Project Report

for

Succour

Version 1.0 approved

Prepared by - Shachi Maurya 19BCE0419

Harsh Agrawal 19BCE2360

Harshita Suresh 19BDS0133

Jose Sharon M 19BDS0164

VIT Vellore

08 June 2021

Table of Contents

1. Contributors

1.1 **Harsh Agrawal** : Frontend and backend of call and chat and mood tracker and connecting the modules frontend as well as backend.

1.2 **Harshita Suresh** : Frontend of the Seeker page and Profile page and user interface.

1.3 **Jose Sharon M** : Frontend for login and sign up page and user interface.

1.4 **Shachi Maurya** : Frontend of Volunteer page ,mood tracker, backend of authorization,compilation of document and presentation.

2. Title Justification

2.1 Problem Statement

Succour : a depression management and suicide prevention android based application is made to track the mood, activities and thoughts of the users according to the data provided by them and then suggest different activities and tasks to motivate and brighten them up. The users can also contact the volunteers and talk to them through messages and calls.. If the user or volunteer wishes then they can also arrange a face to face meeting.

2.2 Project Motivation

The team came up with this idea of making an application to help those depressed and alone due to which committing suicide is becoming a leading cause of death globally. Traditional methods of capturing social, functional, and behavioural data of victims(or seekers) are limited to the information that patients report back to their health care provider at selected points in time. As a result, these data are not accurate accounts of day-to-day functioning, as they are often influenced by biases in self-report. There has been a rapid growth in the use of new technologies such as mobile health applications (apps) to help identify and support those at risk. We assessed adherence of suicide prevention advice in depression management and suicide prevention app to evidence-based clinical guideline recommendations: mood and suicidal thought tracking, safety plan development, recommendation of activities to deter suicidal thoughts, access to support networks, and access to emergency counselling. It can be used to help as well as to get help with an option to become a volunteer or a seeker. The seeker can get in contact with any volunteer he/she likes and chat or video/voice call the volunteer to get help.

2.3 Glossary of terms

2.3.1 Technical Terms

- **Mood Tracker:** A mood tracker is a tool that is used to keep a record of a person's mood at regular intervals.
- **IDE** : Integrated development environment.
- **VoIP**: Voice over Internet protocol used for calling and video calling.
- **XMPP** :Extensible Messaging and Presence Protocol used for messaging /chatting .
- **HTTP**: Hypertext Transfer Protocol used for transferring data file systems and multimedia communication.
- **Flutter SDK:** It is a Software Development Kit with prewritten code, consisting of ready-to-use and customizable widgets, as well as libraries, tools, and documentation that together serve to build cross-platform apps.
- **Firebase:** It is a Backend-as-a-Service (Baas) categorized as a NoSQL database program, which stores data in JSON-like documents.

2.3.2 Non-Technical Terms

- **Seeker:** A user who is depressed and alone and in need of help
- **Volunteer:** A user who wants to help a seeker by chat and calls
- **Chat:** refer to any kind of communication over the Internet that offers a real-time transmission of text messages from sender to receiver

2.4 Functional Requirement Specification

2.4.1 Stakeholders
The stakeholders are the relevant
- Vulnerable groups
- Developers
- People from the health organizations/helpers

2.4.2 Actors and Goals
The list of actors and their goals is given below:
- Volunteer with the goal to help the ones in need (seekers)via online sessions through call, chat or through meeting face to face.
- Seeker with the goal to get help whenever needed from Volunteers via online sessions through call, chat or through meeting face to face and track their daily moods .
- Developers with the goal to help Seeker and Volunteer connect to each other through the android application .

.

2.4.3 Intimating Actors
The Developers act as Intimidating actors as they are the ones proving the participating actors with the android application

2.4.4 Participating Actors
Seeker and Volunteer are the participating actors for the android application.

2. Title Justification

2.1. Software Model
2.1.1. Model used in project
The model used for creation of this Android application is “Spiral Model”.

3.1.2 Model Justification
The timeline for the Project was around 3 months and since all the team members were new to Application Development , we had to go back and forth the modules many times and do changes in the design and requirement fields which are easier with the Spiral Model. Also some other reasons why we chose Spiral Model are:
- Changing requirements can be accommodated easily.
- It allows extensive use of prototypes.
- Requirements can be captured more accurately.
- Users can see the system early.
- Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.
- It is useful when we have budget constraints and risk evaluation.

3.2 Requirements
3.2.1Functional Requirements for each System Feature
3.2.1.1 Login /Signup Page
3.2.1.1.1 Description and Priority
The login page consists of the option whether the user is a volunteer or the victim.According to the type of user the login page will ask for the user details.
Priority=9

3.2.1.1.2 Stimulus/Response Sequences

STIMULUS: The user clicks on the Victim login :

RESPONSE:

1) A new page containing the data to be inputted by the user appears . If the user already has an account they can simply click on login after giving their account information .

2.) Else they fill the data and click on the Sign up option which creates a new account for the user.The user is then taken to the next System feature.

STIMULUS: The user clicks on the Volunteer login
RESPONSE:

1.) A new page containing the data to be inputted by the user appears.If the user already has an account they can simply click on login after giving their account information .

2.) Else they fill the data and click on the Sign up option which creates a new account for the user.The user is then taken to the next System feature.

3.2.1.1.3 Functional Requirements

REQ-1: The User shall be able to click on buttons to navigate to the type of user Login /Signup.
i) If the user clicks on the Volunteer button they are navigated to the Volunteer Login /Signup page.
ii.) If the user clicks on the Victim button , they are navigated to the Victim Login/Signup page.

REQ-2: The User shall be able to give the User details for LOGIN and validate the user.
i.) For Volunteer get the following data for Login:
1.) Email
2.) Password
If the email or password is wrong or is not present in the database as an existing user, then throw an error and ask the User to either Sign Up or check the email /password.
ii.)For Victim get the following data for Login:
1.) Email
2.) Password
If the email or password is wrong or is not present in the database as an existing user, then throw an error and ask the User to either Sign Up or check the email /password.

REQ-3: The User shall be able to give the User details for SIGNUP and the database shall be able to get new user details.
i.) For Volunteer get the following data for Signup:
1.)Name
2.)Date Of Birth
3.) Email
4.)Mobile Number
5.)Password
6.)Medical Practitioner (yes/no) if yes then ask for CRR no.
If the Email/Mobile number is already registered then throw an error stating that the Email/Mobile Number already exists.
ii.) For Victim get the following data for Signup:
1.)Name

2.)Date Of Birth
3.) Email
4.)Mobile Number
5.)Password

If the Email/Mobile number is already registered then throw an error stating that the Email/Mobile Number already exists.

3.2.1.2 Volunteer Mode

3.2.1.2.1 Session Organization

3.2.1.2.1.1 Description and Priority

This feature helps to keep a track of the activity of the Volunteer in terms of the therapy sessions with their patients.It includes by chat ,on voice call, on video call and face to face meet up sessions with the victims.
Priority=9

3.2.1.2.1.2 Stimulus/Response Sequences

Since there are different ways to do a session there will be different stimulus/response sequences for each type of session:

1. Through Message(Chat):

   STIMULUS: User clicks on chat activity .

   RESPONSE: Chat Room is displayed.

   STIMULUS: User posts messages.

   RESPONSE: Message is displayed.

2.)Through Voice/Video Call:

STIMULUS:The user clicks on the dial

RESPONSE: The ring sound produced until call picked up by victim side

STIMULUS: User clicks disconnect

RESPONSE: The call ends

3.)Through Face-To-Face Meet:

STIMULUS: The user clicks on share Location

RESPONSE: The location of user shall be shared with the victim

4.)View Session history:

STIMULUS: The user clicks on view history

RESPONSE: The history of past sessions shall be displayed.

3.2.1.2.1.3 Functional Requirements

REQ-1: The user shall be able to view the chat room and post messages.
REQ-2: The user shall be able to connect and disconnect voice calls and be able to communicate to the other side.
REQ-3: The user shall be able to connect and disconnect the video call and be able to communicate to the other side.

REQ- 4:The user shall be able to share as well as get the location of the victim for face to face meetings .
REQ-5:The history of the sessions shall be saved in the database.

3.2.1.2.2 Progress of patients

3.2.1.2.2.1 Description and Priority

The user can see the analysis of moods of the victim as per victims mood tracker.
Priority=7.

3.2.1.2.2.2 Stimulus/Response Sequences

STIMULUS: The user clicks on progress.
RESPONSE: A chart of the victims (under the treatment of the user) appears.

3.2.1.2.2.3 Functional Requirements

REQ-1: The user shall be able to click on the button to see progress of the victim.
REQ-2: The user shall be able to see the chart of the victim's mood .

3.2.1.3Victim Mode

3.2.1.3.1 Mood Tracker

3.2.1.3.1 Description and Priority

The user can give data about his/her daily mood .
Priority=8

3.2.1.3.2 Stimulus/Response Sequence

SIMULUS: The user clicks on the mood from the given list
RESPONSE: The mood for the day is saved in database

3.2.1.3.3 Functional Requirements

REQ-1: The user shall be able to see and select the mood from the list of moods
REQ-2: The data for mood shall be stored in the database.

3.2.1.3.2 Session Organization

3.2.1.3.2.1 Description and Priority

This feature helps to keep a track of the activity of the user in terms of the therapy sessions with their Volunteers .It includes by chat ,on voice call, on video call and face to face meet up sessions with the Volunteers.
Priority=9

3.2.1.3.2.2 Stimulus/Response Sequences

Since there are different ways to do a session there will be different stimulus/response sequences for each type of session:

1.)Through Message(Chat):

STIMULUS: User clicks on chat activity .

RESPONSE: Chat Room is displayed.

STIMULUS: User posts messages.

RESPONSE: Message is displayed.

2.)Through Voice/Video Call:

STIMULUS:The user clicks on the dial

RESPONSE: The ring sound produced until call picked up by Volunteer side

STIMULUS: User clicks disconnect

RESPONSE: The call ends

3.)Through Face-To-Face Meet:

STIMULUS: The user clicks on share Location

RESPONSE: The location of user shall be shared with the Volunteer

4.)View Session history:

STIMULUS: The user clicks on view history

RESPONSE: The history of past sessions shall be displayed.

3.2.1.3.2.3 Functional Requirements

REQ-1: The user shall be able to view the chat room and post messages.
REQ-2: The user shall be able to connect and disconnect voice calls and be able to communicate to the other side.
REQ-3: The user shall be able to connect and disconnect the video call and be able to communicate to the other side.
REQ- 4:The user shall be able to share as well as get the location of the volunteer for face to face meetings .
REQ-5:The history of the sessions shall be saved in the database.

3.2.1.3.3 Activity / Safety Plan Deployment

3.2.1.3.3.1 Description and priority

Activities(act as Safety Plan Deployment) can be assigned to the user for mood improvement .

Priority=8

3.2.1.3.3.2 Stimulus/Response Sequences

STIMULUS: The user clicks on activity assign
RESPONSE: The activity is assigned.
STIMULUS: The user clicks on activity completed
RESPONSE: The activity assigned is completed

3.2.1.3.3.3 Functional Requirements

REQ-1: The user shall be assigned an activity to lift the mood
REQ-2: The user shall be able to click on completion.

3.2.2 Non- Functional Requirements

3.2.2.1 Performance Requirements

PE-1: The action taken by the users should be performed and loaded on the screen in next 10 milliseconds.

PE-2: The system should display confirmation messages to the user with in 12 milliseconds after the user submits the data.

3.2.2.2 Safety Requirements

The victim and the volunteer are connected to the same server. Making sure that the application is working in harmony with the GPS of the device.

3.2.2.3 Security Requirements

SE-1: The data submitted by the user should not be accessible to non authorised personnel.

SE-2: The messages and the call records should not be available to a third person.

3.2.2.4 Software Quality Attributes

Availability-1 The system shall be available to users all the time.

Availability -2 The system shall always have something to function and always pop up error messages in case of component failure.

Efficiency-1: The system shall provide the right tools to support all its features.

3.2.2.5 Business Rules

BR-1: The Victim can use mood tracker to record the daily mood while the Volunteer can only see the analysis off the tracker for those Victims under his/her treatment.

BR-2: The location sharing from Victim side should be done every time so as to help them in emergency scenario

3.2.3 User Stories

As a Seeker ,I want to be able to track my daily mood, analyse it and be able to talk to a Volunteer via call/ chat ,whenever I feel alone and depressed. If possible I should also be able to meet the Volunteer by sharing location or via video call.

As a Volunteer ,I should be able to see the Seekers seeking my help to overcome depression and loneliness and be able to be in contact with them via call/chat. If possible I would also want to see the Seeker and meet Seeker personally if required. I want the records of the sessions of each Seeker to be stored so I can refer to it whenever possible and also see the Mood analysis of each Seeker

3.2.4 Customer Statement of Requirements

As a customer I need:

- Android Application that can track my mood and give analysis of my daily mood in chart format.
- Users to be able to make call and message other users.
- Maintain records of the sessions between users and share location.

3.2.5 Network Protocols/ Hardware Requirements

3.2.5.1 Network Protocols

- VoIP protocols for calls
- XMPP (Extensible Messaging and Presence Protocol) for chat
- HTTP for mood tracker.

3.2.5.2 Hardware Requirements

- Android smartphone with API level greater than 20(Android OS 5.0 and newer).

3.2.6 External Interface Requirements

- Email id for Sign Up and Login
- Internet connection
- Mobile Phone Charger

3.2.7 Database

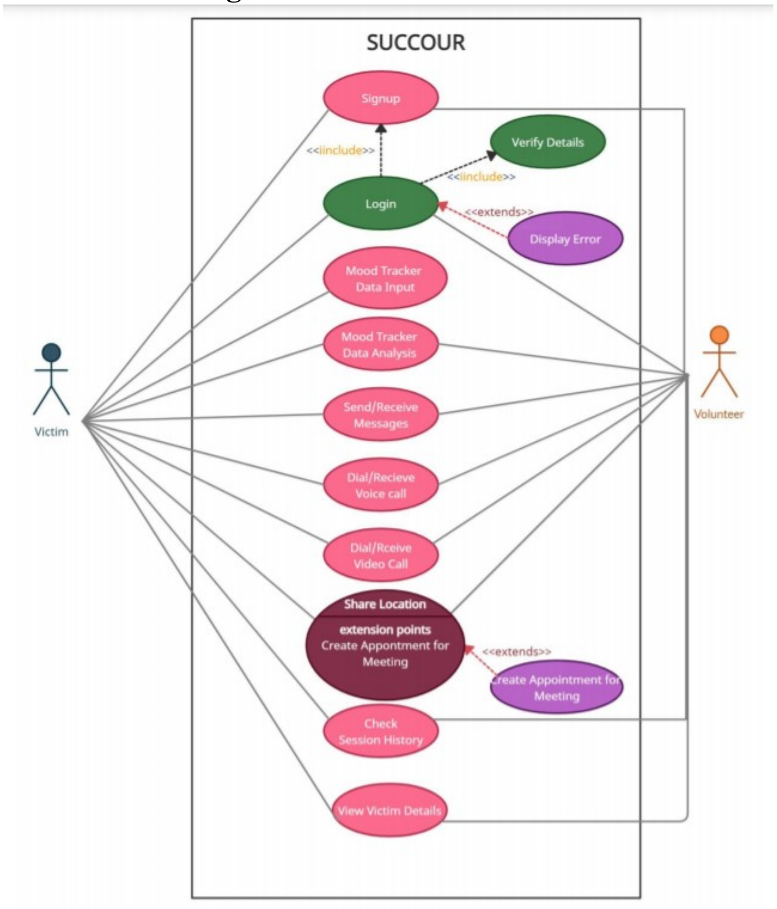
- Firebase for authentication
- SQLite for mood tracker

3.2.8 Objective

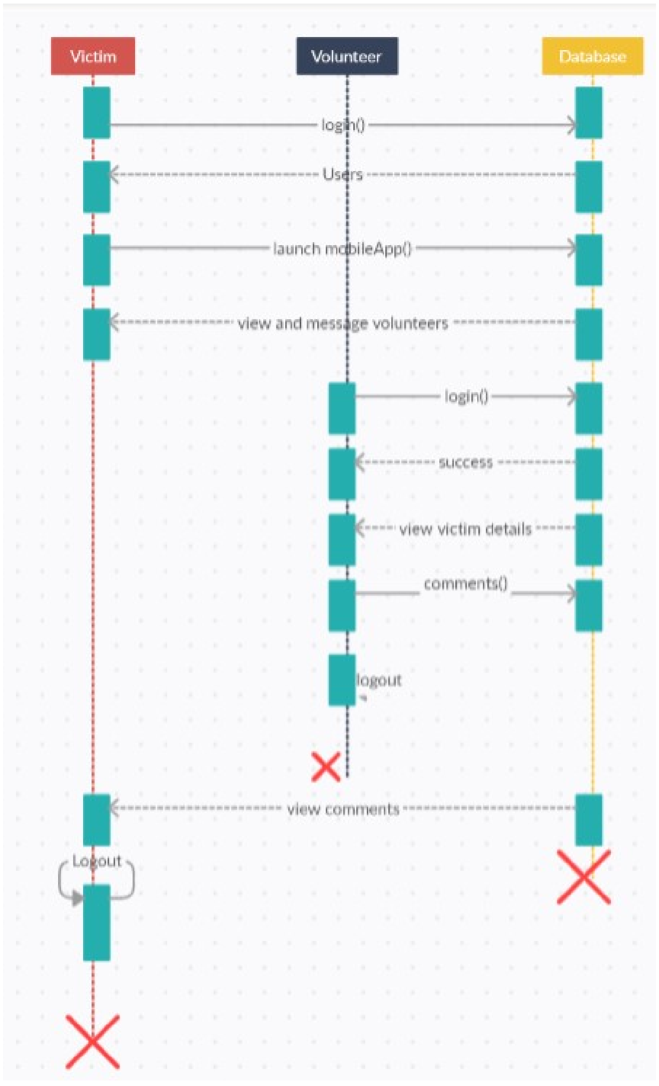
The objective of this Android Application:Succour is to help the vulnerable section of Society who are depressed and alone by creating a platform for them to interact with other people who are either from a Support Group ,NGO or a Medical Practitioner in Psychology so that their negative thoughts and behaviour changes to positive. Also allowing the Seekers to be able to keep a record of their Moods and activities which lead to that mood so they can refrain from doing anything that makes them unhappy.

4. UML diagram

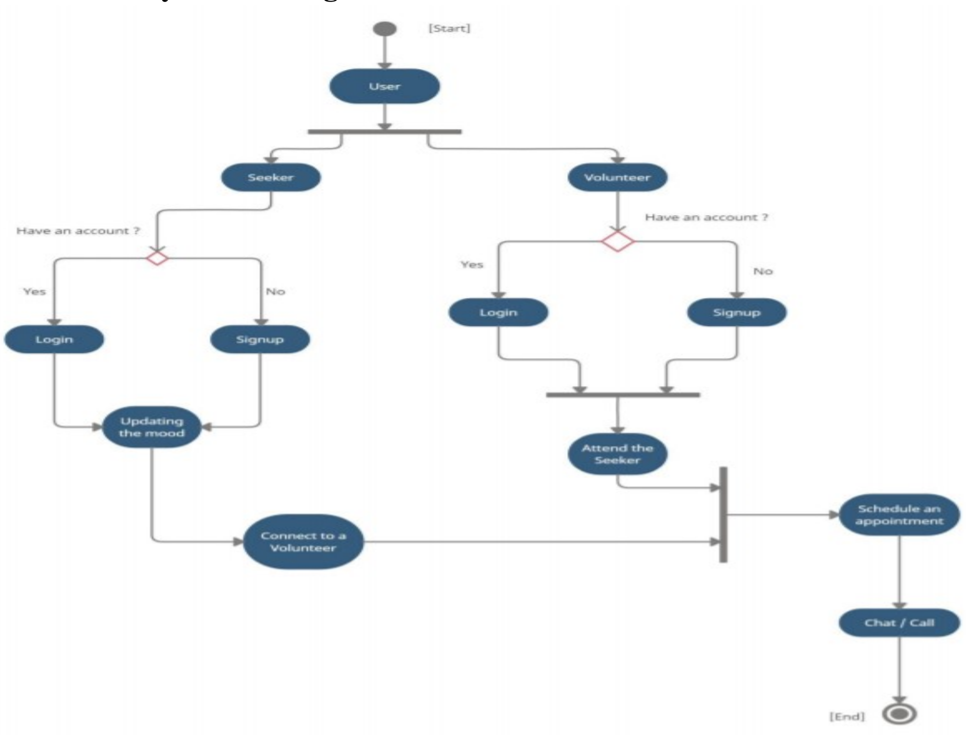
4.1 User case diagram



4.2 Sequence diagram

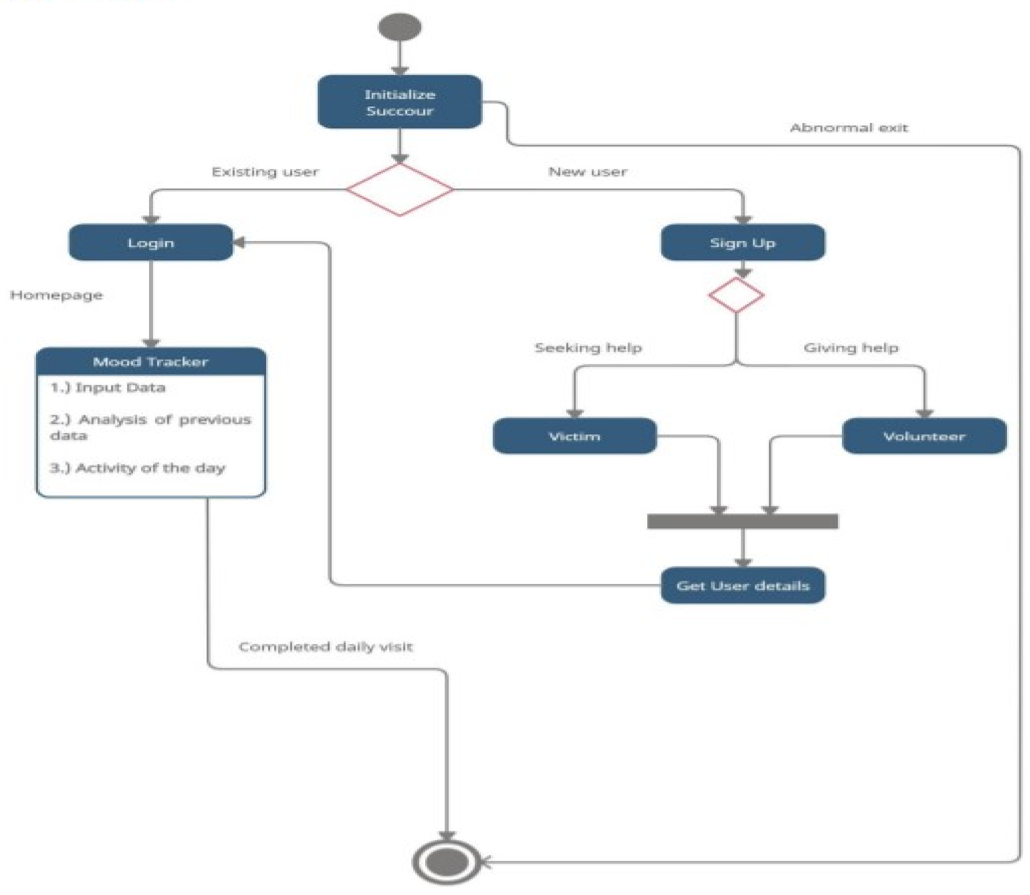


4.3 Activity Chart Diagram

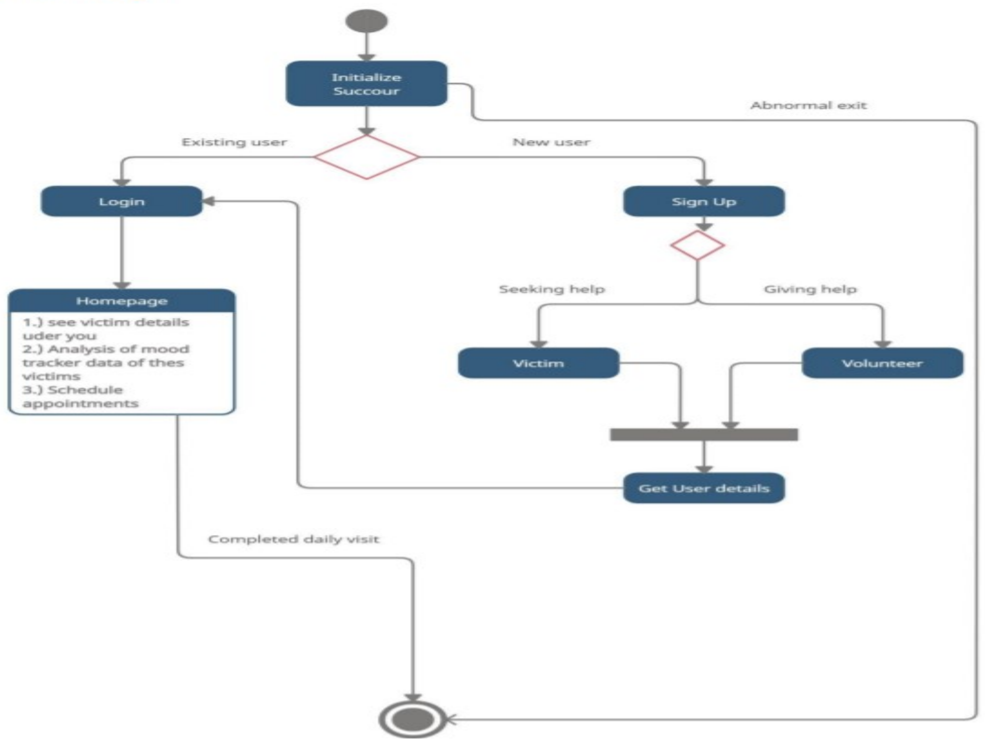


4.4 State Chart Diagram

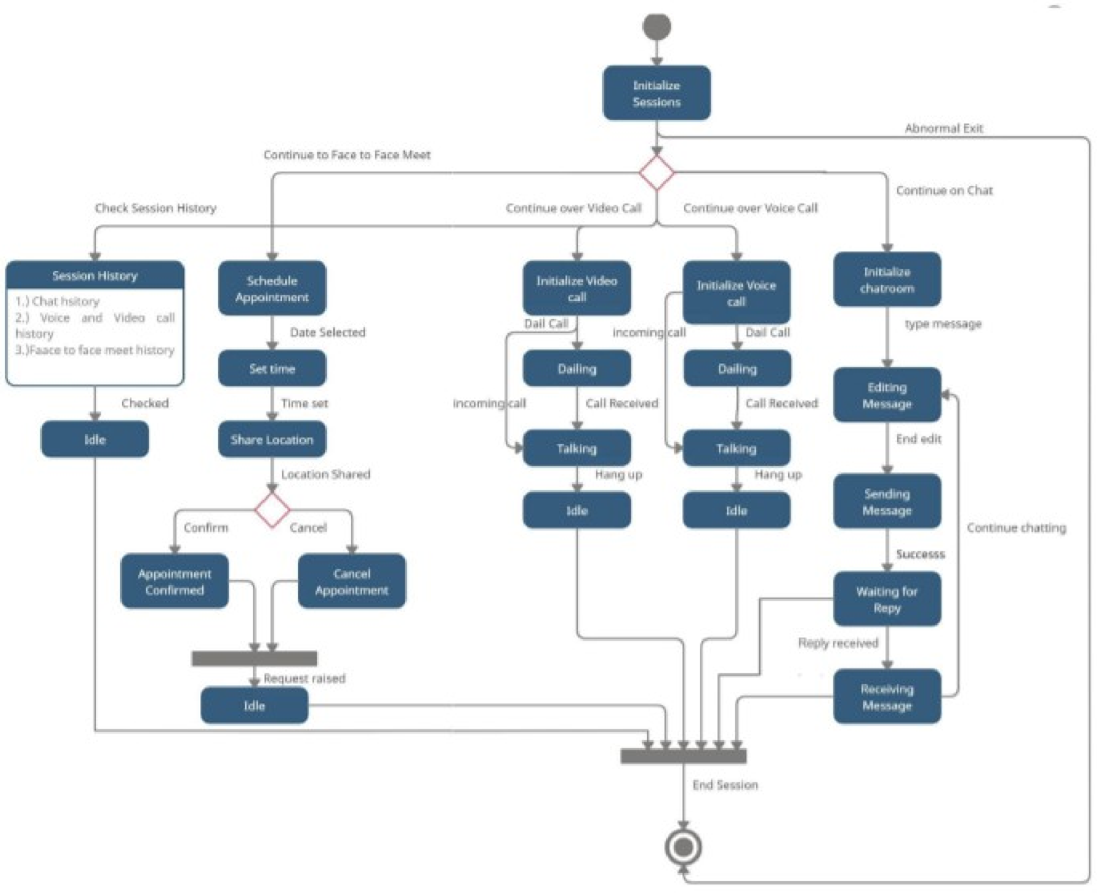
Victim Login



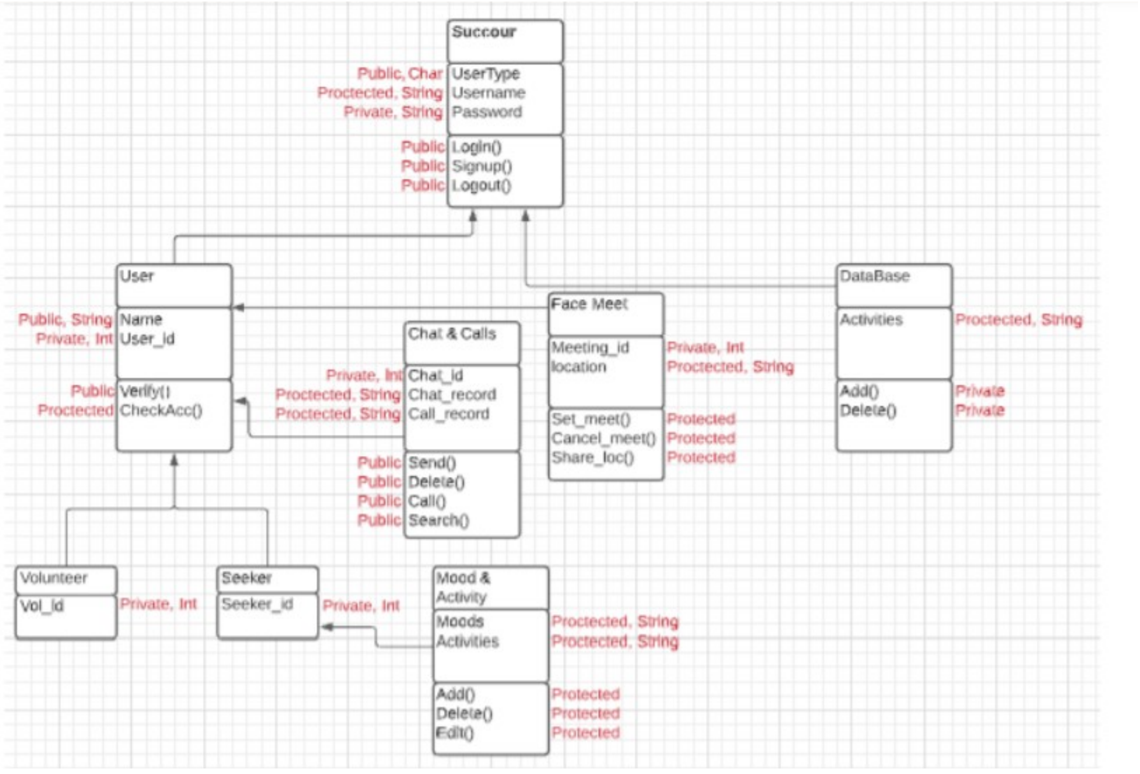
Volunteer Login



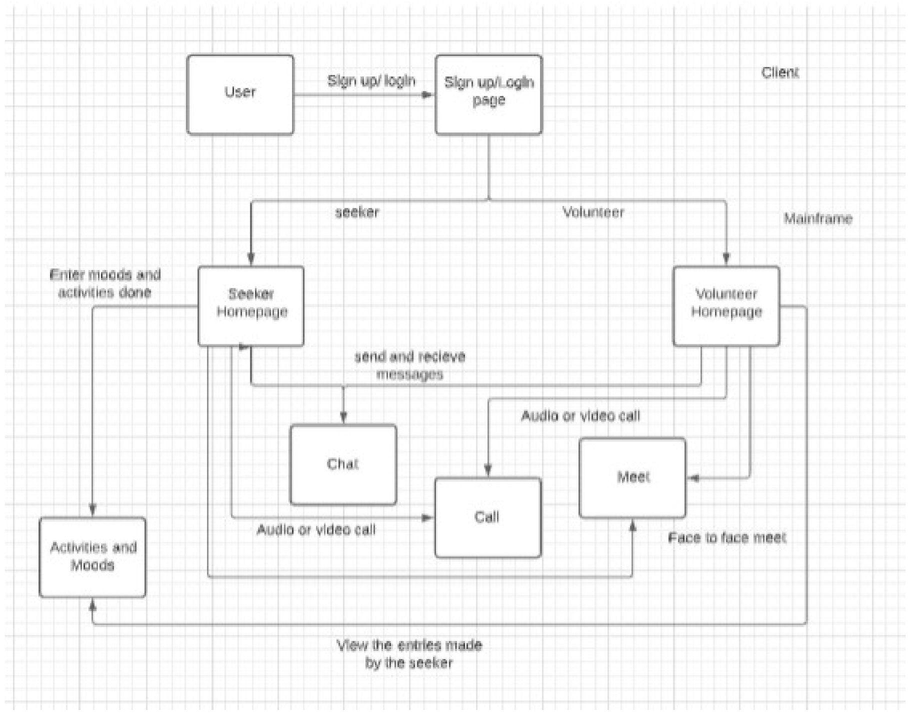
Session Management



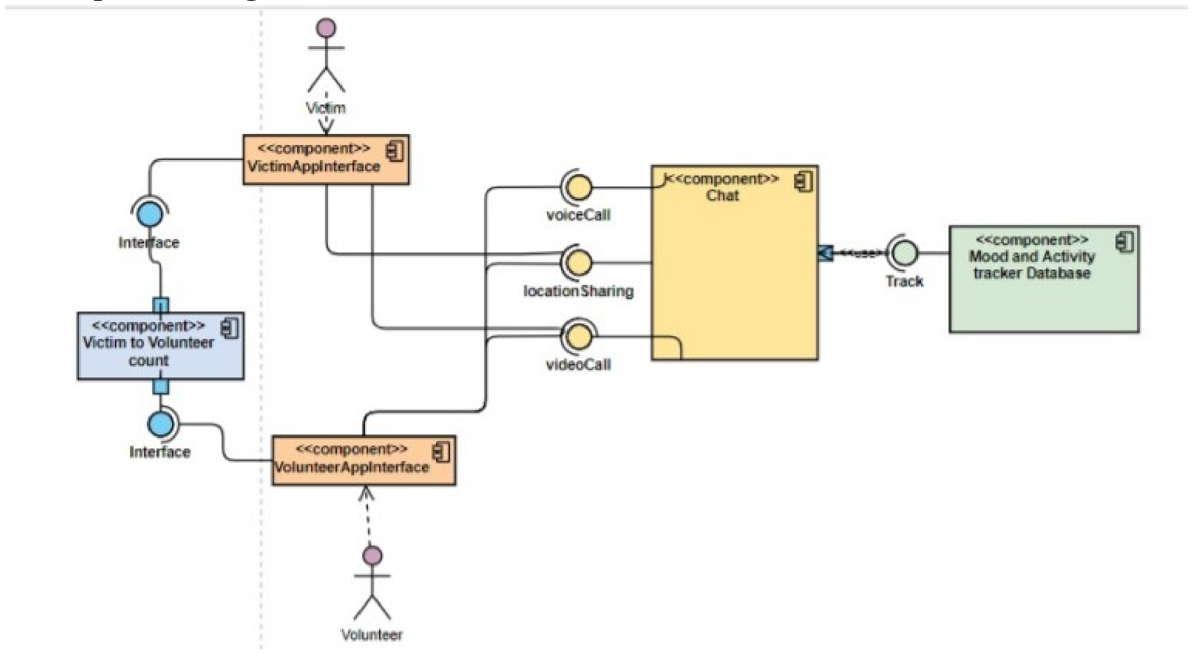
4.5 Class Diagram



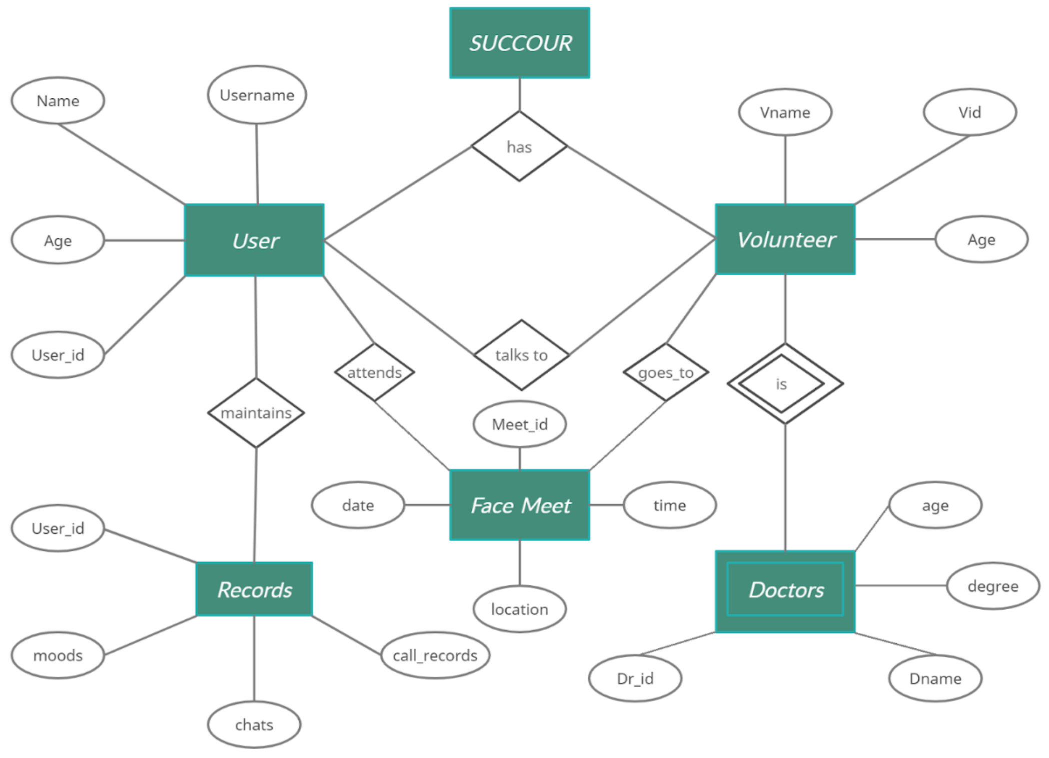
4.6 Collaboration Diagram



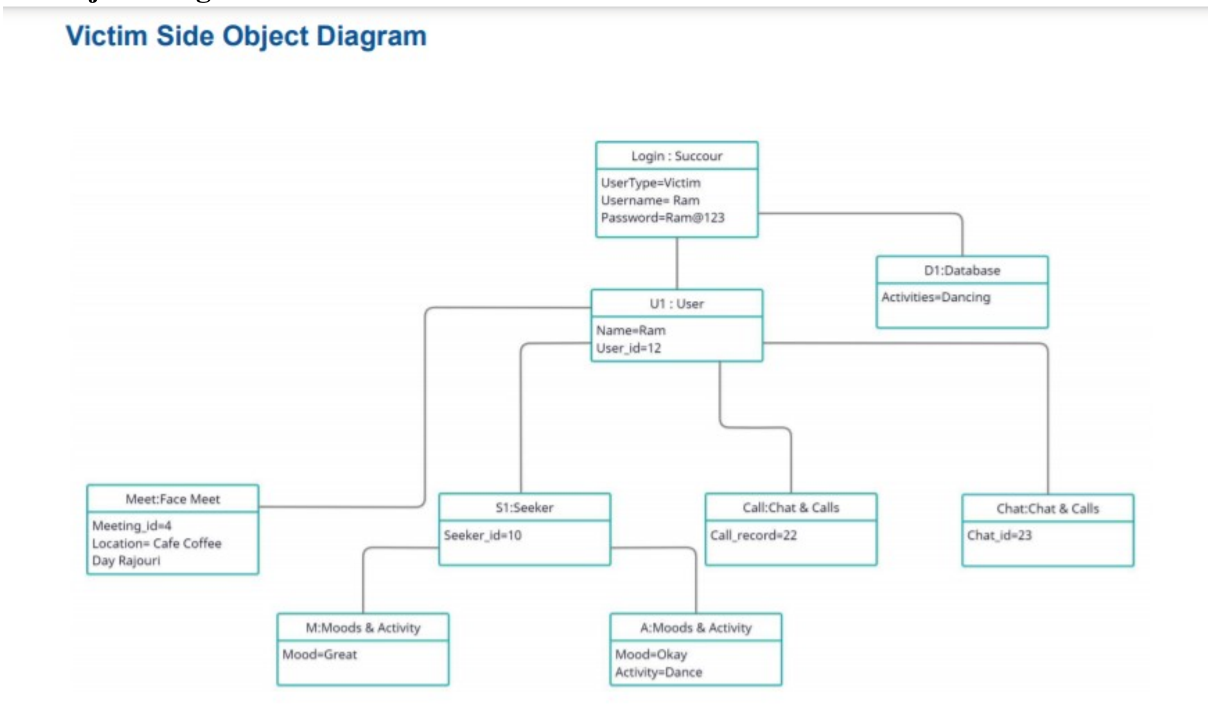
4.7 Component Diagram


Victim
<<component>> VictimAppInterface
Interface
<<component>> Victim to Volunteer count
Interface
<<component>> VolunteerAppInterface
Volunteer
voiceCall
locationSharing
videoCall
<<component>> Chat
<<use>>
Track
<<component>> Mood and Activity tracker Database

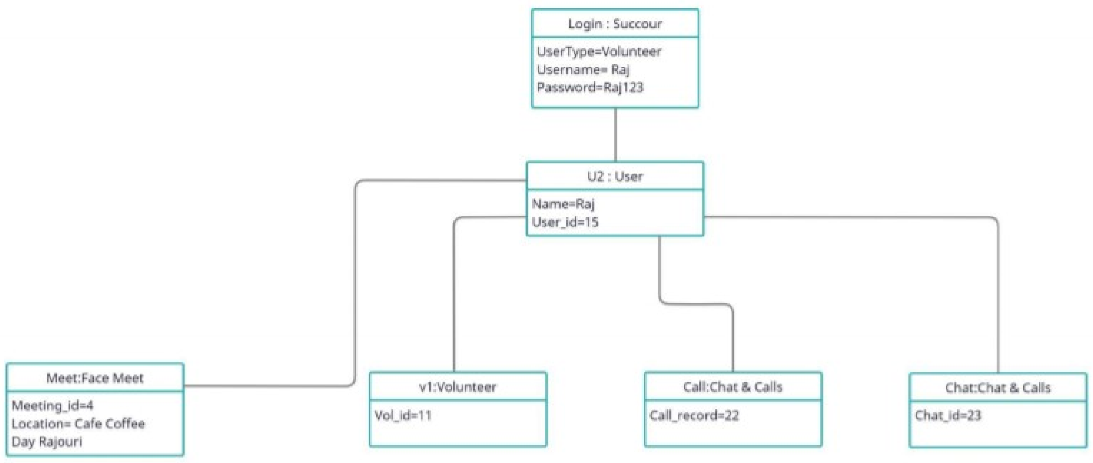
4.8 ER- Diagram



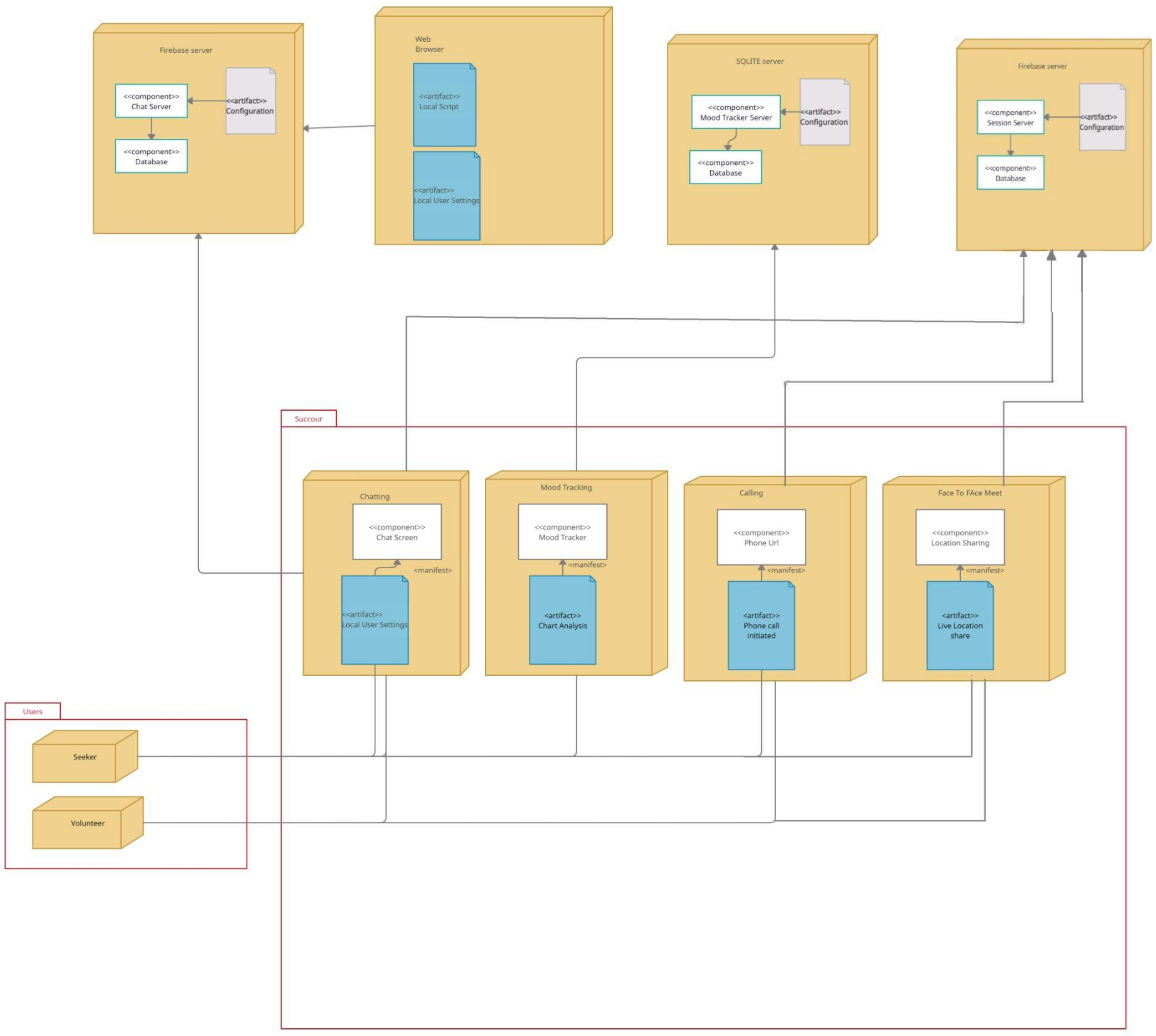
4.9 Object Diagram



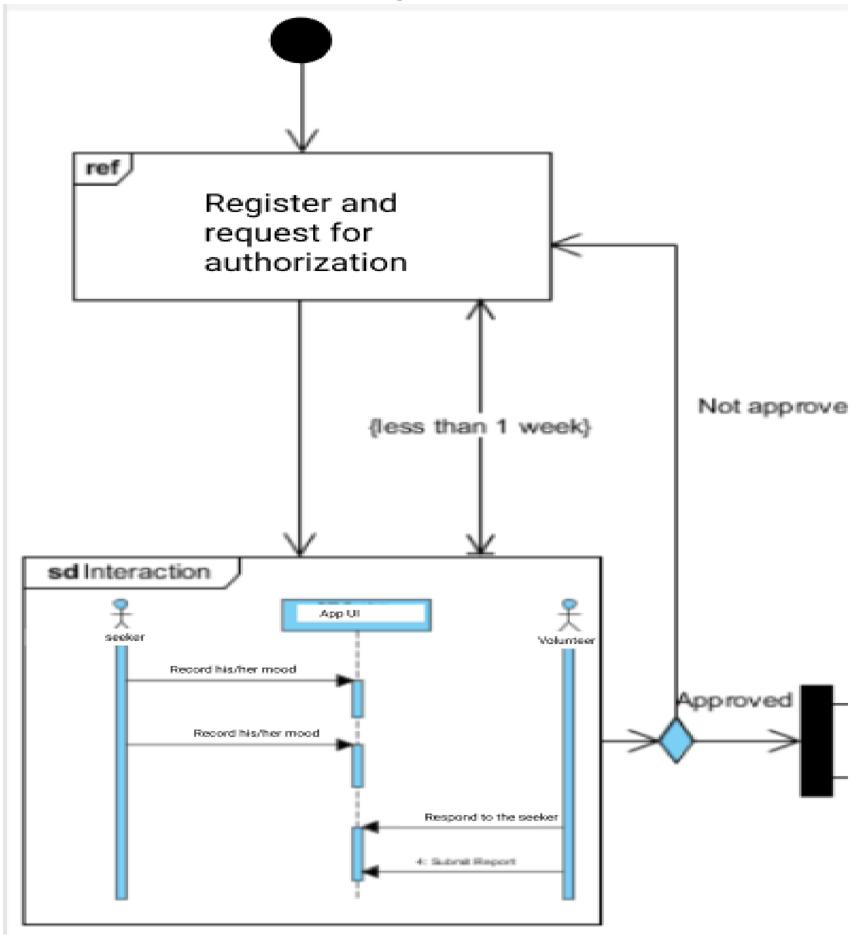
Volunteer Side Object Diagram



4.10 Deployment Diagram

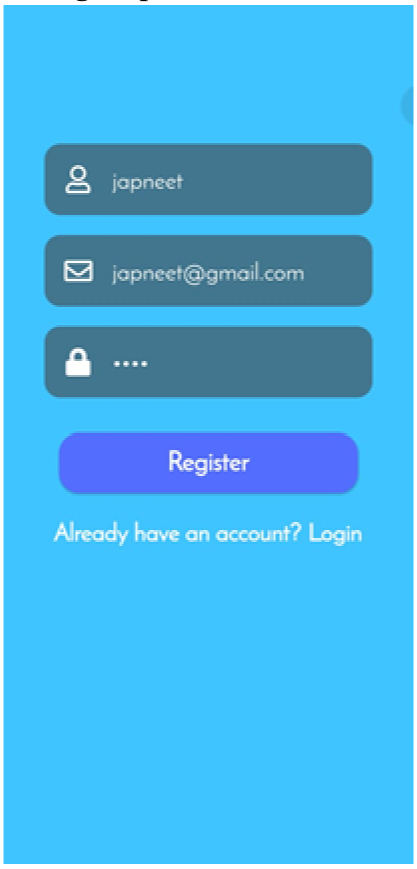

Firebase server
<<component>> Chat Server
<<artifact>> Configuration
<<component>> Database
Web Browser
<<artifact>> Local Script
<<artifact>> Local User Settings
SQLITE server
<<component>> Mood Tracker Server
<<artifact>> Configuration
<<component>> Database
Firebase server
<<component>> Session Server
<<artifact>> Configuration
<<component>> Database
Succour
Chatting
<<component>> Chat Screen
<manifest>
<<artifact>> Local User Settings
Mood Tracking
<<component>> Mood Tracker
<manifest>
<<artifact>> Chart Analysis
Calling
<<component>> Phone Url
<manifest>
<<artifact>> Phone call initiated
Face To FAce Meet
<<component>> Location Sharing
<manifest>
<<artifact>> Live Location share
Users
Seeker
Volunteer

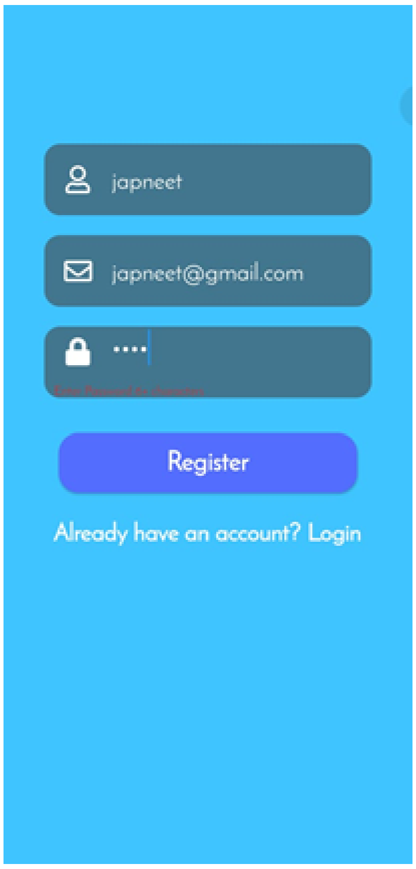
4.11 Interactive overview diagram

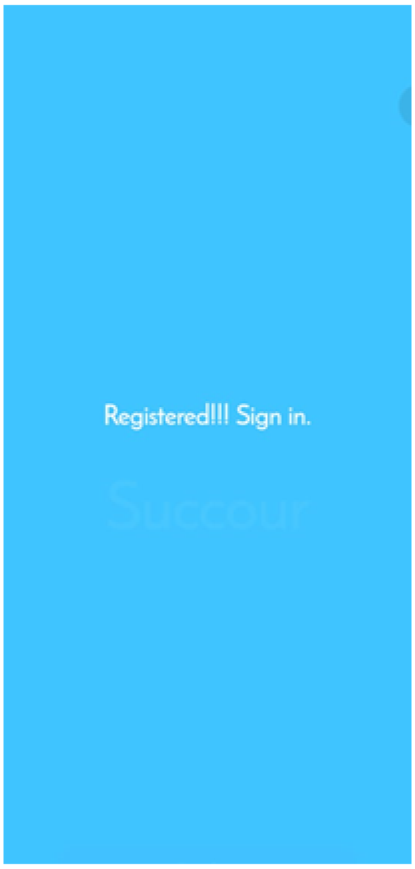


5. Test Cases

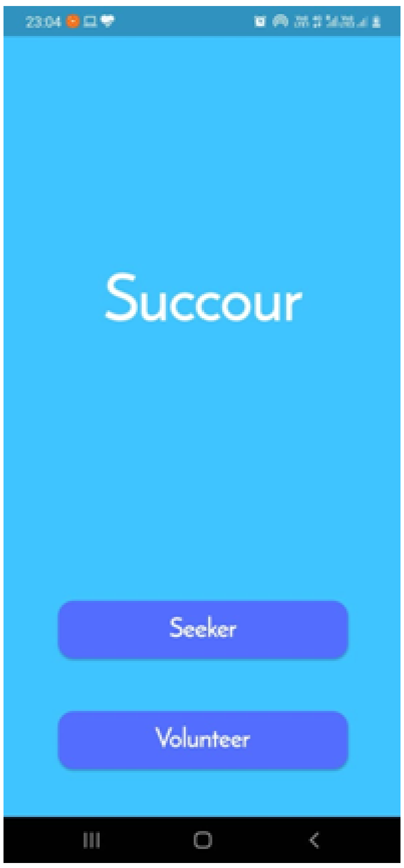
5.1 Sign Up

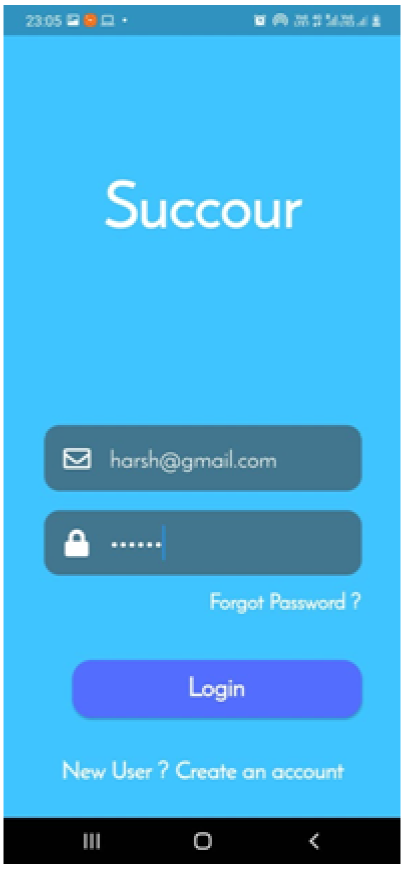

japneet
japneet@gmail.com
····

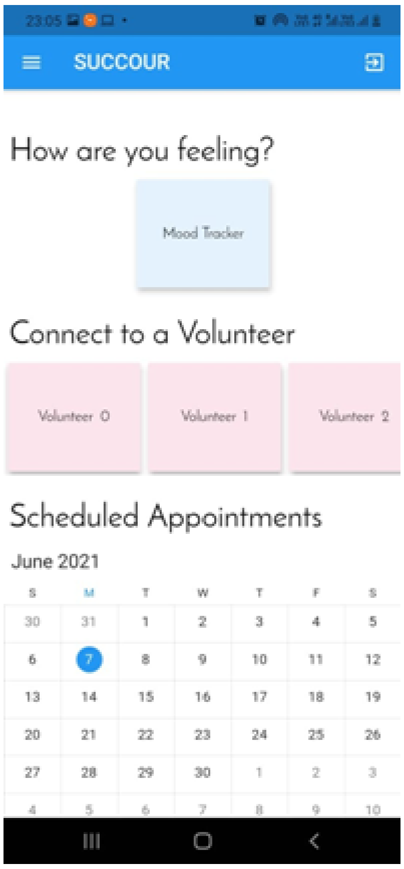

japneet
japneet@gmail.com
····
Enter Password 6+ characters


Registered!!! Sign in.

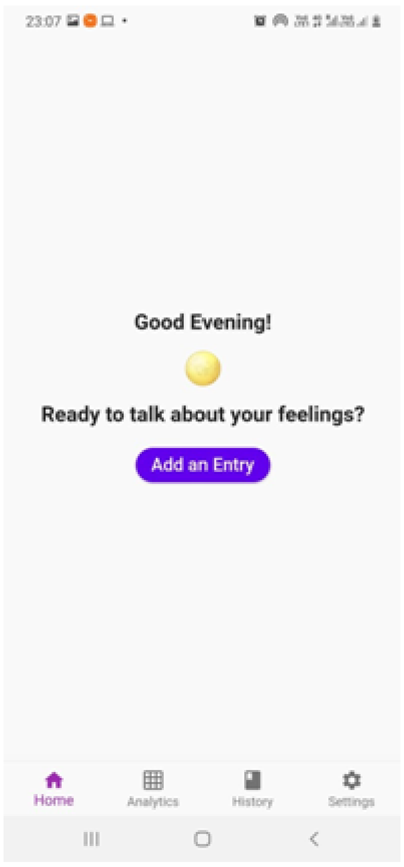
5.2 Login for seeker



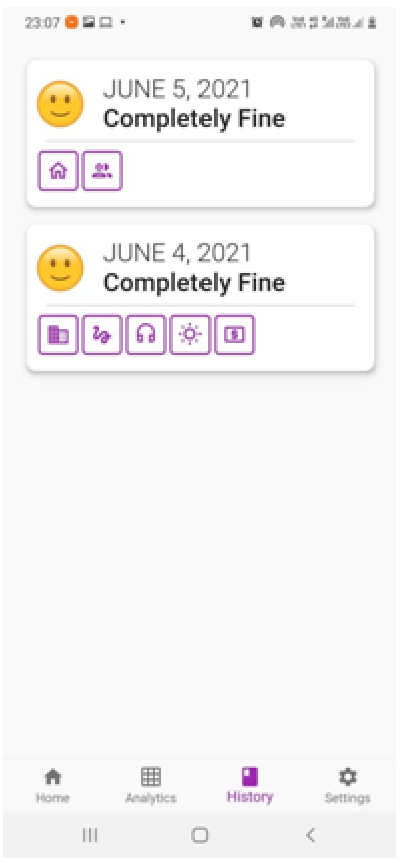




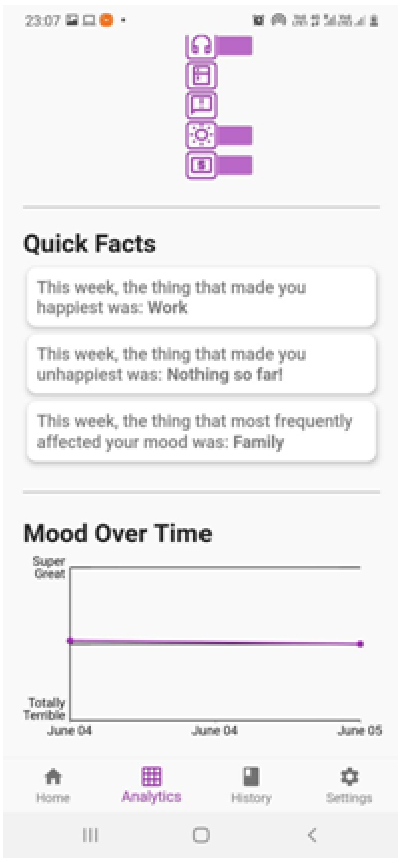
5.3 Mood tracker


Good Evening!

Ready to talk about your feelings?


JUNE 5, 2021
Completely Fine

JUNE 4, 2021
Completely Fine

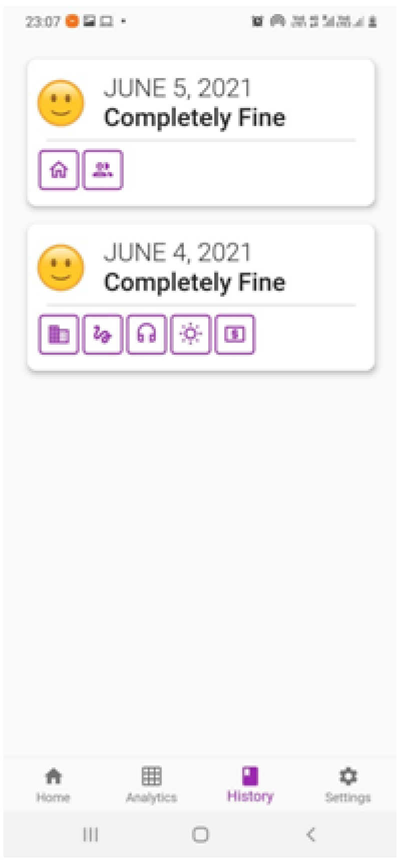

Quick Facts

This week, the thing that made you happiest was: Work

This week, the thing that made you unhappiest was: Nothing so far!

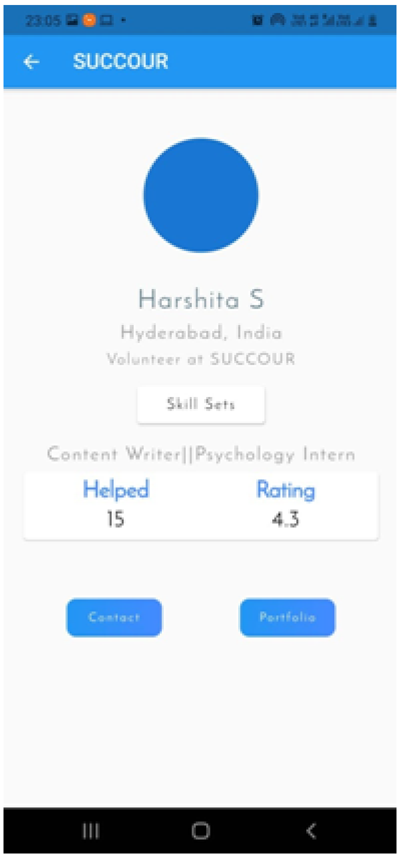
This week, the thing that most frequently affected your mood was: Family

Mood Over Time


JUNE 5, 2021
Completely Fine

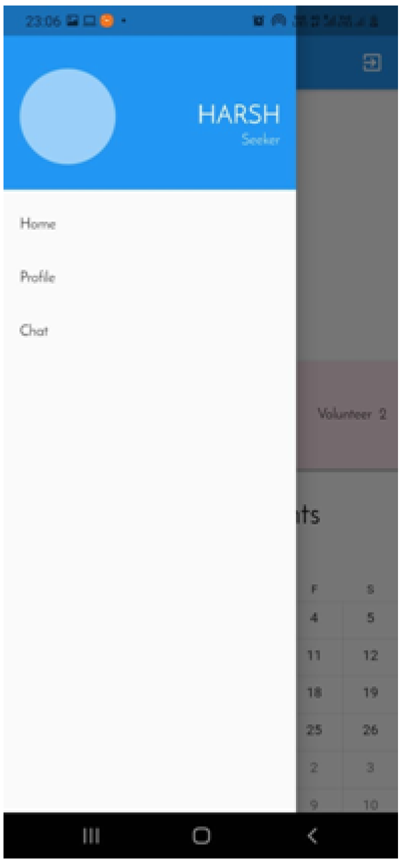
JUNE 4, 2021
Completely Fine

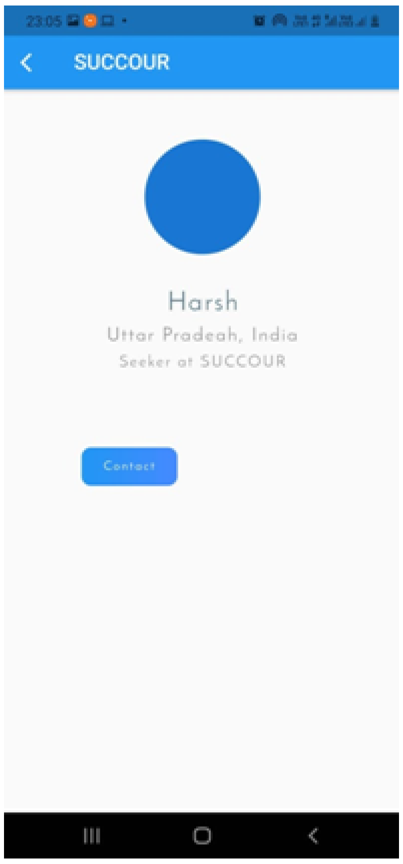
5.4 Seeker Homescreen and navigation



Harshita S
Hyderabad, India
Volunteer at SUCCOUR
Skill Sets
Content Writer||Psychology Intern

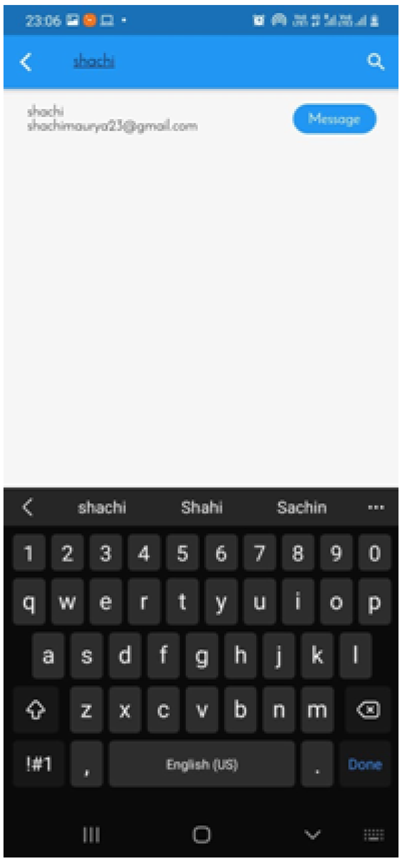
Helped	Rating
15	4.3



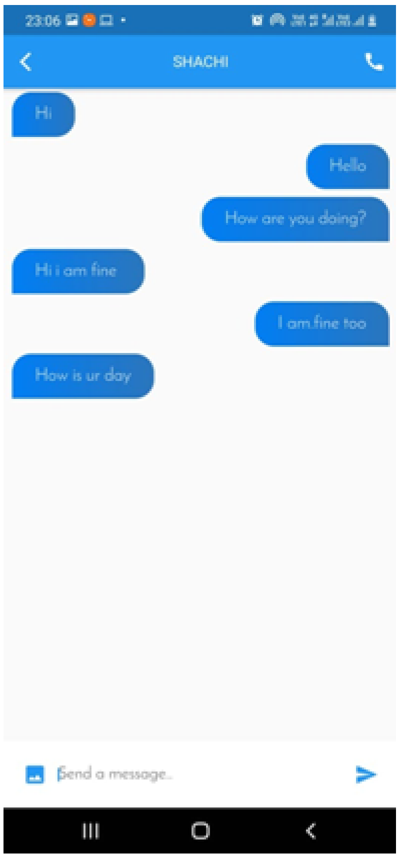


Harsh
Uttar Pradeah, India
Seeker at SUCCOUR

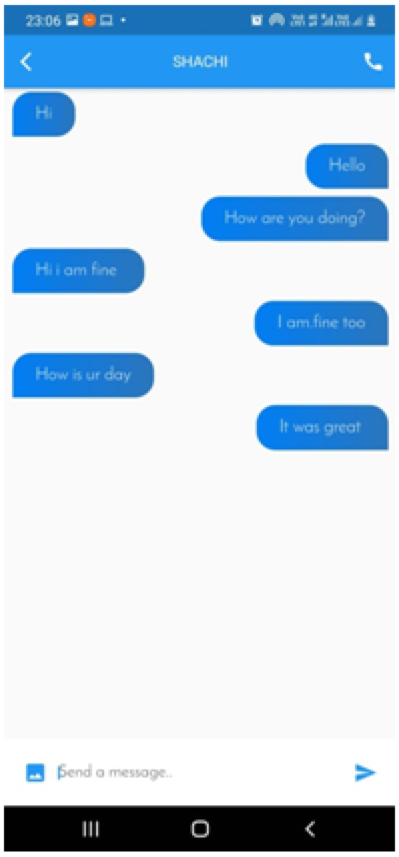
5.5 Chat Screening



shachi
shachimaurya23@gmail.com

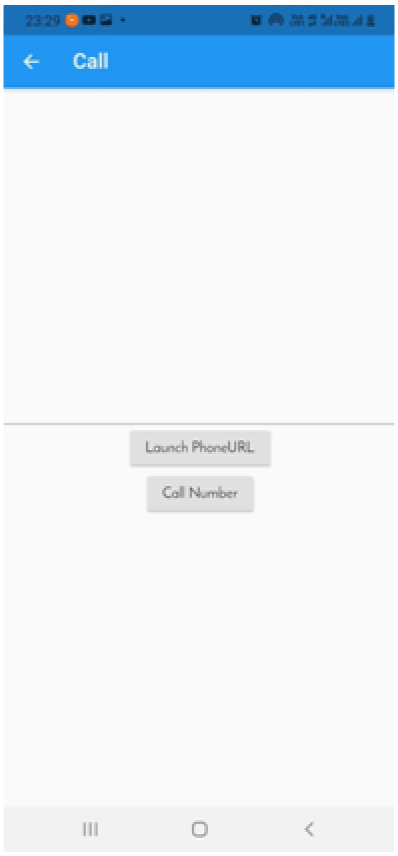


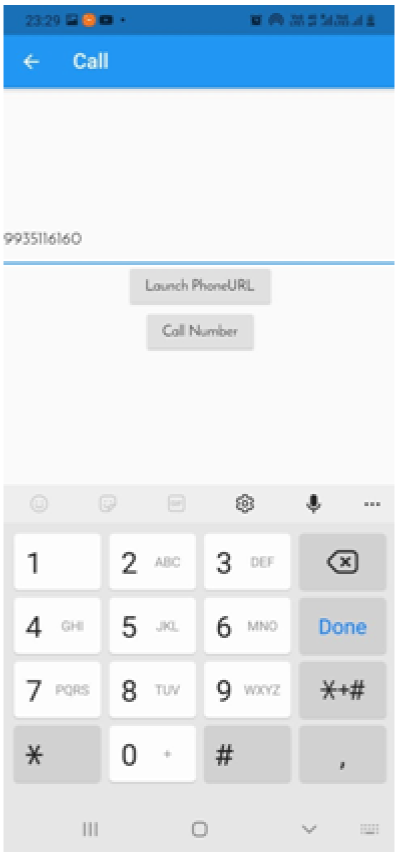
Hi
Hello
How are you doing?
Hi i am fine
I am fine too
How is ur day

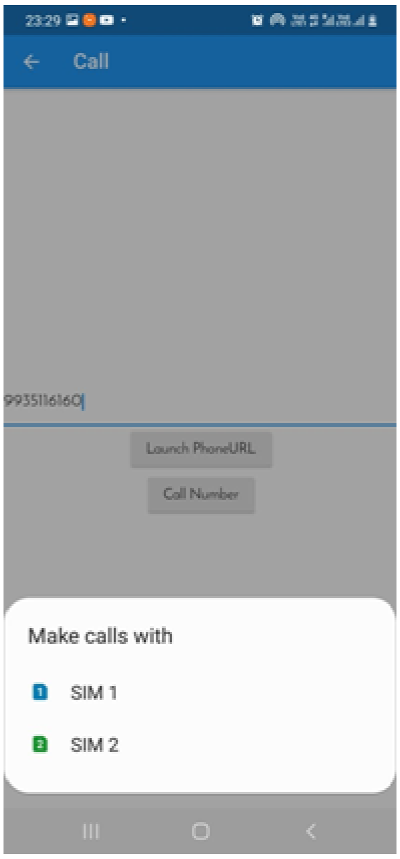


Hi
Hello
How are you doing?
Hi i am fine
I am fine too
How is ur day
It was great

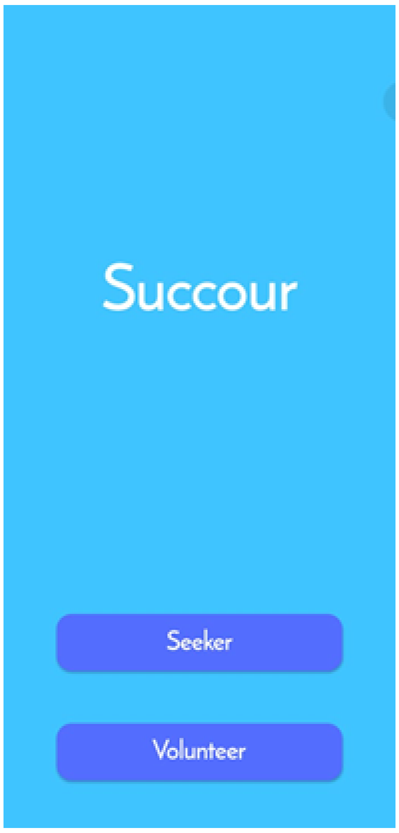
5.6 Call Screen

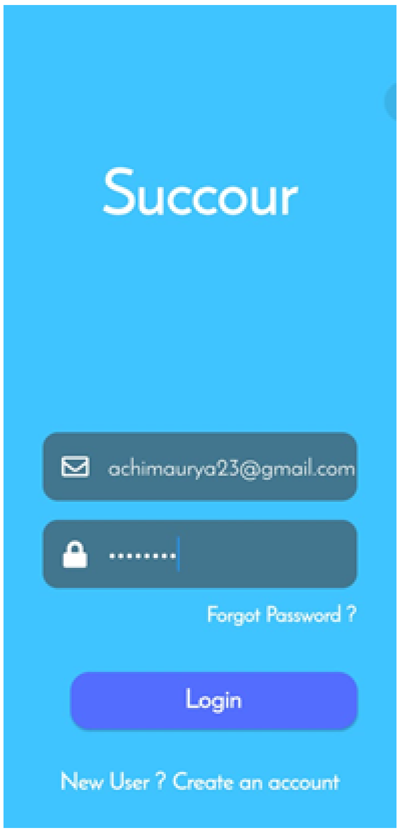


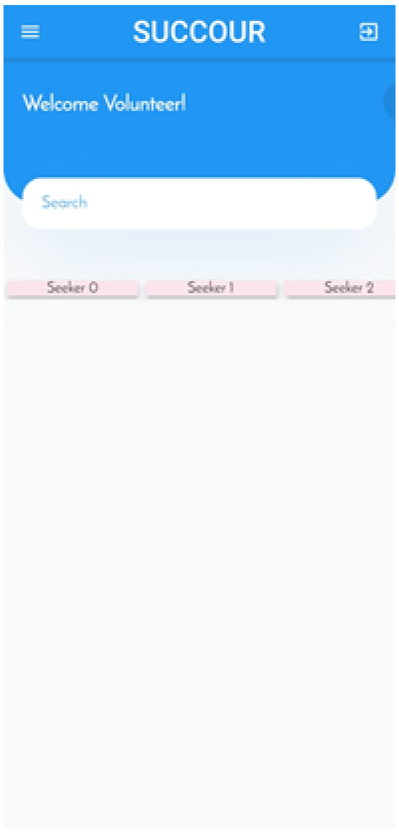

9935116160

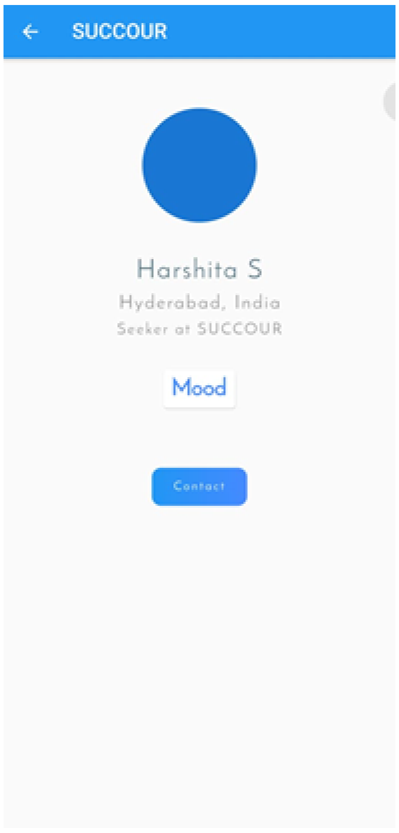

9935116160

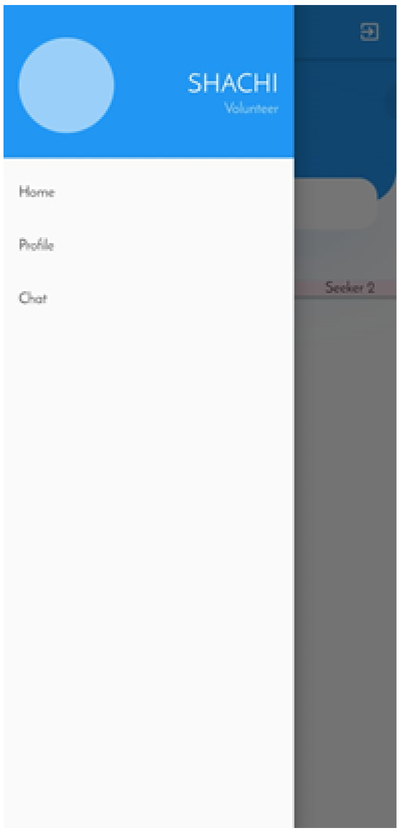
5.7 Volunteer Login and Homescreen

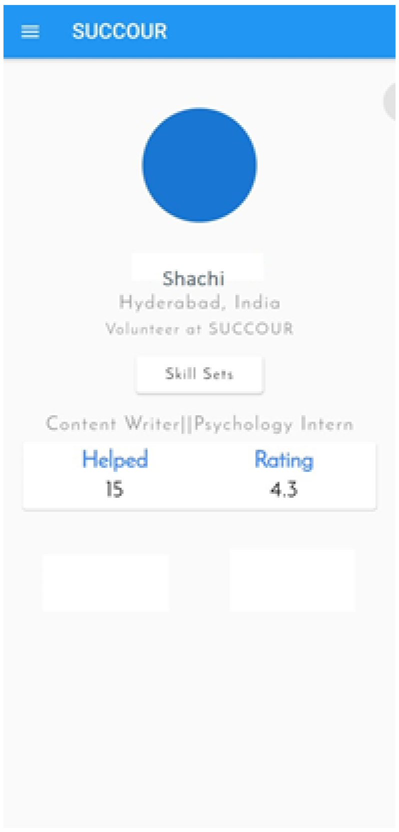



achimaurya23@gmail.com









6. Software Configuration Module

6.1 Configuration Identification

6.1.1 Source Code Modules and Test Cases

- Screens_D
    - Create_new_account_v1.0.dart
    - Login_screen_v1.0.dart
    - Login_screen_v1.1.dart
    - Forgot_password_v1.0.dart
    - Home_screen_v1.0.dart
    - Home_seeker_v1.0.dart
    - Welcome_v1.0.dart
    - ProfileViewv1.0.dart
    - ProfileView_v1,1.dart
    - Profile_v1.2.dart
    - ChatSearchScreen_v1.0.dart
    - Chat_screen_v1.0.dart
    - chats_v1.0.dart
    - Pat_records_v1.0.dart
    - screens_v1.0.dart
- Helper_N
    - Constants_v1.0.dart
    - helper_Functions_v1.0.dart
- Models_D
    - UserId_v1.0.dart
- Mood Tracker_D
    - databasesqf_v1.0.dart
    - historyCard_v1.0.dart
    - moodentryList_v1.0.dart
    - MoodEntry_v1.0.dart
    - routeManager_v1.0.dart
    - themes_v1.0.dart

        - chart_D
            - graph_v1.0.dart
            - moodOverTime_v1.0.dart
            - moodPerReason_v1.0.dart
        - Pages_D
            - analytics_v1.0.dart
            - history_v1.0.dart
            - HomePage_v1.0.dart
            - IconCheatSheet_v1.0.dart
            - setting_v1.0.dart
            - homeApp_D
                - home_v1.0.dart
                - moodSelect_v1.0.dart
                - moodSelect_v1.1.dart
    - Services_D
        - auth_v1.0.dart
        - database_v1.0.dart
    - Widgets_N
        - Password_input_v1.0.dart
        - text_field_input_v1.0.dart
        - widget_v1.0.dart

6.1.2 Requirement Specifications

- Android smartphone with API level greater than 20(Android OS 5.0 and newer).

6.1.3 Tools

- Android Studio
- Flutter
- Firebase

6.2 Baseline definition

6.2.1 Changes in the test cases:

- Addition of first page where the option of type of user appears was done so that one application can be used for both Seeker and Volunteer .It was done after the making of Login /Sign Up page by developer
- Addition of Mood Tracker to the application was done after release of version 1.0 because the objective of the application was to track and analyze the mood of the seeker. It was done by the developer.

6.2.2 Versions of application

- **Succour_v1.0:** (previous version**)**Login and Signup were authenticated and chats were backed up to firebase
- **Succour_v1.1:** (current version)Mood tracker was enabled and connected to the backend.
- **Succour_v1.2:** (future version) Video calls and Session history record will be enabled and location sharing will be done with scheduling appointments.

7. Code:

main.dart

```
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';
import 'screens/screens.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/cupertino.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Succour',
      theme: ThemeData(
        textTheme:
        GoogleFonts.josefinSansTextTheme(Theme.of(context).textTheme),
        primarySwatch: Colors.blue,
        visualDensity: VisualDensity.adaptivePlatformDensity,
      ),
      home: WelcomePage(),
      routes: {
        'Login': (context) => LoginScreen(),
        'ForgotPassword': (context) => ForgotPassword(),
        'CreateNewAccount': (context) => CreateNewAccount(),
      },
    );
  }
}
```

Welcome Page

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:succour/screens/login-screen.dart';
import 'package:succour/screens/login-screen2.dart';
import 'package:succour/screens/screens.dart';

class WelcomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.lightBlueAccent,
      body: Column(
        children: [
          Flexible(
            child: Center(
              child: Text(
                'Succour',
                style: TextStyle(
                    color: Colors.white,
                    fontSize: 60,
                    fontWeight:FontWeight.bold),
              ),
            ),
          ),
          Column(
            children: [
              MaterialButton(
                minWidth: 300.0,
                height: 60,
                onPressed: () {
                  Navigator.push(context,
                      MaterialPageRoute(builder: (context) =>
LoginScreen2()));
                },
                color: Colors.indigoAccent,
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(16),
                ),
                child: Text(
                  "Seeker",
                  style: TextStyle(
                      color: Colors.white,
                      fontWeight: FontWeight.w700,
                      fontSize: 24
                  ),
                ),
              ),
              SizedBox(height: 55),
              MaterialButton(
                minWidth: 300.0,
                height: 60,
                onPressed: () {
                  Navigator.push(context,
                      MaterialPageRoute(builder: (context) =>
LoginScreen()));
                },
                color: Colors.indigoAccent,
                shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(16)
                ),
                child: Text(
                  "Volunteer",
                  style: TextStyle(
                      color: Colors.white,
                      fontWeight: FontWeight.w700,
                      fontSize: 24
                  ),
                ),
              ),
              SizedBox(height: 50),
            ],
          )
        ],
      ),
    );
  }
}
```

Login Screen

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:succour/screens/home-screen.dart';
import 'package:succour/services/auth.dart';
import 'package:succour/services/database.dart';
import 'package:succour/widgets/widgets.dart';
import 'package:succour/helper/helperFunctions.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

class LoginScreen extends StatefulWidget {
  @override
  _LoginScreenState createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {

  AuthMethods authMethods = new AuthMethods();
  DatabseMethods databseMethods =new DatabseMethods();
  TextEditingController emailTextEditingController = new
TextEditingController();
  TextEditingController passwordTextEditingController = new
TextEditingController();
  final formkey = GlobalKey<FormState>();
  bool isLoading = false;
  QuerySnapshot userInfoSnapshot;

  signIn(){
    if(formkey.currentState.validate()){


//HelperFunctions.saveUserNameSharedPreference(userNameTextEditingContro
ller.text);

HelperFunctions.saveUserEmailSharedPreference(emailTextEditingController
.text);

      setState(() {
        isLoading = true;
      });


databseMethods.getUserByUserEmail(emailTextEditingController.text).then(
(val){
        userInfoSnapshot = val;

HelperFunctions.saveUserNameSharedPreference(userInfoSnapshot.docs[0]["n
ame"]);
      });

authMethods.signInWithEmailAndPassword(emailTextEditingController.text,
passwordTextEditingController.text).then((val){
        if(val != null){
          HelperFunctions.saveUserLoggedInSharedPreference(true);
          Navigator.pushReplacement(context,
              MaterialPageRoute(builder: (context) => MyHomePage()));
        }
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
          backgroundColor: Colors.lightBlueAccent,
          body: Column(
            children: [
              Flexible(
                child: Center(
                  child: Text(
                    'Succour',
                    style: TextStyle(
                        color: Colors.white,
                        fontSize: 60,
                        fontWeight: FontWeight.bold),
                  ),
                ),
              ),
              Column(
                crossAxisAlignment: CrossAxisAlignment.end,
                children: [
                  Form(
                    key: formkey,
                    child: Column(children: [
                      TextInputField(
                        icon: FontAwesomeIcons.envelope,
                        hint: 'Email',
                        inputType: TextInputType.emailAddress,
                        inputAction: TextInputAction.next,
                        myController: emailTextEditingController,
                      ),
                      PasswordInput(
                        icon: FontAwesomeIcons.lock,
                        hint: 'Password',
                        inputAction: TextInputAction.done,
                        myController: passwordTextEditingController,
                      ),
                    ],),
                  ),
                  GestureDetector(
                    onTap: () => Navigator.pushNamed(context,
'ForgotPassword'),
                    child: Text(
                      'Forgot Password ?',
                      style: TextStyle(
                        fontSize: 20,
                        fontWeight: FontWeight.bold,
                        color: Colors.white,
                        height: 1.5),
                    ),
                  ),
                  SizedBox(
                    height: 45,
                  ),
                  MaterialButton(
                    minWidth: 300.0,
                    height: 60,
                    onPressed: () {
                      signIn();
                    },
                    color: Colors.indigoAccent,
                    shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(20)
                    ),

                    child: Text(
                      "Login",
                      style: TextStyle(
                          color: Colors.white,
                          fontWeight: FontWeight.w700,
                          fontSize: 24
                      ),
                    ),
                  ),
                  SizedBox(
                    height: 45,
                  ),
                ],
              ),
              Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Text(
                      'New User ? ',
                      style: TextStyle(
                          color: Colors.white,
                          fontSize: 22,
                          fontWeight: FontWeight.w600
                      )
                  ),
                  GestureDetector(
                    onTap: () {
                      Navigator.pushNamed(context, 'CreateNewAccount');
                    },
                    child: Text(
                      'Create an account',
                      style: TextStyle(
                          color: Colors.white,
                          fontSize: 22,
                          fontWeight: FontWeight.w900
                      ),
                    ),
                  ),
                ],
              ),
              SizedBox(
                height: 35,
              ),
            ],
          ),
                  );
            }
          }
```

Create new account

```
import 'dart:ui';

import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:succour/helper/helperFunctions.dart';
import 'package:succour/screens/screens.dart';
import 'package:succour/services/auth.dart';
import 'package:succour/services/database.dart';
import 'package:succour/widgets/widgets.dart';

class CreateNewAccount extends StatefulWidget {

  @override
  _CreateNewAccountState createState() => _CreateNewAccountState();
}

class _CreateNewAccountState extends State<CreateNewAccount> {
  bool isLoading = false;

  final formKey = GlobalKey<FormState>();
  AuthMethods authMethods = new AuthMethods();
  DatabseMethods databseMethods = new DatabseMethods();

  TextEditingController userNameTextEditingController = new 
TextEditingController();
  TextEditingController emailTextEditingController = new 
TextEditingController();
  TextEditingController passwordTextEditingController = new 
TextEditingController();

  signUp() async {

    if(formKey.currentState.validate()){

      Map<String, String> userInfoMap ={
        "name" :  userNameTextEditingController.text,
        "email" : emailTextEditingController.text,
      };

HelperFunctions.saveUserNameSharedPreference(userNameTextEditingControll
er.text);
```

```
HelperFunctions.saveUserEmailSharedPreference(emailTextEditingController
.text);

      setState((){
        isLoading = true;
      });

authMethods.signUpWithEmailAndPassword(emailTextEditingController.text,
passwordTextEditingController.text).then((val){
        //print("${val.uid}");


        databseMethods.uploadUserInfo(userInfoMap);
        HelperFunctions.saveUserLoggedInSharedPreference(false);
        Navigator.pushReplacement(context,
            MaterialPageRoute(builder: (context) => WelcomePage()));

      });

    }
  }

  @override
  Widget build(BuildContext context) {
    Size size = MediaQuery.of(context).size;
    return Scaffold(
          backgroundColor: Colors.lightBlueAccent,
          body: isLoading ? Container(
            child: Center(
              child: Text(
              "Registered!!! Sign in.",
              style: TextStyle(
                  color: Colors.white,
                  fontWeight: FontWeight.w700,
                  fontSize: 24
              ),)
          )) : SingleChildScrollView(
            child: Column(
              children: [
                SizedBox(
                  height: size.width * 0.4,
                ),
                Form(
                  key: formKey,
                  child: Column(
                    children: [
                      Column(
                        children: [
                          TextInputField(
                            icon: FontAwesomeIcons.user,
                            hint: 'User',
                            inputType: TextInputType.name,
                            inputAction: TextInputAction.next,
                            myController: userNameTextEditingController,
                          ),
                          TextInputField(
                            icon: FontAwesomeIcons.envelope,
                            hint: 'Email',
                            inputType: TextInputType.emailAddress,
                            inputAction: TextInputAction.next,
                            myController: emailTextEditingController,
                          ),
                          PasswordInput(
                            icon: FontAwesomeIcons.lock,
                            hint: 'Password',
                            inputAction: TextInputAction.next,
                            myController: passwordTextEditingController,
                          ),
                        ],
                      ),
                      SizedBox(
                        height: 25,
                      ),
                      MaterialButton(
                        minWidth: 300.0,
                        height: 60,
                        onPressed: () {
                          signUp();

                        },
                        color: Colors.indigoAccent,
                        shape: RoundedRectangleBorder(
                            borderRadius: BorderRadius.circular(20)
                        ),

                        child: Text(
                          "Register",
                          style: TextStyle(
                              color: Colors.white,
                              fontWeight: FontWeight.w700,
                              fontSize: 24
                          ),
                        ),
                      ),
                      SizedBox(
                        height: 30,
                      ),
                      Row(
                        mainAxisAlignment: MainAxisAlignment.center,
                        children: [
                          Text(
                            'Already have an account? ',
                            style: TextStyle(
                              color: Colors.white,
                              fontSize: 22,
                              fontWeight: FontWeight.w600
                            )
                          ),
                          GestureDetector(
                            onTap: () {
```

```
                              Navigator.pushReplacement(context,
MaterialPageRoute(builder: (context) => WelcomePage()));
                            },
                            child: Text(
                              'Login',
                              style: TextStyle(
                                color: Colors.white,
                                fontSize: 22,
                                fontWeight: FontWeight.bold
                            ),
                          ),
                          ),
                        ],
                      ),
                      SizedBox(
                        height: 20,
                      ),
                    ],
                  ),
                )
              ],
            ),
          ),
        );
  }
}
```

Grid Screen

```
import 'package:flutter/material.dart';
import 'package:succour/Mood%20Tracker/pages/homeApp/home.dart';
import 'package:succour/helper/constants.dart';
import 'package:succour/screens/profileSeeker.dart';
import 'package:succour/screens/profileView.dart';
import 'package:succour/screens/screens.dart';
import 'package:succour/services/auth.dart';
import 'package:syncfusion_flutter_calendar/calendar.dart';
import 'package:succour/screens/chats.dart';


class MyGridScreen extends StatefulWidget {
  MyGridScreen({Key key}) : super(key: key);

  @override
  _MyGridScreenState createState() => _MyGridScreenState();
}

class _MyGridScreenState extends State<MyGridScreen> {
  AuthMethods authMethods = new AuthMethods();
  @override
  Widget build(BuildContext context) {
    return new Scaffold(
      backgroundColor: Colors.white,
      appBar: new AppBar(
        title: new Text('SUCCOUR'),
        backgroundColor: Colors.blue,
        actions: [
          GestureDetector(
            onTap: (){
              authMethods.signOut();
              Navigator.pushReplacement(context,
MaterialPageRoute(builder: (context) => WelcomePage()));
            },
            child:
            Container(
              padding: EdgeInsets.symmetric(horizontal: 16),
              child: Icon(Icons.exit_to_app)
              ,
            ),
          ),
        ],
      ),
      body: new GestureDetector(
        behavior: HitTestBehavior.opaque,
        onPanDown: (detail) {
          print(detail);
          FocusScope.of(context).requestFocus(new FocusNode());
        },
        child: new ListView(
          physics: const AlwaysScrollableScrollPhysics(),
          shrinkWrap: true,
          children: <Widget>[
            new SizedBox(height: 20.0),
            /*new Container(
              height: 60.0,
              color: Colors.blue,
              child: new Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: <Widget>[
                  new Icon(Icons.hourglass_empty,
                      color: Colors.white, size: 30.0),
                  new Padding(padding: const EdgeInsets.only(right:
5.0)),
                ],
              ),
            ),*/
            new SizedBox(height: 20.0),
            new Container(
              child: new ListView.builder(
                shrinkWrap: true,
                itemCount: 1,
                itemBuilder: (context, index) {
                  return new Column(
                    children: <Widget>[
                      new Container(
                        height: 50.0,
                        color: Colors.white,
                        child: new Row(

                          children: <Widget>[
                            new Padding(
                                padding: const EdgeInsets.only(right:
5.0)),
                            new Text( 'How are you feeling?',
                                style: new TextStyle(
                                    fontSize: 30.0, color:
Colors.black)),
                          ],
                        ),
                      ),
                      new Container(
                        height: 120.0,
                        child: new ListView.builder(
                          shrinkWrap: true,
                          scrollDirection: Axis.horizontal,
                          itemCount: 1,
                          itemBuilder: (context, index) {
                            return new GestureDetector(
                              child: new Card(
                                elevation: 5.0,
                                child: new Container(
                                  color: Colors.blue[50],
                                  height:
MediaQuery.of(context).size.width / 10,
                                  width:
MediaQuery.of(context).size.width / 3,
                                  alignment: Alignment.center,
                                  child: new Text('Mood Tracker'),
                                ),
                              ),
                              onTap: () {
                                Navigator.push(context,
                                    MaterialPageRoute(builder: (context)
=> MoodTracker()));
                              },
                            );
                          },
                        ),
                      ),
                      new SizedBox(height: 20.0),
                      new Container(
                        height: 50.0,
                        color: Colors.white,
                        child: new Row(

                          children: <Widget>[
                            new Padding(
                                padding: const EdgeInsets.only(right:
5.0)),
                            new Text( 'Connect to a Volunteer',
                                style: new TextStyle(
                                    fontSize: 30.0, color:
Colors.black)),
                          ],
                        ),
                      ),
                      new Container(
                        height: 120.0,
                        child: new ListView.builder(
                          shrinkWrap: true,
                          scrollDirection: Axis.horizontal,
                          itemCount: 10,
                          itemBuilder: (context, index) {
                            return new GestureDetector(
                              child: new Card(
                                elevation: 5.0,
                                child: new Container(
                                  color: Colors.pink[50],
                                  height:
MediaQuery.of(context).size.width / 10,
                                  width:
MediaQuery.of(context).size.width / 3,
                                  alignment: Alignment.center,
                                  child: new Text('Volunteer  $index'),
                                ),
                              ),
                              onTap: () {
                                Navigator.push(context,
                                    MaterialPageRoute(builder: (context)
=> ProfileView()));
                              },
                            );
                          },
                        ),
                      ),
                      new SizedBox(height: 20.0),
                      new Container(
                        height: 50.0,
                        color: Colors.white,
                        child: new Row(

                          children: <Widget>[
                            new Padding(
                                padding: const EdgeInsets.only(right:
5.0)),

                            new Text( 'Scheduled Appointments',
                                style: new TextStyle(
                                    fontSize: 30.0, color:
Colors.black)),
                          ],
                        ),
                      ),
                      new Container(
                        child: SfCalendar(
                          view: CalendarView.month,
                        ),
                      ),
                    ],
                  );
                },
              ),
            ),
          ],
        ),
      ),

      drawer: Drawer(
        // Add a ListView to the drawer. This ensures the user can
scroll
        // through the options in the drawer if there isn't enough
vertical
        // space to fit everything.
        child: ListView(
          // Important: Remove any padding from the ListView.
          padding: EdgeInsets.zero,
          children: <Widget>[
            DrawerHeader(
              child:Stack(
                children:<Widget>[
                  Align(
                    alignment: Alignment.centerLeft,
                    child:CircleAvatar(
                      backgroundColor:
Colors.white54,//Image("https://unsplash.com/photos/OhKElOkQ3RE"),
                      radius:50.0,
                    ),
                  ),
                  Align(
                    alignment: Alignment.centerRight,
                    child:Text(Constants.myName.toUpperCase(),
                      style:TextStyle(color:Colors.white, fontSize:24),
                    ),
                  ),
                  Align(
                    alignment: Alignment.centerRight+Alignment(0,0.4),
                    child:Text("Seeker",
                        style: TextStyle(color:Colors.white70)),
                  ),

                ],
              ) ,

              decoration:
              BoxDecoration(
                  color: Colors.blue
              ),
            ),
            ListTile(
              title: Text('Home'),
              onTap: () {
                // Update the state of the app
                // ...
                // Then close the drawer
                Navigator.pop(context);
              },
            ),
            ListTile(
              title: Text('Profile'),
              onTap: () {
                // Update the state of the app
                // ...
                // Then close the drawer
                Navigator.push(context,
                    MaterialPageRoute(builder: (context) =>
ProfileSeeker())
                );

              },
            ),
            ListTile(
              title: Text('Chat'),
              onTap: () {
                // Update the state of the app
                // ...
                // Then close the drawer
                Navigator.push(context,
                    MaterialPageRoute(builder: (context) => Chats()));
              },
            ),
          ],
        ),
      ),
    );
  }
}
```

Chat Screen

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/painting.dart';
import 'package:succour/helper/constants.dart';
import 'package:succour/screens/call.dart';
import 'package:succour/services/database.dart';

class ChatScreen extends StatefulWidget {

  final String userName;
  final String chatRoomId;
  ChatScreen({
    this.userName,
    this.chatRoomId
});
  @override
  _ChatScreenState createState() => _ChatScreenState();
}

class _ChatScreenState extends State<ChatScreen> {

  DatabseMethods databseMethods =new DatabseMethods();
  TextEditingController messageTextEditingController = new
TextEditingController();
  Stream chatMessageStream;
  Widget ChatMessageList(){
    return StreamBuilder(
      stream: chatMessageStream,
      builder: (context, snapshot){
        return snapshot.hasData ? ListView.builder(
            itemCount: snapshot.data.docs.length,
            itemBuilder: (context, index){
              return
MessageTile(snapshot.data.docs[index]["message"],snapshot.data.docs[inde
x]["sendBy"] == Constants.myName);
            }) : Container();
      },

    );

  }

  sendMessage(){
    if(messageTextEditingController.text.isNotEmpty){
      Map<String, dynamic> messageMap={
        "message" : messageTextEditingController.text,
        "sendBy" : Constants.myName,
        "time" : DateTime.now().millisecondsSinceEpoch,
      };
      databseMethods.addConversationMessages(widget.chatRoomId,
messageMap);
      messageTextEditingController.text="";
    }
  }
  @override
  void initState() {

databseMethods.getConversationMessages(widget.chatRoomId).then((val){
      setState(() {
        chatMessageStream = val;
      });
    });
    super.initState();
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        brightness: Brightness.dark,
        centerTitle: true,
        title: RichText(
          textAlign: TextAlign.center,
          text: TextSpan(
            children: [
              TextSpan(
                  text: widget.userName.toUpperCase(),
                  style: TextStyle(
                    fontSize: 16,
                    fontWeight: FontWeight.w400,
                  )),
            ],
          ),
        ),
        leading: IconButton(
            icon: Icon(Icons.arrow_back_ios),
            color: Colors.white,
            onPressed: () {
              Navigator.pop(context);
            }),
        actions: [
          IconButton(
              icon: Icon(Icons.call),
            color: Colors.white,
            onPressed: (){
              Navigator.push(context,
                  MaterialPageRoute(builder: (context) =>
CallNumber()));
            },
          )
        ],

      ),
      body: Container(
        child: Stack(
          children: [
            ChatMessageList(),
            Container(
              alignment: Alignment.bottomCenter,
              child: Container(
                padding: EdgeInsets.symmetric(horizontal: 8),
                height: 70,
                color: Colors.white,
                child: Row(
                  children: <Widget>[
                    IconButton(
                      icon: Icon(Icons.photo),
                      iconSize: 25,
                      color: Theme.of(context).primaryColor,
                      onPressed: () {},
                    ),
                    Expanded(
                      child: TextField(
                        controller: messageTextEditingController,
                        decoration: InputDecoration.collapsed(
                          hintText: 'Send a message..',
                        ),
                        textCapitalization:
TextCapitalization.sentences,
                      ),
                    ),
                    IconButton(
                      icon: Icon(Icons.send),
                      iconSize: 25,
                      color: Theme.of(context).primaryColor,
                      onPressed: () {
                        sendMessage();
                      },
                    ),
                  ],
                ),
              )
            )
          ],
        ),
      ),
    );
  }
}

class MessageTile  extends StatelessWidget {

  final bool isSendByMe;
  final String message;
  MessageTile(this.message, this.isSendByMe);
  @override
  Widget build(BuildContext context) {
    return Container(
      padding: EdgeInsets.only(left: isSendByMe ? 0 : 8 , right:
isSendByMe ? 8 : 0),
      margin: EdgeInsets.symmetric(vertical: 4),
      width: MediaQuery.of(context).size.width,
      alignment: isSendByMe ? Alignment.centerRight :
Alignment.centerLeft,
      child: Container(
        padding: EdgeInsets.symmetric(horizontal: 24, vertical: 14),
        decoration: BoxDecoration(
          gradient: LinearGradient(
            colors: [
              const Color(0xff007EF4),
              const Color(0xff2A75BC)
            ]

          ),
         borderRadius: isSendByMe ?
         BorderRadius.only(
           topLeft: Radius.circular(20),
           topRight: Radius.circular(20),
           bottomLeft: Radius.circular(20),
         ) :
         BorderRadius.only(
           topLeft: Radius.circular(20),
           topRight: Radius.circular(20),
           bottomRight: Radius.circular(20),
         )

        ),
        child: Text(message, style: TextStyle(
          fontSize: 17,
          color: Colors.white70,
        ),),
      ),
    );
  }
}
```

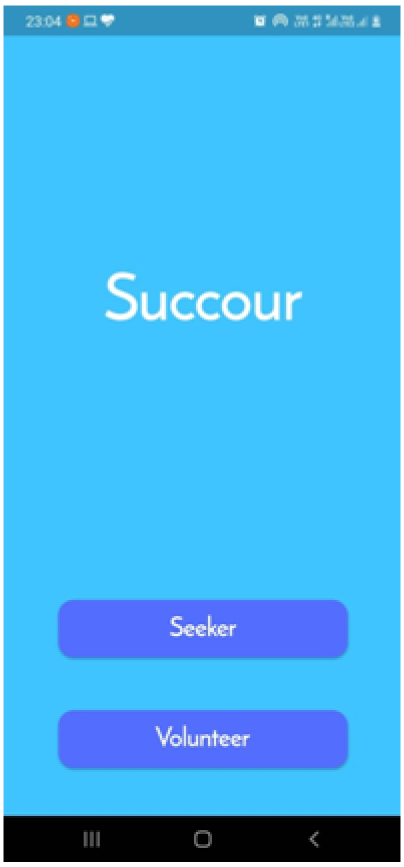
Call

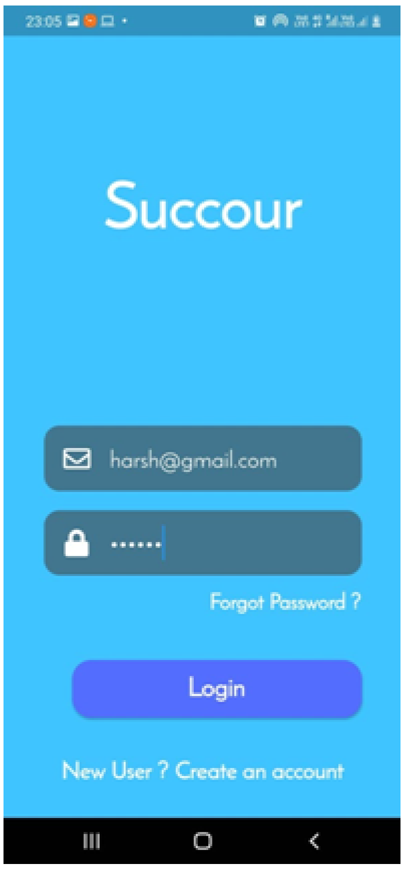
```
import 'package:flutter/material.dart';
import
'package:flutter_phone_direct_caller/flutter_phone_direct_caller.dart';
import 'package:url_launcher/url_launcher.dart';

class CallNumber extends StatefulWidget {
  @override
  _CallNumberState createState() => _CallNumberState();
}

class _CallNumberState extends State<CallNumber> {
  TextEditingController textEditingController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: new AppBar(
        title: new Text('Call'),
        backgroundColor: Colors.blue,
      ),
      body: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Container(
            child: TextFormField(
              keyboardType: TextInputType.phone,
              controller: textEditingController,
              onSaved: (phoneNumber) {
                textEditingController.text = phoneNumber;
              },
            ),
          ),
          // ignore: deprecated_member_use
          RaisedButton(
            child: Text("Launch PhoneURL"),
            onPressed: () {
              _launchPhoneURL(textEditingController.text);
            },
          ),
          // ignore: deprecated_member_use
          RaisedButton(
            child: Text("Call Number"),
            onPressed: () {
              _callNumber(textEditingController.text);
            },
          )
        ],
      ),
    );
  }
}

_callNumber(String phoneNumber) async {
  String number = phoneNumber;
  await FlutterPhoneDirectCaller.callNumber(number);
}

_launchPhoneURL(String phoneNumber) async {
  String url = 'tel:' + phoneNumber;
  if (await canLaunch(url)) {
    await launch(url);
  } else {
    throw 'Could not launch $url';
  }
}
```

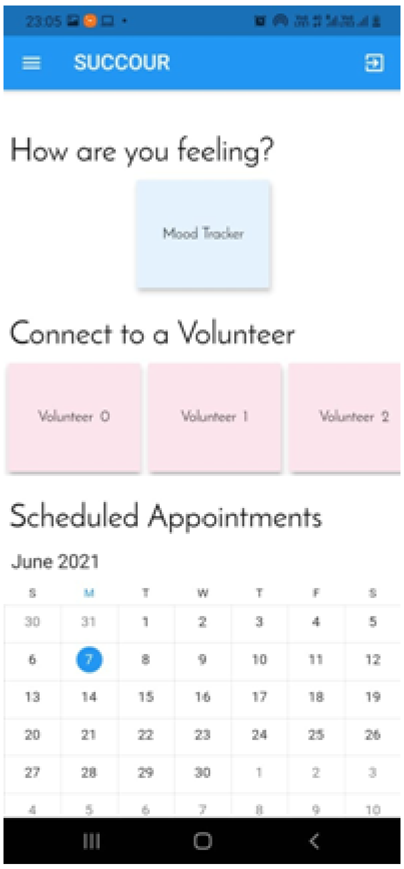
Home Page

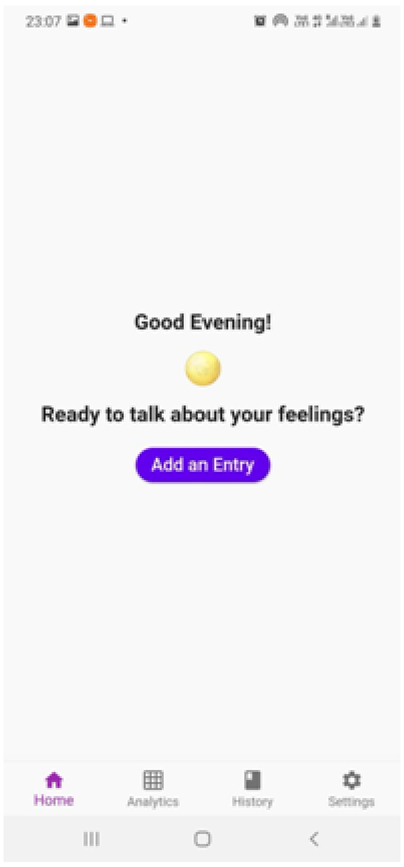
```
import 'package:flutter/material.dart';
import 'package:succour/Mood Tracker/databasesqf.dart';

class HomePage extends StatefulWidget {
  final Function refresh;
  HomePage({@required this.refresh});
  @override
  _HomePageState createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  List<Map<String, dynamic>> moodList = [];

  Future moodsFuture;

  @override
  void initState() {
    super.initState();
    moodsFuture = getMoods();
  }

  getMoods() async {
    final _moodsData = await DBProvider.db.getMood();
    return _moodsData;
  }

  @override
  Widget build(BuildContext context) {
    return Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          FutureBuilder(
              future: getMoods(),
              builder: (context, moodsData) {
                switch (moodsData.connectionState) {
                  case ConnectionState.none:
                  case ConnectionState.waiting:
                  case ConnectionState.active:
                  case ConnectionState.done:
                    var moodList = moodsData.data;
                    DateTime lastEntryDate = moodList != null &&
moodList != 123
                        ? DateTime.parse(moodList[moodList.length -
1]['date'])
                        : DateTime.now().subtract(Duration(days: 1));
                    DateTime now = DateTime.now();
                    int hour = now.hour;
                    String timeMsg = "";
                    String imgPath = "assets/images/sun.png";
                    String readyMsg = "Ready to talk about your
feelings?";
                    if (hour >= 6 && hour < 12) {
                      timeMsg = "Good Morning!";
                      imgPath = "assets/images/sun.png";
                    } else if (hour >= 12 && hour < 17) {
                      timeMsg = "Good Afternoon!";
                      imgPath = "assets/images/sun.png";
                    } else {
                      timeMsg = "Good Evening!";
                      imgPath = "assets/images/moon.png";
                    }

                    Function onPress = () {
                      Navigator.pushNamed(context, '/ms2',
                          arguments: widget.refresh);
                    };

                    //Add this back to enable one entry per day
                    if (lastEntryDate.day == now.day &&
                        lastEntryDate.month == now.month &&
                        lastEntryDate.year == now.year) {
                      readyMsg = "Let's talk more tomorrow!";
                      onPress = null;
                      //return content;
                    }

                    Widget content = Column(
                      children: <Widget>[
                        Text(
                          timeMsg,
                          style: TextStyle(
                              fontSize: 20, fontWeight:
FontWeight.bold),
                          //textAlign: TextAlign.center,
                        ),
                        SizedBox(height: 15),
                        Image.asset(
                          imgPath,
                          height: 40,
                        ),
                        SizedBox(height: 15),
                        Text(
                          readyMsg,
                          style: TextStyle(
                              fontSize: 20, fontWeight:
FontWeight.bold),
                          textAlign: TextAlign.center,
                        ),
                        SizedBox(height: 15),
                        RaisedButton(
                            child: Text("Add an Entry"), onPressed:
onPress),
                      ],
                    );

                    return moodList != null ? content : Container();
                }
                return Container();
              })
        ],
      ),
    );
  }
}
```

8. Screenshots





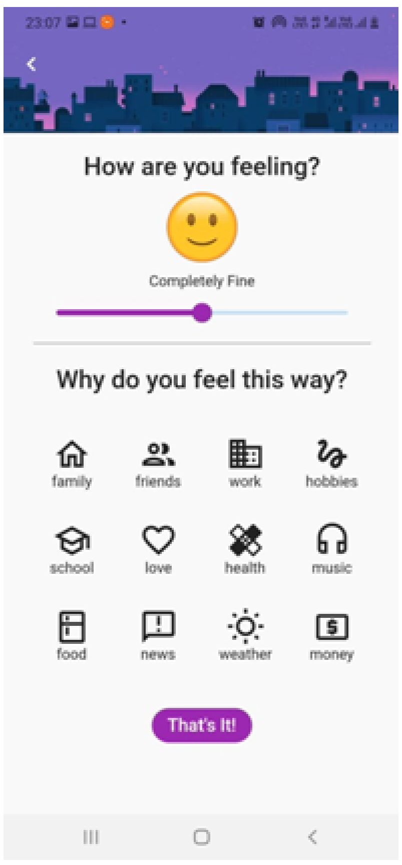




Good Evening!

Ready to talk about your feelings?

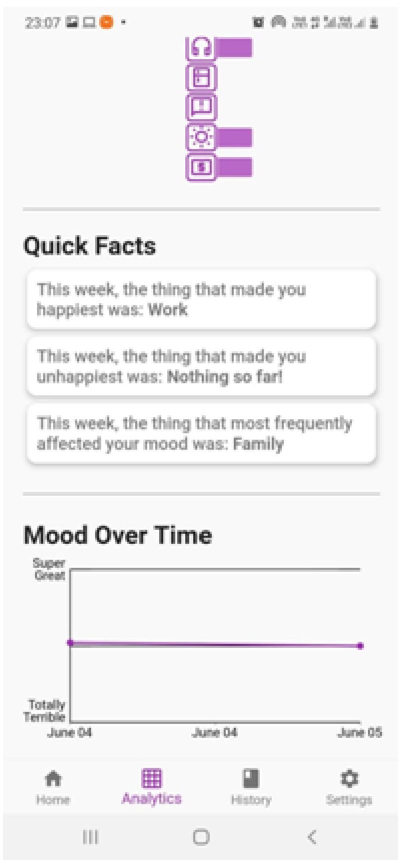
Add an Entry



How are you feeling?

Completely Fine

Why do you feel this way?



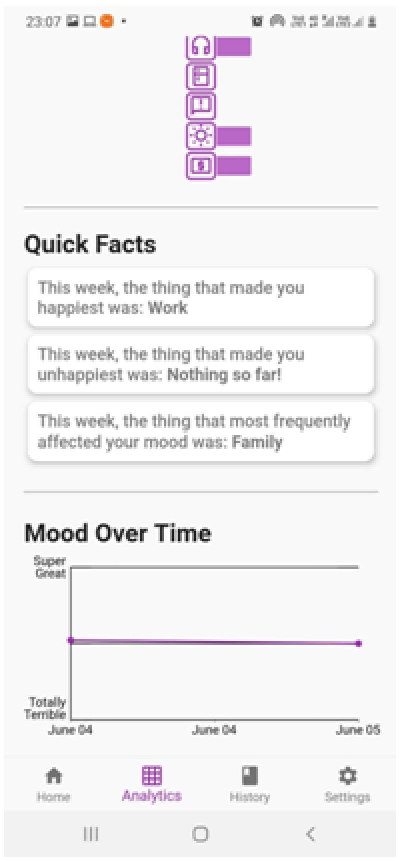
Quick Facts

This week, the thing that made you happiest was: **Work**

This week, the thing that made you unhappiest was: **Nothing so far!**

This week, the thing that most frequently affected your mood was: **Family**

Mood Over Time



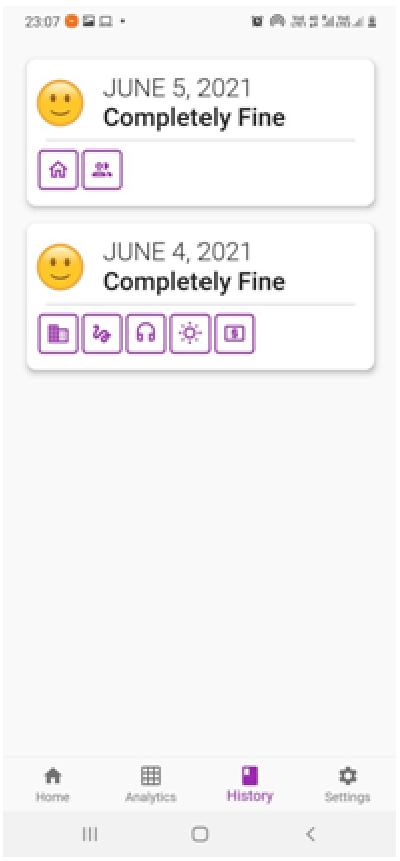
Quick Facts

This week, the thing that made you happiest was: **Work**

This week, the thing that made you unhappiest was: **Nothing so far!**

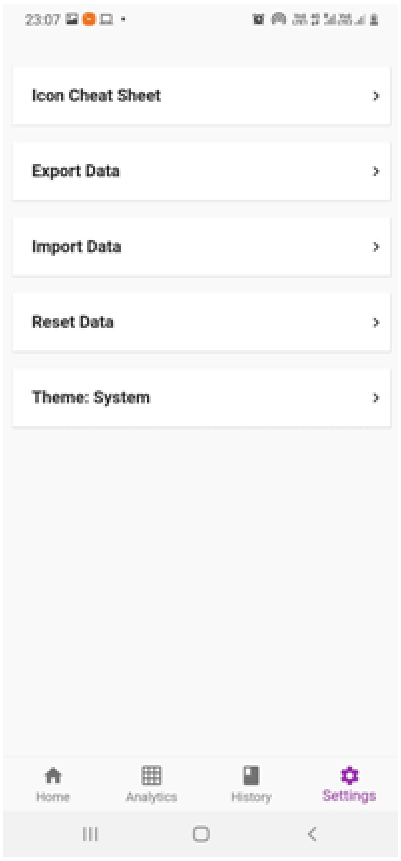
This week, the thing that most frequently affected your mood was: **Family**

Mood Over Time



JUNE 5, 2021
Completely Fine

JUNE 4, 2021
Completely Fine



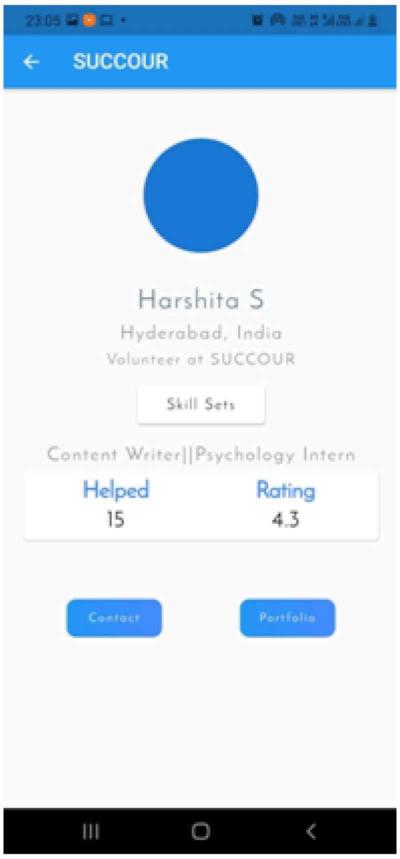
Icon Cheat Sheet ›

Export Data ›

Import Data ›

Reset Data ›

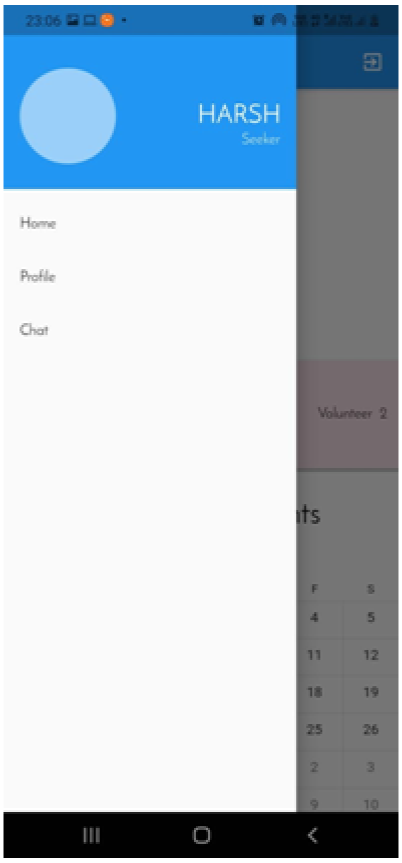
Theme: System ›

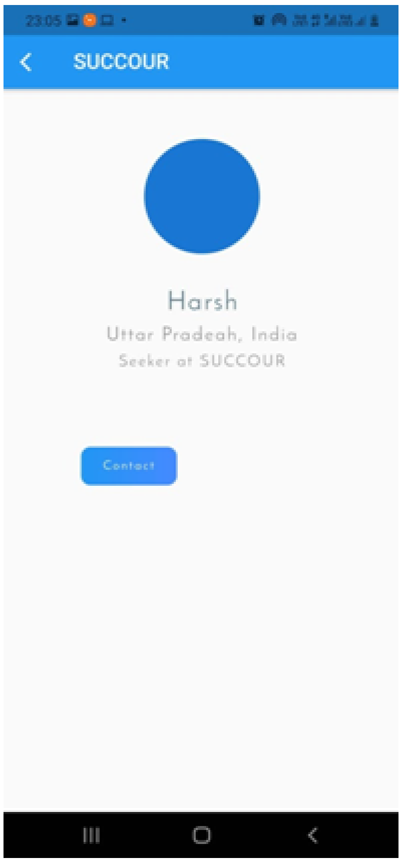


Harshita S
Hyderabad, India
Volunteer at SUCCOUR

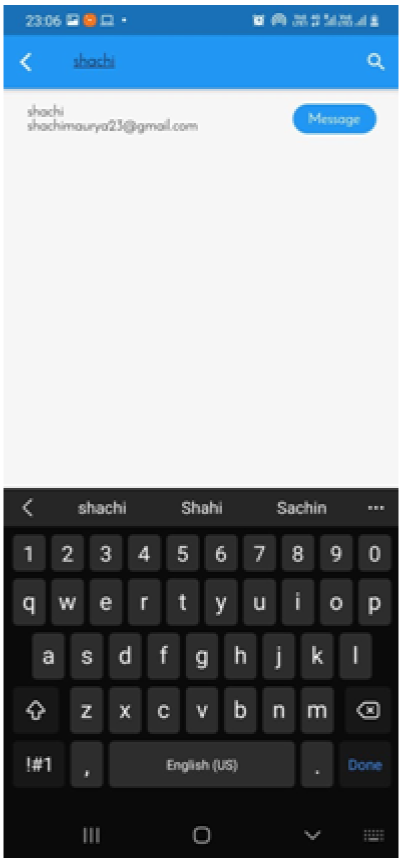
Content Writer||Psychology Intern

Helped	Rating
15	4.3

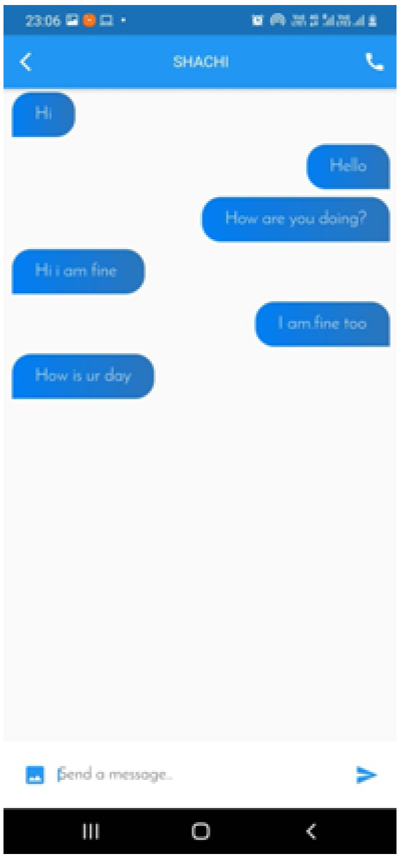




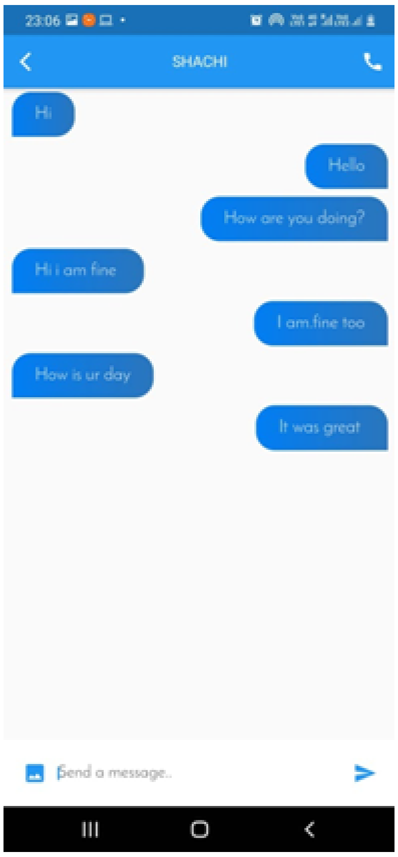
Harsh
Uttar Pradeah, India
Seeker at SUCCOUR



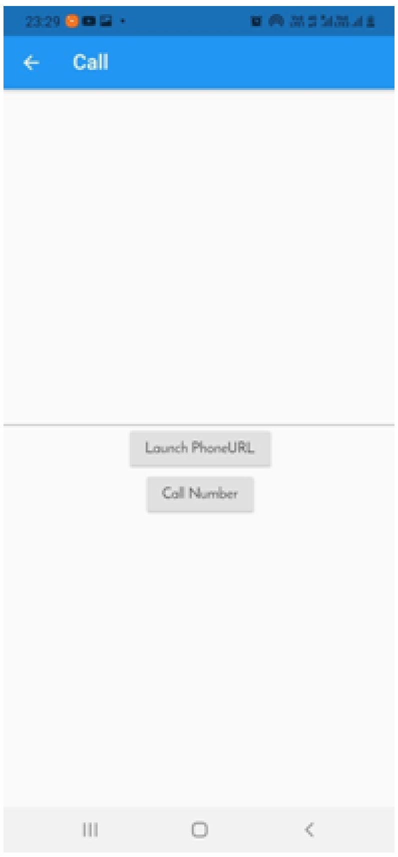
shachi
shachimaurya23@gmail.com

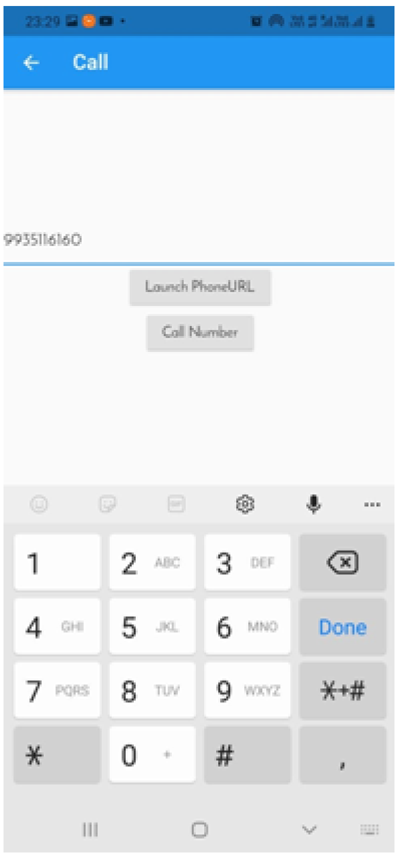


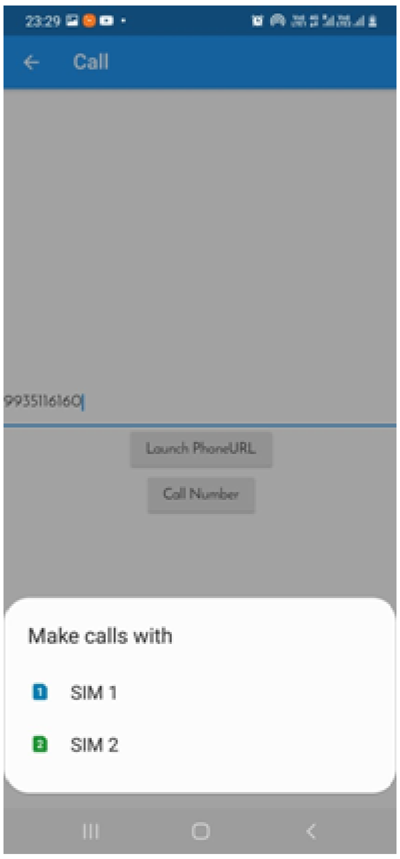
Hi
Hello
How are you doing?
Hi i am fine
I am fine too
How is ur day

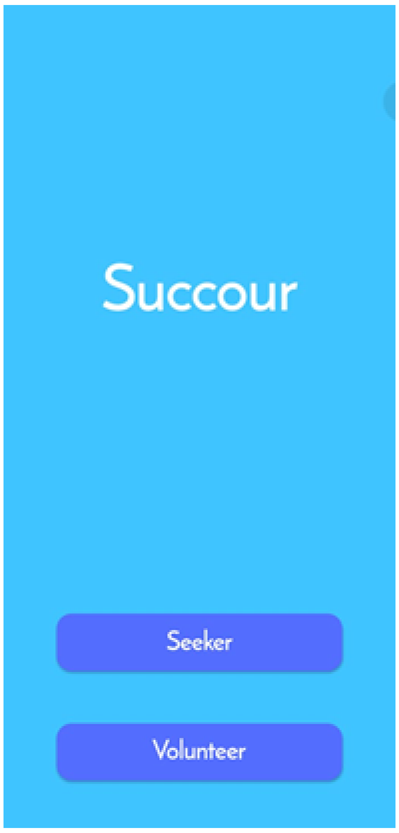


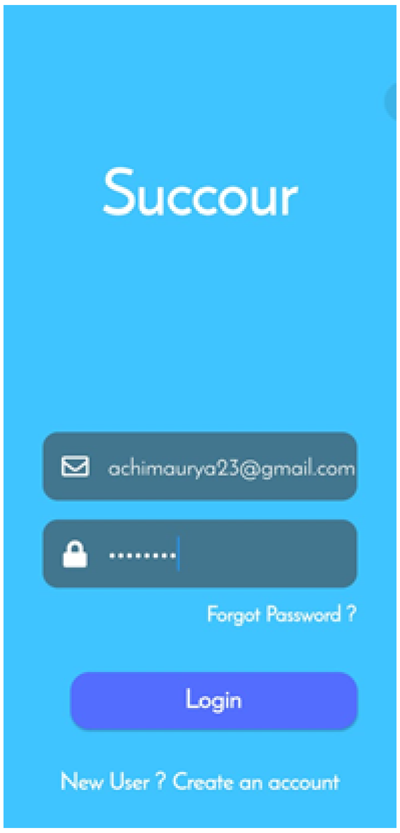
Hi
Hello
How are you doing?
Hi i am fine
I am fine too
How is ur day
It was great

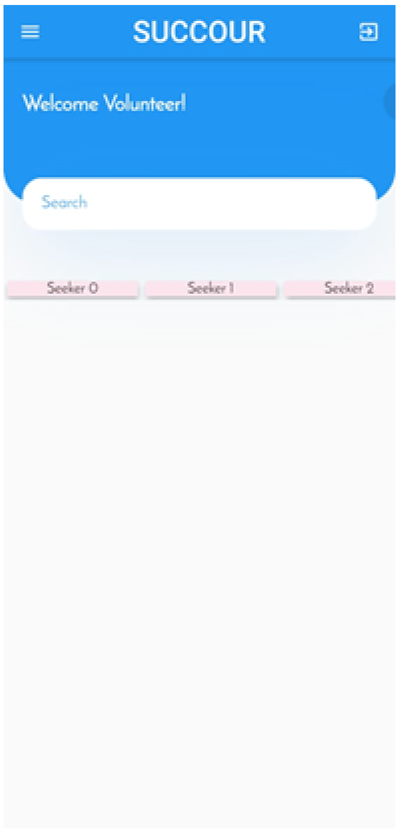

23:29
Call
Launch PhoneURL
Call Number

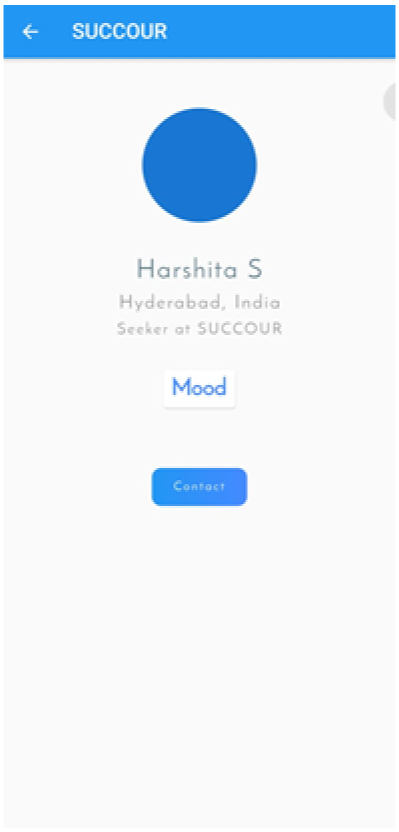

23:29
Call
9935116160
Launch PhoneURL
Call Number
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
Done
7 PQRS
8 TUV
9 WXYZ
*+#
*
0 +
#
,

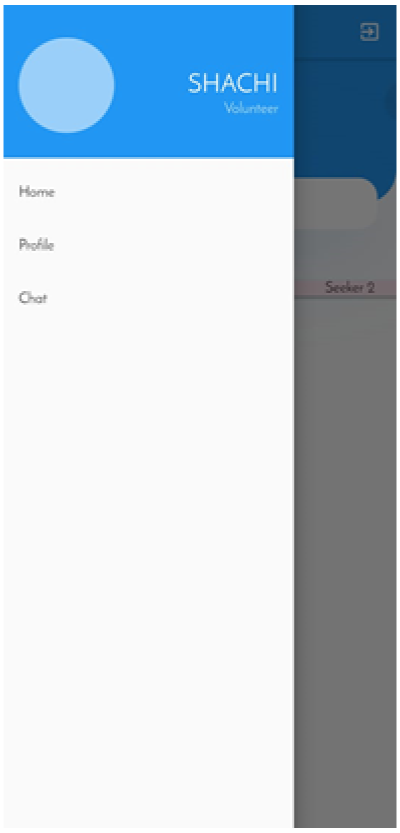

23:29
Call
9935116160
Launch PhoneURL
Call Number
Make calls with
SIM 1
SIM 2

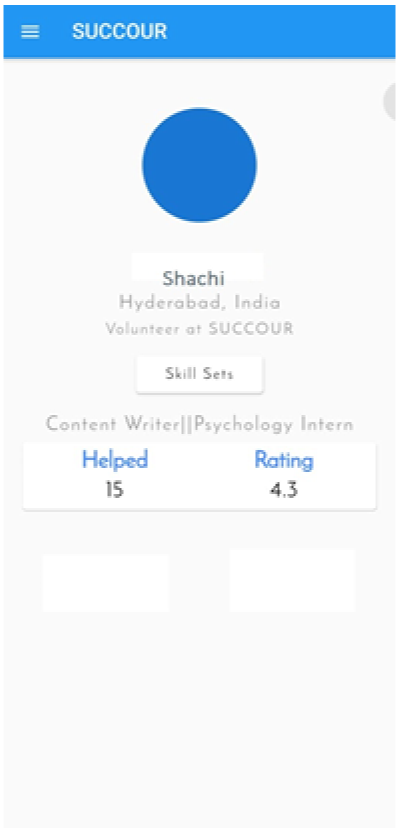

Succour
Seeker
Volunteer

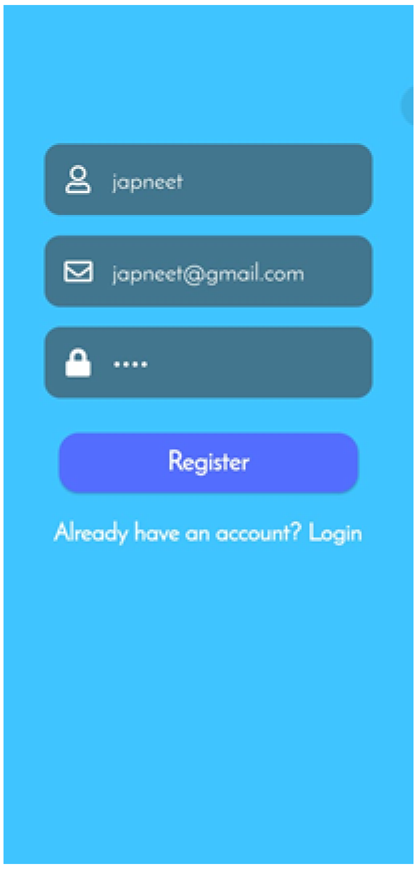

Succour
achimaurya23@gmail.com
••••••••
Forgot Password ?
Login
New User ? Create an account

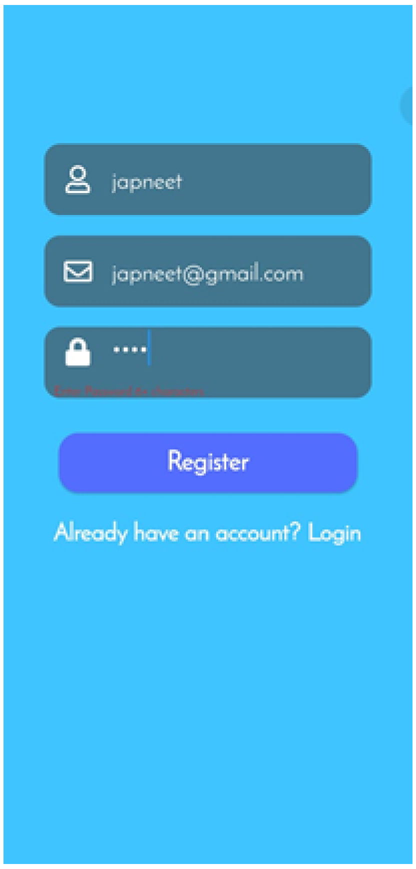

SUCCOUR
Welcome Volunteer!
Search
Seeker 0
Seeker 1
Seeker 2

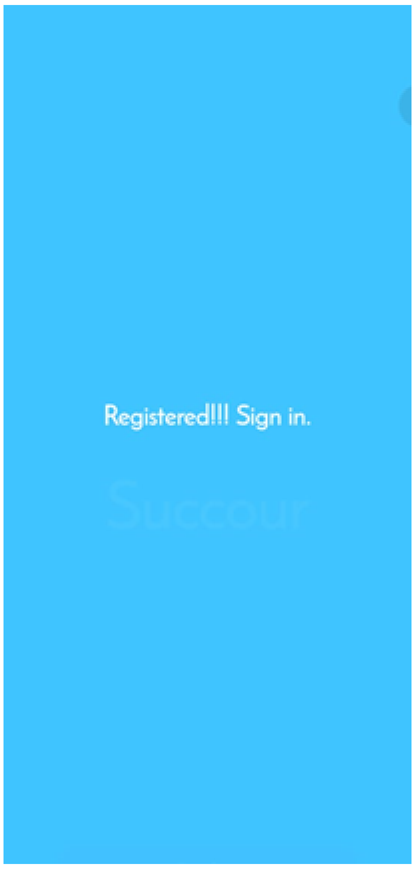












9. Conclusion & Future work

9.1 Conclusion

We have successfully created an app where a user(seeker) can keep the track record of his daily moods with what activity made them feel better and which did not. They can also talk to another person through the chats like an online counselling session and if the user wants they can ask for the counselling session over a call too.Also the user(volunteer) can keep in touch with the seekers under their help .

9.2 Future work

We plan to add more features to the app like group sessions, video calls and also implementing a machine learning module to suggest some activities to the user that might cheer up his mood.Also we want to add location sharing feature using Google maps to allow face to face meetups between seeker and volunteer .

10. References

1. https://flutter.dev › docs › reference › tutorials
2. https://flutterappworld.com/a-digital-diary-and-mood-tracking-app-built-with-flutter/
3. https://medium.com/flutter-community/creating-services-to-do-the-work-in-your-flutter-app-93d6c4aa7697
4. https://protocoderspoint.com/flutter-profile-page-ui-design-social-media-2/
5. https://cloud.google.com/firestore/docs/client/get-firebase
6. https://pub.dev/
7. https://www.figma.com/?fuid=