PART TIME JOB LOCATOR

BY

KEE SOON CHING

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfilment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Perak Campus)

MAY 2015

DECLARATION OF ORIGINALITY

I declare that this report entitled "**PART TIME JOB LOCATOR**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _____

Name : _____

Date : _____

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Ms Saw Seow Hui, who has given me encouragement and guidance throughout the project development. Besides that, I would also like to thank my friend Lye Guang Xing for constantly giving me feedback and suggestion throughout the project.

Lastly, I would also like to express my gratitude to my family who have supported me throughout the course.

ABSTRACT

The purpose of this project is to develop an system for the use of both employers and jobseekers to improve the hiring process of part time vacancy. The proposed system utilise user's current location to generate the nearest distance match between the job opportunity and the jobseeker. Contradicts to conventional job listing board, the system can sort the order of job listing based on the distance to the location of jobseeker or wages of the job opportunity. The system aims to ease and simplify the process of searching part time vacancy of the job seeker and help employer to
list job vacancy more effectively.

This project uses evolutionary prototyping methodology which a minimal functional working prototype is built initially and additional requirement and feature is implemented iteratively later on. This project is developed using Ruby on Rails framework and iOS for web platform and mobile platform respectively. The reason for choosing Ruby on Rails and iOS for this project is because they enable rapid prototyping as not much prior configuration is needed and feature can be added or removed quickly.

The final deliverable includes a web application for employer and a mobile application for jobseeker.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	Application Programming Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transer Protocol
IDE	Integrated Development Environment
IP	Internet Protocol
JSON	Javascript Object Notation
OS	Operating System
RAM	Random Access Memory
REST	Representational State Transfer
SDK	Software Development Kit
SQL	Structured Query Language
SSD	Solid State Drive
SSH	Secure Shell
UI	User Interface
VPS	Virtual Private Server

Chapter 1 : Introduction

1.1 Background

Job search portal was not common in Malaysia until the creation of JobStreet.com in the year 1997. Since then, various job search portal and job career community has appeared in the local market to simplify and fasten the job recruitment process for both the employer and employee. Job search portal enabled employer to reach a wider audience of potential employee and potential hire can search jobs according to their desired criteria such as job category, location of work, expected salary and etc.

JobStreet.com was the first job search platform established in Malaysia which started its operation since 1997. JobStreet.com provides automated profile matching for both employer and employee with their own proprietary algorithm, they also provides various career advancement resources such as language assessment and salary comparison. Another major job search platform established in Malaysia is JenJobs.com which was founded in 2006. JenJobs provides a simple platform to let employer list available opportunity and alerts jobseeker latest jobs that match pre-specified criteria.

Most of the established job search platforms in Malaysia focuses on full time employment which usually requires professional certification such as bachelor degree and etc. However, Part time job and temporary job searching portal remains as a uncultivated niche market here in Malaysia. Most of the part time job listings are scattered around social network groups, online discussion community thus making it difficult for jobseeker to search for part time job and the credibility of the employer may not be verified.

1.2 Problem Statement

Most of the local job portals focus on full time or professional employment which requires full time commitment or formal certification such as university degree or professional certificates, there is a **lack of job portal which focuses on part time job** which can be catered to students and does not require specific hard skill.

Other than that, most of the part time job information are scattered across different website or social media group which are not centralised and hard to discover by user, there may be **inconsistency of job information and outdated information** scattered around the web.

There is also no review mechanism in existing local part time job portal which employer and employee can give review to each other. **Credibility of the employer and the working environment provided could not be easily verified** in existing job portal.

Moreover, currently there is **no local part time job portal which can show nearby part time job opportunity to jobseeker based on their current location**. Jobseeker usually has to manually search and pick for jobs that are nearby to them.

1.3 Motivation

The main motivation for this project is to tackle the niche market of local part time job portal and produce a part time job searching platform which can benefit individuals which does not have formal certification such as university students, housewives and etc. Other than that, employer can also be benefitted as they can reach their target candidate easier by using this platform.

1.4 Objectives

The objectives of the project includes:

i. To develop a backend web application which allows employer to submit job listing and communicate with the mobile application.

ii. To develop an external API (Application Programming Interface) on top of the backend web application to allow the communication between the mobile application and the backend web application.

iii. To develop a mobile application that enables user to view nearby job vacancies based on their current location.

1.5 Impact, significance and contribution

The project will help individual who can't commit full time or doesn't have a formal certification such as university student to find nearby part time job vacancies easily and increase the exposure of job listing for employers. The project will also set a milestone as the first mobile application developed locally which is catered for part time job options. Similar applications has been well received in the United States of America and Canada.

1.6 Project Scope

The main focus of this project is to develop a mobile application that enables user to view nearby job vacancies from their current location and a backend web application which allows employer to input job listing. Employee rating and review system will be implemented into the application as well.

1.7 Structure of the report

This report consists of 6 chapters in total. The first chapter introduces the project which includes the project motivation, objectives, impact significance and scope of the project. In the second chapter, three existing systems in the market are reviewed and compared with the proposed solution. In the third chapter, methodology and tools used to develop the system are elaborated, system requirement and timeline are explained as well. In the fourth chapter, the methodology and tools used in the project in addition to the timeline of the project. The fifth chapter details the implementation and testing done on the project. The final chapter concludes the project and possible future work is discussed.

Chapter 2 : Literature Review

2.1 Review Description

In the following analysis, several job portal applications which currently exist in the marketplace are reviewed and their strength and weakness are compared.

2.2 Jobstreet.com

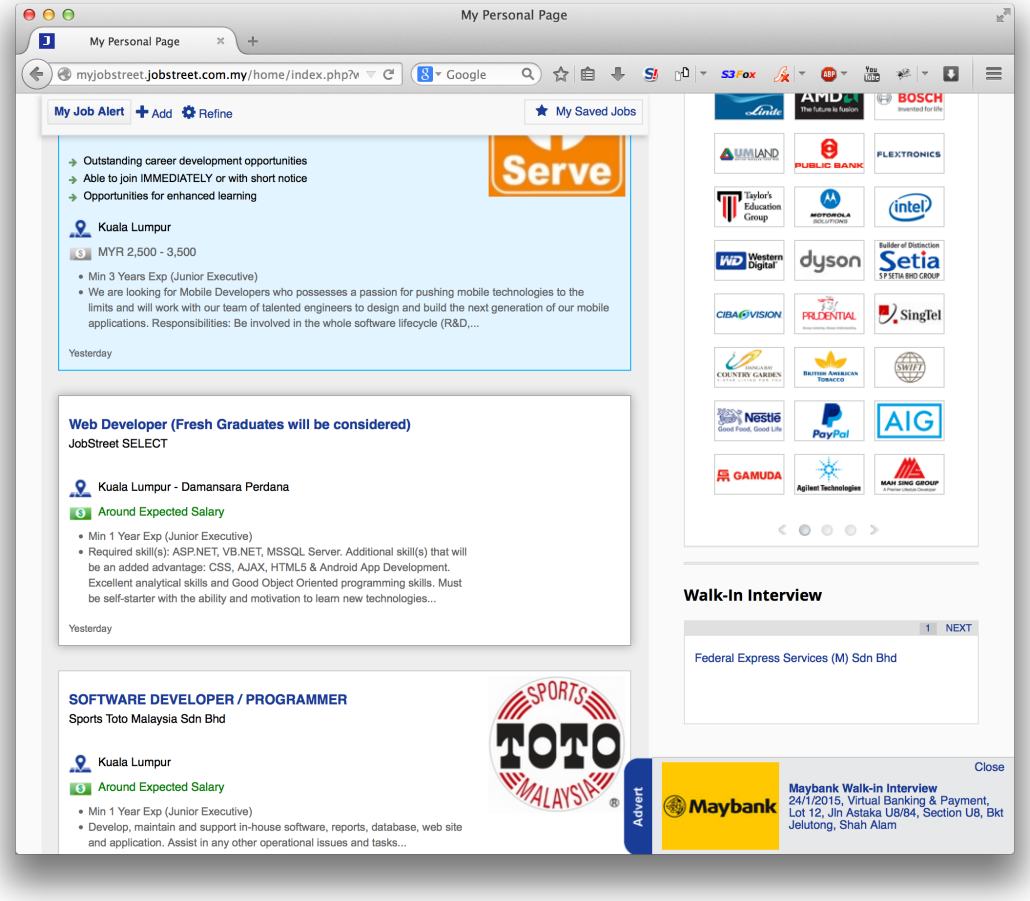

Figure 2-1: Web interface of JobStreet

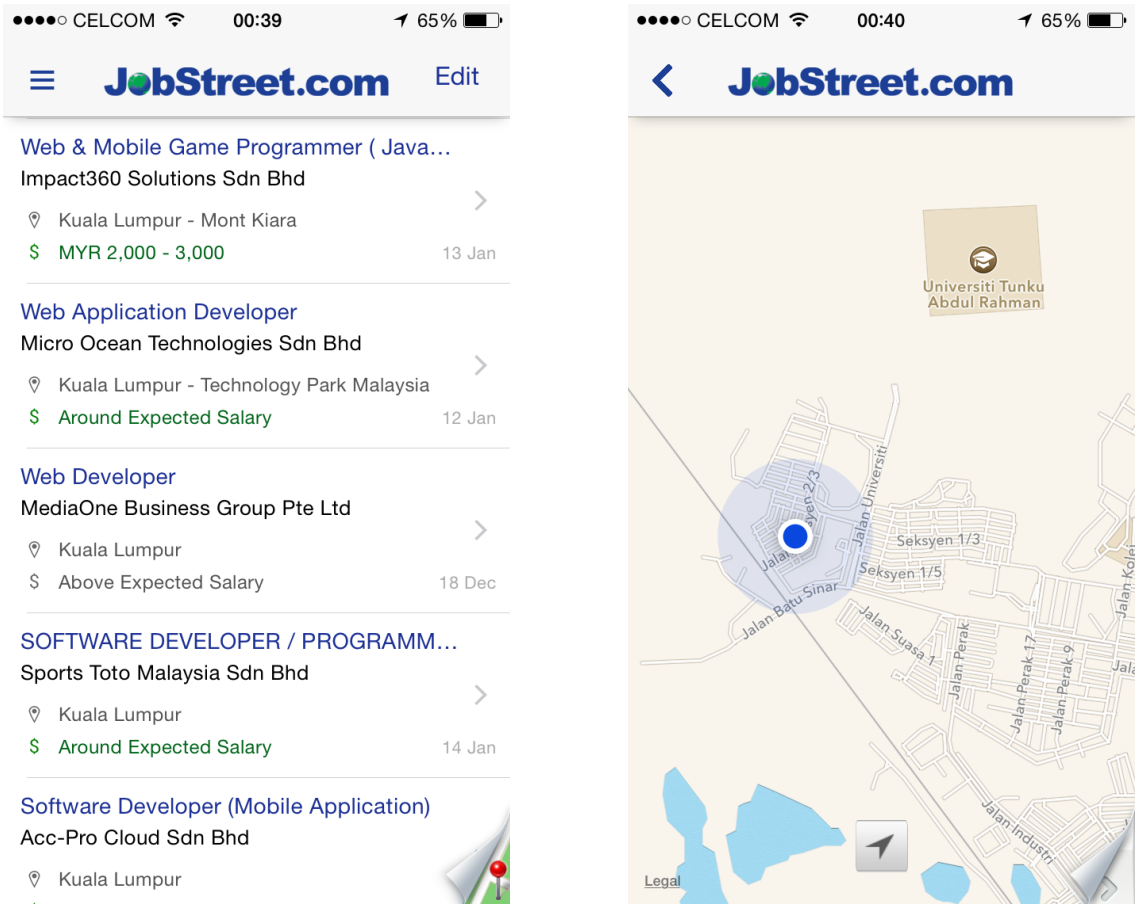

Figure 2-2: Mobile interface of JobStreet

Jobstreet.com is an online job portal that lets employer to post job vacancies and employee to seek job. It span across multiple platform such as website, iOS, Android, Blackberry and Windows Phone.

Other than searching job by specifying various criteria, job seekers can also upload their resume to the online portal and take online assessment such as English Language Assessment to polish their resume.

Jobstreet.com also features a job matching service named LiNa to match jobs which are similar to the criteria specified by the jobseeker (Jobstreet.com 2015).

2.3 JenJobs

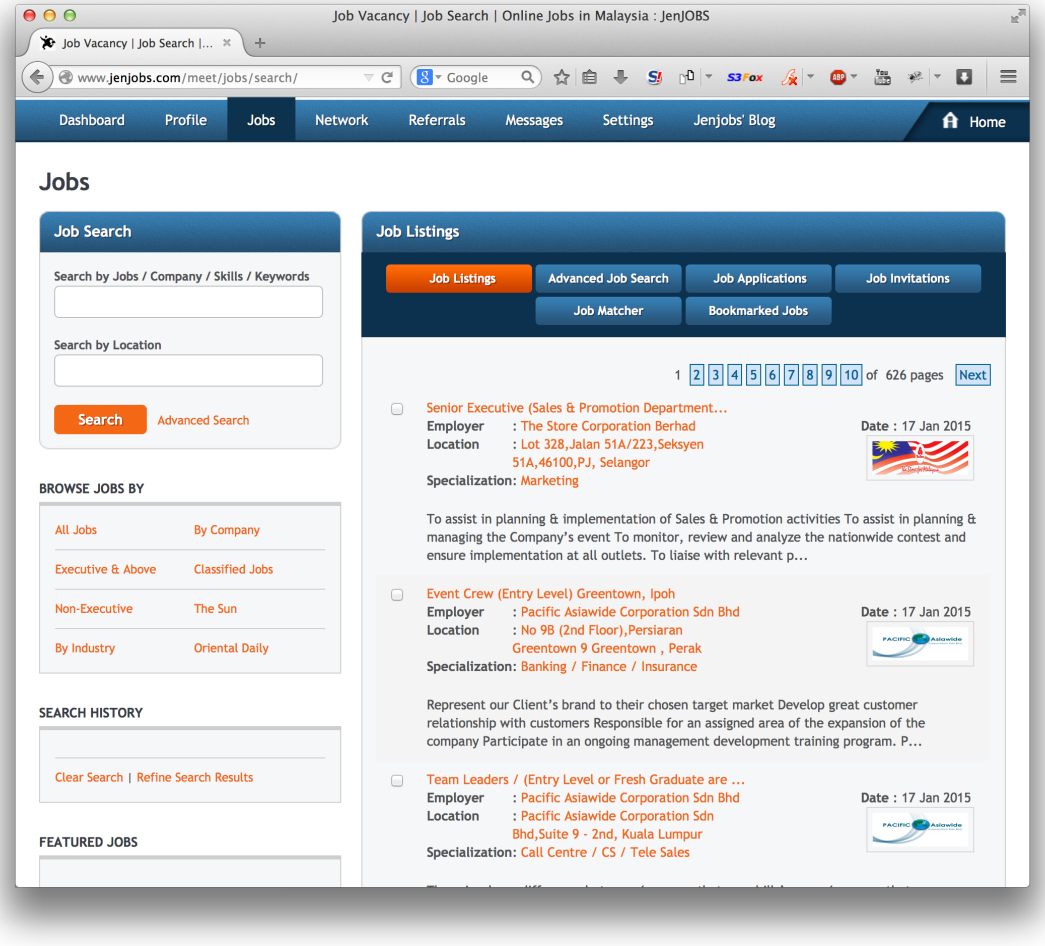

Figure 2-3: Web interface of JenJobs

JenJobs is an online job portal which started to operate since 2006. It provides several features for job seeker such as job search by category, job matcher and resume submission.

Job seekers can input their current skill set, qualifications and expected working environment and the job matcher is automatically select jobs that have met with the criteria specified by the user. Other than that , job seekers can increase their connection by adding friends into their network.

Other than submitting job vacancies, employers can also send invitation to potential hire which fulfil their desired expectations in order to attract the attention of the potential hire.

2.4 Jobless.com.my

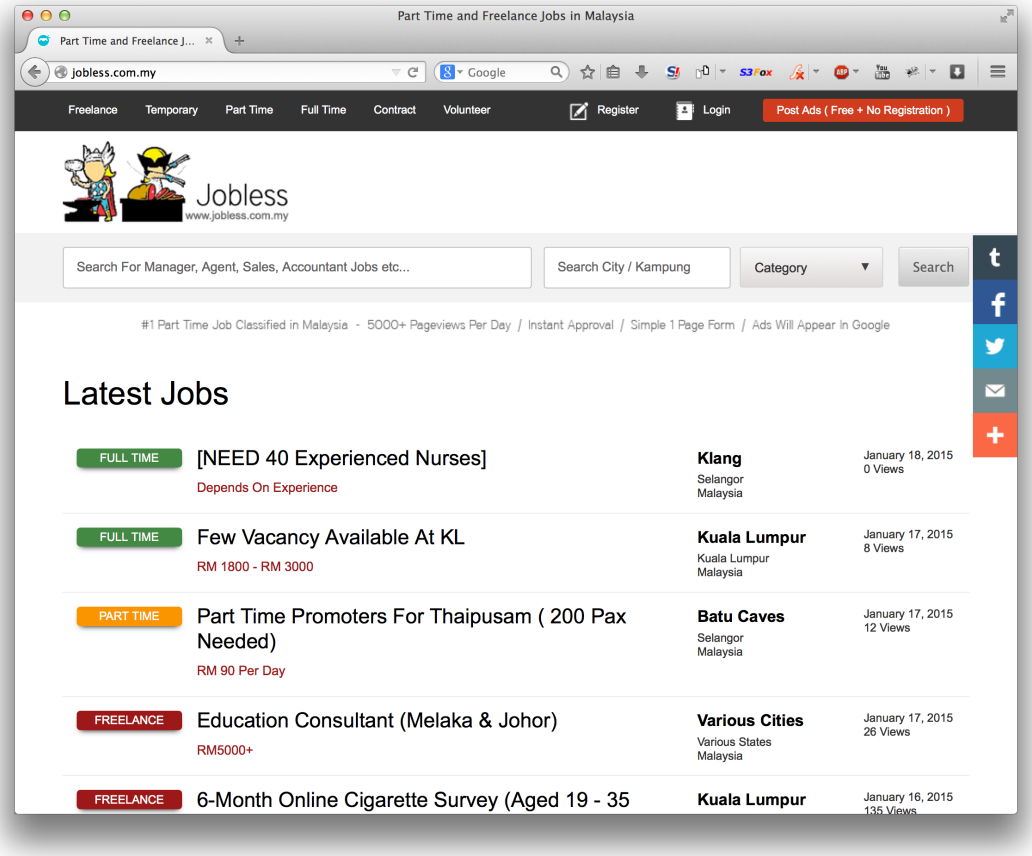

Figure 2-4 : Web interface of Jobless.com.my

Jobless.com.my is a web noticeboard that let any user to post job information and search for certain job details. It features a simple web interface which doesn't require registration in order to find or post job information.

Job seeker can search for different category of job by clicking specific category at the top menu bar. Other than that, user can also type in their desired job position, location and category to search for their desired job.

2.5 Application Comparison

2.5.1 Comparison Table

Application / Criteria	Catered for part time options	Has mobile application	Let user find jobs near their current location	Evaluation and review system
JobStreet.com	✔	✔		
JenJobs				
Jobless.com.my	✔			
Proposed system	✔	✔	✔	✔

Table 2-1: Application comparison table

2.5.2 Criteria Explanation

Criteria	Elaboration
Catered for part time options	The service enables user to find part time jobs which doesn't require external qualifications.
Has mobile application	The service has mobile application platform such as iOS and Android.
Let user find jobs near their current location	The service will list vacant jobs which are near to the current location of the user
Evaluation and review system	The service lets employers review their staff performance and future employer can reference to the previous review of a potential candidate

Table 2-2: Criteria explanation tabler

2.5.3 Comparison Analysis

JobStreet.com currently has more than 11 million jobseekers and 80000 employers (JobStreet.com Annual Report, 2012). JobStreet.com provides an excellent platform for jobseeker and employers with its job matching service, job posting platform and assessment tools. Other than that, JobStreet.com has also expanded to several mobile platform such as iOS , Android, BlackBerry and Windows Phone. However, their focus is more on job that requires professional certification such as university degree and user has to manually search for jobs which are nearby to them.

JenJobs enables user to search or match vacant jobs based on their current skill set, qualifications and expected working environment. Other than that, Job seeker can expand their professional network and employer can send job invitation to a potential hire. Similar to JobStreet.com , JenJobs focuses on professional job which require formal certification and full time commitment. In addition, user has to manually search for jobs which are nearby to them too.

Jobless.com.my on the other hand, feature a simple web interface with decent feature such as job categorisation, job search and job posting. Jobseeker and employer can search for job or post job listing without prior registration. Similar to JenJobs, Jobless.com.my simple web interface only allow user to manually input search criteria and is unable to show nearby available jobs to the user.

Chapter 3 : Methodology

3.1 Methodology

The chosen software development methodology for this project is evolutionary prototyping.

Evolutionary prototyping first start with building a working prototype with minimal functionality and continue to add requirement and feature iteratively. The advantage of evolutionary prototyping is that it allows developer to focus and implement well understood requirement first, and subsequent requirements are added after they have been understood by the developer.

The development model is shown in Figure 3-1.

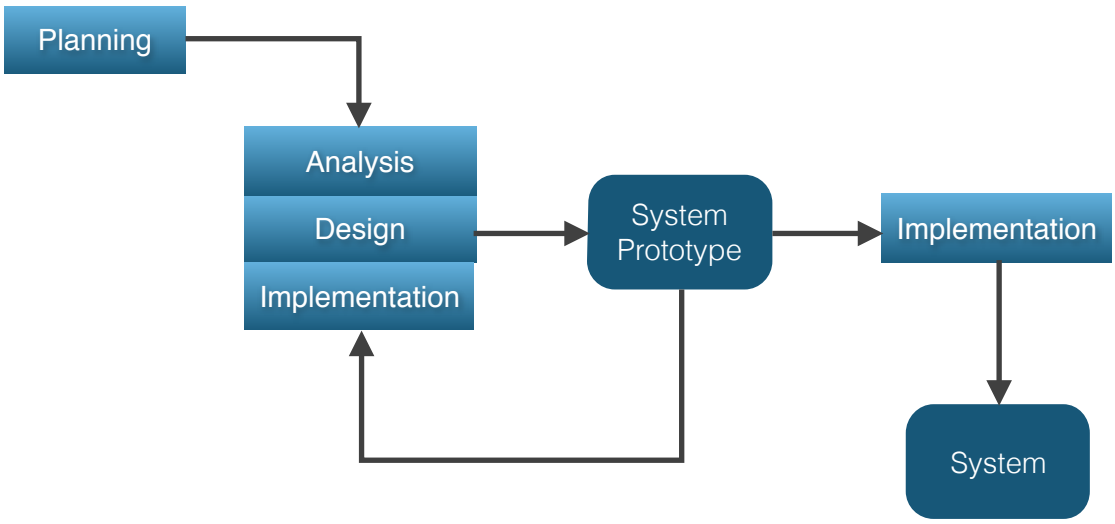

Figure 3-1 : Prototyping model

3.1.1 Planning

Tasks executed in the planning phase include understanding the purpose of the system, defining the scope of the system, analysing the architecture of the system, identifying system requirements, setting project workflow and performing feasibility analysis of the system.

3.1.2 Analysis

Tasks executed in the analysis phase include analysing the gathered system requirements, documenting finalised system requirements and plotting gantt chart for time required for each task of this project.

3.1.3 Design

Tasks executed in the design phase include developing design strategy and mapping requirements into various UML diagram.

3.1.4 Implementation

Tasks executed in the implementation phase include writing the code required to build the system, deploying the system into online web server, testing the system using simulator and actual device and fixing defect if found.

3.1.5 Development tools

Tools that are used to develop the proposed application include :

Google Maps SDK

Map and location marker is rendered using Google Maps SDK for iOS.

Xcode IDE

Xcode is used to develop and debug the iOS application.

MySQL

MySQL is an open source relational database management system which is used for the web application for this project.

Ruby on Rails

Ruby on Rails is an open source web application framework which will be used to develop the web application for this project.

Sublime Text 2

Text editor used for editing web application code for this project.

SourceTree

Graphical user interface software used to manage version control and of the project.

OS X v10.10 Yosemite

The project is developed on a machine running OS X.

3.2 Requirements

3.2.1 User Requirements

1. Jobseeker should be able to register and login into the system.
2. Employer should be able to register and login into the system.
3. Jobseeker should be able to view and discover part time job nearby to their current location.
4. Jobseeker should be able to apply for part time job vacancy.
5. Employer should be able to view and approve/decline job application.
6. Employer should be able to insert job listing into the system.
7. Employer should be able to delete job listing from the system.
8. Jobseeker should be able to give review to job employer.
9. Employer should be able to give review to employee.

3.2.2 System Requirements

3.2.2.1 Functional Requirements

1. The system shall allow user to register as either jobseeker or employer.
2. The system shall authenticate the user using provided credential.
3. The system shall select job listing data from the database using user current location coordinate as criteria.
4. The system shall display available job opportunity with its respective coordinate to the user.
5. The system shall allow user to apply for job vacancy.
6. The system shall allow user to approve/decline job application.
7. The system shall allow user to insert job listing into the database.
8. The system shall allow user to delete job listing from the database.
9. The system shall allow user to insert review and rating into the database.

10. The system shall provide an Application Programming Interface (API) for web request and return JavaScript Object Notation (JSON) to each web request.

3.2.2.2 Non-Functional Requirements

1. The deviation of precision of coordinates of each job listing shall not exceed 500m in radius.
2. The system shall be intuitive for user to use without any external guidance.
3. The system shall be able to exit gracefully should any error or exception is encountered.

3.3 Software and libraries involved

3.3.1 Ruby on Rails

Ruby on rails, or in short form, Rails, is an open source web application framework which encourage rapid prototyping and code reuse. Rails was created by David Heinemeier Hansson when he was developing Basecamp web application. He extracted the rails module from the application and open sourced it on 2004. Rails utilise model-view-controller architecture pattern to segregate data processing and information representation, it also encourages the use of Representational State Transfer (REST) architecture for sending and handling web request . Rails is built using the language Ruby.

The two core philosophy of Rails include “Don’t Repeat Yourself”(DRY) and “Convention Over Configuration” (COC) . “**Dont Repeat Yourself**” is an important principle in software development which states that "Every piece of knowledge must have a single, unambiguous, authoritative representation within a system” [The Pragmatic Programmer, *Andrew Hunt* and *David Thomas*, 2000]. It focuses on

reducing code repetition as much as possible to an extend which modification of any unit element of a system does not affect other unrelated components. By having only a single source of a specific information, the system is easier to maintain and extend without affecting the stability of the whole system.

"**Convention over Configuration**" is a software design principle which aims to increase the productivity of developer by reducing the number of decisions which a developer has to make during the software development lifecycle. By default, Rails is opinionated, it assumes and follows a predefined convention on doing and configuring things. By following this standard, developer can skip most of the trivial configuration and dive into the development of the feature quickly. For example, by using the preset ActiveRecord module of Rails, the class named "Sale" corresponds to the table named "Sales" in the database, there is no need to manually specify which class links to which database unless there is a specific need for it.

Rails has been widely adopted and favoured by many startups as it does not require license fees, well suited for agile development and has an active community which release new patches on a weekly basis. Some of the companies which use Rails to develop their core web application include Twitter, Github, AirBnB, Groupon, Shopify, Twitch and Kickstarter.

3.3.2 Ruby

Ruby is an interpreted language designed and created by Yukihiro Matsumoto, known as "Matz" in the ruby developer community. Ruby has a wide list of features which include object-oriented, dynamic type checking, closures, garbage collection and support for unicode character encoding.

Contradict with the design philosophy of most programming language which is to make the program run faster on the machine, the philosophy on the design of Ruby language is to increase the productivity of the programmer and make the process of developing software more enjoyable (Matsumoto, 2003). Hence the ruby language is designed to be as intuitive and as similar to english as possible. According to Matsumoto, Ruby is said to adhere to the Principle of least astonishment (POLA). Principle of least astonishment is a design philosophy which aims to to produce a least surprising behaviour by utilising users' previous knowledge on how a specific feature should work to match the user expectation when the user uses a new product or feature. In other words, by looking at the syntax of Ruby language, users ranging from beginner to experienced programmer can easily understand what function will be performed by the code. For example, the code below output number 1 to 8 :

```
(1..8).each do |i|
  print i
end
```

3.3.3 Representational State Transfer (REST) Architecture

Representational State Transfer (REST) is an architectural style for applications which provide web services through public accessible Application Programming Interface (API). REST architecture is most commonly used in client-server communication over HTTP (Hypertext Transfer Protocol) although it can be also used over other protocol such as FTP (File Transfer Protocol). The goal of the REST architecture is to allow the creation of highly scalable and maintainable web services. REST has

several principles including stateless, cacheable and uniform interface (Fielding, 2000).

The **Stateless** principle means that the communication between client and server does not require prior knowledge of any server context or preset session state. Each request sent by the client must fully represent a state and all state can only be kept on the client side.

Cacheable is an optional constraint which the response for a particular web request can be cached and returned to client should the same request is issued again. This can reduce the amount of processing that needs to be done by the server as the cache result can be stored in a proxy server and sent to the client again.

REST requires all components to communicate through a **single interface**. This result in high portability as interaction between different service uses the same interface, changes in the service implementation does not affect how each service communicate to each other. HTTP adhere to this constraint by default as it provides a list of uniform method on web resource retrieval and modification, the list includes **GET** method for resource retrieval, **POST** method for resource modification.

An example of a RESTful web request of a resource named '*car*' can be illustrated like this : **GET** http://www.automobile.com/cars/123 where '123' is the resource identifier for *car*. The **GET** request will return the representational state of the resource such as JSON (Javascript Object Notation) , XML (Extensible Markup Language) or other valid resource form. A sample of JSON state representation of the web request is as shown below :

```
{
"Make" : "Perodua",
"Model" : "Myvi",
"Year" : 2014
}
```

3.3.4 Git

Git is an open source distributed version control system developed by Linus Torvalds initially to ease the collaboration of Linux kernel development in 2005 . Git emphasise on several features such as able to handle large projects efficiently, simple design , strong and solid support for non-linear development up to thousands of parallel branches and most importantly it is fully distributed which eliminates the need of a central server (Torvalds, 2007).

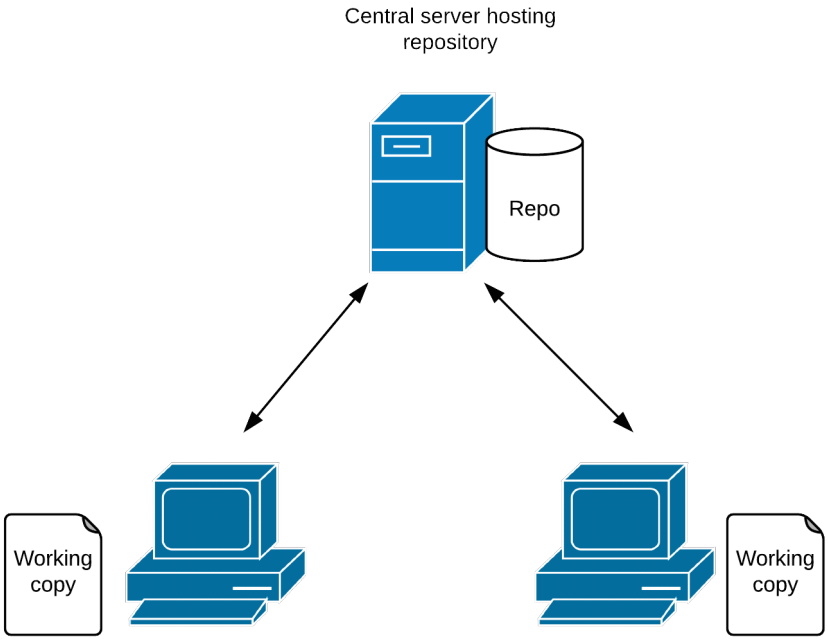

Figure 3-2 : Centralised Version Control

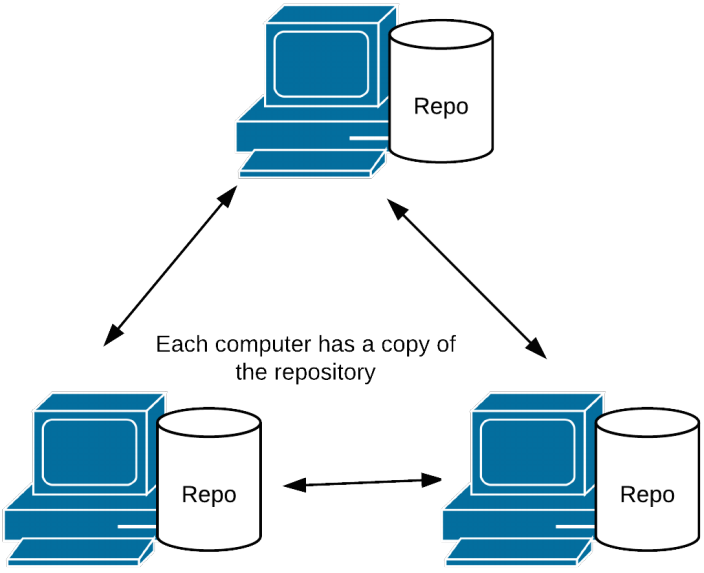

Figure 3-3 : Distributed Version Control

Git has three main states which the file in the repository can reside in which are staged, modified and committed. Modified means that the file has changed but has not committed into the repository. Staged means that the modified file is marked in its current version to go into the next commit. Committed means that the file data is securely stored in the local repository. There are three main section in a git project which include working directory, staging area and git directory.

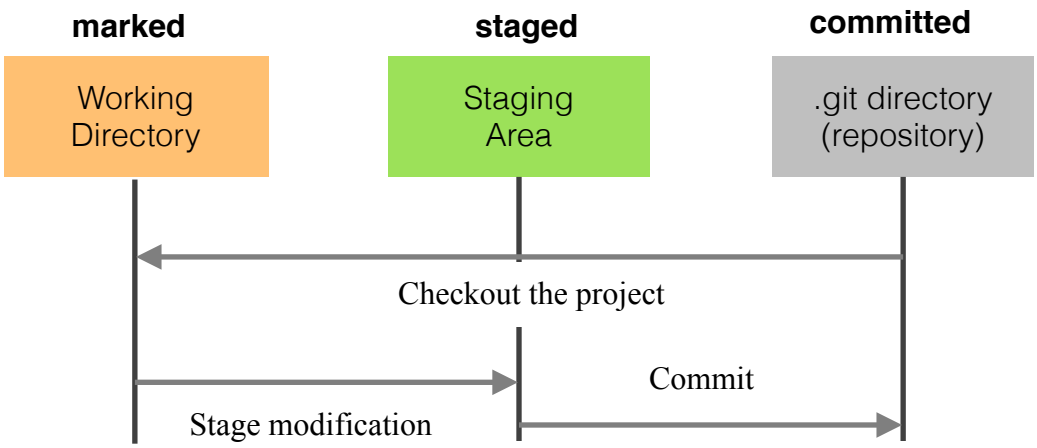

Figure 3-4 : Internal structure of Git

3.4 Installation Steps

There are two main installation procedure for this system which include the web application and mobile application. The installation procedure for each application will be detailed in below.

3.4.1 Web Application

The web application will be hosted on an virtual private server (VPS) hosted on the cloud. DigitalOcean is chosen as the cloud hosting provider as the pricing is affordable and its cloud servers can be scaled easily.

A cloud server will be created using the following specifications : 512 MB RAM , 20GB SSD storage, running Ubuntu version 12.04. Singapore is chosen as the server

region to reduce latency and the communication between the development machine and the server will be encrypted using SSH keys.

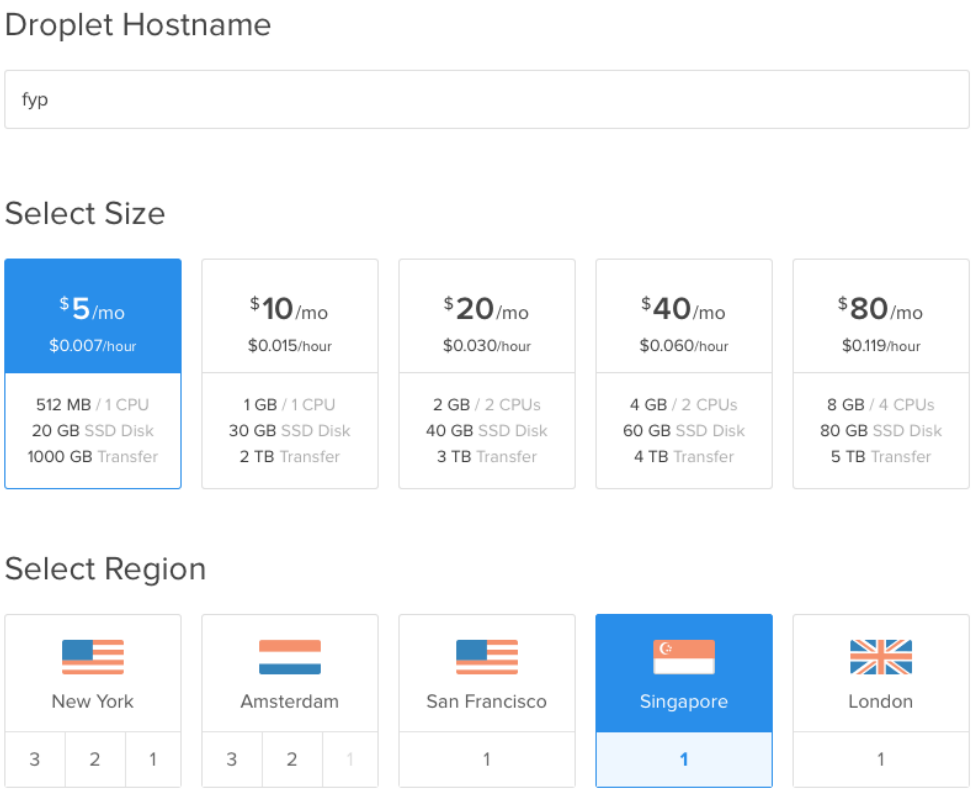

Figure 3-5 : Server size and region selection screen

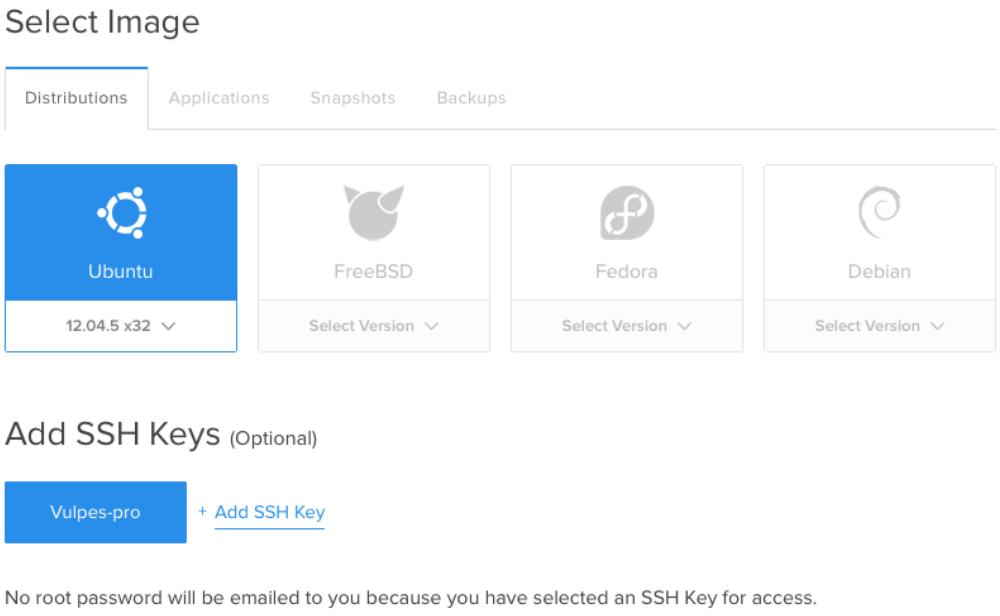

Figure 3-6 : Server image and SSH Keys configuration screen

Ruby language, Rails framework, Nginx web server with passenger module and MySQL database will be installed on the server through SSH using the following commands :

```
$ sudo apt-get update
$ curl -L get.rvm.io | bash -s stable
$ source ~/.rvm/scripts/rvm
$ rvm requirements
$ rvm install 2.2.0
$ rvm use 2.2.0 —default
$ rvm rubygems current
$ gem install rails
$ gem install passenger
$ gem install mysql2
$ rvmsudo passenger-install-nginx-module
$ sudo apt-get install mysql-server libapache2-mod-auth-mysql php5-mysql
$ sudo mysql_install_db
$ sudo /usr/bin/mysql_secure_installation

$ sudo service nginx start
```

Git will also be installed on the server as it will be used to deploy the rails application from the development machine to the server. The installation procedure of Git is shown below :

```
$ sudo apt-get update
$ sudo apt-get install git
$ git config --global user.name “Kee Soon Ching"
$ git config --global user.email “nyancat@1utar.my”
```

After setting up Git, the code repository on the development machine will be deployed to the server using git push command. A database is also created to store the application data using MySQL create database command.

For static assets such as image, Amazon Simple Storage Service (Amazon S3) cloud storage will be used as the storage as it is easily scalable and billed according to the storage space used. Container used to store objects in Amazon S3 are addressed as 'Bucket', a bucket with its region set to Singapore will be created to store images.

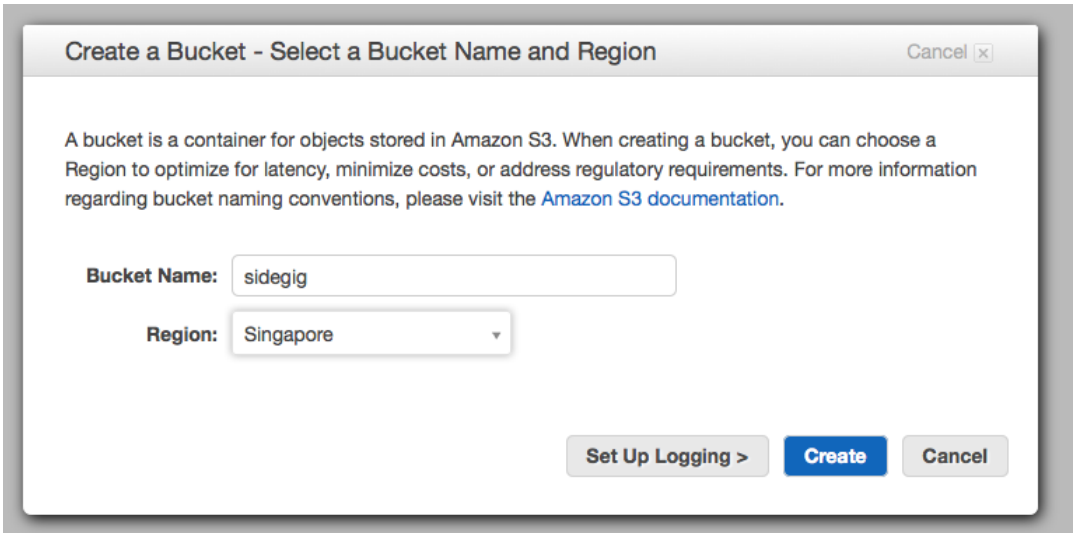

Figure 3-7 : Amazon S3 bucket creation screen

3.4.2 Mobile Application

The mobile application is installed into a physical iPhone running iOS version 8.4 through Xcode.

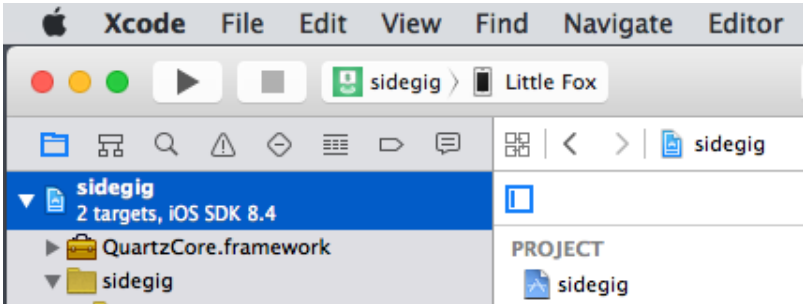

Figure 3-8 : Xcode project interface

3.5 Hardware Involved

3.5.1 Physical iPhone

A physical iPhone running iOS 8.4 manufactured by Apple is used for this project. No additional installation or modification is needed.

3.6 Implementation issues and challenges

There are several issues and challenges which have surfaced during the implementation of this project, one of the challenge includes the communication between the web application and mobile application. During the development phase, the web application code is hosted and executed on the local development machine. In order for the mobile application to communicate with the web application, both development machine and the mobile device must connect to the same local area network and the IP address of the development machine must be fixed and listed in the source code of the mobile application.

Next, the code on both development machine and deployment server must be synchronised once any change is made to ensure consistency and prevent unexpected behaviour. The synchronisation challenge was solved using a version control system to keep track of code repositories on both development machine and deployment server.

3.7 Timeline

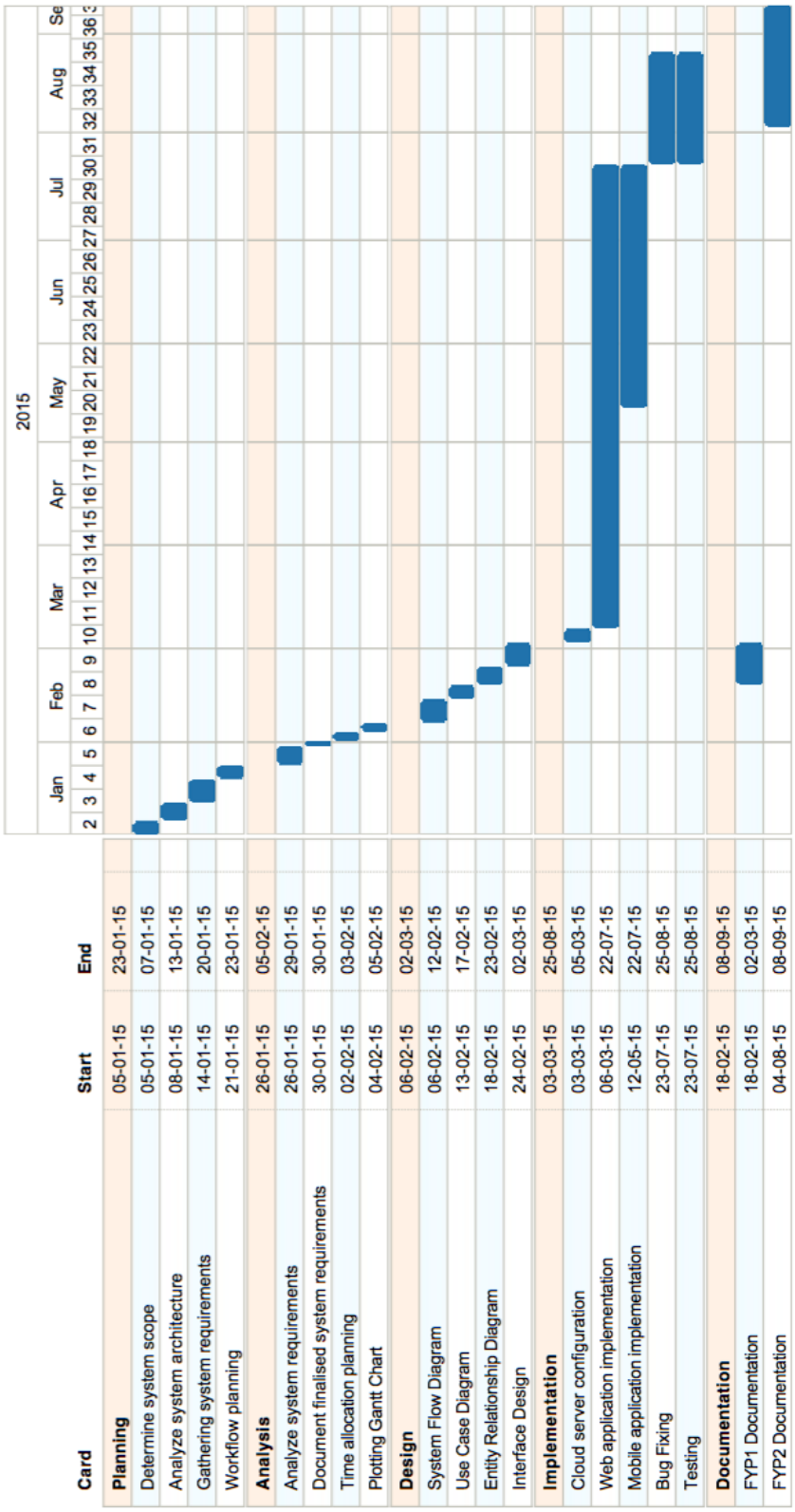

Figure 3-9 : Gantt chart for project timeline

Chapter 4 : System Design

4.1 System Flow

System diagram flow of both mobile application and web application is shown below. Mobile application is used by jobseekers whereas web application is used by employers.

4.1.1 Mobile Application

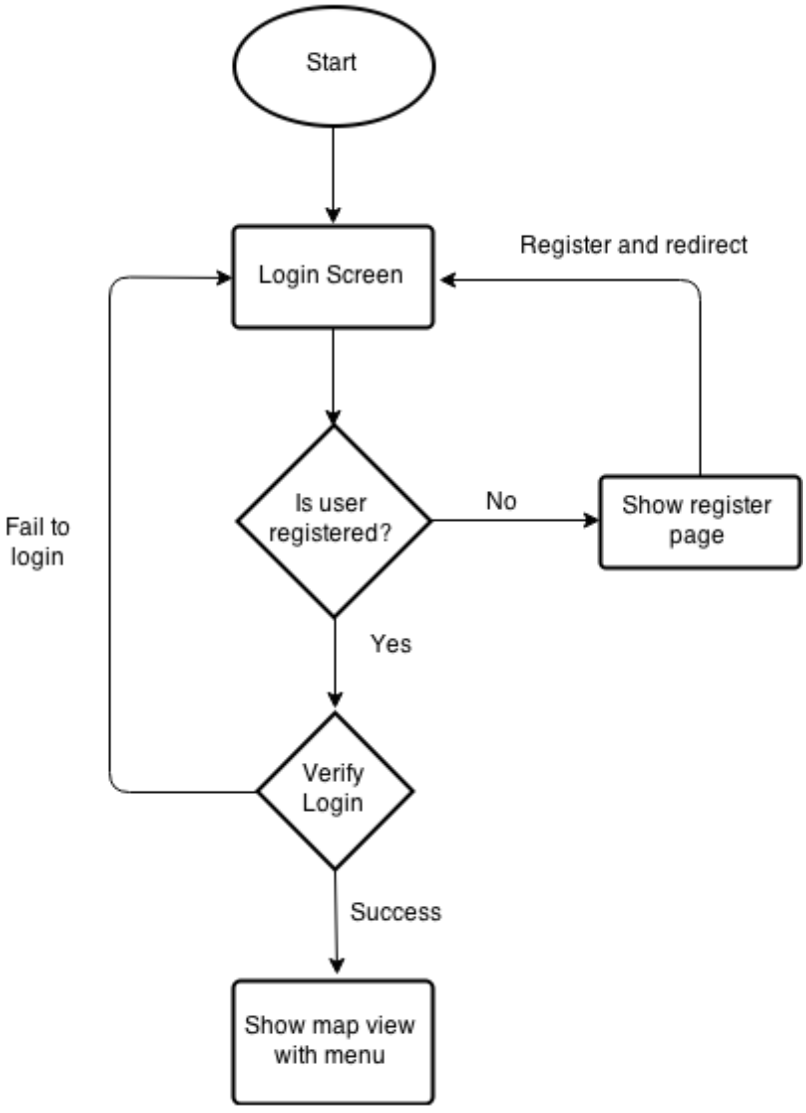

Figure 4-1 : System diagram flow for mobile application (part 1)

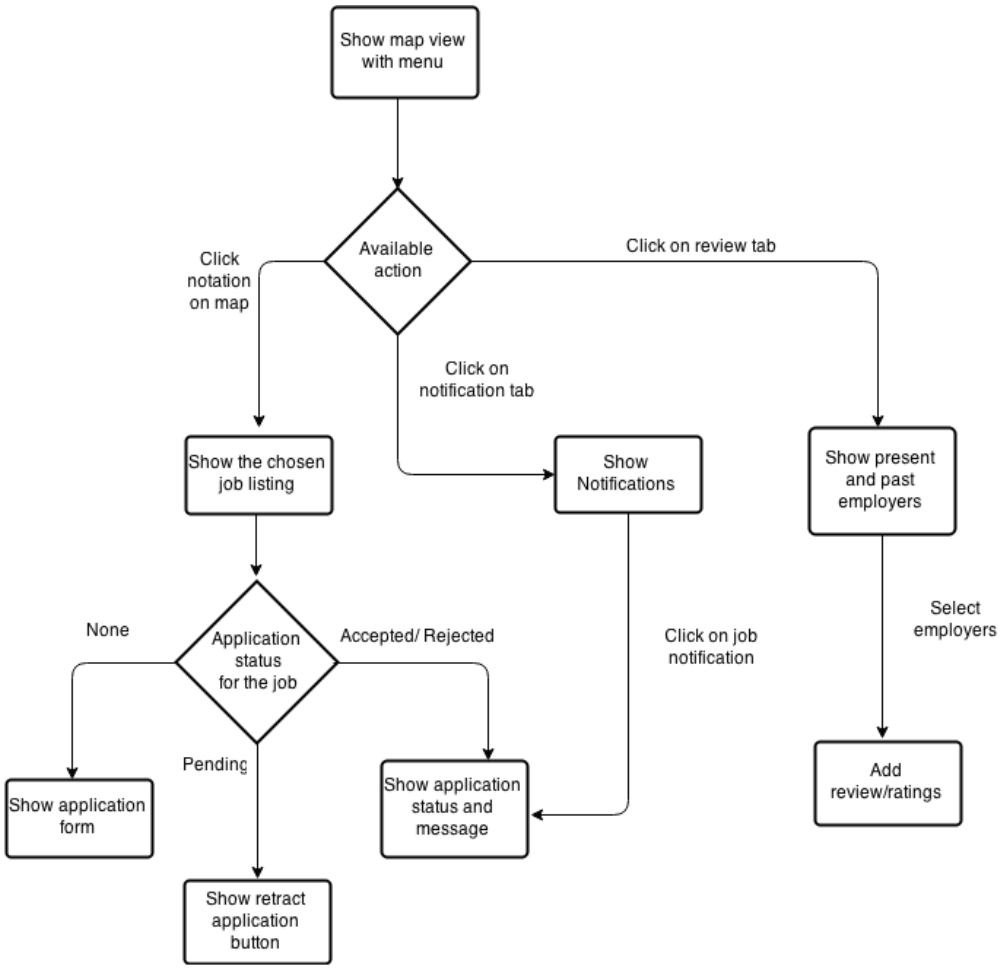

Figure 4-2 : System diagram flow for mobile application (part 2)

4.1.2 Web Application

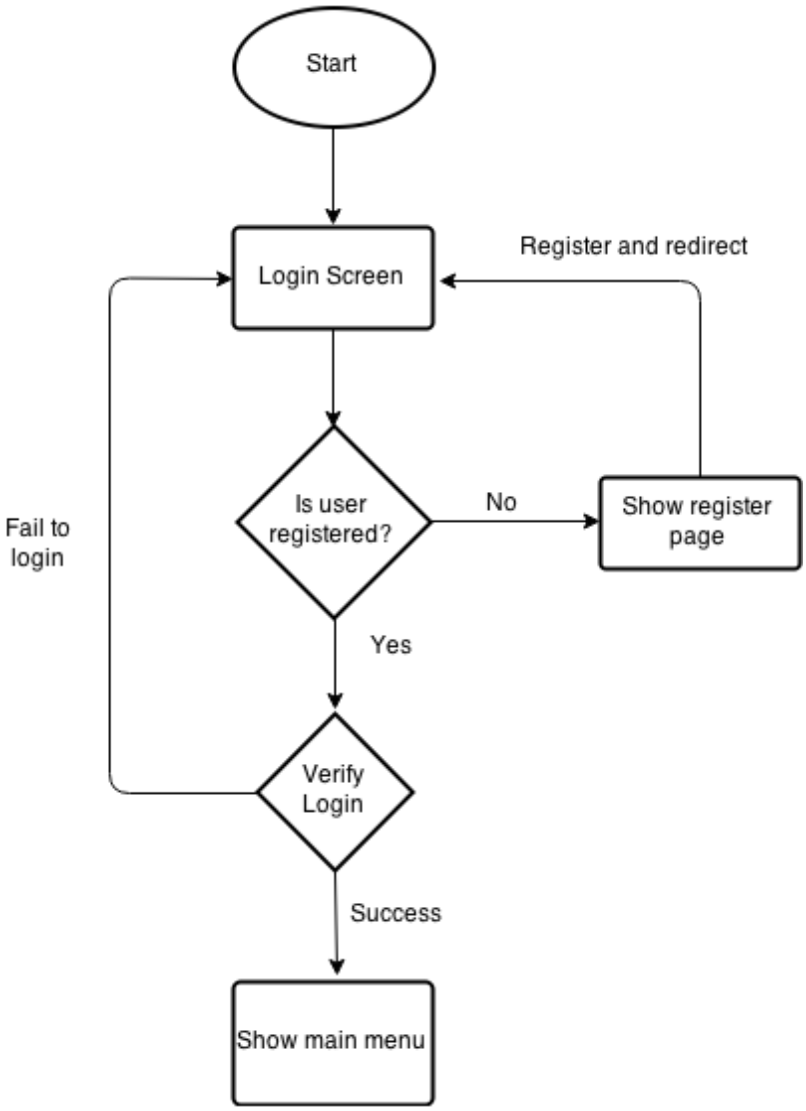

Figure 4-3 : System diagram flow for web application (part 1)

Figure 4-4 : System diagram flow for web application (part 2)

4.2 Use Case Diagram

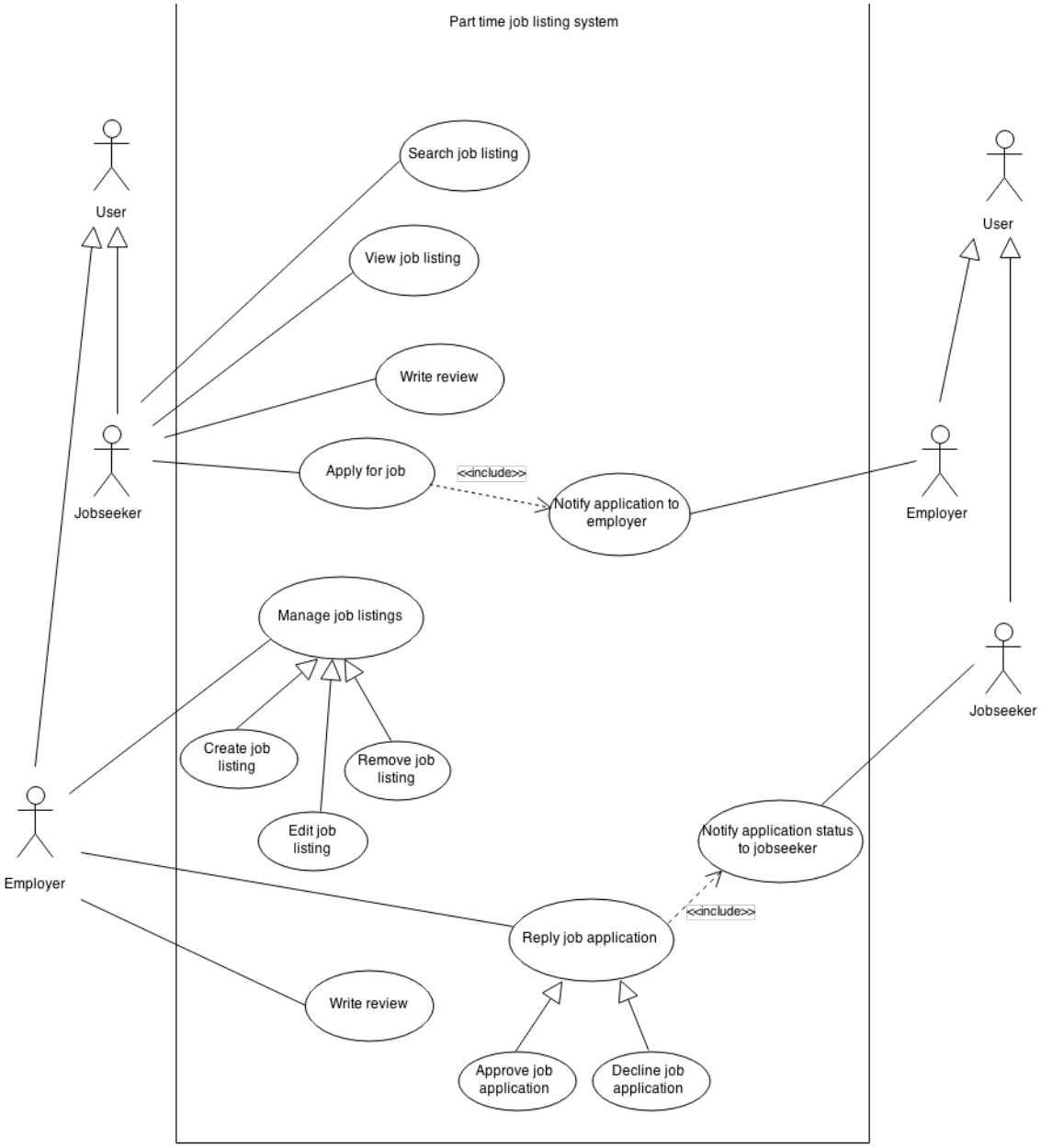

Figure 4-5 : Use-case diagram of part time job listing system

4.3 Use Case Specification

Table 4-1 : Search job listing use case specification

Use Case Name: Search job listing	ID: 1	Importance Level: High
Primary Actor: Jobseeker	Use Case Type: Detail, Essential	
Stakeholders and Interests: Jobseeker - wants to search for available job vacancy		
Brief description: This use case describe how jobseeker search for job vacancy using listing		
Trigger: Job seeker wants to search for available job vacancy Type: External		
Relationships: Association: Jobseeker Include: - Extend: - Generalization: -		
Normal Flow of Events: 1. Jobseeker navigate to job search tab 2. Jobseeker enter desired search criteria 3. System searches database for job vacancy that matches the criteria inputted 4. System show the job vacancy search result		
Sub Flows: Not applicable		
Alternate / Exceptional Flows: Not applicable		

Table 4-2 : View job listing use case specification

Use Case Name: View job listing	ID: 2	Importance Level: High
Primary Actor: Jobseeker	Use Case Type: Detail, Essential	
Stakeholders and Interests: Jobseeker - wants to view job vacancy information		
Brief description: This use case describe how jobseeker view job vacancy information		
Trigger: Jobseeker wants to view job vacancy information Type: External		
Relationships: Association: Jobseeker Include: - Extend: - Generalization: -		
Normal Flow of Events: 1. Jobseeker navigate to job listing 2. Jobseeker select a specific job vacancy 3. System show the job vacancy details		
Sub Flows: Not applicable		
Alternate / Exceptional Flows: Not applicable		

Table 4-3 : Write review use case specification

Use Case Name: Write review	ID: 3	Importance Level: High
Primary Actor: Jobseeker	Use Case Type: Detail, Essential	
Stakeholders and Interests: Jobseeker - wants to write review for an employer		
Brief description: This use case describe how jobseeker write review for an employer		
Trigger: Jobseeker wants to write a review for an employer Type: External		
Relationships: Association: Jobseeker Include: - Extend: - Generalization: -		
Normal Flow of Events: 1. Jobseeker navigate to review tab 2. System show list of employers which the jobseeker has worked with before 3. Jobseeker select a employer from the list 4. Jobseeker enter review information and submit to the system 5. System saves the review information into database		
Sub Flows: Not applicable		
Alternate / Exceptional Flows: Not applicable		

Table 4-4 : Apply for job use case specification

Use Case Name: Apply for job	ID: 4	Importance Level: High
Primary Actor: Jobseeker	Use Case Type: Detail, Essential	
Stakeholders and Interests: Jobseeker - wants to apply for a job		
Brief description: This use case describe how jobseeker apply for a job vacancy		
Trigger: Jobseeker wants to apply for a job Type: External		
Relationships: Association: Jobseeker Include: - Notify application to employer Extend: - Generalization: -		
Normal Flow of Events: 1. Jobseeker navigate to job listing 2. Jobseeker select a specific job vacancy 3. System show the job vacancy details 4. Jobseeker click apply 5. System save the job application data to database		
Sub Flows: Not applicable		
Alternate / Exceptional Flows: Not applicable		

Table 4-5 : Manage job listings use case specification

Use Case Name: Manage job listings	ID: 5	Importance Level: High
Primary Actor: Employer	Use Case Type: Detail, Essential	
Stakeholders and Interests: Employer - wants to manage his/her job vacancy information		
Brief description: This use case describe how employer can manage his/her job vacancy information		
Trigger: Employer wants to manage his/her job vacancy information Type: External		
Relationships: Association: Employer Include: - Extend: - Generalization: - Create job listing, Edit job listing, Remove job listing		
Normal Flow of Events: 1. Employer navigate to job listing page 2. If employer wants to add new job listing The S - 1 : Create job listing is performed 3. If employer wants to edit job listing The S - 2 : Edit job listing is performed 4. If employer wants to delete job listing The S - 3 : Delete job listing is performed		
Sub Flows: S - 1 : Create job listing 1. Employer enter job vacancy information 2. System save the job vacancy information into database S - 2 : Edit job listing 1. Employer navigate to specific job vacancy record 2. Employer change the job vacancy information 3. System update the job vacancy information S - 3 : Remove job listing 1. Employer navigate to specific job vacancy record 2. Employer delete the job vacancy record 3. System the job vacancy record from the database		
Alternate / Exceptional Flows: Not applicable		

Table 4-6 : Reply job application use case specification

Use Case Name: Reply job application	ID: 6	Importance Level: High
Primary Actor: Employer	Use Case Type: Detail, Essential	
Stakeholders and Interests: Employer - wants to reply to a job application		
Brief description: This use case describe how employer reply to a job application		
Trigger: Employer wants to reply to a job application Type: External		
Relationships: Association: Employer Include: - Notify application status to jobseeker Extend: - Generalization: - Approve job application, Decline job application		
Normal Flow of Events: 1. Employer navigate to job application page 2. Employer select a job application record 3. If employer wants to approve a job application The S -1 : Approve job application is performed 4. If employer wants to decline a job application The S - 2 : Decline job application is performed		
Sub Flows: S - 1 : Approve job application 1. Employer click “Accept” button on the job application record 2. System update the job application information S - 2 : Decline job application 1. Employer click “Reject” button on the job application record 2. System update the job application information		
Alternate / Exceptional Flows: Not applicable		

Table 4-7 : Write review (employer) use case specification

Use Case Name: Write review (employer)	ID: 7	Importance Level: High
Primary Actor: Employer	Use Case Type: Detail, Essential	
Stakeholders and Interests: Employer - wants to write review for an (ex) employee		
Brief description: This use case describe how employer write review for an (ex) employee		
Trigger: Employer wants to write a review for an (ex) employee Type: External		
Relationships: Association: Employer Include: - Extend: - Generalization: -		
Normal Flow of Events: 1. Employer navigate to review tab 2. Employer select “Write Review” 3. Employer enter employee ID 4. Employer enter review information and submit to the system 5. System saves the review information into database		
Sub Flows: Not applicable		
Alternate / Exceptional Flows: Not applicable		

4.4 Entity Relationship Diagram

Entity Relationship Diagram (ERD) is used to model objects, relationships and cardinality between different objects.

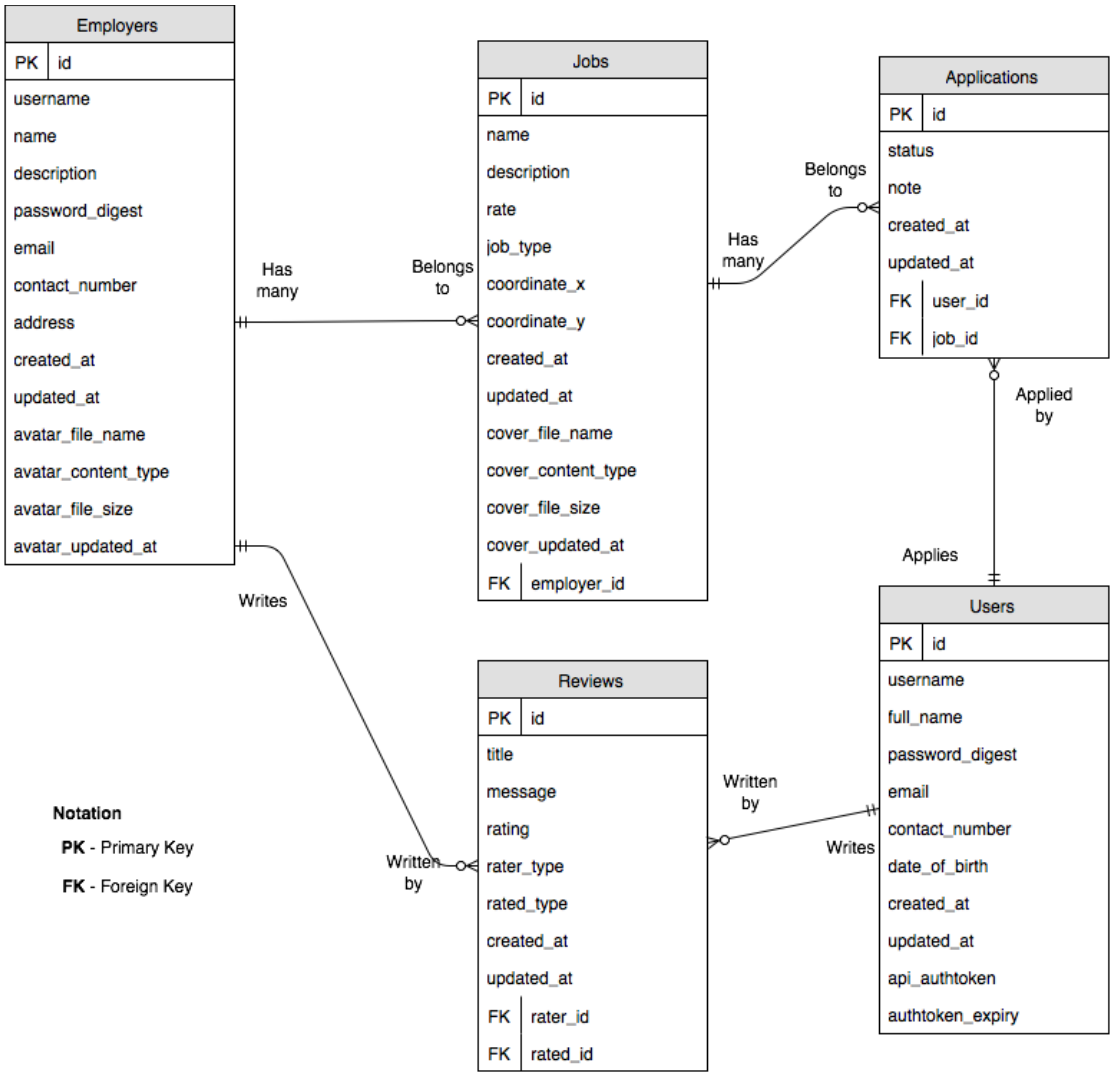

Figure 4-6 : Entity Relationship Diagram of the system

4.5 Data Dictionary

Data dictionary is used to describe objects' attributes such as its usage, storage type and constraints. All attribute in every table is required except stated otherwise, all primary key in every table is auto incremented on creation of new rows.

Table 4-8 : Data Dictionary of Employers table

Attribute name	Contents	Type	Constraints
id	Employer ID	Int(11)	Primary Key
username	Employer's username	Varchar(32)	
name	Employer's business name	Varchar(120)	
description	Employer's business description	Text	
password_digest	Hash digest of employer password	Char(60)	
email	Employer's email	Varchar(70)	
contact_number	Employer's contact number	Varchar(14)	
address	Employer's business address	Text	
created_at	Creation date of the employer record	Datetime	
updated_at	Date of latest update performed on the record	Datetime	
avatar_file_name	Image file name for the employer's avatar	Varchar(255)	Optional
avatar_content_type	Image file type for the employer's avatar	Varchar(255)	Optional
avatar_file_size	Image file size for the employer's avatar	Varchar(255)	Optional
avatar_updated_at	Date of latest update on the employer's avatar image	Varchar(255)	Optional

Table 4-9 : Data Dictionary of Jobs table

Attribute name	Contents	Type	Constraints
id	Job ID	Int(11)	Primary Key
name	Job name	Varchar(50)	
description	Job description	Text	
rate	Rate per hour or total amount paid for the job	Decimal(5,2)	
job_type	Type of the job	Enum('parttime', 'onetime')	
coordinate_x	Longitude of the location of the job	Decimal(15,10)	
coordinate_y	Latitude of the location of the job	Decimal(15,10)	
created_at	Creation date of the job record	Datetime	
updated_at	Date of latest update performed on the record	Datetime	
cover_file_name	Image file name for the job cover picture	Varchar(255)	Optional
cover_content_type	Image file type for the job cover picture	Varchar(255)	Optional
cover_file_size	Image file size for the job cover picture	Varchar(255)	Optional
cover_updated_at	Date of latest update on the job cover image	Varchar(255)	Optional
employer_id	ID of employer which this job belongs to	Int(11)	Foreign Key references table *Employers*(id)

Table 4-10 : Data Dictionary of Applications table

Attribute name	Contents	Type	Constraints
id	Application ID	Int(11)	Primary Key
status	Status of the application	Enum(‘pending’, ‘accepted’, ‘rejected’)	
note	Additional notes on the application	Text	Optional
created_at	Creation date of the application record	Datetime	
updated_at	Date of latest update performed on the record	Datetime	
user_id	ID of the user who applied for this job application	Int(11)	Foreign Key references table *Users*(id)
job_id	ID of the job which this application is for	Int(11)	Foreign Key references table *Jobs*(id)

Table 4-11 : Data Dictionary of Reviews table

Attribute name	Contents	Type	Constraints
id	Review ID	Int(11)	Primary Key
title	Title of the review	Varchar(255)	
message	Content of the review	Text	
rating	Rating score given in the review	Int(11)	
rater_type	Class type of the author of the review	Varchar(255)	
rated_type	Class type of the individual being reviewed	Varchar(255)	
created_at	Creation date of the review record	Datetime	
updated_at	Date of latest update performed on the record	Datetime	
rater_id	ID of the author of this review	Int(11)	Foreign Key references table *Users*(id) / Foreign Key references table *Employers*(id)
rated_id	ID of the individual being reviewed	Int(11)	Foreign Key references table *Users*(id) / Foreign Key references table *Employers*(id)

Table 4-12 : Data Dictionary of Users table

Attribute name	Contents	Type	Constraints
id	User ID	Int(11)	Primary Key
username	User's username	Varchar(32)	
full_name	User's full name	Varchar(120)	
password_digest	Hash digest of user password	Char(60)	
email	User's email	Varchar(70)	
contact_number	User's contact number	Varchar(14)	
date_of_birth	User's date of birth	Text	
created_at	Creation date of the user record	Datetime	
updated_at	Date of latest update performed on the record	Datetime	
api_authtoken	Authorisation token string for the usage of API by the mobile application	Varchar(255)	Optional
authtoken_expiry	Expiration date of the authorisation token	Datetime	Optional

4.6 System Architecture Overview

An overview of the system architecture and the relationship between each component is shown in the below diagram.

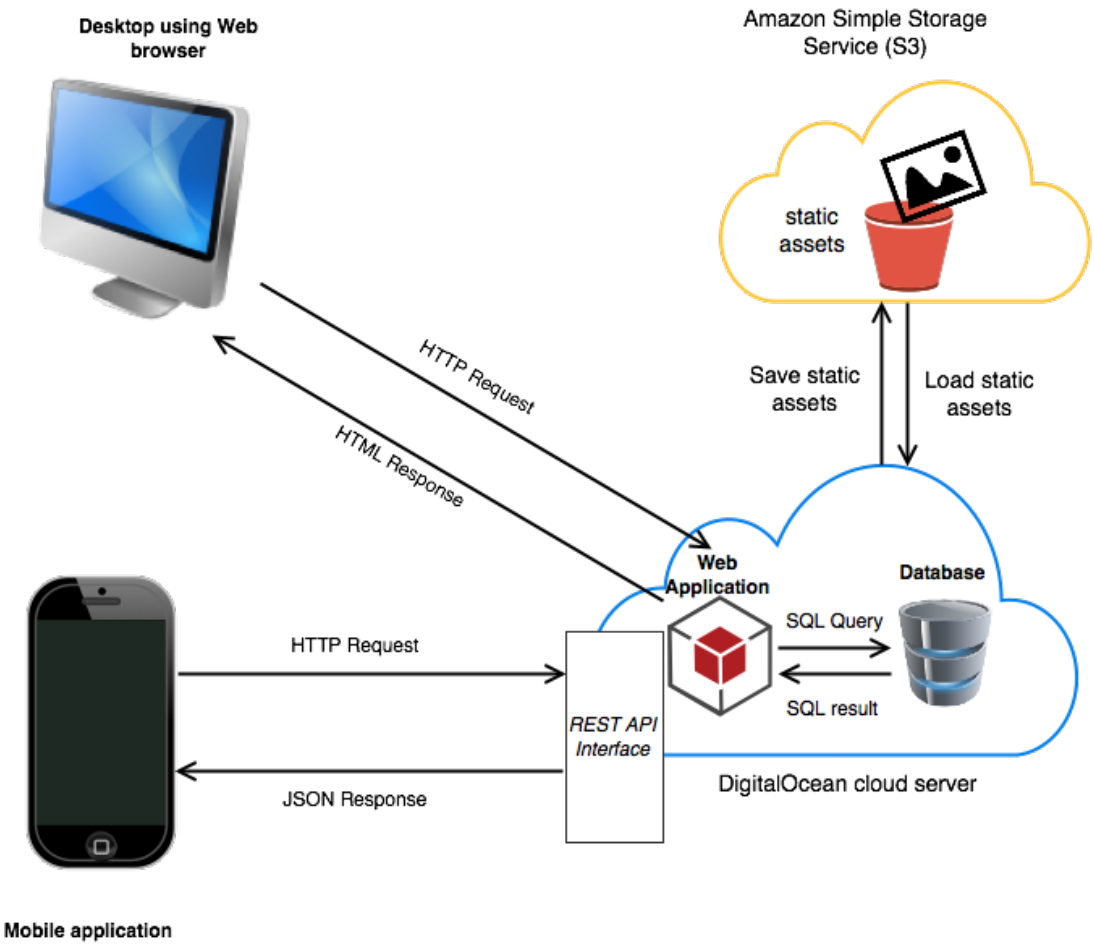

Figure 4-7 : System architecture diagram

Employer uses web browser to access the web application whereas jobseeker uses the mobile application to communicate with the web application via REST Application Programming Interface (API). Web application query data from database using Structured Query Language (SQL) and result is returned using the same format. Static assets such as user-uploaded image is stored in Amazon Simple Storage Service (S3), a scalable cloud storage service provided by Amazon. The resource identifier of the uploaded image is in this format : "sidegig/*instance-class*/*instance-id*/*instance name*_original.*filetype*" to prevent collision of resource identifier. A thumbnail will be auto generated after the upload of the original image with the following format : "sidegig/*instance-class*/*instance-id*/*instance name*_thumbnail.png". For example, a cover photo uploaded for *job* instance with name "mcd" and ID "10" will have two resource identifiers : "sidegig/job/10/mcd_original.png" and "sidegig/job/10/mcd_thumbnail.png".

Chapter 5 : Implementation and Testing

5.1 Graphical User Interface

5.1.1 Mobile Application Graphical User Interface

When user first open the iOS mobile application, a login screen will be shown. User can choose to register a new account or login with existing account.

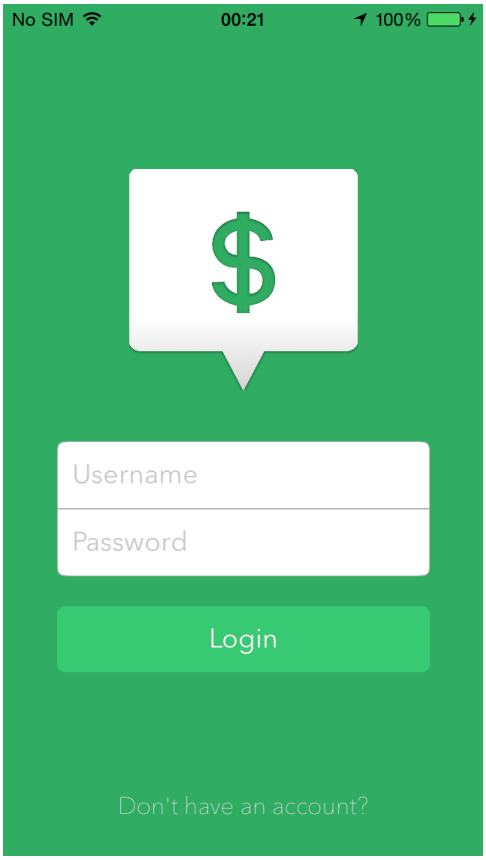

Figure 5-1 : Login view of mobile application

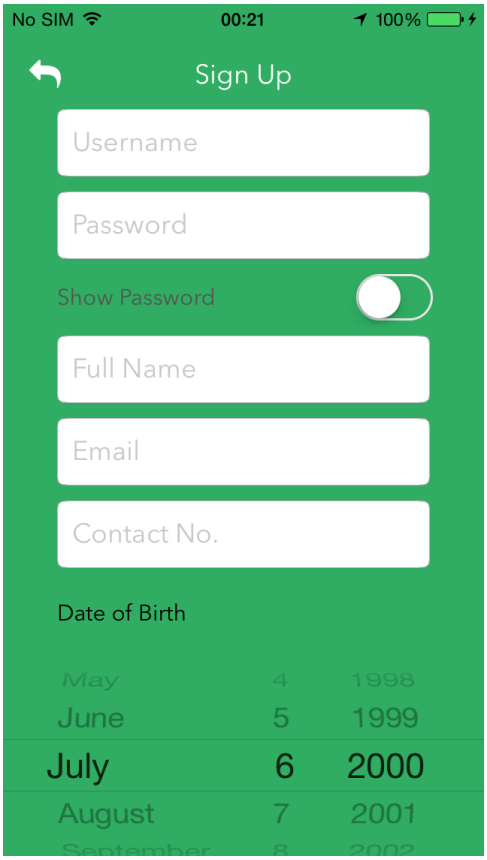

Figure 5-2 : Sign Up view of mobile application

After successfully logged in, user will be redirected to map view. User can navigate around the map to view available job vacancy or search for specific job using the search bar at the top. User can tap on the marker on the map and a popover showing the shop name will appear. User can tap the marker to view the job vacancy details. Other than that, there will be four tabs at the bottom including Map, Notification, Review and Profile. User can tap into one of the tab to view each respective view.

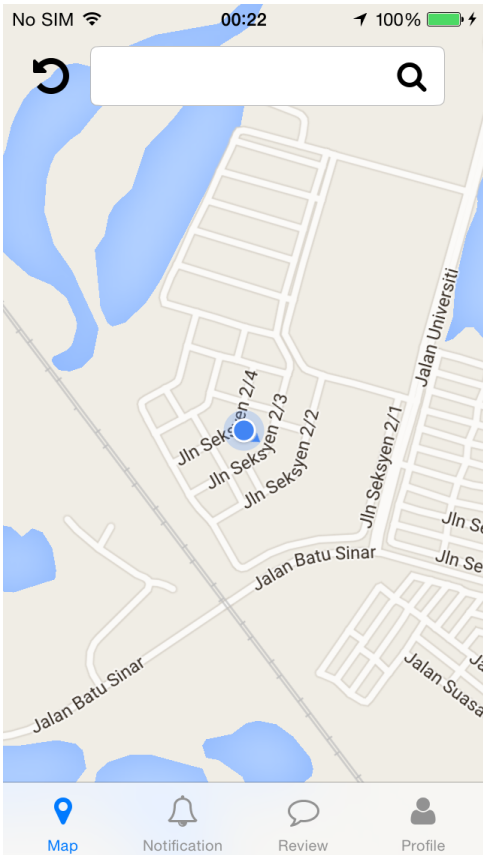

Figure 5-3 : Map view of mobile application

Figure 5-4 : Popover on top of marker on map view

In the job vacancy detail view, user can check the review about the employer or apply for the job vacancy.

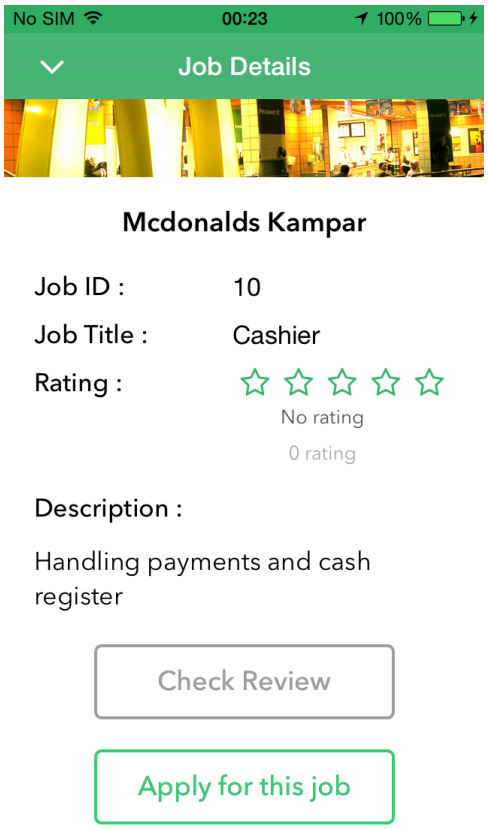

Figure 5-5 : *Job detail* view of mobile application

Figure 5-6 : *Job review* view of mobile application

In the search view, user can enter the search criteria and choose between the nearest option or the highest pay option and the result will be sorted accordingly. User will be redirected to the job vacancy detail page after he/she tap on the job vacancy record on the result list.

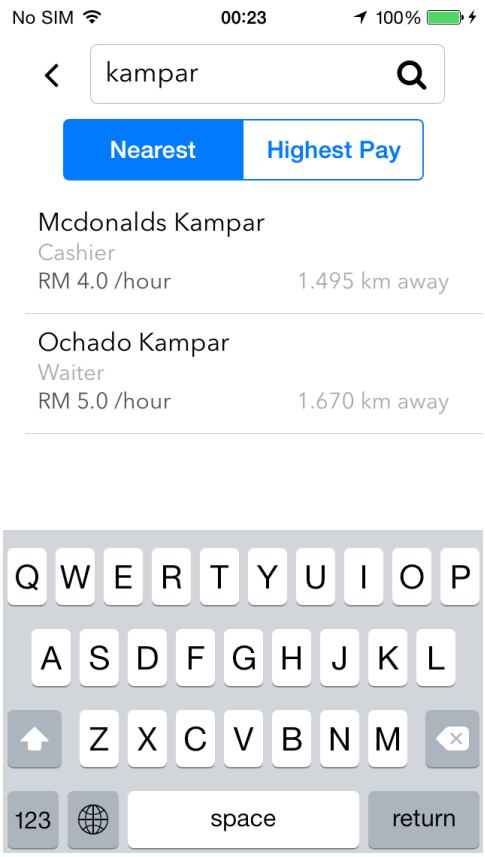

Figure 5-7 : Search view of mobile application with nearest option selected

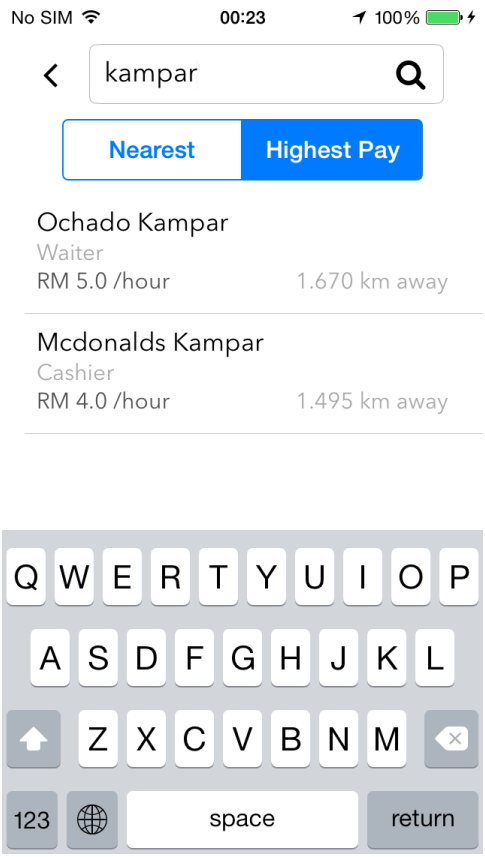

Figure 5-8 : Search view of mobile application with highest pay option selected

In the notification tab, user can check for the application status of the job vacancy. In the review tab, user can select a previously worked job and write a review to the employer of the selected job.

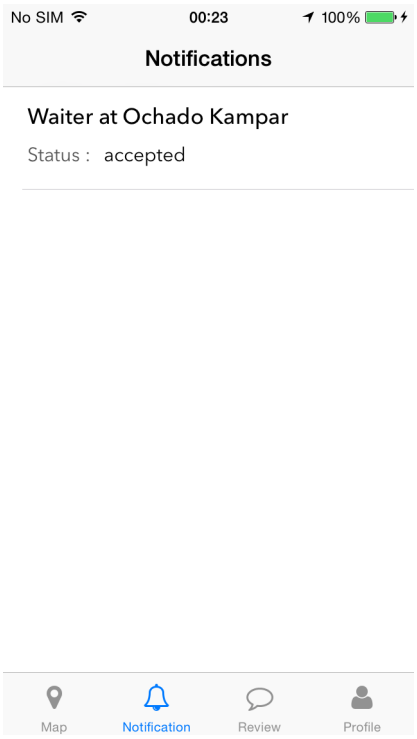

Figure 5-9 : Notification view of mobile application

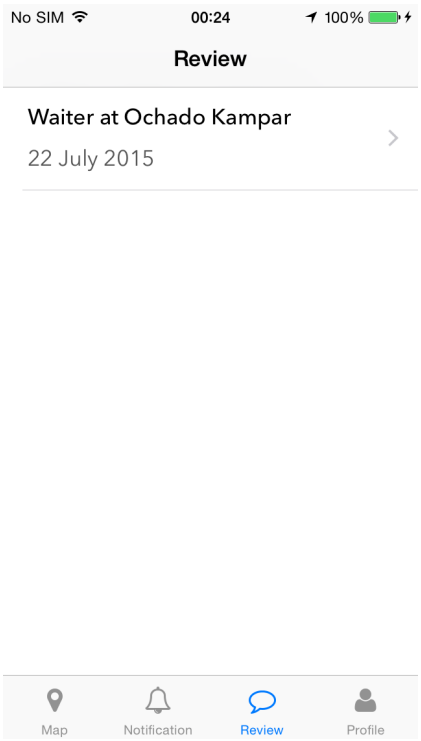

Figure 5-10 : Review view of mobile application

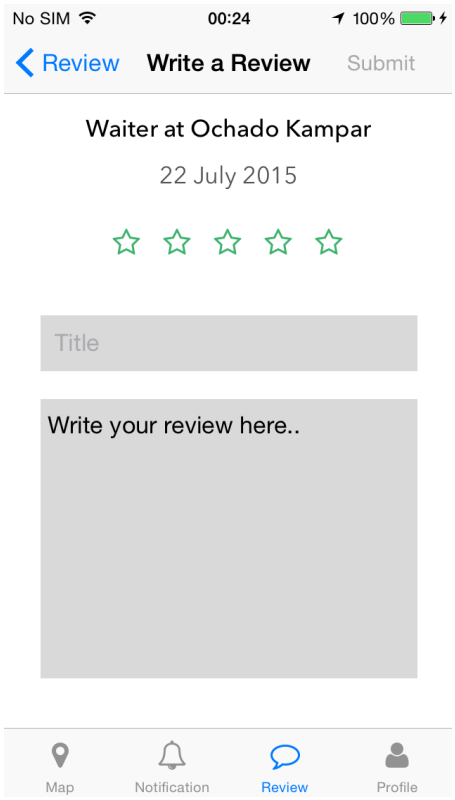

Figure 5-11 : *Write review* subview of mobile application

In the profile tab, user can view and change his/her personal details. User can also check employer reviews about him/her. Other than that, user can also view his/her previous job history.

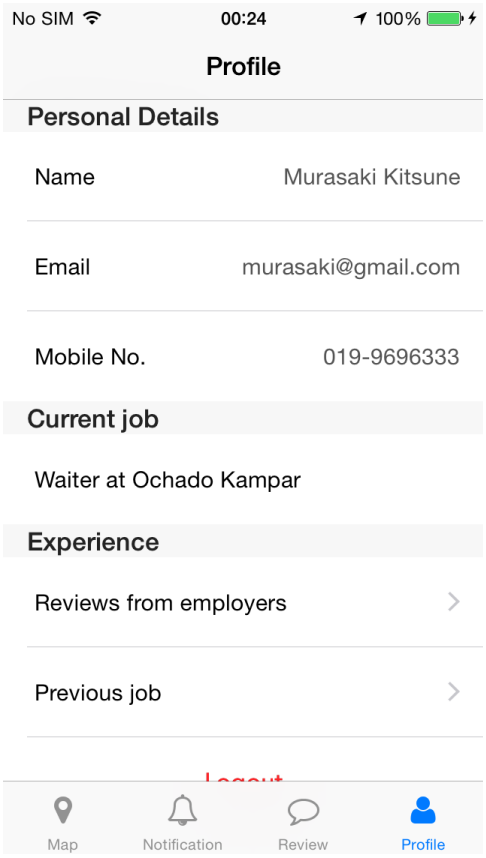

Figure 5-12 : Profile view of mobile application

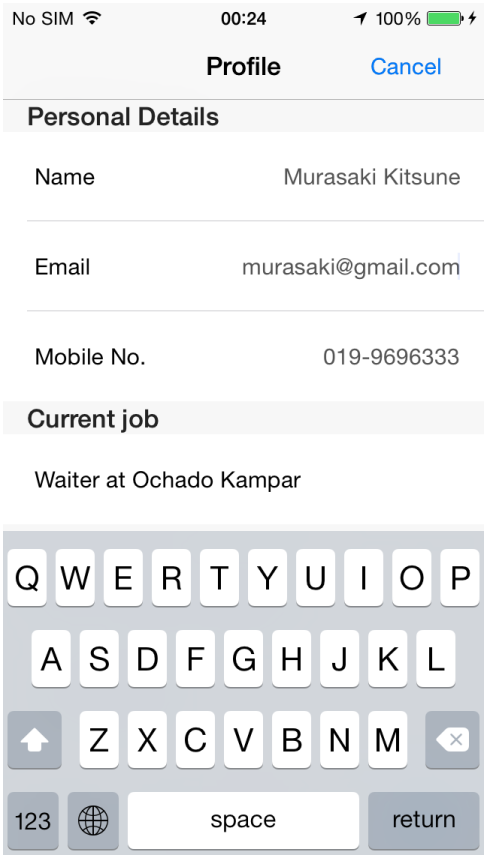

Figure 5-13 : Profile view of mobile application with keyboard layout

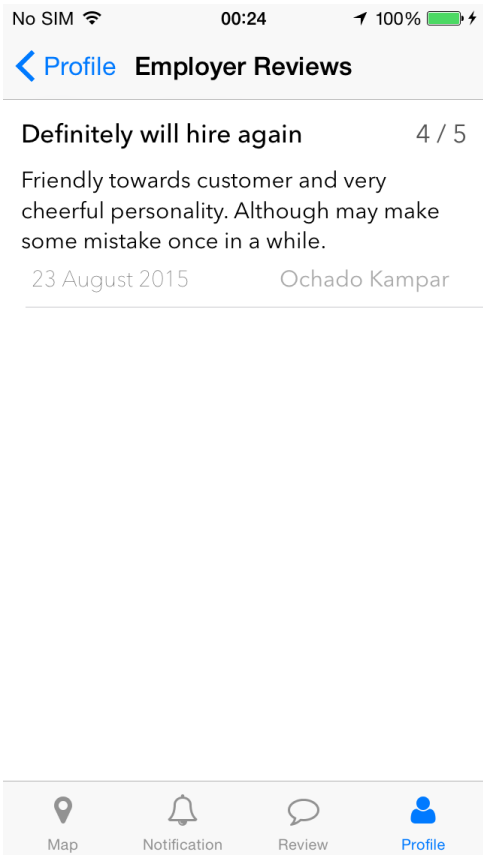

Figure 5-14 : *Employer reviews* subview of mobile application

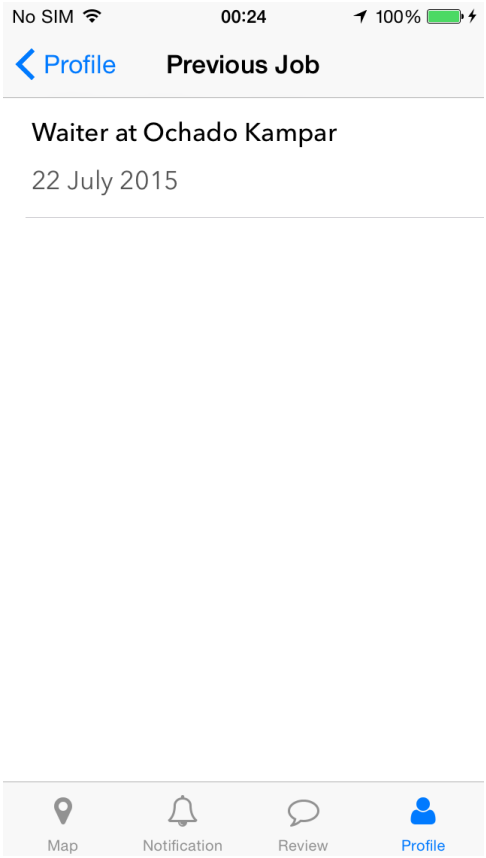

Figure 5-15 : *Previous job* subview of mobile application

5.1.2 Web Application Graphical User Interface

Twitter Bootstrap is used as front-end design framework for web application as it provide several preset responsive feature and design of various HTML components.

On the home page of web application, user can choose to login or sign up.

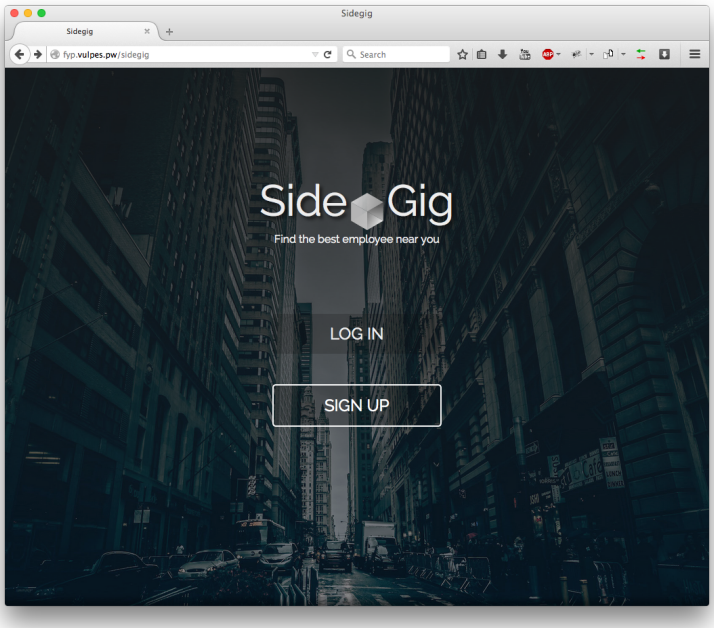

Figure 5-16 : Home page of web application

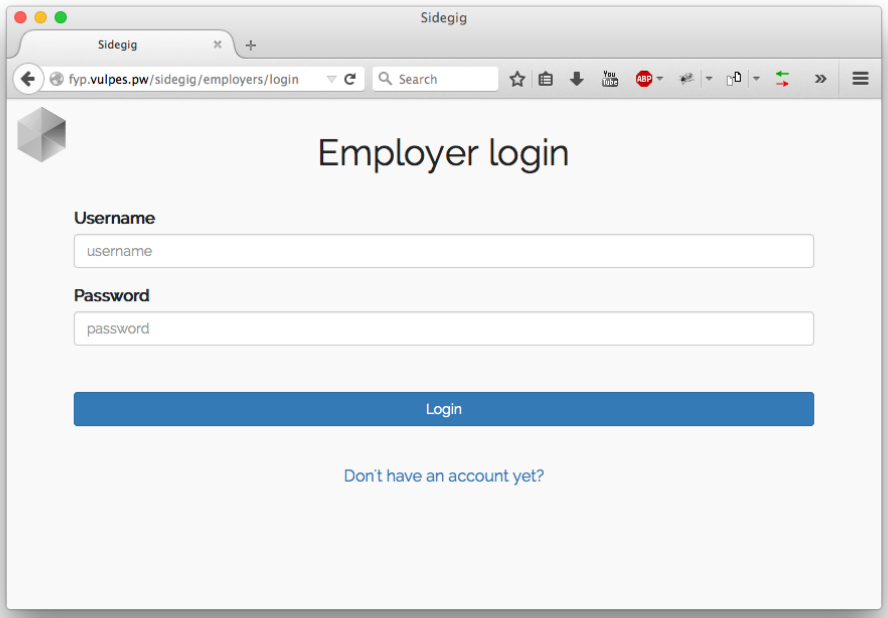

Figure 5-17 : Login page of web application

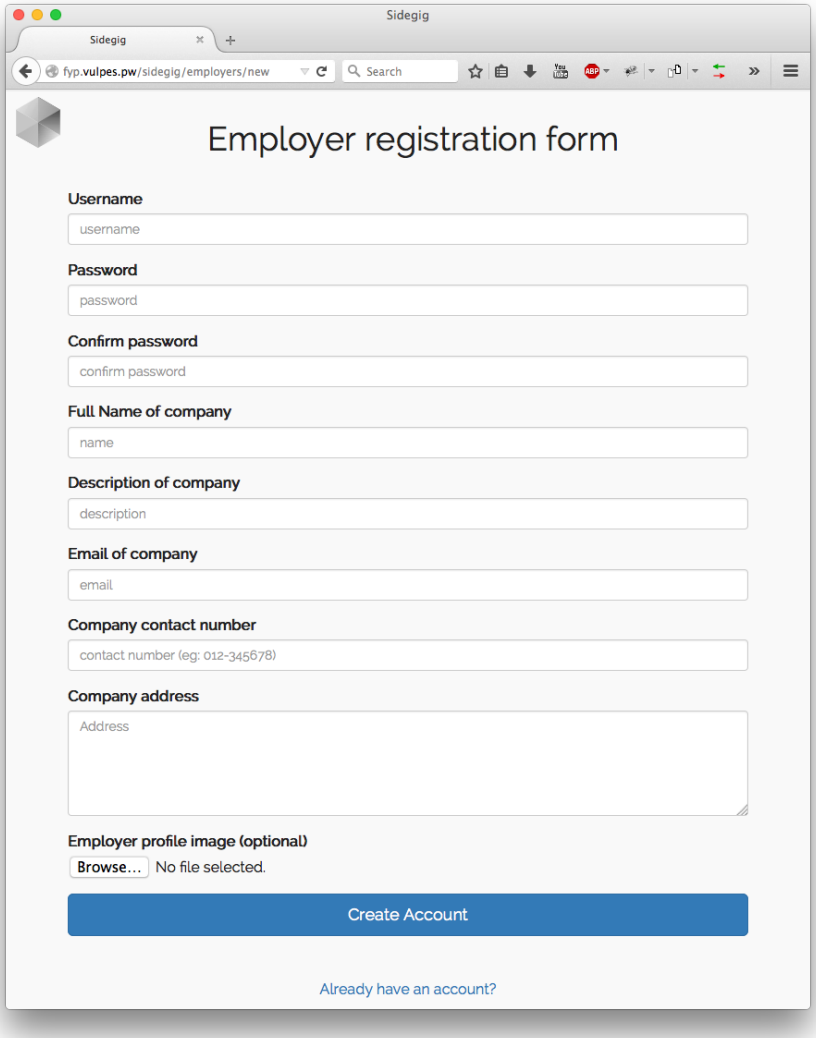

Figure 5-18 : Register page of web application

After successfully logged in, user will be directed to the main page of control panel. User can select action from the drop down menu. There are four tabs on the menu showing “Profile”, “Jobs”, “Application”, “Review”.

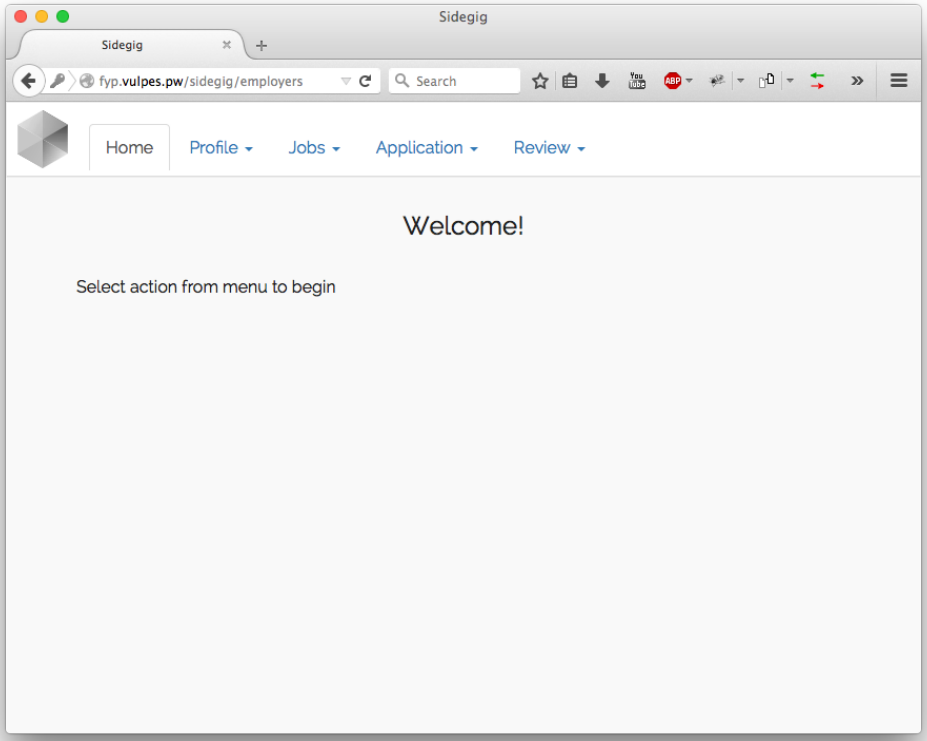

Figure 5-19 : Control panel home page of web application

There are three action in the “Profile” tab, user can choose to view profile, edit profile or logout.

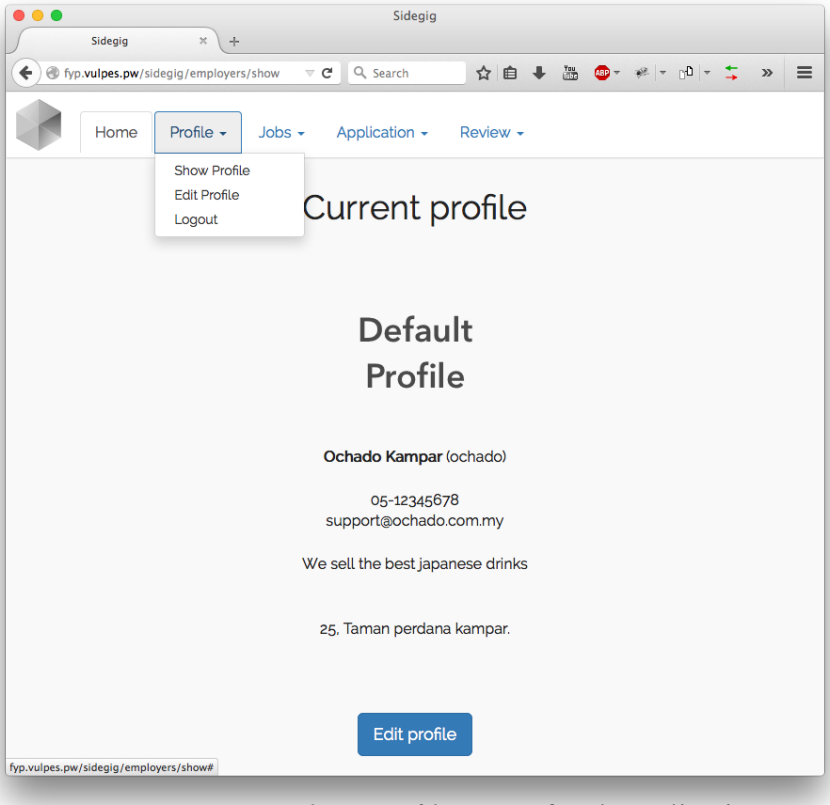

Figure 5-20 : *Show profile* page of web application

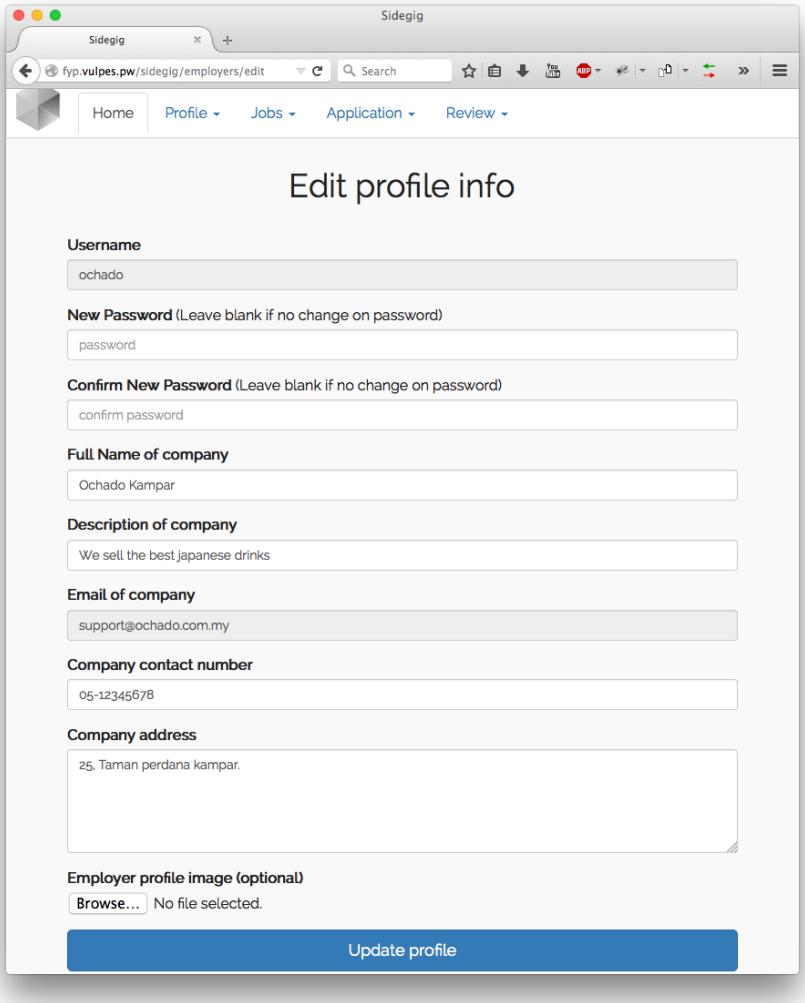

Figure 5-21 : *Edit profile* page of web application

In the “Jobs” tab, user can choose to list previously created job vacancy or create a new job vacancy.

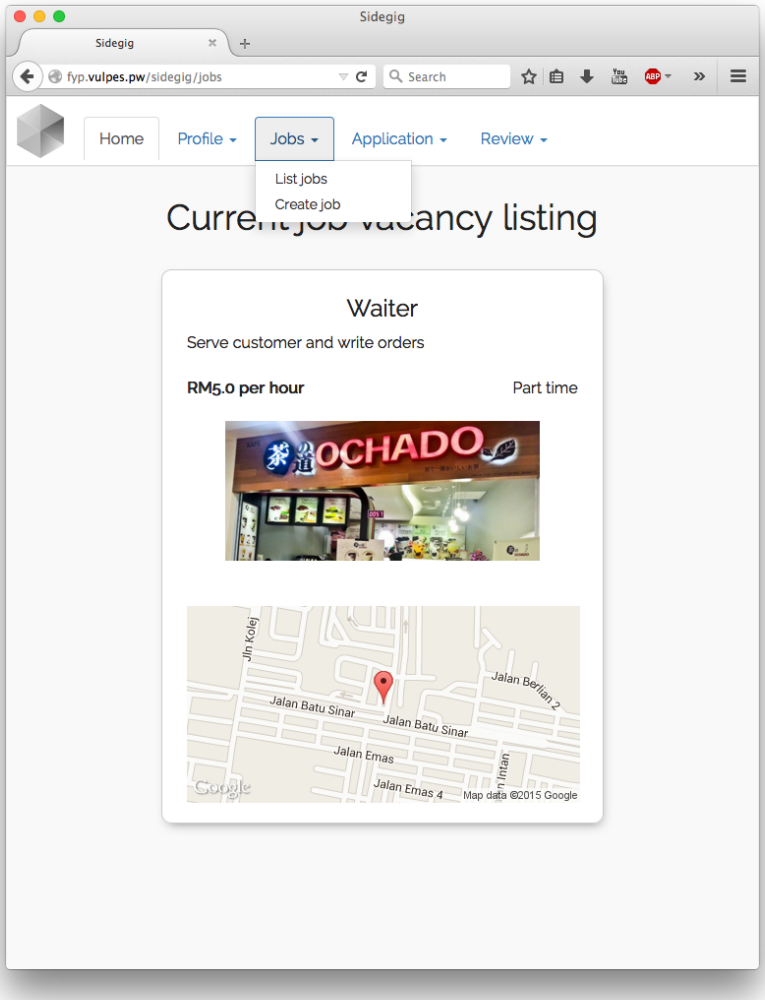

Figure 5-22 : *List jobs* page of web application

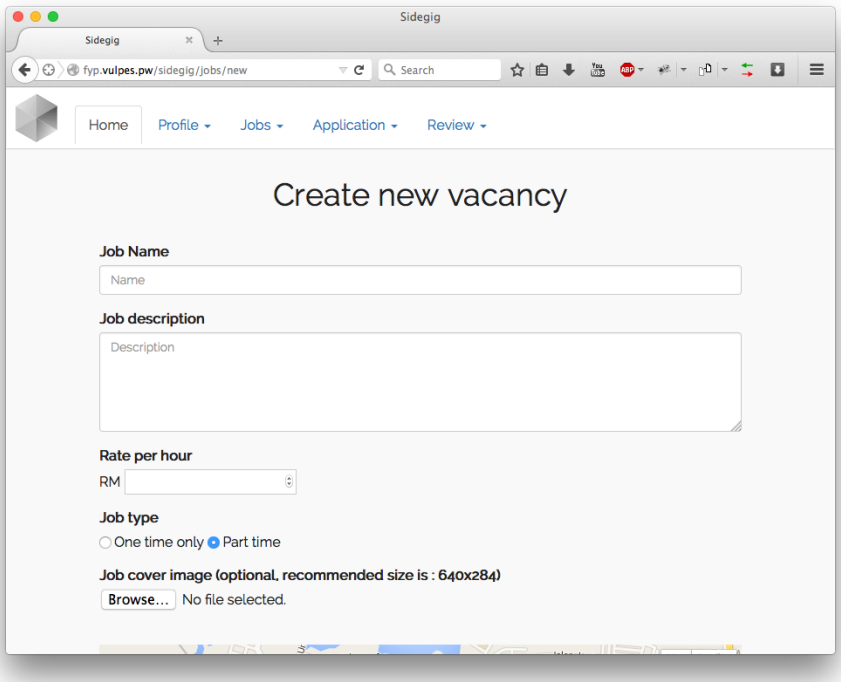

Figure 5-23 : *Create job* page of web application

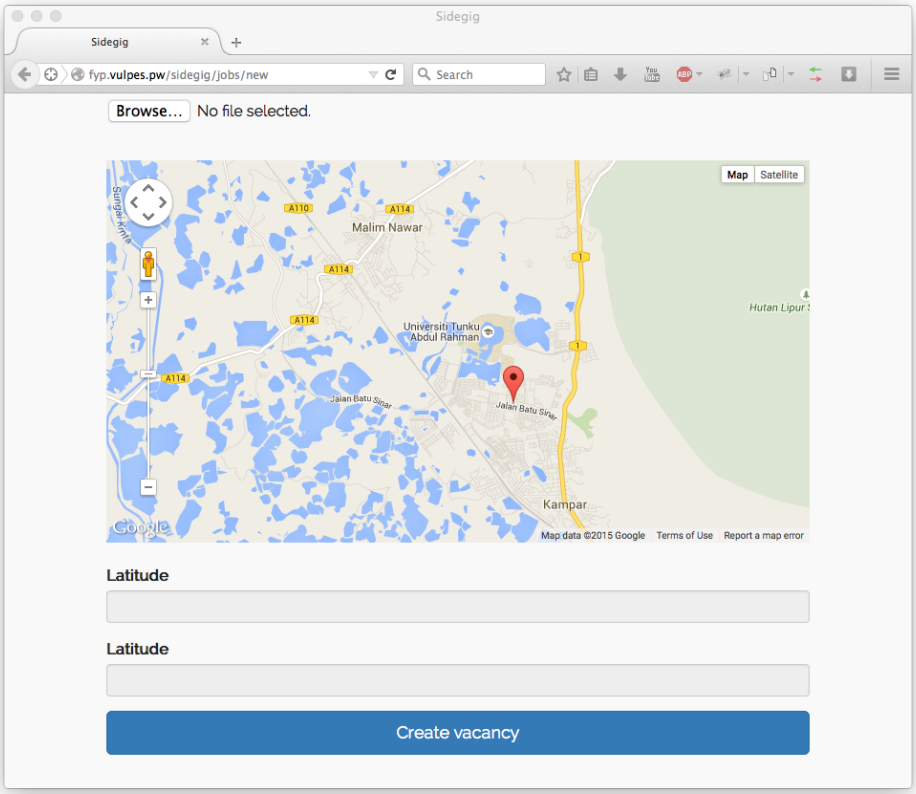

Figure 5-24 : *Create job* page of web application (cont.)

In the "Applications" tab, user can choose to view list of job applications and decide to accept or reject each job application.

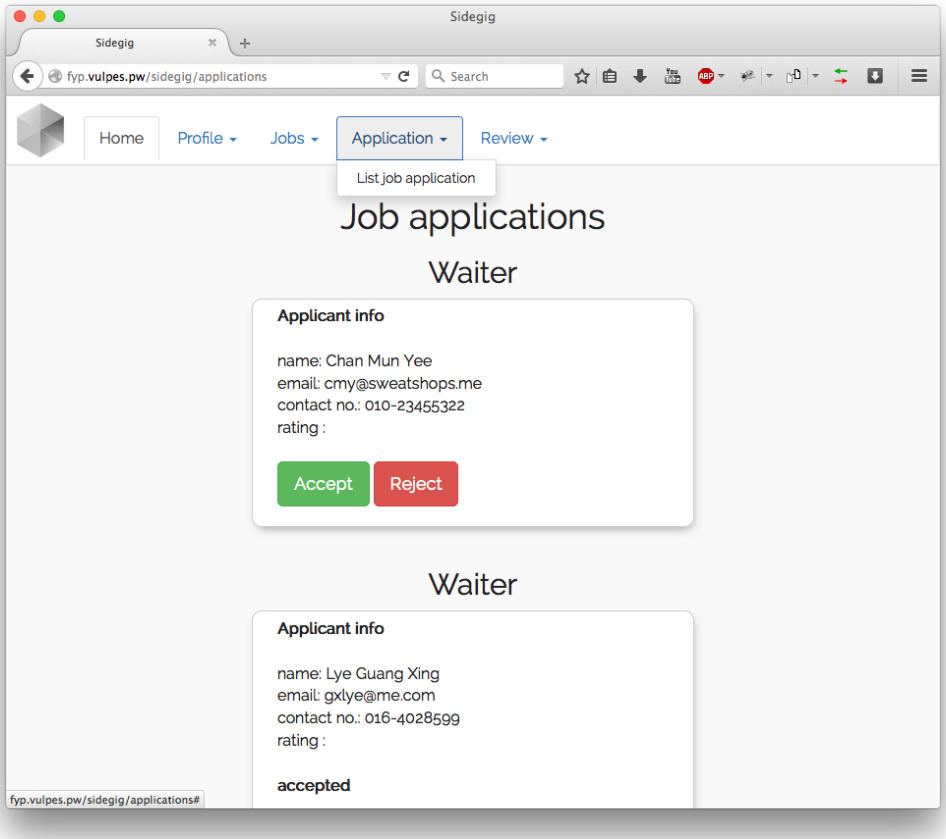

Figure 5-25 : *List job application* page of web application

In the “Review” tab, user can choose to view previously written review, his/her received review from others or write a new review.

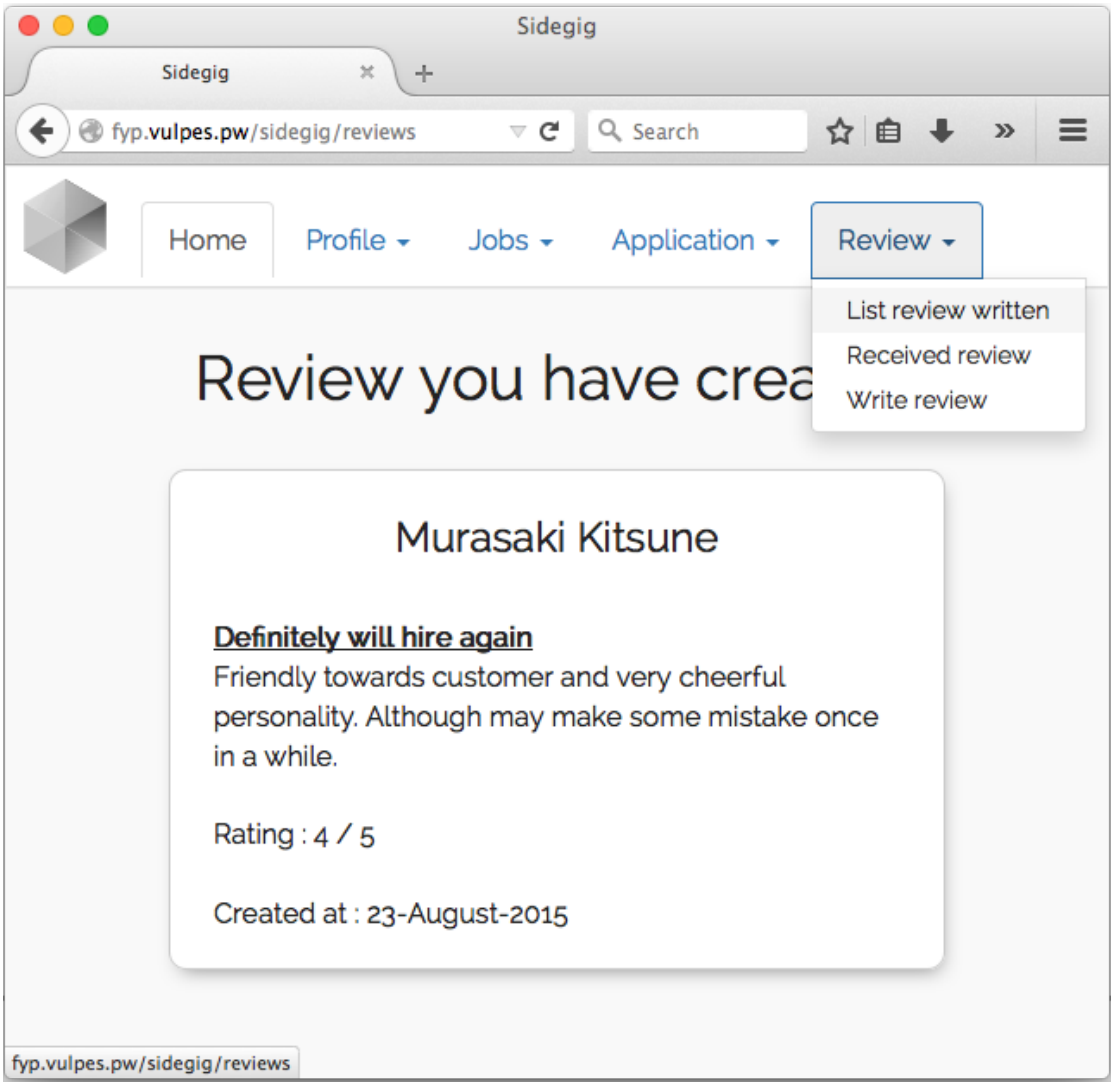

Figure 5-26 : *List review written* page of web application

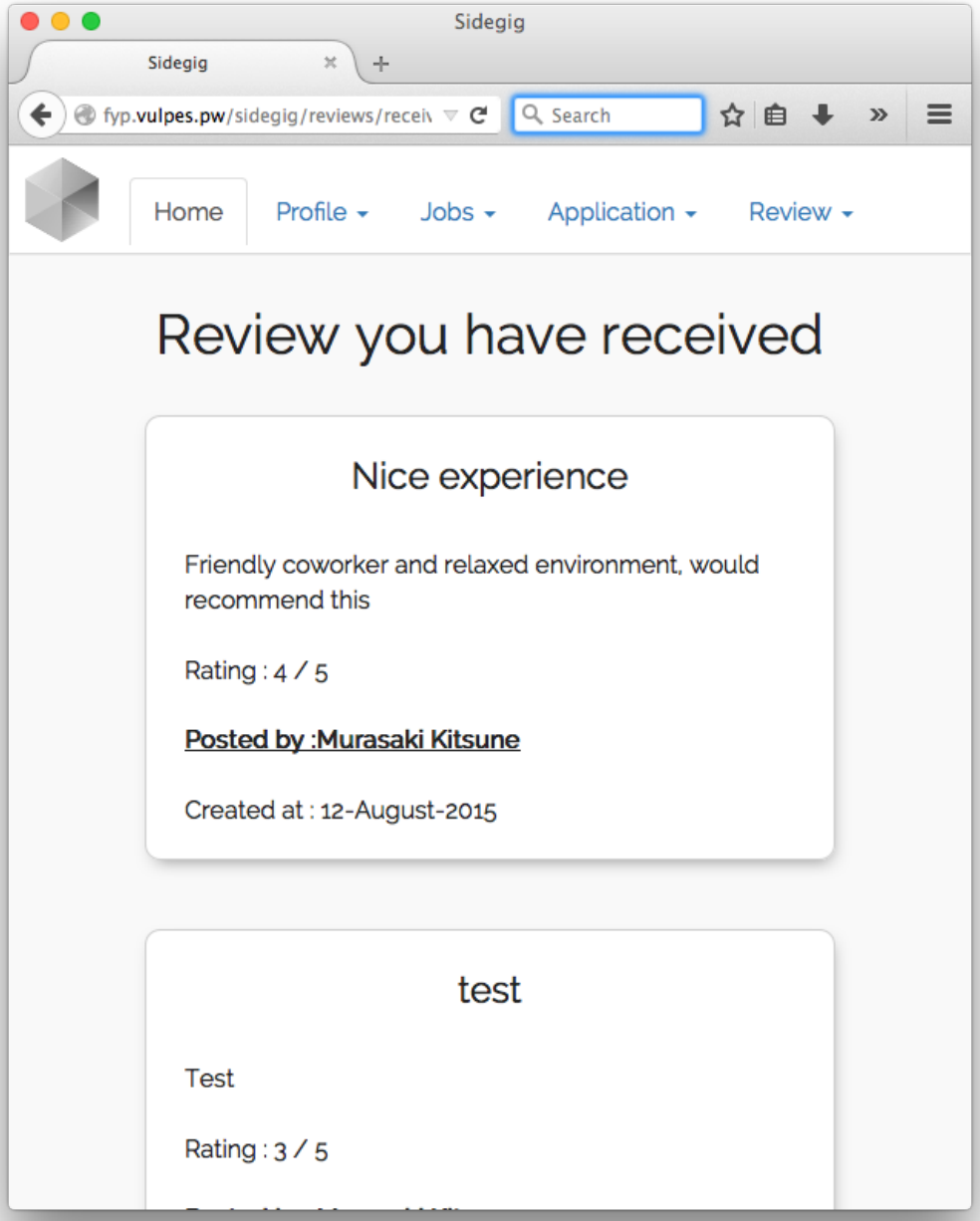

Figure 5-27 : *Received review* page of web application

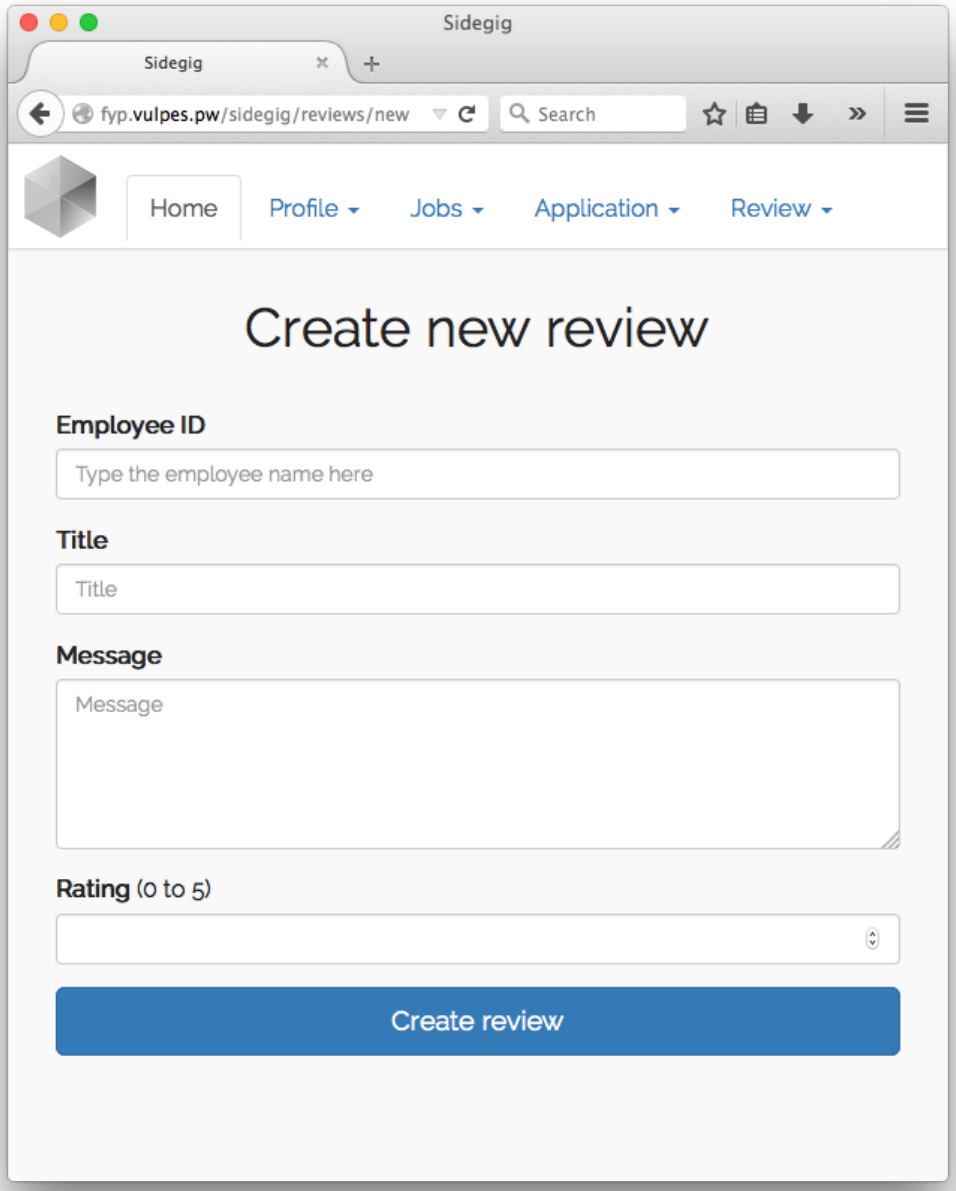

Figure 5-28 : *Create review* page of web application

5.2 Testing

Black-box testing is performed on the system to compare the output by the system with the expected result. Black box testing does not require tester to have prior knowledge of the internal workings of system and it emphasise on the outputs generated by the system triggered by selected inputs and environment conditions. All the testing is done under assumption that internet connection and location services are available to the testing devices.

5.2.1 Mobile application

5.2.1.1 Login Module

Login Module			
Case	**Test Action**	**Test Result**	**Status**
1	Tap on *login* button without filling username or password	Popup alert is shown as warning that mandatory field must be filled	Pass
2	Tap on *login* button without filling username and password	Popup alert is shown as warning that mandatory field must be filled	Pass
3	Tap on *login* button after filling correct username and password	Redirected to *map* view	Pass
4	Tap on *login* button after filling incorrect username and password	Popup alert mentioning user has inputted invalid login credential is shown	Pass

Table 5-1 : Testing on login module of mobile application

5.2.1.2 Register Module

Register Module			
Case	**Test Action**	**Test Result**	**Status**
1	Tap on *submit* button with all the input fields left blank	Popup alert is shown as warning that mandatory field must be filled	Pass
2	Tap on *submit* button with some of the input fields left blank	Popup alert is shown as warning that mandatory field must be filled	Pass
2	Tap on *submit* button after filling all the input fields but with incorrect format on email or contact number field	Popup alert is shown as warning that input field must bein correct format	Pass
3	Tap on *submit* button after filling all the input fields using correct format	Success popup alert is shown	Pass

Table 5-2 : Testing on register module of mobile application

5.2.1.3 Map Module

Map Module			
Case	**Test Action**	**Test Result**	**Status**
1	Tap on the rewind icon	Map view moves its center to user current location	Pass
2	Perform one finger drag on the map view	Map view move accordingly to the drag direction	Pass
3	Tap on marker shown in the map view	A popover containing employer business name is shown	Pass
4	After case 3, tap on the popover	*Job detail* view is shown	Pass
5	Tap on the search bar at top	*Search view* is shown	Pass

Table 5-3 : Testing on map module of mobile application

5.2.1.4 Search Module

Search Module			
Case	**Test Action**	**Test Result**	**Status**
1	Tap on the back icon	Map view is shown	Pass
2	Input search criteria and tap *return*	Records of job vacancy containing search criteria is shown	Pass
3	After case 2, tap *Nearest* option	Search result is sorted by nearest distance, the nearest result is shown on top	Pass
4	After case 2, tap *Highest Pay* option	Search result is sorted by highest pay, the result with highest pay is shown on top	Pass
5	After case 2, tap on one of the record shown in search result	The respective *job detail* view is shown	Pass

Table 5-4 : Testing on search module of mobile application

5.2.1.5 Job Detail Module

Job Detail Module			
Case	**Test Action**	**Test Result**	**Status**
1	Tap on the hide icon	Job detail view is hidden from view and map view is shown	Pass
2	Tap *Check Review* button	Review about the employer of the job is shown	Pass
3	Tap *Apply for this job* button	Success popup alert is shown and the text on the button tapped changes to “Withdraw Application”, a new record is added to the *notification* view	Pass
4	After case 3, tap *Withdraw application* button	The text on the button tapped changes to “Apply for this job”, the respective application record is removed from *notification* view	Pass

Table 5-5 : Testing on *job detail* module of mobile application

5.2.1.6 Review Module

Review Module			
Case	**Test Action**	**Test Result**	**Status**
1	Tap on any previously worked job record	Redirect to *write review* view	Pass
2	After case 1, leave blank on all input fields	*Submit* button is greyed out and does not respond to touch input	Pass
3	After case 1, leave blank on some of the input fields	*Submit* button is greyed out and does not respond to touch input	Pass
4	After case 1, fill in all the input fields	*Submit* button change to blue colour and accept touch input	Pass
5	After case 4, tap on the *Submit* button	Success popup alert is shown and redirect to review view	Pass

Table 5-6 : Testing on review module of mobile application

5.2.1.7 Profile Module

Profile Module			
Case	**Test Action**	**Test Result**	**Status**
1	Tap on any of the following : *name* input field, *email* input field or *mobile no.* input field.	Cursor is shown at the end of the value and *cancel* button is shown on top right	Pass
2	After case 1, edit the value on the input field and tap *return*	The value on the edited input field is changed to the new value	Pass
3	After case 1, edit the value on the input field and tap the cancel button	The value on the edited input field remain unchanged	Pass
4	Tap the row with text “Reviews from employers”	Reviews left by previous employer is shown	Pass
5	Tap the row with text “Previous job”	Previously worked jobs are shown	Pass
6	Tap on *logout* button	Redirect to *login* view	Pass

Table 5-7 : Testing on profile module of mobile application

5.2.2 Web application

5.2.2.1 Login Module

Login Module			
Case	**Test Action**	**Test Result**	**Status**
1	Click on *login* button without filling username or password	Alert is shown as warning that mandatory field must be filled	Pass
2	Click on *login* button without filling username and password	Alert is shown as warning that mandatory field must be filled	Pass
3	Click on *login* button after filling correct username and password	Redirected to control panel index page	Pass
4	Tap on *login* button after filling incorrect username and password	Alert mentioning user has inputted invalid login credential is shown	Pass

Table 5-8 : Testing on login module of web application

5.2.2.2 Profile Module

Profile Module			
Case	**Test Action**	**Test Result**	**Status**
1	In *show profile* view, click on *Edit Profile* button in the bottom	Redirect to *edit profile* view	Pass
2	After case 1, do nothing	Username input field and email input field is greyed out and cannot be edited	Pass
3	After case 2, left all the editable input field empty and click *update profile* button	Alert is shown as warning that mandatory field must be filled	Pass
4	After case 2, fill in all the editable input field except *new password* field and *confirm new password* field, without image attachment then click *update profile* button	Success message is shown at top	Pass
5	After case 2, fill in all the editable input field including *new password* field and *confirm new password* field, the value inputted in *new password* field is the same as *confirm new password* field, without image attachment then click *update profile* button	Success message is shown at top	Pass
6	After case 2, fill in all the editable input field including *new password* field and *confirm new password* field, the value inputted in *new password* field is different with the value in *confirm new password* field, then click *update profile* button	Alert is shown as warning that *password confirmation* value doesn't match *password*	Pass
7	After case 2, fill in *new password* field and some of the other input fields but left *confirm new password* input field blank, click *update profile* button	Alert is shown as warning that *password confirmation* value doesn't match *password*	Pass
8	After case 2, attach an image larger than 5MB and click *update profile* button	Alert is shown as warning that image size is too large	Pass
9	After case 2, attach an image smaller than 5MB and click *update profile* button	Success message is shown at top	Pass

Table 5-9 : Testing on profile module of web application

5.2.2.3 Jobs Module

Jobs Module			
Case	**Test Action**	**Test Result**	**Status**
1	In *list job* view, click on any job vacancy record	Redirect to *edit job* view	Pass
2	After case 1, left all the editable input field empty and click *update vacancy* button	Alert is shown as warning that mandatory field must be filled	Pass
3	After case 1, fill in partial of the form and click *update vacancy* button	Alert is shown as warning that mandatory field must be filled	Pass
4	After case 1, fill in all mandatory input field without image attachment and click *update vacancy* button	Success message is shown at top	Pass
5	After case 1, Attach an image larger than 5MB and click *update vacancy* button	Alert is shown as warning that image size is too large	Pass
6	After case 1, Attach an image smaller than 5MB and click *update vacancy* button	Success message is shown at top	Pass
7	Select *create job* from *Jobs* tab on the top menu	Redirect to *create job* view	Pass
8	After case 7, left all the editable input field empty and click *create vacancy* button	Alert is shown as warning that mandatory field must be filled	Pass
9	After case 7, fill in partial of the form and click *create vacancy* button	Alert is shown as warning that mandatory field must be filled	Pass
10	After case 7, fill in all mandatory input field without image attachment and click *create vacancy* button	Redirect to *list job* view*, s*uccess message is shown at top	Pass
11	After case 7, fill in all mandatory input field, attach an image larger than 5MB and click *create vacancy* button	Alert is shown as warning that image size is too large	Pass
12	After case 7, fill in all mandatory input field, attach an image smaller than 5MB and click *create vacancy* button	Redirect to *list job* view*, s*uccess message is shown at top	Pass

Table 5-10 : Testing on jobs module of web application

5.2.2.4 **Applications Module**

Applications Module			
Case	**Test Action**	**Test Result**	**Status**
1	Select *list job application* from *Applications* tab on the top menu	Redirect to *list job application* view	Pass
2	After case 1, click *accept* button on any pending job application record	The job application status changes to *accepted*	Pass
3	After case 1, click *reject* button on any pending job application record	The job application status changes to *rejected*	Pass

Table 5-11 : Testing on applications module of web application

5.2.2.5 **Reviews Module**

Applications Module			
Case	**Test Action**	**Test Result**	**Status**
1	Select *list review written* from *Reviews* tab on the top menu	Redirect to *list review written* view, your previously written review is shown	Pass
2	Select *received review* from *Reviews* tab on the top menu	Redirect to *received review* view, review written by other about you is shown	Pass
3	Select *write review* from *Reviews* tab on the top menu	Redirect to *write review* view	Pass
4	After case 3, left all the input field empty and click *create review* button	Alert is shown as warning that mandatory field must be filled	Pass
5	After case 3, fill in partial of the form and click *create review* button	Alert is shown as warning that mandatory field must be filled	Pass
6	After case 3, fill in all mandatory input field and click *create review* button	Redirect to *list review written* view, *s*uccess message is shown at top	Pass

Table 5-12 : Testing on reviews module of web application

Chapter 6 : Conclusion

6.1 Conclusion

This project aims to tackle the uncultivated niche market of local part time job application by developing a mobile application for jobseeker and a web application for employer which both of them share the same database.

The mobile application will provide convenience to current jobseeker by showing nearby job opportunity and allows jobseeker to determine the credibility of an employer by gathering peer review information. The web application allows employer to submit job listing with its coordinate pinpointed on a map and it also let employer review potential candidate previous job performance before making a hiring decision. Part time job searching and application process can be simplified and enhanced by using these applications.

Objective of this project is achieved with the utilisation of various technology such as global positioning system (GPS) and location service provided by iOS by default. The implementation of REST API on the web application allows effective and efficient communication between the web application and mobile application. Rails web application framework is also used for faster iteration of the prototype and lesser repeated code.

6.2 Discussion

Currently in the local market (Malaysia), there does not exist a mobile application which is catered for part time job application and there is a high demand for part time workforce. Part time job vacancy information are usually scattered around various online discussion boards and blogs thus making it hard for jobseeker to search for, some part time opportunity may not catered to the potential jobseeker current location and the credibility of the employer can’t be easily identified. By solving this problem,

values can be created as the time and effort used of both jobseeker and employer can be reduced.

6.3 Future Work

Firstly, due to monetary constraint, Secure Sockets Layer (SSL) certificate was not purchased and installed on the cloud server, data passed during communication between mobile application and web application may be susceptible to man-in-the-middle attack and eavesdropping. In the future if sufficient monetary resource is attained, SSL certificate should be purchased from Certificate Authority (CA) and installed on the web server to strengthen security of the overall system.

Secondly, due to the imposed time constraint, payment system was not integrated into the system hence the system cannot receive payment from employer. In the future payment system can be integrated into the system and employer can purchase listing space from the system to list their job vacancy on the system.

References

1. Jobstreet.com.my 2015, *JobStreet.com Malaysia - Jobs in Malaysia, Selangor, Kuala Lumpur, Penang, Johor, Singapore, Asia. Jawatan Kosong*. Available from: <http://www.jobstreet.com.my/> [Accessed 18 January 2015].

2. Jenjobs.com 2015, *Job Vacancy | Job Search | Online Jobs in Malaysia | Jawatan Kosong : JenJOBS*. Available from: <http://http://www.jenjobs.com/> [Accessed 18 January 2015].

3. Jobless.com.my 2015, *Part Time and Freelance Jobs in Malaysia*. Available from: <http://http://www.jobless.com.my/> [Accessed 18 January 2015].

4. JobStreet.com Annual Report 2012. (2012). [online] JobStreet.com, p.8. Available at: http://ir.chartnexus.com/jobstreet/doc/Reports/jobstreet_2012_full.pdf [Accessed 18 January 2015].

5. Fielding, R. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. Ph.D. University of California.

6. Torvalds, L. (2007). *Tech Talk: Linus Torvalds on git*. [video] Available at: https://www.youtube.com/watch?v=4XpnKHJAok8 [Accessed 31 Aug. 2015].

Appendix

A-1 Turnitin originality report

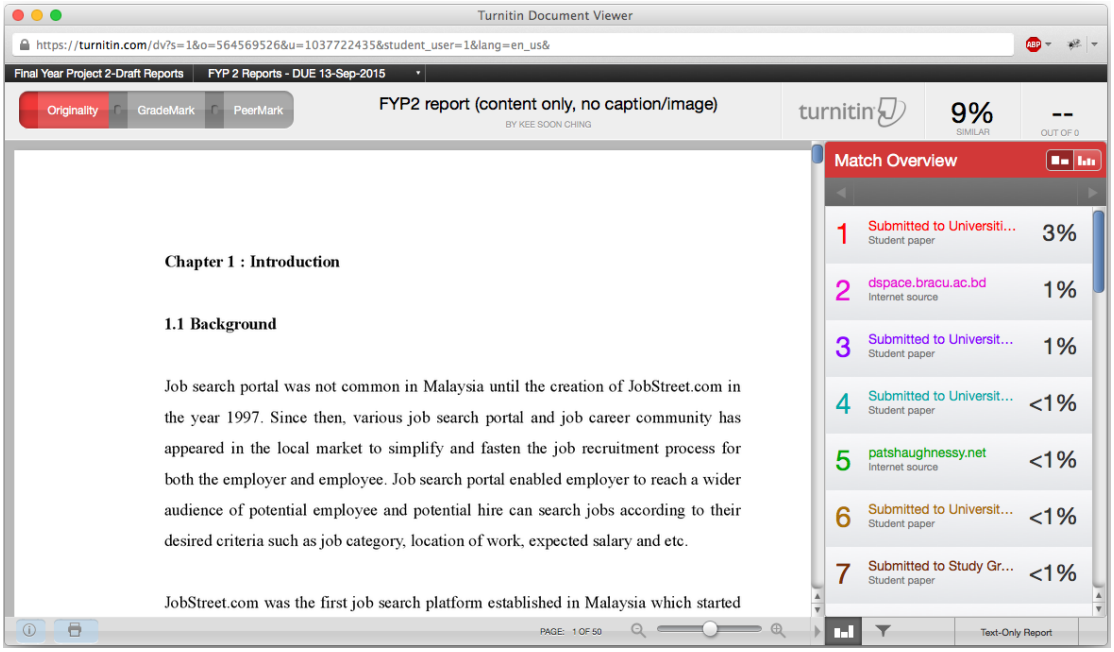

Figure A-1 : Turnitin originality report