```
package com.example.covid_19alertapp.activities;


import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.appcompat.app.AppCompatActivity;


import android.Manifest;

import android.annotation.SuppressLint;

import android.content.Context;

import android.content.Intent;

import android.content.SharedPreferences;

import android.os.Bundle;

import android.text.Editable;

import android.text.TextWatcher;

import android.util.Log;

import android.view.View;

import android.view.inputmethod.InputMethodManager;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;


import com.example.covid_19alertapp.R;

import com.example.covid_19alertapp.extras.Constants;

import com.example.covid_19alertapp.extras.LogTags;
import com.example.covid_19alertapp.extras.Permissions;
import com.google.firebase.FirebaseException;
import com.google.firebase.auth.PhoneAuthCredential;
import com.google.firebase.auth.PhoneAuthProvider;

import java.util.concurrent.TimeUnit;


public class SignUpActivity extends AppCompatActivity {

    Button btnContinue,btnHomeSignup,btnForwardSignup;
    EditText phoneNumber;
    TextView textViewTermsCond;
    public static String PHONE_NUMBER,verification;
    public static boolean ISRETURNEDFROMVERLAYOUT;
    public static SharedPreferences loginSp,userInfo;
    PhoneAuthProvider.OnVerificationStateChangedCallbacks mCallbacks;

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

        // ask permissions
        promptPermissions();
        phoneNumber = findViewById(R.id.editText_phoneNumber);
        btnContinue = findViewById(R.id.btn_continue);
        textViewTermsCond = findViewById(R.id.TextViewTerm);
        btnHomeSignup = findViewById(R.id.home_button_signup_page);
        btnForwardSignup = findViewById(R.id.forward_button_signup_page);

        loginSp =
getSharedPreferences(Constants.USER_LOGIN_INFO_SHARED_PREFERENCES,MODE_PRIVATE);
        userInfo = getSharedPreferences(Constants.USER_INFO_SHARED_PREFERENCES,MODE_PRIVATE);

        if(loginSp.getBoolean(Constants.user_login_state_shared_preference,false)){

            startActivity(new Intent(getApplicationContext(), VerificationPageActivity.class));
            finish();
        }

        mCallbacks=new PhoneAuthProvider.OnVerificationStateChangedCallbacks() {
            @Override
            public void onVerificationCompleted(@NonNull PhoneAuthCredential phoneAuthCredential) {
                Toast.makeText(getApplicationContext(),"Successful",Toast.LENGTH_SHORT).show();
            }

            @Override
            public void onVerificationFailed(@NonNull FirebaseException e) {
                Toast.makeText(getApplicationContext(),"Check Your Internet
Connection",Toast.LENGTH_SHORT).show();

                btnContinue.setEnabled(true);



            }

            @Override

            public void onCodeSent(@NonNull String s, @NonNull PhoneAuthProvider.ForceResendingToken
forceResendingToken) {

                super.onCodeSent(s, forceResendingToken);

                verification=s;




                Toast.makeText(getApplicationContext(),"Code Sent to the
Number",Toast.LENGTH_SHORT).show();

                startActivity(new Intent(getApplicationContext(), VerificationPageActivity.class));

                loginSp.edit().putBoolean(Constants.user_login_state_shared_preference,true).apply();

                btnContinue.setEnabled(true);

                finish();

            }



        };



        if(ISRETURNEDFROMVERLAYOUT)

        {

            PHONE_NUMBER=PHONE_NUMBER.substring(0,4)+" "+PHONE_NUMBER.substring(4);

            phoneNumber.setText(PHONE_NUMBER);
```

```
    ISRETURNEDFROMVERLAYOUT = false;

    btnHomeSignup.setVisibility(View.INVISIBLE);

    btnForwardSignup.setVisibility(View.VISIBLE);


    btnForwardSignup.setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            startActivity(new Intent(getApplicationContext(), VerificationPageActivity.class));


            loginSp.edit().putBoolean(Constants.user_login_state_shared_preference,true).apply();

            finish();

        }

    });

}


phoneNumber.clearFocus();

phoneNumber.setSelection(phoneNumber.getText().toString().length());

phoneNumber.addTextChangedListener(new TextWatcher() {

    @Override

    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    //l

    int countB=phoneNumber.getText().toString().length(),countA=0;

    @SuppressLint("SetTextI18n")

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {

        if(phoneNumber.getText().toString().length()<5)
        {
            phoneNumber.setText("+880 ");
            phoneNumber.setSelection(phoneNumber.getText().toString().length());
        }

        countA = phoneNumber.getText().toString().length();

        if(phoneNumber.getText().toString().length()==9 && countA>countB)
        {
            phoneNumber.setText(phoneNumber.getText().toString()+"-");
            phoneNumber.setSelection(phoneNumber.getText().toString().length());
        }
        countB = countA;
        if(phoneNumber.getText().toString().length()==16)
        {
            hideSoftInput();
        }
    }
    @Override
    public void afterTextChanged(Editable s) { }
});
```

```
        phoneNumber.setOnFocusChangeListener(new View.OnFocusChangeListener() {
            @Override
            public void onFocusChange(View v, boolean hasFocus) {
                if(hasFocus) phoneNumber.setCursorVisible(true);
                else phoneNumber.setCursorVisible(false);
            }
        });

        btnContinue.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(phoneNumber.getText().toString().length()==16)  //Write a function to check phone number
validity
                {
                    PHONE_NUMBER = phoneNumber.getText().toString();
                    PHONE_NUMBER=PHONE_NUMBER.replaceAll("\\s+","");
                    System.out.println(PHONE_NUMBER);
                    userInfo.edit().putString(Constants.user_phone_no_preference,PHONE_NUMBER).apply();
                    sendSms(PHONE_NUMBER);
                    btnContinue.setEnabled(false);


                }
                else
                {
                    phoneNumber.setError("Invalid Number!");
                }
        }
    });

    textViewTermsCond.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            //Write Terms and Condition Page Function
            textViewTermsCond.setTextColor(getResources().getColor(R.color.colorInactive));
        }
    });


    btnHomeSignup.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            finish();
        }
    });


}

public void hideSoftInput() {
    View view1 = this.getCurrentFocus();
    if(view1!= null){
            InputMethodManager imm = (InputMethodManager)
getSystemService(Context.INPUT_METHOD_SERVICE);

            imm.hideSoftInputFromWindow(view1.getWindowToken(), 0);

        }

    }


    public void sendSms(String phoneNo){


        PhoneAuthProvider.getInstance().verifyPhoneNumber(

                phoneNo,        // Phone number to verify

                60,                 // Timeout duration

                TimeUnit.SECONDS,   // Unit of timeout

                this,               // Activity (for callback binding)

                mCallbacks        // OnVerificationStateChangedCallbacks

        );


    }


    /*

    permission needed at start of app

     */


    private Permissions permissions;

    private static final String[] permissionStrings = {

            Manifest.permission.ACCESS_FINE_LOCATION,

            Manifest.permission.ACCESS_BACKGROUND_LOCATION,
            Manifest.permission.ACCESS_WIFI_STATE,

            Manifest.permission.CALL_PHONE

    };


    private void promptPermissions() {


        permissions = new Permissions(this, permissionStrings, Constants.PERMISSION_CODE);


        if(!permissions.checkPermissions())

            permissions.askPermissions();


    }


    @Override

    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull
int[] grantResults) {

        //resolve unresolved permissions


        switch (requestCode){


            case Constants.PERMISSION_CODE:


                try {

                    this.permissions.resolvePermissions(permissions, grantResults);

                }catch (Exception e){

                    Log.d(LogTags.Permissions_TAG, "onRequestPermissionsResult: "+e.getMessage());
        }

        break;

    }

  }


}
```