“BUS TICKET BOOKING ANDROID APPLICATION”

JAVA PROGRAMMING FINAL PAPER

Submitted by:
Participant 01: Nitiwit Kuldiloke
Roll: 2K19/CO/263
Participant 02: Sristi Mitra
Roll: 2K19/CO/389
Participant 03: Zishenendu Sarker
Roll: 2K19/CO/450

Submitted to:
Swetha Allam Maam
Java Programming
Delhi Technological University

November,2021

NAME	ROLL NO	EMAIL I'D
Nitiwit Kuldiloke	2K19/CO/263 (A-4)	nitiwitkuldiloke_2k19co263@dtu.ac.in
Sristi Mitra	2K19/CO/389 (A6)	sristimitra_2k19co389@dtu.ac.in
Zishenendu Sarker	2K19/CO/450 (A6)	zishnendusarker_2k19co450@dtu.ac.in

A comprehensive project report has been submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Computer Engineering

Under the supervision

of

Swetha Allam Maam

(Assistant professor)

Delhi Technological University, formerly

Delhi College of Engineering

Department of Computer Science Engineering
Delhi Technological University, Shahbad Daulatpur, **Main Bawana Road**, Delhi-110042. **India**

Acknowledgment:

We have completed our report on “Bus Ticket Booking Android Application”.Our report won’t be possible without the contribution of several individuals from different walks of life. We would like to express our humble gratitude as well as thank our honorable instructor of the course ‘ Java Programming’ Prof. Swetha Allam, for her direction, constant and spontaneous support, and constructive suggestions. Without her help, this report could not have been a comprehensive one. We would like to thank some of the individuals for lending their help and giving us their invaluable time to make this report happen. We have also taken help from different websites for collecting information.

Table of Contents:

Abstract:

The project is made with objectives to help people to solve the problem of missing the bus or people who want to manage the time and booking ticket in advance and to implement the idea of Java in order to improve the performance of application development. It is a Digital Bus Management System for Bus Transportation services. People who are traveling through a Bus Transportation System or have their own Bus Travel Agency or State/National Government Bus Transport Service can use this system. The key features of the system are Manage Bus, Manage Route, Manage Fares, Manage Bookings, Entire Bus Transport System, Book Online Seats, and much more. This system was made keeping in mind a Real-World Bus Transport Management Problems, Risk and Digital solutions to it.

Introduction:

With the blessing of satisfying communication, we could see nowadays the process of getting the ticket of the bus is tougher. The process is also first come first serve which affects the people with busy timetables. The solution which our group could see and provide is the application which can book the ticket in advance and see the travelling route, timetable of every bus on the app. Hence, the people have the choice to choose and their lives will be easier for them to manage the time for coping up with the busy schedule. Our project uses concepts of Java to provide it with different features that made it stand out from the rest. The app can be an ongoing project as it will be developed as per the people's demand in the future.

➔ Objective:

To help people solve the problem of bus ticket booking, we are developing an application which will help people book tickets at ease and make them feasible for everyone. To investigate and analyze the problems on the existing systems and provide a solution to the customer and management by having an under-stable system that individuals will be able to operate within a short period.

➔ Background:

In 1993, IBM released the first smartphone for general use, which had functions such as a calendar, calculator, global clock, and contact book. The BlackBerry Smartphone, which debuted in 2002 and merged with the novel notion of wireless email, was RIM's next big breakthrough in the realm of mobile application development. The advancement in mobile application creation continues to polish the digits of years, and the biggest blockbuster was seen by the general public in July 2008, when Apple released the App Store for iOS users with only a few thousand applications. Later that year, Android followed suit, launching the Android Market in October of the same year.

Literature Review:

This chapter defines facts and findings on electronic ticketing or e-ticketing after reading some articles, books, websites or journals that are related to the system,decide to describe methodology that are used to develop the system, state out project requirements, explain action plans prior to the end of the project . All the information related to the online system is found by surfing the Internet and going to the library. Literature review is done and findings come out after reading through all the information.

Scope of the Project:

It will help both user and administrator to easily manage their work. Admin which is the bus company will be easier to get the customer information no need to contact by offline or by phone call which might have some issues during the conversation. Same as the customer or user which will be easier in order to book the ticket and know the timetable of the bus and manage their time for it.

Our project aim for the feasibility for both Bus Provider and Bus customer to feel comfortable to use the App

- To assist the staff in capturing the effort spent on their respective working areas.
- To utilize resources in an efficient m `anner by increasing their productivity through automation.

- The system generates types of information that can be used for various purposes.
- It satisfy the user requirement
- Be easy to understand by the user and operator
- Be easy to operate
- Have a good user interface
- Be expandable

Introduction to Java:

Java is a multithreaded, dynamic, distributed, portable, and resilient interpreted programming language that is class based, object oriented, platform agnostic, portable, architecturally neutral, multithreaded, dynamic, and distributed.

- Java's capabilities are not restricted to a single application domain; rather, it may be utilized in a wide range of applications, earning it the moniker "General Purpose Programming Language."
- Java is a class-based/oriented programming language, which implies it supports the object-oriented programming language's inheritance feature.
- Java is an object-oriented programming language, which implies that software written in Java is made up of a variety of different sorts of objects.
- Processor architecture has no bearing on Java code. A Java application developed for any platform's 64-bit architecture will work flawlessly on a 32-bit (or any other architecture) machine.
- Java is a dynamic programming language, which means that it performs many programming behaviors during runtime rather than at compile time, as static programming does.
- A Java program when compiled produce bytecodes. Bytecodes are magic. These bytecodes can be transferred via network and can be executed by any JVM, hence came the concept of ‘Write once, Run Anywhere.
- Java is a robust programming language, which means it can handle errors while the program is running and continue to function in the face of anomalies to some level. Garbage collection is automatic, memory management is robust, exception handling is handled, and type checking is performed.
- Java is a compiled programming language that converts Java code into Java bytes. After that, the JVM is used to run the program.

- Unlike other programming languages, which communicate with the operating system via the user runtime environment, Java adds an extra layer of security by placing the JVM between the program and the operating system.
- Java is a High Level Programming Language the syntax of which is human readable. Java lets programmer to concentrate on what to achieve and not how to achieve. The JVM converts a Java Program to Machine understandable language.

Theory related to Java in the project:

- **Java Libraries:** A Java library is just a collection of classes that have been written by somebody else already. Java is one of the most popular programming languages. Java provides a rich set of libraries, and its standard Java library is a very powerful that contains libraries such as java.lang, java.util, and java.math, etc.
- **Public class:** Each source file can only have one public class. There can be many non-public classes in a source file. The name of the public class should also be the name of the source file, with.java attached at the end. A variable or method that is public means that any class can access it. This is useful for when the variable should be accessible by your entire application. Usually common routines and variables that need to be shared everywhere are declared public.
- **Private class:** The best use of private keywords is to create a fully encapsulated class in Java by making all the data members of that class private. If we make any class constructor private, we cannot create the instance of that class from outside the class. The keyword 'private' in Java is used to establish the limitation of accessibility of the class, method or a variable in the java code block. If a class, method or variable is entitled as private in the program, that means that a particular class, method or variable cannot be accessed by outside the class or method, unlike the public method.
- **Inheritance:** Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. The idea behind inheritance in Java is that you can create new classes that are built upon existing classes. The most important use of inheritance in Java is code reusability. The code that is present in the parent class can be directly used by the child class. Method overriding is also known as runtime polymorphism. Hence, we can achieve Polymorphism in Java with the help of inheritance.
- **Method override:** In Java, method overriding occurs when a subclass (child class) has the same method as the parent class. In other words, method overriding occurs when a subclass offers a customized implementation of a method specified by one of its parent classes. Method overriding is used to provide the specific implementation of a method which is already provided by its superclass. Method overriding is used for runtime polymorphism.

Softwares/frameworks used:

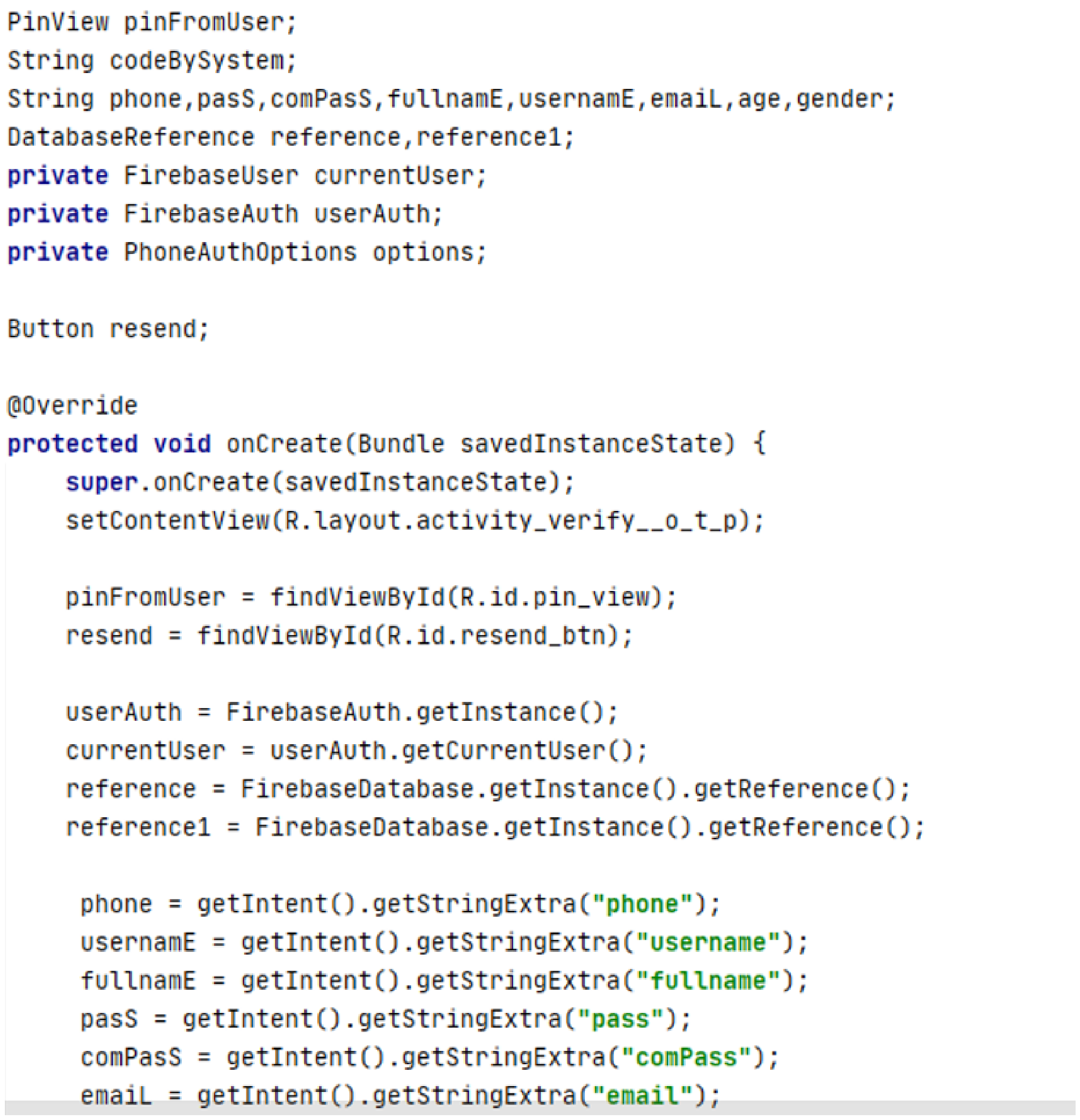
- **Flutter:** Flutter is a free and open-source mobile UI framework created by Google which enables you to construct a native mobile app with only one codebase. This implies you may design two separate apps using the same programming language and codebase (for iOS and Android). Flutter consists of two important parts:
  - ➔ An SDK (Software Development Kit): A collection of tools that are going to help you develop your applications. This includes tools to compile your code into native machine code (code for iOS and Android).
  - ➔ A Framework (UI Library based on widgets): A collection of reusable UI elements (buttons, text inputs, sliders, and so on) that you can personalize for your own needs.
- **Android Studio:** The official integrated development environment (IDE) for Android application development is Android Studio. It's based on IntelliJ IDEA, a Java integrated development environment for software development that includes code editing and developer tools.
- **Firebase Authentication:** Firebase Authentication aims to make building secure authentication systems easy, while improving the sign-in and onboarding experience for end users. It provides an end-to-end identity solution, supporting email and password accounts provides a customizable, open source, drop-in auth solution that handles the UI flows for signing in users. The FirebaseUI Auth component implements best practices for authentication on mobile devices and websites, which can maximize sign-in and sign-up conversion for your app.

Features in the application:

- ➔ **Login and Registration**: The user can just add his/her details and create an account on the app and will be able to use it whenever desired. The data will be stored in cloudFlare which works as a database to keep the data safe. During the process of Registration, the authentication of your mobile number to verify your identity, by the Firebase, will be sent automatically after you fill in the mobile number. One-Time-Password will be generated to your mobile via SMS. We have also included the feature of updating our profile where the user can update his/her name, phone number, and other details.
- ➔ **Add start and destination station**: Just add your Journey Starting Spot & Destination Spot and get all the options of buses and routes to reach there.See details of all the stops and stations so that you can reach the precise destination.
- ➔ **Book seats**: Select the seats of your comforts and convenience and book it along with your amigos and family to be together in a few minutes.
- ➔ **Reservation Details**: Once we are done with selecting your desired seat and booking it for yourself you will be able to see the details of the reservation including your details and the bus details.
- ➔ **See history**: See the history of booked tickets and rebook if you are often traveling through the same route.
- ➔ **Ticket cancellations:** We even have a feature to cancel books for each of your assigned members. By keeping real-world problems and solutions in mind We have designed a system that eliminates real-world bus transport problems
- ➔ **User friendly:** The application will be so interactive that if any customer faces any problem, it will be reserved within the system itself.

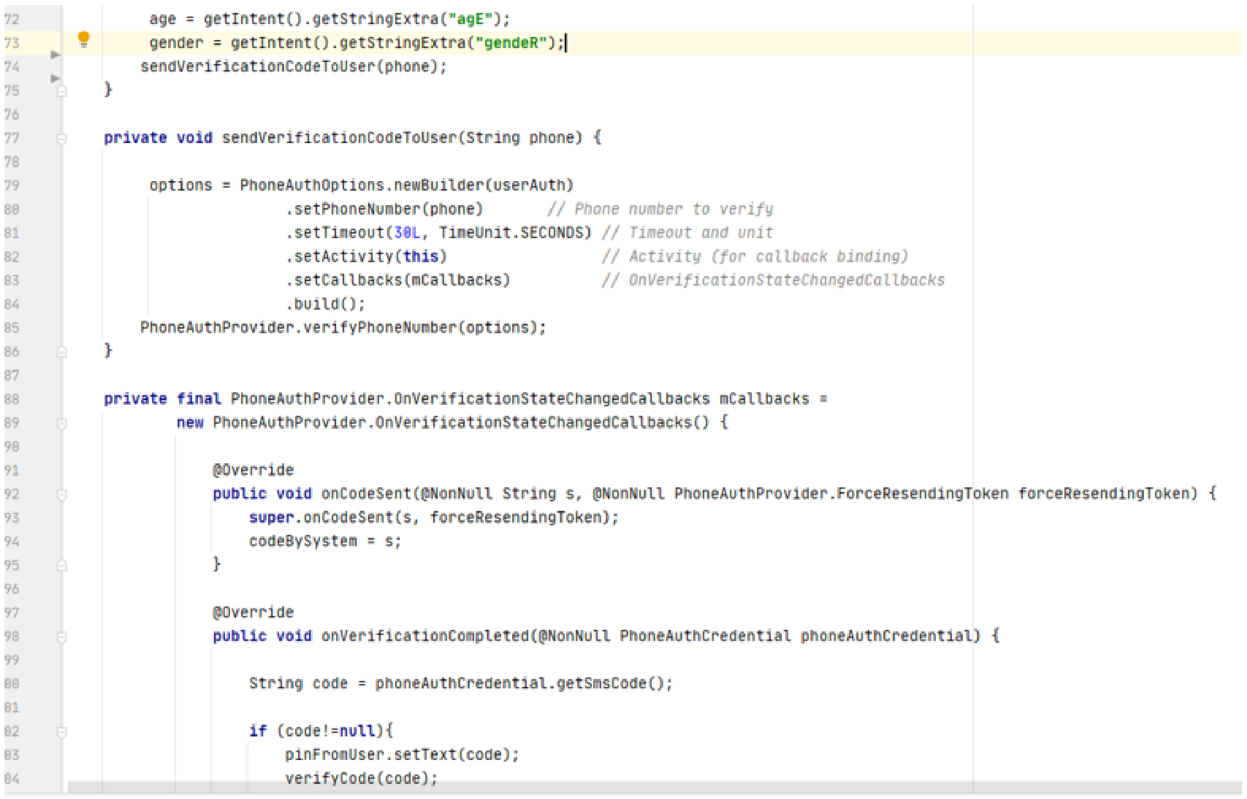
Working Procedure:

- Authentication of Implementation of Java Library:



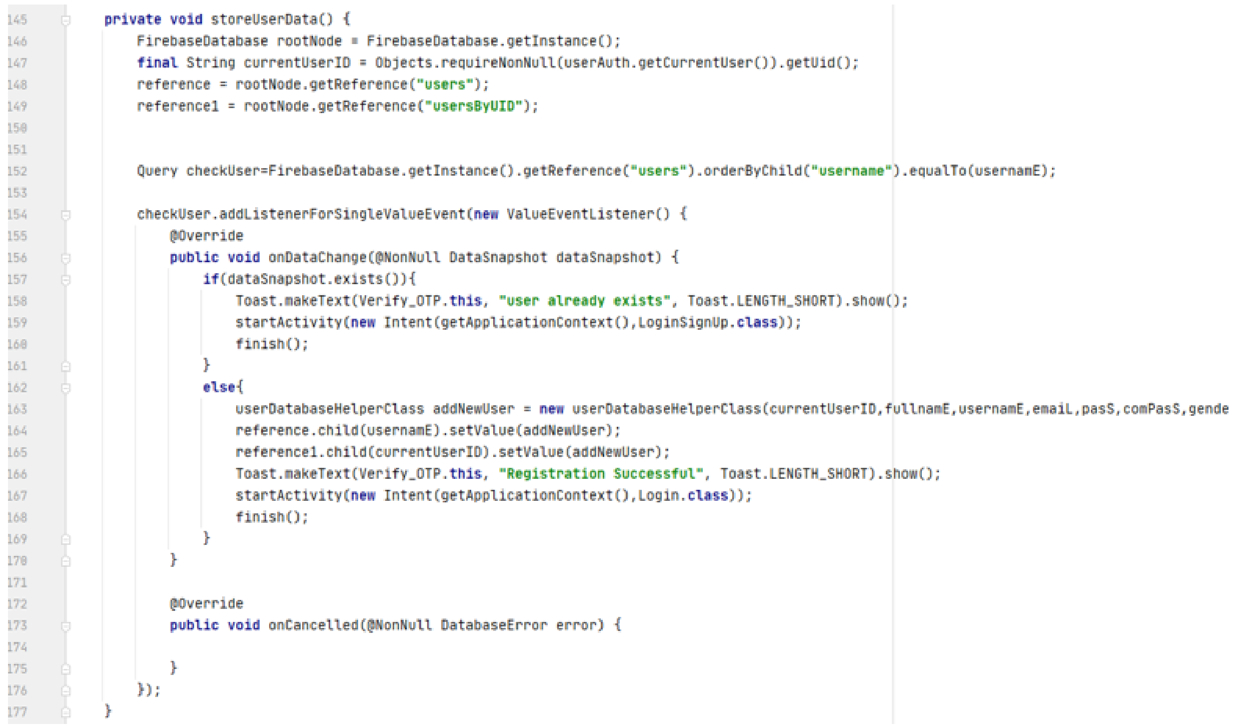
```
PinView pinFromUser;
String codeBySystem;
String phone,pasS,comPasS,fullnamE,usernamE,emaiL,age,gender;
DatabaseReference reference,reference1;
private FirebaseUser currentUser;
private FirebaseAuth userAuth;
private PhoneAuthOptions options;

Button resend;

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

    pinFromUser = findViewById(R.id.pin_view);
    resend = findViewById(R.id.resend_btn);

    userAuth = FirebaseAuth.getInstance();
    currentUser = userAuth.getCurrentUser();
    reference = FirebaseDatabase.getInstance().getReference();
    reference1 = FirebaseDatabase.getInstance().getReference();

     phone = getIntent().getStringExtra("phone");
     usernamE = getIntent().getStringExtra("username");
     fullnamE = getIntent().getStringExtra("fullname");
     pasS = getIntent().getStringExtra("pass");
     comPasS = getIntent().getStringExtra("comPass");
     emaiL = getIntent().getStringExtra("email");
```

Firstly, we have to import the java library. Also, we can see here the concept of method override to incorporate extra work in an already defined class.



```
         age = getIntent().getStringExtra("agE");
         gender = getIntent().getStringExtra("gendeR");
        sendVerificationCodeToUser(phone);
    }

    private void sendVerificationCodeToUser(String phone) {

         options = PhoneAuthOptions.newBuilder(userAuth)
                        .setPhoneNumber(phone)       // Phone number to verify
                        .setTimeout(30L, TimeUnit.SECONDS) // Timeout and unit
                        .setActivity(this)                 // Activity (for callback binding)
                        .setCallbacks(mCallbacks)          // OnVerificationStateChangedCallbacks
                        .build();
        PhoneAuthProvider.verifyPhoneNumber(options);
    }

    private final PhoneAuthProvider.OnVerificationStateChangedCallbacks mCallbacks =
            new PhoneAuthProvider.OnVerificationStateChangedCallbacks() {

                @Override
                public void onCodeSent(@NonNull String s, @NonNull PhoneAuthProvider.ForceResendingToken forceResendingToken) {
                    super.onCodeSent(s, forceResendingToken);
                    codeBySystem = s;
                }

                @Override
                public void onVerificationCompleted(@NonNull PhoneAuthCredential phoneAuthCredential) {

                    String code = phoneAuthCredential.getSmsCode();

                    if (code!=null){
                        pinFromUser.setText(code);
                        verifyCode(code);
```

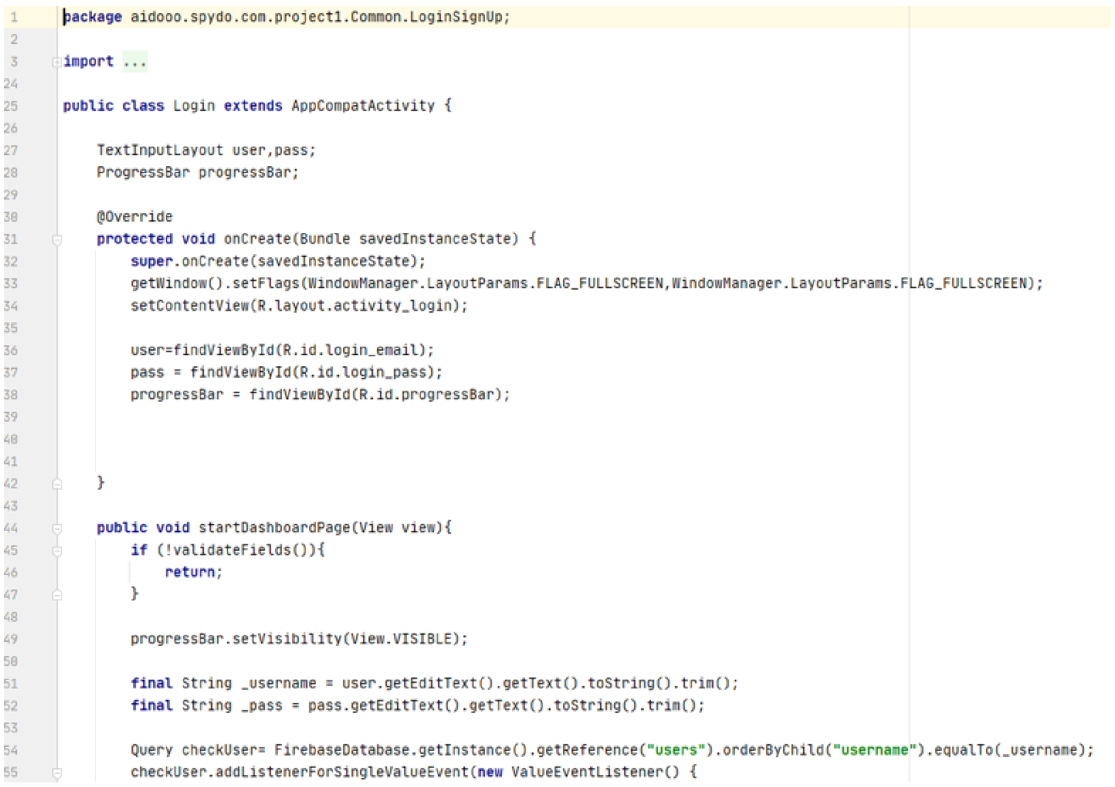
Here, we can see the concepts of private class and public class.



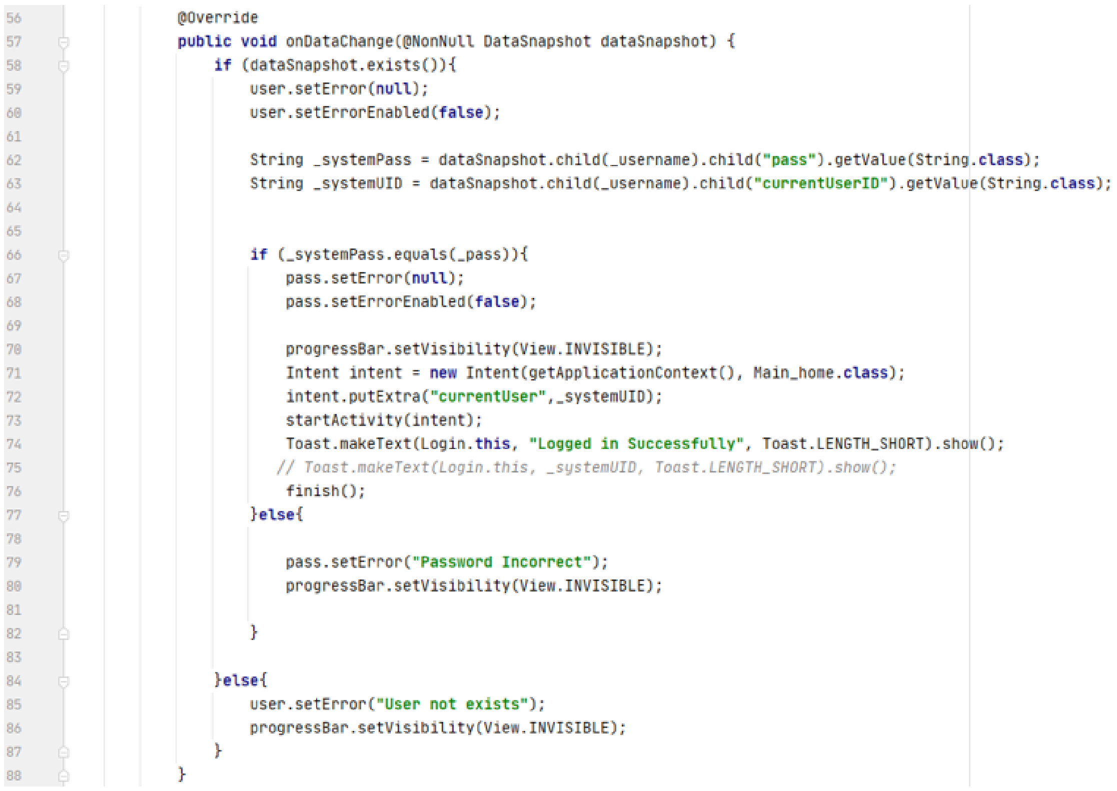
```
    private void storeUserData() {
        FirebaseDatabase rootNode = FirebaseDatabase.getInstance();
        final String currentUserID = Objects.requireNonNull(userAuth.getCurrentUser()).getUid();
        reference = rootNode.getReference("users");
        reference1 = rootNode.getReference("usersByUID");


        Query checkUser=FirebaseDatabase.getInstance().getReference("users").orderByChild("username").equalTo(usernamE);

        checkUser.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                if(dataSnapshot.exists()){
                    Toast.makeText(Verify_OTP.this, "user already exists", Toast.LENGTH_SHORT).show();
                    startActivity(new Intent(getApplicationContext(),LoginSignUp.class));
                    finish();
                }
                else{
                    userDatabaseHelperClass addNewUser = new userDatabaseHelperClass(currentUserID,fullnamE,usernamE,emaiL,pasS,comPasS,gende
                    reference.child(usernamE).setValue(addNewUser);
                    reference1.child(currentUserID).setValue(addNewUser);
                    Toast.makeText(Verify_OTP.this, "Registration Successful", Toast.LENGTH_SHORT).show();
                    startActivity(new Intent(getApplicationContext(),Login.class));
                    finish();
                }
            }

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

            }
        });
    }
```

Here, inside the if statement, calling the method where the boolean private class where it is visible outside the class.

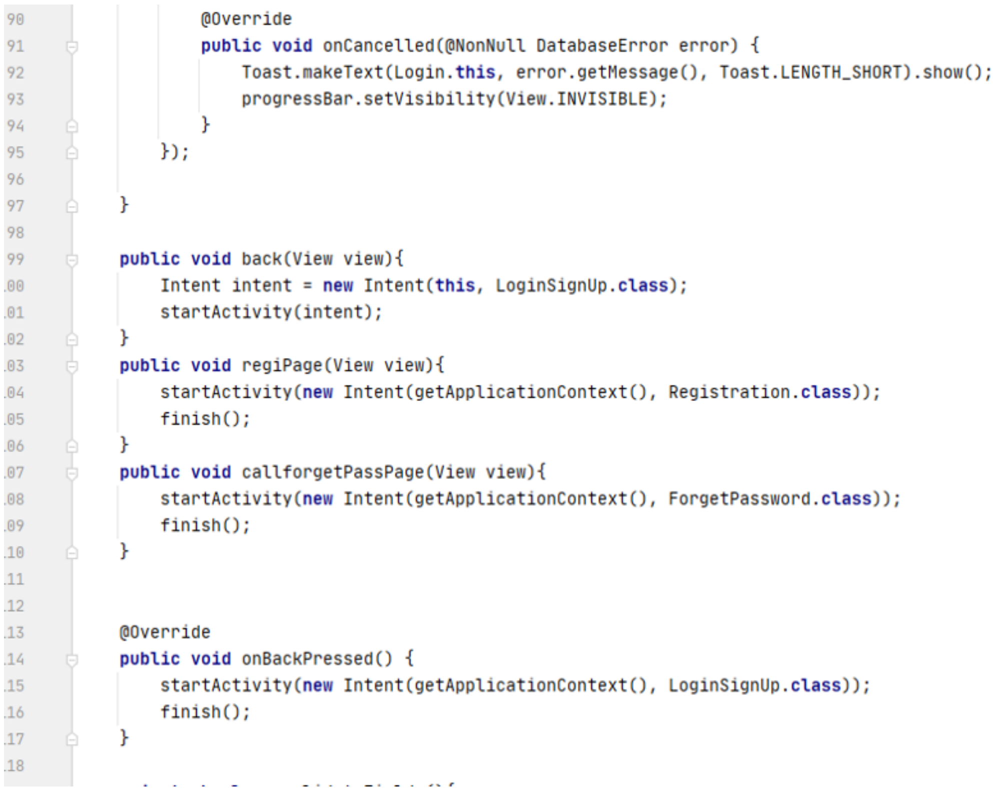
- SignIn:



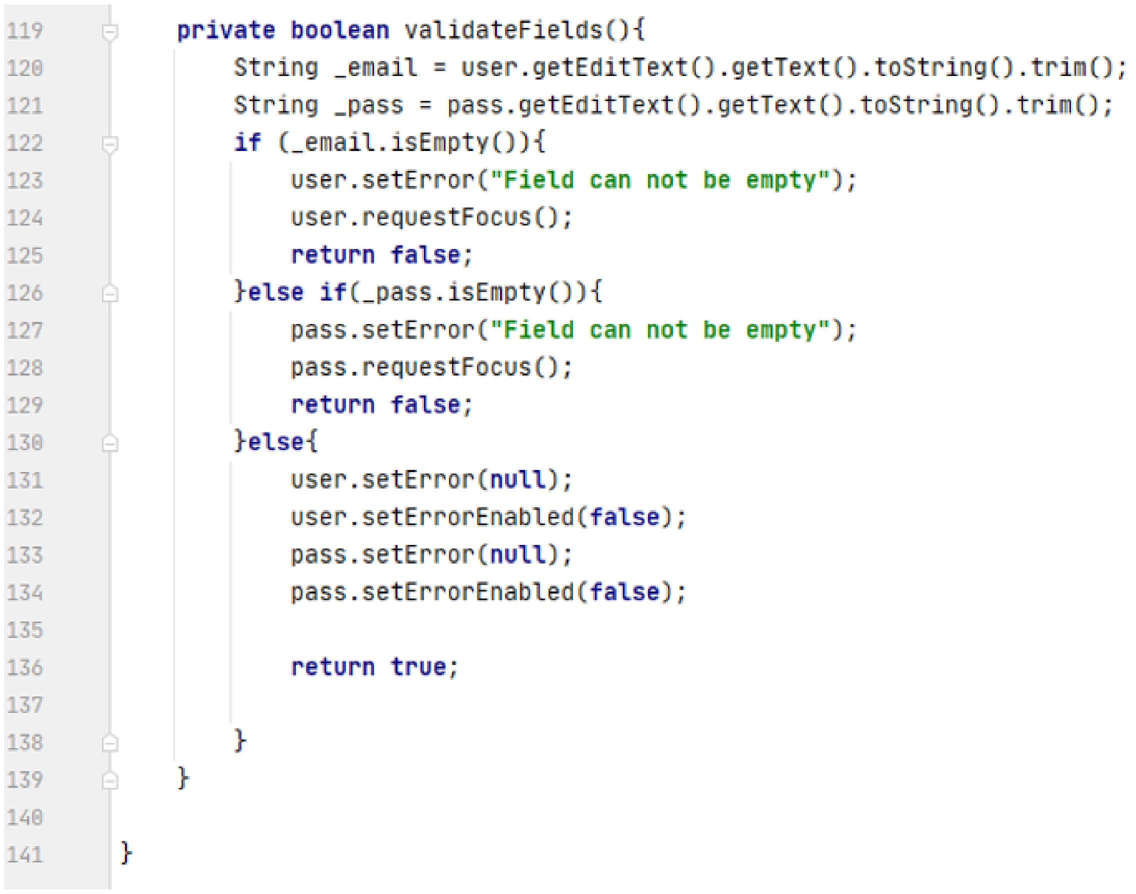
```java
package aidooo.spydo.com.project1.Common.LoginSignUp;

import ...

public class Login extends AppCompatActivity {

    TextInputLayout user,pass;
    ProgressBar progressBar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,WindowManager.LayoutParams.FLAG_FULLSCREEN);
        setContentView(R.layout.activity_login);

        user=findViewById(R.id.login_email);
        pass = findViewById(R.id.login_pass);
        progressBar = findViewById(R.id.progressBar);



    }

    public void startDashboardPage(View view){
        if (!validateFields()){
            return;
        }

        progressBar.setVisibility(View.VISIBLE);

        final String _username = user.getEditText().getText().toString().trim();
        final String _pass = pass.getEditText().getText().toString().trim();

        Query checkUser= FirebaseDatabase.getInstance().getReference("users").orderByChild("username").equalTo(_username);
        checkUser.addListenerForSingleValueEvent(new ValueEventListener() {
```



```java
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                if (dataSnapshot.exists()){
                    user.setError(null);
                    user.setErrorEnabled(false);

                    String _systemPass = dataSnapshot.child(_username).child("pass").getValue(String.class);
                    String _systemUID = dataSnapshot.child(_username).child("currentUserID").getValue(String.class);


                    if (_systemPass.equals(_pass)){
                        pass.setError(null);
                        pass.setErrorEnabled(false);

                        progressBar.setVisibility(View.INVISIBLE);
                        Intent intent = new Intent(getApplicationContext(), Main_home.class);
                        intent.putExtra("currentUser",_systemUID);
                        startActivity(intent);
                        Toast.makeText(Login.this, "Logged in Successfully", Toast.LENGTH_SHORT).show();
                      // Toast.makeText(Login.this, _systemUID, Toast.LENGTH_SHORT).show();
                        finish();
                    }else{

                        pass.setError("Password Incorrect");
                        progressBar.setVisibility(View.INVISIBLE);

                    }

                }else{
                    user.setError("User not exists");
                    progressBar.setVisibility(View.INVISIBLE);
                }
            }
```

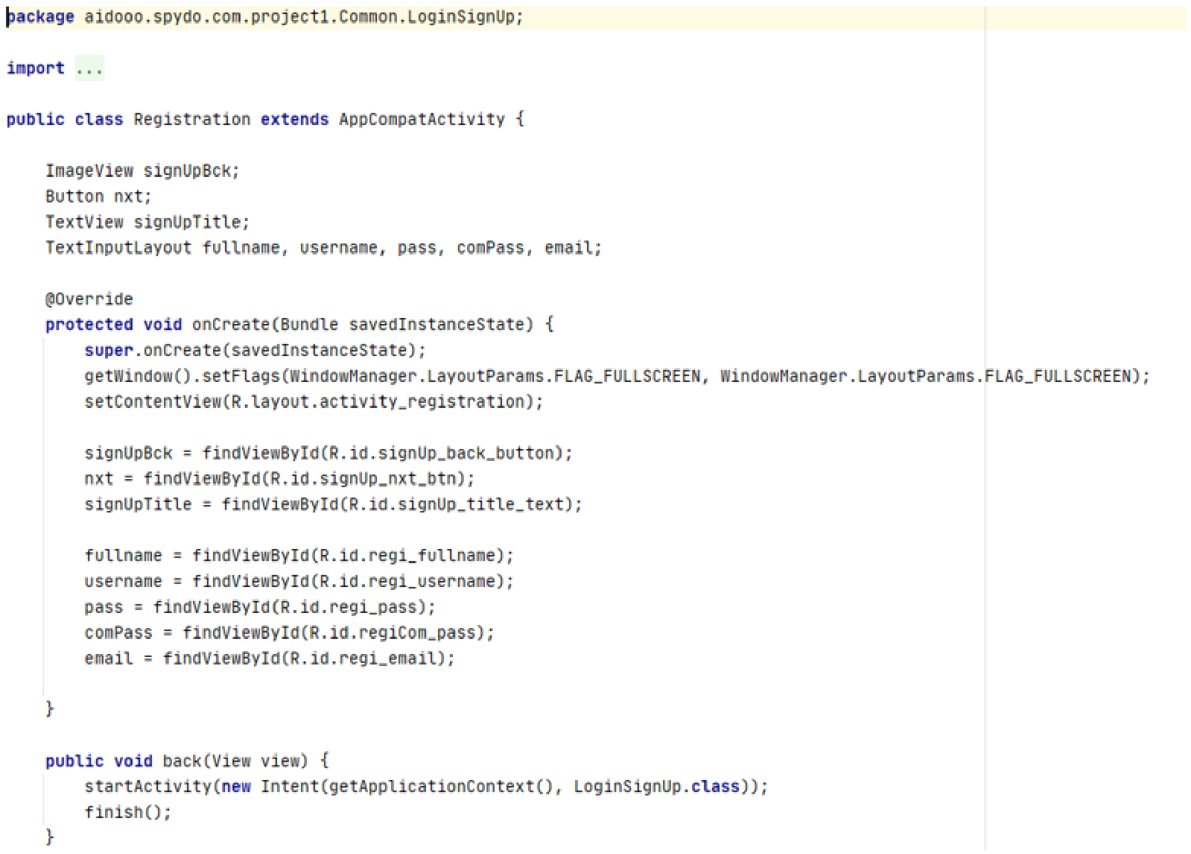


```
            @Override
            public void onCancelled(@NonNull DatabaseError error) {
                Toast.makeText(Login.this, error.getMessage(), Toast.LENGTH_SHORT).show();
                progressBar.setVisibility(View.INVISIBLE);
            }
        });

    }

    public void back(View view){
        Intent intent = new Intent(this, LoginSignUp.class);
        startActivity(intent);
    }
    public void regiPage(View view){
        startActivity(new Intent(getApplicationContext(), Registration.class));
        finish();
    }
    public void callforgetPassPage(View view){
        startActivity(new Intent(getApplicationContext(), ForgetPassword.class));
        finish();
    }


    @Override
    public void onBackPressed() {
        startActivity(new Intent(getApplicationContext(), LoginSignUp.class));
        finish();
    }

```

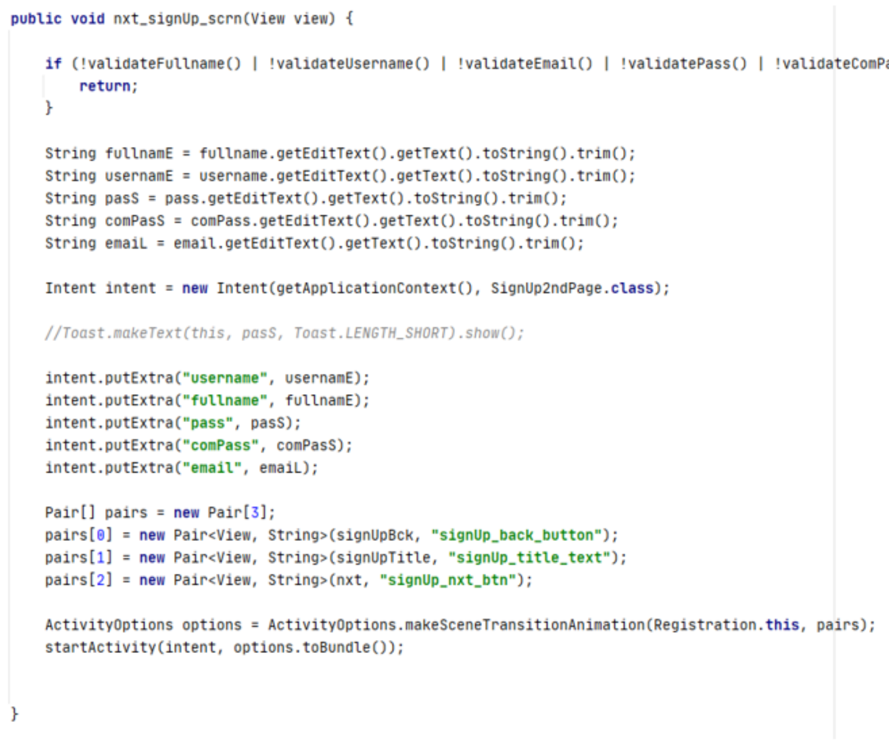


```
    private boolean validateFields(){
        String _email = user.getEditText().getText().toString().trim();
        String _pass = pass.getEditText().getText().toString().trim();
        if (_email.isEmpty()){
            user.setError("Field can not be empty");
            user.requestFocus();
            return false;
        }else if(_pass.isEmpty()){
            pass.setError("Field can not be empty");
            pass.requestFocus();
            return false;
        }else{
            user.setError(null);
            user.setErrorEnabled(false);
            pass.setError(null);
            pass.setErrorEnabled(false);

            return true;

        }
    }

}
```

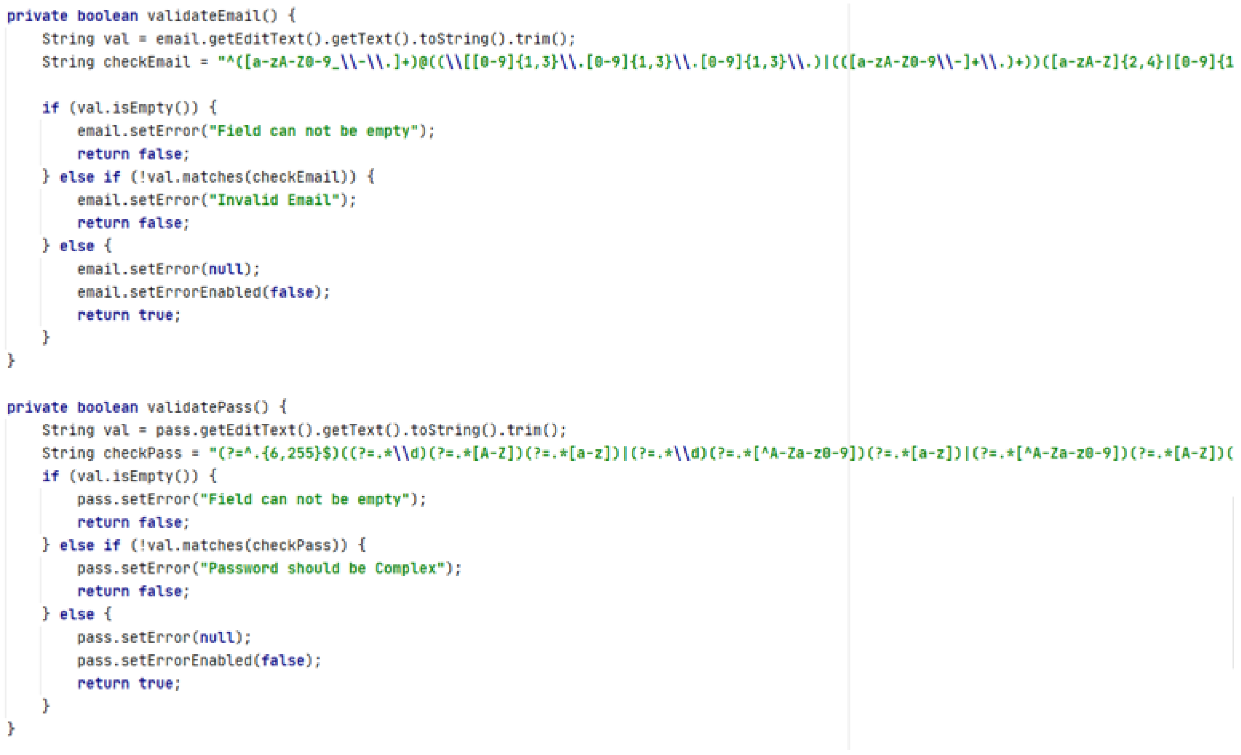
- Registration:



```java
package aidooo.spydo.com.project1.Common.LoginSignUp;

import ...

public class Registration extends AppCompatActivity {

    ImageView signUpBck;
    Button nxt;
    TextView signUpTitle;
    TextInputLayout fullname, username, pass, comPass, email;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);
        setContentView(R.layout.activity_registration);

        signUpBck = findViewById(R.id.signUp_back_button);
        nxt = findViewById(R.id.signUp_nxt_btn);
        signUpTitle = findViewById(R.id.signUp_title_text);

        fullname = findViewById(R.id.regi_fullname);
        username = findViewById(R.id.regi_username);
        pass = findViewById(R.id.regi_pass);
        comPass = findViewById(R.id.regiCom_pass);
        email = findViewById(R.id.regi_email);

    }

    public void back(View view) {
        startActivity(new Intent(getApplicationContext(), LoginSignUp.class));
        finish();
    }
```

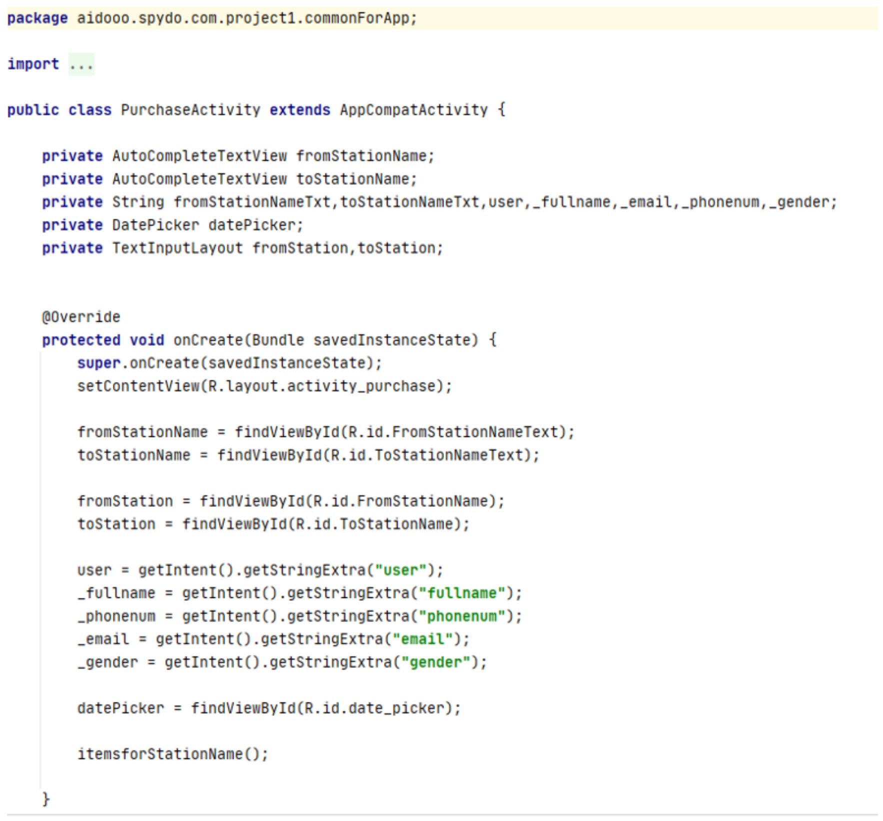


```java
public void nxt_signUp_scrn(View view) {

    if (!validateFullname() | !validateUsername() | !validateEmail() | !validatePass() | !validateComPass()) {
        return;
    }

    String fullnamE = fullname.getEditText().getText().toString().trim();
    String usernamE = username.getEditText().getText().toString().trim();
    String pasS = pass.getEditText().getText().toString().trim();
    String comPasS = comPass.getEditText().getText().toString().trim();
    String emaiL = email.getEditText().getText().toString().trim();

    Intent intent = new Intent(getApplicationContext(), SignUp2ndPage.class);

    //Toast.makeText(this, pasS, Toast.LENGTH_SHORT).show();

    intent.putExtra("username", usernamE);
    intent.putExtra("fullname", fullnamE);
    intent.putExtra("pass", pasS);
    intent.putExtra("comPass", comPasS);
    intent.putExtra("email", emaiL);

    Pair[] pairs = new Pair[3];
    pairs[0] = new Pair<View, String>(signUpBck, "signUp_back_button");
    pairs[1] = new Pair<View, String>(signUpTitle, "signUp_title_text");
    pairs[2] = new Pair<View, String>(nxt, "signUp_nxt_btn");

    ActivityOptions options = ActivityOptions.makeSceneTransitionAnimation(Registration.this, pairs);
    startActivity(intent, options.toBundle());


}
```

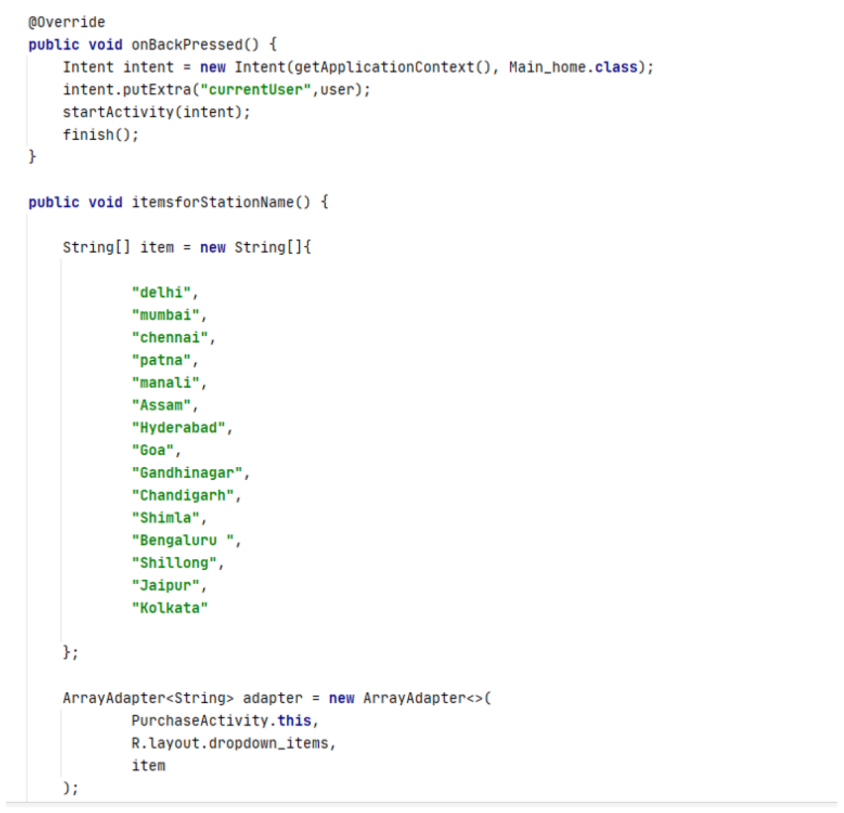


```java
private boolean validateEmail() {
    String val = email.getEditText().getText().toString().trim();
    String checkEmail = "^([a-zA-Z0-9_\\-\\.]+)@((\\[[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.)|(([a-zA-Z0-9\\-]+\\.)+))([a-zA-Z]{2,4}|[0-9]{1

    if (val.isEmpty()) {
        email.setError("Field can not be empty");
        return false;
    } else if (!val.matches(checkEmail)) {
        email.setError("Invalid Email");
        return false;
    } else {
        email.setError(null);
        email.setErrorEnabled(false);
        return true;
    }
}

private boolean validatePass() {
    String val = pass.getEditText().getText().toString().trim();
    String checkPass = "(?=^.{6,255}$)((?=.*\\d)(?=.*[A-Z])(?=.*[a-z])|(?=.*\\d)(?=.*[^A-Za-z0-9])(?=.*[a-z])|(?=.*[^A-Za-z0-9])(?=.*[A-Z])(
    if (val.isEmpty()) {
        pass.setError("Field can not be empty");
        return false;
    } else if (!val.matches(checkPass)) {
        pass.setError("Password should be Complex");
        return false;
    } else {
        pass.setError(null);
        pass.setErrorEnabled(false);
        return true;
    }
}
```

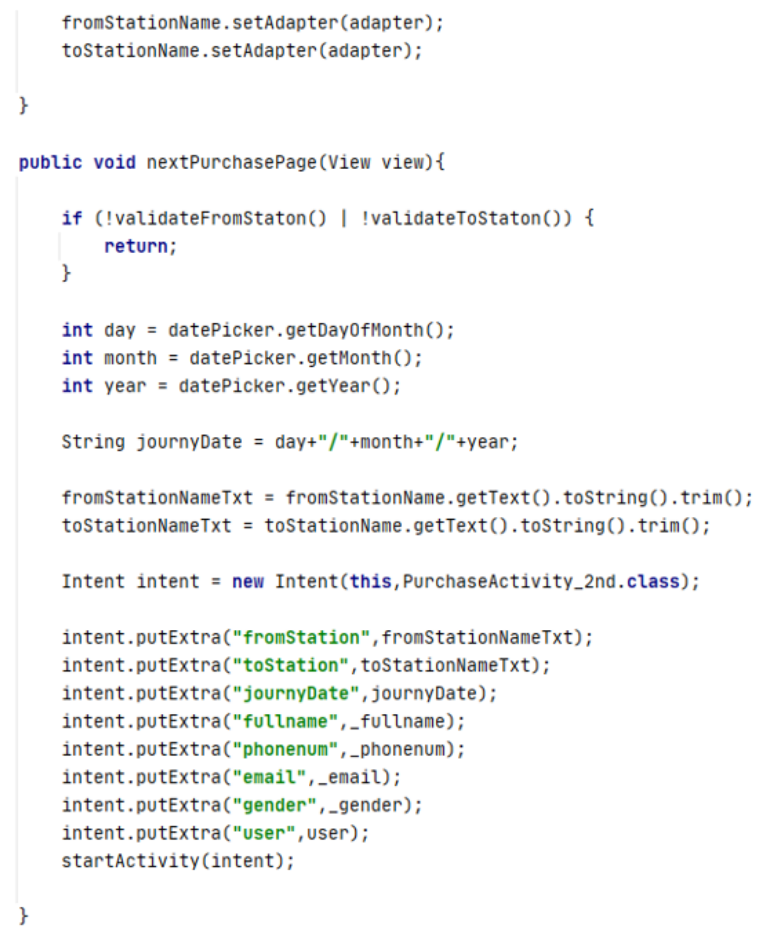
- Booking:



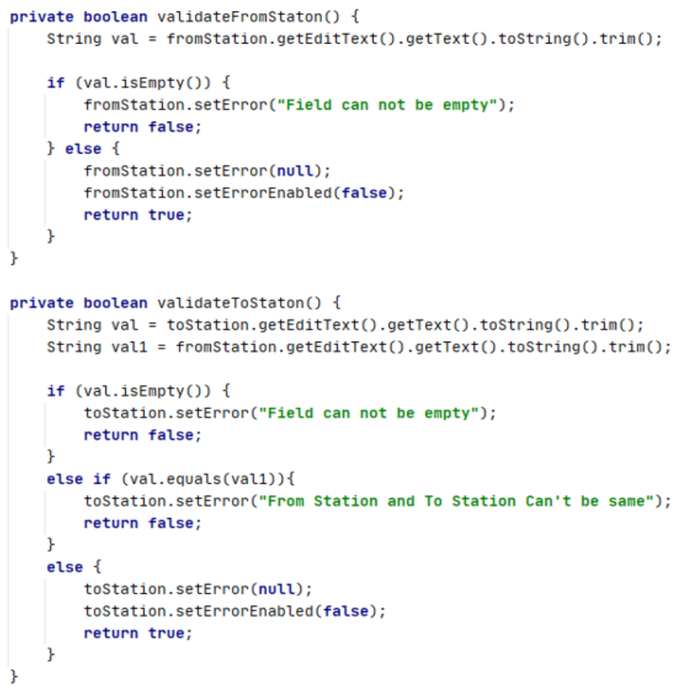
```
package aidooo.spydo.com.project1.commonForApp;

import ...

public class PurchaseActivity extends AppCompatActivity {

    private AutoCompleteTextView fromStationName;
    private AutoCompleteTextView toStationName;
    private String fromStationNameTxt,toStationNameTxt,user,_fullname,_email,_phonenum,_gender;
    private DatePicker datePicker;
    private TextInputLayout fromStation,toStation;


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

        fromStationName = findViewById(R.id.FromStationNameText);
        toStationName = findViewById(R.id.ToStationNameText);

        fromStation = findViewById(R.id.FromStationName);
        toStation = findViewById(R.id.ToStationName);

        user = getIntent().getStringExtra("user");
        _fullname = getIntent().getStringExtra("fullname");
        _phonenum = getIntent().getStringExtra("phonenum");
        _email = getIntent().getStringExtra("email");
        _gender = getIntent().getStringExtra("gender");

        datePicker = findViewById(R.id.date_picker);

        itemsforStationName();

    }
```



```
    @Override
    public void onBackPressed() {
        Intent intent = new Intent(getApplicationContext(), Main_home.class);
        intent.putExtra("currentUser",user);
        startActivity(intent);
        finish();
    }

    public void itemsforStationName() {

        String[] item = new String[]{

                "delhi",
                "mumbai",
                "chennai",
                "patna",
                "manali",
                "Assam",
                "Hyderabad",
                "Goa",
                "Gandhinagar",
                "Chandigarh",
                "Shimla",
                "Bengaluru ",
                "Shillong",
                "Jaipur",
                "Kolkata"

        };

        ArrayAdapter<String> adapter = new ArrayAdapter<>(
                PurchaseActivity.this,
                R.layout.dropdown_items,
                item
        );
```

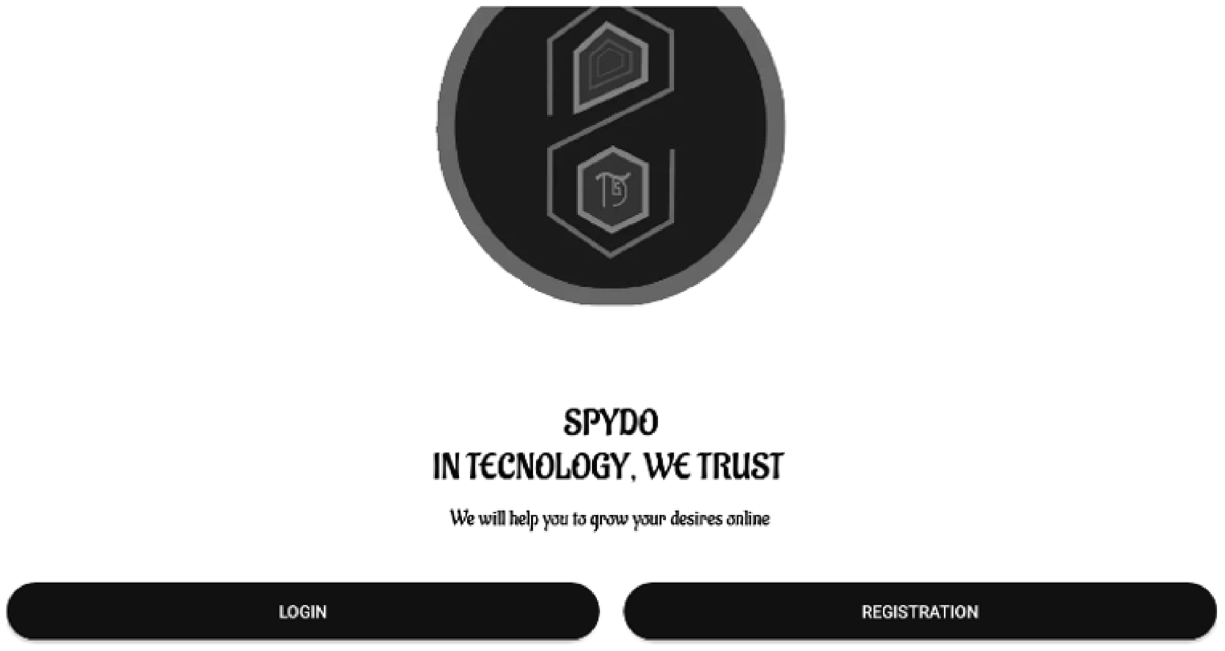


```java
    fromStationName.setAdapter(adapter);
    toStationName.setAdapter(adapter);

}

public void nextPurchasePage(View view){

    if (!validateFromStaton() | !validateToStaton()) {
        return;
    }

    int day = datePicker.getDayOfMonth();
    int month = datePicker.getMonth();
    int year = datePicker.getYear();

    String journyDate = day+"/"+month+"/"+year;

    fromStationNameTxt = fromStationName.getText().toString().trim();
    toStationNameTxt = toStationName.getText().toString().trim();

    Intent intent = new Intent(this,PurchaseActivity_2nd.class);

    intent.putExtra("fromStation",fromStationNameTxt);
    intent.putExtra("toStation",toStationNameTxt);
    intent.putExtra("journyDate",journyDate);
    intent.putExtra("fullname",_fullname);
    intent.putExtra("phonenum",_phonenum);
    intent.putExtra("email",_email);
    intent.putExtra("gender",_gender);
    intent.putExtra("user",user);
    startActivity(intent);

}
```



```java
private boolean validateFromStaton() {
    String val = fromStation.getEditText().getText().toString().trim();

    if (val.isEmpty()) {
        fromStation.setError("Field can not be empty");
        return false;
    } else {
        fromStation.setError(null);
        fromStation.setErrorEnabled(false);
        return true;
    }
}

private boolean validateToStaton() {
    String val = toStation.getEditText().getText().toString().trim();
    String val1 = fromStation.getEditText().getText().toString().trim();

    if (val.isEmpty()) {
        toStation.setError("Field can not be empty");
        return false;
    }
    else if (val.equals(val1)){
        toStation.setError("From Station and To Station Can't be same");
        return false;
    }
    else {
        toStation.setError(null);
        toStation.setErrorEnabled(false);
        return true;
    }
}
```

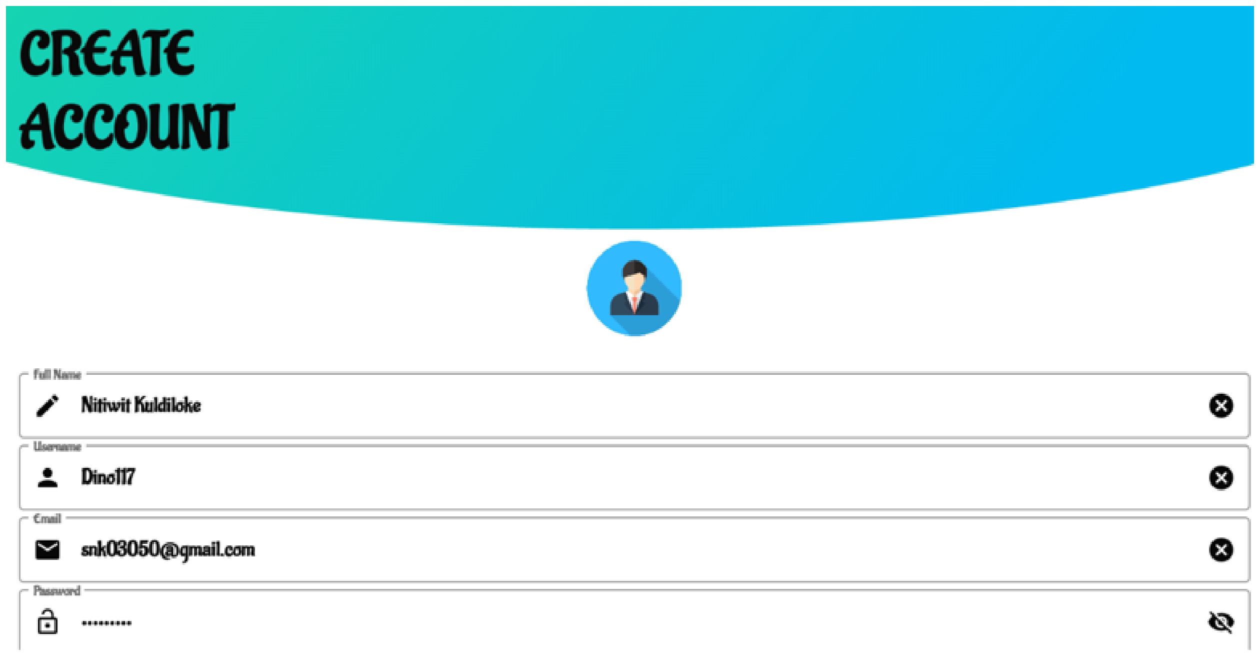
Implementation:



SPYDO
IN TECNOLOGY, WE TRUST

We will help you to grow your desires online

Fig: Front Page of the application



CREATE ACCOUNT

Full Name: Nitiwit Kuldiloke

Username: Dino117

Email: snk03050@gmail.com

Password: •••••••••

Fig: Registration

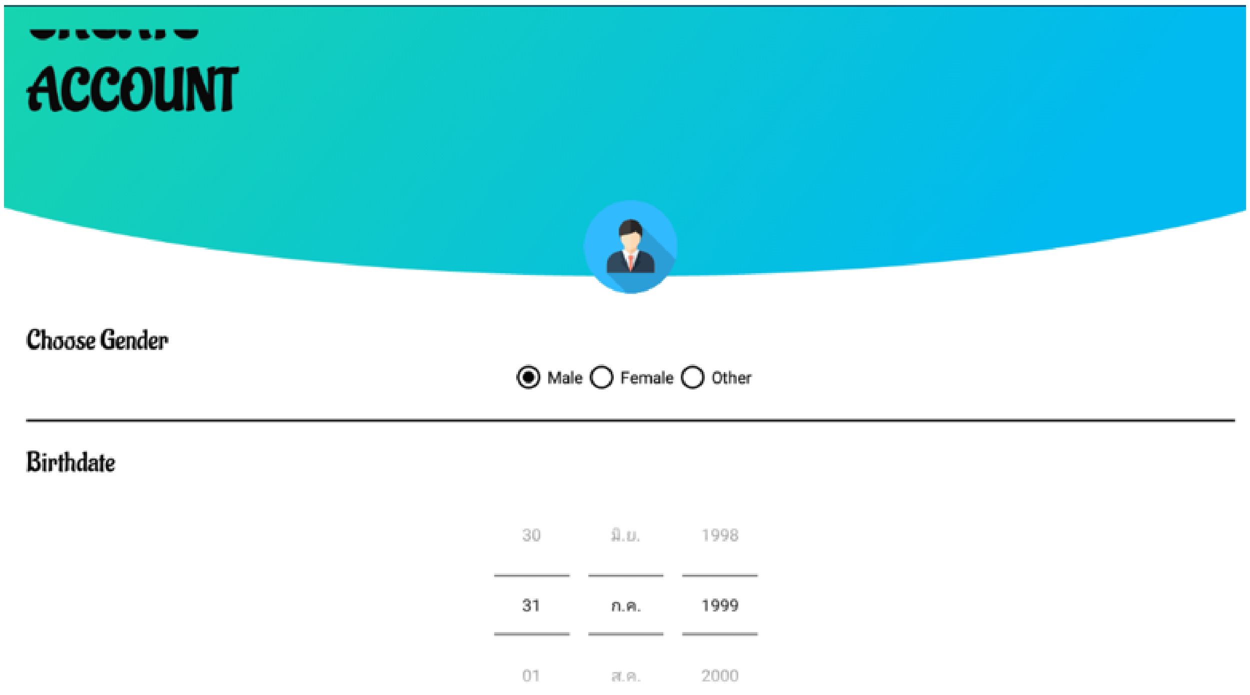


Fig: Next step of Registration

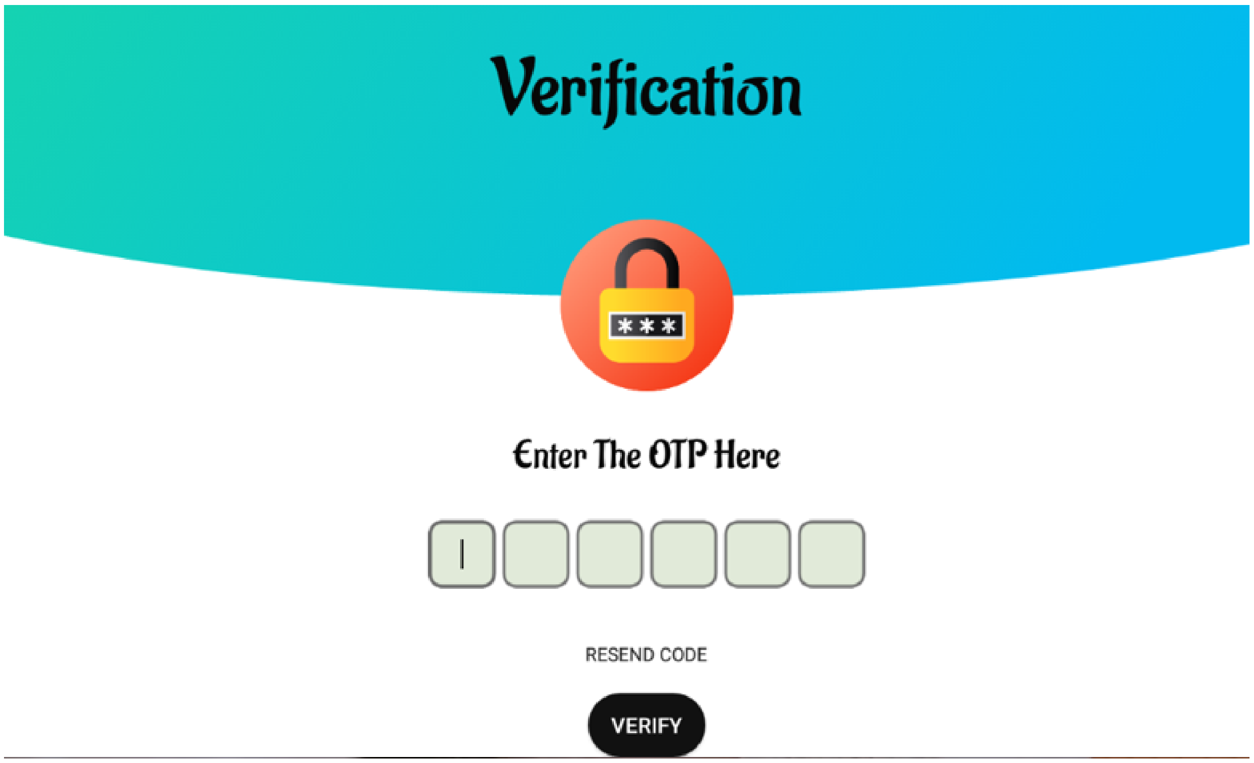


Fig: Verification of OTP

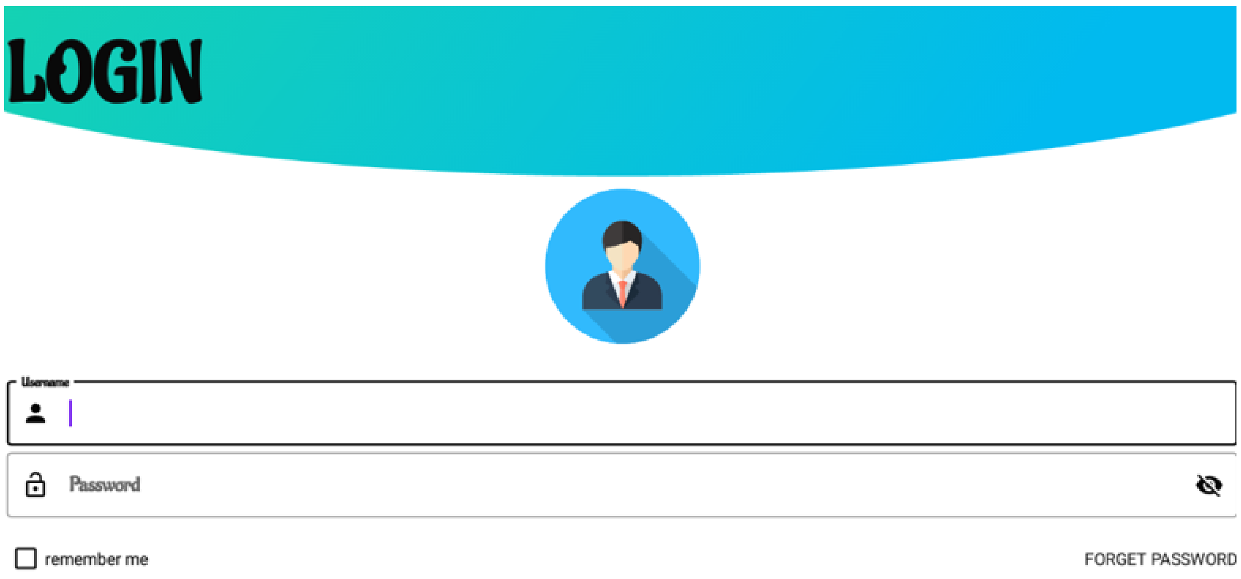


Fig: Login Page

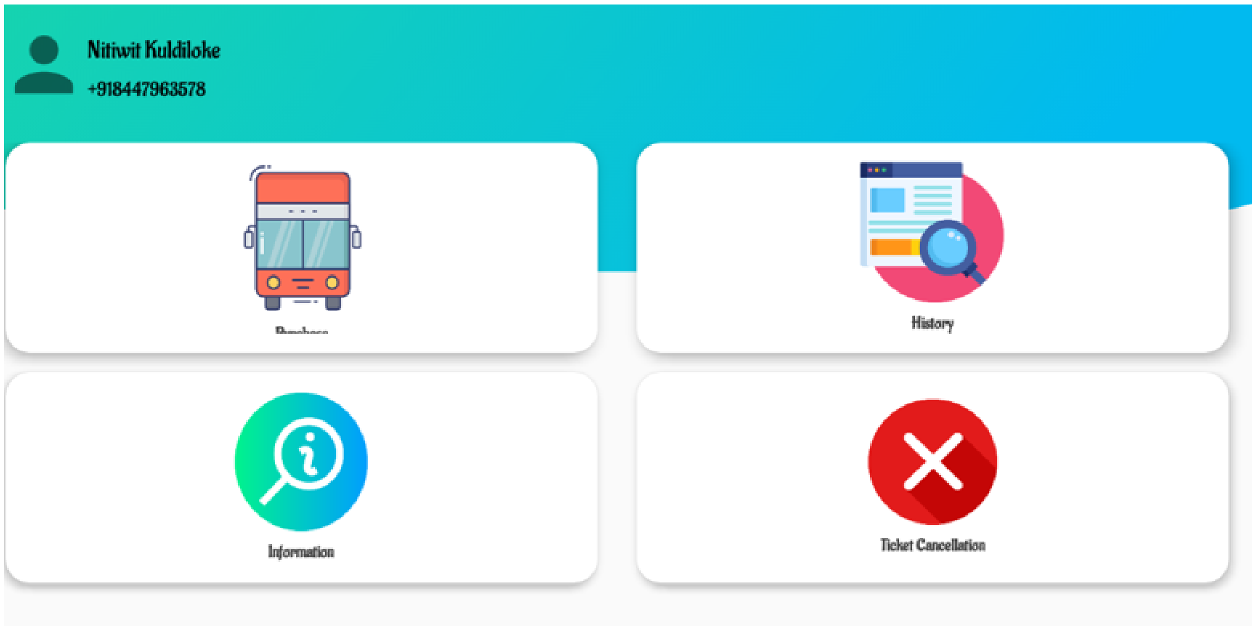


Fig: Main Menu

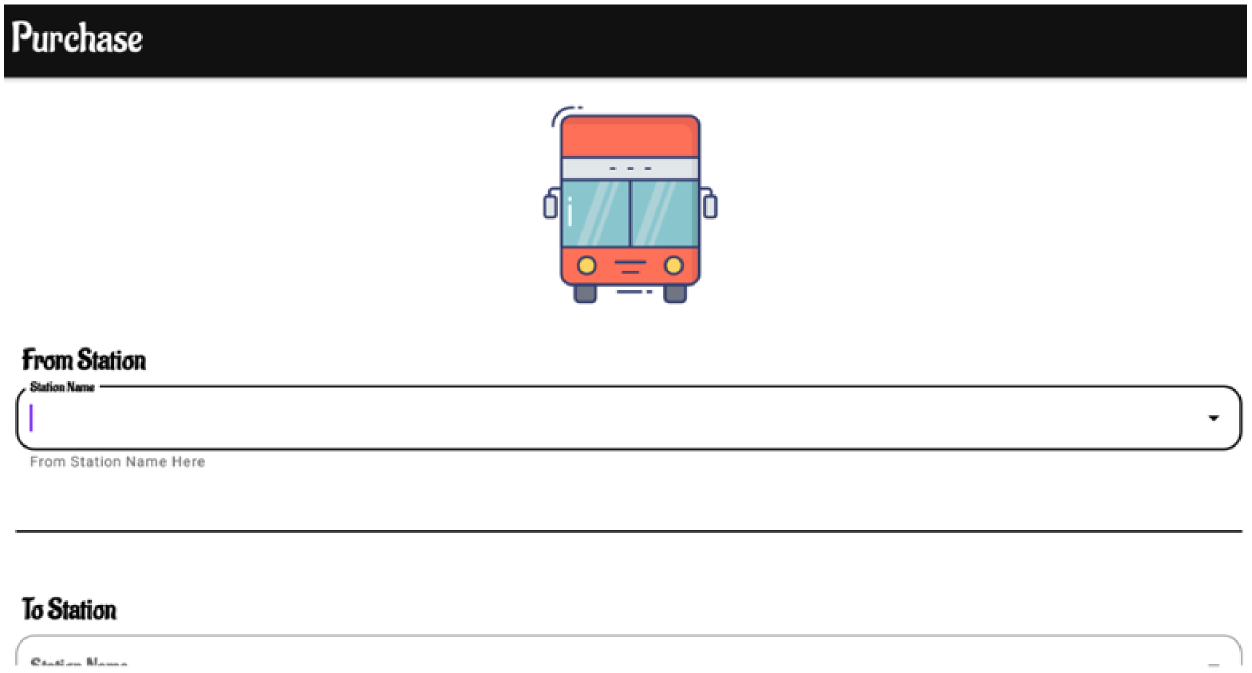


Fig: Reservation

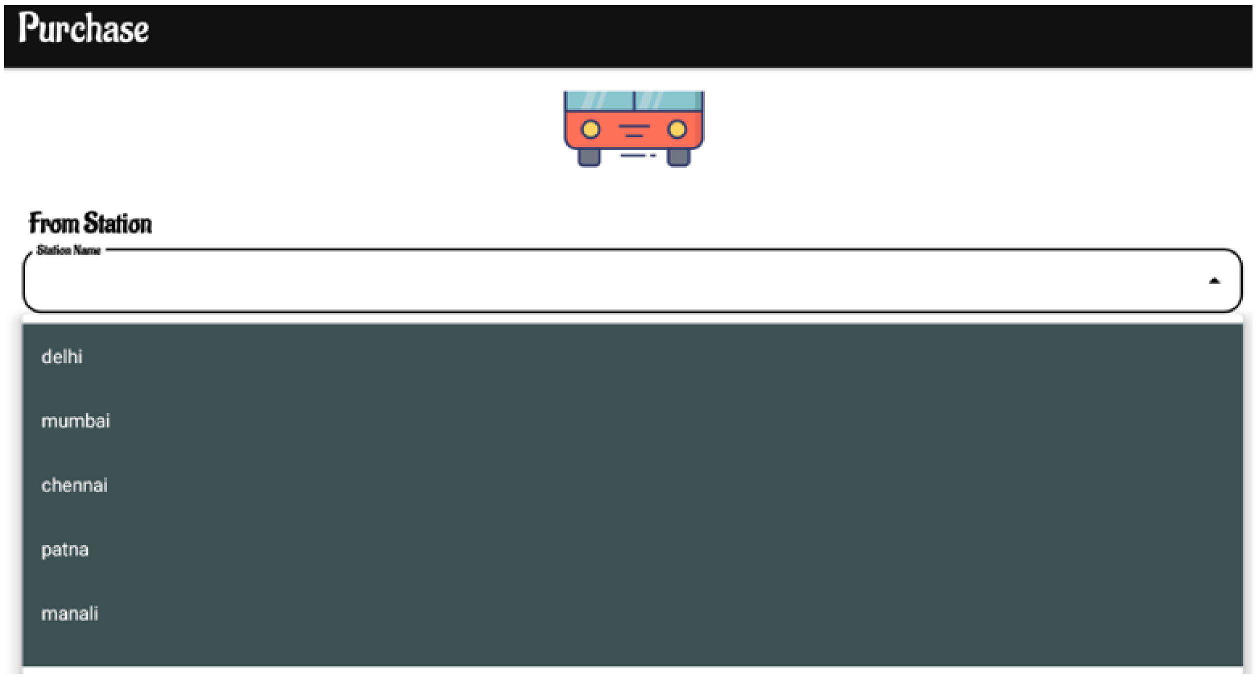
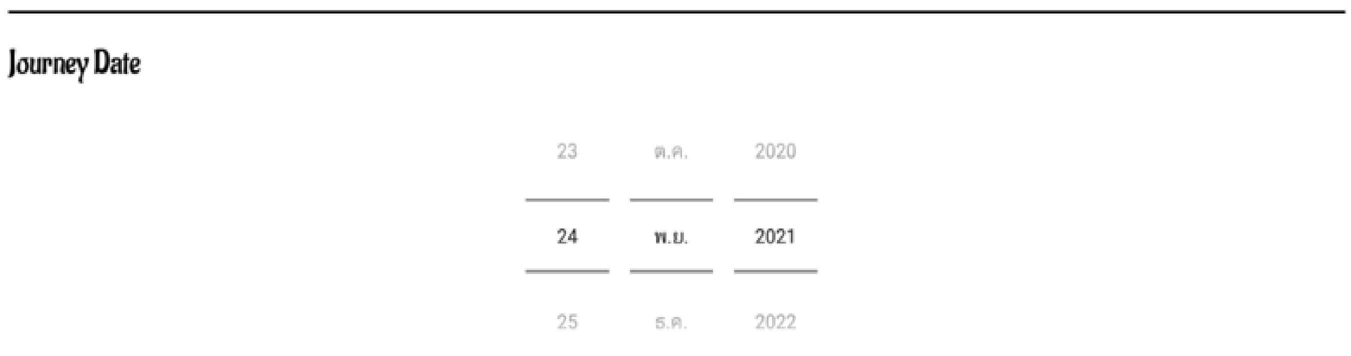


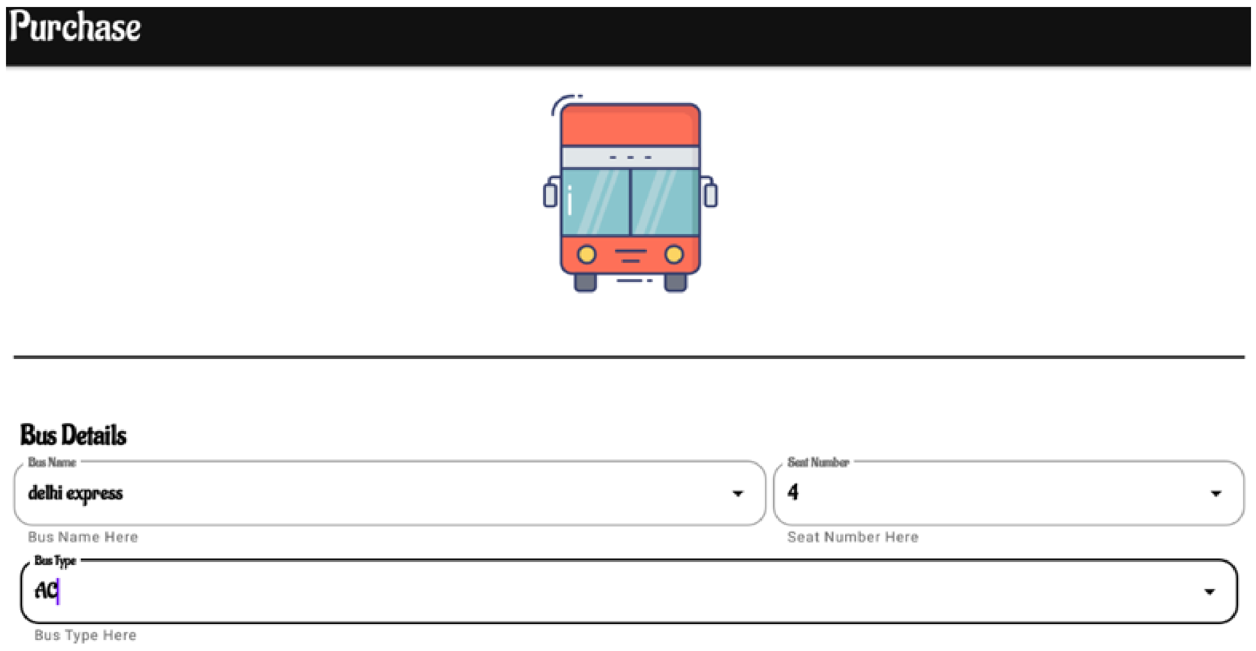
Fig: Selection of Destination



Journey Date

23	ต.ค.	2020
24	พ.ย.	2021
25	ธ.ค.	2022

Fig: Reservation date



Purchase

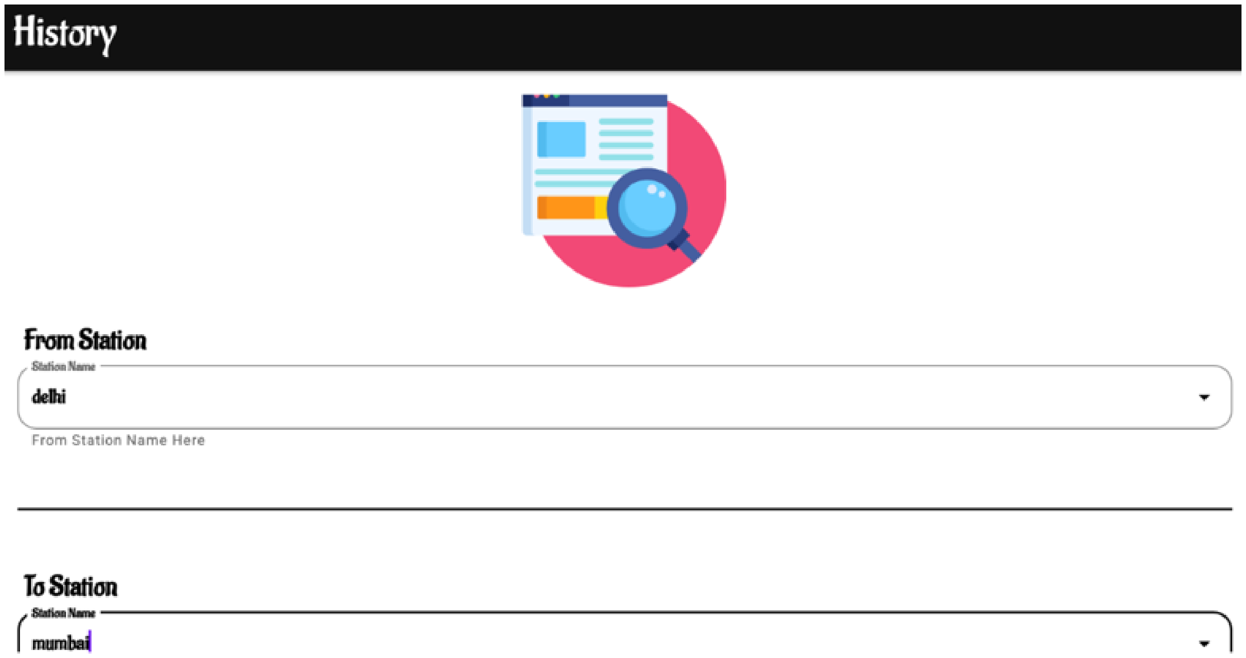
Bus Details

Bus Name: delhi express
Bus Name Here

Seat Number: 4
Seat Number Here

Bus Type: AC
Bus Type Here

Fig: Selection of Bus



History

From Station

Station Name

delhi

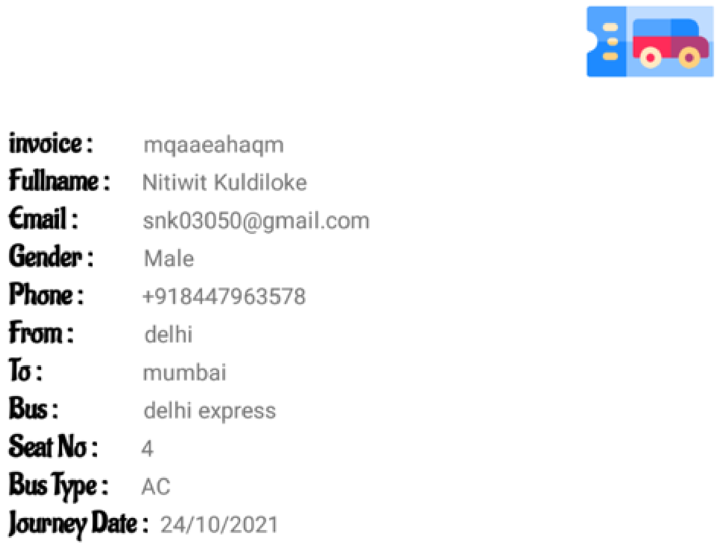
From Station Name Here

To Station

Station Name

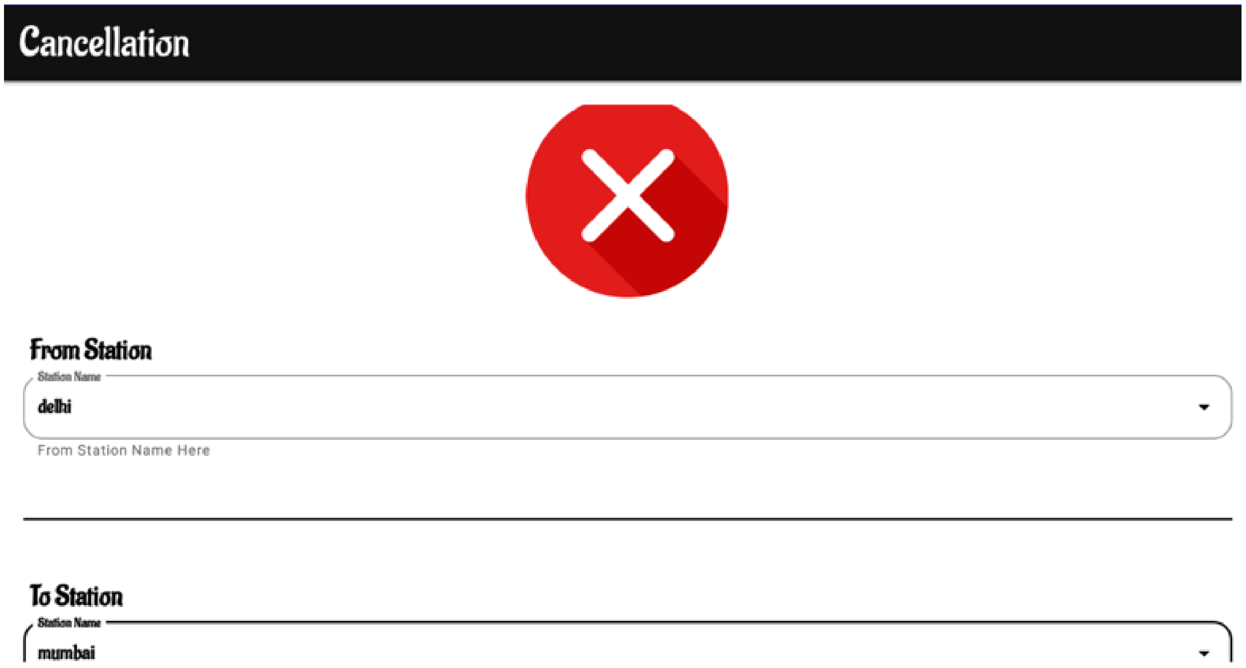
mumbai

Fig: History page



invoice :	mqaaeahaqm
Fullname :	Nitiwit Kuldiloke
Email :	snk03050@gmail.com
Gender :	Male
Phone :	+918447963578
From :	delhi
To :	mumbai
Bus :	delhi express
Seat No :	4
Bus Type :	AC
Journey Date :	24/10/2021

Fig: Ticket



Cancellation

From Station

Station Name

delhi

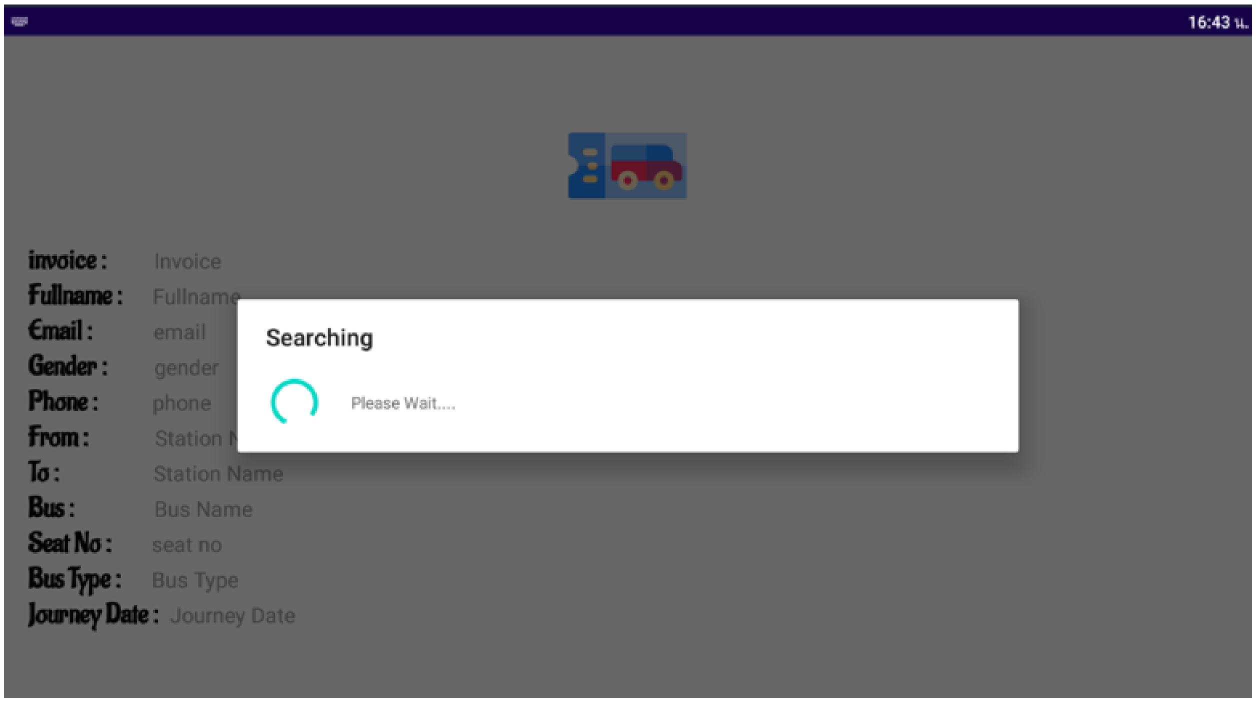
From Station Name Here

To Station

Station Name

mumbai

Fig: Cancellation Process



16:43

invoice : Invoice

Fullname : Fullname

Email : email

Gender : gender

Phone : phone

From : Station N

To : Station Name

Bus : Bus Name

Seat No : seat no

Bus Type : Bus Type

Journey Date : Journey Date

Searching

Please Wait....

Fig: Cancellation Process in Progress

Future Scope:

There is still an available to be developed in this project. The live location feature could be provided in the future if we could contact the bus company or authority to give the bus a track thus the live time-stamp of the bus will be there. And the app could be available to perform multi choice of payment and transaction such as PayPals, Credit Card, Visa, or else. The future of technology is not finished yet; they have the long road to go. We could maintain it and develop it at the same time too.

The following is the feature which could be improved in the future:

- Live tracking of the bus: For those who are waiting for the bus it will help them to time manage themselves.
- Deploy web application: Not to restricted in only mobile even those people who are more available with PC might could book the ticket by themselves
- Develop in iOS: This project is available in Android Version of phone only.
- More Security in Database: We could provide 2 factor authentication in the application.

Conclusion:

As the working process took months, we could accomplish both of our goals for our project. We could finish implementation of an application which solves the problem of ticket booking and together we could learn the concept of Java Programming and various concepts and their use in real life projects which will make us better in future projects.

References:

1. *System Analysis and Design - Overview*. (n.d.). Tutorialspoint. Retrieved November 2, 2021, from https://www.tutorialspoint.com/system_analysis_and_design/system_analysis_and_design_overview.htm
2. *What is a Data Flow Diagram*. (n.d.). Lucidchart. Retrieved November 5, 2021, from https://www.lucidchart.com/pages/data-flow-diagram
3. Shiklo, B. (2021, June 23). *8 Software Development Models: Sliced, Diced and Organized in Charts*. ScienceSoft. Retrieved October 15, 2021, from https://www.scnsoft.com/blog/software-development-models
4. Grzelak M., Napierała Ł., Napierała Ł., Karovič V., Ivanochko I. (2020) Bus Ticket Reservation System Agile Methods of Projects Management. In: Barolli L., Nishino H., Miwa H. (eds) Advances in Intelligent Networking and Collaborative Systems. INCoS 2019. Advances in Intelligent Systems and Computing, vol 1035. Springer, Cham. https://doi.org/10.1007/978-3-030-29035-1_48
5. Nwakanma, Ifeanyi & Etus, Chukwuemeka & Ajere, Ikenna & Agomuo, Uchechukwu. (2015). Online Bus Ticket Reservation System. Statistics and Computing. Vol. 1.
6. Harini, K. & Saithri, A. & Shruthi, M.. (2021). Smart Digital Bus Ticketing System. 10.1007/978-981-15-8221-9_79.
7. *Inheritance in Java - Javatpoint*. (n.d.). Www.Javatpoint.Com. Retrieved October 28, 2021, from https://www.javatpoint.com/inheritance-in-java
8. *Firebase Authentication | Simple, free multi-platform sign-in*. (n.d.). Firebase. Retrieved October 28, 2021, from https://firebase.google.com/products/auth?gclid=CjwKCAiA1uKMBhAGEiwAxzvX95ovctTT0WPCmMrii0je1Dyz3K2p6GUQhwsxI20eueu3MYm9mfzqYxoCJ6EQAvD_BwE&gclsrc=aw.ds