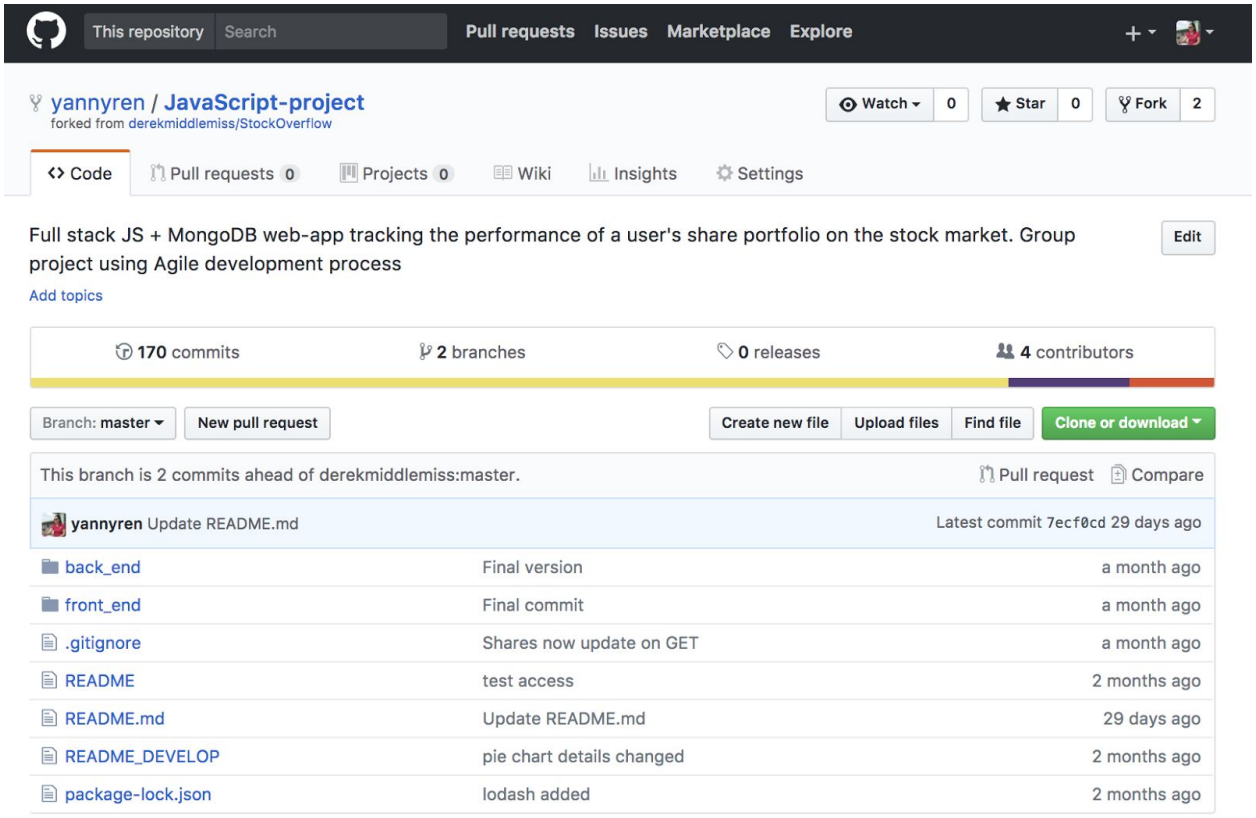
Project Unit
SQA PDA: Software Development
Yan Ren

P1 - Group Project Github



This repository Search Pull requests Issues Marketplace Explore

yannyren / JavaScript-project
forked from derekmiddlemiss/StockOverflow

Watch 0 Star 0 Fork 2

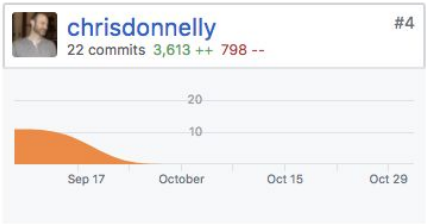
Code Pull requests 0 Projects 0 Wiki Insights Settings

- Pulse
- Contributors
- Traffic
- Commits
- Code frequency
- Dependency graph
- Network
- Forks

Sep 10, 2017 – Nov 3, 2017

Contributions: Commits

Contributions to master, excluding merge commits



© 2017 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub API Training Shop Blog About

P2 - Group Project Brief

README.md

Full Stack JavaScript Project Brief

1. Full stack JS webapp tracking the performance of a user's share portfolio on the stock market.
2. Data stored in a Mongo DB on the back end and updated when required by calls to an external market tracking API.
3. Bring in company news from the GoogleFinance RSS feed.
4. If possible, include simple predictive analysis of stock prices.
5. Practice agile development methodology throughout, with daily stand-ups, backlog charts, sprint plans and retrospectives
6. using Git and GitHub for collaborative development.

Project Name: Stock Overflow

----- MoSCoW -----

MUST

1. Display total value of portfolio
2. Display indivisual trends (7 days range)
3. Talk to external API(given by product manager)
4. Select new shares from market

SHOULD

1. Shares Prediction Model Version 1 (Based on previous 6 weeks trading result)
2. Link to news API(GoogleFinance RSS feed)

COULD

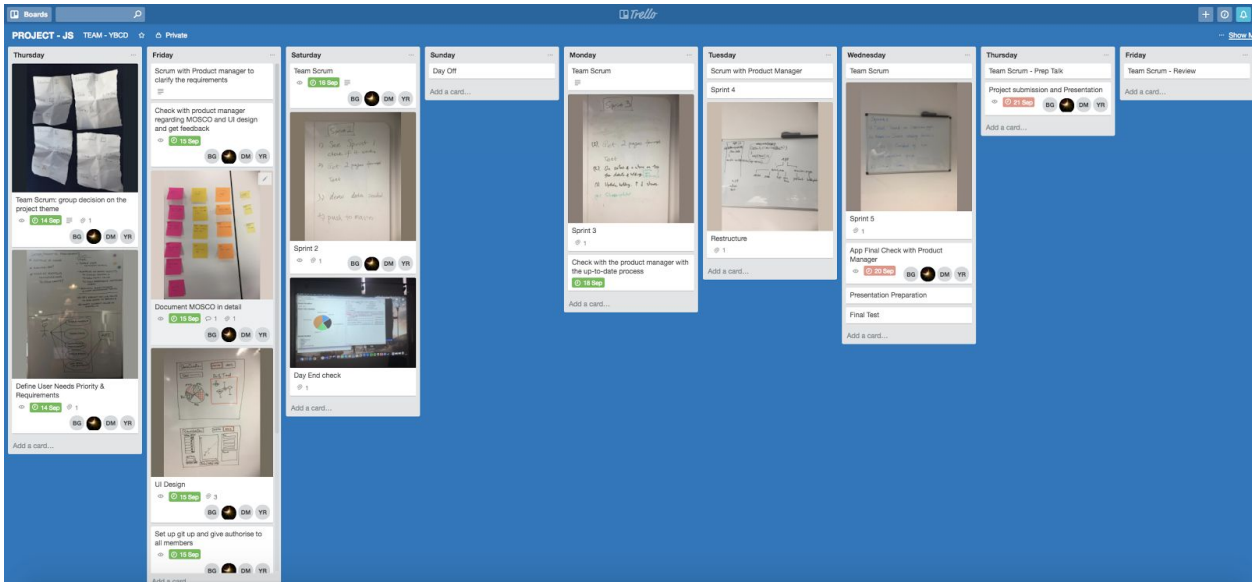
1. Prediction model V2
2. Shares recommendation accrording to V2
3. Shares Comparison

WOULD

1. Live Trading
2. Connection with Bank Account

Note: At the end of the project week, the "must" and "should" parts are done. Next step is to work on "could" and "would" in due course.

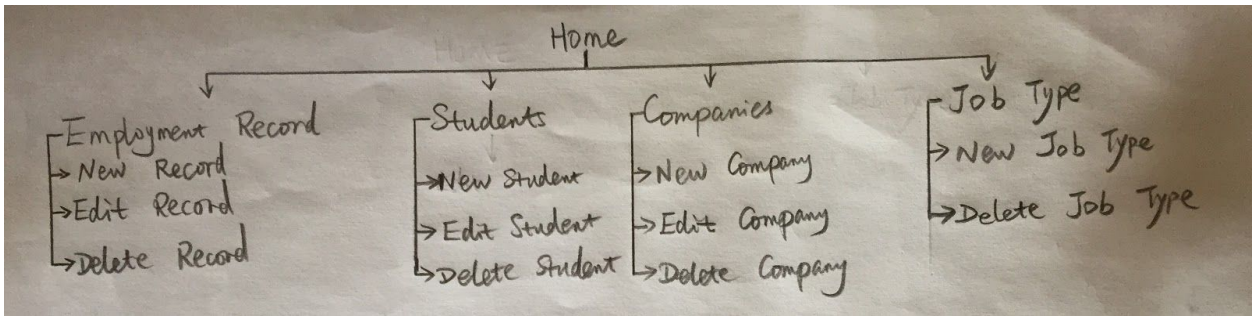
P3 - Group project planning



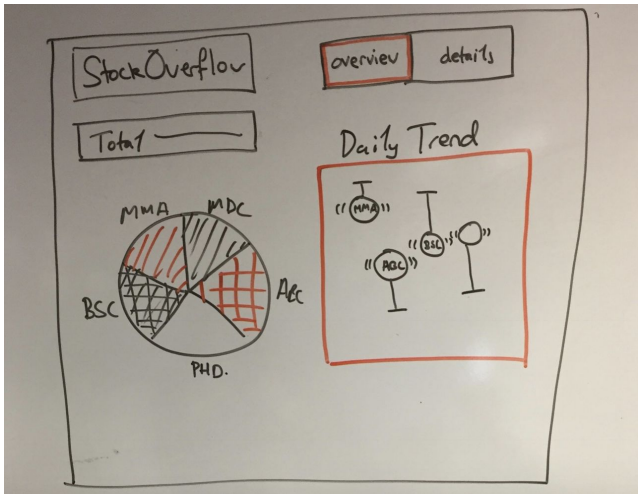
P4 - User acceptance and test plan (for final project)

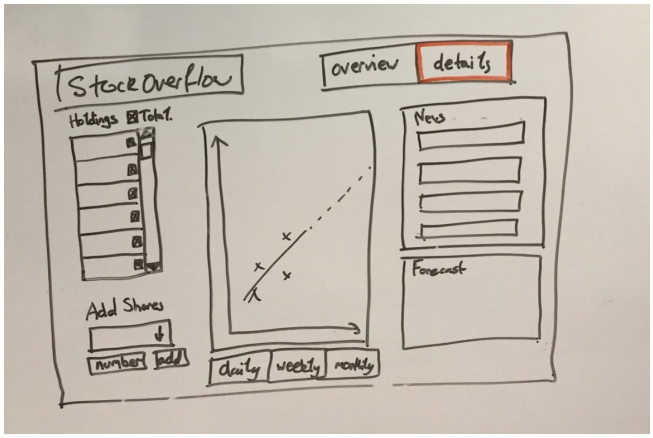
Acceptance Criteria	Expected Result/Output	Pass/Fail
A user can choose between the summary of the stock portfolio and the details page	User click on the switch icon on the righthand corner of the page	pass
A user can choose a stock	User input the stock number then search in the database. Once it is found, click on add button	pass
A user can see the related news of a chosen stock	User choose a stock on the leftside bar. The related recent news will be shown on the right handside column.	pass

P5 User Site Map

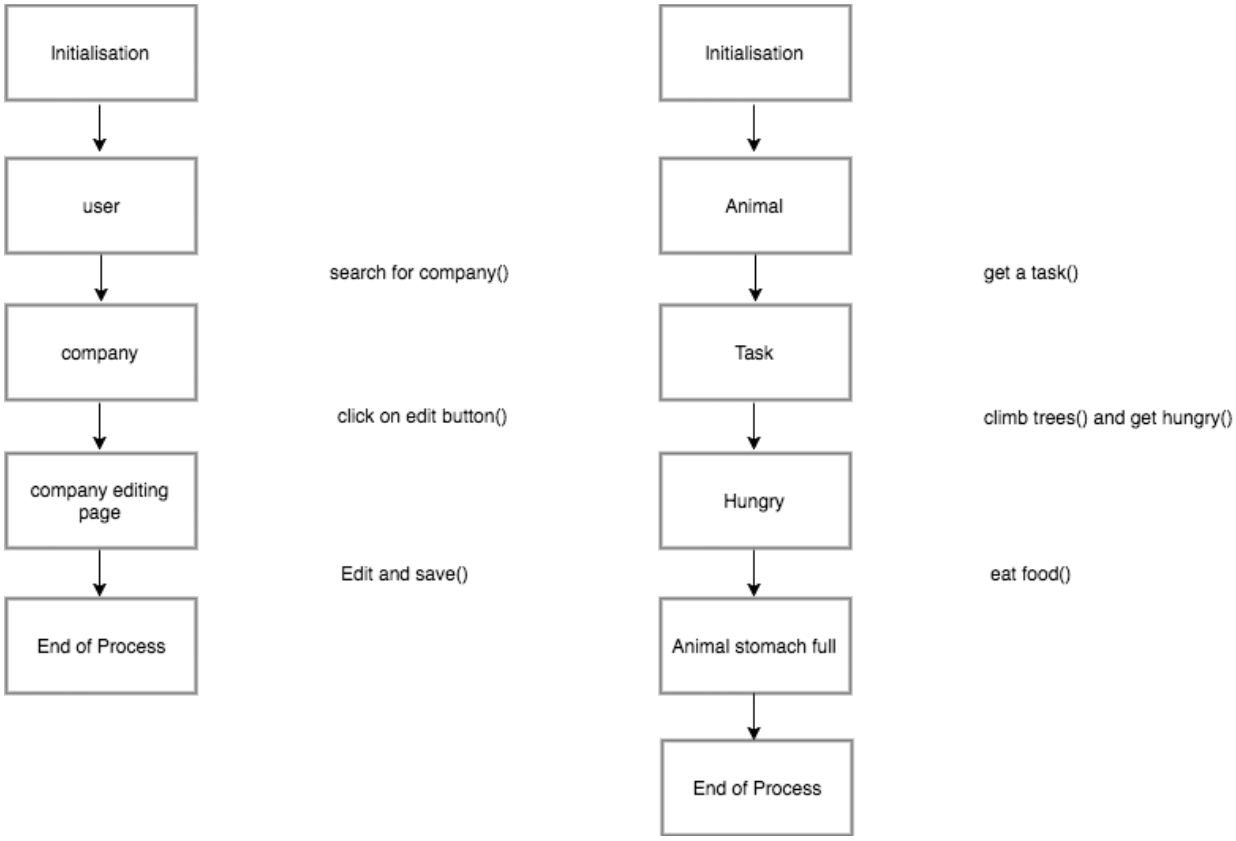


P6 - Wireframe Designs

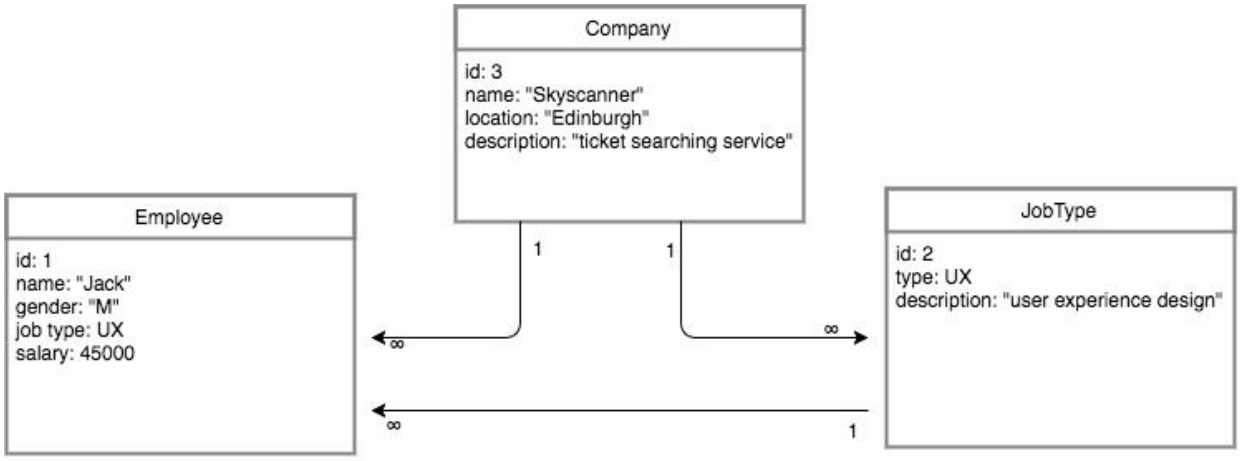


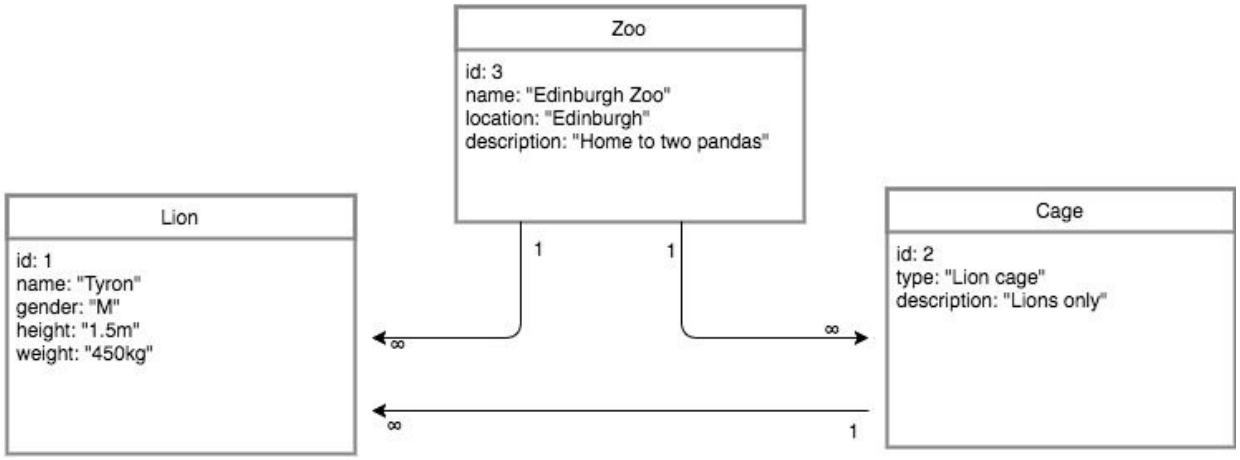


P7 - System Interaction Diagrams (Collaboration Diagrams)



P8 Object Diagrams





P9 Algorithms

In this algorithm, first we check if the animal object in the arraylist of animals is not null. If it is not null, we add the cash value of the animal to the total cash and then return the total.

```
public int getTotalCashValue() {
    int total = 0;

    for(Animal a : this.animals) {
        if (a != null)
            total+=a.getCashValue();
    }
    return total;
}
```

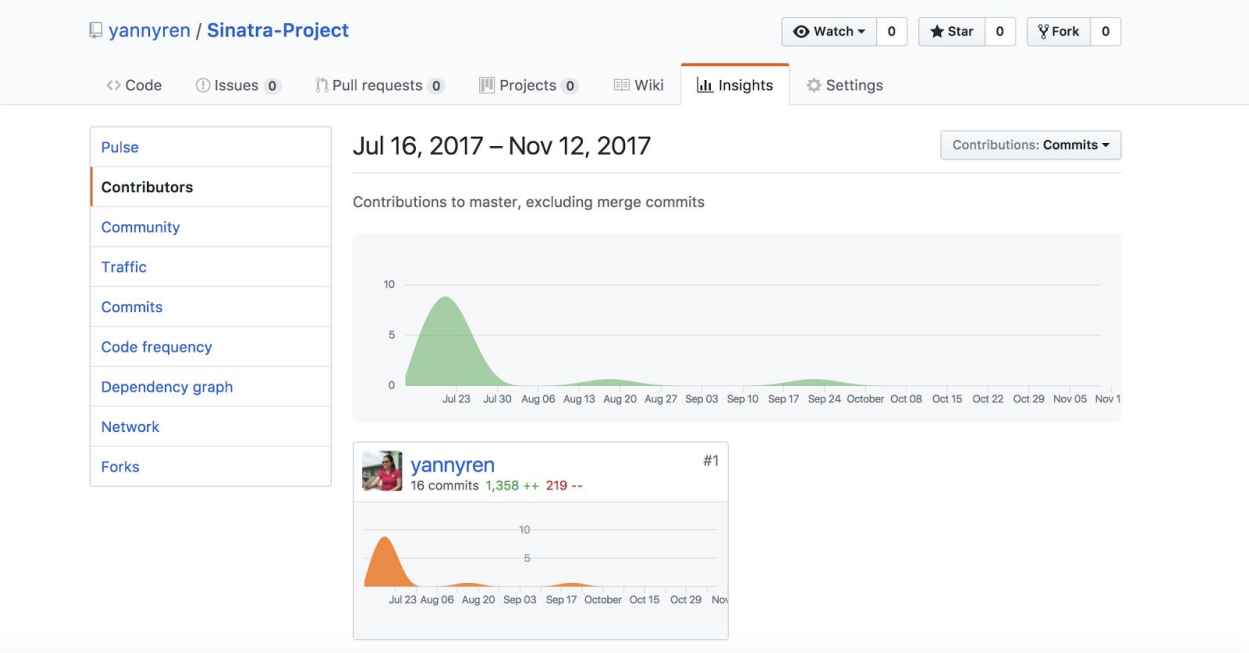
In this algorithm, we want to check if all of the three sides of a triangle is not equal to each other.
If that is the case then the triangle in concern is a scalene. So we have to compare each side with the rest of the two and all results of the three comparisons will need to be false.

```
import java.util.ArrayList;

class Triangle {

    private double side1;
    private double side2;
    private double side3;

    Triangle(double side1, double side2, double side3)  {
        this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;
    }


    boolean isScalene() {
        if((this.side1 != this.side2) && (this.side1 != this.side3) && (this.side2 != this.side3)){
            return true;
        }
            return false;
    }

}
```

P10 Pseudocode

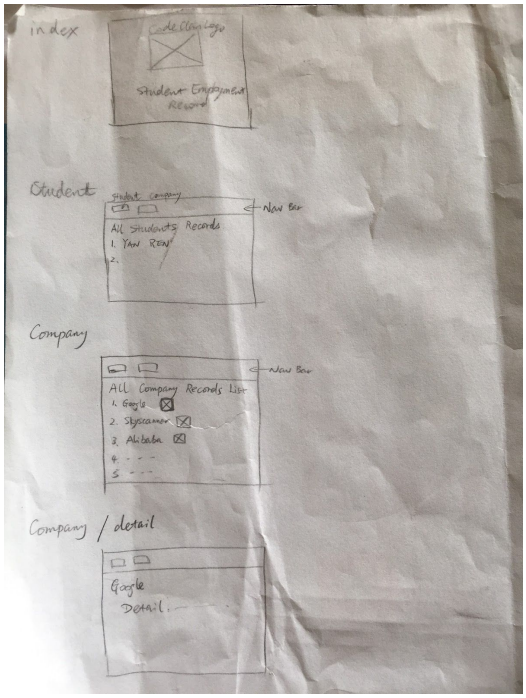
```
public void removeAnimalFromEnclosure(T animal) {
    //check if the concerned animal is in the enclosure using contains method;
    //if it is true, then remove it using remove method;
    //if it is false, then system promps message "no such animal in the enclosure"
```

P11 Project in Github

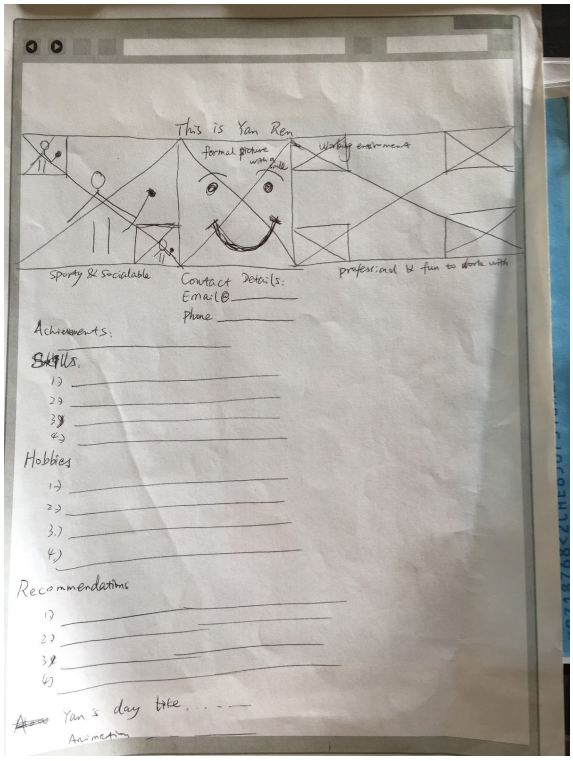


P12 Planning stage

Stage 1



Stage 2



P13 Show user input being processed according to design requirements:

Employment Record | Students | Companies | Job Type | Log Out

New Student Form

Name Jack Ma

Gender Male

Cohort E16

Graduation Date 31/01/2018

Picture Choose file Screen Shot 2... 20.00.16.png

Submit

Employment Record | Students | Companies | Job Type | Log Out

Student List

Add New Student

Student ID	Name	Gender	Cohort	Graduation Date	Company	
4	Craig Lynagh	Male	E14	2017-10-13	Alibaba	Delete
5	Charlie Chaplin	Male	G2	2017-04-10	CodeClan	Delete
6	Andrew Arran	Male	E14	2017-10-13		Delete
7	Alice Prince	Female	G1	2016-09-16		Delete
8	Stacy	Female	E14	2017-10-13		Delete
3	Chris Donnelly	Male	E14	2017-10-13	CodeClan	Delete
10	Ross Fisher	Male	G3	2017-10-21		Delete
11	Yan Ren	Female	E14	2017-10-13		Delete
2	Guy Bach	Male	E14	2017-10-13		Delete
12	Jack Ma	Male	E16	2018-01-31		Delete

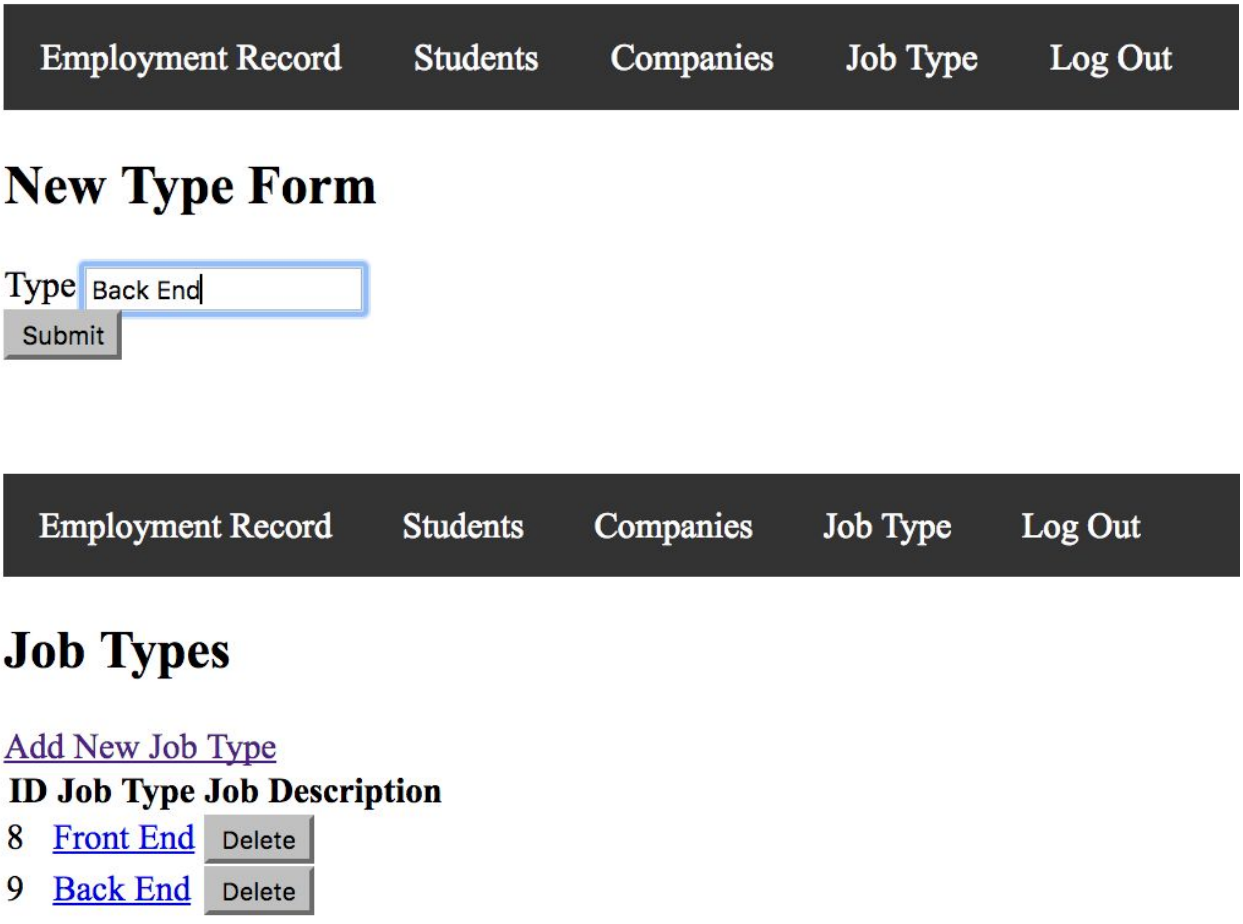
P14 Show an interaction with data persistence

Employment Record | Students | Companies | Job Type | Log Out

Job Types

Add New Job Type

ID	Job Type	Job Description	
8	Front End		Delete



P15 Show the correct output of results and feedback to user

Employment Record | Students | Companies | Job Type | Log Out

Employment Record

Add New Employment Record

Edit	Job ID	Student	Company	Type	
Edit	24	Craig Lynagh	Airbnb	Front End	delete
Edit	25	Andrew Arran	CodeClan	Back End	delete
Edit	26	Alice Prince	Airbnb	UX	delete
Edit	27	Ross Fisher	Airbnb	UX	delete

P16 API being used

```
var AjaxRequest = require( './services/ajax_request.js');
var DetailsPage = require( './views/details_page_view');
var OverviewPage = require( './views/overview_page_view');

// var detailsPage = new DetailsPage( app.refresh, detailsPageElement  );
// var overviewPage = new OverviewPage( app.refresh, overviewPageElement );

var App = function(){
    this.detailsPage = new DetailsPage( this.refresh.bind(this) );
    this.overviewPage = new OverviewPage( this.refresh.bind(this) );
}

App.prototype.refresh = function(){
        var requestData = new AjaxRequest( "http://localhost:3001/api/portfolio" );
        requestData.get( function( data ){
            this.detailsPage.setData( data );
            this.overviewPage.setData( data );
            this.detailsPage.render();
            this.overviewPage.render();
        }.bind(this))
}

App.prototype.start = function(){

        var overviewPageElement = document.getElementById('overviewpage');
        var detailsPageElement = document.getElementById('detailspage');

        overviewPageElement.style.display = 'block';
        detailsPageElement.style.display = 'none';

        var overviewbtn = document.getElementById('overviewbtn');
        overviewbtn.addEventListener('click', function() {
            overviewPageElement.style.display = "block";
            detailsPageElement.style.display = "none";
        })

        var detailsbtn = document.getElementById('detailsbtn')
        detailsbtn.addEventListener('click', function() {
            detailsPageElement.style.display = 'block';
            overviewPageElement.style.display = 'none';
        })

        this.refresh();
}
```

P17 Bug tracking report

User can search company name	Fail	User can edit company detail	Pass
User can add up number of employees in a company	Fail	User can check an employee's detail	Pass
User can change company logos	Fail	User can upload documents	Pass
User can delete a company	Fail	User can find details of a job type	Pass
User can delete the whole employment record	Fail	User can log in/out	Pass

P18 Demonstration of testing

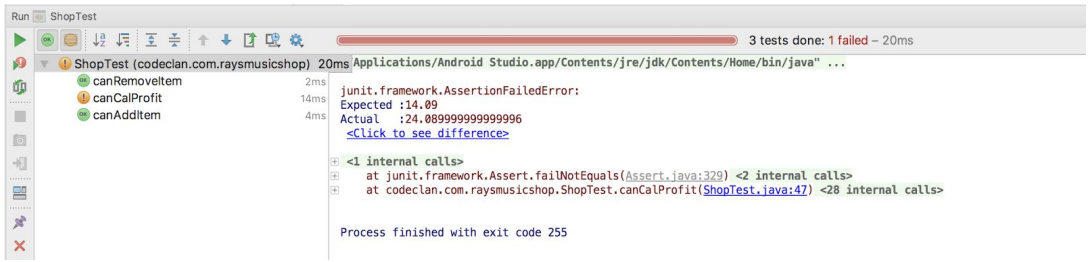
Example of the origin code

```java
package codeclan.com.raysmusicshop;

import java.util.ArrayList;

import codeclan.com.raysmusicshop.Behaviour.Sellable;

/**
 * Created by yanren on 06/11/2017.
 */

public class Shop {
    private ArrayList<Sellable> stock;

    public Shop (ArrayList<Sellable> stock) {
        this.stock = stock;
    }

    public void addItemToStock(Sellable sellable) {
        stock.add(sellable);
    }

    public void removeItemFromStock(Sellable sellable) {
        stock.remove(sellable);
    }

    public double totalPotentialProfit(ArrayList<Sellable> stock) {
        double profit = 10;
        for(Sellable item : stock ) {
            profit += item.calculateMarkup();
        }
        return profit;
    }
}
```

Test file

```java
package codeclan.com.raysmusicshop;

import org.junit.Before;
import org.junit.Test;

import java.util.ArrayList;

import codeclan.com.raysmusicshop.Behaviour.Sellable;

import static junit.framework.Assert.assertEquals;

/**
 * Created by yanren on 06/11/2017.
 */

public class ShopTest {
    Shop shop;
    ArrayList<Sellable> stock;
    Guitar guitar;
    GuitarStrings guitarStrings;

    @Before
    public void before (){
        this.stock = new ArrayList<>();
        this.shop = new Shop(stock);
        this.guitar = new Guitar( buyingPrice: 34.24,  sellingPrice: 45.33,  material: "wood",  colour: "red", Type.STRING,  numOfString: 3);
        this.guitarStrings = new GuitarStrings( buyingPrice: 12.11,  sellingPrice: 15.11,  description: "strong",  strengthLevel: 5);
    }

    @Test
    public void canAddItem(){
        this.shop.addItemToStock(guitar);
        assertEquals( expected: 1, this.stock.size());
    }

    @Test
    public void canRemoveItem(){
        this.shop.addItemToStock(guitar);
        this.shop.removeItemFromStock(guitar);
        assertEquals( expected: 0, this.stock.size());
    }

    @Test
    public void canCalProfit() {
        this.shop.addItemToStock(guitar);
        this.shop.addItemToStock(guitarStrings);
        assertEquals( expected: 14.09, shop.totalPotentialProfit(stock),  delta: 0.01);
    }

}
```

Test failed



Corrected code

Shop totalPotentialProfit()

```java
package codeclan.com.raysmusicshop;

import java.util.ArrayList;

import codeclan.com.raysmusicshop.Behaviour.Sellable;

/**
 * Created by yanren on 06/11/2017.
 */

public class Shop {
    private ArrayList<Sellable> stock;

    public Shop (ArrayList<Sellable> stock) {
        this.stock = stock;
    }

    public void addItemToStock(Sellable sellable) {
        stock.add(sellable);
    }

    public void removeItemFromStock(Sellable sellable) {
        stock.remove(sellable);
    }

    public double totalPotentialProfit(ArrayList<Sellable> stock) {
        double profit = 0;
        for(Sellable item : stock ) {
            profit += item.calculateMarkup();
        }
        return profit;
    }
}
```

Test passed