BA-IKHTIAR: Android Based Job Application for Transgender community.

PROJECT- II DOCUMENTATION

SUBMITTED BY:

AMMARA RASHEED

SUPERVISED BY:

MEMOONA AFSHEEN MALIK

Department of Computer Sciences

Quaid-i-Azam University

Islamabad

Abstract

Ba Ikhtiar is an android based job application for transgender community where they can register themselves by filling some data forms and create there profiles and can search jobs based on their skills and apply for them. They can also contact the advertisers through the app. Similarly advertisers can register themselves on the app and can post jobs and search for applicants and contact them. Android Studio is used for the development of this application in Java and web service is created through Spyder in Python and xampp server is used.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Problem

Generally, it has been observed that there is ignorance for „third gender‟‟ in a third world country like Pakistan. Massive rejections are faced by transgender in almost all the parts of Pakistan. They are often talented in certain fields but due to lack of their acknowledgement in society, they can‟t get any jobs. A myth seems to have formed that they cannot do any work except for singing and dancing despite being educated resulting in the downwards spiral of degradation.

1.2 Idea

It seems that there are certain people who have jobs for this community but due to lack of communication platform they can‟t find any so idea is to develop an android based mobile application for the first time in Pakistan targeting transgender community by helping them to find jobs online in their field of interest. We need to be the voice that fights for their rights and makes living for them less painful.

1.3 Project Overview

Transgender with various skills will register themselves on our application by creating their profile, so on client side advertisers will post their jobs and can also search profiles of the users based on their skills and location like if a company wants an applicant living in vicinity of Islamabad or Rawalpindi than our app provides a filtered search of all the registered users profiles based on their location so from that the client can choose the one suitable for their job.

1.4 Project Deliverables

Table 1 Project Deliverables

Software Project Management Plan (SPMP)	1st deliverable
Software Requirements Specification (SRS)	1st deliverable

Software Design Description (SDD)	2nd deliverable
Software Test Documentation (STD)	2nd deliverable

1.5 Project Organization

1.5.1 Software Process Model

In the implementation of this project I prefer to use V Process Model which shows execution of processes in a sequential manner, also known as Validation and Verification model. It is an extension of waterfall model where each development phase is followed by a testing phase hence helps to save a lot of time than waterfall model. In this way it avoids the downward flow of defects called as Proactive Defect tracking. It is used for the projects where problems are clearly defined.

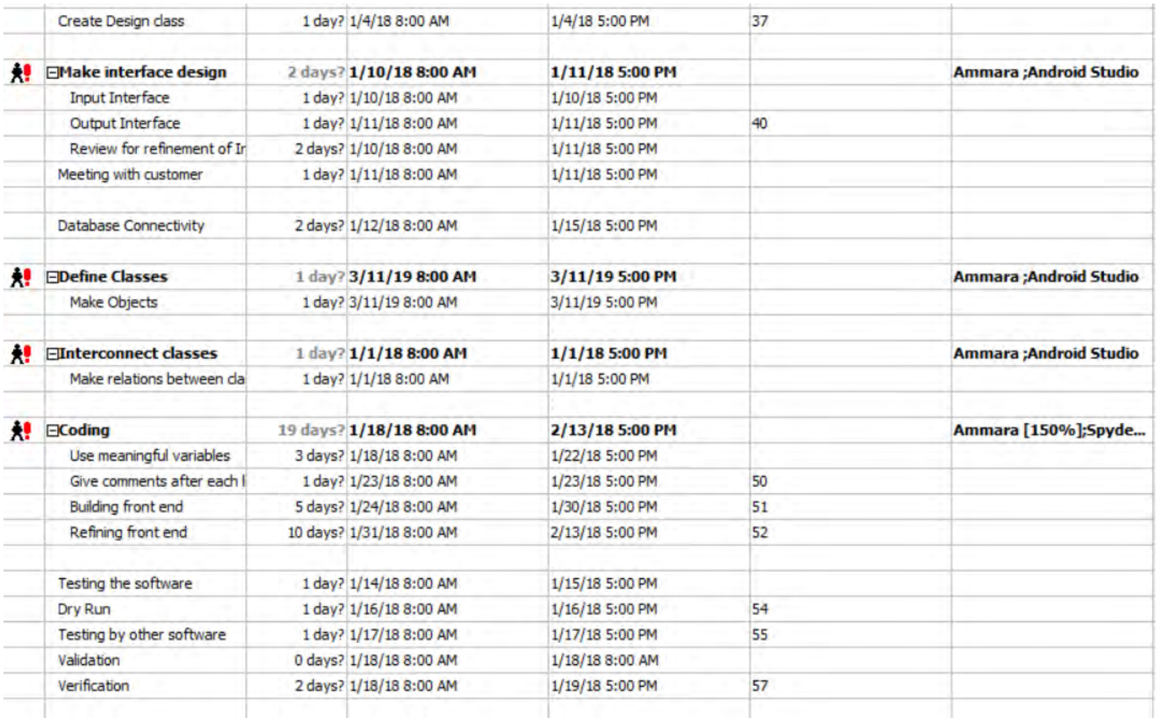
1.5.2 Roles and Responsibilities

Implementation of this application and all other work would be done by Ammara Rasheed BSCS 8th under the supervision of Mam Memoona Afsheen Malik.

1.5.3 Tools and Techniques Used

I will use Android studio to build this application using JAVA and database will be stored at SQL Server and web service is created through Spyder using Python.

1.6 Software Project Management Plan (SPMP):



	Create Design class	1 day?	1/4/18 8:00 AM	1/4/18 5:00 PM	37	
	⊟Make interface design	2 days?	**1/10/18 8:00 AM**	**1/11/18 5:00 PM**		**Ammara ;Android Studio**
	Input Interface	1 day?	1/10/18 8:00 AM	1/10/18 5:00 PM		
	Output Interface	1 day?	1/11/18 8:00 AM	1/11/18 5:00 PM	40	
	Review for refinement of Ir	2 days?	1/10/18 8:00 AM	1/11/18 5:00 PM		
	Meeting with customer	1 day?	1/11/18 8:00 AM	1/11/18 5:00 PM		
	Database Connectivity	2 days?	1/12/18 8:00 AM	1/15/18 5:00 PM		
	⊟Define Classes	1 day?	**3/11/19 8:00 AM**	**3/11/19 5:00 PM**		**Ammara ;Android Studio**
	Make Objects	1 day?	3/11/19 8:00 AM	3/11/19 5:00 PM		
	⊟Interconnect classes	1 day?	**1/1/18 8:00 AM**	**1/1/18 5:00 PM**		**Ammara ;Android Studio**
	Make relations between cla	1 day?	1/1/18 8:00 AM	1/1/18 5:00 PM		
	⊟Coding	19 days?	**1/18/18 8:00 AM**	**2/13/18 5:00 PM**		**Ammara [150%];Spyde...**
	Use meaningful variables	3 days?	1/18/18 8:00 AM	1/22/18 5:00 PM		
	Give comments after each l	1 day?	1/23/18 8:00 AM	1/23/18 5:00 PM	50	
	Building front end	5 days?	1/24/18 8:00 AM	1/30/18 5:00 PM	51	
	Refining front end	10 days?	1/31/18 8:00 AM	2/13/18 5:00 PM	52	
	Testing the software	1 day?	1/14/18 8:00 AM	1/15/18 5:00 PM		
	Dry Run	1 day?	1/16/18 8:00 AM	1/16/18 5:00 PM	54	
	Testing by other software	1 day?	1/17/18 8:00 AM	1/17/18 5:00 PM	55	
	Validation	0 days?	1/18/18 8:00 AM	1/18/18 8:00 AM		
	Verification	2 days?	1/18/18 8:00 AM	1/19/18 5:00 PM	57	

Figure 1 Software Project Management Plan

CHAPTER 2

SOFTWARE REQUIREMENTS SPECIFICATIONS

2.1 Scope

We are going to develop an application named “Ba-Ikhtiar” which provides job facilities to the transgender. This app involves two kind of users.

1. User (transgender only)
2. Advertiser or job holder.

2.1.1 User

- User have to log in first if registered otherwise user have to sign up, which requires your basic info, location, educational and job experience and your skills.
- User can also search for jobs in their area.
- User can make contact to the job provider.

2.1.2 Advertiser

- Similarly, advertiser have to log in if registered otherwise have to sign up first with the name of their organization (if any) or their location and job requirement.
- They can search for users.
- From which they can select the one suitable for their job.
- They can contact the user.
- If the user is willing to do the job or they are jobless than they can pin back otherwise advertiser have to search again.

2.2 Objective

Main objective of the application is to provide jobs to the transgender community without being humiliating, in a respectful way. And for advertisers, goal is to find the reliable person better suited for the job by saving time.

2.3 DEFINATION, ACRONYMS AND ABBRIVIATIONS

Table 2 Acronyms and Abbreviations

Acronyms	**Abbreviations**
User	Transgender only, Advertiser
SPMP	Software Project Management Plan

2.4 OVERALL DESCRIPTION

I have defined product features and operations performed by the application. User first have to register themselves, whenever they visit my app they have to log in first with their email and password to perform any operation. For registration a form will appear, they have to fill in the form comprising their complete information. It displays user profiles and advertisements for all kind of jobs.

2.5 PRODUCT PERSPECTIVE

Ba-Ikhtiar is totally an independent application. It is only for transgender community. It has three active actors i.e. transgender, admin and advertiser. Admin of the system can register both the transgender and the advertiser. Whenever they register themselves a password is created, which is compulsory for log in. Admin can update skills of the users and maintain records in database.

2.5.1 System Interfaces

Ba-Ikhtiar is an Android based application. It is used by the transgender community who are registered at our app to find jobs. It uses a server which store data in database. User can use tablet or android phones to access this application and after logging in they can perform any operation.

2.5.2 User Interfaces

User interface defines how a user can interact with the application so in order to use this application to perform any function, user requires an email and password which must be authenticated. A first-time user of the mobile application should see the log-in and registration page when he/she opens the application. If the user has not registered, he/she should be able to do that on the log-in page. If the user is not a first-time user, he/she should be able to see the search page after just logging in when the application is opened then the user chooses the type of search they want to conduct. Every user should have a profile page where they can edit their e-mail.

2.5.3 Software Interfaces

Our application can interact with Google Location Services in order to perform a filtered search for jobs or applicants in a particular area. Similarly, it can interact with the database in order to get the information of the jobs of interest or required applicants.

2.5.4 Hardware Interfaces

Since the mobile application don‟t have any designated hardware, it does not have any direct hardware interfaces. The location is managed by the Google Locations Services application, mobile phone should be able to connect to GPRS and the hardware connection to the database server is managed by the underlying operating system on the mobile phone and the web server.

2.5.5 Communication Interfaces:

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems

2.6 Product Functions

- This software helps the user to search for jobs in their field of interest.
- Search will provide the filtered results of all the users/advertisement.

- Similarly, it can help advertisers to search for employees suitable for their job.
- System can also perform the following function:

1. User-login
2. Register user
3. Manage profile
4. View profile
5. View Advertisement
6. Search user profile
7. Search Advertisement
8. Add Advertisement
9. Sign out
10. Send Message
11. Wish list
12. Chat
13. Apply for job
14. Update skills

2.7 User Characteristics

Users are assumed to have basic knowledge of the mobiles and Internet browsing, they must have the know-how of English language and can read and write it. The proper user interface, online help and the guide to install and maintain the application must be sufficient to educate the users on how to use the application without any problems.

2.8 Assumptions and Dependencies

The users have sufficient knowledge of mobile/tablets. Mobile should have Internet access and Internet server capabilities. The users know the English language, as the user interface will be provided in English. The product can access the database.

2.9 Specific Requirements

There are some functional and nonfunctional requirements for the software application.

2.9.1 Functional Requirements

System can validate all the correct emails and create the profiles. It can allow to add advertisements from the registered users only. System shall allow admin to manage the profiles of users, advertisements and requests. It can allow all users to modify their data whenever needed and permit them to search for applicants/jobs. System does not store any data regarding problems.

2.9.2 Non-Functional Requirements

Nonfunctional requirements are basically software system attributes which are not directly involved in the system. Following are some of the nonfunctional requirements:

2.9.2.1 Reliability

This application is reliable and responds well to all valid queries, system process the query and display the related result within no time. System will only send non ambiguous results.

2.9.2.2 Maintainability

For any type of system crash due to virus or any operating system issue there is a backup of data at the database to avoid data loss.

2.9.2.3 Security

As far as security requirement is concerned as there are only registered users so the one who is not a user cannot see the activities going on and necessary data is being showed at profile at first after choosing and contacting the applicant further data can be shown according to privacy policy.

2.9.2.4 Availability

This is a 24/7 system. Whenever you have the internet access you can perform the desired operations so if a user can make query, database can send the results.

2.9.2.5 Portability

As it is an android application so it will run only on Android phones only.

2.9.2.6 Performance

Load time for user interface screens is no longer than some milliseconds. Log in verification shouldn‟t take more than 1 or 2 seconds similarly for search queries system will display the related result in no time.

2.10 Use Case Diagram

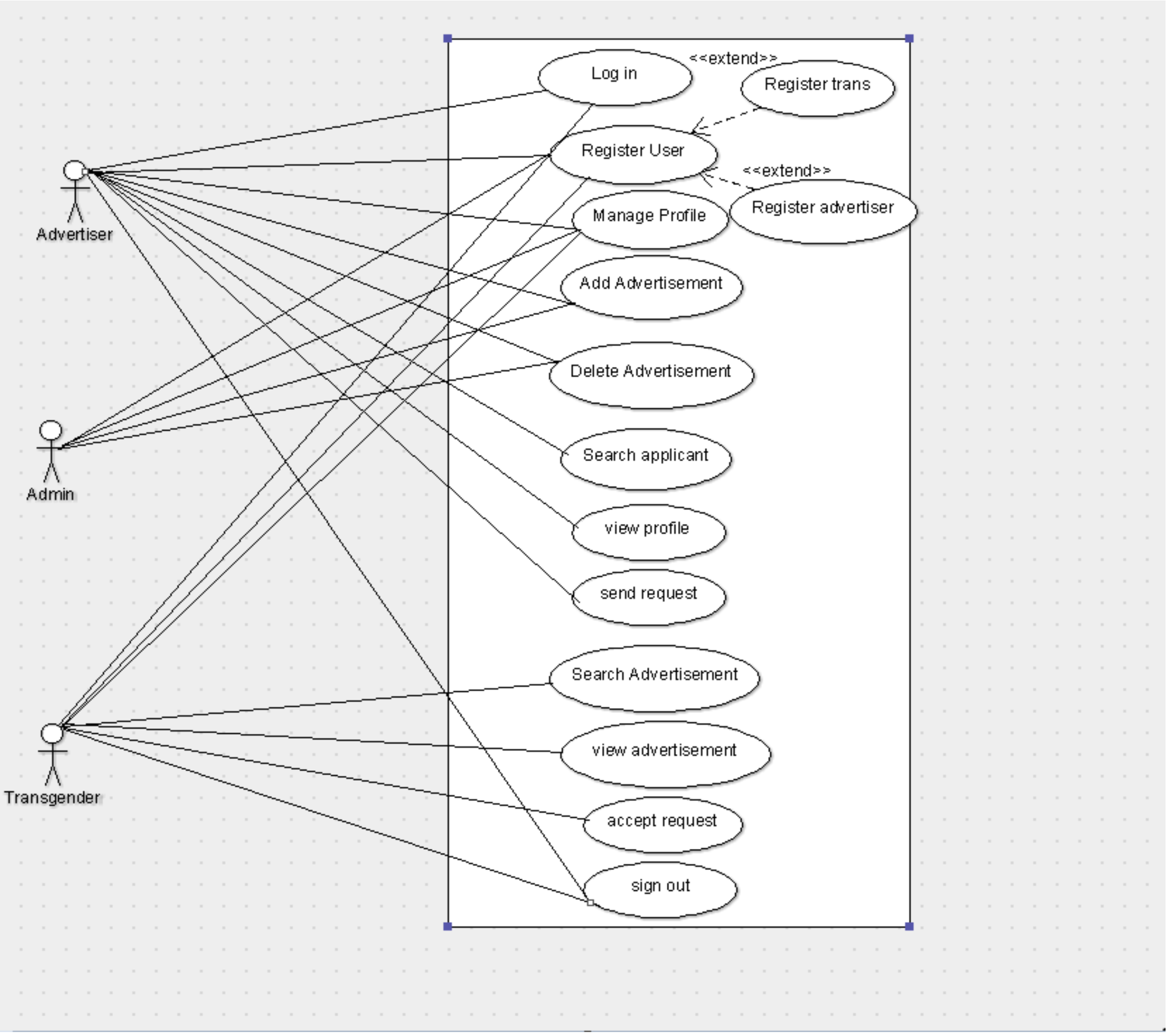


Figure 2 Use Case Diagram

2.11 Use Case Description

A use case diagram is a graphic depiction of the interaction among the elements of the system. A list of use cases mentioned in diagram are described below in detail so that we can look more precisely how a user can interact with system. There are three main actors of this system, two of them are primary and one is secondary.

A primary user is the one whose tasks are getting fulfilled by the system while a supporting actor in a use case in an external actor that provides a service to the system under design. It might be a high-speed printer, a web service, or humans that have to do some research and get back to us.

- **Use Case: Login**

Table 2.1 UC-1: Login

UC-1: Login	
Primary actor	Applicant, Advertiser
Pre-Condition	User must be registered
Post condition	User successfully logged in.
Main success scenarios	1. User selects log in option. 2. System asks to enter email and password 3. User enters valid email and password correctly. 4. System will check email and password. 5. User successfully logged in.

- **Use Case: Register Applicant**

Table 2.3 UC-2: Register Applicant

UC-2: Register Applicant	
Primary actors	Applicants
Pre-condition	Applicant wants to register for this application and applicant must be a transgender.
Post-condition	User Successfully registered
Main success scenarios	1. User selects register as applicant option. 2. Applicant form appears 3. system asks to enter required information. 2. Applicant fills in the required information, 3. Applicant submits form. 4. Applicant successfully registered,
Alternate scenarios	System fails at any time 1. System gets back to previous page and discard all information. 3a) Applicant skips any information or entered incorrect information. 1. System asks to fill all information.

- **Use Case: Register Advertiser**

Table 2.4 UC-3: Register Users

UC-3: Register Users	
Primary actors	Advertiser
Pre-condition	Advertiser wants to register for this application.
Post-condition	Advertiser Successfully registered
Main success scenarios	1. Advertiser selects register as advertiser option. 2. Advertiser form appears. 3. System asks to enter required information. 4. Advertiser fills in the required information, 5. Advertiser submits form. 6. Advertiser successfully registered.
Alternate scenarios	System fails at any time 2. System gets back to previous page and discard all information. 3.a) Advertiser skips any information or entered incorrect information. 1. System asks to fill all information.

- **Use Case: Manage Profile**

Table 2.5 UC-4: Manage Profile

UC-4: Manage Profile	
Primary actors	Applicant, advertiser
Pre-condition	User must be registered and logged in
Post-condition	Profile will be updated successfully
Main success scenarios	1. User opens the profile. 2. Relevant page/ form is displayed by the system to the user. 3. User opens the form and fills in the information. 4. System prompts the message that profile successfully updated.
Alternate scenarios	System fails at any time 1. System erase all the changes previously made. 3. User doesn"t fill in the correct/ complete information. 3.a) Form will not be submitted and ask to fill it completely.

- **Use Case: Add Advertisement**

Table 2.6 UC-5: Add Advertisement

UC-5: Add Advertisement	
Primary actors	Advertiser
Pre-condition	User must be registered and logged in.
Post-condition	Advertisement will be added successfully.
Main success scenarios	1. User selects add advertisement option. 2. System displays form for adding/submitting advertisement. 3. User fills in the form. 4. System stores the advertisement in database and advertisement is displayed.
Alternate Scenarios	System fails at any time. 1. it rolls back all the changes made 3.a) User doesn‟t fill in the correct/ complete information. a) system asks to fill it again properly.

- **Use Case: Update Skills**

Table 2.7 UC-6: Update Skills

UC-6: Update Skills	
Primary actors	Admin
Pre-condition	Admin must be registered and is logged In.
Post condition	Skills added successfully.
Main success scenarios	1. User opens the profile and go to relevant page. 2. Admin adds the skills in applicant form. 3. Skills added successfully.

- **Use Case: Search Applicant**

Table 2.8 UC-7: Search Applicant

UC-7: Search Applicant	
Primary actors	Advertiser
Pre-condition	User must be registered and logged in. GPS should be on.
Post condition	List will be displayed for all applicable applicants.
Main success scenarios	1. User click on search bar 2. System displays the form. 3. Advertiser will write the desired skills and their location. 4. System displays filtered result of all the relevant users.
Alternate scenarios	System fails at any time 1. Rolls back all the changes. 1.a) Log in again or search again. 3.a) Skills criteria not filled in for applicants 3.a.1) fill it properly. 3.a) GPS is off. 3.a.2) turn it on and submit it.

- **Use Case: View Applicant**

Table 2.9 UC-8: View Applicant

UC-8: View Applicant	
Primary actors	Advertiser
Pre-condition	User is logged in and Search is done.
Post condition	Applicant‟s profile will be displayed.
Main success scenarios	1. User clicks on the profile icon. 2. System displays the profile and relevant information.
Alternate scenarios	System fails at any time 1. System rolls back the changes being made Click again on profile icon.

- **Use Case: Wish list**

Table 2.10 UC-9: Wish list

UC-9: Wish list	
Primary actors	Applicant
Pre-condition	User is logged in and profile is viewed/displayed.
Post condition	Job saved successfully.
Main success scenarios	1. User clicks on favorite/save option. 2. System prompts added to Wishlist 3. Job saved successfully.
Alternate scenarios	System fails at any time 1. profile shuts down 2. Interface might not be displayed due to slow connection.

- **Use Case: Search Advertisement**

Table 2.11 UC-10: Search Advertisement

UC-10: Search Advertisement	
Primary actors	Applicants (transgender only)
Pre-condition	Applicant must be logged in and GPS is on.
Post condition	List will be displayed for all the relevant job advertisements.
Main success scenarios	1. User clicks on search bar. 2. Search pane is displayed. 3. User will write their skills/degree/ desired job of their interest. 4. User will click on search. 5. System will display the filtered results of all the relevant advertisements.
Alternate scenarios	System fails at any time 2. Rolls back all the changes. 2.a) Log in again or search again. 3.a) form not filled in for applicants 3.a1) System prompts to fill it properly. 4.a) GPS is off. 4.a1) System asks to turn it on and submit it.

- **Use Case: View Advertisement**

Table 2.12 UC-11: View Advertisement

UC-11: View Advertisement	
Primary actors	Applicant
Pre-condition	User must be logged in and search is already performed.
Post condition	Advertisement will be displayed successfully.
Main success scenarios	1.User clicks on the profile icon. 2. System displays the advertisement and relevant information.
Alternate scenarios	System fails at any time 2. System rolls back the changes being made Click again on displayed advertisement‟s icon.

- **Use Case: Chat**

Table 2.13 UC-12: Chat

UC-12: Chat	
Primary actors	Applicant/Advertiser
Pre-condition	User is logged in.
Post condition	Messaged successfully.
Main success scenarios	1. User opens conversation page. 2. System displays all the conversations. 3. User opens the conversation. 4. If that request meets the criteria than user responds it.
Alternate scenarios	System fails at any time 1. Rolls back all the changes made. 4.a) Requestor doesn‟t meet the criteria so message discarded.

- **Use Case: Sign Out**

Table 2.14 UC-13: Sign Out

UC-13: Sign Out	
Primary actors	Advertiser, Applicant
Pre-condition	User must be logged in.
Post condition	User will be logged out successfully.
Main success scenarios	1. User clicks on sign out option. 2. system prompts, logged out successfully. 3. System displays the main interface/page.
Alternate scenarios	System fails at any time 1. Rolls back all the changes made by admin.

- **Use Case: Apply for Job**

Table 2.15 UC-14: Apply for Job

UC-14: Apply for Job	
Primary actors	Applicant
Pre-condition	User must be logged in.
Post condition	Applied for job successfully.
Main success scenarios	4. User clicks on Apply option. 5. System prompts, applied successfully. 6. System displays the main interface/page.
Alternate scenarios	System fails at any time 1. Rolls back all the changes made by admin.

CHAPTER 3

SYSTEM DESIGN DESCRIPTION

3.1 Design Overview

In the design overview, we can describe the system at architecture level and the complete structure of the system. In this we can explain how the user interacts with the system. In the class diagram, we can show that how the classes can interact with each other define the relationship. Sequence diagram is an interaction diagram, that represents the sequence of messages interacting with objects.

3.2 Requirements Traceability Matrix

Requirement traceability matrix is matrix representation in which we define how the requirements are mapped with the system sequence diagram, design sequence diagram, class diagram and the test cases. When the requirement is changed you just see the requirement traceability matrix instead of digging the whole document.

Table 0.1 Requirements Traceability Matrix

UC Id	Requirement Name	System Sequence diagram	Object Sequence diagram	Interface	Test cases
UC-1	Log in	3.1.1	3.7.1	3.6.1	4.3.1, 4.3.2
UC-2	Register Applicant	3.1.2	3.7.2	3.6.2	4.3.3, 4.3.4
UC-3	Register Advertiser	3.1.3	3.7.3	3.6.3	4.3.5, 4.3.6
UC-4	Search Applicant	3.1.4	3.7.4	3.6.6	4.3.12
UC-5	View Applicant	3.1.6	3.7.5	3.6.6	4.3.13
UC-6	Search Ad	3.1.5	3.7.6	3.6.6	4.3.11

UC-7	View Advertisement	3.1.7	3.7.7	3.6.5	4.3.10
UC-8	Post Advertisement	3.1.8	3.7.8	3.6.7	4.3.7, 4.3.8
UC-9	Update skills		3.7.9		4.3.9
UC-10	Reply/Chat		3.7.10		
UC-11	Send Message		3.7.11		
UC-12	Manage profile	3.1.9			
UC-13	Log out	3.1.10			
UC-14	Apply for job				

3.3 System Sequence Diagrams

A system sequence diagram illustrates input and output events related to our system. System is treated as a black box and the emphasis of the diagram is events that are generated by system for a particular scenario of use-case.

- **Login**

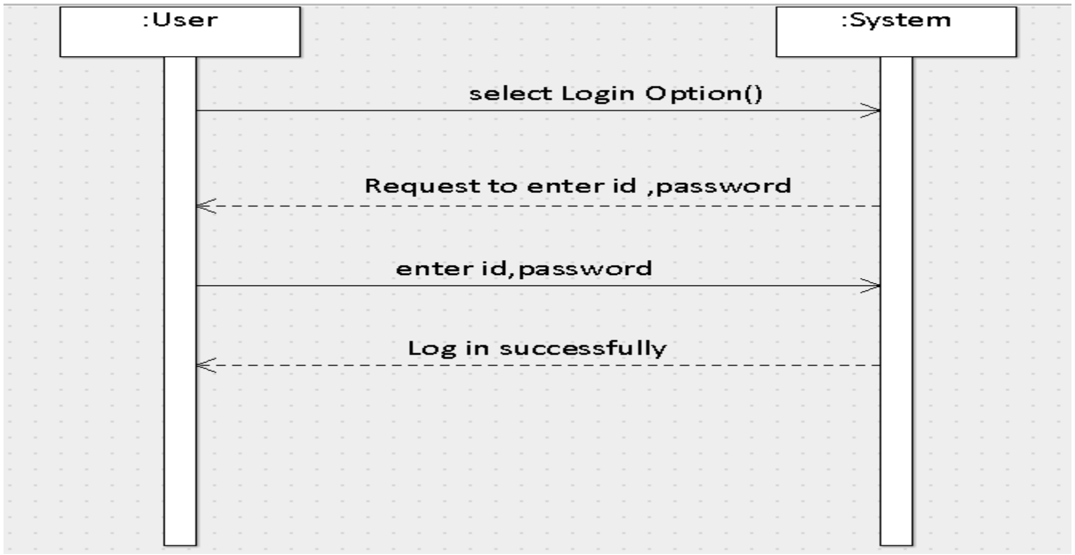


Figure 3 Login SSD

- **Register Applicant**

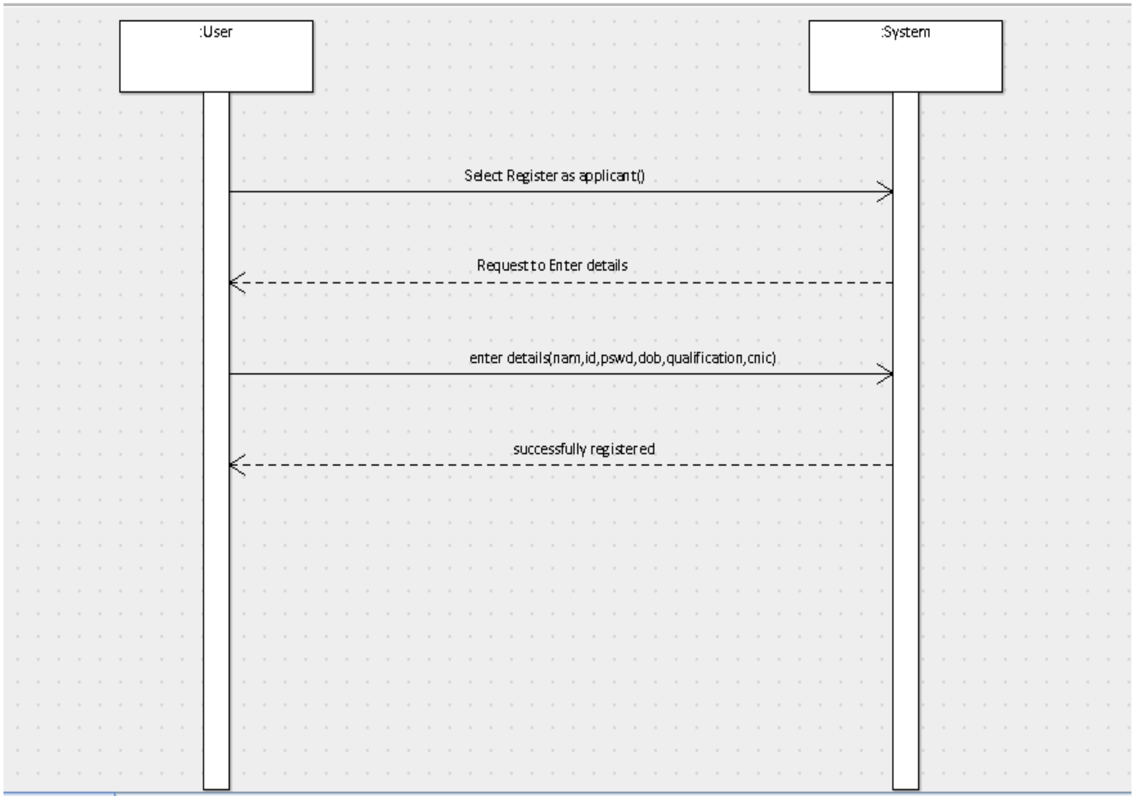

:User
:System
Select Register as applicant()
Request to Enter details
enter details(nam,id,pswd,dob,qualification,cnic)
successfully registered

Figure 4 Register Applicant SSD

- **Register Advertiser**

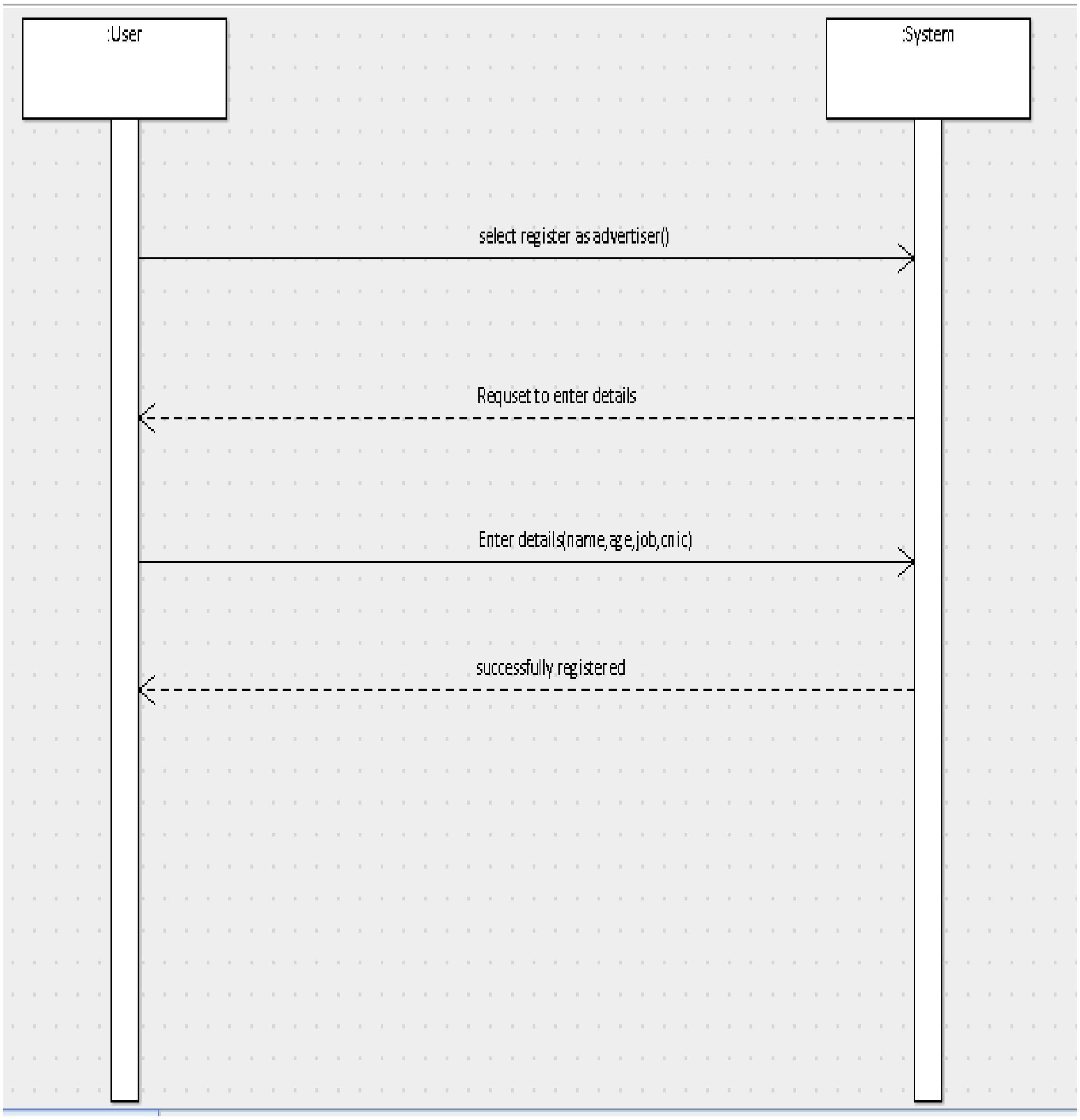

:User
:System
select register as advertiser()
Requset to enter details
Enter details(name,age,job,cnic)
successfully registered

Figure 5 Register Advertiser SSD

- **Search Applicant**

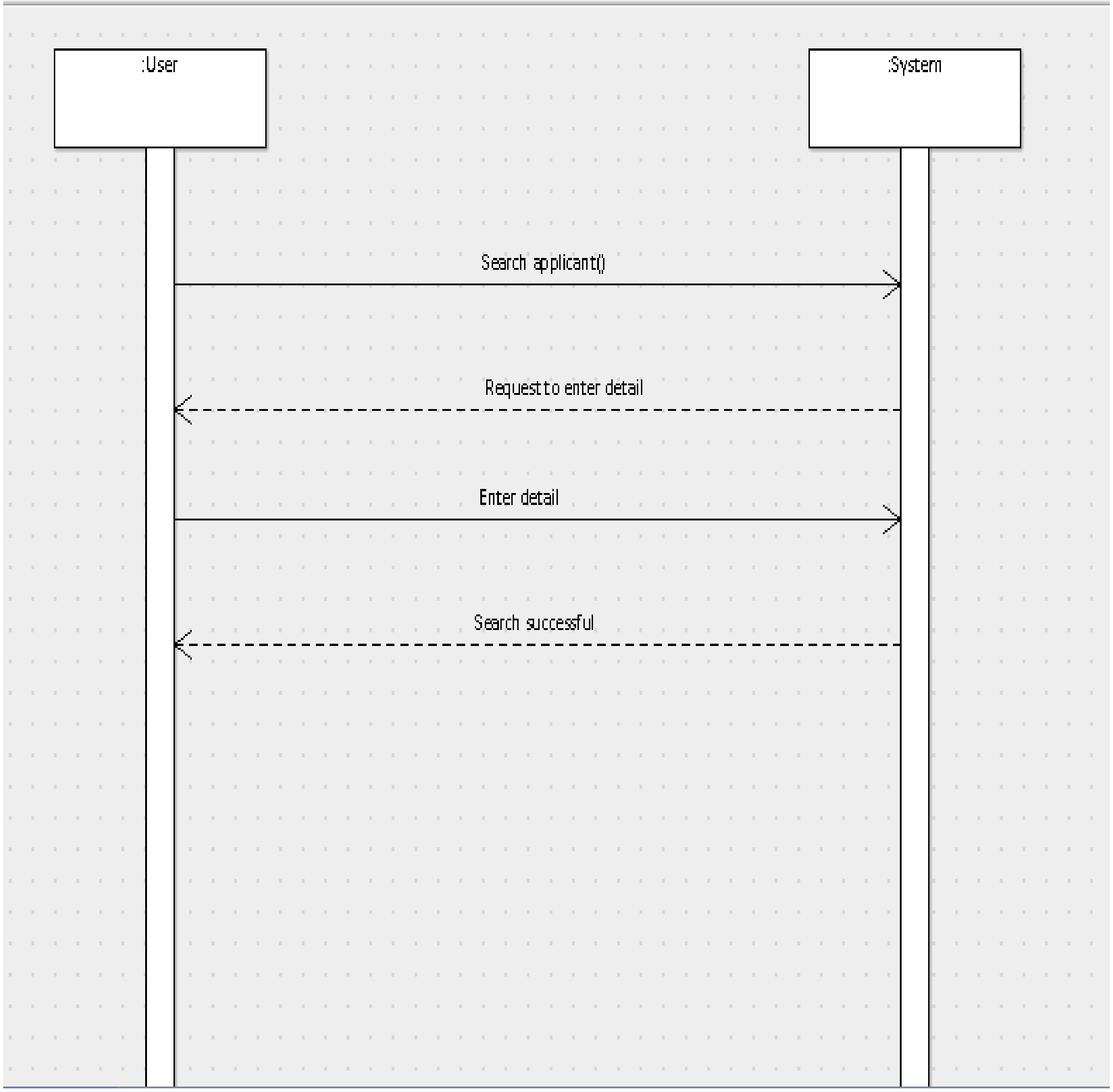


Figure 6 Search Applicant SSD

- **Search Advertisement**

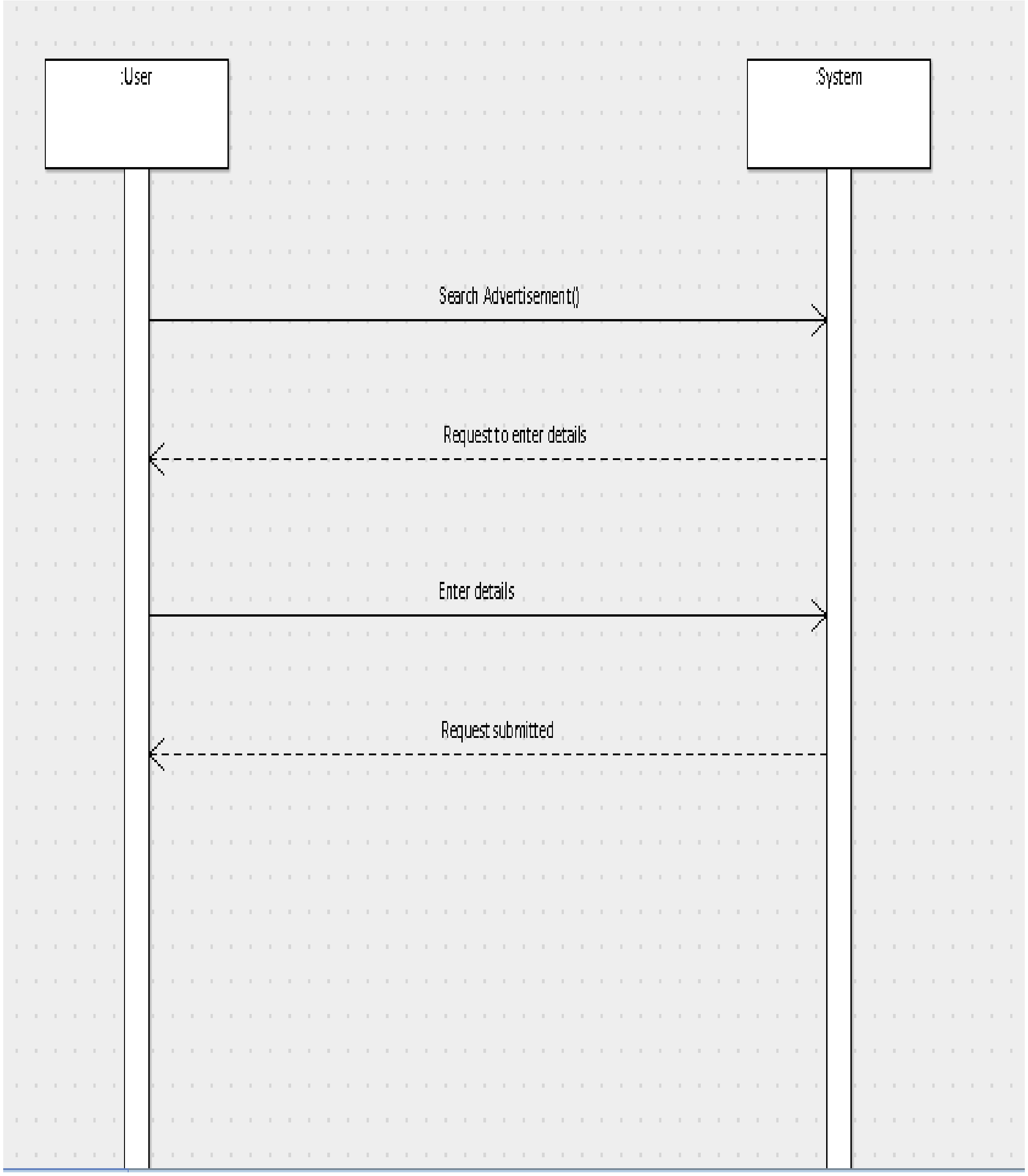


Figure 7 Search Advertisement SSD

- **View Applicant**

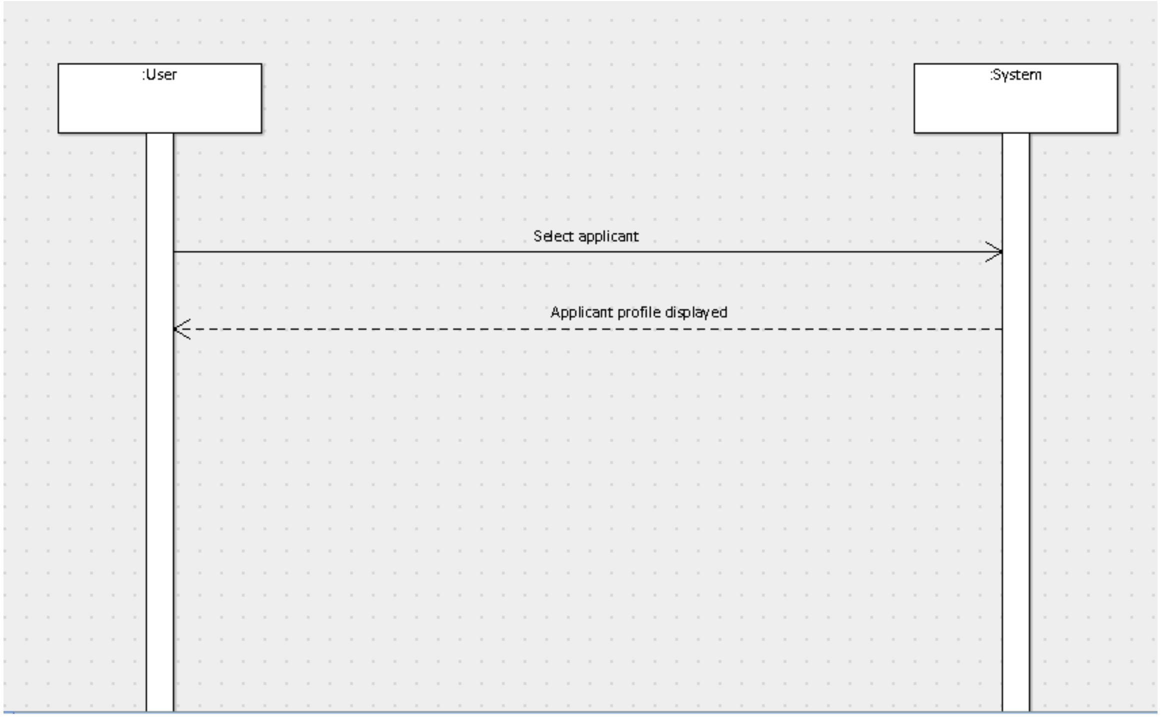


Figure 8 View Applicant SSD

- **View Advertisement**

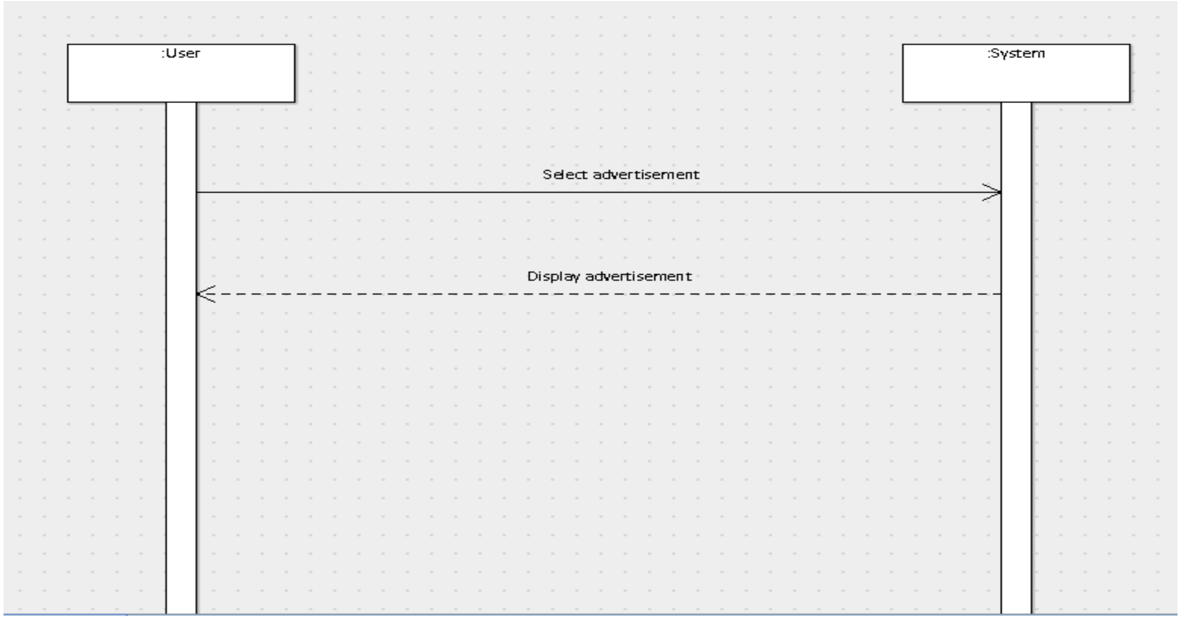


Figure 9 View Advertisement SSD

- **Add Advertisement**

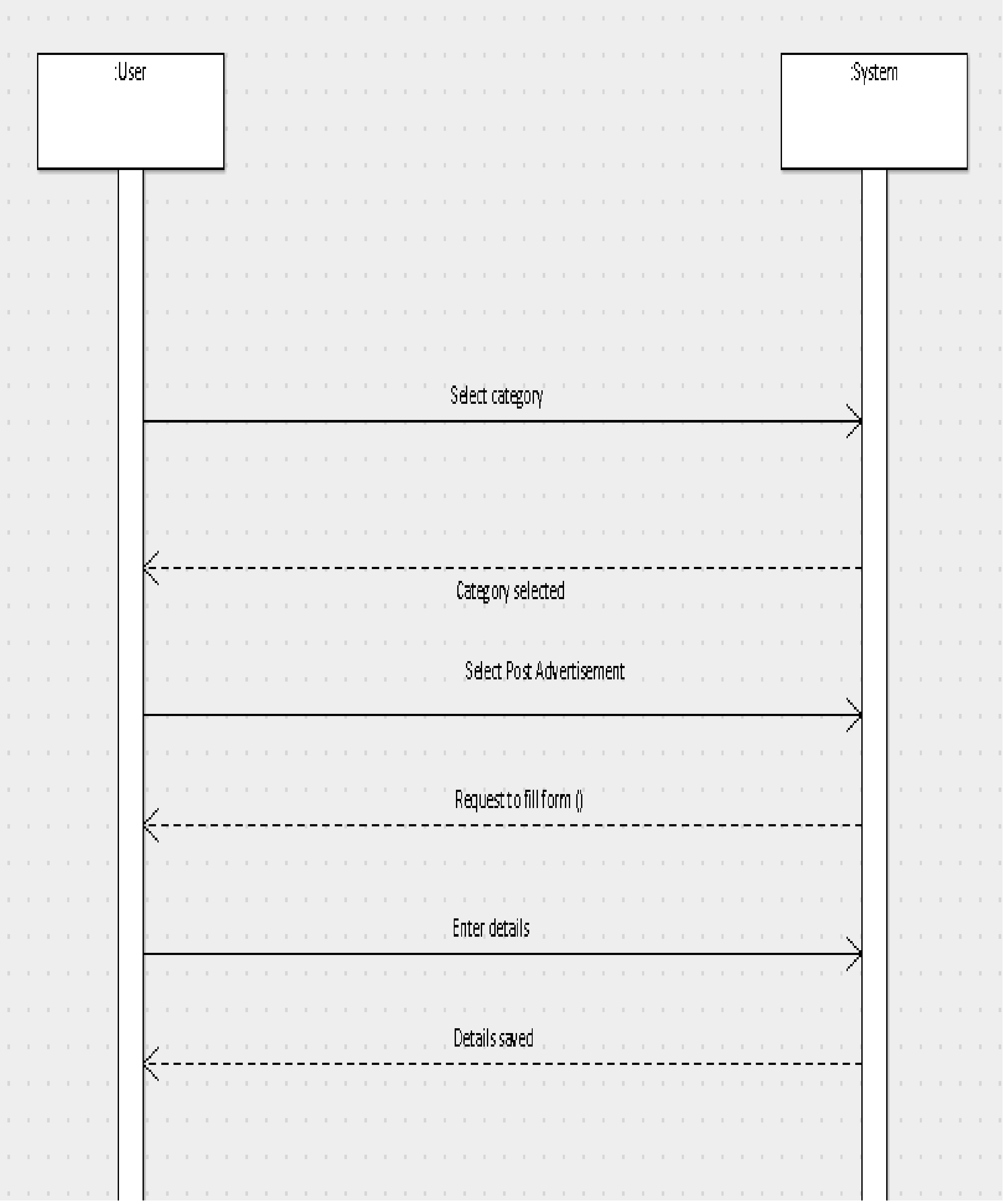


Figure 10 Add Advertisement SSD

- **Manage Profile**

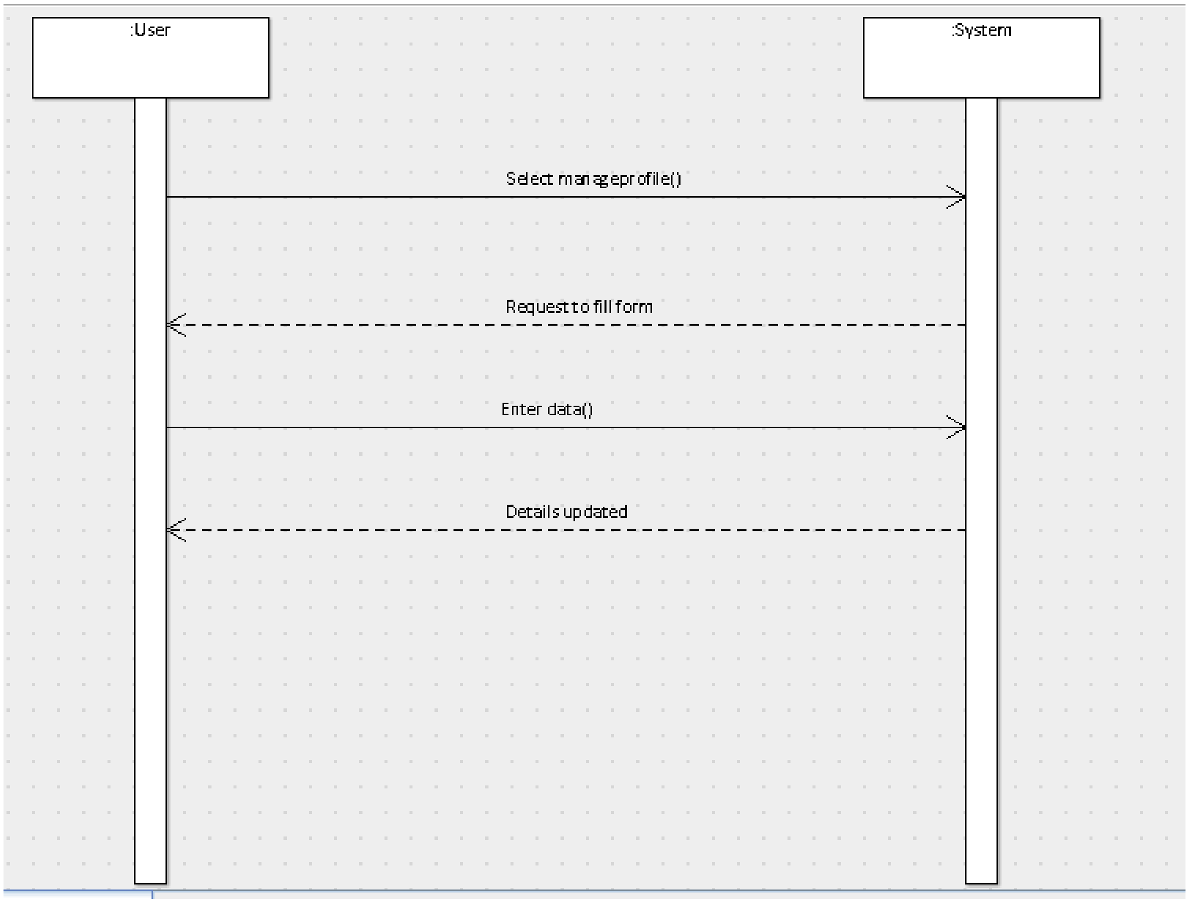


Figure 11 Manage Profile SSD

3.4 Class Diagram

A class diagram in the Unified Modeling Language (UML) 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. It is created by a plug in named SimpleUml by the code we have implemented in Android Studio.

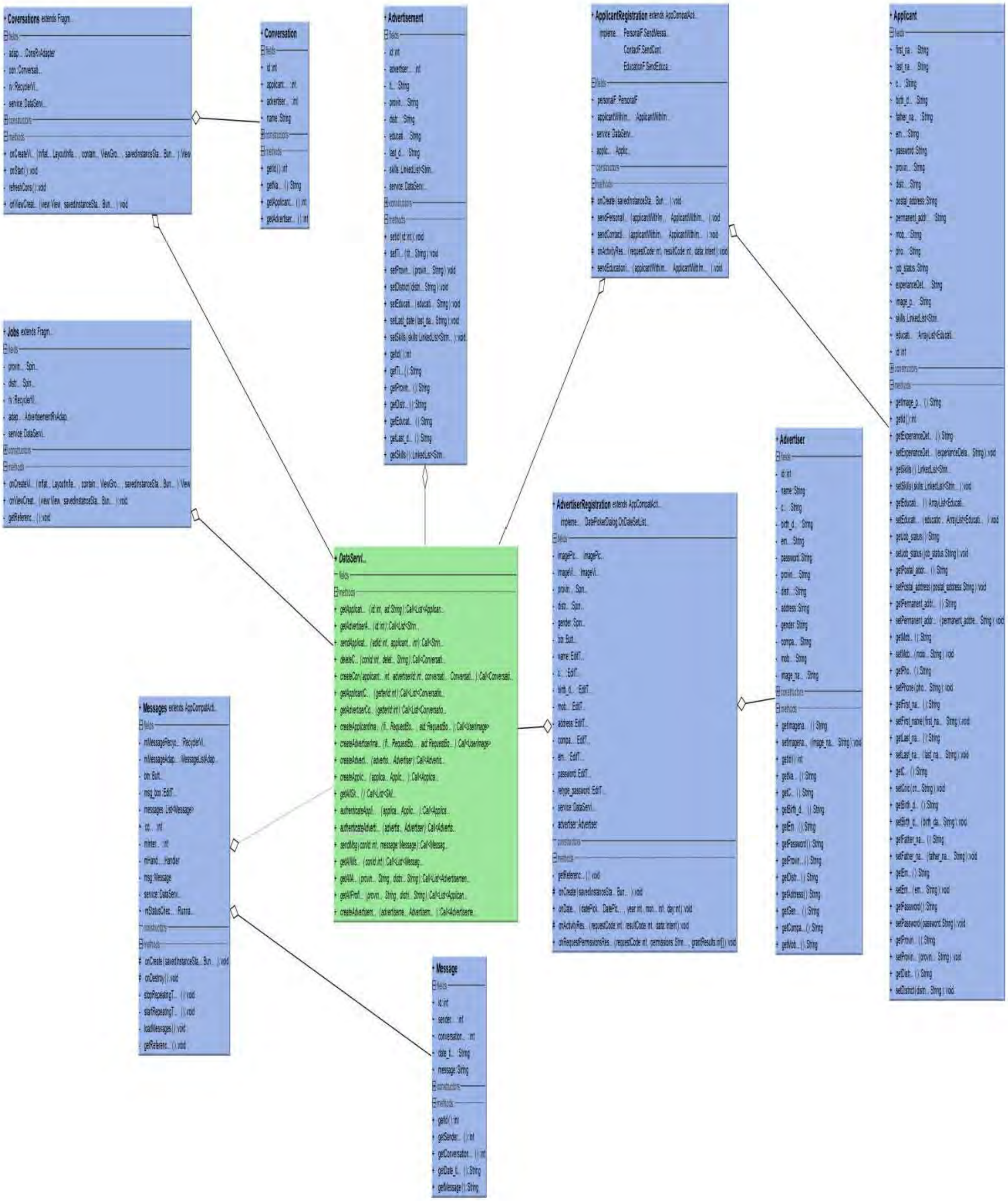

+ Coversations extends Fragm...
fields
- adap... :ConsRvAdapter
- con :Conversati...
- rv :RecyclerVi...
- service :DataServi...
constructors
methods
+ onCreateVi... (inflat... LayoutInfla..., contain... ViewGro..., savedInstanceSta... Bun...) :View
+ onStart() :void
- refreshCons() :void
+ onViewCreat... (view View, savedInstanceSta... Bun...) :void
+ Conversation
fields
+ id :int
+ applicant... :int
+ advertiser... :int
+ name :String
constructors
methods
+ getId() :int
+ getNa... () :String
+ getApplicant... () :int
+ getAdvertiser... () :int
+ Advertisement
fields
- id :int
- advertiser... :int
- ti... :String
- provin... :String
- distr... :String
- educati... :String
- last_d... :String
- skills :LinkedList<Strin...
- service :DataServ...
constructors
methods
+ setId(id:int) :void
+ setTi... (tit... String) :void
+ setProvin... (provin... String) :void
+ setDistrict(distr... String) :void
+ setEducati... (educati... String) :void
+ setLast_date(last_da... String) :void
+ setSkills(skills LinkedList<Strin...) :void
+ getId() :int
+ getTi... () :String
+ getProvin... () :String
+ getDistr... () :String
+ getEducati... () :String
+ getLast_d... () :String
+ getSkills() :LinkedList<Strin...
+ ApplicantRegistration extends AppCompatActi...
impleme... PersonalF.SendMessa...
ContactF.SendCont...
EducationF.SendEduca...
fields
- personalF :PersonalF
- applicantWithIn... :ApplicantWithIn...
- service :DataServ...
- applic... :Applic...
constructors
methods
# onCreate(savedInstanceSta... Bun...) :void
+ sendPersonal... (applicantWithIn... ApplicantWithIn...) :void
+ sendContact... (applicantWithIn... ApplicantWithIn...) :void
# onActivityRes... (requestCode:int, resultCode:int, data:Intent) :void
+ sendEducation... (applicantWithIn... ApplicantWithIn...) :void
+ Applicant
fields
- first_na... :String
- last_na... :String
- c... :String
- birth_d... :String
- father_na... :String
- em... :String
- password :String
- provin... :String
- dist... :String
- postal_address :String
- permanent_addr... :String
- mob... :String
- pho... :String
- job_status :String
- experianceDet... :String
- image_p... :String
- skills :LinkedList<Strin...
- educati... :ArrayList<Educati...
- id :int
constructors
methods
+ getImage_p... () :String
+ getId() :int
+ getExperianceDet... () :String
+ setExperianceDet... (experianceDeta... String) :void
+ getSkills() :LinkedList<Strin...
+ setSkills(skills LinkedList<Strin...) :void
+ getEducati... () :ArrayList<Educati...
+ setEducati... (educatio... ArrayList<Educati...) :void
+ getJob_status() :String
+ setJob_status(job_status:String) :void
+ getPostal_addr... () :String
+ setPostal_address(postal_address:String) :void
+ getPermanent_addr... () :String
+ setPermanent_addr... (permanent_addre... String) :void
+ getMob... () :String
+ setMob... (mob... String) :void
+ getPho... () :String
+ setPhone(pho... String) :void
+ getFirst_na... () :String
+ setFirst_name(first_na... String) :void
+ getLast_na... () :String
+ setLast_na... (last_na... String) :void
+ getC... () :String
+ setCnic(cn... String) :void
+ getBirth_d... () :String
+ setBirth_d... (birth_da... String) :void
+ getFather_na... () :String
+ setFather_na... (father_na... String) :void
+ getEm... () :String
+ setEm... (em... String) :void
+ getPassword() :String
+ setPassword(password:String) :void
+ getProvin... () :String
+ setProvin... (provin... String) :void
+ getDist... () :String
+ setDistrict(distr... String) :void
+ Jobs extends Fragm...
fields
- provin... :Spin...
- distr... :Spin...
- rv :RecyclerVi...
- adap... :AdvertisementRvAdap...
- service :DataServi...
constructors
methods
+ onCreateVi... (inflat... LayoutInfla..., contain... ViewGro..., savedInstanceSta... Bun...) :View
+ onViewCreat... (view View, savedInstanceSta... Bun...) :void
- getReferenc... () :void
+ DataServi...
fields
methods
+ getApplicat... (id:int, ad:String) :Call<List<Applican...
+ getAdvertiserA... (id:int) :Call<List<Strin...
+ sendApplicat... (adId:int, applicant... int) :Call<Strin...
+ deleteC... (conId:int, delet... String) :Call<Conversati...
+ createCon(applicant... int, advertiserId:int, conversati... Conversati...) :Call<Conversati...
+ getApplicantC... (getterId:int) :Call<List<Conversatio...
+ getAdvertiserCo... (getterId:int) :Call<List<Conversatio...
+ createApplicantIma... (fi... RequestBo..., ad:RequestBo...) :Call<UserImage>
+ createAdvertiserIma... (fi... RequestBo..., ad:RequestBo...) :Call<UserImage>
+ createAdverti... (advertis... Advertiser) :Call<Advertis...
+ createApplic... (applica... Applic...) :Call<Applica...
+ getAllSk... () :Call<List<Skil...
+ authenticateAppl... (applica... Applic...) :Call<Applica...
+ authenticateAdverti... (advertis... Advertiser) :Call<Advertis...
+ sendMsg(conId:int, message:Message) :Call<Messag...
+ getAllMs... (conId:int) :Call<List<Messag...
+ getAllA... (provin... String, distri... String) :Call<List<Advertisemen...
+ getAllProfi... (provin... String, distri... String) :Call<List<Applican...
+ createAdvertisem... (advertiseme... Advertisem...) :Call<Advertiseme...
+ AdvertiserRegistration extends AppCompatActi...
impleme... DatePickerDialog.OnDateSetList...
fields
- imagePic... :imagePic...
- imageVi... :ImageVi...
- provin... :Spin...
- distr... :Spin...
- gender :Spin...
- btn :Butt...
- name :EditT...
- c... :EditT...
- birth_d... :EditT...
- mob... :EditT...
- address :EditT...
- compa... :EditT...
- em... :EditT...
- password :EditT...
- retype_password :EditT...
- service :DataServi...
- advertiser :Advertiser
constructors
methods
- getReferenc... () :void
# onCreate(savedInstanceSta... Bun...) :void
+ onDate... (datePick... DatePic..., year:int, mon... int, day:int) :void
# onActivityRes... (requestCode:int, resultCode:int, data:Intent) :void
+ onRequestPermissionsRes... (requestCode:int, permissions:Strin..., grantResults:int[]) :void
+ Advertiser
fields
- id :int
- name :String
- c... :String
- birth_d... :String
- em... :String
- password :String
- provin... :String
- distr... :String
- address :String
- gender :String
- compa... :String
- mob... :String
- image_na... :String
constructors
methods
+ getImagena... () :String
+ setImagena... (image_na... String) :void
+ getId() :int
+ getNa... () :String
+ getC... () :String
+ getBirth_d... () :String
+ getEm... () :String
+ getPassword() :String
+ getProvin... () :String
+ getDistr... () :String
+ getAddress() :String
+ getGen... () :String
+ getCompa... () :String
+ getMob... () :String
+ Messages extends AppCompatActi...
fields
- mMessageRecyc... :RecyclerVi...
- mMessageAdap... :MessageListAdap...
- btn :Butt...
- msg_box :EditT...
- messages :List<Message>
+ co... :int
- minter... :int
- mHand... :Handler
- msg :Message
- service :DataServi...
+ mStatusChec... :Runna...
constructors
methods
# onCreate(savedInstanceSta... Bun...) :void
# onDestroy() :void
- stopRepeatingT... () :void
- startRepeatingT... () :void
- loadMessages() :void
- getReferenc... () :void
+ Message
fields
+ id :int
+ sender... :int
+ conversation... :int
+ date_ti... :String
+ message :String
constructors
methods
+ getId() :int
+ getSender... () :int
+ getConversation... () :int
+ getDate_ti... () :String
+ getMessage() :String

Figure 12 Class Diagram

3.5 Architecture Diagram

Architecture Diagram is used to represent the components of system and interaction between them. System under discussion is based on "Three Tier" Architectural pattern. Three-tier architecture is a software architecture pattern in which the user interface (presentation), functional process logic (business layer), computer data storage and data access are developed and maintained as independent modules Interacting between components of system is shown in diagram. Double arrows represent the interaction from both sides. Similarly, single arrow represents one-way interaction. The singular quality of a three-tier architecture is the separation of the application logic into a distinct logical middle tier of software. The interface tier is relatively free of application processing. The middle tier communicates with the back-end database layer. It is possible to make changes on the presentation level without affecting the other two (business or data access layer). As each tier is independent it is possible to use different sets of developers since the client doesn"t have direct access to the database business logic are more secure.

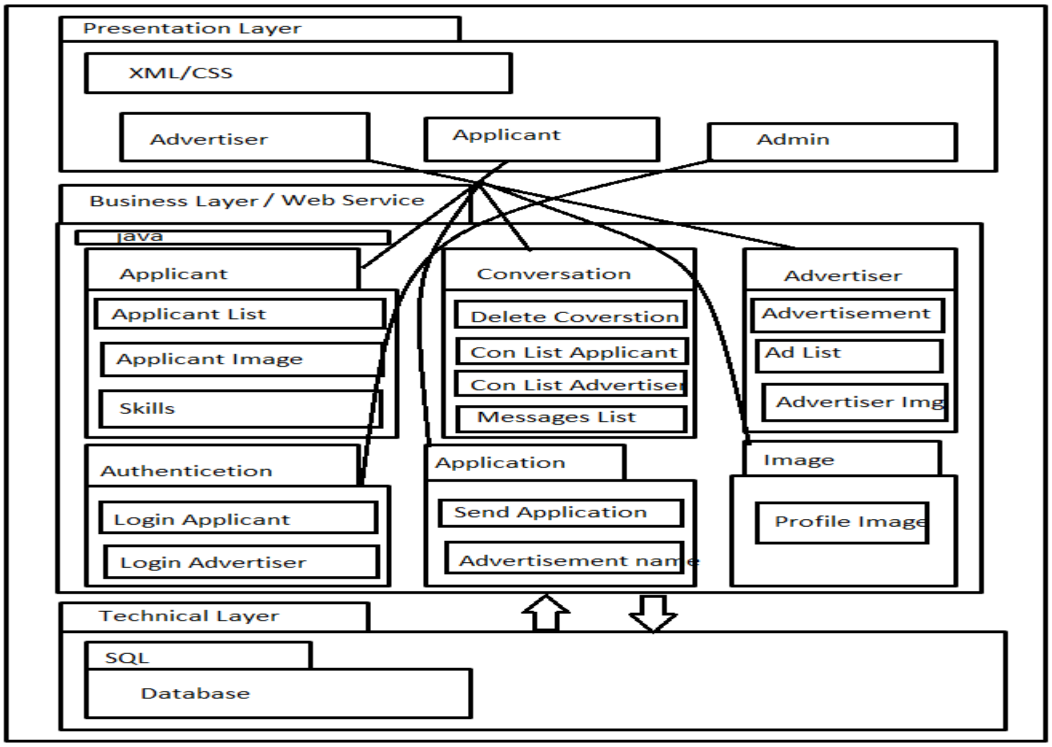


Figure 13 Architecture Diagram

3.5 Entity Relationship Diagram

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases. ERD Plus tool is used to develop it online .

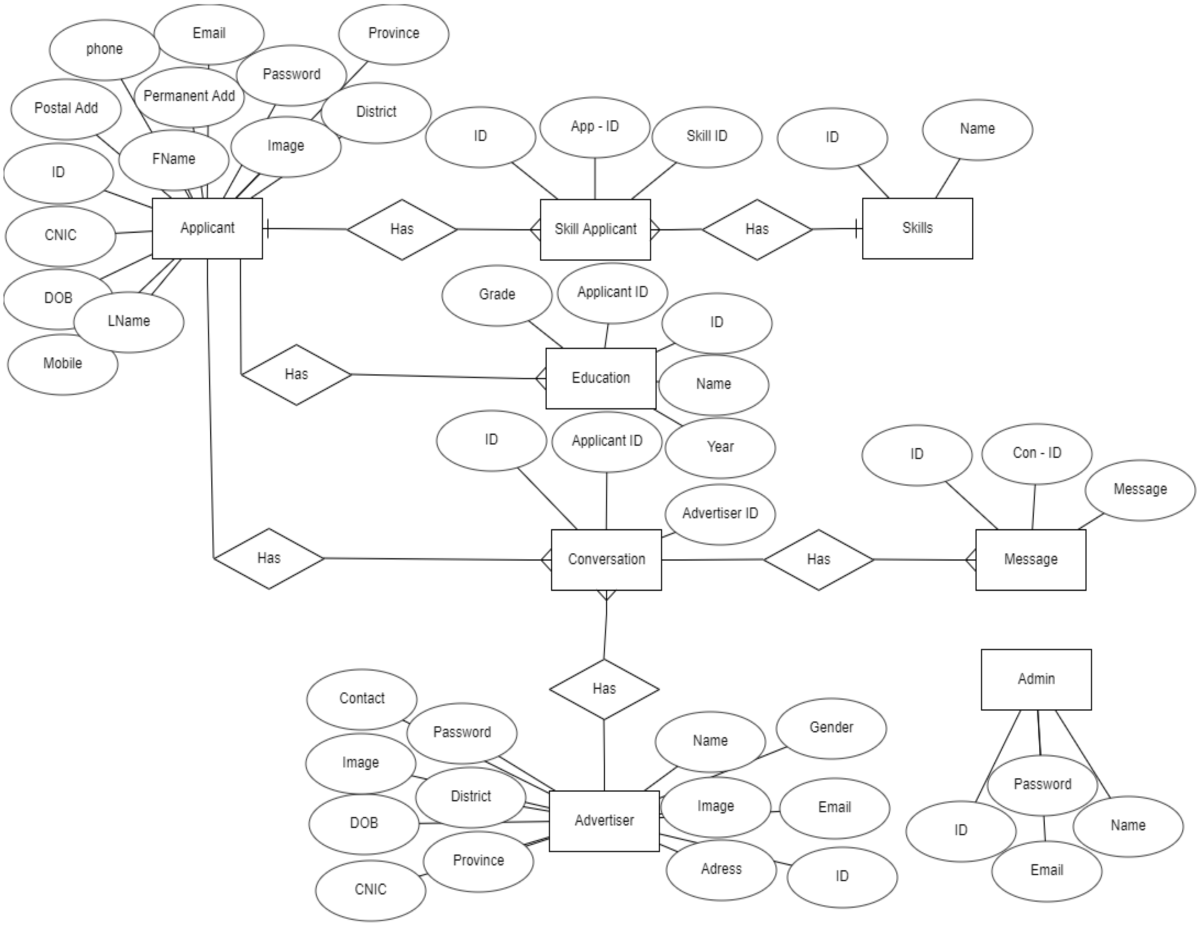
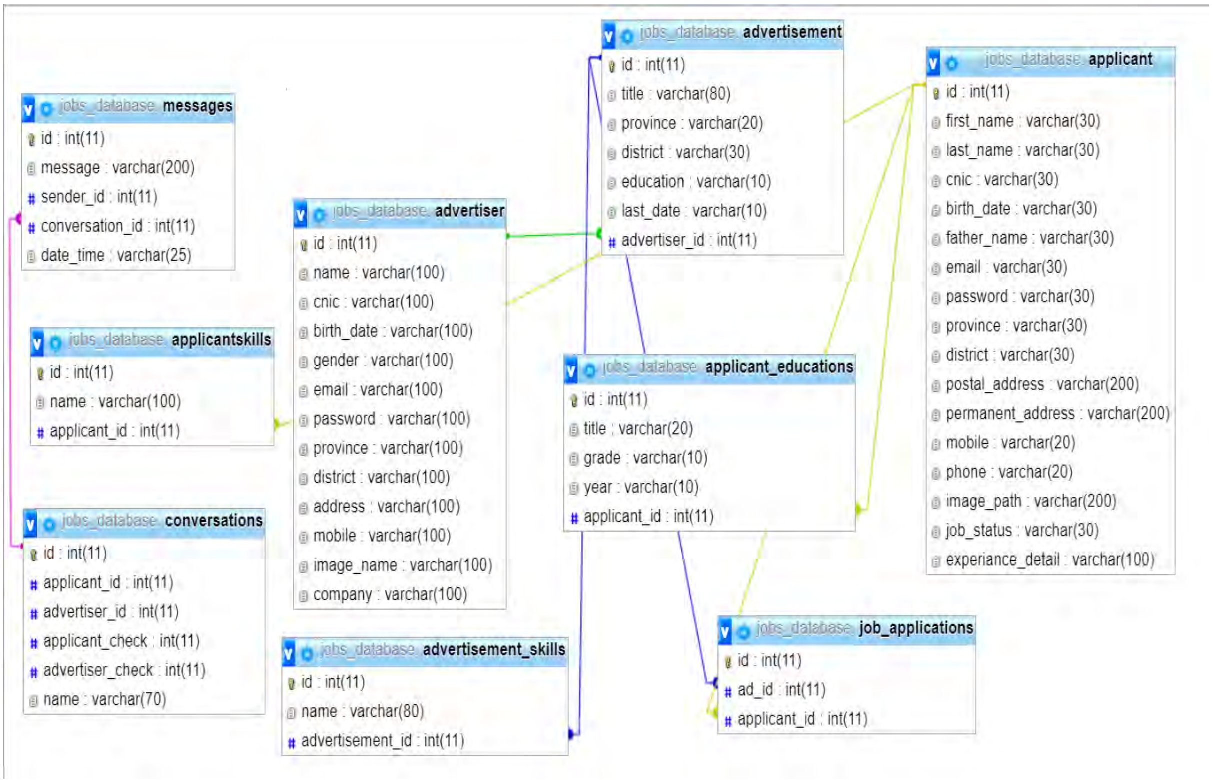


Figure 14 ERD

3.6 Database Design



jobs_database messages
id : int(11)
message : varchar(200)
sender_id : int(11)
conversation_id : int(11)
date_time : varchar(25)

jobs_database applicantskills
id : int(11)
name : varchar(100)
applicant_id : int(11)

jobs_database conversations
id : int(11)
applicant_id : int(11)
advertiser_id : int(11)
applicant_check : int(11)
advertiser_check : int(11)
name : varchar(70)

jobs_database advertiser
id : int(11)
name : varchar(100)
cnic : varchar(100)
birth_date : varchar(100)
gender : varchar(100)
email : varchar(100)
password : varchar(100)
province : varchar(100)
district : varchar(100)
address : varchar(100)
mobile : varchar(100)
image_name : varchar(100)
company : varchar(100)

jobs_database advertisement_skills
id : int(11)
name : varchar(80)
advertisement_id : int(11)

jobs_database advertisement
id : int(11)
title : varchar(80)
province : varchar(20)
district : varchar(30)
education : varchar(10)
last_date : varchar(10)
advertiser_id : int(11)

jobs_database applicant_educations
id : int(11)
title : varchar(20)
grade : varchar(10)
year : varchar(10)
applicant_id : int(11)

jobs_database job_applications
id : int(11)
ad_id : int(11)
applicant_id : int(11)

jobs_database applicant
id : int(11)
first_name : varchar(30)
last_name : varchar(30)
cnic : varchar(30)
birth_date : varchar(30)
father_name : varchar(30)
email : varchar(30)
password : varchar(30)
province : varchar(30)
district : varchar(30)
postal_address : varchar(200)
permanent_address : varchar(200)
mobile : varchar(20)
phone : varchar(20)
image_path : varchar(200)
job_status : varchar(30)
experiance_detail : varchar(100)

Figure 15 Database Design

3.7 Interface design

3.7.1 Simple and Appealing

The interface is simple to use, a user can also use it very easily and efficiently to get there required information. The image is used that makes application appealing.

3.7.2 Responsive

The interface is responsive this means that there is sequential flow of activities going on touch/click and it can adjust its size according to device display Like (landscape, portrait) etc.

3.8 Interfaces

3.8.1 Log in

This interface depicts how the user can log in to the system. User enters id and password.

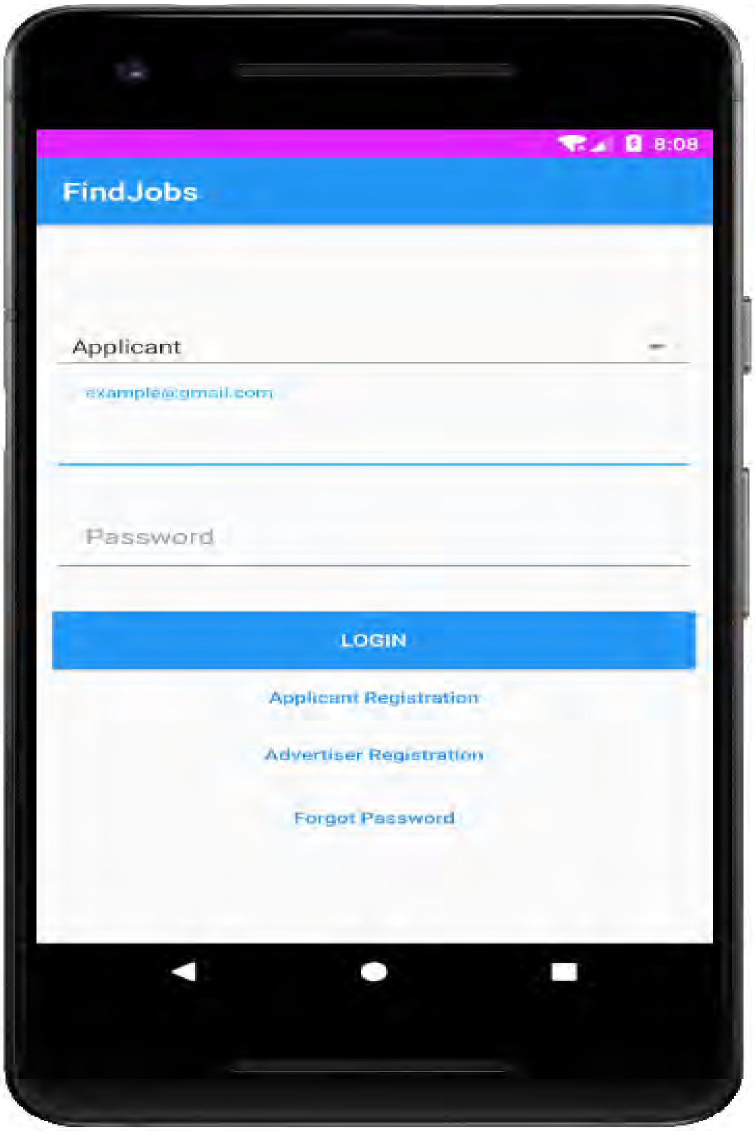


Figure 16 Login UI

3.8.2 Register Applicant & Advertiser

This interface shows how the applicant can be registered, it has 3 forms. One for personal information second for academic information and third for contact details.

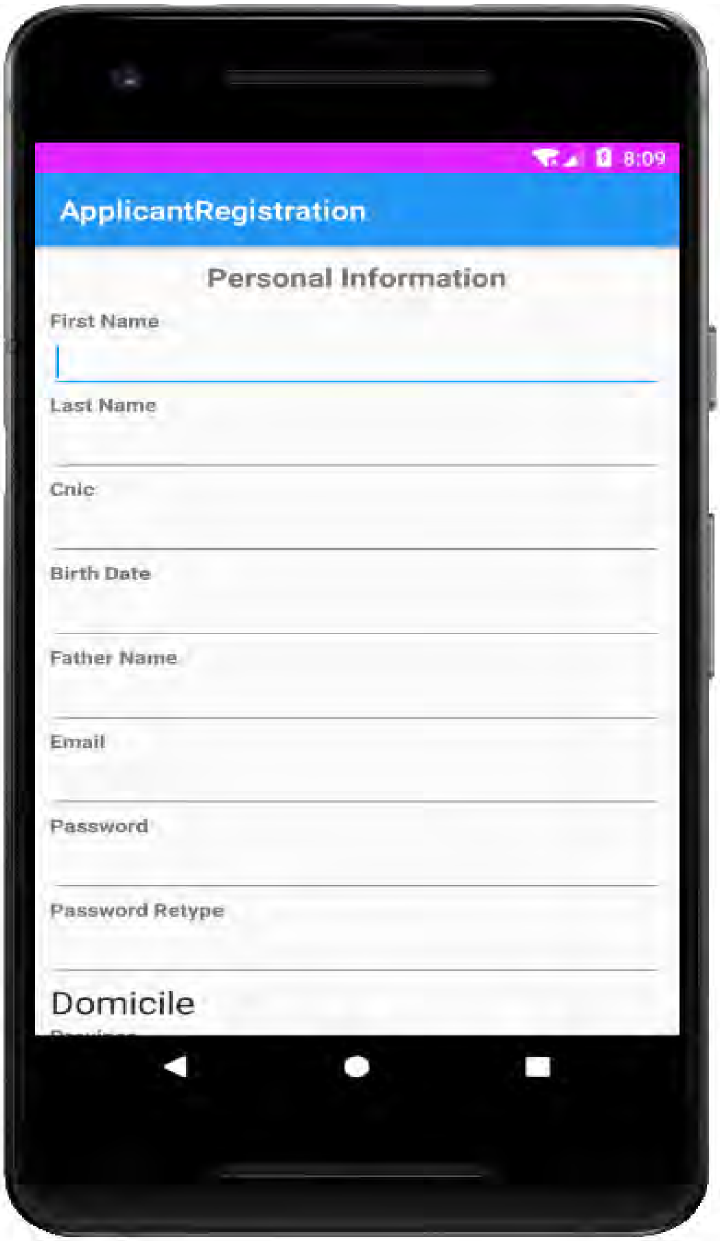


Figure 16 (a)Register Applicant UI

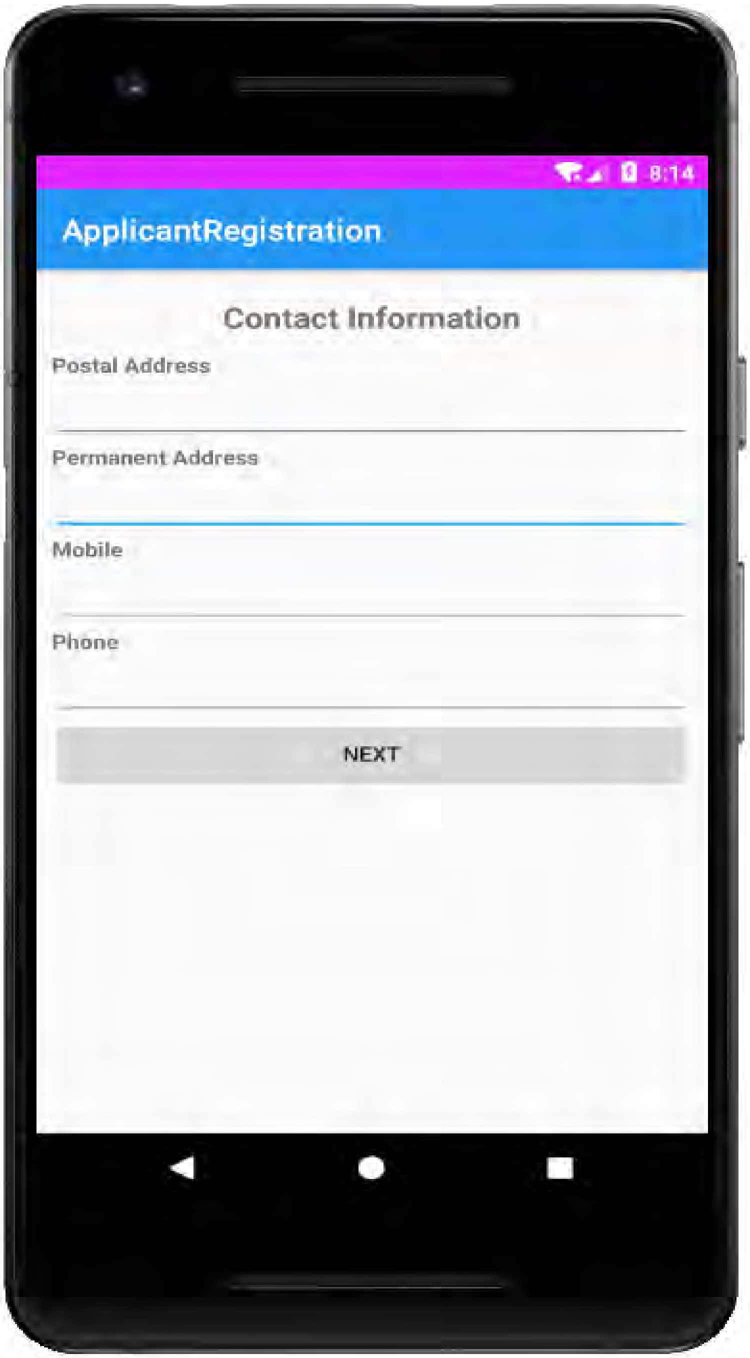
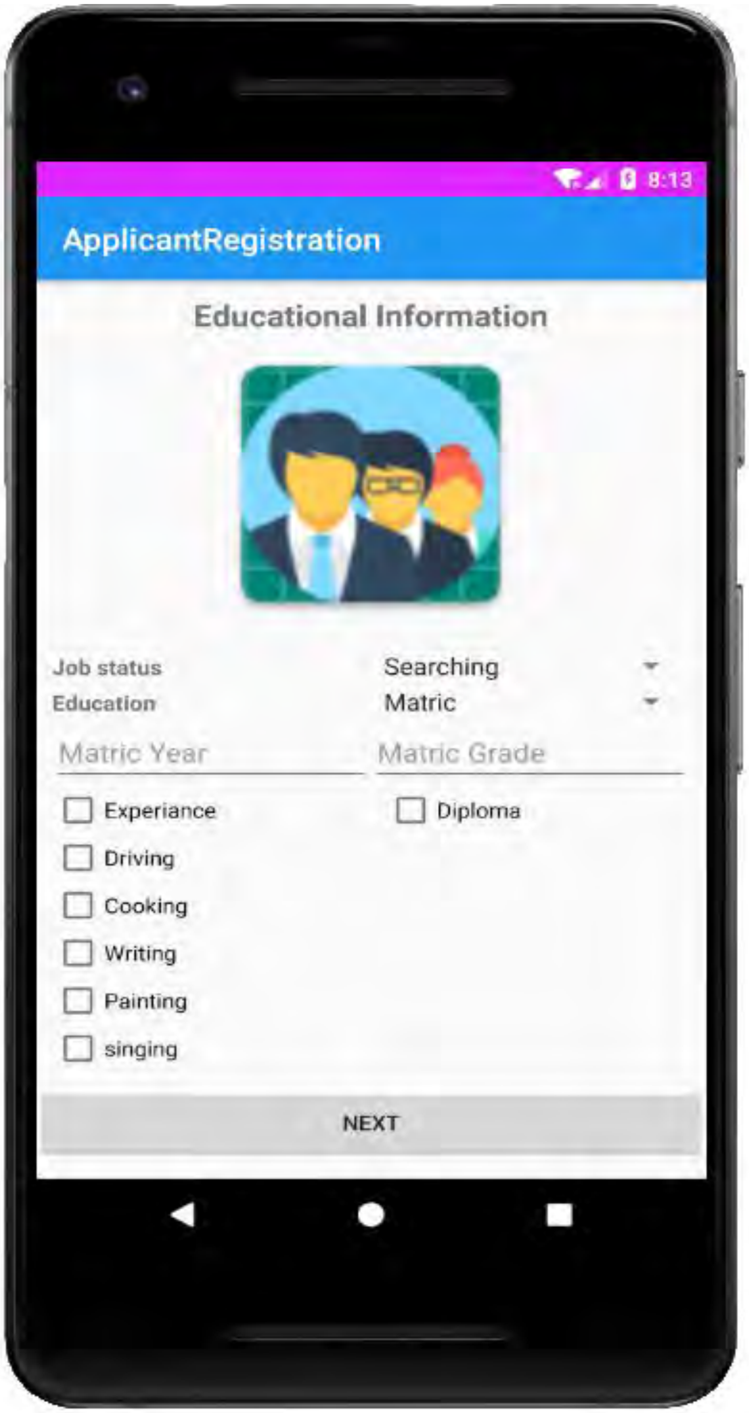


Figure 1 (b)Register Applicant UI



Educational Information

Job status Searching

Education Matric

Matric Year

Matric Grade

□ Experiance

□ Diploma

□ Driving

□ Cooking

□ Writing

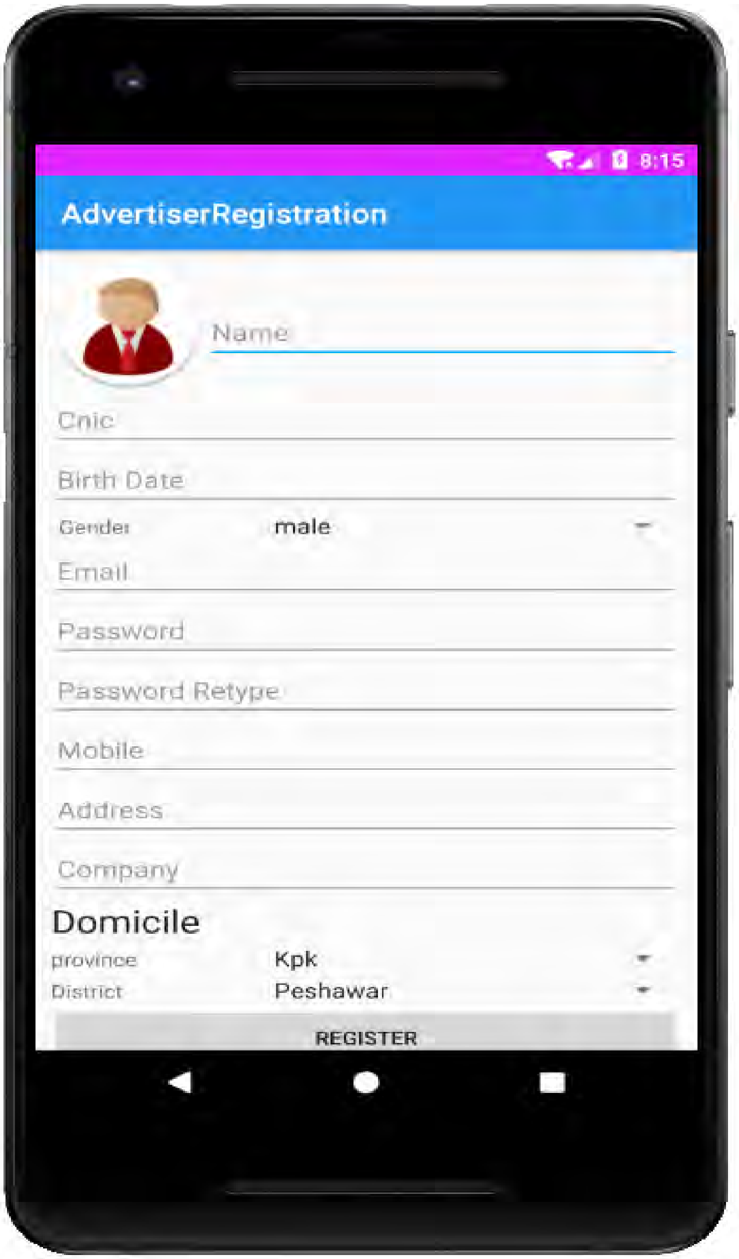
□ Painting

□ singing

NEXT

Figure 16 (c)Register Applicant UI

This interface shows the registration of advertiser, it consists of a form which advertiser will have to fill in first in case for getting register.



AdvertiserRegistration

Gender male

Domicile

province Kpk

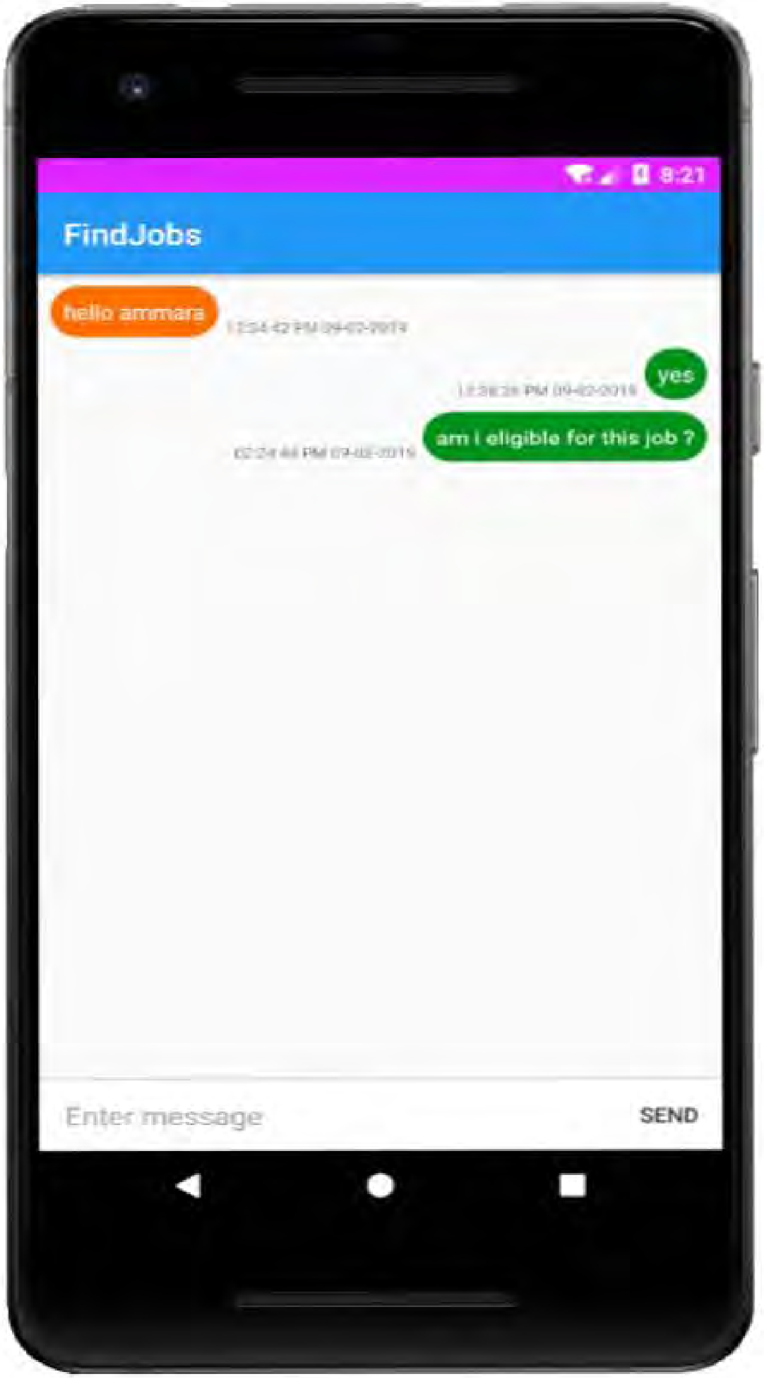
District Peshawar

Figure 17 Register Advertiser UI

.

3.8.3 Conversation

This interface shows conversation page



hello ammara 12:34:42 PM 09-02-2019

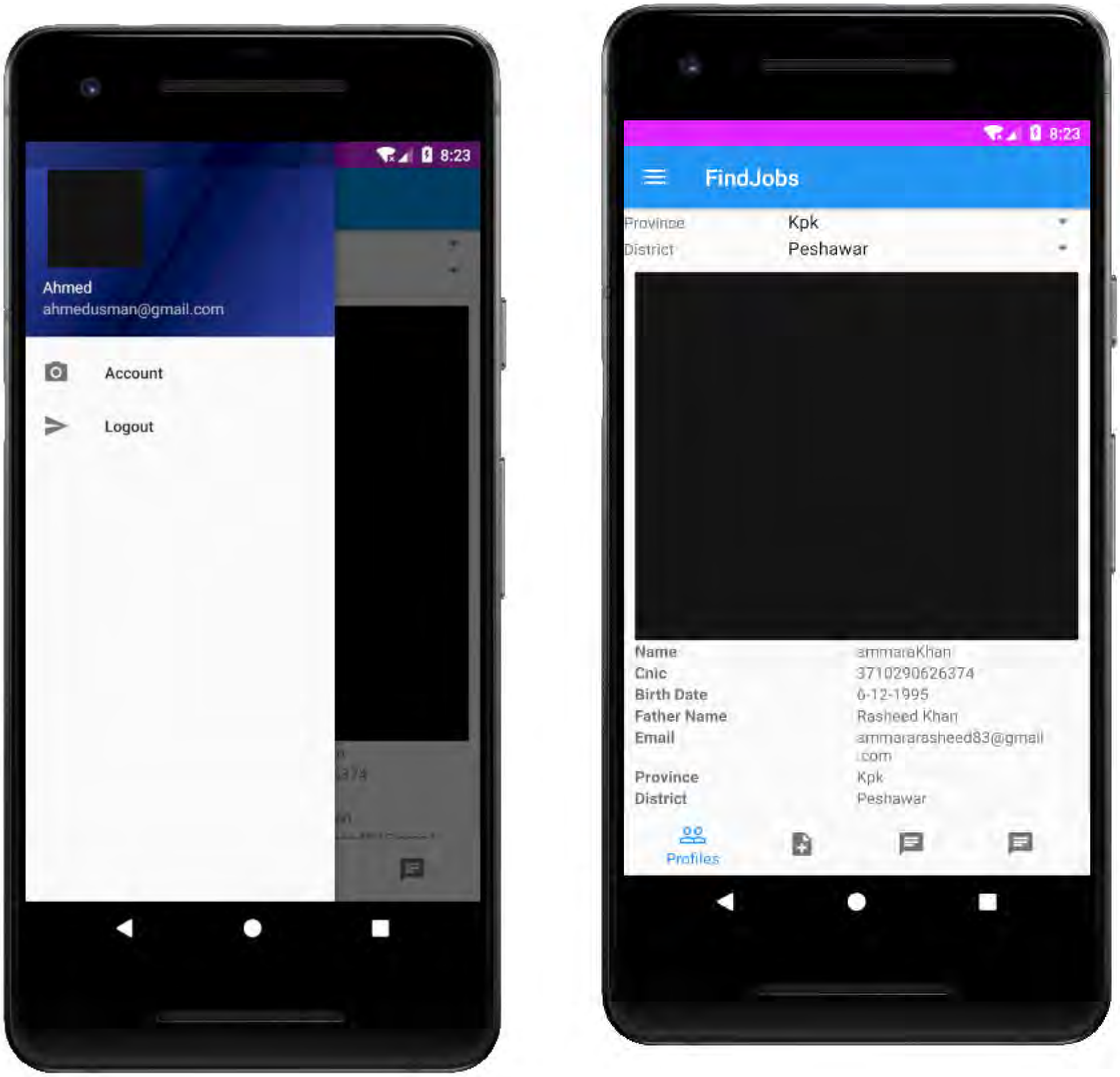
yes 12:38:28 PM 09-02-2019

am i eligible for this job ? 02:24:48 PM 09-02-2019

Figure 18 Conversation UI

3.8.3 Advertiser profile (Search/view/contact Applicant)

This interface depicts the home page of advertiser through which user can search, view and contact applicant.



Ahmed
ahmedusman@gmail.com

Province Kpk
District Peshawar

Name	ammaraKhan
Cnic	3710290626374
Birth Date	6-12-1995
Father Name	Rasheed Khan
Email	ammararasheed83@gmail.com
Province	Kpk
District	Peshawar

Figure19 Advertiser Profile UI

3.9 Design Sequence Diagram

A Sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence.

3.9.1 Log In

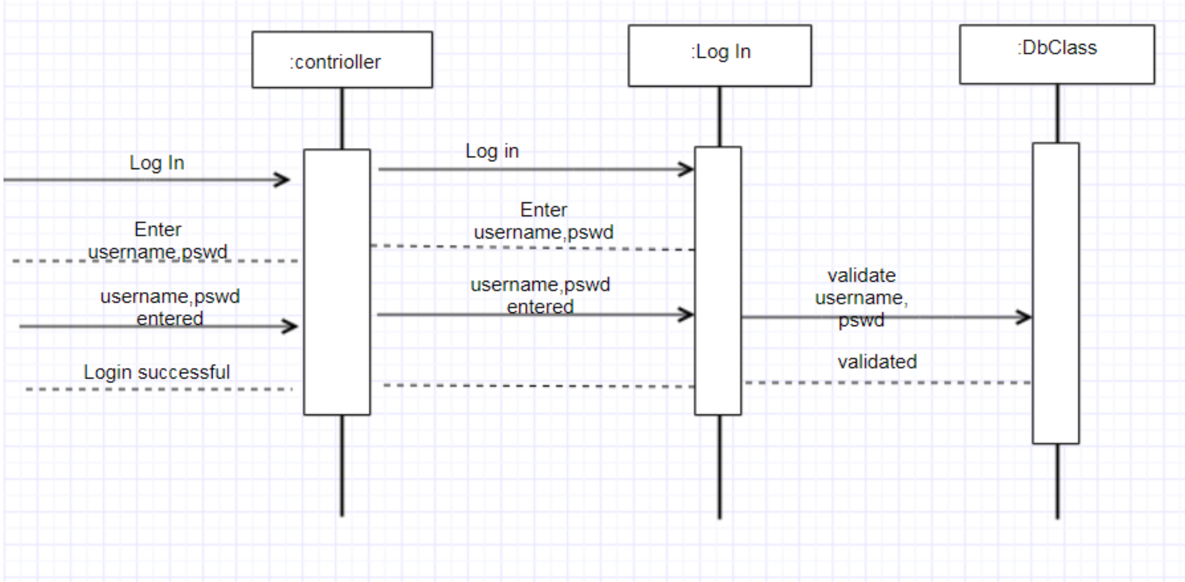


Figure 22 Login SD

3.9.2 Register Applicant

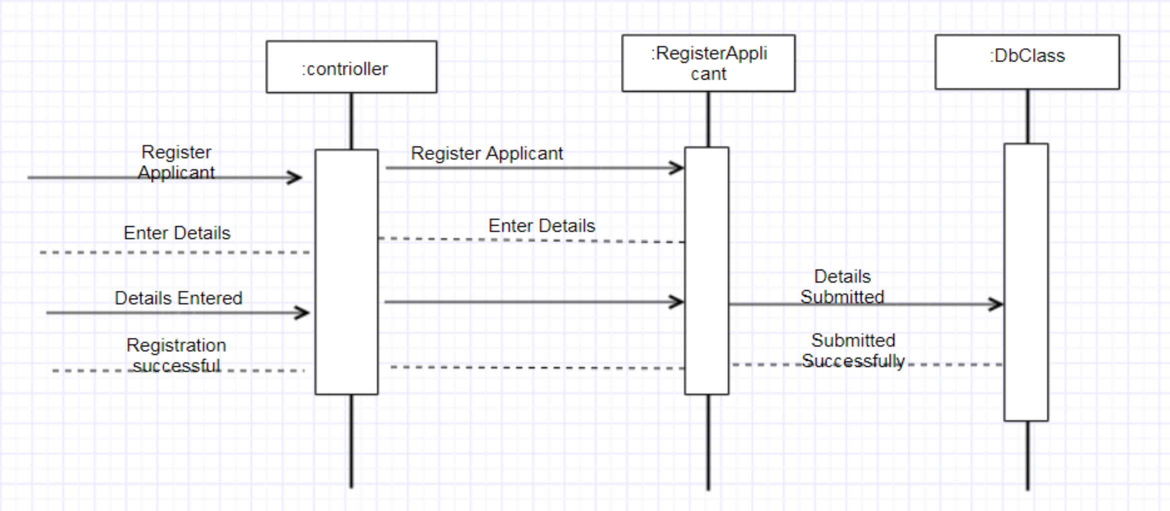


Figure 23 Register Applicant

3.9.3 Register Advertiser

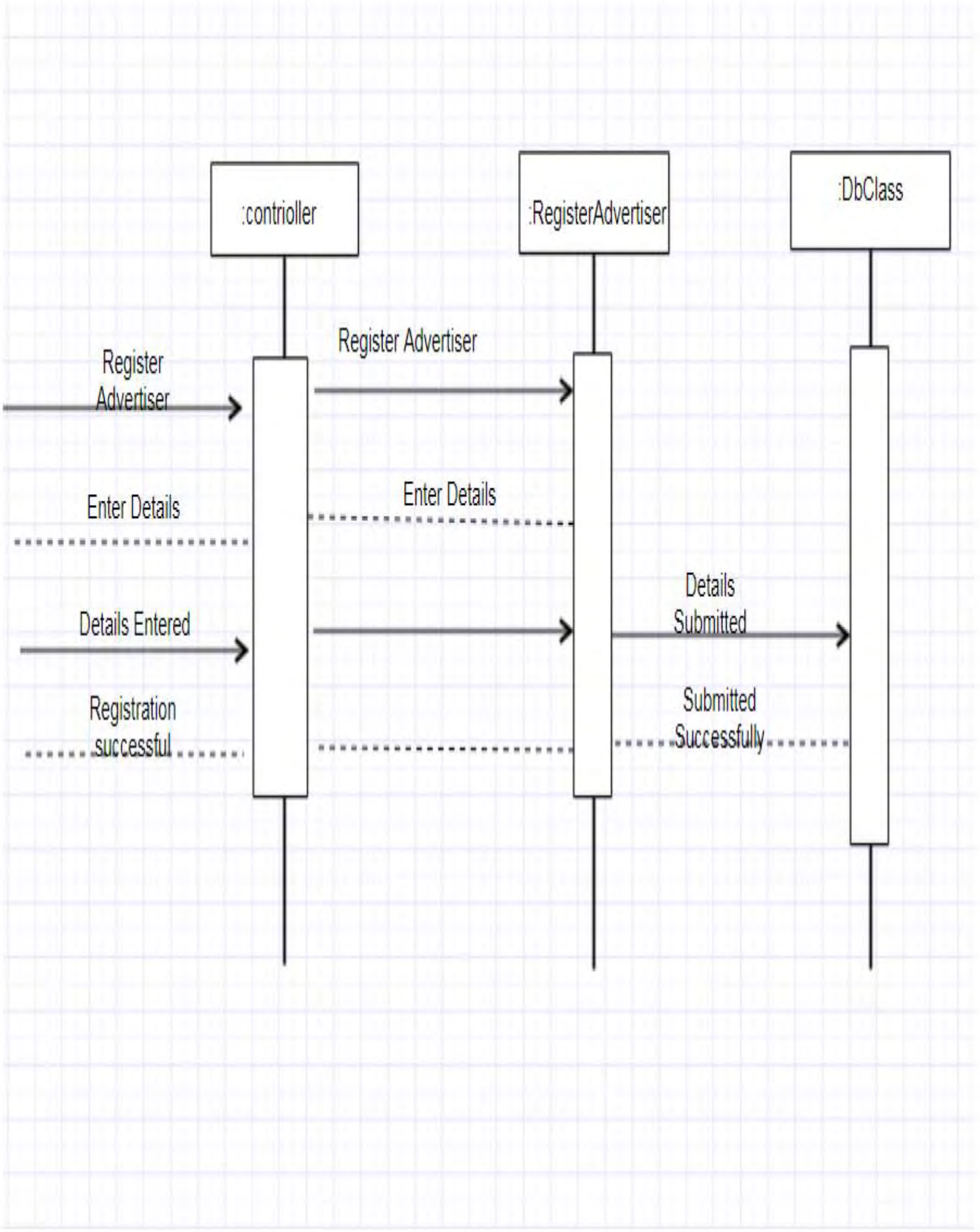


Figure 24 Register Advertiser SD

3.9.4 Search Applicant

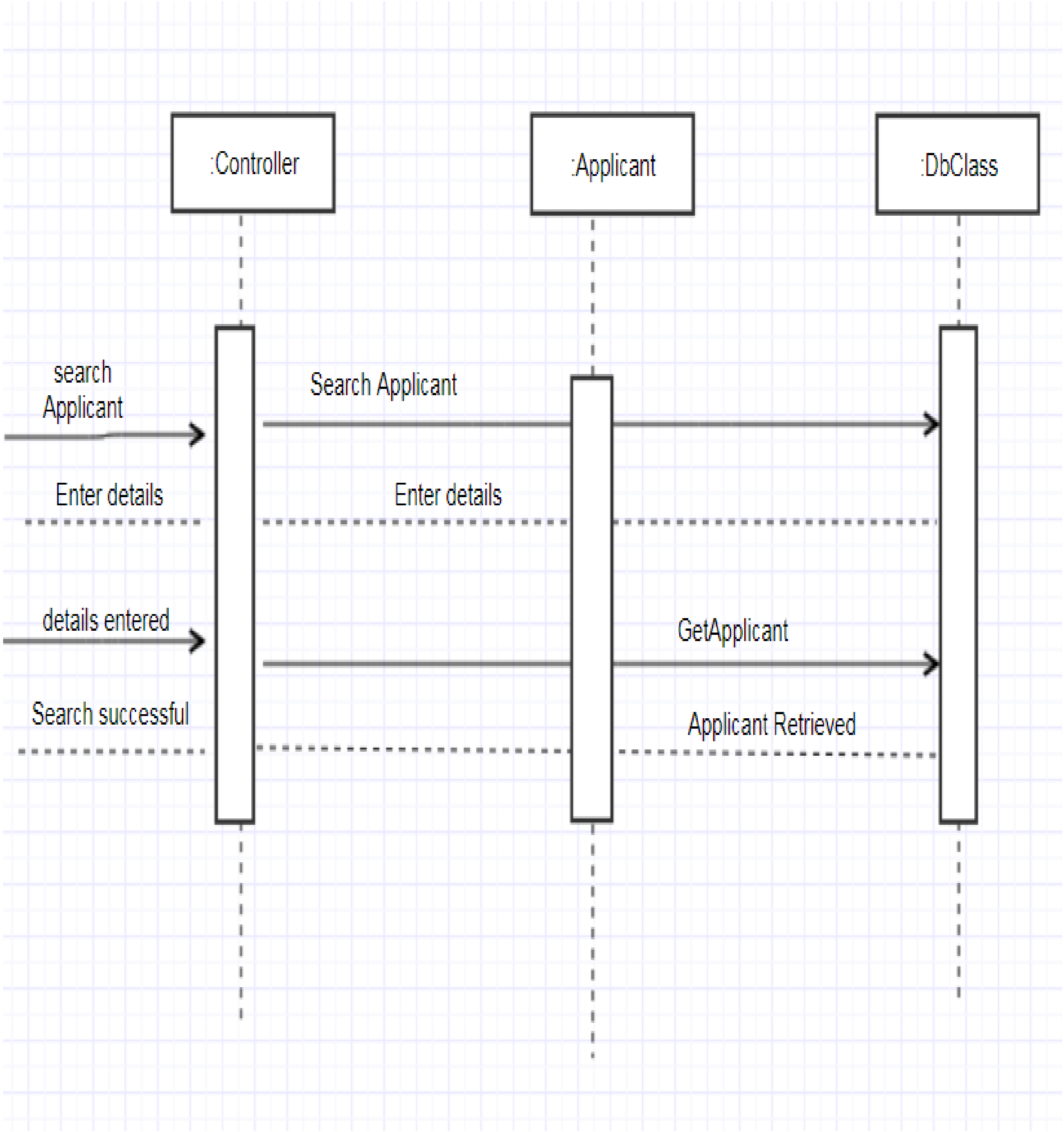


Figure 25 Search Applicant SD

3.9.5 View Applicant

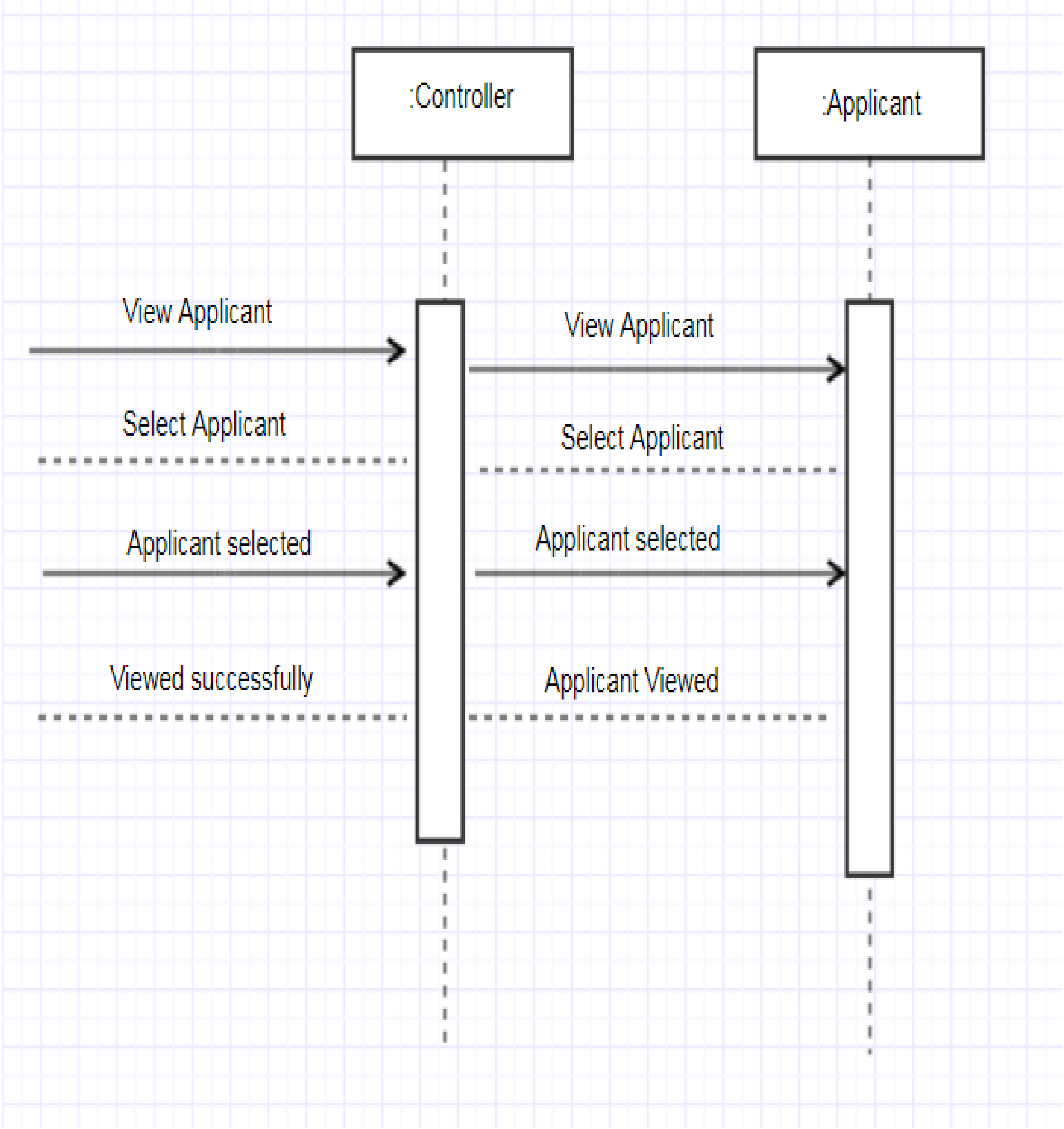


Figure 26 View Applicant SD

3.9.6 Search Advertisement

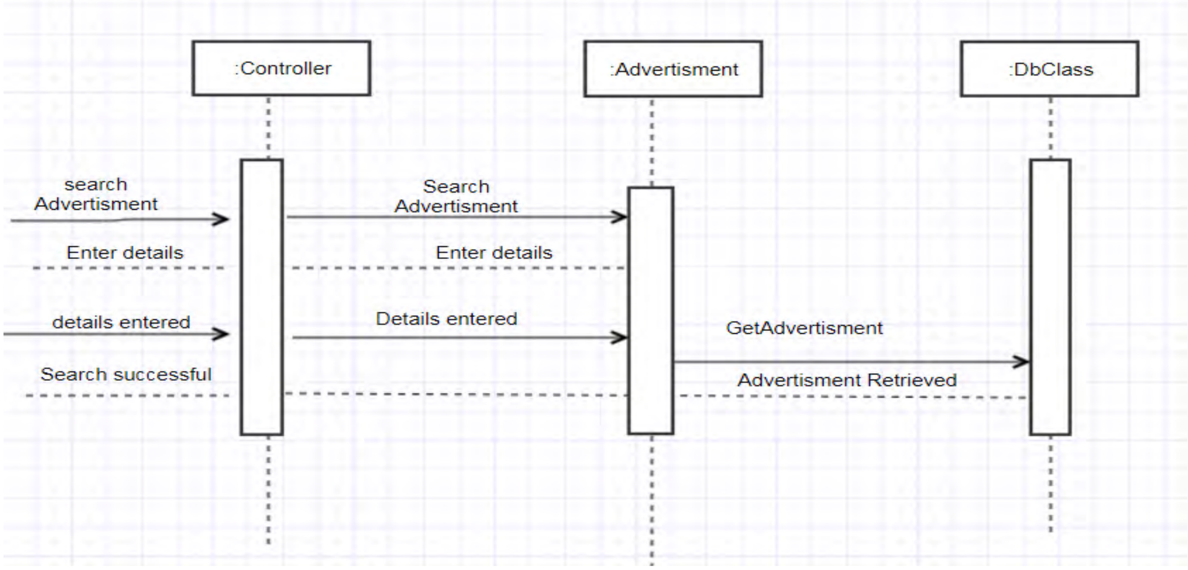


Figure 27 Search Advertisement SD

3.9.7 View Advertisement

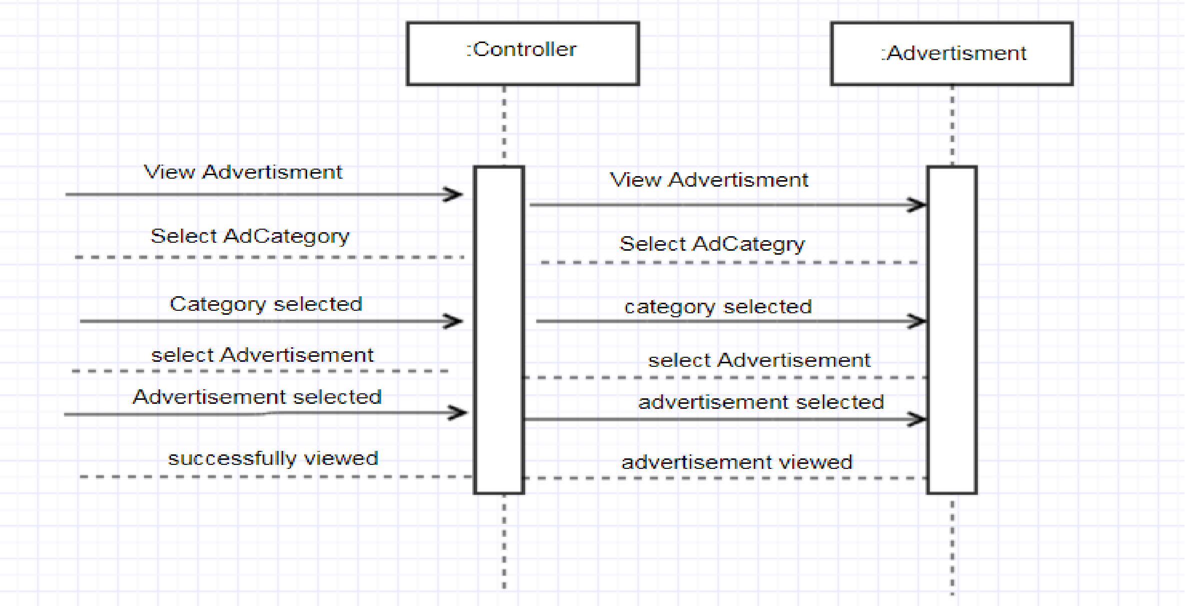


Figure 28 View Advertisement SD

3.9.8 Post Advertisement

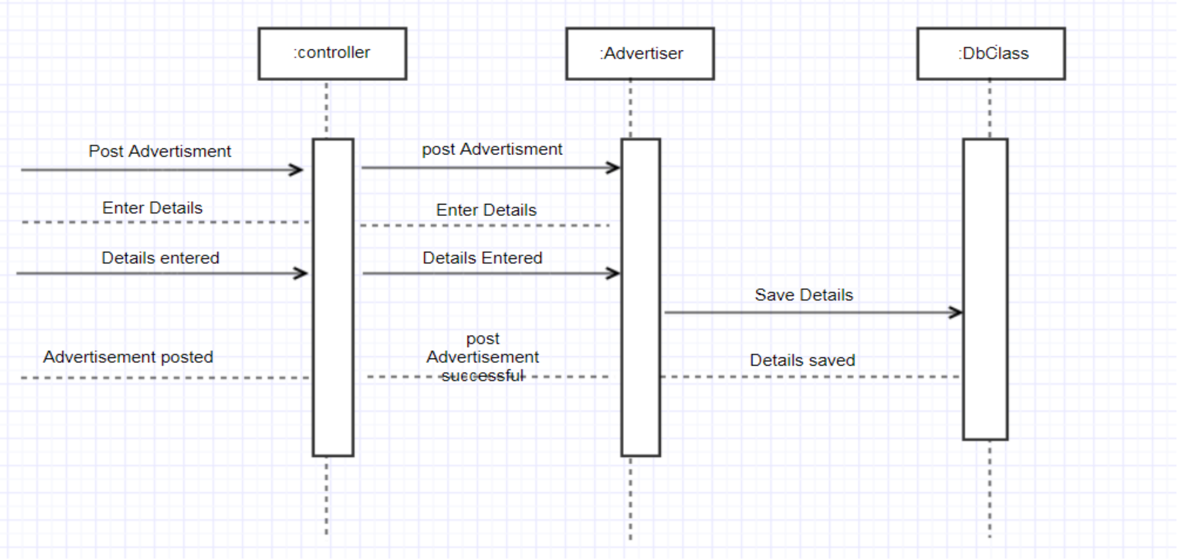


Figure 29 Post Advertisement SD

3.9.9 Accept Message Request

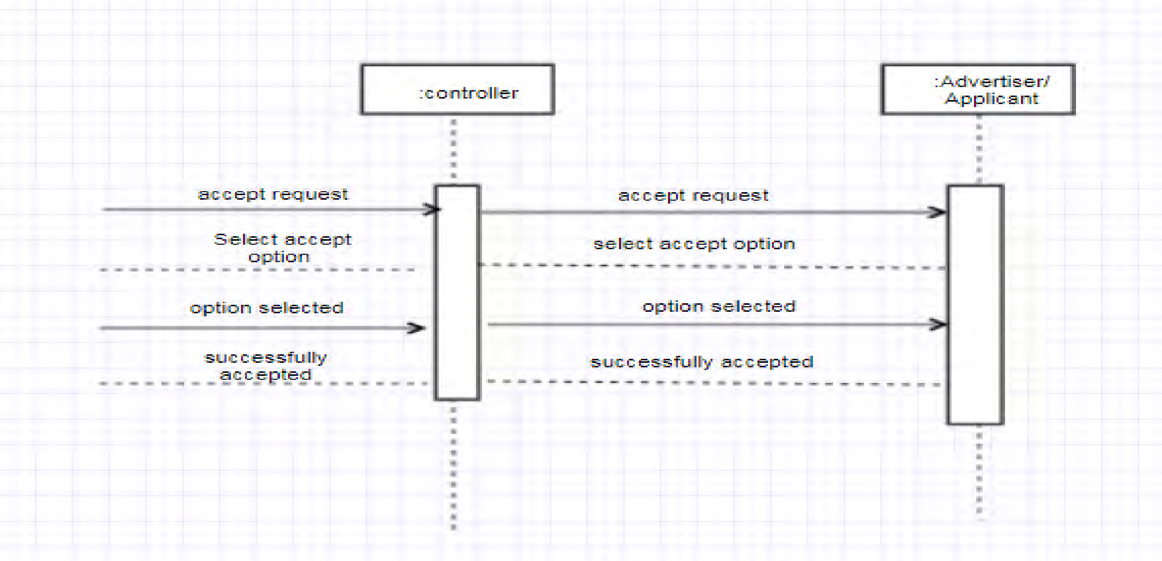


Figure 30 Accept Message Request SD

3.9.10 Send Message

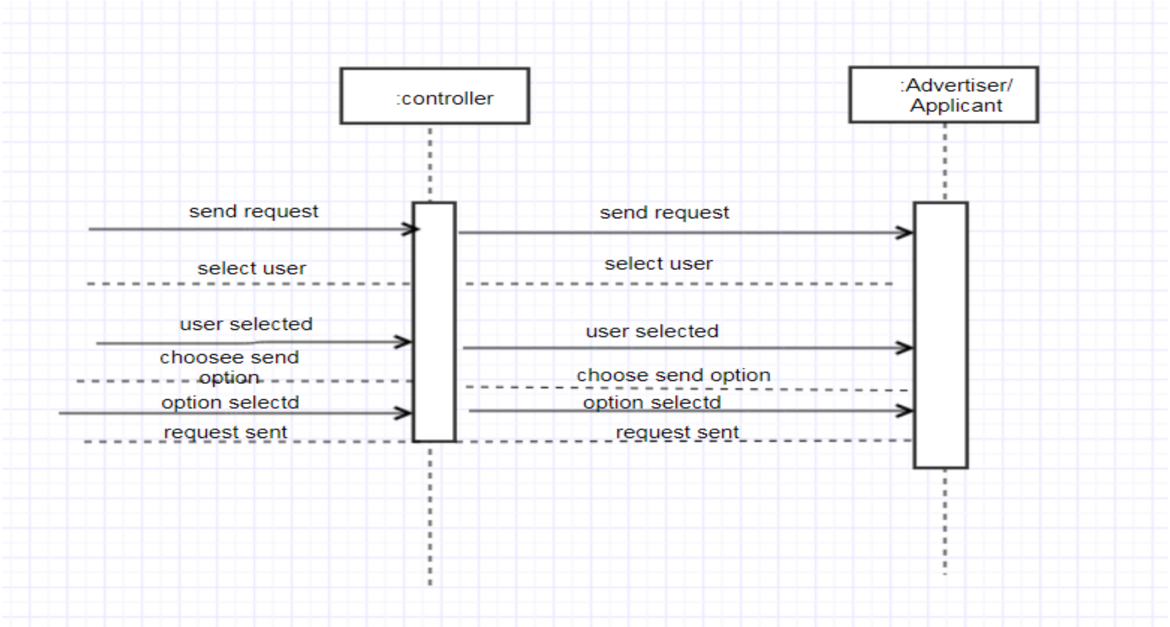


Figure 31 Send Message SD

CHAPTER 4

SOFTWARE TEST DOCUMENT

This chapter illustrates which approach will be used in this project for testing my application, testing tools and techniques and test cases.

4.1 Test Approach

A test approach is the test strategy implementation of a project and defines how testing would be carried out. Test approach has two techniques:

Proactive: An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.

Reactive: An approach in which the testing is not started until after design and coding are completed.

The reactive approach has been used in this project because by this approach, we can analyze the field or tool expert‟s knowledge extremity. We can analyze various risks linked to the project.

4.2 Test Plan

4.2.1 Testing Technique

Testing technique I will use would be black box testing technique in which the internal structure of item being tested will not be visible as the name indicates that so tester will not be able to see inside just like black box. This technique can be used to test both functional and nonfunctional requirements of the system and can indicate errors and missing functions etc.

4.2.2 Test Cases

- **Log In**

Table 4.1 Login TC(a)

ID	A001
Description	Users can Log in to the application
Tester	User
Set Up	Register user with id bubblymalik@gmail.com and password 1234.
Instruction	1. Enter ID bubblymalik@gmail.com 2. Enter Password 1234 3. Click log in button.
Expected Result	User logged in successfully.
Oracle	Pass

- **Log in (Alternative Scenario)**

This test case will describe the alternative scenario of logging in to the system. The entered username and password will be checked from database if it wouldn‟t be found than user cannot logged in.

Table 4.2 Login TC(b)

ID	A002
Description	User cannot log in to the system.
Tester	User
Set Up	Register user with ID bubblymalik@gmail.com and password 1234.
Instruction	1. Enter ID bubblymalik@gmail.com 2. Enter password 123. 3. Click Log in button.
Expected Result	1. User with Id bubblymalik@gmail.com cannot be logged into the system. 2. There is an error in ID or password.
Oracle	Fail

- **Register Applicant**

This test case will tell about the success scenario of registering user when user fills in the required data correctly.

Table 4.1 (a)Register Applicant TC

ID	A003
Description	Applicant successfully registered with ID bubblymalik@gmail.com and password 1234.
Tester	Admin
Set Up	Applicant fills in registration form completely.
Instruction	1. Applicant enters personal details correctly. 2. Applicant enters academic details. 3. Applicant fills in contact information.
Expected Result	1. Admin verify the applicant. 2. Applicant data saved in database. 3. Applicant registered successfully
Oracle	Pass

- **Register Applicant (Alternative scenario)**

This test case will elaborate the alternative scenario of registering applicant when applicant enters incorrect cnic, applicant will not be registered.

Table 4.2 (b)Register Applicant TC

ID	A004
Description	Applicant will not be registered.
Set Up	Applicant fills in the form completely but incorrect.
Instruction	1. Applicant fills in personal information completely. 1.1) Applicant enters invalid cnic. 2. Applicant fills in academic details. 3. Applicant fills in contact information.
Expected Result	1. Admin dismiss applicant‟s registration due to invalid cnic. 2. Data will not be saved to database. 3. Applicant does not get registered.
Oracle	Fail

- **Register Advertiser**

This test case will tell about the success scenario of registering advertiser when user fills in the required data correctly.

Table 4.3 (a)Register Advertiser TC

ID	A005
Description	Advertiser successfully registered with ID ahmedali@gmail.com and password 1234.
Tester	Admin
Set Up	Advertiser fills in registration form completely.
Instruction	4. Advertiser enters name correctly. 5. Advertiser enters cnic . 6. Advertiser fills in contact information.
Expected Result	4. Admin verify the Advertiser. 5. Advertiser data saved in database. 6. Advertiser registered successfully
Oracle	Pass

- **Register Advertiser (Alternate scenario)**

This test case will elaborate the alternative scenario of registering advertiser when advertiser enters incorrect contact, applicant will not be registered.

Table 4.4 (b)Register Advertiser TC

ID	A005
Description	Advertiser will not be registered.
Tester	Admin
Set Up	Advertiser fills in registration form completely.
Instruction	1. Advertiser enters name correctly. 2. Advertiser enters cnic. 3. Advertiser fills in contact information. 3.1) Advertiser enters invalid phone number.
Expected Result	7. Admin checks the data and dismiss advertiser. 8. Advertiser data will not be saved in database. 9. Advertiser will not be registered.
Oracle	Fail

- **Post Advertisement**

This test case will define the successful scenario of posting advertisement by advertiser. Advertiser enters all the particulars and advertisement is stored in database successfully.

Table 4.5 (a)Post Advertisement TC

ID	A007
Description	Advertisement will be posted successfully.
Set Up	Log in as a registered advertiser.
Instruction	1. Advertiser selects post advertisement option. 2. Advertiser enter details like posting date, valid till and category. 3. Advertiser clicks post now. 4. Admin allows the advertisement. 5. Advertisement saved in database.
Expected Result	Advertisement will be posted successfully.
Oracle	Pass

- **Post Advertisement (Alternative scenario)**

This test case will define the alternate scenario of posting advertisement by advertiser. Advertiser enters all the particulars and advertisement is stored in database successfully.

Table 4.6 (b)Post Advertisement TC

ID	A008
Description	Advertisement will not be posted successfully.
Set Up	Log in as a registered advertiser.
Instruction	1. Advertiser selects post advertisement option. 2. Advertiser enter details like posting date, valid till and category. 3. Advertiser clicks post now. 4. Admin checks the advertisement. 4.1) Advertisement is not valid. 4.2) Advertisement already exists.
Expected Result	Advertisement is not posted successfully.
Oracle	Fail

- **Update Skills**

This test case defines the deletion of advertisement by admin from database.

Table 4.7 Delete Advertisement TC

ID	A009
Description	Skills will be updated.
Set Up	Admin will be logged in with Admin rights.
Instruction	1. Admin selects add skills option. 2. Admin adds skills in the database. 3. Admin clicks update option.
Expected Result	Skills updated successfully.
Oracle	Pass

- **View Advertisement**

This test case will describe the successful viewing of the selected advertisement.

Table 4.8 View Advertisement TC

ID	A010
Description	Advertisement will be viewed.
Set Up	1. Applicant should be logged in as a registered user. 2. Applicant has searched the advertisement or it is already been shown on the home page.
Instruction	1. Applicant selects the advertisement.
Expected Result	Advertisement is viewed
Oracle	Pass

- **Search Advertisement**

This test case will describe the searching of advertisement. User selects the category and then search the advertisement; all the posted advertisements will be shown along with related ones.

Table 4.9 Search Advertisement TC

ID	A011
Description	Advertisement will be searched.
Set Up	1. Applicant should be logged in.
Instruction	1. Applicant clicks on search bar. 2. Applicant selects category to be searched.
Expected Result	All the advertisements of the related category will be displayed.
Oracle	Pass

- **Search Applicant**

In this test case success scenario of searching is described, advertiser search the applicants based on qualification and skills.

Table 4.10 Search Applicant TC

ID	A012
Description	Applicant will be searched.
Set Up	Advertiser will be logged in already.
Instruction	1. Advertiser selects search. 2. Advertiser enters the criteria.
Expected Result	List of Applicants will be shown.
Oracle	Pass

- **View Applicant**

This test case describes the success scenario of viewing applicant. Advertiser selects the applicant profile and profile will be viewed.

Table 4.11 View Applicant TC

ID	A013
Description	Applicant will be viewed.
Set Up	1. Advertiser will be logged in. 2. Advertiser has already searched the applicants, list of applicants is shown.
Instruction	1. Advertiser selects applicant profile.
Expected Result	Profile is viewed
Oracle	Pass

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENTS

5.1 Introduction

This document describes project conclusions and future enhancements in my project that what type of features can be added in my project can be added later on to improve it.

5.2 Summary

This application has removed the need the of going door to door for finding jobs manually, it is an android based application that can provide a platform for posting and searching jobs at a same time for two parties i.e. advertiser and applicant (transgender).

5.3 Conclusion

- This application provides the security and protect cnic and phone numbers of applicants as there is a chatting module in app and no need to contact through mobile numbers beforehand and face complications in future.
- Users can contact each other through application through messaging.
- Users can stay up to date about new jobs in town.
- Expired advertisements can be removed automatically on their expiration date.

5.4 Future Enhancements

This application can be enhanced in certain ways in future.

- This application can also be a paid one after sometime and can be a part of play store applications with certain improvements.
- I will develop this application in urdu language with further modifications.
- This application can be developed further for iOS.

References

[1] Software Project Management Plan IEEE 1058-1998.

[3] Software Test Documentation [Online] http://softwaretestingfundamentals.com/test-case/

[4] Test Plan [Online] https://en.wikipedia.org/wiki/Test_plan

[5] User Interface Design https://www.usability.gov/what-and-why/user-interface-design.html

[7] https://en.wikipedia.org/wiki/System_sequence_diagram