**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.

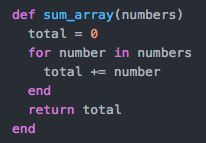
Each point that required details the Assessment Criteria (What you have to show) along with a brief description of the kind of things you should be showing.

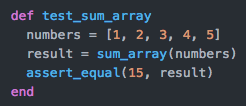
Please fill in each point with screenshot or diagram and description of what you are showing.

**David Telfer**

**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 | |
|  |  | **Description:** | |

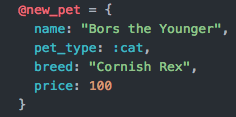
****

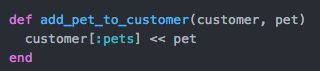
****

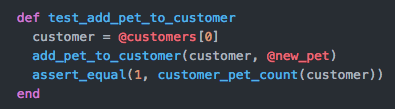
**array%20(terminal).png**

The first screenshot shows a function which is designed to loop through an array and find the sum of the numbers within that array. The second screenshot is to test that this function works – the array contains five numbers which should add up to 15. The 3rd screenshot is this test passing.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | |
|  |  | **Description:** | |

****

****

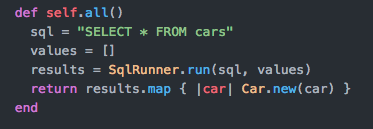
****

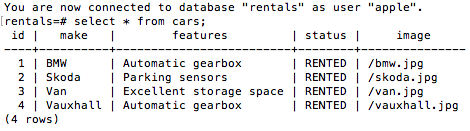
**pet%20(terminal).png**

The first image shows a hash with a new pet (which is a cat). The second image is a function which takes in a customer and pet then adds this pet to that customer. The third image is a test which shows that the customer now has 1 pet and the final image is the result of this test running and passing in terminal.

**Week 3**

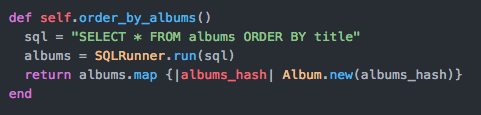
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **I&T** | I.T.3 | Demonstrate searching data in a program. Take screenshots of:  \*Function that searches data  \*The result of the function running | |
|  |  | **Description:** | |

****

****

The first image is the function that searches for everything in the cars table. The second screenshot is the result of that search…it includes all the car data within the database.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | |
|  |  | **Description:** | |

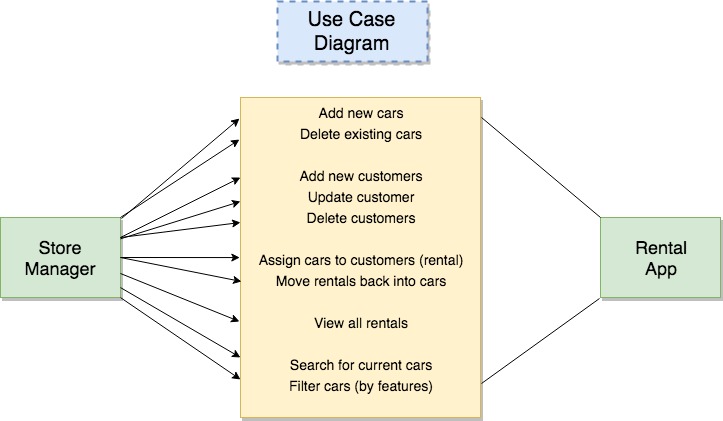
****

**sorting%20(result).png**

The first image shows a function that selects all of the albums in a table and sorts them alphabetically by their title. The second image shows this function in action. Graduation (id2) is listed before Recovery (id1).

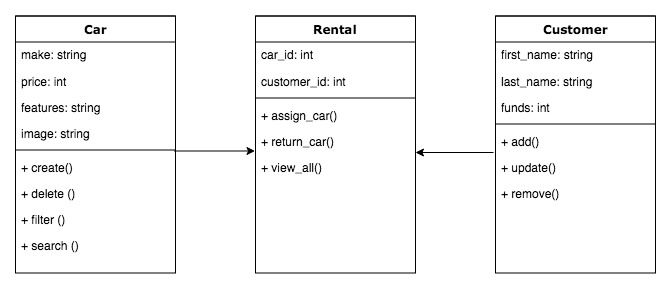
**Week 5 and 6**

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

****

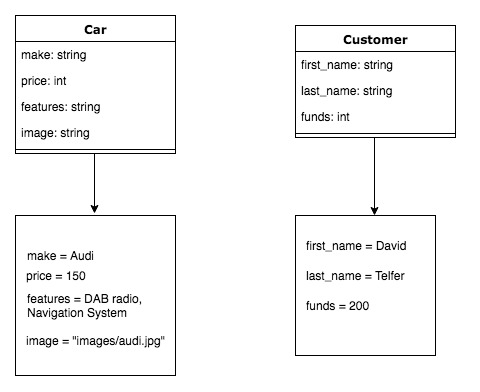
This use case diagram shows the actor (in this case Store Manager) and how they can interact with the Rental App. The middle table includes the functions that a Store Manager is able to carry out within the app.

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

****

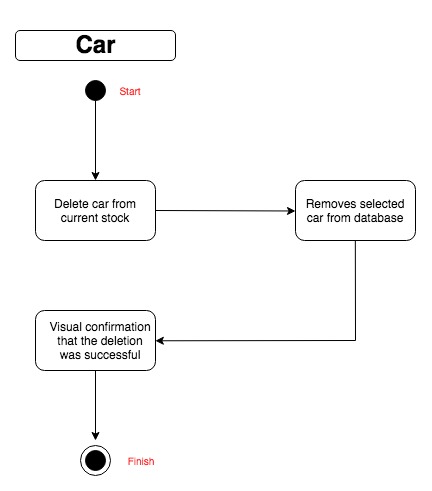
The diagram shows three different classes (Car, Rental and Customer). The top half of each section shows the attributes for each class and their types (string / integer). The bottom half gives examples of the functions that can be performed in each class.

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

****

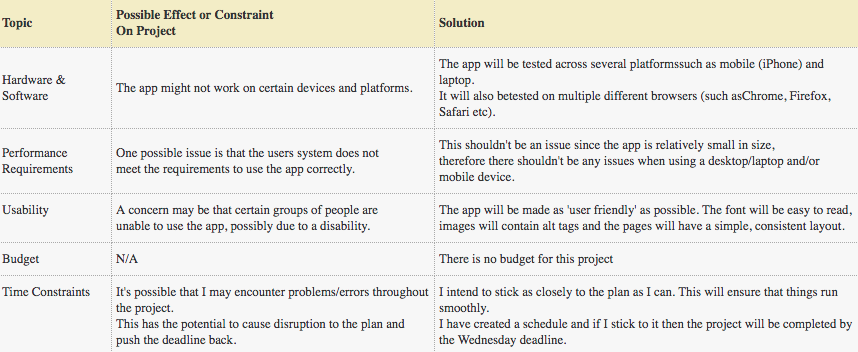
The diagram above shows two classes (Car and Customer) and each of them has an object. These both show examples of the data structure of each object.

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

****

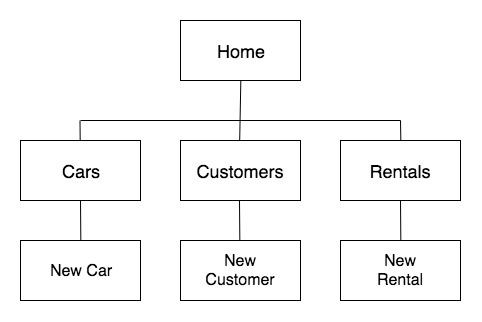
This activity diagram shows one of the functions in the app from start to finish. As you can see when you delete a car from the stock, it removes that instance of the car from the database and displays the user with a confirmation message that this action has worked.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | |
|  |  | **Description:** | |

****

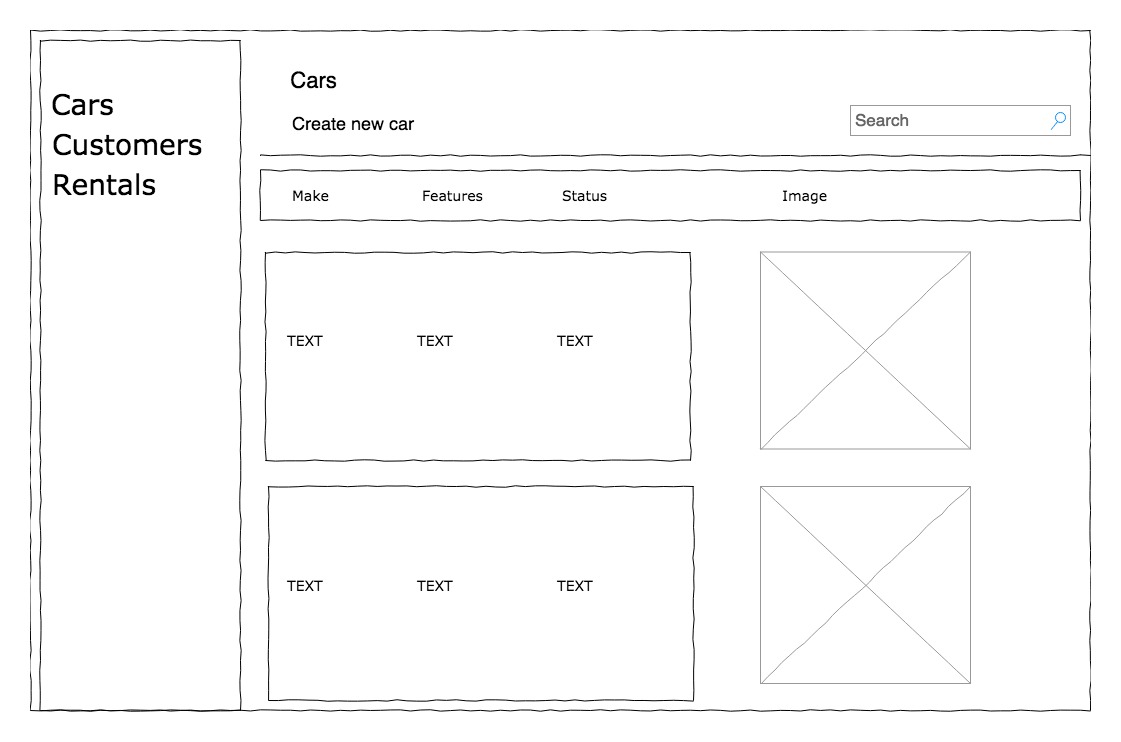
The constraints plan allows you to plan ahead for any possible issues that may arise. By doing so it means that you are prepared in the event that something doesn’t go to plan.

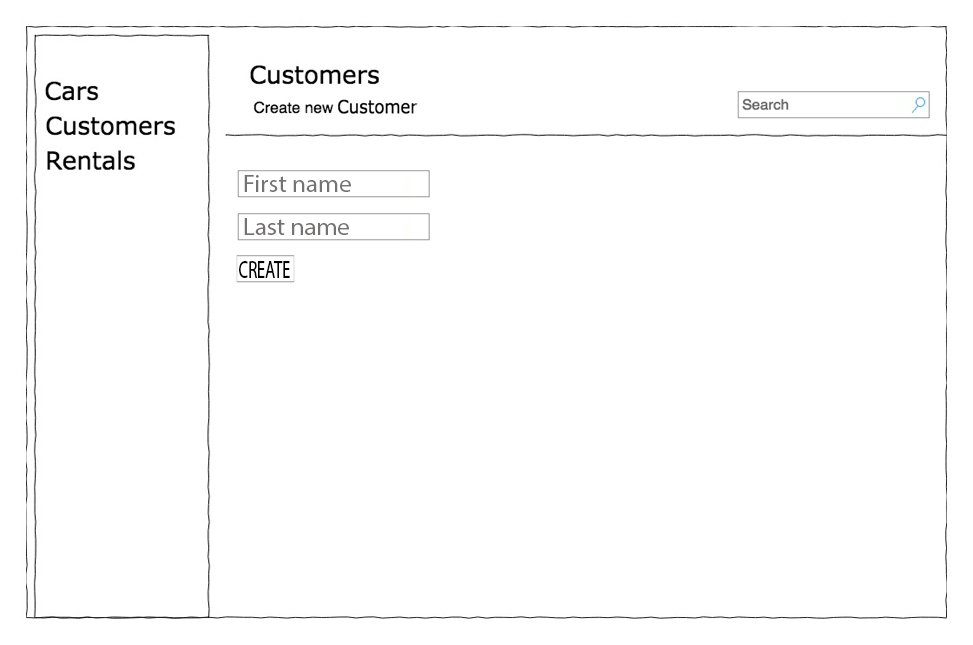
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.5 | User Site Map | |
|  |  | **Description:** | |

****

The site map is the structure of your website/app. As you can see here the user starts on the Home page and from there they are able to visit three child elements/branches, each of which contains another branch. A site map gives you an overview of the site.

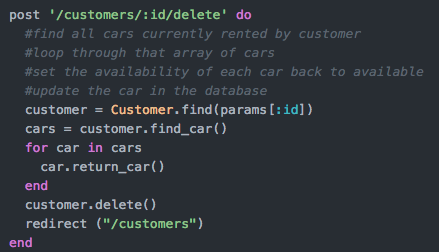
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.6 | 2 Wireframe Diagrams | |
|  |  | **Description:** | |

****

****

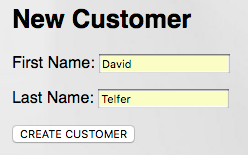
A wireframe is a visual representation of how the final product should look. This is beneficial as it gives you an idea of what is needed in order to accomplish this particular design or see any issues that may occur.

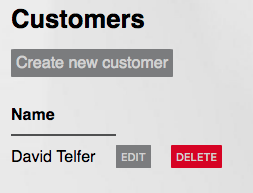
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.10 | Example of Pseudocode used for a method | |
|  |  | **Description:** | |

****

In this function I wanted to find a particular customer who had rented a car and return that car into the available stock. Using pseudocode allowed me to break it down into small steps in order for the function to work.

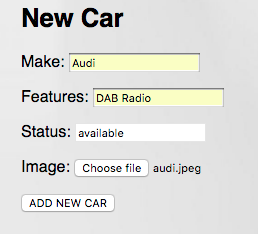
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.13 | Show user input being processed according to design requirements. Take a screenshot of:  \* The user inputting something into your program  \* The user input being saved or used in some way | |
|  |  | **Description:**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

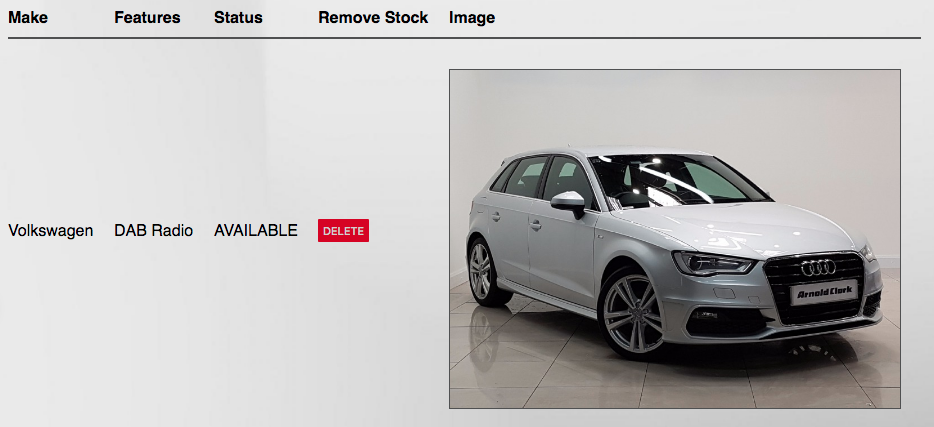
****

****

The first image shows the user inputting details to create a new customer. The second screenshot is the result of this input.

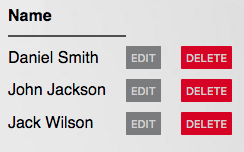
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.14 | Show an interaction with data persistence. Take a screenshot of:  \* Data being inputted into your program  \* Confirmation of the data being saved | |
|  |  | **Description:** | |

****

****

The first image allows the user to input details in order to add a new car to the database, including being able to select an image store locally on your computer. The second result shows the result of inputting this data.

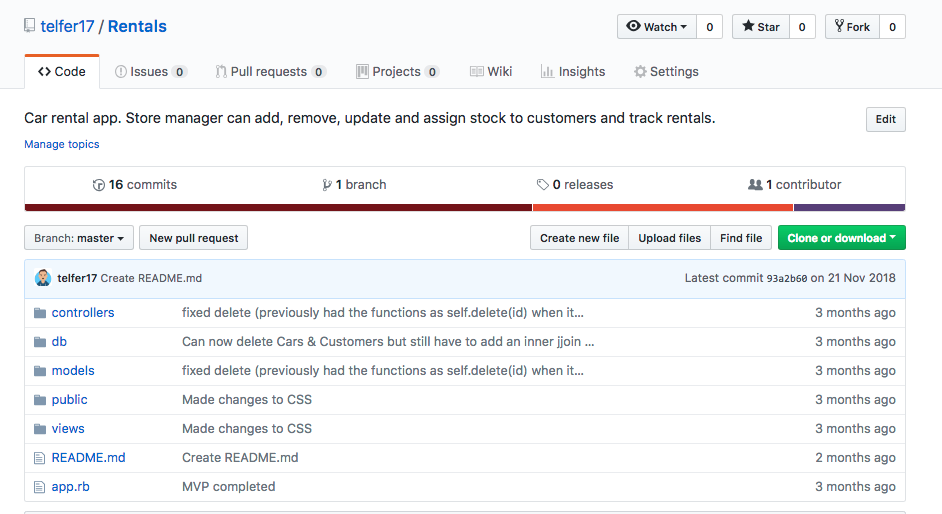
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.15 | Show the correct output of results and feedback to user. Take a screenshot of:  \* The user requesting information or an action to be performed  \* The user request being processed correctly and demonstrated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

****

****

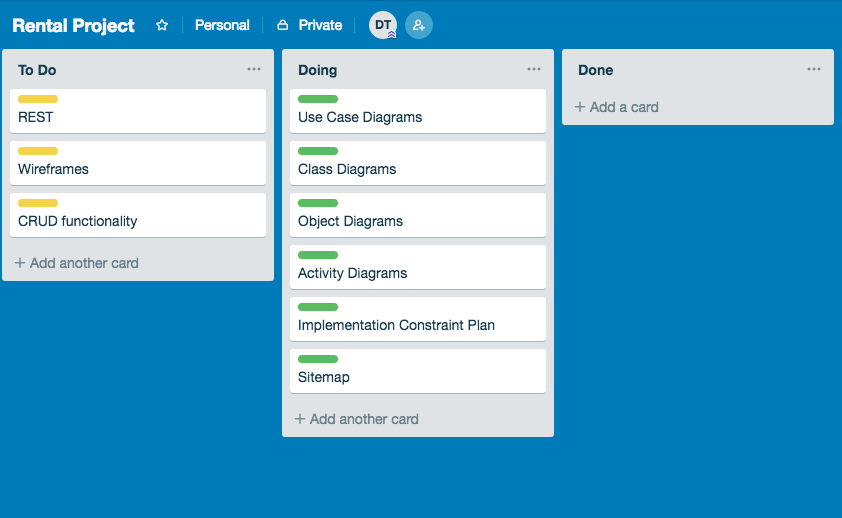
The first image shows a list of current customers in the database. After pressing the delete button, it removes the chosen customer and gives an updated list of customers on the page.

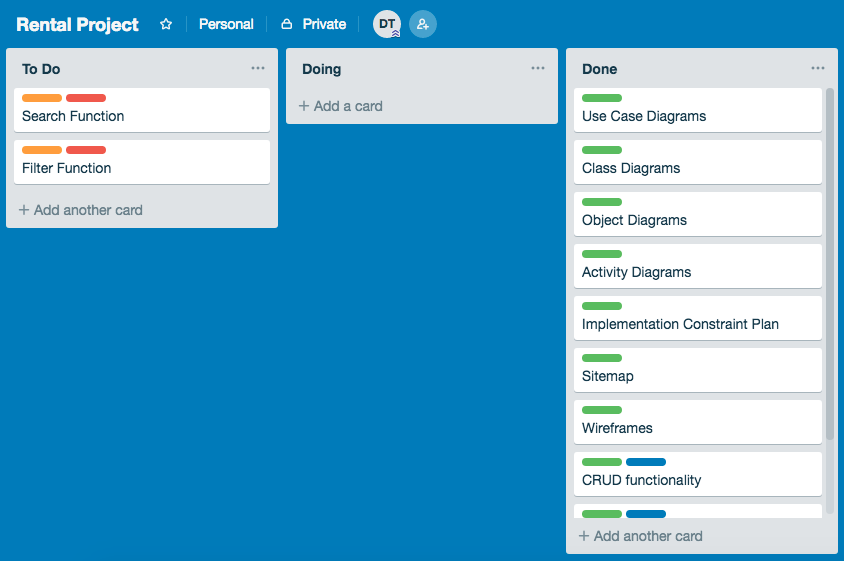
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.11 | Take a screenshot of one of your projects where you have worked alone and attach the Github link. | |
|  |  | **Description:** | |

****

**https://github.com/telfer17/Rentals**

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

****

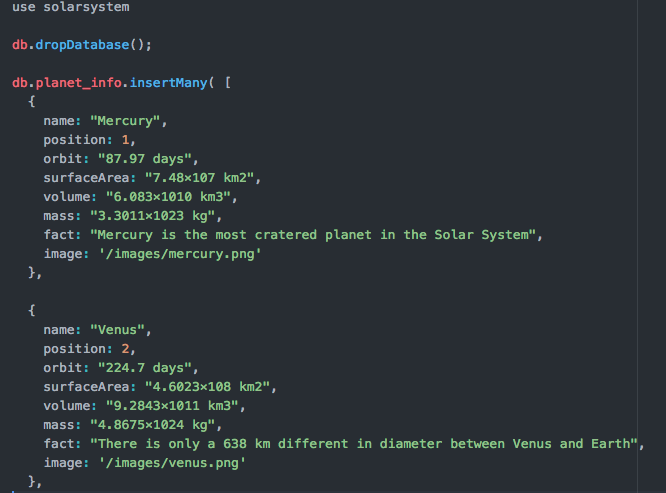
****

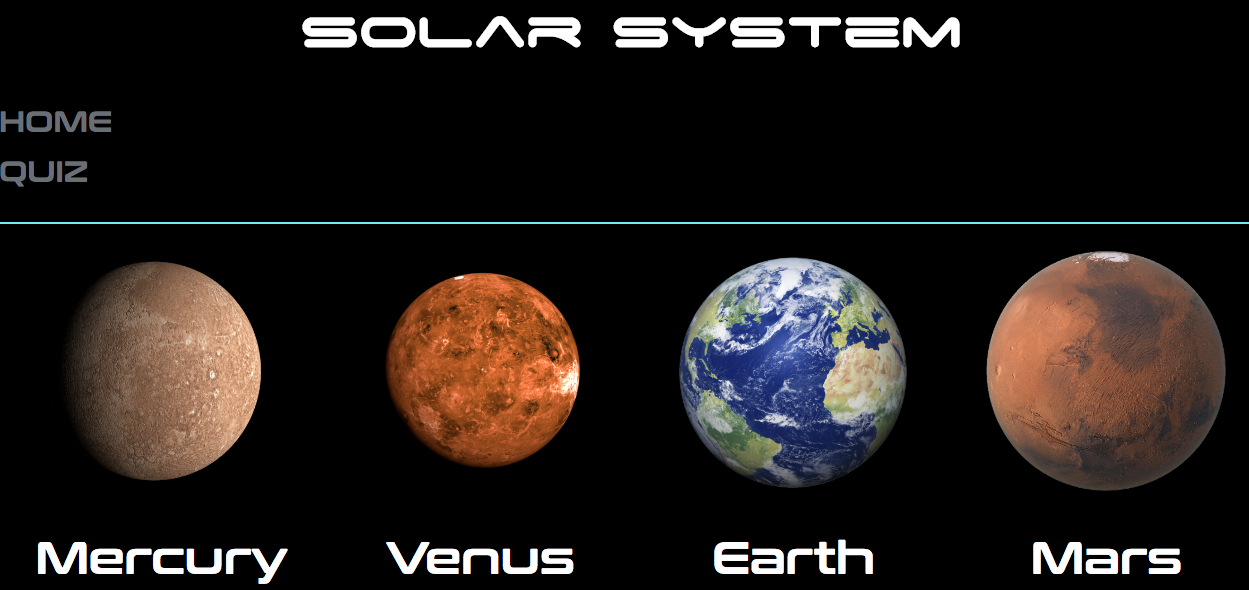
The first image is at the beginning of the project when things were still in the planning phase. The second image is from the end of the project. I had submitted the project but I never had time to implement the search function or the filter function so they stayed on the ‘to-do’ list.

**Week 7**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | |
|  |  | **Description:** | |

****

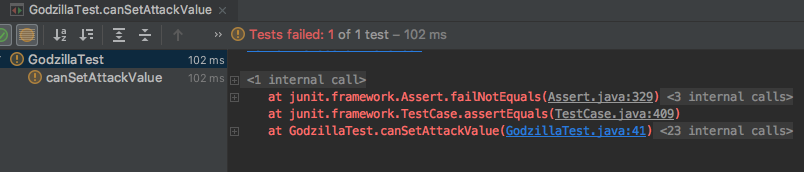
****

****

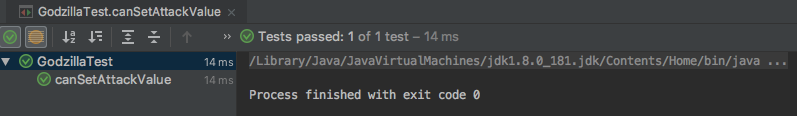
I created this API which stores information about planets in the solar system. The third image shows the planet images and names displaying (name: and image: in the API).

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | |
|  |  | **Description:** | |

****

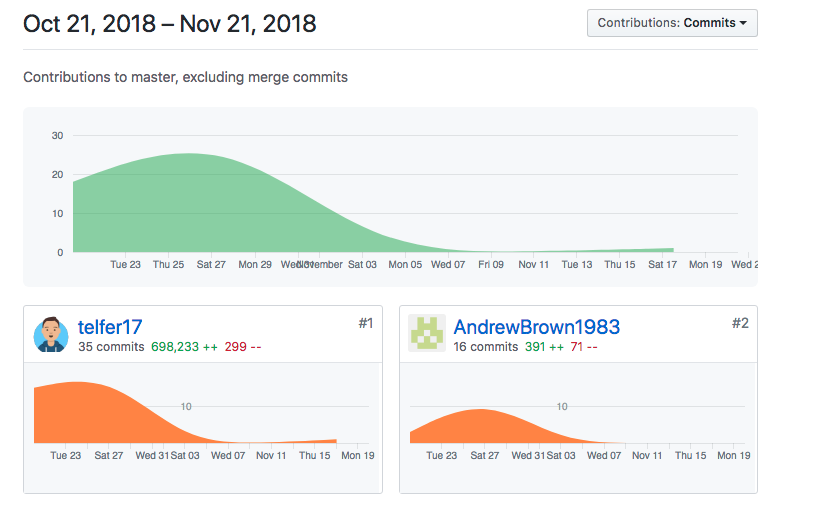
****

****

****

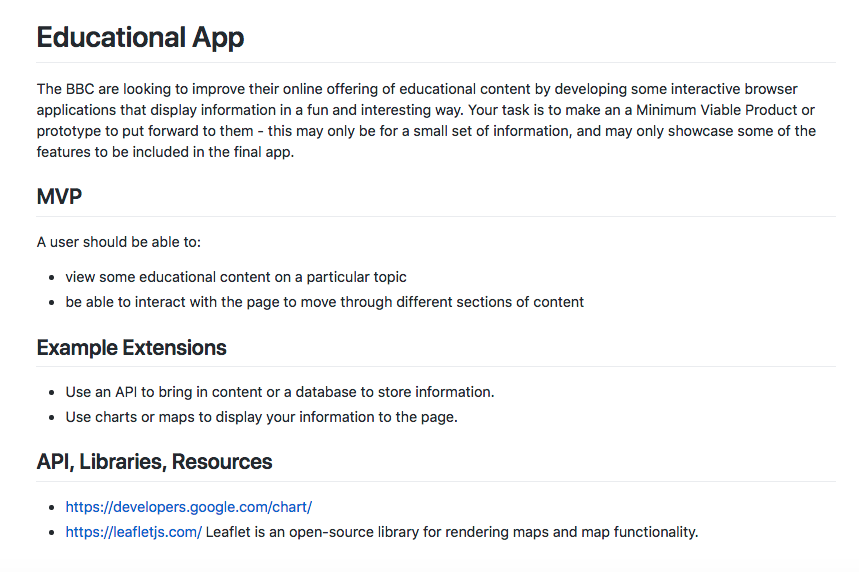
**Week 9**

|  |  |  |  |
| --- | --- | --- | --- |
| **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. | |
|  |  | **Description:** | |

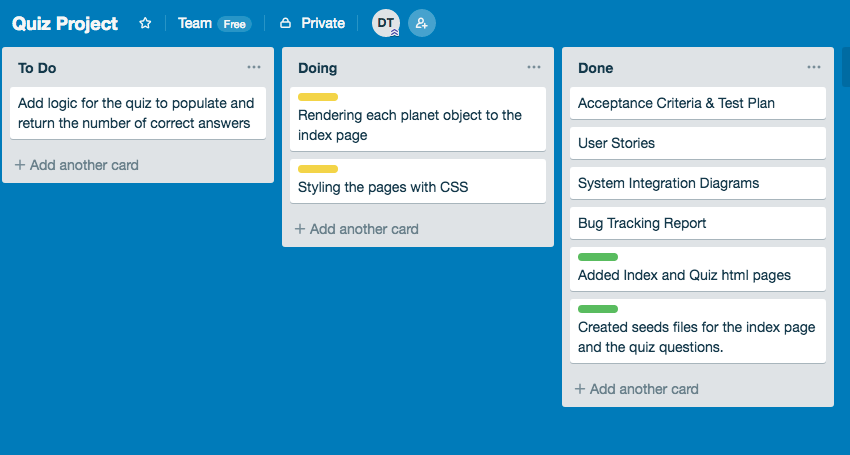
****

This was a pair programming project that we both worked on.

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

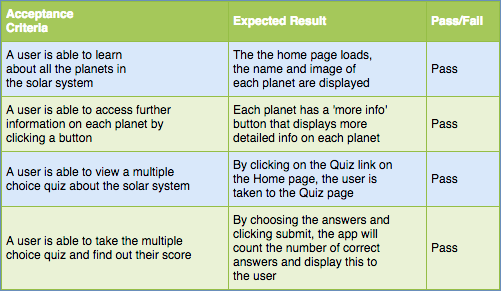
****

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

****

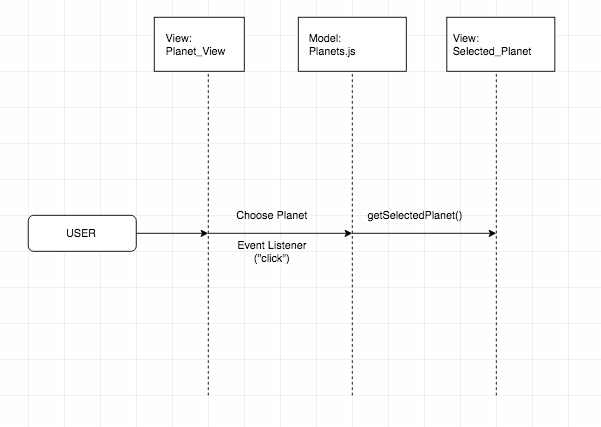
This screenshot was taken after we had already started creating the app. We would add/modify the board on a daily basis.

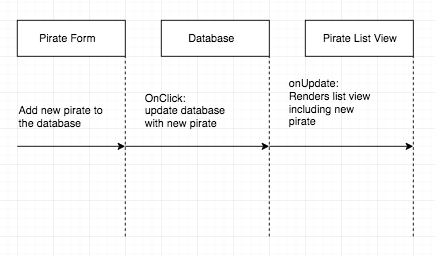
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.4 | Write an acceptance criteria and test plan. | |
|  |  |  | |

****

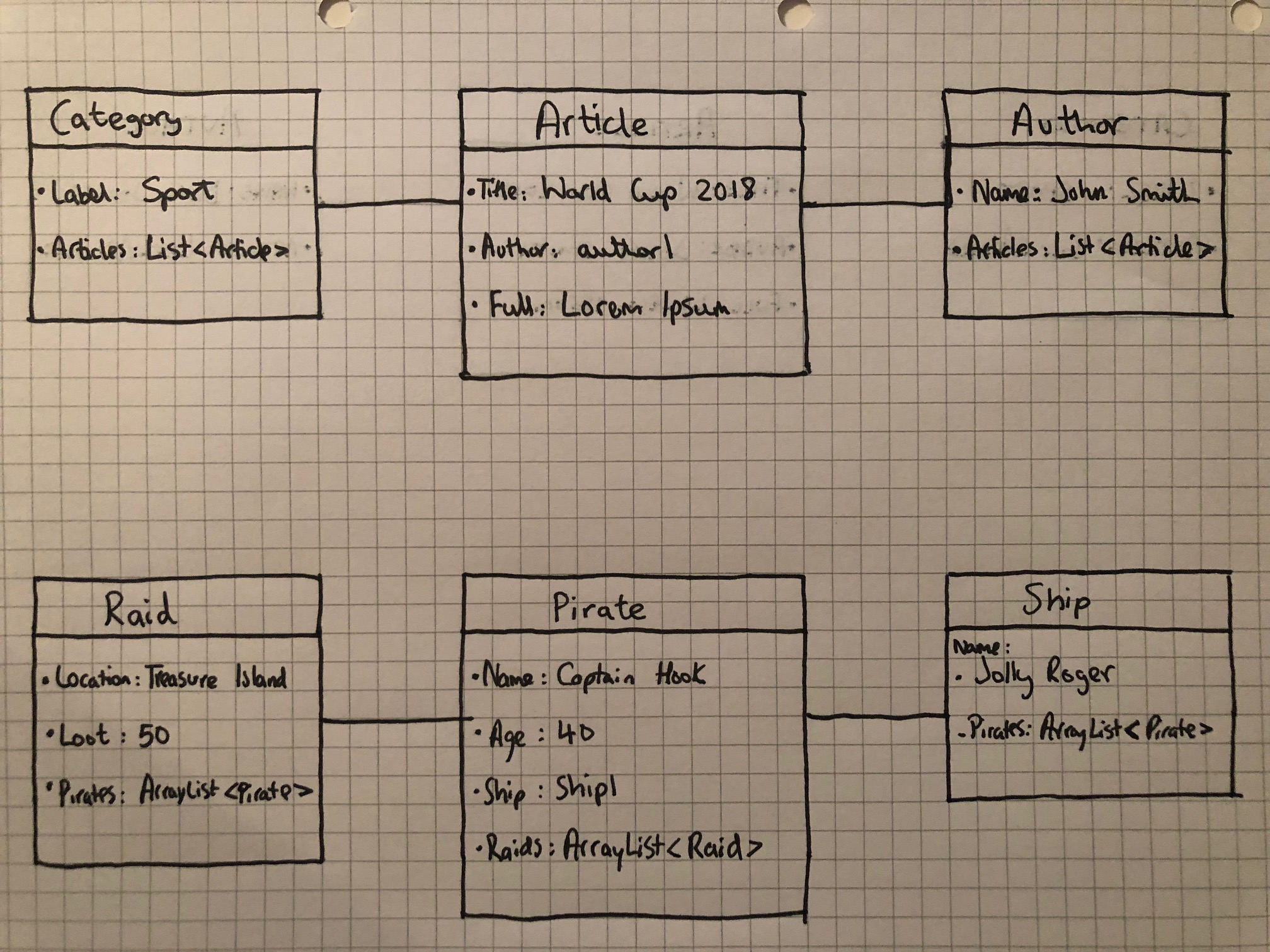
This gives examples what users will do when using the app. One of the benefits of using this in the planning stage is that it allows you to divide these tasks into smaller sections, therefore making it easier and cleaner to code (single responsibility principle).

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

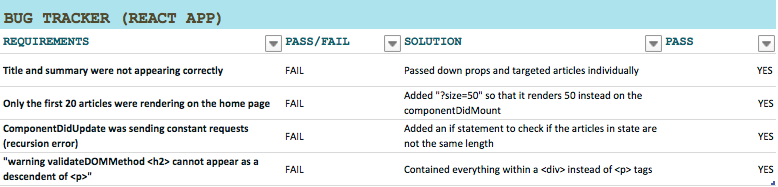
****

****

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

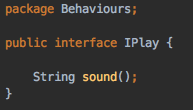
****

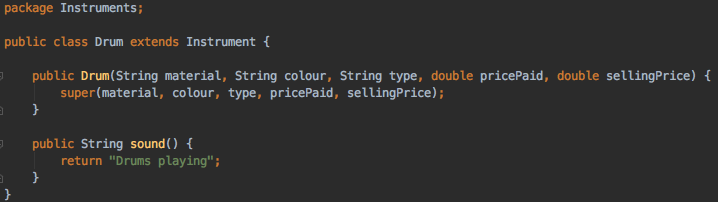
|  |  |  |  |
| --- | --- | --- | --- |
| **Unit** | **Ref** | **Evidence** |  |
| **P** | P.17 | Produce a bug tracking report | |
|  |  | **Description:** | |

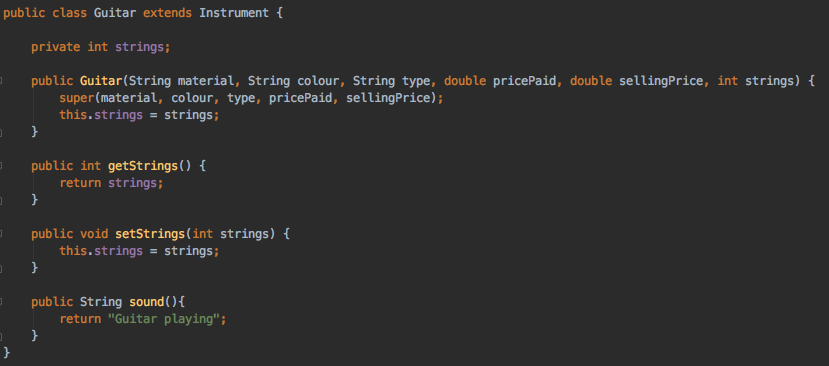
****

**Week 12**

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

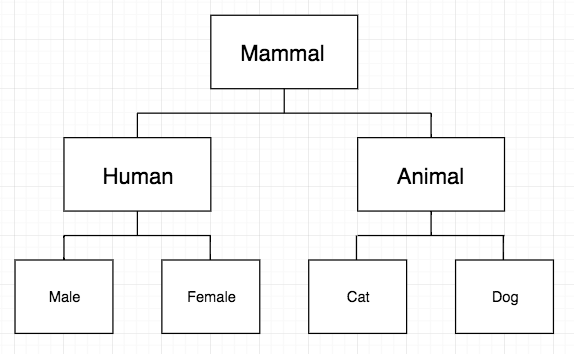
****

****

****

The interface (IPlay) is implemented by both the Drum and Guitar class and they both use the sound() function, however they use it in a different way (they each play a different sound). This is polymorphism.

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

****

Mammal is the main class, then the two sub classes of Human and Animal inherit from Mammal. Below that Male & Female inherit from Human and Cat & Dog inherit from Animal.

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

****

|  |  |  |  |
| --- | --- | --- | --- |
| **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. | |
|  |  | **Description:** | |

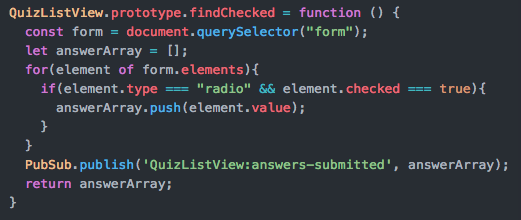
****

****

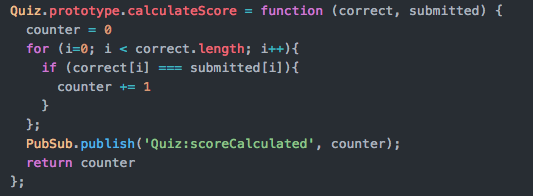
****

****

|  |  |  |  |
| --- | --- | --- | --- |
| **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. | |
|  |  | **Description:** | |



This algorithm was used in a quiz app. It checks the form and for each correct answer pushes it to the answerArray.



This algorithm is designed to calculate the total amount of correct answers. It takes in the number of correct answers and uses a counter to gather the total score.