**Introduction**

In this report, I will be describing what my project is, showing my planning, design of my coursework for the COMP1004 module and the eventual result. The project I've chosen is a single page web application of a typing test website, like available resources such as Monkeytype.com or typingtest.com.

**Project Vision**

The reason I am making this project beyond for a university project is tied to my personal goal to increase my typing since 2020, which lead to me learning how to use all ten of my fingers and applying constant practice to get the muscle memory.

The intended users for this project would be employers and learners. For learners, this could be an effective platform to practice your typing skills, with the stats giving feedback on the user’s progress, giving them some feedback to see how well they’re doing and how effective it is. By extension, employers could use this a way to test potential hirers’ typing test within an interview or a small quick external test. If they’re a user of this website beforehand and they’ve effectively practiced their typing skills, employer will view this positively and more likely to higher them.

This would be important in office jobs or stenographers where typing speed is crucial skill as it impacts efficiency of job performance in these types of jobs and tend to improve collaboration between team members, as communication tends to be written channels over email or similar ways and properly conveying their thoughts and needs quickly and properly results in improved teamwork and quickens the work efficiency. In the cases where accuracy is necessary, the website also fulfils this purpose as it does not accept a word be submitted until the word is correct, ensuring that if a mistake is made, you must go back and fix it. This means that in more administrative tasks such as filing databases or forms or customer service tasks, work can be far quicker and far more accurately with no extra costs.

**Ethical, Legal and Social Considerations**

This is limited due to the scope of my project being a university project but this that may want to consider is the legal concerns around the storage of account information. With my website, it uses Local storage which would be improper for an actual website due to its weak security. Although, the data stored is rather limited in importance as no personal data is stored perhaps only the password as users often reuse passwords. With this in mind, to mitigate this even a bit, hashing and strong passwords practices are enforced to ensure the user has some security.

There are a few social considerations to consider due to the harmless nature of this website but if the project would be introduced into the public, there might be some accessibility concerns, mainly adding a high contrast mode and other customization options for those with disabilities. In addition, it would be useful to add a way to send some feedback on the website as a social feature to allow it to grow more effectively.

A diagram of a game

Description automatically generatedHere are my user stories with my intended features:

**Use case scenarios – edit this**

Use case descriptions are a way to expand upon use cases, explain how they work, the

processes they follow, and conditions needed for the use case to run. I am using these to

further explain make and amend booking as it's easy to expand upon as the condition that

needs to be met is that a table needs to be available to make a booking. The template for

this is from UML @ Classroom book (Seidl, Brandsteidl, Huemer and Kappel, 2012)

|  |  |
| --- | --- |
| Name | page9image63141328Amend Booking |
| Short Description | Amend a booking |
| page9image35689536Precondition | page9image35689152Customer has already made a booking |
| Post Condition | Booking amended |
| Error Situations | There is not a free table at amended time |
| System state in the event of an error | Table cannot be booked at this time |
| page9image36638272Actors | page9image36642304Customer |
| page9image36649984Triggers | page9image36530880Customer needs to amend booking |
| Standard Process | 1. Customer enters booking details 2. System finds booking 3. Customer selects new date/time for  booking 4. Customer changes party size (if  needed) 5. System confirms there is a table free 6. Customer confirms booking |
| page9image36442432  Alternative Process | page9image36535296  5’ Table is not available 6’ System gives the next time slot for a free table 7’ Customer selects new time and confirms booking |

**Architecture**

**Software Development Lifecycle**

During this project, I am using the AGILE methodology, using the SCRUM model to achieve my project, having a scrum meeting at the beginning of every two weeks to outline what my aim is during the 2-week sprint. This ended up totalling up to 8 sprints in total, spanning 16 weeks.

The first week was primarily the planning phase, creating a ULM and flowchart to design the basic logic of the website with a wireframe of what the UI could look like in the end. This is below:

**A diagram of a program

Description automatically generatedA diagram of a program

Description automatically generated**

Here is my initial wireframes:

****

Here is my final wireframes:

In my second sprint,

Here are my sprints paired against the progression of the features included in my user stories:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Features | Sprint 1 | Sprint 2 | Sprint 3 | Sprint 4 | Sprint 5 | Sprint 6 | Sprint 7 | Sprint 8 |
| Able to measure WPM with a test |  | WPM/Timer visible in console, array not randomised | WPM/Timer displayed on website; array randomised and tested | Gathered user feedback, attempted word generation, failed. | Tested and implemented user feedback implemented, with positive reception |  |  |  |
| Sign in/Log in |  |  |  |  | Created HTML of login page, investigated JSON | Log in works after fetching data from JSON | Able to create an account which persists. | Properly tested. |
| Store stats and view them |  |  |  |  |  |  | Stats added to JSON, can be viewed with graph included. | Properly tested. |
| Export stats into JSON |  |  |  |  |  |  | Implemented and tested |  |
| Difficulty options |  |  |  | Timer options implemented | Difficulty options implemented | Difficulty options tested (fixed bugs) |  |  |
| Displaying correct/incorrect inputs |  |  |  | Feature implemented and tested. |  |  |  |  |
| Security features |  |  |  |  |  | Passwords hashed,  Passwords standards enforced | Tested properly |  |
| UI | Wireframe created |  | Wireframe implemented | Took user feedback | Wireframe redone, dark mode and light mode implemented. | Wireframe created for login/signup pages and implemented | Create wireframe for stats page and implemented. |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Not worked on | Started but limited progress (<50%) | Over half way done (>50%) | Nearly finished, if not finished, testing required. (>85%) | Finished (100%) |

|  |  |  |  |
| --- | --- | --- | --- |
| **SPRINT WEEK 1-2** | **SPRINT WEEK 3-4** | **SPRINT WEEK 4-6** | **SPRINT WEEK 7-8** |
| **Goals**  - Create and display user stories  - Create a flowchart to display the logic of website  (basic logic to be used in prototype)  - Create a wireframe using Figma (UI)  -Create UML for prototype logic | **Goals**  - Create a skeleton of the code  - To amend ULM + Flowchart to match actual logic if any mistakes  - Adhere to wireframe - Create a working prototype   * Can do a 60second typing test for array of words. * Gets WPM (words per minute) | **Goals**  - Create new ULM to show new structure  - Redo UI to fit the wireframe  - Display WPM + timer onto website rather than console  - Make correctly inputted words autodelete on submit  - Creation of proper testing  - randomised input array generation | **Goals**  **- I**mplement difficulty options  - Implement timer options  - Creation of DOM diagram  - Tell user typing test is over  - Take some user feedback  - Display errors on text live  - Change WPM calculation  - Put comments in .js file for better readability  - Attempt to use Word generation instead of using a randomised array |
| **Status**  - Completed  - Goals met | **Status**  - Completed  - Goals not met | **Status**  **-** Completed  - Goals met | **Status**  **-** Completed  - Goals not met |
| **Next steps**  - Using UML and flowchart as base create a working prototype  Features of prototype  -timer  -calculation of wpm  - typing the required words in an input box | **Next steps**  - Using UML and flowchart as base create a working prototype  Features of prototype  -timer  -calculation of wpm  - typing the required words in an input box | **Next Steps**  - Implementation of difficulty/timer options  - Letters go red if incorrect, or green if correct  - Creation of new Ulm to show structure  - Gather user feedback from close friends | **Next Steps**  - Creation of ULM to show new structure  - Show WPM live  - Make login system  - Implement user feedback  - Attempt to do word generation, limited success, hard to make genuine words |
| **SPRINT WEEK 9-10** | **SPRINT WEEK 11-12** | **SPRINT WEEK 13-14** | **SPRINT WEEK 15-16** |
| **Goals**  **-** Create ULM to show new structure  - Show live wpm and display it  - Make login system html, allowing user to go back  -Create new wireframe  **-** Clean UI up further (user feedback)  - Once implemented, take more user feedback  - implement a light/dark mode for UI  - Implement difficulty options effectively | **Goals**  - Allow to sign in with their details  - To be able to read from a Json file (holds passwords/users)  - Allow user to create new account and store it in Json file  - Allow password to be shown/hidden  - Allow creation of new user  - Ensure commenting has continued in new code  - Create wireframe log in/sign in/log out html additions and implement them  - Hash passwords for security  - Fix paragraphs not deleting properly bug | **Goals**  **-** Allow WPM to be stored and assigned to their user details  **-** Put theses stats within a Json that can be exported, using blob  - Create wireframe for page which displays theses stats  - Allow stats to be visible on webpage  - Replace alerts with nice UI implementation  - Style log in/sign in/log out html additions according to wireframe  - Allow WPM data to be displayed as a graph (using chart.js)  - Retest old features and add new features to testing | **Next Steps -** Final ULM  - Writing Report  - Add any more QOL features if desired, e.g. more stats in graph, ability to change username/password, ability to reset stats etc  - Do final testing to ensure all features are up to standard  - Get user feedback and fix any found errors |
| **Status**  **-** Complete  - Goals not met | **Status**  **-** Complete  - Goals not met, not possible to meet unless using node.js | **Status**  - Complete  - Goals not met | **Status**  **-**  Incomplete  - Goals not met |
| **Next Steps**  **-** Word/paragraph generation or grabbing it from the internet seems practical for a larger project but not needed for my scope.  - Allow to sign in with their details  - Allow WPM to be stored and assigned to their user details  - Allow it to be displayed as a graph | **Next Steps**  **-** Allow WPM to be stored and assigned to their user details  **-** Put theses stats within a Json that can be exported. Using blob  - Replace alerts with nice UI implementation  - style log in/sign in/log out html additions  - Allow WPM data to be displayed as a graph | **Next Steps -** Final ULM  - Writing Report  - Add any more features if wanted, e.g. more stats in graph, ability to change username/password, ability to reset stats etc | **Next Steps**  - Submission |

User feedback (1.1)

* Found it abrupt that words disappeared after every word, making typing harder as not as natural to do.
* UI could be cleaner.
* Desired for cleaner UI

User feedback (1.2)

* This was taken after user feedback initially implemented.
* Like the change that words are deleted at end of line
* User likes it far better, flows far nicer and UI is far cleaner.
* Only required slightly darker shades for background colour
* Suggested that background dimmed if typing test begins.

User feedback (1.3)

* Well, coloured for both dark and light mode.
* User found bug with starting a new test as a new user after finishing the first one.
  + Fixed shortly after
* Very responsive
* Easy to see and intuitive to use.
* Buttons colouring slightly out of style, stands out.
* Would be nice to have button to change light/dark mode instead of using system settings.

User feedback (2.1)

* Implemented CSS does not work on their laptop for some unknown reason.
* Potentially OS issue as they’re using windows 10, not windows 11 but can’t say for sure.
* Used on my device, User had similar problem to 1.1.
* Confused with paragraphs functions, renamed to sentences.

User feedback (2.2)

* This was taken much later than 1.2 due to timing.
* Appreciated the changes.
* Desired for more stats data
* Too little time to implement more stats data.

User feedback (3.1)

* Taken at the end of the sprints cycle, though seen it throughout as user been helpful in some of my coding challenges as they’re experienced with theses languages.
* Website is done well, dark mode is done well, looks clean.
* Not suitable for all resolutions/screen widths and/or heights not ideal
* Found visual bug, in dark mode, typing turns the text black which blends in with the background.
  + This has been fixed.
* Suggests adding delete account feature to show CRUD application, with any extra features would be great to add but not necessary for the scope of this project (such as more stats, able to change password etc)

User feedback (4.1)

* Taken at the end of the cycle.
* Disliked the visual of shifting words up, leading to mistyping, though minor issue and personal preference.
* Unhappy sign/login doesn’t work without VS code or the bat file.
* Appreciated the styling showing correct/incorrect.
* Overall, decent styling and functions well

**Challenges and constraints**

My primary challenge when doing my project is caused by coding as consequence of my lack of experience with the three languages we have to use: HTML, JavaScript, and CSS. HTML and CSS weren’t too difficult due to their rather simpler nature, but JavaScript took significant time to become component at. While the initial learning curve weren’t too high due to its sharing qualities with both C++ and python, my primary coding language.

What was the initial challenge was understanding how to effectively incorporate the JavaScript into my HTML, so it responds to my inputs using document.getElementById,the “onclick” / ”oninput” attributes and event listeners. Once I understood that, progress was steady.

This was halted in the fourth sprint when I attempted to style the words and reset words at the end of the line. This initially challenged me by figuring out how to style each letter individually, leading to the discovery of spans. Although, it wasn’t’ that simple. It required me to engage with new concepts such as query selector and nth child. This was arguably the hardest part of the coding for this project, requiring some ingenuity to solve the problems.

Next challenge of my coding adventure was inputting and exporting to a JSON which wasn’t hard by itself, but the hardest part was finding out the necessary information to apply it without using node.js as node.js seems so efficient as it that very few people do it. This required some help of the lectures in the lab sessions as other also struggled to solution, including the lecturers. This encouraged me to change my IDE from notepad++ to VS code as an extension called Live Preview circumvented this issue by hosting a local web server when displaying the website which is required to use fetch when getting data from the JSON. This is arguably the biggest constraint of my project, as without this or using the included bat file, it is impossible to login/signup.

A general challenge with this project was juggling it with other university projects and responsibilities, although through the use of the labs, I found it simpler to take those 4hours and work proficiently as much as I can during that time, getting lot of it done early making more time later.

A diagram of a computer

Description automatically generated with medium confidence

Package diagram, class diagram, sitemap