A
PROJECT REPORT
ON
AURO TALK

PROJECT REPORT SUBMITTED IN THE PARTIAL FULFILLMENT FOR THE DEGREE OF

BACHELOR OF SCIENCE
IN

INFORMATION TECHNOLOGY

SUBMITTED

BY

KOMAL MUNDHRA

DRISHTI SHARAF

DHWANI PATEL

2018

UNDER THE GUIDANCE

OF

Mr. Rajiv Katare

AURO UNIVERSITY

SURAT

Academic Year –2020-21

AURO UNIVERSITY
A center for integral and transformational learning

earthspace, Hazira Road earthspace
Hazira Road, Opp ONGC
Surat - 394510, Gujarat, India.
www.aurouniversity.edu.in

CERTIFICATE OF UNDERTAKING

We, **KOMAL MUNDHRA, DHWANI PATEL, DRISHTI SHARAF**, Enrollment No**. 212018005, 222018018, 222018007** hereby declare that project entitled AURO TALK undertaken at School of Information Technology for the partial fulfilment of BSc. IT degree. This Project is my original work and the Project has nor formed the basis for the award of any other degree, associate ship, fellowship or any other similar titles, either in AURO University or any other University of India.

KOMAL MUNDHRA (212018005)

DHWANI PATEL (222018018)

DRISHTI SHARAF (222018007)

AURO TALK

ACKNOWLEDGEMENT

We would like to express my deepest appreciation to all those who provided me the possibility to complete this report. A special gratitude we give to our faculty Mr. Rajiv Katare, whose contribution in stimulating suggestions and encouragement, helped me to coordinate my project especially in writing this report.

CONTENT

FIGURES PAGE NUMBERS

INTRODUCTION OF THE PROJECT “AURO TALK”

Communication is a means for people to exchange messages. The emergence of computer network and telecommunication technologies bears the same objective that is to allow people to communicate. All this while, much efforts has been drawn towards consolidating the device into one and therefore indiscriminate the services. Chatting is a method of using technology to bring people and ideas together despite the geographical barriers. Our project is an example of a chat application. It is made up of applications the client application which runs on the users mobile and server application which runs on any pc on the network. To start chatting our user should login/register in our application. After that user can use all the exciting features of our application.

SOFTWARE/SYSTEM REQUIREMENT STUDY

Functional Requirements

- **User Registration**- User must be able to register for the application through a valid Email Id.
- **Send Message**- Users should be able to send instant messages to anyone from the User list. Users should be notified when the message is successfully delivered to the recipient by displaying a delivered word below to the message sent.
- **Message status**- Users must be able to get information on whether the message sent has been read by the intended recipient. If the recipient reads the message, Seen must appear below to the message read.
- **Search Option**- From 100 numbers of users it is difficult to find one particular user. To solve this problem, users should be able to use search options.
- **New User**- Users can see the list of users with their name which means if new users register one can see the name in the user option.
- **Active users**- Users should be able to see if another user is active online or not.
- **User profile**- User should be able to set their profile picture
- **Logout**- User should be able to logout and if user wants to login again, he/she can login with the same email and password
- **New message Notification**- User should receive notification when a new message is received.
- **Forgot your password**- If a user forgets his/her password then one can reset his/her password with the help of an email.

Non- Functional Requirements

- **Scalability**
  Auro Talk should be able to provide instant messaging services to 1 billion users at any given time.
- **Privacy**
  Messages shared between users should be encrypted to maintain privacy.
- **Performance**
  Application must be lightweight and must send messages instantly.

Technical Requirements

- Firebase
- Android studio
- Android 2.1 (Ap level 7)
- Java
- Internet Connection
- Electricity
- Microsoft Windows 7/8/10 (32-bit or 64-bit)
- 3 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)

- 1280 x 800 minimum screen resolution

Operational Requirements

Application must work on all android mobile and tablet devices. User interface must be consistent on all android devices.

OBJECTIVE

Our main objective is to make a chat application only for Auro University members because there was no such platform where students of different schools can talk. There was no connection between the different schools and faculty members. In our application students and faculty members have to log in with their respective Email Ids.

FEASIBILITY STUDY

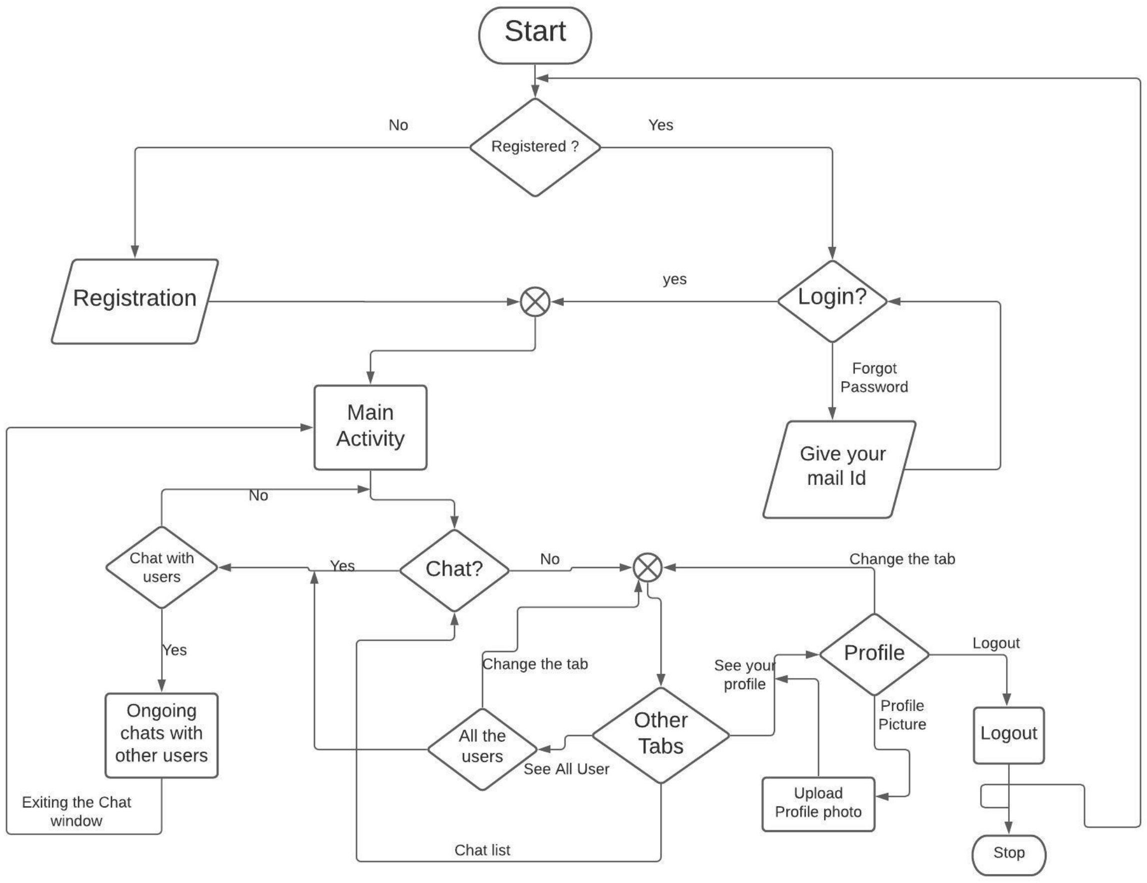
1. Technical Feasibility- The proposed system will require technical resources -minimum 8GB RAM, firebase, Android Studio (Android 2.1), Java and technical team who is capable of converting the ideas into working systems.

2. Economic Feasibility- There was no manual system. The proposed system will not be economically feasible because technical team and some technical resources are need to be paid.

3. Scheduling Feasibility- This assessment is the most important for project success; after all, a project will fail if not completed on time. The proposed system will get completed within a period of two months.

SOFTWARE/SYSTEM DESIGN

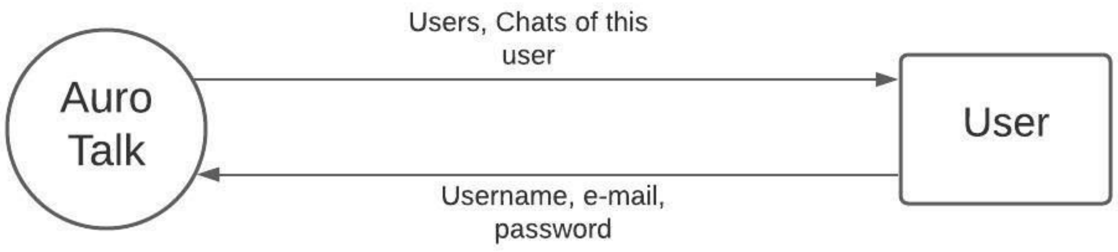
Flowchart



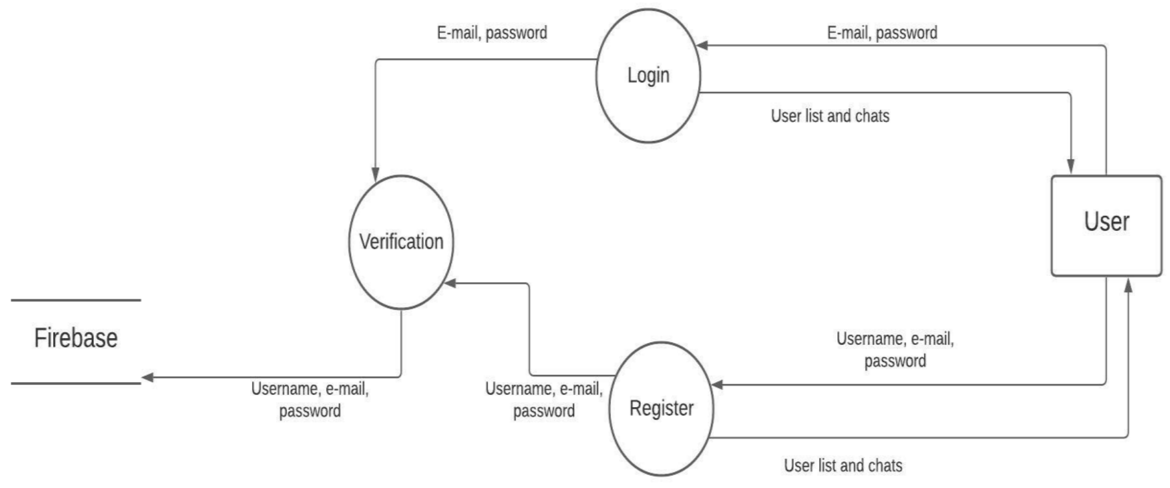
Start
Registered ?
No
Yes
Registration
yes
Login?
Forgot Password
Give your mail Id
Main Activity
No
Chat with users
Yes
Chat?
No
Change the tab
Yes
Ongoing chats with other users
Change the tab
All the users
See All User
Other Tabs
See your profile
Profile
Logout
Profile Picture
Logout
Upload Profile photo
Exiting the Chat window
Chat list
Stop

DFD

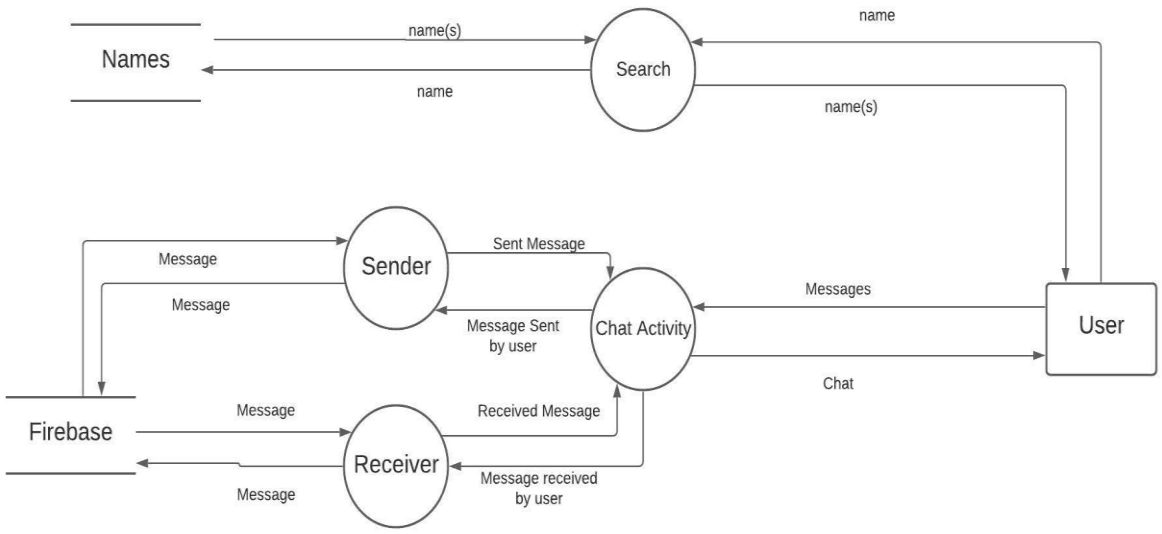
DFD level 0



DFD level 1

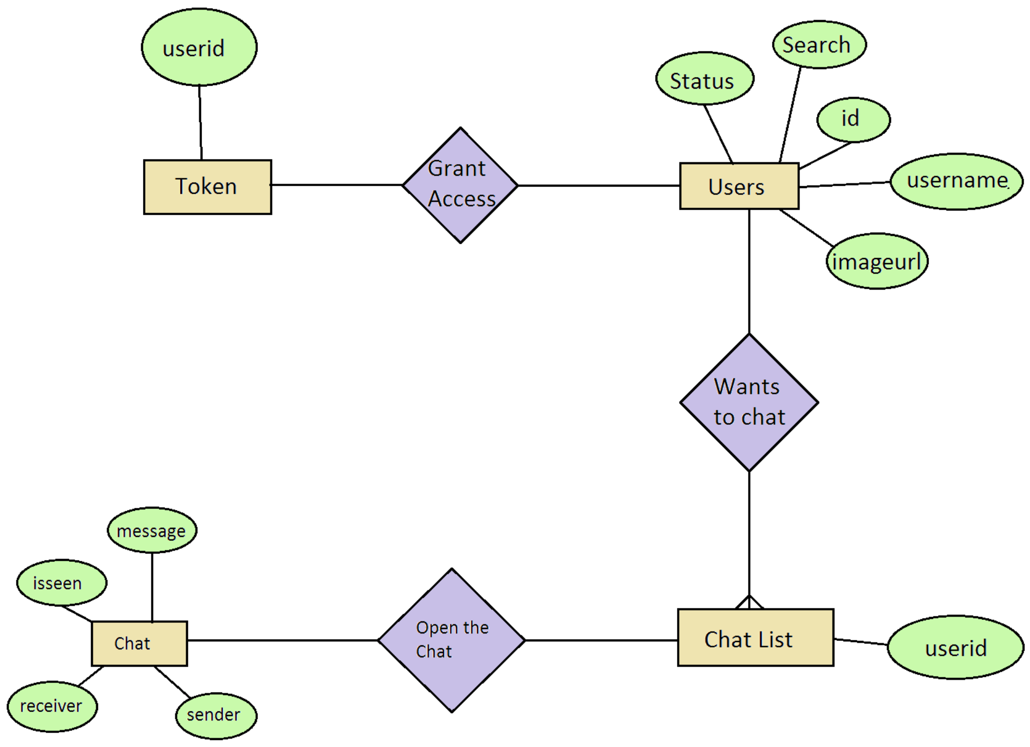

E-mail, password
Login
E-mail, password
User list and chats
Verification
User
Firebase
Username, e-mail, password
Username, e-mail, password
Username, e-mail, password
Register
User list and chats

DFD level 2



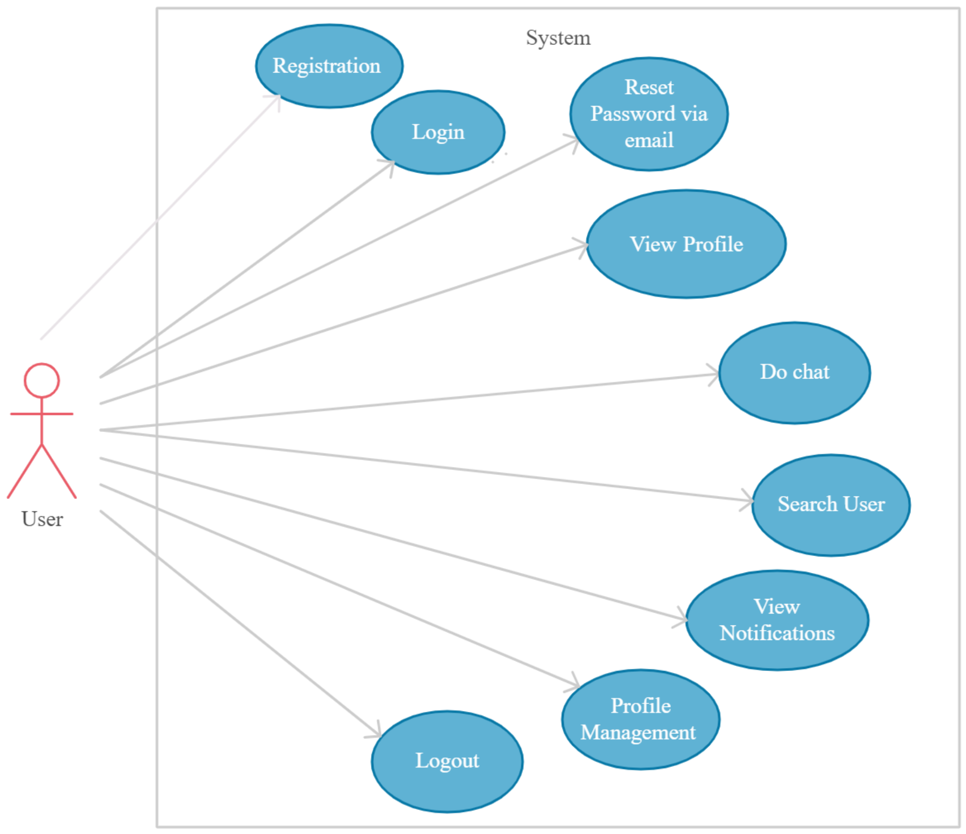
name
name(s)
Names
Search
name
name(s)
Message
Sender
Sent Message
Message
Message Sent by user
Chat Activity
Messages
User
Chat
Firebase
Message
Received Message
Receiver
Message
Message received by user

E-R diagram

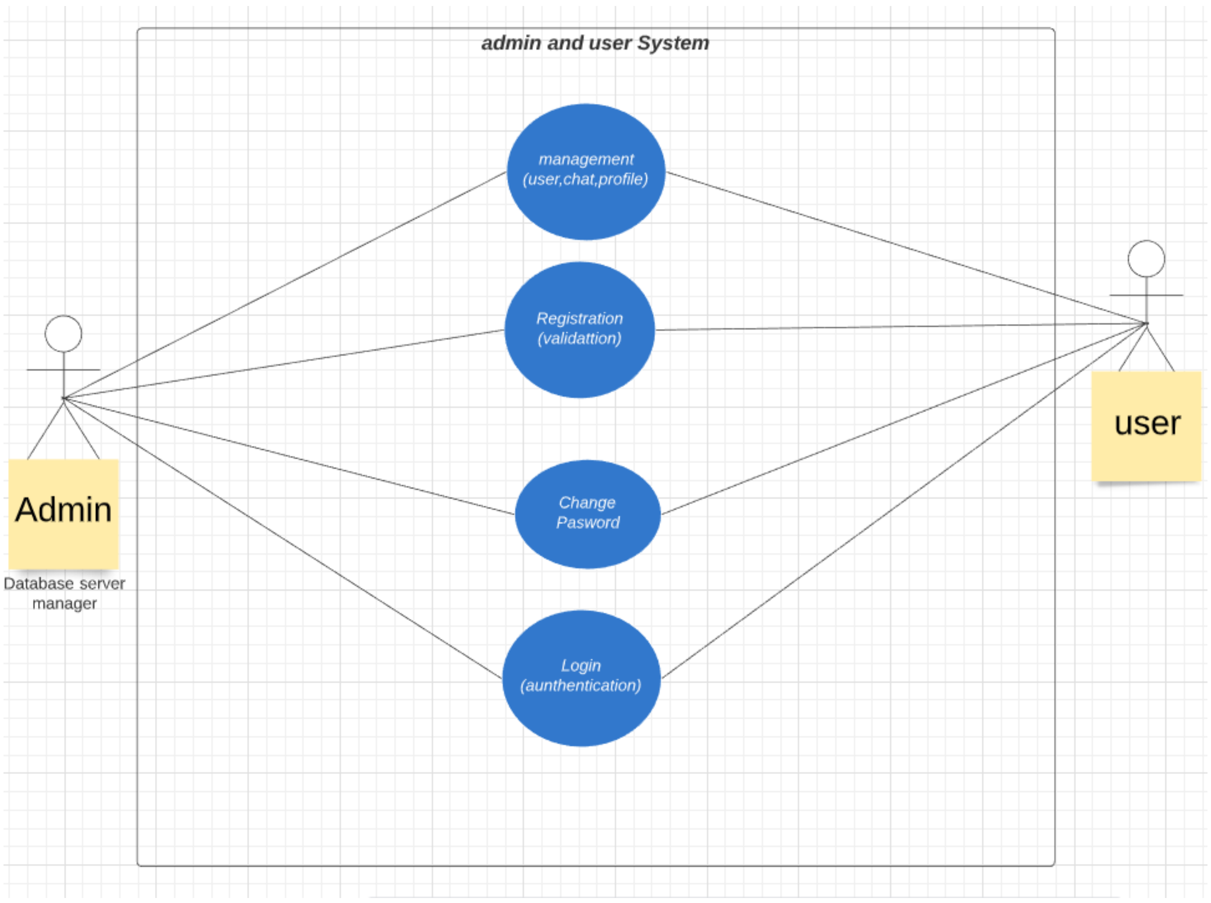

userid
Token
Grant Access
Status
Search
id
Users
username
imageurl
Wants to chat
message
isseen
Chat
receiver
sender
Open the Chat
Chat List
userid

Use Case Diagram

AuroTalk Use Cases



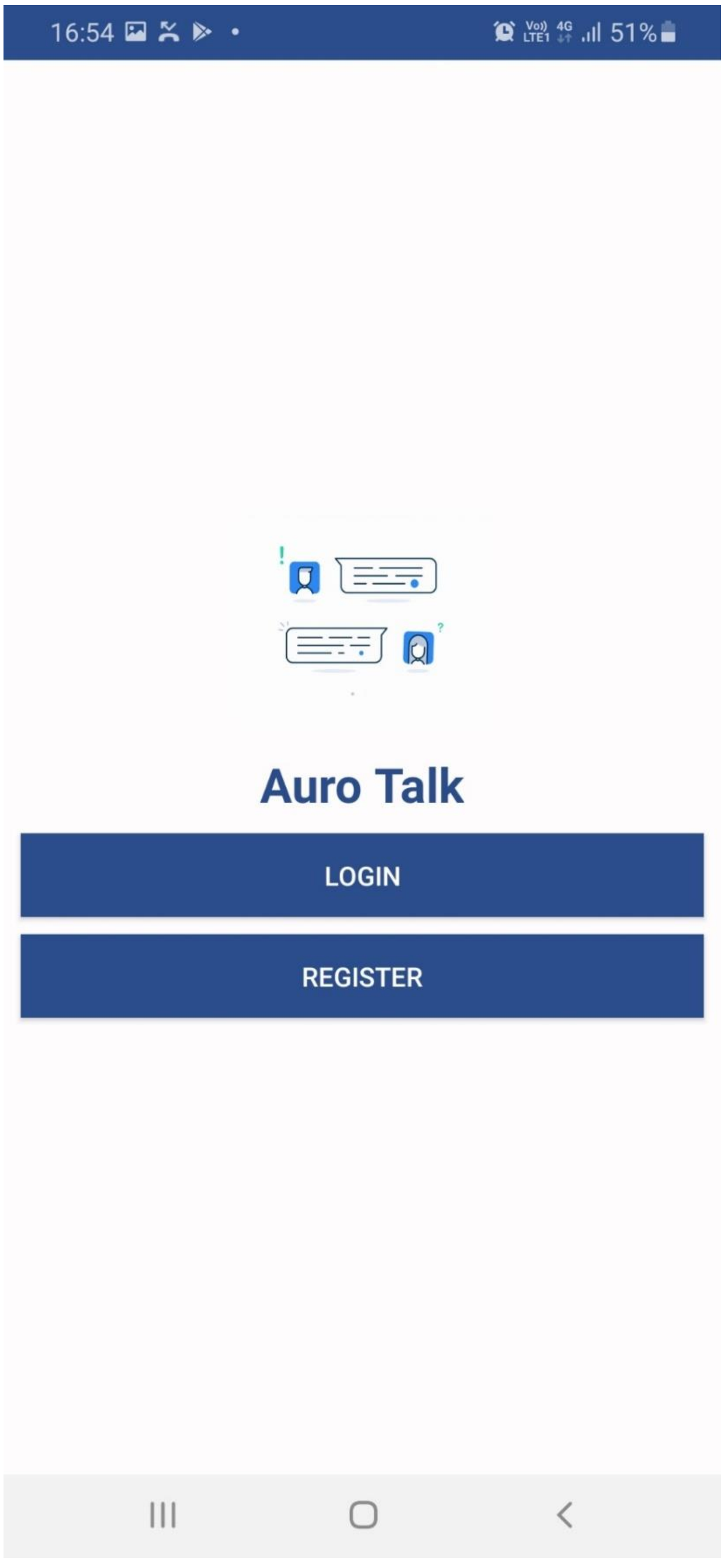
Between Admin and User



FRONT-END SCREEN SHOTS

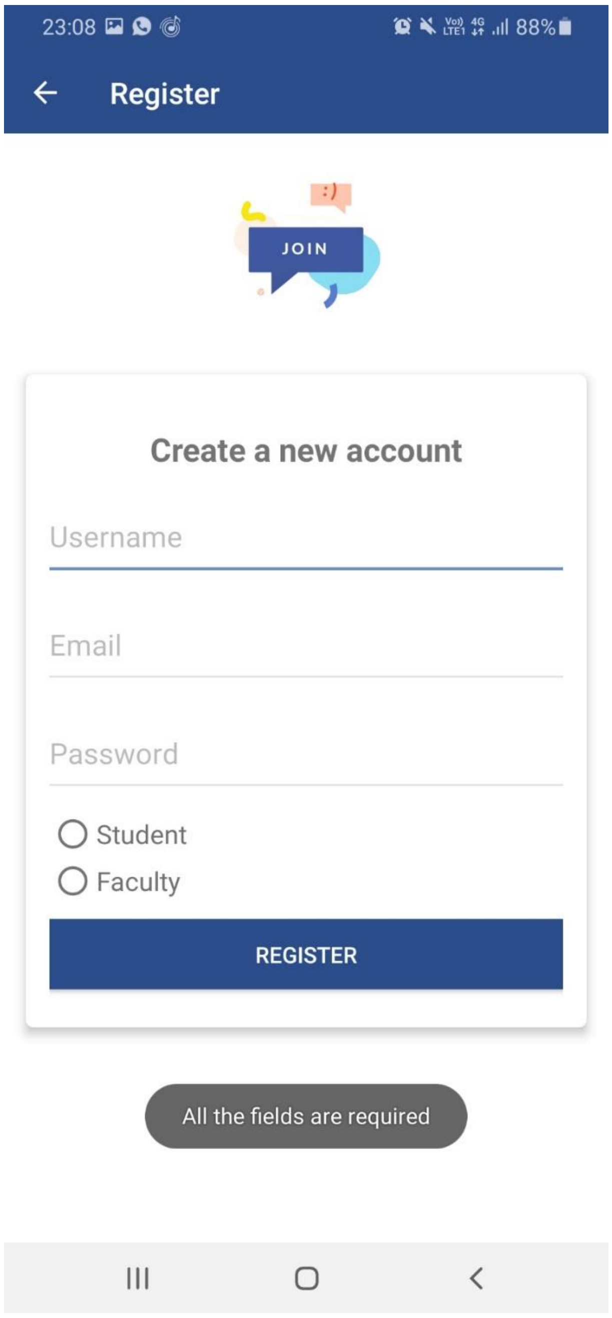
Login/Registration

First page of our application. Users' needs to login/register.



Registration

If any field is left empty then pop-up appears.


23:08 88%

Register

Create a new account

Username

Email

Password

Student

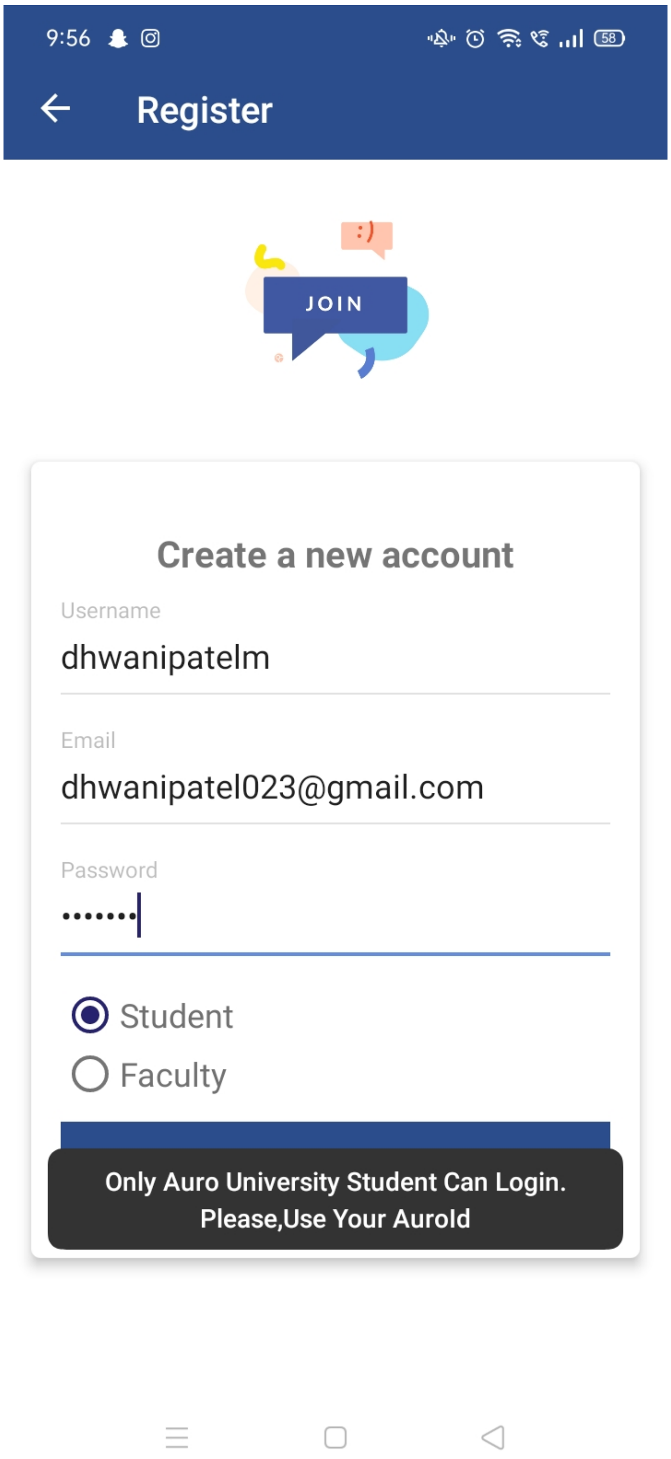
Faculty

REGISTER

All the fields are required

Email- validation

Users' need to enter valid email address (auro university email).


9:56

Register

Create a new account

Username
dhwanipatelm

Email
dhwanipatel023@gmail.com

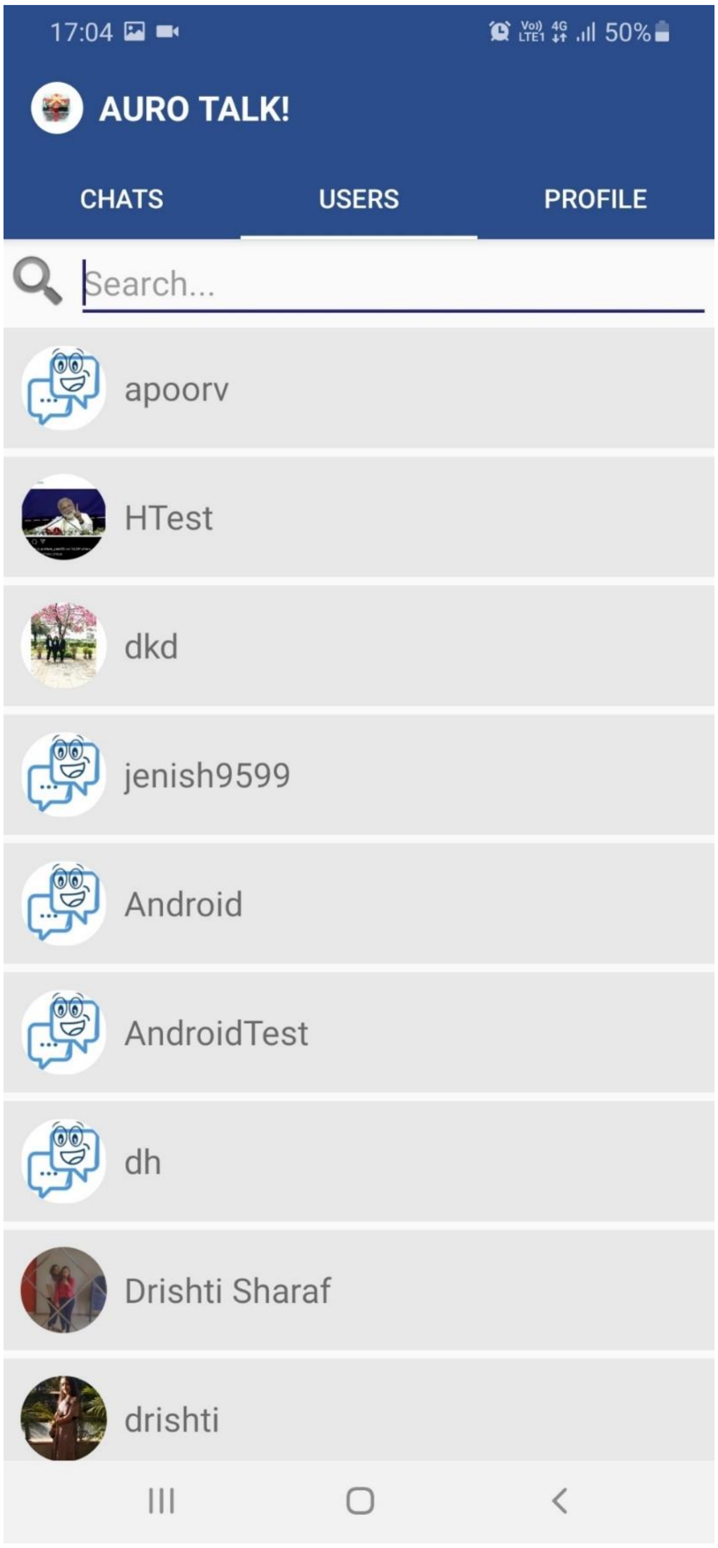
Password
•••••••

[✓ Student]
Faculty

Only Auro University Student Can Login.
Please,Use Your AuroId

Users' page

Here user can see the list of users



apoorv

HTest

dkd

jenish9599

Android

AndroidTest

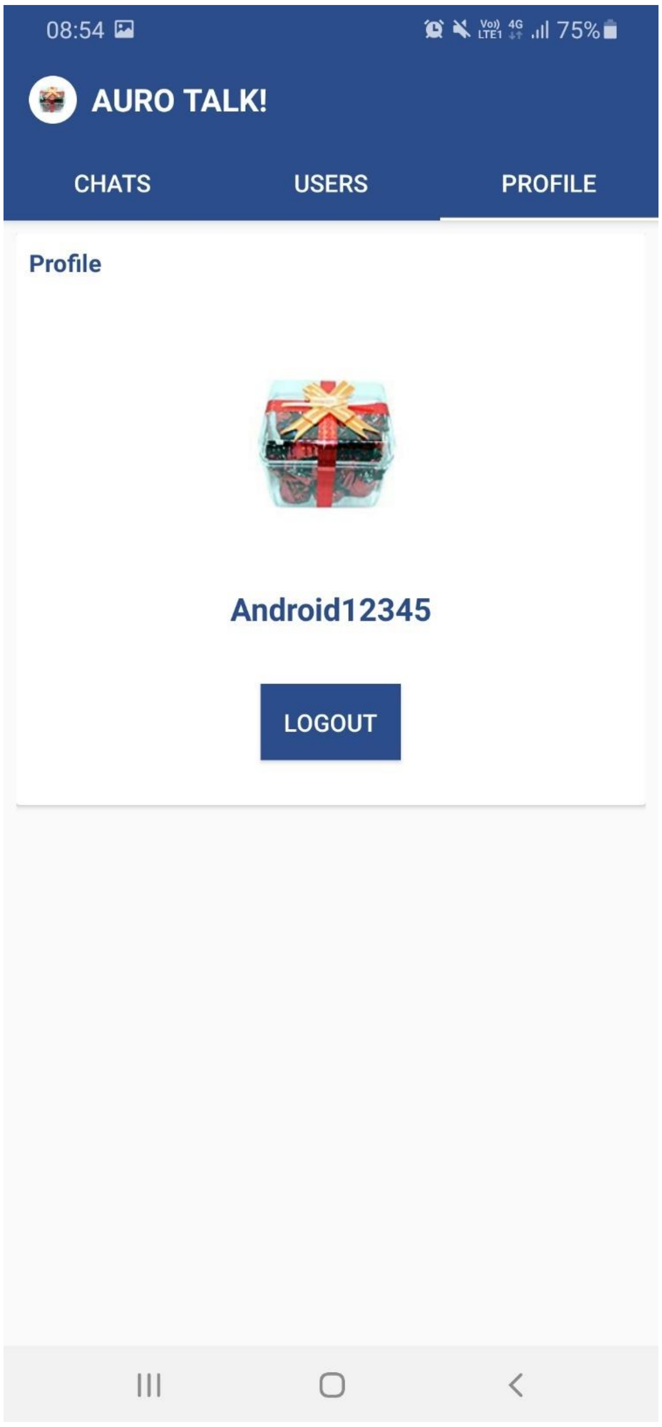
dh

Drishti Sharaf

drishti

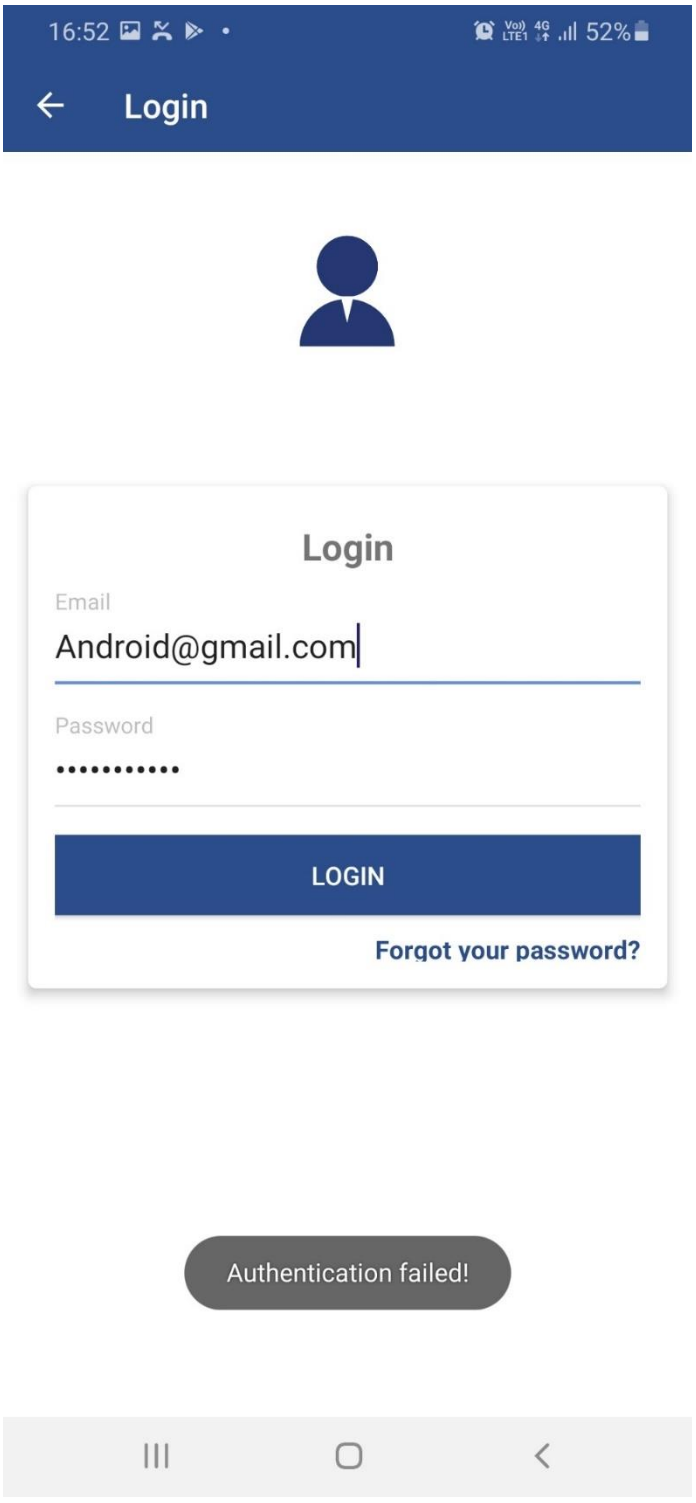
Profile Page

From here, user can logout or change his/her profile picture.



Authentication failed in Login

User have to login with their auro mail id.

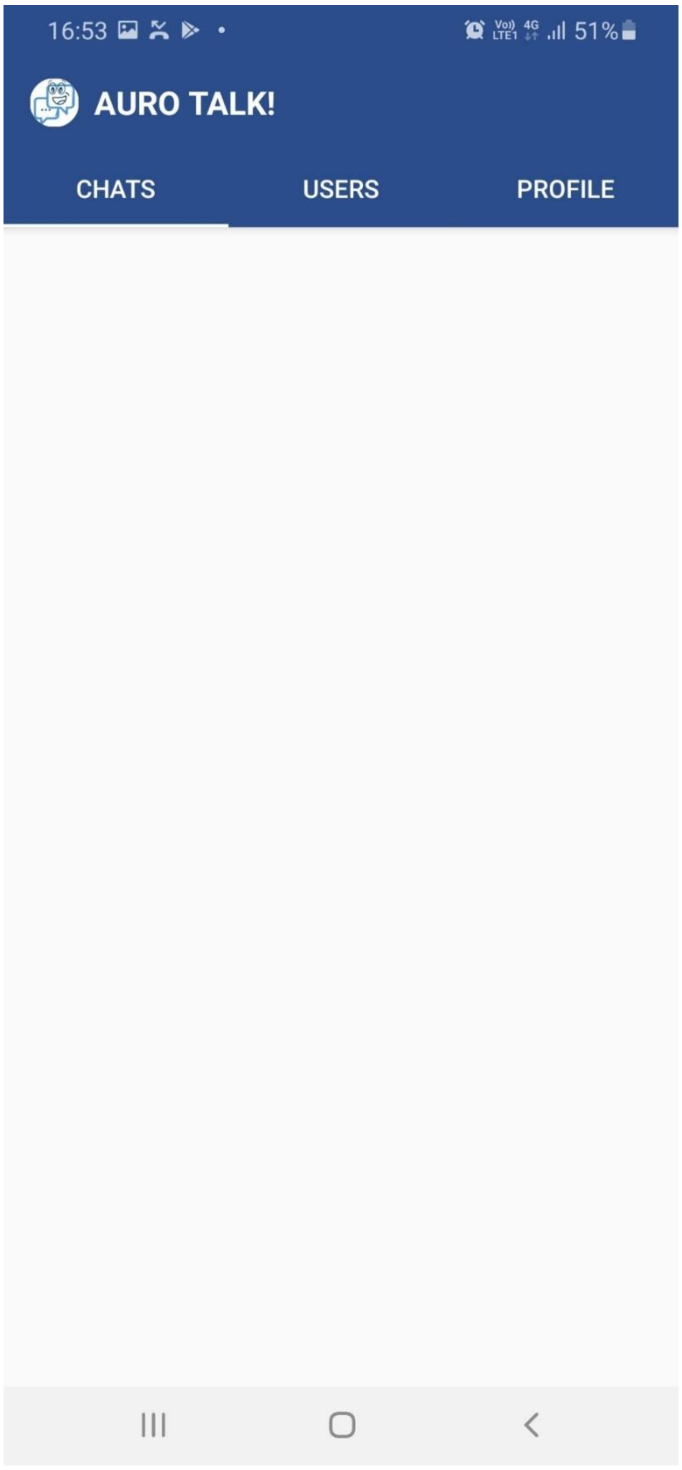


Android@gmail.com

•••••••••••

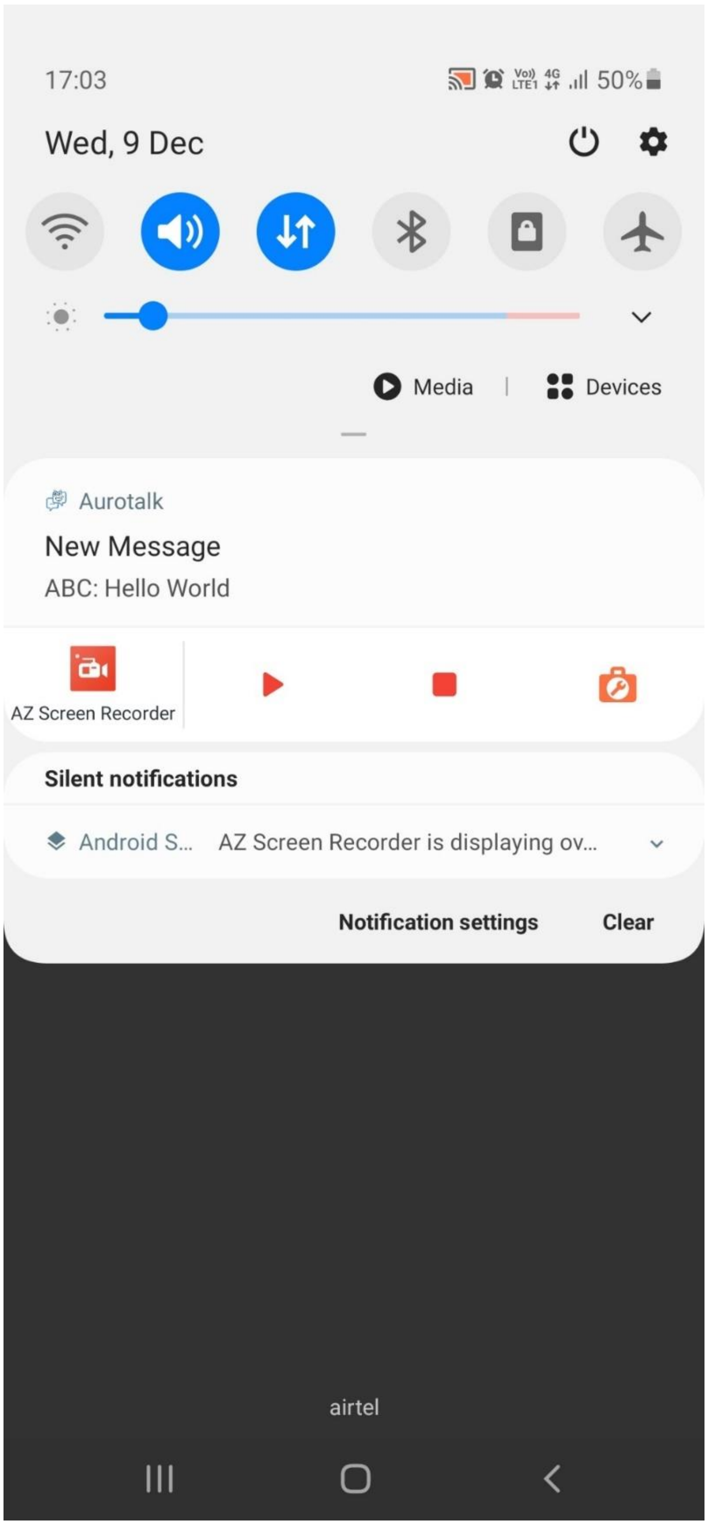
Authentication failed!

As soon as user register/login, this page opens.



New Message Notification

If user gets a new message, he/she will receive a notification.

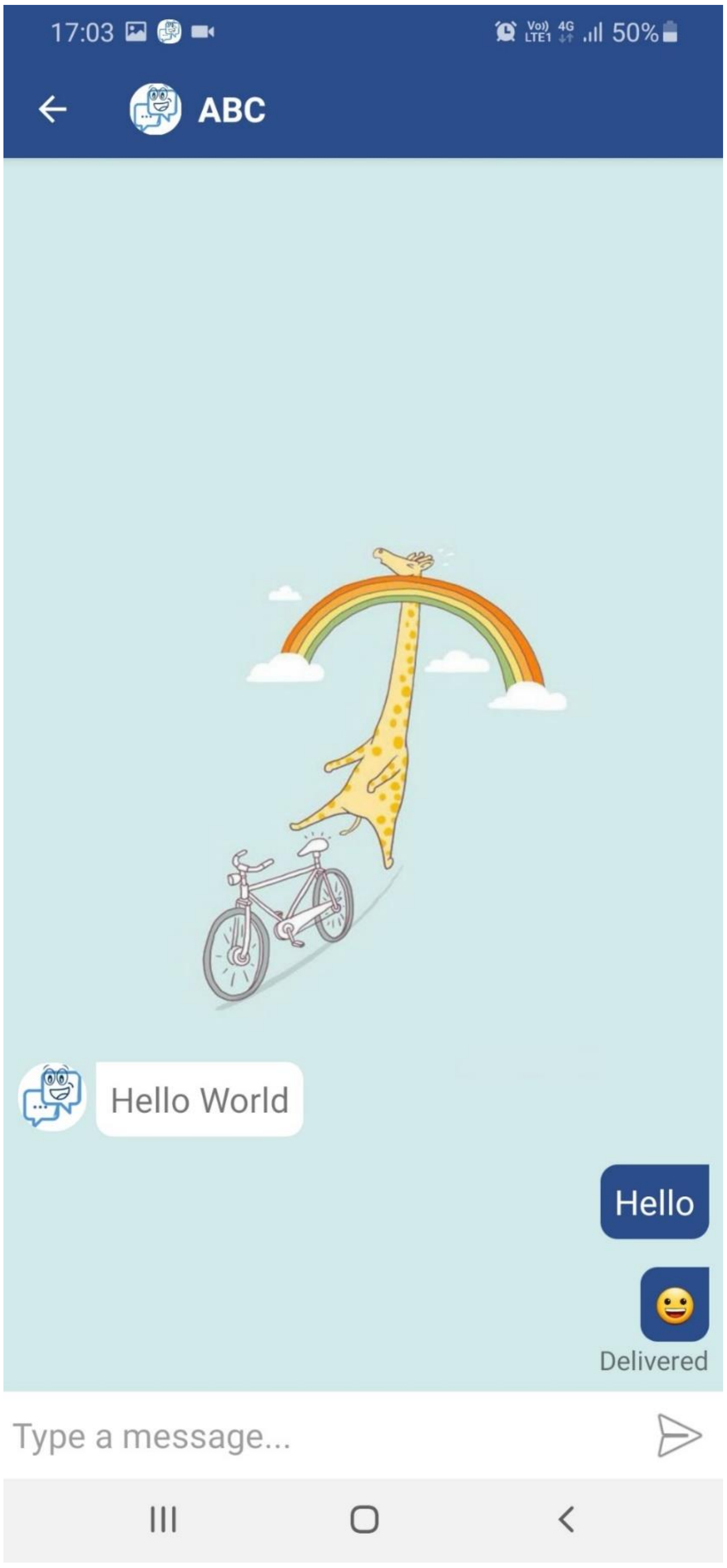


Aurotalk
New Message
ABC: Hello World

Silent notifications

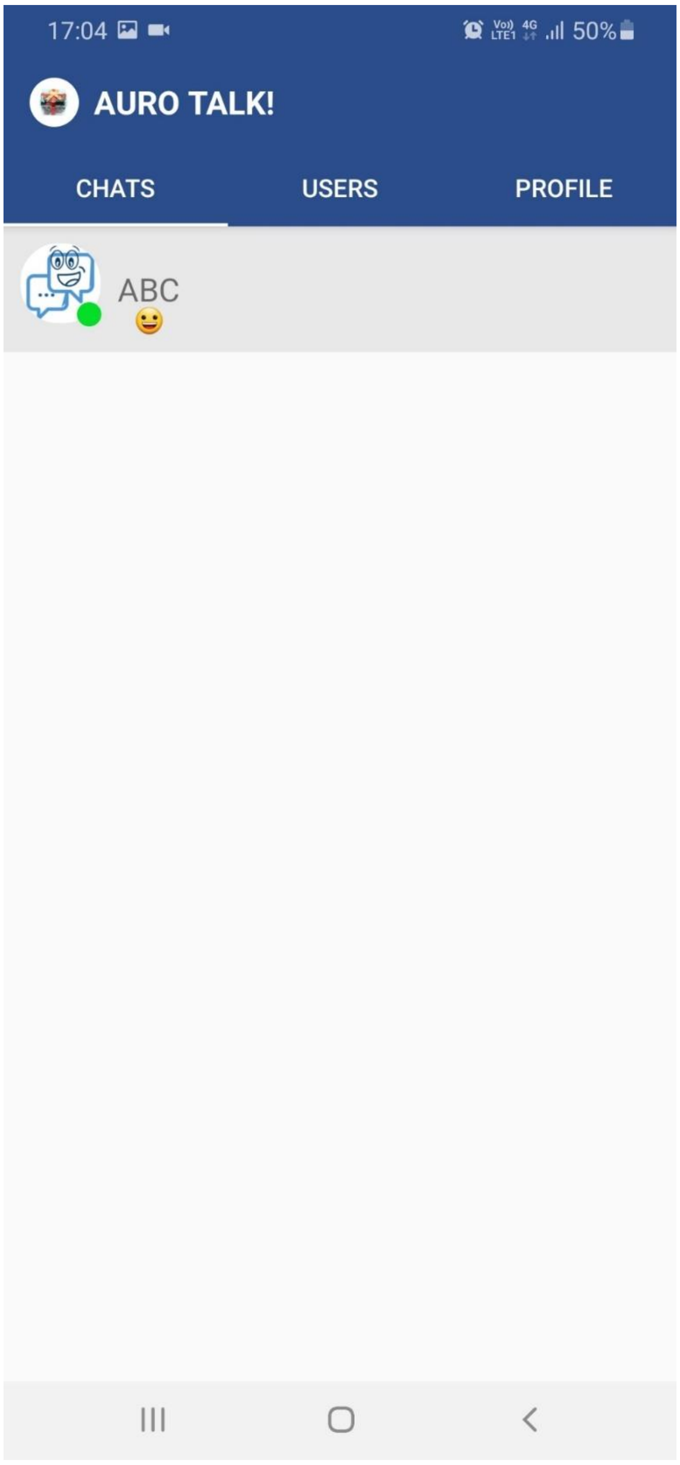
Android S... AZ Screen Recorder is displaying ov...

User can see whether or not the message is delivered.


17:03
ABC
Hello World
Hello
😀
Delivered
Type a message...

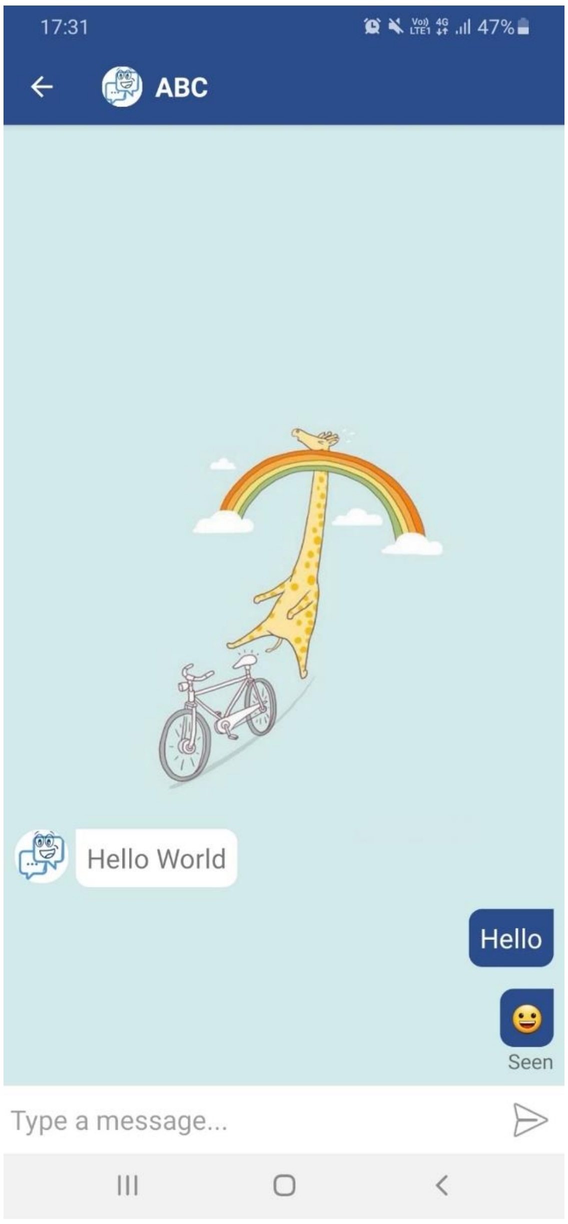
Active Online Status

User can see the online status of another user.



Message Status

User can see that if message sent is read or not.



Hello World

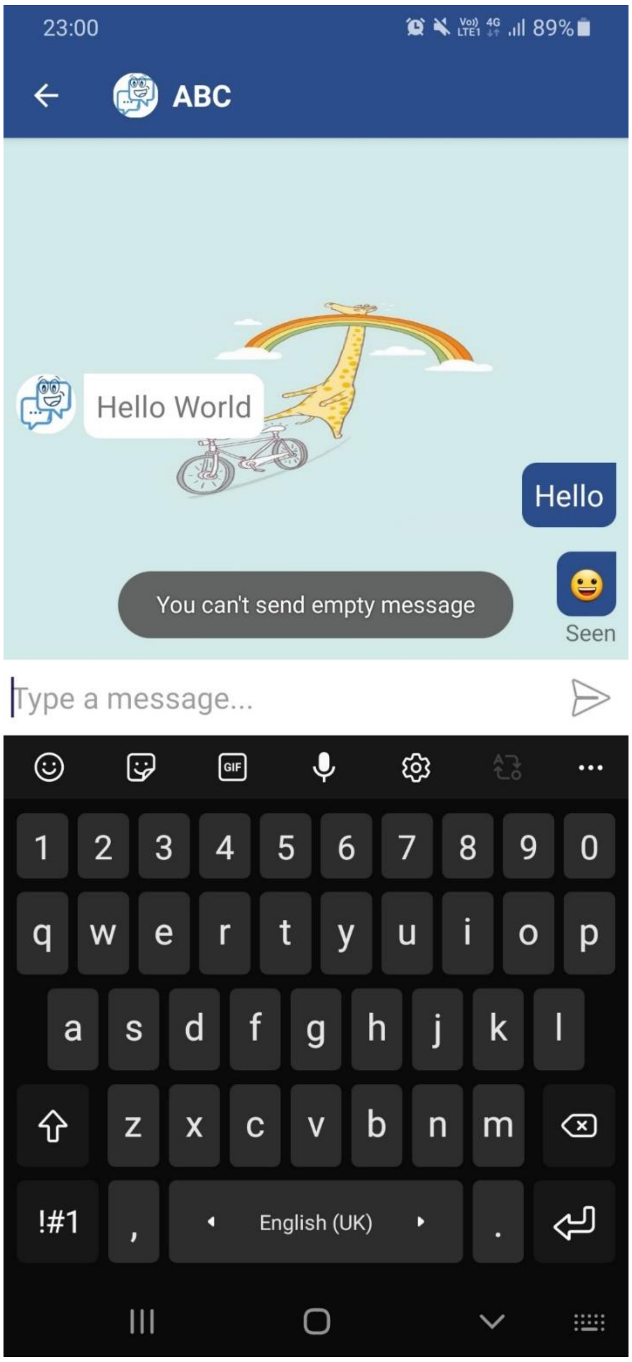
Hello

😀

Seen

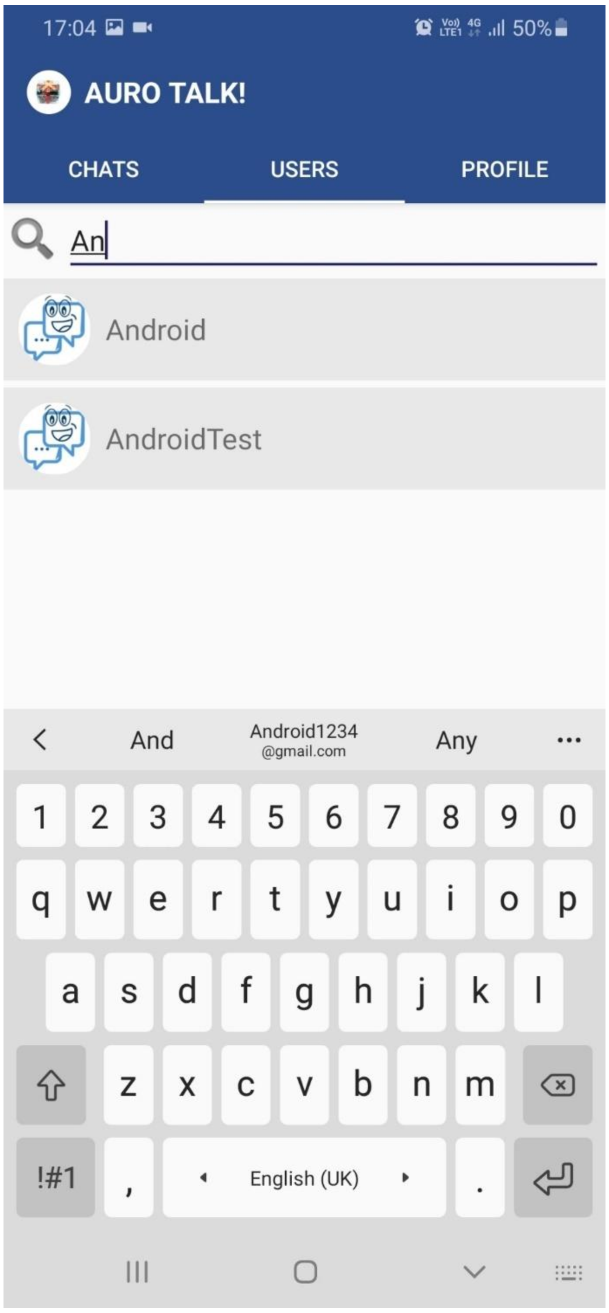
Chat Box

User can’t send empty messages.



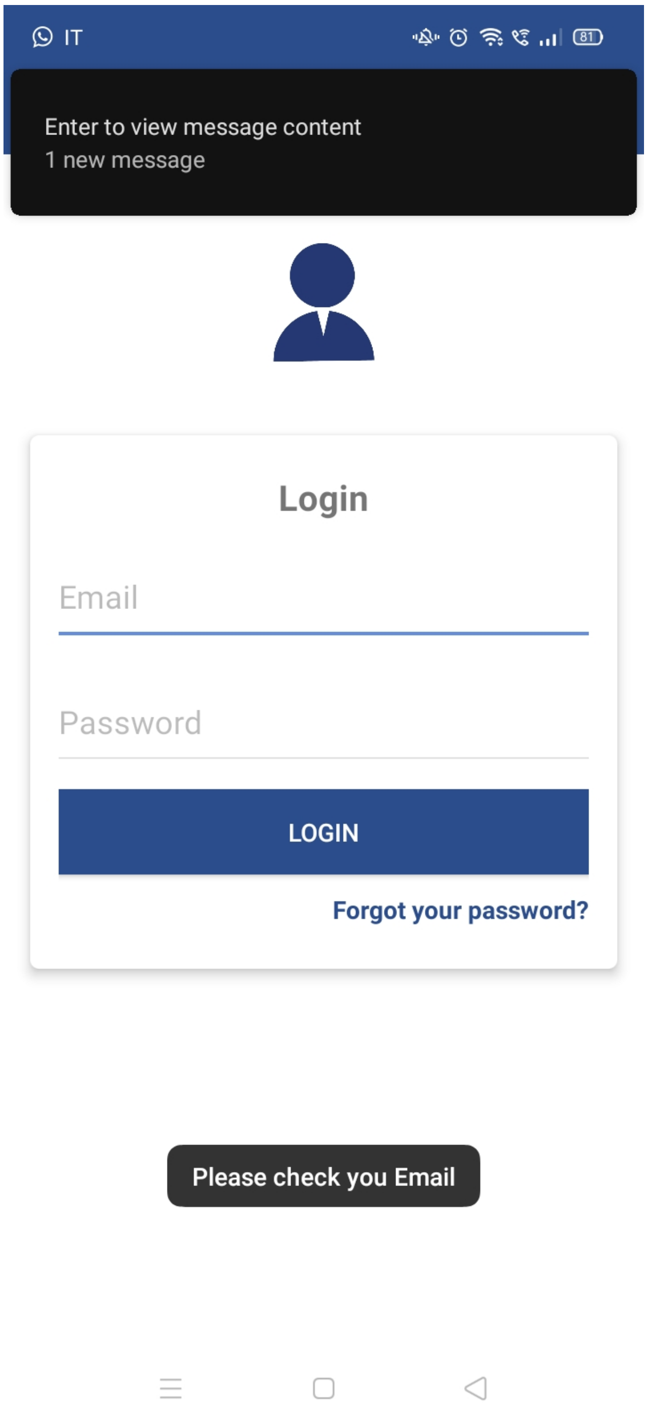
Search Option

Here we can easily search another user easily.



Check Your Email

If a user clicks reset button then pop-up appears.



Enter to view message content
1 new message

Login

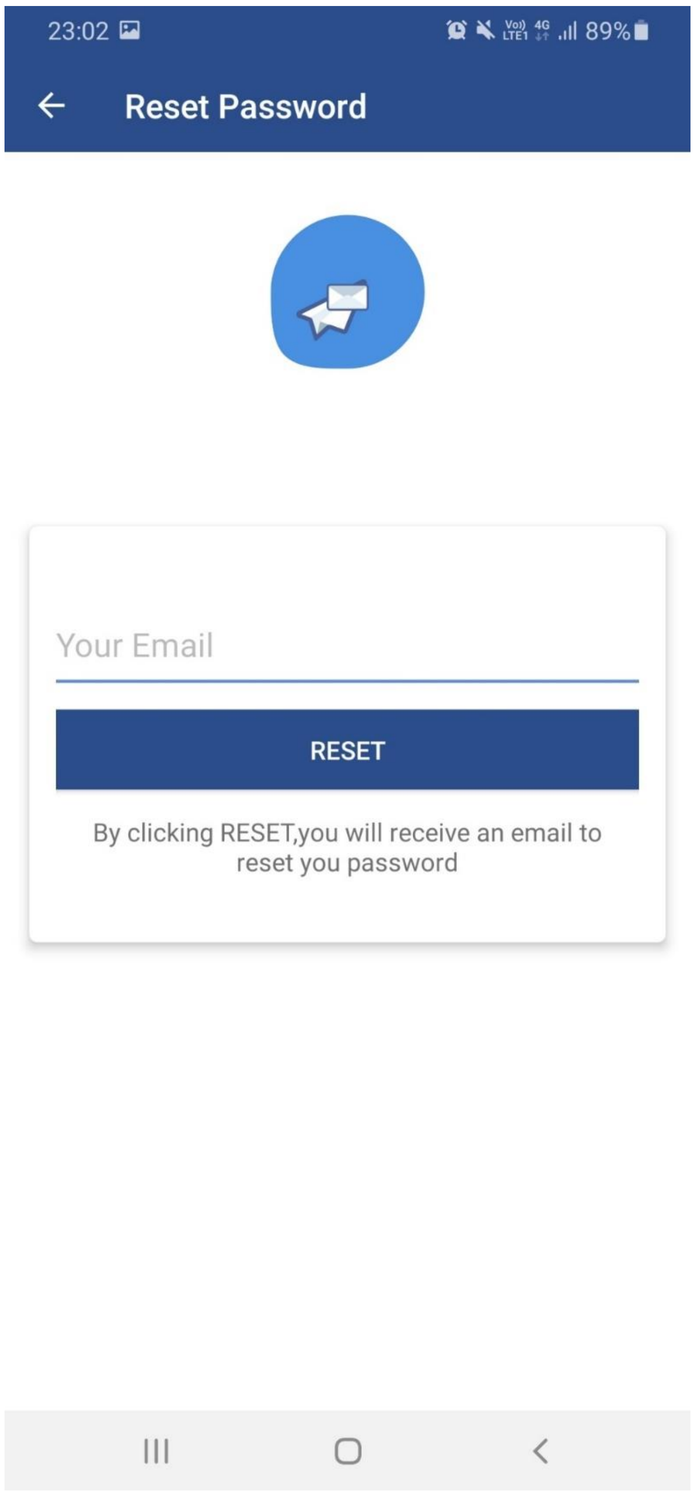
Email

Password

Please check you Email

Reset

If a user clicks on forgot your password button, this page opens.



23:02 89%

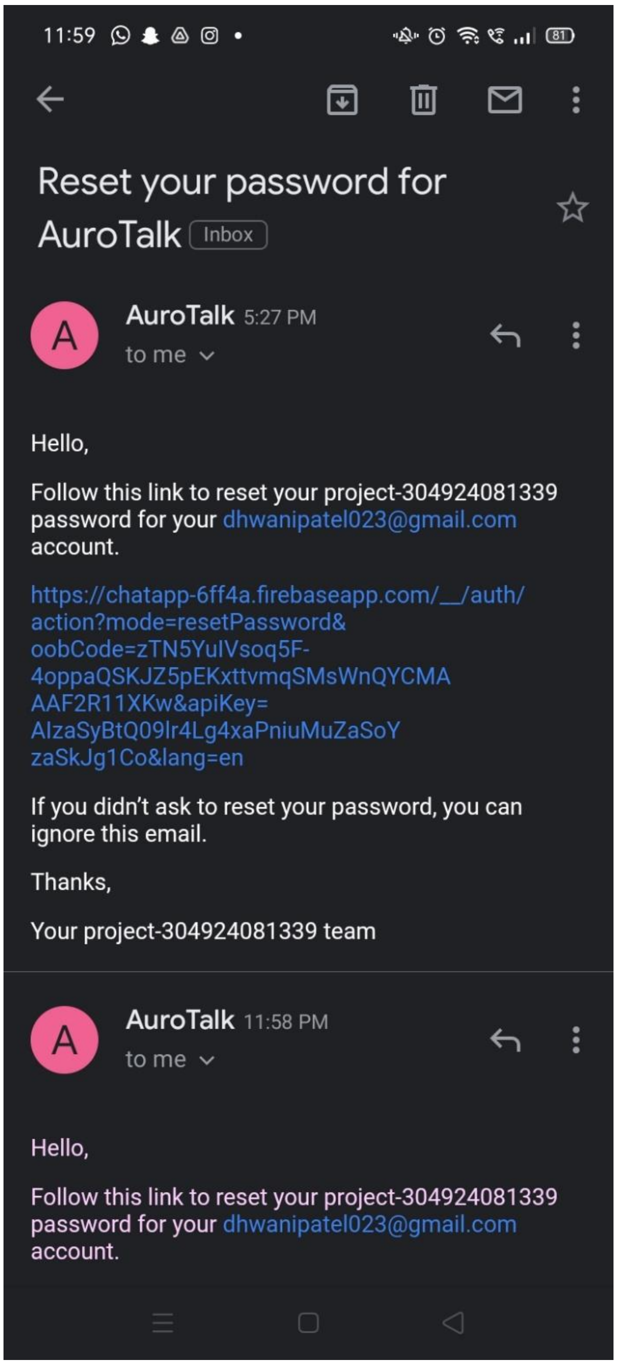
Reset Password

Your Email

RESET

By clicking RESET,you will receive an email to reset you password

User will receive an email regarding the reset of password.


11:59

Reset your password for AuroTalk Inbox

AuroTalk 5:27 PM
to me

Hello,

Follow this link to reset your project-304924081339 password for your dhwanipatel023@gmail.com account.

https://chatapp-6ff4a.firebaseapp.com/__/auth/action?mode=resetPassword&oobCode=zTN5YuIVsoq5F-4oppaQSKJZ5pEKxttvmqSMsWnQYCMAAAF2R11XKw&apiKey=AIzaSyBtQ09lr4Lg4xaPniuMuZaSoYzaSkJg1Co&lang=en

If you didn't ask to reset your password, you can ignore this email.

Thanks,

Your project-304924081339 team

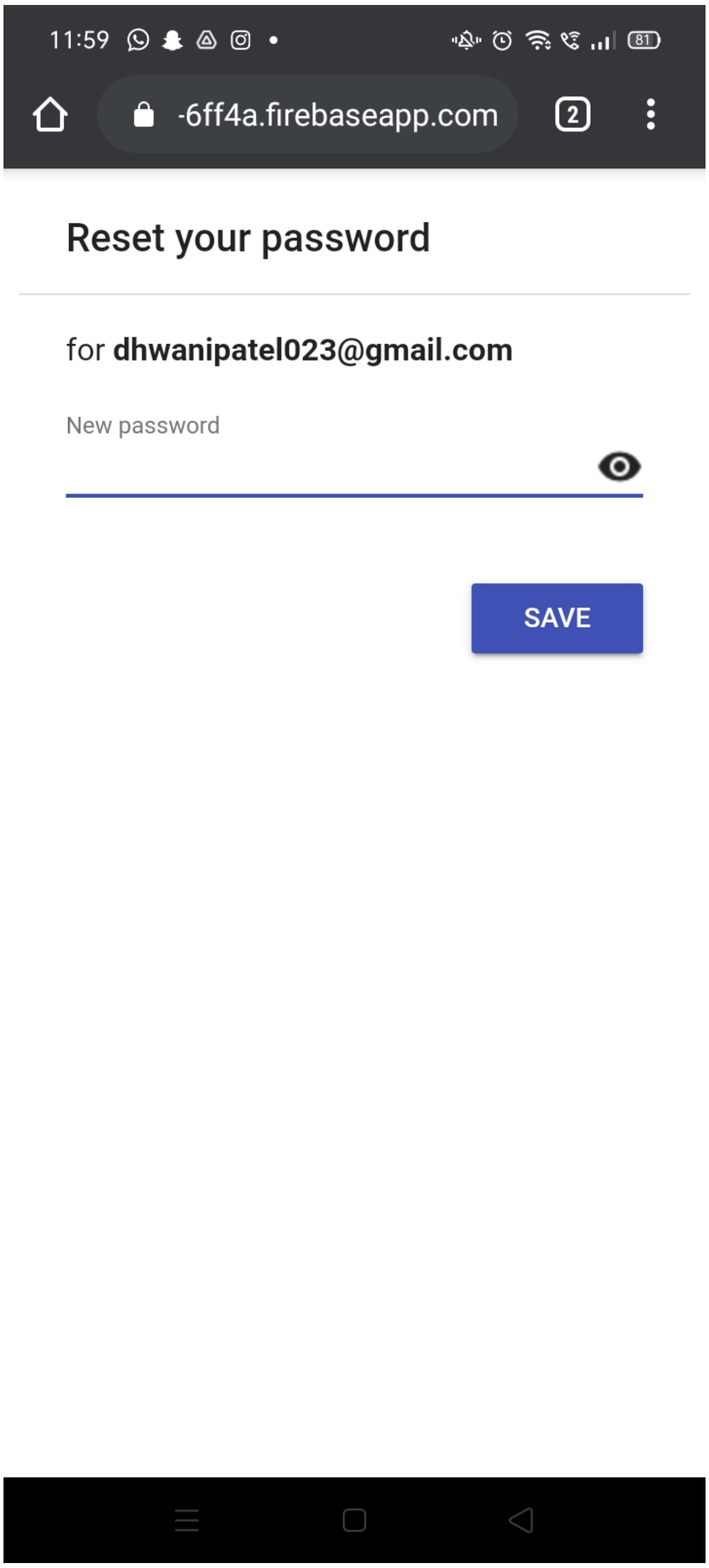
AuroTalk 11:58 PM
to me

Hello,

Follow this link to reset your project-304924081339 password for your dhwanipatel023@gmail.com account.

New Password

From here user can enter new password.



11:59

-6ff4a.firebaseapp.com

Reset your password

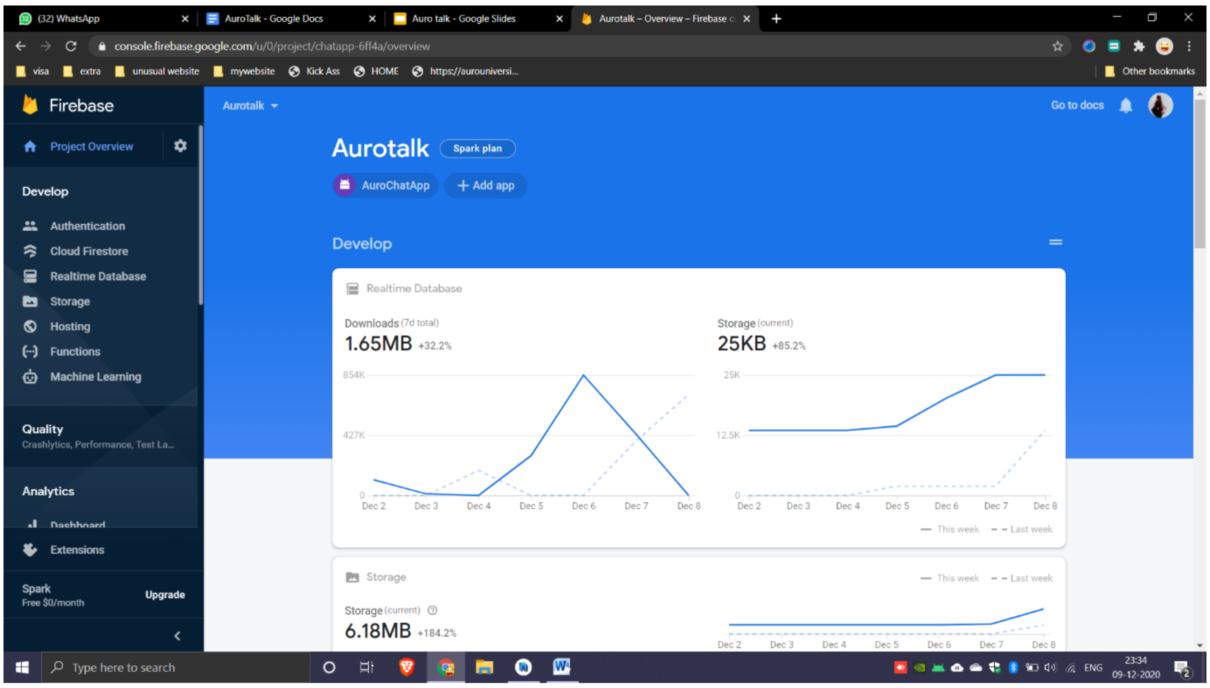
for **dhwanipatel023@gmail.com**

New password

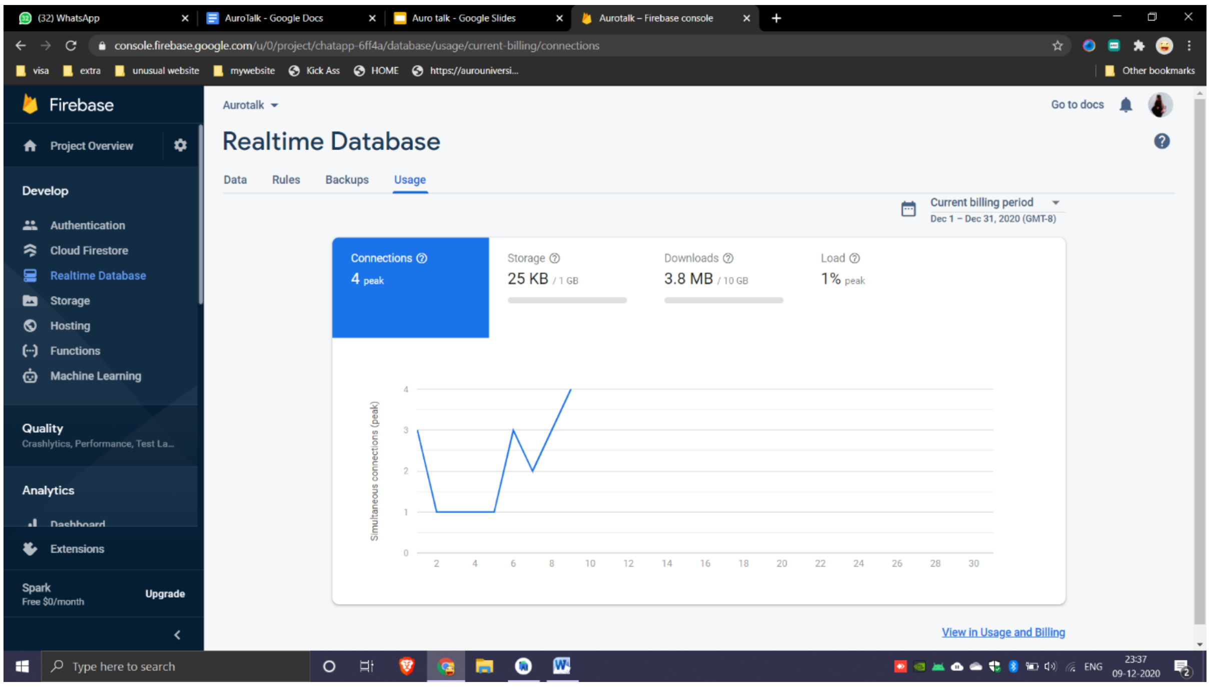
SAVE

BACK-END SCREEN SHOTS

Analytics


Firebase
Project Overview
Develop
Authentication
Cloud Firestore
Realtime Database
Storage
Hosting
Functions
Machine Learning
Quality
Crashlytics, Performance, Test La...
Analytics
Dashboard
Extensions
Spark
Free $0/month
Upgrade
Aurotalk
Go to docs
Aurotalk
Spark plan
AuroChatApp
Add app
Develop
Realtime Database
Downloads (7d total)
1.65MB +32.2%
Storage (current)
25KB +85.2%
This week
Last week
Storage
Storage (current)
6.18MB +184.2%
Type here to search

Database Usage Graph



console.firebase.google.com/u/0/project/chatapp-6ff4a/database/usage/current-billing/connections

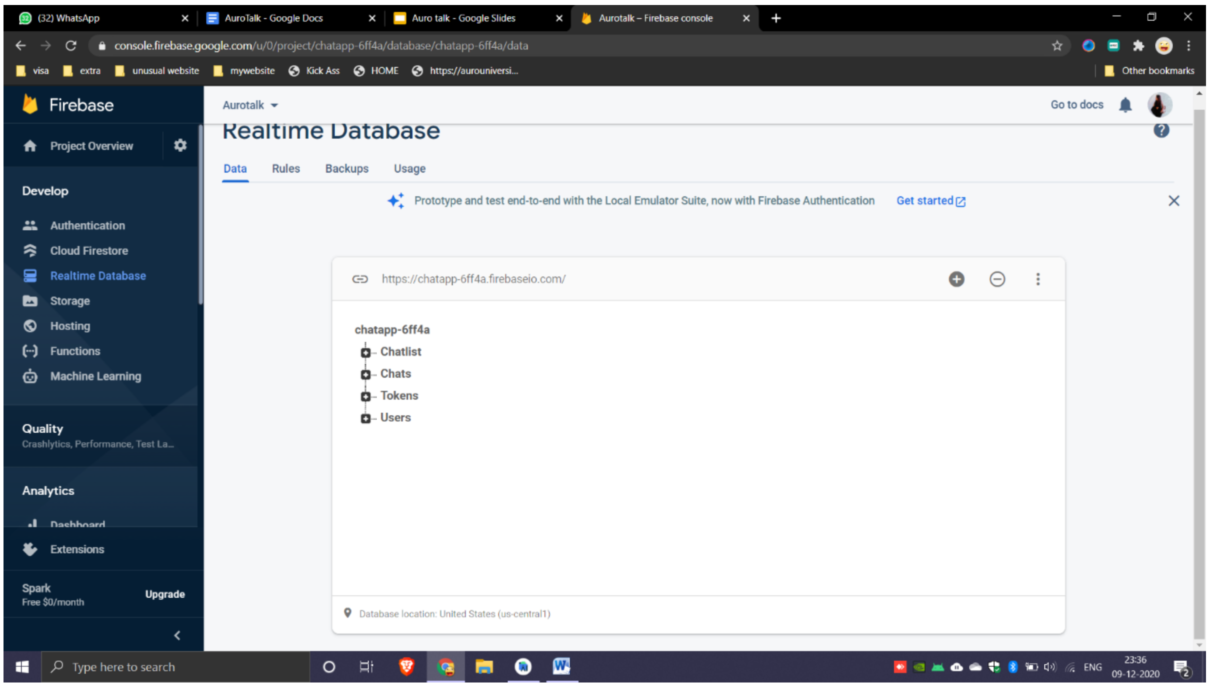
Realtime Database

Current billing period
Dec 1 – Dec 31, 2020 (GMT-8)

Connections: 4 peak
Storage: 25 KB / 1 GB
Downloads: 3.8 MB / 10 GB
Load: 1% peak

Realtime database

It includes chatlist, chats, tokens, users.



Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication Get started

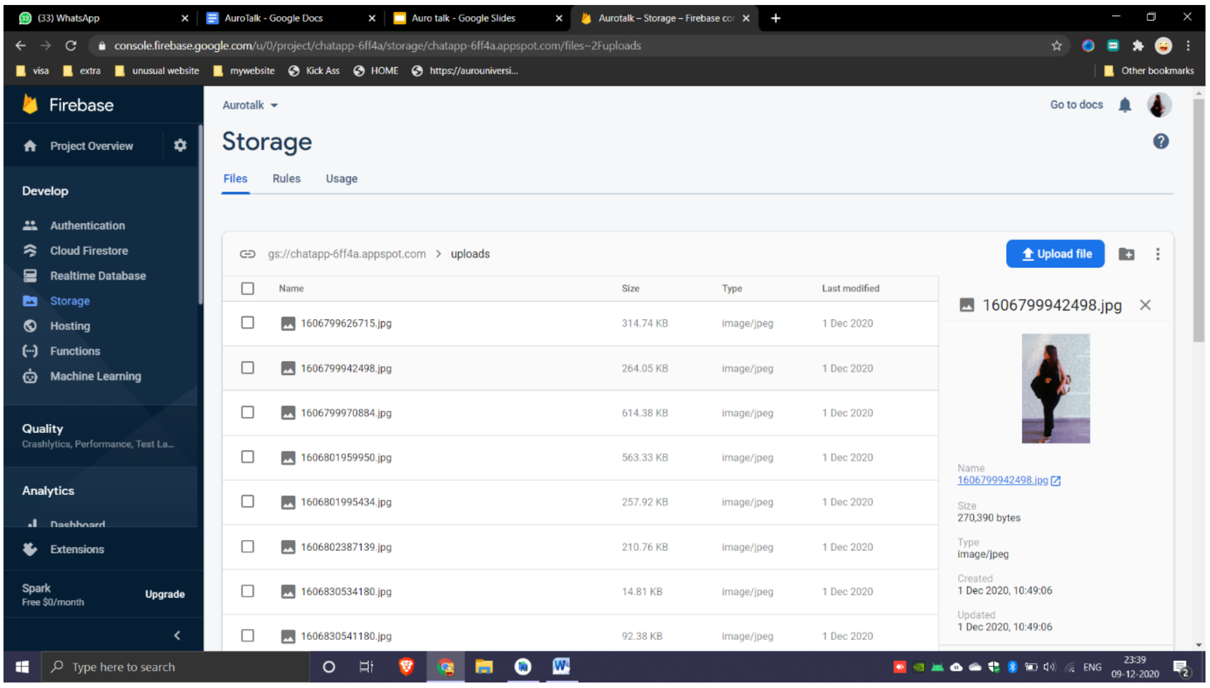
https://chatapp-6ff4a.firebaseio.com/

chatapp-6ff4a
- Chatlist
- Chats
- Tokens
- Users

Database location: United States (us-central1)

Storage

Stores the profile picture of the user.



Name	Size	Type	Last modified
1606799626715.jpg	314.74 KB	image/jpeg	1 Dec 2020
1606799942498.jpg	264.05 KB	image/jpeg	1 Dec 2020
1606799970884.jpg	614.38 KB	image/jpeg	1 Dec 2020
1606801959950.jpg	563.33 KB	image/jpeg	1 Dec 2020
1606801995434.jpg	257.92 KB	image/jpeg	1 Dec 2020
1606802387139.jpg	210.76 KB	image/jpeg	1 Dec 2020
1606830534180.jpg	14.81 KB	image/jpeg	1 Dec 2020
1606830541180.jpg	92.38 KB	image/jpeg	1 Dec 2020

1606799942498.jpg

Name
1606799942498.jpg

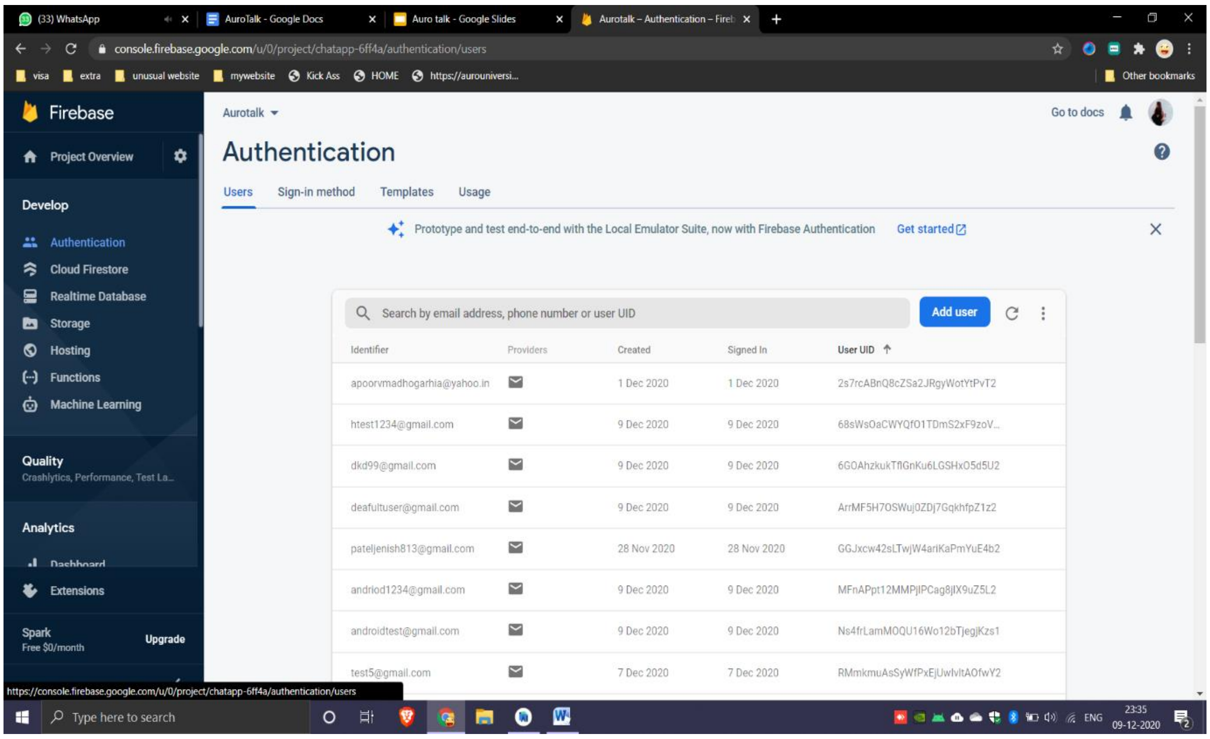
Size
270,390 bytes

Type
image/jpeg

Created
1 Dec 2020, 10:49:06

Updated
1 Dec 2020, 10:49:06

Authentication of User Email Id and user ID.



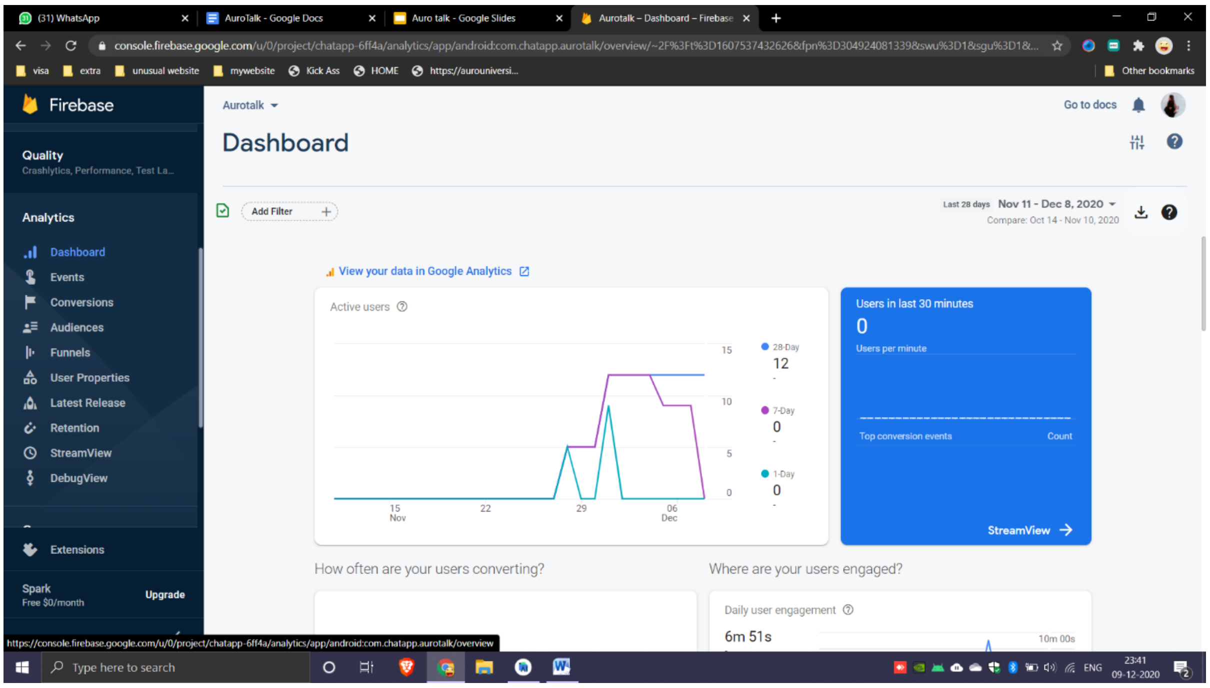
Authentication

Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication

Identifier	Providers	Created	Signed In	User UID
apoorvmadhogarhia@yahoo.in	✉	1 Dec 2020	1 Dec 2020	2s7rcABnQ8cZSa2JRgyWotYtPvT2
htest1234@gmail.com	✉	9 Dec 2020	9 Dec 2020	68sWsOaCWYQfO1TDmS2xF9zoV...
dkd99@gmail.com	✉	9 Dec 2020	9 Dec 2020	6GOAhzkukTflGnKu6LGSHxO5d5U2
deafultuser@gmail.com	✉	9 Dec 2020	9 Dec 2020	ArrMF5H7OSWuj0ZDj7GqkhfpZ1z2
pateljenish813@gmail.com	✉	28 Nov 2020	28 Nov 2020	GGJxcw42sLTwjW4arlKaPmYuE4b2
andriod1234@gmail.com	✉	9 Dec 2020	9 Dec 2020	MFnAPpt12MMPjlPCag8jlX9uZ5L2
androidtest@gmail.com	✉	9 Dec 2020	9 Dec 2020	Ns4frLamMOQU16Wo12bTjegjKzs1
test5@gmail.com	✉	7 Dec 2020	7 Dec 2020	RMmkmuAsSyWfPxEjUwlvltAOfwY2

Dashboard

Here we can see active users' graphs, daily user engaged.



SOURCE CODE

XML files

Color.xml

```
<resources>
  <color name="colorPrimary">#638EDD</color>
  <color name="colorPrimaryDark">#2B4D8B</color>
  <color name="colorAccent">#27226E</color>
</resources>
```

Styles.xml

```
<resources>

  <!-- Base application theme. -->
  <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
     <!-- Customize your theme here. -->
     <item name="colorPrimary">@color/colorPrimary</item>
     <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
     <item name="colorAccent">@color/colorAccent</item>
  </style>

  <style name="MenuStyle" parent="Theme.AppCompat.Light">
     <item name="android:background">@android:color/white</item>
  </style>

</resources>
```

Activity_start.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:gravity="center"
  android:orientation="vertical"
  android:background="#FEFCFC"
  android:layout_height="match_parent"
  tools:context=".StartActivity">

 <pl.droidsonroids.gif.GifImageView
   android:layout_width="150dp"
   android:layout_height="120dp"
   android:background="@drawable/start"/>
```

```
<!--<ImageView
        android:layout_width="72dp"
        android:layout_height="71dp"
        android:background="@drawable/common_google_signin_btn_text_light" />-->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Auro Talk"
        android:layout_marginTop="15dp"
        android:textColor="@color/colorPrimaryDark"
        android:textSize="25sp"
        android:textStyle="bold"/>
  <!-- <ImageView
       android:layout_width="wrap_content"
       android:layout_height="200dp"
       android:background="@drawable/logo"/> -->

   <Button
       android:id="@+id/login"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:layout_marginTop="10dp"
       android:layout_marginLeft="10dp"
       android:layout_marginRight="10dp"
       android:background="@color/colorPrimaryDark"
       android:text="login"
       android:textColor="#fff" />
   <Button
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:background="@color/colorPrimaryDark"
       android:textColor="#fff"
       android:layout_marginLeft="10dp"
       android:layout_marginRight="10dp"
       android:layout_marginTop="10dp"
       android:id="@+id/register"
       android:text="register"/>
</LinearLayout>
```

Barlayout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.Toolbar
xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
```

```
  android:layout_height="wrap_content"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  android:id="@+id/toolbar"
  android:background="@color/colorPrimaryDark"
  android:theme="@style/Base.ThemeOverlay.AppCompat.Dark.ActionBar"
  app:popupTheme="@style/MenuStyle">


</android.support.v7.widget.Toolbar>
```

Activity_register.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="#FFFFFF"
  tools:context=".RegisterActivity">

  <include
     android:id="@+id/toolbar"
     layout="@layout/bar_layout"/>

  <pl.droidsonroids.gif.GifImageView
     android:id="@+id/gif"
     android:layout_width="145dp"
     android:layout_height="150dp"
     android:layout_centerHorizontal="true"
     android:layout_marginTop="60dp"
     android:layout_marginBottom="10dp"
     android:background="@drawable/join" />

  <android.support.v7.widget.CardView
     android:layout_width="match_parent"
     android:layout_height="wrap_content"

     android:layout_below="@+id/gif"
     android:layout_marginStart="15dp"
     android:layout_marginTop="0dp"
     android:layout_marginEnd="15dp"
     android:layout_marginBottom="15dp"
     android:background="@color/colorPrimary"
     android:padding="10dp"
     app:cardCornerRadius="5dp"
     app:cardElevation="5dp">
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/toolbar"
    android:layout_marginTop="10dp"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:text="Create a new account"
        android:textSize="20sp"
        android:textStyle="bold" />

    <com.rengwuxian.materialedittext.MaterialEditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:hint="Username"
        app:met_floatingLabel="normal" />

    <com.rengwuxian.materialedittext.MaterialEditText
        android:id="@+id/email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:hint="Email"
        android:inputType="textEmailAddress"
        app:met_floatingLabel="normal" />

    <com.rengwuxian.materialedittext.MaterialEditText
        android:id="@+id/password"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:hint="Password"
        android:inputType="textPassword"
        app:met_floatingLabel="normal" />

  <!--  <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:text="Are you a?"
        android:textSize="18dp" /> -->
        <RadioGroup
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textAlignment="center">

            <RadioButton
                android:id="@+id/radioButton2"
                android:textColor="#717171"
                android:layout_marginTop="10dp"
                android:textSize="18dp"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Student" />

            <RadioButton
                android:id="@+id/radioButton"
                android:layout_width="match_parent"
                android:textSize="18dp"
                android:textColor="#717171"
                android:layout_height="wrap_content"
                android:text="Faculty" />
        </RadioGroup>

        <Button
            android:id="@+id/btn_register"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:background="@color/colorPrimaryDark"
            android:text="register"
            android:layout_marginBottom="10dp"
            android:textColor="#fff" />


    </LinearLayout>

  </android.support.v7.widget.CardView>
</RelativeLayout>
```

Fragment_user.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
```

```
    tools:context=".Fragments.UsersFragment">


    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="2dp"
        android:hint="Search..."
        android:id="@+id/search_users"
        android:layout_toRightOf="@id/btnsearch"
        />
    <ImageView
        android:id="@+id/btnsearch"
        android:layout_marginTop="5dp"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:background="@android:drawable/ic_search_category_default"
    />

    <android.support.v7.widget.RecyclerView
        android:layout_below="@id/search_users"
        android:id="@+id/recycler_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</RelativeLayout>
```

User_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:padding="10dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="5dp"
    android:background="#E8E8E8">

    <de.hdodenhof.circleimageview.CircleImageView
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:id="@+id/profile_image"
        android:src="@mipmap/ic_launcher"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="username"
        android:id="@+id/username"
        android:layout_toRightOf="@id/profile_image"
        android:layout_toEndOf="@id/profile_image"
        android:layout_marginLeft="10dp"
        android:layout_centerVertical="true"
        android:textSize="18sp"/>

    <de.hdodenhof.circleimageview.CircleImageView
        android:layout_width="15dp"
        android:layout_height="15dp"
        app:civ_border_width="10dp"
        app:civ_border_color="#05df29"
        android:id="@+id/img_on"
        android:visibility="gone"
        android:src="@mipmap/ic_launcher"
        android:layout_below="@id/username"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="-15dp"
        android:layout_toRightOf="@id/profile_image"
        android:layout_toEndOf="@id/profile_image"/>

  <de.hdodenhof.circleimageview.CircleImageView
        android:layout_width="15dp"
        android:layout_height="15dp"
        app:civ_border_width="10dp"
        app:civ_border_color="#bfbfbf"
        android:id="@+id/img_off"
        android:visibility="gone"
        android:src="@mipmap/ic_launcher"
        android:layout_below="@id/username"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="-15dp"
        android:layout_toRightOf="@id/profile_image"
        android:layout_toEndOf="@id/profile_image"/>

   <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/username"
        android:layout_toRightOf="@id/profile_image"
        android:layout_marginTop="5dp"
        android:paddingTop="5dp"
        android:id="@+id/last_msg"
        android:textColor="@color/colorPrimaryDark"
        android:layout_marginLeft="20dp"
        android:maxLines="1"/>

  </RelativeLayout>
```

Fragment_profile.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="8dp"
  tools:context=".Fragments.ProfileFragment">

  <android.support.v7.widget.CardView
     android:layout_width="match_parent"
     android:layout_height="wrap_content">

     <RelativeLayout
        android:layout_width="match_parent"
        android:padding="8dp"
        android:layout_height="wrap_content">

        <TextView
           android:layout_width="wrap_content"
           android:layout_height="wrap_content"
           android:id="@+id/profile"
           android:text="Profile"
           android:textColor="@color/colorPrimaryDark"
           android:textStyle="bold"/>

        <de.hdodenhof.circleimageview.CircleImageView
           android:layout_width="150dp"
           android:layout_height="150dp"
           android:id="@+id/profile_image"
           android:layout_centerHorizontal="true"
           android:layout_marginTop="50dp"
           android:src="@mipmap/ic_launcher"/>

        <TextView
           android:layout_width="wrap_content"
           android:layout_height="wrap_content"
           android:text="username"
           android:layout_below="@id/profile_image"
           android:layout_centerHorizontal="true"
           android:layout_marginTop="15dp"
           android:textColor="@color/colorPrimaryDark"
           android:textStyle="bold"
           android:id="@+id/username"
           android:textSize="18sp"
          />

        <Button
           android:id="@+id/buttonLogout"
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/username"
            android:layout_centerHorizontal="true"
            android:background="@color/colorPrimaryDark"
            android:textColor="#fff"
            android:layout_marginTop="34dp"
            android:text="Logout"
            android:onClick="logout"
            android:layout_marginBottom="20dp"/>

    </RelativeLayout>



  </android.support.v7.widget.CardView>



</RelativeLayout>
```

Menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto">

  <item
     android:id="@+id/logout"
     android:title="Logout"
     app:showAsAction="never"/>

</menu>
```

Activity_login.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="#FFFFFF"
  tools:context=".LoginActivity">
```

```
<include
   android:id="@+id/toolbar"
   layout="@layout/bar_layout"/>

<pl.droidsonroids.gif.GifImageView
   android:layout_width="100dp"
   android:layout_height="100dp"
   android:layout_marginTop="90dp"
   android:layout_centerHorizontal="true"
   android:background="@drawable/login"
   />

<android.support.v7.widget.CardView
   xmlns:Card_View="http://schemas.android.com/apk/res-auto"
   android:layout_width="match_parent"
   android:layout_height="300dp"
   android:layout_margin="15dp"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="true"
   android:padding="10dp"
   Card_View:cardCornerRadius="5dp"
   Card_View:cardElevation="5dp">


<LinearLayout
   android:layout_width="match_parent"
   android:orientation="vertical"
   android:gravity="center_horizontal"
   android:layout_below="@id/toolbar"
   android:padding="16dp"
   android:layout_height="wrap_content">

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Login"
      android:layout_marginTop="5dp"
      android:textSize="20sp"
      android:textStyle="bold"/>

   <com.rengwuxian.materialedittext.MaterialEditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/email"
      android:inputType="textEmailAddress"
      android:layout_marginTop="10dp"
      app:met_floatingLabel="normal"
      android:hint="Email"/>

   <com.rengwuxian.materialedittext.MaterialEditText
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/password"
            android:inputType="textPassword"
            android:layout_marginTop="10dp"
            app:met_floatingLabel="normal"
            android:hint="Password"/>

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="login"
            android:id="@+id/btn_login"
            android:background="@color/colorPrimaryDark"
            android:layout_marginTop="10dp"
            android:textColor="#fff"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="Forgot your password?"
            android:layout_marginTop="10dp"
            android:layout_gravity="end"
            android:textStyle="bold"
            android:id="@+id/forgot_password"
            android:textColor="@color/colorPrimaryDark"/>


    </LinearLayout>

    </android.support.v7.widget.CardView>


</RelativeLayout>
```

Fragment_chat.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Fragments.ChatsFragment">

    <android.support.v7.widget.RecyclerView
        android:id="@+id/recycler_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
</RelativeLayout>
```

Chat_left_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="300dp"
  android:padding="8dp"
  android:layout_height="wrap_content">

  <de.hdodenhof.circleimageview.CircleImageView
      android:layout_width="40dp"
      android:layout_height="40dp"
      android:id="@+id/profile_image"
      android:src="@mipmap/ic_launcher"/>

  <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_toRightOf="@id/profile_image"
      android:layout_marginLeft="5dp"
      android:text="hello"
      android:id="@+id/show_message"
      android:textSize="18sp"
      android:padding="8dp"
      android:background="@drawable/background_left"/>

  <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/txt_seen"
      android:visibility="gone"
      android:layout_below="@id/show_message"/>


</RelativeLayout>
```

Chat_right_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:layout_width="match_parent"
    android:padding="8dp"
    android:layout_height="wrap_content">

    <RelativeLayout
        android:layout_width="300dp"
        android:layout_alignParentEnd="true"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true">

        <de.hdodenhof.circleimageview.CircleImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/profile_image"
            android:visibility="gone"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="@drawable/background_right"
            android:text="hello"
            android:layout_alignParentEnd="true"
            android:id="@+id/show_message"
            android:textSize="18sp"
            android:textColor="#fff"
            android:padding="8dp"
            android:layout_alignParentRight="true" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/txt_seen"
            android:layout_below="@id/show_message"
            android:layout_alignParentRight="true"
            android:layout_alignParentEnd="true" />

    </RelativeLayout>

</RelativeLayout>
```

activity_reset_Password.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:orientation="vertical"
android:background="#FFFFFF"
tools:context=".ResetPasswordActivity">

<include
   android:id="@+id/toolbar"
   layout="@layout/bar_layout"/>
<pl.droidsonroids.gif.GifImageView
   android:layout_width="100dp"
   android:layout_height="100dp"
   android:background="@drawable/forget"
   android:layout_centerHorizontal="true"
   android:layout_marginTop="90dp"/>

<android.support.v7.widget.CardView
   xmlns:Card_View="http://schemas.android.com/apk/res-auto"
   android:layout_width="match_parent"
   android:layout_height="250dp"
   android:layout_margin="15dp"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="true"
   android:background="@color/colorPrimary"
   android:padding="10dp"
   Card_View:cardCornerRadius="5dp"
   Card_View:cardElevation="5dp">

<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical"
   android:padding="16dp">

   <com.rengwuxian.materialedittext.MaterialEditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/send_email"
      android:inputType="textEmailAddress"
      app:met_floatingLabel="normal"
      android:hint="Your Email"
      android:layout_marginTop="20dp"/>

   <Button
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Reset"
      android:id="@+id/btn_reset"
      android:textColor="#fff"
      android:background="@color/colorPrimaryDark"
      android:layout_marginTop="10dp"/>
```

```
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:layout_marginTop="15dp"
            android:layout_marginLeft="15dp"
            android:layout_marginRight="15dp"
            android:textColor="#707071"
            android:text="By clicking RESET,you will receive an email to reset you password"/>

    </LinearLayout>
    </android.support.v7.widget.CardView>
</RelativeLayout>
```

JAVA Code

Resetpassword.java

```
package com.chatapp.aurotalk;

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.chatapp.aurotalk.R;

public class ResetPasswordActivity extends AppCompatActivity {

    EditText send_email;
    Button btn_reset;

    FirebaseAuth firebaseAuth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_reset_password);
        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        getSupportActionBar().setTitle("Reset Password");
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);

        send_email = findViewById(R.id.send_email);
        btn_reset = findViewById(R.id.btn_reset);

        firebaseAuth = FirebaseAuth.getInstance();

        btn_reset.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String email = send_email.getText().toString();

                if (email.equals("")){
                    Toast.makeText(ResetPasswordActivity.this, "All fileds are required!",
Toast.LENGTH_SHORT).show();
                } else {
                    firebaseAuth.sendPasswordResetEmail(email).addOnCompleteListener(new
OnCompleteListener<Void>() {
                        @Override
                        public void onComplete(@NonNull Task<Void> task) {
                            if (task.isSuccessful()){
                                Toast.makeText(ResetPasswordActivity.this, "Please check you Email",
Toast.LENGTH_SHORT).show();
                                startActivity(new Intent(ResetPasswordActivity.this,
LoginActivity.class));
                            } else {
                                String error = task.getException().getMessage(); //catching exception in
form of message
                                Toast.makeText(ResetPasswordActivity.this, error,
Toast.LENGTH_SHORT).show();
                            }
                        }
                    });
                }
            }
        });

    }
}
```

Registeractivity.java

```
package com.chatapp.aurotalk;

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.chatapp.aurotalk.R;
import com.rengwuxian.materialedittext.MaterialEditText;

import java.util.HashMap;

public class RegisterActivity extends AppCompatActivity {

  MaterialEditText username, email, password;
  Button btn_register;

  FirebaseAuth auth;
  DatabaseReference reference;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_register);

     Toolbar toolbar = findViewById(R.id.toolbar);
     setSupportActionBar(toolbar);
     getSupportActionBar().setTitle("Register");
     getSupportActionBar().setDisplayHomeAsUpEnabled(true);

     username = findViewById(R.id.username);
     email = findViewById(R.id.email);
     password = findViewById(R.id.password);
     btn_register = findViewById(R.id.btn_register);

     auth = FirebaseAuth.getInstance();
```

```
        btn_register.setOnClickListener(new View.OnClickListener() {
            @Override

            public void onClick(View view) {
                String txt_username = username.getText().toString();
                String txt_email = email.getText().toString();
                String txt_password = password.getText().toString();

                if (TextUtils.isEmpty(txt_username) || TextUtils.isEmpty(txt_email) ||
TextUtils.isEmpty(txt_password)){
                    Toast.makeText(RegisterActivity.this, "All the fields are required",
Toast.LENGTH_SHORT).show();
                } else if (txt_password.length() < 6 ){
                    Toast.makeText(RegisterActivity.this, "Password must be at least 6 characters",
Toast.LENGTH_SHORT).show();
                }
                else if (!txt_email.endsWith("@aurouniversity.edu.in")){
                    Toast.makeText(RegisterActivity.this, "Enter Valid
Address",Toast.LENGTH_SHORT).show();
                }
                else {
                    register(txt_username, txt_email, txt_password);
                }
            }
        });
    }

    private void register(final String username, String email, String password){

        auth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        if (task.isSuccessful()){
                            FirebaseUser firebaseUser = auth.getCurrentUser();
                            assert firebaseUser != null;
                            String userid = firebaseUser.getUid();

                            reference =
FirebaseDatabase.getInstance().getReference("Users").child(userid);

                            HashMap<String, String> hashMap = new HashMap<>();
                            hashMap.put("id", userid);//id key to userid value
                            hashMap.put("username", username);// username key to username value
                            hashMap.put("imageURL", "default");// default iclaunger
                            hashMap.put("status", "offline");//default status offline
                            hashMap.put("search", username.toLowerCase());// change lowercase and
search
```

```
                        reference.setValue(hashMap).addOnCompleteListener(new
OnCompleteListener<Void>() {
                            @Override
                            public void onComplete(@NonNull Task<Void> task) {
                                if (task.isSuccessful()){
                                    Intent intent = new Intent(RegisterActivity.this, MainActivity.class);
                                    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK                  |
Intent.FLAG_ACTIVITY_NEW_TASK);
                                    startActivity(intent);
                                    finish();
                                }
                            }
                        });
                    } else {
                        Toast.makeText(RegisterActivity.this, "You can't register with this email or
password", Toast.LENGTH_SHORT).show();
                    }
                }
            });
    }
}
```

Chatfragment.java

```
package com.chatapp.aurotalk.Fragments;

import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.Fragment;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import com.chatapp.aurotalk.Adapter.UserAdapter;
import com.chatapp.aurotalk.Model.Chatlist;
import com.chatapp.aurotalk.Model.User;
import com.chatapp.aurotalk.Notifications.Token;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.iid.FirebaseInstanceId;
import com.chatapp.aurotalk.R;
import java.util.ArrayList;
import java.util.List;


public class ChatsFragment extends Fragment {

  private RecyclerView recyclerView;

  private UserAdapter userAdapter;
  private List<User> mUsers;

  FirebaseUser fuser;
  DatabaseReference reference;

  private List<Chatlist> usersList;

  @Override
  public View onCreateView(LayoutInflater inflater, ViewGroup container,
                           Bundle savedInstanceState) {
      View view = inflater.inflate(R.layout.fragment_chats, container, false);

      recyclerView = view.findViewById(R.id.recycler_view);
      recyclerView.setHasFixedSize(true);
      recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));

      fuser = FirebaseAuth.getInstance().getCurrentUser();

      usersList = new ArrayList<>();

      reference                                                              =
FirebaseDatabase.getInstance().getReference("Chatlist").child(fuser.getUid());
      reference.addValueEventListener(new ValueEventListener() {
          @Override
          public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
              usersList.clear();
              for (DataSnapshot snapshot : dataSnapshot.getChildren()){
                  Chatlist chatlist = snapshot.getValue(Chatlist.class);
                  usersList.add(chatlist);
              }

              chatList();
          }

          @Override
          public void onCancelled(@NonNull DatabaseError databaseError) {

          }
      });
```

```
        updateToken(FirebaseInstanceId.getInstance().getToken());


        return view;
    }

    private void updateToken(String token){
        DatabaseReference reference = FirebaseDatabase.getInstance().getReference("Tokens");
        Token token1 = new Token(token);
        reference.child(fuser.getUid()).setValue(token1);
    }

    private void chatList() {
        mUsers = new ArrayList<>();
        reference = FirebaseDatabase.getInstance().getReference("Users");
        reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                mUsers.clear();
                for (DataSnapshot snapshot : dataSnapshot.getChildren()){
                    User user = snapshot.getValue(User.class);
                    for (Chatlist chatlist : usersList){
                        if (user.getId().equals(chatlist.getId())){
                            mUsers.add(user);
                        }
                    }
                }
                userAdapter = new UserAdapter(getContext(), mUsers, true);
                recyclerView.setAdapter(userAdapter);
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });
    }

}
```

Loginacitvity.java

```
package com.chatapp.aurotalk;

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.chatapp.aurotalk.R;
import com.rengwuxian.materialedittext.MaterialEditText;

public class LoginActivity extends AppCompatActivity {

  MaterialEditText email, password;
  Button btn_login;

  FirebaseAuth auth;
  TextView forgot_password;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login);

    Toolbar toolbar = findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);
    getSupportActionBar().setTitle("Login");
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);

    auth = FirebaseAuth.getInstance();

    email = findViewById(R.id.email);
    password = findViewById(R.id.password);
    btn_login = findViewById(R.id.btn_login);
    forgot_password = findViewById(R.id.forgot_password);

    forgot_password.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View view) {
        startActivity(new Intent(LoginActivity.this, ResetPasswordActivity.class));
      }
    });

    btn_login.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View view) {
        String txt_email = email.getText().toString();
```

```
            String txt_password = password.getText().toString();

            if (TextUtils.isEmpty(txt_email) || TextUtils.isEmpty(txt_password)){
               Toast.makeText(LoginActivity.this,       "All       fileds       are       required",
Toast.LENGTH_SHORT).show();
            } else {

               auth.signInWithEmailAndPassword(txt_email, txt_password)
                     .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                        @Override
                        public void onComplete(@NonNull Task<AuthResult> task) {
                           if (task.isSuccessful()){
                              Intent intent = new Intent(LoginActivity.this, MainActivity.class);
                              intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK                |
Intent.FLAG_ACTIVITY_NEW_TASK);
                              startActivity(intent);
                              finish();
                           } else {
                              Toast.makeText(LoginActivity.this,      "Authentication      failed!",
Toast.LENGTH_SHORT).show();
                           }
                        }
                     });
            }
         }
      });
   }
}
```

model/chat.java

```
package com.chatapp.aurotalk.Model;

public class Chat {

   private String sender;
   private String receiver;
   private String message;
   private boolean isseen;

   public Chat(String sender, String receiver, String message, boolean isseen) {
      this.sender = sender;
      this.receiver = receiver;
      this.message = message;
      this.isseen = isseen;
   }

   public Chat() {
   }

   public String getSender() {
      return sender;
   }

   public void setSender(String sender) {
      this.sender = sender;
   }

   public String getReceiver() {
      return receiver;
   }

   public void setReceiver(String receiver) {
      this.receiver = receiver;
   }

   public String getMessage() {
      return message;
   }

   public void setMessage(String message) {
      this.message = message;
   }

   public boolean isIsseen() {
      return isseen;
   }

   public void setIsseen(boolean isseen) {
      this.isseen = isseen;
   }
}
```

Userfragment.java

```
package com.chatapp.aurotalk.Fragments;

import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.Fragment;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.EditText;

import com.chatapp.aurotalk.Adapter.UserAdapter;
import com.chatapp.aurotalk.Model.User;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;
import com.google.firebase.database.ValueEventListener;
import com.chatapp.aurotalk.R;

import java.util.ArrayList;
import java.util.List;


public class UsersFragment extends Fragment {

  private RecyclerView recyclerView;

  private UserAdapter userAdapter;
  private List<User> mUsers;

  EditText search_users;


  @Override
  public View onCreateView(LayoutInflater inflater, ViewGroup container,
                           Bundle savedInstanceState) {

      View view = inflater.inflate(R.layout.fragment_users, container, false);

      recyclerView = view.findViewById(R.id.recycler_view);
      recyclerView.setHasFixedSize(true);
      recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));

      mUsers = new ArrayList<>();

      readUsers();

      search_users = view.findViewById(R.id.search_users);
      search_users.addTextChangedListener(new TextWatcher() {
          @Override
          public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {

          }
```

```
            @Override
            public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {
                searchUsers(charSequence.toString().toLowerCase());
            }

            @Override
            public void afterTextChanged(Editable editable) {

            }
        });

        return view;
    }
  private void searchUsers(String s) {

        final FirebaseUser fuser = FirebaseAuth.getInstance().getCurrentUser();
        Query query =
  FirebaseDatabase.getInstance().getReference("Users").orderByChild("search")
                .startAt(s)
                .endAt(s+"\uf8ff");

        query.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                mUsers.clear();
                for (DataSnapshot snapshot : dataSnapshot.getChildren()){
                    User user = snapshot.getValue(User.class);

                    assert user != null;
                    assert fuser != null;
                    if (!user.getId().equals(fuser.getUid())){
                        mUsers.add(user);
                    }
                }

                userAdapter = new UserAdapter(getContext(), mUsers, false);
                recyclerView.setAdapter(userAdapter);
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });

    }

    private void readUsers() {
        final FirebaseUser firebaseUser = FirebaseAuth.getInstance().getCurrentUser();
        DatabaseReference reference = FirebaseDatabase.getInstance().getReference("Users");

        reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                if (search_users.getText().toString().equals("")) {
                    mUsers.clear();
                    for (DataSnapshot snapshot : dataSnapshot.getChildren()) {
                        User user = snapshot.getValue(User.class);

                        if (!user.getId().equals(firebaseUser.getUid())) {
                            mUsers.add(user);
                        }

                    }

                    userAdapter = new UserAdapter(getContext(), mUsers, false);
                    recyclerView.setAdapter(userAdapter);
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });
    }
}
```

model/chatlist.java

```
package com.chatapp.aurotalk.Model;

public class Chatlist {
    public String id;

    public Chatlist(String id) {
        this.id = id;
    }

    public Chatlist() {
    }

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
}
```

model/user.java

```
package com.chatapp.aurotalk.Model;

public class User {

    private String id;
    private String username;
    private String imageURL;
    private String status;
    private String search;

    public User(String id, String username, String imageURL, String status, String search) {
        this.id = id;
        this.username = username;
        this.imageURL = imageURL;
        this.status = status;
        this.search = search;
    }

    public User() {

    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getImageURL() {
        return imageURL;
    }
    public void setImageURL(String imageURL) {
        this.imageURL = imageURL;
    }

    public String getStatus() {
        return status;
    }

    public void setStatus(String status) {
        this.status = status;
    }

    public String getSearch() {
        return search;
    }

    public void setSearch(String search) {
        this.search = search;
    }
}
```

Startactivity.java

```
package com.chatapp.aurotalk;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.chatapp.aurotalk.R;

public class StartActivity extends AppCompatActivity {

    Button login, register;

    FirebaseUser firebaseUser;

    @Override
    protected void onStart() {
        super.onStart();

        firebaseUser = FirebaseAuth.getInstance().getCurrentUser();
```

```
            //check if user is null
            if (firebaseUser != null){
                Intent intent = new Intent(StartActivity.this, MainActivity.class);
                startActivity(intent);
                finish();
            }
        }

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_start);



            login = findViewById(R.id.login);
            register = findViewById(R.id.register);

            login.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    startActivity(new Intent(StartActivity.this, LoginActivity.class));
                }
            });

            register.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    startActivity(new Intent(StartActivity.this, RegisterActivity.class));
                }
            });
        }
    }
```

Profilefragment.java

```
package com.chatapp.aurotalk.Fragments;

import android.app.ProgressDialog;
import android.content.ContentResolver;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.webkit.MimeTypeMap;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.chatapp.aurotalk.Model.User;
import com.google.android.gms.tasks.Continuation;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.StorageTask;
import com.google.firebase.storage.UploadTask;
import com.chatapp.aurotalk.R;

import java.util.HashMap;

import de.hdodenhof.circleimageview.CircleImageView;

import static android.app.Activity.RESULT_OK;


public class ProfileFragment extends Fragment {

  CircleImageView image_profile;
  TextView username;
  Button Logout;
  DatabaseReference reference;
  FirebaseUser fuser;

  StorageReference storageReference;
  private static final int IMAGE_REQUEST = 1;
  private Uri imageUri;
  private StorageTask uploadTask;


  @Override
  public View onCreateView(LayoutInflater inflater, ViewGroup container,
                           Bundle savedInstanceState) {

     // Inflate the layout for this fragment
```

```
        View view = inflater.inflate(R.layout.fragment_profile, container, false);

        image_profile = view.findViewById(R.id.profile_image);
        username = view.findViewById(R.id.username);

        storageReference = FirebaseStorage.getInstance().getReference("uploads");

        fuser = FirebaseAuth.getInstance().getCurrentUser();
        reference = FirebaseDatabase.getInstance().getReference("Users").child(fuser.getUid());

        reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                User user = dataSnapshot.getValue(User.class);
                username.setText(user.getUsername());
                if (user.getImageURL().equals("default")){
                    image_profile.setImageResource(R.mipmap.ic_launcher);
                } else {
                    Glide.with(getContext()).load(user.getImageURL()).into(image_profile);
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });

        image_profile.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                openImage();
            }
        });

        return view;
    }

    private void openImage() {
        Intent intent = new Intent();
        intent.setType("image/*");
        intent.setAction(Intent.ACTION_GET_CONTENT);
        startActivityForResult(intent, IMAGE_REQUEST);
    }

    private String getFileExtension(Uri uri){
        ContentResolver contentResolver = getContext().getContentResolver();
        MimeTypeMap mimeTypeMap = MimeTypeMap.getSingleton();
        return mimeTypeMap.getExtensionFromMimeType(contentResolver.getType(uri));
    }
    private void uploadImage(){
        final ProgressDialog pd = new ProgressDialog(getContext());
        pd.setMessage("Uploading");
        pd.show();

        if (imageUri != null){
            final StorageReferencefileReference=
  storageReference.child(System.currentTimeMillis()
                +"."+getFileExtension(imageUri));

            uploadTask = fileReference.putFile(imageUri);
            uploadTask.continueWithTask(new        Continuation<UploadTask.TaskSnapshot,
  Task<Uri>>() {
                @Override
                public  Task<Uri>  then(@NonNull  Task<UploadTask.TaskSnapshot>  task)  throws
  Exception {
                    if (!task.isSuccessful()){
                        throw  task.getException();
                    }

                    return  fileReference.getDownloadUrl();
                }
            }).addOnCompleteListener(new OnCompleteListener<Uri>() {
                @Override
                public void onComplete(@NonNull Task<Uri> task) {
                    if (task.isSuccessful()){
                        Uri downloadUri = task.getResult();
                        String mUri = downloadUri.toString();

                        reference                                                    =
  FirebaseDatabase.getInstance().getReference("Users").child(fuser.getUid());
                        HashMap<String, Object> map = new HashMap<>();
                        map.put("imageURL", ""+mUri);
                        reference.updateChildren(map);

                        pd.dismiss();
                    } else {
                        Toast.makeText(getContext(), "Failed!", Toast.LENGTH_SHORT).show();
                        pd.dismiss();
                    }
                }
  }).addOnFailureListener(new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    Toast.makeText(getContext(), e.getMessage(), Toast.LENGTH_SHORT).show();
                    pd.dismiss();
                }
            });
        } else {
            Toast.makeText(getContext(), "No image selected", Toast.LENGTH_SHORT).show();
        }
    }

    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == IMAGE_REQUEST && resultCode == RESULT_OK
                && data != null && data.getData() != null){
            imageUri = data.getData();

            if (uploadTask != null && uploadTask.isInProgress()){
                Toast.makeText(getContext(),         "Upload         in         preogress",
Toast.LENGTH_SHORT).show();
            } else {
                uploadImage();
            }
        }
    }
}
```

Messageadapter.java

```
package com.chatapp.aurotalk.Adapter;

import android.content.Context;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import com.bumptech.glide.Glide;
import com.chatapp.aurotalk.Model.Chat;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.chatapp.aurotalk.R;

import java.util.List;

public class MessageAdapter extends RecyclerView.Adapter<MessageAdapter.ViewHolder>
{

    public static  final int MSG_TYPE_LEFT = 0;
    public static  final int MSG_TYPE_RIGHT = 1;
    private Context mContext;
    private List<Chat> mChat;//list view for chat
    private String imageurl;

    FirebaseUser fuser;

    public MessageAdapter(Context mContext, List<Chat> mChat, String imageurl){
        this.mChat = mChat;
        this.mContext = mContext;
        this.imageurl = imageurl;
    }

    @NonNull
    @Override
    public MessageAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent,
int viewType) {
        if (viewType == MSG_TYPE_RIGHT) {
            View view = LayoutInflater.from(mContext).inflate(R.layout.chat_item_right, parent,
false);
            return new MessageAdapter.ViewHolder(view);
        } else {
            View view = LayoutInflater.from(mContext).inflate(R.layout.chat_item_left, parent,
false);
            return new MessageAdapter.ViewHolder(view);
        }
    }

    @Override
    public void onBindViewHolder(@NonNull MessageAdapter.ViewHolder holder, int
position) {

        Chat chat = mChat.get(position);

        holder.show_message.setText(chat.getMessage());

        if (imageurl.equals("default")){
            holder.profile_image.setImageResource(R.mipmap.ic_launcher);
        } else {
            Glide.with(mContext).load(imageurl).into(holder.profile_image);
        }

        if (position == mChat.size()-1){
            if (chat.isIsseen()){ //setting text for text_seen id in chatfrag
                holder.txt_seen.setText("Seen");
            } else {
                holder.txt_seen.setText("Delivered");
            }
        } else {
            holder.txt_seen.setVisibility(View.GONE);//invisible
        }
    }

    @Override
    public int getItemCount() {
        return mChat.size();
    }

    public  class ViewHolder extends RecyclerView.ViewHolder{

        public TextView show_message;
        public ImageView profile_image;
        public TextView txt_seen;

        public ViewHolder(View itemView) {
            super(itemView);

            show_message = itemView.findViewById(R.id.show_message);
            profile_image = itemView.findViewById(R.id.profile_image);
            txt_seen = itemView.findViewById(R.id.txt_seen);
        }
    }

    @Override
    public int getItemViewType(int position) {
        fuser = FirebaseAuth.getInstance().getCurrentUser();
        if (mChat.get(position).getSender().equals(fuser.getUid())){
            return MSG_TYPE_RIGHT;
        } else {
            return MSG_TYPE_LEFT;
        }
    }
}
```

Useradapter.java

```
package com.chatapp.aurotalk.Adapter;

import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.graphics.Color;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
```

```java
import android.widget.TextView;

import android.widget.SimpleCursorAdapter;
import com.bumptech.glide.Glide;
import com.chatapp.aurotalk.MessageActivity;
import com.chatapp.aurotalk.Model.Chat;
import com.chatapp.aurotalk.Model.User;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.chatapp.aurotalk.R;

import java.util.List;


public class UserAdapter extends RecyclerView.Adapter<UserAdapter.ViewHolder> {



    private Context mContext;
    private List<User> mUsers;
    private boolean ischat;

    String theLastMessage;

    public UserAdapter(Context mContext, List<User> mUsers, boolean ischat){
        this.mUsers = mUsers;
        this.mContext = mContext;
        this.ischat = ischat;
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(mContext).inflate(R.layout.user_item, parent, false);
        return new UserAdapter.ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position) {

        final User user = mUsers.get(position);
        holder.username.setText(user.getUsername());
        if (user.getImageURL().equals("default")){
            holder.profile_image.setImageResource(R.mipmap.ic_launcher);
        } else {
            Glide.with(mContext).load(user.getImageURL()).into(holder.profile_image);
        }

        if (ischat){
            lastMessage(user.getId(), holder.last_msg);//displaying last image
        } else {
            holder.last_msg.setVisibility(View.GONE);
        }
    //status update
        if (ischat){
            if (user.getStatus().equals("online")){
                holder.img_on.setVisibility(View.VISIBLE);
                holder.img_off.setVisibility(View.GONE);
            } else {
                holder.img_on.setVisibility(View.GONE);
                holder.img_off.setVisibility(View.VISIBLE);
            }
        } else {
            holder.img_on.setVisibility(View.GONE);
            holder.img_off.setVisibility(View.GONE);
        }

        holder.itemView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(mContext, MessageActivity.class);
                intent.putExtra("userid", user.getId()); //username stores
                mContext.startActivity(intent);
            }
        });
    }

    @Override
    public int getItemCount() {
        return mUsers.size();
    }

    public  class ViewHolder extends RecyclerView.ViewHolder{

        public TextView username;
        public ImageView profile_image;
        private ImageView img_on;
        private ImageView img_off;
        private TextView last_msg;

        public ViewHolder(View itemView) {
            super(itemView);

            username = itemView.findViewById(R.id.username);
            profile_image = itemView.findViewById(R.id.profile_image);
            img_on = itemView.findViewById(R.id.img_on);
            img_off = itemView.findViewById(R.id.img_off);
            last_msg = itemView.findViewById(R.id.last_msg);
        }
    }

    //check for last message
    private void lastMessage(final String userid, final TextView last_msg){
        theLastMessage = "default";
        final    FirebaseUser    firebaseUser    =    FirebaseAuth.getInstance().getCurrentUser();
//initializing firebase user
        DatabaseReference  reference  =  FirebaseDatabase.getInstance().getReference("Chats");
//initializing firebase reference
reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot snapshot : dataSnapshot.getChildren()){
                    Chat chat = snapshot.getValue(Chat.class);
                    if (firebaseUser != null && chat != null) { // checking new msgs matching them
                        if                  (chat.getReceiver().equals(firebaseUser.getUid())                &&
chat.getSender().equals(userid) ||
                                chat.getReceiver().equals(userid)                                        &&
chat.getSender().equals(firebaseUser.getUid())) {
                            theLastMessage = chat.getMessage();
                        }
                    }
                }
//no msg events
                switch (theLastMessage){
                    case  "default":
                        last_msg.setText("No Message");
                        break;

                    default:
                        last_msg.setText(theLastMessage);
                        break;
                }

                theLastMessage = "default";
            }
//error handle
            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });
    }
}
```

Messageactivity.java

```
package com.chatapp.aurotalk;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.chatapp.aurotalk.Adapter.MessageAdapter;
import com.chatapp.aurotalk.Fragments.APIService;
import com.chatapp.aurotalk.Model.Chat;
import com.chatapp.aurotalk.Model.User;
import com.chatapp.aurotalk.Notifications.Client;
import com.chatapp.aurotalk.Notifications.Data;
import com.chatapp.aurotalk.Notifications.MyResponse;
import com.chatapp.aurotalk.Notifications.Sender;
import com.chatapp.aurotalk.Notifications.Token;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;
import com.google.firebase.database.ValueEventListener;
import com.chatapp.aurotalk.R;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

import de.hdodenhof.circleimageview.CircleImageView;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class MessageActivity extends AppCompatActivity {

  CircleImageView profile_image;
```

```
    TextView username;

    FirebaseUser fuser;
    DatabaseReference reference;

    ImageButton btn_send;
    EditText text_send;

    MessageAdapter messageAdapter;
    List<Chat> mchat;

    RecyclerView recyclerView;

    Intent intent;

    ValueEventListener seenListener;

    String userid;

    APIService apiService;

    boolean notify = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_message);

        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        getSupportActionBar().setTitle("");
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        toolbar.setNavigationOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                // and this
                startActivity(new                                    Intent(MessageActivity.this,
  MainActivity.class).setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP));
            }
        });

        apiService = Client.getClient("https://fcm.googleapis.com/").create(APIService.class);

        recyclerView = findViewById(R.id.recycler_view);
        recyclerView.setHasFixedSize(true);
        LinearLayoutManagerlinearLayoutManager=new
  LinearLayoutManager(getApplicationContext());
        linearLayoutManager.setStackFromEnd(true);
        recyclerView.setLayoutManager(linearLayoutManager);
        profile_image = findViewById(R.id.profile_image);
        username = findViewById(R.id.username);
        btn_send = findViewById(R.id.btn_send);
        text_send = findViewById(R.id.text_send);

        intent = getIntent();
        userid = intent.getStringExtra("userid");
        fuser = FirebaseAuth.getInstance().getCurrentUser();

        btn_send.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                notify = true;
                String msg = text_send.getText().toString();
                if (!msg.equals("")){
                    sendMessage(fuser.getUid(), userid, msg);
                } else {
                    Toast.makeText(MessageActivity.this,   "You   can't   send   empty   message",
    Toast.LENGTH_SHORT).show();
     }
                text_send.setText("");
            }
        });


        reference = FirebaseDatabase.getInstance().getReference("Users").child(userid);

        reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                User user = dataSnapshot.getValue(User.class);
                username.setText(user.getUsername());
                if (user.getImageURL().equals("default")){
                    profile_image.setImageResource(R.mipmap.ic_launcher);
                } else {
                    //and this
                    Glide.with(getApplicationContext()).load(user.getImageURL()).into(profile_ima
    ge);
                }

                readMesagges(fuser.getUid(), userid, user.getImageURL());
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });

        seenMessage(userid);
    }

    private void seenMessage(final String userid){
        reference = FirebaseDatabase.getInstance().getReference("Chats");
        seenListener = reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot snapshot : dataSnapshot.getChildren()){
                    Chat chat = snapshot.getValue(Chat.class);
                    if (chat.getReceiver().equals(fuser.getUid()) && chat.getSender().equals(userid)){
                        HashMap<String, Object> hashMap = new HashMap<>();
                        hashMap.put("isseen", true);
                        snapshot.getRef().updateChildren(hashMap);
                    }
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });
    }

    private void sendMessage(String sender, final String receiver, String message){

        DatabaseReference reference = FirebaseDatabase.getInstance().getReference();

        HashMap<String, Object> hashMap = new HashMap<>();
        hashMap.put("sender", sender);
        hashMap.put("receiver", receiver);
        hashMap.put("message", message);
        hashMap.put("isseen", false);

        reference.child("Chats").push().setValue(hashMap);


        // add user to chat fragment
        final             DatabaseReference            chatRef            =
   FirebaseDatabase.getInstance().getReference("Chatlist")
                .child(fuser.getUid())
                .child(userid);

        chatRef.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                if (!dataSnapshot.exists()){
                    chatRef.child("id").setValue(userid);
                }
            }
            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });

        final                    DatabaseReference                    chatRefReceiver                    =
   FirebaseDatabase.getInstance().getReference("Chatlist")
                .child(userid)
                .child(fuser.getUid());
        chatRefReceiver.child("id").setValue(fuser.getUid());

        final String msg = message;

        reference = FirebaseDatabase.getInstance().getReference("Users").child(fuser.getUid());
        reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                User user = dataSnapshot.getValue(User.class);
                if (notify) {
                    sendNotifiaction(receiver, user.getUsername(), msg);
                }
                notify = false;
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });
    }

    private void sendNotifiaction(String receiver, final String username, final String message){
        DatabaseReference tokens = FirebaseDatabase.getInstance().getReference("Tokens");
        Query query = tokens.orderByKey().equalTo(receiver);
        query.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot snapshot : dataSnapshot.getChildren()){
                    Token token = snapshot.getValue(Token.class);
                    Data   data   =   new   Data(fuser.getUid(),   R.mipmap.ic_launcher,   username+":
   "+message, "New Message",
                           userid);

                    Sender sender = new Sender(data, token.getToken());

                    apiService.sendNotification(sender)
                           .enqueue(new Callback<MyResponse>() {
                               @Override
```

```
                              public void
onResponse(Call<MyResponse>call,Response<MyResponse>response) {
                                  if (response.code() == 200){
                                      if (response.body().success != 1){
                                          Toast.makeText(MessageActivity.this,"Failed!",Toast.LENGTH_S
HORT).show();
                                      }
                                  }
                              }

                              @Override
                              public void onFailure(Call<MyResponse> call, Throwable t) {

                              }
                          });
              }
          }

          @Override
          public void onCancelled(@NonNull DatabaseError databaseError) {

          }
      });
   }

   private void readMesagges(final String myid, final String userid, final String imageurl){
       mchat = new ArrayList<>();

       reference = FirebaseDatabase.getInstance().getReference("Chats");
       reference.addValueEventListener(new ValueEventListener() {
           @Override
           public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
               mchat.clear();
               for (DataSnapshot snapshot : dataSnapshot.getChildren()){
                   Chat chat = snapshot.getValue(Chat.class);
                   if (chat.getReceiver().equals(myid) && chat.getSender().equals(userid) ||
                           chat.getReceiver().equals(userid) && chat.getSender().equals(myid)){
                       mchat.add(chat);
                   }

                   messageAdapter = new MessageAdapter(MessageActivity.this, mchat, imageurl);
                   recyclerView.setAdapter(messageAdapter);
               }
           }

           @Override
           public void onCancelled(@NonNull DatabaseError databaseError) {

           }
       });
```

```
    }

    private void currentUser(String userid){
        SharedPreferences.Editoreditor=getSharedPreferences("PREFS",MODE_PRIVATE).
edit();
        editor.putString("currentuser", userid);
        editor.apply();
    }

    private void status(String status){
        reference = FirebaseDatabase.getInstance().getReference("Users").child(fuser.getUid());

        HashMap<String, Object> hashMap = new HashMap<>();
        hashMap.put("status", status);

        reference.updateChildren(hashMap);
    }

    @Override
    protected void onResume() {
        super.onResume();
        status("online");
        currentUser(userid);
    }

    @Override
    protected void onPause() {
        super.onPause();
        reference.removeEventListener(seenListener);
        status("offline");
        currentUser("none");
    }
}
```

TESTING AND IMPLEMENTATION

The testing process focuses on the logical intervals of the software ensuring that all statements have been tested and on functional interval is conducting tests to uncover errors and ensure that defined input will produce actual results that agree with the required results. Program level testing, modules level testing integrated and carried out.

Testing Methods

There are two major type of testing they are :

1. White Box Testing
2. Black Box Testing

1.White Box Testing

White box sometimes called "Glass box testing" is a test case design uses the control structure of the procedural design to drive test case.

2.Black Box Testing

Black box testing focuses on the functional requirements of the software. This is black box testing enables the software engineering to derive a set of input conditions that will fully exercise all functional requirements for a program. Black box testing is not an alternative to white box testing rather it is complementary approach that is likely to uncover a different class of errors that white box methods like.

- Interface errors
- Performance in data structure
- Performance errors
- Initializing and termination errors

1. Unit testing

Unit testing is a software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is the smallest testable part of an application. In procedural programming a unit may be an individual function or procedure. Ideally, each test case is independent from the others: substitutes like method stubs, objects, fakes and test harnesses can be used to assist testing a module in isolation.

2. Integration Testing

This testing is sometimes called Integration and Testing. Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before system testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates and delivers as its output the integrated system ready for system testing.

3. Validation Testing

Validation Testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in a manner that can reasonably expected by a customer. After validation test has been conducted, one of the following two possible conditions exists. The functions or performance characteristics confirm to specification and are accepted.

- In the register and login modules, all the fields must be filled.
- In the email user can login only with their auro mail id
- Password should not be less than six letters or characters

4. Acceptance Testing

User acceptance of a system is a key factor of any system. The system under consideration is tested for the acceptance by constantly keeping in 91 touch with the prospective system users at the same time of developing and marketing changes whenever required. This is done in regard to the following points:

- Input Screen Design
- Output Screen Design

LIMITATIONS

- User cannot send attachments example image, gifs, videos.
- UI is not consistent on all the devices.
- It is not suitable for iOS devices.

FUTURE ENHANCEMENTS

- User should be able to see the date and time when his/her contact was last using the application.
- User should be able to send audio, video and images as attachments. Audio formats that the application should support: mp3. Video formats that the application should support: mp4, gif. Image formats that the application should support: jpg, png.
- In case a user's device crashes, a backup of their chat history must be stored on remote database servers to enable recoverability.
- User should be able to change their password.

CONCLUSION

In the conclusion to the project, following were the learning outcomes gained:

i. Identify a substantial research and/or development project including the contemporary engineering, scientific or research fields.

ii. Combine and demonstrate synthesis of new knowledge with the application of relevant underlying theory, skills, concepts and methodologies.

iii. Analyse, interpret, explain and evaluate results generated during the project.

BIBLIOGRAPHY

- https://www.geeksforgeeks.org/
- https://www.tutorialspoint.com/index.html
- https://getstream.io/tutorials/android-chat
- www.youtube.com
- www.stackoverflow.com