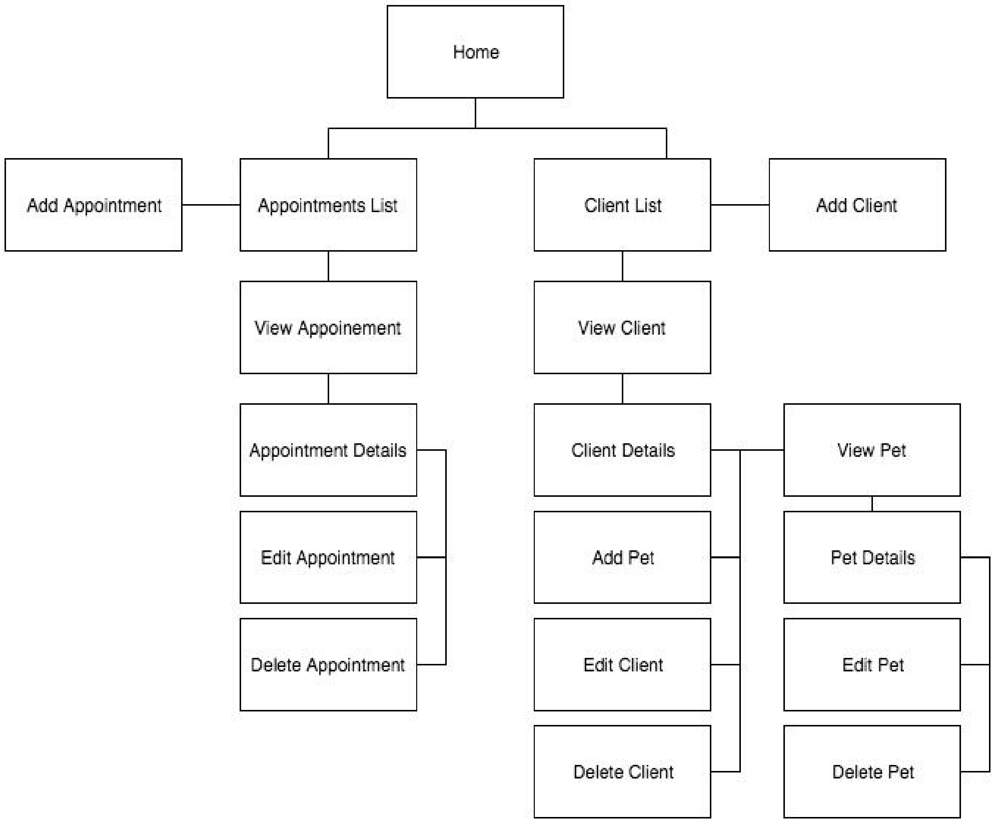
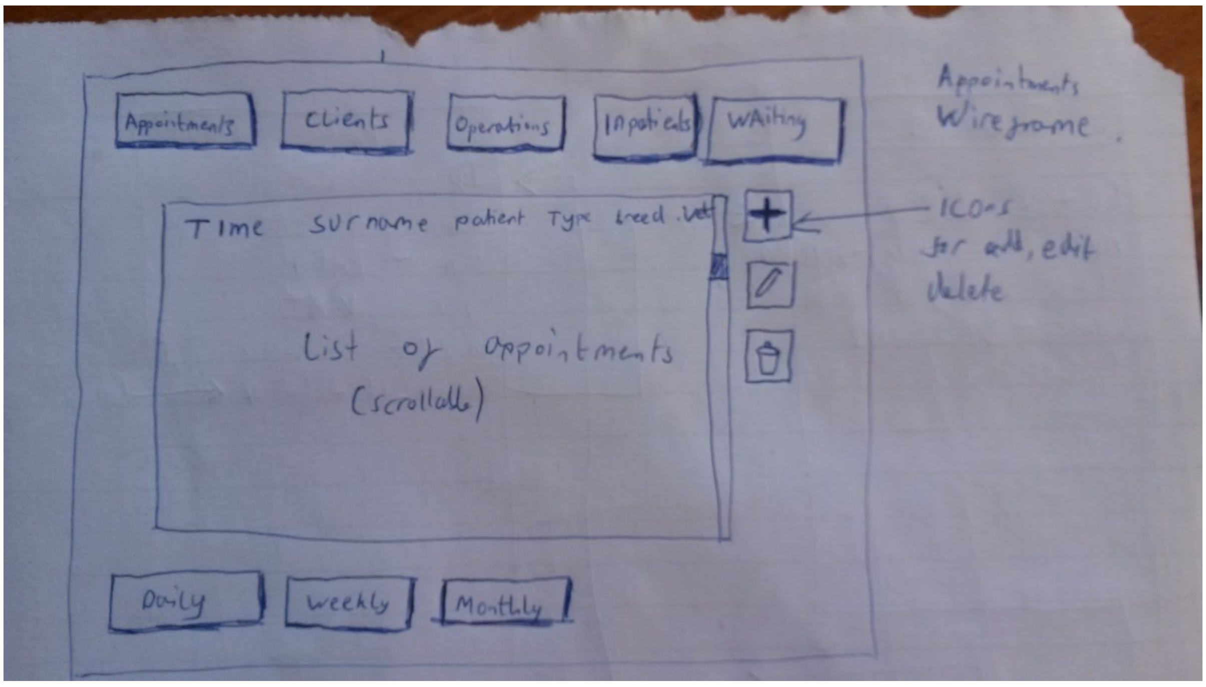
Chris Hunter - Project Unit(SQA PDA: Software Development)

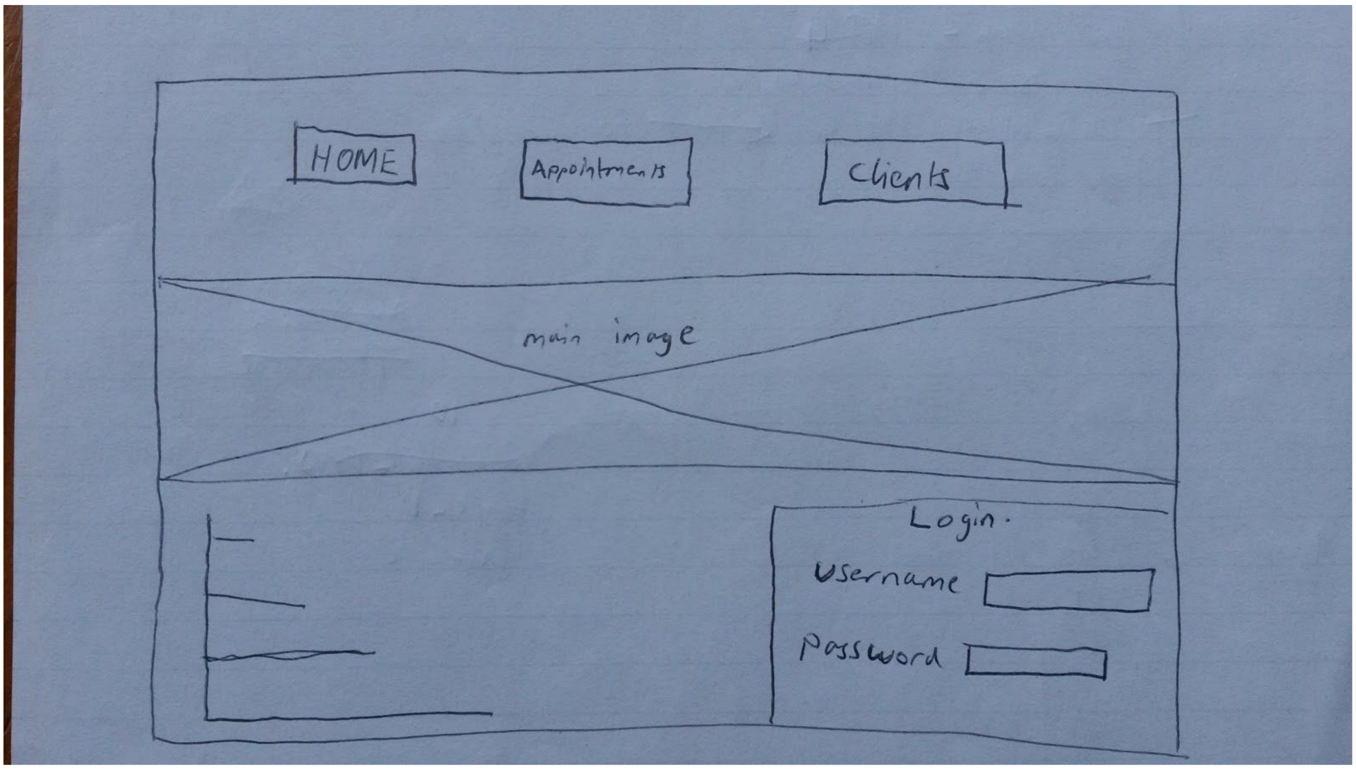
P 5. Site Map



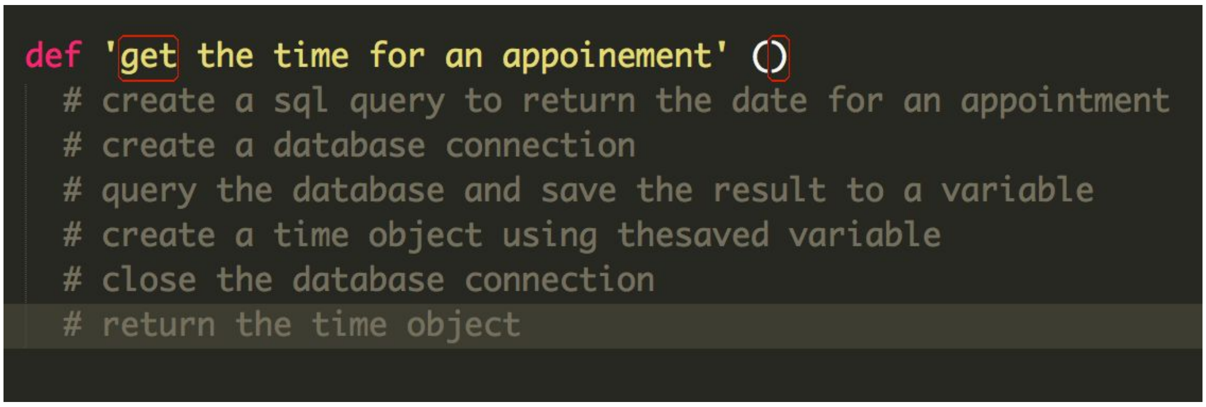
P 6. (i) Wireframe


Appointments
Clients
Operations
Inpatients
Waiting
Time surname patient Type breed vet
List of appointments (scrollable)
Daily
Weekly
Monthly
Appointments Wireframe.
Icons for add, edit delete

P 6. (ii) Wireframe

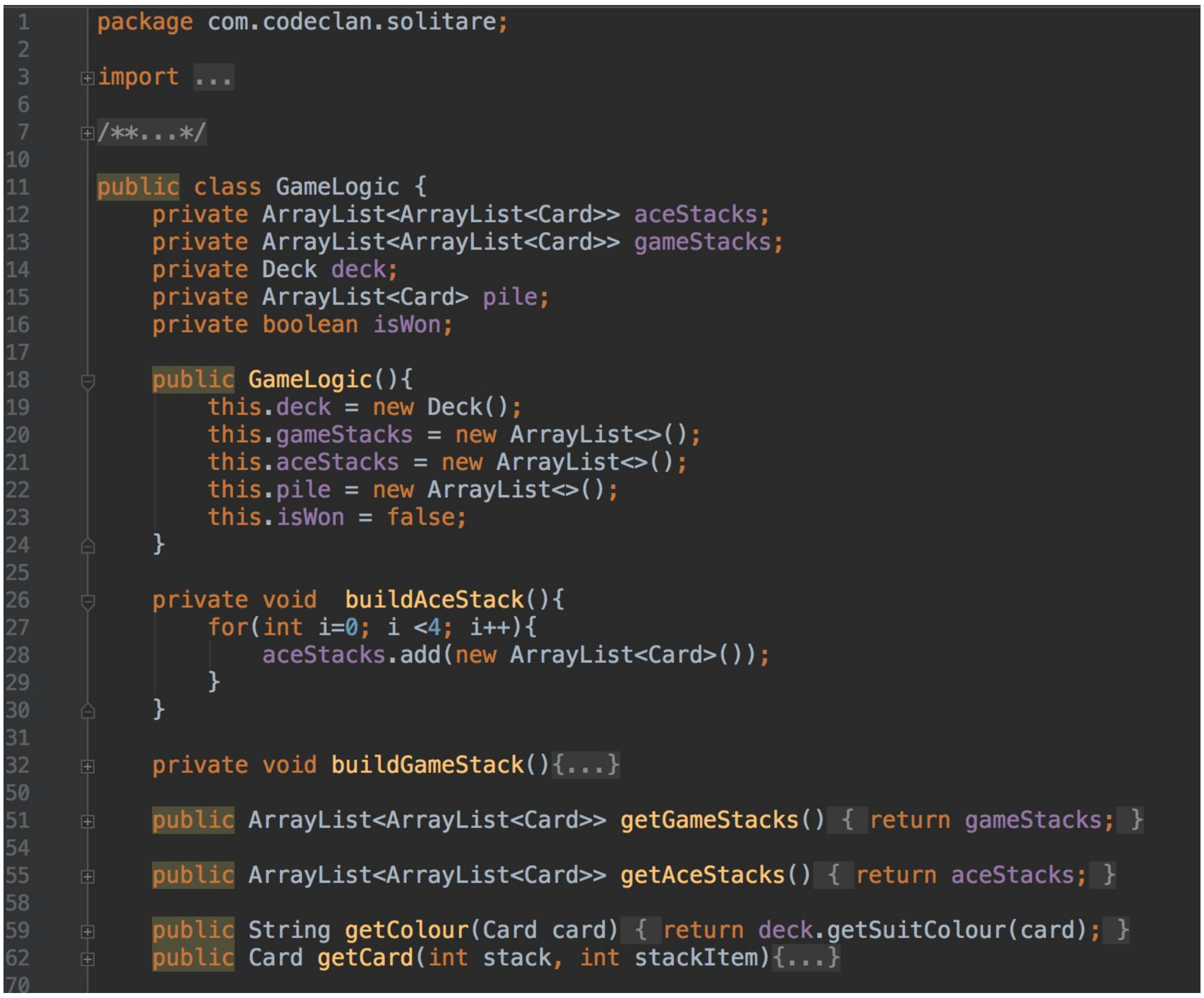


P 10. Pseudocode



```
def 'get the time for an appoinement' ()
  # create a sql query to return the date for an appointment
  # create a database connection
  # query the database and save the result to a variable
  # create a time object using thesaved variable
  # close the database connection
  # return the time object
```

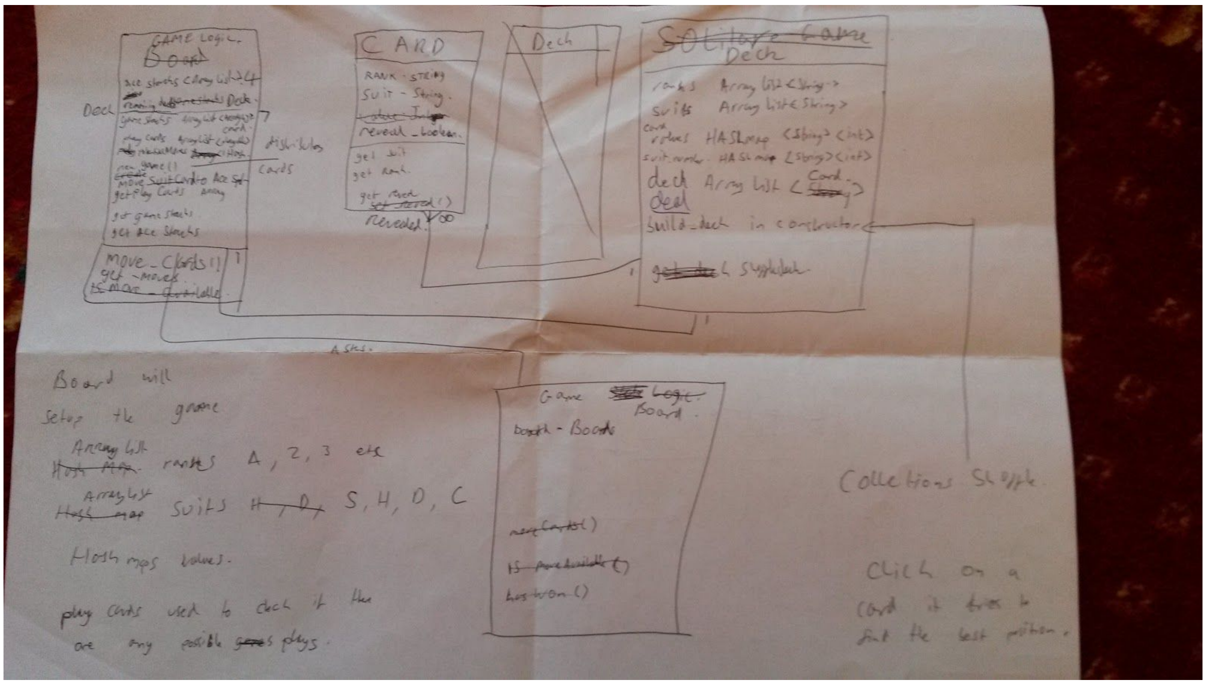
P 11 Screenshot of personal project and GitHub Link:
(https://github.com/Orkem/CC_Solitare)



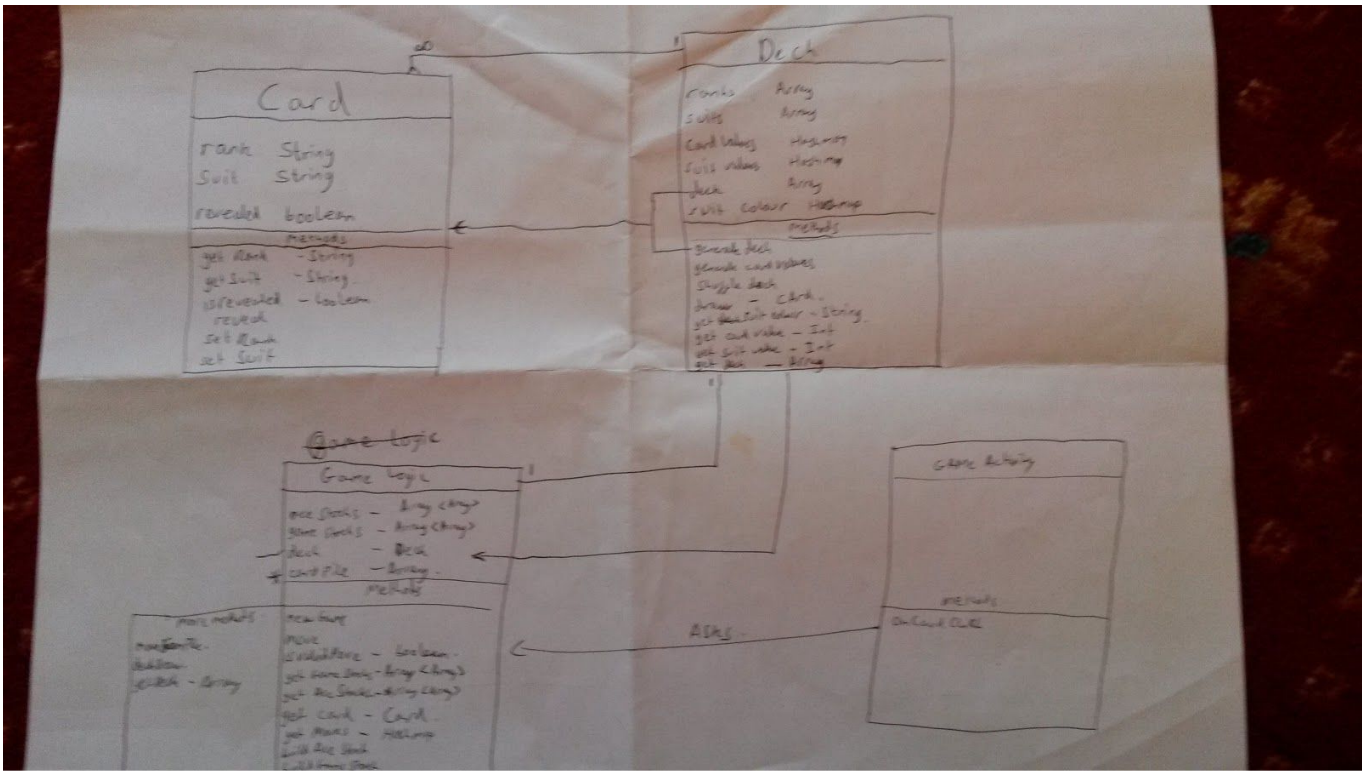
```java
package com.codeclan.solitare;

import ...

/**...*/

public class GameLogic {
    private ArrayList<ArrayList<Card>> aceStacks;
    private ArrayList<ArrayList<Card>> gameStacks;
    private Deck deck;
    private ArrayList<Card> pile;
    private boolean isWon;

    public GameLogic(){
        this.deck = new Deck();
        this.gameStacks = new ArrayList<>();
        this.aceStacks = new ArrayList<>();
        this.pile = new ArrayList<>();
        this.isWon = false;
    }

    private void  buildAceStack(){
        for(int i=0; i <4; i++){
            aceStacks.add(new ArrayList<Card>());
        }
    }

    private void buildGameStack(){...}

    public ArrayList<ArrayList<Card>> getGameStacks() { return gameStacks; }

    public ArrayList<ArrayList<Card>> getAceStacks() { return aceStacks; }

    public String getColour(Card card) { return deck.getSuitColour(card); }
    public Card getCard(int stack, int stackItem){...}
```

P 12 Pictures/Screenshots of different planning stages to show development

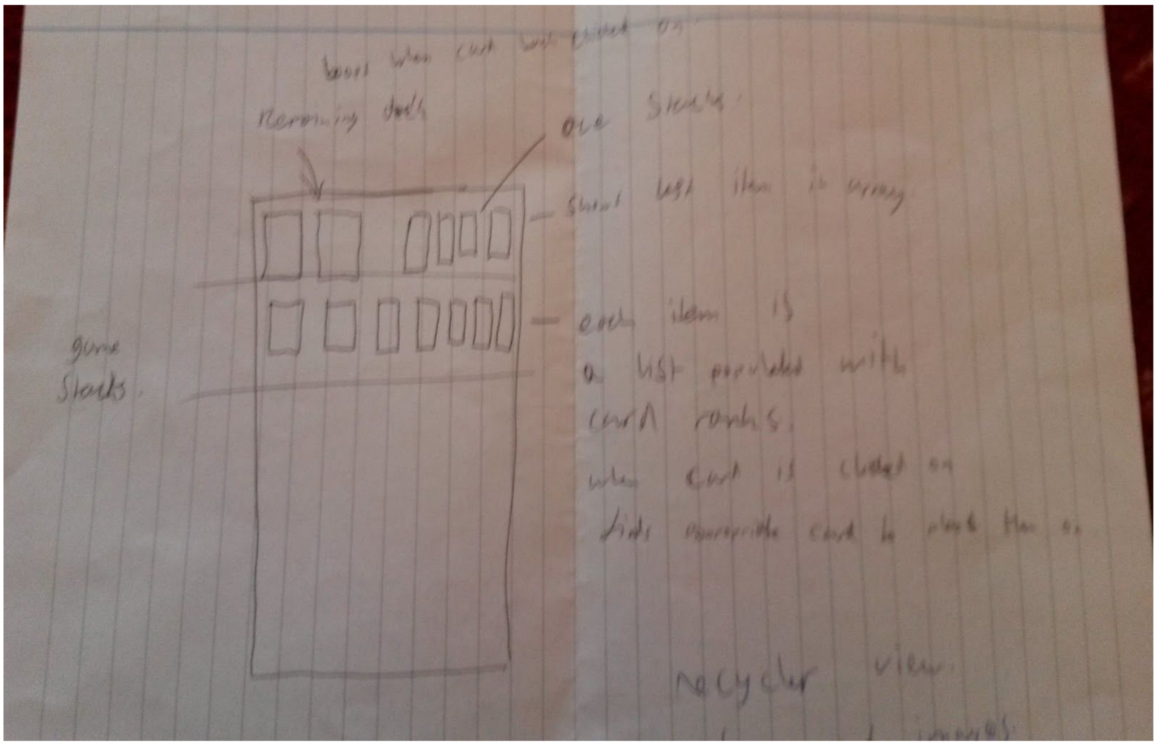
Class Diagram initial design



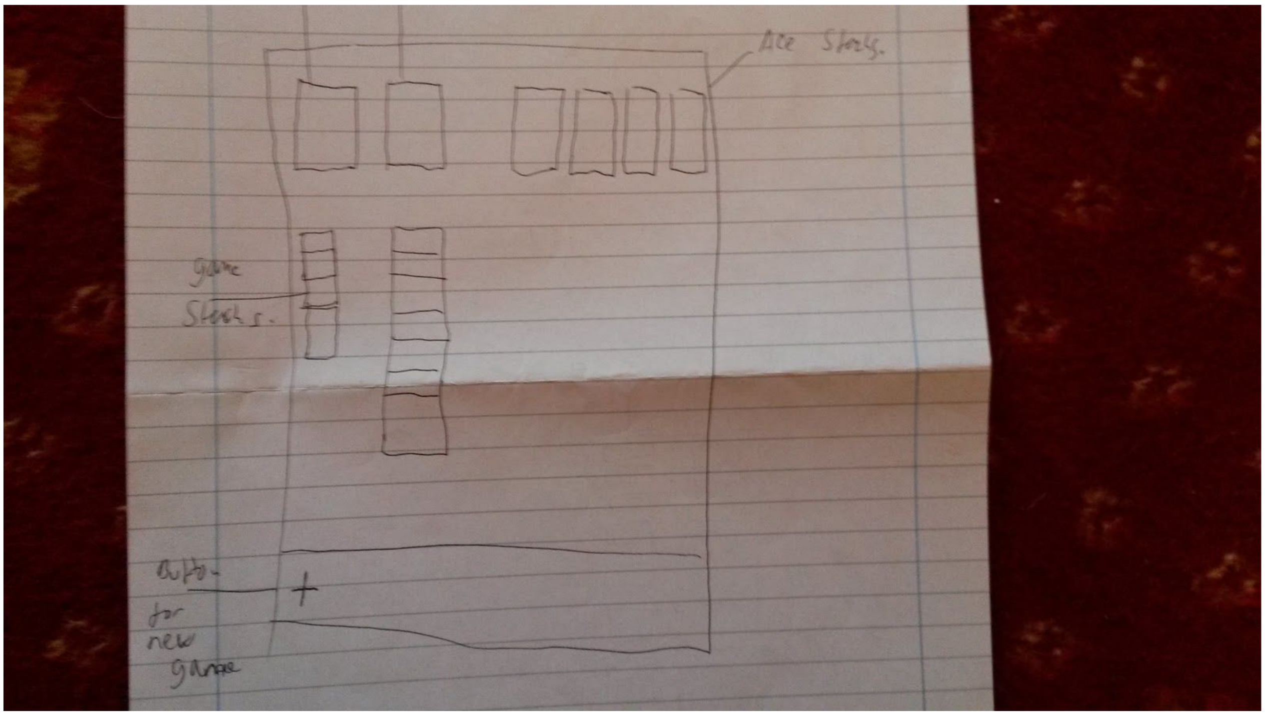
Class Diagram after



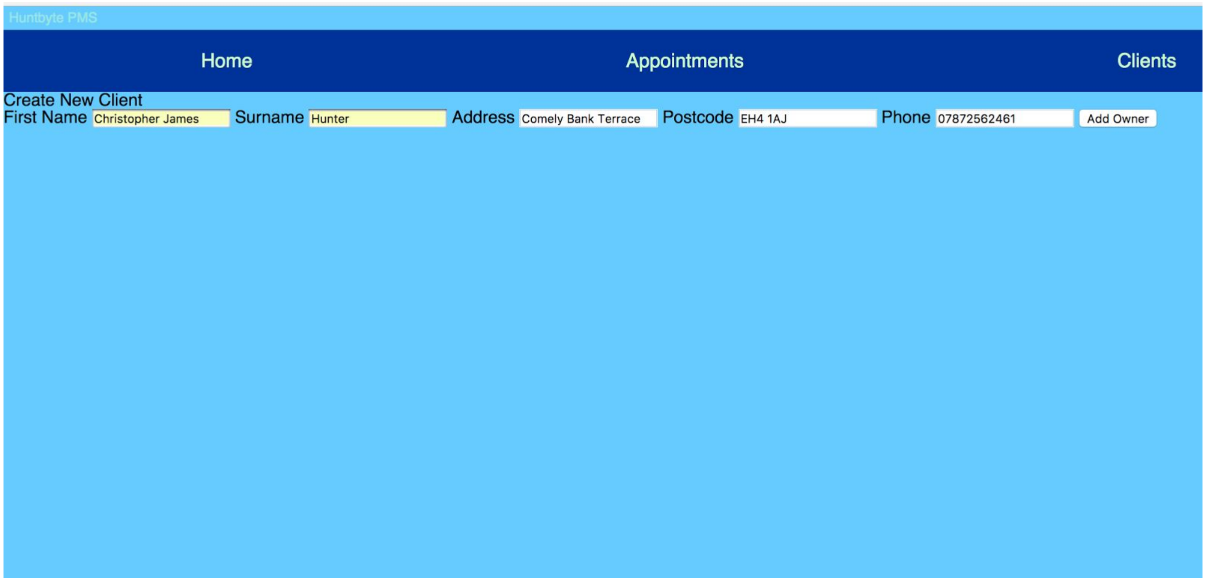
WireFrame of GameActivity initial design


[illegible] when card [illegible] clicked on
Remaining deck
ace Stacks
Shows last item in array
each item is a list populated with card ranks. when card is clicked on finds appropriate card to place [illegible]
game Stacks.
Recycler view.

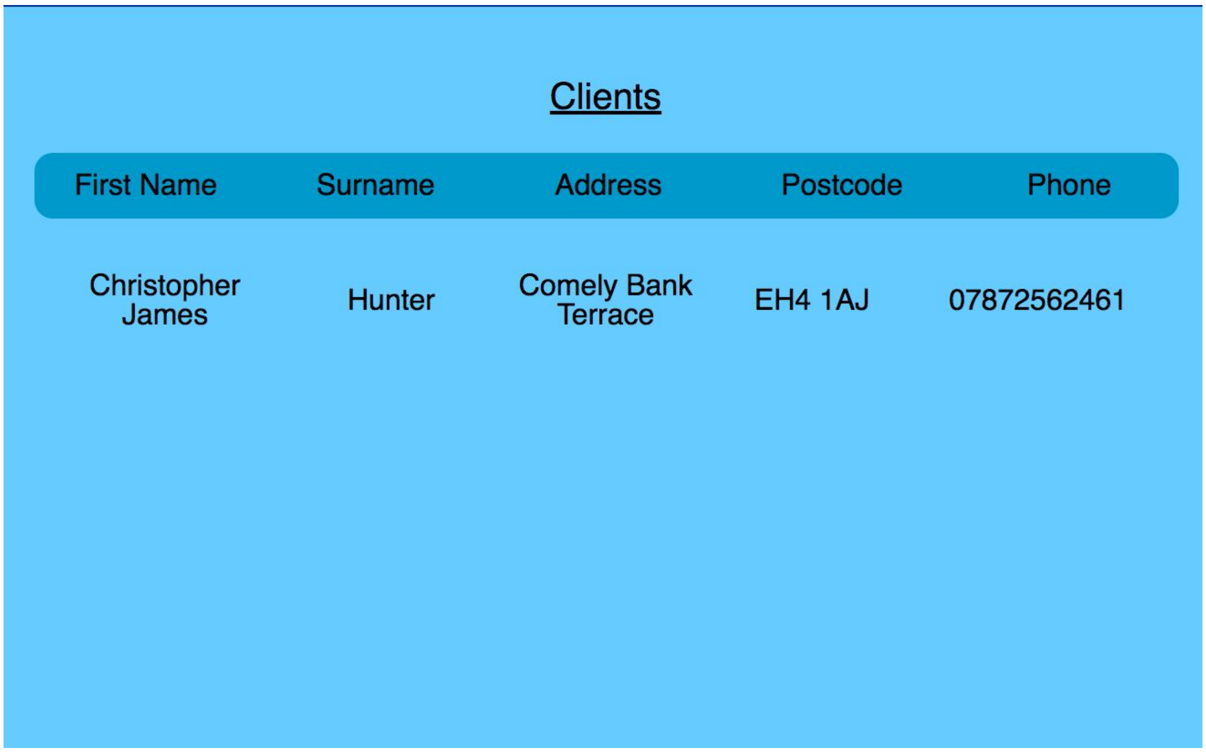
WireFrame of GameActivity After



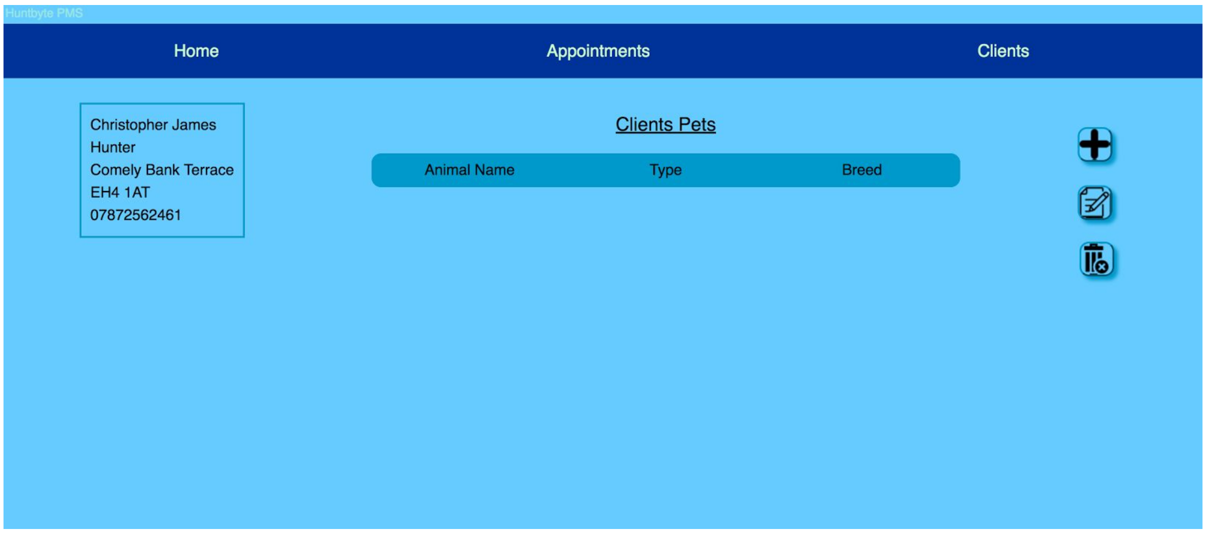
P 13. User input being processed according to design requirements.



Create New Client
First Name Christopher James Surname Hunter Address Comely Bank Terrace Postcode EH4 1AJ Phone 07872562461 Add Owner



First Name	Surname	Address	Postcode	Phone
Christopher James	Hunter	Comely Bank Terrace	EH4 1AJ	07872562461

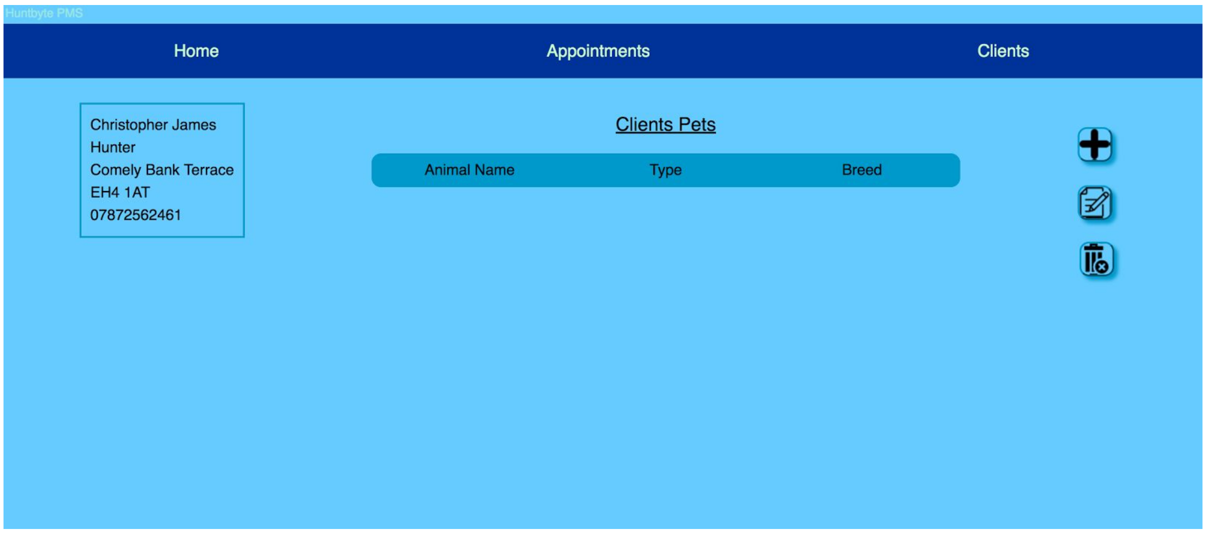


Christopher James
Hunter
Comely Bank Terrace
EH4 1AT
07872562461

Clients Pets

Animal Name	Type	Breed

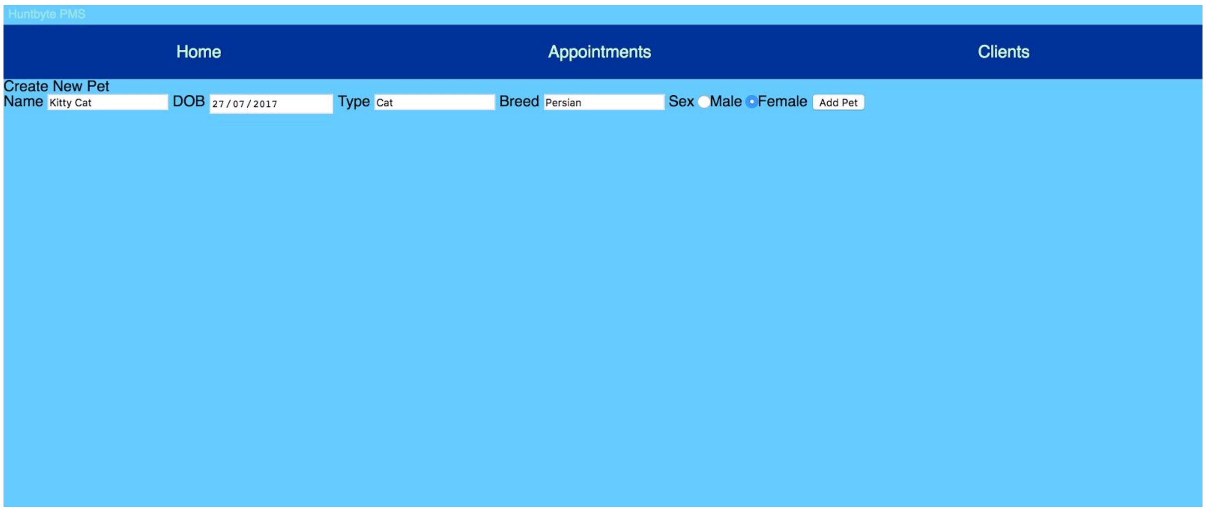
P 14. Data persistence



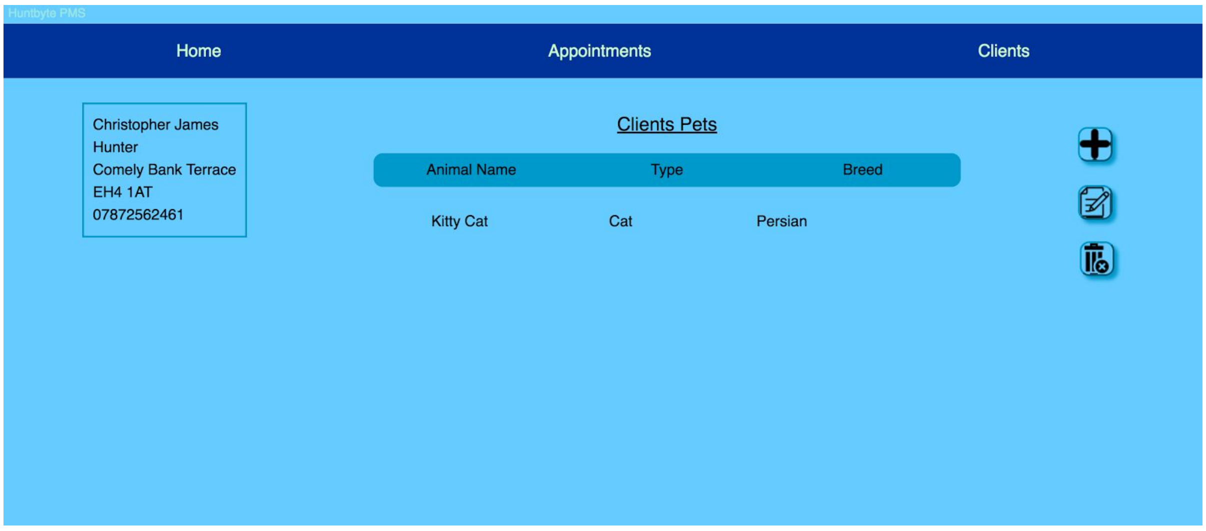
Christopher James
Hunter
Comely Bank Terrace
EH4 1AT
07872562461

Clients Pets

Animal Name	Type	Breed



Create New Pet
Name Kitty Cat DOB 27/07/2017 Type Cat Breed Persian Sex Male [✓ Female] Add Pet

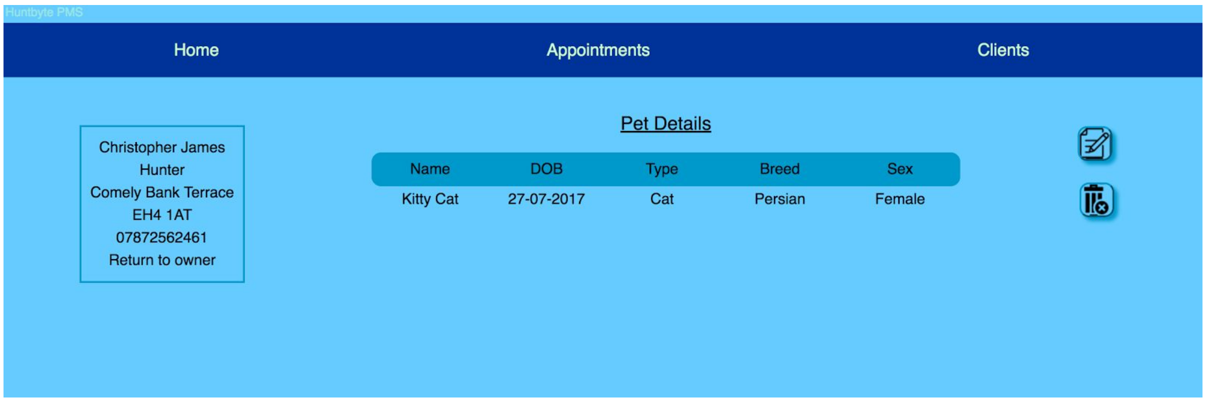

Huntbyte PMS

Home | Appointments | Clients

Christopher James Hunter
Comely Bank Terrace
EH4 1AT
07872562461

Clients Pets

Animal Name	Type	Breed
Kitty Cat	Cat	Persian


Huntbyte PMS

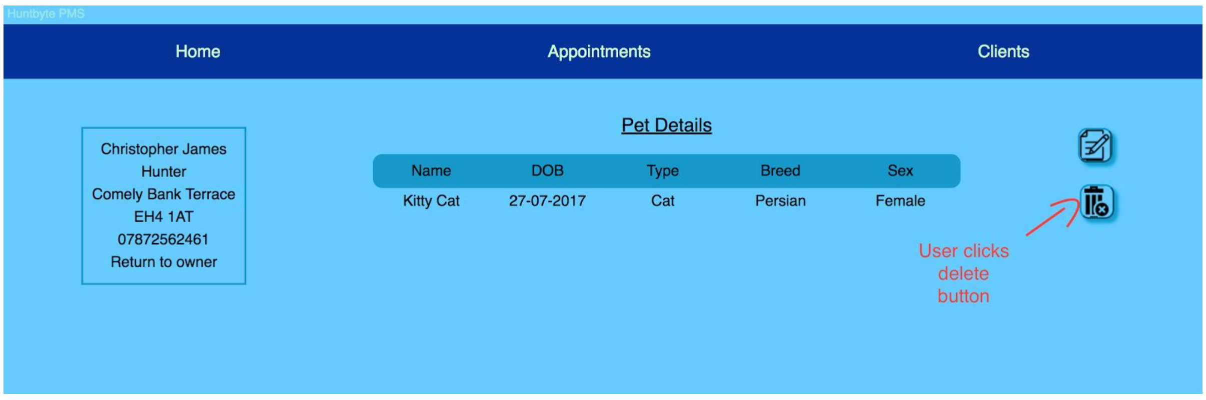
Home | Appointments | Clients

Christopher James Hunter
Comely Bank Terrace
EH4 1AT
07872562461
Return to owner

Pet Details

Name	DOB	Type	Breed	Sex
Kitty Cat	27-07-2017	Cat	Persian	Female

P 15. Output of results and feedback to user


Huntbyte PMS

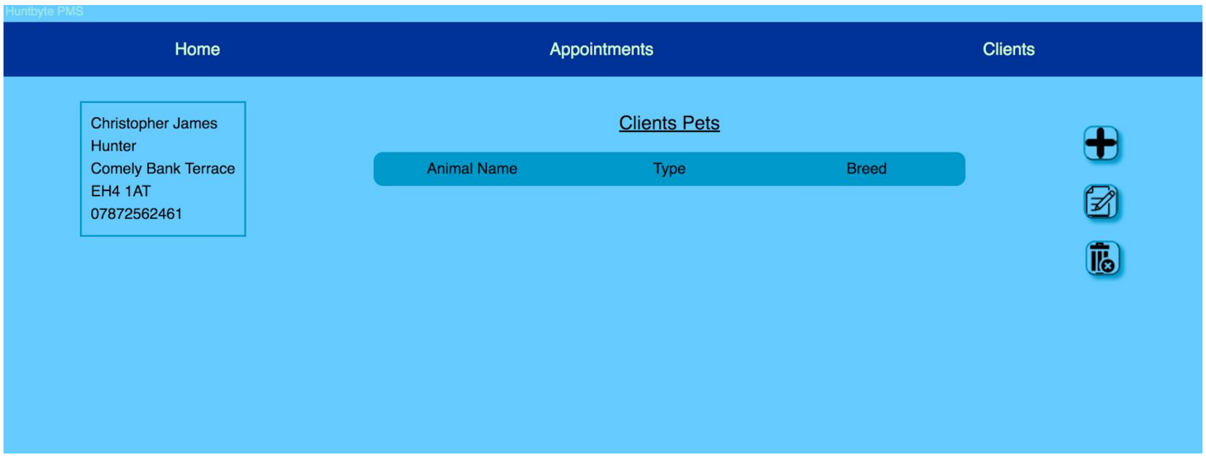
Home | Appointments | Clients

Christopher James Hunter
Comely Bank Terrace
EH4 1AT
07872562461
Return to owner

Pet Details

Name	DOB	Type	Breed	Sex
Kitty Cat	27-07-2017	Cat	Persian	Female

User clicks delete button

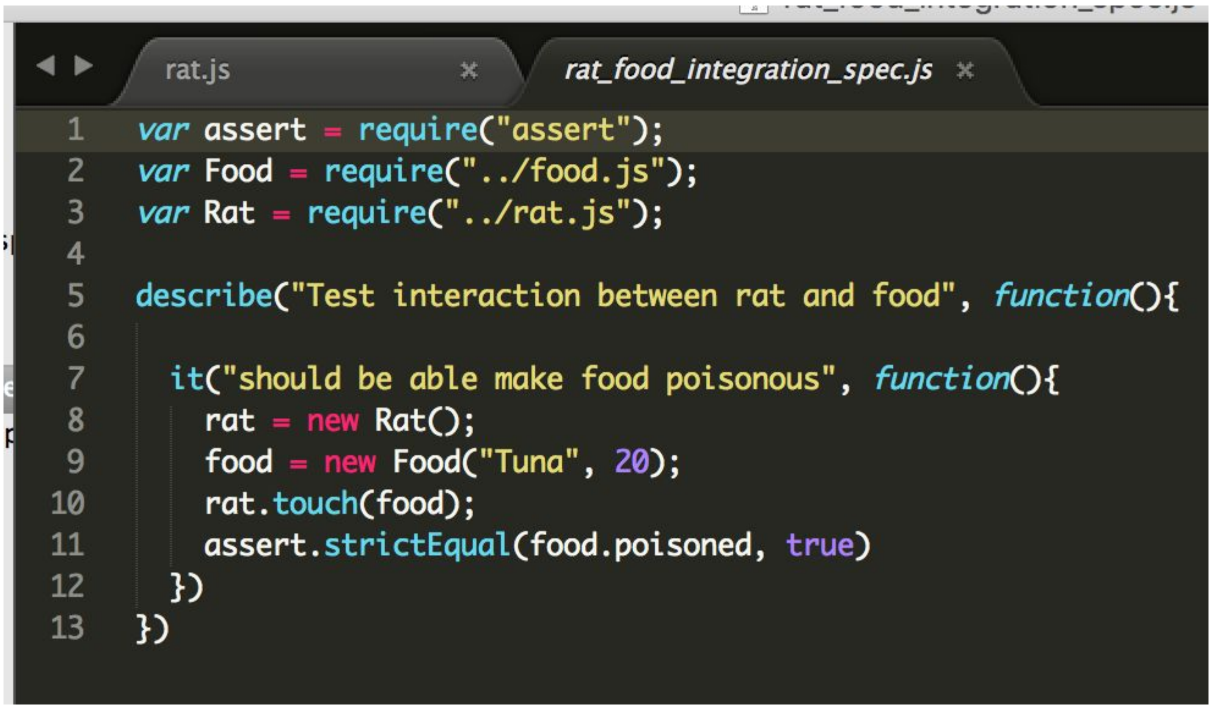


Christopher James Hunter
Comely Bank Terrace
EH4 1AT
07872562461

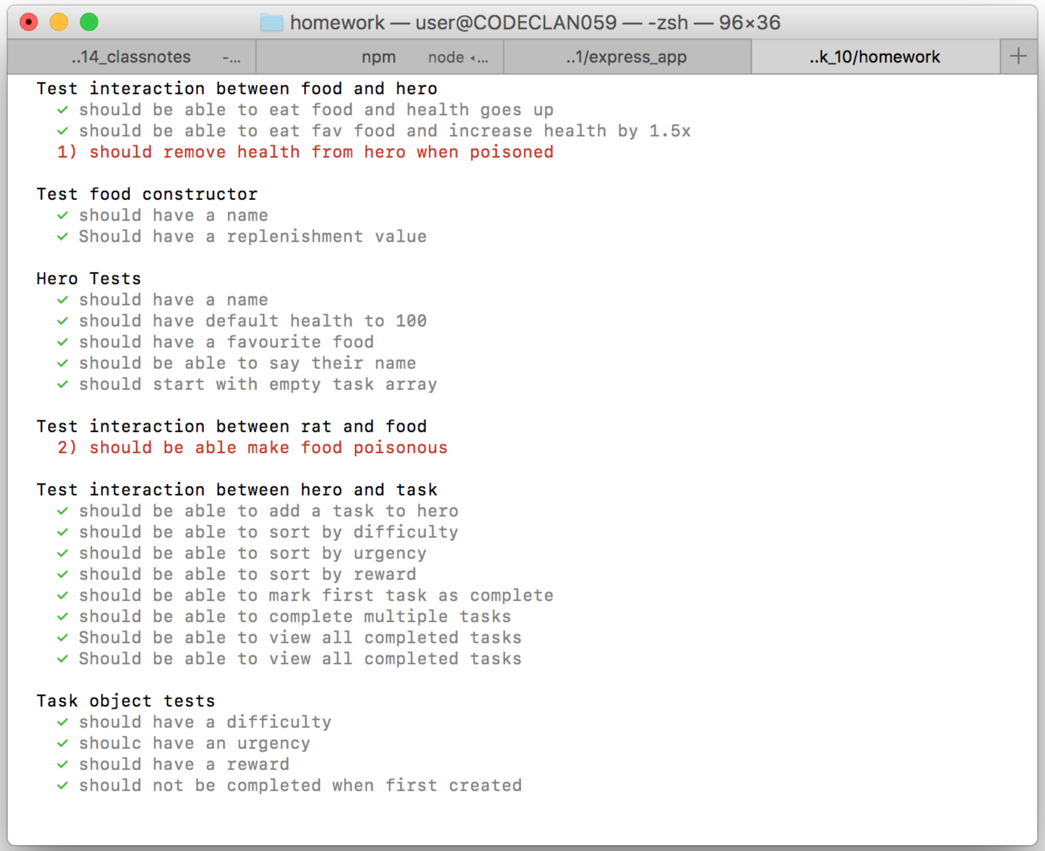
Clients Pets

Animal Name	Type	Breed

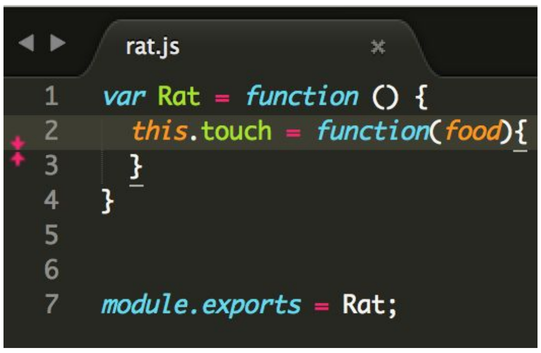
P 18 Testing example



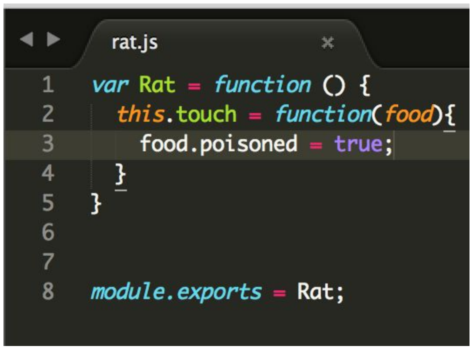
```
var assert = require("assert");
var Food = require("../food.js");
var Rat = require("../rat.js");

describe("Test interaction between rat and food", function(){

  it("should be able make food poisonous", function(){
    rat = new Rat();
    food = new Food("Tuna", 20);
    rat.touch(food);
    assert.strictEqual(food.poisoned, true)
  })
})
```



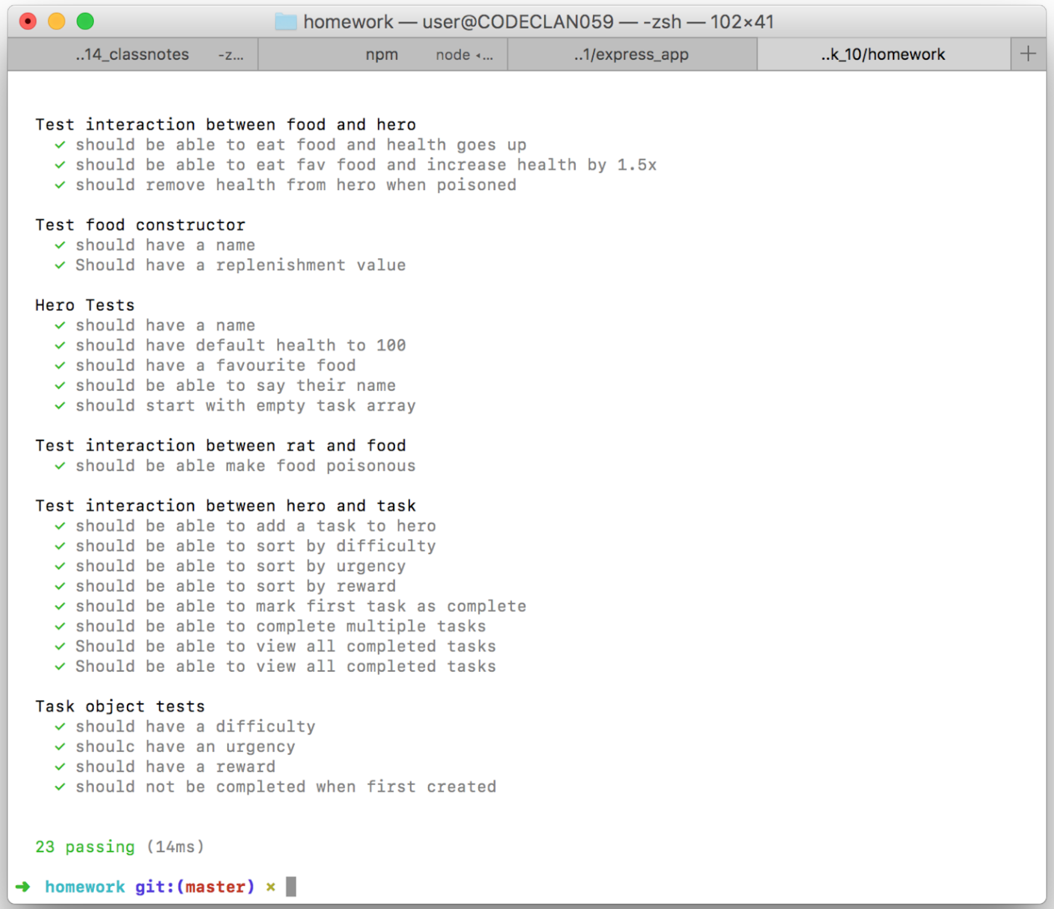
```
homework — user@CODECLAN059 — -zsh — 96×36
..14_classnotes   -...   npm   node •...   ..1/express_app   ..k_10/homework

Test interaction between food and hero
  ✓ should be able to eat food and health goes up
  ✓ should be able to eat fav food and increase health by 1.5x
  1) should remove health from hero when poisoned

Test food constructor
  ✓ should have a name
  ✓ Should have a replenishment value

Hero Tests
  ✓ should have a name
  ✓ should have default health to 100
  ✓ should have a favourite food
  ✓ should be able to say their name
  ✓ should start with empty task array

Test interaction between rat and food
  2) should be able make food poisonous

Test interaction between hero and task
  ✓ should be able to add a task to hero
  ✓ should be able to sort by difficulty
  ✓ should be able to sort by urgency
  ✓ should be able to sort by reward
  ✓ should be able to mark first task as complete
  ✓ should be able to complete multiple tasks
  ✓ Should be able to view all completed tasks
  ✓ Should be able to view all completed tasks

Task object tests
  ✓ should have a difficulty
  ✓ shoulc have an urgency
  ✓ should have a reward
  ✓ should not be completed when first created
```



```
rat.js

var Rat = function () {
  this.touch = function(food){
  }
}


module.exports = Rat;
```



```
rat.js

var Rat = function () {
  this.touch = function(food){
    food.poisoned = true;
  }
}


module.exports = Rat;
```



```
Test interaction between food and hero
  ✓ should be able to eat food and health goes up
  ✓ should be able to eat fav food and increase health by 1.5x
  ✓ should remove health from hero when poisoned

Test food constructor
  ✓ should have a name
  ✓ Should have a replenishment value

Hero Tests
  ✓ should have a name
  ✓ should have default health to 100
  ✓ should have a favourite food
  ✓ should be able to say their name
  ✓ should start with empty task array

Test interaction between rat and food
  ✓ should be able make food poisonous

Test interaction between hero and task
  ✓ should be able to add a task to hero
  ✓ should be able to sort by difficulty
  ✓ should be able to sort by urgency
  ✓ should be able to sort by reward
  ✓ should be able to mark first task as complete
  ✓ should be able to complete multiple tasks
  ✓ Should be able to view all completed tasks
  ✓ Should be able to view all completed tasks

Task object tests
  ✓ should have a difficulty
  ✓ shoulc have an urgency
  ✓ should have a reward
  ✓ should not be completed when first created


23 passing (14ms)

➜  homework git:(master) ✗
```