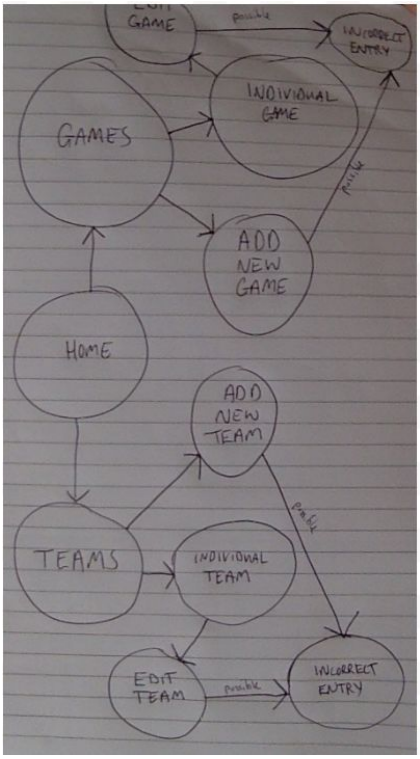
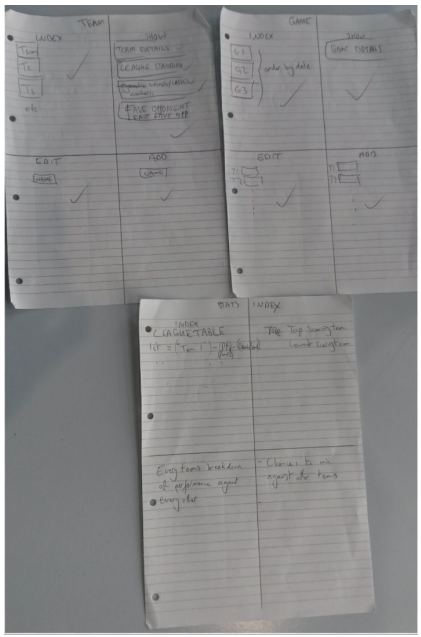
Evidence for Project Unit

Joe Stafford
E21

P.5 User Sitemap



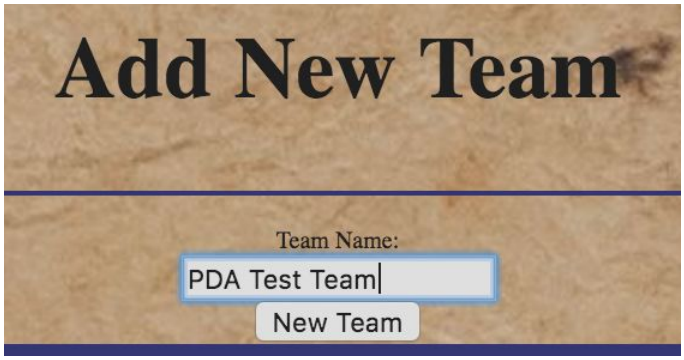
P.6. Wireframes



P.10 Pseudocode

```
it('should return an array of game objects from data garnered from database table 'games'',
function()){
  # create an empty array
  # for each set of data create new 'game' object
  # add new 'game' object to empty array
  # return array of 'game' objects
}
```

P.13 User input being processed according to design requirements



P.14 Show an interaction with data persistence

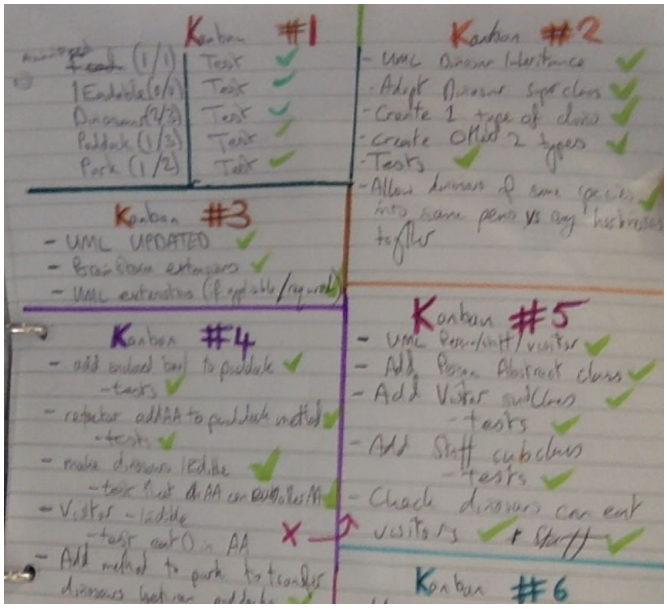
P.15

P.11 - Screenshot of project (file structure)

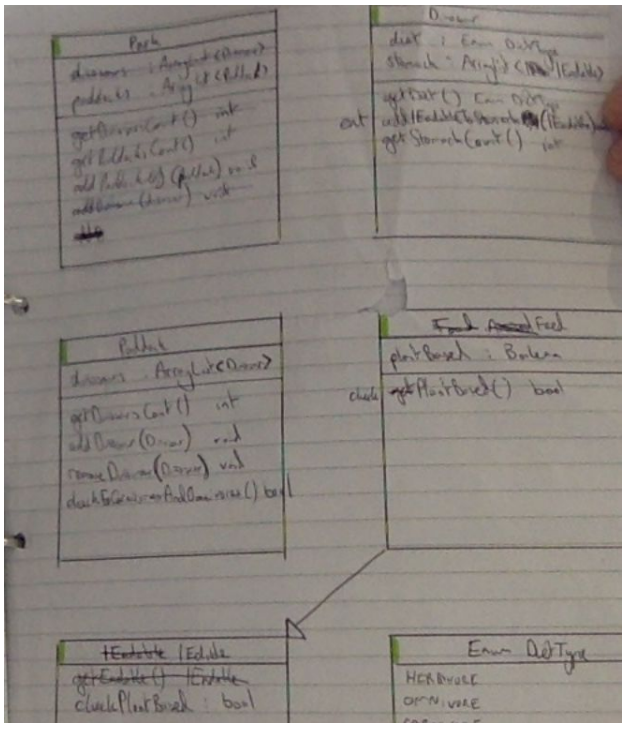
- Link to github repo - https://github.com/JoeStafford1986/java_park_management_system

P.12 - Take screenshots or photos of your planning and the different stages of development to show changes

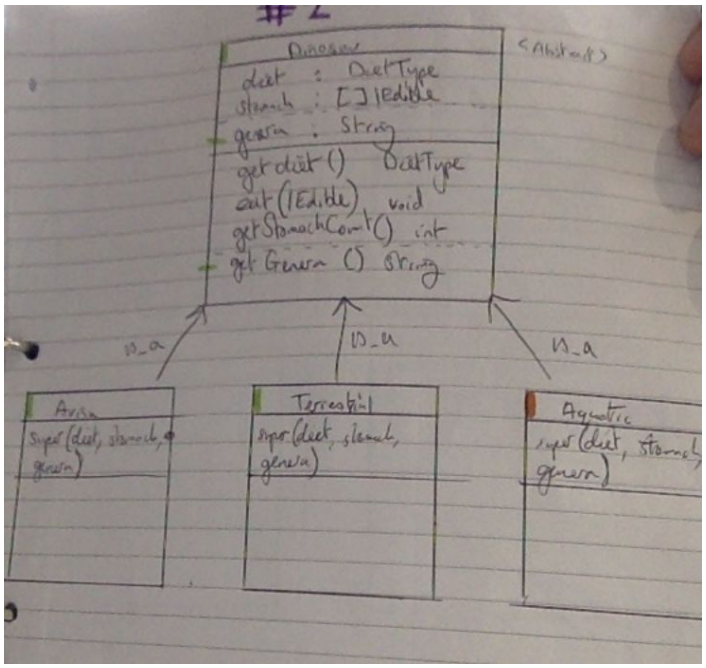
- Kanban



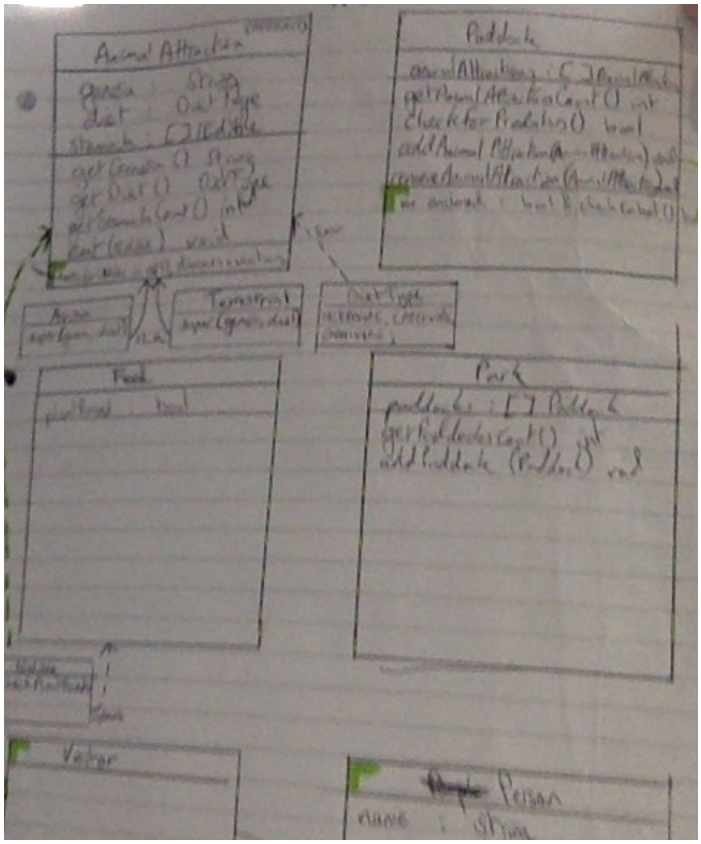
- UML diagrams evolve over time
    - 1



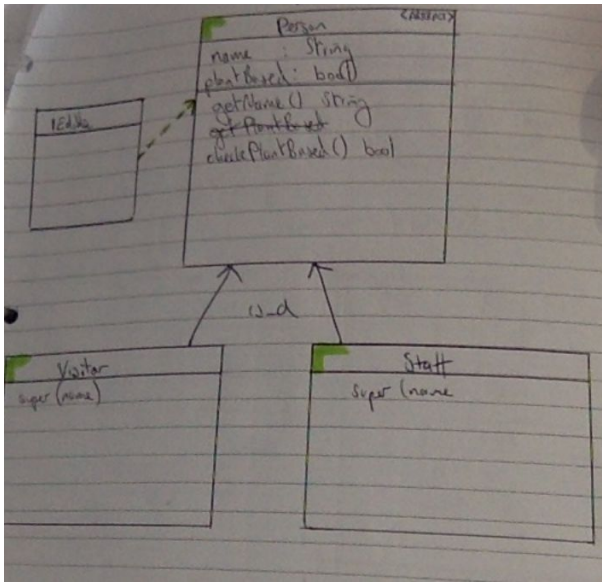
- 2


#2
Dinosaur
<Abstract>
diet : DietType
stomach : [] IEdible
genera : String
getDiet() DietType
eat(IEdible) void
getStomachCount() int
getGenera() String
IS_A
IS_A
IS_A
Arctic
super(diet, stomach, genera)
Terrestrial
super(diet, stomach, genera)
Aquatic
super(diet, stomach, genera)

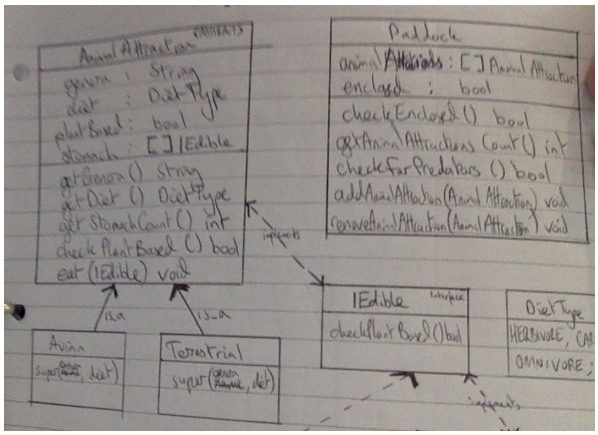
- 3


Animal Attraction
Paddock
Park
Person

- 4


Person
name : String
checkPlantBased() bool
IEdible
is-a
Visitor
super(name)
Staff
super(name

- 5


Animal Attraction
genera : String
diet : DietType
plantBased : bool
stomach : [] IEdible
getGenera() String
getDiet() DietType
getStomachCount() int
checkPlantBased() bool
eat(IEdible) void
Paddock
animalAttractions : [] Animal Attraction
enclosed : bool
checkEnclosed() bool
getAnimalAttractionsCount() int
checkForPredators() bool
addAnimalAttraction(Animal Attraction) void
removeAnimalAttraction(Animal Attraction) void
inspects
IEdible
checkPlantBased() bool
DietType
HERBIVORE, CAR
OMNIVORE
is-a
is-a
Avian
Terrestrial
inspects

P.18 - Demonstrate testing in your program. Take screenshots of:

Example of test code
Example of test failing

```java
import People.Staff;
import org.junit.Before;
import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class StaffTest {
    private Staff staff;

    @Before
    public void before() {
        staff = new Staff( name: "Denis Nedry",  caloricContent: 3500);
    }

    @Test
    public void canGetCaloricContent() {
        assertEquals( expected: 3500, staff.getCaloricContent());
    }

    @Test
    public void canSetCaloricContent() {
        staff.setCaloricContent(3400);
        assertEquals( expected: 3400, staff.getCaloricContent());
    }

    @Test
    public void canGetName() {
        assertEquals( expected: "Dennis Nedry", staff.getName());
    }

    @Test
    public void canCheckPlantBased() {
```

StaffTest › before()

4 tests done: 1 failed - 50ms

```
xpected :Dennis Nedry
ctual   :Denis Nedry
<Click to see difference>

<2 internal calls>
   at StaffTest.canGetName(StaffTest.java:28) <23 internal calls>


rocess finished with exit code 255
```

Example of test code fixed
Example of tests passing

```java
import People.Staff;
import org.junit.Before;
import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class StaffTest {
    private Staff staff;

    @Before
    public void before() {
        staff = new Staff( name: "Dennis Nedry",  caloricContent: 3500);
    }

    @Test
    public void canGetCaloricContent() {
        assertEquals( expected: 3500, staff.getCaloricContent());
    }

    @Test
    public void canSetCaloricContent() {
        staff.setCaloricContent(3400);
        assertEquals( expected: 3400, staff.getCaloricContent());
    }

    @Test
    public void canGetName() {
        assertEquals( expected: "Dennis Nedry", staff.getName());
    }

    @Test
    public void canCheckPlantBased() {
```

StaffTest › before()

All 4 tests passed - 3ms

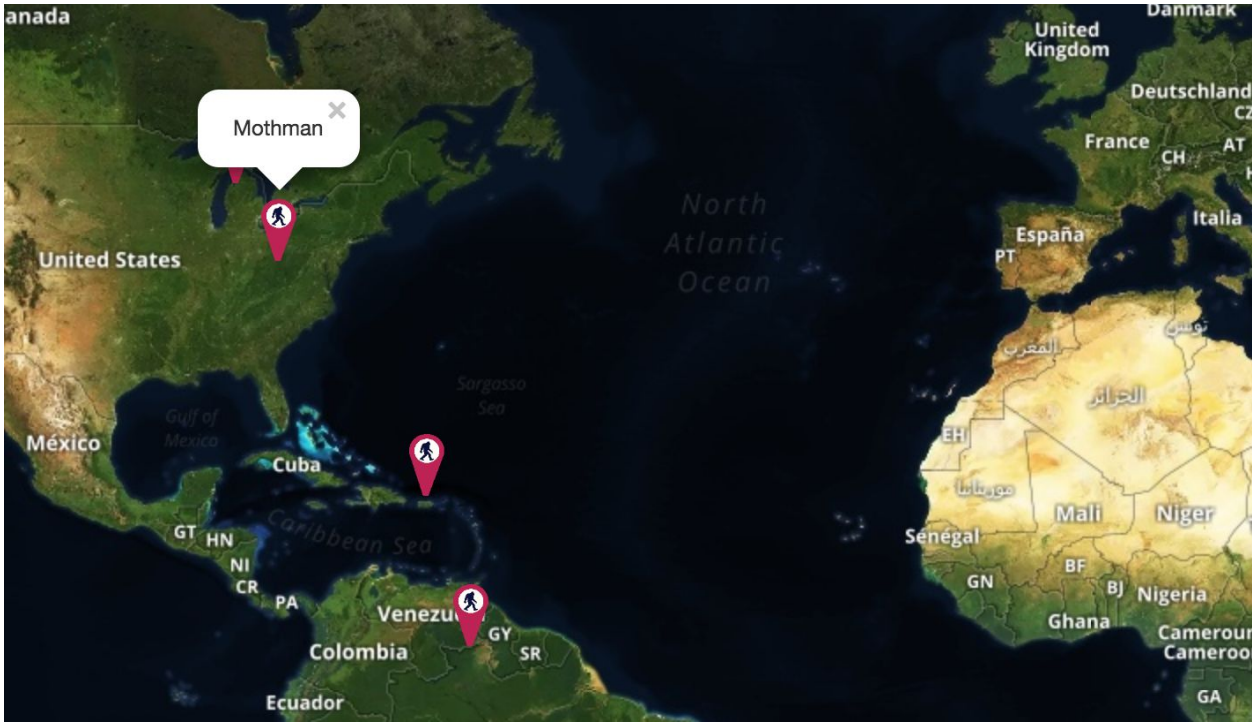
```
.ibrary/Java/JavaVirtualMachines/jdk1.8.0_162.jdk/Contents/Home/bin/java ...

rocess finished with exit code 0
```

P.16 - Show an API being used within your program

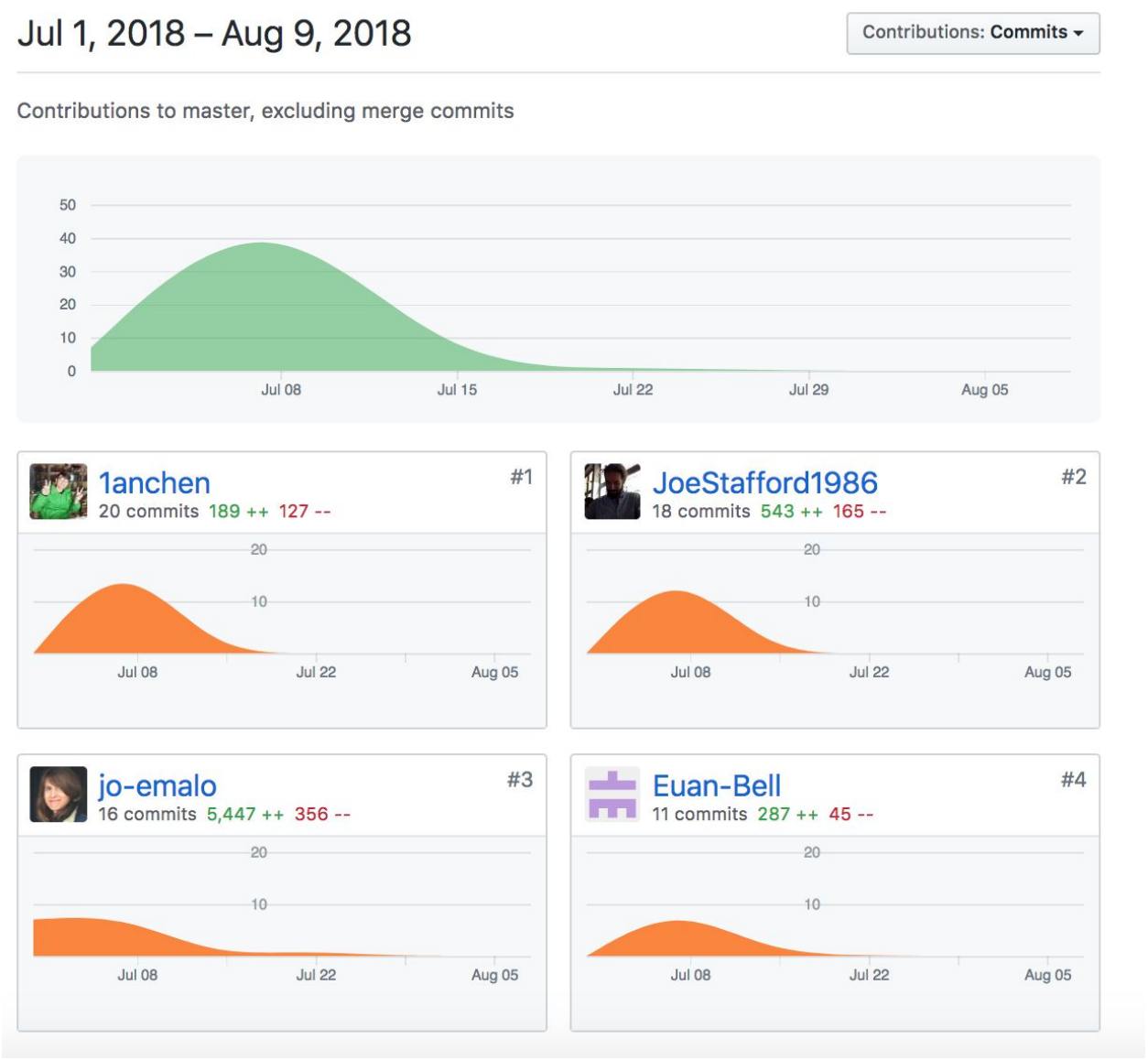
Code that implements the api

```
const Leaflet = require('leaflet');
const PubSub = require('../helpers/pub_sub.js');
const LeafletSidebar = require('leaflet-sidebar');

const MapView = function() {
  this.cryptids = null
  this.markerLayer = Leaflet.layerGroup([]);
  this.markerArray = [];
  this.myMap = Leaflet.map('map',{
    maxBounds:[ [-80, -160], [120, 160] ],
    zoomControl:false
  }).setView([22, 200], 3);
  // 22 ++ set the map down, 170 ++ set map to the left
}

MapView.prototype.renderMap = function() {

  Leaflet.tileLayer('https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}.png?access_token={accessToken}', {
    attribution: 'Map data &copy; <a href="https://www.openstreetmap.org/">OpenStreetMap</a> contributors, <a
    href="https://creativecommons.org/licenses/by-sa/2.0/">CC-BY-SA</a>, Imagery © <a href="https://www.mapbox.com/">Mapbox</a>',

    // 1 is max zoom out, 10 is max zoom in. We have locked zoom level to between 6 + 3.

    maxZoom: 9,
    minZoom: 2,
    id: 'mapbox.streets-satellite',
    accessToken: 'pk.eyJ1Ijoiam9tYWxvIiwiYSI6ImNqajlxenFjdjMzZGYza3BndDF0cHJwNG8ifQ.GxdRYwwkA1aQ4I4R1sOt3Q'
  }).addTo(this.myMap);
```

The leaflet API in use in the webapp



P.1 - Take a screenshot of the contributor's page on Github from your group project to show the team you worked with.



P.2 - Take a screenshot of your project brief

js_group_project

Cryptozoology educational mapping app using vanilla JavaScript and NoSQL.

Educational Cryptozoology App

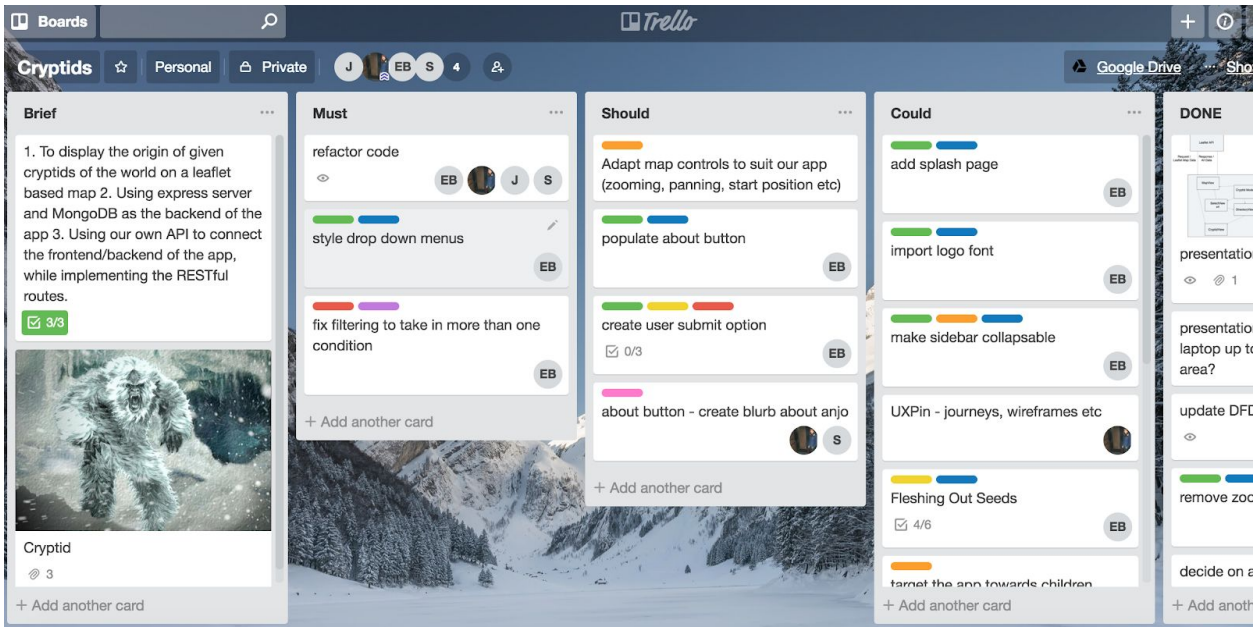
MVP

- To display the origin of given cryptids of the world on a leaflet based map.
- Using express Server and MongoDB as the backend of the app.
- Using our own API to connect the frontend/backend of the app, while implementing the REST Routes.

Extensions

- Multi-layered view of the map, displaying more detailed information, the deeper into the views.
- onClick pop-up gives more detailed view.
- Get image to display.
- Directory of cryptids which onClick will relocate the map.

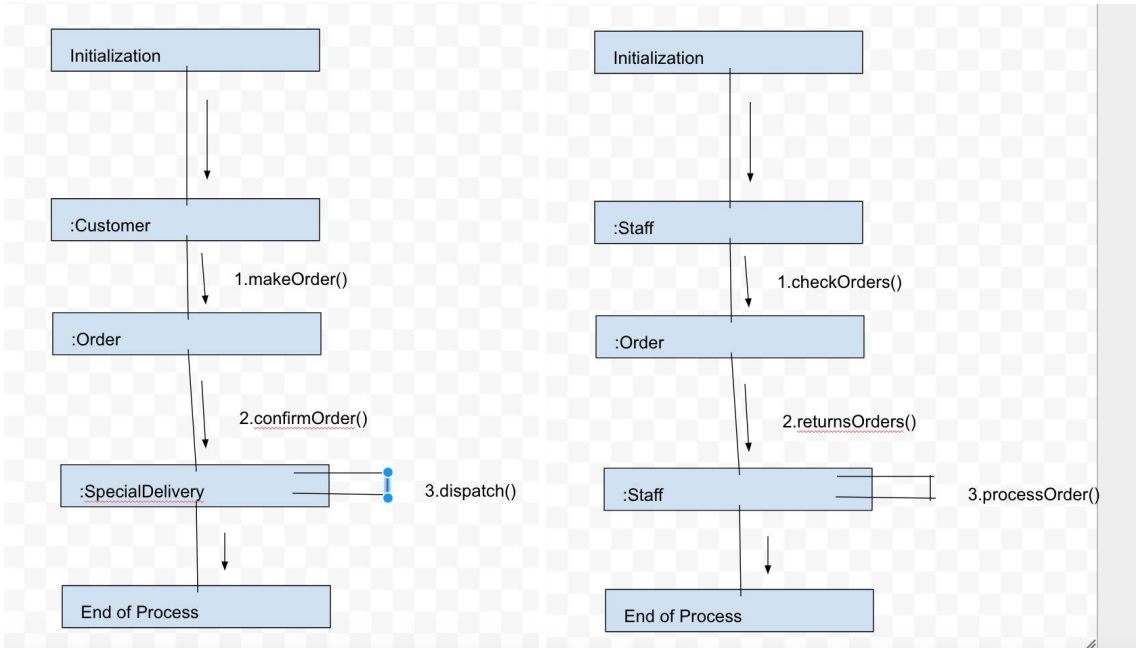
P.3 - Take a screenshot of group planning



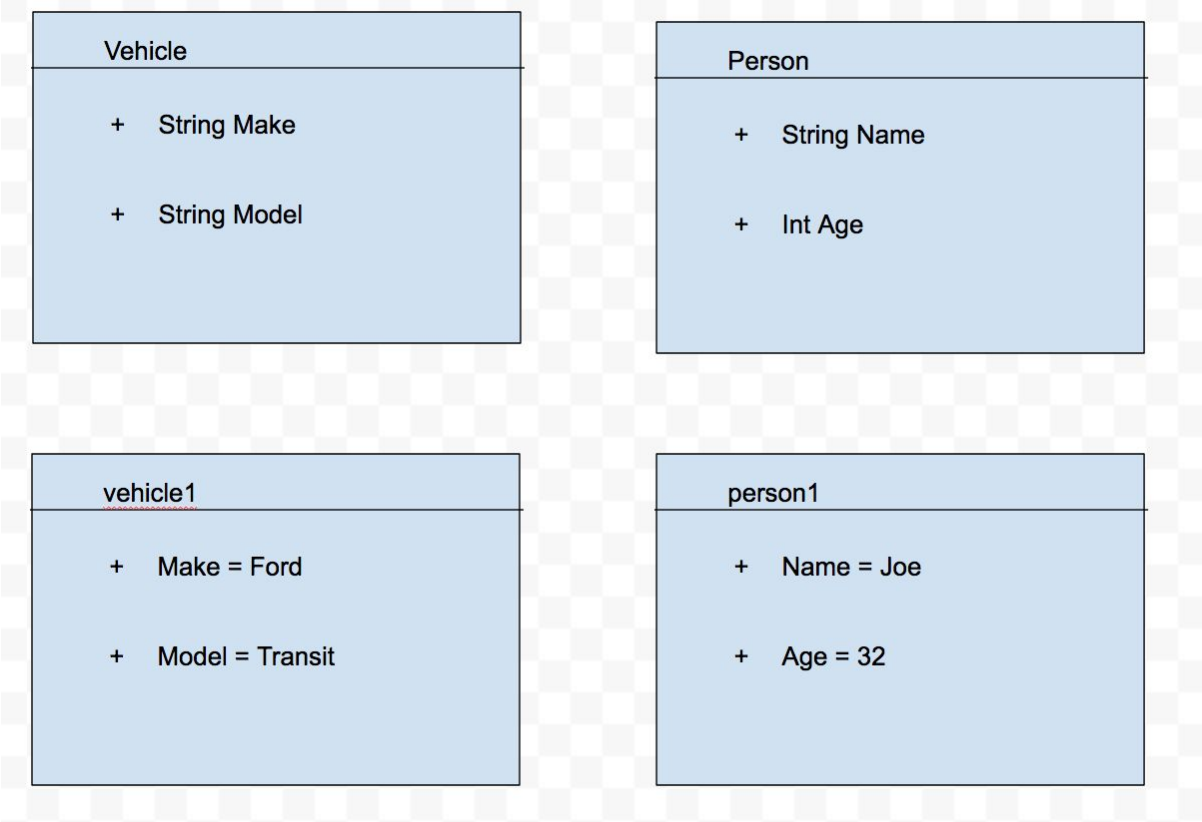
P.4 - Write an acceptance criteria and test plan

Acceptance Criteria	Expected Result/Output	Pass/Fail
A user can access a list of cryptids	List of cryptids is displayed on left toolbar	Pass
Map zooms in when cryptid mapmarker is clicked	Map zooms centred on cryptid	Pass

P.7 - Produce 2 system interaction diagrams



P.8 - Produce two object diagrams



P.9 - Select 2 algorithms

```
61        public int getCaloricContentInStomach() {
62            int allCalories = 0;
63            for (IEdible food : stomach) {
64                allCalories += food.getCaloricContent();
65            }
66            return allCalories;
67        }
```

Here I loop through all of the objects of type IEdible in the stomach array. I add the IEdibles caloric content to a counter and finally return the counter when all of the objects in the array have been looped over.

```
4   class DataManager
5
6     def self.sort_teams()
7       teams = Team.all()
8       sorted = teams.sort_by{|team| [team.get_wins_count(), team.get_goal_difference()]}
9       return sorted.reverse()
10    end
11
12  end
```

In this algorithm I loop over all teams stored in the treams array, and reposition them in the array by first sorted according to number of wins, and secondly by their goal difference (if number of wins is equal to another teams).

P.17 - Produce a bug tracking report

Bug	Pass/Fail	Bug	Pass/Fail
User should be able to add a team	Pass	User cannot input game if a score is missing	Pass
User should be able to delete a team	Pass	User cannot delete games	Fail
User should be able to manipulate the score of an existing game	Fail	User can upload custom image to team	Fail