Science for Kids

IMAT3451 Final Year Project



Contents

[1 Introduction 3](#_Toc512207276)

[2 Requirements Specification 5](#_Toc512207277)

[3 Program Design 7](#_Toc512207278)

[3.1 GUI (Graphical User Interfaces) & Related UML Diagram 7](#_Toc512207279)

[3.2 Model, View & Controller UML 15](#_Toc512207280)

[3.3 Use Case Description 19](#_Toc512207281)

[3.4 Entity Relationship Diagram 24](#_Toc512207282)

[4 Development/Implementation 25](#_Toc512207283)

[4.1 MVC 25](#_Toc512207284)

[4.2 Coding Views 26](#_Toc512207285)

[4.3 Coding Models 31](#_Toc512207286)

[4.4 Coding Controller 34](#_Toc512207287)

[5 Testing 40](#_Toc512207288)

[5.1 Testing Against Use Case Descriptions 40](#_Toc512207289)

[5.2 Testing Project Against Objectives 44](#_Toc512207290)

[5.3 Testing Against Non-Functional Requirements 46](#_Toc512207291)

[6 Critical Analysis and Conclusion 47](#_Toc512207292)

[7 Appendix 50](#_Toc512207293)

[7.1 Project Contract 51](#_Toc512207294)

[7.2 Ethical Review Form 53](#_Toc512207295)

[7.3 BCS Checklist 57](#_Toc512207296)

[7.4 Gantt Chart 59](#_Toc512207297)

[7.5 Literature review 60](#_Toc512207298)

[7.6 UML Diagram before 65](#_Toc512207299)

[7.7 UML Diagram After 66](#_Toc512207300)

[7.8 ERD Diagram 69](#_Toc512207301)

[7.9 User Manual 70](#_Toc512207302)

[7.10 Progress Reports 75](#_Toc512207303)

[7.11 createUserEventHandler code 80](#_Toc512207304)

[7.12 loginEventHandlerCode 81](#_Toc512207305)

[7.13 LearningScreenPane TakeQuizBtn YoungerQuiz Code 82](#_Toc512207306)

[7.14 LearningScreenPane TakeQuizBtn OlderQuiz Code 83](#_Toc512207307)

[7.15 YoungQuizScreen Marking Question Answer Code 84](#_Toc512207308)

[7.16 OlderQuizScreen Marking Questionnaire Code 85](#_Toc512207309)

[7.17 JUnit Final Test Image 86](#_Toc512207310)

# Introduction

*The Program*

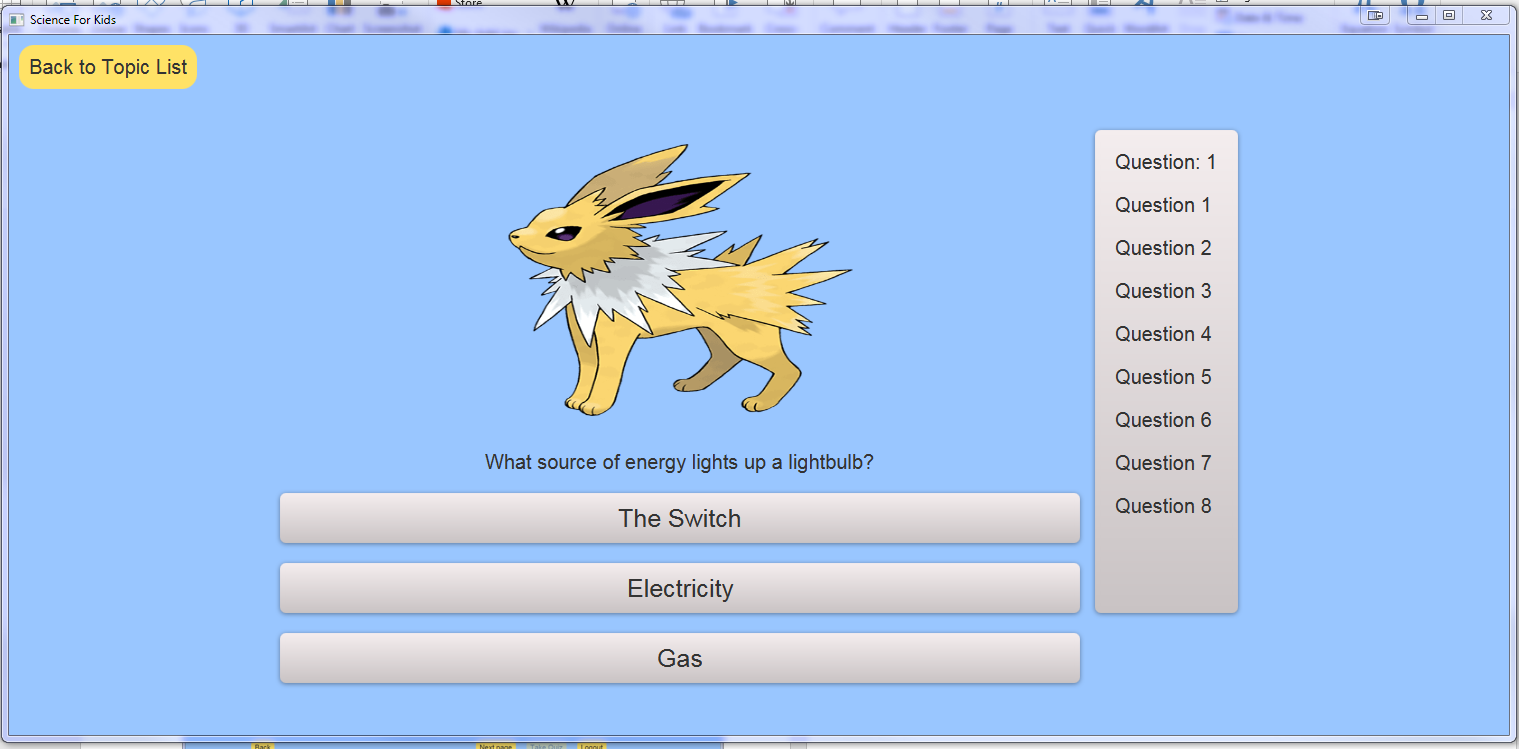
ScienceForKids is a Java Application targeted at kids between the ages of 7 – 12.

It allows the user to create their own account and select from a multitude of different learning materials of varying difficulties from Year 4 to Year 7. The learning material is in-line with the national curriculum, after they’re satisfied with their studying the user can choose to be quizzed on what they've learnt. Years 4 - 5 have a simpler quiz format compared to Years 6 – 7 which have a more recognised exam style quiz.

If the user is logged into their own account the program will save their topics completed and a counter of lifetime marks achieved after they have completed the quiz to the database, if they don’t complete the quiz by either not selecting an answer or not pressing the ‘Mark Questionnaire’ button then the topic will not show up in their topics completed column in the database and the total marks will not be updated.

If the user wishes to not record any information they can simply become a 'Guest User' by selecting the 'Continue as Guest' option from the main page, this allows access to the whole program with no 'strings attached'.

An Image below of the Year 4 – 5 Quiz Screen can be seen below; more Images can be seen [here](#_User_Manual).



These learning materials can be easily updated and added to by accessing the MySQL database.

The database contains all the data used by the program, all the images, questions and user data. Creating SQL scripts to add new learning material is very simple and after a couple of uses, can be mastered by anyone, in the future if this became a real product the database would migrate to an online server, and because of the live connection from the program to the database anyone using the program can instantly access new learning material the moment the script is run and database updated.

*Why?*

This project is a personal interest of mine, whilst at school I did not like my science lessons as they were very tedious, and I could not concentrate leading to an average grade and being unable to pursue the subject further. Outside of school I found various websites and YouTube channels dedicated to making science interesting, so when I had my own learning autonomy I developed an interest in physics, astrophysics to be exact.

I also have a little sister who intensely dislikes the teaching of science in her all girls secondary school, but when I talk to her about various science-based subjects is very engaged and seems to show an interest.

Therefore, I want to make a program that can help young people through the boring teachings of science and maybe stimulate them to do further research and learning so that all doors remain open to them throughout their life as a student.

To do this the application is aimed to be used by teachers and parents to better engage their kids in science, using modern technologies as an aid for regular teaching. The application will aid kids learning of science but will not be a replacement for real life school teaching.

I chose to develop the product in Java, firstly because it was the language I felt most comfortable with at the time and had previous experience with smaller projects and programs that I had written in my second year of university; and secondly because my future interests at the time lay in software development with Java software development at the top of the list.

Integrating various software development methods and implementing them in this project will surely help my development as a software developer, this project also gives my portfolio and experience more substance.

*The more general goals of this project are:*

1. To produce a program that looks and acts professionally.
2. To gain a deeper understanding of the problems surrounding women’s low engagement in STEM subjects.
3. To enhance my basic knowledge of Java and SQL.
4. To gain an understanding of how Java and MySQL databases interact.

I hope this has been an informative introduction and has somewhat sparked your interest in this project, the rest of this document shows the programs designs, research done, decisions made and how many ideas were transformed into one concrete Java Application.

# Requirements Specification

The requirements specification of a project starts before any sort of system development, it documents the functional and non-functional requirements the system must have for a successful project. These can change anytime during the project evolution and can help prevent a software projects failure.

Functional Requirements

|  |  |
| --- | --- |
| *Login Screen:* | |
| Login BUTTON | Checks with database that a correct Username and Password has been entered and loads up their account. |
| Register BUTTON | Opens a form for the user to create a new account. |
| createAccount BUTTON | Saves the new username and password the user has inputted into their respective fields in the database. |
| Close BUTTON | Closes the program. |
| Select Year of study | |
| Year4 BUTTON | Links to relevant subject material for their learning level on next screen. |
| Year5 BUTTON | Links to relevant subject material for their learning level on next screen. |
| Year5 BUTTON | Links to relevant subject material for their learning level on next screen. |
| Year6 BUTTON | Links to relevant subject material for their learning level on next screen. |
| *Select Topic Screen: (Chemistry, biology, physics)* | |
| Big BUTTON for Chemistry | Links to subtopic screen. |
| Big BUTTON for Biology | Links to subtopic screen. |
| Big BUTTON for Space | Links to subtopic screen. |
| Close BUTTON | Closes the program. |
| Each topic screen has a couple sub topic buttons.  *Sub Topic screens:* | |
| BUTTON for each subtopic | Links to relevant page with learning materials. |
| goHome BUTTON | Returns to the ‘Home Screen’ which is defined as the ‘Select Topic Screen’. |
| Close BUTTON | Closes the program. |
| *Information Screens:* | |
| nextPage BUTTON | Links to next page of Learning Material. |
| prevPage BUTTON | Links to previous page of Learning Material. |
| goHome BUTTON | Returns to the ‘Home Screen’ which is defined as the ‘Select Topic Screen’. |
| takeQuiz BUTTON | Links to a quiz ‘form’ the user can complete. |
| *Quiz Screens:* | |
| Mark BUTTON | Checks answers inputted vs correct ones and links to Feedback Screen where their score will be shown. |
| goHome BUTTON (include warning) | Returns to the ‘Home Screen’ which is defined as the ‘Select Topic Screen’. Warning box will appear to make sure the user doesn’t accidently click go home during quiz. |
| *Feedback Screens:* | |
| backToTopics BUTTON | Returns to the subtopics screen. |

Non-Functional Requirements

* There will be sufficient Documentation for first time users to understand how to use the program properly.
* There will be sufficient Documentation of system design for future programmers to understand how each function should work in case of future faults.
* Disaster Recovery: 2 Backups shall be made once a week, one in solid form on an external drive and online.
* Deployment: This program shall be made downloadable via a CD.
* Users attempting to login more than 4 times in a minute shall have an error message appear and the program will close.
* No personal data about the user shall be stored in the databases.
* This program shall be able to run by users working with low spec computers.
* None of the programs code will be accessible by users.

# Program Design

This Design section contains the GUI & Related UML Diagrams, Use Case Descriptions and the ERD.

As this projects development methodology is half waterfall half iterative, this section falls under the waterfall half, all ideas of the expected graphical layouts, class UML and expected Entity-Relationships are contained within.

Each GUI Design has its related class UML underneath with an explanation of its variables and functions. The entity relationship diagram in the appendix is linked alongside its text description, and the use cases describe the most important functions of the system.

## GUI (Graphical User Interfaces) & Related UML Diagram

You can see the entire UML Diagram before the iterative development started [here](#_UML_Diagram)

Event Handler functions will be explained in the Controller UML.

The UML has been designed with basic knowledge of Java, once my knowledge of this programming language evolves, the UML Diagram will be sure to evolve alongside it.

Click [here](#_Coding_Views) to see GUI Coding Development

|  |  |
| --- | --- |
| Main Menu / Register Screen | |
|  | |
| https://i.gyazo.com/e745fe5f8794f11d1ac9f2df239899ab.png | newUsername & newPassword are two Strings that are retrieved from the Username and Password TextFields.  registerBtn Triggers addCreateUserEventHandler.  guestBtn triggers addGuestEventHandler.  loginBtn triggers addMainScreenLoginEventHandler.  The MainScreenPane function returns the entire MainScreenPane class so it can be accessed by the Root Pane during runtime.  getNewUsername & getNewPassword functions return the saved newUsername & newPassword Strings. |

|  |  |
| --- | --- |
| Login Screen | |
|  | |
| https://i.gyazo.com/5c86b81acc303e31f5bd5a99cf34c74b.png | username & password Strings respectivley contain the username & password Strings the user inputs.  loginBtn triggers addDefaultLoginEventHandler.  goBackBtn triggers addGoBackEventHandler.  LoginScreenPane function returns the entire LoginScreenPane class so it can be accessed by the Root Pane during runtime.  getUsername & getPassword functions return the saved username and password Strings. |

|  |  |
| --- | --- |
| Selecting level of study | |
|  | |
| https://i.gyazo.com/2b1410360c6372697a31f2b090f2c372.png | username is a String that will be loaded into a Label in the pane by the use of the setUsername function.  studyYear is an int that saves the studyYear selected.  year3Btn, year4Btn, year5Btn, year6Btn & year7Btn Buttons trigger their respective event handlers:  addYear3EventHandler  addYear4EventHandler  addYear5EventHandler  addYear6EventHandler  addYear7EventHandler  LevelOfStudyPane function returns the entire LevelOfStudyPane class so it can be accessed by the Root Pane during runtime. |

|  |  |
| --- | --- |
| Choosing which subject to study | |
|  | |
| https://i.gyazo.com/56a2fd0d45dad308157bf55e6ee0582a.png | studyYear is an Int that is loaded into a label by the setStudyYear function. This is purely visual.  physicsBtn, biologyBtn & chemistryBtn trigger their respective event handlers:  addPhysicsEventHandler  addBiologyEventHandler  addChemistryEventHandler  goBackBtn triggers addGoBackEventHandler  ChoosingSubjectPane function returns the entire ChoosingSubjectPane class so it can be accessed by the Root Pane during runtime. |

|  |  |
| --- | --- |
| Subject Content Homepage | |
|  | |
| https://i.gyazo.com/f23ad8b898b5a379b7fed097d2f694be.png | subjectChosen is a String that is loaded into a Label by the setSubjectChosen function. This is purely visual.  studyYear is a Int that is loaded into a Label bt the setStudyYear function. This is also purely visual.  ChoosingTopicPane function returns the entire ChoosingTopicPane class so it can be accessed by the Root Pane during runtime  getTopicChosen returns the listView that contains all of the topics loaded in from the relevant subject & study year fields in the mySQL database.  addTopicEventHandler is triggered by clicking on a topic within the listView. |

|  |  |
| --- | --- |
| Example Learning Screen | |
|  | |
| https://i.gyazo.com/b09117b020b6e660f91f8319df158a7a.png | txtSubjectMaterial is a TextArea that is filled with the use of the setTxtSubjectMaterial function.  nextPage Button triggers the addNextPageEventHandler.  topicChosen is a String containing the name of the Topic Chosen to study, it is loaded into a Label by the use of the setTopicTitle function. It is purely visual.  takeQuiz is a Button that triggers addTakeQuizEventHandler.  LearningScreenPane function returns the entire LearningScreenPane class so it can be accessed by the Root Pane during runtime  getTxtSubjectMaterial returns the TextArea the will contain the Subject Material. |

|  |  |
| --- | --- |
| Quiz Screen for ages 7-10 | |
|  | |
| https://i.gyazo.com/419740d6d94bf5b4197e3edb8fdea014.png | questionText is a TextArea that the question is loaded into using the setQuestionText function.  questionNumber is an int that is also used to set the question number label, the setQuestionNumber function is used to set it, and to return the int getQuestionNumber is used.  answer1Btn, answer2Btn, answer3Btn Buttons trigger their respective event handlers:  addAnswer1BtnEventHandler  addAnswer2BtnEventHandler  addAnswer3BtnEventHandler  backToTopicsBtn triggers addBackToTopicsBtnEventHandler.  YoungerQuizPane function returns the entire YoungerQuizPane class so it can be accessed by the Root Pane during runtime |

|  |  |
| --- | --- |
| Quiz Screen for ages 11-12 | |
|  | |
| https://i.gyazo.com/6d253b775115e3ad850b5b313fb92a33.png | questionText is a TextArea that the question is loaded into by using the getQuestionTextObj.  answer1ComboBox, answer2ComboBox & answer3ComboBox contain all the multiple choice answers to each question and are populated by the function populateComboBoxes.  markQuestionnareBtn triggers addMarkQuestionnaireBtnEventHandler.  backToTopicsBtn triggers addBackToTopicsBtnEventHandler.  overallMark is an int the displays the question number it is set by the setOverallMark  function, it is purely visual.  OlderQuizPane function returns the entire OlderQuizPane class so it can be accessed by the Root Pane during runtime. |

## Model, View & Controller UML

|  |  |
| --- | --- |
| ScienceForKidsRootPane | |
| https://i.gyazo.com/9ed264b2d9758f3f0ae2eebd59e97d76.png | A new instance of ScienceForKidsRootPane is created each time the program is loaded.  Each pane will be accessed by the controller through the ScienceForKidsRootPane, e.g. When a Logout Button is pressed, the controller will get the MainScreenPane.  The controller also has access to every single function within each of the View Panes through this Root Pane. E.g.: When the Label needs to be set for the topic in the learningScreenPane, we access it like so:  view.getLearningScreenPane().setTopicTitle(); |

Click [here](#_Coding_Models) to see Model coding development

|  |
| --- |
| ScienceForKidsProfile |
| https://i.gyazo.com/a0141eac17d1a0e2ced7f36838525cee.png  ScienceForKidsProfile is the main model for the program, the controller will have access to all of the methods contained in the other 3 models through it.  The function names are self descriptive.  The user table in the database is reflected in this model as it has a total number or Marks, a total number of Topics Completed and the functions of the user model.  The same can be said for the subjectMaterial and the Questionnaire models.  The getSubjectMaterial and setSubjectMaterial methods allow us to retrieve the information stored in the relevant tables in the database and save it into the subjectMaterial model.  The getQuestionnaire and setQuestionnaire methods allow us to retrieve the information stored in the relevant tables in the database and save it into the Questionnaire model. |

|  |  |
| --- | --- |
| User | |
| https://i.gyazo.com/e8873e87bc77cf6758baddd5e8f97d43.png | This model allows the controller to quickly setup a new User when the user is logging into the program or creating an account.  It is a lightweight model and has limited useage. |

|  |  |
| --- | --- |
| SubjectMaterial | |
| https://i.gyazo.com/656c28380c9de06a88dbf9061abf0e7d.png | The subjectMaterial Model is needed 4 times, when the user has chosen the study year, subject, topic and if they wish to take the end of topic quiz.  setTopicYear is used when the user has chosen their level of study. E.g. Year ‘4’ (4 only saved as variable is an int).  setSubject will be used when the user has chosen the subject they wish to study e.g.: Physics.  These variables are used to accurately select the various topics to load into the choosingTopicPane’s listView from the database.  setTopicName will be used when the user has selected the topic they wish to study from the listView.  This action will also use the setTopicContents function to save the chosen topics contents into the model and load it into the LearningScreenPane’s TextArea for the user to read. |

|  |  |
| --- | --- |
| Questionnaire | |
| https://i.gyazo.com/961ef63b3a7300f60c927787c3677fd0.png | The Questionnaire Model is used ever time a question is to be loaded into the program. E.g. with the youngerQuizPanes multiple choice buttons, after the user selects an answer a new question is needed.  All of the questions are loaded into the program using the saved Topic ID from the database to identify them.  setQuestionNumber hold the questions number in the Questionnaire.  setQuestionText holds the text for the question.  setQuestionAnswer holds the answer in String form.  setQuestionAttempted is a Boolean that it initially false, then turns true when a question is completed.  setMulitpleChoiceAnswers holds the String of multipleChoiceAnswers in the database to be Split and assigned a button/combo box space. |

Click [here](#_Coding_Controller) to see Controller Coding Development

|  |  |
| --- | --- |
| ScienceForKidsController | |
| https://i.gyazo.com/bf8a92f6d011b1ba29b1e964e5710646.png | **mainScreenPane Event Handlers**  *createUserEventHandler –* Communicate with database to check if user exists, if not, new user will be created in model then inserted into database.  If it does, an alert box will appear.  *guestEH –* This button transfers the user straight to the levelOfStudyPane.  *mainScreenLoginEH –* This button transfers the user to the loginScreenPane.  **loginScreenPane EHs**  *defaultLoginEH –* Communicate with database to check if user exists, if it does, check that the hashed passwords match, if they do, allow the user to enter.  If the user doesn’t exist of passwords don’t match, an alert box will appear.  **levelOfStudyPane EHs**  *year3EH –* Sets the models studyYear as 3 & transfers user to choosingSubjectPane.  *year4EH –* Sets the models studyYear as 4 & transfers user to choosingSubjectPane.  *year5EH –* Sets the models studyYear as 5 & transfers user to choosingSubjectPane.  *year6EH –* Sets the models studyYear as 6 & transfers user to choosingSubjectPane.  *year7EH –* Sets the models studyYear as 7 & transfers user to choosingSubjectPane. |
| **choosingSubjectPane EHs**  *physicsEH –* Sets the models subject to physics and transfers the user to the choosingTopicPane.  *biologyEH –* Sets the models subject to biology and transfers the user to the choosingTopicPane.  *chemistryEH –* Sets the models subject to chemistry and transfers the user to the choosingTopicPane.  **choosingTopicPane EHs**  *topicChosenEH –* If the user double clicks on a topic in the listView, set the models topicName to the item selected and then transfer the user to the learningScreenPane.  This also triggers a database query to obtain the selected topics learning material text by using the models studyYear, subject and topic as the search variables. Then loading the text into the learningScreenPanes textArea.  **learningScreenPane EHs**  *nextPageEH –* This Button loads the next page of learning material stored in the mySQL database.  *takeQuizEH –* This Button transfers the user to the youngerQuizPane only after all the learning material pages have been accessed.  This also triggers a database query to Select the Question Text, Question Answer and Multiple Choice answers and load them into their respective textBoxes, labels and Buttons.  **youngerQuizPane EHs**  *answer1BtnEH & answer2BtnEH & answer3BtnEH*  When respective Button pressed, if that Buttons text == questionAnswer then increase totalMarks in the model by 1, if not, load in the next question. Rinse and repeat until all questions have been answered. When questionnaire is over, an alertbox will popup with their total score.  **olderQuizPane EHs**  *markQuestionnaireBtnEH –* When Button is pressed, compare all of the users answers against their respective stored questionAnswer’s, each correct answer increases the totalMark in the model by 1. An alert box will appear displaying their total score.  **Global EHs**  *goBackEH –* This event handler will take the user back to the previous page.  *backToTopicsBtnEH –* This event handler will be used in both quizPanes, it will guide the user from the quiz back into the choosingTopicScreen to select a new topic to study after completing the quiz for the previous topic. | |

## Use Case Description

These use case descriptions explain how users will perform the activities this program provides.

It clearly outlines starting from the first user action, what the user and the system both must do to complete the use cases function.

The exception conditions are reasons for a use case function failing, this is usually down but not limited to validation errors.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Login to Program – [Link](#_Login_to_Program) to test – [Link](#_LoginScreenPane_Coding_Progression) to implementation | |
| **Brief Description** | User clicks Login Button on Main Menu to bring up the form.  User inputs username and password and presses Login Button.  System checks username and password vs ones stored in the database.  If the details are correct the user will gain access to the program and their stored data. | |
| **Actors** | User | |
| **Pre-conditions** | Account must exist | |
| **Post conditions** | Allows user access to Program with their saved data. | |
| **Flow of activities** | Actor | System |
| 1. User presses Login Button on homepage. 2. User enters their Username and Password and presses the Login Button. | * 1. Login Form Appears   2.1 System verifies Username against Username list in database.  2.2 If username exists, UserID is saved and the password is verified against the password with the same userID.  2.3 Show user to ‘Selecting Level of Study” screen. |
| **Exception Conditions** | * 1. If username doesn’t exist return “Username Doesn’t Exist or is wrong”.   2. If Password doesn’t verify against stored password return “Password is wrong”. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Register an account – [Link](#_Register_an_account) to test – [Link](#_MainScreenPane_Coding_Progression) to implementation | |
| **Brief Description** | User inputs their chosen username and password into the appropriate boxes.  System checks if the username already exists.  If the username doesn’t exist, then a new account is made, and they’ll be automatically transferred to the ‘Selecting Level of Study’ screen. | |
| **Actors** | User | |
| **Pre-conditions** | Username must not exist. | |
| **Post conditions** | New user is created in the database. | |
| **Flow of activities** | Actor | System |
| 1. User inputs username and password into the text boxes. 2. The user presses the Register Button. | * 1. Validation of username and password   2.1 System checks against its already stored usernames.  2.2 System adds a new user to the database.  2.3 Transfer user to ‘selection of study’ screen. |
| **Exception Conditions** | * 1. If username or password contains characters other than Letters or Numbers return “Username and Password must only contain Letters or Numbers” before checking against Database.   If username is already contained within the database, return “Username is taken”. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Guest User – [Link](#_Guest_User) to test – [Link](#_MainScreenPane) to implementation | |
| **Brief Description** | If the user doesn’t wish to make an account or Login, they can choose to be a guest, this means all their scores won’t be scored and they can use the program anonymously. | |
| **Actors** | User | |
| **Pre-conditions** | N/A | |
| **Post conditions** | N/A | |
| **Flow of activities** | Actor | System |
| 1. User clicks on Guest Button | 1.1 System transferrers User to ‘selection of study’ screen. |
| **Exception Conditions** | None. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Selecting Level of Study – [Link](#_Selecting_Level_of) to test – [Link](#_LevelOfStudyPane_Image) to implementation | |
| **Brief Description** | Each button links to a different Level of study.  e.g. If the user selects the ‘Year 3’ button, the Year 3 Science Subjects show up.  This is the same for all levels of study. | |
| **Actors** | User | |
| **Pre-conditions** | User must be Logged In or entered as a guest. | |
| **Post conditions** | N/A | |
| **Flow of activities** | Actor | System |
| 1. User selects their level of study year | * 1. System transferrers User to ‘Different Subjects’ screen.   2. System saves study year into the model. |
| **Exception Conditions** | None. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Selecting Subject – [Link](#_Selecting_Subject) to test – [Link](#_ChoosingSubjectPane_Coding_Progress) to implementation | |
| **Brief Description** | Each button links to a different subject, these subjects have different topics in.  e.g.: If Year 3 is selected, then Physics, topics loaded in from the database will produce a list whereby only those that have those two fields in are shown. | |
| **Actors** | User | |
| **Pre-conditions** | User must be Logged In or entered as a guest.  User must have chosen a Level of Study. | |
| **Post conditions** | Topics are loaded in from database. | |
| **Flow of activities** | Actor | System |
| 1. User selects the subject they wish to study | * 1. System loads relevant Topics from the Subject into the program.   2. System transferrers User to ‘Select Topic’ screen. |
| **Exception Conditions** | None. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Selecting Topic – [Link](#_Selecting_Topic) to test – [Link](#_ChoosingTopicPane_Coding_Progressio) to implementation | |
| **Brief Description** | The user selects which topic they want to study from a list.  This then takes the user to the learning screen which contains all the information stored in the relevant topic Contents Database Column. | |
| **Actors** | User | |
| **Pre-conditions** | User must be Logged In or entered as a guest.  User must have chosen a Level of Study.  User must have selected a subject to study.  Program must have a working connection to the database | |
| **Post conditions** | Contents of topic are loaded in from database. | |
| **Flow of activities** | Actor | System |
| 1. User selects the topic they wish to study. | * 1. System transferrers User to ‘Learning screen’.   2. System Loads topic contents from relevant database column into the designated area in the Learning Screen. |
| **Exception Conditions** | * 1. If the database cannot load topic contents check database connection. If database connection is alive, reload data. If data cannot be loaded return “Sorry but this topic cannot be loaded at the minute, try again later or restart the program, sorry for your inconvenience”. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Taking Quiz – [Link](#_Taking_Quiz) to test – [Link](#_YoungerQuizPane_Image) to implementation | |
| **Brief Description** | Once the user has completed learning the topic content there will be a Quiz available to take.  This will be accessed through a quiz button, once pressed the questions will be loaded in from the database.  Available answers will be displayed below each question, the user will then select the answer they think is correct from the corresponding buttons/combo box below the question and continue. | |
| **Actors** | User | |
| **Pre-conditions** | User must be Logged In or entered as a guest.  User must have chosen a Level of Study.  User must have selected a subject to study.  Program must have a working connection to the database.  User must have selected a topic to study. | |
| **Post conditions** | N/A | |
| **Flow of activities** | Actor | System |
| 1. User clicks on the quiz button. 2. User selects an answer from the buttons/combo box below the question. | * 1. System changes to a quiz screen.   2. System loads questions from database into set text fields.   3. System loads multiple choice answers into Buttons or Combo Boxes. |
| **Exception Conditions** | 1.1 - 1.3 If database cannot load the Quiz check database connection. If database connection is alive, reload data. If data cannot be loaded return “Sorry but this quiz cannot be loaded at the minute, try again later or restart the program, sorry for your inconvenience”. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Marking Quiz – [Link](#_Marking_Quiz) to test – [Link](#_Coding_OlderQuizPane) to implementation | |
| **Brief Description** | Once they have completed the quiz they will click the mark button.  Clicking the Mark button will trigger a popup box to ensure they user has really finished the test.  Once they click ‘OK’ the program will check the users answers against the answers within the database and display a score. | |
| **Actors** | User | |
| **Pre-conditions** | User must be Logged In or entered as a guest.  User must have chosen a Level of Study.  User must have selected a subject to study.  Program must have a working connection to the database.  User must have selected a topic to study.  User must be on the quiz screen. | |
| **Post conditions** | Set completedTopic Boolean column in database to YES for relevant topic.  Update questionnaireScore INT column in database for relevant topic.  Set questionAttempted Boolean column in database to YES for each question answered. | |
| **Flow of activities** | Actor | System |
| 1. User clicks mark button. 2. User Clicks Close | * 1. System compares value inputted by user vs value stored in the answer column of the database for relevant questionID.   2. System updates completedTopic Boolean column to YES, questionnaireScore updated to equal the number of questions the user got correct, and questionAttempted Boolean column updated to YES for each question attempted.   3. System shows user their questionnaireScore.   2.1 System redirects user back to Topic Screen |
| **Exception Conditions** | 1.1 - 1.2 If database cannot load the Quiz answers check database connection. If database connection is alive, reload data. If data cannot be loaded return “Sorry but this quiz cannot be marked at the minute, try again later, sorry for your inconvenience”. | |

## Entity Relationship Diagram

An entity relationship diagram (ERD), is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. (Rouse, 2014).

The diagram [is shown here](#_ERD_Diagram) in appendix.

This ERD allows multiple different tables to communicate with each other.

The database will contain the following important data:

Users username, password, topics completed and total marks.

Subjects different topics, those topics year of study and contents.

Questionnaires related to the different topics are stored with a variable number of questions, the questions each having their own question number, question text, answer and a variable amount of multiple choice answers.

The ERD can been seen in text form below.

Key: primaryKey is underlined **foreignKey** is bold

User (userID, username, password, topicsCompleted, totalMarks)

PhysicsStudies (physicsStudiesID, **physicsID**, **userID**)

BiologyStudies (biologyStudiesID, **biologyID**, **userID**)

ChemistryStudies (chemistryStudiesID, **chemistryID**, **userID**)

Physics (physicsID, topicYear, topicName, topicContents, questionnaireScore, completedTopic)

Biology (biologyID, topicYear, topicName, topicContents, questionnaireScore, completedTopic)

Chemistry (chemistryID, topicYear, topicName, topicContents, questionnaireScore, completedTopic)

PhysicsQuestions (physicsQuestionsID, questionNumber, questionText, questionAnswer, questionAttempted, **physicsID**)

BiologyQuestions (biologyQuestionsID, questionNumber, questionText, questionAnswer, questionAttempted, **biologyID**)

ChemistryQuestions (chemistryQuestionsID, questionNumber, questionText, questionAnswer, questionAttempted, **chemistryID**)

*References*

Rouse, M. (2014). *What is entity relationship diagram (ERD)? - Definition from WhatIs.com*. [online] SearchDataManagement. Available at: https://searchdatamanagement.techtarget.com/definition/entity-relationship-diagram-ERD [Accessed 21 Apr. 2018].

# Development/Implementation

This development/implementation of the system section is very important as it describes how the program was made, any problems that were found and solutions thought of.

It’s also important because when integrating two styles of development methodology, waterfall and iterative, it shows where the actual program differs from the designs produced before development and why. As Java and MySQL are being learnt throughout the development process, the designs are sure to change when new knowledge is gained and acted upon, specifically the UML Design.

The development/implementation process for this project happened as follows:

* Coding Views, Models and a skeleton Controller in accordance with system design
* Navigational Functionality for each view is written
* Functionality from Objectives and Use cases are written
* If a view or a Model needs changing to correctly work with the functionality because of new knowledge learnt it will be changed and updated in the UML Diagram.
* Repeat this until all functionality from objectives and use cases have been added

All Finished View Images included in appendices are skinned with CSS, competent CSS was known prior to development therefore the implementation of CSS is very limited.

Use the navigational links freely throughout this section to better understand each individual component as there is a return link for every link.

Before development went underway I gained extra knowledge of existing programs/websites through extending my literature review, this was in response to my supervisor’s feedback of my Interim Hand-in. This is shown on [2nd](#_2) and [3rd](#_3) Progress Report.

After each view, model and significant function design had been developed the Eclipse Project was saved onto an external hard drive in line with the risk analysis solutions.

## MVC

MVC architecture was chosen because of the previous knowledge held from past projects that have been undertook at university and its widespread use in the professional word.

MVC or Model View Controller is an architectural pattern that separates a programs code into three different components.

‘Model objects are the parts of the application that implement the logic for the application's data domain.’

‘Views are the components that display the application's user interface (UI).’

‘Controllers are the components that handle user interaction, work with the model, and ultimately select a view to render that displays UI.’

‘In an MVC application, the view only displays information; the controller handles and responds to user input and interaction.’ (All above citations from Msdn.microsoft.com, 2018)

## Coding Views

**Main coding period of views between 10-23rd March**

[Link to pre-development UML Diagram](#_UML_Diagram_before)

[Link to post-development UML Diagram](#_UML_Diagram_After)

For each view there is a comparison between the UML pre-development and the UML post-development. This comparison shows the growth of the program during development and the new knowledge gained from continuously learning Java.

Because of the way MVC works a description of **why** these new variables were needed/old variables changed, will be included in the [Coding Controller](#_Coding_Controller) section as this is where most of the problem solving and all of the functional development is done, this ‘Coding Views’ section will describe **what** those changed functions/variables do, already described functions/variables from the [GUI & Related UML](#_GUI_(Graphical_User) section will not be repeated.

For all UML Diagrams in this section, BorderPane was extended for each class and GridPanes were used to shape the program. I decided to add GridPanes to a BorderPane and not use any other JavaFX GUI because of previous experience using these two and their respective ease of use.

All the views previously contained Strings or ints to display text e.g.:

String username

setUsername (String username) {

this.username = username}

have been changed to labels, TextFields… and the reason why will be discussed in the [development of the controller.](#_LevelOfStudyPane_Coding_Progression)

### Coding MainScreenPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/e745fe5f8794f11d1ac9f2df239899ab.png | https://i.gyazo.com/601bace7fab63f7e77a20b6bcb032f2f.png |
| newUsername changed to TextField to access the TextField itself instead of a saved String, this also allows for more commands.  newPassword changed to PasswordField, this allows text entered into Field to be shown as black dots, this aligns with my objectives.  passwordCheck added to be able to validate user input (check inputted passwords against eachother).  New functions added provide access to the new Private variables.  [Link to Coding MainScreenPane section of Controller](#_MainScreenPane)  [Finished View Image](#_MainScreenPane_Image) | |

### Coding LoginScreenPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/5c86b81acc303e31f5bd5a99cf34c74b.png | https://i.gyazo.com/1228401f694d93d4ed928ae66bbb9dbc.png |
| username changed to TextField to access the TextField itself instead of a saved String, this also allows for more commands.  Password changed to PasswordField, this allows text entered into Field to be shown as black dots, this aligns with my objectives.  Functions to directly get the text from the above TextField and PasswordFields created for efficiency of code.  [Link to Coding LoginScreenPane section of Controller](#_LoginScreenPane)  [Finished View Image](#_LoginScreenPane_Image) | |

### Coding LevelOfStudyPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/2b1410360c6372697a31f2b090f2c372.png | https://i.gyazo.com/377d9fce5b1d6556d673cd8ec78c6c8d.png |
| Removed year3Btn as no year 3 content provided anymore.  logoutBtn and its assosicated EventHandler added so user can logout from the LevelOfStudyPane.  addYear3EventHandler() removed as it wasn’t needed anymore.  [Link to Coding LevelOfStudyPane section of Controller](#_LevelOfStudyPane)  [Finished View Image](#_LevelOfStudyPane_Image) | |

### Coding ChoosingSubjectPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/56a2fd0d45dad308157bf55e6ee0582a.png | https://i.gyazo.com/1ffe50e1461fadb4b1a31020a0ba38e7.png |
| logoutBtn and its assosicated EventHandler added so user can logout from ChoosingSubjectPane.  This is a simple view pane so nothing else needed adding.  [Link to Coding ChoosingSubjectPane section of Controller](#_ChoosingSubjectPane)  [Finished View Image](#_ChoosingSubjectPane_Image) | |

### Coding ChoosingTopicPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/f23ad8b898b5a379b7fed097d2f694be.png |  |
| Combined the two visual notifiers of previous selection into 1 label.  Included a back button, logout button and the ListView to load in the topics associated with the subjectChosen and studyYear chosen.  Added respective EventHandlers.  Added getTopicListView, this will allow population of the ListView from the database.  [Link to Coding ChoosingTopicPane section of Controller](#_ChoosingTopicPane)  [Finished View Image](#_ChoosingTopicPane_Image) | |

### Coding LearningScreenPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/b09117b020b6e660f91f8319df158a7a.png | https://i.gyazo.com/3d9383fd318cdcf77d2974e1cc9a81be.png |
| Renamed all txtSubjectMaterial functions to topicMaterial and nextPage to nextContentPage whilst coding controller for ease of understanding reasons.  logoutBtn and respective EventHandler added, consistent navigation is key.  An ImageView was added to the LearningScreenPane due to the positive effect it has on remembering topic material and increased enjoyment of learning, this was learned from the literature that was reviewed.  getTakeQuizButton added so that it can be accessed by the controller to control when the user is able to take the quiz, e.g. only after all pages of the topic have been read.  goBack button and EventHandler added so the user can easily chose another topic to learn if they accidentally misclicked on a topic or have gotten bored.  [Link to Coding LearningScreenPane section of Controller](#_LearningScreenPane)  [Finished View Image](#_LearningScreenPane_Image) | |

### Coding YoungerQuizPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/419740d6d94bf5b4197e3edb8fdea014.png | https://i.gyazo.com/79406cd55f9151a7f1d213a5dfe70cf8.png |
| questionNumber moved to question Model.  questionText changed from a TextArea to a Label because the visual effects of a TextArea are detrimental to the look of the program. Whereas when you use a Label its pure text with no area/border surrounding it. This allows for a cleaner look within the program.  Having getAnswer1Btn, 2 and 3, allows the program to access the buttons to disable them after the quiz has finished.  [Link to Coding YoungerQuizPane section of Controller](#_YoungerQuizPane)  [Finished View Image](#_YoungerQuizPane_Image) | |

### Coding OlderQuizPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/6d253b775115e3ad850b5b313fb92a33.png | https://i.gyazo.com/9b371f4e5f2e3bedb8dc9f513b972b1b.png |
| OverallMark to Label, this displays the overallMark attained by the user.  questionText has been changed to a Label, same reason as in YoungerQuizPane.  A single markQuestionnaireBtn is displayed at the end of the page when the user has completed the test.  The test has been extended to 5 multiple choice questions, each of them is displayed via a questionLabel and a answerComboBox, with a Image located above if contained in the database.  The populateComboBoxes function has been removed as all comboBoxes now accessible by the controller via their respective get methods. It was decided to include a backToTopicsBtn so that after the user has taken the quiz, if they wish to, they can choose another topic to learn about.  [Link to Coding OlderQuizPane section of Controller](#_OlderQuizPane)  [Finished View Image](#_OlderQuizPane_Image) | |

### Coding ScienceForKidsRootPane

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/9ed264b2d9758f3f0ae2eebd59e97d76.png | https://i.gyazo.com/aca1fd755d7dc0555c7f50183dae4253.png |
| Only thing needing to be changed during iterations of development was to add the setPane() function that allows the controller to change what view the program shows.  [Link to Coding ScienceForKidsRootPane section of Controller](#_ScienceForKidsRootPane) | |

## Coding Models

[Link to pre-development UML Diagram](#_UML_Diagram_before)

[Link to post-development UML Diagram](#_UML_Diagram_After)

The Model package of the program contains different models, these models all store volatile data and are vital when switching panes because they allow the controller to load in the stored model data.

### Coding Question Model

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/961ef63b3a7300f60c927787c3677fd0.png | https://i.gyazo.com/db5fa22e0c0a3e84d1ad9eb5a1c5a67b.png |
| Renamed Class to Question due to irellivency to the name and the new Questionnaire class that needed its own model.  questionImg added due to extended literature Review analysis of other learning programs/websites containing images with content and questions.  Relevant get and set methods for Image added. | |

### Coding User Model

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/e8873e87bc77cf6758baddd5e8f97d43.png | https://i.gyazo.com/c9b05ca9e720d4413c780e4beaa2ecc1.png |
| After working with the MySQL database for some time it made sense to store all columns from the User table into a model, this is why userId, TotalMarks and topicsCompleted have been added to the User model.  Their relevant get and set methods have been added to. | |

### Coding Questionnaire Model

|  |
| --- |
| Before |
| https://i.gyazo.com/48bf890cd1241b3040528092f55f4c24.png |
| The reasoning for this addition to the model package will be discussed more in the controllers [YoungerQuizPane](#_YoungerQuizPane_Coding_Progression) & [OlderQuizPane](#_OlderQuizPane_Coding_Progression) sections.  Questionnaire is a sub-class of Question, it is the format that allows questions to be added to a list via the addQuestionFunction. Whilst many different questions are stored in the questionnaire, you can access it through the getQuestionByQuestionNumber function.  When the program needs to load in a new questionnaire it can be cleared with the resetQuestionnaire function. |

### Coding Topic Model

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/656c28380c9de06a88dbf9061abf0e7d.png | https://i.gyazo.com/d3358ca1f3bf846569d2129cac76e18b.png |
| topicId was a value that needed to be stored to access the correct values in the database.  splitTopicContents is a String[] variable, it allows for multiple split strings ‘Different parts of the topic contents’ to be saved in the model.  topicsSplitNumber was added to define the number of different parts of the topic that need to be displayed, it is a count for the amount of times the next page button is needed.  topicSplitDisplayCount was added to be able to save the amount of times the next page button has been pushed so we can compare it against the topicSplitNumber to see how many more times a new page needs to be loaded in. | |

### Coding ScienceForKidsProfile Model

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/a0141eac17d1a0e2ced7f36838525cee.png | https://i.gyazo.com/05b2fe1b8e1432b1e89f902c09a329e2.png |
| Total Marks is now being used to count the amount of questions the user has gotten correct in the Younger and Older quiz pane. This is more important in the Younger Quiz because the total mark needs to be updated after every answer, whereas in the Older Quiz it just needs to be calculated when the markQuestionnaire button is pressed.  Removed numberOfTopicsCompleted because it was decided to be an irellevant statistic in further development, it proved to be more hassle than it was worth.  resetTotalMark is to be called whenever the user leaves the Quiz Panes.  SubjectMaterial renamed to Topic for ease of understanding.  Added Questionnaire Model.  Relevant set and get methods added. | |

## Coding Controller

[Link to pre-development UML Diagram](#_UML_Diagram_before)

[Link to post-development UML Diagram](#_UML_Diagram_After)

Using MVC’s architectural pattern this section contains the controller, as its name alludes to the controller contains all the functional programming, for each Event Handler in the views an action event is coded and attached to it. This is the brain of the program.

All new function additions will not be explained in the table but in their own relevant paragraphs within the Coding Progression Sections.

All Controller functions

|  |  |
| --- | --- |
| Before | After |
| https://i.gyazo.com/bf8a92f6d011b1ba29b1e964e5710646.png | https://i.gyazo.com/b84ced769f89448b9ffe1b93cb019ac7.png |

### MainScreenPane Coding Progression

[View Coding changes](#_Coding_MainScreenPane,_main)

*Developing Registering an account Use Case*

The code for createUserEventHandler can be found [here](#_createUserEventHandler_code)

It may look daunting for non-programming people but there’s an explanation of what’s happening within the code above it.

This was the first MySQL piece of code written and a couple problems arose because of this inexperience.

**Problem**:

Couldn’t check stored variable of username against usernames in database for an unknown reason.

This is needed to check if user already exists which is the pre-condition in the Use Case Description.

**Solution**:

Used <https://stackoverflow.com/questions/11292468/how-to-check-if-value-exists-in-a-mysql-database> For help, tried this way instead of looping through all the username values and checking each one, it works because the SELECT function in SQL already loops through the entire data set so no loop programmed in Java is needed.

Then all that’s needed is to check the results set, if the length of ‘myRs’ is < 1 then the Username cannot exist, therefore the program allows this username to be registered.

**Problem**: Displaying the password as \*\*\*\*\* instead of regular alphabet letters.

**Solution**: <https://docs.oracle.com/javafx/2/ui_controls/password-field.htm>

Implement class PasswordField. This JavaFx UI Control allows for this important visual feature to be used, instead of a normal TextField.

**Problem**

How to store a secure password, after some more research the conclusion that normal hashing algorithms should not be used for security reasons was reached.

To meet the objective ‘To include a hashing algorithm to store passwords in database for increased security of account.’ In the [Project Contract](#_Project_Contract) a new method would need to be implemented.

**Solution**

I researched an independent hashing algorithm and implemented in into my program.

Implementing <http://www.mindrot.org/projects/jBCrypt/> JBCrypt password hashing algorithm.

*Developing the Guest User Use Case.*

Problems: Not knowing how to switch windows.

This may seem like a basic, first thing you learn function, but the previous projects undertaken all used tabs as a form of navigation and not transitioning between windows.

Solution:

**public** **void** setPane(BorderPane pane) {

**this**.setCenter(pane);

}

This function works because all Views are sub-classes of the BorderPane class as they all extend BorderPane, so they’re all accessible by this function. After implementing this function all navigation problems were solved as to switch between screens all that’s needed is.

view.setPane(**insert** **view here**);

### LoginScreenPane Coding Progression

[View Coding changes](#_Coding_LoginScreenPane,_main)

Use case description for Login

*Developing Login Use Case*

After developing the Registering an Account feature the login was easy and smooth. The code for it can be found [here](#_loginEventHandlerCode), it follows the Use Case’s flow of activities.

### LevelOfStudyPane Coding Progression

[View Coding changes](#_Coding_LevelOfStudyPane,_main)

The Selecting Level Of Study Use Case code was relatively simple, each button was assigned a year and once pressed the year’s int would be stored in the model.

The Year 3 section of the program was removed because meetings earlier on in the project with my supervisor made me aware that it might not be needed. Having year 4 as the basic knowledge year turned out better as there’s not much of a difference in the learning capabilities and the topic content between a year 3 & 4 student.

One problem that spanned across the entire program was loading previously saved text from the Model into the Labels in the next View. E.g.: Username entered being displayed in the heading label.

This was down to inexperience with Java in the design section of this project.

The solution to the problem above was changing the Private variable from a String/int to the Label itself, then defining the Label in the main body without reinitialising it and supplying a get method below. This get method allows the controller direct access to the Label and usage of the Label.SetText(); method.

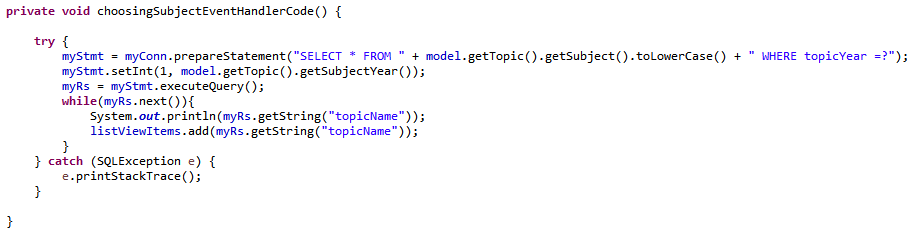
### ChoosingSubjectPane Coding Progression

[View Coding changes](#_Coding_ChoosingSubjectPane,_main)

The Selecting Subject Use Case Code is below, it shows the systems flow of activities.

System loads relevant Topics from the Subject into the program.

System transferrers User to ‘Select Topic’ screen.



Nothing of note happened within this implementation section, the above code shows how the program selects all the relevant topics by their subjectYear & subject and then loads them all into the list view in the ChoosingTopicPane.

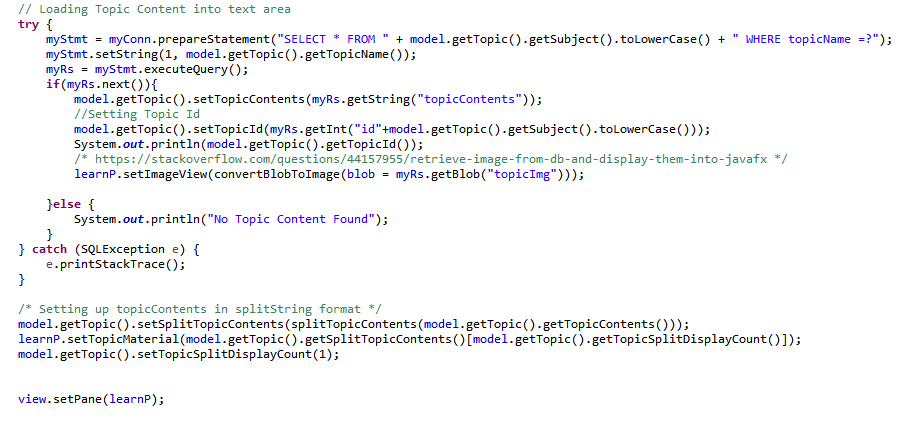
### ChoosingTopicPane Coding Progression

[View Coding changes](#_Coding_ChoosingTopicPane,_main)

The Choosing Topic Use Case code can be seen below, it shows the code aligning with the systems flow of activities.

System transferrers User to ‘Learning screen’.

System Loads topic contents from relevant database column into the designated area in the Learning Screen.



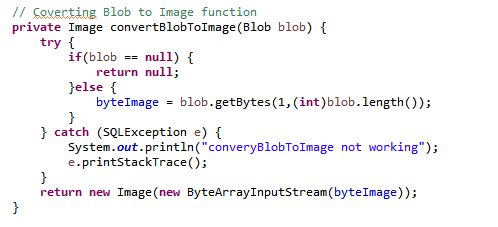
It also shows the split String usage which will be discussed in the LearningScreenPane section.

The link

<https://stackoverflow.com/questions/44157955/retrieve-image-from-db-and-display-them-into-javafx>

was used to gain knowledge on how to insert Images stored in a database into the program, from the research done in the extended literature review, having images within the program is consistent among professional programs.

A reusable generic function (shown below) was then developed after this so that only 1 bit of code was needed to convert Blobs in a database into Images.



### LearningScreenPane Coding Progression

[View Coding changes](#_Coding_LearningScreenPane,_main)

An interesting problem occurred during this iteration of development, how to contain all the topic content in one field in the database whilst having multiple different pieces of the topic in the program and then loading them into the content viewer when the nextContentBtn button is pushed.

The solution was to create 2 new variables in Topic model to contain the number of different pages needed and the split String[] splitTopicContents, that allows us to set the number of pages needed in the model and pre-split them before loading.

Also including a topicSplitDisplayCount so we can access the number of the displayed topicContent String that has been split to correctly load in the next one.

This solution has also helped me find a way to also load in a variable amount of answers into the combo boxes for the olderQuizPane.

The TakeQuizBtn also proved to be a long and extensive task.

The first problem was how to load in relevant topic questions from database as they’re stored in different fields in the database.

The solution was to create a new variable ‘topicId’ in the class ‘Topic’ in the package ‘model’, with Get and Set methods, this allows me to store the ID of the topic and SELECT the relevant questions in database to load them into programs questionnaire model.

This posed another problem, how to store all questions loaded in model?

This led to the development of a new Model called Questionnaire which is a sub-class of Question.

The Questionnaire Model allows for any number of questions from the databased to be saved in the model all at once and loaded into the programs view at runtime.

The code to load the YoungerQuizPane can be found [here](#_LearningScreenPane_TakeQuizBtn_Youn).

The code to load the OlderQuizPane can be found [here](#_LearningScreenPane_TakeQuizBtn_Olde).

### YoungerQuizPane Coding Progression

[View Coding changes](#_Coding_YoungerQuizPane,_main)

The code for the YoungerQuizPane marking function can be found [here](#_YoungQuizScreen_Marking_Question).

Allowing for a more interactive experience the idea to change the image for every different question in younger quiz pane arose.

Therefore, a new Variable in Question Model ‘Private Image questionImg’ was made, it loads a questionImg blob from database and replaces the current Image in the ImageView in younger quiz pane every time an answer is given.

In the Use Case Systems Flow of activities 2.1 It states the ‘system redirects the user back to the Topic Screen’, in the real program the quiz’s buttons/Combo boxes are greyed out using

yqP.getAnswer1Btn().setDisable(**true**);

oqP.getCombobox2().setDisable(**true**);

The only button that can be selected on the view is the ‘Back to Topics’ Button.

### OlderQuizPane Coding Progression

[View Coding changes](#_Coding_OlderQuizPane,_main)

The code for the OlderQuizPane marking function can be found [here](#_OlderQuizScreen_Marking_Questionnai).

The only problem that occurred in this iteration of coding wasdisplaying multiple images for the different questions, solved by creating 4 new Image views in my olderQuizPane and adding them to the GridPane.

Also added if NULL check in convertImageToBlob function displayed in the previous page to counter NullPointerExceptions.

### Skinning Program with CSS

The final part of the development project was to ‘skin’ the program with CSS.

This was quick and easy as a previous web design project had been completed so no further knowledge was needed.

Final version of the programs GUI can be seen in the [Instructions with screenshots](#_Instructions_w/_screenshots) appendix.

*References*

Msdn.microsoft.com. (2018). *ASP.NET MVC Overview*. [online] Available at: https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx [Accessed 16 Apr. 2018].

# Testing

Testing may be the most important part of a software development lifecycle, it proves whether the program has succeeded or failed, often testing is the reason why the program also won’t fail in the future.

This section contains

* The testing of use case description functionality.
* Testing the project against objectives in project contract.
* Testing the project against non-functional requirements.

## Testing Against Use Case Descriptions

As specified in the objectives of the project contract JUnit testing will be used to test if the core use case descriptions are working. JUnit is a unit testing framework for Java.

The tests will be in the Projects Workspace, the information in the proceeding tests will contain:

* What the test is testing.
* What the expected result is.
* What the actual result is.

Each test will have sample data that is expected to pass and sample data that is expected to fail.

The final screenshot of JUnit test results can be seen [here](#_JUnit_Final_Test) in the appendix.

*Keywords:*

*assertEquals* is a JUnit function that compares two values and returns true IF both values are the same.

*assertTrue* is a JUnit function that takes a query and returns true if the query returns true.

### Login to Program

This test saves username and password input into the model then compares them against the values in the database in the same way the Login feature does. Then it creates a new user in the model and checks the loaded in values from the database against the expected result.

The test opens a connection to the MySQL database, and selects all the values from the table user for each part.

The first part of the test inputs ‘fakeUsername’ into the model and checks it against the database.

*assertEquals*("The results set should return false", **false**, myRs.next());

If there are no results, myRs.next() = false, then the test passes.

This part **PASSED** as myRs.next() equalled false.

The second part of the test inputs ‘realUsername’ and ‘realPassword’ into the model and checks it against the database.

*assertEquals*("The Username should return realUsername", "realUsername", myRs.getString("username"));

If the results set from the column username has a value, decrypt the associated password and compare it against the one stored in the model.

assertTrue(BCrypt.checkpw(model.getUser().getPassword(), myRs.getString("password")));

This part **PASSED** as the decrypted password when compared with the stored password equalled true.

The third part loads in all the Logged in Users data into the model and checks expected values.

assertEquals("Id field should be the same as the id for the user in the database", 44, model.getUser().getUserId());

assertEquals("Username field should contain", "realUsername", model.getUser().getUsername());

assertEquals("TotalMarks field should contain 0", 0, model.getUser().getTotalMarks());

assertEquals("topicsCompleted field should contain an empty string", "", model.getUser().getTopicsCompleted());

This part **PASSED** as all the expected data equalled the real live data.

### Register an account

This test first checks if the passwords entered are the same, if they aren’t then assertEquals will return false. After this the test then queries the database to see if a known existing username exists, then it tests for a username that is known to not exist in the system and the hashing algorithm hashing an inputted password then un-hashing it.

A new user row is then INSERTED into the database with the previously entered Username and Password. The database is then queried about the newly entered username and password.

All parts of this test **PASSED,** and a new row can be seen in the MySQL database as shown below.

https://i.gyazo.com/979937e5b845ed7708d1bf13a011dfb8.png

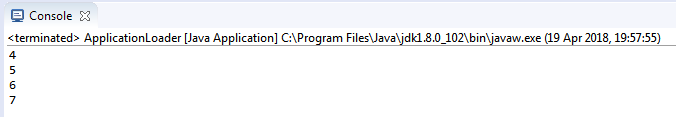
### Guest User

Because of the nature of this function it cannot be tested as it purely switches views as seen in its use case with only 1 flow of activity.

### Selecting Level of Study

Because of the nature of this function it cannot be tested with JUnit as it purely switches views and saves an int in the model as seen in its use case with only 1 flow of activity.

Therefore, a simple System.out.println(); function will be used every time a Study Level is selected to print the study year from the model into the console.



Using the ‘Back’ Button in the ChoosingSubjectPane the models study year is shown to change to its corresponding button from each click. This test **PASSED**.

### Selecting Subject

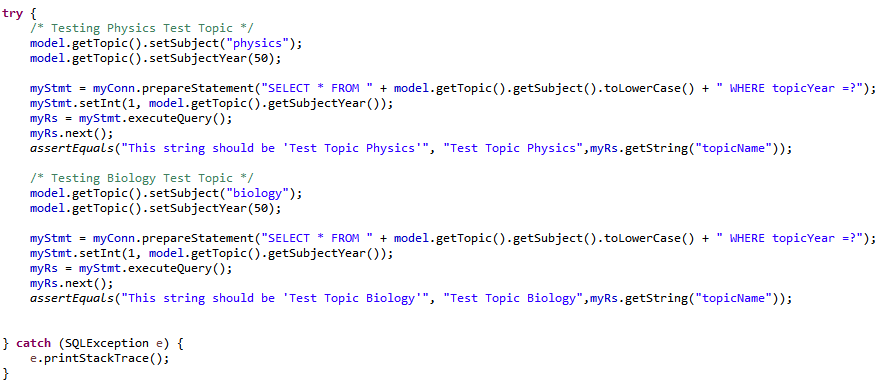
After the user selects a subject, the related topics must be loaded into the database.

This test will try to find two known topics in the database through selection of the topics specific studyYear and Subject.

The two rows can be seen below, one in the biology table and the other in the physics table.

https://i.gyazo.com/6fbf806c04d4b95d62675dd55f022667.png

https://i.gyazo.com/8563ac7df079b5b97c6f4736469ca93c.png



Both assertEquals functions **PASSED.**

### Selecting Topic

After the user has selected the topic the related topic content has to be loaded into the program.

This test will try to find the known topic content for both of the above database rows.

Not that the Physics Test Topic Content’**s’** contains an S and the Biology doesn’t, this is to further make sure different tables are being accessed and checked.

https://i.gyazo.com/1f0e0df80041e532fb37caffbd0e4414.pnghttps://i.gyazo.com/1cbe8c0e0dfce94d321473e538301d81.png

This test **PASSED**.

### Taking Quiz

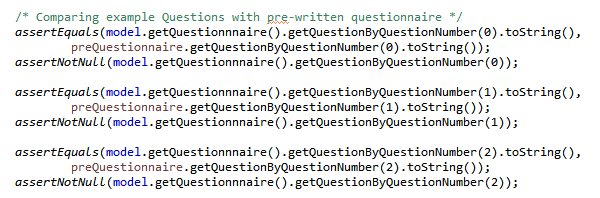
Even though there are two different Quiz Panes only one test will be run for this section, this is because we are testing values selected from the database against a pre-written questionnaire.

The only difference between the two Quiz Panes is the GUI and variable number of multipleChoiceAnswers.

This test will take an example quiz from the biology table that is related to the previous ‘Test Topic Biology’ in the Selecting Subject Test, this test will only contain 3 questions instead of the usual 8 each containing an image.

The test will compare the data contained in the questionText, questionAnswer, multipleChoiceAnswers and questionImg columns from the biologyquestions table in the database.

The test will pass if all the pre-written sample questions match up with those selected from the database AND if the questionImg field returns Not Null.



The model Questionnaire is the one loaded in from the database.

The prequestionnaire it the one pre-written to match the database fields.

All 3 assertEquals and assertNotNull tests **PASSED**.

### Marking Quiz

Even though there are two different Quiz Panes only one test will be run for this section, this is because we are testing values selected from the database against pre-written questionnaire answers.

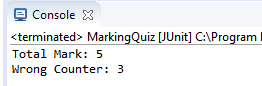
The only difference between the two Quiz Panes is the GUI and variable number of multipleChoiceAnswers.

This test will take the same example quiz as above and increment a total marks counter for each pre-written correct answer. The total marks should end up as 5 as 5 pre-written answers will be correct and 3 will be wrong, the wrong answers will have a counter of their own that should total 3.

*assertEquals*(model.getTotalMark(), 5);

*assertEquals*(wrongCount, 3);

This test **PASSED** as both assertEquals functions passed, the values of the variables can be seen below.



## Testing Project Against Objectives

These objectives are specified in the [Project Contract](#_Project_Contract) each objective must be met, if not, a valid reason must be given as to why this objective was not met, a successful project should meet over 90% of its initial objectives.

|  |  |
| --- | --- |
| To investigate science project software requirements and produce a Requirements Specification. | Checkmark |
| Early Requirements specification can be seen [here](#_Early_Requirements_Specification) | |
| In-depth fully referenced literature review on students aged 7-12 learning style, whilst also touching on the statistics of girls studying STEM subjects. | Checkmark |
| Expansive Literature review of 2600 words can be seen [here](#_Literature_review) in appendicies | |
| To design my programs learning features around the findings from my literature review. | Checkmark |
| Images, multiple choice questions, tests short and often are some of the many things included in my program from the literature review findings. | |
| Development methodology for the software life cycle will be decided and implemented. | Checkmark |
| Waterfall and Iterative development decided upon and implemented. | |
| Design the GUI of the desktop application for functions such as Login Screen and quizzing screen. | Checkmark |
| Designs for GUI can be found [here](#_GUI_(Graphical_User) in the Program Design Chapter | |
| Design database using Entity Relationship diagram. | Checkmark |
| The Entity Relationship Diagram designed on the MySQL Workbench was forward engineered into the working MySQL Database. | |
| Create a database that stores users’ data and score tables. | Checkmark |
| The users Username, Password, Total Marks and Topics Completed is stored. | |
| To include a hashing algorithm to store passwords in database for increased security of account. | Checkmark |
| Hashing algorithm implementation explained [here](#_MainScreenPane) | |
| Allow users to learn subject material, be tested and receive feedback. | Checkmark |
| Feedback limited to score | |
| Use JUnit and other professional methods of testing for each objective to make sure they work exactly as proposed. | Checkmark |
| JUnit tests where coded using example data, they all passed. | |
| Include error handling within code to increase robustness & reliability of program. | Checkmark |
| Try catch blocks implemented throughout Controller code to catch any exceptions | |
| Create test plan for the software for verification and validation checks specified in the Requirement Specification documents |  |
| All validation & verfication checks where implemented into the system but the test plan descriptions only tested a few of them. | |
| Final report will contain critical evaluation of each stage of the entire project. | Checkmark |
| Critical evaulation completed. | |

## Testing Against Non-Functional Requirements

|  |  |
| --- | --- |
| There will be sufficient Documentation for first time users to understand how to use the program properly. | Checkmark |
| Link to Instructions w/ screenshots tutorial in appendix [here](#_Instructions_w/_screenshots). | |
| There will be sufficient Documentation of system design for future programmers to understand how each function should work in case of future faults. | Checkmark |
| Both the [Design section](#_Program_Design) and the [Development/Implementation](#_Development/Implementation) of this report provide enough design documentation for the use of this program to be understanded and faults fixed. | |
| Disaster Recovery: 2 Backups shall be made once a week, one in solid form on an external drive and online. | Checkmark |
| During development mulitple backups were made onto a USB stick and onto dropbox. | |
| Deployment: This program shall be made downloadable via a CD. | Checkmark |
| CD picked up from FOTAC has the executable file on it. | |
| Users attempting to login more than 4 times in a minute shall have an error message appear and the program will close. | Close |
| The reason this objective was not met is because a decision was made mid-development to not include it, only young children will be using this program therefore this type of security check is superfluous. | |
| No personal data about the user shall be stored in the databases. | Checkmark |
| Only data stored pertaining to the user is a made up username and password | |
| This program shall be able to run by users working with low spec computers. | Checkmark |
| The files needed to run the program are very small in size and it has been tested on low spec laptops. | |
| None of the programs code will be accessible by users. | Checkmark |
| Only file contained on the CD given to users will be the executable file and a pdf containing instructions on how to use ScienceForKids. | |

# Critical Analysis and Conclusion

*The overall project.*

The program achieves the projects main goal, to produce an application that allows kids to learn science. The programs features are not extensive but those that are implemented are solid, contain necessary validation and the programs learning features are ‘future-proofed’ for any number of new topics that wish to be added to the program.

Graphical designs are generally one of my shortcomings, previous projects have good working functionality but still look novice-like, the key reason why this project hasn’t followed along with the previous failings is because of the research done in the [literature review](#_UML_Diagram) on already existing professional applications, the simple navigation for kids, colour schemes and much more.

This has helped me gain an eye for at least having good simplistic designs and it’s shown itself with this project.

I am particularly proud of my transition from design to implementation because of the software development methodologies that were mixed, creating the basic outline of the program from scratch, correctly using the Model View controller architecture pattern, then adapting the existing designs to overcome and problems within the many iterations of development.

One issue, because of the way I was using split string, was obtaining a variable amount of multiple choice answers from the database without an error occurring, I knew I needed to loop the entire String and then add each new answer to the combo box but couldn’t figure out how.

Using an incremental local variable, I was able to define how many different answers were in the database and use this variable as the split ‘limit’ for the function.

*Taking this project further.*

Given more time on this project there are some functionality additions I would like to make.

The first of which is a personal profile, imagine a button just after the login screen that shows a personal profile. It contains a list of topics the user has completed, the overall score received for each topic and other additions that are unthought of yet. This would allow kids using the application to check their progress and maybe revisit earlier topics to re-test themselves and earn a higher score.

The second one is a program that helps the developer (me) to quickly add new topic content and questionnaires to the database through a GUI, instead of having to edit saved SQL scripts.

This would greatly decrease the wasted time having to edit an SQL script and increases the enjoyment the developer has whilst creating new content and how efficient the process becomes, as overall its sure to save time in the future.

The third addition was a feature initially thought about but that didn’t make the requirements specification or early design stage, this was an opt-in leaderboard, the leaderboard would contain your username, number of topics completed and total marks.

This leaderboard would create competition among the users and any active users would be rewarded as frequent updates will include new topic content and if they can learn and acquire a top score they can top the leaderboards for an amount of time.

To stop users from just finishing all the topics without gaining a good mark to increase their leaderboard position there could be an overall percentage mark of all topics completed by one user on the leaderboard, instead of just ranking by the number of marks gained and or the number of topics completed. A workaround could also be that if a user scores below 60% in a quiz, that topic is not deemed as completed.

The final addition to the program in the foreseeable future would be creating more questions for each topic and having them randomized so that when taking the quiz, not everyone would have the same questions in the same format and that when re-taking the quiz, new questions appear so that users cannot easily brute force a high score.

*Project Management & development methodology*

When allocating the amount of time needed to do certain tasks of the project I think I did a good job for everything except the amount of time needed to code the controller, I could’ve done with more time.

In the Gantt chart [here](#_Gantt_Chart) you can see the time I allocated for view is over half of the time I allocated for controller and this was much more time than I needed, in the future I would probably allocate around 20% of my time to the view, 10% to the models and the remaining 70% to coding the controller. I also found allocating the time when I couldn’t work on my project was very helpful as this allowed for a more realistic development timeline.

When deciding what to develop it really helps to put pen to paper and jot down many different ideas until you come up with something reasonable, this is why I chose to implement the waterfall approach to this project, because it’s easier to develop a program when you already know the end goal and have planned the journey to it.

I already knew how the MVC architecture worked from previous smaller projects and I felt comfortable to design a program with this pattern.

After the basic design of the system was completed and coded it allowed me to develop my ideas further, implement more advanced features and then go back to my original UML diagrams to alter them to suit the needs of my new ideas or solutions to various problems.

This was a very important part of the development, as using an iterative development methodology was necessary to adapt to the new programming knowledge learnt whilst finding these solutions.

This iterative development was also imperative to the success of my project because the first literature review carried out was weak and did not research existing applications, as can be seen in this [progress report](#_2), therefore this development methodology was necessary to apply the knowledge gained from the extended literature review.

Therefore, I believe the approach taken to this project was the correct one and will be used in the future for more personal projects.

*Personal gains & growth*

During the development of this project I learnt just how complex even a basic application could become, it made me grow an appreciation for programmers that deal with programs with 1000x more content than this one.

I also grew more competent in being able to translate the solutions I thought of into proper working code, I believe this one ability will be key for the rest of my programming life.

The academic objectives that were touched upon in the introduction were recognised, Java software development is at the top of my list and this project surely helped me gain experience coding in Java.

I was also looking to add projects to my portfolio and this is definitely one that will be at the top.

If I did this project again with the knowledge I have gained I would implement a couple more ‘quality of life’ features, nothing too ground breaking, but something like the profile that was described above would be good, I was also keep better tabs on my design decisions as to be better able to explain and defend them in this report.

*Tools*

Using Java was a personal choice of mine and I felt Java and JavaFX (The GUI library) didn’t limit me in my abilities to produce the designs and my ideas.

In particular the JavaFX library was very easy-to-use, had a very clean finish and when compared to other Java libraries like swing, is cleaner and easier to maintain.

It also comes with tonnes of supporting documentation such as the extensive API and online tutorials that helped enormously and when I do future Java programs JavaFX will be used by default.

Using MySQL over SQLite was a hard decision to make because of the nature of this project but deciding on MySQL is something I feel like prepares the program for implementation into a real-world scenario, this is because the database can instantly be put online, whereas SQLite is a local database file system and therefore would need migrating before the program could access it online.

*Conclusion*

The ScienceForKids application has met a personal goal, it has become an application that can be used by kids of all ages to learn about science and it can become an even greater learning tool through further development.

With all of the objectives met this project has been a success and has helped develop certain skills such as time allocation within a projects management, transitioning ideas from conception into a working application. This project has boosted my confidence and has me wanting to challenge myself further in my future pieces of work.

Total Word Count: 10940

# Appendix

## Project Contract

**Student Name: Edward Tallentire**

**P-number: P15230019**

**Programme: G40051 Computer Science**

**Email: P15230019@my365.dmu.ac.uk**

**Project Title: Science for Kids**

**Project Proposer: Self proposed**

**Supervisor**

Tarjana Yagnik, School of Science and Informatics, [tarjana.yagnik@dmu.ac.uk](mailto:tarjana.yagnik@dmu.ac.uk)

**Introduction**

Science for kids is a desktop software that will help kids between the ages of 7-12 learn and develop an interest in science.

The software will be written in Java and will have a MySQL Database to store profiles for individuals that wish to record their progress and test scores in, these individual profiles will be accessible through a login screen.

This desktop application will be used by teachers and parents to better engage their kids in science, using modern technologies as an aid for regular teaching.

This application will aid kids learning of science but will not be a replacement for real life school teaching.

**Project Background**

This project is a personal interest of mine, whilst at school I did not like my science lessons as they were very tedious, and I could not concentrate leading to an average grade and being unable to pursue the subject further. Outside of school I found various websites and YouTube channels dedicated to making science interesting, so when I had my own learning autonomy I developed an interest in physics, astrophysics to be exact.

I also have a little sister who intensely dislikes the teaching of science in her all girls secondary school, but when I talk to her about various science-based subjects is very engaged and seems to show an interest.

This is why I want to make a program that can help young people through the boring teachings of science and maybe stimulate them to do further research and learning so that all doors remain open to them throughout their student life.

**Aim/Objectives/Deliverables**

**Aims**:

Develop software named ‘Science for Kids’ in Java that will encourage kids to learn science, participate in end of unit quizzes, and receive feedback on their answers.

Have individual profiles, science material and test papers stored in and retrieved from the database.

**Objectives**:

* To investigate science project software requirements and produce a Requirements Specification.
* In-depth fully referenced literature review on students aged 7-12 learning style, whilst also touching on the statistics of girls studying STEM subjects.
* To design my programs learning features around the findings from my literature review.
* Development methodology for the software life cycle will be decided and implemented.
* Design the GUI of the desktop application for functions such as Login Screen and quizzing screen.
* Design database using Entity Relationship diagram
* Create a database that stores users’ data and score tables
* To include a hashing algorithm to store passwords in database for increased security of account.
* Allow users to learn subject material, be tested and receive feedback.
* Use JUnit and other professional methods of testing for each objective to make sure they work exactly as proposed.
* Include error handling within code to increase robustness & reliability of program.
* Create test plan for the software for verification and validation checks specified in the Requirement Specification documents
* Final report will contain critical evaluation of each stage of the entire project.

**Deliverables**

First Submission

* Project Contract
* Ethical Review
* BCS Checklist
* Gantt chart

Interim Submission

* Literature review
* Requirement Specification

Final Submission

* Design documents
* Design ER diagram
* Coding the software
* Testing the software
* Writing a final report
* Appendices

**Resources**

Hardware: O/S Windows 7, RAM: 8GB, Processor: Intel i7-4790k, 4GHz, 2: 60Hz BenQ Monitors

Software: JAVA, MYSQL

Network: Internet

**Constraints**

* Questions within the quiz be will multiple choice, as open-ended answers will be hard to mark within the developed software and kids tend to misspell words.
* Time: 29 Weeks to develop, fully test and document a program whilst also being a member of a DMU Sports team and completing other module projects may reduce some of the complexity of functions that would be included otherwise.

**Sources of Information**

* Library access,
* Internet access
* Horstmann, C. (2014). *Big Java Early Objects*. 5th ed. John Wiley and Sons, Inc.
* Docs.oracle.com. (2017). *Java Platform SE 7*. [online] Available at: https://docs.oracle.com/javase/7/docs/api/ [Accessed 23 Oct. 2017].

**Risk Analysis**

My computer breaking down:

As a part of project process, weekly backup will be scheduled one with university H drive and another with my external hard drive.

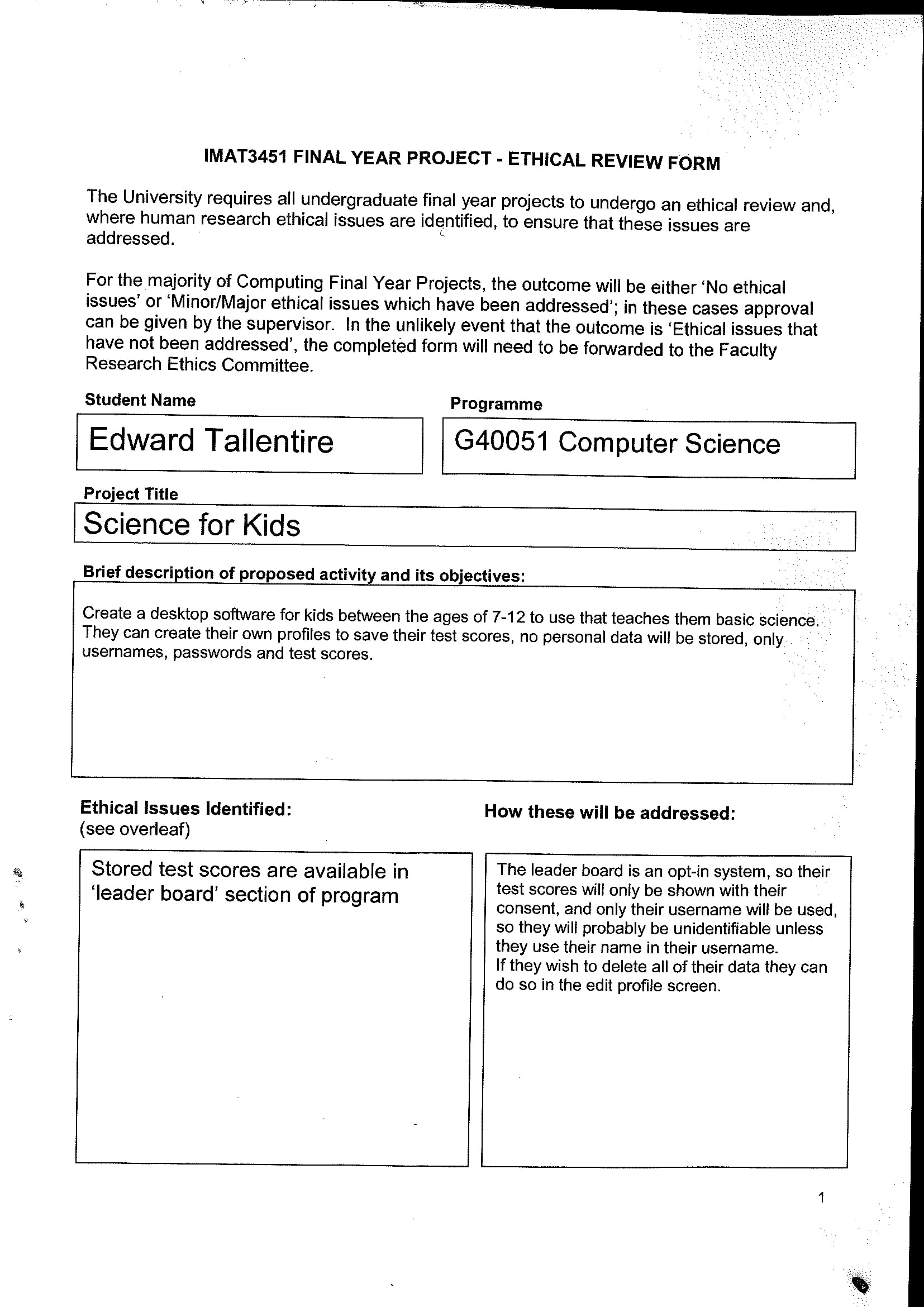
Fix it quickly or continually save all progress on an external drive and whilst it is being repaired continue project using university resources such as computers in labs and libraries.

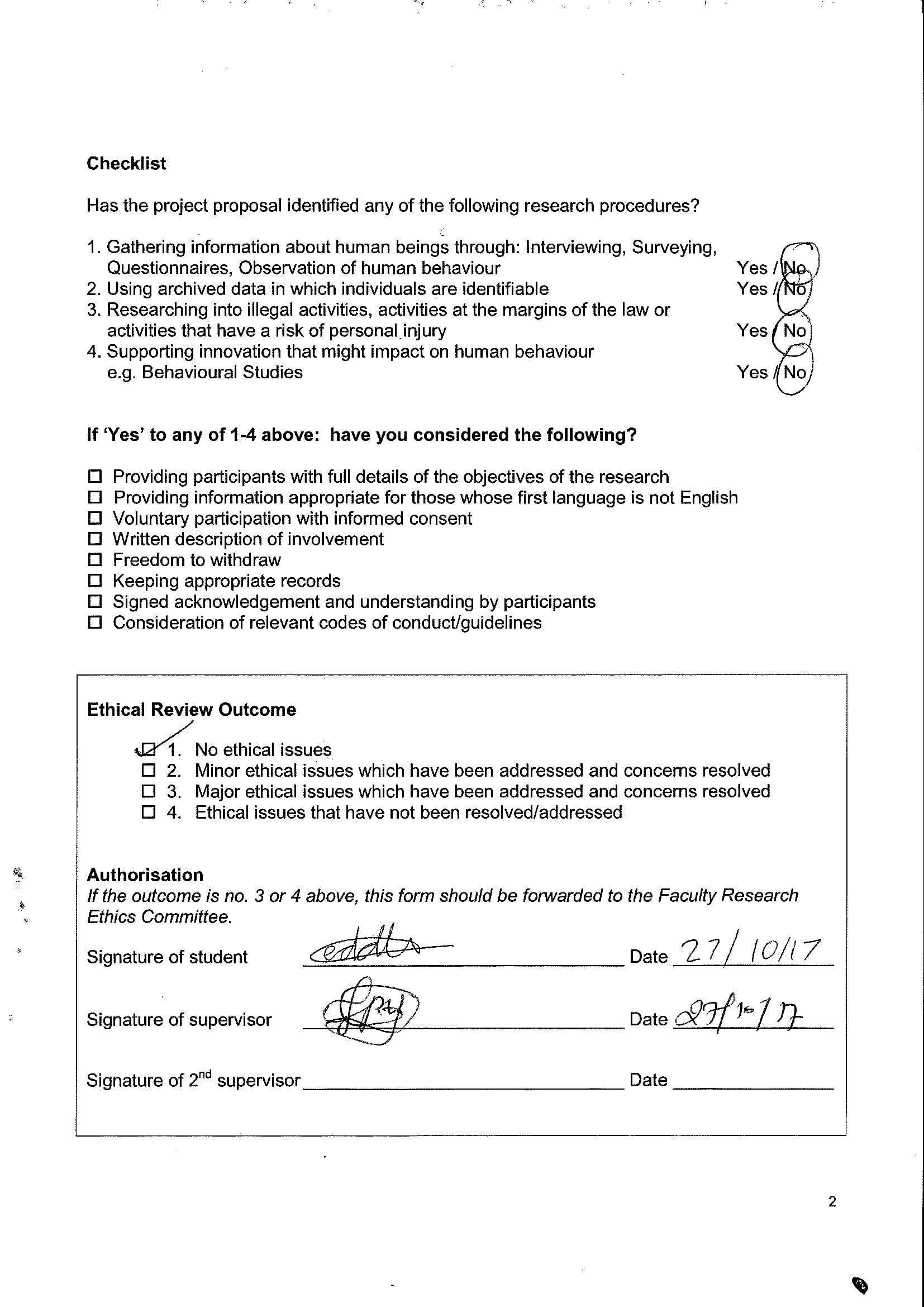
Basic Knowledge of programming languages used.

I have basic knowledge of Java, SQL and CSS, this may pose a problem during the project by delaying development because of the need to learn more complexities of each language, whilst coding with them.

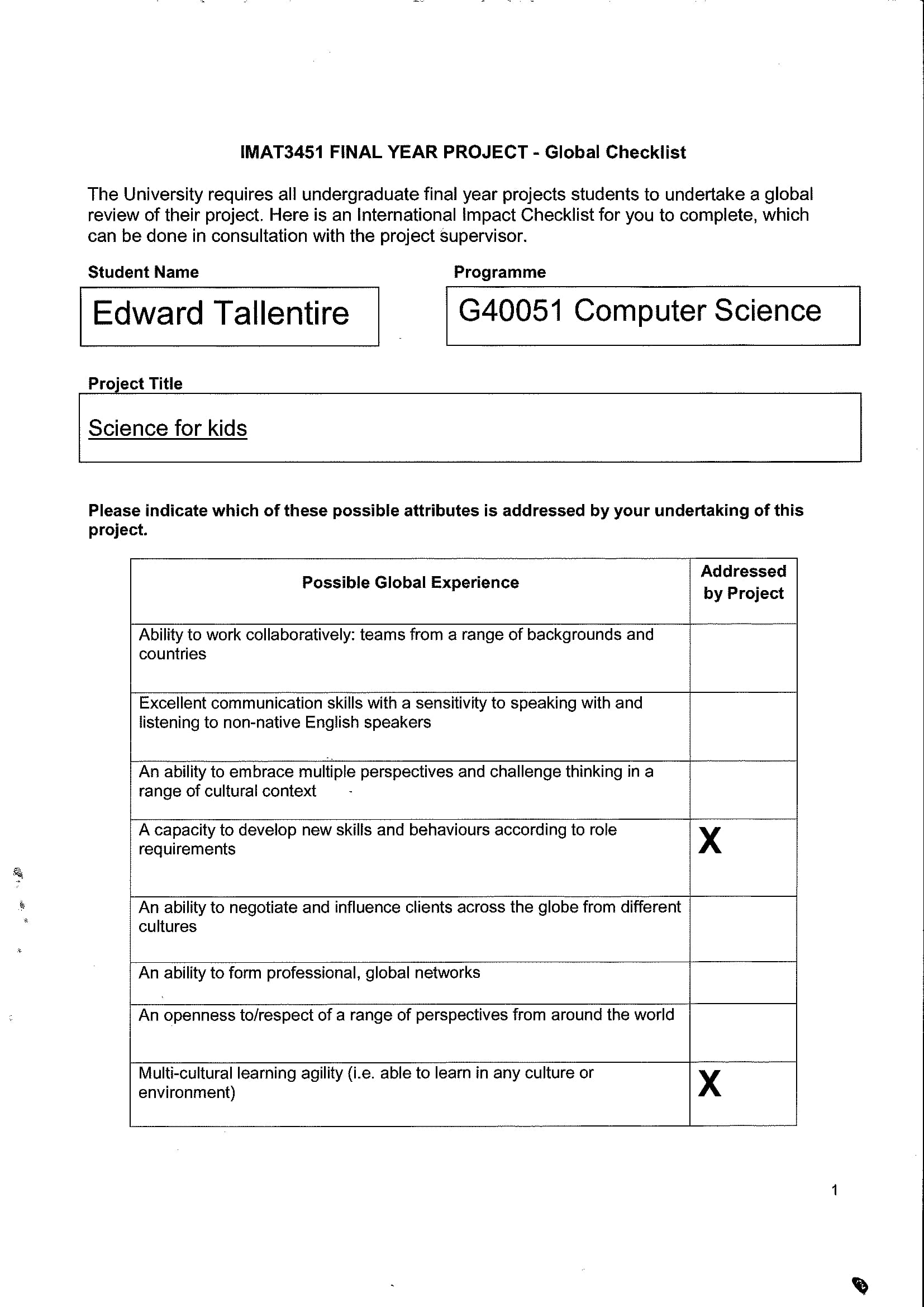
**Schedule of Activities:** Gantt chart is included

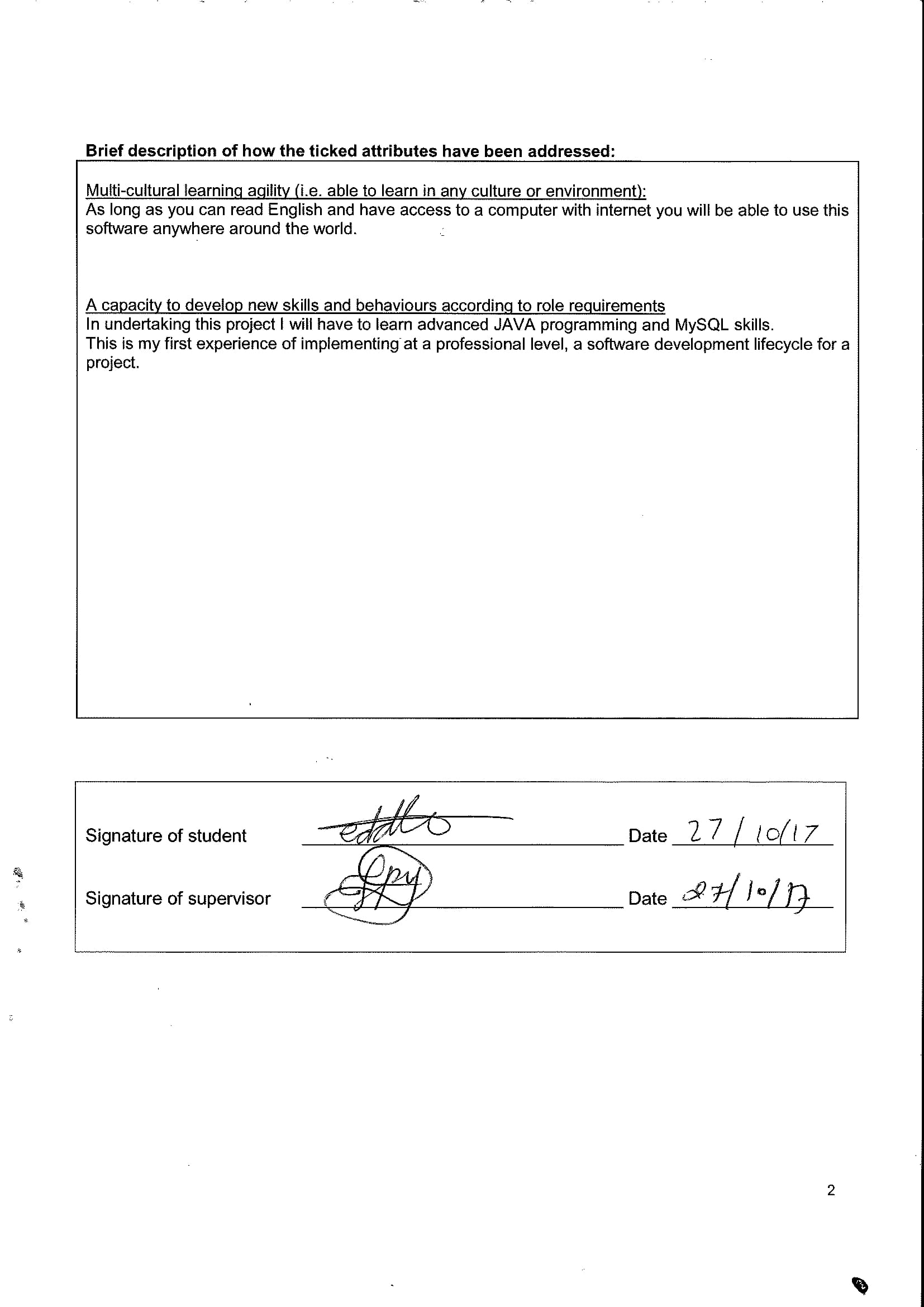
## Ethical Review Form



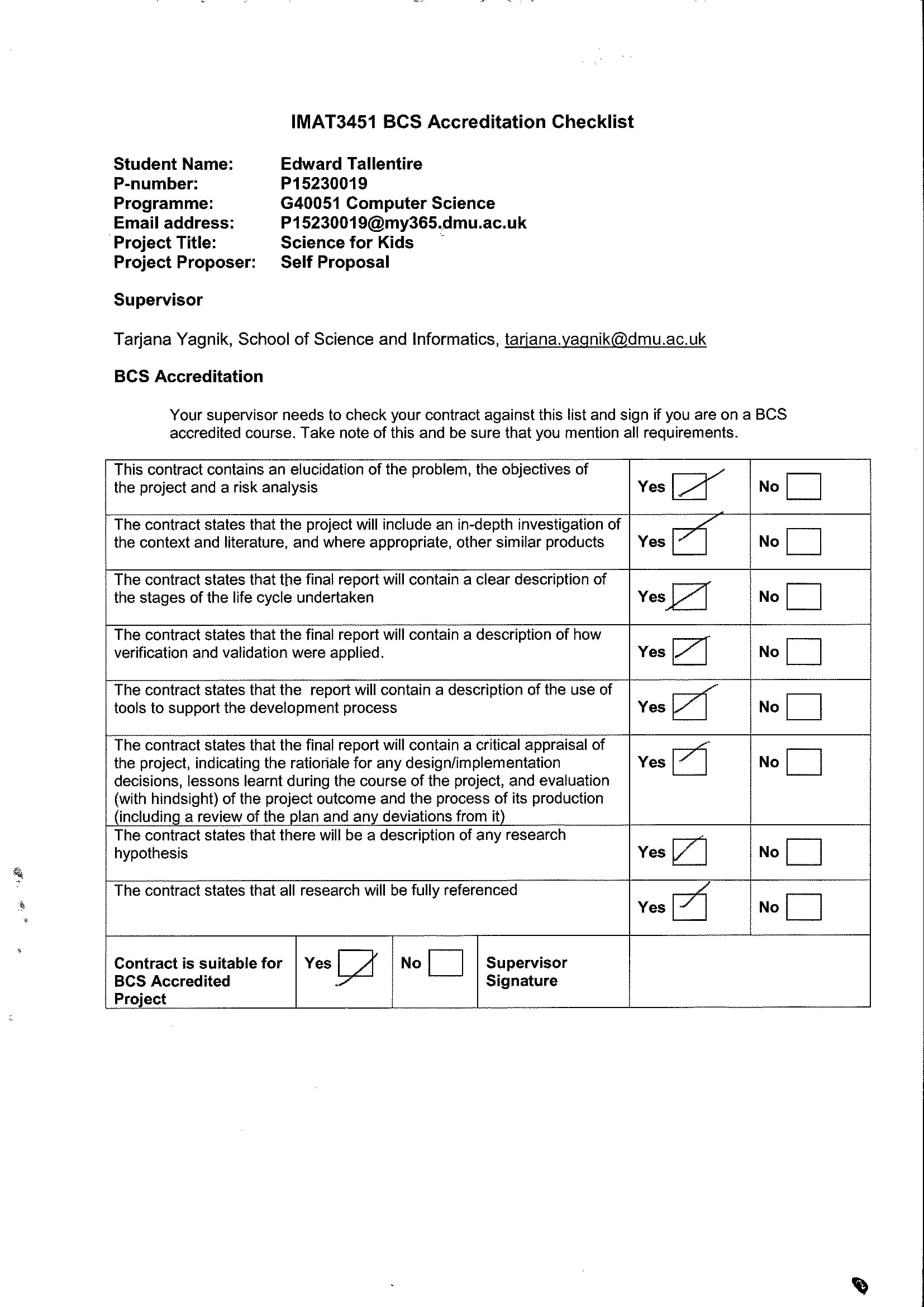


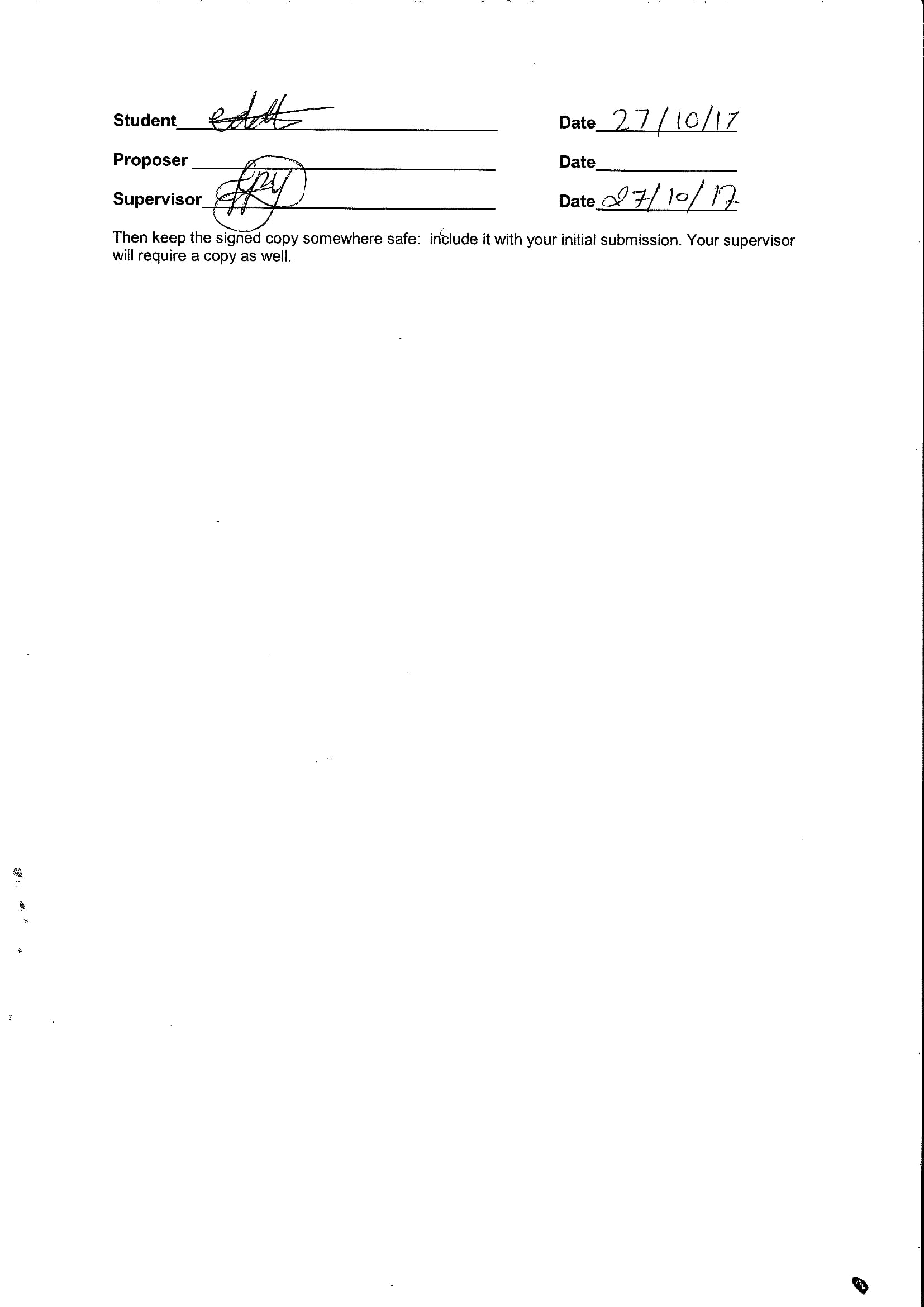
## Global Checklist



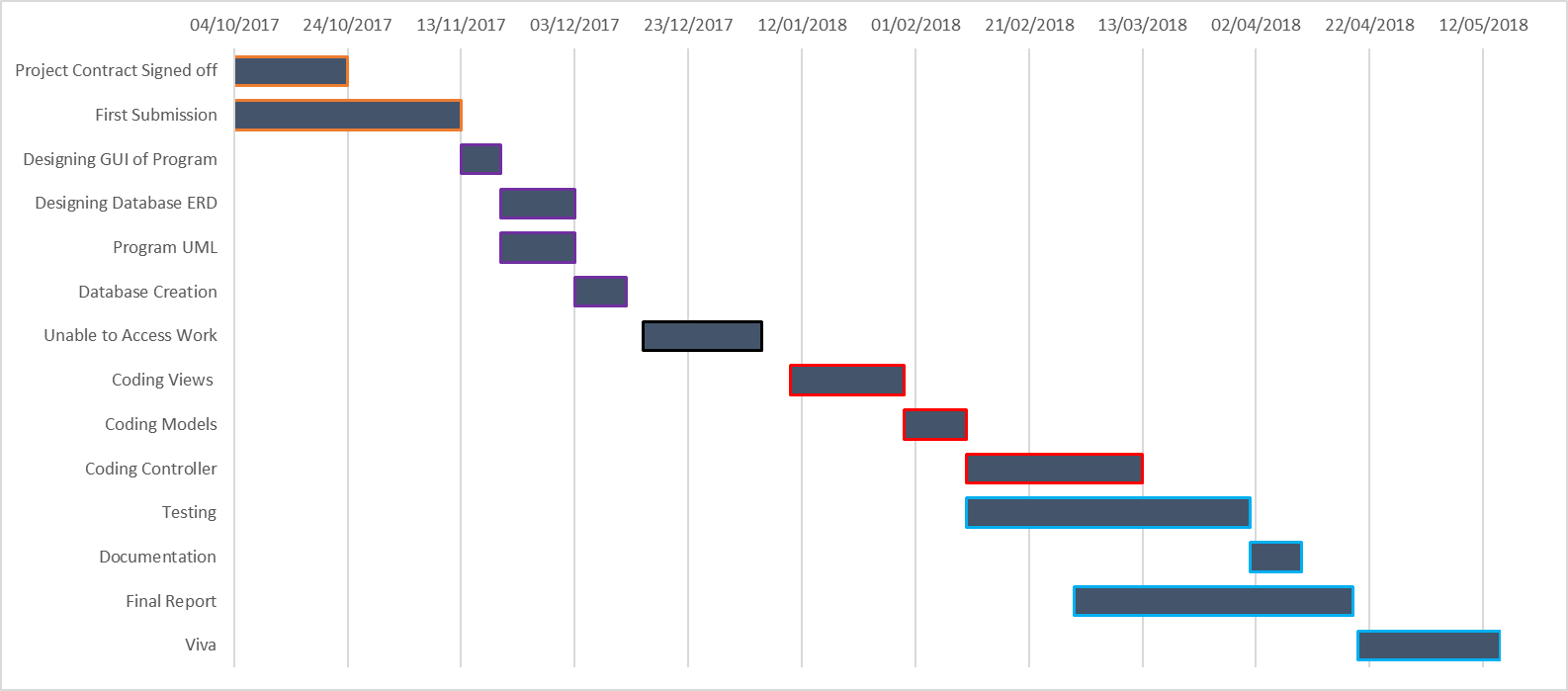


## BCS Checklist





## Gantt Chart



## Literature review

*Introduction.*

Finding the right techniques to teach children and teenagers can be challenging, they may not allocate enough time to fully learn each subject; or may just complete the bare minimum to alleviate their parents/teacher’s complaints. This will lead to poor recollection of learning materials during tests and examinations which are becoming even more important today than for previous generations. Another research topic relevant to the project is women’s representation in STEM jobs and how important it is to tackle subject gender bias when teaching children.

This literature review will also focus on rewarding users based on their results and how to structure meaningful feedback.

*Time allocation.*

We know that allocating time for things that we don’t want to do or perceive as useless and boring is very hard, even as an adult struggling to ‘find’ time for chores can be hard. Children view school the very same way, ‘maths classes are useless’ or “When are we ever going to use this Pythagoras function”, and thus their bias against learning begins at a young age.

Dufresne and Kobasigawa 1989 study (cited in Mazzoni & Nelson 2014), ‘assessed spontaneous allocation of study time for children in Grades 1, 3, 5 and 7’, the children were asked to learn related and unrelated pairs of information until they remembered them perfectly. As the program being developed is targeted towards people around the ages of 7-12 I am only focusing on the results of the Grade 3, 5 and 7 children. The results showed that ‘Grade 3 children spend around the same amount of time on the hard pairs as they did the easy pairs, whereas the older children allocated more time to the learning of the harder pairs. This research has shown me that I may need to include within my program an indication of how much time should be spent on each of my learning materials, so that the younger children using my program will not be disadvantaged when trying to learn harder material.

*Women’s motivation for STEM.*

In recent years there has been an increase in the number of women in STEM jobs, many have accounted this to the rise of feminism and other gender equality movements. And ‘although the gender gap has dramatically narrowed in recent decades, women remain underrepresented in many

science-related fields’ conclusion drawn from the National Science Foundation 2008 studies (cited in Leaper, Farkas and brown 2011).

There could be a range of factors that relate to this fact, such as bias when teaching students, societies general views on gender norms, peer pressure and even parental expectations.

(Leaper Farkas and Brown 2011) conducted a study into ‘Adolescent Girls’ Experiences and Gender-Related Beliefs in Relation to Their Motivation in Math/Science and English’.

The study ‘examined social and personal factors in relation to adolescent girls’ motivation in STEM (math/science) versus non-STEM (English) subjects.’ They found that ‘Personal factors also may influence girls’ math and science achievement. Learning about feminism and endorsing gender equality were positively associated with girls’ motivation in math and science.’

This is relevant to the project as one of the programs main goals is to increase the participation of girls in STEM related subjects. It shows that care is needed when phrasing the learning material so that it isn’t gender specific e.g.: ‘ryan did this’, or ‘max had a’.

This same study also analysed girls from different sociocultural backgrounds and the ‘Preliminary analyses indicated significant ethnic differences in academic motivation and achievement.’ This was shown by the results: ‘White European American girls scored higher than Latina girls in M/S grades. English grades, and M/S motivation.’

These cultural differences will need to be taken into consideration when the program is developed so that no child feels discriminated against when using the program and can fully benefit from it.

*Rewards.*

Intrinsic motivation is defined as performing an action or behaviour because you enjoy the activity itself (Study.com, 2017). In an ideal world we would want children to be Intrinsically motivated to learn for themselves, to learn because they enjoy the challenge, to be more knowledgeable.

As such I have planned little rewarding content within my program, the only things that may be defined as rewards are congratulatory messages within the program if the students have done well on tests, or the leader board system where users can see their overall scores compared to other users.

The findings of a meta-analysis done by (Deci, Koestner and Ryan (1999)) show that ‘tangible rewards offered for engaging in, completing, or doing well at a task were deleterious to intrinsic motivation’ and in respect to age effects, ‘tangible rewards are more detrimental for children than for college students …. and that verbal rewards do not enhance free-choice behaviour for children…’.

This free choice behaviour is very important as the program is in aid of, and not meant to replace real world teaching. The program is meant for children to be able to develop and enhance their knowledge of science, whilst keeping them personally motivated to study science or STEM related subjects further down the line; as stated above it is detrimental for rewards to be given out to children as it negatively effects their freedom of choice to study, so therefore any rewards within the program will have to be very carefully implemented.

*How to structure questions.*

When thinking about how to test the user’s knowledge a lot of things spring to mind, what parts of the learning material should be tested? will the test cover everything being taught? What sort of a timeframe is the user working with? And how feasible is it to give feedback to everyone?

The main source of my reason to go with multiple choice questions is that the feedback to the answers for these questions are objective. When giving feedback on an open-ended question you must subjectively think about their answers and where they went wrong and therefore tailor your answers to each individual person. Whereas with multiple choice questions I can hardcode in the correct answers, which will leave me some time to develop more useful functions within the program.

The website (Helpteaching.com, 2017) has given me some useful tips on how to structure my questions. Such as ‘Don't use excessive wording when creating the test question stem. Be clear and concise in your word and phrase choices.’

‘Be sure to use test questions that test knowledge, application, comprehension, analysis and evaluation throughout your test to get the best overall sense of the student's understanding and mastery of a subject matter.’

*Periodical Feedback.*

Getting feedback is an integral part of the learning process but how much feedback should be given?

Should there be minimal feedback which would allow the student to find out for themselves where they went wrong? or at the other end of the scale, detailed feedback pertaining to each individual wrong answer?

Feedback Interventions are one of the most common ways of learning, FI’s provide people with information of their performance on the related task. E.g.: KR interventions (Knowledge of results) is a form of Feedback Intervention, an example would be at the end of a test it displays your score or how long it took you to complete the test. Most people would assume that FI’s would always help in any situation but ‘Fls are double-edged swords (U.S. Congress, Office of Technology Assessment, 1987) because Fls do not always increase performance and under certain conditions are detrimental to performance.’ (Cited in Kluger and DeNisi, 1996).

A study was conducted to find out the effects that 3 different types of feedback had on test performance, these feedback types were no feedback, answer feedback, or verification feedback.

Answer feedback states the correct answer, verification feedback replies with a correct or incorrect and no feedback gives nothing. The results were as follows ‘Without feedback, participants corrected only a small proportion of their errors, but improved dramatically following verification feedback. However, performance following verification feedback was still well below the level of error correction observed following answer feedback (Marsh et al., 2012).

After seeing these results, I will make sure to include the actual answer to each multiple-choice question in the feedback section to my quiz, not just a correct/incorrect statement with a mark at the bottom.

I chose this study because my program will be using multiple choice as a testing method, therefore I wanted the literature selected to be as transferrable as possible. Also, in this study after receiving feedback they answered short questions about the same topics, this applies to the real-world effects that the program being developed could have.

*Analysis of Websites*

Websites used.

<https://www.khanacademy.org/>

<https://www.bbc.com/education/levels/zbr9wmn>

<https://uk.ixl.com/>

**Navigation**

*Khan academy*

From Homepage to topic: 4 clicks, all the subjects with varying levels of difficulty are displayed on the Home Page, an option to Sign Up takes you to a different page but it’s not necessary to view content. The login feature creates a ‘pop-up’ box that covers the entire webpage, this can be closed. When a subject is chosen from the homepage a flow-pane like list is displayed with topics and small corresponding images. Once chosen a rather complex activity screen loads with various items to complete, alike to a timeline. Once you have completed 2 quizzes and all the learning material there is an ‘End of Unit Test’.

*BBC Bitesize*

From Homepage to topic: 4 clicks, the first thing you are faced with is choosing your level of study this ranges from KS1 to GCSE. After selecting your study level, a small navigation bar is included above the study level title. At this screen you can select the subject you wish to study; each subject has a related small grayscale picture. This leads to a page displaying the topic titles and subtitles in a long list, once the topic is chosen it loads a new page with the broken-down sections of the topic in small interactive titbits.

*IXL*

From Homepage to topic: 2 clicks, this website has the benefit of teaching maths or English, therefore they can skip the subject selection screen if you want to study maths, otherwise you can navigate to English through an unclear link.

Once you’ve chosen your level of study you are blasted with an extremely long webpage with lists upon lists of subject matter, Ctrl + F helps.

*Summary*

After navigating through these 3 websites I found some common themes.

The average number of clicks to navigate from homepage to learning material is 3.5 (rounded to nearest half). This means that I should make sure my users don’t have to navigate through too many screens before they get to the learning material. 3 or 4 clicks of navigation will be ideal.

After being faced with IXL’s lists upon lists of topics, an important design feature will be to make sure the user isn’t faced with this daunting prospect as we want to make learning as enjoyable as possible, not making it seem like an impossible task to learn everything.

When using Khan academy, the login pop-up feature was useful and convenient.

There was a theme of using small images that are relevant to either the topic or subject, this should be implemented within ScienceForKids as it seems to work very well, also my target audience are old children, therefore having images dotted about the program will only serve to increase their enjoyment of the program.

**Tools used to help user learn**

*Khan academy*

They use a mix of YouTube videos and documents with text and diagrams. Both are well implemented, especially the document which looks clean. Unfortunately producing YouTube videos for the topics in ScienceForKids will be too much work to complete successfully, especially with only 1 person actively working on the entire project, within the timeframe given.

*BBC* *Bitesize*

These titbits have small interactive scenes within them which ‘test’ you as you go along learning. BBC Bitesize does not save user progress.

To guide the user to their next topic they have a part at the end of the page containing various images with links to relevant topics.

*IXL*

Very limited learning features, only quizzes users. If the user gets incorrect answer they show user a basic example with answers.

Must become a member to use the website for ‘unlimited practice’.

IXL will be talked about more in Quiz Techniques & Feedback.

*Summary*

Prevalent use of YouTube/videos for learning, common theme of interaction with website. Unfortunately, neither are viable for the project.

Common use of images to go alongside text and to help with explanation of subject matter.

**Length of subject matter**

*Khan academy*

Videos on average 6.5 minutes, Documents on average 5 pages for higher level topics and for lower level topics around 3.

*BBC Bitesize*

Each topic is broken down into small sections, each section has a ~2 Minute video with an interactive scene and a small amount of text.

*IXL*

N/A learning is done by quizzing, there is no obvious learning materials on the website that I can find.

*Summary*

Khan academy and BBC Bitesize both split their topics into various smaller sub-sections. But the videos were a substantially different in their time.

**Quiz Techniques & Feedback**

*Khan academy*

Signing up saves users progress.

Contains quizzes that are placed at the end of a sub-section of a topic.

In the quiz even if you get the correct answer it gives you an option to show a step-by-step solution via various hints, if you don’t wish to have the hints you can just click on the video related to the sub-topic.

*BBC Bitesize*

Contains quizzes within the learning content instead of it being an individual section, testing early and often.

In the quiz if you get the wrong answer it doesn’t show you the correct answer, but it lets you retry the question with some hints.

*IXL*

IXL provide Analytics, ‘Immediate insights into student performance’, ‘Valuable information at your fingertips’, ‘Data for more efficient teaching’, ‘IXL will track your score, and the questions will automatically increase in difficulty as you improve!’ from <https://uk.ixl.com/analytics/>

Nothing new here, only difference is the level of analysis done. Time recorded + a ‘smart score’.

‘IXL's SmartScore is a dynamic measure of progress towards mastery, rather than a percentage grade. It tracks your skill level as you tackle progressively more difficult questions.’

*Summary*

The IXL analysis is very interesting but irrelevant to the project.

The theme here seems to be testing in small bits, often with feedback/step-by-step instructions on where they went wrong.

**Colour Schemes/Design**

Web designs are going to be a little different than application designs, so the summary will be liberal.

*Khan academy*

Varying blue colours throughout the background of the headings, white & grey used for the background of the main content. Content grouped into white boxes, with grey as the true background. Main navigation bar at the top.

*BBC Bitesize*

Light-Deep-Blue, Orange background for Headers, Black Font Colour, Grey Object Background, break-lines in-between sections of webpage. Page is heavily sectioned to make content change obvious.

*IXL*

Each year of study has a different colour associated with it, green is the main colour of the website.

Grey & white used as background of grouped content. Each bit of content is grouped into its own pane. With 2 navigation bars at the top.

*Summary*

Varying colours of blue are the primary colours for the heading sections of these webpages, e.g.: Topic Headers/sections and navigation.

Content is heavily grouped in sections instead of having a constant scrolling page.

All 3 had navigation bars at the top, this may not be implemented into the program because of difference between how websites & applications are made.

Bibliography

Mazzoni, G. and Nelson, T. (2014). *Metacognition and cognitive neuropsychology*. New York, New York: Psychology Press, p.12.

Leaper, C., Farkas, T. and Brown, C. (2011). Adolescent Girls’ Experiences and Gender-Related Beliefs in Relation to Their Motivation in Math/Science and English. *Journal of Youth and Adolescence*, 41(3), pp.268-282.

National Science Foundation. (2008). Bachelor’s degrees, by sex and field: 1998–2007. Retrieved August 24, 2010 from http://www. nsf.gov/statistics/wmpd/degrees.cfm#bachelor.

Deci, E., Koestner, R. and Ryan, R. (1999). A meta-analytic review of experiments examining the effects of extrinsic rewards on intrinsic motivation. *Psychological Bulletin*, 125(6), pp.627-668.

Study.com. (2017). *Intrinsic Motivation in Psychology: Definition, Examples & Factors - Video & Lesson Transcript | Study.com*. [online] Available at: http://study.com/academy/lesson/intrinsic-motivation-in-psychology-definition-examples-factors.html [Accessed 17 Nov. 2017].

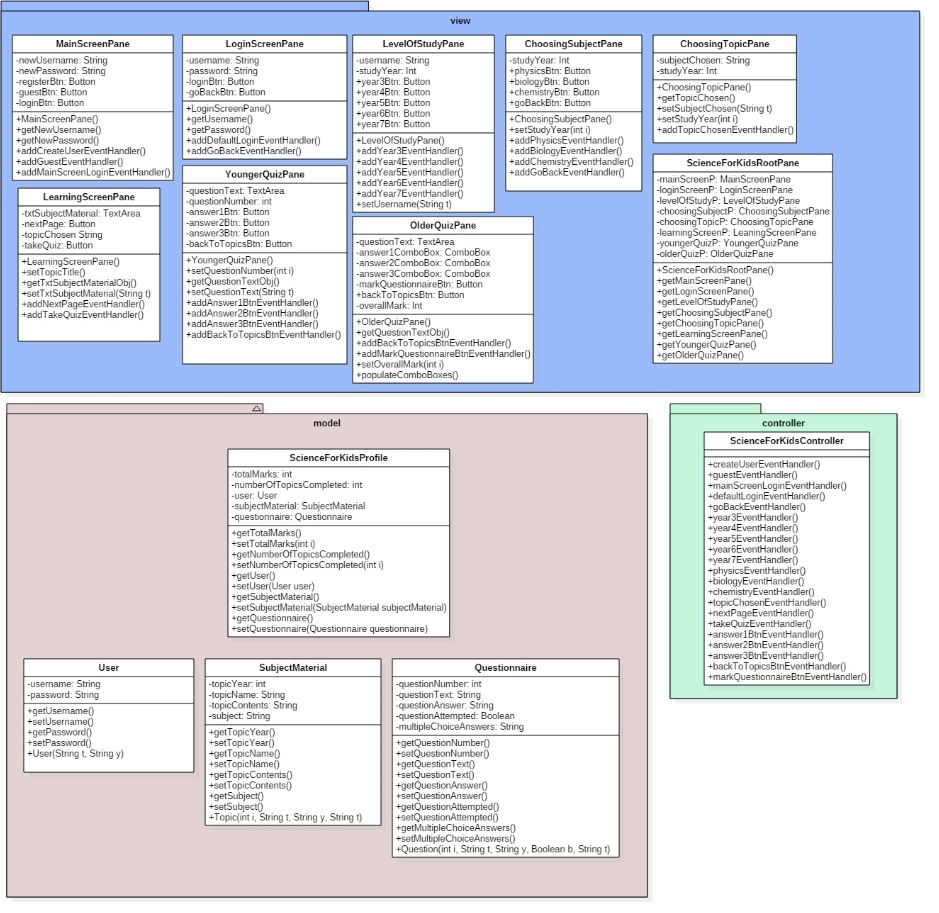
Helpteaching.com. (2017). *How to Write Good Test Questions - HelpTeaching.com*. [online] Available at: http://www.helpteaching.com/about/how\_to\_write\_good\_test\_questions/ [Accessed 17 Nov. 2017].

Kluger, A. and DeNisi, A. (1996). The effects of feedback interventions on performance: A historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological Bulletin*, 119(2), pp.254-284.

Marsh, E., Lozito, J., Umanath, S., Bjork, E. and Bjork, R. (2012). Using verification feedback to correct errors made on a multiple-choice test. *Memory*, 20(6), pp.645-653.

## UML Diagram before

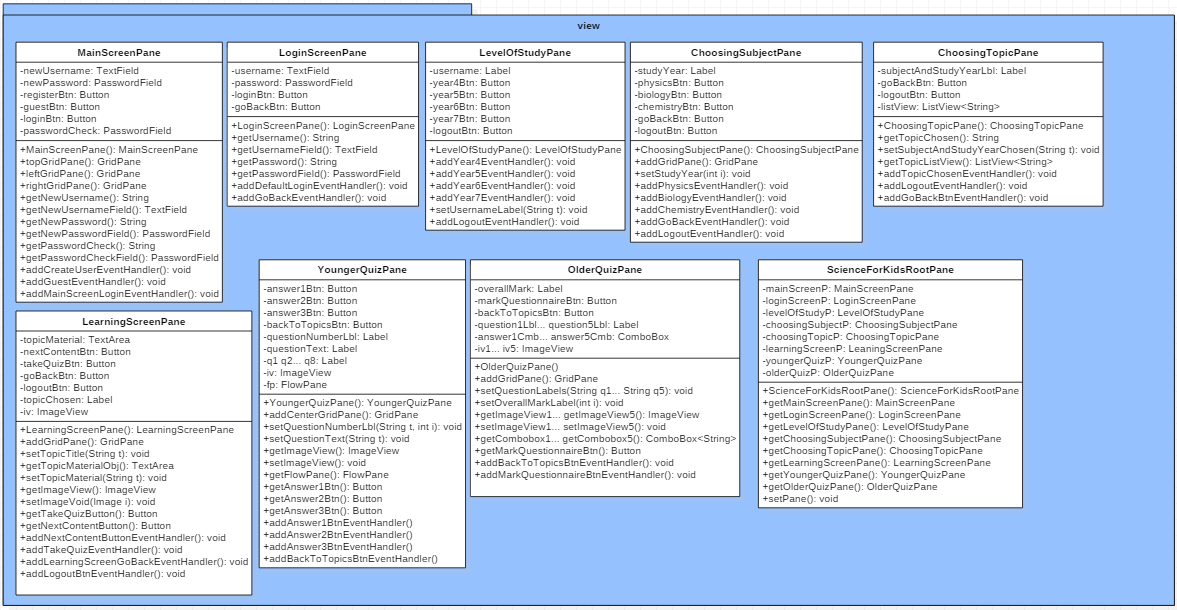
Link to Development/Implementation 🡪 Coding [Views](#_Coding_Views)/[Models](#_Coding_Models)/[Controller](#_Coding_Controller)



## UML Diagram After

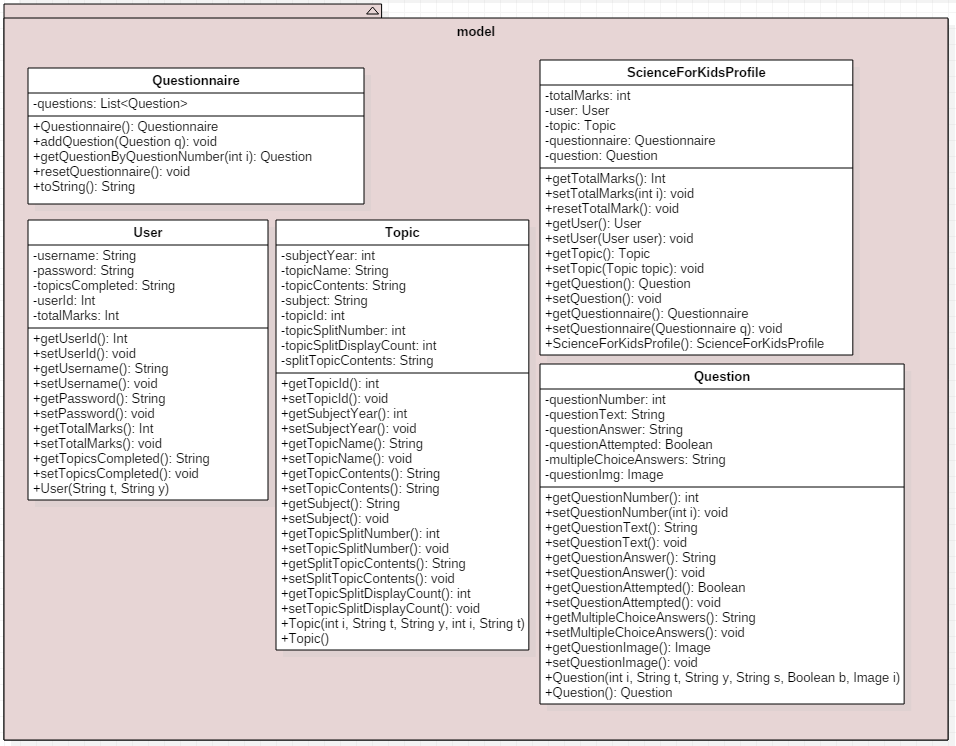
### View

Link to Development/Implementation 🡪 Coding [Views](#_Coding_Views)



### Model

Link to Development/Implementation 🡪 Coding [Model](#_Coding_Models)



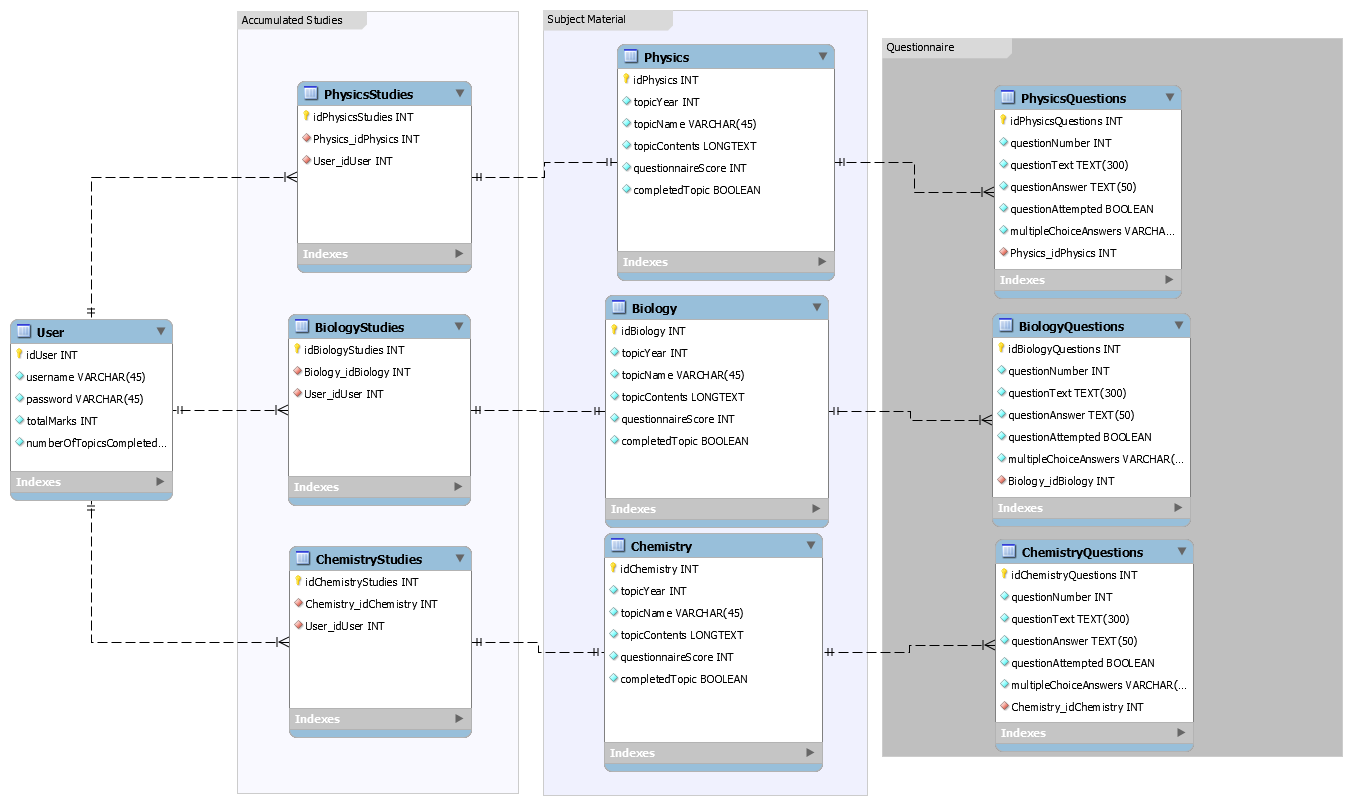
### Controller

Link to Development/Implementation 🡪 Coding [Controller](#_Coding_Controller)



## ERD Diagram

[Back to ERD Design](#_Entity_Relationship_Diagram)



## User Manual

Welcome to the ScienceForKids User Manual!

The program is made to be easy to use but if you get stuck and want to know more, read this manual for the low down on how to get smart.

### MainScreenPane Image

[(Back to Implementation)](#_Coding_MainScreenPane,_main)



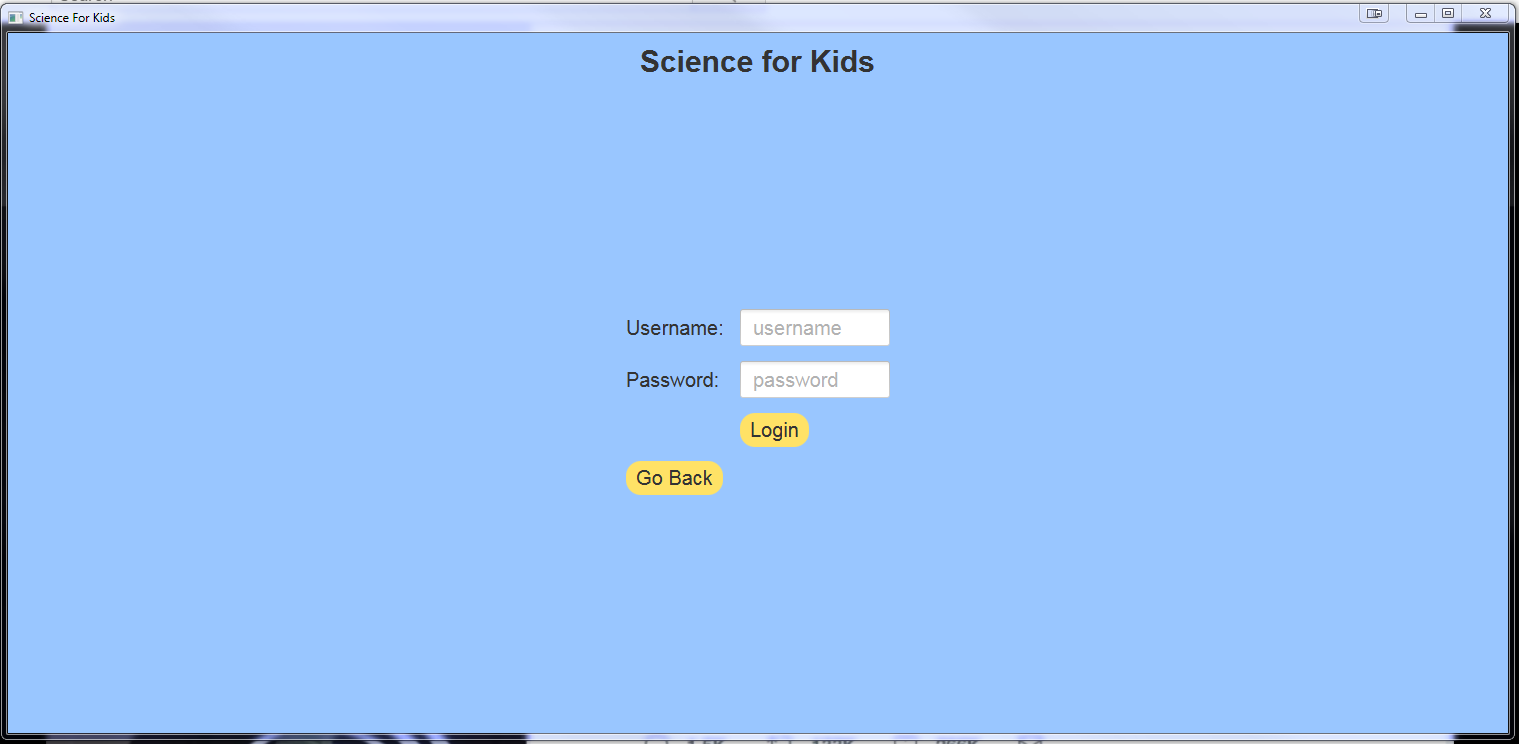
If you wish to save the topics you have completed and the total marks from all the tests, then you should create an account. Simply input your own custom username and personal password but make sure to remember these as there is no current way to retrieve lost usernames or password yet. Maybe try writing it down somewhere where you won’t forget or lose sight of it.

If you simply wish to use the program without saving any of your scores click on the Continue as Guest button indicated by the arrow in red.

If you have previously registered an account, click the login button to be transferred to the login page pictured below.

### LoginScreenPane Image

[(Back to Implementation)](#_Coding_LoginScreenPane,_main)

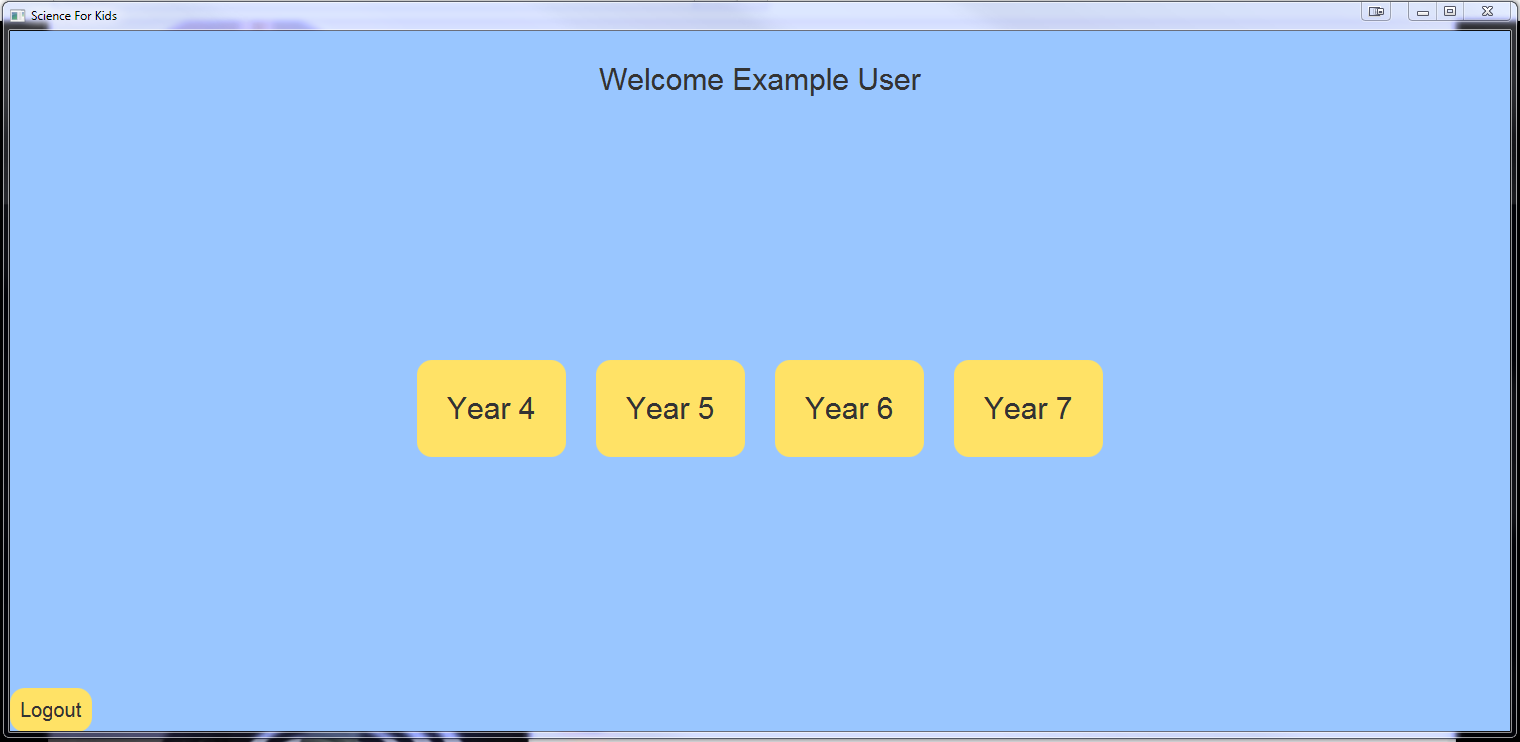


If you have previously made an account and remember your username and password enter them into the appropriate boxes above and click the login button, this will transfer you to the screen below where you will select the level at which you wish to study.

In the future there will be a ‘forgot password?’ option for those of you who have forgotten your password, but right now we have given you unlimited tries at logging in, if you really can’t remember your password then we are sorry but you will have to make a new account, click the Go Back button to navigate to the main page pictured above.

### LevelOfStudyPane Image

[(Back to Implementation)](#_Coding_LevelOfStudyPane,_main)



If you wish to logout at any time, press the logout button that will be located somewhere on your screen, like the one shown in the bottom left hand corner here.

To continue through the program simply press one of the four ‘Year’ buttons in yellow, these buttons select the level of topics that you feel like learning about today. The buttons will transfer you to the next screen to select which subject you wish to study.

### ChoosingSubjectPane Image

[(Back to Implementation)](#_Coding_ChoosingSubjectPane,_main)

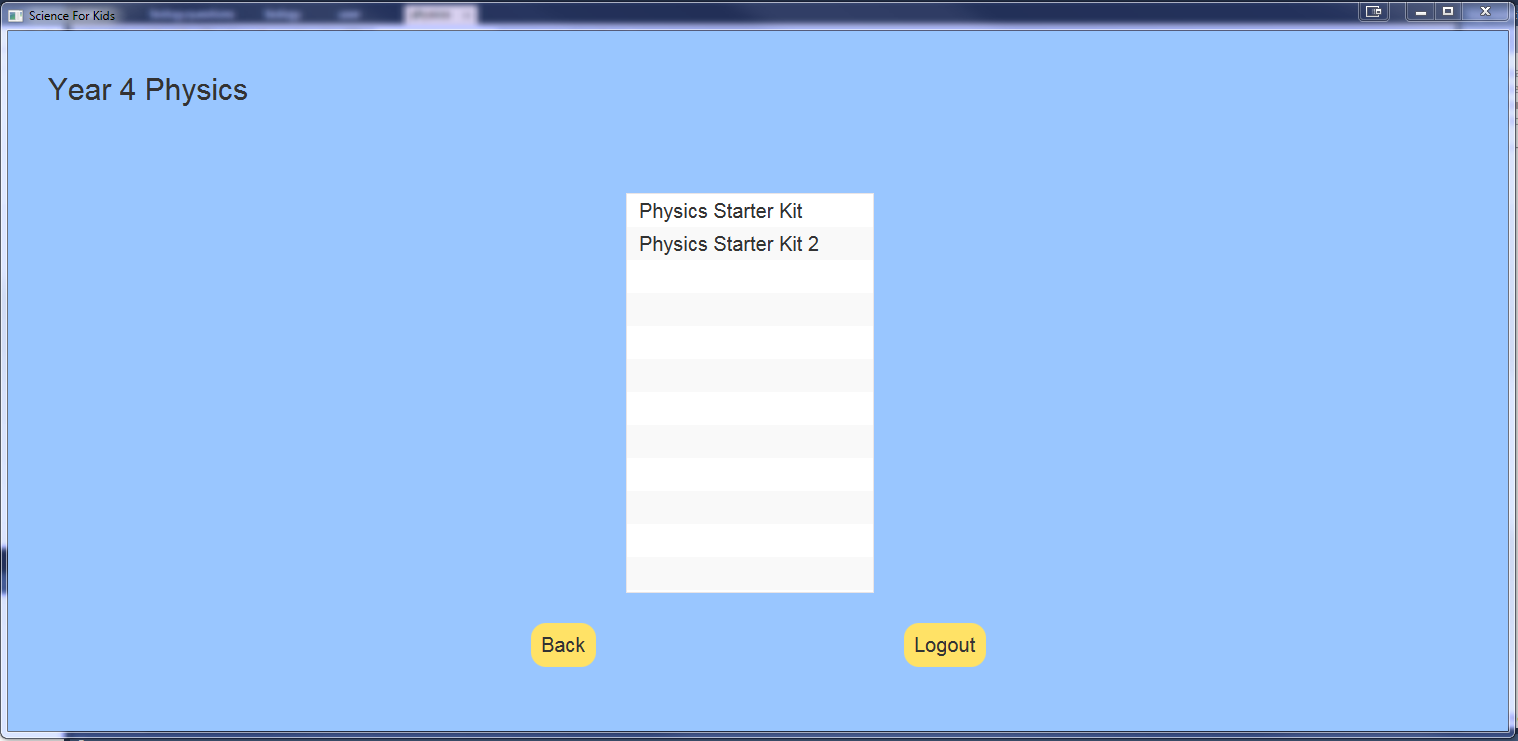


Select which subject you wish to study, this transfers you to the screen below where you can select which topic you wish to learn from.

If you want to change your level of study, simply press the ‘Go Back’ button in the bottom left to reselect a new year.

### ChoosingTopicPane Image

[(Back to Implementation)](#_Coding_ChoosingTopicPane,_main)

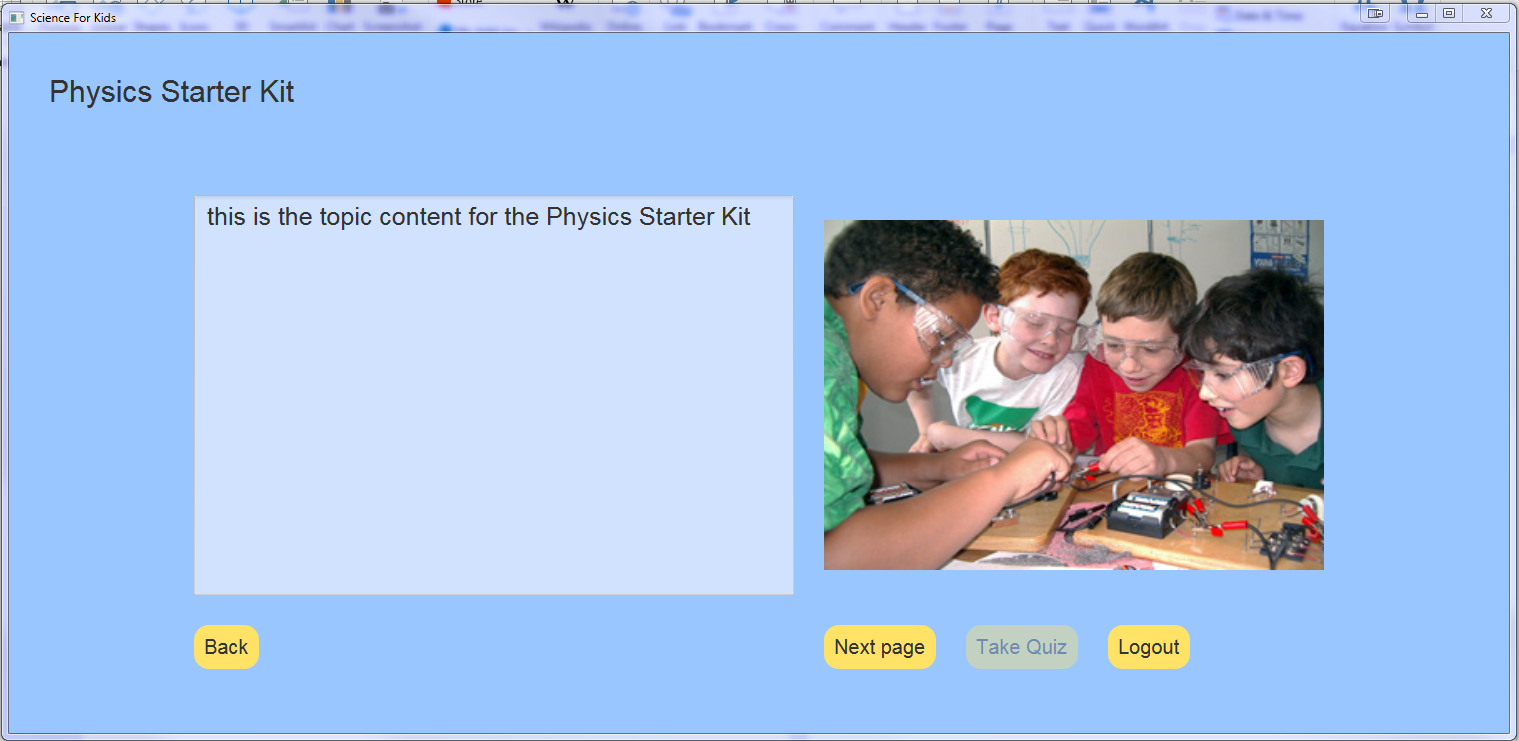


Now you can select which topic you want to learn, at the lower levels basic starter kits are available as an introduction into the subject, at higher levels harder topics will be available, as time passes more and more topics will be added to each list so be on the lookout!

Just click on the topic name once and you will be transferred to the screen below which will contain the topics contents.

### LearningScreenPane Image

[(Back to Implementation)](#_Coding_LearningScreenPane,_main)

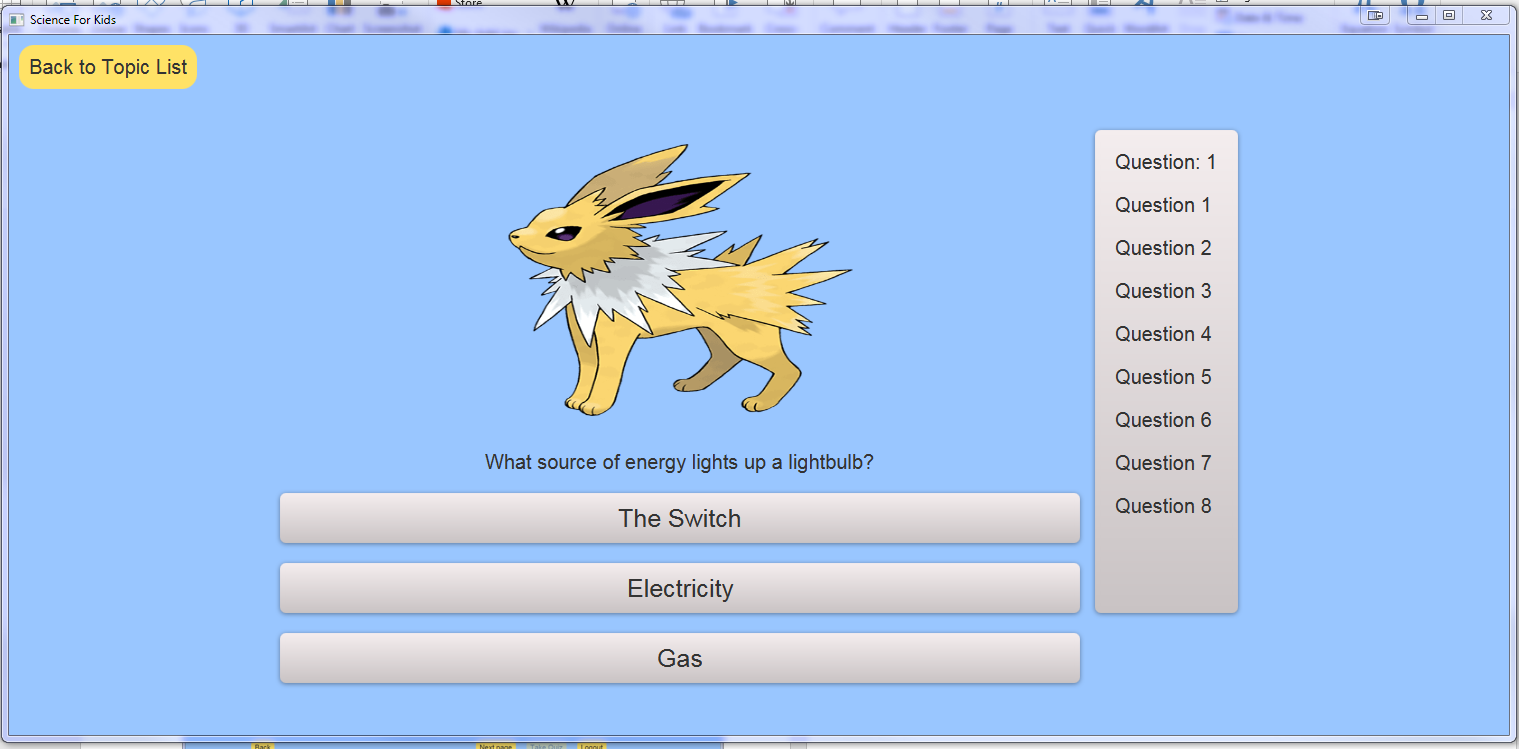


You’ve finally got to the most important part of the program, the learning screen, this light blue box will contain the topic contents for your chosen topic, each topic will have multiple pages of contents so once you’re done studying one page you can click the Next Page button to see the next page of content.

Once you have studied all available pages of content the ‘Take Quiz’ button shown here will become clickable, when you feel ready to test your newly learnt knowledge press the Take Quiz Button.

### YoungerQuizPane Image

[(Back to Implementation)](#_Coding_YoungerQuizPane,_main)

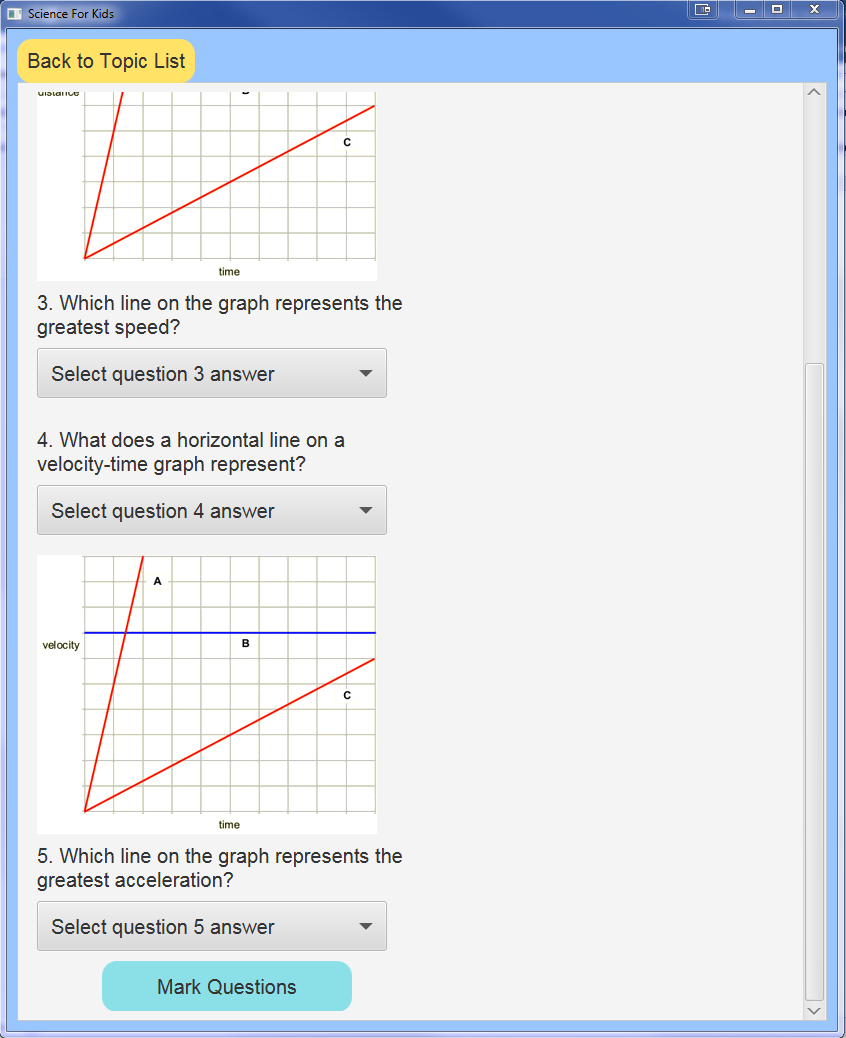


This is the screen you will be presented with when taking a quiz if you have chosen a topic from the years 4 to 5, each question will have 3 answers 1 of which is correct, select the correct answer from the buttons.

At the end you will be presented with your score, press the Back To Topic List button in the top left-hand corner afterwards to navigate away from the quiz screen.

### OlderQuizPane Image

[(Back to Implementation)](#_Coding_OlderQuizPane,_main)



If you have selected a topic from the years 6/7 this is the screen you will be presented with, use the scroller on the right-hand side of the page or your mouse wheel to navigate up and down the page.

Read each question carefully and click on the drop-down box to select your answer from a varying list of answers.

For the Mark Questions button to work you will have to have answered all the questions, once you have received your score look to the top-left corner of the page to the Back To Topics List to navigate away from the page.

## Progress Reports

### 1

**IMAT3451 Final Year Project Periodic Progress Report (PPR)**

Programme/Course Title: G40051 Computer Science

Name: Edward Tallentire Assessment Period:

Project Title: undecided Report Number: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Objectives for Period:*

Completion of the following

Project Contract

Gantt chart

Ethical form

Global checklist

BCS checklist

*Summary of Progress for Period:*

Completed and/or signed off

Project Contract

Gantt Chart

Ethical Form

Global Checklist

BCS checklist

*Problem Areas and Suggested Solutions:*

Project Name

Which Database software to use, suggestions include Oracle and MySQL.

What data should and shouldn’t not be collected

Risk Analysis, what actually constitutes as a risk in software development.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

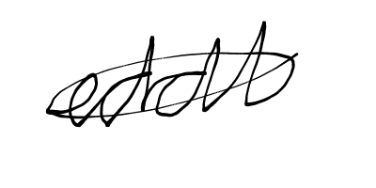
*Objectives, Deliverables & Plan for Next Period:*

Literature Review

Functional and Non-Functional Requirements

Date of Next Review: 17/11/17

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

****

Student’s Signature: Date:27/10/17

Comments (if any):

### 2

**IMAT3451 Final Year Project Periodic Progress Report (PPR)**

Programme/Course Title: G40051 Computer Science

Name: Edward Tallentire Assessment Period:

Project Title: ScienceForKids Report Number: 2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Objectives for Period:*

Completion of the following:

Literature Review

Functional and Non-Functional Requirements

*Summary of Progress for Period:*

Completed and handed in Literature Review.

Completed but forgot to attach Functional and Non-Functional requirements to Interim Hand-in.

*Problem Areas and Suggested Solutions:*

My literature review didn’t critically analyse any existing learning programs or websites.

Go over existing professional websites & programs to gather ideas for my GUI Designs and potentially extend the functionality of my program

Allow more time before hand-in deadlines to get everything into one document instead of handing it in straight after completion.

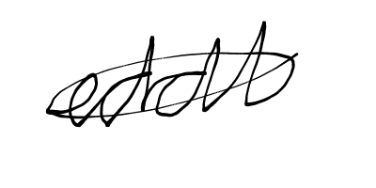
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Objectives, Deliverables & Plan for Next Period:*

GUI Designs

Extend Literature review

Date of Next Review: 19/01/18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

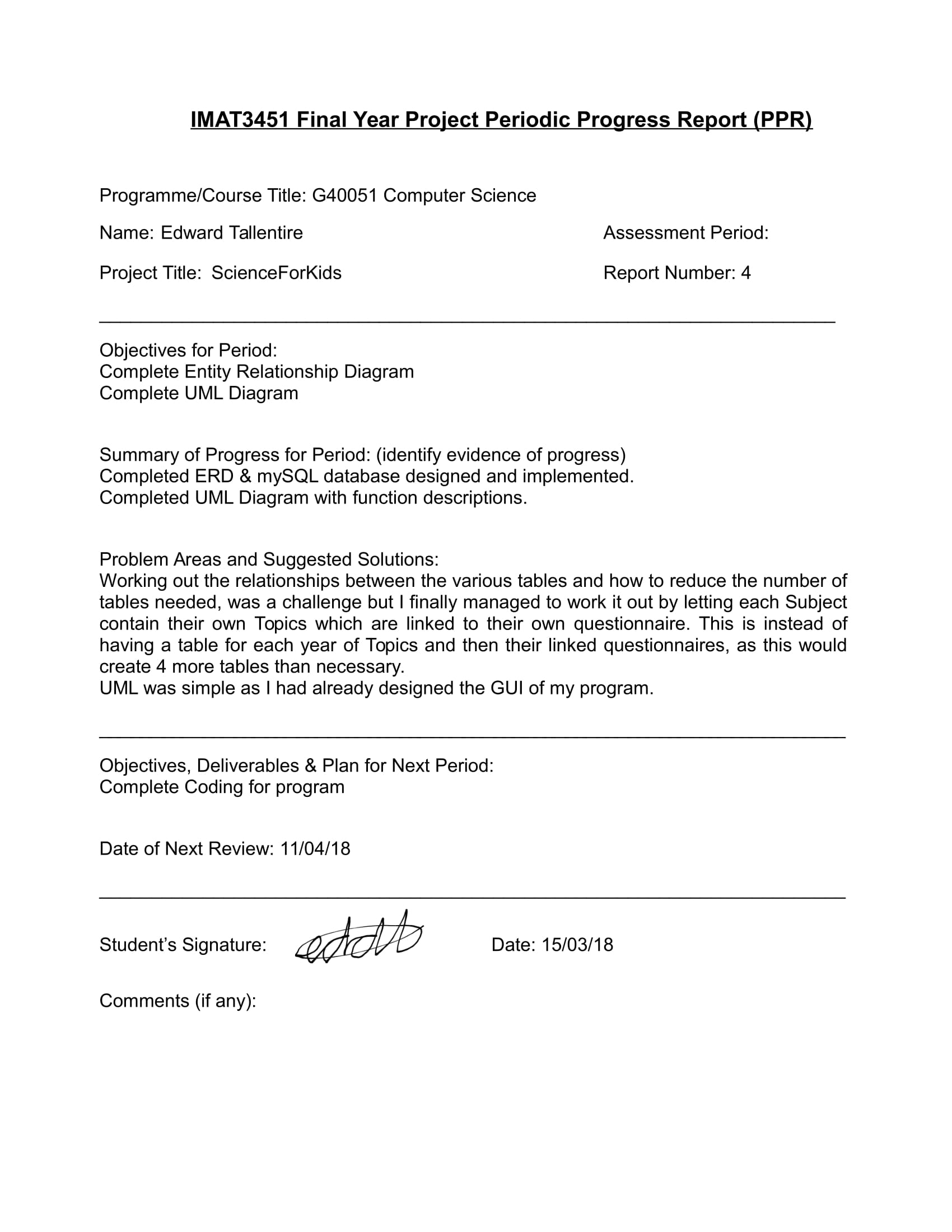
Student’s Signature: Date: 17/11/17

Comments (if any):

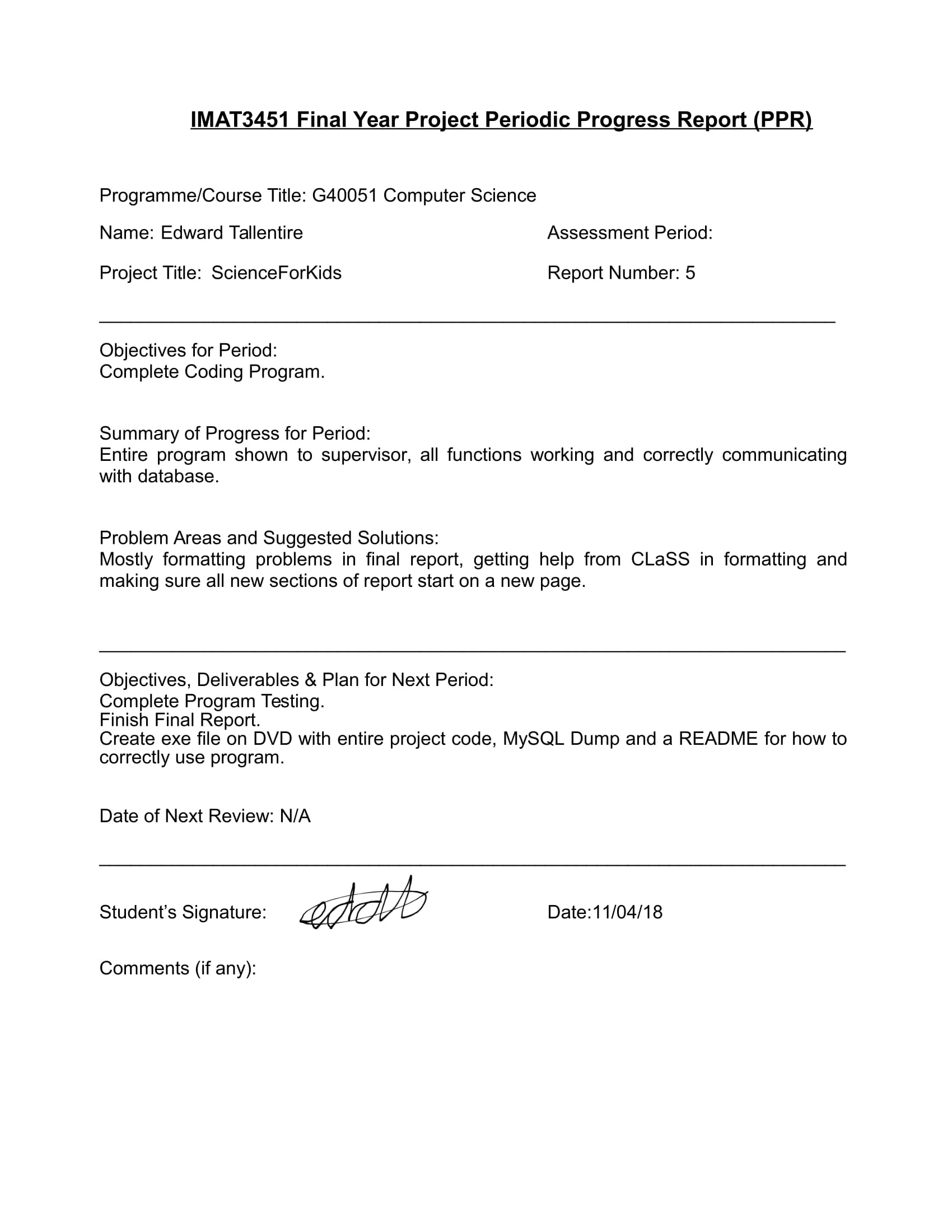
### 3



### 4

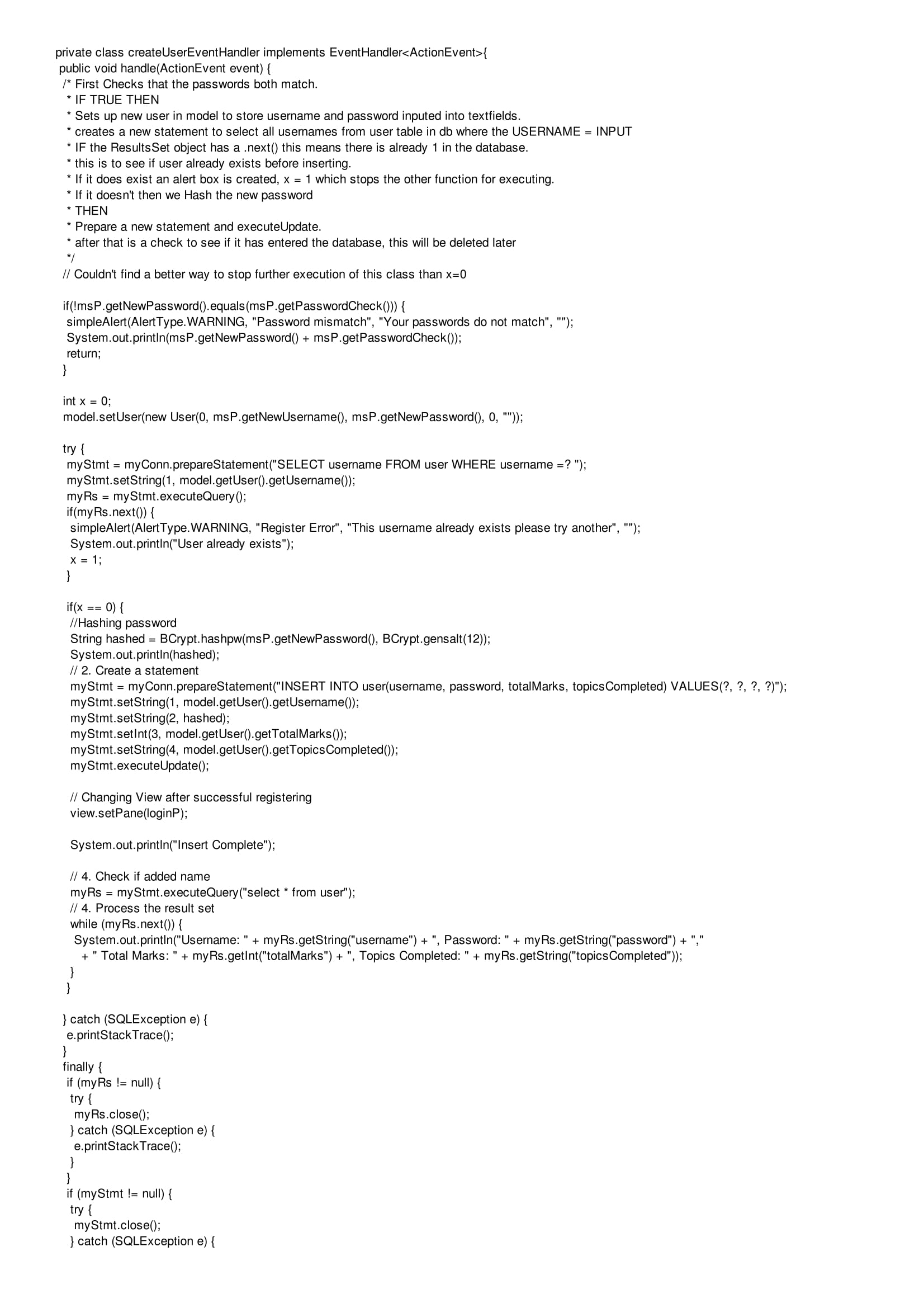


### 5



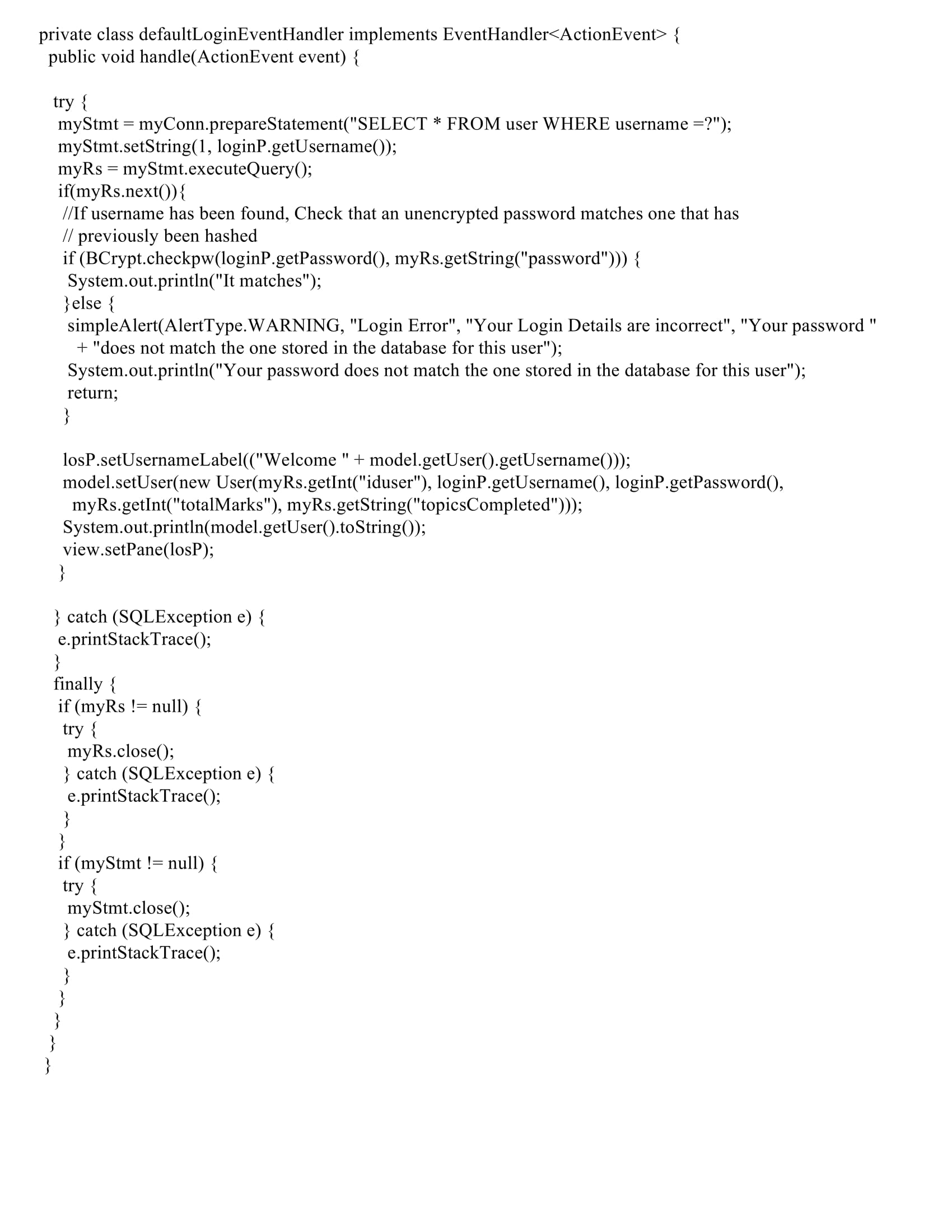
## createUserEventHandler code

[Back to Controller Coding](#_MainScreenPane)



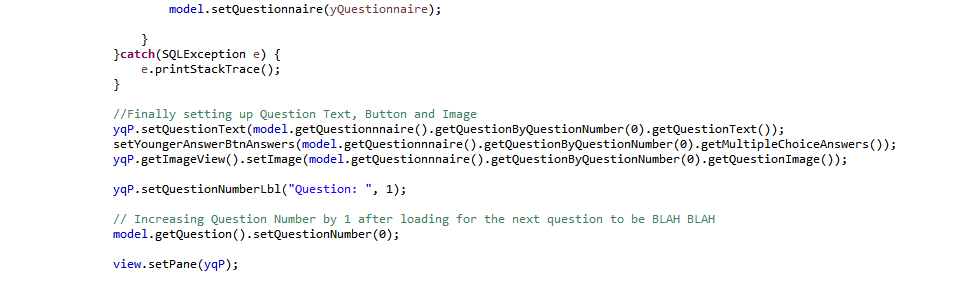


## loginEventHandlerCode

[Back to Controller Coding](#_LoginScreenPane)

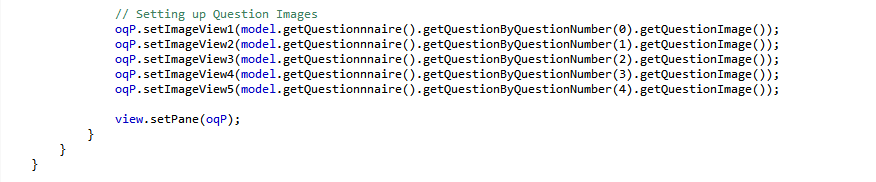
## LearningScreenPane TakeQuizBtn YoungerQuiz Code

[Back to controller development](#_LearningScreenPane_Coding_Progressi)

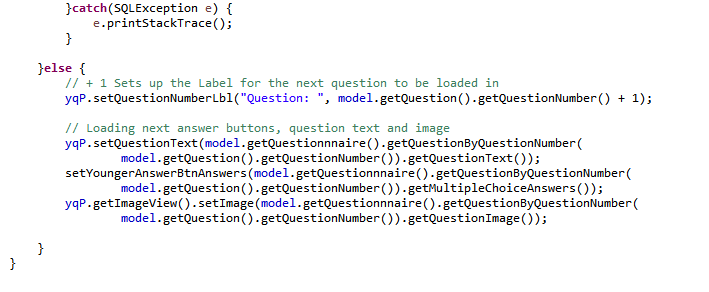
## LearningScreenPane TakeQuizBtn OlderQuiz Code

[Back to controller development](#_LearningScreenPane)

## YoungQuizScreen Marking Question Answer Code

[Back to Controller Development](#_YoungerQuizPane)

## OlderQuizScreen Marking Questionnaire Code

[Back to Controller Development](#_OlderQuizPane)



## JUnit Final Test Image

[Back to testing](#_Testing_Against_Use)

