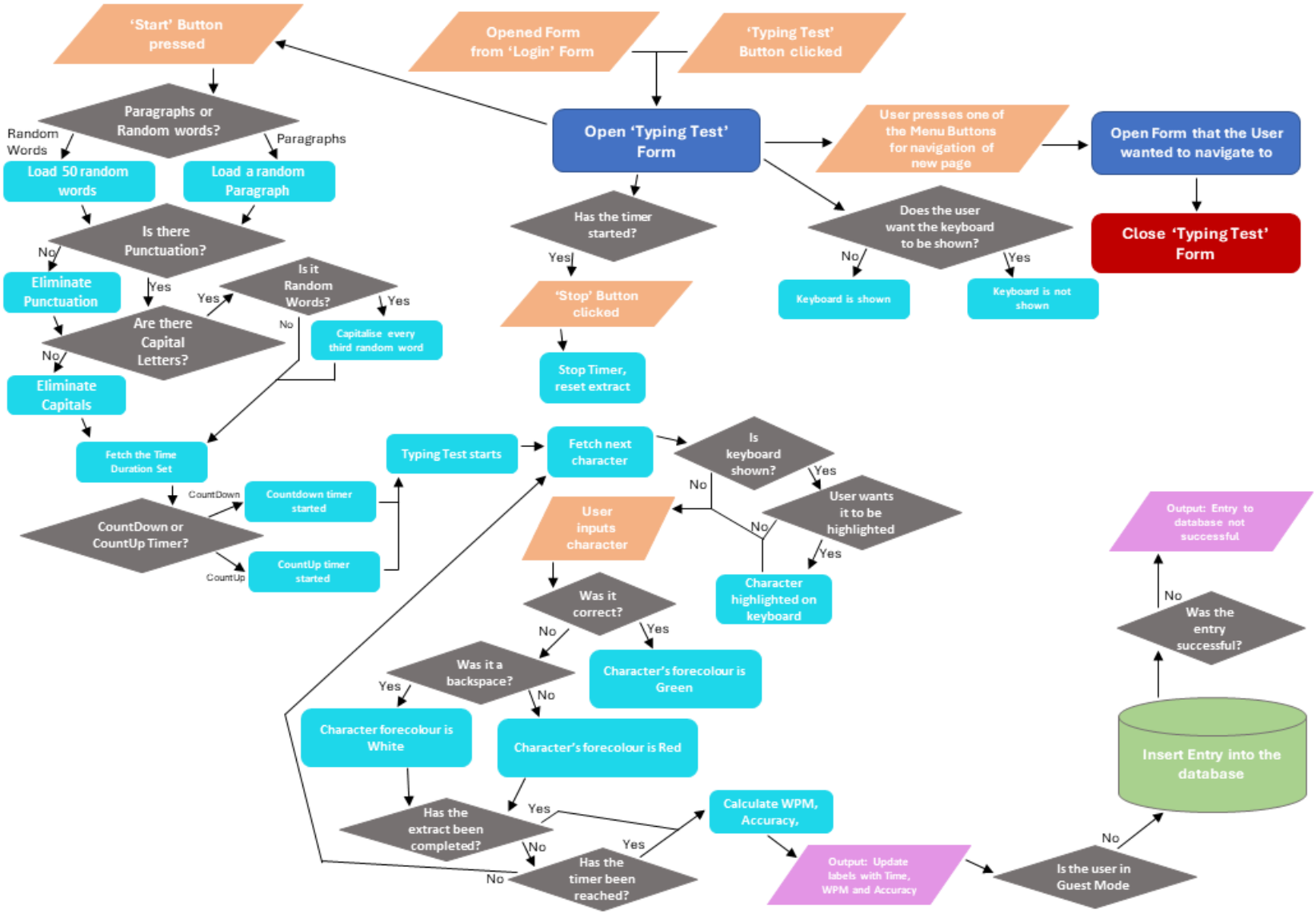
Login Form


'Log Out' Button clicked from other pages
Start
Close 'Login' Form
'Back' Button clicked from 'Add User' Form
Open 'Add User' Form
'Create Account' Button clicked
Open 'Login' Form
End Program
'Cancel' Button clicked
Input Username & Password
'Submit' Button clicked
Output: You have X amount of Entries left
Check if username is in the database
No
Yes
Has the no. of unsuccessful entries reached 3?
Is username in database?
No
Yes
Output: Login unsuccessful Please try again
Check if password is in the database
No
Is password in database?
Yes
Close 'Login' Form
Open 'Typing Test' Form

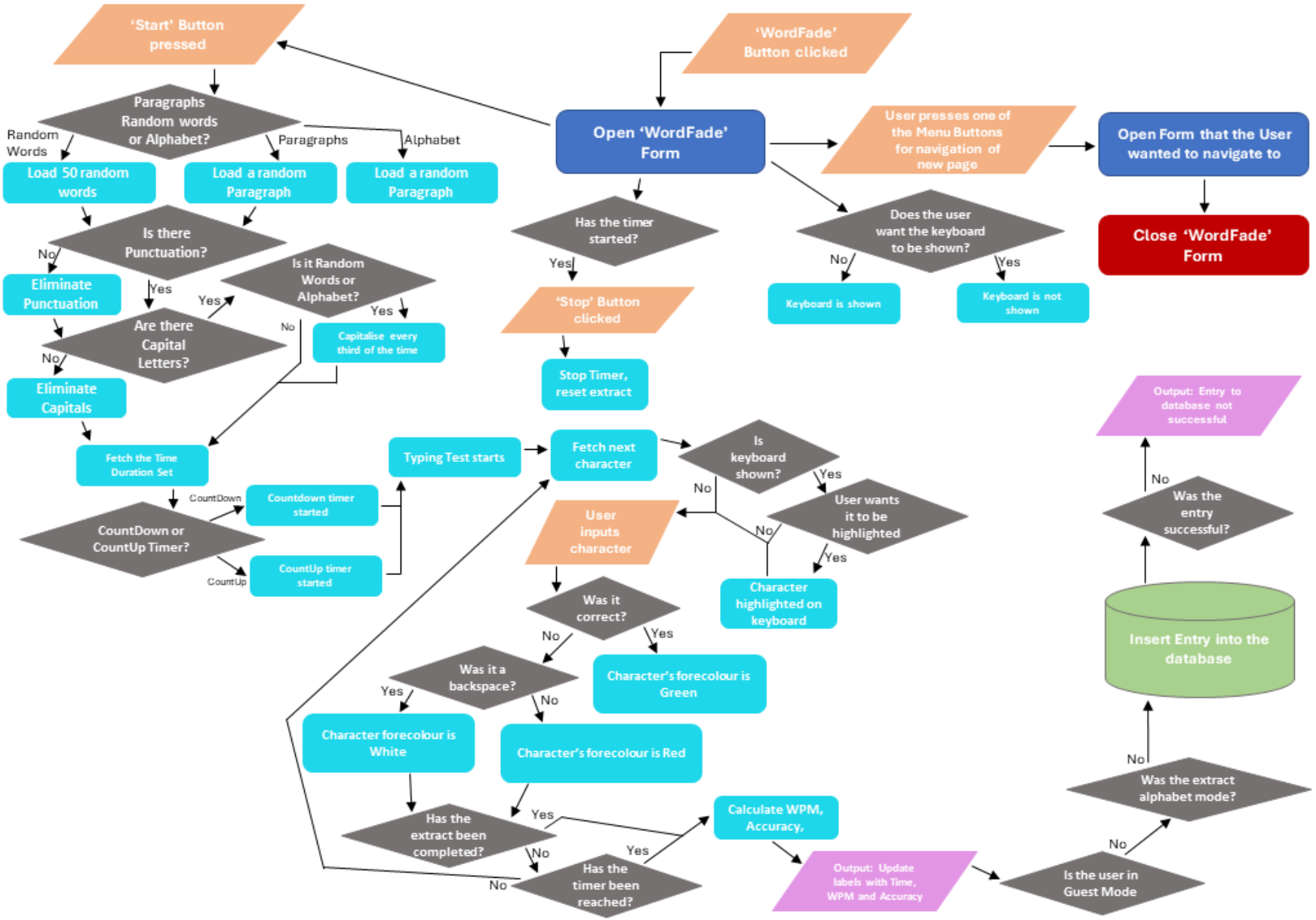
Add User Form



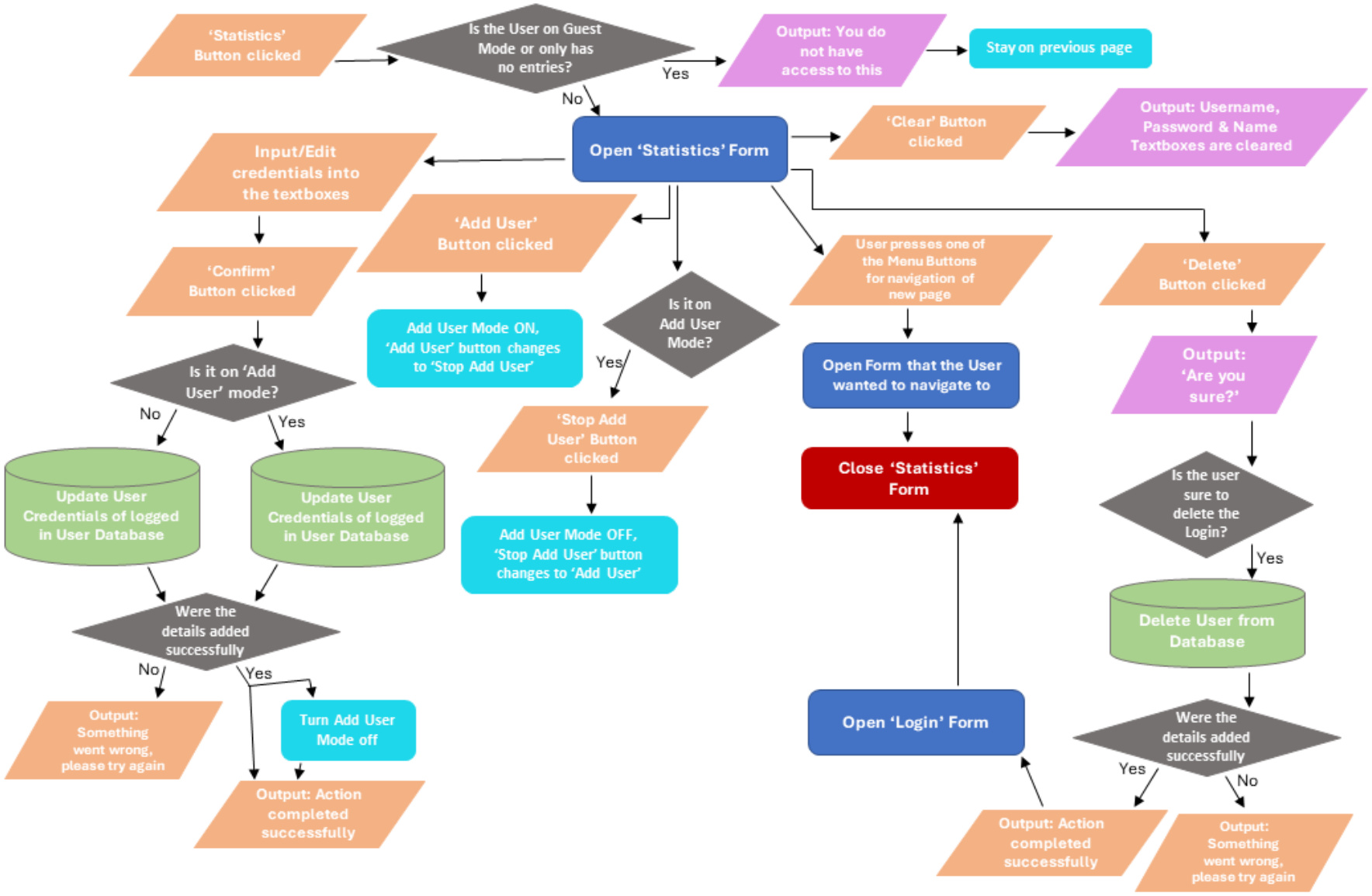
Typing Test Form



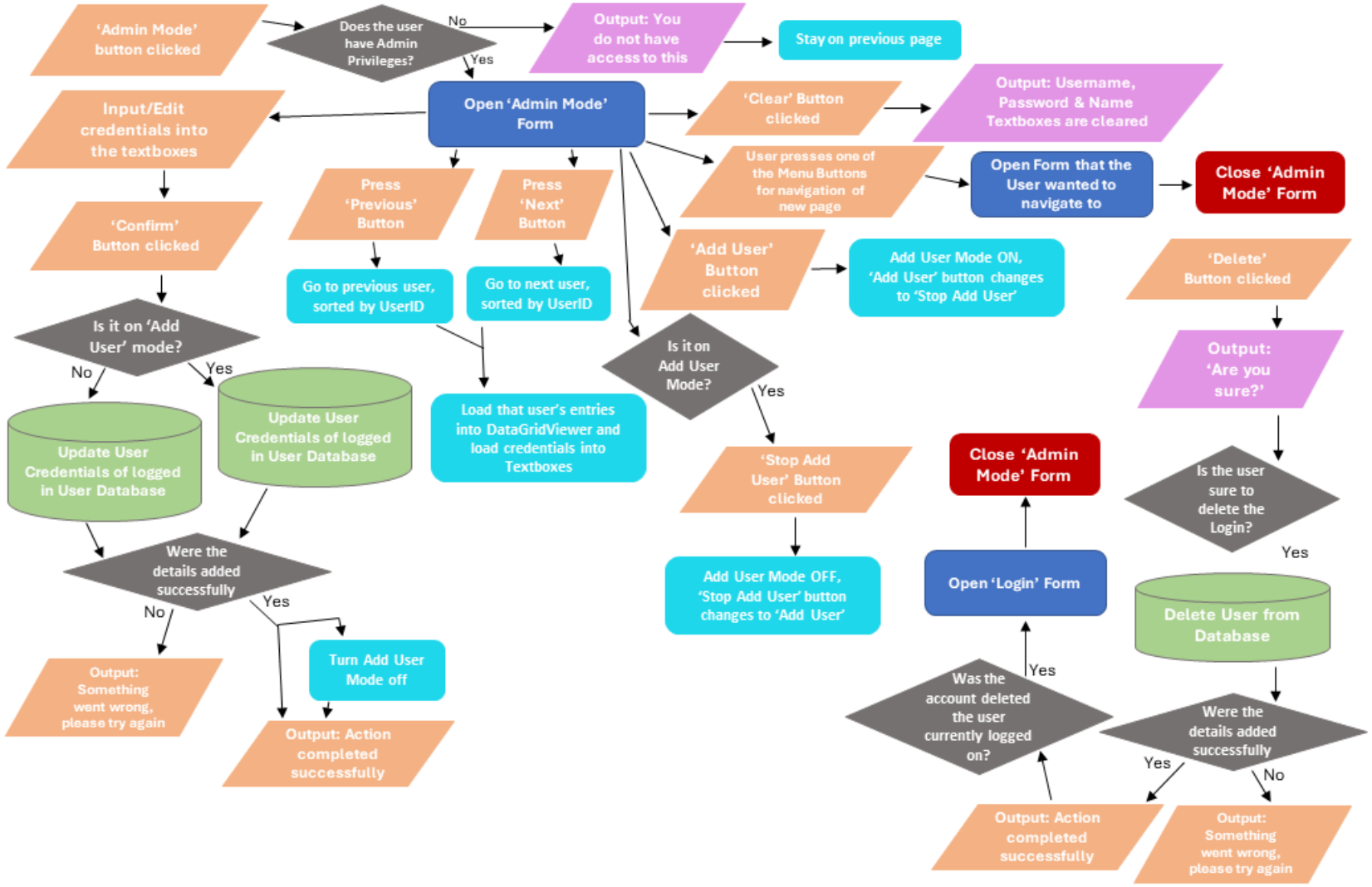
Word Fade Form



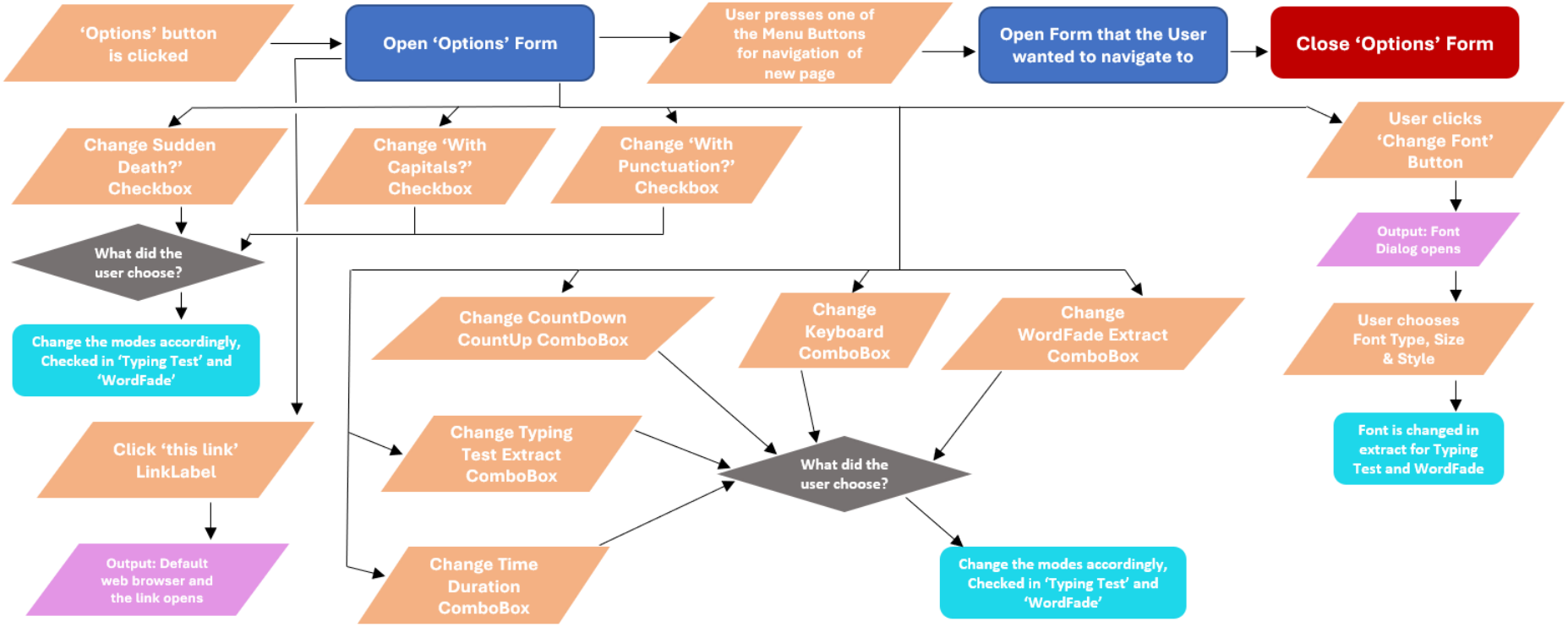
Statistics Form



Admin Form


'Admin Mode' button clicked
Does the user have Admin Privileges?
No
Yes
Output: You do not have access to this
Stay on previous page
Open 'Admin Mode' Form
Input/Edit credentials into the textboxes
'Confirm' Button clicked
Is it on 'Add User' mode?
No
Yes
Update User Credentials of logged in User Database
Update User Credentials of logged in User Database
Were the details added successfully
No
Yes
Output: Something went wrong, please try again
Turn Add User Mode off
Output: Action completed successfully
Press 'Previous' Button
Press 'Next' Button
Go to previous user, sorted by UserID
Go to next user, sorted by UserID
Load that user's entries into DataGridViewer and load credentials into Textboxes
'Clear' Button clicked
Output: Username, Password & Name Textboxes are cleared
User presses one of the Menu Buttons for navigation of new page
Open Form that the User wanted to navigate to
Close 'Admin Mode' Form
'Add User' Button clicked
Add User Mode ON, 'Add User' button changes to 'Stop Add User'
Is it on Add User Mode?
Yes
'Stop Add User' Button clicked
Add User Mode OFF, 'Stop Add User' button changes to 'Add User'
'Delete' Button clicked
Output: 'Are you sure?'
Is the user sure to delete the Login?
Yes
Delete User from Database
Were the details added successfully
Yes
No
Output: Something went wrong, please try again
Was the account deleted the user currently logged on?
Yes
Output: Action completed successfully
Open 'Login' Form
Close 'Admin Mode' Form

Options Form



Usability Features

Simple Design

This will make the solution easy to navigate and will less likely confuse users. I want a user of any age and the ability to use my solution with no trouble and with ease. Without this, they will not use my solution and will discard it for a more user-friendly experience.

Menu Panel

This will be so that all the different forms are easily accessible and all in one place throughout the different forms in my solution. This is also for easy navigation and to ease confusion. The justification for this feature is the same as for the previous one.

Big Buttons

Big buttons will also ease confusion and make the program easy to navigate as no button is minute enough to be missed.

Big Modifiable Font

A big font will let users with poorer eyesight be able to use this solution. This will be used in the extracts for 'Typing Test' and 'WordFade'. Having the option to modify the size of the font is also very useful for the user, as they can tailor the experience to their specific eyesight.

Dyslexic Font Option

This will let users with Dyslexia have the ability to use my solution. In some cases, it is hard to read fonts in a program and have an option for a Dyslexic font which would then be used through the solution, which would be very useful for them.

Other Options (Punctuation, Capital Letters etc.)

Each user will be different and will want to practise different aspects of touch typing, e.g. punctuation. A level of customisation will let users tailor the experience of practising their skills. Otherwise, each user would have to conform to the same method of practising, which would not be as helpful for them.

Guest Mode

Targeted to users that will use the solution maybe a couple of times on the device and would not like a login account. This is so the program is still accessible without the database attached (in case of an error or the database is not present) to it or a login account. However, the Statistics and Admin forms would not be accessed.

Information for User

This will be located in the Options form, where useful information and tips in regards to touch typing will be located. This will help the user understand some of the nuances and correct some of their techniques. Additionally, this will be where the user will be able to download Microsoft Access Runtime if needed and other problems relating to the function of the program.

Variables, Classes & Data Structures

Each class is form with a new form in the program. This is so when a form is opened, all the structures and variables are initialised with it.

Class	Purpose
loginForm	Where the application opens and closes from. Leading to either logging in or creating an account and passes variables between other classes. Validation required when logging in, checks inputs against the database credentials.
addUserForm	All the functionality when creating an account and inserting it into the database. Has validation of checking the inputs against rules of the database. E.g. at least 5 characters long but below 20 characters. Navigates back to login class after use.
typerForm	Main feature, where a user can perform a Typing Test
wordFadeForm	Similar to typerForm however, some data structures are different as there are different controls needed to be handled.
statUserForm	Validation needed when initialising the class, as a user's account needs to have at least 1 entry made in it. Its purpose is to show statistics for that user so they can reflect an be informed on their progress and a way to modify their own credentials.
statAdminForm	Its purpose is to provide a raw look into the database showing every user's entries and a way to modify other users credentials. Validation needed when initialising the class as the user's account needs to have Admin Privileges.
optionsForm	Purpose is to provide customisation to the user's experience when performing a Typing Test or WordFade. Also gives useful tips about Touch Typing to the user.

Form (class)	Variables	Purpose	Type
Login Form (loginForm)	connectionString	Holds the connection string to connect to the database.	String
	dataUsername	Holds the username of the logged in user, to be passed between other classes.	String
	dataName	Holds the name of the logged in user, to be passed between other classes.	String
	dataUserID	Holds the userID of the logged in user, to be passed between other classes.	String
	timeDuration	Option variable holding the setting for the Time Duration, passed from optionsForm to either typerForm or wordFadeForm	String
	countType	Option variable holding the setting for the countdown or up, passed from optionsForm to either typerForm or wordFadeForm	String
	suddenDeath	Option variable holding the setting for whether it's Sudden Death Mode, passed from optionsForm to either typerForm or wordFadeForm	Boolean
	extractType	Option variable holding the setting for the type of extract for the Typing Test form, passed from optionsForm to typerForm	String
	extractWordFadeType	Option variable holding the setting for the type of extract for the WordFade form, passed from optionsForm to wordFadeForm	String

	capitals	Option variable holding the setting for whether there are capitals, passed from optionsForm to either typerForm or wordFadeForm	Boolean
	Punctuation	Option variable holding the setting for whether there are punctuation, passed from optionsForm to either typerForm or wordFadeForm	Boolean
	showKeyboard	Option variable holding the setting for the keyboard, passed from optionsForm to either typerForm or wordFadeForm	String
	entriesPresent	Holds the value for whether the user has made at least one entry on the account. Used for validation when entering the Statistics Form	Boolean
	globalFont	Holds the globalFont, which can be changed in the optionsForm and can be passed into the typerForm and wordFadeForm	Font
	attemptCount	Holds the number of Attempts the user has done to log into an account	Integer
	red	Used as colours for the keyboard highlighting in both typerForm and wordFadeForm. As both classes use it, better to declare here and then pass into the classes when needed	Color
	lightGreen		
	lightBlue		
	purple		
	yellow		
	blue		
	orange		
	darkGreen		
Typing Test Form (typerForm)	extract	Holds the whole extract about to be tested on in a string	String
	words	Holds the whole extract about to be tested on as an array with an item being a word in the extract	String Array
	charList	Holds the current word being typed on as an array with an item being a character in the word	String Array
	countChar	Is a counter for the correct amount of letters typed for the current word, is reset when starting a new word	Integer
	countCharEssay	Is a counter for the correct amount of letters type for the whole extract. Is reset for the next test. Is used to calculate WPM.	Integer
	countWord	Is a counter for the correct amount of words the user has typed for the whole extract. Is reset for the next test.	Integer
	currentFaults	Is a counter for the current amount of faults/errors the user has.	Integer
	totalFaults	Is a counter for the total amount of faults/errors the user has accumulated. Is used to calculate Accuracy.	Integer
The WordFade Form (wordFadeForm) uses the same variables, however, are declared separate.			
Statistics Form (statUserF	addUser	Variable for whether the form is in Add a User mode	Boolean
	dataset	Used to fill dataset with entries of the specific user	DataSet

	chartDataset	Used to provide data for graphs	DataSet
	leaderboardDataset	User to provide data for the leaderboard DataGridViewer	DataSet
	previousUsername	Retains the current username as a string, therefore when checking if the modifications to the account, we do not check when the username has not been changed	String
Admin Form (statAdminForm)	addUser	Variable for whether the form is in Add a User mode	Boolean
	dataset	Used to fill DataGridVeiwer with entries	DataSet
	totalRows	Used to find the amount of user accounts in the database. To then cycle through them with the 'Previous' and 'Next' buttons	Integer
	currentRow	Holds the current row of the user loaded.	Integer
	previousUsername	Retains the current username as a string, therefore when checking if the modifications to the account, we do not check when the username has not been changed	String

	Data Structures	Purpose
Login Form	LogOut	Clear variables that have information pertaining to the credentials of the logged in person. Will be passed into every form when clicking the Log Out button
	ClearTextbox	Clears the textboxes after every attempt and when the user goes to the 'Add User' Form or logs in
	CheckDatabase	Checks the database connection is valid, if not then limit user to only Guest Mode (no database connection needed)
Typing Test Form	FetchRandomParagraphs	Fetch a random paragraph from a text file. Split into lines. Returns a string of the paragraphs.
	FetchRandomWords	Fetch 50 random words (or letters if Alphabet) from a text file. Split into lines. Creates a words array.
	IntoWords	For the Paragraphs extract, split the extract string into an array of words
	IntoEssay	For Random Words and Alphabet, converts word array back to string. So that it can be shown in the RichTextBox
	WithCaps	Adds capitals to Random Words or Alphabet, does it a third of the time so generates a random number out of 3
	CapsFirstLetter	Checks if the random number is 0 then capitalise the letter
	TurnColourCharacters	Changes the forecolour of the character in the RichTextBox
	TurnColourBackground	Changes the colour of the background of the RichTextBox to show when the test has started/stopped
	TimerPauseMode	Starts the timer with a 3 second countdown before the test starts
	Countdown	If in Countdown mode, then start the timer counting down from the limit
	Countup	If in Countup mode, then start the timer from 0 and count up to the limit
	MeasureTime	Calculates the results for Accuracy and WPM, then enter the results into the database
	ChangeKeyColour	Changes a specific keyboard key to a colour depending on the

		next character needing to be pressed
The WordFade Form (wordFadeForm) uses the same data structure, however, are declared separate as they will use different controls, but wills still have the same logic.		
Statistics Form	GetUserDetails	Will get the users credentials from database and loads into Textboxes
	Max	Get the maximum WPM and Accuracy and load it into the labels
	Average	Get the average WPM and Accuracy and load it into the labels
	LeaderboardChange	Loads the leaderboard into the DataGridViewer depending on the ComboBox choice
	ChartWPMLoad	Loads user results into Chart control as WPM / Date
	ChartAccuracyLoad	Loads user results into Chart control as Accuracy / Date
Admin Form	LoadDataGridUser	Load the entries for the user into the DataGridVeiwer
	LoadDateDataGridUser	Load the entries for the user into the DataGridVeiwer with a specified starting and end date
	GetUserDetails	Will get the users credentials from database and loads into Textboxes.
Options Form	ChangeComboItem	Updates ComboBox depending on previous settings
	ChangeCheckBoxMode	Updates CheckBox depending on previous settings

The program will use one database, made in Microsoft Access. It will be relational where there will be multiple tables within the database. This will make organisation easier and connecting to only one database is easier to manage. Validation will occur in the program before inserting it into the database. This limits the likelihood of errors and erroneous data being in the database, risking corruption.

The database comprises of two tables. One called [userData] which stores the login details of each user, the other is called [entryData] which stores each individual entry into the table. Here below, are the fields within the database.

	Database Field	Purpose	Type
userData	UserID	Primary key for each individual login account. Helps to differentiate and make every record individual	AutoNumber
	Username	The user's username when logging into the account. This will also be unique to each user.	Short Text
	Password	In order to secure each account safely a password will be required. Unique to every user which they can pick	Short Text
	Name	What the user would like to be called by when using the program. I will be used mainly for Admin Form so the Admin knows who the person who owns that login, as usually Username are unidentifiable	Short Text
	Admin	Yes or No value, which determines Admin Privileges in the program	Short Text
entryData	EntryID	Primary key for each individual entry in the table. For differentiation and uniqueness	AutoNumber
	UserID	Foreign key taken from [userData] to reference an entry with a user.	Number
	Date	To enter the data the entry was performed on	Date/Time

	WPM	'Word per Minute' to note down how fast the user was typing for	Number
	Accuracy	Represents the percentage of mistakes the user has done in comparison to the whole extract	Number
	Mode	The mode of how the entry was conducted. E.g. 'Paragraphs' or 'Random Words'	Short Text
	Time Duration	How long the user was typing for	Short Text
	Capital Letters	Specifies whether Capital Letters were present in the extract	Short Text
	Punctuation	Specifies whether Punctuation were present in the extract	Short Text

Test Data

In terms of justifying each test I have plans to carry out in my development process, which for some achieve the same thing I will reference below:

Navigation: There must be a method to navigate between forms in my solution in accordance to what the user wants to do. Therefore, a test must be done to test the validity of the methods.

Validation For Database: There must be validation in a solution in order to confirm the connection and the information flowing between the database and the form. This will avoid errors and legitimise the efficacy of the database.

Validation for User Account: To check the validity of the user when logging into the account. The program needs to check the user against the credential.

Validation for User Error: There must be validation in a solution to check for user errors that are not wanted or needed which could have consequences if not checked for.

Validation for Navigation: Need to check the user before allowing them into certain sections into the program.

Security: To check that tests are in place to limit the vulnerabilities in the program that would lead to the user's account being compromised.

Input: Need to make sure that a control handling an input is working correctly at the right time.

Typing Feature working correctly: The essential feature of my solutions, where this feature will need to be functioning correctly, for other parts like the 'Statistics' form to work correctly too.

Timer: The timer needs to be working correctly, in order to show the validity of the WPM results for a user

Fetching extract from text files: Without the extracts, there would be no Typing Feature, so need to test that they are loading in.

Fetching/Inserting information into/from the database: Need a method of interacting with the database. Therefore, it must be tested to see if it works.

Settings: Need to keep the options the solution currently has shown to the user and valid. Otherwise, the user will not know what the settings currently are or how to change them

Check Font Dialog: Need to see whether the font dialog works, in order to achieve the usability feature of the user being able to change the type, style and size

	Test	Test Data	Expected Output	Why?
Login Form	'Create Account' button leads to the 'Add User' form	Click 'Create Account' button	'Add a User' form shows, the login form is hidden	Navigation
	Textboxes can be inputted into by the user	Type in 'Username' textbox Type in 'Password' textbox	The input is entered and shows up on screen correctly	Input
	User will not be able to log in unless both input Textboxes are inputted. And if the text is beneath 5 characters.	Click 'Submit' with only the 'Username' Textbox has been inputted	MessageBox prompt saying 'Please input details in both textboxes'	Validation for Database, Validation for User Error
	When wrong credentials are inputted, it is not a successful login and will not be allowed in	Click 'Submit' with the wrong username Click 'Submit' with the correct username but wrong password relating to it	MessageBox prompt saying 'Login unsuccessful, please try again'	Validation for User Account
	When the credentials are correct, the login is successful. The 'Typing Test' form opens	Enter in correct credentials and click 'Submit'	'Login' form closes and 'Typing Test' form opens	Validation for User Account, Navigation
	If database connection is not present, it informs the user	Take Microsoft Access file out of program folders, and try to log in, clicking 'Submit'	MessageBox prompt saying 'Connection to database unsuccessful, please use Guest Mode instead'	Validation for Database
	User can access the Main Program without a login by using 'Guest Mode'	Click 'Guest Mode' button	'Login form closes and 'Typing Test' form opens	Navigation
	When an unsuccessful attempt has been made, a label appears saying how many more attempts are allowed before the form closes	Get first attempt wrong Get second attempt wrong Get third attempt wrong	Label shown 'You have 2 attempts left' Label changes 'You have 1 attempt wrong Application closes 0	Security against Brute Force attacks
	Program is closed when the user wants it to	Click 'Cancel' button	Application closes	Navigation
Add User	User can go back to Login form without adding a user	Click 'Back' button	'Login' form shows and 'Add a User' form closes	Navigation

	Textboxes can be inputted into by the user	Type in 'Username' textbox Type in 'Password' textbox Type in 'Retry Password' textbox Type in 'Name' Textbox	The is entered and shows up on screen correctly and responsively	Input
	User will not be able to add an account unless all Textboxes are inputted	Click 'Submit' with only the 'Username' Textbox has been inputted	MessageBox prompt saying 'Please input details in both textboxes'	Validation for Database, Validation for User Error
	'Password' and 'Retry Password' textboxes have to have the same input, if not the account will not be added	Type in 'Password' and 'Retry Password' with different inputs add fill in the other textboxes then click 'Submit'	MessageBox prompt saying 'Make sure the password entered are correct'	Validation for Database, Validation for User Error
	If the Username wanting to be added is already a pre-existing login then prevent from adding it again	Type in 'XXXX' in the 'Username' textbox and fill in the other textboxes and click 'Submit'	MessageBox prompt saying 'Username has already been taken, please use a different one'	Validation for Database
	If database connection is not present, it informs the user	Take Microsoft Access file out of program folders, and try to add an account, clicking 'Submit'	MessageBox prompt saying 'Connection to database unsuccessful, please use Guest Mode instead'	Validation for Database
	When the other requirements are met, a MessageBox shows to check if the action should be completely	Enter in credentials and click 'Submit'	MessageBox prompt saying 'Are you sure?'	Validation for User Error
	When the MessageBox shows after submitting, if the clicked 'Yes' then the account is added	Enter in credentials and click 'Submit' then 'Yes'	MessageBox prompt saying 'User has been successfully added'	Validation for User Error, Inserting information into the database
	When the MessageBox shows after submitting, if clicked 'No' then the account is not added and directed back to the form	Enter in credentials and click 'Submit' then 'No'	'Add a User' form is shown with no account being added into the database	Validation for User Error
Typing Test Form	User can navigate to the WordFade form through the menu bar	Click 'WordFade' button in menu bar	WordFade form opens, Typing Test form closes	Navigation
	User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar	Click 'Statistics' button with no entries being made on that account Click 'Statistics' button when in Guest Mode	MessageBox prompt saying 'You cannot access this being in Guest Mode or not having at least 1 entry'	Navigation, Validation for Navigation

	User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar	Click 'Statistics' button when logged into an account that has at least 1 entry made	'Statistics' form opens, Typing Test form closes	Navigation, Validation for Navigation
	User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar	Click 'Admin' button when user does not have Admin Privileges Click 'Admin' button when in Guest Mode	MessageBox prompt saying 'You do not have access to this'	Navigation, Validation for Navigation
	User can navigate to the 'Admin' form when the criteria is met through the menu bar	Click 'Admin' button when user does have Admin Privileges	'Admin' form opens, Typing Test form closes	Navigation, Validation for Navigation
	User can navigate to the 'Options' form through the menu bar	Click 'Options' button	'Options' form opens, Typing Test form closes	Navigation
	Extract and Input Textbox is not editable	Try to type in the Extract Textbox Try to type in the Input Textbox	Nothing changes	Input, Typing Feature working correctly
	Before a Typing Test has started, the user cannot input into the Input Textbox	Have Typing Test form open, try to input text into Input Textbox	No input is registered	Input, Typing Feature working correctly
	User can log out of the program	Click 'Log Out'	Login form opens, Typing Test form closes	Navigation
	An extract is loaded into the form when a Typing Test is about to start	Click 'Start' button	An extract is presented on the screen	Fetching extract from text files, Typing Feature working correctly
	A count down from 3 starts, when at 0 the Typing Test starts every 0.1s	Click 'Start' button	Time label counts from 3 to 0, then restart timer to start counting for the Typing Test	Timer, Typing Feature working correctly
	When the Typing Test starts, the user is now able to input into the Input Textbox	Click 'Start' button, try to input text into Input Textbox	User can input textbox into the Textbox	Input, Typing Feature working correctly
	When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'	Click the 'Start' button	The button is Red and the text is now 'Stop'	Typing Feature working correctly
	WHEN THE TYPING TEST HAS STARTED			
	Input Textbox is now editable	Type in Input Textbox	Inputs are registered	Input, Typing Feature working correctly

	Timer is updated in a label every 0.1s until the end or is stopped	Start the Typing Test	Label is updated to new time value	Timer
	If the character the user types is correct, change the forecolour for that letter on the extract to Green, keyboard key highlighted is updated	Type the next correct letter into the Input Textbox	The character is now Green, keyboard key highlighted is updated to next character	Input, Typing Feature working correctly
	If the character the user types is incorrect, change the forecolour for that letter on the extract to Red	Type an incorrect character	The character is now Red	Input, Typing Feature working correctly
	If the user has made errors and does not correct them, make all subsequent characters Red regardless of if they are correct, and do not move onto next word	Type an incorrect character then type in correct characters and try to press space to move onto next word	Subsequent characters are Red, and the Input Textbox is not cleared	Input, Typing Feature working correctly
	If the user has did not make any errors but backspaces on a Green character, change it to original colour	Type a correct character, then delete it	Character now changes to original colour, keyboard key highlighted is updated to previous character	Input, Typing Feature working correctly
	If the user has made errors, and backspaces to correct it, make the Red character now original colour	Type in an incorrect character and delete it	Character now changes to original colour,	Input, Typing Feature working correctly
	If the user tries to backspace to a previous word, stop at the start of the current word	Try to press backspace on a clear Input Textbox after completing a word	Nothing occurs,	Input, Typing Feature working correctly
	If the word is correctly typed completely and the user presses space onto the next word, clear the Input Textbox to make room for the new character	Type the word correctly and then press space	Input Textbox is cleared	Input, Typing Feature working correctly
	If the extract has been completed, stop the Test, calculate and show the results	Type the whole extract	Test has been stopped and Input Textbox is cleared and cannot be edited, calculate and show in the label WPM, Accuracy and Time	Input, Typing Feature working correctly, Timer

	If the Time Limit is reached, stop the Test, calculate and show the results	Wait the whole duration of the time	Test has been stopped and Input Textbox is cleared and cannot be edited, calculate and show in the label WPM, Accuracy and Time	Input, Typing Feature working correctly, Timer
	Once results are calculated and the user is not in Guest Mode, input them into the database corresponding to the user	Log into the program as a user and complete a Typing Test	Results are inputted into the database corresponding to the user by their 'UserID'	Validation of User Account, Inserting information into the database
	Once results are calculated and the user is in Guest Mode, do not input results into the database	Be in Guest Mode and complete a Typing Test	Results are not inputted into the database. But are still calculated and shown to the user.	Validation of User Account
	If the 'Stop' button is clicked, stop the Typing Test	Click 'Stop' button	Typing Test is stopped, results are not calculated, Input Textbox is cleared	Typing Feature working correctly
	If the database cannot be connected to input the results, send a prompt to the user	Complete a Typing Test without the database present in the programs folder	MessageBox Prompt saying 'Database connection is not present'	Validation for Database
WordFade Form	User can navigate to the 'Typing Test' form through the menu bar	Click 'Typing Test' button in menu bar	Typing Test form opens, WordFade form closes	Navigation
	User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar	Click 'Statistics' button with no entries being made on that account Click 'Statistics' button when in Guest Mode	MessageBox prompt saying 'You cannot access this being in Guest Mode or not having at least 1 entry'	Navigation, Validation for Navigation
	User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar	Click 'Statistics' button when logged into an account that has at least 1 entry made	'Statistics' form opens, WordFade form closes	Navigation, Validation for Navigation
	User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar	Click 'Admin' button when user does not have Admin Privileges Click 'Admin' button when in Guest Mode	MessageBox prompt saying 'You do not have access to this'	Navigation, Validation for Navigation

	User can navigate to the 'Admin' form when the criteria is met through the menu bar	Click 'Admin' button when user does have Admin Privileges	'Admin' form opens, WordFade form closes	Navigation, Validation for Navigation
	User can navigate to the 'Options' form through the menu bar	Click 'Options' button	'Options' form opens, WordFade form closes	Navigation
	Extract Textbox is not editable	Try to type in the Extract Textbox	Nothing changes	Input, Typing Feature working correctly
	Before a Typing Test has started, the user cannot input into the Input Textbox	Have Typing Test form open, try to input text into Input Textbox	No input is registered	Input, Typing Feature working correctly
	User can log out of the program	Click 'Log Out'	Login form opens, Typing Test form closes	Navigation
	An extract is loaded into the program, and the first 3 words are in the Textboxes when a WordFade is about to start	Click 'Start' button	First three words of the extract is presented	Fetching extract from text files, Typing Feature working correctly
	A count down from 3 starts, when at 0 the Typing Test starts every 0.1s	Click 'Start' button	Time label counts from 3 to 0, then restart timer to start counting for the Typing Test	Timer, Typing Feature working correctly
	When the WordFade starts, the user is now able to input into the Input Textbox	Click 'Start' button, try to input text into Input Textbox	User can input textbox into the Textbox	Input, Typing Feature working correctly
	When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'	Click the 'Start' button	The button is Red and the text is now 'Stop'	Typing Feature working correctly
	WHEN THE TYPING TEST HAS STARTED			
	Input Textbox is now editable	Type in Input Textbox	Inputs are registered	Input, Typing feature working correctly
	Timer is updated in a label every 0.1s until the end or is stopped	Start the Typing Test	Label is updated to new time value	Timer, Typing feature working correctly
	If the character the user types is correct, change the forecolour for that letter on the extract to Green, keyboard key highlighted is updated	Type the next correct letter into the Input Textbox	The character is now Green, keyboard key highlighted is updated to next character	Input, Typing feature working correctly

	If the character the user types is incorrect, change the forecolour for that letter on the extract to Red	Type an incorrect character	The character is now Red	Input, Typing feature working correctly
	If the user has made errors and does not correct them, make all subsequent characters Red regardless of if they are correct, and do not move onto next word	Type an incorrect character then type in correct characters and try to press space to move onto next word	Subsequent characters are Red, and the Input Textbox is not cleared	Input, Typing feature working correctly
	If the user has did not make any errors but backspaces on a Green character, change it to original colour	Type a correct character, then delete it	Character now changes to original colour, keyboard key highlighted is updated to previous character	Input, Typing feature working correctly
	If the user has made errors, and backspaces to correct it, make the Red character now original colour	Type in an incorrect character and delete it	Character now changes to original colour	Input, Typing feature working correctly
	If the user tries to backspace to a previous word, stop at the start of the current word	Try to press backspace on a clear Input Textbox after completing a word	Nothing occurs,	Input, Typing feature working correctly
	If the word is correctly typed completely and the user presses space onto the next word, clear the Input Textbox to make room for the new character. Move the two other words down and load in the third word	Type the word correctly and then press space	Input Textbox is cleared, 2nd word moves into the 1st Textbox, 3rd word moves into the 2nd Textbox, the next word after is loaded into the 3rd Textbox	Fetching extract from text files, Input, Typing feature working correctly
	If near the end of the essay, 3rd and/or 2nd textboxes will have no string in it (to signify end of essay)	Type whole extract correctly	'thirdWordRichTextBox' will have no string in it, next word 'secondWordRchText Box; will have no word in it. Next word is end of extract	Validation, Typing feature working correctly

	If the extract has been completed, stop the Test, calculate and show the results	Type the whole extract	Test has been stopped and Input Textbox is cleared and cannot be edited, calculate and show in the label WPM, Accuracy and Time	Input, Typing Feature working correctly, Timer
	If the Time Limit is reached, stop the Test, calculate and show the results	Wait the whole duration of the time	Test has been stopped and Input Textbox is cleared and cannot be edited, calculate and show in the label WPM, Accuracy and Time	Input, Typing Feature working correctly, Timer
	Once results are calculated and the user is not in Guest Mode, input them into the database corresponding to the user	Log into the program as a user and complete a Typing Test	Results are inputted into the database corresponding to the user by their 'UserID'	Validation of User Account, Inserting information into the database
	Once results are calculated and the user is in Guest Mode, do not input results into the database	Be in Guest Mode and complete a Typing Test	Results are not inputted into the database. But are still calculated and shown to the user.	Validation of User Account
	If the 'Stop' button is clicked, stop the Typing Test	Click 'Stop' button	Typing Test is stopped, results are not calculated, Input Textbox is cleared	Typing Feature working correctly
	If the database cannot be connected to input the results, send a prompt to the user	Complete a Typing Test without the database present in the programs folder	MessageBox Prompt saying 'Database connection is not present'	Validation for Database
Statistics Form	User can navigate to the 'Typing Test' form through the menu bar	Click 'Typing Test' button in menu bar	Typing Test form opens, Statistics form closes	Navigation
	User can navigate to the 'WordFade' form through the menu bar	Click 'WordFade' button in menu bar	WordFade form opens, Statistics form closes	Navigation
	User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar	Click 'Admin' button when user does not have Admin Privileges Click 'Admin' button when in Guest Mode	MessageBox prompt saying 'You do not have access to this'	Navigation, Validation for Navigation
	User can navigate to the 'Admin' form when the criteria is met through the menu bar	Click 'Admin' button when user does have Admin Privileges	'Admin' form opens, Typing Test form closes	Navigation, Validation for Navigation

	User can navigate to the 'Options' form through the menu bar	Click 'Options' button	'Options' form opens, Typing Test form closes	Navigation
	Loads user details from database into the Textboxes	Open Statistics form	User credentials are loaded into the textboxes	Fetching information from the database
	User can edit 'Username', 'Password' and 'Name' Textboxes but not 'UserID' Textbox or 'Admin Privileges' Checkbox	Try to input in 'Username' Textbox Try to input in 'Password' Textbox Try to input in 'Name' Textbox Try to input in 'UserID' Textbox Try to change 'Admin Privileges' Checkbox	The inputs should be registered in 'Username', 'Password' and 'Name' Textboxes but not 'UserID' Textbox or 'Admin Privileges' Checkbox	Input
	'Add User' button was not clicked before and the user edits their credentials with each textbox having text in it, 'Confirm' button is clicked and the credentials are updated if all the inputs are above 5 characters	Edit credentials in the Textboxes with all at least 5 characters in length and click 'Confirm'	Update user credentials in the database, MessageBox prompt saying 'Action Completed'	Updating information into the database, Validation for database, security
	'Add User' button was not clicked before and the user edits their credentials and 'Confirm' button is clicked. However, a textbox does not have text in it so does not update	Edit credentials in the Textboxes and click 'Confirm'. But with a textbox without text in it Edit credentials in the Textboxes and click 'Confirm'. But with a textbox with text below 5 characters	MessageBox prompt saying 'Please make sure each Textbox has an input and at least 5 characters'	Updating information into the database, Validation for database, security
	When 'Add User' is clicked, clear the textboxes and change the button to 'Stop Add User'. It's now in Add User Mode	Click 'Add User'	Clear user credential textboxes, change button text to 'Stop Add User'	Input

	User goes into Add User mode, the user adds in credentials in every textbox and clicks 'Confirm' to add user	Clicks 'Add User', user adds in credentials for new user, each textbox has text in it, clicks 'Confirm'	Insert new user credentials in the database, MessageBox prompt saying 'Action Completed', Typing Test form opens, Statistics form closes, the user is now logged in as the added user	Inserting information into the database, Validation for database, security
	User goes into Add User mode, the user adds in credentials clicks 'Confirm' to add user. If the textboxes have no input or below 5 character the action will not be completed	Clicks 'Add User', user adds in credentials for new user, each textbox has text in it, clicks 'Confirm'. With a textbox having no input Clicks 'Add User', user adds in credentials for new user, each textbox has text in it, clicks 'Confirm'. With a textbox having an input below 5 characters	MessageBox prompt saying 'Please make sure each Textbox has an input and at least 5 characters'	Inserting information into the database, Validation for database, security
	User wants to go out of Add User Mode	Clicks 'Stop Add User'	Button changes to 'Add User', user credentials are loaded back into the program	Input
	User wants to delete user by click 'Delete' button	Clicks 'Delete' user button	MessageBox Prompt saying, 'Are you sure?'	Validation for User Error
	User wants to delete user by click 'Delete' button, and clicks 'Yes' to MessageBox	Clicks 'Delete' button, then 'Yes' to MessageBox	MessageBox prompt 'User deleted', then opens Login form and closes Statistics form	Validation for User Error, Updating information into the database, Validation For User Account
	User wants to delete user by click 'Delete' button, and clicks 'No' to MessageBox	Clicks 'Delete' button, then 'Not' to MessageBox	Nothing happens	Validation for User Error
	If the user is in Add User mode and clicks the 'Add User button, stop it	Click 'Add User' then 'Delete' button	MessageBox prompt saying 'You cannot delete a user you have not created yet'	Validation for User Error
	If the clicks the 'Clear' button, clear the Textboxes	Click 'Clear' button	'Username', 'Password' and 'Name' textboxes are cleared of input	Input

	Loads 'Highest WPM', 'Average WPM', 'Highest Accuracy', 'Average Accuracy' into labels	Open Statistics form	Labels would be correctly updated to new values	Fetching information from the database
	Load Graph controls to 'Average WPM / Date' and 'Average Accuracy / Date' as Line Graphs for that user	Open Statistics form	Graphs would be correctly loaded	Fetching information from the database
	ComboBox would have the options of 'Highest WPM', 'Average WPM', 'Highest Accuracy', 'Average Accuracy' and 'Most Entries'	Open Statistics form, click the ComboBox	ComboBox would have the correct options	Input
	When choosing an option in the ComboBox, the DataGridVeiwer would be changed in accordingly to the option chosen	Choose 'Highest WPM' Choose 'Average WPM' Choose 'Highest Accuracy' Choose 'Average Accuracy' Choose 'Most Entries'	Show 'Highest WPM' leaderboard Show 'Average WPM' leaderboard Show 'Highest Accuracy' leaderboard Show 'Average Accuracy' leaderboard Show 'Most Entries' leaderboard	Input, Fetching information from the database
Admin Form	User can navigate to the 'Typing Test' form through the menu bar	Click 'Typing Test' button in menu bar	Typing Test form opens, Admin form closes	Navigation
	User can navigate to the 'WordFade' form through the menu bar	Click 'WordFade' button in menu bar	WordFade form opens, Admin form closes	Navigation
	User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar	Click 'Statistics' button with no entries being made on that account Click 'Statistics' button when in Guest Mode	MessageBox prompt saying 'You cannot access this being in Guest Mode or not having at least 1 entry'	Navigation, Validation for Navigation
	User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar	Click 'Statistics' button when logged into an account that has at least 1 entry made	'Statistics' form opens, Admin form closes	Navigation, Validation for Navigation
	User can navigate to the 'Options' form through the menu bar	Click 'Options' button	'Options' form opens, Typing Test form closes	Navigation

	Loads the first user details from database into the Textboxes and DataGridVeiwer	Open Admin form	The first user credentials are loaded into the textboxes from the database and the entries into the DataGridVeiwer	Navigation
	User can edit 'Username', 'Password', 'Name' Textboxes and 'Admin Privileges' Checkbox but not 'UserID' Textbox	Try to input in 'Username' Textbox Try to input in 'Password' Textbox Try to input in 'Name' Textbox Try to input in 'UserID' Textbox Try to change 'Admin Privileges' Checkbox	The inputs should be registered in 'Username', 'Password', 'Name' Textboxes and 'Admin Privileges' Checkbox but not the 'UserID' Textbox	Input
	User can go to next user by clicking the '→' button	Click '→' button	The next user is loaded into the textboxes, DataGridVeiwer is updated with the user's entries	Fetching information from the database
	User can go to the previous user by clicking the '←' button	Click '←' button	The previous user is loaded into the textboxes, DataGridVeiwer is updated with the user's entries	Fetching information from the database
	'Add User' button was not clicked before and the user edits their credentials with each textbox having text in it, 'Confirm' button is clicked and the credentials are updated	Edit credentials in the Textboxes and click 'Confirm'	Update user credentials in the database, MessageBox prompt saying 'Action Completed'	Updating information into the database, Validation for database, security
	'Add User' button was not clicked before and the user edits their credentials and 'Confirm' button is clicked. However, a textbox does not have text in it so does not update	Edit credentials in the Textboxes and click 'Confirm'. But with a textbox without text in it Edit credentials in the Textboxes and click 'Confirm'. But with a textbox with text below 5 characters	MessageBox prompt saying 'Please make sure each Textbox has an input and at least 5 characters'	Updating information into the database, Validation for database, security

	When 'Add User' is clicked, clear the textboxes and change the button to 'Stop Add User'. It's now in Add User Mode	Click 'Add User'	Clear user credential textboxes, change button text to 'Stop Add User'	Input
	User goes into Add User mode, the user adds in credentials in every textbox and clicks 'Confirm' to add user	Clicks 'Add User', user adds in credentials for new user, each textbox has text in it, clicks 'Confirm'	Insert new user credentials in the database, MessageBox prompt saying 'Action Completed', Typing Test form opens, Statistics form closes, the user is now logged in as the added user	Inserting information into the database, Validation for database, security
	User goes into Add User mode, the user adds in credentials clicks 'Confirm' to add user. If the textboxes have no input or below 5 character the action will not be completed	Clicks 'Add User', user adds in credentials for new user, each textbox has text in it, clicks 'Confirm'. With a textbox having no input Clicks 'Add User', user adds in credentials for new user, each textbox has text in it, clicks 'Confirm'. With a textbox having an input below 5 characters	MessageBox prompt saying 'Please make sure each Textbox has an input and at least 5 characters'	Inserting information into the database, Validation for database, security
	User wants to go out of Add User Mode	Clicks 'Stop Add User'	Button changes to 'Add User', user credentials are loaded back into the program	Input
	User wants to delete user by click 'Delete' button	Clicks 'Delete' user button	MessageBox Prompt saying, 'Are you sure?'	Validation for User Error
	User wants to delete user which is their own login by click 'Delete' button, and clicks 'Yes' to MessageBox	Clicks 'Delete' button, then 'Yes' to MessageBox	MessageBox prompt 'User deleted', then opens Login form and closes Statistics form	Validation for User Error, Updating information into the database, Validation for User Account
	User wants to delete user which is not their own login by click 'Delete' button, and clicks 'Yes' to MessageBox	Clicks 'Delete' button, then 'Yes' to MessageBox	MessageBox prompt 'User deleted'	Validation for User Error, Updating information into the database

	User wants to delete user by click 'Delete' button, and clicks 'No' to MessageBox	Clicks 'Delete' button, then 'Not' to MessageBox	Nothing happens	Validation for User Error
	If the user is in Add User mode and clicks the 'Delete' button, stop it	Click 'Add User' then 'Delete' button	MessageBox prompt saying 'You cannot delete a user you have not created yet'	Validation for User Error
	If the clicks the 'Clear' button, clear the Textboxes	Click 'Clear' button	'Username', 'Password' and 'Name' textboxes are cleared of input	Input
	User can specify the dates the entries are between entries in DataGridViewer	Specify the start and end date with the two DateTimePickers, then click the 'Search' button	Entries in DataGridVeiwer are updated to the new conditions	Input, Fetching information from the database
	User can search for a specific username, and it will be loaded into the Textboxes and DataGridVeiwer	Enter the username into the textbox and press the 'Search' button	User details and entries are updated to that user	Input, Fetching information from the database
	User tries to search for a username not in the program then prompt the user	Enter the username that is not in the database into the textbox and press the 'Search' button	MessageBox prompt saying 'User has not been found'	Input, Fetching information from the database, Validation for database
Options Form	All pre-existing settings are loaded into the controls	Open Options form	All pre-existing settings are loaded into the controls	Settings
	User can navigate to the WordFade form through the menu bar	Click 'WordFade' button in menu bar	WordFade form opens, Typing Test form closes	Navigation
	User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar	Click 'Statistics' button with no entries being made on that account Click 'Statistics' button when in Guest Mode	MessageBox prompt saying 'You cannot access this being in Guest Mode or not having at least 1 entry'	Navigation, Validation for Navigation
	User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar	Click 'Statistics' button when logged into an account that has at least 1 entry made	'Statistics' form opens, Typing Test form closes	Navigation, Validation for Navigation
	User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar	Click 'Admin' button when user does not have Admin Privileges Click 'Admin' button when in Guest Mode	MessageBox prompt saying 'You do not have access to this'	Navigation, Validation for Navigation

	User can navigate to the 'Admin' form when the criteria is met through the menu bar	Click 'Admin' button when user does have Admin Privileges	'Admin' form opens, Typing Test form closes	Navigation, Validation for Navigation
	User can navigate to the 'Typing Test' form through the menu bar	Click 'Options' button	'Options' form opens, Typing Test form closes	Navigation
	User can log out of the program	Click 'Log Out'	Login form opens, Typing Test form closes	Navigation
	When the type of extract for Typing Test ComboBox is changed, it will be changed in the Typing Test	Change ComboBox to 'Paragraphs', then open Typing Test form and press 'Start' Change ComboBox to 'Random Words', then open Typing Test form and press 'Start'	The extract loaded is a 'Paragraph' The extract loaded is comprised of 'Random Words'	Input, Settings, Fetching extract from text files
	When the type of extract for WordFade ComboBox is changed, it will be changed in the WordFade test	Change ComboBox to 'Paragraphs', then open WordFade form and press 'Start' button Change ComboBox to 'Random Words', then open WordFade form and press 'Start' button Change ComboBox to 'Alphabet', then open WordFade form and press 'Start' button	The extract loaded is a 'Paragraph' The extract loaded is comprised of 'Random Words' The extract loaded is comprised of alphabet letters	Input, Settings, Fetching extract from text files
	When the CountDown or CountUp ComboBox is changed, it will be changed in the Typing Test and WordFade	Change ComboBox to 'CountDown' and open either Typing Test or WordFade and press 'Start' Change ComboBox to 'CountUp' and open either Typing Test or WordFade and press 'Start'	Timer is counting down from the limit Timer is counting up to the limit	Input, Settings, Timer
	When the Time Duration ComboBox is changed, it will be changed in the Typing Test and WordFade	Change ComboBox to (15,30,45,60) seconds and open Typing Test and click 'Start' Change ComboBox to (15,30,45,60) seconds and open WordFade and click 'Start'	Time limit is changed to new limit	Input, Settings, Timer

	When the Capitals checkbox is changed, it is changed for Typing Test and WordFade	Tick the Checkbox and open either Typing Test or WordFade and press 'Start' Untick the Checkbox and open either Typing Test and WordFade and press 'Start'	The extract has capitals The extract does not have capitals	Input, Settings, Fetching extract from text files
	When the Punctuation checkbox is changed, it is changed for Typing Test and WordFade (this only applies to 'Paragraph' extracts	Tick the Checkbox and open either Typing Test or WordFade and press 'Start' Untick the Checkbox and open either Typing Test and WordFade and press 'Start'	The extract has punctuation The extract does not have punctuation	Input, Settings, Fetching extract from text files
	When the Sudden Death checkbox is changed, it is changed for Typing Test and WordFade	Tick the Checkbox and open either Typing Test or WordFade and press 'Start' Untick the Checkbox and open either Typing Test and WordFade and press 'Start'	When an error has been made, stop the Test When an error has been made, carry on	Input, Settings, Typing Feature working correctly
	When the Keyboard ComboBox has changed, it will be changed in the Typing Test and WordFade	Change ComboBox to 'Shown and Highlighted' and open either Typing Test or WordFade and press 'Start' Change ComboBox to 'Shown and Not Highlighted' and open either Typing Test or WordFade and press 'Start' Change ComboBox to 'Not Show' and open either Typing Test or WordFade and press 'Start'	Keyboard is shown and next character needed to be pressed is highlighted Keyboard is shown and the next character needed is not highlighted Keyboard is not shown at all	Input, Settings, Typing Feature working correctly
	User can change the font type, style and size for the Typing Test and WordFade	Click 'Change Font' button, pick the font type, style and size and then open either Typing Test or WordFade and press 'Start'	Extract has font type, style and size has been changed to new setting	Input, Check font dialog, Typing Feature working correctly
	User can click a link to be directed to a website to download Microsoft Access Runtime	Click the link label	Link is open in the users default web browser	Input, Validation for Database

Post Development Phase

After developing my solution, there will be stringent testing performed further than the degree of the test data shown above. This will mainly focus on the vulnerabilities of my solution that involves the login system primarily. It will be penetration tested.

I plan to encrypt and store hashes of passwords in my database rather than sort unsalted passwords in my database that anyone could access and compromise that account. This would make my solution more secure and safer to use.

I have the intention to develop my application further by having an online element to the project. Even though I have aimed for this solution to be supported offline. I feel that an optional, more social element will help improve the quality of life of my solution as adding a competitive aspect, will drive the usage of my solution. This will include being able to specifically friend others, so you can see a limited view of their statistics and compared and compete within your circle of friends.

Another competitive feature I would like to add is where results of specific paragraphs will be logged, and the user's friends can compete against the specific extract to try and beat it. This would work as a leaderboard and will encourage a social and competitive atmosphere.

I would like to build upon the statistics part of my solution by logging users' weaker letters and words. This would be useful for them to see in detail where they are at their weakest. I would also like to add a feature where building upon the weak letters, words in extracts will be focused on them. This will expose users to the weakest parts of their touch typing and tackle it head on.

After every update, test data and tables will be used to check the adequacy of the solutions and highlight any bugs or vulnerabilities that are found or have been made due to the update. This is because an update could break the functionality of the connection to the database or logic within the SQL or wider code.

Development

It should be noted that not all of the code is shown in the 'Development' section. All of the iterations are included in the folders and at the end of this document in the 'Code For Versions' section.

The release builds are found in the '.../bin/Release/[version]' in each folder

First Version

My first version focused on the main features of the solutions without the integration of the database. This meant working on the basic functionality of the solution therefore, not particularity directed as the 'look' of my solution at the moment. This included designing and naming all the controls that I will use in the project at a later date.

Each main component of the program is separated into forms. A form for each main aspect of the program helps me be organised. Where then I aimed to have the majority of the program comprised of functions which can be called. I have decided this I want my code to be well structured and modular in nature.

Login Form

For the login system, I wanted to have the basic functionality. This meant giving access to the main program when the user has inputted the correct credentials, having validation that restricts access to an account, a guest mode access and the application closing after 3 tries. However, as this version did not have the access to a database, I could not implement the functionality for communicating with the database. I decided against adding the database in the first version because I aimed to have the basic functionality nailed first.

This stage of the development relates to the analysis stage because it begins the development of the basic functionality of a simple typing program. Where the user has the ability to input characters and gains results of WPM, Accuracy and Time Duration. The design stage helped me plan out what to develop first and how the controls are displayed in my program. All of the work going forward has been based off of the work in the analysis and design stage.

Passing Variables and functions across the program

The Login Form is the Start Up object of the project and I have set up the program to only close the application when the Start Up object closes. So in this case, the Login Form is the first and last form in the application. This means that after the user logs in (or enters Guest Mode) and enters the main program. The login form will be hidden but not closed entirely until the user logs out.

Due to this, I can have the main global variables in the Login Form and be able to pass and modify the values of them through the other forms. This will be especially useful when the user customises their experience in the Options Form. I will additionally include some of the functions that are used in the program in here.

In reference to my success criteria in my Analysis section. Declaring these variables will aid in creating functionality in the Options form when choosing options, and the login credentials for Admin and Statistic sections, because without this, there will be no method to transfer data across forms.

```
Imports System.Data.OleDb

Public Class loginForm

    Friend dataUsername As String = " XXXX" 'All friend variables are global that will be used across the form
    Friend dataName As String = "XXXX"
    Friend dataAdmin As Boolean
    Friend Shared dataUserID As String
    Friend timeDuration = "30"
    Friend countType = "CountDown"
    Friend suddenDeath = False
    Friend extractType = "Paragraphs"
    Friend extractWordFadeType = "Paragraphs"
    Friend capitals = True
    Friend punctuation = True
    Friend showKeyboard = "ShowHighlight"
    Friend entriesPresent = False
    Friend globalFont As New Font("Calibri", 28, Font.Style.Regular)
    Friend attemptCount = 3

    Sub UpdateLabelMode(formNow) 'Fetches the mode and updates the label, label is present in every page and is referenced there too
        formNow.modeLabel.Text = "MODE : " & extractType
    End Sub


    Sub LogOut() 'When logging out, the variables with data pertaining to the login is cleared
        dataUserID = ""
        dataUsername = ""
        dataName = ""
        dataAdmin = False
    End Sub
```

Logging into the database

The primary function of the form. In this version, I have implemented variables that the hold the credentials of the user in variables after the login has been successful. There is validation for when the user logs in unsuccessfully. A label becomes visible informing the user how many attempts are left. This is a main feature in making an Account System.

```
    Sub ClearTextBox() 'After each entry or login the textbox is cleared of inputs
        usernameTextBox.Clear()
        passwordTextBox.Clear()
    End Sub

    Sub deductAttempt() 'Deducts an entry from the counter and updates/shows the text label
        attemptCount -= 1
        Select Case attemptCount
            Case 2
                attemptsLabel.Visible = True
                attemptsLabel.Text = "You have 2 attempts left"
            Case 1
                attemptsLabel.Text = "You have 1 attempt left"
            Case 0
                Me.Close() 'After 3 attempts the program closes
        End Select
    End Sub


    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Try
            If usernameTextBox.Text.Trim = "" Or passwordTextBox.Text.Trim = "" Then 'Validation : Checks to see if characters are in
either TextBox
                MessageBox.Show("You have entered your credentials wrong. Make sure that the Username and Password are at least 5
characters in Length. Please try again", "Submit Credentials", MessageBoxButtons.OK, MessageBoxIcon.Error)
            Else
                Dim inputUsername As String
                Dim inputPassword As String
                inputUsername = usernameTextBox.Text.Trim 'Assiging User Inputs to variables, Trim Function takes out spaces
                inputPassword = passwordTextBox.Text.Trim
                If inputUsername <> dataUsername Or inputPassword <> "abc" Then 'Checks to see if they match
                    MessageBox.Show("Username or Password is incorrect. Please try again", "Submit Credentials", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
                    ClearTextBox()
                    deductAttempt() 'minus attempt from list
                ElseIf inputUsername = "XXXX" And inputPassword = "abc" Then 'If the passwords match and are correct
                    Select Case "Yes" 'Checks admin privileges, at the moment I've left it as just "Yes"
                        Case "Yes"
                            dataAdmin = True
                        Case "No"
                            dataAdmin = False
                    End Select
                    ClearTextBox()
                    attemptCount = 3 'reset counter
                    attemptsLabel.Visible = False
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Hide() 'Hides form and does not close as this is the startup form and need variables associated with it
                End If
            End If
        Catch ex As NullReferenceException
            MessageBox.Show("Something went wrong, please try again", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
```

```
            ClearTextBox()
        End Try
    End Sub
```

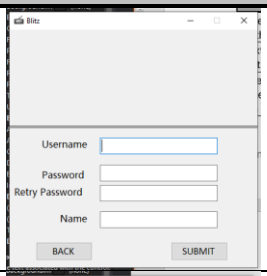
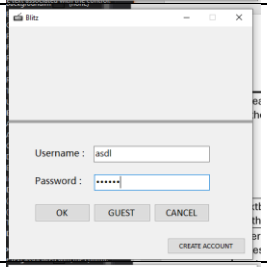
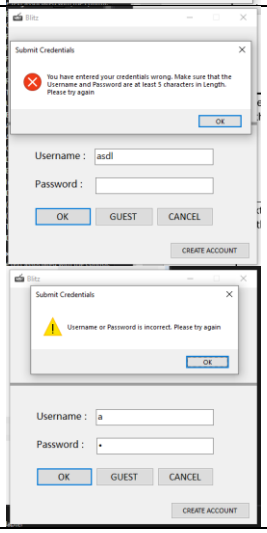
Guest Mode

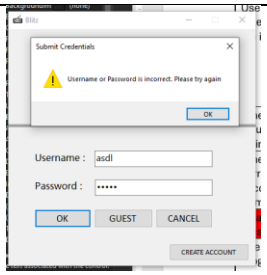
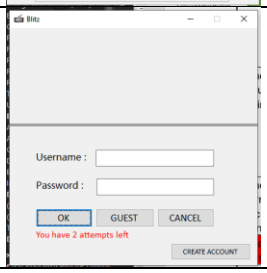
I've added functionality to the Guest mode button, where the variable dataUsername will hold the value "Guest" that will be used for validation in the rest of the program. E.g. checking the value of variable if in Guest mode do not add entry or not allowing access to Admin and Statistics sections

```
    Private Sub guestBtn_Click(sender As Object, e As EventArgs) Handles guestBtn.Click
        dataUsername = "Guest" 'Update variable
        ClearTextBox() 'Clear textbox of inputs so that so sensitive information isn't present
        Dim frm As New typerForm
        frm.Show()
        Me.Hide()
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested

Test	Result	Evidence	Comments
'Create Account' button leads to the 'Add User' form			
Textboxes can be inputted into by the user			
User will not be able to log in unless both input Textboxes are inputted. And if the text is beneath 5 characters.			The form does not check for if the username or password is below 5 characters. Should output MessageBox in previous test.

When wrong credentials are inputted, it is not a successful login and will not be allowed in			The credentials are not fetched from the database
When the credentials are correct, the login is successful. The 'Typing Test' form opens			The username is "XXXX" and password is "abc". The credentials are not fetched from the database
If database connection is not present, it informs the user		N/A	Database is not integrated in this version
User can access the Main Program without a login by using 'Guest Mode'			
When an unsuccessful attempt has been made, a label appears saying how many more attempts are allowed before the form closes			
Program is closed when the user wants it to		N/A	

Remedial Actions for Next Version

In the next iteration, I will need to integrate a Database and set up communication between the program and table including data of login accounts. This will enable the program to also specify who has Admin Privileges and change it accordingly.

Add User Form

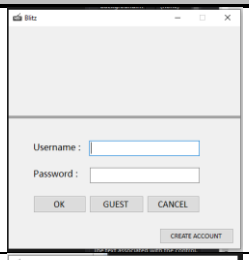
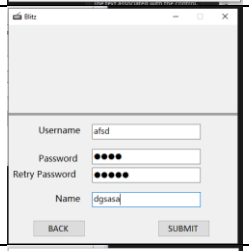
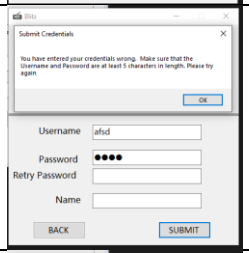
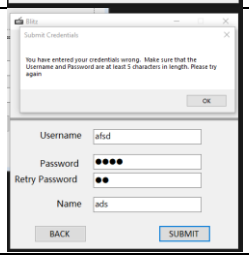
As I did not connect the database in this iteration, there is no functionality needed to be implemented in this version. However, I did add the controls that will be needed in the future. I also added some validation.

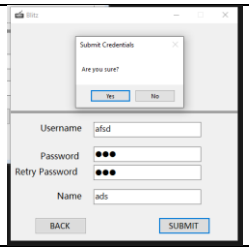
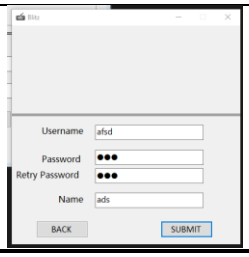
```
'Importing modules needed for future use
Imports System.Data.OleDb

Public Class addUserForm

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Dim result As DialogResult
        If passTextBox.Text = retryPassTextBox.Text And userTextBox.Text.Trim.Length <> 0 And passTextBox.Text.Trim.Length <> 0 And
nameTextBox.Text.Trim.Length <> 0 Then 'Validation : Checking password entries match and that no textboxes and no characters in them
```

```
            result = MessageBox.Show("Are you sure?", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then  If clicked yes then carry on
                    Dim username, password, myName, admin As String 'Initialise variables to then be passed into SQL
                    username = userTextBox.Text.Trim 'Trim them out of whitespace (avoids errors)
                    password = passTextBox.Text.Trim
                    myName = nameTextBox.Text.Trim
                    admin = "No" 'Default value for new user
                End If
            Catch ex As Exception
                MessageBox.Show("Something has gone wrong. Please try again.") 'For database, if the connection or SQL fails
            End Try
        Else
            MessageBox.Show("You have entered your credentials wrong.  Make sure that the Username and Password are at least 5
characters in length. Please try again", "Submit Credentials")  Validation
        End If
    End Sub

    Private Sub addUserForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    End Sub
End Class
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested

Test	Result	Evidence	Comments
User can go back to Login form without adding a user			
Textboxes can be inputted into by the user			
User will not be able to add an account unless all Textboxes are inputted			
'Password' and 'Retry Password' textboxes have to have the same input, if not the account will not be added			
If the Username wanting to be added is already a pre-existing login then prevent from adding it again		N/A	Need to add database integration to check pre-existing users
If database connection is not present, it informs the user		N/A	Database is not integrated in this version

When the other requirements are met, a MessageBox shows to check if the action should be completely			
When the MessageBox shows after submitting, if the clicked 'Yes' then the account is added		N/A	Database is not integrated in this version
When the MessageBox shows after submitting, if clicked 'No' then the account is not added and directed back to the form			

Remedial Actions for Next Version

In the next version I will integrate the database connection. I will then create the 'userData' table that will be used to store the users credentials. This relates to my success criteria in my analysis stage. As I need a method of adding new users and a method of storing the data.

Typing Test Form

My aim was to get the basic functionality of the Typing Test, with reference to the Success Criteria. However, I did not create any functionality pertain got the keyboard or Database connection. However, it should be noted that testing pertaining to the Keyboard is in the Options form as it controls the behaviour of the control.

Loading words from a Text File

I have used the 'FakeItEasy' framework to fetch and read the contents of my textiles from. It has been designed to create an array with each item being a line of text in the text file. For the paragraph text file, the function 'FetchRandomParagraphs' check the number of items (lines or paragraphs) in the array and randomly choose a line/paragraph from the selection and returns a string of the paragraph. This done using System.Random. It is then divide into a word array using the 'IntoWords' function.

A similar method is done with the random words text file with the function 'FetchRandomWords'. However, as each line is just one word. I fetch 50 random items/words from the text file and append it to a new list. The function returns a word array. I use the 'IntoEssay' function to convert the word array to a single string for the essayRichTextBox.

```
'Importing modules needed for future use
Imports System.Data.OleDb

    Function FetchRandomParagraphs() As String 'Fetch paragraphs from textfile
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt") 'Add each line as a
string in an array
        Dim essayExtract As String
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Integer
        randomNumber = Generator.Next(0, lineCount) 'Generate a random number in the range of 1 to the amount of lines in the array
        essayExtract = lines.ElementAt(randomNumber).ToString 'Pass the string at the random index value into the varaibles and return
the variable
        Return essayExtract
    End Function
```

```
    Function FetchRandomWords() 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\randomWords.txt") 'initialises an array
on the words on each line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50 'Will add 50 words to the array
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            Return wordList
        End If
    End Function

      Function IntoWords(essay) As String() 'Dividing string into a words array
        Dim wordList As String()
        wordList = essay.Split(" ")
        Return wordList
    End Function

    Function IntoChar(wordList, y) As String() 'For the word that the user is on, it gets split up into an array of letters
        Dim stringInformation As New Globalization.StringInfo(wordList(y))
        Dim characterList(stringInformation.LengthInTextElements - 1) As String
        For i As Integer = 0 To stringInformation.LengthInTextElements - 1
            characterList(i) = stringInformation.SubstringByTextElements(i, 1)
        Next
        Return characterList
    End Function


    Function IntoEssay(wordList) 'With Random Words mode, need to get base extract as string, so converts wordList back to string
        Dim essaySpaces As String
        For Each word In wordList
            essaySpaces += word + " " 'Need to add dspaces as without, all the words will be one long string
        Next
        Dim essay As String = essaySpaces.Remove(essaySpaces.Length - 1, 1) 'From the for loop the string will end on a space which will
cause issues with the typing game so elimnating it solves the problem
        Return essay
    End Function
```

Timer

The timer used is a control already in Visual Studio 2019, that I have set to 'tick' every 100ms or 0.1s. This means that I can update the form and labels every 0.1s.

I have two functions for either Countdown or Countup mode. I have decided to implement it as a function call, where depending on the option chosen, every tick it will call the right function.

```
    Private Sub wordTimer_Tick() Handles wordTimer.Tick
        Select Case loginForm.countType
            Case "CountDown"
                Countdown()
            Case "CountUp"
                CountUp()
        End Select
    End Sub
```

The Timer will be started when the 'Start' button is pressed. Where the 'TimerPauseMode' function is called to reset the global variables and start the timer. The functions are implemented where the timeLeft variable will increment until the duration is met. The first duration is the 3 second pause before a test and resets the duration to the time limit and carries on.

```
    Sub TimerPauseMode(duration) 'Starts the timer in pause mode
        startType = False 'Starting type (puts the Countdown/Countup functino in pause mode)
        timeLeft = Convert.ToInt32(duration) 'global variable timeLeft used to be the duration of the pause before test starts aka (3)
        wordTimer.Start() 'Starts timer
    End Sub

    Sub Countdown()
        If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
            timeLeft -= 0.1 'Decrements by 0.1
            timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
        Else
            If startType = True Then 'If currently playing a typing game then make the timer stop and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
```

```
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START" 'Reset Start Button
                wordTimer.Stop()
                MeasureTime() 'Measure
            Else
                timeDoneLabel.Text = "GO" 'If not currently playing and its countdown at the start then reset timer and start countdown
at specific time duration
                inputTextBox.ReadOnly = False 'Lets user input in textbox
                startType = True 'Changes variable to signfy the test has started
                timeLeft = timeDuration 'Resets duration to actual time limit for test
                wordTimer.Stop() 'Resets timer
                wordTimer.Start()
            End If
        End If
    End Sub

    Sub CountUp()
        If startingTimer = "CountDown" Then 'If we are in the countdown pause mode before type game starts then ...
            If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
                timeLeft -= 0.1 'Decrements by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'Resets timer to 0 and starts typing game
                timeDoneLabel.Text = "GO"
                timeLeft = 0 'Starting time is 0 as we going from 0 to limit
                inputTextBox.ReadOnly = False  Lets user input in the box
                wordTimer.Start()
                startingTimer = "CountUp" 'Resets variable to countup mode for the test
            End If
        Else 'When the test has started
            If timeLeft < timeDuration Then 'Checks to see if time limit has been reached, if not then ..
                timeLeft += 0.1 'Increment timer by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
            Else 'When time limit has been reached then stop timer and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START"
                wordTimer.Stop() 'Resets timer
                MeasureTime()
            End If
        End If
    End Sub
```

Touch Typing Test

I have developed the Typing test by using counters. When the 'Start; button is pressed. Two main variables will be loaded. A string of the whole paragraph/random words (extract) and an array of words in the string (words). There are 4 main counters:

- countCharEssay: Keeps track of all correct letters typed
- countChar: Keeps track of correct characters typed on the current word
- countWord: Keeps track of what word the user is typing
- currentFaults: Keeps track of the currentFaults

The program will track the words that the player needs to currently. And make checks if the characters have been typed correctly and consecutively and update the counters accordingly. When the word has been completed. The countChar is reset to 0 and the countWord is incremented by 1. When the countWord is updated, it acts as an index position when the 'IntoChar' function is called to split the word/item corresponding to the array words[countWord] into a character array. This is what is used to check if the character typed is correct. If not correct, the currentFault counter in incremented and when corrected the counter is decremented. So when there are no mistakes, the counter = 0.

The essayRichTextBox forecolour is modified accordingly using the counters, countCharEssay and the currentFaults. Using the function 'TurnColourCharacters'.

```
    Sub TurnColourCharacters(start, lengthEnd, color) 'Function to changes a specific number of character's colours in the richtextbox
e.g when a letter goes green
        essayRichTextBox.SelectionStart = start 'Start of the text
        essayRichTextBox.SelectionLength = lengthEnd 'Length of the text to be selected
        essayRichTextBox.SelectionColor = color 'Forecolour that the selected text is changed to
    End Sub

    Sub TurnColourBackground(colourBack) 'Function to change the colour of the background
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub
```

```
    Sub TurnColourBackground(colourFont, colourBack) 'Function to change the colour of the background and forecolor of RichTextBox
        essayRichTextBox.ForeColor = colourFont
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

 Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'If whole word
is correct (wordlength = inputlength) and inputTextbox contains word currently being written
                countChar = 0 'Reset counter for the new word
                countCharEssay += 1 'Increment total charcters typed
                countWord += 1 'Increment counter to go to next word
                inputTextBox.Clear() 'Reset inputTextBox (so old word is cleared form textbox)
                If countChar = inputTextBox.TextLength And words.Length = countWord Then  'If all words are completed, then stop test
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Otherwise move onto next word in the array
                    charList = IntoChar(words, countWord)
                End If
            ElseIf (inputTextBox.Text.EndsWith(charList(countChar)) = True) And (countChar + 1 = inputTextBox.TextLength) Then 'checks
last character typed end with the inputted char and the lengths are the same
                TurnColourCharacters(countCharEssay, 1, Color.Green) 'Turn character green
                countChar += 1 'increment counters
                countCharEssay += 1
            ElseIf (inputTextBox.TextLength > countChar) Then 'If inputTextBox length is higher than correct charcters then there's a
mistake been made
                TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                currentFaults += 1 'increment fault counter
            End If
        Catch

        End Try
    End Sub
```

Keyboard

In this version, I decided to focus on the functionality of the Typing Test so did not develop any functionality of the Keyboard. Instead, I added the layout of the controls on the form seen below. I added the colours of the keys corresponding to the finger which should press that key.

Results

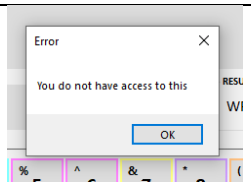
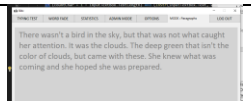
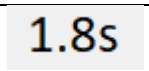
The results are calculated after a successful Typing Test which is done by calling the function ‘MeasureTime’. After the calculations, the labels are updated. And then an entry would be made into the database however, I cannot do that as I have not implemented the connection the database yet.

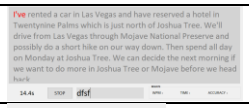
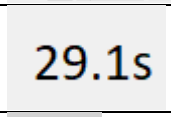
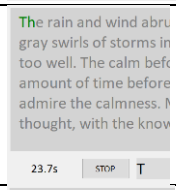
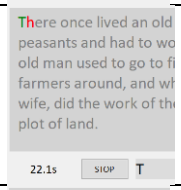
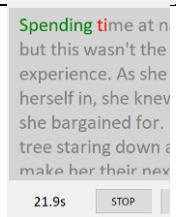
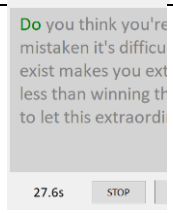
```
    Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished before time limit has
been reached
            timeTakenNow = timeLeft
```

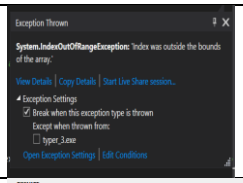
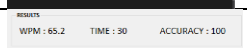
```
        Else 'countdown mode when game is finished before time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        accuracy = (100 - 100 * (totalFaults / countCharEssay))
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1) 'Add results to labels to show to the user
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested

Test	Result	Evidence	Comments
User can navigate to the WordFade form through the menu bar		N/A	
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar			Needs a database to inquire about how many entries have been made.
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		N/A	There is not a database for the access to be given to the user
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar			
User can navigate to the 'Admin' form when the criteria is met through the menu bar		N/A	
User can navigate to the 'Options' form through the menu bar		N/A	
Extract and Input Textbox is not editable		N/A	
Before a Typing Test has started, the user cannot input into the Input Textbox		N/A	
User can log out of the program		N/A	
An extract is loaded into the form when a Typing Test is about to start			It has been fetched from the text file
A count down from 3 starts, when at 0 the Typing Test starts every 0.1s			

When the Typing Test starts, the user is now able to input into the Input Textbox			
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'			Colour hasn't been changed to red
Input Textbox is now editable			
Timer is updated in a label every 0.1s until the end or is stopped			
If the character the user types is correct, change the forecolour for that letter on the extract to Green, keyboard key highlighted is updated			
If the character the user types is incorrect, change the forecolour for that letter on the extract to Red			
If the user has made errors and does not correct them, make all subsequent characters Red regardless of if they are correct, and do not move onto next word			
If the user has did not make any errors but backspaces on a Green character, change it to original colour			An 'IndexOutOfRangeException' is shown on Line 185, when I try to backspace a green letter at the end of the word
If the user has made errors, and backspaces to correct it, make the Red character now original colour			In addition, I have noticed that this may be due to the currentFault counter not decrementing only incrementing so therefore invalidates the selection length.
If the user tries to backspace to a previous word, stop at the start of the current word			
If the word is correctly typed completely and the user presses space onto the next word, clear the Input Textbox to make room for the new character			

If the extract has been completed, stop the Test, calculate and show the results			An 'IndexOutOfRangeException' is thrown on line 173
If the Time Limit is reached, stop the Test, calculate and show the results			An 'IndexOutOfRangeException' is shown on Line 185, when I try to type after a word at the same time the test ends. Additionally, the Accuracy does not work correctly. Additionally, I a whole test is carried out without any correct characters. An 'OverflowException' is done on line 148 due to dividing by 0.
Once results are calculated and the user is not in Guest Mode, input them into the database corresponding to the user		N/A	No database is connected for an entry to be made or not be made
Once results are calculated and the user is in Guest Mode, do not input results into the database		N/A	No database is connected for an entry to be made or not be made
If the 'Stop' button is clicked, stop the Typing Test		N/A	
If the database cannot be connected to input the results, send a prompt to the user			A database has not been connected to

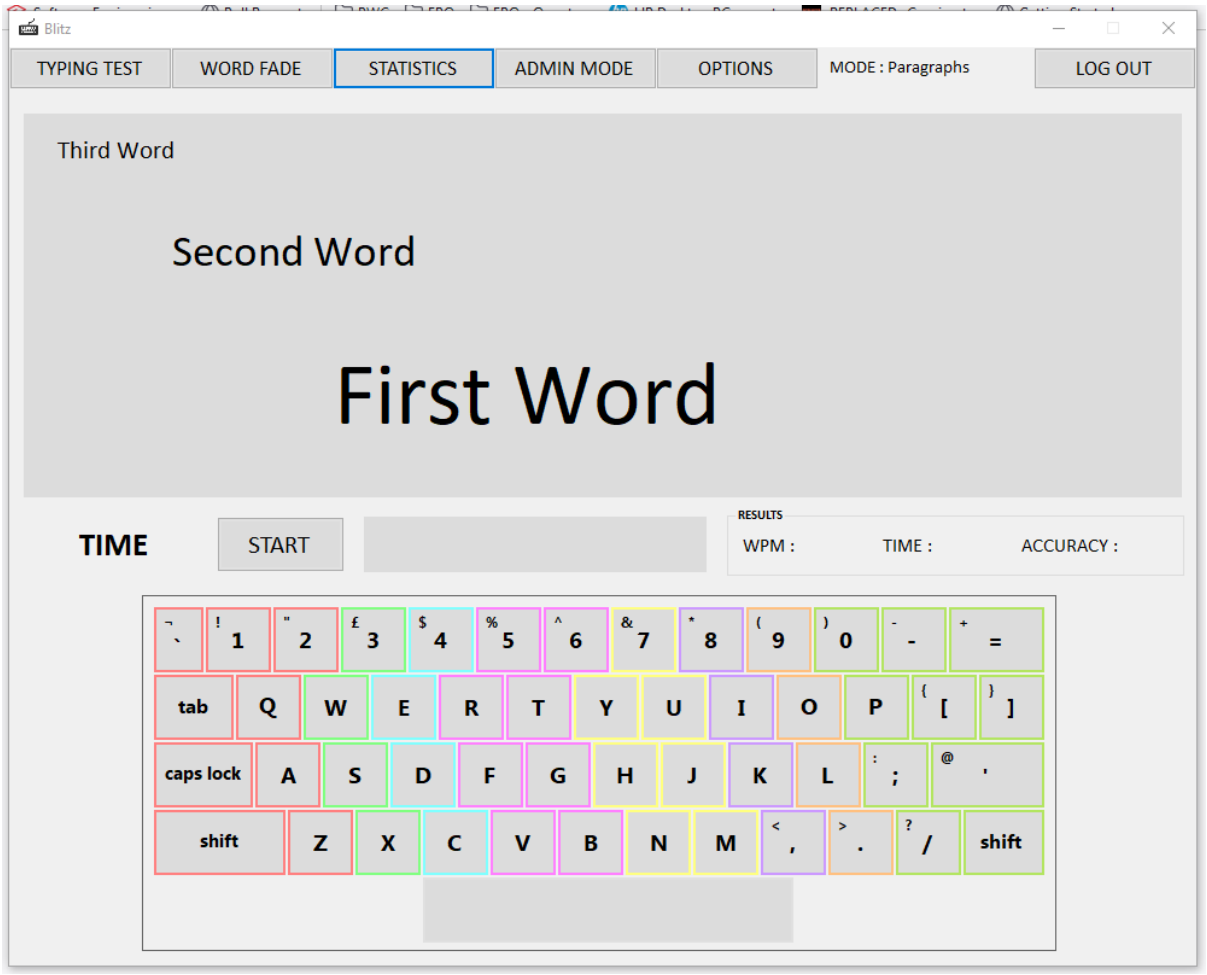
Remedial Actions for Next Version

I will add the connection to the database. This will mean that I can then add the functionality of inserting an entry into the database relating to the user logged in and whether the user is a Guest. Additionally, I will add in the functionality of the keyboard where the next character needed to be typed will be highlighted on the keyboard. I will also need to correct the bugs in the Typing Test where the colours corresponding to the characters typed is working erroneously. Another bug that I will need to address is when no correct inputs have been made, when calculating the results it will then throw an exception. I will address this by having a Try Function and will not insert an entry into the database. This is because the test will be erroneous and will not benefit the user when looking for improvements.

WordFade Form

Due to this Typing Test Form having the same functionality of the Typing Test and the focus of having the basic functionality of it nailed. I decided to implement it into the WordFade Form too. Once I have completed the Typing Test without any significant bugs I will then implement it.

I did add and name the controls which will be used seen below:



Testing

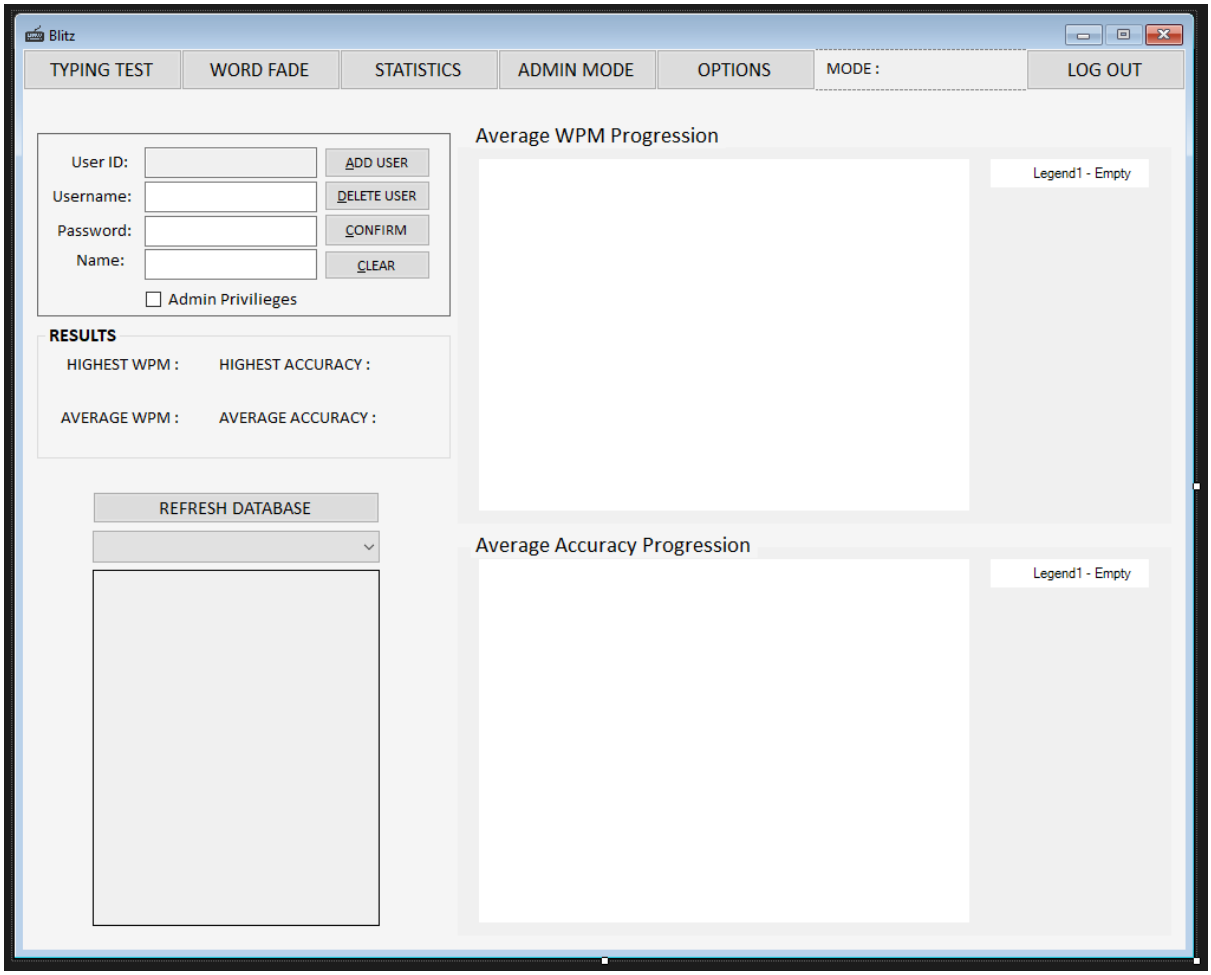
Due to the lack of functionality implemented in this iteration. I have decided that testing this iteration of the form would not be beneficial and would only waste time.

Remedial Actions for Next Version

Once the I am happy with the functionality of the Typing Test Form, I will integrate the logic into the WordFade Form. Otherwise, I will have double the number of bugs which will waste my time trying to track both forms and the fixes that are implemented in them.

Statistics Form

Due to my decision of not implementing the connection to that database. I could not develop any functionality into this form. However, I did manage to add and name the controls. A screenshot of the form is displayed below.



Testing

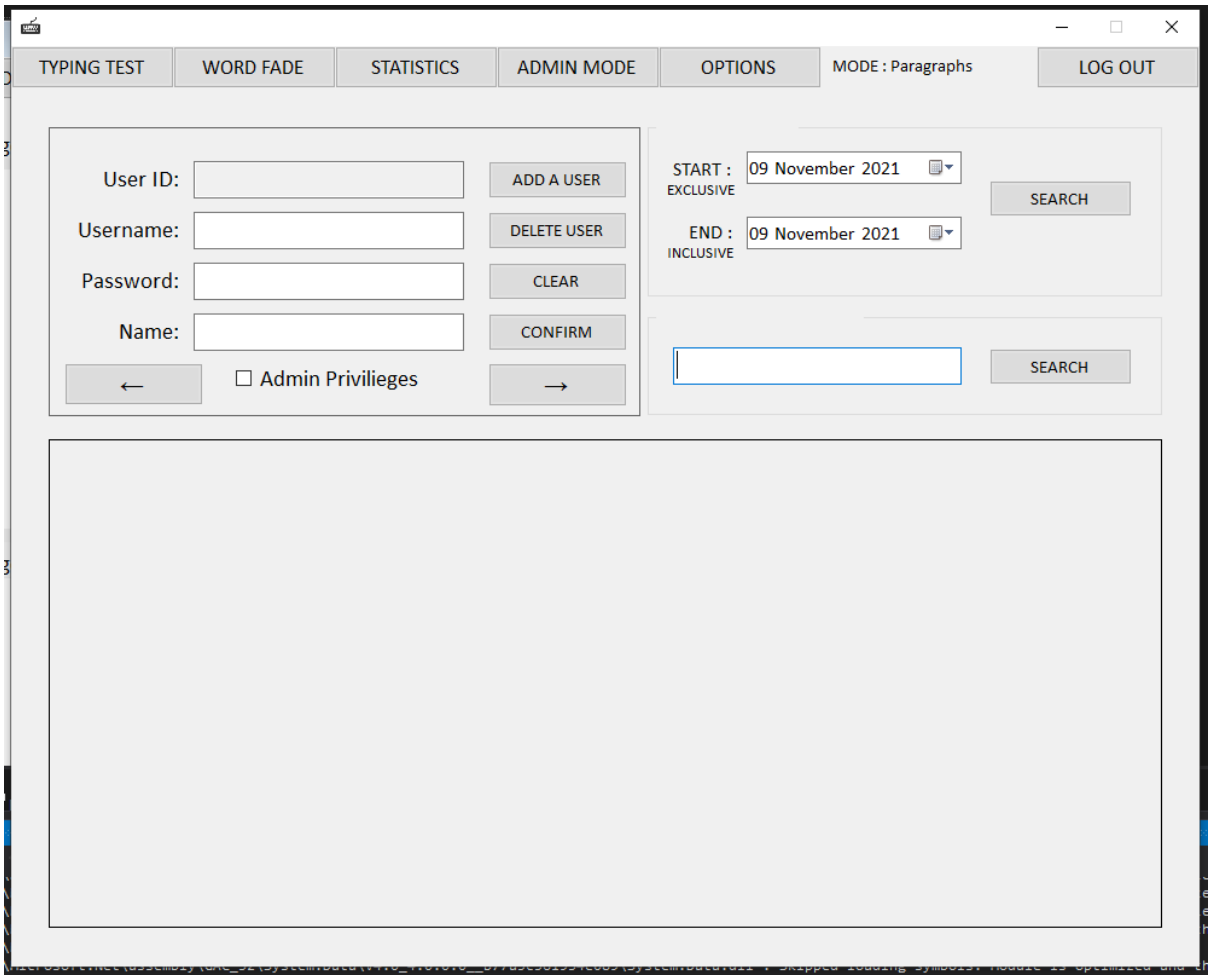
Due to the lack of functionality implemented in this iteration. I have decided that testing this iteration of the form would not be beneficial and would only waste time.

Remedial Actions for Next Version

In the next iteration, I will implement the connection to the database. Due to this, I can then add the basic functionality to the Statistics Form.

Admin Form

Due to my decision of not implementing the connection to that database. I could not develop any functionality into this form. However, I did manage to add and name the controls. A screenshot of the form is displayed below.



Testing

Due to the lack of functionality implemented in this iteration. I have decided that testing this iteration of the form would not be beneficial and would only waste time.

Remedial Actions for Next Version

In the next iteration, I will implement the connection to the database. Due to this, I can then add the basic functionality to the Statistics Form.

Options Form

For the Options form, I wanted to have the basic functionality for the ComboBoxes and passing the variables and modifying their values accordingly. The variables used are located in the Login Class which is the start up object and closing object. This means that I will have the guarantee, that the variables will always be available for use.

Updating ComboBoxes and CheckBoxes and Updating Option Variables

I have made a function called 'AddToComboBox' which adds the options from a string array, into the ComboBox when the form loads. Then a function called 'ChangeComboItem' is called. Which changes the option the ComboBox is on depending on the variable value linked to the ComboBox.

The same logic is applied to CheckBox functions called 'ChangeCheckBoxMode' and

```
Public Class optionsForm


    Dim timeDurationArray As String() = {"15", "30", "45", "60"} 'String arrays containing the options for the comboboxes
    Dim countTypeArray As String() = {"CountDown", "CountUp"}
    Dim extractTypeArray As String() = {"Paragraphs", "Random Words"}
    Dim wordFadeExtractTypeArray As String() = {"Paragraphs", "Random Words", "Alphabet"}

    Sub AddToComboBox(comboBoxName, itemArray) 'When form loads in, the string array items are added into a specified ComboBox
        comboBoxName.Items.Clear() 'Clears a ComboBox of previous items
        For Each item As String In itemArray
            comboBoxName.Items.Add(item)
        Next
    End Sub

    Sub ChangeComboItem(comboBoxName, itemArray, item) 'Changes the index of the ComboBox depending on pre-existing variable values
        comboBoxName.SelectedIndex = Array.IndexOf(itemArray, item)
    End Sub

    Sub ChangeCheckBoxMode(checkbox, mode) 'Changes the tick of the CheckBox depending on pre-existing variable values
        checkbox.Checked = mode
    End Sub

    Sub ChangeComboBoxKeyboard(comboBoxName, item) 'ComboBox keyboard options changes variable value
        Select Case item
            Case "ShowHighlight"
                comboBoxName.selectedIndex = 0
            Case "Show"
                comboBoxName.selectedIndex = 1
            Case "Nothing"
                comboBoxName.selectedIndex = 2
        End Select
    End Sub

    Function changeItem(comboBoxName) 'Changes variable value depending on option chosen
        Dim variable As String
        variable = comboBoxName.SelectedItem.ToString()
        Return variable
    End Function

    Private Sub optionsForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        AddToComboBox(countComboBox, countTypeArray) 'Adds items to ComboBoxes
        AddToComboBox(timeDurationComboBox, timeDurationArray)
        AddToComboBox(extractComboBox, extractTypeArray)
        AddToComboBox(wordFadeExtractComboBox, wordFadeExtractTypeArray)
        ChangeComboItem(countComboBox, countTypeArray, loginForm.countType) 'Changes selected items based on previous history in the
form
        ChangeComboItem(timeDurationComboBox, timeDurationArray, loginForm.timeDuration)
        ChangeComboItem(extractComboBox, extractTypeArray, loginForm.extractType)
        ChangeComboItem(wordFadeExtractComboBox, wordFadeExtractTypeArray, loginForm.extractWordFadeType)
        ChangeCheckBoxMode(suddenDeathCheckBox, loginForm.suddenDeath)
        ChangeCheckBoxMode(capsCheckBox, loginForm.capitals)
        ChangeCheckBoxMode(punctuationCheckBox, loginForm.punctuation)
        ChangeComboBoxKeyboard(keyBoardComboBox, loginForm.showKeyboard)
        loginForm.UpdateLabelMode(Me)
    End Sub
```

Passing Variables to Have a New Value

I logic behind passing variables between forms and modifying their value is the same throughout the different form is the same. For each ComboBox when the item selected changes, change the associated variable to the same value. The function 'changeItem' is when working with ComboBoxes. For Checkboxes, I have taken the value from the CheckBox and passed it into the Boolean variable.

For the font I have set up a font dialog for the user to easily navigate and select a new font family, size and style which is set as a Font Variable.

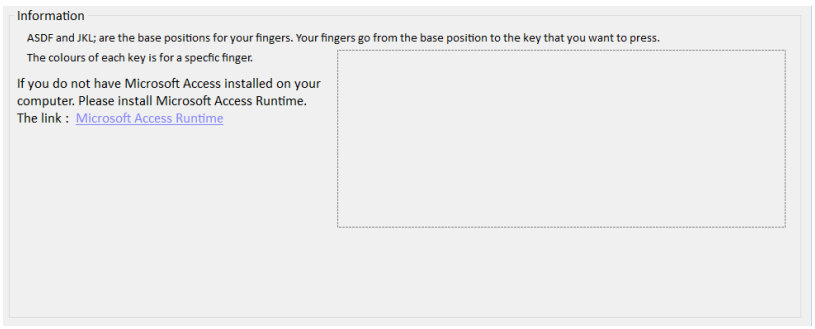
```
    Private Sub countComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles countComboBox.SelectedIndexChanged
        loginForm.countType = changeItem(countComboBox)
    End Sub

    Private Sub timeDurationComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
timeDurationComboBox.SelectedIndexChanged
        loginForm.timeDuration = changeItem(timeDurationComboBox)
    End Sub

    Private Sub extractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles extractComboBox.SelectedIndexChanged
        loginForm.extractType = changeItem(extractComboBox)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub suddenDeathCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles suddenDeathCheckBox.CheckedChanged
        loginForm.suddenDeath = suddenDeathCheckBox.Checked
    End Sub

    Private Sub capsCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles capsCheckBox.CheckedChanged
        loginForm.capitals = capsCheckBox.Checked
    End Sub

    Private Sub punctuationCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles punctuationCheckBox.CheckedChanged
        loginForm.punctuation = punctuationCheckBox.Checked
    End Sub

    Private Sub wordFadeExtractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
wordFadeExtractComboBox.SelectedIndexChanged
        loginForm.extractWordFadeType = changeItem(wordFadeExtractComboBox)
    End Sub

    Private Sub keyBoardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles keyBoardComboBox.SelectedIndexChanged
        Select Case keyBoardComboBox.SelectedIndex
            Case 0
                loginForm.showKeyboard = "ShowHighlight"
            Case 1
                loginForm.showKeyboard = "Show"
            Case 2
                loginForm.showKeyboard = "Nothing"
        End Select
    End Sub

    Private Sub changeFontBtn_Click(sender As Object, e As EventArgs) Handles changeFontBtn.Click
        If (changeFontDialog.ShowDialog() = DialogResult.OK) Then
            loginForm.globalFont = changeFontDialog.Font
        End If
    End Sub
```

Linked Label and Other Information

I have planned to add information about the solution and other helpful information to help aid the user in improving their Touch-Typing Skill. A key point that the sur should be aware of is how to get the database to work without Microsoft Access being downloaded on the system. This is a driver as explained in my analysis and I have said this to be a link label. I have made it so that when the link label is pressed, the default browser opens to this link https://support.microsoft.com/en-us/office/download-and-install-microsoft-365-access-runtime-185c5a32-8ba9-491e-ac76-91cbe3ea09c9.

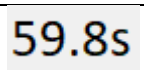
```
    Private Sub linkLabel_LinkClicked(sender As Object, e As LinkLabelLinkClickedEventArgs) Handles linkLabel.LinkClicked
        Dim address = "https://support.microsoft.com/en-us/office/download-and-install-microsoft-365-access-runtime-185c5a32-8ba9-491e-
ac76-91cbe3ea09c9"
        linkLabel.LinkVisited = True
        System.Diagnostics.Process.Start(address)
    End Sub
```

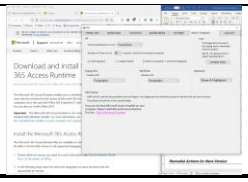
I have added information regarding the positioning of fingers on keys.



Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested

Test	Results	Evidence	Comments
All pre-existing settings are loaded into the controls		N/A	
User can navigate to the WordFade form through the menu bar		N/A	
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar			Needs a database to inquire about how many entries have been made.
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		N/A	There is not a database for the access to be given to the user
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar		N/A	
User can navigate to the 'Admin' form when the criteria is met through the menu bar		N/A	
User can navigate to the 'Typing Test' form through the menu bar		N/A	
User can log out of the program		N/A	
When the type of extract for Typing Test ComboBox is changed, it will be changed in the Typing Test			
When the type of extract for WordFade ComboBox is changed, it will be changed in the WordFade test		N/A	The WordFade form's functionality has not been implemented yet
When the CountDown or CountUp ComboBox is changed, it will be changed in the Typing Test and WordFade		N/A	The works in Typing Test and not WordFade as the functionality has not been implemented
When the Time Duration ComboBox is changed, it will be changed in the Typing Test and WordFade			The works in Typing Test and not WordFade as the functionality has not been implemented

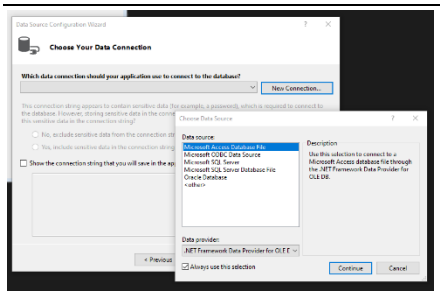
When the Capitals checkbox is changed, it is changed for Typing Test and WordFade		N/A	Functionality not been implemented into the Typing Test
When the Punctuation checkbox is changed, it is changed for Typing Test and WordFade (this only applies to 'Paragraph' extracts		N/A	Functionality not been implemented into the Typing Test
When the Sudden Death checkbox is changed, it is changed for Typing Test and WordFade		N/A	Functionality not been implemented into the Typing Test
When the Keyboard ComboBox has changed, it will be changed in the Typing Test and WordFade		N/A	Keyboard's functionality not been implemented into the Typing Test
User can change the font type, style and size for the Typing Test and WordFade		N/A	Functionality has not been implemented
User can click a link to be directed to a website to download Microsoft Access Runtime			

Remedial Actions for Next Version

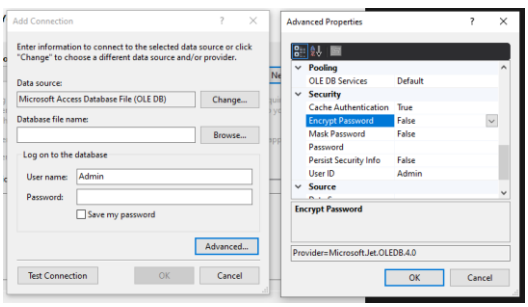
I will need to add more helpful information about the keyboard and the solution to aid the user. Additionally I need to make sure that in the Typing and WordFade Form, there is functionality for the options chosen to have a significant part in the user's experience.

Second Version

Connecting to the Database



In order to connect to the database, Visual Studio has features of adding Data Sources where you can implement a connection to a specific database. You are also given a connection string in order for the program to know the location of the file. There was also an option to add a login into the database, and options to encrypt and add security information. I decided against adding this in as I would not have enough time to fully implement the security into my solution and I was aware that I need to focus on the core element of my solutions that do not have any current functionality yet.



With the connection string, I was given It was `"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=”C:\Users\[myUserName]\OneDrive\A Level\Computer Science\Project\Version 2\userDatabase.accdb"`. Even thought that this is a valid method for connection to the database. I

recognised that this would not account for other users on different accounts with different files paths. Instead it would return an error. Therefore, I changed this to `"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\userDatabase.accdb"`. This way the executable only had to be in the same Directory as the database file.

Login Form

Added more Variables

I have added the 'showKeyboard' variable and variables containing the colours of the keys. Additional to that I've added a global font to be used across the Typer and WordFade Form.

```
Public Class loginForm 'All friend variables are global that will be used across the form

    Friend connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\userDatabase.accdb" 'Connection string used
to connect to the database
    Private conn As New OleDbConnection
    Friend dataUsername As String  'Current user logged in credentials are kept in the variables
    Friend dataName As String
    Friend dataAdmin As Boolean
    Friend Shared dataUserID As String
    Friend timeDuration = "30" 'Option variables to be passed into various forms
    Friend countType = "CountDown"
    Friend suddenDeath = False
    Friend extractType = "Paragraphs"
    Friend extractWordFadeType = "Paragraphs"
    Friend capitals = True
    Friend punctuation = True
    Friend showKeyboard = "ShowHighlight"
    Friend globalFont As New Font("Calibri", 28, Font.Style.Regular)
    Friend entriesPresent = False 'Used to determine whether user can enter the Statistics form
    Friend attemptCount = 3 'Used as a counter for the attempts until applicatoin closes
    Friend red As Color = Color.FromArgb(255, 128, 128) 'Used as colours for keyboard
    Friend lightGreen As Color = Color.FromArgb(128, 255, 128)
    Friend lightBlue As Color = Color.FromArgb(128, 255, 255)
    Friend purple As Color = Color.FromArgb(255, 128, 255)
    Friend yellow As Color = Color.FromArgb(255, 255, 128)
    Friend blue As Color = Color.FromArgb(204, 153, 255)
    Friend orange As Color = Color.FromArgb(255, 192, 128)
    Friend darkGreen As Color = Color.FromArgb(179, 229, 97)
```

SQL for the solution

In order to find whether the user has any entries on the account I have implemented this function. This is mainly for the navigation into the Statistics form.

```
a a a a a a a a a a a a
    Sub GetAmountEntries() 'Checks the amount of entries, with that user to stop any database errors
        conn.Open()
        Dim SQLGetEntries = "SELECT Count(entryData.EntryID) AS [Entries] FROM userData INNER JOIN entryData ON userData.UserID =
entryData.UserID GROUP BY userData.Username HAVING (userData.Username)='" & dataUsername & "';"
        Dim commandGetEntries = New OleDbCommand(SQLGetEntries, conn)
        If commandGetEntries.ExecuteScalar() Is Nothing Or commandGetEntries.ExecuteScalar() = "1" Then
            entriesPresent = False
        Else
            entriesPresent = True
        End If
        conn.Close()
    End Sub
```

Logging into the database

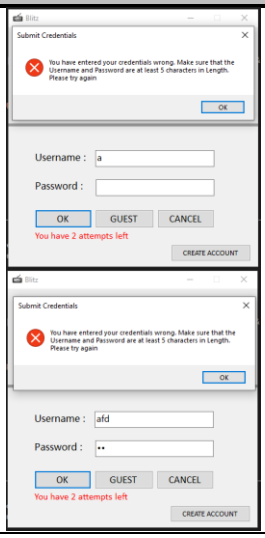
I have implemented SQL commands to check whether the username is present in the database. If so fetch the password and check the Database password against the inputted Password. If correct that check the record for Admin privileges and update the variables accordingly.

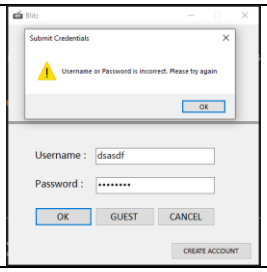
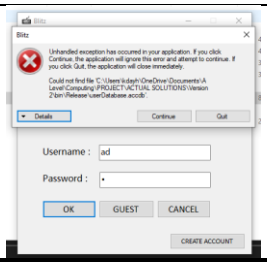
```
a a a a a a a a a a a a
    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Try
            conn.Open()
            If usernameTextBox.Text.Trim = "" Or passwordTextBox.Text.Trim = "" Or usernameTextBox.TextLength < 5 Or
passwordTextBox.TextLength < 5 Then 'Validation : Checks to see if characters are in either TextBox
                MessageBox.Show("You have entered your credentials wrong. Make sure that the Username and Password are at least 5
```

```
characters in Length. Please try again", "Submit Credentials", MessageBoxButtons.OK, MessageBoxIcon.Error)
                conn.Close()
            Else
                Dim inputUsername As String
                Dim inputPassword As String
                Dim dataPassword As String
                inputUsername = usernameTextBox.Text.Trim 'Assiging User Inputs to variables, Trim Function takes out spaces
                inputPassword = passwordTextBox.Text.Trim
                Dim SQLpassword As String = "SELECT Password FROM [userData] WHERE Username= '" & inputUsername & "';" 'Formatting SQL
Code to fetch password where username matches
                Dim SQLadmin As String = "SELECT Admin FROM userData WHERE Username= '" & inputUsername & "';" 'SQL code to fetch Admin
Privileges
                Dim SQLuserID As String = "SELECT [UserID] FROM [userData] WHERE Username='" & inputUsername & "';"
                Dim commandPassword As New OleDbCommand(SQLpassword, conn)
                Dim commandAdmin As New OleDbCommand(SQLadmin, conn)
                Dim commandUserID As New OleDbCommand(SQLuserID, conn)
                dataPassword = commandPassword.ExecuteScalar()
                If dataPassword Is Nothing Or dataPassword <> inputPassword Then 'Checks if SQL fetch password fetches nothing (due to
wrong username) or does not match
                    MessageBox.Show("Username or Password is incorrect. Please try again", "Submit Credentials", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
                    ClearTextBox()
                    conn.Close()
                    deductAttempt() 'minus attempt from list
                ElseIf dataPassword.ToString = inputPassword Then 'If the passwords match and are correct
                    dataUsername = inputUsername
                    dataUserID = commandUserID.ExecuteScalar()
                    Dim adminCheck = commandAdmin.ExecuteScalar()
                    Select Case adminCheck 'Checks admin privileges
                        Case "Yes"
                            dataAdmin = True
                        Case "No"
                            dataAdmin = False
                    End Select
                    ClearTextBox()
                    conn.Close()
                    attemptCount = 3 'reset counter
                    attemptsLabel.Visible = False
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Hide() 'Hides form and does not close as this is the startup form and need variables associated with it
                End If
            End If
        Catch ex As NullReferenceException
            MessageBox.Show("Something went wrong, please try again", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
            ClearTextBox()
            conn.Close()
        End Try
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 are not shown here.

Test	Result	Evidence	Comments
User will not be able to log in unless both input Textboxes are inputted. And if the text is beneath 5 characters.			

When wrong credentials are inputted, it is not a successful login and will not be allowed in			
When the credentials are correct, the login is successful. The 'Typing Test' form opens		N/A	If database connection is not present, it informs the user
If database connection is not present, it informs the user			I compiled a released version and deleted the Microsoft Access file from the program folder. I then ran the program and tried to log into this. This caused the program to crash.

Remedial Actions for Next Version

I need to implement functionality where the program checks for the connection and informs the user accordingly. Additionally, I would like for specific features (Statistics Form and the Admin Form) to be restricted from the user if the database connection cannot be made and put into Guest mode instead.

Add User Form

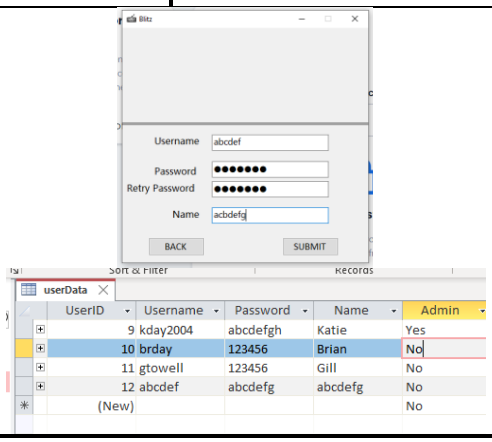
Adding a User

I have implemented SQL commands to insert the user into the database. This only occurs when the validating requirement have been met. The validation implemented is where all inputs are presents and the username and password are above or equal to 5 characters.

```
a a a a a a a a a a a
    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Dim result As DialogResult
        If passTextBox.Text = retryPassTextBox.Text And userTextBox.Text.Trim.Length <> 0 And passTextBox.Text.Trim.Length <> 0 And
nameTextBox.Text.Trim.Length <> 0 Then 'Validation : Checking password entries match and that no textboxes and no characters in them
            result = MessageBox.Show("Are you sure?", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then
                    Dim username, password, myName, admin As String
                    username = userTextBox.Text.Trim
                    password = passTextBox.Text.Trim
                    myName = nameTextBox.Text.Trim '
                    admin = "No"
                    Dim sqlAddUser As String = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username,
@password, @name, @admin);" 'SQL for inserting crednetials for new record
                    Dim commandInsert As New OleDbCommand(sqlAddUser, conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteNonQuery() 'Execute command to database
                    MessageBox.Show("User has been added")
                    conn.Close() 'Closes database connection
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            Catch ex As Exception 'Exception if database does not connect or something goes wrong
                MessageBox.Show("Something has gone wrong. Please try again.")
            End Try
        Else
            MessageBox.Show("You have entered your credentials wrong.  Make sure that the Username and Password are at least 5
characters in length. Please try again", "Submit Credentials")
        End If
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 are not shown here.

Test	Result	Evidence	Comments
If the Username wanting to be added is already a pre-existing login then prevent from adding it again		N/A	Function not implemented
If database connection is not present, it informs the user		N/A	Function not implemented
When the MessageBox shows after submitting, if the clicked 'Yes' then the account is added			

Remedial Actions for Next Version

I need to add the remaining validation into the f-form as it relates to my success criteria.

Typing Test Form

Modifying the Essay

When the user customises their experience, there should be functionality implemented to deal with it. In this case, I have added functionality into 'FetchRandomParagraphs' for the Paragraph extracts, which can strip the Punctuation and Capitals that are already in the paragraphs.

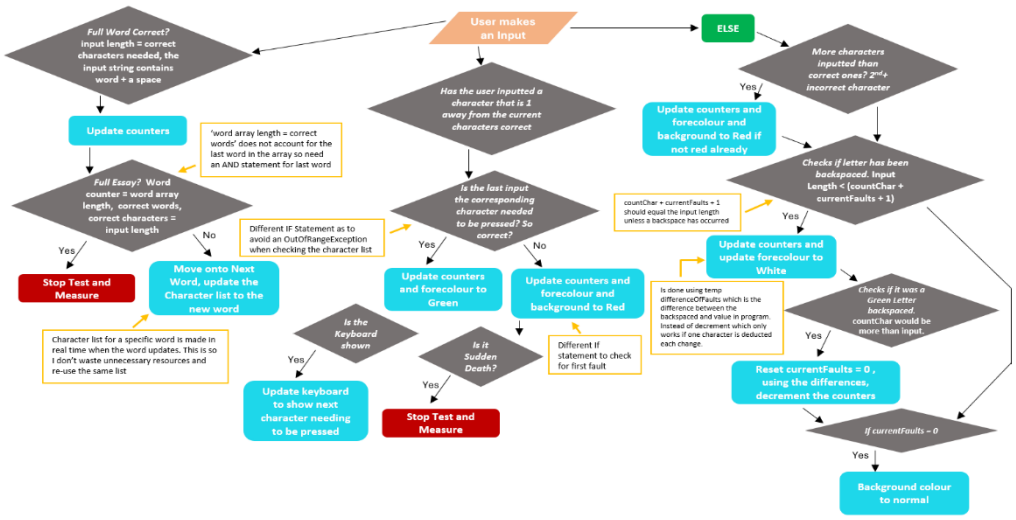
```
a a a a a a a a a a a a

    Function FetchRandomParagraphs() As String 'Fetch paragraphs from textfile
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt") 'Add each line as a
string in an array
        Dim essayExtract As String
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Int32
        randomNumber = Generator.Next(0, lineCount) 'Generate a random number in the range of 1 to the amount of lines in the array
        essayExtract = lines.ElementAt(randomNumber).ToString 'Pass the string at the random index value into the varaibles and return
the variable
        If loginForm.punctuation = False Then 'Checks if need to strip string of punctuation (currently has punctuation added)
            essayExtract = Regex.Replace(essayExtract, "[ ](?=[ ])|[^-_A-Za-z0-9 ]", String.Empty) 'function to strip punctuation
        End If
        If loginForm.capitals = False Then 'Checks if captials letter is False (current has captials added)
            Return essayExtract.ToLower()
        Else
            Return essayExtract
        End If
    End Function
```

For the 'FetchRandomWords' function, I have added a system where when after the word array has been created with 50 words being passed into from the text file. The variable 'captials' is True, then the 'WithCaps' function is called. This is an iterative function which generates a random number between 0 to 2. The 'CapsFirstLetter' is called which splits the current item in the word array into a character array and if the random number generated equals 0 (1/3 chance) then the first item (letter) in the character array is changed to be upper case. The character array is then convert to a string and is return the word array's current item.

```
a a a a a a a a a a a

    Function FetchRandomWords() 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\randomWords.txt") 'initialises on array
on the words on each line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50 'Will add 50 words to the array
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            If loginForm.capitals = True Then 'If statement checks whether to add Captials to the words (they are all currently lower
case)
                wordList = WithCaps(wordList)
            End If
            Return wordList
        End If
    End Function

    Function WithCaps(wordList As String()) 'Adds capitals in random words mode
        Dim Generator As System.Random = New System.Random()
        For i = 0 To wordList.Length - 1
            Dim randomNumber = Generator.Next(0, 2) 'a random number is generated
            wordList(i) = CapsFirstLetter(wordList, i, randomNumber)
        Next
        Return wordList
    End Function

    Function CapsFirstLetter(wordList, i, randomNumber) 'Called by 'WithCaps' to capatlise the first letter of the word
        Dim characterArray() As Char = wordList(i).ToCharArray
        If randomNumber = 0 Then 'Making chances of Captilisation 1/2 chance from the random number
            characterArray(0) = Char.ToUpper(characterArray(0))
        End If
        Return New String(characterArray)
    End Function
```

Touch Typing Test

I have significantly changed the functionality of the Typing Test and when it registers a correct and incorrect entry. This is because I wanted the program to update counters when the user backspaces too little or too much. I have made this flow chart to represent more accurately each If statement and the meaning behind it.



The main logic is the same, however of there are separate sections for if the user backspaces and a better way of updating the counters in comparison to previously. (The yellow boxes are just for explanation).

```
a a a a a a a a a a a
    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'Checks if full
word is correct and the user has pressed spaces to go onto next word
                countChar = 0
                countCharEssay += 1
                countWord += 1
                essayRichTextBox.SelectionStart = countCharEssay + 1
                inputTextBox.Clear() 'Resets input box
                If countChar = inputTextBox.TextLength And words.Length = countWord Then 'Checks if full essay has been completed
                    timeDoneLabel.Text = "STOP"
                    startBtn.Text = "START"
                    inputTextBox.ReadOnly = True
                    inputTextBox.Clear()
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Goes onto next word
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                End If
            ElseIf (countChar + 1 = inputTextBox.TextLength) And (countChar <> charList.Length) And currentFaults = 0 Then 'Checks if it
is next inputted letter compared to input, and there are no currentFaults (in case the user backspaces)
                If (inputTextBox.Text.EndsWith(charList(countChar)) = True) Then 'This is put here to avoid an OutOfRangeException,
checks after the range conditioms are met
                    TurnColourCharacters(countCharEssay, 1, Color.Green)
                    countChar += 1
                    countCharEssay += 1
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Else 'Checks 1st incorrect character
                    currentFaults += 1
                    totalFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    essayPanel.BackColor = Color.IndianRed
                    essayRichTextBox.BackColor = Color.IndianRed
                    If loginForm.suddenDeath = True Then
                        timeDoneLabel.Text = "STOP"
                        startBtn.Text = "START"
                        inputTextBox.ReadOnly = True
                        inputTextBox.Clear()
                        startType = False
                        wordTimer.Stop()
                        ChangeAllKeyColourGrey()
                    End If
                End If
            Else
                If (inputTextBox.TextLength > (countChar)) Then 'Checks if any inputted letter are incorrect, checks the 2nd+ incorrect
character
                    totalFaults += 1
                    currentFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    If essayPanel.BackColor <> Color.IndianRed Then
                        essayPanel.BackColor = Color.IndianRed
                        essayRichTextBox.BackColor = Color.IndianRed
                    End If
                End If
                If inputTextBox.TextLength < (countChar + currentFaults + 1) Then 'Checks if a letter has been backspaced
                    differenceOfFaults = (countChar + currentFaults) - inputTextBox.TextLength 'Used difference instead of decrement,
mutliple charcters can be dedcuted at once (highlighting)
                    currentFaults -= differenceOfFaults
                    TurnColourCharacters(countCharEssay + currentFaults, differenceOfFaults, Color.Black)
                    If inputTextBox.TextLength < (countChar) And currentFaults <= 0 Then 'Checks if a green letter has been backspaced,
countChar would e more than the input
                        currentFaults = 0 'Do not want it to become -1
                        countChar -= (countChar - inputTextBox.TextLength) 'subtracts from counts using the different between values
                        countCharEssay -= (countChar - inputTextBox.TextLength + 1)
                        If loginForm.showKeyboard = "ShowHighlight" Then
                            ChangeKeyColour(countChar) 'changes key highlighted to previous key
                        End If
                    End If
                    If currentFaults = 0 Then
                        essayPanel.BackColor = Color.Gainsboro
                        essayRichTextBox.BackColor = Color.Gainsboro
                    End If
                End If
            End If
        Catch
        End Try
    End Sub
```

Keyboard

I have implemented the keyboard functionality as procedures which are called upon in the previous extract of code. The 'ChangeKeyColour' procedure highlights (by changing the back colour) a specific panel depending on the next character needing to be pressed. It is hard coded by using Select Case statements. The 'ChangeAllKeyColourGrey' procedure changes all the colours back the normal background. I did not have time to implement a method of checking the character pressed before so therefore instead I made it so all the keys would revert ack to its default colour.

```
a a a a a a a a a a a

    Sub ColourPanel(color, Panel) 'Function to Colour the Panel
        Panel.BackColor = color
    End Sub

    Sub ChangeKeyColour([char]) 'Checks the next key needed to be pressed and highlights key on keyboard
        Dim currentChar As String
        ChangeAllKeyColourGrey() 'Chagnes previous keys too all grey
        If charList.Length > ([char]) Then 'So that there is not a Out of Range Exception with charlist([char])
            currentChar = charList([char])
            Select Case currentChar.ToLower
                Case "a"
                    ColourPanel(loginForm.red, aPanel)
                Case "b"
                    ColourPanel(loginForm.purple, bPanel)
                Case "c"
                    ColourPanel(loginForm.lightBlue, cPanel)
                Case "d"
                    ColourPanel(loginForm.lightBlue, dPanel)
                Case "e"
                    ColourPanel(loginForm.lightBlue, ePanel)
                Case "f"
                    ColourPanel(loginForm.purple, fPanel)
                Case "g"
                    ColourPanel(loginForm.purple, gPanel)
                Case "h"
                    ColourPanel(loginForm.yellow, hPanel)
                Case "i"
                    ColourPanel(loginForm.blue, iPanel)
                Case "j"
                    ColourPanel(loginForm.yellow, jPanel)
                Case "k"
                    ColourPanel(loginForm.blue, kPanel)
                Case "l"
                    ColourPanel(loginForm.orange, lPanel)
                Case "m"
                    ColourPanel(loginForm.yellow, mPanel)
                Case "n"
                    ColourPanel(loginForm.yellow, nPanel)
                Case "o"
                    ColourPanel(loginForm.orange, oPanel)
                Case "p"
                    ColourPanel(loginForm.darkGreen, pPanel)
                Case "q"
                    ColourPanel(loginForm.red, qPanel)
                Case "r"
                    ColourPanel(loginForm.purple, rPanel)
                Case "s"
                    ColourPanel(loginForm.lightGreen, sPanel)
                Case "t"
                    ColourPanel(loginForm.purple, tPanel)
                Case "u"
                    ColourPanel(loginForm.yellow, uPanel)
                Case "v"
                    ColourPanel(loginForm.purple, vPanel)
                Case "w"
                    ColourPanel(loginForm.lightGreen, wPanel)
                Case "x"
                    ColourPanel(loginForm.lightGreen, xPanel)
                Case "y"
                    ColourPanel(loginForm.yellow, yPanel)
                Case "z"
                    ColourPanel(loginForm.red, zPanel)
                Case Else
            End Select
            If Char.IsUpper(charList([char])) = True Then 'Checks if letter is a capital
                ColourPanel(loginForm.red, shiftLeftPanel)
                ColourPanel(loginForm.darkGreen, shiftRightPanel)
            End If
        ElseIf [char] = charList.Length Then 'When end of a word press space
            ColourPanel(Color.Silver, spacePanel)
        End If
    End Sub

    Sub ChangeAllKeyColourGrey() 'Resets entire keyboard colour to 'Gray'
        ColourPanel(Color.Gainsboro, aPanel)
        ColourPanel(Color.Gainsboro, bPanel)
        ColourPanel(Color.Gainsboro, cPanel)
        ColourPanel(Color.Gainsboro, dPanel)
        ColourPanel(Color.Gainsboro, ePanel)
        ColourPanel(Color.Gainsboro, fPanel)
        ColourPanel(Color.Gainsboro, gPanel)
```

```
        ColourPanel(Color.Gainsboro, hPanel)
        ColourPanel(Color.Gainsboro, iPanel)
        ColourPanel(Color.Gainsboro, jPanel)
        ColourPanel(Color.Gainsboro, kPanel)
        ColourPanel(Color.Gainsboro, lPanel)
        ColourPanel(Color.Gainsboro, mPanel)
        ColourPanel(Color.Gainsboro, nPanel)
        ColourPanel(Color.Gainsboro, oPanel)
        ColourPanel(Color.Gainsboro, pPanel)
        ColourPanel(Color.Gainsboro, qPanel)
        ColourPanel(Color.Gainsboro, rPanel)
        ColourPanel(Color.Gainsboro, sPanel)
        ColourPanel(Color.Gainsboro, tPanel)
        ColourPanel(Color.Gainsboro, uPanel)
        ColourPanel(Color.Gainsboro, vPanel)
        ColourPanel(Color.Gainsboro, wPanel)
        ColourPanel(Color.Gainsboro, xPanel)
        ColourPanel(Color.Gainsboro, yPanel)
        ColourPanel(Color.Gainsboro, spacePanel)
        ColourPanel(Color.Gainsboro, zPanel)
        ColourPanel(Color.Gainsboro, shiftLeftPanel)
        ColourPanel(Color.Gainsboro, shiftRightPanel)
    End Sub
```

Additionally, I have added an option to disable the highlighting and the keyboard showing entirely. This part of the code is within the Touch Typing Test section for highlighting and the showing of the Keyboard is checked when the form is loaded in. This is shown below.

```
a a a a a a a a a a a

    Private Sub typerForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        conn.ConnectionString = loginForm.connectionString
        If loginForm.showKeyboard = "Nothing" Then
            keyBoardPanel.Visible = False 'hiding the keyboard from veiw
        End If
        essayRichTextBox.Font = loginForm.globalFont ‘setting the font
        conn.Open()
    End Sub
```

Font

The font is modified using a font dialog and is passed into the Typer Form. The code above is where the textbox is changed to the new font.

Results

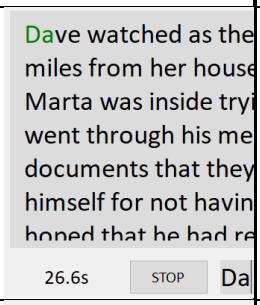
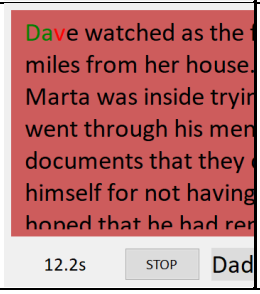
I have added SQL where an entry will be made into the database, linking to the specific user, using the Primary UserID in userData as a Foreign Key in entryData. This will only be carried out when not in Guest Mode.

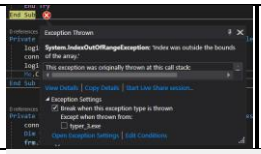
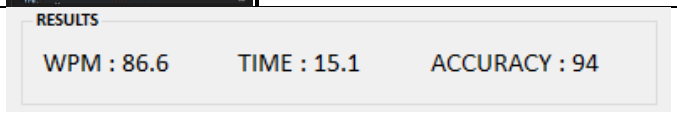
```
a a a a a a a a a a a

Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished before time limit has
been reached
            timeTakenNow = timeLeft
        Else 'countdown mode when game is finished beofre time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        accuracy = (100 - 100 * (totalFaults / countCharEssay))
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1) 'Add results to labels to show to the user
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)

        If loginForm.dataUsername <> "Guest" Then 'Adding the entry into the database is not an option for a Guest user
            Dim SQLinsertentry = "INSERT INTO [entryData] ([UserID], [Date], [WPM], [Accuracy], [Mode], [Time Duration], [Capital
Letters], [Punctuation]) VALUES (@userID, @date, @wpm, @accuracy, @mode, @timeduration, @capitals, @punctuation)"
            Dim commandInsertEntry As New OleDbCommand(SQLinsertentry, conn) 'SQL for adding in the new entry in [entryTable]
            commandInsertEntry.Parameters.AddWithValue("@userID", loginForm.dataUserID)
            commandInsertEntry.Parameters.Add("@date", OleDbType.Date).Value = Date.Today()
            commandInsertEntry.Parameters.AddWithValue("@wpm", Math.Round(wordsPerMinute, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@accuracy", Math.Round(accuracy, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@mode", loginForm.countType.ToString)
            commandInsertEntry.Parameters.AddWithValue("@timeduration", loginForm.timeDuration.ToString)
```

```
            Select Case loginForm.capitals 'checks what captials variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "Yes")
                Case Else
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "No")
            End Select
            Select Case loginForm.punctuation 'checks what punctuation variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@punctuation", "Yes")
                Case Else
                    Select Case loginForm.extractType 'Undefined and No depends on whether it Random Words or Essay (Puncutation option
only affects Essay)
                        Case "CountDown"
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "No")
                        Case Else
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "UnDefined")
                    End Select
            End Select
            Try
                commandInsertEntry.ExecuteNonQuery()
            Catch e As Exception
                MessageBox.Show(e.ToString)
            End Try
        End If
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 are not shown here.

Test	Result	Evidence	Comments
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar			
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		N/A	
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'			Did not change to red
If the user has did not make any errors but backspaces on a Green character, change it to original colour			
If the user has made errors, and backspaces to correct it, make the Red character now original colour			

If the extract has been completed, stop the Test, calculate and show the results			IndexOutOfRangeException triggered
If the Time Limit is reached, stop the Test, calculate and show the results			
Once results are calculated and the user is not in Guest Mode, input them into the database corresponding to the user			
Once results are calculated and the user is in Guest Mode, do not input results into the database		N/A	
If the database cannot be connected to input the results, send a prompt to the user		N/A	No functionality implemented

Remedial Actions for Next Version

For the next version, I will try to implement a way for the program to check whether a connection to the database can be initialised. Otherwise, if the program tries to make a connection, an exception will be thrown, and the program will crash.

WordFade Form

Due to this Typing Test Form having the same functionality of the Typing Test and the focus of having the basic functionality of it nailed. I decided to implement it into the WordFade Form too. Once I have completed the Typing Test without any significant bugs I will then implement it.

Testing

Due to the lack of functionality implemented in this iteration. I have decided that testing this iteration of the form would not be beneficial and would only waste time.

Remedial Actions for Next Version

I am now happy with the main functionality of the Typing Test being completed. For the next iteration, I will implement the functionality into the WordFade form. This is because there is no functionality currently present in the form apart from controls that do on do anything.

Statistics Form

Adding and Editing Own User Account

When the form loads in, the users login credentials are fetched from the database and inserted into textboxes. The procedures which does this are called 'GetRowNumber' to fetch the record and 'GetUserDetails' to fetch display the credentials in the textboxes. They will be able to view 'UserID, 'Username', 'Password' and 'Name'. However, they will not be able to edit the 'UserID'. They can make edits and save them when pressing the 'Confirm' button. This is done by taking the inputs made by the user and using an 'UPDATE' SQL Statement to the user's record. I have also included validation against the username being the same as a current user. Using the 'previousUsername' variable and an SQL statement to check if the account is already present.

There is also an Add User mode. Where when pressing the 'Add User' button, the mode is signified by the button changing to 'Cancel Add'. The textboxes are cleared, and the user can input credentials for a new user. I have added is validation for no inputs and the username and password is below 5 characters in length. To add the users into the database I have used an 'INSERT' SQL statement. However, I have not yet implemented the functionality of when adding a new user, the program then completely logs the user out to then log in again. I want to add this as it may be confusing on how is logged in and who is using the account.

The delete button uses a 'DELETE' SQL statement to remove a record. However, I have not added any functionality where after deletion the user is logged out. This is very important as a user is logged into an account which now does not exist. This would cause issues regarding database integrity.

```
    Function GetRowNumber(idNumber) 'To find the rowNumber (record number) of the user in the table
        Dim SQLGetUserRow = "SELECT * FROM userData" 'Fetches whole table
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUserRow, conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        Dim rowNum As Integer
        rowNum = 0
        For Each row As DataRow In dataset.Tables("userData").Rows
            If row.Item("UserID") = Convert.ToInt32(idNumber) Then 'If the UserID in record matches UserID then return Row
                Exit For
            End If
            rowNum += 1
        Next row
        Return rowNum
    End Function

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes
        conn.Open()
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(0).Item(0) 'Row 0 as its  only 1 record fetched,
        usernameTextBox.Text = dataset.Tables("userData").Rows(0).Item(1) 'Item 'number' to signify the column value
        passwordTextBox.Text = dataset.Tables("userData").Rows(0).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(0).Item(3)
        If dataset.Tables("userData").Rows(0).Item(4) = "Yes" Then 'Changes checkbox value depending on value in record
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        conn.Close()
        previousUsername = usernameTextBox.Text.Trim
    End Sub

a a a a a a a a a a
```

```
a a a a a a a a a a a

Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        If addUserBtn.Text = "&ADD USER" Then 'If is not in Add User mode
            Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. You'll be logged out",
"Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then
                    conn.Open()
                    Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text
                    Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text
                    Dim commandDelete As New OleDbCommand(SQLDeleteUser, conn)
```

```
                    Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, conn)
                    commandDeleteData.ExecuteReader() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                    commandDelete.ExecuteScalar() 'Then delete user
                    conn.Close()
                    MessageBox.Show("User has been successfully deleted", "Successful Deletion")
                Else
                End If
            Catch ex As Exception
                MessageBox.Show("Something has gone wrong", "Error")
            End Try
        Else
            MessageBox.Show("You are in Add User Mode, you cannot delete an account you have not created yet", "Error")
        End If
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear() 'Ease of use for clearing all inputs from textboxes
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails(loginForm.dataUsername)
            LeaderboardChange(leaderboardComboBox.SelectedIndex)
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
inputs have at least 5 characters for username and password
            MessageBox.Show("You have entered your credentials wrong. Please try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';"
            Dim commandCheck As New OleDbCommand(SQLCheckUser, conn)
            conn.Open()
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            conn.Close()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                conn.Open()
                Dim username, password, myName, admin As String
                username = usernameTextBox.Text.Trim
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
                End Select
                Dim sqlAddUser As String
                If addUserBtn.Text = "&CANCEL ADD USER" Then
                    sqlAddUser = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);"
                Else
                    sqlAddUser = "UPDATE [userData] SET [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" &
myName & "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";"
                End If
                Dim commandInsert As New OleDbCommand(sqlAddUser, conn)
                commandInsert.Parameters.AddWithValue("@username", username)
                commandInsert.Parameters.AddWithValue("@password", password)
                commandInsert.Parameters.AddWithValue("@name", myName)
                commandInsert.Parameters.AddWithValue("@admin", admin)
                commandInsert.ExecuteNonQuery()
                MessageBox.Show("User credentials have been modified", "Modification Successful")
                conn.Close()
            End If
        End If
    End Sub
```

User Statistics

When the form loads in, the ‘Max’ and ‘Average’ procedures are called which selects the record of the highest or average WPM or accuracy for a specific user. The ‘MAX’ and ‘AVG’ functions are used in SQL to do this. They are then outputted into the labels

```
    Sub Max(columnName, labelName) 'Fetches record by Maximum value of either (WPM or Accuracy' for a specific user
        Dim SQLGetMax = "SELECT MAX (" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQLGetMax, conn)
        conn.Open()
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString 'Rounds variable shown to 1 dp
        conn.Close()
    End Sub

    Sub Average(columnName, labelName) 'Fetches record by Average value of either (WPM or Accuracy' for a specific user
        Dim SQlGetAverage = "SELECT AVG(" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQlGetAverage, conn)
        commandGet.Parameters.AddWithValue("@column", columnName)
        conn.Open()
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString 'Rounds variable shown to 1 dp
        conn.Close()
    End Sub
```

Leaderboard

Triggered when the leaderboardComboBox is changed, the 'LeaderboardChange' function fetches and updates the DataGridViewer. This is done by changing the variable containing the SQL command to a new command depending on the option selected in the ComboBox. This is done using a Select Case statement. After that, the command is executed and the DataGridViewer is updated. This function is also called upon when the form first loads in.

```
a a a a a a a a a a a

    Private Sub leaderboardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
leaderboardComboBox.SelectedIndexChanged
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub
```

```
a a a a a a a a a a a

    Sub LeaderboardChange(indexValue) 'Depending on option selected in ComboBox, the SQL relating to leaderboardin DataGridViewer is
changed
        dataset.Clear()
        Dim SQLleaderboard As String
        Select Case indexValue
            Case 0
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.WPM) AS [Max Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 1
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.Accuracy) AS [Max Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 2
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.Accuracy) AS [Avg Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 3
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.WPM) AS [Avg Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case Else
        End Select
        Dim commandLeaderboard As New OleDbDataAdapter(SQLleaderboard, conn) 'SQL is excuted
        conn.Open()
        commandLeaderboard.Fill(leaderboardDataset, "Leaderboard")
        conn.Close()
        dataGridView.DataSource = leaderboardDataset.Tables("Leaderboard") 'Data is filled into the DataGridViewer
        leaderboardDataset.Tables.Remove("LeaderBoard") 'Reset and clear dataset
    End Sub
```

Charts

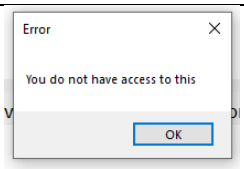
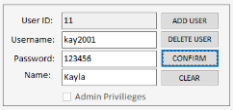
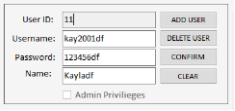
Using the two chart controls I made in the previous version. I have tried to add functionality to featch the data and output in onto the graphs.

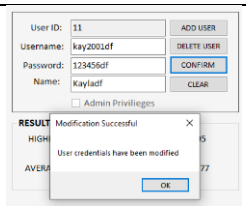
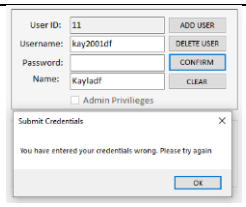
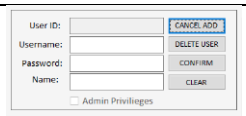
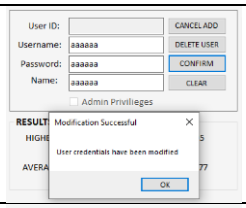
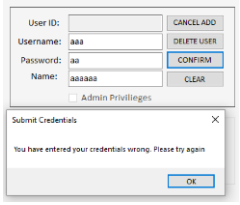
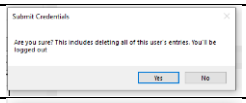
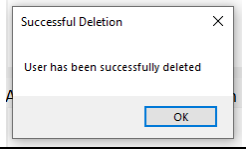
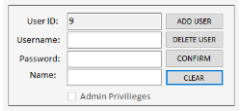
```
a a a a a a a a a a a

    Sub ChartWPMLoad()
        Dim SQLgetWPM = "SELECT DISTINCTROW Avg(entryData.WPM) AS WPM, entryData.Date FROM userData INNER JOIN entryData ON
userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLgetWPM, conn)
        conn.Open()
        commandGetWPM.Fill(chartDataset, "WPMTable")
        'Add a new series
```

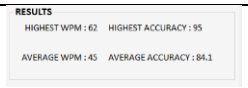
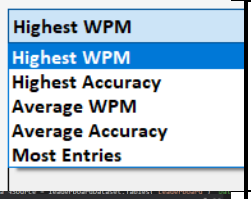
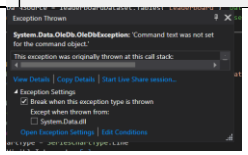
```
        Dim SeriesName As New Series
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userWPMChart.DataSource = chartDataset.Tables("WPMTable")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "WPM"
        userWPMChart.Series.Add(SeriesName)
        conn.Close()
    End Sub

    Sub ChartAccuracyLoad()
        Dim SQLGetAccuray = "SELECT DISTINCTROW Avg(entryData.Accuracy) AS Accuracy, entryData.Date FROM userData INNER JOIN entryData
ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLGetAccuray, conn)
        conn.Open()
        commandGetWPM.Fill(chartDataset, "SQLGetAccuray")
        'Add a new series
        Dim SeriesName As New Series
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userAccuracyChart.DataSource = chartDataset.Tables("SQLGetAccuray")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "Accuracy"
        userAccuracyChart.Series.Add(SeriesName)
        conn.Close()
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested.

Test	Result	Evidence	Comments
User can navigate to the 'Typing Test' form through the menu bar		N/A	
User can navigate to the 'WordFade' form through the menu bar		N/A	
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar			
User can navigate to the 'Admin' form when the criteria is met through the menu bar		N/A	
User can navigate to the 'Options' form through the menu bar		N/A	
Loads user details from database into the Textboxes			
User can edit 'Username', 'Password' and 'Name' Textboxes but not 'UserID' Textbox or 'Admin Privileges' Checkbox			

'Add User' button was not clicked before and the user edits their credentials with each textbox having text in it, 'Confirm' button is clicked and the credentials are updated if all the inputs are above 5 characters			Does not update data variables. Due to this, when the Refresh Button is clicked, it throws OutOfRangeExceptions, (Line 36)
'Add User' button was not clicked before and the user edits their credentials and 'Confirm' button is clicked. However, a textbox does not have text in it so does not update			
When 'Add User' is clicked, clear the textboxes and change the button to 'Stop Add User'. It's now in Add User Mode			
User goes into Add User mode, the user adds in credentials in every textbox and clicks 'Confirm' to add user			Does not log the user out or reset data variables
User goes into Add User mode, the user adds in credentials clicks 'Confirm' to add user. If the textboxes have no input or below 5 character the action will not be completed			
User wants to go out of Add User Mode		N/A	Throws OutOfRangeExceptions, (Line 36)
User wants to delete user by click 'Delete' button			
User wants to delete user by click 'Delete' button, and clicks 'Yes' to MessageBox			Does not log the user out or reset data variables
User wants to delete user by click 'Delete' button, and clicks 'No' to MessageBox		N/A	
If the user is in Add User mode and clicks the 'Add User' button, stop it		N/A	
If the clicks the 'Clear' button, clear the Textboxes			

Loads ‘Highest WPM’, ‘Average WPM’, ‘Highest Accuracy’, ‘Average Accuracy’ into labels			
Load Graph controls to ‘Average WPM / Date’ and ‘Average Accuracy / Date’ as Line Graphs for that user			Need to correct this
ComboBox would have the options of ‘Highest WPM’, ‘Average WPM’, ‘Highest Accuracy’, ‘Average Accuracy’ and ‘Most Entries’			
When choosing an option in the ComboBox, the DataGridVeiwer would be changed in accordingly to the option chosen			All leaderboards but ‘Most Entries’ have been implemented, if ‘Most Entries’ is clicked then an exception is thrown (Line 81)

Remedial Actions for Next Version

For the next iteration, I will correct and add the functionality of the graphs outputting the results of a specific user. Additionally, I will make sure that when modifying the accounts, that the user will be logged out and data variables will be reset when necessary toa void integral issues when adding new entries into the database. I will also add the ‘Most Entries’ SQL so the DataGridViewer can be properly updated to show the leaderboard.

Admin Form

Adding and Editing All User Accounts

The functionality is the same as the Statistics Form ‘Adding and Editing Own User Account’ section however the difference is that the user can modify the Admin Privileges and add modify the credentials of their own user and other user account. However, I have not yet added functionality where the solution checks whether the account being deleted is their own and therefore would log them out of it.

The have added the ‘Previous’ and ‘Next’ button. Where, it moves onto the next record in userData, so the next user and works in a cyclical structure where when it reaches the end of the records. If the user presses ‘Next’, it returns back to the starting user.

This is the code for the ‘Next’ and ‘Previous’ buttons.

```
a a a a a a a a a a a

    Private Sub nextBtn_Click(sender As Object, e As EventArgs) Handles nextBtn.Click 'totalRow variable is declared and values changed
with Line 87 (fetches the number of records (accounts)
        currentRow += 1
        Select Case currentRow
            Case totalRows 'reached the end of the table, go back to beginning
                currentRow = 0
        End Select
        GetUserDetails() 'Updates textboxes
```

```
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub previousBtn_Click(sender As Object, e As EventArgs) Handles previousBtn.Click
        currentRow -= 1
        Select Case currentRow
            Case -1  'Reached the start of the table, go to the end
                currentRow = totalRows - 1
        End Select
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub
```

This is the code for the modification of user details.

```
a a a a a a a a a a a

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then 'If currently in Add User Mode, come out of it
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails()
            totalRows = dataset.Tables(0).Rows.Count
            LoadDataGridUser()
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. If you are deleting your own
account then you'll be logged out", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
        Try
            If result = DialogResult.Yes Then
                conn.Open()
                Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim commandDelete As New OleDbCommand(SQLDeleteUser, conn)
                Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, conn)
                commandDeleteData.ExecuteScalar() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                commandDelete.ExecuteScalar() 'Then delete user
                conn.Close()
                currentRow = 0 'Resets naivagation of user accounts
                GetUserDetails()
                totalRows = dataset.Tables(0).Rows.Count 'Fetches new total accounts number
                LoadDataGridUser()
                MessageBox.Show("User has been successfully deleted", "Successful Deletion")
            End If
        Catch ex As Exception
            MessageBox.Show("Something has gone wrong", "Error")
        End Try
        conn.Close()
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear() 'Ease of use for clearing all inputs from textboxes
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
        Dim SQL As String
        Dim SQLUsername As String
        Dim SQLAdmin As String
        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
if there are no inputs in the textbox
            MessageBox.Show("You have entered your credentials wrong. Please try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';" 'SQL to
check if username is already present
            Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database, and whether is has been modified
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                Dim username, password, myName, admin As String 'Declares and assigns input values to variables
                username = usernameTextBox.Text.Trim 'Trim to avoid whitespace being present in database and when comparing for the
login process
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
```

```
                End Select
                If addUserBtn.Text = "&CANCEL ADD USER" Then 'If the Add User button was pressed, then Insert account to database
                    SQL = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);"
                    Dim commandInsert As New OleDbCommand(SQL, loginForm.conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteScalar()
                Else 'If not, update the currently logged on account
                    SQL = "UPDATE [userData] SET [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" & myName
& "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";"
                    Dim commandInsert As New OleDbCommand(SQL, loginForm.conn)
                    commandInsert.ExecuteScalar()
                End If
                MessageBox.Show("Action completed successfully", "Confirmation completed")
                currentRow = 0
                addUser = False
                addUserBtn.Text = "&ADD USER"
                GetUserDetails()
                totalRows = dataset.Tables(0).Rows.Count
                LoadDataGridUser()
            End If
        End If
    End Sub
```

Searching for a Specific User

The logic behind searching for a specific user is within the two extracts of code below. With 'GetUserDetails' fetches the specific users details and uploads them into the textboxes. However, I have not yet added any validation if the username is wrong.

```
a a a a a a a a a a a

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes when a specific username has been
specified
        conn.Open() 'SAME LOGIC
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(0)
        usernameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(1)
        passwordTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(3)
        If dataset.Tables("userData").Rows(currentRow).Item(4) = "Yes" Then
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        conn.Close()
    End Sub
```

The 'Search' button

```
a a a a a a a a a a a

    Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles searchBtn.Click
        Dim userSearch As String
        userSearch = userSearchTextBox.Text 'Takes the value from user input as a variable
        conn.Open()
        Dim SQLSelectData = "SELECT username from [userData] WHERE username = '" & userSearch & "';"
        Dim commandCheckUser As New OleDbCommand(SQLSelectData, conn)
        conn.Close()
        currentRow = 0 'CurrentRow needs to equal 0, as the Logic applied would have only one record be shown (only one record for one
user with that username). After when 'Previous or 'Next' btn is pressed, resets naivgation to start from 0
        GetUserDetails(userSearch) 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub
```

Searching within a Specific Date Range

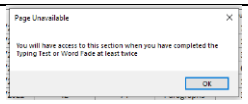
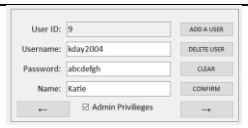
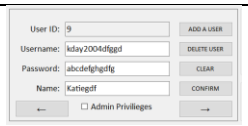
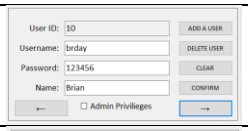
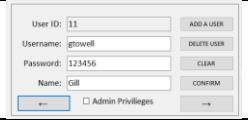
The functionality for specifying entries is with an SQL statement stating WHERE date1 BETWEEN date2. So the logic for executing commands is identical as fetching entries without conditions.

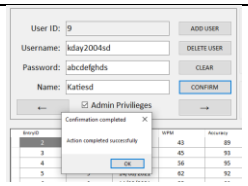
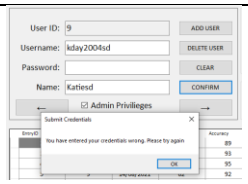
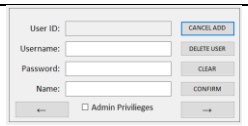
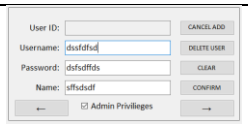
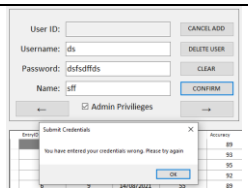
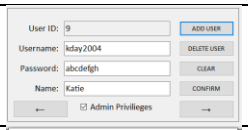
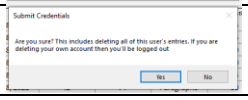
```
a a a a a a a a a a a

    Sub LoadDateDataGridUser(startDate, endDate) 'Same as the previous, however there are parameters for the dates
        conn.ConnectionString = loginForm.connectionString
        conn.Open()
```

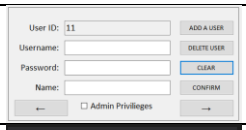
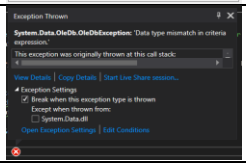
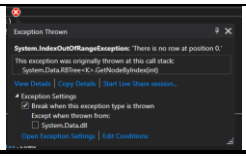
```
        Dim SQLSelectDateData = "SELECT * FROM [entryData] WHERE UserID =" & userIDTextBox.Text & " AND Date BETWEEN @start AND @end"
        Dim commandSelect As New OleDbCommand
        commandSelect.Connection = conn
        commandSelect.CommandText = SQLSelectDateData
        commandSelect.Parameters.Add("@start", startDatePicker.Value)
        commandSelect.Parameters.Add("@start", endDatePicker.Value)
        dataset.Clear()
        Dim adapter As New OleDbDataAdapter(commandSelect)
        adapter.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Filling DataGridView with data from the dataset
        conn.Close()
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested.

Test	Result	Evidence	Comments
User can navigate to the 'Typing Test' form through the menu bar		N/A	
User can navigate to the 'WordFade' form through the menu bar		N/A	
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar			
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		N/A	
User can navigate to the 'Options' form through the menu bar		N/A	
Loads the first user details from database into the Textboxes and DataGridVeiwer			
User can edit 'Username', 'Password', 'Name' Textboxes and 'Admin Privileges' Checkbox but not 'UserID' Textbox			
User can go to next user by clicking the '→' button			
User can go to the previous user by clicking the '←' button			

'Add User' button was not clicked before and the user edits their credentials with each textbox having text in it, 'Confirm' button is clicked and the credentials are updated			
'Add User' button was not clicked before and the user edits their credentials and 'Confirm' button is clicked. However, a textbox does not have text in it so does not update			
When 'Add User' is clicked, clear the textboxes and change the button to 'Stop Add User'. It's now in Add User Mode			
User goes into Add User mode, the user adds in credentials in every textbox and clicks 'Confirm' to add user			
User goes into Add User mode, the user adds in credentials clicks 'Confirm' to add user. If the textboxes have no input or below 5 character the action will not be completed			
User wants to go out of Add User Mode			
User wants to delete user by click 'Delete' button			
User wants to delete user which is their own login by click 'Delete' button, and clicks 'Yes' to MessageBox		N/A	Successfully deletes user, however does not log the user out
User wants to delete user which is not their own login by click 'Delete' button, and clicks 'Yes' to MessageBox		N/A	
User wants to delete user by click 'Delete' button, and clicks 'No' to MessageBox		N/A	
If the user is in Add User mode and clicks the 'Cancel Add' button, stop it			

If the clicks the 'Clear' button, clear the Textboxes			
User can specify the dates the entries are between entries in DataGridViewer			Line 41, OleDbException thrown
User can search for a specific username, and it will be loaded into the Textboxes and DataGridVeiwer		N/A	
User tries to search for a username not in the program then prompt the user			Line 69, IndexOutOfRange Exception

Remedial Actions for Next Version

Need to add validation for the 'Delete' button to log out. Otherwise, if the user deletes their own account. They will be logged into an account that is not in the database. It will cause integrity issues when adding new entries.

Additionally, I need the solve the problems involving finding entries between specific dates. It is a feature I have said I will add and needs to be fixed. This is the same for having an incorrect search for a username. The solution will crash and not work. This needs to be fixed also.

Options Form

The only functionality I have added in the options Form is in line 64 where I added '`loginForm.GetAmountEntries()`'. This is so the form checks whether the current user has entries associated with their account and can open the Statistics form in accordance.

However, as said before, I have functionality to the variables that hold the options for the user. Where the captials and punctuation can be changed etc.. Giving a more customisable experience for the user in comparison to the first version where the Options form did not do anything for the Touch Typing experience.

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 are not shown here.

Test	Result	Evidence	Comments
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar		N/A	
User can navigate to the 'Statistics' form, provided the criteria is met through the			

menu bar			
When the type of extract for WordFade ComboBox is changed, it will be changed in the WordFade test		N/A	I have not yet implemented any functionality into the WordFade Form
When the CountDown or CountUp ComboBox is changed, it will be changed in the Typing Test and WordFade		N/A	Only added the function in Typing Test and not WordFade
When the Time Duration ComboBox is changed, it will be changed in the Typing Test and WordFade		N/A	Only added the function in Typing Test and not WordFade
When the Capitals checkbox is changed, it is changed for Typing Test and WordFade		hopes and dreams were dashed that day. it should have been expected, but it still came as a shock. the warning signs had been ignored in favor of the possibility, however remote, that it could actually happen. that possibility had grown from hope to an undeniable belief it must be destiny. that was until it wasn't and the hopes and dreams came crashing down.	Only added the function in Typing Test and not WordFade
When the Punctuation checkbox is changed, it is changed for Typing Test and WordFade (this only applies to 'Paragraph' extracts		Green vines attached to the trunk of the tree had wound themselves toward the top of the canopy Ants used the vine as their private highway avoiding all the creases and crags of the bark to freely move at top speed from top to bottom or bottom to top depending on their current chore At least this was the way it was supposed to be Something had damaged	Only added the function in Typing Test and not WordFade
When the Sudden Death checkbox is changed, it is changed for Typing Test and WordFade		N/A	Only added the function in Typing Test and not WordFade
When the Keyboard ComboBox has changed, it will be changed in the Typing Test and WordFade		N/A	Only added the function in Typing Test and not WordFade
User can change the font type, style and size for the Typing Test and WordFade		N/A	Only added the function in Typing Test and not WordFade

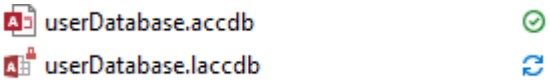
Remedial Actions for Next Version

For the next version, I will add more information to aid the user in the improvement of their skills. This will involve adding a picture into the PictureBox of a picture of the keyboard and explaining the meanings behind the key and discussing the correct posture when typing.

Third Version

Validating the Database Connection

For some of the core features in my solution, I need to be able to communicate with a database. However, there will be a chance where either the database file is not present before the start of the application or during the runtime of it.



The method I have currently used to establish connections to the database is with a different 'OleDbConnection' variable for each form. When connected, a locking file is created. So no changes can be made when the database is opened (in this case the database cannot be deleted mid

connection). However, between the opening and closing of forms and the 'Options form' that does not connect to the database at all. There are chances for the database to move location and cause errors.

Due to this, I have modified the strucutre of how my solutions handles the connection to the database. Instead, I have only one 'OleDbConnection' variable originating from the 'Login form'. Which will be passed into and be opened and closed dependent on the 'Login form'. Using this method means that I do not have to validate the database in runtime and only at the start.

For the valdiatoin at the start, I have created a function called 'CheckStartDatabase'. Which if the connection cannot be made then limits the user to only the Guest Mode where the database would ont be used at all.

Login Form

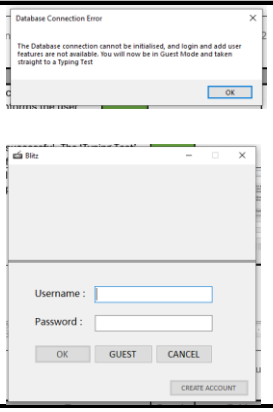
Validating Database Connection at the Start

Below is the function 'CheckStartDatabase' which checks the connection when the starting form is loaded in.

```
a a a a a a a a a a a

    Sub CheckStartDatabase()
        Try 'In a Try so it catches any exceptoins if database connection cannot be met
            conn.Open()
        Catch ex As Exception
            MessageBox.Show("The Database connection cannot be initialised, and login and add user features are not available. You will
now be in Guest Mode and taken straight to a Typing Test", "Database Connection Error")
            dataUsername = "Guest"
            submitBtn.Enabled = False
            createAccBtn.Enabled = False
        End Try
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 & 2 are not shown here.

Test	Result	Evidence	Comments
If database connection is not present, it informs the user			Note: The 'OK' and 'Create Account' button are disabled and cannot be clicked. (Hard to see in picture)

Remedial Actions for Next Version

I need to add in a picture for the PictureBox of the branding for the solution. This will be done by adding a picture to the resources of the project and it being included in the release build of the solution. I will also make some 'Quality of Life' improvements to this form (and across the forms in

the next iteration), this will include making the form look better. E.g. colour coding the buttons where the more vibrant colours will be for where the user is most likely wanting to go. This make the navigation of the form (and the solution as a whole) easier)

Add User Form

I have added functionality where the database checks if the username is already present using a SQL command. If present then stop the user adding it to the database.

There were changes made to this iteration is the elimination of the 'OleDbConnection' variable. As this was declared and made a connection to it in the 'Load' event. The 'Load' event is not needed anymore so was taken out too.

```
a a a a a a a a a a a

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Dim result As DialogResult
        If passTextBox.Text = retryPassTextBox.Text And passTextBox.Text.Trim.Length >= 5 And userTextBox.Text.Trim.Length >= 5 And
nameTextBox.Text.Trim.Length <> 0 Then 'Validation : Checking password entries match and that no textboxes and no characters in them
            result = MessageBox.Show("Are you sure?", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            If result = DialogResult.Yes Then
                Dim username, password, myName, admin As String
                username = userTextBox.Text.Trim
                password = passTextBox.Text.Trim
                myName = nameTextBox.Text.Trim '
                admin = "No"
                Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & username & "';"
                Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
                Dim checkUserPresent = commandCheck.ExecuteScalar()
                If checkUserPresent = username Then 'Checks if username is already present in database
                    MessageBox.Show("A user with this username is already present, please use a different username")
                Else
                    Dim sqlAddUser As String = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username,
@password, @name, @admin);"
                    Dim commandInsert As New OleDbCommand(sqlAddUser, loginForm.conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteNonQuery() 'Execute command to database
                    MessageBox.Show("User has been added")
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            End If
        Else
            MessageBox.Show("You have entered your credentials wrong.  Make sure that the Username and Password are at least 5
characters in length. Please try again", "Submit Credentials")
        End If
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 & 2 are not shown here.

Test	Result	Evidence	Comments
If the Username wanting to be added is already a pre-existing login then prevent from adding it again			
If database connection is not present, it informs the user		N/A	Confirmed at the start of the Application, and unable to delete database in runtime (no test needed)

Remedial Actions for Next Version

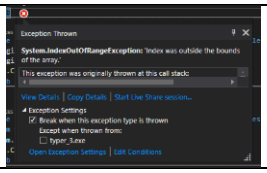
I will also make some 'Quality of Life' improvements to this form (and across the forms in the next iteration), this will include making the form look better. The reasoning and explanation is stated previously (same in the 'Remedial Actions' in the 'Login Form' section 'Version 3').

Typing Test Form

No significant functionality was added in this iteration apart from the new database connection.

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 & 2 are not shown here.

Test	Result	Evidence	Comments
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'			
If the extract has been completed, stop the Test, calculate and show the results			
If the database cannot be connected to input the results, send a prompt to the user		N/A	Confirmed at the start of the Application, and unable to delete database in runtime (no test needed)

Remedial Actions for Next Version

I need to find another way to check when the extract had been fully typed without causing exceptions. Otherwise, the solution would crash and stop working. I will also make some 'Quality of Life' improvements to this form (and across the forms in the next iteration), this will include making the form look better. The reasoning and explanation is stated previously (same in the 'Remedial Actions' in the 'Login Form' section 'Version 3'). This will include making the 'Start' button's back colour become red when pressed (to signify when pressed again the test will stop). Another aspect in particular I would like to add, is the auto switch of the 'Focus' (the tab index the user is currently on) from the 'Start' button (when pressed), back to the 'inputTextBox'. This is for ease of use so the user does not have to click the 'inputTextBox' every time. And instead can tab index to the 'Start; button from the 'inputTextBox', press enter and be sent back to the 'inputTextBox'.

WordFade Form

Differences when implementing the logic from the Typing Test Form into the WordFade Form

The main logic behind the loading in of text files and the string array created are the same. Apart from the added option of the alphabet text file. The method of checking the validity of a user input is the

same too. However, the presentation of the essay to the user is different, as unlike the Typing Test form, there are three separate RichTextBox's to represent the next word in the queue.

For this, I have created the procedure 'RefreshWordOrder'. At the start of the test, and after every word completed, this is called. Where it refreshes the text shown in each RichTextBox to the new word. I have also added in validation where the procedure checks the distance between the end of the word array and the current word the user is typing. This is to avoid 'OutOfRange Exceptions' cause by the solution finding the next item in the 'words' array when at the end of the array and clearing the previous words from the RichTextBox's.

```
a a a a a a a a a a a

    Function RefreshWordOrder(count)
        Select Case words.Length - count - 1 'Checks how close the wordCount is to the end of the array (to avoid OutOfRange
exceptions), then the correct RichTextBox's are cleared of text
            Case 1 '2 words away from end
                thirdWordRichTextBox.Clear()
                secondWordRichTextBox.Text = words(count + 1)
                firstWordRichTextBox.Text = words(count)
            Case 0 '1 word away from the end
                thirdWordRichTextBox.Clear()
                secondWordRichTextBox.Clear()
                firstWordRichTextBox.Text = words(count)
            Case -1 'When the procedure is called for the end of test
            Case Else '>2 words away
                thirdWordRichTextBox.Text = words(count + 2)
                secondWordRichTextBox.Text = words(count + 1)
                firstWordRichTextBox.Text = words(count)
        End Select
    End Function
```

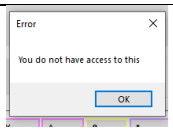
Another difference between the logic which I needed to change, was how the solution handled the forecolour of the text to accurately represent the progress of the user on that word. E.g. original colour to green forecolour when the letter was typed correctly. Therefore, when calling the 'TurnColour' procedure, we pass the 'countChar' variable instead of the 'countCharEssay' variable. The only use for the 'countCharEssay' variable in the WordFade form is when calculating the WPM.

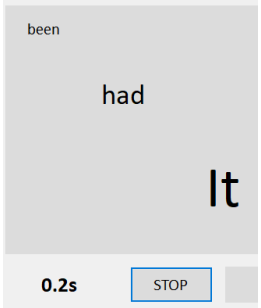
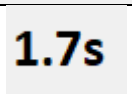
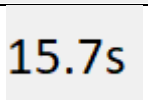
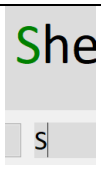
```
a a a a a a a a a a a

            Dim differenceOfFaults As Int32
        Try
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'Checks if full
word is correct and the user has pressed spaces to go onto next word
                countChar = 0
                countCharEssay += 1
                countWord += 1
                RefreshWordOrder(countWord)
                inputTextBox.Clear() 'Resets input box
                If countChar = inputTextBox.TextLength And words.Length = countWord Then 'Checks if full essay has been completed
                    timeDoneLabel.Text = "STOP"
                    startFadeBtn.Text = "START"
                    inputTextBox.ReadOnly = True
                    inputTextBox.Clear()
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Goes onto next word
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                End If
            ElseIf (countChar + 1 = inputTextBox.TextLength) And (countChar <> charList.Length) And currentFaults = 0 Then
                If (inputTextBox.Text.EndsWith(charList(countChar)) = True) And (countChar + 1 = inputTextBox.TextLength) Then 'Checks
if inputted letter is correct
                    TurnColour(countChar, 1, Color.Green)
                    countChar += 1
                    countCharEssay += 1
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Else 'Checks 1st incorrect character
                    currentFaults += 1
                    totalFaults += 1
                    firstWordRichTextBox.BackColor = Color.IndianRed
                    secondWordRichTextBox.BackColor = Color.IndianRed
                    thirdWordRichTextBox.BackColor = Color.IndianRed
                    essayPanel.BackColor = Color.IndianRed
                    TurnColour(countChar, (inputTextBox.TextLength - countChar), Color.Red)
                    If loginForm.suddenDeath = True Then
                        timeDoneLabel.Text = "STOP"
                        startFadeBtn.Text = "START"
```

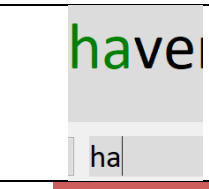
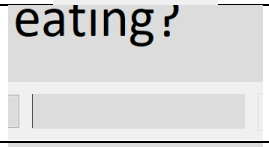
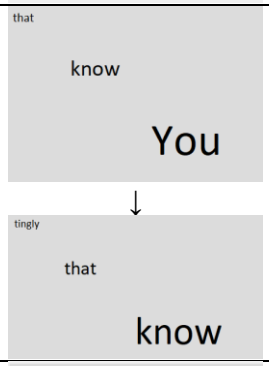
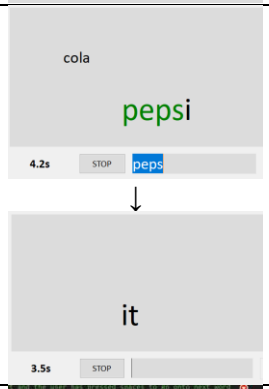
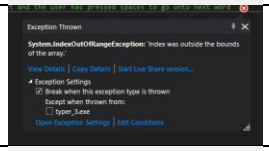
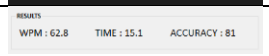
```
                        inputTextBox.ReadOnly = True
                        inputTextBox.Clear()
                        startType = False
                        wordTimer.Stop()
                        ChangeAllKeyColourGrey()
                    End If
                End If
            Else
                If (inputTextBox.TextLength > (countChar)) Then 'Checks if any inputted letter are incorrect, checks the 2nd+ incorrect
character
                    totalFaults += 1
                    currentFaults += 1
                    TurnColour(countChar, (inputTextBox.TextLength - countChar), Color.Red)
                    If essayPanel.BackColor <> Color.IndianRed Then
                        firstWordRichTextBox.BackColor = Color.IndianRed
                        secondWordRichTextBox.BackColor = Color.IndianRed
                        thirdWordRichTextBox.BackColor = Color.IndianRed
                        essayPanel.BackColor = Color.IndianRed
                    End If
                End If
                If inputTextBox.TextLength < (countChar + currentFaults + 1) Then 'Checks if a red letter has been backspaced
                    differenceOfFaults = (countChar + currentFaults) - inputTextBox.TextLength
                    currentFaults -= differenceOfFaults
                    TurnColour(countChar + currentFaults, differenceOfFaults, Color.Black)
                    If inputTextBox.TextLength < (countChar) And currentFaults <= 0 Then 'Checks if a green letter has been backspaced
                        currentFaults = 0
                        countChar -= (countChar - inputTextBox.TextLength) 'subtracts from counts accordingly
                        countCharEssay -= (countChar - inputTextBox.TextLength + 1)
                        If loginForm.showKeyboard = "ShowHighlight" Then
                            ChangeKeyColour(countChar) 'changes key highlighted to previous key
                        End If
                    End If
                    If currentFaults = 0 Then
                        firstWordRichTextBox.BackColor = Color.Gainsboro
                        secondWordRichTextBox.BackColor = Color.Gainsboro
                        thirdWordRichTextBox.BackColor = Color.Gainsboro
                        essayPanel.BackColor = Color.Gainsboro
                    End If
                End If
            End If
        Catch
        End Try
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 & 2 are not shown here.

Test	Result	Evidence	Comments
User can navigate to the 'Typing Test' form through the menu bar		N/A	
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar			
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		N/A	
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar			
User can navigate to the 'Admin' form when the criteria is met through the menu bar		N/A	

User can navigate to the 'Options' form through the menu bar		N/A	
Extract Textbox is not editable		N/A	
Before a Typing Test has started, the user cannot input into the Input Textbox		N/A	
User can log out of the program		N/A	
An extract is loaded into the program, and the first 3 words are in the Textboxes when a WordFade is about to start			
A count down from 3 starts, when at 0 the Typing Test starts every 0.1s			
When the WordFade starts, the user is now able to input into the Input Textbox			
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'			
Input Textbox is now editable			
Timer is updated in a label every 0.1s until the end or is stopped			
If the character the user types is correct, change the forecolour for that letter on the extract to Green, keyboard key highlighted is updated			
If the character the user types is incorrect, change the forecolour for that letter on the extract to Red			
If the user has made errors and does not correct them, make all subsequent characters Red regardless of if they are correct, and do not move onto next word			

If the user has did not make any errors but backspaces on a Green character, change it to original colour			
If the user has made errors, and backspaces to correct it, make the Red character now original colour			
If the user tries to backspace to a previous word, stop at the start of the current word			
If the word is correctly typed completely and the user presses space onto the next word, clear the Input Textbox to make room for the new character. Move the two other words down and load in the third word			
If near the end of the essay, 3rd and/or 2nd textboxes will have no string in it (to signify end of essay)			
If the extract has been completed, stop the Test, calculate and show the results			
If the Time Limit is reached, stop the Test, calculate and show the results			
Once results are calculated and the user is not in Guest Mode, input them into the database corresponding to the user		N/A	
Once results are calculated and the user is in Guest Mode, do not input results into the database		N/A	
If the ‘Stop’ button is clicked,		N/A	

stop the Typing Test			
If the database cannot be connected to input the results, send a prompt to the user		N/A	Confirmed at the start of the Application, and unable to delete database in runtime (no test needed)

Remedial Actions for Next Version

The remedial actions for this form are the same actions as they are in the Typing Test form.

Statistics Form

Adding and Editing Own User Account

I have amended the problems in regards to the 'Delete' button not logging the user out after they delete their own account. I have done this by calling the 'LogOut' procedure from the Login form and then opening the Login form and closing the Statistics form.

```
a a a a a a a a a a a

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        If addUserBtn.Text = "&ADD USER" Then 'If is not in Add User mode
            Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. You'll be logged out",
"Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then
                    Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text
                    Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text
                    Dim commandDelete As New OleDbCommand(SQLDeleteUser, loginForm.conn)
                    Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, loginForm.conn)
                    commandDeleteData.ExecuteReader() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                    commandDelete.ExecuteScalar() 'Then delete user
                    MessageBox.Show("Your account has been successfully deleted, you will now be logged out", "Successful Deletion")
                    loginForm.LogOut() 'Data variables are cleaned of values
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            Catch ex As Exception
                MessageBox.Show("Something has gone wrong", "Error")
            End Try
        Else
            MessageBox.Show("You are In Add User Mode, you cannot delete an account you have Not created yet", "Error")
        End If
    End Sub
```

Another amendment I have made is when confirming the edits made to the user's account. Now, when the 'Confirm' button is pressed and edits or the creation of a new account have been successful. A prompt is shown to the user, to inform them that they are now logged into the edited user's account (e.g. if they make a new account, then they are logged into that one). Therefore, I have added functionality to update the data variables and to direct them back to the Typing Test form if a new user. This is because if logged into a new account, they would not have any entries and would cause exceptions in the Chart Controls.

```
a a a a a a a a a a a

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click

        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
inputs have at least 5 characters for username and password
            MessageBox.Show("You have entered your credentials wrong. Please Try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';"
            Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                Dim username, password, myName, admin As String
                username = usernameTextBox.Text.Trim
                password = passwordTextBox.Text.Trim
```

```
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
                End Select
                Dim sqlAddUser As String
                If addUserBtn.Text = "&CANCEL ADD USER" Then
                    sqlAddUser = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);" 'For adding a new user
                Else
                    sqlAddUser = "UPDATE [userData] Set [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" &
myName & "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";" 'Updating current records credentials
                End If
                Dim commandInsert As New OleDbCommand(sqlAddUser, loginForm.conn)
                commandInsert.Parameters.AddWithValue("@username", username)
                commandInsert.Parameters.AddWithValue("@password", password)
                commandInsert.Parameters.AddWithValue("@name", myName)
                commandInsert.Parameters.AddWithValue("@admin", admin)
                commandInsert.ExecuteNonQuery()
                MessageBox.Show("User credentials have been modified, you are now logged in as the edited user", "Modification
Successful")
                loginForm.dataUsername = username
                loginForm.dataName = myName
                loginForm.dataAdmin = adminCheckBox.Checked
                If addUser = True Then 'Ejects user from statistics form if a new user has been added
                    addUser = False
                    Dim sqlGetUserID = "SELECT [UserID] FROM [userData] WHERE username = @username" 'Update data variable to new user
                    Dim commandUserID As New OleDbCommand(sqlGetUserID, loginForm.conn) 'UserID would only be changed if it's a new user
                    commandUserID.Parameters.AddWithValue("@username", username)
                    loginForm.dataUserID = commandUserID.ExecuteScalar()
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Close()
                End If
                GetUserDetails(loginForm.dataUsername) 'Updates user's credentials
                addUserBtn.Text = "&ADD USER"
            End If
        End If
    End Sub
```

Leaderboard

I have added SQL for the 'Total Entries' and added an 'ORDER BY ___ DESC' to each SQL string, so that it is presented correctly to the leaderboard as a ranking instead of records by 'UserID'. Additionally, I discovered that two SQL commands were outputting for the wrong ComboBox and amended it.

```
a a a a a a a a a a a

    Sub LeaderboardChange(indexValue) 'Depending on option selected in ComboBox, the SQL relating to leaderboardin DataGridViewer is
changed
        dataset.Clear()
        Dim SQLleaderboard As String
        Select Case indexValue
            Case 0
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.WPM) AS [Max Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY MAX(entryData.WPM) DESC;"
            Case 1
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.Accuracy) AS [Max Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY MAX(entryData.Accuracy) DESC;"
            Case 2
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.WPM) AS [Avg Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY AVG(entryData.WPM) DESC;"
            Case 3
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.Accuracy) AS [Avg Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY AVG(entryData.Accuracy) DESC;"
            Case 4
                SQLleaderboard = "SELECT userData.Username, Count(entryData.EntryID) AS [Amount Of Entries] FROM userData INNER JOIN
entryData ON userData.UserID = entryData.UserID GROUP BY userData.Username ORDER BY COUNT(entryData.EntryID) DESC;"
            Case Else
        End Select
        Dim commandLeaderboard As New OleDbDataAdapter(SQLleaderboard, loginForm.conn) 'SQL is excuted
        commandLeaderboard.Fill(leaderboardDataset, "Leaderboard")
        dataGridView.DataSource = leaderboardDataset.Tables("Leaderboard") 'Data is filled into the DataGridViewer
        leaderboardDataset.Tables.Remove("LeaderBoard") 'Reset and clear dataset
    End Sub
```

Charts

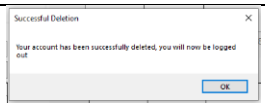
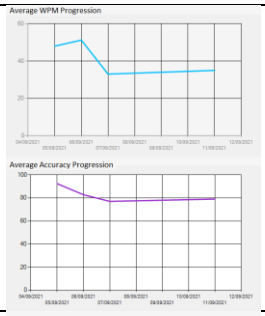
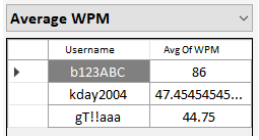
I have added the functionality of the chart controls to output results stored in the database. Additionally, I have added a series to each Chart control that are colour coded to differentiate the charts.

```
a a a a a a a a a a a

Sub ChartWPMLoad()
        Dim SQLgetWPM = "SELECT DISTINCTROW Avg(entryData.WPM) AS WPM, entryData.Date FROM userData INNER JOIN entryData ON
userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLgetWPM, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "WPMTable")
        'Add a new series
        Dim SeriesName As New Series
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").BackColor = Color.WhiteSmoke
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Black
        SeriesName.Color = Color.DeepSkyBlue
        SeriesName.BorderWidth = 3
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "WPM"
        userWPMChart.DataSource = chartDataset.Tables("WPMTable")
        userWPMChart.Series.Add(SeriesName)
    End Sub

    Sub ChartAccuracyLoad()
        Dim SQLGetAccuray = "SELECT DISTINCTROW Avg(entryData.Accuracy) AS Accuracy, entryData.Date FROM userData INNER JOIN entryData
ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLGetAccuray, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "SQLGetAccuray")
        'Add a new series
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").BackColor = Color.WhiteSmoke
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Black
        Dim SeriesName As New Series
        SeriesName.Color = Color.DarkOrchid
        SeriesName.BorderWidth = 3
        SeriesName.Name = "Accuracy"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "Accuracy"
        userAccuracyChart.DataSource = chartDataset.Tables("SQLGetAccuray")
        userAccuracyChart.Series.Add(SeriesName)
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Versions 1 & 2 are not shown here.

Test	Result	Evidence	Comments
'Add User' button was not clicked before and the user edits their credentials with each textbox having text in it, 'Confirm' button is clicked and the credentials are updated if all the inputs are above 5 characters		N/A	
User goes into Add User mode, the user adds in credentials in every textbox and clicks 'Confirm' to add user		N/A	

User wants to go out of Add User Mode		N/A	
User wants to delete user by click 'Delete' button, and clicks 'Yes' to MessageBox			User is logged out and taken back to 'Login form'
Load Graph controls to 'Average WPM / Date' and 'Average Accuracy / Date' as Line Graphs for that user			
When choosing an option in the ComboBox, the DataGridVeiwer would be changed in accordingly to the option chosen			The act of choosing a leaderboard in the ComboBox and the DataGridViewer changing accordingly is working well. However, there are formatting errors for decimal places needed to be sorted out.

Remedial Actions for Next Version

I will also make some 'Quality of Life' improvements to this form (and across the forms in the next iteration), this will include making the form look better. The reasoning and explanation is stated previously (same in the 'Remedial Actions' in the 'Login Form' section 'Version 3').

Admin Form

Adding and Editing All User Accounts

I have amended the problems in regards to the 'Delete' button not logging the user out after they delete their own account. I have done this by calling the 'LogOut' procedure from the Login form and then opening the Login form and closing the Statistics form. This is only done when the user deletes their own account and not another person's. So the solution checks for this condition.

```
a a a a a a a a a a a

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. If you are deleting your own
account then you'll be logged out", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
        Try
            If result = DialogResult.Yes Then
                Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim commandDelete As New OleDbCommand(SQLDeleteUser, loginForm.conn)
                Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, loginForm.conn)
                commandDeleteData.ExecuteScalar() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                commandDelete.ExecuteScalar() 'Then delete user
                If userIDTextBox.Text = loginForm.dataUserID Then 'Checks if the account deleted is currently logged in
                    loginForm.Show()
                    Me.Close()
                Else
                    currentRow = 0 'Resets navigation of user accounts
                    GetUserDetails()
                    totalRows = dataset.Tables(0).Rows.Count 'Fetches new total accounts number
                    LoadDataGridUser()
                    MessageBox.Show("User has been successfully deleted", "Successful Deletion")
                End If
            End If
        Catch ex As Exception
            MessageBox.Show("Something has gone wrong", "Error")
        End Try
    End Sub
```

Searching for a Specific User

The previous iteration could successfully search and load in a specific user's data depending on the user input. However, did not have validation when the user input is a wrong/invalid username.

```
a a a a a a a a a a a

    Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles searchBtn.Click
        Dim userSearch As String
        userSearch = userSearchTextBox.Text 'Takes the value from user input as a variable
        Dim SQLSelectData = "SELECT username from [userData] WHERE username = '" & userSearch & "';"
        Dim commandCheckUser As New OleDbCommand(SQLSelectData, loginForm.conn)
        Try
            If commandCheckUser.ExecuteScalar Is Nothing Then
                MessageBox.Show("No user has been found with this name", "Error")
            Else
                currentRow = 0 'CurrentRow needs to equal 0, as the Logic applied would have only one record be shown (only one record
for one user with that username). After when 'Previous or 'Next' btn is pressed, resets naivgation to start from 0
                GetUserDetails(userSearch) 'Updates textboxes
                LoadDataGridUser() 'Updates DataGridViewer with entries
            End If
        Catch ex As Exception
        End Try
    End Sub
```

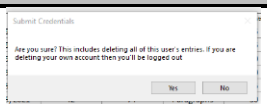
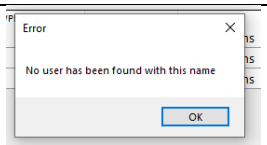
Searching within a Specific Date Range

I have corrected the errors made when trying to search and filter using dates to look at entries for that user. I had found a forum (click this link) which showed a solution to a poster how had a similar problem to me. This solution now works.

```
a a a a a a a a a a a

    Sub LoadDateDataGridUser(startDate, endDate) 'Same as the previous, however there are parameters for the dates
        Dim SQLSelectDateData = "SELECT * FROM [entryData] WHERE UserID =" & userIDTextBox.Text & " AND Date BETWEEN @start AND @end"
        Dim commandSelect As New OleDbCommand
        commandSelect.Connection = loginForm.conn
        commandSelect.CommandText = SQLSelectDateData
        commandSelect.Parameters.Add("@start", OleDbType.Date).Value = startDatePicker.Value
        commandSelect.Parameters.Add("@start", OleDbType.Date).Value = endDatePicker.Value
        dataset.Clear()
        Dim adapter As New OleDbDataAdapter(commandSelect)
        adapter.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Filling DataGridView with data from the dataset
    End Sub
```

Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 & 2 are not shown here.

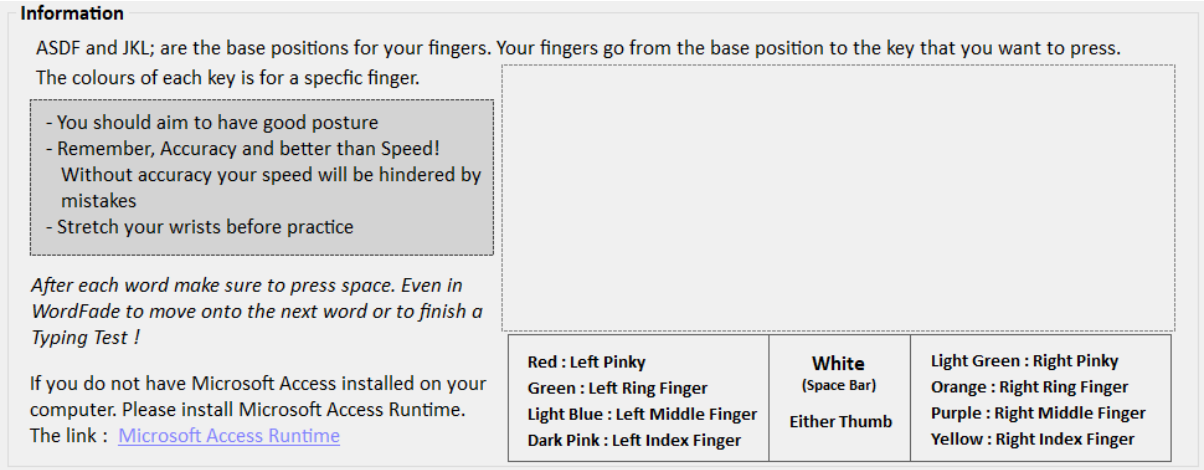
Test	Result	Evidence	Comments
User wants to delete user which is their own login by click 'Delete' button, and clicks 'Yes' to MessageBox			
User can specify the dates the entries are between entries in DataGridViewer			
User tries to search for a username not in the program then prompt the user			

Remedial Actions for Next Version

I will also make some 'Quality of Life' improvements to this form (and across the forms in the next iteration), this will include making the form look better. The reasoning and explanation is states previously (same in the 'Remedial Actions' in the 'Login Form' section 'Version 3').

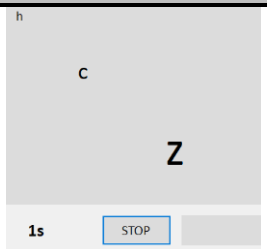
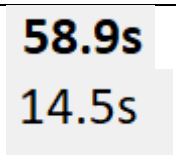
Options Form

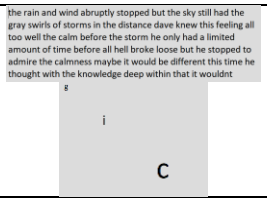
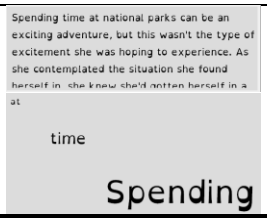
There was no new code added in this section apart from controls in the information section to explain more about how the solution work. As shown below:



Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Version 1 & 2 are not shown here.

Test	Result	Evidence	Comments
When the type of extract for WordFade ComboBox is changed, it will be changed in the WordFade test			
When the CountDown or CountUp ComboBox is changed, it will be changed in the Typing Test and WordFade		N/A	
When the Time Duration ComboBox is changed, it will be changed in the Typing Test and WordFade			

When the Capitals checkbox is changed, it is changed for Typing Test and WordFade			
When the Punctuation checkbox is changed, it is changed for Typing Test and WordFade (this only applies to 'Paragraph' extracts		N/A	
When the Sudden Death checkbox is changed, it is changed for Typing Test and WordFade		N/A	
When the Keyboard ComboBox has changed, it will be changed in the Typing Test and WordFade		N/A	
User can change the font type, style and size for the Typing Test and WordFade			

Remedial Actions for Next Version

I will also make some 'Quality of Life' improvements to this form (and across the forms in the next iteration), this will include making the form look better. The reasoning and explanation is states previously (same in the 'Remedial Actions' in the 'Login Form' section 'Version 3'). Additionally, I need to add in a picture for the PictureBox of the keyboard for the solution. This will be done by adding a picture to the resources of the project and it being included in the release build of the solution.

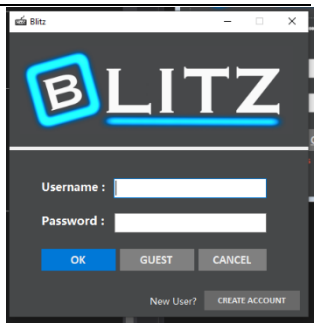
My Final Solution

The majority of aesthetic improvements and other improvements made in this solution is for ease of use. This in particular aids the competition of the usability features set out in the 'Analysis' section.

Login Form

Aesthetic Improvements

I have made a logo with the name 'Blitz' in Paint which has been added to the Resources section of the Build Solution. This is included with the released version where it is shown in the PictureBox.

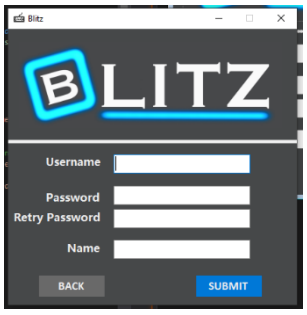


I have additionally changed the colours of the controls where the 'OK' button has been coloured 'Blue' in comparison to all other buttons being 'Gray' to make it easier for the user to navigate the 'Login form'.

Add User Form

Aesthetic Improvements

The improvements made have a similar explanation to the 'Version 4 – Login Form' section.



Typing Test Form

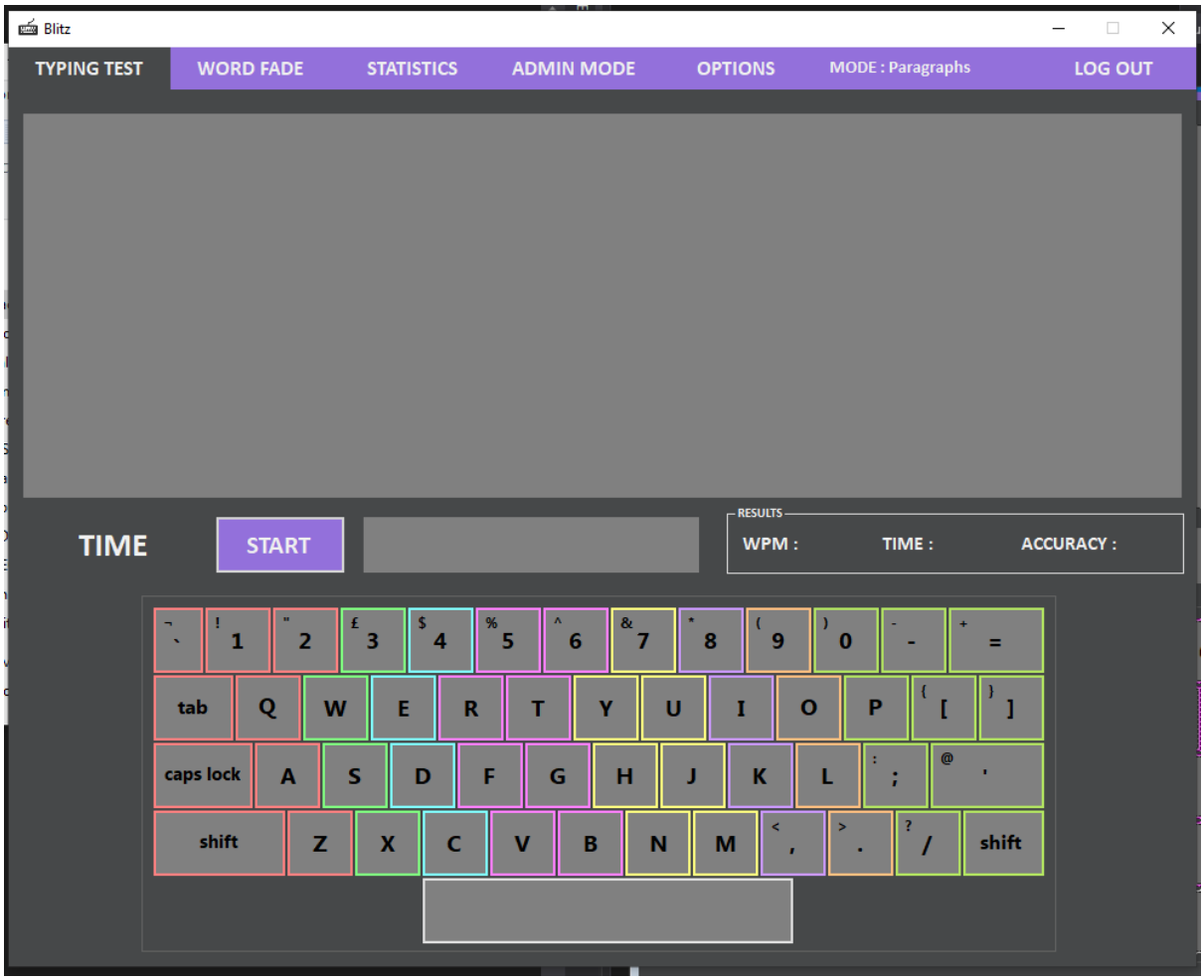
Auto Focusing with Tab Indexes

In the 'Properties' section for each control, I can specify the index and order which the focus can be navigated. I have decided to change the 'inputTextBox' tab index to 7 (1-6 are for the menu bar navigation) and the 'Start' button to 8. Then when focused on the 'inputTextBox' the user must press 'Tab' on the keyboard to focus onto the 'Start' button. I then added a function to jump the focus back to index 7 after the 'Start' button is pressed. This is to help the user practise the skill more easily. Not having to leave the keyboard to navigate the form. The function used to change indexes is JumpToIndex(). I have made the function to pass an index value in as the destination for the focus. So then I can use it to re-focus the index back to the 'Start' button when a test has been completed (called in MeasureTime()).

```
a a a a a a a a a a a

    Public Sub JumpToIndex(ByVal I As Integer)
        For Each control As Control In Me.Controls
            If control.TabIndex = I Then
                control.Focus()
            End If
        Next
    End Sub
```

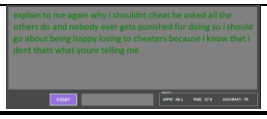
Aesthetic Improvements

For each form apart from the 'Login form' and the 'Add User form', there is a distinct colour scheme for each form. In this case 'Medium Purple'. The reasonings for the improvements made have a similar explanation to the 'Version 4 – Login Form' section.



Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Versions 1, 2 & 3 are not shown here.

Test	Result	Evidence	Comments
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'			
If the extract has been completed, stop the Test, calculate and show the results			

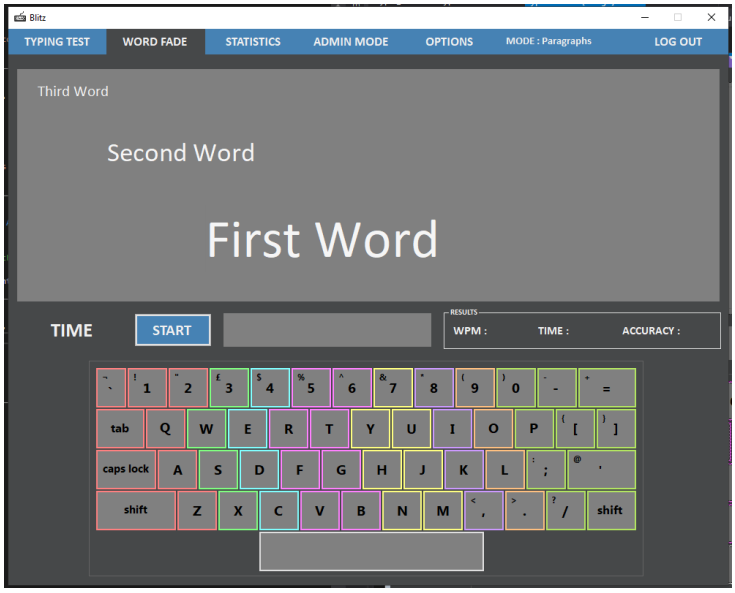
WordFade Form

Auto Focusing with Tab Indexes

The thought process and functionality behind the feature is the same as the 'Version 4 – Typing Test Form – Auto Focusing with Tab Indexes'.

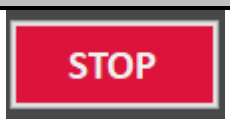
Aesthetic Improvements

The reasonings for the improvements made have a similar explanation to the 'Version 4 – Login Form' section.



Testing

Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. Please note: Already Green test results from Versions 1, 2 & 3 are not shown here.

Test	Result	Evidence	Comments
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'			
If the extract has been completed, stop the Test, calculate and show the results			

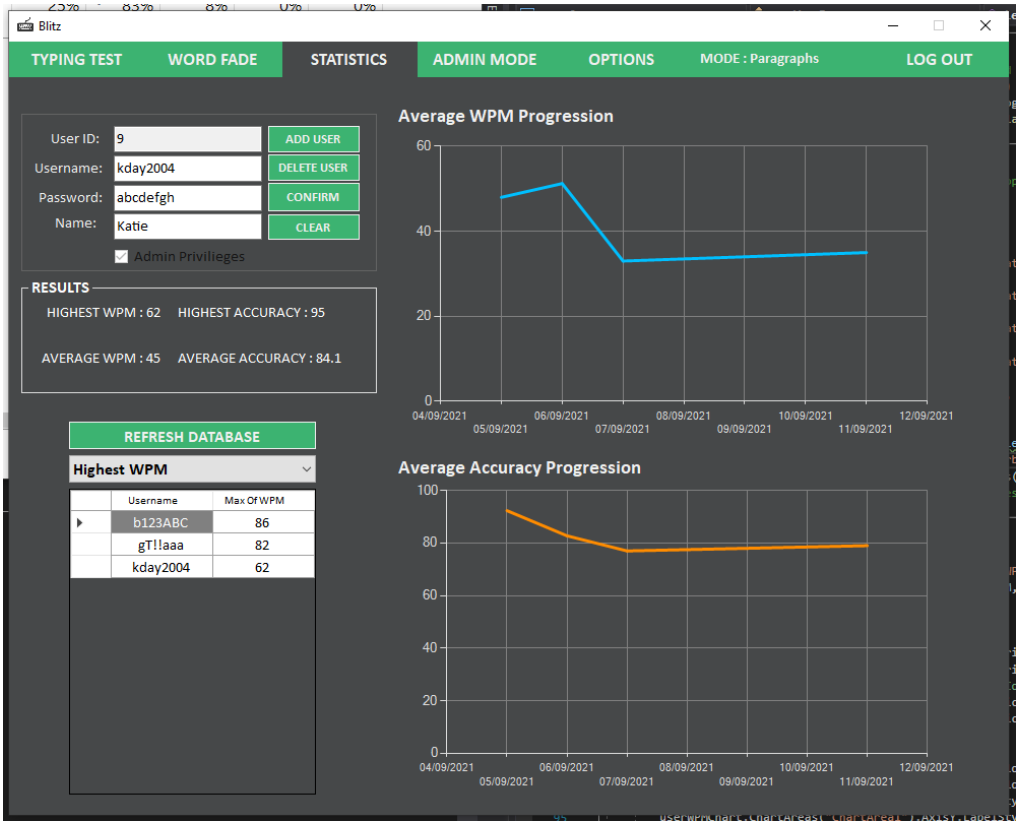
Statistics Form

Aesthetic Improvements

The reasonings for the improvements made have a similar explanation to the 'Version 4 – Login Form' section. There was code changed in the 'Statistics form' regarding the graph and the series created and the colours associated to it.

```
Function ChartWPMLoad()
        Dim SQLgetWPM = "SELECT DISTINCTROW Avg(entryData.WPM) AS WPM, entryData.Date FROM userData INNER JOIN entryData ON
userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
```

```
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLgetWPM, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "WPMTable")
        'Add a new series
        Dim SeriesName As New Series
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Gray
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Gray
        userWPMChart.ChartAreas("ChartArea1").BackColor = Color.FromArgb(70, 72, 73)
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Gainsboro
        SeriesName.Color = Color.DeepSkyBlue
        SeriesName.BorderWidth = 3
        userWPMChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Gainsboro
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userWPMChart.DataSource = chartDataset.Tables("WPMTable")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "WPM"
        userWPMChart.Series.Add(SeriesName)
    End Function

    Function ChartAccuracyLoad()
        Dim SQLGetAccuray = "SELECT DISTINCTROW Avg(entryData.Accuracy) AS Accuracy, entryData.Date FROM userData INNER JOIN entryData
ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLGetAccuray, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "SQLGetAccuray")
        'Add a new series
        Dim SeriesName As New Series
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Gray
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Gray
        userAccuracyChart.ChartAreas("ChartArea1").BackColor = Color.FromArgb(70, 72, 73)
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Gainsboro
        SeriesName.Color = Color.DarkOrange
        SeriesName.BorderWidth = 3
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Gainsboro
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userAccuracyChart.DataSource = chartDataset.Tables("SQLGetAccuray")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "Accuracy"
        userAccuracyChart.Series.Add(SeriesName)
    End Function
```



Remaining Amber Test Data

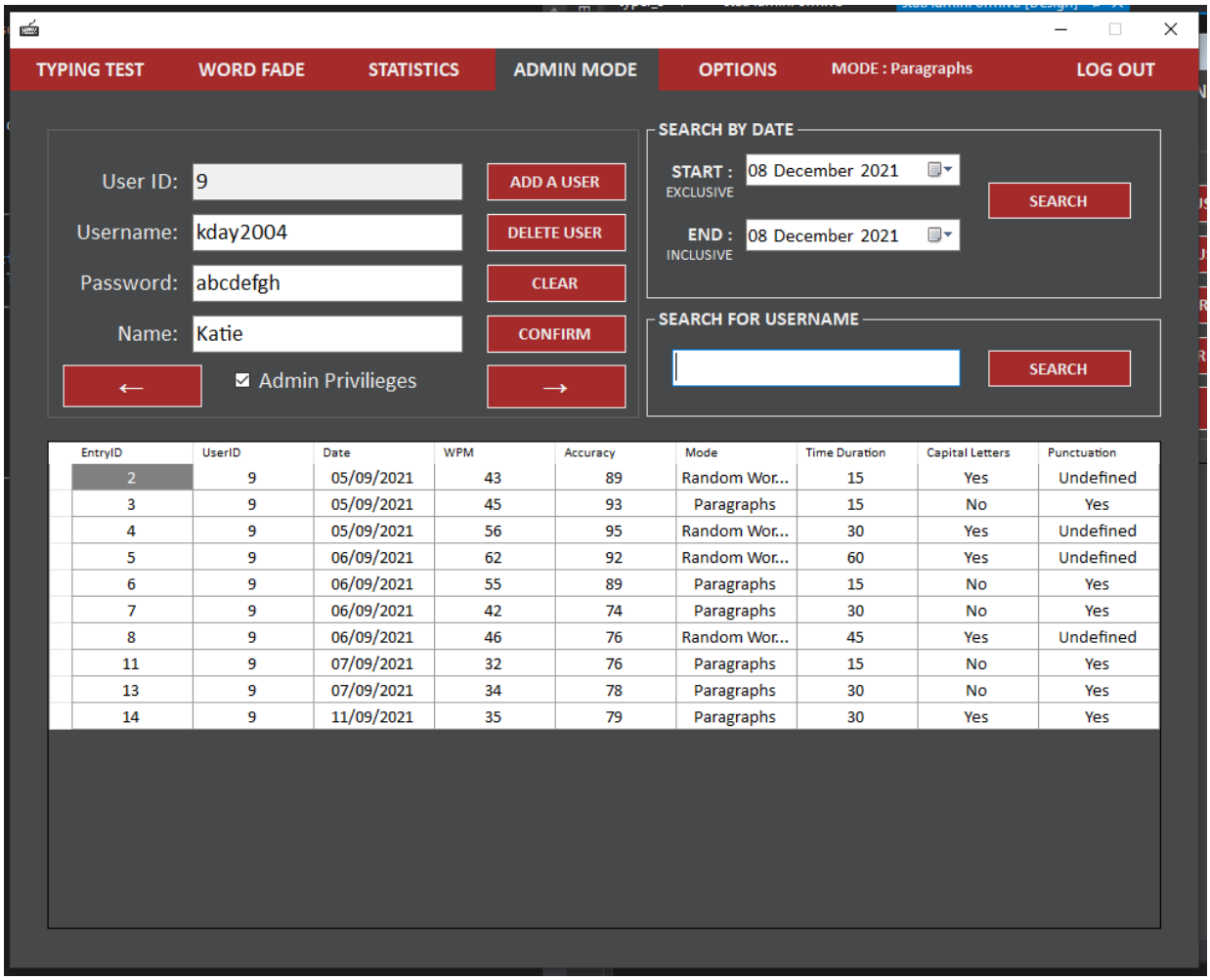
This feature was not addressed in this version.

Test	Result	Evidence	Comments
When choosing an option in the ComboBox, the DataGridVeiwer would be changed in accordingly to the option chosen		Average Accuracy Username / Avg Of WPM b123ABC 86 gT!!aaa 44.75 kday2004 34.93333333...	The act of choosing a leaderboard in the ComboBox and the DataGridViewer changing accordingly is working well. However, there are formatting errors for decimal places needed to be sorted out.

Admin Form

Aesthetic Improvements

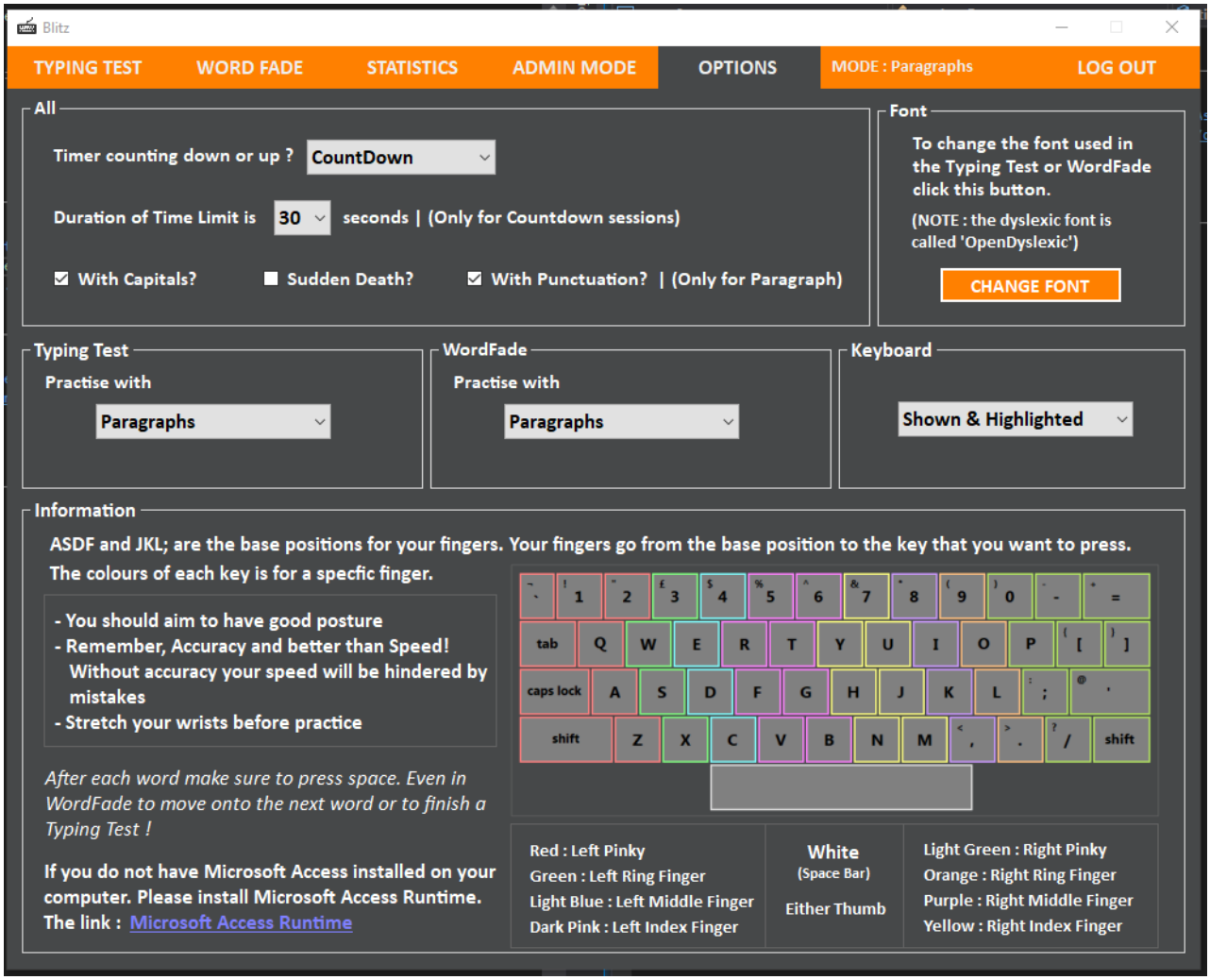
The reasonings for the improvements made have a similar explanation to the 'Version 4 – Login Form' section.



Options Form

Aesthetic Improvements

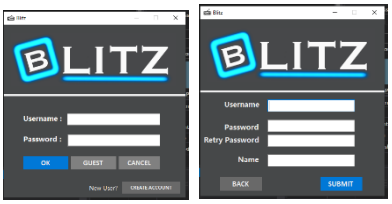
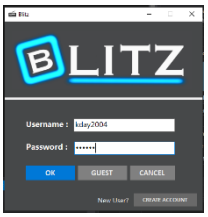
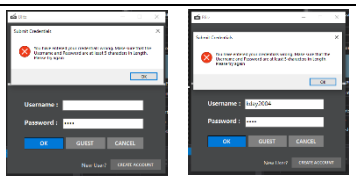
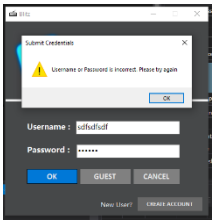
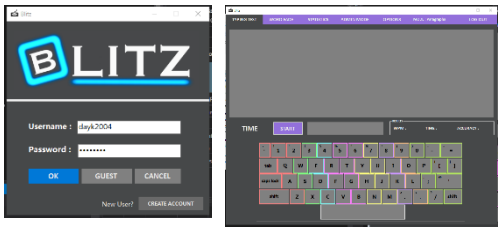
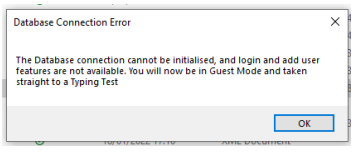
The reasonings for the improvements made have a similar explanation to the 'Version 4 – Login Form' section.

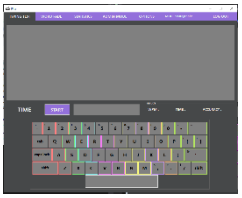
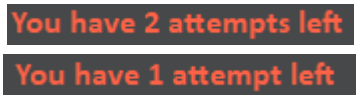


Post Development Testing

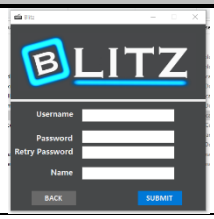
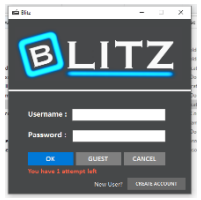
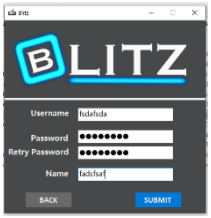
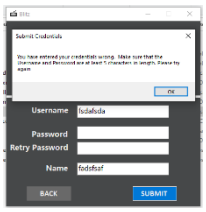
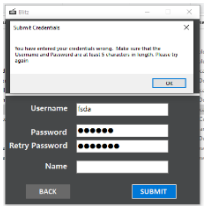
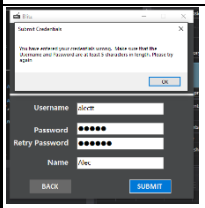
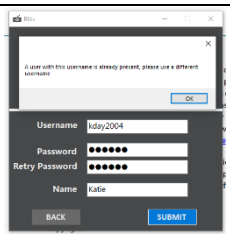
Green: Result is the expected output, Amber: Result is somewhat the expected output, Red: Result is either void or test is not available to be tested. There are annotations relating to the evidence presented in each of the test data. Here, I have tested the function and the robustness of the program.

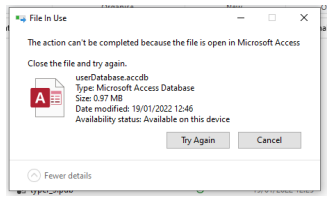
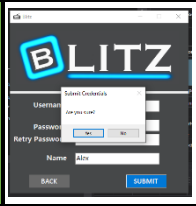
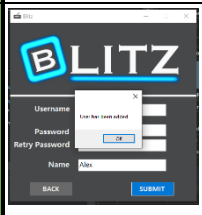
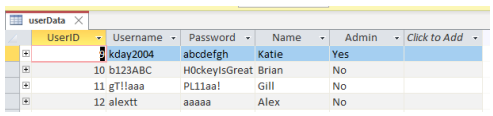
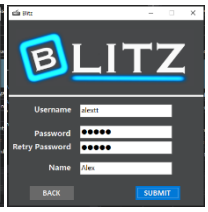
Login Form

Test	Result	Evidence & Comments
'Create Account' button leads to the 'Add User' form		 'Create Account' button successfully closes the first form and opens the second form (Add User Form)
Textboxes can be inputted into by the user		 Form is able to detect and show the correct inputs into the 'Username' Textbox and the 'Password' Textbox
User will not be able to log in unless both input Textboxes are inputted. And if the text is beneath 5 characters.		 Both examples show variants of the same conditions that would cause an error to be shown. Here they are detected, with the warning MessageBox being shown.
When wrong credentials are inputted, it is not a successful login and will not be allowed in		 Inputting the wrong credentials are successfully flagged by the program with the correct MessageBox being shown in response.
When the credentials are correct, the login is successful. The 'Typing Test' form opens		 Correct credentials, closes the Login Form and opens the Typing Form
If database connection is not present, it informs the user		 When launching executable with Access File not present shows this.

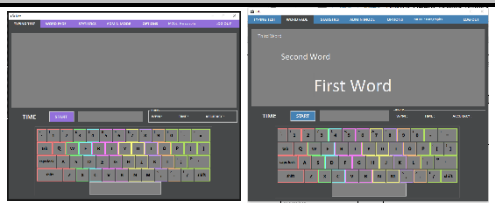
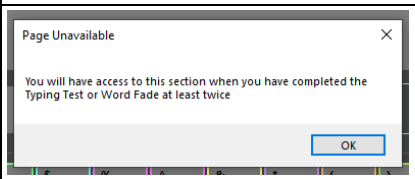
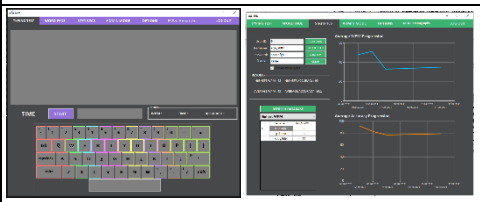
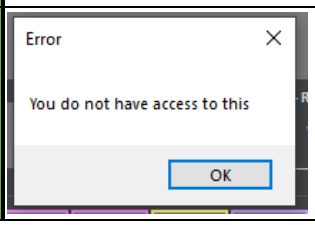
User can access the Main Program without a login by using 'Guest Mode'		  The Typing Test form opens the Login form closes.
When an unsuccessful attempt has been made, a label appears saying how many more attempts are allowed before the form closes		 After the 3rd failed attempt, the program closes.
Program is closed when the user wants it to		I cannot show evidence of this as it would be a screenshot of my background, which is not credible.

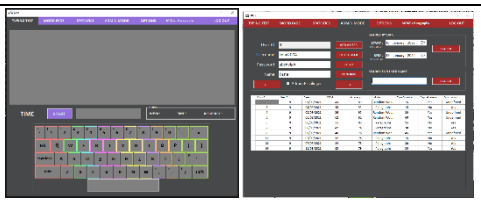
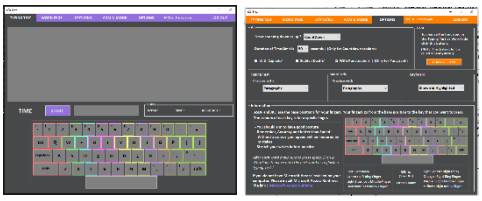
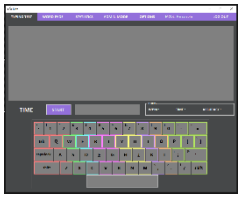
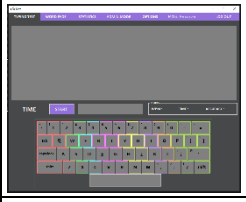
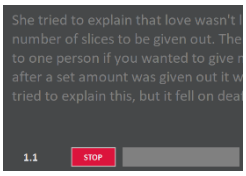
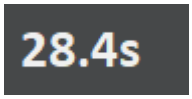
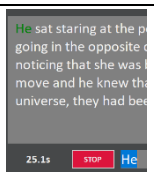
Add User Form

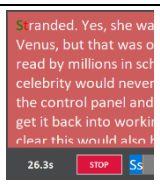
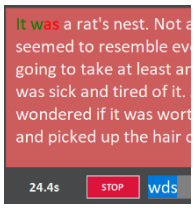
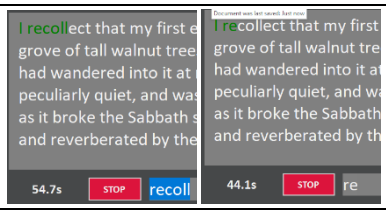
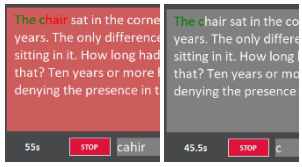
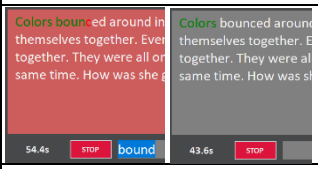
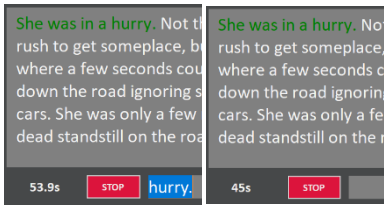
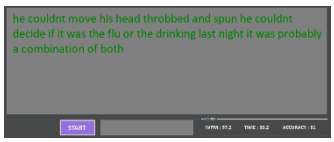
Test	Result	Evidence & Comments
User can go back to Login form without adding a user		  The 'Back' button successfully opens Login Form and closes the Add User Form.
Textboxes can be inputted into by the user		 Form is able to detect and show the correct inputs into the 'Username' Textbox and the 'Password' Textbox
User will not be able to add an account unless all Textboxes are inputted		  Both examples show variants of the same conditions that would cause an error to be shown. Here they are detected, with the warning MessageBox being shown.
'Password' and 'Retry Password' textboxes must have the same input, if not the account will not be added		 The password textboxes do not equal, so a MessageBox is shown instead of adding it to the program.
If the Username wanting to be added is already a pre-existing login then prevent from adding it again		 A MessageBox is shown, informing the user of the error. And the account is not added.

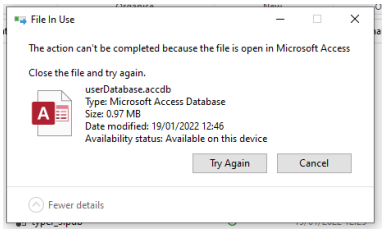
If database connection is not present, it informs the user		 The database connection is checked at the beginning and it then opened for the rest of the time. It cannot be closed whilst the connection is open.
When the other requirements are met, a MessageBox shows to check if the action should be completely		 The MessageBox has been successfully shown after the conditions are met.
When the MessageBox shows after submitting, if the clicked 'Yes' then the account is added		  The account is successfully added to Database and a MessageBox is shown.
When the MessageBox shows after submitting, if clicked 'No' then the account is not added and directed back to the form		  The form goes back to previous state with the inputs retained.

Typing Test Form

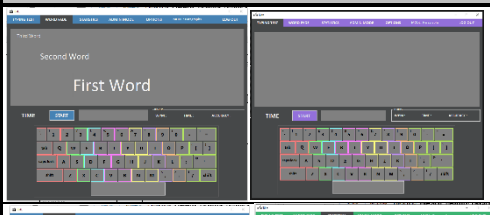
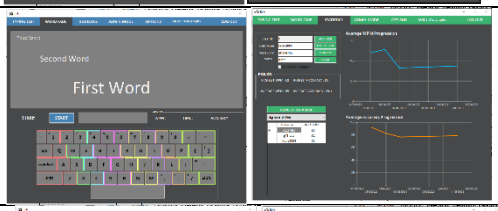
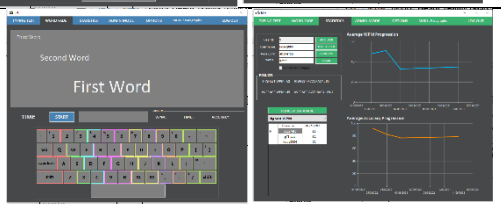
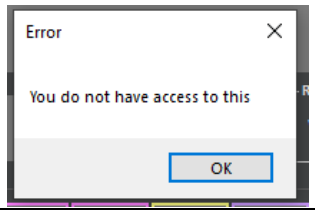
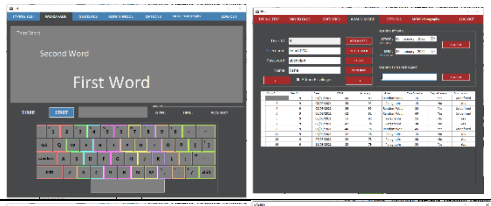
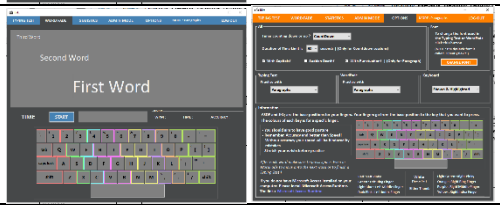
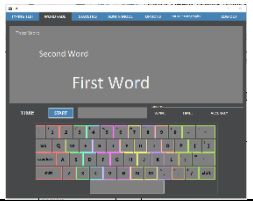
Test	Result	Evidence & Comments
User can navigate to the WordFade form through the menu bar		 Typing Form successfully opened WordFade form
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar		 MessageBox prompt successfully opens
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		 Typing Form successfully opened Statistics form
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar		 MessageBox prompt successfully opens

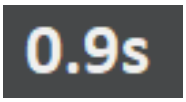
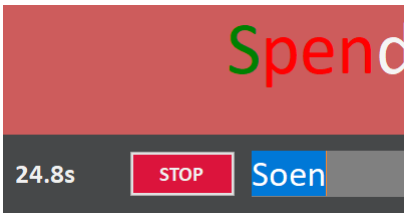
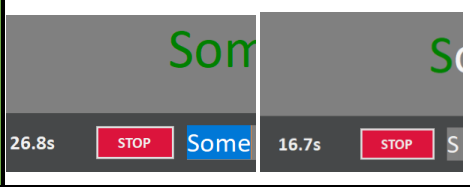
User can navigate to the 'Admin' form when the criteria is met through the menu bar		 Typing Form successfully opened Admin form
User can navigate to the 'Options' form through the menu bar		 Typing Form successfully opened Options form
Extract and Input Textbox is not editable		 I cannot type into the Extract and Input textboxes when a Typing Test has not been started
Before a Typing Test has started, the user cannot input into the Input Textbox		 I cannot type into the Input textbox when a Typing Test has not been started
User can log out of the program		I cannot show evidence of this as it would be a screenshot of my background, which is not credible.
An extract is loaded into the form when a Typing Test is about to start		 Extract successfully loaded in
A count down from 3 starts, when at 0 the Typing Test starts every 0.1s		 Timer countdowns from 3 to 0s
When the Typing Test starts, the user is now able to input into the Input Textbox		 Textboxes now editable
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'		 Button changes back colour to Red
Timer is updated in a label every 0.1s until the end or is stopped		 Timer counts down/up in 0.1 intervals
If the character the user types is correct, change the forecolour for that letter on the extract to Green, keyboard key highlighted is updated		 The character inputted by user is correctly checked and its forecolour changed

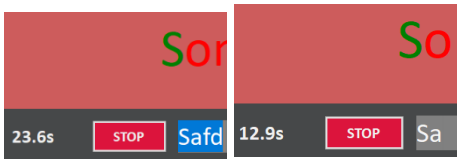
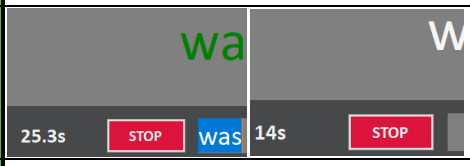
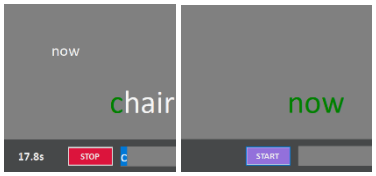
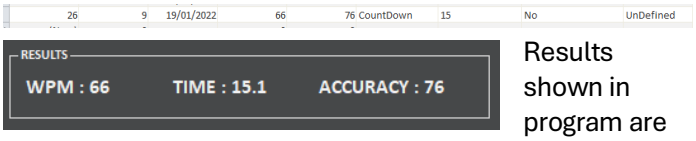
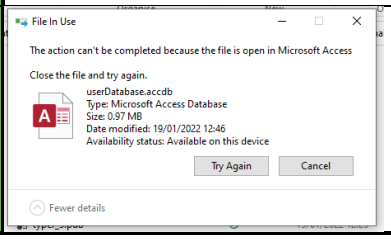
If the character the user types is incorrect, change the forecolour for that letter on the extract to Red		 The character inputted by user is correctly checked and its and back colour forecolour changed
If the user has made errors and does not correct them, make all subsequent characters Red regardless of if they are correct, and do not move onto next word		 The character inputted by user is correctly checked and its and back colour forecolour changed
If the user has did not make any errors but backspaces on a Green character, change it to original colour		 The character inputted by user is correctly checked and its forecolour changed
If the user has made errors, and backspaces to correct it, make the Red character now original colour		 The character inputted by user is correctly checked and its and back colour forecolour changed
If the user tries to backspace to a previous word, stop at the start of the current word		 Program will not let me backspace onto the previous word
If the word is correctly typed completely and the user presses space onto the next word, clear the Input Textbox to make room for the new character		 in The Input textbox is successfully cleared and moved onto the next word
If the extract has been completed, stop the Test, calculate and show the results		 Results successfully outputted (bottom right of photo)
If the Time Limit is reached, stop the Test, calculate and show the results		 Results successfully calculated and outputted
Once results are calculated and the user is not in Guest Mode, input them into the database corresponding to the user		 Results shown in program are inputted as a record in [entryData] table
Once results are calculated and the user is in Guest Mode, do not input results into the database		I cannot show evidence of this as it would be a screenshot of the table and the absence of the record (the entry), which is not credible
If the 'Stop' button is clicked, stop the Typing Test		 Test has been stopped

If the database cannot be connected to input the results, send a prompt to the user			The database connection is checked at the beginning and it then opened for the rest of the programs run time.

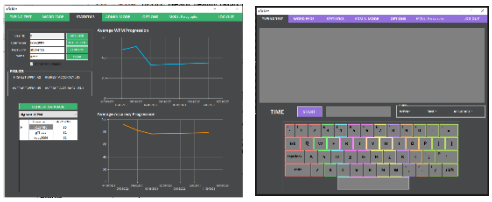
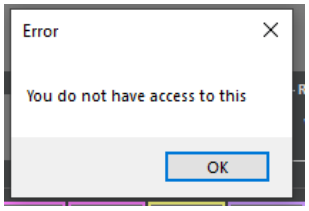
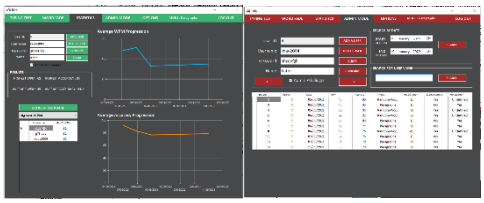
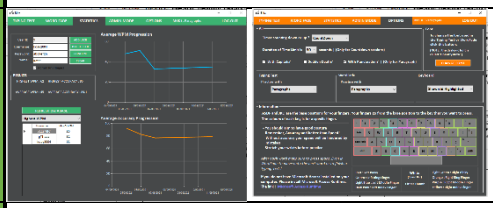
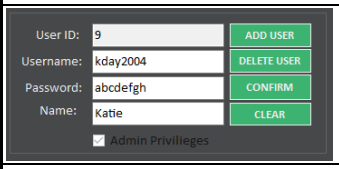
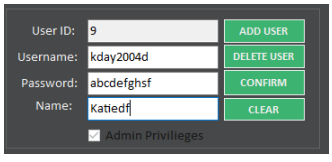
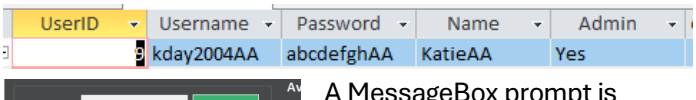
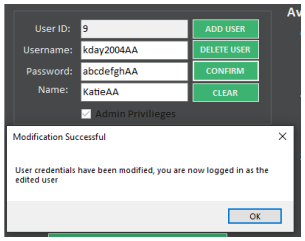
WordFade Form

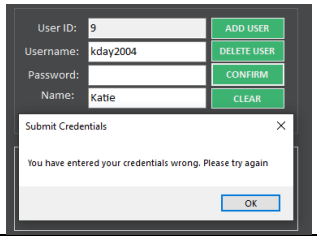
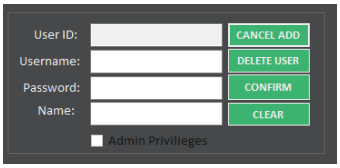
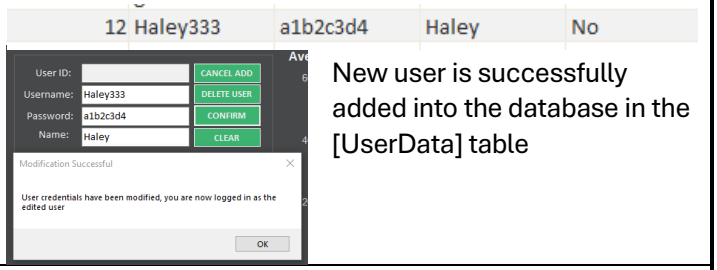
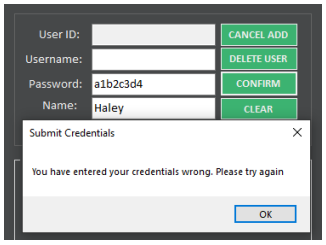
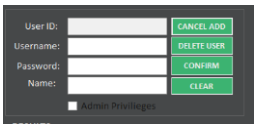
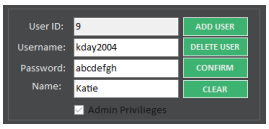
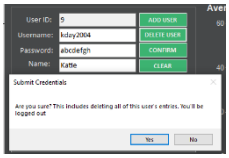
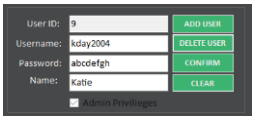
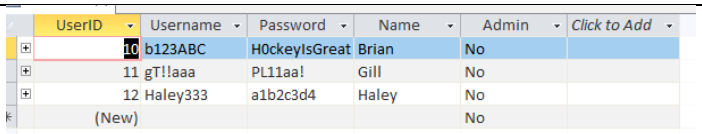
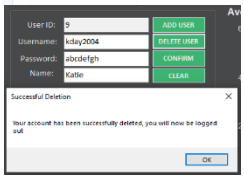
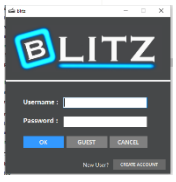
Test	Result	Evidence & Comments	
User can navigate to the 'Typing Test' form through the menu bar			WordFade Form successfully opened Typing form
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar			WordFade Form successfully opened Statistics form
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar			WordFade Form successfully opened Statistics form
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar			MessageBox prompt successfully opens
User can navigate to the 'Admin' form when the criteria is met through the menu bar			WordFade Form successfully opened Admin form
User can navigate to the 'Options' form through the menu bar			WordFade Form successfully opened Options form
Extract Textbox is not editable			I cannot type into the Extract textbox when a WordFade Test has not been started

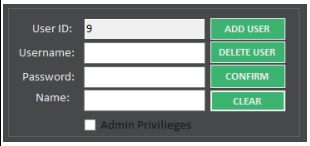
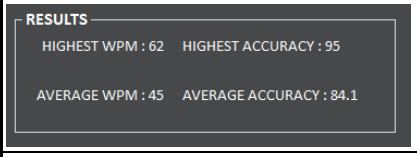
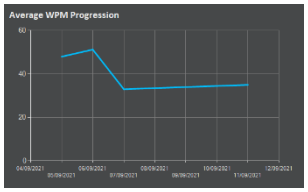
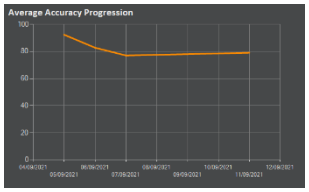
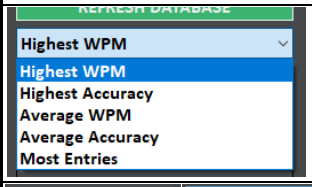
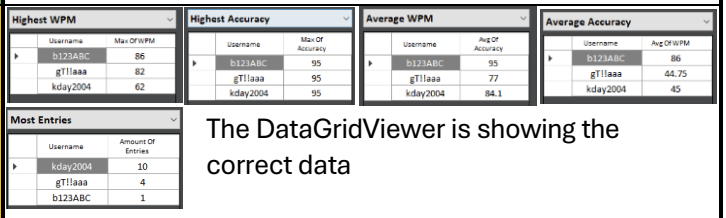
Before a Typing Test has started, the user cannot input into the Input Textbox		 I cannot type into the Input textbox when a WordFade Test has not been started
User can log out of the program		I cannot show evidence of this as it would be a screenshot of my background, which is not credible.
An extract is loaded into the program, and the first 3 words are in the Textboxes when a WordFade is about to start		 First 3 words of the extract successfully loaded into the form.
A count down from 3 starts, when at 0 the Typing Test starts every 0.1s		 Timer countdowns from 3 to 0s
When the WordFade starts, the user is now able to input into the Input Textbox		 Textboxes now editable
When the 'Start' button is clicked, change the colour to Red and update the text to 'Stop'		 Button changes back colour to Red
Timer is updated in a label every 0.1s until the end or is stopped		 Timer counts down/up in 0.1 intervals
If the character the user types is correct, change the forecolour for that letter on the extract to Green, keyboard key highlighted is updated		 The character inputted by user is correctly checked and its forecolour changed
If the character the user types is incorrect, change the forecolour for that letter on the extract to Red		 The character inputted by user is correctly checked and its forecolour and back colour changed
If the user has made errors and does not correct them, make all subsequent characters Red regardless of if they are correct, and do not move onto next word		 The character inputted by user is correctly checked and its forecolour and back colour changed
If the user has did not make any errors but backspaces on a Green character, change it to original colour		 The character inputted by user is correctly checked and its forecolour changed

If the user has made errors, and backspaces to correct it, make the Red character now original colour		 The character inputted by user is correctly checked and its forecolour changed
If the user tries to backspace to a previous word, stop at the start of the current word		 Program will not let me backspace onto the previous word
If the word is correctly typed completely and the user presses space onto the next word, clear the Input Textbox to make room for the new character. Move the two other words down and load in the third word		 The word are successfully laoded into the textboxes and the Input textbox has been successfully cleared.
If near the end of the essay, 3rd and/or 2nd textboxes will have no string in it (to signify end of essay)		 Textboxes are successfully showing no input to signify the end of the extract
If the extract has been completed, stop the Test, calculate and show the results		 Results successfully outputted (right of photo)
If the Time Limit is reached, stop the Test, calculate and show the results		 Results successfully outputted
Once results are calculated and the user is not in Guest Mode, input them into the database corresponding to the user		 Results shown in program are inputted as a record in [entryData] table
Once results are calculated and the user is in Guest Mode, do not input results into the database		I cannot show evidence of this as it would be a screenshot of the table and the absence of the record (the entry), which is not credible
If the 'Stop' button is clicked, stop the Typing Test		 Test has been stopped
If the database cannot be connected to input the results, send a prompt to the user		 The database connection is checked at the beginning, and it then opened for the rest of the programs run time.

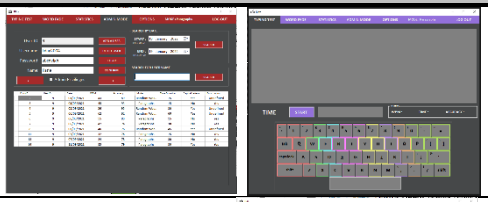
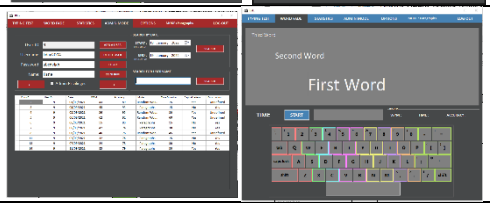
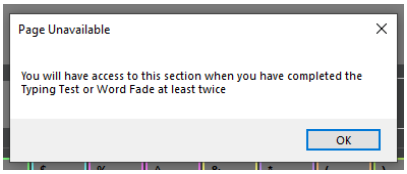
Statistics Form

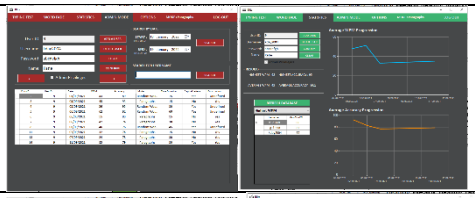
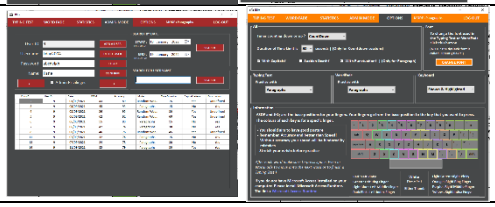
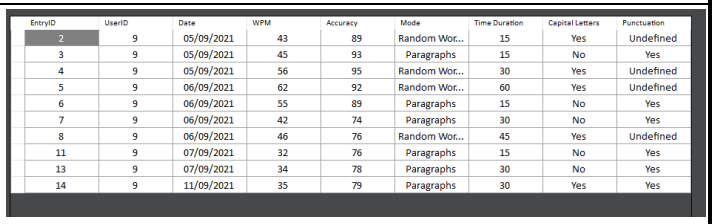
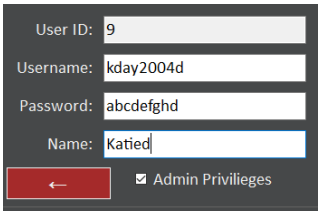
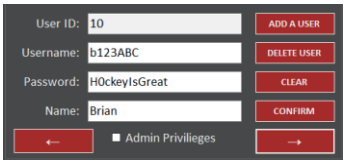
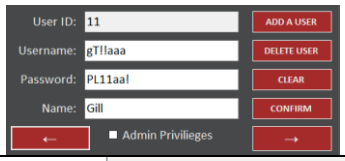
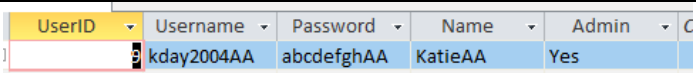
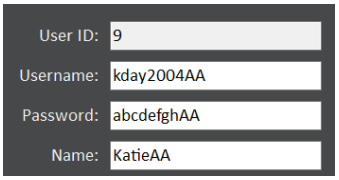
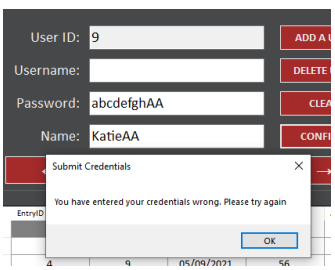
Test	Result	Evidence & Comments
User can navigate to the 'Typing Test' form through the menu bar		 Statistics Form successfully opened Typing form
User can navigate to the 'WordFade' form through the menu bar		 Statistics Form successfully opened WordFade form
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar		 MessageBox prompt successfully shown
User can navigate to the 'Admin' form when the criteria is met through the menu bar		 Statistics Form successfully opened Admin form
User can navigate to the 'Options' form through the menu bar		 Statistics Form successfully opened Options form
User can log out of the program		I cannot show evidence of this as it would be a screenshot of my background, which is not credible.
Loads user details from database into the Textboxes		 User's credentials are successfully loaded into the correct textboxes
User can edit 'Username', 'Password' and 'Name' Textboxes but not 'UserID' Textbox or 'Admin Privileges' Checkbox		 User's credentials textbox can be inputted into
'Add User' button was not clicked before and the user edits their credentials with each textbox having text in it, 'Confirm' button is clicked and the credentials are updated if all the inputs are above 5 characters		  A MessageBox prompt is shown to the user and the updated credentials are in the database
'Add User' button was not		A MessageBox prompt is shown to the user and the

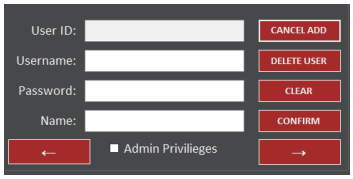
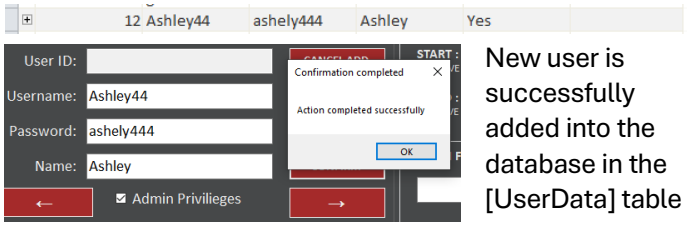
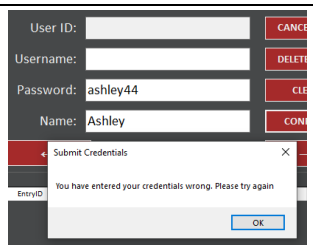
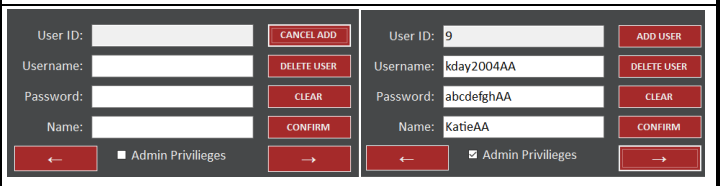
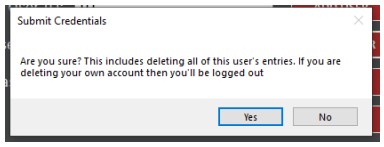
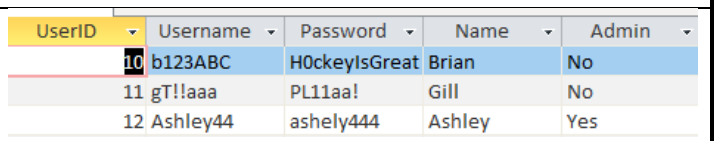
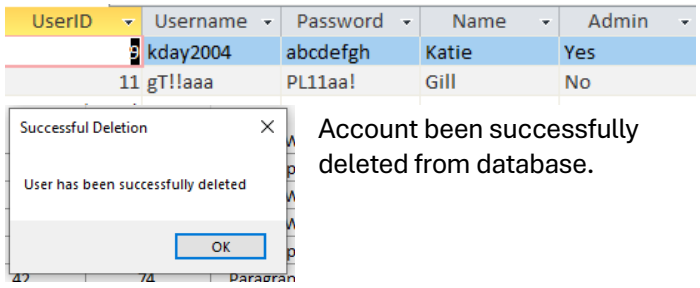
clicked before and the user edits their credentials and 'Confirm' button is clicked. However, a textbox does not have text in it so does not update		 updated credentials are in the database
When 'Add User' is clicked, clear the textboxes and change the button to 'Stop Add User'. It's now in Add User Mode		 All textboxes and the checkbox are cleared of input and the button text has changed to 'Stop Add User'
User goes into Add User mode, the user adds in credentials in every textbox and clicks 'Confirm' to add user		 New user is successfully added into the database in the [UserData] table
User goes into Add User mode, the user adds in credentials clicks 'Confirm' to add user. If the textboxes have no input or below 5 character the action will not be completed		 The correct MessageBox prompt is shown to the user and the user is not added
User wants to go out of Add User Mode		  Has succesfully gone out of 'Add User'
User wants to delete user by click 'Delete' button		 The correct MessageBox prompt appears
User wants to delete user by click 'Delete' button, and clicks 'No' to MessageBox		 Goes back to normal state
User wants to delete user by click 'Delete' button, and clicks 'Yes' to MessageBox		   'XXXX2004' account has been deleted from the database. The correct MessageBox prompt is shown and the user is redirected to the Login form.

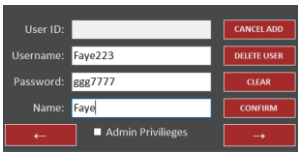
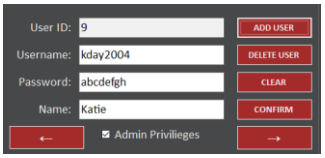
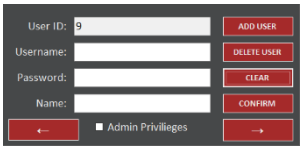
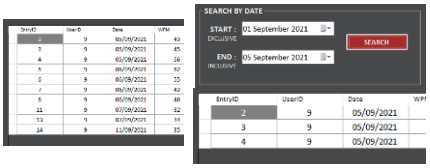
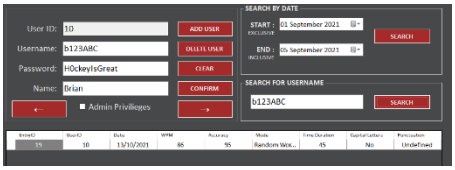
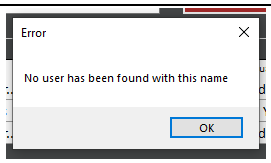
If the clicks the 'Clear' button, clear the Textboxes		 Text in the textboxes are successfully cleared. Apart from UserID which is not editable
Loads 'Highest WPM', 'Average WPM', 'Highest Accuracy', 'Average Accuracy' into labels		 The labels contain correct data fetched from the current accounts entries
Load Graph controls to 'Average WPM / Date' and 'Average Accuracy / Date' as Line Graphs for that user		  Correct graphs are loaded into program with correct data
ComboBox would have the options of 'Highest WPM', 'Average WPM', 'Highest Accuracy', 'Average Accuracy' and 'Most Entries'		 The ComboBox holds the correct entries
When choosing an option in the ComboBox, the DataGridVeiwer would be changed in accordingly to the option chosen		 The DataGridViewer is showing the correct data However, there are formatting errors for decimal places needed to be sorted out.

Admin Form

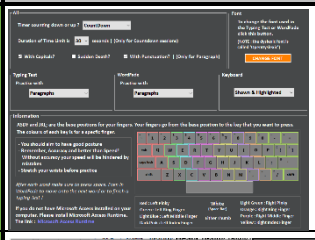
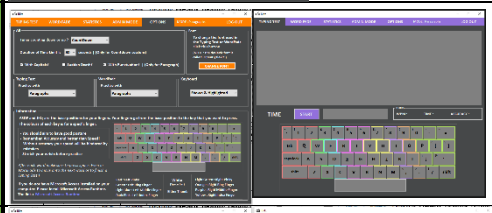
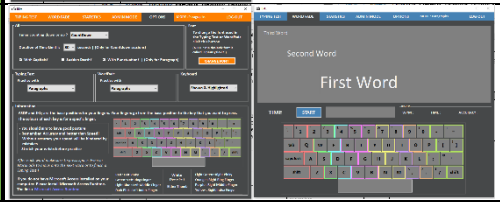
Test	Result	Evidence & Comments
User can navigate to the 'Typing Test' form through the menu bar		 Admin Form successfully opened Typing form
User can navigate to the 'WordFade' form through the menu bar		 Admin Form successfully opened WordFade form
User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar		 MessageBox prompt successfully opens

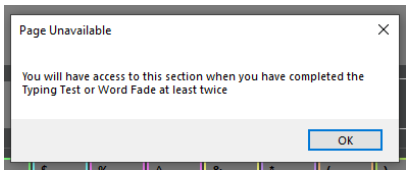
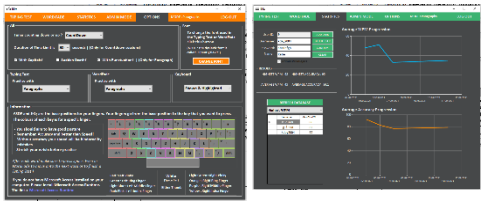
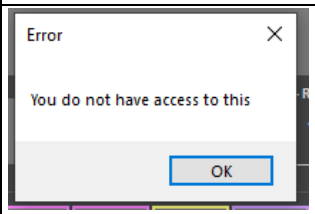
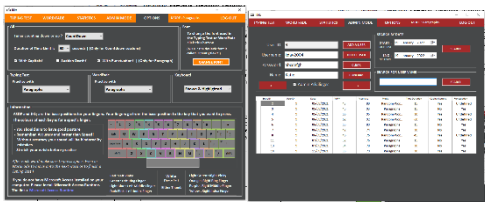
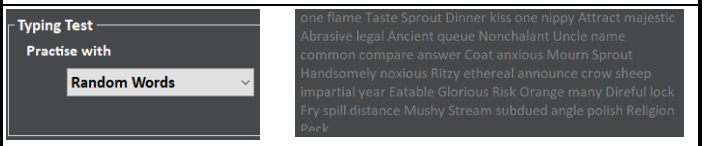
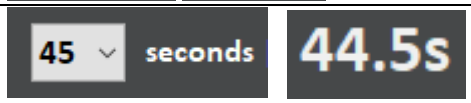
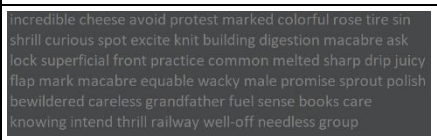
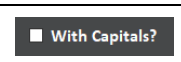
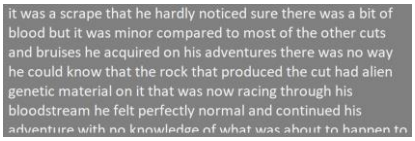
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		 Admin Form successfully opened Statistics form
User can navigate to the 'Options' form through the menu bar		 Admin Form successfully opened Options form
User can log out of the program		I cannot show evidence of this as it would be a screenshot of my background, which is not credible.
Loads the first user details from database into the Textboxes and DataGridVeiwer		 DataGridViewer successfully loads in entries for that user
User can edit 'Username', 'Password', 'Name' Textboxes and 'Admin Privileges' Checkbox but not 'UserID' Textbox		 User's credentials textbox can be inputted into
User can go to next user by clicking the '→' button		 Successfully goes to next user and loads credentials into the textboxes
User can go to the previous user by clicking the '←' button		 Successfully goes to previous user and loads credentials into the textboxes
'Add User' button was not clicked before and the user edits their credentials with each textbox having text in it, 'Confirm' button is clicked and the credentials are updated		  Updated credentials are in the database
'Add User' button was not clicked before and the user edits their credentials and 'Confirm' button is clicked. However, a textbox does not have text in it so does not update		 The MessageBox prompt is correctly shown

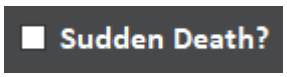
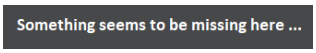
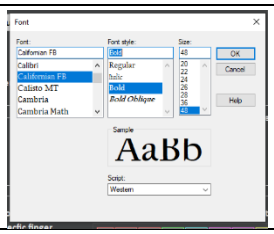
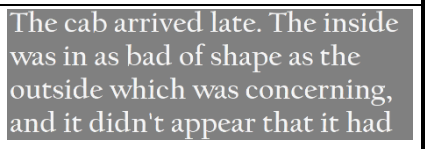
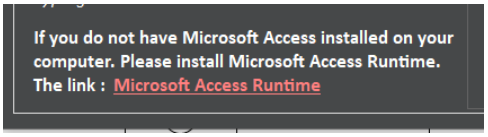
When 'Add User' is clicked, clear the textboxes and change the button to 'Stop Add User'. It's now in Add User Mode		 All textboxes and the checkbox are cleared of input and the button text has changed to 'Stop Add User'
User goes into Add User mode, the user adds in credentials in every textbox and clicks 'Confirm' to add user		 New user is successfully added into the database in the [UserData] table
User goes into Add User mode, the user adds in credentials clicks 'Confirm' to add user. If the textboxes have no input or below 5 character the action will not be completed		 The correct MessageBox prompt is shown to the user and the user is not added
User wants to go out of Add User Mode		 Has succesfully gone out of 'Add User'
User wants to delete user by click 'Delete' button		 The correct MessageBox prompt shown
User wants to delete user which is their own login by click 'Delete' button, and clicks 'Yes' to MessageBox		 'XXXX2004' account has been deleted from the database. The correct MessageBox prompt is shown and the user is redirected to the Login form.
User wants to delete user which is not their own login by click 'Delete' button, and clicks 'Yes' to MessageBox		 Account been successfully deleted from database.
User wants to delete user by click 'Delete' button, and clicks 'No' to MessageBox		 Returns back to the original state

If the user is in Add User mode and clicks the 'Cancel Add' button, stop it		  successfully clears and reloads credentials
If the clicks the 'Clear' button, clear the Textboxes		 Successfully clears textboxes of text
User can specify the dates the entries are between entries in DataGridViewer		 DataGridViewer is updated to show the relevant dates
User can search for a specific username, and it will be loaded into the Textboxes and DataGridVeiwer		 Search is successful and account details and entries are shown
User tries to search for a username not in the program then prompt the user		 The correct MessageBox prompt is shown

Options Form

Test	Results	Evidence & Comments
All pre-existing settings are loaded into the controls		 The options made in the current session and then a user goes to a different page then comes back to the Options form. Options are preserved.
User can navigate to the Typing Test form through the menu bar		 Options Form successfully opened Typing form
User can navigate to the WordFade form through the menu bar		 Options Form successfully opened WordFade form

User cannot navigate to the 'Statistics' form if they have more than one entry and they are not in Guest Mode through the menu bar		 MessageBox prompt successfully opens
User can navigate to the 'Statistics' form, provided the criteria is met through the menu bar		 Options Form successfully opened Statistics form
User cannot navigate to the 'Admin' form when the account does not have Admin Privileges or that they are in Guest Mode through the menu bar		 MessageBox prompt successfully shown
User can navigate to the 'Admin' form when the criteria is met through the menu bar		 Options Form successfully opened Admin form
User can log out of the program		I cannot show evidence of this as it would be a screenshot of my background, which is not credible.
When the type of extract for Typing Test ComboBox is changed, it will be changed in the Typing Test		 ComboBox change successfully determines extract
When the type of extract for WordFade ComboBox is changed, it will be changed in the WordFade test		 ComboBox change successfully determines extract
When the Time Duration ComboBox is changed, it will be changed in the Typing Test and WordFade		 Time duration successfully changes
When the CountDown or CountUp ComboBox is changed, it will be changed in the Typing Test and WordFade		 The timer is changed dependent on the ComboBox option.
When the Capitals checkbox is changed, it is changed for Typing Test and WordFade		  Extract is changed
When the Punctuation checkbox is changed, it is changed for Typing Test and WordFade (this only applies to 'Paragraph' extracts		  Extract is changed

When the Sudden Death checkbox is changed, it is changed for Typing Test and WordFade		 Tests are changed to stop after one mistake
When the Keyboard ComboBox has changed, it will be changed in the Typing Test and WordFade		   Keyboard is changed accordingly
User can change the font type, style and size for the Typing Test and WordFade		  Font correctly changes
User can click a link to be directed to a website to download Microsoft Access Runtime		 Browser opens to correct website

Evaluation

To What Extent is my Success Criteria Met?

Here, I have the success criteria I defined in my Analysis section and have evaluated the completion of them using my final developed solution. Below are the meanings for the colour given to each criterion.

Green: Success Criteria completed, nothing more needs to be done
Amber: Success Criteria has been attempted, however there are aspects that need to be addressed
Red: Success Criteria is not present in my final solution

①	**User can successfully login with a specified username and password? Does the program show an error if there is unsuccessful login attempt? Does the program exit if attempts exceed 3 fails?**	**Green**
The 'Post Development Testing' section using the 'Test Data' provides evidence of the connections to the database, checking user inputs against credentials and performing the right behaviours for either a successful or failed login, and records the number of login attempts. The evidence is found in 'Post Development Testing' in the 'Login Form' section. All the features listed, complete the Success Criterion with no further development needed.		

②	**A new user can input their credentials, and it is successfully added to the database? Does the program validates the inputs e.g., no input in a textbox? Does the program show an error when a username is taken?**	**Green**
The 'Post Development Testing' section using the 'Test Data' provides evidence of the connections to the database, checking user inputs against credentials and performing the right behaviours for either a successful or failed addition of account to the database, and performs validation checks. These validation checks include the length of 'Username' and 'Password' being at least 5 characters and making sure all textboxes have an input. The evidence is found in 'Post Development Testing' in the 'Add User Form' section. All the features listed complete the Success Criterion with no further development needed.		

③	**Is there a guest mode for the user which doesn't add entries to the database?**	**Green**
There is the option for the user to go into 'Guest Mode' using button in the 'Login Form' and the program does not add entries into the database. The user is not allowed into the 'Statistics form' and the 'Admin form' as they are not logged into an account. The evidence is found in 'Post Development Testing' is predominately in the 'Login Form' section. All the features listed complete the Success Criterion with no further development needed.		

④	**After a successful login, does the main window open as the first form?**	**Green**
Yes, the main window 'Typing Test form' opens. The evidence is found in 'Post Development Testing' in 'Login Form' section. This feature completes the Success Criterion with no further development		

needed.

5	**Can the user navigate to the different forms within the program with ease? And these forms show up on the user's request?**	**Green**

There is a menu bar in the same location for all forms (apart from the 'Login form' and the 'Add User form' as they are not a part of the main program). The menu bar includes a button for each form. There is validation for the 'Statistics form' if the user has no entries or in Guest mode. And the 'Admin form' has validation against users who do not have admin privilege. The evidence is found in 'Post Development Testing' at the start of the test data for all main program forms. These features complete the Success Criterion with no further development needed.

6	**Can the user choose the type of extract they will be tested on e.g., Paragraphs, Random Words & Alphabet (only for WordFade)?**	**Amber**

In the 'Options form' there are two ComboBoxes (one for Typing Test and other for WordFade, with the options corresponding to calling certain functions that access either 'paragraphs.txt', 'randomWords.txt' or 'alaphbet.txt'. The evidence is found in 'Post Development Testing' at the start of the test data in the 'Options form'. These features complete the Success Criterion. However, I have graded this criterion as an Amber because I feel that I could've diversified further the types of extracts. E.g., literature, genres of words and languages. The 3 choices I have integrated are very simple in comparison. Additionally, there are only 1000 words in 'randomWords.txt' with no method of selecting a difficulty of a word.

7	**Can the user choose to eliminate punctuation and/or capitals from the extract?**	**Green**

In the 'Options form' there are two Checkboxes (one for Captials and one for Punctuation) that affect both the Typing Test and WordFade extracts clearly and consistently. The evidence is found in 'Post Development Testing' at the start of the test data in the 'Options form'. These features complete the Success Criterion with no further development needed.

8	**Can the user choose the time duration of the typing test, and whether it is a Countdown or Count Up?**	**Green**

In the 'Options form' there is a ComboBox that has options for '15', '30', '45' and '60' seconds. Which changes the time duration in both the WordFade and the Typing Test. There is a ComboBox for whether the counter should countdown or up which successfully works too. The evidence is found in 'Post Development Testing' at the start of the test data in the 'Options form', 'Typing Test form' and 'WordFade form'. These features complete the Success Criterion with no further development needed.

9	**When a new typing test has started, a new randomised extract is loaded into the program depending on the options chosen above?**	**Green**

Both the 'Typing Test form' and 'WordFade form' fetches a randomised extract from the Text files and

outputs onto the program successfully. The evidence is found in 'Post Development Testing' at the start of the test data in the 'Options form', 'Typing Test form' and 'WordFade form'. These features complete the Success Criterion with no further development needed.

⑩	**After each test, is the WPM and Accuracy calculated shown to the user and inserted into the database as a new entry?**	**Amber**

The results are calculated and shown to the user. They are successfully added as an entry into the [entryData] table in the database. However, I have graded this an Amber because I believe that the Accuracy results can be unreliable. Accuracy is calculated as the percentage of (total number of faults the use has made in the test / total number of characters successfully typed).

```
accuracy = (100 - 100 * (totalFaults / countCharEssay))
```

However, if the user makes more mistakes than successfully keystrokes then the percentage becomes a negative number. This situation is better than having the percentage become over 100% however, it can be confusing for the user to gain an understanding of what happened. Additionally, I can skew the Accuracy / Date graph located in the 'Statistics form'.

Another aspect that needs to be addressed is the calculation of the time duration if the user has completed the whole essay before the timer has completed. In the final solution the full time duration is inserted in the entry and not the time the user has done it in. This is misleading as the entry would show the wrong time duration.

The evidence is found in 'Post Development Testing' in the 'Typing Test form' and the 'WordFade form'.

⑪	**Can the user choose to do a Typing Test or WordFade?**	**Green**

The program allows the user to choose to do either feature by clicking the 'Typing Test' or 'WordFade' button in the menu bar in all main program forms. There are separate forms for a Typing Test and a WordFade. The evidence is found in 'Post Development Testing' at the start of the test data for all main program forms. These features complete the Success Criterion with no further development needed.

⑫	**Is there an option for Sudden Death mode?**	**Green**

There is a checkbox located in the 'Options form' that toggles the mode. It is successfully registered by the program and the Tests end after a mistake is made. The evidence is found in 'Post Development Testing' in the 'Options form'. This feature completes the Success Criterion with no further development needed.

⑬	**Is there a Leaderboard that the user can access and see the top scores in order from other users in the database, and the user can choose from?**	**Amber**

In the 'Statistics form' there is a ComboBox and a DataGridViewer beneath it. The ComboBox is used to determine the type of leaderboard and the DataGridViewer presents the data. I have graded this an Amber because even though the core fundamentals of the success criteria have been completed.

When data is collated and shown to the user, the formatting is not present to limit floating numbers to 2 decimal places. E.g. if the averaged result of WPM comes to a recurring decimal, this number will be shown. This does not look pleasing and would be better with formatting to make it easier for the user to interpret and understand the results. The evidence is found in 'Post Development Testing' in the 'Statistics form' and is the last test data.

14	**Is there a form where the user can see each users' entries, have the ability to search through those entries? Can this form only be accessed through Admin Privileges where the program checks for these privileges?**	**Green**

This criterion is the sole focus of the 'Admin form'. Where there is a DataGridViewer to show a user's entries and DateTimePicker's used to make queries. Each main program form checks whether the current user has Admin Privileges when the user pressed the 'Admin' button in the menu bar. The evidence for the validation of Admin Privileges is found in all main program forms test data. And the other part is found in 'Post Development Testing' in the 'Admin form'. This feature completes the Success Criterion with no further development needed.

15	**Can users change the font type, style and size used in the Typing Test/Word? Do they have the option for the font of the program to be in the form of 'OpenDyslexic'?**	**Amber**

In the 'Options form' there is a button called 'Change Font' that opens a Font Dialog. This changes the font type, style and size of the textboxes holding the extracts in 'Typing Test form' and the 'WordFade Form'. Unfortunately, the 'OpenDyslexic' font is not available in default Windows, so it must be downloaded by the user. I have included the font file in the program files but not added instructions in the 'Options form' as to how to do this, this is why I am grading this an Amber. Additional to this, the font changes only affect the textboxes and not the menu bar and other buttons within the program. This still can be disorientating for the user when navigating around the form.

16	**Can the user see a graph of WPM / Date and Accuracy / Date and see their progress throughout their time using the program?**	**Green**

In the 'Statistics form' there are two separate Chart Controls which take the average result for each date and plot it onto a line graph using the entries for that user in the database. The evidence is found in 'Post Development Testing' at the start of the test data for all main program forms. These features complete the Success Criterion with no further development needed.

17	**Can the user see a keyboard layout on the screen, with a specific key highlighted when that character needs to be pressed next? Can the user turn this off?**	**Amber**

In the 'Typing Test form' and the 'WordFade form' the keyboard graphic is shown. With a specific letter or number highlighting corresponding to the next character needing to be pressed. In the 'Options form' there is a ComboBox to turn off the highlighting and to not show the keyboard entirely. The evidence is found in 'Post Development Testing' in the 'Typing Test form', 'WordFade form' and the 'Options form'.

However, I have graded this an Amber because the keyboard does not highlight keys for punctuation

characters. This element was not covered in my Test Data but is still an important feature missed.

⑱	**Do users have the ability to modify and delete their own accounts?**	**Green**

In the 'Statistics form' the user can modify and delete their own account. There are validation checks to make sure the new modifications follow the rules of the database. The evidence is found in 'Post Development Testing' in the 'Statistics form'. This feature completes the Success Criterion with no further development needed.

⑲	**Can users with Admin Privileges edit and delete any other user? Can they give other user Admin Privileges too?**	**Amber**

In the 'Admin form' the user can modify and delete anyone else's account, there is a checkbox which can give a user Admin Privileges. However, I have graded this criterion an Amber because I have not validated the circumstance where the Admin deleted their own account when they are the last Admin left. This would mean the 'Admin form' is locked out from anyone as no one has the Admin Privileges to access it.

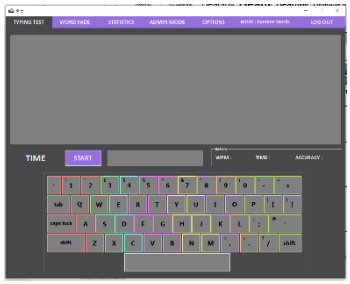
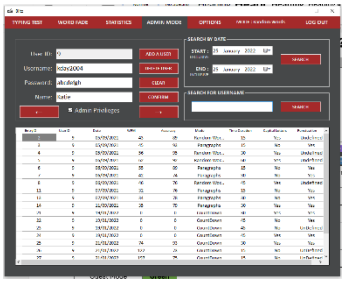
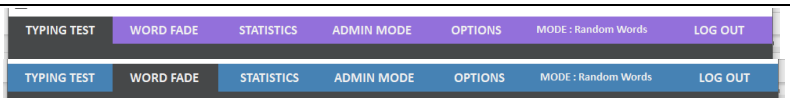
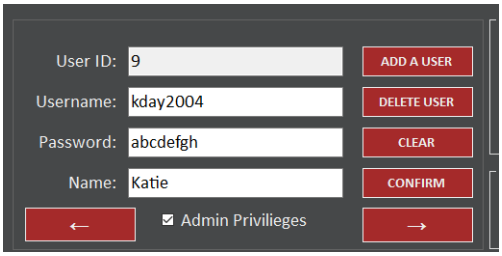
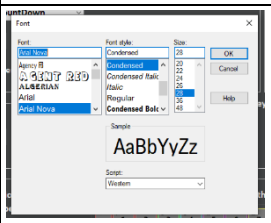
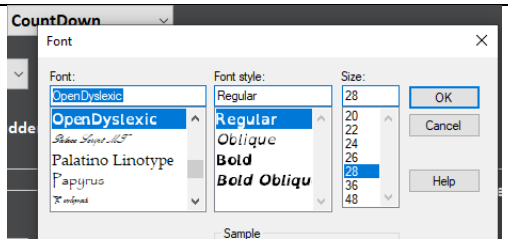
⑳	**Do the users have a way to download Microsoft Access Runtime to then have access to the database?**	**Green**

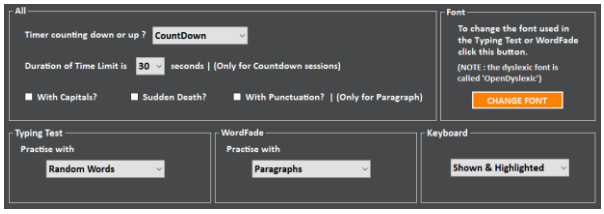
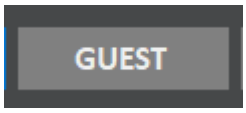
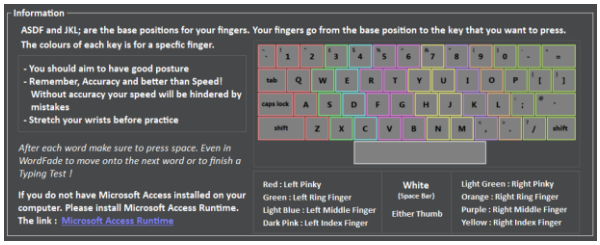
In the 'Options form', there is information on the file and where to download it. The evidence is found in 'Post Development Testing' in the 'Options form'. This feature completes the Success Criterion with no further development needed.

How can the Success Criteria be Met in the Future?

No.	Comments on How to Fix the Criteria
⑥	To solve the issue of lack of extract, genre, and word types. I can add the whole dictionary to multiple text files. This would be divided into 'Easy', 'Medium' and 'Hard' difficulties that the user can choose from. This can be in terms of how common a word is in the dictionary. Additionally, I can add books into the programs and be selected depending on the genre.
⑩	I will change the calculation involving 'accuracy' variable to (number of correctly typed keys/total keys typed), as a percentage. This will give the user a better understanding and will stop the value from going negative.
⑬	I can change the DefaultCellStyle property for the DataGridView, where there is a format section and I can modify the decimal places shown.
⑮	Can add a 'README.txt' that goes through the steps of downloading the font. Additionally, I can add a label in the 'Options form' which directs the user to the text file. I will add functionality to change the font of the whole program.
⑰	Add functionality where the program checks for punctuation and highlights the corresponding key.
⑲	Add validation to check if the account being deleted is the only account with Admin Privileges and stop the action.

To What Extent is my Usability Features Met?

Success Criteria	Results	Comments
Simple Design	**Green**	  I believe that the form is very clear and simple in design. This is clear from the intuitive design and clearly labelled buttons making it straightforward for the user.
Menu Panel	**Green**	 The menu is present throughout all of the main program forms and are clear and consitent throughout. This feature had been completed with no further imrpovement needed.
Big Buttons	**Amber**	 Buttons could be bigger to make it easier to not accidentally click a button next to another. This is especially evident with the 'Admin form' and 'Statistics form' with the buttons used for modifying the account(s).
Big Modifiable Font	**Amber**	 The font dialog opens and correctly updates the font for the Textboxes containing the extracts. However, I believe that due to the font options not affecting the buttons and labels in the form it is hindering the usability of the feature especially when considering a user having trouble navigating the form.
Dyslexic Font Option	**Amber**	 The use of the font dialog when using 'OpenDyslexic' works seamlessly. However, I feel that having a clear method for the user on how to download the font is not present and is hindering the usability of this feature.

Other Options	**Green**	 All the modes/options for the user are located in the 'Options form' and work. This feature had been completed with no further imrpovement needed.
Guest Mode	**Green**	 The Guest Mode has been implemented successfully for people with no account and for people who cannot access the database. As there are methods of users using Typign Test and WordFade without a connection to the database needed. This feature has been completed with no further imrpovement needed. This picture is taken from the 'Login Form'
Information for User	**Amber**	 There is a significant portion in the 'Options form' relating to the improvement of the user's touch typing and fixes to problems within the program. However, there is no information relating to downloading the Dyslexic font or finding the old database if not present within the database.

How can the Usability Features be Met in the Future?

Success Criteria	Comments on How Features can be Met in the Future?
Big Buttons	Overall, I will need to dock all the controls and then be able to give the option to the user to have the program 'maximised'. The original size would be its minimum, however then I can make the button appear bigger on the screen. The user will also have the option to change the size of the program depending on their preference.
Big Modifiable Font	I will create a procedure to handle and modify the font across all the controls within the specific form and have it be called each time a form is loaded.
Dyslexic Font Option	I will add a 'README.txt' in the program files to explain the steps taken to downloading the font. I will include the font file within the program files.
Information for User	Extra information regarding the navigation of the form, downloading of fonts and finding missing database files will be included in the 'README.txt'.

Maintenance

A problem which can arise when maintaining the program is a new Windows OS or a new version of VB.NET. This could cause errors within my code and interrupt a connection with the Microsoft Access database. Therefore, it is important for there to be constant, regular updates to the program to counter the possible changes/bugs.

In terms of the connection to the database, the provider used is 'Microsoft.ACE.OLEDB.12.0'. This provider supports Access products post 2007 to 2016. There is a chance that the user could have a 'newer version' of an Access database they inserted into the program files that the provider is 'out of date' for.

Additionally, for users that do not have Microsoft Office, if Microsoft 365 Access Runtime stops being supported then the features connected to the database cannot be accessed. Instead they will only be able to use 'Guest mode'.

When manipulating the database using SQL I have inconsistently used parametrised SQL queries along with concatenated SQL queries. Parametrised SQL queries are easier to maintain and harder to be subject to errors or SQL Injection attacks due to the increased validation. Therefore I will need to make sure that I retrospectively change each SQL query and enforce the new standard going forward.

In terms of Admin Privileges and access to accounts. There will need to be more levels of access and controls to help the accounts be more secure on who has control over what features. E.g. a 'lower tier' Admin should be allowed to modify the entries for a specific user, this should be only allowed to be used by the Head Admin.

Another aspect that might need to be addressed in the future is the possibility of the link for Microsoft Access Runtime not being supported and the download moving to another location. I need to be aware of the likelihood and be prepared to quicky change the link associated the label in the future.

Limitations

The dimensions of the main program forms is limiting when considering new features and controls being added, due to them being small. In the future, I will make the main program forms be maximised and dock all of the controls so they can keep their locations on the forms. This will additionally solve other limitations such as the small buttons and small font text. However, this will help the future developer be able to add features and controls to the form without the program feeling clutter and making it harder to navigate.

Another limitation is the limited number of extracts that the user can be tested against. There are around 140 sentences and 1000 words. The user will inevitability see the same extract in a test when using the program for a substantial amount of time. This decreases the longevity of my project which is worrying and will cause the user to leave. In the future, I will need to add more extracts to the program in order to increase the longevity of the project.

Another limitation of my project is the questionable method of how I passed variables into different forms. I used the access modifier 'friend' for the variables that I wanted to pass. However, this method is open to accidental changes in values. In the future, I will change the access modifiers will add get and set methods to change/access the variables in a safer and more secure way.

Code for Versions

Version 1

Login Form

```
Imports System.Data.OleDb

Public Class loginForm

    Friend dataUsername As String = "XXXX" 'All friend variables are global that will be used across the form
    Friend dataName As String = "XXXX"
    Friend dataAdmin As Boolean
    Friend Shared dataUserID As String
    Friend timeDuration = "30"
    Friend countType = "CountDown"
    Friend suddenDeath = False
    Friend extractType = "Paragraphs"
    Friend extractWordFadeType = "Paragraphs"
    Friend capitals = True
    Friend punctuation = True
    Friend showKeyboard = "ShowHighlight"
    Friend entriesPresent = False
    Friend globalFont As New Font("Calibri", 28, Font.Style.Regular)
    Friend attemptCount = 3


    Sub UpdateLabelMode(formNow) 'Fetches the mode and updates the label, label is present in every page and is referenced there too
        formNow.modeLabel.Text = "MODE : " & extractType
    End Sub


    Sub LogOut() 'When logging out, the variables with data pertaining to the login is cleared
        dataUserID = ""
        dataUsername = ""
        dataName = ""
        dataAdmin = False
    End Sub

    Sub ClearTextBox() 'After each entry or login the textbox is cleared of inputs
        usernameTextBox.Clear()
        passwordTextBox.Clear()
    End Sub

    Sub deductAttempt() 'Deducts an entry from the counter and updates/shows the text label
        attemptCount -= 1
        Select Case attemptCount
            Case 2
                attemptsLabel.Visible = True
                attemptsLabel.Text = "You have 2 attempts left"
            Case 1
                attemptsLabel.Text = "You have 1 attempt left"
            Case 0
                Me.Close() 'After 3 attempts the program closes
        End Select
    End Sub


    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Try
            If usernameTextBox.Text.Trim = "" Or passwordTextBox.Text.Trim = "" Then 'Validation : Checks to see if characters are in
either TextBox
                MessageBox.Show("You have entered your credentials wrong. Make sure that the Username and Password are at least 5
characters in Length. Please try again", "Submit Credentials", MessageBoxButtons.OK, MessageBoxIcon.Error)
            Else
                Dim inputUsername As String
                Dim inputPassword As String
                inputUsername = usernameTextBox.Text.Trim 'Assiging User Inputs to variables, Trim Function takes out spaces
                inputPassword = passwordTextBox.Text.Trim
                If inputUsername <> dataUsername Or inputPassword <> "abc" Then 'Checks to see if they match
                    MessageBox.Show("Username or Password is incorrect. Please try again", "Submit Credentials", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
                    ClearTextBox()
                    deductAttempt() 'sminus attempt from list
                ElseIf inputUsername = "XXXX" And inputPassword = "abc" Then 'If the passwords match and are correct
                    Select Case "Yes" 'Checks admin privileges, at the moment I've left it as just "Yes"
                        Case "Yes"
                            dataAdmin = True
                        Case "No"
                            dataAdmin = False
                    End Select
                    ClearTextBox()
                    attemptCount = 3 'reset counter
                    attemptsLabel.Visible = False
                    Dim frm As New typerForm
                    frm.Show()
```

```
                    Me.Hide() 'Hides form and does not close as this is the startup form and need variables associated with it
                End If
            End If
        Catch ex As NullReferenceException
            MessageBox.Show("Something went wrong, please try again", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
            ClearTextBox()
        End Try
    End Sub

    Private Sub cancelBtn_Click(sender As Object, e As EventArgs) Handles cancelBtn.Click
        Application.Exit()
    End Sub

    Private Sub createAccBtn_Click(sender As Object, e As EventArgs) Handles createAccBtn.Click
        Dim frm As New addUserForm
        frm.Show()
        Me.Hide()
    End Sub

    Private Sub loginForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        attemptsLabel.Visible = False
    End Sub


    Private Sub guestBtn_Click(sender As Object, e As EventArgs) Handles guestBtn.Click
        dataUsername = "Guest" 'Update variable
        ClearTextBox() 'Clear textbox of inputs so that so sensitive information isn't present
        Dim frm As New typerForm
        frm.Show()
        Me.Hide()
    End Sub

End Class
```

Add User Form

```
'Importing modules needed for future use
Imports System.Data.OleDb

Public Class addUserForm
    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Dim result As DialogResult
        If passTextBox.Text = retryPassTextBox.Text And userTextBox.Text.Trim.Length <> 0 And passTextBox.Text.Trim.Length <> 0 And
nameTextBox.Text.Trim.Length <> 0 Then 'Validation : Checking password entries match and that no textboxes and no characters in them
            result = MessageBox.Show("Are you sure?", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then 'If clicked yes then carry on
                    Dim username, password, myName, admin As String 'Initialise variables to then be passed into SQL
                    username = userTextBox.Text.Trim 'Trim them out of whitespace (avoids errors)
                    password = passTextBox.Text.Trim
                    myName = nameTextBox.Text.Trim
                    admin = "No" 'Default value for new user
                End If
            Catch ex As Exception
                MessageBox.Show("Something has gone wrong. Please try again.") 'For database, if the connection or SQL fails
            End Try
        Else
            MessageBox.Show("You have entered your credentials wrong.  Make sure that the Username and Password are at least 5
characters in length. Please try again", "Submit Credentials") 'Validation
        End If
    End Sub

    Private Sub addUserForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    End Sub
End Class
```

Typing Test Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions
Imports FakeItEasy

Public Class typerForm
    Dim extract As String 'Declaring all the variables
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
```

```
    Friend startType As Boolean
    Dim conn As New OleDbConnection
    Friend startingTimer = "Countdown"
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Function FetchRandomParagraphs() As String 'Fetch paragraphs from textfile
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt") 'Add each line as a
string in an array
        Dim essayExtract As String
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Int32
        randomNumber = Generator.Next(0, lineCount) 'Generate a random number in the range of 1 to the amount of lines in the array
        essayExtract = lines.ElementAt(randomNumber).ToString 'Pass the string at the random index value into the varaibles and return
the variable
        Return essayExtract
    End Function

    Function FetchRandomWords() 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\randomWords.txt") 'initialises on array
on the words on each line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50 'Will add 50 words to the array
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            Return wordList
        End If
    End Function

    Function IntoWords(essay) As String() 'Dividing string into a words array
        Dim wordList As String()
        wordList = essay.Split(" ")
        Return wordList
    End Function

    Function IntoChar(wordList, y) As String() 'For the word that the user is on, it gets split up into an array of letters
        Dim stringInformation As New Globalization.StringInfo(wordList(y))
        Dim characterList(stringInformation.LengthInTextElements - 1) As String
        For i As Integer = 0 To stringInformation.LengthInTextElements - 1
            characterList(i) = stringInformation.SubstringByTextElements(i, 1)
        Next
        Return characterList
    End Function



    Function IntoEssay(wordList) 'With Random Words mode, need to get base extract as string, so converts wordList back to string
        Dim essaySpaces As String
        For Each word In wordList
            essaySpaces += word + " " 'Need to add dspaces as without, all the words will be one long string
        Next
        Dim essay As String = essaySpaces.Remove(essaySpaces.Length - 1, 1) 'From the for loop the string will end on a space which will
cause issues with the typing game so elimnating it solves the problem
        Return essay
    End Function

    Sub TimerPauseMode(duration) 'Starts the timer in pause mode
        startType = False 'Starting type (puts the Countdown/Countup functino in pause mode)
        timeLeft = Convert.ToInt32(duration) 'global variable timeLeft used to be the duration of the pause before test starts akak (3)
        wordTimer.Start() 'Starts timer
    End Sub

    Sub Countdown()
        If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
            timeLeft -= 0.1 'Decrements by 0.1
            timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
        Else
            If startType = True Then 'If currently playing a typing game then make the timer stop and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START" 'Reset Start Button
                wordTimer.Stop()
                MeasureTime() 'Measure
            Else
                timeDoneLabel.Text = "GO" 'If not currently playing and its countdown at the start then reset timer and start countdown
at specifc time duration
                inputTextBox.ReadOnly = False 'Lets user input in textbox
                startType = True 'Changes variable to signfy the test has started
                timeLeft = timeDuration 'Resets duration to actual time limit for test
                wordTimer.Stop() 'Resets timer
                wordTimer.Start()
            End If
        End If
    End Sub

    Sub CountUp()
        If startingTimer = "CountDown" Then 'If we are in the countdown pause mode before type game starts then ...
```

```
            If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
                timeLeft -= 0.1 'Decrements by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'Resets timer to 0 and starts typing game
                timeDoneLabel.Text = "GO"
                timeLeft = 0 'Starting time is 0 as we going from 0 to limit
                inputTextBox.ReadOnly = False 'Lets user input in the box
                wordTimer.Start()
                startingTimer = "CountUp" 'Resets variable to countup mode for the test
            End If
        Else 'When the test has started
            If timeLeft < timeDuration Then 'Checks to see if time limit has been reached, if not then ..
                timeLeft += 0.1 'Increment timer by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
            Else 'When time limit has been reached then stop timer and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START"
                wordTimer.Stop() 'Resets timer
                MeasureTime()
            End If
        End If
    End Sub

    Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished before time limit has
been reached
            timeTakenNow = timeLeft
        Else 'countdown mode when game is finished beofre time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        accuracy = (100 - 100 * (totalFaults / countCharEssay))
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1) 'Add results to labels to show to the user
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)
    End Sub
    Sub TurnColourCharacters(start, lengthEnd, color) 'Function to changes a specific number of character's colours in the richtextbox
e.g when a letter goes green
        essayRichTextBox.SelectionStart = start 'Start of the text
        essayRichTextBox.SelectionLength = lengthEnd 'Length of the text to be selected
        essayRichTextBox.SelectionColor = color 'Forecolour that the selected text is changed to
    End Sub

    Sub TurnColourBackground(colourBack) 'Function to change the colour of the background
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Sub TurnColourBackground(colourFont, colourBack) 'Function to change the colour of the background and forecolor of RichTextbox
        essayRichTextBox.ForeColor = colourFont
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'If whole word
is correct (wordlength = inputlength) and inputTextbox contains word currently being written
                countChar = 0 'Reset counter for the new word
                countCharEssay += 1 'Increment total charcters typed
                countWord += 1 'Increment counter to go to next word
                inputTextBox.Clear() 'Reset inputTextBox (so old word is cleared form textbox)
                If countChar = inputTextBox.TextLength And words.Length = countWord Then  'If all words are completed, then stop test
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Otherwise move onto next word in the array
                    charList = IntoChar(words, countWord)
                End If
            ElseIf (inputTextBox.Text.EndsWith(charList(countChar)) = True) And (countChar + 1 = inputTextBox.TextLength) Then 'checks
last character typed end with the inputted char and the lengths are the same
                TurnColourCharacters(countCharEssay, 1, Color.Green) 'Turn character green
                countChar += 1 'increment counters
                countCharEssay += 1
            ElseIf (inputTextBox.TextLength > countChar) Then 'If inputTextBox length is higher than correct charcters then there's a
mistake been made
                TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                currentFaults += 1 'increment fault counter
            End If
        Catch

        End Try
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        conn.Close()
        loginForm.Show()
```

```
        Me.Close()
    End Sub


    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        conn.Close()
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub startBtn_Click(sender As Object, e As EventArgs) Handles startBtn.Click
        essayRichTextBox.ShowSelectionMargin = True
        Select Case loginForm.extractType
            Case "Random Words"
                words = FetchRandomWords()
                extract = IntoEssay(words)
                essayRichTextBox.Text() = extract
                countChar = 0
                countWord = 0
                charList = IntoChar(words, countWord)
            Case "Paragraphs"
                extract = FetchRandomParagraphs()
                words = IntoWords(extract)
                essayRichTextBox.Text() = extract
                countChar = 0
                countWord = 0
                charList = IntoChar(words, countWord)
            Case Else
                MessageBox.Show("Something has gone wrong", "Error")
        End Select
        If startBtn.Text = "START" Then
            countCharEssay = 0
            currentFaults = 0
            timeLeft = 0
            startingTimer = "CountDown"
            inputTextBox.Clear()
            TurnColourBackground(Color.Gray, Color.LightGray)
            startBtn.Text = "STOP"
            TimerPauseMode(3)
        Else
            timeDoneLabel.Text = "TIME"
            inputTextBox.ReadOnly = True
            inputTextBox.Clear()
            startBtn.Text = "START"
            TurnColourBackground(Color.Gray, Color.FromArgb(70, 72, 73))
            wordTimer.Stop()
        End If
    End Sub

    Private Sub wordTimer_Tick() Handles wordTimer.Tick
        Select Case loginForm.countType
            Case "CountDown"
                Countdown()
            Case "CountUp"
                CountUp()
        End Select
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click

    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        conn.Close()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        conn.Close()
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Hide()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub typerForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

WordFade Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions

Public Class wordFadeForm
    Dim extract As String
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
    Friend startType As Boolean
    Friend startingTimer = "Countdown"
    Dim essayExtract As String
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Private Sub wordFadeForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
    End Sub


    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
    End Sub

    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged

    End Sub

    Private Sub startFadeBtn_Click(sender As Object, e As EventArgs) Handles startFadeBtn.Click

    End Sub

    Private Sub wordTimer_Tick(sender As Object, e As EventArgs) Handles wordTimer.Tick
    End Sub

End Class
```

Statistics Form

```
Imports System.Data.OleDb
Imports System.Windows.Forms.DataVisualization.Charting

Public Class statUserForm
    Dim conn As New OleDbConnection

    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim leaderboardDataset As New DataSet
```

```
    Dim addUser As Boolean
    Dim chartDataset As New DataSet



    Function LeaderboardChange(indexValue)
        dataset.Clear()
        Dim SQLleaderboard As String
        Select Case indexValue
            Case 0
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Max(entryData.WPM) AS [Max Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 1
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Max(entryData.Accuracy) AS [Max Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 2
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Avg(entryData.Accuracy) AS [Avg Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 3
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Avg(entryData.WPM) AS [Avg Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 4
                SQLleaderboard = "SELECT userData.Username, Count(entryData.EntryID) AS [Amount Of Entries] FROM userData INNER JOIN
entryData ON userData.UserID = entryData.UserID GROUP BY userData.Username ORDER BY Count(entryData.EntryID) DESC;"
            Case Else
        End Select

    End Function



    Private Sub statUserForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'TODO: This line of code loads data into the 'UserDatabaseDataSet.entryData' table. You can move, or remove it, as needed.
        loginForm.UpdateLabelMode(Me)
        leaderboardComboBox.SelectedIndex = 0
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If

    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear()
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
        End If
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs) Handles refreshBtn.Click
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub leaderboardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
```

```
leaderboardComboBox.SelectedIndexChanged
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub
End Class
```

Admin Form

```
Imports System.Data.OleDb
Imports System.Globalization
Imports System.IO

Public Class statAdminForm
    Dim conn As New OleDbConnection
    Dim commandSelect As OleDbDataAdapter
    Dim userSearch As String
    Dim totalRows As Int32
    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim addUser As Boolean
    Dim previousUsername



    Private Sub statAdminForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        addUser = False
    End Sub


    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        typerForm.Show()
        Me.Hide()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles searchBtn.Click
    End Sub

    Private Sub nextBtn_Click(sender As Object, e As EventArgs) Handles nextBtn.Click
        currentRow += 1
        Select Case currentRow
            Case totalRows
                currentRow = 0
        End Select
    End Sub

    Private Sub previousBtn_Click(sender As Object, e As EventArgs) Handles previousBtn.Click
        currentRow -= 1
        Select Case currentRow
            Case -1
                currentRow = totalRows - 1
        End Select
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs)
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
```

```
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. If you are deleting your own
account then you'll be logged out", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear()
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click

    End Sub

    Private Sub searchDateBtn_Click(sender As Object, e As EventArgs) Handles searchDateBtn.Click

    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub
End Class
```

Options Form

```
Public Class optionsForm


    Dim timeDurationArray As String() = {"15", "30", "45", "60"} 'String arrays containing the options for the comboboxes
    Dim countTypeArray As String() = {"CountDown", "CountUp"}
    Dim extractTypeArray As String() = {"Paragraphs", "Random Words"}
    Dim wordFadeExtractTypeArray As String() = {"Paragraphs", "Random Words", "Alphabet"}

    Sub AddToComboBox(comboBoxName, itemArray) 'When form loads in, the string array items are added into a specified ComboBox
        comboBoxName.Items.Clear() 'Clears a ComboBox of previous items
        For Each item As String In itemArray
            comboBoxName.Items.Add(item)
        Next
    End Sub

    Sub ChangeComboItem(comboBoxName, itemArray, item) 'Changes the index of the ComboBox depending on prexisting variable values
        comboBoxName.SelectedIndex = Array.IndexOf(itemArray, item)
    End Sub

    Sub ChangeCheckBoxMode(checkbox, mode) 'Changes the tick of the CheckBox depending on prexisting variable values
        checkbox.Checked = mode
    End Sub

    Sub ChangeComboBoxKeyboard(comboBoxName, item) 'ComboBox keyboard options changes variable value
        Select Case item
            Case "ShowHighlight"
                comboBoxName.selectedIndex = 0
            Case "Show"
                comboBoxName.selectedIndex = 1
            Case "Nothing"
                comboBoxName.selectedIndex = 2
        End Select
    End Sub

    Function changeItem(comboBoxName) 'Changes variable value dpedning on option chosen
        Dim variable As String
        variable = comboBoxName.SelectedItem.ToString()
        Return variable
    End Function

    Private Sub optionsForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        AddToComboBox(countComboBox, countTypeArray) 'Adds items to ComboBoxes
        AddToComboBox(timeDurationComboBox, timeDurationArray)
        AddToComboBox(extractComboBox, extractTypeArray)
        AddToComboBox(wordFadeExtractComboBox, wordFadeExtractTypeArray)
        ChangeComboItem(countComboBox, countTypeArray, loginForm.countType) 'Changes selected items based on previous history in the
form
        ChangeComboItem(timeDurationComboBox, timeDurationArray, loginForm.timeDuration)
        ChangeComboItem(extractComboBox, extractTypeArray, loginForm.extractType)
        ChangeComboItem(wordFadeExtractComboBox, wordFadeExtractTypeArray, loginForm.extractWordFadeType)
        ChangeCheckBoxMode(suddenDeathCheckBox, loginForm.suddenDeath)
        ChangeCheckBoxMode(capsCheckBox, loginForm.capitals)
        ChangeCheckBoxMode(punctuationCheckBox, loginForm.punctuation)
        ChangeComboBoxKeyboard(keyBoardComboBox, loginForm.showKeyboard)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
```

```
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub countComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles countComboBox.SelectedIndexChanged
        loginForm.countType = changeItem(countComboBox)
    End Sub

    Private Sub timeDurationComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
timeDurationComboBox.SelectedIndexChanged
        loginForm.timeDuration = changeItem(timeDurationComboBox)
    End Sub

    Private Sub extractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles extractComboBox.SelectedIndexChanged
        loginForm.extractType = changeItem(extractComboBox)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub suddenDeathCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles suddenDeathCheckBox.CheckedChanged
        loginForm.suddenDeath = suddenDeathCheckBox.Checked
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub capsCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles capsCheckBox.CheckedChanged
        loginForm.capitals = capsCheckBox.Checked
    End Sub

    Private Sub punctuationCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles punctuationCheckBox.CheckedChanged
        loginForm.punctuation = punctuationCheckBox.Checked
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub wordFadeExtractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
wordFadeExtractComboBox.SelectedIndexChanged
        loginForm.extractWordFadeType = changeItem(wordFadeExtractComboBox)
    End Sub

    Private Sub keyBoardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles keyBoardComboBox.SelectedIndexChanged
        Select Case keyBoardComboBox.SelectedIndex
            Case 0
                loginForm.showKeyboard = "ShowHighlight"
            Case 1
                loginForm.showKeyboard = "Show"
            Case 2
                loginForm.showKeyboard = "Nothing"
        End Select
    End Sub

    Private Sub changeFontBtn_Click(sender As Object, e As EventArgs) Handles changeFontBtn.Click
        If (changeFontDialog.ShowDialog() = DialogResult.OK) Then
            loginForm.globalFont = changeFontDialog.Font
        End If
    End Sub


    Private Sub linkLabel_LinkClicked(sender As Object, e As LinkLabelLinkClickedEventArgs) Handles linkLabel.LinkClicked
        Dim address = "https://support.microsoft.com/en-us/office/download-and-install-microsoft-365-access-runtime-185c5a32-8ba9-491e-
ac76-91cbe3ea09c9"
        linkLabel.LinkVisited = True
        System.Diagnostics.Process.Start(address)
    End Sub
End Class
```

Version 2

Login Form

```
Imports System.Data.OleDb

Public Class loginForm 'All friend variables are global that will be used across the form

    Friend connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\userDatabase.accdb" 'Connection string used
to connect to the database
    Private conn As New OleDbConnection
    Friend dataUsername As String  'Current user logged in credentials are kept in the variables
    Friend dataName As String
    Friend dataAdmin As Boolean
    Friend Shared dataUserID As String
    Friend timeDuration = "30" 'Option variables to be passed into various forms
    Friend countType = "CountDown"
    Friend suddenDeath = False
    Friend extractType = "Paragraphs"
    Friend extractWordFadeType = "Paragraphs"
    Friend capitals = True
    Friend punctuation = True
    Friend showKeyboard = "ShowHighlight"
    Friend globalFont As New Font("Calibri", 28, Font.Style.Regular)
    Friend entriesPresent = False 'Used to determine whether user can enter the Statistics form
    Friend attemptCount = 3 'Used as a counter for the attempts until applicatoin closes
    Friend red As Color = Color.FromArgb(255, 128, 128) 'Used as colours for keyboard
    Friend lightGreen As Color = Color.FromArgb(128, 255, 128)
    Friend lightBlue As Color = Color.FromArgb(128, 255, 255)
    Friend purple As Color = Color.FromArgb(255, 128, 255)
    Friend yellow As Color = Color.FromArgb(255, 255, 128)
    Friend blue As Color = Color.FromArgb(204, 153, 255)
    Friend orange As Color = Color.FromArgb(255, 192, 128)
    Friend darkGreen As Color = Color.FromArgb(179, 229, 97)

    Sub UpdateLabelMode(formNow) 'Fetches the mode and updates the label, label is present in every page and is referenced there too
        formNow.modeLabel.Text = "MODE : " & extractType
    End Sub


    Sub LogOut() 'When logging out, the variables with data pertaining to the login is cleared
        dataUserID = ""
        dataUsername = ""
        dataName = ""
        dataAdmin = False
    End Sub

    Sub ClearTextBox() 'After each entry or login the textbox is cleared of inputs
        usernameTextBox.Clear()
        passwordTextBox.Clear()
    End Sub

    Sub deductAttempt() 'Deducts an entry from the counter and updates/shows the text label
        attemptCount -= 1
        Select Case attemptCount
            Case 2
                attemptsLabel.Visible = True
                attemptsLabel.Text = "You have 2 attempts left"
            Case 1
                attemptsLabel.Text = "You have 1 attempt left"
            Case 0
                Me.Close() 'After 3 attempts the program closes
        End Select
    End Sub

    Sub GetAmountEntries() 'Checks the amount of entries, with that user to stop any database errors
        conn.Open()
        Dim SQLGetEntries = "SELECT Count(entryData.EntryID) AS [Entries] FROM userData INNER JOIN entryData ON userData.UserID =
entryData.UserID GROUP BY userData.Username HAVING (userData.Username)='" & dataUsername & "';"
        Dim commandGetEntries = New OleDbCommand(SQLGetEntries, conn)
        If commandGetEntries.ExecuteScalar() Is Nothing Or commandGetEntries.ExecuteScalar() = "1" Then
            entriesPresent = False
        Else
            entriesPresent = True
        End If
        conn.Close()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Try
            conn.Open()
            If usernameTextBox.Text.Trim = "" Or passwordTextBox.Text.Trim = "" Or usernameTextBox.TextLength < 5 Or
passwordTextBox.TextLength < 5 Then 'Validation : Checks to see if characters are in either TextBox
                MessageBox.Show("You have entered your credentials wrong. Make sure that the Username and Password are at least 5
characters in Length. Please try again", "Submit Credentials", MessageBoxButtons.OK, MessageBoxIcon.Error)
                conn.Close()
            Else
                Dim inputUsername As String
                Dim inputPassword As String
                Dim dataPassword As String
                inputUsername = usernameTextBox.Text.Trim 'Assiging User Inputs to variables, Trim Function takes out spaces
                inputPassword = passwordTextBox.Text.Trim
                Dim SQLpassword As String = "SELECT Password FROM [userData] WHERE Username= '" & inputUsername & "';" 'Formatting SQL
Code to fetch password where username matches
```

```
                Dim SQLadmin As String = "SELECT Admin FROM userData WHERE Username= '" & inputUsername & "';" 'SQL code to fetch Admin
Privileges
                Dim SQLuserID As String = "SELECT [UserID] FROM [userData] WHERE Username='" & inputUsername & "';"
                Dim commandPassword As New OleDbCommand(SQLpassword, conn)
                Dim commandAdmin As New OleDbCommand(SQLadmin, conn)
                Dim commandUserID As New OleDbCommand(SQLuserID, conn)
                dataPassword = commandPassword.ExecuteScalar()
                If dataPassword Is Nothing Or dataPassword <> inputPassword Then 'Checks if SQL fetch password fetches nothing (due to
wrong username) or does not match
                    MessageBox.Show("Username or Password is incorrect. Please try again", "Submit Credentials", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
                    ClearTextBox()
                    conn.Close()
                    deductAttempt() 'minus attempt from list
                ElseIf dataPassword.ToString = inputPassword Then 'If the passwords match and are correct
                    dataUsername = inputUsername
                    dataUserID = commandUserID.ExecuteScalar()
                    Dim adminCheck = commandAdmin.ExecuteScalar()
                    Select Case adminCheck 'Checks admin privileges
                        Case "Yes"
                            dataAdmin = True
                        Case "No"
                            dataAdmin = False
                    End Select
                    ClearTextBox()
                    conn.Close()
                    attemptCount = 3 'reset counter
                    attemptsLabel.Visible = False
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Hide() 'Hides form and does not close as this is the startup form and need variables associated with it
                End If
            End If
        Catch ex As NullReferenceException
            MessageBox.Show("Something went wrong, please try again", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
            ClearTextBox()
            conn.Close()
        End Try
    End Sub

    Private Sub cancelBtn_Click(sender As Object, e As EventArgs) Handles cancelBtn.Click
        Application.Exit()
    End Sub

    Private Sub createAccBtn_Click(sender As Object, e As EventArgs) Handles createAccBtn.Click
        Dim frm As New addUserForm
        frm.Show()
        Me.Hide()
    End Sub

    Private Sub loginForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        conn.ConnectionString = connectionString
        attemptsLabel.Visible = False
    End Sub


    Private Sub guestBtn_Click(sender As Object, e As EventArgs) Handles guestBtn.Click
        dataUsername = "Guest" 'Update variable
        ClearTextBox() 'Clear textbox of inputs so that so sensitive information isn't present
        Dim frm As New typerForm
        frm.Show()
        Me.Hide()
    End Sub

End Class
```

Add User Form

```
'Importing modules needed for future use
Imports System.Data.OleDb

Public Class addUserForm
    Dim conn As New OleDbConnection

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        conn.Close() 'Close databas connection
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Dim result As DialogResult
        If passTextBox.Text = retryPassTextBox.Text And userTextBox.Text.Trim.Length <> 0 And passTextBox.Text.Trim.Length <> 0 And
nameTextBox.Text.Trim.Length <> 0 Then 'Validation : Checking password entries match and that no textboxes and no characters in them
            result = MessageBox.Show("Are you sure?", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then
                    Dim username, password, myName, admin As String
                    username = userTextBox.Text.Trim
                    password = passTextBox.Text.Trim
                    myName = nameTextBox.Text.Trim '
                    admin = "No"
                    Dim sqlAddUser As String = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username,
@password, @name, @admin);" 'SQL for inserting crednetials for new record
```

```
                    Dim commandInsert As New OleDbCommand(sqlAddUser, conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteNonQuery() 'Execute command to database
                    MessageBox.Show("User has been added")
                    conn.Close() 'Closes database connection
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            Catch ex As Exception 'Exception if database does not connect or something goes wrong
                MessageBox.Show("Something has gone wrong. Please try again.")
            End Try
        Else
            MessageBox.Show("You have entered your credentials wrong.  Make sure that the Username and Password are at least 5
characters in length. Please try again", "Submit Credentials")
        End If
    End Sub

    Private Sub addUserForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        conn.ConnectionString = loginForm.connectionString
        conn.Open() 'Opening connection to database
    End Sub
End Class
```

Typing Test Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions
Imports FakeItEasy

Public Class typerForm
    Dim extract As String 'Declaring all the variables
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
    Friend startType As Boolean
    Dim conn As New OleDbConnection
    Friend startingTimer = "Countdown"
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Function FetchRandomParagraphs() As String 'Fetch paragraphs from textfile
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt") 'Add each line as a
string in an array
        Dim essayExtract As String
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Int32
        randomNumber = Generator.Next(0, lineCount) 'Generate a random number in the range of 1 to the amount of lines in the array
        essayExtract = lines.ElementAt(randomNumber).ToString 'Pass the string at the random index value into the varaibles and return
the variable
        If loginForm.punctuation = False Then 'Checks if need to strip string of punctuation (currently has punctuation added)
            essayExtract = Regex.Replace(essayExtract, "[ ](?=[ ])|[^-_A-Za-z0-9 ]", String.Empty) 'function to strip punctuation
        End If
        If loginForm.capitals = False Then 'Checks if captials letter is False (current has captials added)
            Return essayExtract.ToLower()
        Else
            Return essayExtract
        End If
    End Function

    Function FetchRandomWords() 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\randomWords.txt") 'initialises on array
on the words on each line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50 'Will add 50 words to the array
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            If loginForm.capitals = True Then 'If statement checks whether to add Captials to the words (they are all currently lower
case)
                wordList = WithCaps(wordList)
            End If
            Return wordList
        End If
    End Function
```

```
    Function WithCaps(wordList As String()) 'Adds capitals in random words mode
        Dim Generator As System.Random = New System.Random()
        For i = 0 To wordList.Length - 1
            Dim randomNumber = Generator.Next(0, 2) 'a random number is generated
            wordList(i) = CapsFirstLetter(wordList, i, randomNumber)
        Next
        Return wordList
    End Function

    Function CapsFirstLetter(wordList, i, randomNumber) 'Called by 'WithCaps' to capatlise the first letter of the word
        Dim characterArray() As Char = wordList(i).ToCharArray
        If randomNumber = 0 Then 'Making chances of Captilisation 1/2 chance from the random number
            characterArray(0) = Char.ToUpper(characterArray(0))
        End If
        Return New String(characterArray)
    End Function

    Function IntoWords(essay) As String() 'Dividing string into a words array
        Dim wordList As String()
        wordList = essay.Split(" ")
        Return wordList
    End Function

    Function IntoChar(wordList, y) As String() 'For the word that the user is on, it gets split up into an array of letters
        Dim stringInformation As New Globalization.StringInfo(wordList(y))
        Dim characterList(stringInformation.LengthInTextElements - 1) As String
        For i As Integer = 0 To stringInformation.LengthInTextElements - 1
            characterList(i) = stringInformation.SubstringByTextElements(i, 1)
        Next
        Return characterList
    End Function

    Function IntoEssay(wordList) 'With Random Words mode, need to get base extract as string, so converts wordList back to string
        Dim essaySpaces As String
        For Each word In wordList
            essaySpaces += word + " " 'Need to add dspaces as without, all the words will be one long string
        Next
        Dim essay As String = essaySpaces.Remove(essaySpaces.Length - 1, 1) 'From the for loop the string will end on a space which will
cause issues with the typing game so elimnating it solves the problem
        Return essay
    End Function

    Sub TurnColourCharacters(start, lengthEnd, color) 'Function to changes a specific number of character's colours in the richtextbox
e.g when a letter goes green
        essayRichTextBox.SelectionStart = start 'Start of the text
        essayRichTextBox.SelectionLength = lengthEnd 'Length of the text to be selected
        essayRichTextBox.SelectionColor = color 'Forecolour that the selected text is changed to
    End Sub

    Sub TurnColourBackground(colourBack) 'Function to change the colour of the background
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Sub TurnColourBackground(colourFont, colourBack) 'Function to change the colour of the background and forecolor of RichTextbox
        essayRichTextBox.ForeColor = colourFont
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Sub TimerPauseMode(duration) 'Starts the timer in pause mode
        startType = False 'Starting type (puts the Countdown/Countup functino in pause mode)
        timeLeft = Convert.ToInt32(duration) 'global variable timeLeft used to be the duration of the pause before test starts akak (3)
        wordTimer.Start() 'Starts timer
    End Sub

    Sub Countdown()
        If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
            timeLeft -= 0.1 'Decrements by 0.1
            timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
        Else
            If startType = True Then 'If currently playing a typing game then make the timer stop and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START" 'Reset Start Button
                wordTimer.Stop()
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime() 'Measure
            Else
                timeDoneLabel.Text = "GO" 'If not currently playing and its countdown at the start then reset timer and start countdown
at specifc time duration
                inputTextBox.ReadOnly = False 'Lets user input in textbox
                startType = True 'Changes variable to signfy the test has started
                timeLeft = timeDuration 'Resets duration to actual time limit for test
                wordTimer.Stop() 'Resets timer
                wordTimer.Start()
            End If
        End If
    End Sub

    Sub CountUp()
        If startingTimer = "CountDown" Then 'If we are in the countdown pause mode before type game starts then ...
            If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
                timeLeft -= 0.1 'Decrements by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
```

```
            Else 'Resets timer to 0 and starts typing game
                timeDoneLabel.Text = "GO"
                timeLeft = 0 'Starting time is 0 as we going from 0 to limit
                inputTextBox.ReadOnly = False 'Lets user input in the box
                wordTimer.Start()
                startingTimer = "CountUp" 'Resets variable to countup mode for the test
            End If
        Else 'When the test has started
            If timeLeft < timeDuration Then 'Checks to see if time limit has been reached, if not then ..
                timeLeft += 0.1 'Increment timer by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
            Else 'When time limit has been reached then stop timer and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START"
                wordTimer.Stop() 'Resets timer
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime()
            End If
        End If
    End Sub

    Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished before time limit has
been reached
            timeTakenNow = timeLeft
        Else 'countdown mode when game is finished beofre time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        Try
            accuracy = (100 - 100 * (totalFaults / countCharEssay))
        Catch e As Exception
            accuracy = 0
        End Try
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1) 'Add results to labels to show to the user
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)

        If loginForm.dataUsername <> "Guest" Or countCharEssay = 0 Then 'Adding the entry into the database is not an option for a Guest
user, otherwise if countCharEssay is 0 then do not add in the entry
            Dim SQLinsertentry = "INSERT INTO [entryData] ([UserID], [Date], [WPM], [Accuracy], [Mode], [Time Duration], [Capital
Letters], [Punctuation]) VALUES (@userID, @date, @wpm, @accuracy, @mode, @timeduration, @capitals, @punctuation)"
            Dim commandInsertEntry As New OleDbCommand(SQLinsertentry, conn) 'SQL for adding in the new entry in [entryTable]
            commandInsertEntry.Parameters.AddWithValue("@userID", loginForm.dataUserID)
            commandInsertEntry.Parameters.Add("@date", OleDbType.Date).Value = Date.Today()
            commandInsertEntry.Parameters.AddWithValue("@wpm", Math.Round(wordsPerMinute, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@accuracy", Math.Round(accuracy, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@mode", loginForm.countType.ToString)
            commandInsertEntry.Parameters.AddWithValue("@timeduration", loginForm.timeDuration.ToString)
            Select Case loginForm.capitals 'checks what captials variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "Yes")
                Case Else
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "No")
            End Select
            Select Case loginForm.punctuation 'checks what punctuation variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@punctuation", "Yes")
                Case Else
                    Select Case loginForm.extractType 'Undefined and No depends on whether it Random Words or Essay (Puncutation option
only affects Essay)
                        Case "CountDown"
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "No")
                        Case Else
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "UnDefined")
                    End Select
            End Select
            Try
                commandInsertEntry.ExecuteNonQuery()
            Catch e As Exception
                MessageBox.Show(e.ToString)
            End Try
        End If
    End Sub

    Sub ColourPanel(color, Panel) 'Function to Colour the Panel
        Panel.BackColor = color
    End Sub

    Sub ChangeKeyColour([char]) 'Checks the next key needed to be pressed and highlights key on keyboard
        Dim currentChar As String
        ChangeAllKeyColourGrey() 'Chagnes previous keys too all grey
        If charList.Length > ([char]) Then 'So that there is not a Out of Range Exception with charlist([char])
            currentChar = charList([char])
            Select Case currentChar.ToLower
                Case "a"
                    ColourPanel(loginForm.red, aPanel)
```

```
                Case "b"
                    ColourPanel(loginForm.purple, bPanel)
                Case "c"
                    ColourPanel(loginForm.lightBlue, cPanel)
                Case "d"
                    ColourPanel(loginForm.lightBlue, dPanel)
                Case "e"
                    ColourPanel(loginForm.lightBlue, ePanel)
                Case "f"
                    ColourPanel(loginForm.purple, fPanel)
                Case "g"
                    ColourPanel(loginForm.purple, gPanel)
                Case "h"
                    ColourPanel(loginForm.yellow, hPanel)
                Case "i"
                    ColourPanel(loginForm.blue, iPanel)
                Case "j"
                    ColourPanel(loginForm.yellow, jPanel)
                Case "k"
                    ColourPanel(loginForm.blue, kPanel)
                Case "l"
                    ColourPanel(loginForm.orange, lPanel)
                Case "m"
                    ColourPanel(loginForm.yellow, mPanel)
                Case "n"
                    ColourPanel(loginForm.yellow, nPanel)
                Case "o"
                    ColourPanel(loginForm.orange, oPanel)
                Case "p"
                    ColourPanel(loginForm.darkGreen, pPanel)
                Case "q"
                    ColourPanel(loginForm.red, qPanel)
                Case "r"
                    ColourPanel(loginForm.purple, rPanel)
                Case "s"
                    ColourPanel(loginForm.lightGreen, sPanel)
                Case "t"
                    ColourPanel(loginForm.purple, tPanel)
                Case "u"
                    ColourPanel(loginForm.yellow, uPanel)
                Case "v"
                    ColourPanel(loginForm.purple, vPanel)
                Case "w"
                    ColourPanel(loginForm.lightGreen, wPanel)
                Case "x"
                    ColourPanel(loginForm.lightGreen, xPanel)
                Case "y"
                    ColourPanel(loginForm.yellow, yPanel)
                Case "z"
                    ColourPanel(loginForm.red, zPanel)
                Case Else
            End Select
            If Char.IsUpper(charList([char])) = True Then 'Checks if letter is a capital
                ColourPanel(loginForm.red, shiftLeftPanel)
                ColourPanel(loginForm.darkGreen, shiftRightPanel)
            End If
        ElseIf [char] = charList.Length Then 'When end of a word press space
            ColourPanel(Color.Silver, spacePanel)
        End If
    End Sub

    Sub ChangeAllKeyColourGrey() 'Resets entire keyboard colour to 'Gray'
        ColourPanel(Color.Gainsboro, aPanel)
        ColourPanel(Color.Gainsboro, bPanel)
        ColourPanel(Color.Gainsboro, cPanel)
        ColourPanel(Color.Gainsboro, dPanel)
        ColourPanel(Color.Gainsboro, ePanel)
        ColourPanel(Color.Gainsboro, fPanel)
        ColourPanel(Color.Gainsboro, gPanel)
        ColourPanel(Color.Gainsboro, hPanel)
        ColourPanel(Color.Gainsboro, iPanel)
        ColourPanel(Color.Gainsboro, jPanel)
        ColourPanel(Color.Gainsboro, kPanel)
        ColourPanel(Color.Gainsboro, lPanel)
        ColourPanel(Color.Gainsboro, mPanel)
        ColourPanel(Color.Gainsboro, nPanel)
        ColourPanel(Color.Gainsboro, oPanel)
        ColourPanel(Color.Gainsboro, pPanel)
        ColourPanel(Color.Gainsboro, qPanel)
        ColourPanel(Color.Gainsboro, rPanel)
        ColourPanel(Color.Gainsboro, sPanel)
        ColourPanel(Color.Gainsboro, tPanel)
        ColourPanel(Color.Gainsboro, uPanel)
        ColourPanel(Color.Gainsboro, vPanel)
        ColourPanel(Color.Gainsboro, wPanel)
        ColourPanel(Color.Gainsboro, xPanel)
        ColourPanel(Color.Gainsboro, yPanel)
        ColourPanel(Color.Gainsboro, spacePanel)
        ColourPanel(Color.Gainsboro, zPanel)
        ColourPanel(Color.Gainsboro, shiftLeftPanel)
        ColourPanel(Color.Gainsboro, shiftRightPanel)
    End Sub

    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'Checks if full
word is correct and the user has pressed spaces to go onto next word
                countChar = 0
```

```
                countCharEssay += 1
                countWord += 1
                essayRichTextBox.SelectionStart = countCharEssay + 1
                inputTextBox.Clear() 'Resets input box
                If countChar = inputTextBox.TextLength And words.Length = countWord Then 'Checks if full essay has been completed
                    timeDoneLabel.Text = "STOP"
                    startBtn.Text = "START"
                    inputTextBox.ReadOnly = True
                    inputTextBox.Clear()
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Goes onto next word
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                End If
            ElseIf (countChar + 1 = inputTextBox.TextLength) And (countChar <> charList.Length) And currentFaults = 0 Then 'Checks if it
is next inputted letter compared to input, and there are no currentFaults (in case the user backspaces)
                If (inputTextBox.Text.EndsWith(charList(countChar)) = True) Then 'This is put here to avoid an OutOfRangeException,
checks after the range conditioms are met
                    TurnColourCharacters(countCharEssay, 1, Color.Green)
                    countChar += 1
                    countCharEssay += 1
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Else 'Checks 1st incorrect character
                    currentFaults += 1
                    totalFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    essayPanel.BackColor = Color.IndianRed
                    essayRichTextBox.BackColor = Color.IndianRed
                    If loginForm.suddenDeath = True Then
                        timeDoneLabel.Text = "STOP"
                        startBtn.Text = "START"
                        inputTextBox.ReadOnly = True
                        inputTextBox.Clear()
                        startType = False
                        wordTimer.Stop()
                        ChangeAllKeyColourGrey()
                    End If
                End If
            Else
                If (inputTextBox.TextLength > (countChar)) Then 'Checks if any inputted letter are incorrect, checks the 2nd+ incorrect
character
                    totalFaults += 1
                    currentFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    If essayPanel.BackColor <> Color.IndianRed Then
                        essayPanel.BackColor = Color.IndianRed
                        essayRichTextBox.BackColor = Color.IndianRed
                    End If
                End If
                If inputTextBox.TextLength < (countChar + currentFaults + 1) Then 'Checks if a letter has been backspaced
                    differenceOfFaults = (countChar + currentFaults) - inputTextBox.TextLength 'Used difference instead of decrement,
mutliple charcters can be dedcuted at once (highlighting)
                    currentFaults -= differenceOfFaults
                    TurnColourCharacters(countCharEssay + currentFaults, differenceOfFaults, Color.Black)
                    If inputTextBox.TextLength < (countChar) And currentFaults <= 0 Then 'Checks if a green letter has been backspaced,
countChar would e more than the input
                        currentFaults = 0 'Do not want it to become -1
                        countChar -= (countChar - inputTextBox.TextLength) 'subtracts from counts using the different between values
                        countCharEssay -= (countChar - inputTextBox.TextLength + 1)
                        If loginForm.showKeyboard = "ShowHighlight" Then
                            ChangeKeyColour(countChar) 'changes key highlighted to previous key
                        End If
                    End If
                    If currentFaults = 0 Then
                        essayPanel.BackColor = Color.Gainsboro
                        essayRichTextBox.BackColor = Color.Gainsboro
                    End If
                End If
            End If
        Catch
        End Try
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        conn.Close()
        loginForm.Show()
        Me.Close()
    End Sub


    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        conn.Close()
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub startBtn_Click(sender As Object, e As EventArgs) Handles startBtn.Click
        essayRichTextBox.ShowSelectionMargin = True
        Select Case loginForm.extractType
            Case "Random Words"
                words = FetchRandomWords()
```

```
                extract = IntoEssay(words)
                essayRichTextBox.Text() = extract
                countChar = 0
                countWord = 0
                charList = IntoChar(words, countWord)
            Case "Paragraphs"
                extract = FetchRandomParagraphs()
                words = IntoWords(extract)
                essayRichTextBox.Text() = extract
                countChar = 0
                countWord = 0
                charList = IntoChar(words, countWord)
            Case Else
                MessageBox.Show("Something has gone wrong", "Error")
        End Select
        If startBtn.Text = "START" Then
            countCharEssay = 0
            currentFaults = 0
            timeLeft = 0
            startingTimer = "CountDown"
            inputTextBox.Clear()
            startBtn.Text = "STOP"
            TimerPauseMode(3)
        Else
            timeDoneLabel.Text = "TIME"
            inputTextBox.ReadOnly = True
            inputTextBox.Clear()
            startBtn.Text = "START"
            If loginForm.showKeyboard = "ShowHighlight" Then
                ChangeAllKeyColourGrey()
            End If
            wordTimer.Stop()
        End If
    End Sub

    Private Sub wordTimer_Tick() Handles wordTimer.Tick
        Select Case loginForm.countType
            Case "CountDown"
                Countdown()
            Case "CountUp"
                CountUp()
        End Select
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        conn.Close()
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        conn.Close()
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Hide()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub typerForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        conn.ConnectionString = loginForm.connectionString
        If loginForm.showKeyboard = "Nothing" Then
            keyBoardPanel.Visible = False 'hiding the keyboard from veiw
        End If
        essayRichTextBox.Font = loginForm.globalFont 'Setting the font
        conn.Open()
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

WordFade Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions

Public Class wordFadeForm
    Dim extract As String
```

```
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
    Friend startType As Boolean
    Friend startingTimer = "Countdown"
    Dim essayExtract As String
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Private Sub wordFadeForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
    End Sub


    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
    End Sub

    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged

    End Sub

    Private Sub startFadeBtn_Click(sender As Object, e As EventArgs) Handles startFadeBtn.Click

    End Sub

    Private Sub wordTimer_Tick(sender As Object, e As EventArgs) Handles wordTimer.Tick
    End Sub

End Class
```

Statistics Form

```
Imports System.Data.OleDb
Imports System.Windows.Forms.DataVisualization.Charting

Public Class statUserForm

    Dim conn As New OleDbConnection 'Declaring the variables
    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim leaderboardDataset As New DataSet
    Dim addUser As Boolean
    Dim chartDataset As New DataSet
    Dim previousUsername As String 'Stores the last username the user is loaded onto

    Function GetRowNumber(idNumber) 'To find the rowNumber (record number) of the user in the table
        Dim SQLGetUserRow = "SELECT * FROM userData" 'Fetches whole table
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUserRow, conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        Dim rowNum As Integer
```

```
        rowNum = 0
        For Each row As DataRow In dataset.Tables("userData").Rows
            If row.Item("UserID") = Convert.ToInt32(idNumber) Then 'If the UserID in record matches UserID then return Row
                Exit For
            End If
            rowNum += 1
        Next row
        Return rowNum
    End Function

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes
        conn.Open()
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(0).Item(0) 'Row 0 as its only 1 record fetched,
        usernameTextBox.Text = dataset.Tables("userData").Rows(0).Item(1) 'Item 'number' to signify the column value
        passwordTextBox.Text = dataset.Tables("userData").Rows(0).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(0).Item(3)
        If dataset.Tables("userData").Rows(0).Item(4) = "Yes" Then 'Changes checkbox value depending on value in record
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        conn.Close()
        previousUsername = usernameTextBox.Text.Trim
    End Sub

    Sub Max(columnName, labelName) 'Fetches record by Maximum value of either (WPM or Accuracy' for a specific user
        Dim SQLGetMax = "SELECT MAX (" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQLGetMax, conn)
        conn.Open()
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString 'Rounds variable shown to 1 dp
        conn.Close()
    End Sub

    Sub Average(columnName, labelName) 'Fetches record by Average value of either (WPM or Accuracy' for a specific user
        Dim SQlGetAverage = "SELECT AVG(" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQlGetAverage, conn)
        commandGet.Parameters.AddWithValue("@column", columnName)
        conn.Open()
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString 'Rounds variable shown to 1 dp
        conn.Close()
    End Sub

    Sub LeaderboardChange(indexValue) 'Depending on option selected in ComboBox, the SQL relating to leaderboardin DataGridViewer is
changed
        dataset.Clear()
        Dim SQLleaderboard As String
        Select Case indexValue
            Case 0
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.WPM) AS [Max Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 1
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.Accuracy) AS [Max Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 2
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.Accuracy) AS [Avg Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 3
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.WPM) AS [Avg Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case Else
        End Select
        Dim commandLeaderboard As New OleDbDataAdapter(SQLleaderboard, conn) 'SQL is excuted
        conn.Open()
        commandLeaderboard.Fill(leaderboardDataset, "Leaderboard")
        conn.Close()
        dataGridView.DataSource = leaderboardDataset.Tables("Leaderboard") 'Data is filled into the DataGridViewer
        leaderboardDataset.Tables.Remove("LeaderBoard") 'Reset and clear dataset
    End Sub


    Sub ChartWPMLoad()
        Dim SQLgetWPM = "SELECT DISTINCTROW Avg(entryData.WPM) AS WPM, entryData.Date FROM userData INNER JOIN entryData ON
userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLgetWPM, conn)
        conn.Open()
        commandGetWPM.Fill(chartDataset, "WPMTable")
        'Add a new series
        Dim SeriesName As New Series
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userWPMChart.DataSource = chartDataset.Tables("WPMTable")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "WPM"
        userWPMChart.Series.Add(SeriesName)
        conn.Close()
    End Sub

    Sub ChartAccuracyLoad()
        Dim SQLGetAccuray = "SELECT DISTINCTROW Avg(entryData.Accuracy) AS Accuracy, entryData.Date FROM userData INNER JOIN entryData
ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLGetAccuray, conn)
        conn.Open()
```

```
        commandGetWPM.Fill(chartDataset, "SQLGetAccuray")
        'Add a new series
        Dim SeriesName As New Series
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userAccuracyChart.DataSource = chartDataset.Tables("SQLGetAccuray")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "Accuracy"
        userAccuracyChart.Series.Add(SeriesName)
        conn.Close()
    End Sub

    Private Sub statUserForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'TODO: This line of code loads data into the 'UserDatabaseDataSet.entryData' table. You can move, or remove it, as needed.
        loginForm.UpdateLabelMode(Me)
        conn.ConnectionString = loginForm.connectionString
        loginForm.UpdateLabelMode(Me)
        currentRow = GetRowNumber(loginForm.dataUserID)
        GetUserDetails(loginForm.dataUsername)
        leaderboardComboBox.SelectedIndex = 0
        Dim SQLuser = "SELECT * FROM userData"
        Dim SQLentry = "SELECT * FROM entryData"
        conn.Open()
        Dim commandUser = New OleDbDataAdapter(SQLuser, conn)
        Dim commandEntry = New OleDbDataAdapter(SQLentry, conn)
        commandUser.Fill(leaderboardDataset, "userData")
        commandUser.Fill(leaderboardDataset, "entryData")
        leaderboardDataset.Tables("userData").PrimaryKey = {leaderboardDataset.Tables("userData").Columns("UserID")}
        leaderboardDataset.Tables("entryData").PrimaryKey = {leaderboardDataset.Tables("userData").Columns("EntryID")}
        Dim userDataRelation As DataRelation = New DataRelation("Leaderboard", leaderboardDataset.Tables("userData").Columns("UserID"),
leaderboardDataset.Tables("entryData").Columns("UserID"))
        leaderboardDataset.Relations.Add(userDataRelation)
        conn.Close()
        Try
            Max("WPM", highestWPMLabel)
            Max("Accuracy", highestAccuracyLabel)
            Average("WPM", averageWPMLabel)
            Average("Accuracy", averageAccuracyLabel)
        Catch ex As Exception
            MessageBox.Show(ex.ToString)
        End Try
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If

    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        If addUserBtn.Text = "&ADD USER" Then 'If is not in Add User mode
            Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. You'll be logged out",
"Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then
                    conn.Open()
                    Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text
                    Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text
                    Dim commandDelete As New OleDbCommand(SQLDeleteUser, conn)
                    Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, conn)
                    commandDeleteData.ExecuteReader() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                    commandDelete.ExecuteScalar() 'Then delete user
                    conn.Close()
                    MessageBox.Show("User has been successfully deleted", "Successful Deletion")
                Else
                End If
            Catch ex As Exception
                MessageBox.Show("Something has gone wrong", "Error")
            End Try
        Else
            MessageBox.Show("You are in Add User Mode, you cannot delete an account you have not created yet", "Error")
        End If
```

```
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear() 'Ease of use for clearing all inputs from textboxes
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails(loginForm.dataUsername)
            LeaderboardChange(leaderboardComboBox.SelectedIndex)
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
inputs have at least 5 characters for username and password
            MessageBox.Show("You have entered your credentials wrong. Please try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';"
            Dim commandCheck As New OleDbCommand(SQLCheckUser, conn)
            conn.Open()
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            conn.Close()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                conn.Open()
                Dim username, password, myName, admin As String
                username = usernameTextBox.Text.Trim
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
                End Select
                Dim sqlAddUser As String
                If addUserBtn.Text = "&CANCEL ADD USER" Then
                    sqlAddUser = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);"
                Else
                    sqlAddUser = "UPDATE [userData] SET [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" &
myName & "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";"
                End If
                Dim commandInsert As New OleDbCommand(sqlAddUser, conn)
                commandInsert.Parameters.AddWithValue("@username", username)
                commandInsert.Parameters.AddWithValue("@password", password)
                commandInsert.Parameters.AddWithValue("@name", myName)
                commandInsert.Parameters.AddWithValue("@admin", admin)
                commandInsert.ExecuteNonQuery()
                MessageBox.Show("User credentials have been modified", "Modification Successful")
                conn.Close()
            End If
        End If
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs) Handles refreshBtn.Click
        GetUserDetails(loginForm.dataUsername)
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub leaderboardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
leaderboardComboBox.SelectedIndexChanged
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub
End Class
```

Admin Form

```
Imports System.Data.OleDb
Imports System.Globalization
Imports System.IO
```

```
Public Class statAdminForm
    Dim conn As New OleDbConnection
    Dim commandSelect As OleDbDataAdapter
    Dim totalRows As Int32
    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim addUser As Boolean
    Dim previousUsername

    Sub LoadDataGridUser() 'Load current users entries and output onto DataGridViewer
        conn.ConnectionString = loginForm.connectionString
        conn.Open()
        If dataGridView.RowCount > 0 Then
            dataGridView.Rows.Clear()
        End If
        Dim SQLSelectData = "SELECT * FROM [entryData] WHERE UserID = " + userIDTextBox.Text 'Fetch entry records for the current user
        commandSelect = New OleDbDataAdapter(SQLSelectData, conn)
        dataset.Clear()
        commandSelect.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Fill DataGridViewer with the [EntryData} table (table 1)
        conn.Close()
    End Sub

    Sub LoadDateDataGridUser(startDate, endDate) 'Same as the previous, however there are parameters for the dates
        conn.ConnectionString = loginForm.connectionString
        conn.Open()
        Dim SQLSelectDateData = "SELECT * FROM [entryData] WHERE UserID =" & userIDTextBox.Text & " AND Date BETWEEN @start AND @end"
        Dim commandSelect As New OleDbCommand
        commandSelect.Connection = conn
        commandSelect.CommandText = SQLSelectDateData
        commandSelect.Parameters.Add("@start", startDatePicker.Value)
        commandSelect.Parameters.Add("@start", endDatePicker.Value)
        dataset.Clear()
        Dim adapter As New OleDbDataAdapter(commandSelect)
        adapter.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Filling DataGridView with data from the dataset
        conn.Close()
    End Sub

    Sub GetUserDetails() 'Fetch user details and inputs into the textboxes and checkboxes
        conn.Open()
        Dim SQLGetUser = "SELECT * FROM [userData]"
        commandSelect = New OleDbDataAdapter(SQLGetUser, conn)
        dataset.Clear()
        commandSelect.Fill(dataset, "userData") 'NOTE that the Row is dependent on currentRow variable (used to speifcy a specific user)
        userIDTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(0) 'Row 0 as its  only 1 record fetched
        usernameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(1) 'Item 'number' to signify the column value
        passwordTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(3)
        If dataset.Tables("userData").Rows(currentRow).Item(4) = "Yes" Then  'Changes checkbox value depending on value in record
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        conn.Close()
        previousUsername = usernameTextBox.Text.Trim
    End Sub

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes when a specific username has been
specified
        conn.Open() 'SAME LOGIC
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(0)
        usernameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(1)
        passwordTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(3)
        If dataset.Tables("userData").Rows(currentRow).Item(4) = "Yes" Then
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        conn.Close()
        previousUsername = usernameTextBox.Text.Trim
    End Sub


    Private Sub statAdminForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        conn.ConnectionString = loginForm.connectionString
        currentRow = 0 'Resets varibles
        GetUserDetails() 'Loads in first user's record in textboxes
        LoadDataGridUser() 'Loads user entries to DataGridViewer
        totalRows = dataset.Tables(0).Rows.Count 'Fetches total number of
        addUser = False
    End Sub


    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        typerForm.Show()
        Me.Hide()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
```

```
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles searchBtn.Click
        Dim userSearch As String
        userSearch = userSearchTextBox.Text 'Takes the value from user input as a variable
        conn.Open()
        Dim SQLSelectData = "SELECT username from [userData] WHERE username = '" & userSearch & "';"
        Dim commandCheckUser As New OleDbCommand(SQLSelectData, conn)
        conn.Close()
        currentRow = 0 'CurrentRow needs to equal 0, as the Logic applied would have only one record be shown (only one record for one
user with that username). After when 'Previous or 'Next' btn is pressed, resets naivgation to start from 0
        GetUserDetails(userSearch) 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub nextBtn_Click(sender As Object, e As EventArgs) Handles nextBtn.Click 'totalRow variable is declared and values changed
with Line 87 (fetches the number of records (accounts)
        currentRow += 1
        Select Case currentRow
            Case totalRows 'reached the end of the table, go back to beginning
                currentRow = 0
        End Select
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub previousBtn_Click(sender As Object, e As EventArgs) Handles previousBtn.Click
        currentRow -= 1
        Select Case currentRow
            Case -1  'Reached the start of the table, go to the end
                currentRow = totalRows - 1
        End Select
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs)
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then 'If currently in Add User Mode, come out of it
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails()
            totalRows = dataset.Tables(0).Rows.Count
            LoadDataGridUser()
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. If you are deleting your own
account then you'll be logged out", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
        Try
            If result = DialogResult.Yes Then
                conn.Open()
                Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim commandDelete As New OleDbCommand(SQLDeleteUser, conn)
                Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, conn)
                commandDeleteData.ExecuteScalar() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                commandDelete.ExecuteScalar() 'Then delete user
                conn.Close()
                currentRow = 0 'Resets naivagation of user accounts
                GetUserDetails()
                totalRows = dataset.Tables(0).Rows.Count 'Fetches new total accounts number
                LoadDataGridUser()
                MessageBox.Show("User has been successfully deleted", "Successful Deletion")
            End If
        Catch ex As Exception
```

```
            MessageBox.Show("Something has gone wrong", "Error")
        End Try
        conn.Close()
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear() 'Ease of use for clearing all inputs from textboxes
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
        Dim SQL As String
        Dim SQLUsername As String
        Dim SQLAdmin As String
        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
if there are no inputs in the textbox
            MessageBox.Show("You have entered your credentials wrong. Please try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';" 'SQL to
check if username is already present
            Dim commandCheck As New OleDbCommand(SQLCheckUser, conn)
            conn.Open()
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            conn.Close()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database, and whether is has been modified
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                Dim username, password, myName, admin As String 'Declares and assigns input values to variables
                username = usernameTextBox.Text.Trim 'Trim to avoid whitespace being present in database and when comparing for the
login process
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                conn.Open()
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
                End Select
                If addUserBtn.Text = "&CANCEL ADD USER" Then 'If the Add User button was pressed, then Insert account to database
                    SQL = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);"

                    Dim commandInsert As New OleDbCommand(SQL, conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteScalar()
                Else 'If not, update the currently logged on account
                    SQL = "UPDATE [userData] SET [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" & myName
& "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";"
                    Dim commandInsert As New OleDbCommand(SQL, conn)
                    commandInsert.ExecuteScalar()
                End If
                MessageBox.Show("Action completed successfully", "Confirmation completed")
                conn.Close()
                currentRow = 0
                addUser = False
                addUserBtn.Text = "&ADD USER"
                GetUserDetails()
                totalRows = dataset.Tables(0).Rows.Count
                LoadDataGridUser()
            End If
        End If
    End Sub

    Private Sub searchDateBtn_Click(sender As Object, e As EventArgs) Handles searchDateBtn.Click
        LoadDateDataGridUser(startDatePicker.Value.ToShortDateString, endDatePicker.Value.ToShortDateString)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub
End Class
```

Options Form

```
Public Class optionsForm


    Dim timeDurationArray As String() = {"15", "30", "45", "60"} 'String arrays containing the options for the comboboxes
    Dim countTypeArray As String() = {"CountDown", "CountUp"}
    Dim extractTypeArray As String() = {"Paragraphs", "Random Words"}
    Dim wordFadeExtractTypeArray As String() = {"Paragraphs", "Random Words", "Alphabet"}

    Sub AddToComboBox(comboBoxName, itemArray) 'When form loads in, the string array items are added into a specified ComboBox
        comboBoxName.Items.Clear() 'Clears a ComboBox of previous items
        For Each item As String In itemArray
            comboBoxName.Items.Add(item)
```

```
        Next
    End Sub

    Sub ChangeComboItem(comboBoxName, itemArray, item) 'Changes the index of the ComboBox depending on prexisting variable values
        comboBoxName.SelectedIndex = Array.IndexOf(itemArray, item)
    End Sub

    Sub ChangeCheckBoxMode(checkbox, mode) 'Changes the tick of the CheckBox depending on prexisting variable values
        checkbox.Checked = mode
    End Sub

    Sub ChangeComboBoxKeyboard(comboBoxName, item) 'ComboBox keyboard options changes variable value
        Select Case item
            Case "ShowHighlight"
                comboBoxName.selectedIndex = 0
            Case "Show"
                comboBoxName.selectedIndex = 1
            Case "Nothing"
                comboBoxName.selectedIndex = 2
        End Select
    End Sub

    Function changeItem(comboBoxName) 'Changes variable value dpedning on option chosen
        Dim variable As String
        variable = comboBoxName.SelectedItem.ToString()
        Return variable
    End Function

    Private Sub optionsForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        AddToComboBox(countComboBox, countTypeArray) 'Adds items to ComboBoxes
        AddToComboBox(timeDurationComboBox, timeDurationArray)
        AddToComboBox(extractComboBox, extractTypeArray)
        AddToComboBox(wordFadeExtractComboBox, wordFadeExtractTypeArray)
        ChangeComboItem(countComboBox, countTypeArray, loginForm.countType) 'Changes selected items based on previous history in the
form
        ChangeComboItem(timeDurationComboBox, timeDurationArray, loginForm.timeDuration)
        ChangeComboItem(extractComboBox, extractTypeArray, loginForm.extractType)
        ChangeComboItem(wordFadeExtractComboBox, wordFadeExtractTypeArray, loginForm.extractWordFadeType)
        ChangeCheckBoxMode(suddenDeathCheckBox, loginForm.suddenDeath)
        ChangeCheckBoxMode(capsCheckBox, loginForm.capitals)
        ChangeCheckBoxMode(punctuationCheckBox, loginForm.punctuation)
        ChangeComboBoxKeyboard(keyBoardComboBox, loginForm.showKeyboard)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub countComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles countComboBox.SelectedIndexChanged
        loginForm.countType = changeItem(countComboBox)
    End Sub

    Private Sub timeDurationComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
timeDurationComboBox.SelectedIndexChanged
        loginForm.timeDuration = changeItem(timeDurationComboBox)
    End Sub

    Private Sub extractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles extractComboBox.SelectedIndexChanged
        loginForm.extractType = changeItem(extractComboBox)
        loginForm.UpdateLabelMode(Me)
    End Sub
```

```
    Private Sub suddenDeathCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles suddenDeathCheckBox.CheckedChanged
        loginForm.suddenDeath = suddenDeathCheckBox.Checked
    End Sub

    Private Sub capsCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles capsCheckBox.CheckedChanged
        loginForm.capitals = capsCheckBox.Checked
    End Sub

    Private Sub punctuationCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles punctuationCheckBox.CheckedChanged
        loginForm.punctuation = punctuationCheckBox.Checked
    End Sub


    Private Sub wordFadeExtractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
wordFadeExtractComboBox.SelectedIndexChanged
        loginForm.extractWordFadeType = changeItem(wordFadeExtractComboBox)
    End Sub

    Private Sub keyBoardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles keyBoardComboBox.SelectedIndexChanged
        Select Case keyBoardComboBox.SelectedIndex
            Case 0
                loginForm.showKeyboard = "ShowHighlight"
            Case 1
                loginForm.showKeyboard = "Show"
            Case 2
                loginForm.showKeyboard = "Nothing"
        End Select
    End Sub

    Private Sub changeFontBtn_Click(sender As Object, e As EventArgs) Handles changeFontBtn.Click
        If (changeFontDialog.ShowDialog() = DialogResult.OK) Then
            loginForm.globalFont = changeFontDialog.Font
        End If
    End Sub


    Private Sub linkLabel_LinkClicked(sender As Object, e As LinkLabelLinkClickedEventArgs) Handles linkLabel.LinkClicked
        Dim address = "https://support.microsoft.com/en-us/office/download-and-install-microsoft-365-access-runtime-185c5a32-8ba9-491e-
ac76-91cbe3ea09c9"
        linkLabel.LinkVisited = True
        System.Diagnostics.Process.Start(address)
    End Sub
End Class
```

Version 3

Login Form

```
Imports System.Data.OleDb

Public Class loginForm  'All friend variables are global that will be used across the form

    Friend connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\userDatabase.accdb" ' Connection string
used to connect to the database
    Public conn As New OleDbConnection 'Main connection used throughtout the soltuion
    Friend dataUsername As String
    Friend dataName As String
    Friend dataAdmin As Boolean
    Friend Shared dataUserID As String
    Friend timeDuration = "30" 'Option variables to be passed into various forms
    Friend countType = "CountDown"
    Friend suddenDeath = False
    Friend extractType = "Paragraphs"
    Friend extractWordFadeType = "Paragraphs"
    Friend capitals = True
    Friend punctuation = True
    Friend showKeyboard = "ShowHighlight"
    Friend entriesPresent = False 'Used to determine whether user can enter the Statistics form
    Friend globalFont As New Font("Calibri", 28, Font.Style.Regular)
    Friend attemptCount = 3 'Used as a counter for the attempts until applicatio nn closes
    Friend red As Color = Color.FromArgb(255, 128, 128) 'Used as colours for keyboard
    Friend lightGreen As Color = Color.FromArgb(128, 255, 128)
    Friend lightBlue As Color = Color.FromArgb(128, 255, 255)
    Friend purple As Color = Color.FromArgb(255, 128, 255)
    Friend yellow As Color = Color.FromArgb(255, 255, 128)
    Friend blue As Color = Color.FromArgb(204, 153, 255)
    Friend orange As Color = Color.FromArgb(255, 192, 128)
    Friend darkGreen As Color = Color.FromArgb(179, 229, 97)

    Sub CheckStartDatabase()
        Try 'In a Try so it catches any exceptoins if database connection cannot be met
            conn.Open()
        Catch ex As Exception
            MessageBox.Show("The Database connection cannot be initialised, and login and add user features are not available. You will
now be in Guest Mode and taken straight to a Typing Test", "Database Connection Error")
            dataUsername = "Guest"
            submitBtn.Enabled = False
            createAccBtn.Enabled = False
        End Try
    End Sub

    Sub UpdateLabelMode(formNow) 'Fetches the mode and updates the label, label is present in every page and is referenced there too
        formNow.modeLabel.Text = "MODE : " & extractType
    End Sub

    Sub ClearTextBox() 'After each entry or login the textbox is cleared of inputs
        usernameTextBox.Clear()
        passwordTextBox.Clear()
    End Sub

    Sub LogOut() 'When logging out, the variables with data pertaining to the login is cleared
        dataUserID = ""
        dataUsername = ""
        dataName = ""
        dataAdmin = False
    End Sub

    Sub GetAmountEntries() 'Checks the amount of entries, with that user to stop any database errors
        Dim SQLGetEntries = "SELECT Count(entryData.EntryID) AS [Entries] FROM userData INNER JOIN entryData ON userData.UserID =
entryData.UserID GROUP BY userData.Username HAVING (userData.Username)='" & dataUsername & "';"
        Dim commandGetEntries = New OleDbCommand(SQLGetEntries, conn)
        If commandGetEntries.ExecuteScalar() Is Nothing Or commandGetEntries.ExecuteScalar() = "1" Then
            entriesPresent = False
        Else
            entriesPresent = True
        End If
    End Sub

    Sub deductAttempt() 'Deducts an entry from the counter and updates/shows the text label
        attemptCount -= 1
        Select Case attemptCount
            Case 2
                attemptsLabel.Visible = True
                attemptsLabel.Text = "You have 2 attempts left"
            Case 1
                attemptsLabel.Text = "You have 1 attempt left"
            Case 0
                Me.Close() 'After 3 attempts the program closes
        End Select
    End Sub

    Private Sub cancelBtn_Click(sender As Object, e As EventArgs) Handles cancelBtn.Click
        Application.Exit()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Try
            If usernameTextBox.Text.Trim = "" Or passwordTextBox.Text.Trim = "" Or usernameTextBox.TextLength < 5 Or
```

```
passwordTextBox.TextLength < 5 Then 'Validation : Checks to see if characters are in either TextBox
                MessageBox.Show("You have entered your credentials wrong. Make sure that the Username and Password are at least 5
characters in Length. Please try again", "Submit Credentials", MessageBoxButtons.OK, MessageBoxIcon.Error)
            Else
                Dim inputUsername As String
                Dim inputPassword As String
                Dim dataPassword As String
                inputUsername = usernameTextBox.Text.Trim 'Assiging User Inputs to variables, Trim Function takes out spaces
                inputPassword = passwordTextBox.Text.Trim
                Dim SQLpassword As String = "SELECT Password FROM [userData] WHERE Username= '" & inputUsername & "';" 'Formatting SQL
Code to fetch password where username matches
                Dim SQLadmin As String = "SELECT Admin FROM userData WHERE Username= '" & inputUsername & "';" 'SQL code to fetch Admin
Privileges
                Dim SQLuserID As String = "SELECT [UserID] FROM [userData] WHERE Username='" & inputUsername & "';"
                Dim commandPassword As New OleDbCommand(SQLpassword, conn)
                Dim commandAdmin As New OleDbCommand(SQLadmin, conn)
                Dim commandUserID As New OleDbCommand(SQLuserID, conn)
                dataPassword = commandPassword.ExecuteScalar()
                If dataPassword Is Nothing Or dataPassword <> inputPassword Then 'Checks if SQL fetch password fetches nothing (due to
wrong username) or does not match
                    MessageBox.Show("Username or Password is incorrect. Please try again", "Submit Credentials", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
                    ClearTextBox()
                    deductAttempt() 'minus attempt from list
                ElseIf dataPassword.ToString = inputPassword Then 'If the passwords match and are correct
                    dataUsername = inputUsername
                    dataUserID = commandUserID.ExecuteScalar()
                    Dim adminCheck = commandAdmin.ExecuteScalar()
                    Select Case adminCheck 'Checks admin privileges
                        Case "Yes"
                            dataAdmin = True
                        Case "No"
                            dataAdmin = False
                    End Select
                    ClearTextBox()
                    attemptCount = 3 'reset counter
                    attemptsLabel.Visible = False
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Hide() 'Hides form and does not close as this is the startup form and need variables associated with it
                End If
            End If
        Catch ex As NullReferenceException
            MessageBox.Show("Something went wrong, please try again", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
            ClearTextBox()
        End Try
    End Sub

    Private Sub createAccBtn_Click(sender As Object, e As EventArgs) Handles createAccBtn.Click
        Dim frm As New addUserForm
        frm.Show()
        Me.Hide()
    End Sub

    Private Sub loginForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        conn.ConnectionString = connectionString
        attemptsLabel.Visible = False
        CheckStartDatabase() 'Checks and opens the connection to the database
    End Sub

    Private Sub guestBtn_Click(sender As Object, e As EventArgs) Handles guestBtn.Click
        dataUsername = "Guest"
        ClearTextBox()
        Me.Hide()
        Dim frm As New typerForm
        frm.Show()
    End Sub

End Class
```

Add User Form

```
Imports System.Data.OleDb

Public Class addUserForm

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Dim result As DialogResult
        If passTextBox.Text = retryPassTextBox.Text And passTextBox.Text.Trim.Length >= 5 And userTextBox.Text.Trim.Length >= 5 And
nameTextBox.Text.Trim.Length <> 0 Then 'Validation : Checking password entries match and that no textboxes and no characters in them
            result = MessageBox.Show("Are you sure?", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            If result = DialogResult.Yes Then
                Dim username, password, myName, admin As String
                username = userTextBox.Text.Trim
                password = passTextBox.Text.Trim
                myName = nameTextBox.Text.Trim '
                admin = "No"
```

```
                Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & username & "';"
                Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
                Dim checkUserPresent = commandCheck.ExecuteScalar()
                If checkUserPresent = username Then 'Checks if username is already present in database
                    MessageBox.Show("A user with this username is already present, please use a different username")
                Else
                    Dim sqlAddUser As String = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username,
@password, @name, @admin);"
                    Dim commandInsert As New OleDbCommand(sqlAddUser, loginForm.conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteNonQuery() 'Execute command to database
                    MessageBox.Show("User has been added")
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            End If
        Else
            MessageBox.Show("You have entered your credentials wrong.  Make sure that the Username and Password are at least 5
characters in length. Please try again", "Submit Credentials")
        End If
    End Sub
End Class
```

Typing Test Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions
Imports FakeItEasy

Public Class typerForm
    Dim extract As String 'Declaring all the variables
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
    Friend startType As Boolean
    Friend startingTimer = "Countdown"
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Function FetchRandomParagraphs() As String 'Fetch paragraphs from textfile
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt") 'Add each line as a
string in an array
        Dim essayExtract As String
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Int32
        randomNumber = Generator.Next(0, lineCount) 'Generate a random number in the range of 1 to the amount of lines in the array
        essayExtract = lines.ElementAt(randomNumber).ToString 'Pass the string at the random index value into the varaibles and return
the variable
        If loginForm.punctuation = False Then 'Checks if need to strip string of punctuation (currently has punctuation added)
            essayExtract = Regex.Replace(essayExtract, "[ ](?=[ ])|[^-_A-Za-z0-9 ]", String.Empty) 'function to strip punctuation
        End If
        If loginForm.capitals = False Then 'Checks if captials letter is False (current has captials added)
            Return essayExtract.ToLower()
        Else
            Return essayExtract
        End If
    End Function

    Function FetchRandomWords() 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\randomWords.txt") 'initialises on array
on the words on each line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50 'Will add 50 words to the array
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            If loginForm.capitals = True Then 'If statement checks whether to add Captials to the words (they are all currently lower
case)
                wordList = WithCaps(wordList)
            End If
            Return wordList
        End If
    End Function

    Function WithCaps(wordList As String()) 'Adds capitals in random words mode
```

```
        Dim Generator As System.Random = New System.Random()
        For i = 0 To wordList.Length - 1
            Dim randomNumber = Generator.Next(0, 2) 'a random number is generated
            wordList(i) = CapsFirstLetter(wordList, i, randomNumber)
        Next
        Return wordList
    End Function

    Function CapsFirstLetter(wordList, i, randomNumber) 'Called by 'WithCaps' to capatlise the first letter of the word
        Dim characterArray() As Char = wordList(i).ToCharArray
        If randomNumber = 0 Then 'Making chances of Captilisation 1/2 chance from the random number
            characterArray(0) = Char.ToUpper(characterArray(0))
        End If
        Return New String(characterArray)
    End Function

    Function IntoWords(essay) As String() 'Dividing string into a words array
        Dim wordList As String()
        wordList = essay.Split(" ")
        Return wordList
    End Function

    Function IntoChar(wordList, y) As String() 'For the word that the user is on, it gets split up into an array of letters
        Dim stringInformation As New Globalization.StringInfo(wordList(y))
        Dim characterList(stringInformation.LengthInTextElements - 1) As String
        For i As Integer = 0 To stringInformation.LengthInTextElements - 1
            characterList(i) = stringInformation.SubstringByTextElements(i, 1)
        Next
        Return characterList
    End Function

    Function IntoEssay(wordList) 'With Random Words mode, need to get base extract as string, so converts wordList back to string
        Dim essaySpaces As String
        For Each word In wordList
            essaySpaces += word + " " 'Need to add dspaces as without, all the words will be one long string
        Next
        Dim essay As String = essaySpaces.Remove(essaySpaces.Length - 1, 1) 'From the for loop the string will end on a space which will
cause issues with the typing game so elimnating it solves the problem
        Return essay
    End Function

    Sub TurnColourCharacters(start, lengthEnd, color) 'Function to changes a specific number of character's colours in the richtextbox
e.g when a letter goes green
        essayRichTextBox.SelectionStart = start 'Start of the text
        essayRichTextBox.SelectionLength = lengthEnd 'Length of the text to be selected
        essayRichTextBox.SelectionColor = color 'Forecolour that the selected text is changed to
    End Sub

    Sub TurnColourBackground(colourBack) 'Function to change the colour of the background
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Sub TurnColourBackground(colourFont, colourBack) 'Function to change the colour of the background and forecolor of RichTextbox
        essayRichTextBox.ForeColor = colourFont
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Sub TimerPauseMode(duration) 'Starts the timer in pause mode
        startType = False 'Starting type (puts the Countdown/Countup functino in pause mode)
        timeLeft = Convert.ToInt32(duration) 'global variable timeLeft used to be the duration of the pause before test starts akak (3)
        wordTimer.Start() 'Starts timer
    End Sub

    Sub Countdown()
        If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
            timeLeft -= 0.1 'Decrements by 0.1
            timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
        Else
            If startType = True Then 'If currently playing a typing game then make the timer stop and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START" 'Reset Start Button
                wordTimer.Stop()
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime() 'Measure
            Else
                timeDoneLabel.Text = "GO" 'If not currently playing and its countdown at the start then reset timer and start countdown
at specifc time duration
                inputTextBox.ReadOnly = False 'Lets user input in textbox
                startType = True 'Changes variable to signfy the test has started
                timeLeft = timeDuration 'Resets duration to actual time limit for test
                wordTimer.Stop() 'Resets timer
                wordTimer.Start()
            End If
        End If
    End Sub

    Sub CountUp()
        If startingTimer = "CountDown" Then 'If we are in the countdown pause mode before type game starts then ...
            If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
                timeLeft -= 0.1 'Decrements by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'Resets timer to 0 and starts typing game
```

```
                timeDoneLabel.Text = "GO"
                timeLeft = 0 'Starting time is 0 as we going from 0 to limit
                inputTextBox.ReadOnly = False 'Lets user input in the box
                wordTimer.Start()
                startingTimer = "CountUp" 'Resets variable to countup mode for the test
            End If
        Else 'When the test has started
            If timeLeft < timeDuration Then 'Checks to see if time limit has been reached, if not then ..
                timeLeft += 0.1 'Increment timer by 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s" 'Updates label
            Else 'When time limit has been reached then stop timer and measure
                startType = False
                inputTextBox.Clear() 'Resets inputTextBox
                inputTextBox.ReadOnly = True 'Stops users editing in it
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START"
                wordTimer.Stop() 'Resets timer
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime()
            End If
        End If
    End Sub

    Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished before time limit has
been reached
            timeTakenNow = timeLeft
        Else 'countdown mode when game is finished beofre time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        Try
            accuracy = (100 - 100 * (totalFaults / countCharEssay))
        Catch e As Exception
            accuracy = 0
        End Try
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1) 'Add results to labels to show to the user
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)
        If loginForm.dataUsername <> "Guest" Or countCharEssay = 0 Then 'Adding the entry into the database is not an option for a Guest
user, otherwise if countCharEssay is 0 then do not add in the entry
            Dim SQLinsertentry = "INSERT INTO [entryData] ([UserID], [Date], [WPM], [Accuracy], [Mode], [Time Duration], [Capital
Letters], [Punctuation]) VALUES (@userID, @date, @wpm, @accuracy, @mode, @timeduration, @capitals, @punctuation)"
            Dim commandInsertEntry As New OleDbCommand(SQLinsertentry, loginForm.conn) 'SQL for adding in the new entry in [entryTable]
            commandInsertEntry.Parameters.AddWithValue("@userID", loginForm.dataUserID)
            commandInsertEntry.Parameters.Add("@date", OleDbType.Date).Value = Date.Today()
            commandInsertEntry.Parameters.AddWithValue("@wpm", Math.Round(wordsPerMinute, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@accuracy", Math.Round(accuracy, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@mode", loginForm.countType.ToString)
            commandInsertEntry.Parameters.AddWithValue("@timeduration", loginForm.timeDuration.ToString)
            Select Case loginForm.capitals 'checks what captials variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "Yes")
                Case Else
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "No")
            End Select
            Select Case loginForm.punctuation 'checks what punctuation variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@punctuation", "Yes")
                Case Else
                    Select Case loginForm.extractType 'Undefined and No depends on whether it Random Words or Essay (Puncutation option
only affects Essay)
                        Case "CountDown"
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "No")
                        Case Else
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "UnDefined")
                    End Select
            End Select
            Try
                commandInsertEntry.ExecuteNonQuery()
            Catch e As Exception
                MessageBox.Show(e.ToString)
            End Try
        End If
    End Sub

    Sub ColourPanel(color, Panel) 'Function to Colour the Panel
        Panel.BackColor = color
    End Sub

    Sub ChangeKeyColour([char]) 'Checks the next key needed to be pressed and highlights key on keyboard
        Dim currentChar As String
        ChangeAllKeyColourGrey() 'Chagnes previous keys too all grey
        If charList.Length > ([char]) Then 'So that there is not a Out of Range Exception with charlist([char])
            currentChar = charList([char])
            Select Case currentChar.ToLower
                Case "a"
                    ColourPanel(loginForm.red, aPanel)
                Case "b"
                    ColourPanel(loginForm.purple, bPanel)
```

```
                Case "c"
                    ColourPanel(loginForm.lightBlue, cPanel)
                Case "d"
                    ColourPanel(loginForm.lightBlue, dPanel)
                Case "e"
                    ColourPanel(loginForm.lightBlue, ePanel)
                Case "f"
                    ColourPanel(loginForm.purple, fPanel)
                Case "g"
                    ColourPanel(loginForm.purple, gPanel)
                Case "h"
                    ColourPanel(loginForm.yellow, hPanel)
                Case "i"
                    ColourPanel(loginForm.blue, iPanel)
                Case "j"
                    ColourPanel(loginForm.yellow, jPanel)
                Case "k"
                    ColourPanel(loginForm.blue, kPanel)
                Case "l"
                    ColourPanel(loginForm.orange, lPanel)
                Case "m"
                    ColourPanel(loginForm.yellow, mPanel)
                Case "n"
                    ColourPanel(loginForm.yellow, nPanel)
                Case "o"
                    ColourPanel(loginForm.orange, oPanel)
                Case "p"
                    ColourPanel(loginForm.darkGreen, pPanel)
                Case "q"
                    ColourPanel(loginForm.red, qPanel)
                Case "r"
                    ColourPanel(loginForm.purple, rPanel)
                Case "s"
                    ColourPanel(loginForm.lightGreen, sPanel)
                Case "t"
                    ColourPanel(loginForm.purple, tPanel)
                Case "u"
                    ColourPanel(loginForm.yellow, uPanel)
                Case "v"
                    ColourPanel(loginForm.purple, vPanel)
                Case "w"
                    ColourPanel(loginForm.lightGreen, wPanel)
                Case "x"
                    ColourPanel(loginForm.lightGreen, xPanel)
                Case "y"
                    ColourPanel(loginForm.yellow, yPanel)
                Case "z"
                    ColourPanel(loginForm.red, zPanel)
                Case Else
            End Select
            If Char.IsUpper(charList([char])) = True Then 'Checks if letter is a capital
                ColourPanel(loginForm.red, shiftLeftPanel)
                ColourPanel(loginForm.darkGreen, shiftRightPanel)
            End If
        ElseIf [char] = charList.Length Then 'When end of a word press space
            ColourPanel(Color.Silver, spacePanel)
        End If
    End Sub

    Sub ChangeAllKeyColourGrey() 'Resets entire keyboard colour to 'Gray'
        ColourPanel(Color.Gainsboro, aPanel)
        ColourPanel(Color.Gainsboro, bPanel)
        ColourPanel(Color.Gainsboro, cPanel)
        ColourPanel(Color.Gainsboro, dPanel)
        ColourPanel(Color.Gainsboro, ePanel)
        ColourPanel(Color.Gainsboro, fPanel)
        ColourPanel(Color.Gainsboro, gPanel)
        ColourPanel(Color.Gainsboro, hPanel)
        ColourPanel(Color.Gainsboro, iPanel)
        ColourPanel(Color.Gainsboro, jPanel)
        ColourPanel(Color.Gainsboro, kPanel)
        ColourPanel(Color.Gainsboro, lPanel)
        ColourPanel(Color.Gainsboro, mPanel)
        ColourPanel(Color.Gainsboro, nPanel)
        ColourPanel(Color.Gainsboro, oPanel)
        ColourPanel(Color.Gainsboro, pPanel)
        ColourPanel(Color.Gainsboro, qPanel)
        ColourPanel(Color.Gainsboro, rPanel)
        ColourPanel(Color.Gainsboro, sPanel)
        ColourPanel(Color.Gainsboro, tPanel)
        ColourPanel(Color.Gainsboro, uPanel)
        ColourPanel(Color.Gainsboro, vPanel)
        ColourPanel(Color.Gainsboro, wPanel)
        ColourPanel(Color.Gainsboro, xPanel)
        ColourPanel(Color.Gainsboro, yPanel)
        ColourPanel(Color.Gainsboro, spacePanel)
        ColourPanel(Color.Gainsboro, zPanel)
        ColourPanel(Color.Gainsboro, shiftLeftPanel)
        ColourPanel(Color.Gainsboro, shiftRightPanel)
    End Sub

    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'Checks if full
word is correct and the user has pressed spaces to go onto next word
                countChar = 0
                countCharEssay += 1
                countWord += 1
```

```
                essayRichTextBox.SelectionStart = countCharEssay + 1
                inputTextBox.Clear() 'Resets input box
                If countChar = inputTextBox.TextLength And words.Length = countWord Then 'Checks if full essay has been completed
                    timeDoneLabel.Text = "STOP"
                    startBtn.Text = "START"
                    inputTextBox.ReadOnly = True
                    inputTextBox.Clear()
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Goes onto next word
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                End If
            ElseIf (countChar + 1 = inputTextBox.TextLength) And (countChar <> charList.Length) And currentFaults = 0 Then 'Checks if it
is next inputted letter compared to input, and there are no currentFaults (in case the user backspaces)
                If (inputTextBox.Text.EndsWith(charList(countChar)) = True) Then 'This is put here to avoid an OutOfRangeException,
checks after the range conditioms are met
                    TurnColourCharacters(countCharEssay, 1, Color.Green)
                    countChar += 1
                    countCharEssay += 1
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Else 'Checks 1st incorrect character
                    currentFaults += 1
                    totalFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    essayPanel.BackColor = Color.IndianRed
                    essayRichTextBox.BackColor = Color.IndianRed
                    If loginForm.suddenDeath = True Then
                        timeDoneLabel.Text = "STOP"
                        startBtn.Text = "START"
                        inputTextBox.ReadOnly = True
                        inputTextBox.Clear()
                        startType = False
                        wordTimer.Stop()
                        ChangeAllKeyColourGrey()
                    End If
                End If
            Else
                If (inputTextBox.TextLength > (countChar)) Then 'Checks if any inputted letter are incorrect, checks the 2nd+ incorrect
character
                    totalFaults += 1
                    currentFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    If essayPanel.BackColor <> Color.IndianRed Then
                        essayPanel.BackColor = Color.IndianRed
                        essayRichTextBox.BackColor = Color.IndianRed
                    End If
                End If
                If inputTextBox.TextLength < (countChar + currentFaults + 1) Then 'Checks if a letter has been backspaced
                    differenceOfFaults = (countChar + currentFaults) - inputTextBox.TextLength 'Used difference instead of decrement,
mutliple charcters can be dedcuted at once (highlighting)
                    currentFaults -= differenceOfFaults
                    TurnColourCharacters(countCharEssay + currentFaults, differenceOfFaults, Color.Black)
                    If inputTextBox.TextLength < (countChar) And currentFaults <= 0 Then 'Checks if a green letter has been backspaced,
countChar would e more than the input
                        currentFaults = 0 'Do not want it to become -1
                        countChar -= (countChar - inputTextBox.TextLength) 'subtracts from counts using the different between values
                        countCharEssay -= (countChar - inputTextBox.TextLength + 1)
                        If loginForm.showKeyboard = "ShowHighlight" Then
                            ChangeKeyColour(countChar) 'changes key highlighted to previous key
                        End If
                    End If
                    If currentFaults = 0 Then
                        essayPanel.BackColor = Color.Gainsboro
                        essayRichTextBox.BackColor = Color.Gainsboro
                    End If
                End If
            End If
        Catch
        End Try
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub


    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub startBtn_Click(sender As Object, e As EventArgs) Handles startBtn.Click
        essayRichTextBox.ShowSelectionMargin = True
        Select Case loginForm.extractType
            Case "Random Words"
                words = FetchRandomWords()
                extract = IntoEssay(words)
                essayRichTextBox.Text() = extract
                countChar = 0
                countWord = 0
```

```
                charList = IntoChar(words, countWord)
            Case "Paragraphs"
                extract = FetchRandomParagraphs()
                words = IntoWords(extract)
                essayRichTextBox.Text() = extract
                countChar = 0
                countWord = 0
                charList = IntoChar(words, countWord)
            Case Else
                MessageBox.Show("Something has gone wrong", "Error")
        End Select
        If startBtn.Text = "START" Then
            countCharEssay = 0
            currentFaults = 0
            timeLeft = 0
            startingTimer = "CountDown"
            inputTextBox.Clear()
            startBtn.Text = "STOP"
            TimerPauseMode(3)
        Else
            timeDoneLabel.Text = "TIME"
            inputTextBox.ReadOnly = True
            inputTextBox.Clear()
            startBtn.Text = "START"
            If loginForm.showKeyboard = "ShowHighlight" Then
                ChangeAllKeyColourGrey()
            End If
            wordTimer.Stop()
        End If
    End Sub

    Private Sub wordTimer_Tick() Handles wordTimer.Tick
        Select Case loginForm.countType
            Case "CountDown"
                Countdown()
            Case "CountUp"
                CountUp()
        End Select
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Hide()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub typerForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        If loginForm.showKeyboard = "Nothing" Then
            keyBoardPanel.Visible = False 'hiding the keyboard from veiw
        End If
        essayRichTextBox.Font = loginForm.globalFont 'Setting the font
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

WordFade Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions

Public Class wordFadeForm
    Dim extract As String
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
```

```
    Friend startType As Boolean
    Friend startingTimer = "Countdown"
    Dim essayExtract As String
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Function FetchRandomParagraphs() As String
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt")
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Int32
        randomNumber = Generator.Next(0, lineCount)
        essayExtract = lines.ElementAt(randomNumber).ToString
        If loginForm.punctuation = False Then
            essayExtract = Regex.Replace(essayExtract, "[ ](?=[ ])|[^-_A-Za-z0-9 ]", String.Empty)
        End If
        If loginForm.capitals = False Then 'if capital? is off then make the extract
            Return essayExtract.ToLower()
        Else
            Return essayExtract
        End If
    End Function

    Function IntoWords(essay) As String() 'Dividing string into a words array
        Dim wordList As String()
        wordList = essay.Split(" ")
        Return wordList
    End Function

    Function FetchRandomWords(filePath) 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & filePath) 'initialises on array on the words on each
line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList As String()
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            If loginForm.capitals = True Then
                wordList = WithCaps(wordList)
            End If
            Return wordList
        End If
    End Function

    Sub RefreshWordOrder(count)
        Select Case words.Length - count - 1 'Checks how close the wordCount is to the end of the array (to avoid OutOfRange
exceptions), then the correct RichTextBox's are cleared of text
            Case 1 '2 words away from end
                thirdWordRichTextBox.Clear()
                secondWordRichTextBox.Text = words(count + 1)
                firstWordRichTextBox.Text = words(count)
            Case 0 '1 word away from the end
                thirdWordRichTextBox.Clear()
                secondWordRichTextBox.Clear()
                firstWordRichTextBox.Text = words(count)
            Case -1 'When the procedure is called for the end of test
            Case Else '>2 words away
                thirdWordRichTextBox.Text = words(count + 2)
                secondWordRichTextBox.Text = words(count + 1)
                firstWordRichTextBox.Text = words(count)
        End Select
    End Sub



    Function WithCaps(wordList As String()) 'Adds capitals in random words mode
        Dim Generator As System.Random = New System.Random()
        For i = 0 To wordList.Length - 1
            Dim randomNumber = Generator.Next(0, 2) 'a random number is generated
            wordList(i) = CapsFirstLetter(wordList, i, randomNumber)
        Next
        Return wordList
    End Function

    Function CapsFirstLetter(wordList, i, randomNumber)
        Dim characterArray() As Char = wordList(i).ToCharArray
        If randomNumber = 0 Then 'Making chances of Captilisation 1/2 chance from the random number
            characterArray(0) = Char.ToUpper(characterArray(0))
        End If
        Return New String(characterArray)
    End Function

    Function IntoChar(wordList, y) As String() 'For the word that the user is on, it gets split up into an array of letters
        Dim stringInformation As New Globalization.StringInfo(wordList(y))
        Dim characterList(stringInformation.LengthInTextElements - 1) As String
        For i As Integer = 0 To stringInformation.LengthInTextElements - 1
            characterList(i) = stringInformation.SubstringByTextElements(i, 1)
        Next
        Return characterList
    End Function
```

```
    Sub TurnColour(start, lengthEnd, color)
        firstWordRichTextBox.SelectionStart = start
        firstWordRichTextBox.SelectionLength = lengthEnd
        firstWordRichTextBox.SelectionColor = color
    End Sub

    Sub TurnColourTextBox(textbox, start, lengthEnd, color)
        textbox.SelectionStart = start
        textbox.SelectionLength = lengthEnd
        textbox.SelectionColor = color
    End Sub

    Sub TurnColourBackgroundWordFade(colourFont, colourBack)
        firstWordRichTextBox.ForeColor = colourFont
        secondWordRichTextBox.ForeColor = colourFont
        thirdWordRichTextBox.ForeColor = colourFont
        essayPanel.BackColor = colourBack
        firstWordRichTextBox.BackColor = colourBack
        secondWordRichTextBox.BackColor = colourBack
        thirdWordRichTextBox.BackColor = colourBack
    End Sub

    Sub PauseTimer(duration)
        timeLeft = Convert.ToInt32(duration)
        wordTimer.Start()
    End Sub

    Sub Countdown()
        If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
            timeLeft -= 0.1
            timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
        Else
            If startType = True Then 'If currently playing a typing game then make the timer stop and measure
                timeDoneLabel.Text = "STOP"
                inputTextBox.ReadOnly = True
                inputTextBox.Clear()
                startType = False
                startFadeBtn.Text = "START"
                wordTimer.Stop()
                MeasureTime()
            Else
                timeDoneLabel.Text = "GO" 'If not currently playing and its countdown at the start then reset timer and start countdown
at specifc time duration
                inputTextBox.ReadOnly = False
                startType = True
                timeLeft = timeDuration
                wordTimer.Stop()
                wordTimer.Start()
            End If
        End If
    End Sub

    Sub CountUp()
        If startingTimer = "CountDown" Then 'If we are in the countdown before type game stats then ..
            If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
                timeLeft -= 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'Resets timer to 0 and starts typing game
                timeLeft = 0
                startingTimer = "CountUp"
                inputTextBox.ReadOnly = False
            End If
        Else
            If timeLeft < timeDuration Then 'Checks to see if time limit has been reached
                timeLeft += 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'When time limit has been reached then stop timer and measure
                timeDoneLabel.Text = "STOP"
                startFadeBtn.Text = "START"
                inputTextBox.ReadOnly = True
                inputTextBox.Clear()
                startType = False
                wordTimer.Stop()
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime()
            End If
        End If
    End Sub

    Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished beofre time limit has
been reached
            timeTakenNow = timeLeft
        Else 'countdown mode when game is finished beofre time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        accuracy = (100 - 100 * (totalFaults / countCharEssay))
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1)
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
```

```
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)
        If loginForm.dataUsername <> "Guest" And loginForm.extractWordFadeType <> "Alphabet" Then 'Adding the entry into the database is
not an option for a Guest user or for the extract type Alphabet
            Dim SQLinsertentry = "INSERT INTO [entryData] ([UserID], [Date], [WPM], [Accuracy], [Mode], [Time Duration], [Capital
Letters], [Punctuation]) VALUES (@userID, @date, @wpm, @accuracy, @mode, @timeduration, @capitals, @punctuation)"
            Dim commandInsertEntry As New OleDbCommand(SQLinsertentry, loginForm.conn) 'SQL for adding in the new entry in [entryTable]
            commandInsertEntry.Parameters.AddWithValue("@userID", loginForm.dataUserID)
            commandInsertEntry.Parameters.Add("@date", OleDbType.Date).Value = Date.Today()
            commandInsertEntry.Parameters.AddWithValue("@wpm", Math.Round(wordsPerMinute, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@accuracy", accuracy.ToString)
            commandInsertEntry.Parameters.AddWithValue("@mode", loginForm.countType.ToString)
            commandInsertEntry.Parameters.AddWithValue("@timeduration", loginForm.timeDuration.ToString)
            Select Case loginForm.capitals 'checks what captials variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "Yes")
                Case Else
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "No")
            End Select
            Select Case loginForm.punctuation 'checks what punctuation variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@punctuation", "Yes")
                Case Else
                    Select Case loginForm.extractType 'Undefined and No depends on whether it Random Words or Essay (Puncutation option
only affects Essay)
                        Case "CountDown"
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "No")
                        Case Else
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "UnDefined")
                    End Select
            End Select
            Try
                commandInsertEntry.ExecuteNonQuery()

            Catch e As Exception
                MessageBox.Show(e.ToString)
            End Try
        End If
    End Sub

    Sub ColourPanel(color, Panel) 'Function to Colour the Panel
        Panel.BackColor = color
    End Sub

    Sub ChangeKeyColour([char]) 'Checks the next key needed to be pressed and highlights key on keyboard
        Dim currentChar As String
        ChangeAllKeyColourGrey() 'Chagnes previous keys too all grey
        If charList.Length > ([char]) Then 'So that there is not a Out of Range Exception with charlist([char])
            currentChar = charList([char])
            Select Case currentChar.ToLower
                Case "a"
                    ColourPanel(loginForm.red, aPanel)
                Case "b"
                    ColourPanel(loginForm.purple, bPanel)
                Case "c"
                    ColourPanel(loginForm.lightBlue, cPanel)
                Case "d"
                    ColourPanel(loginForm.lightBlue, dPanel)
                Case "e"
                    ColourPanel(loginForm.lightBlue, ePanel)
                Case "f"
                    ColourPanel(loginForm.purple, fPanel)
                Case "g"
                    ColourPanel(loginForm.purple, gPanel)
                Case "h"
                    ColourPanel(loginForm.yellow, hPanel)
                Case "i"
                    ColourPanel(loginForm.blue, iPanel)
                Case "j"
                    ColourPanel(loginForm.yellow, jPanel)
                Case "k"
                    ColourPanel(loginForm.blue, kPanel)
                Case "l"
                    ColourPanel(loginForm.orange, lPanel)
                Case "m"
                    ColourPanel(loginForm.yellow, mPanel)
                Case "n"
                    ColourPanel(loginForm.yellow, nPanel)
                Case "o"
                    ColourPanel(loginForm.orange, oPanel)
                Case "p"
                    ColourPanel(loginForm.darkGreen, pPanel)
                Case "q"
                    ColourPanel(loginForm.red, qPanel)
                Case "r"
                    ColourPanel(loginForm.purple, rPanel)
                Case "s"
                    ColourPanel(loginForm.lightGreen, sPanel)
                Case "t"
                    ColourPanel(loginForm.purple, tPanel)
                Case "u"
                    ColourPanel(loginForm.yellow, uPanel)
                Case "v"
                    ColourPanel(loginForm.purple, vPanel)
                Case "w"
                    ColourPanel(loginForm.lightGreen, wPanel)
                Case "x"
                    ColourPanel(loginForm.lightGreen, xPanel)
                Case "y"
                    ColourPanel(loginForm.yellow, yPanel)
```

```
                Case "z"
                    ColourPanel(loginForm.red, zPanel)
                Case Else
            End Select
            If Char.IsUpper(charList([char])) = True Then 'Checks if letter is a capital
                ColourPanel(loginForm.red, shiftLeftPanel)
                ColourPanel(loginForm.darkGreen, shiftRightPanel)
            End If
        ElseIf [char] = charList.Length Then 'When end of a word press space
            ColourPanel(Color.Silver, spacePanel)
        End If
    End Sub

    Sub ChangeAllKeyColourGrey() 'Colours all keys back to grey
        ColourPanel(Color.Gainsboro, aPanel)
        ColourPanel(Color.Gainsboro, bPanel)
        ColourPanel(Color.Gainsboro, cPanel)
        ColourPanel(Color.Gainsboro, dPanel)
        ColourPanel(Color.Gainsboro, ePanel)
        ColourPanel(Color.Gainsboro, fPanel)
        ColourPanel(Color.Gainsboro, gPanel)
        ColourPanel(Color.Gainsboro, hPanel)
        ColourPanel(Color.Gainsboro, iPanel)
        ColourPanel(Color.Gainsboro, jPanel)
        ColourPanel(Color.Gainsboro, kPanel)
        ColourPanel(Color.Gainsboro, lPanel)
        ColourPanel(Color.Gainsboro, mPanel)
        ColourPanel(Color.Gainsboro, nPanel)
        ColourPanel(Color.Gainsboro, oPanel)
        ColourPanel(Color.Gainsboro, pPanel)
        ColourPanel(Color.Gainsboro, qPanel)
        ColourPanel(Color.Gainsboro, rPanel)
        ColourPanel(Color.Gainsboro, sPanel)
        ColourPanel(Color.Gainsboro, tPanel)
        ColourPanel(Color.Gainsboro, uPanel)
        ColourPanel(Color.Gainsboro, vPanel)
        ColourPanel(Color.Gainsboro, wPanel)
        ColourPanel(Color.Gainsboro, xPanel)
        ColourPanel(Color.Gainsboro, yPanel)
        ColourPanel(Color.Gainsboro, spacePanel)
        ColourPanel(Color.Gainsboro, zPanel)
        ColourPanel(Color.Gainsboro, shiftLeftPanel)
        ColourPanel(Color.Gainsboro, shiftRightPanel)
    End Sub


    Private Sub wordFadeForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        firstWordRichTextBox.Font = New Font(loginForm.globalFont.Name, 56, loginForm.globalFont.Style)
        secondWordRichTextBox.Font = New Font(loginForm.globalFont.Name, 28, loginForm.globalFont.Style)
        thirdWordRichTextBox.Font = New Font(loginForm.globalFont.Name, 16, loginForm.globalFont.Style)
        loginForm.UpdateLabelMode(Me)
        If loginForm.showKeyboard = "Nothing" Then
            keyBoardPanel.Visible = False 'hiding the keybaord from veiw
        End If
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
    End Sub
```

```
    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'Checks if full
word is correct and the user has pressed spaces to go onto next word
                countChar = 0
                countCharEssay += 1
                countWord += 1
                RefreshWordOrder(countWord)
                inputTextBox.Clear() 'Resets input box
                If countChar = inputTextBox.TextLength And words.Length = countWord Then 'Checks if full essay has been completed
                    timeDoneLabel.Text = "STOP"
                    startFadeBtn.Text = "START"
                    inputTextBox.ReadOnly = True
                    inputTextBox.Clear()
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Goes onto next word
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                End If
            ElseIf (countChar + 1 = inputTextBox.TextLength) And (countChar <> charList.Length) And currentFaults = 0 Then
                If (inputTextBox.Text.EndsWith(charList(countChar)) = True) And (countChar + 1 = inputTextBox.TextLength) Then 'Checks
if inputted letter is correct
                    TurnColour(countChar, 1, Color.Green)
                    countChar += 1
                    countCharEssay += 1
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Else 'Checks 1st incorrect character
                    currentFaults += 1
                    totalFaults += 1
                    firstWordRichTextBox.BackColor = Color.IndianRed
                    secondWordRichTextBox.BackColor = Color.IndianRed
                    thirdWordRichTextBox.BackColor = Color.IndianRed
                    essayPanel.BackColor = Color.IndianRed
                    TurnColour(countChar, (inputTextBox.TextLength - countChar), Color.Red)
                    If loginForm.suddenDeath = True Then
                        timeDoneLabel.Text = "STOP"
                        startFadeBtn.Text = "START"
                        inputTextBox.ReadOnly = True
                        inputTextBox.Clear()
                        startType = False
                        wordTimer.Stop()
                        ChangeAllKeyColourGrey()
                    End If
                End If
            Else
                If (inputTextBox.TextLength > (countChar)) Then 'Checks if any inputted letter are incorrect, checks the 2nd+ incorrect
character
                    totalFaults += 1
                    currentFaults += 1
                    TurnColour(countChar, (inputTextBox.TextLength - countChar), Color.Red)
                    If essayPanel.BackColor <> Color.IndianRed Then
                        firstWordRichTextBox.BackColor = Color.IndianRed
                        secondWordRichTextBox.BackColor = Color.IndianRed
                        thirdWordRichTextBox.BackColor = Color.IndianRed
                        essayPanel.BackColor = Color.IndianRed
                    End If
                End If
                If inputTextBox.TextLength < (countChar + currentFaults + 1) Then 'Checks if a red letter has been backspaced
                    differenceOfFaults = (countChar + currentFaults) - inputTextBox.TextLength
                    currentFaults -= differenceOfFaults
                    TurnColour(countChar + currentFaults, differenceOfFaults, Color.Black)
                    If inputTextBox.TextLength < (countChar) And currentFaults <= 0 Then 'Checks if a green letter has been backspaced
                        currentFaults = 0
                        countChar -= (countChar - inputTextBox.TextLength) 'subtracts from counts accordingly
                        countCharEssay -= (countChar - inputTextBox.TextLength + 1)
                        If loginForm.showKeyboard = "ShowHighlight" Then
                            ChangeKeyColour(countChar) 'changes key highlighted to previous key
                        End If
                    End If
                    If currentFaults = 0 Then
                        firstWordRichTextBox.BackColor = Color.Gainsboro
                        secondWordRichTextBox.BackColor = Color.Gainsboro
                        thirdWordRichTextBox.BackColor = Color.Gainsboro
                        essayPanel.BackColor = Color.Gainsboro
                    End If
                End If
            End If
        Catch
        End Try
    End Sub

    Private Sub startFadeBtn_Click(sender As Object, e As EventArgs) Handles startFadeBtn.Click
        Select Case loginForm.extractWordFadeType
            Case "Random Words"
                words = FetchRandomWords("\TextFiles\randomWords.txt")
                countChar = 0
                countCharEssay = 0
                countWord = 0
                charList = IntoChar(words, countWord)
                RefreshWordOrder(countWord)
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeKeyColour(countChar)
```

```
                End If
            Case "Paragraphs"
                extract = FetchRandomParagraphs()
                words = IntoWords(extract)
                countChar = 0
                countWord = 0
                RefreshWordOrder(countWord)
                charList = IntoChar(words, countWord)
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeKeyColour(countChar)
                End If
            Case "Alphabet"
                words = FetchRandomWords("\TextFiles\alphabet.txt")
                countChar = 0
                countCharEssay = 0
                countWord = 0
                charList = IntoChar(words, countWord)
                RefreshWordOrder(countWord)
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeKeyColour(countChar)
                End If
            Case Else
                MessageBox.Show("Something has gone wrong", "Error")
        End Select
        If startFadeBtn.Text = "START" Then
            startType = False
            timeLeft = 0
            currentFaults = 0
            startingTimer = "CountDown"
            inputTextBox.Clear()
            startFadeBtn.Text = "STOP"
            PauseTimer(3)
        Else
            timeDoneLabel.Text = "TIME"
            inputTextBox.ReadOnly = True
            inputTextBox.Clear()
            startType = False
            startFadeBtn.Text = "START"
            If loginForm.showKeyboard = "ShowHighlight" Then
                ChangeAllKeyColourGrey()
            End If
            wordTimer.Stop()
        End If
    End Sub

    Private Sub wordTimer_Tick(sender As Object, e As EventArgs) Handles wordTimer.Tick
        Select Case loginForm.countType
            Case "CountDown"
                Countdown()
            Case "CountUp"
                CountUp()
        End Select
    End Sub

End Class
```

Statistics Form

```
Imports System.Data.OleDb
Imports System.Windows.Forms.DataVisualization.Charting

Public Class statUserForm

    'Declaring the variables
    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim leaderboardDataset As New DataSet
    Dim addUser As Boolean
    Dim chartDataset As New DataSet
    Dim previousUsername As String 'Stores the last username the user is loaded onto

    Function GetRowNumber(idNumber) 'To find the rowNumber (record number) of the user in the table
        Dim SQLGetUserRow = "SELECT * FROM userData" 'Fetches whole table
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUserRow, loginForm.conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        Dim rowNum As Integer
        rowNum = 0
        For Each row As DataRow In dataset.Tables("userData").Rows
            If row.Item("UserID") = Convert.ToInt32(idNumber) Then 'If the UserID in record matches UserID then return Row
                Exit For
            End If
            rowNum += 1
        Next row
        Return rowNum
    End Function

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, loginForm.conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(0).Item(0) 'Row 0 as its only 1 record fetched,
        usernameTextBox.Text = dataset.Tables("userData").Rows(0).Item(1) 'Item 'number' to signify the column value
```

```
        passwordTextBox.Text = dataset.Tables("userData").Rows(0).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(0).Item(3)
        If dataset.Tables("userData").Rows(0).Item(4) = "Yes" Then 'Changes checkbox value depending on value in record
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        previousUsername = usernameTextBox.Text.Trim
    End Sub

    Sub Max(columnName, labelName) 'Fetches record by Maximum value of either (WPM or Accuracy' for a specific user
        Dim SQLGetMax = "SELECT MAX (" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQLGetMax, loginForm.conn)
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString 'Rounds variable shown to 1 dp
    End Sub

    Sub Average(columnName, labelName) 'Fetches record by Average value of either (WPM or Accuracy' for a specific user
        Dim SQlGetAverage = "SELECT AVG(" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQlGetAverage, loginForm.conn)
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString 'Rounds variable shown to 1 dp
    End Sub

    Sub LeaderboardChange(indexValue) 'Depending on option selected in ComboBox, the SQL relating to leaderboardin DataGridViewer is
changed
        dataset.Clear()
        Dim SQLleaderboard As String
        Select Case indexValue
            Case 0
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.WPM) AS [Max Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY MAX(entryData.WPM) DESC;"
            Case 1
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, MAX(entryData.Accuracy) AS [Max Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY MAX(entryData.Accuracy) DESC;"
            Case 2
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.WPM) AS [Avg Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY AVG(entryData.WPM) DESC;"
            Case 3
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, AVG(entryData.Accuracy) AS [Avg Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username ORDER BY AVG(entryData.Accuracy) DESC;"
            Case 4
                SQLleaderboard = "SELECT userData.Username, Count(entryData.EntryID) AS [Amount Of Entries] FROM userData INNER JOIN
entryData ON userData.UserID = entryData.UserID GROUP BY userData.Username ORDER BY COUNT(entryData.EntryID) DESC;"
            Case Else
        End Select
        Dim commandLeaderboard As New OleDbDataAdapter(SQLleaderboard, loginForm.conn) 'SQL is excuted
        commandLeaderboard.Fill(leaderboardDataset, "Leaderboard")
        dataGridView.DataSource = leaderboardDataset.Tables("Leaderboard") 'Data is filled into the DataGridViewer
        leaderboardDataset.Tables.Remove("LeaderBoard") 'Reset and clear dataset
    End Sub


    Sub ChartWPMLoad()
        Dim SQLgetWPM = "SELECT DISTINCTROW Avg(entryData.WPM) AS WPM, entryData.Date FROM userData INNER JOIN entryData ON
userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLgetWPM, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "WPMTable")
        'Add a new series
        Dim SeriesName As New Series
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").BackColor = Color.WhiteSmoke
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Black
        userWPMChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Black
        SeriesName.Color = Color.DeepSkyBlue
        SeriesName.BorderWidth = 3
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "WPM"
        userWPMChart.DataSource = chartDataset.Tables("WPMTable")
        userWPMChart.Series.Add(SeriesName)
    End Sub

    Sub ChartAccuracyLoad()
        Dim SQLGetAccuray = "SELECT DISTINCTROW Avg(entryData.Accuracy) AS Accuracy, entryData.Date FROM userData INNER JOIN entryData
ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLGetAccuray, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "SQLGetAccuray")
        'Add a new series
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").BackColor = Color.WhiteSmoke
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Black
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Black
        Dim SeriesName As New Series
        SeriesName.Color = Color.DarkOrchid
        SeriesName.BorderWidth = 3
        SeriesName.Name = "Accuracy"
```

```
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "Accuracy"
        userAccuracyChart.DataSource = chartDataset.Tables("SQLGetAccuray")
        userAccuracyChart.Series.Add(SeriesName)
    End Sub

    Private Sub statUserForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'TODO: This line of code loads data into the 'UserDatabaseDataSet.entryData' table. You can move, or remove it, as needed.
        loginForm.UpdateLabelMode(Me)
        loginForm.UpdateLabelMode(Me)
        currentRow = GetRowNumber(loginForm.dataUserID)
        GetUserDetails(loginForm.dataUsername)
        leaderboardComboBox.SelectedIndex = 0
        Dim SQLuser = "SELECT * FROM userData"
        Dim SQLentry = "SELECT * FROM entryData"
        Dim commandUser = New OleDbDataAdapter(SQLuser, loginForm.conn)
        Dim commandEntry = New OleDbDataAdapter(SQLentry, loginForm.conn)
        commandUser.Fill(leaderboardDataset, "userData")
        commandUser.Fill(leaderboardDataset, "entryData")
        leaderboardDataset.Tables("userData").PrimaryKey = {leaderboardDataset.Tables("userData").Columns("UserID")}
        leaderboardDataset.Tables("entryData").PrimaryKey = {leaderboardDataset.Tables("userData").Columns("EntryID")}
        Dim userDataRelation As DataRelation = New DataRelation("Leaderboard", leaderboardDataset.Tables("userData").Columns("UserID"),
leaderboardDataset.Tables("entryData").Columns("UserID"))
        leaderboardDataset.Relations.Add(userDataRelation)
        Try
            Max("WPM", highestWPMLabel)
            Max("Accuracy", highestAccuracyLabel)
            Average("WPM", averageWPMLabel)
            Average("Accuracy", averageAccuracyLabel)
        Catch ex As Exception
            MessageBox.Show(ex.ToString)
        End Try
        ChartWPMLoad()
        ChartAccuracyLoad()
    End Sub
    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click

    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If

    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub
    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        If addUserBtn.Text = "&ADD USER" Then 'If is not in Add User mode
            Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. You'll be logged out",
"Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then
                    Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text
                    Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text
                    Dim commandDelete As New OleDbCommand(SQLDeleteUser, loginForm.conn)
                    Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, loginForm.conn)
                    commandDeleteData.ExecuteReader() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                    commandDelete.ExecuteScalar() 'Then delete user
                    MessageBox.Show("Your account has been successfully deleted, you will now be logged out", "Successful Deletion")
                    loginForm.LogOut() 'Data variables are cleaned of values
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            Catch ex As Exception
                MessageBox.Show("Something has gone wrong", "Error")
            End Try
        Else
            MessageBox.Show("You are In Add User Mode, you cannot delete an account you have Not created yet", "Error")
        End If
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear() 'Ease of use for clearing all inputs from textboxes
        passwordTextBox.Clear()
        nameTextBox.Clear()
```

```
        adminCheckBox.Checked = False
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails(loginForm.dataUsername)
            LeaderboardChange(leaderboardComboBox.SelectedIndex)
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click

        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
inputs have at least 5 characters for username and password
            MessageBox.Show("You have entered your credentials wrong. Please Try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';"
            Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                Dim username, password, myName, admin As String
                username = usernameTextBox.Text.Trim
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
                End Select
                Dim sqlAddUser As String
                If addUserBtn.Text = "&CANCEL ADD USER" Then
                    sqlAddUser = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);" 'For adding a new user
                Else
                    sqlAddUser = "UPDATE [userData] Set [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" &
myName & "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";" 'Updating current records credentials
                End If
                Dim commandInsert As New OleDbCommand(sqlAddUser, loginForm.conn)
                commandInsert.Parameters.AddWithValue("@username", username)
                commandInsert.Parameters.AddWithValue("@password", password)
                commandInsert.Parameters.AddWithValue("@name", myName)
                commandInsert.Parameters.AddWithValue("@admin", admin)
                commandInsert.ExecuteNonQuery()
                MessageBox.Show("User credentials have been modified, you are now logged in as the edited user", "Modification
Successful")
                loginForm.dataUsername = username
                loginForm.dataName = myName
                loginForm.dataAdmin = adminCheckBox.Checked
                If addUser = True Then 'Ejects user from statistics form if a new user has been added
                    addUser = False
                    Dim sqlGetUserID = "SELECT [UserID] FROM [userData] WHERE username = @username" 'Update data variable to new user
                    Dim commandUserID As New OleDbCommand(sqlGetUserID, loginForm.conn) 'UserID would only be changed if it's a new user
                    commandUserID.Parameters.AddWithValue("@username", username)
                    loginForm.dataUserID = commandUserID.ExecuteScalar()
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Close()
                End If
                GetUserDetails(loginForm.dataUsername) 'Updates user's credentials
                addUserBtn.Text = "&ADD USER"
            End If
        End If
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs) Handles refreshBtn.Click
        GetUserDetails(loginForm.dataUsername)
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub leaderboardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
leaderboardComboBox.SelectedIndexChanged
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

Admin Form

```
Imports System.Data.OleDb
Imports System.Globalization
Imports System.IO

Public Class statAdminForm
    Dim commandSelect As OleDbDataAdapter
    Dim totalRows As Int32
    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim addUser As Boolean
    Dim previousUsername

    Sub LoadDataGridUser() 'Load current users entries and output onto DataGridViewer
        If dataGridView.RowCount > 0 Then
            dataGridView.Rows.Clear()
        End If
        Dim SQLSelectData = "SELECT * FROM [entryData] WHERE UserID = " + userIDTextBox.Text 'Fetch entry records for the current user
        commandSelect = New OleDbDataAdapter(SQLSelectData, loginForm.conn)
        dataset.Clear()
        commandSelect.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Fill DataGridViewer with the [EntryData} table (table 1)
    End Sub

    Sub LoadDateDataGridUser(startDate, endDate) 'Same as the previous, however there are parameters for the dates
        Dim SQLSelectDateData = "SELECT * FROM [entryData] WHERE UserID =" & userIDTextBox.Text & " AND Date BETWEEN @start AND @end"
        Dim commandSelect As New OleDbCommand
        commandSelect.Connection = loginForm.conn
        commandSelect.CommandText = SQLSelectDateData
        commandSelect.Parameters.Add("@start", OleDbType.Date).Value = startDatePicker.Value
        commandSelect.Parameters.Add("@start", OleDbType.Date).Value = endDatePicker.Value
        dataset.Clear()
        Dim adapter As New OleDbDataAdapter(commandSelect)
        adapter.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Filling DataGridView with data from the dataset
    End Sub

    Sub GetUserDetails() 'Fetch user details and inputs into the textboxes and checkboxes
        Dim SQLGetUser = "SELECT * FROM [userData]"
        commandSelect = New OleDbDataAdapter(SQLGetUser, loginForm.conn)
        dataset.Clear()
        commandSelect.Fill(dataset, "userData") 'NOTE that the Row is dependent on currentRow variable (used to speifcy a specific user)
        userIDTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(0) 'Row 0 as its  only 1 record fetched
        usernameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(1) 'Item 'number' to signify the column value
        passwordTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(3)
        If dataset.Tables("userData").Rows(currentRow).Item(4) = "Yes" Then  'Changes checkbox value depending on value in record
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        previousUsername = usernameTextBox.Text.Trim
    End Sub

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes when a specific username has been
specified
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, loginForm.conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(0)
        usernameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(1)
        passwordTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(3)
        If dataset.Tables("userData").Rows(currentRow).Item(4) = "Yes" Then
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        previousUsername = usernameTextBox.Text.Trim
    End Sub


    Private Sub statAdminForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        currentRow = 0 'Resets varibles
        GetUserDetails() 'Loads in first user's record in textboxes
        totalRows = dataset.Tables(0).Rows.Count 'Fetches total number of accounts
        LoadDataGridUser() 'Loads user entries to DataGridViewer
        addUser = False
    End Sub


    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        typerForm.Show()
        Me.Hide()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
```

```
"Page Unavailable")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles searchBtn.Click
        Dim userSearch As String
        userSearch = userSearchTextBox.Text 'Takes the value from user input as a variable
        Dim SQLSelectData = "SELECT username from [userData] WHERE username = '" & userSearch & "';"
        Dim commandCheckUser As New OleDbCommand(SQLSelectData, loginForm.conn)
        Try
            If commandCheckUser.ExecuteScalar Is Nothing Then
                MessageBox.Show("No user has been found with this name", "Error")
            Else
                currentRow = 0 'CurrentRow needs to equal 0, as the Logic applied would have only one record be shown (only one record
for one user with that username). After when 'Previous or 'Next' btn is pressed, resets naivgation to start from 0
                GetUserDetails(userSearch) 'Updates textboxes
                LoadDataGridUser() 'Updates DataGridViewer with entries
            End If
        Catch ex As Exception
        End Try
    End Sub

    Private Sub nextBtn_Click(sender As Object, e As EventArgs) Handles nextBtn.Click 'totalRow variable is declared and values changed
with Line 87 (fetches the number of records (accounts)
        currentRow += 1
        Select Case currentRow
            Case totalRows 'reached the end of the table, go back to beginning
                currentRow = 0
        End Select
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub previousBtn_Click(sender As Object, e As EventArgs) Handles previousBtn.Click
        currentRow -= 1
        Select Case currentRow
            Case -1  'Reached the start of the table, go to the end
                currentRow = totalRows - 1
        End Select
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs)
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then 'If currently in Add User Mode, come out of it
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails()
            totalRows = dataset.Tables(0).Rows.Count
            LoadDataGridUser()
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. If you are deleting your own
account then you'll be logged out", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
        Try
            If result = DialogResult.Yes Then
                Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim commandDelete As New OleDbCommand(SQLDeleteUser, loginForm.conn)
                Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, loginForm.conn)
                commandDeleteData.ExecuteScalar() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                commandDelete.ExecuteScalar() 'Then delete user
                If userIDTextBox.Text = loginForm.dataUserID Then 'Checks if the account deleted is currently logged in
                    loginForm.Show()
                    Me.Close()
                Else
                    currentRow = 0 'Resets navigation of user accounts
                    GetUserDetails()
                    totalRows = dataset.Tables(0).Rows.Count 'Fetches new total accounts number
                    LoadDataGridUser()
                    MessageBox.Show("User has been successfully deleted", "Successful Deletion")
```

```
                End If
            End If
        Catch ex As Exception
            MessageBox.Show("Something has gone wrong", "Error")
        End Try
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear() 'Ease of use for clearing all inputs from textboxes
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
        Dim SQL As String
        Dim SQLUsername As String
        Dim SQLAdmin As String
        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
if there are no inputs in the textbox
            MessageBox.Show("You have entered your credentials wrong. Please try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';" 'SQL to
check if username is already present
            Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database, and whether is has been modified
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                Dim username, password, myName, admin As String 'Declares and assigns input values to variables
                username = usernameTextBox.Text.Trim 'Trim to avoid whitespace being present in database and when comparing for the
login process
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
                End Select
                If addUserBtn.Text = "&CANCEL ADD USER" Then 'If the Add User button was pressed, then Insert account to database
                    SQL = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);"
                    Dim commandInsert As New OleDbCommand(SQL, loginForm.conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteScalar()
                Else 'If not, update the currently logged on account
                    SQL = "UPDATE [userData] SET [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" & myName
& "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";"
                    Dim commandInsert As New OleDbCommand(SQL, loginForm.conn)
                    commandInsert.ExecuteScalar()
                End If
                MessageBox.Show("Action completed successfully", "Confirmation completed")
                currentRow = 0
                addUser = False
                addUserBtn.Text = "&ADD USER"
                GetUserDetails()
                totalRows = dataset.Tables(0).Rows.Count
                LoadDataGridUser()
            End If
        End If
    End Sub

    Private Sub searchDateBtn_Click(sender As Object, e As EventArgs) Handles searchDateBtn.Click
        LoadDateDataGridUser(startDatePicker.Value.ToShortDateString, endDatePicker.Value.ToShortDateString)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

Options Form

```
Public Class optionsForm

    Dim timeDurationArray As String() = {"15", "30", "45", "60"} 'String arrays containing the options for the comboboxes
    Dim countTypeArray As String() = {"CountDown", "CountUp"}
    Dim extractTypeArray As String() = {"Paragraphs", "Random Words"}
    Dim wordFadeExtractTypeArray As String() = {"Paragraphs", "Random Words", "Alphabet"}

    Sub AddToComboBox(comboBoxName, itemArray) 'When form loads in, the string array items are added into a specified ComboBox
        comboBoxName.Items.Clear() 'Clears a ComboBox of previous items
        For Each item As String In itemArray
            comboBoxName.Items.Add(item)
        Next
    End Sub
```

```
    Sub ChangeComboItem(comboBoxName, itemArray, item) 'Changes the index of the ComboBox depending on prexisting variable values
        comboBoxName.SelectedIndex = Array.IndexOf(itemArray, item)
    End Sub

    Sub ChangeCheckBoxMode(checkbox, mode) 'Changes the tick of the CheckBox depending on prexisting variable values
        checkbox.Checked = mode
    End Sub

    Sub ChangeComboBoxKeyboard(comboBoxName, item) 'ComboBox keyboard options changes variable value
        Select Case item
            Case "ShowHighlight"
                comboBoxName.selectedIndex = 0
            Case "Show"
                comboBoxName.selectedIndex = 1
            Case "Nothing"
                comboBoxName.selectedIndex = 2
        End Select
    End Sub

    Function changeItem(comboBoxName) 'Changes variable value dpedning on option chosen
        Dim variable As String
        variable = comboBoxName.SelectedItem.ToString()
        Return variable
    End Function

    Private Sub optionsForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        AddToComboBox(countComboBox, countTypeArray) 'Adds items to ComboBoxes
        AddToComboBox(timeDurationComboBox, timeDurationArray)
        AddToComboBox(extractComboBox, extractTypeArray)
        AddToComboBox(wordFadeExtractComboBox, wordFadeExtractTypeArray)
        ChangeComboItem(countComboBox, countTypeArray, loginForm.countType) 'Changes selected items based on previous history in the
form
        ChangeComboItem(timeDurationComboBox, timeDurationArray, loginForm.timeDuration)
        ChangeComboItem(extractComboBox, extractTypeArray, loginForm.extractType)
        ChangeComboItem(wordFadeExtractComboBox, wordFadeExtractTypeArray, loginForm.extractWordFadeType)
        ChangeCheckBoxMode(suddenDeathCheckBox, loginForm.suddenDeath)
        ChangeCheckBoxMode(capsCheckBox, loginForm.capitals)
        ChangeCheckBoxMode(punctuationCheckBox, loginForm.punctuation)
        ChangeComboBoxKeyboard(keyBoardComboBox, loginForm.showKeyboard)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub countComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles countComboBox.SelectedIndexChanged
        loginForm.countType = changeItem(countComboBox)
    End Sub

    Private Sub timeDurationComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
timeDurationComboBox.SelectedIndexChanged
        loginForm.timeDuration = changeItem(timeDurationComboBox)
    End Sub

    Private Sub extractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles extractComboBox.SelectedIndexChanged
        loginForm.extractType = changeItem(extractComboBox)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub suddenDeathCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles suddenDeathCheckBox.CheckedChanged
        loginForm.suddenDeath = suddenDeathCheckBox.Checked
```

```
    End Sub

    Private Sub capsCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles capsCheckBox.CheckedChanged
        loginForm.capitals = capsCheckBox.Checked
    End Sub

    Private Sub punctuationCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles punctuationCheckBox.CheckedChanged
        loginForm.punctuation = punctuationCheckBox.Checked
    End Sub


    Private Sub wordFadeExtractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
wordFadeExtractComboBox.SelectedIndexChanged
        loginForm.extractWordFadeType = changeItem(wordFadeExtractComboBox)
    End Sub

    Private Sub keyBoardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles keyBoardComboBox.SelectedIndexChanged
        Select Case keyBoardComboBox.SelectedIndex
            Case 0
                loginForm.showKeyboard = "ShowHighlight"
            Case 1
                loginForm.showKeyboard = "Show"
            Case 2
                loginForm.showKeyboard = "Nothing"
        End Select
    End Sub

    Private Sub changeFontBtn_Click(sender As Object, e As EventArgs) Handles changeFontBtn.Click
        If (changeFontDialog.ShowDialog() = DialogResult.OK) Then
            loginForm.globalFont = changeFontDialog.Font
        End If
    End Sub

    Private Sub linkLabel_LinkClicked(sender As Object, e As LinkLabelLinkClickedEventArgs) Handles linkLabel.LinkClicked
        Dim address = "https://support.microsoft.com/en-us/office/download-and-install-microsoft-365-access-runtime-185c5a32-8ba9-491e-
ac76-91cbe3ea09c9"
        linkLabel.LinkVisited = True
        System.Diagnostics.Process.Start(address)
    End Sub
End Class
```

Final Version

Login Form

```
Imports System.Data.OleDb

Public Class loginForm  'All friend variables are global that will be used across the form

    Friend connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\userDatabase.accdb" ' Connection string
used to connect to the database
    Public conn As New OleDbConnection
    Friend dataUsername As String
    Friend dataName As String
    Friend dataAdmin As Boolean
    Friend Shared dataUserID As String
    Friend timeDuration = "30" 'Option variables to be passed into various forms
    Friend countType = "CountDown"
    Friend suddenDeath = False
    Friend extractType = "Paragraphs"
    Friend extractWordFadeType = "Paragraphs"
    Friend capitals = True
    Friend punctuation = True
    Friend showKeyboard = "ShowHighlight"
    Friend entriesPresent = False 'Used to determine whether user can enter the Statistics form
    Friend globalFont As New Font("Calibri", 28, Font.Style.Regular)
    Friend attemptCount = 3 'Used as a counter for the attempts until applicatio nn closes
    Friend red As Color = Color.FromArgb(255, 128, 128) 'Used as colours for keyboard
    Friend lightGreen As Color = Color.FromArgb(128, 255, 128)
    Friend lightBlue As Color = Color.FromArgb(128, 255, 255)
    Friend purple As Color = Color.FromArgb(255, 128, 255)
    Friend yellow As Color = Color.FromArgb(255, 255, 128)
    Friend blue As Color = Color.FromArgb(204, 153, 255)
    Friend orange As Color = Color.FromArgb(255, 192, 128)
    Friend darkGreen As Color = Color.FromArgb(179, 229, 97)

    Sub UpdateLabelMode(formNow) 'Fetches the mode and updates the label, label is present in every page and is referenced there too
        formNow.modeLabel.Text = "MODE : " & extractType
    End Sub

    Sub ClearTextBox() 'After each entry or login the textbox is cleared of inputs
        usernameTextBox.Clear()
        passwordTextBox.Clear()
    End Sub

    Sub CheckStartDatabase()
        Try
            conn.Open()
        Catch ex As Exception
            MessageBox.Show("The Database connection cannot be initialised, and login and add user features are not available. You will
now be in Guest Mode and taken straight to a Typing Test", "Database Connection Error")
            dataUsername = "Guest"
            submitBtn.Enabled = False
            createAccBtn.Enabled = False
        End Try
    End Sub

    Sub LogOut() 'When logging out, the variables with data pertaining to the login is cleared
        dataUserID = ""
        dataUsername = ""
        dataName = ""
        dataAdmin = False
    End Sub

    Sub GetAmountEntries() 'Checks the amount of entries, with that user to stop any database errors
        Dim SQLGetEntries = "SELECT Count(entryData.EntryID) AS [Entries] FROM userData INNER JOIN entryData ON userData.UserID =
entryData.UserID GROUP BY userData.Username HAVING (userData.Username)='" & dataUsername & "';"
        Dim commandGetEntries = New OleDbCommand(SQLGetEntries, conn)
        If commandGetEntries.ExecuteScalar() Is Nothing Or commandGetEntries.ExecuteScalar() = "1" Then
            entriesPresent = False
        Else
            entriesPresent = True
        End If
    End Sub

    Sub deductAttempt() 'Deducts an entry from the counter and updates/shows the text label
        attemptCount -= 1
        Select Case attemptCount
            Case 2
                attemptsLabel.Visible = True
                attemptsLabel.Text = "You have 2 attempts left"
            Case 1
                attemptsLabel.Text = "You have 1 attempt left"
            Case 0
                Me.Close() 'After 3 attempts the program closes
        End Select
    End Sub

    Private Sub cancelBtn_Click(sender As Object, e As EventArgs) Handles cancelBtn.Click
        Application.Exit()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Try
            If usernameTextBox.Text.Trim = "" Or passwordTextBox.Text.Trim = "" Or usernameTextBox.TextLength < 5 Or
```

```
passwordTextBox.TextLength < 5 Then 'Validation : Checks to see if characters are in either TextBox
                MessageBox.Show("You have entered your credentials wrong. Make sure that the Username and Password are at least 5
characters in Length. Please try again", "Submit Credentials", MessageBoxButtons.OK, MessageBoxIcon.Error)
            Else
                Dim inputUsername As String
                Dim inputPassword As String
                Dim dataPassword As String
                inputUsername = usernameTextBox.Text.Trim 'Assiging User Inputs to variables, Trim Function takes out spaces
                inputPassword = passwordTextBox.Text.Trim
                Dim SQLpassword As String = "SELECT Password FROM [userData] WHERE Username= '" & inputUsername & "';" 'Formatting SQL
Code to fetch password where username matches
                Dim SQLadmin As String = "SELECT Admin FROM userData WHERE Username= '" & inputUsername & "';" 'SQL code to fetch Admin
Privileges
                Dim SQLuserID As String = "SELECT [UserID] FROM [userData] WHERE Username='" & inputUsername & "';"
                Dim commandPassword As New OleDbCommand(SQLpassword, conn)
                Dim commandAdmin As New OleDbCommand(SQLadmin, conn)
                Dim commandUserID As New OleDbCommand(SQLuserID, conn)
                dataPassword = commandPassword.ExecuteScalar()
                If dataPassword Is Nothing Or dataPassword <> inputPassword Then 'Checks if SQL fetch password fetches nothing (due to
wrong username) or does not match
                    MessageBox.Show("Username or Password is incorrect. Please try again", "Submit Credentials", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
                    ClearTextBox()
                    deductAttempt() 'minus attempt from list
                ElseIf dataPassword.ToString = inputPassword Then 'If the passwords match and are correct
                    dataUsername = inputUsername
                    dataUserID = commandUserID.ExecuteScalar()
                    Dim adminCheck = commandAdmin.ExecuteScalar()
                    Select Case adminCheck 'Checks admin privileges
                        Case "Yes"
                            dataAdmin = True
                        Case "No"
                            dataAdmin = False
                    End Select
                    ClearTextBox()
                    attemptCount = 3 'reset counter
                    attemptsLabel.Visible = False
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Hide() 'Hides form and does not close as this is the startup form and need variables associated with it
                End If
            End If
        Catch ex As NullReferenceException
            MessageBox.Show("Something went wrong, please try again", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
            ClearTextBox()
        End Try
    End Sub

    Private Sub createAccBtn_Click(sender As Object, e As EventArgs) Handles createAccBtn.Click
        Dim frm As New addUserForm
        frm.Show()
        Me.Hide()
    End Sub

    Private Sub loginForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        conn.ConnectionString = connectionString
        attemptsLabel.Visible = False
        CheckStartDatabase() 'Checks and opens the connection to the database
    End Sub

    Private Sub guestBtn_Click(sender As Object, e As EventArgs) Handles guestBtn.Click
        dataUsername = "Guest"
        ClearTextBox()
        Me.Hide()
        Dim frm As New typerForm
        frm.Show()
    End Sub

End Class
```

Add User Form

```
Imports System.Data.OleDb

Public Class addUserForm

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub submitBtn_Click(sender As Object, e As EventArgs) Handles submitBtn.Click
        Dim result As DialogResult
        If passTextBox.Text = retryPassTextBox.Text And passTextBox.Text.Trim.Length >= 5 And userTextBox.Text.Trim.Length >= 5 And
nameTextBox.Text.Trim.Length <> 0 Then 'Validation : Checking password entries match and that no textboxes and no characters in them
            result = MessageBox.Show("Are you sure?", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            If result = DialogResult.Yes Then
                Dim username, password, myName, admin As String
                username = userTextBox.Text.Trim
                password = passTextBox.Text.Trim
                myName = nameTextBox.Text.Trim '
                admin = "No"
```

```
                Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & username & "';"
                Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
                Dim checkUserPresent = commandCheck.ExecuteScalar()
                If checkUserPresent = username Then 'Checks if username is already present in database
                    MessageBox.Show("A user with this username is already present, please use a different username")
                Else
                    Dim sqlAddUser As String = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username,
@password, @name, @admin);"
                    Dim commandInsert As New OleDbCommand(sqlAddUser, loginForm.conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteNonQuery() 'Execute command to database
                    MessageBox.Show("User has been added")
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            End If
        Else
            MessageBox.Show("You have entered your credentials wrong.  Make sure that the Username and Password are at least 5
characters in length. Please try again", "Submit Credentials")
        End If
    End Sub
End Class
```

Typing Test Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions
Imports FakeItEasy

Public Class typerForm
    Dim extract As String
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
    Friend startType As Boolean
    Friend startingTimer = "Countdown"
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Function FetchRandomParagraphs() As String
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt")
        Dim essayExtract As String
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Int32
        randomNumber = Generator.Next(0, lineCount)
        essayExtract = lines.ElementAt(randomNumber).ToString
        If loginForm.punctuation = False Then
            essayExtract = Regex.Replace(essayExtract, "[ ](?=[ ])|[^-_A-Za-z0-9 ]", String.Empty)
        End If
        If loginForm.capitals = False Then 'if capital? is off then make the extract
            Return essayExtract.ToLower()
        Else
            Return essayExtract
        End If
    End Function

    Function FetchRandomWords() 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\randomWords.txt") 'initialises on array
on the words on each line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            If loginForm.capitals = True Then
                wordList = WithCaps(wordList)
            End If
            Return wordList
        End If
    End Function

    Function IntoWords(essay) As String() 'Dividing string into a words array
        Dim wordList As String()
        wordList = essay.Split(" ")
        Return wordList
```

```
    End Function

    Function IntoChar(wordList, y) As String() 'For the word that the user is on, it gets split up into an array of letters
        Dim stringInformation As New Globalization.StringInfo(wordList(y))
        Dim characterList(stringInformation.LengthInTextElements - 1) As String
        For i As Integer = 0 To stringInformation.LengthInTextElements - 1
            characterList(i) = stringInformation.SubstringByTextElements(i, 1)
        Next
        Return characterList
    End Function

    Function IntoEssay(wordList) 'With Random Words mode, need to get base extract as string, so converts wordList back to string
        Dim essaySpaces As String
        For Each word In wordList
            essaySpaces += word + " " 'Need to add dspaces as without, all the words will be one long string
        Next
        Dim essay As String = essaySpaces.Remove(essaySpaces.Length - 1, 1) 'From the for loop the string will end on a space which will
cause issues with the typing game so elimnating it solves the problem
        Return essay
    End Function

    Function WithCaps(wordList As String()) 'Adds capitals in random words mode
        Dim Generator As System.Random = New System.Random()
        For i = 0 To wordList.Length - 1
            Dim randomNumber = Generator.Next(0, 2) 'a random number is generated
            wordList(i) = CapsFirstLetter(wordList, i, randomNumber)
        Next
        Return wordList
    End Function

    Function CapsFirstLetter(wordList, i, randomNumber) 'Called by 'WithCaps' to capatlise the first letter of the word
        Dim characterArray() As Char = wordList(i).ToCharArray
        If randomNumber = 0 Then 'Making chances of Captilisation 1/2 chance from the random number
            characterArray(0) = Char.ToUpper(characterArray(0))
        End If
        Return New String(characterArray)
    End Function

    Sub TurnColourCharacters(start, lengthEnd, color)
        essayRichTextBox.SelectionStart = start
        essayRichTextBox.SelectionLength = lengthEnd
        essayRichTextBox.SelectionColor = color
    End Sub

    Sub TurnColourBackground(colourBack) 'Function to change the colour of the background
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Sub TurnColourBackground(colourFont, colourBack)
        essayRichTextBox.ForeColor = colourFont
        essayRichTextBox.BackColor = colourBack
        essayPanel.BackColor = colourBack
    End Sub

    Sub TimerPauseMode(duration)
        startType = False 'Starting type (puts the Countdown/Countup functino in pause mode)
        timeLeft = Convert.ToInt32(duration)
        wordTimer.Start()
    End Sub

    Sub Countdown()
        If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
            timeLeft -= 0.1
            timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
        Else
            If startType = True Then 'If currently playing a typing game then make the timer stop and measure
                startType = False
                inputTextBox.Clear()
                inputTextBox.ReadOnly = True
                timeDoneLabel.Text = "STOP"
                startBtn.Text = "START"
                startBtn.BackColor = Color.MediumPurple
                TurnColourBackground(Color.DimGray, Color.FromArgb(70, 72, 73))
                wordTimer.Stop()
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime()
            Else
                timeDoneLabel.Text = "GO" 'If not currently playing and its countdown at the start then reset timer and start countdown
at specifc time duration
                inputTextBox.ReadOnly = False
                startType = True
                TurnColourBackground(Color.WhiteSmoke, Color.Gray)
                timeLeft = timeDuration
                wordTimer.Stop()
                wordTimer.Start()
            End If
        End If
    End Sub

    Sub CountUp()
        If startingTimer = "CountDown" Then 'If we are in the countdown before type game stats then ...
            If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
                timeLeft -= 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'Resets timer to 0 and starts typing game
                timeLeft = 0
```

```
                wordTimer.Start()
                startingTimer = "CountUp"
                timeDoneLabel.Text = "GO"
                TurnColourBackground(Color.WhiteSmoke, Color.Gray)
                inputTextBox.ReadOnly = False
            End If
        Else
            If timeLeft < timeDuration Then 'Checks to see if time limit has been reached
                timeLeft += 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'When time limit has been reached then stop timer and measure
                timeDoneLabel.Text = "STOP"
                inputTextBox.ReadOnly = True
                inputTextBox.Clear()
                startType = False
                startBtn.Text = "START"
                startBtn.BackColor = Color.MediumPurple
                wordTimer.Stop()
                TurnColourBackground(Color.DimGray, Color.FromArgb(70, 72, 73))
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime()
            End If
        End If
    End Sub

    Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished before time limit has
been reached
            timeTakenNow = timeLeft
        Else 'countdown mode when game is finished beofre time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        Try 'Catches if there is an overflow exception from no inputs being typed into the Test
            accuracy = (100 - 100 * (totalFaults / countCharEssay))
        Catch ex As Exception
            accuracy = 0
        End Try
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1) 'Add results to labels to show to the user
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)
        If loginForm.dataUsername <> "Guest" Or countCharEssay = 0 Then 'Adding the entry into the database is not an option for a Guest
user
            Dim SQLinsertentry = "INSERT INTO [entryData] ([UserID], [Date], [WPM], [Accuracy], [Mode], [Time Duration], [Capital
Letters], [Punctuation]) VALUES (@userID, @date, @wpm, @accuracy, @mode, @timeduration, @capitals, @punctuation)"
            Dim commandInsertEntry As New OleDbCommand(SQLinsertentry, loginForm.conn) 'SQL for adding in the new entry in [entryTable]
            commandInsertEntry.Parameters.AddWithValue("@userID", loginForm.dataUserID)
            commandInsertEntry.Parameters.Add("@date", OleDbType.Date).Value = Date.Today()
            commandInsertEntry.Parameters.AddWithValue("@wpm", Math.Round(wordsPerMinute, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@accuracy", Math.Round(accuracy, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@mode", loginForm.countType.ToString)
            commandInsertEntry.Parameters.AddWithValue("@timeduration", loginForm.timeDuration.ToString)
            Select Case loginForm.capitals 'checks what captials variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "Yes")
                Case Else
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "No")
            End Select
            Select Case loginForm.punctuation 'checks what punctuation variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@punctuation", "Yes")
                Case Else
                    Select Case loginForm.extractType 'Undefined and No depends on whether it Random Words or Essay (Puncutation option
only affects Essay)
                        Case "CountDown"
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "No")
                        Case Else
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "UnDefined")
                    End Select
            End Select
            Try
                commandInsertEntry.ExecuteNonQuery()
            Catch e As Exception
                MessageBox.Show(e.ToString)
            End Try
        End If
        JumpToIndex(8)
    End Sub

    Sub ColourPanel(color, Panel) 'Function to Colour the Panel
        Panel.BackColor = color
    End Sub

    Sub ChangeKeyColour([char]) 'Checks the next key needed to be pressed and highlights key on keyboard
        Dim currentChar As String
        ChangeAllKeyColourGrey() 'Chagnes previous keys too all grey
        If charList.Length > ([char]) Then 'So that there is not a Out of Range Exception with charlist([char])
            currentChar = charList([char])
            Select Case currentChar.ToLower
                Case "a"
```

```
                    ColourPanel(loginForm.red, aPanel)
                Case "b"
                    ColourPanel(loginForm.purple, bPanel)
                Case "c"
                    ColourPanel(loginForm.lightBlue, cPanel)
                Case "d"
                    ColourPanel(loginForm.lightBlue, dPanel)
                Case "e"
                    ColourPanel(loginForm.lightBlue, ePanel)
                Case "f"
                    ColourPanel(loginForm.purple, fPanel)
                Case "g"
                    ColourPanel(loginForm.purple, gPanel)
                Case "h"
                    ColourPanel(loginForm.yellow, hPanel)
                Case "i"
                    ColourPanel(loginForm.blue, iPanel)
                Case "j"
                    ColourPanel(loginForm.yellow, jPanel)
                Case "k"
                    ColourPanel(loginForm.blue, kPanel)
                Case "l"
                    ColourPanel(loginForm.orange, lPanel)
                Case "m"
                    ColourPanel(loginForm.yellow, mPanel)
                Case "n"
                    ColourPanel(loginForm.yellow, nPanel)
                Case "o"
                    ColourPanel(loginForm.orange, oPanel)
                Case "p"
                    ColourPanel(loginForm.darkGreen, pPanel)
                Case "q"
                    ColourPanel(loginForm.red, qPanel)
                Case "r"
                    ColourPanel(loginForm.purple, rPanel)
                Case "s"
                    ColourPanel(loginForm.lightGreen, sPanel)
                Case "t"
                    ColourPanel(loginForm.purple, tPanel)
                Case "u"
                    ColourPanel(loginForm.yellow, uPanel)
                Case "v"
                    ColourPanel(loginForm.purple, vPanel)
                Case "w"
                    ColourPanel(loginForm.lightGreen, wPanel)
                Case "x"
                    ColourPanel(loginForm.lightGreen, xPanel)
                Case "y"
                    ColourPanel(loginForm.yellow, yPanel)
                Case "z"
                    ColourPanel(loginForm.red, zPanel)
                Case Else
            End Select
            If Char.IsUpper(charList([char])) = True Then 'Checks if letter is a capital
                ColourPanel(loginForm.red, shiftLeftPanel)
                ColourPanel(loginForm.darkGreen, shiftRightPanel)
            End If
        ElseIf [char] = charList.Length Then 'When end of a word press space
            ColourPanel(Color.Silver, spacePanel)
        End If
    End Sub

    Sub ChangeAllKeyColourGrey() 'Resets entire keyboard colour to 'Gray'
        ColourPanel(Color.Gray, aPanel)
        ColourPanel(Color.Gray, bPanel)
        ColourPanel(Color.Gray, cPanel)
        ColourPanel(Color.Gray, dPanel)
        ColourPanel(Color.Gray, ePanel)
        ColourPanel(Color.Gray, fPanel)
        ColourPanel(Color.Gray, gPanel)
        ColourPanel(Color.Gray, hPanel)
        ColourPanel(Color.Gray, iPanel)
        ColourPanel(Color.Gray, jPanel)
        ColourPanel(Color.Gray, kPanel)
        ColourPanel(Color.Gray, lPanel)
        ColourPanel(Color.Gray, mPanel)
        ColourPanel(Color.Gray, nPanel)
        ColourPanel(Color.Gray, oPanel)
        ColourPanel(Color.Gray, pPanel)
        ColourPanel(Color.Gray, qPanel)
        ColourPanel(Color.Gray, rPanel)
        ColourPanel(Color.Gray, sPanel)
        ColourPanel(Color.Gray, tPanel)
        ColourPanel(Color.Gray, uPanel)
        ColourPanel(Color.Gray, vPanel)
        ColourPanel(Color.Gray, wPanel)
        ColourPanel(Color.Gray, xPanel)
        ColourPanel(Color.Gray, yPanel)
        ColourPanel(Color.Gray, spacePanel)
        ColourPanel(Color.Gray, zPanel)
        ColourPanel(Color.Gray, shiftLeftPanel)
        ColourPanel(Color.Gray, shiftRightPanel)
    End Sub

    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If countWord = words.Length Then 'Check if essay has been completed
                timeDoneLabel.Text = "STOP"
```

```
                startBtn.Text = "START"
                startBtn.BackColor = Color.MediumPurple
                inputTextBox.ReadOnly = True
                inputTextBox.Clear()
                startType = False
                wordTimer.Stop()
                MeasureTime()
                Exit Sub
            End If
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'Checks if full
word is correct and the user has pressed spaces to go onto next word
                countChar = 0
                countCharEssay += 1
                countWord += 1
                essayRichTextBox.SelectionStart = countCharEssay + 1
                inputTextBox.Clear() 'Resets input box
                If countChar = inputTextBox.TextLength And words.Length <> countWord Then 'Goes onto next word
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                End If
            ElseIf (countChar + 1 = inputTextBox.TextLength) And (countChar <> charList.Length) And currentFaults = 0 Then 'Checks if
inputted letter is correct
                If (inputTextBox.Text.EndsWith(charList(countChar)) = True) Then
                    TurnColourCharacters(countCharEssay, 1, Color.Green)
                    countChar += 1
                    countCharEssay += 1
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Else 'Checks 1st incorrect character
                    currentFaults += 1
                    totalFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    essayPanel.BackColor = Color.IndianRed
                    essayRichTextBox.BackColor = Color.IndianRed
                    If loginForm.suddenDeath = True Then
                        timeDoneLabel.Text = "STOP"
                        startBtn.Text = "START"
                        startBtn.BackColor = Color.MediumPurple
                        inputTextBox.ReadOnly = True
                        inputTextBox.Clear()
                        startType = False
                        wordTimer.Stop()
                        TurnColourBackground(Color.DimGray, Color.FromArgb(70, 72, 73))
                        ChangeAllKeyColourGrey()
                    End If
                End If
            Else
                If (inputTextBox.TextLength > (countChar)) Then 'Checks if any inputted letter are incorrect, checks the 2nd+ incorrect
character
                    totalFaults += 1
                    currentFaults += 1
                    TurnColourCharacters(countCharEssay, (inputTextBox.TextLength - countChar), Color.Red)
                    If essayPanel.BackColor <> Color.IndianRed Then
                        essayPanel.BackColor = Color.IndianRed
                        essayRichTextBox.BackColor = Color.IndianRed
                    End If
                End If
                If inputTextBox.TextLength < (countChar + currentFaults + 1) Then 'Checks if a letter has been backspaced
                    differenceOfFaults = (countChar + currentFaults) - inputTextBox.TextLength
                    currentFaults -= differenceOfFaults
                    TurnColourCharacters(countCharEssay + currentFaults, differenceOfFaults, Color.WhiteSmoke)
                    If inputTextBox.TextLength < (countChar) And currentFaults <= 0 Then 'Checks if a green letter has been backspaced
                        currentFaults = 0
                        countChar -= (countChar - inputTextBox.TextLength) 'subtracts from counts
                        countCharEssay -= (countChar - inputTextBox.TextLength + 1)
                        If loginForm.showKeyboard = "ShowHighlight" Then
                            ChangeKeyColour(countChar) 'changes key highlighted to previous key
                        End If
                    End If
                    If currentFaults = 0 Then
                        essayPanel.BackColor = Color.Gray
                        essayRichTextBox.BackColor = Color.Gray
                    End If
                End If
            End If
        Catch
        End Try
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub


    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Public Sub JumpToIndex(ByVal I As Integer)
        For Each control As Control In Me.Controls
            If control.TabIndex = I Then
                control.Focus()
```

```
            End If
        Next
    End Sub

    Private Sub startBtn_Click(sender As Object, e As EventArgs) Handles startBtn.Click
        essayRichTextBox.ShowSelectionMargin = True
        JumpToIndex(7)
        If startBtn.Text = "START" Then
            Select Case loginForm.extractType
                Case "Random Words"
                    words = FetchRandomWords()
                    extract = IntoEssay(words)
                    essayRichTextBox.Text() = extract
                    countChar = 0
                    countWord = 0
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Case "Paragraphs"
                    extract = FetchRandomParagraphs()
                    words = IntoWords(extract)
                    essayRichTextBox.Text() = extract
                    countChar = 0
                    countWord = 0
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Case Else
                    MessageBox.Show("Something has gone wrong", "Error")
            End Select
            startType = False
            countCharEssay = 0
            currentFaults = 0
            timeLeft = 0
            startingTimer = "CountDown"
            inputTextBox.Clear()
            TurnColourCharacters(0, essayRichTextBox.TextLength, Color.Gray)
            essayPanel.BackColor = Color.FromArgb(70, 72, 73)
            essayRichTextBox.BackColor = Color.FromArgb(70, 72, 73)
            startBtn.Text = "STOP"
            startBtn.BackColor = Color.Crimson
            TimerPauseMode(3)
        Else
            timeDoneLabel.Text = "TIME"
            inputTextBox.ReadOnly = True
            inputTextBox.Clear()
            startType = False
            startBtn.Text = "START"
            startBtn.BackColor = Color.MediumPurple
            TurnColourBackground(Color.Gray, Color.FromArgb(70, 72, 73))
            If loginForm.showKeyboard = "ShowHighlight" Then
                ChangeAllKeyColourGrey()
            End If
            wordTimer.Stop()
        End If
    End Sub

    Private Sub wordTimer_Tick() Handles wordTimer.Tick
        Select Case loginForm.countType
            Case "CountDown"
                Countdown()
            Case "CountUp"
                CountUp()
        End Select
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True And loginForm.dataUsername <> "Guest" Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Hide()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub typerForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        If loginForm.showKeyboard = "Nothing" Then
            keyBoardPanel.Visible = False 'hiding the keybaord from veiw
        End If
        essayRichTextBox.Font = loginForm.globalFont
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
```

```
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

WordFade Form

```
Imports System.Data.OleDb
Imports System.IO
Imports System.Text.RegularExpressions

Public Class wordFadeForm
    Dim extract As String
    Dim words As String()
    Dim charList As String()
    Dim countChar As Int32
    Dim countCharEssay As Int32
    Dim countWord As Int32
    Dim currentFaults As Int32
    Dim totalFaults As Int32
    Friend timeLeft As Double
    Friend startType As Boolean
    Friend startingTimer = "Countdown"
    Dim essayExtract As String
    Dim timeDuration = Convert.ToDouble(loginForm.timeDuration)

    Function FetchRandomParagraphs() As String
        Dim lines As String() = File.ReadAllLines(Directory.GetCurrentDirectory() & "\TextFiles\paragraphs.txt")
        Dim lineCount = lines.Length
        Dim Generator As System.Random = New System.Random()
        Dim randomNumber As Int32
        randomNumber = Generator.Next(0, lineCount)
        essayExtract = lines.ElementAt(randomNumber).ToString
        If loginForm.punctuation = False Then
            essayExtract = Regex.Replace(essayExtract, "[ ](?=[ ])|[^-_A-Za-z0-9 ]", String.Empty)
        End If
        If loginForm.capitals = False Then 'if capital? is off then make the extract
            Return essayExtract.ToLower()
        Else
            Return essayExtract
        End If
    End Function

    Function IntoWords(essay) As String() 'Dividing string into a words array
        Dim wordList As String()
        wordList = essay.Split(" ")
        Return wordList
    End Function

    Function FetchRandomWords(filePath) 'Fetches random words in the 'randomWord.txt'
        Dim lines() As String = File.ReadAllLines(Directory.GetCurrentDirectory() & filePath) 'initialises on array on the words on each
line
        Dim Generator As System.Random = New System.Random()
        Dim length As Int32 = lines.Length
        Dim randomNumber As Int32
        Dim list As New List(Of String)
        Dim wordList As String()
        If lines.Length = 0 Then 'Catches error where the text file is empty
            MessageBox.Show("No words have been found, Something has gone wrong", "Error")
        Else
            For i = 1 To 50
                randomNumber = Generator.Next(0, lines.Length) 'Adds a random word to essay by randomly generating an index value and
fetching word in there
                list.Add(lines(randomNumber))
            Next
            wordList = list.ToArray
            If loginForm.capitals = True Then
                wordList = WithCaps(wordList)
            End If
            Return wordList
        End If
    End Function

    Sub RefreshWordOrder(count)
        Select Case words.Length - count - 1
            Case 1
                thirdWordRichTextBox.Clear()
                secondWordRichTextBox.Text = words(count + 1)
                firstWordRichTextBox.Text = words(count)
            Case 0
                thirdWordRichTextBox.Clear()
                secondWordRichTextBox.Clear()
                firstWordRichTextBox.Text = words(count)
            Case -1

            Case Else
                thirdWordRichTextBox.Text = words(count + 2)
                secondWordRichTextBox.Text = words(count + 1)
                firstWordRichTextBox.Text = words(count)
        End Select
    End Sub
```

```
    Function WithCaps(wordList As String()) 'Adds capitals in random words mode
        Dim Generator As System.Random = New System.Random()
        For i = 0 To wordList.Length - 1
            Dim randomNumber = Generator.Next(0, 2) 'a random number is generated
            wordList(i) = CapsFirstLetter(wordList, i, randomNumber)
        Next
        Return wordList
    End Function

    Function CapsFirstLetter(wordList, i, randomNumber)
        Dim characterArray() As Char = wordList(i).ToCharArray
        If randomNumber = 0 Then 'Making chances of Captilisation 1/2 chance from the random number
            characterArray(0) = Char.ToUpper(characterArray(0))
        End If
        Return New String(characterArray)
    End Function

    Function IntoChar(wordList, y) As String() 'For the word that the user is on, it gets split up into an array of letters
        Dim stringInformation As New Globalization.StringInfo(wordList(y))
        Dim characterList(stringInformation.LengthInTextElements - 1) As String
        For i As Integer = 0 To stringInformation.LengthInTextElements - 1
            characterList(i) = stringInformation.SubstringByTextElements(i, 1)
        Next
        Return characterList
    End Function

    Sub TurnColour(start, lengthEnd, color)
        firstWordRichTextBox.SelectionStart = start
        firstWordRichTextBox.SelectionLength = lengthEnd
        firstWordRichTextBox.SelectionColor = color
    End Sub

    Sub TurnColourTextBox(textbox, start, lengthEnd, color)
        textbox.SelectionStart = start
        textbox.SelectionLength = lengthEnd
        textbox.SelectionColor = color
    End Sub

    Sub TurnColourBackgroundWordFade(colourFont, colourBack)
        firstWordRichTextBox.ForeColor = colourFont
        secondWordRichTextBox.ForeColor = colourFont
        thirdWordRichTextBox.ForeColor = colourFont
        essayPanel.BackColor = colourBack
        firstWordRichTextBox.BackColor = colourBack
        secondWordRichTextBox.BackColor = colourBack
        thirdWordRichTextBox.BackColor = colourBack
    End Sub

    Sub PauseTimer(duration)
        timeLeft = Convert.ToInt32(duration)
        wordTimer.Start()
    End Sub

    Sub Countdown()
        If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
            timeLeft -= 0.1
            timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
        Else
            If startType = True Then 'If currently playing a typing game then make the timer stop and measure
                timeDoneLabel.Text = "STOP"
                inputTextBox.ReadOnly = True
                inputTextBox.Clear()
                startType = False
                startFadeBtn.Text = "START"
                startFadeBtn.BackColor = Color.SteelBlue
                wordTimer.Stop()
                TurnColourBackgroundWordFade(Color.DimGray, Color.FromArgb(70, 72, 73))
                MeasureTime()
            Else
                timeDoneLabel.Text = "GO" 'If not currently playing and its countdown at the start then reset timer and start countdown
at specifc time duration
                TurnColourBackgroundWordFade(Color.WhiteSmoke, Color.Gray)
                inputTextBox.ReadOnly = False
                startType = True
                timeLeft = timeDuration
                wordTimer.Stop()
                wordTimer.Start()
            End If
        End If
    End Sub

    Sub CountUp()
        If startingTimer = "CountDown" Then 'If we are in the countdown before type game stats then ..
            If timeLeft > 0 Then 'Checks to see if needing to countdown is still valid, then decreases by 0.1
                timeLeft -= 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
            Else 'Resets timer to 0 and starts typing game
                timeLeft = 0
                startingTimer = "CountUp"
                timeDoneLabel.Text = "GO"
                TurnColourBackgroundWordFade(Color.WhiteSmoke, Color.Gray)
                inputTextBox.ReadOnly = False
            End If
        Else
            If timeLeft < timeDuration Then 'Checks to see if time limit has been reached
                timeLeft += 0.1
                timeDoneLabel.Text = Math.Round(timeLeft, 1) & "s"
```

```
            Else 'When time limit has been reached then stop timer and measure
                timeDoneLabel.Text = "STOP"
                TurnColourBackgroundWordFade(Color.WhiteSmoke, Color.Gray)
                startFadeBtn.Text = "START"
                inputTextBox.ReadOnly = True
                inputTextBox.Clear()
                startType = False
                wordTimer.Stop()
                If loginForm.showKeyboard = "ShowHighlight" Then
                    ChangeAllKeyColourGrey()
                End If
                MeasureTime()
            End If
        End If
    End Sub
    Sub MeasureTime()
        Dim wordsPerMinute As Double
        Dim accuracy As Int32
        Dim timeTakenNow As Double
        timeDoneLabel.Text = " "
        If timeLeft = 0 Or timeLeft >= timeDuration Then 'When countdown mode goes to 0 then set time taken to the full duration or When
countup mode goes to time limit
            timeTakenNow = timeDuration
        ElseIf timeLeft < timeDuration And startingTimer = "CountUp" Then 'in countup mode when game is finished beofre time limit has
been reached
            timeTakenNow = timeLeft
        Else 'countdown mode when game is finished beofre time limit has been reached
            timeTakenNow = timeDuration - timeLeft
        End If
        wordsPerMinute = ((countCharEssay) / 5) / (timeTakenNow / 60.0) 'WPM, words are defined as every 5 characters correctly typed
        Try 'Catches if there is an overflow exception from no inputs being typed into the Test
            accuracy = (100 - 100 * (totalFaults / countCharEssay))
        Catch ex As Exception
            accuracy = 0
        End Try
        timeTakenLabel.Text = "TIME : " & Math.Round(timeTakenNow, 1)
        wmpLabel.Text = "WPM : " & Math.Round(wordsPerMinute, 1)
        accuracyLabel.Text = "ACCURACY : " & Math.Round(accuracy, 1)
        If loginForm.dataUsername <> "Guest" And loginForm.extractWordFadeType <> "Alphabet" Then 'Adding the entry into the database is
not an option for a Guest user or for the extract type Alphabet
            Dim SQLinsertentry = "INSERT INTO [entryData] ([UserID], [Date], [WPM], [Accuracy], [Mode], [Time Duration], [Capital
Letters], [Punctuation]) VALUES (@userID, @date, @wpm, @accuracy, @mode, @timeduration, @capitals, @punctuation)"
            Dim commandInsertEntry As New OleDbCommand(SQLinsertentry, loginForm.conn) 'SQL for adding in the new entry in [entryTable]
            commandInsertEntry.Parameters.AddWithValue("@userID", loginForm.dataUserID)
            commandInsertEntry.Parameters.Add("@date", OleDbType.Date).Value = Date.Today()
            commandInsertEntry.Parameters.AddWithValue("@wpm", Math.Round(wordsPerMinute, 2).ToString)
            commandInsertEntry.Parameters.AddWithValue("@accuracy", accuracy.ToString)
            commandInsertEntry.Parameters.AddWithValue("@mode", loginForm.countType.ToString)
            commandInsertEntry.Parameters.AddWithValue("@timeduration", loginForm.timeDuration.ToString)
            Select Case loginForm.capitals 'checks what captials variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "Yes")
                Case Else
                    commandInsertEntry.Parameters.AddWithValue("@capitals", "No")
            End Select
            Select Case loginForm.punctuation 'checks what punctuation variable is and changes value inserted accordingly
                Case True
                    commandInsertEntry.Parameters.AddWithValue("@punctuation", "Yes")
                Case Else
                    Select Case loginForm.extractType 'Undefined and No depends on whether it Random Words or Essay (Puncutation option
only affects Essay)
                        Case "CountDown"
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "No")
                        Case Else
                            commandInsertEntry.Parameters.AddWithValue("@punctuation", "UnDefined")
                    End Select
            End Select
            Try
                commandInsertEntry.ExecuteNonQuery()

            Catch e As Exception
                MessageBox.Show(e.ToString)
            End Try
        End If
        JumpToIndex(8)
    End Sub

    Sub ColorPanel(color, Panel) 'Function to Colour the Panel
        Panel.BackColor = color
    End Sub

    Sub ChangeKeyColour([char]) 'Checks the next key needed to be pressed and highlights key on keyboard
        Dim currentChar As String
        ChangeAllKeyColourGrey() 'Chagnes previous keys too all grey
        If charList.Length > ([char]) Then 'So that there is not a Out of Range Exception with charlist([char])
            currentChar = charList([char])
            Select Case currentChar.ToLower
                Case "a"
                    ColorPanel(loginForm.red, aPanel)
                Case "b"
                    ColorPanel(loginForm.purple, bPanel)
                Case "c"
                    ColorPanel(loginForm.lightBlue, cPanel)
                Case "d"
                    ColorPanel(loginForm.lightBlue, dPanel)
                Case "e"
                    ColorPanel(loginForm.lightBlue, ePanel)
                Case "f"
                    ColorPanel(loginForm.purple, fPanel)
```

```
                Case "g"
                    ColorPanel(loginForm.purple, gPanel)
                Case "h"
                    ColorPanel(loginForm.yellow, hPanel)
                Case "i"
                    ColorPanel(loginForm.blue, iPanel)
                Case "j"
                    ColorPanel(loginForm.yellow, jPanel)
                Case "k"
                    ColorPanel(loginForm.blue, kPanel)
                Case "l"
                    ColorPanel(loginForm.orange, lPanel)
                Case "m"
                    ColorPanel(loginForm.yellow, mPanel)
                Case "n"
                    ColorPanel(loginForm.yellow, nPanel)
                Case "o"
                    ColorPanel(loginForm.orange, oPanel)
                Case "p"
                    ColorPanel(loginForm.darkGreen, pPanel)
                Case "q"
                    ColorPanel(loginForm.red, qPanel)
                Case "r"
                    ColorPanel(loginForm.purple, rPanel)
                Case "s"
                    ColorPanel(loginForm.lightGreen, sPanel)
                Case "t"
                    ColorPanel(loginForm.purple, tPanel)
                Case "u"
                    ColorPanel(loginForm.yellow, uPanel)
                Case "v"
                    ColorPanel(loginForm.purple, vPanel)
                Case "w"
                    ColorPanel(loginForm.lightGreen, wPanel)
                Case "x"
                    ColorPanel(loginForm.lightGreen, xPanel)
                Case "y"
                    ColorPanel(loginForm.yellow, yPanel)
                Case "z"
                    ColorPanel(loginForm.red, zPanel)
                Case Else
            End Select
            If Char.IsUpper(charList([char])) = True Then 'Checks if letter is a capital
                ColorPanel(loginForm.red, shiftLeftPanel)
                ColorPanel(loginForm.darkGreen, shiftRightPanel)
            End If
        ElseIf [char] = charList.Length Then 'When end of a word press space
            ColorPanel(Color.Silver, spacePanel)
        End If
    End Sub

    Sub ChangeAllKeyColourGrey() 'Colours all keys back to grey
        ColorPanel(Color.Gray, aPanel)
        ColorPanel(Color.Gray, bPanel)
        ColorPanel(Color.Gray, cPanel)
        ColorPanel(Color.Gray, dPanel)
        ColorPanel(Color.Gray, ePanel)
        ColorPanel(Color.Gray, fPanel)
        ColorPanel(Color.Gray, gPanel)
        ColorPanel(Color.Gray, hPanel)
        ColorPanel(Color.Gray, iPanel)
        ColorPanel(Color.Gray, jPanel)
        ColorPanel(Color.Gray, kPanel)
        ColorPanel(Color.Gray, lPanel)
        ColorPanel(Color.Gray, mPanel)
        ColorPanel(Color.Gray, nPanel)
        ColorPanel(Color.Gray, oPanel)
        ColorPanel(Color.Gray, pPanel)
        ColorPanel(Color.Gray, qPanel)
        ColorPanel(Color.Gray, rPanel)
        ColorPanel(Color.Gray, sPanel)
        ColorPanel(Color.Gray, tPanel)
        ColorPanel(Color.Gray, uPanel)
        ColorPanel(Color.Gray, vPanel)
        ColorPanel(Color.Gray, wPanel)
        ColorPanel(Color.Gray, xPanel)
        ColorPanel(Color.Gray, yPanel)
        ColorPanel(Color.Gray, zPanel)
        ColorPanel(Color.Gray, spacePanel)
        ColorPanel(Color.Gray, shiftLeftPanel)
        ColorPanel(Color.Gray, shiftRightPanel)
    End Sub


    Private Sub wordFadeForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        firstWordRichTextBox.Font = New Font(loginForm.globalFont.Name, 56, loginForm.globalFont.Style)
        secondWordRichTextBox.Font = New Font(loginForm.globalFont.Name, 28, loginForm.globalFont.Style)
        thirdWordRichTextBox.Font = New Font(loginForm.globalFont.Name, 16, loginForm.globalFont.Style)
        loginForm.UpdateLabelMode(Me)
        If loginForm.showKeyboard = "Nothing" Then
            keyBoardPanel.Visible = False 'hiding the keybaord from veiw
        End If
    End Sub

    Public Sub JumpToIndex(ByVal I As Integer)
        For Each control As Control In Me.Controls
            If control.TabIndex = I Then
                control.Focus()
            End If
```

```
        Next
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub inputTextBox_TextChanged(sender As Object, e As EventArgs) Handles inputTextBox.TextChanged
        Dim differenceOfFaults As Int32
        Try
            If countWord = words.Length Then 'Check if essay has been completed
                timeDoneLabel.Text = "STOP"
                startFadeBtn.Text = "START"
                startFadeBtn.BackColor = Color.MediumPurple
                inputTextBox.ReadOnly = True
                inputTextBox.Clear()
                startType = False
                wordTimer.Stop()
                MeasureTime()
                Exit Sub
            End If
            If (countChar + 1 = inputTextBox.TextLength) And (InStr(inputTextBox.Text, words(countWord) + " ") > 0) Then 'Checks if full
word is correct and the user has pressed spaces to go onto next word
                countChar = 0
                countCharEssay += 1
                countWord += 1
                RefreshWordOrder(countWord)
                inputTextBox.Clear() 'Resets input box
                If countChar = inputTextBox.TextLength And words.Length = countWord Then 'Checks if full essay has been completed
                    timeDoneLabel.Text = "STOP"
                    startFadeBtn.Text = "START"
                    inputTextBox.ReadOnly = True
                    inputTextBox.Clear()
                    startType = False
                    wordTimer.Stop()
                    MeasureTime()
                Else 'Goes onto next word
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                End If
            ElseIf (countChar + 1 = inputTextBox.TextLength) And (countChar <> charList.Length) And currentFaults = 0 Then
                If (inputTextBox.Text.EndsWith(charList(countChar)) = True) And (countChar + 1 = inputTextBox.TextLength) Then 'Checks
if inputted letter is correct
                    TurnColour(countChar, 1, Color.Green)
                    countChar += 1
                    countCharEssay += 1
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Else 'Checks 1st incorrect character
                    currentFaults += 1
                    totalFaults += 1
                    firstWordRichTextBox.BackColor = Color.IndianRed
                    secondWordRichTextBox.BackColor = Color.IndianRed
                    thirdWordRichTextBox.BackColor = Color.IndianRed
                    essayPanel.BackColor = Color.IndianRed
                    TurnColour(countChar, (inputTextBox.TextLength - countChar), Color.Red)
                    If loginForm.suddenDeath = True Then
                        timeDoneLabel.Text = "STOP"
                        startFadeBtn.Text = "START"
                        startFadeBtn.BackColor = Color.SteelBlue
```

```
                        inputTextBox.ReadOnly = True
                        inputTextBox.Clear()
                        startType = False
                        wordTimer.Stop()
                        TurnColourBackgroundWordFade(Color.DimGray, Color.FromArgb(70, 72, 73))
                        ChangeAllKeyColourGrey()
                    End If
                End If
            Else
                If (inputTextBox.TextLength > (countChar)) Then 'Checks if any inputted letter are incorrect, checks the 2nd+ incorrect
character
                    totalFaults += 1
                    currentFaults += 1
                    TurnColour(countChar, (inputTextBox.TextLength - countChar), Color.Red)
                    If essayPanel.BackColor <> Color.IndianRed Then
                        firstWordRichTextBox.BackColor = Color.IndianRed
                        secondWordRichTextBox.BackColor = Color.IndianRed
                        thirdWordRichTextBox.BackColor = Color.IndianRed
                        essayPanel.BackColor = Color.IndianRed
                    End If
                End If
                If inputTextBox.TextLength < (countChar + currentFaults + 1) Then 'Checks if a red letter has been backspaced
                    differenceOfFaults = (countChar + currentFaults) - inputTextBox.TextLength
                    currentFaults -= differenceOfFaults
                    TurnColour(countChar + currentFaults, differenceOfFaults, Color.WhiteSmoke)
                    If inputTextBox.TextLength < (countChar) And currentFaults <= 0 Then 'Checks if a green letter has been backspaced
                        currentFaults = 0
                        countChar -= (countChar - inputTextBox.TextLength) 'subtracts from counts accordingly
                        countCharEssay -= (countChar - inputTextBox.TextLength + 1)
                        If loginForm.showKeyboard = "ShowHighlight" Then
                            ChangeKeyColour(countChar) 'changes key highlighted to previous key
                        End If
                    End If
                    If currentFaults = 0 Then
                        firstWordRichTextBox.BackColor = Color.Gray
                        secondWordRichTextBox.BackColor = Color.Gray
                        thirdWordRichTextBox.BackColor = Color.Gray
                        essayPanel.BackColor = Color.Gray
                    End If
                End If
            End If
        Catch
        End Try
    End Sub

    Private Sub startFadeBtn_Click(sender As Object, e As EventArgs) Handles startFadeBtn.Click
        JumpToIndex(7)
        If startFadeBtn.Text = "START" Then
            Select Case loginForm.extractWordFadeType
                Case "Random Words"
                    words = FetchRandomWords("\TextFiles\randomWords.txt")
                    countChar = 0
                    countCharEssay = 0
                    countWord = 0
                    charList = IntoChar(words, countWord)
                    RefreshWordOrder(countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Case "Paragraphs"
                    extract = FetchRandomParagraphs()
                    words = IntoWords(extract)
                    countChar = 0
                    countWord = 0
                    RefreshWordOrder(countWord)
                    charList = IntoChar(words, countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Case "Alphabet"
                    words = FetchRandomWords("\TextFiles\alphabet.txt")
                    countChar = 0
                    countCharEssay = 0
                    countWord = 0
                    charList = IntoChar(words, countWord)
                    RefreshWordOrder(countWord)
                    If loginForm.showKeyboard = "ShowHighlight" Then
                        ChangeKeyColour(countChar)
                    End If
                Case Else
                    MessageBox.Show("Something has gone wrong", "Error")
            End Select
            startType = False
            timeLeft = 0
            currentFaults = 0
            startingTimer = "CountDown"
            inputTextBox.Clear()
            TurnColourBackgroundWordFade(Color.Gray, Color.FromArgb(70, 72, 73))
            startFadeBtn.Text = "STOP"
            startFadeBtn.BackColor = Color.Crimson
            PauseTimer(3)
        Else
            timeDoneLabel.Text = "TIME"
            inputTextBox.ReadOnly = True
            inputTextBox.Clear()
            startType = False
            startFadeBtn.Text = "START"
            startFadeBtn.BackColor = Color.SteelBlue
            TurnColourBackgroundWordFade(Color.Gray, Color.FromArgb(70, 72, 73))
```

```
            If loginForm.showKeyboard = "ShowHighlight" Then
                ChangeAllKeyColourGrey()
            End If
            wordTimer.Stop()
        End If
    End Sub

    Private Sub wordTimer_Tick(sender As Object, e As EventArgs) Handles wordTimer.Tick
        Select Case loginForm.countType
            Case "CountDown"
                Countdown()
            Case "CountUp"
                CountUp()
        End Select
    End Sub

End Class
```

Statistics Form

```
Imports System.Data.OleDb
Imports System.Windows.Forms.DataVisualization.Charting

Public Class statUserForm

    'Declaring the variables
    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim leaderboardDataset As New DataSet
    Dim addUser As Boolean
    Dim chartDataset As New DataSet
    Dim previousUsername As String 'Stores the last username the user is loaded onto

    Function GetRowNumber(idNumber) 'To find the rowNumber (record number) of the user in the table
        Dim SQLGetUserRow = "SELECT * FROM userData" 'Fetches whole table
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUserRow, loginForm.conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        Dim rowNum As Integer
        rowNum = 0
        For Each row As DataRow In dataset.Tables("userData").Rows
            If row.Item("UserID") = Convert.ToInt32(idNumber) Then 'If the UserID in record matches UserID then return Row
                Exit For
            End If
            rowNum += 1
        Next row
        Return rowNum
    End Function

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, loginForm.conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(0).Item(0)  'Row 0 as its only 1 record fetched
        usernameTextBox.Text = dataset.Tables("userData").Rows(0).Item(1) 'Item 'number' to signify the column value
        passwordTextBox.Text = dataset.Tables("userData").Rows(0).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(0).Item(3)
        If dataset.Tables("userData").Rows(0).Item(4) = "Yes" Then  'Changes checkbox value depending on value in record
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        previousUsername = usernameTextBox.Text.Trim
    End Sub

    Sub Max(columnName, labelName) 'Fetches record by Maximum value of either (WPM or Accuracy' for a specific user
        Dim SQLGetMax = "SELECT MAX (" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQLGetMax, loginForm.conn)
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString  'Rounds variable shown to 1 dp
    End Sub

    Sub Average(columnName, labelName) 'Fetches record by Average value of either (WPM or Accuracy' for a specific user
        Dim SQlGetAverage = "SELECT AVG(" & columnName & ") FROM entryData WHERE UserID =" & loginForm.dataUserID & ";"
        Dim commandGet = New OleDbCommand(SQlGetAverage, loginForm.conn)
        labelName.Text += Math.Round(commandGet.ExecuteScalar, 1).ToString 'Rounds variable shown to 1 dp
    End Sub

    Sub LeaderboardChange(indexValue)  'Depending on option selected in ComboBox, the SQL relating to leaderboardin DataGridViewer is
changed
        dataset.Clear()
        Dim SQLleaderboard As String
        Select Case indexValue
            Case 0
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Max(entryData.WPM) AS [Max Of WPM] FROM userData INNER JOIN
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 1
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Max(entryData.Accuracy) AS [Max Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 2
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Avg(entryData.Accuracy) AS [Avg Of Accuracy] FROM userData INNER
JOIN entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 3
                SQLleaderboard = "SELECT DISTINCTROW userData.Username, Avg(entryData.WPM) AS [Avg Of WPM] FROM userData INNER JOIN
```

```
entryData ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username;"
            Case 4
                SQLleaderboard = "SELECT userData.Username, Count(entryData.EntryID) AS [Amount Of Entries] FROM userData INNER JOIN
entryData ON userData.UserID = entryData.UserID GROUP BY userData.Username ORDER BY Count(entryData.EntryID) DESC;"
            Case Else
        End Select
        Dim commandLeaderboard As New OleDbDataAdapter(SQLleaderboard, loginForm.conn) 'SQL is excuted
        commandLeaderboard.Fill(leaderboardDataset, "Leaderboard")
        dataGridView.DataSource = leaderboardDataset.Tables("Leaderboard")  'Data is filled into the DataGridViewer
        leaderboardDataset.Tables.Remove("LeaderBoard") 'Reset and clear dataset
    End Sub

    Sub ChartWPMLoad()
        Dim SQLgetWPM = "SELECT DISTINCTROW Avg(entryData.WPM) AS WPM, entryData.Date FROM userData INNER JOIN entryData ON
userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLgetWPM, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "WPMTable")
        'Add a new series
        Dim SeriesName As New Series
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Gray
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Gray
        userWPMChart.ChartAreas("ChartArea1").BackColor = Color.FromArgb(70, 72, 73)
        userWPMChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Gainsboro
        SeriesName.Color = Color.DeepSkyBlue
        SeriesName.BorderWidth = 3
        userWPMChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Gainsboro
        userWPMChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Gainsboro
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userWPMChart.DataSource = chartDataset.Tables("WPMTable")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "WPM"
        userWPMChart.Series.Add(SeriesName)
    End Sub

    Sub ChartAccuracyLoad()
        Dim SQLGetAccuray = "SELECT DISTINCTROW Avg(entryData.Accuracy) AS Accuracy, entryData.Date FROM userData INNER JOIN entryData
ON userData.[UserID] = entryData.[UserID] GROUP BY userData.Username, entryData.Date HAVING (((userData.Username)='" &
loginForm.dataUsername & "')) ORDER BY entryData.Date;"
        Dim commandGetWPM As New OleDbDataAdapter(SQLGetAccuray, loginForm.conn)
        commandGetWPM.Fill(chartDataset, "SQLGetAccuray")
        'Add a new series
        Dim SeriesName As New Series
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorGrid.LineColor = Color.Gray
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorGrid.LineColor = Color.Gray
        userAccuracyChart.ChartAreas("ChartArea1").BackColor = Color.FromArgb(70, 72, 73)
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.MajorTickMark.LineColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.MajorTickMark.LineColor = Color.Gainsboro
        SeriesName.Color = Color.DarkOrange
        SeriesName.BorderWidth = 3
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LineColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LineColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisX.LabelStyle.ForeColor = Color.Gainsboro
        userAccuracyChart.ChartAreas("ChartArea1").AxisY.LabelStyle.ForeColor = Color.Gainsboro
        SeriesName.Name = "WPM"
        SeriesName.ChartType = SeriesChartType.Line
        SeriesName.IsVisibleInLegend = False
        userAccuracyChart.DataSource = chartDataset.Tables("SQLGetAccuray")
        SeriesName.XValueMember = "Date"
        SeriesName.YValueMembers = "Accuracy"
        userAccuracyChart.Series.Add(SeriesName)
    End Sub

    Private Sub statUserForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'TODO: This line of code loads data into the 'UserDatabaseDataSet.entryData' table. You can move, or remove it, as needed.
        Me.EntryDataTableAdapter.Fill(Me.UserDatabaseDataSet.entryData)
        loginForm.UpdateLabelMode(Me)
        currentRow = GetRowNumber(loginForm.dataUserID)
        GetUserDetails(loginForm.dataUsername)
        leaderboardComboBox.SelectedIndex = 0
        Dim SQLuser = "SELECT * FROM userData"
        Dim SQLentry = "SELECT * FROM entryData"
        Dim commandUser = New OleDbDataAdapter(SQLuser, loginForm.conn)
        Dim commandEntry = New OleDbDataAdapter(SQLentry, loginForm.conn)
        commandUser.Fill(leaderboardDataset, "userData")
        commandUser.Fill(leaderboardDataset, "entryData")
        leaderboardDataset.Tables("userData").PrimaryKey = {leaderboardDataset.Tables("userData").Columns("UserID")}
        leaderboardDataset.Tables("entryData").PrimaryKey = {leaderboardDataset.Tables("userData").Columns("EntryID")}
        Dim userDataRelation As DataRelation = New DataRelation("Leaderboard", leaderboardDataset.Tables("userData").Columns("UserID"),
leaderboardDataset.Tables("entryData").Columns("UserID"))
        leaderboardDataset.Relations.Add(userDataRelation)
        Try
            Max("WPM", highestWPMLabel)
            Max("Accuracy", highestAccuracyLabel)
            Average("WPM", averageWPMLabel)
            Average("Accuracy", averageAccuracyLabel)
        Catch ex As Exception
            MessageBox.Show(ex.ToString)
        End Try
        ChartWPMLoad()
        ChartAccuracyLoad()
    End Sub

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
```

```
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If

    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        If addUserBtn.Text = "&ADD USER" Then
            Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. You'll be logged out",
"Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
            Try
                If result = DialogResult.Yes Then
                    Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text
                    Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text
                    Dim commandDelete As New OleDbCommand(SQLDeleteUser, loginForm.conn)
                    Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, loginForm.conn)
                    commandDeleteData.ExecuteReader() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                    commandDelete.ExecuteScalar() 'Then delete user
                    MessageBox.Show("Your account has been successfully deleted, you will now be logged out", "Successful Deletion")
                    loginForm.LogOut() 'Data variables are cleaned of values
                    Dim frm As New loginForm
                    frm.Show()
                    Me.Close()
                End If
            Catch ex As Exception
                MessageBox.Show("Something has gone wrong", "Error")
            End Try
        Else
            MessageBox.Show("You are in Add User Mode, you cannot delete an account you have not created yet", "Error")
        End If
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear()
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails(loginForm.dataUsername)
            LeaderboardChange(leaderboardComboBox.SelectedIndex)
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
inputs have at least 5 characters for username and password
            MessageBox.Show("You have entered your credentials wrong. Please try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';"
            Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                Dim username, password, myName, admin As String
                username = usernameTextBox.Text.Trim
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
```

```
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
                End Select
                Dim sqlAddUser As String
                If addUserBtn.Text = "&CANCEL ADD USER" Then
                    sqlAddUser = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);" 'For adding a new user
                Else
                    sqlAddUser = "UPDATE [userData] SET [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" &
myName & "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";" 'Updating current records credentials
                End If
                Dim commandInsert As New OleDbCommand(sqlAddUser, loginForm.conn)
                commandInsert.Parameters.AddWithValue("@username", username)
                commandInsert.Parameters.AddWithValue("@password", password)
                commandInsert.Parameters.AddWithValue("@name", myName)
                commandInsert.Parameters.AddWithValue("@admin", admin)
                commandInsert.ExecuteNonQuery()
                MessageBox.Show("User credentials have been modified, you are now logged in as the edited user", "Modification
Successful")
                loginForm.dataUsername = username
                loginForm.dataName = myName
                loginForm.dataAdmin = adminCheckBox.Checked
                If addUser = True Then 'Ejects user from statistics form if a new user has been added
                    addUser = False
                    Dim sqlGetUserID = "SELECT [UserID] FROM [userData] WHERE username = @username" 'Update data variable to new user
                    Dim commandUserID As New OleDbCommand(sqlGetUserID, loginForm.conn) 'UserID would only be changed if it's a new user
                    commandUserID.Parameters.AddWithValue("@username", username)
                    loginForm.dataUserID = commandUserID.ExecuteScalar()
                    Dim frm As New typerForm
                    frm.Show()
                    Me.Close()
                End If
                GetUserDetails(loginForm.dataUsername) 'Updates user's credentials
                addUserBtn.Text = "&ADD USER"
            End If
        End If
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs) Handles refreshBtn.Click
        GetUserDetails(loginForm.dataUsername)
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub leaderboardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
leaderboardComboBox.SelectedIndexChanged
        LeaderboardChange(leaderboardComboBox.SelectedIndex)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

Admin Form

```
Imports System.Data.OleDb
Imports System.Globalization
Imports System.IO

Public Class statAdminForm
    Dim totalRows As Int32
    Dim currentRow As Int32
    Dim dataset As New DataSet
    Dim addUser As Boolean
    Dim previousUsername

    Sub LoadDataGridUser() 'Load current users entries and output onto DataGridViewer
        Dim commandSelect As New OleDbDataAdapter
        If dataGridView.RowCount > 0 Then
            dataGridView.Rows.Clear()
        End If
        Dim SQLSelectData = "SELECT * FROM [entryData] WHERE UserID = " + userIDTextBox.Text 'Fetch entry records for the current user
        commandSelect = New OleDbDataAdapter(SQLSelectData, loginForm.conn)
        dataset.Clear()
        commandSelect.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Fill DataGridViewer with the [EntryData} table (table 1)
    End Sub

    Sub LoadDateDataGridUser(startDate, endDate) 'Same as the previous, however there are parameters for the dates
        Dim SQLSelectDateData = "SELECT * FROM [entryData] WHERE UserID =" & userIDTextBox.Text & " AND Date BETWEEN @start AND @end"
        Dim commandSelect As New OleDbCommand
        commandSelect.Connection = loginForm.conn
        commandSelect.CommandText = SQLSelectDateData
        commandSelect.Parameters.Add("@start", OleDbType.Date).Value = startDatePicker.Value
        commandSelect.Parameters.Add("@start", OleDbType.Date).Value = endDatePicker.Value
        dataset.Clear()
        Dim adapter As New OleDbDataAdapter(commandSelect)
        adapter.Fill(dataset, "entryData")
        dataGridView.DataSource = dataset.Tables(1) 'Filling DataGridView with data from the dataset
```

```
    End Sub

    Sub GetUserDetails() 'Fetch user details and inputs into the textboxes and checkboxes
        Dim SQLGetUser = "SELECT * FROM [userData]"
        Dim commandSelect As New OleDbDataAdapter(SQLGetUser, loginForm.conn)
        dataset.Clear()
        commandSelect.Fill(dataset, "userData") 'NOTE that the Row is dependent on currentRow variable (used to speifcy a specific user)
        userIDTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(0) 'Row 0 as its  only 1 record fetched
        usernameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(1) 'Item 'number' to signify the column value
        passwordTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(3)
        If dataset.Tables("userData").Rows(currentRow).Item(4) = "Yes" Then  'Changes checkbox value depending on value in record
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        previousUsername = usernameTextBox.Text.Trim
    End Sub

    Sub GetUserDetails(username) 'Fetch user details and inputs into the textboxes and checkboxes when a specific username has been
specified
        Dim SQLGetUser = "SELECT * FROM userData WHERE username = '" & username & "'"
        Dim commandGetDetails = New OleDbDataAdapter(SQLGetUser, loginForm.conn)
        dataset.Clear()
        commandGetDetails.Fill(dataset, "userData")
        userIDTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(0)
        usernameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(1)
        passwordTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(2)
        nameTextBox.Text = dataset.Tables("userData").Rows(currentRow).Item(3)
        If dataset.Tables("userData").Rows(currentRow).Item(4) = "Yes" Then
            adminCheckBox.Checked = True
        Else
            adminCheckBox.Checked = False
        End If
        previousUsername = usernameTextBox.Text.Trim
    End Sub


    Private Sub statAdminForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        loginForm.UpdateLabelMode(Me)
        currentRow = 0 'Resets varibles
        GetUserDetails() 'Loads in first user's record in textboxes
        totalRows = dataset.Tables(0).Rows.Count 'Fetches total number of accounts
        LoadDataGridUser() 'Loads user entries to DataGridViewer
        addUser = False
    End Sub


    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        typerForm.Show()
        Me.Hide()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub optBtn_Click(sender As Object, e As EventArgs) Handles optBtn.Click
        Dim frm As New optionsForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub
    Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles searchBtn.Click
        Dim userSearch As String
        userSearch = userSearchTextBox.Text 'Takes the value from user input as a variable
        Dim SQLSelectData = "SELECT username from [userData] WHERE username = '" & userSearch & "';"
        Dim commandCheckUser As New OleDbCommand(SQLSelectData, loginForm.conn)
        Try
            If commandCheckUser.ExecuteScalar Is Nothing Then
                MessageBox.Show("No user has been found with this name", "Error")
            Else
                currentRow = 0 'CurrentRow needs to equal 0, as the Logic applied would have only one record be shown (only one record
for one user with that username). After when 'Previous or 'Next' btn is pressed, resets naivgation to start from 0
                GetUserDetails(userSearch) 'Updates textboxes
                LoadDataGridUser() 'Updates DataGridViewer with entries
            End If
        Catch ex As Exception
        End Try
    End Sub

    Private Sub nextBtn_Click(sender As Object, e As EventArgs) Handles nextBtn.Click
        currentRow += 1
        Select Case currentRow
            Case totalRows 'reached the end of the table, go back to beginning
                currentRow = 0
```

```
        End Select
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub previousBtn_Click(sender As Object, e As EventArgs) Handles previousBtn.Click
        currentRow -= 1
        Select Case currentRow
            Case -1 'Reached the start of the table, go to the end
                currentRow = totalRows - 1
        End Select
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub refreshBtn_Click(sender As Object, e As EventArgs)
        GetUserDetails() 'Updates textboxes
        LoadDataGridUser() 'Updates DataGridViewer with entries
    End Sub

    Private Sub addUserBtn_Click(sender As Object, e As EventArgs) Handles addUserBtn.Click
        If addUser = True Then 'If currently in Add User Mode, come out of it
            currentRow = 0
            addUser = False
            addUserBtn.Text = "&ADD USER"
            GetUserDetails()
            totalRows = dataset.Tables(0).Rows.Count
            LoadDataGridUser()
        Else
            addUser = True
            addUserBtn.Text = "&CANCEL ADD USER"
            userIDTextBox.Clear()
            usernameTextBox.Clear()
            passwordTextBox.Clear()
            nameTextBox.Clear()
            adminCheckBox.Checked = False
        End If
    End Sub

    Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        Dim result = MessageBox.Show("Are you sure? This includes deleting all of this user's entries. If you are deleting your own
account then you'll be logged out", "Submit Credentials", MessageBoxButtons.YesNo) 'Last chance to edit entry
        Try
            If result = DialogResult.Yes Then
                Dim SQLDeleteUser = "DELETE FROM [userData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim SQLDeleteUserData = "DELETE FROM [entryData] WHERE UserID = " & userIDTextBox.Text & ";"
                Dim commandDelete As New OleDbCommand(SQLDeleteUser, loginForm.conn)
                Dim commandDeleteData As New OleDbCommand(SQLDeleteUserData, loginForm.conn)
                commandDeleteData.ExecuteScalar() 'Deleting Entries from Typing Test comes first as it includes a foreign key
                commandDelete.ExecuteScalar() 'Then delete user
                If userIDTextBox.Text = loginForm.dataUserID Then 'Checks if the account deleted is currently logged in
                    loginForm.Show()
                    Me.Close()
                Else
                    currentRow = 0 'Resets navigation of user accounts
                    GetUserDetails()
                    totalRows = dataset.Tables(0).Rows.Count 'Fetches new total accounts number
                    LoadDataGridUser()
                    MessageBox.Show("User has been successfully deleted", "Successful Deletion")
                End If
            End If
        Catch ex As Exception
            MessageBox.Show("Something has gone wrong", "Error")
        End Try
    End Sub

    Private Sub clearBtn_Click(sender As Object, e As EventArgs) Handles clearBtn.Click
        usernameTextBox.Clear() 'Ease of use for clearing all inputs from textboxes
        passwordTextBox.Clear()
        nameTextBox.Clear()
        adminCheckBox.Checked = False
    End Sub

    Private Sub confirmEditBtn_Click(sender As Object, e As EventArgs) Handles confirmEditBtn.Click
        Dim SQL As String
        Dim SQLUsername As String
        Dim SQLAdmin As String
        If passwordTextBox.Text.Trim.Length < 5 Or usernameTextBox.Text.Trim.Length < 5 Or nameTextBox.Text.Trim.Length = 0 Then 'Checks
if there are no inputs in the textbox
            MessageBox.Show("You have entered your credentials wrong. Please try again", "Submit Credentials")
        Else
            Dim SQLCheckUser As String = "SELECT [Username] FROM userData WHERE [Username]='" & usernameTextBox.Text.Trim & "';" 'SQL to
check if username is already present
            Dim commandCheck As New OleDbCommand(SQLCheckUser, loginForm.conn)
            Dim checkUserPresent = commandCheck.ExecuteScalar()
            If checkUserPresent = usernameTextBox.Text.Trim And previousUsername <> usernameTextBox.Text.Trim Then 'Checks if username
is already present in database, and whether is has been modified
                MessageBox.Show("A user with this username is already present, please use a different username")
            Else
                Dim username, password, myName, admin As String 'Declares and assigns input values to variables
                username = usernameTextBox.Text.Trim 'Trim to avoid whitespace being present in database and when comparing for the
login process
                password = passwordTextBox.Text.Trim
                myName = nameTextBox.Text.Trim
                Select Case adminCheckBox.Checked
                    Case True
                        admin = "Yes"
                    Case Else
                        admin = "No"
```

```
                End Select
                If addUserBtn.Text = "&CANCEL ADD USER" Then 'If the Add User button was pressed, then Insert account to database
                    SQL = "INSERT INTO [userData] ([Username], [Password], [Name], [Admin]) VALUES (@username, @password, @name,
@admin);"
                    Dim commandInsert As New OleDbCommand(SQL, loginForm.conn)
                    commandInsert.Parameters.AddWithValue("@username", username)
                    commandInsert.Parameters.AddWithValue("@password", password)
                    commandInsert.Parameters.AddWithValue("@name", myName)
                    commandInsert.Parameters.AddWithValue("@admin", admin)
                    commandInsert.ExecuteScalar()
                Else 'If not, update the currently logged on account
                    SQL = "UPDATE [userData] SET [Username] = '" & username & "', [Password] = '" & password & "', [Name] = '" & myName
& "' , [Admin] = '" & admin & "' WHERE UserID = " & userIDTextBox.Text & ";"
                    Dim commandInsert As New OleDbCommand(SQL, loginForm.conn)
                    commandInsert.ExecuteScalar()
                End If
                MessageBox.Show("Action completed successfully", "Confirmation completed")
                currentRow = 0
                addUser = False
                addUserBtn.Text = "&ADD USER"
                GetUserDetails()
                totalRows = dataset.Tables(0).Rows.Count
                LoadDataGridUser()
            End If
        End If
    End Sub

    Private Sub searchDateBtn_Click(sender As Object, e As EventArgs) Handles searchDateBtn.Click
        LoadDateDataGridUser(startDatePicker.Value.ToShortDateString, endDatePicker.Value.ToShortDateString)
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

End Class
```

Options Form

```
Public Class optionsForm

    Dim timeDurationArray As String() = {"15", "30", "45", "60"}
    Dim countTypeArray As String() = {"CountDown", "CountUp"}
    Dim extractTypeArray As String() = {"Paragraphs", "Random Words"}
    Dim wordFadeExtractTypeArray As String() = {"Paragraphs", "Random Words", "Alphabet"}

    Sub AddToComboBox(comboBoxName, itemArray)
        comboBoxName.Items.Clear()
        For Each item As String In itemArray
            comboBoxName.Items.Add(item)
        Next
    End Sub

    Sub ChangeComboItem(comboBoxName, itemArray, item)
        comboBoxName.SelectedIndex = Array.IndexOf(itemArray, item)
    End Sub

    Sub ChangeCheckBoxMode(checkbox, mode)
        checkbox.Checked = mode
    End Sub

    Sub ChangeComboBoxKeyboard(comboBoxName, item)
        Select Case item
            Case "ShowHighlight"
                comboBoxName.selectedIndex = 0
            Case "Show"
                comboBoxName.selectedIndex = 1
            Case "Nothing"
                comboBoxName.selectedIndex = 2
        End Select
    End Sub

    Function changeItem(comboBoxName)
        Dim variable As String
        variable = comboBoxName.SelectedItem.ToString()
        Return variable
    End Function

    Private Sub typingBtn_Click(sender As Object, e As EventArgs) Handles typingBtn.Click
        Dim frm As New typerForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub wordFadeBtn_Click(sender As Object, e As EventArgs) Handles wordFadeBtn.Click
        Dim frm As New wordFadeForm
        frm.Show()
        Me.Close()
    End Sub

    Private Sub statsBtn_Click(sender As Object, e As EventArgs) Handles statsBtn.Click
        loginForm.GetAmountEntries()
```

```
        If loginForm.entriesPresent = True Then
            Dim frm As New statUserForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You will have access to this section when you have completed the Typing Test or Word Fade at least twice",
"Page Unavailable")
        End If
    End Sub

    Private Sub allStatsBtn_Click(sender As Object, e As EventArgs) Handles allStatsBtn.Click
        If loginForm.dataAdmin = True Then
            Dim frm As New statAdminForm
            frm.Show()
            Me.Close()
        Else
            MessageBox.Show("You do not have access to this", "Error")
        End If
    End Sub

    Private Sub backBtn_Click(sender As Object, e As EventArgs) Handles backBtn.Click
        loginForm.LogOut()
        loginForm.Show()
        Me.Close()
    End Sub

    Private Sub optionsForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        AddToComboBox(countComboBox, countTypeArray) 'Adds items to ComboBoxes
        AddToComboBox(timeDurationComboBox, timeDurationArray)
        AddToComboBox(extractComboBox, extractTypeArray)
        AddToComboBox(wordFadeExtractComboBox, wordFadeExtractTypeArray)
        ChangeComboItem(countComboBox, countTypeArray, loginForm.countType) 'Changes selected items based on previous history in the
form
        ChangeComboItem(timeDurationComboBox, timeDurationArray, loginForm.timeDuration)
        ChangeComboItem(extractComboBox, extractTypeArray, loginForm.extractType)
        ChangeComboItem(wordFadeExtractComboBox, wordFadeExtractTypeArray, loginForm.extractWordFadeType)
        ChangeCheckBoxMode(suddenDeathCheckBox, loginForm.suddenDeath)
        ChangeCheckBoxMode(capsCheckBox, loginForm.capitals)
        ChangeCheckBoxMode(punctuationCheckBox, loginForm.punctuation)
        ChangeComboBoxKeyboard(keyBoardComboBox, loginForm.showKeyboard)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub countComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles countComboBox.SelectedIndexChanged
        loginForm.countType = changeItem(countComboBox)
    End Sub

    Private Sub timeDurationComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
timeDurationComboBox.SelectedIndexChanged
        loginForm.timeDuration = changeItem(timeDurationComboBox)
    End Sub

    Private Sub extractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles extractComboBox.SelectedIndexChanged
        loginForm.extractType = changeItem(extractComboBox)
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub suddenDeathCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles suddenDeathCheckBox.CheckedChanged
        loginForm.suddenDeath = suddenDeathCheckBox.Checked
        loginForm.UpdateLabelMode(Me)
    End Sub

    Private Sub capsCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles capsCheckBox.CheckedChanged
        loginForm.capitals = capsCheckBox.Checked
    End Sub

    Private Sub punctuationCheckBox_CheckedChanged(sender As Object, e As EventArgs) Handles punctuationCheckBox.CheckedChanged
        loginForm.punctuation = punctuationCheckBox.Checked
    End Sub

    Private Sub wordFadeExtractComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
wordFadeExtractComboBox.SelectedIndexChanged
        loginForm.extractWordFadeType = changeItem(wordFadeExtractComboBox)
    End Sub

    Private Sub keyBoardComboBox_SelectedIndexChanged(sender As Object, e As EventArgs) Handles keyBoardComboBox.SelectedIndexChanged
        Select Case keyBoardComboBox.SelectedIndex
            Case 0
                loginForm.showKeyboard = "ShowHighlight"
            Case 1
                loginForm.showKeyboard = "Show"
            Case 2
                loginForm.showKeyboard = "Nothing"
        End Select
    End Sub

    Private Sub changeFontBtn_Click(sender As Object, e As EventArgs) Handles changeFontBtn.Click
        If (changeFontDialog.ShowDialog() = DialogResult.OK) Then
            loginForm.globalFont = changeFontDialog.Font
        End If
    End Sub

    Private Sub linkLabel_LinkClicked(sender As Object, e As LinkLabelLinkClickedEventArgs) Handles linkLabel.LinkClicked
        Dim address = "https://support.microsoft.com/en-us/office/download-and-install-microsoft-365-access-runtime-185c5a32-8ba9-491e-
ac76-91cbe3ea09c9"
        linkLabel.LinkVisited = True
        System.Diagnostics.Process.Start(address)
    End Sub
End Class
```

Bibliography

Here below are the websites/articles that aided me in learning and developing my solution.

Stack Overflow. [Online]. Available at: https://stackoverflow.com/

Microsoft. Visual Basic Documentation. [Online]. Available at: https://docs.microsoft.com/en-us/dotnet/visual-basic/

Manipulating Arrays

https://www.dotnetheaven.com/article/how-to-manipulate-arrays-in-vb.net

Accessing Textiles

https://stackoverflow.com/questions/22358290/read-a-text-file-in-vb

Writing SQL Queries and executing them in Database:

https://social.msdn.microsoft.com/forums/en-US/e4344a25-93e5-40d4-a97d-c5ee0b573577/running-access-query-in-vbnet
https://www.codeproject.com/Questions/443188/Access-Select-query-in-vb-net

Existing Solutions

Typeracer. [Online]. Available at: https://play.typeracer.com/

TouchTyper. [Online]. Available at: http://touchtyper.net/en/

Touch Typing Study. [Online]. Available at: https://www.typingstudy.com/

Typsey. [Software]. Download at: https://www.typesy.com/

Keyblaze. [Software]. Download at: https://www.nchsoftware.com/typingtutor/index.html

RataType. [Online]. Available at: https://www.ratatype.com/