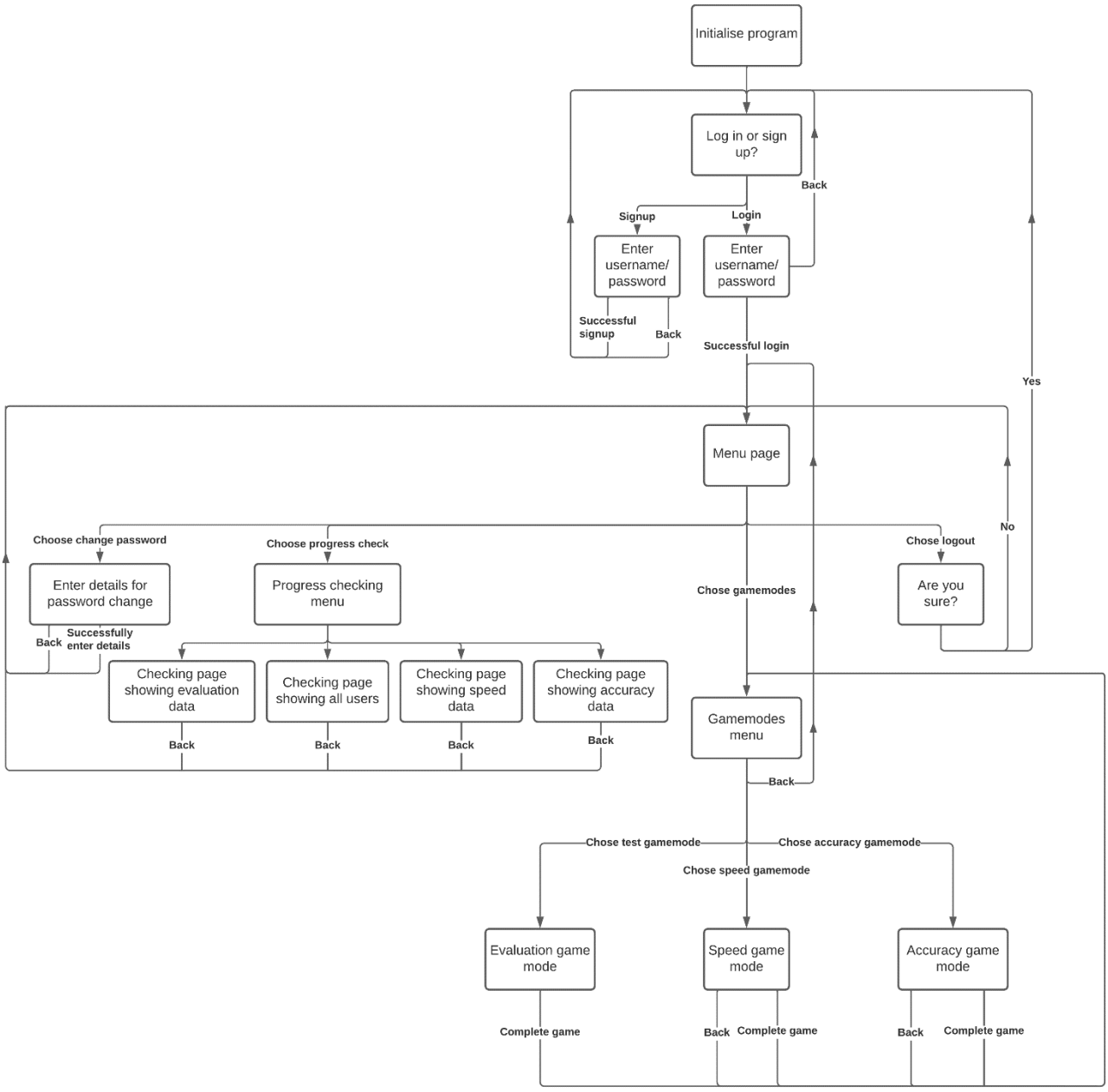
Criterion B: Design

Navigation

I plan for my program to include a web-page-like navigation system where you go between different windows, represented by the following diagram.

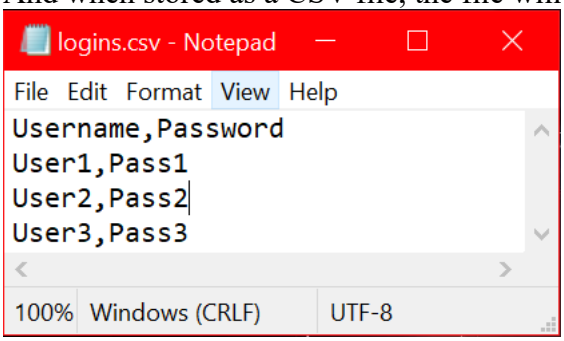


Login/signup page

The login/sign up pages must be able to access a database to store and check usernames and passwords. The CSV file will store data for the user's username and password, in the following format:

Username	Password
(User1)	*(Pass1)*
(User2)	*(Pass2)*
(User3)	*(Pass3)*
…	*…*

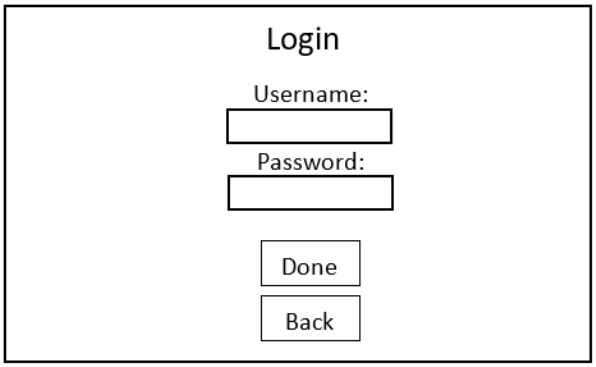
And when stored as a CSV file, the file will look like this:

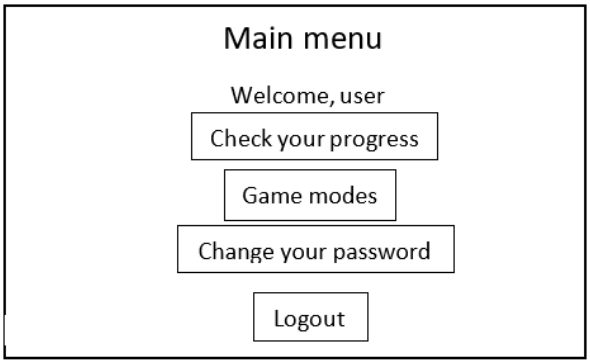


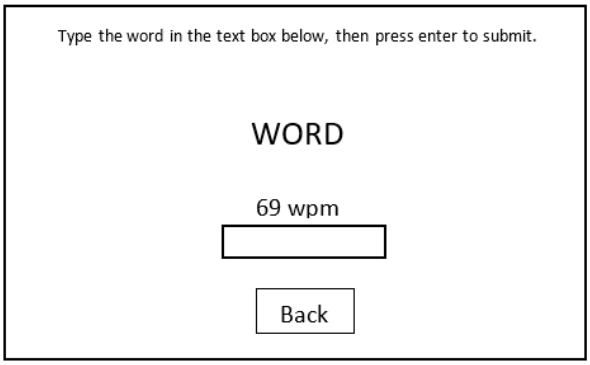
It contains two headings, which must be taken into account when accessing data from the array.

Page designs

The following show examples of how my application will look, for some of the different pages.





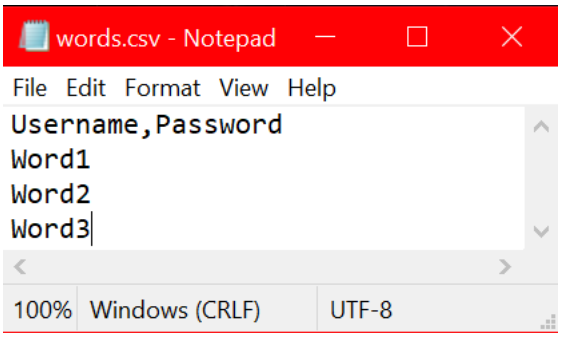


Game modes

Words file

A database of words can be accessed by each game mode.

Words
(Word1)
(Word2)
(Word3)
…



Evaluation file

Data for the "evaluation" game mode will be stored in three files, one for the total score and one for the speed/accuracy

Time/Date	Username	Score	Wpm
(Time and date 1)	*(User1)*	*(Score1)*	*(WPM1)* wpm
(Time and date 2)	*(User2)*	*(Score2)*	*(WPM2)* wpm
(Time and date 3)	*(User3)*	*(Score3)*	*(WPM3)* wpm
…	…	…	…

Speed file

########	*(User1)*	*(User2)*	…
(Word1)	*(Time)* of *(Number of times answered correctly)*	*(Time)* of *(Number of times answered correctly)*	…
(Word2)	*(Time)* of *(Number of times answered correctly)*	*(Time)* of *(Number of times answered correctly)*	…
(Word3)	*(Time)* of *(Number of times answered correctly)*	*(Time)* of *(Number of times answered correctly)*	…
…	…	…	…

Accuracy file

########	*(User1)*	*(User2)*	…
(Word1)	*(Number of times answered correctly)* of *(Total number of times answered)*	*(Number of times answered correctly)* of *(Total number of times answered)*	…
(Word2)	*(Number of times answered correctly)* of *(Total number of times answered)*	*(Number of times answered correctly)* of *(Total number of times answered)*	…
(Word3)	*(Number of times answered correctly)* of *(Total number of times answered)*	*(Number of times answered correctly)* of *(Total number of times answered)*	…
…	…	…	…

Testing

Success criteria	How to test it
Allow different users to create accounts and log into them	This can be tested by inputting an example username and password into the signup page, and then attempt to log in to the program using it. The data being saved can be checked by closing the program and attempting to log in again, or by simply opening the CSV file to check the data is there.
Allow the user to choose a mode to practise their typing	Attempting to navigate through the different game mode pages, ensuring that the user would be able to choose which mode to do.
Take user input from their keyboard	Checking whether the user is able to type into the given boxes on each game mode.
Recording which words/letters they get right/wrong the most	Playing each of the game modes, and then checking the CSV files afterwards to confirm that the data has been recorded (correctly).
Able to suggest words/letters more often if the user gets them wrong a lot	Playing the “test” game mode, answering them with a range of speed and accuracy, and then playing either the speed/accuracy game modes and recording how often each one is suggested, and seeing if that correlates to the speed/accuracy.
Show the user which words/letters they are best/worst at typing	Using the speed/accuracy progress checking pages, and seeing that data is presented there and that it is accurate.
Show the user how they have improved over time	Using the general progress checking page, and seeing that data is presented there and that it is accurate
Allow different users to compare their scores with each other	Completing the “test” game mode on multiple accounts and checking that one user can see both their own and other users’ scores on one window.