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: An array of train station name and used add function to add new stop, Edinburgh Waverley to the list/array.

```
1
2
3  stops = [ "Croy", "Cumbernauld", "Falkirk High", "Linlithgow", "Livingston", "Haymarket" ]
4
```

```
2
3  stops = [ "Croy", "Cumbernauld", "Falkirk High", "Linlithgow", "Livingston", "Haymarket" ]
4
5  def add(stops, new_stop)
6    return stops << new_stop
7  end
8  p add(stops, "Edinburgh Waverley")
9
10
11 # stops.push("Edinburgh Waverley")
```

```
Ruby_Tasks- — user@users-MacBook-Pro-6 — ..W/Ruby_Tasks- — -zsh —...
➜  Ruby_Tasks- git:(master) ✗ ruby task_A.rb
["Croy", "Cumbernauld", "Falkirk High", "Linlithgow", "Livingston",
 "Haymarket", "Edinburgh Waverley"]
➜  Ruby_Tasks- git:(master) ✗
```

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: A hash of parts in the UK and with information. The function total_population shows the sum of the population from Scotland, Wales and England in the UK.

```ruby
united_kingdom = [
  {name: "Scotland",population: 5295000,capital: "Edinburgh"},
  {name: "Wales",population: 3063000,capital: "Swansea"},
  {name: "England",population: 53010000,capital: "London"}
]
```

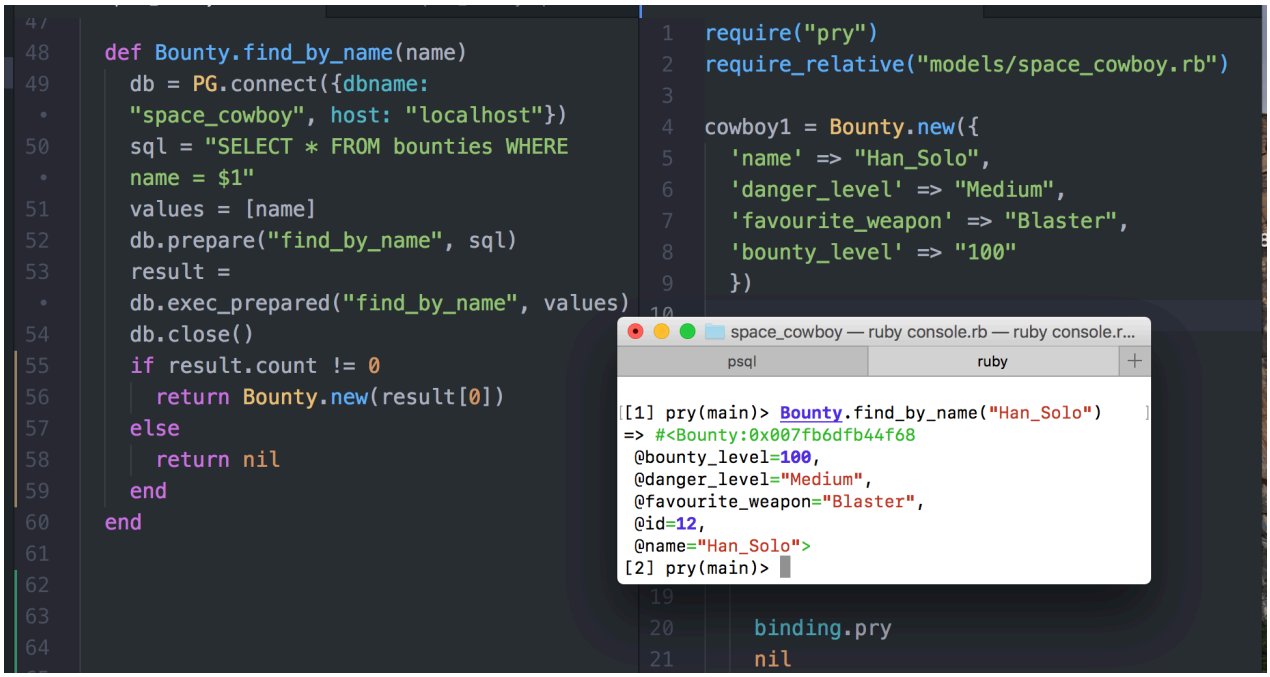
countries.rb

```ruby
def total_population(countries)
  total = 0
  for country in countries
    total += country[:population]
    end
  return total
end

p total_population(united_kingdom)
```

```
Ruby_Tasks- — user@users-MBP-6 — ..W/Ruby_Task...
➜  Ruby_Tasks- git:(master) ✗ ruby countries.rb
61368000
➜  Ruby_Tasks- git:(master) ✗
```

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: Function is called find _by_name and it show up the value of the name key, which is “Han_Solo”.



```
def Bounty.find_by_name(name)
  db = PG.connect({dbname:
  "space_cowboy", host: "localhost"})
  sql = "SELECT * FROM bounties WHERE
  name = $1"
  values = [name]
  db.prepare("find_by_name", sql)
  result =
  db.exec_prepared("find_by_name", values)
  db.close()
  if result.count != 0
    return Bounty.new(result[0])
  else
    return nil
  end
end
```

```
require("pry")
require_relative("models/space_cowboy.rb")

cowboy1 = Bounty.new({
  'name' => "Han_Solo",
  'danger_level' => "Medium",
  'favourite_weapon' => "Blaster",
  'bounty_level' => "100"
  })
```

space_cowboy — ruby console.rb — ruby console.r...

psql | ruby

```
[1] pry(main)> Bounty.find_by_name("Han_Solo")
=> #<Bounty:0x007fb6dfb44f68
 @bounty_level=100,
 @danger_level="Medium",
 @favourite_weapon="Blaster",
 @id=12,
 @name="Han_Solo">
[2] pry(main)>
```

```
    binding.pry
    nil
```

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: Data has screening time and tickets number. The function popular_time shows the screening time sold the most tickets.

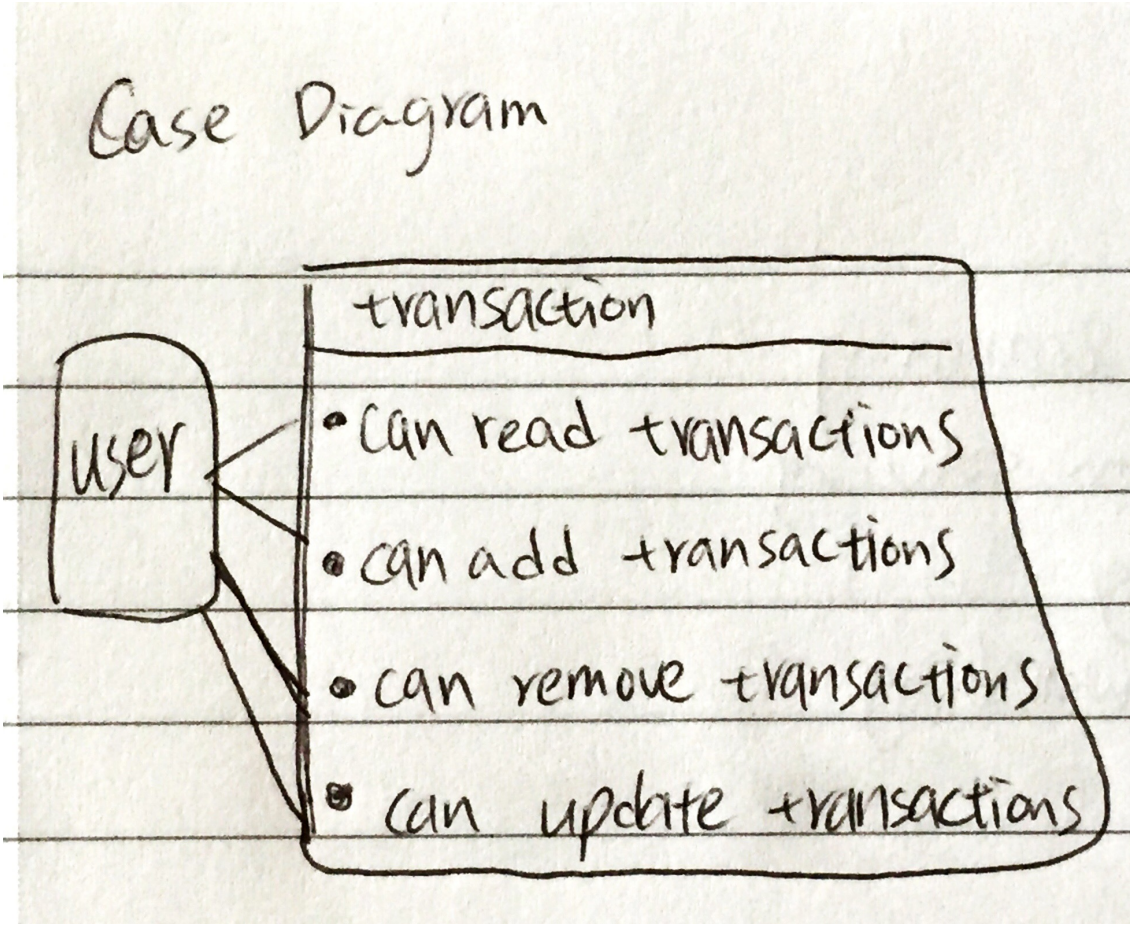
```ruby
Screening.delete_all()

screening1 = Screening.new({ 'screen_time' => '10:00'})
screening1.save()
screening2 = Screening.new({ 'screen_time' => '12:00'})
screening2.save()
screening3 = Screening.new({ 'screen_time' => '14:00'})
screening3.save()

```

```ruby
def popular_time()
  sql = "SELECT screen_time, COUNT(tickets.*) FROM tickets INNER JOIN
  screenings ON tickets.screening_id = screenings.id GROUP BY screen_time
  ORDER BY count DESC limit 1"
  values = []
  tickets_count = SqlRunner.run(sql, values)
  return tickets_count.first["screen_time"]
end
```

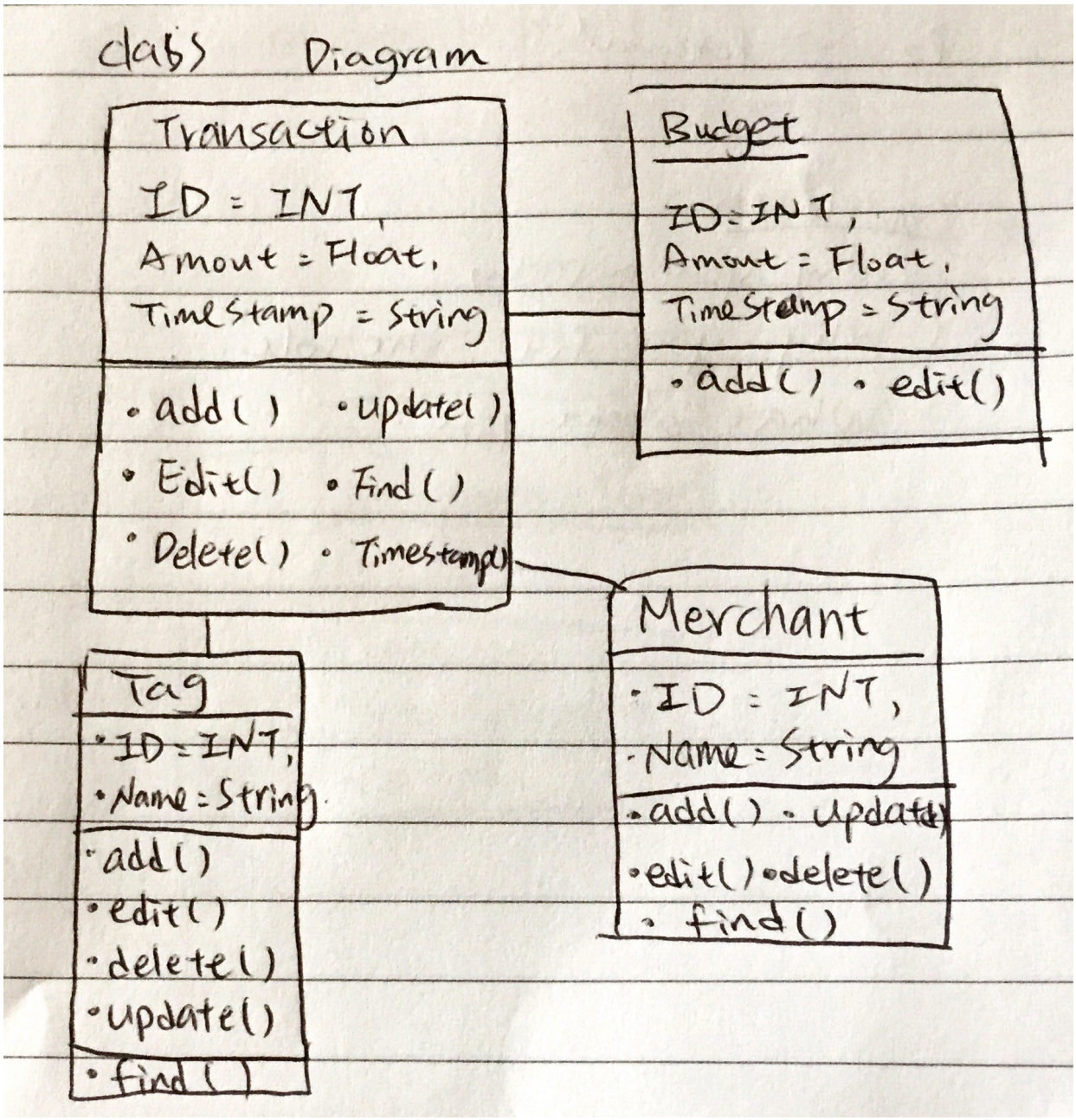
```
cinema_hw — ruby console.rb — ruby — ruby console.rb — 83×5

[1] pry(main)> screening1.popular_time
=> "10:00"
[2] pry(main)>
[3] pry(main)>
```

Week 5 and 6

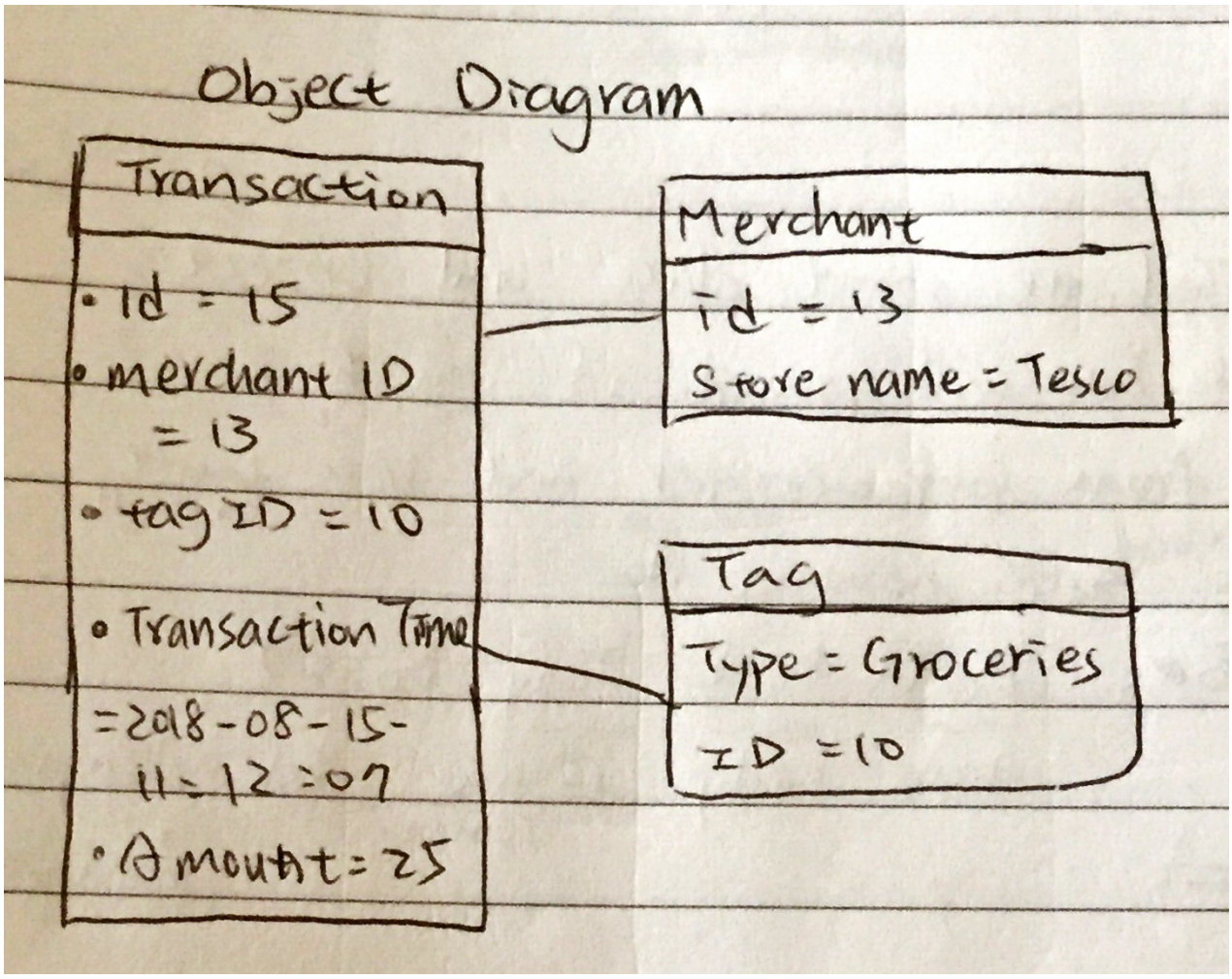
Unit	Ref	Evidence
A&D	A.D.1	A Use Case Diagram
		The functions of the spending tacking system that user can interact with.



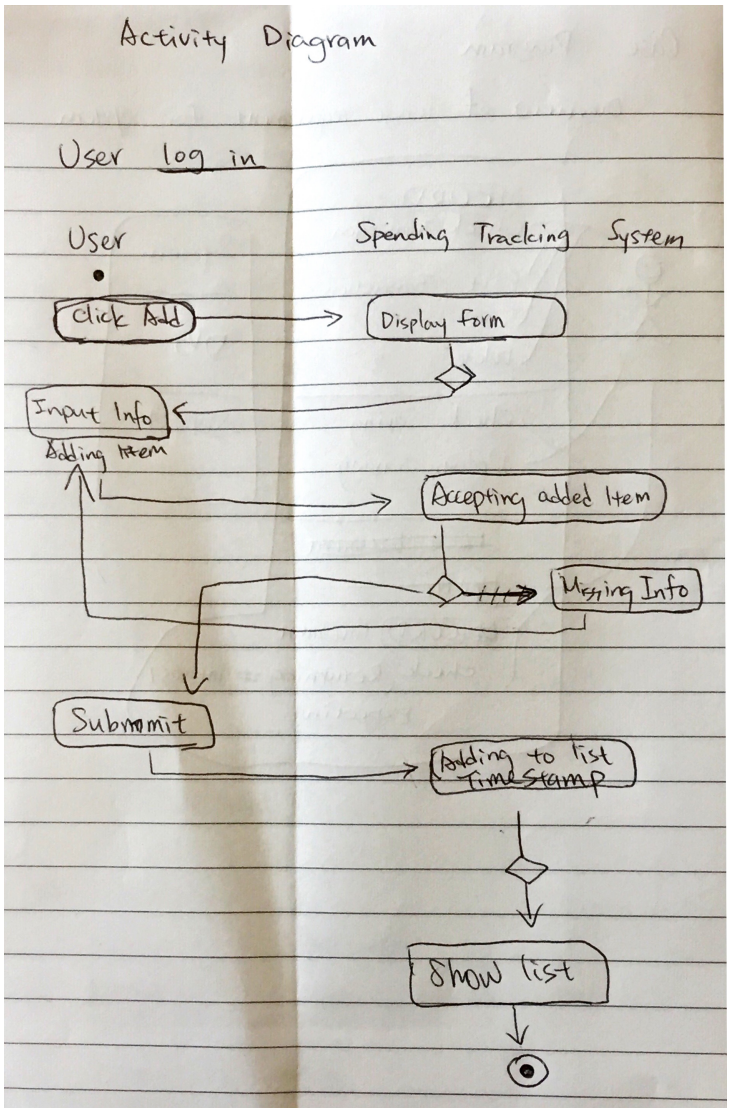
Unit	Ref	Evidence
A&D	A.D.2	A Class Diagram
		The types of items and functions in different classes that used in spending tracking system



Unit	Ref	Evidence
A&D	A.D.3	An Object Diagram
		Description: the relationship between transaction table and merchant/tag table.



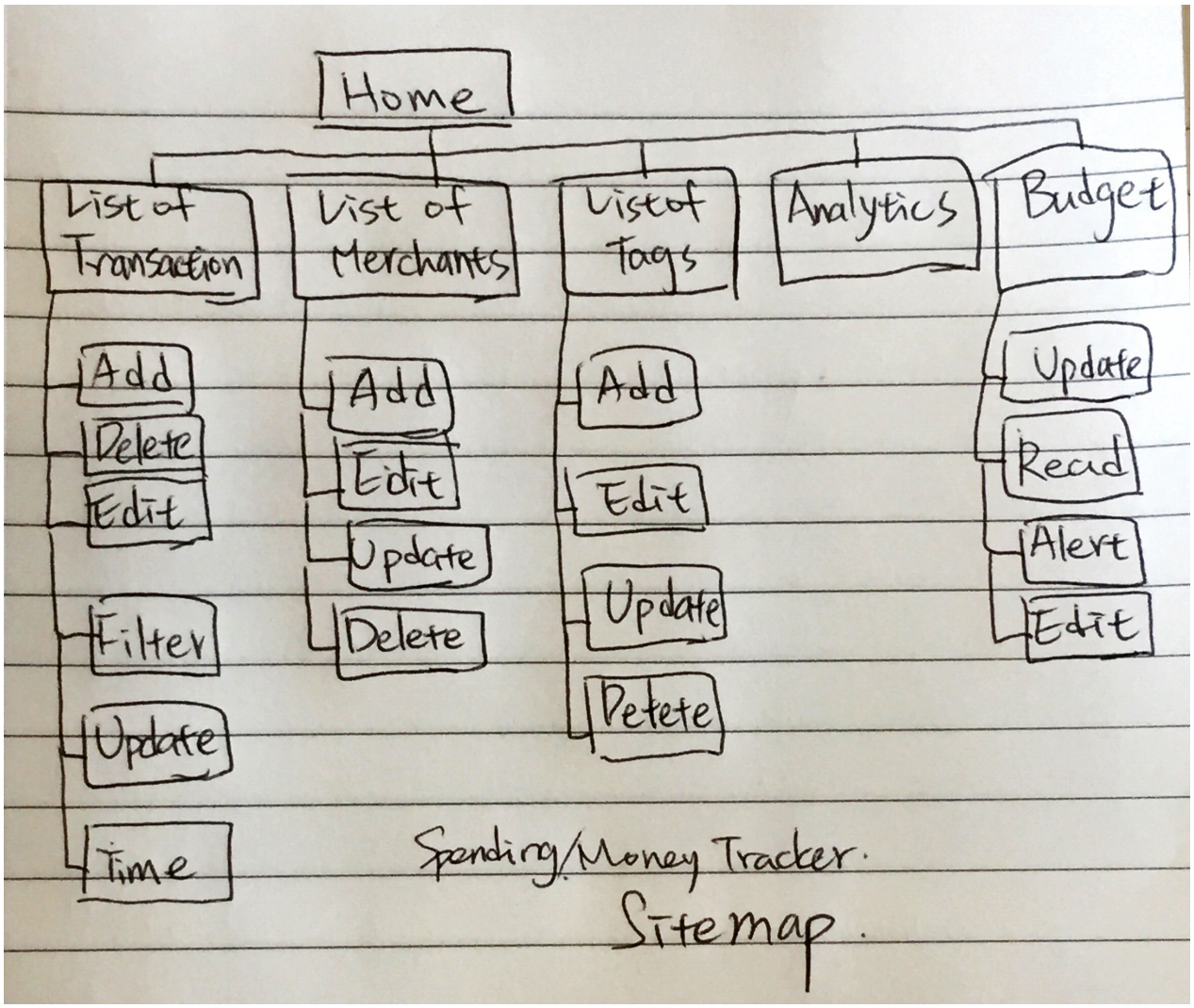
Unit	Ref	Evidence
A&D	A.D.4	An Activity Diagram
		The route of user using add new transaction info to spending tracking system.



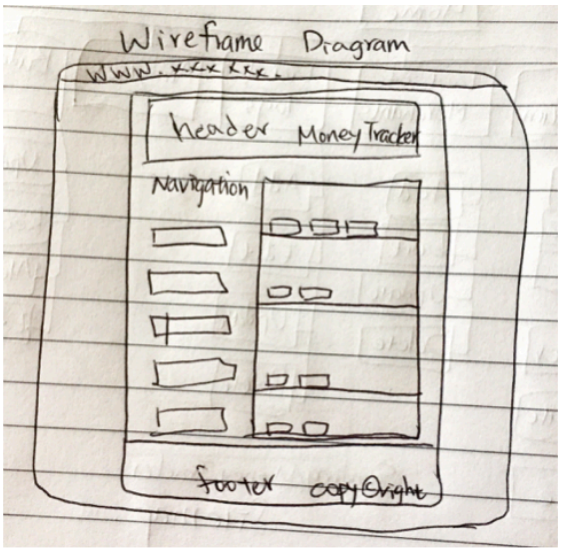
Unit	Ref	Evidence
A&D	A.D.6	Produce an Implementations Constraints plan detailing the following factors: *Hardware and software platforms *Performance requirements *Persistent storage and transactions *Usability *Budgets *Time
		Description: The difficulties and solutions.

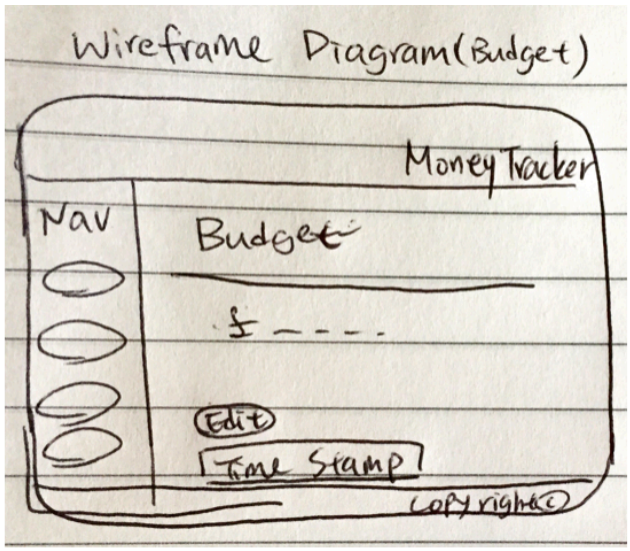
Topic	Possible Effect of Constraint on Product	Solution
Hardware and software platforms	Content may display differently in Mobile screen.	Plan for mobile and set up in CSS.
Performance requirements	Occurring errors.	Regularly checking app performance and ask third person to text the product.
Persistent storage and transactions	Memory storage may not be enough.	Regular clear out system automatically. Extend to bigger memory space.
Usability	For people have color blind.	Different front and size or add extra underlines.
Budgets	Limited budget.	Use more free online resources.
Timelimitations	One week	Follow the plan and well organised teamwork

Unit	Ref	Evidence
P	P.5	User Site Map
		Description: The navigation options in transaction webpage and its' own functions.



Unit	Ref	Evidence
P	P.6	2 Wireframe Diagrams
		Description: The frame work of spending tracker application website.

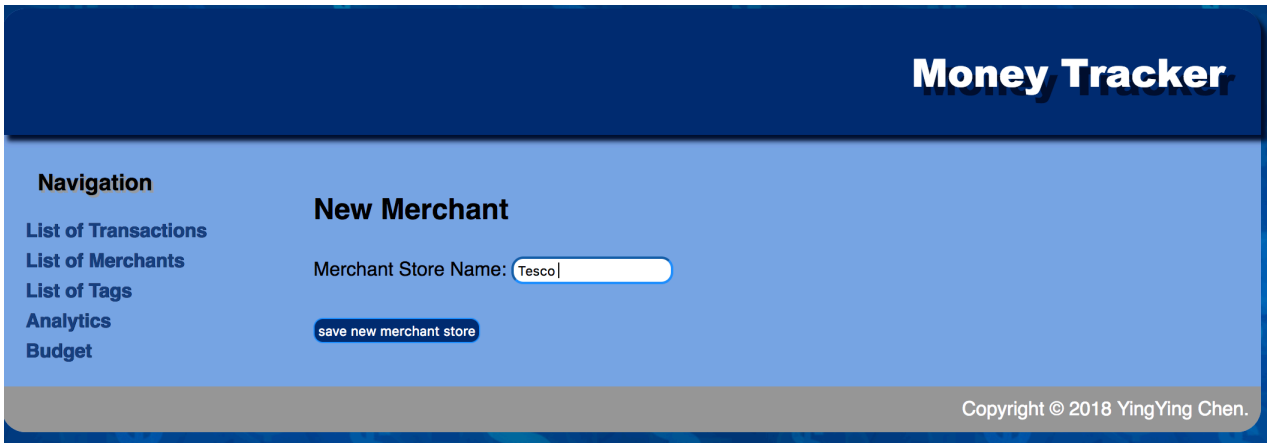


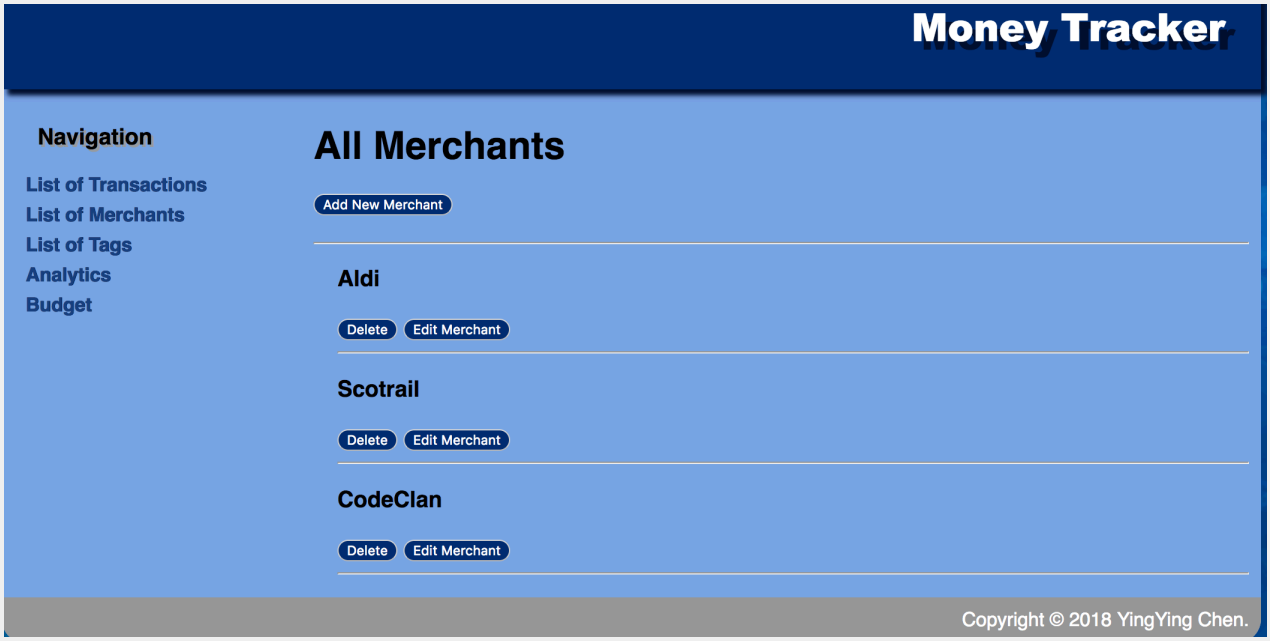


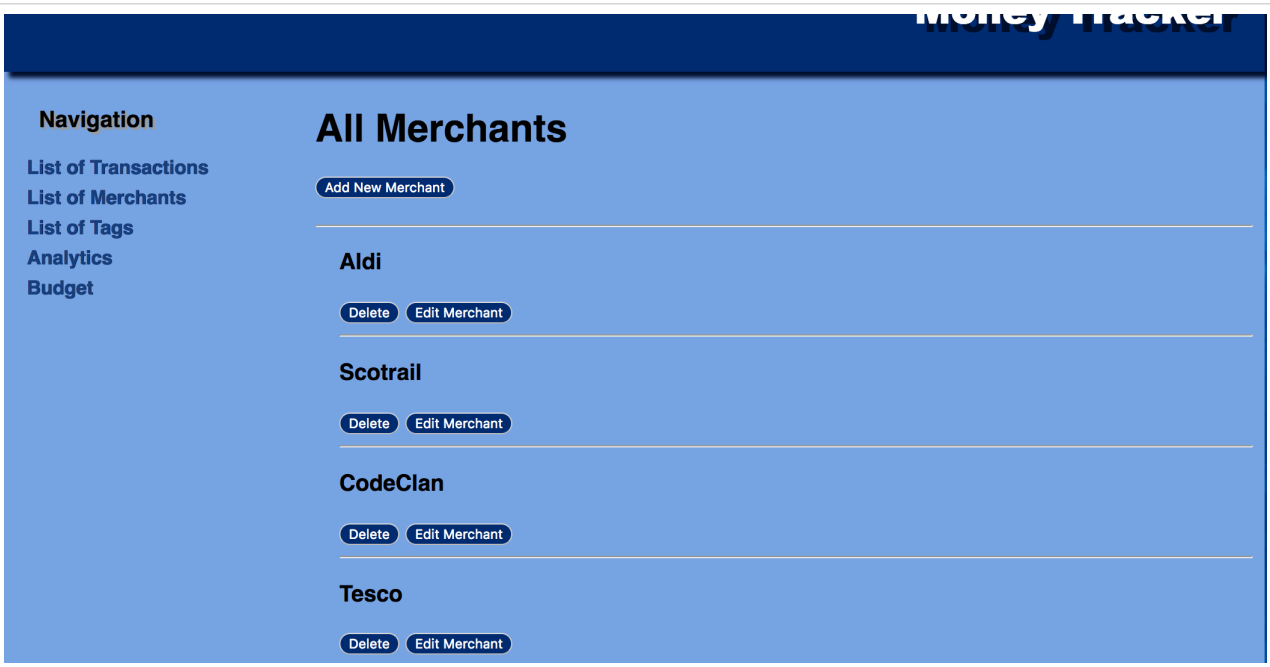
Unit	Ref	Evidence
P	P.10	Example of Pseudocode used for a method
		Description: A plan of a functionality, filter_by_tag_and_merchant( ) in transaction class in spending tracker application.

This function should filter out transactions with specific tag and merchants and display in time desc order, filter_by_tag_and_merchant( ) {
Each filter require one tag type and one merchant store name
A list of tag type
A list of merchant store names
Each transaction should have timestamp
Filter out transactions with tags are not the same as the selected tag
Filter out transactions with merchants are not the same as the selected merchant
Organise selected transactions in Desc time order.
}

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: Add New Merchant in to merchant list.

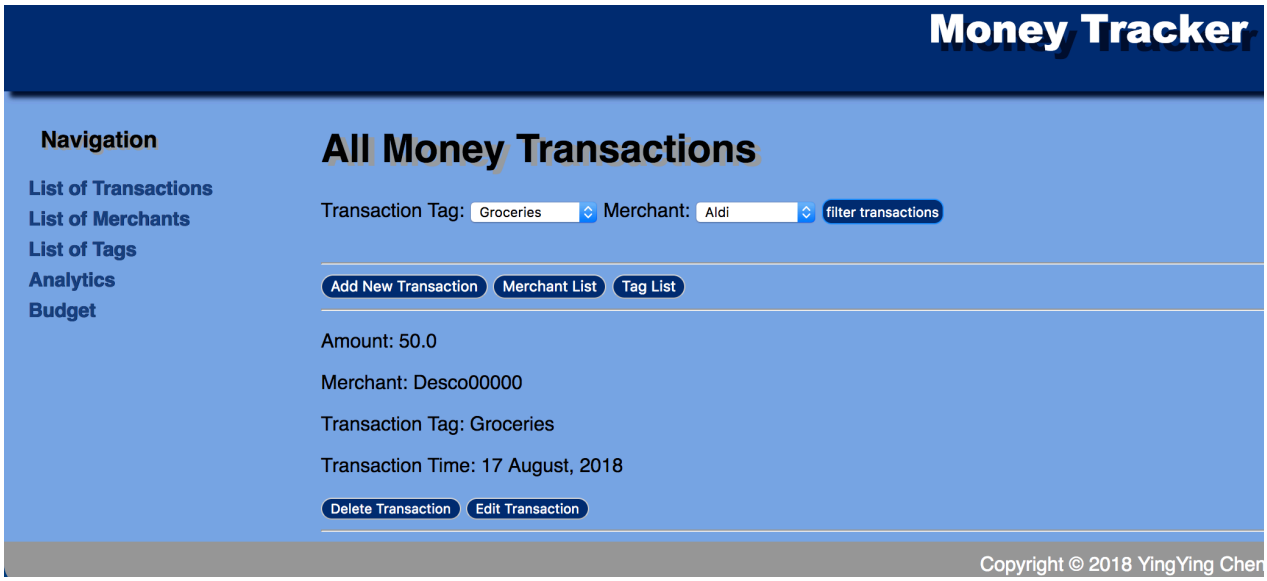




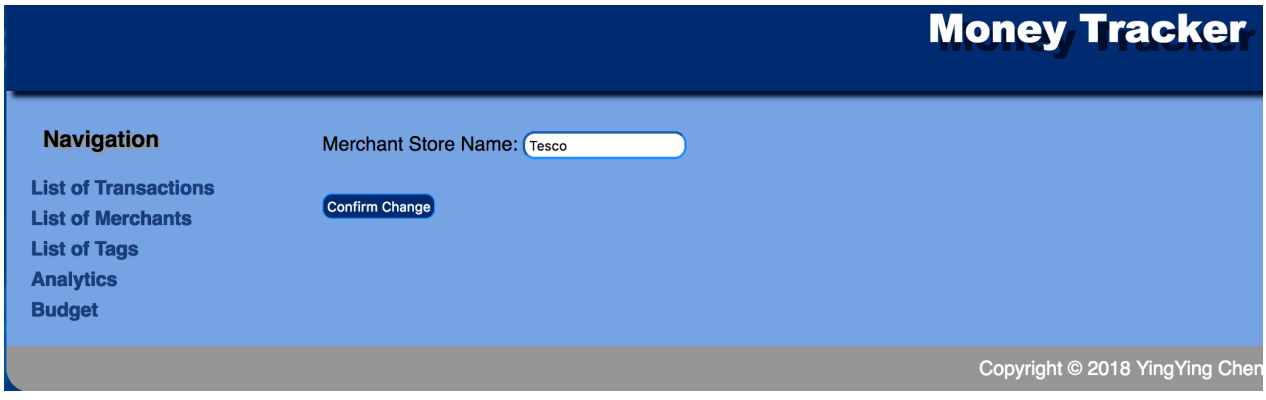


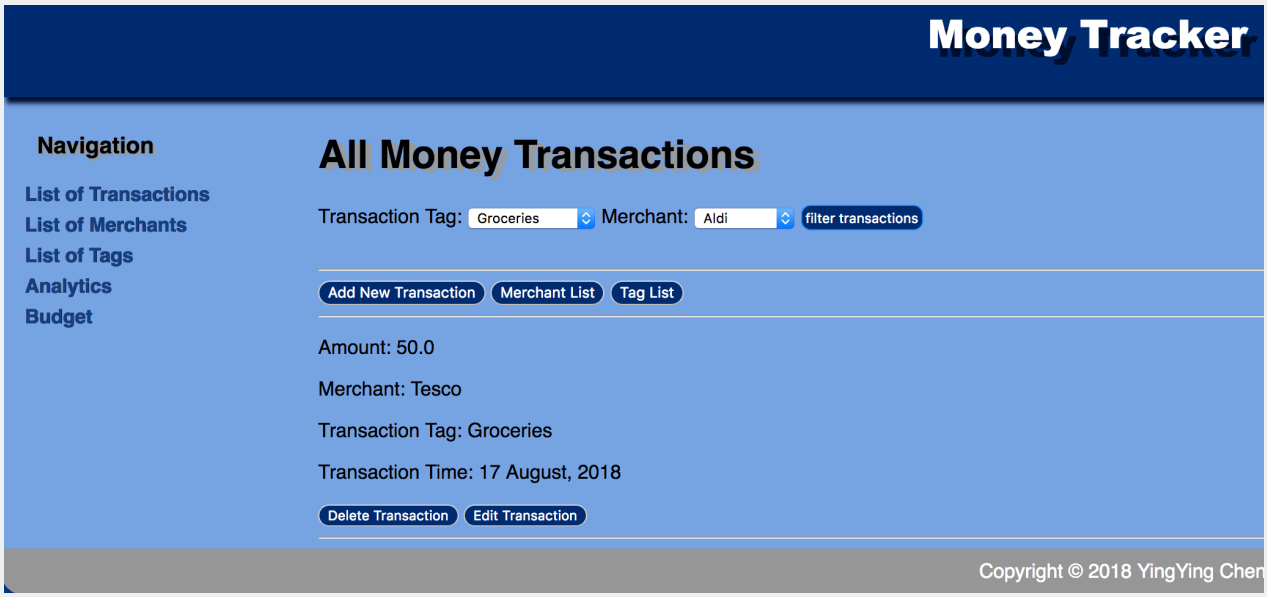

```
psql (10.4)
Type "help" for help.

moneytracker=# select * from merchants;
 id | store_name
----+------------
  3 | Aldi
  4 | Scotrail
  5 | CodeClan
  6 | Tesco
(4 rows)

moneytracker=#
```

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: Update merchant name in transaction.



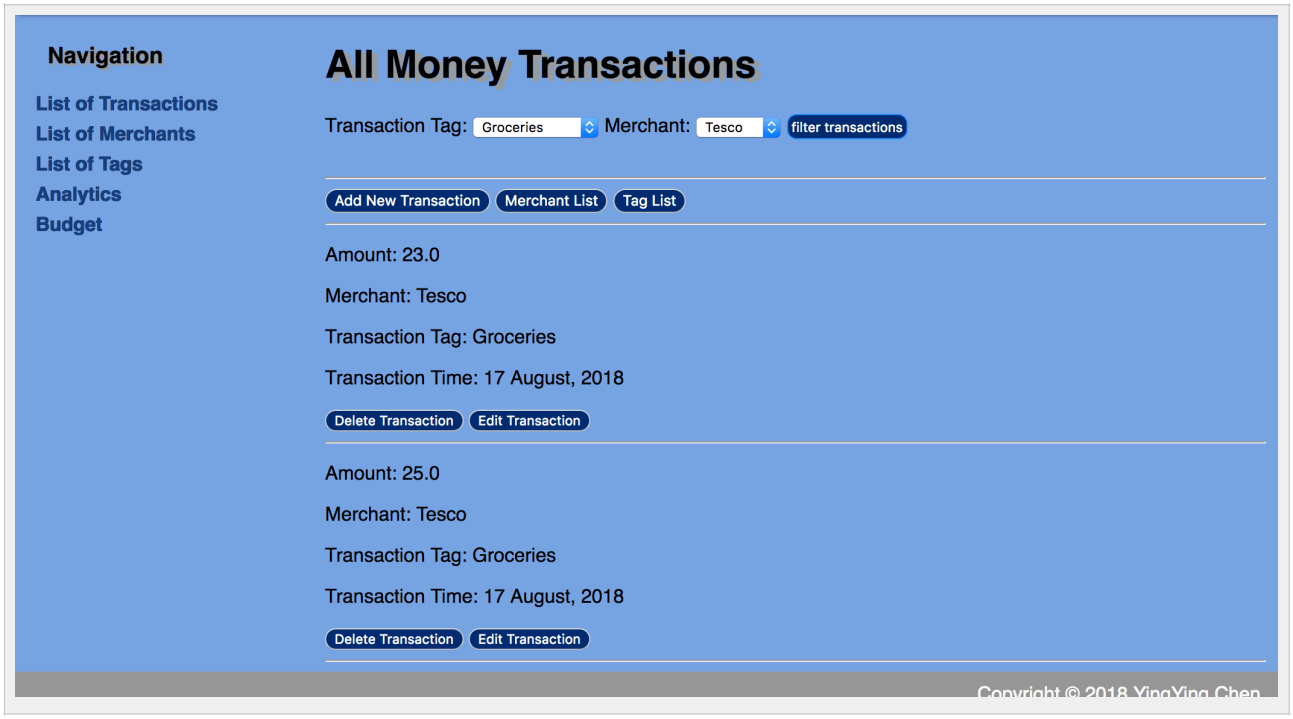

```
moneytracker=# select * from merchants
moneytracker-# ;
 id | store_name
----+------------
  3 | Aldi
  4 | Scotrail
  8 | Desco00000
(3 rows)

moneytracker=#
```



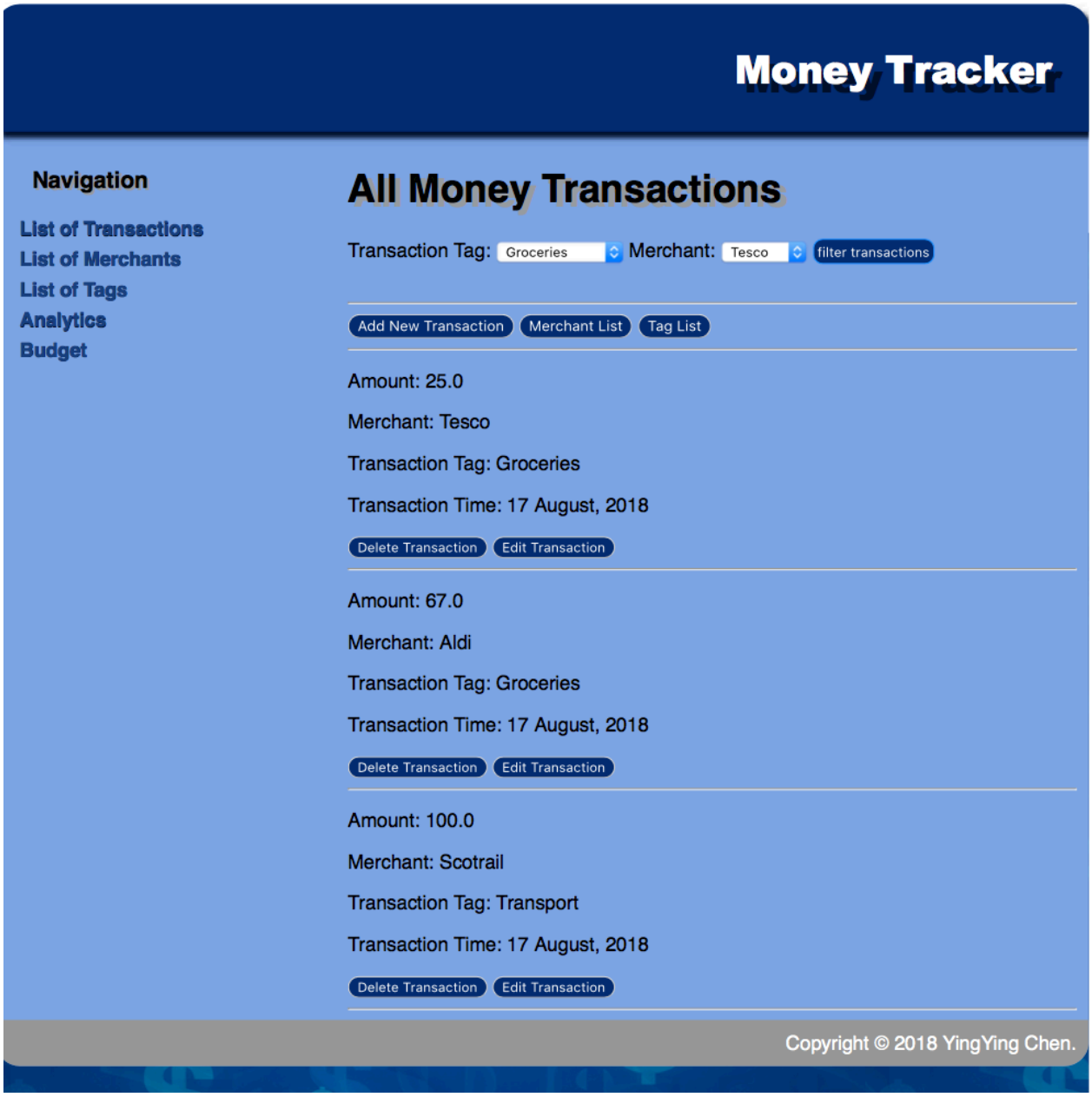



```
moneytracker=# select * from merchants;
 id | store_name
----+------------
  3 | Aldi
  4 | Scotrail
  8 | Tesco
(3 rows)

moneytracker#
```

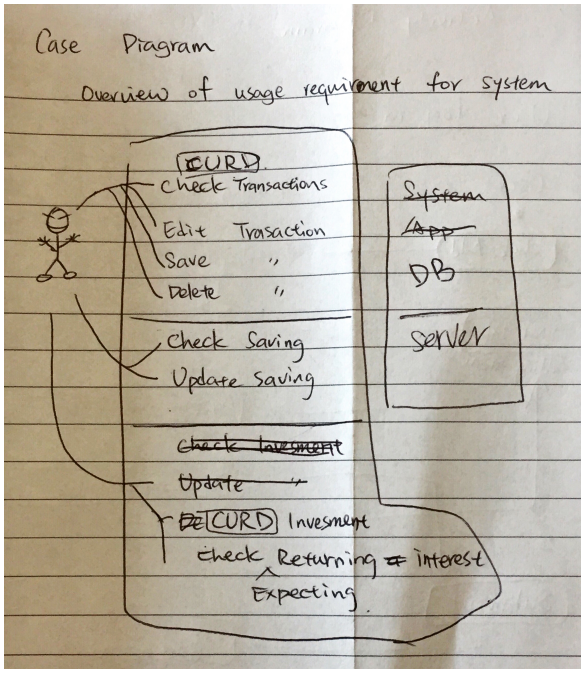
Unit	Ref	Evidence
P	P.15	Show the correct output of results and feedback to user. Take a screenshot of: * The user requesting information or an action to be performed * The user request being processed correctly and demonstrated in the program
		Description: Filter out transactions with tag, Groceries and merchant, Tesco.

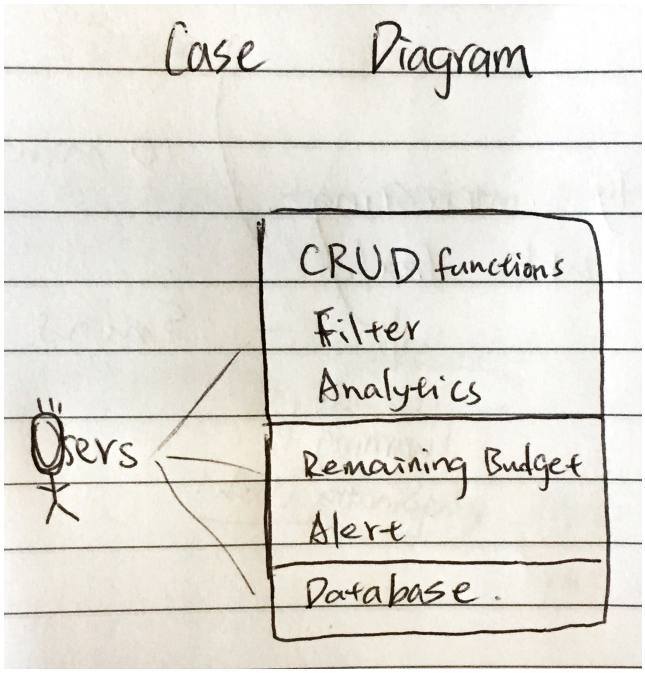

Navigation
List of Transactions
List of Merchants
List of Tags
Analytics
Budget
All Money Transactions
Transaction Tag: Groceries
Merchant: Tesco
filter transactions
Add New Transaction
Merchant List
Tag List
Amount: 23.0
Merchant: Tesco
Transaction Tag: Groceries
Transaction Time: 17 August, 2018
Delete Transaction
Edit Transaction
Amount: 25.0
Merchant: Tesco
Transaction Tag: Groceries
Transaction Time: 17 August, 2018
Delete Transaction
Edit Transaction
Copyright © 2018 YingYing Chen

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/YYChen01988/Transaction_Tracker_Project



Unit	Ref	Evidence
P	P.12	Take screenshots or photos of your planning and the different stages of development to show changes.
		Description: The picture on the left is at the first stage of Spending tracker project and the right picture is a new case diagram in the end of project.

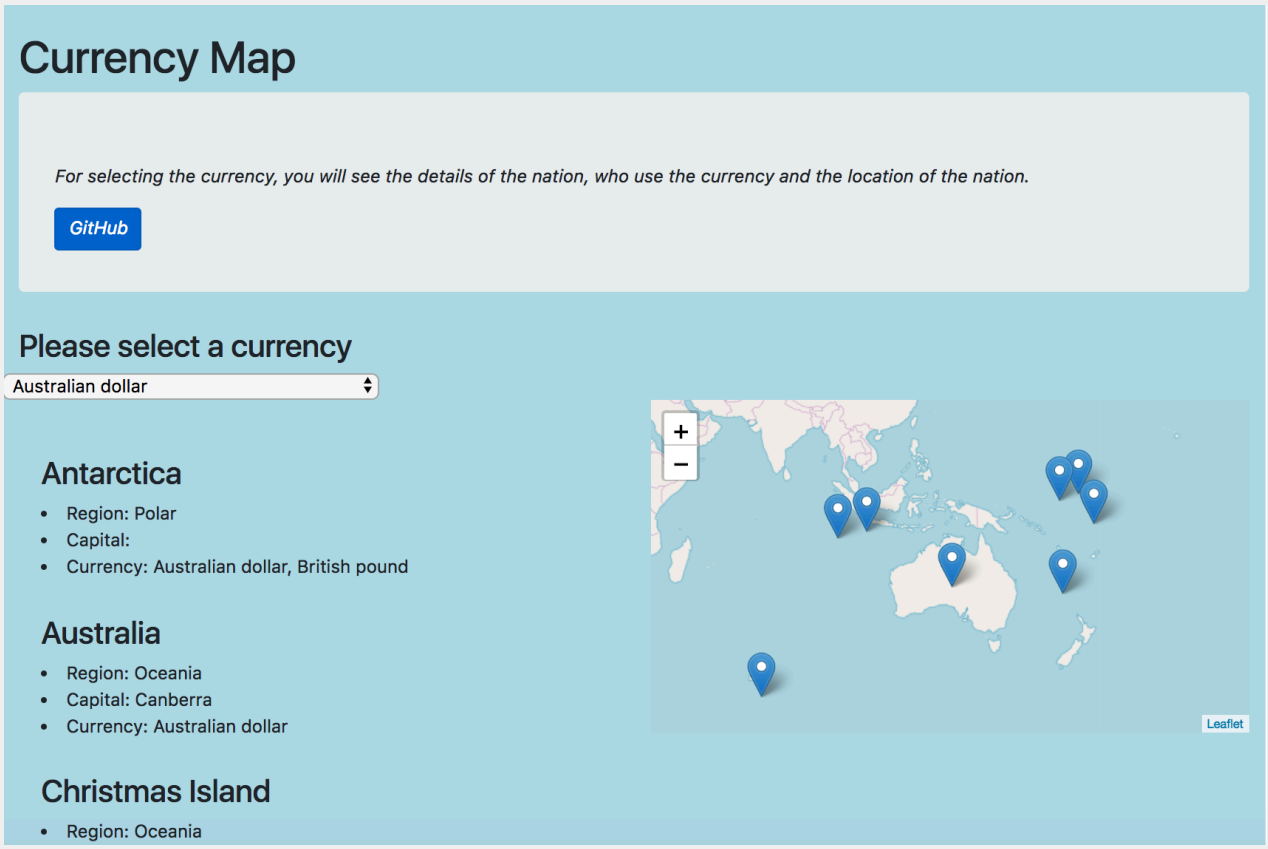




Week 7

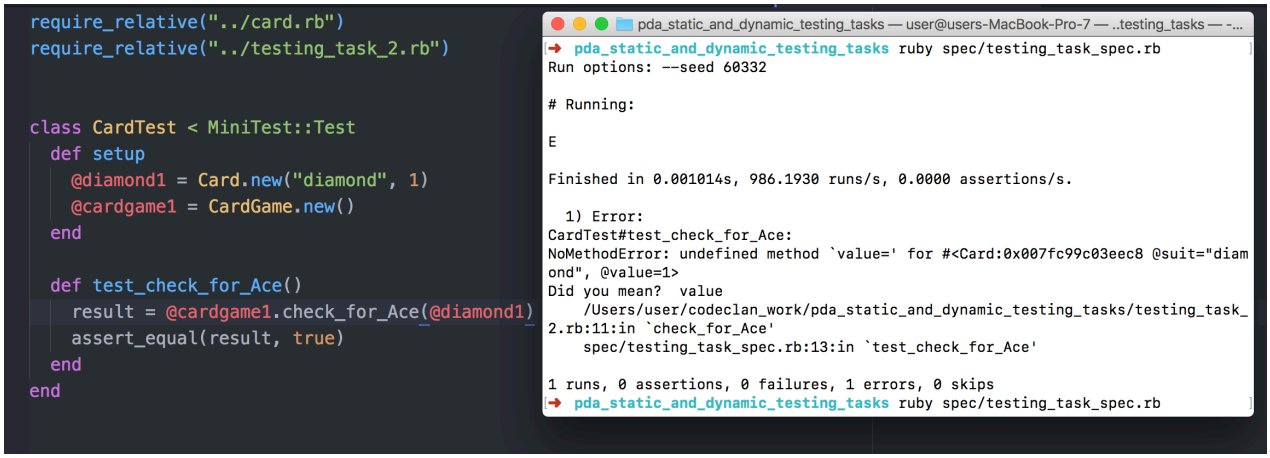
Unit	Ref	Evidence
P	P.16	Show an API being used within your program. Take a screenshot of: * The code that uses or implements the API * The API being used by the program whilst running
		Description: The link of the API in the example application (https://restcountries.eu/rest/v2/all)

```
const PubSub = require('../helpers/pub_sub.js');

const SelectView = function(select){
  this.select = select;
}

SelectView.prototype.bindEvents = function(){
  PubSub.subscribe("CurrencyList:names-ready", (event) => {
    this.poplulateList(event.detail);
  });

  this.select.addEventListener('change', (event) => {
    PubSub.publish("SelectView:currency_name-selected", event.target.value);
  });
};

SelectView.prototype.poplulateList = function(names){
  names.forEach((name, index) => {
    const option = document.createElement('option');
    option.textContent = name;
    option.value = index;
    this.select.appendChild(option);
  });
};

module.exports = SelectView;
```



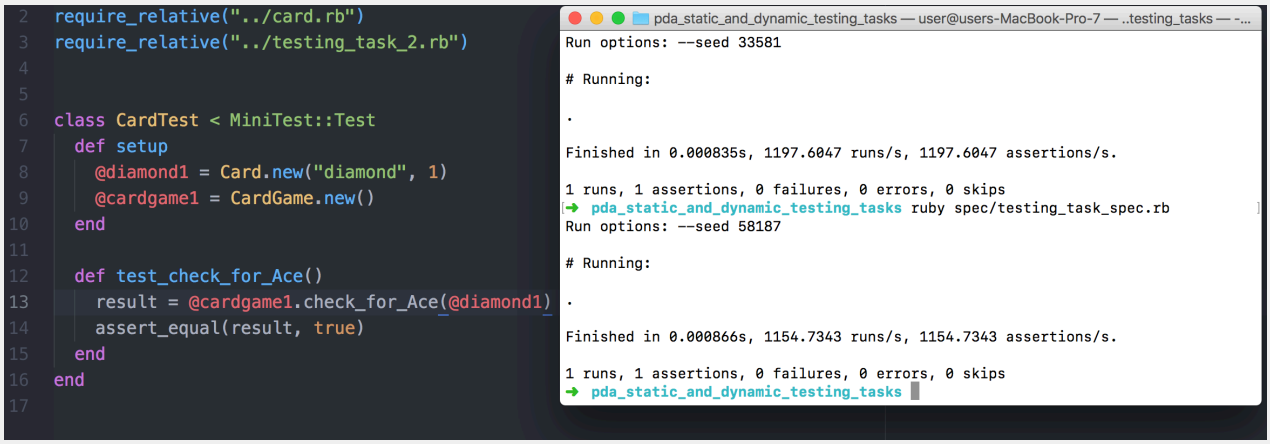

```
const PubSub = require('../helpers/pub_sub');

const MapWrapper = function(container){
  this.container = container;
  this.coords = [55.8654192, -4.258020999999999];
  this.map = L.map(this.container);
  this.osmLayer = new L.TileLayer("http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png");
  this.map.setView(this.coords, 10).addLayer(this.osmLayer);
  this.markers = []
}
MapWrapper.prototype.bindEvents = function(){
  PubSub.subscribe("CurrencyList:country-ready", (event) => {
    // map.removeLayer(marker);
    this.markers.forEach((m)=>{
      this.map.removeLayer(m);
    });
```

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: Running unit test with ruby code.



```ruby
require_relative("../card.rb")
require_relative("../testing_task_2.rb")


class CardTest < MiniTest::Test
  def setup
    @diamond1 = Card.new("diamond", 1)
    @cardgame1 = CardGame.new()
  end

  def test_check_for_Ace()
    result = @cardgame1.check_for_Ace(@diamond1)
    assert_equal(result, true)
  end
end
```

```
pda_static_and_dynamic_testing_tasks — user@users-MacBook-Pro-7 — ..testing_tasks — -...
➜  pda_static_and_dynamic_testing_tasks ruby spec/testing_task_spec.rb
Run options: --seed 60332

# Running:

E

Finished in 0.001014s, 986.1930 runs/s, 0.0000 assertions/s.

  1) Error:
CardTest#test_check_for_Ace:
NoMethodError: undefined method `value=' for #<Card:0x007fc99c03eec8 @suit="diam
ond", @value=1>
Did you mean?  value
    /Users/user/codeclan_work/pda_static_and_dynamic_testing_tasks/testing_task_
2.rb:11:in `check_for_Ace'
    spec/testing_task_spec.rb:13:in `test_check_for_Ace'

1 runs, 0 assertions, 0 failures, 1 errors, 0 skips
➜  pda_static_and_dynamic_testing_tasks ruby spec/testing_task_spec.rb
```



```ruby
require_relative("../card.rb")
require_relative("../testing_task_2.rb")


class CardTest < MiniTest::Test
  def setup
    @diamond1 = Card.new("diamond", 1)
    @cardgame1 = CardGame.new()
  end

  def test_check_for_Ace()
    result = @cardgame1.check_for_Ace(@diamond1)
    assert_equal(result, true)
  end
end

```

```
pda_static_and_dynamic_testing_tasks — user@users-MacBook-Pro-7 — ..testing_tasks — -...
Run options: --seed 33581

# Running:

.

Finished in 0.000835s, 1197.6047 runs/s, 1197.6047 assertions/s.

1 runs, 1 assertions, 0 failures, 0 errors, 0 skips
➜  pda_static_and_dynamic_testing_tasks ruby spec/testing_task_spec.rb
Run options: --seed 58187

# Running:

.

Finished in 0.000866s, 1154.7343 runs/s, 1154.7343 assertions/s.

1 runs, 1 assertions, 0 failures, 0 errors, 0 skips
➜  pda_static_and_dynamic_testing_tasks
```

```ruby
def self.cards_total(cards)
  total=0
  for card in cards
    total += card.value
    return "You have a total of" + total
  end
end

end
```

```
pda_static_and_dynamic_testing_tasks — user@users-...
ting_task_spec.rb
Run options: --seed 15697

# Running:

E..

Finished in 0.000988s, 3036.4370 runs/s, 2024.2913 as
sertions/s.

  1) Error:
CardTest#test_cards_total:
NoMethodError: undefined method `cards_total' for #<C
ardGame:0x007fe5938276e0>
    spec/testing_task_spec.rb:25:in `test_cards_total
'

3 runs, 2 assertions, 0 failures, 1 errors, 0 skips
➜  pda_static_and_dynamic_testing_tasks
```

testing_ta... | customer.rb | testing_ta... | card.rb

```ruby
  def check_for_Ace(card)
    if card.value == 1
      return true
    else
      return false
    end
  end
  def highest_card(card1,
  card2)
  if card1.value > card2.value
    return card1.suit
  else
    card2.suit
  end
end
def cards_total(cards)
  total=0
  for card in cards do
    total += card.value
  end
  return "You have a total of
  " + total.to_s
end
end
```

testing_task_2.rb 29:4

customer_spec.rb | testing_task_spec.rb

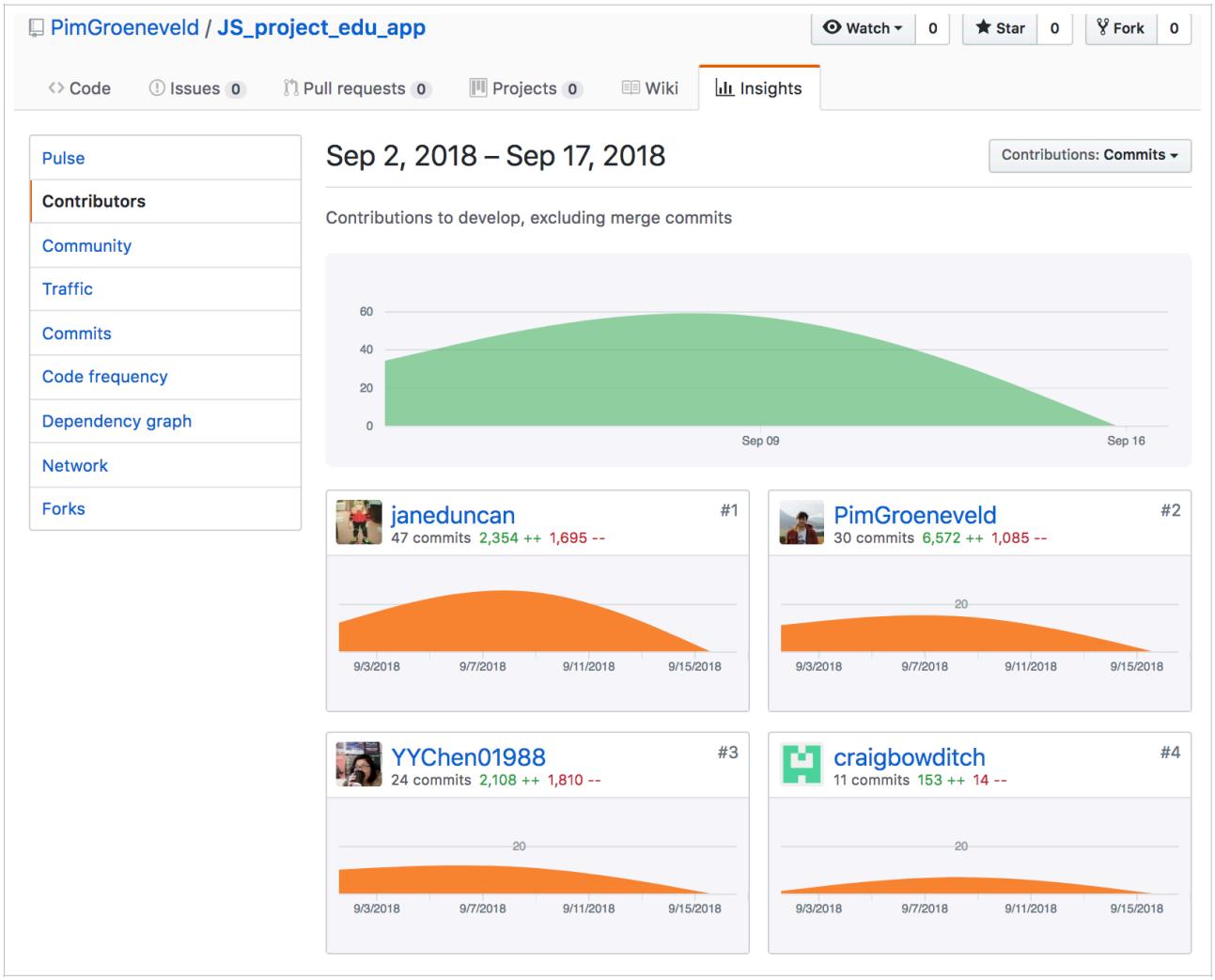
```ruby
class CardTest < MiniTest::Test
  def setup
    @diamond1 = Card.new("diamond", 1)
    @club2 = Card.new("club", 2)
    @cardgame1 = CardGame.new()
    @cards= [@diamond1, @club2]
  end

  def test_check_for_Ace()
    result =
    @cardgame1.check_for_Ace(@diamond1)
    assert_equal(true, result)
  end

  def test_highest_card()
    result =
    @cardgame1.highest_card(@diamond1,
    @club2)
    assert_equal("club", result)
  end

  def test_cards_total()
    result = @cardgame1.cards_total(@cards)
    assert_equal("You have a total of 3",
    result)
```

LF UTF-8 Ruby 0 fil

```
...

Finished in 0.000952s, 3151.2596 runs/s, 3151.2596 ass
ertions/s.

3 runs, 3 assertions, 0 failures, 0 errors, 0 skips
➜  pda_static_and_dynamic_testing_tasks ruby spec/test
ing_task_spec.rb
Run options: --seed 49566

# Running:

E..

Finished in 0.001076s, 2788.1034 runs/s, 1858.7356 ass
ertions/s.

  1) Error:
CardTest#test_cards_total:
NoMethodError: undefined method `cards_total' for #<Ca
rdGame:0x007fd7ad834bb8>
    spec/testing_task_spec.rb:25:in `test_cards_total'

3 runs, 2 assertions, 0 failures, 1 errors, 0 skips
➜  pda_static_and_dynamic_testing_tasks ruby spec/test
ing_task_spec.rb
Run options: --seed 25510

# Running:

...

Finished in 0.000970s, 3092.7822 runs/s, 3092.7822 ass
ertions/s.

3 runs, 3 assertions, 0 failures, 0 errors, 0 skips
➜  pda_static_and_dynamic_testing_tasks
```

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: https://github.com/PimGroeneveld/JS_project_edu_app/graphs/contributors



Unit	Ref	Evidence
P	P.2	Take a screenshot of the project brief from your group project.
		Description: This is the details of our project MVP and additional extension parts.

README.md

Educational App

The BBC are looking to improve their online offering of educational content by developing some interactive browser applications that display information in a fun and interesting way. Your task is to make an a Minimum Viable Product or prototype 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.

MVP

Education Language App

A user should be able to:

- Select a language from a dropdown menu
- A flashcard then displays a phrase in the chosen language
- Click a button to reveal the phrase in English
- Click a button to listen to audio of phrase

Extensions

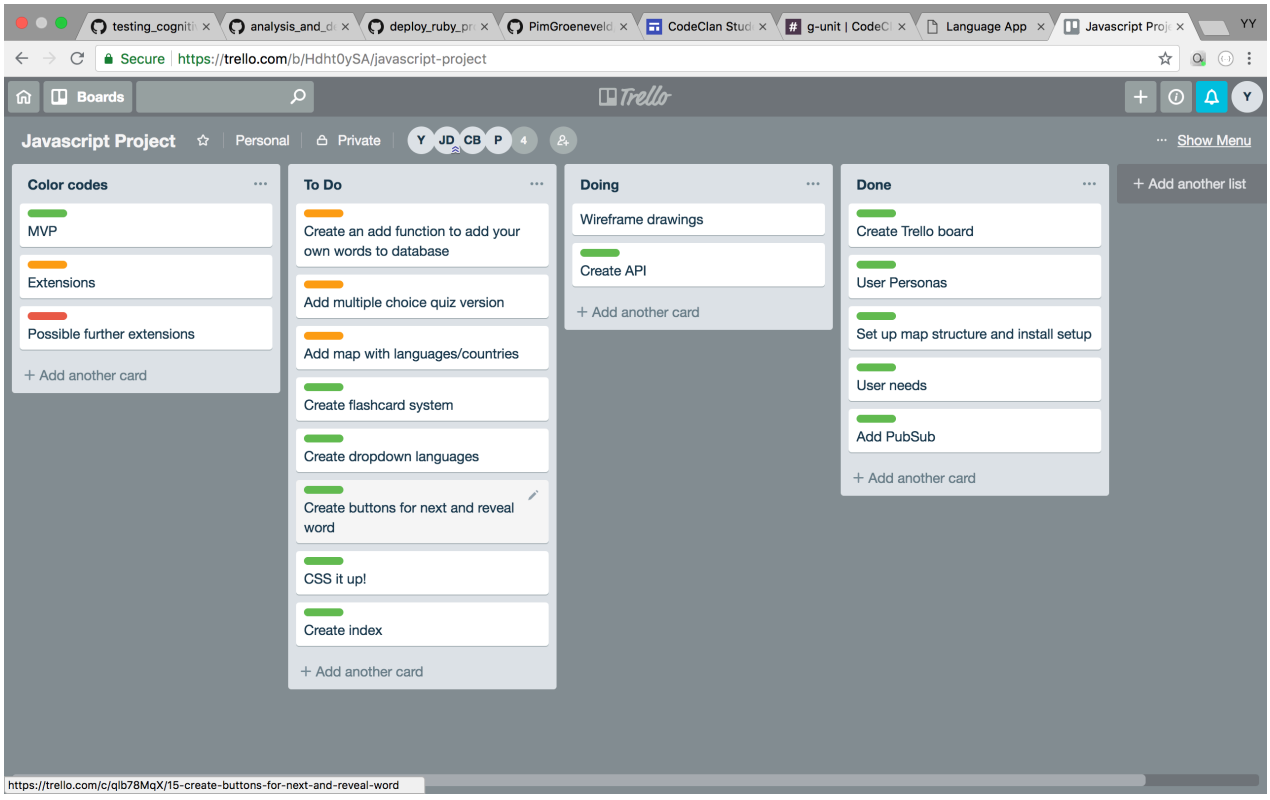
- Word guessing game
- Facts section (using an API to display map)
- Machine learning API
- Add phonetic spelling of phrase to flashcard

API, Libraries, Resources

- https://cloud.google.com/maps-platform/ An open-source library for rendering maps and map functionality.
- https://developer.oxforddictionaries.com An open-source dictionary API (Pim has key)

(Put the API key file in .gitignore)

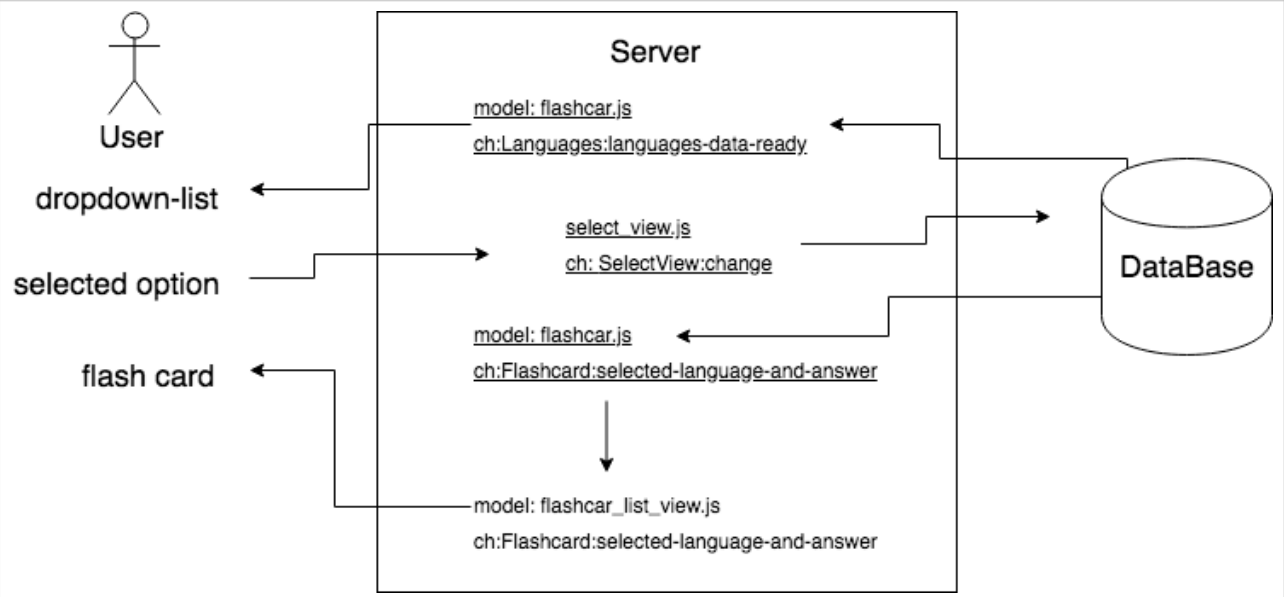
Unit	Ref	Evidence
P	P.3	Provide a screenshot of the planning you completed during your group project, e.g. Trello MOSCOW board.
		Description: Trello shows the features and functions we want to finished in a time scale. https://trello.com/b/Hdht0ySA/javascript-project

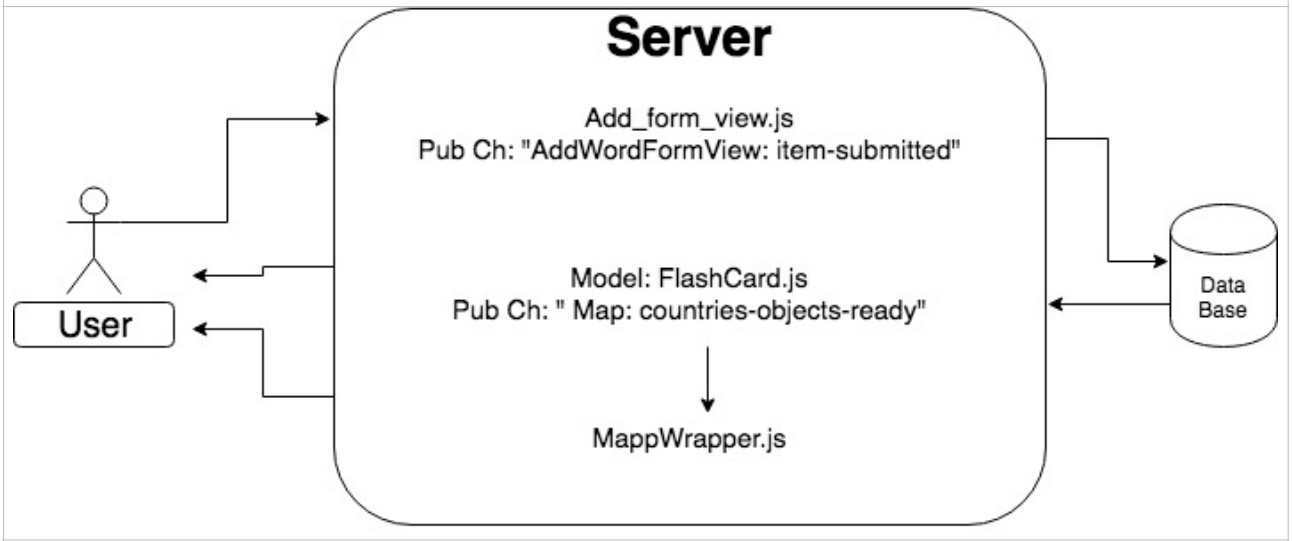


Unit	Ref	Evidence
P	P.4	Write an acceptance criteria and test plan.
		The acceptance criteria for language learning application for group project.

Acceptance Criteria	Expected Result/output	Pass/Fail
Simple user interface	Clean flashcard view	Passed
Practical Content	Add audio and extra information about the foreign languages	Passed
Fun in Learning	Add Quiz part	Passed

Unit	Ref	Evidence
P	P.7	Produce two system interaction diagrams (sequence and/or collaboration diagrams).
		Description: user select change and pass it to database and cause the response to other functions.

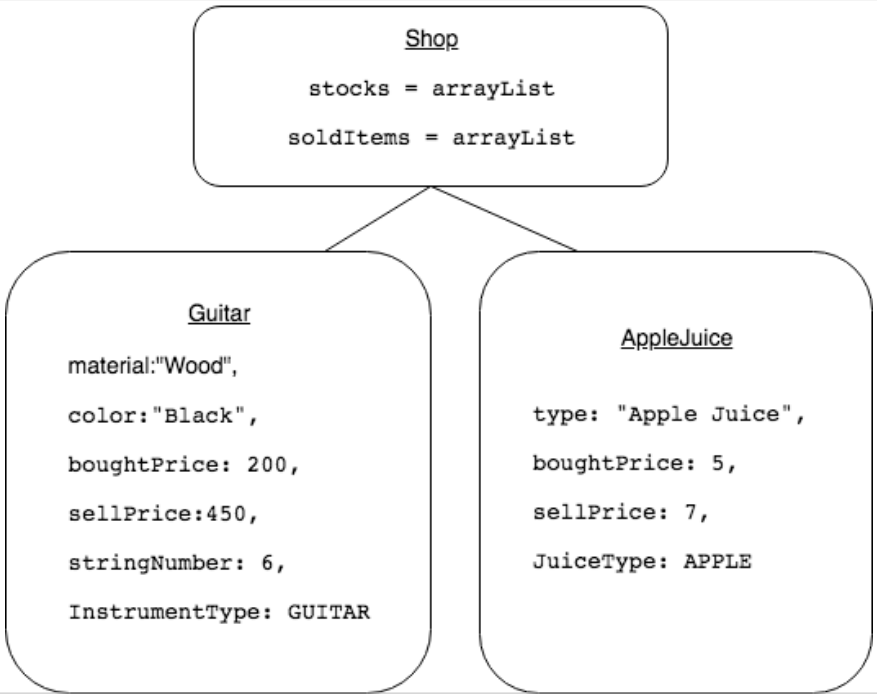




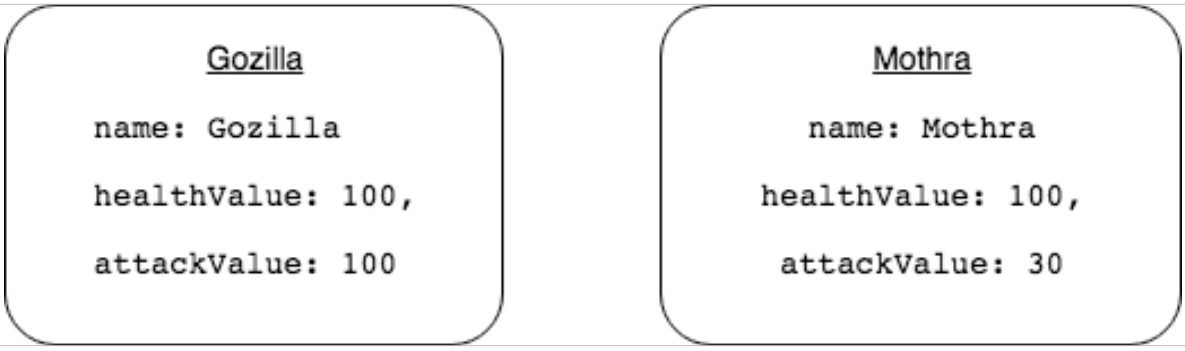
Paste Screenshot here

Unit	Ref	Evidence
P	P.8	Produce two object diagrams.
		Description: **Fist Object Diagram shows a shop sells items, which includes Guitar and Apple Juice.** **Second Object Diagram shows two Kaiju objects but they are not related.**

Object Diagram1. MusicShop Project
(Shop sells items, which includes Guitar and Apple Juice)



Object Diagram2. Kaiju Project
(Gozilla and Mothra inherit from the abstract class Kaiju. However, Gozilla and Mothra are not related.)



Paste Screenshot here

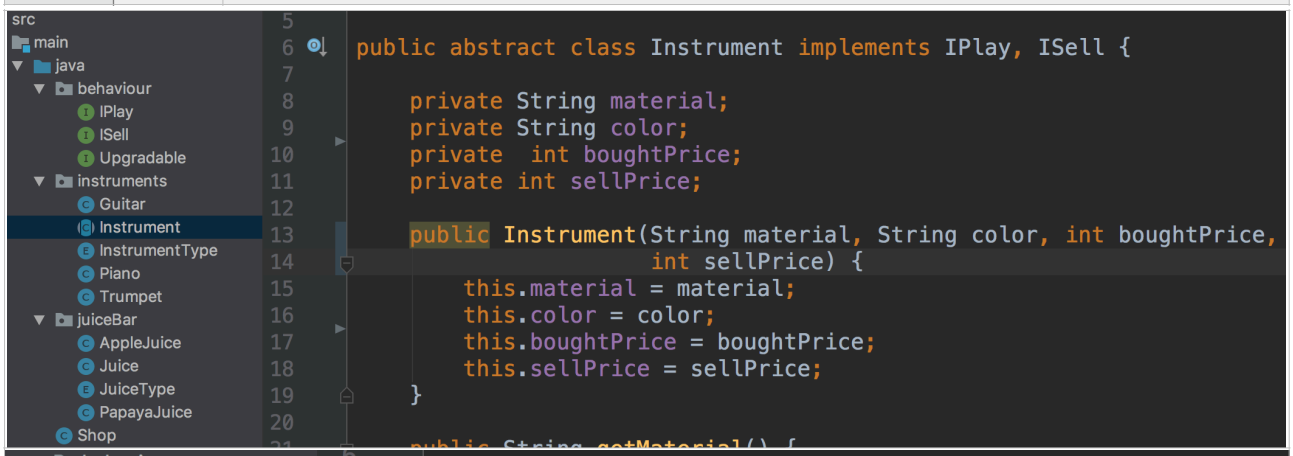
Unit	Ref	Evidence
P	P.17	Produce a bug tracking report
		Description: Error show up in the group project.

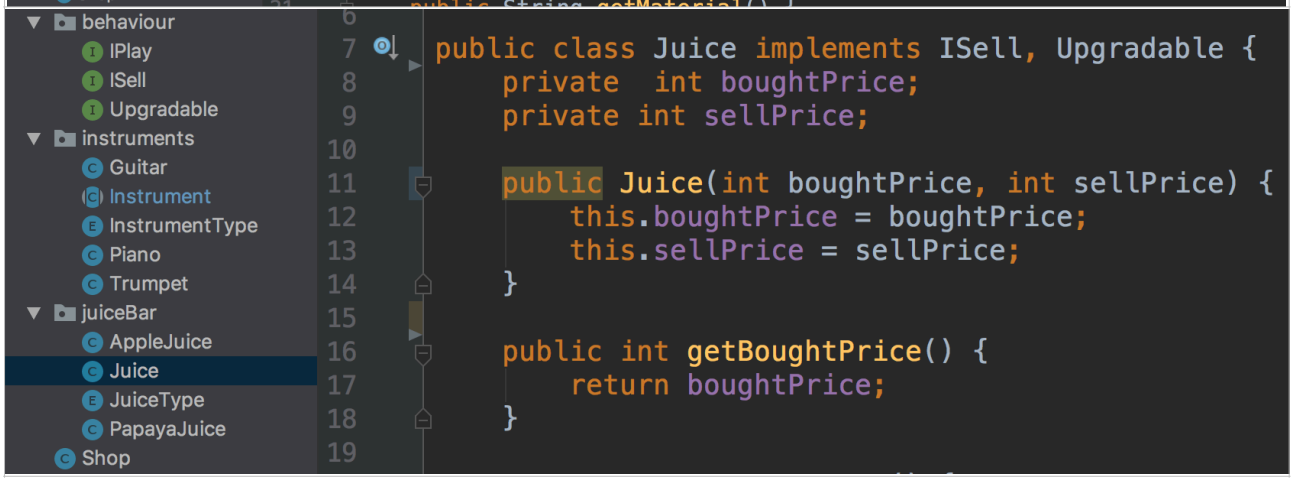
Debugging Event	Status	Solution	Result
Empty lines in the end of seeds.js file	Error, Mongoldb can not read the file	Remove empty lines	Pass. Data displayed in Mongodb
Missing comma in the seeds.js	Error, Mongoldb can not read the file	Add the comma	Pass. Data displayed in Mongoldb
Database name is incorrect in the server.js	Insomnia can not find the route	Change the dbname to correct one	Route available, passed.
Typo in server.js (bdoy.parser)	Insomnia shows error	Correct typo	Passed
Logic error; try to get value from key [language.translation[0]	Undefined content	Console log event and pick the data we want. [language.translation[event.detail]	Passed

Paste Screenshot here

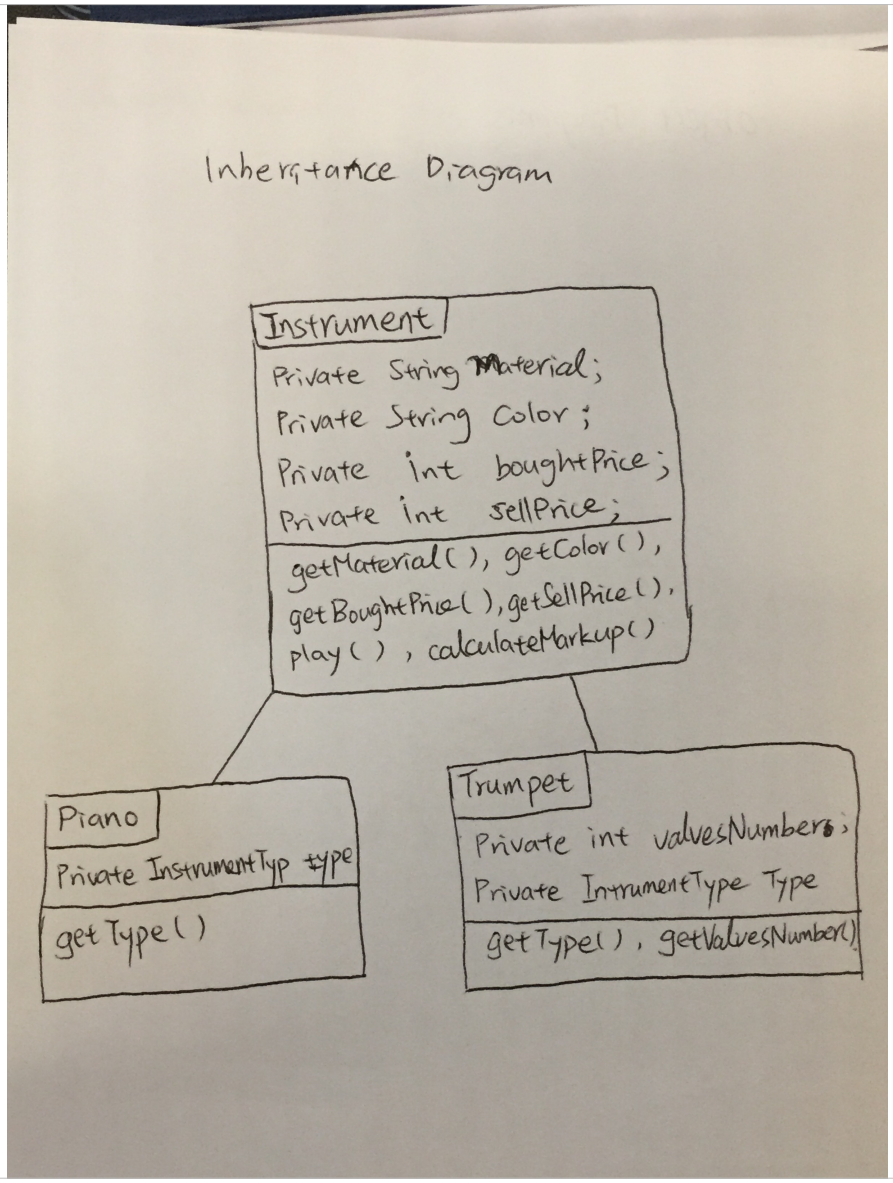
Week 12

Unit	Ref	Evidence
I&T	I.T.7	The use of Polymorphism in a program and what it is doing.
		Description: two classes using same interface e.g.. instrument class and Juice class using ISell interface, so they can be sell in the shop





Unit	Ref	Evidence
A&D	A.D.5	An Inheritance Diagram
		Description: Like class diagram but subclasses have aero to superclass



Unit	Ref	Evidence
I&T	I.T.1	The use of Encapsulation in a program and what it is doing.
		Description: name is private, so the name can not be modified directly.

```java
public class SweetSlot extends Slot {

    private String name;
    private ArrayList<Sweet> sweets;
    private ProductType productType;
    public SweetSlot(double price, int code, int capacity) {
        super(price, code, capacity);
        this.productType = ProductType.SWEET;
        this.sweets = new ArrayList<>();
        this.name = "Chocolate";
    }

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

```java
@Test
public void changeName() {
    sweetSlot.name = "Candy";
    assertEquals( expected: "Candy",sweetSlot.getName());
}
```

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: Godzilla inherits from Kaiju class.

```java
public abstract class Kaiju {
    private String name;
    private int healthValue;
    private int attackValue;
    public Kaiju(String name, int healthValue, int attackValue) {
        this.name = name;
        this.healthValue = healthValue;
        this.attackValue = attackValue;
    }
    public void attack(Vehicle vehicle){
        vehicle.healthValue -= this.attackValue;
    }
```

```java
public class Godzilla extends Kaiju{
    public Godzilla(String name, int healthValue, int attackValue) {
        super(name, healthValue, attackValue);
    }
    public void attack(Vehicle vehicle){
        vehicle.healthValue -= this.getAttackValue()*2;
    }
```

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: BlackJack project. I choose them is because each of they stands for simple and complicate algorithms.

```java
    public void playersDecide(ArrayList<Player> players, Dealer dealer){
        for (Player player : players){
            if (player.getHandValue() <= 16){
                Card card = dealer.deal(deck);
                player.getCard(card);
            }
        }
    }
```

In the BlackJack, the player meant to take another card if the total cards value is less than 16.

```java
public ArrayList<Player> getWinners(ArrayList<Player> players, Dealer dealer){
    //check who is winner 1. dealer busted(players who stay win
    ArrayList<Player> winners = new ArrayList<>();
    if (dealer.getHandValue()>21){

        for (Player player :players){
            if (player.getBusted() == false){
                winners.add(player);
            }
        }
        return winners;
    }else{
        Boolean isDraw = false;
        for (Player player: players){
            //!player.getBusted() == false
            if (player.getHandValue() >= dealer.getHandValue() && !player.getBusted()){
                winners.add(player);

                if (player.getHandValue() == dealer.getHandValue() && !player.getBusted()){
                    if (isDraw == false){
                        winners.add(dealer);
                    }
                    isDraw = true;
                }
            }
        }
        return winners;
    }
```

In the Black Jack, there are situations that indicate the winner.
1. When dealer is busted and player is not then the player win.
2. When dealer is not busted but player is, then dealer wins.
3. When dealer is not busted and so as player. And dealer and player have same value of the cards then its a draw.
4. When dealer is not busted and so as player. And dealer has bigger value of the cards than player, then dealer wins.
5. When dealer is not busted and so as player. And player has bigger value of the cards than dealer, then player wins.