System Testing

System Testing

Testing – Any text highlighted in red in the table below is an example of the data used.

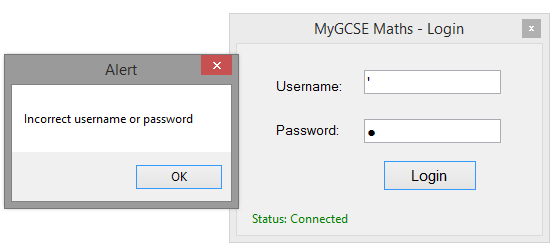
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test** | **Description** | **Data type** | **Expected Result** | **Actual Result** | **Reference** |
| 1 | Attempting to login into the system by trying to bypass the login screen using an SQL Injection technique. For example, entering a single quotation mark is generally a good way of checking for SQL vulnerability. | Erroneous  1’ OR 1 = 1 –  and  ‘ | I’ve used parameterized queries, so the system execute the login credentials literally. It should say “incorrect username or password”. | Pass | Screenshot 1.0 & 1.1 |
| 2 | Able to login to the system as an Administrator. | Typical  admin  12345 | The system should direct the user to the correct form, which is the admins form. | Pass | Screenshot 2.0, 2.1, 2.2 |
| 3 | The welcome label should show a welcome message to the admin. | Typical | The label to display their Firstname and Lastname along with a welcome message | Pass | Screenshot 3.0 & 3.1 |
| 4 | Adding a new user of the rank admin to the system. The system should capture the UserID, Firstname, Lastname and Rank entered. | Typical  1  Jason  Mark  Admin | Message box saying user added successfully and it should add the user to the SQL database. It should also generate their username, password and email. | Pass | Screenshot 4.0 & 4.1 |
| 5 | Adding a new user with an invalid userId containing a mix of strings and integers | Erroneous  48!7d7d2 | The program will say that the UserID entered is invalid and won’t add the user | Pass | Screenshot 5.0 |
| 6 | Adding a new user to the system and for their Firstname and Lastname, I will add non alphabetic characters like &^%$£ | Erroneous  “M\*^%£  ark”  “Tho\*^ma  s-Lukasz  “£$s%” | I expect the system to remove every non-alphabetic characters (excluding – for double barrelled names) in the given string. If the input is  Sus&^%a”$n , it should return Susan | Pass | Screenshot 6.0 & 6.1 |
| 7 | Adding a new user with a very long Firstname and Lastname | Erroneous | Should show an error message in the text box | Pass | Screenshot 7.0 |
| 8 | Adding a new user with a pre-existing UserID | Erroneous  4566 | It should say UserID already exists and will not add the user | Pass | Screenshot 8.0 & 8.1 |
| 9 | Adding a new class into the system | Typical  B  213 | A message box that will say Successful | Pass | Screenshot 9.0 |
| 10 | Adding a new class (block, room), but for the room number add a character longer than 5 characters | Erroneous  F  sdfgdfgdfg | A JIT error message should be displayed since I didn’t Programmatically catch the exception. | Pass | Screenshot 10.0 |
| 11 | Adding a class block with data longer than 1 character | Erroneous  FFF | The text box only takes 1 character. So it won’t even let me enter more than 1. | Pass | Screenshot 11.0 |
| 12 | Using the search button to see all the classes in the system | Typical | The system should show the information of all the classes in the database, along with the teachers assigned to the classes. | Pass | Screenshot 12.0 & 12.1 |
| 13 | Adding a new user of the rank teacher, and assigning that teacher to an already existing class. | Typical  8655  Rachel  Varghese  Teacher  Block B | A message box that will say Successful | Pass | Screenshot 13.0 & 13.1 |
| 14 | The system should generate the email & username of the teacher accurately in line with the schema used by the college. | Typical | The system should generate the username and email of the teacher by mimicking the college’s schema for generating usernames and emails. | Pass | Screenshot 14.0 & 14.1 |
| 15 | Viewing all the classes in the system | Typical | Show all classes in the system along with assigned teachers | Pass | Screenshot 15.0 |
| 16 | Using the search feature to search for all the users in the system | Typical | A list of all the users found from the SQL database. | Pass | Screenshot 16.0 & 16.1 |
| 17 | Using the search feature to search for a user that doesn’t exists. | Typical  matthew | A message box saying the user wasn’t found | Pass | Screenshot 17.0 |
| 18 | Using the search feature to search for all users with a “an” somewhere in their name | Typical  an | A list of all the users found | Pass | Screenshot 18.0 & 18.1 |
| 19 | Update the email, password and class of a user. | Typical | A message box that will say Successful | Pass | Screenshot 19.0, 19.1, 19.2, 19.3 |
| 19A | Update the user email address with an invalid email address. | Erroneous  hellothere  @hotmail. | The regular expression used to validate the email address will not match the email entered and thus show an error message saying that the email entered was invalid | Pass | Screenshot 19A.0  This screenshot can be found on page 179. |
| 20 | Deleting a user from the system | Typical | A message box that will say Successful | Pass | Screenshot 20.0 & 20.1 |
| 21 | Adding a new user of the rank student to the system, and assign the student to an existing class. | Typical  4436  Uchenna  Okafor  Block B | A message box that will say Successful | Pass | Screenshot 21.0 |
| 22 | The system should generate the students email and username accurately, in line with the colleges schema for generating student  credentials | Typical | It should generate the email and username, and mimic the SFX college email and password scheme | Pass | Screenshot 22.0 & 22.1 |
| 23 | Admin logs out of the system | Typical | Takes the user back to the login screen | Pass | Screenshot 23.0 & 23.1 |
| 24 | Able to login to the system as a Teacher | Typical  rvarghese  dhgjnx21q | The system should re-direct the user to the teacher form. | Pass | Screenshot 24.0 & 24.1 |
| 25 | The welcome label should welcome the teacher logged in and show the information of the class they teach | Typical | Shows teachers first name & last name along with the class they’re currently teaching. | Pass | Screenshot 25.0 & 25.1 |
| 26 | Teacher searching for the students in their class. | Typical | I’ve implemented an algorithm that will dynamically display each student found onto the teacher form in a grid like effect consisting of picture boxes. For each found student I expect to see a picture box and label representation of the student. | Pass | Screenshot 26.0 & 26.1 |
| 27 | The students count label to show the number of students found after the search | Typical | An Integer of the numbers of students found. | Pass | Screenshot 27.0 |
| 28 | Teacher searching for all students whose name contain “o” | Typical  o | A display of all the students in the teachers class whose name  contain the letter “o” | Pass | Screenshot 28.0 & 28.1 |
| 29 | Teacher searches for a student that doesn’t exist in their class | Typical  mary | A message box saying that no student was found | Pass | Screenshot 29.0 |
| 30 | Attempt an SQL injection when searching for a user. | Erroneous  'Drop Table tblUsers --  AND ‘ | Since all my SQL queries are parameterized, the system should treat the malicious SQL code literally, and should say the student was not found. | Pass | Screenshot 30.0 & 30.1 |
| 31 | The teacher exporting the MyGCSE Maths credentials(username and password) for each student in their class | Typical | The system should export the name, username and password of each student in the teacher’s class in the format  Name: [???]  Username: [???]  Password: [???]  into a txt file and to the chosen file path | Pass | Screenshot 31.0 & 31.1 |
| 32 | Test if teacher can use the question generator form to generate 10 multi choice algebra questions of grade difficulty B. Then check if it added to the SQL table successfully. | Typical  Algebra  B  True  10 | A message box saying questions added successfully | Pass | Screenshot 32.0 & 32.1 |
| 33 | Test if teacher can use the question generator form to generate 10 multi choice shapes questions of grade difficulty C. | Typical  Shapes  C  True  10 | A message box saying questions added successfully | Pass | Screenshot 33.0 & 33.1 |
| 33A | After generating shapes questions, check the shape was and its labels were accurately drawn. Also, check the answer is calculated accurately. | Typical | The shape should be drawn, labelled and generated accurately. The answer to the question should be correct also. | Pass | Screenshot 33A.0 & 33A.1 |
| 34 | Test if teacher can use the question generator form to generate 6 non multiple choice handling data questions of grade difficulty A. Then check if it added to the SQL table successfully. | Typical  Handling  Data  A  False  6 | A message box saying questions added successfully | Pass | Screenshot 34.0 & 34.1 |
| 35 | The teacher adding a single choice question manually | Typical | Message box saying action was successful | Pass | Screenshot 35.0 |
| 36 | Teacher adding a multiple choice question manually | Typical | Question added successfully | Pass | Screenshot 36.0 |
| 37 | Teacher adding an answer that is longer than 100 characters to a question. | Erroneous | A JIT error message should be displayed since I didn’t Programmatically catch the exception. | Pass | Screenshot 37.0 |
| 38 | Teacher searching for all the questions in the system | Typical | All the questions in the system should show in a tree view under a topic node | Pass | Screenshot 38.0 |
| 39 | Teacher searching for all the shapes questions | Typical | Only shapes questions should show under the shape node | Pass | Screenshot 39.0 |
| 40 | Teacher searching the system for all the B grade questions | Typical | System shows all the B grade questions | Pass | Screenshot 40.0 & 40.1 |
| 41 | Teacher searching the system for all the B grade algebra questions | Typical | Should show only B grade algebra questions. | Pass | Screenshot 41.0 |
| 42 | Teacher edits the details of a pre-existing question | Typical | Should say edit successful | Pass | Screenshot 42.0, 42.1 & 42.2 |
| 43 | Teacher deletes a mass amount of pre-exiting questions | Typical | Should say deleted x amount of questions successfully | Pass | Screenshot 43.0 |
| 44 | Teacher attempts to mass delete questions without selecting any questions | Erroneous | Should show an error message saying they haven’t selected any items to delete | Pass | Screenshot 44.0 |
| 45 | Teacher to add a set of questions to a quiz and name the quiz “AQA Maths Quiz 5” | Typical | A message box saying it was successful | Pass | Screenshot 45.0 & 45.1 |
| 46 | Teacher to view all quizzes in the system | Typical | The system should show every question under each quiz. | Pass | Screenshot 46.0 |
| 47 | Teacher to export a mark scheme of a quiz | Typical | It should export a txt file in the format  Question: [???]  Answer: [???]  In the chosen save location. | Pass | Screenshot 47.0 & 47.1 |
| 48 | Teacher logs out of the system using the logout button | Typical | Logs out the system and returns back to the login form | Pass | Screenshot 48.0 & 48.1 |
| 49 | Able to login as a student | Typical  Okaforu  4436  wxrslmLM | Should direct the user to the student form | Pass | Screenshot 49.0 & 49.1 |
| 50 | Student attempts an uncompleted quiz that has been set by teacher | Typical | The system will prompt user if they want to start the quiz before beginning | Pass | Screenshot 50.0 & 50.1 |
| 51 | The system to display a single answer question to a student | Typical | It will not show any radio buttons, only provides the user with a text box to enter the answer on. | Pass | Screenshot 51.0 |
| 52 | The system should display a multiple choice question to a student | Typical | Each false answer and question should be displayed using radio buttons in a randomized order | Pass | Screenshot 52.0 |
| 53 | Student answers a multiple choice question correctly | Typical  9cm^2 | Message box saying answer correct | Pass | Screenshot 53.0 |
| 54 | Student answers a single answer question correctly | Typical  64m | Message box saying answer correct | Pass | Screenshot 54.0 |
| 55 | Answer a question incorrectly | Typical | Message box saying answer was incorrect | Pass | Screenshot 55.0 |
| 56 | Enter a long string as the answer of a question | Erroneous | The answer should be treated literally and should display wrong answer message box | Pass | Screenshot 56.0 |
| 57 | When a quiz is completed the system should save the score and stores it in the SQL database | Typical | A message box showing the user their score, and saying the quiz result was saved successfully | Pass | Screenshot 57.0, 57.1 & 57.2 |
| 58 | The student to have no outstanding quizzes | Typical | The system should say that the student has no outstanding quizzes, and will therefore allow them to click a link to start a set of random questions. | Pass | Screenshot 58.0 |
| 59 | Start a set of random questions – this is only possible if the student has no outstanding quizzes | Typical | The system should prompt the user before loading up questions | Pass | Screenshot 59.0 & 59.1 |
| 60 | Student completing all the randomly chosen questions. | Typical | The system should show the students’ performance to the system but should not save the result. | Pass | Null – The result is the same as screenshot 57.0 |
| 61 | While there are no questions in the database, the student should try and complete a set of random questions. | Typical | An error message should show saying that there are no questions available in the system |  | Screenshot 60.0, 60.1 & 61.2 |
| 62 | System able to log the student out of the system | Typical | The system should log the student out and take them back to the login screen | Pass | Screenshot 62.0 |
| 63 | Login as a teacher search for a student and view their progress | Typical | Should show the students’ progress in a graph | Pass | Screenshot 63.0 & 63.1 |
| 64 | Select the minimum date for when viewing student’s results. The acceptable date range is 01/10/2014 till the current date.    For example, 01/10/2014 – 24/03/2015 | Extreme | The system will show all quiz results between 01/01/2014 and the current date. | Pass | Screenshot 64.0 |
| 65 | Selecting a time frame below 01/01/2014 | Erroneous | A message box should pop up saying that it can’t access data before 01/01/2014 | Pass | Screenshot 65.0 |
| 66 | Export a summary of the overall quiz performance for all students in the teacher’s class. | Typical | Should export the summary in a txt file in the format  [Title]  Name: [???]  Average: [???]  Total quizzes completed: [???] | Pass | Screenshot 66.0 & 66.1 |

Integration Testing

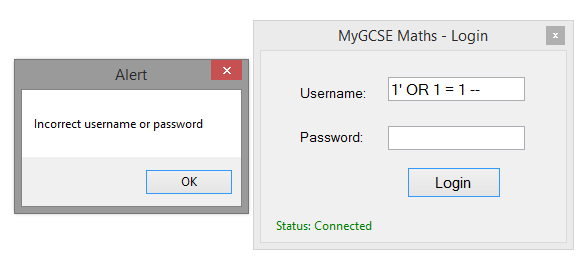
|  |  |  |
| --- | --- | --- |
|  | **Description** | **Evidence** |
| 1 | The admin logs into the system and creates a classroom called “Block B” room 213. | Screenshot 2.0, 2.1, 2.2  Screenshot 3.0 & 3.1  Screenshot 9.0 |
| 2 | The admin then creates a new user (teacher) “Rachel Varghese” and assigns her to Block B. This means she now teaches all the students that have been also been assigned to Block B. | Screenshot 13.0 & 13.1 |
| 3 | The admin then creates a new user (student) called “Uchenna Okafor” and assigns him to Block B. This means Uchenna profile is visible to Rachel Varghese and all other teachers that have been assigned to teach in Block B. | Screenshot 21.0 & 22.0 |
| 4 | The admin then logs out of the system, and the user Rachel Varghese logs in. | Screenshot 23.0 & 23.1  Screenshot 24.0 & 24.1 |
| 5 | Rachel then uses the question generator feature to generate questions and manually add questions to the system.  Rachel then sets a new quiz called “AQA Maths Quiz 5” which consists of all the questions she had just generated for all the students in here class to complete. | Screenshot 32.0 & 32.1  Screenshot 33.0 & 33.1  Screenshot 33A.0 & 33A.1  Screenshot 34.0 & 34.1  Screenshot 35.0 & 36.0  Screenshot 45.0 & 45.1  Screenshot 46.0 |
| 6 | Rachel then logs out of the system | Screenshot 48.0 & 48.1 |
| 7 | Uchenna then logs into the system and is presented with the “AQA Maths Quiz 5” quiz which he hasn’t yet completed. | Screenshot 49.0 & 49.1  Screenshot 50.0 |
| 8 | Uchenna then starts the quiz, and answers the questions set by his teacher. He answers some questions right and some wrong. | Screenshot 50.1 & 51.0  Screenshot 52.0 & 53.0  Screenshot 54.0 & 55.0  Screenshot 56.0 |
| 9 | Uchenna’s then finished answering all the allocated questions, the system marks his work, shows him the results, which is then saved in the SQL database, meaning only his teachers can see his performance.  Uchenna is now presented with a screen that notifies him that he has completed all outstanding quizzes. | Screenshot 57.0, 57.1 & 57.2  Screenshot 58.0 |
| 10 | Rachel then logs back into the system, searches for Uchenna’s name, clicks his picture and is then able to view his quiz performance for “AQA Maths Quiz 5”.  The teacher is also able to export an overview of the performance of every student in her class to a text file onto her computer. | Screenshot 63.0 & 63.1  Screenshot 66.0 & 66.1 |

This integrated test shows that each module of my system works well when used together.

Testing Cross Reference



Screenshot 1.0

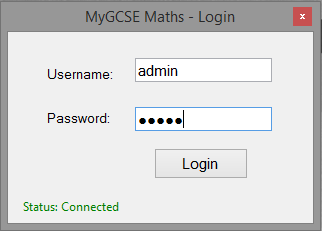


Screenshot 1.1

Shows that the system is not vulnerable against malicious SQL Injection attacks.



Screenshot 2.0



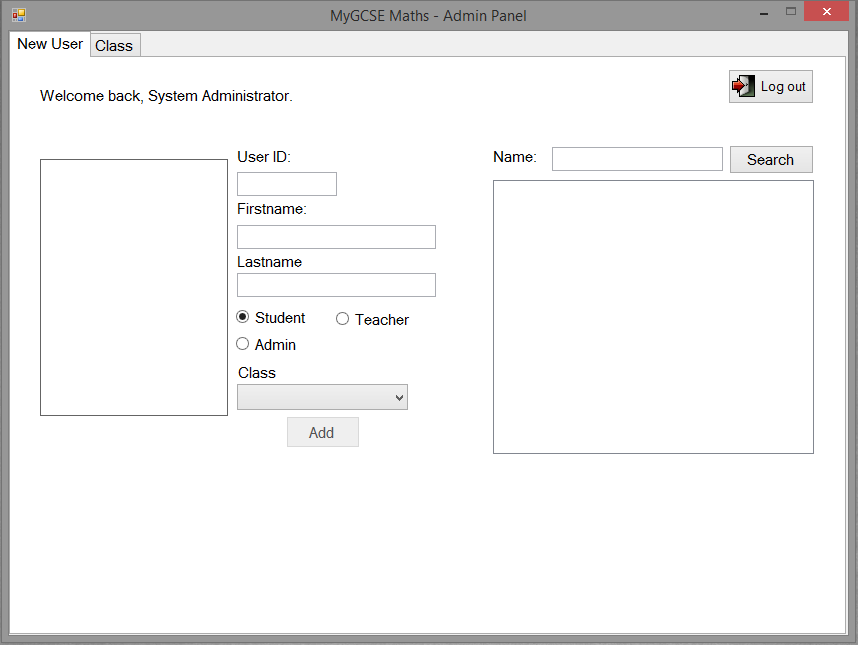
When logging in, the system will check if the username and password entered exists in the system. If they exist (see screenshot 2.0), it will then assess the users rank and then display the correct form to the user depending on the value of their rank.

0 = Student

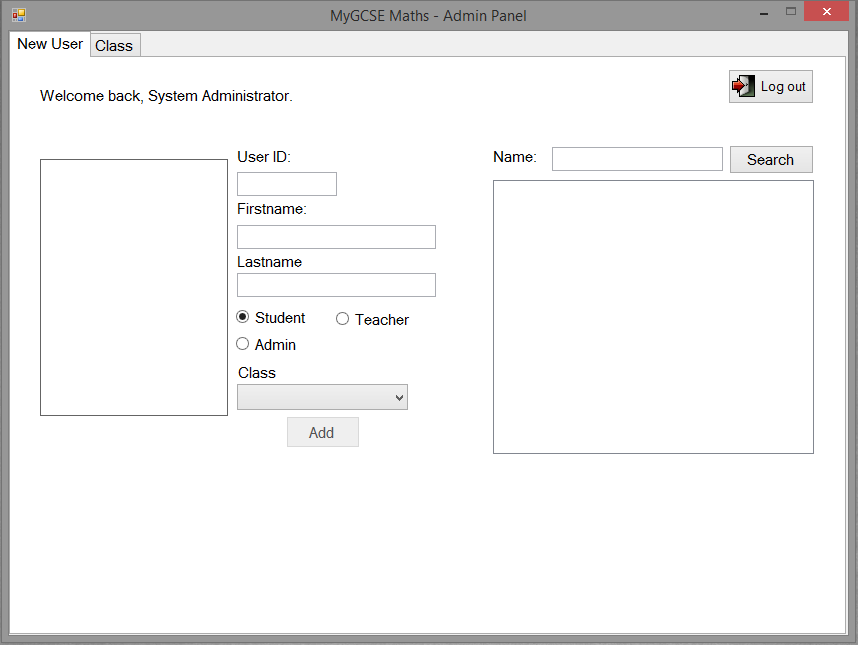
1 = Teacher

2 = Admin

Screenshot 2.1



Screenshot 2.2



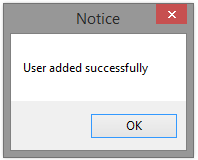
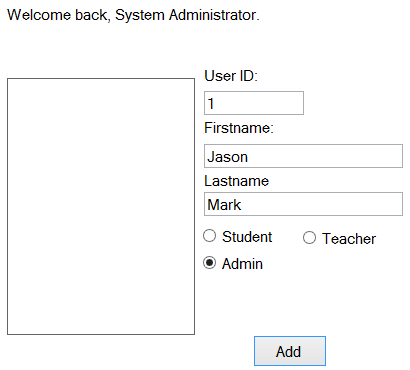
Screenshot 3.0



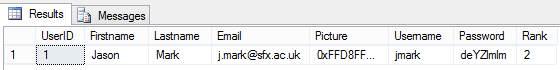
Screenshot 3.1

The welcome label shows the users first name and last name. This is as it’s supposed to be.

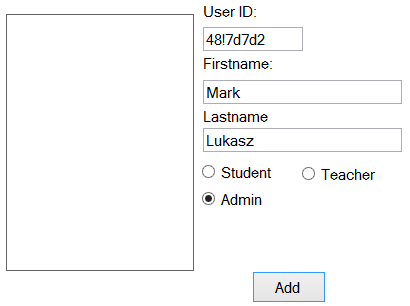
Screenshot 4.0



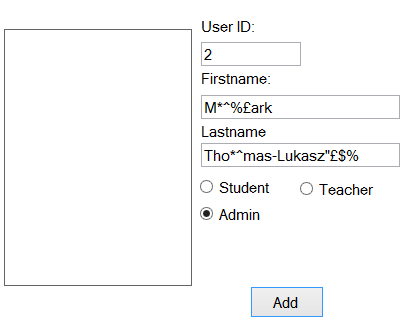
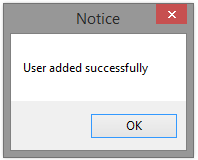
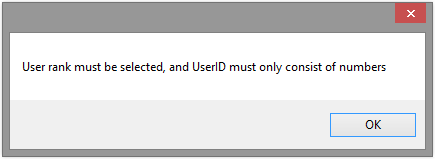
The user has been added successfully to the user table in the SQL database. The system then generates their username and email based on their Firstname, Lastname and rank. The system also generates their password to be a random set of characters, and gives the user a default image if none was selected. This shows that adding users in my system works perfectly.



Screenshot 4.1

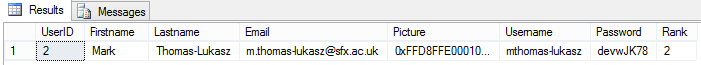


Screenshot 5.0

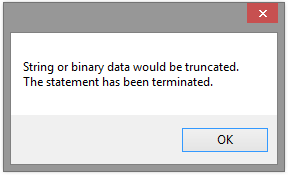
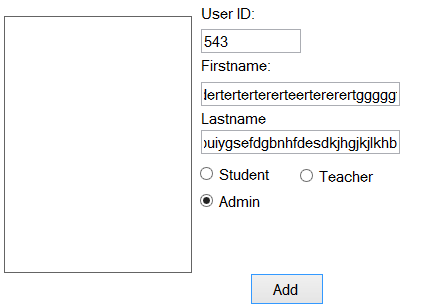


Screenshot 6.0

The system will use the ParseStr() Function to remove every occurrence of a non-alphabetic character and add the new string to the system. The system automatically removes the non-valid characters from the First name and Last name textboxes.

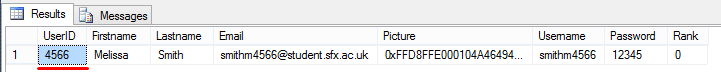


Screenshot 6.1

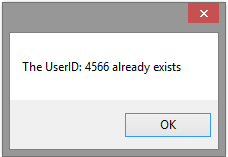
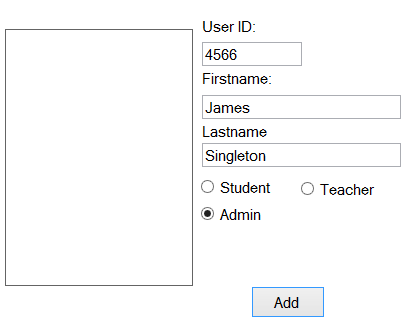


Screenshot 7.0

Pre-existing user



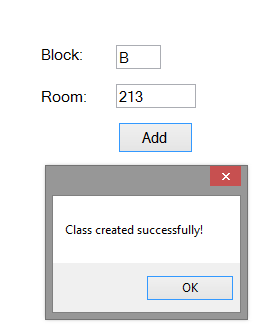
Screenshot 8.0



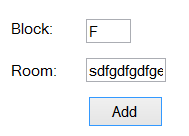
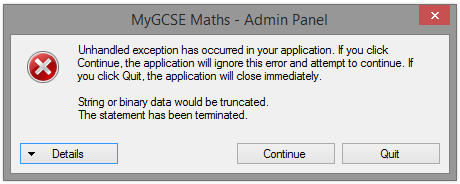
Screenshot 8.1

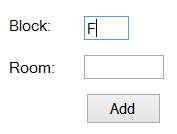
Screenshot 8.0 shows that there is already a user in the SQL table with the UserID 4566. Therefore, trying to add a new user to the system with that UserID it will not work because UserID field is a primary key and therefore cannot be duplicated.

Screenshot 9.0



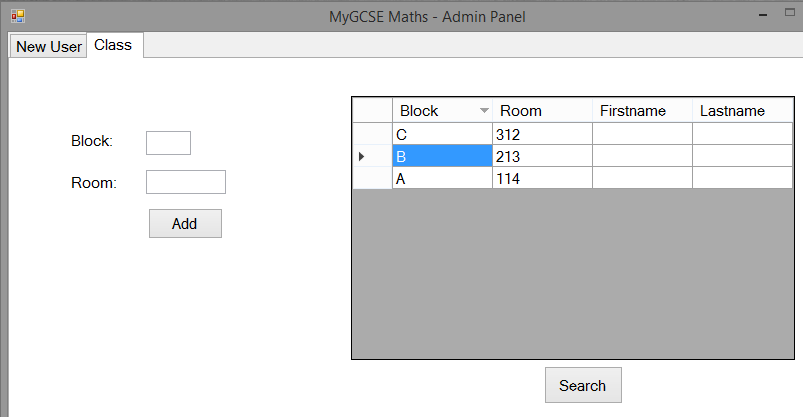
Screenshot 10.0





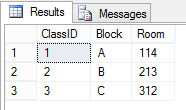
Doesn’t let me add anymore since the max capacity is 1 character.

Screenshot 11.0

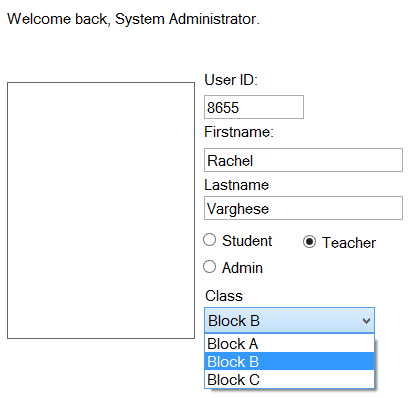


The data grid view shows all the classes in the system. It shows the class that was previously added in screenshot 9.0. It also shows all the other classes I have added. Since no teachers have been assigned to a class yet, it won’t show anything for the First name and last name columns.

Screenshot 12.0

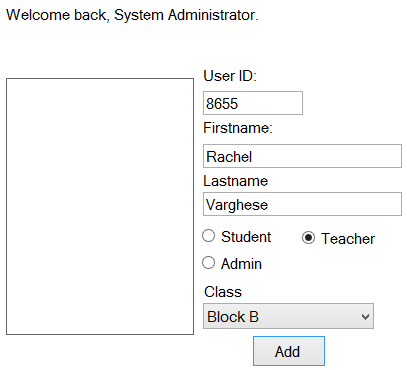
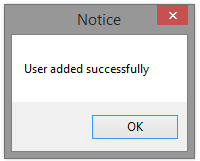


Screenshot 12.1



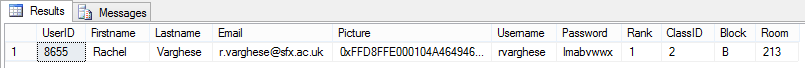
The combo box will only show a list of all the classes that exist in the system/database. Screenshot 12.0 and 12.1 shows all the classes from the SQL database.

Screenshot 13.0



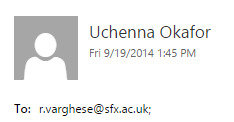
This means that the teacher has been successfully added to the system and has now been assigned to teach all the students in Block B.

Screenshot 13.1



Screenshot 14.0

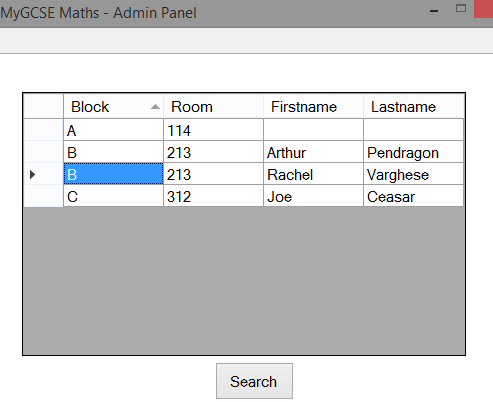
The image in screenshot 14.0 shows the data created when a new user is added into my system using the admin panel. I added the user Rachel Varghese as a teacher in screenshot 13.1. The system generated the email and username based upon the Firstname, Lastname, and User rank. The image in screenshot 14.1, shows an email I sent to Rachel Varghese in the past. The email address is her college email, and it’s the same as the email my system generated. This is proof that my algorithm can generate emails according to the schema used by my college. Also, the password was generated was just randomized. The username generated is also accurate. This is the username the teacher’s would use to login to the college systems. The use of generating user’s credentials, is that it makes it easier for the admin to add new accounts. Less input is required, and is easier for the user to remember their credentials since it’s identical to their college credentials.

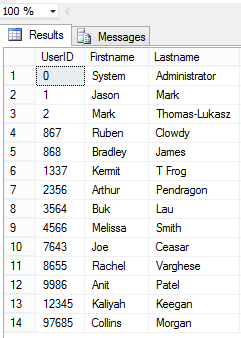
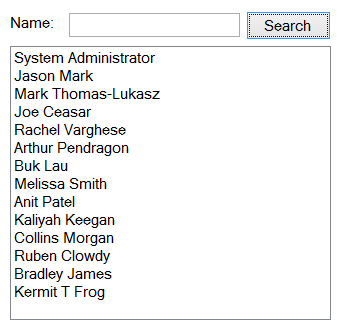


Screenshot 14.1

This diagram shows all the classes and the teachers assigned to them. Two teachers are assigned to Block B, and one teacher to Block C. No teachers are currently assigned to Block A. This shows that my database relationship works properly. This allows multiple teachers to be assigned to one class, and one teacher to be assigned to multiple classes.

Screenshot 15.0

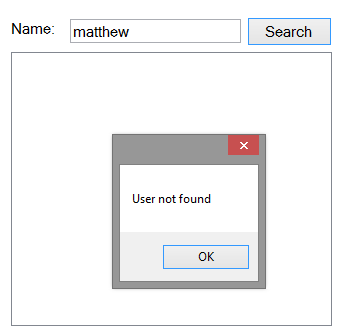




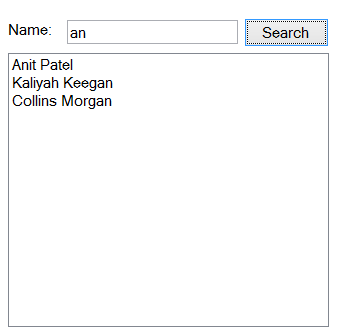
Screenshot 16.1

All the users in the SQL database.

Screenshot 16.0

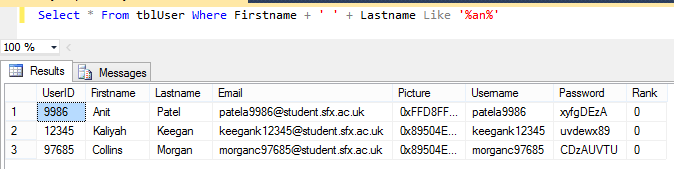


Screenshot 17.0

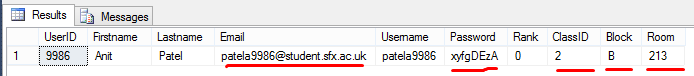


Only the users in the SQL table who have “an” in their name will be shown in the listbox. See screenshot 18.0 and 18.1

Screenshot 18.0



Screenshot 18.1



Before

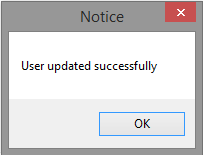
Before

Screenshot 19.0



After

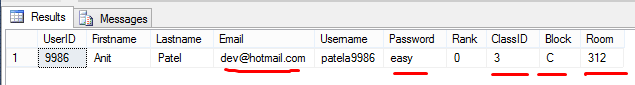
Screenshot 19.1



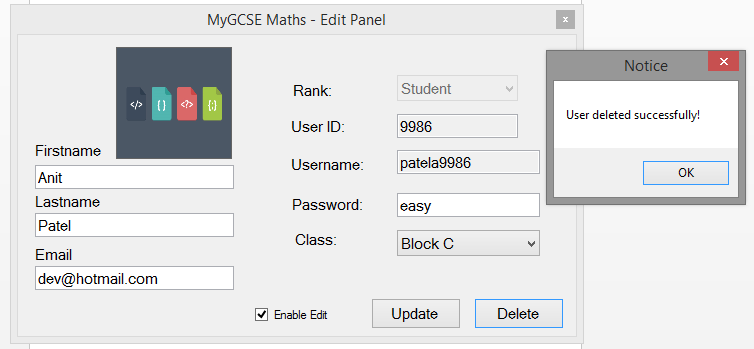
Screenshot 19.0 and 19.1 shows the data of the user in the SQL table before it was edited via my system. Screenshot 19.2 and 19.3 shows the data after it has been edited. This proves my SQL update statement works very well.

Screenshot 19.2

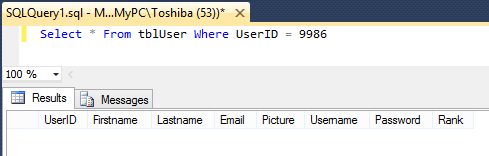
After



Screenshot 19.3

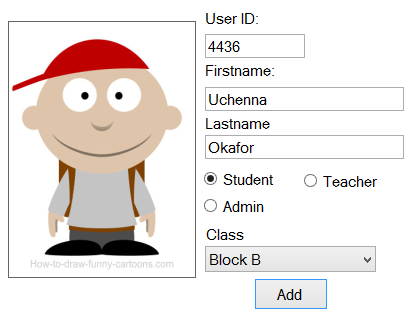


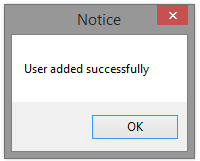
Screenshot 20.0



This proves the SQL delete statement works because when searched for, the user cannot be found.

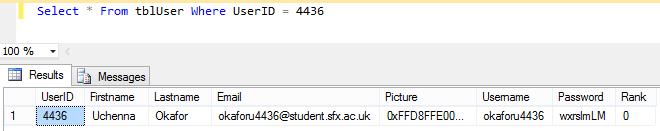
Screenshot 20.1



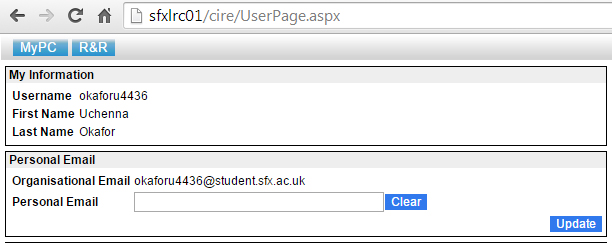


Screenshot 21.0

This also shows importing images to user accounts works.

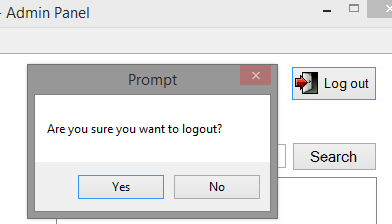


Screenshot 22.0

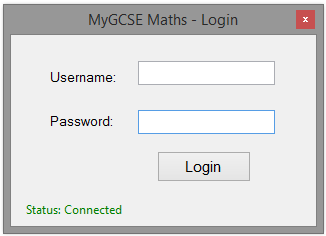


Screenshot 22.1 shows my personal college details, this means my system generates student credentials accurate to the college’s schema. This means when logging into my system, it will be easier to remember my credentials.

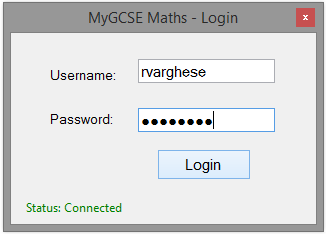
Screenshot 22.1



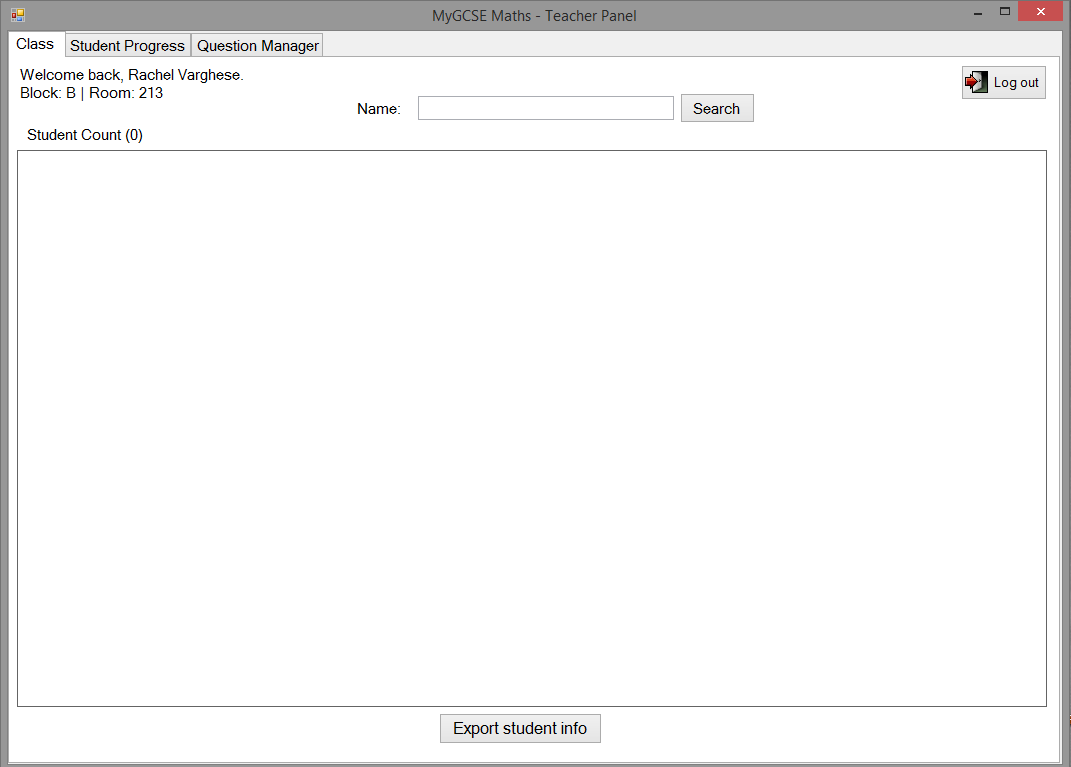
Screenshot 23.0



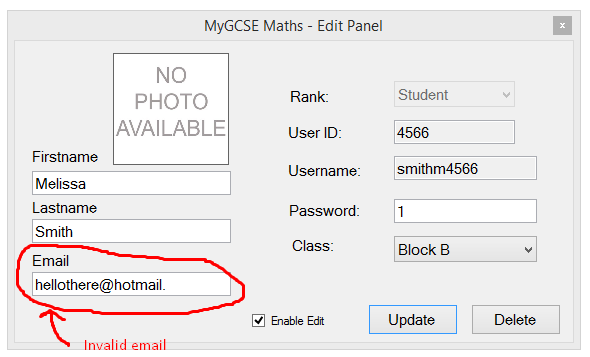
Screenshot 23.1



Screenshot 24.0

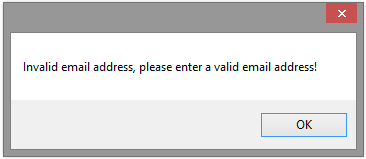


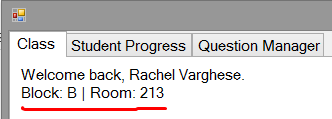
Screenshot 24.1



Screenshot 19A.0

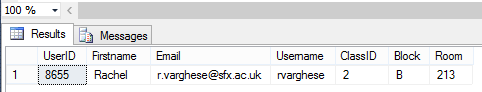
This shows that the regex expression used to validate this email is working.



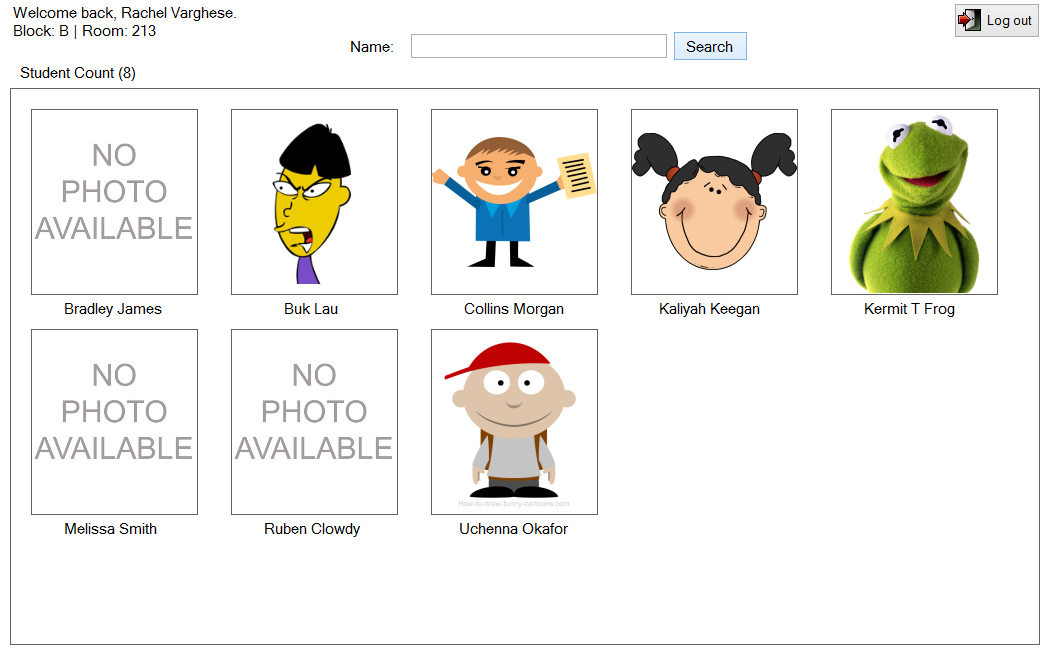


The welcome back label shows a welcome message to the teacher and shows them the information of the class they’re currently teaching. This shows that allocating a teacher to a class has been successful.

Screenshot 25.0

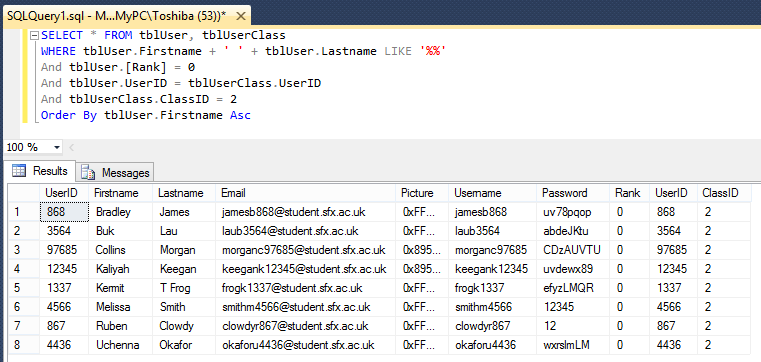


Screenshot 25.1



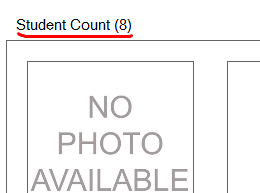
Screenshot 26.0

The SQL query in screenshot 26.1 shows all the students in the teacher’s class that matches the name searched for. Because the search text is empty, a list of all the students in the teacher’s class will be displayed. I coded an algorithm that would read the data from the SQL query response and display each student found in a predefined pattern, which is the grid like pattern (Screenshot 26.0). Each picture box and label represents a student that has been assigned to the teacher’s class.



Screenshot 26.1

esfsdf

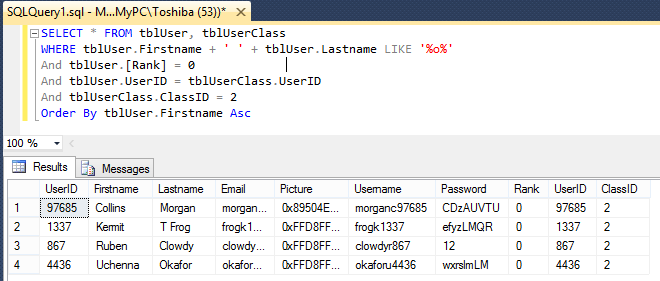


Screenshot 27.0

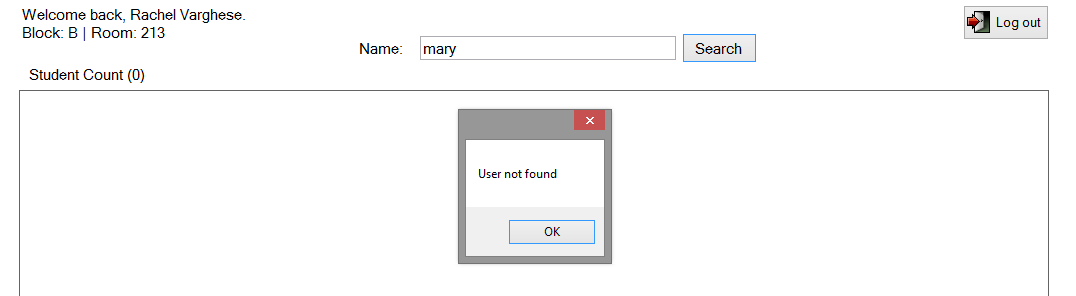


Screenshot 28.0

The system only shows the students found that contain “o” in their name. Screenshot 28.1 shows the SQL query that was used behind the scenes.

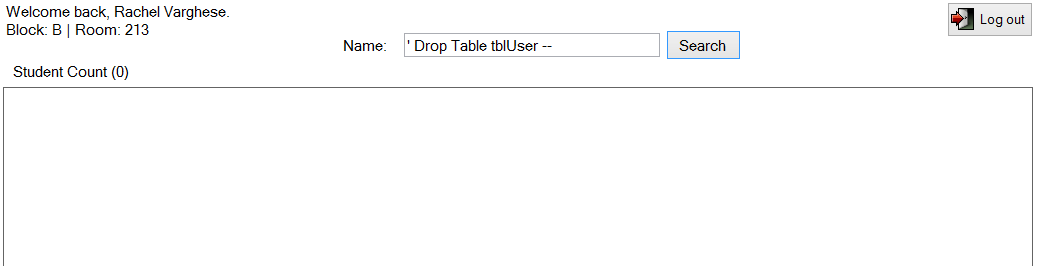
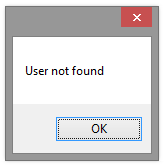


Screenshot 28.1

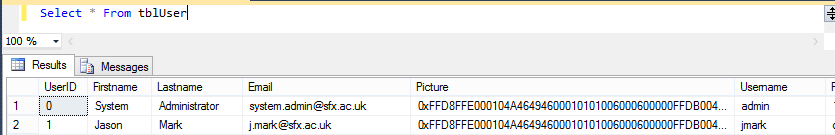


Screenshot 29.0

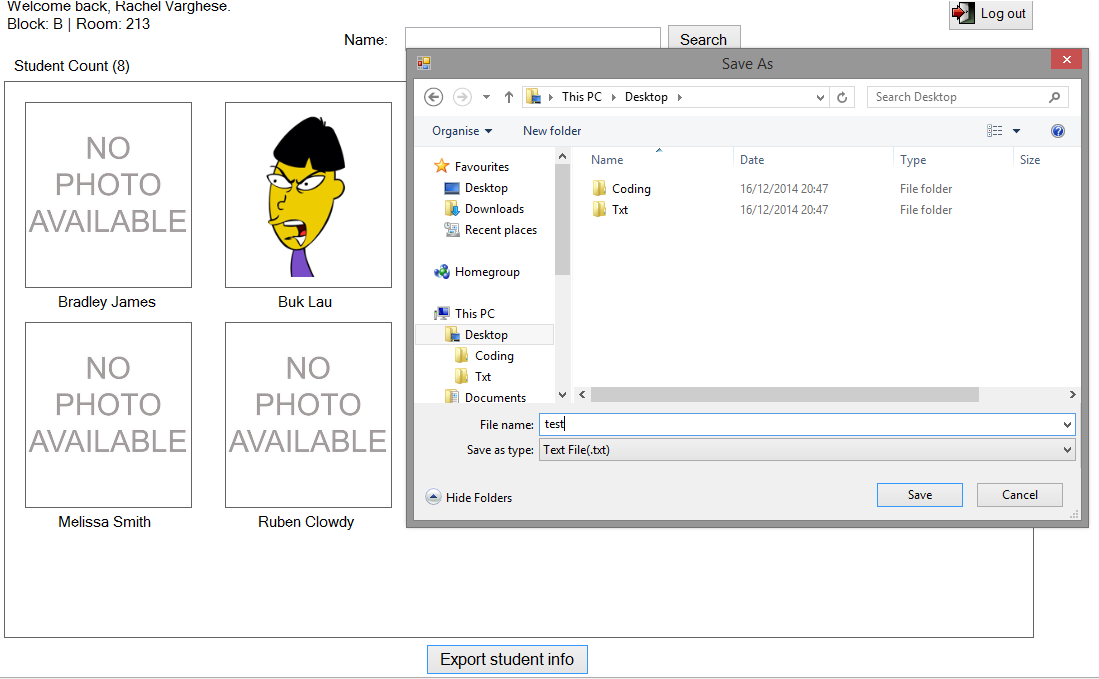
This is a valid SQL Injection attack, but because I parameterized all my SQL queries, it prevents my system from being subject to SQL injection attacks. If I didn’t implement parameterized queries, then all my tables would have been dropped. This test proves my system is not vulnerable to SQL injection. My tables are still intact, see Screenshot 30.1 for proof.



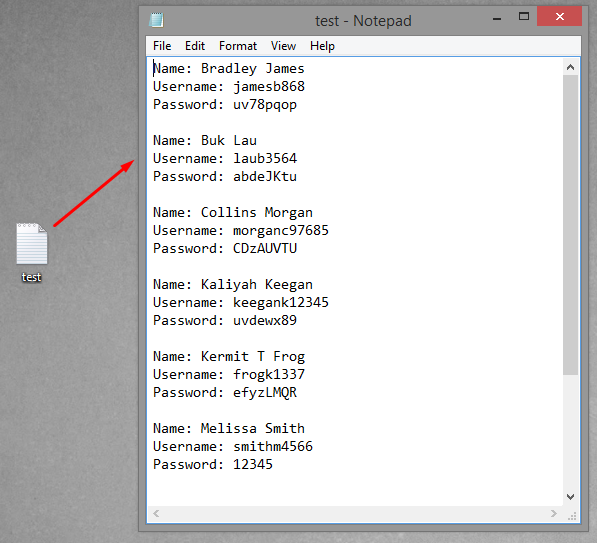
Screenshot 30.0



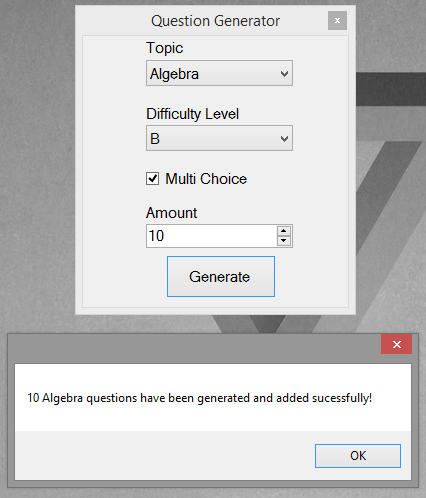
Screenshot 30.1



Screenshot 31.0

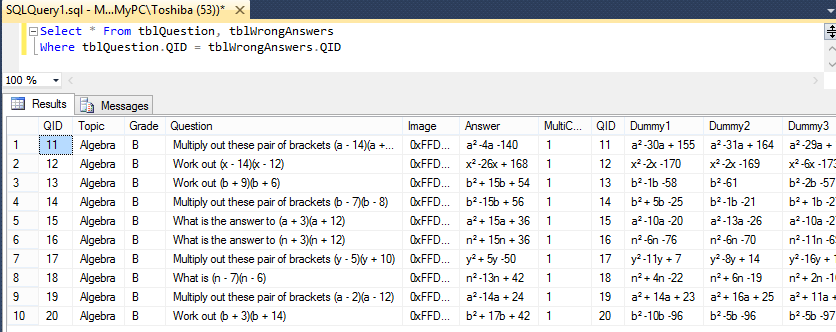


Screenshot 31.1

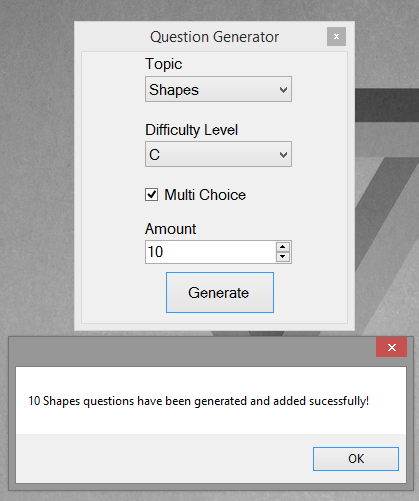


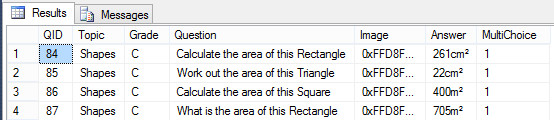
Screenshot 32.0

Generated questions are added to the SQL question table automatically. See screenshot 32.1 for proof.



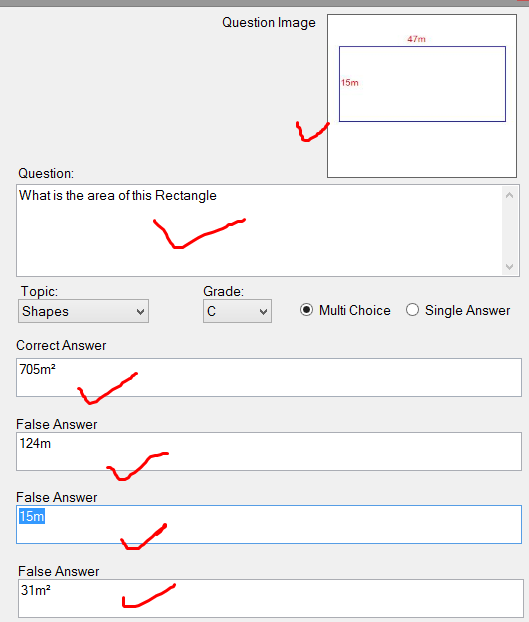
Screenshot 32.1





Screenshot 33.1

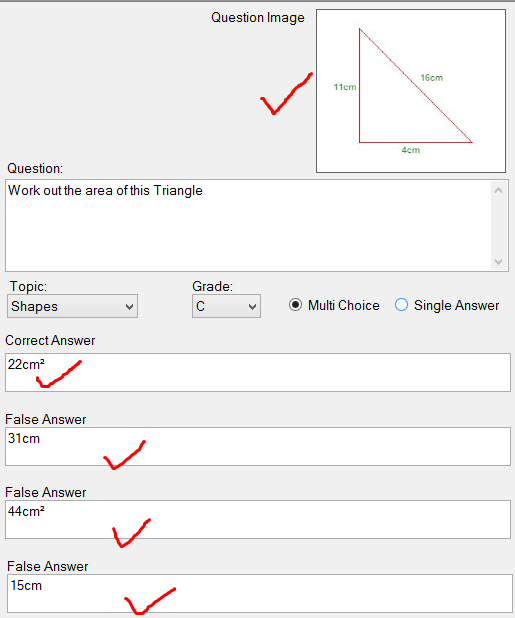
Screenshot 33.0



This is an example of one of the generated questions. This shows that the system is able to generate rectangle shape questions. The algorithm implemented is able to draw the shape, and label each side.

System is able to calculate the correct answer to the questions, and the false answers for the multiple choice questions.

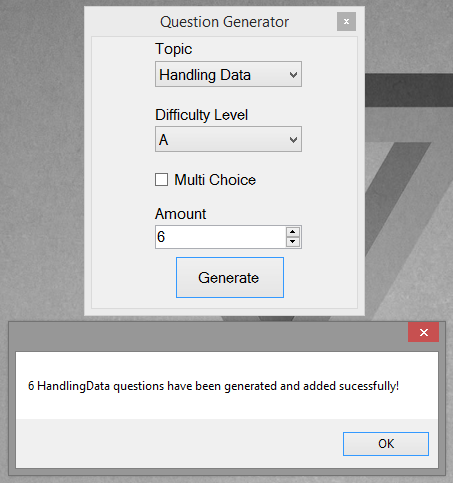
Screenshot 33A.0



System able to generate a triangle and label the sides accurately.

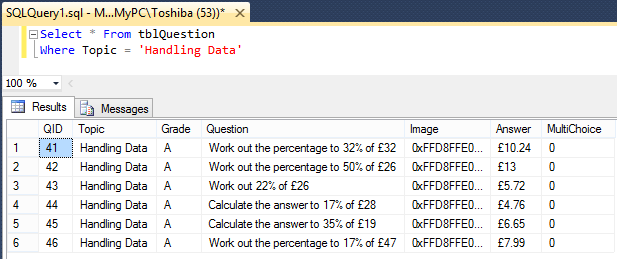
Screenshot 33A.1

The pictures in Screenshot 33A.0 & 33A.1 show 2 samples of the 10 shapes questions that had just been generated in screenshot 33.0. Once a question is generated, they’re added to the SQL database, then I can view the question in more detail via my system. And that’s what I did in Screenshot 33A.0 & 33A.1. This shows that my shapes generating algorithm works perfectly. It has drawn the shapes at runtime using the graphics object in VB.NET, and it has also applied the labels correctly. The system has also been successful in calculating the correct answer to the question, and generating false answers.

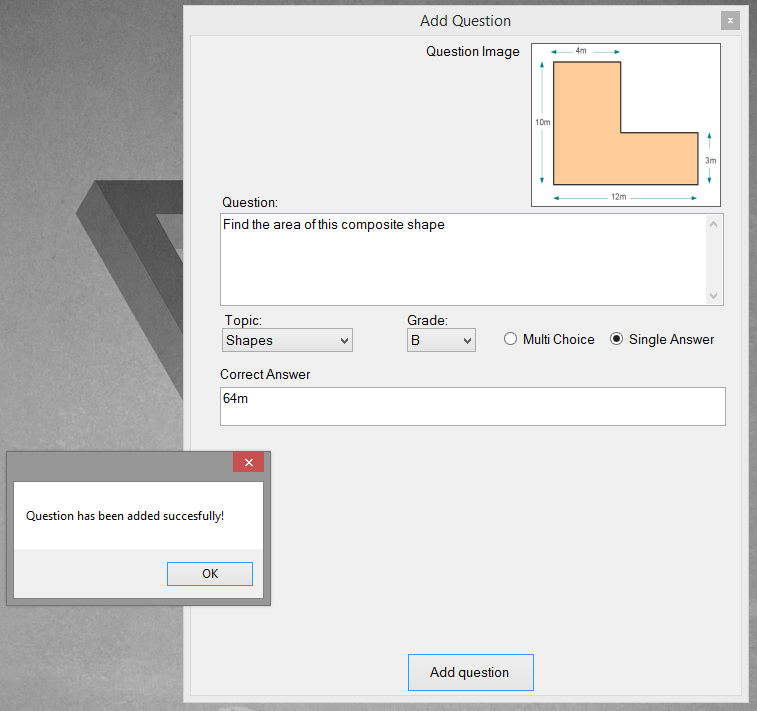


Screenshot 34.0

These are the questions that have been generated in Screenshot 34.0

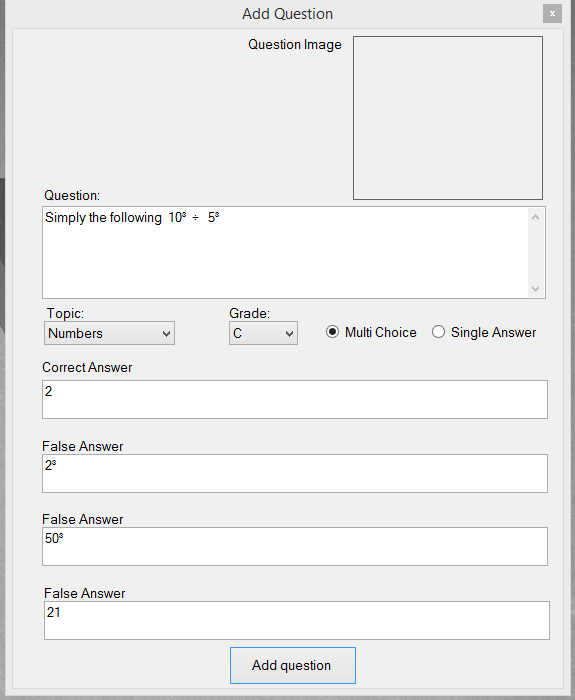


Screenshot 34.1

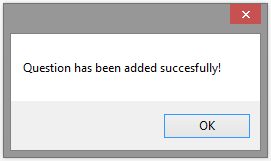


Screenshot 35.0

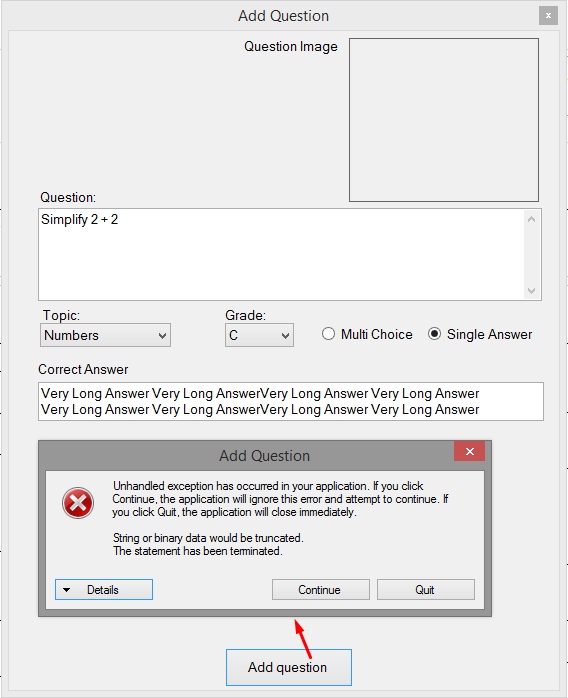
This shows that questions can be added manually.



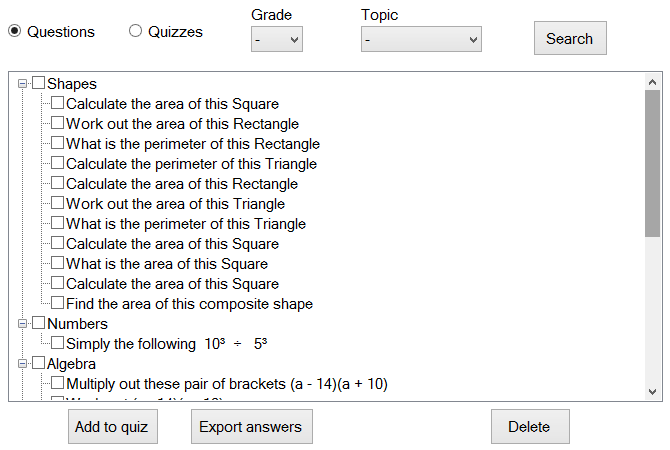
Screenshot 36.0



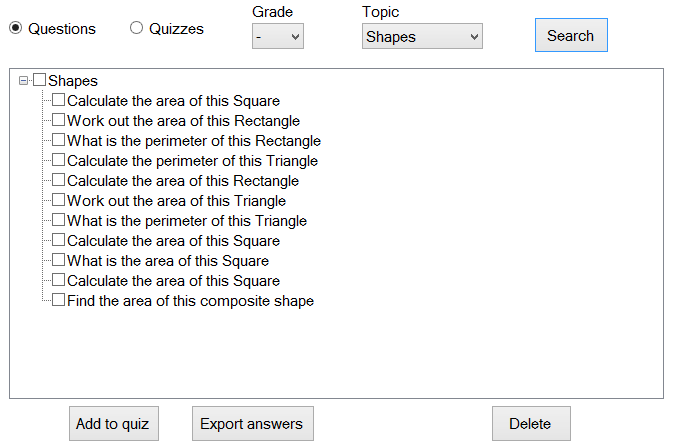
This shows that questions can be added manually.



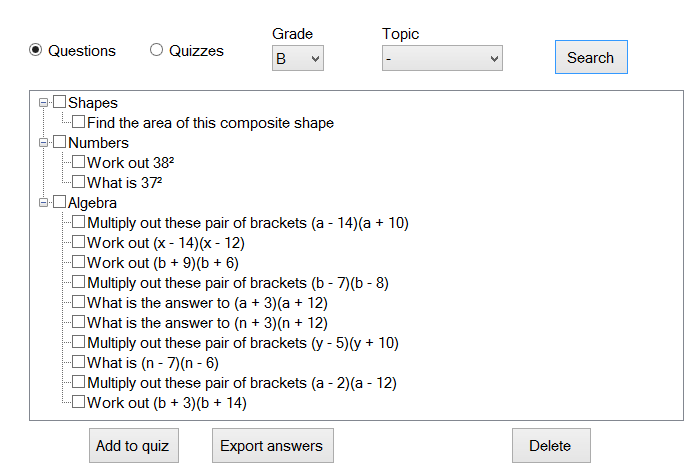
Screenshot 37.0



Screenshot 38.0

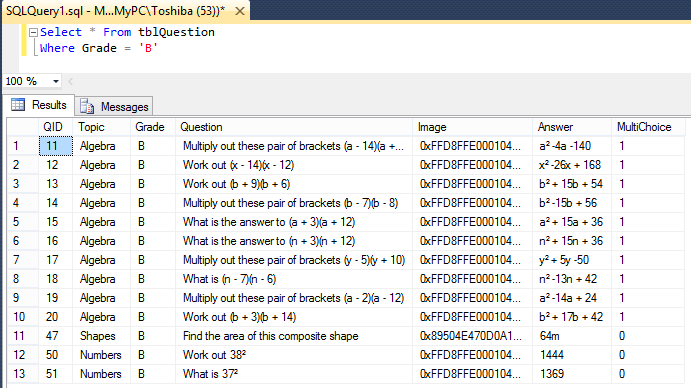


Screenshot 39.0

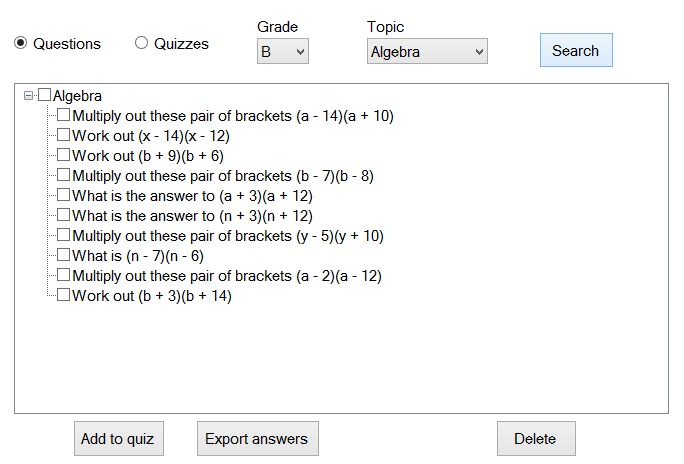


Screenshot 40.0

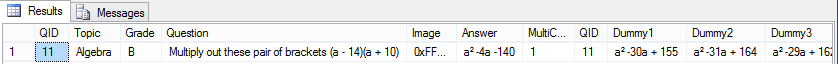
This is the underlying SQL query used to retrieve the data of all the questions of the grade difficulty B



Screenshot 40.1

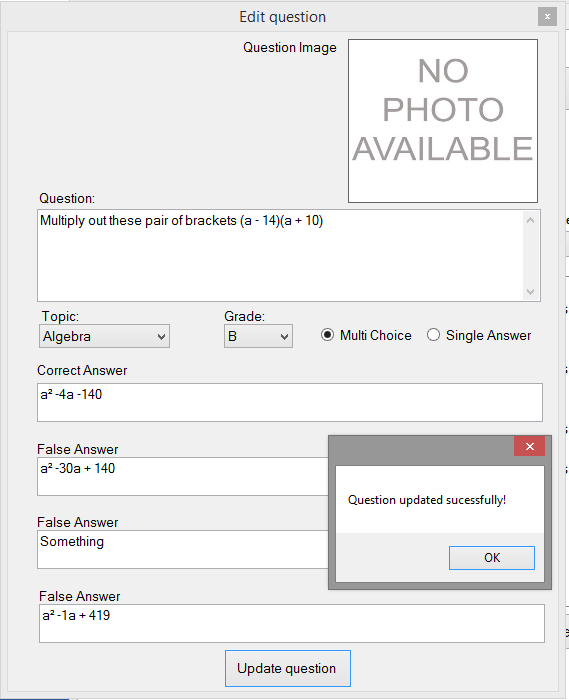


Screenshot 41.0



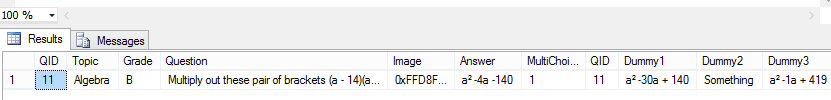
Before the question was edited.

Screenshot 42.0



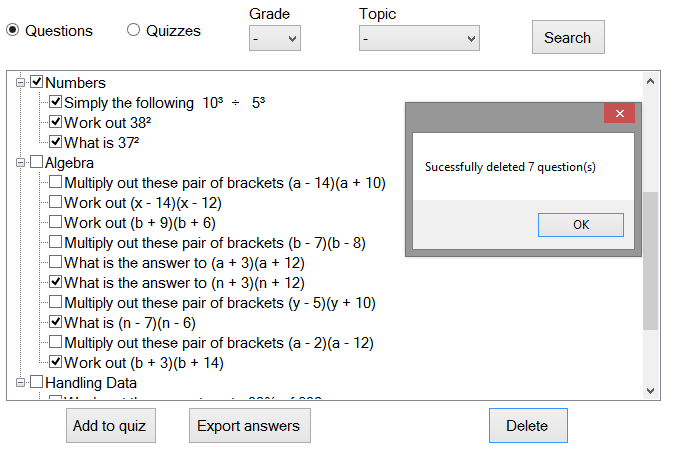
This shows updating questions in my system works perfectly. Even multi choice questions.

Screenshot 42.1

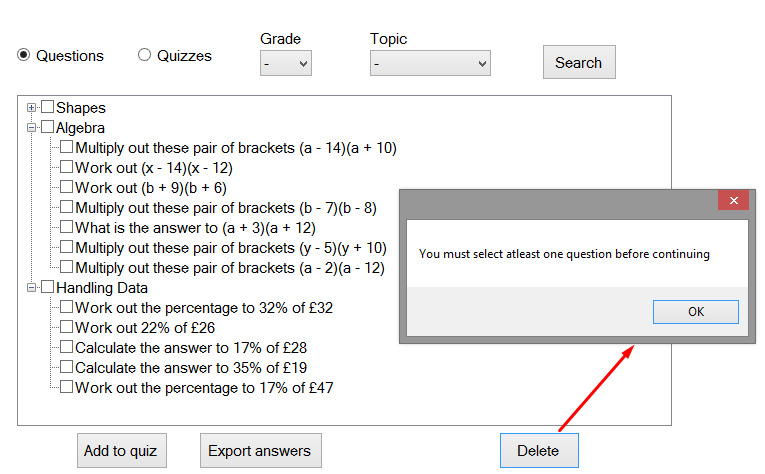


After the question was edited.

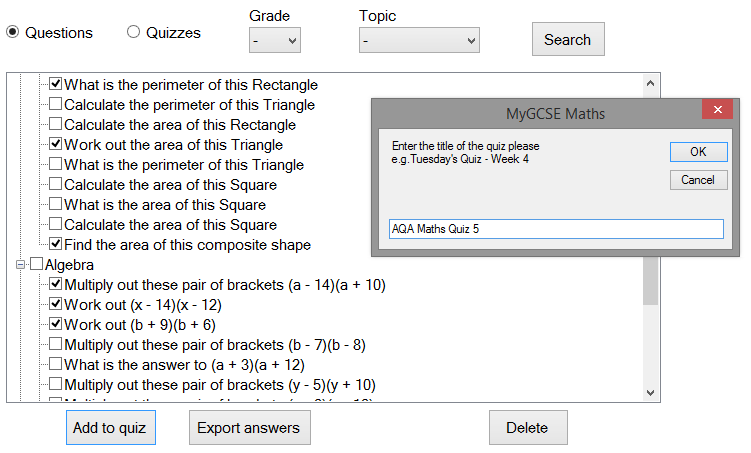
Screenshot 42.2



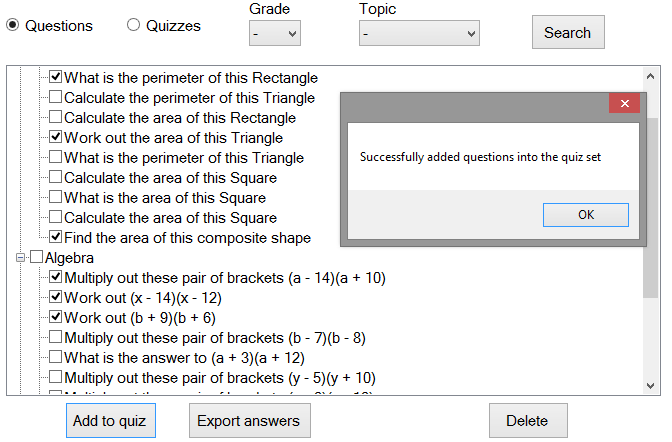
Screenshot 43.0



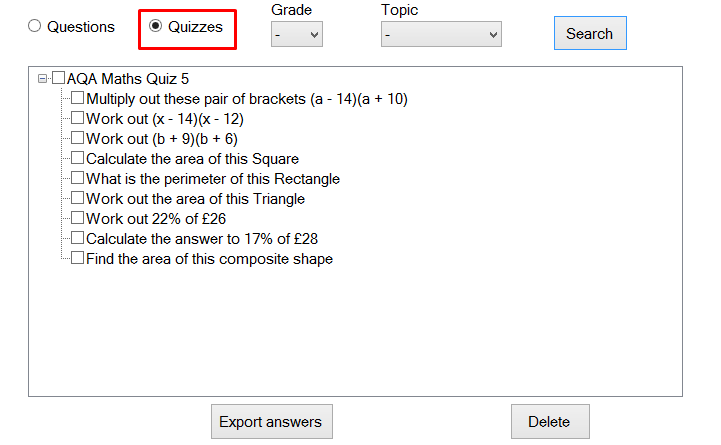
Screenshot 44.0



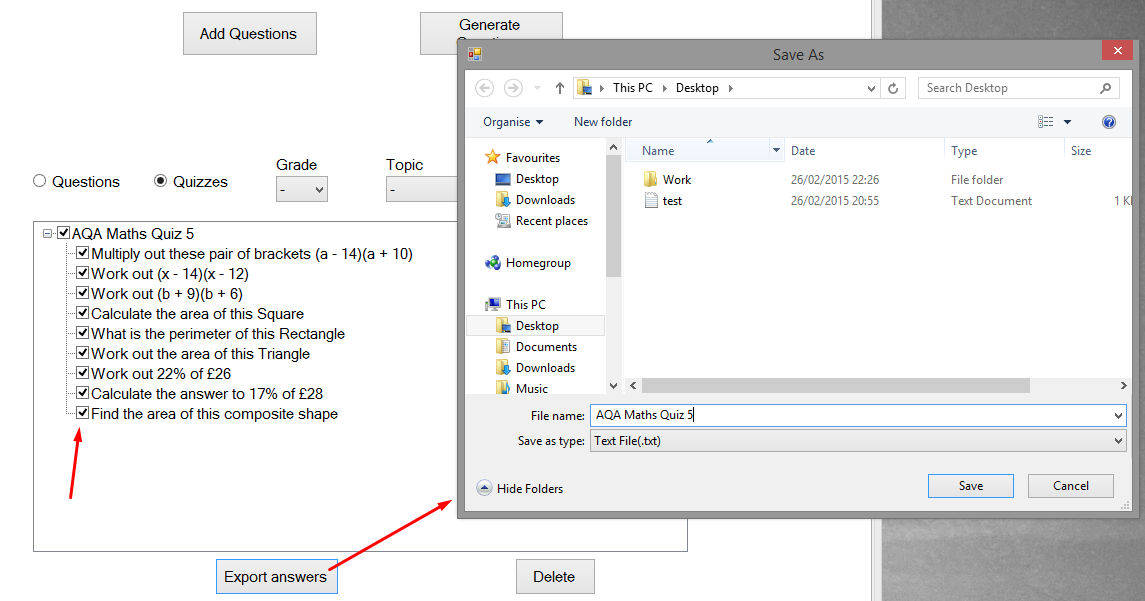
Screenshot 45.0



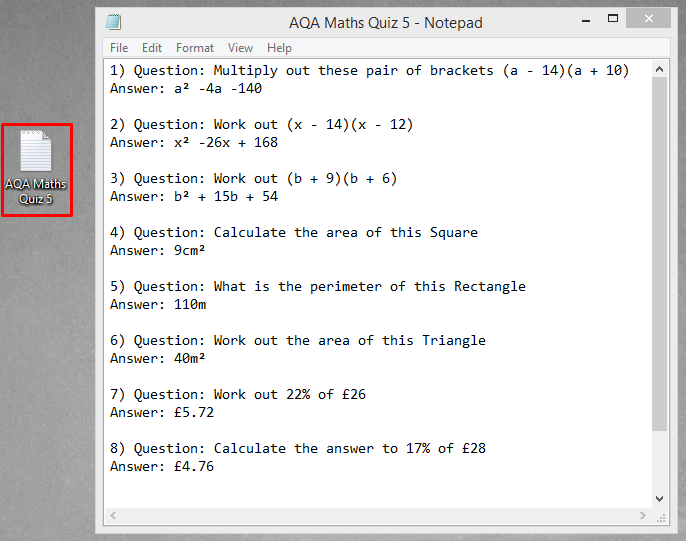
Screenshot 45.1



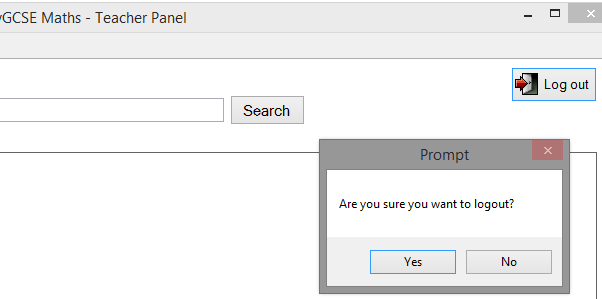
Screenshot 46.0



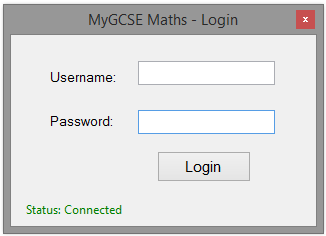
Screenshot 47.0



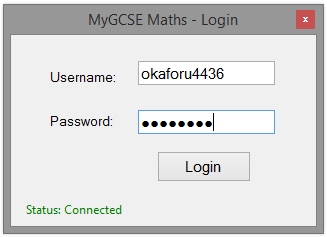
Screenshot 47.1



Screenshot 48.0

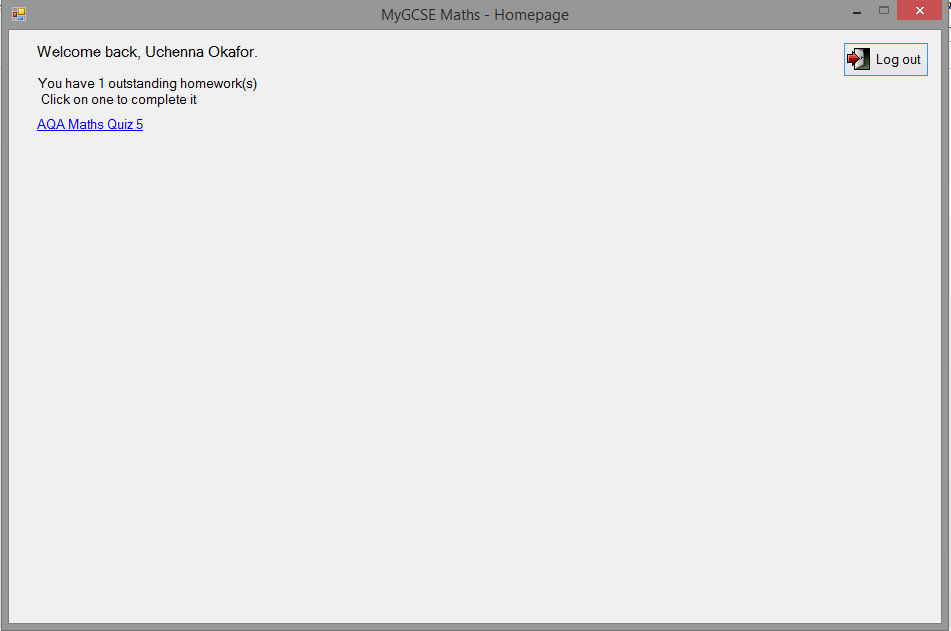


Screenshot 48.1

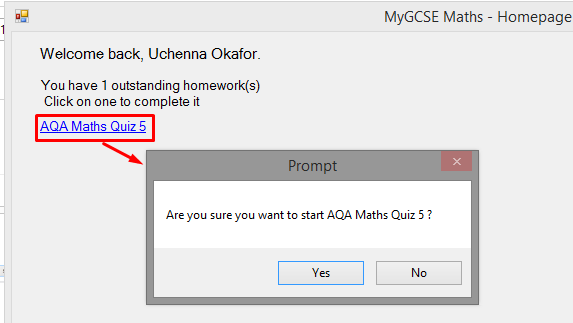


Screenshot 49.0

Screenshot 49.1



This is an outstanding quiz that the student has not yet completed. This was set previously by their teacher in screenshot 45.0 and 45.1

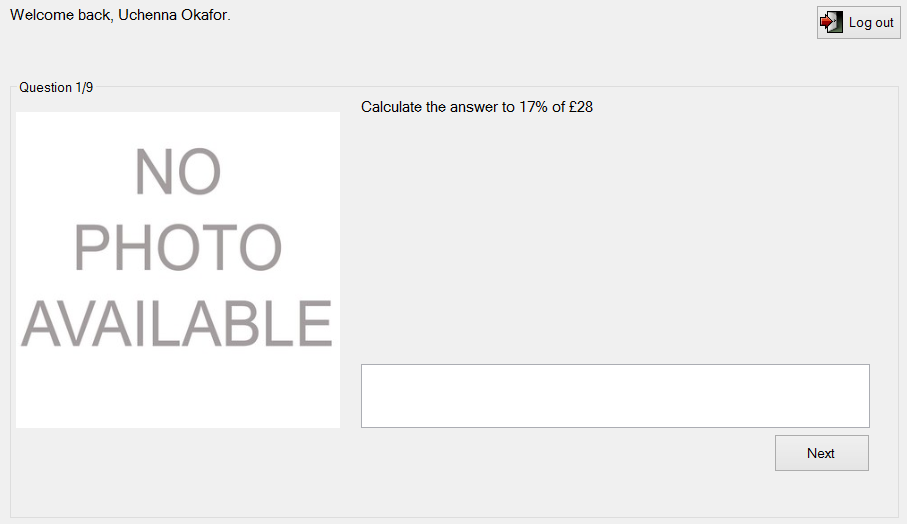


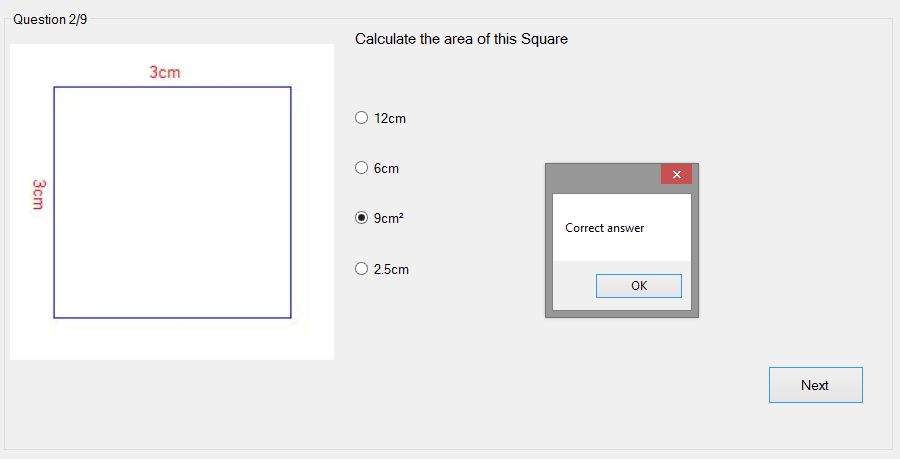
An algorithm is used to display a list of all the quizzes the student hasn’t completed. When the quiz/label is checked, the system prompts the user to start, if they click yes, the system will then fetch all the question IDs associated with the chosen quiz. It then loads all the QuestionIDs of a quiz into a queue data structure and de-queues the queues to get the next question ID each time a student answers a question.

Screenshot 50.0

Screenshot 51.0

Screenshot 50.1



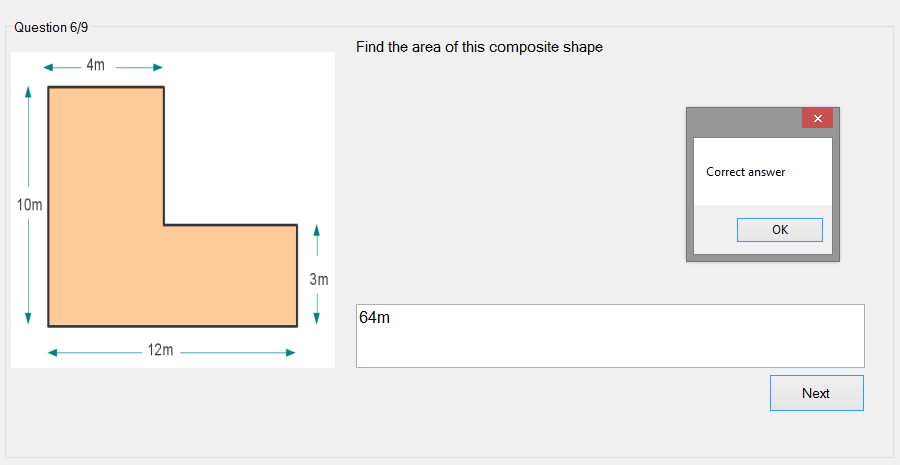


Screenshot 53.0

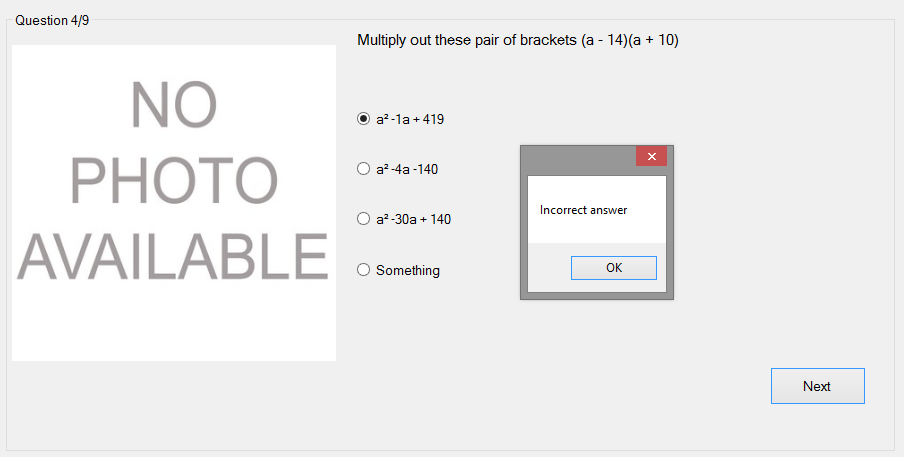
Screenshot 52.0

Once the next button is clicked, the user is alerted if they got the answer wrong or right and the system will then fetch the next question until there are no more questions.

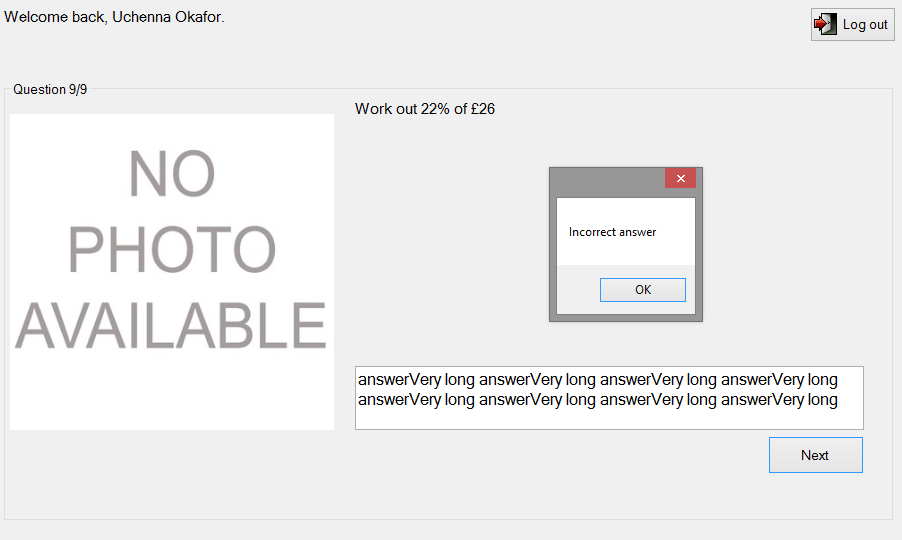
This was the shape question that was generated by the system at run time. This question was generated in screenshot 33.0. The radio buttons contain the text of the false answers and correct answer all in a randomized order.



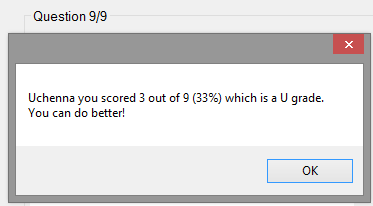
Screenshot 54.0



Screenshot 55.0

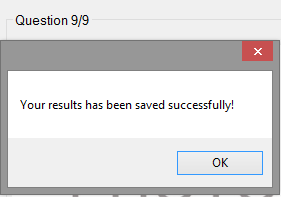


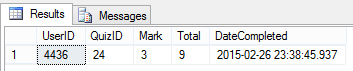
Screenshot 56.0



Screenshot 57.0

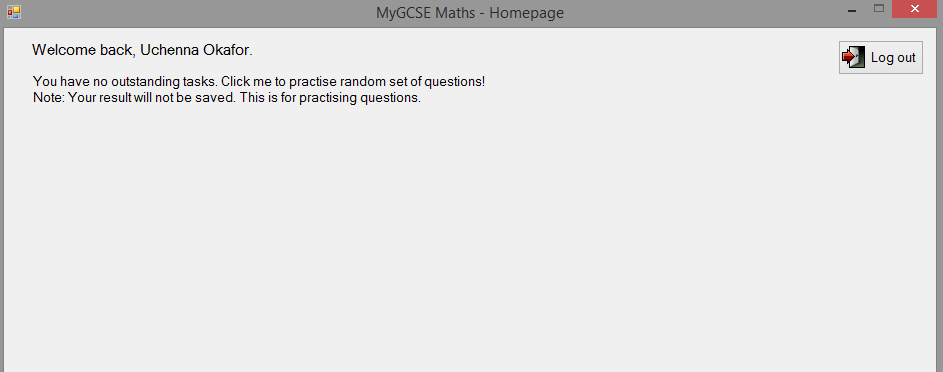
This is the message that will be seen when the user has completed all the questions in the quiz. If they get below a C they get a message saying well done, if they get less than a C it says you can do better.



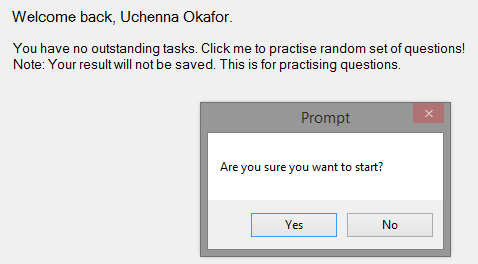


Screenshot 57.2

Screenshot 57.1



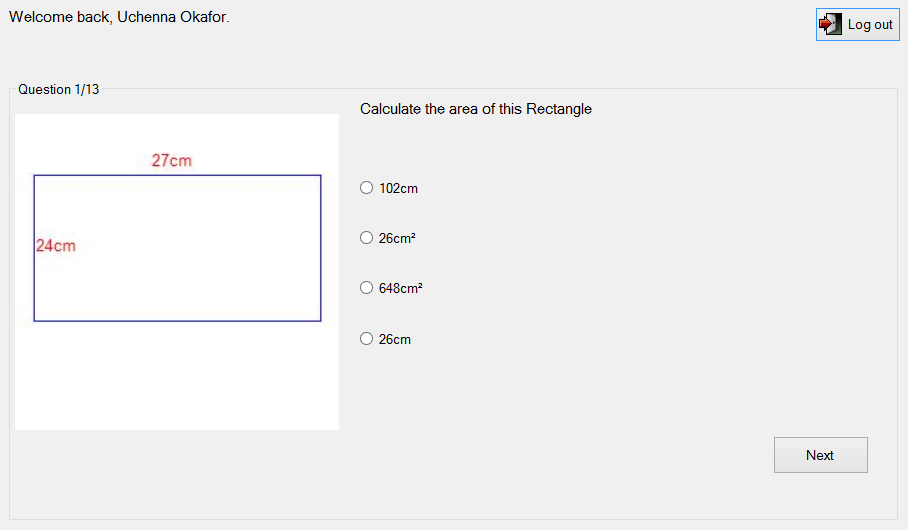
Screenshot 58.0



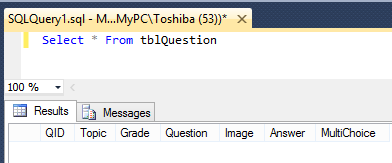
Clicking the label prompts to start a set of random quizzes that are already in the system.

Selects a random number of questions between 10 and 20 for the students to answer. The result of the random set of questions are only shown to the students. The teachers do not see this, and therefore it is not saved in the SQL table. The teacher can only see the result of the quizzes they set students.

Screenshot 59.0

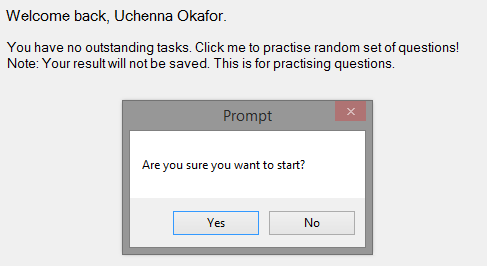


Screenshot 59.1

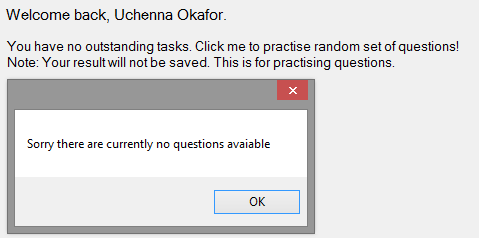


Screenshot 61.0

I have deleted all the questions in the database, so when the student requests to do a set of random questions, the system will alert them that there are no questions available in the system

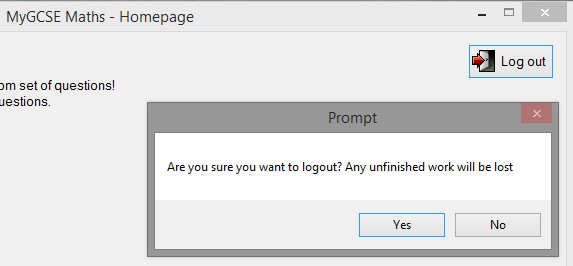


Screenshot 61.1



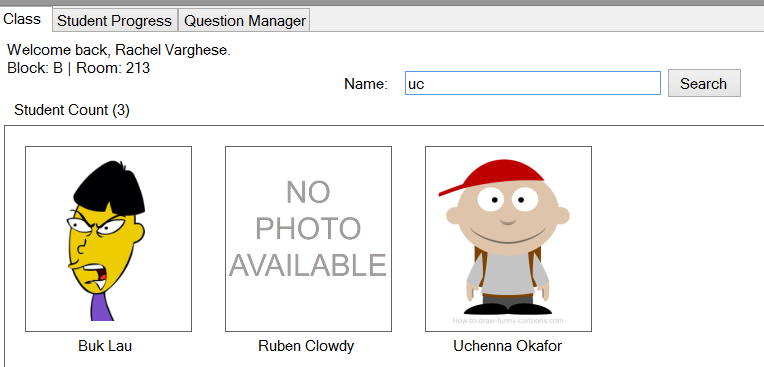
Screenshot 61.2

Screenshot 61.2



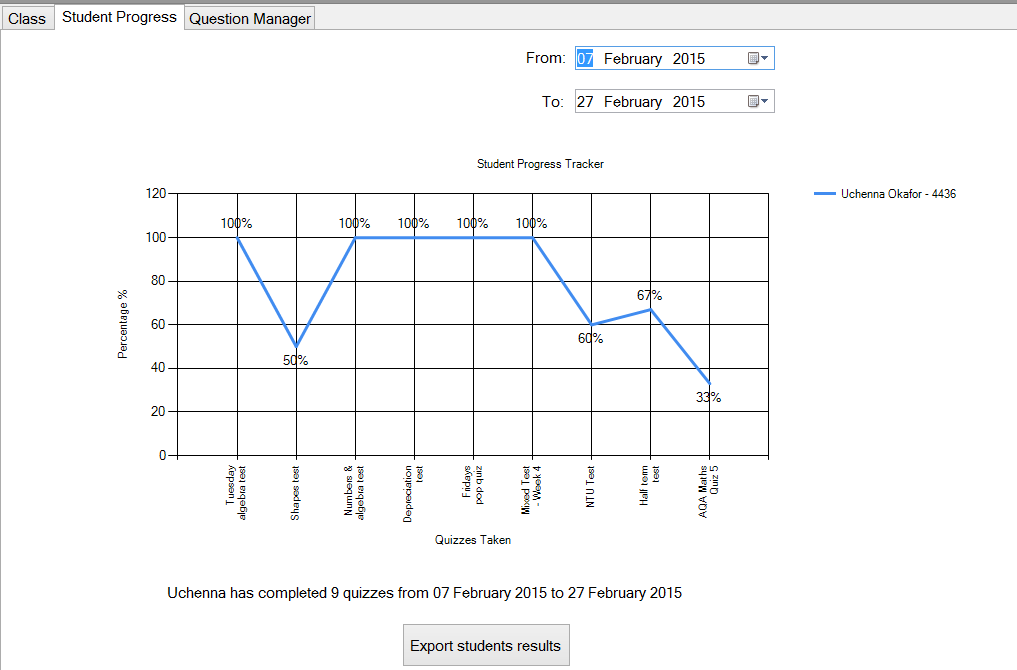
Screenshot 62.0

The logout message is different from the logout message in the admin and teacher form. This is because this message is overridden in the students form.

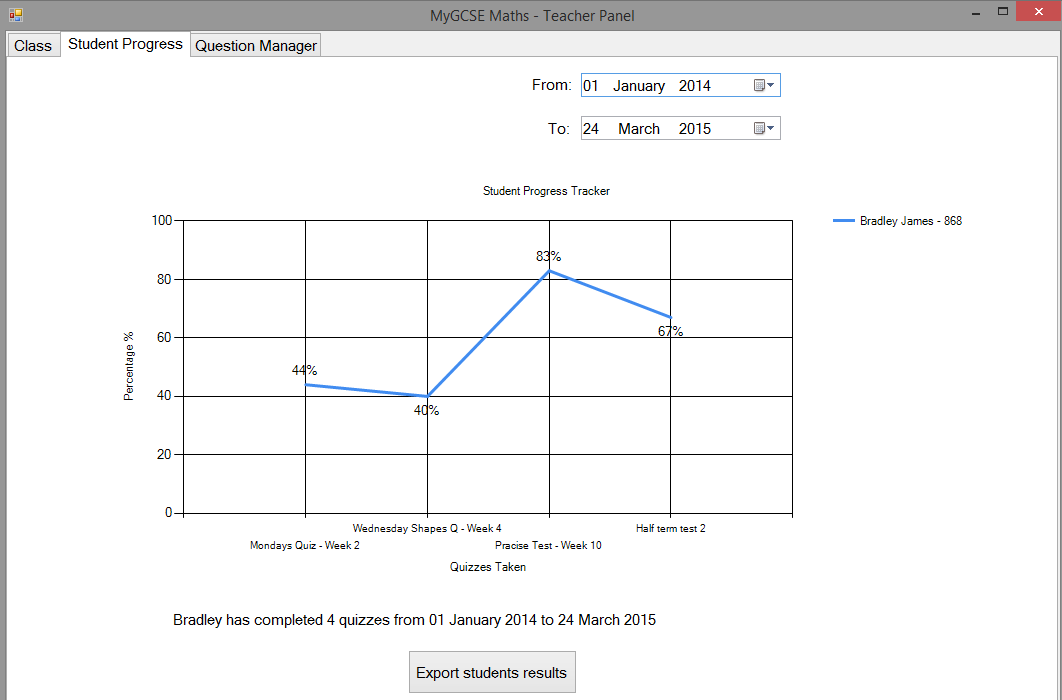


Screenshot 63.0

Clicking the picture box takes the teacher to the next tab page which is Student Progress tab. The system will then load the students’ progress on a graph.



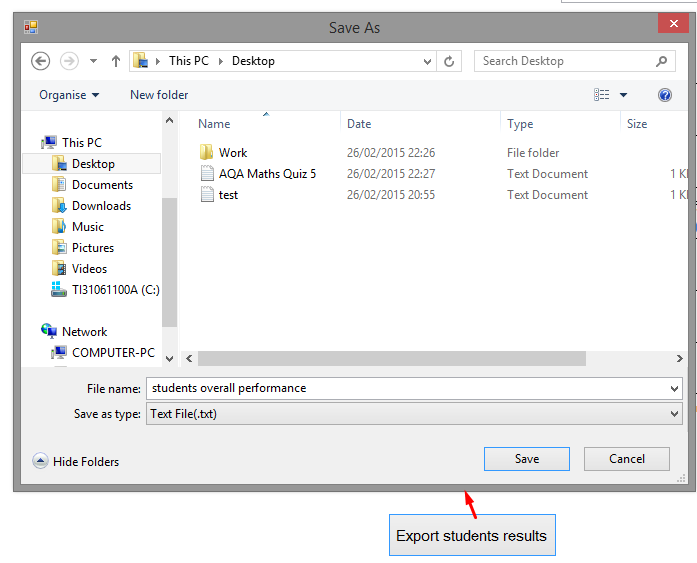
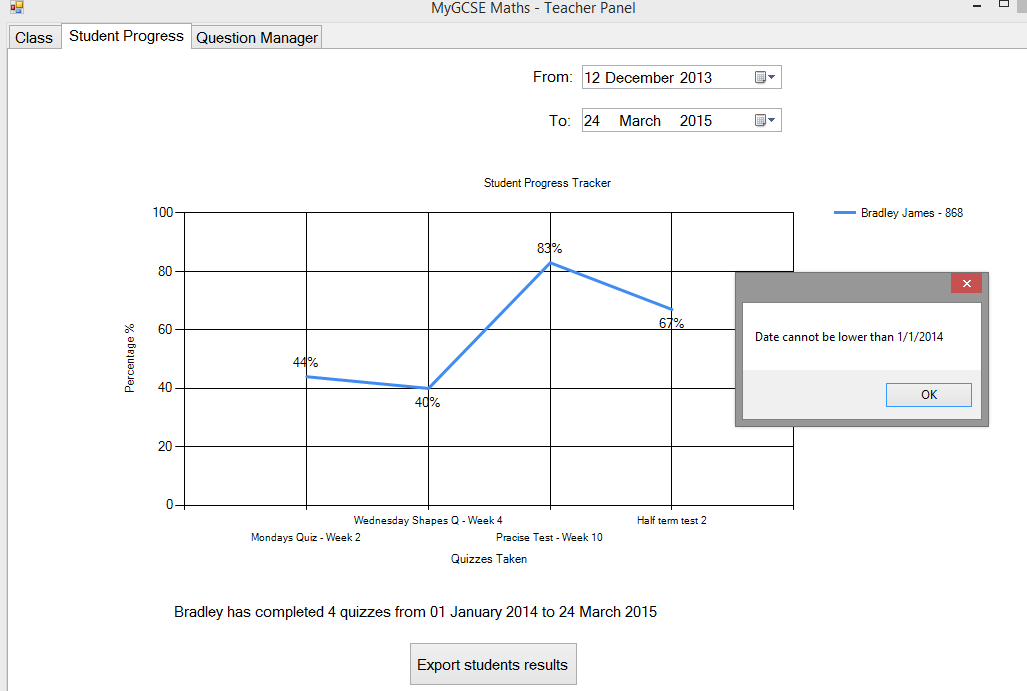
This is the result of the students’ performance on a quiz they have recently completed in screenshot 57.0, 57.1



Screenshot 64.0

Boundary data

Screenshot 63.1



Date is below 01/01/2014. Therefore error message is shown.

Screenshot 65.0

Screenshot 66.0

Screenshot 65.0



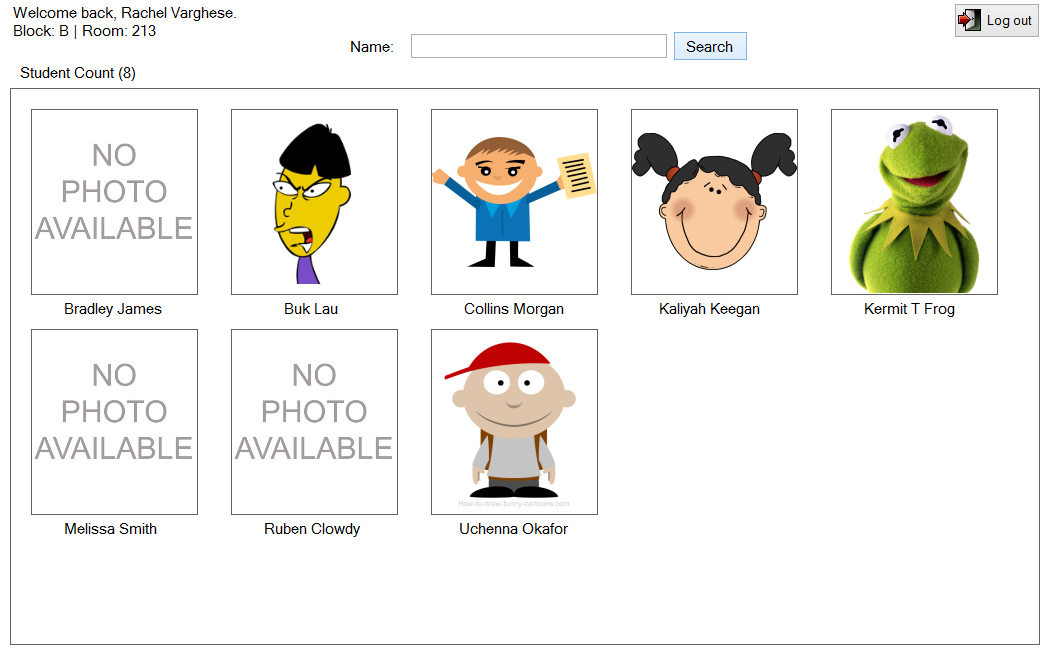
Screenshot 66.1

An algorithm is used to get a list of all the students in the teachers’ class who have completed one or more quizzes. Each student found is then sorted from lowest to highest lowest according to their overall quiz average using a quick sort algorithm.

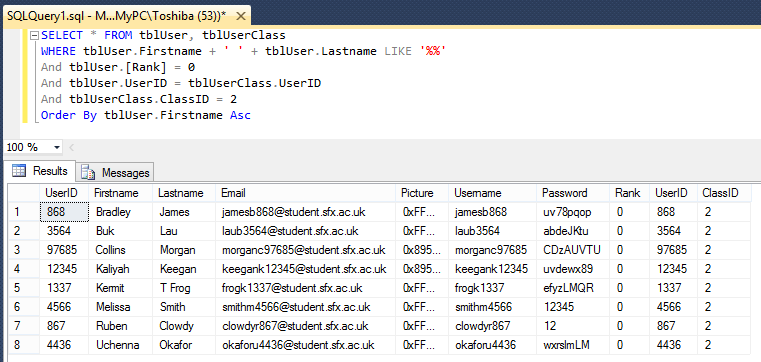
This .txt file shows how many quizzes each student has completed and the average of all those quizzes. This way the teacher can know each students overall performance. Therefore this shows that my quick sort algorithm works perfectly as the students are sorted in order of low to high.

Trace Tables

**CreateDynamicStudents algorithm**



The image below is the SQL query performed in order to retrieve the data of all the students in a teacher’s class.

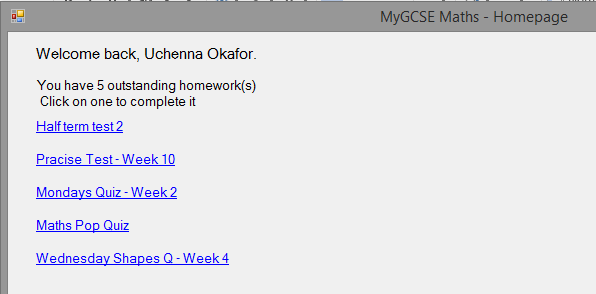


|  |
| --- |
| Code:  Private Sub CreateDynamicStudents(students As List(Of User))  ClearControls() 'Clears all the pre-exiting controls in the panel first.  Invoke(Sub() btnSearch.Enabled = False)  'Initial locations  Dim xLocation As Integer = 20  Dim yLocation As Integer = 20  Dim amountCreated As Integer = 0  For i = 0 To students.Count - 1  Dim picBox As New CustomPictureBox  'Instanciate the CustomPictureBox object.  'Defines all its properties.  picBox.Name = students(i).Firstname & " " & students(i).Lastname  picBox.UserID = students(i).UserID  picBox.Username = students(i).Username  picBox.Password = students(i).Password  picBox.Size = New Size(167, 186)  picBox.Location = New Point(xLocation, yLocation)  picBox.BorderStyle = BorderStyle.FixedSingle  picBox.SizeMode = PictureBoxSizeMode.Zoom  picBox.Cursor = Cursors.Hand  picBox.Image = Image.FromStream(students(i).Image.ToStream())  CreateDynamicLabels(picBox.Name, xLocation, yLocation)  'Creates the label of the student the picturebox represents underneath the picturebox  amountCreated += 1  xLocation += 200  'Checks the amount created. If it's 5. The it changes it increments the yLocation.  'Which moves it down, and resets the xLocation to the left of the Parent Container.  If amountCreated = 5 Then  xLocation = 20  yLocation += 220  amountCreated = 0  End If  AddHandler picBox.Click, AddressOf StudentPicture\_Clicked  Invoke(Sub() Me.studentsView.Controls.Add(picBox))  'Adds the object to the Panel1.Controls.  'We also add a click event handler.  'We're invoking the methods because we're running this on a seperate thread'  'And to make a safetly call we need to Invoke the call using a delegate sub.  'Or we'd get a CrossThreadCallExcpetion thrown.  Next  Invoke(Sub() Me.studentsView.Focus())  Invoke(Sub() btnSearch.Enabled = True)  End Sub |

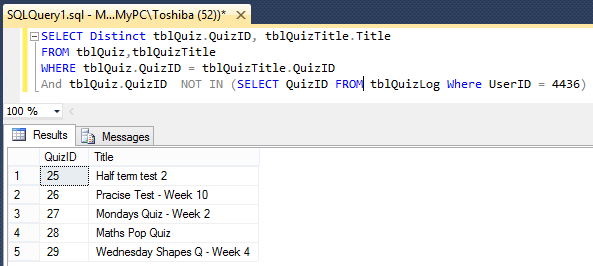
This is the trace table for the CreateDynamicStudents algorithm. This algorithm is only for placing the picture boxes in the correct place, it does not account for the labels created that show the students name. The calculations required to place the label in the right place is the GuessEstimateX algorithm which can be found in the detailed algorithms in the system maintenance section.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **i** | **xLocation** | **yLocation** | **amountCreated** | **Picbox.Name** | **Picbox.UserID** | **Picbox.Username** | **Picbox.Password** | **Picbox.Size** | **Picbox.Location** |
| 0 | 20 | 20 | 0 | Bradley James | 868 | jamesb868 | uv78pqop | {Width=167, Height=186} | {X=20,Y=20} |
| 1 | 220 | 20 | 1 | Buk Lau | 3564 | laub3564 | abdeJKtu | {Width=167, Height=186} | {X=220,Y=20} |
| 2 | 420 | 20 | 2 | Collins Morgan | 97685 | morganc97685 | CDzAUVTU | {Width=167, Height=186} | {X=420,Y=20} |
| 3 | 620 | 20 | 3 | Kaliyah Keegan | 12345 | keegank12345 | uvdewx89 | {Width=167, Height=186} | {X=620,Y=20} |
| 4 | 820 | 20 | 4 | Kermit T Frog | 1337 | frogk1337 | efyzLMQR | {Width=167, Height=186} | {X=820,Y=20} |
| 5 | 20 | 240 | 0 | Melissa Smith | 4566 | smithm4566 | 12345 | {Width=167, Height=186} | {X=20,Y=240} |
| 6 | 220 | 240 | 1 | Ruben Clowdy | 867 | clowdyr867 | 12 | {Width=167, Height=186} | {X=220,Y=240} |
| 7 | 420 | 240 | 2 | Uchenna Okafor | 4436 | okaforu4436 | wxrslmLM | {Width=167, Height=186} | {X=420,Y=240} |

**CreateQuizLbls algorithm**

****

The SQL query used selects all the quizzes a student hasn’t completed yet. Using the CreateQuizLbls algorithm, the system will display each uncompleted quiz in a defined pre-defined pattern. When the user clicks the link label it will prompt to start the quiz and fetch all the questions in that quiz.



|  |
| --- |
| Code:  Private Sub CreateQuizLbls(quizList As List(Of Quiz))  ClearLinkLbls() 'Clears pre-existing CustomLinkLabels on the form  'Initial location  Dim xLocation As Integer = 25  Dim yLocation As Integer = 86  Dim amountCreated As Integer = 0  For i = 0 To quizList.Count - 1  Dim linkLbl As New CustomLinkLabel 'Creates a new instance on each iteration  linkLbl.QuizID = quizList(i).QuizID  linkLbl.Text = quizList(i).Title  linkLbl.Location = New Point(xLocation, yLocation)  linkLbl.AutoSize = True  'Defines the properties of the CustomLinkLabel  amountCreated += 1  yLocation += 33  'Checks the amount of labels created. If it's equal to 10  'It increments the xLocation to move to the right of the screen.  'And resets the yLocation and amount created.  If amountCreated = 10 Then  xLocation += 350  yLocation = 86  amountCreated = 0  End If  'Adds an event handler for the clicked event  AddHandler linkLbl.Click, AddressOf PromptStartQuiz  Me.Controls.Add(linkLbl)  Next  End Sub |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **i** | **xLocation** | **yLocation** | **Amount Created** | **linklbl.QuizID** | **linklbl.Text** | **linklbl.Location** |
| 0 | 25 | 86 | 0 | 25 | Half term test 2 | {X=25,Y=86} |
| 1 | 25 | 119 | 1 | 26 | Practice Test – Week 10 | {X=25,Y=119} |
| 2 | 25 | 152 | 2 | 27 | Mondays Quiz – Week 2 | {X=25,Y=152} |
| 3 | 25 | 185 | 3 | 28 | Maths Pop Quiz | {X=25,Y=185} |
| 4 | 25 | 218 | 4 | 29 | Wednesday Shapes Q – Week 4 | {X=25,Y=218} |

**Quick sort algorithm**

|  |
| --- |
| **Code:**  Private Class Quicksort  Public Shared Function Sort(arrayList As List(Of StudentQuizInfo)) As List(Of StudentQuizInfo)  'If list is empty of has 1 item then its already sorted  If arrayList.Count <= 1 Then  Return arrayList  Else  'Makes a call to the recursive sort algorithm  Dim sortedList As List(Of StudentQuizInfo) = SortList(0, arrayList.Count - 1, arrayList)  Return sortedList  End If  End Function  Private Shared Function SortList(lowIndex As Integer, hiIndex As Integer, arrayList As List(Of StudentQuizInfo)) As List(Of StudentQuizInfo)  If hiIndex > lowIndex Then  Dim initialPivot As Integer = GetPivotPosition(lowIndex, hiIndex, arrayList)  SortList(lowIndex, initialPivot - 1, arrayList)  SortList(initialPivot + 1, hiIndex, arrayList)  End If  Return arrayList 'Returns the sorted list at the end of the recusive loop  End Function  Private Shared Function GetPivotPosition(lowIndex As Integer, hiIndex As Integer, arrayList As List(Of StudentQuizInfo)) As Integer  Dim x As Integer = lowIndex  'Uses the first index as the starting point  Dim y As Integer = hiIndex - 1  'The pivot is going to be used to compare each item in the list  Dim pivot As Integer = arrayList(hiIndex).Average  Do  'Compares each item in the left sub-list that is greater than the pivot  While arrayList(x).Average <= pivot AndAlso x < hiIndex  x += 1  End While  'Compares each item in the right sub-list that is less than the pivot  While arrayList(y).Average >= pivot AndAlso y > lowIndex  y -= 1  End While  If x < y Then Swap(x, y, arrayList) ' Swaps X and Y  Loop While x < y 'Keeps looping until Y is greater than X  ' Swaps the data from the right to the new position in the list  If arrayList(x).Average > pivot Then Swap(x, hiIndex, arrayList) 'Exchanges the data  Return x ' Returns the position of the pivot  End Function  Private Shared Sub Swap(indexFrom As Integer, indexTo As Integer, ByRef arrayList As List(Of StudentQuizInfo))  Dim selectedStudent As StudentQuizInfo = arrayList(indexFrom) 'selected student to swap  arrayList(indexFrom) = arrayList(indexTo)  arrayList(indexTo) = selectedStudent  'Swaps the current with the specified index  End Sub  End Class |

**Note:** Avg means Average. This is the average percentage the student has achieved in all tests they have completed.

**arrayList**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Pivot** | **[0]** | **[1]** | **[2]** | **[3]** | **[4]** | **[5]** | **[6]** | **[7]** |
|  | Ruben  Avg 56 | Bradley  Avg 58 | Kermit  Avg 67 | Buk  Avg 64 | Uchenna  Avg 78 | Melissa  Avg 51 | Kaliyah  Avg 65 | Collins  Avg 100 |
| 100 | Ruben  Avg 56 | Bradley  Avg 58 | Kermit  Avg 67 | Buk  Avg 64 | Uchenna  Avg 78 | Melissa  Avg 51 | Kaliyah  Avg 65 | Collins  Avg 100 |
| 65 | Ruben  Avg 56 | Bradley  Avg 58 | Melissa  Avg 51 | Buk  Avg 64 | Kaliyah  Avg 65 | Kermit  Avg 67 | Uchenna  Avg 78 | Collins  Avg 100 |
| 64 | Ruben  Avg 56 | Bradley  Avg 58 | Melissa  Avg 51 | Buk  Avg 64 | Kaliyah  Avg 65 | Kermit  Avg 67 | Uchenna  Avg 78 | Collins  Avg 100 |
| 51 | Melissa  Avg 51 | Bradley  Avg 58 | Ruben  Avg 56 | Buk  Avg 64 | Kaliyah  Avg 65 | Kermit  Avg 67 | Uchenna  Avg 78 | Collins  Avg 100 |
| 56 | Melissa  Avg 51 | Ruben  Avg 56 | Bradley  Avg 58 | Buk  Avg 64 | Kaliyah  Avg 65 | Kermit  Avg 67 | Uchenna  Avg 78 | Collins  Avg 100 |
| 78 | Melissa  Avg 51 | Ruben  Avg 56 | Bradley  Avg 58 | Buk  Avg 64 | Kaliyah  Avg 65 | Kermit  Avg 67 | Uchenna  Avg 78 | Collins  Avg 100 |
|  | Melissa  Avg 51 | Ruben  Avg 56 | Bradley  Avg 58 | Buk  Avg 64 | Kaliyah  Avg 65 | Kermit  Avg 67 | Uchenna  Avg 78 | Collins  Avg 100 |

The first row contains the unsorted data and the last row contains the sorted data. The algorithm sorted based on the total average mark of each student.