****

**LOOK LOCK**

****

**GROUP: MAD 313**

**PROJECT REPORT**

**SUBMITTED TO:**

Mr. SAKKARVARTHI RAMANATHAN

Mrs. SAKSHI SHARMA

Ms. SILVIYA PASKALEVA

Mr. GILLES-PHILIPPE GRÉGOIRE

Mr. HARSHKUMAR DAVE

**SUBMITTED BY:**

Arshdeep Kaur (1895972)

Gurjot Kaur (1894829)

Manpreet Singh (1896507)

Manjinder Singh (1896356)

Simranjeetkaur (1895442)

Dalwinder Singh (1895607)

**Table of Contents**

1. **Introduction**…………………………………………………………………………2
   1. Purpose…………………………………………………………………………...2
   2. Scope……………………………………………………………………………..2
2. **General Description**....……………………………………………………………...3
   1. Product Functions….………………………………………………………….....3
   2. User Characteristics………………………………………………………….......3
3. **Specific Requirements**……………………………………………………………...4
   1. Functional Requirements….……………………………………………………..4
   2. Non-Functional Requirements…………………………………………………...4
4. **Analysis Models**…………………………………………………………………… 5
   1. Use Case Diagram………………………………………………………………5
      1. Use Case Scenario………………………………………………………7
   2. Class Diagram…………………………………………………………………..19
   3. Entity Relationship Diagram…………………………………………………..20
   4. Screens…………………………………………………………………………..20
5. **Database** ……………..……………………….…….………….….……….....……31
   1. Logical Model………………….………....…………..………...……...……....31
   2. Database Script….………………………………………….………………......32
6. **References**……………………………………………………………….………...36
7. **INTRODUCTION**

Quality jobs are not easy to come by, especially if you are looking for a job change. There are various reasons behind that. Sometimes, the companies outsource the hiring process to third-party companies and sometimes, it ends up with internal hiring.

Whatever it may be, the whole process can be quite tiring. Fortunately, with the influx of apps, now you can do the job search while you are on the go. You can find apps for job searches on multiple platforms like Play Store and App Store.

Job-hunting apps make it easy to hunt for your next opportunity from anywhere, day or night.

"LookLock" app is also based on finding part-time jobs, anywhere and anytime. This especially designed for finding part-time jobs only.This app is going to help all students and those who are looking to earn extra with valuable tips, which can come handy when applying for a job.

So, to stay on the top of your job hunt, stay connected with this app.

**1.1 Purpose**

At LookLock, our mission is to help people get part-time jobs. This app will connect millions of people to new opportunities. With the pretty simple interface of app will help user to explore more opportunity related to their interests.

The whole experience is designed to be pleasant, and there are numerous mechanisms for making sure everyone operates with confidence, transparency, and accountability.

People who want to stay close to home can use this app to find jobs in cities nearby. Also, job seekers can prioritize jobs based on their first, second, and third choice, as well as track the positions they have applied to, the interviews they’ve been on, and the offers they have received.

The foremost motive this app to make user happy by helping them to introduce with new opportunities, which will enhance their work experience and rise up their allowances.

**1.2 Scope**

Job hunting can be time consuming and lengthy, so it’s imperative for people to use their search time wisely. There are an endless amount of resources that job seekers can use as they look for a position, but incorporating apps into a job search can allow people to work smarter, not harder, when looking for a job.

* The "LookLock" application is a part-time job finder app which will target students and those who already have full-time job and want to earn more.
* This app will increase the transparency between job seekers and companies. These initiatives break the barriers of job seekers and skip the process of going to employment agencies. Job seeker can directly interact with the recruiters.
* Job seeker can search for by factors such as location, job title, industry, and salary range.
* Having hundreds of benefits of this application it will become first choice of job seekers. This may boost up the usage of this application.

The future scope of this application will be in next versions:

* There will be more filters to find perfect job.
* We will be adding different format to create resume online.
* This application will show upcoming job fairs and seminars.
* User will have option to share their jobs with their friends by social media. Such as Facebook, whats app, Instagram, Snapchat as well as SMS.
* Job seeker can directly message to employer for certain queries in the application.

1. **GENERAL DESCRIPTION**

LookLock application is a job finding app with the motto of Look for job Lock for future. This application will help user to find a job by sitting in house or while studying. It is very easy to use, so userdoes not need to learn anything before using this app. It will let consumer know about which jobs are currently available in his area and how much will get for an hour. LookLock will provide list of various types of jobs according to the search of user on the same time job seeker can also apply in minutes.

* 1. **Product Function**

Looklock should be an easy job finding application for all users, it will help job seekers to get a job in their cities, areas and according to their qualification. User can view description of job and also can save that job in their list. Every job seeker can apply jobs in their favourite companies and firms and can get good salary.

* 1. **User Characteristics**

LookLock app will target different users which are detailed below:

* Job Hunters
* Employers

1. **Specific Requirements**

The requirements of this mobile application are the needs of the users which will be using in this application. It compiles all the functional and non-functional requirements which are listed here.

**3.1 Functional Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Description** | **Must or Could** |
| FR-1 | Sign Up | An Application must allow job seekers to register themselves. It requires applicant’s full name, age, gender, address, e-mail, phone number, password and confirm password. | Must |
| FR-2 | Sign In | App must have Sign In feature that takes user’s e-mail and password. | Must |
| FR-3 | View Available Jobs | Users can view all available jobs according to their search. | Must |
| FR-4 | Apply | An Applicant can apply more than one job | Must |
| FR-5 | Search | Application must allow applicants to search jobs | Must |
| FR-6 | Save jobs | Application must allow applicants to save their favorite jobs. | Must |
| FR-7 | Update | Applicants can Update their Profile information. | Must |
| FR-8 | Password Recovery | Applicants can retrieve their password if forgot | Must |
| FR-9 | View user`s profile | Admin can update app, register user, register job provider | Must |
| FR-10 | Post, delete jobs | Company can register to provide jobs | Must |

**3.2 Non- functional Requirements**

**NFR01**.

The platform must be Android and minimum version Of Android.

**NFR02- Usability**

The system should be easy to work for modelling, visualizing of the softbody andinteracting with it. For example, varying the parameters at runtime needs a GUI for thatso user can enter the parameters easily or allowing user to add multiple displays for a simulation.

**NFR03- Response Time**

The system must response at a reasonable time. For some applications, response at realtime is important, whereas for others are not.

**NFR04- Resource Utilization**

Identifies how much resource is needed at a unit of time so that the system can operate.

This non-functional requirement depends on the efficiency of algorithms. For a giventime step, the more computation is needed, the more resources are needed forComputation.

**NFR05- Accuracy**

Identifies how the created soft body object and its behaviour at simulation time are near toreality. Accuracy depends on LOD and time step. Less time step and more LOD lead to amore accurate softbody object.

**NFR06- Portability**

The system should work on other platforms and operating systems.

**NFR07- Stability**

The simulation shouldn’t crash at simulation time. Stability depends on the size of time

both in shape and behaviour and response to the actions in real time.

1. **Analysis Models**

In this documentation, there are three types of models to describe the functionality, information and behaviour of the system.

* 1. **Use Case Diagram**

A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behaviour, and not the exact method of making it happen. Use cases once specified can be denoted both textual and visual representation. The different symbols used in this diagram are explained below-

 Actor specifies a role played by a user or any other systemthat interacts with the subject.



It defines an interaction between an actor and a system to achieve a goal.



This box indicates the boundary of the system you want to highlight or focus on.



A solid linking line is conecting user with use cases, it indicates that use case and actor are communicating with each other through messages.

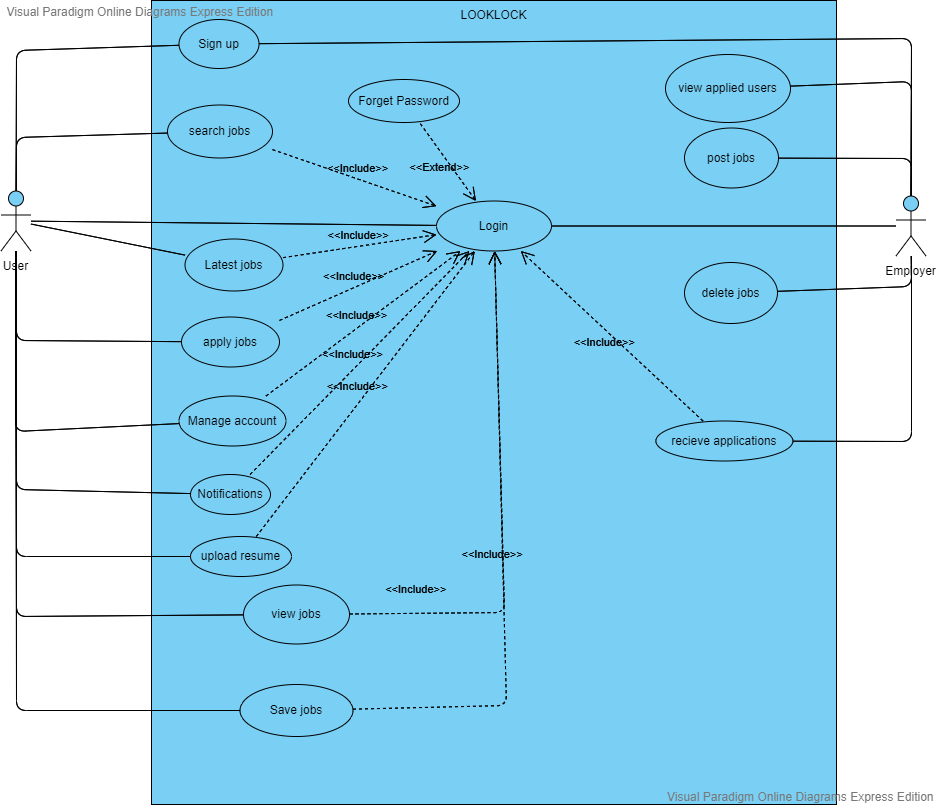


Fig:- Use Case-1

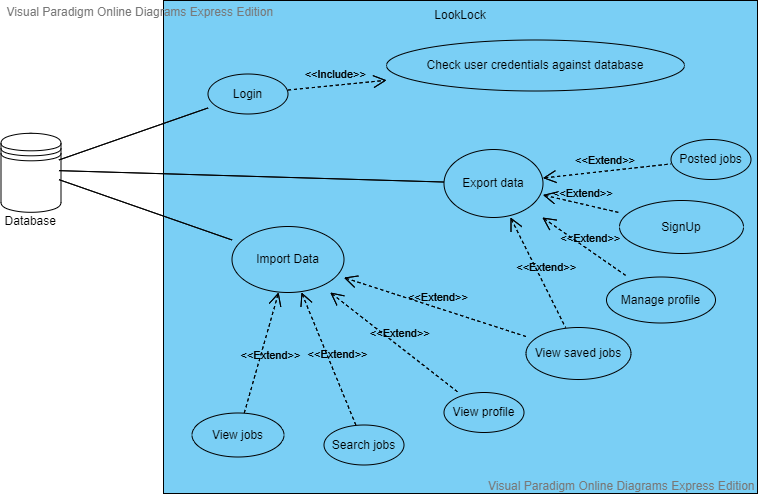


Fig:- Use case-2

* 1. **Use Case Scenarios**

|  |  |
| --- | --- |
| **Use case Scenarios Number** | **Scenario Name** |
| UC-1 | Register |
| UC-2. | Login |
| UC-3. | Edit profile |
| UC-4. | Search Jobs |
| UC-5. | View Jobs |
| UC-6. | Apply Jobs |
| UC-7. | Delete Posted Jobs |
| UC-8 | Logout |
| UC-9 | Settings |
| UC-10 | Post Jobs |
| UC-11 | Edit Jobs |

#### UC-1: Register

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-1 |
| Author(s): | Team |
| Version: | None |
| Name: | Register New User |
| Pre-Condition(s): | The user is non-registered and does not have an account on LOOK LOCK |
| Post- Condition(s): | The user is directed to the Login Page of the Mobile application. |
| Trigger: | The user has clicked on the Register button. |
| Normal Flow: | 1. User clicks on the Register button to create the user profile. 2. The user enters the details in the form and sets up the password. 3. A confirmation message is sent to the user on the user's email address. 4. User is redirected to the login page. |
| Alternate Flow: |  |
| Exceptional Flow(s): | **Exception:** There is an occurrence of internet connection failure.  1. The system will display the message “No Internet Connection, try Again”. |
| Related Actor(s): | **Primary**-Non-Registered User  **Secondary**-Admin |
| Related Use Case(s): | User Login |

Table 1. UC-1: Register New User

#### UC-2: Login

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-2 |
| Author(s): | Team |
| Version: | None |
| Name: | Login |
| Pre- Condition(s): | The Actor is registered and has an account on LOOK LOCK. |
| Post- Condition(s): | The Actor is directed to the home page specific to the user/admin. |
| Trigger: | The Actor has clicked on the Login button. |
| Normal Flow: | 1. Actor clicks on the Login button. 2. The actor enters the id and the password in the form. 3. The actor is directed to the homepage specific to the actor. |
| Alternate Flow: | 1. User clicks on “forget password”. 2. The user enters the email used for registration. 3. Password will be sent to the registered email address of the user. 4. The user is directed to the Login page. 5. The user enters the user name and the password in the form. 6. The user is directed to the homepage specific to the user. |
| Exceptional Flow(s): | **Exception:** The Actor enters a wrong password or id.  1. The system will prompt the message “Wrong User/admin id or password, try again”. |
| Related Actor(s): | **Primary**-User, Admin |
| Related Use Case(s): | Register |

Table 2. UC-2: Login

#### UC-3 – Edit Profile

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-3 |
| Author(s): | Team |
| Version: | None |
| Name: | Edit profile. |
| Pre- Condition(s): | 1. The user is registered and has an account on LOOK LOCK. 2. User logged in with the correct id and password. |
| Post- Condition(s): | The user can see his updated profile after editing.  . |
| Trigger: | The user has clicked on the Edit Profile in their dashboard. |
| Normal Flow: | 1. User clicks on the Edit Profile button. 2. User changes the personal information in the form. 3. User clicks on the submit button. 4. A confirmation message is displayed on the screen “Changes successfully saved” |
| Exceptional Flow(s): | **Exception:** The User can go back without edit personal information. |
| Related Actor(s): | **Primary**-User |
| Related Use Case(s): | Settings  Edit Profile |

Table 3. UC-3: Edit profile

#### UC-4: Job Search

|  |  |
| --- | --- |
| System: | A Mobile Application – Look Lock |
| Identifier: | UC-4 |
| Author(s): | Team |
| Version: | None |
| Name: | Search Job |
| Pre- Condition(s): | The user must be a registered user and job details must exist in the database. |
| Post- Condition(s): | The User can be able to view the job details and the requirements for the different jobs according to search. |
| Trigger: | The user has clicked on the “Search” button. |
| Normal Flow: | 1. The user clicks on the “Search” button 2. Search by title. 3. Clicks on the search option. |
| Exceptional Flow(s): | **Exception:** The User has given wrong details while searching the job. |
| Related Actor(s): | **Primary**-User |
| Related Use Case(s): | Login |

Table 4. UC-4: Search Jobs

#### UC-5: View Jobs

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-5 |
| Author(s): | Team |
| Version: | None |
| Name: | View Jobs |
| Pre- Condition(s): | User must exist and their details also must exist in the database. |
| Post- Condition(s): | User can save a particular job and apply. |
| Trigger: | The user has clicked on particular job. |
| Normal Flow: | 1. The user logged into the system using their credentials. 2. User clicks on the available jobs. 3. User clicks on the applied jobs. 4. User can see the details of the jobs. |
| Alternate Flow: | User can also use search option to view other type of jobs. |
| Exceptional Flow(s): | None. |
| Related Actor(s): | **Primary**- User |
| Related Use Case(s): | Login  Apply jobs |

Table.5: UC-5: View Jobs

#### UC-6: Apply Jobs

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-6 |
| Author(s): | Team |
| Version: | None |
| Name: | Apply Jobs |
| Pre- Condition(s): | User must be registered and fulfil all the requirements. |
| Post- Condition(s): | User will get confirmation of applied job. |
| Trigger: | The user has clicked on the “Apply” button. |
| Normal Flow : | 1. The User clicks on the “Apply” button. 2. Job will get applied. 3. User can view other jobs. |
| Alternate Flow: | User can save job for future without applying. |
| Exceptional Flow(s): | User can go back without applying job. |
| Related Actor(s): | **Primary**-Admin |
| Related Use Case(s): | Login  View Jobs |

Table 6: UC-6: Apply Jobs

#### UC-7: Delete Posted jobs

|  |  |
| --- | --- |
| System: | A Mobile Application – LookLock |
| Identifier: | UC-7 |
| Author(s): | Team |
| Version: | None |
| Name: | Delete Posted jobs |
| Pre- Condition(s): | Employer must exist and Posted job also must exist in the database. |
| Post- Condition(s): | Employer will be redirected to the list of the jobs and can see the updated details. |
| Trigger: | The Employer has clicked on “Delete”. |
| Normal Flow: | 1. The Employer clicks on the “Delete” button. 2. Employer deletes the job. 3. The job details will be deleted from the list of jobs. |
| Alternate Flow: | None |
| Exceptional Flow(s): | None. |
| Related Actor(s): | **Primary**-Employer |
| Related Use Case(s): | Login  Posted jobs |

Table 7: UC-7: Delete jobs

#### UC-8: Logout

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-8 |
| Author(s): | Team |
| Version: | None |
| Name: | Logout |
| Pre- Condition(s): | Actors must have their credentials registered in the database. |
| Post- Condition(s): | The actor is directed to the Main screen with user login and admin login buttons. |
| Trigger: | The actor has clicked on “Logout” in their dashboard |
| Normal Flow: | 1. User clicks on the Logout. 2. The system logouts the actor out of the system, 3. Actor will be redirected to main screen |
| Alternate Flow: | None |
| Exceptional Flow(s): | None. |
| Related Actor: | **Primary**- User |
| Related Use Case(s): | Login |

Table 8: UC-8: Logout

#### UC-9: Settings

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-8 |
| Author(s): | Team |
| Version: | None |
| Name: | Settings |
| Pre- Condition(s): | Users must have their credentials registered in the database. |
| Post- Condition(s): | The user is directed to the Settings page. |
| Trigger: | The actor has clicked on “Settings” in their dashboard |
| Normal Flow: | 1. User clicks on the Settings. 2. User can update their profile. 3. User can change the password. |
| Alternate Flow: | None |
| Exceptional Flow(s): | None. |
| Related Actor(s): | **Primary**- User |
| Related Use Case(s): | Login  Edit Profile |

Table 9: UC-9:Settings

#### UC-10:Post jobs

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-8 |
| Author(s): | Team |
| Version: | None |
| Name: | Post Jobs |
| Pre- Condition(s): | Employers must have their credentials registered in the database. |
| Post- Condition(s): | The user can view all the jobs posted on the application**.** |
| Trigger: | The employer has clicked on “post jobs” in their dashboard |
| Normal Flow: | 1. Employer can post jobs. 2. Admin will give approval to post the job. 3. User can view the jobs. |
| Alternate Flow: | None |
| Exceptional Flow(s): | None. |
| Related Actor(s): | **Primary**- employer |
| Related Use Case(s): | Login |

Table 10: UC-10: Post jobs

#### UC-11:Edit jobs

|  |  |
| --- | --- |
| System: | A Mobile Application – LOOK LOCK |
| Identifier: | UC-8 |
| Author(s): | Team |
| Version: | None |
| Name: | Edit Jobs |
| Pre- Condition(s): | Employers must have their credentials registered in the database. |
| Post- Condition(s): | The user can view all the edited jobs posted on the application**.** |
| Trigger: | The employer has clicked on “Edit jobs” in their dashboard |
| Normal Flow: | 1. Employer can click on the Edit job button. 2. Employer can add new jobs by title, description, location and salary. |
| Alternate Flow: | None |
| Exceptional Flow(s): | None. |
| Related Actor(s): | **Primary**- admin, employer |
| Related Use Case(s): | Login |

Table 13: UC-13: Edit jobs

* 1. **Class Diagram**

a class diagram in the [Unified Modelling Language (UML)](https://en.wikipedia.org/wiki/Unified_Modeling_Language) is **a type of static structure diagram** that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

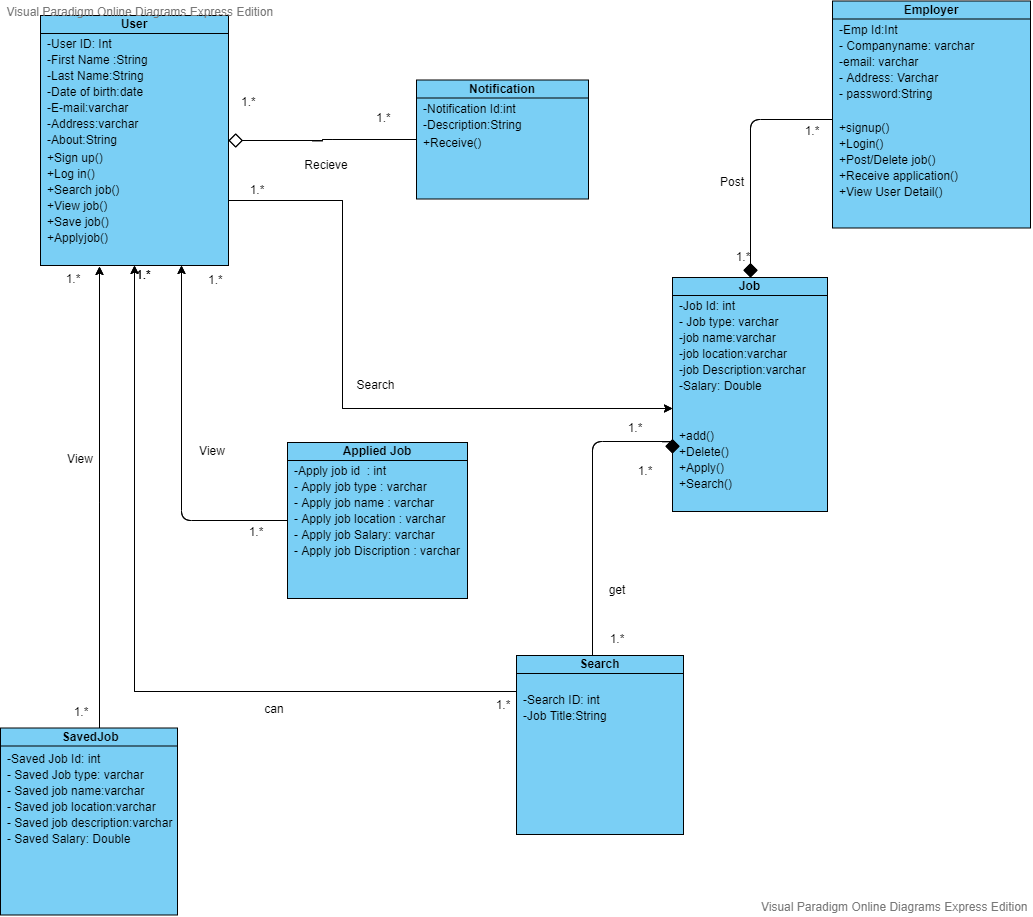


Fig: Class Diagram

* 1. **Entity Relationship Diagram**

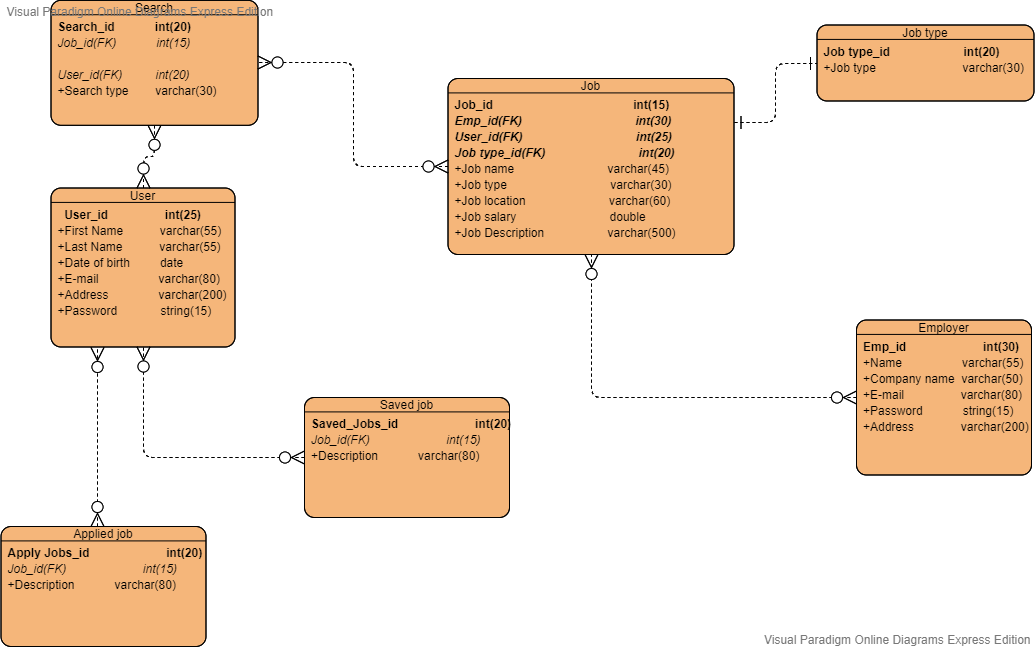
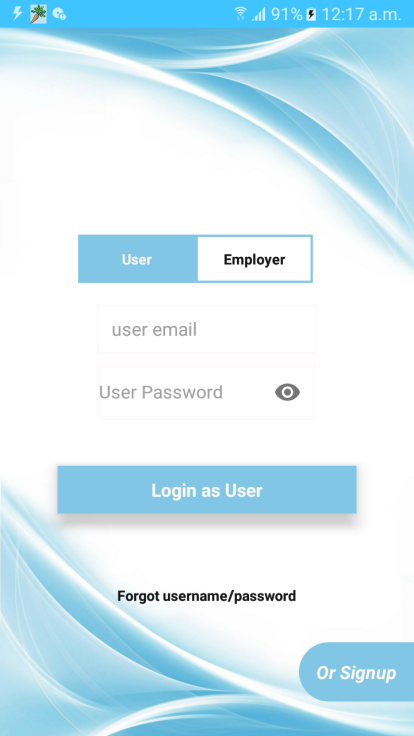


Fig: Entity Relationship Diagram

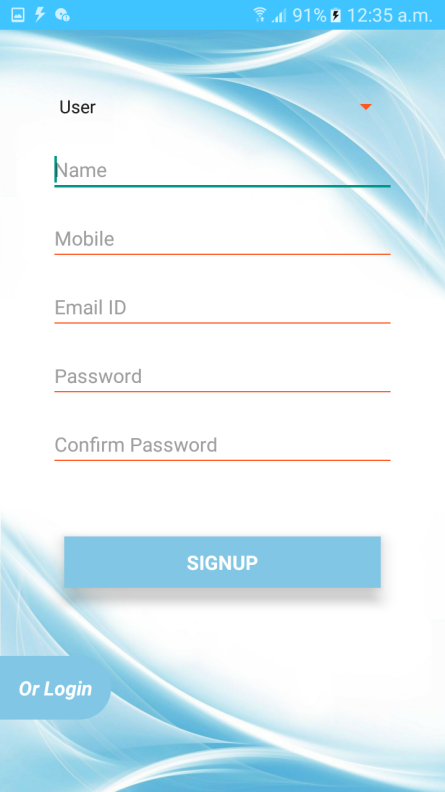
* 1. **Screens**
* **Splash Screen**

****

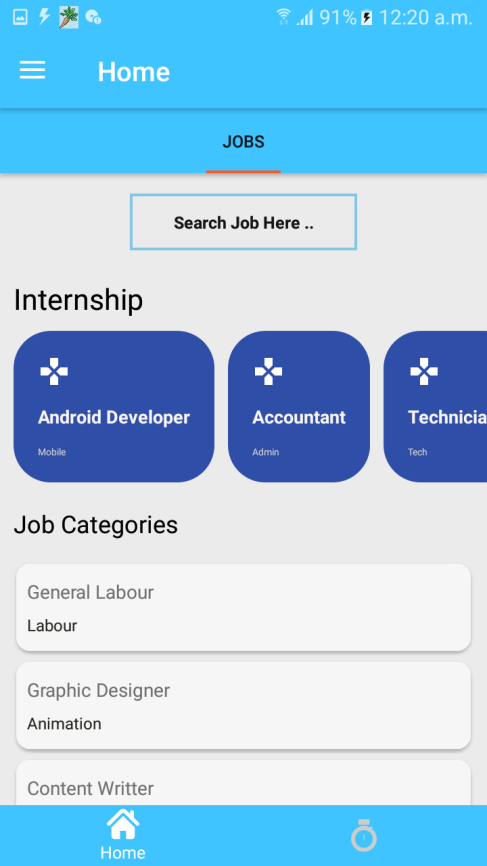
* **Login Screen**

****

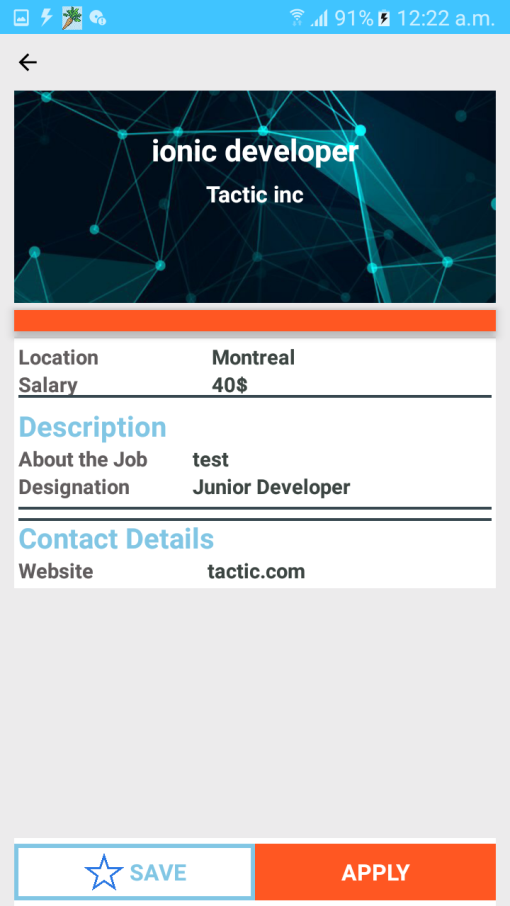
* **Sign up**



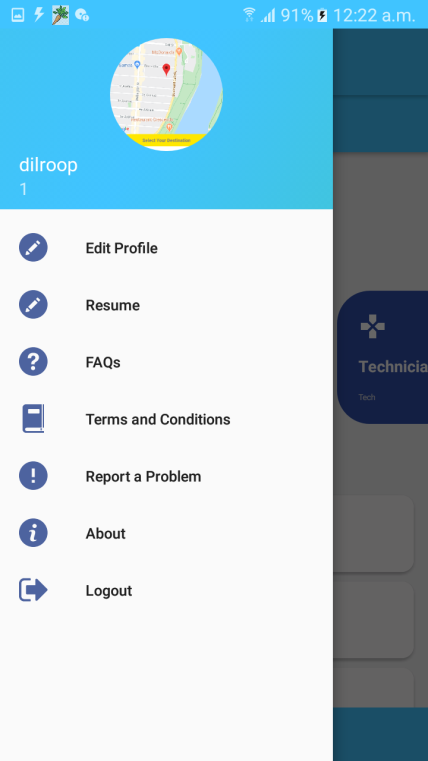
* **Home screen**

****

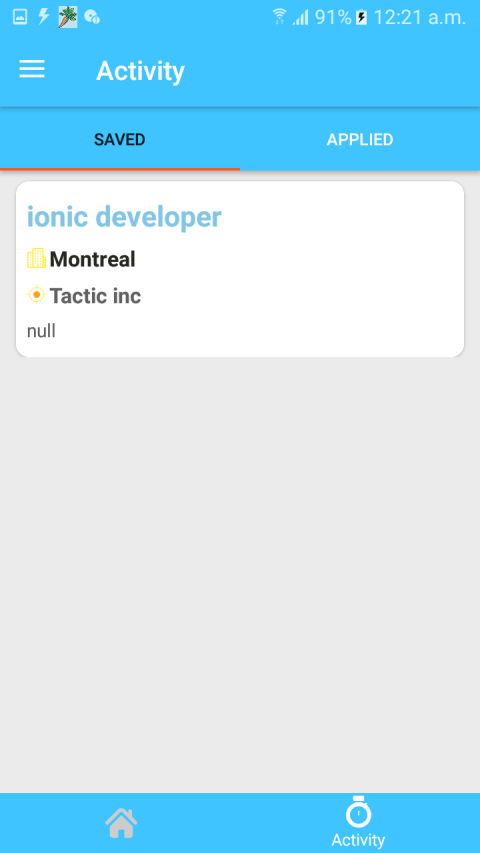
* **Job detail Screen**

****

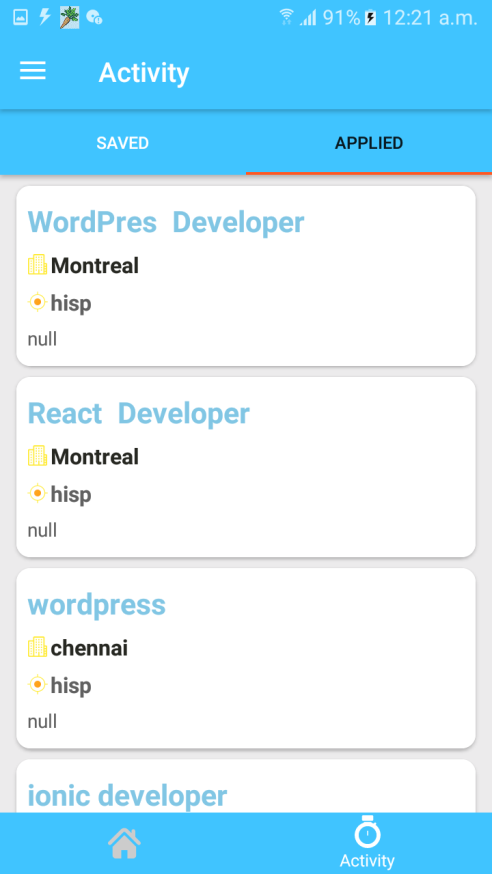
* **Menu screen**

****

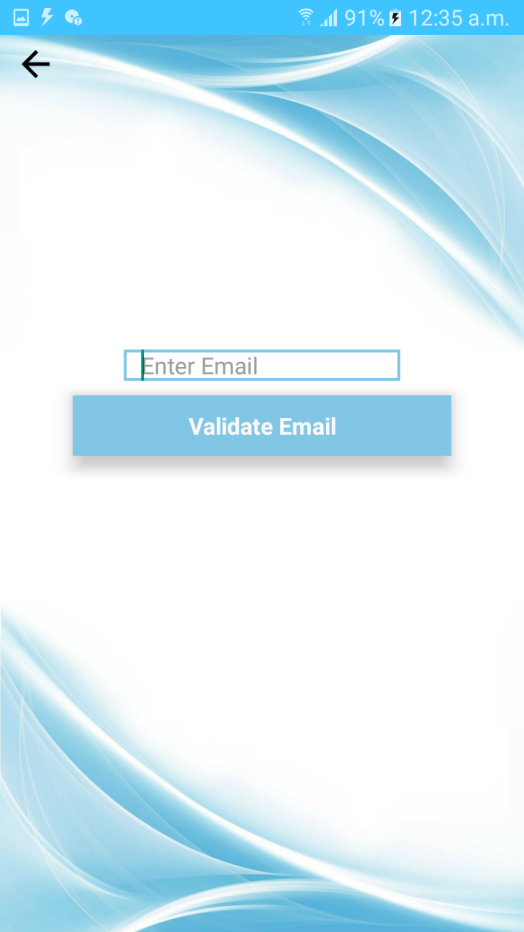
* **Saved job Screen**

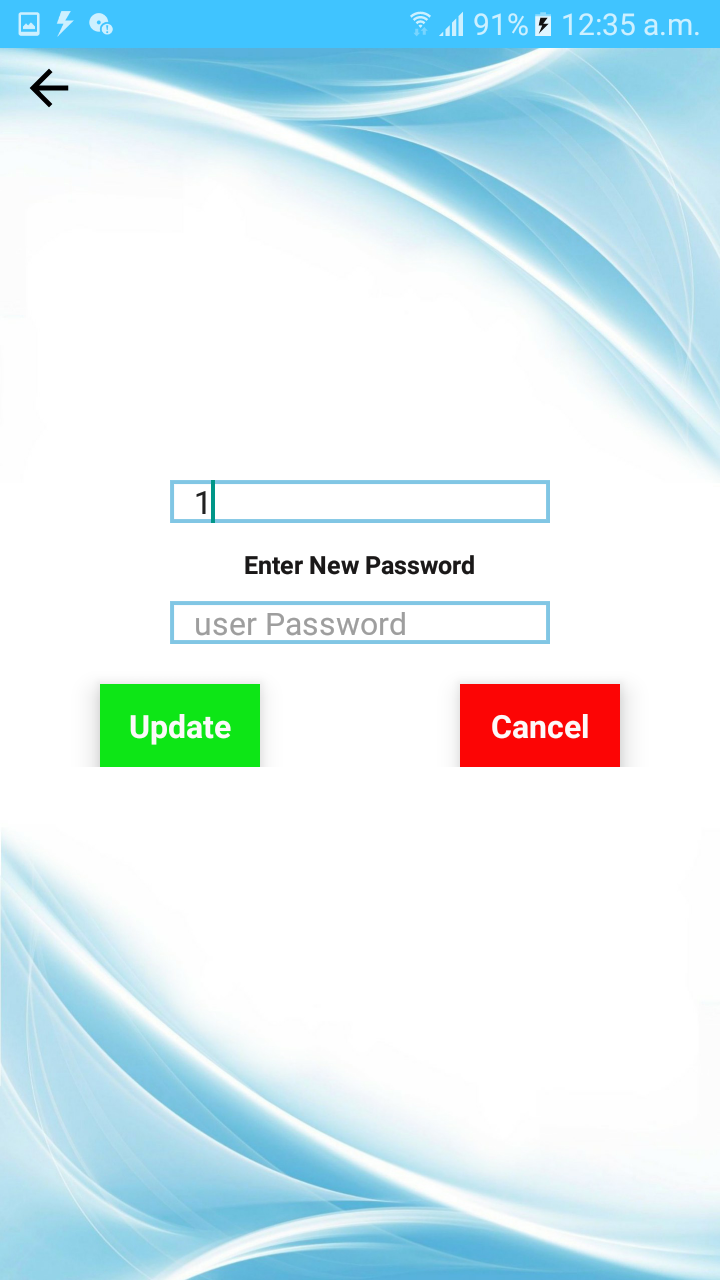
****

* **Applied jobs Screen**

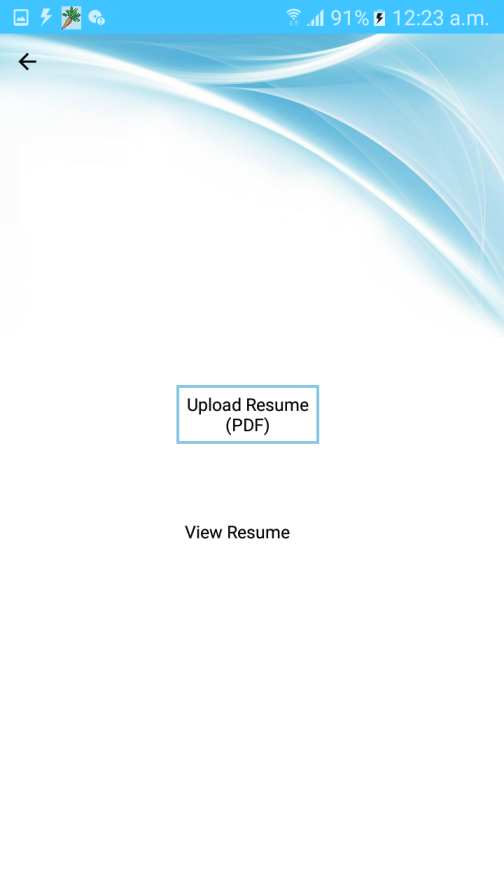
****

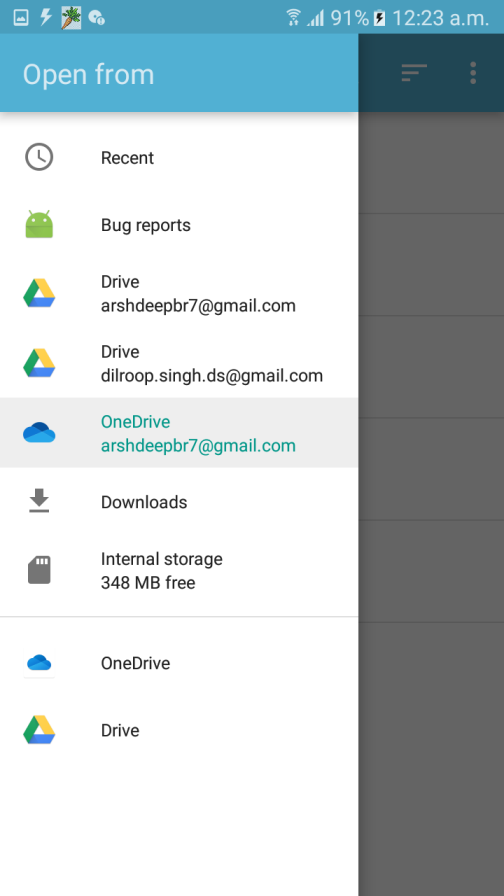
* **Forget Password**

****

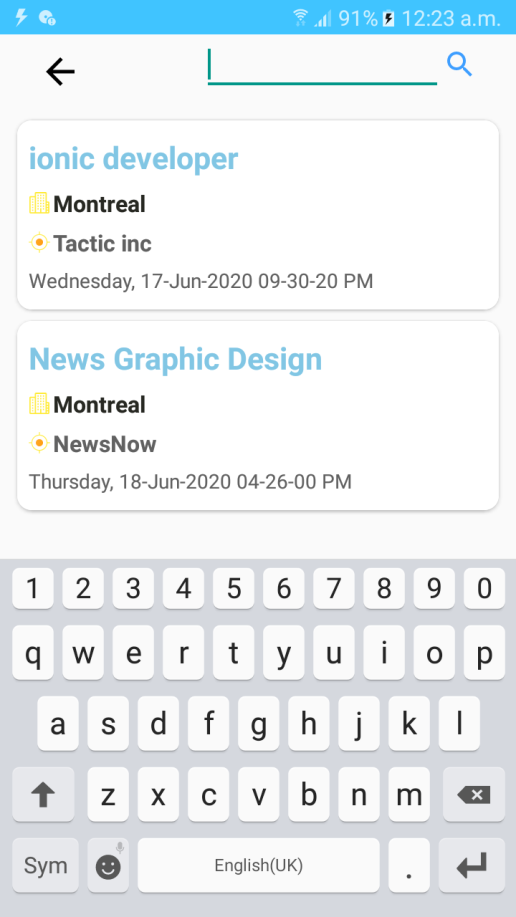
****

* **Update resume Screen**

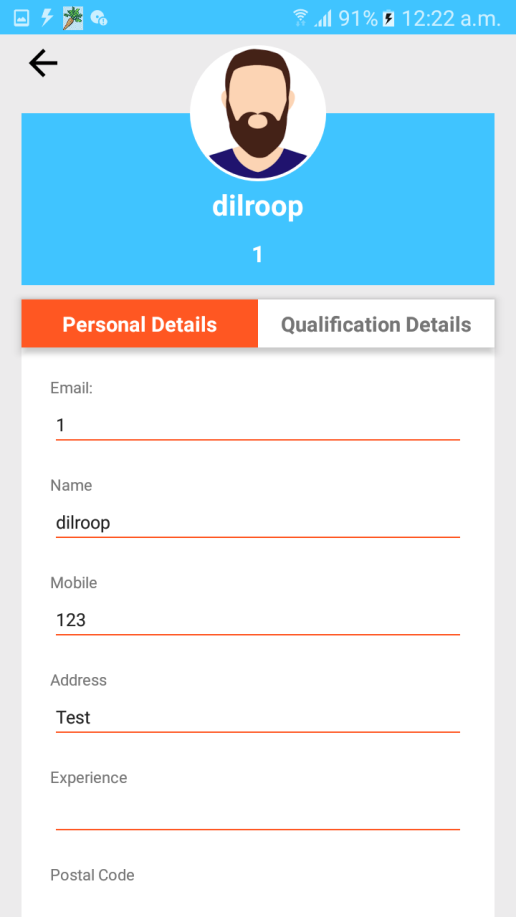
****

****

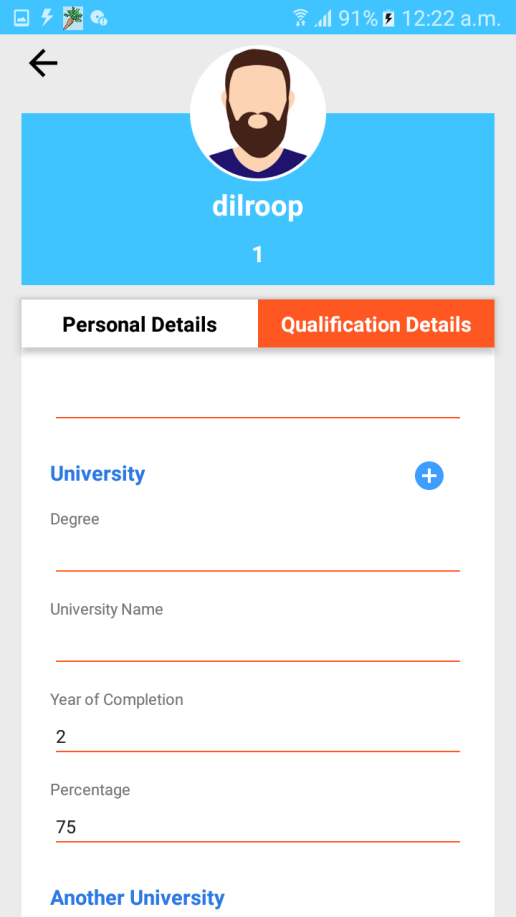
* **Search Screen**

****

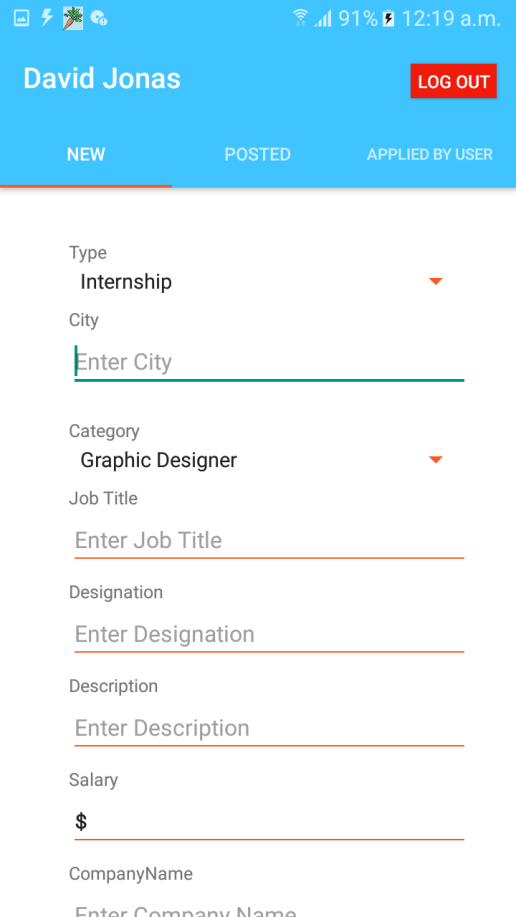
* **Edit profile- personal detail Screen**

****

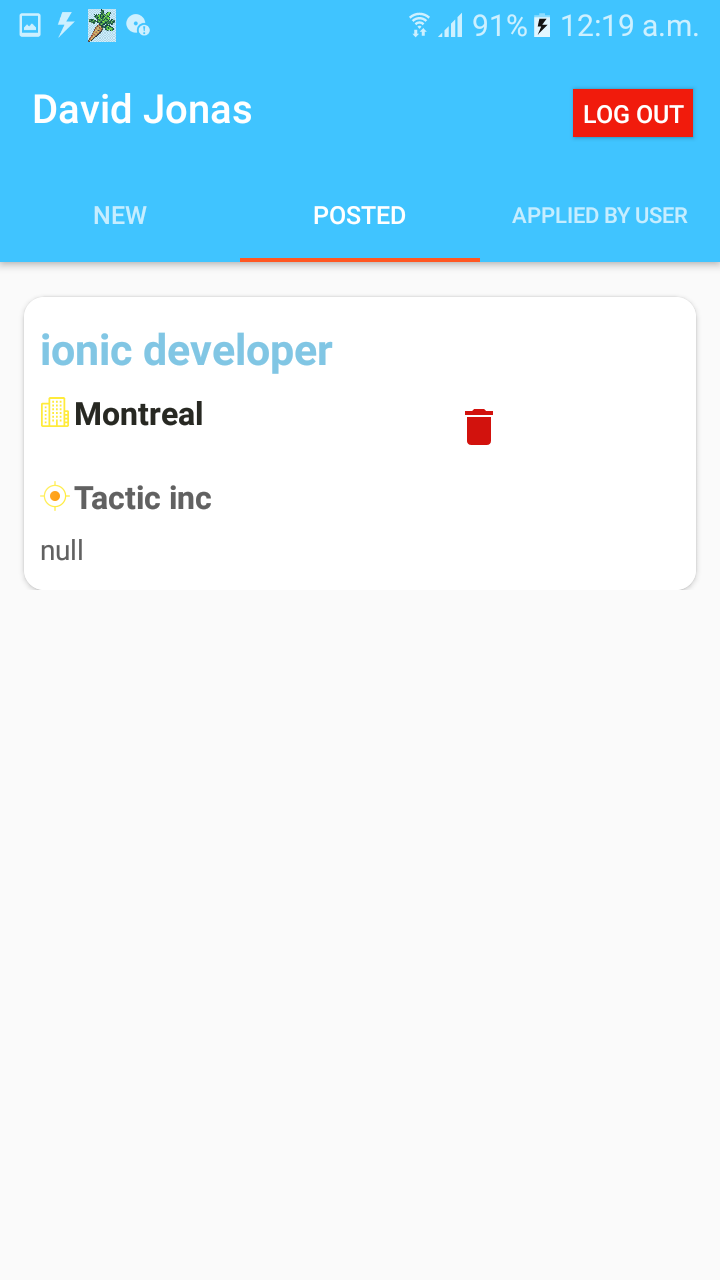
* **Edit profile- qualification detail Screen**

****

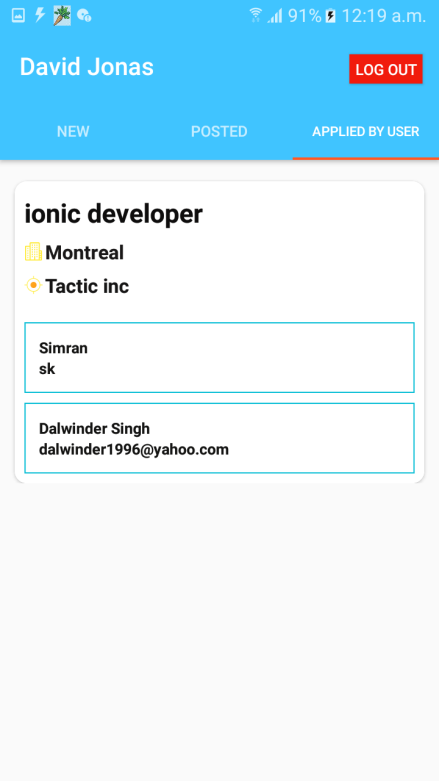
* **Employer`s home Screen**

****

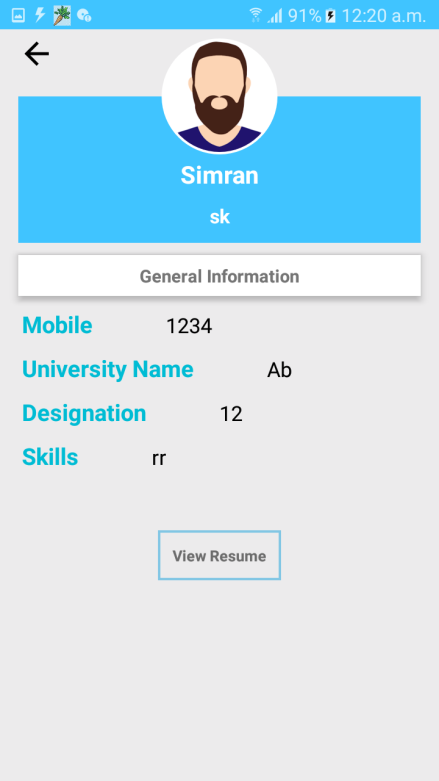
* **Posted job Screen**

****

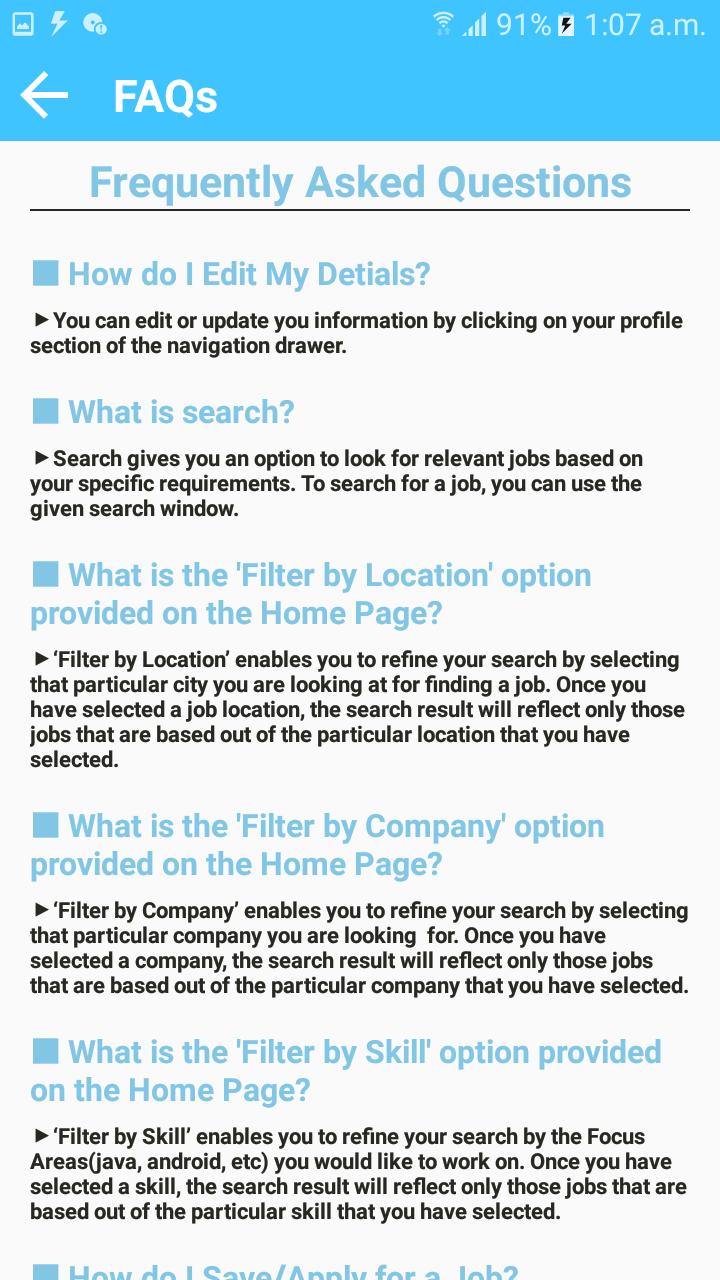
* **Applied by User Screen**

****

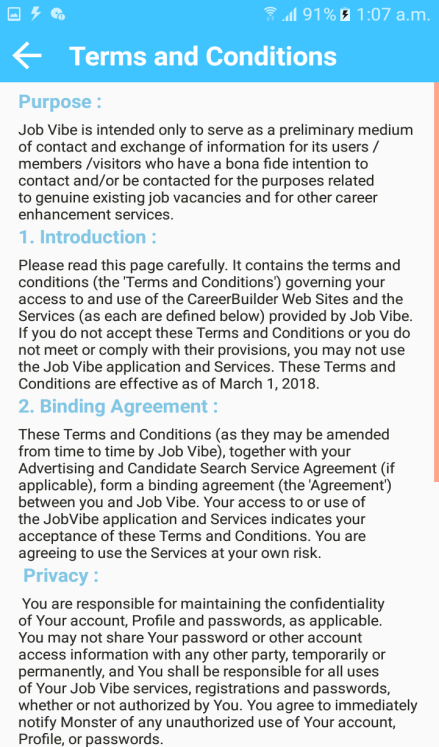
* **User detail Screen**

****

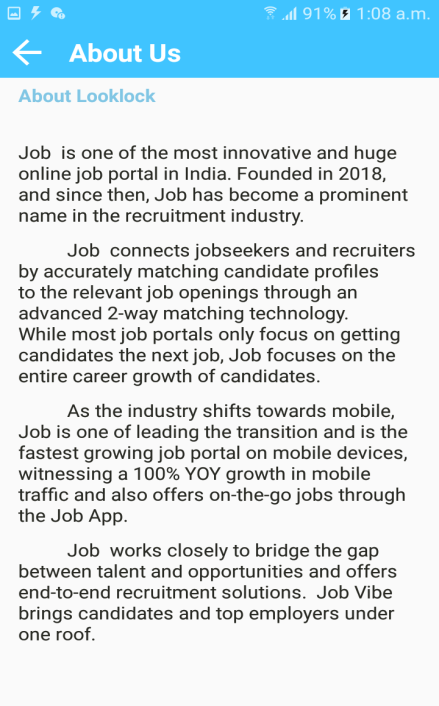
* **FAQ Screen**

****

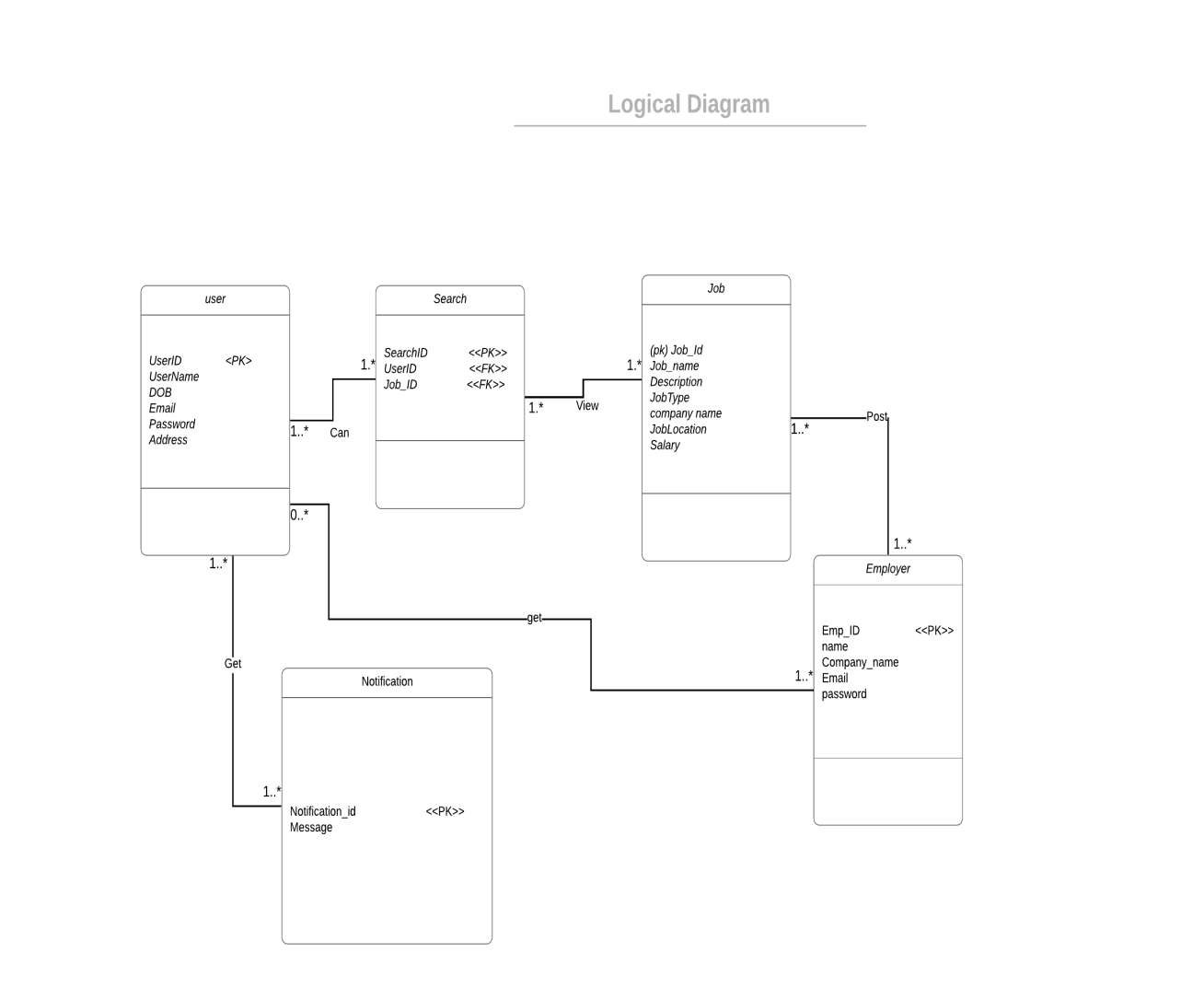
* **Terms and conditions Screen**

****

* **About Screen**

****

1. **Database**
   1. **Logical Model**

****

* 1. **Database Script**

create database JobFinderDB;

Use JobFinderDB;

...................................

// Table Creation

..........................

1.)create table User(

User\_id int(10) not null,

User\_name varchar(60),User\_address varchar(100),

User\_phone varchar(30),

User\_email varchar(50),

User\_password varchar(30),

User\_experience varchar(30),

User\_city varchar(30),

User\_postalcode varchar(30),

User\_dob varchar(30),

User\_gender varchar(30),

User\_degree varchar(30),

User\_fos varchar(30),

primary key (User\_id)

);

insert into User(User\_id,User\_fname,User\_address,User\_phone,User\_email,User\_password,User\_experience,User\_city,User\_postalcode,User\_dob,User\_gender,User\_degree,User\_fos)

values(100,"Simranjeet kaur","rue sherley, lasalle","+1514000000","s@gmail.com","ab1997",”2years”,”lasalle”,”H8N001”,”07/09/1992”,”Female”,”BCA”,”computer Science”),

values(101,"Gurjot kaur",”Newman Blvd, montreal","+1514000001","g@gmail.com","1996",”4years”,”Montreal”,”H4PA06”,”07/09/1996”,”Female”,”BTech”,”computer Science”),

values(102,"Arshdeep kaur","St. Jospeh street,quebec","+1514000002","a@gmail.com","19abc",”1years”,”quebec”,”L8K0H1”,”23/05/1994”,”Female”,”MCA”,”computer technician”),

values(104,"Manpreet singh","145, annapot,toronto","+1514000003","m@gmail.com","kol95",”4years”,”toronto”,”A6N053”,”07/12/1997”,”Male”,”BSC”,”Mathematics”));

Select \* from User;

2.)create table User\_qualification(

Userqu\_id int(10) not null,

Userqu\_board varchar(60),Userqu\_school varchar(70),

Userqu\_completionyear varchar(30),

Userqu\_university varchar(80),

Userqu\_uniname varchar(80),

Userqu\_unipercentage varchar(30),

Userqu\_experience varchar(100),

Userqu\_designation varchar(100),

Userqu\_skills varchar(200),

primary key (Userqu\_id)

);

3.)create table Employer(

Emp\_id int(10) not null,

Emp\_name varchar(60),

Emp\_Companyname varcha(60) ,

Emp\_phone varchar(20),

Emp\_email varchar(30),

Emp\_password varchar(30),

primary key (Emp\_id)

);

insert into Employer(Emp\_id,Emp\_name,Emp\_Companyname,Emp\_phone,Emp\_email,Emp\_password)

values(200,"Jeet","GoodFood","+1438000000","gfcareers@gmail.com","1112\*"),

values(201,"Jot","Arima","+151400232","arimacareers.com","2233@"),

values(202,"Deep","Public Sac","+151489898","psjobs@gmail.com","4455$"),

values(204,"Preet","Mastro","+1438000003","talent@mastrogmail.com","#6677"),

values(205,"Inder","Mcdonald's","+1114000007","mcdonalds@gmail.com","&6688"),

values(206,"Pal","Tim Hortons","+15140000278","timjobs@gmail.com","!9999");

Select \* from Employer;

4.)create table jobs(

job\_id int(10) not null,

job\_type varchar(60),

job\_name varchar(60) ,

job\_location varchar(30),

job\_salary varchar(30),

job\_desc varchar(500),

primary key (job\_id)

);

insert intojobs(job\_id ,job\_type,job\_name,job\_location,job\_salary,job\_desc)

values(1001,"abc","def",Montreal","$100",”itis def job by abc company with 100 dollar pay. ”)

values(1002,"abc","def",Toronto","$200",”itis def job by abc company with 200 dollar pay. ”)

values(1003,"abc","def",calgary","$00",”itis def job by abc company with 500 dollar pay. ”)

values(1006,"abc","def",Lasalle","$700","itis def job by abc company with 700 dollar pay. ”);

Select \* from jobs;

5 .)create table Savedjobs(

savejob\_id int(10) not null,

job\_type varchar(60),

job\_name varchar(60) ,

job\_location varchar(30),

job\_salary varchar(30),

job\_desc varchar(500),

primary key (savejob\_id)

foreign key (job\_id)

);

insert into Savedjobs(Savejob\_id ,job\_type,job\_name,job\_location,job\_salary,job\_desc)

values(121,"abc","def",Montreal","$100",”itis def job by abc company with 100 dollar pay. ”,”1001”),

values(122,"abc","def",Toronto","$200",”itis def job by abc company with 200 dollar pay. ”,”1002”);

6 .)create table appliedjobs(

applyjob\_id int(10) not null,

job\_type varchar(60),

job\_name varchar(60) ,

job\_location varchar(30),

job\_salary varchar(30),

job\_desc varchar(500),

primary key (savejob\_id)

foreign key (job\_id)

);

insert into appliedjobs(applyjob\_id ,job\_type,job\_name,job\_location,job\_salary,job\_desc)

values(021,"abc","def",Montreal","$100",”itis def job by abc company with 100 dollar pay. ”,”1001”),

values(022,"abc","def",Toronto","$200",”itis def job by abc company with 200 dollar pay. ”,”1002”);

**References**

* <https://developer.android.com/docs>
* <https://www.visual-paradigm.com/guide/uml-unified-modeling-language>
* <http://agilemodeling.com/artifacts/classDiagram.htm>
* <https://android.googlesource.com/platform/frameworks/base/+/master/core/java/android/text/>
* Notes from previous semesters.