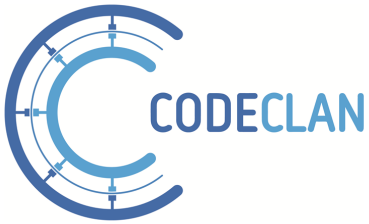


Evidence Gathering Document for SQA Level 8 Professional Developer Award.

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:

```ruby
fruits_array = ['apple', 'banana', 'pear', 'grape']

def print_array(array)
  puts array
end

print_array(fruits_array)
```

```
➜  ruby_evidence git:(master) ✗ ruby array_evidence.rb
apple
banana
pear
grape
```

Description here

This is an example of an array (fruits) being used in a small function. The function prints arrays.

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

```ruby
person = {
    :name => 'Michael',
    :age => 26,
    :favourite_flavour => 'Caramel Choo Choo'
}

def get_favourite_flavour(person)
  person[:favourite_flavour]
end

print get_favourite_flavour(person)
```

```
➜  ruby_evidence git:(master) ✗ ruby array_evidence.rb
Caramel Choo Choo%
```

Description here

This shows an example of hash and a function that gets favourite flavours of ice cream when given a person hash

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:

```ruby
people = [

        {
            :name => 'Michael',
            :age => 26,
            :favourite_flavour => 'Caramel Choo Choo'

        },
        {
            :name => 'Clare',
            :age => 25,
            :favourite_flavour => 'Phish Food'
        },
        {
            :name => 'Amy',
            :age => 25,
            :favourite_flavour => 'Strawberry Cheesecake'
        }
]

def get_person_who_likes_flavor(array, flavour)
  found = nil
  array.each do |x|
    if x[:favourite_flavour] === flavour
       found = x
    end
  end
  return found
end

puts get_person_who_likes_flavor(people, flavour 'Phish Food')
puts get_person_who_likes_flavor(people, flavour 'Strawberry Cheesecake')
puts get_person_who_likes_flavor(people, flavour 'Mint Choc Chip')
```

```
➜  ruby_evidence git:(master) ✗ ruby ruby.rb
{:name=>"Clare", :age=>25, :favourite_flavour=>"Phish Food"}
{:name=>"Amy", :age=>25, :favourite_flavour=>"Strawberry Cheesecake"}
```

Description here

This function searches through an array of hashes and returns the person whose favourite flavour equals the provided parameter

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:

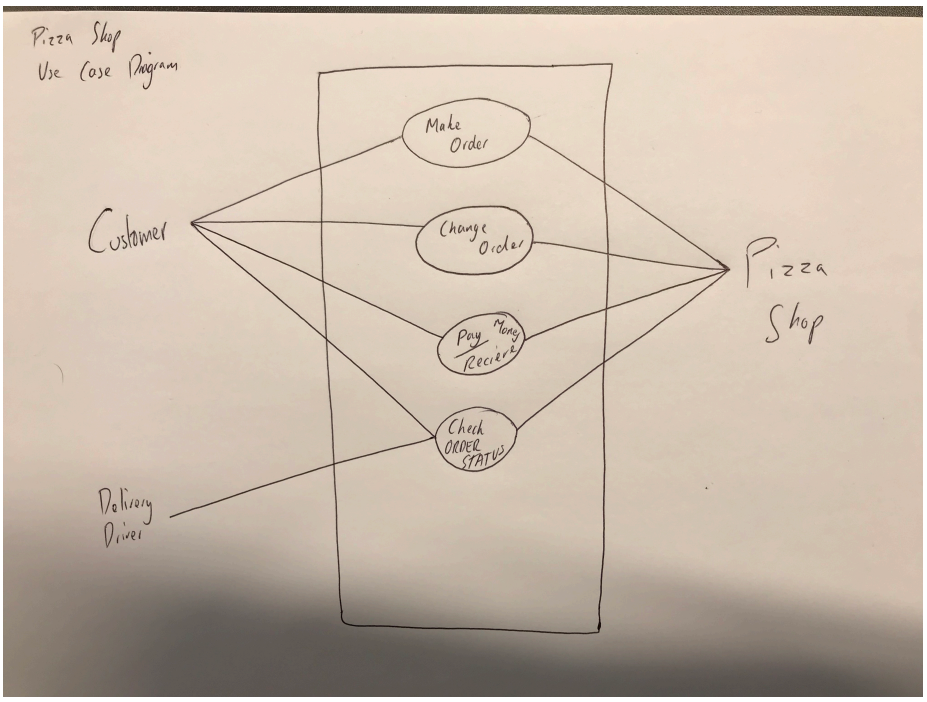
```ruby
names_array = ['joan of arc', 'shuri', 'steve rogers', 'stephen strange', 'iron man', 'thor']

def sort_array_in_order_of_length(array)
  array.sort_by(&:length)
end

puts "original names array #{names_array}"
puts "sorted names array #{sort_array_in_order_of_length(names_array)}"
```

```
➜  ruby_evidence git:(master) ✗ ruby ruby.rb
original names array ["joan of arc", "shuri", "steve rogers", "stephen strange", "iron man", "thor"]
sorted names array ["thor", "shuri", "iron man", "joan of arc", "steve rogers", "stephen strange"]
```

This function sorts an array of items into length order from shortest to longest

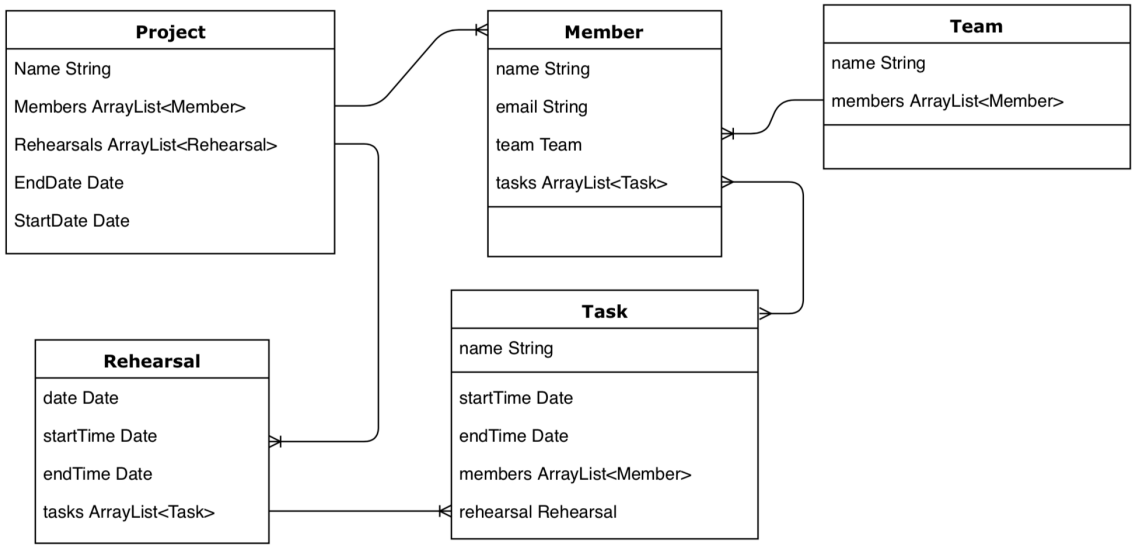
Week 5 and 6

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



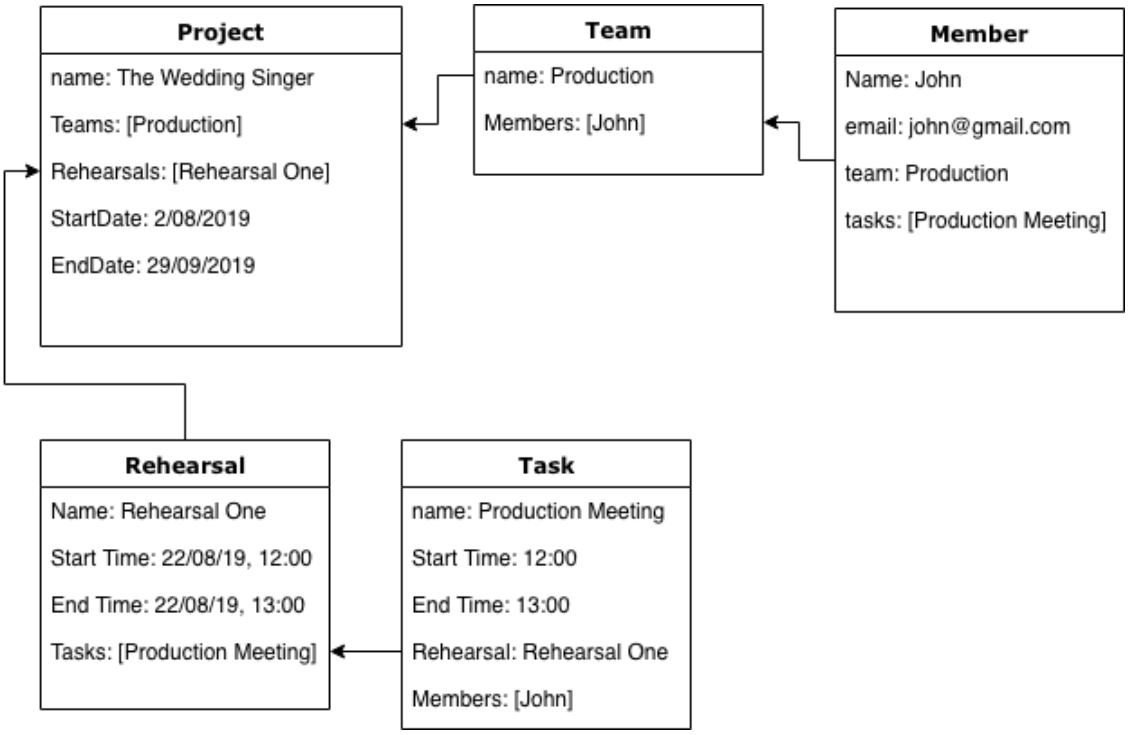
This is a use case diagram created for a Pizza Shop and its potential users

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



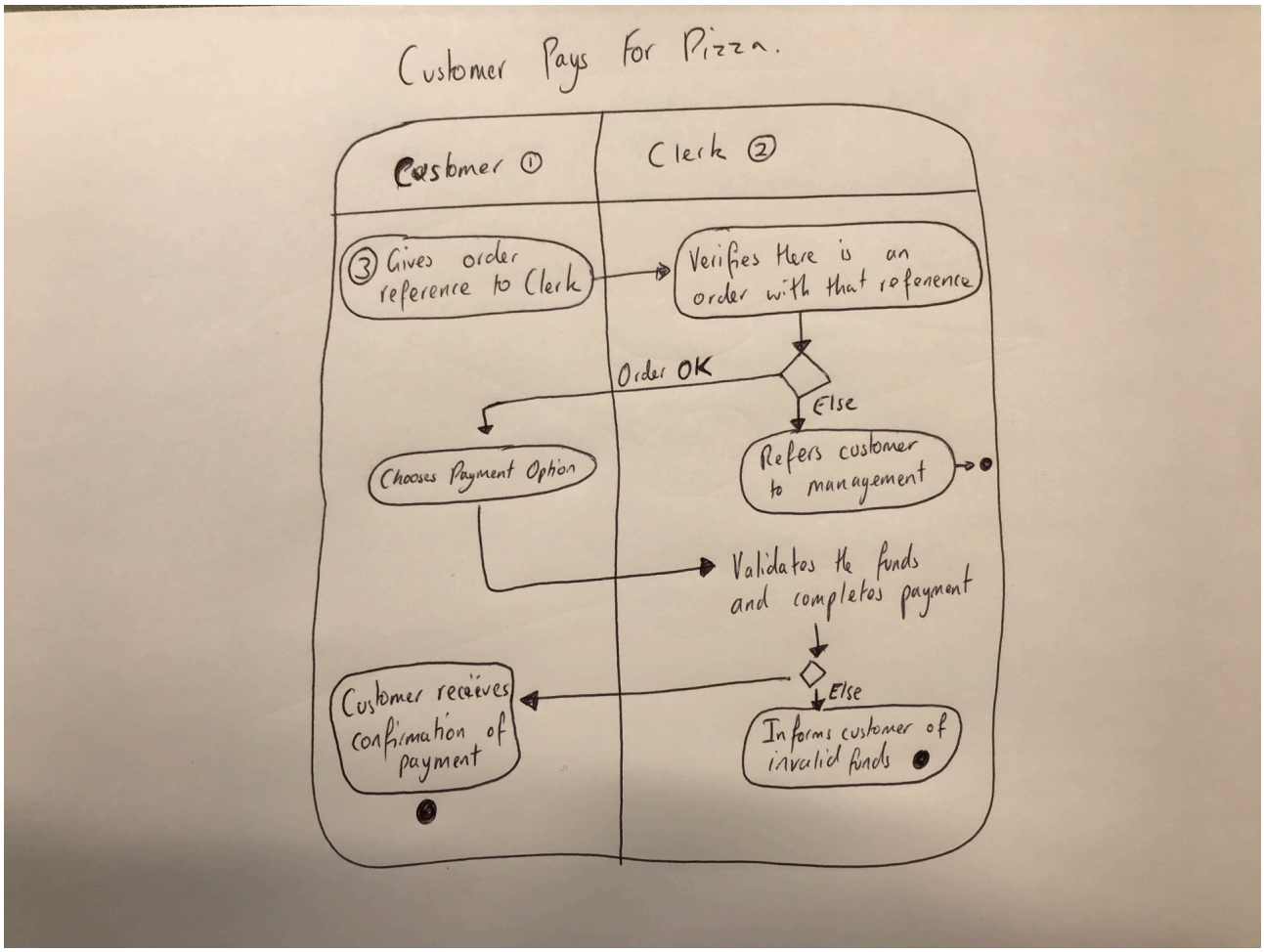
This is a class diagram for a Rehearsal Tracking App

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



An Object Diagram from a Rehearsal Tracking App

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



An activity diagram for a customer paying for a pizza order at a shop

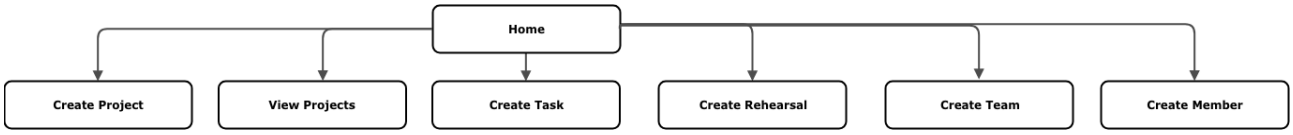
Unit	Ref	Evidence
A&D	A.D.6	Produce an Implementations Constraints plan detailing the following factors: *Hardware and software platforms *Performance requirements *Persistent storage and transactions *Usability *Budgets *Time
		Description:

Constraints

Topic	Possible Effect of Constraint on Product	Solution
Hardware and Software Platforms	Charts may require large downloads	Optimise code to reduce file sizes
Performance Requirements	No data loaded without internet connection	Cache data to local storage
Persistent Storage and Transactions	Data will be out of date if updated remotely by another user	Inform user that
Usability	App requires a lot of charts to be rendered, device sizes may affect usability	Make sure that charts are responsive and provide filters
Budgets	Multiplatform support may be limited due to lack of dev team experience	Increase budget to allow for the hire of more experienced members/ provide training time for staff
Time Limitations	Development takes place over 5 days, all features may not be complete	Increase development time to ensure MVP completion

Implementations and Constraints plan which details the factors relating to a Rehearsal Tracking App

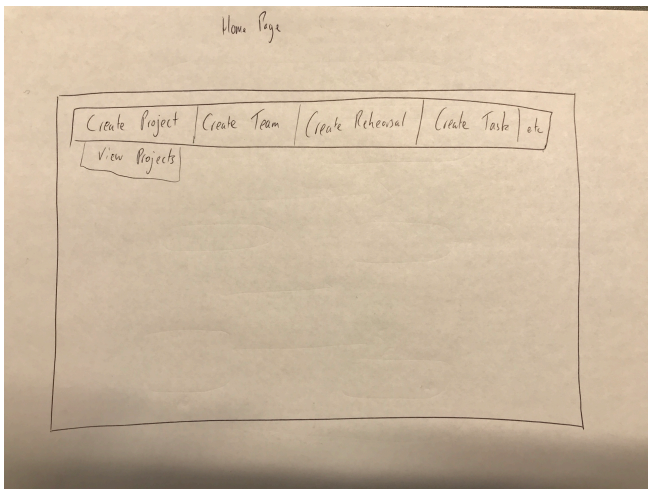
Unit	Ref	Evidence
P	P.5	User Site Map
		Description:

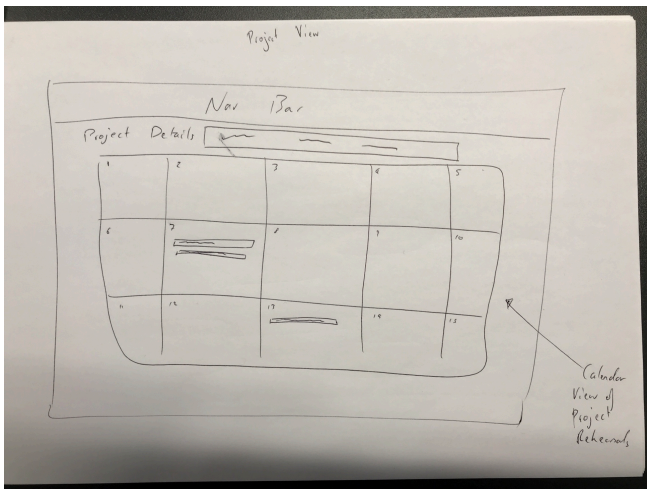


A site map for a Rehearsal Tracking App

Unit	Ref	Evidence
P	P.6	2 Wireframe Diagrams
		Description:

This is an example of two wire frames from a Rehearsal Tracking App



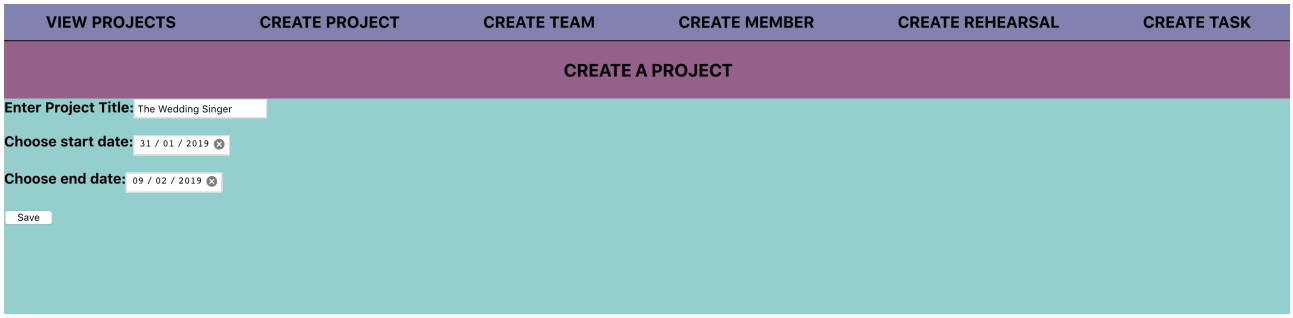


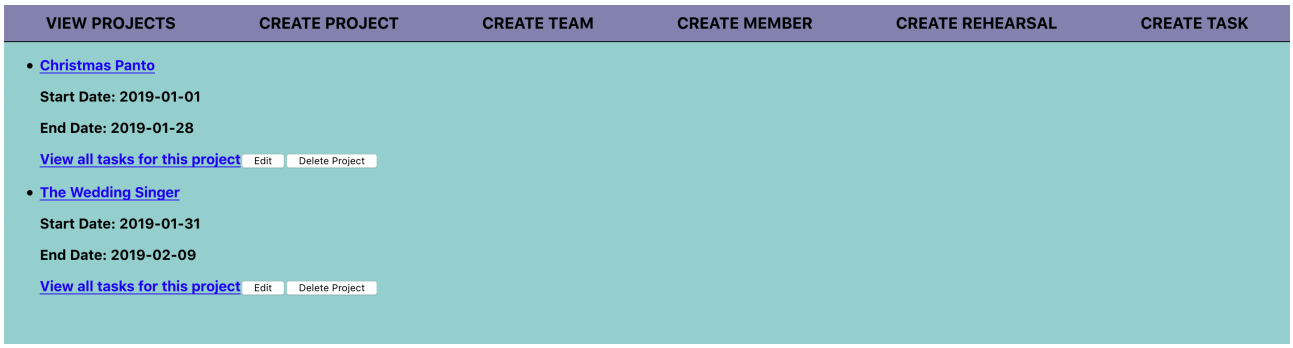
Unit	Ref	Evidence
P	P.10	Example of Pseudocode used for a method
		Description:

```
it('should find a pizza order in the database given that pizza order ID'), function() {
#     should query the database searching through pizza_orders with a given ID
#     should then create a new pizza order object and save it to a variable
#     should then return that variable
}
```

An example of a method which uses an SQL query to interact with a database and return an entry object

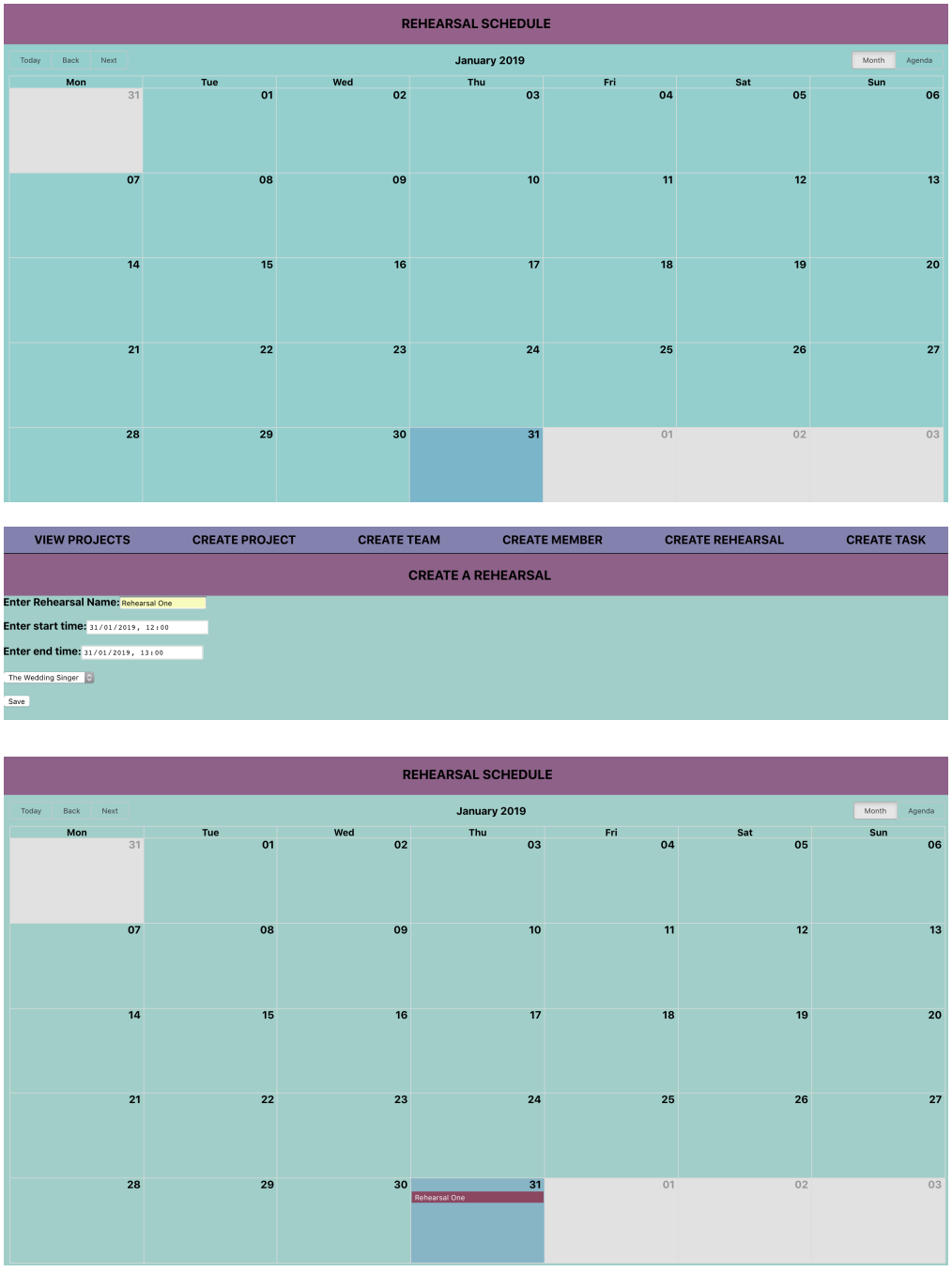
Unit	Ref	Evidence
P	P.13	Show user input being processed according to design requirements. Take a screenshot of: * The user inputting something into your program * The user input being saved or used in some way
		Description:





An example of someone inputting Project Data into the app and then it saving and being available to view in the Projects View

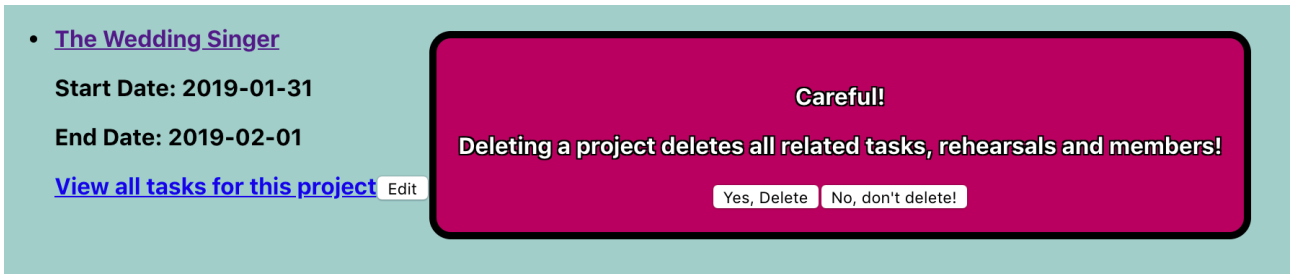
Unit	Ref	Evidence
P	P.14	Show an interaction with data persistence. Take a screenshot of: * Data being inputted into your program * Confirmation of the data being saved
		Description:

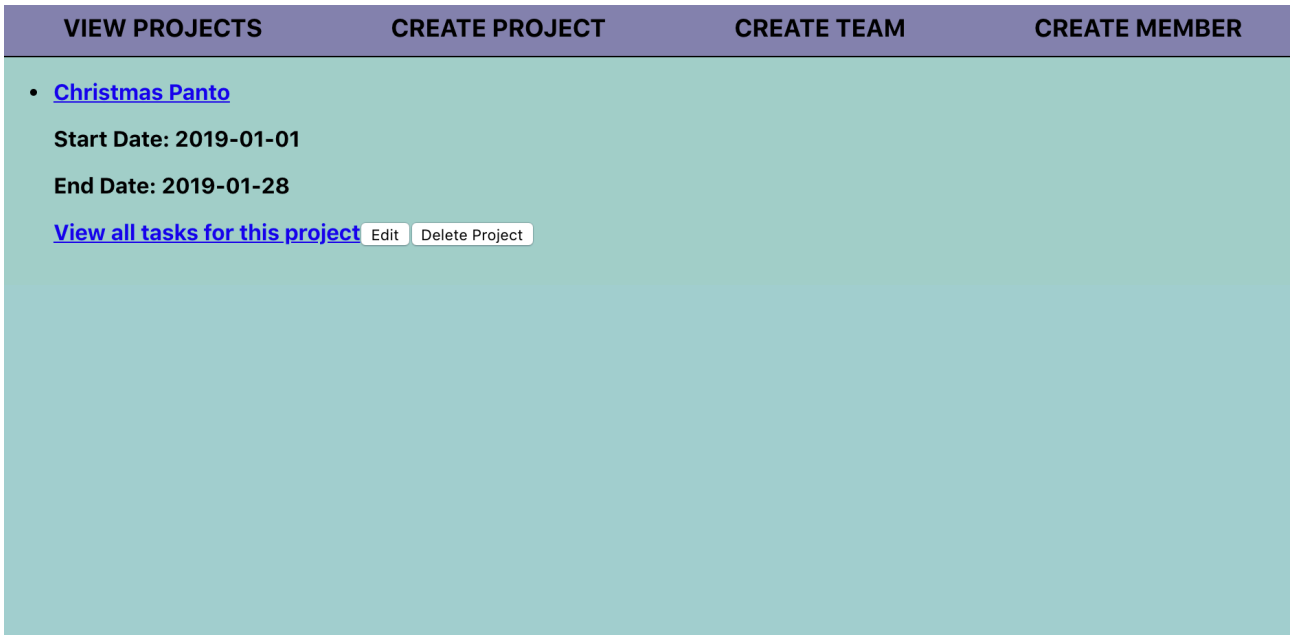


An example of a rehearsal being added to a Project and then appearing in the Project Calendar View

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:

An example of a user deleting a project and then the project no longer appearing in the View Projects view

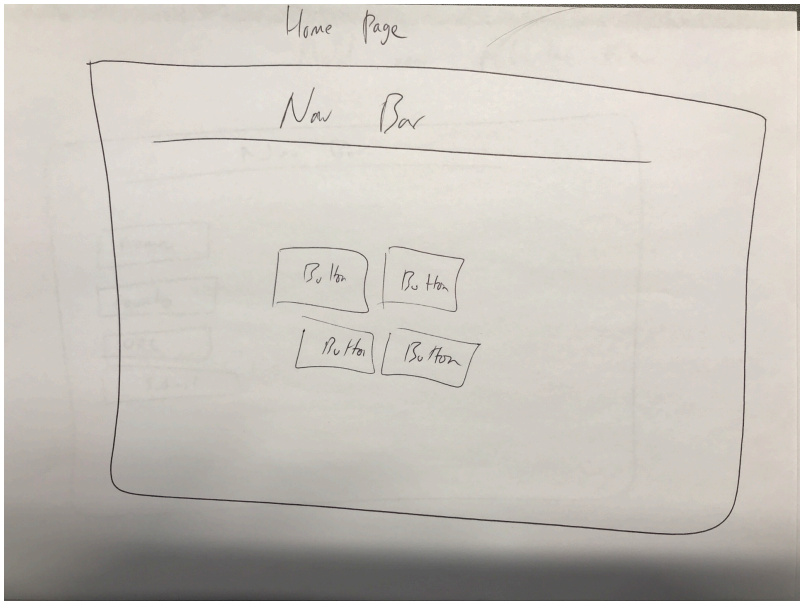


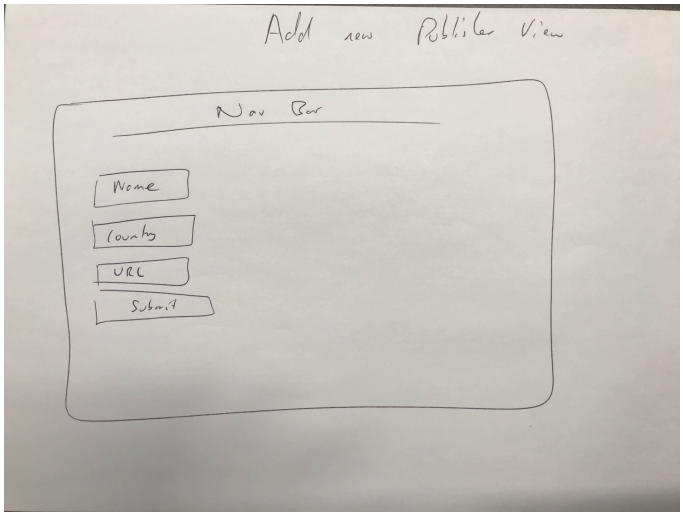


Unit	Ref	Evidence
P	P.11	Take a screenshot of one of your projects where you have worked alone and attach the Github link.
		Description:

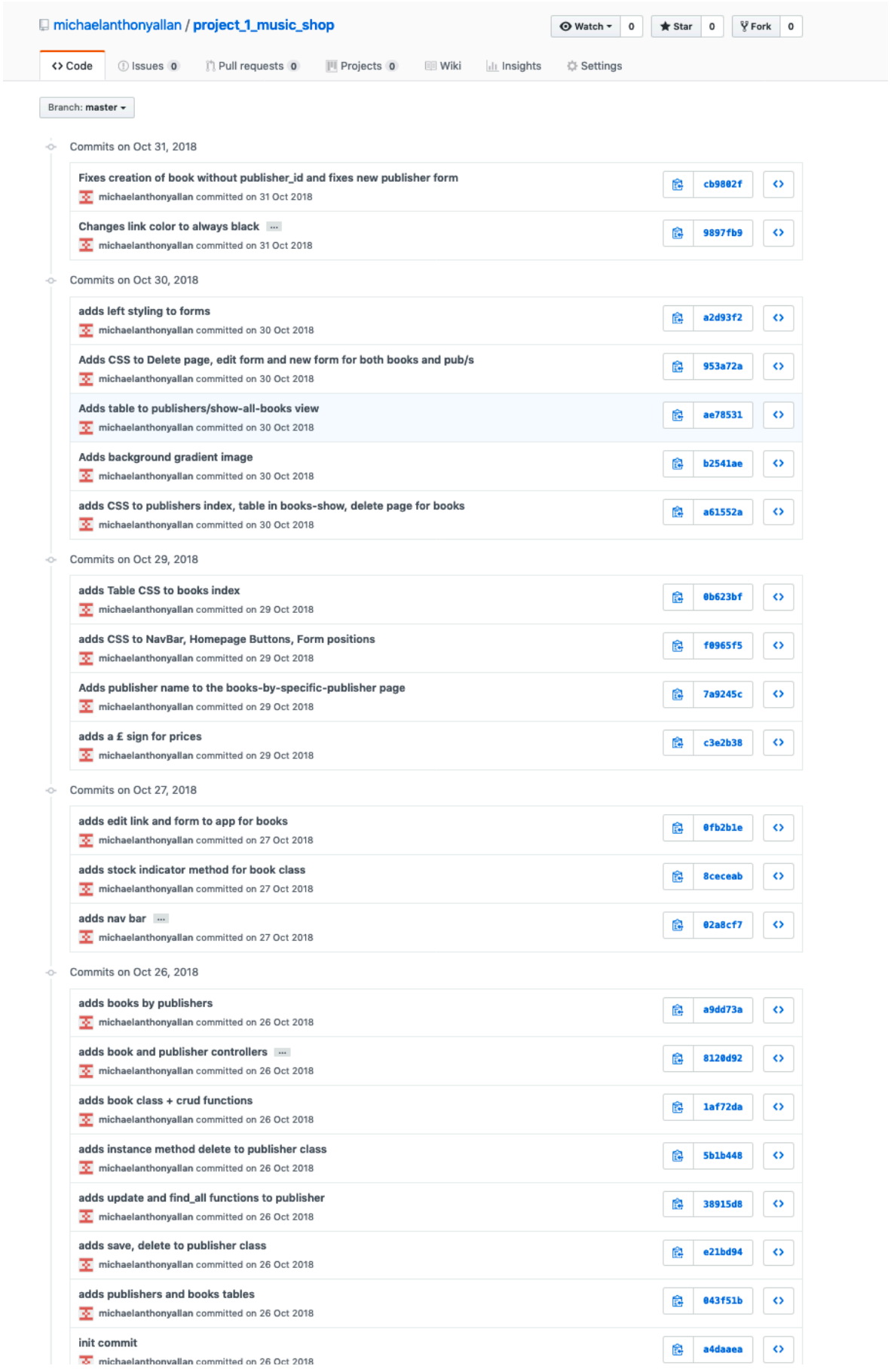
https://github.com/michaelanthonyallan/project_1_music_shop

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





Screenshots are examples of my planning along with a list of commits made to the development of an inventory app which show the additions made during development


michaelanthonyallan / project_1_music_shop
Watch 0
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
Branch: master
Commits on Oct 31, 2018
Fixes creation of book without publisher_id and fixes new publisher form
michaelanthonyallan committed on 31 Oct 2018
cb9802f
Changes link color to always black
michaelanthonyallan committed on 31 Oct 2018
9897fb9
Commits on Oct 30, 2018
adds left styling to forms
michaelanthonyallan committed on 30 Oct 2018
a2d93f2
Adds CSS to Delete page, edit form and new form for both books and pub/s
michaelanthonyallan committed on 30 Oct 2018
953a72a
Adds table to publishers/show-all-books view
michaelanthonyallan committed on 30 Oct 2018
ae78531
Adds background gradient image
michaelanthonyallan committed on 30 Oct 2018
b2541ae
adds CSS to publishers index, table in books-show, delete page for books
michaelanthonyallan committed on 30 Oct 2018
a61552a
Commits on Oct 29, 2018
adds Table CSS to books index
michaelanthonyallan committed on 29 Oct 2018
0b623bf
adds CSS to NavBar, Homepage Buttons, Form positions
michaelanthonyallan committed on 29 Oct 2018
f0965f5
Adds publisher name to the books-by-specific-publisher page
michaelanthonyallan committed on 29 Oct 2018
7a9245c
adds a £ sign for prices
michaelanthonyallan committed on 29 Oct 2018
c3e2b38
Commits on Oct 27, 2018
adds edit link and form to app for books
michaelanthonyallan committed on 27 Oct 2018
0fb2b1e
adds stock indicator method for book class
michaelanthonyallan committed on 27 Oct 2018
8ceceab
adds nav bar
michaelanthonyallan committed on 27 Oct 2018
02a8cf7
Commits on Oct 26, 2018
adds books by publishers
michaelanthonyallan committed on 26 Oct 2018
a9dd73a
adds book and publisher controllers
michaelanthonyallan committed on 26 Oct 2018
8120d92
adds book class + crud functions
michaelanthonyallan committed on 26 Oct 2018
1af72da
adds instance method delete to publisher class
michaelanthonyallan committed on 26 Oct 2018
5b1b448
adds update and find_all functions to publisher
michaelanthonyallan committed on 26 Oct 2018
38915d8
adds save, delete to publisher class
michaelanthonyallan committed on 26 Oct 2018
e21bd94
adds publishers and books tables
michaelanthonyallan committed on 26 Oct 2018
043f51b
init commit
michaelanthonyallan committed on 26 Oct 2018
a4daaea

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:

```
class Pokemon {
  constructor() {
    this.data = null;
    this.pokemonData = null;
    this.speciesData = null;
  }

  bindEvents(){
    PubSub.subscribe( channel: 'SelectView:change',  callback: (event) => {
      console.log('change event called');
      const pokemonIndex = event.detail;
      this.getPokemon(pokemonIndex);
      this.getSpecies(pokemonIndex);
    })
  }

  getData(){
      const url = "https://pokeapi.co/api/v2/pokemon/";
      const request = new Request(url);
      request.get()
          .then((pokemon) => {
              this.data = pokemon;
              PubSub.publish( channel: "Pokemon:all-pokemon-ready", this.data);
              console.log("pokemon receiving data:", pokemon);
          })
  };

  getPokemon(pokemonIndex) {
      const pokemonIndexInteger = parseInt(pokemonIndex,  radix: 10);
      const adjustedPokemonIndex = (pokemonIndexInteger + 1);
      const pokemonURL = `https://pokeapi.co/api/v2/pokemon/${adjustedPokemonIndex}`;
      const pokemonRequest = new Request(pokemonURL);
      pokemonRequest.get()
          .then((pokemon) => {
              this.pokemonData = pokemon;
              PubSub.publish( channel: "Pokemon:selected-pokemon-ready", this.pokemonData);
              console.log("Pokemon broadcasting selected pokemon", pokemon);
          });
  };


  getSpecies(pokemonIndex) {
    const pokemonIndexInteger = parseInt(pokemonIndex,  radix: 10)
    const adjustedPokemonIndex = (pokemonIndexInteger + 1)
    const pokemonURL = `https://pokeapi.co/api/v2/pokemon-species/${adjustedPokemonIndex}`
    const speciesRequest = new Request(pokemonURL)
    speciesRequest.get()
        .then((species) => {
          this.speciesData = species
          PubSub.publish( channel: "Pokemon:selected-species-ready", this.speciesData);
          console.log("Pokemon broadcasting selected species", species);
```

Examples of the code that uses the API url to get data and an example from the devTools that show the api getting that data live

```
pokemon receiving data:                                                pokemon.js:30
▶ {count: 949, next: "https://pokeapi-215911.firebaseapp.com/api/v2/pokemon/?
  offset=20&limit=20", previous: null, results: Array(20)}

11                                                                     select_view.js:24

change event called                                                    pokemon.js:16

change: 11                                                             select_view.js:27

⊗ ▶Uncaught ReferenceError: uppercase is not defined                   pokemon_view.js:69
    at pokemon.abilities.forEach (pokemon_view.js:69)
    at Array.forEach (<anonymous>)
    at PokemonView.getAbilities (pokemon_view.js:67)
    at PokemonView.render (pokemon_view.js:41)
    at
HTMLDocument._helpers_pub_sub_js__WEBPACK_IMPORTED_MODULE_0__.default.subscr
ibe (pokemon_view.js:18)
    at Object.publish (pub_sub.js:7)
    at pokemonRequest.get.then (pokemon.js:42)

Pokemon broadcasting selected pokemon                                  pokemon.js:43
▶ {abilities: Array(2), base_experience: 178, forms: Array(1), game_indices:
  Array(20), height: 11, …}

Species view calling with Species!                                     species_view.js:15
▶ {base_happiness: 70, capture_rate: 45, color: {…}, egg_groups: Array(1), ev
  olution_chain: {…}, …}

Pokemon broadcasting selected species                                  pokemon.js:57
▶ {base_happiness: 70, capture_rate: 45, color: {…}, egg_groups: Array(1), ev
  olution_chain: {…}, …}

⊗ ▶GET file:///Users/michael/codeclan/week_7/pokemon_homework_ap       undefined:1
p/public/undefined net::ERR_FILE_NOT_FOUND

>
```

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:

Paste Screenshot here

Description here

```
@Test
public void canAddPlaneToHanger(){
    glasgowAirport.addPlaneToHanger(plane, hanger_1);
    assertEquals( expected: false, hanger_1.hasPlane());
}
```

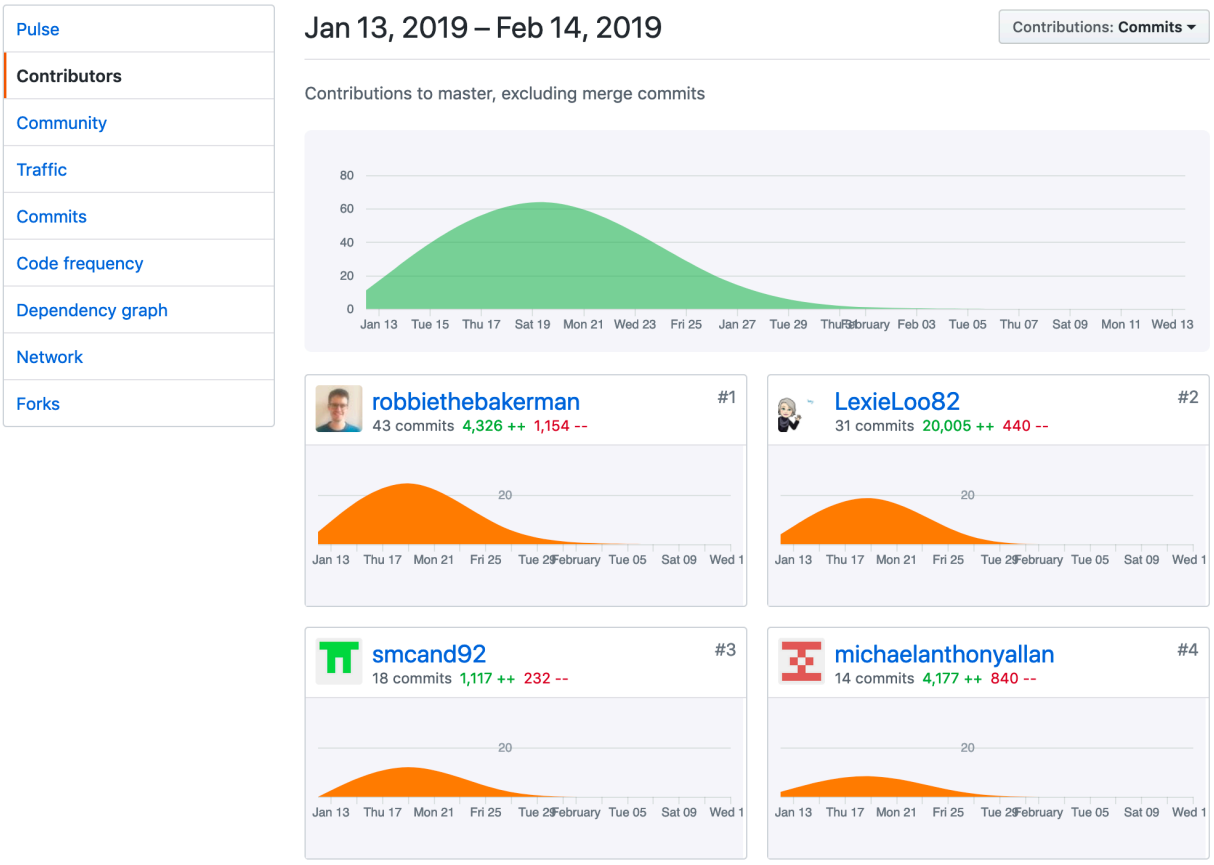
```
java.lang.AssertionError:
Expected :false
Actual   :true
```

```
@Test
public void canAddPlaneToHanger(){
    glasgowAirport.addPlaneToHanger(plane, hanger_1);
    assertEquals( expected: true, hanger_1.hasPlane());
}
```

```
AirportTest 5 ms
    canAddPlaneToHanger 5 ms
/Library/Java/JavaVirtualMachines/jdk-11.0.2.jdk/Contents/Home/bin/java ...

Process finished with exit code 0
```

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:



The contributors page for our group project

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

Java, Spring, React Project

The app will have RESTful routes to serve up data to the frontend

MVP

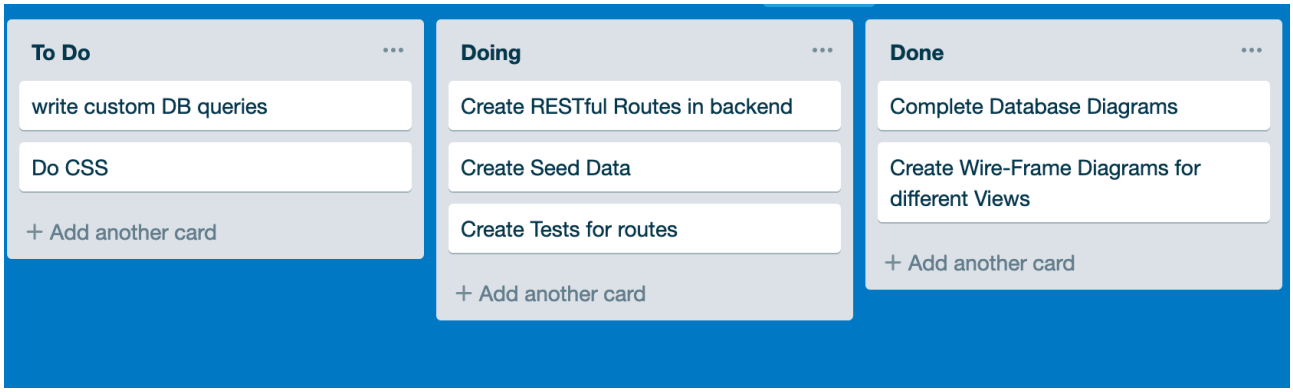
To create a rehearsal scheduling app for a production company.

The user will be able to view a visual representation of the rehearsal schedule
The user will be able to create and edit rehearsals and add attendees to them
The user will be able to expand the rehearsal to view more detail of each rehearsal

Extension

The view can be filtered by rehearsal type or attendee
The app won't allow for the same person to be required for two rehearsals at the same time
The user can record actual attendance post rehearsal

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



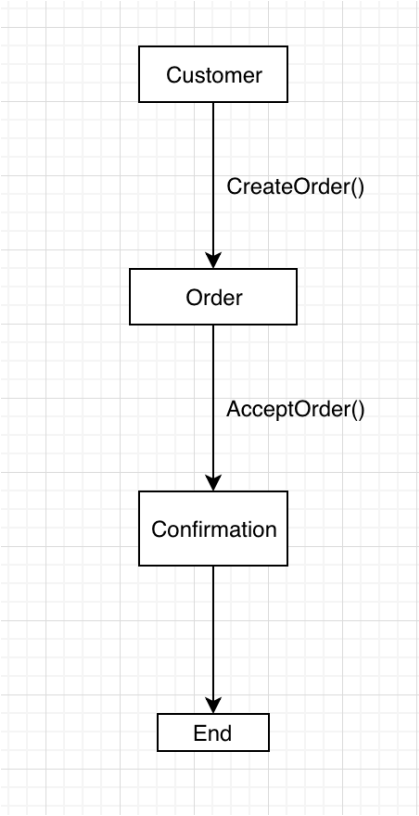
<u>This is a trello board for our group project showing</u>

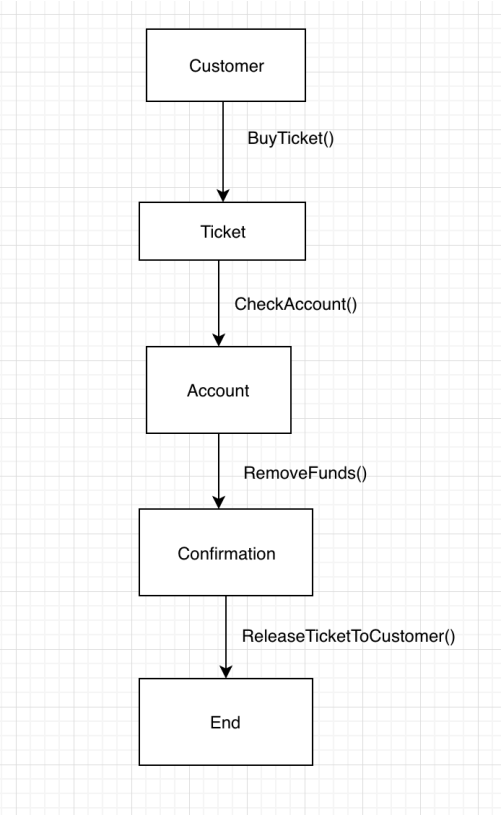
Unit	Ref	Evidence
P	P.4	Write an acceptance criteria and test plan.

Acceptance Criteria and Test Plan	
Criteria	Test
As a user who wishes to add a rehearsal I can create a rehearsal	A user can add a rehearsal and the data is persisted to the database
A user wishes to see when the rehearsals are for a given project	User can select a project and view a calendar of events for that project
A user wishes to delete a rehearsal from a project	A user can click on a delete a rehearsal button and the corresponding rehearsal will be deleted from the database

This screenshot is of an acceptance criteria and test plan which was devised for the rehearsal tracking app, it shows the criteria for acceptance of the app and the associated tests

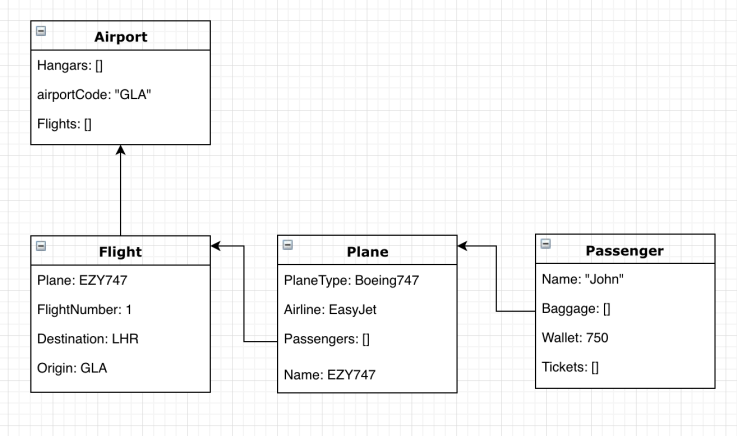
Unit	Ref	Evidence
P	P.7	Produce two system interaction diagrams (sequence and/or collaboration diagrams).
		Description:

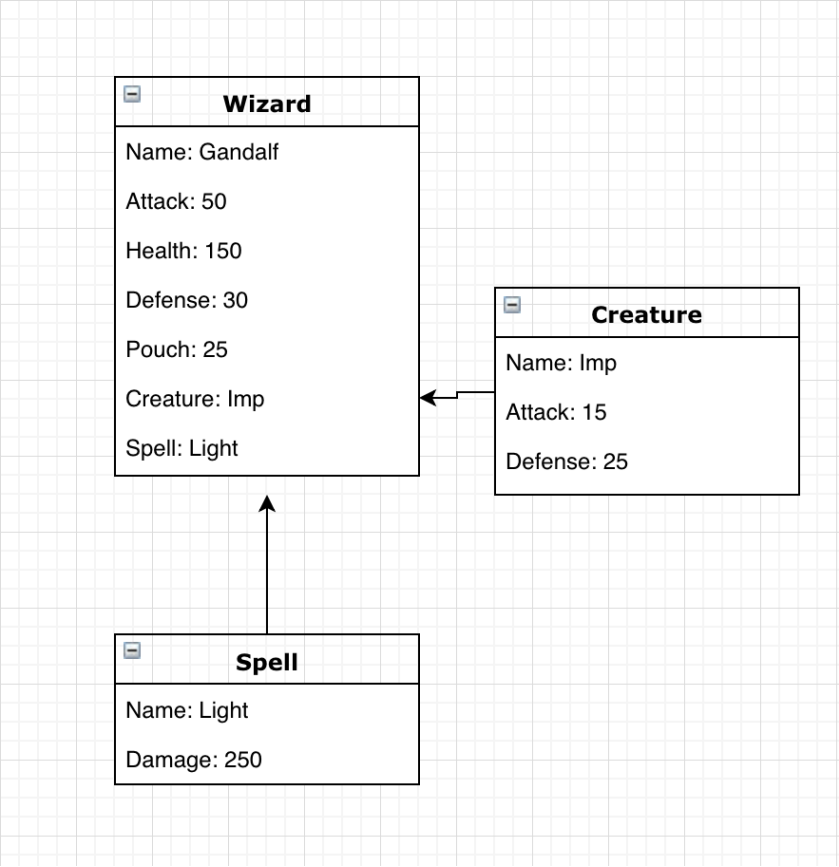




The first screenshot shows the interaction of a customer with an online ordering system and the next screenshot shows a customer interacting with an online ticket ordering system

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





The first screenshot is of objects in an airport system and the second is a selection of objects from a roleplaying game

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

Category	Label	Value
Bug ID	ID number	#1
	Name	Calendar - Can't see all rehearsals on a day
	Reporter	Mike A
Bug overview	Summary	When viewing a project with many rehearsals on one day I cannot see all the rehearsals in the calendar view
Bug details	Steps to reproduce	View a project with more than 3 rehearsals on one day
	Expected result	The calendar should show all rehearsals on a given day
	Actual result	The calendar only shows 3 rehearsals maximum
Bug ID	ID number	#2
	Name	Delete Bug
	Reporter	Steven
Bug overview	Summary	The delete button does not do anything
Bug details	Steps to reproduce	Use the latest build from GitHub and press the delete button on any item
	Expected result	Item should be removed from the database
	Actual result	Nothing is deleted from the database but the redirect happens
Bug ID	ID number	#3
	Name	Cannot add member to task
	Reporter	Alex
Bug overview	Summary	The add member to task button does not take you to add a member
Bug details	Steps to reproduce	Click the add a member button
	Expected result	The add member to ask page should render on click
	Actual result	Nothing happens

A bug tracking spreadsheet for a rehearsal tracking app

Week 12

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

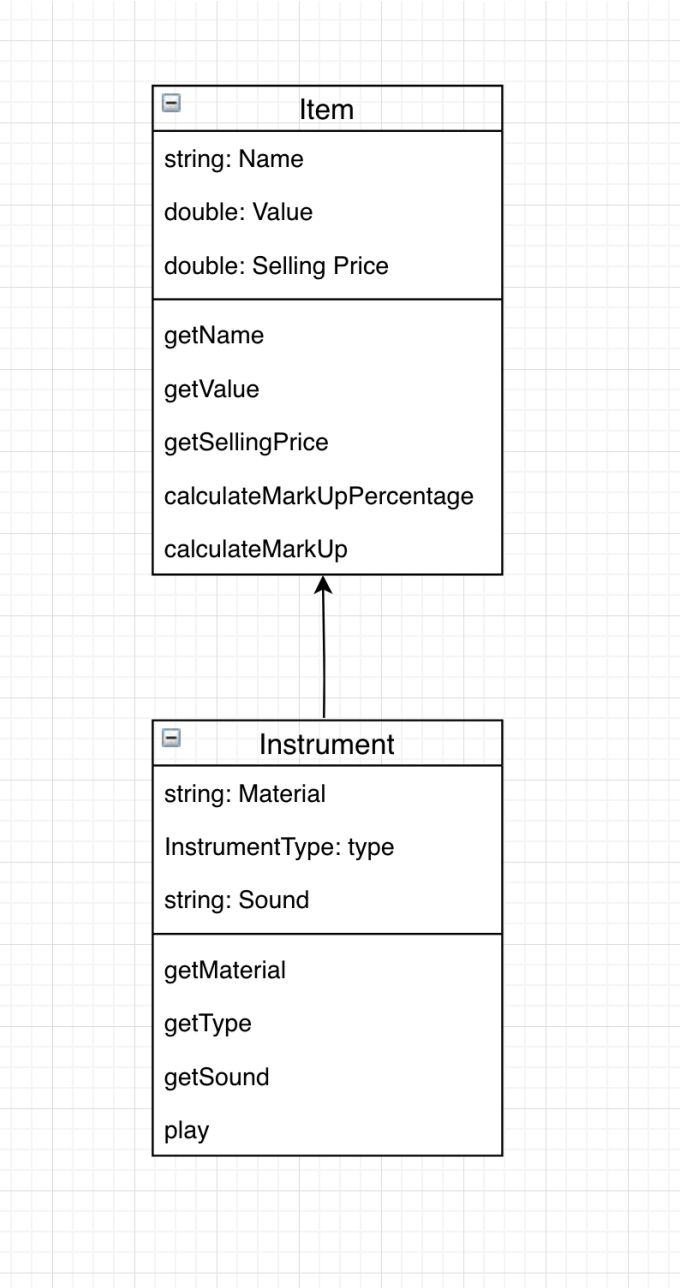
```java
public class Shop implements ISell {

    private ArrayList<Item> stock;
    private String name;

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

    public void addToStock(Item item) {
        stock.add(item);
    }
```

```java
public class Instrument extends Item implements IPlay{

    private String material;
    private InstrumentType type;
    private String sound;


    public Instrument(String name, String material, InstrumentType type, String sound, double value, double selling_price) {
        super(name, value, selling_price);
        this.material = material;
        this.type = type;
        this.sound = sound;
    }
```

```java
@Test
public void addToStock() {
    shop.addToStock(instrument);
    assertEquals( expected: 1, shop.getTotalNumberOfItemsInStock());
}
```

In these examples the first screenshot shows the Shop Class has an arrayList of Item.
The second shows that Instruments extend the Item class.
The third screenshot shows that the Shop can add instances of the Instrument class to its arrayList of Item class instances therefore showing polymorphism.

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



An inheritance diagram showing Instrument inheriting all fields and functions from Item

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

```java
public class Plane {

    private PlaneType model;
    private Airline airline;
    private ArrayList<Passenger> passengers;

    public Plane(PlaneType model, Airline airline){
        this.airline = airline;
        this.model = model;
        this.passengers = new ArrayList<>();
    }

    public PlaneType getModel() {
        return model;
    }

    public Airline getAirline() { return airline; }

    public ArrayList<Passenger> getPassengers() { return passengers; }
```

The instance variables are inaccessible outside of the class because they are set to ‘private’ meaning that they can only be accessed by using the Getter methods, which are shown in the screenshot. Encapsulating the fields prevents users from setting them to invalid or inconsistent values.

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:

```java
public class Item implements ISell{

    String name;
    double value;
    double selling_price;

    public Item(String name, double value, double selling_price) {
        this.name = name;
        this.value = value;
        this.selling_price = selling_price;
    }
```

```java
public class Instrument extends Item implements IPlay{

    private String material;
    private InstrumentType type;
    private String sound;


    public Instrument(String name, String material, InstrumentType type, String sound, double value, double selling_price) {
        super(name, value, selling_price);
        this.material = material;
        this.type = type;
        this.sound = sound;
    }
```

```java
instrument = new Instrument( name: "Casio PSR400", material: "Plastic", InstrumentType.KEYBOARD, sound: "Plink Plonk", value: 150.00, selling_price: 295.50);
```

```java
@Test
public void canGetMarkUpPercentage(){
    assertEquals( expected: "60.0%", instrument.calculateMarkupPercentage());
}
```

The first screenshot shows the Item class from which Instrument inherits

The second screenshot shows the Instrument class which inherits from the Item class

The third screenshot shows an instance of Instrument which contains all the fields from both Item and Instrument

The fourth screenshot shows a test showing that an instance of Instrument class can use the Item method ‘calculateMarkupPercentage’

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:

```
public ArrayList<Hanger> getEmptyHangers(){
    ArrayList<Hanger> emptyHangers = new ArrayList<>();
    for (Hanger hanger : hangers){
        if (!hanger.hasPlane()) {
            emptyHangers.add(hanger);
        }
    }
    return emptyHangers;
}
```

This algorithm was required to create a new list of hangers from an existing array list of hangers based on the criteria that each hanger examined holds currently no planes.

```
public void sellTicketForFlight(Passenger passenger, Flight flight){
    if (flight.getNumberOfBookings() < flight.getPlane().getMaxCapacity()){
        Ticket ticket = createTicketForFlight(flight);
        passenger.removeMoneyFromWallet(ticket.getCost());
        passenger.addTicket(ticket);
        flight.addTicket(ticket);
    }
}
```

This algorithm was required to sell a ticket to a passenger, it allows a complete selling of a ticket to a passenger if there is room on the flight and if so the ticket is created and passed to the flight and passenger.