

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.

Week 2

Unit	Ref	Evidence
I&T	I.T.5	<p>Demonstrate the use of an array in a program. Take screenshots of:</p> <ul style="list-style-type: none">*An array in a program*A function that uses the array*The result of the function running <p>Description:</p>

Paste Screenshot here

```
1 chickens = [
2   {name: "Margaret", age: 2, eggs:0},
3   {name: "Hetty", age: 1, eggs:2},
4   {name: "Henritta", age: 3, eggs:1},
5   {name: "Audrey", age: 2, eggs:0},
6   {name: "Mabel", age: 5, eggs:1},
7 ]
8
9 def count_eggs(array)
10   total_eggs = 0
11
12   for bird in array
13     total_eggs += bird[:eggs]
14     # bird[:eggs] = 0
15   end
16
17   return total_eggs.to_s() + " eggs collected"
18 end
19
20
21 p count_eggs(chickens)
```

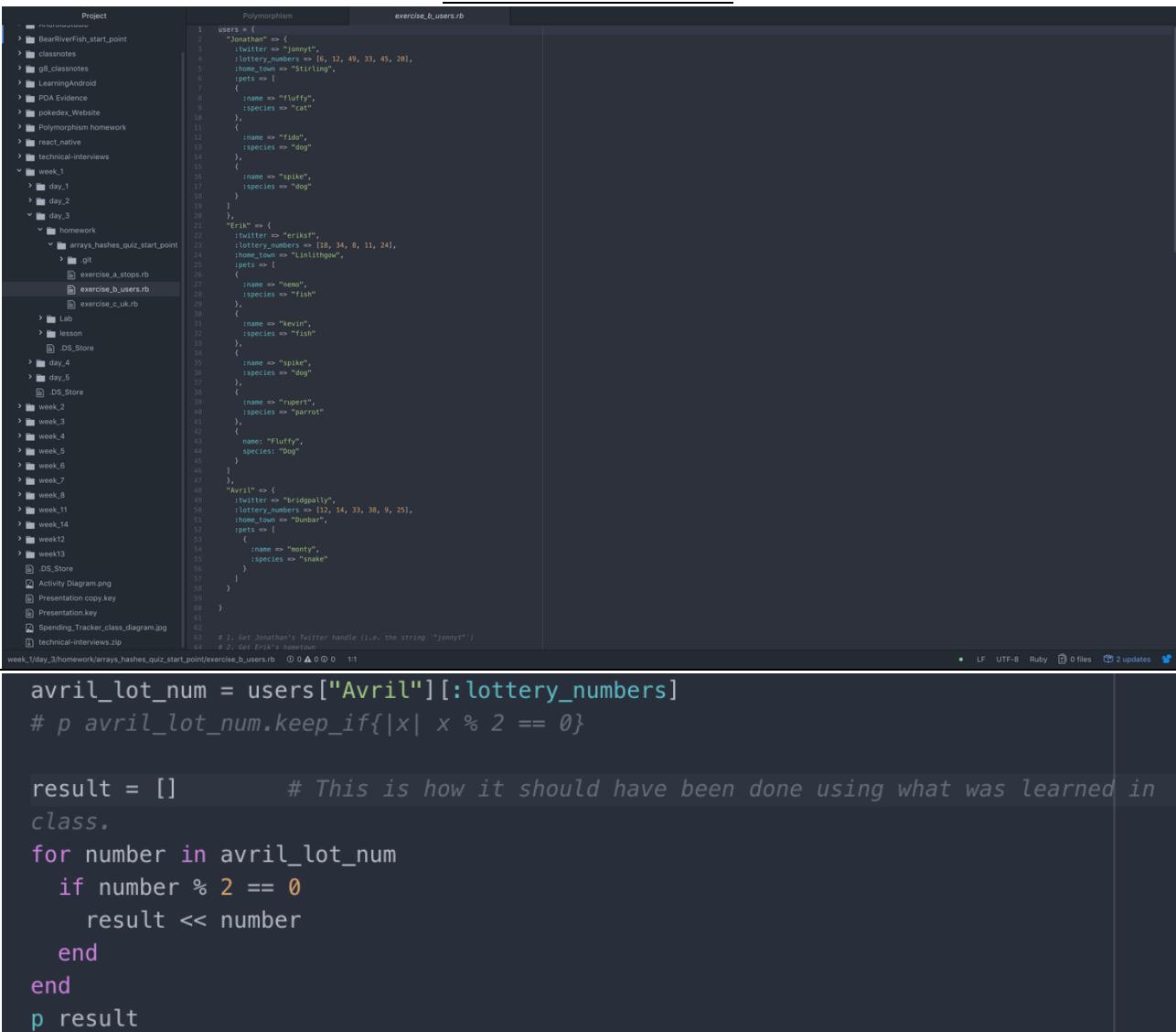
[→ loops ruby loops_in_functions.rb
"4 eggs collected"]

Description here

The function initializes a counter total_eggs and then loops through the array to add the eggs to the counter to find the total number of eggs within the array.

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:	

Paste Screenshot here



```

Project
- BearRiverFish_start_point
- classnotes
- gd_classnotes
- LearningAndroid
- PDA_Evidence
- pokede_Website
- Polymorphism homework
- react_native
- technical-interviews
- week_3
  - day_1
  - day_2
  - day_3
    - homework
      - arrays_hashes_quiz_start_point
        - .git
          - exercise_a_stops.rb
          - exercise_b_users.rb
          - exercise_c_uk.rb
    - Lab
    - lesson
      - DS_Store
  - day_4
  - day_5
    - DS_Store
- week_2
- week_3
- week_4
- week_5
- week_6
- week_7
- week_8
- week_11
- week_14
- week12
- week13
  - DS_Store
  - Activity Diagram.png
  - Presentation copy.key
  - Presentation.key
  - Spending_Tracker.class_diagram.jpg
  - technical-interviews.zip
week_3/day_3/homework/arrays_hashes_quiz_start_point/exercise_b_users.rb ① 0 ② 0 ③ 0 ④ 11

Polymorphism                               exercise_b_users.rb
1 USERS = [
2   {"Jonathan" => {
3     :twitter => "jonnyt",
4     :lottery_numbers => [6, 12, 49, 33, 45, 28],
5     :home_town => "Stirling",
6     :pets => [
7       {
8         :name => "fluffy",
9         :species => "cat"
10      },
11      {
12        :name => "fido",
13        :species => "dog"
14      },
15      {
16        :name => "spike",
17        :species => "dog"
18      }
19    ],
20    "Erik" => {
21      :twitter => "eriksf",
22      :lottery_numbers => [18, 34, 8, 11, 24],
23      :home_town => "Linlithgow",
24      :pets => [
25        {
26          :name => "nemo",
27          :species => "fish"
28        },
29        {
30          :name => "keytar",
31          :species => "fish"
32        },
33        {
34          :name => "spike",
35          :species => "dog"
36        },
37        {
38          :name => "rigert",
39          :species => "parrot"
40        },
41        {
42          :name => "fluffy",
43          :species => "dog"
44        }
45      ]
46    },
47    "Avril" => {
48      :twitter => "bridgpally",
49      :lottery_numbers => [12, 14, 33, 38, 9, 25],
50      :home_town => "Dunbar",
51      :pets => [
52        {
53          :name => "monty",
54          :species => "snake"
55        }
56      ]
57    }
58  }
59  # 1. Get Jonathan's Twitter handle (i.e. the string "jonnyt")
60  # 2. Get Erik's hometown
61
62  # p avril_lot_num.keep_if{|x| x % 2 == 0}
63
64  result = [] # This is how it should have been done using what was learned in
65  class.
66  for number in avril_lot_num
67    if number % 2 == 0
68      result << number
69    end
70  end
71  p result

```

```

codeclan_work — craigdunlop@craigs-MacBook-Pro-2 — ..codeclan_work — -zs
→ codeclan_work ruby week_1/day_3/homework/arrays_hashes_quiz_start_point/exercis
ise_b_users.rb
[12, 14, 38]
→ codeclan_work []

```

Description here

The function creates an empty array called result. It then loops through the array of lottery numbers and if it is an even number then it is added to the result array.

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:	

Paste Screenshot here

```

def order()
  sql = "SELECT * FROM pizza_orders WHERE customer_id = $1"
  values = [@id]
  orders = SqlRunner.run(sql,values)
  return orders.map{|order_hash|PizzaOrder.new(order_hash)}
end
end

From: /Users/craigdunlop/Documents/codeclan_work/week_3/day_3/pizza_sh
ole.rb @ line 29 :

24:   order1.save()
25:   order2.save()
26:   cust1_orders = customer1.order()
27:
28:   binding.pry
=> 29: nil

[(1) pry(main)> cust1_orders
=> [#<PizzaOrder:0x007fbce5aec578
  @customer_id=1,
  @id=1,
  @quantity=2,
  @topping="Cheese">,
 #<PizzaOrder:0x007fbce5aec488
  @customer_id=1,
  @id=2,
  @quantity=2,
  @topping="Veg">]
(2) pry(main)> []

```

Description here

The function searches through the database for orders which have the same customer id being searched. it then returns an array of pizza's.

Unit	Ref	Evidence	
I&T	I.T.4	<p>Demonstrate sorting data in a program. Take screenshots of:</p> <ul style="list-style-type: none"> *Function that sorts data *The result of the function running 	Description:

Paste Screenshot here

```
def self.all()
    sql = "SELECT * FROM transactions ORDER BY transaction_date DESC"
    results = SqlRunner.run(sql)
    transactions = results.map{|transaction|Transaction.new(transaction)}
    return transactions
end
```

```
spending_tracker=# SELECT * FROM transactions ORDER BY transaction_date DESC;
 id | amount | transaction_date | merchant_id | tag_id
----+-----+-----+-----+-----+
 96 | 23.57 | 2018-10-03 |      73 |      74
 97 | 32.00 | 2018-10-03 |      80 |      74
 95 | 34.50 | 2018-10-02 |      73 |      74
 94 | 75.40 | 2018-10-01 |      73 |      74
 88 | 23.50 | 2018-09-12 |      73 |      74
(5 rows)

spending_tracker=#
```

RESULTS OF FUNCTION

Description here

Transactions are sorted by most recent order.

Week 5 and 6

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

Paste Screenshot here



Description here

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

Paste Screenshot here



Description here

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

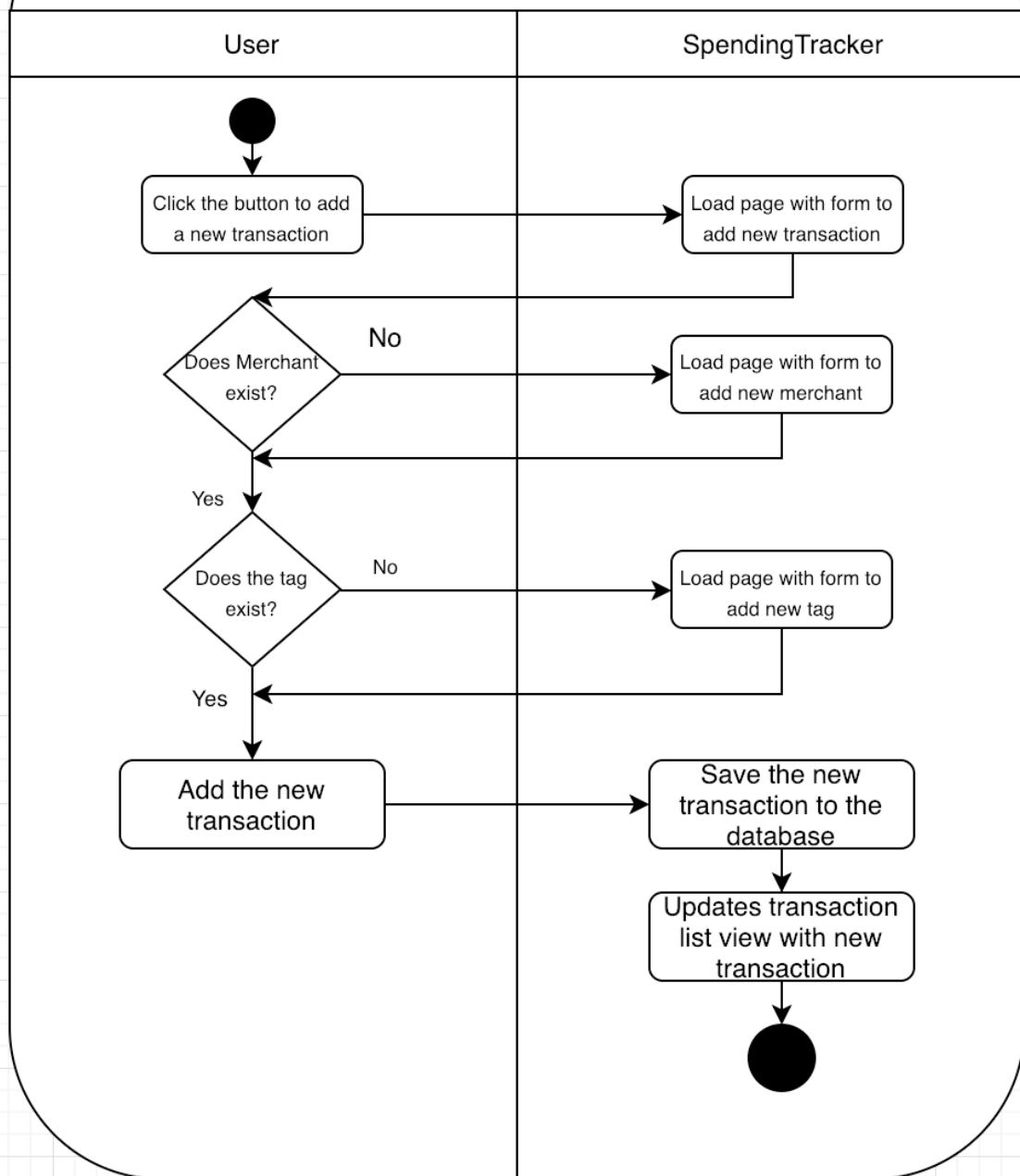
Paste Screenshot here



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

Paste Screenshot here

User adding a new transaction



Description here

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:	

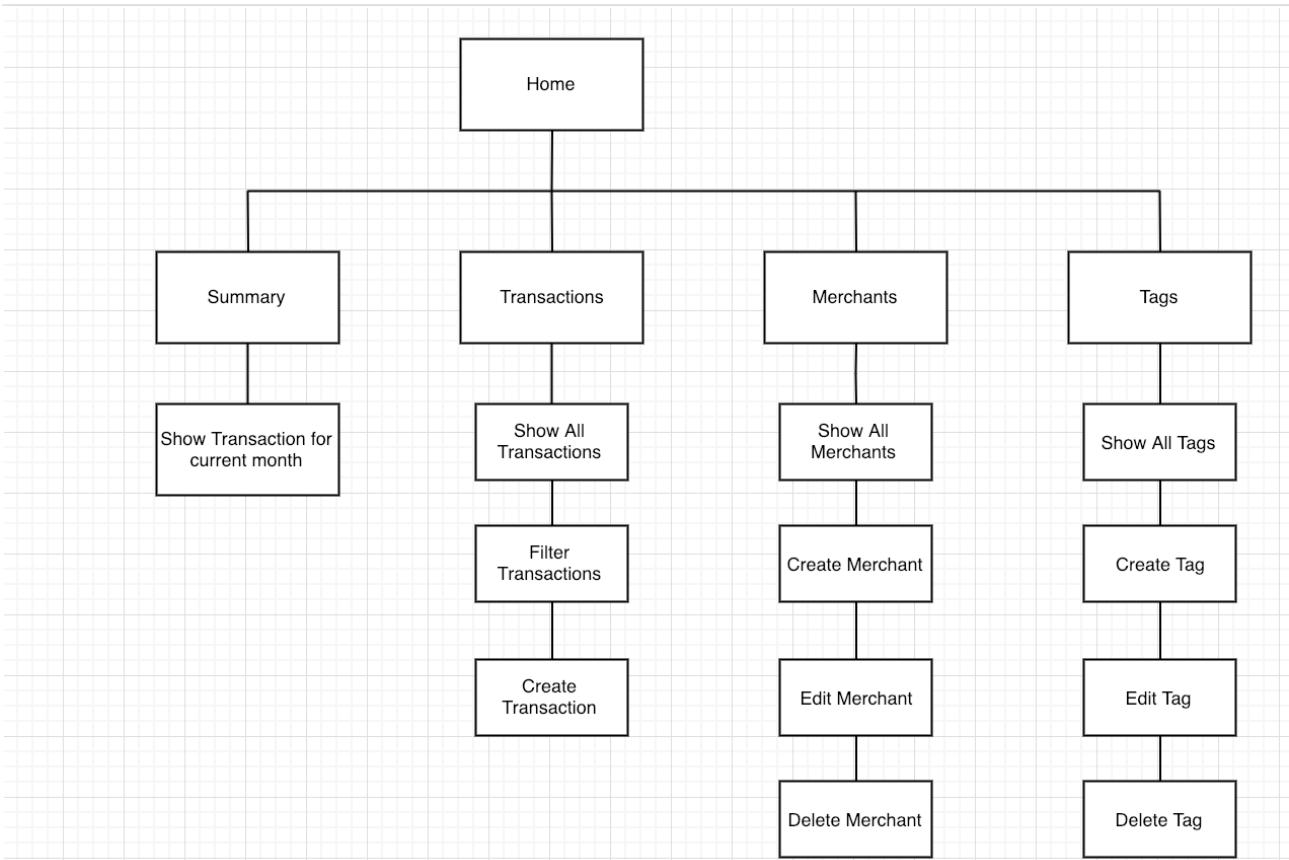
Paste Screenshot here

1	Constraint Category	Implementation Constraint	Solution
2	Hardware and Software Platforms	Intended to be used at home and on the move by users, so should support web and mobile use.	Ensure website design is responsive so it is legible on mobile devices.
3	Performance and Requirements	Must run on mobile devices. Multiple users may impact website performance.	Perform performance tests and ensure web performance is optimised where possible.
4	Persistent Storage and Transactions	Must be able to store Categories, Merchants and Transactions created by user.	Using PostreSQL and implementing CRUD methods to allow data to be stored,read, updated and deleted
5	Usability	implementing forms on mobile must be easily readable and ensure buttons are easily clickable on smaller devices	implement good responsive design to ensure applications remains user-friendly on different devices.
6	Budgets	No Budget required	Only one person working on project for free.
7	Time Limitations	Only 5 days available for completion of project.	Ensure project is planned out successfully with realistic deadlines and requirements. Manage time effectively

Description here

Unit	Ref	Evidence	
P	P.5	User Site Map	
		Description:	

Paste Screenshot here

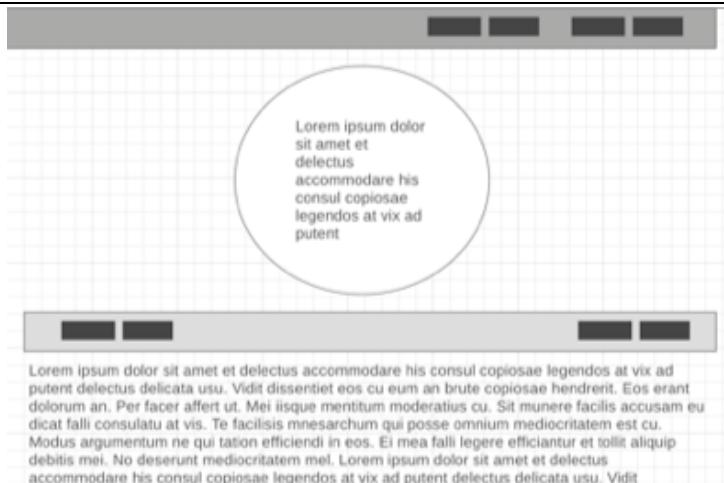


Unit	Ref	Evidence
P	P.6	<p>2 Wireframe Diagrams</p> <p>Description:</p>

Paste Screenshot here

Summary

Spending	Total Spent: £XX.xx
Category	£XX.XX
Category	£XX.XX
Category	£XX.XX
Category	£XX.XX
Category	£XX.XX



Description here

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

Paste Screenshot here

```

def buy_ticket(screening)
    # find screening
    sql = "SELECT films.* FROM
        screenings INNER JOIN films ON film_id = films.id WHERE films.id = $1"
    values = [screening.film_id]

    film_at_screening = SqlRunner.run(sql,values)
    result = film_at_screening.first
    price = result["price"].to_i
    # if customer has enough funds remove price from customer funds
    if @funds > price && screening.seats_available
        pay_ticket(price)
    # buy ticket for screening
    screening.sell_ticket
    return "Ticket successfully purchased"
else
    return "Unsuccessful"
end
end

```

Description here

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:

Paste Screenshot here

Transactions		
Create new Transaction	FILTER BY DATE:	Filter by status
Thu - 03 January	Spa Town Coffee	£50.00
Mon - 12 November	Amazon	£50.00
Wed - 03 October	YANKEE CANDLE	£32.00
Wed - 03 October	Amazon	£23.57
Tue - 02 October	Amazon	£34.50
Mon - 01 October	Amazon	£75.40

January Transactions		
FILTER BY DATE: <input type="text"/> <input type="text"/> Filter by date		
Thu - 03 January	Spa Town Coffee	£50.00

Description here

The user inputs a date which then brings back the transactions for that given date.

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:	

Paste Screenshot here

Add New Transaction						
Amount: £	<input type="text" value="50"/>	Select a Merchant:	<input type="text" value="Spa Town Coffee"/>	Select a Category:	<input type="text" value="Shopping"/>	Transaction Date: <input type="text" value="03/01/2019"/> Add transaction

Transactions		
Create new Transaction	FILTER BY DATE: <input type="text"/> <input type="text"/>	Filter by date
Thu - 03 January	Spa Town Coffee	£50.00
Mon - 12 November	Amazon	£50.00
Wed - 03 October	YANKEE CANDLE	£32.00

Description here

User inputs a transactions amount with a merchant category and transaction date. This is then saved to the databased and will be rendered when the transactions page is viewed.

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:

Paste Screenshot here

Transactions		
Create new Transaction	FILTER BY DATE:	<input type="text" value="January 2019"/> Filter by date
Thu - 03 January	Spa Town Coffee	£50.00
Mon - 12 November	Amazon	£50.00
Wed - 03 October	YANKEE CANDLE	£32.00
Wed - 03 October	Amazon	£23.57
Tue - 02 October	Amazon	£34.50
Mon - 01 October	Amazon	£75.40

January Transactions

January Transactions		
Thu - 03 January	Spa Town Coffee	£50.00

Description here

The user inputs a month/year they would like to filter transactions by and press the filter by date button which will then bring back a new view with all the transactions within that month/year.

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:

Paste Screenshot here

The screenshot shows a web-based spending tracker application. At the top, there's a navigation bar with links for Home, Summary, Merchants, Tags, and Transactions. Below the navigation is a teal-colored header bar with the text "Welcome User, View your monthly summary below". Underneath this, there's a large teal box containing the word "Overview:" followed by budget and spending details: Budget: £1550.0, Spent: £50.0, Available: £1500.0. Below this section is a table showing a single transaction: Thu - 03 January, Spa Town Coffee, £50.00.

<u>Description here</u>		
<u>https://github.com/cd9393/Week5_Spending_Tracker</u>		

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

Paste Screenshot here

https://trello.com/b/4200whrY/stock-tracker-features

Trello

Stock Tracker Features

Features

- User should be able to make projections based on financial modelling
- user should be able to filter graphs/data by timeframe day/week/month/6month/1year
- search bar for cryptos which filters the list
- User should be add a crypto to their portfolio
- User should be able to view their total portfolio value
- Users should be able to see a list of cryptocurrency prices
- User should be able to see a chart of individual items performance within their porfolio
- User Should be able to view individual performance of items in their portfolio
- user should be able to input purchase price and have profit/loss shown on each individual/overall
- User should be able to view a chart of their current value portfolio

+ Add another card

Doing

- + Add a card

Blockers

- + Add a card

Done

- + Add a card

+ Add another list

https://trello.com/b/4200whrY/stock-tracker-features

Trello

Stock Tracker Features

Features

- User should be able to make projections based on financial modelling
- user should be able to filter graphs/data by timeframe day/week/month/6month/1year
- search bar for cryptos which filters the list

+ Add another card

Doing

- User should be able to view a chart of their current value portfolio

Blockers

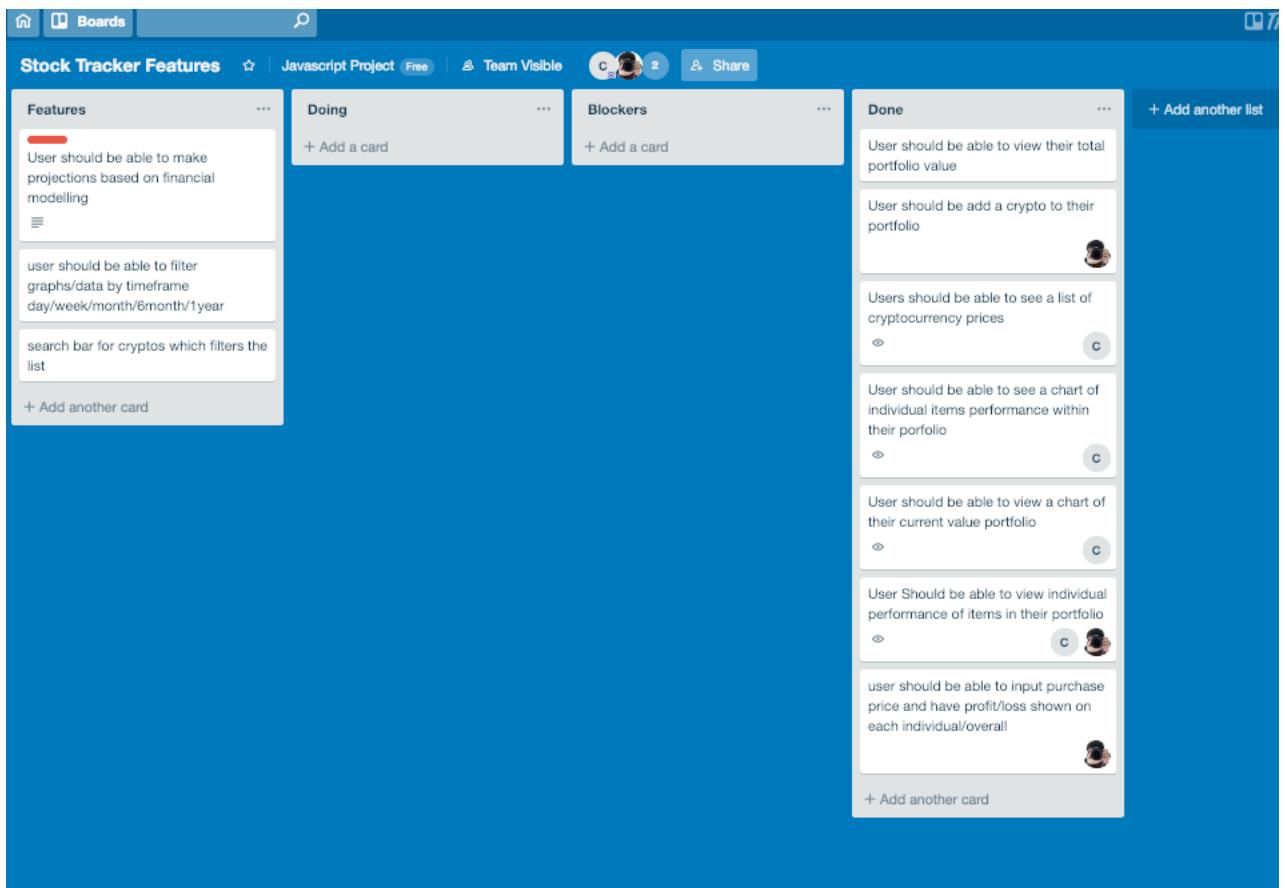
- + Add a card

Done

- User should be able to view their total portfolio value
- User should be add a crypto to their portfolio
- Users should be able to see a list of cryptocurrency prices
- User should be able to see a chart of individual items performance within their porfolio

+ Add another card

+ Add another list



Description here

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:	

Paste Screenshot here

```
const Request = function (url) {
  this.url = url;
}

Request.prototype.get = function () {
  return fetch(this.url).then(response => response.json());
};

module.exports = Request;
```

```
Pokemon.prototype.getData = function () {
  const url = "https://pokeapi.co/api/v2/pokemon/";
  const request = new Request(url);

  request.get().then(data => {
    this.data = data.results;
    PubSub.publish('Pokemon:pokemon-data-loaded', this.data);
  })
  // request.get((data) => {
  //   this.data = data.message;
  //   PubSub.publish('Dogs:dog-data-loaded', this.data);
  //   // });
}
}
```

```
PokemonListView.prototype.createPokedex = function (pokemonArray) {
  const pokedex = document.createElement('ul');
  pokedex.classList.add('pokedex');
  pokemonArray.forEach((pokemon, index) => {
    const number = index+1;
    const listItem = document.createElement('li');
    listItem.id = pokemon.name;
    listItem.value = pokemon.name;
    listItem.textContent = `----No${number} ${pokemon.name.capitalize()}`;
    listItem.addEventListener("click", (event) => {
      const selectedPokemon = event.target.id;
      PubSub.publish("pokemonListView: Pokemon-selected", selectedPokemon)
    })
    pokedex.appendChild(listItem)
  })
  return pokedex;
};
```

The screenshot shows a web-based Pokédex application with a red header containing the word "Pokédex". Below the header is a search bar with the placeholder "Search". A "Filter Pokemon By Type" dropdown menu is open, showing options like "All", "Normal", "Fire", "Water", "Grass", "Electric", "Ice", "Fairy", "Poison", "Ground", "Rock", "Bug", "Psychic", "Dragon", "Fighting", "Steel", "Ghost", "Dark", "Fairy", and "Unknown". The main content area displays a list of 11 Pokémons in rounded rectangular boxes:

- No1 Bulbasaur
- No2 Ivysaur
- No3 Venusaur
- No4 Charmander
- No5 Charmeleon
- No6 Charizard
- No7 Squirtle
- No8 Wartortle
- No9 Blastoise
- No10 Caterpie
- No11 Metapod

Description here

An API request is made to get all the pokemon information from the API. This is then published out using PubSub to the views page which then creates the list of pokemon as shown above.

Unit	Ref	Evidence
P	P.18	<p>Demonstrate testing in your program. Take screenshots of:</p> <ul style="list-style-type: none"> * 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:

Paste Screenshot here

```

require_relative('card.rb')
class CardGame # no initialize statement class has no variables.

def checkforAce(card)
  if card.value = 1 # this will always set the value of the card to be one 1 should be ==
    return true
  else
    return false
  end
end

def highest_card(card1 card2) #dif instead of def, no comma between card1,card2
  if card1.value > card2.value
    return card.name #card isn't a defined variable .should be card1. # .name isn't a class method should be suit.
  else
    card2 # return statement missing
  end
end
end # too many ends statements. This ends the class.

def self.cards_total(cards)
  total # should be =0
  for card in cards
    total += card.value
  return "You have a total of" + total # this should be outside the for loop to return the total. total should be included in the string with {}
end
end

```

```

**FIGURE 1- TEST CODE**

```

from PDA Evidence/Static_and_Dynamic_Task_A/specs/testing_spec.rb:3:in `<main>'
+ codeclan_work ruby PDA\ Evidence/Static_and_Dynamic_Task_A/specs/testing_spec.rb
/Users/craigdunlop/Documents/codeclan_work/PDA Evidence/Static_and_Dynamic_Task_A/testing_task_2.rb:46:in `<class:CardGame>': undefined local vari
able or method `card2' for CardGame:Class (NameError)
 from /Users/craigdunlop/Documents/codeclan_work/PDA Evidence/Static_and_Dynamic_Task_A/testing_task_2.rb:35:in `<top (required)>'
 from PDA Evidence/Static_and_Dynamic_Task_A/specs/testing_spec.rb:3:in `require_relative'
 from PDA Evidence/Static_and_Dynamic_Task_A/specs/testing_spec.rb:3:in `<main>'


```

**FIGURE 2- TEST FAILING TO RUN**

```

5
6 require_relative('card.rb')
7 class CardGame
8
9
10 def checkforAce(card)
11 if card.value == 1
12 return true
13 else
14 return false
15 end
16 end
17
18 def highest_card(card1,card2)
19 if card1.value > card2.value
20 return card1
21 else
22 card2
23 end
24 end
25
26 def self.cards_total(cards)
27 total = 0
28 for card in cards
29 total += card.value
30 end
31 return "You have a total of #{total}"
32 end
33
34

```

**FIGURE 3-CORRECTED TEST CODE**

```

[→ specs git:(master) ✘ ruby testing_spec.rb
Run options: --seed 65002

Running:

.....
Finished in 0.001206s, 3316.7496 runs/s, 3316.7496 assertions/s.

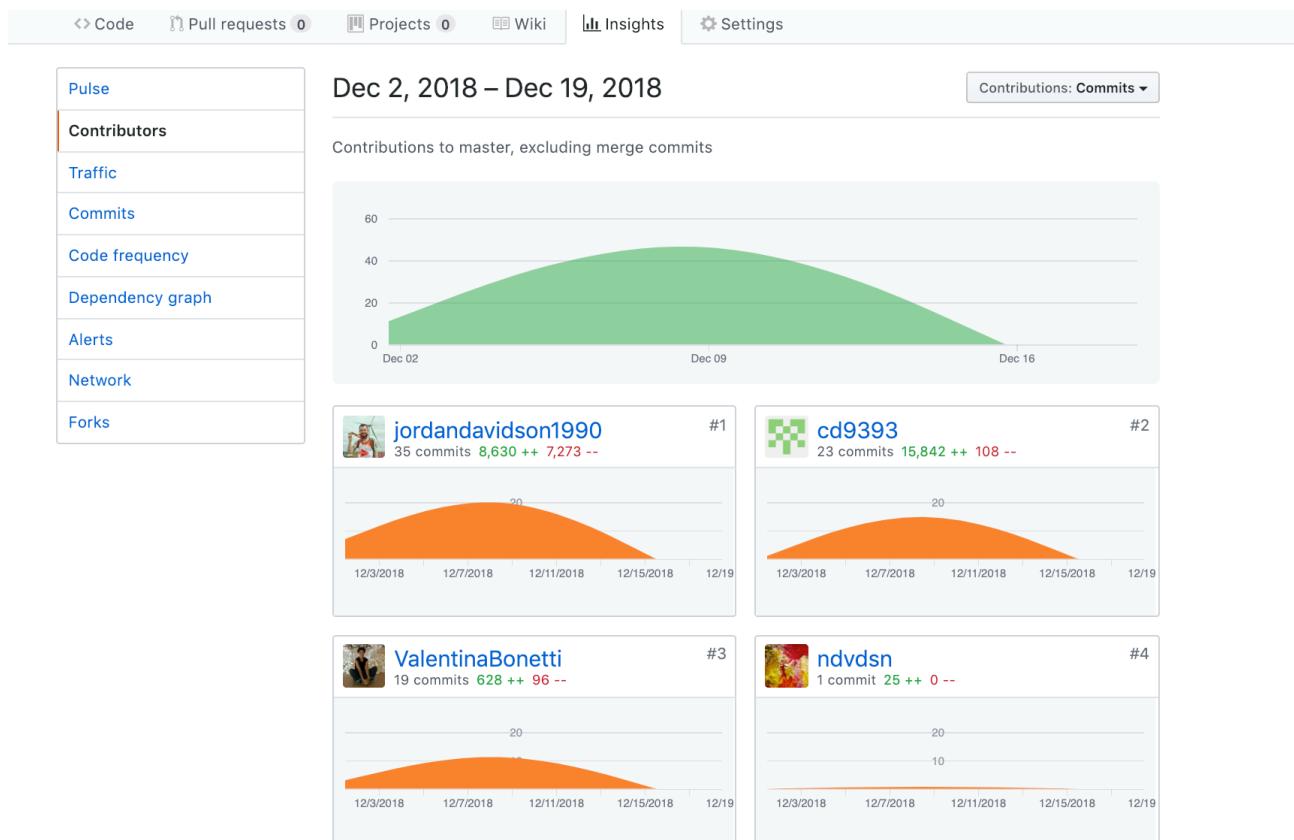
4 runs, 4 assertions, 0 failures, 0 errors, 0 skips
[→ specs git:(master) ✘]

```

**FIGURE 4-TESTS PASSING**  
[Description here](#)

| 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. |  |
|      |     | <b>Description:</b>                                                                                             |  |

### Paste Screenshot here



**Description here**  
Contributors page from Group project.

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

### Paste Screenshot here

## Restaurant Booking System:

You have been tasked to create a booking system for a brand new restaurant. The restaurant needs a way to book and arrange tables for customers who are booking over the phone. This system is for the staff to use.

### MVP:

Your system must be able to:

- Allow a customer to book a table at the restaurant for a particular time and date
- Update a booking, for example if the customer wants to change a booking time
- Display a list of bookings for a given date
- Display a list of customers ordered by frequency of visits

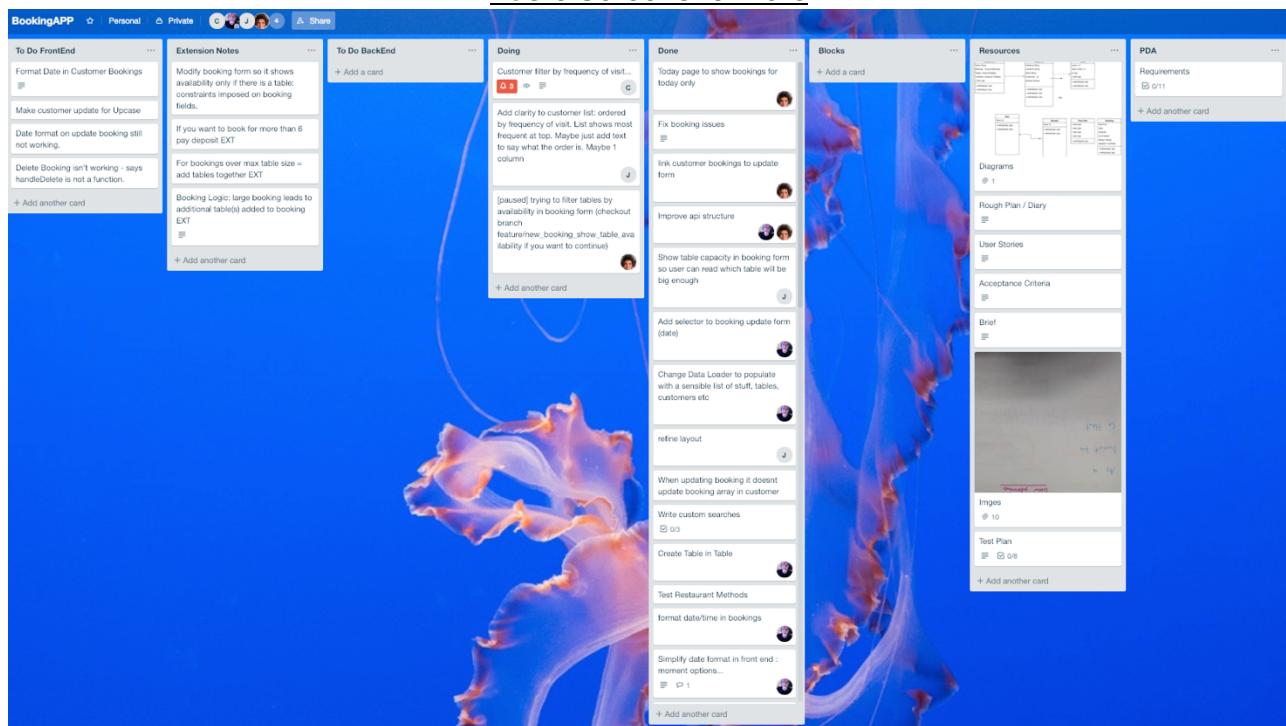
### Project Extensions:

- Don't allow double bookings
- Add a customer's receipt to a booking so you can view their previous orders and how much they spent
- Calculate how much a customer has spent over a given period of time
- Give discounts to frequent customers
- Whatever features you think would be beneficial to a restaurant

## Project Brief from Group Project

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

### Paste Screenshot here



### Description here

| <b>Unit</b> | <b>Ref</b> | <b>Evidence</b>                             |  |
|-------------|------------|---------------------------------------------|--|
| <b>P</b>    | P.4        | Write an acceptance criteria and test plan. |  |
|             |            |                                             |  |

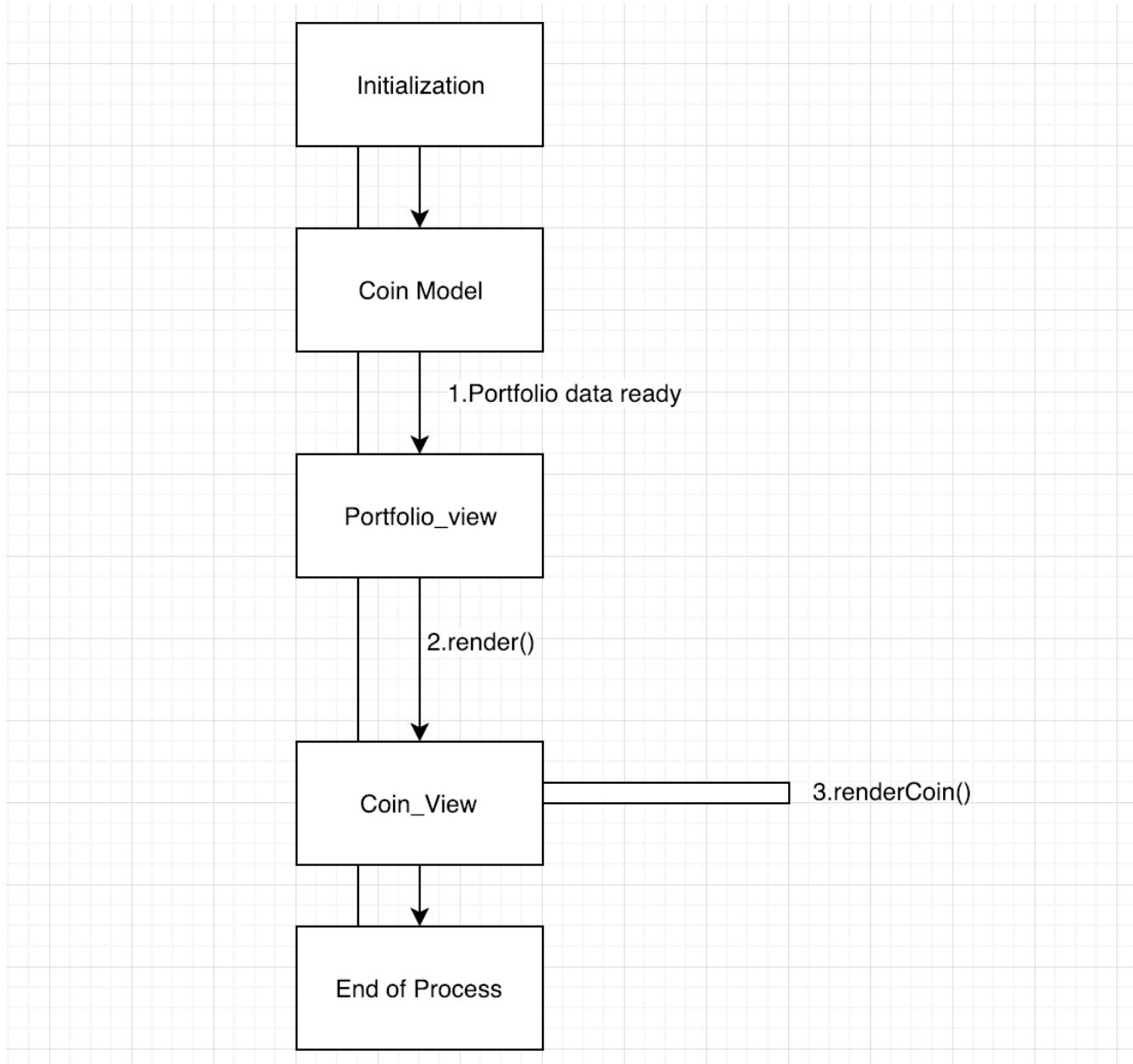
**Paste Screenshot here**

| <b>Acceptance Criteria -<br/>“A user should be able to...”</b>                                          | <b>Expected Output/Result</b>                                                                                                                                                                                                                             | <b>Test Result (Pass/Fail)</b> |
|---------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------|
| view total current value.                                                                               | <ul style="list-style-type: none"> <li>The main page should display total current value when a user visits the home page of the website</li> </ul>                                                                                                        | Pass                           |
| view individual and total performance trends.                                                           | <ul style="list-style-type: none"> <li>The user should be able to view total performance trends when the individual element is clicked</li> </ul>                                                                                                         | Pass                           |
| retrieve a list of share prices from an external API and allow the user to add shares to her portfolio. | <ul style="list-style-type: none"> <li>A user should be able to view a list of all API entries retrieved when the ‘Crypto’ button is clicked</li> <li>A user should be able to add a Crypto asset to the list when the ‘add’ button is clicked</li> </ul> | Pass                           |
| View a chart of the current values in her portfolio.                                                    | <ul style="list-style-type: none"> <li>The user should be able to view a performance chart of the asset when the portfolio asset is clicked</li> </ul>                                                                                                    | Pass                           |

**Description here**

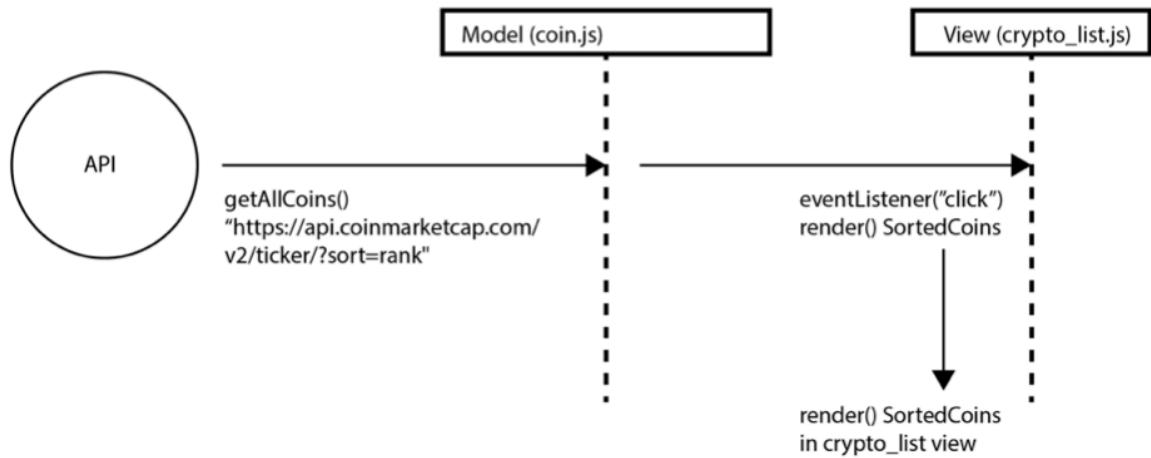
| <b>Unit</b> | <b>Ref</b> | <b>Evidence</b>                                                                   |  |
|-------------|------------|-----------------------------------------------------------------------------------|--|
| <b>P</b>    | P.7        | Produce two system interaction diagrams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

**Paste Screenshot here**



---

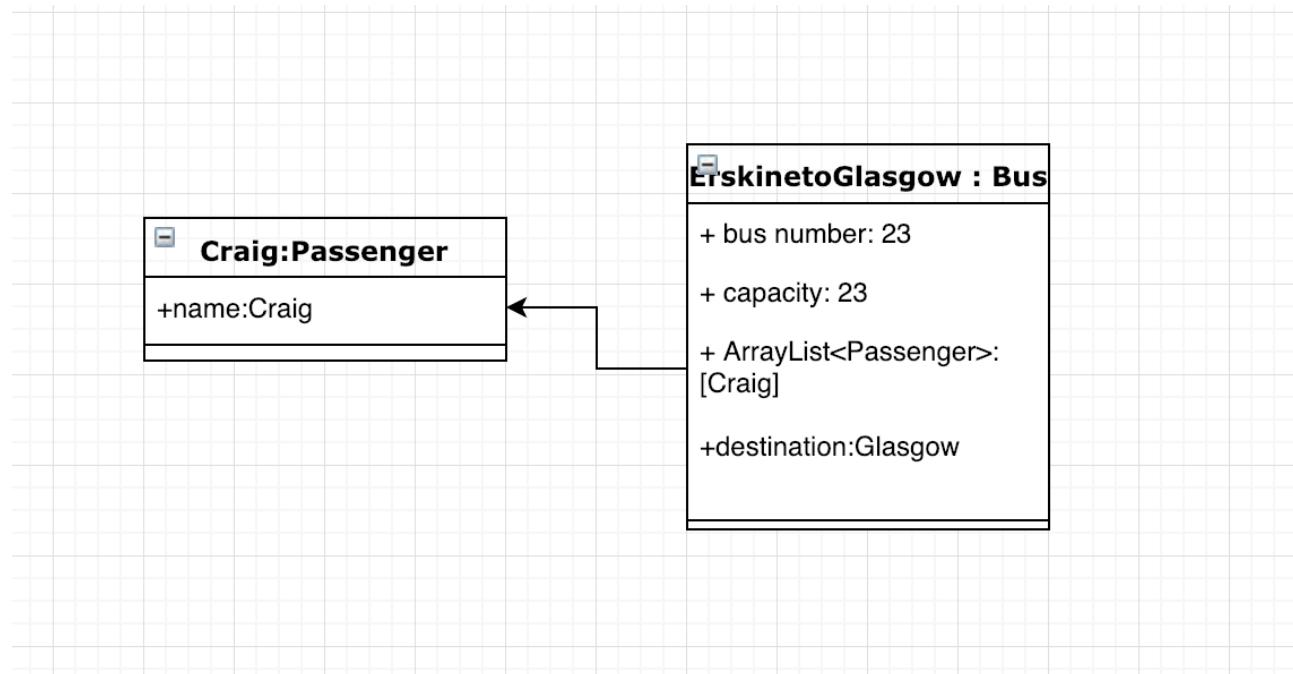
**FIGURE 5-COLLABORATION DIAGRAM TO DISPLAY PORTFOLIO IN CRYPTOTRACKER**

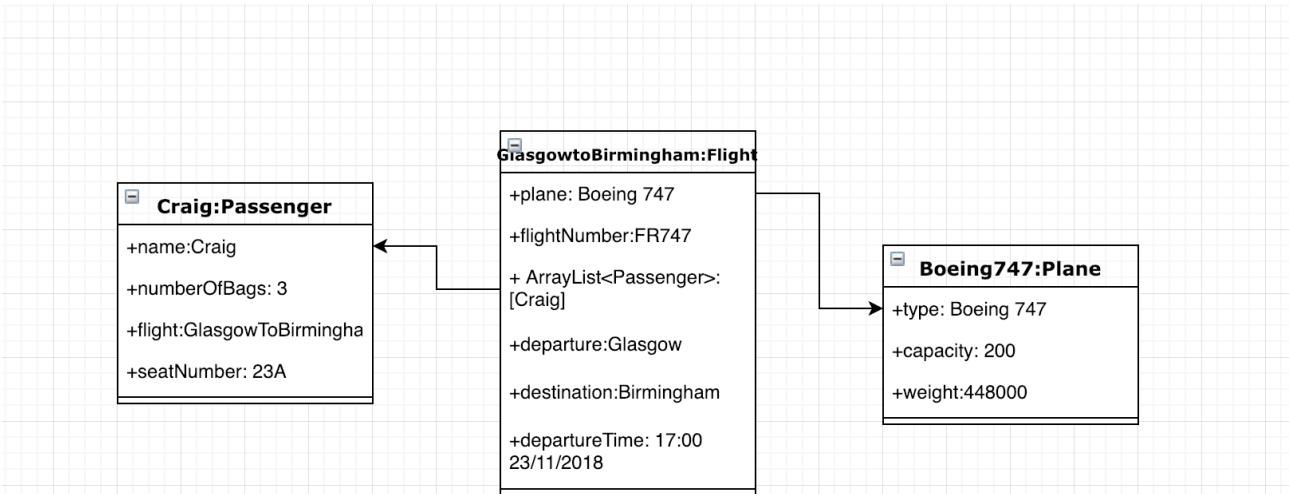


**FIGURE 6-SEQUENCE DIAGRAM TO SHOW HOW LIST OF CRYPTOCURRENCIES ARE RENDERED IN CRYPTOTRACKER**  
Description here

| Unit | Ref | Evidence                     |  |
|------|-----|------------------------------|--|
| P    | P.8 | Produce two object diagrams. |  |
|      |     | <b>Description:</b>          |  |

Paste Screenshot here






---

### **Description here**

| <b>Unit</b> | <b>Ref</b> | <b>Evidence</b>               |  |
|-------------|------------|-------------------------------|--|
| P           | P.17       | Produce a bug tracking report |  |
|             |            | <b>Description:</b>           |  |

### **Paste Screenshot here**

| Bug Identified                                                                                                                   | Solution                                                                                                            | Date        |
|----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|-------------|
| Portfolio not displaying on screen - waiting for data to be returned from multiple API calls, leading to asynchronous behaviour. | Declare 'async' function and add 'await' expression to resolve asynchronous behaviour.                              | 04 Nov 2018 |
| Portfolio not rendering when delete button is clicked - subscribe eventListener not receiving data.                              | Initiate top level function in App.js (entry point), triggering bindEvents() function to pass data to eventListener | 04 Nov 2018 |
| Webpage refreshes when submit buttons is clicked.                                                                                | Event trigger - added preventDefault() to resolve form submit issue.                                                | 02 Nov 2018 |
| SearchCoin() function is returning an 'undefined' value in the model                                                             | 'return' keyword is required outside of the loop statement                                                          | 02 Nov 2018 |
| Adding of API 'Price' values to existing database price results in a 'NaN'                                                       | API price require to be parsed to decimal, using parseFloat(Coin-Price)                                             | 02 Nov 2018 |

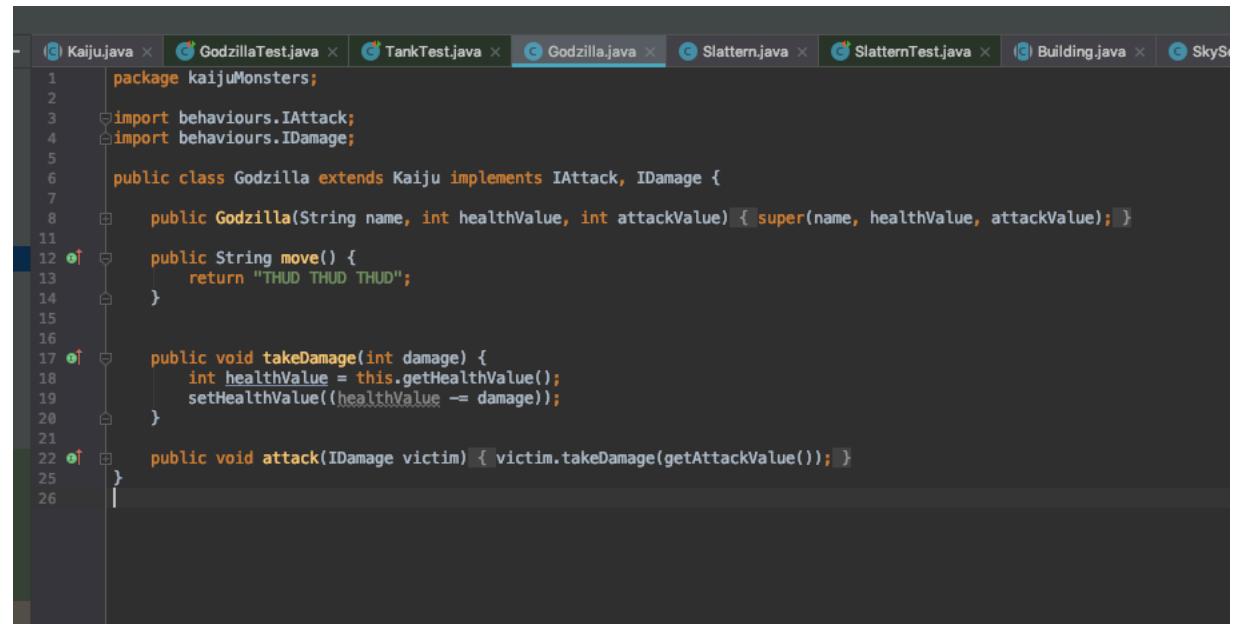
---

### **Description here**

### **Week 12**

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

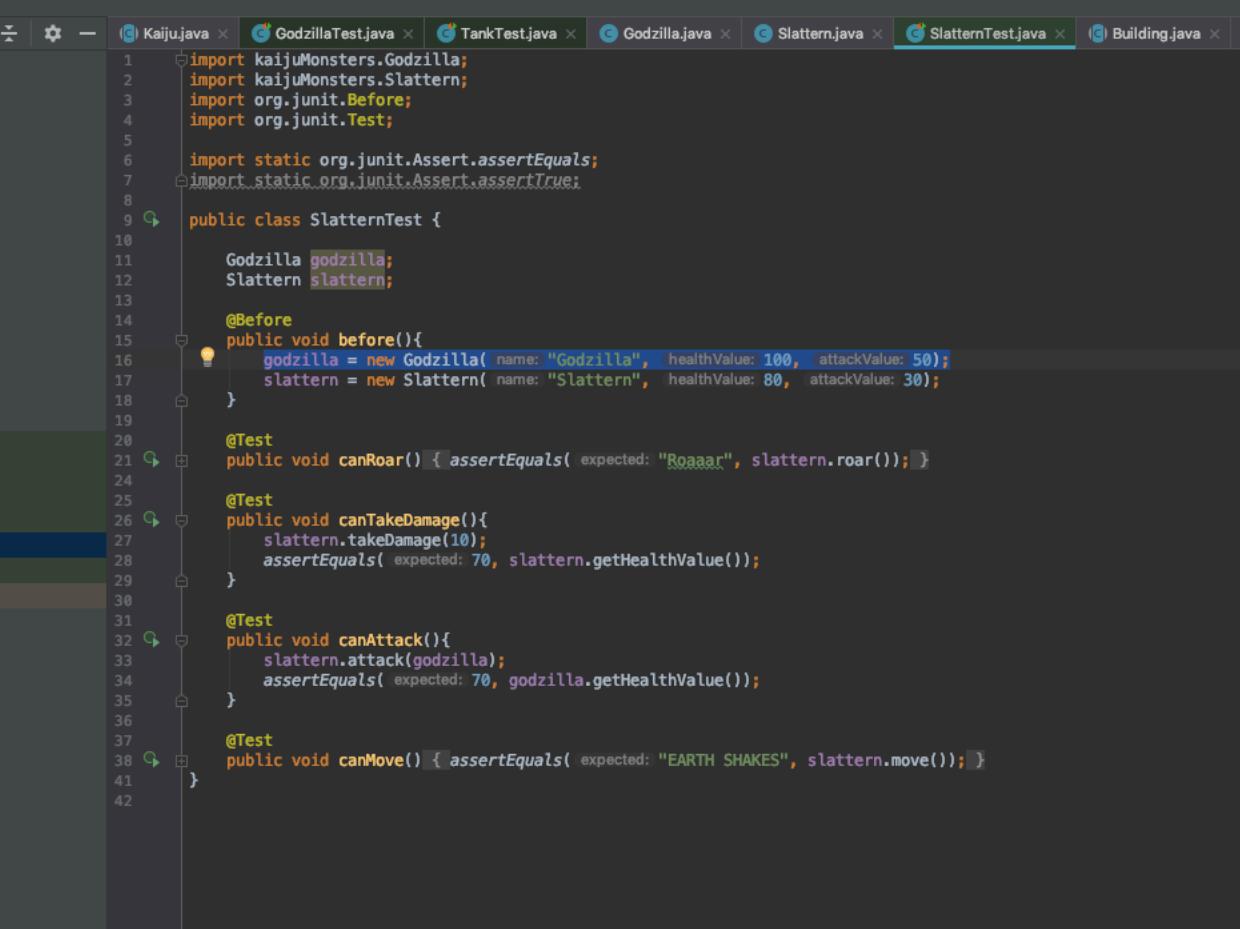
Paste Screenshot here



```

1 package kaijuMonsters;
2
3 import behaviours.IAttack;
4 import behaviours.IDamage;
5
6 public class Godzilla extends Kaiju implements IAttack, IDamage {
7
8 public Godzilla(String name, int healthValue, int attackValue) { super(name, healthValue, attackValue); }
9
10 public String move() {
11 return "THUD THUD THUD";
12 }
13
14 public void takeDamage(int damage) {
15 int healthValue = this.getHealthValue();
16 setHealthValue(healthValue - damage);
17 }
18
19 public void attack(IDamage victim) { victim.takeDamage(getAttackValue()); }
20
21}
22
23
24
25
26

```



```

1 import kaijuMonsters.Godzilla;
2 import kaijuMonsters.Slattern;
3 import org.junit.Before;
4 import org.junit.Test;
5
6 import static org.junit.Assert.assertEquals;
7 import static org.junit.Assert.assertTrue;
8
9 public class SlatternTest {
10
11 Godzilla godzilla;
12 Slattern slattern;
13
14 @Before
15 public void before(){
16 godzilla = new Godzilla(name: "Godzilla", healthValue: 100, attackValue: 50);
17 slattern = new Slattern(name: "Slattern", healthValue: 80, attackValue: 30);
18 }
19
20 @Test
21 public void canRoar() { assertEquals(expected: "Roaaaa", slattern.roar()); }
22
23 @Test
24 public void canTakeDamage(){
25 slattern.takeDamage(10);
26 assertEquals(expected: 70, slattern.getHealthValue());
27 }
28
29 @Test
30 public void canAttack(){
31 slattern.attack(godzilla);
32 assertEquals(expected: 70, godzilla.getHealthValue());
33 }
34
35 @Test
36 public void canMove() { assertEquals(expected: "EARTH SHAKES", slattern.move()); }
37
38 }
39
40
41

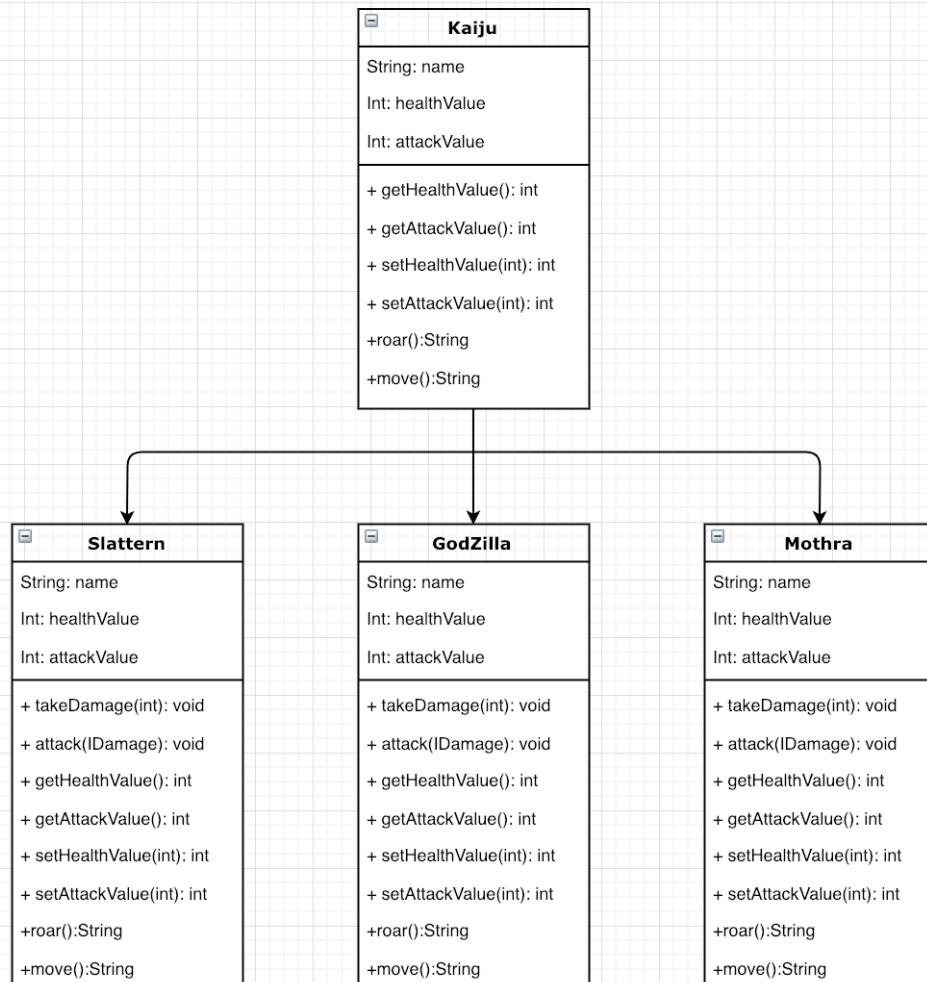
```

Description here

Godzilla is A Kaiju, Godzilla, IAttack and IDamage. The attack method takes in an IAttack type . As the Godzilla implements the IAttack interface it is able to be passed into the attack method.

| Unit | Ref   | Evidence                                          |  |
|------|-------|---------------------------------------------------|--|
| A&D  | A.D.5 | An Inheritance Diagram<br><br><b>Description:</b> |  |

Paste Screenshot here



Description here

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

Paste Screenshot here

```

1 package kaijuMonsters;
2
3 public abstract class Kaiju {
4
5 private String name;
6 private int healthValue;
7 private int attackValue;
8
9 public Kaiju(String name, int healthValue, int attackValue){
10 this.name = name;
11 this.healthValue = healthValue;
12 this.attackValue = attackValue;
13 }
14
15 public String roar() { return "Roaaar"; }
16
17 public int getHealthValue() {
18 return healthValue;
19 }
20
21 public void setHealthValue(int healthValue) {
22 this.healthValue = healthValue;
23 }
24
25 public int getAttackValue() {
26 return attackValue;
27 }
28
29 public void setAttackValue(int attackValue) {
30 this.attackValue = attackValue;
31 }
32
33 public abstract String move();
34
35 }
36
37
```

Description here

declaring variables as private to prevent it being accessed by other classes and writing public methods in order to access and edit the variables.

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

Paste Screenshot here

```
1 package kaijuMonsters;
2
3 public abstract class Kaiju {
4
5 private String name;
6 private int healthValue;
7 private int attackValue;
8
9 public Kaiju(String name, int healthValue, int attackValue){
10 this.name = name;
11 this.healthValue = healthValue;
12 this.attackValue = attackValue;
13 }
14
15 public String roar() { return "Roaaar"; }
16
17 public int getHealthValue() {
18 return healthValue;
19 }
20
21 public void setHealthValue(int healthValue) {
22 this.healthValue = healthValue;
23 }
24
25 public int getAttackValue() {
26 return attackValue;
27 }
28
29 public void setAttackValue(int attackValue) {
30 this.attackValue = attackValue;
31 }
32
33 public abstract String move();
34
35 }
36
37 }
```

FIGURE 7- A CLASS

```
1 package kaijuMonsters;
2
3 import behaviours.IAttack;
4 import behaviours.IDamage;
5
6 public class Godzilla extends Kaiju implements IAttack, IDamage {
7
8 public Godzilla(String name, int healthValue, int attackValue) { super(name, healthValue, attackValue); }
9
10 public String move() {
11 return "THUD THUD THUD";
12 }
13
14
15 public void takeDamage(int damage) {
16 int healthValue = this.getHealthValue();
17 setHealthValue((healthValue -= damage));
18 }
19
20 public void attack(IDamage victim) { victim.takeDamage(getAttackValue()); }
21
22 }
23
24 }
```

FIGURE 8- GODZILLA CLASS INHERITS FROM KAIJU

```
public class GodzillaTest {
 Godzilla godzilla;
 Slattern slattern;

 @Before
 public void before(){
 godzilla = new Godzilla(name: "Godzilla", healthValue: 100, attackValue: 50);
 slattern = new Slattern(name: "Slattern", healthValue: 80, attackValue: 30);
 }
}
```

FIGURE 9- GODZILLA OBJECT

```

import static org.junit.Assert.assertEquals;

public class GodzillaTest {

 Godzilla godzilla;
 Slattern slattern;

 @Before
 public void before(){
 godzilla = new Godzilla(name: "Godzilla", healthValue: 100, attackValue: 50);
 slattern = new Slattern(name: "Slattern", healthValue: 80, attackValue: 30);
 }

 @Test
 public void canRoar() { assertEquals(expected: "Roaaar", godzilla.roar()); }

 @Test
 public void canTakeDamage(){
 godzilla.takeDamage(10);
 assertEquals(expected: 90, godzilla.getHealthValue());
 }

 @Test
 public void canAttack(){
 godzilla.attack(slattern);
 assertEquals(expected: 30, slattern.getHealthValue());
 }

 @Test
 public void canMove() { assertEquals(expected: "THUD THUD THUD", godzilla.move()); }
}

```

**FIGURE 10- METHOD USING INFORMATION INHERITED FROM ANOTHER CLASS**  
Description here

| 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. |
|      |     | <b>Description:</b>                                                                                                                                                   |

Paste Screenshot here

```

public int generateSeatNumber(){
 int seatNumber = generateRandom();
 while(this.bookedSeats.contains(seatNumber)){
 seatNumber = generateRandom();
 }
 this.bookedSeats.add(seatNumber);
 return seatNumber;
}

public void addPassenger(Passenger passenger1) {
 if(availableSeats()>0) {
 this.passengers.add(passenger1);
 this.availableSeats -= 1;
 passenger1.setSeatNumber(generateSeatNumber());
 }
}

```

**Generate Seat number function will assign the seat number a random number and then will run a while loop to check whether the seat number already exists in the booked seats array, if it does then it will assign a new random number and repeat the while loop until it assigns a number which doesn't exist in the booked seats array and then add this to the booked seats array and return the seat number.**

```

Coin.prototype.individualCoinPriceData = function (symbol) {
 const individualCoinData = new
 Request(`https://www.alphavantage.co/query?function=DIGITAL_CURRENCY_DAILY&symbol=${symbol}&market=USD&apikey=SZGMIHDEPWLBE9NI`);

 const todaysPrice = new
 Request(`https://www.alphavantage.co/query?function=CURRENCY_EXCHANGE_RATE&from_currency=${symbol}&to_currency=USD&apikey=SZGMIHDEPWLBE9NI`);

 const realTimeInfo = todaysPrice.get().then((data)=>{
 const realtimeInfo = {
 name: data["Realtime Currency Exchange Rate"]["2. From_Currency Name"],
 symbol: data["Realtime Currency Exchange Rate"]["1. From_Currency Code"] ,
 date: data["Realtime Currency Exchange Rate"]["6. Last Refreshed"],
 close: data["Realtime Currency Exchange Rate"]["5. Exchange Rate"] ,
 }
 return realtimeInfo;
 })

 realTimeInfo.then((data) => {
 this.realtime = data;
 individualCoinData.get().then((data) => {
 this.singleCoinData = data["Time Series (Digital Currency Daily)"]
 this.singleCoinResult = data
 const dates = Object.keys(this.singleCoinData)
 dateInfo = []
 dates.forEach((date) => {
 const closePrice = this.singleCoinData[date]["4b. close (USD)"]
 const info = {
 name: this.singleCoinResult["Meta Data"]["3. Digital Currency Name"],
 symbol: this.singleCoinResult["Meta Data"]["2. Digital Currency Code"],
 date: date,
 close: closePrice
 };
 dateInfo.push(info)
 })
 dateInfo.unshift(this.realtime)
 console.log(dateInfo);
 PubSub.publish("coin:chosen-coin-price-History",dateInfo)
 })
 })
}

```

**This function combines the information from two different API requests. It initially creates an object for the real time information for the coin and then does another API request to create the historic pricing data of the coin. It will then push the real-time information to the start of the array list and publish it out.**

**Description here**