

**Evidence Gathering Document for SQA Level 8 Professional Developer Award.**

This document is designed for you to present your screenshots and diagrams relevant to the PDA and to also give a short description of what you are showing to clarify understanding for the assessor.

Each point that required details the Assessment Criteria (What you have to show) along with a brief description of the kind of things you should be showing.

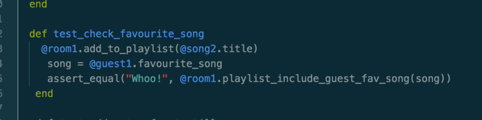
Please fill in each point with screenshot or diagram and description.

**Week 2**

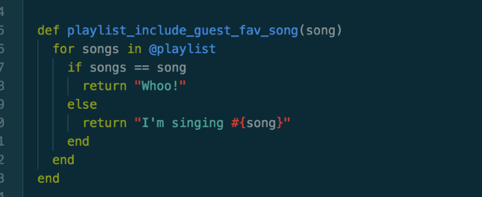
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.5 | Demonstrate the use of an array in a program. Take screenshots of:  \*An array in a program  \*A function that uses the array  \*The result of the function running | |



Here I have created a blank array for guestlist and playlist.



I then created a test which added a song to the playlist array.



I then created a method to make sure that the test worked. Here you can see the array being checked to see if one of the guests favourite songs is in it.

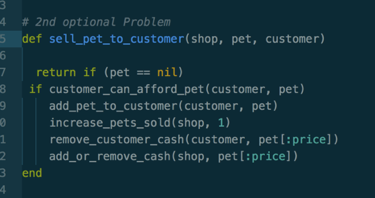


You can see the test successfully running.

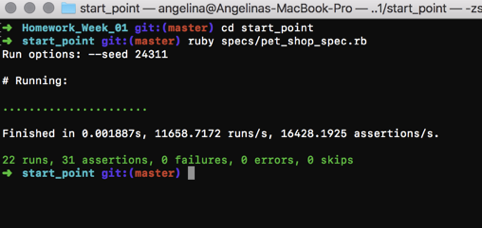
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.6 | Demonstrate the use of a hash in a program. Take screenshots of:  \*A hash in a program  \*A function that uses the hash  \*The result of the function running | |



In this screenshot you can see that a hash has been created for a ‘Pet Shop’.



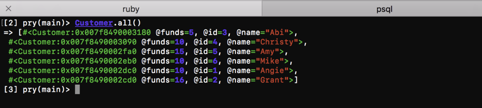
You can see that I am calling other methods inside this to see if the customer can afford a pet or not. It is then increasing the shops sold pets by one, taking the money out of the customers wallet and adding it to the shop till.



Here you can see that the test ran successfully.

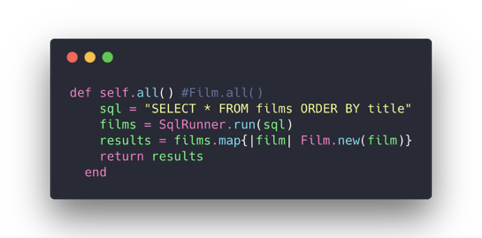
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.3 | Demonstrate searching data in a program. Take screenshots of:  \*Function that searches data  \*The result of the function running | |

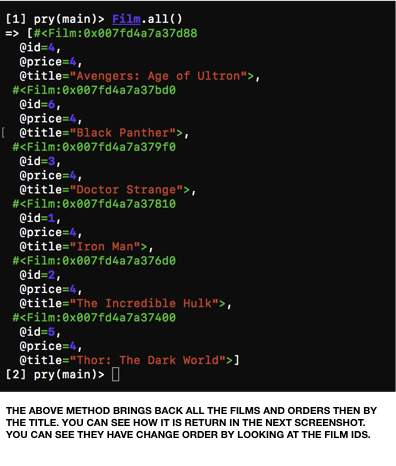




This is a method that selects all customers in the database and returns them. You can see the result in the next screenshot.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.4 | Demonstrate sorting data in a program. Take screenshots of:  \*Function that sorts data  \*The result of the function running | |

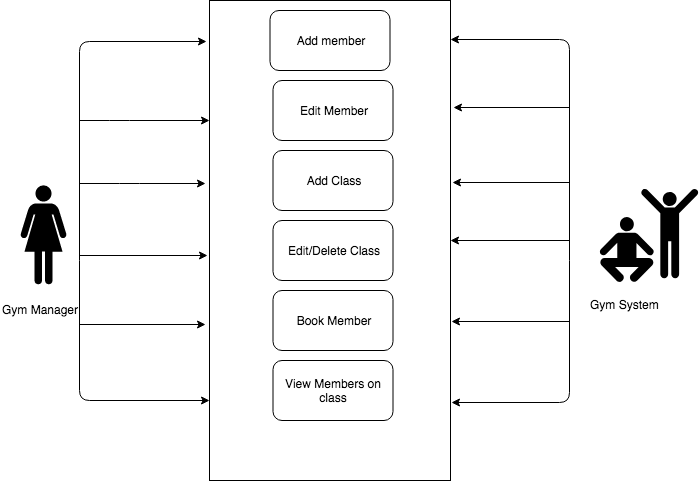


****

The above method brings back all the films and orders then by the title. You can see how it is return in the next screenshot. You can see they have change order by looking at the Film ids.

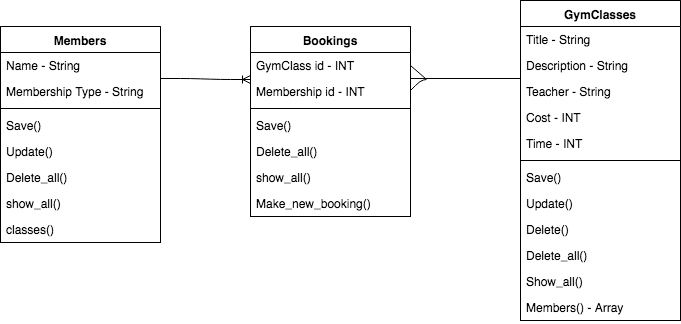
**Week 5**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **A&D** | A.D.1 | A Use Case Diagram | |



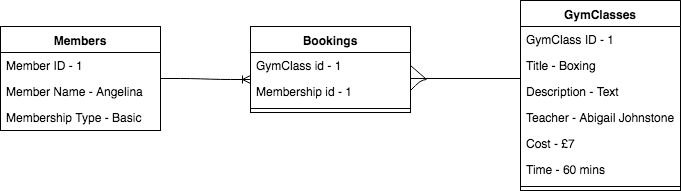
This shows that what the manger should be able to do.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **A&D** | A.D.2 | A Class Diagram | |



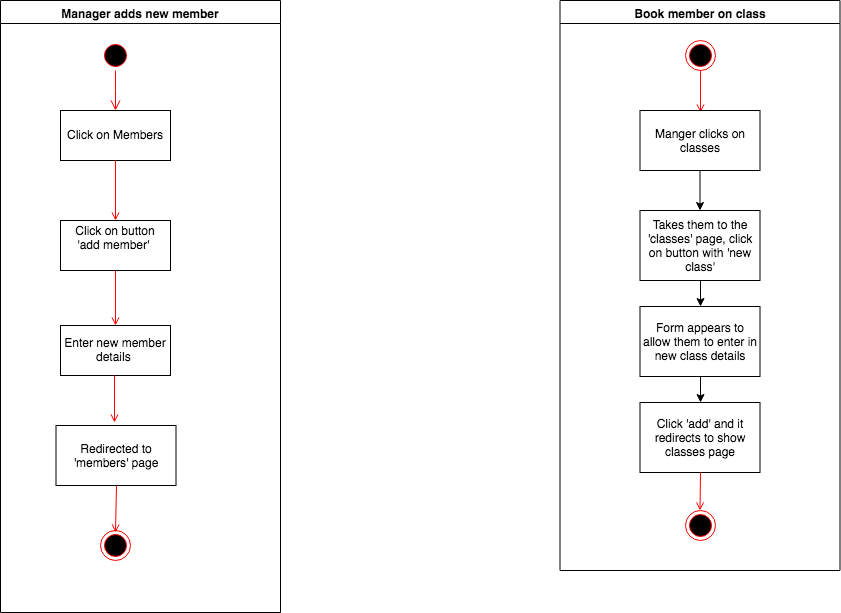
This shows the relationship between the three classes

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **A&D** | A.D.3 | An Object Diagram | |



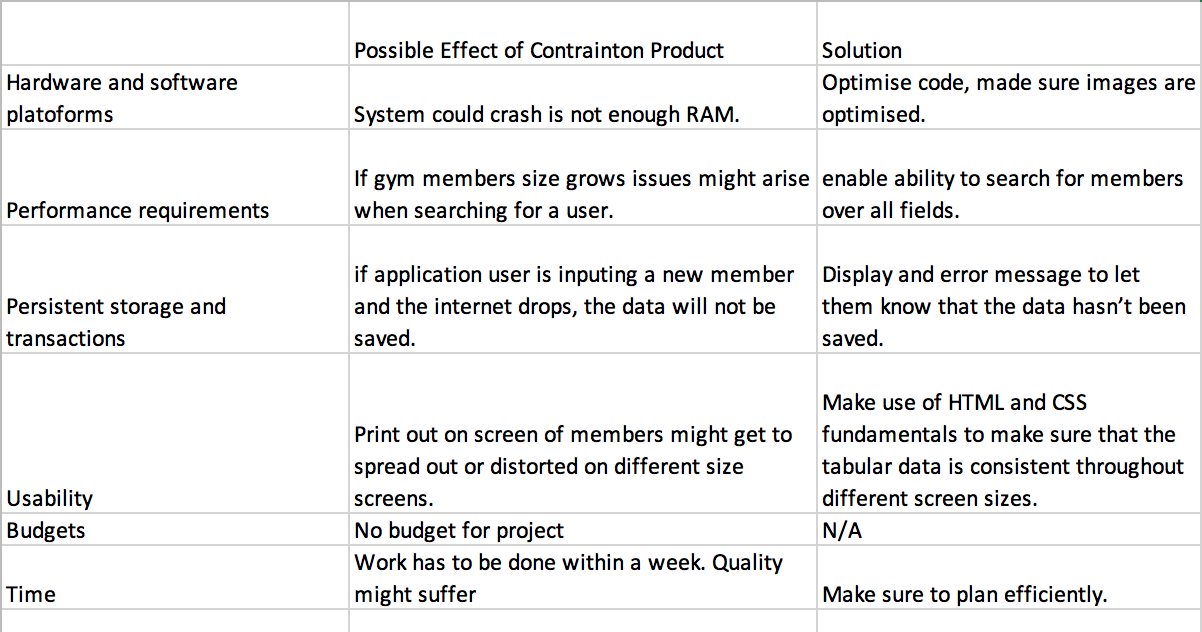
This shows what the information would be if entered

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **A&D** | A.D.4 | An Activity Diagram | |



Shows the flow of adding a new manager and booking a member onto a class.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **A&D** | A.D.6 | Produce an Implementations Constraints plan detailing the following factors:  \*Hardware and software platforms  \*Performance requirements  \*Persistent storage and transactions  \*Usability  \*Budgets  \*Time | |



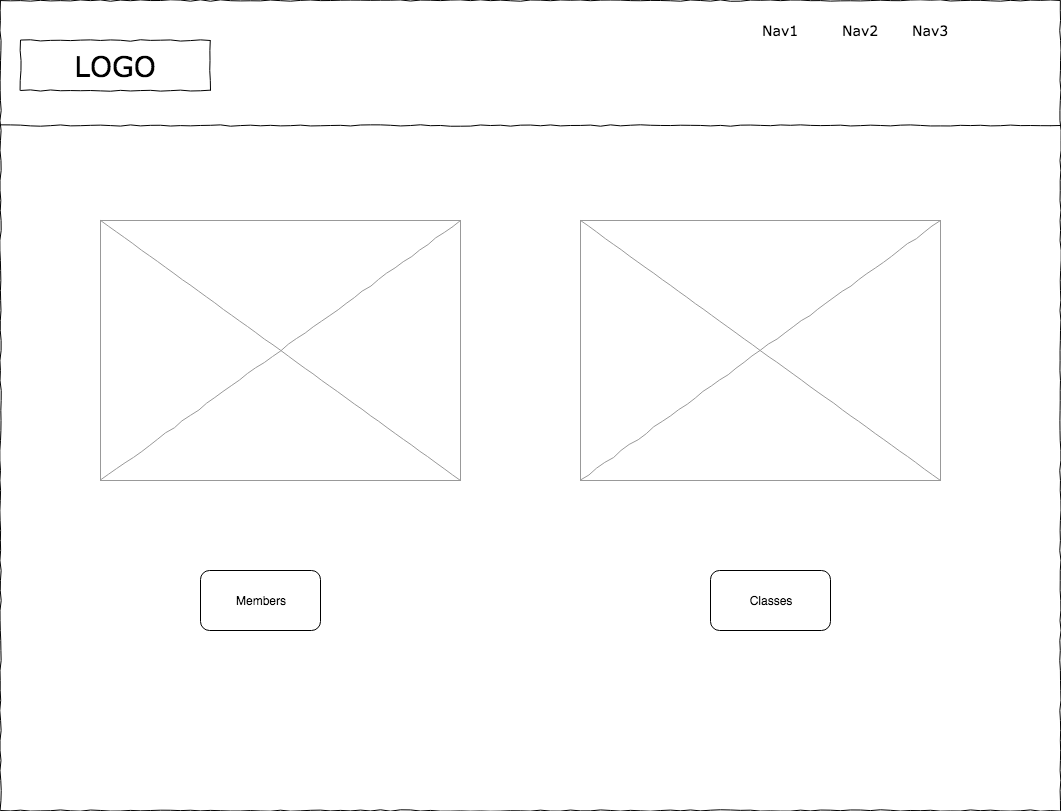
Some of the constraints do not matter on a project of this size.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.5 | User Site Map | |



Site Map for my Gym Website.

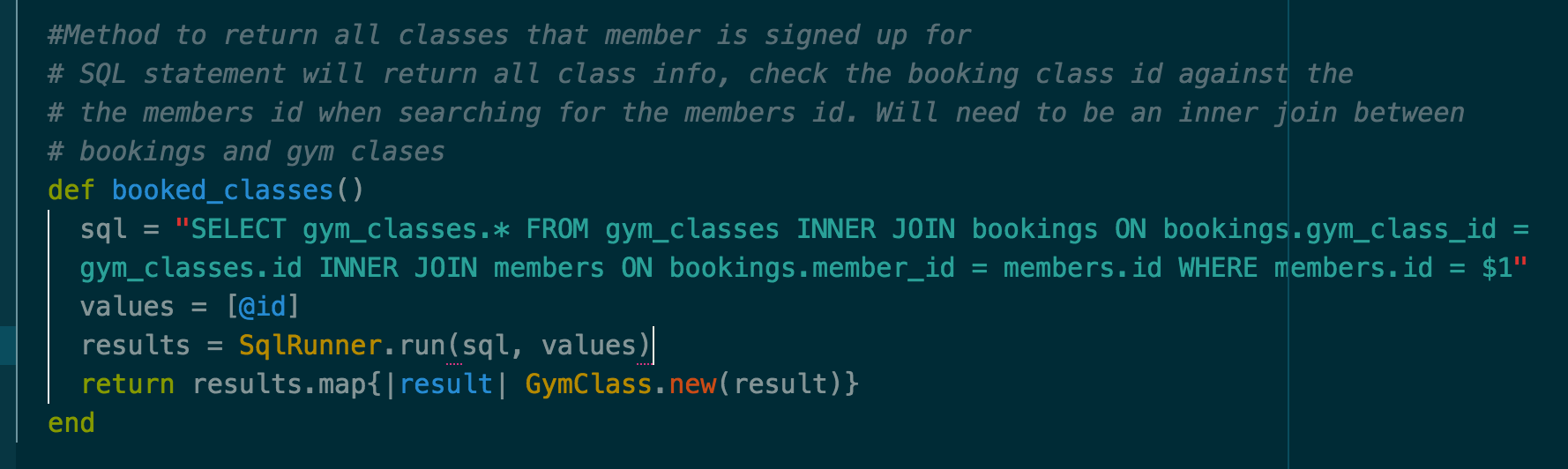
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.6 | 2 Wireframe Diagrams | |





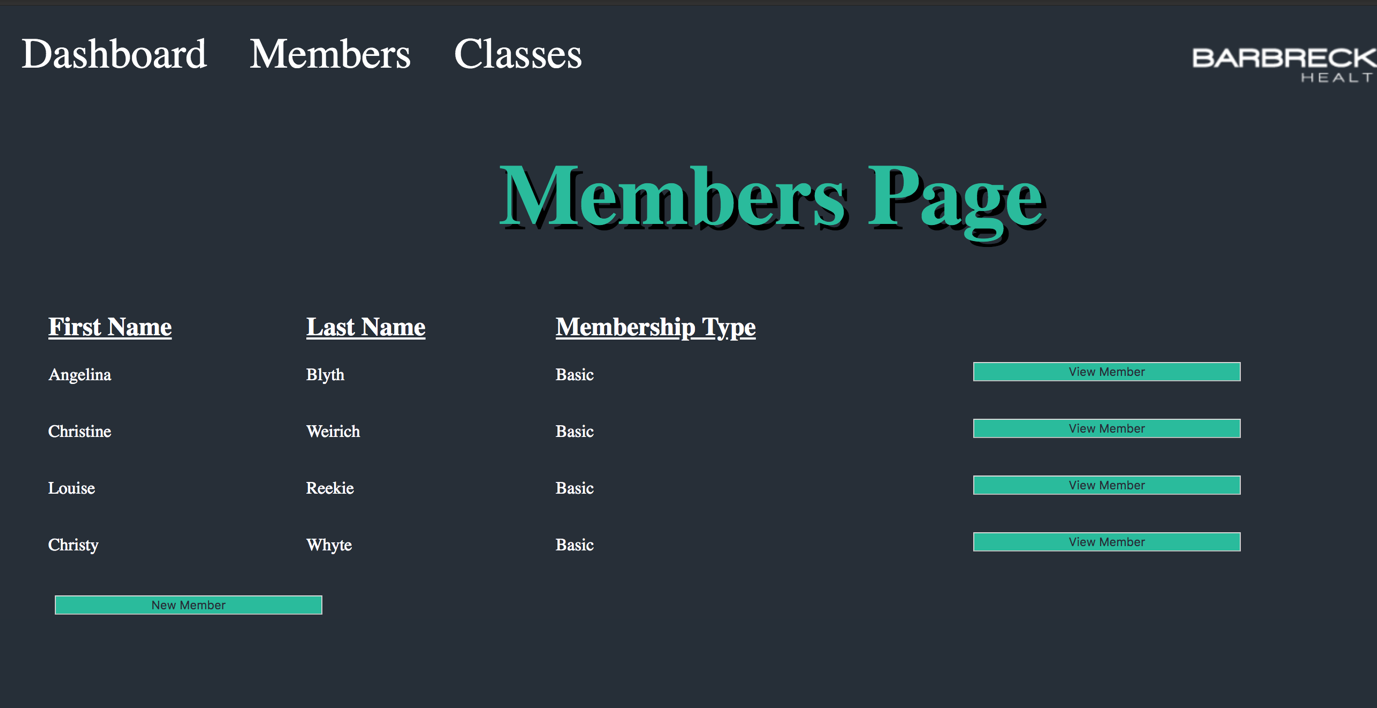
The first wireframe shows the websites dashboard, the second wireframe shows what you will see when you click on the class link.

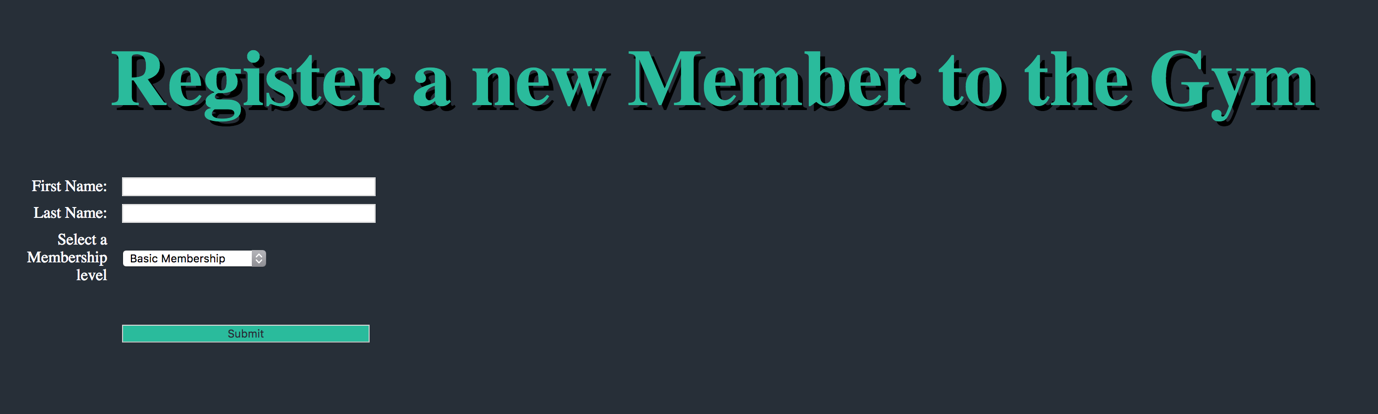
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.10 | Example of Pseudocode used for a method | |



Pseudocode for when I wanted to return all the classes that a member had signed up for.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.13 | Show user input being processed according to design requirements. Take a screenshot of:  \* The user inputting something into your program  \* The user input being saved or used in some way | |









The first screenshot shows the members list page, the user then clicks on the ‘New Member’ button.

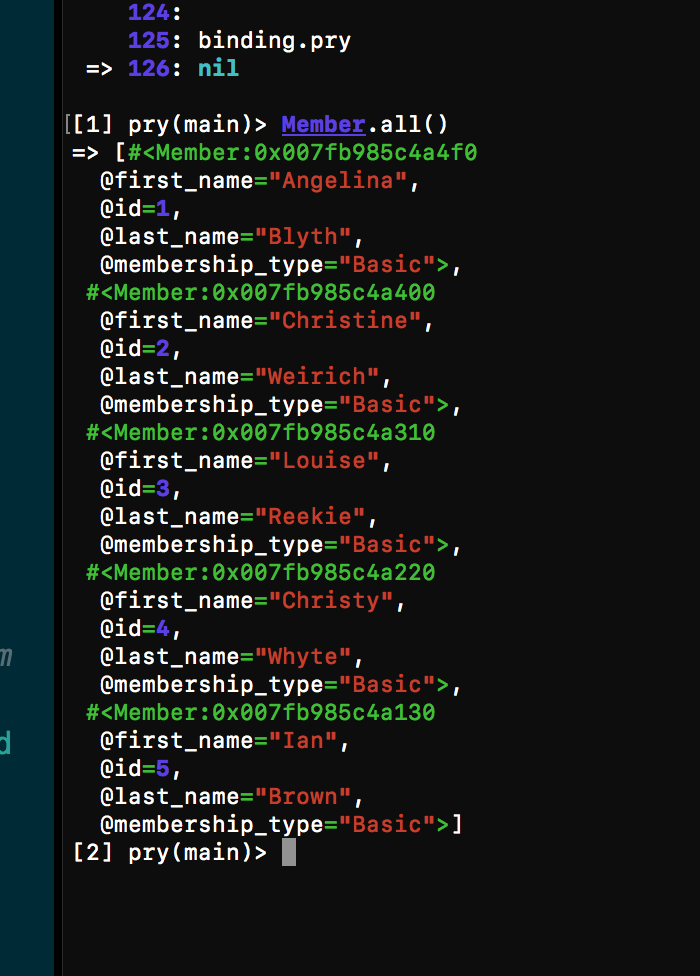
That takes them to the ‘add’ page.

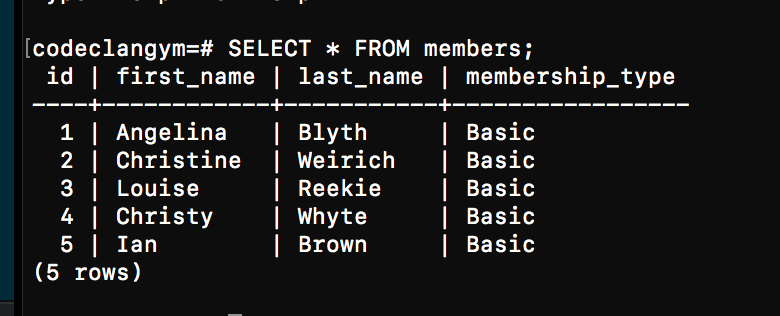
New members details are added in.

User clicks on ‘Submit’ and they are redirected to the members page.

You can see that the new member has been added to the bottom of the list.

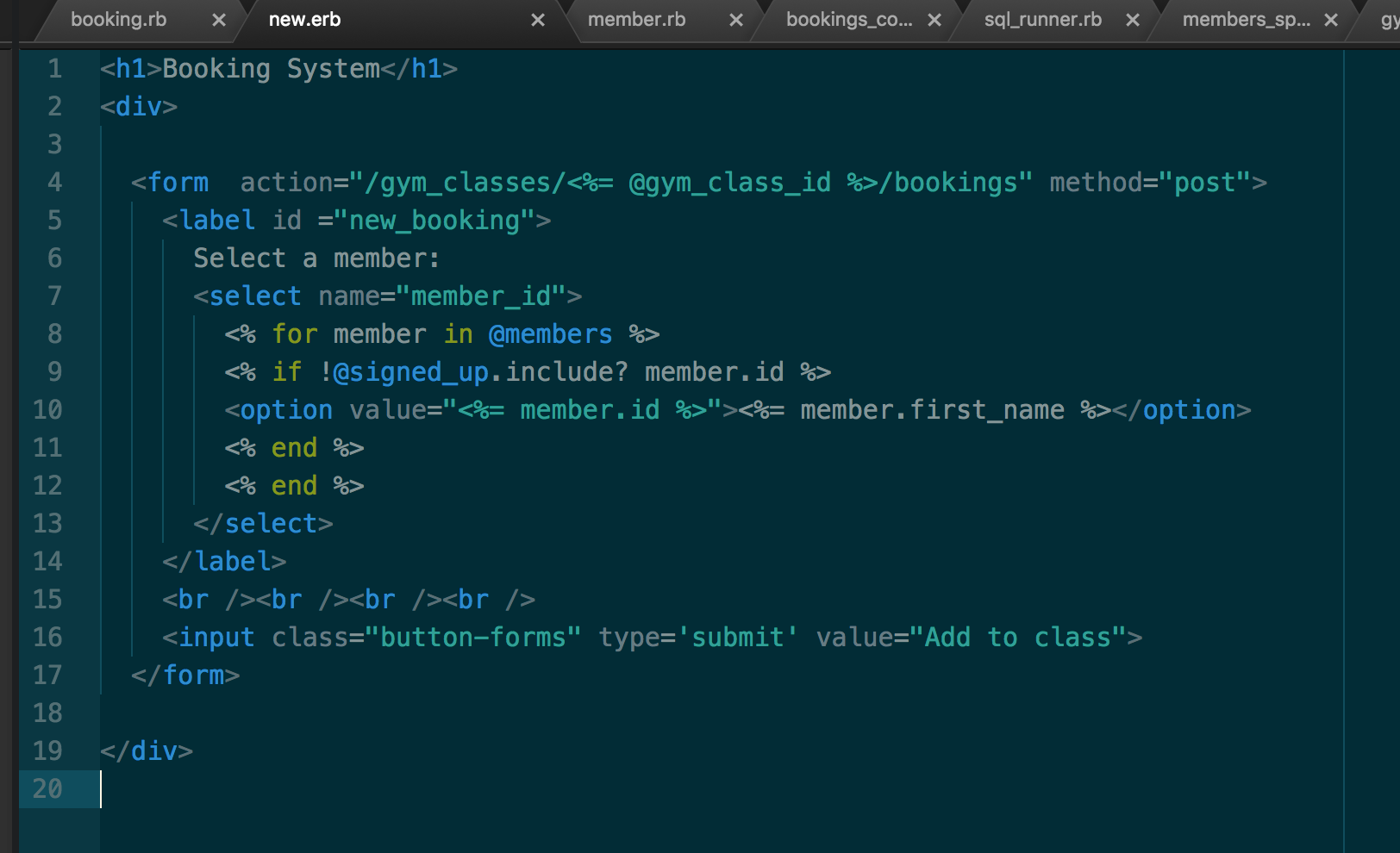
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.14 | Show an interaction with data persistence. Take a screenshot of:  \* Data being inputted into your program  \* Confirmation of the data being saved | |

****

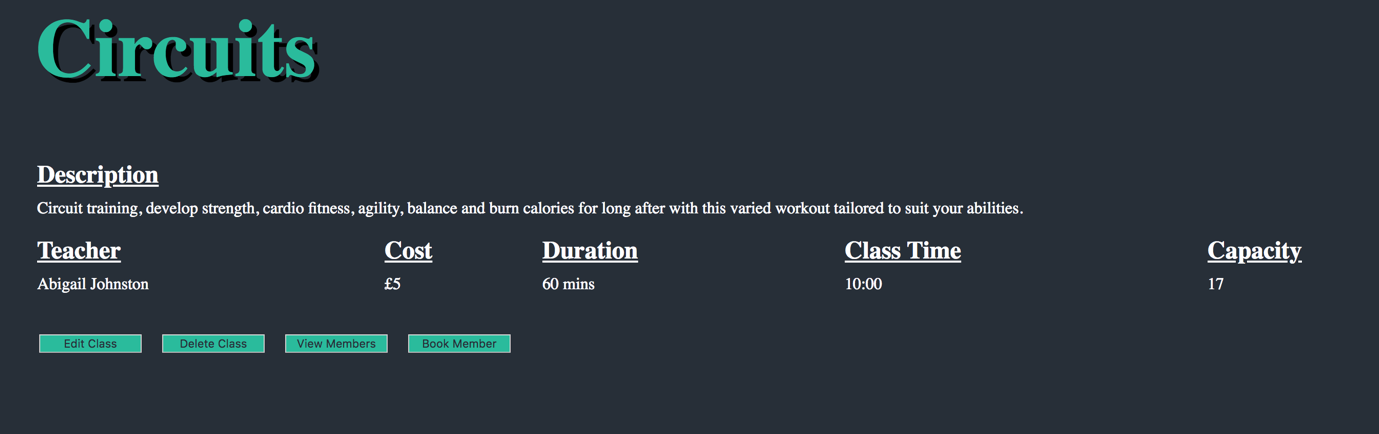
****

You can see in both screenshots that the user entered has been saved to the database, querying the database return the members table or calling the method to return all members returns the list.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.15 | Show the correct output of results and feedback to user. Take a screenshot of:  \* The user requesting information or an action to be performed  \* The user request being processed correctly and demonstrated in the program | |

****

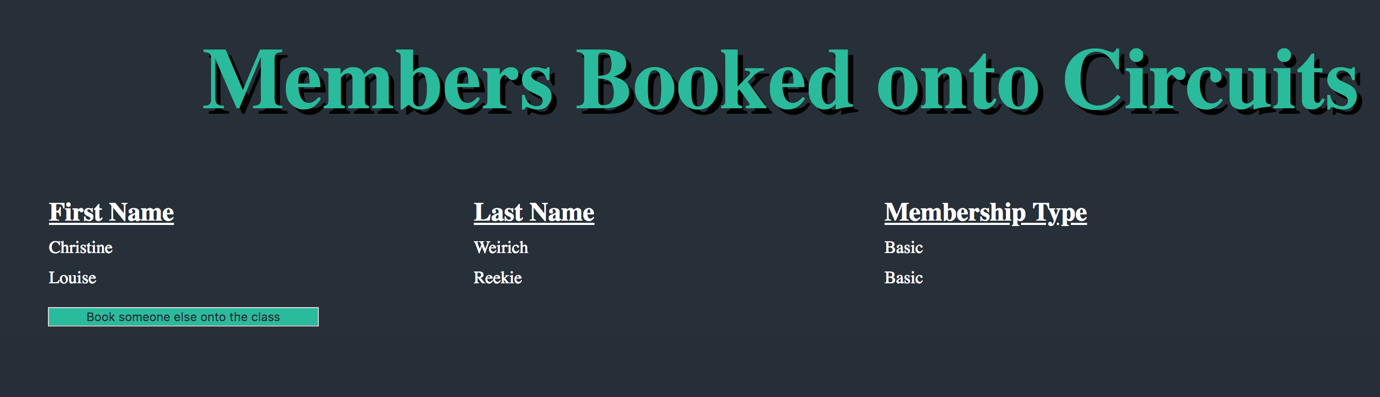
This is showing how a manager can add a member into a gym class and produce a new booking for that member/class.

****

Display on the gym class page, gives you the option to ‘Book Member’.



Simple dropdown list of members currently not booked onto the system. Part of the code in the first screenshot limits the return of members in the drop down so a member is not able to be doubled booked onto a class.

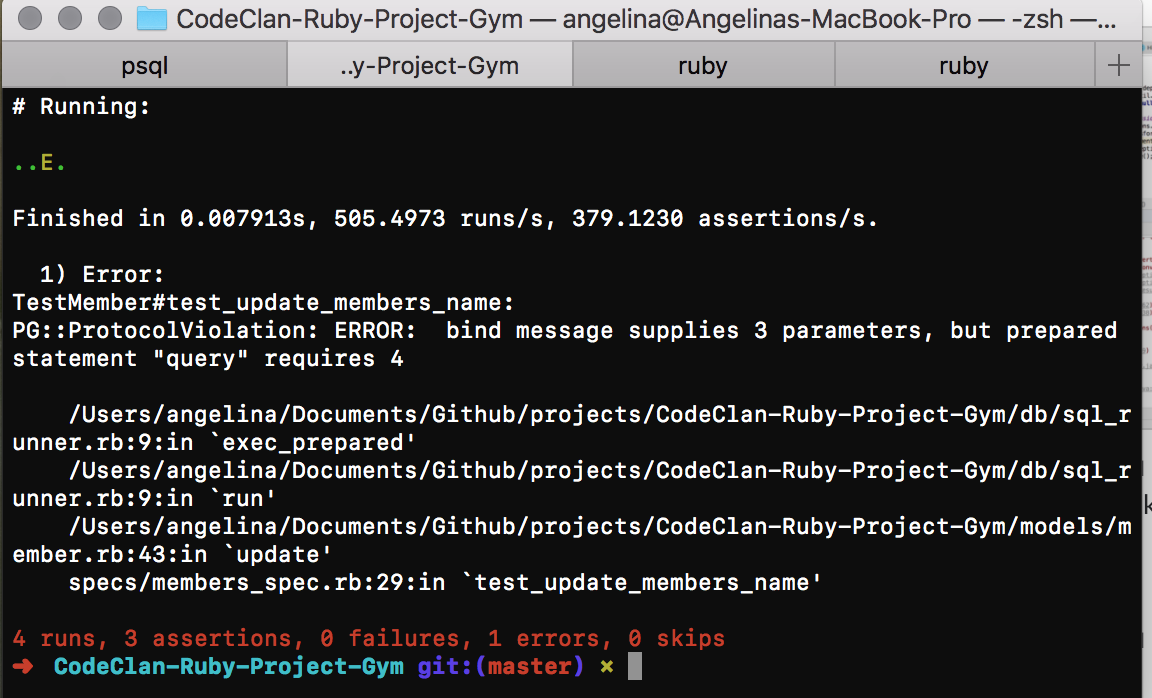


Picking the member named “Louise” you can see she now is booked in for the gym class.

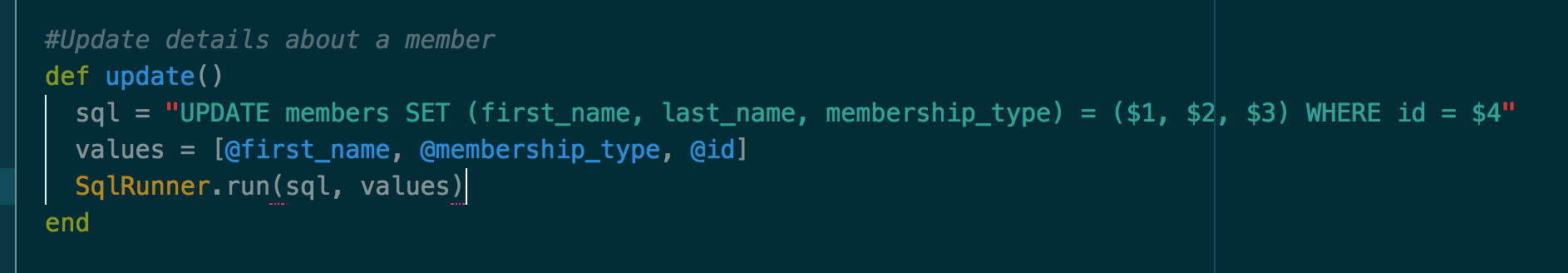
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.18 | Demonstrate testing in your program. Take screenshots of:  \* Example of test code  \* The test code failing to pass  \* Example of the test code once errors have been corrected  \* The test code passing | |

****

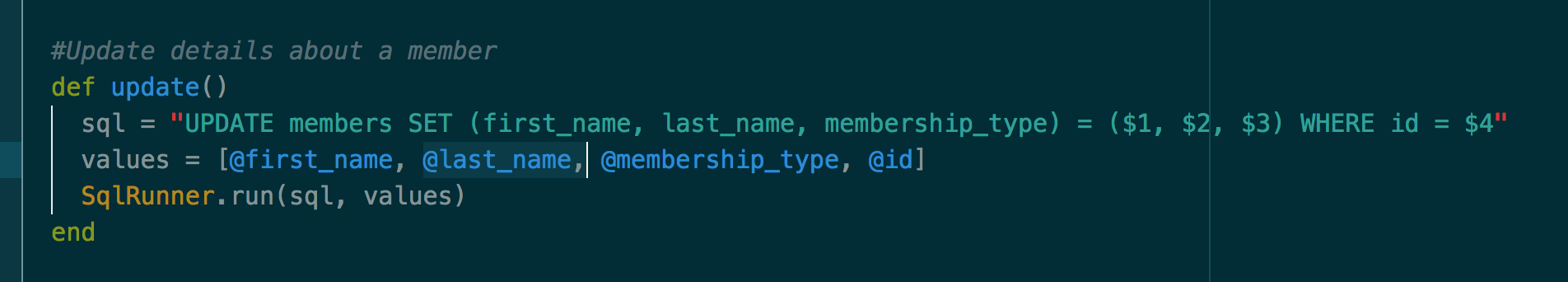
Test written to update a members first name.

****

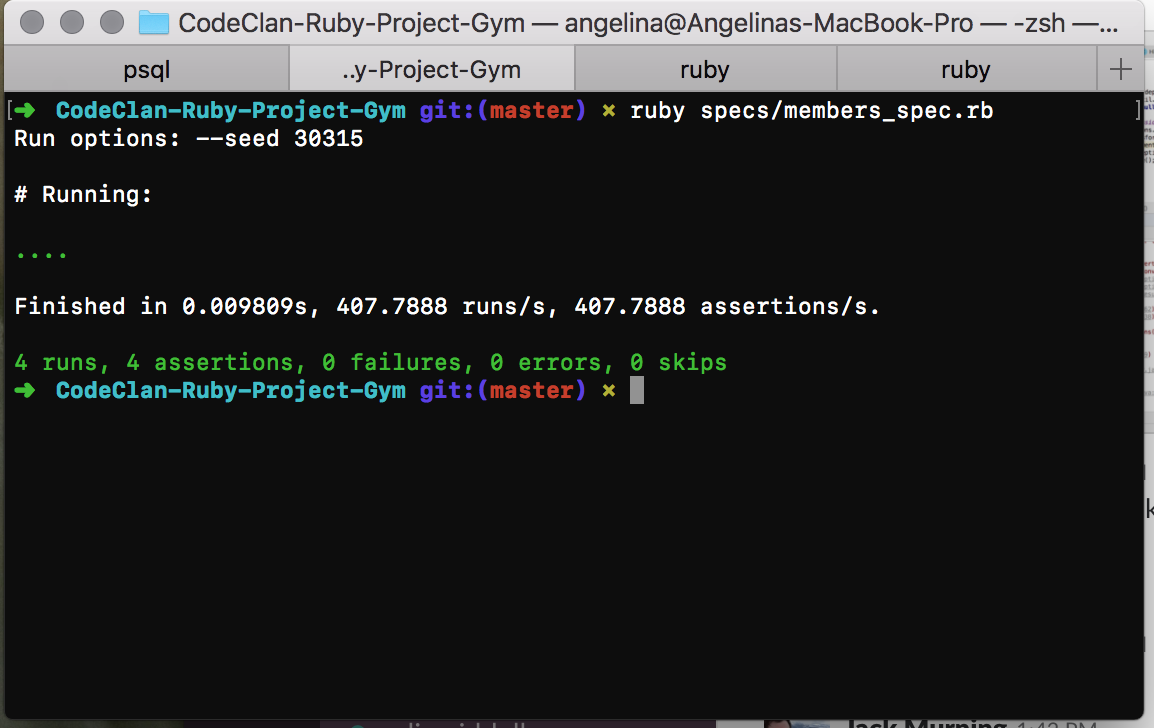
Test is failing as the ‘Update’ method isn’t written correctly.



Update method that has been written that is incorrect.



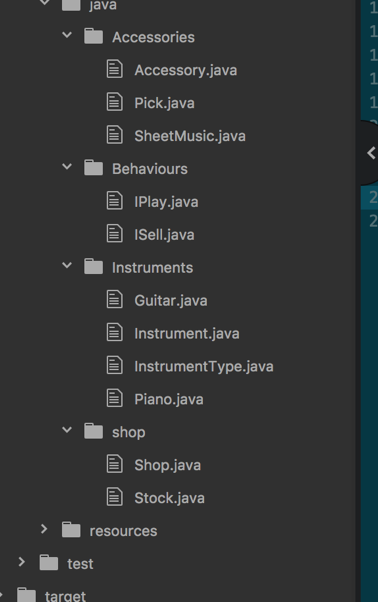
Update method now written correctly.



Test is now passing due to the method being correct.

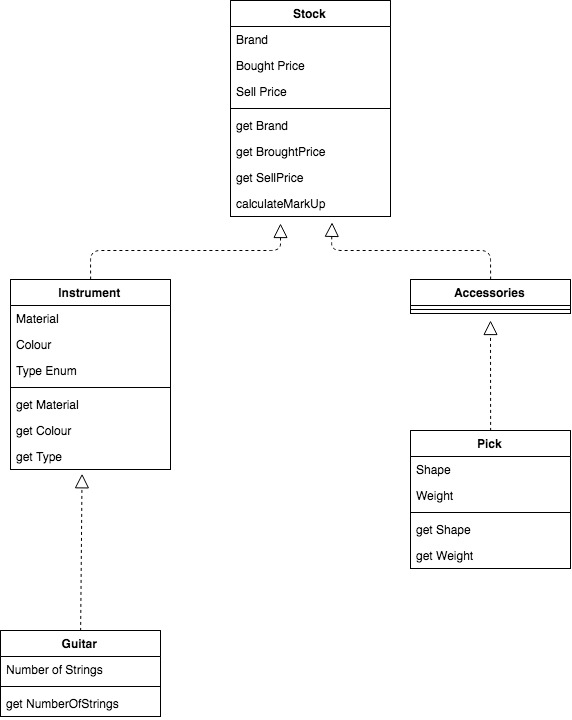
**Week 7**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.7 | The use of Polymorphism in a program and what it is doing. | |

****

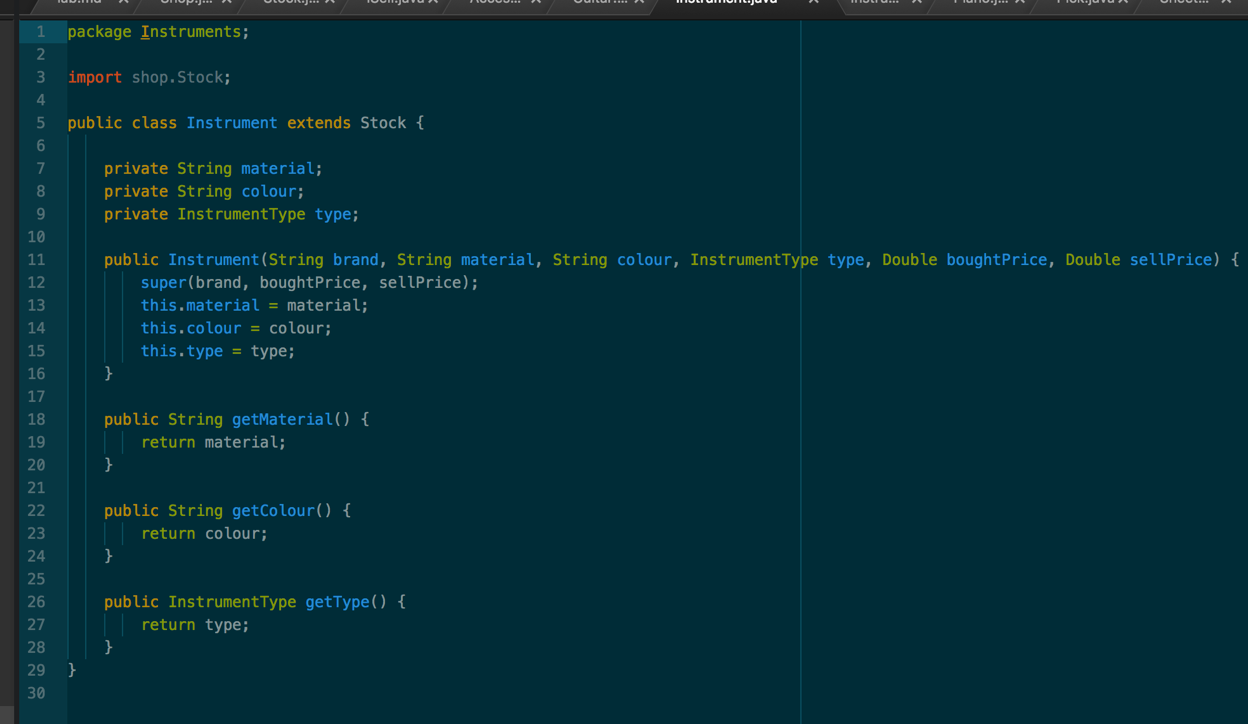
All stock items implement’s the ISell interface. This allows the shop to see projected profit from selling the accessories or the instruments.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **A&D** | A.D.5 | An Inheritance Diagram | |

****

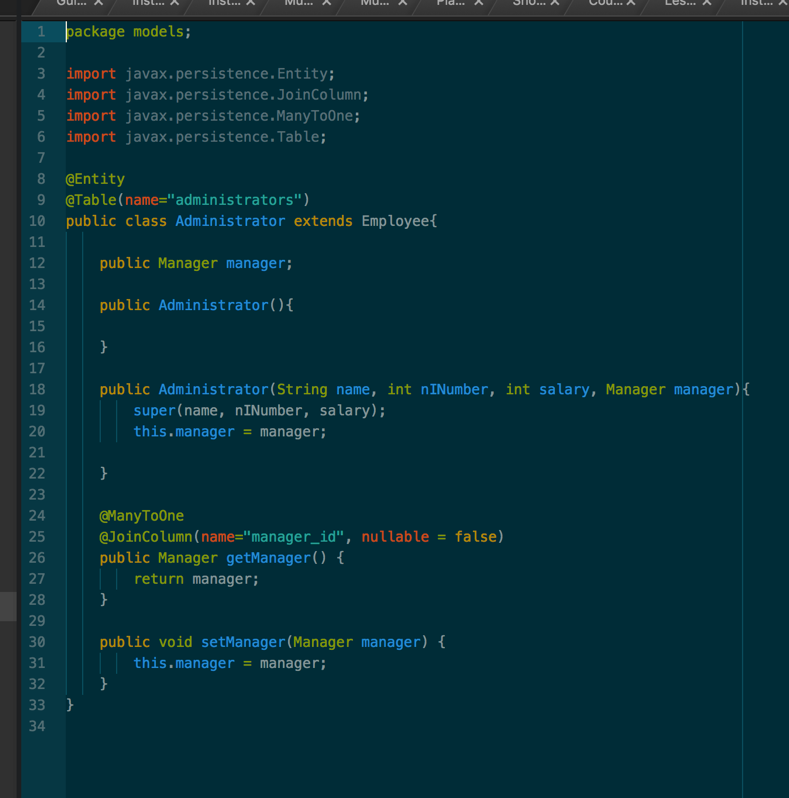
Above shows a guitar and a pick all inheriting from the Stock class, this filters down to Instruments and Accessories.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.1 | The use of Encapsulation in a program and what it is doing. | |

****

The screenshot shows encapsulation within the Instruments class. The class has private properties which can only be accessed by the public getters & setters.

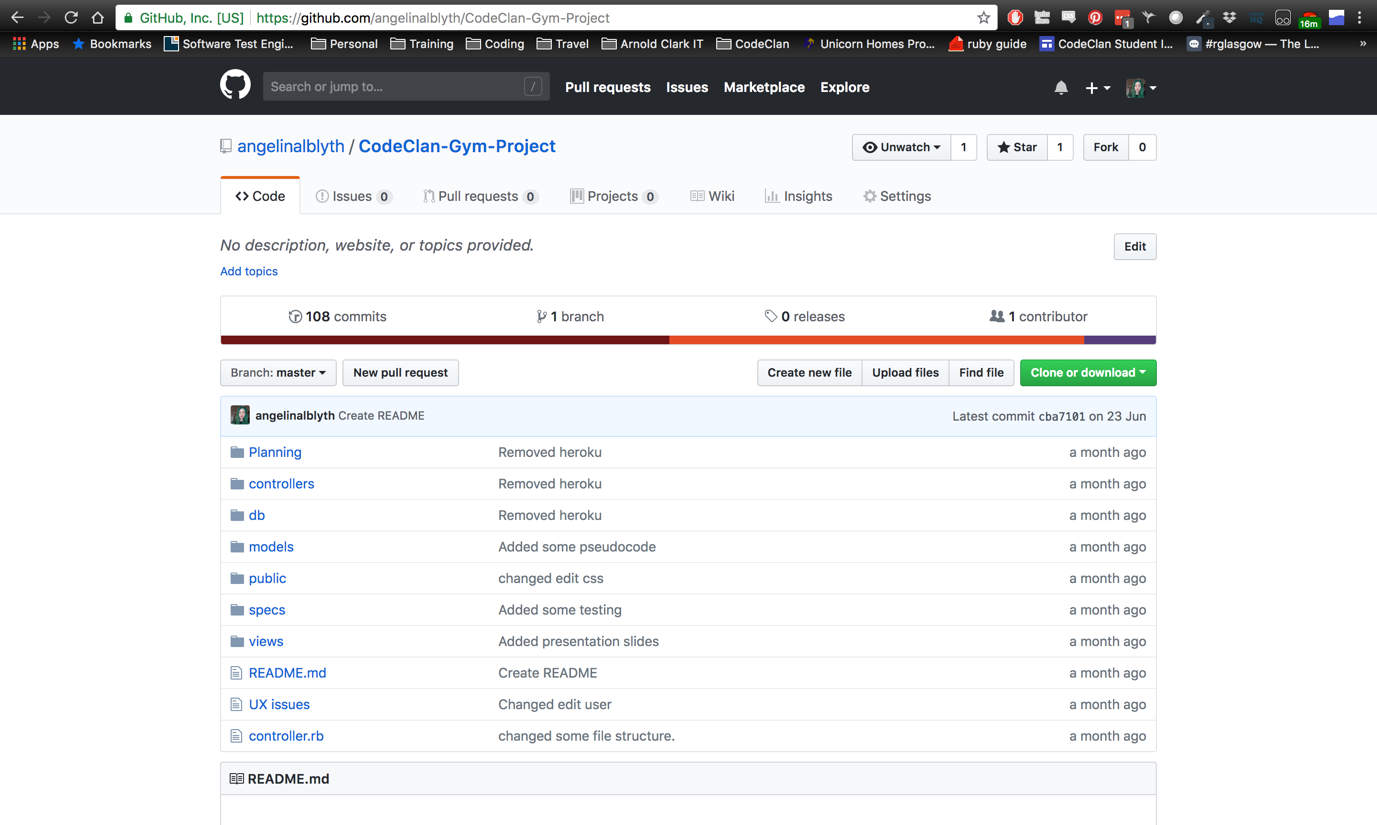
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.2 | Take a screenshot of the use of Inheritance in a program. Take screenshots of:  \*A Class  \*A Class that inherits from the previous class  \*An Object in the inherited class  \*A Method that uses the information inherited from another class. | |

****

Employee here is the superclass. The Administrator class then inherits all the attributes from the Employee class making it a child.

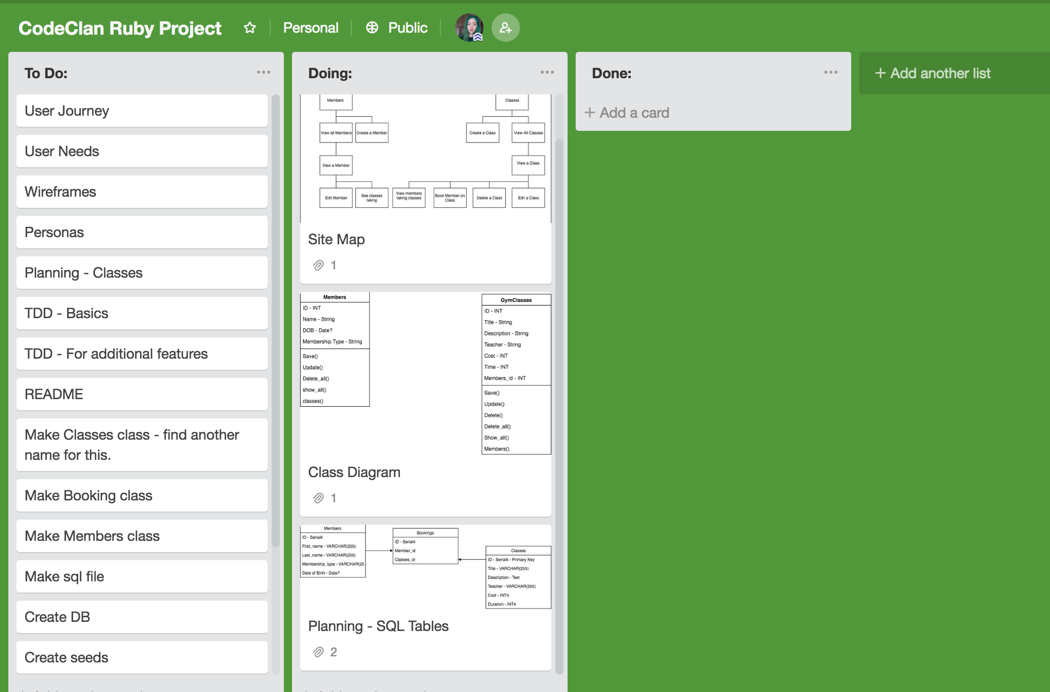
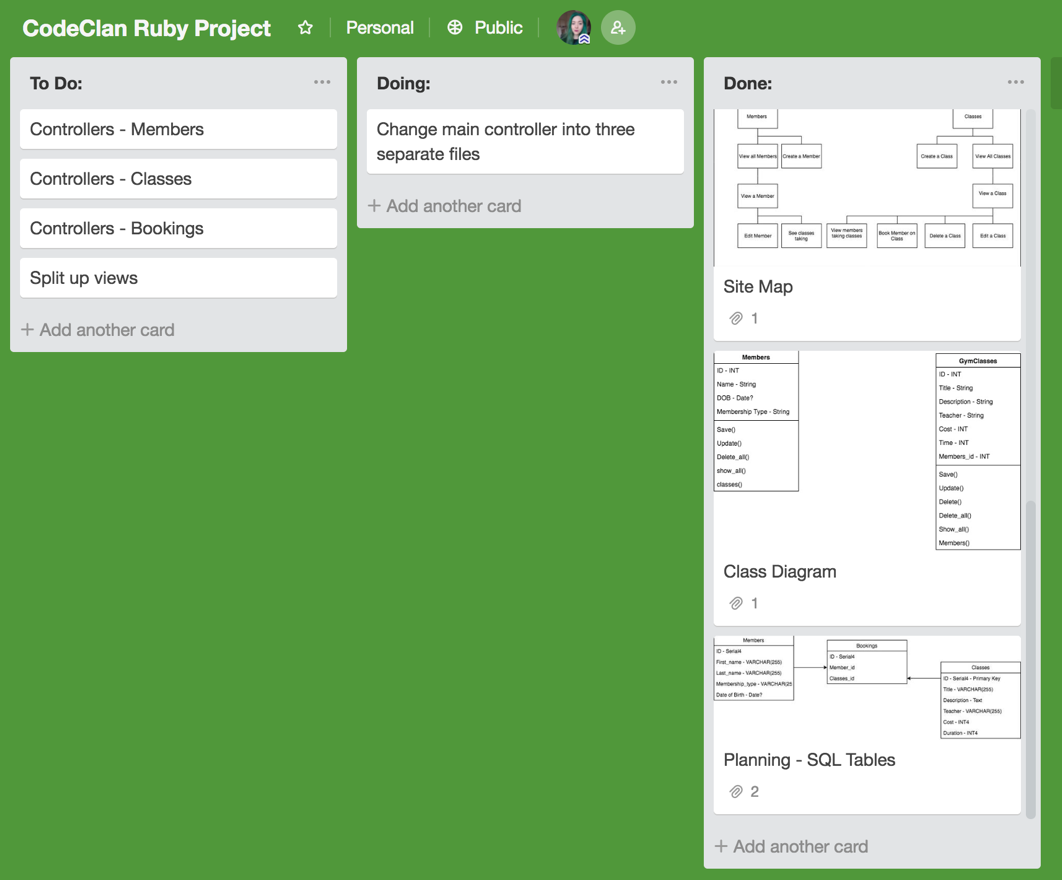
**Week 10**

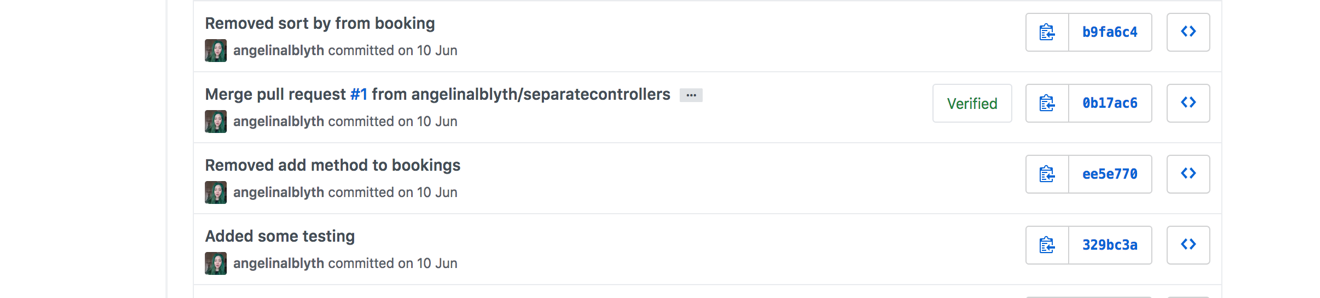
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.11 | Take a screenshot of one of your projects where you have worked alone and attach the Github link. | |

****

<https://github.com/angelinalblyth/CodeClan-Gym-Project>

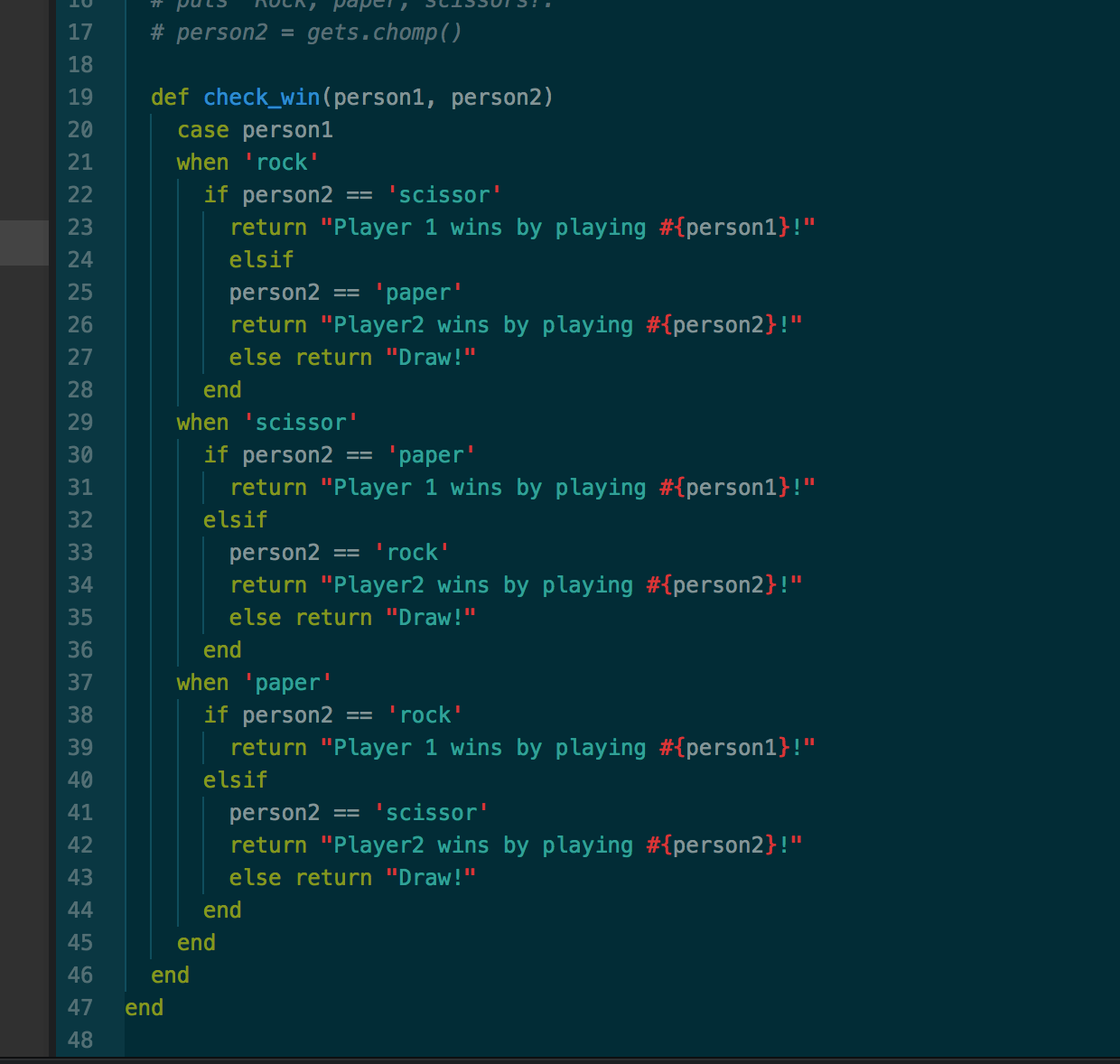
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.12 | Take screenshots or photos of your planning and the different stages of development to show changes. | |

****

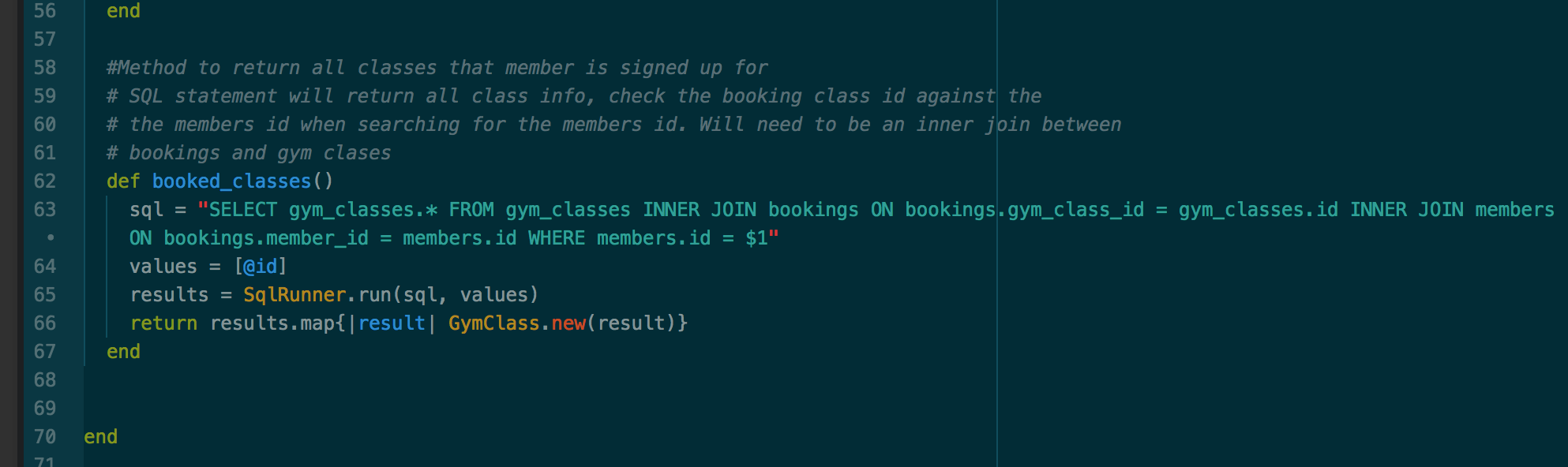
****

At first my project was going to be run from the one controller as I wasn’t too confident in doing separate controllers for each of the classes. Once I had the main controller working I then branch off in GitHub and spent time implementing the different controllers. After a few tweaks it worked and I then merged into master bringing those separated files into the master branch.

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.9 | Select two algorithms you have written (NOT the group project). Take a screenshot of each and write a short statement on why you have chosen to use those algorithms. | |

****

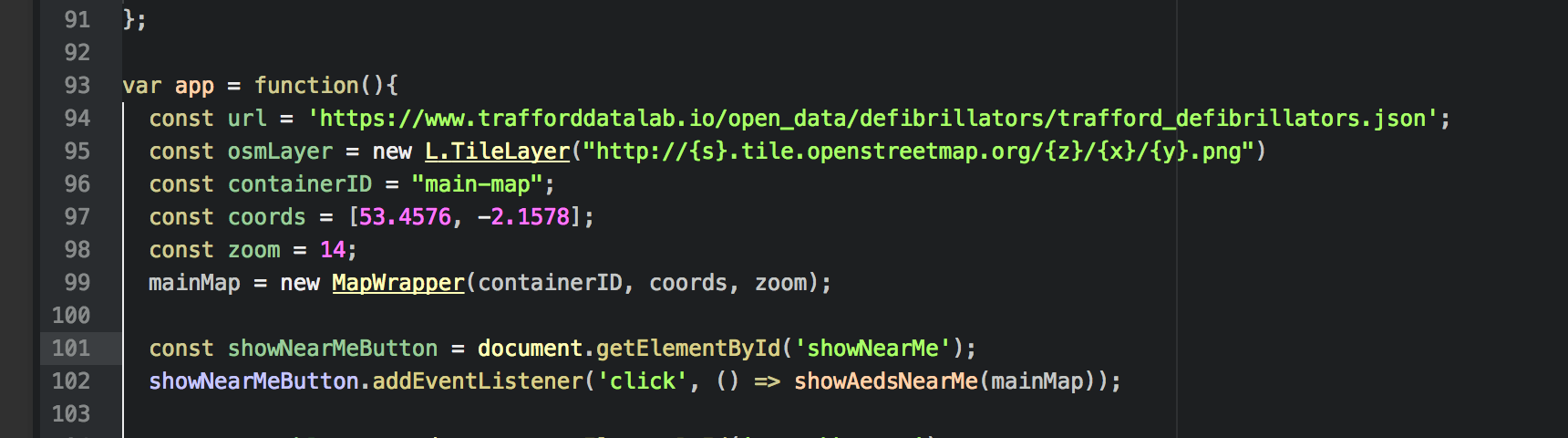
This is from the Rock, Paper, Scissors game that I created. The method checks what the first player has selected, goes to that if statements and then checks to see what the second person has selected printing out who has one, either Player 1 or Player 2.



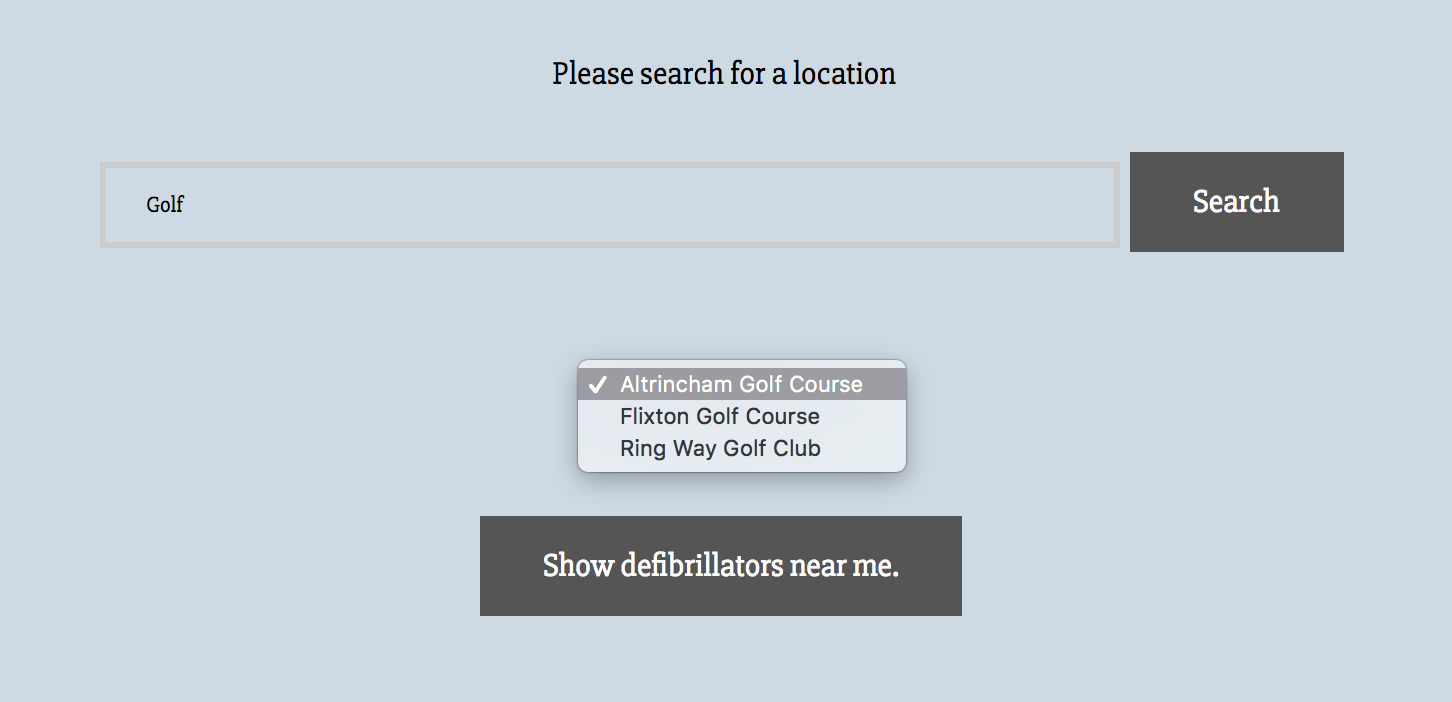
This method is part of my Ruby Project, it returns all the classes that a particular member is signed up for. This helped when displaying a dropdown menu within the HTML file to only display members names that weren’t already signed up for a particular gym class.

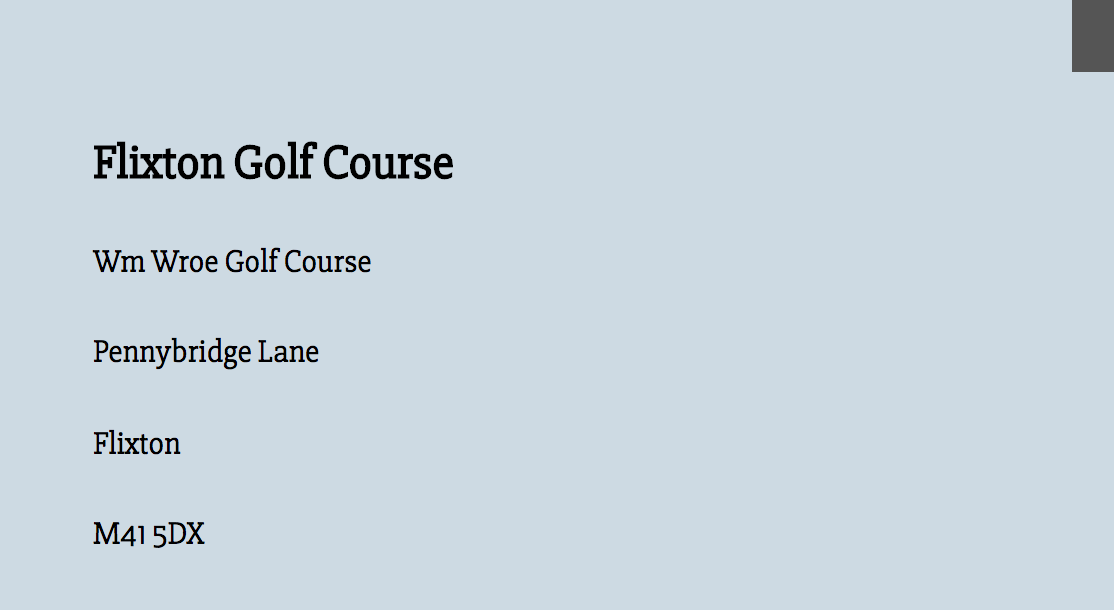
**Week 12**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.16 | Show an API being used within your program. Take a screenshot of:  \* The code that uses or implements the API  \* The API being used by the program whilst running | |

****

On line 94 you can see that I have brought in an api to be used within my app.





Further two screenshots show that when searching for a location, anything matching the filter returns a list of possible locations, when selected it then pulls back the details of the location. This is all from the API.

**Week 15**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.1 | Take a screenshot of the contributor’s page on Github from your group project to show the team you worked with. | |

**Paste Screenshot here**

**Description here**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.2 | Take a screenshot of the project brief from your group project. | |

**Paste Screenshot here**

**Description here**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.3 | Provide a screenshot of the planning you completed during your group project, e.g. Trello MOSCOW board. | |

**Paste Screenshot here**

**Description here**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.4 | Write an acceptance criteria and test plan. | |

**Paste Screenshot here**

**Description here**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.7 | Produce two system interaction diagrams (sequence and/or collaboration diagrams). | |

**Paste Screenshot here**

**Description here**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.8 | Produce two object diagrams. | |

**Paste Screenshot here**

**Description here**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.17 | Produce a bug tracking report | |

**Paste Screenshot here**

**Description here**