

Chapter 1. Introduction

Introduction

Android Based Tutor Recommendation System is an Android application built for bringing students and tutors at a common platform.. It is an unique android application where students and tutors can connect to each other based on their geographic locations. Students can enter their requirements and tutors can also list themselves providing important attributes like their qualification, specialization and fees.

1.1 Overview

Android Based Tutor Recommendation System is a system developed for finding tutors online and it also provides and also provides a platform to tutors to increase their reach without spending on advertising. Students and Tutors will be connected to each other and will be preferred based on their geographic locations. Feedbacks will be provided by students to tutor(s) in the form of “Star Ratings” which will help other students to choose among the pool of tutors.

This specific topic is chosen because students face a lot of problem in finding tutors, also some people have time and skills to teach but find it hard to find students. So, this project will bridge the gap between students and tutors.

1.2 Background and Motivation

The process of finding tutors manually has become an outdated approach. It is not only an hectic task but even sometimes leads to wastage of money and time.

Also, some students/tutors who have the skill to teach and also have time but fail to find students who need tutors.

To overcome all these drawbacks and many more than that we intend to develop a system which not only removes the tedious task of finding tutors manually but also helps in maintaining a system which will help in proper & appropriate evaluation of tutor's work , regularity and even punctuality of a tutor based on feedback provided by the students. The

purpose of developing Android Based Tutor Finder System is to computerized the tradition way of finding tutors.

1.3 Problem Statement and Objectives

Suppose a person 'X' is currently in higher studies or looking for job. Now he has to bear more or less of his expenses. Tuition finder will help him find a tuition where he can use his educational skills and experiences.

Now, let a person 'Y' is a concerned guardian or student who is desperately seeking for someone with proficient skills and knowledge to help in his studies.

This app will help them finding what they want. This app will also facilitate them with the privilege of finding tutor/tuition associated with subject preference, location, salary etc.

There are various problems in the existing system:

- Tiredness in finding tutors offline/manually.
- No proper GPS – based Mobile application is present in our city for Finding Tutors
- Existing systems do not make use of “Star Ratings” to give feedback to several tutors.

The objective of our project are as follows:

- To develop an android application to bring students and tutors on a common platform.
- To categorize tutors based on their geographic locations.
- To ensure that tutors are entering verified and genuine information.

1.4 Scope of the Project

As the project uses several specifications for finding tutors online, it has a wide variety of applications. Some of them are given below:

- Students can find tutors online based on their geographic locations and nearby areas.
- Tutors can be rated with the help of “Star Ratings” which can be used to provide feedbacks by the students.
- This project can be deployed in various cities covering different locations as tutors go on listing themselves. It can be used in any place that has tutors and their demand.

1.5 Project Description

It will be an android based application where two interfaces will be built-one for the students and other will be for the tutors.

Students can sign up and enter their basic details like name, class, subjects they wish to study, and other preferences. On the other hand, tutors will sign up and enter their details, validate their qualifications, details about the subjects they can teach, location preferences and their fees.

After this, students will be connected to several tutors according to their geographic locations and several other filters mentioned above. They will get details of every tutor from their area. Students interested in any tutor can connect with him/her through their contact details mentioned.

Students will be able to give feedback to tutor(s) with the help of Star Rating available in the application. This feature will help others to decide among pool of tutors available. For marketing of our project and to connect more and more students as well as tutors, we will advertise our project in several educational institutions starting with our college. In this tutor finding android application students and the faculty and students can register. The search of the subject expert is also filtered based on the location so that it will be easy for the finders to select the appropriate tutor. Based on the students selection the tutors will be made available for their special classes and then they will be intimated through notifications. This system provides an efficient search and gives effective result for the users. Future scope of the project is that in future the search can be completed using Algorithms.

1.6 Team Organization

Amay Dubey, Avinash Jaisingh, Aastha Hurkat, Ishan Sharma

Along with doing preliminary investigation and understanding the limitations of current system, we studied about the topic and its scope and surveyed various research papers related to the project and the technology that is to be used. Documentation was also a part of the work done by us in this project. The synopsis, project presentation and other documentation related work was also made by us.

Divesh Sharma, Kajal Dhanotia

We investigated and found the right technology and studied in depth about it. For the implementation of the project, we collected the object data and trained the model for it. Implementation logic for the project objective and coding of internal functionalities was also done by us. Also, we worked on backend design for storing results in database for maintaining logs.

1.7 Report Structure

The project *Android Based Tutor Recommendation System* is primarily concerned with the and whole project report is categorized into five chapters.

Chapter 1: Introduction- introduces the background of the problem followed by the objectives for the project undertaken. The chapter describes the scope and applications of the project. Further, the chapter gives the details of team members and their contribution in development of project which is then subsequently ended with report outline.

Chapter 2: Review of Literature- explores the work done in the area of Project undertaken and discusses the limitations of existing system and highlights the issues and challenges of project area. The chapter finally ends up with the requirement identification for present project work based on findings drawn from reviewed literature.

Chapter 3: Proposed System - starts with the project proposal based on the requirements identified, followed by the benefits of the project. The chapter also illustrate software engineering paradigm used along with different design representations. The chapter also includes diagram and details of major modules of the project. Chapter also gives insights of different type of feasibility study carried out for the project undertaken.

Chapter 4: Implementation - includes the details of different Technology/ Techniques/ Tools/ Programming Languages used in developing the Project. The chapter also includes the different user interface designed in project along with their

functionality. Further it discuss the experiment results along with testing of the project. The chapter ends with evaluation of project on different parameters like accuracy and efficiency.

Chapter 5: Conclusion - Concludes with objective wise analysis of results and limitation of present work which is then followed by suggestions and recommendations for further improvement.

Chapter 2. Review of Literature

Review of Literature

Now-a-days these computers and internet are providing every information as a knowledge source. Even though everything is automated in the field of learning we prefer for traditional method. In this busy world searching of tutors for any subject is a very difficult job. In order to make it simple here with we proposed an idea to find a subject expert tutor through android application. The proposed work has the common platform where the teacher and the student can access on their respective available views.

2.1 Preliminary Investigation

The existing system for communication between guardian and tutor is via tuition media or by passing the tuition news to known people. The first one is not so dependable though they charge a big percentage of tuition fee from both teacher and tutor. And sometimes they are unprofessional-tend to pass high value tuition to less qualified tutors. Also they can't cover 100% of the potential tuition opportunity. The second one covers much less. So the probability of finding a good tuition is quite low here. The existing system present for finding tuitions is mainly manual. People have to go through their contacts and trust on the mouth publicity of tutors. This application has been mainly developed by keeping in mind the college students who can teach school students.

2.2 Limitations of Current System

All the existing systems do not categorize on the basis of location which is significant part of our application.

2.3 Survey of Existing Systems

Sno	Existing system	Features	Disadvantages	Limitations
1.	www.tutor.com	Online tutoring available, feedback system	Only online functioning, only website	Does not categorize students based on their location.
2.	TutorFinder Application	Chat system available	No verification/validation of tutors	Not popular in India
3.	BuddySchool.com	Website based, student feedback available	Limited number of subjects	Not functional in India, Only website functioning

Table No. 1- Survey of existing systems

The survey of existing systems brought us in terms with the loopholes of the functioning websites. The major reason we were able to detect, was that all these websites had no means for searching the tutor based on their locations. Also, these websites/applications were not functional in India. This way, we were able to identify the functional requirements of our project.

2.4 Requirement Identification and Analysis for Project

We have seen in our nearby surroundings that mainly college students are willing to give their time to use their knowledge and help other students who want tuitions in their school. They can use their knowledge to help them and also earn money side by side. With our application, we are providing them with a platform which will connect them. The tutors can enter the subjects they want to teach and the students can look up them through our android application.

The process of finding a tutor is time consuming. The rise in educational standards has led to the need for additional coaching along with basic schooling. An Android application for finding tutors can help a parent find tutors without consulting any third party.

2.5 User Requirements

The success of a project lies on the satisfaction of users. To develop a system we first need to find out the requirements of the different users so that full satisfaction of users can be attained which is a measure of successfulness of a project. Requirements are description that the system should do under some constraints. User requirements are statements in natural language along with diagrams of what services the system is expected to provide to system user and the constraints under these services. We have met different types of users of our system. We met different user groups of our system and discussed with them about their requirements. User requirements are as follow.

2.5.1 Functional Requirements:

Functional requirements are description of services. Besides it also includes how system will react for particular input and behave in particular situations. It is supported by non-functional requirements. The functional requirement of our system is given below:

1. The System shall avail user to sign up as provider into the system:
 - a. Any person throughout the country can open an account as a provider in our system by providing his/her basic information.
2. The system shall avail user to sign up as a tutor into the system.
 - a. Any user throughout the country can open an account in our system as a tutor by providing required information.
3. The system shall facilitate provider to post tuition advertisement.
 - a. Provider can post advertisement for a tuition by giving detailed information of the tuition and requirement of his/her desired tutor.
4. The system shall facilitate tutors to search for providers.
 - a. Tutors can search for desired tuition by selecting locality and other desired requirements of tuition.
5. The system shall facilitate tutors to get contact information of his/her desired tuition.

6. The system shall facilitate providers to update tuition status.
 - a. Provider can update tuition status by marking it as booked.
7. The system shall facilitate the providers to report any tutor for any misbehave.

2.5.2 Non-Functional Requirements:

Non-functional requirements are the criteria that can be used to judge the operation of a system. These criteria defines some constraints on the services and functions of the system. We have met user groups of our system and tried to find out non-functional requirements. These are enlisted below:

1. The system shall respond quickly, consume less memory.
 - a. The system shall start within 1 second.
 - b. The system size shall not exceed 20MB.
 - c. The system UI shall be user friendly.
 - d. The search result shall have to be shown in less than 1 second.
 - e. The duration of receiving contact information should not be more than 1 seconds.
2. The system shall fit screen of any type of device such as smartphone, tab etc.
 - a. The system shall support down to Gingerbread version of Android.
3. The system shall require internet connection for viewing information.
 - a. Acquires and updates information regarding providers list and their availability from software server via Internet.
4. The system data should be secure and reliable.
 - a. The system shall be amenable to the rules and regulations.
 - b. The data shouldn't be modified without login.
 - c. The data should not be contradictory.

2.6 Conclusion

This chapter reviews the literature surveys that have been done during the research work. The related work that has been proposed by many researchers has been discussed. The research

papers related to tutition finder application have been shown which discussed about different methods to build and online tutor recommendation portal.

Chapter 3. Proposed System

Proposed System

3.1 The Proposal

The proposal is to deploy a system to find tutors online which can help students to find tutors in their geographic area using GPS. The students can select the best tutor among the pool of available tutors on our application. Students can select tutors on the basis of location, fees, experience, subjects known, timings etc. Students can also rate tutors with the help of “Star Ratings” available on our platform.

3.2 Feasibility Study

A feasibility study is an analysis of how successfully a system can be implemented, accounting for factors that affect it such as economic, technical and operational factors to determine its potential positive and negative outcomes before investing a considerable amount of time and money into it.

3.2.1 Technical

Finding tutors online can be done through GPS enabled Android application. For this, the user would require an android smartphone with GPS enabled system. Internet is also required for the functioning of the Tutor Finder Application.

3.2.2 Economical

Finding tutors manually is very costly and time consuming, so we need a feasible solution to this problem and Android Based Tutor Finder System is a feasible application which is build in no cost and can be used for free. This application is economically affordable and does not incur any kind on hidden costs.

3.2.3 Operational

The main motto of our system is to reduce the manual efforts of finding tutors and finding students. The system is able to do that accurately and efficiently making the system operationally feasible.

3.3 Design Representation

3.3.1 Activity Diagram

Activity diagram is another important **diagram** in UML to describe the dynamic aspects of the system. **Activity diagram** is basically a flowchart to represent the flow from one **activity** to another **activity**. The **activity** can be described as an operation of the system. The control flow is drawn from one operation to another.

As soon as we launch the application, we can see the signup/login page. The signup/login interface is same for both tutor and the student. On logging in, the user has to pass the authentication steps, if not, then you might be entering incorrect details. The individual may sign up entering his/her details and preferences. The tutor may register himself as a tutor and mention qualification details and fee details. The student can then search the tutor based on their locality. All the registered tutor will appear in the search list as per their locality.

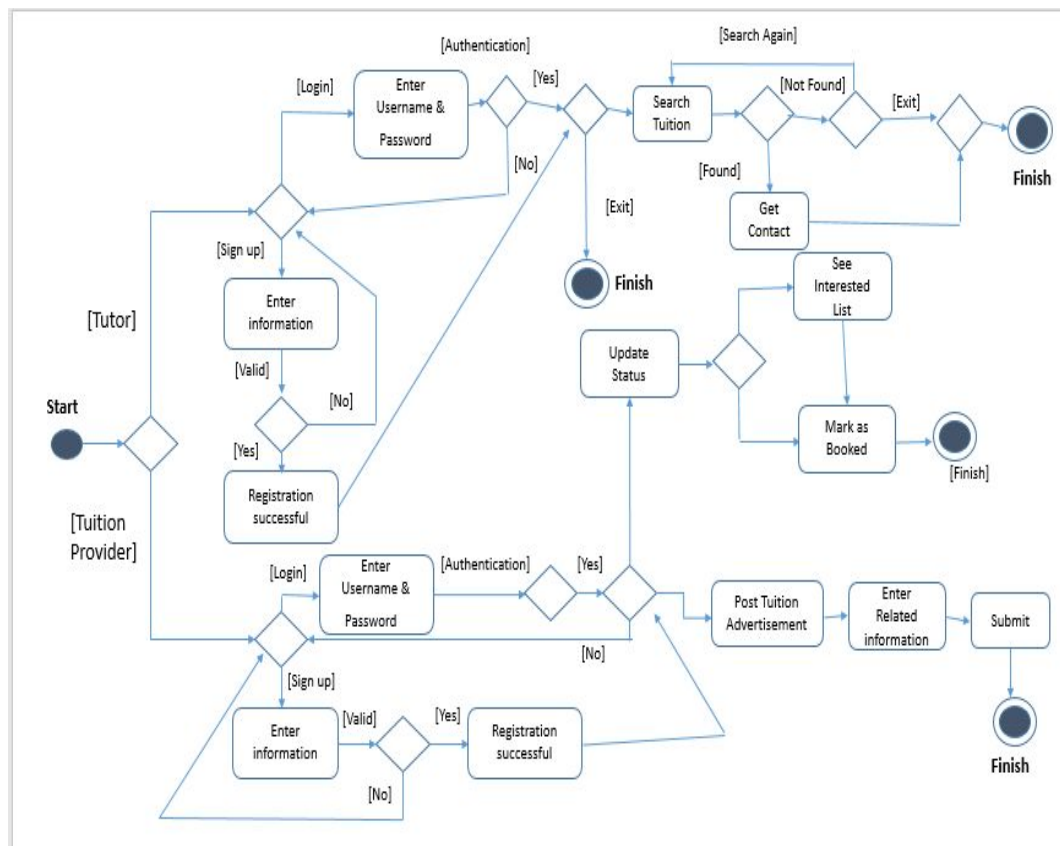


Figure-1: Activity Diagram

3.3.2 Use Case Diagram:

A use case represents the actors involved in an interaction. This is then supplemented by additional information describing the interaction with the system. The additional information may be textual description or one or more graphical models such as UML sequence or state charts.

We have found four main use cases of our system. These are listed below:

Use cases of our system:

1. Signing up as a provider.
2. Signing up as a tutor.
3. Posting advertisement.
4. Search for tuition.
5. Updating tuition status.

These use cases are described in detail in the following.

Use Case 1: Signing up as a provider.

Actor: Provider.

Precondition:

1. System displays home page of Tuition finder.

Main Success Scenario:

1. System displays “I am Tutor” and “I am Provider” options.
2. User clicks on the “I am Provider” option.
3. System displays provider’s activity page.
4. System shows options “log in” and “register”.
5. User clicks on “register” option.
6. System displays registration page containing input fields: “Full Name”, “Username”, “Password”, “Re-enter the password”, “Phone no”, “Email Address”, “Locality”.
7. User fills up the fields with accurate information.
8. User clicks on “Confirm”.
9. System authenticates that information provided by user are valid.

10. System displays message “successfully registered”.

Post Condition: User is successfully registered into the system as a provider.

Use Case 2: Signing up as a tutor.

Actor: Tutor.

Precondition:

1. System displays home page of Tuition Finder.

Main Success Scenario:

1. System displays “I am Tutor” and “I am Provider” options.
2. User clicks on the “I am Tutor” option.
3. System displays tutor’s activity page.
4. System shows options “log in” and “register”.
5. User clicks on “register” option.
6. System displays registration page containing input fields: “Full Name”, “Username”, “Educational Background”, “Password”, “Re-enter the password”, “Phone no”, “Email Address”.
7. User fills up the fields with accurate information.
8. User clicks on “Confirm”.
9. System displays message “successfully registered”

Post Condition: User is successfully registered into the system as a tutor.

Alternative Course:

2.a User clicks on “I am provider” option.

2.a.01. Use case ends.

5.a User clicks on “log in” option.

5.a.01 System displays log in page containing input fields “Username” and “Password”.

5.a.02 User fills up the fields with accurate information.

5.a.03 User clicks on “Confirm” from “Confirm” and “Cancel”.

5.a.04 System authenticates username and password as valid.

12.a User clicks on “Cancel”.

12.a.01 Resume at 4.

Use Case 3: Posting advertisement.

Actor: Provider.

Precondition:

1. User is successfully logged in into the system as a provider.

Main Success Scenario:

1. System displays two options “Update status” and “Post a tuition”.
2. User selects “post a tuition”.
3. System displays dropdown menus “group”, “class”, “locality”, “days in a week” and input fields levelled as “salary”, “subjects”, “contact number” and “other description”.
4. User fills up the input fields and dropdown menus with appropriate information and then clicks on “submit” button.
5. System displays message “Successfully posted”.

Alternative Scenario:

1.a.01. User click on “Update status”.

1.a.02. Use case ends.

Use Case 5: Search for tuition.

Actor: Tutor.

Precondition:

1. User is successfully logged in into the system as a tutor.

Main Success Scenario:

1. System displays dropdown menus “Locality”, “Group”, “Class Range”.
2. User selects from those dropdown menus according to his preference.
3. System displays the list of available tuitions of particular group and class range in a particular area.
4. User clicks on the “get contact” of provider of his/her choice.
5. System displays contact number of provider and his/her unique id to the provider.

Post Condition: Contact number of provider is successfully delivered to tutor.

Alternative Course:

- 4.a User clicks on “Exit”.
- 4.a.01 Use case ends.

Use Case 6: Updating tuition status.

Actor: Provider

Precondition:

1. User is successfully logged in into the system as a provider.

Main Success Scenario:

1. System displays two options “Update status” and “Post a tuition”.
2. User clicks on update status.
3. System displays options “See Interested tutors” and “Mark a tuition as booked”
4. User clicks on the “Mark a tuition as booked” option.
5. System shows list of tuitions posted by the provider.
6. User selects a particular tuition to mark it as booked.
7. System removes the associated tuition post from User Interface.

Post Condition: System displays message “Post has been removed”.

Alternative Course:

1.a User clicks on “Post a tuition”.

1.a.01 Use case ends.

3.a User clicks on “See interested tutors”.

3.a.01. System displays list of tutors, their names, educational background and tutor id.

3.a.02. User selects a tutor id and clicks on “report this tutor”.

3.a.03. System displays “Successfully Reported”.

UML Use Case Diagrams. Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors)

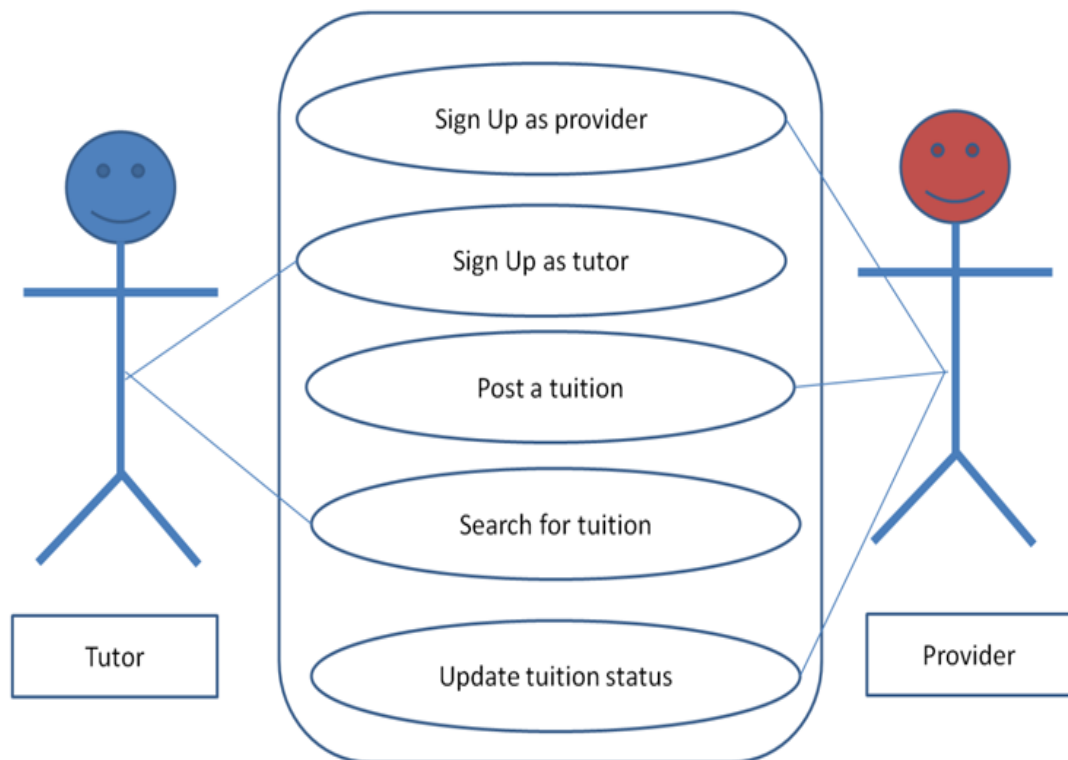


Figure-2:Use Case Diagram

3.3.3 Sequence Diagram

Sequence diagrams are sometimes called event **diagrams** or event scenarios. A **sequence diagram** shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur.

Here, the sequence diagram has one LifeLine/Actor and five activations, namely, SignUp, Output, Input, Login and Database. Upon launching, the application displays the Sign Up page. The user may sign up by filling in the necessary details. The takeinfo() message is passed. If the user tries to login, authenticate() is activated and database is checked for id password. If authentication is successful, the user is directed to his dashboard.

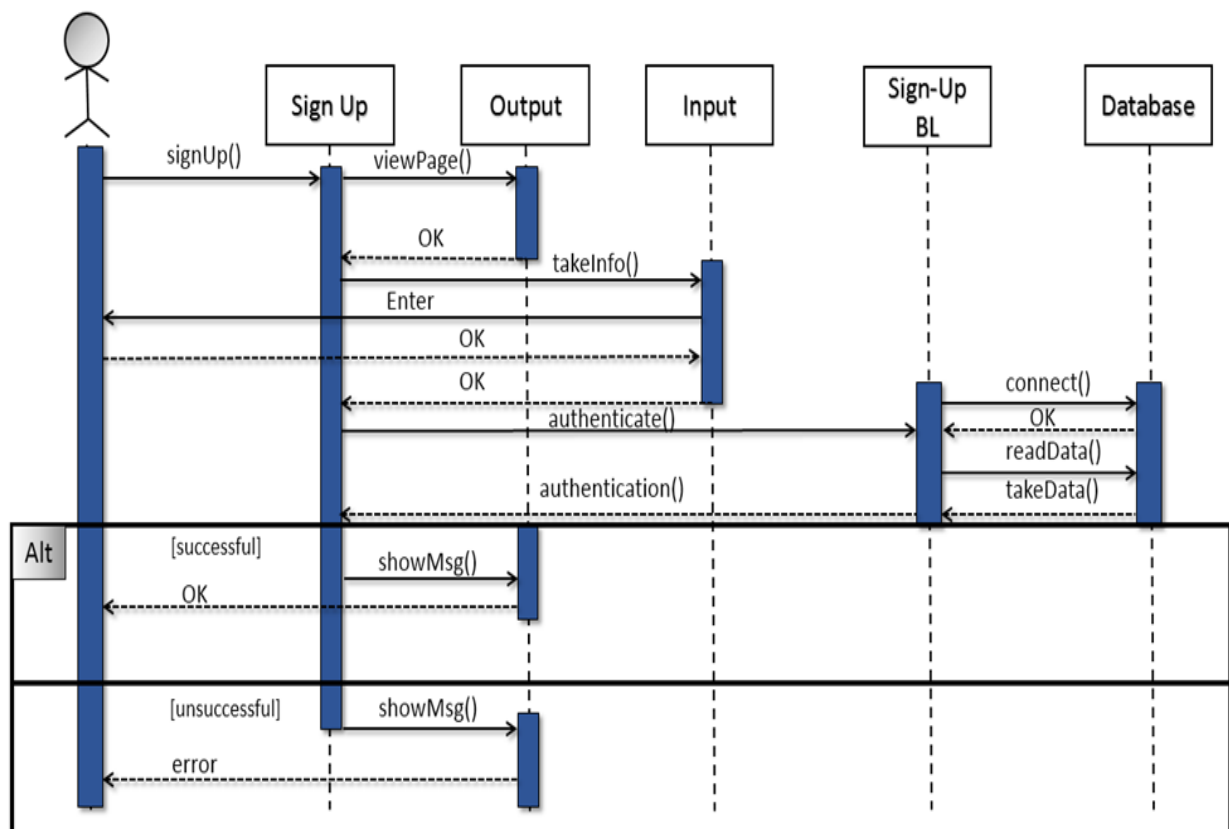


Figure-3: Sequence Diagram

3.3.4 Entity Relationship Diagram:

An **entity relationship diagram (ERD)** shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties.

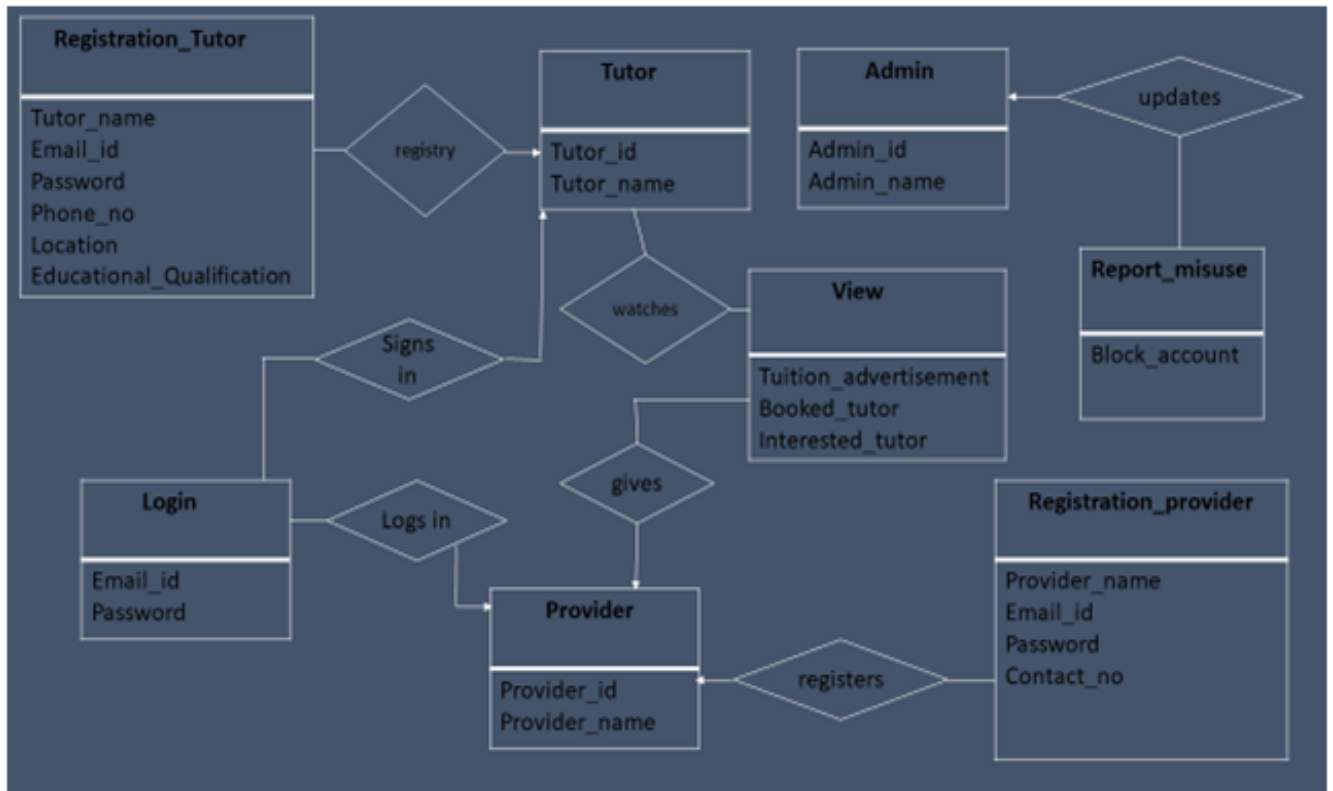


Figure-4: Entity Relationship Diagram

3.3.5 Flow Chart

A **flowchart** is a type of diagram that represents an algorithm, workflow or process. The **flowchart** shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem.

In our application, you may signup as a student and as a tutor. For signing up as a student, you need to enter the details required, and then you can view the list of registered tutors on the platform. You can also filter your search based on the preferred location. For signing up as a tutor, you have to enter the required details and then you can mention the number of days you are available, availability, qualifications, other preferences, etc. You can also update your profile at any given point of time.

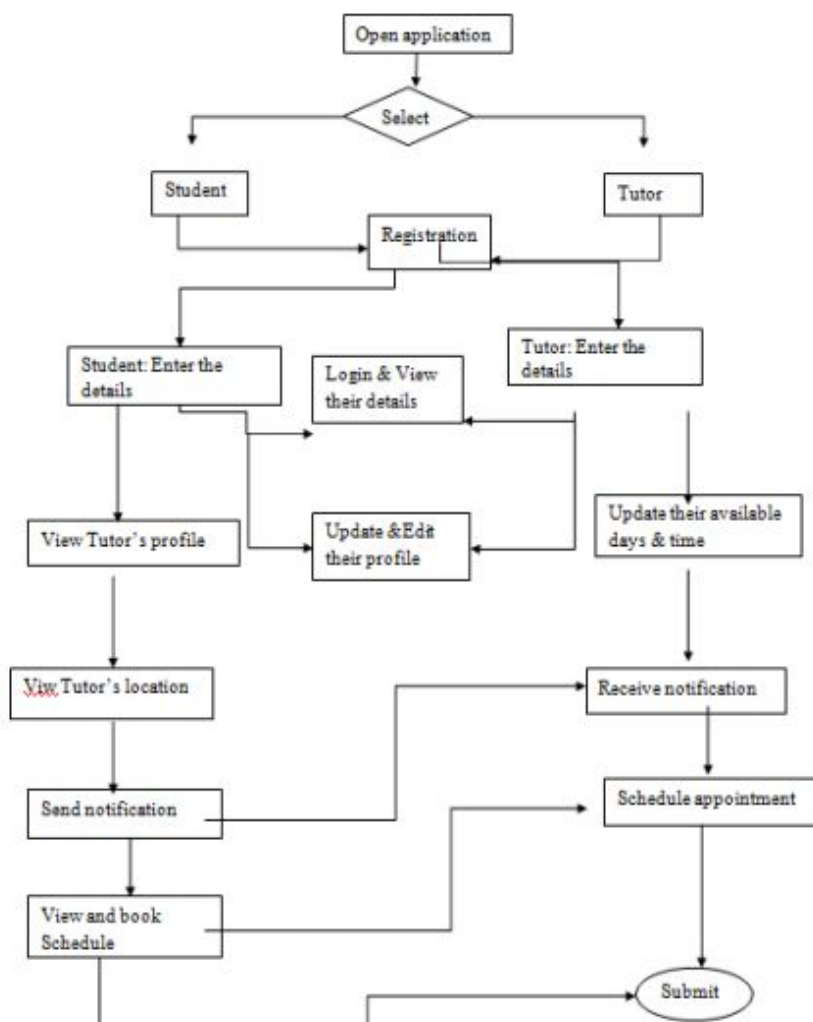


Figure-5: Flow Chart

3.4 Deployment Requirements

There are various requirements (hardware, software and services) to successfully deploy the system. These are mentioned below :

3.4.1 Hardware

Android mobile with android version greater than 4.0.0 and with GPS system properly enabled and working.

3.4.2 Software

1. Android Studio
2. Java JDK
3. Firebase

3.5 Benefits of the Proposed System

The current system had a lot of challenges that are overcome by this system :

- **Cost Effective :** The proposed system is cost effective as there will not be any need to go to several places and find tutors manually.
- **Ease of Finding Tutors:** Tutors can be found easily and can be chosen among the pool of available tutors.
- **Verified and Genuine Tutors:** Every tutor will be asked to submit their Photo Identity Card and Address Proofs such as AADHAR Card, Driving License, etc. This will stop tutors to a certain extent from doing something wrong.
- **Feedbacks:** Students can rate and give feedback to several tutors via “Star ratings” and this will help other students to select tutors among the pool of tutors.
- **GPS Based System :** Our application has GPS Enabled system which makes it easier to find tutors in our geographic location.

Chapter 4. Implementation

Implementation

For the problem of finding tutors/students manually, the system is designed in such a way so as to ease the process by using an application to find tutors/students via GPS.

4.1 Technique Used

Android Based Tutor Finder System consists of Android Technology, GPS system, Internet and Android System.

Android is a Linux based operating system for mobile devices that includes an operating system, middleware and key applications. It is based on the Linux Kernel and developed by Google and later the Open Handset Alliance (OHA). It allows writing managed code in Java language. Android is under version 2 of the Apache Software License (ASL). The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform. Android is a powerful Operating System supporting a large number of applications in Smartphones. These applications make life more comfortable and advanced for the users. Hardware that support Android are mainly based on ARM architecture platform. The world is experiencing accelerated growth in Smartphone ownership. As Smartphones has become more familiar to people and finding use in the day to day lives, their influence on society continues to grow. In our project we have developed a Smartphone application for finding tutors for students.

4.2 Languages Used

4.2.1 Java

Java is a set of computer software and specifications developed by James Gosling at Sun Microsystems, which was later acquired by the Oracle Corporation, that provides a system for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones

to enterprise servers and supercomputers. Java applets, which are less common than standalone Java applications, were commonly run in secure, sandboxed environments to provide many features of native applications through being embedded in HTML pages

4.2.2 XML

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The W3C's XML 1.0 Specification and several other related specification—all of them free open standards—define XML.

The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services.

4.2.3 Why Java?

Java is most preferably used for Android applications development because it is the native language being used. Java has huge open source support with many libraries and tools available to make a developer's life easier. Also, Java protects us from many problems in the inherent code like memory leaks bad pointer etc. Java also helps us create a better security model so that your app remains secure.

4.3 Screenshots

The Following are the screenshots of the result of the project.

There are six screens that our application displays:

- Main Page
- Login Page
- Common Interface Page
- Search Page
- Tutions List
- Update and Delete Page

4.3.1 Main Page

When user click on our system, the system show the home page which contains fields for user registration. Button for sign up and login and two edit text field get the sign up data. After filling up sign up field user click on signup button then our system stores the signup data into backend database. For backend database we have used firebase database. If the data stored successfully then our system shows the message “signup successful”. Below, this is our system sign-up page.

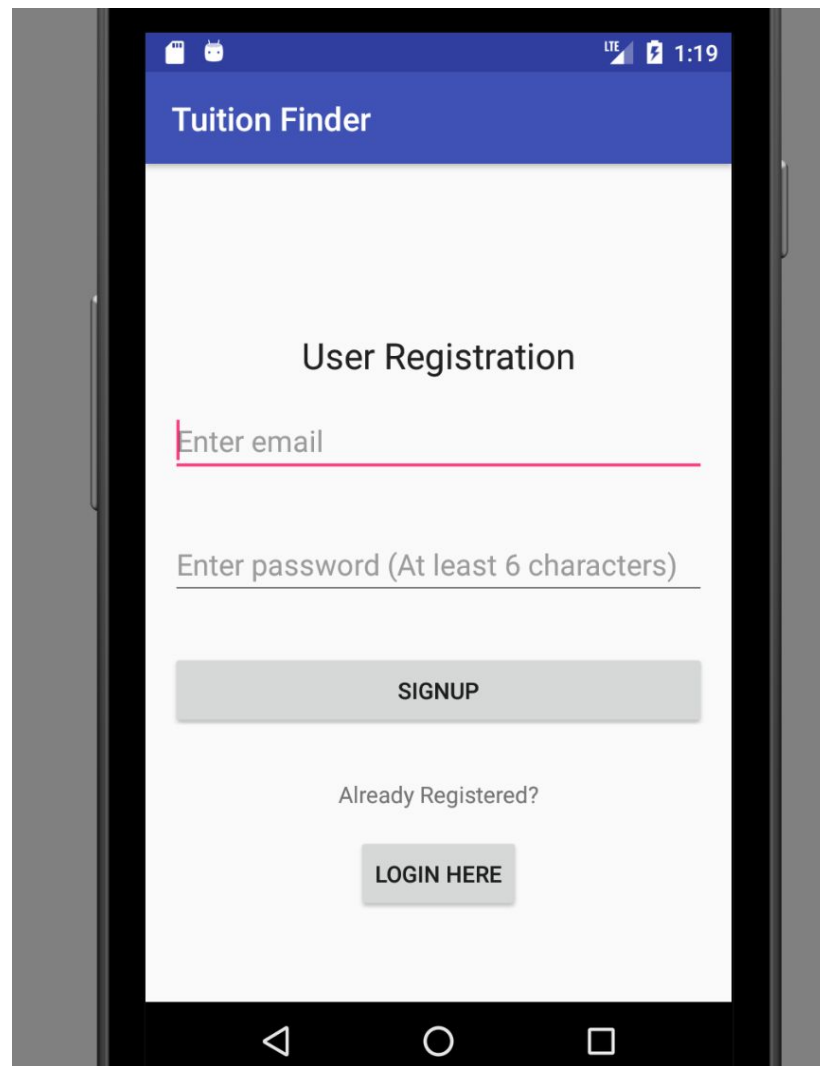


Fig:6- Registration Page

4.3.2 Login Page

When user click on login button, the system show the login page which contain 1 button and 2 edit text field. Button for login and two edit text field get the login data. After filling up edit text field user click on login button then our system get the email and password from backend database and check if those data match then system show the message as “login successful”

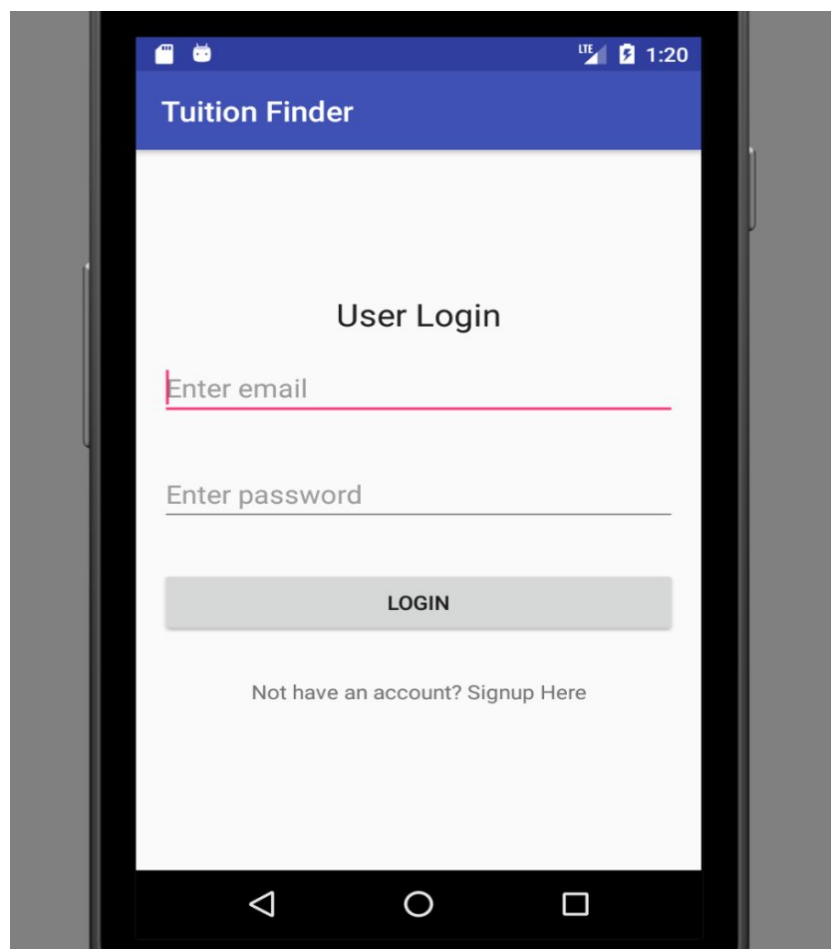


Fig:7- login page

If logged in successfully then system shows profile activity of the user.

4.3.3 Common Interface Page

After login successfully system show common interface page with message “Welcome to our app”. Main page contain 3 button. Button 1 for search tuition. Button 2 for post tuition. Button 3 for logout. User can choose one of them at a time.

This is our main page,

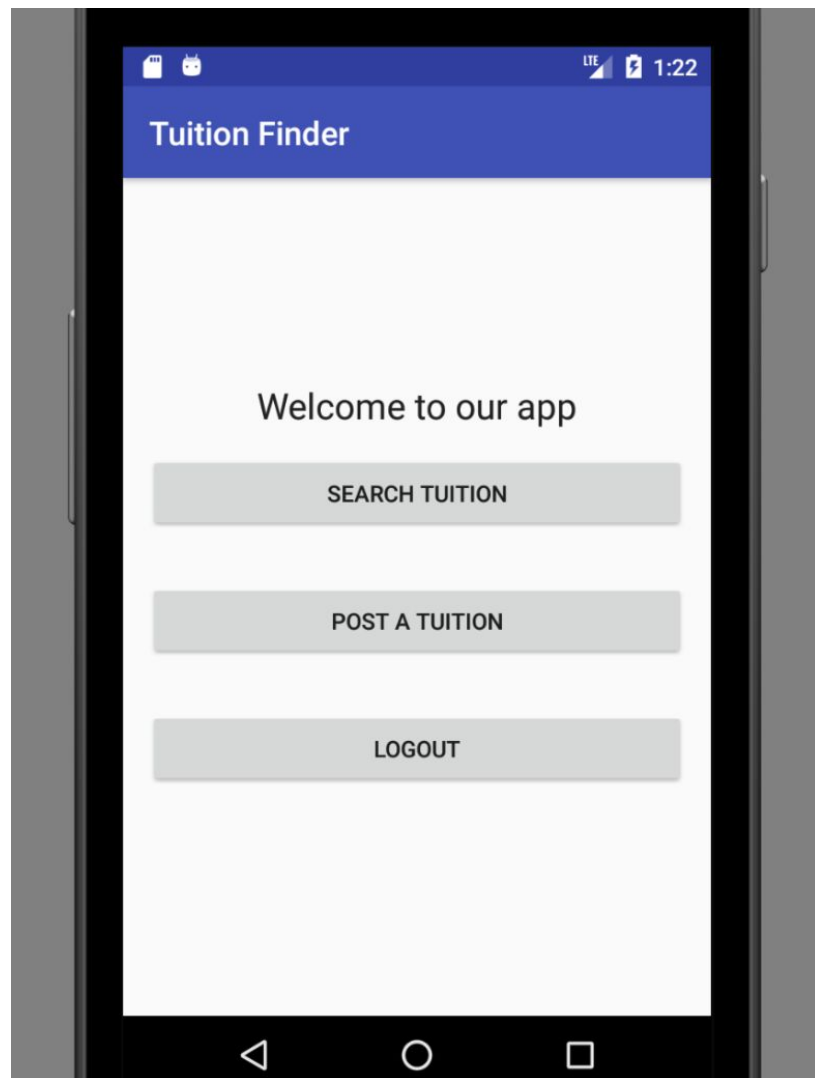


Fig:8- common interface page

4.3.4 Search Page

If user clicked on search button in main page then system shows search page, where user can search for tuition according to location

This is our search page

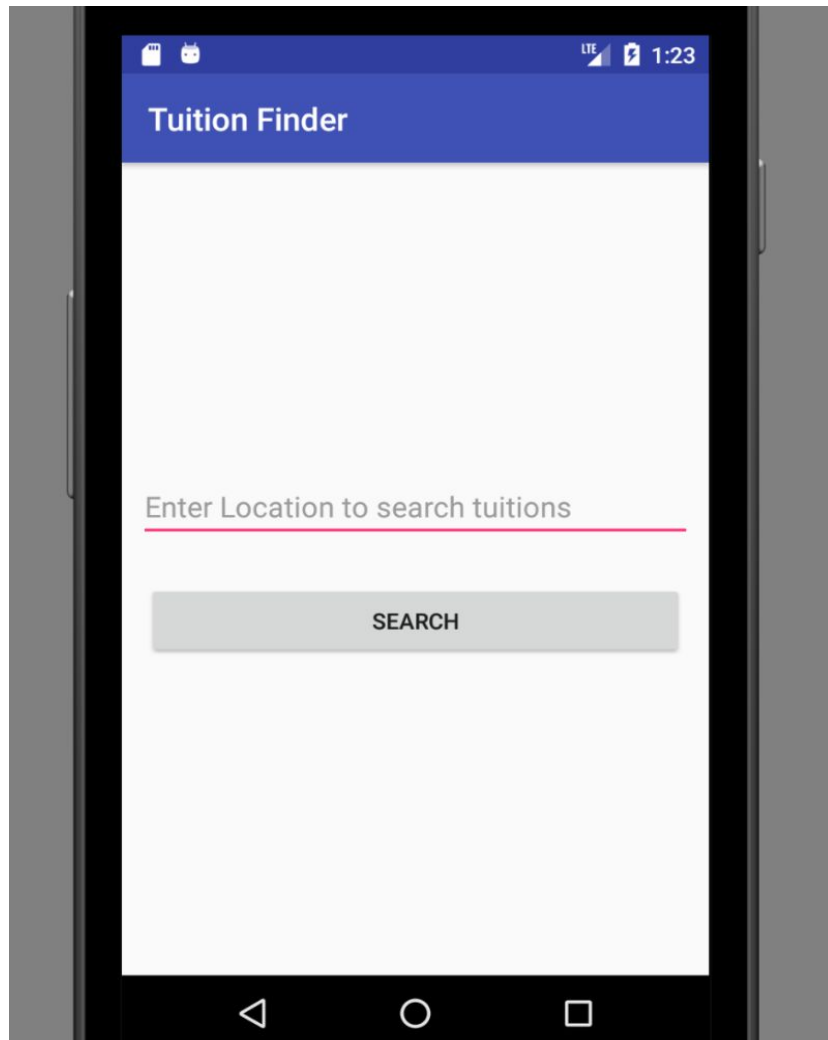


Fig:9- for search page

If user fill up edit text field with location then system shows the tuition list according to location.

4.3.5 Tutions List Page

When User clicked on search button in search page then system shows tuition list page. This is tuition list page

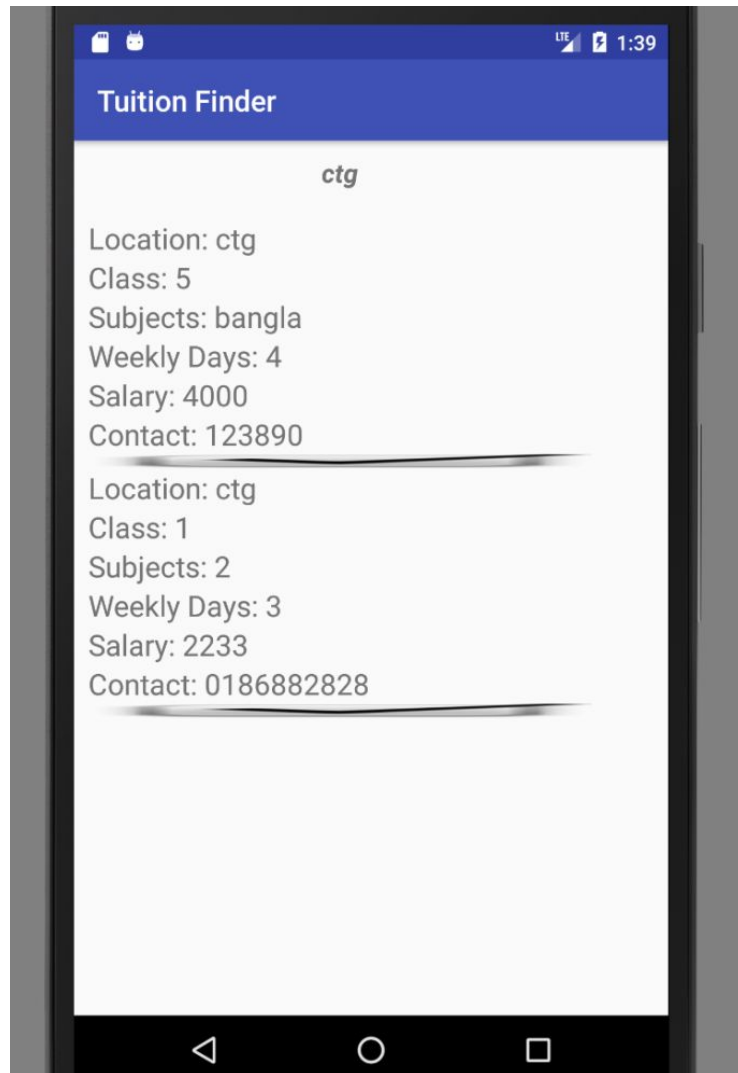


Fig:10- for tuition list

If user press on specific tuition for a few second, system shows update and delete window.

4.3.6 Update and Delete Window

When user pushed for some moment on specific tuition then system show this page where user can update and delete tuition information. This window contain 2 button. Button 1 for update and button 2 for delete.

This is update and delete window.

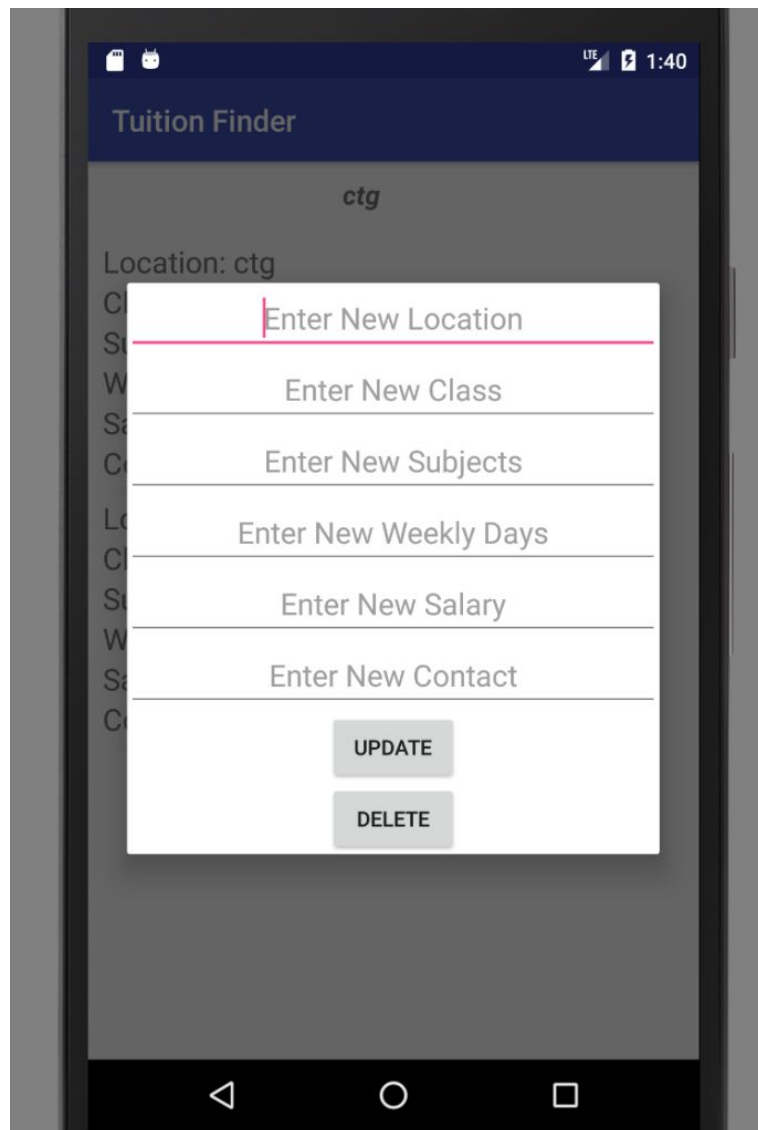


Fig:11- update and delete window.

4.3.7 Post Page

If user clicked on post button in main page then system shows post page, where user can post for tuition. There are several text field in post page and two button. Text field contain description of the tuition such as location, class, salary, days, subject and contact. Button one for submit and another for logout. If user click on submit button then system store data into backend database from text field.

This is post page.

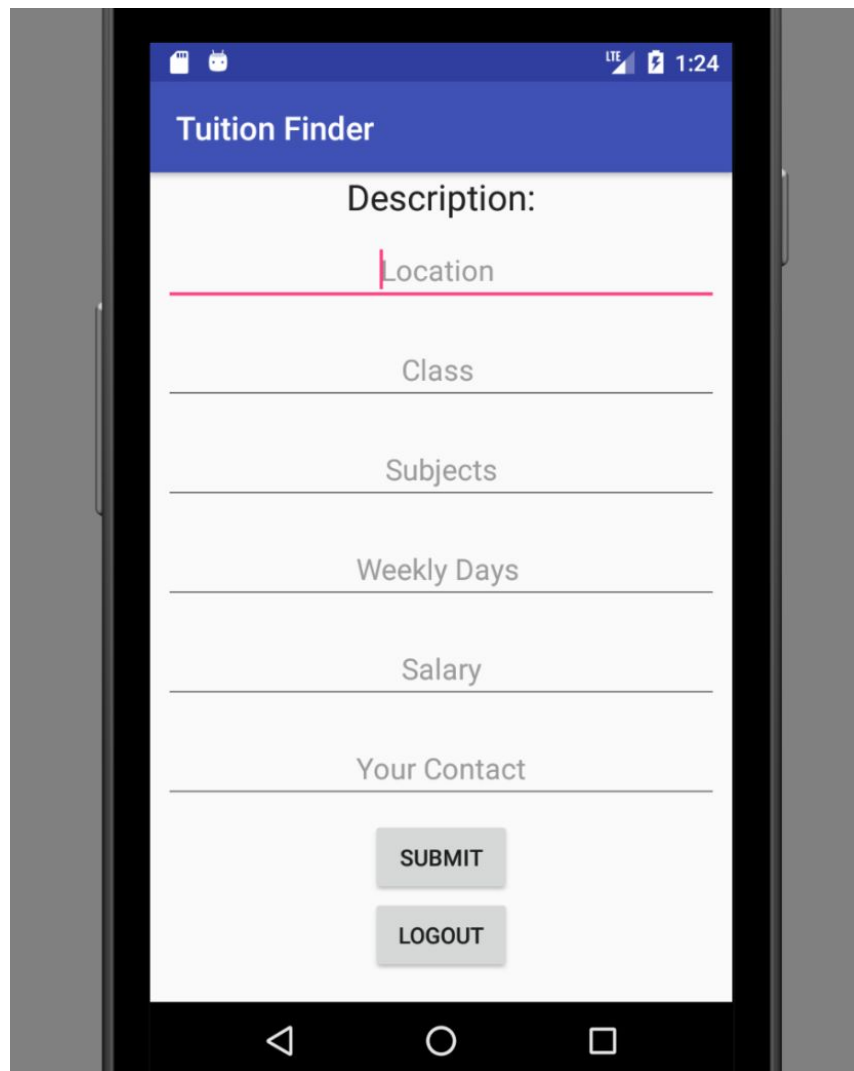


Fig:12- Post page

4.4 Testing

Testing is the process of evaluation of a system to detect differences between given input and expected output and also to assess the feature of the system. Testing assesses the quality of the product. It is a process that is done during the development process.

Strategies Used

Tests can be conducted based on two approaches –

- Functionality testing
- Implementation testing

The testing method used here is Black Box Testing. It is carried out to test functionality of the program. It is also called ‘Behavioral’ testing. The tester in this case, has a set of input values and respective desired results. On providing input, if the output matches with the desired results, the program is tested ‘ok’, and problematic otherwise.

Test Case No.	Test Case Description	Test Data	Test Result	Status
1	Launch Application	-	Main page displayed	Success
2	Check response when invalid email and password is entered	Email: diveshsharma36@gmail.com Password: divesh789	Invalid Login message displayed	Success
3	Check user login with valid data	Email: diveshsharma36@gmail.com Password: divesh789@	Successfully logged in	Success
4	Search tuton by location	Put “airport road” in search box	A tuton listed in airport road displayed	Success
5	Post tuton- Enter details and click submit	Location=airport road Subjects= All Wekly days= Mo-Fri Salary=INR 3000 Contact=9993303520	Tution details uploaded successfully	Success

6	Update tuton by pressing on tuton name for two seconds, enter details and press update	New location=sudama nagar	Tution details updated successfully, checked by searching with “sudama nagar”	Success
7	Delete tuton by pressing Delete button in the update and delete window	-	Tution deleted successfully	Success
8	Logout by pressing the logot button	-	Logged out successfully, directed to main page of application	Success

Table No-2- Test Cases

Chapter 5. Conclusion

Conclusion

In this tutor finding android application students and the faculty and students can register. Based on the latitude and longitudinal mapping the tutor location is mapped to the student. The search of the subject expert is also filtered based on the location so that it will be easy for the finders to select the appropriate tutor. Based on the students selection the tutors will be made available for their special classes and then they will be intimated through notifications. This system provides an efficient search and gives effective result for the users.

In this tutor finding android application students and the faculty and students can register. Based on the latitude and longitudinal mapping the tutor location is mapped to the student. The search of the subject expert is also filtered based on the location so that it will be easy for the finders to select the appropriate tutor. Based on the students selection the tutors will be made available for their special classes and then they will be intimated through notifications. This system provides an efficient search and gives effective result for the users. Future scope of the project is that in future the search can be completed using Algorithms of machine learning to make the system smart by giving results taking into account the reviews, ratings of the tutor along with location. We can also create an account based approach where each student can have their account linked with their respective tutors so that they can schedule classes and can connect on the same platform.

5.1 Limitations of the Work

- The application is not built for iOS devices.
- The application has no facility for recommending tutors.

5.2 Suggestion and Recommendations for Future Work

This system can also be developed for IOS Platform in the near future.

Chapter 6. Bibliography

Bibliography

1. Dolly Panchal, Mili Sanghavi, Shikha Devi Pandey, Elizabeth George.”An android application for finding tutors using data mining techniques” International Journal of Scientific & Engineering Research, Vol 7, Iss 4, April 2016.
2. Calculate distance, bearing, and more between latitude/longitude points. Chris Veness 2002- 2012[Online] <http://www.movable-type.co.uk/scripts/Latlong.html>.
3. Pankti Doshi, Pooja Jain, Abhishek Shakwala.” Location Based Services and Integration of Google Maps in Android” International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 3 Issue 3 March,2014 Page No. 5072-5077.
4. Manav Singhal, Anupam Shukla, “Implementation of Location based Services in Android using GPS and Web Services ”, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, January 2012.
5. Android Developers <http://developer.android.com/reference/android/>
6. Google Directions API <https://developers.google.com/maps/documentation/directions/>
7. Google Places API <https://developers.google.com/places/documentation/>
8. <https://firebase.google.com/>
9. Alejandro J. and Nicu S., "Multimodal Human computer interaction: A survey", IEEE International Workshop on Human Computer Interaction, China, 2005.
10. Fatima Al-Dhaen and Hussein Karam, "Robust Virtual Reality Environment for Improving Disability Learning Skills", International Conference on Higher Education in the 21 Century, Bahrain, 2007.
11. Guo Shaoqing, "Task-driven approach to the content", E-education research, vol. 234, Jul. 2006, pp. 57-59.

Appendix A

Source Code

```

MainActivity

private void registerUser() {
    String email = editTextEmail.getText().toString().trim();
    String password = editTextPassword.getText().toString().trim();
    if (TextUtils.isEmpty(email)) {
        Toast.makeText(context: this, text: "Please enter email", Toast.LENGTH_LONG).show();
        return;
    }
    if (TextUtils.isEmpty(password)) {
        Toast.makeText(context: this, text: "Please enter password", Toast.LENGTH_LONG).show();
        return;
    }
    progressDialog.setMessage("Registering Please Wait...");
    progressDialog.show();
    firebaseAuth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(activity: this, (task) -> {
            if (task.isSuccessful()) {
                startActivity(new Intent(getApplicationContext(), Comm_inter.class));
            } else {
                Toast.makeText(context: MainActivity.this, text: "Registration Error", Toast.LENGTH_LONG).show();
            }
            progressDialog.dismiss();
        });
}

@Override
public void onClick(View view) {
    if (view == buttonSignup) {
        registerUser();
    }
}

```

Sample code for Signup Page

```

Comm_inter onCreate() new OnClickListener onClick()

Button ref_to_tutor, ref_to_prov, ref_to_logout;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_comm_inter);
    firebaseAuth = FirebaseAuth.getInstance();
    ref_to_prov= (Button) findViewById(R.id.ref_to_prov);
    ref_to_tutor= (Button) findViewById(R.id.ref_to_tutor);
    ref_to_logout= (Button) findViewById(R.id.ref_to_logout);

    ref_to_prov.setOnClickListener((view) → {

        Intent intent= new Intent(getApplicationContext(),AdvertingActivity.class );
        startActivity(intent);
    });
    ref_to_tutor.setOnClickListener((view) → {

        Intent intent= new Intent(getApplicationContext(),Searching.class );
        startActivity(intent);
    });
    ref_to_logout.setOnClickListener((view) → {
        firebaseAuth.signOut();
        //closing activity

        Intent intent= new Intent(getApplicationContext(),MainActivity.class );
        startActivity(intent);
    });
}
}

```

Sample code for Login Page

```

tuition_lists onCreate()

String mail;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tuition_lists);
    databaseReference = FirebaseDatabase.getInstance().getReference("tuitionposts");

    firebaseAuth = FirebaseAuth.getInstance();
    mail = firebaseAuth.getCurrentUser().getEmail().toString();

    Bundle bundle = getIntent().getExtras();
    gotLoc1= bundle.getString(key:"key");
    tvSearch= (TextView) findViewById(R.id.tvSloc);
    tvSearch.setText(gotLoc1);
    listViewposts= (ListView) findViewById(R.id.listViewposts);
    postlists= new ArrayList<>();
    listViewposts.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public boolean onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
            TuitionInformation tuitionInformation = postlists.get(i);
            showUpdateDeleteDialog(tuitionInformation.getTidl(), tuitionInformation.getMail1());
            return true;
        }
    });
}
}

```

Sample code for common interface page

```

LoginActivity userLogin()
private void userLogin() {
    String email = editTextEmail.getText().toString().trim();
    String password = editTextPassword.getText().toString().trim();
    if (TextUtils.isEmpty(email)) {
        Toast.makeText(context, "Please enter email", Toast.LENGTH_LONG).show();
        return;
    }
    if (TextUtils.isEmpty(password)) {
        Toast.makeText(context, "Please enter password", Toast.LENGTH_LONG).show();
        return;
    }
    progressDialog.setMessage("Logging In Please Wait...");
    progressDialog.show();
    firebaseAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(activity, this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                progressDialog.dismiss();
                if (task.isSuccessful()) {
                    startActivity(new Intent(getApplicationContext(), Comm_inter.class));
                }
                else {
                    Toast.makeText(context, "Incorrect Username or Password",
                        Toast.LENGTH_LONG).show();
                }
            }
        });
}
}

```

Sample code for search page

```

PostLists
public class PostLists extends ArrayAdapter<TuitionInformation> {
    private Activity context;
    private List<TuitionInformation> postlist;
    public PostLists(Activity context, List<TuitionInformation> postlist) {
        super(context, R.layout.list_layout, postlist);
        this.context = context;
        this.postlist = postlist;
    }

    @NonNull
    @Override
    public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {
        LayoutInflater inflater = context.getLayoutInflater();
        View listViewItem = inflater.inflate(R.layout.list_layout, root: null, attachToRoot: true);
        TextView textLoc = (TextView) listViewItem.findViewById(R.id.tvloc);
        TextView textclass = (TextView) listViewItem.findViewById(R.id.tvclass);
        TextView textsub = (TextView) listViewItem.findViewById(R.id.tvsubjects);
        TextView textWdays = (TextView) listViewItem.findViewById(R.id.tvweekdays);
        TextView textsalary = (TextView) listViewItem.findViewById(R.id.tvsalary);
        TextView textcontact = (TextView) listViewItem.findViewById(R.id.tvcontact);
        //TextView textmail = (TextView) listViewItem.findViewById(R.id.tvmail);
        TuitionInformation tuitionInformation = postlist.get(position);
        textLoc.setText("Location: " + tuitionInformation.getLocation1());
        textclass.setText("Class: " + tuitionInformation.getClass1());
        textsub.setText("Subjects: " + tuitionInformation.getSubject1());
        textWdays.setText("Weekly Days: " + tuitionInformation.getWeek_days1());
        textsalary.setText("Salary: " + tuitionInformation.getSalary1());
        textcontact.setText("Contact: " + tuitionInformation.getContact1());
        //textmail.setText("Mail: " + tuitionInformation.getMail1());
        return listViewItem;
    }
}

```

Sample code for tuton list page

```

public class Searching extends AppCompatActivity {
    EditText LocSearch;
    String locSearch1;
    Button ref_to_tuulist1;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_searching);
        LocSearch= (EditText) findViewById(R.id.locSearch);

        ref_to_tuulist1= (Button) findViewById(R.id.ref_to_tuulist);

        ref_to_tuulist1.setOnClickListener((view) → {
            lets_convert();
            Intent intent= new Intent( packageContext:Searching.this,tuition_lists.class );
            intent.putExtra( name:"key", locSearch1);
            startActivity(intent);
        });
    }
    public void lets_convert(){
        locSearch1= LocSearch.getText().toString();
    }
}

```

Sample code for update and delete page

Appendix-B

Interaction Sheet

Week	Date	Remarks	Guide's Sign
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			

Appendix-C

Research Paper