NEW HORIZON COLLEGE OF ENGINEERING

Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

A MINI PROJECT REPORT

for

Mini Project in Mobile Application Development (20CSE77A)

on

DARSHANA

Submitted by

RANGANATH.K

USN: 1NH19CS739, Sem-Sec: 7-E

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Academic Year: 2022-23(ODD SEM)

NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

CERTIFICATE

This is to certify that the mini project work titled

DARSHANA

submitted in partial fulfillment of the degree of Bachelor of Engineering in Computer Science and Engineering by

RANGNATH.K
USN: 1NH19CS739

DURING

ODD SEMESTER 2022-2023

For

Course: Mini Project in Mobile Application Development-20CSE77A

Signature of Reviewer Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner *Signature with date*

1._____ _____

2._____ _____

PLAGARISM CERTIFICATE

DARSHANA 6819792

ORIGINALITY REPORT

15%	9%	0%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Visvesvaraya Technological University, Belagavi Student Paper	6%
2	Submitted to Sri Lanka Institute of Information Technology Student Paper	3%
3	www.databasejournal.com Internet Source	1%
4	www.javatpoint.com Internet Source	1%
5	wiki.fluidproject.org Internet Source	1%
6	www.geeksforgeeks.org Internet Source	1%
7	Submitted to Softwarica College Of IT & E-Commerce Student Paper	1%
8	www.kollywoodzone.com Internet Source	<1%
9	www.coursehero.com Internet Source	<1%
10	Submitted to CSU, San Jose State University Student Paper	<1%
11	Submitted to University of Hull Student Paper	<1%
12	Submitted to Limerick Institute of Technology Student Paper	<1%
13	Submitted to National School of Business Management NSBM, Sri Lanka Student Paper	<1%
14	troindia.in Internet Source	<1%
15	www.rkimball.com Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

ABSTRACT

DARSHANA is the bridge between Temples and Devotees. For all of us who are unable to go to temples or simply do not have the time to go to temples, Darshana provides the perfect solution! Darshana helps you to connect with your favourite temples wherever you are. And it is an authentic platform for you to attain your spiritual needs. Explore all our temples - Read the history and specialty of each temple while going through our Temple gallery to enhance your spiritual experience. Book Pujas for your loved ones - On their birthdays, special occasions, to keep bad luck away with over different pujas to choose from. Give Darshans to your God by directly donating to the temples through us! Thinking of visiting the temple? Book your slots using our VIRTUAL Queue Feature so that you can visit your temple without having to wait in line and following Governmental guidelines!

“Darshana” is an android application to book the temple tickets easily. Users who have an android phone & active internet connection can start searching for temples easily through this application. And the app is made using Android Studio with all functionalities of project implemented on large scale and it is secured and provides firebase authentication with backend storage.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal, New Horizon College of Engineering, for his constant support and encouragement.

I am grateful to **Dr. Anandhi R J,** Professor and Dean - Academics, NHCE, for her unfailing encouragement and suggestions, given to me in the course of my project work.

I would also like to thank **Dr. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I also express my gratitude to **Ms. Jeevitha R,** Sr. Assistant Professor, Department of Computer Science and Engineering, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

RANGANATH.K

USN: 1NH19CS739

INDEX

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 ANDROID

Android is software that is open source, which means that anyone can take the software and use it and it is also based on the operating system called Linux. It is used to make applications that are supposed to run on mobile gadgets that have screens of various sizes and the application adjusts accordingly to the varied sizes of the screens. Android helps in development of applications and the applications so developed can be uploaded on Google Play Store and the everyone will be able to use the application.

1.1.1 Linux kernel

Linux is the operating system that is at the bottom of everything. It's beneath all the separable layers. Linux operating system, specifically the version number 3.6 has several patch-like additions to it. It has approximately 115 patches. A basic level of abstraction is given here, and this type of abstraction includes abstraction within the physical gadget and some important drivers like keyboard, display and photos etc. are also given along with everything else. The kernel is great at managing things like networking between devices and various other network handling operations are also taken care of by the kernel very efficiently. It ensures that gigantic large number of arrays of device drivers also works in a good way without much complications, and all this makes the painful process of interfacing to the hardware that stands on the peripheral very easy.

1.1.2 Libraries

Linux kernel is below a bunch of libraries that comprise of web browser engine that are open-source and the one specific web browser that it includes is WebKit, and a very famous library which goes by the name libc, and even databases are supported and more so when the database is SQLite and this SQLite database is actually a very nice repository which is also useful when it comes to sharing and storing of the information that belongs to the

application. There are libraries that support the recording and playing of audio and videos. It supports SSL libraries which are held accountable for the security of internet, etc.

Android Libraries:

These libraries are different from the libraries mentioned above as these are very specific java based libraries that are meant to cater only Android development and not the basic functionalities like network connection and WebKit browser engine. Some of the examples of the libraries that are in these set of libraries are an addition to the existing collection of facilities that help the user to build interface and also lets them draw graphics as well as lets them access databases that they may or may not have created. A gist of the essential libraries that are java based and are available under Android and the libraries that any android developer would have access to be given as follows:

- android.app – It gives the permission to user the application model and is one of the pillars of android application development.
- android.content – It provides the user the access to content and lets the user publish and lets one part of the application to communicate and exchange data with other parts of the application.
- android.database – Used to facilitate the user to get the data that is published and is given by the content providers and it consists of classes from SQLite database management system.

1.1.3 Android Runtime

Android Runtime is at the third position when it comes to architecture and is present at last But two layer. One of the core component names as Dalvik Virtual Machine is also provided By Android runtime. Dalvik Virtual Machine is a virtual machine that converts class files to dalvik executable. This type of Virtual machine just made and made efficient for Android. Feature of linux that includes multi-threading as well as memory management and used by Dalvik Virtual Machine. In java, it is said that it is intrinsic Core libraries are provided by android and then these core libraries allow Android application developers to publish application made using Android with the support of Java programming languages.

1.1.4 Application Framework

Many higher-level services are given in the shape of Java classes. The java developers do have the permission to utilize these services in their so developed product. The services that are packed into the android framework are as follows:

- Activity Manager – It manages all the parts of the application lifecycle and even the stack of the activities that are stacked one above the other and are pushed and popped as per the requirement.
- Content Providers – This gives the application developers some freedom to publish the application that they have made along with other applications
- Resource Manager –Non-code embedded resources are made available to the people. There is various non-code embedded resources such as strings, colour settings and lay-outs of user interface.
- Notifications Manager – This is required to make sure that there are options like alerts and notifications to the user.
- View System – There are certain number of views that are used to create interfaces that user used in the applications.

1.2 Activity Life Cycle

An activity can be best described as just a single screen that is a part of a larger application that is made in android. We can say that it is very much same as a window of a desktop. Generally an application can and should have not just one but many activities such that one activity leads to the other one when an event occurs on the first activity.

There is a number of phases that an activity goes through throughout its existence and this is called activity lifecycle. All these phases that the activity goes through are managed by activity stacks. And it's not that once we start an activity the previous activities are destroyed. Whenever a new activity is created the old activities don't cease to exist. They do exist but are present in the form of a stack such that one activity is above the other depending how recent was the activity and when was the last time it was accessed.

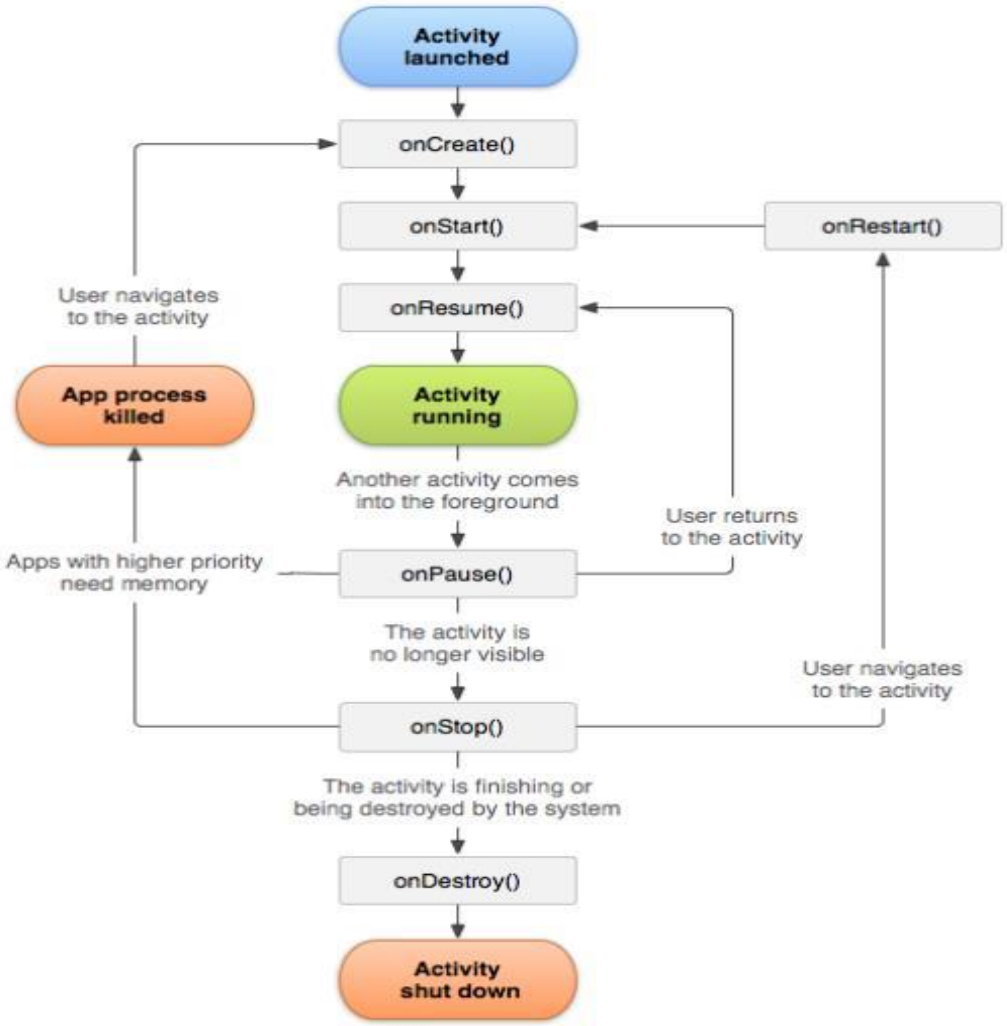


Fig 1.2.1 Android Activity Life Cycle.

1. onCreate()

Whenever an activity is formed this is the function or the method that is called and this is done to do all the work that has to be done only once throughout the cycle of the application activities and should not be performed again and again. Actions like doing the static work and the process of data item binding to whatever element or widget it has to be displayed in is done in the onCreate function. And sometimes when we restart the application, we might want to know the state the application was previously in for the app to function properly, so in such cases also the onCreate method is used.

2. onStart()

Once the activity is seen by the user, we can call this method. It is executed after the onCreate method is called when the activity is running for the first time and is the first to be called when it is the second time the app is running. This function is then followed by the function that is used to start the application again, that is, onResume method when the activity has been in the background and is needed in the foreground.

3. onRestart()

When the activity has been stopped for a long time but is now required by the user then this method is referred. It is called when the activity has gone in its initial background created and his It is invoked after the activity has been stopped and is again needed by the user then the onRestart() is called and then again the starting function is used after that.

4. onResume()

On Resume is called when the activity is at the bottom of the stack or maybe somewhere in Between but the user then demands for the same page again then the page is then moved from the bottom or intermediate position in the stack to the top again. So at this particular instance of time, the activity is on the topmost position of the stack and is then interacting with the user who demanded the page.

5. onPause()

It is invoked when an activity is going into the background but has not yet been killed. It is a counterpart to onResume(). When an activity is launched in front of another activity, this callback will be invoked on the top activity (currently on screen). The activity, under the active activity, will not be created until the active activity's onPause() returns, so it is recommended that heavy processing should not be done in this part.

6. onStop()

It's called when the user can't seem to figure out what's is happening in the application. When the activity is take away from the background, onRestart() is called, followed by the function onDestroy() when the activity is terminated or finished so basically when nothing else has to

be done by the user in the application so made, and nothing when the activity is just in the background. Note that in low memory conditions, which is a situation wherein the system in a way loses its ability to handle large amounts of data or where the system does not have enough memory to keep the activity's process running after its onPause() function is called, this method may never be called.

7. onDestroy()

When an activity is getting ended as the reason for this action could be due to the invocation of the finish() method or it could also be due to any activity that has led to the demolition of the application in order to save space, the underneath method invoked is destroy.

1.3 About Mini Project

"Darshana" is an android application to book the temple tickets easily. it is the bridge between Temples and Devotees. For all of us who are unable to stand in the queue and no need of waiting the for puja tickets, our application provides the perfect solution to book the temples for pujas, special archana's, premium darshana tickets and even they can donate and this application is secured and provides firebase authentication with backend storage.

- Darshana helps you to connect with your favourite temples wherever you are. And it is an authentic platform for you to attain your spiritual needs.
- Explore all our temples - Read the history and specialty of each temple while going through our Temple gallery to enhance your spiritual experience.
- Book Pujas for your loved ones - On their birthdays, special occasions, to keep bad luck away with over different pujas to choose from.
- Give Darshans to your God by directly donating to the temples through us!
- Thinking of visiting the temple? Book your slots using our VIRTUAL Queue Feature so that you can visit your temple without having to wait in line and following Governmental guidelines!

1.4 Objective

- The main objective of developing this project is get know about Mobile Application Development and their documentation.
- By developing mini projects students can be capable of solving a real time problems related to computer science field.
- This projects helps to get more knowledge about Android development and deploying by implementing it in own way.
- The objective of developing such a computerized program reduces the paper work and it's safe.

1.5 Advantage

- Darshana helps you to connect with your favourite temples wherever you are. And it is an authentic platform for you to attain your spiritual needs.
- Explore all our temples - Read the history and specialty of each temple while going through our Temple gallery to enhance your spiritual experience.
- Book Pujas for your loved ones - On their birthdays, special occasions, to keep bad luck away with over different pujas to choose from.
- Book your slots using our VIRTUAL Queue Feature so that you can visit your temple without having to wait in line and following Governmental guidelines!

CHAPTER 2

REQUIREMENT

2.1 MOBILE SPECIFICATION

- Operating System: Android
- Version: Lollipop and above
- RAM: Minimum 6 GB
- ROM: Minimum 10GB.

2.2 LAPTOP SPECIFICATION

- Android Operating System
- RAM: Minimum 8 GB
- Android Studio latest version
- Java 8 and above
- Keyboard , mouse and monitor
- Firebase connection

CHAPTER 3

ANALYSIS AND DESIGN

3.1 DESIGN GOALS

- Allow users to easily see and make it user friendly.
- Give users an easy way to back out (i.e. change & error recovery).
- Allow user to complete their task without being distracted by software or losing train of thought, which is while they are reading and typing.
- Everything will be made it by menu selection for easy understand.
- Give users access to information they need to complete their task.
- Wishing the user whenever they open the screen.
- Collects the details one by one without overloading.

Fig 3.1.1 Designing goals of project.

3.2 ALGORITHM / PSEUDOCODE

1. User is presented with the options – Login and Register.
    - Login-when the user enters the right credentials.
    - Register- when the user enters all the fields, the user is registered they are directed to the main us page.

2. On successful login, the user is taken to the page or activity where there are 6 options.
    i. Profile- In profile, the user can view their username and their all details like name, rashi and email-id.
    ii. Temples- Here, the user can see all the temples available for booking.
    iii. Bookings- the user can visible all his bookings with details.
    iv. Maps- Google map for searching all temples.
    v. About-us- it's completely about developer and the application information
    vi. Feedback – the user can give feedback related to the application and its services.
3. When a user decides to book any temple first they need select temple he wants then its shows options like booking for pujas or queue than they need to enter details like number of people, timings and date.
4. Once the user enters all the details correctly finally its shows the booked details of user.
5. Logout

CHAPTER 4

IMPLEMENTATION

4.1 ANDROID CONCEPTS

4.1.1 Toast

When an operation is carried out in the background by an application, it might become necessary at times to showcase the status of such an operation to the user of the application. This can be achieved by a toast which is nothing but simple feedback, shown or displayed as a small popup about the above-mentioned operation.

So as to ensure that the visibility of the current page on which the user is in, isn't disturbed and the user experience not distorted, the toast fills only a minimum amount of space required, usually in the bottom middle of the screen. This minimum length or space occupied is again decided by the message of the toast.

This ensures that the current activity on which the user was interacting with remains visible. These messages displayed by a toast message are operation feedbacks. The feedbacks change in a time duration. To ensure that the same feedback isn't shown at all times, which will make the user confused, the toast has a property in which the toasts

4.1.2 Explicit Intent

Explicit intent is used to ensure that we navigate from one page to another within the same application and is also used to pass the data from one end to the other so that the other activity is able to use the data passed without any difficulty. For example, when we want to pass values like, number of days the person wants to take leave for to the next page where the data is used to print the form and stuff then the intent is used. More specifically the intent’s put extra method is used to pass data from the form to the printing page where it is received by first declaring a bundle that uses the get intent method and gets the bundle and the get string method is used wherein, we specify the key of the value that we are willing to

retrieve and after the key specification the value so retrieved is stored in a string that is later used in the set text method of various widgets that are used to show the result.

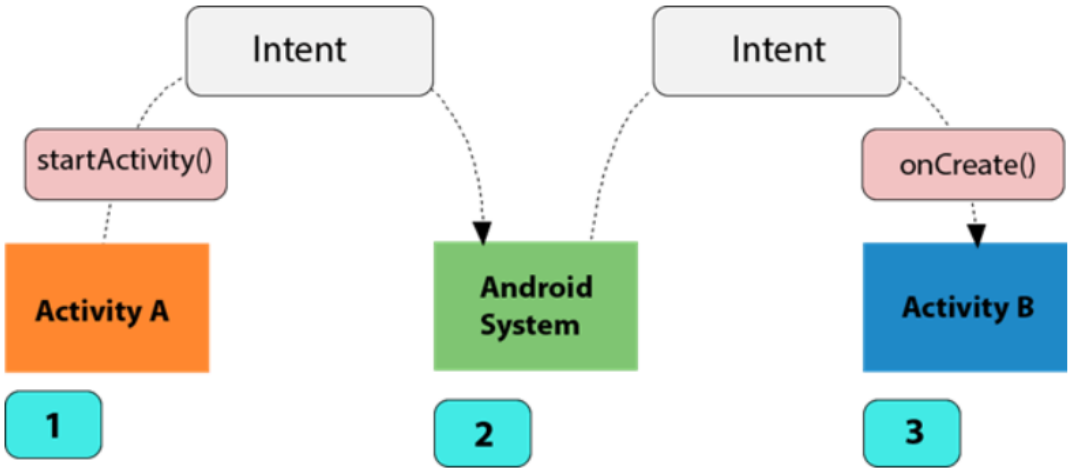


Fig 4.1.1 Intents.

4.1.3 ImageView

The imageview class is used to have an image in the application. Usage of images makes the application more attractive and makes it a bit user friendly so it makes sense for an android developer to include images as much as possible to simplify the information that has to be conveyed to the user and so the android developer uses imageview. An imageview widget can be added from the pallet by normal drag and drop or by modifying the XML code directly and adding the imageview tags in the code.

The images that need to be supplied to the imageview to make it effective is added to the drawable folder and then its source is specified within the imageview tags. There are ids that are used to map the imageviews in the java code and to refer it in the backend. So it becomes easy for us to be specific when there are multiple imageviews in the layout and we want to change the properties of only one or more imageviews but each modification is specific to the view. We can specify the height and width of the image so that image is not all over the place and has some constraints. We can change the properties of the image and also make it function like a button by setting onclick listeners in the java code and that makes it function

in a way that's similar to a button. ImageView can be added to linear layout and constraint layout.

4.1.4 TextView

A textview is one of the most basic things in the android and it is used almost everywhere. No matter what the app is that we are making we cannot avoid using textview as it is used to display text that they don't want the user to tweak and twist. It is simply used to only display hardcoded values so that the basic information is communicated in the most efficient way.

But it's not so easy to do everything right with the text view as there are plenty of things that can go wrong and make the user experience a bad one. It usually happens when the texview is too small and the text is not properly visible to the user. In such cases the developer should change the text size and make it more visible to the user and also ensure that the color that is chosen to display the text is also a more readable one and doesn't blend in with the background as that can again result into reduced readability of the text which would again result in a bad experience.

Another very common mistake that people make which using textview is setting the alignment incorrectly. The text in the textview has to be aligned in a way that not only makes it easier to understand but also boosts the aesthetic factor of the application as no one wants to use an application where the text is not properly aligned.

The font of the text can be anything as android gives us various numerous fonts at our disposal that make it very easy for us to make our application look professional and chic and also is aesthetically pleasing and hence more number of users would be tempted to use the site.

The textviews also have an id that lets any developer easily access the data of the textview and make use of it while writing the backend code.

4.1.4 Buttons

Buttons are clickable items in the android and the buttons also form the basic building blocks of an application and button can also be of different types such as an image button. The difference between an image button and a normal button is that the normal button only has a text on top of which tells us what the button is supposed to do when the user clicks on it. On the contrary, the image button doesn't have any text on it, it only has an image and that image tells us the functionality of the button and what it's supposed to do. But it's not like we have to use only either an image button or a text button which is otherwise known as an ordinary button, we can also choose both. So we can have image as well as text on a button.

There are various attributes associated with a button. In case of an imagebutton, the button's image source has to be specified and also the height and the width of the image needs to mentioned to ensure that the button is within some sort of constraints and is not overly huge or overly small. In case of an ordinary button, we need to specify the text that needs to be on the button which would ensure that its clearly communicating its functionality and isn't gibberish.

Like the image button, here also we have to give some constraints with respect to height and width. In both the cases, we need to write a function that clearly tells what the operation is that needs to be performed whenever that particular button is click and there are two ways of doing so- one of the ways is that we map the button to the java class and there get hold of the button using the id and then using that variable we set the on click listener where we have to write the operation that needs to be performed, and another way is by simply writing the function in the java class and the referring to that particular function in the xml part of the layout using android:onClick.

4.1.5 Service

An entry point that is general purpose when it comes to making an application but not in foreground is called a service and it is used for multitudes of reasons. The number of reasons

for running a service is huge. It's basically something that works in the background to make work run properly without errors and the best part about it is that it doesn't even require the application to run in the foreground. This is especially useful when the task that we are trying to perform is a lengthy one and requires resources.

A user interface is not given to a service, the service is interface less and it is always in the background that's why it's not given a UI. Activities like playing music and sending a message via an SMS are some of the operations that a service can perform.

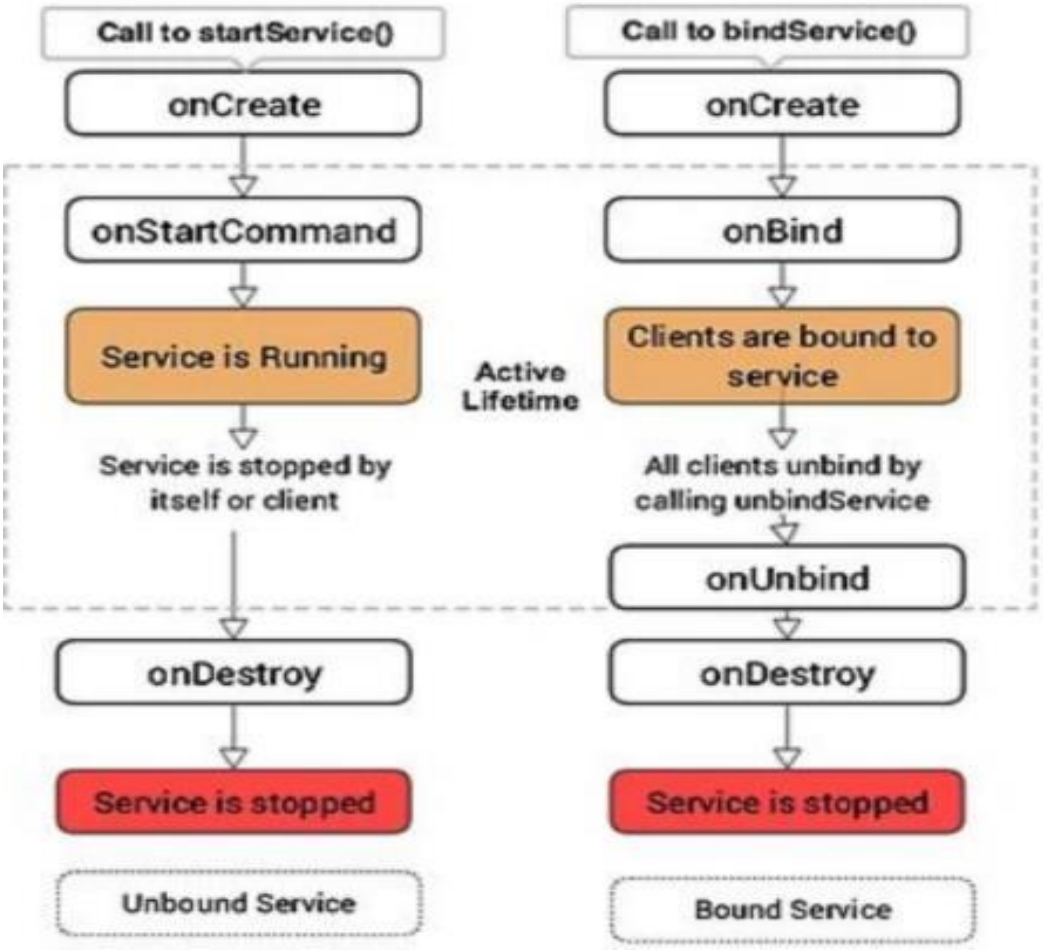


Fig 4.1.2 Bounded and unbounded service.

4.2 Functionality of the project

4.2.1 Login page

There's a login page to the application wherein there are two editboxes that are used to that the input from the user and once the input is taken, and the user has submitting the values, in the backend we first check if the values are empty strings, and if they are, then the same is informed to the user and the user is asked to enter some valid values. Another validation that is performed is we check if the values entered by the user are present in the database or not. If the value is present then the user is directed to the about us page and from there they read about what the whole application is all about and when they click on continue they are directed to the dashboard where they are free to use the services of the application. This application also has an admin login, so before we look for the userdetails in the database we check if the values entered correspond to the admin login credentials and if they do match with the admin credentials then the user is directed to the page where all the main implementation are provided

4.2.2 Registration Page

The registration page of the application is used to make sure that the user who doesn't already have their details in the database but are willing to use the application services are not stopped from doing so and then they do so by registering themselves.

On the registration page, editbox is used to take input from the user and the user enters the username and password. The password is entered twice in two separate boxes to confirm if the user has entered the right password.

When the user clicks on the submit button then the backend logic is executed where we perform multiple validations. The first validation is that we check if all the input boxes are filled and if the boxes are not Filled then a toast message is displayed that tells the user that filling up of all the input fields is necessary.

If all the fields are filled then we check if the username already exists in the database and if it already exists then the user is asked to change the username. Moving on, we check if the passwords entered in both the input fields match and if they don't then the error message is displayed in the form of toast.

Now if the user has managed to enter everything properly and has passed all the validations then the user registration is considered successful and the data is inserted into the database and the success message is given to the user in the form of a toast. The user can then go back to the login page and use the same credentials to login and continue using the service.

4.2.3 Dashboard

The dashboard appears after we have entered all the input fields properly and has also clicked on the signing button on the signup page. The dashboard has 7 options and those options are as follows:

- My profile
- Temples
- Tickets
- Map
- Logout
- Feedback
- About-us.

My Profile:

My profile page lets the user know when they created the account, and if they need to verify their details they can see in this page, so far along with basic details such as their email, phone number, name, date of birth and rashi, and this details are stored in the database and they retrieved from database and displayed here.

```
// Initialising the text view and imageview
avatartv = view.findViewById(R.id.avatartv);
nam = view.findViewById(R.id.nametv);
email = view.findViewById(R.id.emailtv);
phn = view.findViewById(R.id.mobno);
rashi = view.findViewById(R.id.rashi);
dob = view.findViewById(R.id.dob);
fab = view.findViewById(R.id.fab);
//Query query = databaseReference.orderByChild("email").equalTo(firebaseUser.g

root.addValueEventListener(new ValueEventListener() {

    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        Name=dataSnapshot.child("Name").getValue().toString();
        emaill=dataSnapshot.child("Email").getValue().toString();
        phone=dataSnapshot.child("Mobile").getValue().toString();
        rash=dataSnapshot.child("Rashi").getValue().toString();
        dob1=dataSnapshot.child("DOB").getValue().toString();

        nam.setText("   "+Name);
        email.setText("   "+emaill);
        phn.setText("   "+phone);
```

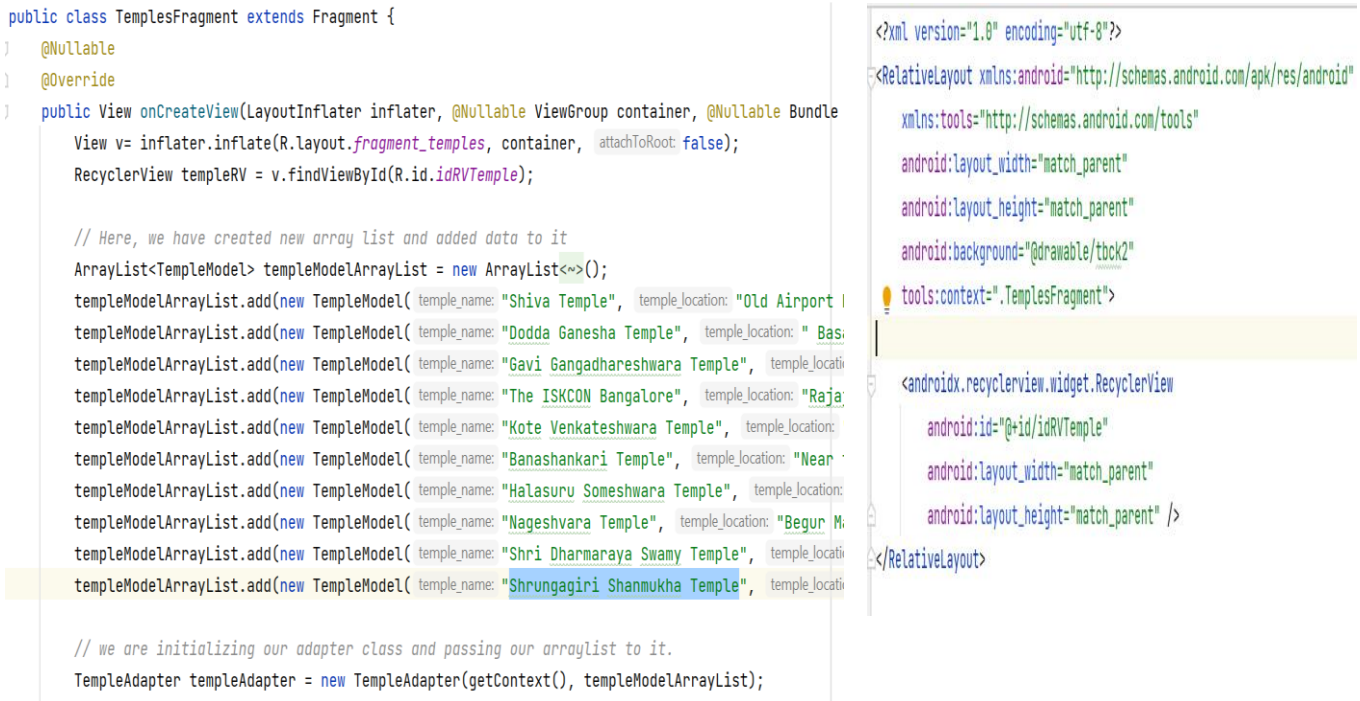
```
<TextView
    android:id="@+id/nametv"
    android:layout_width="362dp"
    android:layout_height="48dp"
    android:layout_marginStart="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginTop="5dp"
    android:drawableStart="@drawable/ic_baseline_prof_person"
    android:textColor="@color/black"
    android:textSize="25sp" />

<TextView
    android:id="@+id/emailtv"
    android:layout_width="381dp"
    android:layout_height="50dp"
    android:layout_marginStart="5dp"
    android:layout_marginLeft="5dp"
    android:drawableStart="@drawable/ic_baseline_prof_mail"
    android:textColor="@color/colorBlack"
    android:textSize="25sp" />
```

Fig 4.2.1Profile Layout Design.

Temples:

In this page application will display list of temples using card view and recycle view the temple's will be displayed within in the card view with details like temple name, temple location and image of the temple so user can select any temple which they want to visit.



```
public class TemplesFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle
        View v= inflater.inflate(R.layout.fragment_temples, container,  attachToRoot: false);
        RecyclerView templeRV = v.findViewById(R.id.idRVTemple);

        // Here, we have created new array list and added data to it
        ArrayList<TempleModel> templeModelArrayList = new ArrayList<~>();
        templeModelArrayList.add(new TempleModel( temple_name: "Shiva Temple",  temple_location: "Old Airport
        templeModelArrayList.add(new TempleModel( temple_name: "Dodda Ganesha Temple",  temple_location: " Bas
        templeModelArrayList.add(new TempleModel( temple_name: "Gavi Gangadhareshwara Temple",  temple_locati
        templeModelArrayList.add(new TempleModel( temple_name: "The ISKCON Bangalore",  temple_location: "Raja
        templeModelArrayList.add(new TempleModel( temple_name: "Kote Venkateshwara Temple",  temple_location:
        templeModelArrayList.add(new TempleModel( temple_name: "Banashankari Temple",  temple_location: "Near
        templeModelArrayList.add(new TempleModel( temple_name: "Halasuru Someshwara Temple",  temple_location:
        templeModelArrayList.add(new TempleModel( temple_name: "Nageshvara Temple",  temple_location: "Begur M
        templeModelArrayList.add(new TempleModel( temple_name: "Shri Dharmaraya Swamy Temple",  temple_locati
        templeModelArrayList.add(new TempleModel( temple_name: "Shrungagiri Shanmukha Temple",  temple_locati

        // we are initializing our adapter class and passing our arraylist to it.
        TempleAdapter templeAdapter = new TempleAdapter(getContext(), templeModelArrayList);
```

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/tbck2"
    tools:context=".TemplesFragment">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/idRVTemple"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</RelativeLayout>
```

Fig 4.2.2 Temples Fragment Design.

Tickets:

In this fragment its shows all the tickets booked by user, here also the card view and recycler view used to design the tickets to display in this fragment and here the tickets data will be retrieved from the database and going display in the card view with information like temple name, location, image, type of puja or virtual queue, amount number of people and date of darshana.

```
public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Null
    View v= inflater.inflate(R.layout.fragment_tickets, container,  attachToRoot: false);
    RecyclerView ticketRV = v.findViewById(R.id.idRVTicket);

    uid = FirebaseAuth.getInstance().getCurrentUser().getUid();
    FirebaseDatabase db=FirebaseDatabase.getInstance();
    DatabaseReference root=db.getReference().child("Users").child(uid);

    // Here, we have created new array list and added data to it
    ArrayList<TicketModel> ticketModelArrayList = new ArrayList<~>();

    root.child("Tickets").addValueEventListener(new ValueEventListener() {

        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            for (DataSnapshot postSnapshot: dataSnapshot.getChildren()) {

                tempname=postSnapshot.child("TempName").getValue().toString();
                temploc=postSnapshot.child("TempLoca").getValue().toString();
```

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

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/idRVTicket"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@drawable/tbck1" />
</RelativeLayout>
```

Fig 4.2.3 Tickets Fragment Design.

Map:

The map fragment designed using the google map fragment with api key provided from them, it shows the location map for user with current location and markers for the all temples listed in this application.

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

    <fragment
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/google_map"
        android:name="com.google.android.gms.maps.SupportMapFragment"/>

</RelativeLayout>
```

```
LatLng Dharmatemp = new LatLng( v: 12.965696173486803,  v1: 77.5834416250491);
LatLng Sshanmukatemp = new LatLng( v: 12.912718129854374,  v1: 77.52888775203188);

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Initialize view
    View view = inflater.inflate(R.layout.fragment_map, container,  attachToRoot: false);
    client = LocationServices.getFusedLocationProviderClient(getActivity());
    // Initialize map fragment
    SupportMapFragment supportMapFragment = (SupportMapFragment)

            getChildFragmentManager().findFragmentById(R.id.google_map);
    if (ActivityCompat.checkSelfPermission(getContext(),
            Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
```

Fig 4.2.4 Map Fragment Design.

Logout:

Logout option just like button it present in the navigation drawer that designed in the main fragment so it shows the all option in that if we click the logout button will remove the current user link with the database and open the login page by closing the login session.

Feedback:

Feedback plays a very important role in android development, whether our product is working properly or not, or if the layout is okay or not is all communicated between the developers and the users using the feedback mechanism. The feedback mechanism is implemented using an edittext to take the text from the user who wants to submit the feedback. The user when clicks on the submit button we perform a validation check to see if whatever the user has entered is a valid text or not. Basically, we check if the input given by the user is an empty string or does it have characters. If the entered data is an empty string, then an error is displayed and the user is asked to fill the feedback box before submitting and the same is the text in the toast message.

```
<EditText
    android:id="@+id/editTextMessage"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:lines="4"
    android:layout_marginLeft="10dp"
    android:textColor="#0B160B"
    android:hint="Type your message here"/>

<Button
    android:id="@+id/buttonSend"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_marginTop="20dp"
    android:text="Send"
    android:background="#D764DDC6"
    android:elevation="8dp"
    android:textColor="#000000"/>
```

```
builder.setPositiveButton( text: "Send", new DialogInterface.OnClickListener() {
    @SuppressLint("LongLogTag")
    @Override
    public void onClick(DialogInterface dialog, int which) {
        Log.i( tag: "Send email",  msg: "");
        String[] TO = {"1nh19cs739.ranganath.k@gmail.com"};
        String subject = "[Feedback] For Darshana Service";
        String message = e1.getText().toString().trim() + "\n\nSent By\n" +Name+"\n>>"+Email+"<<";

        Intent emailIntent = new Intent(Intent.ACTION_SENDTO);

        emailIntent.setData(Uri.parse("mailto:"));
        //emailIntent.setType("message/rfc822");
        emailIntent.putExtra(Intent.EXTRA_EMAIL, TO);

        emailIntent.putExtra(Intent.EXTRA_SUBJECT, subject);
        emailIntent.putExtra(Intent.EXTRA_TEXT,message);

        try {
            startActivity(emailIntent);
```

Fig 4.2.5 Feedback Fragment Design.

About-us:

This about us fragment is mainly for introducing our self to user that who we are and why we developed this application and what this application dose so like this it answers many questions of user and provide option to connect with us.

```
<androidx.cardview.widget.CardView
    android:id="@+id/instagram"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    android:layout_margin="10dp"
    android:clickable="true"
    app:cardBackgroundColor="@color/gray"
    app:cardCornerRadius="12dp"
    app:cardElevation="5dp"
    tools:ignore="KeyboardInaccessibleWidget">

    <ImageView
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:layout_gravity="center_vertical"
        android:layout_marginLeft="40dp"
        android:layout_marginRight="10dp"
        android:src="@drawable/instagram"
        tools:ignore="ContentDescription" />
```

```
instagram.setOnClickListener(new View.OnClickListener ()
{
    @Override
    public void onClick(View v) {
        Intent myWebLink = new Intent(android.content.Intent.ACTION_VIEW);
        myWebLink.setData(Uri.parse("http://www.instagram.com"));
        startActivity(myWebLink);
    }
});
facebook.setOnClickListener(new View.OnClickListener ()
{
    @Override
    public void onClick(View v) {
        Intent myWebLink = new Intent(android.content.Intent.ACTION_VIEW);
        myWebLink.setData(Uri.parse("http://www.facebook.com"));
        startActivity(myWebLink);
    }
});
twitter.setOnClickListener(new View.OnClickListener ()
{
```

Fig 4.2.6 About Us Fragment Design.

4.3 Firebase Database Connectivity

A document-model NoSQL database stored in the cloud, Firebase uses. It allows you to store data and synchronise it in real-time among users while being scaleable horizontally. Applications that are utilised on various devices, like mobile apps, would benefit greatly from this. Firebase is designed for offline use and has robust user-based security that enables serverless apps.

Built with automated scaling in mind, Firebase is based on the Google infrastructure. In addition to the typical features of a NoSQL database, Firebase also offers analytics, authentication, performance monitoring, messaging, crash reporting, and many other features. It is integrated with many other products due to the fact that it is a Google product. Google Ads, AdMob, Google Marketing Platform, the Play Store, Data Studio, BigQuery, Slack, and Jira are all integrated in this.

4.3.1 Working with Firebase

A Firebase project is a collection of resources that may include databases, user accounts, analytics, and other things that can be shared among several client applications. A Firebase application is a solitary application that the Firebase Project can support. Multiple Firebase applications may be contained within a single Firebase project.Visit the Firebase website at Firebase.Google.com to establish a Firebase project. Click the Go to Console button located in the upper right corner of the screen. You can build your project at the console after doing this.

4.3.2 Firebase: Realtime Database

A database located in the cloud, the Firebase Realtime Database, stores data in JSON. Every connected client receives a real-time synchronisation of the data. When we create cross-platform applications using our iOS and JavaScript SDKs, all of our clients share a single Realtime Database instance and instantly get updates with the most recent data.

We can store and sync data amongst our users in real-time using the Firebase Realtime Database, a NoSQL database. It is a sizable JSON object that the developers can control immediately. The Firebase database gives the application the most recent value of the data as well as modifications to that data by using a single API. Our consumers can easily access their data from any platform, including the web and mobile devices, thanks to real-time syncing.

The Realtime database facilitates communication between our users. We can construct our app without using servers because it comes with mobile and browser SDKs. The Real-time Database SDKs employ local caching on the device to serve and save changes when our users go offline. When the device connects to the internet, the local data is automatically synchronised.

4.3.2 Key capabilities

The ability to offer all offline and online services is possessed by a real-time database. These features include scaling over numerous databases, accessibility from the client device, and many others.

Data synchronisation, as opposed to HTTP requests, is used by the Firebase Real-time database. The updates are delivered in milliseconds to any connected device. It offers collaborative and immersive experiences while paying no attention to network coding.
Because the Firebase Database SDK saves our data to disc, Firebase apps function normally even when they are offline. Upon re-establishing connectivity, the client device receives the missed modifications.
An application server is not required to access the Firebase Real-time database. It is directly accessible from a mobile device or web browser. Through the Firebase Real-time Database Security Rules, which are expression-based rules that are run as data is read or written, data validation and security are provided.

We can fulfil the data requirements of our app by distributing our data across numerous database instances in a single Firebase project thanks to the Firebase Real-time Database on Blaze Pricing Plan. Our project will simplify authentication using Firebase authentication, and users will be authenticated in our database instances. With unique Firebase real-time database rules accessible for each database instance, you may restrict access to the data in each database.

A scalable and adaptable database from Firebase and Google Cloud Platform, Cloud Firestore is utilised for server development, mobile, and web. In order to modify the behaviour and appearance of our software without forcing users to download an update, it stores key-value pairs that are supplied by the developer. In addition to our website's HTML, CSS, and JavaScript, it also hosts resources given by our developers, such as graphs, fonts, and icons.

CHAPTER 5

RESULTS

5.1 Login and Sign-Up page

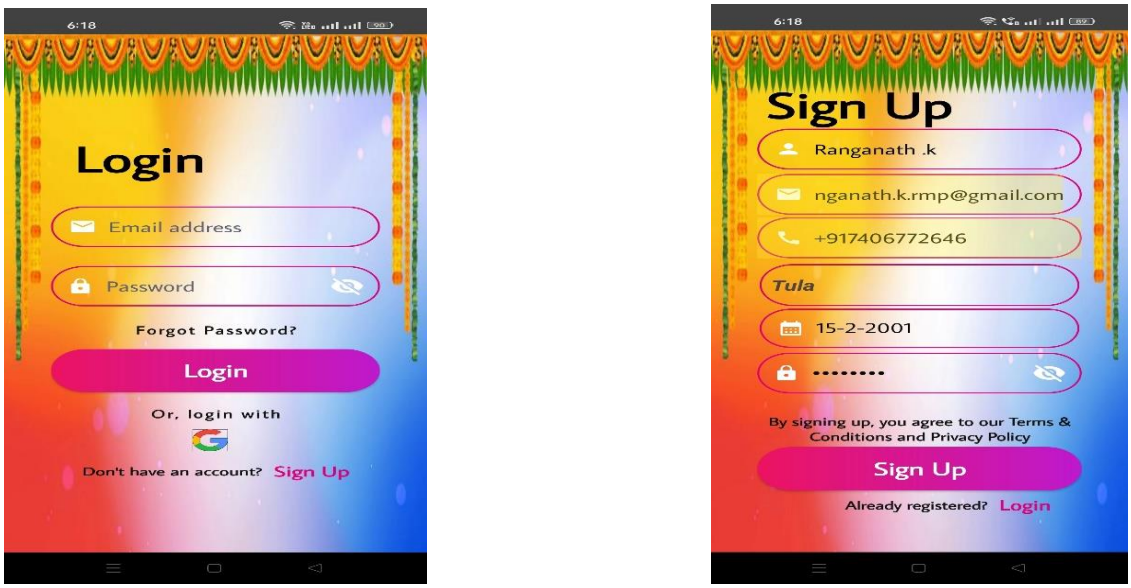


Fig 5.1.1 Login and Signup Page.

5.2 Home Screen and Navigation Drawer

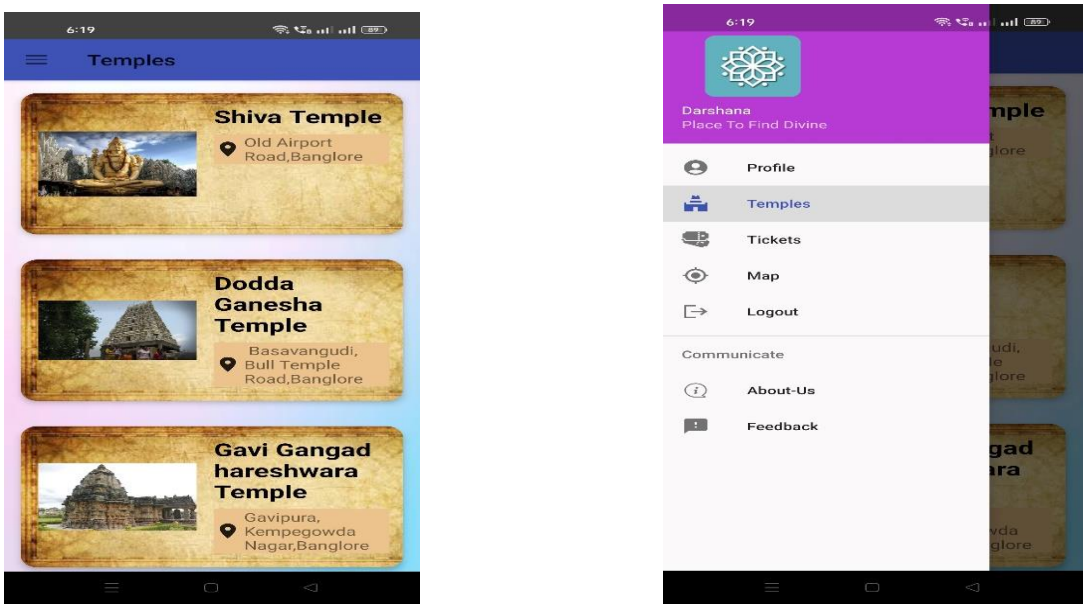


Fig 5.2.1 Home Page and Navigation Drawer.

5.3 Profile page

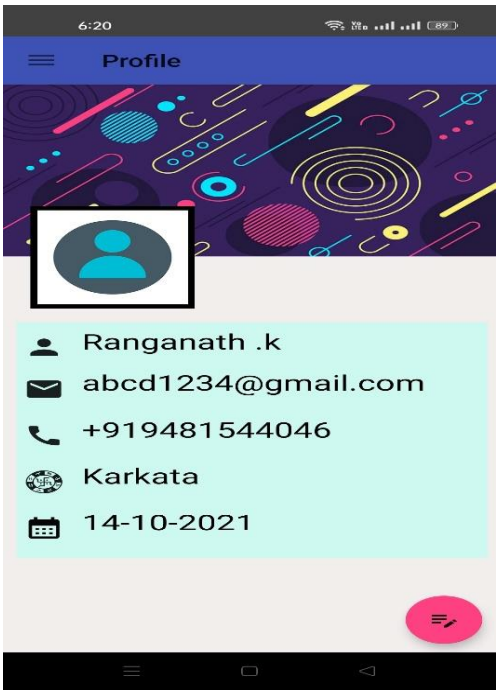


Fig 5.3.1 Profile Page.

5.4 Booked Tickets page

Fig 5.4.1 Booked Tickets Page.

5.5 Map Page

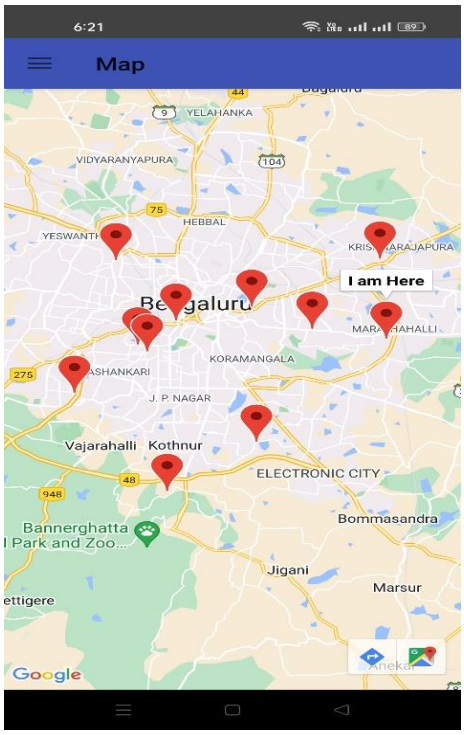


Fig 5.5.1 Google Map with Markers.

5.6 Feedback and About Us Page

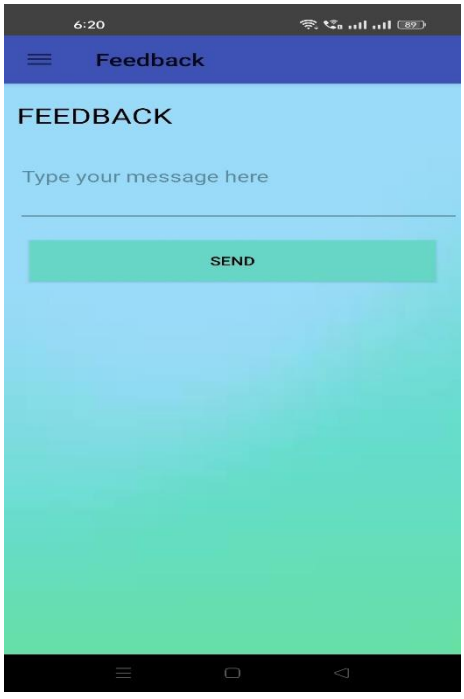


Fig 5.6.1 Feedback and About Us page.

CHAPTER 6

CONCLUSION

In conclusion, this android application named “Darshana” makes the user simpler project to help the devotees in the digital way and it is the bridge between Temples and Devotees. For all of us who are unable to go to temples or simply do not have the time to go to temples, Darshana provides the perfect solution!

It is an android application to book the temple tickets easily. Users who have an android phone & active internet connection can start searching for temples easily through this application. And the app is made using Android Studio with all functionalities of project implemented on large scale and it is secured and provides firebase authentication with backend storage.

REFERENCES

[1] https://developer.android.com

[2] https://www.geeksforgeeks.org

[3] https://firebase.google.com

[4] https://www.w3schools.in

[5] https://www.tutorialspoint.com

[6] https://www.netsolutions.com

[7] https://www.guru99.com

[8] APPLICATION DEVELOPMENT WITH ANDROID: A REVIEW (ISSN (PRINT): 2393-8374, (ONLINE): 2394- 0697, VOLUME-5, ISSUE-4, 2018)

[9] A Survey Paper on Introduction to Android and Development Process (Volume: 07 Issue: 06 June 2020, 2020, IRJET | Impact Factor value: 7.529 | ISO 9001:2008 Certified Journal)