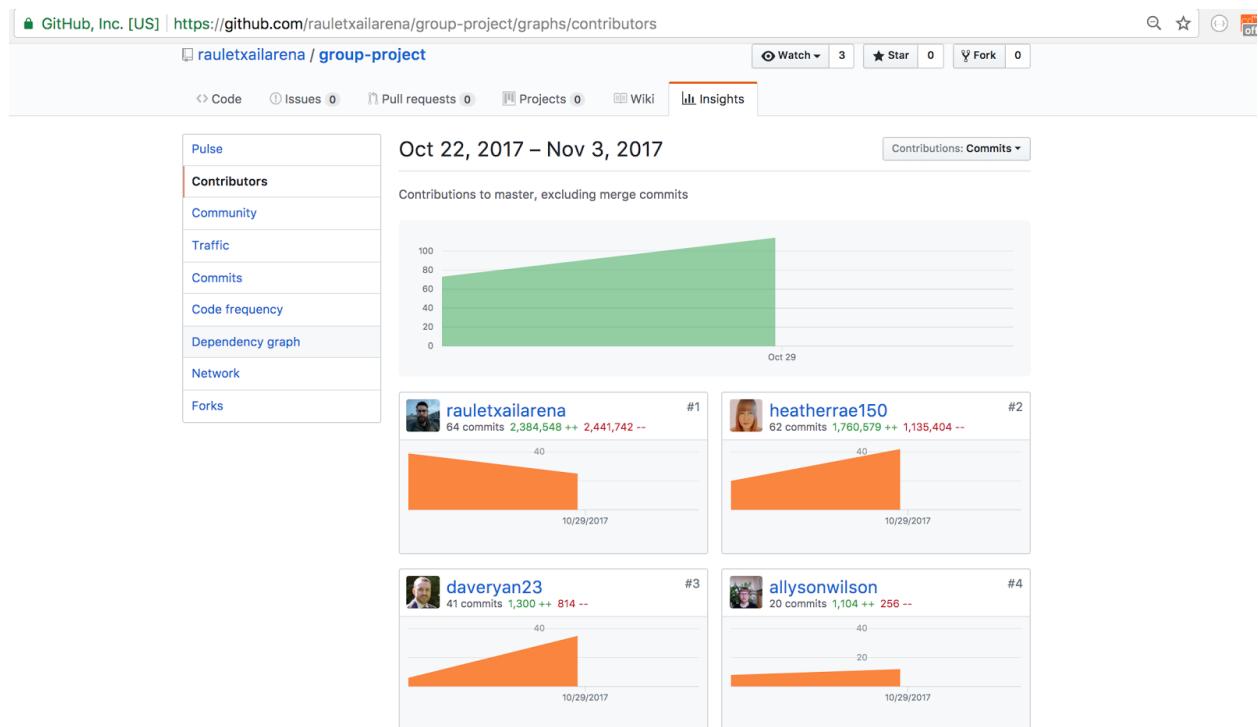


Allyson Wilson  
Cohort 15  
10/11/2017

P1



P2

## Educational App

The BBC are looking to improve their online offering of educational content by developing some interactive apps that display information in a fun and interesting way.

Your task is to make an MVP to put forward to them - this may only be for a small set of information, and may only showcase some of the features to be included in the final app. You might use an API to bring in content or a database to store facts. The topic of the app is your choice, but here are some suggestions you could look into:

- Interactive timeline, e.g. of the history of computer programming
- Interactive map of a historical event - e.g. World War 1, the travels of Christopher Columbus

## MVP

- Display some information about a particular topic in an interesting way
- Have some user interactivity using event listeners, e.g. to move through different sections of content

Some samples of existing apps for inspiration:

## # Our MVP

Should be able to take in user input and display it in a map

Should display information about the area

- pubs
- weather
- events

Interactivity – user can click on map or input postcode or city and retrieve accurate information according to input.

## Extensions

restaurants

crime???

historical data???

bus routes

taxis

## P3

The Trello board has the following columns and items:

- Must Do**: Documentation, Query Helper, Views, HTML, CSS, user should be able to select different info from nav bar to see different pins on map and different info panel, refactor events class to not use javascript interface event key word.
- Should Do**: Responsive, animate page features, Limit n. of pubs to 20, XMLHttpRequest is called from the request\_helper, but also from event helper. (and maybe others). Factor out.
- Could Do**: more api's, geolocation, Persist data: favorite venues, pubs, create an evening plan, RSS feed google news, social media, functionality to share night out plan with friends - email, fb etc, allow user to amend range of search eg 1, 5, 10 miles, allow user to click on item and link to google search???
- Would Do**: Add a card... (placeholder)
- Done**: UX, Wireframes, MapWrapper, Mongo DB, write MVP, add button to submit, submit button for pos, HR - weather api help, Request Helper, Add a card... (placeholder)

## P4 Acceptance Criteria Test Plan

Acceptance Criteria	Expected Result/Output	Pass/Fail
User can bring up a map of an area by inputting a postal code	Users will obtain map information relative to post code entered	Pass
User can choose what kind of information relative to location they would like to see on map	Users will be able to click a button representing information type and see it represented on map	Pass
User will be able to get further information on selections made	User can click on map markers to see details	Pass
User can plan out their evening by saving selections to favourites	Users selections will be saved and data persisted	Pass
User can share plan for their evening out with friends using social media	User will be able to click on social media buttons to access their accounts and share their plan.	Fail
User can find out about events information for the area selected	Users will see a list of events taking place in the area in the next 24 hours	Pass

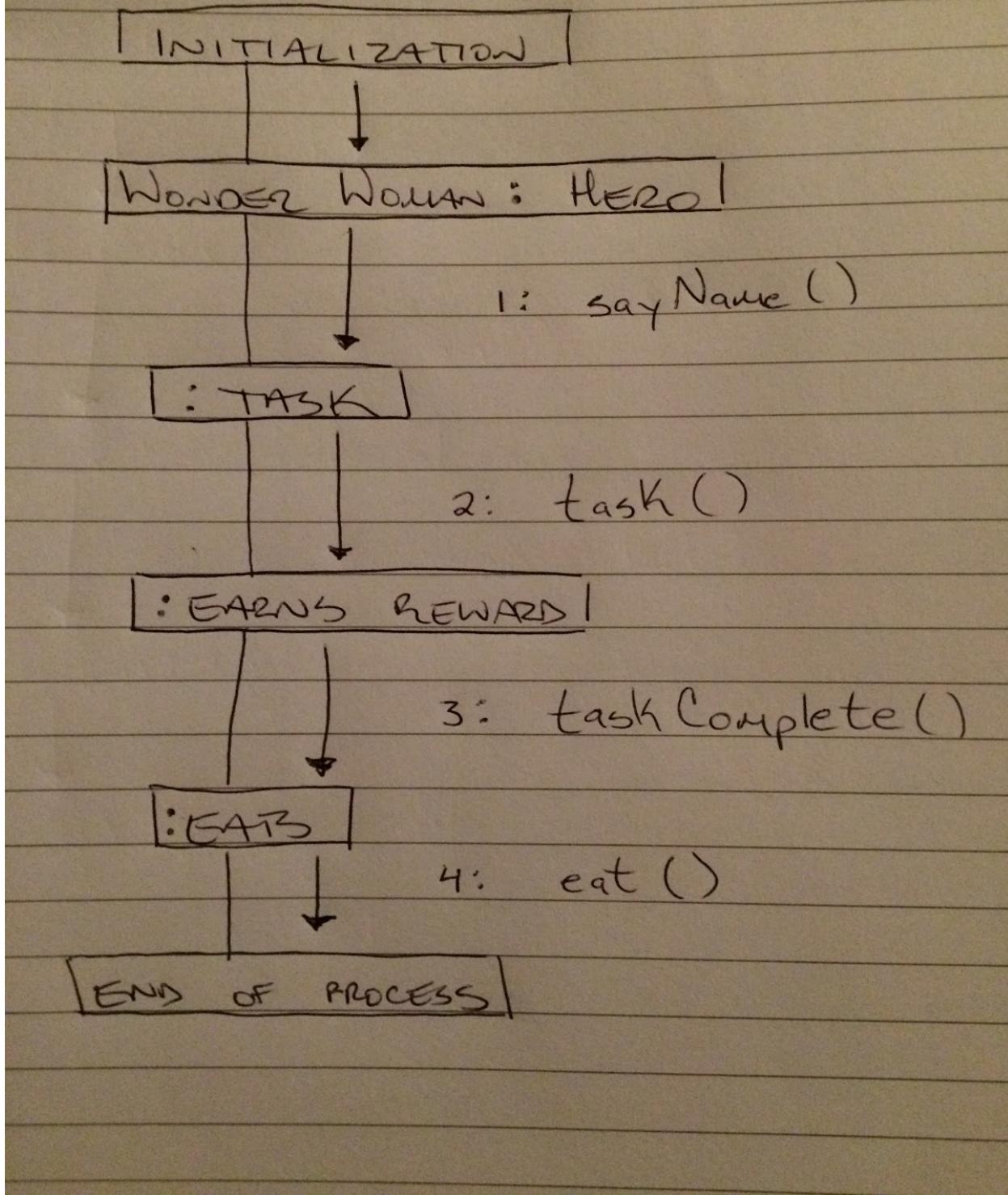
P5 User Sitemap

## P6 Wireframes

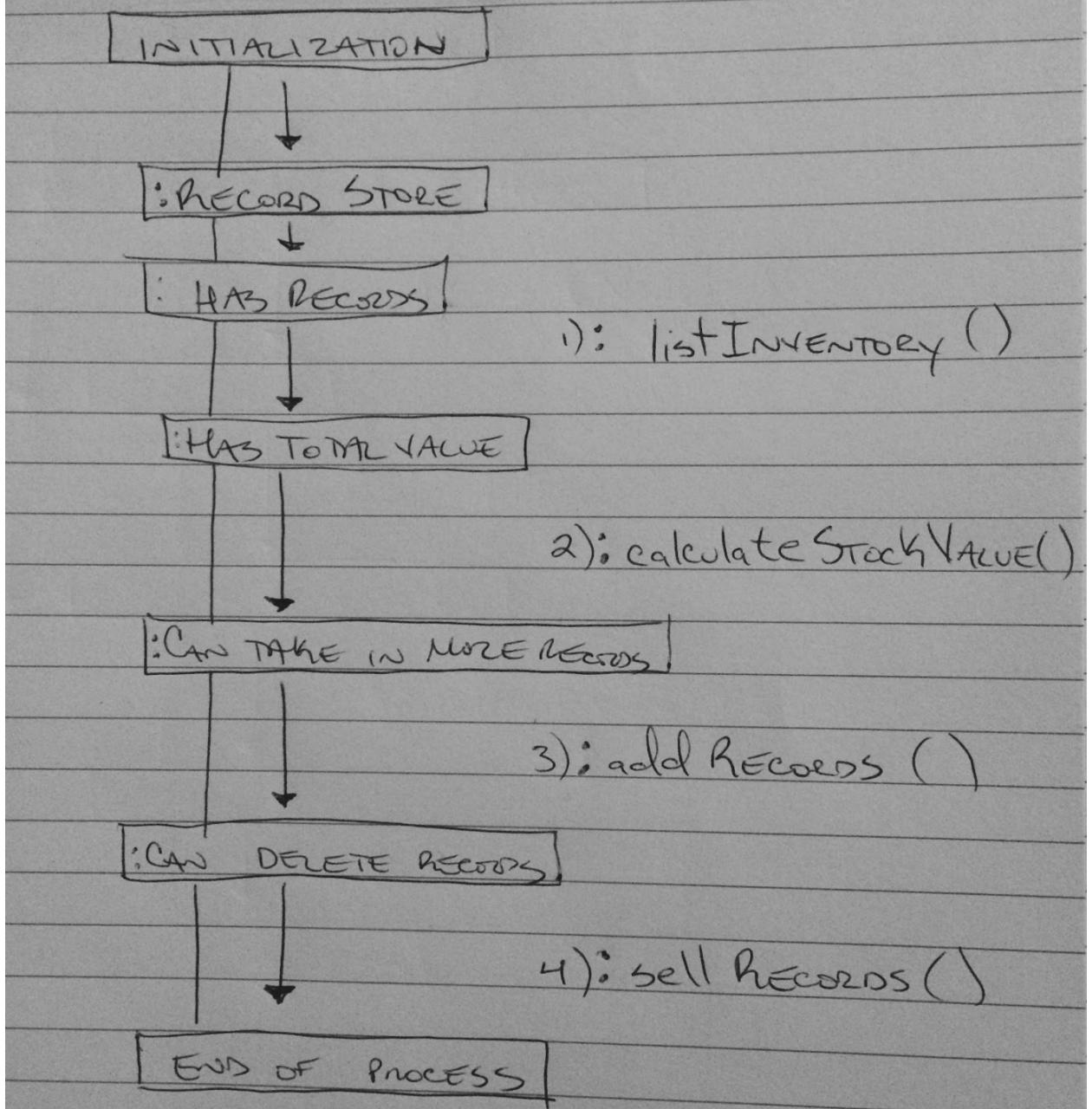


## P7 System Interaction Diagrams

# COLLABORATION DIAGRAM



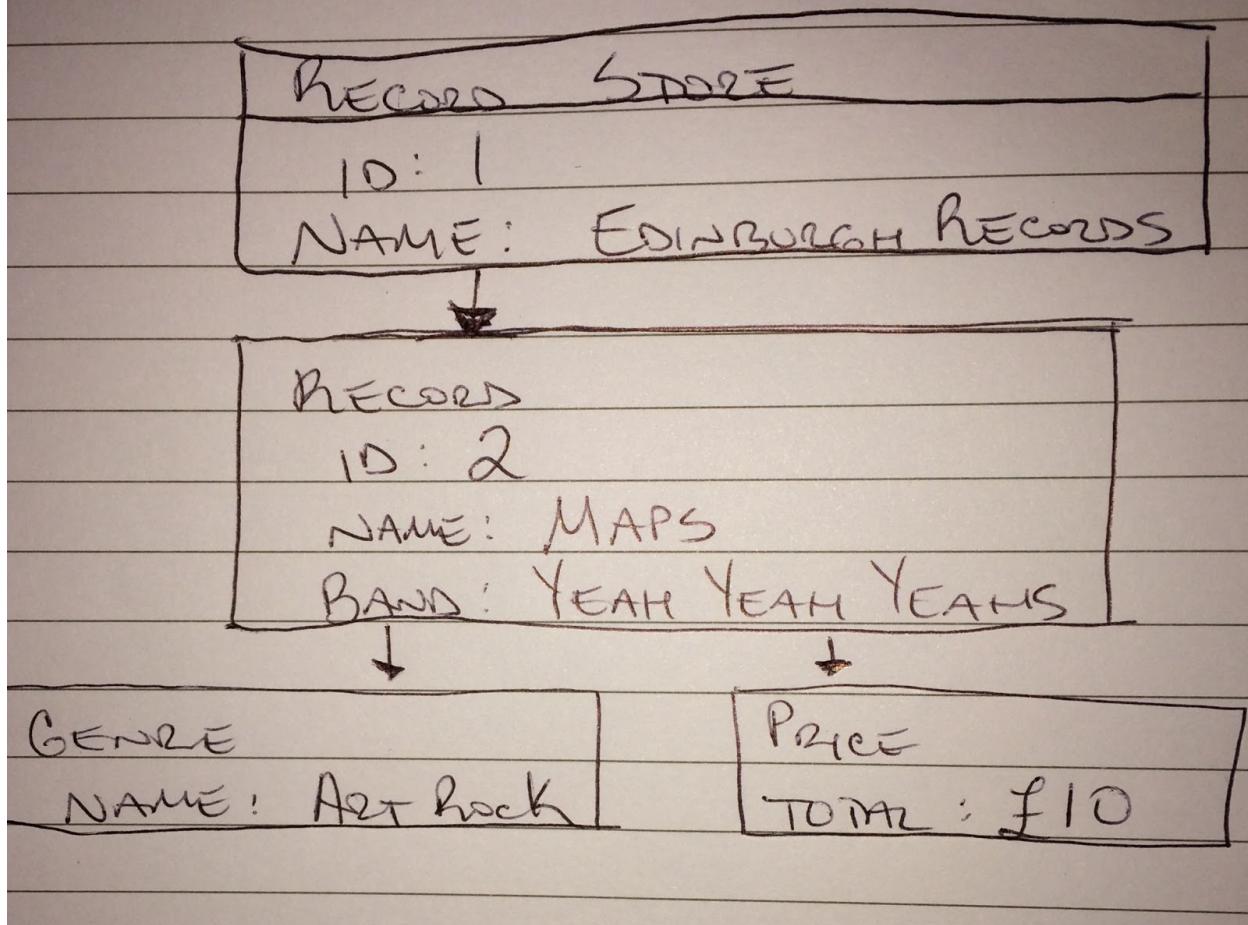
# COLLABORATION DIAGRAM



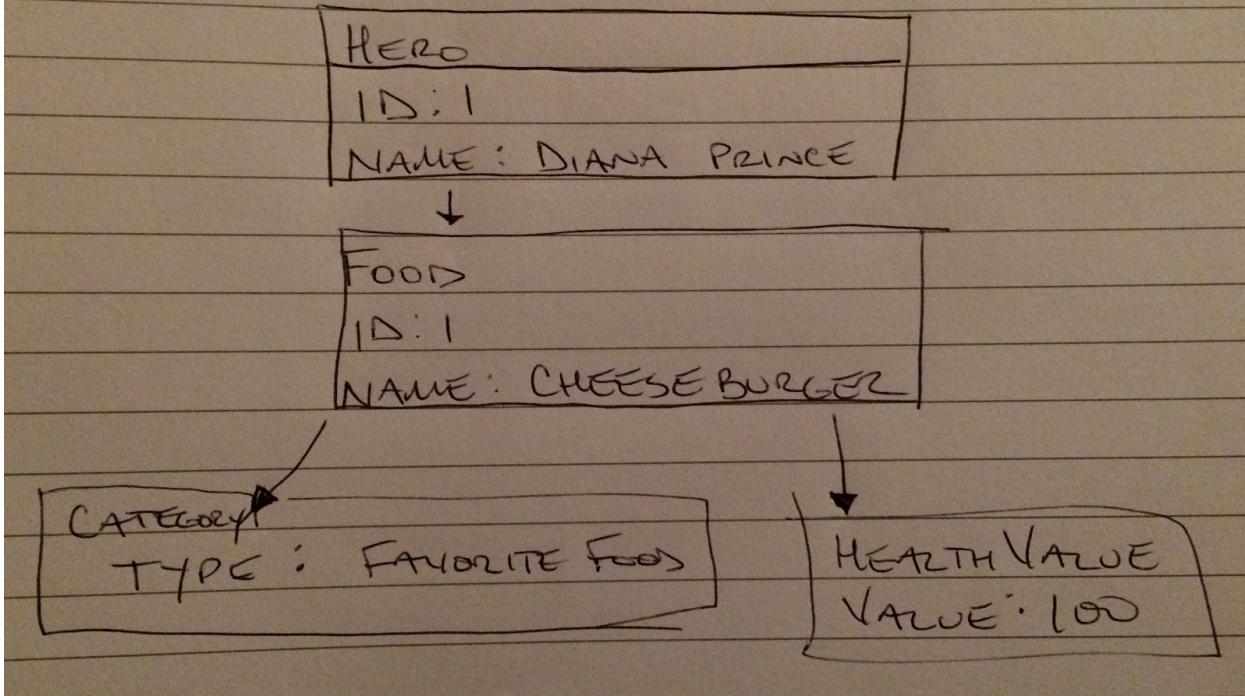
P8 Object diagrams

## OBJECT

## DIAGRAM



## OBJECT DIAGRAM



```
public class Ward {  
    public ArrayList<Patient> rooms;  
    public int capacity;  
    public Gender gender;  
  
    public Ward(Gender gender){  
        this.rooms = new ArrayList<>();  
        this.capacity = 10;  
        this.gender = gender;  
    }  
  
    public ArrayList<Patient> getRooms () {  
        return rooms;  
    }  
    public Gender getGender() { return gender; }  
    public int getCapacity() { return capacity; }  
  
    public void admitPatient(Patient patient) {  
        if (rooms.size() < capacity) {  
            rooms.add(patient);  
        }  
    }  
  
    public void dischargePatient(Patient patient) { rooms.remove( patient ); }  
  
    public void admitCorrectGender(Patient patient) {  
        if (patient.gender == this.gender) {  
            rooms.add( patient );  
        }  
    }  
}
```

The admitPatient algorithm compares the patients currently admitted to a room, to the capacity of the room and will not allow additional patients to be admitted. This helped me reach my MVP for my Java project.

The admitCorrectGender algorithm uses an if statement to compare two enums, the patient's gender with the room's designated gender. This would allow only female patients to be admitted to the maternity ward. This was an extension I was able to do above and beyond the MVP for my Java project.

P 10 PSEUDOCODE

```

need to make adoptions/new page
2 drop down boxes for heroes and animals

once selections made
adoption button which will add new pair to adoptions page

will need animal id joined to hero id to create adoption function |

```

## P 11 Screenshot of solo project

The screenshot shows the Android Studio interface. On the left, the project structure is displayed under the 'app' folder. It includes 'manifests', 'java' (containing packages like 'com.example.patientmanagement' with classes such as 'Gender', 'Hospital', 'Nutrition', 'Patient', 'Status', and 'Ward'), 'res', and 'Gradle Scripts'. On the right, a code editor is open for the 'Ward.java' file. The code defines a 'Ward' class with fields for rooms (ArrayList<Patient>), capacity (int), and gender (Gender). It includes methods for admitting patients (admitPatient) and discharging patients (dischargePatient). The 'admitPatient' method checks if the room size is less than capacity before adding the patient. The 'dischargePatient' method removes the patient from the rooms. The 'admitCorrectGender' method adds the patient to the rooms if their gender matches the ward's gender. The code editor has line numbers from 3 to 51.

```

Ward
import java.util.ArrayList;

/*
 * Created by allysonwilson on 9/22/17.
 */

public class Ward {
    public ArrayList<Patient> rooms;
    public int capacity;
    public Gender gender;

    public Ward(Gender gender) {
        this.rooms = new ArrayList<Patient>();
        this.capacity = 10;
        this.gender = gender;
    }

    public ArrayList<Patient> getRooms () {
        return rooms;
    }

    public Gender getGender() { return gender; }

    public int getCapacity() { return capacity; }

    public void admitPatient(Patient patient) {
        if (rooms.size() < capacity) {
            rooms.add(patient);
        }
    }

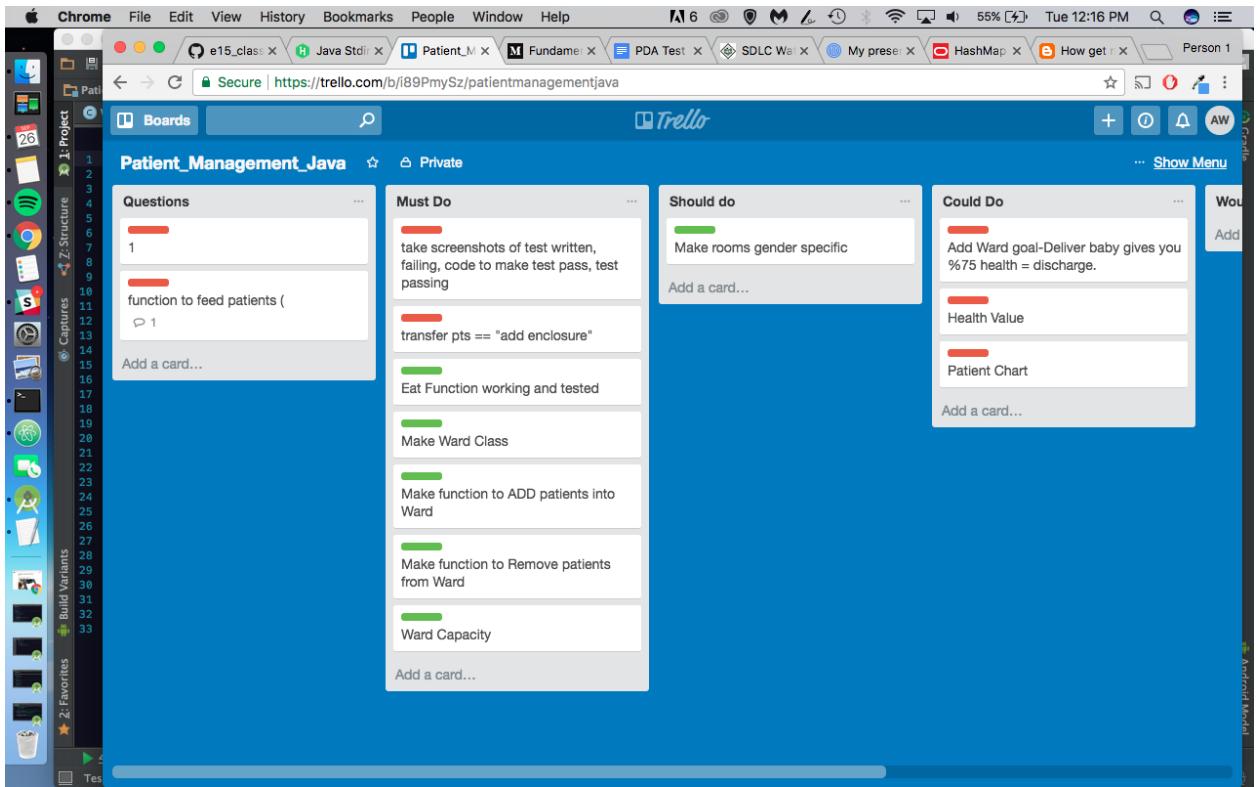
    public void dischargePatient(Patient patient) { rooms.remove( patient ); }

    public void admitCorrectGender(Patient patient) {
        if (patient.gender == this.gender) {
            rooms.add( patient );
        }
    }
}

```

[https://github.com/allysonwilson/Patient\\_Management\\_Java\\_Project](https://github.com/allysonwilson/Patient_Management_Java_Project)

## P 12 photo of planning



## P 13 USER INPUT



## P 14 DATA PERSISTENCE

## P 15 RESULTS/ FEEDBACK TO USER

The screenshot shows a web browser window on a Mac OS X desktop. The title bar reads "Chrome File Edit View History Bookmarks People Window Help". The address bar shows the URL "localhost:4567/animals". The main content area displays a green header with three tabs: "Heroes", "Animals" (which is selected), and "Adoptions". Below the tabs is a large green section titled "Animals" in bold green text. Three cards are displayed in a grid:

Species	Adoptable	Adoption Month
Basking Shark	yes	August
Golden Oriole	no	September
Adder	no	July

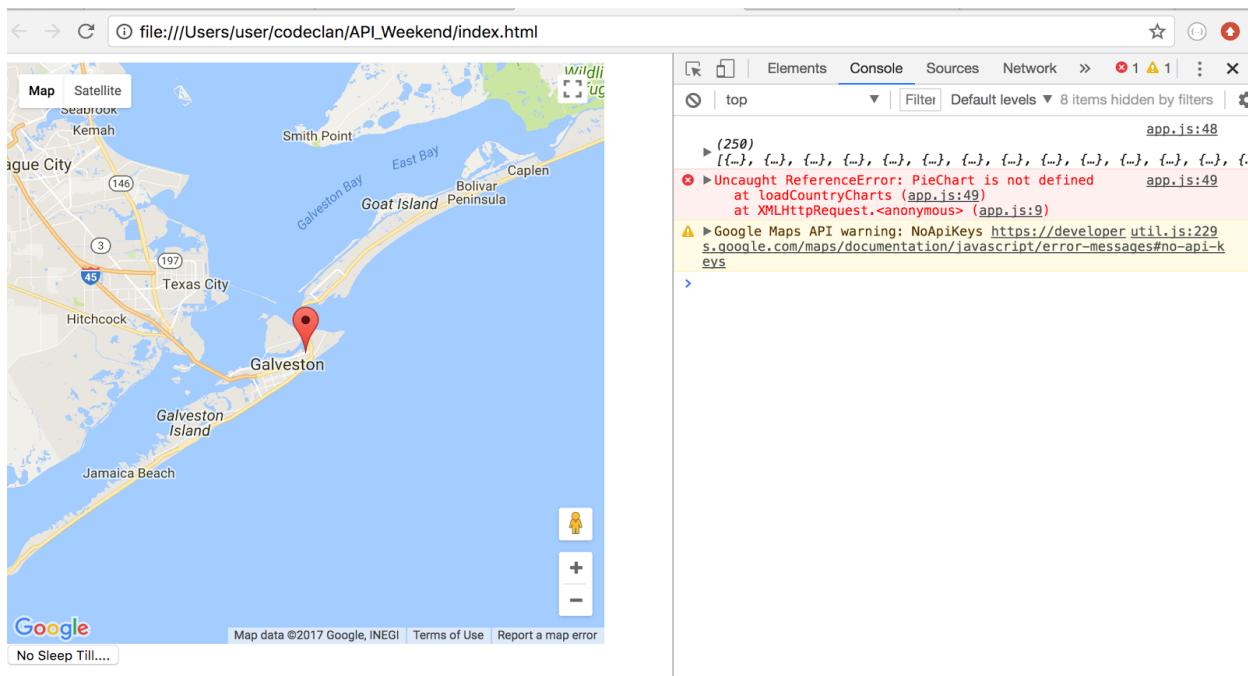
A vertical toolbar on the left side of the desktop includes icons for Mail, Calendar, Notes, Spotify, Google Chrome, Safari, and Adopt. The "Adopt" icon is highlighted with a blue background.

## P 16 Code that implements API

```
53
54 var initialize = function(){
55     var mapDiv = document.getElementById('main-map');
56     var center = { lat: 29.313 , lng: -94.7766 };
57     var mainMap = new MapWrapper(mapDiv, center, 10);
58     mainMap.addMarker(center,"Galveston,TX my birthplace.");
59     // mainMap.addClickEvent();
60
61     var noSleepTillBrooklyn = function () {
62         var brooklyn = { lat:40.665535 , lng: -73.96974};
63         mainMap.googleMap.setCenter(brooklyn);
64     }
65
66     var brooklynButton = document.getElementById('brooklyn');
67     brooklynButton.addEventListener('click', noSleepTillBrooklyn);
68 }
69
70 window.addEventListener('load', initialize);
71
```

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Country Maps</title>
    <!-- <script src="https://code.highcharts.com/highcharts.js"></script> -->
    <!-- <script src="https://code.highcharts.com/modules/exporting.js"></script> -->
    <!-- <script src="piechart.js"></script> -->
    <script src="columnchart.js"></script> -->
    <link rel='stylesheet' type='text/css' href='public/style.css'>
```

P 16 API being used whilst program is running.



## P17 Bug Tracking Report

User must be able to see local weather forecast	Failed	Used another api that allowed post code to be a search query rather than latitude and longitude	Passed
Event has a location and a description	Failed	Had to access the location by using different keys of JSON object	Passed
User must be able to save an event, pub, or restaurant to favourites	Failed	Had to create model objects in MongoDB	Passed
Events must be shown for next 24 hours	Failed	Had to find correct way to query API and structure the URL	Passed
User must be able to delete from favourites	Failed	Create a remove button with a function to remove item from MongoDB	Passed

## P 18 Testing

### Test code

```
1 var assert = require("assert");
2 var Hero = require("../hero.js");
3 var Task = require("../task.js");
4 var Food = require("../food.js");
5
6 beforeEach(function () {
7   wonderWoman = new Hero("Diana Prince", 100, "cheese burger");
8   flyAirplane = new Task(5,10, 100);
9   eatCheeseBurger = new Task(-5 , 100, 5);
10 });
11
12 describe("Hero", function() {
13
14   it("should have a name", function() {
15     assert.strictEqual(wonderWoman.name, "Diana Prince");
16   });
17
18   it("should have a favouriteFood", function() {
19     assert.strictEqual(wonderWoman.favouriteFood, "cheese burger");
20   });
21
22   it("should have a health value", function() {
23     assert.strictEqual(wonderWoman.healthValue, 100);
24   });
25
26   it("should be able to say name", function() {
27     assert.strictEqual(wonderWoman.sayName(), "My name is Diana Prince!");
28   });
29
30   it("should be able add task", function() {
31     wonderWoman.addTask(flyAirplane);
32     assert.strictEqual(wonderWoman.todo.length, 1);
33   });
34
35   it("should be able remove task", function() {
36     wonderWoman.addTask(flyAirplane);
37     wonderWoman.addTask(flyAirplane);
38     wonderWoman.addTask(eatCheeseBurger);
39     wonderWoman.removeTask(0, 1);
40     assert.strictEqual(wonderWoman.todo.length, 2);
41   });
42 })
```

### Tests Failing

```
→ hero_rat git:(master) ✘ mocha specs --reporter nyan
8  _-----'-----'
3  | /_\_\
0  ~|_( x .x)
     ""   ""

8 passing (17ms)
3 failing

1) Hero
    should be able to say name:
    TypeError: wonderWoman.sayName is not a function
    at Context.<anonymous> (specs/hero_spec.js:27:36)

2) Hero
    should be able add task:
    TypeError: wonderWoman.addTask is not a function
    at Context.<anonymous> (specs/hero_spec.js:31:17)

3) Hero
    should be able remove task:
    TypeError: wonderWoman.addTask is not a function
    at Context.<anonymous> (specs/hero_spec.js:36:17)
```

## Corrected Code

```
hero.js          hero_spec.js
1 var Hero = function (name, health, favouriteFood) {
2     this.name = name;
3     this.healthValue = 100;
4     // shouldn't I be able to set this = to healthValue?
5     this.favouriteFood = favouriteFood;
6     this.todo = [];
7     // this.wallet = [];
8 };
9
10 Hero.prototype.sayName = function () {
11     return ("My name is " + this.name + "!");
12 };
13
14 Hero.prototype.addTask = function (task) {
15     this.todo.push(task);
16 };
17
18 Hero.prototype.removeTask = function (index , deleteCount) {
19     this.todo.splice(index , deleteCount);
20 };
21
22 Hero.prototype.eat = function () {
23     var newHealthValue = food.replenishmentValue += this.healthValue;
24     return newHealthValue;
25 };
26
```

The test code passing

```
  "devDependencies": {
    "mocha": "^4.0.1"
  },
  "dependencies": {},
  "repository": {
    "type": "git",
    "url": "git+https://github.com/allysonwilson/hero_rat.git"
  },
  "bugs": {
    "url": "https://github.com/allysonwilson/hero_rat/issues"
  },
  "homepage": "https://github.com/allysonwilson/hero_rat#readme",
  "description": ""
}
```

Is this ok? (yes)

```
→ hero_rat git:(master) ✘ mocha specs --reporter nyan
```

```
11  _--_---_---_,-----,
0  _-_---_---_ | / \_/\_
0  _-_---_---_~|_( ^_.^)
      ""   ""
```

```
11 passing (14ms)
```

```
→ hero_rat git:(master) ✘
```

---