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	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:

Paste Screenshot h

Description here

```ruby
def pets_by_breed(shop, breed)
  types = shop[:pets]
  breed_array = []

  for type in types
    if type[:breed] == breed
      breed_array.push(type)
    end
  end
  return breed_array
end
```

```ruby
def test_all_pets_by_breed__found
  pets = pets_by_breed(@pet_shop, "British Shorthair")
  assert_equal(2, pets.count)
end

def test_all_pets_by_breed__not_found
  pets = pets_by_breed(@pet_shop, "Dalmation")
  assert_equal(0, pets.count)
end
```

```
➜ Downloads pet_shop_start_point
➜ pet_shop_start_point git:(master) atom .
➜ pet_shop_start_point git:(master) ruby specs/pet_shop_spec.rb
Run options: --seed 16821

# Running:

.................

Finished in 0.001832s, 9279.4761 runs/s, 9279.4761 assertions/s.

17 runs, 17 assertions, 0 failures, 0 errors, 0 skips
➜ pet_shop_start_point git:(master)
```

We are defining the pets by breed function with the parameter “shop” and “breed” then stating that the breed array is equal to an empty array. In the second part we are creating a for loop and stating that if the breed value within the type hash is equal to breed then add a breed type before returning the breed.

In the test we are testing whether all pets by breed found are equal to 2 and breeds not found

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: The hash is within the pets array, the function is demonstrating a for loop which goes over each hash within the array and returns each pet name value within each hash.

Paste Screenshot here

```ruby
for pet in pets
  p pet[:name]
end

pets.each { |pet| p pet[:name]}
```

```ruby
pets = [
{
    name: "Sir Percy",
    pet_type: :cat,
    breed: "British Shorthair",
    price: 500
},
```

```
➜  pet_shop_start_point git:(master) ruby specs/pet_shop_spec.rb
Run options: --seed 31736

# Running:

.................

Finished in 0.002553s, 6658.8324 runs/s, 6658.8324 assertions/s.

17 runs, 17 assertions, 0 failures, 0 errors, 0 skips
➜  pet_shop_start_point git:(master)
```

Description here

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 update() method can be called to change the name of an existing customer in the cinema database. The sql in the method allows you to change the name of a customer and save it to the database.

Paste Screenshot here

Description here

```
def update()

  sql = "UPDATE customers SET(customer_name, funds) = ($1, $2)
  WHERE id = $3"

  values = [@customer_name, @funds, @id]

  SqlRunner.run(sql, values)
end
```

```
=> #<Customer:0x007faf4d89e3c0 @customer_name="Cristiano", @funds=100, @id=41>
[29] pry(main)>
```

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: The following function allows you to order the films by price via SQL.

Paste Screenshot here

Description here

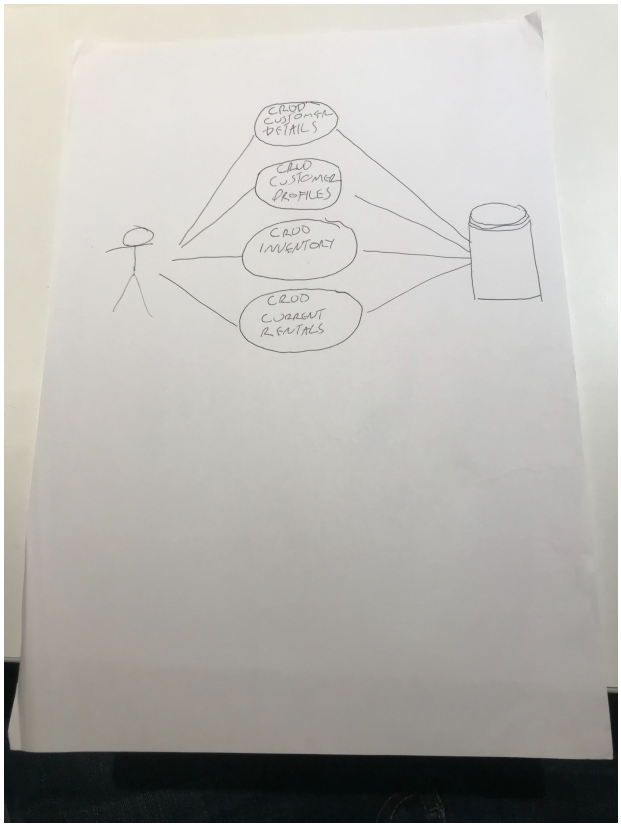
```
def self.order_by_price()
  sql = "SELECT * FROM films ORDER BY price DESC;"
  films = SqlRunner.run(sql)
  return films.map{|film| Film.new(film)}
```

```
[1] pry(main)> Film.order_by_price()
=> [#<Film:0x007f89bebdbb28 @id=8, @price=10, @title="Spider-Man">,
 #<Film:0x007f89bebdba60 @id=7, @price=7, @title="Inception">]
[2] pry(main)>
```

Week 5 and 6

Unit	Ref	Evidence
A&D	A.D.1	A Use Case Diagram
		Description: The diagram shows the user functionality within the app. The user should be able to use the CRUD(create, ready, update, delete) methods within each section of the app.

Paste Screenshot here

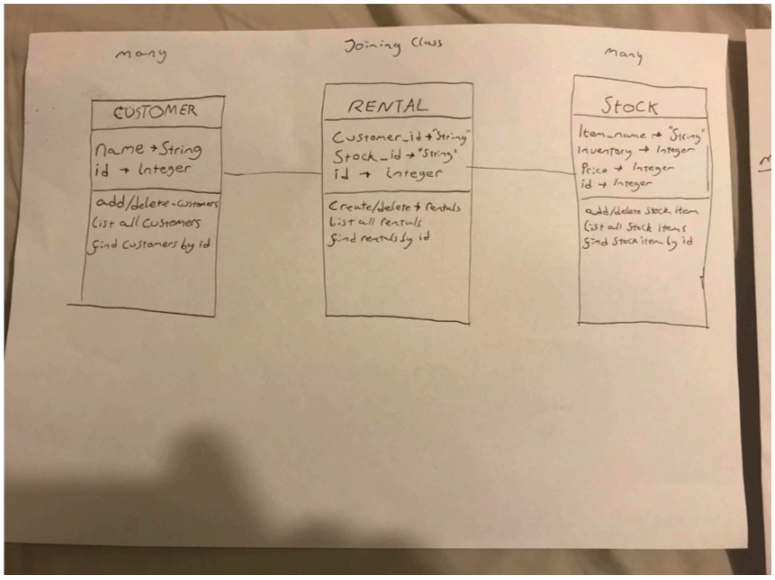


Description here

Unit	Ref	Evidence
A&D	A.D.2	A Class Diagram
		Description: The diagram shows the three classes within my rental system app and represents the many to many relationship with the rental class linking the customer and stock class together.

Paste Screenshot here

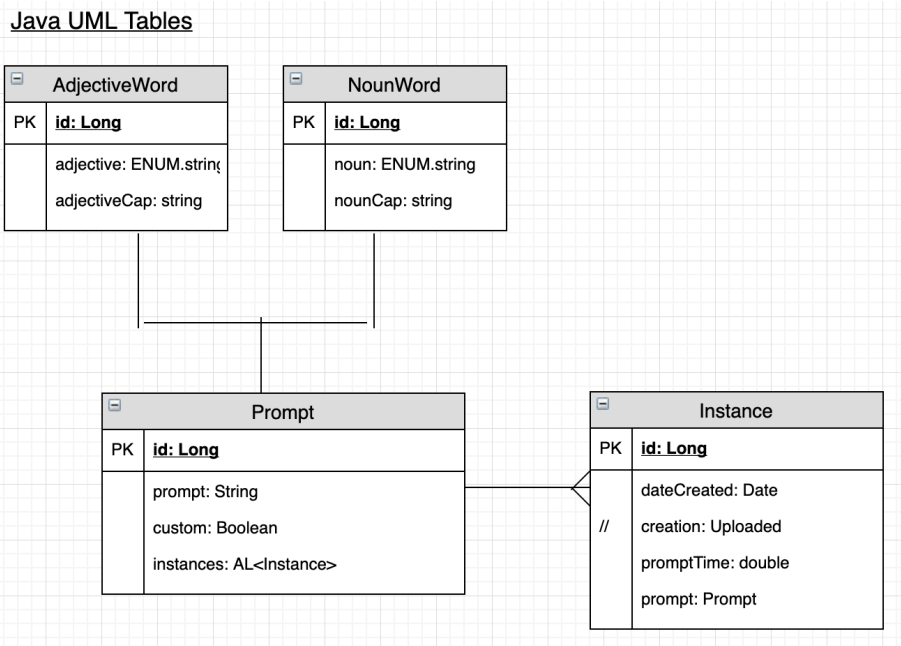
Description here



Unit	Ref	Evidence
A&D	A.D.3	An Object Diagram
		Description: Object diagram display the relationship between the classes and the data flow.

Paste Screenshot here

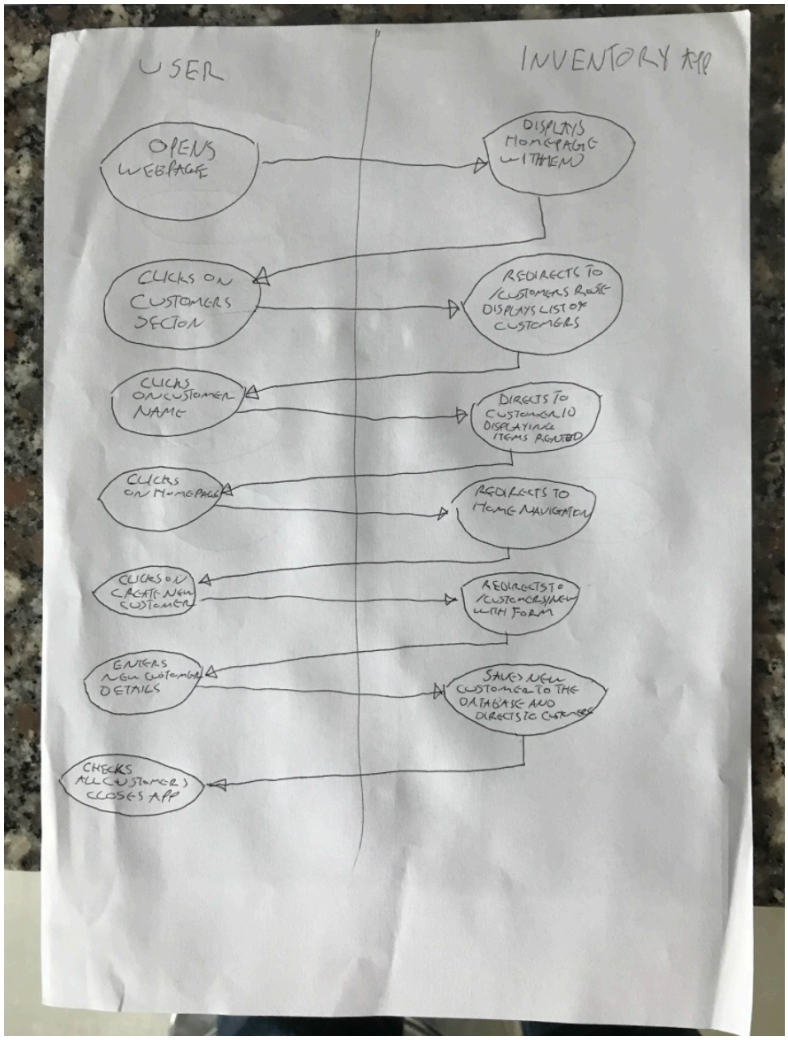
Description here



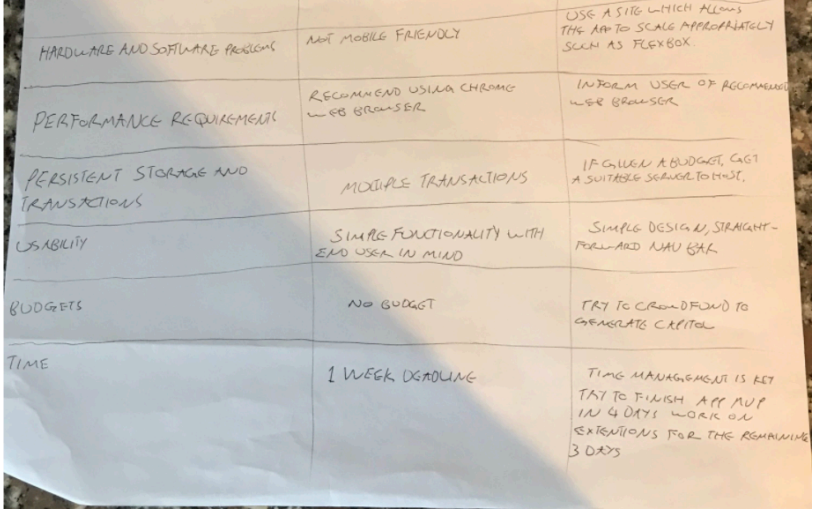
Unit	Ref	Evidence
A&D	A.D.4	An Activity Diagram
		Description: The diagram below demonstrates the interaction between user and app from when the user opens the application to closing.

Paste Screenshot here

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: The plan below demonstrates a plan with potential solutions given the restraint when building the app.



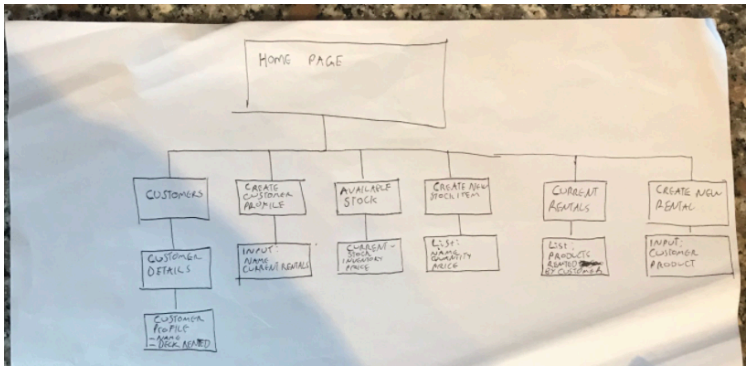
Paste Screenshot here

Description here

Unit	Ref	Evidence
P	P.5	User Site Map
		Description: Illustrates the pages on the Homepage and steps to get to each page.

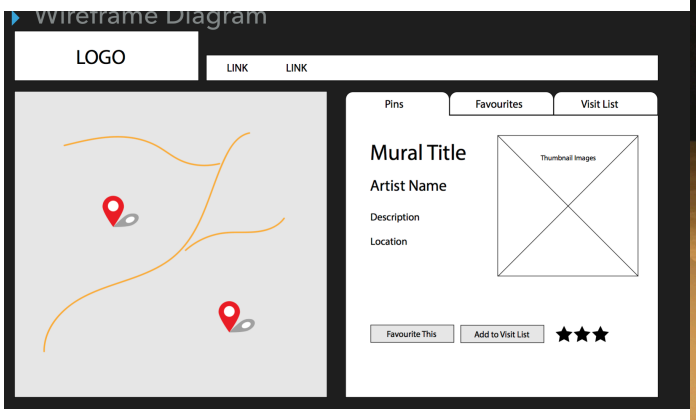
Paste Screenshot here

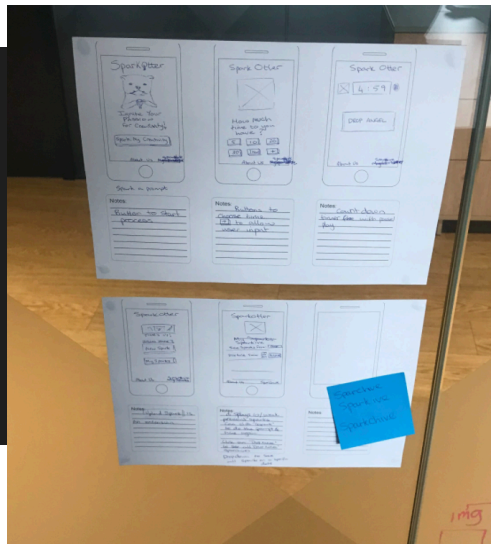
Description here



Unit	Ref	Evidence
P	P.6	2 Wireframe Diagrams
		Description: Diagram on the left is from the 'Paint the Toon' app which displays a map on the left with a nav bar at the top and details of each site on the right. Diagram on the right depicts our mobile app 'SparkOtter' which each page displayed with what content each page will display.

Paste Screenshot here





Description here

Unit	Ref	Evidence
P	P.10	Example of Pseudocode used for a method
		Description: Code is explaining what is happening within the save function and the steps taken within the function.

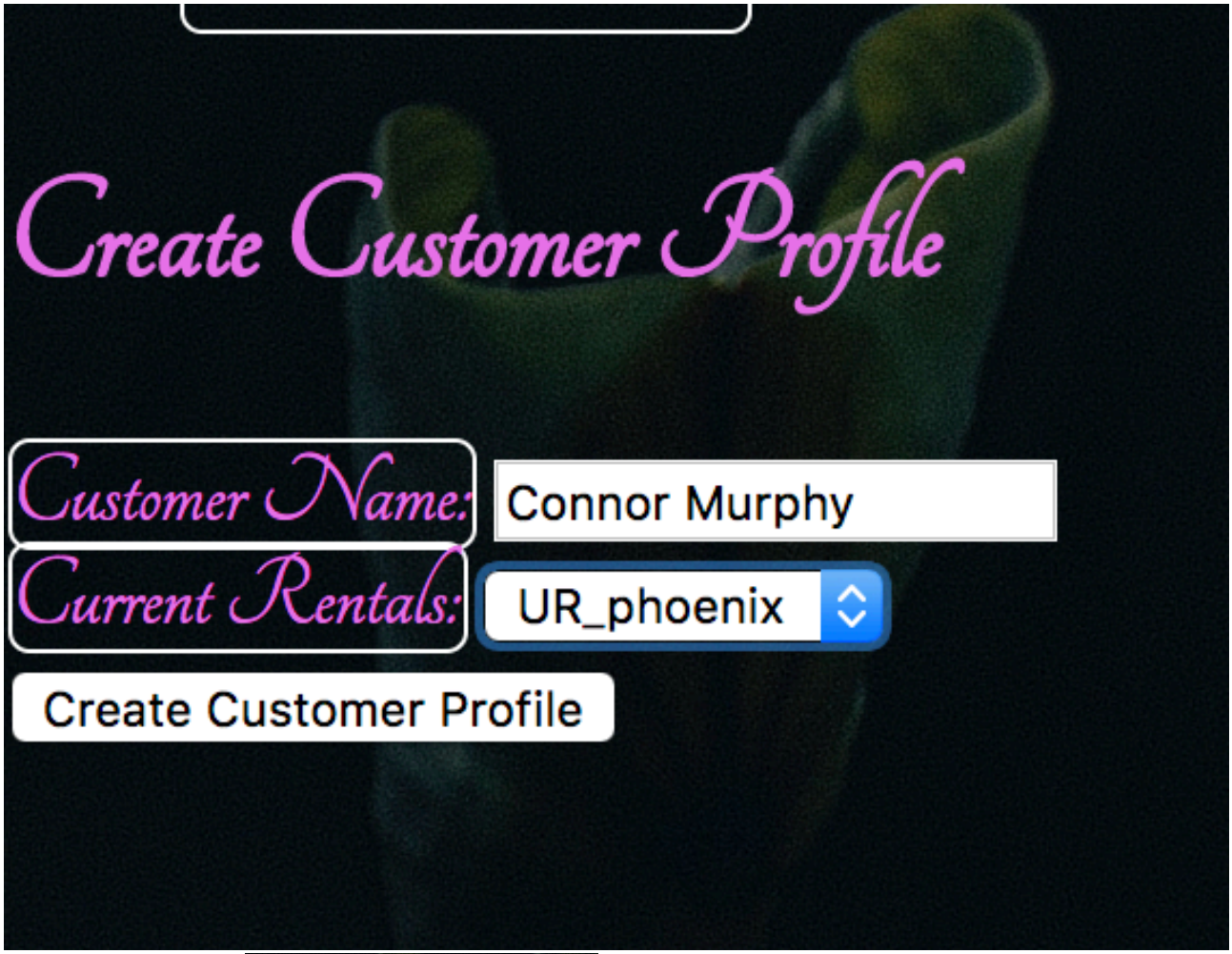
```
def save()
  sql = "INSERT INTO rentals
  (customer_id, stock_id) VALUES ($1, $2) RETURNING id"
  values = [@customer_id, @stock_id]
  results = SqlRunner.run(sql, values)
  @id = results.first()['id'].to_i
  stock = Stock.find(@stock_id)
  stock.remove_inventory()
  stock.update()

  # Find the stock by stock_id
  # Call a method no the stock to reduce Inventory
  # Update that stock
end
```

Paste Screenshot here

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:The user is creating a new customer profile displaying their name what item they are renting. The app redirects to the customer details page which displays the new customer created.

Paste Screenshot here



Description

here

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 save method is inputting saved customer and stock details into the database which is returning a list of all customers including the new customer saved.

Paste Screenshot here

```
def save()
  sql = "INSERT INTO rentals
  (customer_id, stock_id) VALUES ($1, $2) RETURNING id"
  values = [@customer_id, @stock_id]
  results = SqlRunner.run(sql, values)
  @id = results.first()['id'].to_i
  stock = Stock.find(@stock_id)
  stock.remove_inventory()
  stock.update()
```

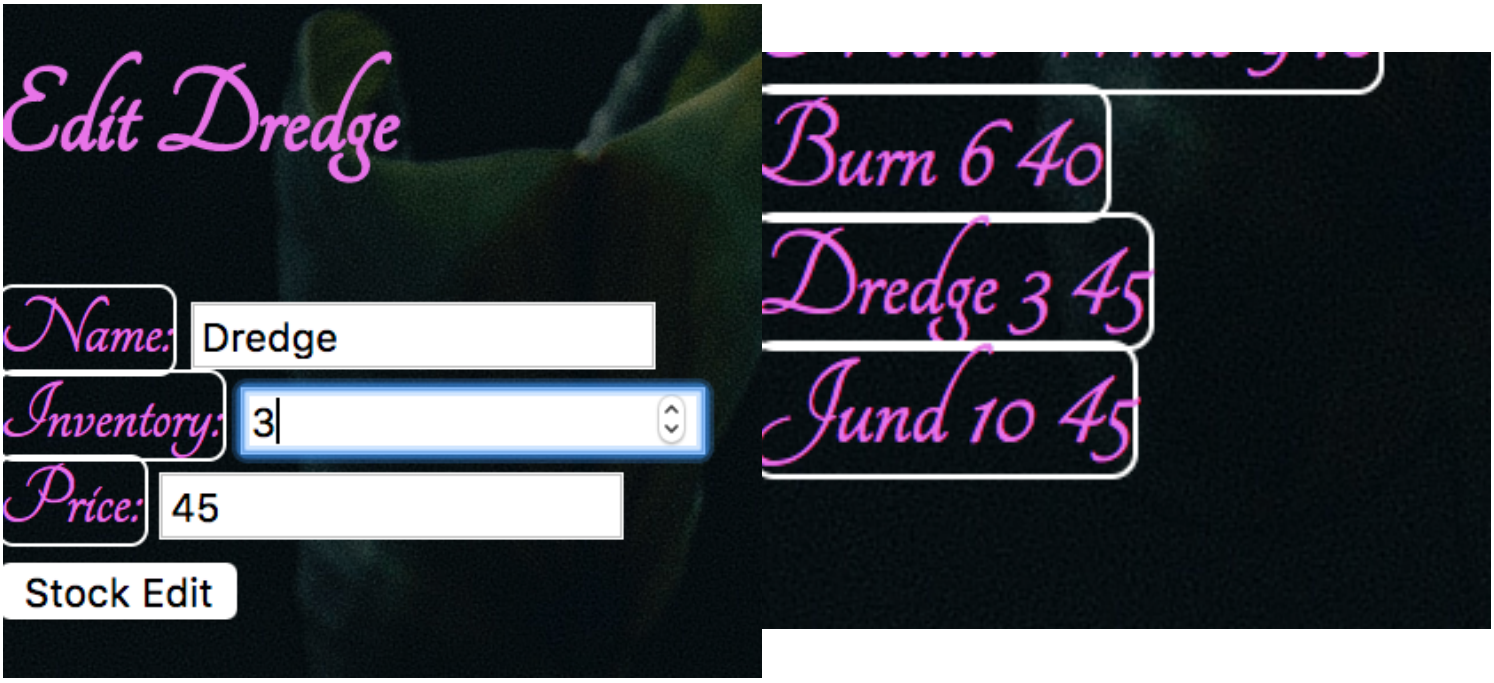
```
rental_system=# SELECT * FROM rentals;
 id | customer_id | stock_id
----+-------------+----------
 86 |          63 |       51
 87 |          64 |       49
 88 |          65 |       51
 89 |          63 |       49
 90 |          64 |       56
 91 |          63 |       53
 92 |          63 |       53
(7 rows)
```

Description here

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 user is wishing to update their inventory of a particular stock item. They have changed the price and quantity of the stock item which is then saved and updated accordingly.

Paste Screenshot here

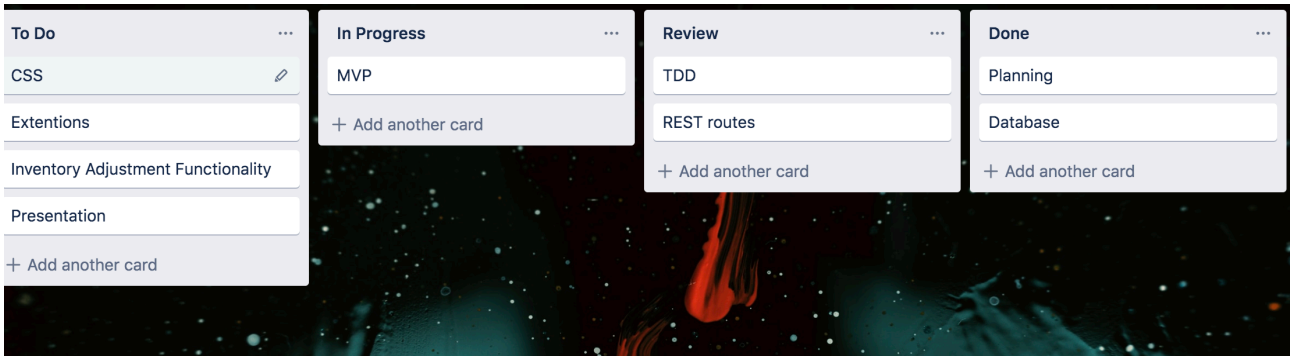
Description here



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/Martinporter13/Project-1

Paste Screenshot here

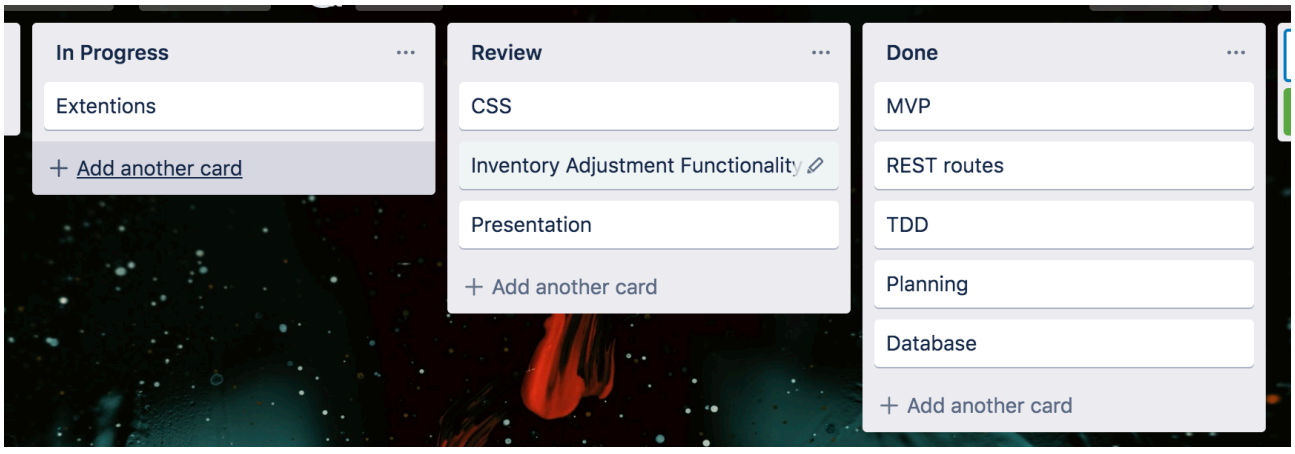
Description here

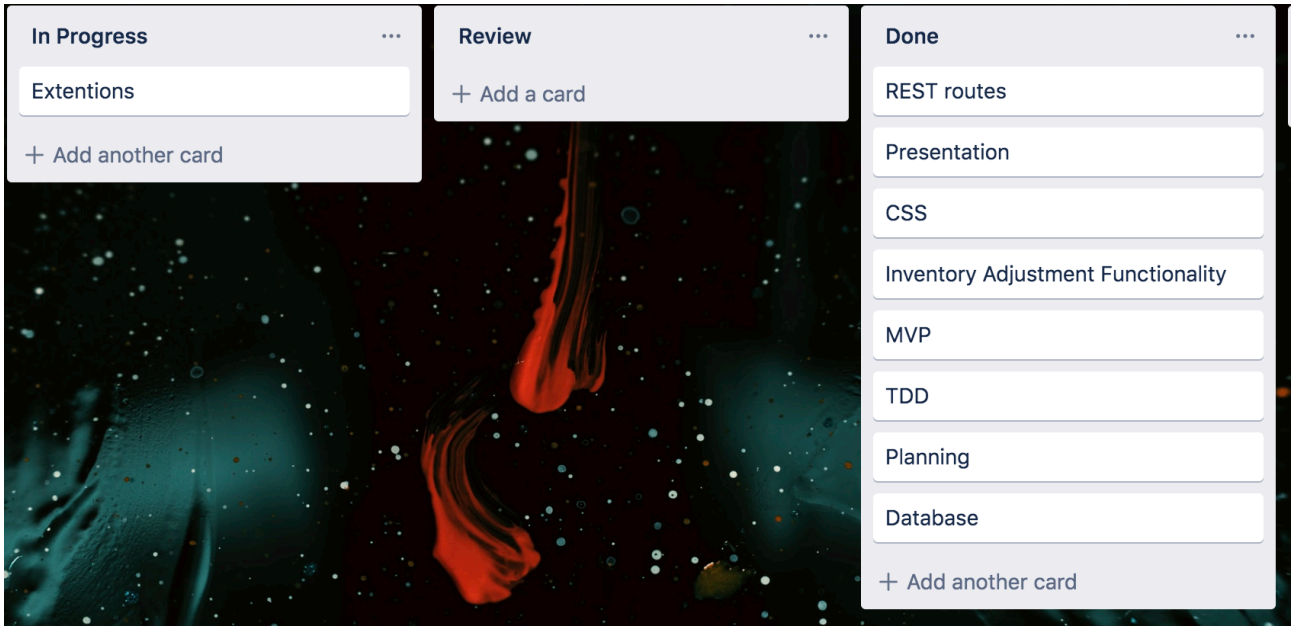


Unit	Ref	Evidence
P	P.12	Take screenshots or photos of your planning and the different stages of development to show changes.
		Description: The following Trello board displays the progress made throughout the project.

Paste Screenshot here

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: The first screenshot displays the url of where the API data is being taken from and the next shot shows the information being displayed on the browser.

```
    <div>
      <h3>Title: {{thread.data.title}}</h3>
      <p>Subreddit: {{thread.data.subreddit}}</p>
      <p>Author: {{thread.data.author}}</p>
      <p>URL: {{thread.data.url}}</p>
    </div>
</template>

<script>
export default {
```

Paste Screenshot here

```
      threads: [],
      selectedThread: null
    };
  },
  mounted(){
    fetch('https://www.reddit.com/r/ScottishFootball.json')
    .then(res => res.json())
    .then(data => this.threads = data)

  eventBus.$on('thread-selected', (thread) =>{
    this.selectedThread = thread;
    })
  },
  components: {
```

Description here

Scottish Football Reddit Threads

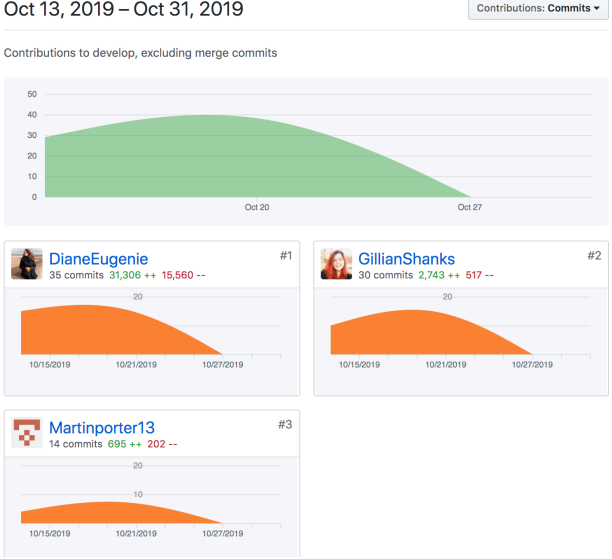
- /r/Scottishfootball improvement project
- Club Statement: Craig Levein | Heart Of Midlothian Football Club
- “Levein Out” sprayed across club crest outside Tynecastle
- Craig Levein Sacked
- Twitter: Rangers fans on the bus home last night singing vile song about Scott Brown's sister and Tommy B
- Budge
- Sir Alex Ferguson officially opens Aberdeen's Cormack Park training facility
- Aberdeen FC Cormack Park Officially Open!
- The sack race
- Paddy Power view on the old "my nan could make It in that leauge" trope.
- Gerrard & Defoe let Morelos know it's time to come in for the night
- Id pay the severance fee myself if I could...
- St Johnstone 1-0 Hearts
- This question came up on The Chase today
- Michael Stewart seems pleased...
- Ross County 0-4 Rangers
- Celtic hold ‘emergency talks’ with Lazio and UEFA | Allan McGregor a semi final doubt
- James Forrest signs new 4 year deal.
- Celtic 2-0 St Mirren
- Motherwell 2-1 Kilmarnock
- Hibernian 2-2 Livingston
- Hexit delayed again
- Hibernian Vs Livingston
- Dundee United owner Mark Ogren intends to continue backing the club even if they fail to reach the Scottis
- Celtic and Lazio charged by UEFA
- Hamilton 0-1 Aberdeen

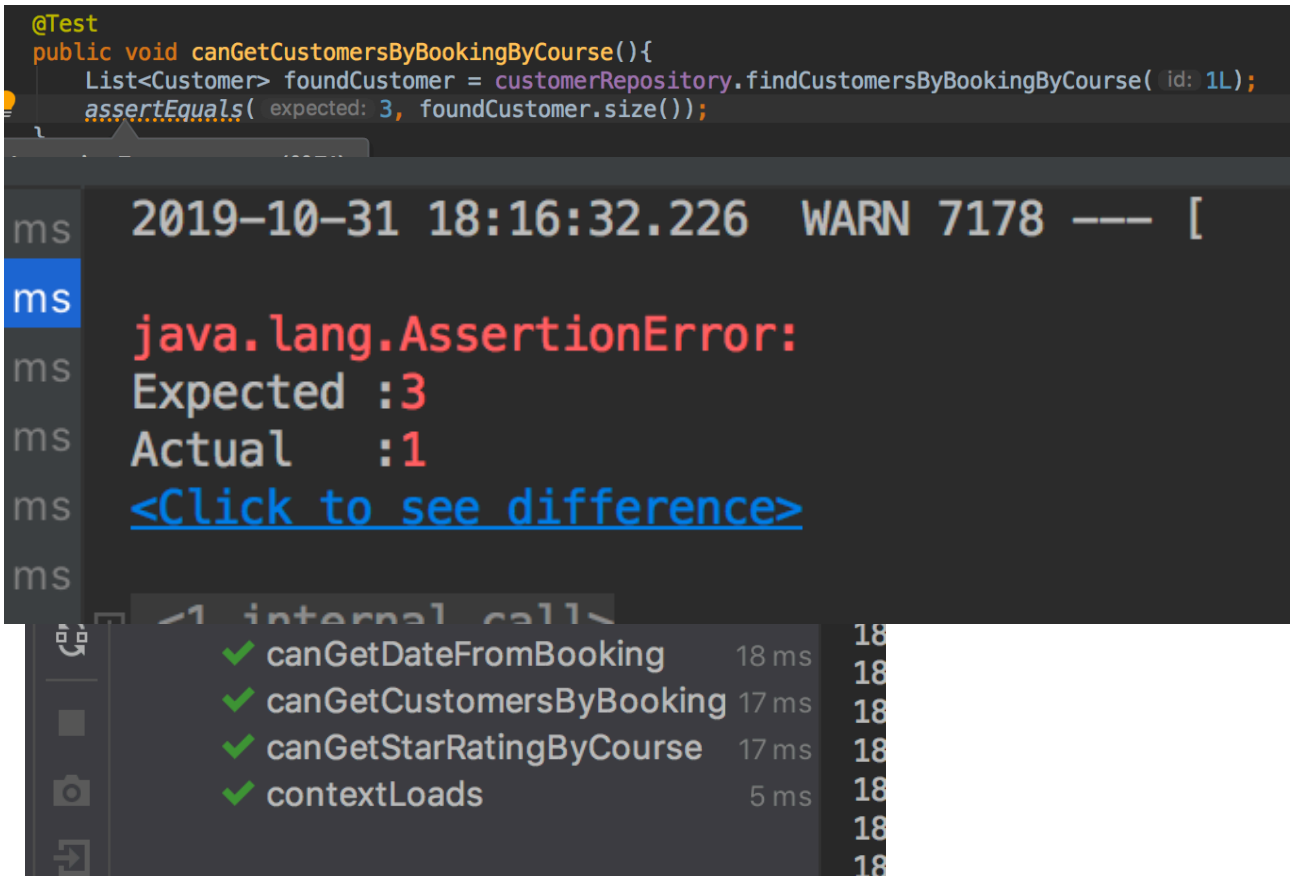
Title: Aberdeen FC Cormack Park Officially Open!

Subreddit: ScottishFootball

Author: ISD1982

URL : https://www.reddit.com/r/ScottishFootball/comments/dppf7i/aberdeen_fc_cormack_park_officially_open/





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: Test failed because the wrong unit was input under assertEquals, changed the unit from 3 to 1 and got the test passing as a result

Paste Screenshot here

Unit	Ref	Evidence
P	P.2	Take a screenshot of the project brief from your group project.
		Description: Please see below the project brief for SparkOtter

Paste Screenshot here

Description here

We want to create an app that generates a phrase to help inspire creatives based on words generated and allow them to revisit it.

MVP:
Make an API of Random words.
Allow user to select a countdown time.
Display 1 prompt phrase.
Present countdown.
Save prompt, countdown time & date/time with a filter of what to view.
Containers are: Main, about page & saved prompts.
Mobile Friendly.

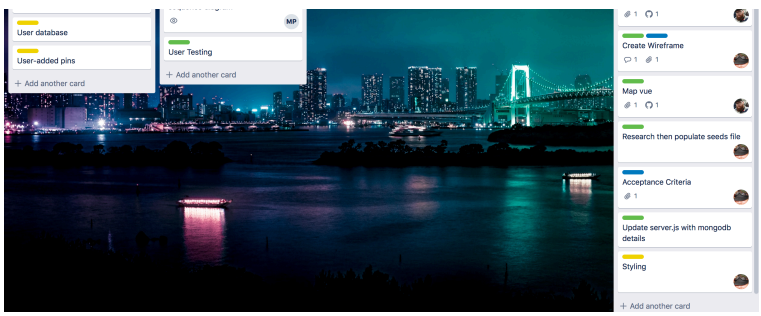
Extensions:

Canvas allow user to draw.
Create personal prompts.
Categorise prompt for selection by user.
Creating a shareable link and share via email, twitter, insta etc.
Potential public server of everyone's work.
Other languages.
Daily prompt via email.

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

Paste Screenshot here

Description here



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

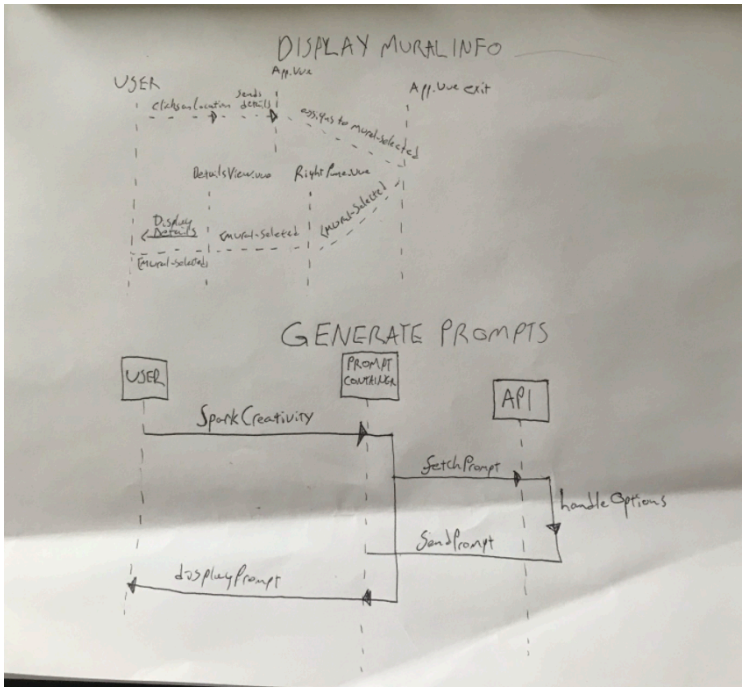
Paste Screenshot here

Description here

Acceptance Criteria

Stage: Initial Planning - Thursday 14/12/17

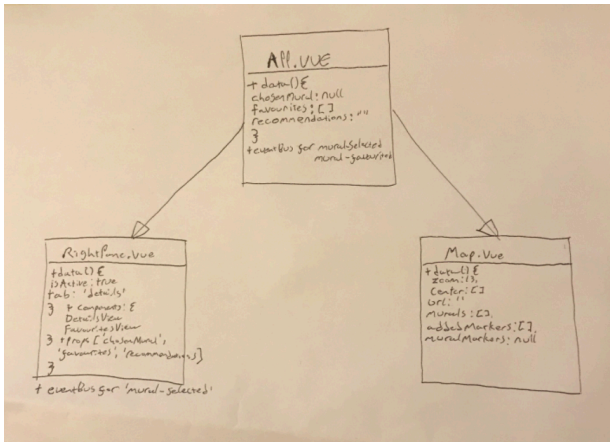
Acceptance Criteria	Expected Result/Output	Pass / Fail
A user is able to click on a pin placed on a map.	The pin will show the user the relevant mural and details when clicked.	
A user is able to favourite mural to their own list.	The user's favourites list will show murals that have been favourited when the button within the pin details is clicked.	
A user is able to rate a mural out of 5 stars.	The user can set a rating of one to five by clicking on the star in the pin details, updating the db.	
A user is able to write recommendations of places nearby the mural.	The user can submit their own recommendations for each mural, they will be added to the db so it can be viewed alongside the details.	



Unit	Ref	Evidence
P	P.7	Produce two system interaction diagrams (sequence and/or collaboration diagrams).
		Description: The top diagram is a sequence diagram demonstrating what happens when a user clicks on an icon on the map from my group 'Paint the Toon' project which contained a map. When the user clicks on a marker it displays the information of the mural in the rightPane section of the site. When the location is clicked on, the program sends the coordinates via eventBus which is assigned to mural-selected in App.vue before being passed on to RightPane.vue which sends the details over to DetailsView.vue which in turn displays relevant information about the murals. **The second diagram is a collaboration diagram from my group Java project 'SparkOtter'. The diagram shows what happens when a user clicks on the 'Spark Creativity' button. The prompt container calls on a fetch prompt method to get relevant information from the API which passes on relevant information and sends it back to the prompt container which then displays the prompts to the user.**

Paste Screenshot here

Description here



Unit	Ref	Evidence
P	P.8	Produce two object diagrams.
		Description: The diagram above shows the relationship regarding the data flow in my group Javascript project 'Paint the Toon'.

Paste Screenshot here

Description here

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

Paste Screenshot here

Bug/Error	Solution	Date
Double clicking map zooms in as well as adding a point to it.	Added logic to map.vue to override this feature.	10/09/2019
Link to images broken when trying to display in the details component.	Images moved from assets folder to hosted folder online. URLs in the seeds folder amended to point there.	09/09/2019
Details not automatically showing when pin is clicked.	Adding event bus to RightPane.vue.	09/09/2019
Clicking on pin no longer switches to details view.	V-if statement added to component.	10/09/2019
Typing in one text field in the Favourites tab will populate all of them.		
Spelling errors and duplicate coordinates in db.	Changes made to correct errors.	11/10/2019
Deleting objects from the favourite list only working on the front-end. Db object remains.		

Description here

Week 12

Unit	Ref	Evidence
I&T	I.T.7	The use of Polymorphism in a program and what it is doing.
		Description: The screenshots below showing a music shop app taking in a variety of instruments. The shop class takes in an array list of items (stock) which implements the ISell interface. The method is removing stock by item to see that items can be added or removed from the array.

Paste Screenshot here

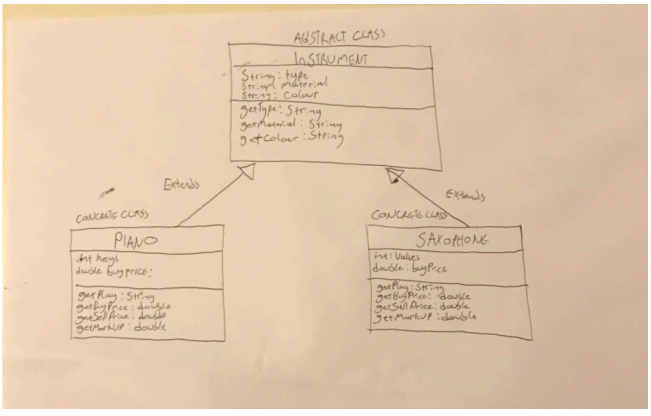
```
public class ShopTest {

    Shop shop;
    Guitar guitar;
    Piano piano;
    Saxophone saxophone;

    @Before
    public void before(){
        shop = new Shop( name: "RaysMusicShop");
        guitar = new Guitar( type: "electric",  materia
        piano = new Piano( type: "keyboard",  material:
        saxophone = new Saxophone( type: "the best typ
    }

    @Test
    public void canGetName() { assertEquals( expected:

    @Test
    public void stockStartsEmpty() { assertEquals( ex

    @Test
    public void canAddGuitar(){
        shop.addStockItem(guitar);
```

Description here

```
public class Shop {

    private String name;
    private ArrayList<ISell> stock;

    public Shop(String name){
        this.stock = new ArrayList<ISell>();
        this.name = name;
    }

    public String getName(){
        return name;
    }
```

```
public void removeStockByItem(ISell item){
    stock.remove(item);
}
```

```
@Test
public void canRemoveStockByItem(){
    shop.addStockItem(guitar);
    shop.addStockItem(piano);
    shop.addStockItem(saxophone);
    shop.removeStockByItem(piano);
    assertEquals( expected: 2, shop.stockCount());
}
```



Unit	Ref	Evidence
A&D	A.D.5	An Inheritance Diagram
		Description: The instrument class as shown is the abstract class and the saxophone and Piano are the classes which inherit from the abstract class.

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.
		Description: Encapsulation is a key concept in object-oriented programming. It illustrates the concept of bundling data and methods that work on that data within one unit eg a class. It is often used to hide the state of an object from outside. The getters are used so that data can be accessed from outside the class. The setter is so that the data can be set from outside the class.

Paste Screenshot here

Description here

```
public class Customer {

    private String name;
    private int budget;
    private ArrayList<Car> purchases;

    public Customer(String name, int budget){
        this.name = name;
        this.budget = budget;
        purchases = new ArrayList<Car>();

    }

    public String getName() {
        return name;
    }

    public int getBudget() {
        return budget;
    }

    public int getPurchaseCount(){
        return this.purchases.size();
    }

    public void addPurchase(Car car){
        this.purchases.add(car);
    }

    public void setBudget(int budget) {
        this.budget = budget;
    }
}
```

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: The example below is that of a theme park which uses inheritance. The abstract class is attractions In which dodgems inherits from and gains certain properties seen in super() in public Dodgems().

Paste Screenshot here

Description here

```java
@Override
public double defaultPrice() {
    return 4.50;
}
```

```java
@Override
public double priceFor(Visitor visitor) {
    if (visitor.getAge() < 12) {
        return this.defaultPrice() / 2;
    }
    return this.defaultPrice();
}
```

```java
import behaviours.IReviewed;
public abstract class Attraction implements IReviewed {
    private String name;
    private int rating;
    private int visitCount;

    public Attraction(String name, int rating, int visitCount) {
        this.name = name;
        this.rating = rating;
        this.visitCount = 0;
    }

    public String getName() { return name; }

    public int getRating() { return rating; }

    public int getVisitCount() { return visitCount; }
}
```

```java
package attractions;

import behaviours.IReviewed;
import behaviours.ITicketed;
import people.Visitor;

public class Dodgems extends Attraction implements IReviewed, ITicketed {

    public Dodgems(String name, int rating, int visitCount) {

        super(name, rating, visitCount);
    }
```

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.

Unit	Ref	Evidence
		Description The first algorithm assesses whether the customer's budget is greater than the price of the vehicle before allowing them the option to add the car to purchase. Once it has been confirmed that the customer is able to purchase the car the till updates the amount of cash in the register with the original amount plus the price of the car. **The second algorithm demonstrates a for loop which checks over all the room names to see if the dining room name matches the required name then when it does return the name of the room.** **I chose these algorithms as they are good representations of common algorithms in programming: if statements and for loops.**

Paste Screenshot here

```
public void sellVehicle(Customer customer, Car car) {
    if (car.getPrice() < customer.getBudget()){
        customer.addPurchase(car);
        this.removeCar(car);
        customer.setBudget(customer.getBudget() - car.getPrice());
        till.setCash(till.getCash() + car.getPrice());
    }
```

```
public DiningRoom findDiningroom(String name){
    DiningRoom foundRoom = null;
    for(DiningRoom diningRoom : this.diningRooms) {
        if (diningRoom.getName() == name) {
            foundRoom = diningRoom;
        }
    }
    return foundRoom;
```

Description here