2020

Programming Project

JHONNY MEDRANO-CHUQUIRIMA
CENTER NUMBER:10854
CANDIDATE NUMBER:8059
ST MICHAEL'S CATHOLIC COLLEGE

CONTENTS

ANALYSIS

INTRODUCTION

Economics is a fast-demanding subject amongst schools, the subject can be really challenging because there is a lot of equations and scenarios that will be tested in an exam, the more practice they get will benefit them in the real exam.

DEFINING THE PROBLEM AND STAKEHOLDERS

A QUIZ THAT IS DESIGNED TO TEST ECONOMICS STUDENTS.

My quiz will be testing the skills of economic students. The students will have to create a login in order to participate in this quiz so their data and score can be saved when they finish the quiz. It will be a multiple-choice question and the student will have to pick one of the options to get the answer right, once they get the answer right, they will get one mark and their score will increment by one each time. If they get the answer wrong, then it means that they will not any points. The aim is for the teacher to keep track of the students learning and see if they are revising. The aim for the students is to get, as many points possible so after everyone has finished the quiz there can be leader board with everyone's points.

STAKEHOLDERS

My stakeholder will be Mrs Kaur-Hender she is the economics teacher in the school and she will need this program to track the students learning, their progress after each test they take and how motivated each student is after they do not get the results they want and work hard to get even better results. My other end-user will be the A level students since I need to know both perspectives. One for the teacher to see how she wants the test to be e.g. the answer to the questions, the questions themselves and how she could record the student's results. The students will help me analyse if the test is appropriate for them to see if they need a harder test or an easier test. This quiz will be helpful to both stakeholders because the teacher can see the score overall and see the class weaknesses and the student cans see where they are failing the most and put even more effort into those particular parts of the course to improve their score and knowledge. This is quiz beneficial for both stakeholders.

CONTROLS FOR THE SOLUTION.

The quiz will be easy to answer as it will only have limited options like "A"," B","C"," D"," E"," F" or "1","2","3","4","5","6" there won't be any complications, any student will be able to understand how to answer. They will not need any special type of skills to use this quiz, apart from using their knowledge to complete the test to their best of their ability.

TARGET AUDIENCE

The economics quiz is a revision test that will challenge students' knowledge, those who are doing A level or AS economics will be able to take this quiz or whoever is learning economics. The platform for this quiz can be on either a PC or a laptop. The quiz will be aimed for anyone that is over 15 as it is a quiz that is only designed to challenge A level economic students and for the teacher to see how well they are doing. The quiz will be challenging and complex for the students. It will be challenging to the extent that the students may know the answer but confuse them with another that seems to be right but confuses them.

Justification of how the problem can be solved by computational methods.

Thinking abstractly

My quiz will be abstracted, this is because the quiz will need to collect points so it will need a point system and many more useful features to make this quiz fun and efficient for both stakeholders.

Features of abstraction within the economics quiz:

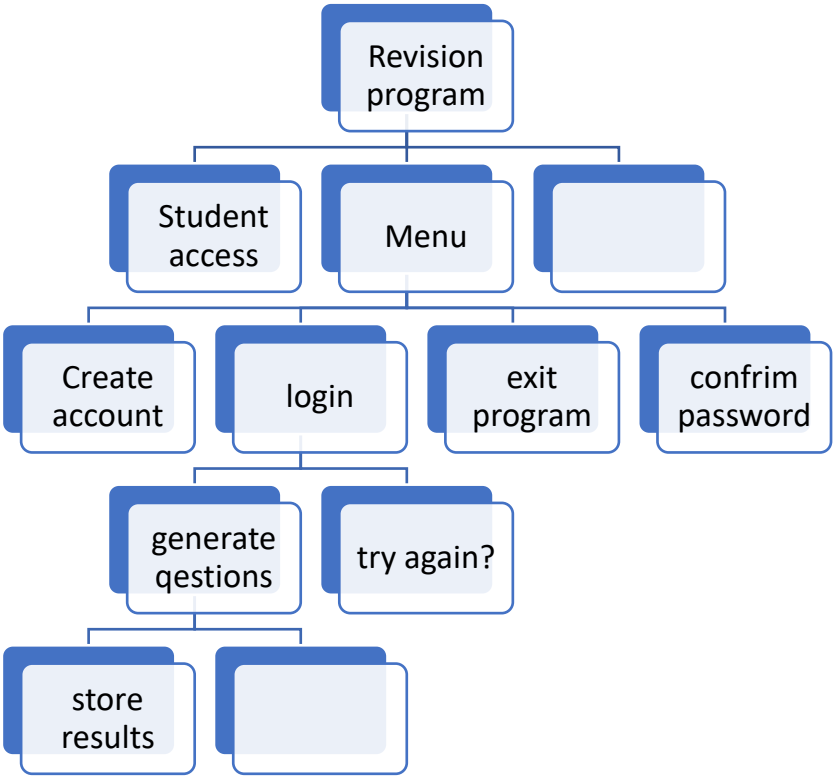
- Leader board, this feature will show all the students that participated in the quiz and then put them in a leader board the student with the most amount of points will be at the top and them will go down with the other students (highest to lowest). This will only appear when the user enters the option in the menu and the leader board will be created in an external file.
- The animation in the quiz will be heavily simplified, there are other quizzes that have a lot of graphic with colours and banners but my quiz will be much formal, due to the fact that the stakeholders (Economics A level class) taking the quiz will be mature and will not really care about animations but will rather focus on the quiz to get the highest mark possible to show the teacher that he/she have learned a lot and compete with their classmates.
- The questions will also be simplified since they will be multiple choice they will only have a set of answers to choose from in this case there would not be long questions with long answers but much simplified.

THINKING AHEAD

- Thinking ahead is the process of identifying inputs and outputs to a solution before you have tackled the problem, you can then think of potential problems and then you will have time to think of how the problem could be solved.
- Therefore, in this quiz I will need to work put what the user will input in their quiz and then what to output depending on what they inputted.
- Database that saves all the students details, the program will ask every student to create a profile and then they will be saved in a database with every single bit of information they saved.
- Point system, if the quiz doesn't have this then the quiz will not be as reliable as the other ones, the points will add up each time the user answer a question which is correct. This then reduces the risk of conflict between the students in who got the most points. If the student gets the question wrong, then they will gain 0 points and continue to the next one.

- The leader board, this quiz will save all the scores of every student once they have finished the quiz. Their score will be put automatically updated in the database in order from highest to lowest or viceversa.

THINKING PROCEDURALLY AND DECOMPOSITION



Decomposition is the breaking down of a big problem being broken down into smaller sub tasks so they can be tackled easier and then be inputted into one whole program. This will help me because it will be neat and efficient. It will be easier to fix errors if they are any because they are in sub routines. This simplifies the program and is much more understandable it will help.

- To decompose the program, I used a top down diagram which breaks down individual aspects of the program into separate tasks, which makes it easier to work on.
- The leaf notes would be the tasks that would need to be in the program, so it works for every task.

THINKING LOGICALLY

- Thinking logically outlines any decisions points and then whether it results in branching or looping. Branching looks at a decision and then that decision could result in a number of different outcomes, looping redoes that section of code until a decision is met. Decision within a code are likely to affect the flow of the program and what happens. This will improve the program because it will have to go step by step at the beginning for example: they create an account in order to play. Then the user will sign in and then the main menu will be shown to them.
- When the menu is shown it will be up to the user what he wants to do from then. The user can either start the game or change his details. This will cause the program to follow different instructions. However, the program will not be able to create a score board if no student has participated.

- When the user inputs the answer there is a decision point for the program to check if the answer they inputted for that particular question is right or wrong this will determine if they get a point for answering that question or if they gain none and go to the next one.

THINKING CONCURRENTLY

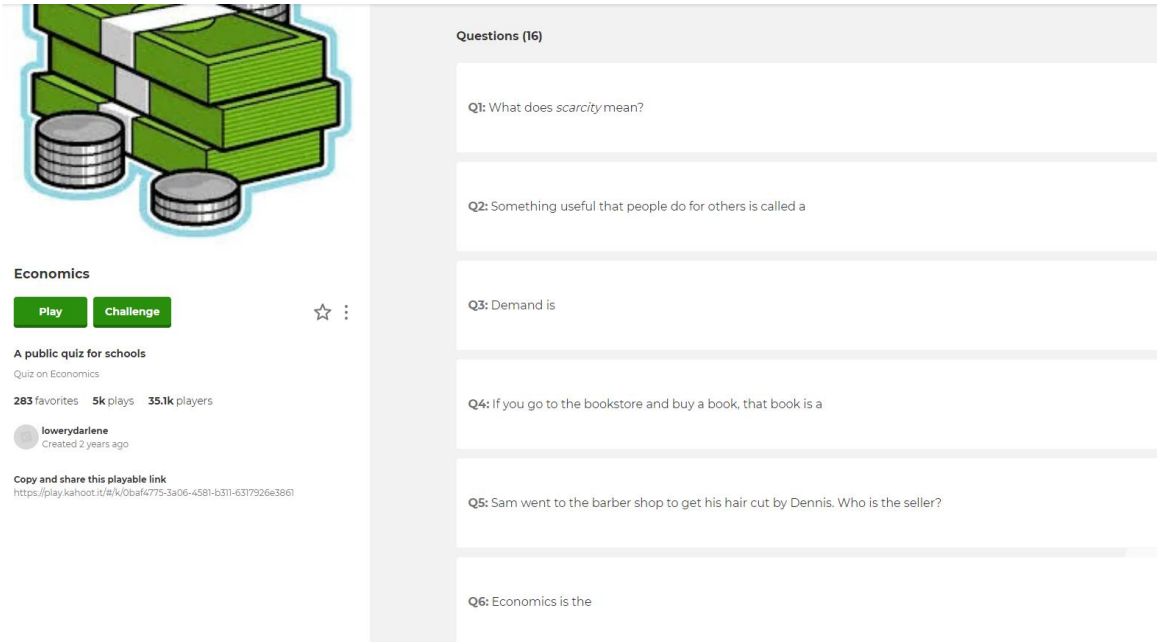
- Thinking concurrently is the process of working on different task that can be tackled at the same time.
- While the student is creating their profile, it will have to keep asking them for them details as well as saving the previous details inputted into the database.

- At the end of the game, the student will want to see the score board and it will mean that the program will have to create an external file as well as compare all of the student's result to put them in order of highest to lowest. It will need to compare all results and sort them out by descending order. The benefit is that it will be done in a few seconds and will sort the list for the class to see.

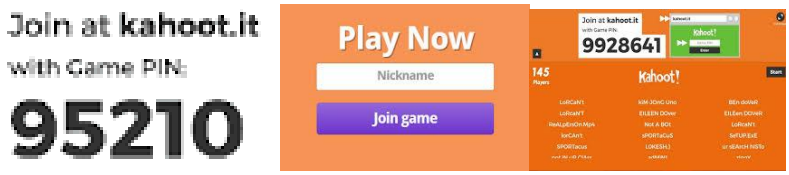
RESEARCH

LOOKING AT OTHER QUIZZES : KAHOOT

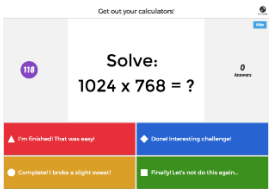
Kahoot is a platform where people can make quizzes of any kind, this platform was developed by the team called Johan Brand, Jamie Brooker and Morten Versvik. The research was done by Alf Inge Wang and his colleagues in the Norwegian University of Science and Technology. The aim for this platform is to administer quizzes, discussions or surveys it acts in real time and the whole class can take part of the test. As the person answers the questions right they accumulate a streak and increase their score as their streak gets higher.



HOW TO ACCESS: At first, the participants will need to enter a pin that then will redirect then to a screen where they can enter their name and access the class where the test is. There are examples of what it looks like, it has a pin to join and the next steps. In my quiz the students will need to create an account and each of them will have their own personal details and will take the same test as everyone else with the same multiple questions



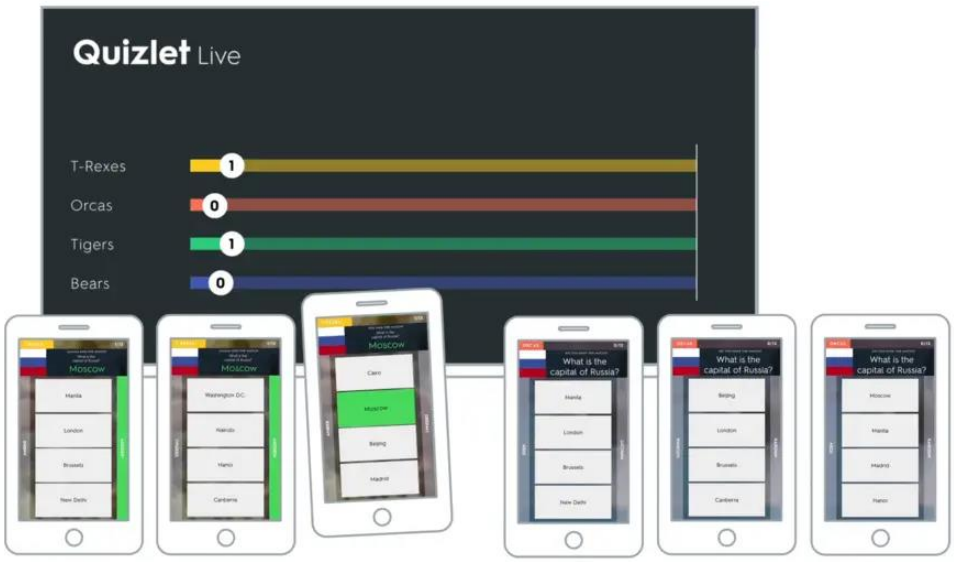
CONTROLS: Once you are, inside the game and the game starts, you will have options to choose from in the main screen and the player has to choose the right answer. The mouse is the control since the user uses their knowledge to choose any of the answers. The actual control for my quiz will be letters or numbers from the keyboard.



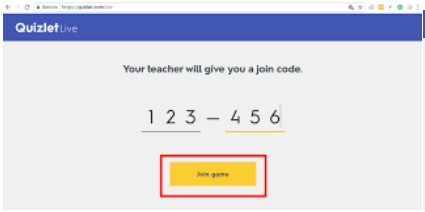
From, the picture it is visible that the user has 4 options to choose from, sometimes it can less, and in some circumstances more than one answer can be correct. This is one of the quizzes that is used in a lot of schools, it helps students fill gaps in their knowledge and pay attention. There are more quizzes that teachers use and they really help.

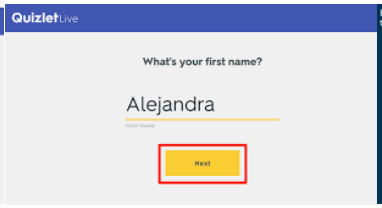
LOOKING AT ANOTHER QUIZ: QUIZZLET

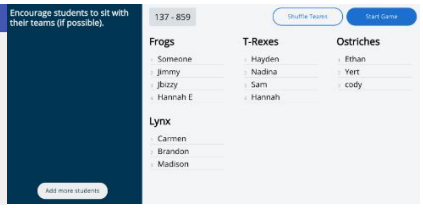
Quizlet is another platform that students and teacher use, the difference between this test is that the users have the questions and answers in front of them whereas in kahoot they have the questions and answers in a different screen and they get to click one of the answer in their own device. Quizlet was invented by Andrew Sutherland. He created a test for himself for a French exam, once he finished that and got good results his friends quickly asked for it and shared it, that's how it started to grow. The goal is to help students and teachers practise and master whatever they are learning



HOW TO ACESS: Each student will have to input a code that will then allow them to enter the quiz and join the screen, once all the students join. It is really similar to Kahoot, they both need a code. Here are some pictures to show how it will look like. Once they input that code, they will have to enter a valid name in order for the teacher to know who is who. Finally, when all students join, they will be distributed equally into different groups.







CONTROLS: Once they are all into their groups, they will have to join them so they all can compare answers and pick the right one. They can tap or click the answer, if they get an answer wrong then they will start from 0 and start answering the same questions again but in a different order.

INTERVIEW PLAN

I will now need to ask the students and the teachers about the quiz, from the research I have an idea of how the quizzes can be set up, now I need information from students to develop my ideas and take them further, for this I will create a questionnaire that will help me understand the preferences.

Main Points

- **What makes a quiz successful?**
- **How useful can a quiz be?**

Topic development

What makes a quiz successful?

1. Does the quiz need to be animated?
2. Should my quiz have sound effects?
3. Would you prefer to try the quiz again if you did not get the result you wanted?
4. Should there be a grading system?
5. Would you like to see if the student tries again to get higher marks?
6. Should I add a timer?

How useful can a quiz be?

1. Do you feel more confident after testing yourself?
2. Do you feel like you can challenge yourself enough?
3. From your experience do quizzes work with students?

INTERVIEW REVIEW – SECTION 1

Most of the questions were directed to both stakeholders because they both have knowledge about quizzes. Following up from this section I have concluded that the quiz shouldn't be animated at all, and not put sounds on the game during the game because as Mrs. Kaur-Hender said it would distract the students, so I have decided that I

will add sound effect at the beginning and end of the test where there will be crucial decision points. Like starting or ending the game.

INTERVIEW REVIEW – SECTION 2

From this interview with one of the students, I decided to focus in Theme 2 instead of all 4 themes. I decided this because I realised that if I try to implement all themes it will stress the user. In this interview the student also has said that they are not interested in animation as it could distract the user from the actual quiz.

INTERVIEW DATA REVIEW

ANIMATION: The quiz does not need to be animated like other quizzes, this is more appealing for my target audience because they are mature and will not care if there is such thing. There will be a normal screen with a solid background colour or a wallpaper that is popular for the students that will take this quiz but will not be animated. However, I do realise this will be too boring so I will try and include memes that will make them smile and then continue with their quiz.

SOUND EFFECTS: I have decided that they should be some for example when the user creates a profile and start the quiz because they are kind of big points to the program because it triggers another option. So, I will insert a sound effect in every option that has a big impact to the program, I will try and make sure the quiz isn't as boring as every student thinks it is but make it stand out.

MENU: There will need to be a clear menu that is simple and that the user will be able to follow with no problem at all. It shouldn't really be a problem because my target audience is for 15+ students.

BACKGROUND: The background does not have to be all colourful for my target audience but it also doesn't need to be boring with a solid colour. I will try and adapt a wallpaper if possible and try to make it so simple and boring.

COMPETITIVE: Student are always competing to demonstrate who is the best in the class, this quiz will be a challenge to see who can get the top result. The tension of not knowing what anyone got until everyone finishes the quiz. Once they get the results some will want to try again and get better results that the one who got the highest the first time.

PROPOSED SOLUTION

START-UP: The user will be asked to create a profile to participate in this quiz. The program will ask personal questions that the user can answer with no difficulty.

MENU: This is the second thing the user will see. The menu will be showed once the user has created the profile. There will be different menus for different parts of the program. For example, at first, they will have to either login, sign up or exit the program. If they login or sign up there will be following menus, like start the quiz or to try it again. They could want to change their details so there are different menus. However, if a user does not input any valid choice then the program will ask the user to re-enter an option until it is valid.

QUESTIONS: The questions will be challenging, they will be multiple choice so they will have to enter a letter or number. However, like in the menu if the user accidently or purposefully presses the wrong key they will have to re-enter an option until a valid one is chosen.

STUDENT PROGRESS: The progress of a student will be saved once each finishes their session and save it to a CSV file, this will show them their marks. However, it will not show them the questions they got wrong because they need to make sure they don't guess and know what the answer is for each question. It will also show a percentage for what they got. If they want to retake the test another record will be made for them and not overwrite the original result, so they know y how much they have improved.

GUI: This program will be presented in python using the Tkinter library in order to make the program more interactive for the users so it will not be dull. This function will not be distracting the student, as it will not be as lively either.

END OF TEST: At the end of the test the student will see the menu again, in this menu he will have options to see their name in the leader board and see how far up or down they are in terms of the class.

FEATURES EXPLAINED

FEATURES	Justification/limitation
Clear Menu	Jefferson wanted a menu that had clear instructions and were easier to follow; this tells me that the menu should be straightforward.
Sound Effect	In the interview the student did not want sound effects in everything, just in the important ones e.g. Start and end.
Graphics	As Jefferson and Emily said, the game doesn't have to be animated because older students do not look for animations, they will only be a distraction. However, I also won't be able to animate as my programming skills are limited.
Amount of questions	During the interview Mrs. Kaur-Hender mentioned that long and simple questions are not suitable for their students because it will not be as effective as short and hard questions. These questions will be more challenging. The amount of questions will be large but will only focus in one Theme rather than in all of them.
Background	I will try and change the background, so it is not dull and boring since a quiz that is very boring will put off students.

Leader Board	At the end of the test it will help students how they have done, this will reduce tension and anxiety for students.
Time Limit	From the interview, the idea to add a timer has been discarded since it will add unnecessary pressure to the students. However, I will add a timer but not for the questions

SOFTWARE AND HARDWARE REQUIRIMENTS

REQUIREMENTS	JUSTIFICATION
HARDWARE	
Monitor	User needs to see the quiz and the questions in order to answer them.
Keyboard	Required to select an answer and enter it.
Mouse	Allows user to navigate and access the program and put it in full screen if they want to.
Memory: 1GB + RAM	
33 MHz	Minimum requirement of processor to run the software.
5 Gb of Hard drive space	This word document will be heavy weighted after it has been finished, the program will also be heavy weighted, it will also require to have external csv files.
Speakers	To play the sound files.
SOFTWARE	
Operating system: Windows 10 Python Microsoft Excel	This will be essential as I will need an OS that will work with python and you might not be able to download python in other platforms, in other platform the user might have to download more software in order to get python installed which will consume secondary storage.

Python libraries	I will need to access libraries in python so I can then import the codes I need and create a login or a database for example, there will be different functions and methods that will be used.
DB Browser for Sqlite	This is needed to open the database and look at all the entries that have been added or updated.
NotePad or Brackets for the HTML	This is required because the revision for the quiz where all the information's and links for the videos will be available
Atom	This is used for the website aspect of the program.

USER REQUIREMENTS

- Clear menus that users will be able to follow with no difficulty.
- The user will be able to use the program even if there isn't any internet.
- Users should be able to change their details if they wish to.
- The program will create a file that will save details.
- The program should validate before going to next question or variable (presence check and passwords)
- If data is invalid (No data inputted or Boundary error) it will repeat until the program accepts the data.
- If the credentials that the program asked for are valid, they will be stored inside a database.
- The user will be able to start the quiz, change personal details, access the revision for the quiz and exit the quiz.
- They user will have links to videos to help them understand the topics.
- The economics teacher will be allowed to have overall control of the database and see the results.
- Users will create their own login and passwords.
- Both students and admins will be able to log in using their usernames and passwords by checking the databases.

SUCCESS CRITERIA FOR PROTOTYPE 1

- The program will allow the user to log in or sign up.
- Program will be understandable.
- The program will use validation to make sure everything is filled out
- Program will save the details into their variables and insert hem into a file.
- The program will allow to overwrite in their file to correct their details.
- Th program will validate the email.
- The program will show menus and it will be responsive.
- The program will have a point system to count all the points when the user gets the points right.

- The user will be allowed to display their results and their details if they wish.
- A leader board will be created automatically when more than 1 student takes the quiz.
- Check password length.

There will be more success criterion in the further prototypes in order to ensure the efficiency of the program and in order to meet the criteria that the user wants.

APPROACH TO THESE CRITERIAS

For this success criteria and user requirements, most of them are complex like validating an email and traversing a file in order to find unique usernames and passwords in order to log in into their specific login as a student or a teacher. There will be a lot of research done in order to accomplish this and do this successfully. The decomposition and abstraction will help tackle this much quicker and will make it easier to understand what I'm programming.

LIMITATIONS AND CONSTRAINTS TO REQUIREMENTS

I will be working in school grounds, as a student I don't have access to all the software I will be needing in order to create the program. I am using python which is used in school grounds but not all the libraries that I will be using are installed and will take a long process for the technician to download it, this is also a problem because when downloaded it will only stay in one monitor, if anyone else takes that seat I will not be able to use the library. This will cause problems so I will need to bring my own laptop into school

I will use all the time efficiently, however validating the email by creating a html site will consume a large proportion of my time. Also, if I do the website it will need to look professional to engage the users even more, but it takes a lot of time, if this happens my program will end up being incomplete. Also, to get a website I will need to pay an amount of money (for the domain name), an automated email to always send an email using python.

Another limitation is security of the user's email, as the website will not be fully secured as it only verifies the email, this doesn't stop others from intercepting, take the email and send them various phishing emails.

All the details that will be saved in the monitor are available to anyone who has access to the monitor, this means that the files are exposed to anyone, this causes security problems.

PROTOTYPE 1

USER INTERFACE

REVISION QUIZ
Question: Do you have an account? 1. Yes, I do 2. No, I do not

```
Enter command…
```

As soon as the user runs the program, this will be the first question that will appear to them. They will need to pick either of those choices, if not then the program will keep on repeating the question. The following 2 scenarios will occur.

```
Enter command… 1


Username:

Password:
```

The program will ask for their login details. This will only be true for the admin as no one else will have an account. If the admin enters one of the credentials wrong then it will display the menu again.

```
Enter command… 2.
```

When the user enters that command, the program will proceed to ask them if they want to sign up, the next box shows this. This is another crucial point, the user will need to answer otherwise it will not continue.

```
Question: Do you want to sign up?


    1. Yes, I do
    2. No, I do not


Enter command…
```

This is the first question that the user will be asked and is given 2 options to choose from so the user can continue with the program. This is one of the crucial questions; without an answer the program will not be able to continue. If the user inputs another option that is not required there will be a message outputted:

```
Enter command…3

The command you entered is not recognised therefore I can accept your request.
```

```
Enter command…
```

The program will only go on if the user inputs the given options. If they input something else like an option K it will output the message the user can only type Y or N or the option 1 and 2.

```
Enter command…1


What is your name?

>

What is your surname?

>

What is your Date of Birth?

>

Please enter an email:

>

Please create a username:

>

Please create a password:

>
```

This will ask the user for the essential information that the program needs so the account can be created. This information is needed because those are requirements needed to sign up.

```
Enter command…2


I am sorry but you need to sign up in order to use the program
```

If the user enters command 2 then a message will come up to tell the user that without signing up it is not possible to access the quiz.

Once the user has logged in or signed up to use the program, they will be shown the main menu.

1. Start the quiz.
2. Change personal details
3. Access revision for quiz
4. Exit

Enter command…

This is the Main Menu, each option leads to a different path.

Enter command… 1

What curves are mainly used for Microeconomics?

Etc..

The quiz will start asking economics-based questions. I have decided to focus on one topic instead of the whole microeconomics and macroeconomics aspect, I done this because I realised there is too much to cover and I will not have enough time to finish.

Enter command… 2

Name: …

Surname: …

DOB: …

Email: ….

Username: …

Password: …

The user will have a chance to change their details if they wish. Except their name, this is because when they will be creating their profile, they won't know they have this option and they might troll with this.

Enter command…3

Opens HTML Window

I am currently working in setting up a HTML site. This will only be opened if the user wishes to. The HTML will only have information regarding the test, they will be allowed until the quiz starts. Once the quiz starts, the revision will close.

Enter command…4

Do you want to see your results? Y/N

Exit

DESIGN

1) **Menus** – The software should be able to direct the user to whatever option they have. The menus will have different options, the user will be able to see them clearly and won't get confused, when they do this, it will output and do the intended purposes.

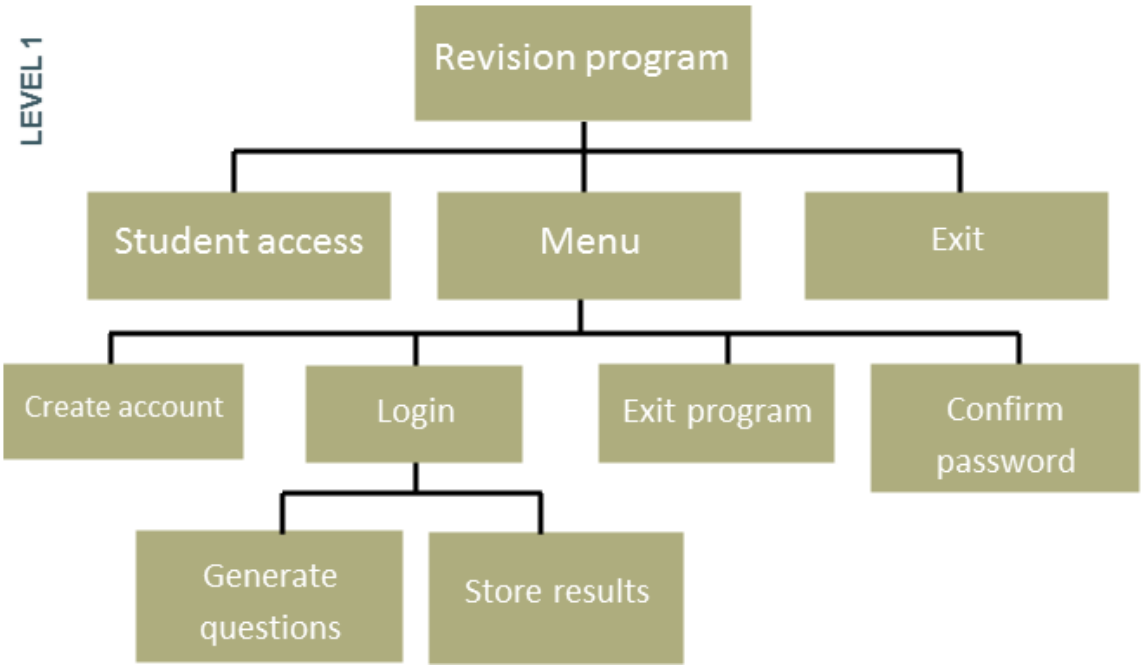
2) **Admin Login –** A basic system that will take two inputs from the user, a login and a password. The teacher will be the only one who has access to the databases to see who has signed up and if they have started the quiz or not. Both inputs must be right otherwise it will output a message error and ask again or go back to the creating a profile. In this database the teacher will be allowed to see who is in the database.

3) **Validation –** The users will have to sign up in order to use the program, for this the program will ask them personal details. The program will make sure that every file is inputted. They also must put valid usernames. (No spaces between the usernames or passwords.)

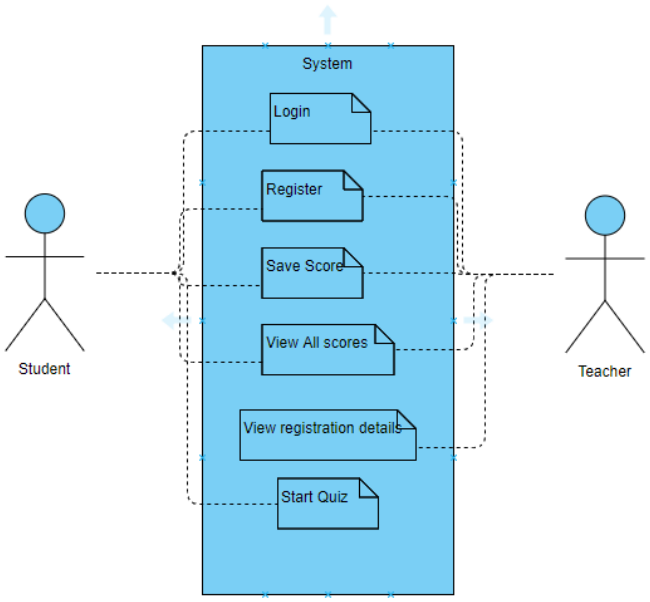
4) **Graphical User Interface –** The user will be able to navigate and use the program efficiently with the GUI.

TOP DOWN DIAGRAM

Breaking the program into smaller sub-tasks is essential in order to make this problem easier to understand and complete. I have all the requirements in place, and I am ready to start developing the program. I will be starting with prototype 1.

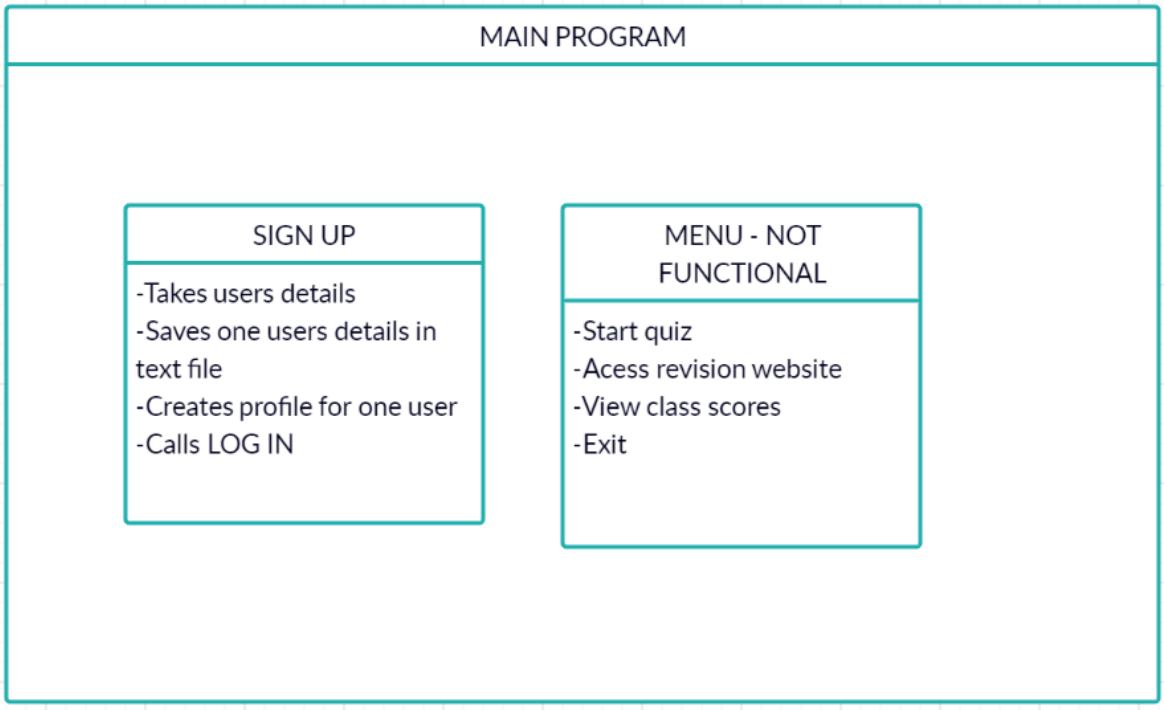


USER CASE DIAGRAM



The student will be able to take the quiz, their scores will be saved into another file that only the admin will see. The teacher will have its own login that allows them to enter the admin area.

DIAGRAM FOR PROTOTYPE 1



This is the modular diagram for prototype 1, the reason for it not being so developed is because for this prototype I want it to be able to create a profile, the program will take in details that are essential to create a profile as in name, surname, DOB, username and password. This will be saved in a txt file which will always be recorded, once the profile is created it will show the menu straight away showing the user the options they have. However, they will not be able to pick one of the options because the options in the menu don't have any command to them. This is a very simple version of the program. For prototype 2 I want to make the menu functional, allow the user to log in and not having to create a profile each time the program runs. I also want to create a database that can save multiple details and have it saved more than one student at a time.

BLACK BOX PROTOTYPE 1 TEST PLAN

ENTRY	EXPECTED OUTCOME
User opens program	Question asking the user if they want to create an account.
Check if the user enters the right option	Validates the users input and continues depending on the input.
Creating the profile	The program will ask for the users details and save them in a external file
Presence check	The program will iterate until the user has inputted something.
Check password is >8 character.	The program will keep asking until the password is between the correct length
Displaying the menu	Show the user the different options inside the menu

WHITE BOX PROTOTYPE 1 TEST PLAN

Test	Expected Response from program
Process of program when run	The function main() will be call, that function calls other functions inside the program.

The process of the program when a user enters an input	The program uses IF and WHILE statements to redirect the next outcome of the program.
The process that happens when the user inputs a username or password.	The password will be validates using a range check, once this is done it will be saved in a txt file.
The process when an option from the menu is selected.	The menu will call other functions in order to output what the user has chosen.

PSEUDOCODE AND FLOWCHARTS

Pseudocode will help me start the program and have a clear idea of what I want the program to achieve and perform. This is also simpler than programming because standard English is used so anyone can understand what it does. This also is part of decomposition because it breaks my program in different parts and write algorithms for them.

```
IMPORT CSV
IMPORT OS

DEF validate (string, chars):
      return TRUE in [c in strings for c in chars]

DEF user_profile():
      Sign_up: “Do you want to create an account? Y/N: ”
      VALIDATE Sign_up
      WHILE Sign_up == N
            IF Sign_up == N THEN
            OUTPUT “You have to create an account to participate in the quiz. ”
            Sign_up: “Do you want to create an account? Y/N: ”

      IF Sign_up == Y THEN
            Name: “Enter your First Name: ”
            VALIDATE Name

            Surname: “Enter your surname: ”
            VALIDATE Surname

            DoB: “Enter your Date of Birth: ”
            VALIDATE Dob

            Email: “Please enter a valid email: ”
            VALIDATE Surname

            Username: “Enter your surname: ”
            VALIDATE Username
```

```
            Password: “Enter your surname: ”
            OUYPUT “Password should be longer than 8 characters but less than 16”
            VALIDATE Password

            OPEN CSV file
            RECORD Name
            RECORD Surname
            RECORD Dob
            RECORD Email
            RECORD Username
            RECORD Password
            CLOSE CSV file

DEF display_intro():
      title = “** Jhonny’s Revision Program **”
      OUTPUT (“*”*len(title))
      OUTPUT(title)
      OUTPUT(“*”*len(title))

DEF display_separator():
      OUTPUT (“-”*24)

DEF display_menu():

      NEW LINE
      ARRAY Menu: “[1. Start the Quiz, 2. Change Personal Details, 3. Access Revision for
quiz, 4. Exit]”
      OUTPUT Menu
DEF get_user_input():

      User_input: “Enter your choice”
      VALIDATE User_input
      RETURN User_input

DEF main()

      User_profile()
      display_intro()
      display_menu()
      display_separator()

      Option = get_user_input()
      WHILE Option != 4 THEN
            Option = get_user_input()
      OUTPUT “Exit the quiz”

main()
```

VARIABLES/ STRUCTURES TABLE

Variable/Structure Name	Purpose	Data Type	Validation
Validate()	This is used to validate the menu further on.	Subroutine	This uses validation for further
User_profile()	This is responsible for making sure that the user answers all the questions if they want to create a profile.	Subroutine	It uses presence check and makes sure the user does input a value or string.
Display_intro()	This will be used to organise the program and make it look better.	Subroutine	
Display_menu()	This will be to output the menu	Subroutine	Make sure the subroutine is called at the right time.
Display_separator()	This organised the program better making it neater and easier to understand	Subroutine	
Main()	This is the subroutine that made sure all the previous subroutines were called in order and was organised.	Subroutine	

DEVELOPMENT

```
import csv
import os

def validate(strings, chars):
    return True in [c in strings for c in chars]

def user_profile():

    sign_up = " "
    while sign_up!= "Y" and sign_up!= "N":
        sign_up = input("Do you want to create an account? Y/N: ").upper()
        while sign_up == "N":
            if sign_up == "N":
                print("You have to create an account to participate in the quiz.")
                sign_up = input("Do you want to create an account? Y/N: ").upper()
```

This is simple and does not have any GUI, it doesn't add complexity to the program, this only asked if they wanted to create an account instead of letting them sign in if they were the administrator. I will implement this in the next prototypes.

```
if sign_up == "Y":

    name = ""
    while name == "":
        name = str(input("Enter your first name: "))

    surname = ""
    while surname == "":
        surname = input("Enter your surname: ")

    DoB = ""
    print("Please enter your DOB like this: 09 10 2003 with no spaces")
    while DoB.isnumeric() == False:
        DoB = input("What is your Date of Birth? ")

    Email = ""
    print("Please enter a valid email.")
    while Email == "":
        Email = input("Enter your email: ")

    Username = ""
    while Username == "":
        Username = input("Create a username: ")


    Password = ""
    print("Password should be longer than 8 characters but less than 16")
    while len(Password) <8 or len(Password)>16:
        Password = input("Create a password: ")

        f = open("userinfo.txt","w")
        f.write("User information"+"\n")
        f.write(str(name)+"\n")
        f.write(str(surname)+"\n")
        f.write(str(DoB)+"\n")
        f.write(str(Email)+"\n")
        f.write(str(Username)+"\n")
        f.write(str(Password)+"\n")
        f.close()
    print("Record has been written to file")
```

This is the code to take the details of the user when they create an account, it validates to see if they have entered something into the variables.

In the DOB we have a special function that checks that the DOB only has whole numbers instead of a string to make sure the DOB is right and be inputted in a file after.

This shows how the file is created and what will be saved in them. The user will be allowed to access that file later.

```
def display_intro():
    title = "** Jhonny's Revision Program**"
    print("*" * len(title))
    print(title)
    print("*" * len(title))

def display_menu():
    print("\n")
    menu_list = ["1.Start the quiz.","2.Change Personal Details.","3. Acess Revision for quiz.","4.E
    print(menu_list)
    valid = False
    while not valid:
        choice = input("Enter an option: ")
        if not validate(choice,"1234"):
            print("Invalid option")
    print("\n")

def admin_menu():
    print("Hello")
def display_separator():
    print("-" * 24)

def get_user_input():
    user_input = int(input("Enter your choice: "))
    while user_input > 3 or user_input <=0:
        print("Invalid menu option.")
        user_input = int(input("Enter your choice: "))
    else:
        return user_input


def main():
    log_in()
    display_intro()
    display_menu()
    display_separator()

    option = get_user_input()
    total = 0
    while option != 4:
```

There are different subroutines, they help organise the program to make it readable for the user, the menu is also displayed.

```
RESTART: E:\A level computing\Programming Project\Actual Program for cs project.py
Do you want to create an account? Y/N: n
You have to create an account to participate in the quiz.
Do you want to create an account? Y/N:
Do you want to create an account? Y/N:
Do you want to create an account? Y/N: y
Enter your first name:
Enter your first name: jho
Enter your surname:
Enter your surname: g
Please enter your DOB like this: 09 10 2003 with no spaces
What is your Date of Birth?
What is your Date of Birth? f
What is your Date of Birth? 04
Please enter a valid email.
Enter your email: efg
Create a username: rgtfv
Password should be longer than 8 characters but less than 16
Create a password: ytgfewdsbgr
Record has been written to file
**************************
** Jhonny's Revision Program**
**************************


['1.Start the quiz.', '2.Change Personal Details.', '3. Acess Revision for quiz.', '4.Exit']
Enter an option: 1
Enter an option: 2
Enter an option: 3
Enter an option: 4
Enter an option:
Invalid option
Enter an option: 5
Invalid option
Enter an option:
```

The main() functions calls all other subroutines in the order displayed, as you can see it does validate everything, I will need to further validate the email addresses to make sure they are valid and also validate the length of the DOB because the program will accept any value if it's a whole number.

TESTING

Testing or Validation is to check if the program works. The user is going to be in a scenario where they need to pick an option, it can vary from a Yes or a No, or to just simple numbers (Menus). If the user inputs anything not recognized by the program it will output an error or a message telling the user what they have inputted is not right in the program. Depending on each case, the program might repeat itself or it won't.

I will be able to check this using an IF statement so if they input something that is not one of the options, a display message will come up and tell them that they weren't supposed to do that. Instead, they had to type one single number or letter that will then carry on with the program. If they don't input the options given by the question the program won't accept it because it wasn't in any of the options.

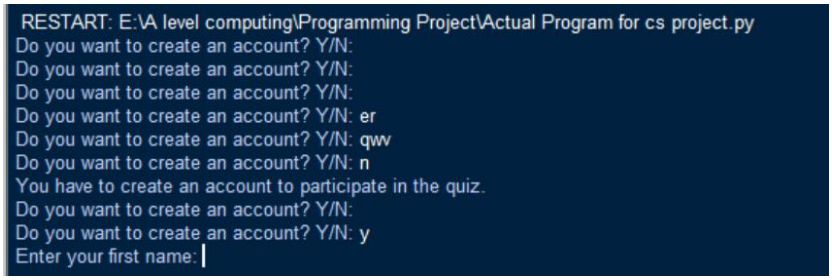
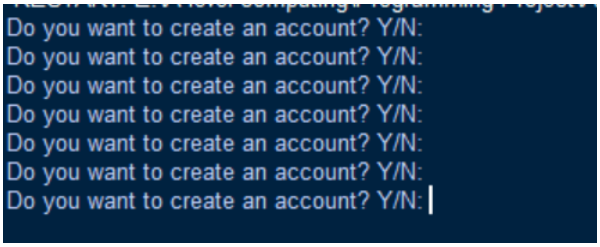
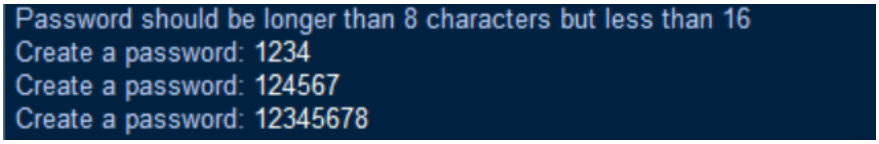
If they press enter it will also not be valid because they are required to write something and not leave anything blank especially with the personal information. Presence check needs to be tested to check if the user has just

ignored the question or pressed enter. If they do press Enter, the option will repeat itself until the user inputs something I will be able to do this with a loop which can be a WHILE or a FOR loop.

The user will be asked questions at the beginning, e.g. “Do you have an existing account? ” etc.. the user will obviously need to provide an answer if they want the program to continue, especially if they want to take the quiz. When this happens, the program will start to ask them personal questions that need to be answered. They need to answer these questions in order to make their profile.

BLACKBOX TESTING PROTOTYPE 1

Success Criteria	Achieved?	Justification	Screenshot
User opens program and asks if the user wants to create an account.	Yes	This needs to be asked in order to continue with the program.	

Checks if the user enters the right option	Yes,	This validates the user input; it will trigger the continuation of the program.	 The program validates invalid spaces, or invalid inputs, it only accepts Y or N.
Creating the profile	No	The details must be saved in order log in.	
Presence check	Yes	The program should not allow for invalid inputs.	 The program iterates until a valid value is entered.
Check password length	Yes	If the password is too short then it will be too weak, therefore range validation needs to take place to avoid this.	 The variable is validated until the password has a range of 8 characters, if its anything below that it will not accept it.
Displaying the menu	Yes	The user needs to have the option to take the quiz, once they log in.	 The user is able to see the different options they have.

WHITEBOX TESTING FOR PROTOTYPE 1

Test	Expected Response from program	Successful? /Evidence	Actions to be taken

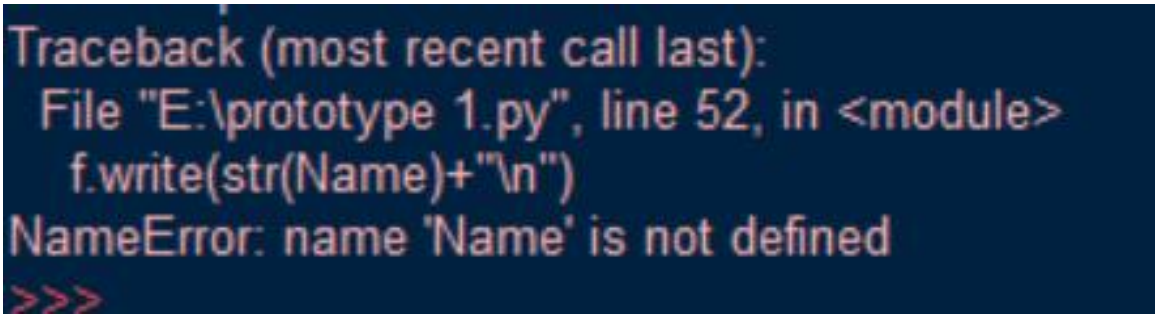
Process of program when run	The function main() will be call, that function calls other functions inside the program.	Yes
		`def main():`
		`  user_profile()`
		`  display_intro()`
		`  display_menu()`
		`  display_separator()`
		Once the program is run by the user it calls these functions in order.
			N/A

The process of the program when a user enters an input	The program uses IF and WHILE statements to redirect the next outcome of the program.	Yes
		`sign_up = " "`
		`while sign_up!= "Y" and sign_up!= "N":`
		`  sign_up = input("Do you want to create an account? Y/N: ").upper()`
		`  while sign_up == "N":`
		`  if sign_up == "N":`
		`  print("You have to create an account to participate in the quiz ")`
		`  sign_up = input("Do you want to create an account? Y/N: ").upper()`
			N/A

The process that happens when the user inputs a username or password.	The password will be validates using a range check, once this is done it will be saved in a txt file.	Yes
		`Username = ""`
		`while Username == "":`
		`  Username = input("Create a username: ")`
		`Password = ""`
		`print("Password should be longer than 8 characters but less than 16")`
		`while len(Password) <8 or len(Password)>16:`
		`  Password = input("Create a password: ")`
		The program checks the username and password to see if they are empty, if they are the program will ask until the user inputs a value. Additionally, the password has a range check in order to calidate the length.
			In prototype 2 I will try and implement a function that hides the password while typing it for security measures.

The process when an option from the menu is selected.	The menu will call other functions in order to output what the user has chosen.	No
		`user_input = int(input("Enter your choice: "))`
		`while user_input > 3 or user_input <=0:`
		`  print("Invalid menu option. ")`
		`  user_input = int(input("Enter your choice: "))`
		The menu is shown but its not responsive because there is no functions to be called.
			I will try and complete these options in prototype 2 in order to create a responsive menu.

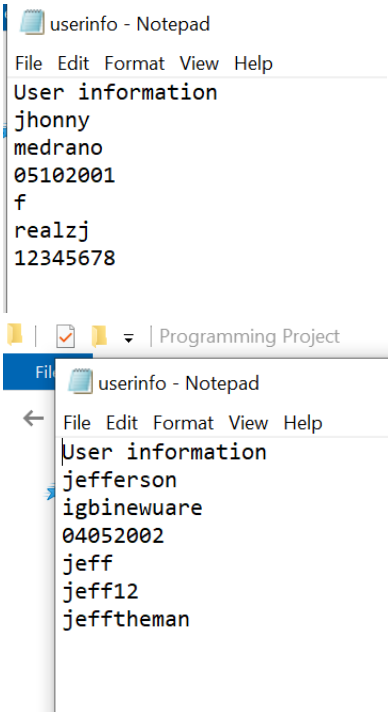
ERROS DURING DEVELOPMENT



As for the errors, there wasn't many, the only one was with writing the users details into the txt file, the variable name was FirstName but I put write Name into the txt, the program didn't have a variable called name which caused a syntax error. There was not another syntax error I came about. There was not a lot of development in this prototype due to the thinking and processing on how to start this program.

FIXING ERRORS IN PROTOTYPE 1

After fixing the syntax error, I had to put the correct variable names inside the write txt file in order to save the details. However, after fixing this I came across another problem which was not intended. First, when I tried saving multiple students in the txt file, the txt file couldn't save more than 2 students. This is because when the program is run again then the new details entered by that user will overwrite the previous ones deleting the previous profile.



This is what the txt file looked like after the program stored the details. Until this point the program was working fine, it allowed the user to create a profile, the program saved the users details. Until the next user created a profile.

This is the second user who created a profile, as the txt file shows, the previous user disappears, and it doesn't have 2 profiles.

I realised I couldn't fix this instantly because I would have needed to develop a csv file or a database in order to save more than one student. This will be fixed in prototype 2.

BETA TESTING

In order to improve the program further I asked a few to take the quiz and to give me feedback at the end. Of course, in this prototype I am aware that the program is not developed enough to judge the quiz and everything else. In this prototype the user will only be testing the log in and sign up system. This part is the most essential because it will be how the program starts.

After a few students tried out the program they told me that they created an account but there was no option to log in which was true, after the student created their account it went on to the menu straight away. I asked how I could improve this; they said that it would be better if it designed like a website, which allows them to create an account and then redirect them to a login window which lets them login. Some students also said that when they were creating their profile, they accidently typed a number in their name, but it still went through.

In prototype 2 I will fix errors that have been spotted by users by adding other functions that check if the variable contains invalid inputs as well as the presence check.

REAL WORLD UTILITY

There will be options that will be answered with letter or there will be command that will need to be accessed by the user, the user will have no problem to access them if they try to access it with a lower case or a upper case they will have no problems because I will use the .upper () and the .lower () this will make the users life much easier because they will have no problem accessing the files if they have used the wrong case.

If they make a mistake with one of the commands the options will be shown to the user so they don't make the same mistake the commands will appear to him reminding them the correct command they will be also be shown the files that are present have been created by them.

The user will not need to write a lot in the program they will be asked simple questions that will not require a lot of thinking so they will not get bored, they will be need to be asked songs and the audience for this is usually teenager because they are keener to listen to songs and have more knowledge of this.

EVALUATION

After creating my first protype, I came across some problems, I haven't been able to create a database yet like it was first intended so it doesn't quite fulfil the whole purpose as it is only saved in a txt file. This is only the first prototype, so I made sure that the main components are done, and it creates the profile and displays a menu, the menu is not fully responsive yet. It does validate the options and makes sure that only the options given are inputted or else it will output an error message. Another problem I came across was how I was going to start the quiz and count all the points since I was intending to import the questions from a csv file, but I haven't figured out how to output them 1 by one. I also haven't created the HTML file for the revision, this is only temporary, and I will try and achieve all these failed requirements in the upcoming prototypes. I also have not managed to validate the email or send anything to their actual email. My programming skills are limited, and I will have to continue researching in order to fulfil failed requirements.

I have failed to complete many of the criteria's I proposed myself to do. I plan to complete most of them in the second prototype leaving prototype 3 with commenting and maintaining the program because at the moment the program is not commented at all and its not easy to follow if another programmer was to go through the program.

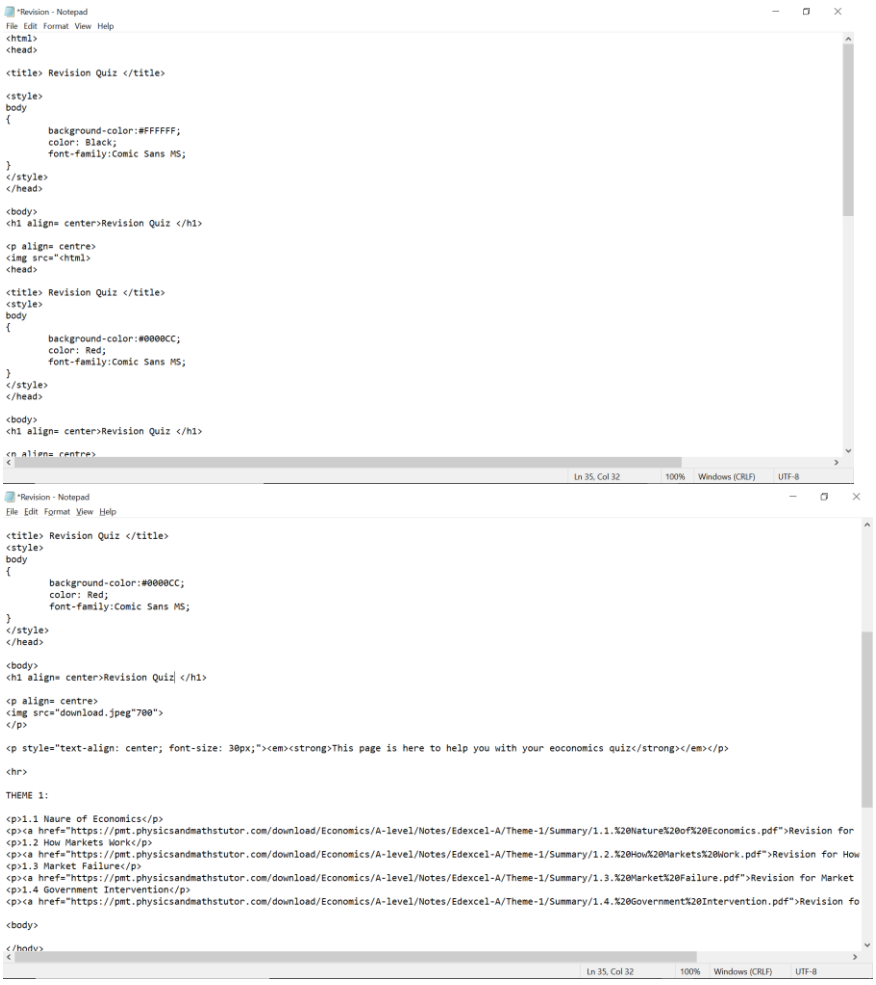
I must keep on researching on txt files because without that the profiles cannot be created. I will try and implement other txt files or change it to another type of database that can save multiple users at the same time no matter the amount of details. I will also be keeping touch with the stakeholders to talk about the program and what they think about it, as well as the students because they are the ones using the program the most, with the quiz.

After analysing beta testing, I have decided that I will create some kind of window that will allow a user to log in through there rather than on python user interface, this will take time since I currently don't know how to do that but the research will help me complete this and meet the students requirements as well as the teachers.

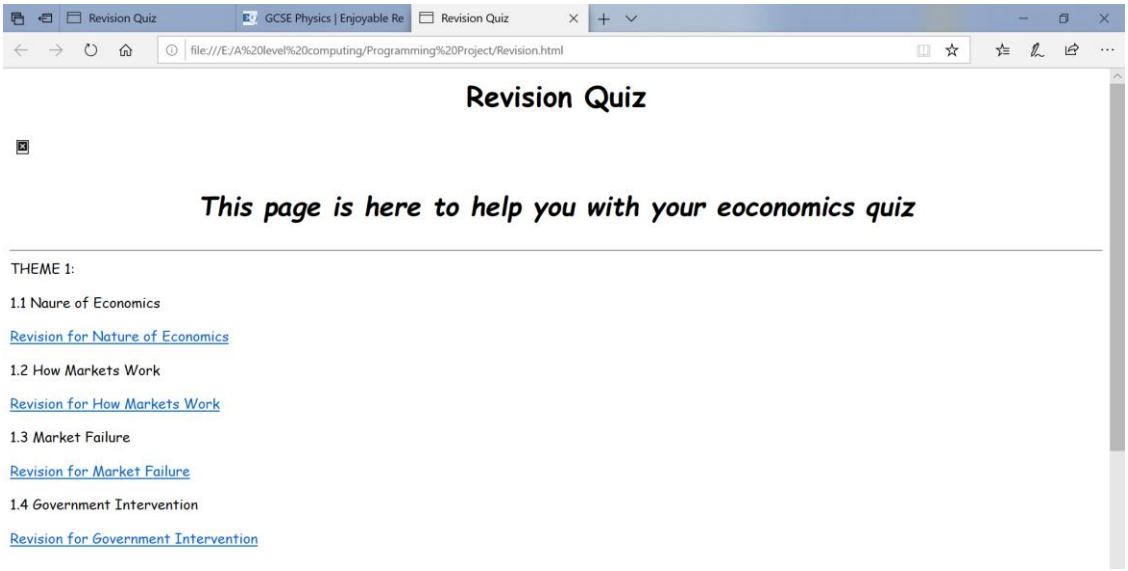
PROTOTYPE 2

RESEARCH FOR HTML

For my prototype 2 I started to do some research on how to build a revision page of my own, this led me to start using HTML to build a simple website that works. The following shows the development of the webpage.



```
*Revision - Notepad
File Edit Format View Help
<html>
<head>

<title> Revision Quiz </title>

<style>
body
{
        background-color:#FFFFFF;
        color: Black;
        font-family:Comic Sans MS;
}
</style>
</head>

<body>
<h1 align= center>Revision Quiz </h1>

<p align= centre>
<img src="<html>
<head>

<title> Revision Quiz </title>
<style>
body
{
        background-color:#0000CC;
        color: Red;
        font-family:Comic Sans MS;
}
</style>
</head>

<body>
<h1 align= center>Revision Quiz </h1>

<p align= centre>
                                                    Ln 35, Col 32   100%   Windows (CRLF)   UTF-8

*Revision - Notepad
File Edit Format View Help

<title> Revision Quiz </title>
<style>
body
{
        background-color:#0000CC;
        color: Red;
        font-family:Comic Sans MS;
}
</style>
</head>

<body>
<h1 align= center>Revision Quiz </h1>

<p align= centre>
<img src="download.jpeg"700">
</p>

<p style="text-align: center; font-size: 30px;"><em><strong>This page is here to help you with your eoconomics quiz</strong></em></p>

<hr>

THEME 1:

<p>1.1 Naure of Economics</p>
<p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.1.%20Nature%20of%20Economics.pdf">Revision for
<p>1.2 How Markets Work</p>
<p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.2.%20How%20Markets%20Work.pdf">Revision for How
<p>1.3 Market Failure</p>
<p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.3.%20Market%20Failure.pdf">Revision for Market
<p>1.4 Government Intervention</p>
<p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.4.%20Government%20Intervention.pdf">Revision fo

<body>

</body>
                                                    Ln 35, Col 32   100%   Windows (CRLF)   UTF-8
```

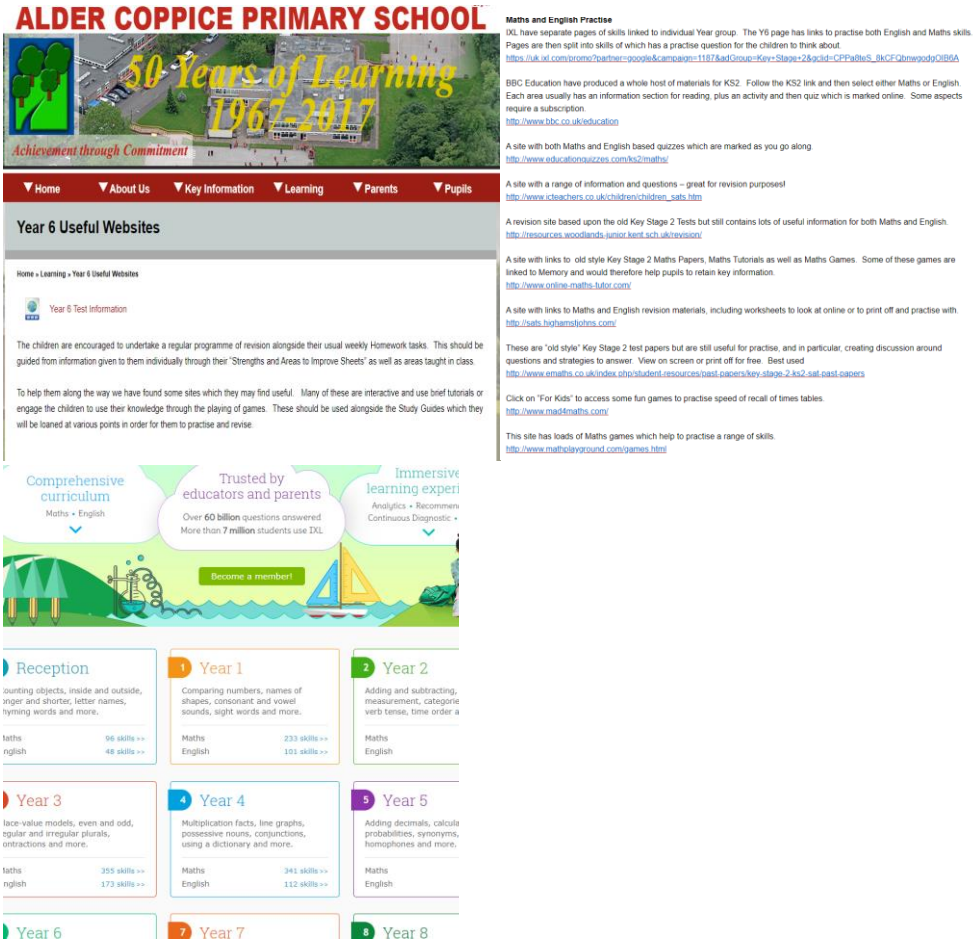


This is what I have done so far, it looks simple and plain. I plan to change the style and make sure everything is in a way it attracts the user to use the site instead of seeing it and be put off by how it looks. I also done my own research into looking how other revision pages look like.

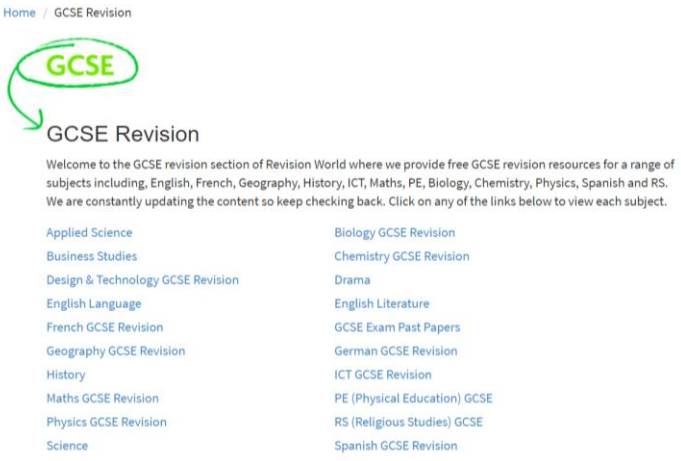
RESEARCH FOR OTHER REVISION PAGES

Primary Revision pages

I will be exploring and researching about different pages, this is for a year 6 school, it looks similar to my page. I assume the page is plain and empty because the parents that will be using this will only care about accesing the links that will redirect them to the right place. Those links lead to other websitrs that are more developed and contain a lo tmore information. For example the first link that is given to the them (https://uk.ixl.com/promo?partner=google&campaign=1187&adGroup=Key+Stage+2&gclid=CPPa8teS_8kCFQbnwgodgOIB6A) redirects them to this site that helps students from reception all the way up to Year 13, this shite contains more colours and even allows the user to create an account and become a member. However, to become a member the family will have to pay which can be an inconvinience,



The following pages is from a GCSE page, this gives me an idea of how my page should look like and helps me to understand that students are not interested in any type of fonts or backgrounds they rather just practice and go straight to the point otherwise they will lose time. This page also has links but they are organised in the middle, it is also organised in alphabetical which gave me an idea of how to organise the information later on when the page is finalised, in my original HTML I want to add pictures because in Economics there is a lot of graphs so I am sure that the students will appreciate if I added pictures to help them



I tried looking for revision pages that is only dedicated to sixth formers but I was unable to gain access to it because I needed to pay, it is called uplearn and has many courses for different subjects, it has a video and it shows that each topic has their own videos explaining step by step on what to do and how to answer exam questions.

DESIGN

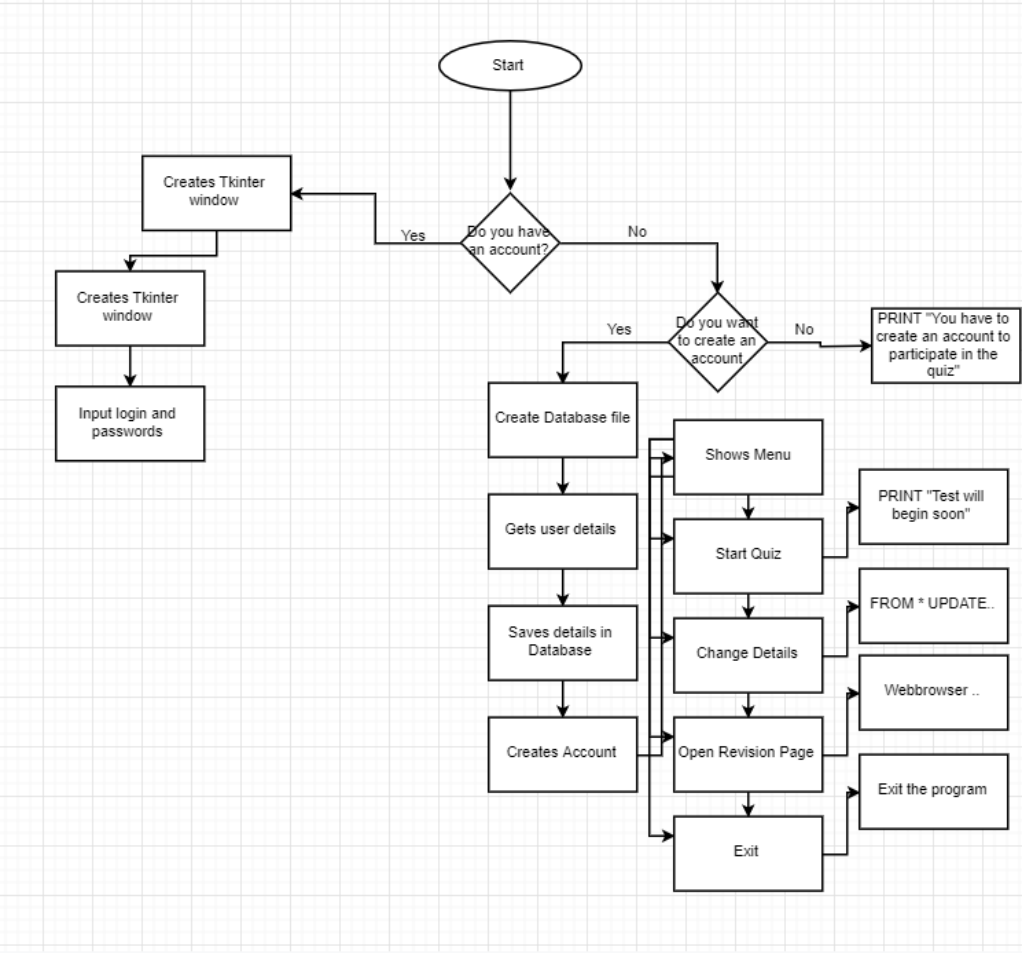
PSEUDOCODE AND FLOWCHARTS

```
IMPORT SQLITE3
IMPORT CSV
IMPORT OS
IMPORT CODECS
IMPORT WEBBROWSER
FROM TKINTER IMPORT *

CONEECT TO students' database

DEF create_table():
       CREATE TABLE IF NOT EXISTS students
        FirstName text
        Surname text
        DOB Integer
        Email Text

DEF login ():

       CREATE window (500x600)
       windows name is ("Economics Quiz")

       Label1 = Label(Name "Login System", relief = "solid", width =20,
font=("arial",19,"bold"))
       PLACE Label1 (x=90,y=53)

       Label2 = Label(Name "Username", width=20,font=("arial",10,"bold"))
       PLACE Label2 (x=80, y=130)

       Entry1 = Entry (root, textvar=fn)
       PLACE Entry1 (x=227, y=135)

       Label3 = Label (Name "Password", width=20, font= ("arial, 10," bold"))
       PLACE Label3 (x=227, y=200)

       Entry2 = Entry (root, textvar=ps)
       PLACE Entry2 (x=227, y=200)

       B1 = Button (Name "Exit", width=12, bg="red",fg="white", command=exit1)
       PLACE B1 (x=200, y=350)
```

```
	root.mainloop()

DEF log_in():

	Log_in: “Do you have an account? Y/N”
	VALIDATE Log_in
	IF Log_in == Y THEN
		Login()
	ELSE:
		User_profile()

DEF validate (string, chars):
	return TRUE in [c in strings for c in chars]

DEF user_profile():
	Sign_up: “Do you want to create an account? Y/N: ”
	VALIDATE Sign_up
	WHILE Sign_up == N
		IF Sign_up == N THEN
		OUTPUT “You have to create an account to participate in the quiz. ”
		Sign_up: “Do you want to create an account? Y/N: ”

	IF Sign_up == Y THEN
		Name: “Enter your First Name: ”
		VALIDATE Name

		Surname: “Enter your surname: ”
		VALIDATE Surname

		DoB: “Enter your Date of Birth: ””
		VALIDATE Dob

		Email: “Please enter a valid email: ”
		VALIDATE Surname

		Username: “Enter your surname: ”
		VALIDATE Username

		Password: “Enter your surname: ”
		OUYPUT “Password should be longer than 6 characters but less than 10”
		VALIDATE Password

		FirstName = FirstName
		Surname = Surname
		DOB = DOB
		Email = Email
		INSERT Details into Students’ database
		CALL Display_menu()
```

```
            #OPEN CSV file
            #RECORD Name
            #RECORD Surname
            #RECORD Dob
            #RECORD Email
            #RECORD Username
            #RECORD Password
            #CLOSE CSV file

DEF display_intro():
      title = "** Jhonny's Revision Program **"
      OUTPUT ("*"*len(title))
      OUTPUT(title)
      OUTPUT("*"*len(title))

DEF display_separator():
      OUTPUT ("-"*24)

DEF display_menu():

      NEW LINE
      ARRAY Menu: "[1. Start the Quiz, 2. Change Personal Details, 3. Access Revision for
quiz, 4. Exit]"
      OUTPUT Menu
DEF get_user_input():

      User_input: "Enter your choice"
      VALIDATE User_input
      RETURN User_input

DEF menu_option(index):

      IF index IS 1 THEN
            OUTPUT "Test will start soon"
      ELSEIF index IS 2 THEN
            OUTPUT "you will be able to change details"
      ELSE:
            new 2
            url =
      file:///D:/A%20level%20computing/Programming%20Project/Revision.html
            OPEN webbrowser (url,new = new)


DEF main()

      Log_in()
      Create_table()
      display_intro()
      get_user_input()
      display_separator()
```

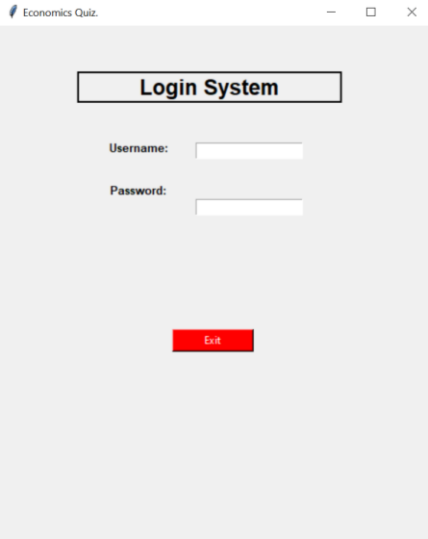
```
        Option = get_user_input()
        WHILE Option != 3 THEN
                Option = get_user_input()
        OUTPUT “Exit the quiz”

main()
```



In my first prototype I was only able to save the information of one student in a CSV file at a time, whenever I run the program again, the previous student details will be overwritten and lost. Whereas, this time I have created a database called Students that is able to save all the students in the class. I have also made the menu responsive, the test doesn't start but it does display a message saying "Test will begin soon", for the second option which allows the user to change their personal details also displays a message, and finally for the Revision Page I was able to open a new window in Google Chrome or Internet Explorer using another python library called webbrowser. This window is not fully finished but it does have a Title and links for each topic inside Theme 1. I still need to complete the HTML and make it neater, user friendly and appealing to use. In this HTML file I will add pictures and links to YouTube videos. If for any reason the user has no access to Internet; there will be summarised notes from the videos. Finally, the Exit options works.

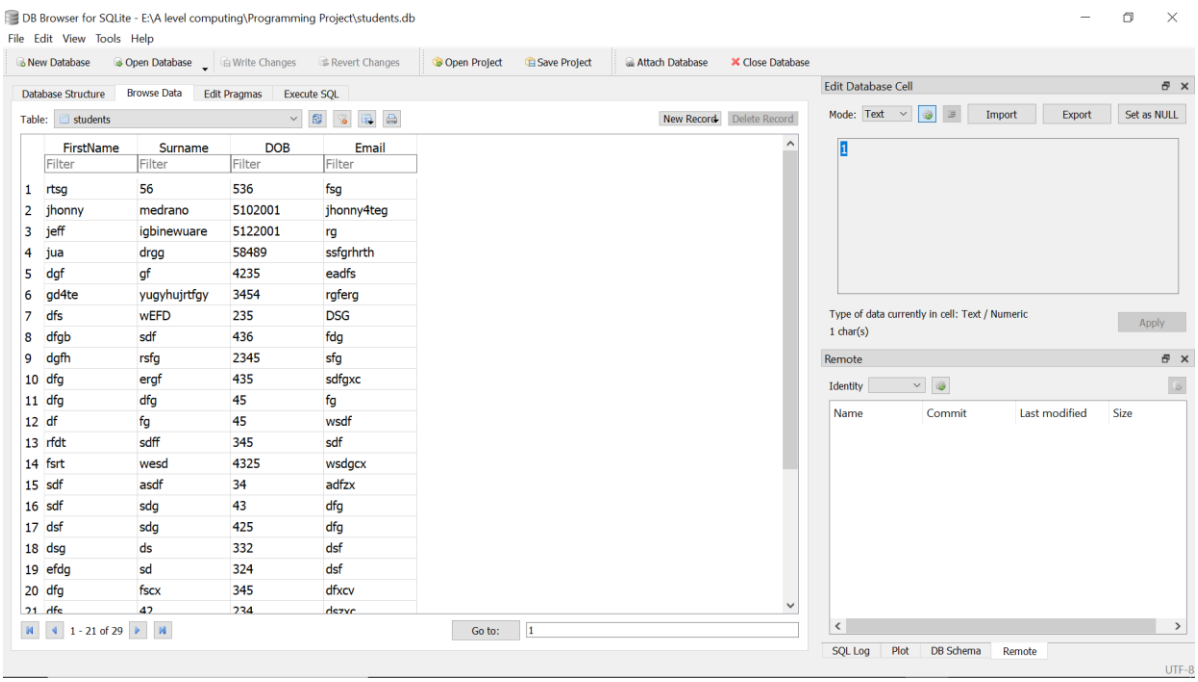
The flowchart also shows the steps from the beginning, all the way until the program end.

LOG IN window:



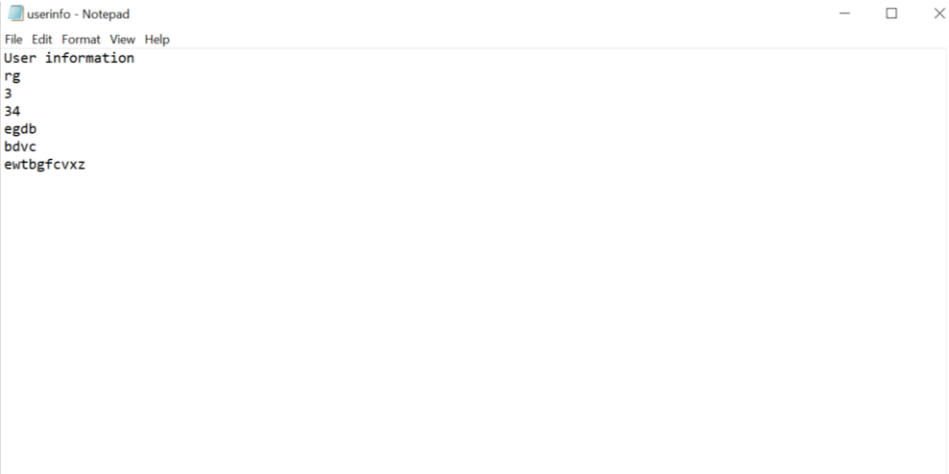
Log in: I also have developed this Tkinter window that will only work for the teacher to have an overview of the database. This window has 2 Entry's in which the user can type anything but will not do anything, this is still unfinished and does not respond, only the exit button works. However, I am having an issue on switching back to python; so far it only works if the Tkinter window is closed manually and I look forward to fixing this in the future. I'm debating if I should make this Tkinter window available for students due to the time I have available and the complexity.

Database: I decided to use SQLite3 for my database because SQL will need a server, the disadvantage of using SQLite3 is that I need an external downloaded application (DB browser for SQLite3) to open the database and look at all the students which have been saved inside. There is also another way to make sure that this database works without using another app, SQL command can be used to call all First names or Surnames. I will need to use this SQL commands to allow the user to Update their details



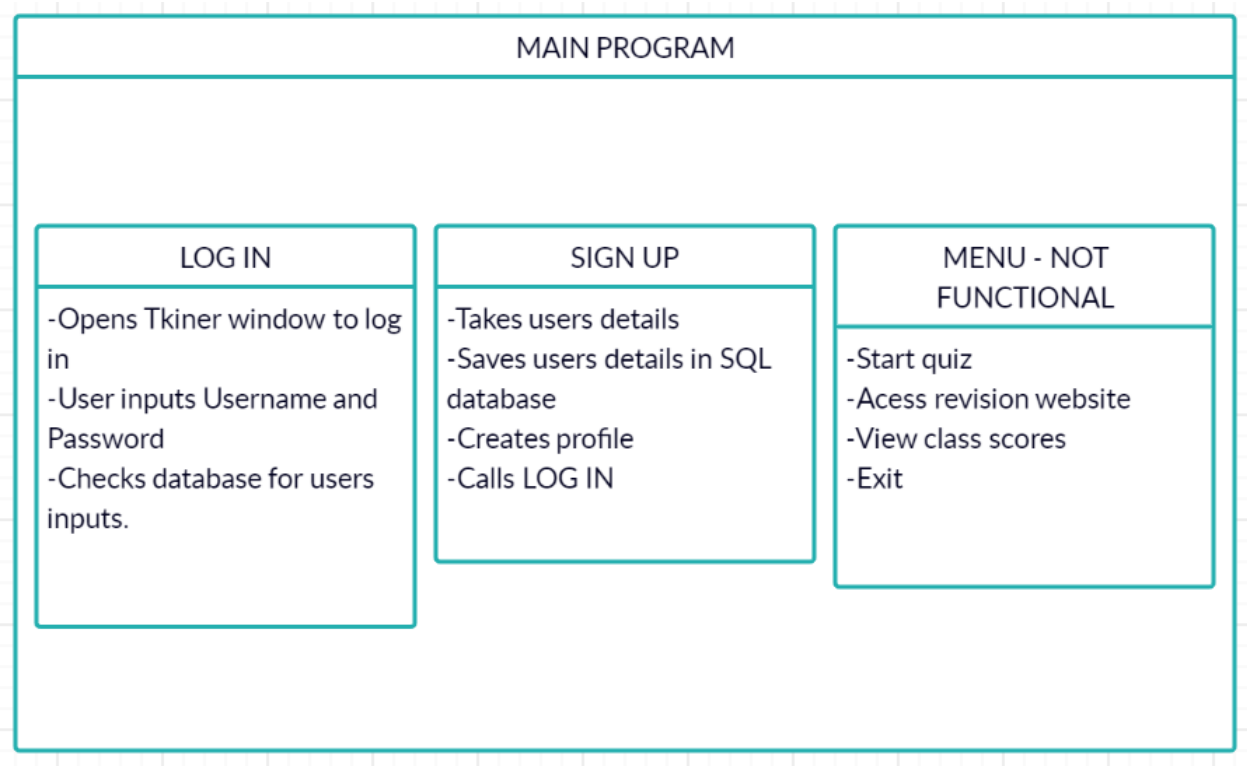
This is what the Database looks like once it is opened, as you can see the files are written, I need to make sure the program also validates the DOB and to enter a valid email. The database works correctly which is the main

purpose. If we compare this to the old file, we had I can see that this is more efficient. If I wanted to do this with csv files, I would have needed to create multiple files each time for each student.



This is a picture of the old file; it looks inefficient and unorganised whereas the new database looks better and more efficient. This separates the names in columns, this also doesn't need special to be downloaded, this can be opened with notepad. This table was useful if only one test was taking the quiz, it did what it was intended, created and saved important details. An important factor to mention is that before realising I was going to make a database; I already made the code to update the details from a csv file. Unfortunately, I did not take a screenshot so I will be using pseudocode.

DIAGRAM FOR PROTOTYPE 2



This is the second version for the modular diagram, this is a more advanced version because, the user can sign up and see a menu which is still not functional. However, this time the user can log in to the Tkinter window and once they have successfully logged in, they will be shown the menu in the python interface. The development in this section has improved a lot since a database is being linked to the python interface and is able to check that database once the user is trying log in. In the next prototype the user will be shown a menu which will be fully functional, and the admin area will also be created.

DATABASE DIAGRAM FOR PROTOTYPE 2

USERS	
FirstName	String
Surname	String
DOB	Integer
Email	String
Username	String
Password	String

VARIABLES/ STRUCTURE TABLE UPDATED

Variable/Structure Name	**Purpose**	**Data Type**	**Validation**
Conn	Variable that connects to the database	Variable	
Create_table()	This creates the database. This database will be used for	Subroutines	This uses “IF NOT EXISTS”
Login()	Creates a tkinter window and	Subroutines	
Log_in()	T6his is to ask the question if someone has followers	Subroutines	
Sign_up	This stores the answer and get	Variable	Presence check to then be inputted into the Database.

FirstName	Stores the String given by the user.	Variable	Presence check to then be inputted into the Database.
Surname	Stores the String given by the user.	Variable	Presence check to then be inputted into the Database.
DOB	Stores the value given by the user.	Variable	Presence check to then be inputted into the Database.
Email	Stores the String given by the user.	Variable	Presence check to then be inputted into the Database.
Username	Stores the String given by the user.	Variable	Presence check to then be inputted into the Database.
Password	Stores the String given by the user.	Variable	Presence check to then be inputted into the Database.
Get_user_input()	This is linked to the menu and will get the option that is given to them.	Subroutine	Compare values to know if the menu list goes over a specific value.
Label1	This is to show the “Login System”	Variable	The position of the writing
Label2	This shows the “Username” writing	Variable	The position of the writing
Label3	This shows the “Password”	Variable	The position of the writing
Entry1	Allows the user to enter the username	Variable	There is no validation yet
Entry2	Allows the user to enter the password	Variable	There is no validation yet
B1	This will be used to exit the Tkinter window	Variable	There will be a command in order to exit the Tkinter window.
fn	This is in order to get the First name from the user	Variable	Makes sure the variable is a string inside the tkinter entry

Ps	This is to get the password from the user	Variable	Makes sure the variable is a string inside the tkinter entry

TEST PLAN FOR PROTOTYPE 2 BLACK BOX

Entry	Expected outcome
Database	A database to be created if one doesn't already exist in order to save multiple users.
Takes User Details	Validates, makes sure field is not empty and saves them in a database
Presence Check	Iterates if the field has been left empty, continues if the field has a valid input.
User wants to log in	Graphical User Interface opens
Validates DOB and Email	Allow valid emails, iterate if not. Won't allow characters for the DOB
Fully functional menu	When the user enters the option, the program should run.
HTML	If the user wants to open the revision website, this should open
Exit Button	The GUI closes

TEST PLAN FOR PROTOTYPE 2 WHITE BOX

Test	Expected Process
GUI	The Tkinter window will be shown with the login system, the different entries.
Database	The database will be created before the program asks the user if they have an account
The database will save details	To save the details, when they are inputted straight from the python interface.
Tkinter entries	The GUI will allow for user inputs

Validation of Tkinter Entries	The program will validate the username and password in order to access the quiz. This will use SQL commands to traverse the database
After user logs in	Other functions will be called once they have access.

DEVELOPMENT

```
import sqlite3
import csv
import os
import codecs
import webbrowser
from tkinter import *

conn = sqlite3.connect('students.db')

c = conn.cursor()

def create_table():
    c.execute("""CREATE TABLE IF NOT EXISTS students (
                FirstName Text,
                Surname Text,
                DOB Integer,
                Email Text)""")
```

This is the code for the second prototype, it shows that I am importing many libraries, luckily all these libraries are available to me in school campus, while my code progresses I won't need the csv because I connected an SQLite database to my program.

The SQLite3 library is what allows me to connect to the database and create one.

Both codecs and webbrowser library are libraries to open a file which was going to be used to be used to open my HTML file for the option given in the menu.

This also shows the way of the database being created if it hasn't already. I have mainly used subroutines, this is because I don't want the whole program failing if only one thing is wrong, it helps me identify errors quicker because the program gives me the line it was found in.

```
def login():
    root =Tk()#Creates GUI
    root.geometry("500x600")#This establishes how big the window opened will be
    root.title("Economics Quiz.")#Title of the window that has been opened


    def printt():
        print("Demo tkinter")
    fn=StringVar()
    ps=StringVar()
    def exit1():
        exit()

    label1=Label(root,text="Login System",relief = "solid",width=20,font=("arial",19,"bold"))
    label1.place(x=90,y=53)#Places Label 1 in the given position

    label2=Label(root,text="Username: ",width=20,font=("arial",10,"bold"))
    label2.place(x=80,y=130)#Places Label 2 in the given position

    entry1=Entry(root,textvar=fn)
    entry1.place(x=227,y=135)#Places Entry 1 in the given position

    label3=Label(root,text="Password: ",width=20,font=("arial",10,"bold"))
    label3.place(x=80,y=179)#Places Label 3 in the given position

    entry2=Entry(root,textvar=ps)
    entry2.place(x=227,y=200)#Places Entry 2 in the given position

    b1=Button(root,text="Exit",width=12,bg="red",fg="white",command=exit1)
    b1.place(x=200,y=350)#Places button 1 in the given position

    root.mainloop()
```

This shows how the GUI is designed, as you can see, there is 2 subroutines built inside the login(), for example exit1() is used for the button, the user has an option to exit the program and whenever they press it, that function will be called.

The rest are what is inside the GUI, the tkinter window only has a login, but doesn't respond yet because it is not linked to any database at the moment, however the admin will be allowed to log in and have an overview.

```
def log_in():
    log_in = ""
    while log_in!= "Y" and log_in!="N":
        log_in = input("Do you have an account? Y/N: ").upper()
        if log_in == "Y":
            login()#If the user inputs Y it will call this function
        else:
            user_profile()#If they dont have an account it will ask them if they want to create one


def user_profile():

    sign_up = " "
    while sign_up!= "Y" and sign_up!= "N":
        sign_up = input("Do you want to create an account? Y/N: ").upper()
        while sign_up == "N":
            if sign_up == "N":
                print("You have to create an account to participate in the quiz.")
                sign_up = input("Do you want to create an account? Y/N: ").upper()


    if sign_up == "Y":

        FirstName = ""
        while FirstName == "":
            FirstName = input("Enter your first name: ")

        Surname = ""
        while Surname == "":
            Surname = input("Enter your surname: ")

        DOB = ""
        print("Please enter your DOB like this: 09 10 2003 with no spaces")
        while DOB.isnumeric() == False:
            DOB = input("What is your Date of Birth? ")

        Email = ""
        print("Please enter a valid email.")
        while Email == "":
            Email = input("Enter your email: ")

        Username = ""
        while Username == "":
            Username = input("Create a username: ")


        Password = ""
        print("Password should be longer than 6 characters but less than 10")
        while len(Password) <6 or len(Password)>10:
            Password = input("Create a password: ")

        FirstName = FirstName
        Surname = Surname
        DOB = DOB
        Email = Email
        c.execute ("INSERT INTO students (FirstName, Surname, DOB, Email) VALUES (?,?,?,?)",
                   (FirstName, Surname, DOB, Email))
        conn.commit()
        display_menu()
```

If the user wishes to create an account this subroutine called user_profile(), in this subroutine the program takes essential details that are needed to create an account like: First name, Surname, DOB, Email and finally they will create a Username, to ensure that all this credentials are filled in, the program will use presence check, even if the user presses enter the program will repeat until a valid input has been entered.

Also, at the bottom it shows we input variables into a database, we name the variables and assign them to their original input, the command “INSERT INTO” will add all the inputs given by the user

```
        #f = open("userinfo.txt","w")
        #f.write("User information"+"\n")
        #f.write(str(FirstName)+"\n")
        #f.write(str(Surname)+"\n")
        #f.write
        #(str(DOB)+"\n")
        #f.write(str(Email)+"\n")
        #f.write(str(Username)+"\n")
        #f.write(str(Password)+"\n")
        #f.close()
        #print("Record has been written to file")
    else:
        exit()
```

This used to be the old file that was created when the user made their profile, as said previously this is no longer the case because the csv file is inefficient. The reason this is read because its no longer in use but I might use it to save other details in the future. It doesn’t affect the program at all.

```
def display_intro():#This helps organise the program
    title = "** Jhonny's Revision Program**"
    print("*" * len(title))
    print(title)
    print("*" * len(title))

def display_menu():
    menu_list = ["1.Start the quiz.","2.Change Personal Details.","3. Acess Revision for quiz.","4.Exit"]
    print(menu_list[0])
    print(menu_list[1])
    print(menu_list[2])
    print(menu_list[3])


def display_separator():
    print("-" * 24)

def get_user_input():
    user_input = int(input("Enter your choice: "))
    while user_input > 4 or user_input <=0:
        print("Invalid menu option. ")
        user_input = int(input("Enter your choice: "))
    else:
        return user_input

def menu_option(index):
    index=index+1
    if index is 1:
        print("Test will start soon")
    elif index is 2:
        print("You will be able to start the quiz soon")
    else:
        new = 2
        url= "file:///D:/A%20level%20computing/Programming%20Project/Revision.html"
        webbrowser.open(url,new=new)

def main():
    log_in()
    create_table()
    display_intro()
    get_user_input()
    display_separator()

    option = get_user_input()
    correct = 0
    while option != 3:
        correct = menu_option(option)
        option = get_user_input()

    print("Exit the quiz. ")
    display_separator()


main()
```

The display separator gives a title to the actual python program and displays before the menu is shown.

The menu is also showed here, and it shown inside an array, this adds complexity to the program, the way it gets the input from the user is under the subroutine get_user_input()

The code to opening the HTML file is allowed because the webbrowser library has been imported at the beginning, the way it works is by calling the actual URL from the HTML window and will open a new window in the browser.

Lastly, the main() subroutine is called which calls all the functions in order to make sure the program is organised.

ERRORS DURING DEVELOPMENT

```
File "E:\A level computing\Programming Project\Official Program.py", line 619, in log_in
  login()
File "E:\A level computing\Programming Project\Official Program.py", line 577, in login
  label1=label(root,text="Login System",relief = "solid",width=20,font=("arial",19,"bold"))
NameError: name 'label' is not defined
```

This is the one of the many syntax errors that I encountered during this development. The Tkinter build up was difficult to counter, tiny mistakes could make the program crash. In the error it said that the variable label was not defined. I look at that line and all the lines of Tkinter and tried changing every variable around it.

This is the line which was wrong:

```
label1=label(root,text="Login System",relief = "solid",width=20,font=("arial",19,"bold"))
label1.place(x=90,y=53)#Places Label 1 in the given position
```

This is the line that the program outputted as an error, the error was that the label after the equal sign had to be capitalised to “label1=Label” once I done that and fixed it correctly opened the Tkinter window which allowed the user to input 2 credentials. It didn’t allow them to log in yet.

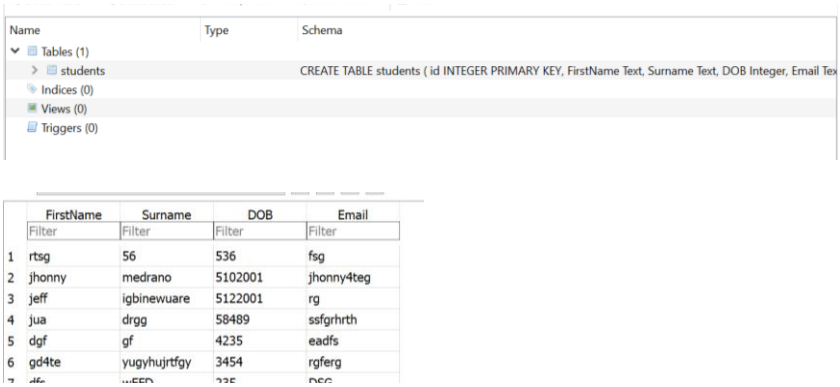
Most of the other errors during this development were to do with this Tkinter build up, the majority were to do with capitalising a word.

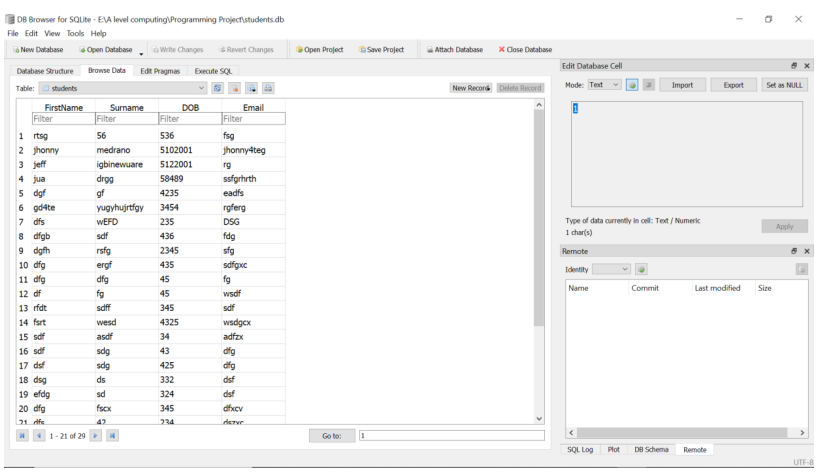
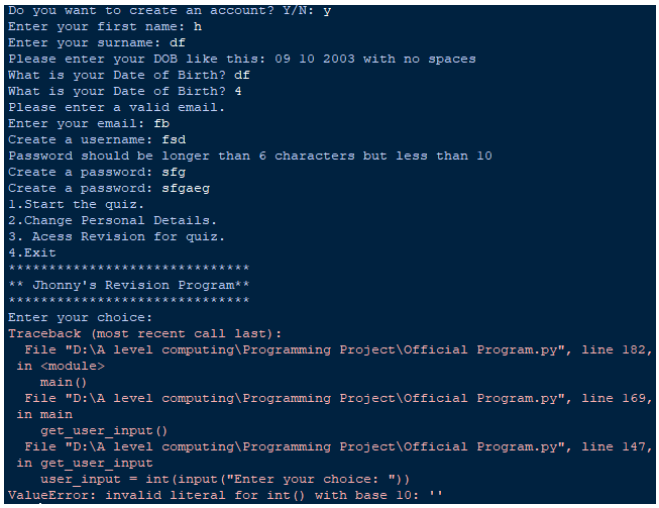
There was another error, it wasn't a syntax error, it was a logic error, the menu was functional this time; it allowed the user to open the revision website if they inputted the option. The program had to accept the user input "3" in order to open the website but this wasn't the case, when the user inputted "2" the program opened the site, this had to do with the index which was assigned incorrectly in another function. I tried making it global which didn't work so then I created a subroutine to pass it on, this didn't work either, so the menu was again unsuccessful. The code will be shown below.

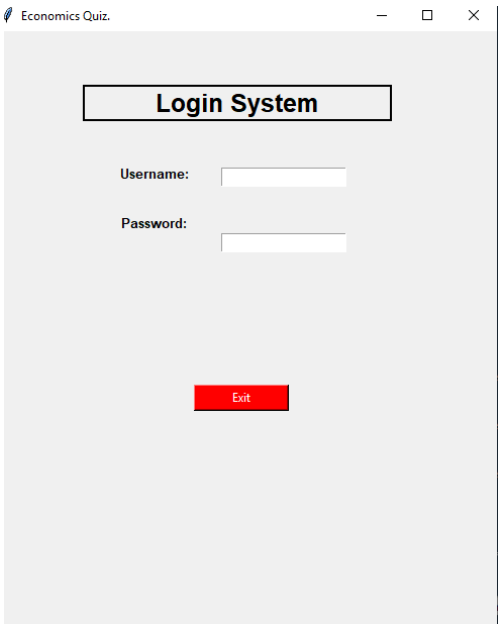
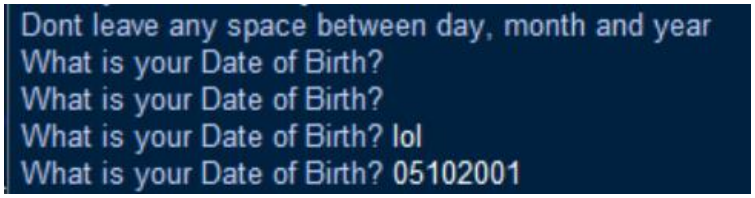
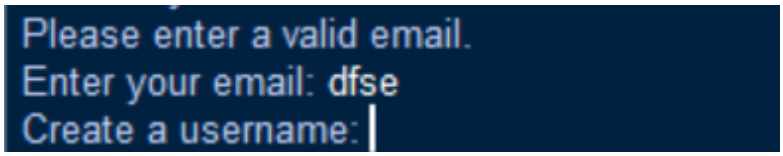
```
def menu_option(index):
    index=index+1
    if index is 1:
        print("Test will start soon")
    elif index is 2:
        print("You will be able to start the quiz soon")
    else:
        new = 2
        url= "file:///D:/A%20level%20computing/Programming%20Project/Revision.html"
        webbrowser.open(url,new=new)
```

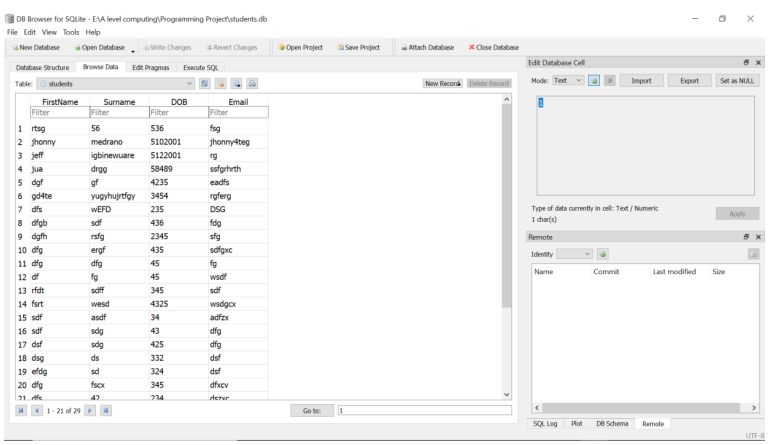
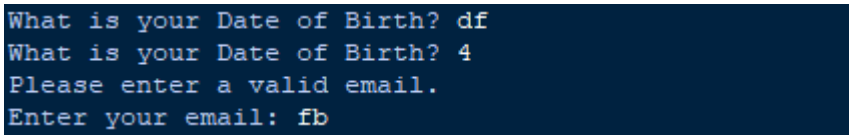
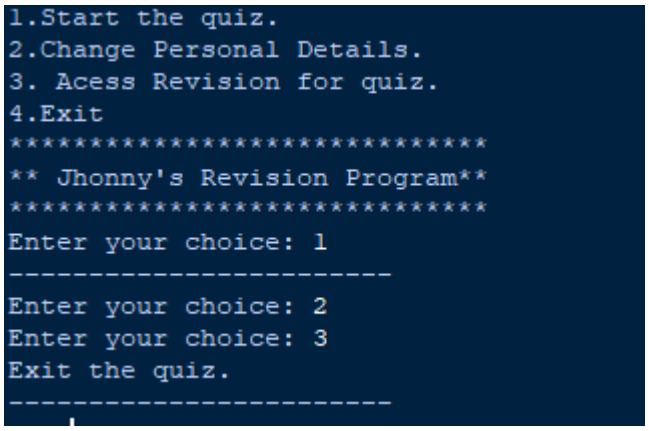
TESTING

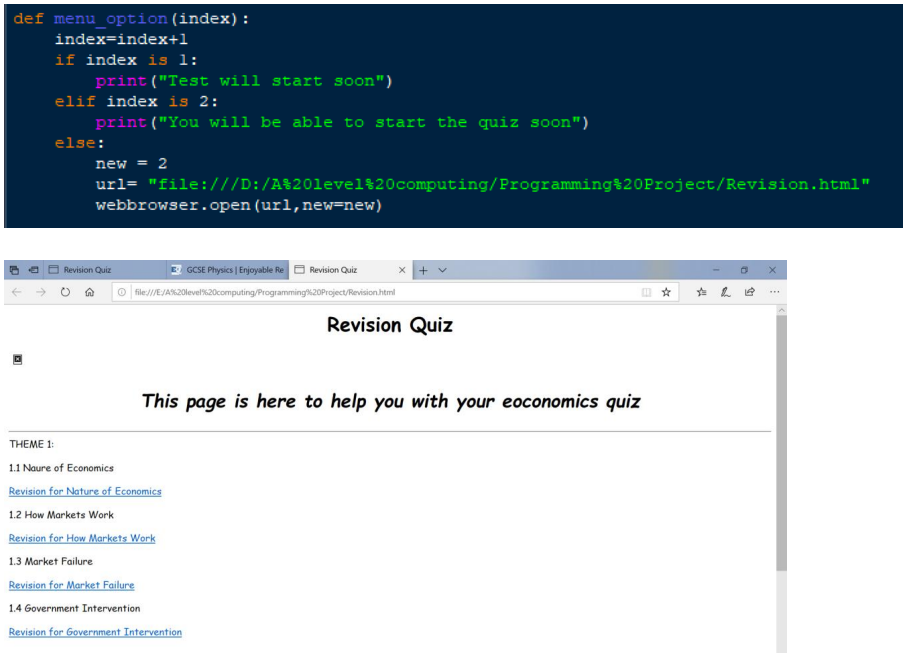
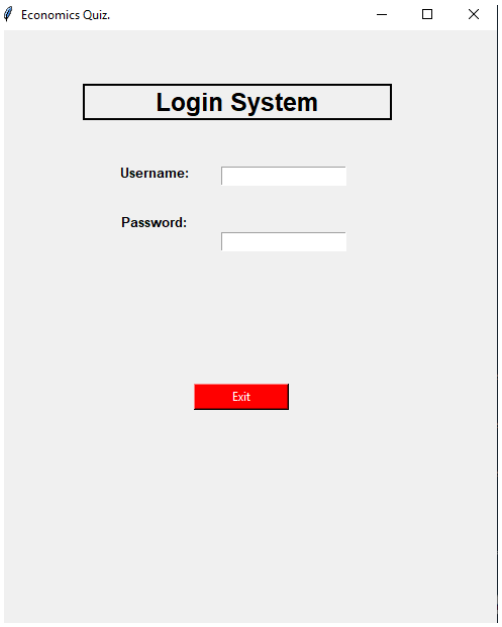
BLACKBOX TESTING

Success Criteria	Achieved?	Justification	Screenshot
Database	Yes	This database had to be created in order to save multiple users in a database, in prototype 1 only one person could be saved.	 FirstName Surname DOB Email 1 rtsg 56 536 fsg 2 jhonny medrano 5102001 jhonny4teg 3 jeff igbinewuare 5122001 rg 4 jua drgg 58489 ssfgrhrth 5 dgf gf 4235 eadfs 6 gd4te yugyhujrtfgy 3454 rgferg It's a simple database that takes their names, surnames, DOB and Email.

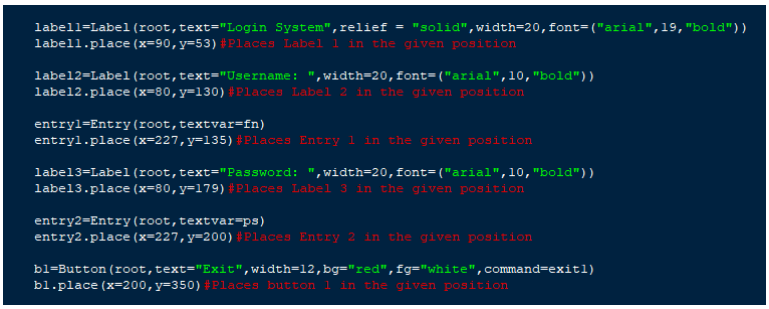
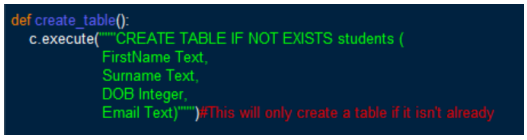
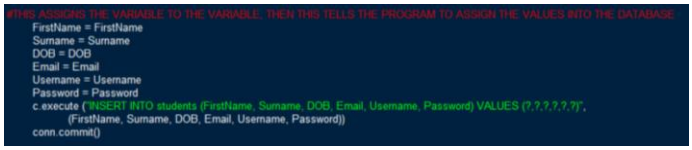
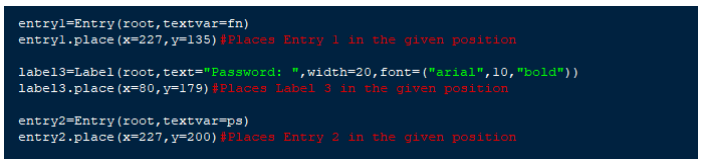
Takes User Details	Yes	The database records every detail that has been taken and insert them dynamically into the database.	
Presence Check	Partly	The meu is not fully responsive, it will give a syntax error instead of asking the user what option they want to input again.	

Users wanting to log in	Yes	This was created with tkinter and works, it shows the other window.	
Validates DOB and Email	Partly	This should always be validated, so the profile can be created successfully.	 The DOB has been able to be validated and not allow empty spaces or strings, it only accepts integers. However, the email is not validated.  The program continues to the next variable even if the user enters an invalid email.

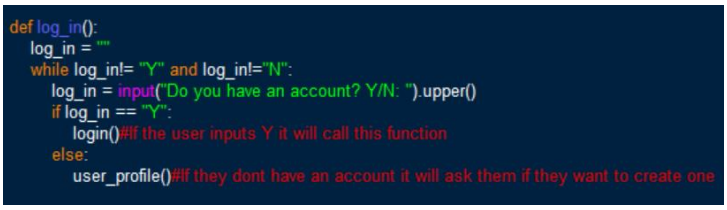
Creates Database and stores all the details given.	Yes	The database is successful at saving all the credentials under their specific row and column.	
Validates DOB and Email	No	Unfortunately, the program doesn't validate the length of the DOB yet nor it check if the email is valid, it will accept any value given as long as it's the right data type.	
Fully functional menu	No	The array isn't fully functional, if I input a valid option will output it in the wrong order. When I input 2 it opens the HTML	

Opens HTML	Yes	Even though the menu isn't fully functional it opens HTML thanks to the webbrowser library.	
Exit button	Yes	This should allow the user to go back to the Pythons interface to look at the menu.	 Once the exit button Is pressed the program will close the GUI window and go back to the python interface.

WHITEBOX TESTING

Test	Expected Process	Successful? / Evidence	Actions to take
GUI	The Tkinter window will be shown with the login system, the different entries.		In prototype 2, I will make the password hidden by adding "*" to any character types in
Database	The database will be created before the program asks the user if they have an account	Yes 	N/A
The database will save details	To save the details, when they are inputted straight from the python interface.	Yes 	N/A
Tkinter entries	The GUI will allow for user inputs	Yes  Both entries are shown once the user decides they want to log in, this is the code to represent the entries inside the GUI	

Validation of Tkinter Entries	The program will validate the username and password in order to access the quiz. This will use SQL commands to traverse the database	No The user is not able to log in yet, they can input anything into the Tkinter entries.	For the next prototype I am aiming to validate the inputs. This will be done by using SQL commands by traversing the array and finding the unique username and password. I will also check if the username has been already been taken by another user.

After user logs in	Other functions will be called once they have access.	Yes
		 
		def log_in():
		log_in = ""
		while log_in!= "Y" and log_in!="N":
		log_in = input("Do you have an account? Y/N: ").upper()
		if log_in == "Y":
		login()#If the user inputs Y it will call this function
		else:
		user_profile()#If they dont have an account it will ask them if they want to create one
			I will have to call more functions in protype 3 in order to fulfil users.

BETA TESTING

For this prototype I allowed other students to try this program out. They saw that there was a change at the start of the program, they now saw that there is a log in option. They said that this is much better than before because the log in window looked like an actual professional website. They could input the username and password, but they still couldn't log in because the menu wasn't working. Once they closed the Tkinter window the menu will pop up, but it still wasn't functional, so they weren't satisfied. They didn't know but there was a database which was created to allow multiple student to have accounts.

The users liked the program so far, but still wanted to have the result. For prototype 3, the majority of user requirements and success criteria will be met, and I will have a larger sample of students to try the quiz.

EVALUATION

So, in this prototype I advanced a lot, I created a database and the basics of GUI, I achieved more user requirements which is the main goal since I have achieved all this I will need to have another meeting with the stakeholders, both students and teachers. This meeting will help me update my user requirements and improve

my success criteria. I did research to create the GUI. From the testing I have realised that I have not met many of the criteria I initially proposed myself. However, I will try and implement all the things, there are more tests I will test out in the third prototype like the hashing algorithm to the passwords and test the functionality of my HTML because right now it's really basic HTML coding, I want the HTML file to be more complex and the functionality to improve the user-friendly approach. By user friendly I mean that the website will be easy to use and then I will allow a stakeholder to try this website and give me an insight of what works and what doesn't, since I am the developer I will be doing normal tests but in a day to day usage a user might wonder off and do something unexpected.

For the third prototype I aim to complete all my success criteria and achieve all user requirements, finish the code and try to implement the GUI system to both student and administrator, with this I will need to separate the rights because only the admin will have full access to the command of the database.

There are many criteria's that I still have not met in prototype 2 which I intended to do since prototype 1 didn't meet many either. For prototype 3 I intend to make the menu responsive to all the options the user has. The extensive research done has stopped me from developing further. However, now that I understand how to build a Tkinter GUI window it will be much easier to create the admin area. I still need to research a lot like the validation of the username and password inside the login function because it uses SQL commands inside the Tkinter window.

I also still have not met the requirement of the validation of all variables which I intended to do in prototype 1, the email is the main cause, I still have not found a way I can successfully validate the email correctly and make sure it's in a valid format.

The maintenance is also not consistent now due to the lack of comments in the code, but this will change at the end of prototype 3. I will make sure every function and variable is explained so if another programmer wants to use my code for any reason the code will be understandable and easy to follow.

There has not been a lot of further development because there were not that many errors during this prototype, mostly they came from the Tkinter build up and it was because I didn't capitalise some words which I amended after and worked correctly.

PROTOTYPE 3

I'm currently seeking out to previous stakeholders in order to discuss the design and the program. The reason for this is because the GUI has been almost completed and all commands work accordingly; from this interview I'm going to gather constructive criticism so I can improve the program whether it's in terms of the Tkinter design, structure of the quiz or the HTML.

From having an interview with Mrs. Kaur-Hender she has come up with things she would like to change about my program and things that are not needed. These are the aspect that my client wants me to implement or change:

- Create another file that only displays the name of the student and their score.
- User should be able to access the revision site.
- The student shouldn't have access to the teacher's system.
- Make the test challenging.
- Validate all inputs.
- Make the menu inside the Tkinter window instead of the python interface.

- Update the revision website.

FURTHER SUCCESS CRITERIA FOR PROTOTYPE 3

The user requirements have allowed me to create a success criterion which is helps me to finish the program.

- Program will not crash after confirmation of exit button.
- Program will follow sub routines as they are given, even if the program has already been through every other subroutine.
- Program will display quiz.
- The email should be validated, loop if wrong.

These success criteria have been changed so it fits the users' needs as well as the development for the program. These success criteria changed due to my client's needs. I also had an interview with one of the students and decided to change my user requirements.

UPDATED USER REQUIREMENTS

- Change background colour for Tkinter GUI.
- Students being able to get feedback at the end of the quiz.
- Display score at the end of quiz automatically with feedback.

These requirements will also help me fit the user requirements need and make it more appealing to the user(student) which will be the one taking the quiz. The student wants me to change the colour because the current one looks very "robotic" and "unreal".

DESIGN

PSEDUCODE OF PROGRAM PART I

This is the essential part of the program because, it takes the users details, it saves them in a SQL database which can be accessed by the teacher, the GUI is also built here, all starting from menus and everything that runs the program. The comments will not be implemented here because there is no space for it. However, I will be explaining all the functions one by one.

```
IMPORT SQLITE3
IMPORT CSV
IMPORT OS
IMPORT WEBBROWSER
FROM TKINTER IMPORT *
FROM TKINTER IMPORT MESSAGE BOX AS MS
FROM TKINTER IMPORT TK, BUTTON
IMPORT BACK

CONEECT TO students' database

DEF create_table():
       CREATE TABLE IF NOT EXISTS students
        FirstName text
```

```
        Surname text
        DOB Integer
        Email Text
                 Username Text
                 Password Text
```

This function is used to create the SQL database, this is done so the students or teachers details can be saved, there is 2 extra columns that have been created since the last prototype. This is to ensure that each user has their own login and password.

```
DEF get_selected_row(event):

    GLOBAL selected_tuple
    index=list1.curselection()[0]
    selected_tuple=list1.GET(index)
    entry1.DELETE(0,END)
    entry1.INSERT(END,selected_tuple[1])
    entry2.DELETE(0,END)
    entry2.INSERT(END,selected_tuple[2])
    entry3.DELETE(0,END)
    entry3.INSERT(END,selected_tuple[3])
    entry4.DELETE(0,END)
    entry4.INSERT(END,selected_tuple[4])
    entry5.DELETE(0,END)
    entry5.INSERT(END,selected_tuple[5])
    entry6.DELETE(0,END)
    entry6.INSERT(END,selected_tuple[6])

DEF view_command():
list1.DELETE(0,END)
FOR row in back.view():
         list1.INSERT(END,row)


DEF search_command():
list1.DELETE(0,END)
FOR row in
back.SEARCH(Name_text.GET(),Surname_text.GET(),DoB_text.GET(),Email_text.GET(),Userna
me_text.GET(),Password_text.GET()):
         list1.INSERT(END,row)


DEF add_command():

back.INSERT(Name_text.GET(),Surname_text.GET(),DoB_text.GET(),Email_text.GET(),Userna
me_text.GET(),Password_text.GET())
list1.DELETE(0,END)
list1.INSERT(END,Name_text.GET(),Surname_text.GET(),DoB_text.GET(),Email_text.GET(),U
```

```
sername_text.GET(),Password_text.GET())


DEF delete_command():
back.DELETE(selected_tuple[0])


def update_command():
back.UPDATE(selected_tuple[0],Name_text.GET(),Surname_text.GET(),DoB_text.GET(),Email
_text.GET(),Username_text.GET(),Password_text.GET())

DEF registration():

CREATE window

Label1=Label(window,text="Students' Login Details")
PLACE Label1 IN grid(row=0,column=2)

Label2=Label(window,text="Name")
PLACE Label2 IN grid(row=1,column=0)

Label3=Label(window,text="Surname")
PLACE Label3 IN grid(row=2,column=0)

Label4=Label(window,text="DoB")
PLACE Label4 IN grid(row=3,column=0)

Label5=Label(window,text="Email")
PLACE Label5 IN grid(row=4,column=0)

Label6=Label(window,text="Username")
PLACE Label6 IN grid(row=5,column=0)

Label7=Label(window,text="Password")
PLACE Label7 IN grid(row=6,column=0)

GLOBAL entry1
GLOBAL Name_text
Name_text=StringVar()
entry1=Entry(window,textvariable=Name_text)
PLACE entry1 IN grid(row=1,column=1)

GLOBAL entry2
GLOBAL Surname_text
Surname_text=StringVar()
entry2=Entry(window,textvariable=Surname_text)
PLACE entry2 IN grid(row=2,column=1)

GLOBAL entry3
GLOBAL DoB_text
DoB_text=StringVar()
```

```
entry3=Entry(window,textvariable=DoB_text)
PLACE entry3 IN grid(row=3,column=1)

GLOBAL entry4
GLOBAL Email_text
Email_text=StringVar()
entry4=Entry(window,textvariable=Email_text)
PLACE entry4 IN grid(row=4,column=1)

GLOBAL entry5
GLOBAL Username_text
Username_text=StringVar()
entry5=Entry(window,textvariable=Username_text)
PLACE entry5 IN grid(row=5,column=1)

GLOBAL entry6
GLOBAL Password_text
Password_text=StringVar()
entry6=Entry(window,textvariable=Password_text)
PLACE entry6 IN grid(row=6,column=1)

GLOBAL list1
list1=Listbox(window,height=20,width=59)
PLACE list1 IN grid(row=1,column=3, rowspan=6, columnspan=2)

scrl=Scrollbar(window)
scrl.grid(row=1,column=2, sticky='ns',rowspan=6)

list1.CONFIGURE(yscrollcommand=scrl.set)
scrl.CONFIGURE(command=list1.yview)

list1.BIND('<<ListboxSelect>>',get_selected_row)

b1=Button(window,text="View all",width=12, command=view_command)
PLACE b1 IN grid(row=7, column=0)

b2=Button(window,text="Add entry",width=12,command=add_command)
PLACE b2 IN grid(row=9, column=0)

b3=Button(window,text="Delete entry",width=12,command=delete_command)
PLACE b3 IN grid(row=11, column=0)

b4=Button(window,text="Search",width=12,command=search_command)
PLACE b4 IN grid(row=7, column=1)

b5=Button(window,text="Update",width=12,command=update_command)
PLACE b5 IN grid(row=9, column=1)

b6=Button(window,text="Further Options",width=12,command=menu2)
PLACE b6 IN grid(row=11, column=1)
```

```
window.attributes("-topmost", True)

window.mainloop()
```

All of this is to create the admin part of the program. This makes sure it opens the GUI which allows them to see all the details of each student that has created an account in order to participate. All these commands are also linked to another program which will be shown further on.

```
DEF menu ():

DEF Create():

        Names = []
        Scores = []
        Scores.append(counter)

        Name = ""
        WHILE Name == "" or Name.isnumeric() == True:
        Name = INPUT("Enter your first name: ")
        Names.append(Name)
        Score = open("Score.csv","a+")
        WITH OPEN("Score.csv","a", newline="") as file:
               writer = csv.writer(file)
               writer.writerow((Names, Scores))
```

This is to create a CSV file which will be useful, it will be useful because it will save all the students scores when they finish taking the quiz. It will do this by taking the name, appending into a list and then write whatever is in that array into the CSV file. I had to make the counter global which is shown in the next def function in order to be able to append it into the array called counter and then into the CSV without skipping any line.

```
    def quiz():

        root.destroy()
        GLOBAL counter
        counter = 0

        PRINT("1.demand curve")
        PRINT("2.supply curve")
        PRINT("3.ad")
        PRINT("4.as")

valid = False
WHILE not valid:

       q1 = INPUT("Which curve slops upwards? ")
       IF not validate(q1,"1234"):
           PRINT("Answer is not valid")
```

```
   ELSE:
        valid = True

IF q1 == "2":
    counter = counter+1

ELSE:
     PRINT("Incorrect")

      PRINT("\n")
      PRINT("1.Leather")
      PRINT("2.Pork")
      PRINT("3.Gravy")
      PRINT("4.Fish")
      valid = False
      WHILE not valid:
              q2 = INPUT("Which of the following products are in joint supply with
meat?")
      IF not validate(q2,"1234"):
            PRINT ("Answer is not valid")
      ELSE:
             valid = True
      IF q2 == "1":
           counter = counter+1
      ELSE:
           PRINT("Incorrect")

PRINT("\n")
PRINT("1.Elastic demand for exported goods")
PRINT("2.Elastic demand for imported goods")
PRINT("3.Inelastic demand for exported goods")
PRINT("4.Inelastic demand for imported goods")
valid = False
WHILE not valid:
       q3 = INPUT("Which of the follwoing would reduce the impact of a tarrif?")
      IF not validate(q3,"1234"):
          PRINT("Answer is not valid")

    ELSE:
        valid = True

IF q3 == "4":
     counter = counter + 1

ELSE:

     PRINT("Incorrect, start revising")
```

```
PRINT("\n")
PRINT("1.Derived supply")
PRINT("2.Structural Unemployment")
PRINT("3.Derived Demand")
PRINT("4.Seasonal Unemployment")
valid = False
WHILE not valid:
      q4 = INPUT("The demand for construction workers being dependent on the demand
for new     housing referred to as")

     IF not validate(q4,"1234"):
           PRINT("Answer is not valid")

      ELSE:
           valid = True

  IF q4 == "3":
       counter = counter+1

  ELSE:
        PRINT("Incorrect")

PRINT("\n")
PRINT("1.Imports")
PRINT("2.Savings")
PRINT("3.Taxes")
PRINT("4.Exports")
valid = False

WHILE not valid:
     q5 = INPUT("An increase in which of the following will directly lead to an
increase in AD")

     IF not validate(q5,"1234"):
           PRINT("Answer is not valid")

       ELSE:
           valid = True
IF q5 == "4":
      counter = counter+1
ELSE:
      PRINT("Incorrect")
PRINT("\n")
PRINT("1.Fall in the availability of credit")
PRINT("2.Rise in corporation tax")
PRINT("3.Fall in the expected rate of return")
PRINT("4.Fall in the cost of borrowing")
valid = False
WHILE not valid:
```

```
       q6 = INPUT("Which of these factors is most likely to lead an increase in
investment spending by firms?")

      IF not validate(q6,"1234"):
             PRINT("Answer is not valid")
       ELSE:
           valid = True

IF q6 == "4":
      counter = counter+1
ELSE:
     PRINT("Incorrect")

PRINT("\n")
Print("1.Solicitor")
PRINT("2.Lawyer")
PRINT("3.Fast food restaurant employee")
PRINT("4.University Professor")
valid = False
WHILE not valid:
         q7 = INPUT("Which of these jobs is likely to have a more elastic supply?"
       IF not validate(q7,"1234"):
           PRINT("Answer is not valid")
        ELSE:
            valid = True
IF q7 == "3":
       counter = counter+1
ELSE:
        PRINT("Incorrect")

PRINT("\n")
PRINT("1.Falling income tax receipts")
PRINT("2.A rise in company profits")
PRINT("3.Fewer people claiming Job Seekers Allowance")
PRINT("4.A rising rate of inflation")
valid = False
WHILE not valid:
       q8 = INPUT("Each of the following is a characteristic of a positive output gap
except")
    IF not validate(q8,"1234"):
          PRINT("Answer is not valid")
    ELSE:
           valid = True

IF q8 == "1":
      counter = counter+1
ELSE:
```

```
       PRINT("Incorrect")


PRINT("\n")
PRINT("1.GDP")
PRINT("2.Expected years of schooling")
PRINT("3.GNI per capita")
PRINT("4.Mean years of schooling")
valid = False
WHILE not valid:
        q9 = INPUT("Which of the following is not part of the HDI")
      IF not validate(q9,"1234"):
          PRINT("Answer is not valid")
       ELSE:
          valid = True
IF q9 == "1":
      counter = counter+1
ELSE:
       PRINT("Incorrect")



PRINT("\n")
PRINT("1.A reduction in the base rate of interest in the economy")
PRINT("2.An increase in import volumes")
PRINT("3.An Increase In Export Volumes")
PRINT("4.A reduction in inwards foreign investment")
valid = False
WHILE not valid:
       q10 = input("Which one of the following is likely to cause appreciation of a
country's currency?")
       IF not validate(q10,"1234"):
              PRINT("Answer is not valid")
        ELSE:
             valid = true
IF q10 == "3":
       counter = counter+1

ELSE:
        PRINT("Incorrect")



PRINT("\n")
PRINT("1.An ageing population")
PRINT("2.A rise in employment")
PRINT("3.Reduced spending on public services")
PRINT("4.A rise in the income tax rate")
valid = False
```

```
WHILE not valid:
       q11 = input("Which of the following is most likely to increase the size of the
fiscal deficit?")
       IF not validate(q11,"1234"):
            PRINT("Answer is not valid")
       ELSE:
              valid = True
IF q11 == "1":
        counter = counter+1
ELSE:
      PRINT("Incorrect")



PRINT("\n")
PRINT("1.AD is likely to rise")
PRINT("2.There would be downward pressure on real incomes")
PRINT("3.Import volumes may rise")
PRINT("4.UK goods and services will become less competitive")
valid = False
WHILE not valid:
            q12 = INPUT("Which one of the following is the likely consequence of a
rise in the unemployment rate?")
            IF not validate(q12,"1234"):
                 PRINT("Answer is not valid")
            ELSE:
                 valid = True
IF q12 == "2":
       counter = counter+1
ELSE:
       PRINT("Incorrect")


PRINT("\n")
PRINT("1.To meet the extra demand")
PRINT("2.To make profit")
PRINT("3.To benefit society")
PRINT("4.To use up unused resources")
valid = False
WHILE not valid:
       q13 = INPUT("What is the most likely reason for a firm to supply more when the
price of a good increases?")

        IF not validate(q13,"1234"):
               PRINT("answer is not valid")
        ELSE:
                valid = true
IF q13 == "2":
```

```
       counter = counter+1
ELSE:
       PRINT("Incorrect")


PRINT("\n")
PRINT("1.£1")
PRINT("2.£19")
PRINT("3.-£1")
PRINT("4.£29")
valid = False
WHILE not valid:
      q14 = INPUT("If total costs rise from £300 to £319 and average costs fall from
£30 to 29 how much is marginal cost?")
      IF not validate(q14,"1234"):
            PRINT("Answer is not valid")
       ELSE:
             valid = True
IF q14 == "2":
    counter = counter+1
ELSE:
    PRINT("Incorrect")



PRINT("\n")
PRINT("1.A shift to the right in the AD curve")
PRINT("2.A shift to the left in the short run AS curve")
PRINT("3.A shift to the right in the long run AS curve")
PRINT("4.A shift to the right in the sort run aggregate supply curve")
valid = False
WHILE not valid:
       q15 = INPUT("A fall in the cost of imported raw materials is likely to lead
to")
      IF not validate(q15,"1234"):
           PRINT("Answer is not valid")
       else:
           valid = True
IF q15 == "4.":
     counter = counter+1
ELSE:
PRINT("Incorrect")
PRINT("\n")



IF 0 <= counter <=3:
      PRINT("Don't worry you have plenty of time to revise and catch up, your score
```

```
was",counter)
ESELIF 3< counter <=6:
      PRINT("hehe, you're doing  well, keep it up, your score was",counter)
ELSEIF 6 < counter <=9:
      PRINT("Damn, keep working like this and you will get an A in that exam, your
score was,",counter)
ELSEIF 9 < counter <=12:
      PRINT("Uhh, you were so close to getting full marks, your score was",counter)
ELSEIF 12 < counter <=14:
      PRINT("Alright Economics geek, we get it now, you're the best, you're score
was",counter)
ELSEIF counter == "15":
      PRINT("Well, you my friend, got full marks, just go and take a day off")

Create()
menu()
```

This is the quiz that the student will take, they have different options to choose from. If the options that the user inputs are not valid then it will tell them and ask them until the option they input is within the options that are available. When the user get the answer right then the counter will increase, if they don't answer right then the counter stays the same, depending on how high the number stored in the counter is, it will output one of the following options shown above below the if statement.

```
DEF Revision():

        new = 2
        url= file:///D:/A%20level%20computing/Programming%20Project/Check.html
        webbrowser.open(url,new=new)
```

When one of the buttons are pressed for the students menu then it will redirect them to this HTML site.

```
 DEF results():

       with OPEN("Score.csv","r",newline="") as file:
              List = list(csv.reader(file))
              FOR row in List:
                   PRINT(row)

It will print out all the names and scores that are stored in the CSV file

    DEF last():

        root.destroy()
        login()
```

```
    root= Tk()
    root.geometry("250x250")
    root.title("Menu")
    root.configure(background='grey')



    b1=Button(root,text="Start Quiz",width=12,bg="grey",fg="white",command=quiz)
    b1.place(x=50,y=50)#Places button 1 in the given position

    b2=Button(root,text="Access Revision
Website",width=20,bg="grey",fg="white",command=Revision)
    b2.place(x=50,y=100)

    b3=Button(root,text="View Class Scores", width=20,bg="grey",fg="white",command
=results)
    b3.place(x=50,y=150)#Places button 1 in the given position

    b4=Button(root,text="Exit",width=12,bg="grey",fg="white",command=last)
    b4.place(x=50,y=200)#Places button 4 in the given position

    root.attributes("-topmost", True)
    root.mainloop()
```

This is the initial login system for the teacher or the student.

```
DEF menu2():
     DEF results():
           with open("Score.csv","r",newline="") as file:
                 List = LIST(csv.reader(file))
                 FOR row in List:
                       PRINT(row)



    root= Tk()
    root.geometry("250x250")
    root.title("Furhter Options")
    root.configure(background='grey')

    b1=Button(root,text="Open Score
Board",width=12,bg="grey",fg="white",command=results)
    b1.place(x=50,y=50)#Places button 1 in the given position

    b2=Button(root,text="Exit",width=12,bg="grey",fg="white",command=root.destroy)
    b2.place(x=50,y=150)#Places button 1 in the given position
```

```
    root.attributes("-topmost", True)
    root.mainloop()
```

This is to show the Further options for the teachers.

```
DEF login():

  root =Tk()#Creates GUI
  root.geometry("500x600")#This establishes how big the window opened will be
  root.title("Economics Quiz.")#Title of the window that has been opened


    def close():

        close = messagebox.askyesno("Economics Quiz","Confirm if you want to exit")

        if close == True:

            root.destroy()

            main()

        else:

            pass



    def login1():

        #Establish Connection

        with sqlite3.connect('students.db') as db:

            c = db.cursor()


        #Find user If there is any take proper action

        find_user = ('SELECT * FROM students WHERE username = ? and password = ?')

        c.execute(find_user,[(Username.get()),(Password.get())])

        result = c.fetchall()

        if result:
```

```
            if messagebox.askyesno("Question","Are you a student?") == True:
                root.destroy()
                menu()
            else:
                Username.get()
                root.destroy()
                registration()
        else:
            ms.showerror('Oops!','Username Not Found or Password does not match.')


    def Wclose():
        root.destroy()
        main()


    Username=StringVar()
    Password=StringVar()
    #Identity=StringVar()


    root.configure(background='grey')


    label1=Label(root,text="Login System",relief =
"solid",width=20,font=("arial",19,"bold"))
    label1.place(x=90,y=53)#Places Label 1 in the given position


    label2=Label(root,text="Username: ",width=10,font=("arial",10,"bold"))
    label2.place(x=90,y=130)#Places Label 2 in the given position


    label3=Label(root,text="Password: ",width=10,font=("arial",10,"bold"))
```

```
    label3.place(x=90,y=180)#Places Label 3 in the given position

    entry1=Entry(root,textvar=Username)
    entry1.place(x=185,y=130)#Places Entry 1 in the given position
    entry1.focus_force()

    entry2=Entry(root,show="*",textvar=Password)
    entry2.place(x=185,y=180)#Places Entry 2 in the given position

    b1=Button(root,text="Exit",width=12,bg="grey",fg="white",command=root.destroy)
    b1.place(x=200,y=350)#Places button 1 in the given position

    b2=Button(root,text="Log in",width=12,bg="grey",fg="white",command=login1)
    b2.place(x=200,y=300)

    root.attributes("-topmost", True)

    root.protocol('WM_DELETE_WINDOW', close)

    root.mainloop()

def log_in():

    log_in = ""
    while log_in!= "Y" and log_in!="N":
        log_in = input("Do you have an account? Y/N: ").upper()
    if log_in == "N":
```

```
        user_profile()
        print("Please log in...")
        login()
        main()
    elif log_in == "Y":
        print("Please log in...")
        login()
        main()


regex = '^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$'
def check(Email):
    if(re.search(regex,Email)):
        print("Valid Email")
    else:
        print("Invalid Email")


def user_profile():


    sign_up = ""
    while sign_up!= "Y" and sign_up!= "N":
        sign_up = input("Do you want to create an account? Y/N: ").upper()
        while sign_up == "N":
            if sign_up == "N":
                print("You have to create an account to participate in the quiz.")
                sign_up = input("Do you want to create an account? Y/N: ").upper()


    if sign_up == "Y":
```

```
FirstName = ""
while FirstName == "" or FirstName.isnumeric() == True:
    FirstName = input("Enter your first name: ")


Surname = ""
while Surname == "" or Surname.isnumeric() == True:
    Surname = input("Enter your surname: ")


DOB = ""
print("Dont leave any space between day, month and year")
while DOB.isnumeric() == False:
    DOB = input("What is your Date of Birth? ")




global Email
Email = ""
print("Please enter a valid email.")
while Email == "":
    Email = input("Enter your email: ")
    check(Email)



Username = ""
while Username == "":
    Username = input("Create a username: ")
```

```
            i=0
            c.execute('SELECT Username FROM students')
            result=c.fetchall()
            for i in range(len(result)):
                if result[i][0] == Username:
                    print("Your username is taken")
                while result[i][0] == Username:
                    Username = ""
                    while Username == "":
                        Username = input("Create a username: ")
                        if result[i][0] == Username:
                            print("Your username is taken")



        Password = ""
        print("Password should be longer than 6 characters but less than 10")
        while len(Password) <6 or len(Password)>10:
            Password = input("Create a password: ")


        FirstName = FirstName
        Surname = Surname
        DOB = DOB
        Email = Email
        Username = Username
        Password = Password
        c.execute ("INSERT INTO students (FirstName, Surname, DOB, Email, Username,
Password) VALUES (?,?,?,?,?,?)",
                   (FirstName, Surname, DOB, Email, Username, Password))
```

```
        conn.commit()


        #display_menu()
    else:
        exit()


def main():
    log_in()
    user_profile()


main()
```

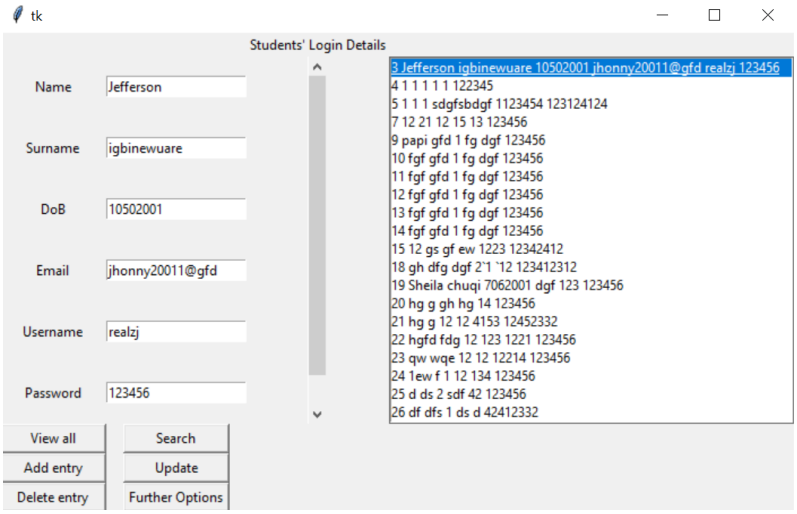
PSEDUCODE OF PROGRAM PART II

```
IMPORT SQLITE3

DEF connect():
conn=sqlite3.connect("students.db")
cur=conn.cursor()
conn.commit()
conn.close()

DEF insert(FirstName,Surname,DOB,Email,Username,Password):
conn=sqlite3.CONNECT("students.db")
cur=conn.cursor()
cur.execute("INSERT INTO students VALUES (NULL,
?,?,?,?,?,?)",(FirstName,Surname,DOB,Email,Username,Password))
conn.commit()
conn.close()
view()

DEF view():
conn=sqlite3.CONNECT("students.db")
cur=conn.cursor()
cur.execute("SELECT * FROM students")
row=cur.fetchall()
```

```
conn.close()
RETURN row

DEF search(FirstName="",Surname="",DOB="",Email="",Username="",Password=""):
conn=sqlite3.connect("students.db")
cur=conn.cursor()
cur.execute("SELECT * FROM students WHERE FirstName=? OR Surname=? OR DOB=?  OR
Email=?  OR  Username=?  OR
Password=?",(FirstName,Surname,DOB,Email,Username,Password))
row=cur.fetchall()
conn.close()
RETURN row

DEF delete(id):
conn=sqlite3.connect("students.db")
cur=conn.cursor()
cur.execute("DELETE FROM students  where id=?",(id,))
conn.commit()
conn.close()

DEF update(id,FirstName,Surname,DOB,Email,Username,Password):
conn=sqlite3.CONNECT("students.db")
cur=conn.cursor()
cur.execute("UPDATE students SET FirstName=? ,Surname=? , DOB=? ,  Email=? ,
Username=? , Password=? where
id=?",(FirstName,Surname,DOB,Email,Username,Password,id))
conn.commit()
conn.close()

connect()
```

There is a reason for this, there is 2 programs because I have added a SQLite backend, this means that the user is able to manipulate a database within the Tkinter GUI, this is done by linking the SQL to Tkinter and connecting both programs. This is done by importing the name of the back end into the main program.

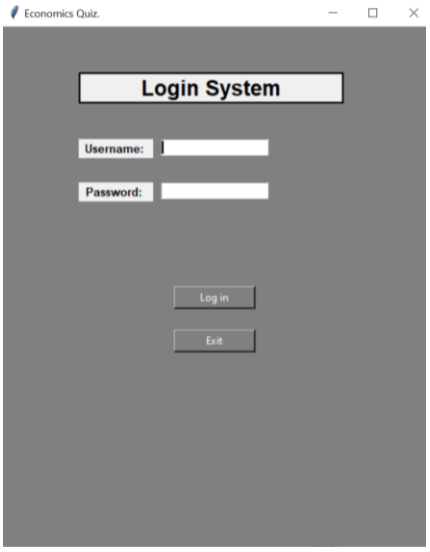
In the second program there is different functions which are all being used, for example in the view function, it calls all the data that is inside the database and displays them in an extended entry; it also shows each thing in other entries, this avoids confusion.



Each entry has a different header, this is to show the information that is shown on the right. All the buttons are shown below are linked to the functions in the back program, except the further options that I implemented in inside the normal one. What happens is that I established a connection between both main and back program and call the functions inside other functions.

UPDATES TO THE GUI AND HTML

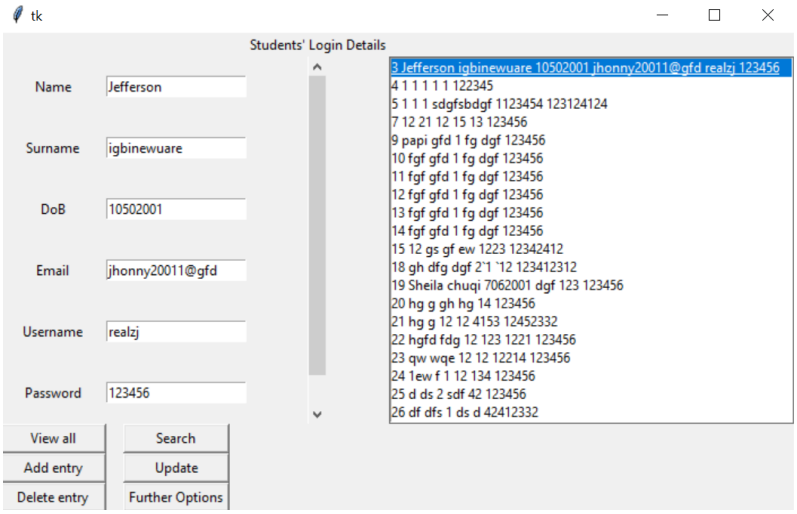
One of the user requirements was to change the background colour for the GUI, they wanted me to do this because it looked bad and robotic.



During this change I went and spoke to some of the economics students and asked them if this was an improvement to the previous tkinter window and they gave me positive comments.

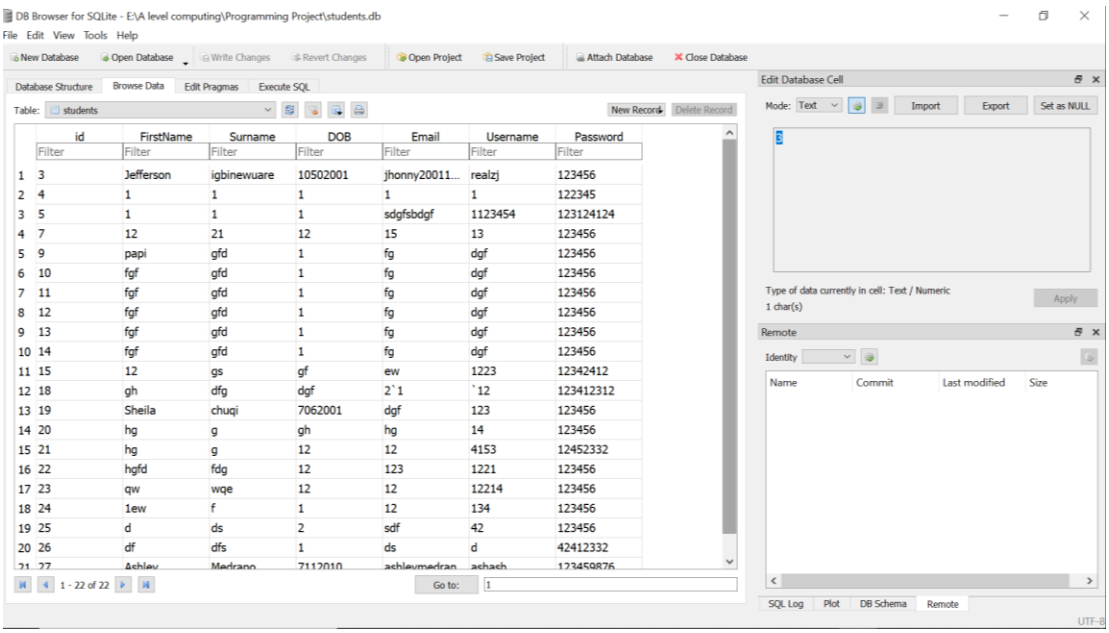
This version also has buttons that both work and take them to the next menu depending if they are the teacher or student. I also implemented a message.

In this prototype I went on and created a teacher admin area, this part of Tkinter allowed them to manipulate the SQL database directly without having to open or installing any software unless they wanted to.

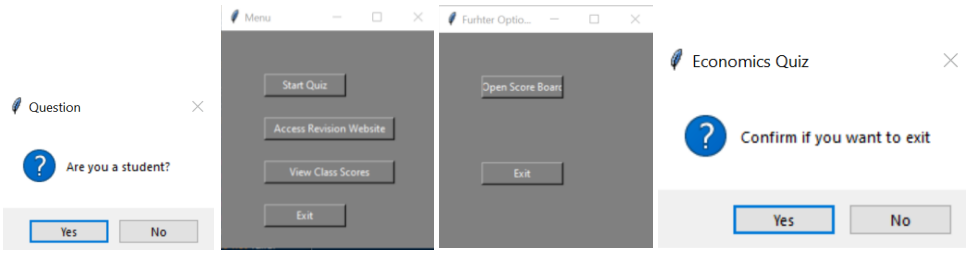


This is the admin section for the teachers. They have different options which will allow them to do many things such as: looking at student details, they can also create a profile for a student if they wished to, they can delete any irrelevant fields, they can search any student inside the database in the Entries which are provided and the update button which allows the admin to refresh the database and see new changes made to the database.

The last option displays a little menu which allows the teacher to see the students results.



As shown above the database is still available if the teachers want to access it, I also added more columns into the database and allowed the user to create a Username and a Password.



The pop up are there to help with the program, if the user clicks that they are a student it will show them the following menu with the options given to them. Each of the buttons work and have their function. If the user is entering as a teacher then click no, it will recirect them to the admin register where they can see all the stuent that are in the database. In that same window there is a option called Further options, this will output the further options menu shown, like showing them the scoreboard, there will be other options which will be finished by the end of the prototype 3 which displays the students in alphabetical order and the other option to display it in descending order. However, this could be a limitation for either lack of python techniques or time limitation.

SUBROUTINE/ STRUCTURE TABLE UPDATED

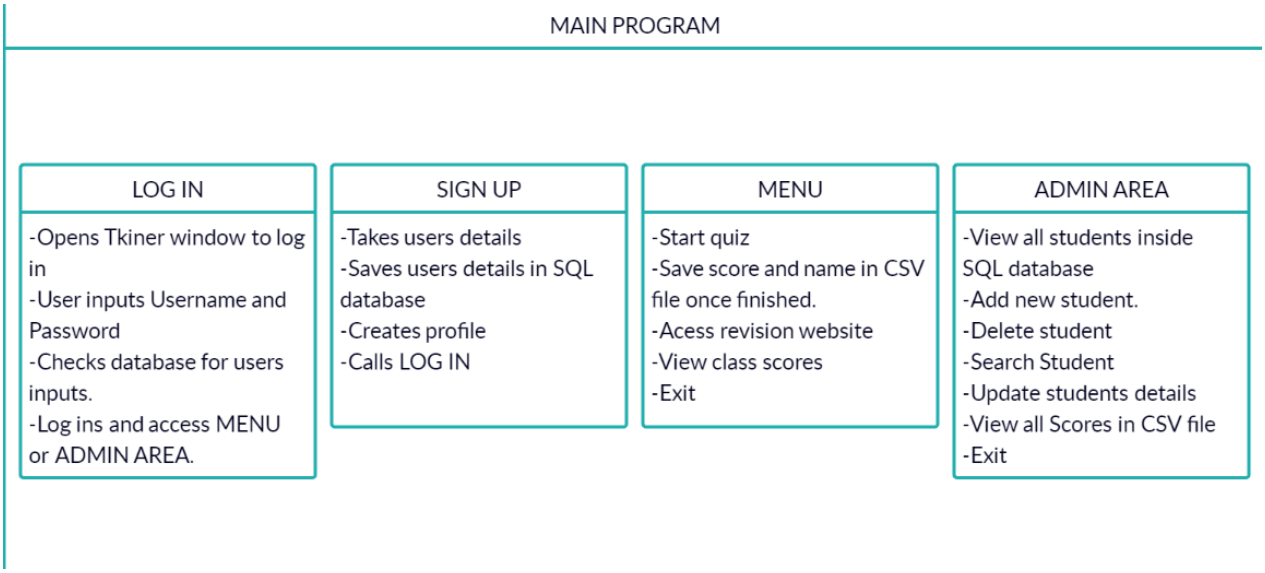
Variable/Structure Name	Purpose	Data Type
Validate	This is used to validate options further on, the quiz will use in order to validate the user's input.	Subroutine
Get_selected_row	This is part of the GUI which gets students details and display them in each entry.	Subroutine
View_command	This is one of the functions inside the GUI which prints out all the data inside the SQL database	Subroutine
Search_command	This is one of the functions inside the GUI which allows the teacher to type any information inside the entries, click search and all related data which is wanted by the admin.	Subroutine
Add_command	This is one of the functions inside the GUI which allows the admin to add a student into the database, allowing them to create a profile for those who haven't yet.	Subroutine
Delete_command	This is one of the functions inside the GUI which allows the admin to delete any student	Subroutine

	from the database for whatever reason, it does this by traversing and finding the ID assigned to the student.	
Update_command	This is one of the functions inside the GUI which allows the admin to change data which might be wrong, it does this by also traversing and finding the Id linked to that specific student.	Subroutine
Registration	This is responsible for the build-up of the admin area, it has all the labels, entries, buttons, commands, the grid inside the admin area, the configuration and the layout of everything inside the Tkinter window.	Subroutine
Menu	This subroutine is special because this has more subroutines inside them because it creates the menu, this has the quiz, creates the CSV file, opens up the HTML file, shows the results inside the CSV file and its also the build-up for the menu.	Subroutine
Create	Creates a CSV if there isn't one yet, inputs all the data that there is needed inside. The name and the score that each student got.	Subroutine
Quiz	This is what is responsible for outputting the quiz to the user and takes care of counting the score	Subroutine

Revision	It opens the Revision website.	Subroutine
Results	It shows all the results inside the CSV file if there is any.	Subroutine
Last	When the user exits the menu, it will output the Log in window again.	Subroutine
Menu2	This is the second menu for GUI for the admin area, it's for the further options.	Subroutine
results	Shows the results in CSV file.	Subroutine
Login	Creates login Tkinter window	Subroutine
Close	This displays the message of confirmation if the user really wants to exit the login window or not.	Subroutine
Login1	This is to validate the username and the password is inside the SQL database, if the user is found then it will go on and show the menu. However, if the username or password doesn't match the records inside the database.	Subroutine
Wclose	When the user closes the window, it will close the Tkinter and ask if the user has an account again.	Subroutine
Log_in	This is responsible for asking if the user if they have an account or not, it must be validated.	Subroutine

check	This is to check the email further on when the user is creating an account.	Subroutine
User_profile	This is to get all the details of the user.	Subroutine
Main	This is what calls every subroutine in order.	Subroutine

DIAGRAM FOR PROTOTYPE 3



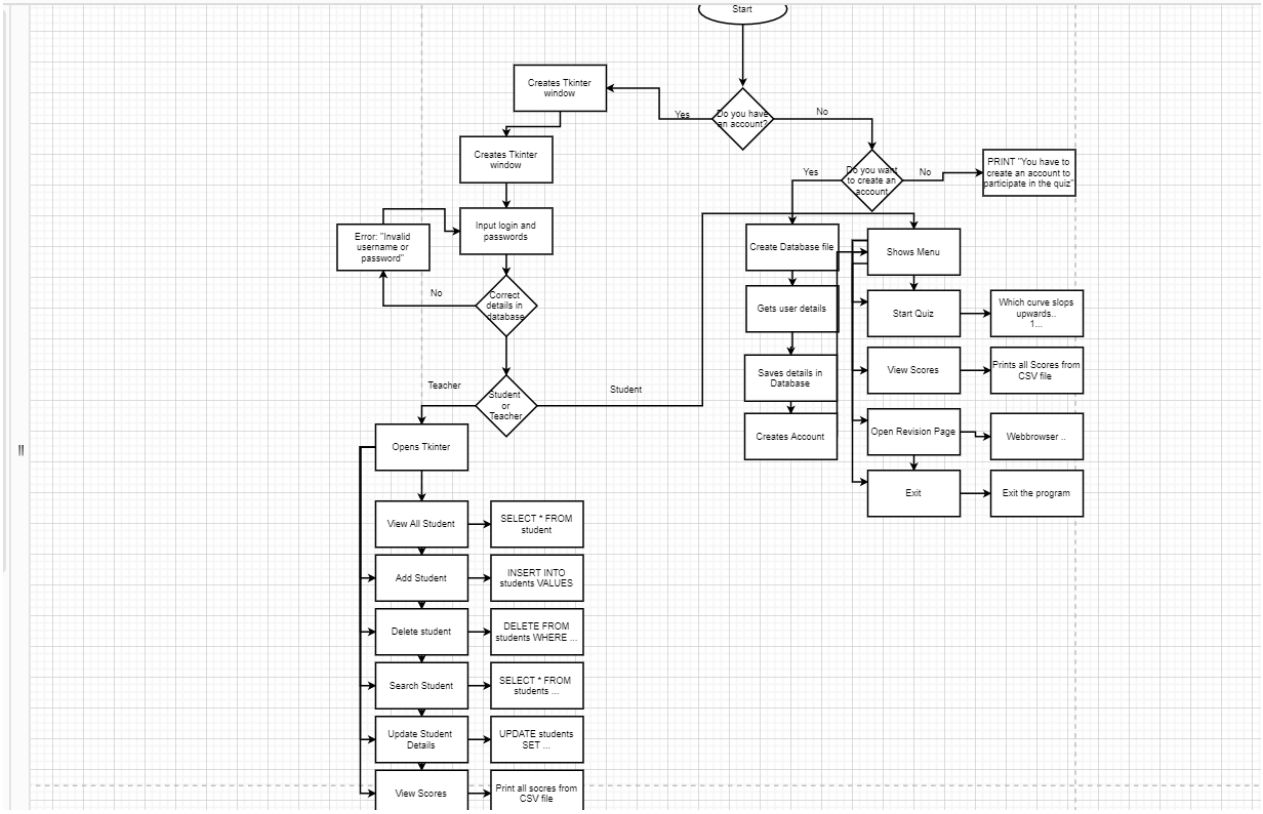
This is the final and updated version of the modular diagram, it has had a lot of changes since prototype 1 because it has finally connected to a database, everything is functional all the functions within the program make all these main 4 functions inside the program run properly and accordingly. The development for this has improved even more in this prototype as the user can see the menu in another Tkinter window instead of the python interface and it has finally been fully functional. The admin area has also been implemented into the program which allows the admin to do many different things to the database and the information it has inside. This is the last modular diagram because everything has been achieved since prototype 1. The quiz is taken inside the python interface which doesn't seem to be a problem.

DATABASE DIAGRAM FOR PROTOTYPE 3

USERS	
username	String
password	String

Scores	
ID	Int
Score	varchar

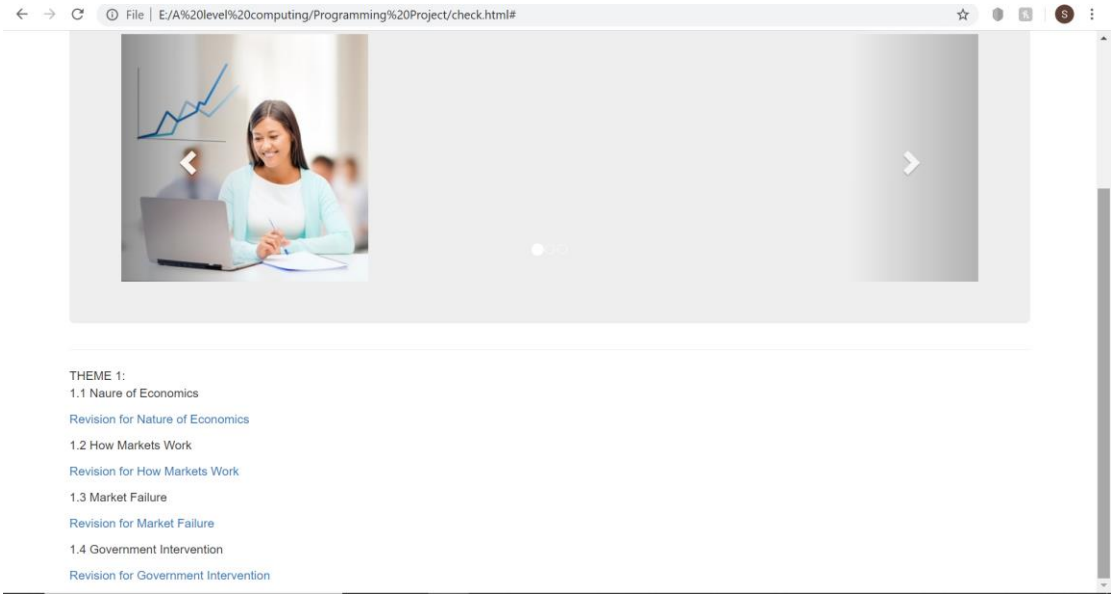
FLOWCHART FOR PROTOTYPE 3



This is the final version of the flowchart; this shows the student menu and the admin area and the functions inside them. The menu for the student has been updated and the update details has not been implemented for the student, it has been implemented for the teacher.

HTML

I have also worked on the HTML, the revision website for the economics revision. I have added a couple of extra features which makes the website look much more professional and user-friendly.



There are 3 pictures that are displayed automatically, it is active and show statistics which can intrigue students. I have done this with the help of Atom, an application that helps me build with HTML, CSS and Java Script. The advantage for this is that JavaScript is a functional language and lets me add more functionalities to the page than previously.

```
<!DOCTYPE html>
<html>
<head>
  <!-- navbar code -->
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/
<!-- Carasouel -->
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>


  <link rel="stylesheet" type="text/css" href="css/style.css">

  <meta charset="utf-8">


  <title> Revision Quiz </title>
</head>
<body>
  <nav class="navbar navbar-default">
    <div class="container-fluid">
      <div class="navbar-header">

        <a class="navbar-brand" href="#"><Center><strong>Revision Quiz</strong></Center</a>
      </div>
      <p class="nav navbar-nav">
        <p class="active"><a href="#">Home </a>
        </p>
      </p>
    </div>
  </nav>

  <h1><center><strong>Revision Quiz</strong></center></h1>

  <p style="text-align: center; font-size: 30px;"><em><strong>This page is here to help you with your eoconomics quiz</strong></em></p>
```

```
<head>
  <div id="myCarousel" class="carousel slide" data-ride="carousel">
    <!-- Indicators -->
    <ol class="carousel-indicators">
      <li data-target="#myCarousel" data-slide-to="0" class="active"></li>
      <li data-target="#myCarousel" data-slide-to="1"></li>
      <li data-target="#myCarousel" data-slide-to="2"></li>
    </ol>
    <!-- Wrapper for slides -->
              <div class="carousel-inner">
                <div class="item active">
                  <img src=https://content.ool.co.uk/wp-content/uploads/Economics-A-level-CTA-image.jpg>
                </div>

                <div class="item">
                  <img src=https://www.pmtcourses.com/wp-content/uploads/2019/10/economics-calculator-pen-grey.jpg>
                </div>

                <div class="item">
                  <img src="https://fee.org/media/25770/upward_trend_charts_economics_mini.jpg?center=0.6375,0.51333333333333331&mode=crop&width=1920&r
                </div>
              </div>

              <!-- Left and right controls -->
              <a class="left carousel-control" href="#myCarousel" data-slide="prev">
                <span class="glyphicon glyphicon-chevron-left"></span>
                <span class="sr-only">Previous</span>
              </a>
              <a class="right carousel-control" href="#myCarousel" data-slide="next">
                <span class="glyphicon glyphicon-chevron-right"></span>
                <span class="sr-only">Next</span>
              </a>

            </a>
          </div>
        </div>


    <hr>

    THEME 1:

    <p>1.1 Naure of Economics</p>
    <p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.1.%20Nature%20of%20Economics.pdf">Rev
    <p>1.2 How Markets Work</p>
    <p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.2.%20How%20Markets%20Work.pdf">Revisi
    <p>1.3 Market Failure</p>
    <p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.3.%20Market%20Failure.pdf">Revision f
    <p>1.4 Government Intervention</p>
    <p><a href="https://pmt.physicsandmathstutor.com/download/Economics/A-level/Notes/Edexcel-A/Theme-1/Summary/1.4.%20Government%20Intervention.pdf">R

    <body>

    </body>
    </p>


    </body>
    </html>
```

This is the code that shows the development, it shows the changes that have been made since prototype 2. For example, now there is the carousel which is active and can be changed from left to right, it shows different pictures that might intrigue the students. There is no links attached to the pictures, but they are automatic and goes to the next picture automatically.

TEST PLAN FOR PROTOTYPE 3 BLACK BOX

For this prototype I am going to be testing all aspects in the program, the buttons inside the Tkinter, the entries and the validation it must take. This is important because it is the final version.

Entry	Expected outcome
Create another file that only displays the name of the student and their score.	Opens a window that displays the student's names and their scores.
User should be able to access the revision site.	Opens finished HTML file when the user wishes to.

The student shouldn't have access to the teacher's system.	There should be a specific ID printed that can be given to the teacher so only they have access to the Admin Area
Make the test challenging	The quiz is challenging for A level students, because it links them to Theme 1 and 2.
Validate all inputs.	String variables should only allow strings and not numbers.
Validate Tkinter inputs	Traverse the database and let the user log in if the details inside the entries meet the ones in the database.
Make the menu inside the Tkinter window instead of the python interface	The program will open a Tkinter window once the user logs in.
Update the revision website.	Final version of the HTML.
Fully functional menu	All options in the menu to be responsive to the user.
Login Button	To log in or not allow the user to enter the if the credentials are not right.
Close button	This should display an error message asking if the user really want to close the program
Email validation	It should validate emails that have the format of a normal email and iterate if it doesn't meet that requirement.
Teacher ID	The teacher should be given their unique ID in order to log in and not mix students and teachers in the database.
Tkinter priority	For the Tkinter GUI to appear in front of the Python interface, instead of being behind.
Counter	This should be incremented each time the user answers a question right.

CSV file to save score and names	The names and scores should be saved underneath each other. The name should be in column 1 and the score in column 2.
View Scores	The scores inside the CSV file should be printed out to show the user the scores.
Feedback	The user will get feedback once they have finished the quiz in order to improve.
Teachers area Tkinter entries validation	The Tkinter entries should be validated so they can't add empty files in the database.
Teachers area Tkinter Buttons	The teacher will have different types of options and they will be working.
Scrollbar inside Tkinter	If the database is too big, the teacher will use this to scroll down to see further students.
Further options menu	Shows more options inside the database
Display Score board in descending order	Print the names and the scores depending in the score the students got
Display Score board in alphabetical order	Print the names and the scores depending on their names
Exit button	Exits the Tkinter and goes back to the python interface
Hide Password	The password will show "***" when the user enters a character

TEST PLAN FOR PROTOTYPE 3 WHITEBOX

Test	Expected Process
Creation of CSV file	It will create a CSV file if one doesn't exist already
Password validation	It will validate how long the password is
Username validation	It will validate if the username has been taken already

Username and password during login in	SQL command to traverse the database to find the password and username
Appending into arrays	Creation of 2 arrays in order to append a name in one and the score in the other
Appending into the CSV file	The arrays will append their values into the CSV file
SQL command	this will be used inside the admin area
GUI for admin area	If the user logs in as a teacher, the GUI will come up
Counter validation	The counter can only increase if the answer is correct.
Range check for counter	The counter will be validated at the end of the quiz
Scores inside CSV	They will be printed out along with the user's name
Priority GUI	The GUI will come first after the function is called.
Hidden Password	The way the password is shown when typed inside the Tkinter changes to "***"
Email validation	Validates email and outputs if its valid or not.

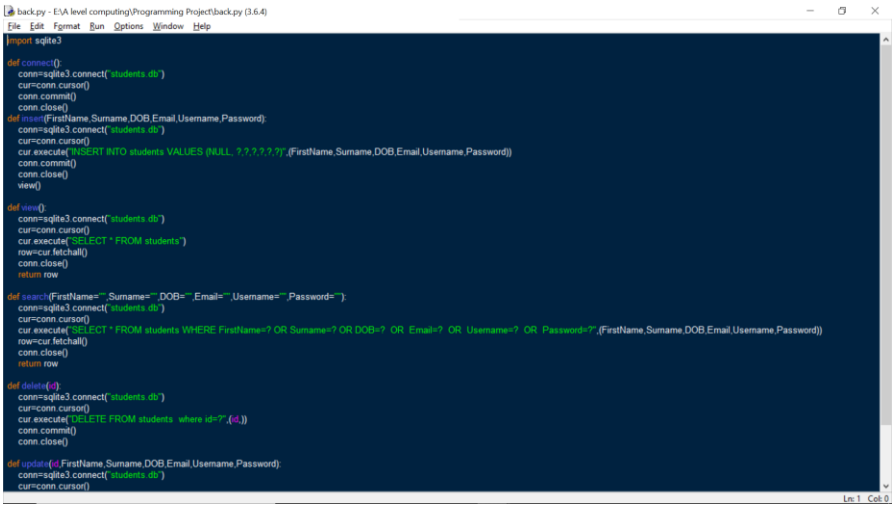
TEST PLAN FOR EVALUATION

Create another file that only displays the name of the student and their score.	Another file will be opened when the user wants to see their results
User should be able to access the revision site.	The user must have access to the site if the user clicks on that option
The student shouldn't have access to the teacher's system.	There should be separate login and passwords that allows the teacher to access the admin area
Make the test challenging	Range of questions from theme 1 and theme 2
Validate all inputs.	Variables should only allow inputs which are dew

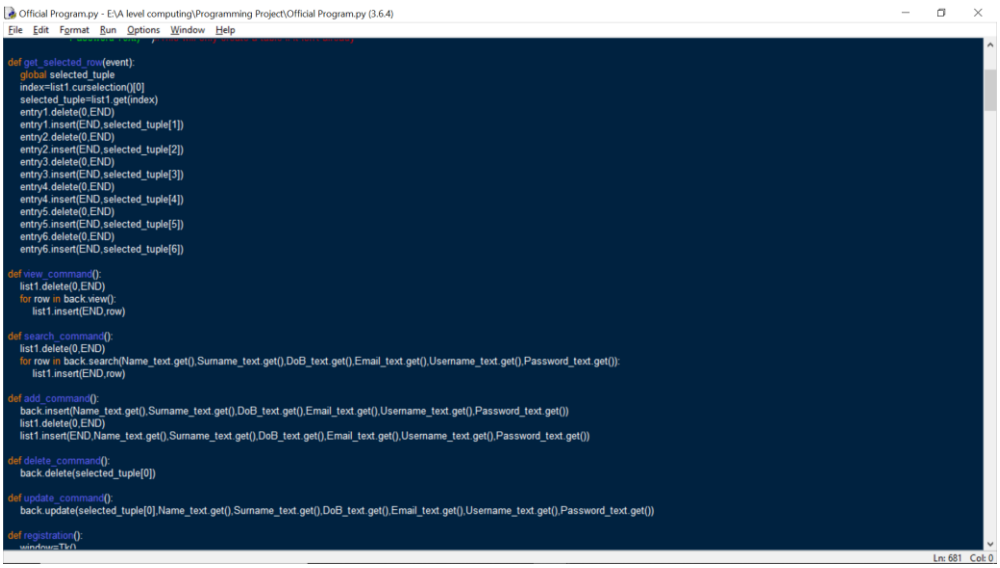
Validate Tkinter inputs	Not allow empty spaces in Tkinter.
Make the menu inside the Tkinter window instead of the python interface	Tkinter menu
Update the revision website.	To complete the HTML version of the website
Fully functional menu	All options in menu to respond.

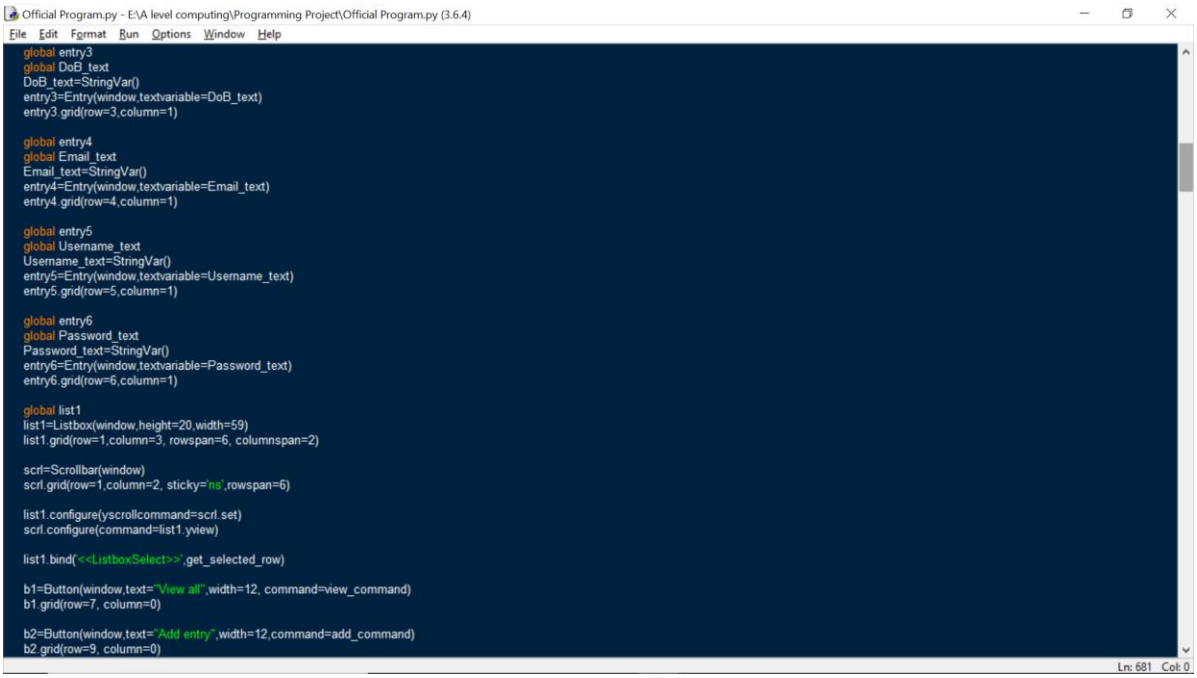
DEVELOPMENT

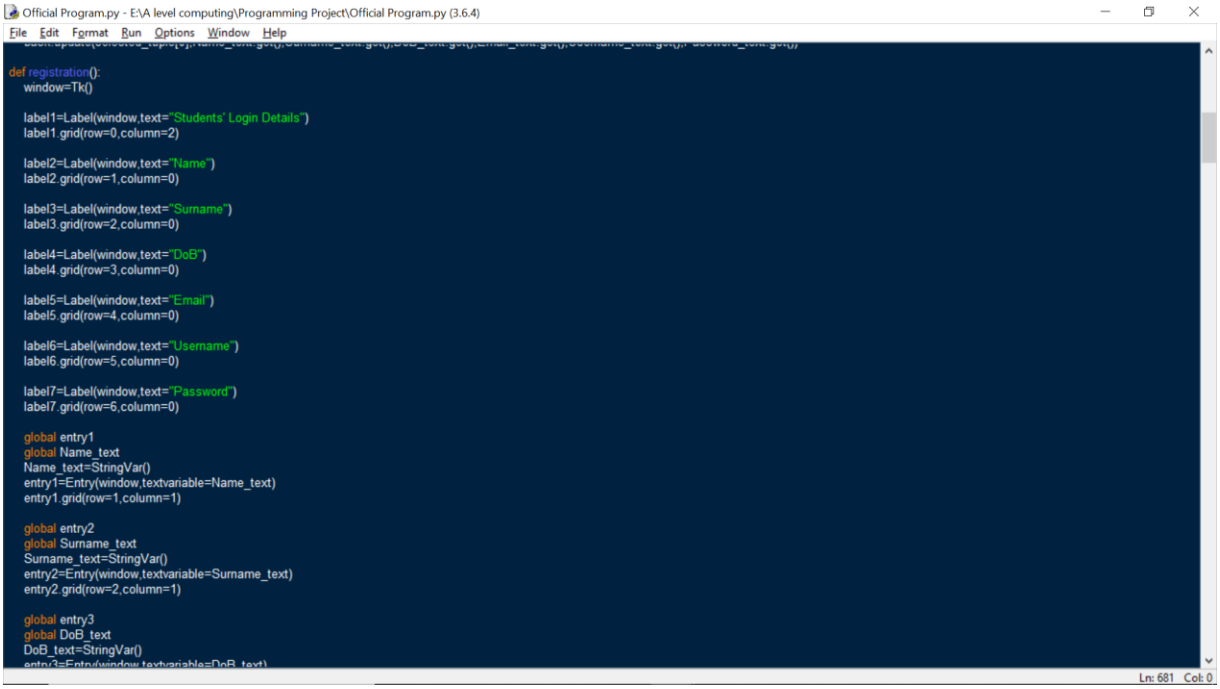
Most of the development for prototype 3 was creating the Admin part of the program. So creating the backend, connecting to the SQL database and trying to figure out how everything will be linked to each other in order to make it work. At the beginning I had problems, because I decided to put it a subroutine which variables can only be accessed within each subroutine, it took me a while to realise that I had to globalise a few variables linked to the GUI in order to make it work. This problem came up in the main program.

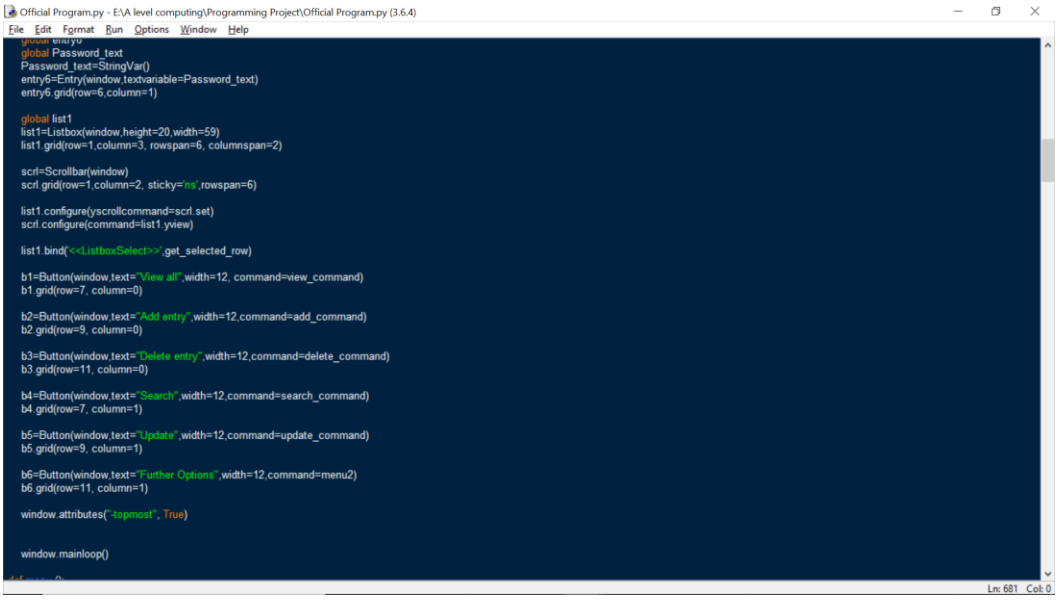


There is always a connection set in each of the subroutines this is because a connection cannot be set for all subroutines previously. The first subroutine creates the main connection that will be called at the end of the program to link both main and back program. Also, at the beginning I didn't create an ID for each of the user that register, so the admin wasn't be able to delete or update the student details because it didn't have a specific key to get. However, I managed to fix this by creating another column for each user, called it ID, this is what helped me to update or delete the student's details.


Official Program.py - E:\A level computing\Programming Project\Official Program.py (3.6.4)
File Edit Format Run Options Window Help
def get_selected_row(event):
global selected_tuple
index=list1.curselection()[0]
selected_tuple=list1.get(index)
entry1.delete(0,END)
entry1.insert(END,selected_tuple[1])
entry2.delete(0,END)
entry2.insert(END,selected_tuple[2])
entry3.delete(0,END)
entry3.insert(END,selected_tuple[3])
entry4.delete(0,END)
entry4.insert(END,selected_tuple[4])
entry5.delete(0,END)
entry5.insert(END,selected_tuple[5])
entry6.delete(0,END)
entry6.insert(END,selected_tuple[6])
def view_command():
list1.delete(0,END)
for row in back.view():
list1.insert(END,row)
def search_command():
list1.delete(0,END)
for row in back.search(Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get()):
list1.insert(END,row)
def add_command():
back.insert(Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get())
list1.delete(0,END)
list1.insert(END,Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get())
def delete_command():
back.delete(selected_tuple[0])
def update_command():
back.update(selected_tuple[0],Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get())
def registration():
Ln: 681 Col: 0


Official Program.py - E:\A level computing\Programming Project\Official Program.py (3.6.4)
File Edit Format Run Options Window Help
global entry3
global DoB_text
DoB_text=StringVar()
entry3=Entry(window,textvariable=DoB_text)
entry3.grid(row=3,column=1)
global entry4
global Email_text
Email_text=StringVar()
entry4=Entry(window,textvariable=Email_text)
entry4.grid(row=4,column=1)
global entry5
global Username_text
Username_text=StringVar()
entry5=Entry(window,textvariable=Username_text)
entry5.grid(row=5,column=1)
global entry6
global Password_text
Password_text=StringVar()
entry6=Entry(window,textvariable=Password_text)
entry6.grid(row=6,column=1)
global list1
list1=Listbox(window,height=20,width=59)
list1.grid(row=1,column=3, rowspan=6, columnspan=2)
scrl=Scrollbar(window)
scrl.grid(row=1,column=2, sticky='ns',rowspan=6)
list1.configure(yscrollcommand=scrl.set)
scrl.configure(command=list1.yview)
list1.bind('<<ListboxSelect>>',get_selected_row)
b1=Button(window,text="View all",width=12, command=view_command)
b1.grid(row=7, column=0)
b2=Button(window,text="Add entry",width=12,command=add_command)
b2.grid(row=9, column=0)
Ln: 681 Col: 0

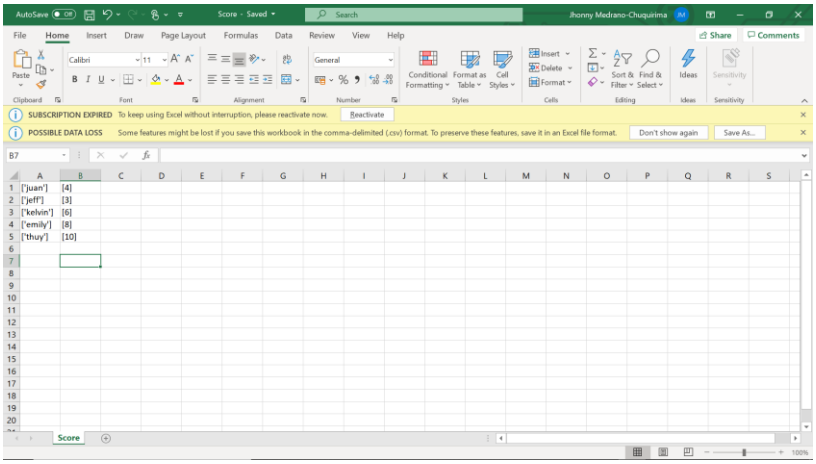




The code that is shown above is the main program, this program links to the back program. Each of the options are also inside a subroutine, they can only be accessed with the selected tuple which had to be globalised so it can communicate with the other sub routines and perform what it's meant to do. Apart from the selected tuple being globalised the entries also had to be globalised otherwise when the user tried to input a new student into the database it will not add it, the buttons didn't have a functionality unless the entries were globalised which makes sense because the entry was inside a subroutine which didn't let the backend access it.

```
def Create():

    Names = []
    Scores = []
    Scores.append(counter)
    Name = ""
    while Name == "" or Name.isnumeric() == True:
        Name = input("Enter your first name: ")
    Names.append(Name)
    Score = open("Score.csv","a+")
    with open("Score.csv","a", newline="") as file:
        writer = csv.writer(file)
        writer.writerow((Names, Scores))
```

There is also another part of development I had to do. I created a code that creates an empty CSV file which was then able to save the students names and the score they get. This made sure to validate the name, to not be a numeric value or to not be an empty space. The newline = “” made sure that there were no rows with an empty field. This was a problem at the beginning because it did what it was meant to do which was save the name and the score of the student, but there was a problem, it left a row empty and saved the next student in the next row which made it look messy.

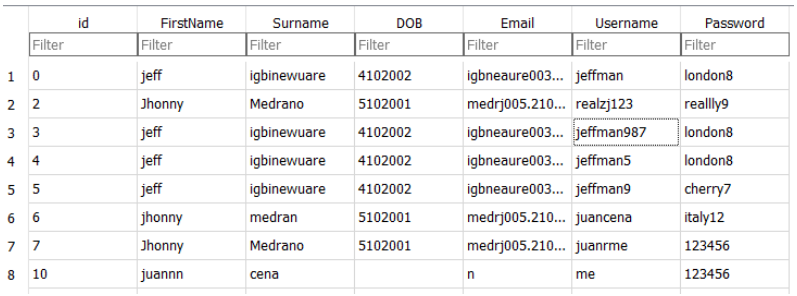
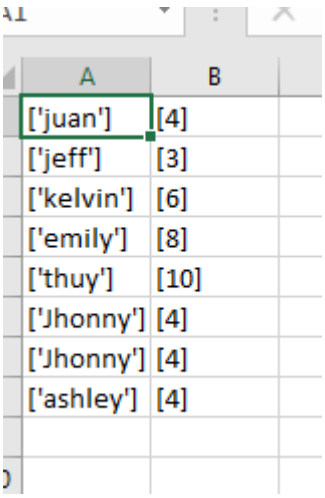
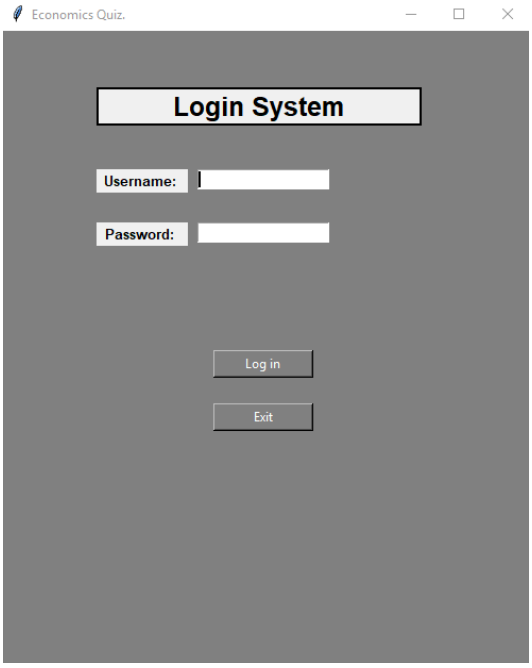


The reason it has brackets and quotation marks around the name is because the data is being appended from an array.

This is what the CSV file looks like, it saves the name in Column A and the scores in Column B, right now I am trying to display the scores in terms of highest to lowest or according to alphabetical order for the names.

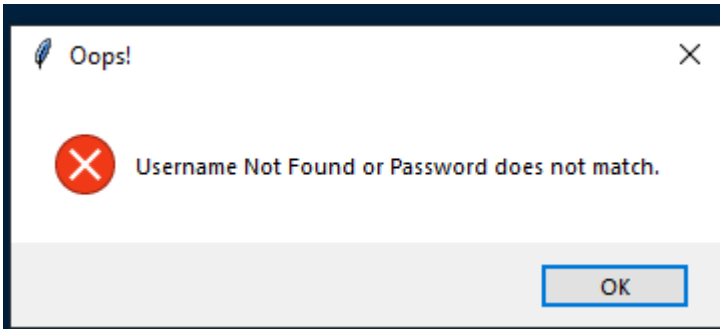
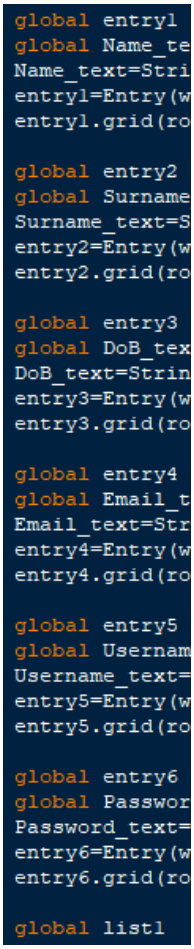
PROGRAMMING TEHCNIQUES USED

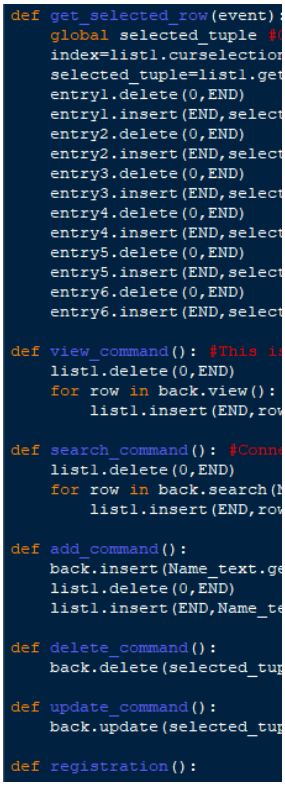
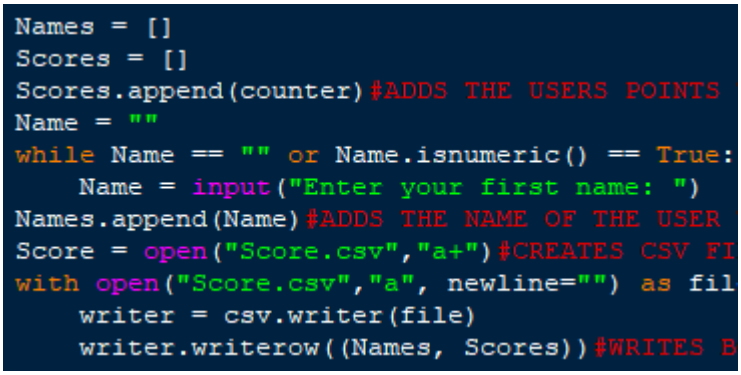
Techniques	**Definition**	**Evidence**

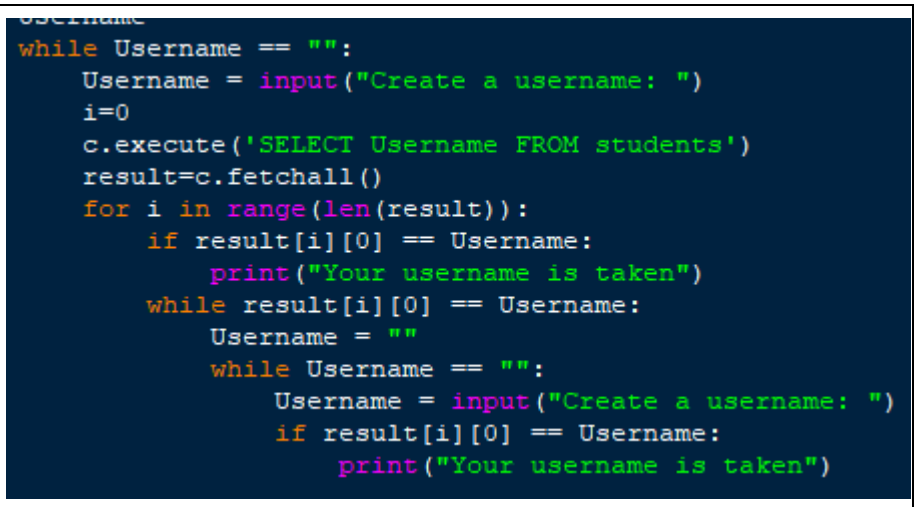
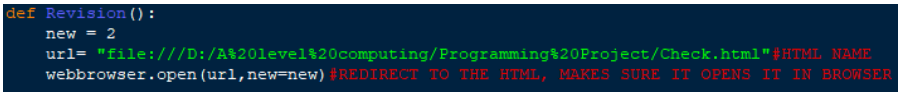
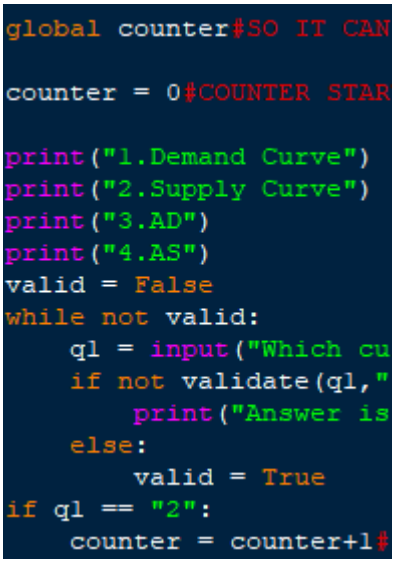
Database	Sets of data which is held in a computer, it can be accessed in various ways.	
CSV file	It's a delimited text file. Uses commas to separate values and each line is a data record.	
Tkinter GUI	An interface to the Tk GUI toolkit and is python's GUI.	

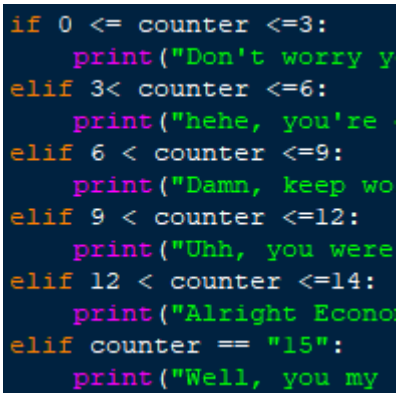
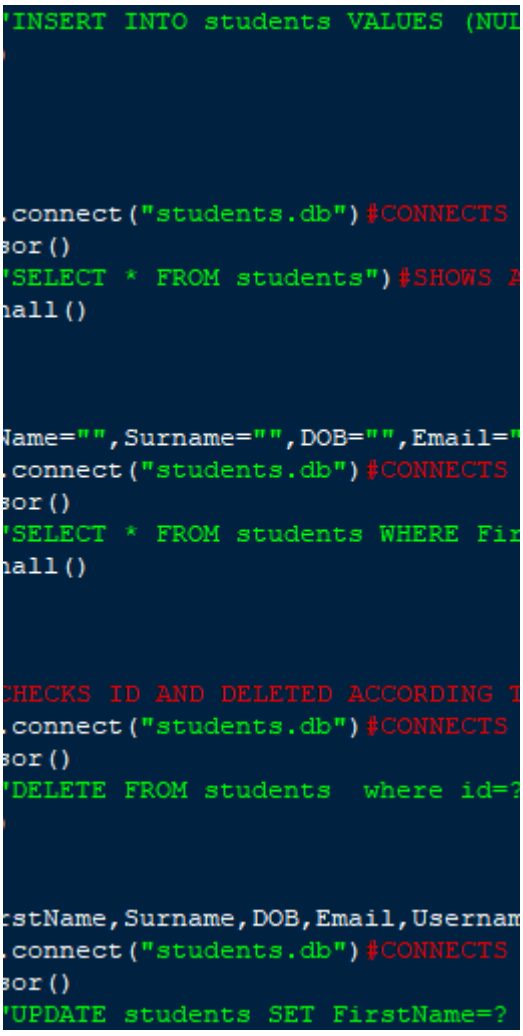
	id	FirstName	Surname	DOB	Email	Username	Password
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	0	jeff	igbinewuare	4102002	igbneaure003...	jeffman	london8
2	2	Jhonny	Medrano	5102001	medrj005.210...	realzj123	realllly9
3	3	jeff	igbinewuare	4102002	igbneaure003...	jeffman987	london8
4	4	jeff	igbinewuare	4102002	igbneaure003...	jeffman5	london8
5	5	jeff	igbinewuare	4102002	igbneaure003...	jeffman9	cherry7
6	6	jhonny	medran	5102001	medrj005.210...	juancena	italy12
7	7	Jhonny	Medrano	5102001	medrj005.210...	juanrme	123456
8	10	juannn	cena		n	me	123456

	A	B
	['juan']	[4]
	['jeff']	[3]
	['kelvin']	[6]
	['emily']	[8]
	['thuy']	[10]
	['Jhonny']	[4]
	['Jhonny']	[4]
	['ashley']	[4]

MessageBox inside Tkinter	A module used to display a message box inside python applications.	
Global variables	A global variable is a variable that can be accessed anywhere in the program.	

Subroutines	Set of instructions which perform a specific task.	
Array data structure	It's a data structure that consists of a collection of elements.	

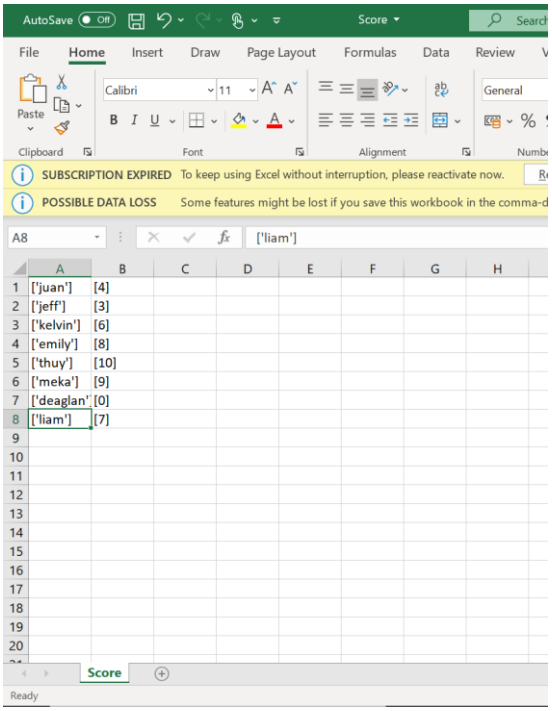
Iteration	The repetition of a process	
Webbrowser library	Module that provides a high-level interface which allows web-based documents to be displayed.	
Counter	A variable which stores the value of a number and will increase if the program tells it to.	

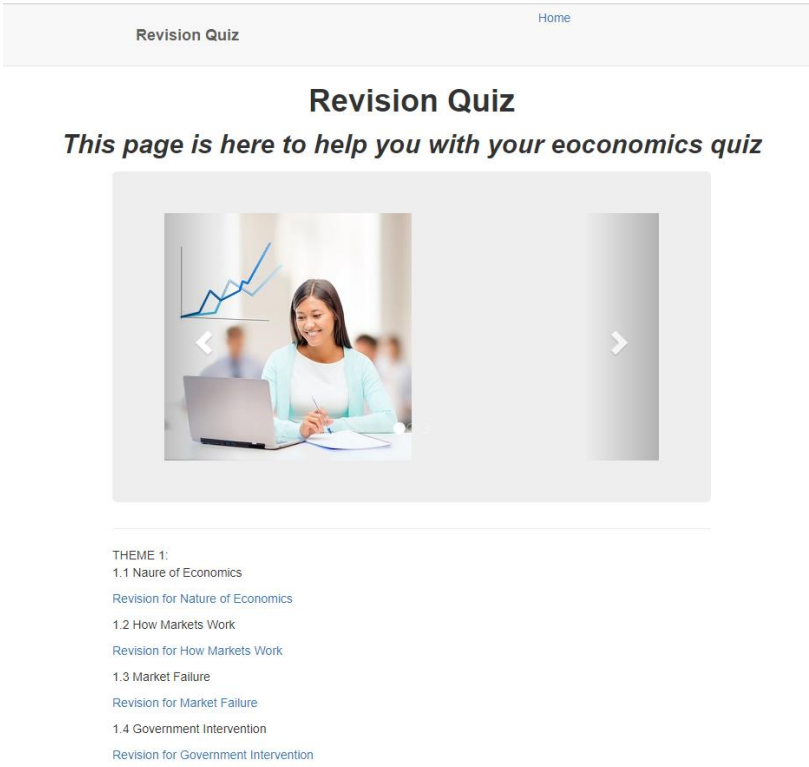
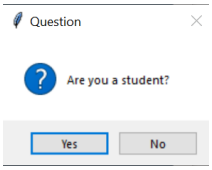
Range check	It's a validation technique. Used to check the value of data and see if its between a certain range.	
SQL commands	They are commands coded into SQL statements which communicate with the database to perform specific tasks inside the database.	

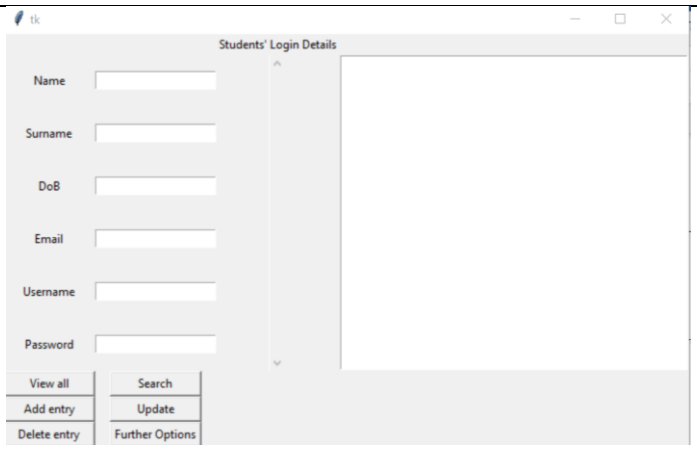
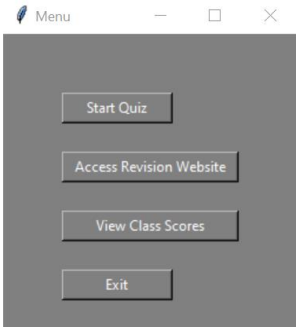
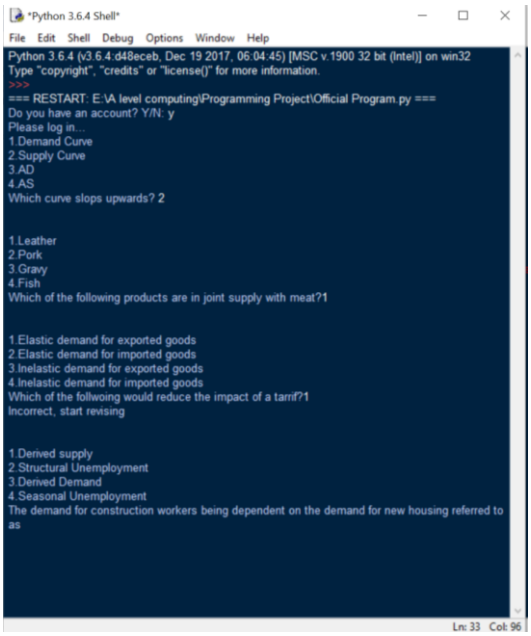
Regex validation		`regex = '^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$'`
		`def check(Email):`
		`  if(re.search(regex,Email)):`
		`  print("Valid Email")`
		`  else:`
		`  print("Invalid Email")`

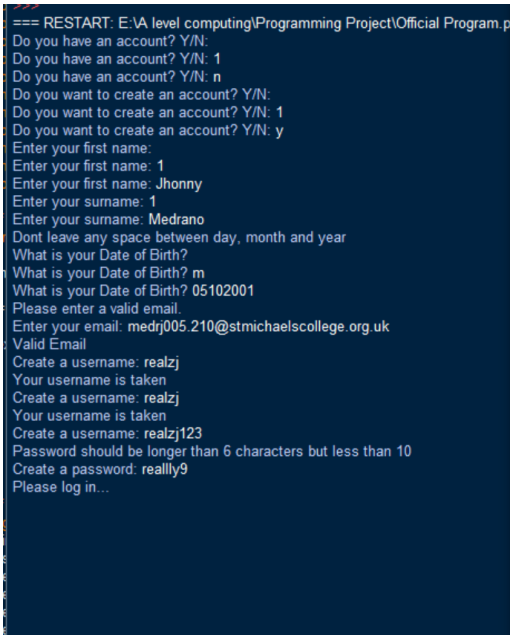
TESTING

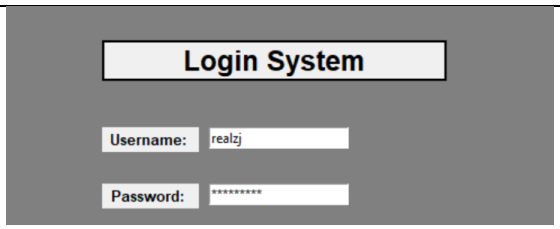
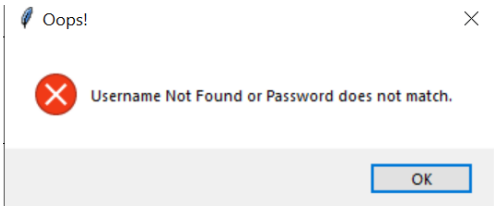
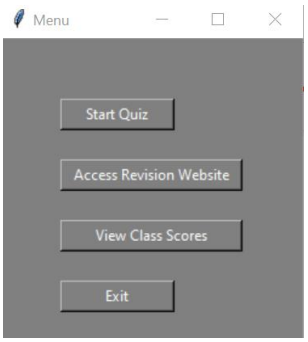
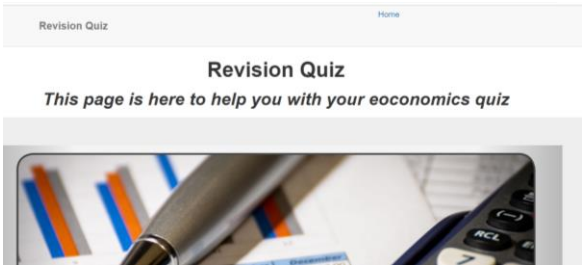
BLACKBOX TESTING

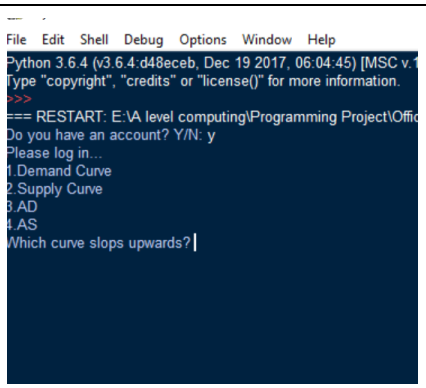
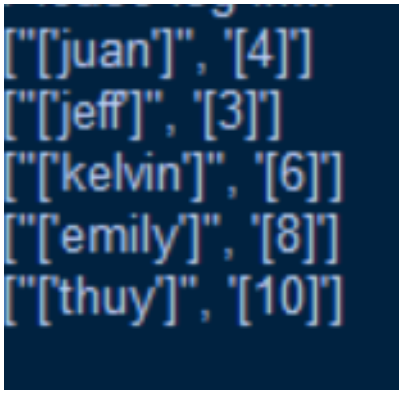
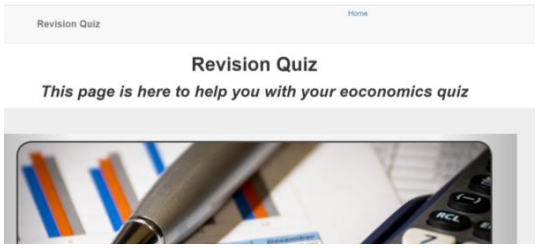
Success Criteria	Achieved?	Justification	Screenshot
Create another file that only displays the name of the student and their score.	Yes	A csv is created, the counter which is being added in the quiz will then be appended into the array and so will the name, this will then be written into a CSV file.	 The program automatically creates this CSV file called scores if it doesn't already, right after it saves names in column A and scores in column B. It does this successfully because it doesn't leave any rows empty.

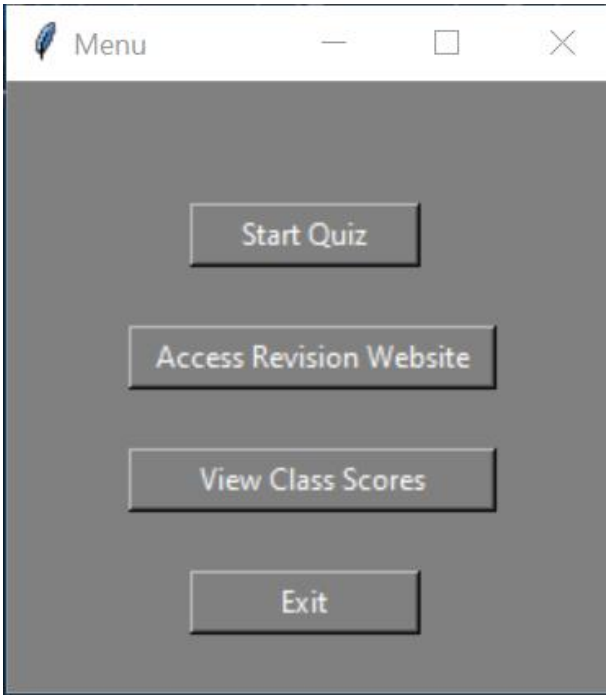
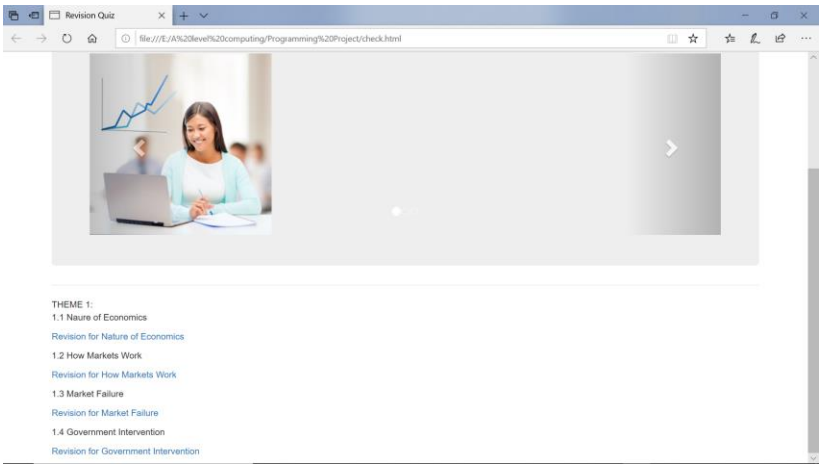
User should be able to access the revision site.	Yes		
The student shouldn't have access to the teacher's system.	Partially	I have added a pop up that asks if a user is a student or a teacher. If so depending on what they choose it will lead to different Tkinter Windows	 If the user chooses no, then the program will output. It will show the admin area which only teachers are going to view. This is to create a profile; I limited the amount of power the admin has. The teacher cannot change the name or the score a student got, this was to avoid a student changing their own score or trolling others by changing the name.

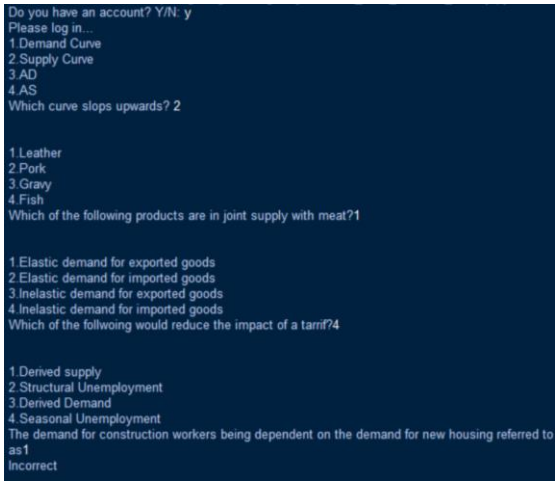
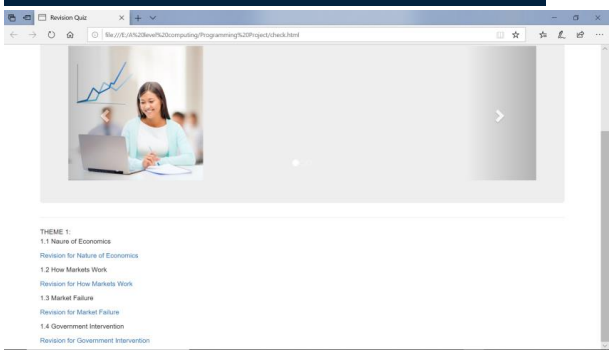
			 However, if the user chooses yes in that pop up then it will output the menu which students have to see. This is displayed to the students. 	
Make the test challenging	Yes	I have implemented some challenging questions straight from the teacher.		This are the questions that are asked, if the user gets the answer right it will continue and add up the counter without them knowing, if they get the answer correct then it will tell them but continue. However, the counter will not be incremented.

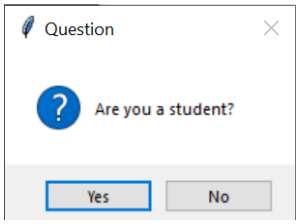
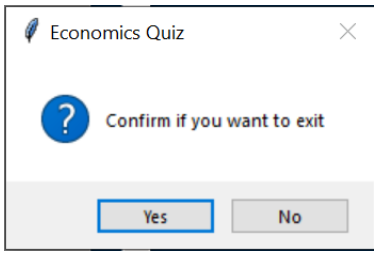
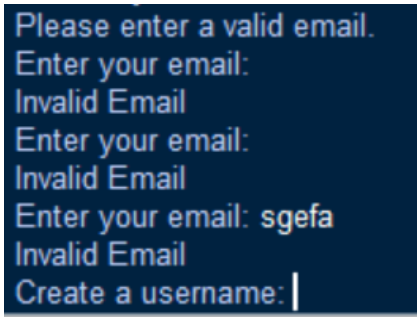
Validate all inputs.	Yes	In prototype 2, I had problems with validating all inputs, the most I could validate was if there was anything inputted. However, for this prototype I managed to validate every input, e.g. when the program asks the user to enter their name it will only accept a string value, whereas before it accepted both numeric and strings. I also managed to validate the email using regex.	 Apart from validating normal inputs, I decided to go ahead and check inside the database and compare it to the username the user wants to create, if the username has been already been taken then it will not let the user have the same one.
Validate Tkinter inputs	Yes	This is to make sure that a username and password are inside the database and the login works properly and doesn't let anyone who	

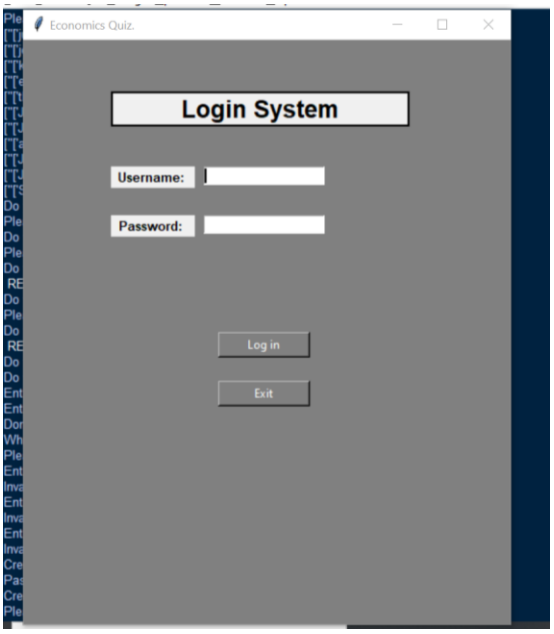
		doesn't have a profile to login as a teacher nor as a student.	  The program does not let the user access if the username or password are wrong.
Make the menu inside the Tkinter window instead of the python interface	Yes	I managed to put the quiz in Tkinter form which wasn't too hard, I assigned different command to each button to output different	
Update the revision website.	Yes	Added a few more functionalities	

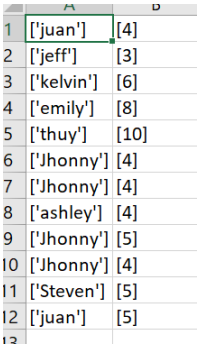
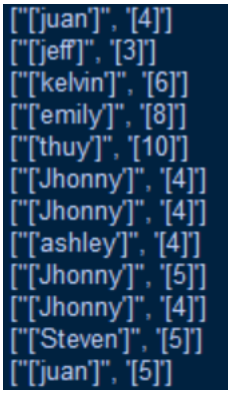
Fully functional menu	Yes	All the commands have their own functionalities	  

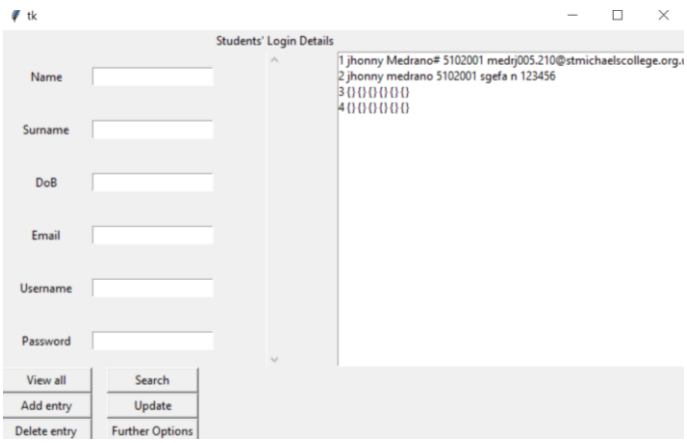
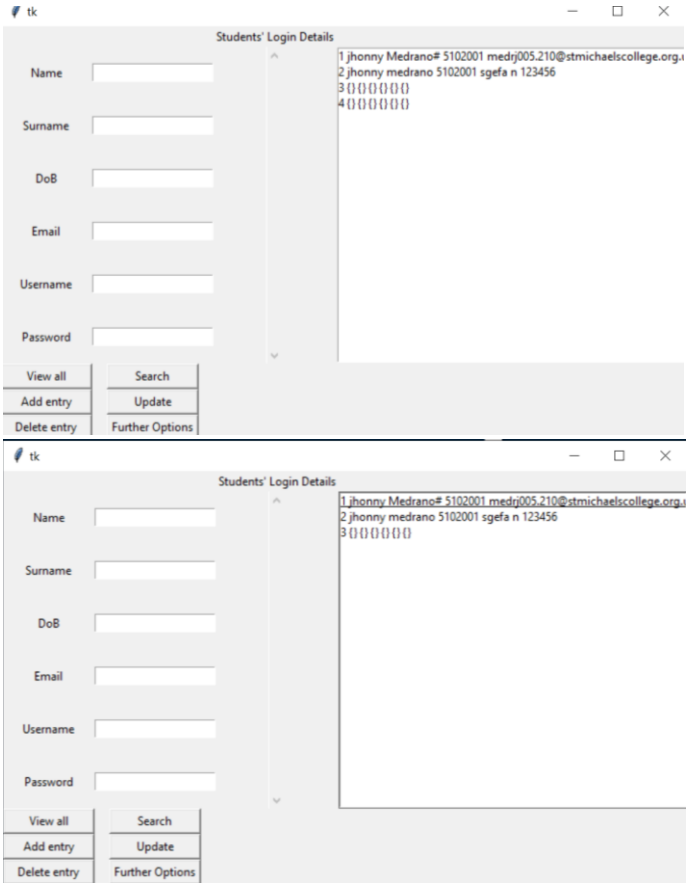
Make the menu inside the Tkinter window instead of the python interface	Yes	Easier to access, less moving around.	 This is the menu that the user sees right after they log in as a student.
Update the revision website	No	This will be used by the student; they have a choice of whether they wanted to revise before the quiz started or not.	 This is what the student will see once they choose to see the revision website. It hasn't been completed, but it has general links to Edexcel revision official sites.

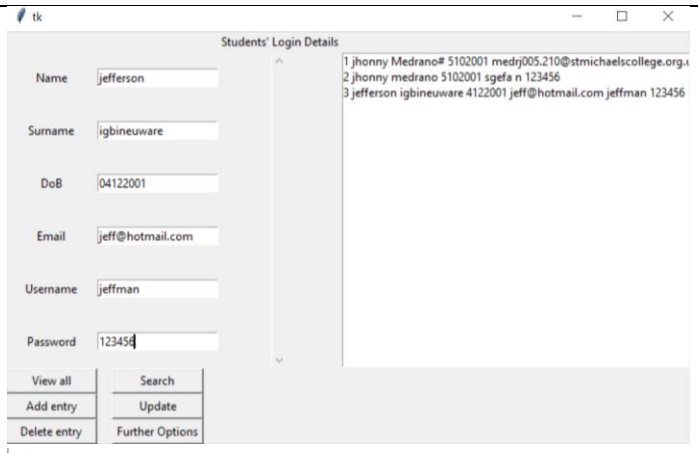
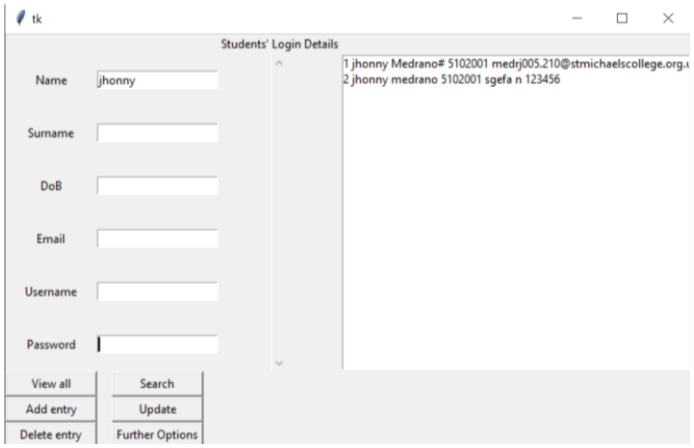
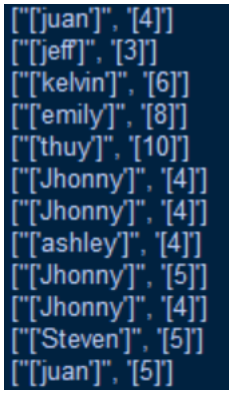
Fully Functional menu	Yes	This is to allow the student or teacher to navigate further in the program.	   The final is to exit, this will allow the user to go back to the GUI. 

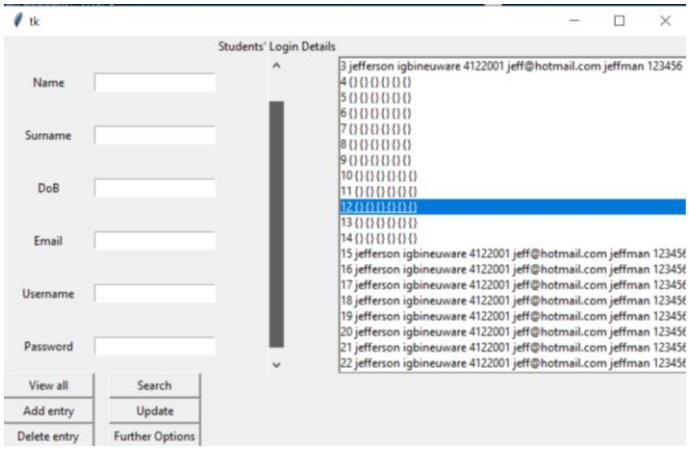
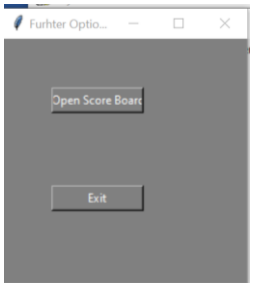
Login Button	Yes	This will allow the user to enter the program and have access to the menu and the quiz.	
Close button	Yes	In order to ensure that the user really wants to exit.	 If the users clicks yes then it will go back to the python interface and start the program again.
Email validation	No	The email must be validated in order to create a profile.	 The program asks the user to enter an email. The email is validated and does tell the user if their email is valid or not but it doesn't iterate if the email is invalid.

Teacher ID	No	This is to give a unique login to the teacher so only they have access to the admin area	There is no entry for an ID in the GUI for the login.
Tkinter priority	Yes	The GUI appears on top of the python interface, this saves the student time since they don't have to switch between windows.	 The GUI opens when the user creates a profile or they enter that they have an acccount.
Counter	Yes	This is needed in order to track the number of answer the user got right.	 At the end of the quiz their score is displayed along as feedback.

CSV file to save name and score	Yes	This is essential for the scoreboard; this will save all students' scores and names	 This is the CSV file and the details that are saved, it saves the names in Column A and the scores in Column B
View Scores	Yes	To allow the user to view how well or bad they are doing compared to the others.	 This is what the program will output when the user wants to view the scores inside the CSV file.
Feedback	Yes	This will be given at the end of the quiz along with their score.	 The feedback changes depending on the score they get. This is thanks to the range check, I will be explaning this further in white box testing.

Teachers area Tkinter entries validation	No	This is needed so the teacher doesn't create profiles with empty fields.	 Fields 3 and 4 are empty, this happened because the teacher has clicked add entry twice, this created 2 empty profiles
Teachers area Tkinter buttons	Yes	This is to give the teacher multiple functionalities	

			   All the buttons work, theyview all the students, adds them, deletes them, update their information, search specific students and display the score board.

Scrollbar inside Tkinter	Yes	This is to give the teacher access to all the students.	 I added empty profiles and duplicated jeffersons profile in order to increase the size of the database, the scrollbar works once the database has more than 22 students.
Further options menu	Partly	To allow the teacher to see the rest of the options they had	The menu was supposed to have 2 extra options inside, the menu displays 2 options which are open socreboard and exit. 
Display Score board in descending order	No	To create an actual scoreboard and ranking the students	I did not implement this in the menu
Display Score board in alphabetical order	No	To create a scoreboard in alphabetical order	I did not implement this in the menu

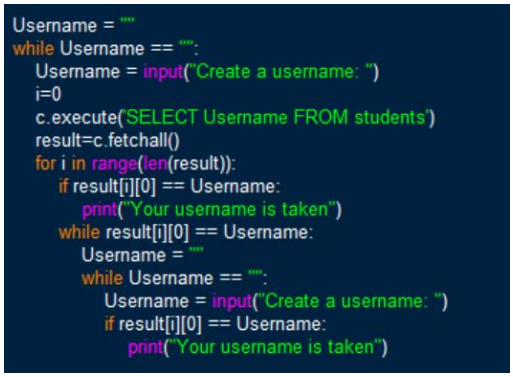
Exit button	No	To exit the program	The exit button inside the Further options menu only closes that specific menu, not the Admin area.
Hide password	Yes	Creates security, no one can see the password apart from the admin.	 The password changes and other people around the user won't be able to see.

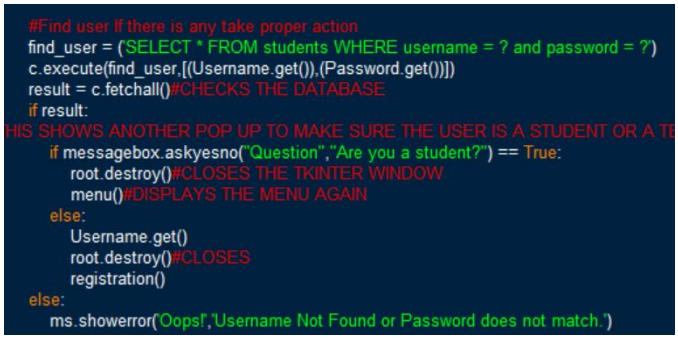
WHITE BOX TESTING

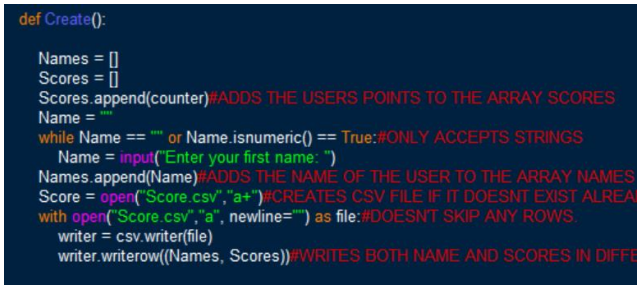
Test	Expected Process	Successful? / Evidence	Actions to be taken

Creation of CSV file	It will create a CSV file if one doesn't exist already	Yes
		`Score = open("Score.csv","a+")#CREATES CSV FILE IF IT DOESNT EXIST ALR`
		`with open("Score.csv","a", newline="") as file:#DOESNT SKIP ANY ROWS.`
		This is the creation of the Score CSV file, the "a+" creates the CSV file if it doesn't already exist
			N/A

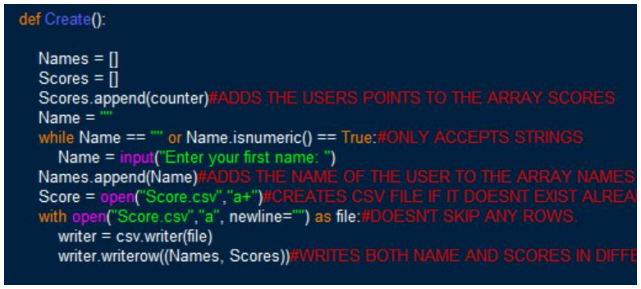
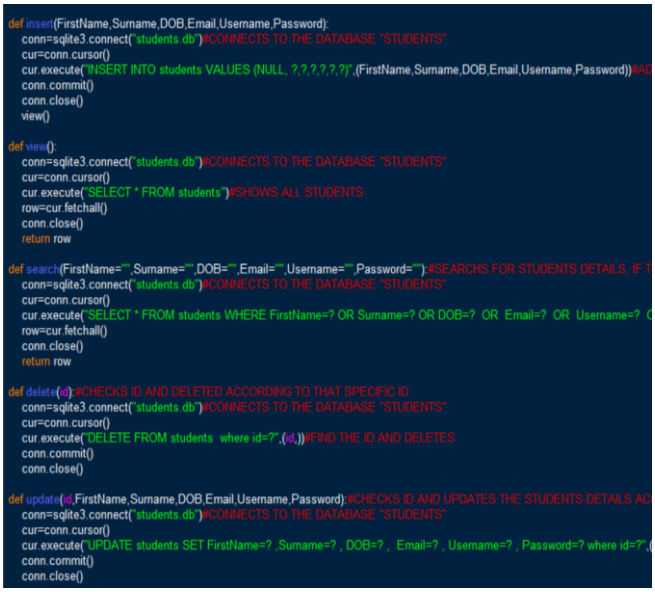
Password validation	It will validate how long the password is	Yes
		`Password = ""`
		`print("Password should be longer than 6 characters but less than 10")`
		`while len(Password) <6 or len(Password)>10:#CHECKS THE LENGTH OF THE PASSWORD`
		`  Password = input("Create a password: ")`
		The password must not be less than 6 or more than 10 characters, this is an ideal length password
			If I was to continue, I will make sure this password was made more secure, for example add numbers and other special characters.

<table>
<tr>
<td>Username validation</td>
<td>It will validate if the username has been taken already</td>
<td>Yes
 

This traverses the whole database using a SQL command in order to find another match</td>
<td>N/A</td>
</tr>
<tr>
<td>Username and password during login in</td>
<td>SQL command to traverse the database to find the password and username</td>
<td>Yes
 

This is to traverse the array and find the username and password, if it found then it will output the menu, if it doesn't find it then it will output an error message.</td>
<td>N/A</td>
</tr>
<tr>
<td>Appending into arrays</td>
<td>Creation of 2 arrays in order to append a name in one and the score in the other</td>
<td>Yes
 

There is 2 arrays created, named "Names" and "Scores"</td>
<td>I would like to implement another database instead of having a CSV file, this way the scores can be shown inside the Tkinter</td>
</tr>
</table>

Appending into the CSV file	The arrays will append their values into the CSV file	Yes  The function csv.write with write.writerow allows the	N/A
SQL command	this will be used inside the admin area	Yes 	N/A

GUI for admin area	If the user logs in as a teacher, the GUI will come up	Yes
		 
		10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8(
		File Edit Format Run Options Window Help
		def delete_command():
		back.delete(selected_tuple[0])
		def update_command():
		back.update(selected_tuple[0],Name_text.get(),Surname_text.get(),DoB_text.get
		def registration():
		window=Tk()
		label1=Label(window,text="Students' Login Details")#Declares the name for label
		label1.grid(row=0,column=2)#Declares the position of label1
		label2=Label(window,text="Name")#Declares the name for label2
		label2.grid(row=1,column=0)#Declares the position of label2
		label3=Label(window,text="Surname")#Declares the name for label3
		label3.grid(row=2,column=0)#Declares the position of label3
		label4=Label(window,text="DoB")#Declares the name for label4
		label4.grid(row=3,column=0)#Declares the position of label4
		label5=Label(window,text="Email")#Declares the name for label5
		label5.grid(row=4,column=0)#Declares the position of label5
		label6=Label(window,text="Username")#Declares the name for label6
		label6.grid(row=5,column=0)#Declares the position of label6
		label7=Label(window,text="Password")#Declares the name for label7
		label7.grid(row=6,column=0)#Declares the position of label7
		#GLOBALISED VARIABLES SO THEY CAN BE ACCESSED ANYTIME THEY A
		global entry1
		global Name_text
		Name_text=StringVar()
		entry1=Entry(window,textvariable=Name_text)
		entry1.grid(row=1,column=1)#Declares the position of entry1
		global entry2
		global Surname_text	

10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10

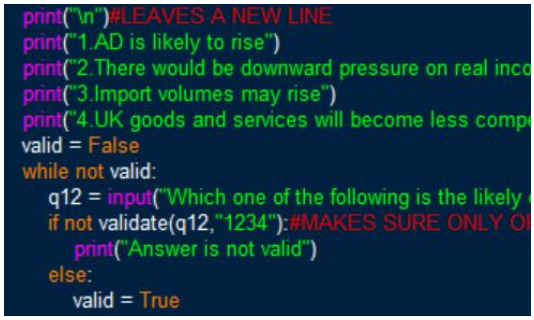
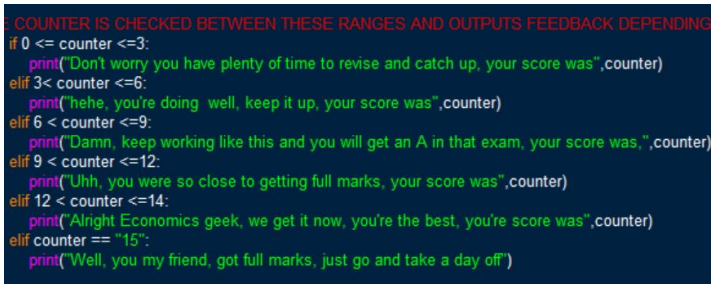
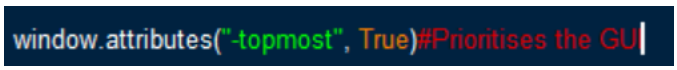
File Edit Format Run Options Window Help

```
#GLOBALISED VARIABLES SO THEY CAN BE ACCESSED ANY TIME
global entry1
global Name_text
Name_text=StringVar()
entry1=Entry(window,textvariable=Name_text)
entry1.grid(row=1,column=1)#Declares the position of entry1

global entry2
global Surname_text
Surname_text=StringVar()
entry2=Entry(window,textvariable=Surname_text)
entry2.grid(row=2,column=1)#Declares the position of entry2

global entry3
global DoB_text
DoB_text=StringVar()
entry3=Entry(window,textvariable=DoB_text)
entry3.grid(row=3,column=1)#Declares the position of entry3

global entry4
global Email_text
Email_text=StringVar()
entry4=Entry(window,textvariable=Email_text)
entry4.grid(row=4,column=1)#Declares the position of entry4

global entry5
global Username_text
Username_text=StringVar()
entry5=Entry(window,textvariable=Username_text)
entry5.grid(row=5,column=1)#Declares the position of entry5

global entry6
global Password_text
Password_text=StringVar()
entry6=Entry(window,textvariable=Password_text)
entry6.grid(row=6,column=1)#Declares the position of entry6

global list1
list1=Listbox(window,height=20,width=59)
list1.grid(row=1,column=3, rowspan=6, columnspan=2)#Declares the posit
```

10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_

File Edit Format Run Options Window Help

```
global list1
list1=Listbox(window,height=20,width=59)
list1.grid(row=1,column=3, rowspan=6, columnspan=2)#Declares the position of

scrl=Scrollbar(window)
scrl.grid(row=1,column=2, sticky='ns',rowspan=6)

list1.configure(yscrollcommand=scrl.set)
scrl.configure(command=list1.yview)#Binds the scrollbar to the list

list1.bind('<<ListboxSelect>>',get_selected_row)

b1=Button(window,text="View all",width=12, command=view_command)
b1.grid(row=7, column=0)#Declares the position of B1

b2=Button(window,text="Add entry",width=12,command=add_command)
b2.grid(row=9, column=0)#Declares the position of B2

b3=Button(window,text="Delete entry",width=12,command=delete_command)
b3.grid(row=11, column=0)#Declares the position of B3

b4=Button(window,text="Search",width=12,command=search_command)
b4.grid(row=7, column=1)#Declares the position of B4

b5=Button(window,text="Update",width=12,command=update_command)
b5.grid(row=9, column=1)#Declares the position of B5

b6=Button(window,text="Further Options",width=12,command=menu2)
b6.grid(row=11, column=1)#Declares the position of B6

window.attributes("-topmost", True)#


window.mainloop()

def menu (): #SHOWS THE MENY TO STUDENTS ONCE THEY HAVE LOGGED IN


    def Create():

        Names = []
```

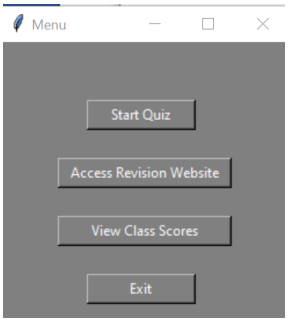
Counter validation	The counter can only increase if the answer is correct.	 Yes	N/A
Range check for counter	The counter will be validated at the end of the quiz	Yes 	I would like to have another interview with the teacher in order to talk about the feedback that can be given depending on their score.
Scores inside CSV	They will be printed out along with the user's name	Yes 	In the future I would create another database that can be opened when the user wants and be shown like the students inside the admin area.
Priority GUI	The GUI will come first after the function is called.	Yes 	N/A
Hidden Password	The way the password is shown when typed	Yes 	N/A

	inside the Tkinter changes to "***"		

Email validation	Validates email and outputs if its valid or not.	Partially
		`#THIS IS WHAT IS IN CHARGE OF VALIDATING THE EMAI`
		`regex = '^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$'`
		`def check(Email):`
		`  if(re.search(regex,Email)):`
		`  print("Valid Email")`
		`  else:`
		`  print("Invalid Email")`
		`global Email`
		`Email = ""`
		`print("Please enter a valid email.")`
		`while Email == "":`
		`  Email = input("Enter your email: ")`
		`  check(Email)#CHECKS THE EMAIL`
			I would like to create a loop if the email is not validated, at the moment the program is able to understand if the email is valid or not but it wont iterate if it doesn't meet the criteria.

BETA TESTING

After completing my alpha testing, I then decided to let one student and one teacher to try the program out for themselves and to let me know if I could still improve this further. The student took the test and gave me ideas to improve the menu, she wanted the menu to be centred because the start quiz and exit buttons were too far to the left for her liking. I went on and changed the axis inside the menu, I changed them from x=50 to x=75 making them aligned with the rest of the buttons which made it look neater and much more professional.



This is the outcome of the new menu. All buttons are aligned.

The student also said that, when she took the quiz she realised than she made a mistake and typed "12" for one of the questions as her answer; the problem was that the program accepted this input, the answer was wrong even if they typed the right answer with the rest of the other options. However, it was not meant to do this, the program is only required to take one option not all 4. I realised that even though it takes 2 values or more it will not increase the counter which is

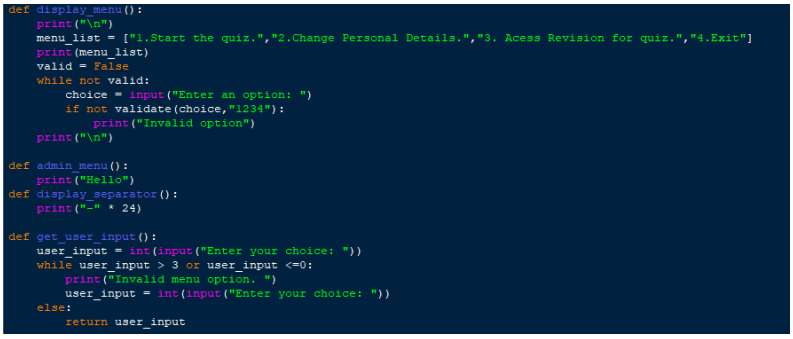
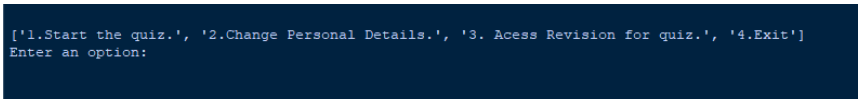
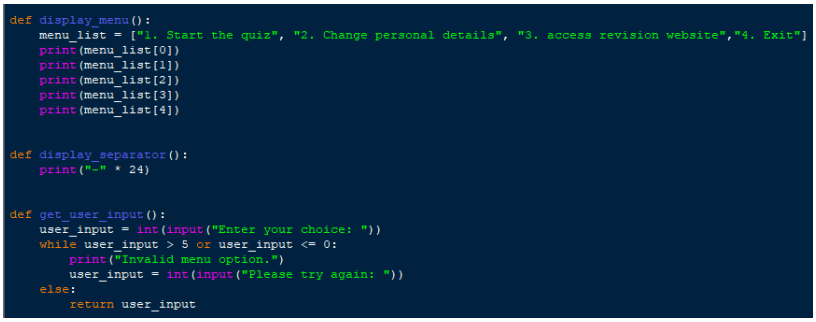
The student also mentioned a forgotten password method inside the login system window. She mentioned that her memory with passwords are not the best and she would like to have that option implemented within the

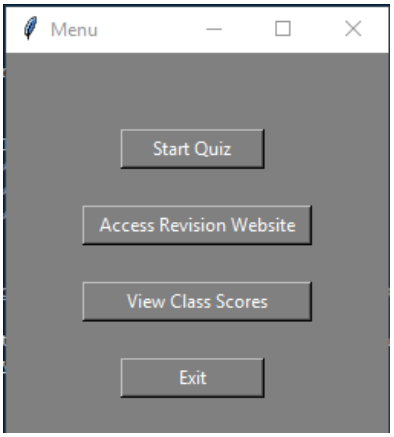
window just in case she wanted to take this quiz in the future without having to create another profile. This of course was a shock to me, I never really thought of this.

As for the teacher she mentioned that, her admin area was completely fine except the further options menu, she mentioned that when she clicks exit, the window doesn't close only the menu does, this frustrated her because the exit button should work for both windows and let her go back to her work. Hearing this, I went on and fixed that menu, which was messy like the students one.

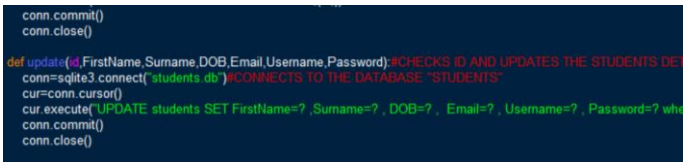
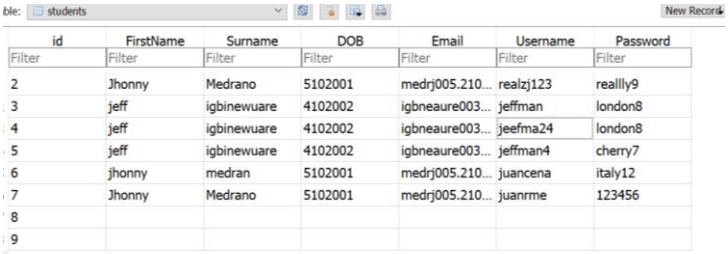
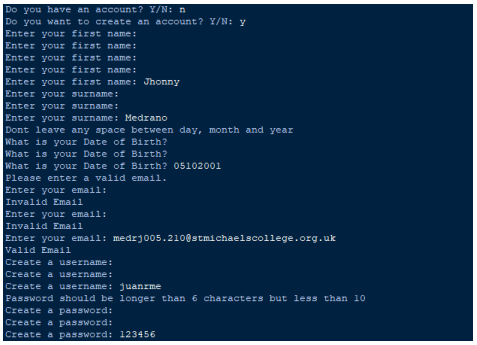
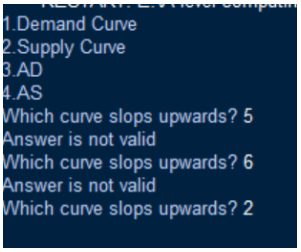
Another problem, that the student found out was that if they inputted a space into a variable it will accept it an continue, this will leave an empty space in the database. I put this as one of my limitations, because I have not found a way of fixing this problem in any way.

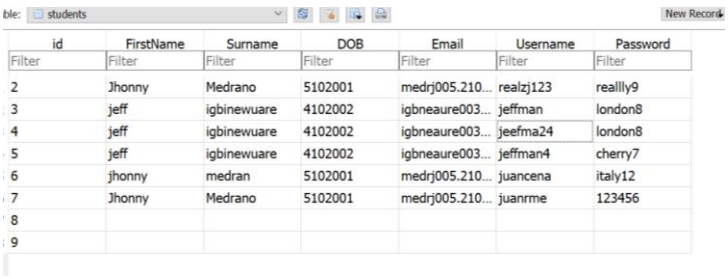
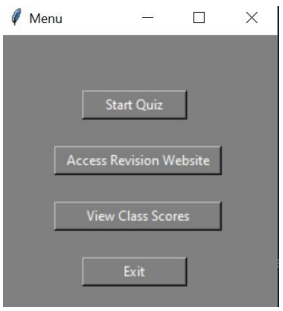
EVALUATING USERS REQUIREMENTS AND LIMITATIONS

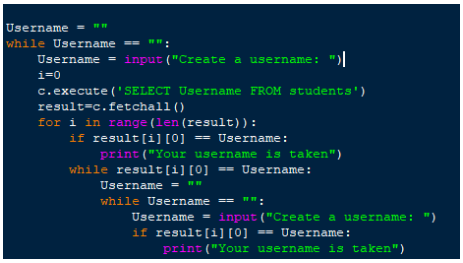
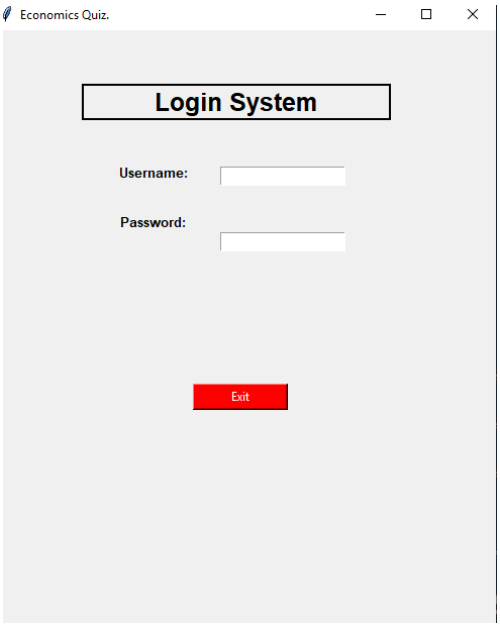
User's requirements	Achieved?	Justification	Screenshot
Clear menus that users will be able to follow with no difficulty	Yes	At the beginning in prototype 1 and 2 I was displaying the menu within the python window and not inside the Tkinter with the rest of the code. At the start of prototype 3, I changed it so it fits the user requirements, this limits the movement of window to window and tried to keep everything within the Tkinter	 This is the initial code for the menu, I tried to keep it as simple as possible for the user a Tkinter window set up and it was all asked through python. This was the outcome to this code:  During prototype 2, I still had the same menu, the only change I made is that I made it HTML wasn't fully built. 

			Finally, for prototype 3, I changed this completely, I no longer had intentions of keepi
			python. This is the code and what it looks like.
			
			`root= Tk()`
			`root.geometry("250x250")`
			`root.title("Menu")`
			`root.configure(background='grey')`
			
			`b1=Button(root,text="Start Quiz",width=12,bg="grey",fg="white",command=quiz)`
			`b1.place(x=75,y=50)#Places button 1 in the given position`
			
			`b2=Button(root,text="Access Revision Website",width=20,bg="grey",fg="white",command=Revision)`
			`b2.place(x=50,y=100)`
			
			`b3=Button(root,text="View Class Scores", width=20,bg="grey",fg="white",command =results)`
			`b3.place(x=50,y=150)#Places button 1 in the given position`
			
			`b4=Button(root,text="Exit",width=12,bg="grey",fg="white",command=last)`
			`b4.place(x=75,y=200)#Places button 4 in the given position`
			
			`root.attributes("-topmost", True)`
			
			`root.mainloop()`
			
			 
			
			This is functional menu, its got 4 different options which the user can choose from. The code shows how each button is got its own role. I will not be showing the commands, but this does demonstrate that the menu works.

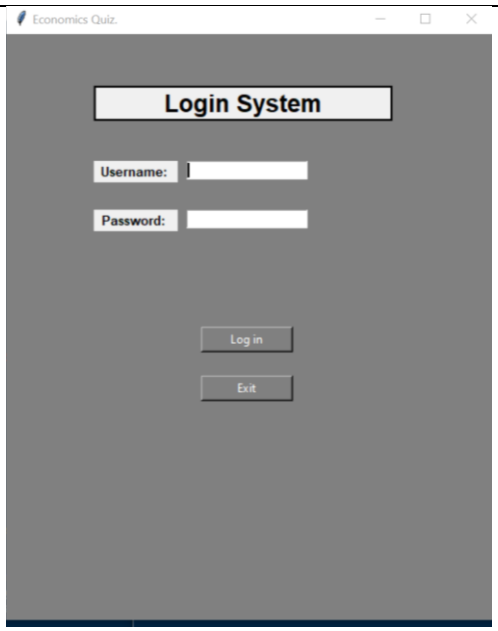
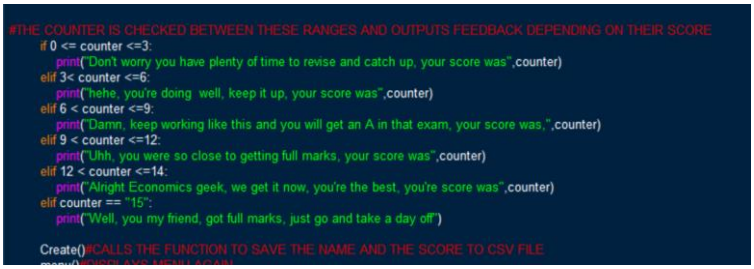
The user will be able to use the program even if there isn't any internet	Yes	The code itself doesn't needs internet. The HTML however I thought it did, but it doesn't	`#SHOWN IN MENU, USED TO OPEN REVISION WEBSITE`
			`def Revision():`
			`new = 2`
			`url= "file:///D:/A%20level%20computing/Programming%20Project/Check.html"#HTML NAME`
			`webbrowser.open(url,new=new)#REDIRECT TO THE HTML, MAKES SURE IT OPENS IT IN BROWSER`

		because the file is saved inside my USB stick.	
Users should be able to change their details if they wish to	Only the teacher can change details.	The teacher can update the details	
The program will create a file that will save details.	Yes	Database is created in order to do this	
If data is invalid, it will repeat until the program accept the data	Yes	Validation is a success and check every variable	 For the tests it also does this. 

			There is only 4 options given and if you type anything that is not given it will not accep
If the credentials that the program asked for, are valid then it will save the details	Yes	It will be saved to a database	
Start the quiz, change personal details, access the revision for the quiz and exit the quiz.	Yes	The student will have a menu that will have all these options and will work normally.	The menu is displayed, whatever option they choose will be outputted. 
The user will have links to videos to help them understand the topics	Yes	These are inside the database.	 They are not only videos but official Edexcel websites which opens their topic straight
The economics teacher will be allowed to have overall control of the database	Yes	The admin area gives the control to change, add, delete or update details.	

and see the results			
Users will create their own logins and passwords	yes	Usernames are validating so each one can have their own unique username	 This shows how it validates the username and it will allow the user to have their own
Both students and admins will be able to log in if their username and password is inside the database.	Yes	It will check the database every time the user clicks log in and check if their credentials are inside the database and if they are valid	
Change colour for Tkinter GUI	Yes	Changed the background to grey	

```
Username = ""
while Username == "":
    Username = input("Create a username: ")
    i=0
    c.execute('SELECT Username FROM students')
    result=c.fetchall()
    for i in range(len(result)):
        if result[i][0] == Username:
            print("Your username is taken")
        while result[i][0] == Username:
            Username = ""
            while Username == "":
                Username = input("Create a username: ")
                if result[i][0] == Username:
                    print("Your username is taken")
```

			
Students being able to get feedback at the end of the quiz.	Yes	At the end the program will check the range and give them a comment and their score	 This is to check the score and output their feedback accordingly
Display score at the end of the quiz automatically with feedback	Yes	At the end it is shown.	

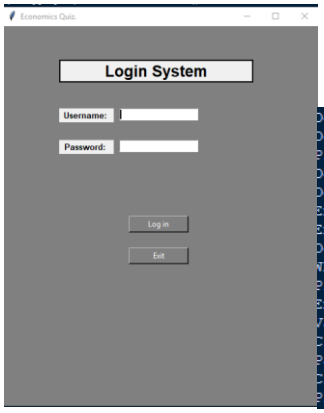
FURTHER DEVELOPMENT AND LIMITATIONS

- No background music or sound effects during the game and the menu's making program dull.
- Scoreboard is very simple compared to my initial goal.
- Teacher and student only have one option of seeing the scores, I wanted to extend this to 3 options. Alphabetical order, normal order, and descending order.
- The GUI still looks robotic, it has improved from prototype 2 but it's not what I wanted to make.

MAINTENANCE

The program is made in a way that is easy to follow and shows which function is executed. It is broken down in different functions which connect to each other, this happens when they are called right after the other. Every function has comments which helps other programmers know what is happening in that function. All the variable names are clear for other programmers to know what they are and what they will store.

SUCCESS CRITERIA EVALUATION.

Success Criteria	Achieved?	Justification	Screenshot
The program will allow the user to log in or sign up	Yes	At the start of the program the program will ask the user if they have an account, depending on their answer it will show the different outcomes. Either the questions to create a profile or the login window	 Do you have an account? Y/N: u Do you have an account? Y/N: y Please log in... Do you have an account? Y/N: n Do you want to create an account? Y/N: y Enter your first name: jhonny Enter your surname: medran Dont leave any space between day, month and year What is your Date of Birth? 05102001 Please enter a valid email. Enter your email: medrj005.210@stmichaelscollege.org.uk Valid Email Create a username: juancena Password should be longer than 6 characters but less than 10 Create a password: italy12 Please log in... Here, there are both outcomes shown depending if they have an account or not. The development for the Tkinter window started in prototype 2 and had some bugs. details were taken since prototype 1 but they were not validated properly.
Program will be responsible for the layout of tkinter making	Yes	The code written is responsible for how the GUI looks like and how it	`root.attributes("-topmost", True)` This is the code line used to prioritise the login window, before in prototype 2, I alwa problem in which the tkinter window always appeared behind every other window a to manually click into the Tkinter window which was annoying after a while. The layo buttons and entries are also shown in the code, the colour and everything to make

<table>
<tr>
<td>it understandable</td>
<td></td>
<td>will be outputted, for example the login window will come up first ignoring windows behind it, due to its importance.</td>
<td>everything readable is inside the code,
<pre>
Surname_text=StringVar()
entry2=Entry(window,textvariable=Surname_text)
entry2.grid(row=2,column=1)

global entry3
global DoB_text
DoB_text=StringVar()
entry3=Entry(window,textvariable=DoB_text)
entry3.grid(row=3,column=1)

global entry4
global Email_text
Email_text=StringVar()
entry4=Entry(window,textvariable=Email_text)
entry4.grid(row=4,column=1)

global entry5
global Username_text
Username_text=StringVar()
entry5=Entry(window,textvariable=Username_text)
entry5.grid(row=5,column=1)

global entry6
global Password_text
Password_text=StringVar()
entry6=Entry(window,textvariable=Password_text)
entry6.grid(row=6,column=1)

global list1
list1=Listbox(window,height=20,width=59)
list1.grid(row=1,column=3, rowspan=6, columnspan=2)

scr1=Scrollbar(window)
scr1.grid(row=1,column=2, sticky='ns',rowspan=6)

list1.configure(yscrollcommand=scr1.set)
scr1.configure(command=list1.yview)

list1.bind('<<ListboxSelect>>',get_selected_row)

b1=Button(window,text="View all",width=12, command=view_command)
b1.grid(row=7, column=0)

b2=Button(window,text="Add entry",width=12,command=add_command)
b2.grid(row=9, column=0)

b3=Button(window,text="Delete entry",width=12,command=delete_comma
b3.grid(row=11, column=0)

b4=Button(window,text="Search",width=12,command=search_command)
b4.grid(row=7, column=1)

b5=Button(window,text="Update",width=12,command=update_command)
b5.grid(row=9, column=1)

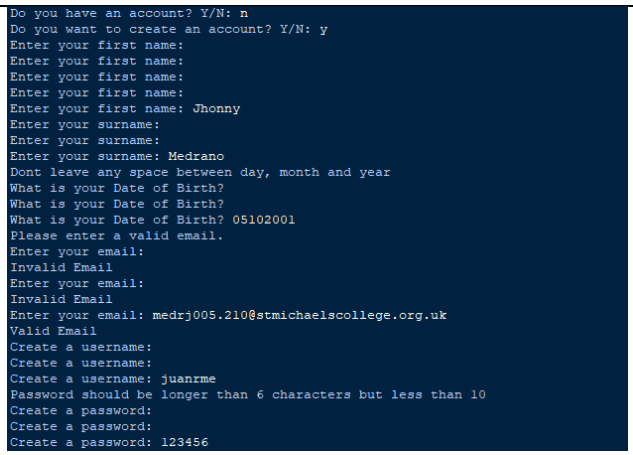
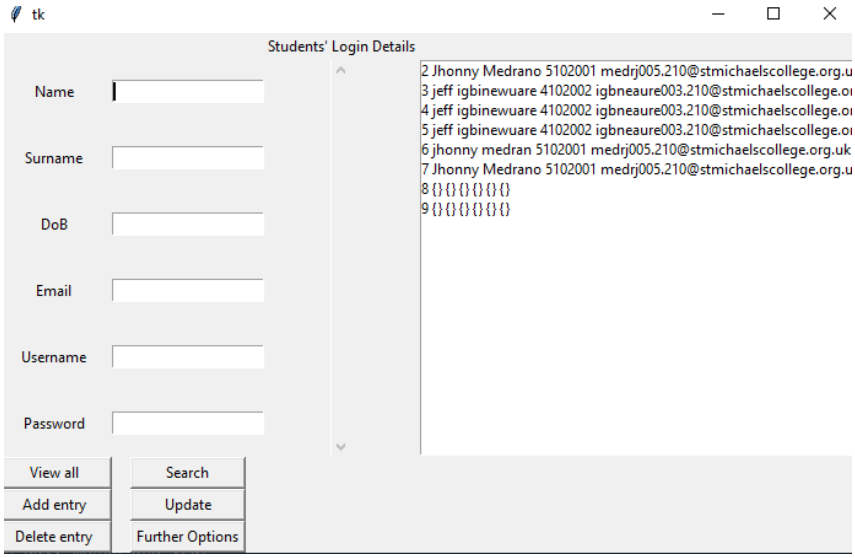
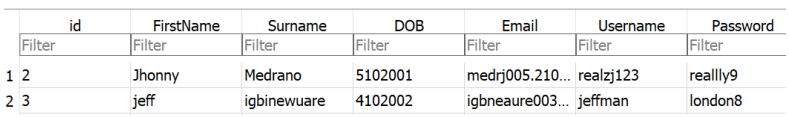
b6=Button(window,text="Further Options",width=12,command=menu2)
b6.grid(row=11, column=1)
</pre>
The admin area was more accurate than the other Tkinter windows, this is due to the use of columns and rows, whereas in the menu and login system I used x and y values to determine the place of an entry, button, title or header.</td>
</tr>
<tr>
<td>The program will use validation to make sure everything is filled out for the users</td>
<td>Yes</td>
<td>I used 2 different functions to check the normal strings. Presence check so a string cannot be skipped if empty and the numeric function to not input numbers where only a string is required</td>
<td>
<pre>
FirstName = ""
while FirstName == "" or FirstName.isnumeric() == True:
    FirstName = input("Enter your first name: ")

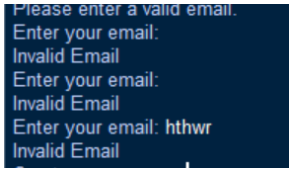
Surname = ""
while Surname == "" or Surname.isnumeric() == True:
    Surname = input("Enter your surname: ")

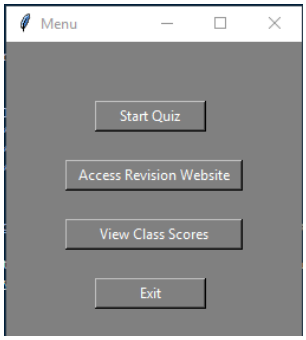
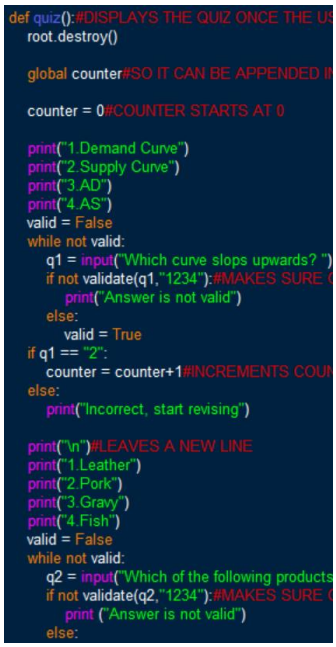
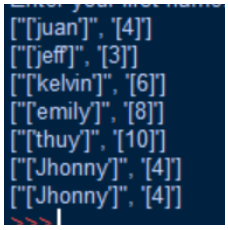
DOB = ""
print("Dont leave any space between day, month and year")
while DOB.isnumeric() == False:
    DOB = input("What is your Date of Birth? ")

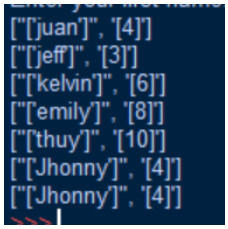
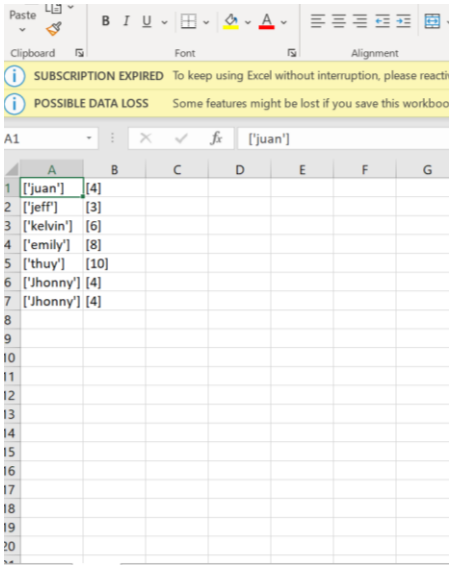
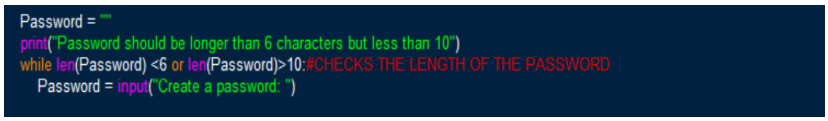
global Email
Email = ""
print("Please enter a valid email.")
while Email == "":
    Email = input("Enter your email: ")
    check(Email)

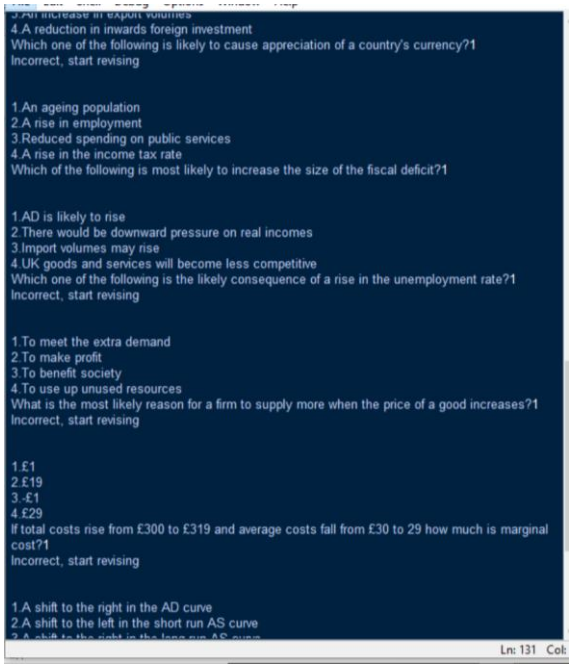
Username = ""
while Username == "":
    Username = input("Create a username: ")
    i=0
    c.execute('SELECT Username FROM students')
    result=c.fetchall()
    for i in range(len(result)):
        if result[i][0] == Username:
            print("Your username is taken")
        while result[i][0] == Username:
            Username = ""
            while Username == "":
                Username = input("Create a username: ")
                if result[i][0] == Username:
                    print("Your username is taken")
</pre>
For example, in the code we can see that the program only takes strings and not an
However, this change in the email, username and password because an email might
numbers. Everything has to be filled in otherwise it will not continue to the next que</td>
</tr>
</table>

			 I deliberately left them blank and clicked enter to show that the presence check is ac it does work like its supposed to. If the user doesn't input anything it will keep askin valid value is inputted.
The program will validate presence check in the admin area.	No	I have not managed to do this in the final prototype, when the teacher decides to add a new student to the database using the admin area it will add empty spaces instead of	 Here there is a picture of the admin area, ID 8 and ID 9 are empty there is no values the entries, this was only done because the admin clicked add entry when there was string inside the entries given. However, a student is not able to do this in the creati their profile
Program will save the details inputted into a separate file	Yes	This was possible by creating a SQL database with different columns. The	

		primary key, name, surname, DOB, email, username and password.	Here there is an example of how the database stores the student's details, this is a v seeing the records inside the database, the admin can also see all the student which saved. All records are saved inside the SQL database.
The program will allow to overwrite in their file to correct their details	No	The program will not allow a student to overwrite their own details	The program doesn't facilitate the student with changing their details, this is becaus time, I don't have enough time to complete this. It was one of my main success crite got caught up finishing everything else.
The admin will be allowed to change the students details within the admin area	Yes	The admin had overall control of this, therefore they will have the ability to update the user's details.	 The option "Update" allows the teacher to overwrite one of the student's name or username.
The program will validate the email	Yes	The program can validate if an email is authentic or if it isn't.	  The program recognises when the email that has been inputted in valid or not.

The program will show menus and it will fully responsive	Yes	After I decided to move the menu to a Tkiner window I gave all the button commands and they are working perfectly.	
The program will have a point system to count all the points when the user gets the answer right.	Yes	The counter is responsible to track the student progress throughout the quiz.	 Counter starts from 0 and its incremented if they get the question right
The user will be allowed to display their results and their details if they wish	Yes	There is an option in the quiz that allows them to see their score and the rest of his classmates scores as well.	 One of the options that is shown on the menu is show the results.

A leader board will be created automatically when more than 1 student takes the quiz	Yes	This will be created just after the user takes the quiz, after it creates the external file it will save the students name and their score	  The csv file saves the name and score but the user can also see them without having the file
Check password length	Yes	This is done with a range function; it checks the length of the string and if it is not the length which is required then it will not continue and keep repeating it until the length is valid.	
Program will not crash after	Yes	In prototype 2, I had	

confirmation of exit button		problems in closing the Tkinter window without making the whole program crash. This was fixed in prototype 3.	
Program will follow sub routines in order and if they have to, then they will be repeated	Yes	The main() sub routine will be responsible for calling all other functions in the order that is required.	
Program will display the quiz	Yes	There is an option in the menu, the quiz is finally finished so they can take it.	

The email is validated, loop if it's wrong.	No	It's true that the program validates if the email is authentic or not, the problem is that if the email is not valid it will continue to asking for the username. This is not achieved because I can't seem to get the regex to loop. I have tried researching and applying my own programming techniques but I have not been able to fix this bug.	

LIMITATIONS AND CONSTRAINTS

As a programmer I have analysed all problems I have been aware of thanks to Beta and Alpha testing. There is no validation for a teacher, this means that a student could log in and click that they are not a student and the admin area will appear to them, this means that they can troll and sabotage or corrupt other students' details. This limitation was due to the inability of giving the teacher an Identification key, the reason for this

Another limitation, is that the table is visible but there were more initial criteria that I wanted to meet or example, I wanted to show the table in alphabetical order or in highest to lowest, this however was not met, I tried many times and the only thing I could accomplish was show the names in alphabetical order but it didn't show the score next to them or it showed the scores in descending order but it didn't show the names next to it.

EVALUATION

As my final prototype, I have been able to achieve almost all the user requirements I proposed myself at the beginning of prototype 1 and unfinished requirements that were not met in prototype 2. In this prototype there is a database that can save all the students in one class and even of every school. There is no limit to the number of students that want to sign up. The program is easy to follow, it flows easily and the configuration and layout of every Tkinter window is in order, databases, csv files and the python interface. All python variables are validated which means that, the success criteria were successful and met, the only problem there found was that the email is not iterated if its invalid, the program successfully checks if its valid or not but if doesn't iterate like it meant to do. It does iterate with the other variables such as name, surname, username and password if it doesn't meet the requirements. For example, the password must be a certain length which is validated before creating the profile.

The username is also validated, one of the requirements was to have a unique username assigned to each student, there was an error in prototype 2 which allowed 2 users to have the same username which caused confusion because they had access to the same profile. This was fixed in prototype 3 and it made sure that the user had to input another username. The program also allows the user to access a revision website which couldn't be finished due to the time restriction, the development of the program was very complex and took a lot of time. Especially the manipulation of the SQL database inside the Tkinter window. Also, I am not as informed in the HTML field as I am with python therefore my programming skills were limited. I had to do a lot of research in order to get the HTML file to its final version.

The menu is responsive and has been implemented inside the Tkinter GUI which wasn't my initial goal at first, having it inside a Tkinter GUI is more efficient than having it in the python interface. There were some user requirements which were not met, in prototype 1 I wanted to allow the user to change their details in case they changed their mind in terms of the username or if they made a mistake with their details. However, this was not completed, the only user that can change the details is the admin. The only solution is for the student to ask the teacher to fix their details for them, but I initially wanted the student to dot his themselves and not have to ask the teacher for this.

The counter inside the quiz is encapsulated, and the program is able to count the points that each student gets, the program also creates a CSV file if it doesn't exist already and inputs the name and the score that each student gets, this is another criterion which has been met. The counter is also privatised because when one user exits and another user log in again within the same running program the counter doesn't start accumulating from where the last user finished but it resets, and it starts again like it should do.

The leader board is another criterion which has been met, the program successfully shows the user or the teacher the table inside the CSV file.

APPENDIX

FINAL CODE

10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)

File Edit Format Run Options Window Help

```
import sqlite3 #this will allow me to establish connection with the database
import csv#SCORES
#import re
import os #Helps to open google windows
import codecs #This is used to open files inside the computer
import webbrowser #This is used to open the HTML file
from tkinter import * #This is used for the GUI
from tkinter import messagebox as ms
import tkinter.messagebox
from tkinter import messagebox
from tkinter import Tk, Button
import back

def validate(strings, chars):
    return True in [c in strings for c in chars]#This will validate inputs given by the use further on the program

conn = sqlite3.connect('students.db') #Establishes a connection to the database

c = conn.cursor()

#IF THE DATABASE IS NOT ALREADY CREATED, IT WILL CREATE IT WITH ALL THESE COLUMNS

c.execute("""CREATE TABLE IF NOT EXISTS students (
                id INTEGER PRIMARY KEY,
                FirstName Text,
                Surname Text,
                DOB Integer,
                Email Text,
                Username Text,
                Password Text)""")#This will only create a table if it isn't already

def get_selected_row(event):
    global selected_tuple #Globalised variable in order to access other functions
    index=list1.curselection()[0]
    selected_tuple=list1.get(index)
    entry1.delete(0,END)
    entry1.insert(END,selected_tuple[1])
    entry2.delete(0,END)
    entry2.insert(END,selected_tuple[2])
    entry3.delete(0,END)
    entry3.insert(END,selected_tuple[3])
```

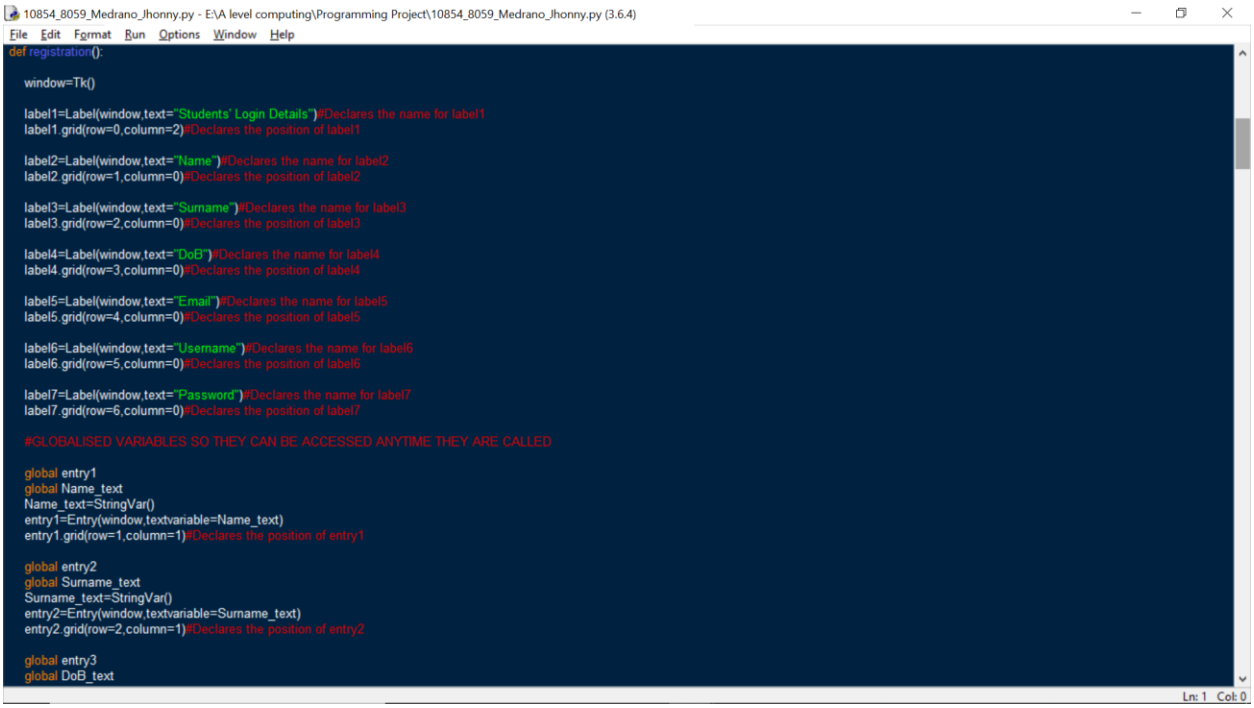
Ln: 1 Col: 0

10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)

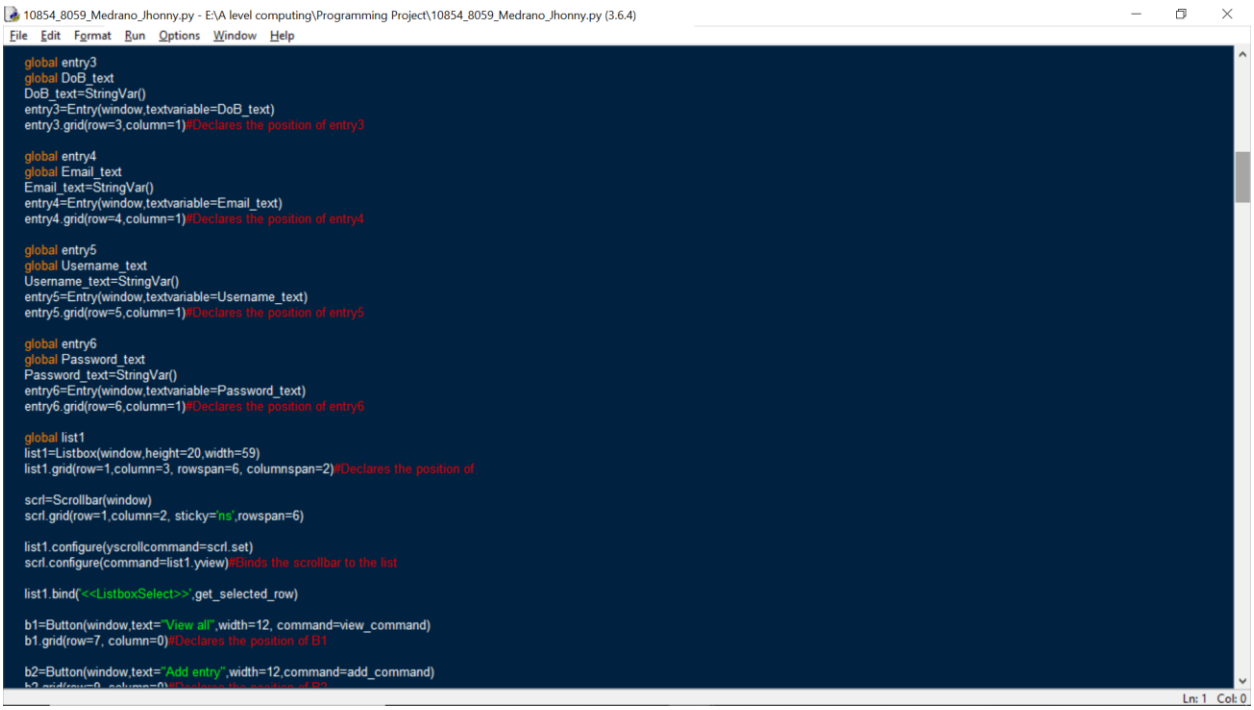
File Edit Format Run Options Window Help

```
def get_selected_row(event):
    global selected_tuple #Globalised variable in order to access other functions
    index=list1.curselection()[0]
    selected_tuple=list1.get(index)
    entry1.delete(0,END)
    entry1.insert(END,selected_tuple[1])
    entry2.delete(0,END)
    entry2.insert(END,selected_tuple[2])
    entry3.delete(0,END)
    entry3.insert(END,selected_tuple[3])
    entry4.delete(0,END)
    entry4.insert(END,selected_tuple[4])
    entry5.delete(0,END)
    entry5.insert(END,selected_tuple[5])
    entry6.delete(0,END)
    entry6.insert(END,selected_tuple[6])

def view_command(): #This is connected to the program "back" allows the admin to see all students in database.
    list1.delete(0,END)
    for row in back.view():
        list1.insert(END,row)

def search_command(): #Connected to backend, allows admin to search for a specific student or value
    list1.delete(0,END)
    for row in back.search(Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get()): #Gets all results found to those specific values entered
        list1.insert(END,row)#Shows results in extended entry

def add_command():
    back.insert(Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get())
    list1.delete(0,END)
    list1.insert(END,Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get())

def delete_command():
    back.delete(selected_tuple[0])

def update_command():
    back.update(selected_tuple[0],Name_text.get(),Surname_text.get(),DoB_text.get(),Email_text.get(),Username_text.get(),Password_text.get())

def registration():

    window=Tk()
```

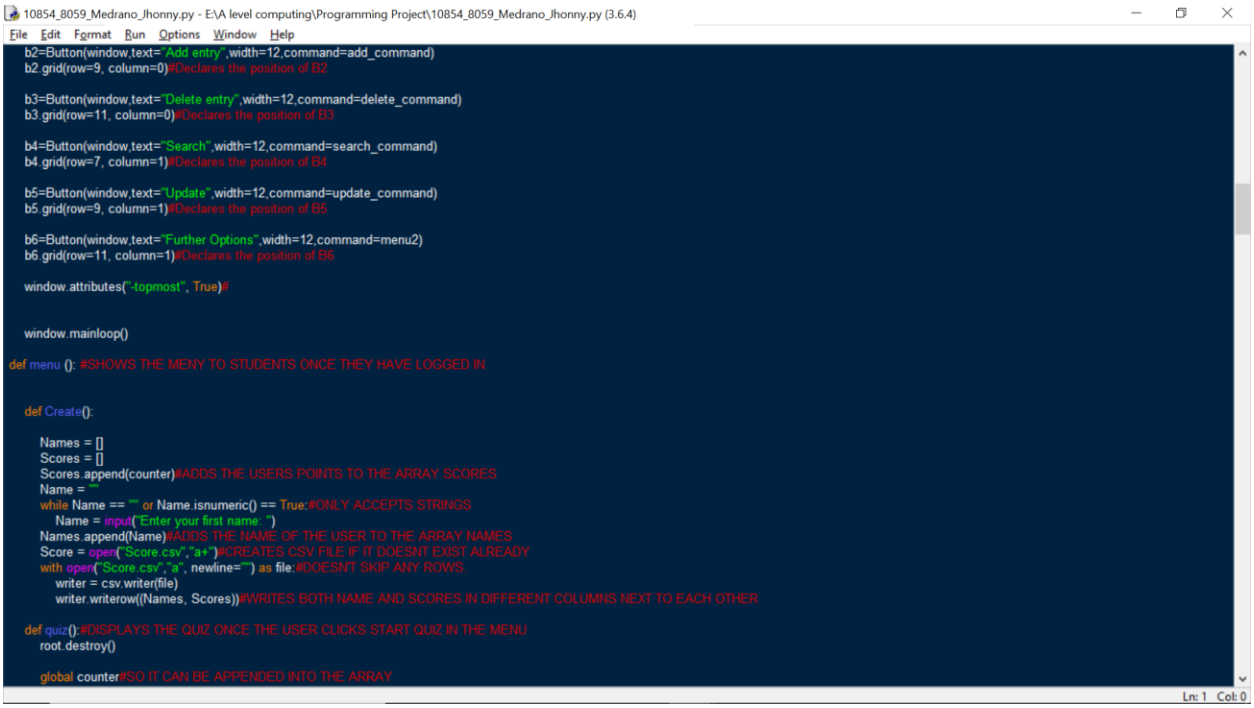
Ln: 1 Col: 0



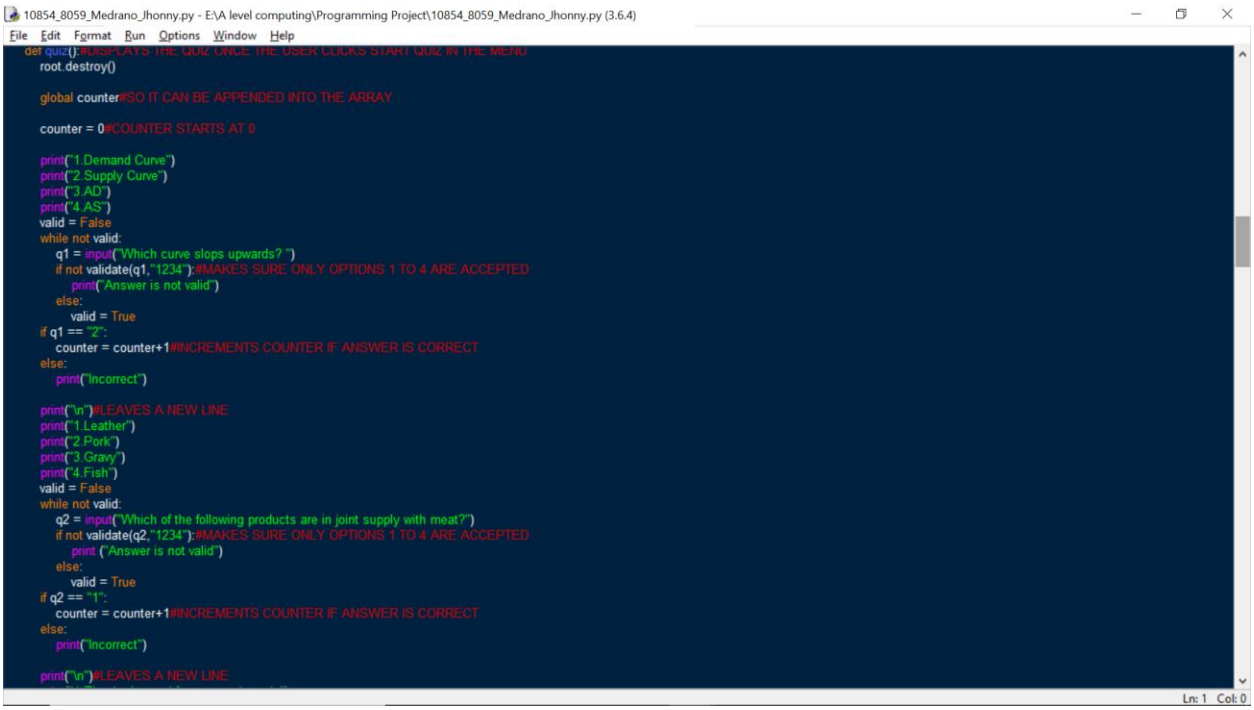
```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
def registration():

    window=Tk()

    label1=Label(window,text="Students' Login Details")#Declares the name for label1
    label1.grid(row=0,column=2)#Declares the position of label1

    label2=Label(window,text="Name")#Declares the name for label2
    label2.grid(row=1,column=0)#Declares the position of label2

    label3=Label(window,text="Surname")#Declares the name for label3
    label3.grid(row=2,column=0)#Declares the position of label3

    label4=Label(window,text="DoB")#Declares the name for label4
    label4.grid(row=3,column=0)#Declares the position of label4

    label5=Label(window,text="Email")#Declares the name for label5
    label5.grid(row=4,column=0)#Declares the position of label5

    label6=Label(window,text="Username")#Declares the name for label6
    label6.grid(row=5,column=0)#Declares the position of label6

    label7=Label(window,text="Password")#Declares the name for label7
    label7.grid(row=6,column=0)#Declares the position of label7

    #GLOBALISED VARIABLES SO THEY CAN BE ACCESSED ANYTIME THEY ARE CALLED

    global entry1
    global Name_text
    Name_text=StringVar()
    entry1=Entry(window,textvariable=Name_text)
    entry1.grid(row=1,column=1)#Declares the position of entry1

    global entry2
    global Surname_text
    Surname_text=StringVar()
    entry2=Entry(window,textvariable=Surname_text)
    entry2.grid(row=2,column=1)#Declares the position of entry2

    global entry3
    global DoB_text
                                                                    Ln: 1   Col: 0
```



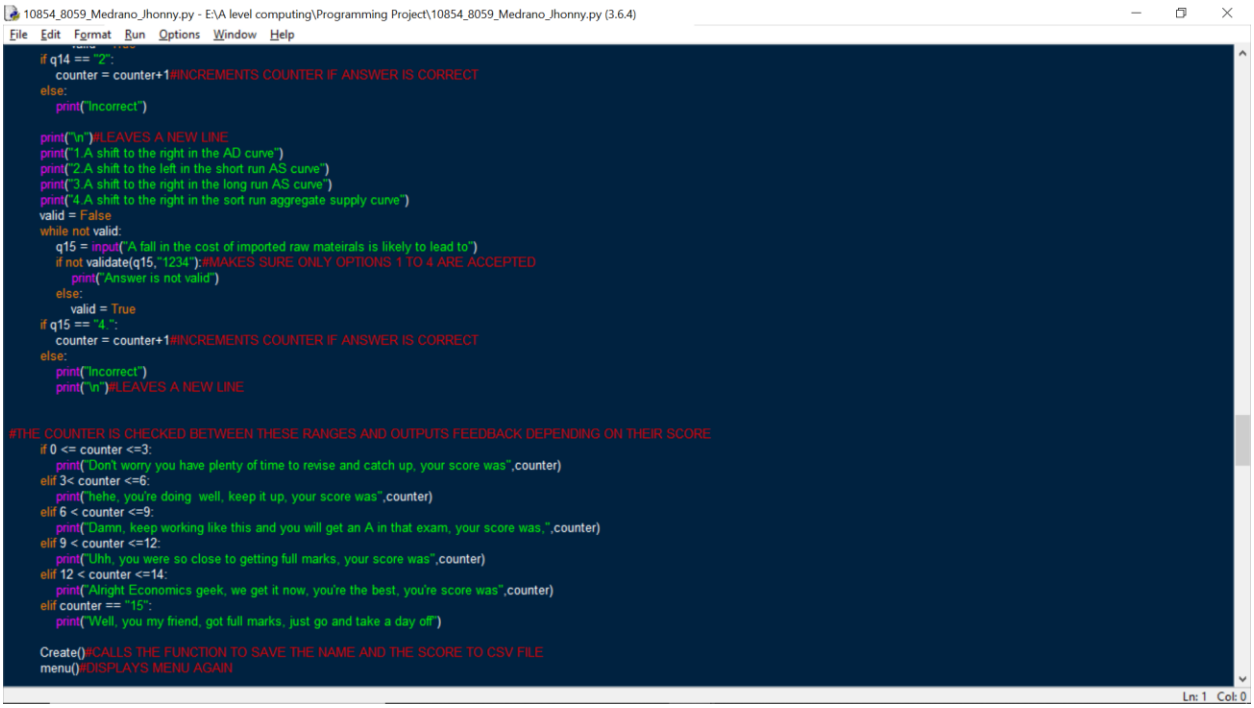
```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
    global entry3
    global DoB_text
    DoB_text=StringVar()
    entry3=Entry(window,textvariable=DoB_text)
    entry3.grid(row=3,column=1)#Declares the position of entry3

    global entry4
    global Email_text
    Email_text=StringVar()
    entry4=Entry(window,textvariable=Email_text)
    entry4.grid(row=4,column=1)#Declares the position of entry4

    global entry5
    global Username_text
    Username_text=StringVar()
    entry5=Entry(window,textvariable=Username_text)
    entry5.grid(row=5,column=1)#Declares the position of entry5

    global entry6
    global Password_text
    Password_text=StringVar()
    entry6=Entry(window,textvariable=Password_text)
    entry6.grid(row=6,column=1)#Declares the position of entry6

    global list1
    list1=Listbox(window,height=20,width=59)
    list1.grid(row=1,column=3, rowspan=6, columnspan=2)#Declares the position of

    scrl=Scrollbar(window)
    scrl.grid(row=1,column=2, sticky='ns',rowspan=6)

    list1.configure(yscrollcommand=scrl.set)
    scrl.configure(command=list1.yview)#Binds the scrollbar to the list

    list1.bind('<<ListboxSelect>>',get_selected_row)

    b1=Button(window,text="View all",width=12, command=view_command)
    b1.grid(row=7, column=0)#Declares the position of B1

    b2=Button(window,text="Add entry",width=12,command=add_command)
                                                                    Ln: 1   Col: 0
```



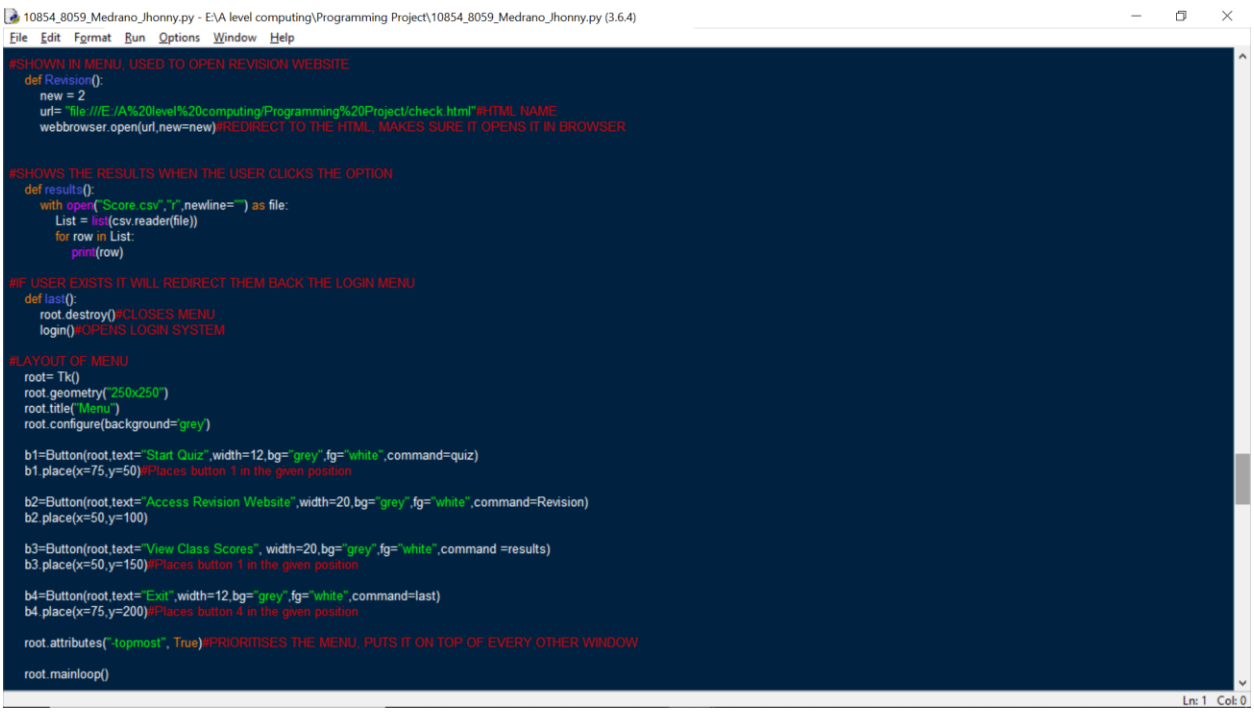
```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
    b2=Button(window,text="Add entry",width=12,command=add_command)
    b2.grid(row=9, column=0)#Declares the position of B2

    b3=Button(window,text="Delete entry",width=12,command=delete_command)
    b3.grid(row=11, column=0)#Declares the position of B3

    b4=Button(window,text="Search",width=12,command=search_command)
    b4.grid(row=7, column=1)#Declares the position of B4

    b5=Button(window,text="Update",width=12,command=update_command)
    b5.grid(row=9, column=1)#Declares the position of B5

    b6=Button(window,text="Further Options",width=12,command=menu2)
    b6.grid(row=11, column=1)#Declares the position of B6

    window.attributes("-topmost", True)#


    window.mainloop()

def menu (): #SHOWS THE MENY TO STUDENTS ONCE THEY HAVE LOGGED IN


    def Create():

        Names = []
        Scores = []
        Scores.append(counter)#ADDS THE USERS POINTS TO THE ARRAY SCORES
        Name = ""
        while Name == "" or Name.isnumeric() == True:#ONLY ACCEPTS STRINGS
            Name = input("Enter your first name: ")
        Names.append(Name)#ADDS THE NAME OF THE USER TO THE ARRAY NAMES
        Score = open("Score.csv","a+")#CREATES CSV FILE IF IT DOESNT EXIST ALREADY
        with open("Score.csv","a", newline="") as file:#DOESNT SKIP ANY ROWS
            writer = csv.writer(file)
            writer.writerow((Names, Scores))#WRITES BOTH NAME AND SCORES IN DIFFERENT COLUMNS NEXT TO EACH OTHER

    def quiz():#DISPLAYS THE QUIZ ONCE THE USER CLICKS START QUIZ IN THE MENU
        root.destroy()

        global counter#SO IT CAN BE APPENDED INTO THE ARRAY
Ln: 1 Col: 0
```



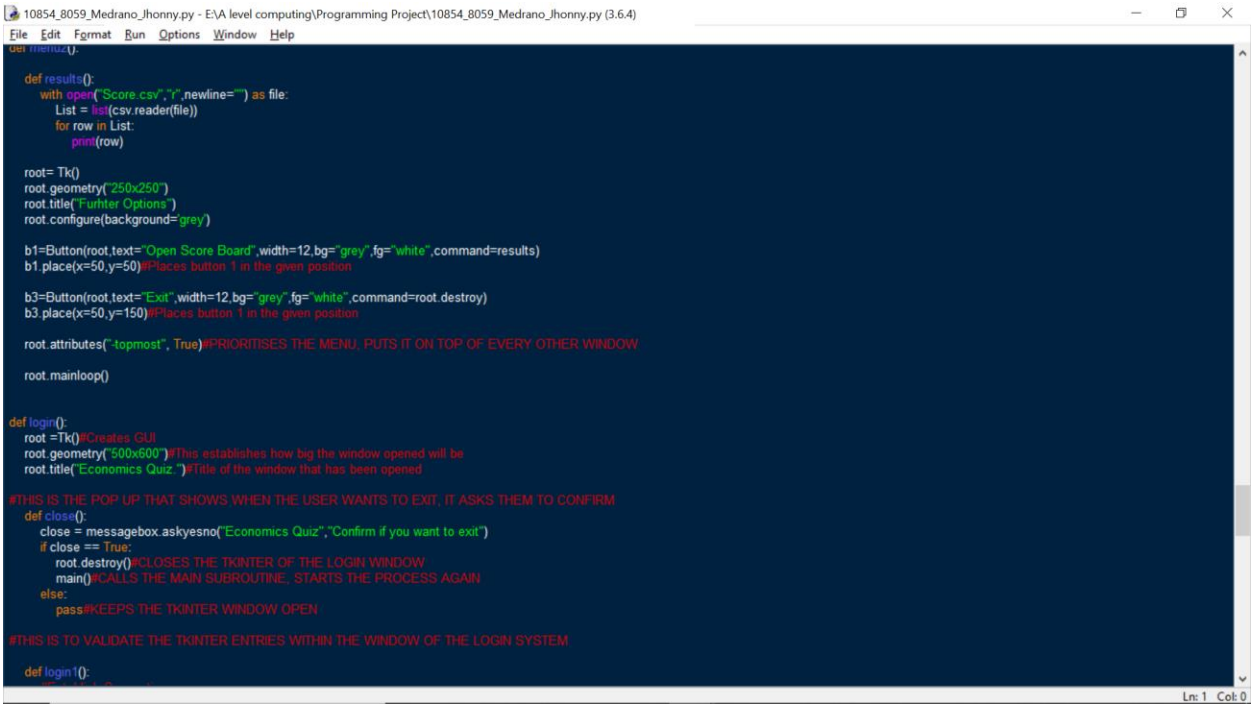
```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
    def quiz():#DISPLAYS THE QUIZ ONCE THE USER CLICKS START QUIZ IN THE MENU
        root.destroy()

        global counter#SO IT CAN BE APPENDED INTO THE ARRAY

        counter = 0#COUNTER STARTS AT 0

        print("1.Demand Curve")
        print("2.Supply Curve")
        print("3.AD")
        print("4.AS")
        valid = False
        while not valid:
            q1 = input("Which curve slops upwards? ")
            if not validate(q1,"1234"):#MAKES SURE ONLY OPTIONS 1 TO 4 ARE ACCEPTED
                print("Answer is not valid")
            else:
                valid = True
        if q1 == "2":
            counter = counter+1#INCREMENTS COUNTER IF ANSWER IS CORRECT
        else:
            print("Incorrect")

        print("\n")#LEAVES A NEW LINE
        print("1.Leather")
        print("2.Pork")
        print("3.Gravy")
        print("4.Fish")
        valid = False
        while not valid:
            q2 = input("Which of the following products are in joint supply with meat?")
            if not validate(q2,"1234"):#MAKES SURE ONLY OPTIONS 1 TO 4 ARE ACCEPTED
                print ("Answer is not valid")
            else:
                valid = True
        if q2 == "1":
            counter = counter+1#INCREMENTS COUNTER IF ANSWER IS CORRECT
        else:
            print("Incorrect")

        print("\n")#LEAVES A NEW LINE
Ln: 1 Col: 0
```



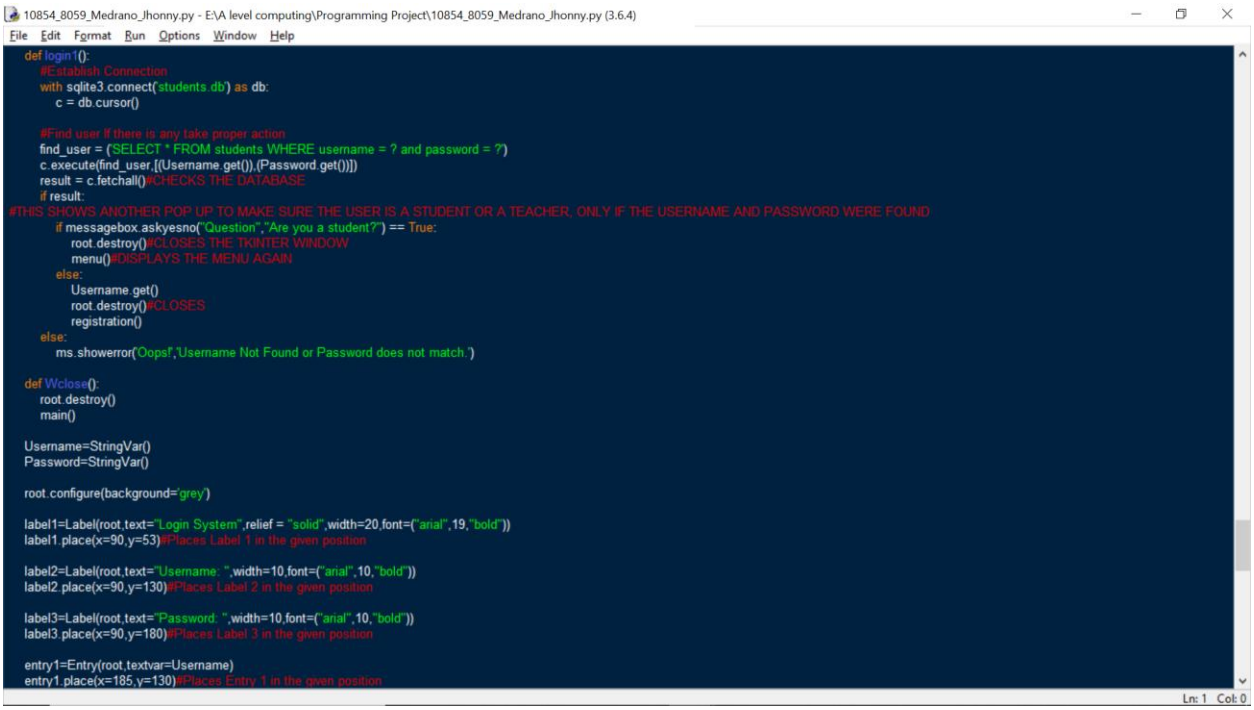
```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
    if q14 == "2":
        counter = counter+1#INCREMENTS COUNTER IF ANSWER IS CORRECT
    else:
        print("Incorrect")

    print("\n")#LEAVES A NEW LINE
    print("1.A shift to the right in the AD curve")
    print("2.A shift to the left in the short run AS curve")
    print("3.A shift to the right in the long run AS curve")
    print("4.A shift to the right in the sort run aggregate supply curve")
    valid = False
    while not valid:
        q15 = input("A fall in the cost of imported raw mateirals is likely to lead to")
        if not validate(q15,"1234"):#MAKES SURE ONLY OPTIONS 1 TO 4 ARE ACCEPTED
            print("Answer is not valid")
        else:
            valid = True
    if q15 == "4 ":
        counter = counter+1#INCREMENTS COUNTER IF ANSWER IS CORRECT
    else:
        print("Incorrect")
        print("\n")#LEAVES A NEW LINE


#THE COUNTER IS CHECKED BETWEEN THESE RANGES AND OUTPUTS FEEDBACK DEPENDING ON THEIR SCORE
    if 0 <= counter <=3:
        print("Don't worry you have plenty of time to revise and catch up, your score was",counter)
    elif 3< counter <=6:
        print("hehe, you're doing  well, keep it up, your score was",counter)
    elif 6 < counter <=9:
        print("Damn, keep working like this and you will get an A in that exam, your score was,",counter)
    elif 9 < counter <=12:
        print("Uhh, you were so close to getting full marks, your score was",counter)
    elif 12 < counter <=14:
        print("Alright Economics geek, we get it now, you're the best, you're score was",counter)
    elif counter == "15":
        print("Well, you my friend, got full marks, just go and take a day off")

    Create()#CALLS THE FUNCTION TO SAVE THE NAME AND THE SCORE TO CSV FILE
    menu()#DISPLAYS MENU AGAIN
```



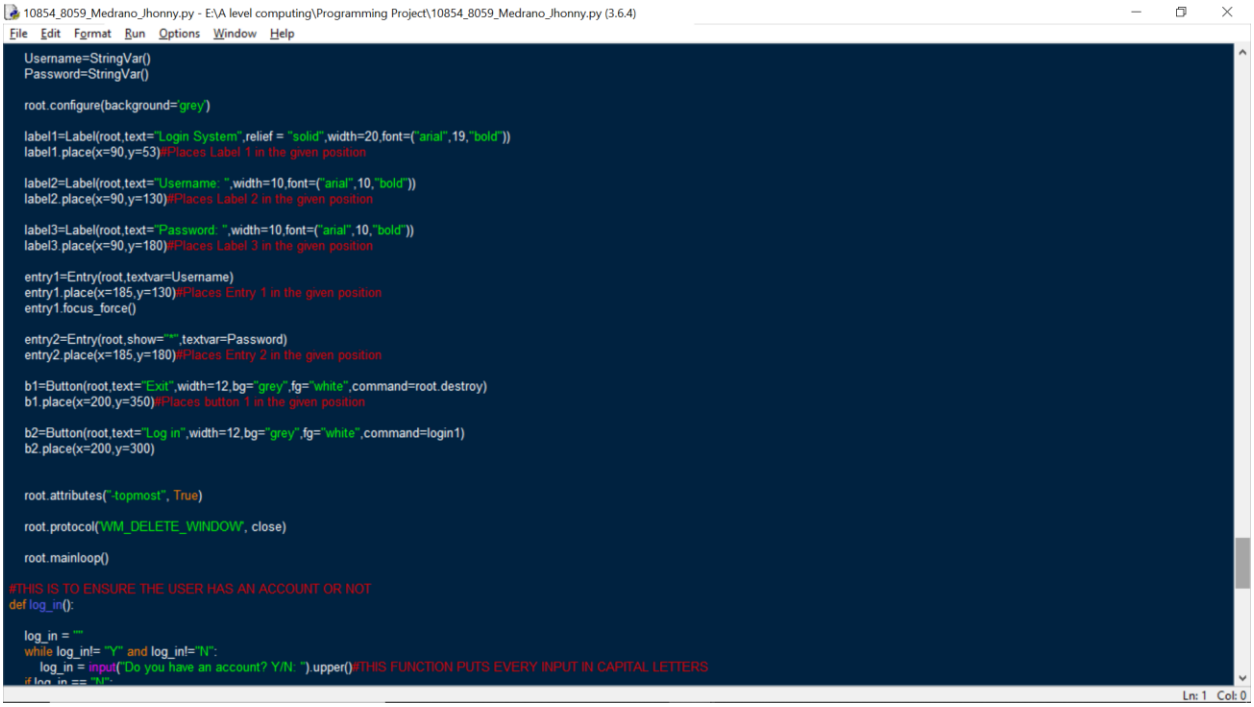
```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
#SHOWN IN MENU, USED TO OPEN REVISION WEBSITE
def Revision():
    new = 2
    url= "file:///E:/A%20level%20computing/Programming%20Project/check.html"#HTML NAME
    webbrowser.open(url,new=new)#REDIRECT TO THE HTML, MAKES SURE IT OPENS IT IN BROWSER


#SHOWS THE RESULTS WHEN THE USER CLICKS THE OPTION
def results():
    with open("Score.csv","r",newline="") as file:
        List = list(csv.reader(file))
        for row in List:
            print(row)

#IF USER EXISTS IT WILL REDIRECT THEM BACK THE LOGIN MENU
def last():
    root.destroy()#CLOSES MENU
    login()#OPENS LOGIN SYSTEM

#LAYOUT OF MENU
root= Tk()
root.geometry("250x250")
root.title("Menu")
root.configure(background='grey')

b1=Button(root,text="Start Quiz",width=12,bg="grey",fg="white",command=quiz)
b1.place(x=75,y=50)#Places button 1 in the given position

b2=Button(root,text="Access Revision Website",width=20,bg="grey",fg="white",command=Revision)
b2.place(x=50,y=100)

b3=Button(root,text="View Class Scores", width=20,bg="grey",fg="white",command =results)
b3.place(x=50,y=150)#Places button 1 in the given position

b4=Button(root,text="Exit",width=12,bg="grey",fg="white",command=last)
b4.place(x=75,y=200)#Places button 4 in the given position

root.attributes("-topmost", True)#PRIORITISES THE MENU, PUTS IT ON TOP OF EVERY OTHER WINDOW

root.mainloop()
```

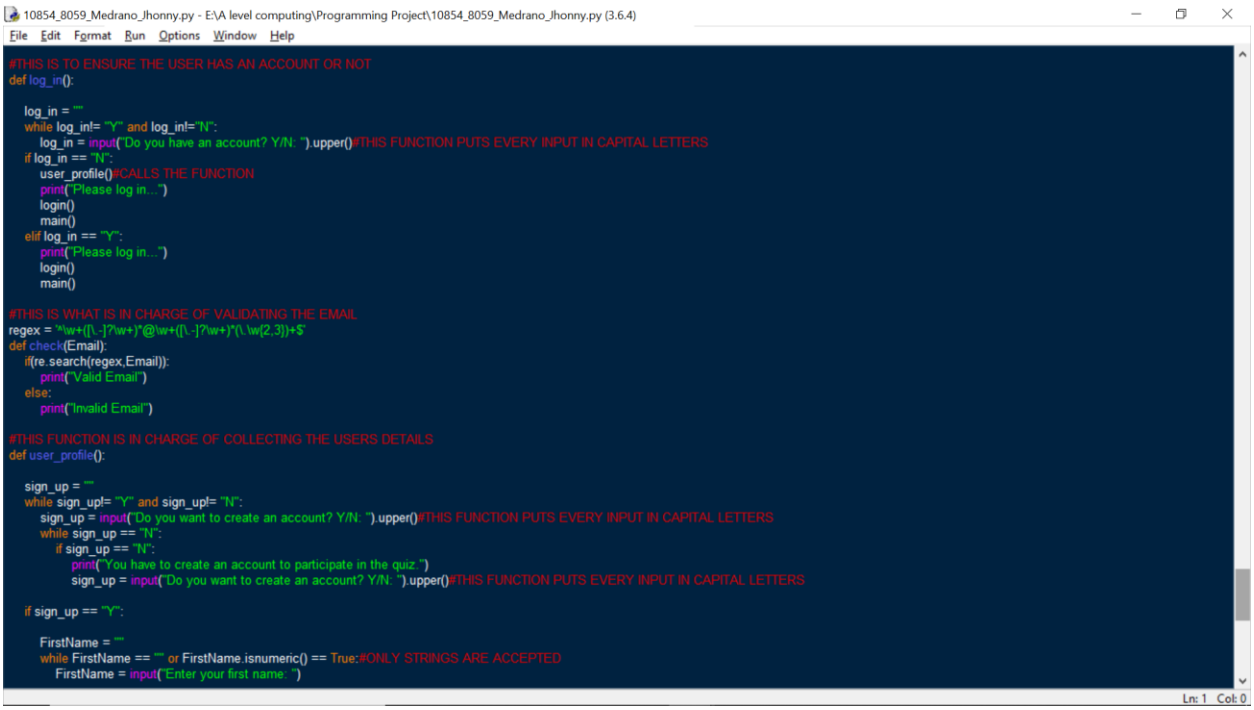


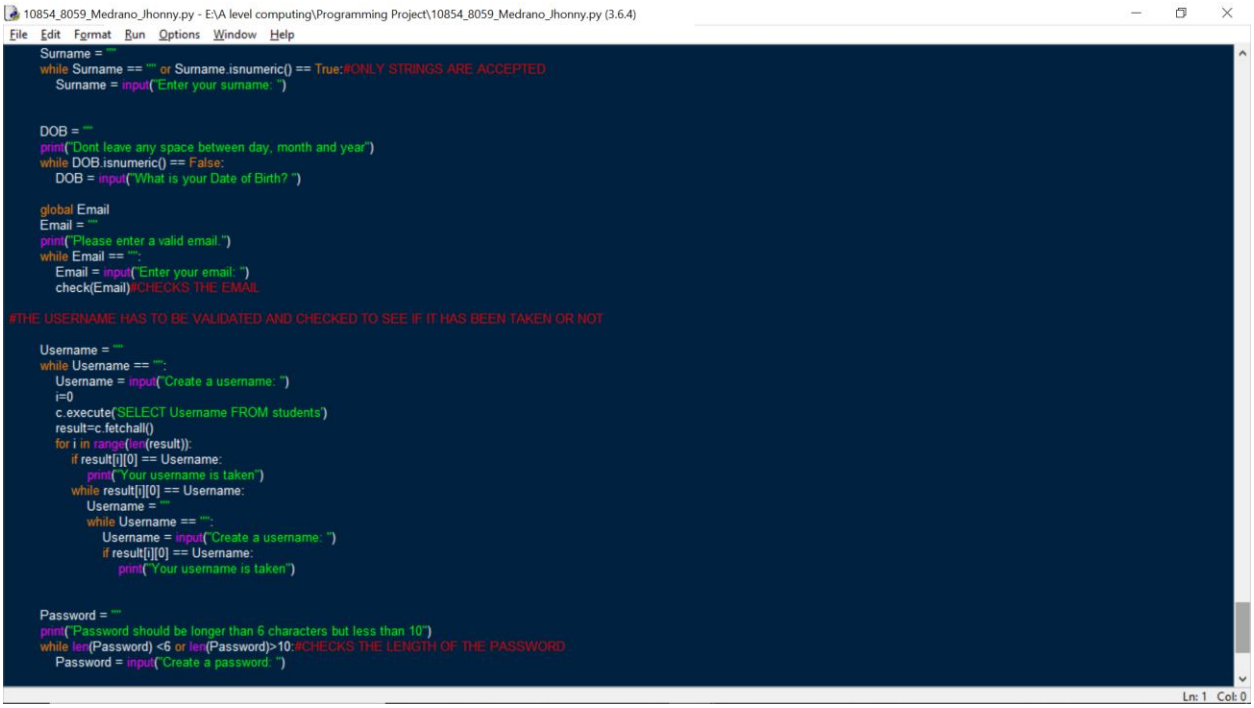
```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
def menu2():

    def results():
        with open("Score.csv","r",newline="") as file:
            List = list(csv.reader(file))
            for row in List:
                print(row)

    root= Tk()
    root.geometry("250x250")
    root.title("Furhter Options")
    root.configure(background='grey')

    b1=Button(root,text="Open Score Board",width=12,bg="grey",fg="white",command=results)
    b1.place(x=50,y=50)#Places button 1 in the given position

    b3=Button(root,text="Exit",width=12,bg="grey",fg="white",command=root.destroy)
    b3.place(x=50,y=150)#Places button 1 in the given position

    root.attributes("-topmost", True)#PRIORITISES THE MENU, PUTS IT ON TOP OF EVERY OTHER WINDOW

    root.mainloop()


def login():
    root =Tk()#Creates GUI
    root.geometry("500x600")#This establishes how big the window opened will be
    root.title("Economics Quiz.")#Title of the window that has been opened

#THIS IS THE POP UP THAT SHOWS WHEN THE USER WANTS TO EXIT, IT ASKS THEM TO CONFIRM
    def close():
        close = messagebox.askyesno("Economics Quiz","Confirm if you want to exit")
        if close == True:
            root.destroy()#CLOSES THE TKINTER OF THE LOGIN WINDOW
            main()#CALLS THE MAIN SUBROUTINE, STARTS THE PROCESS AGAIN
        else:
            pass#KEEPS THE TKINTER WINDOW OPEN

#THIS IS TO VALIDATE THE TKINTER ENTRIES WITHIN THE WINDOW OF THE LOGIN SYSTEM

    def login1():
Ln: 1 Col: 0
```



```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
    def login1():
        #Establish Connection
        with sqlite3.connect('students.db') as db:
            c = db.cursor()

        #Find user If there is any take proper action
        find_user = ('SELECT * FROM students WHERE username = ? and password = ?')
        c.execute(find_user,[(Username.get()),(Password.get())])
        result = c.fetchall()#CHECKS THE DATABASE
        if result:
#THIS SHOWS ANOTHER POP UP TO MAKE SURE THE USER IS A STUDENT OR A TEACHER, ONLY IF THE USERNAME AND PASSWORD WERE FOUND
            if messagebox.askyesno("Question","Are you a student?") == True:
                root.destroy()#CLOSES THE TKINTER WINDOW
                menu()#DISPLAYS THE MENU AGAIN
            else:
                Username.get()
                root.destroy()#CLOSES
                registration()
        else:
            ms.showerror('Oops!','Username Not Found or Password does not match.')

    def Wclose():
        root.destroy()
        main()

    Username=StringVar()
    Password=StringVar()

    root.configure(background='grey')

    label1=Label(root,text="Login System",relief = "solid",width=20,font=("arial",19,"bold"))
    label1.place(x=90,y=53)#Places Label 1 in the given position

    label2=Label(root,text="Username: ",width=10,font=("arial",10,"bold"))
    label2.place(x=90,y=130)#Places Label 2 in the given position

    label3=Label(root,text="Password: ",width=10,font=("arial",10,"bold"))
    label3.place(x=90,y=180)#Places Label 3 in the given position

    entry1=Entry(root,textvar=Username)
    entry1.place(x=185,y=130)#Places Entry 1 in the given position
Ln: 1 Col: 0
```

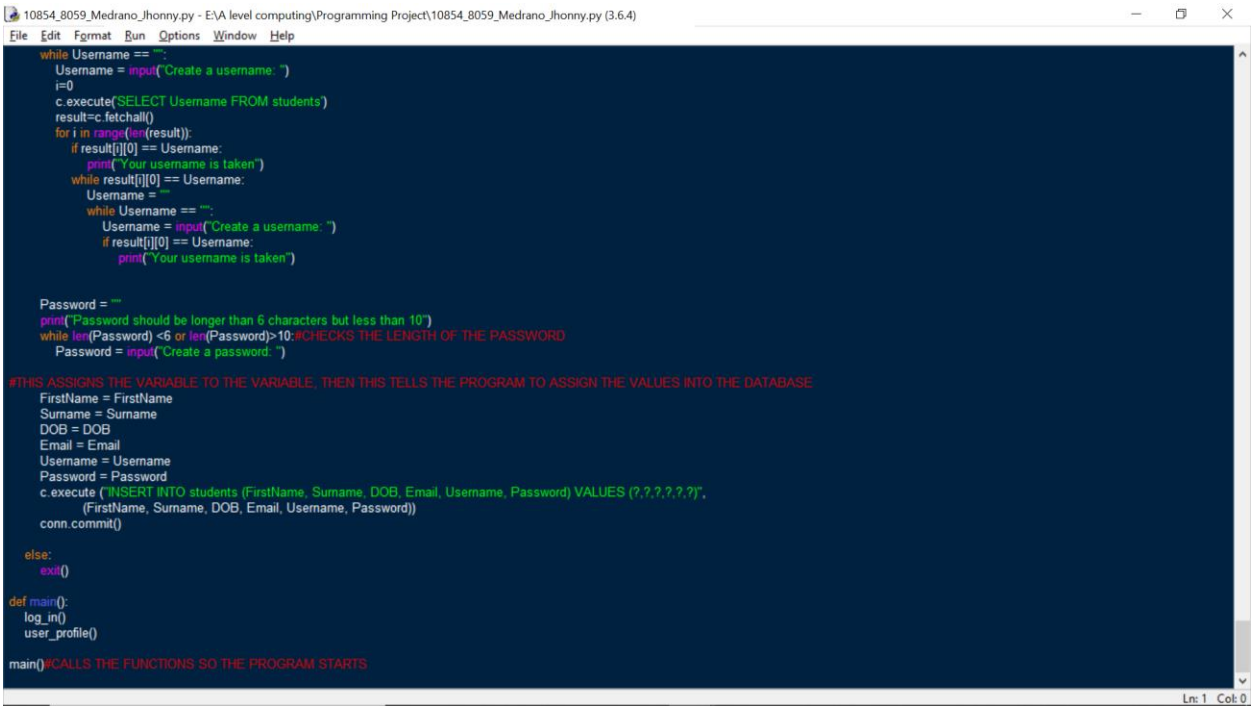

10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
Username=StringVar()
Password=StringVar()
root.configure(background='grey')
label1=Label(root,text="Login System",relief = "solid",width=20,font=("arial",19,"bold"))
label1.place(x=90,y=53)#Places Label 1 in the given position
label2=Label(root,text="Username: ",width=10,font=("arial",10,"bold"))
label2.place(x=90,y=130)#Places Label 2 in the given position
label3=Label(root,text="Password: ",width=10,font=("arial",10,"bold"))
label3.place(x=90,y=180)#Places Label 3 in the given position
entry1=Entry(root,textvar=Username)
entry1.place(x=185,y=130)#Places Entry 1 in the given position
entry1.focus_force()
entry2=Entry(root,show="*",textvar=Password)
entry2.place(x=185,y=180)#Places Entry 2 in the given position
b1=Button(root,text="Exit",width=12,bg="grey",fg="white",command=root.destroy)
b1.place(x=200,y=350)#Places button 1 in the given position
b2=Button(root,text="Log in",width=12,bg="grey",fg="white",command=login1)
b2.place(x=200,y=300)
root.attributes("-topmost", True)
root.protocol('WM_DELETE_WINDOW', close)
root.mainloop()
#THIS IS TO ENSURE THE USER HAS AN ACCOUNT OR NOT
def log_in():
log_in = ""
while log_in!= "Y" and log_in!="N":
log_in = input("Do you have an account? Y/N: ").upper()#THIS FUNCTION PUTS EVERY INPUT IN CAPITAL LETTERS
Ln: 1 Col: 0


10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
#THIS IS TO ENSURE THE USER HAS AN ACCOUNT OR NOT
def log_in():
log_in = ""
while log_in!= "Y" and log_in!="N":
log_in = input("Do you have an account? Y/N: ").upper()#THIS FUNCTION PUTS EVERY INPUT IN CAPITAL LETTERS
if log_in == "N":
user_profile()#CALLS THE FUNCTION
print("Please log in...")
login()
main()
elif log_in == "Y":
print("Please log in...")
login()
main()
#THIS IS WHAT IS IN CHARGE OF VALIDATING THE EMAIL
regex = '^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$'
def check(Email):
if(re.search(regex,Email)):
print("Valid Email")
else:
print("Invalid Email")
#THIS FUNCTION IS IN CHARGE OF COLLECTING THE USERS DETAILS
def user_profile():
sign_up = ""
while sign_up!= "Y" and sign_up!= "N":
sign_up = input("Do you want to create an account? Y/N: ").upper()#THIS FUNCTION PUTS EVERY INPUT IN CAPITAL LETTERS
while sign_up == "N":
if sign_up == "N":
print("You have to create an account to participate in the quiz:")
sign_up = input("Do you want to create an account? Y/N: ").upper()#THIS FUNCTION PUTS EVERY INPUT IN CAPITAL LETTERS
if sign_up == "Y":
FirstName = ""
while FirstName == "" or FirstName.isnumeric() == True:#ONLY STRINGS ARE ACCEPTED
FirstName = input("Enter your first name: ")
Ln: 1 Col: 0



```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
    Surname = ""
    while Surname == "" or Surname.isnumeric() == True:#ONLY STRINGS ARE ACCEPTED
        Surname = input("Enter your surname: ")


    DOB = ""
    print("Dont leave any space between day, month and year")
    while DOB.isnumeric() == False:
        DOB = input("What is your Date of Birth? ")

    global Email
    Email = ""
    print("Please enter a valid email.")
    while Email == "":
        Email = input("Enter your email: ")
        check(Email)#CHECKS THE EMAIL

#THE USERNAME HAS TO BE VALIDATED AND CHECKED TO SEE IF IT HAS BEEN TAKEN OR NOT

    Username = ""
    while Username == "":
        Username = input("Create a username: ")
        i=0
        c.execute('SELECT Username FROM students')
        result=c.fetchall()
        for i in range(len(result)):
            if result[i][0] == Username:
                print("Your username is taken")
            while result[i][0] == Username:
                Username = ""
                while Username == "":
                    Username = input("Create a username: ")
                    if result[i][0] == Username:
                        print("Your username is taken")


    Password = ""
    print("Password should be longer than 6 characters but less than 10")
    while len(Password) <6 or len(Password)>10:#CHECKS THE LENGTH OF THE PASSWORD
        Password = input("Create a password: ")
```

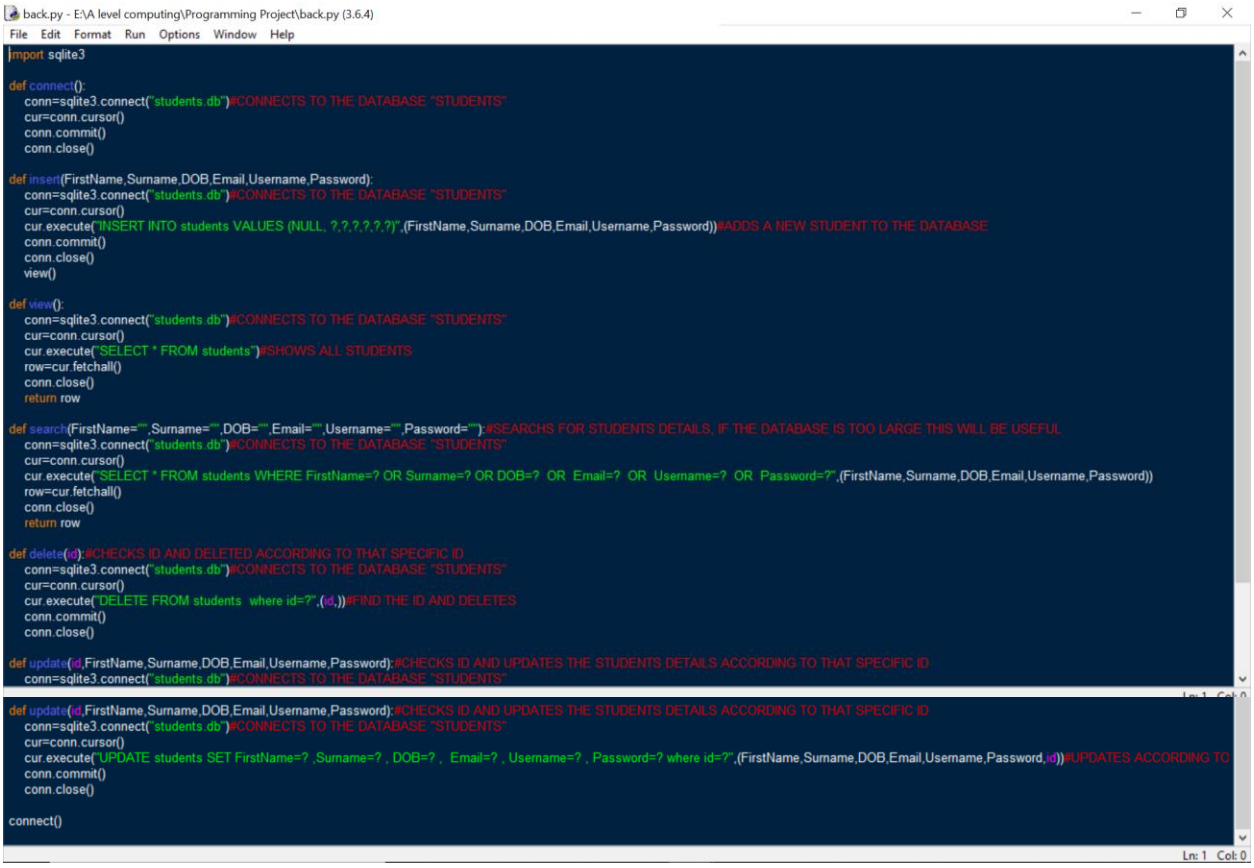
Ln: 1 Col: 0



```
10854_8059_Medrano_Jhonny.py - E:\A level computing\Programming Project\10854_8059_Medrano_Jhonny.py (3.6.4)
File Edit Format Run Options Window Help
    while Username == "":
        Username = input("Create a username: ")
        i=0
        c.execute('SELECT Username FROM students')
        result=c.fetchall()
        for i in range(len(result)):
            if result[i][0] == Username:
                print("Your username is taken")
            while result[i][0] == Username:
                Username = ""
                while Username == "":
                    Username = input("Create a username: ")
                    if result[i][0] == Username:
                        print("Your username is taken")


    Password = ""
    print("Password should be longer than 6 characters but less than 10")
    while len(Password) <6 or len(Password)>10:#CHECKS THE LENGTH OF THE PASSWORD
        Password = input("Create a password: ")

#THIS ASSIGNS THE VARIABLE TO THE VARIABLE, THEN THIS TELLS THE PROGRAM TO ASSIGN THE VALUES INTO THE DATABASE
    FirstName = FirstName
    Surname = Surname
    DOB = DOB
    Email = Email
    Username = Username
    Password = Password
    c.execute ("INSERT INTO students (FirstName, Surname, DOB, Email, Username, Password) VALUES (?,?,?,?,?,?)",
               (FirstName, Surname, DOB, Email, Username, Password))
    conn.commit()

else:
    exit()

def main():
    log_in()
    user_profile()

main()#CALLS THE FUNCTIONS SO THE PROGRAM STARTS
```

Ln: 1 Col: 0

SECOND PART OF CODE



```
back.py - E:\A level computing\Programming Project\back.py (3.6.4)
File Edit Format Run Options Window Help
import sqlite3

def connect():
    conn=sqlite3.connect("students.db")#CONNECTS TO THE DATABASE "STUDENTS"
    cur=conn.cursor()
    conn.commit()
    conn.close()

def insert(FirstName,Surname,DOB,Email,Username,Password):
    conn=sqlite3.connect("students.db")#CONNECTS TO THE DATABASE "STUDENTS"
    cur=conn.cursor()
    cur.execute("INSERT INTO students VALUES (NULL, ?,?,?,?,?,?)",(FirstName,Surname,DOB,Email,Username,Password))#ADDS A NEW STUDENT TO THE DATABASE
    conn.commit()
    conn.close()
    view()

def view():
    conn=sqlite3.connect("students.db")#CONNECTS TO THE DATABASE "STUDENTS"
    cur=conn.cursor()
    cur.execute("SELECT * FROM students")#SHOWS ALL STUDENTS
    row=cur.fetchall()
    conn.close()
    return row

def search(FirstName="",Surname="",DOB="",Email="",Username="",Password=""):#SEARCHS FOR STUDENTS DETAILS, IF THE DATABASE IS TOO LARGE THIS WILL BE USEFUL
    conn=sqlite3.connect("students.db")#CONNECTS TO THE DATABASE "STUDENTS"
    cur=conn.cursor()
    cur.execute("SELECT * FROM students WHERE FirstName=? OR Surname=? OR DOB=?  OR  Email=?  OR  Username=?  OR  Password=?",(FirstName,Surname,DOB,Email,Username,Password))
    row=cur.fetchall()
    conn.close()
    return row

def delete(id):#CHECKS ID AND DELETED ACCORDING TO THAT SPECIFIC ID
    conn=sqlite3.connect("students.db")#CONNECTS TO THE DATABASE "STUDENTS"
    cur=conn.cursor()
    cur.execute("DELETE FROM students  where id=?",(id,))#FIND THE ID AND DELETES
    conn.commit()
    conn.close()

def update(id,FirstName,Surname,DOB,Email,Username,Password):#CHECKS ID AND UPDATES THE STUDENTS DETAILS ACCORDING TO THAT SPECIFIC ID
    conn=sqlite3.connect("students.db")#CONNECTS TO THE DATABASE "STUDENTS"
Ln: 1   Col: 0
```

```
def update(id,FirstName,Surname,DOB,Email,Username,Password):#CHECKS ID AND UPDATES THE STUDENTS DETAILS ACCORDING TO THAT SPECIFIC ID
    conn=sqlite3.connect("students.db")#CONNECTS TO THE DATABASE "STUDENTS"
    cur=conn.cursor()
    cur.execute("UPDATE students SET FirstName=? ,Surname=? , DOB=? ,  Email=? , Username=? , Password=? where id=?",(FirstName,Surname,DOB,Email,Username,Password,id))#UPDATES ACCORDING TO
    conn.commit()
    conn.close()

connect()
Ln: 1   Col: 0
```

INTERVIEWS

Q. Does the quiz need to be animated?

The quizzes do not need to be animated, if it challenges us and teaches us something we did not know before.

No, this quiz is only for education and learning purposes, the animation will also distract the student and could make them get some answer wrong.

Q. Should my quiz have sound effects?

It would be more appealing, not for every single option but for the ones that are big like to start the quiz or finish the quiz.

Like I said before, the quiz should not distract the students while doing their test, this could distract them and can cause them to get a question wrong.

Q. Would you prefer to try the quiz again if you did not get the result you wanted?

Yes, I would like to try again because it would not be something, I should be proud of. I also would not want that to be my result. I think there needs to be clear instructions for when the game has ended if we want to start again.

Q. Should there be a timer?

I don't think a timer will be a good idea for the students because most of us don't work better under pressure we actually work worst because we will need more time or sometimes we wouldn't and waste time so it will be better if we could work at our own pace.

I wouldn't know how to answer this question because from my 15 years of experience, I have met students where they work much better under pressure because the adrenaline kicks in and get the energy which allows them to think much faster and perform better, but this isn't the same for everyone, some students work worst under pressure because they feel nervous about getting a question wrong and be put off.

Q. Would you like to see if the student tries again to get higher marks?

Absolutely, I want to see if they have another go they will improve their mark because it will notify me if they have tried and improve.

SECTION 2 OF INTERVIEW

Q. Do you feel more confident after testing yourself?

Yes, depends on the subject.

Q. Do you feel like you challenge yourself enough?

Sometimes, I feel like I cannot be asked to do much so no, not to my full potential

Q. From your experience do quizzes work with students?

I have tested my students for over 15 years including in a private school, they do help because students will remember the questions that they got wrong and will go and find out by themselves to find the actual answer. In other words, I think they are extremely helpful. Short and hard questions are more effective rather than long and simple questions.

SECTION 3 OF INTERVIEW

I then decided to create another quiz to another classmate, I changed the questions, this quiz gave me better knowledge of what my class struggled the most. The questions I asked are the following:

ME: Jhonny Medrano Classmate: Emily Voang

As an economic student what would you recommend testing other economists?

As an economics student, I would recommend testing other economists on why consumer behaviour varies to person to person because it is an important feature in the economy.

Which theme is harder? 1,2,3,4? (Choose 1). This will be the theme you'll be the tested on.

Out of the 4 themes, I would say theme 2 is the hardest so far, as it consists of heavy context and understanding. The ability to link policies and their drawbacks/ benefits can be difficult because there are so many possible outcomes.

Should I create another window for revision before the quiz begins?

I think that this is a good idea as it can remind me of topics I haven't done in a long time.

Should I include both diagrams and definitions in this quiz?

Diagrams and definitions would assist responses as it can be comforting to have something to when answering a question.

Should you be penalised if you get a question wrong?

I wouldn't want to be penalised if I got a question wrong as I already tend to doubt myself and would feel discouraged. A simple correction would be preferred

Would you like your score to be shown at the end of the quiz?

Yes, I would appreciate if my score was shown to me as I can use the scores to notice any improvements. It'd also be good if the score had a summary of what went if the score had a summary of what went well and what could be improved so I'd know exactly what I will need to work on.

Should the quiz be animated?

I wouldn't really pay attention much to whether it was animated or not. It could be a motivational aspect but it could also be distraction from the questions in the quiz.