**User Testing**

**Valid and invalid entries (user)**

Text

Description automatically generated

Input validation or user login page

Text

Description automatically generated

Empty input when typing user ID

Text

Description automatically generated

Server

User.py (client)

Invalid user login (validation for user login)

Text

Description automatically generated

Server

User.py (client)

Invalid username or email for when trying to reset password.

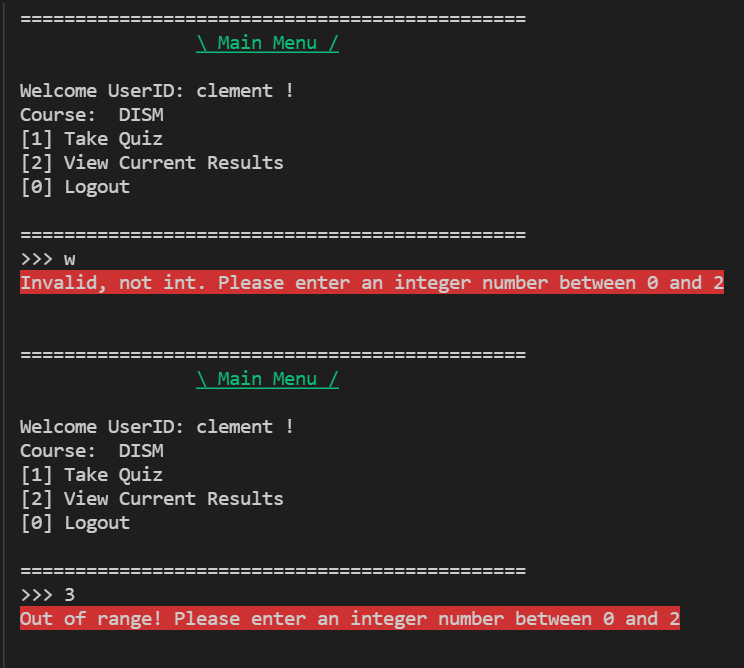
Text

Description automatically generated

User.py (client)

Server

Compliance of password when resetting password ( after username and email entered was correct, allow user to change new password)



Invalid input in main menu after logging into a user successfully. Only accept input from the range specified.

Text

Description automatically generated

Invalid input during taking of quiz. Tells user which input they accept only

Text

Description automatically generated

Invalid input when submitting the quiz (allows for jumping to question)

Text

Description automatically generatedText

Description automatically generated

Validation of input when viewing of previous results

**Question with randomized answer**

Text

Description automatically generated

**Question without randomized answer**

Text

Description automatically generated

**Valid and invalid entries (Admin)**

Text

Description automatically generated

Login page. Password is **1Qwer$#@!**

if any password is typed into it, it will be invalid and user will not be able to enter admin script

Text

Description automatically generated

Validation of input in admin Menu (only accepts number specified in range)

This is the same throughout for User Settings, setup question pool, quiz setting, edit module, edit course and generate report menus as shown below:

Text

Description automatically generated

User setting menu

Text

Description automatically generated

Setup question menu (choose topic first)

Text

Description automatically generated

Setup quiz questions (after chosen topic)

Text

Description automatically generated

Quiz Setting Menu

Text

Description automatically generated

Edit module menu

Text

Description automatically generated

Edit course menu

**Exported CSV file (quiz\_results.csv)**

Text

Description automatically generated

In CSV file in vscode

Graphical user interface, calendar

Description automatically generated

Viewed in excel with many data for each attempt made for quiz

**Documentation-User Guide (Admin)**

**Fields + Data Types for input-output files**

Input/Output File: './CA1 Assignment/Server folder/question\_pool.txt'

In txt file:

A picture containing text

Description automatically generated

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Question Title** | **Option a)** | **Option b)** | **Option c)** | **Option d)** | **Correct Answer** | **Marks** | **Topic** |
| string-question-1 | a) string-question-1-ans-a | b) string-question-1-ans-b | c) string-question-1-ans-c | d) string-question-1-ans-d | a) | 2 | String |
| **Data type** | | | | | | | |
| String | String | String | String | String | String | Integer | String |

Used for storing of questions with its options, answer, marks and topic assigned to the question. This will be used as questions for the actual quiz

Input/Output File: './CA1 Assignment/Server folder/quiz\_setting.txt'

In txt file:

Text

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Description of setting** | **Actual setting** | **Data type for actual setting** |
| Time for quiz: | 3 | Integer (string for “Unlimited value) |
| Number of questions each topic | 4 for string topic, 3 for list topic and 5 for socket topic | Integer values for each topic |
| Maximum number of attempts for quiz: | 3 | Integer (string for “Unlimited value) |
| Randomize Options for quiz | No | String |
| Quiz tested for each module | Module PSEC: Socket & List Quiz 1  Module NETF: Socket Quiz  Module ACG: Quiz 1 | String |

Using for storing of quiz settings that affects how the quiz will be run in user.py.

**Every other value below Quiz tested for each module holds the quizzes and topics for each module as shown below**

Text

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Module Name** | **Quizzes** | **Topics Assigned to quiz** |
| PSEC | Socket & List Quiz 1  String Quiz  +  String Quiz | Socket & List Quiz 1 topics: socket + list  String Quiz topics: string |
| NETF | Quiz 1  +  Socket Quiz | Quiz 1 topics: string + list  Socket Quiz topics: socket |
| ACG | Quiz 1  +  Quiz 2 | Quiz 1 topics: list + string + socket  Quiz 2 topics: string |
| **Data Type** | | |
| String | String | String |

Input/Output File: './CA1 Assignment/Server folder/userid\_pswd.txt'

In txt file:

Text

Description automatically generated

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User ID** | **Encrypted**  **Password** | **Actual Password** | **Email** | **Course** |
| Admin | pwd: wla!@#$rewQ1 | 1Qwer$#@! | **None (empty)** | **None (empty)** |
| clement | pwd: wla!@#$rewQ1 | 1Qwer$#@! | clement@gmail.com | DISM |
| johnT | pwd: wla$rewQ1 | 1Qwer$ | johnT@gmail.com | DAAA |
| johnL | pwd: wla!wW2 | 2Ww! | johnL@gmail.com | DIT |
| Leonard | pwd: wla!1loL | Lol1! | leoanrd@gmail.com | DISM |
| **Data Type** | | | | |
| String | String | String | String | String |

Using for storing of account, password, email and course assigned. This will be used for login in user.py and knowing the course and which quiz to test when taking the quiz

Input/Output File: './CA1 Assignment/Server folder/courses.txt’

In txt file:

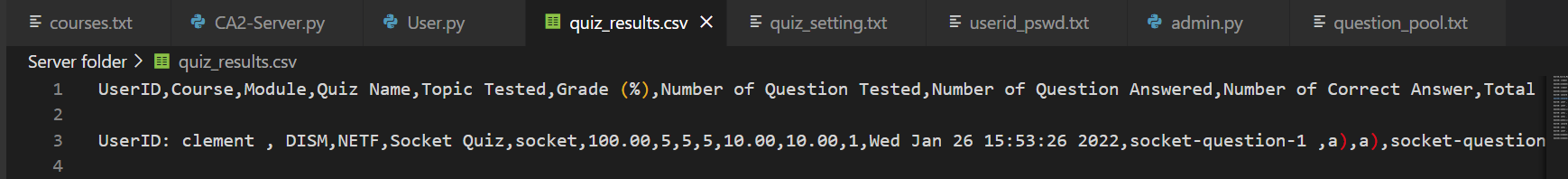
Graphical user interface, text, application

Description automatically generated

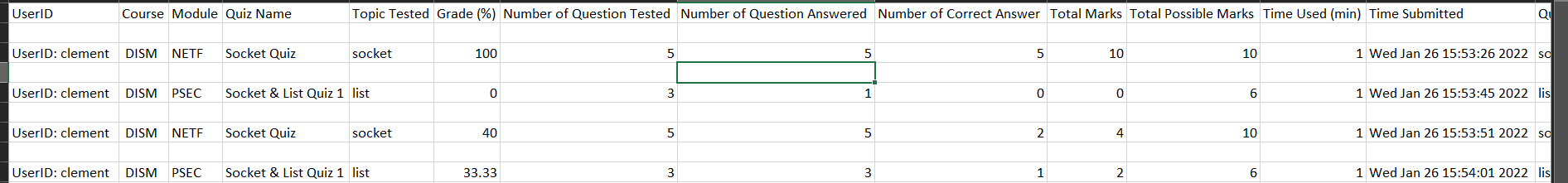
|  |  |
| --- | --- |
| **Course Name** | **Module(s) Assigned to course** |
| DISM | PSEC + NETF |
| DAAA | ACG |
| DIT | PSEC + ACG |
| **Data Types** | |
| String | String |

Used for storing of modules assigned to courses. This will help the quiz to know which modules are assigned to the user given the user’s course.

Input/Output File: './CA1 Assignment/Server folder/quiz\_results.csv'

In csv file:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CSV Headers | | | | | | | | | | | | | | | | |
| UserID | Course | Module | Quiz Name | Topic tested | Grade (%) | Number of Question Tested | Number of Question Answered | Number of Correct Answer | Total Marks | Total Possible Marks | Time Used (min) | Time Submitted | Question 1 | Q1 Correct Answer | Q1 User Answer |
| Data | | | | | | | | | | | | | | | | |
| clement | DISM | NETF | Socket Quiz | socket | 100.00 | 5 | 5 | 5 | 10.00 | 10.00 | 1 | Wed Jan 26 15:53:26 2022 | socket-question-1 | a) | a) |
| **Data Type** | | | | | | | | | | | | | | | | | |
| string | string | string | string | string | float | integer | integer | integer | float | float | integer | string | string | string | string |

Viewed in excel :

This file stores the data from a user’s attempt with relevant information about the attempt. After Time Submitted, it will continue from Question 1 , Q1 Correct Answer, Q1 User Answer …. Question 7, Q7 Correct Answer, Q7 User Answer where 7 is the total number of questions in the question pool for the given topic.

For unanswered questions, User Answer will be left as “Unanswered” while for questions that are not tested in the quiz will be left as “Not Tested”

For newly added questions, Previous attempt users will indicate “N/A” as seen below :

A screenshot of a computer

Description automatically generated with medium confidence

Cells in csv file viewed in excel

**Procedure to run scripts**

To run **admin.py**, simply press the run button or ctrl + f5

For **CA2-Server.py** in server folder, simply press the run button or

1. Type in terminal : cd '.\Server folder\'
2. Type in terminal : py .\CA2-Server.py

For **user.py** in client folder, simply press the run button or

1. Type in terminal: cd '.\Client folder\'
2. Type in terminal: py .\User.py

**Post-Implementation Support**

For support, Please contact 90715477 or [leonardsoh.20@ichat.sp.edu.sg](mailto:leonardsoh.20@ichat.sp.edu.sg)

**Documentation-User Guide (User)**

This will be a step-to-step documentation user guide to show users how to attempt and submit quiz + viewing their attempts.

1. Firstly, run the script of user.py by pressing the run button or entering ctrl + f5 on your keyboard. you will be prompted a login menu as shown below:

Text

Description automatically generated

**Please remember ensure that CA2-Server in server folder is also running**

1. In here, Please key in the numbering for what you want to do. E.g. 1 to login or 0 to quit the programme. Usually to quit or go back, press 0 and press enter to go back
2. For Users who are unsure what their passwords are, Please click number 2 and enter your userID with its appropriate email as shown below in order to reset password:

Graphical user interface, text

Description automatically generatedWe will use the account : user id : Clement, password: 1Qwer$#@!, email : [clement@gmail.com](mailto:clement@gmail.com)

User.py (client)

Server

1. After successful keying in of **correct** userID and password, You will now be able to enter a new password as shown below:

Text

Description automatically generated

Server

User.py (client)

1. With that, now you will be able to login. Press 1 and press enter at the login Menu page

Text

Description automatically generated

1. Now fill in the appropriate fields, by entering your userID and password as shown below:
2. Text

   Description automatically generated

Server

User.py (client)

1. You will now be shown the main menu where you can see your previous attempts of the quiz or take the quiz as shown below:

Text

Description automatically generated

1. You can take the quiz by pressing 1 and pressing enter on your keyboard. You will then be prompted to enter the module. After that, details of the quiz will be shown as shown below. To quit, press 0 any time. :

Text

Description automatically generated

Text

Description automatically generated

Details of the quiz is shown. To take the quiz, choose a specific topic as shown above: enter 1 for quiz testing socket topic, 2 for quiz testing list topic and 0 to go back.

1. After choosing a topic to take, now you are currently taking the quiz as shown below:

Text

Description automatically generated

1. Please enter a,b,c,d for the option you think is the correct answer. To view next question, enter n on your keyboard. To go back to previous question, press p on your keyboard. The Current Time left will also be displayed.

Text

Description automatically generated

1. After answering the last question, there will be a question summary to show all the answers you put for each question. You will then be prompted to confirm to submit the quiz or be able to go to a selected question based on the number you put.

Text

Description automatically generated

1. If you have any unanswered questions, the quiz will let you know by showing you the index of which question is left unanswered. You are able to go to those questions by keying in the index of the question.

For e.g. if you have question 1 unanswered or you want to change your answer, pressing 1 on the keyboard will allow you to go back to question 1.

After submitting the quiz by pressing 0, The results of your attempt will be shown:

Text

Description automatically generated

1. After pressing enter to continue, you will come back to main menu:

Text

Description automatically generated

1. To view your attempts, you can press 2 on your keyboard and enter.

You will be shown all your attempts as shown below:

Text

Description automatically generated

1. To view more in depth of the attempt, type in the index of the attempt. It then show all questions being tested in the chosen attempt and the answer you have answered for that question.

Graphical user interface, text

Description automatically generated

1. After displaying all questions tested, you will be given a summary of the attempt as shown below:

Text

Description automatically generated

1. With this, you have successfully attempted the quiz and viewed your results for each attempt. You can now quit the programme by pressing 0 from the Main menu to go back to the login menu :

Text

Description automatically generated

1. Finally, press 0 on your keyboard and press enter to quit.

Text

Description automatically generated

With this, we have come to the end of the user’s step by step user-guide for User.py.

**Documentation-User Guide (Admin)**

1. You will first be prompted to login into admin.

Text

Description automatically generated

Default password is **1Qwer$#@!**

1. After logging in, you will be able to do a lot of different configurations that all ultimately affect the quiz.

Text

Description automatically generated

1. With this, we have come to the end of the user’s step-by-step guide for admin.py.

**Post-Implementation Support**

For any unexpected errors/bugs that occurred that requires maintenance, Please contact 90715477 or [leonardsoh.20@ichat.sp.edu.sg](mailto:leonardsoh.20@ichat.sp.edu.sg)

**Documentation System**

**Application Overview**

Graphical user interface, diagram

Description automatically generated with medium confidence

**Diagram can be viewed here as well:**

<https://viewer.diagrams.net/?tags=%7B%7D&highlight=0000ff&edit=_blank&layers=1&nav=1&title=PSEC%20system%20overview#Uhttps%3A%2F%2Fdrive.google.com%2Fuc%3Fid%3D12HQP_bQ2akA8XIo-ILPLhxOq_TAovm9Y%26export%3Ddownload>

Server folder will hold all relevant text files that stores data needed for the quiz and CA2-Server.py will be constantly running. In the client folder, user.py will make a connection if necessary to CA2-Server.py and requests for the relevant data it needs from a specific file. CA2-Server.py will then transmit the relevant data the client needs and ends the connection upon finish transmission of data.

For admin.py, it will be isolated and just a simple run of admin.py will allow the admin user to modify and make changes to all text files if done so on the admin script.

**Programme Design**

**Pseudocode In admin:**

Programme start:

Call function mainMenu()

function mainMenu

while True

input chosenOption

if chosenOption == 0

quit programme

elif chosenOption == 1

call userSetting()

elif chosenOption == 2

call setupQuestion()

elif chosenOption == 3

call quizSetting()

elif chosenOption == 4

call moduleSetting()

elif chosenOption == 5

call courseSetting()

elif chosenOption == 6

call generateReport()

else

print error message, out of range

function userSetting()

while True

input userOption

if userOption == 0

break

goes back to mainMenu()

elif userOption == 1

print current user list from userid\_pswd with courses

elif chosenOption == 2

call registerUser()

registers a user with password + email + courses in userid\_pswd.txt

elif chosenOption == 3

call deleteUser()

deletes a user from userid\_pswd

elif chosenOption == 4

call resetPwd()

change a user password from userid\_pswd

elif chosenOption == 5

call changeCourse()

changes a user course

elif chosenoption == 6

call deleteAttempt()

delete an attempt from quiz\_results.csv

else

print error message for out of range

function setupQuestion()

while True

print(all defined topics)

input chosenOption

if chosenOption == 0

break

elif chosenOption == -1:

addQuestion()

adds a new question with a new topic

elif chosenOption >0 and chosenOption <= number of topics

select chosen topic

while True

input userOption

if userOption == 0

break

goes back to choosing topic

elif userOption == 1

call printQuestions()

print all questions in the topic in question\_pool.txt

elif chosenOption == 2

call addQuestion()

adds a new question into question\_pool.txt

elif chosenOption == 3

call deleteQuestion()

delete a selected question in question\_pool.txt

elif chosenOption == 4

call editQuestionMenu()

else

print error message for out of range

else

print error message for out of range

function editQuestionMenu()

while True

prints all question in question\_pool.txt

input chosenOption (choose a question from question pool list)

if chosenOption == 0

break

goes back to setupQuestion()

elif chosenOption == 1:

call editQuestionTitle()

edits the question title of the selected question in question\_pool.txt

elif chosenOption == 2

call editOption()

edits a selected option from a-d of the selected question in question\_pool.txt

elif chosenOption == 3

call changeMarks()

changes the mark of the selected question in question\_pool.txt

elif chosenOption == 4

call randomOption()

randomize options for selected question in question\_pool.txt

else

print error message for out of range

function quizSetting()

while True

prints all quiz settings from quiz\_setting.txt

input userOption

if userOption == 0

break

goes back to mainMenu()

elif userOption == 1

call writeSetting()

Editing time for quiz in quiz\_setting.txt

elif chosenOption == 2

call writeSetting()

Editing number of questions for each topic in quiz\_setting.txt

elif chosenOption == 3

call writeSetting()

Editing number of attempts for quiz in quiz\_setting.txt

elif chosenOption ==4

call writeSetting()

randomize options when user takes quiz

elif chosenOption == 5

call chooseQuiz()

allows user to choose a quiz to test for the selected module

else

print error message for out of range

function moduleSetting()

while True

prints all available modules

input userOption

if userOption == 0

break

elif userOption == -1

add a new module

elif userOption == -2

Delete a defined module

elif userOption > 0 and userOption <= number of modules

select chosen module

while True:

print all quizzes of chosen module

input userOption

if userOption == 0

break

elif userOption == -1

call addQuiz()

adds a quiz to the selected module

elif userOption == -2

call deleteQuiz()

deletes a defined quiz from the selected module

elif userOption > 0 and userOption <= number of quizzes in module

select chosenQuiz

call editQuizTopic(chosenQuiz)

else

print error message for out of range

else

print error message for out of range

function editQuizTopic(chosenQuiz)

while True

print current topics of chosen quiz

input userOption

if userOption == 0

break

elif userOption == 1

add a defined topic that is not already in quiz into quiz

elif userOption == 2

remove a topic from quiz

elif userOption == 3

change quiz name

else

print error message for out of range

function courseSetting()

while True

print all available courses

input userOption

if userOption == 0

break

elif userOption == -1

call addCourse()

add course

elif userOption == -2

call deleteCourse()

delete a defined course

elif userOption > 0 and userOption <= number of courses

select chosen course

while True

print all modules in course

input userOption

if userOption == 0

break

elif userOption == 1

call addModuleToCourse()

add module to course

elif userOption == 2

call deleteModuleFromCourse()

delete module form course

elif userOption == 3

call changeCourseName()

change course name

else

print error message for out of range

function generateReport()

while True

prints all module tested in results.csv

input userOption

if userOption == 0

break

elif userOption >0 and userOption <= number of modules tested

select module

print quiz tested for module

input userOption

if userOption != 0

select quiz tested

print all topics tested for quiz

input userOption

if userOption != 0

select topic

print number of questions tested

if userOption != 0

select number of questions tested

if userOption == 1

print a report based on all attempts given the module, quiz name, topic and number of questions tested

elif userOption == 2

print a report based on only highest attempt for each user given the module, quiz name, topic and number of questions tested

else

print error message for out of range

**Pseudocode in User.py**

Programme start

Call loginMenu()

Function loginMenu()

while True

input chosenOption

if chosenOption == 0

quit programme

elif chosenOption == 1

call loginUser()

returns userID (userID is none when loginUser is invalid / user quits )

if userID != none

call mainMenu(userID)

elif chosenOption == 2

call resetPwd()

resets password if user enters correct userID with email

function mainMenu(userID)

while true

input chosenOption

if chosenOption == 0

break

goes back to loginMenu()

elif chosenOption == 1

call takeQuiz()

take quiz after choosing module

elif chosenOption == 2

call viewResults()

print previous attempts made in quiz\_results.csv

function takeQuiz()

i = 0

questionTested = get list of random questions from question\_pool.txt

(number of questions based on quiz\_setting.txt)

while true

print selected question from random question list

input answer (only accept a,b,c,d,n,p)

if answer != ‘N’ or answer != ‘P’

store answer in answerList

i += 1

questionTested[i] ( goes to next question )

if i == length of questiontested list

call calculateResults() ( Submitting of attempt )

prints results stats

elif answer == ‘N’

i += 1

questionTested[i] ( goes to next question )

elif answer == ‘p’

i -= 1

questionTested[i] ( goes to previous question )

**Features + Status of completion of scripts**

**User Script**

|  |  |
| --- | --- |
| **Basic Feature** | **Status (% completed)** |
| Login | 100 |
| Compliance of password | 100 |
| Reset Password (given correct userID and email and sent to server side for validation) | 100 |
| Allow one user to connect to server to login with user id and take quiz | 100 |
| Validation of input values | 100 |
| Take quiz | 100 |
| View previous attempts of quizzes when connected to server | 100 |
| Allows simple navigation - P for previous question, N for next question | 100 |

|  |  |
| --- | --- |
| **Advanced Feature** | **Status (% completed)** |
| Allow Multiple users to connect to server and take the quiz (multi threading) | 100 |
| Background timer that exits quiz when time is over ( need user to input before quitting ) | 100 |
| Quizzes is based on course assigned to user. To take quiz, user will have to choose a module assigned to the course and then choose the topic to take quiz. (topic is assigned to quizzes) | 100 |

**Server Script**

|  |  |
| --- | --- |
| **Basic Feature** | **Status (% completed)** |
| Allows threading (client) | 100 |
| Writing of client attempt and new password(when client reset password) | 100 |

|  |  |
| --- | --- |
| **Advanced Feature** | **Status (% completed)** |
| Multithreading (client) | **100** |

**Admin Script**

|  |  |
| --- | --- |
| **Basic Feature** | **Status (% completed)** |
| Validation of input values | 100 |
| User Settings (view user list, register user, delete user, reset user password) | 100 |
| Compliance of password | 100 |
| Setup pool of quiz Questions ( each question is assigned a topic ) | 100 |
| Quit settings  (time, attempts, specify the number of questions for each topic, current quiz tested for each module) | 100 |
| Modules (add, delete module) | 100 |
| Quizzes in modules (add, delete + edit) | 100 |
| For editing quizzes in modules ( Add, Remove Topic + Change quiz name) | 100 |
| Generate Report following annex-4 (based on quiz\_results.csv) | 100 |
| Logging in into admin | 100 |

|  |  |
| --- | --- |
| **Advanced Feature** | **Status (% completed)** |
| Allows Admin to specify attempts for the quiz, eg, single only, multiple, or unlimited attempts | 100 |
| Allows admin to randomize options during quiz | 100 |
| Add, delete and edit courses | 100 |
| In edit course, add module(s) to course or delete module from course or change course name. (each user is assigned a course) | 100 |
| Choosing of a specific quiz in a selected module to test | 100 |
| Delete User Attempt ( from quiz\_results.csv) | 100 |
| Change User course | 100 |
| Setting of time in quiz (min) or unlimited | 100 |
| Display all grades or highest grades for users in generate report | 100 |