Artificial Intelligence Maths Tutor

Documentation

A-Level Project  
James Bungay

Table of contents

Analysis 4

Background to the problem and client 4

The current system 4

Interview with the client, Mrs Taylor 6

Specification part I: Specification for student-side of the program 9

Specification part II: Specification for teacher-side of the program 10

Specification part III: Specification for the canvas user-interface control 12

Specification part IV: Specification for the artificial intelligence character recognition engine 13

Time plan for the project 15

Limitations of the system 16

Designs 17

High-level overview of the program’s structure 17

Proposed data structures 18

Database design 19

System and data security 26

User interface design 28

Design of core algorithms 44

Pseudocode of core algorithms 44

Design of core SQL statements 49

Prototyping of the AI character classification system 52

Prototyping of the drawing canvas UI control 63

Research sources for program development 68

Technical Solution 69

Testing of the Technical Solution 73

Introduction 73

Test Plan 73

Documented Testing 87

Evaluation 152

Evaluative interview with the client, Mrs Taylor 152

Review of specification 155

Is this actually artificial intelligence? 160

Conclusion 160

***The annotated technical solution source code has been attached as an appendix.***

# Analysis

## Background to the problem and client

In recent years, student intake to the Ecclesbourne School has significantly increased, from seven class-groups of 30 students per year to nine class-groups in the most recent cohort. With the increased intake, more students have a variety of backgrounds and come from a variety of primary schools; therefore, the range of level of education between new Year 7 students is becoming pointedly greater.

Due to this, the maths department of the school is becoming increasingly concerned about their ability to bring every student up to the same level in order to begin secondary education, and teach the fundamentals of maths which should have been taught in primary school to the students who are missing that knowledge.

Up to this point in time, students who have not had a sufficient primary education in maths would have been given extra support and teaching time to learn the maths they missed. However, with a limited number of teachers and an increasing number of students, the maths department cannot provide as much extra teaching as they would like to do, and are struggling with the current system of marking paper homeworks which they set for the large number of students.

In order to help solve this problem, the head of the maths department, Mrs Megan Taylor, who is the client for this project, has come forward. She has decided that she would like a computer program to be developed, which can be used to set tasks to students and class-groups for students to complete, replacing paper homework tasks. She would also like students to be able to use the program independently, to practice maths topics which they struggle with. An interview has been conducted with Mrs Taylor to find out exactly what she and the maths department want from the computer program.

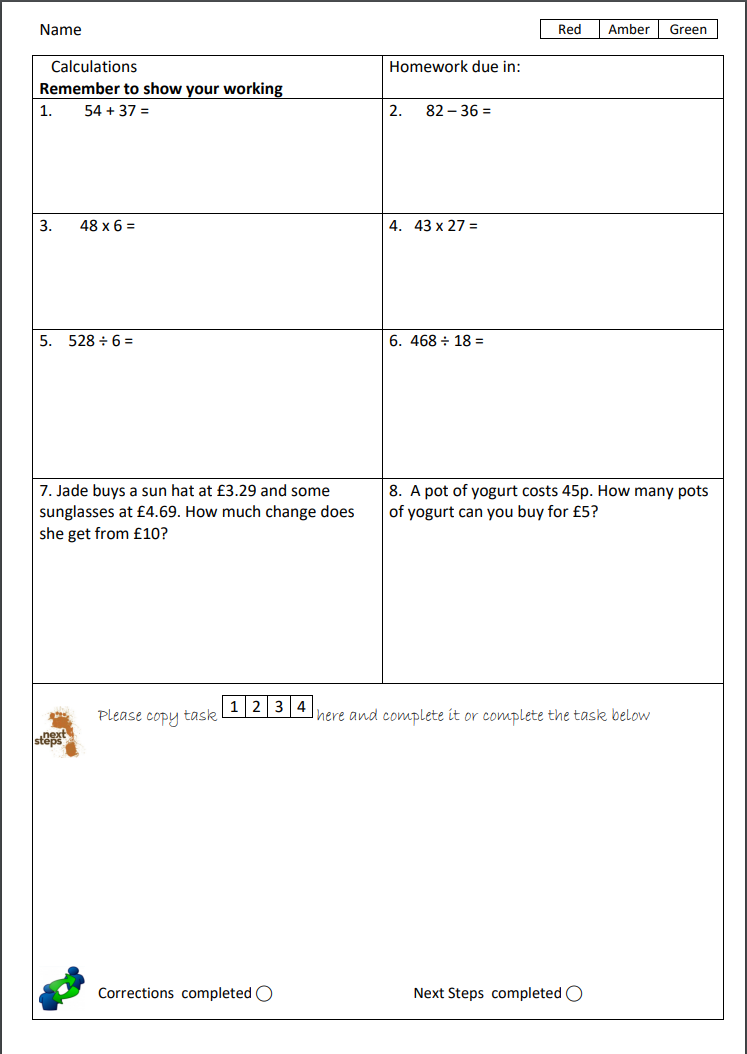
## The current system

The current system revolves around setting paper homeworks for students, as mentioned above. This involves many tasks for teachers of the maths department, all of which add up to be incredibly labour-intensive. These tasks are:

* Printing a copy of the homework sheet for each student (which is costly).
* Distributing these sheets to students, which could involve bringing them to the students’ tutors if the students do not have a maths class on the day the homework is assigned.
* Remembering to collect homeworks from students.
* Marking each question of each student’s homework by hand, and calculating their score.
* Hand-writing feedback for each individual student on their homework sheet.
* Recording student’s scores in a physical mark-book, which may involve multiple teachers if a student has multiple class-groups.
* Returning these homework tasks to students so they can see their feedback, which generally happens around 1 week after the students finish the homeworks, once they are all marked.

When multiple homework sheets are set per week for all students in the school, the time taken to complete all of these tasks is incredibly difficult to find.

Here is an example of a paper homework which is used by the maths department:



Printing these paper sheets for over 200 students is costly to the department.

Feedback and suggestions of what questions a student can try next are written here by the teacher who marked the homework.

Each of these questions is hand-marked and checked by a teacher.

With having over 200 students in a year group, the task of marking these paper homeworks, like the one shown above, is incredibly labour-intensive. The need to hand-write feedback for each student, which is generally very repetitive, is also very time consuming. Recording scores in mark-books and keeping records organised is difficult. The time taken to manage these tasks relating to the homeworks takes away from the time which the teachers have to give students additional help if they need it. The system which will be developed will aim to solve these problems by replacing the paper homeworks, and will provide the maths department with far more time for other tasks.

## Interview with the client, Mrs Taylor

*Could you give a brief description of the functionality you would like your maths-tutoring system to have?*

“I would like a piece of software which I can use to replace the paper homeworks that we have been setting to new students in the past. I want students to be able to do short quizzes on a computer, which have been set as assignments in place of these paper homeworks, made up of questions on simple topics such as addition and shapes. I want students to be able to attempt these assignments they are set as many times as they like, and to be able to go back and try old assignments again, if they ever forget any skills. It would be great if students could choose topics of questions to practice in their own time, which they can be quizzed on in a similar way to how I envisage assignments would be like, but automatically generated without any input from myself or other teachers. Another important part of the system is that students should get feedback on assignments they do, and also be given suggestions on how they can practice to improve. I also want a portal, which myself and other teachers can log in to and access, in order to set assignments to students.”

*So, you want the teacher-part and student-part of the system to be separated?*

“Yes, I do, since teachers and students will be using completely different parts of the system.”

*What sort of technical skill level will the student users have?*

“The students who I intend will use this system will be new year 7 students, so around 11 years old. They will come from various different schools and may well have very poor skills using a computer. Therefore, I would like it if the student parts of the system could be used without a keyboard, as I imagine that some of my students will only have experience with a phone or tablet and will have never used a keyboard before.”

*What hardware do you envisage students using the system with?*

“Most students will use the system with their own school-issued notebook laptops, most of which have touchscreens. Some of the students who struggle the most in class are given a touchscreen Windows tablet. As I said before, I want the students to be able to use the system without a keyboard.”

*What are the major problems with your current system of paper homeworks, that you want to improve with this new system?*

“The paper homeworks take far too long to mark by hand. Writing feedback for each student is time consuming, and very repetitive too. The task of doing all this, for over 200 students, means that students have to wait over a week for their homework to be returned and for them to see their feedback, by which point they will have moved on to a new topic and may not have time to improve if their feedback indicates they need to.”

*What sort of feedback do you want the students to receive?*

“After completing an assignment, I would like students to be told their score of how many questions they got correct and how many they got wrong. I would also like them to be given a suggestion to try an assignment again if their score was low, which we would have told them with the old paper homework system. It would be fantastic if they could be given this feedback instantly after completing an assignment, which would be a massive improvement over the 1-week turnover for feedback that we have with paper homeworks. Additionally, I would like a student to be able to easily see which question topics they struggle most with, and which they are best at, so that they know what to practice to improve their maths skills.”

*How would you like the user interface of the student-side of the program to look?*

“Since the students may not be very technically skilled, I would like the user interface to be simple, and important parts to be labelled clearly. Anything, such as feedback graphs, with any complexity to it, needs to have something explaining how to use it. To make it friendly for young students, I would like there to be some simple colour within the design, and there to be no text-dense parts to the program. Everything also needs to be big enough to easily use with a touchscreen.”

*What functionality would you like the teacher-part of the system to have, apart from just setting assignments?*

“While teachers are setting assignments to students, the due date must be able to be set, and a note left for students with each assignment. These assignments must be able to contain any number of questions. Teachers should be able to create the questions for assignments too, which will be automatically marked when students do them. I would also like teachers to be able to view assignments they have set in the past, and see how well students have done on them, including the number of times a student has attempted any given assignment. Some quick and easy-to-digest graphed statistics on each assignment a teacher has set, such as a breakdown of how many students have completed an assignment and how many haven’t, would be very useful too. Finally, I would like myself and other teachers to be able to view details on students, and some simple statistics on each student, such as the assignments they have completed and how many questions they have tried, so we can see if any students are falling behind.

*How would you like the user interface of the teacher-side of the program to look? Any different to the student-side?*

“Myself and other teachers will be using the system with standard desktop PCs. Therefore, none of the big buttons and large text is needed for the reasons it was needed for students. I would like the user interface for teachers to look similar to other programs I have used on my Windows PC, so that it will be easy and familiar for myself and other teachers to use.”

*How would you like teachers to be able to log in to the system? Differently to students?*

“I would like there to be a different log in screen for teachers and students. There needs to be far greater security when teachers log in, compared to the student log in, since I can’t have anyone getting access to a teacher’s account, where confidential student details could be seen. I would like myself, and other teachers, to be able to log in using a username and password. I want my username to be easy to remember, and also want to be able to change my password when I am logged in, in case anyone finds it out.”

This interview, along with the preliminary discussion within the previous ‘background to the problem’ and ‘current system’ sections, has been used to create a specification for the system I will develop. This specification follows on the next page.

## Specification part I: Specification for student-side of the program

1. Students must be able to log in in a simple, fast and easy manner, simply by writing their initials using a mouse or touchscreen.
   1. Security of log-in is not important, since there is no incentive for a student to want to log in as another student. Therefore, if there are multiple students with the same initials, the system can ask the student to confirm which student out of the multiple matches they are.
2. Students must be able to view and complete assignments which have been set to them by a teacher.
   1. Questions in an assignment which the student has chosen to do should be displayed to the student one-by-one.
   2. A student must be able to draw/hand-write their answer using a mouse or touch-screen, rather than typing in an answer with a keyboard.
   3. The student’s answer to a question must be immediately automatically marked, by comparison to a stored correct answer.
   4. The student must then receive instant feedback on whether they got the question correct or incorrect.
   5. The student must be given instant simple feedback after they complete an assignment.
      1. They must be informed how many questions they got wrong and how many they got correct.
      2. They must also be given some form of additional feedback based on their score which is easy for a young user to understand, such as a happy or sad face or a short sentence congratulating them.
   6. The student’s result in an assignment should be stored for future analytics and statistics.
3. Students should be able to practice maths topics of their choosing, in their own time, without any input from or tasks being set by a teacher.
   1. Quizzes on any combination of chosen topics must be automatically generated for the student to complete.
   2. The student must be able to choose how many questions they would like their quiz to contain.
   3. The student must be given instant simple feedback after they complete a quiz, the same as detailed in specification I, part 2e.
4. Students must be able to view their past attempted assignments, including their score and date they made the attempt.
   1. These must be displayed, listed, to the student in a large and clear format, which is appropriate for high-DPI laptop displays and touchscreens.
   2. Students must be able to attempt past assignments again with the click of a button.
5. Students must be given automatically-generated feedback on their progress in learning question topics.
   1. This should adapt as a student progresses with completing more assignments and questions, to indicate to a student what they can do to improve further.
   2. The feedback should be given in a manner that is easy for a young student user to understand.
6. Students must be able to interact with the program without the use of a keyboard for any purpose. This will make the program far simpler to interact with for the young students which are likely to be using it.
7. There should be no complex design features in the user-interface of the student-side of the program, in order to make it easy to use for the young end-users.
   1. Buttons should be large and clear, in order to be appropriate for use with a trackpad or touch-screen and easy for young student end-users to interact with.
   2. There should be no tabbed interfaces or menus which could be difficult to navigate and understand - instead, everything should be accessible from a single ‘home’ panel.
   3. Any displayed data, such as graphs, should have text explaining their use to the user.
8. All of the details of students, such as their name, gender, class-group, and teacher should be stored in a database. All data regarding a student’s assignments which they have been set, and a history of their past assignments and question attempts, should be stored in the same, single database also.
   1. All data should be stored in a single database, which can be stored on a shared drive accessible everywhere in the school, to simplify data storage so the program can run on any student laptop whilst connected to the school network.

## Specification part II: Specification for teacher-side of the program

1. Teachers should be able to log in with a username and password, using a simple and easy-to-use user interface.
   1. The security of the teacher-side of the program is critically important (especially compared to that of the student-side), since access to a teacher account would allow someone to view and change confidential student details. Therefore, security measures should be taken to ensure no unauthorised access to the teacher-side of the program is possible (without extenuating effort, as no measure can possibly be 100% secure).
   2. Teachers must be able to change the password they use to log in, once logged in to the system, so that they can change it if accidentally shared, to ensure system security.
   3. Usernames must be easy to remember.
2. Teachers must be able to easily and quickly create and set assignments to students.
   1. An assignment must be able to be set to any number of whole class-groups, and any number of students independently of a class-group.
   2. Teachers must be able to choose what questions are in their assignment. This will be done by creating and choosing a quiz to set for the assignment, detailed in specification II, part 3.
   3. Teachers must be able to specify a deadline for assignments.
   4. Teachers must be able to leave a note for students when setting an assignment.
3. Teachers must be able to create a quiz, which can be set as an assignment to students.
   1. Teachers must be able to create and add questions to a quiz, by inputting a question and its answer.
   2. A quiz must be able to contain any number of questions (at least one), made up of any combination of question topics.
   3. Teachers must be able to name their quiz, in order to easily find it to set for an assignment.
   4. Quizzes must have the option to appear to students in the order teachers create them, or in a random order.
4. Teachers must be able to view the assignments they have set to class-groups and students.
   1. Details on whether a student has completed an assignment and or not, along with a student’s best score and number of attempts, must be shown for all students who an assignment has been set to.
   2. It would be desirable for statistics on assignments to be displayed in graphs, such as a chart of completion of each assignment, to be fast and easy for teachers to use and understand.
5. Teachers must be able to view statistics about their students that they teach, in order to identify students that may be struggling. A history of assignments that a student has completed should be able to be viewed by a teacher too.
6. The user interfaces of the teacher-side of the program should follow same design characteristics that most other Windows programs have, in order for teachers to be able to learn to use the program quickly and the UI seem familiar and easy to use. This will also keep the UI clean and as minimalistic as possible. These characteristics are as follows:
   1. UI panels should have white or grey backgrounds.
   2. Most buttons should be the standard Windows grey pop-up buttons, unless serving a purpose that would be aided by straying from this.
   3. Data input boxes should comprise only of text-boxes, combo-boxes, date-time pickers and numeric-up-down boxes so that teachers will already be familiar with them from other programs.
   4. The UI should be designed to be easy and fast to use, considering one of the key purposes of this system is to be faster than the old paper system.
7. All key parts of the user interface should be within one Windows Form, rather than spread across multiple forms, to make switching between parts of the UI more fluid than if a new window was to open, which could potentially be in a different part of the screen to that which the user was working in previously.
   1. The UI must therefore be kept organised by using a tabbed interface, or a similar approach using a menu and panels to separate key parts of the program intuitively.

In order for students to be able to fully interact with the maths tutoring program without the use of a keyboard at all (and therefore just a mouse or touchscreen), students must answer questions by hand-drawing their answer (be it a number, letter, symbol or shape), and also log in to the system by drawing their initials (as specified above), rather than typing a username.

Visual Basic does not have a user-interface control for the user to draw, which will be required for these tasks. Therefore, I will have to develop my own canvas user-interface control which the user can draw in:

## Specification part III: Specification for the canvas user-interface control

1. The user must be able to draw in a ‘canvas’ using a mouse or a touchscreen, to be able to draw characters such as letters, numerical digits, and other symbols (e.g. +, !, ?, etc.). The user must also be able to draw simple diagrams of objects such as shapes.
2. The line drawn by a canvas should be smooth and clear:
   1. Corners and ends of lines should be rounded, i.e. a drawn line should have a circular end, rather than an abrupt square end.
   2. Antialiasing should be applied to drawn lines to smooth diagonal sections. This is necessary due to the nature of the majority of computer displays having a vertical/horizontal matrix of pixels causing diagonal lines to essentially be made up of a line stepped vertically and horizontally, causing a jagged appearance without smoothing.
   3. There should be no gaps in a line that is intended to be continuous, and no visual artefacts in the line.
3. Only one colour needs to be available for the user to draw with - in order to be appropriate for all uses, this colour should be black.
4. The thickness of drawn lines needs to be easily changeable on a canvas-by-canvas basis.
   1. For example, drawing letters of initials to log in may need a thinner line to have distinguished detail in drawings, compared to drawing a simple shape.
5. There must be functionality for a user to clear any drawing on a canvas that they have made, i.e. reset to a blank white canvas.
6. Similarly, any specific canvas needs to be able to be cleared simply by calling a function within an algorithm. For example, if a user has answered one question, the canvas needs to be able to be cleared automatically for the user to then answer another question using it.
7. The canvas control should be able to be generated in any form of the program, wherever it may need to be used, by calling a function. Each canvas needs to have these following properties able to be set when created, in order to serve different purposes:
   1. Width and height of canvas.
   2. Position of canvas within the form or panel it is being created in.
   3. Whether the canvas has a border line to show the edges of the canvas, or not.
8. The drawn contents of a canvas must be able to be retrieved in bitmap format, for further use or processing.

Automatically marking a student’s answer to any question is made into a significantly harder task when the answer is given as hand-written characters in a bitmap. The handwritten initials of a student, used to log in, will also need to be transformed to be usable and processable by a computer.

To complete these tasks, an artificial intelligence method will need to be used to recognise what character, symbol, or else, has been drawn, and assign a string label to that image which a computer is capable of understanding and using. The specification for this artificial intelligence character recognition algorithm is as follows:

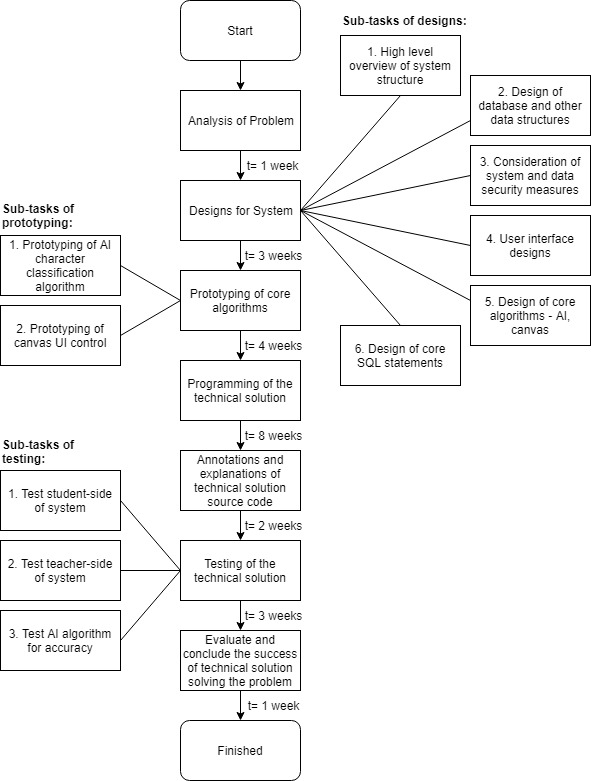
## Specification part IV: Specification for the artificial intelligence character recognition engine

1. The AI method for character recognition must be callable as a single function from anywhere in the end-user side of program, passing the necessary inputs (such as a bitmap) into the function and returning the output label of the bitmap passed in.
   1. Essentially only one line of code should be required to be written to get a string (this string being the label assigned to the contents of the bitmap image) output, from a bitmap image input to a function.
   2. The artificial intelligence method should be in the background, ‘invisible’ to the end-user, without any control by student users being necessary for its operation.
2. The algorithm must be able to classify hand-drawn digits (0-9), upper-case alphabetic characters (A-Z), symbols (e.g. +, -, etc.), and shapes from the standardised processed input image.
   1. A label must be assigned to said image, which is a string representing what the AI method ‘thinks’ is the most likely character that is drawn in the image.
   2. This label must be returned from a function as a string type variable.
3. A bitmap image containing a single drawn character must be passed as a parameter into the function in bitmap format.
   1. This image must be able to be:
      1. Any dimensions
      2. Either black and white or any colour or combination of colours
   2. ... and still be able to be recognised and have a label assigned to it by the AI method.
4. Another input parameter for the AI character classification function will be the type of character that the input image contains, i.e. is it a letter (A-Z), a number (0-9), a symbol, or a shape. This will improve the accuracy of the character recognition. This information will always be available, since it is easily known what sort of answer a question will have (e.g. 9+4 will have a numerical answer), and other uses of the character classification method, such as logging in, will always be classifying letters.
5. The bitmap image which is passed to the AI character recognition engine function must be cropped to remove any white-space surrounding the character.
6. The bitmap image must be compressed to a small standardised (every input image the same) size, such as 20x20 pixels, once white-space has been cropped out, to improve the speed of the classification algorithm and reduce the file-size of the data that would need to be stored in a dataset.
7. Once a bitmap image containing a single character has been passed to the character recognition system to be classified, the label for the image should be returned in a short period of time (within 0.5 seconds) – the code must be efficient for this specification to be met.
   1. This will ensure that the end-user does not need to wait for any significant amount of time whilst the AI engine is operating.
8. The AI function should finish executing within the above time specification on low-performance hardware (such as a dual-core 15-watt laptop CPU), as it must operate well on laptops and tablets which students are likely to be using, which will usually have said low-performance hardware.
9. There must be functionality to be able to train the AI to recognise individual additional characters or symbols.
   1. This must be a reasonably fast process, (i.e. must not require in the order of magnitude of 100+ images of a character to train the AI) and be able to recognise a new character with reasonable accuracy from just 5-10 input example images.
10. The training of the artificial intelligence method for exhaustive types of input (i.e. characters A-Z and numbers 0-9 where there is a limited number of possible labels, since there are no more characters than A-Z and no more number digits than 0-9) does not need to complete quickly or be optimally efficient, as this only needs to be done once (a dataset can be created with a high-performance machine), prior to the program being used by end-users on low-performance computers.
11. The AI character recognition algorithm must label characters correctly with an accuracy of 90% (i.e. characters are labelled correctly 90% of the time).
    1. This chosen accuracy of 90% is realistic for an algorithm which will be trained with a limited amount of data (see specification IV, part 9a above), but also will be aided by limiting the number of possible labels for an input image by knowing the type of character it is classifying (see specification IV, part 3).
    2. This accuracy of 90% is also high enough that the program will be a good experience to use, for example students will not regularly get false-positives when they actually got a question wrong, or false-negatives when they actually got a question correct.

To meet these criteria, a nearest-neighbour AI method should be used for character recognition. This is an appropriate AI method to use, because it can be trained using a small dataset, and can provide the accuracy specified. There are other AI methods, such as using a neural network, that could provide higher accuracy, but the gains are small and are outweighed by the significantly decreased efficiency of those algorithms.

## Time plan for the project

I have created a flow-chart, breaking down the tasks involved in this project, and the time period I expect to complete them within. The time which is expected to be required to transition between each stage of the project has been recorded as a label beside the connections between each stage of the flow-chart below.



## Limitations of the system

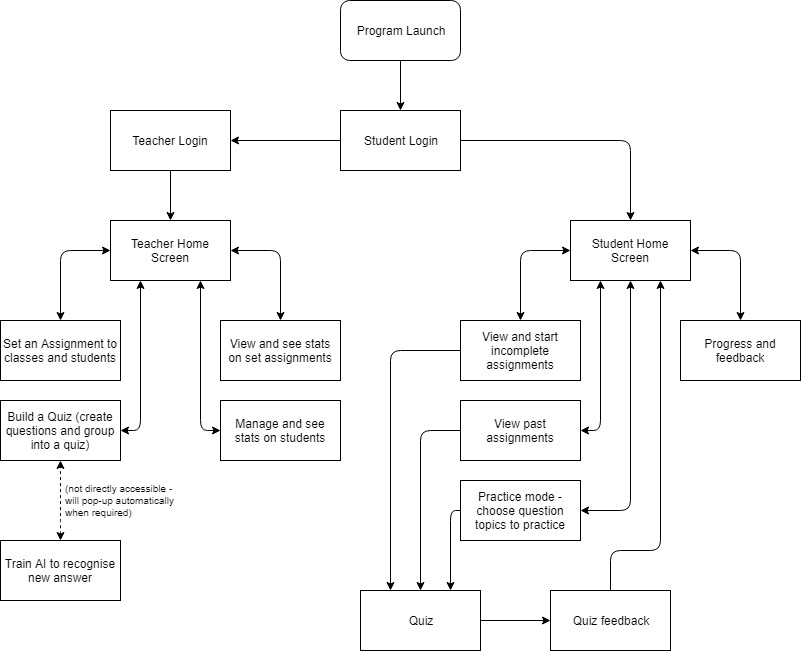
The system, as set out by the specification, will be a fairly complete solution to the problem of not having enough time to mark paper homeworks and analyse their results, and also not having enough time to satisfactorily teach all students the fundamentals of maths that they are missing, as it should provide a solution to these two main criteria by providing alternative digital homeworks and a means for students to teach themselves additional maths fundamentals to get them to a satisfactory level of understanding. However, there is only a short period of time to develop this project, and there is no time or availability of students to heavily use the program over a long period of time.

This lack of availability of students will prevent extensive testing of how the program will perform once the database has become significantly large being possible to do. To counter this problem of performance once the amount of data being processed becomes very large, I will focus on ensuring that data is only read from the database when absolutely necessary, i.e. if a student wants to do an assignment, only the questions of that assignment will be loaded from the database, rather than all possible questions available being loaded (as this would be inefficient, and operation could become incredibly slow when a large dataset of questions develops over time).

# Designs

## High-level overview of the program’s structure

The following is a flowchart showing a high-level overview of the structure program:



Each of the above processes will be separated by being contained within their own panel on a windows form, with only one panel visible to the user at once to simplify the use of the program. The one exception to this is that the ‘view and start incomplete assignments’ process, will be integrated into the student home screen. The student home screen will be the first panel that students see when they have successfully logged into the system. Since the main purpose of the system, for students, is to do assignments that have been set for them by a teacher, it would be logical to immediately display their assignments to them, within the home screen, in order to make the system easier and more intuitive to use.

## Proposed data structures

There will be multiple instances of data storage in the program. The data storage can be split into two categories: firstly, training data for the artificial intelligence character recognition algorithm, and secondly, any data relating to student details, teacher details, questions, assignments and etc.

### Training data for the artificial intelligence character classification algorithm:

Text files will be used to store the training data for the artificial intelligence method. Text files have been chosen over a database for this for their fast access speeds (compared to databases which are more complex to access), because despite there being a large amount of data, all of the data will be regarding essentially the same thing, and therefore will not need to be searched through (which would be more difficult and less efficient with a text file than a database).

There will be a large amount of data because the AI method will need to be trained with many instances of each character, symbol, or other drawing that it will be able to recognise. The images it is trained with will involve the image being processed and a string being created from it, this string being stored in a text file with its label, for comparison against for classifying images at a later time (this process of how the AI will be trained and will function will be described further later on).

The data will all be regarding the same thing (hence why text files are appropriate) because the only data being stored will be hundreds of different strings and labels, generated from training images - no other data will be stored in these text files. By specification IV, part 4, one of the input parameters for the AI character classification function will be the type of character that the input image contains (i.e. number, letter, shape, etc.). So, if for example, the AI is told that the input is a numerical digit, it would only need to compare the input against training data for numbers, not letters and other symbols as well. Therefore, the training data for each type of input will be stored in a separate text file.

There will be a separate text file for training data on each of these input types:

* Letters A-Z
* Numerical digits 0-9, and a . (decimal place)
* Other symbols, e.g. ! ? - +
* Shapes

### Other data, such as student details and assignments:

All other data will be stored in a database. A database has been chosen due to the wide variety of data that will be stored. Additionally, complex processing and retrieval of the data will be necessary for tasks such as creating statistics on a student’s progress in questions, which SQL will be incredibly useful for. Text files could not provide the structure, organisation, and ability to search through are large quantity of varied data, that a database does offer.

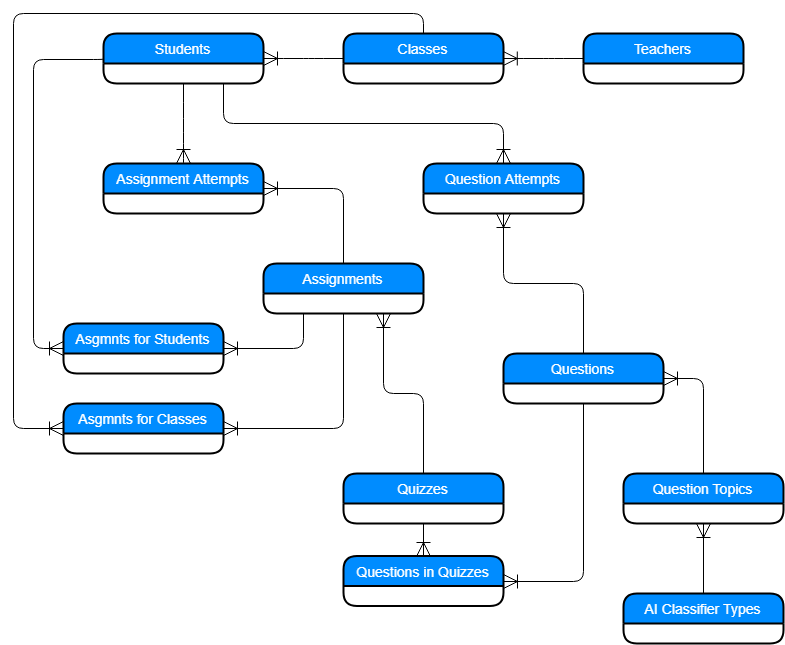
The following data is anticipated to be needed to be stored in this database:

* Student details, including first and last name, gender, DOB, their class-group and additional information for ensuring students are not being disadvantaged progress-wise by their circumstances, such as their SEN support status and pupil premium status.
* Teacher details, including their title, first and last name.
* Usernames and passwords for teachers to log in to the system with.
* Association of class-groups with their teacher.
* Assignments which have been created by teachers, and details of which class-groups and students assignments have been set to, set and due dates, any attached notes left by teachers for students, and which questions an assignment comprises of.
* A record of students’ attempts on assignments that have been set, including their score and date of completion.
* Questions, including their question text, answer, and topic.
* Details on question topics, such as their name, and details on how the AI will check answers of questions of said topic (such as the type of answer, i.e. letters, digits, shape, which will be expected, and therefore which training data text file should be used when classifying answers).
* A record of students’ attempts on individual questions, for use for creating statistics and feedback.

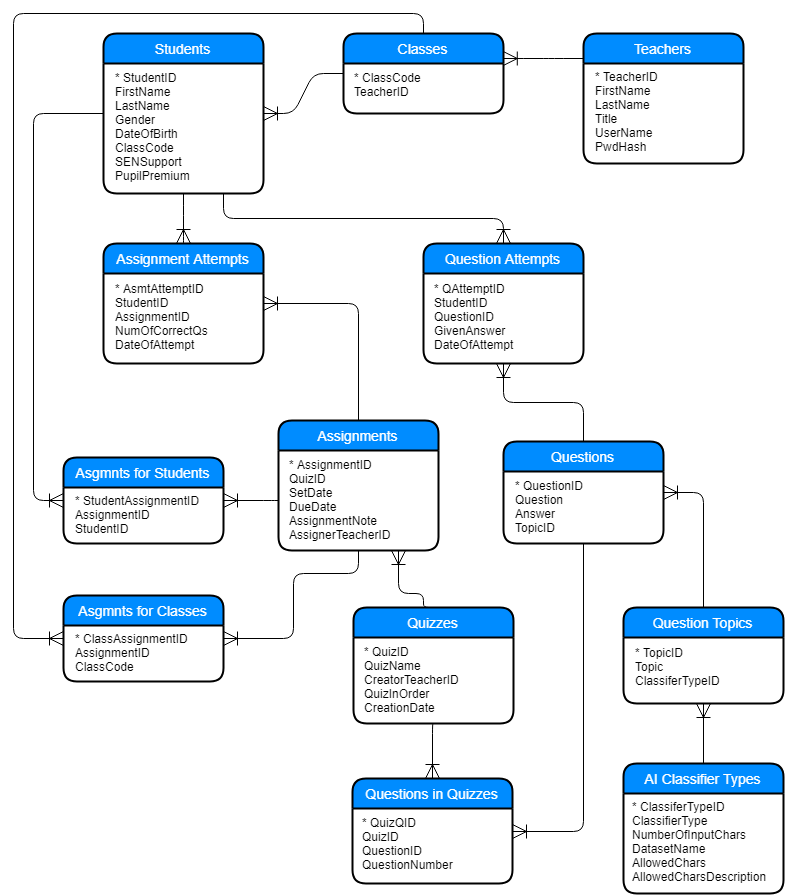
## Database design

The variety of data which is anticipated to be stored in the database will need to be stored across multiple normalised tables, in order to reduce data redundancy (the repetition of data fields) and therefore reduce the storage space taken for the database.

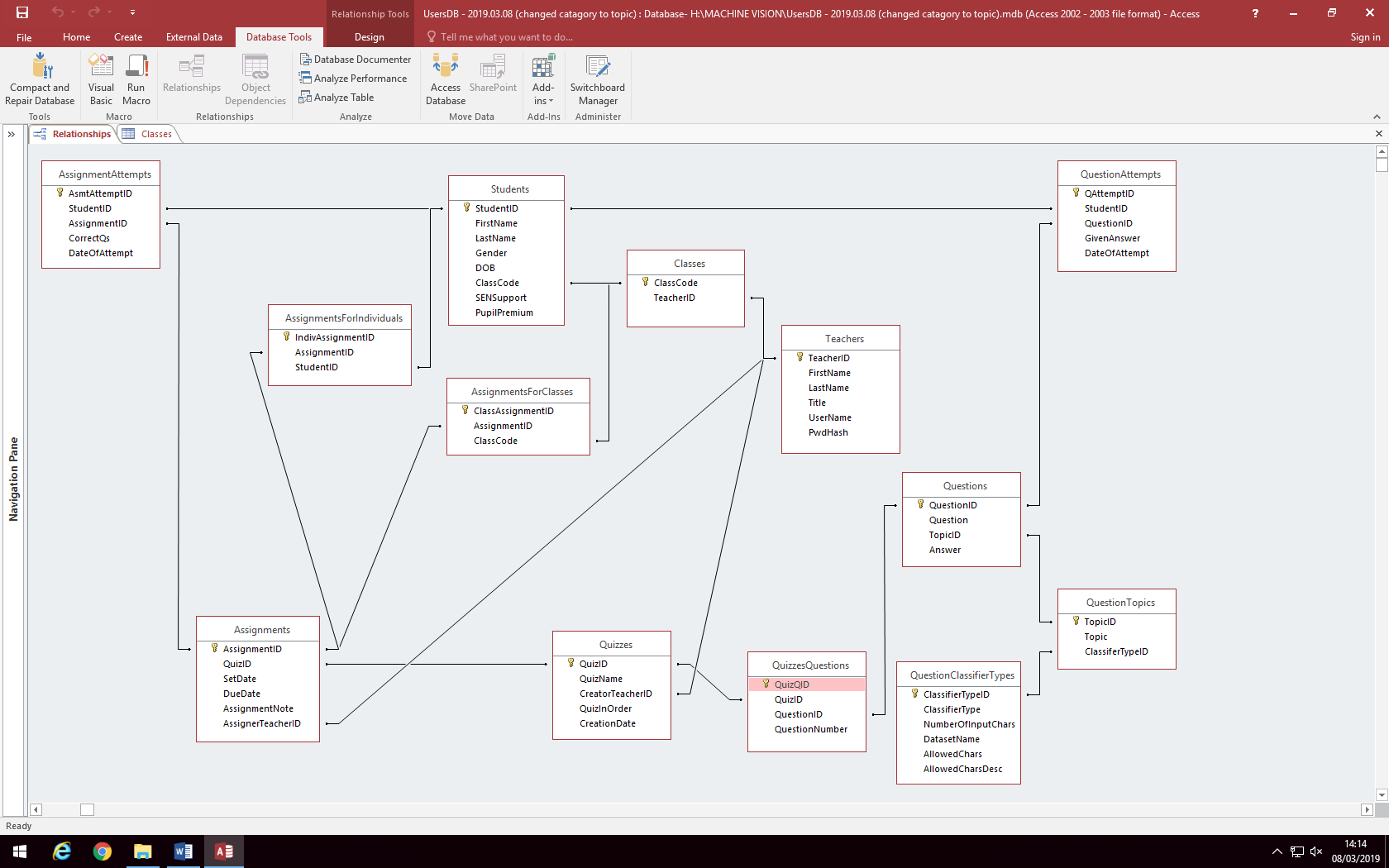
### Database entity relationship diagram:



### Database schema:



From the above database schema, I have created the tables and fields detailed in Microsoft Access. The view of relationships between fields within the database from Access is on the next page:



### Data dictionary:

Here is a data dictionary for all of the fields of each table, which was set out in the above database schema.

**Table:** Students  
The students table is used for storing details of students.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| StudentID | Integer (AutoNumber) | 16 | Primary key |
| FirstName | String | 16 | Stores first name of student |
| LastName | String | 16 | Stores last name of student |
| Gender | Char | 1 | Stores gender of student (M/F) |
| DOB | Date/Time |  | Stores student’s date of birth, for statistics |
| ClassCode | String | 3 | Class-code for student’s class-group. |
| SENSupport | Boolean |  | Identifies whether a student has SEN support or not, for equality statistics. |
| PupilPremium | Boolean |  | Identifies whether a student has pupil premium support or not, for equality statistics. |

**Table:** Teachers  
The teachers table is used for storing details of teachers, similarly to the students table.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| TeacherID | Integer (AutoNumber) | 16 | Primary key |
| FirstName | String | 16 | Stores first name of teacher |
| LastName | String | 16 | Stores last name of teacher |
| Title | String | 4 | Stores title (mr, mrs, miss, etc.) of teacher |
| UserName | String | 32 | Stores the username which a teacher will use to log in to the system. |
| PwdHash | String | 128 | Stores the encrypted hash of a teacher’s password, which is used to log in. |

**Table:** Classes  
This table is used to store a list of class-codes for class-groups, and which teacher teaches each class-group.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| ClassCode | String | 3 | Primary key. No need for additional autonumber primary key, as all classcodes (e.g. 13S, 7B, etc.) are unique. |
| TeacherID | Integer | 16 | ID of teacher who teaches a class-group |

**Table:** Questions  
This table is used to store details on questions.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| QuestionID | Integer (AutoNumber) | 16 | Primary key |
| Question | String | 256 | Stores the text of question, which will be displayed to a student when they are answering the question. |
| Answer | String | 32 | Stores the correct answer to a question. |
| TopicID | Integer | 16 | The ID of the topic which a question belongs to. |

**Table:** QuestionTopics  
This table is used to store details on question topics.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| TopicID | Integer (AutoNumber) | 16 | Primary key |
| Topic | String | 32 | Name of a question topic, e.g. ‘shapes’ |
| ClassifierTypeID | Integer | 16 | ID for the type of classifier which the AI character recognition algorithm should use when answering questions of this topic, e.g. alphabetical, numerical, etc. |

**Table:** QuestionClassifierTypes  
This table stores details on how the AI will classify different types of answers (i.e. numerical answers, alphabetical answers, drawn shape answers, etc.).

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| ClassiferTypeID | Integer (AutoNumber) | 16 | Primary key |
| ClassifierType | String | 32 | Name of the classifier type which the AI character recognition algorithm should use when answering questions of this topic, e.g. alphabetical, numerical, symbolic, etc. |
| NumberOfInputChars | Integer | 2 | Number of input characters which are expected from an answer to a question with a topic with this classifier type. Would be ‘0’ if could be any number of characters, e.g. an addition question could have an answer with any number of characters. |
| DatasetName | String | 256 | File-path and file-name of the AI dataset text-file for use when classifying answers of topics with this classifier type. |
| AllowedChars | String | 64 | Characters which are allowed for an answer. E.g. for ClassiferType ‘numerical’, AllowedChars would be ‘0123456789’. |
| AllowedCharsDesc | String | 128 | Written description of the characters that are allowed for an answer to a question; displayed to a teacher who is creating questions. |

**Table:** QuestionAttempts  
This table stores details of every attempt a student makes on a question, which can be processed for statistics and feedback.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| QAttemptID | Integer (AutoNumber) | 16 | Primary key |
| StudentID | Integer | 16 | ID of student who attempted a particular question. |
| QuestionID | Integer | 16 | ID of the question which was attempted. |
| GivenAnswer | String | 32 | Answer which the student gave for a question. This can be compared to the actual answer to a question in the ‘Questions’ table, to determine whether the student’s attempt was correct or not. |
| DateOfAttempt | Date/Time |  | Date and time which the student attempted a question. |

**Table:** Quizzes  
This table stores details of quizzes, which are combinations of questions that can be set as an assignment.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| QuizID | Integer (AutoNumber) | 16 | Primary key |
| QuizName | String | 64 | Stores the name of a quiz. |
| CreatorTeacherID | Integer | 16 | ID of the teacher who created a quiz. |
| QuizInOrder | Boolean |  | Determines whether a quiz should be displayed in a random order or in a specified order. |
| CreationDate | Date/Time |  | Stores the data which a quiz was created. |

**Table:** QuizzesQuestions  
This table stores details of which questions quizzes have. There is an entry in this table for each question a quiz has, associating the QuestionID and QuizID.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| QuizQID | Integer (AutoNumber) | 16 | Primary key |
| QuizID | Integer | 16 | ID of the quiz which contains an associated question. |
| QuestionID | Integer | 16 | ID of the question which will be in an associated quiz. |
| QuestionNumber | Integer | 4 | Stores the question number for a quiz – the order that questions should appear to students in a quiz. |

**Table:** Assignments  
This table stores details of assignments which have been set, such as the ID of the quiz which is assigned, and set and due dates.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| AssignmentID | Integer (AutoNumber) | 16 | Primary key |
| QuizID | Integer | 16 | Stores the ID of the quiz, which contains a set of questions, which is assigned. |
| SetDate | Date/Time |  | Stores the date that an assignment is set to appear to students. |
| DueDate | Date/Time | 16 | Stores the date of the deadline of an assignment. |
| AssignmentNote | String | 256 | Stores an optional note which teachers can leave for students when setting an assignment. |
| AssignerTeacherID | Integer | 16 | Stores the ID of the teacher who set an assignment. |

**Table:** AssignmentsForClasses  
This table stores which class-groups an assignment has been set to. There will be a row in this table for each class-group an assignment has been set to, associating the AssignmentID and ClassCode.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| ClassAssignmentID | Integer (AutoNumber) | 16 | Primary key |
| AssignmentID | Integer | 16 | Stores the ID of the assignment which has been set to a class-group. |
| ClassCode | String | 3 | Stores the class-code of the class-group which an assignment has been set to. |

**Table:** AssignmentsForIndividuals  
This table stores which students, independent of a class-group as a whole, that an assignment has been set to. There will be a row in this table for each student an assignment has been set to, associating the AssignmentID and StudentID.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| IndivAssignmentID | Integer (AutoNumber) | 16 | Primary key |
| AssignmentID | Integer | 16 | Stores the ID of the assignment which has been set to an individual student. |
| ClassCode | String | 3 | Stores the ID of the student which an assignment has been set to. |

**Table:** AssignmentAttempts  
This table stores details of every attempt a student makes on an assignment, which can be processed for statistics and feedback.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Size (chars) | Description of Purpose |
| AsmtAttemptID | Integer (AutoNumber) | 16 | Primary key |
| StudentID | Integer | 16 | Stores the ID of a student who has made an attempt on an assignment. |
| AssignmentID | Integer | 16 | Stores the ID of the assignment that was attempted. |
| CorrectQs | Integer | 4 | Stores the number of questions which a student got correct in an assignment. This can be compared to a count of the number of questions in a quiz from the ‘QuizzesQuestions’ table, to determine the percentage score a student got in an assignment. |
| DateOfAttempt | Date/Time |  | Stores the date which a student completed an attempt on an assignment. |

## System and data security

### Database and teacher-side of program:

As stated in specification II, part 1a, the security of the teacher-side of the program is very important. Therefore, multiple measures will be taken to ensure the security of the data, both within the database and within the program – both the ability to access the data, and the data itself, must be secured, as one weak point in security results in other security measures being far less effective.

Access to the teacher side of the program will be secured using username and password access. Each teacher will have their own username and password to access the system. No access will be possible without a correctly inputted password. To prevent passwords being obtained by potentially malicious users by watching a teacher log in, the password input-box will be obfuscated, i.e. the input will appear as ‘\*\*\*\*\*\*\*\*’, rather than ‘password’, for example.

A teacher’s inputted username and password will be checked against the database for a match, when they attempt a log in. If there is no match, i.e. the username or password is incorrect, the user will only be informed that their ‘username or password is incorrect’ – they will not be told which of the two, or if both, are incorrect. This adds an extra layer of security, as users attempting unauthorised access will not know if they have a correct username or password, even if one is correct, which will slow brute-force attacks on the log in.

Since teachers’ passwords are stored in the database, even with the security measures to prevent unauthorised access within the program, if a malicious user were to gain access to the database, they could obtain stored passwords from there. In order to prevent this, passwords in the database will be encrypted, instead of being stored as plaintext. The SHA1 hashing algorithm will be used to encrypt passwords, creating a hash, and that hash will be stored in the database, instead of the plaintext password. For example, instead of the plain-text password ‘password123’ being stored in the database, a matching SHA1 hash will be stored ‘12f6b5bd04e581f6b59f21ae7e0450188b8d’ (n.b. this is a hash for a different password and not the matching hash). This prevents users with access to the database from reading passwords in plain-text and then being able to use them to log in to the system.

However, despite storing a hash rather than a plain-text password, a user with access to the database could still potentially obtain a plain-text password from a hash. If this password hash was for a user using a common password (e.g. ‘password123’), a malicious user could look-up the hash in an online hash-dictionary, and potentially obtain the matching plain-text password. An additional security measure will be added to prevent this, which is salting the password hashes with a fixed salt (fixed - the same salt for all passwords). A password salt is set random string of characters appended to a password before hashing, to add complexity to the plain-text password and therefore prevent a hash from being looked-up in a hash dictionary. Therefore, even if a user had access to the database, and had obtained a password hash from the database, they would not be able to get a plain-text password which could be used to log in to the system, even if the password was very simple.

This high level of security is sufficient to prevent even fairly sophisticated attacks on the system, and will therefore keep access to the teacher-side of the program secure.

### Student-side of program:

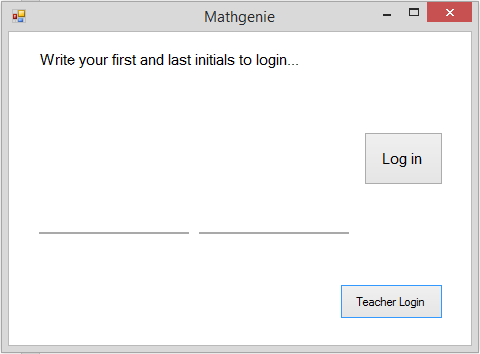
The security of the student-side of the program is far less important, Therefore, logging in with initials and no additional security measures is sufficient, as it would not be an issue, and also not of interest to any students, to access the program logged in as another student – they would have nothing to gain. The advantage of the ease and simplicity of the initials log in system far outweighs the lack of security, when there is no need for security here at all.

Within the student-side of the program, there will be no ability to change the database used for the data-source of the program, and no ability to manually change data in the database (all reads and writes to the database are done ‘behind the scenes’). This will prevent students from being able to misuse the system, such as by changing entries in the database to give the impression that they have completed an assignment, when in fact they haven’t.

## User interface design

### Student login:

The first windows form to appear when the program is launched is the student login screen. The design of this form is a blend of the specification for the design of both the student and teacher parts of the program, as this will appear for both groups of users. This combination of users means the design must be appropriate for the standard computers teachers will be using (so normal Windows design is needed), and also the laptops and tablets with high-DPI screens that students will be using.

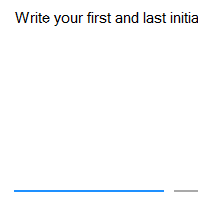


A yes/no button and text asking the user if they are the student found in the database matching the drawn initials will appear here, if a matching user is found once the user has pressed log in. If multiple users match one set of initials, it will ask which of the users the student is, and log in the selected user. This low-security log in method is appropriate, since as explained in the system security section, there is no incentive for a student to access a another student’s user.

Button for teachers to click to log in to the teacher side of the system. The program starts up to the student login rather than the teacher log in, as the system needs to be easier to use for students. This button is smaller and less appropriate for touch-screens than the above student log in button, as students would not need to use this button.

Log in button for students. Large enough to be used with high-DPI displays or touchscreens, but also follows the standard Windows grey colour-palette for design.

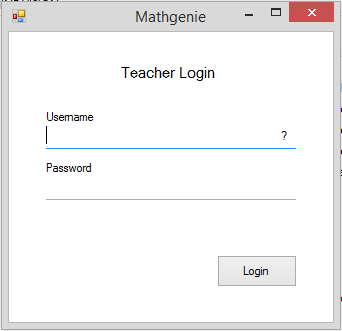
Students write their initials here in order to log in. The bitmap images of the drawings of their initials from here are passed to the AI character classifier to identify a student. When the mouse or a touch-screen stylus is hovered over these areas, the line below illuminates blue to show the user they are within the drawing area.





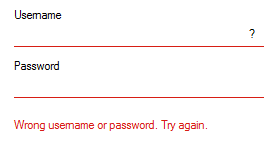
### Teacher login:

This user interface is far more similar to a standard Windows design than the student user-interfaces. It follows a simple and clean design to make it easy to use.



Here are input boxes for a teacher’s username and password. When an input box is selected, the line below is illuminated blue to indicate this. If the ‘?’ is clicked, a message-box appears to remind teachers of the standard format of usernames.

This button, which follows a standard Windows grey design, will log in a teacher with the inputted details.



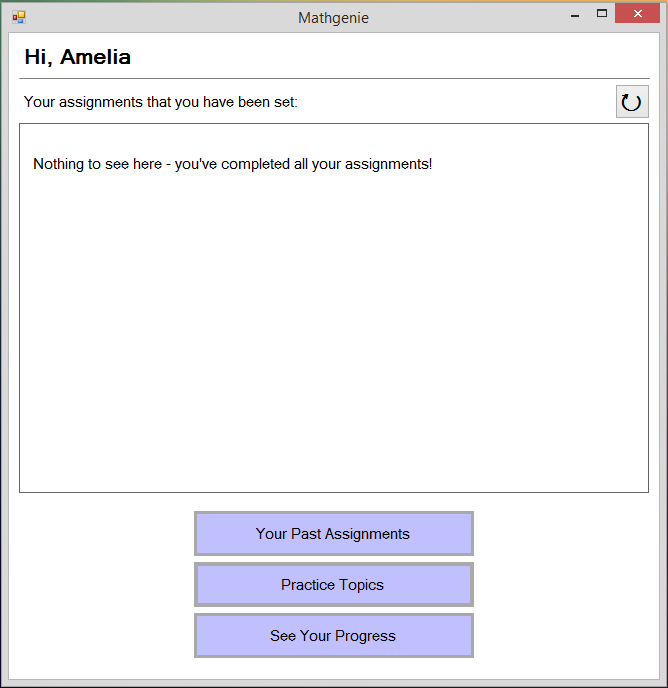
If the username and password inputted do not return a match from the database, a message appears informing the user of this. The lines underneath the input boxes are illuminated red, to provide an additional indication that the user needs to input another username and password.

### Student home page / incomplete assignments:

This section of the student user-interface, and all other following sections, will be separated into their own panels, and all contained within the same windows form. This prevents delays while new windows forms are opening, and instead switching between sections of the program is done by hiding and displaying panels within one windows form.

This button here, when clicked, refreshes the assignments displayed in the box below. To reduce file size of the program, an ASCII character (a rotating arrow) will be used instead of an image, on the refresh button. This is an intuitive and universal symbol for ‘reload’ or ‘refresh’.

The text at the top of the home page is adaptive, and changes based on the logged in student’s first name, to welcome them. The example user, Amelia Capewell, has been used here.



These buttons provide links to all other parts of the student section of the system. When clicked, the ‘student home’ panel here will be hidden, and the panel for the respective button will be made visible. These buttons are large and have large text, which makes them appropriate for use on a touchscreen or high-DPI laptop screen.

This box here will be used to display assignments which a user has been set, but has not yet completed. It will include a dynamically generated button for each assignment, to start the assignment. It will also include details, such as due date, and assigner teacher, of assignments, in dynamically generated labels. Since the Visual Basic list-box control does not have functionality for buttons or multi-line, and is also not appropriate for touch-screens and high-DPI displays due to being compact and small, this will be entirely dynamically generated using labels, buttons, and lines.

### Student’s completed past assignments:

This button, when clicked, will change the visible panel back to the ‘student home’ panel, therefore acting as a back button. It is placed in the top-right corner so as to be easy to find, so students cannot get lost in the user-interface and can always easily find their way back to the home panel.

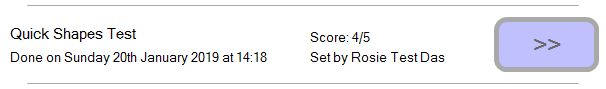
This button here, when clicked, refreshes the assignments displayed in the box below. It is similar to the one explained on the previous page.



This check-box here is used to toggle how the past assignments will be displayed in the box above. One display mode will show each attempt on an assignment individually, and the other display mode will group multiple attempts of the same assignment, showing the best score out of all attempts, and the number of attempts.

This box here will be used to display a history of assignments which a user has completed. It will be dynamically generated, like the one described on the previous page. Details shown for each assignment will include the student’s score on an assignment, and the date the student completed an assignment. Each assignment will have a dynamically generated button, which when clicked will allow the user to attempt an assignment again.

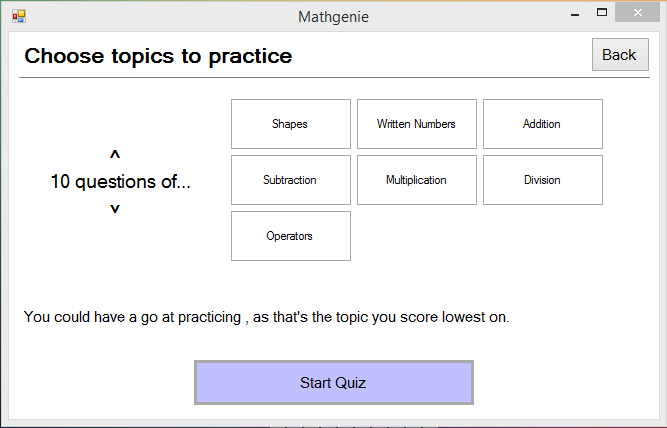
Here is an example of what a past completed assignment would look like in the ‘completed past assignments’ panel:



### Student topic practice mode:

The title text here, and on other sections of the student UI, is large and bold, to make it easy to read on a high-DPI screen.

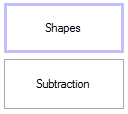
This here will be used to select how many questions the student would like to answer in their custom quiz they are setting up. This works similarly to the VB numeric-up-down control, except it will be custom made from a label, to display the text, and two buttons, to increment the number up or down. This is because the VB control is compact and not suitable for high-DPI screens. This is also simpler than the VB control, making it easier for young student users to understand and use.



These buttons will be dynamically generated, with one button for each topic of questions stored in the system database. They are toggle buttons (i.e. when clicked they toggle between an ‘on’ or ‘off’ state) used to select which topics of questions the student user wants their custom practice quiz to contain. These toggle buttons, which will be a custom control as there is no native VB UI control supporting toggling buttons, are intuitive and easy to use, which is important so that young student users will be able to use the system easily and not become confused. When toggled in the ‘on’ state, to indicate that questions of the type written on the button should be included in the quiz, their border will be highlighted, as shown on the right (where one ‘on’ and one ‘off’ button is shown). The highlight colour of the toggle buttons is the same as the colour of large buttons in the rest of the student-side of the system, for consistency.

Back button, to return to the ‘student home’ panel, making the UI easy to navigate. As explained on a similar button previously.

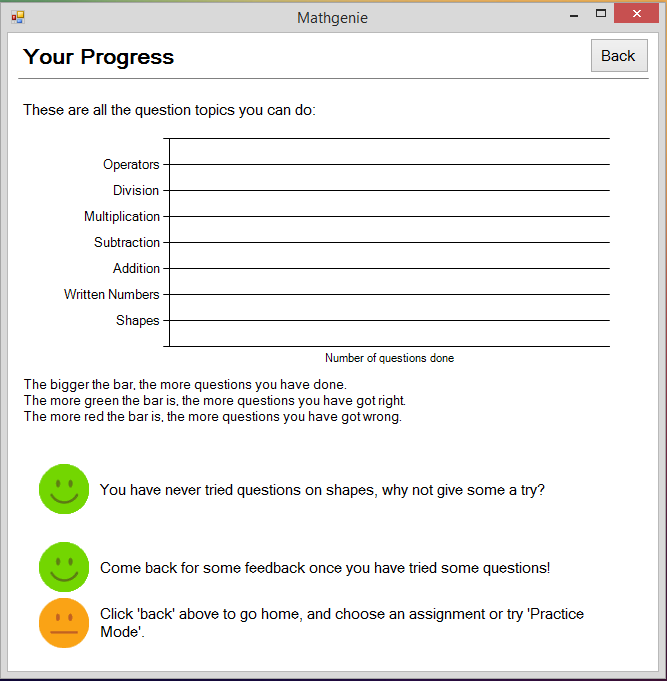
This button, when clicked, will start a quiz made up of the topics the student user has chosen. It is large and clear with large text to make it easy to use with touchscreens.



### Student’s progress and feedback:

This dynamically generated graph here will show a student how many questions they have done in each topic. It will be a stacked bar chart, with a bar for each question topic from the database (shown with some examples on the y axis). Each bar will be split into a green and red section, to show a student how many questions they have got correct and incorrect respectively. A bar chart has been chosen for its simplicity, and therefore it being easy for a young user to understand.

Static text will be displayed here, to explain to a young student user how to understand the graph of their progress in different question topics.



In this space here, dynamically generated customised feedback will appear for students. Each student will see different feedback, which will change over time as a student uses the program, on what topics they are best at, what they could try to do to improve, and what question areas they need to improve on. This feedback will appear in short sentences, one feedback point next to each face. The text will be large to make it easier to read for a young student user.

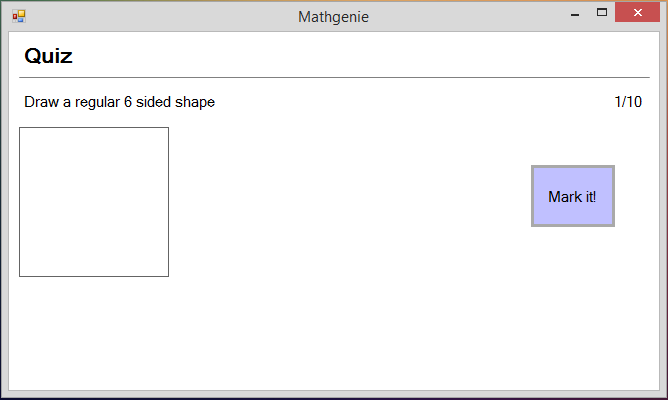
These faces here appear next to customised feedback for a student. They help make the UI more young-user-friendly, by reducing the text-density of the panel.

### Active quiz panel:

The current question number, and the total number of questions in the active quiz, will be shown here. As with the question text, this text is large and clear.

Here, the question text for the current question, obtained from the database, will be displayed. The text is large and clear so a student user can read it easily. The question will also be spoken using text-to-speech. An example question is shown here.

Canvases will be dynamically generated here, with however many that are needed to answer the current question being displayed. One is shown here as an example.



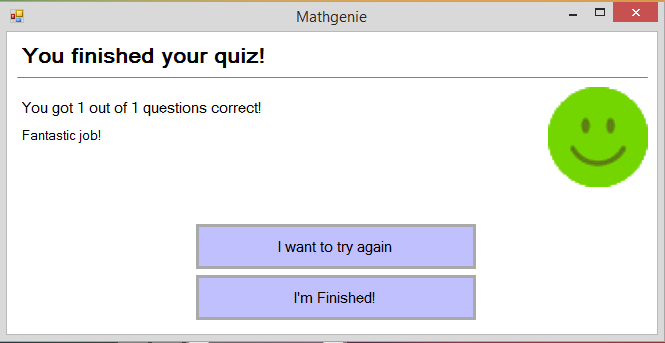
This button, when clicked, will call the artificial intelligence character classification algorithm to check the students drawn answer to the current question. The button is large for use on touchscreens, and follows the same colour theme as the rest of the system.

Immediate feedback, as to whether a student has got a question correct or incorrect, will be shown here, when the ‘mark it’ button is clicked and the AI character classification algorithm is used to check if a user’s answer to a question is correct or not.

### Post-quiz feedback panel:

Instant feedback on how well a student did in their quiz they just completed is shown here. The number of questions they got correct, compared to the total number in the quiz, is shown. Additionally, a short sentence is also shown, which changes depending on the percentage of questions they got correct. The text will be large and clear to make it easy to read on a high-DPI screen. Some example feedback is shown here.

An emotion face is displayed as an extra way for students to interpret feedback. This will help young students understand feedback. This face will vary in colour, from bright green to orange, depending on the percentage of questions a student answered correctly in the quiz they just completed.

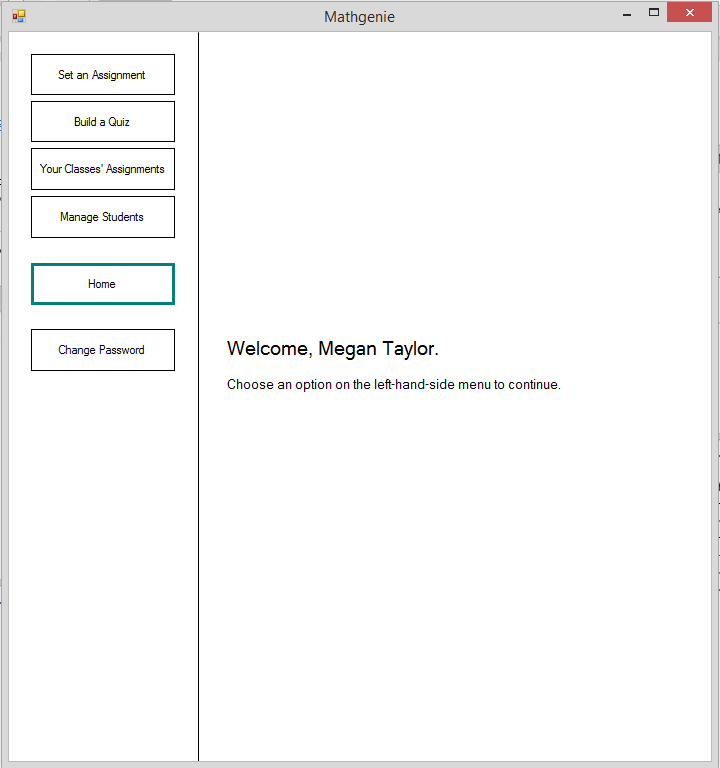


There will be two buttons here. The top button allows a user to repeat the quiz they did again, if they are not satisfied with how many questions they got correct based on their feedback. The lower button sends the user back to the student home screen, so they can do other things within the system without having to restart the program. The buttons are large, to be easy to use on a touchscreen, and conform with the system colour scheme.

### Teacher home page:

Similarly to the student-side of the system, this section of the teacher user-interface, and all other following sections, will be separated into their own panels, and all contained within the same windows form.

The teacher’s home page follows the standard Windows grey colour scheme, to make the system seem immediately familiar to teachers who have never used the program before. The text here is adaptive, and changes based on the logged in teacher’s name from the database, to welcome them. The client, Megan Taylor, has been used as an example here.

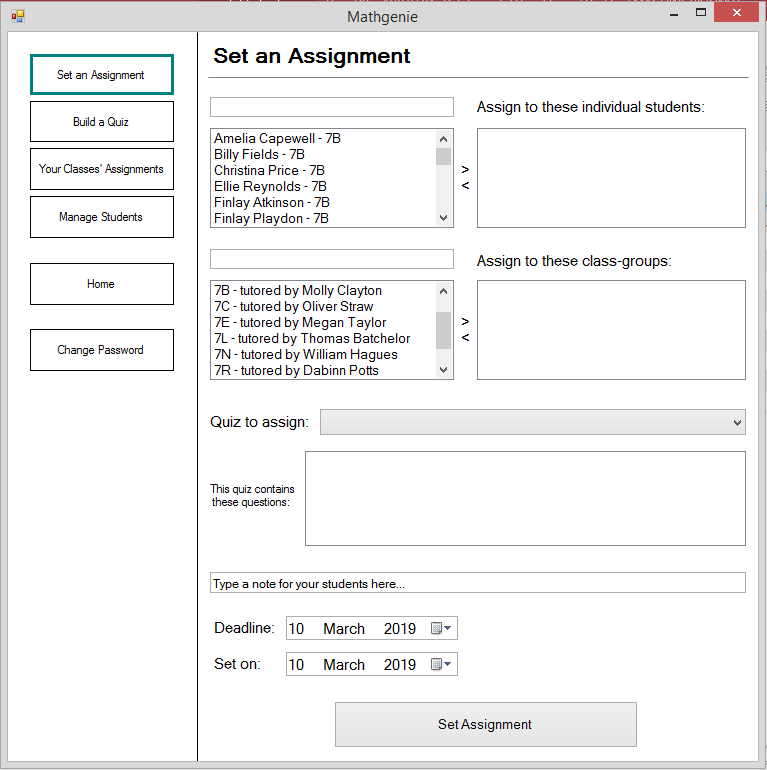


On the left hand side of this panel, there is a menu bar. For each section of the teacher-side of the system, there is a button in the menu, which when clicked, hides the visible panel and shows the respective panel for whichever button was clicked. The button whose panel is currently visible is highlighted with a coloured border, just as the ‘home’ button is here. This menu bar appears on the left hand side of every panel/part of the teacher user interface, in order to make navigation easy and intuitive.

### Teacher-side: Set an assignment panel:

Search box for filtering the below student selection box. Will provide ‘instant search’ so the user will not have to press a search button, and will instead update with every keystroke.

List-box of students, which can be selected to set an assignment to. The contents will be ordered by class-group, and then ordered by name alphabetically within class-groups, to make it intuitive to use. When a student is double clicked, they will appear in the right-hand list-box to have the assignment set for, and if double-clicked in the right-hand box, will be removed back to the left-hand box. The similar boxes below are for assigning assignments to whole class-groups, and work the same as these student list-boxes.



These input boxes are for selecting the due date and set date for an assignment. Date-time-pickers will be used, so a calendar will appear when clicked which a teacher can select a date from, for ease of use. Will default to today’s date.

When a quiz is selected in the drop-down box, the questions contained in the quiz will automatically appear in this list-box, so that the teacher user knows what questions they are setting in their assignment.

Drop down box for choosing a quiz for the assignment. Will be populated with all quizzes from the database automatically.

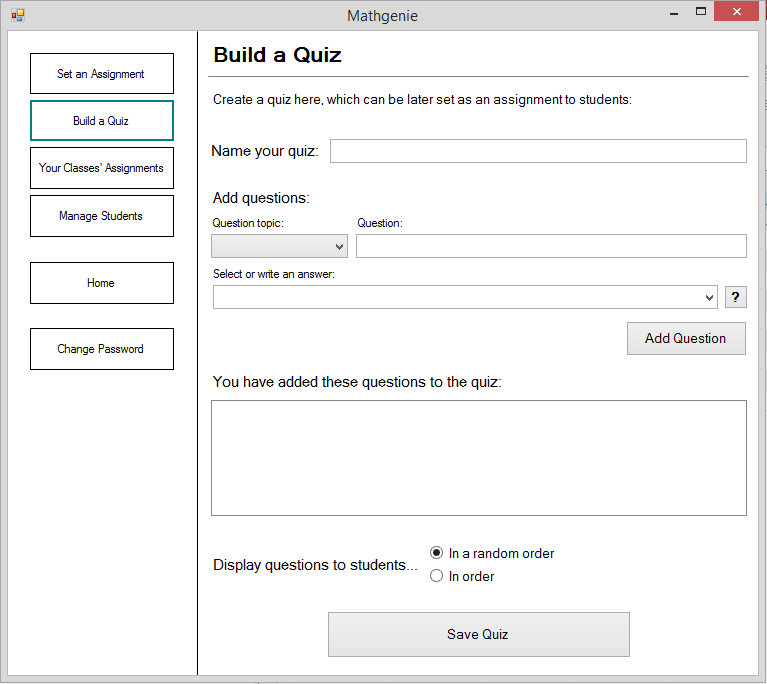
A note for students, that will appear before a student starts an assignment, can be written in this input-box. The shown text will be visible by default, so teachers know what the input box is for.

### Teacher-side: Build a quiz panel:

Here, there will be a combo-box for the teacher user to select a topic (e.g. shapes, addition, etc.) for the question they are creating to add to their quiz. This will be pre-populated with all available question topics from the database.

This input text-box will be used for a teacher to write the question-text for a question they are creating. Its use is labelled above. All input boxes use the same text-size to keep the UI neat.

This input text-box will be used for a teacher to name the quiz they are creating. The input-box will be clearly labelled, like all other elements of this form, with a label beside it, in order to make the UI easy to understand and use.



This list-box is used to display the questions a teacher has created and which have been added to the quiz they are creating. The VB list-box control is appropriate here due to it being a standard Windows control and therefore will be immediately familiar to teacher users. Questions will automatically appear here when the ‘Add Question’ button is clicked.

Used to input the answer to question being created. If the answer the teacher inputs is not yet supported by the AI, the ‘train AI’ form opens automatically when the ‘add question’ button is clicked.

Radio buttons have been used here, for a teacher user to choose whether they want their quiz they are creating to appear in a random order to students, or in the order that they added the questions. The VB radio button control is appropriate here, as when one button is checked, the other one is automatically unchecked. This is better than the alternative of a check-box, as separating the two options makes it easier to understand and use.

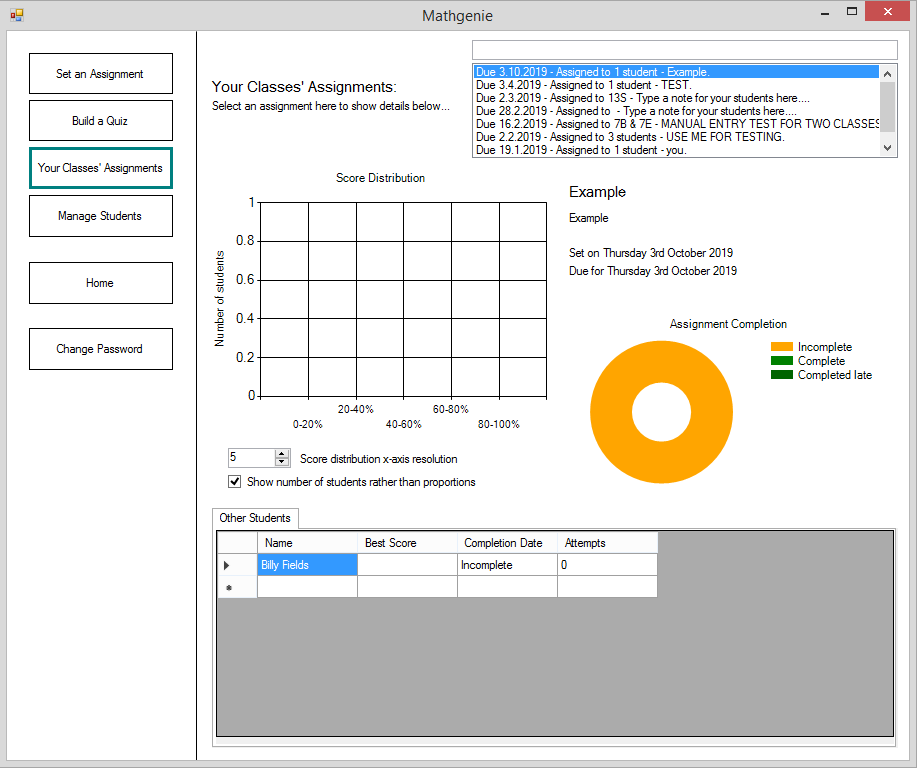
This button follows the standard Windows grey colour palette. When clicked, the details of the question that a teacher inputted will be stored in the database. The input-boxes for adding a question will then be cleared for the user’s convenience, so they can immediately begin adding another question.

### Teacher-side: View set assignments panel:

Please see the next page for notes on this UI panel.

Note 1

Note 2



Note 6

Note 5

Note 4

Note 3

Note 7

**Note 1:** Search box for filtering the below assignment selection box. Will provide ‘instant search’ so the user will not have to press a search button, and will instead update with every keystroke.

**Note 2:** List-box of assignments, which can be selected to view statistics of and a breakdown of scores of the students that have attempted it. The contents will be ordered by the date assignments are due, with the most recent assignments at the top of the list and old assignments filtering down the list, to make it intuitive to use. Only assignments which the logged in teacher set themselves are visible, so that the list does not get overpopulated and difficult to use. When the panel first opens, the most-recently set assignment at the top of the list-box will automatically be selected, to make the program more convenient to use. One example assignment is shown, selected, in the list-box on the UI design.

**Note 3:** Details of the assignment which is selected in the list-box will appear here. These details will be retrieved from the database when an assignment is clicked, and will include the name of the quiz for the assignment, any note which a teacher left when creating the assignment, and the set and due dates of the assignment.

**Note 4:** Here, there will be a doughnut chart (a variation of pie chart), showing a simple breakdown of the completion of the selected assignment. There will be three sections, all intuitively coloured (colours can be seen in the legend by the doughnut chart, on the UI design), split into proportion of students who have not completed the assignment, proportion who completed it on time, and proportion who completed it late. This chart provides an easy and fast way for teachers to look through assignments and find ones which have unsatisfactory completion.

**Note 5:** Here, there will be a bar chart of the distribution of scores which students who have completed the selected assignment have achieved. This can be used by teachers to see how well the population of students has done in the selected assignment. It is an easy way for teachers to see whether most students have done badly (i.e. there will be more students in the bars on the left) or whether most students have done well (i.e. there will be more students in the bars on the right), or anywhere in between.

**Note 6:** These are controls to customise the above score distribution bar chart. The top one will be a VB numeric-up-down input-box, in order to change the resolution of the x axis, i.e. to change the number of vertical bars on the graph, to give different views of the score distribution. The other one, below, will be a VB check-box, which changes the way the y axis is displayed. When unchecked, the percentage of students in each region of the score distribution is shown on the y axis. When checked, the number of students in each region of the score distribution is shown on the y axis instead.

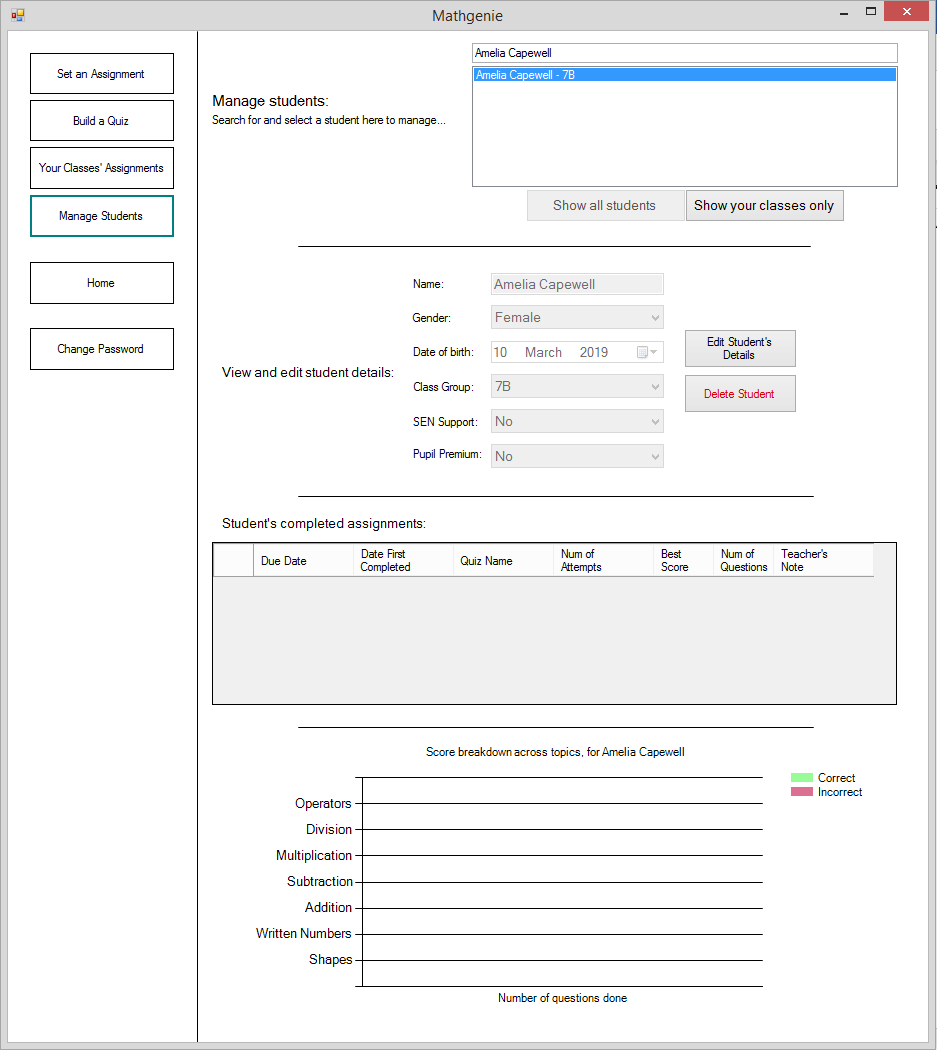
**Note 7:** Here, there will be a tabbed interface, split up with a tab for each class that an assignment has been set to, and one other tab for students who an assignment has been set to independently of a class-group. Within each tab there will be a VB data-grid-view UI control, which will list all of the students in the class-group a particular tab is for, and details such as their best score and number of attempts in the currently selected assignment.

### Teacher-side: Manage students panel:

Please see the next page for notes on this UI panel.

Note 1

Note 2



Note 5

Note 7

Note 6

Note 4

Note 3

**Note 1:** Search box for filtering the below student selection box. Will provide ‘instant search’ so the user will not have to press a search button, and will instead update with every keystroke.

**Note 2:** List-box of students, which can be selected to view details of. The contents will be ordered by class-group, and then ordered by name alphabetically within class-groups, to make it intuitive to use. When a student is clicked, the below UI elements will update to show details on the selected student. When the panel first opens, the student at the top of the list-box will automatically be selected, to make the program more convenient to use. In the UI design, one example student, Amelia Capewell, has been shown.

**Note 3:** Here are two buttons which are used to filter the list-box for selecting a student to view details of. They toggle between showing all students in the system in the list-box, or just students which are in a class-group taught by the logged in teacher. When one button is clicked, it will be disabled and the other button beside it will be enabled, essentially toggling between the two.

**Note 4:** This section of the user interface is for viewing and editing the selected student’s basic details. On the left, are 6 input boxes, for a student’s name, DOB, gender, class-group, SEN support and pupil premium status. By default, these boxes are read-only and are for viewing these details only. If the ‘edit student details’ button is clicked, these input boxes will be enabled for editing. The button’s text will then change to state ‘save changes’ which when clicked will save the newly inputted details in the input-boxes to the database, and make the boxes read-only again. The ‘delete student’ button, which follows the same design and size as the above button to make the UI uniform and neat, when clicked, will ask a user for confirmation, and when given, will delete a student’s data from the database.

**Note 5:** This division line, and the others above and below it, separate the UI into different sections to keep the panel organised. Given that this is one of the panels in the system with the most different UI elements, the divider lines are important so that the panel is still easy to use, despite its complexity.

**Note 6**: Here, a VB data-grid-view UI control will be used to display details on the assignments the selected student has attempted. These details can be seen in the column headings on the UI design. The data-grid-view control is very compact and uses small text, but this is appropriate for the teacher-side of the system, since it will be used with standard computers, rather than the high-DPI laptops and touchscreens that the student-side of the system will be used with.

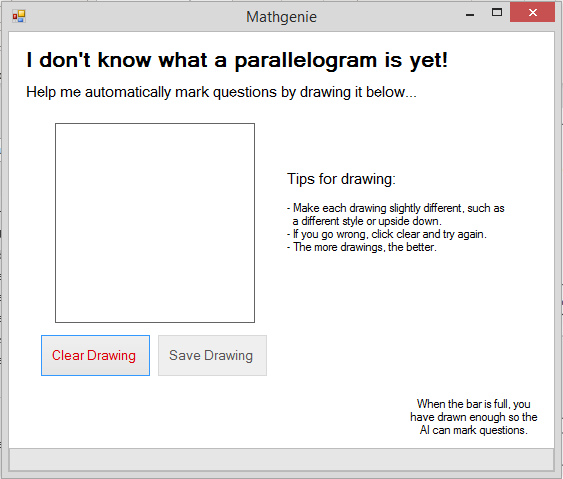
**Note 7:** This dynamically generated graph here will display how many questions the selected student has completed in each topic. It will be a stacked bar chart, with a bar for each question topic from the database (shown with some examples on the y axis). Each bar will be split into a green and red section, to show how many questions the selected student has got correct and incorrect respectively. This is similar to the topic-breakdown chart on the feedback panel of the student-side of the system, except that a legend and chart title have been added here, instead of the simpler text description of how to use the chart which is present on the student version. The legend and chart title will make the graph easy to use for a teacher, but could be difficult for a young student user to understand.

### Teacher-side: Train AI to recognise a new answer:

This will be an extra form, which will pop-up when the ‘add question’ button on the ‘build a quiz’ panel is clicked, if there are no entries in any of the AI training dataset text-files for the answer which the teacher has inputted as being the correct answer for their new question they are creating.

This title text is automatically set when the form opens. The answer that the teacher user wrote for the question they were creating, which the AI needs training to recognise, will appear in place of the example ‘parallelogram’ here. This title text is large and bold compared to the rest of the form, so that the teacher clearly knows what to draw.

This is my custom-created canvas UI control. The user will draw the shape, symbol or character specified in the bold title in this canvas, in order to train the AI to recognise it. The canvas has a black border around it, to indicate to the user where to draw. To the right-hand side of the canvas, there is some text, giving the user tips on how to best draw in order to train the AI most effectively.



This button, when clicked, will clear the canvas which the user is drawing on, by calling the clear canvas shared function. It is marked in red so that the user cannot easily accidentally click it, which would be frustrating.

This button is automatically enabled once the user has begun drawing in the canvas. It is disabled before so that they cannot mistakenly try to train the AI with a blank image, which would damage the AI dataset.

This bar will fill as the user draws more, to indicate the progress of the AI being trained to the user in an easy to understand manner. Once it is full, the user will know that the AI has been trained sufficiently. A message will also be shown to the user to tell them that the training of the AI with their new answer has been completed. The VB progress-bar UI control will be used, as it will be familiar to the user as something that indicates the progress of a task.

## Design of core algorithms

There are three core algorithms that will benefit from explicit design, due to either their complexity, or their important role within the system. These are:

* The artificial intelligence character classification algorithm
* The drawing canvas UI control
* The quiz system

The most complex SQL code for a few key parts of the program will also be designed here. These parts of the program will be detailed in the following section ‘design of core SQL statements’.

## Pseudocode of core algorithms

### Artificial intelligence character classification algorithm - pseudocode:

InputBitmap 🡨 Image drawn by user for classification

:: Convert InputBitmap to black and white:

FOR EACH pixel IN InputBitmap

Avg = R + G + B / 3 :: Average brightness of pixel

SET pixel TO RGB COLOUR (Avg, Avg, Avg)

NEXT

:: Crop white space surrounding drawing in InputBitmap:

FOR EACH pixel IN InputBitmap FROM TOP-LEFT CORNER

IF pixel IS COLOUR BLACK

RightPixel 🡨 pixel(x, y)

BottomPixel 🡨 pixel(x, y)

IF FIRST BLACK PIXEL FROM LEFT

LeftPixel 🡨 pixel(x, y)

END IF

IF FIRST BLACK PIXEL FROM TOP

TopPixel 🡨 pixel(x, y)

END IF

END IF

NEXT

CROP InputBitmap TO LeftPixel, RightPixel, TopPixel, BottomPixel

COMPRESS InputBitmap TO 20x20 pixels

:: Choose label for InputBitmap

HighestSimilarity 🡨 0

ChosenLabel 🡨 “”

FOR EACH IMAGE IN DataSet

Similarity 🡨 0

FOR EACH pixel IN InputBitmap

IF DatasetImage.Pixel(pixel.X, pixel.Y) = pixel THEN

Similarity 🡨 Similarity + 1

ELSE

Similarity 🡨 Similarity -1

END IF

NEXT

Similarity 🡨 Similarity / 400

:: 400 = Number of pixels in image

IF Similarity > HighestSimilarity THEN

HighestSimilarity 🡨 Similarity

ChosenLabel 🡨 Label of current dataset image

END IF

NEXT

OUTPUT ChosenLabel

### Drawing canvas UI control - pseudocode:

LineThickness 🡨 20 :: 20 used as example line thickness

LinePoints() 🡨 NULL

WHEN MOUSE POSITION ENTERS BOUNDS OF CANVAS

LastMouseCoords 🡨 MOUSE POSITION

ADD LastMouseCoords TO LinePoints()

NEXT

DO WHILE MOUSE POSITION WITHIN BOUNDS OF CANVAS

IF MOUSE POSITION <> LastMouseCoords THEN :: i.e. if mouse moves

ADD LastMouseCoords TO LinePoints()

Distance 🡨 Dist. between MOUSE POSITION and LastMouseCoords

:: Fill in gap if distance is large:

DO UNTIL Distance < LineThickness / 2

ADD point TO LinePoints() WHERE point IS BETWEEN LAST  
TWO POINTS ADDED TO LinePoints()

Distance 🡨 Dist. Between MOUSE POSITION AND MOST  
RECENTLY ADDED POINT TO LinePoints()

LOOP

LastMouseCoords 🡨 MOUSE POSITION

END IF

FOR EACH NEW point IN LinePoints()

DRAW DOT WITH DIAMETER LineThickness ON CANVAS AT point(x, y)

:: These dots will overlap to create a smooth rounded line

NEXT

LOOP

### The quiz system - pseudocode:

QuestionsForQuiz() 🡨 NULL

QuizInOrder 🡨 NULL

NumberOfQuestions 🡨 NULL

CorrectQuestionsCount 🡨 0

CurrentQuestion 🡨 1

:: Set up quiz:

IF QUIZ CALLED BY ASSIGNMENT BEING SELECTED THEN

QuestionsForQuiz() 🡨 QuestionText, Answer, OF QUESTIONS FROM  
 QuestionsOfQuizzes DATABASE TABLE WHERE QuizID  
 = QuizID WHERE QuizID = AssignmentID OF  
 ASSIGNMENT WHICH WAS SELECTED

QuizInOrder 🡨 GET FROM DATABASE FOR AssignmentID OF ASSIGNMENT  
 WHICH WAS SELECTED

NumberOfQuestions 🡨 COUNT OF ELEMENTS IN QuestionsForQuiz()

ELSEIF QUIZ CALLED FROM PRACTICE MODE THEN :: i.e. randomly generated  
 quiz is wanted

QuestionsForQuiz() 🡨 GET QUESTIONS FROM Questions DATABASE TABLE  
 WHERE TopicID = ONE OF TOPICS WHICH WAS CHOSEN  
 BY USER IN PRACTICE MODE SET UP

QuizInOrder 🡨 FALSE

NumberOfQuestions 🡨 AMOUNT CHOSEN BY USER IN PRACTICE MODE SET UP

END IF

:: Display quiz questions to user one by one and check answers:

GivenAnswer 🡨 NULL

NumberOfCanvasesToShow 🡨 NULL

IF QuizInOrder = FALSE THEN

SeenQuestionIndexes() 🡨 NULL

END IF

DO WHILE CurrentQuestion <= NumberOfQuestions

IF QuizInOrder = TRUE THEN

OUTPUT QuestionsForQuiz(CurrentQuestion).QuestionText

ELSE

DO

ChosenQuestion 🡨 RANDOM INTEGER BETWEEN 1 AND  
 NumberOfQuestions

LOOP UNTIL SeenQuestionIndexes() DOES NOT CONTAIN  
 ChosenQuestion

ADD ChosenQuestion TO SeenQuestionIndexes()

OUTPUT QuestionsForQuiz(ChosenQuestion).QuestionText

END IF

NumberOfCanvasesToShow 🡨 COUNT OF NUMBER OF CHARACTERS IN STRING  
 QuestionsForQuiz(CurrentQuestion).Answer

DO

DISPLAY UI ELEMENT Canvas

LOOP FOR NumberOfCanvasesToShow

INPUT HAND-DRAWN ANSWER ON CANVASES

AnswerCharacter 🡨 NULL

FOR EACH Canvas IN UI CONTROLS

AnswerCharacter 🡨 CALL AI CHARACTER CLASSIFIER FUNCTION USING  
 Canvas.Image AS PARAMETER

GivenAnswer 🡨 GivenAnswer + AnswerCharacter

NEXT

IF GivenAnswer = QuestionsForQuiz(CurrentQuestion).Answer THEN

OUTPUT ‘Your answer was correct’

CorrectQuestionsCount 🡨 CorrectQuestionsCount + 1

ELSE

OUTPUT ‘Your answer was incorrect’

END IF

CurrentQuestion 🡨 CurrentQuestion + 1

LOOP

:: Give user instant feedback:

Score 🡨 CorrectQuestionsCount / NumberOfQuestions

OUTPUT ‘You got ’ & CorrectQuestionsCount & ‘ out of ’ & NumberOfQuestions  
 & ‘ questions correct’

IF Score > 0.7 THEN

DISPLAY HAPPY EMOTION FACE

ELSEIF Score < 0.3 THEN

DISPLAY SAD EMOTION FACE

ELSE

DISPLAY NEUTRAL EMOTION FACE

END IF

## Design of core SQL statements

SQL statements which are part of the core functionality of my system have been designed here. I have chosen a spread of SQL statements to design before implementing them into the system, in order to have designs for SQL statements containing all the SQL commands I will use.

### Reading un-attempted assignments for a student from the database:

“SELECT Assignments.AssignmentID, Quizzes.QuizName, Assignments.SetDate, Assignments.DueDate, Teachers.FirstName, Teachers.LastName, Assignments.AssignmentNote, Quizzes.QuizInOrder, Assignments.QuizID FROM Teachers INNER JOIN Quizzes INNER JOIN Assignments ON Quizzes.QuizID = Assignments.QuizID ON Teachers.TeacherID = Assignments.AssignerTeacherID INNER JOIN AssignmentsForIndividuals ON Assignments.AssignmentID = AssignmentsForIndividuals.AssignmentID WHERE AssignmentsForIndividuals.StudentID=" & LoggedInStudent\_ID & " AND (SELECT COUNT(AssignmentAttempts.AssignmentID) AS NumberOfAssignmentAttempts FROM AssignmentAttempts WHERE AssignmentAttempts.StudentID=" & LoggedInStudent\_ID & " AND AssignmentAttempts.AssignmentID = Assignments.AssignmentID;)=0 ORDER BY Assignments.DueDate DESC;”

This is a parametrised cross-tabular aggregate (contains nested SQL) SQL statement, for reading all assignments from the database that have been set to a student that they have not yet attempted. This joins the Teachers, Quizzes, Assignments and AssignmentsForIndividuals tables using INNER JOIN commands. The Teachers and Assignments tables are joined by TeacherID, the Assignments and AssignmentsForIndividuals tables are joined by AssignmentID, and the Quizzes and Assignments tables are joined by QuizID. A SELECT COUNT command is used within the WHERE command of the main SQL statament. The COUNT command is part of a nested SQL statement, to count the number of attempts on each assignment which was found in the main SQL statement from the Assignments table. This is done by counting the number of rows in the AssignmentAttempts table where StudentID is one specified in a parameter, and AssignmentID is the current AssignmentID found in the main SQL statement containing this nested one. If this returns 0 (hence the ‘=0’ after the nested statement), i.e. there are no attempts, then the assignment is one of the assignments returned from the SQL statement. Finally, the ORDER BY command is used to order the returned assignments by ‘DueDate’, and DESC causes these to be in descending order.

### Reading past attempted assignments for a student from the database, grouped by assignment:

“SELECT Assignments.AssignmentID, Quizzes.QuizName, (SELECT Max(AssignmentAttempts.DateOfAttempt) FROM AssignmentAttempts WHERE AssignmentAttempts.AssignmentID=Assignments.AssignmentID AND AssignmentAttempts.StudentID=" & LoggedInStudent\_ID & ") AS DateOfLatestAttempt, (SELECT Count(AssignmentAttempts.AssignmentID) FROM AssignmentAttempts WHERE AssignmentAttempts.AssignmentID=Assignments.AssignmentID AND AssignmentAttempts.StudentID=" & LoggedInStudent\_ID & ") AS NumberOfAttempts, (SELECT Max(AssignmentAttempts.CorrectQs) FROM AssignmentAttempts WHERE AssignmentAttempts.AssignmentID=Assignments.AssignmentID AND AssignmentAttempts.StudentID=" & LoggedInStudent\_ID & ") AS HighestCorrectQs, (SELECT Count(QuizzesQuestions.QuizID) FROM QuizzesQuestions WHERE QuizzesQuestions.QuizID=Assignments.QuizID) AS TotalNumberOfQs, Teachers.FirstName, Teachers.LastName, Assignments.AssignmentNote, Quizzes.QuizInOrder, Assignments.QuizID FROM Teachers INNER JOIN Quizzes INNER JOIN Assignments ON Quizzes.QuizID = Assignments.QuizID ON Teachers.TeacherID = Assignments.AssignerTeacherID INNER JOIN AssignmentsForIndividuals ON Assignments.AssignmentID = AssignmentsForIndividuals.AssignmentID INNER JOIN AssignmentAttempts ON Assignments.AssignmentID = AssignmentAttempts.AssignmentID WHERE AssignmentAttempts.StudentID=" & LoggedInStudent\_ID & " GROUP BY Assignments.AssignmentID, Quizzes.QuizName, Teachers.FirstName, Teachers.LastName, Assignments.AssignmentNote, Quizzes.QuizInOrder, Assignments.QuizID ORDER BY MAX(AssignmentAttempts.DateOfAttempt) DESC;”

This is a parametrised cross-tabular aggregate SQL statement, for reading all assignments from the database which a student has attempted in the past. It reads primarily from the ‘AssignmentAttempts’ table, and groups multiple attempts of the same assignment together into one row. The following data is selected by the SELECT statement:

* The ID of assignments.
* The name of the quiz.
* The most recent date of attempt for each attempt of the same assignment (by using a MAX function on AssignmentAttempts.DateOfAttempt, for each row of the AssignmentAttempts table with the same AssignmentID).
* The number of attempts a student has made on each assignment, by using a COUNT function to count the number of times an AssignmentID is present alongside an inputted StudentID (which is a parameter to the SQL function) in the AssignmentAttempts table.
* The highest number of questions a student has got correct on all attempts of each assignment, by using a MAX function on AssignmentAttempts.CorrectQs similarly to above.
* Counting the number of questions in each Assignment which a student has attempted, by using a COUNT function on the number of questions in the QuizzesQuestions table for the QuizID of each Assignment (which are associated in the Assignments table), for each AssignmentID that is associated with the parametrised StudentID in the AssignmentAttempts table.
* The first and last names of the teacher who set each assignment.
* The assignment note for each assignment.
* Whether the quiz of each assignment is in order or randomised (so that the data required to attempt an assignment again has been read to the table here).
* The QuizID of the quiz which an assignment contains.

The output of this SQL query is grouped by AssignmentID, QuizName, teacher FirstName and LastName, AssignmentNote, QuizInOrder, and QuizID, using the GROUP BY command. This means that each outputted row from the SQL query, where those data fields are the same, will be grouped into one. This is done since this SQL query is for reading all attempts of assignments, where each attempt on the same assignment (i.e. the same AssignmentID) is grouped together.

Finally, the output of this SQL query is ordered by the most recent attempt date for each outputted assignment. This is obtained for each output row, using a MAX function on the AssignmentAttempts.DateOfAttempt for each output row, getting the highest date for each assignment.

### Write the result of a quiz to the database:

"INSERT INTO AssignmentAttempts (StudentID, AssignmentID, CorrectQs, DateOfAttempt) VALUES (" & LoggedInStudent\_ID & "," & ActiveAssignmentID & ",'" & CorrectAnswersCount & "', #" & DateTime.Now & "#)"

This is a parametrised SQL statement which uses the INSERT INTO command to add an entry to the AssignmentAttempts table. Firstly, the columns to insert values for in the new row are specified, then, following the VALUES command, parameters are present, which will contain values to be written to the respective columns specified beforehand.

### Update modified student details in the database:

“UPDATE Students SET Students.FirstName = '" & FName & "', Students.LastName = '" & LName & "', Students.DOB = #" & Dtp\_StudentDOB.Value & "#, Students.Gender = '" & Cmb\_StudentGender.Text & "', Students.ClassCode = ' " & Cmb\_StudentClassGroup.Text & "', Students.SENSupport = " & SEN & ", Students.PupilPremium = " & PupPrem & " WHERE Students.StudentID = " & SelectedStudentID & ";”

This is a parametrised SQL statement which uses the UPDATE command to change values of specified columns, for rows where the StudentID is the integer specified in the ‘SelectedStudentID’ parameter. The SET command is used to specify the columns to change, and each column name is followed by a parameter, which will contain the value to update the column with. In real use, this will only change the values of each specified column for one student at once, as each row in the Students table has a different StudentID.

### Delete a student’s details from the database:

“DELETE FROM Students WHERE Students.StudentID = " & SelectedStudentID & ";”

This is a parametrised SQL statement which uses the DELETE command to remove entries from the ‘Students’ table where the StudentID is the integer specified in the ‘SelectedStudentID’ parameter. This will only ever remove one row at a time in real use, since there is one line in the Students table containing a respective student’s personal details.

## Prototyping of the AI character classification system

Here is some documentation of the prototyping of the combination of different methods of choosing a label for an image inputted into the AI character classifier. Also included here is documentation of how the fundamental AI training data-sets (numerical answers and alphabetical answers) were created.

The AI character classification algorithm uses a nearest-neighbour method to choose a label for an inputted bitmap. This essentially means that the AI fundamentally works by comparing an input bitmap, which it does not know what it is, against many other labelled (knows what it is, e.g. it knows it is a picture of an ‘8’) bitmap images, and labels the input bitmap with the label of the most similar bitmap in its training dataset. With my algorithm, rather than just comparing the input image to each dataset image once, I will use multiple methods of choosing which is the most similar image in the training set, and take an average of the outputs of the multiple methods to choose a final label for the unknown input image, to improve the accuracy of the classification.

I will use four methods of comparing the input bitmap image to a dataset of labelled images:

* Comparing the input image, cropped down and fit to a square, against a dataset of labelled images, and choosing the label of the most similar image.
* Comparing the input image, cropped down and stretched to fit a square, against a dataset of labelled images, and choosing the label of the most similar image.
* Comparing the input image, cropped down and fit to a square, against a dataset of labelled images, and taking an average of the similarly between the input image and all images in the dataset of each label. The label which has the highest average similarity will be chosen.
* Comparing the input image, cropped down and stretched to fit a square, against a dataset of labelled images, and taking an average of the similarly between the input image and all images in the dataset of each label. The label which has the highest average similarity will be chosen.

The label which has the highest score, from all of the above four methods of choosing a label, will then be outputted as the final chosen label for the unknown input image.

The prototyping begins on the following page.

### Processing the input image before classification:

#### Stretching the input image to a 20x20 pixel square:

For comparing the input image to the training images in the AI data-set, some processing must be done first. This will be converting the input image to black and white, then compressing and stretching the input image to a 20x20px square.

20x20 pixels in size has been chosen since it is small enough that the AI can compare hundreds of images of that size very quickly, but also large enough to show enough detail to differentiate between characters and shapes, so that the AI can classify with reasonable accuracy.

The VB resize function can be used to stretch an image to 20x20 pixels. This function changes an image’s dimensions to ones inputted, so by inputting a width and height of 20 pixels, an image that is larger will be compressed, and a non-square image will be stretched to fit the dimensions.

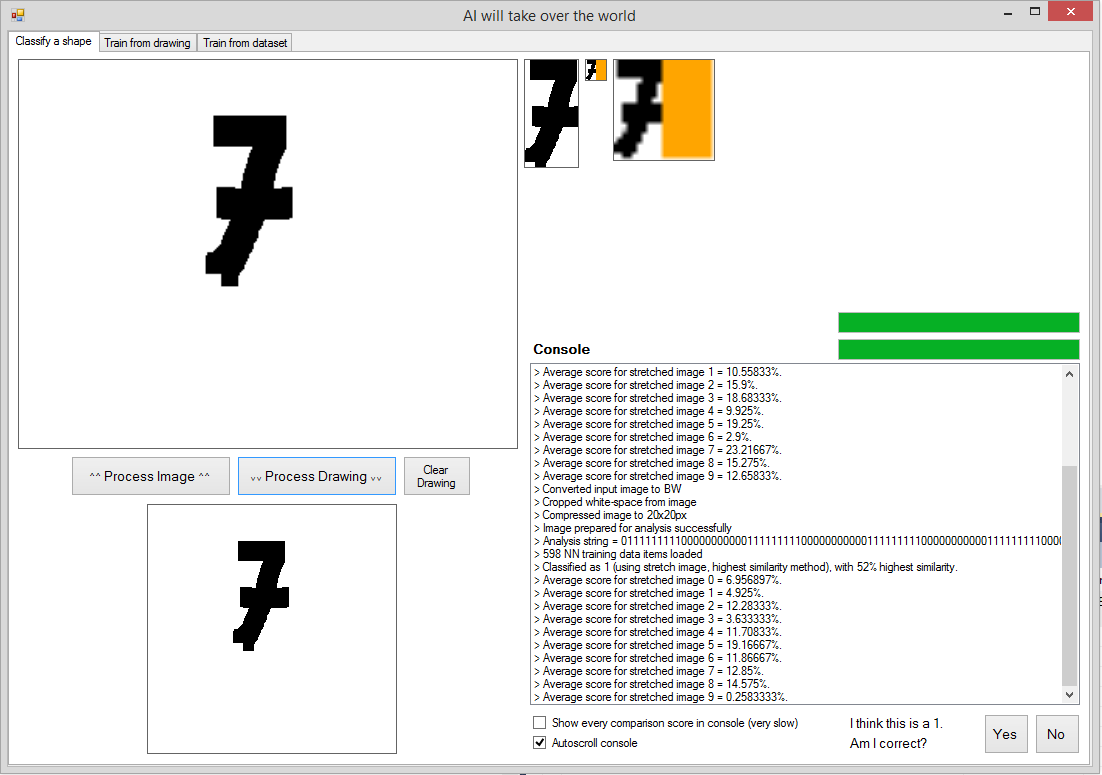
Function BMStretchToSize(ByVal bm As Bitmap, ByVal w As Integer, ByVal h As Integer)

Return New Bitmap(bm, w, h)

End Function

#### Fitting the input image to a padded 20x20 pixel square:

Fitting the input image to a 20x20 pixel square, without stretching, is as an additional method of classifying an input image as well as just comparing the input image stretched to fit a 20x20 pixel square:



An orange background to the 20x20 pixel fitted image, rather than a white background, has been used here in development, to obviously show where the image was fitted, for testing. I started by fitting the cropped image to the top-left corner, as easier to program.

Function BMFitToSize(ByVal bm As Bitmap, ByVal GreatestWorH As Integer)

Dim oldW As Integer = bm.Width

Dim oldH As Integer = bm.Height

Dim newW, newH As Integer

'Creates a bitmap with the same aspect ratio as the original, but downsized to a maximum width or height of a given number of pixels (20).

If oldW > oldH Then

newW = GreatestWorH

newH = (oldH / oldW) \* GreatestWorH

Else

newW = (oldW / oldH) \* GreatestWorH

newH = GreatestWorH

End If

bm = New Bitmap(bm, newW, newH) 'New bitmap created, but not square (i.e. has no white space surrounding to make it a square).

Dim bm\_NEW As New Bitmap(GreatestWorH, GreatestWorH)

Graphics.FromImage(bm\_NEW).FillRectangle(Brushes.Orange, 0, 0, bm\_NEW.Width, bm\_NEW.Height)

'Paints the downsized, non-square bitmap that was created onto a square bitmap. This is needed so all bitmaps end up with the same width and height.

For y = 0 To bm.Height - 1

For x = 0 To bm.Width - 1

bm\_NEW.SetPixel(x, y, bm.GetPixel(x, y))

Next

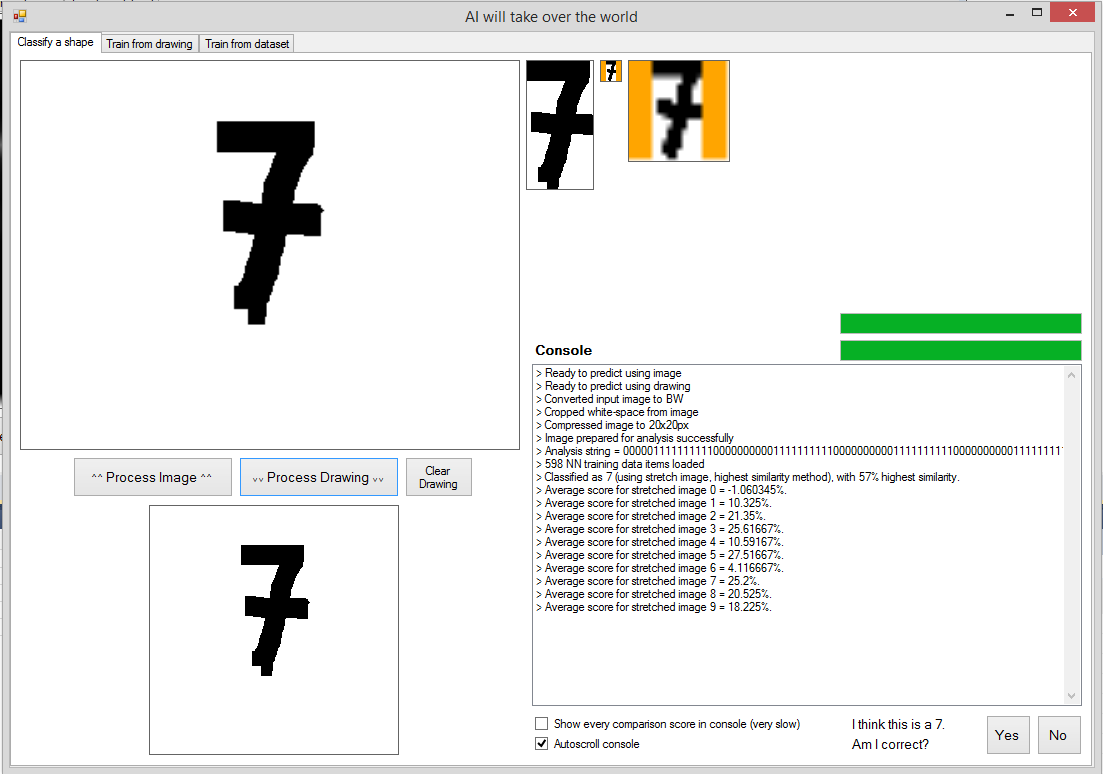
Next

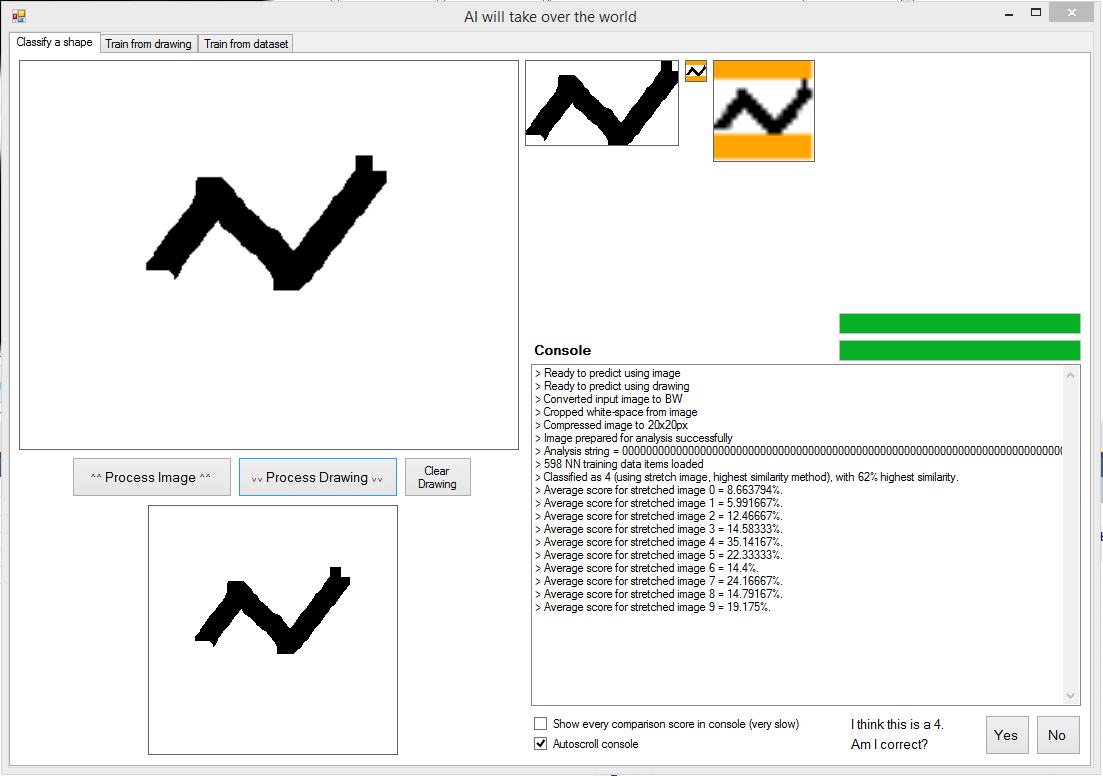
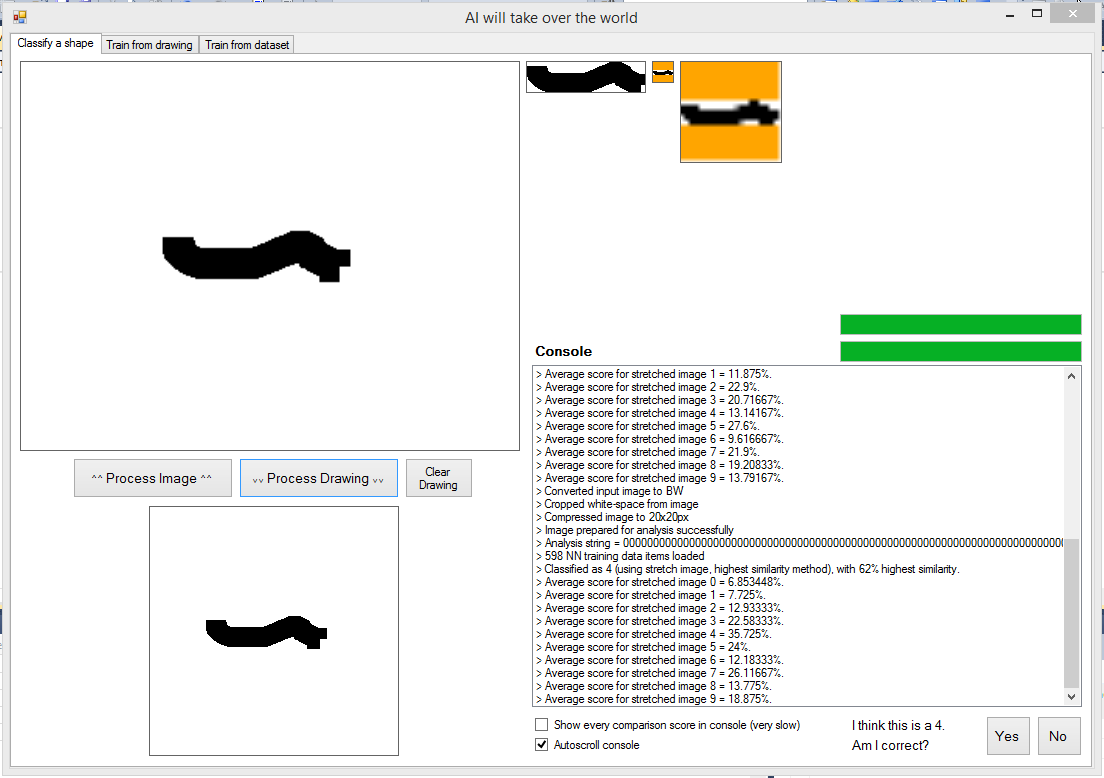
Return bm\_NEW

End Function

Once this was functioning, I got the image to be centred in a 20x20 pixel square, with padding either side, instead of to the left.

In the following screenshots, the far right-hand image at the top is the one inputted into the AI algorithm for comparison against dataset images:





Function BMFitToSize(ByVal bm As Bitmap, ByVal GreatestWorH As Integer)

Dim oldW As Integer = bm.Width

Dim oldH As Integer = bm.Height

Dim newW, newH As Integer

'Creates a bitmap with the same aspect ratio as the original, but downsized to a maximum width or height of a given number of pixels (20).

If oldW > oldH Then

newW = GreatestWorH

newH = (oldH / oldW) \* GreatestWorH

Else

newW = (oldW / oldH) \* GreatestWorH

newH = GreatestWorH

End If

bm = New Bitmap(bm, newW, newH) 'New bitmap created, but not square (i.e. has no white space surrounding to make it a square).

Dim bm\_NEW As New Bitmap(GreatestWorH, GreatestWorH)

Graphics.FromImage(bm\_NEW).FillRectangle(Brushes.Orange, 0, 0, bm\_NEW.Width, bm\_NEW.Height)

Dim Xoffset As Integer = (GreatestWorH - newW) / 2

Dim Yoffset As Integer = (GreatestWorH - newH) / 2

'Paints the downsized, non-square bitmap that was created onto a square bitmap. This is needed so all bitmaps end up with the same width and height.

For y = 0 To bm.Height - 1

For x = 0 To bm.Width - 1

bm\_NEW.SetPixel(x + Xoffset, y + Yoffset, bm.GetPixel(x, y))

Next

Next

Return bm\_NEW

End Function

An odd-numbered width or height (e.g. 13x20 pixels) cannot be centred properly in a 20x20 pixel image, but is off-side by one pixel. This doesn’t matter however, since all images are treated the same way and therefore will be offset from the centre in the same direction always (can be seen on second example screenshot). Essentially, since the same fitting-to-square algorithm will be used for processing images which the AI is trained with, and images which are inputted into the AI algorithm to be classified, an image with an odd-numbered width or height will always be centred slightly to the left or top, not variable, so this won’t affect classification accuracy.

No validation is required around defining the Xoffset and Yoffset variables, as it would never be doing an impossible division (i.e. dividing by zero).

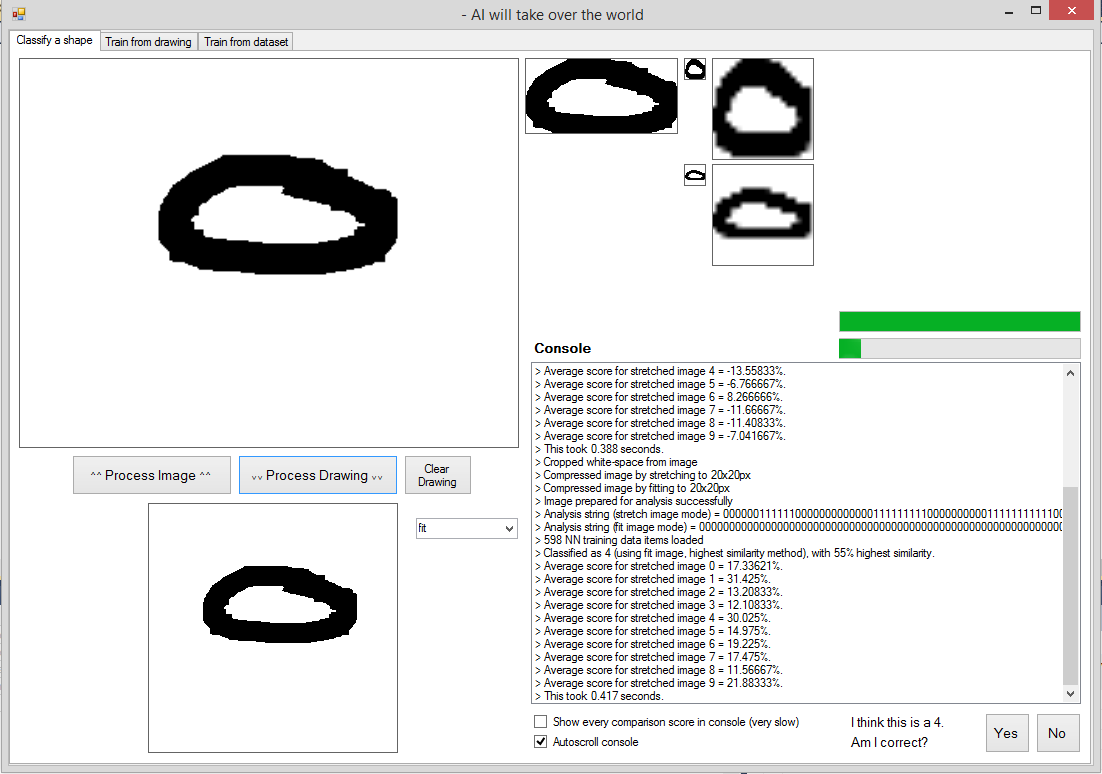
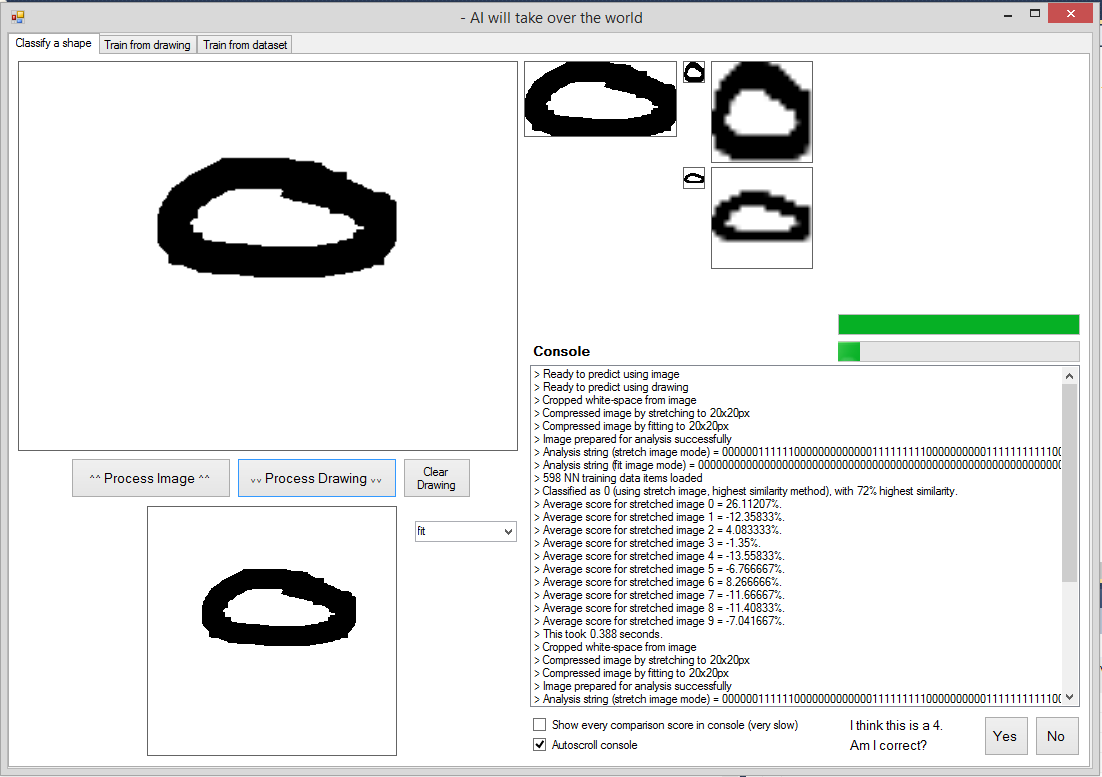
### Using multiple methods for classifying the input image, to improve accuracy:

Sometimes fit image and sometimes stretch image classification modes have a higher certainty. Sometimes only one is correct.

The scores for each label that come out of the functions are ranked from 10 to 1 (with 10 being the best and 1 the worst). The ranks are added (i.e. if all 4 functions classified an image with label ‘3’ as the most likely, each function would have a rank of 10 for label ‘3’, and the total rank-score for ‘3’ would be 40), and the label with the highest overall rank from all functions is chosen.

#### Using two methods:

* Stretch image, highest similarity classification
* Fit image, highest similarity classification



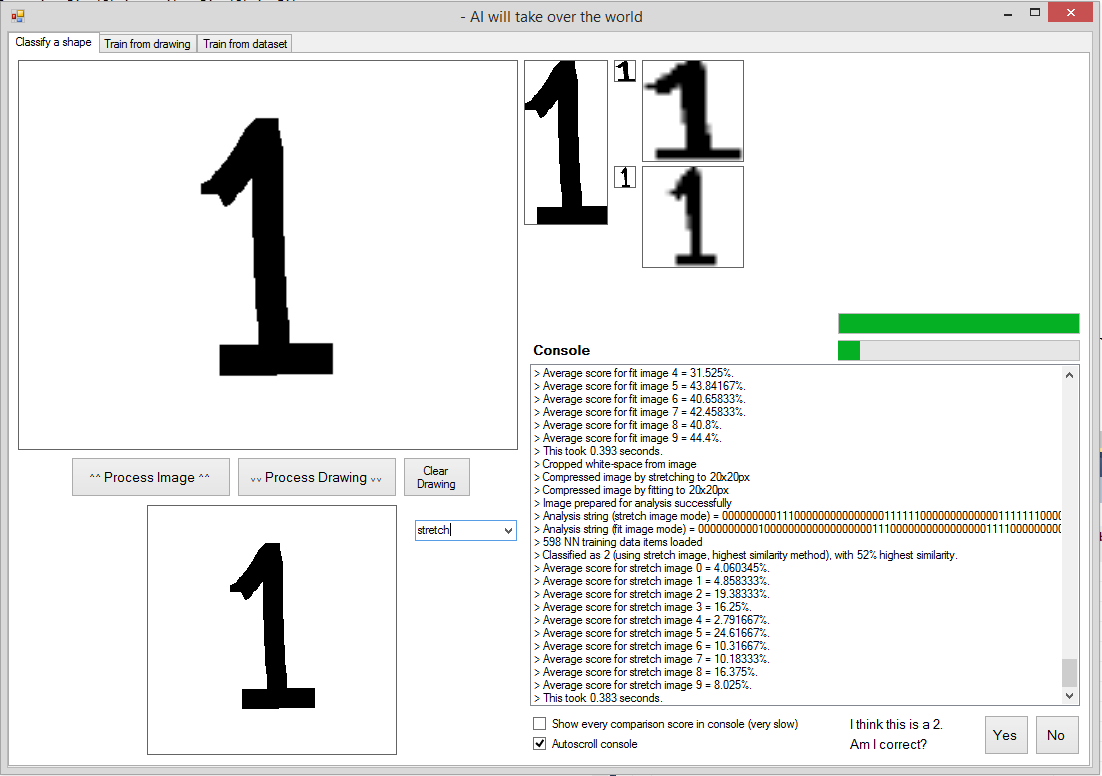
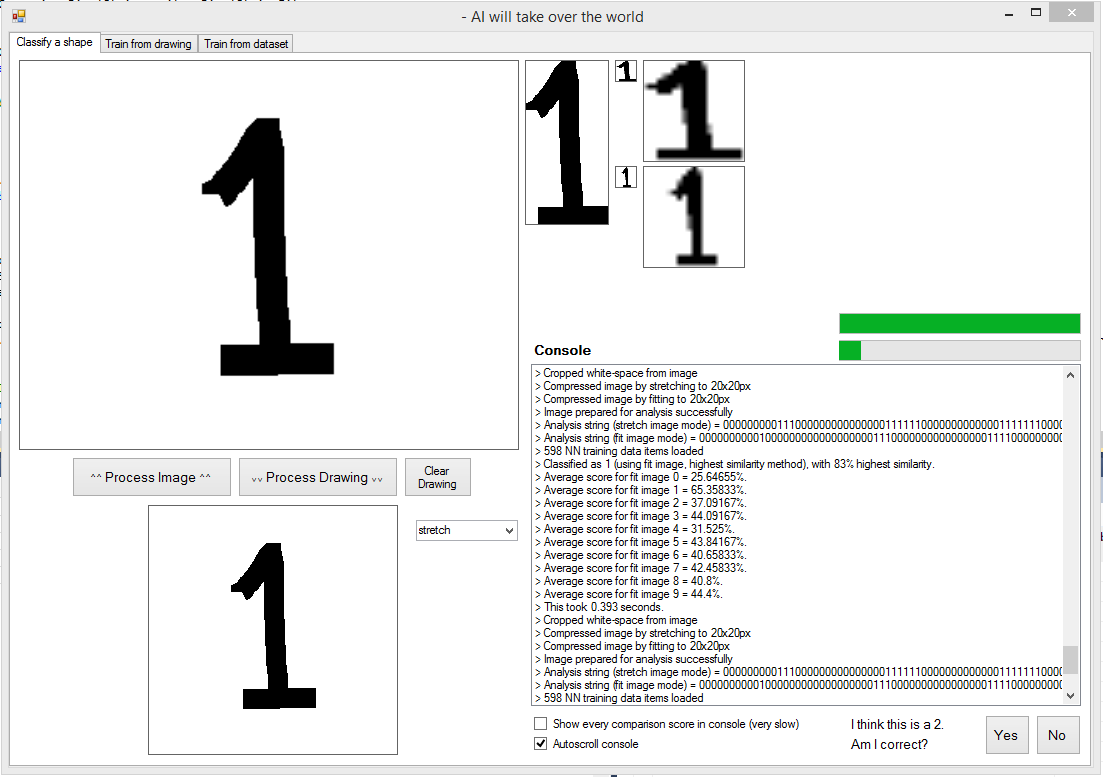
In the screenshot of the console above, stretch image classification had 72% highest similarity and correctly identified the image as a ‘0’. Fit image classification had 55% highest similarity and incorrectly identified the image as a ‘4’. The image was in fact a zero, which can be seen in the screenshot above the console. Therefore, by combining methods of choosing a label for an input image, the correct label can be chosen even when one method is incorrect.

Different classification modes will often classify images as different labels. When they disagree (like above) there are a few ways of solving the problem:

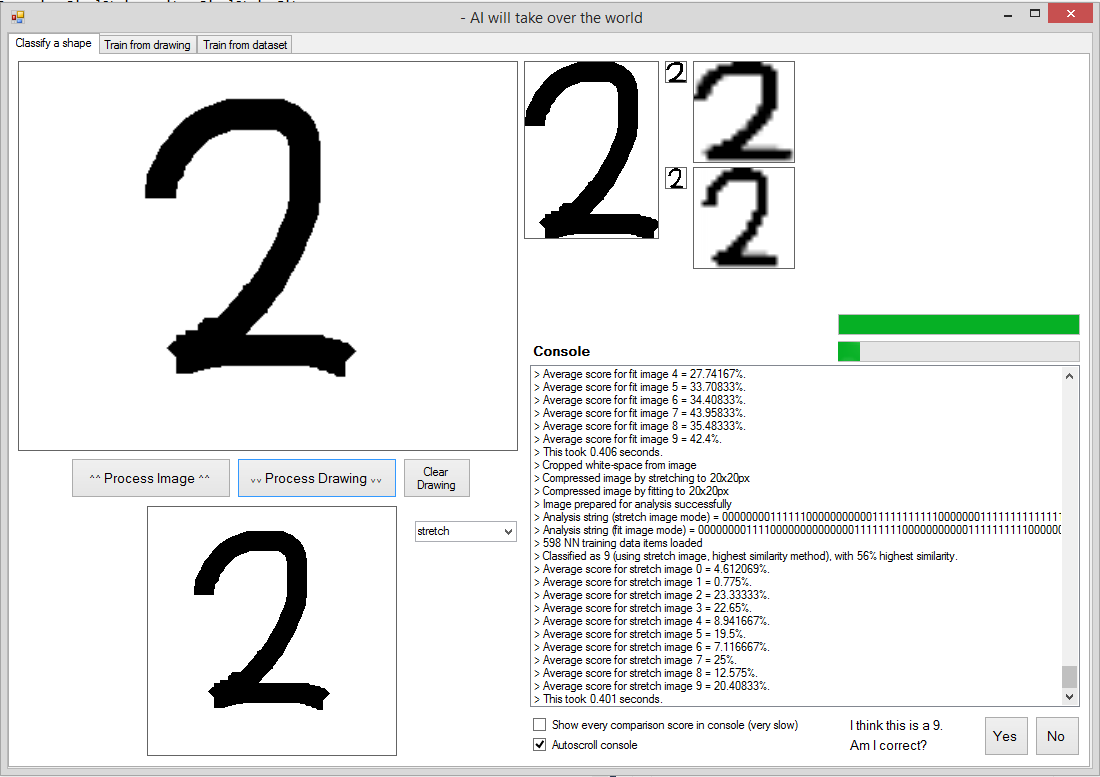
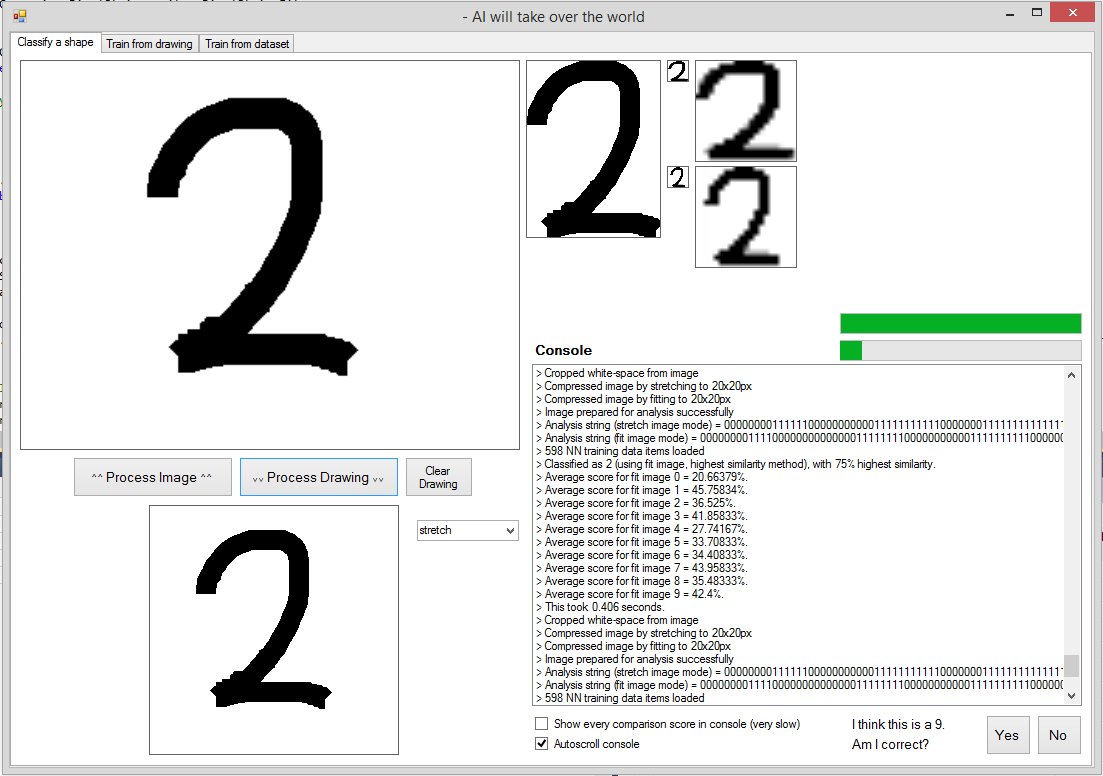
* Choose the label from the classification mode with the highest certainty.
* Look at the scores/certainties for each possible label from each classification mode (fit, stretch, average fit, average stretch), and add them. The label with the highest overall score/certainty will be chosen as the final classification.
* Simpler version of above: Majority voting of the final output label from many classification methods. E.g. labels 4, 9, 1, 4 are outputted, choose 4.

The second method will be used, of adding the certainties from each classification method, and picking the label with the highest combined certainty.

Further evidence of where using two classification methods helps accuracy, when one method is wrong and one is still correct:



^^ In this case, fit mode has a higher certainty and is correct, while stretch mode is wrong.

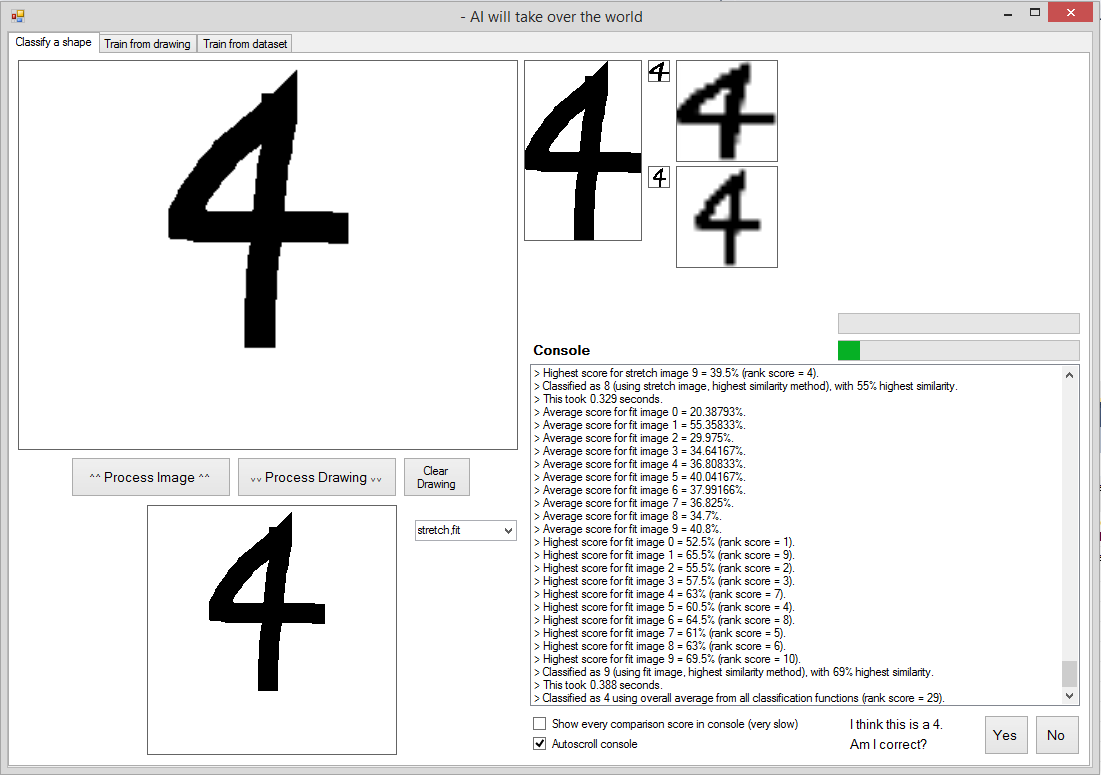
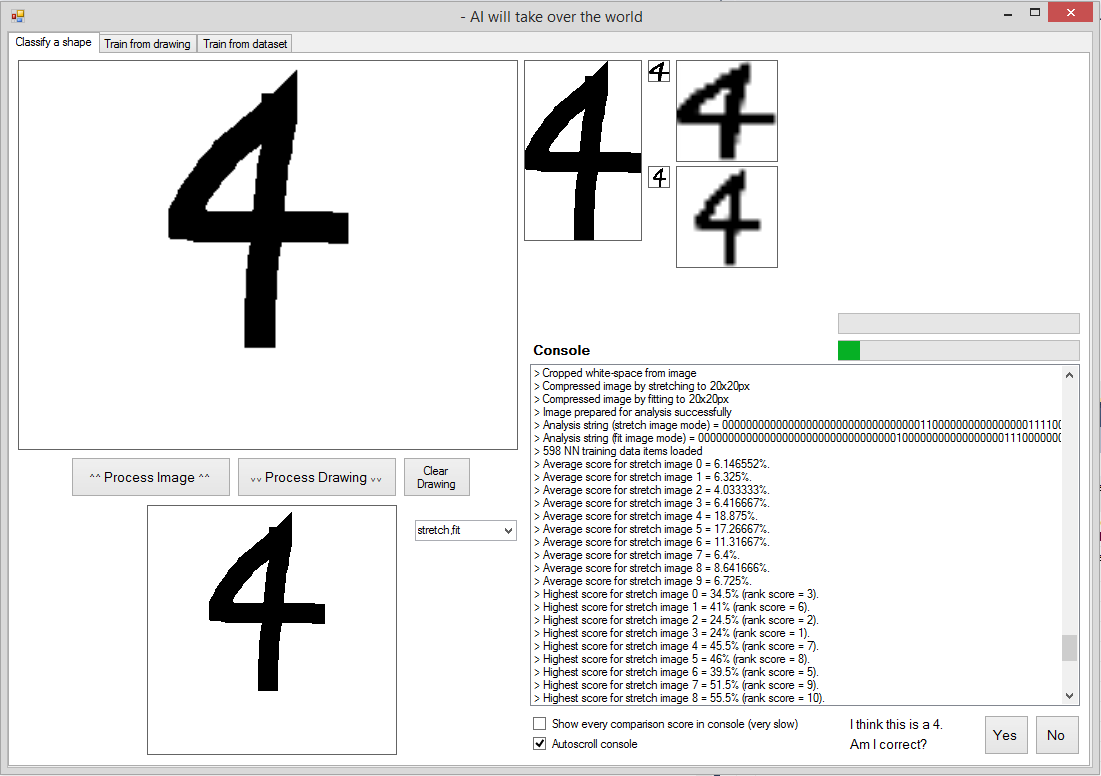


^^ In this case, fit is correct, stretch is wrong. It can also be noted that the average scores (which are not yet used in classification) for stretch mode indicate the correct answer, so the combination of more classification modes (such as averaging) could improve reliability even further.

#### Using four methods:

* Stretch image, highest similarity classification
* Fit image, highest similarity classification
* Stretch image, average similarity classification
* Fit image, average similarity classification

Here is an example where using these four methods worked to improve the classifier:

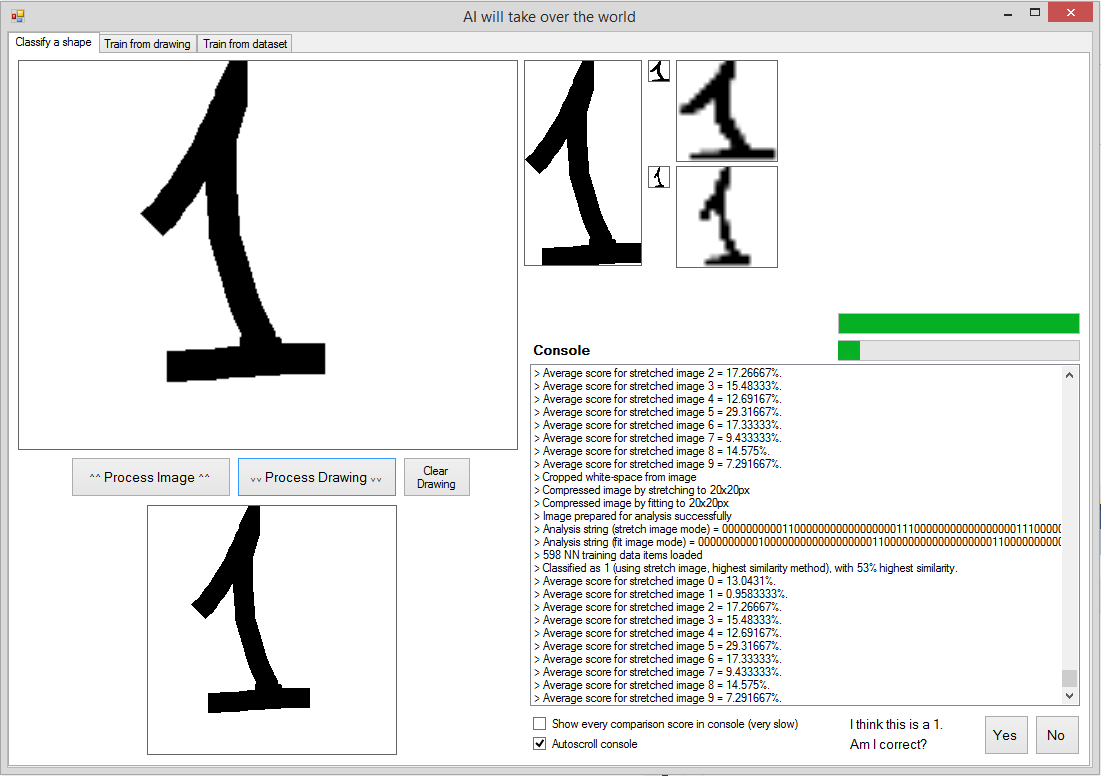


Here, two additional methods for classifying the input image have been used. The original two methods classified the input image as the label of the image in the training data-set which is it most similar to.

The two new methods for classification take an average of the similarity between the input image and all images from the training data-set for each possible label, and classifies the input image with the label which had the highest average similarity. This prevents one anomalously similar image in the training data-set to the input image, from skewing the results, and therefore should improve the accuracy of classification.

In the screenshot on the previous page, the stretch image, highest score function incorrectly classified the image as ‘8’, and the fit image, highest score function incorrectly classified the image as ‘9’. However, one averaging method classified the input image as a ‘4’, and the other averaging method, despite not classifying the input image as a ‘4’, gave it a high score (i.e. outputted that ‘4’ is one of the more likely labels). Therefore, overall, the image was correctly classified as ‘4’.

However, the average scoring methods are not always correct:

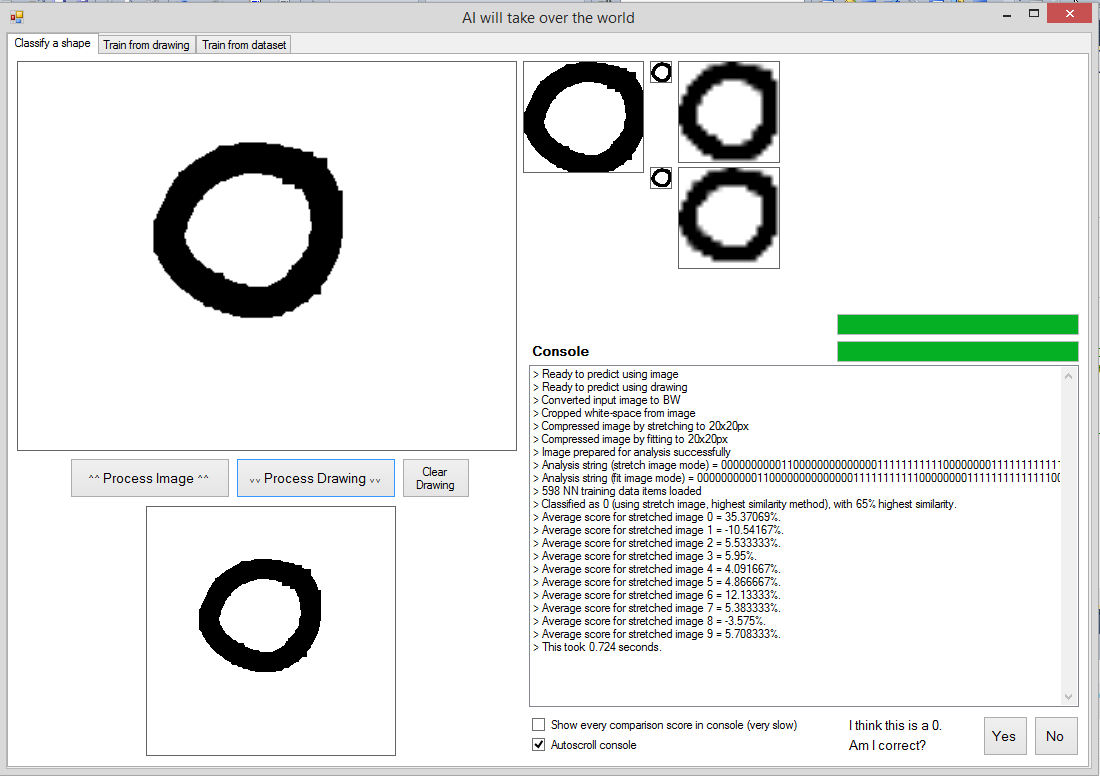


In the above screenshot, the average score method is very wrong, but the image is still classified as a ‘1’ correctly, because the highest-score classification method is used. Therefore in some cases, the average-score classification methods are not correct, so multiple methods including the highest-similarity classification methods are still needed, with their results weighted, to give an overall result to be used for classification.

### Improving speed of classification:

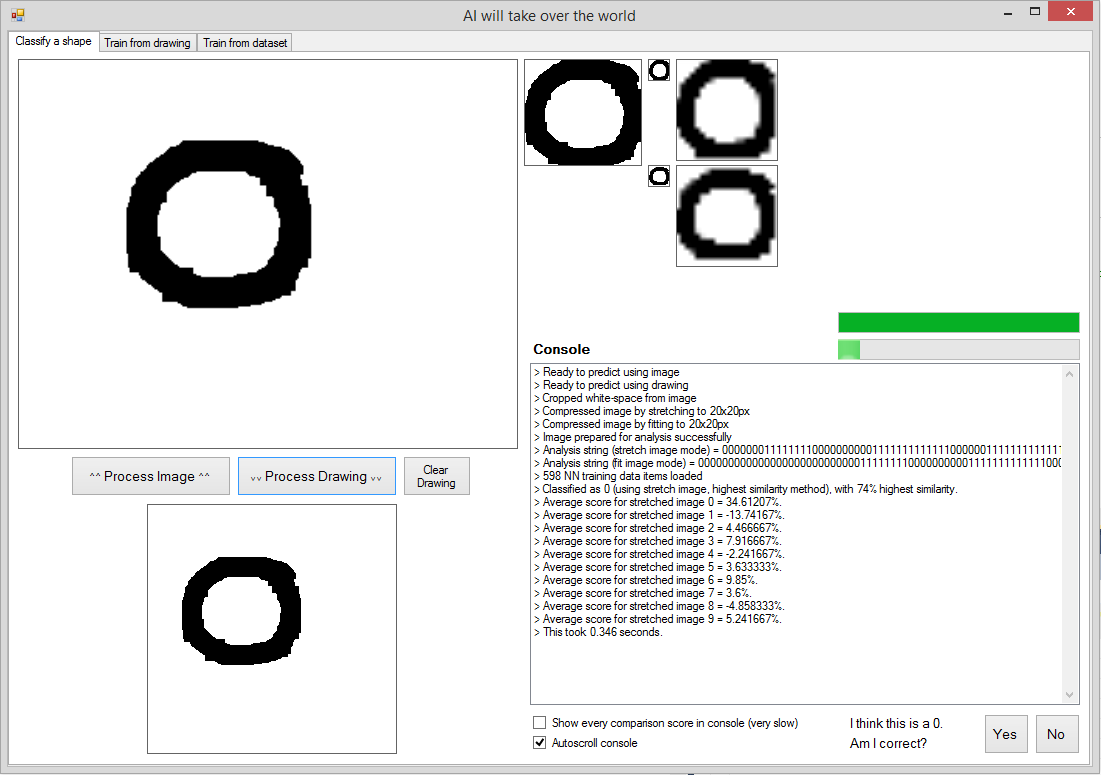
**The speed of classification can be improved by not converting an input image to black and white.**

Here are some screenshots of the console from the prototyping UI, with and without converting to black and white, showing the significant improvement in the speed of the algorithm completing:



Converted to black and white

Took 0.724 seconds



Took 0.346 seconds (less than half the time)

No conversion to black and white

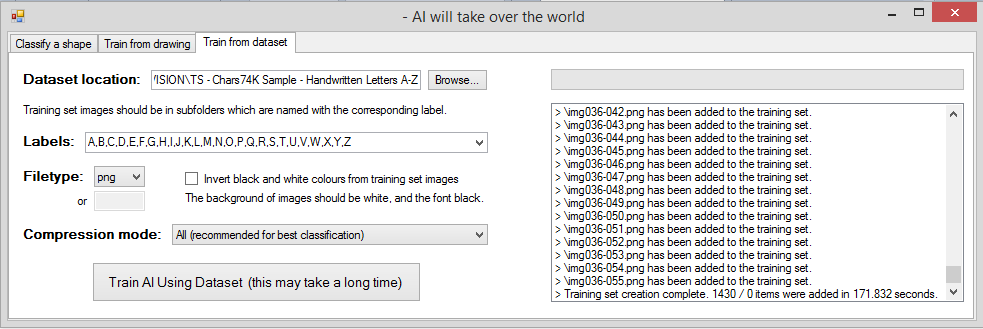
Converting to black and white is not necessary as canvas drawings (which will be the inputs from the real program for classification), as they will already be black-and-white.

### Creating fundamental AI training data-set text files:

This user interface, shown below, was developed as part of the prototyping of the AI algorithm, to create large data-sets of processed images for the AI to classify any written upper-case letter, and any written number.

This UI was only used once, since once the data-sets are created, containing training data for all letters and numbers, no more training will need to be done for these types of inputs.

The screenshot below shows the UI, being used to create a dataset of training data for classifying letters.



The code iterated through each image in the dataset location specified in the UI, processing it using the different methods explained in prototyping above, and storing a text-representation of the black-and-white 20x20 image, with one character for each pixel: 0 for white, 1 for black. Two strings of numbers, which I have named ‘pixel strings’ are stored for each training image: one for the image processed to stretch to a 20x20 image, and one for the image processed to fit to a padded 20x20 image. Here is a section of this dataset text-file:

labels:A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z  
1:A:00000000000000011000000000000000011111100000000000011111…  
2:A:00000000001100000000000000000111100000000000000011111100…  
3:A:00000001100000000000000000111100000000000000011111100000…

The numbers on each line continue for 801 characters - two 400 character pixel strings, separated by a colon.

### Classification AI must be able to be called as a function, returning a value:

In testing, the code for classification was in a button, and stored bitmaps for processing in PictureBoxes on a UI rather than in variables. This was advantageous when testing as I could see whether the code was functioning and correctly processing the bitmaps or not, but cannot be called from a function as the specification requires, so must be changed. I then transferred the code into a function and edited the code so no PictureBoxes were being used for data storage, as it will need to function without UI elements.

## Prototyping of the drawing canvas UI control

The following basic functionality will be needed, due to points in specification part III:

* A user should be able to draw a line by moving their mouse (or using a touchscreen). The line should be rendered following the path that the mouse takes.
* The drawn line should be smooth and clear.

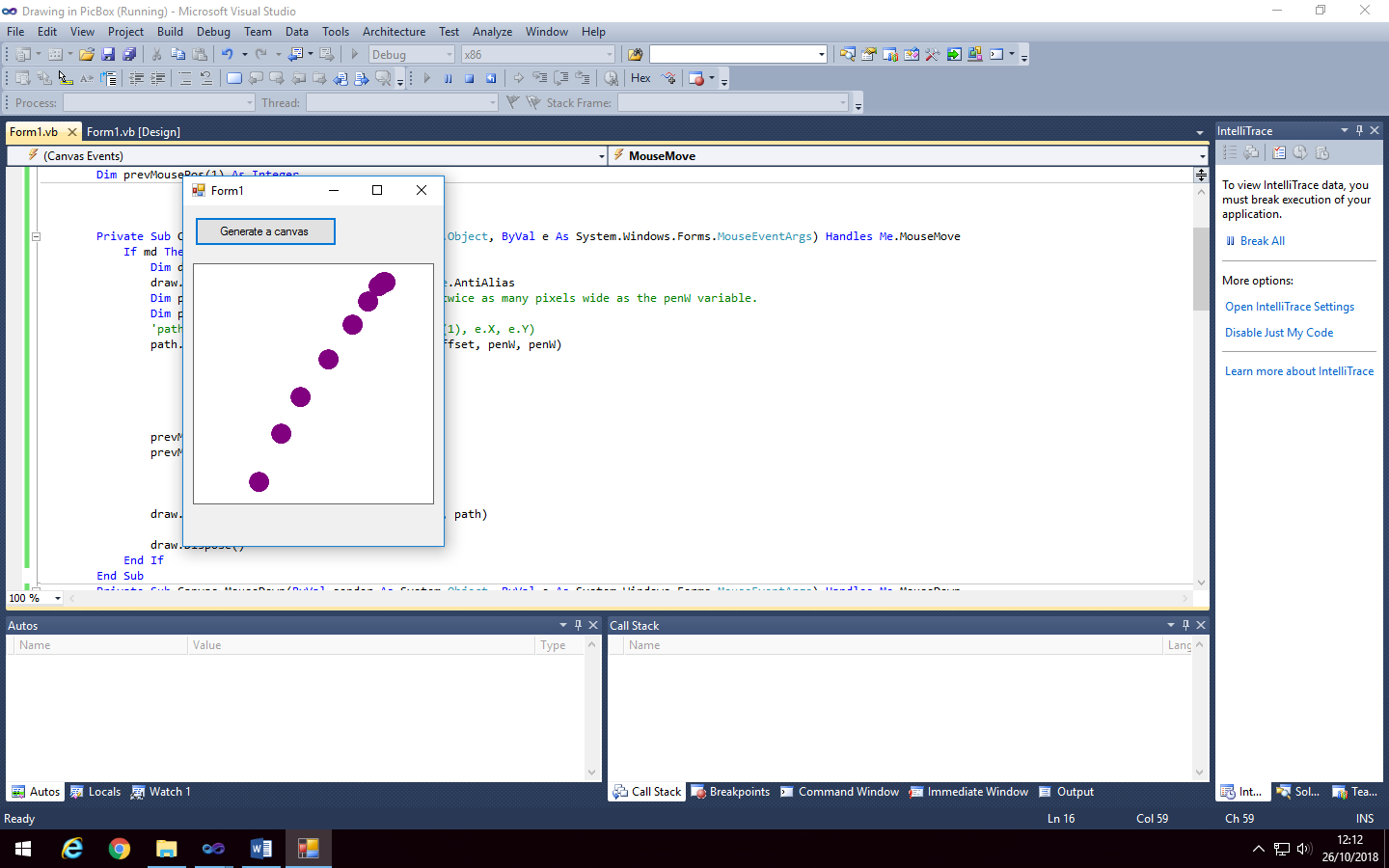
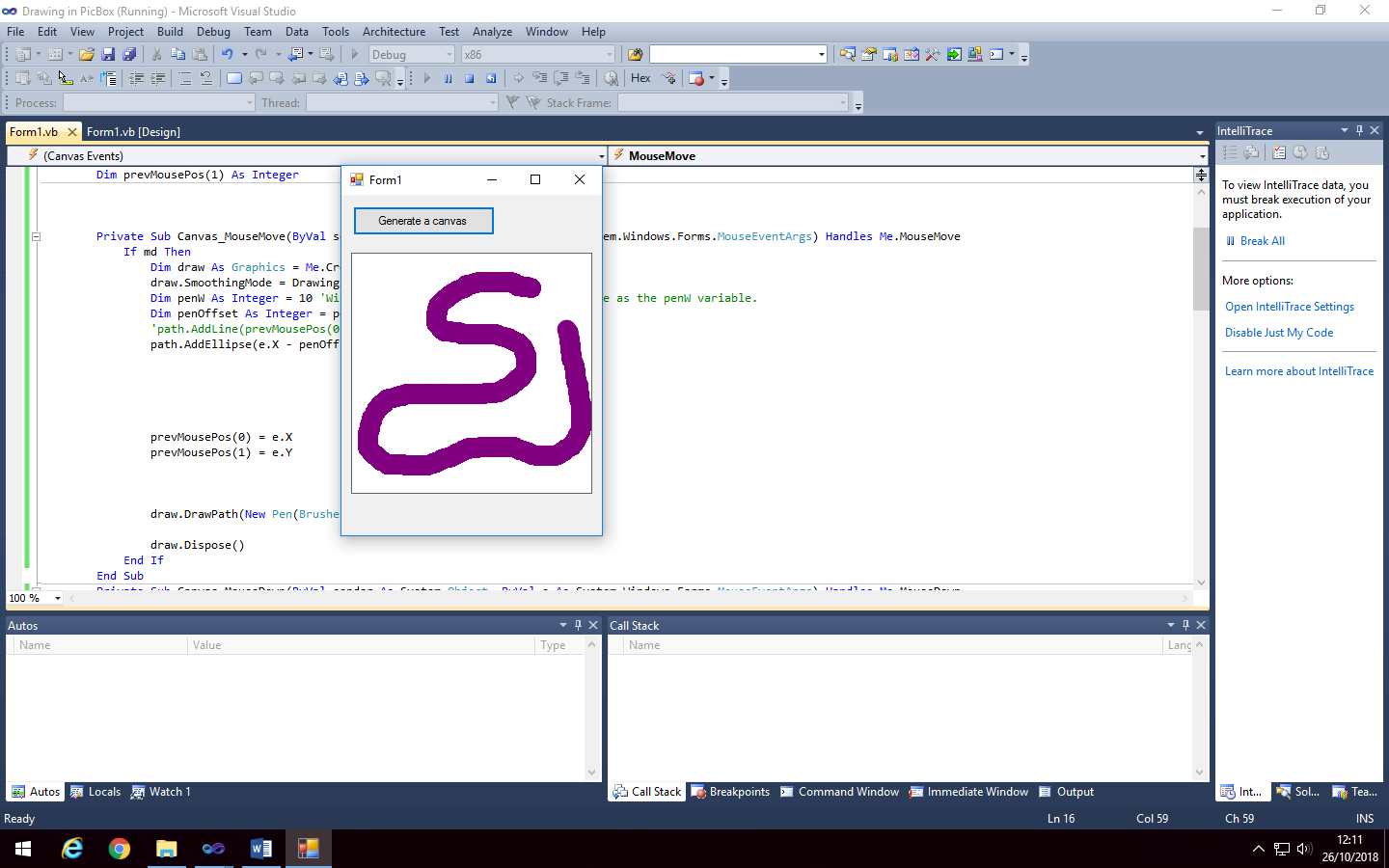
There is no native control within vb.net that allows a user to draw by dragging their mouse (or finger on a touchscreen) to create a line. Therefore, I will need to create a new class to provide such functionality.

I have named this class ‘Canvas’, and it inherits all functions and properties from the ‘PictureBox’ class, since picture-boxes have functionality for displaying graphics, which is similar to the outcome I am looking for with my canvas class (a user-drawn line is graphical). By doing this, I am essentially going to be making a duplicate of the ‘PictureBox’ class, but that can be drawn on.

### Rendering a line on the canvas which follows the mouse path:

I have used multiple methods of drawing a line that follows the mouse position in the development of the canvas control, until I found reached one which had no issues and fit the criteria of the specification. They are detailed across the next few pages.

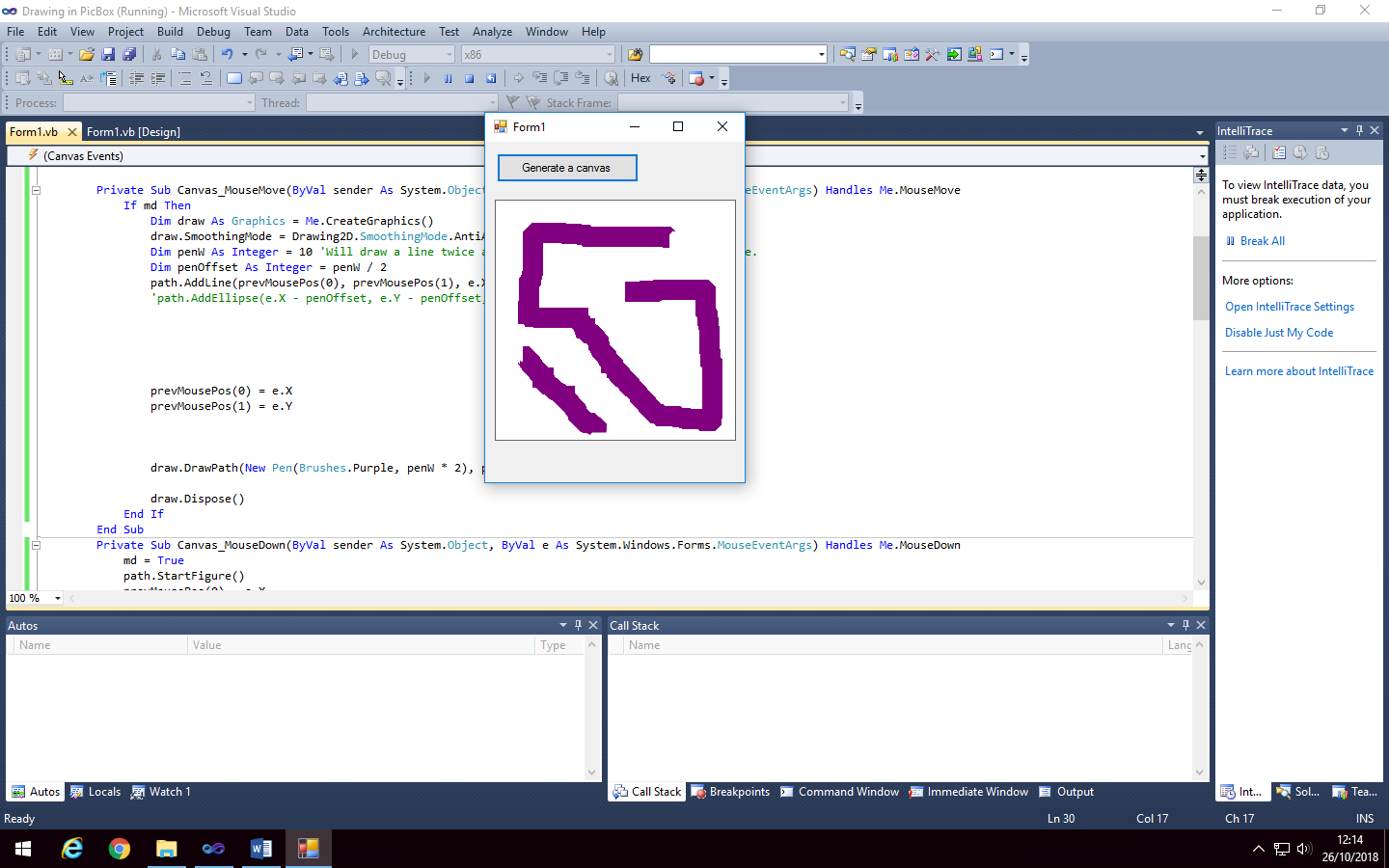
#### Method 1: Drawing a dot every time the mouse position changes:



This leaves gaps when the mouse is moved quickly, as there is a significant difference in the mouse position between each time the mouse position is updated.

#### Method 2: Using a joined path of points:

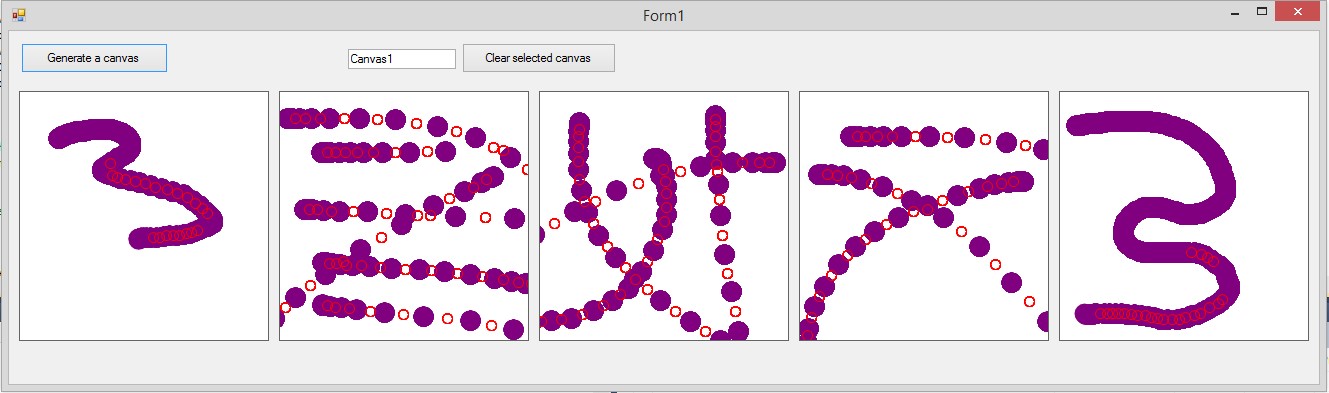
Why not just use a path of points, and use the vb function DrawPath to join the points without any custom functionality?



This results in artefacts with diagonal lines, and non-rounded ends of lines, since the points are joined using rectangular lines (non-rounded), and the rendered line can only be vertical or horizontal, so diagonal lines result in artefacts due to being drawn like a ‘zig-zag’.

#### Method 3: Drawing a line of dots, but with the gaps between mouse positions filled:

Instead, I will draw a dot for each queried mouse position, and if the distance between those positions is greater than a set value, another dot will be drawn between the previous and current mouse position. The distances will then be checked again, and if the distance is still not satisfactory, more dots will be drawn until the impression of a continuous line is given.

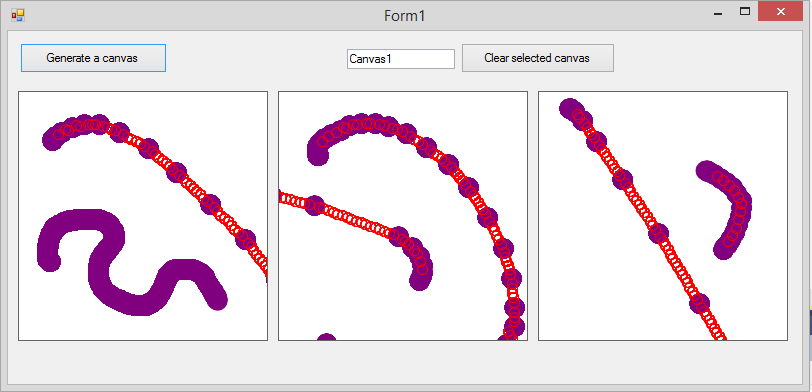


Shown above, another ellipse is added if the distance between two rendered ellipses is greater than 5 pixels. For testing purposes, the additional ellipses are smaller and in a red colour, to distinguish between them and the ellipses rendered at the mouse positions.

Canvases 2 through 4 show how the extra ellipses are placed exactly between the ellipses rendered at mouse positions.

Canvas 5 shows the sensitivity for extra ellipses to be rendered. It can be seen that the line is smooth until the red circles appear, where some jagged edges appear due to gaps in the rendered ellipses. This shows that the selected maximum distance between ellipses of 5 pixels is appropriate for a 20 pixel wide brush, therefore, for adaptability, the maximum distance allowed gap between ellipses will be set as **0.25 \* pen width**.

This increases the maximum number of ellipses that can be rendered from 60 to 120 per second (whilst using a 60hz display and a mouse with a minimum 60hz polling rate). This still does not create perfectly smooth lines without any gaps, however, so the code to add additional ellipses will be looped until the maximum distance between any ellipse is no more than 5 pixels, giving a theoretically infinite number of ellipses that can be rendered per second (of course limited by processing power and the speed of a user’s hand moving a mouse).

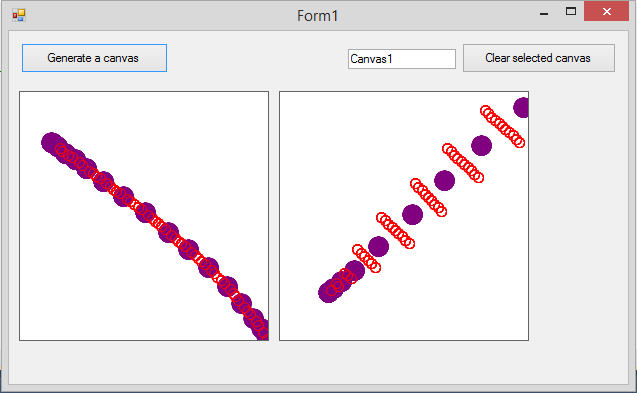


Shown above, the detailed changes have been made so more than one extra ellipse can be rendered between mouse positions, in order to give a smooth line with no gaps.

The long line in the right-most canvas shows how many ellipses are now rendered between mouse position ellipses. More ellipses are rendered if the gap is larger - this allows the code to be the most efficient, as there is no excess unneeded ellipses drawn.

The purple line with no extra ellipses on the left-most canvas, which was drawn slowly, has no extra ellipses rendered, since there are no gaps between the original ellipses at the mouse positions, and drawing extra ellipses would be inefficient and unnecessary.

**There is an error however:**



Dim newX, newY As Integer

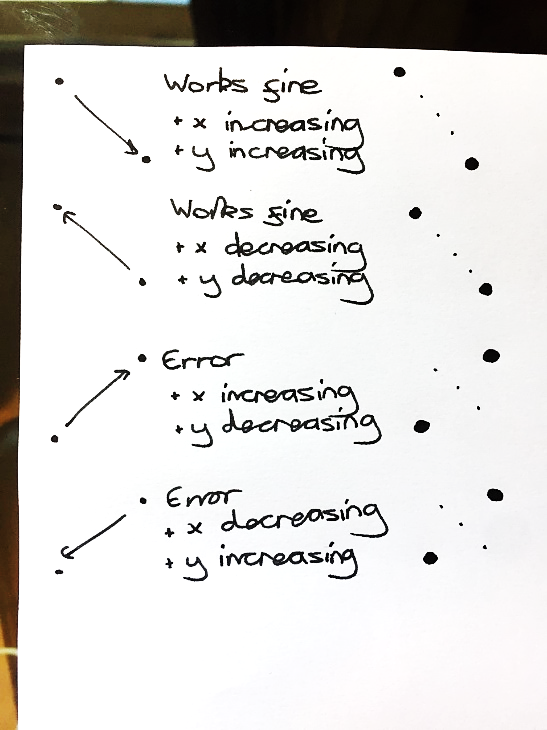
For Each n In offsets

newX = Xsml + (xdiff \* n) - penOffset

newY = Ysml + (ydiff \* n) - penOffset

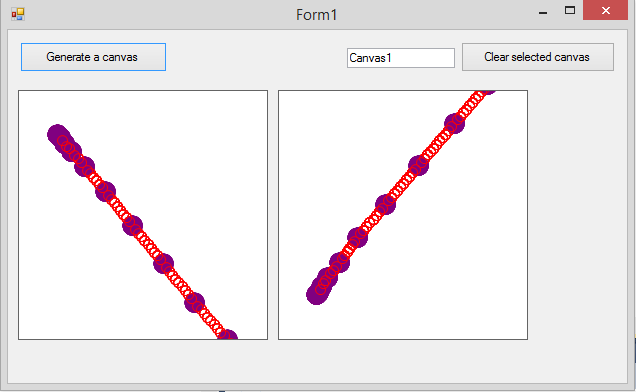
testpath.AddEllipse(newX, newY, penW, penW)

Next

In certain conditions, the extra ellipses which join the gap between mouse position ellipses are drawn in the wrong place.

I discovered the issue by looking at how conditions are changing in different situations, shown in the handwritten notes on the right. If the x and y coordinates of the mouse position are **not** both increasing or decreasing together (i.e. if one **is** increasing whilst one is decreasing), then the y and x coordinates of the extra ellipses are grouped incorrectly, resulting in them being rendered in the wrong positions (n.b. x and y begin at zero in the top-left corner).

This is fixed simply by reversing the order that y coordinates are grouped with the x coordinates, if both x and y mouse position are not increasing or decreasing together.



Dim newX, newY As Integer

For Each n In offsets

newX = Xsml + (xdiff \* n) - penOffset

If e.X >= prevMousePos.X And e.Y >= prevMousePos.Y Or e.X <= prevMousePos.X And e.Y <= prevMousePos.Y Then

newY = Ysml + (ydiff \* n) - penOffset

Else

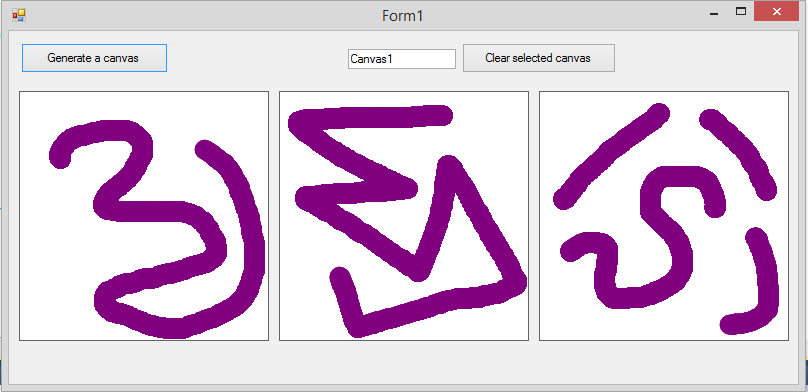
newY = Ylrg - (ydiff \* n) - penOffset

End If

testpath.AddEllipse(newX, newY, penW, penW)

Next

As shown in the image above, the issue has been fixed.



The ellipses which fill the gaps are now added to the same path of ellipses as the mouse position (instead of being rendered separately in red for testing purposes), so they appear identical.

This is better than using a path of points (method 2), as it rounds the corners and ends of drawn lines, and does not result in artefacts with diagonal lines.

The highest level of anti-aliasing of the line that the vb.net graphics function provides is used here, to render as smooth a line as possible.

n.b: Within the final technical solution, the drawing code has been modified slightly compared to the sections shown here, in order for the code to be able to be called from a different module to the one the canvas code is stored within.

## Research sources for program development

### Research for main student and teacher parts of system:

http://www.informit.com/articles/article.aspx?p=2036581&seqNum=9

https://docs.microsoft.com/en-us/dotnet/visual-basic/developing-apps/windows-forms/how-to-draw-shapes-with-the-ovalshape-and-rectangleshape-controls

http://vbcity.com/forums/t/28176.aspx

http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

https://stackoverflow.com/questions/2651682/resizing-a-two-dimensional-array

http://www.vbforums.com/showthread.php?727147-RESOLVED-Finding-max-value-in-array-without-looping-or-sorting-possible

https://www.experts-exchange.com/questions/28704265/VB-NET-Pie-Chart-Segment-Colours.html

https://www.dreamincode.net/forums/topic/347142-how-to-make-a-stacked-bar/

https://stackoverflow.com/questions/7745609/sql-select-only-rows-with-max-value-on-a-column

### Teacher login research:

https://stackoverflow.com/questions/10924238/encrypting-to-sha1-visual-basic-vb-2010

https://en.wikipedia.org/wiki/Salt\_(cryptography)

https://accounts.google.com/signin (for design research)

# Technical Solution

The entire source code for the technical solution of the project has been attached as an appendix alongside the main documentation, due to the large quantity of code. Later in the documentation of the technical solution, I will include a list of page references to the most complex and interesting parts of the code.

The technical solution is in the form of one executable program, with the teacher-side of the system being accessed as a branch from the student-login - this is how I planned, which can be seen in the high-level overview of the program’s structure, in the system designs. All of the student-side user interfaces, apart from login, are separated into panels within one main Windows form (named ‘StuMain’, within the code), for the speed, ease of use and user-friendliness reasons that have been previously explained. All of the teacher-side user interfaces, apart from login and AI training, are separated into panels, within one main Windows form (named ‘TeachMain’, within the code) - this is a separate Windows form to that that contains the teacher-side UI panels. AI training is in it’s own Windows form (‘Form\_Train’), to improve program load speed, as it will only need to be used irregularly so it does not make sense to open it with the other teacher-side parts of the program every time.

The code is separated into multiple modules within one VB solution, in order to be organised sensibly. These are:

* Form\_StuLogin.vb  
  Windows form object - code relating to the student login is located here.
* Form\_StuMain.vb  
  Windows form object - code relating to the rest of the student-side of the system is here.
* Form\_TeachLogin.vb  
  Windows form object - code relating to the teacher login is located here.
* Form\_TeachMain.vb  
  Windows form object - code relating to the rest of the teacher-side of the system is here.
* Form\_Train.vb  
  Windows form object - this is an additional form, which pops-up automatically when a teacher is creating a question and the AI needs training to recognise the answer to it. Code relating to this AI training UI is located here.
* Module\_SharedClasses.vb

Module object - this module is standalone and has no windows form associated with it, unlike the other Windows form objects above. This contains a variety of classes and subroutines, which are made accessible in the other modules by importing this module. These classes and subroutines are code which is used by multiple parts of the program, such as all code relating to the AI algorithm, and my custom-made canvas class.

Storing classes and subroutines which are used multiple times throughout the program in the separate Module\_SharedClasses module has various benefits:

* It reduces the file-size of the program by not having identical code duplicated between modules.
* It makes the program far easier to maintain in the future, since, in order to change something within the program, code will not be duplicated so any changes will only need to be made once.
* The code is more organised, as any shared class can be found in this one module, rather than being spread across all modules.

The annotated source code, as explained, is attached as an appendix. Here is the list of references to points of interest within the code:

Explanation of these parts of code has been done within the annotations of the source code.

|  |  |
| --- | --- |
| Section of Code | Location in Source Code Document |
| The canvas drawing class, a custom class (inheriting the VB PictureBox class) which allows users to draw on a canvas UI object which can be dynamically generated. In-depth explanation can be found in the source code annotations. | Module\_SharedClasses, page 61, lines 495-753. |
| The artificial intelligence class, which is a complex scientific model used for classifying an input bitmap of a drawn character, with a string-type label. The AI character classification algorithm is based on nearest-neighbour classification methodology. This involves many recursive algorithms and accessing data-sets stored in text-files. The AI class contains two public subroutines, one for classifying input bitmaps, and one for training the AI to recognise new characters. | Module\_SharedClasses, page 66, lines 760-1406. |
| The VB SAPI text-to-speech function has been utilised to speak questions out-loud to a user, as well as being able to read them from the screen. Multi-threading is used here, to execute the SAPI speech function on a separate thread to the main system, to prevent the system from hanging. | Module\_SharedClasses, page 60, lines 423-435. |
| The widely-used subroutines for reading from and writing to a database. The database which is used by the system contains 13 tables, and many cross-tabular parameterised SQL statements are executed through these subroutines (these will be pointed to later). | Module\_SharedClasses, page 61, lines 440-488. |
| Teacher passwords are hashed using the SHA1 hashing algorithm, to be stored securely in the database. The passwords are also salted, to prevent the use of rainbow tables to reverse password hashes. | Form\_TeachLogin, page 6, lines 8-27 & 31-41. |
| I have developed a sorting algorithm which outputs the order of numerical integer array elements of an input array. This is used within the nearest neighbour AI character classification algorithm to order the scores of the possible labels, calculated by the algorithm. Further explanation can be found in the annotations surrounding the code. | Module\_SharedClasses, page 69, lines 986-1040. |
| The SuperListBox class is a replacement for the VB list-box UI control. It functions in a similar way to the VB list-box – items can be added to the SuperListBox to be displayed. However, it allows up to 4 lines of text per item, including a larger title-text line, and an action button for each item. This class inherits from the VB panel class. Further explanation can be found in the source code annotations. | Module\_SharedClasses, page 54, lines 41-269. |
| Here is a section of code where a SuperListBox is being set up to display assignments which a student has attempted in the past. This is called when the StuMain form first launches. | Form\_StuMain, page 13, lines 320-403. |
| The selector-button class is a custom class inheriting from the VB button class. It is similar to a button, but has a toggle function, allowing it to be in an ‘on’ or ‘off’ state, toggled by being clicked. It works similarly to the VB check-box UI control, but is far more suitable for touchscreens which the system will be used with, due to having a larger area to click. | Module\_SharedClasses, page 54, lines 16-36. |
| Selector-buttons (mentioned in the section of code for the selector-button class above) are dynamically generated in the user-interface by this section of code. All assignment topics are read from the database, and one selector-button is dynamically generated for each question topics. This is done so that the user can choose which topics to see questions of, in a randomly generated quiz from the ‘practice mode’ panel of the student-side of the system. | Form\_StuMain, page 10, lines 59-92. |
| Graphs displayed in the UI, including bar charts and pie charts, have been used within the system to display statistics to teachers and students. This section of code is for setting up and displaying a bar chart and a pie chart in the ‘Your Classes Assignments’ section of the teacher-side of the program. | Form\_TeachMain, page 36, lines 847-965. |
| The code for the quiz system, which displays assignments from the database and randomly generated quizzes to student users, can be found in this section of code. This includes code for reading questions from the database for an assignment, randomly generating a quiz, dynamically displaying canvas UI controls, marking questions, giving feedback, and repeating a quiz identically. Each function of the quiz system is separated into different subroutines, each labelled within the code. | Form\_StuMain, page 17, lines 546-869. |
| This section of code includes three cross-tabular parametrised aggregate SQL statements for reading assignments from the database that a student has attempted in the past, and assignments they have yet to attempt. The first SQL statement reads all assignments from the database that a student has not yet attempted, but has been assigned. The second SQL statement reads all past attempts of assignments to their own separate row. The third SQL statement reads all past attempts of assignments, with all attempts of the same assignment grouped together. In depth explanation of this final SQL statement can be seen in the ‘design of core SQL statements’ section of the system designs. | Form\_StuMain, page 12, lines 197-254. |
| More SQL code, using many different functions including COUNT, MAX, GROUP BY, ORDER BY, DELETE, UPDATE and INSERT INTO, can be found throughout the source code of the technical solution, primarily within the Form\_StuMain and Form\_TeachMain modules. SQL statements involving all of the above functions can be seen in the ‘design of core SQL statements’ section of the designs. | N/A |
| When a question is displayed to the student-user in a quiz, the required number of canvases for a student to answer the current quiz question with is calculated. Afterwards, the calculated number of canvases are dynamically generated in the user-interface. | Form\_StuMain, page 20, lines 721-747. |
| I have used defensive programming techniques throughout the technical solution of the system. Here is one example, wherein before executing the code for processing and storing an assignment which a teacher-user has created, checks are done to ensure that the necessary user-inputs have been made to store the assignment successfully without missing data. If not, messages are displayed to the user explaining what they must do, and the subroutine is exited before the remainder of the code can execute. | Form\_TeachMain, page 30, lines 481-502. |
| Here is another example of defensive programming techniques used in the system. In this section of code, and many other similar sections of code, before reading data from the database to a data-table of a data-set, the table which data will be read to is checked if it exists, and if so, is deleted. This prevents data being duplicated, if data was read twice without clearing the table. Two examples of this are referenced in the location within the source code, in the right hand column. | Form\_StuMain, page 25, lines 151 & 175. |
| When teachers are creating questions for a quiz, data validation is used to check that the user-inputted correct answer for a question being created only contains allowed characters. These allowed characters vary based on question topic, and are retrieved from the database when the question topic is chosen. | Form\_TeachMain, page 48, lines 1572-1583. |

This list of sections of code is not entirely exhaustive, however it does bookmark all significant sections of the technical solution.

# Testing of the Technical Solution

## Introduction

The following table will be used as a template for all my tests:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: The reason the test is being done. | | | | | |
| Test No. | **Test Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| e.g. 1 | N, B, or E, where:  N = normal  B = boundary  E = erroneous | The data being inputted to the system. | The expected returned result from the system. | The actual returned result from the system. | **✔**  or  **X** |

.  
Evidence will then be appended below each testing table.

I will test part of the technical solution to ensure that it functions without error. I will then, within my evaluation, be able to state that the technical solution meets each point of the specification, in a fully functional manner, with reference to my testing which I will complete.

Boundary and erroneous tests will be conducted where appropriate. Since the system has minimal instances of user input that is not validated by the nature of the input (i.e. topics for a randomly generated quiz are chosen by the user using dynamically generated buttons, instead of a text-input, for example), there will be many situations where there is no possible boundary or erroneous data that could be inputted.

## Test Plan

The following test plan details the tests that will be required in order to test that the technical solution meets the specification, and is fully functional.

### Tests for the student and teacher log in systems:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 1 | Normal | Test that student login functions correctly with a valid input of initials. Also tests AI character classification performance. | I F | “Are you Isabella Forsyth?” |
| 2 | Normal | “” | H B | “Are you Holly Burton?” |
| 3 | Boundary | Test that student login functions correctly with a valid input of initials, but initials do not correspond to a student. | R W | “RW is not recognised. Try again.” Followed by the canvases being automatically cleared. |
| 4 | Boundary | Test that the student login works correctly with inputted initials that correspond to multiple students. | A C | “Are you Amelia Capewell?”  *Input: No*  “Are you Alice Crossley?” |
| 5 | Erroneous | Test that the system correctly deals with an invalid input, that is not alphabetical. | 3 ! | “No matching student found.” |
| 6 | Normal | Test that the teacher login accepts a valid username and password. | U: “jessicabishop”  P: “password” | Logs in to system as Jessica Bishop. |
| 7 | Normal | Test that the teacher login does not allow login with an incorrect password. | U: “jessicabishop”  P: “hello” | “Incorrect username or password” |
| 8 | Erroneous | Test that the teacher login does not crash when ‘log in’ is clicked with no inputted username or password. | No input for username or password. | “Incorrect username or password” |

### Tests for the student-side of the system:

### Tests for the SuperListBox displaying incomplete assignments in student home panel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 9 | Normal | Test that the SuperListBox in the student-side of the system, which displays incomplete assignments, shows assignments correctly. | Logged in as Amelia Capewell, database contains 3 outstanding assignments. | 3 assignments displayed, as per evidence screenshot of database. |
| 10 | Normal | Test that the SuperListBox in the student-side of the system, which displays incomplete assignments, shows assignment due dates and an assignment’s creator correctly. | Logged in as Isabella Forsyth, database contains 1 outstanding assignment. | Correct details displayed, as per evidence screenshot of database. |
| 11 | Normal | Test that the SuperListBox in the student-side of the system, which displays incomplete assignments, correctly displays a message when a student has no outstanding assignments. | Logged in as Isabella Forsyth, database contains no outstanding assignments. | “Nothing to see here!” message displayed. |
| 12 | Normal | Test that the action button for a SuperListBox item, in the SuperListBox which displays incomplete assignments, starts the correct quiz for the assignment the button is for, when clicked. | Logged in as Amelia Capewell, click button beside ‘Quick Shapes Quiz’ assignment on SuperListBox. | Begins a quiz, where first question displayed is the first question of the Quick Shapes Quiz. |

### Tests for the SuperListBox displaying past completed assignments in ‘Your Completed Assignments’ panel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 13 | Normal | Test that the SuperListBox in the ‘Your Past Assignments’ panel of the student-side of the system, which displays past assignments a student has attempted, is empty when no assignments have yet been done. | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’ button to change panel. | Empty SuperListBox. |
| 14 | Normal | Test that a completed assignment, once attempted, is displayed in the SuperListBox on the ‘Your Past Assignments’ panel, which displays past assignments a student has attempted. | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’ button to change panel, having completed the ‘Quick Shapes Quiz’ assignment, and no other assignments. | One assignment displayed – ‘Quick Shapes Quiz’, with correct details as per evidence screenshot of database. |
| 15 | Normal | Test that the action button for a SuperListBox item, in the SuperListBox which displays past attempts on assignments, starts the correct quiz for the assignment the button is for, when clicked. | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’, click button beside ‘Quick Shapes Quiz’ past assignment on SuperListBox. | Begins a quiz, where first question displayed is the first question of the Quick Shapes Quiz. |
| 16 | Normal | Test that the second attempt of the assignment from test 15, is displayed in the SuperListBox on the ‘Your Past Assignments’ panel, separately from the first attempt. | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’, ‘Group Assignments’ checkbox is **unchecked**. | Two attempts of the assignment ‘Quick Shapes Quiz’ displayed. |
| 17 | Normal | Following on from test 16, this tests that, when the ‘Group Assignments’ checkbox is checked, the two attempts of the same assignment which are displayed in the SuperListBox are grouped. | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’, ‘Group Assignments’ checkbox is **checked**. | ‘Quick Shapes Quiz’ assignment displayed only, with 2 attempts grouped as one item. |

### Tests for the set-up of a randomly generated quiz in ‘Practice Topics’:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 18 | Normal | Test that the selector-buttons for choosing question topics for a randomly generated quiz are all generated. | Student logged in, click ‘Practice Topics’ button. | A button to be present for each topic in the database evidence screenshot. |
| 19 | Normal | Test that, when clicked, selector-buttons will toggle to ‘on’ (i.e. border will be highlighted in blue), and when clicked again once highlighted, will toggle to ‘off’. | Student logged in, click ‘Practice Topics’ button. Click a selector-button once. Click it again after. | Selector-button should highlight when clicked. It should go back to the default black border when clicked again. |
| 20 | Normal | Test that multiple selector-buttons can be in their ‘on’ state at once (i.e. have a highlighted blue border). | Student logged in, click ‘Practice Topics’ button. Click 3 different selector-buttons once. | The three clicked selector-buttons should be highlighted. |
| 21 | Erroneous | Test that a randomly generated quiz will not start when the quiz has been set to have 0 questions. | Student logged in, click ‘Practice Topics’ button, attempt to set number of questions to ‘0’, by clicking the number-down button. | Not possible to lower number of questions below ‘1’. |
| 22 | Erroneous | Test that a randomly generated quiz will not start when no topics have been chosen using the selector-buttons. | Student logged in, click ‘Practice Topics’ button, no topic buttons toggled to ‘on’ state. | Message shown to user, informing them to choose at least one topic to practice. |
| 23 | Normal | Test that a randomly generated quiz is created correctly, with questions only of the topics chosen in ‘Practice Mode’, and with the correct number of questions chosen also. | Student logged in, click ‘Practice Topics’ button. Choose ‘shapes’ topic with 3 questions. Start the quiz. | Three questions of topic ‘shapes’ should be displayed in a quiz one-by-one. |
| 24 | Normal | Another test, to test that a randomly generated quiz is created correctly, with questions only of the topics chosen in ‘Practice Mode’, and with the correct number of questions chosen also. | Student logged in, click ‘Practice Topics’ button. Choose ‘written numbers’ and ‘addition’ topics with 4 questions. Start the quiz. | Four questions, of topics ‘written numbers’ and ‘addition’ should be displayed in a quiz, one-by-one. |

### General tests of the quiz system:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 25 | Normal | Test that the correct number of canvases are dynamically generated for a student to answer a question with. | Question on ‘shapes’ topic displayed. | One canvas should be displayed for drawing of shape. |
| 26 | Normal | Test that the correct number of canvases are dynamically generated for a student to answer a question with. | Question on ‘written numbers’ topic displayed, with a three-digit answer. | Three canvases should be displayed - one for each digit. |
| 27 | Normal | Test that the correct number of canvases are dynamically generated for a student to answer a question with. | Question on ‘written numbers’ topic displayed, with a single-digit answer. | One canvas should be displayed, as the answer is a single digit. |
| 28 | Normal | Test that the score of correct answers that a student achieves in a quiz is correctly counted and displayed as feedback after a quiz is finished. Test that correct emotion face is displayed. | Start a randomly generated quiz with 4 questions. Get 3 correct, 1 wrong. | Score of 3 out of 4 (i.e. 75%) displayed, along with a green emotion face. |
| 29 | Normal | Test that the score of correct answers that a student achieves in a quiz is correctly counted and displayed as feedback after a quiz is finished. Test that correct emotion face is displayed. | Start a randomly generated quiz with 4 questions. Get 1 correct, 3 wrong. | Score of 1 out of 4 (i.e. 25%) displayed, along with a red emotion face. |
| 30 | Normal | Test that repeat-quiz function repeats the previous quiz absolutely identically, and therefore functions correctly. | Start a randomly generated quiz with 3 questions. When finished, click ‘try again’ (aka. repeat quiz) button, and complete quiz. | 3 questions should be displayed. When the repeat-quiz button is clicked, the same 3 questions should be displayed, in the same order. |
| 31 | Normal | Test that the correct questions of the quiz which an assignment contains are displayed in the correct order. This will also test that only the questions for a quiz, and no others, are displayed when a quiz from an assignment is being completed. | Start a quiz (which has a specified order), and work through all questions. | The questions of the quiz, which will be shown in a database evidence screenshot, should be displayed in the correct order. |

### Tests of the accuracy of the AI question marking:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 32 | Normal | Test that the AI marks a question correctly, when the correct answer is drawn in the answer canvas(es), by classifying the bitmap from the canvas with the corresponding character drawn in the bitmap. | Question of ‘written numbers’ topic. Input correct answer. | “You correctly answered with ‘xxx’.” |
| 33 | Normal | Test that the AI marks a question correctly, when the correct answer is drawn in the answer canvas(es). | Question of ‘shapes’ topic. Input correct answer. | “You correctly answered with ‘xxx’.” |
| 34 | Normal | Test that the AI marks a question correctly, when the correct answer is drawn in the answer canvas(es). | Question of ‘subtraction’ topic. Input correct answer. | “You correctly answered with ‘xxx’.” |
| 35 | Normal | Test that the AI marks a question as incorrect, when an incorrect answer is drawn in the answer canvas(es). | Question of ‘shapes’ topic. Input incorrect answer of ‘xxx’. | “You incorrectly answered with ‘xxx’. The correct answer was ‘aaa’.” |
| 36 | Normal | Test that the AI marks a question as incorrect, when an incorrect answer is drawn in the answer canvas(es). | Question of ‘addition’ topic. Input incorrect answer of ‘xxx’. | “You incorrectly answered with ‘xxx’. The correct answer was ‘aaa’.” |
| 47 | Erroneous | Test that the system does not crash, and does mark a question as incorrect, when an answer of the wrong format is drawn in the answer canvas(es). If, for example, a shape is drawn in the answer canvases for a question where a numerical answer is expected (e.g. an addition question), the AI will not classify the answer as a shape, and will instead classify it as a number (even though it is not a number), since only numerical answers are expected so numbers are the only possible outputs for the AI classifier. | Question of ‘addition’ topic. Draw a square in all answer canvases. | “You incorrectly answered with ‘xxx’. The correct answer was ‘aaa’.”  …where ‘xxx’ is any random classification returned from the AI, within the expected input characters (0-9). |

### Tests for the feedback and graphs displayed to a student in ‘See Your Progress’ panel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 38 | Normal | Test that the graph of a breakdown of performance across question topics displays well, with many completed questions across topics. Also tests the dynamically generated feedback text, below the graph. This will simulate a user who has a couple hours usage of the system. | Logged in as James Bungay, clicked ‘See Your Progress’ button. | Graph should display with large bars completely on screen, and small bars still visible. Text feedback should be displayed - text for a topic to try should be for a topic that has no attempted questions yet. |
| 39 | Boundary | Test that the graph and dynamically generated feedback text do not have error in the way they display, for a new student user who has not answered a single question before. | Logged in as Thomas Kelly, clicked ‘See Your Progress’ button. | Graph of topics should display no bars. Text feedback should not be blank or have error, despite having no data to base feedback on (as user has done nothing). |
| 40 | Normal | Using the user from the previous test, test that, once some questions have been completed by a new user, this progress is displayed in the graph on the ‘See Your Progress’ panel. This will test that the red and green sections of bars in the topics graph, to represent correct and incorrect questions, display correctly. | With user from test above, do 5 practice questions in ‘shapes’ topic (3 correct, 2 wrong). Go to ‘See Your Progress’ panel. | A single bar should be visible on the topics graph, with a 3 unit wide green section, and a 2 unit wide red section. There should not be a bar beside other topics, as no questions will have been attempted for these topics. |

### Tests of the teacher-side of the system:

### Tests of the ‘Set an Assignment’ panel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 41 | Normal | Test that the search box for selecting students to set an assignment to works correctly. | ‘Amelia’ | Amelia Capewell  Amelia Clayton  Amelia Brindley |
| 42 | Boundary | Test that the search box for selecting students, is not case sensitive. | ‘smITH’ | Savannah Smith  Nico Smith |
| 43 | Boundary | Test that the search box will still return a student, if their first and last names are put in reverse order. | ‘Fields Billy’ | Billy Fields |
| 44 | Normal | Test that all quizzes available in the database are visible in the ‘Quiz to assign’ combo-box. | Click ‘Quiz to assign’ box, to see drop-down-list. | All quizzes shown in an evidence screenshot of the database should be visible, with no duplicates. |
| 45 | Normal | Test that, when a quiz is selected using the ‘Quick to assign’ combo-box, the contained questions appear in the list-box below. | Click ‘Quiz to assign’, click ‘Quick Shapes Test’. | The 5 questions in the ‘Quick Shapes Quiz’ should be visible with their answers, as per the evidence screenshot of the database. |
| 46 | Erroneous | Test that an assignment cannot be set with no students or class-groups chosen. | Select no students or class-groups, click ‘Set Assignment’. | Message: “At least one class-group or student must be chosen for the assignment to be set for.” |
| 47 | Erroneous | Test that an assignment cannot be set with the deadline in the past. | Set deadline to ‘1/1/1990’, click ‘Set Assignment’. | Message: “Assignment cannot be set in the past.” |
| 48 | Erroneous | Test that an assignment cannot be set, with no quiz chosen using the ‘Quiz to assign’ combo-box. | Don’t choose a quiz, click ‘Set Assignment’. | Message: “Please choose a quiz to assign.” |
| 49 | Normal | Test that an assignment can be set successfully, without the system crashing or showing an error. | Input valid assignment details, with ‘7L’ class-group chosen. | Message: “Assignment set”, and assignment has appeared in database. |
| 50 | Normal | Test that an assignment, when set to a student, appears in the student’s list of incomplete assignments. | Set ‘Addition Quiz’ to Daisy Kinsey. | Logged in as student ‘Daisy Kinsey’ prior to setting assignment - no assignments set.  Then: set ‘Addition Quiz’ to Daisy Kinsey.  Then: Log in as Daisy Kinsey, ‘Addition Quiz’ should be set. |
| 51 | Normal | Test that the ‘Addition Quiz’ assignment set in the above test, has not appeared for any other students, other than ‘Daisy Kinsey’ who it was set for. | Log in as student ‘Emily Davis’. | The ‘Addition Quiz’ assignment should not be present in their list of incomplete assignments. |
| 52 | Normal | Test that the assignment set to a whole class-group is visible to members of that class-group, and no other students. This will be done using the assignment set to ‘7L’ in test 49. | Log in as three different students: ‘Ethan Das’ & ‘Holly Burton’ of 7L, and ‘Esme Halford’ of 7C. | Assignment set in test 49 should be visible to both students of 7L (Ethan Das and Holly Burton). The assignment should not be visible to ‘Esme Halford’, as they are in 7C, which the assignment was not set for. |

### Tests of the ‘Build a Quiz’ panel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 53 | Normal | Tests that, when the ‘Add Question’ button is clicked to add a question to the quiz, the question appears, with its answer, in the list-box of questions below the ‘Add Question’ button. | Fill in question details of…  Topic: ‘Subtraction’, Question: ‘What is 11-9?’, Answer: ‘2’. Then click ‘Add Question’ button. | “What is 11-9? (answer: 2)” should appear in the questions list-box. |
| 54 | Normal | Test that quiz details successfully store in the database. | Quiz named: “Test quiz 1.” Add 2 questions of topic ‘addition’, and 1 question of topic ‘shapes’. Set to ‘In order’. Click the ‘Save Quiz’ button. | The details described in the test data section should appear in the database exactly as they were inputted. |
| 55 | Normal | Test that quiz details successfully store in the database. | Quiz named: “Test quiz 2.” Add 1 question of topic ‘multiplication’, and 3 question of topic ‘written numbers’. Set to ‘In a random order’. Click the ‘Save Quiz’ button. | The details described in the test data section should appear in the database exactly as they were inputted, with ‘0’ for questions numbers, as set to random order. |
| 56 | Normal | Test that, for a question topic without a limited number of possible answers (e.g. shapes, since there is no limit to number of shapes, but ‘addition’ would only have answers with digits 0-9), the ‘train AI’ panel appears, when a new answer is typed into the answer input-box. | Set question topic combo-box to ‘Shapes’. Write a question of ‘draw a diamond”. Write an answer of ‘Diamond’. Click ‘Add Question’ button. | The ‘Train AI’ window should appear when the ‘Add Question’ button is clicked. At the top of the window, the text “I don’t know what a diamond is yet” should be present. |
| 57 | Normal | Test that, for a question topic without a limited number of possible answers, the ‘train AI’ panel does not appear when an answer is chosen from the answer input-box (that is an option in the drop-down box as the AI already knows how to classify it), rather than a new answer being typed in. | Set question topic combo-box to ‘Shapes’. Write a question of ‘draw a square”. Select answer of ‘Square’ from the drop-down of the answer selection input-box. Click ‘Add Question’ button. | The ‘Train AI’ window should **not** appear when the ‘Add Question’ button is clicked. The question should immediately appear in the questions list-box below. |
| 58 | Erroneous | Test that a quiz cannot be saved by clicking the ‘Save Quiz’ button, without a quiz name inputted. | Add a question to the quiz, but leave quiz name blank. Click ‘Save Quiz’. | Message “Please check you have named your quiz and added at least one question to it” should appear. |
| 59 | Erroneous | Test that a quiz cannot be saved by clicking the ‘Save Quiz’ button, without any questions added to the quiz. | Set the quiz name to ‘Test quiz 3’. Do not add any questions to the quiz (questions list-box should be empty). Click ‘Save Quiz’. | Message “Please check you have named your quiz and added at least one question to it” should appear. |

### Tests of the AI training form:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 60 | Normal | Test that the ‘Save Drawing’ button is not enabled, until the user has drawn something on the canvas. | Draw line on canvas. | ‘Save Drawing’ button should be disabled before drawing. Once a line is drawn, the button should enable automatically. |
| 61 | Normal | Test that the ‘Clear Drawing’ button clears the canvas, and the ‘Save Drawing’ button is disabled automatically when the canvas is cleared to white. | Draw line on canvas. Click ‘Clear Drawing’. | Canvas should be plain white after the ‘Clear Drawing’ button is clicked. The ‘Save Drawing’ button should be disabled too. |
| 62 | Normal | Test that the correct label, which the AI is being trained to recognise, is displayed in the bold text at the top of the window. | Add ‘Shapes’ question with answer ‘Octagon’ in ‘Build a quiz’ panel. Train AI panel should appear. | The bold text at the top of the train AI window which appears should state “I don’t know what an octagon is yet”. |
| 63 | Normal | Test that pixel-strings generated from drawings in the canvas, along with the correct label, are successfully stored in the correct AI data-set text file for the relevant character type (i.e. if drawing a shape, stored in the “trainingset\_shapes.txt” text file. | Following on from the previous test, draw an octagon in the canvas. Click ‘Save Drawing’. Repeat two more times. | The AI training data-set text file for shapes (“trainingset\_shapes”) should have three additional lines appended to it, each with unique IDs, the label ‘octagon’ and two 400 character strings of 0’s and 1’s, separated by colons. The lines should not have been present prior to the three octagons being drawn. |
| 64 | Normal | Test that the panel stating that ‘enough drawings have been made for the AI to be trained with’ appears after 5 drawings have been drawn in the canvas and saved using the ‘Save Canvas’ button. | Following on from the previous test, draw an octagon and click ‘Save Drawing’ two more times. | After clicking the ‘Save Drawing’ button for the 5th time (2nd time in this test), the panel should appear, covering up the canvas and other buttons. |

### Tests of the ‘Your Classes’ Assignments’ panel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 65 | Normal | Test that all of the assignments set by the logged in teacher (and no assignment set by others) appear correctly in the assignment selection list-box, with correct details for each assignment. | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. | Assignment which have been set by Megan Taylor, and no others, should be displayed in the list-box, along with their due date and other details. These will be shown in an evidence screenshot of the database. |
| 66 | Erroneous | Test to see if multiple assignments can be selected at once using the assignment selection list-box, which would cause an error as only one assignment’s details can be displayed at once. | Click any assignment. Details should display. Attempt to click another assignment whilst holding ‘ctrl’ on keyboard. | When the second assignment is clicked whilst holding ‘ctrl’ it should be selected and details shown, and the first one de-selected. |
| 67 | Normal | Test that the details (such as quiz name, assignment note, due date) for the correct assignment display below the assignment selection list-box, when an assignment is selected. | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | The details which correspond to the selected assignment should appear. These can be confirmed using an evidence screenshot of the database. |
| 68 | Normal | Test that the correct scores that students have achieved in the assignment selected using the assignment selection list-box, are summarised correctly in the score-distribution graph. | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | Student scores for the selected assignment, which can be confirmed using an evidence screenshot of the database, should be displayed in the correct columns of the graph. |
| 69 | Normal | Test that the number of students who have completed the assignment selected using the assignment selection list-box, is summarised correctly in the completion pie-chart. | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | The number of students who have not completed, have completed and have completed the assignment late, which can be confirmed using an evidence screenshot of the database, should be displayed with correct proportions in the pie chart. |
| 70 | Normal | Test that the resolution of score distribution graph’s x-axis changes when the numeric up down input-box’s (which is labelled for setting x axis resolution) value is changed. | Whilst on “Your Classes’ Assignments” panel, with an assignment selected, reduce the value in the numeric-up-down box from the default of 5, to ‘2’. | The number of columns displayed in the score distribution graph should decrease from 5 columns, to 2 columns. |
| 71 | Erroneous | Test that it is not possible to set the resolution of the score distribution graph to zero, using the numeric-up-down input-box labelled for setting x axis resolution. | Click the down button of the numeric-up-down box to decrease value, until it will no longer decrease. Attempt to type in ‘0’ to the box. | Resolution will decrease to 1 from the default of 5, in steps of 1, when the down button is clicked. Will not decrease further. Will not be able to type ‘0’ into the input-box. |
| 72 | Normal | Test that a panel for each class-group, and one for other individual students, that an assignment is set to (which will each contain a data-grid-view of details, tested in the next test) appear when an assignment is selected. | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | A panel should appear for each class-group which the selected assignment has been set to, and a panel for other students independent of a class-group. There should not be a panel for any other class-groups. This can be confirmed using an evidence screenshot of the database. |
| 73 | Normal | Test that the correct students (only ones for the class-group which the containing panel of the DGV is for), with the correct scores and statistics, appear in a data-grid-view within the panels. | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | Each class-group panel should only contain students whose class-group is that which the panel is for (shown by the panel name). This can be confirmed using an evidence screenshot of the database. |
| There is no need to test the search box for assignments on this panel, as the same code is used as the search box tested by tests 44-46. | | | | |

### Tests of the ‘Manage Students’ panel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 74 | Normal | Test that only students within class-groups taught by the logged in teacher appear in the student selection list-box, when the ‘Show your classes only’ button is clicked. | Log in as teacher ‘William Hagues’. Click ‘show your classes only’ in ‘Manage Students’ panel. | Before clicking the button, all students should be visible in the list-box. After clicking, only students of 7N, the class-group that William Hagues teaches, should be visible. |
| 75 | Normal | Test that the correct details for a student appear, when a student is clicked in the student selection list-box. | Click ‘Ellie Reynolds - 7B’ in the student selection list-box. | The details for Ellie Reynolds appear. These will be confirmed to be correct in a screenshot of the database. |
| 76 | Normal | Test that when the “Edit Student’s Details” button is clicked with a student selected, the input-boxes containing student details are enabled for editing. | Select student ‘Edward Boyce - 7N’. Click ‘Edit Student’s Details’ button. | The six input-boxes for student details should enable (i.e. no longer be read-only). |
| 77 | Normal | Following on from the previous test, test that, after editing student details in the input-boxes and clicking the ‘Save Changes’ button, a student’s details are successfully updated in the database. | With ‘Edward Boyce’ selected: edit SEN Support to ‘No’, and change class group to ‘7S’. Click ‘Save Changes’ button, and click ‘Yes’ in message-box, to confirm. | Within the ‘Students’ table of the database, the row for ‘Edward Boyce’ should contain the updated values for SEN support and class-group, after save changes has been clicked. |
| 78 | Normal | Test that the ‘Delete Student’ button successfully deletes a student from the database. | Select student ‘Hue Walsh’, click ‘Delete Student’, click ‘Yes’ in confirmation message-box. | Before, there should be a row in the ‘Students’ table of the database for Hue Walsh. After, the row should have been removed. |
| 79 | Normal | Test that assignments completed by the student chosen in the student selection list-box are displayed in the “Student’s completed assignments” data-grid-view. | Select student ‘Isabella Forsyth’ (who completed two attempts of an assignment in test 15), in the student selection list-box. | One assignment should be displayed in the “Student’s completed assignments” data-grid-view. ‘Num. of attempts’ should be ‘2’, as Isabella Forsyth attempted the same assignment twice in test 15. |
| 80 | Erroneous | Following on from the previous test, test that assignments completed by students other than the one chosen in the student selection list-box, are not shown in the “Student’s completed assignments” data-grid-view. | Select student ‘Finlay Atkinson’ in the student selection list-box. | No assignments should be shown in the data-grid-view (as the student has not completed any assignments) - in particular, the assignments completed by Isabella Forsyth are not visible. |
| There is no need to test the search box for students on this panel, as the same code is used as the search box tested by tests 44-46. | | | | |
| There is no need to test the graph of a breakdown of scores across topics for the selected student on this panel, as the same function is used to populate the graph, as is used in the graph on the student-side of the system, tested by tests 41-43. | | | | |

### Other tests for the teacher-side of the system:

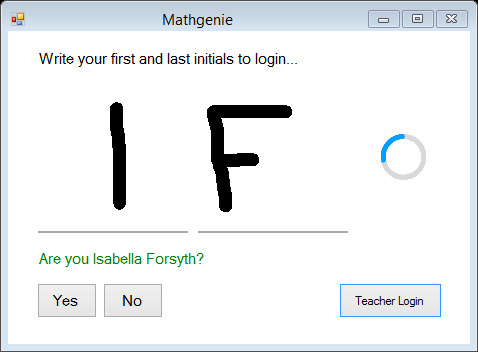
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Type | Test Purpose | Test Data | Expected Result |
| 81 | Normal | Test that a teacher’s name is displayed on the welcome text, on the teacher home panel. | Log in as teacher ‘Oliver Straw’. | “Welcome, Oliver Straw” displayed on the home panel. |
| 82 | Normal | Test that passwords are changed successfully using the ‘Change Password’ button in the menu bar of the teacher-side of the system. | Log in as ‘Dabinn Potts’, using ‘password’ as password. Change password to ‘hello’. Log in as ‘Dabinn Potts’ using ‘hello’ as password. | Password hash stored in the database changes, and the password used to log in changes. |

## Documented Testing

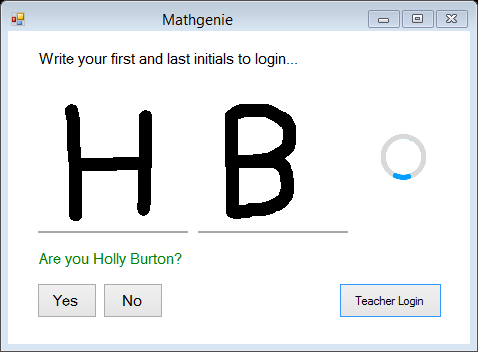
Each test which was set out in the test plan will now be conducted, and documented here. As with the test plan, tests for each section of the system will be grouped together.

### Tests for the student and teacher log in systems:

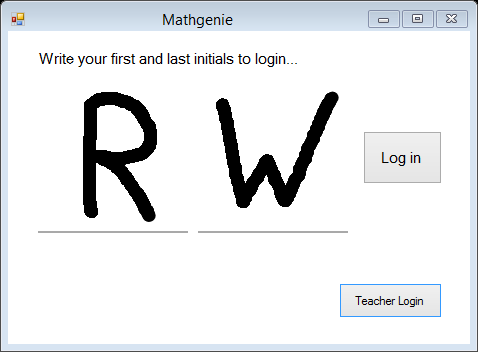
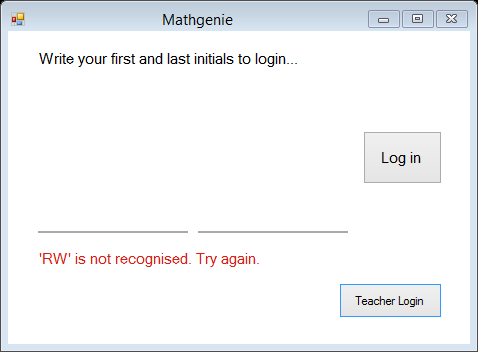
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that student login functions correctly with a valid input of initials. Also tests AI character classification performance. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 1 | Normal | I F | “Are you Isabella Forsyth?” | “Are you Isabella Forsyth?” | **✔** |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that student login functions correctly with a valid input of initials. Also tests AI character classification performance. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 2 | Normal | H B | “Are you Holly Burton?” | “Are you Holly Burton?” | **✔** |

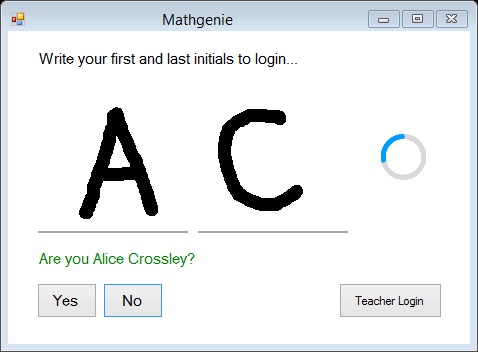


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that student login functions correctly with a valid input of initials, but initials do not correspond to a student. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 3 | Boundary | R W | “RW is not recognised. Try again.” Followed by the canvases being automatically cleared. | “RW is not recognised. Try again.” Canvases were automatically cleared. | **✔** |

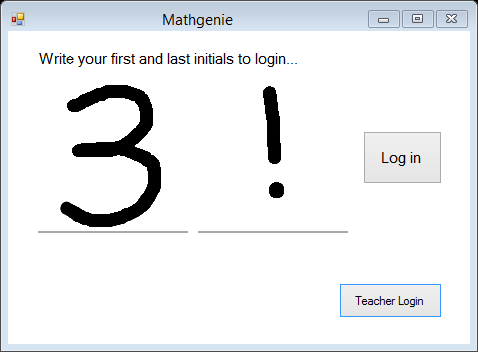
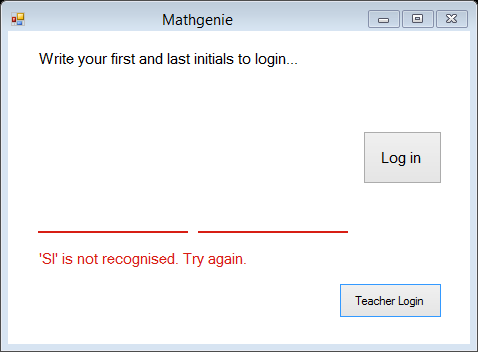
Clicked

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the student login works correctly with inputted initials that correspond to multiple students. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 4 | Boundary | A C | “Are you Amelia Capewell?”  *Input: No*  “Are you Alice Crossley?” | “Are you Amelia Capewell?”  *Input: No*  “Are you Alice Crossley?” | **✔** |

Clicked

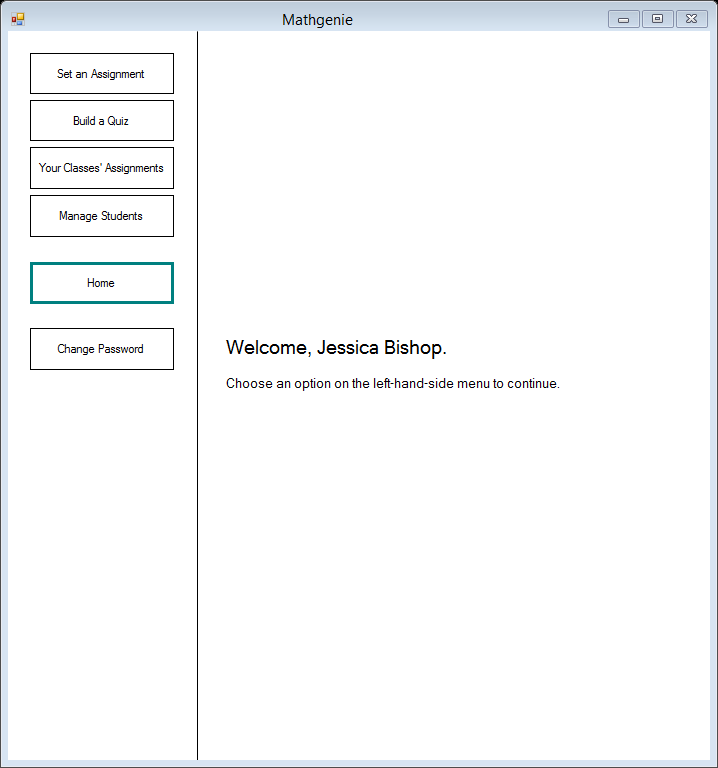
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the system correctly deals with an invalid input, that is not alphabetical. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 5 | Erroneous | 3 ! | Message stating there is no matching student found. | “SI is not recognised. Try again.” | **✔** |

Clicked

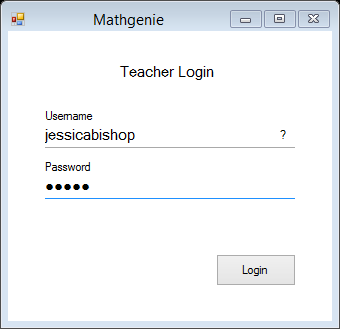
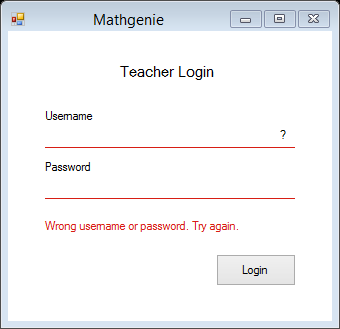
The AI algorithm classified the drawn characters ‘3 !’ as ‘S I’ because only alphabetical inputs are expected as the input for initials. Therefore, the only possible outputs for the AI to choose from are letters A-Z (i.e. ‘3’ and ‘!’ are not included). ‘S’ looks similar to ‘3’, and ‘I’ looks similar to ‘!’, hence the classification output of ‘SI’.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the teacher login accepts a valid username and password. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 6 | Normal | U: “jessicabishop”  P: “password” | Logs in to system as Jessica Bishop. | Logged in to system as Jessica Bishop - login form closes, and teacher-side of system opens. | **✔** |

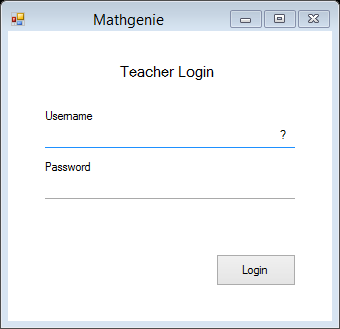
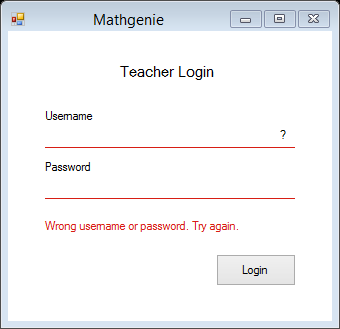
Clicked

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the teacher login does not allow login with an incorrect password. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 7 | Normal | U: “jessicabishop”  P: “hello” | “Incorrect username or password” | “Wrong username or password. Try again.” | **✔** |

Clicked

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the teacher login does not crash when ‘log in’ is clicked with no inputted username or password. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 8 | Erroneous | No input for username or password. | “Incorrect username or password” | “Wrong username or password. Try again.” | **✔** |

Clicked

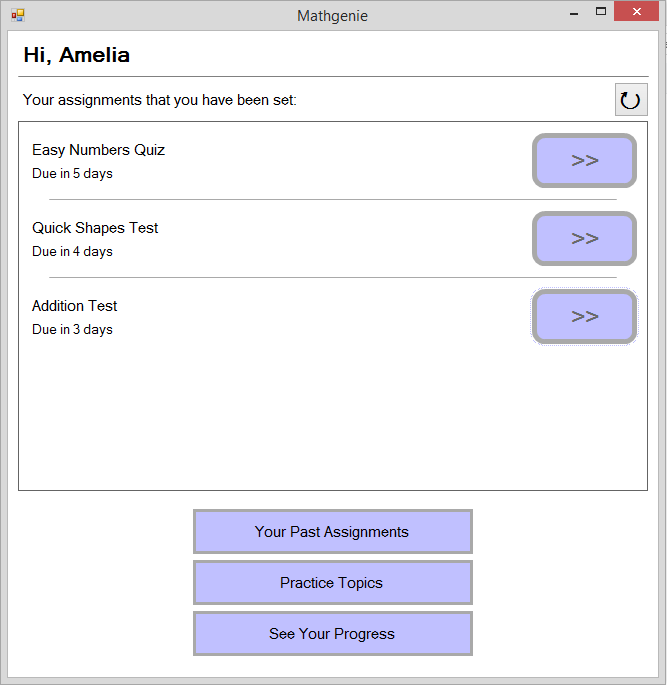
The “Wrong username or password. Try again” message was displayed, since a log in attempt to the system will have been made with a username and password that are both null. This will of course return no result from the database, and as such, the inputted username and password are incorrect.

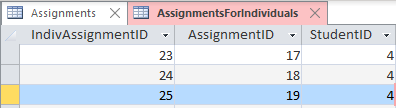
### Tests for the student-side of the system:

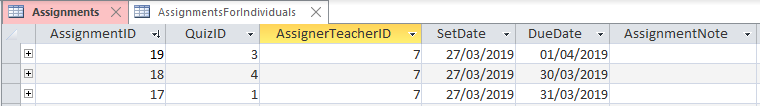
### Tests for the SuperListBox displaying incomplete assignments in student home panel:

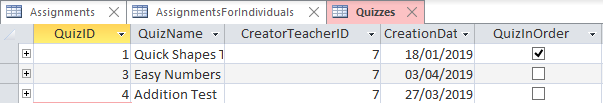
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the SuperListBox in the student-side of the system, which displays incomplete assignments, shows assignments correctly. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 9 | Normal | Logged in as Amelia Capewell, database contains 3 outstanding assignments. | 3 assignments displayed, as per evidence screenshot of database. | 3 assignments were displayed, as per evidence screenshot of database. | **✔** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the SuperListBox in the student-side of the system, which displays incomplete assignments, shows assignment due dates and assignment quiz names correctly. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 10 | Normal | Logged in as Amelia Capewell, database contains 3 outstanding assignments. | Correct details displayed, as per evidence screenshot of database. | Correct details were displayed, as per evidence screenshot of database. | **✔** |



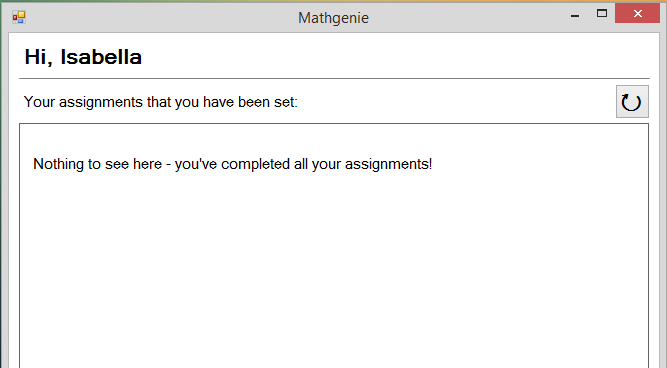




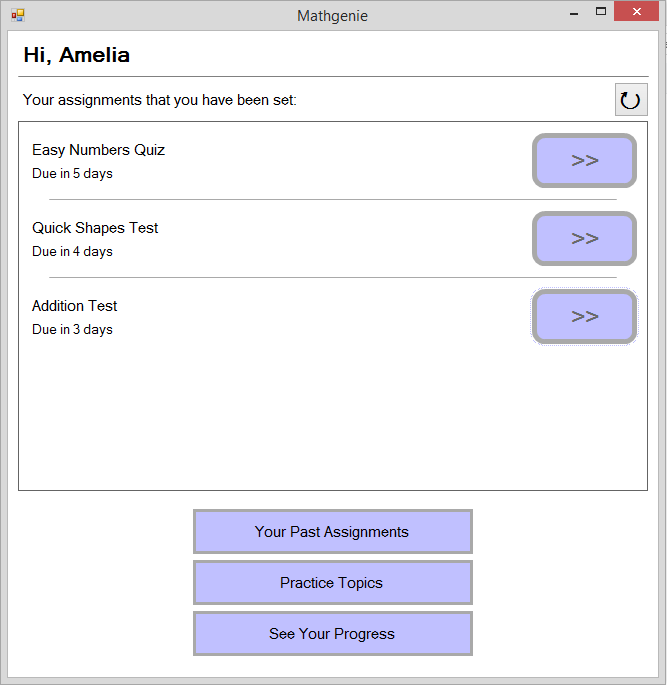


Each assignment shown in the second database screenshot above, has a QuizID. As can be seen in the third database screenshot, the QuizIDs each have a quiz name. The quiz names in the third screenshot correspond with the due dates on the second screenshot, and are associated correctly in the SuperListBox on the screenshot on the previous page. The first database screenshot shows that those three assignments have been set to the student with StudentID of 4, which is Amelia Capewell.

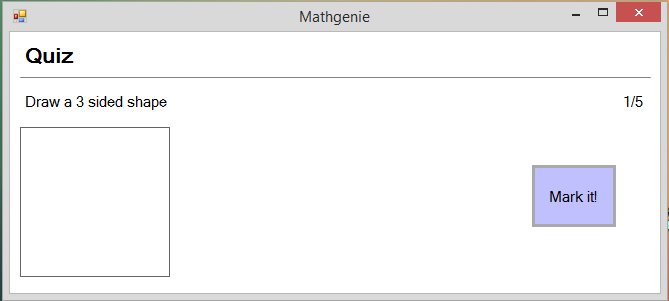
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the SuperListBox in the student-side of the system, which displays incomplete assignments, correctly displays a message when a student has no outstanding assignments. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 11 | Normal | Logged in as Isabella Forsyth, database contains no outstanding assignments for her. | “Nothing to see here” message displayed. | “Nothing to see here - you’ve completed all your assignments!” message was displayed. | **✔** |



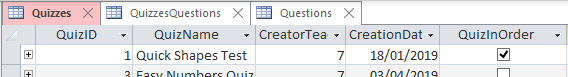
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the action button for a SuperListBox item, in the SuperListBox which displays incomplete assignments, starts the correct quiz for the assignment the button is for, when clicked. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 12 | Normal | Logged in as Amelia Capewell, click button beside ‘Quick Shapes Quiz’ assignment on SuperListBox. | Begins a quiz, where first question displayed is the first question of the Quick Shapes Quiz. | A quiz began, where first question displayed was the first question of the Quick Shapes Quiz. | **✔** |

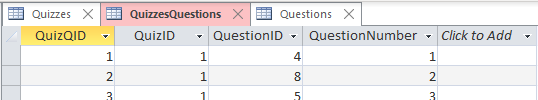


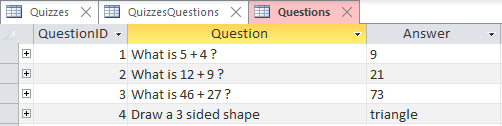
Clicked



This panel appears.

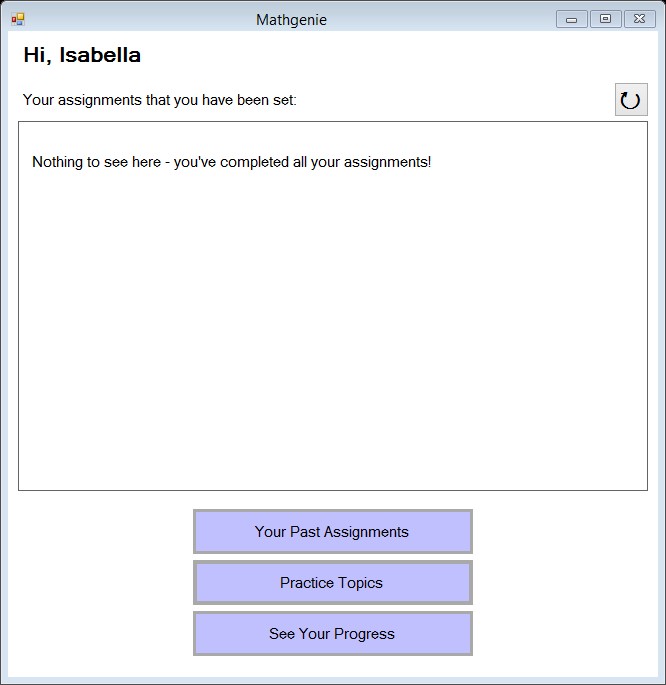




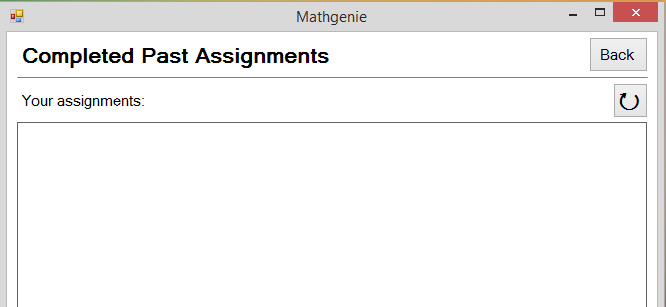
  
‘Draw a 3 sided shape’ is the first question in the quick shapes quiz, therefore the test is successful.

### Tests for the SuperListBox displaying past completed assignments in ‘Your Completed Assignments’ panel:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the SuperListBox in the ‘Your Past Assignments’ panel of the student-side of the system, which displays past assignments a student has attempted, is empty when no assignments have yet been done. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 13 | Normal | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’ button to change panel. | Empty SuperListBox. | SuperListBox was empty. | **✔** |

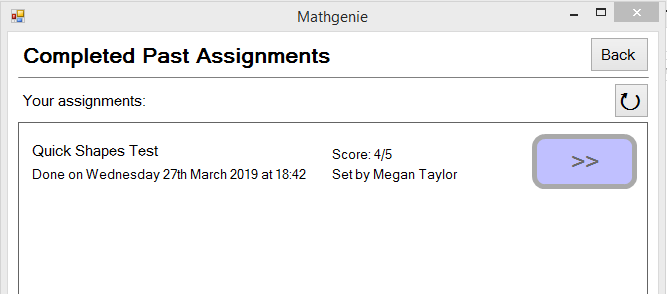


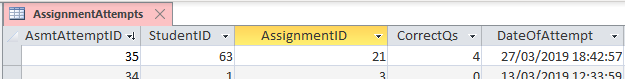
Clicked button, on student home panel.



SuperListBox is empty.

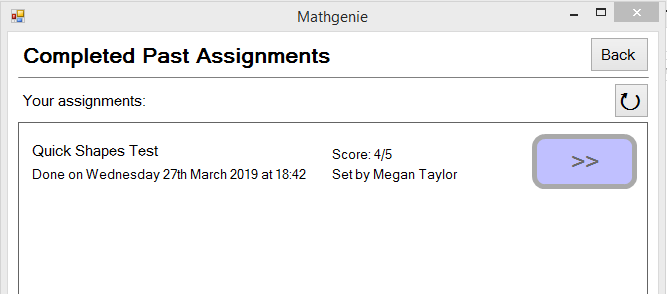
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a completed assignment, once attempted, is displayed in the SuperListBox on the ‘Your Past Assignments’ panel, which displays past assignments a student has attempted. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 14 | Normal | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’ button to change panel, having completed the ‘Quick Shapes Quiz’ assignment, and no other assignments. | One assignment displayed – ‘Quick Shapes Quiz’, with correct details as per evidence screenshot of database. | One assignment was displayed – ‘Quick Shapes Quiz’, with correct details displayed, as per evidence screenshot of database. | **✔** |



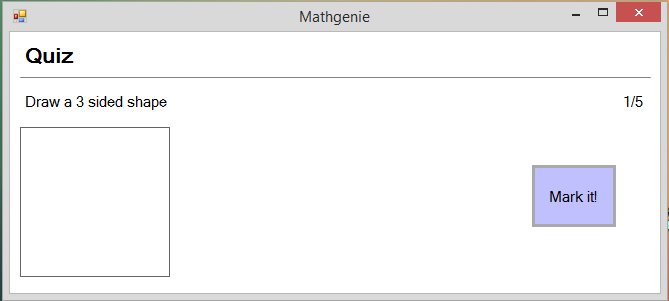


The details for the assignment displayed in the UI, are the same as those from the database. It has been shown in previous screenshots that AssignmentID ‘21’ is the assignment with the quiz ‘Quick Shapes Test’. Therefore, test 14 is successful.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the action button for a SuperListBox item, in the SuperListBox which displays past attempts on assignments, starts the correct quiz for the assignment the button is for, when clicked. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 15 | Normal | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’, click button beside ‘Quick Shapes Quiz’ past assignment on SuperListBox. | Begins a quiz, where first question displayed is the first question of the Quick Shapes Quiz. | A quiz began, where first question displayed was the first question of the Quick Shapes Quiz. | **✔** |

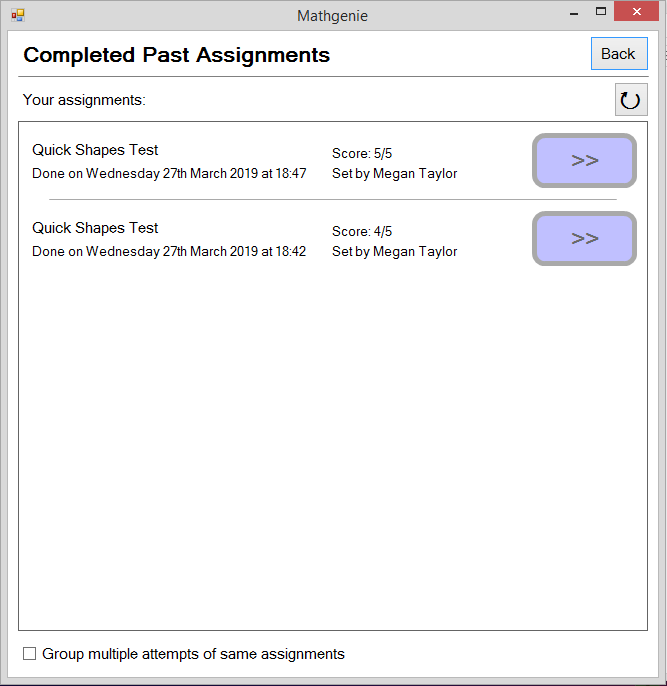


Clicked



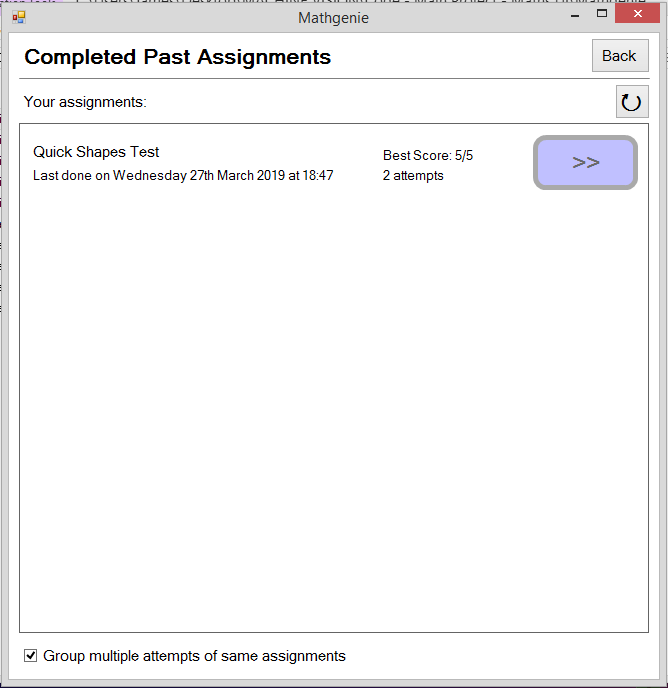
This panel appears, which as shown in test 12, is the first question of the Quick Shapes Test. Therefore, test successful.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the second attempt of the assignment from test 15, is displayed in the SuperListBox on the ‘Your Past Assignments’ panel, separately from the first attempt. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 16 | Normal | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’, ‘Group Assignments’ checkbox is **unchecked**. | Two attempts of the assignment ‘Quick Shapes Quiz’ displayed. | Two attempts of the assignment ‘Quick Shapes Quiz’ are displayed in the SuperListBox. | **✔** |



Unchecked

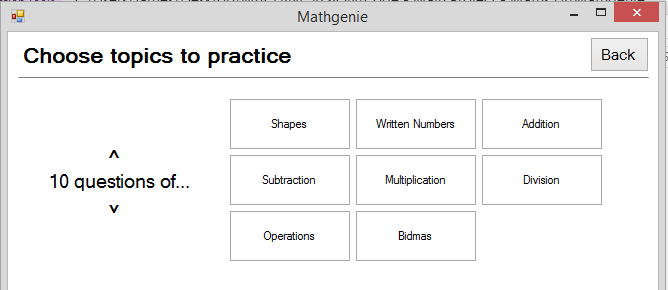
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Following on from test 16, this tests that, when the ‘Group Assignments’ checkbox is checked, the two attempts of the same assignment which are displayed in the SuperListBox are grouped. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 17 | Normal | Logged in as Isabella Forsyth, clicked ‘Your Past Assignments’, ‘Group Assignments’ checkbox is **checked**. | ‘Quick Shapes Quiz’ assignment displayed only, with 2 attempts grouped as one item. | ‘Quick Shapes Quiz’ assignment was the one item which was displayed, with the 2 attempts grouped as one item. | **✔** |

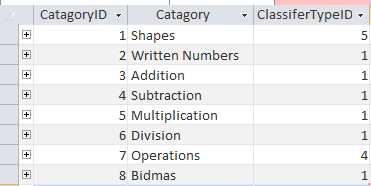


Checked

### Tests for the set-up of a randomly generated quiz in ‘Practice Topics’:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the selector-buttons for choosing question topics for a randomly generated quiz are all generated. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 18 | Normal | Student logged in, click ‘Practice Topics’ button. | A button to be present for each topic. | A button was present in the UI for each topic in the provided database evidence screenshot. | **✔** |



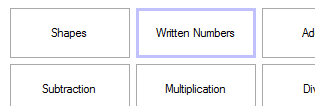
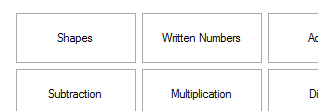


For each topic in the above screenshot of the database, a button has been dynamically generated in the UI panel.

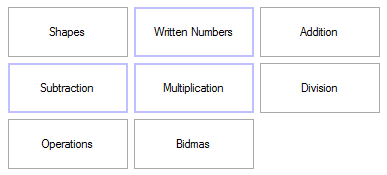
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that, when clicked, selector-buttons will toggle to ‘on’ (i.e. border will be highlighted in blue), and when clicked again once highlighted, will toggle to ‘off’. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 19 | Normal | Student logged in, click ‘Practice Topics’ button. Click a selector-button once. Click it again after. | Selector-button should highlight when clicked. It should go back to the default black border when clicked again. | Selector-button highlights when clicked. It goes back to the default black border when clicked again. | **✔** |

Clicked again.

Clicked.

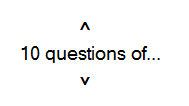
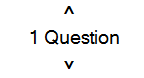
 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that multiple selector-buttons can be in their ‘on’ state at once (i.e. have a highlighted blue border). | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 20 | Normal | Student logged in, click ‘Practice Topics’ button. Click 3 different selector-buttons once. | The three clicked selector-buttons should be highlighted. | The three clicked selector-buttons are highlighted. | **✔** |



Three of the selector-buttons were clicked, and are now highlighted.

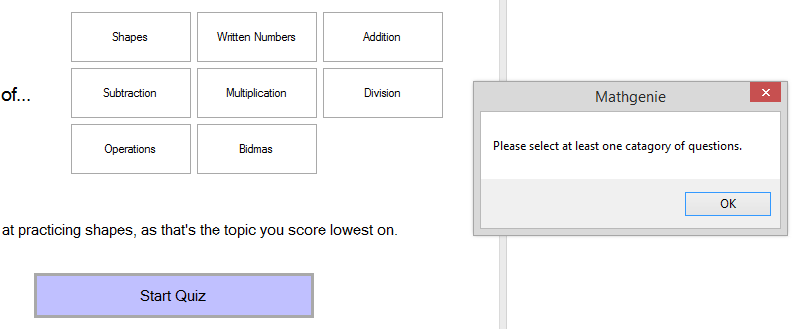
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a randomly generated quiz will not start when the quiz has been set to have 0 questions. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 21 | Erroneous | Student logged in, click ‘Practice Topics’ button, attempt to set number of questions to ‘0’, by clicking the number-down button. | Not possible to lower number of questions below ‘1’. | It is not possible to lower the number of questions below ‘1’, as it is limited within the code (bookmarked in the technical solution section). | **✔** |

Value will not reduce below ‘1’.

Down button clicked to lower value.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a randomly generated quiz will not start when no topics have been chosen using the selector-buttons. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 22 | Erroneous | Student logged in, click ‘Practice Topics’ button, no topic buttons toggled to ‘on’ state. | Message shown to user informing them to choose at least one topic to practice. | Message-box appeared, stating “Please select at least one category of questions”. | **✔** |

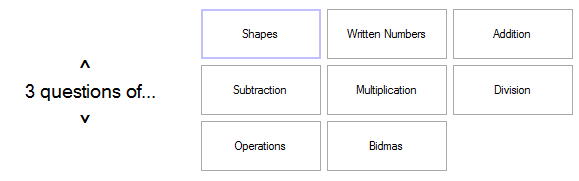


Message appears.

Clicked.

No topic-selection buttons highlighted.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a randomly generated quiz is created correctly, with questions only of the topics chosen in ‘Practice Mode’, and with the correct number of questions chosen also. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 23 | Normal | Student logged in, click ‘Practice Topics’ button. Choose ‘shapes’ topic with 3 questions. Start the quiz. | Three questions of topic ‘shapes’ should be displayed in a quiz one-by-one. | Three questions of topic ‘shapes’ were displayed in a quiz, one-by-one. No questions of other topics were displayed. | **✔** |



3 questions.

Only ‘shapes’ topic has been chosen.

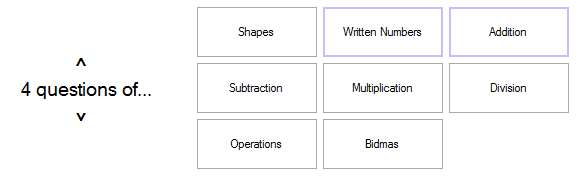
The following questions appeared, which are all of the shapes topic. The question numbers can be seen in the top-right, and the question in the top-left. Test was successful.







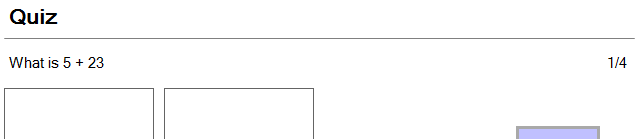
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Another test, to test that a randomly generated quiz is created correctly, with questions only of the topics chosen in ‘Practice Mode’, and with the correct number of questions chosen also. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 24 | Normal | Student logged in, click ‘Practice Topics’ button. Choose ‘written numbers’ and ‘addition’ topics with 4 questions. Start the quiz. | Four questions, of topics ‘written numbers’ and ‘addition’ should be displayed in a quiz, one-by-one. | Four questions, of topics ‘written numbers’ and ‘addition’ were displayed in a quiz, one-by-one. No questions of other topics were displayed. | **✔** |

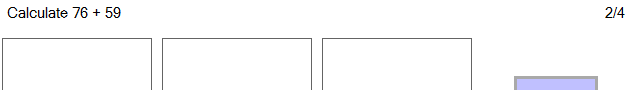


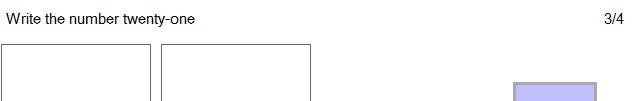
Note the chosen topics.

4 questions.

The following questions appeared, which are all of the written numbers or addition topics. The question numbers can be seen in the top-right, and the question in the top-left. Test was successful.



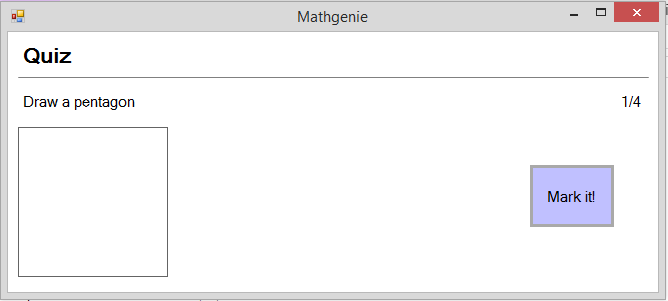






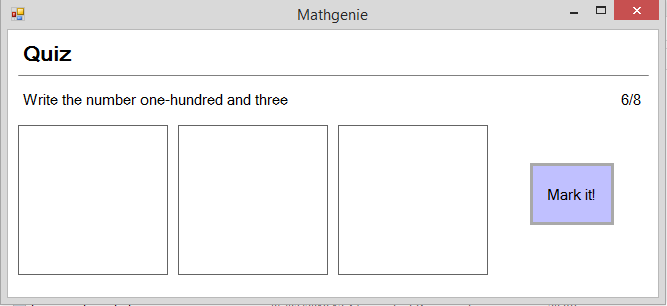
### General tests of the quiz system:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct number of canvases are dynamically generated for a student to answer a question with. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 25 | Normal | Question on ‘shapes’ topic displayed. | One canvas should be displayed for drawing of shape. | One canvas was dynamically generated and displayed for drawing of a shape. | **✔** |



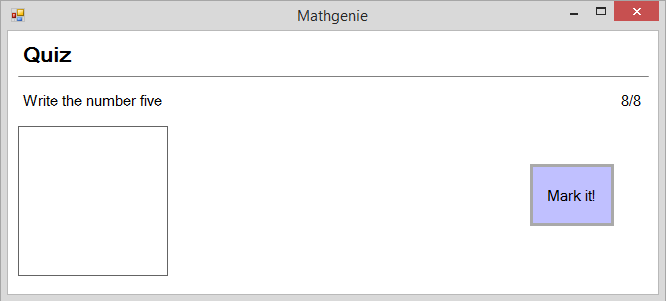
Question is a shapes question. One canvas generated, as is correct.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct number of canvases are dynamically generated for a student to answer a question with. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 26 | Normal | Question on ‘written numbers’ topic displayed, with a three-digit answer. | Three canvases should be displayed - one for each digit. | Three canvases were dynamically generated and displayed - one for each digit. | **✔** |



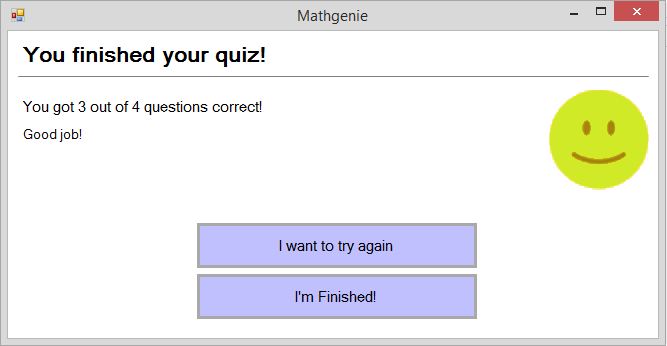
Three canvases generated, for a 3 digit answer.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct number of canvases are dynamically generated for a student to answer a question with. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 27 | Normal | Question on ‘written numbers’ topic displayed, with a single-digit answer. | One canvas should be displayed, as the answer is a single digit. | One canvas was dynamically generated and displayed, as the answer is a single digit. | **✔** |

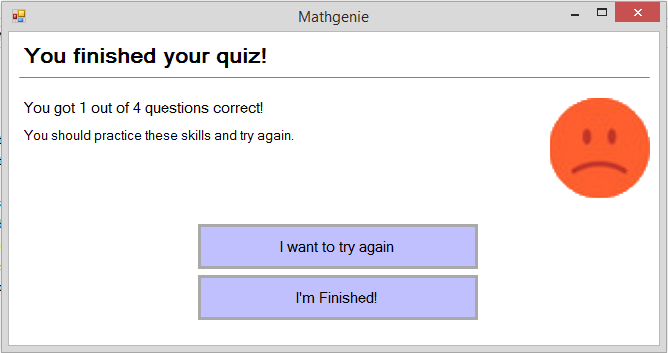


One canvas generated, for a single digit answer.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the score of correct answers that a student achieves in a quiz is correctly counted and displayed as feedback after a quiz is finished. Test that correct emotion face is displayed. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 28 | Normal | Start a randomly generated quiz with 4 questions. Get 3 correct, 1 wrong. | Score of 3 out of 4 (i.e. 75%) should be displayed, along with a green emotion face. | Score of “You got 1 out of 4 questions correct” was displayed, along with a green emotion face. | **✔** |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the score of correct answers that a student achieves in a quiz is correctly counted and displayed as feedback after a quiz is finished. Test that correct emotion face is displayed. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 29 | Normal | Start a randomly generated quiz with 4 questions. Get 1 correct, 3 wrong. | Score of 1 out of 4 (i.e. 25%) should be displayed, along with a red emotion face. | Score of “You got 1 out of 4 questions correct” was displayed, along with a red emotion face. | **✔** |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that repeat-quiz function repeats the previous quiz absolutely identically, and therefore functions correctly. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 30 | Normal | Start a randomly generated quiz with 3 questions. When finished, click ‘try again’ (aka. repeat quiz) button, and complete quiz. | 3 questions should be displayed. When the repeat-quiz button is clicked, the same 3 questions should be displayed, in the same order. | 3 questions were displayed. When the repeat-quiz button was clicked, the same 3 questions were displayed, in the same order. Evidence screenshots are provided of the 3 questions shown in each quiz. | **✔** |

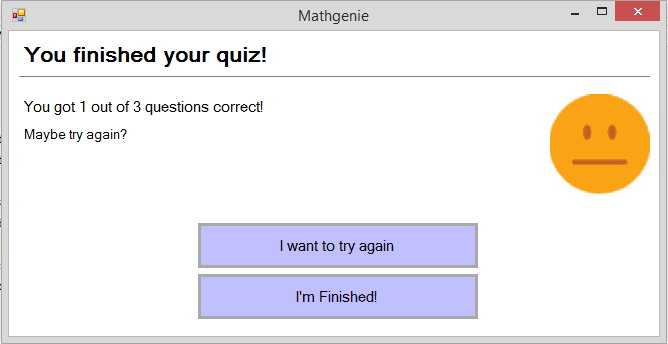
These three questions were displayed:







Then, the quiz finished panel appeared, and the ‘I want to try again’ button clicked:



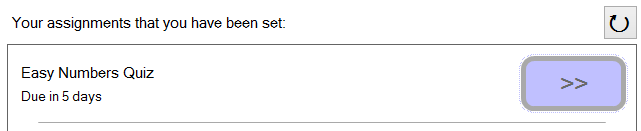
After, the same three questions from before were shown, in the same order (note the question numbers to see the order):







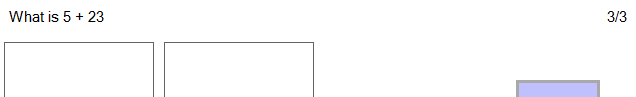
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct questions of the quiz which an assignment contains are displayed in the correct order. This will also test that only the questions for a quiz, and no others, are displayed when a quiz from an assignment is being completed. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 31 | Normal | Start ‘Easy Numbers Quiz’ quiz (which has a specified order), and work through all questions. | The questions of the quiz should be displayed in the correct order (the order they are set as in the database). | The questions of the quiz, which are shown in the database evidence screenshots, were displayed in the correct order (order which is stored in the database). | **✔** |



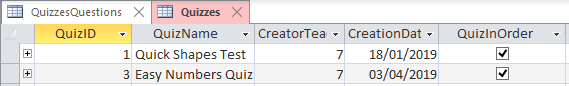
Clicked to start quiz.

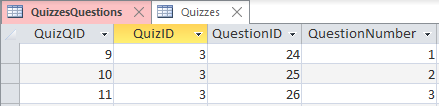


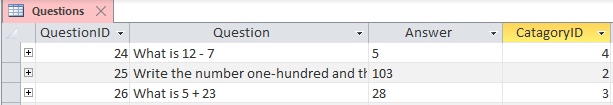




These above questions which were displayed in order (note the question numbers in the top-right of the screenshots) are the same questions, and were displayed in the same order, as that is stored in the database:

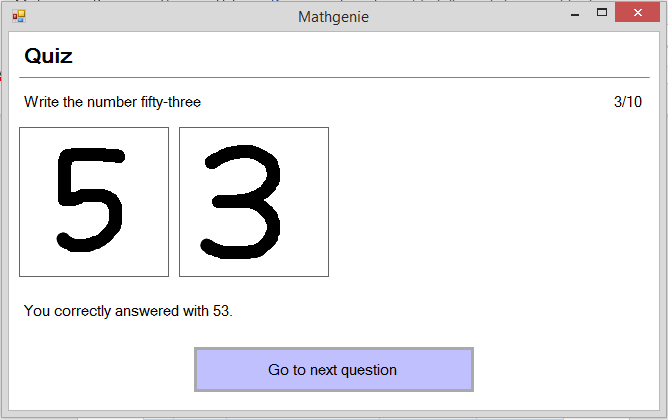




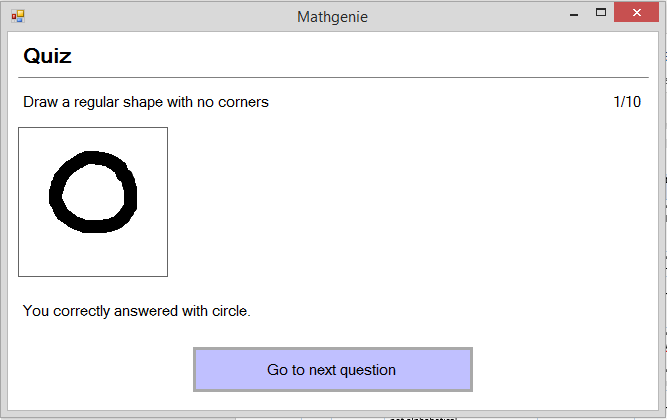


### Tests of the accuracy of the AI question marking:

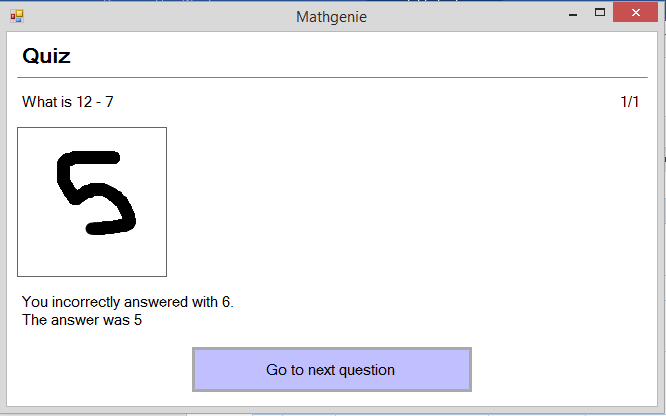
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the AI marks a question correctly, when the correct answer is drawn in the answer canvas(es), by classifying the bitmap from the canvas with the corresponding character drawn in the bitmap. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 32 | Normal | Question of ‘written numbers’ topic. Input correct answer of ‘53’. | “You correctly answered with ‘53’.” | “You correctly answered with ‘53’.” | **✔** |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the AI marks a question correctly, when the correct answer is drawn in the answer canvas(es). | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 33 | Normal | Question of ‘shapes’ topic. Draw correct answer of ‘circle’. | “You correctly answered with ‘circle’.” | “You correctly answered with ‘circle’.” | **✔** |

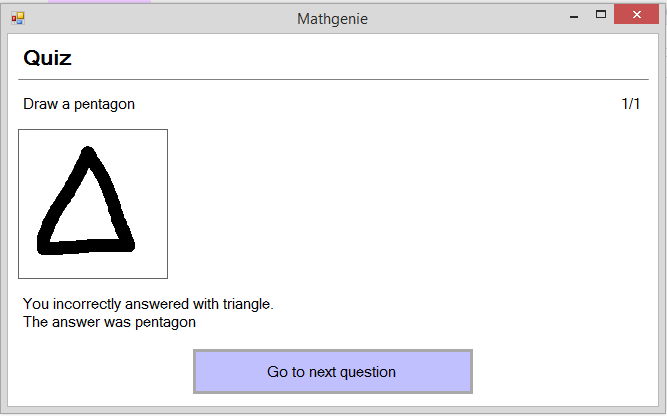


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the AI marks a question correctly, when the correct answer is drawn in the answer canvas(es). | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 34 | Normal | Question of ‘subtraction’ topic. Input correct answer of ‘’. | “You correctly answered with 5.” | “You incorrectly answered with 6. The answer was 5.” | **X** |

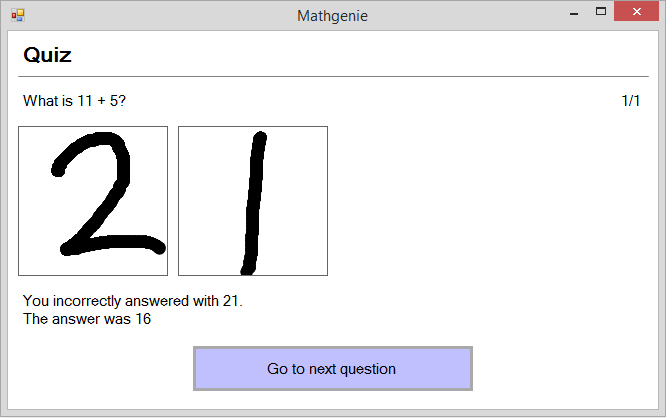


**Test 34 was unsuccessful, because the AI did not correctly classify the drawn number as ‘5’, and instead classified it as a ‘6’. The fact that this test failed is not a problem, since as set out in specification IV, part 11, the AI character classification algorithm needs to be only 90% accurate. In fact, it is impossible to get an algorithm with 100% accuracy, because what number a drawn digit really is could be disputed between different human eyes. As can be seen across all the tests involving the AI, it does meet the criteria of 90% accuracy.**

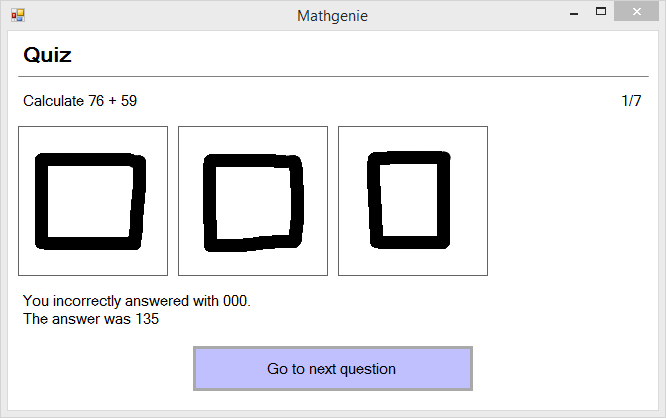
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the AI marks a question as incorrect, when an incorrect answer is drawn in the answer canvas(es). | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 35 | Normal | Question of ‘shapes’ topic. Input incorrect answer of ‘triangle’. | “You incorrectly answered with ‘triangle’. The answer was ‘pentagon’.” | “You incorrectly answered with triangle. The answer was pentagon.” | **✔** |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the AI marks a question as incorrect, when an incorrect answer is drawn in the answer canvas(es). | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 36 | Normal | Question of ‘addition’ topic. Input incorrect answer of ‘21’. | “You incorrectly answered with ‘21’. The answer was ‘16’.” | “You incorrectly answered with 21. The answer was 16.” | **✔** |

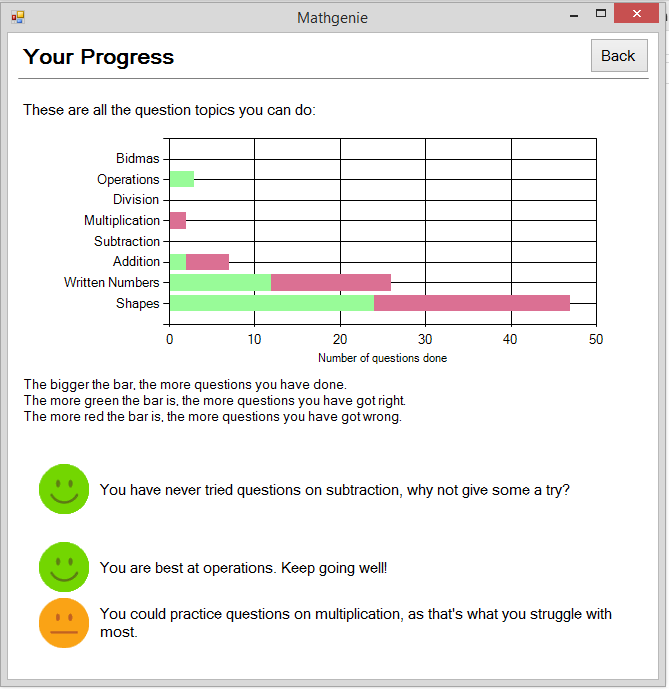


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the system does not crash, and does mark a question as incorrect, when an answer of the wrong format is drawn in the answer canvas(es). If, for example, a shape is drawn in the answer canvases for a question where a numerical answer is expected (e.g. an addition question), the AI will not classify the answer as a shape, and will instead classify it as a number (even though it is not a number), since only numerical answers are expected so numbers are the only possible outputs for the AI classifier. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 37 | Erroneous | Question of ‘addition’ topic. Draw a square in all answer canvases. | “You incorrectly answered with ‘xxx’. The answer was ‘135’.”  …where ‘xxx’ is any random classification returned from the AI, within the expected input characters (0-9). | “You incorrectly answered with 000. The answer was 135.”  The AI classified the squares as 0’s, since ‘0’ is the most similar looking numerical digit to a square. | **✔** |

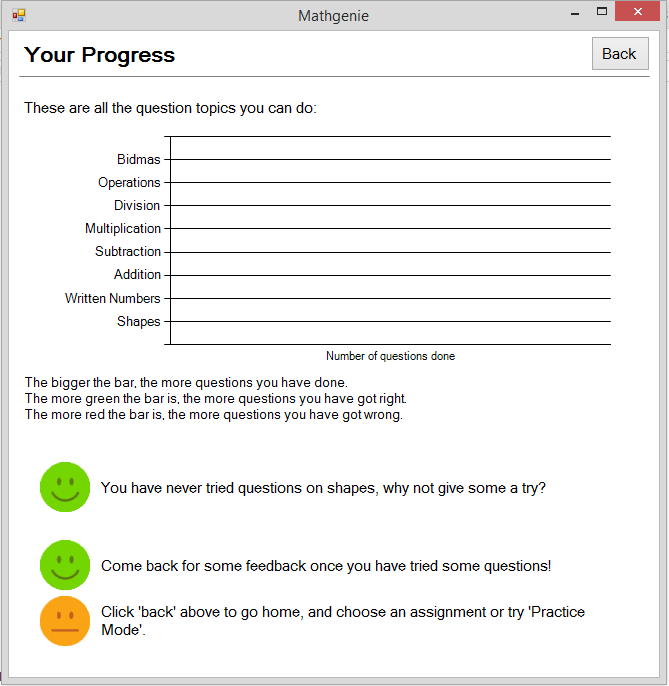


### Tests for the feedback and graphs displayed to a student in ‘See Your Progress’ panel:

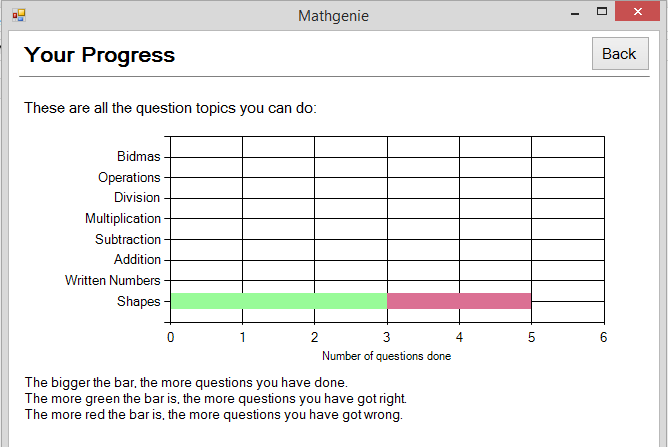
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the graph of a breakdown of performance across question topics displays well, with many completed questions across topics. Also tests the dynamically generated feedback text, below the graph. This will simulate a user who has a couple hours usage of the system. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 38 | Normal | Logged in as James Bungay, clicked ‘See Your Progress’ button. | Graph should display with large bars completely on screen, and small bars still visible. Text feedback should be displayed - text for a topic to try should be for a topic that has no attempted questions yet. | Graph displays with large bars completely on screen; small bars are not too small so as not to be visible. Text feedback is displayed - text for a topic to try is for topic ‘division’ - this topic has not been attempted yet, as seen in the graph. | **✔** |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the graph and dynamically generated feedback text do not have error in the way they display, for a new student user who has not answered a single question before. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 39 | Boundary | Logged in as Thomas Kelly, clicked ‘See Your Progress’ button. | Graph of topics should display no bars. Text feedback should not be blank or have error, despite having no data to base feedback on (as user has done nothing). | Graph of topics has no bars displayed. Text feedbacks are not blank, and are replaced with messages suggesting how to get started with the system - see screenshots, compared to those from test 38 (for a user with usage of the system). | **✔** |



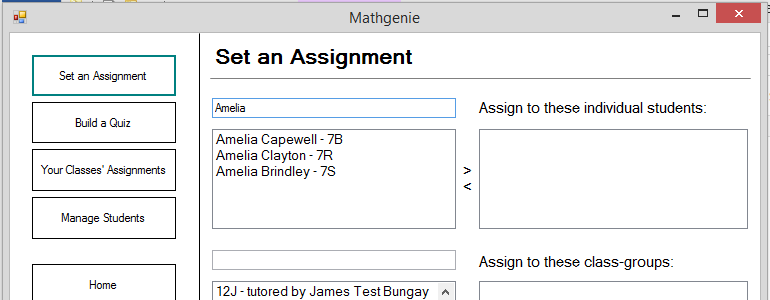
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Using the user from the previous test, test that, once some questions have been completed by a new user, this progress is displayed in the graph on the ‘See Your Progress’ panel. This will test that the red and green sections of bars in the topics graph, to represent correct and incorrect questions, display correctly. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 40 | Normal | With user from test above, do 5 practice questions in ‘shapes’ topic (3 correct, 2 wrong). Go to ‘See Your Progress’ panel. | A single bar should be visible on the topics graph, with a 3 unit wide green section, and a 2 unit wide red section. There should not be a bar beside other topics, as no questions will have been attempted for these topics. | A single bar is visible on the topics graph, with a 3 unit wide green section, and a 2 unit wide red section. There are not any bars beside other topics (which would be in error, since no questions have been tried for these topics). | **✔** |



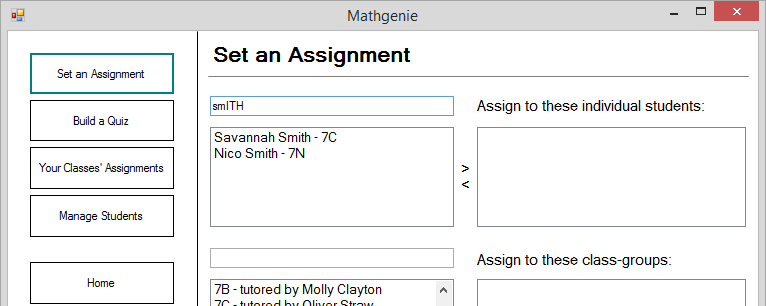
### Tests of the teacher-side of the system:

### Tests of the ‘Set an Assignment’ panel:

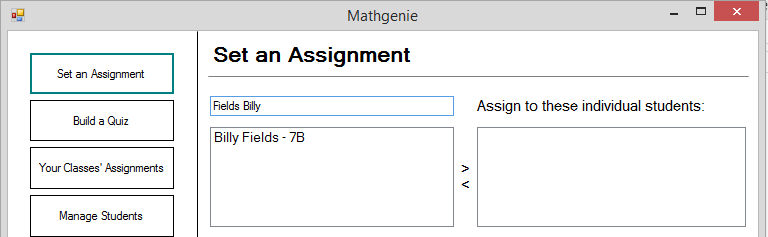
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the search box for selecting students to set an assignment to works correctly. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 41 | Normal | ‘Amelia’ | Amelia Capewell  Amelia Clayton  Amelia Brindley | Amelia Capewell  Amelia Clayton  Amelia Brindley | **✔** |



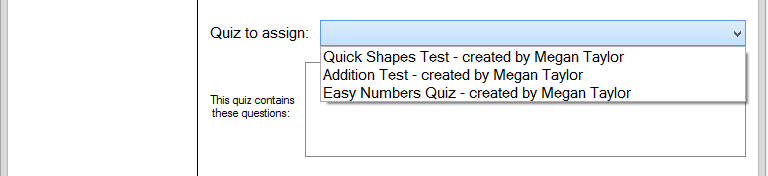
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the search box for selecting students, is not case sensitive. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 42 | Boundary | ‘smITH’ | Savannah Smith  Nico Smith | Savannah Smith  Nico Smith | **✔** |

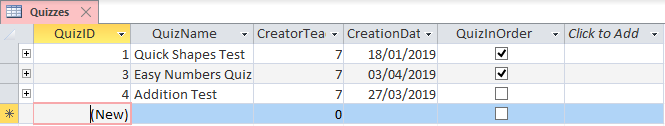


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the search box will still return a student, if their first and last names are put in reverse order. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 43 | Boundary | ‘Fields Billy’ | Billy Fields | Billy Fields | **✔** |

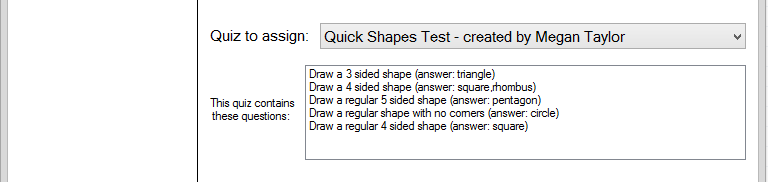


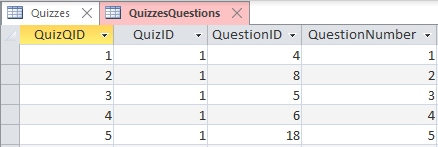
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that all quizzes available in the database are visible in the ‘Quiz to assign’ combo-box. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 44 | Normal | Click ‘Quiz to assign’ box, to see drop-down-list. | All quizzes in the database should be visible, with no duplicates. | All quizzes shown in the evidence screenshot of the database are visible, with no duplicates. | **✔** |

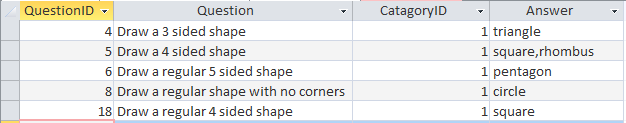




|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that, when a quiz is selected using the ‘Quiz to assign’ combo-box, the contained questions appear in the list-box below. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 45 | Normal | Click ‘Quiz to assign’, click ‘Quick Shapes Test’. | The 5 questions in the ‘Quick Shapes Test’ should be visible with their answers, in the list-box. | The 5 questions in the ‘Quick Shapes Test’ are visible with their answers, in the list-box, as per the evidence screenshot of the database. | **✔** |

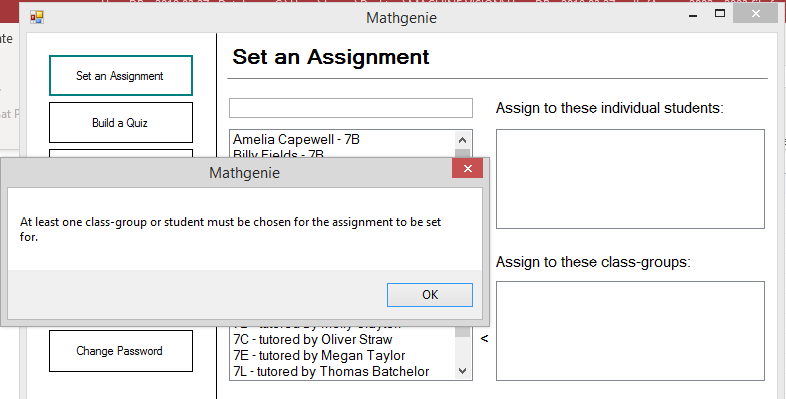






The QuizID of ‘Quick Shapes Test’ is ‘1’ (see database screenshot in test 44). As can be seen from the above database screenshots, the questions for the quiz with QuizID have QuestionIDs which correspond to those of the questions displayed in the list-box on the UI. Therefore, test successful.

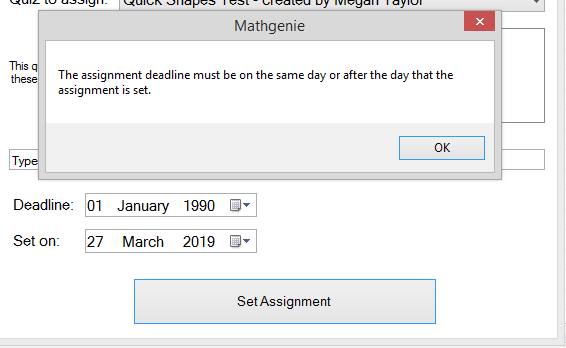
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that an assignment cannot be set with no students or class-groups chosen. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 46 | Erroneous | Select no students or class-groups, click ‘Set Assignment’. | Message: “At least one class-group or student must be chosen for the assignment to be set for.” | Message-box appears, stating “At least one class-group or student must be chosen for the assignment to be set for.” | **✔** |



Message-box appeared.

Set to no students or class-groups.

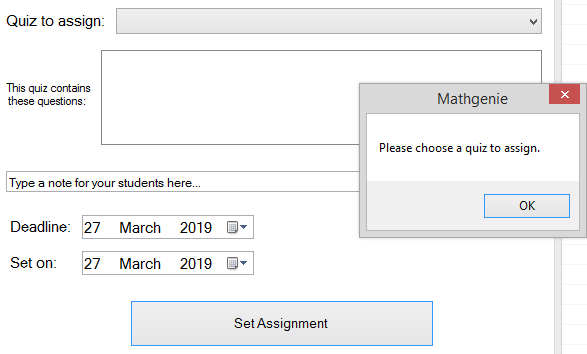
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that an assignment cannot be set with the deadline in the past. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 47 | Erroneous | Set deadline to ‘1/1/1990’, click ‘Set Assignment’. | Message: “Assignment cannot be set in the past.” | Message-box appeared, stating “Assignment cannot be set in the past.” | **✔** |



Message-box appeared.

Deadline in the past.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that an assignment cannot be set, with no quiz chosen using the ‘Quiz to assign’ combo-box. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 48 | Erroneous | Don’t choose a quiz, click ‘Set Assignment’. | Message: “Please choose a quiz to assign.” | Message-box appears, stating “Please choose a quiz to assign.” | **✔** |

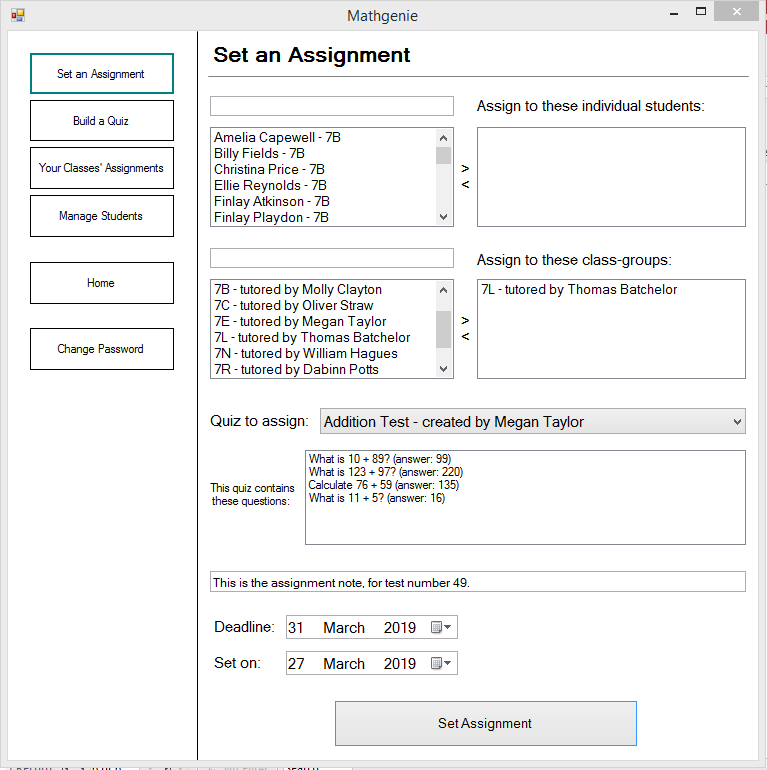


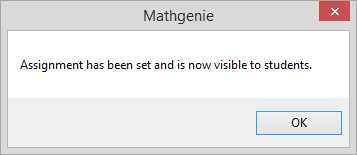
Message-box appeared.

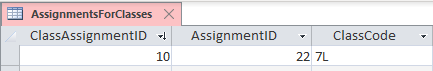
No quiz chosen for the assignment.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that an assignment can be set successfully, without the system crashing or showing an error. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 49 | Normal | Input valid assignment details, with ‘7L’ class-group chosen, ‘Addition Test’ chosen as quiz to assign, and a deadline in the future. | Message should appear stating that the assignment has been set. | Message-box appears, stating “Assignment set”, and assignment has appeared in database. | **✔** |

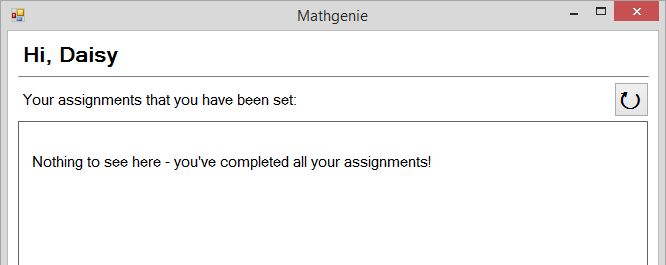
Evidence for test 49 on next page…



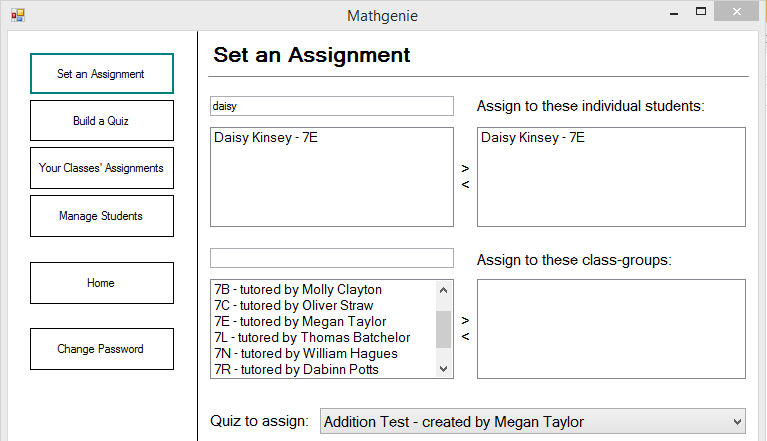




|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that an assignment, when set to a student, appears in the student’s list of incomplete assignments. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 50 | Normal | Set ‘Addition Test’ to Daisy Kinsey. | Logged in as student ‘Daisy Kinsey’ prior to setting assignment - no assignments set.  *Then: set ‘Addition Test’ to Daisy Kinsey.*  Then: Log in as Daisy Kinsey, ‘Addition Test’ should be set. | No assignments set to Daisy Kinsey beforehand.  After setting the assignment, logging in as Daisy Kinsey showed that ‘Addition Test’ was assigned to her. | **✔** |

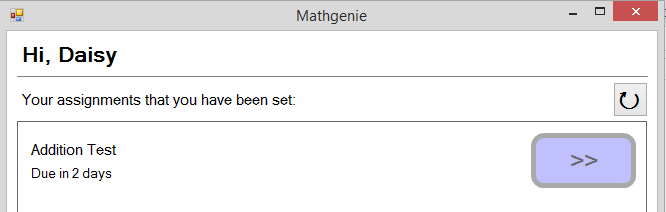


Before: No assignments set.



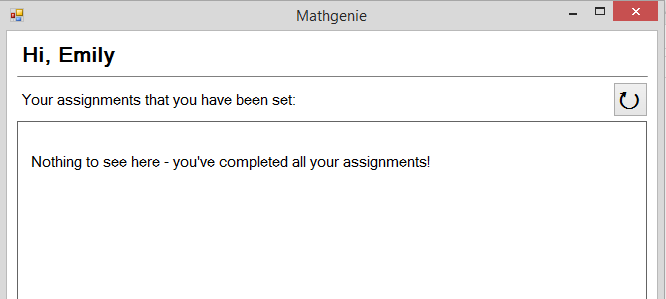
Set to Daisy Kinsey.

Addition Test set.



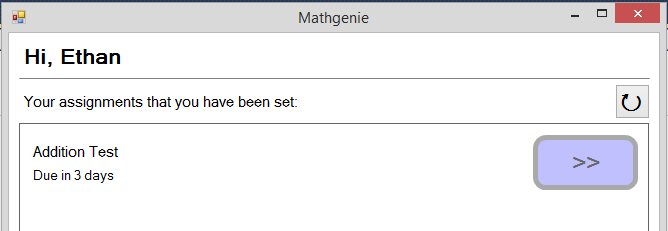
After: Addition Test has been set to Daisy Kinsey.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the ‘Addition Quiz’ assignment set in the above test, has not appeared for any other students, other than ‘Daisy Kinsey’ who it was set for. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 51 | Normal | Log in as student ‘Emily Davis’. | The ‘Addition Quiz’ assignment should not be present in their list of incomplete assignments. | The ‘Addition Quiz’ assignment is not present in Emily Davis’ list of incomplete assignments. | **✔** |

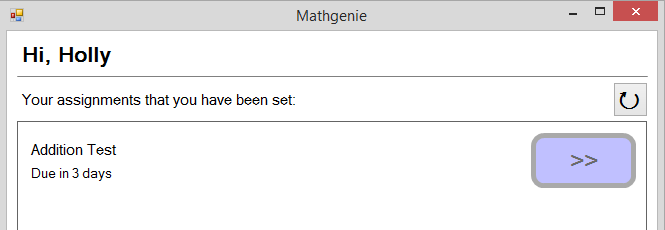


No assignments set for Emily Davis.

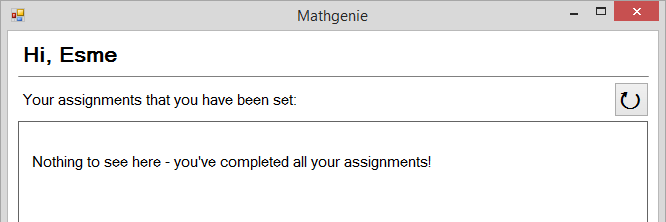
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the assignment set to a whole class-group is visible to members of that class-group, and no other students. This will be done using the assignment set to ‘7L’ in test 49. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 52 | Normal | Log in as three different students: ‘Ethan Das’ & ‘Holly Burton’ of 7L, and ‘Esme Halford’ of 7C. | Assignment set in test 49 should be visible to both students of 7L (Ethan Das and Holly Burton). The assignment should not be visible to ‘Esme Halford’, as they are in 7C, which the assignment was not set for. | Assignment set in test 49 is visible to both students of 7L (Ethan Das and Holly Burton). The assignment is not visible to ‘Esme Halford’, as they are in 7C, which the assignment was not set for. | **✔** |



Addition Test has been set to Ethan Das of 7L.



Addition Test has been set to Holly Burton of 7L.

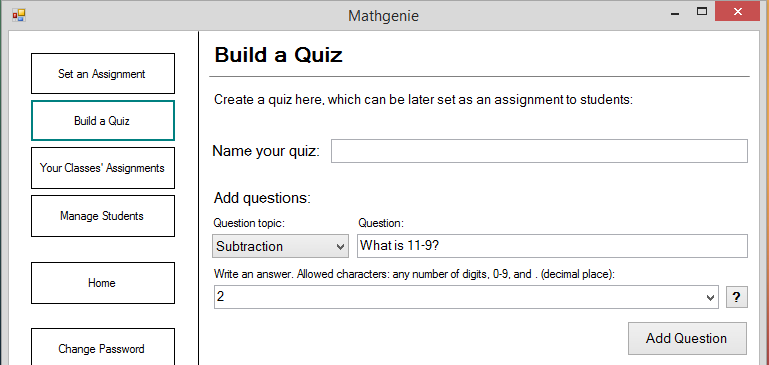


Addition Test has **not** been set to Esme Halford of 7C.

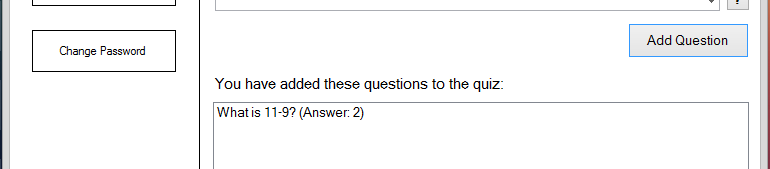
As can be seen in the database screenshot for test 49, the ‘Addition Test’ assignment has been set to students of the class-group 7L. Ethan Das and Holly Burton are in 7L, so the assignment appeared to them. Esme Halford is in 7C, so the assignment did not appear for him. Therefore, test successful.

### Tests of the ‘Build a Quiz’ panel:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Tests that, when the ‘Add Question’ button is clicked to add a question to the quiz, the question appears, with its answer, in the list-box of questions below the ‘Add Question’ button. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 53 | Normal | Fill in question details of…  Topic: ‘Subtraction’, Question: ‘What is 11-9?’, Answer: ‘2’. Then click ‘Add Question’ button. | “What is 11-9? (answer: 2)” should appear in the questions list-box. | “What is 11-9? (answer: 2)” appeared in the questions list-box. | **✔** |

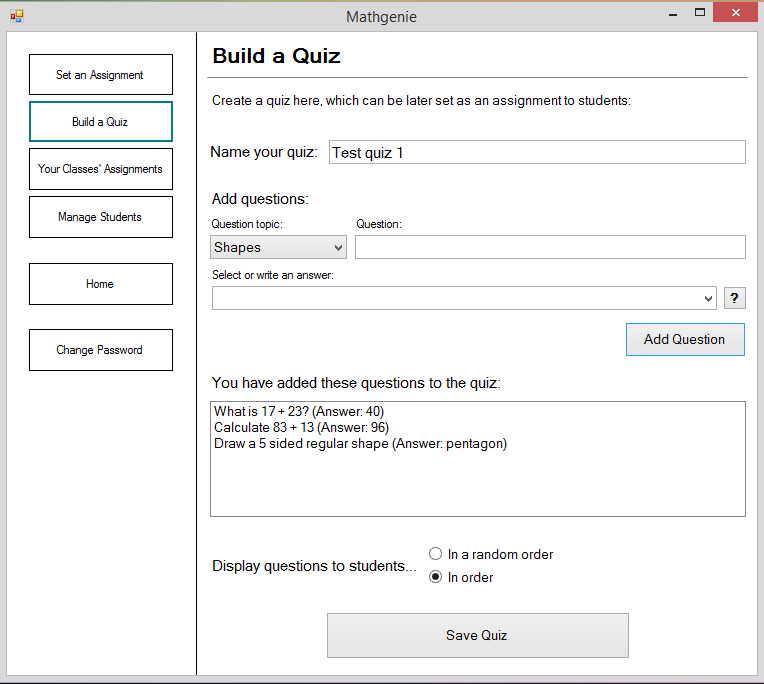


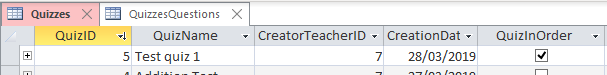
Specified detailed filled in.

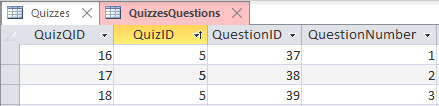


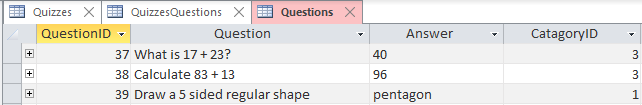
Appeared as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that quiz details successfully store in the database. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 54 | Normal | Quiz named: “Test quiz 1.” Add 2 questions of topic ‘addition’, and 1 question of topic ‘shapes’. Set to ‘In order’. Click the ‘Save Quiz’ button. | The details described in the test data section should appear in the database exactly as they were inputted. | The details described in the test data section appeared in the database exactly as they were inputted. | **✔** |

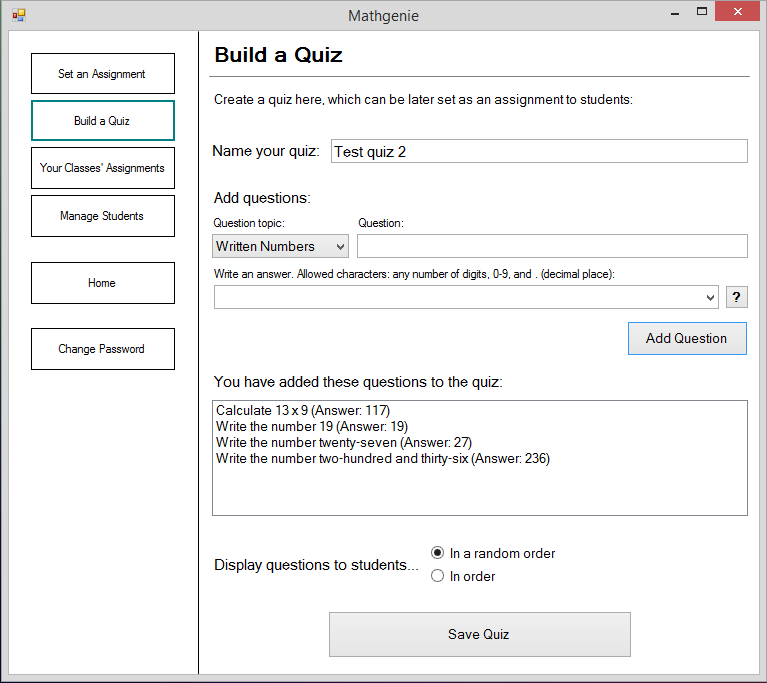


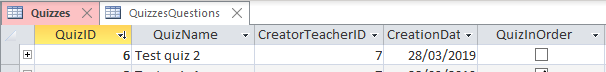


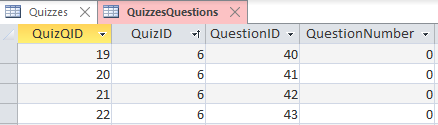


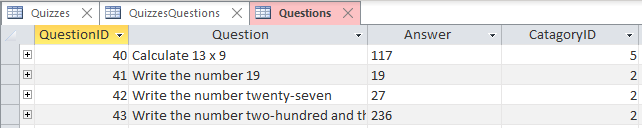


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that quiz details successfully store in the database. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 55 | Normal | Quiz named: “Test quiz 2.” Add 1 question of topic ‘multiplication’, and 3 question of topic ‘written numbers’. Set to ‘In a random order’. Click the ‘Save Quiz’ button. | The details described in the test data section should appear in the database exactly as they were inputted, with ‘0’ for questions numbers, as set to random order. | The details described in the test data section appeared in the database exactly as they were inputted, with ‘0’ for questions numbers, as set to random order. | **✔** |

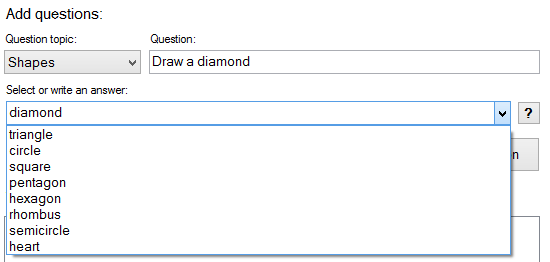






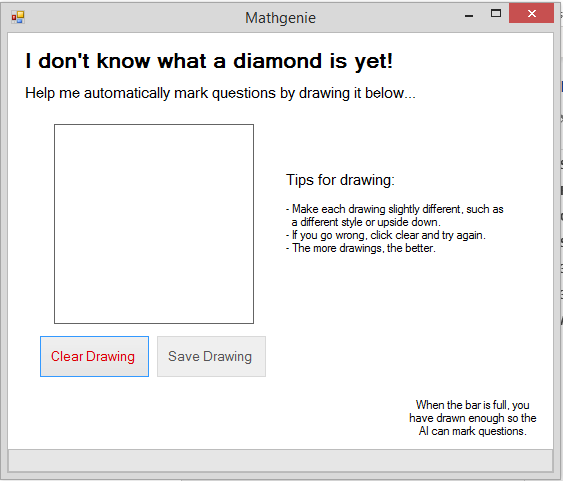


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that, for a question topic with an unlimited number of possible answers (e.g. shapes, since there is no limit to number of shapes, but ‘addition’ would only have answers with digits 0-9), the ‘train AI’ panel appears, when a new answer is typed into the answer input-box. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 56 | Normal | Set question topic combo-box to ‘Shapes’. Write a question of ‘draw a diamond”. Write an answer of ‘diamond’. Click ‘Add Question’ button. | The ‘Train AI’ window should appear when the ‘Add Question’ button is clicked. At the top of the window, the text “I don’t know what a diamond is yet” should be present. | The ‘Train AI’ window appeared when the ‘Add Question’ button was clicked. At the top of the window, the text “I don’t know what a diamond is yet” is present. | **✔** |



‘Add Question’ button then clicked.

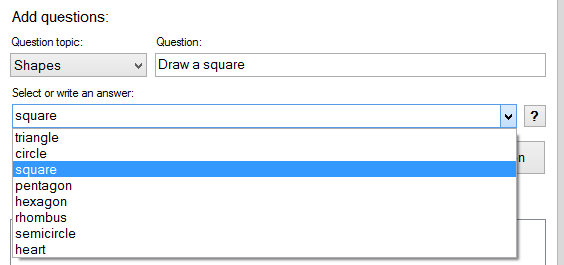
‘Diamond’ is not in the drop-down list of answers which the AI currently knows.



This ‘Train AI’ form then appeared, as it should do.

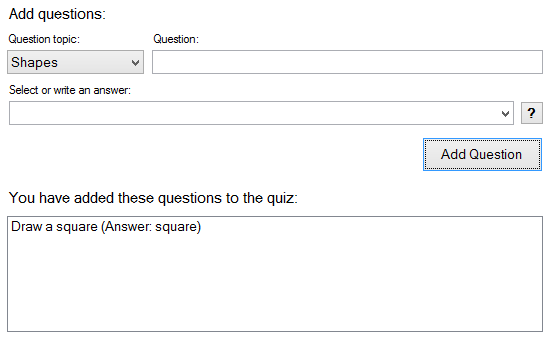
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that, for a question topic without a limited number of possible answers, the ‘train AI’ panel does not appear when an answer is chosen from the answer input-box (that is an option in the drop-down box as the AI already knows how to classify it), rather than a new answer being typed in. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 57 | Normal | Set question topic combo-box to ‘Shapes’. Write a question of ‘draw a square”. Select answer of ‘Square’ from the drop-down of the answer selection input-box. Click ‘Add Question’ button. | The ‘Train AI’ window should **not** appear when the ‘Add Question’ button is clicked. The question should immediately appear in the questions list-box below. | The ‘Train AI’ window did not appear when the ‘Add Question’ button was clicked. The question immediately appeared in the questions list-box below. | **✔** |

See evidence for test 57 on the next page.



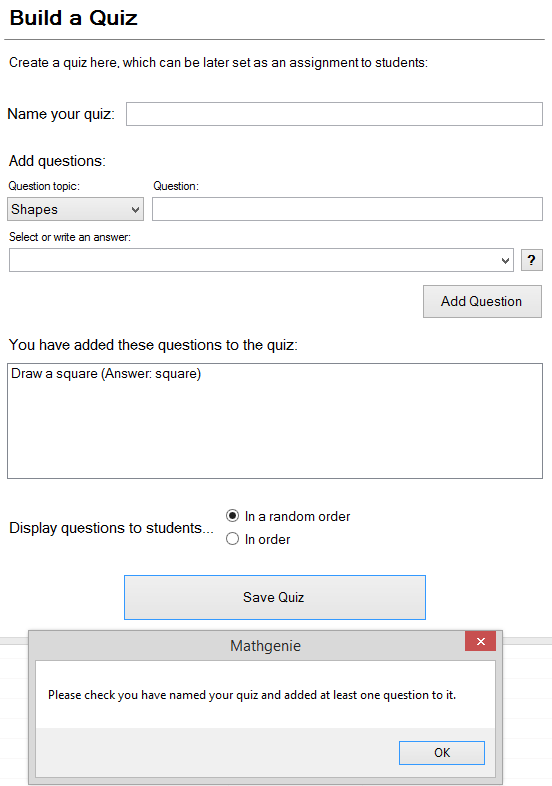
‘Add Question’ button then clicked.

‘Square’ is in the drop-down list of answers which the AI currently knows.



The question then immediately appeared in the questions list-box, without the train AI window appearing.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a quiz cannot be saved by clicking the ‘Save Quiz’ button, without a quiz name inputted. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 58 | Erroneous | Add a question to the quiz, but leave quiz name blank. Click ‘Save Quiz’. | Message “Please check you have named your quiz and added at least one question to it” should appear. | Message-box appeared “Please check you have named your quiz and added at least one question to it”. | **✔** |

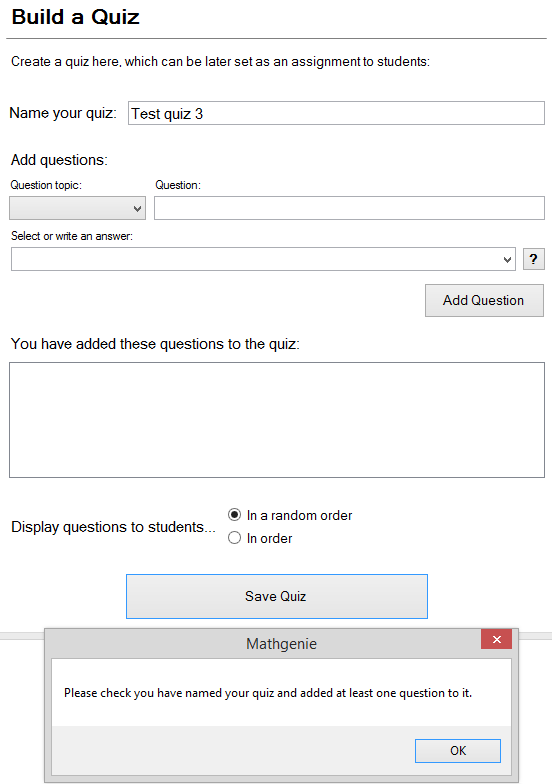


This message appears.

Clicked

No name for the quiz has been inputted.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a quiz cannot be saved by clicking the ‘Save Quiz’ button, without any questions added to the quiz. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 59 | Erroneous | Set the quiz name to ‘Test quiz 3’. Do not add any questions to the quiz (questions list-box should be empty). Click ‘Save Quiz’. | Message “Please check you have named your quiz and added at least one question to it” should appear. | Message-box appeared “Please check you have named your quiz and added at least one question to it”. | **✔** |



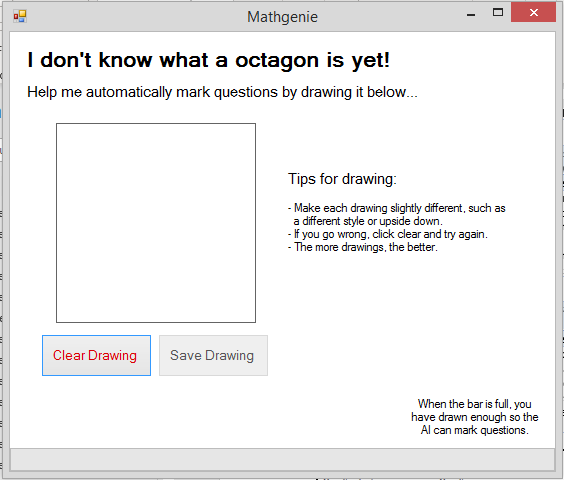
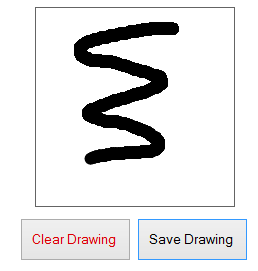
This message appears.

Clicked

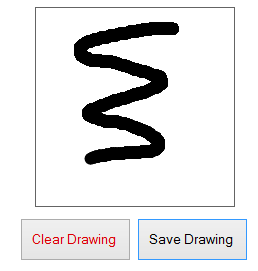
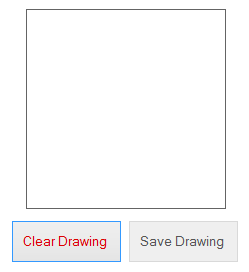
No questions have been added to the quiz.

### Tests of the AI training form:

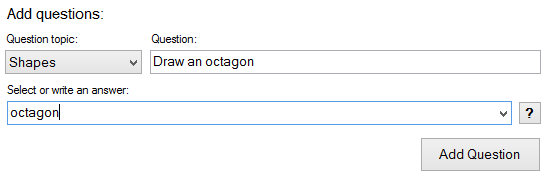
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the ‘Save Drawing’ button is not enabled, until the user has drawn something on the canvas. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 60 | Normal | Draw line on canvas. | ‘Save Drawing’ button should be disabled before drawing. Once a line is drawn, the button should enable automatically. | ‘Save Drawing’ button is disabled before drawing. When a line was drawn, the button enabled automatically. | **✔** |

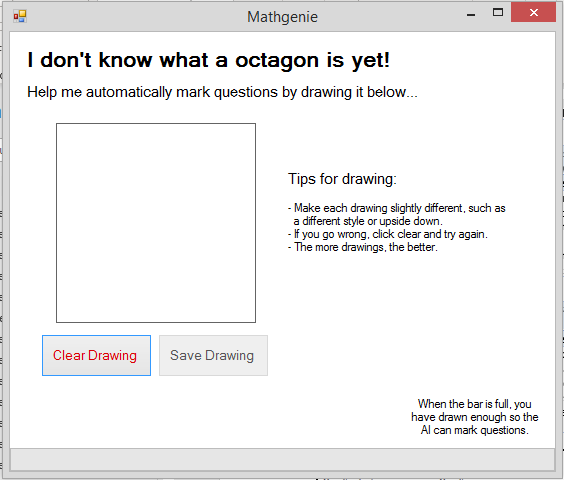
 

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the ‘Clear Drawing’ button clears the canvas, and the ‘Save Drawing’ button is disabled automatically when the canvas is cleared to white. | | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | | **Pass?** |
| 61 | Normal | Draw line on canvas. Click ‘Clear Drawing’. | Canvas should be plain white after the ‘Clear Drawing’ button is clicked. The ‘Save Drawing’ button should be disabled too. | | Canvas was plain white after the ‘Clear Drawing’ button was clicked. The ‘Save Drawing’ button was disabled too. | **✔** |

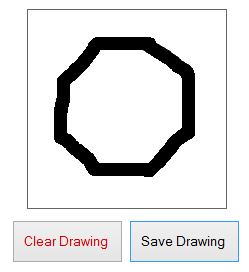
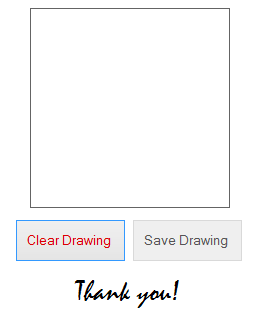
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct label, which the AI is being trained to recognise, is displayed in the bold text at the top of the window. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 62 | Normal | Add ‘Shapes’ question with answer ‘Octagon’ in ‘Build a quiz’ panel. Train AI panel should appear. | The bold text at the top of the train AI window which appears should state “I don’t know what an octagon is yet”. | The bold text at the top of the train AI window which appears says “I don’t know what an octagon is yet”. | **✔** |



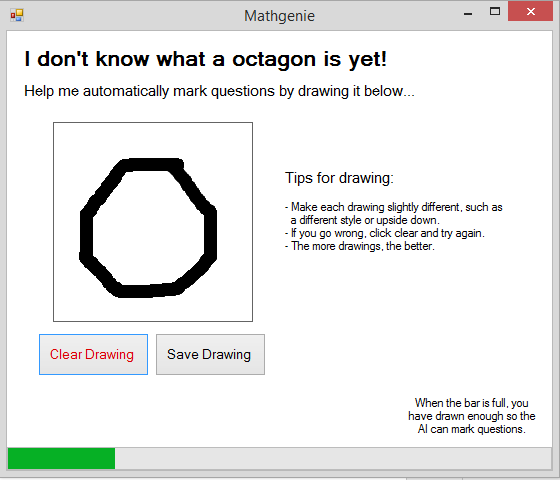
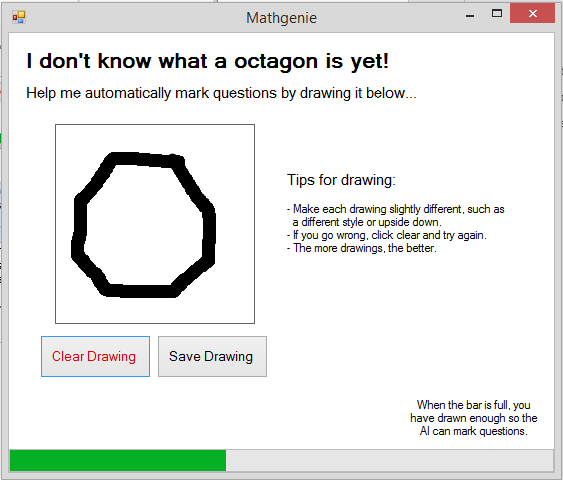


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that pixel-strings generated from drawings in the canvas, along with the correct label, are successfully stored in the correct AI data-set text file for the relevant character type (i.e. if drawing a shape, stored in the “trainingset\_shapes.txt” text file. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 63 | Normal | Following on from the previous test, draw an octagon in the canvas. Click ‘Save Drawing’. Repeat two more times. | The AI training data-set text file for shapes (“trainingset\_shapes”) should have three additional lines appended to it, each with unique IDs, the label ‘octagon’ and two 400 character strings of 0’s and 1’s, separated by colons. The lines should not have been present prior to the three octagons being drawn. | The AI training data-set text file for shapes (“trainingset\_shapes”) should had 3 additional lines appended to it, each with unique IDs, the label ‘octagon’ and two 400 character strings of 0’s and 1’s, separated by colons. The lines were not present prior to the three octagons being drawn. | **✔** |

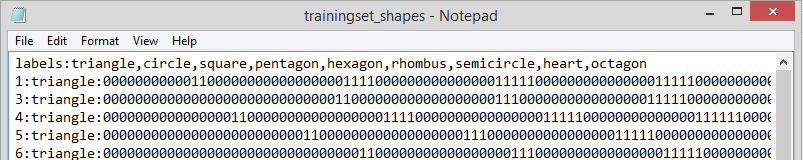
An Octagon was drawn, and the ‘Save Drawing’ button clicked.

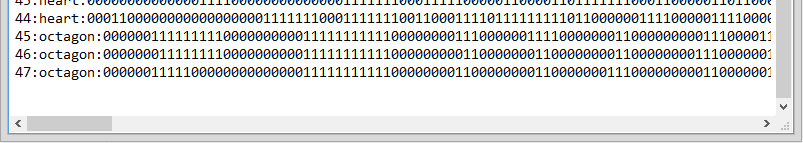
 

These two other octagons were then drawn too. The progress-bar can be seen filling here:

Here is a screenshot of the start and end of the “trainingset\_shapes.txt” text-file, after the three octagons have been drawn:

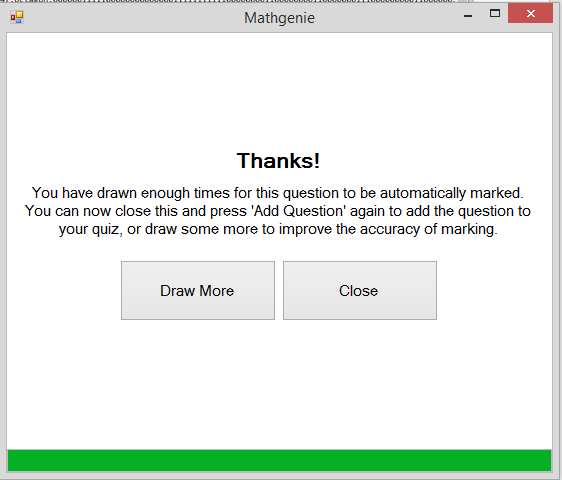




It can be seen that ‘octagon’ has been added to the list of labels on the first line of the text file, and that three lines have been added to the end, in the following correct format:  
[unique ID] : [label] : [400 char pixel string] : [400 char pixel string]

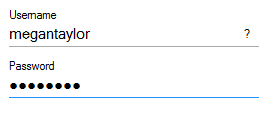
Therefore, test 63 is successful.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the panel stating that ‘enough drawings have been made for the AI to be trained with’ appears after 5 drawings have been drawn in the canvas and saved using the ‘Save Canvas’ button. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 64 | Normal | Following on from the previous test, draw an octagon and click ‘Save Drawing’ two more times. | After clicking the ‘Save Drawing’ button for the 5th time (2nd time in this test), the panel should appear, covering up the canvas and other buttons. | After clicking the ‘Save Drawing’ button for the 5th time (2nd time in this test), the panel appeared, covering up the canvas and other buttons. | **✔** |



### Tests of the ‘Your Classes’ Assignments’ panel:

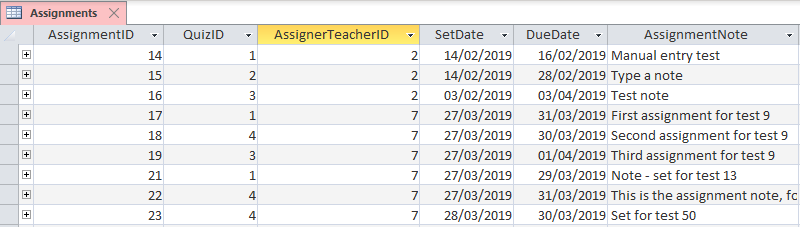
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that all of the assignments set by the logged in teacher (and no assignment set by others) appear correctly in the assignment selection list-box, with correct details for each assignment. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 65 | Normal | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. | Assignments which have been set by Megan Taylor, and no others, should be displayed in the list-box, along with their due date and other details. These will be shown in an evidence screenshot of the database. | Assignments which have been set by Megan Taylor (teacherID ‘7’), and no others, are displayed in the list-box, along with their due date and assignment note. These details are shown in evidence screenshots of the database. | **✔** |



Assignments shown here.

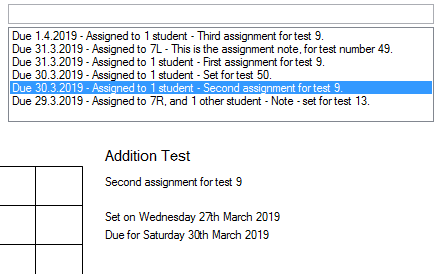
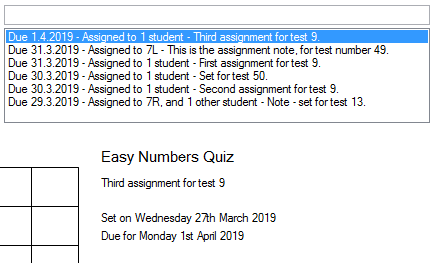
Logged in as Megan Taylor.





Only the assignments set by Megan Taylor (ID 7) are visible in the system UI. Other assignments can be seen in the database, that are not visible in the UI, as they are not set by Megan Taylor.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test to see if multiple assignments can be selected at once using the assignment selection list-box, which would cause an error as only one assignment’s details can be displayed at once. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 66 | Erroneous | Click any assignment. Details should display. Attempt to click another assignment whilst holding ‘ctrl’ on keyboard. | When the second assignment is clicked whilst holding ‘ctrl’ it should be selected and details shown, and the first one de-selected. | When the second assignment is clicked whilst holding ‘ctrl’ it is selected and it’s details are shown, and the first one de-selected. | **✔** |

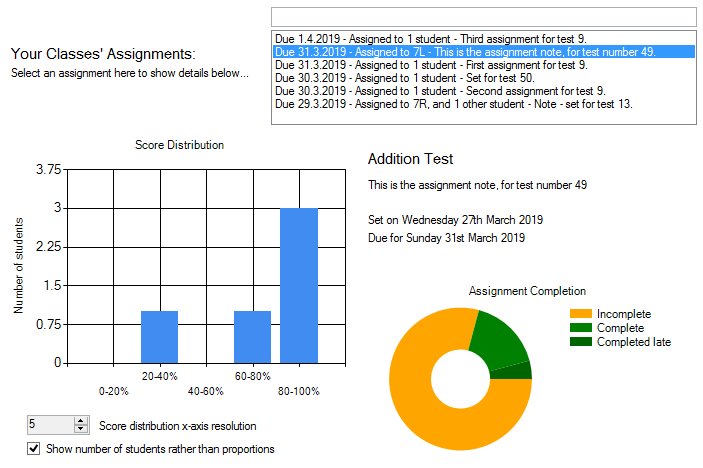
Second assignment is clicked whilst holding ‘ctrl’ key. Second assignment is selected, and first assignment’s details are replaced by second assignment’s details.

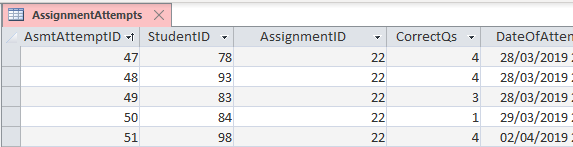
First assignment is clicked. Details appear below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the details (such as quiz name, assignment note, due date) for the correct assignment display below the assignment selection list-box, when an assignment is selected. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 67 | Normal | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | The details which correspond to the selected assignment should appear. These can be confirmed using an evidence screenshot of the database. | The details which correspond to the selected assignment appear. These are confirmed in an evidence screenshot of the database. | **✔** |

See screenshots from test 66 - when the assignment ‘Due 1.4.2019 - Assigned to 1 student - Third assignment for test 9” is clicked, the details below in the screenshot are shown. As per the database screenshot for test 65, it can be seen that the details of 1/4/2019 for due date, 27/3/2019 for set date, and “Third assignment for test 9” for assignment note, are corresponding details for an assignment, therefore have been displayed correctly in the UI.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct scores that students have achieved in the assignment selected using the assignment selection list-box, are summarised correctly in the score-distribution graph. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 68 | Normal | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | Student scores for the selected assignment, which can be confirmed using an evidence screenshot of the database, should be displayed in the correct columns of the graph. | Student scores for the selected assignment, which are confirmed by the evidence screenshots of the database, are displayed in the correct columns of the graph. | **✔** |





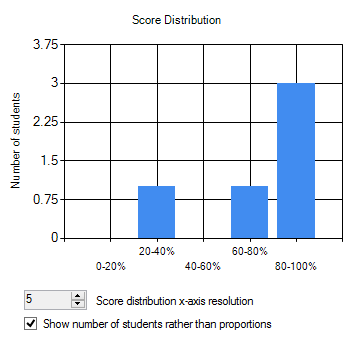
Three students scored 4/4 (100%), one student scored 3/4 (75%) and one student scored 1/4 (25%). Therefore, the test is successful, as in the graph there are three students in the 80-100% region, 1 student in the 60-80% region, and 1 student in the 20-40% region.

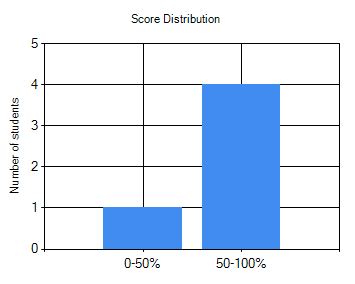
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the number of students who have completed the assignment selected using the assignment selection list-box, is summarised correctly in the completion pie-chart. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 69 | Normal | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | The number of students who have not completed, have completed and have completed the assignment late, should be displayed with correct proportions in the pie chart. | The number of students who have not completed, have completed and have completed the assignment late, which is confirmed using an evidence screenshot of the database, are displayed with correct proportions in the pie chart. | **✔** |

See the evidence screenshots for test 68. The due date for this assignment is 31/3/2019 - 4 students completed the assignment before then, and one student completed it after (as per the DateOfAttempt column in the database screenshot). Therefore, since the ‘complete’ section of the pie chart is 4x larger than the ‘completed late’ section, this data is being displayed correctly.

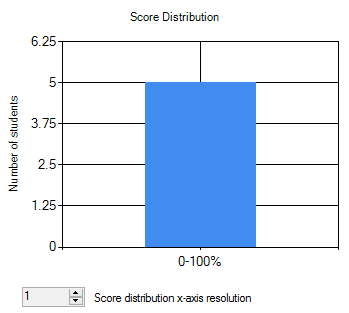
The class-group, 7L, which the assignment was set to, contains 23 students. Since 18 of these have not completed the assignment, the proportion of the pie chart taken by the yellow ‘incomplete’ section appears to be displaying correctly also. Therefore, test 69 is successful.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the resolution of score distribution graph’s x-axis changes when the numeric up down input-box’s (which is labelled for setting x axis resolution) value is changed. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 70 | Normal | Whilst on “Your Classes’ Assignments” panel, with an assignment selected, reduce the value in the numeric-up-down box from the default of 5, to ‘2’. | The number of columns displayed in the score distribution graph should decrease from 5 columns, to 2 columns. | The number of columns displayed in the score distribution graph decreased from 5 columns, to 2 columns. | **✔** |



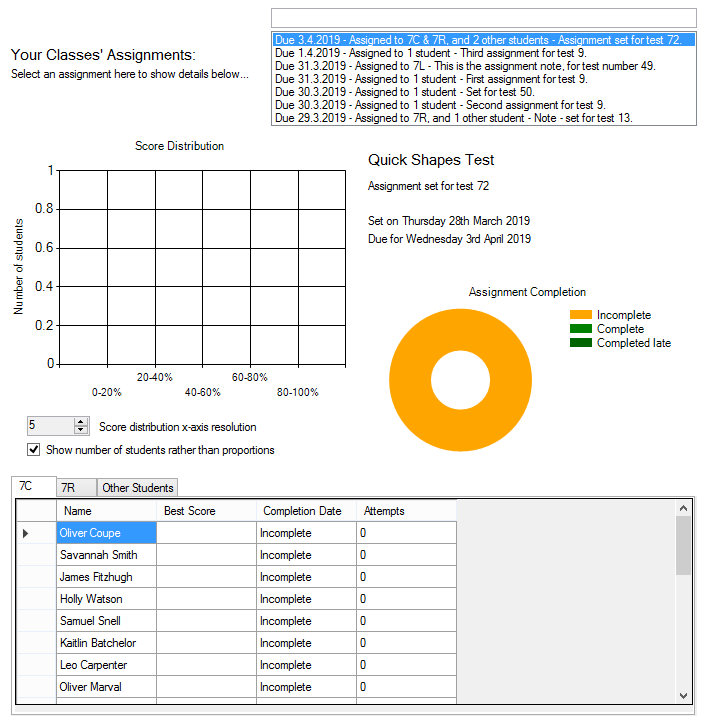


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that it is not possible to set the resolution of the score distribution graph to zero, using the numeric-up-down input-box labelled for setting x axis resolution. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 71 | Erroneous | Click the down button of the numeric-up-down box to decrease value, until it will no longer decrease. Attempt to type in ‘0’ to the box. | Resolution will decrease to 1 from the default of 5, in steps of 1, when the down button is clicked. Will not decrease further. Will not be able to type ‘0’ into the input-box. | Resolution decreased to 1 from the default of 5, in steps of 1, when the down button was clicked. It did not decrease further. It is not possible to type ‘0’ into the input-box. | **✔** |



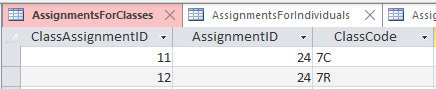
Minimum value is one when using the down button, limited by the code. Cannot write ‘0’ into the box, as typing input is disabled (hence the grey colour).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a panel for each class-group, and one for other individual students, that an assignment is set to (which will each contain a data-grid-view of details, tested in the next test) appear when an assignment is selected. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 72 | Normal | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | A panel should appear for each class-group which the selected assignment has been set to, and a panel for other students independent of a class-group. There should not be a panel for any other class-groups. | A panel appeared for each class-group the assignment is set to (7C and 7R), and a panel appeared for other students independent of a class-group. There is not a panel for any other class-groups, as is the expected behaviour. | **✔** |

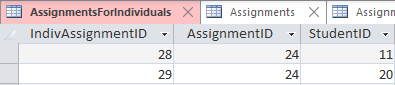


3 panels have appeared, for 7C, 7R, and other students. See below database screenshots to see that these are the groups that the assignment was set for.

Assignment is set to both 7C and 7R:

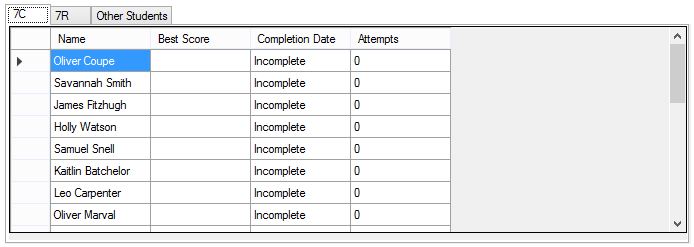


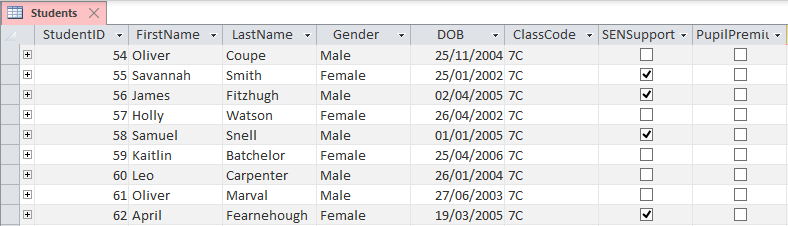
Assignment is also set to two other students, hence the presence of the ‘Other Students’ panel:

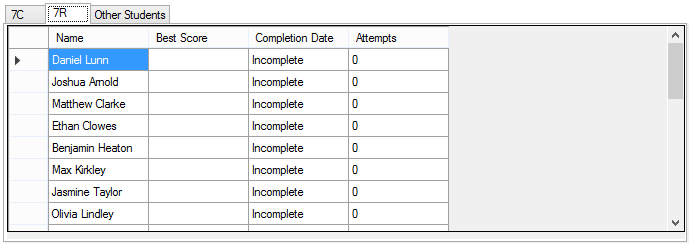


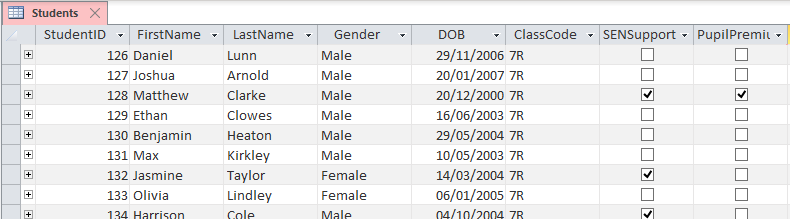
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct students (only ones for the class-group which the containing panel of the DGV is for), with the correct scores and statistics, appear in a data-grid-view within the panels. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 73 | Normal | Log in as teacher ‘Megan Taylor’. Click “Your Classes’ Assignments” button in menu to display respective panel. Select an assignment. | Each class-group panel should only contain students whose class-group is that which the panel is for (shown by the panel name). | Each class-group panel only contains students whose class-group is that which the panel is for. | **✔** |

Using the assignment from test 72 for this test:





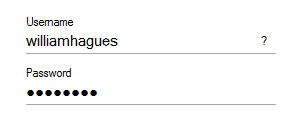


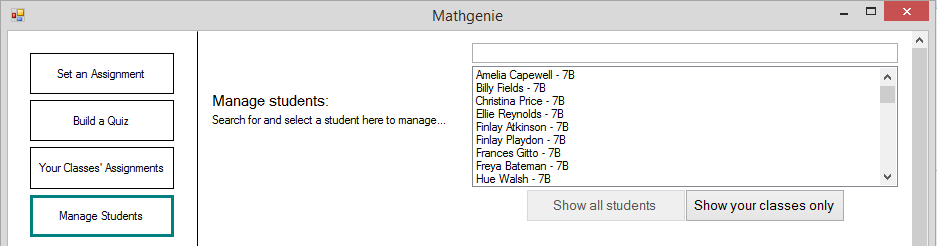


The two sets of screenshots show that the students displayed in the data-grid-views are members of the class-group which the data-grid-view is for. Therefore, the test was successful.

### Tests of the ‘Manage Students’ panel:

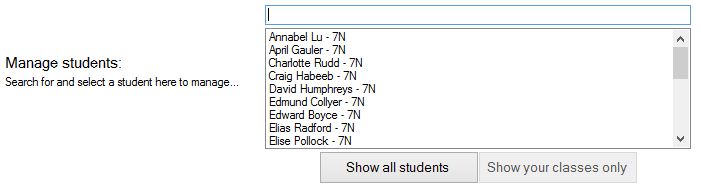
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that only students within class-groups taught by the logged in teacher appear in the student selection list-box, when the ‘Show your classes only’ button is clicked. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 74 | Normal | Log in as teacher ‘William Hagues’. Click ‘show your classes only’ in ‘Manage Students’ panel. | Before clicking the button, all students should be visible in the list-box. After clicking, only students of 7N, the class-group that William Hagues teaches, should be visible. | Before clicking the button, all students should were visible in the list-box. After clicking, only students of 7N, the class-group that William Hagues teaches, were visible. | **✔** |



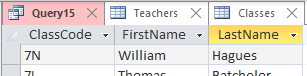


Clicked

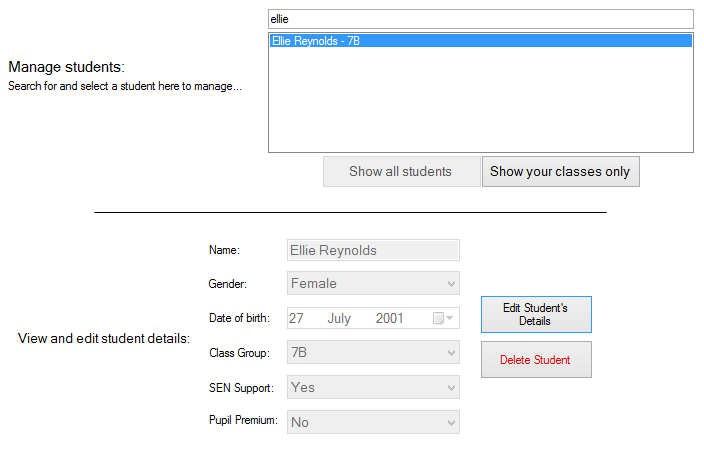
In the screenshot below, you can see that the first students displayed in the list-box are now from 7N (i.e. the students shown above, in 7B, are not shown). It can also be seen that the scroll bar is larger in the screenshot below, compared to the one above. This is because only students of William Hagues are being shown.



See below, that William Hagues teaches the class-group ‘7N’, and therefore the fact that only students from 7N are shown above shows that this test was successful:

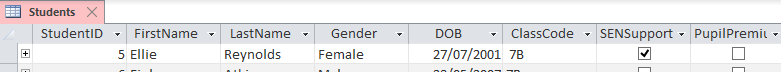


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the correct details for a student appear, when a student is clicked in the student selection list-box. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 75 | Normal | Click ‘Ellie Reynolds - 7B’ in the student selection list-box. | The details for Ellie Reynolds should appear. | The details for Ellie Reynolds appeared. See the database evidence screenshot to confirm her details. | **✔** |

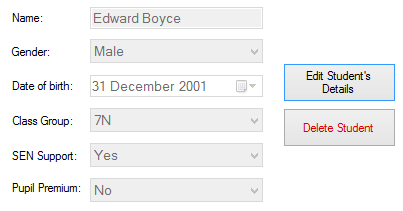
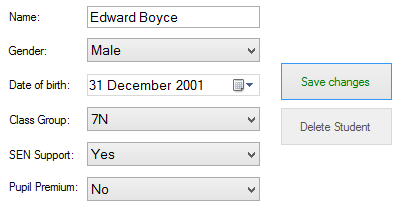


Correct details are shown here.

The details shown in the database screenshot below are the same as those displayed in the UI above. Therefore, test 75 is successful.



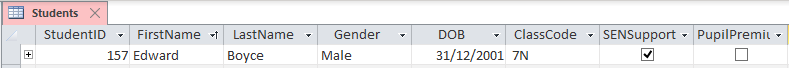
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that when the “Edit Student’s Details” button is clicked with a student selected, the input-boxes containing student details are enabled for editing. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 76 | Normal | Select student ‘Edward Boyce - 7N’. Click ‘Edit Student’s Details’ button. | The six input-boxes for student details should enable (i.e. no longer be read-only). | The six input-boxes for student details have enabled (i.e. are no longer read-only). | **✔** |

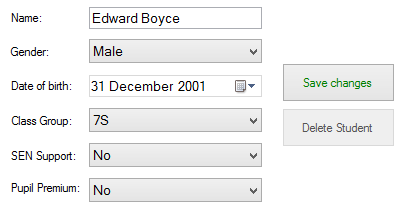
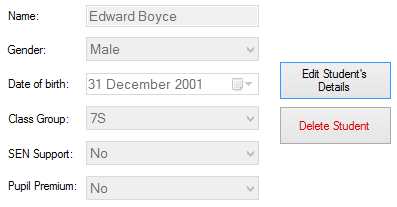
 

In the above screenshots, it can be seen that the input-boxes for student details are enabled when the “Edit Student’s Details” button is clicked. Therefore, test 76 is successful.

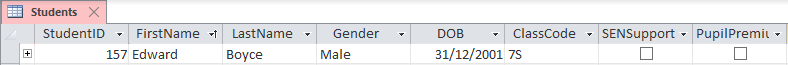
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Following on from the previous test, test that, after editing student details in the input-boxes and clicking the ‘Save Changes’ button, a student’s details are successfully updated in the database. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 77 | Normal | With ‘Edward Boyce’ selected: edit SEN Support to ‘No’, and change class group to ‘7S’. Click ‘Save Changes’ button, and click ‘Yes’ in message-box, to confirm. | Within the ‘Students’ table of the database, the row for ‘Edward Boyce’ should contain the updated values for SEN support and class-group, after save changes has been clicked. | Within the ‘Students’ table of the database, the row for ‘Edward Boyce’ contains the updated values for SEN support and class-group, after the save changes button was clicked. | **✔** |

Before:



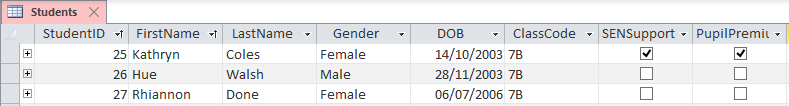
 

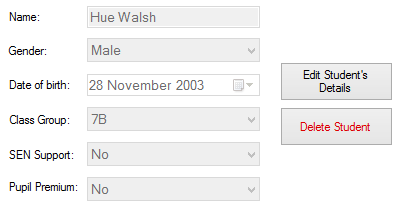
After - details in database have changed:



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that the ‘Delete Student’ button successfully deletes a student from the database. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 78 | Normal | Select student ‘Hue Walsh’, click ‘Delete Student’, click ‘Yes’ in confirmation message-box. | Before, there should be a row in the ‘Students’ table of the database for Hue Walsh. After, the row should have been removed. | Before, there was a row in the ‘Students’ table of the database for Hue Walsh. After, the row was removed. | **✔** |

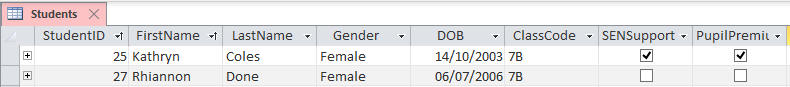
Before - it can be seen that Hue Walsh is present in the database:



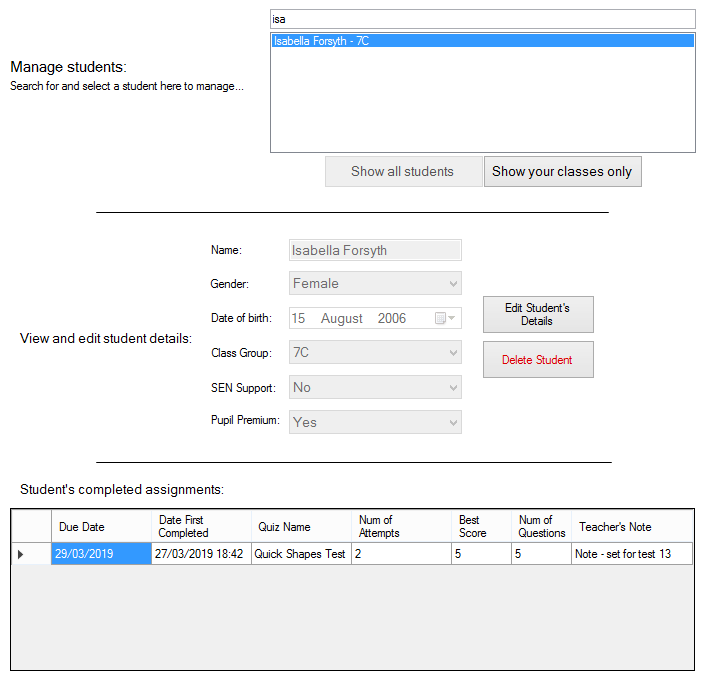


Clicked ‘Delete Student’ button.

After - Hue Walsh has been deleted from the database (you can see that his entry to the database is no longer present, between the two surrounding entries):



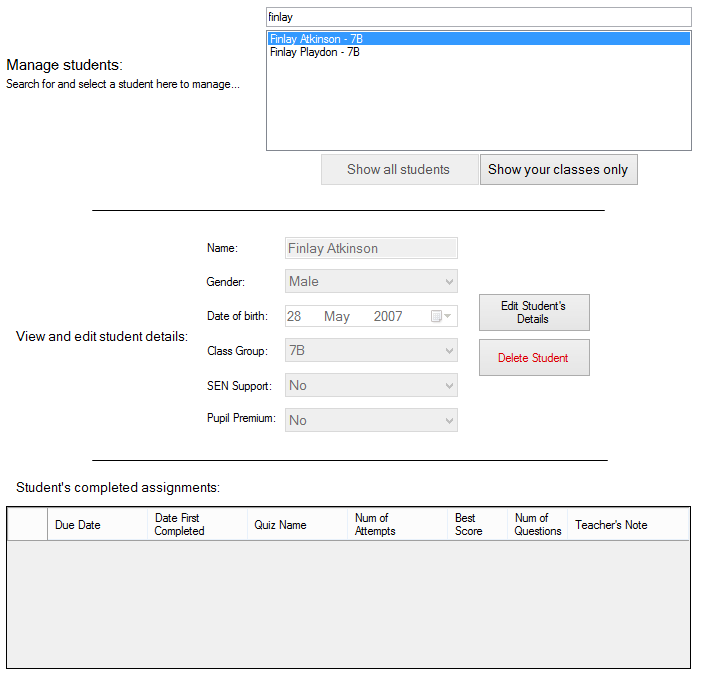
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that assignments completed by the student chosen in the student selection list-box are displayed in the “Student’s completed assignments” data-grid-view. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 79 | Normal | Select student ‘Isabella Forsyth’ (who completed two attempts of an assignment in test 15), in the student selection list-box. | One assignment should be displayed in the “Student’s completed assignments” data-grid-view. ‘Num. of attempts’ should be ‘2’, as Isabella Forsyth attempted the same assignment twice in test 15. | One assignment is displayed in the “Student’s completed assignments” data-grid-view. ‘Num. of attempts’ is ‘2’, as Isabella Forsyth attempted the same assignment twice in test 15. | **✔** |



One assignment is shown, with two attempts, as expected.

Isabella Forsyth is selected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Following on from the previous test, test that assignments completed by students other than the one chosen in the student selection list-box, are not shown in the “Student’s completed assignments” data-grid-view. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 80 | Erroneous | Select student ‘Finlay Atkinson’ in the student selection list-box. | No assignments should be shown in the data-grid-view (as the student has not completed any assignments) - in particular, the assignments completed by Isabella Forsyth are not visible. | No assignments are shown in the data-grid-view (as the student has not completed any assignments). As was expected, the assignments completed by Isabella Forsyth are not visible. | **✔** |



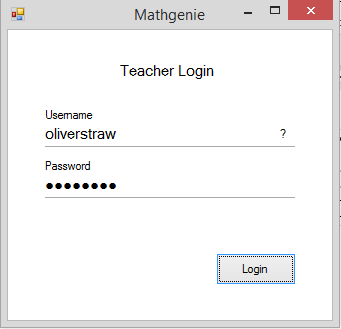
No assignments are shown. In particular, Isabella Forsyth’s completed assignments are not shown.

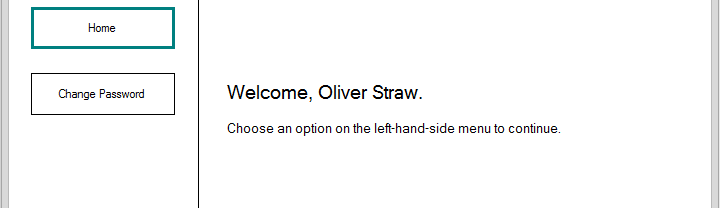
Finlay Atkinson is selected.

The previous two tests (number 79 and 80) can also be used to see that the search-box for students on this panel functions correctly.

### Other tests for the teacher-side of the system:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that a teacher’s name is displayed on the welcome text, on the teacher home panel. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 81 | Normal | Log in as teacher ‘Oliver Straw’. | “Welcome, Oliver Straw” should be displayed on the home panel. | “Welcome, Oliver Straw” was displayed on the home panel. | **✔** |



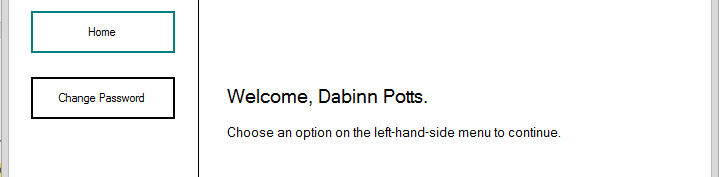


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test purpose: Test that passwords are changed successfully using the ‘Change Password’ button in the menu bar of the teacher-side of the system. | | | | | |
| No. | **Type** | **Test Data** | **Expected Result** | **Actual Result** | **Pass?** |
| 82 | Normal | Log in as ‘Dabinn Potts’, using ‘password’ as password. Change password to ‘hello’. | Password hash stored in the database should change. | Password hash stored in the database changed, therefore the password that the hash is for must have been changed successfully. | **✔** |

Before changing password:

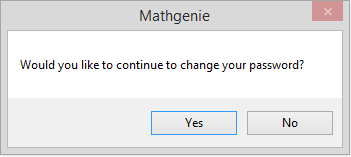


Changing the password:

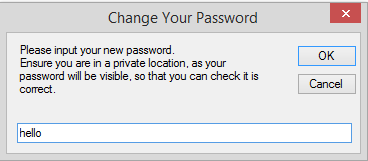


Logged in as Dabinn Potts.

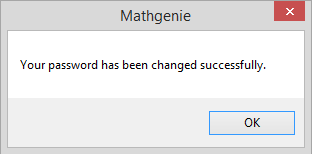
Clicked



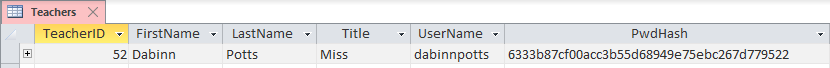
Clicked



Clicked



After changing the password to ‘hello’:



It can be seen that the password hash (column is named ‘PwdHash’) has changed between the database screenshots from before and after changing the password. Therefore, test successful.

# Evaluation

## Evaluative interview with the client, Mrs Taylor

I have conducted an interview with Mrs Taylor shortly after the development of the system was finished, and its use had begun within the mathematics department, in order to assess its performance in replacing the old paper homework system. The interview is as follows.

*Are you and other teachers in the Maths department finding the system easy to use?*

“Yes we are – the log in process is simple and quick, and the teacher system is intuitively organised and easy to navigate using the menu bar. I like how each function of the system is separated into its own section, which can be accessed from the menu. The user-interface feels natural to use and is visually similar to other software packages I have used in the past, so I had no problems getting to know the system.”

*What are your thoughts on the statistics the system provides regarding the assignments you have set, and regarding your students?*

“The score distribution graph and pie chart of completion which is automatically generated for each assignment is incredibly useful for quickly determining which assignments students have struggled with more. We had nothing like this with the paper homeworks, and had to manually count how many homeworks had been returned to us, and calculate an average score for a whole class-group, which was very time consuming. Regarding the statistics on individual students in the ‘Manage Students’ panel, the bar chart showing a breakdown of the number of questions a student has tried in each topic, and their success in these questions, is helpful for assessing which topics students are struggling with more and we therefore need to cover again in class.

*Do your students who have already tried the system find it easy to use?*

“So far, I have introduced the system to one of my year 7 classes as a replacement for the paper homework tasks that we used to use. The students have not had any trouble using the system; they seem to find it especially easy to use given that they do not need a keyboard. They find the initials log-in system and drawing their answers very easy to do, and it allows them to focus on the maths, rather than struggling to use the system. The student user-interface is organised and minimalistic as I requested, and students are having no problems finding each part of the program.”

*What are your thoughts on the feedback that students are given after assignments?*

“Students being told whether they have got a question right or wrong straight after doing the question is great – it allows them to rethink their strategy on any other similar questions later in an assignment, rather than continuously making the same mistakes. The automatic marking and storing of student scores in assignments is already saving me plenty of time with the class-group I have introduced the system to, and I can foresee it helping the other teachers of the mathematics department greatly too.”

*What about the ‘Your Progress’ panel that students can use to get feedback? Is that useful for students and yourself?*

“Students have been using the ‘Your Progress’ panel, alongside the ‘Practice Topics’ section of the system, so that they can be given direction on what topics they can practice in order to improve, without having to wait to see myself or another teacher to be given feedback. This prevents students from wasting time waiting to see a teacher, and saves our time too.”

*Are the questions, which are already stored in the system, the right skill level for the students?*

“The questions and variety of topics which are available for students to answer in the system are the correct skill level for the new year 7 students, who will be using this system. They are of similar difficulty to the paper homework that I gave you as an example, before the new system was developed, which is exactly the type of questions we are looking for.”

*Is the functionality of adding new questions easy to use?*

“Adding new questions to the system in the ‘Build a Quiz’ section of the teacher-system is quick and easy. I like how we will be able to add new questions of specific topics, if we find a topic that students are struggling with particularly.”

*Are there any improvements you would like a future version of the system to include?*

“A useful improvement to the system would be the addition of more statistical tools for teachers, to analyse statistics with. For example, further analysis of assignments that we have set, to pick out questions that whole class-groups have struggled with, would be useful so that we can go over those topics in class again. The score-distribution graph for each assignment is very helpful in allowing us to see which assignments students have been struggling on, but does not identify individual questions. However, the current statistics which the new system provides are far superior and easier to use than the analysis we previously did of the paper homeworks.”

*Overall, how well do you find the system meets your original criteria?*

“Overall, the system which you have developed is a significant improvement over the old paper homework system. It is already saving me time, which was previously used marking and analysing homework, which I will be able to use more effectively to give students extra help with their maths. I am sure that the saving of time which the system is giving me will be seen by other teachers too, when they introduce the system to their class-groups.

Regarding my other criteria, my students are finding the system easy to use, especially with their touchscreen laptops and tablets. Setting assignments is a very fast process, and the fact that students are given immediate feedback on their assignments, rather than having to wait a week for them to be marked, is very helpful. Therefore, the system meets my criteria very well.”

## Review of specification

I will review each point of the specification, evaluating whether the technical solution for the system has met each specification criteria, based on the testing of the system and the evaluative interview.

### Review of specification part I: Specification for student-side of the program

1. Students are able to log in using a simple, fast and easy manner, by writing their initials using a mouse or touchscreen.
   1. Security of log-in is not important - therefore, the method used in the technical solution to deal with multiple students matching inputted initials is acceptable.
2. Students are able to view and complete assignments which have been set to them by a teacher.
   1. Questions in an assignment which the student has chosen to do are displayed to the student one-by-one.
   2. A student is able to draw/hand-write their answer using a mouse or touch-screen, rather than typing in an answer with a keyboard.
   3. The student’s answer to a question is immediately automatically marked.
   4. The student then receives instant feedback on whether they got the question correct or incorrect.
   5. The student is given instant simple feedback after they complete an assignment.
      1. They are informed how many questions they got wrong and how many they got correct.
      2. They are given some form of additional feedback based on their score which is easy for a young user to understand - this is the emotion-faces, which are based on their score percentage in a quiz.
   6. The student’s result in an assignment are stored for future analytics and statistics.
3. Students are able to practice maths topics of their choosing, in their own time, without any input from or tasks being set by a teacher.
   1. Quizzes on any combination of chosen topics are automatically generated for the student to complete.
   2. The student is able to choose how many questions they would like their quiz to contain.
   3. The student is given instant simple feedback after they complete a quiz, the same as detailed in specification I, part 2e.
4. Students are able to view their past attempted assignments, including their score and date they made the attempt.
   1. These are displayed, listed, to the student in a large and clear format - the SuperListBox UI control.
   2. Students are able to attempt past assignments again with the click of a button - the action buttons on SuperListBoxes.
5. Students are given automatically-generated feedback on their progress in learning question topics - in the ‘Your Progress’ panel.
   1. This adapts as a student progresses with completing more assignments and questions.
   2. The feedback is given in a manner that is easy for a young student user to understand - 3 short dynamically-generated sentences, and an easy to read bar chart.
6. Students are able to interact with the program without the use of a keyboard for any purpose - both log-in and question answering is done by drawing.
7. There are no complex design features in the user-interface of the student-side of the program, in order to make it easy to use for the young end-users.
   1. Buttons are large and clear.
   2. There are no tabbed interfaces or menus which could be difficult to navigate and understand - instead, everything is accessible from a single ‘home’ panel.
   3. The one complex UI feature - the graph in the ‘Your Progress’ panel - has text explaining its use to the user.
8. All of the details of students, such as their name, gender, class-group, and teacher are stored in a database. All data regarding a student’s assignments which they have been set, and a history of their past assignments and question attempts, are stored in the same, single database also.
   1. All data is stored in a single database, which can be stored on a shared drive accessible everywhere in the school, to simplify data storage so the program can run on any student laptop whilst connected to the school network.

### Review of specification part II: Specification for teacher-side of the program

1. Teachers are able to log in with a username and password, using a simple and easy-to-use user interface.
   1. The security of the teacher-side of the program is critically important (especially compared to that of the student-side), since access to a teacher account would allow someone to view and change confidential student details. Therefore, password hashing, password salting, and obfuscating password input are security measures taken to ensure the log-in process is secure.
   2. Teachers are able to change the password they use to log in, once logged in to the system, so that they can change it if accidentally shared, to ensure system security.
   3. Usernames are easy to remember.
2. Teachers are able to easily and quickly create and set assignments to students.
   1. An assignment is able to be set to any number of whole class-groups, and any number of students independently of a class-group.
   2. Teachers are able to choose what questions are in their assignment. This is done by creating and choosing a quiz to set for the assignment.
   3. Teachers are able to specify a deadline for assignments.
   4. Teachers are able to write a note when setting an assignment.
3. Teachers are able to create a quiz, which can be set as an assignment to students.
   1. Teachers are able to create and add questions to a quiz, by inputting a question and its answer.
   2. A quiz is able to contain any number of questions (at least one), made up of any combination of question topics.
   3. Teachers are able to name their quiz.
   4. Quizzes have the option to appear to students in the order teachers create them, or in a random order.
4. Teachers are able to view the assignments they have set to class-groups and students.
   1. Details on whether a student has completed an assignment or not, along with a student’s best score and number of attempts, are shown for all students who an assignment has been set to - in the data-grid-views on the “Your Classes’ Assignments” panel.
   2. Statistics on assignments are displayed in graphs, including a chart of completion of each assignment, and a bar chart of score-distribution across an assignment.
5. Teachers are able to view statistics about their students that they teach. This includes a history of assignments that a student has completed, and a bar chart of their scores across the different question topics - both within the ‘Manage Students’ panel.
6. The user interfaces of the teacher-side of the program follow the same design characteristics that most other Windows programs have, in order for teachers to be able to learn to use the program quickly and the UI seem familiar and easy to use. These characteristics are as follows:
   1. UI panels have white or grey backgrounds.
   2. Buttons are the standard Windows grey pop-up buttons - apart from the menu buttons, but these benefit from the ability to be highlighted to show which panel is currently displayed.
   3. Data input boxes comprise only of text-boxes, combo-boxes, date-time pickers and numeric-up-down boxes - standard Windows UI controls.
   4. The UI is easy and fast to use - important since one of the key purposes of this system is to be faster than the old paper system.
7. All parts of the user interface are within one Windows Form, rather than spread across multiple forms, to make switching between parts of the UI more fluid than if a new window was to open. The one exception to this is the ‘Train AI’ form, which is in a separate form as it will rarely be used, so load-time can be reduced by having it separate.
   1. The UI is kept organised by using a tabbed interface, to separate key parts of the program intuitively.

### Review of specification part III: Specification for the canvas user-interface control

1. The user is able to draw in a ‘canvas’ using a mouse or a touchscreen, and is able to draw characters including letters, numerical digits, other symbols (e.g. +, !, ?, etc.), and shapes. This is done using the Canvas class.
2. The line drawn by a canvas is smooth and clear:
   1. Corners and ends of lines are rounded.
   2. Antialiasing is applied to drawn lines to smooth diagonal sections.
   3. There are no gaps in a line that is intended to be continuous, and there are no visual artefacts in drawn lines.
3. Only one colour is available for the user to draw with - black has been chosen, as it is appropriate for all purposes in this system. This was specified as being appropriate in the original specification.
4. The thickness of drawn lines is changeable in the set-up code for a canvas.
5. There is functionality for a user to clear any drawing on a canvas that they have made - the Canvas.ClearCanvas() function, called by a button press, can be used to do this.
6. Any specific canvas needs to be able to be cleared by calling a function within an algorithm - again, this can be done using the Canvas.ClearCanvas() function.
7. The canvas controls are able to be generated in any form of the program, by calling the Canvas.CreateCanvas() function. Each canvas has these following properties able to be set when created, in order to serve different purposes:
   1. Width and height of canvas.
   2. Position of canvas within the form or panel it is being created in.
   3. Whether the canvas has a border line to show the edges of the canvas, or not.
8. The drawn contents of a canvas are able to be retrieved in bitmap format, for further use or processing - for example, for use by the AI character classification algorithm.

### Review of specification part IV: Specification for the artificial intelligence character recognition engine

1. The AI method for character recognition is callable as a single function from anywhere in the end-user side of program, passing the necessary inputs (a bitmap and the classification type) into the function and returning the output label of the bitmap passed in. The AI.ClassifyImage() function is used for this.
   1. Essentially, only one line of code is required to be written to get an output label from an input bitmap.
   2. The artificial intelligence method operates in the background, ‘invisible’ to the end-user, without any control by student users being necessary for its operation.
2. The algorithm is able to classify hand-drawn digits (0-9), upper-case alphabetic characters (A-Z), symbols (e.g. +, -, etc.), and shapes from the standardised processed input image.
   1. A label is assigned to said image, which is a string representing what the AI method ‘thinks’ is the most likely character that is drawn in the image.
   2. This label is returned from a function as a string type variable.
3. A bitmap image containing a single drawn character can be passed as a parameter into the function in bitmap format.
   1. This image is able to be:
      1. Any dimensions
      2. Either black and white or any colour or combination of colours (a coloured image can be converted to B/W for the algorithm, using the AI.BMConvertToBW() function.
4. Another input parameter for the AI character classification function is the type of character that the input image contains, i.e. is it a letter (A-Z), a number (0-9), a symbol, or a shape. This is used to improve the accuracy of the character recognition, by reducing the number of possible labels for the AI to choose between to output.
5. The bitmap image which is passed to the AI character recognition engine function is cropped to remove any white-space surrounding the character.
6. The bitmap image is compressed to a small standardised size of 20x20 pixels, once white-space has been cropped out, to improve the speed of the classification algorithm and reduce the file-size of the data that is stored in the training dataset text-files (the largest of these text-files, containing 1500 training items, is only 1131 KB in size).
7. The AI character classification algorithm executes completely in **just 0.25 seconds** - half that of the 0.5 second target which was specified in specification part IV, section 7. Evidence for this can be seen in the prototyping of the AI algorithm within the designs section of this documentation, by observing the screenshots of the console output.
   1. This means that the end-user does not need to wait for any noticeable amount of time whilst the AI algorithm is operating.
8. The AI function finishes executing within the above time specification on low-performance hardware - the time period of 0.25 seconds for the AI algorithm was achieved using a dual-core 15 watt laptop CPU, and a basic SATA 6Gbit/s SSD - very reminiscent of the hardware present in the laptops and tablets which students will be using the system with.
9. It is possible to train the AI to recognise individual additional characters or symbols - this is done using the AI.AddBMToTrainingSet() subroutine.
   1. This are a reasonably fast process - the AI is able to recognise a new character with reasonable accuracy from just 5 input example images.
10. The training of the artificial intelligence method for exhaustive types of input (i.e. characters A-Z and numbers 0-9 where there is a limited number of possible labels, since there are no more characters than A-Z and no more number digits than 0-9) was done once, in the prototyping stage of system development, by creating training data-set text-files.
11. The AI character recognition algorithm is able to label characters correctly, with the specified accuracy of 90%. Evidence of this can be seen throughout the documentation.

## Is this actually artificial intelligence?

Artificial intelligence is a very broad and varied subject, and whether a specific algorithm is truly artificial intelligence is often debated. Generally, an algorithm is defined as being artificial intelligence if it is both **autonomous** and **adaptive**.

The character classification algorithm that is specified within specification part IV is autonomous, because it assigns a label to a hand-drawn character without human intervention. The algorithm does not use reinforcement learning (improving accuracy as it is used more), since as per specification IV, part 2b, the AI method should be invisible to the student users. However, it can still be classed as having some adaptive behaviour, since it is able to learn new characters to recognise, as stated in specification IV, part 9.

Artificial intelligence is also often defined as any computer algorithm showing human behaviour or solving a task that would be thought of as requiring human intelligence. Recognising what character a hand-drawn character is, is very much a task that would be thought of as requiring human intelligence.

The algorithm to recognise hand-drawn characters is autonomous, adaptive, and solves a human task. Therefore, it can be strongly argued that the algorithm is indeed artificial intelligence.

## Conclusion

As can be seen by the above review of the specification, my technical solution meets all of the criteria set out in the specification which was defined prior to the development of the system. This is evidenced by the extensive testing which I have carried out and documented. Mrs Megan Taylor, the client for the system, was pleased with the technical solution that I have developed, and will soon be putting it into use for new students to the school.

Despite the success of the testing of the technical solution, the primarily-positive feedback from the evaluative interview, and the completeness of the solution as shown by the review of the specification, there are improvements that could be made my technical solution for the system, if not for the impediment of time.

The clearest imperfection in the technical solution of the system is the 90% accuracy of the AI character classification algorithm. This 10% inaccuracy results in the occasional question being falsely marked (be it false-positive or false-negative), and drawn initials in the student log-in system occasionally being labelled incorrectly. This could be improved by implementing a neural network artificial intelligence method, instead of my chosen nearest-neighbour AI method. A complex neural network with many layers (a layer is a group of neurons, which in turn process an input into an output - in this case an input bitmap into an output string) can be used to achieve accuracy of classification within the region of 98-99%. This would result in many less falsely marked answers to questions by the AI, but of course there would still be the occasional, albeit rare, incorrect output from the AI. It is an impossible task to achieve 100% accuracy of character classification, since what character a hand-drawn character really is could be debated between two people - for example, it could be debated whether the character ‘I’ is the 9th letter of the alphabet, or the number one.

However, the use of the nearest-neighbour artificial intelligence method (which is the method I used for the system) allows the use of relatively small datasets and therefore incredibly fast training of the AI (for example, the training of the AI by teachers to mark new answers, simply by drawing five images of the answer that is being learned). If a neural network was implemented instead of the nearest-neighbour method, the benefits of fast training would be lost, which make the task of having user-friendly AI training particularly problematic. Implementing a neural network into the system would add significant complexity to the development process, which, over-and-above the complexity of the employed nearest-neighbour artificial intelligence method, would not have been possible in the time period I had for the development of my system.

Another possible improvement to the system, which was identified in the evaluative interview with Mrs Taylor, would be the addition of more statistical tools into the system for teachers to use. She stated that the statistics about students and assignments that teachers are provided with by the system are very useful, compared to the previous method of calculating and documenting statistics on paper-assignments beforehand. However, she stated that a future version of the system could automatically identify and pick out questions and topics in assignments which the whole student population struggle with, and inform teachers of this so that they can assist whole class-groups all together. This would be in addition to the statistics about students’ worst topics that the system currently provides, on a student-by-student basis.

Overall, it can be concluded that the system is a complete and extensive solution to replace the paper homeworks that Mrs Megan Taylor, and other teachers of the Ecclesbourne School mathematics department, previously used. Testing has been successful, the technical solution is complete, and my client is pleased with the finished system.