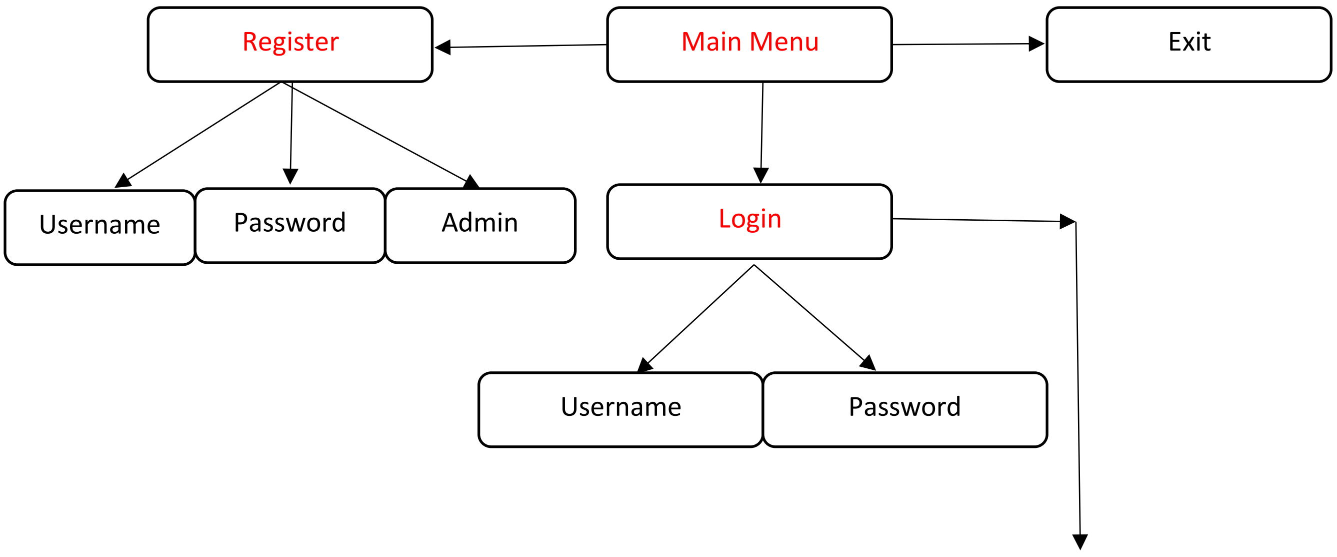
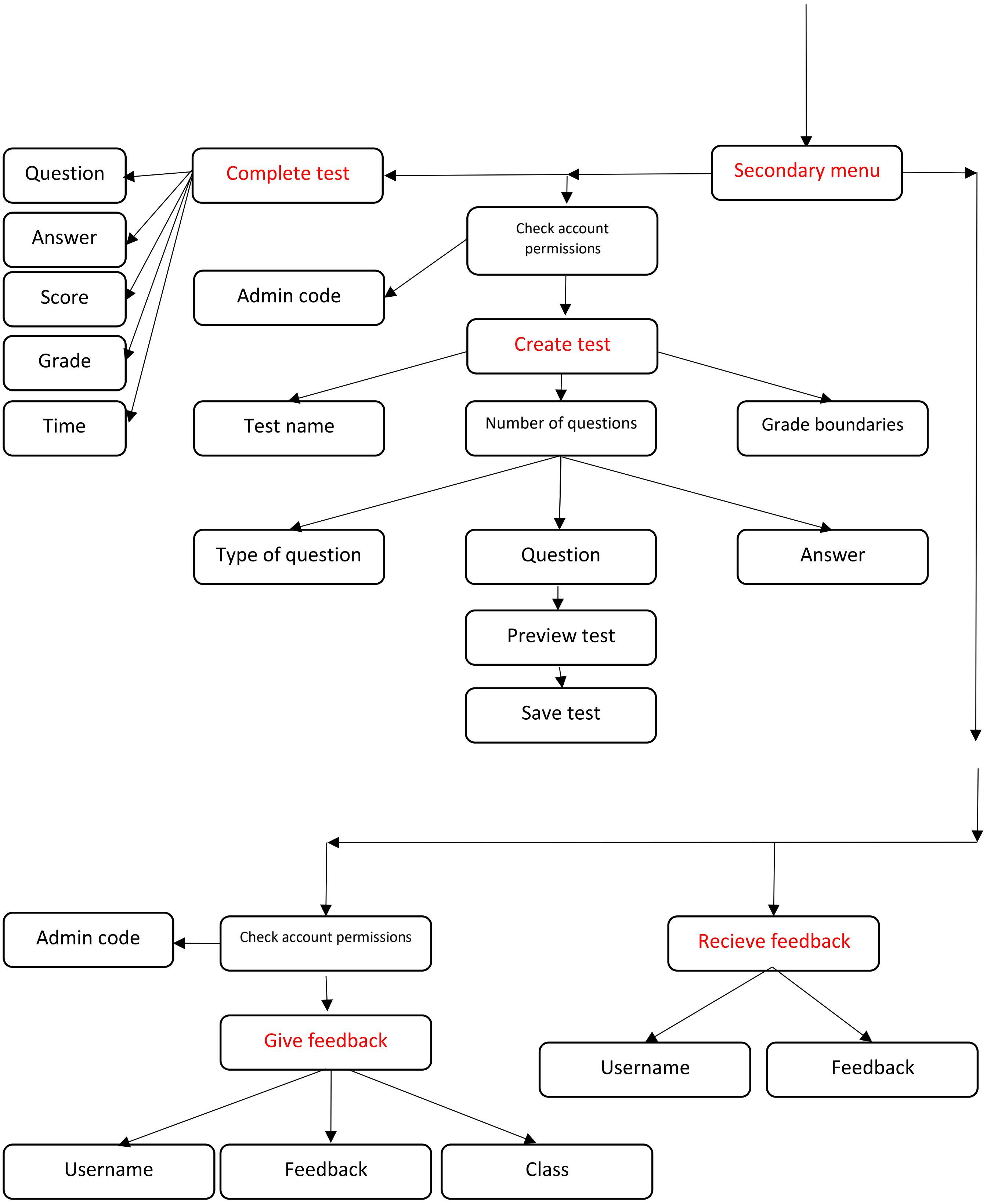
Design – Test creator

Decomposing the problem

To create the design to develop my program I will need to decompose my solution into smaller categories, to find these categories I will decompose the problem into a 'node layout'.





From the diagram I can decompose my problem into the following aspects:

1. Main menu (Allows the program to be accessed by allowing the user to either login or register)

2. Secondary menu (Allows all users to select between completing tests and receiving feedback, and allows users with admin permissions to also create tests and give feedback)
3. Registration (Allows the user to create an account)
4. Login (Verifies the user is allowed to use the program)
5. Creating a test (Allows tests to be created if the account has admin permissions)
6. Completing a test (Allows tests to be done by all users)
7. Giving feedback (Allows feedback to be written for a specific user or all users in a class)
8. Receiving feedback (Allows feedback to be read)

The main menu will give the user four options: to register an account, login using an already created account, get help, or exit the program. Both registering and logging in will open a new window with input boxes to enter both a username and password along with an enter button for the program to accept the inputs. Receiving help will print a pre-made help document with a guide for logging in and creating a new account. Exiting the program will simply close the program. The main menu is necessary to access several features of the program. **Otherwise, the user would not be able to show that they are authorised to use the program and without this the main features of the program will be inaccessible.**

The secondary menu will give the user the user six options: creating a test, completing a test, giving feedback, receiving feedback, get help and exit the program. The secondary menu is necessary to access the core features of the program, it is separate from the main menu as it requires a successful login to access. This is to prevent any unauthorised user from accessing the core features of the program. **Otherwise, any user can access the program. Also, without the secondary menu the user cannot access any part of the program.** Creating a test will open a new window with input boxes to enter a number of questions, a question and answer, along with a button for the program to accept the current question and allow another to be entered. This is opened in a new window as only users with authorisation to the admin features of the program can use this, **otherwise it could be accessed without verification of the user.** Completing a test will open a new window and give input boxes to select a test and enter an answer for the outputted questions. **Otherwise, the user cannot view the question and enter their answers.** Giving feedback will give input boxes to select a user or class and enter feedback for them. This is opened in a new window as only users with authorisation to the admin features of the program can use this, **otherwise it could be accessed without verification of the user.** Receiving feedback will output feedback linked to the username inputted at login. **Otherwise, the feedback would not be relevant to the user viewing it as it would not be the feedback aimed to them.** Receiving help will print a pre-made help document with a guide on what each button does and the features of the program. Exiting the program will simply close the program.

The program has been broken down in this way as each part represents a group of functions and GUI's that when put together, complete the process required e.g registering an account. **Otherwise, I will find it difficult to know what features should be prioritised and implemented first. Also, without this the problem is much more difficult to implement a solution for as I will be solving one large problem as opposed to smaller and easier sub-problems.**

Describe the solution

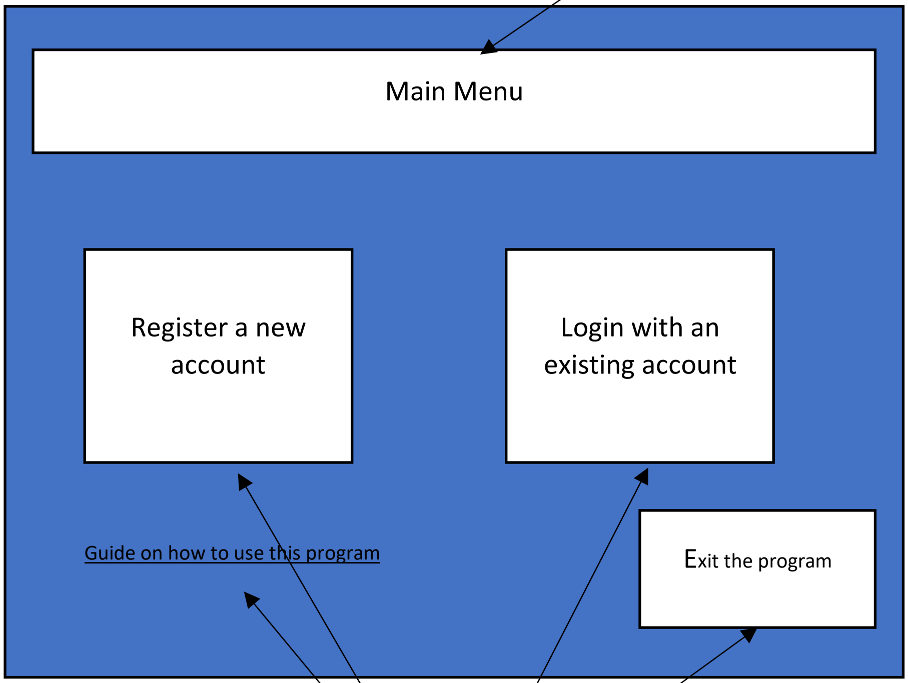
Main menu

Design

Purpose and justification: Allows the user to choose between registering an account and logging in with an existing account. This is suitable for the problem as this determines if the user has authorisation to the program and what level of authorisation they have. **Without this, any user can access the program, this is a security issue. Also, without this all users that have authorisation to the program can access special features that should only be used by users with admin status (teachers) such as creating a test and giving feedback.** The GUI's labels and buttons are suitable as it gives a visual representation of the options that are accessible, **without it the user would not be able to see and select the option they would like to access**. Furthermore, the buttons prevent the user from interacting with the program beyond the given buttons. **Without buttons that execute functions when clicked, the user could potentially access information that they should not be able to access or cause the program to crash.**

Without the diagram of the GUI, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.

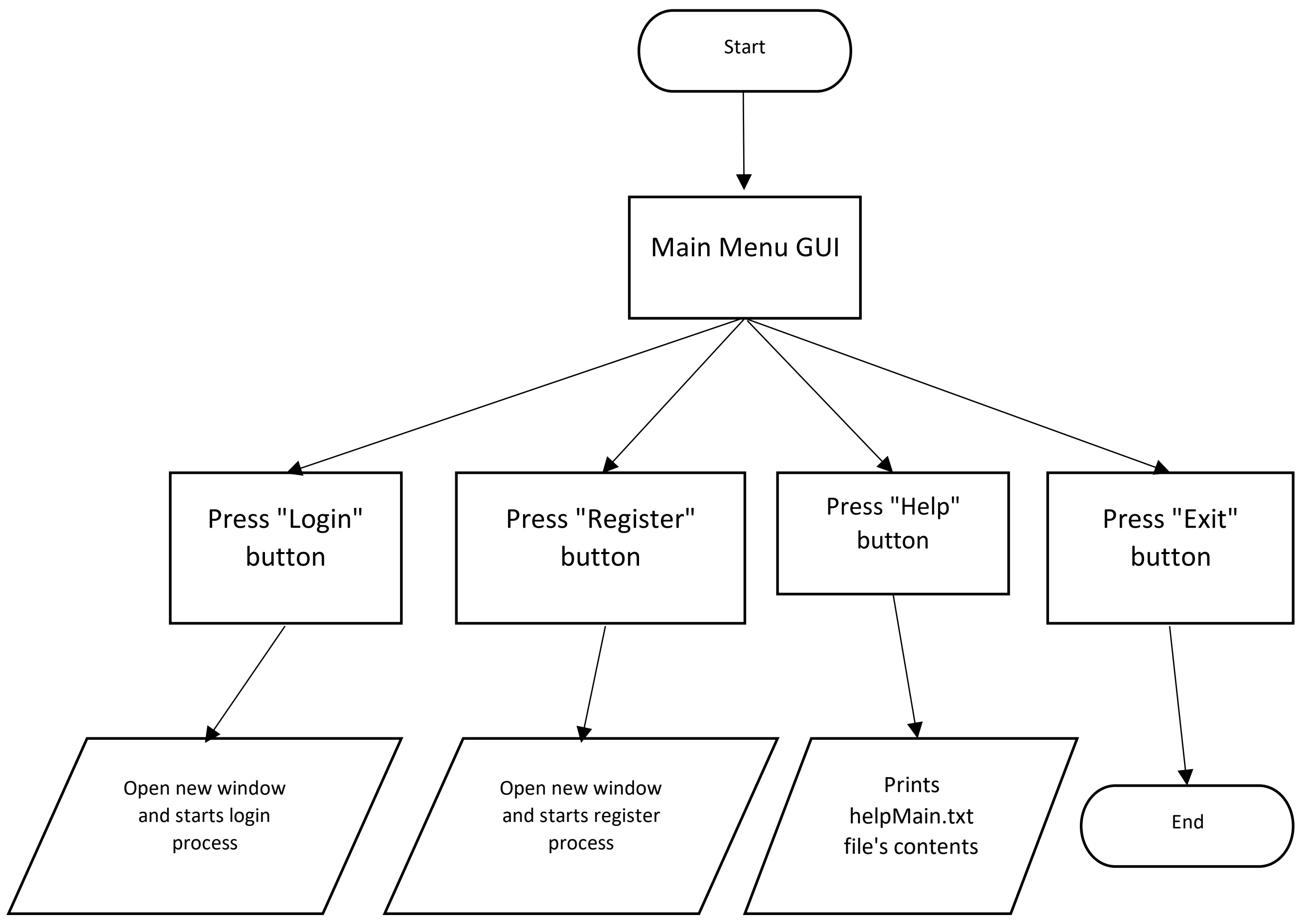
Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using**


Main Menu
Register a new account
Login with an existing account
Guide on how to use this program
Exit the program

Use of seperate main menu GUI meets the success criteria in: each system is windowed (1), and main menu (3)

Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** These buttons also meet the success criteria in: Register button (4), Login button (6), Guide button (9) and Exit button (11)

Flowchart


Start
Main Menu GUI
Press "Login" button
Press "Register" button
Press "Help" button
Press "Exit" button
Open new window and starts login process
Open new window and starts register process
Prints helpMain.txt file's contents
End

Pseudocode

```
OPEN main menu
IF login button pressed
    OPEN login
ELIF register button pressed
    OPEN register
ELIF help button pressed
    PRINT(help.txt)
ELIF exit button pressed
    END program
ENDIF
```

Input/Output table

Input	Function	Output
Press login	Starts login process	Starts login process
Press register	Starts registering process	Starts registering process

Press help	Gives help guide	Prints help.txt file
Press exit	Exits program	Ends program

Variables table

Name	Type	Validation	Description	Justification
loginButton	String	Button must be pressed. If the button is not pressed, then the process does not start	Starts login process	Allows user to start logging in. Allows only the process that the user wants to use to start. **Otherwise, the user would not be able to choose what feature they want to access**
registerButton	String	Button must be pressed. If the button is not pressed, then the process does not start	Once pressed starts registering process	Allows user to start registering an account. Allows only the process that the user wants to use to start. **Otherwise, the user would not be able to choose what feature they want to access**
helpButton	String	Button must be pressed. If the button is not pressed, then the help guide does not open	Once pressed gives help	Gives user guide on how to use the program, if they do not understand a part of it. **Otherwise, the user would not be able to choose what feature they want to access**
exitButton	String	Button must be pressed. If the button is not pressed, then the program does not close	Once pressed ends the program	Allows the user to exit the program, if they want to close it

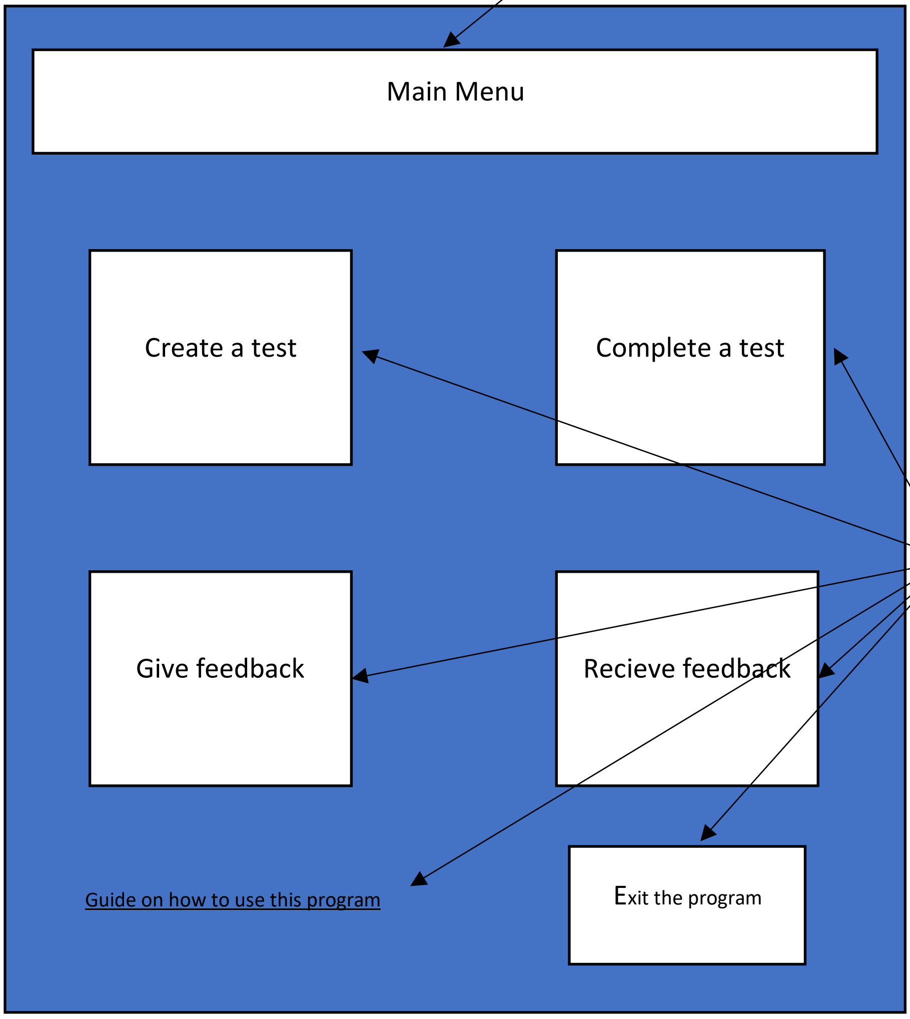
Secondary menu

Design

Purpose and justification: Allows the user to choose what feature of the program they want to access. This is suitable for the problem as this allows the user to select what feature they want to access. **Without this, the user would not be able to pick the option they want**. The GUI's labels and buttons are suitable as it gives a visual representation of the options that are accessible, **without it the user would not be able to see and select the option they would like to access**. Furthermore, the buttons prevent the user from interacting with the program beyond the given buttons. **Without buttons that execute functions when clicked, the user could potentially access information that they should not be able to access or cause the program to crash.** Also, the GUI allows only the buttons that the account logged in on has authorisation to, to be accessed. **Without this, admin features such as creating a test and giving feedback could be accessed without the correct level of authorisation.**

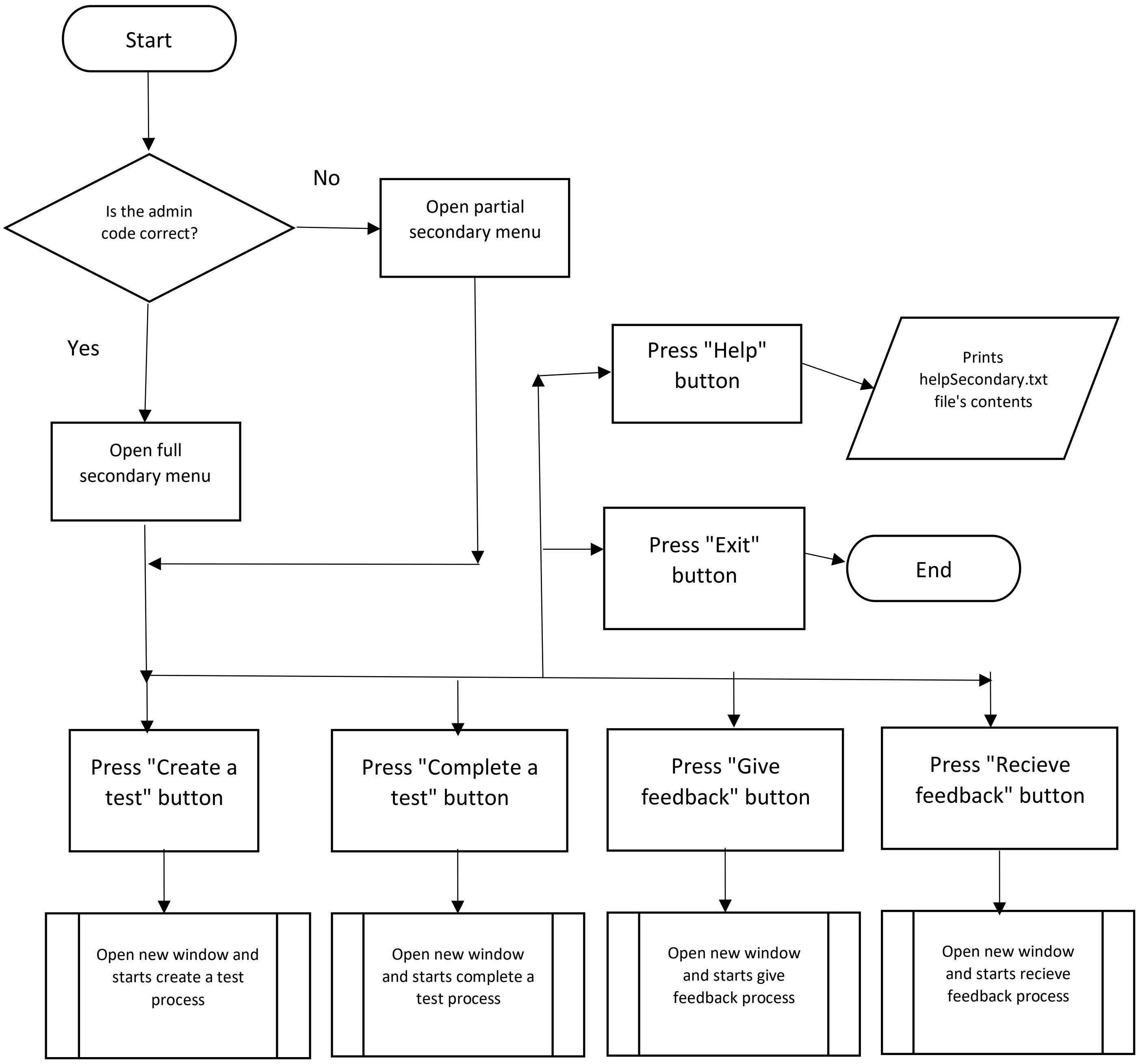
Without the diagram of the GUI, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.

Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using**. This GUI meets the succes criteria in: Each system is windowed (1) and Main menu (3)



Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know the function of each button.** These buttons meet the success criteria in: Guide button (9), Exit button (11), Create a test button (12), Complete a test button (18), Feedback sender button (21) and Feedback receiver button (24)

Flowchart


Start
Is the admin code correct?
No
Open partial secondary menu
Yes
Open full secondary menu
Press "Help" button
Prints helpSecondary.txt file's contents
Press "Exit" button
End
Press "Create a test" button
Press "Complete a test" button
Press "Give feedback" button
Press "Recieve feedback" button
Open new window and starts create a test process
Open new window and starts complete a test process
Open new window and starts give feedback process
Open new window and starts recieve feedback process

<u>Pseudocode</u>

```
IF inputtedAdminCode==”CorrectAdminCode1”
        OPEN secondary menu full
ELSE
OPEN secondary menu partial
ENDIF

IF create button pressed
        OPEN create
```

```
ELIF complete button pressed
    OPEN complete
IF give button pressed
    OPEN give
ELIF receive button pressed
    OPEN recieve
ELIF help button pressed
    PRINT(help.txt)
ELIF exit button pressed
    END program
ENDIF
```

<u>Input/Output table</u>

Input	Function	Output
User's admin code	Holds inputted admin code to be checked	Allows the admin features to be displayed if the account has admin status
Press create	Starts process to create a test	Starts process to create a test
Press complete	Starts process to complete a test	Starts process to complete a test
Press give	Starts process to give feedback	Starts process to give feedback
Press receive	Starts process to receive feedback	Starts process to receive feedback
Press help	Gives help guide	Prints help.txt file
Press exit	Exits program	Ends program

<u>Variables table</u>

Name	Type	Validation	Description	Justification
createButton	String	Button must be pressed. If the button is not pressed, then the process does not start	Starts process to create a test	Allows user to start creating a test. Allows only the process that the user wants to use to start. **Otherwise, the user would not be able to choose what feature they want to access**

InputtedAdminCode	String	None	Admin code needs to be checked	Needed to save admin code to login.csv file. **Otherwise, the user would not be able to authorise their account as an admin**
completeButton	String	Button must be pressed. If the button is not pressed, then the process does not start	Starts process to complete a test	Allows user to start completing a test. Allows only the process that the user wants to use to start. **Otherwise, the user would not be able to choose what feature they want to access**
giveButton	String	Button must be pressed. If the button is not pressed, then the process does not start	Starts process to give feedback	Allows user to start giving feedback. Allows only the process that the user wants to use to start. **Otherwise, the user would not be able to choose what feature they want to access**
recieveButton	String	Button must be pressed. If the button is not pressed, then the process does not start	Starts process to receive feedback	Allows user to start receiving feedback. Allows only the process that the user wants to use to start. **Otherwise, the user would not be able to choose what feature they want to access**
helpButton	String	Button must be pressed. If the button is not pressed, then the help guide does not open	Once pressed gives help	Gives user guide on how to use the program, if they do not understand a

				part of it. **Otherwise, the user would not be able to choose what feature they want to access**
exitButton	String	Button must be pressed. If the button is not pressed, then the program does not close	Once pressed ends the program	Allows the user to exit the program, if they want to close it

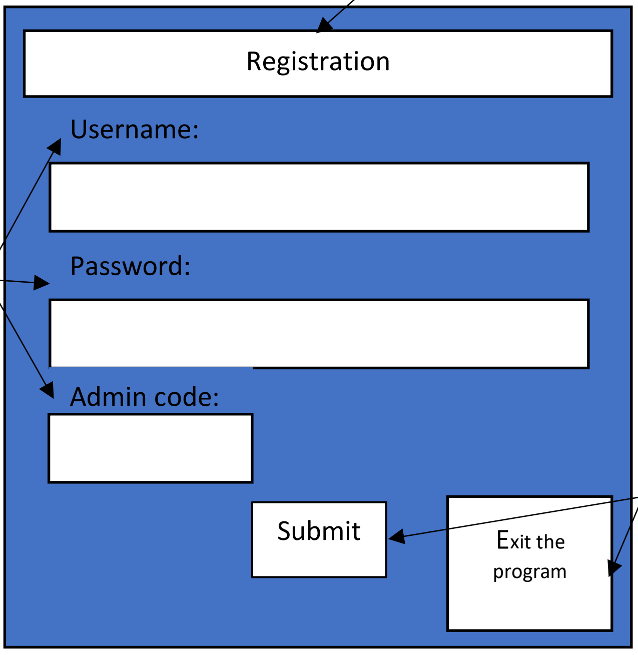
<u>Register</u>

<u>Design</u>

Purpose and justification: Allows the user to enter a username, password and admin code into input boxes. This is suitable for the problem as it allows the user's inputs to be accessed by the program. **Without this, the user would not be able to enter their desired username and password. Also, without this the program would not be able to read the user's inputs and save them to a database.** Furthermore, **without this the user would not be able to authorise their account as an admin.** The GUI's labels, buttons and input boxes are suitable as it gives a visual representation of the information required, **without it the user would not be able to see what information is required from them and where they should input their login details**. Furthermore, the button and input boxes prevent the user from interacting with the program beyond the given methods. **Without a button that executes the saving function, the user would not be able to save their login details and without the input boxes, the user would not be able to enter their login details.**

Without the diagram of the GUI, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.

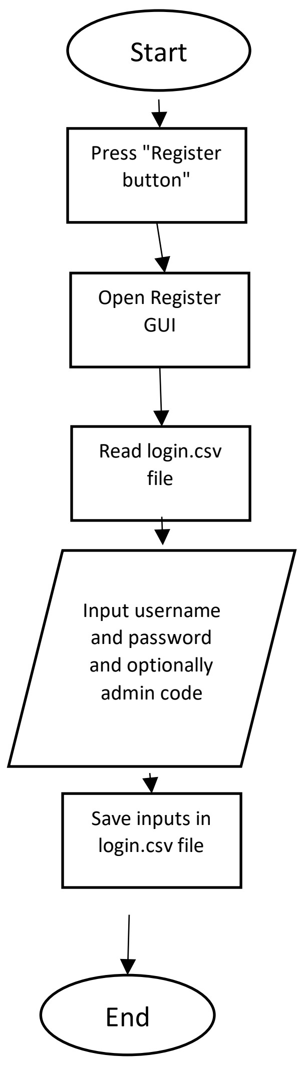
Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using.** This GUI meets the success criteria in: Each system is windowed (1) and Registration system (5)



Each input box has a label to make it clear what is to be inputted by the user. **Otherwise, the user would not know what information to enter into the input boxes.** These input boxes meet the success criteria in: Each input is provided using a keyboard (2)

Each button has a label to make the purpose of each button clear to the user. **Otherwise, the user would not know the function of the button.** These buttons meet the success criteria in: Exit button (11)

<u>Flowchart</u>


Start
Press "Register button"
Open Register GUI
Read login.csv file
Input username and password and optionally admin code
Save inputs in login.csv file
End

<u>Pseudocode</u>

```
INPUT register button pressed
OPEN login.csv file
DEFINE register function
    PRINT (“Input username”)
    PRINT (“Input password”)
    IF username AND password is NOT NULL
        ADD username to login.csv
        ADD password to login.csv
        ADD adminCode to login.csv
    ELSE
        OPEN register function
    ENDIF
```

```
SAVE .csv file
END register function
```

Input/Output table

Input	Function	Output
Press register	Starts registering process	Starts registering process
Login	Opens the login.csv file	Provides location for details to be saved
User's username	Holds inputted username to be saved	Allows user to create their own username
User's password	Holds inputted password to be saved	Allows user to create their own password
User's admin code	Holds inputted admin code to be saved	Allows user to give their account admin status, if they enter the correct code

Variables table

Name	Type	Validation	Description	Justification
registerButton	String	Button must be pressed, otherwise the process to save the account does not start	Once pressed starts registering process	Allows user to start registering an account. Allows the user to save their login details once they have filled the input boxes and are ready. **Otherwise, the user would not be able to register their account when they are ready.**
inputtedUsername	String	Input is not empty, input does not start with a number, input is at least 8 characters long. If the input does not meet the criteria the details will not be saved, and an error message will appear	Username to be registered	Needed to save username to login.csv file. Limiting character length and the first character, makes all usernames easy to find in a spreadsheet and appropriate as a username. **Otherwise, the user would not be able to create their own username**

InputtedPassword	String	Input is not empty, input does not start with a number, input is at least 8 characters long. If the input does not meet the criteria the details will not be saved, and an error message will appear	Password to be registered	Needed to save password to login.csv file. Only allows a strong password, increasing security. **Otherwise, the user would not be able to create their own password**
InputtedAdminCode	String	None	Admin code needs to be checked	Needed to save admin code to login.csv file. **Otherwise, the user would not be able to authorise their account as an admin**

Login

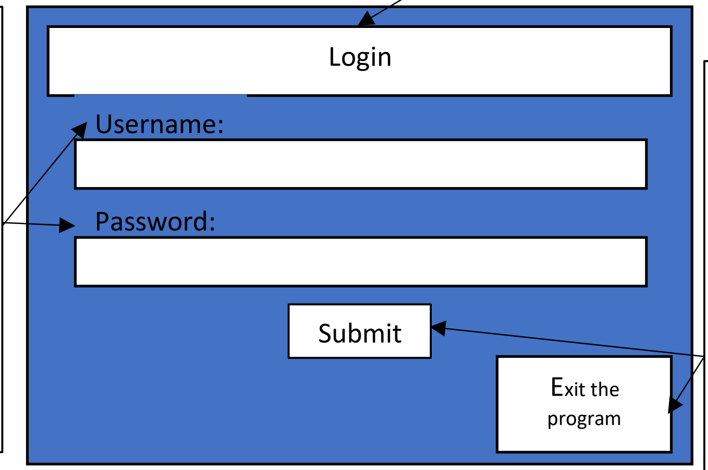
Design

Purpose and justification: Allows the user to enter a username and password into input boxes. This is suitable for the problem as it allows the user's inputs to be accessed by the program. **Without this, the user would not be able to enter their account's username and password**. Also, **without this the program would not be able to read the user's inputs and check the inputs against the database**. The GUI's labels, buttons and input boxes are suitable as it gives a visual representation of the information required, **without it the user would not be able to see what information is required from them and where they should input their login details**. Furthermore, the button and input boxes prevent the user from interacting with the program beyond the given methods. **Without a button that executes the check login information function, the user would not be able to verify their authentication details and without the input boxes, the user would not be able to enter their login details.**

Without the diagram of the GUI, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.

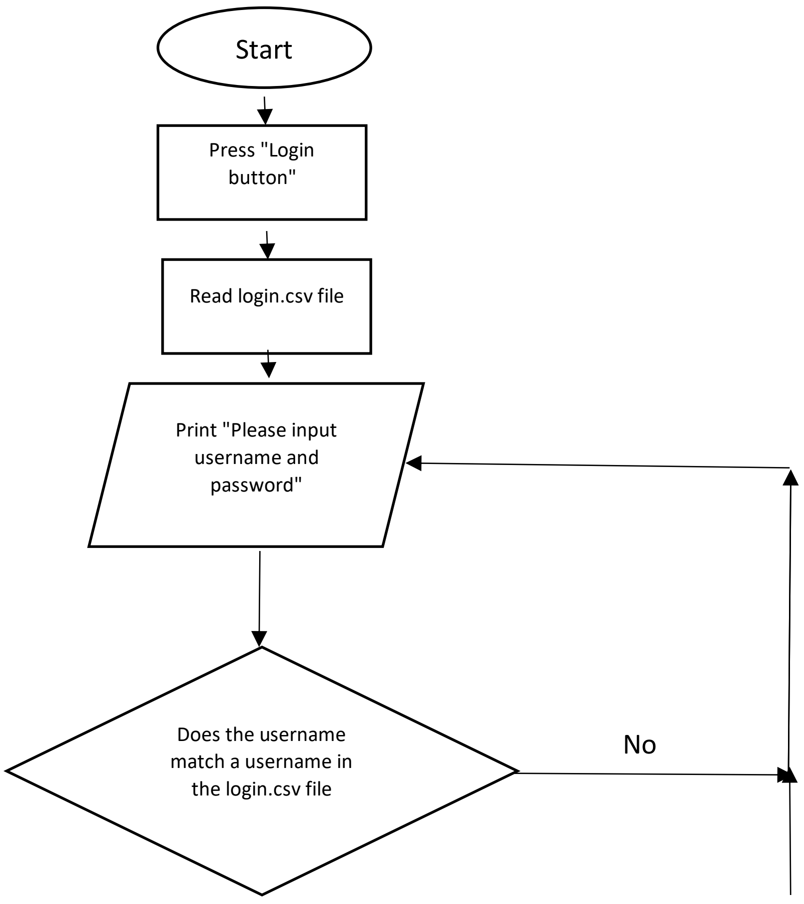
Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using.** This GUI meets the success criteria in: Each system is windowed (1) and Login system (7)

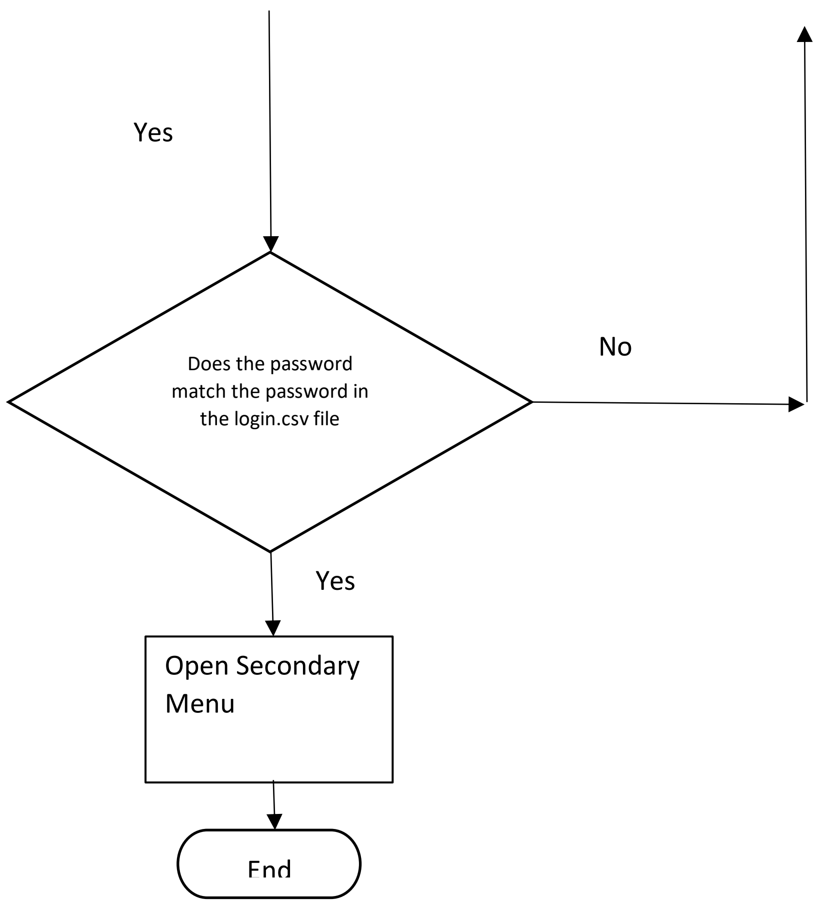
Each input box has a label to make it clear what is to be inputted by the user. **Otherwise, the user would not know what information to enter into the input boxes.** These input boxes meet the success criteria in: Each input is provided using a keyboard (2)



Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** These buttons meet the success criteria in: Exit button (11)

Flowchart


Start
Press "Login button"
Read login.csv file
Print "Please input username and password"
Does the username match a username in the login.csv file
No



Pseudocode

```
INPUT login button pressed
OPEN login.csv file
PRINT (“Input username”)
PRINT (“Input password”)
DEFINE login function(inputtedUsername,inputtedPassword)
FOR username in login.csv
        CHECK username
                IF username==inputtedUsername AND password==inputtedPassword
                        PRINT (“Login successful”)
                        OPEN main menu
                        BREAK
                ELSE
                        PRINT (“Login unsuccessful, try again”)
                        OPEN login function
```

```
            ENDIF
ENDFUNCTION login function
```

login function(inputtedUsername,inputtedPassword)

Input/Output table

Input	Function	Output
Press login	Starts login process	Starts login process
Login	Opens the login.csv file	Opens all saved login details
User’s username	Holds inputted username to be saved	Allows user to type their saved username
User’s password	Holds inputted password to be saved	Allows user to type their saved password

Variables table

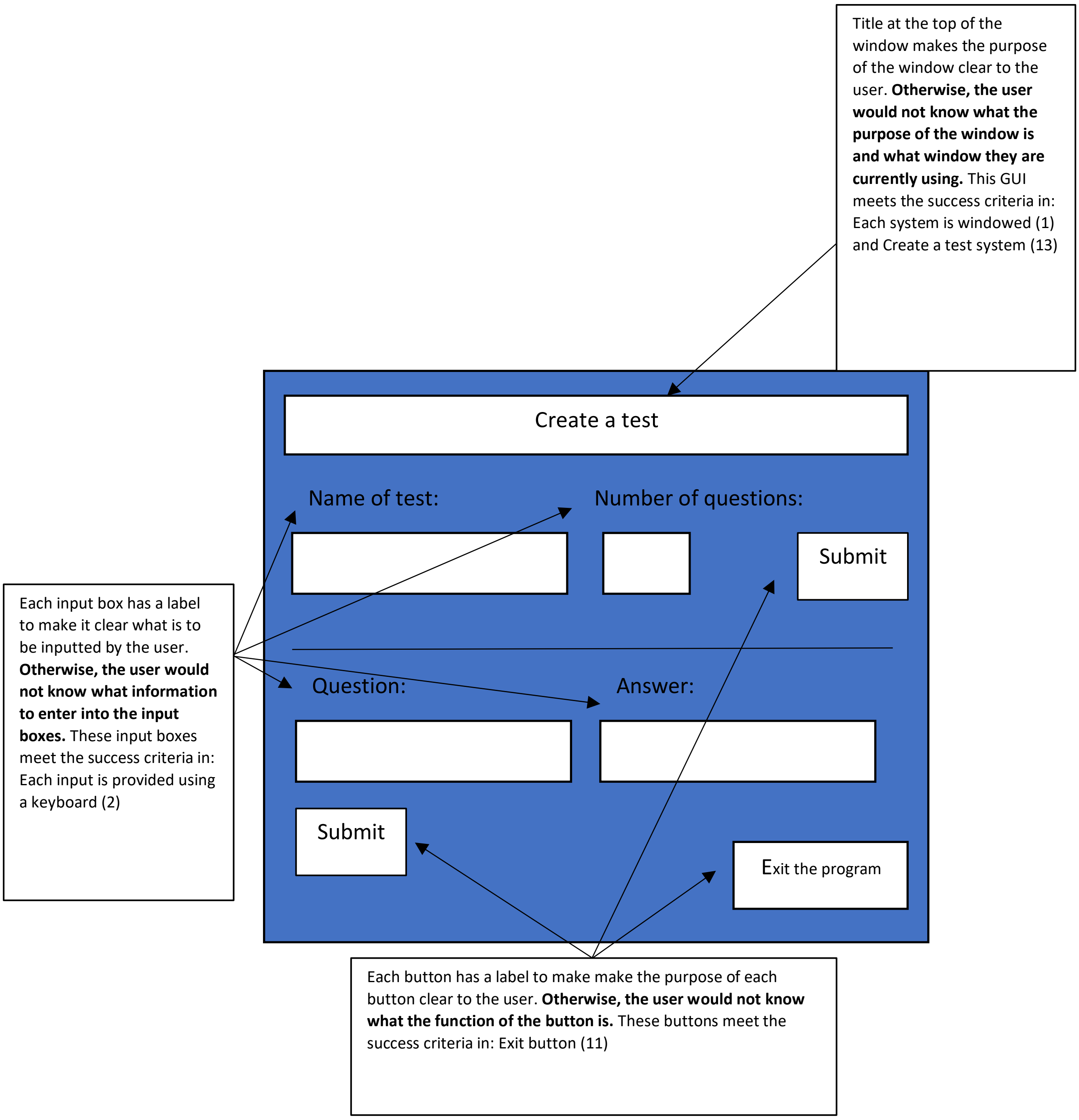
Name	Type	Validation	Description	Justification
loginButton	String	Button must be pressed, otherwise the process does not start, allows the user to start the process once they are ready	Starts login process	Allows user to start logging in. **Otherwise, the user would not be able to login when they are ready**
inputtedUsername	String	Input is not empty, input does not start with a number, input is at least 8 characters long. If the input does not meet the criteria the details will not be saved, and an error message will appear	Username to be registered	Needed to check if the username is in the login.csv file. **Otherwise, the user would not be able to enter their username for verification**
InputtedPassword	String	Input is not empty, input does not start with a number, input is at least 8 characters long. If the input does not meet the criteria the details will not be saved, and an error message will appear	Password to be registered	Needed to check if password matches username in login.csv file. **Otherwise, the user would not be able to enter their password for verification**

<u>Creating test</u>

<u>Design</u>

Purpose and justification: Allows the user to enter questions and answers into input boxes. This is suitable for the problem as it allows the user's inputs to be accessed by the program. **Without this, the user would not be able to enter their desired questions and answers**. Also, **without this the program would not be able to read the user's inputs and save them to a database**. The GUI's labels, button and input boxes are suitable as it gives a visual representation of the information required, **without it the user would not be able to see what information is required from them and where they should input the details of the test**. Furthermore, the button and input boxes prevent the user from interacting with the program beyond the given methods. **Without a button that executes the saving function, the user would not be able to save the current question and answer entered and without the input boxes, the user would not be able to enter the test information.**

Without the diagram of the GUI, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.

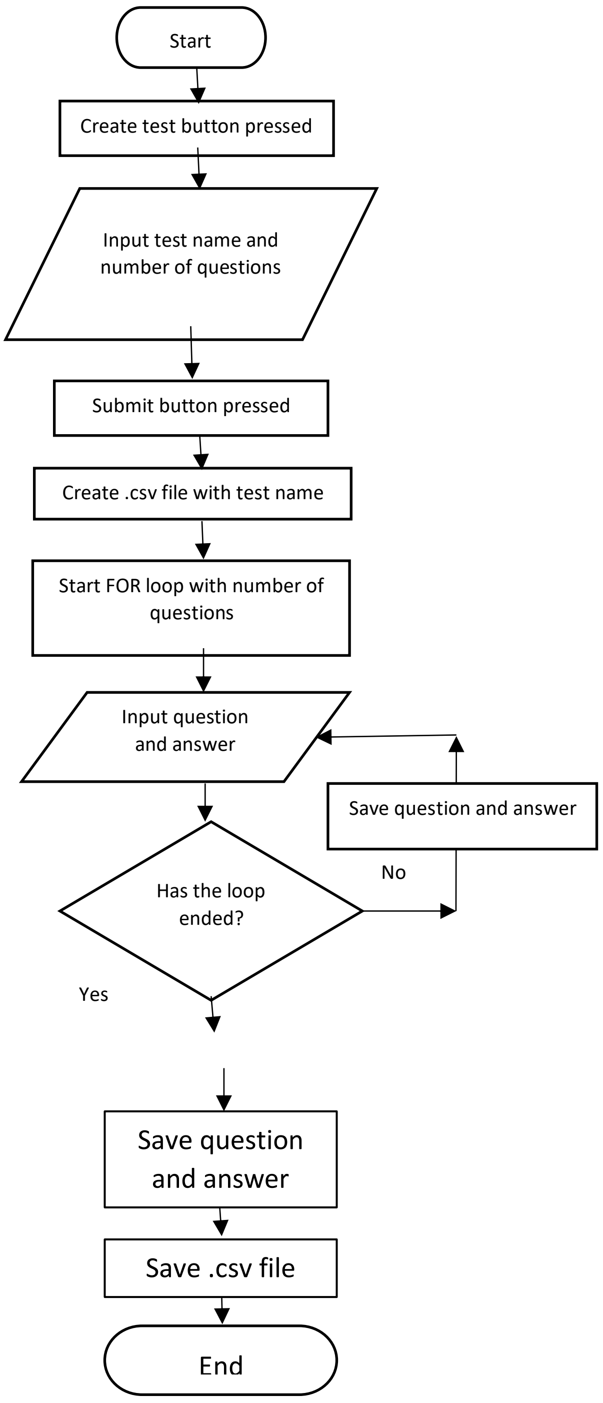


Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using.** This GUI meets the success criteria in: Each system is windowed (1) and Create a test system (13)

Each input box has a label to make it clear what is to be inputted by the user. **Otherwise, the user would not know what information to enter into the input boxes.** These input boxes meet the success criteria in: Each input is provided using a keyboard (2)

Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** These buttons meet the success criteria in: Exit button (11)

<u>Flowchart</u>


Start
Create test button pressed
Input test name and number of questions
Submit button pressed
Create .csv file with test name
Start FOR loop with number of questions
Input question and answer
Save question and answer
No
Has the loop ended?
Yes
Save question and answer
Save .csv file
End

<u>Pseudocode</u>

```
INPUT name of test
CREATE name of test.csv file
OPEN name of test.csv file
PRINT (“Input number of questions to add”)
FOR 0 to inputtedNumberOfQuestions
        PRINT (“Input question”)
        PRINT (“Input answer”)
        ADD question to .csv file
        ADD answer to .csv file
SAVE .csv file
```

Input/Output table

Input	Function	Output
Press create	Starts process to create a test	Starts process to create a test
Name of test	Holds the name of the test	Allows the user to enter a name for the test file
Test	Opens newly created test.csv file	Allows the user to add to the file
Number of questions	Holds the number of questions to be added	Allows the user to add a specific number of questions and answers
Question	Holds the question	Allows the user to enter a question
Answer	Holds the answer	Allows the user to enter an answer
Enter	Allows the question to be saved and the next question to be entered	Saves previous question and answer and allows a new question and answer to be entered

Variables table

Name	Type	Validation	Description	Justification
createButton	String	Button must be pressed, otherwise the process does not start, allows the user to start the process once they are ready	Starts process to create a test	Allows user to start creating a test. **Otherwise, the user would not be able to start creating a test when they are ready**
inputtedNameOfTest	String	Input is not empty, if the input does not meet the criteria an error message will appear	Name of the test file to be created	Needed to differentiate between tests. **Otherwise, the user would not be able to name their test**
inputtedNumberOfQuestions	Integer	Input is not empty, input is a number, if the input does not meet the	Number of questions to be added	Needed to determine the length of the for loop. **Otherwise, the user would**

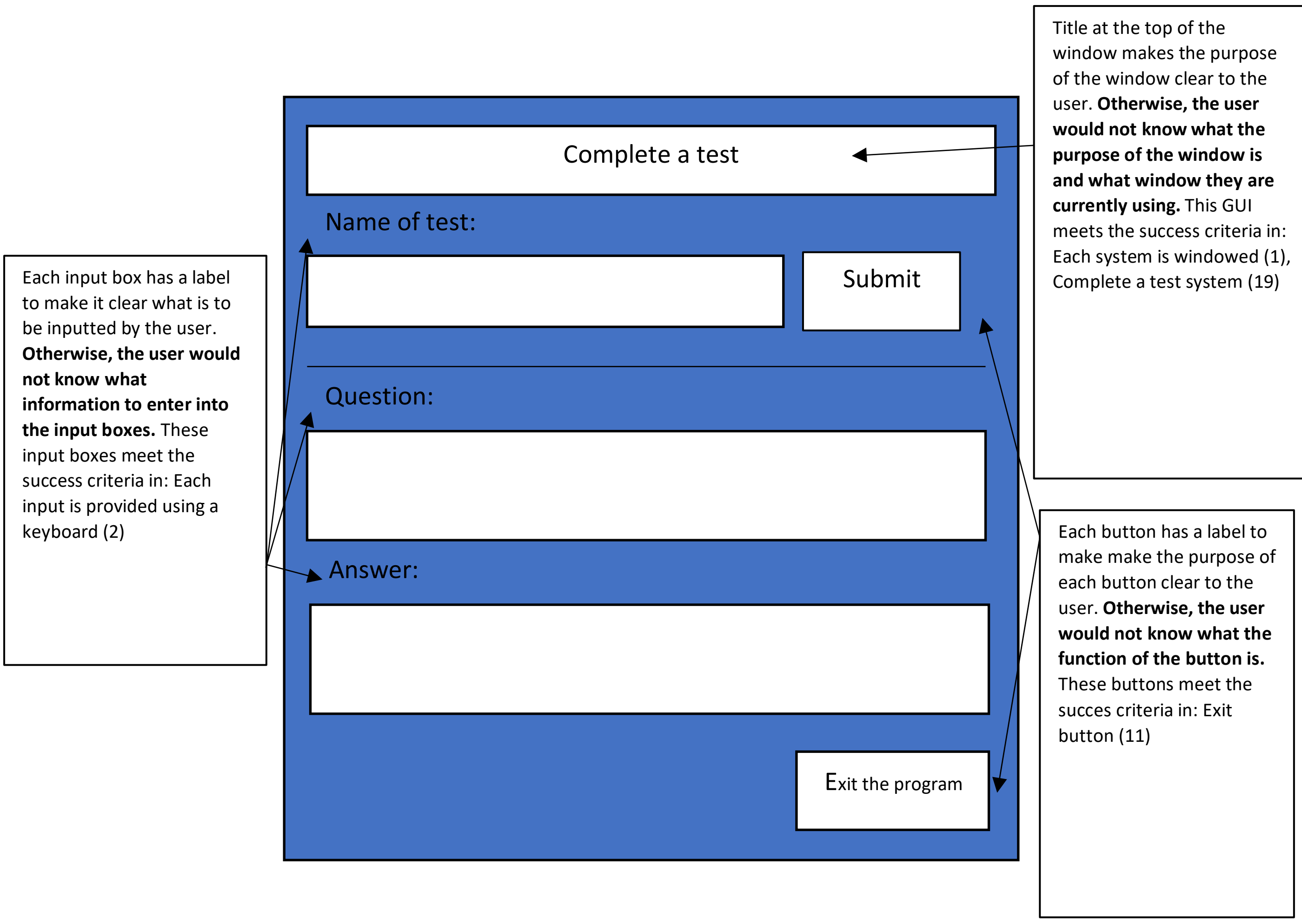
		criteria an error message will appear		**not be able to enter how long the test will be**
inputtedQuestion	String	Input is not empty, if the input does not meet the criteria an error message will appear	User's question to be added	Needed to allow the user to enter a question. **Otherwise, the user would not be able to enter a question for the test**
inputtedAnswer	String	Input is not empty, if the input does not meet the criteria an error message will appear	User's answer to be added	Needed to allow the user to enter an answer. **Otherwise, the user would not be able to enter an answer for the test**
enterButton	String	Button must be pressed, otherwise the process does not start, allows the user to start the process once they are ready	Button to save current question and allow another question to be added	Needed to save current question and allow another to be entered. **Otherwise, the user would not be able to save the question and answer when they are ready and have filled both input boxes**

Completing test

Design

Purpose and justification: Allows the user to enter their answer into the input box. This is suitable for the problem as it allows the user's inputs to be accessed by the program. **Without this, the user would not be able to enter their answer**. Also, **without this the program would not be able to read the user's inputs and save them to a database**. The GUI's labels, button and input boxes are suitable as it gives a visual representation of the information required, **without it the user would not be able to see what information is required from them and where they should input the details of the test.** Also, the question text box allows questions to be inserted into the GUI, **without this the question would not be visible on the GUI and would be difficult for a user to answer**. Furthermore, the button and input boxes prevent the user from interacting with the program beyond the given methods. **Without a button that executes the saving function, the user would not be able to submit their current answer and view the next question and without the input boxes, the user would not be able to enter the test information.**

Without the diagram of the GUI, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.

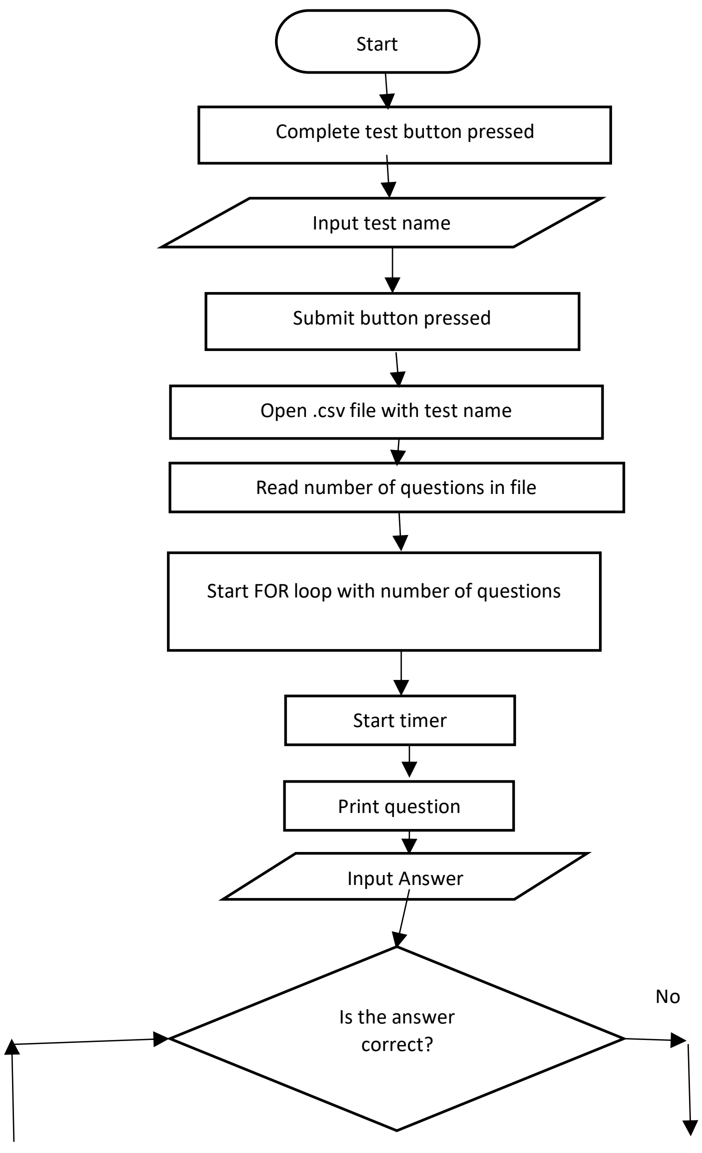


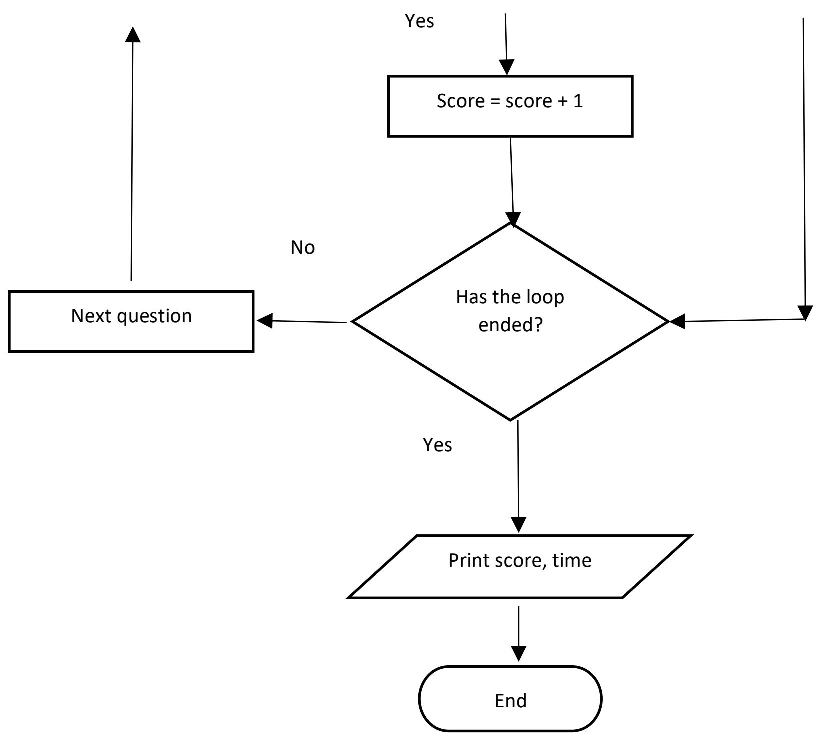
Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using.** This GUI meets the success criteria in: Each system is windowed (1), Complete a test system (19)

Each input box has a label to make it clear what is to be inputted by the user. **Otherwise, the user would not know what information to enter into the input boxes.** These input boxes meet the success criteria in: Each input is provided using a keyboard (2)

Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** These buttons meet the succes criteria in: Exit button (11)

<u>Flowchart</u>


Start
Complete test button pressed
Input test name
Submit button pressed
Open .csv file with test name
Read number of questions in file
Start FOR loop with number of questions
Start timer
Print question
Input Answer
Is the answer correct?
No



Pseudocode

```
INPUT complete button pressed
INPUT name of test
OPEN name of test excel file
START timer
READ questions
FOR questions in excel file
    PRINT question
    PRINT (“Input answer to question”)
    CHECK answer
    IF answer=inputtedAnswer
        Score=Score+1
    ENDIF
PRINT (score)
PRINT (timer)
SAVE score
```

Input/Output table

Input	Function	Output
Press complete	Starts process to complete cate a test	Starts process to complete a test
Name of test	Holds the name of the test	Allows the user to enter the name of the test that they want to complete
Test	Opens test file	Allows the user open the test they want to complete
Timer	Starts a timer	Shows long it takes a user to complete a test
Question	Holds the question	Prints the question to be answered
Answer	Holds the answer	Allows the user to enter the answer to the question
Enter	Checks if the answer is correct and prints next question	Checks if the inputted answer is correct and adds to the score appropriately, and prints next question

<u>Variables table</u>

Name	Type	Validation	Description	Justification
completeButton	String	Button must be pressed, otherwise the process does not start, allows the user to start the process once they are ready	Starts process to complete a test	Allows user to start completing a test. **Otherwise, the user would not be able to start completing a test when they are ready**
inputtedNameOfTest	String	Input is not empty, if the input does not meet the criteria an error message will appear	Name of the test file to be created	Needed to differentiate between tests. **Otherwise, the user would not be able to enter the name of a test to complete**
timer	String	No requirements, doesn't require anything from the user	Time taken to complete the test	Needed to show how long it takes a user to complete a test, also used to determine whether the user is cheating
question	String	No requirement, does not require	Question from the test file	Needed to give the user a question to

		anything from the user		answer. **Otherwise, the user would not be able to see the question to answer**
inputtedAnswer	String	Input is not empty, if the input does not meet the criteria an error message will appear	User's answer to be checked	Needed to allow the user to enter an answer and check if it is correct. **Otherwise, the user would not be able to enter their answer to the question**
enterButton	String	Button must be pressed, otherwise the process does not start, allows the user to start the process once they are ready	Button to check answer and show the next question	Needed to check if the answer is correct and print the next question. **Otherwise, the user would not be able to see if their answer is correct**

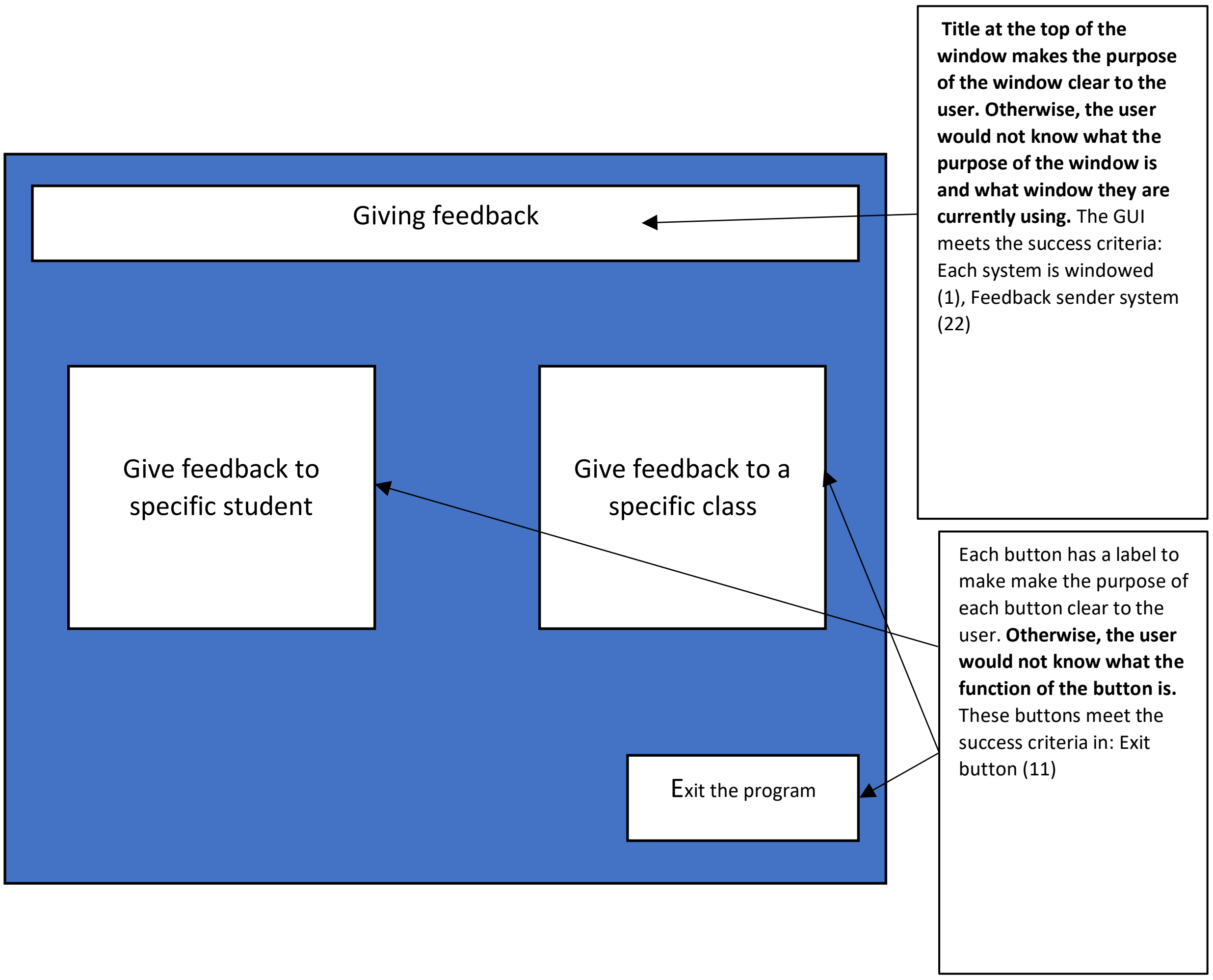
<u>Giving feedback</u>

<u>Design</u>

Purpose and justification: Allows the user to enter a user's studentID or a class code, and feedback into input boxes. This is suitable for the problem as it allows the user's inputs to be accessed by the program. **Without this, the user would not be able to enter their desired target and feedback. Also, without this the program would not be able to read the user's inputs and save them to the database.** The GUI's labels, buttons and input boxes are suitable as it gives a visual representation of the information required, **without it the user would not be able to see what information is required from them and where they should input the details of the feedback**. Furthermore, the buttons and input boxes prevent the user from interacting with the program beyond the given methods. **Without a button that executes the saving function, the user would not be able to save the feedback entered and without the input boxes, the user would not be able to enter the feedback information.**

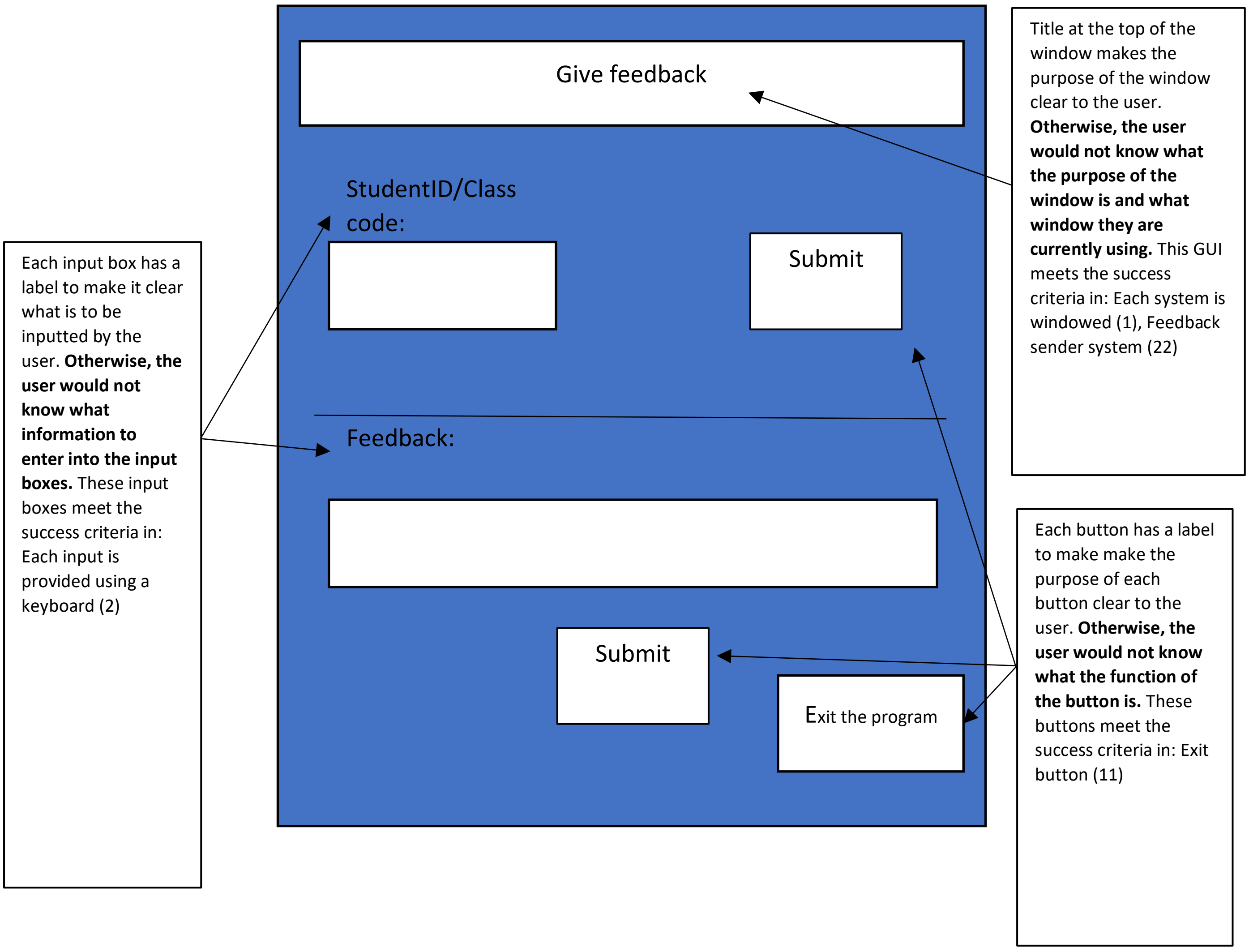
Without the diagram of the GUIs, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find

it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.


Giving feedback
Give feedback to specific student
Give feedback to a specific class
Exit the program

Title at the top of the window makes the purpose of the window clear to the user. Otherwise, the user would not know what the purpose of the window is and what window they are currently using. The GUI meets the success criteria: Each system is windowed (1), Feedback sender system (22)

Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** These buttons meet the success criteria in: Exit button (11)

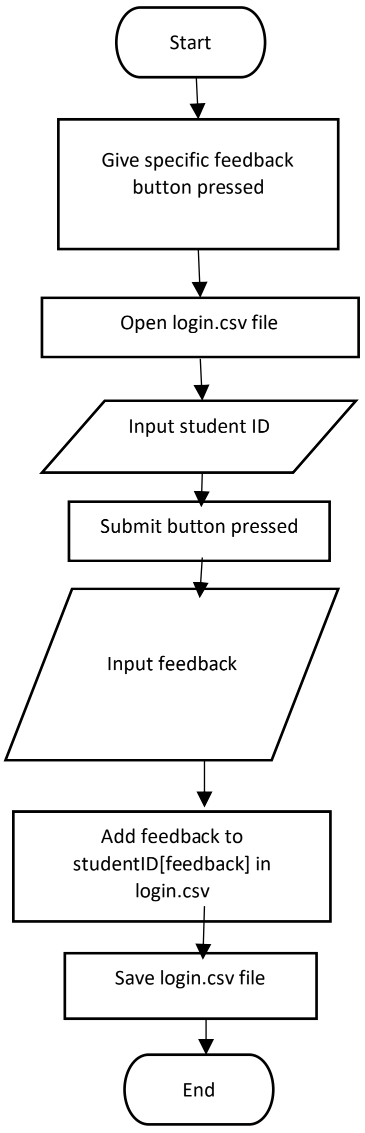


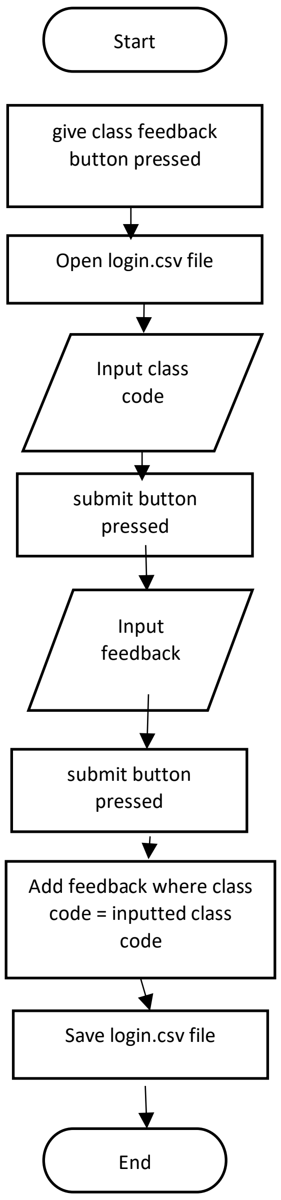
Each input box has a label to make it clear what is to be inputted by the user. **Otherwise, the user would not know what information to enter into the input boxes.** These input boxes meet the success criteria in: Each input is provided using a keyboard (2)

Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using.** This GUI meets the success criteria in: Each system is windowed (1), Feedback sender system (22)

Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** These buttons meet the success criteria in: Exit button (11)

<u>Flowchart</u>





<u>Pseudocode</u>

```
INPUT give feedback button pressed
DEFINE give feedback function
        OPEN login.csv file
        PRINT (login.csv)
        PRINT ("Would you like to give feedback to a specific user or an entire class")
        IF buttonPressed=specificUser
        PRINT("Enter the student's ID")
                FOR studentID in login.csv file
                        CHECK studentID
                IF studentID=inputtedStudentID
                    PRINT ("Input feedback")
                    studentID[feedback]=inputtedFeedback
                ELSE
                    PRINT ("Student ID not found, try again")
                    OPEN give feedback function
                ENDIF
        SAVE .csv file
    ELSE
        PRINT("Enter the class code")
        PRINT("Input feedback")
        FOR studentID in login.csv
            CHECK classCode
                IF classCode=inputtedClassCode
                    studentID[feedback]=inputtedFeedback
                ENDIF
    SAVE .csv file
    ENDIF
ENDFUNCTION give feedback function
```

```
IF admin code=NOT NULL
    OPEN give feedback function
ENDIF
```

Input/Output table

Input	Function	Output
Press give feedback	Starts process to give feedback	Starts process to give feedback
Login	Opens the login.csv file	Opens all saved login details
Student ID	Holds inputted student ID to have feedback sent to	Allows user to give feedback to a specific user
Feedback	Holds inputted feedback to be entered	Allows user to type feedback

Class code	Holds inputted class code to have feedback sent to	Allows user to give feedback to an entire class

Variables table

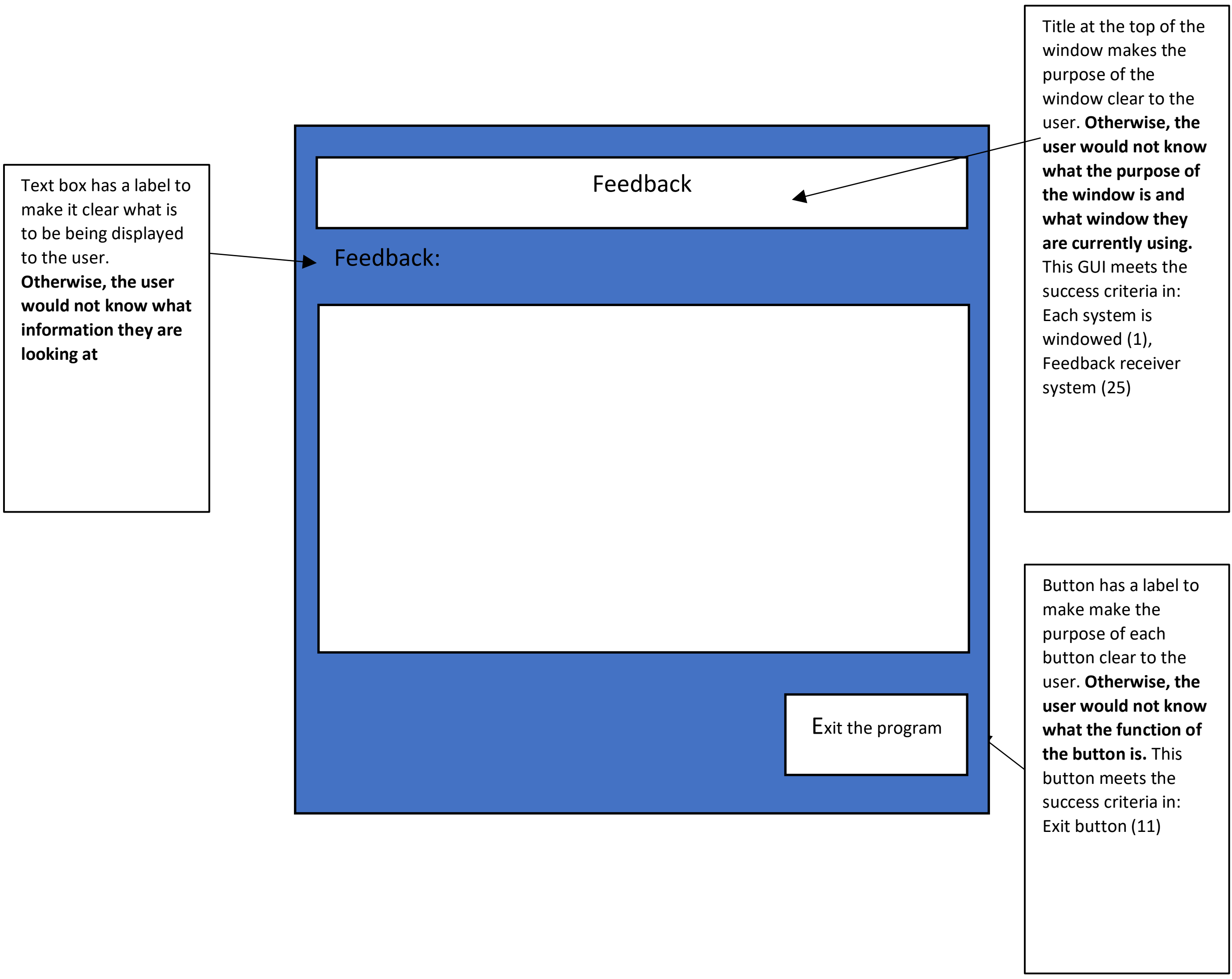
Name	Type	Validation	Description	Justification
giveFeedbackButton	String	Button must be pressed, otherwise the process does not start, allows the user to start the process once they are ready	Starts process to give feedback	Allows user to start giving feedback. **Otherwise, the user would not be able to start giving feedback when they are ready**
InputtedStudentID	String	Input is not empty, if the input does not meet the criteria an error message will appear	Student to send feedback to	Needed to send feedback to specific user. **Otherwise, the user would not be able to send feedback to a specific user**
inputtedFeedback	String	Input is not empty, if the input does not meet the criteria an error message will appear	Feedback to be saved	Needed to send and save feedback for a user. **Otherwise, the user would not be able to type feedback**
inputtedClassCode	String	Input is not empty, if the input does not meet the criteria an error message will appear	Class to send feedback to	Needed to send feedback to an entire class. **Otherwise, the user would not be able to send feedback to an entire class**

Receiving feedback

Design

Purpose and justification: Allows the user to view feedback in a text box. This is suitable for the problem as it allows the feedback to be inserted into the GUI. **Without this, the user would not be able to view their feedback**. The GUI's labels and button are suitable as it gives a visual representation of the feedback, **without it the user would not be able to see what feedback they have been given.** Furthermore, the button prevents the user from interacting with the program beyond the given method

Without the diagram of the GUI, I would find it difficult to create the window needed to fulfil its purpose. Without the pseudocode, I would find it difficult to create the functions as I would not have a visual representation of how each function interacts with each other, I would also find it difficult to prioritise which step to the solution needs to be implemented first. Without the pseudocode, I would find it difficult arranging the functions in an efficient manner and I would find it difficult to know what functions are needed. Without the input/output table and the variable table, I would find it difficult to track what variables are used in the solution to the sub-problem, what the purpose of the variable is and what conditions the variable needs to meet.

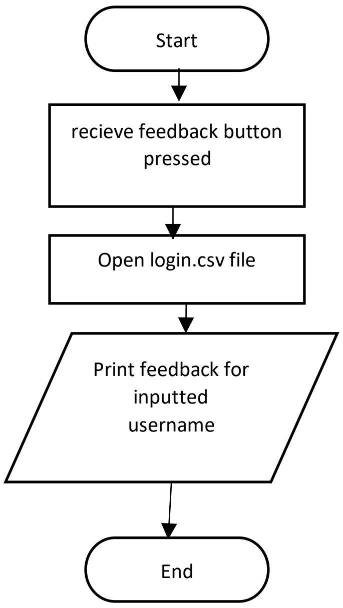


Text box has a label to make it clear what is to be being displayed to the user. **Otherwise, the user would not know what information they are looking at**

Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using.** This GUI meets the success criteria in: Each system is windowed (1), Feedback receiver system (25)

Button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** This button meets the success criteria in: Exit button (11)

<u>Flowchart</u>


Start
recieve feedback button pressed
Open login.csv file
Print feedback for inputted username
End

Pseudocode

```
INPUT receive feedback button pressed
OPEN login.csv file
FOR Username in login.csv file
        CHECK inputtedUsername
                IF inputtedUsername=Username and Username[feedback]=NOT NULL
                        PRINT(Username[feedback])
                ELSE
                PRINT("No feedback found")
```

Input/Output table

Input	Function	Output
Press receive feedback	Starts process to receive feedback	Starts process to receive feedback
Login	Opens the login.csv file	Opens all saved login details
Username	Holds inputted username (inputted at login)	Allows feedback to be checked for the specific user
Feedback	Holds feedback linked to the specific user	Allows the user to receive feedback

Variables table

Name	Type	Validation	Description	Justification
receiveFeedback Button	String	Button must be pressed, otherwise the process does not start, allows	Starts process to receive feedback	Allows user to start receiving feedback. **Otherwise, the**

		the user to start the process once they are ready		**user would not be able to start receiving feedback when they are ready**
inputtedUsername	String	Input is not empty, input does not start with a number, input is at least 8 characters long. If the input does not meet the criteria the details will not be saved, and an error message will appear	Username to receive feedback	Needed to check if the user has feedback. **Otherwise, the user would not be able to see their specific feedback**
Feedback	String	No requirements, does not require anything from the user	Feedback received by the user	Needed to print any saved feedback for the user. **Otherwise, the feedback would not appear**

<u>Describe the approach to testing</u>

<u>Client feedback</u>

Before I start developing my solution, I will conduct an interview with my client, to see if there is anything I can improve on in the design of the solution

1. What aspects of the GUIs do you like?

"All user interfaces are windowed, allowing someone to easily switch between multiple features of the program or use multiple at once. Also, every interface has a title and labels explaining what the purpose of the window is, and how a someone can interact with the window."

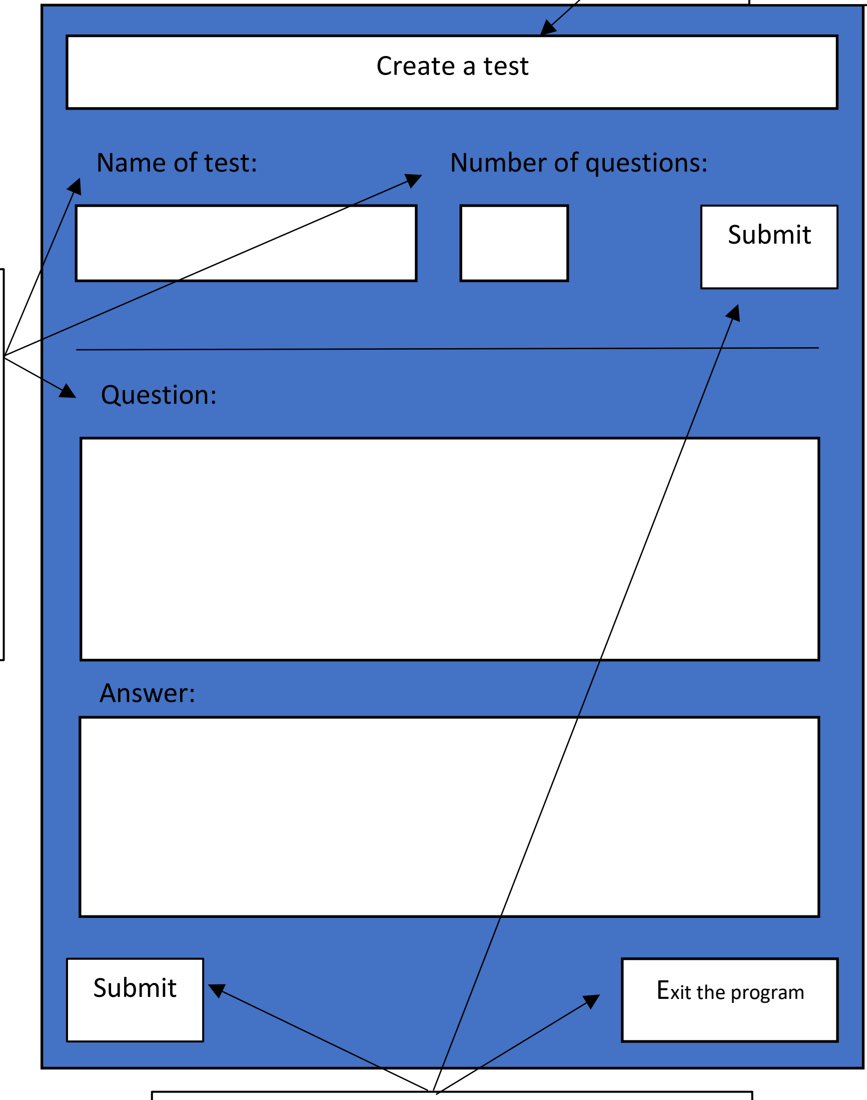
2. What aspects of the GUIs do you not like?

"The interface for the creating test feature looks too confined, I feel like it would be difficult to write questions and answers in the input boxes because of how small they are"

From this interview I can conclude that most of the designs of the GUIs are sufficient for my client's needs. However, I will modify the creating a test GUI to make it easier to use.

I do this by making the input boxes larger. **Otherwise, the user may find it difficult to type into the input boxes**

Title at the top of the window makes the purpose of the window clear to the user. **Otherwise, the user would not know what the purpose of the window is and what window they are currently using.** This GUI meets the success criteria in: Each system is windowed (1) and Create a test system (13)



Each input box has a label to make it clear what is to be inputted by the user. **Otherwise, the user would not know what information to enter into the input boxes.** These input boxes meet the success criteria in: Each input is provided using a keyboard (2)

Larger input boxes to make typing questions and answers easier. **Otherwise, the user may find it difficult to create the test**

Each button has a label to make make the purpose of each button clear to the user. **Otherwise, the user would not know what the function of the button is.** These buttons meet the success criteria in: Exit button (11)

Iterative testing

I will test my program through iterative testing. I will develop a solution to one of the decomposed problems and once this is complete, I will move onto the next decomposed problem. **Otherwise, I will find it difficult to ensure that each part of the program is working correctly. I will test all variables used in each decomposed problem to ensure that it works with no errors**. When developing the program, I will record every error received and record how I solved it with screenshots and explanations in a Word document. **Otherwise, maintenance would be difficult as I would find it hard to remember how I structured the program and developed each function, with this document post-development maintenance will be much easier**. Iterative testing and development is good for my solution as the problem has been decomposed in multiple smaller problems, most of which a solution can be developed before another feature has been introduced.

Test data for this will include: if the program runs with no errors, if the correct GUI or message box appears when a button is pressed, if the correct error message appears if input boxes have not been filled correctly, if labels, buttons and input boxes appear in the correct position on a GUI, if input boxes are accessible to the user, if buttons execute the correct function

Summative testing

I will also start testing my program towards the end of my development. This testing will be documented in my evaluation, which will be recorded in a Word document. This will help me determine the effectiveness of my program as a solution to my client's requirements and will ensure that it meets all their necessary criteria. **Otherwise, I cannot not evaluate how successful my solution is in fixing my client's problem, I also would not be able to determine whether the success criteria have been met.**

White and black box testing

I will test my program through both white and black box testing. During the development of the program and thus during the iterative testing, I will introduce white box testing to ensure that my program handles all inputs correctly. I will use the variables and inputs below to test the code I have written. **Without doing white box testing, I will not know how my program reacts to different user inputs, this testing will help improve the security of my program as well as improve the usability of my program**. During the end of the development of the program and thus during the summative testing I will introduce black box testing to ensure that my program is suitable for my client's requirements, in black box testing no knowledge of how the program works is used, this is useful as my client along with all users using the program, will have little knowledge of how the code works, therefore this reflects potential scenarios where my program may break. **Without black box testing, I will find it difficult to evaluate whether my solution has fulfilled my success criteria.**

Variable	Type of variable	Success criteria	Justification

Username	String	Holds inputted value, does not allow special characters	Username must be simple. Otherwise, it would be difficult for a user to remember their username and it would be difficult for a teacher to determine who owns the account
Password	String	Holds inputted value	Input must be assigned to the variable for comparison with login details file
Admin code	String	Holds inputted value	Input must be assigned to the variable for comparison with the correct defined code
Name of test	String	Holds inputted value, does not allow special characters	Name of test must be simple, input must be assigned to the variable to create a .csv file with the variable name
Number of questions	Integer	Holds inputted number, does not allow extreme values	Input must be a number to create a finite loop, number must not be extreme to crash the program or create a large test file
Question	String	Holds inputted value, does not allow special characters	Question must be simple, input must be assigned to the variable to add the inputted question to the test file
Answer	String	Holds inputted value, does not allow special characters	Answer must be simple, input must be assigned to the variable to add the inputted answer to the test file
Timer	String	Holds time, time is accurate	Time must be recorded accurately for evaluation
Score	Integer	Holds number, number remains an integer	Score must be a whole number for calculations
Grade	String	Holds inputted value, does not allow special characters	Grade must be simple, input must be

			assigned to the variable to add the inputted grade to the test file
Class code	String	Holds inputted value, does not allow special characters	Class code must be simple, input must be assigned to the variable to check users on the login file
Feedback	String	Holds inputted value, does not allow special characters	Feedback must be simple, input must be assigned to the variable to add to a user in the login file
Student ID	String	Holds number, number remains an integer	Score must be a whole number for calculations

Summative testing

Test Number	Test Data	Test Type	Justification	Expected Outcome
1 (Main menu testing)	Press the register an account button	Normal	Otherwise, I would not know if pressing the register an account button will open the registration window	Registration window opens
2	Press the login button	Normal	Otherwise, I would not know if pressing the login button will open the login window	Login window opens
3	Press the guide button	Normal	Otherwise, I would not know if pressing the guide button will open the guide	Guide window opens
4	Press the exit button	Normal	Otherwise, I would not know if pressing the exit button will exit the program	Program closes
5 (Registration testing)	Username: Password: Admin code:	Erroneous	Otherwise, I would not know if not inputting data in the required input boxes would create an error	Error message appears
6	Username: username1 Password: Admin code:	Boundary	Otherwise, I would not know if not inputting data in the required	Error message appears

			input boxes would create an error	
7	Username: Password: password1 Admin code:	Boundary	Otherwise, I would not know if not inputting data in the required input boxes would create an error	Error message appears
8	Username: username1 Password: password1 Admin code:	Normal	Otherwise, I would not know if correctly entering details will save the data to the database	Data is added to the database. Confirmation message appears
9	Username: Password: Admin code: testadmin	Boundary	Otherwise, I would not know if not inputting data in the required input boxes would create an error	Error message appears
10	Username: username1 Password: password1 Admin code: testadmin	Normal	Otherwise, I would not know if correctly entering details will save the data to the database	Data is added to the database. Confirmation message appears
11	Username: username1 Password: password1 Admin code: notanadmincode (admin code will be the wrong code)	Boundary	Otherwise, I would not know if a wrong admin code can still lead to an account being added correctly	Data is added to the database. Warning message appears to indicate that account is not given admin status
12	Username: 1username Password: Password Admin code:	Boundary	Otherwise, I would not know if entering a username that starts with a number will lead to an error	Error message appears
13	Username: 1username Password: 1Password Admin code:	Erroneous	Otherwise, I would not know if entering a username and password that starts with a number will lead to an error	Error message appears
14	Username: username Password: 1Password Admin code:	Boundary	Otherwise, I would not know if entering a password that starts with a number will lead to an error	Error message appears
15	Username: User Password: Password Admin code:	Boundary	Otherwise, I would not know if entering a username below 8	Error message appears

			characters will lead to an error	
16	Username: Username Password: Pass Admin code:	Boundary	Otherwise, I would not know if entering a password below 8 characters will lead to an error	Error message appears
17	Username: User Password: Pass Admin code:	Erroneous	Otherwise, I would not know if entering a username and password below 8 characters will lead to an error	Error message appears
18	Username: noadmin1 Password: noadmin2 (This will be in the database)	Normal	Otherwise, I would not know if correctly entering details will be accepted	Secondary menu (without admin features) opens
19	Username: testusername Password: (Username will be in the database)	Boundary	Otherwise, I would not know if not inputting data in the required input boxes would create an error	Error message appears
20	Username: Password: testpassword (Password will be in the database)	Boundary	Otherwise, I would not know if not inputting data in the required input boxes would create an error	Error message appears
21	Username: notausername Password: notapassword (This will not be in the database)	Erroneous	Otherwise, I would not know if inputting wrong data in both boxes would create an error	Error message appears
22	Username: Password: notapassword (Password will not be in the database)	Erroneous	Otherwise, I would not know if not inputting data in the required input boxes would create an error	Error message appears
23	Username: notausername Password: (Username will not be in the database)	Erroneous	Otherwise, I would not know if not inputting data in the required input boxes would create an error	Error message appears
24	Press the login button	Normal	Otherwise, I would not know if pressing the login button will start the login process	Login process starts

25 (Secondary menu testing)	Press the complete a test button	Normal	Otherwise, I would not know if pressing the complete a test button will open the complete a test window	Complete a test window opens
26	Press the receive feedback button	Normal	Otherwise, I would not know if pressing the receive feedback button will open the receive feedback window	Receive feedback window opens
27	Press the create a test button	Normal	Otherwise, I would not know if pressing the create a test button will open the registration window	Create a test window opens
28	Press the give feedback button	Normal	Otherwise, I would not know if pressing the give feedback button will open the give feedback window	Give feedback window opens
29 (Test creation testing) (Naming the test)	Name of test: test123	Normal	Otherwise, I would not know if entering a valid file name would create a test file	Create a test window opens
30	Name of test:	Erroneous	Otherwise, I would not know if not entering a file name would create a test file	Error message opens
31	Press submit button	Normal	Otherwise, I would not know if pressing the submit button will open the create a test window	Create a test window opens
32 (Adding to the test)	Question: question Answer: answer	Normal	Otherwise, I would not know if correctly adding a question and answer will save to the database	Data is added to the database. Confirmation message appears
33	Question: question Answer: answer1,answer2,answer3	Normal	Otherwise, I would not know if correctly adding a question and answers will save to the database	Data is added to the database. Confirmation message appears
34	Question: question Answer:	Boundary	Otherwise, I would not know if not filling all the input boxes would show an error message	Error message appears

35	Question: Answer: answer	Boundary	Otherwise, I would not know if not filling all the input boxes would show an error message	Error message appears
36	Question: Answer:	Erroneous	Otherwise, I would not know if not filling all the input boxes would show an error message	Error message appears
37	Press the submit button	Normal	Otherwise, I would not know if pressing the submit button adds the data to the database	Data is added to the database. Confirmation message appears
38	Press the preview button	Normal	Otherwise, I would not know if pressing the preview button displays a preview of the test	Preview window appears
39	Press the delete last button	Normal	Otherwise, I would not know if pressing the delete last button deletes the question and answer/s entered	Data is deleted. Confirmation message appears
40	Press the finish button	Normal	Otherwise, I would not know if pressing the finish button opens the grades window	Grade window opens
41 (Adding grades)	A:6 B:5 C:4 D:3 E:2 F:1	Normal	Otherwise, I would not know if correctly entering the grades allows them to be saved to the database	Data is added to the database
42	A:6 B:3 C:2 D: E: F:	Boundary	Otherwise, I would not know if not filling all input boxes causes an error	Error message appears
43	A: B: C: D: E: F:	Erroneous	Otherwise, I would not know if not filling all input boxes causes an error	Error message appears
44	A:3 B:5 C:4 D:3 E:2 F:1	Boundary	Otherwise, I would not know if incorrectly overlapping grade boundaries will lead to an error message	Error message appears
45	A:1 B:1	Normal	Otherwise, I would not know if grade	Grades are saved

	C:1 D:1 E:1 F:1		boundaries that are all equal will be accepted	
46	Press check total marks button (With an empty test file)	Erroneous	Otherwise, I would not know if pressing the check total marks button on an empty test displays an error message	Error message appears
47	Press submit button	Normal	Otherwise, I would not know if pressing the submit button adds the grades to the database	Data is added to the database. Confirmation message appears
48 (Test competi tion testing) (Naming the test)	Name of test: testfile (File exists)	Normal	Otherwise, I would not know if entering the name of a test file that exists will open the test completion window	Test completion window opens
49	Name of test: notatestfile (File does not exist)	Erroneous	Otherwise, I would not know if entering the name of a test file that does not exist will show an error message	Error message appears
50	Name of test:	Erroneous	Otherwise, I would not know if not entering any input will show an error message	Error message appears
51	Press submit button	Normal	Otherwise, I would not know if pressing the submit button opens the test completion window	Test completion window opens
52 (Comple ting the test)	Answer: answer1 (1st correct answer for a question)	Normal	Otherwise, I would not know if entering a correct answer would be accepted as correct and display a new question	Input accepted as a correct answer; new question displayed
53	Answer: answer2 (2nd correct answer for a question)	Normal	Otherwise, I would not know if entering an alternative correct answer would be accepted as correct and display a new question	Input accepted as a correct answer; new question displayed

54	Answer: notAnAnswer	Normal	Otherwise, I would not know if entering an incorrect answer will be accepted as incorrect and display a new question	Input accepted; new question displayed
55	Answer:	Erroneous	Otherwise, I would not know if not entering any input will show an error message	Error message appears
56 (Results window)	Press try again button	Normal	Otherwise, I would not know if pressing the try again button re-opens the test completion window	Test completion window opens
57	Press close button	Normal	Otherwise, I would not know if pressing the close button closes the results window	Results window closes
58 (Feedback sender testing) (Selecting type of feedback)	Press give feedback to a specific student button	Normal	Otherwise, I would not know if pressing the give feedback to a specific student button opens the name user window	Name user window opens
59	Press give feedback on a whole test button	Normal	Otherwise, I would not know if pressing the give feedback on a whole test button opens the name test window	Name test window opens
60 (naming user GUI)	Name of test: realTest Name of user: realUser	Normal	Otherwise, I would not know if the give feedback window opens when existing names are given	Give feedback GUI opens
61	Name of test: realTest Name of user: realUser2 (This user will exist in the login database, but will not have a recorded attempt on the test)	Erroneous	Otherwise, I would not know if an error message will appear if the invalid inputs are given	Error message appears
62	Name of test: notRealTest Name of user: realUser	Erroneous	Otherwise, I would not know if an error message will appear if	Error message appears

			the invalid inputs are given	
63	Name of test: notRealTest Name of user: notRealuser	Erroneous	Otherwise, I would not know if an error message will appear if the invalid inputs are given	Error message appears
64	Name of test: realTest Name of user:	Erroneous	Otherwise, I will not know if an error message will appear if the blank inputs are given	Error message appears
65	Name of test: notRealUser Name of user:	Erroneous	Otherwise, I will not know if an error message will appear if the blank inputs are given	Error message appears
66	Name of test: Name of user: realUser	Erroneous	Otherwise, I will not know if an error message will appear if the blank inputs are given	Error message appears
67	Name of test: Name of user: notRealUser	Erroneous	Otherwise, I will not know if an error message will appear if the blank inputs are given	Error message appears
68	Name of test: Name of user:	Erroneous	Otherwise, I will not know if an error message will appear if the blank inputs are given	Error message appears
69	Press print all test names button	Normal	Otherwise, I would not know if a list of all existing test names is printed	Window with list of test names opens
70	Press print all usernames button	Normal	Otherwise, I would not know if a list of all existing usernames is printed	Window with list of usernames opens
71	Press submit button	Normal	Otherwise, I would not know if the give feedback window will	Give feedback window opens

			open if valid inputs are given	
72 (naming test GUI)	Name of test: realTest	Normal	Otherwise, I would not know if entering an existing test name will open the give feedback window	Give feedback window opens
73	Name of test: notRealTest	Erroneous	Otherwise, I would not know if entering a non-existing test name will give an error	Error message appears
74	Name of test:	Erroneous	Otherwise, I would not know if entering a blank input will open an error message	Error message appears
75 (Giving feedbac k GUI)	Feedback: feedback	Normal	Otherwise, I would not know if a valid input would be saved	Feedback is saved
76	Feedback:	Erroneous	Otherwise, I would not know if an error message will appear if a blank input is given	Error message appears
77	Press submit button	Normal	Otherwise, I would not know if pressing the submit button with a valid input will save the feedback	Feedback is saved
78 (Receivi ng feedbac k)	Feedback receiver window	Normal	Otherwise, I would not know if the feedback receiver window correctly opens with all relevant feedback	Feedback receiver window opens and displays relevant feedback