# Functional Specification

Name: Adrian Rabbitt

Student Number: 14500447

Title: Jobder

Finish Date: 23/11/2017

# Jobder Functional Specification

## Table of Contents

# 1). Introduction

## 1.1) ***Overview***

The Android application Jobder is a virtual Employment System for all fields and skills of work. The application will try and overcome the employment barriers that numerous people are faced with. The main target group I hope to help are students, people with no qualifications and poor lexical skills. The application will consist of two modes: The Employer and the Employee. The Employee is responsible for filling in his/her details. The Employee's main features will consist of creating/editing their profile, viewing their conversations and swapping jobs if in search.

The Employer mode will consist of a login / sign up, the employer will be brought to an activity where he/she can pick a field in which they are in search of Employees e.g. painting, gardening etc. When the Employer picks his/her field in search of employment they will be brought to a map displaying the Employees registered to that field according to their location with their profile picture displayed. The Employer will click on the employee's profile picture which will allow them to view their CVs and previous ratings from other employers. When the Employee has studied the Employee's CV they will have the chance to message the Employee and try to agree on a fixed term of work.

There will be a web application monitoring the stats of employment and the ratings of Employees with easy-to-view charts and graphs. The web application is only for statistics to view for the public.

This is a very brief overview on which the application stands.  There are many customable features on the application which will allow the Employer or Employee adjust their needs to the application.

## 1.2) ***Business Context***

The economic crash in 2008 left a significant number of people without employment in the last couple of years. However, the economic climate has been improving with new jobs being created.   Although the media is focusing on the upturn, unfortunately the situation has not improved for certain groups. For people with no qualifications, poor lexical skills and students, the main problem encountered is communication rather than available work. I have studied this after personal experience with employment difficulties and have been thinking of ways to overcome this problem. The creation of this application that I will design will try and overcome these employment barrier's.

The application Jobder has all the features to grow and explore the recruitment industry. This application will differ from other employment applications due to its wide target groups and its ease, simplicity and graphical features.

The application has similar features to Snapchat and Tinder: two of the most successful and creative applications of the decade. The introduction of snap maps into Snapchat was a very clever move for the chat application. This feature was a key elimination of the cons of the "fake" social media bringing peoples physical characters into the limelight.

The two applications were developed for conversational use, however the application that I am developing is borrowing some of their popular features, e.g. maps, but applying them in a more serious context. The Jobder application in a recruitment app for the casual labour market.

## 1.3) ***Glossary***

- *Java*
  Java is an object-oriented programming language which has an independent platform.

- *Xml*
  XML stands for extensible Mark-up Language, like HTML. The language was designed to store and transport data which was designed to be self-descriptive.

- *SQL*
  Structured Query Language, SQL is a standardized query language for requesting information from a database.

- *Java Script.*
  Java Script is an object-oriented computer programming language commonly used to create interactive effects within web browsers and web applications.

- *PHP*
  PHP is Hypertext Pre-processor, an open source, server-side, HTML embedded scripting language used to create dynamic Web pages.

- *HTLM5*
  HTML5 is a mark language used for structuring and presenting content on the World Wide Web.

- HT*ML/CSS*
  Short for Hypertext Mark-up Language, the authoring language used to create documents on the World Wide Web. CSS stands for Cascading Style Sheets.

# 2). General Description

## 2.1 ***Product / System Functions***

- *Login/Sign up*
  When the users of Jobder start's the application, they will be asked to Sign up or Login. When the user enters his/her password it is submitted to the NoSQL cloud database on firebase. This database will have a record and relation of all members to allow use of the Android system.

- *Google Maps API displaying images according to their location.*
  When the Employer pick's a certain area of work to search, he/she will be brought to Google maps API where they are able to view all the available Employees in a certain range. Their profile pictures will be displayed using a bitmap according to their location through the local GPS provider. The data involved includes images, personal information and latitude and longitude coordinates will be stored in a database and will be retrieved on notice.

- *Instant Messaging.*
  The application Jobder will have the feature of instant messaging for Employers and Employees to communicate. This will be implemented using FCM which is a cross-platform messaging solution. This allows the application to notify a client when there is data available to sync. The Employer will contact the Employee first.

- *Rating System*
  Jobder will allow Employers to rate Employees according to their completed work. This is to allow Employers to choose carefully between Employees.

- *Push Notifications*
  Employees and Employers will be signalled using push notifications. The notification system that I will implement will be provided through goggle's mobile platform Firebase.

- *Statistics of Android applications through employment.*
  The web application in the system will display employment information using graphs this will be implemented using JavaScript and HTML/CSS. The web application will display simple graphs using the data from the Android application.

## 2.2) ***User Characteristics and Objectives***

The user community can be broken down into two groups. One being the Employers and the other being Employees. Firstly, we will look at the Employers who will be the assessors in the employment application where their focus is finding an individual that will meet their needs. The application will have to be designed in such a way that will allow the Employer as fast and as easy way as possible to get an Employee that will suit their business needs.

This will be achieved through the simple interface design and fast accessibility of the employee’s information. The Employer will need to build up a trust in the application therefore there will be rating system implemented so the Employer will know the standard of work the employee has produced in their past.

The Employee application will be designed so the user of the application will find it easy to create a profile. This will be achieved by creating a simple application interface that will use fields to create and simply edit their Curriculum Vitae.

The target user groups for this application would be students and people with poor lexical skills. These target groups consist of a large sample of people with no qualifications.

## 2.3) ***Operational Scenarios***

### Employer

1) ***LOGIN / SIGN UP***
   The Employer will be given the choice to either sign up or login.

2) ***Choose area of employment search***
   When the employer clicks on certain fields e.g. Painting. All the employee's profiles in that set area will be displayed by their profile pictures, age and distance on Google maps API.

3) ***View Profile***
   The Employer can now see all the available workers on Google Maps API according to their profile picture. The Employer can now click onto their picture and view their profile which will consist of their CV. If the Employer is satisfied with the Employee he/she has the opportunity to instant message the employee to agree on a fixed term of work.

4) ***Rating***
   When the work is completed the Employer has the chance to rate the Employee and comment on his/her work, this feature will only be visible for the Employer to avoid certain conflicts between the two parties.

5) ***Settings***
   The employee will have customizable settings. He/she will have the ability to view workers inside a certain range of distance. e.g. view all the workers within a 20-km radius.

### Employee

1) ***LOGIN / SIGN UP***
   The Employee will be given the choice to either sign up or login.

2) ***Create Profile/ Edit Profile***
   Here the Employee will be allowed to create a new profile or edit their profile. For their profile picture they will be allowed to take a picture using the built-in camera from the application or upload a picture from their album. The Employee will have certain fields to fill in e.g. Skills, Description, Previous Employment. The Employee will fill in these fields which their grammar and a spelling will automatically be corrected. This

layout will help people with poor lexical skills and people who never created a CV to complete it under these simple headings. The Employee will have to pick certain fields he/she is interested in working in, e.g. Painting, Pub Worker.

3) ***Settings***
The Employee has the chance to personalise their account. They can choose whether to display or hide their profile, change profile and set their location to either Static or Roaming.

## 2.4*) Constraints*

1) ***Login/ Sign Up***
- o Key Functions
  - Keep users distinguished.
  - Allow new users to register.
- o Requirements
  - Differentiate users in database.
- o Constraint
  - People may create fake accounts and give Employees false ratings.

**2)** ***Connection to Internet (Data Connection)***
- o Key Functions
  - Jobder users must have access to the Internet for retrieval of data.
  - Jobder users know when to turn off their data connection when the application is not in use.
- o Requirements
  - Allow the application to signal the user to turn on cellular if not on.
  - Create warnings that there might be significant use of data.
- o Constraint
  - User may not have access to the Internet.

**3)** ***Location on device is turned on***
- o Key Functions
  - When using Jobder the applications location settings must be turned on to retrieve the employee's current location.
  - Google Maps must be viewable on all devices.
- o Requirements
  - If the users Location is not on create a pop up to turn on the user's location.
  - Make sure the Google maps API key is accepted on all Android applications.

- Constraint
  - User may not have their location turned on.

4) ***Screen Resolution***

- Key Functions
  - The Interface of the Android application is viewable on all Android devices with respect to the applications dimensions.
  - Colour is displayed evenly on all Android devices.
- Requirements
  - The application is designed so the application will fit perfectly aligned with all Android devices.
- Constraint
  - Some devices may not support the application.

5) ***Push Notifications***

- Key Functions
  - Notifications received and sent when the user is alerted.
  - Notifications will not jam up home screen when there are many alerts.
- Requirements
  - Separate the push notification system established separate from databases.

    .
- Constraint
  - Notifications may get blocked.

6) ***Server Memory***

- Key Functions
  - Stores data relevant to the application.
- Requirements
  - Server has enough memory to support large number of users.
- Constraint
  - Server may run out of memory for the application.

# 3). Functional Requirements

## 3.1) *Login / Security.*

*Description.*

This is the first feature of the application Jobder, here the Employer / Employee is asked to either sign up or login. When the user downloads the Android application, they will not have access to view the applications services until they sign up. When the user of the application has signed up they will be automatically logged in to their account as an Employer/ Employee.

*Criticality.*

This feature is important within the application as it defines the user to their registered account and allows users to create their independent accounts. Without this feature there would be no purpose to the application.

*Technical Issues*

The user needs to have access to the Internet to connect to the Server that hosts the user’s credentials and allows the user to register to the Server. The application must retrieve the user’s credentials and check if they are matched on the server. This will allow the user access to the application.

*Dependencies with other requirements*

This is the first level of security the application provides, without the application depending on the server there would be no level of privacy and security leaving the application pointless.

## 3.2) *Display the Employer and Employee home page.*

*Description.*

Now that the user has logged in successfully to the Android application, they are brought to their home screens. The Employers home screen will consist of a list of tabs in scroll view containing all the jobs they are available to look for in search of Employees. This page will also contain links to the Employers settings and Conversations. The Employees home screen will consist of basic features the Employees has to choose from including settings, personal features and conversations.

*Criticality*

These features are straight forward influencing the users to their different environments. However, the Employers home screen consists of tabs in scroll view listing the available fields of employment. These fields are retrieved from the server to avail of automatic change and if new fields of work are introduced to the Android system. These fields are critical to the Employer as it is their first step in viewing the Employee's profiles.

*Technical Issues*

The home screen of the Employer will contain a running background thread that will connect to the server and retrieve the list of available jobs to the Employer. When the application is loaded, and if the Employer has no access to the updated list of employment fields this may lead to an error further down it the Employers application.

*Dependencies with other requirements*

This feature is the first step in Employer to Employee communication, it is very important that the Employer has a connection to retrieve this updated data. To prevent this from happening the application should be designed to automatically switch on the user's data connection and alert the user if there is no potential connection to the Internet.

### 3.3) ***Creating Employees Profiles***

*Description.*

The Employees home page of the application has the feature to create or edit their profile. This profile will consist of the Employees Curriculum Vitae including their profile picture. The Employees profile will be the advertisement to the Employer for their preferred position of work. The user will have the choice to upload their profile picture from their camera album or take the picture using the applications built in camera.

*Criticality*

This feature of the application is quite critical to the Employee as it is their only way of retrieving the Employers attention.

*Technical Issues*

The retrieval of the user’s photo to the profile must be able to handle the photo size and scale it down to the correct position. If the user selects to retrieve their photo from their camera album the application must have access to access the user’s photos.  When the Employees application is completed their details must be uploaded to the server along with their photo where the Employer will have access to view.

*Dependencies with other requirements*

This feature of the application is also dependant on the client server architecture, including Internet access. The Android application is also required that the mobile device has enough memory and a suitable camera to obtain the profile pictures.

## 3.4) ***Google Maps and Location***

*Description*

When the Employer clicks on an individual field of employment, He/she is brought to Google Maps API where the employees are displayed on the map according to their profile pictures. The Employees profile pictures are displayed on Google maps according to their GPS location.  When the Employer clicks on the Employees profile picture they are brought to the Employers profile containing their Curriculum Vitae.

*Criticality*

The Profile pictures displayed on Google Maps API according to their location is part of the main feature of this application. This feature is the main communication interface of the application which allows Employers to interact with Employees using the application.

*Technical Issues*

Connecting to Google map API will require getting an API key from the google maps console, this will allow access to view scaled maps through the application.  The Employees location will be determined by their Longitude and Latitude values generated by the Location Manager.

*Dependencies with other requirements*

The positioning of the Employees profile pictures on Google Maps is determined by the Employees location therefore the Android device is required to have its location turned on when using the application.

### 3.5) ***Real Time Instant Messaging.***

*Description*
Within the application the Employer will alert the Employee using Realtime instant messaging. The chat will be hosted within the application with the main purpose of communication between the Employer and the Employee.

*Criticality*

Real time messaging is critical within the application as it is the only way of contact between the two parties. The speed and delivery of the messaging is also an important factor as we are dealing with Employment.

*Technical Issues*

The real time messaging feature within the application will be implemented through Firebase.
Technical Issues may arise when syncing the data for saved conversations.

*Dependencies with other requirements*
Real Time instant messaging will require Internet access and an API key for installation.

### 3.6) ***Push Notifications.***

*Description.*
The two parties in the application will be notified by push notification, to alert of messaging, new employees in the system, highly recommended employees etc.
The notification format will be in the form of a standard notification displaying the user with an event.

*Criticality*

Push notifications within the application are critical in terms of instant alerts and improving the speed of communication.

*Technical Issues*

Push Notifications are received and sent through the online mobile platform Firebase. There may be potential issues due to the build-up of notifications that could arise.

*Dependencies with other requirements*

Push Notifications will require Internet access and an API key for installation.

### 3.7) ***Grammar Fixing.***

*Description.*

When the Employee is filling in their Curriculum Vitae their grammar will be checked using the spell checker service in Android.

*Criticality*

This Feature is important to the Employee as it will help them identify grammar mistakes and prompt the user to fix them.

*Technical Issues*

This may lead to difficulties when I am dealing with an open source spell checker.

*Dependencies with other requirements*

Spell checker is independent and has no dependencies, but it requires the text view to be greater that a certain size in which the text is inserted.

### 3.8) ***Web Application.***

*Description.*

The Android application is accompanied by a web application for the only purpose of monitoring Employment data.  The web application will have a simple interface displaying charts showing the number of jobs in different employment sectors. The users of the application will be the public as there will be no personal information displayed.

*Criticality*

The web application is not critical to the system as it is only recording certain data and displaying it to the public.

*Technical Issues*

The web application will have to be connected to the same server as the Android application, this may cause difficulties when querying data.

*Dependencies with other requirements*

The webserver is dependent on the data provided by the Employees and Employers Application.

# 4). System Architecture

Fig (4.1)

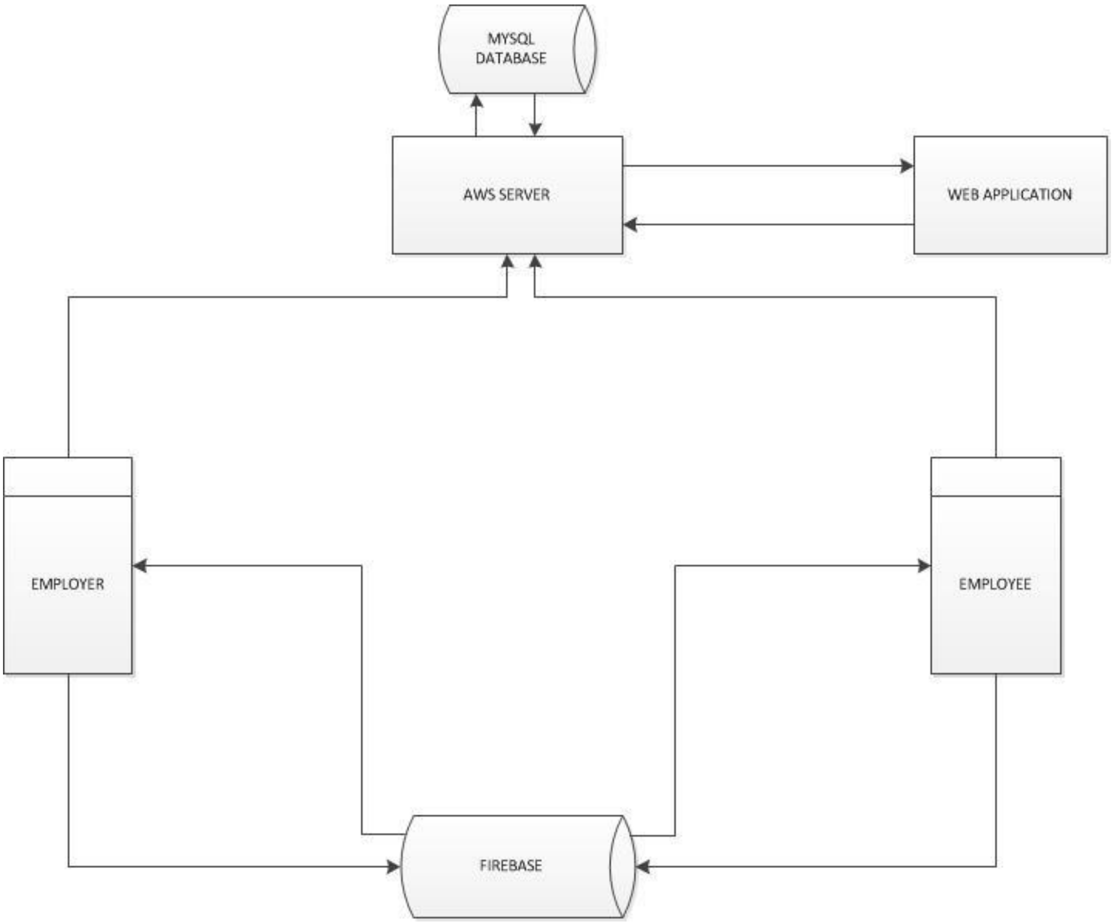


Fig 4.1) Illustrates the architecture of the product. As the above diagram shows there are five distinct aspects to the architecture.

## ***Employer & Employee***

The Employer side of the application. Here the employer logs in to the system for their personal use. The Employer version of the application will be connected to the Google mobile platform firebase. The application will also be connected to the AWS server through PHP scripts.

## ***AWS Server***

The AWS server will be used for hosting the web application and MYSQL database along with the storage of php scripts.

## ***MYSQL Database***

The MYSQL database will be used for storing information that is requested by the web application such as the latitude and longitude variables.

## ***Web Application***

The Web application will be used just for viewing data and statistics for monitoring the mobile application. This side of the application is only used solely for information needs, showing where there is a peaks and downfalls in the employment industry.

## ***Firebase Server***

Google`s online mobile platform will be used for connecting data that is requested by both parties (Employer and Employee). This platform will provide FCM (Firebase Cloud Messaging) which will automatically update the data and notify the client and other data that is available to sync. Firebase will also hold the data that is personal to each user group e.g. photos, curriculum vitae.

# 5). High-Level Design

Fig (5.1)

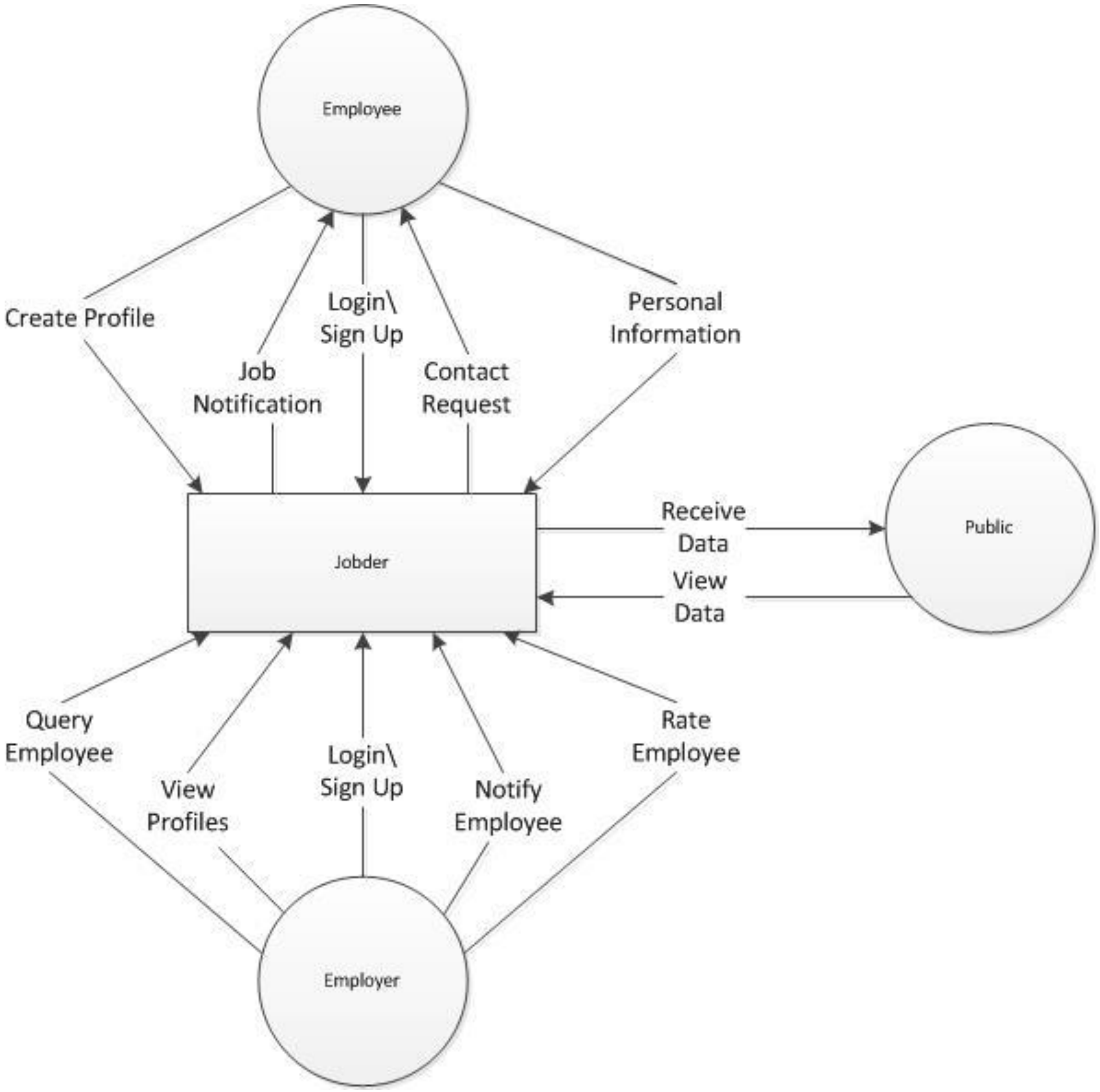


Fig (5.1) Shows the Context Diagram for Jobder.

Fig (5.2)

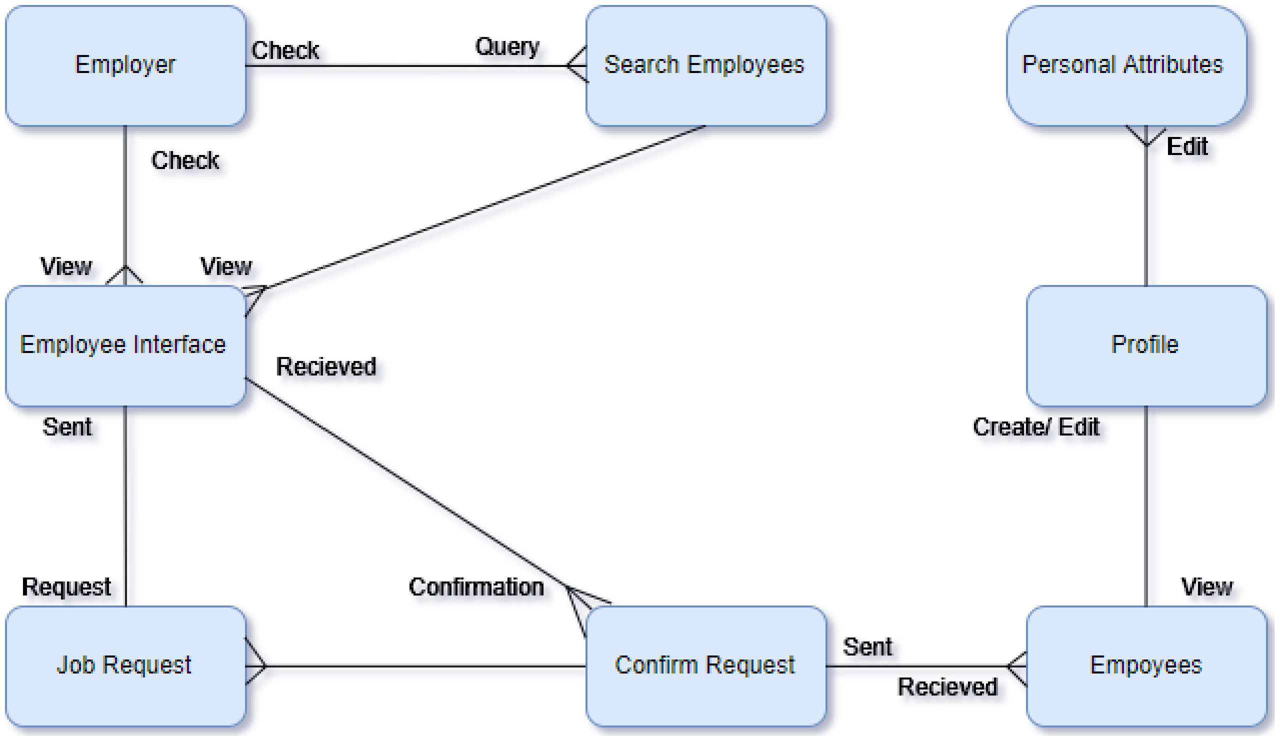


Fig (5.2) shows the LDS Diagram for Jobder.

Fig(5.3)

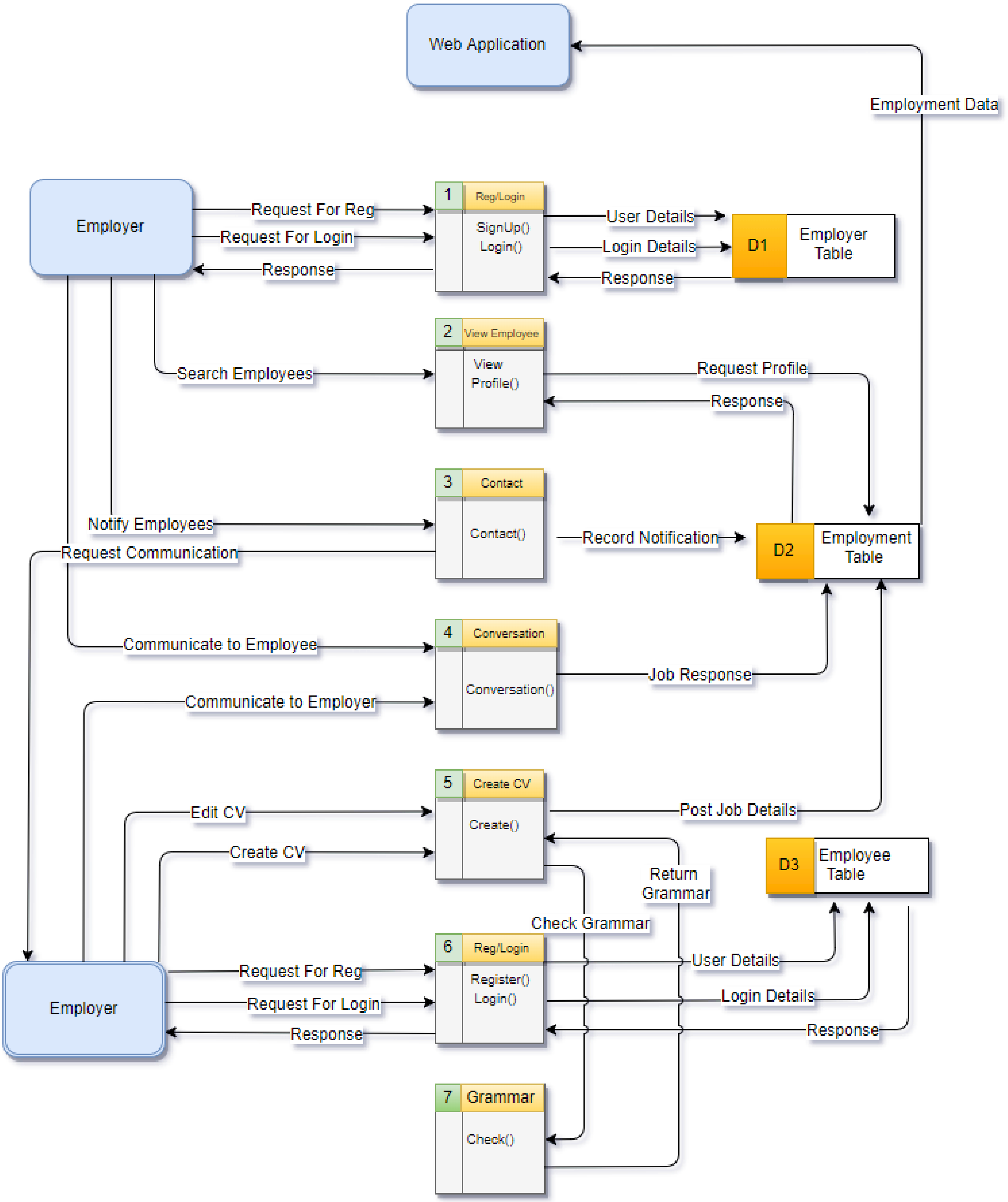
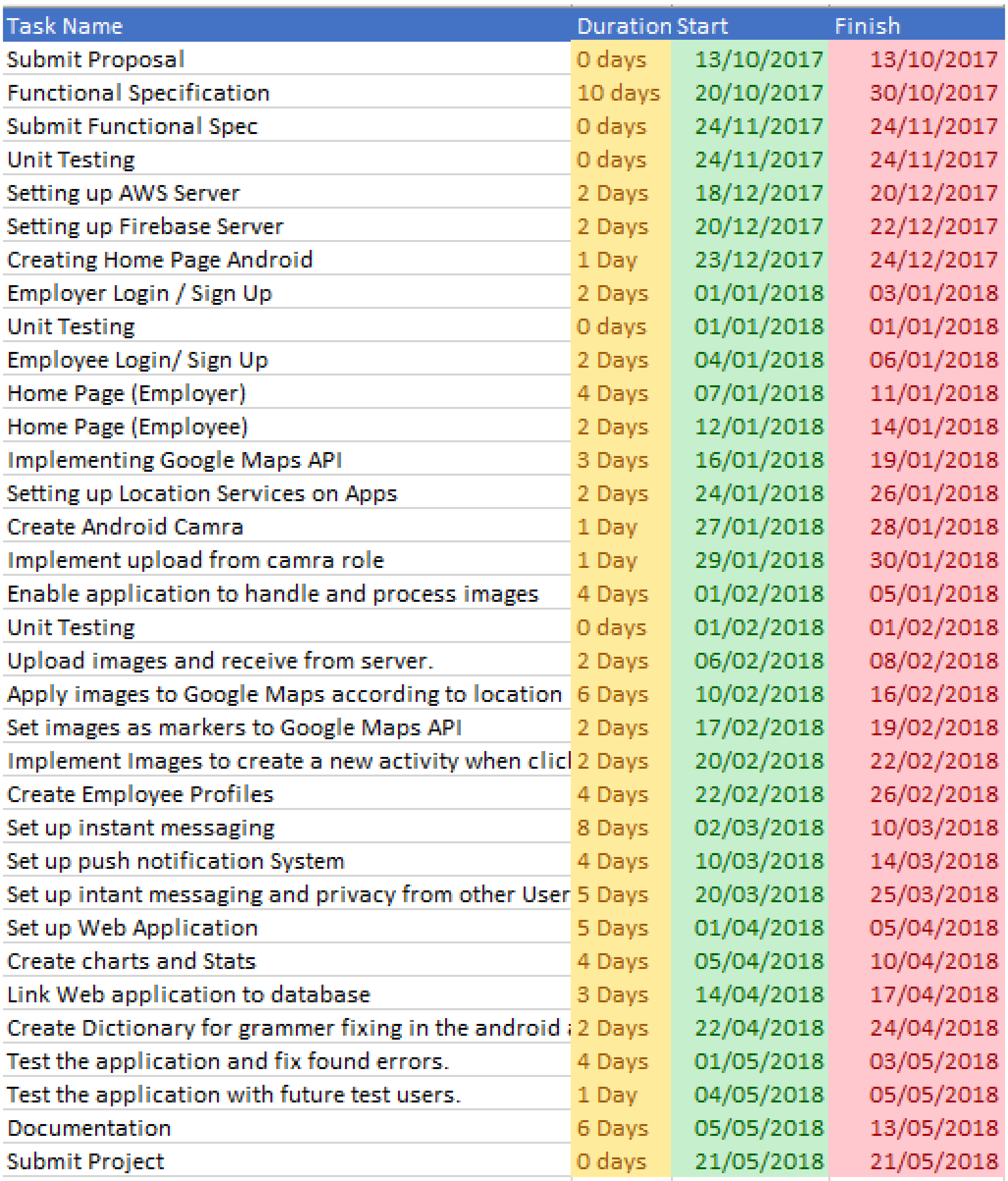


Fig (5.3) Shows the DFD Diagram for Jobder.

# 6). Preliminary Schedule

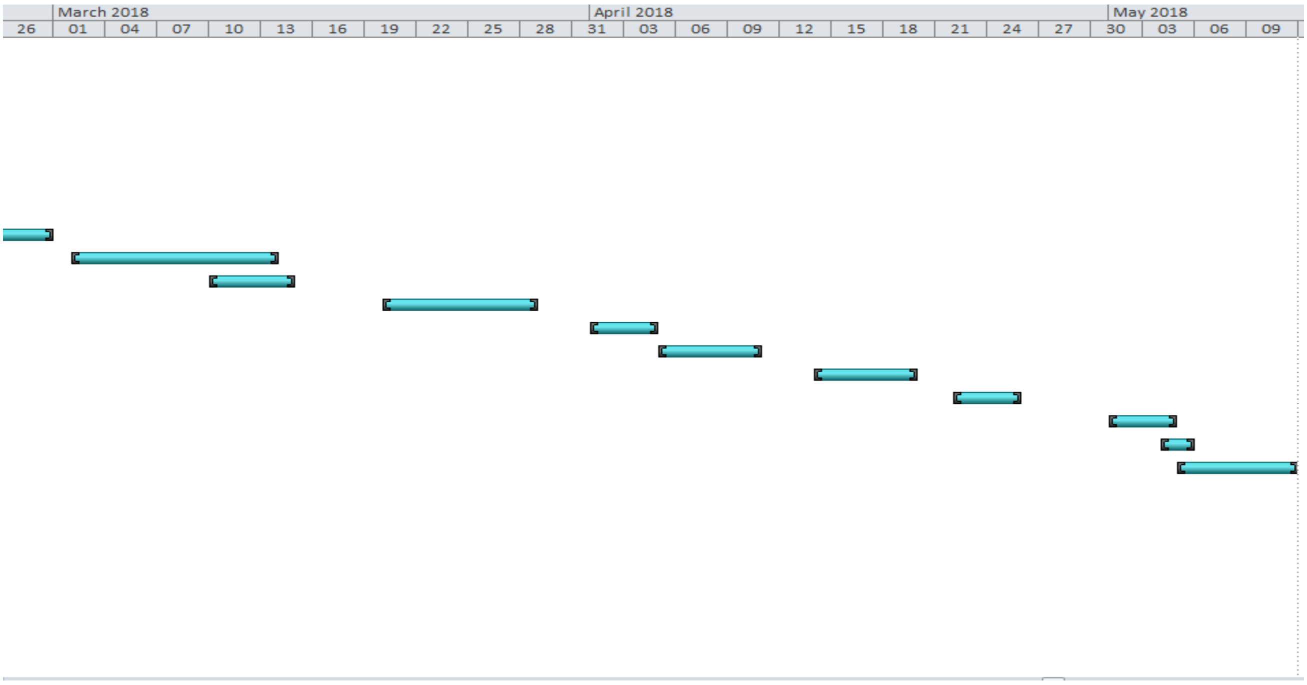
## Task List (6.1)

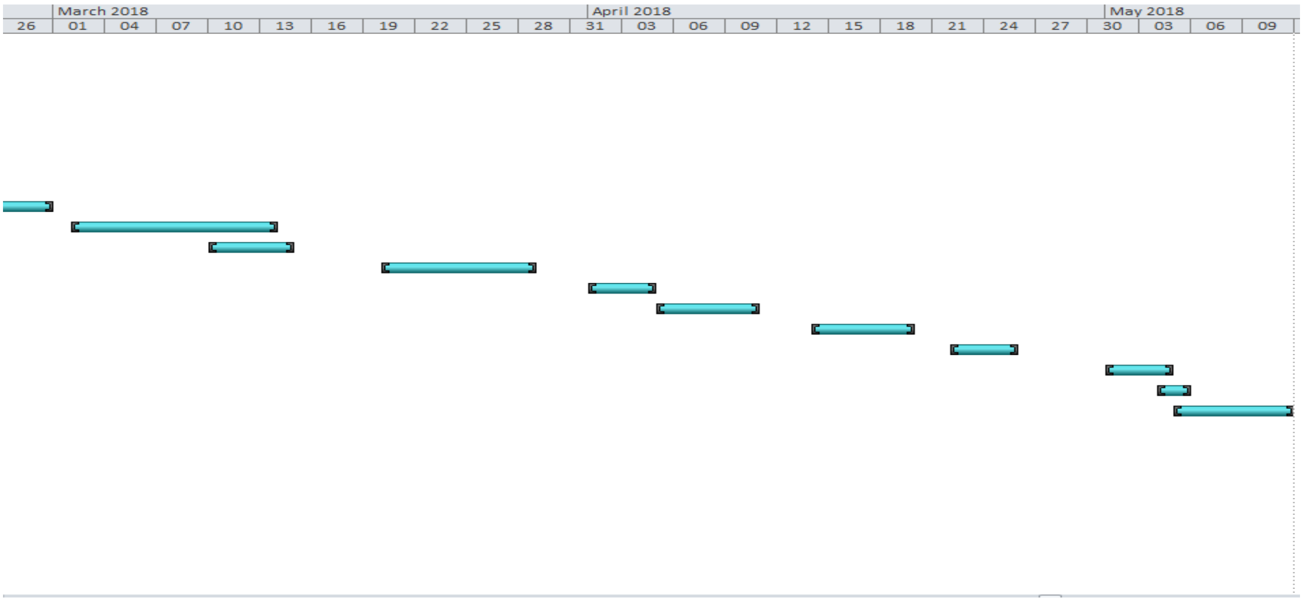


| Task Name | Duration | Start | Finish |
|---|---|---|---|
| Submit Proposal | 0 days | 13/10/2017 | 13/10/2017 |
| Functional Specification | 10 days | 20/10/2017 | 30/10/2017 |
| Submit Functional Spec | 0 days | 24/11/2017 | 24/11/2017 |
| Unit Testing | 0 days | 24/11/2017 | 24/11/2017 |
| Setting up AWS Server | 2 Days | 18/12/2017 | 20/12/2017 |
| Setting up Firebase Server | 2 Days | 20/12/2017 | 22/12/2017 |
| Creating Home Page Android | 1 Day | 23/12/2017 | 24/12/2017 |
| Employer Login / Sign Up | 2 Days | 01/01/2018 | 03/01/2018 |
| Unit Testing | 0 days | 01/01/2018 | 01/01/2018 |
| Employee Login/ Sign Up | 2 Days | 04/01/2018 | 06/01/2018 |
| Home Page (Employer) | 4 Days | 07/01/2018 | 11/01/2018 |
| Home Page (Employee) | 2 Days | 12/01/2018 | 14/01/2018 |
| Implementing Google Maps API | 3 Days | 16/01/2018 | 19/01/2018 |
| Setting up Location Services on Apps | 2 Days | 24/01/2018 | 26/01/2018 |
| Create Android Camra | 1 Day | 27/01/2018 | 28/01/2018 |
| Implement upload from camra role | 1 Day | 29/01/2018 | 30/01/2018 |
| Enable application to handle and process images | 4 Days | 01/02/2018 | 05/01/2018 |
| Unit Testing | 0 days | 01/02/2018 | 01/02/2018 |
| Upload images and receive from server. | 2 Days | 06/02/2018 | 08/02/2018 |
| Apply images to Google Maps according to location | 6 Days | 10/02/2018 | 16/02/2018 |
| Set images as markers to Google Maps API | 2 Days | 17/02/2018 | 19/02/2018 |
| Implement Images to create a new activity when clicl | 2 Days | 20/02/2018 | 22/02/2018 |
| Create Employee Profiles | 4 Days | 22/02/2018 | 26/02/2018 |
| Set up instant messaging | 8 Days | 02/03/2018 | 10/03/2018 |
| Set up push notification System | 4 Days | 10/03/2018 | 14/03/2018 |
| Set up intant messaging and privacy from other User | 5 Days | 20/03/2018 | 25/03/2018 |
| Set up Web Application | 5 Days | 01/04/2018 | 05/04/2018 |
| Create charts and Stats | 4 Days | 05/04/2018 | 10/04/2018 |
| Link Web application to database | 3 Days | 14/04/2018 | 17/04/2018 |
| Create Dictionary for grammer fixing in the android a | 2 Days | 22/04/2018 | 24/04/2018 |
| Test the application and fix found errors. | 4 Days | 01/05/2018 | 03/05/2018 |
| Test the application with future test users. | 1 Day | 04/05/2018 | 05/05/2018 |
| Documentation | 6 Days | 05/05/2018 | 13/05/2018 |
| Submit Project | 0 days | 21/05/2018 | 21/05/2018 |

Testing will also be carried out during the development process.

# Gannt Chart (6.2)





# 7). Appendices

- https://developers.google.com/maps/documentation/android-api/
- https://firebase.google.com/docs/cloud-messaging/
- https://firebase.google.com/docs/cloud-messaging/
- https://developer.android.com/guide/topics/text/spell-checker-framework.html
- https://developer.android.com/reference/android/location/LocationManager.html
- https://aws.amazon.com/
- https://developer.android.com/reference/android/widget/RatingBar.html
- https://firebase.google.com/docs/auth/
- https://www.w3schools.com/js/