Pmate Android Application

BY

Md. Abdullah Al Masud

ID: 161 15 6882

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Dr. Sheak Rashed Haider Noori

Associate Professor

Department of CSE

Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

DECEMBER, 2019

APPROVAL

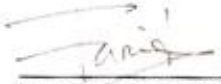
This Project titled **"Pmate- A Virtual Assistant"**, submitted by Md. Abdullah Al Masud, ID No: 161-15-6882 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 05-12-2019.

BOARD OF EXAMINERS

Dr. Syed AkhterHossain — **Chairman**
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Saiful Islam — **Internal Examiner**
Senior Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Shaon Bhatta Shuvo — **Internal Examiner**
Senior Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

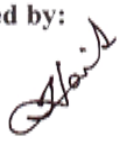


Dr. Dewan Md. Farid — **External Examiner**
ssociate Professor
Department of Computer Science and Engineering
United International University

DECLARATION

I hereby declare that, this project has been done by me under the supervision of **Dr. Sheak Rashed Haider Noori, Associate Professor, Department of CSE,** Daffodil International University. I also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:



Dr. Sheak Rashed Haider Noori,

Associate Professor

Department of CSE

Daffodil International University

Submitted by:

Md. Abdullah Al Masud

ID: 161-15-6882

Department of CSE

Daffodil International University

ACKNOWLEDGEMENTS

I have given my efforts to this thesis. However, it would not have been possible without the kind support and help of many individuals. I would like to express my deepest appreciation to all those who provided me the possibility to complete this report.

I would like to express my gratitude to Dr. Sheakh Rashed Haider Noori, my project advisor for providing me an opportunity to work on this project, which gave me a great exposure to develop the skills towards Android Application Development. I thank him for his generous advice and encouragement which helped me to complete the project successfully.

In addition, my gratitude is extended to Dr. Mark James Bartholomew for giving me the opportunity to work as a teaching assistant.

Lastly, I would also like to acknowledge with much appreciation the crucial role of my department head, Professor Dr Syed Akhter Hossain, who provided me with his precious time and kind help to finish this thesis. I also give my deepest thanks to all the faculty members and staff of CSE department of Daffodil International University.

Abstract

The purpose of this project was to develop an android application which will work as our own assisstant. The specific function of this application was to maintain our schedule. This application will take the necessary information from the user then it will remind them about their schedule. This application will help you to know about other people's schedule. This app is able to fix up your meeting and appointment with other's people.

The technology used in this project are the Android development Kit (SDK) , internal storage and external storage. The Android SDK is used to take care of general Android application development, internal storage is used for storing the user's login information and user's data and the external storage is used for storing user's details, login information and user's data.

In this project I have set out to identify any methods currently used in related application to store and fetching data from different storage and how they can be implemented successfully. From this information I have find my own way, I have carried through the ideas, incorporating my own thoughts, to formulate suggestions on how this could be done.

TABELS OF CONTENTS

CONTENTS

PAGE NO

TABELS OF FIGURES

FIGURES **PAGE NO**

Chapter 1
Introduction

1.1 Context

To conclude our educational process we had to carry out with a project. My project was about to developing an android app. Tcehnology made our life more easier. Now a days we can't think without technology beacuse we are using it in our every phase of life. Android application is also an another gift of technlogy. we can do many difficult task within a moment by using an android application. Since it is inside our phone so we can carry it always with us. So that I choosed android application which can assist us all the time. My project was about to developing an android application as a virtual assistant. This app has two purpose. firstly it will remind us about our daily schedule and secondly it will help us to know about others schedule and interact with that. Actually this android app will work as our assistant which will help us to do those things. This app will notify when, where and with whom you have your next schedule and if you give permission then other people can also interact with your schedule. as well as you will also get the oppurtunity to know other's schedule. You can make your own schedule, you can request others people for fix up a schedule through this app. Fist of all you have to sign up through this app by using some of your information then you have to login using your username and password. through this app you can search for your friends and collegues. you can add them on your friendlist. through this app you can watch your friend's schedule and make an request to fix up a schedule with them. if you accept anyone's schedule request then this app will let them know about your acceptance and it will add that schedule request in your schedule list.

1.2 Motivation

Few moths ago I worked as a teaching assistant with one of my honorable teacher named Dr. Mark Bartholomew, He had his own personal assistant. Almost everyday I saw that his assistant is facing problem to fix up his schedule and others people were waiting for him for a long time beacuse they didn't know when he will be free. From this problem I got an idea which can minimize this problem. Then I searched for

different type of android apps which work as a assistant But those app wasn't eligible to minimize those problem. maximum android apps which actually work as assistant do some simple work like play song for you, telling you story etc. for solving those problem we need someone or someting which will tell you about your schedule and let you know about other's schedule. If you know about others schedule Then you will know when they will be free so you wouldn't have to wait for someone for a long time And if you know when someone will be free then you can ask for a schedule on his free time. if an android app can work as an assistant to manage your regular schedule then it will minimize those problem. when you need someone to manage your regular schedule it is obvious there will be some mistake and there is only few people who can afford an assistant for maintaining his regular schedule But when a app can do the same work with more accuracy then you don't need to pay somone else. After that a question arised that why you will let everyone know about your schedule. I started to find a solution for this problem and I got the the way in which you can specify with whom you are sharing your schedule. So finally this app will help to you overcome those problem. it will save your time and money

1.3 Aims and Objectives

The aim of this experiment was to use an android application as our assistant which can assist to manage our regular schedule. The application will help you to interact with your clients, collegues and friends.

The core objective which have been designated as fundamental to the project are:

1. Apply a sign up form for collecting the data of a user
2. Varify the login details of the user
3. Preserve the login information for further execution
4. Preserve the schedule information inside an specific database
5. Examine the relationship between the user and others.
6. Notify the user about the schedule and friend request
7. Notify the users about their schedule

1.4 Expected Outcome

1. An android application which will be able to perform the task written below.

2. Making schedule for the user.
3. Showing your friend's schedule.
4. Showing your schedule to your friends
5. Searching for the new friends
6. Making friendship with other user
7. Requesting for a schedule
8. Fixing schedule with your friends

-

Chapter 2

PROJECT REQUIREMENTS

2.1 user requirements

The mobile application is mainly objected to provide user a schedule management application. This application not only keep track of user schedule , but also help user to keep track on other's schedule. This application will give you the oppurtunity to make friends whom schedule you need to track.

fig 2.1 shows all the functionality that can be performed by user

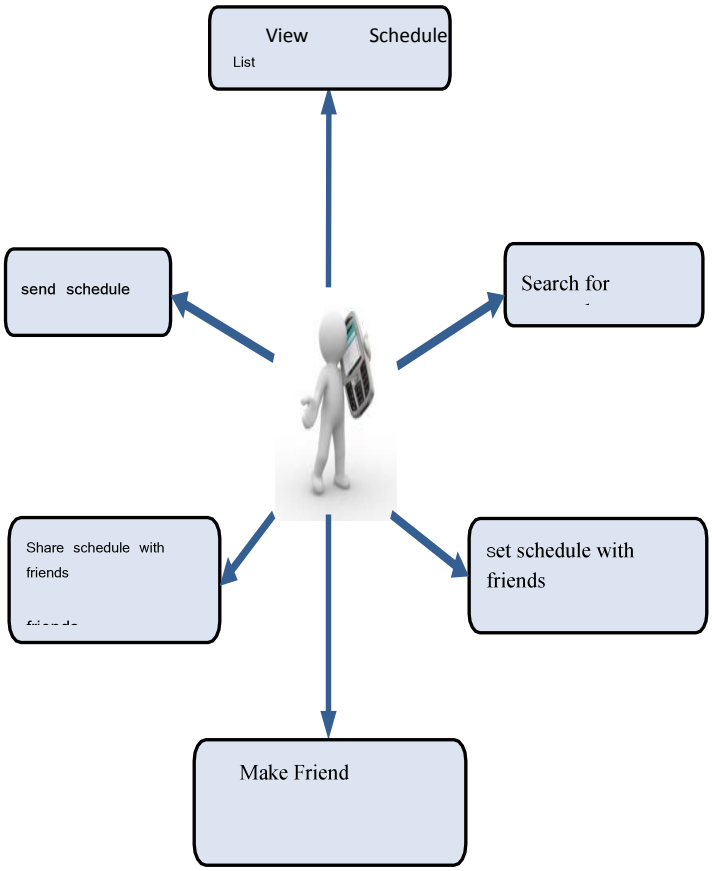


Figure 2.1 Pmate Use Case Diagram

2.2 View schedule

This feature wiill give the opportutiny to the user that the user can look up on his own schedule. The application will get the date from the phone and it will show all the schedule on that date.

Figure 2.2 shows the functional flow in application to display schedule list.

2.3 Search People

This application allows the user to search for others people. you can search your friends and collegues through this app. First of all you have to type their username or name then hit the search button and you will find all the people according to your search name

Figure 2.3 shows the search people flow.

2.4 Make Friend

This application allows to make friend. you can add your friend through this app. First of all you have to search for them by using their username or name. you will find all the people according to your search name. then you can send them friend request

Figure 2.4 shows Make friends flow.

2.5 Send Schedule Request

This application allows to Request other users for their Schedule. First of all you have to search for them by using their username or name. you will find all the people according to your search name. then you can send them request for a schedule.

Figure 2.5 shows send schedule request flow.

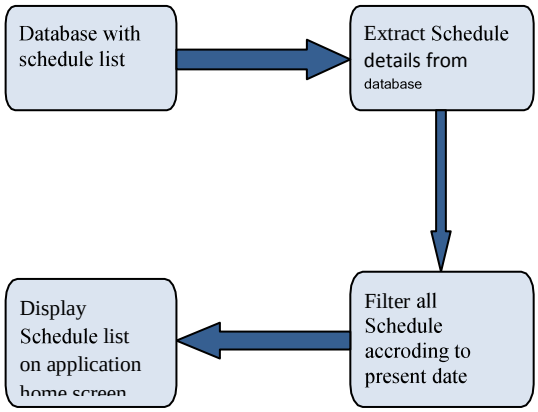


Figure 2.2 Display Schedule List

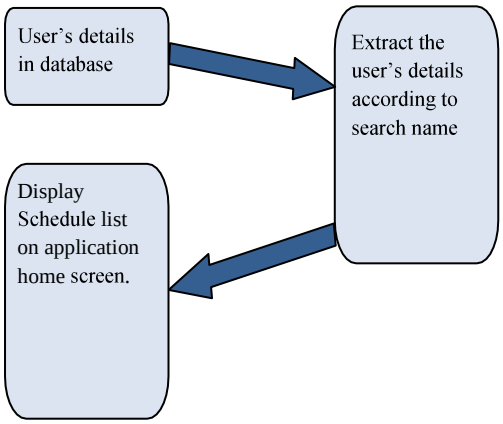


Figure 2.3 Search People

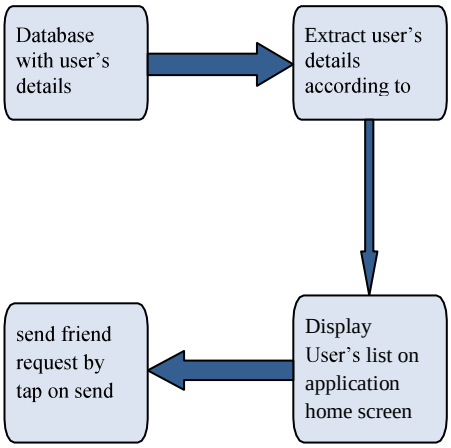


Figure 2.4 Make Friend

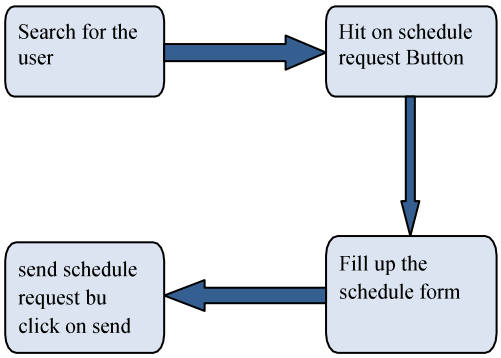


Figure 2.5 Send Schedule Request

2.6 Share Schedule with Friends

This application allows the user to share his schedule with their friends. First of all you have to fill up schedule form then you have fill the checkbox that will make sure that you want share your schedule with friends.

fig 2.6 shows share schedule with friends flow.

2.7 Set Schedule With Friends

This feature wiill give the opportutiny to the user that the user fix schedule with other user. One user can send schedule request to another user. If the user accept your request, this app will set a schedule on your schedule list.

fig 2.7 shows share schedule with friends flow.

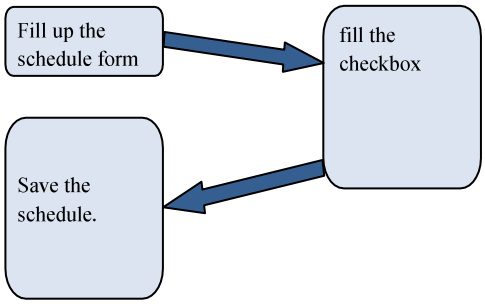


Figure 2.6 Share Schedule with friends

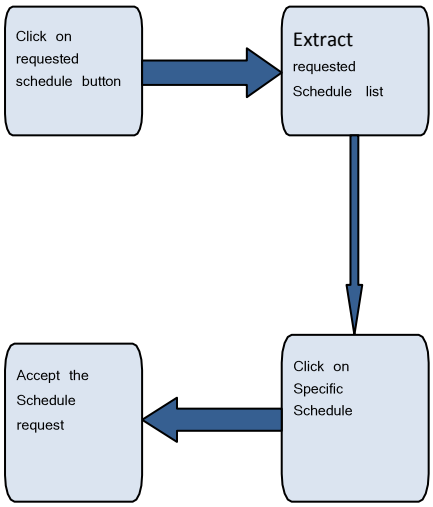


Figure 2.7 Set Schedule with friends

2.8 System Requirements

An android application can ideally be developed on any of the following platforms:

1. Windows
2. OS X
3. Linux

2.9 Software requirements

1. Java Development Kit (JDK) 7 or later version
2. Android SDK
3. Android studio

2.10 Hardware requirements

1. 2GB Ram
2. 400 MB hard disk space
3. At least 1 GB for Android SDK, emulator system images, and caches
4. 1280 x 800 minimum screen resolution

Chapter 3

ANDROID FRAMEWORK AND SOFTWARE ENVIRONMENT

Android is a platform architected in the form of a software stack. The stack components consist of applications, application frameworks, run-time environment, libraries and Linux kernel [1]. The operating system is developed by Google and is made open source so that users can extend the functionality and usefulness of applications. The android applications are developed in java using android software development kit [2].

An android application is built of different types of components. Each type serves a distinct purpose and has a distinct lifecycle that defines how the component is created and destroyed. The application components are activities, services, broadcast receivers, content providers. These components are loosely coupled by the application manifest file AndroidManifest.xml. This file has all the essential information about each component of the application and how they interact [3].

3.1 Components in Android Application

Components used to build "Pmate"application are:

1. Activities
2. Fragments
3. Services
4.

- **Activities**

An activity in android represents a single screen with which the user interacts. A single application can contain several activities that are bound together. Among these several activities only can be the main activity which handles the main UI functionality when application is launched. Though an application can have several activities, only one can be in active state at a time. Each time when a new activity starts previous activity is stopped but the data is preserved [4].

The First activity in "Pmate" is named as "Splash_Activity". The splash activity is declared as following in AndroidManifest.xml. Intent filter in main activity describes the operation to be performed for an activity and type of intents to be received.

```
<activity android:name=".Splash_Activity">
    <intent-filter>
        <action
android:name="android.intent.action.MAIN" />

        <category
android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Other Activity in "Pmate" are named as SignupActivity, ProfileActivity2, AllProductActivity, PAllproductActivity , SAllProductactivity, RequestedScheduleActivity, SentScheduleActivity, ShowScheduleActivity, ShowSchedule_Activity are declared as following in AndroidManifest.xml.

All the activities of a running application are managed in an activity stack and every activity goes through series of states. The state of each Activity is determined by its position on the activity stack. When a new activity is started, it becomes active and is placed on the top of the stack. The previous activity always remains below it in the stack, and will not come to the foreground again until user navigates back using the Back button so the next activity down on the stack moves up and becomes active [4].

An activity transition through four states while moving in and out of stack at the time of creation and destruction [4]:

Active or running: An activity in the foreground of the screen (at the top of the stack) receiving user input.

Paused: An activity is visible but will not have focus.

Stopped: A stopped activity is not visible to user, but will remain in memory retaining all state information.

Inactive: An activity that is killed or not yet launched, it's inactive. An inactive activity does not exist on activity stack.

- **Services**

Services does not require user interface. It performs operation with long run-time in the background. A service does not have an independent thread, it uses of the main thread from hosting process. For example we can download some files or play music in background while getting involved in some other application [5].

“Pmate” uses one service components “Myservice” declared as following in AndroidManifest.xml.

```
<service android:name=".MyService" />
```

In the above code snippet attribute “android:name” indicates the name of service subclass that implements service
Typically a service can be in one of following two states [5]:

Started: A service is started when startService() is called by an application component, such as an activity. A service can run in the background indefinitely, even if the component that started service is destroyed. It does not returns any results to caller, thus once a service has done its task it stops itself.

Bound: An application component binds to a service by calling method bindService(). A bound service allows components to interact with the service, send requests, and get results within same or different processes. A service runs as long as application component is bound to it and gets destroyed once it is unbounded.

There are some call back methods which should be overridden depending upon the implementation, to handle the service lifecycle. Some of the important call back

methods in service are onStartCommand(), onBind(), onCreate(), onDestroy() [5].

Figure 3.1 shows the lifecycle of a service.

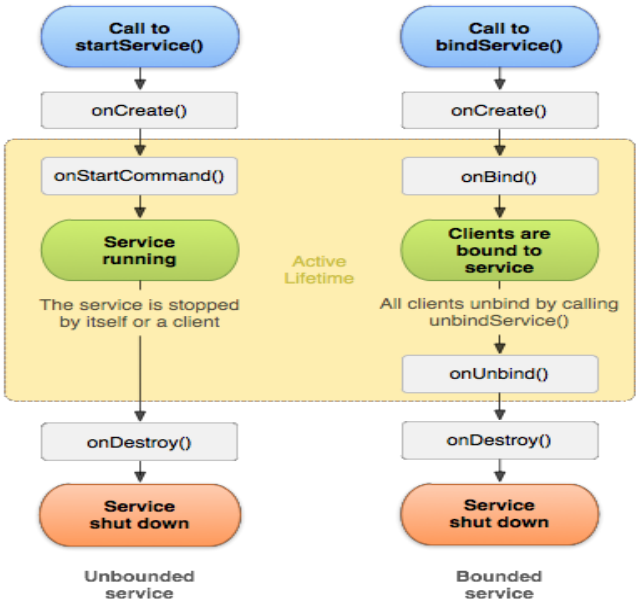


Figure 3.1 Android Service Life Cycle [5]

3.2 Setup for Android Application Development

3.2.1 Prerequisite installations

- **JDK Software**

A Java Development Kit (JDK) is an environment for writing Java applications. It consists of tools that developers need to compile, debug and run Java applications. One can download the appropriate setup version of JDK from Oracle's website and follow the instructions of setup wizard to install and configure JDK. And then, edit the JAVA_HOME from environment variables to point to directory where the JDK software is located.

- **Android SDK**

An Android SDK is a software development kit that allows developers to create applications on the Android platform. It includes sample source code, development tools, an emulator, debugger, and required libraries to build Android applications. We can download and install android SDK from android's official website as an additional package that comes with android studio or the stand- alone SDK package [6].

- **Android Studio IDE**

We can download the android studio IDE from android developer's website by clicking on download link. After the .exe file is downloaded click on the file and follow instructions on the setup wizard to complete installation [6].

Android Studio is an official integrated development environment supported by Google for Android application development. It provides variety of useful features which makes android development much easier. Some these features are
– easy layout editing by dragging and dropping UI components, providing code suggestions while writing the code, have sample code templates to build common application features, able to generate multiple apk file generation associated with same project and gradle-based build automation system [7].
Figure 3.4 Summarizes all the basic steps and development phases involved while developing an android application.

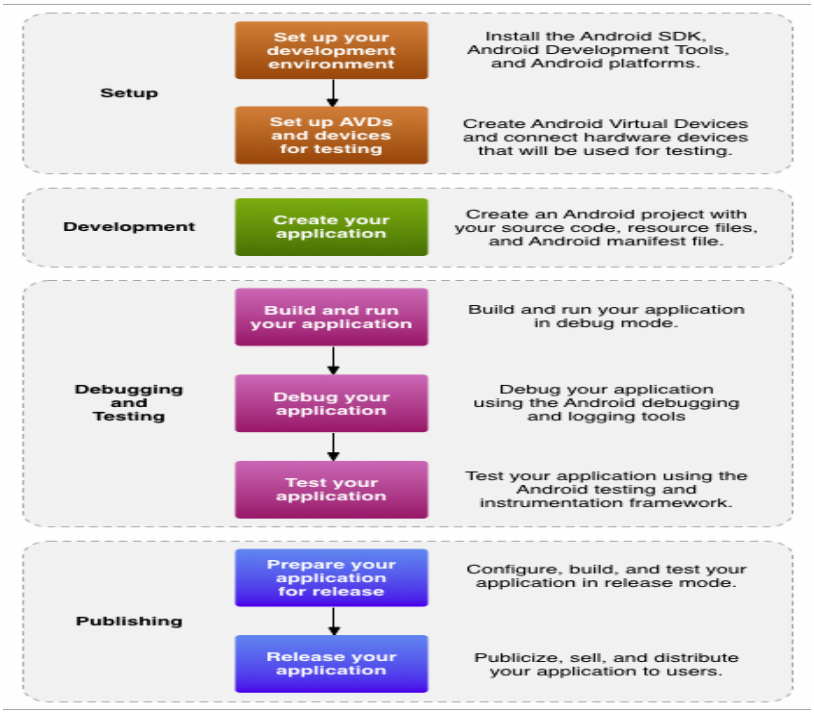


Figure 3.2 Development Phases in an Android Application [8]

Chapter 4

FEATURES IMPLEMENTATION

4.1 Sign Up Information:

The application installion requires user's log in details, so user needs to registration first with user's username, name, email, phone number and few more information including.

Figure 4.1 shows the signup screen while click on signup button.

To extract this data from the user interface Asyntask is used. doInbackground() method is used which ensures that UI stays responsive. Once the background thread finishes its task onPostExecute() method is called which publishes the result in main UI thread. All the data entered by the user saved in an external database and to do this put all the data into some variables named username, name, email, phone, country, organization. then save the data as a Backgroudtask.In Backgroundtask make an url where the database exist. After that made an url connection using httpurlconnection. Once the connection are set used outputstream and bufferedwritter to save the data inside the database.

Following is the code snippet for signup screen:

```
String user=username.getText().toString();
String name=uname.getText().toString();
String email=uemail.getText().toString();
String phone=uphone.getText().toString();
String country=ucountry.getText().toString();
String org=uorganization.getText().toString();
String pass=upass.getText().toString();

class Backgroundtask extends AsyncTask<String,Void,String> {

 String Register_url;

      @Override
      protected void onPreExecute()
      {
         Register_url="http://safetech.ga/User_Registration/registration.php";

            }
```

```
try {
    URL url=new URL(Register_url);
    HttpURLConnection
httpURLConnection=(HttpURLConnection)
url.openConnection();
    httpURLConnection.setRequestMethod("POST");
    httpURLConnection.setDoOutput(true);
    OutputStream outputStream=
httpURLConnection.getOutputStream();
    BufferedWriter bufferedWriter=new
BufferedWriter(new
OutputStreamWriter(outputStream,"UTF-8"));
    String data_string=URLEncoder.encode("user","UTF-
8")+ "=" +URLEncoder.encode(user,"UTF-8")+"&"+
            URLEncoder.encode("name","UTF-8")+ "="
+URLEncoder.encode(name,"UTF-8")+"&"+
            URLEncoder.encode("email","UTF-8")+ "="
+URLEncoder.encode(email,"UTF-8")+"&"+
            URLEncoder.encode("phone","UTF-8")+ "="
+URLEncoder.encode(phone,"UTF-8")
            +"&"+
            URLEncoder.encode("country","UTF-8")+ "="
+URLEncoder.encode(country,"UTF-8")+"&"+
            URLEncoder.encode("organization","UTF-8")+
"=" +URLEncoder.encode(organization,"UTF-8")
            +"&"+
            URLEncoder.encode("password","UTF-8")+ "="
+URLEncoder.encode(password,"UTF-8")+"&"+
            URLEncoder.encode("friend","UTF-8")+ "="
+URLEncoder.encode(friend,"UTF-8")+"&"+
    bufferedWriter.write(data_string);
    bufferedWriter.flush();
    bufferedWriter.close();
    outputStream.close();
}
```

4.2 Login details:

The application requires login infromation before user gets the features. User needs to verify user's account before using the app. User needs2 to provide username and password.

Figure 4.2 shows the login screen popped up while installing application.

The data entered in login page needs to check for varifying the identity of the user. first of all username and password put into two variable. then send the data as a

Backgroudtask.In Backgroundtask make an url where the database exist. After that made an url connection using httpurlconnection. Once the connection are set used outputstream and bufferedwritter to send the data inside the database. To do this used Asynctask class, doInbackground() methood and onPostExecute() method.

Once you send the username and password you need to get the verfication result. you can get back the verification result using InputStream and BufferedReader.
Following is the code snippet for login screen:

```
InputStream inputStream=httpURLConnection.getInputStream();
BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream,"iso-8859-1"));
String response="";
String Line="";

while((Line=bufferedReader.readLine())!=null)
{
   response+=Line;

}
bufferedReader.close();
inputStream.close();
httpURLConnection.disconnect();

return response;
```

when the get back result is true you need to take the user into another activity and store the login details till the user logout or if the get back result is false you have to tell the user about this error.

```
try{
   if(result.equals("Login successfully"))
   {
      store();

      ProfileActivity2();
   }
   else
   {

Toast.makeText(getApplicationContext(),result,Toast.LENGTH_LONG).show();
   }
```

Following is the code snippet for login screen.

Store the login data inside the app using sharedpreference and editors. It will remember about your login status. Once you store the login status, you don't need login agiain till you logout from the application.

Following is the code snippet for login screen.

```
SharedPreferences  prefers =
getSharedPreferences("com.example.shuvo.pmate2_login_status;",MODE_PRIVATE);
SharedPreferences.Editor editor=prefers.edit();
editor.putString("login_status","on");
editor.commit();
```

4.3 Profile Activity information:

A successful login will redirect the user into the profile activity class where the user will get all the feature of this application. In this profile actvity class there is some button for different feature.Each button do one or more specific work. Few button will redirect user in another activity.

The Button name was declared as Logout , Schedule, Friend, Add, Requested and Sent_Rqst.

Figure 4.3 shows the profile activity screen while login successfully.

4.4 Schedule List

Pmate application allows user to view user's schedule list. when user's click on Schedule Button it will shows the schedule list in a list view. From the list view user will be able to know about the details of the schedule. It will show the schedule list for the present date.

Figure 4.4 shows the schedule list while ask for schedule.

Once a user clicked on schedule button the application bring all the data from enternal database which was created to store all the schedule data for the user. One phone can

be used by different user so at the time of fetching data the application will check the user identity by using their username. The application will check the schedule date at the time of fetching data by using device date because the application will show the schedule list only for the present date.

Following is the code snippet for fetching schedule list.

```
Cursor res= mydb.getallData(User);
if(res.getCount()==0)
{
    showallmessage("Error","Nothing Found");
    return;

}
StringBuffer buffer=new StringBuffer();
while(res.moveToNext()){
    buffer.append("Id :"+ res.getString(0)+"\n");
    buffer.append("Schedule_Name :"+
res.getString(1)+"\n");
    buffer.append("Start_Time :"+
res.getString(2)+"\n");
    buffer.append("End_Time :"+ res.getString(3)+"\n");
    buffer.append("Date :"+ res.getString(4)+"\n");
    buffer.append("Person :"+ res.getString(5)+"\n");
    buffer.append("Location :"+ res.getString(6)+"\n");
```

For fetching the data of schedule list Databasehelper class was used in this application.

Following is the code snippet of Databasehelper class.

```
public Cursor getallData(String user)

{
    Calendar calendar= Calendar.getInstance();
    String currentdate=
DateFormat.getDateInstance(DateFormat.DATE_FIELD).forma
t(calendar.getTime());
    SQLiteDatabase db=this.getWritableDatabase();
    Cursor res= db.rawQuery("Select * from
newscheduleTable where Userid= '" + user + "'  and
Date= '" + currentdate + "' " ,null);

    return res;
}
```

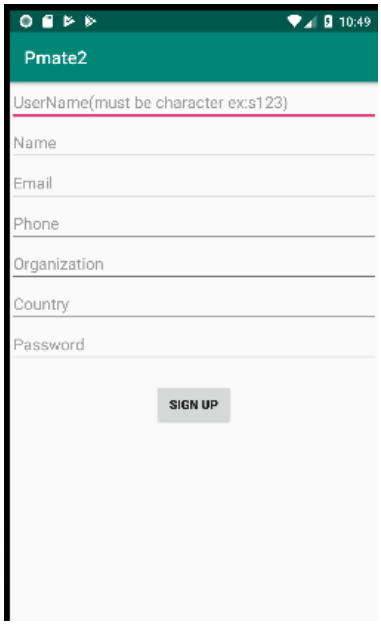


Figure 4.1 Application Installation – Signup Screen

Figure 4.2 Application Installation – Login Screen

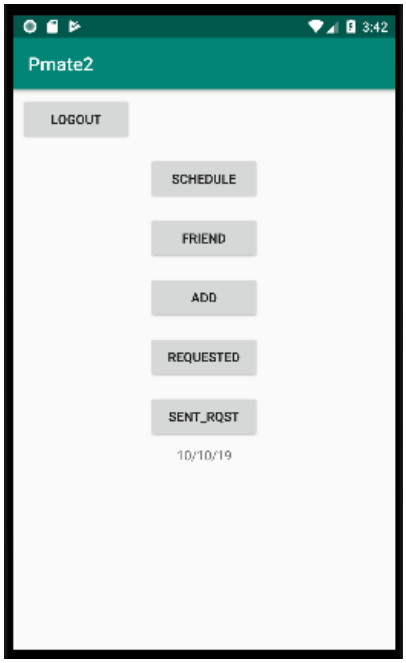


Figure 4.3 Profile Activity Screen

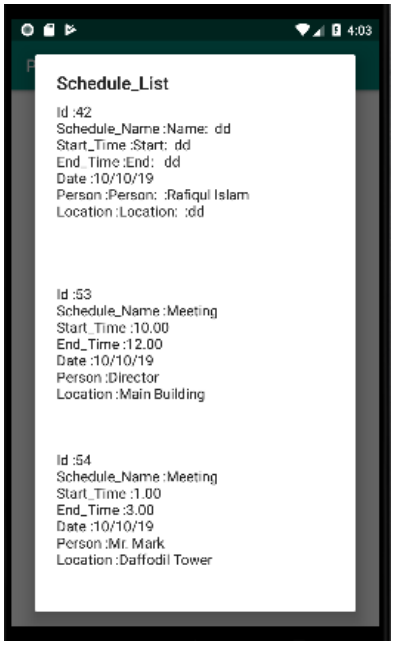


Figure 4.4 Schedule List Screen

4.5 Friends Information:

Once a user clicked on friend button, the application will open another activity for the user. By using this feature user can get information about user's friend and other users. This activity contains a search box named search and a button named friendlist. Figure 4.5 shows the aplication screen while click on Friend Button.

4.6 Search Friend

In fig-4.5 search box will help the user to find others people and friend. The user need to put the username or name in the search box and click on the search button. Once user click on the search button the application will give the user details for the name which user entered in the searchbox.
Figure 4.6 shows the user details while click on search Button.

In fig-4.6 shows the search result for the others people. This search result will give the user few information about the searched people like name, email and organization. The search result is showing on a list view so that you can get many people's information for the name you searched and you can select one person. For featuring this the application used a java class named PAllProductActivity which extends ListActivity. The application take the data from the search box. To extract this data from the user interface Asyntask is used. doInbackground() method is used which ensures that UI stays responsive. Once the background thread finishes its task onPostExecute() method is called which publishes the result in main UI thread as a list Item.

4.7 In Friend list

Once the user select a specific person it will redirect the user in another activity which is actually the person's profile whom user select. If the selected person is in user's friendlist then it will redirect the user in inmyfriendlist activity with selected person's information, If the selected person is in user's friendrequest list then it will redirect the user in inmyfriendreqlist activity with selected person's information, If the user is in that people's friendrequest list then it will redirect the user in inotherfriendlist

activity with selected person's information, Otherwise it will redirect the user in Outsidefriendlist activity with selected person's information.

Figure 4.7 shows the Inmyfriendlistactivity while the searched people is in user's friend list.

Inmyrirendlistactivity user will get three features. user can remove that people from friendlist by clicking remove button, user can see the schedule list of that person which is shared with friend by clicking on schedule button or can make an schedule request by clicking request Button.

4.8 Friends Schedule

Once user click on friend schedule button it will take the friend's username from inmyfriendlistactivity. then the application will retrive friend's schedule data from friends schedule database and the schedule data will arrive as a list view.

Figure 4.8 shows the friend's schedule list.

In fig 4.8 shows the schedule list of user's specific friend which shared by friend. The schedule list will give the user few information about user's freind schedule like schedule name, start time, end time, date etc. The schedule list is showing on a list view so that you can get information about more than one schedule of your friend. For featuring this the application used a java class named SAllProductActivity which extends ListActivity. The application take the friend's information from inmyfriendlistactivity . To extract this data from the user interface Asyntask is used. doInbackground() method is used which ensures that UI stays responsive. pDialog was used for showing a message on onPreExecute() method. Once the background thread finishes its task onPostExecute() method is called which publishes the result in main UI thread as a list Item.

Figure 4.5 Friend Screen

Figure 4.6 search Details

-

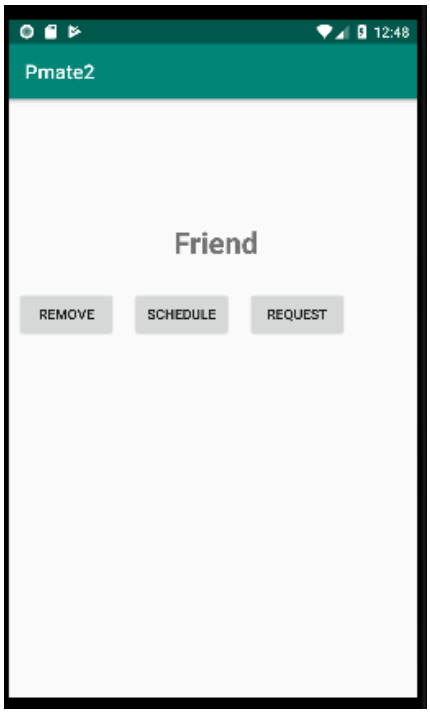


Figure 4.7 Friernd's profile

Figure 4.8 Friend's Schedule list

4.9 Schedule request

There is another feature in inmyfriendlist activity for the user. An User can make a request to his friend for an schedule. Once a user click on Request button, the application will redirect the user in another activity named ScheduleReqActivity. In this activity user will get a schedule req form. User has to fill up the form. The forrm is about to take few information from the user about the schedule. This form need a name, start time, end time, date and location.

Figure 4.9 shows the schedule request form.

Once user fill up this form and click on submit button, The information of this page will be taken. This information will store as a schedule request to user's friend schedule request list and also as a sent schedule request to user's own sent schedule request list.

for doing this, application needs the table name of user's friend's schedule request list and user's sent schedule request list. By using username of user's and friend's, the application will find out the table name.

This task will be done as a Bacground Task. To extract this data from the user interface Asyntask is used. doInbackground() method is used which ensures that UI stays responsive. In doInbackground() method a httpurlconnection was made by using an specific valid url link. OutputStream and BufferedWritter was used to send the data from application to database.

4.10 User and Other User Relation

This application can identify the relation between the user and other users. If a user search about any other users, this application will identify the relation between the user and other users. the realation could be friend, sent request, recieved request or nothing. The reason of identifying relationship is important to get next features like if you are friend then you can remove other person from your friendlist and you can see his schedule as a friend, if other person sent you friend request then you need to get that option to accept or ignore his request.

For doing this work , this application check the searched user's username in user's friend table, friendreqtable and friendreqbytable.

This task will be done as a Bacground Task. To extract this data from the user interface Asyntask is used. doInbackground() method is used which ensures that UI stays responsive. this task needs an url connection.

Following is the code snippet for identifying user and other user realtion.

```
List<NameValuePair> params = new ArrayList<NameValuePair>();
params.add(new BasicNameValuePair("name", s1));
params.add(new BasicNameValuePair("friendtable", s4));
params.add(new BasicNameValuePair("friendreqtable", s5));
params.add(new BasicNameValuePair("user", user));
params.add(new BasicNameValuePair("friendreqbytable", s6));
// getting JSON string from URL
JSONObject json = jParser.makeHttpRequest(url_all_products, "POST", params);
```

4.11 Unknown User's profile

If a user search about a person who isn't connected with the user then it will redirect the user in another activity named outsidefriedlistactivity. This activity will give three features to the user. user can send friend request, user can see the schedule of that user which shared as public, user can make a schedule request.

If User send friend request then this data will store in other user's database table named friendreqby with user's few information so that the other user can get that notification about this friend request.

Schedule button will work as similiar as friend's schedule button but this time it will show the schedule which is only shared with public.and the last one is user can make an schedule request by clicking on Request button. This feature will work as same as friend's schedule request button. This request will be shown to the other user as a schedule request.

Figure 4.10 shows the Unknow user's profile.

4.12 Friendlist details

In friend Activity there was two feature one of them was search for people and another was user friend list. By clicking on friendlist button user will get user's friendlist that user added as a friend.

This friend list will come as a list view. this list view activity used setOnItemClicklistner so that user can select one specific friend.

Figure 4.11 shows the friendlist.

Friend list data was retrived from the user's database table name friendlist using friend's username. Data was retrived as a jsonarray from an external database using php.

Once user select a specific friend from friendlist it will redirect user inmyfriendlist activity. Then the user can do the same described before.

for making this task complete, apllication used setOnItemClicklistner.

Following is the code snippet for SetOnItemClicklistner.

```
lv.setOnItemClickListener(new OnItemClickListener() {

   @Override
   public void onItemClick(AdapterView<?> parent, View view,
                     int position, long id) {
      // getting values from selected ListItem
      String name = ((TextView) view.findViewById(R.id.name)).getText()
            .toString();
      String org = ((TextView) view.findViewById(R.id.price)).getText()
            .toString();
      String User= ((TextView) view.findViewById(R.id.pid)).getText()
            .toString();

      Intent in = new Intent(getApplicationContext(),
            Friendlistfrnd_Activity.class);

      in.putExtra("name", name);
      in.putExtra("org",org);
      in.putExtra("User",User);
      in.putExtra("fuser",user);
      startActivityForResult(in, 100);
   }
});
```

4.13 Friend Request Feature

This Application is used a friend request feature. Through this feature an user can get the information about friend request. This feature will come once a user click on Friend_Reqt Button. It will show those user's information who sent friend request to the user. This feature used a java class named FriendReqlistActivity. To retrive the data from user friend request list user's username is needed. Asynctask method was used to extract the data from user interface. doInbackgroud() method was used to retrive the data from database and on postExecute() method the data was sent to view as a list view with SetOnItemClickListener.

Figure 4.13 shows the Friend Request list feature.

Once user choose an specific request, It will take the user in another activity where the user will get two option. One option is for accept the request and another is for ignore the request.If user

accept the request then the the request will be deleted from the request list and this user information will be added user's friend list. In the other this data will be removed from that user's request list and user's information will be added in that user's friend list. If the user Ignored the request then this information will be deleted from both user's request list.

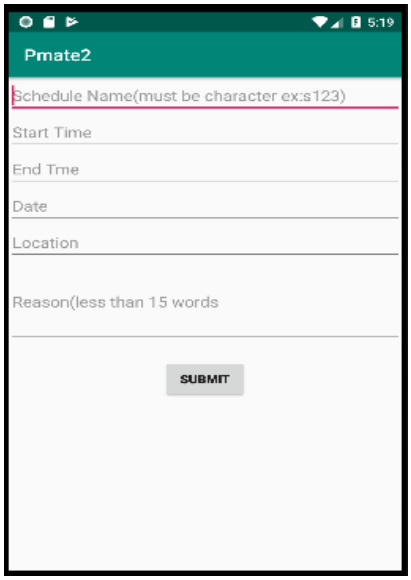


Figure 4.9 Schedule request form

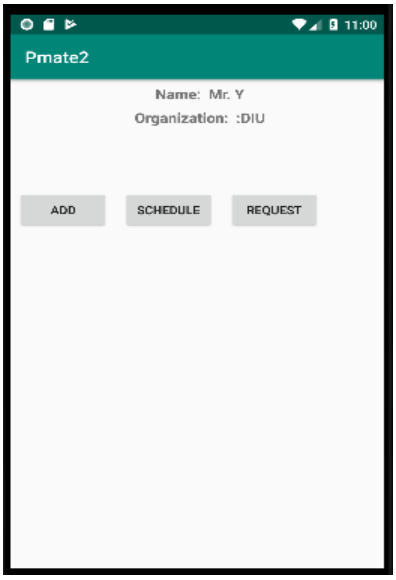


Figure 4.10 Unknown User's Profile

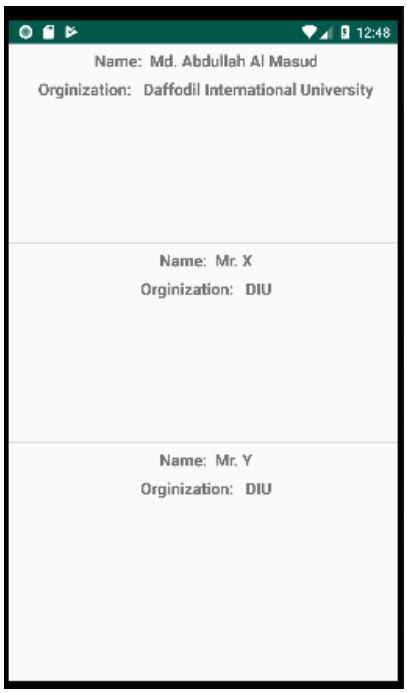


Figure 4.11 Friendlist Feature

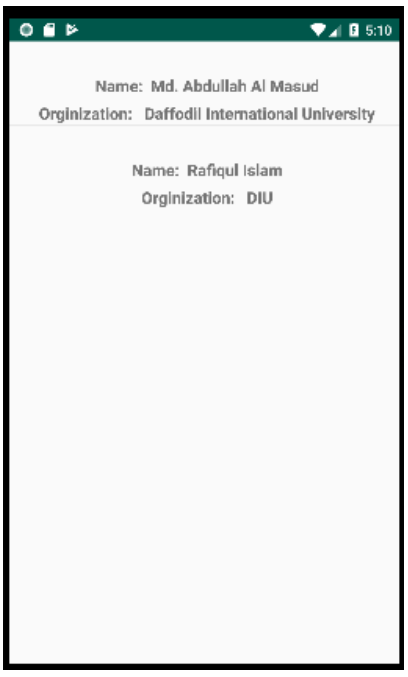


Figure 4.13 Friend request list

4.14 Add Schedule

By using this feature an user can set his own schedule. There is actually a schedule form that user need to fill up for every schedule with start time and date. There is also two privacy setting

one is for public and another is for friend. An user can set that who will be able to see his schedule. If the user select public for his schedule then everyone will be able to see user's schedule or if The user selct friend then only the people from user's friendlis will get user's schedule and If user's leave that blank then it will save only for the user, only user will see that schedule.

If the user choose public option then the data will save both in external database and internal database. In external database the data will store in public table and in internal database it will store in schedule table. Appplication used Databasehelper extends with SQLiteOpenhelpre. Data is stored in SQliteDatabase. SQLiteDatabase used to store data in entarnal database. First of all created a table by using sqlitedatabase with column name. Then the data was inserted using insertdata method. If the user leave this option blank and click on submit button then the data will save only in internal storage.

Following is the code snippet for Store data in internal database.

```
public void onCreate(SQLiteDatabase db) {
    String createtable= "CREATE TABLE "  + TABLE_NAME +
"(ID INTEGER PRIMARY KEY AUTOINCREMENT,Name TEXT, Start
TEXT, ENDT TEXT, DATE TEXT, PERSON TEXT, LOCATION TEXT,
USERID TEXT )";
    db.execSQL(createtable);


}
public boolean insertdata(String name , String start,
String endt , String Date, String person, String
location, String userid)
{
    SQLiteDatabase db=this.getWritableDatabase();
    ContentValues contentValues= new ContentValues();
    contentValues.put(Col2,name);
    contentValues.put(Col3,start);
    contentValues.put(Col4,endt);
    contentValues.put(Col5,Date);
    contentValues.put(Col6,person);
    contentValues.put(Col7,location);
    contentValues.put(Col8,userid);
    long result=
db.insert(TABLE_NAME,null,contentValues);
}
```

4.15 Requested Schedule

This feature will notify the user about his requested schedule. If any user sent a schedule request to the user then it will store in user's requested schedule list. Once user clicked on Requested button It will retrive all the requested schedule data and this data will be shown as a view list. From this list, the user will know who sent the request and also about time, date and etc.

The product of this list view with setOnItemClickListener. Once user click one product it will take the user in another activity where user will get two option. one is for accept the request and another is Ignore the request. If user aceept the Schedule

request then this schedule will automatically save as user's schedule in user's schedule list and willl notify the other user about this acceptance. If user ignored the request then it will remove the request from user's requested schedule list and it will notify the user who sent that schedule request abot this ignorance.

For doing this task, Asynctask was used to extract the data from User interface. Data was stored as a Backgroud task.

4.16 Sent Request Information

There is another Feature added in this application. There is a button in main screen name sent_reqt. This button was designed to get the information about sent request by user. when a user send a schedule request to other user it will store in user's sent schedule request table.

Once user clicked on sent_reqt button the sent schedule request information will be shown in a list view. It will show the current status of the schedule also wheras it's accepted or not.

Chapter 5

Implementation and Testing

5.1 Database implementation

In this project every feature is depend on data so it was very important to make an useful database. Database store all the data which the application needs. In this application only one table was created inside the database named registration table But when a user will sign up in this application, it will create seven more table for the user named Friend, FriendReq, FriendReqBy, ScheduleAll, ScheduleFrnd, ScheduleReq, ScheduleReqBy. Every table is essential for different task.

Registration Table

This table is used to store user's information at the time of sign up. This data will help a user to login to this application. This table will store user's all the information like username, name, email, password etc. If user's information is needed anywhere in this application, this table will provide the information

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
ID	Int(11)	User Id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data collectors for the site
User	Varchar(50)	Username		No	This field is for user username.

					Username is essential for a user.
Column Name	**Type**	**Descriptive name**	**Index Column**	**Allows Nulls**	**Description**
Name	Varchar(50)	User Name		No	This field is the name of the user which will desplayed to others.
Email	Varchar(50)	User Email		No	This field is Email of the user.
Phone	Varchar(20)	Phone Number		Yes	This field is the phone number of the user
Counrty	Varchar(50)	User's Country		Yes	This field is the country name of the user.
Organization	Varchar(60)	User's Organization		Yes	This field is for the user's Organization where he works
Password	Varchar(20)	User's Password		No	This field is the password of user's which is needed to log in.

Table name Registration

Friend table

This table store the data of user's friend Including username, name and organization. When a user ask for his frindlist, the data will retrive from this table. A specific friend table was used for every user.

Table name Friend table

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
id	Int(11)	Friend id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data collectors for the site
Column Name	**Type**	**Descriptive name**	**Index Column**	**Allows Nulls**	**Description**
User	Varchar(50)	Username		No	This field is for user username. Username is essential for a user.

Name	Varchar(50)	User's Name		No	This field is the name of the user which will desplayed to others.
Organization	Varchar(60)	User's Organization		No	This field is for the user's Organization where he works

FriendReq Table

This table stored the data of those people whom the user sent friend request including that person's username, name and organization. When the user will ask those people's information whom he sent friend request, the application will give that information from this table.

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
id	Int(11)	Friend id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data collectors for the site
User	Varchar(50)	Username		No	This field is for

					user username. Username is essential for a user.
Name	Varchar(50)	User's Name		No	This field is the name of the user which will desplayed to others.
Organization	Varchar(60)	User's Organization		No	This field is for the user's Organization where he works

Table name- Friend Req table

Friend Req by Table

This table stored the data of those people who sent friend request to the user including that person's username, name and organization. When the user will ask about those people's information who sent the friend request to user, the application will give that information from this table.

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
id	Int(11)	Friend id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data

					collectors for the site
User	Varchar(50)	Username		No	This field is for user username. Username is essential for a user.
Name	Varchar(50)	User's Name		No	This field is the name of the user which will desplayed to others.
Organization	Varchar(60)	User's Organization		No	This field is for the user's Organization where he works

Table name- Friend Req by table.

Schedule All Table

This table stored the data of user's public schedule. If the user share a schedule publicly then the information about that schedule will store here. so that if a person ask about user's schedule who isn't in user's friendlist, the application will give him the schedule from this ScheduleAll table.

This table contains schedule id, schedule name, schedule start time, schedule end time, date, location , person and comment.

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
id	Int(11)	Friend id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data collectors for the site
Name	Varchar(50)	Schedule Name		Yes	This field is for Schedule Name. you can set any name for your schedule
Start	Varchar(10)	Schedule Start Time		No	This field is the Start time of schedule. you have to put the schedule start time.
End	Varchar(10)	Schedule End Time		Yes	This field is for the schedule end

					time. The end time of a schedule will store here.
Date	Varchar(15)	Schedule Date		No	This field is for the schedule Date. The Date of a schedule will store here.
Location	Varchar(50)	Schedule Location		Yes	This field is for the schedule Location. The Location of a schedule will store here.
Person	Varchar(50)	Other user		Yes	This field is for the name of other with whom the user have schedule.
Comment	Varchar(50)	Comment		Yes	This field is for store other's user Comment.

Table name-ScheduleAll table

Schedule Friend Table

This table stored the data of user's friend schedule. If the user share a schedule with friend then the information about that schedule will store here. so that if user's friend ask about user's schedule, the application will give him the schedule from this ScheduleAll table.

This table contains schedule id, schedule name, schedule start time, schedule end time, date, location , person and comment.

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
id	Int(11)	Friend id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data collectors for the site
Name	Varchar(50)	Schedule Name		Yes	This field is for Schedule Name. you can set any name for your schedule

Start	Varchar(10)	Schedule Start Time		No	This field is the Start time of schedule. you have to put the schedule start time.
End	Varchar(10)	Schedule End Time		Yes	This field is for the schedule end time. The end time of a schedule will store here.
Date	Varchar(15)	Schedule Date		No	This field is for the schedule Date. The Date of a schedule will store here.
Location	Varchar(50)	Schedule Location		Yes	This field is for the schedule Location. The Location of a schedule will store here.
Person	Varchar(50)	Other user		No	This field is for the name of other user with whom the user have schedule.
Comment	Varchar(50)	Comment		Yes	This field is for store other's user Comment.

Table name-Schedule Frnd table

ScheduleRequest Table

This table stored the data of user's requested schedule. If a user sent a schedule request to other user then the requested schedule information will store here. so that if user ask for those schedule information, the application will give that information from here. this table will also inform about the current status of the requested schedule.

This table contains schedule id, username, schedule name, schedule start time, schedule end time, date, location , person and comment.

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
id	Int(11)	Friend id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data collectors for the site
Username	Varchar(50)	User's Username		No	This field will store the username of the user to whom the schedule sent
Name	Varchar(50)	Schedule Name		Yes	This field is for Schedule Name. you can set any name for your schedule

Start	Varchar(10)	Schedule Start Time		No	This field is the Start time of schedule. you have to put the schedule start time.
End	Varchar(10)	Schedule End Time		Yes	This field is for the schedule end time. The end time of a schedule will store here.
Date	Varchar(15)	Schedule Date		No	This field is for the schedule Date. The Date of a schedule will store here.
Location	Varchar(50)	Schedule Location		Yes	This field is for the schedule Location. The Location of a schedule will store here.
Person	Varchar(50)	Other user		No	This field is for the name of other user with whom the user have schedule.
Status	Varchar(20)	Schedule		Yes	This field will store the requested schedule current status.

Table name ScheduleAll table

Schedule Req By Table

This table stored the data of requested schedule sent by other user to the user. If any friend of user sent schedule request to the user then that data will store. so that the user can requested schedule sent by his friend.

This table contains schedule id, Username, schedule name, schedule start time, schedule end time, date, location , person and comment.

Column Name	Type	Descriptive Name	Index Column	Allows Nulls	Description
id	Int(11)	Friend id	Unique Index	No	This field is a unique alphanumeric identifier for the site. This identifier should be used by all data collectors for the site
Username	Varchar(50)	Friend's Username		No	This field contain the friend's username who sent the request.
Name	Varchar(50)	Schedule Name		Yes	This field is for Schedule Name. you can set any name for your schedule

Start	Varchar(10)	Schedule Start Time		No	This field is the Start time of schedule. you have to put the schedule start time.
End	Varchar(10)	Schedule End Time		Yes	This field is for the schedule end time. The end time of a schedule will store here.
Date	Varchar(15)	Schedule Date		No	This field is for the schedule Date. The Date of a schedule will store here.
Location	Varchar(50)	Schedule Location		Yes	This field is for the schedule Location. The Location of a schedule will store here.
Person	Varchar(50)	Friend Name		No	This field is for the name of user's friend who sent the

					schedule request

Table name-Schedule Req by Table

5.2 Testing

For testing this application, I created four user's account in this application. This four account will interact with each other. By ding this, I will testify all the features of Pmate.

View Schedule:

Set few schedule for today Using one of the user's account. Then click on schedule button. It will show the schedule List.

Figure 5.1 shows the schedule list

Friend list Check

For this test we add some people in a user's friendlist. when the user click on friendlist button the list of his friend will come in a list view.

Fig 5.2 shows the Fireind list.

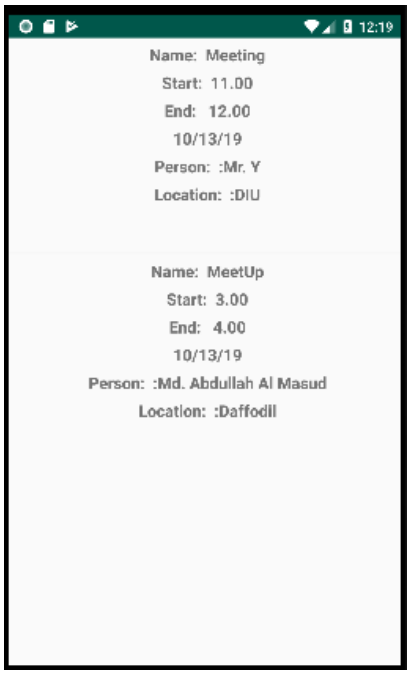


Figure 5.1 schedule view

Figure 5.2 Friend List

Send Friend Request

For this thee specific user was registered. Two of them will send a friend request to other user.

when a user sent friend request to other request then it will save in friend request list. so once the user click on Friend_reqt button the requested friend list will appear

Figure 5.3 show the Requested friend list.

Send Schedule Request

For doing this, two user was registered through this app name Rafiqul Islam and Md. Abdullah AL Masud. One of them will send schedule request to other user. so the shedule request will add in requested list, who get the request and other schedule request will add in sent request list, who sent the request.

Here Md. Rafiqul islam sent a schedule request to Md. Abdullah Al Masud. So it will add in Requested schedule list for Abdullah Al masud and for Rafiqul Islam it will add in sent Request list

Figure 5.4 shows the sent schedule request list

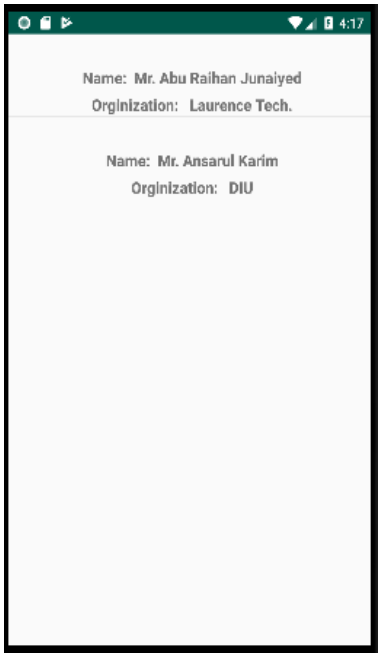


Figure 5.3 Requested Friend list

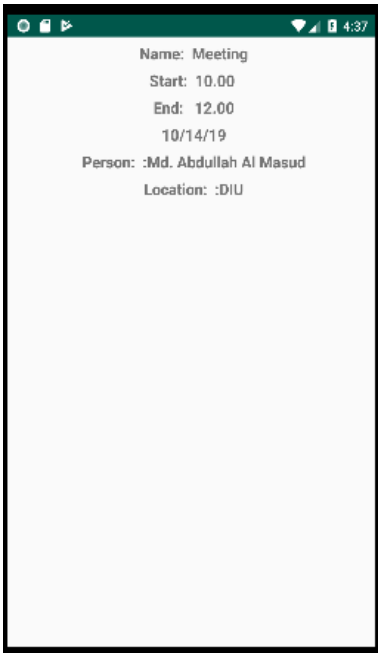


Figure 5.4 sent schedule request

Chapter 6

Future work And Conclusion

6.1 Fututre Work

This application can perfetcly handle user schedule. Along with this application can give you the opportunity to interact with your friend's schedule by sharing schedule with the user But i want to make it more useful. The idea is to add a new feature with this app which will remind the user few minutes before the schedule, This app will be able to send notification to the user. In future work the user interface will be more attractive and user friendly.

6.2 Conclusion

As a virtual assistant this app will do something different than existing virtual assitant And it is also more useful than other existing virtual assistant app. This application will take care of your schedule. It will you about your schedule to your friends and clients And you can also get the schedule information about your friends and clients.

This app also offer an important feature that you can make an schedule through this app with your friends and valuable client.

REFERENCES

[1] Techotopia. "An Overview of the Android Architecture." [Online].
Available: http://www.techotopia.com/index.php/An_Overview_of_the_Android_Architecture.
Accessed in October 2015.

[2] Wikipedia. "Android software development." [Online]. Available:
https://en.wikipedia.org/wiki/Android_software_development. Accessed in July 2015.

[3] Google Inc. "Application Fundamentals | Android Developers." [Online].
Available: http://developer.android.com/guide/components/fundamentals.html. Accessed in
July 2015.

[4] Google Inc. "Activities | Android Developers." [Online]. Available:
http://developer.android.com/guide/components/activities.html. Accessed in July 2015.

[5] Google Inc. "Services | Android Developers." [Online].
Available: http://developer.android.com/guide/components/services.html. Accessed in October
2015.

[6] Google Inc. "Installing the Android SDK | Android Developers." [Online].
Available: http://developer.android.com/sdk/installing/index.html. Accessed in July
2015.

[7] Google Inc. "Android Studio Overview | Android Developers."
[Online]. Available:http://developer.android.com/tools/studio/index.html. Accessed in July 2015.

[8] Google Inc. "Developer Workflow | Android Developers."
[Online]. Available:http://developer.android.com/tools/workflow/index.html. Accessed in
November 2015.

Pmate__final_WITH

ORIGINALITY REPORT

22%	14%	3%	21%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Southern New Hampshire University - Continuing Education Student Paper	4%
2	Submitted to Siddaganga Institute of Technology Student Paper	3%
3	Submitted to William S. Hart Union High School District Student Paper	2%
4	Submitted to Daffodil International University Student Paper	1%
5	Submitted to Nottingham Trent University Student Paper	1%
6	Submitted to University of Moratuwa Student Paper	1%
7	Submitted to University of Greenwich Student Paper	1%
8	www.theappguruz.com Internet Source	1%

9	Submitted to University of London External System Student Paper	1 %
10	Submitted to Charotar University of Science And Technology Student Paper	< 1 %
11	Submitted to Middle East College of Information Technology Student Paper	< 1 %
12	Submitted to University of Northampton Student Paper	< 1 %
13	Submitted to Lebanese University Student Paper	< 1 %
14	Submitted to Middlesex University Student Paper	< 1 %
15	Submitted to Thames Valley University Student Paper	< 1 %
16	slidelegend.com Internet Source	< 1 %
17	androidclue4u.blogspot.com Internet Source	< 1 %
18	Submitted to Manchester Metropolitan University Student Paper	< 1 %

19	Submitted to Institute of Technology Blanchardstown Student Paper	<1%
20	krishikosh.egranth.ac.in Internet Source	<1%
21	Submitted to American University of the Middle East Student Paper	<1%
22	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	<1%
23	Submitted to University of Westminster Student Paper	<1%
24	Submitted to University of Queensland Student Paper	<1%
25	ijceronline.com Internet Source	<1%
26	Submitted to University of Sheffield Student Paper	<1%
27	dspace.library.daffodilvarsity.edu.bd:8080 Internet Source	<1%
28	Android Apps for Absolute Beginners, 2012. Publication	<1%
29	Submitted to CSU, San Jose State University	