

WEB TECHNOLOGY

MINI PROJECT

Women Safety Application

Mahima-202215018

Rasigapriya-2022115036

ABSTRACT

Women's safety remains a significant concern in contemporary society, and technology can serve as a crucial instrument in delivering effective solutions. This initiative introduces a **Women Safety Application**, created with **Android Studio**, aimed at promoting security, ease of use, and accessibility. The application merges emergency features with practical tools, rendering it a multifunctional resource for daily activities. The application incorporates a specialized emergency **SMS** alert button that transmits pre-set messages, including the user's location, to designated contacts, facilitating prompt communication in crisis situations. Additionally, a **calculator** feature is included to augment the app's practicality in everyday scenarios, thus enhancing its versatility. An **animation button** introduces a creative and interactive element, enriching the user experience. Furthermore, the app provides a secure login and registration system, enabling users to customize their settings and manage their emergency contacts effectively. The creation of this application responds to the urgent necessity for women to possess a dependable safety resource at their fingertips. It utilizes contemporary technology to offer a straightforward yet impactful solution aimed at improving safety and convenience. This project not only underscores the potential of mobile applications in tackling real-world challenges but also highlights the significance of incorporating user-friendly and adaptable features.

INTRODUCTION

Problem Statement

Safety is a fundamental human right, yet women across the globe face significant challenges in ensuring their safety, particularly in emergencies. Situations such as harassment, stalking, or sudden health issues demand immediate action and communication, but existing tools often fall short in offering the necessary functionality and ease of use. While there are safety applications available, many are limited to basic features, lack reliability, or fail to integrate practical daily utilities. This creates a significant gap, as women need a solution that not only addresses their safety concerns but is also versatile enough to become part of their everyday life.

Purpose of the application

The primary purpose of the Women Safety Application is to empower women by providing a reliable, user-friendly, and feature-rich mobile application. The app is designed to cater to critical safety needs while also integrating functionalities for daily use. By combining emergency support with engaging and practical features, the app ensures that users are well-equipped to handle emergencies and enhance their overall user experience.

This application aims to:

- Provide immediate support and communication during emergencies.
- Integrate everyday utilities to encourage regular usage.
- Offer a secure and personalized user interface to enhance usability.

Functionalities

The Women Safety Application includes a comprehensive set of features that cater to both safety and practicality:

1. Emergency SMS Alert Button

- A dedicated button that allows users to send an alert SMS to pre-saved contacts during emergencies.
- The alert message includes the user's predefined message and their real-time location to ensure prompt assistance.

2. Calculator

- A built-in calculator for everyday tasks, adding a practical utility to the app and encouraging users to engage with it regularly.

3. Animation Button

- An interactive feature that displays animations, creating an engaging user experience.
- Adds a creative and light-hearted element to the application to make it more enjoyable.

4. Login and Registration System

- A secure system that allows users to register and log in to manage their profiles and emergency contacts.
- Ensures that only authorized users have access to sensitive features like emergency alerts.

Significance of the project

This project demonstrates how mobile technology can be leveraged to address the critical safety concerns of women. The app not only provides immediate support during

emergencies but also integrates utilities that enhance its value and usability. By creating a comprehensive tool that blends safety, convenience, and engagement, this application serves as an example of how technology can empower users and solve real-world problems.

The Women Safety Application reflects the potential of innovative mobile solutions in fostering a safer and more secure environment for women, while also addressing their everyday needs.

TECH STACK

The **Women Safety Application** is developed using Android Studio, a powerful Integrated Development Environment (IDE) tailored for building Android applications. Android Studio offers an extensive suite of tools and features that simplify the development process, making it an ideal choice for creating modern, feature-rich mobile applications.

When compared to traditional technologies, such as standalone code editors or generic IDEs, **Android Studio** offers several distinct advantages that make it the preferred choice for Android app development. Being purpose-built, Android Studio provides platform-specific optimization with tools like a visual layout editor, resource management, and real-time previews, streamlining the design and development process.

It includes rich library support, such as Jetpack, which allows developers to quickly implement common functionalities like navigation, user authentication, and background services. Debugging is made efficient with features like Logcat, integrated debugging tools, and an emulator, reducing development time and improving application quality. The IDE enhances development speed with advanced code completion, refactoring tools, and inline documentation, while also minimizing errors.

Moreover, it supports seamless database integration through inbuilt solutions like SQLite, enabling straightforward data management without relying on external servers. As the official IDE for Android development, **Android Studio** benefits from extensive community support, offering a wealth of documentation and resources to address common development challenges.

About android studio

Android Studio is based on IntelliJ IDEA and serves as the official IDE for Android app development. It provides developers with tools and packages essential for creating robust, user-friendly applications.

Key features of Android Studio include:

- **Code Editor:** Offers intelligent code completion, syntax highlighting, and code analysis tools.
- **Layout Editor:** Allows drag-and-drop design with real-time previews for multiple device configurations.

- **Android Emulator:** Enables testing apps across various Android versions and device sizes.
- **Build System:** Based on Gradle, ensuring efficient and flexible app builds.

Packages and Tools Used

To implement the functionalities of the Women Safety Application, the following Android components and packages were utilized:

1. Emergency SMS Alert Button

- **android.telephony.SmsManager:** Enables sending SMS messages programmatically.
- **android.location.LocationManager:** Fetches the user's real-time location to include in the SMS message.

2. Calculator

- **android.widget.Button** and **android.widget.TextView:** Used to create interactive buttons for number inputs and display the results of calculations.
- **Custom Logic in Java/Kotlin:** Handles the calculation logic for addition, subtraction, multiplication, and division.

3. Animation Button

- **android.view.animation.Animation:** Provides support for various animation types (e.g., fade-in, rotate, slide).
- **android.graphics.drawable:** Used to create engaging visuals for the animation feature.

4. Login and Registration System

- **android.database.sqlite.SQLiteDatabase:** Acts as the inbuilt database to store user credentials and emergency contacts securely.
- **android.widget.EditText** and **android.widget.Button:** For user input and interactive registration/login actions.
- **Password Hashing (Optional):** Implemented through native libraries for additional security, ensuring user data protection.

Database details

The app uses SQLite, the lightweight and inbuilt database system in Android Studio, for managing user data. SQLite is ideal for mobile applications due to its:

- Low Overhead: Small storage footprint, ensuring the app remains efficient.
- Ease of Use: No setup required; data management can be performed with simple SQL queries.
- Security: Allows implementation of encryption for sensitive data like passwords.

In this project, SQLite is used to store:

- User Information: Login credentials (username, password).
- Emergency Contacts: Names and phone numbers of saved contacts for the alert functionality.

By using Android Studio and its associated tools, the Women Safety Application was efficiently developed with streamlined design, functionality, and database integration, ensuring a seamless and reliable user experience.

PROGRAMS

For Animation

This code defines an **AnimationActivity** class in an Android application to demonstrate various animations. It initializes UI components such as TextView, Buttons, and ImageViews, and loads animations from XML resources (e.g., fade_in, rotate, slide_in, zoom_in). The animations are triggered when specific buttons or views are clicked. For example, clicking the fade button applies a fade-in effect to the title text, while clicking the rotate image applies a rotation animation. This implementation enhances interactivity and user engagement with smooth, predefined visual effects.

```
package com.example.womensafetyapp;

import android.os.Bundle;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class AnimationActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_animation);
        TextView titleText = findViewById(R.id.titleText);
        Button fadeButton = findViewById(R.id.fadeButton);
        ImageView rotateImage = findViewById(R.id.rotateImage);
        Button slideButton = findViewById(R.id.slideButton);
        ImageView zoomImage = findViewById(R.id.zoomImage);
        Animation fadeIn = AnimationUtils.loadAnimation(this, R.anim.fade_in);
        Animation rotate = AnimationUtils.loadAnimation(this, R.anim.rotate);
        Animation slideIn = AnimationUtils.loadAnimation(this, R.anim.slide_in);
        Animation zoomIn = AnimationUtils.loadAnimation(this, R.anim.zoom_in);
        titleText.startAnimation(fadeIn);
        fadeButton.setOnClickListener(v -> titleText.startAnimation(fadeIn));
        rotateImage.setOnClickListener(v -> rotateImage.startAnimation(rotate));
        slideButton.setOnClickListener(v -> slideButton.startAnimation(slideIn));
        zoomImage.setOnClickListener(v -> zoomImage.startAnimation(zoomIn));
    }
}
```

For calculator

This code defines a **CalculatorActivity** class, implementing a basic calculator for an Android application. It initializes an EditText input field for user entries and buttons for digits, operators (+, -, *, /, %), and other functions like equal (=) and clear (C).

Each button is assigned an **OnClickListener** to handle user interactions. Number buttons append digits or a decimal point to the input field, while operator buttons store the first number and the selected operator, clearing the input field for the second number. The equal button computes the result based on the selected operator, and the clear button resets all fields.

The arithmetic operations are handled by parsing user inputs as doubles. Special cases, like division by zero, are addressed with error messages. The calculator supports addition, subtraction, multiplication, division, and modulus operations, providing accurate results and a simple interface for user interaction.

```
package com.example.womensafetyapp;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import androidx.appcompat.app.AppCompatActivity;

public class CalculatorActivity extends AppCompatActivity {

    private EditText inputField;
    private double firstNumber;
    private double secondNumber;
    private String operator;

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

        inputField = findViewById(R.id.inputField);
        Button btnAdd = findViewById(R.id.btnAdd);
        Button btnSubtract = findViewById(R.id.btnSubtract);
        Button btnMultiply = findViewById(R.id.btnMultiply);
        Button btnDivide = findViewById(R.id.btnDivide);
        Button btnModulus = findViewById(R.id.btnModulus);
        Button btnEqual = findViewById(R.id.btnEqual);
        Button btnClear = findViewById(R.id.btnClear);
        Button btnZero = findViewById(R.id.btnZero);
        Button btnOne = findViewById(R.id.btnOne);
        Button btnTwo = findViewById(R.id.btnTwo);
        Button btnThree = findViewById(R.id.btnThree);
        Button btnFour = findViewById(R.id.btnFour);
        Button btnFive = findViewById(R.id.btnFive);
        Button btnSix = findViewById(R.id.btnSix);
        Button btnSeven = findViewById(R.id.btnSeven);
        Button btnEight = findViewById(R.id.btnEight);
        Button btnNine = findViewById(R.id.btnNine);
        Button btnDot = findViewById(R.id.btnDot);
```



```

btnZero.setOnClickListener(v -> appendToInputField("0"));

btnOne.setOnClickListener(v -> appendToInputField("1"));
btnTwo.setOnClickListener(v -> appendToInputField("2"));
btnThree.setOnClickListener(v -> appendToInputField("3"));
btnFour.setOnClickListener(v -> appendToInputField("4"));
btnFive.setOnClickListener(v -> appendToInputField("5"));
btnSix.setOnClickListener(v -> appendToInputField("6"));
btnSeven.setOnClickListener(v -> appendToInputField("7"));
btnEight.setOnClickListener(v -> appendToInputField("8"));
btnNine.setOnClickListener(v -> appendToInputField("9"));
btnDot.setOnClickListener(v -> appendToInputField("."));
btnAdd.setOnClickListener(v -> handleOperatorClick("+"));
btnSubtract.setOnClickListener(v -> handleOperatorClick("-"));
btnMultiply.setOnClickListener(v -> handleOperatorClick("*"));
btnDivide.setOnClickListener(v -> handleOperatorClick("/"));
btnModulus.setOnClickListener(v -> handleOperatorClick("%"));
btnEqual.setOnClickListener(v -> calculateResult());
btnClear.setOnClickListener(v -> clearInputField());
}

private void appendToInputField(String text) {
    inputField.append(text);
}

private void handleOperatorClick(String operator) {
    firstNumber = Double.parseDouble(inputField.getText().toString());
    this.operator = operator;
    inputField.setText(""); }

private void clearInputField() {
    inputField.setText("");
    firstNumber = 0;
    secondNumber = 0;
}

private void calculateResult() {
    if (operator != null && !inputField.getText().toString().isEmpty()) {
        secondNumber = Double.parseDouble(inputField.getText().toString());
        double result;
        switch (operator) {
            case "+":
                result = firstNumber + secondNumber;

```

```

        break;

    case "-":
        result = firstNumber - secondNumber;
        break;
    case "*":
        result = firstNumber * secondNumber;
        break;
    case "/":
        if (secondNumber != 0) {
            result = firstNumber / secondNumber;
        } else {
            inputField.setText("Error");
            return;
        }
        break;
    case "%":
        result = firstNumber % secondNumber;
        break;
    default:
        inputField.setText("Error");
        return;}
inputField.setText(String.valueOf(result));
operator = null;    }}}

```

For Database connection

The DBHelper class extends **SQLiteOpenHelper** to manage a local SQLite database for storing user information, specifically usernames and passwords. In the onCreate() method, it creates a users table with columns for id, username, and password, where the id is an auto-incrementing primary key. The onUpgrade() method ensures that if the database version is updated, the existing table is dropped and recreated. The addUser() method allows for inserting new user records into the table by accepting a username and password as input, returning true upon successful insertion. The validateUser() method checks if a user with the provided username and password exists in the database, returning true if

found and false otherwise. This class provides essential functionality for handling user registration and login within the app, leveraging SQLite for local storage.

```
package com.example.womensafetyapp;

import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DBHelper extends SQLiteOpenHelper {

    private static final String DB_NAME = "WomenSafety.db";
    private static final int DB_VERSION = 1;

    public DBHelper(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("CREATE TABLE users (id INTEGER PRIMARY KEY\nAUTOINCREMENT, username TEXT, password TEXT)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS users");
        onCreate(db);
    }

    public boolean addUser(String username, String password) {
        SQLiteDatabase db = getWritableDatabase();

        db.execSQL("INSERT INTO users (username, password) VALUES (?, ?)", new\nObject[]{username, password});

        return true;
    }

    public boolean validateUser(String username, String password) {
        SQLiteDatabase db = getReadableDatabase();
```

```
Cursor cursor = db.rawQuery("SELECT * FROM users WHERE username = ? AND  
password = ?", new String[]{username, password});  
  
boolean isValid = cursor.getCount() > 0;
```

For emergency activity

The EmergencyActivity class is a key part of an Android application that allows users to manage emergency contacts and send alerts in case of an emergency. It initializes a local SQLite database, where emergency contact details (name and phone number) are stored in the Contacts table. The activity checks for permissions (such as reading contacts and sending SMS) at runtime, ensuring compatibility with Android versions M (6.0) and above. Upon launching, the UI loads stored contacts and animates buttons for adding a new contact and sending an alert. The addEmergencyContact() method allows users to input and save emergency contact details, while sendEmergencySMS() sends an SMS alert to all saved contacts with a predefined message. The contact list is dynamically updated, with the ability to delete contacts from the database. Animations are applied to enhance user interaction, and a series of user-friendly dialogs and Toast messages guide the user throughout the process. This class plays a critical role in providing users with an efficient way to manage contacts and send out emergency alerts.

```
package com.example.womensafetyapp;  
  
import android.Manifest;  
import android.content.ContentValues;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.database.Cursor;
```

```

import android.database.sqlite.SQLiteDatabase;
import android.os.Build;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.telephony.SmsManager;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.womensafetyapp.R;

public class EmergencyActivity extends AppCompatActivity {
    private static final int PERMISSIONS_REQUEST_CODE = 100;
    private LinearLayout contactListLayout;
    private SQLiteDatabase db;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_emergency);
        db = openOrCreateDatabase("EmergencyContacts", MODE_PRIVATE, null);
        db.execSQL("CREATE TABLE IF NOT EXISTS Contacts (id INTEGER PRIMARY KEY,
name
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (checkSelfPermission(Manifest.permission.READ_CONTACTS) !=
PackageManager.PERMISSION_GRANTED
                || checkSelfPermission(Manifest.permission.SEND_SMS) !=
PackageManager.PERMISSION_GRANTED) {

```

```

        requestPermissions(new String[]{
            Manifest.permission.READ_CONTACTS,
            Manifest.permission.SEND_SMS
        }, PERMISSIONS_REQUEST_CODE);
    }
}

Animation buttonClick = AnimationUtils.loadAnimation(this, R.anim.button_click);
Animation fadeIn = AnimationUtils.loadAnimation(this, R.anim.fade_in);
Button btnAddContact = findViewById(R.id.btnAddContact);
Button btnSendAlert = findViewById(R.id.btnSendAlert);
contactListLayout = findViewById(R.id.contactListLayout);
btnAddContact.startAnimation(fadeIn);
loadContacts();
btnAddContact.setOnClickListener(v -> {
    v.startAnimation(buttonClick);
    addEmergencyContact();
});
btnSendAlert.setOnClickListener(v -> {
    v.startAnimation(buttonClick);
    sendEmergencySMS();
});
}

private void loadContacts() {
    Cursor cursor = db.rawQuery("SELECT * FROM Contacts", null);
    contactListLayout.removeAllViews();
    if (cursor.moveToFirst()) {
        do {
            String name = cursor.getString(1);
            String phone = cursor.getString(2);
            addContactToUI(name, phone);
        } while (cursor.moveToNext());
    }
}

```

```

        } while (cursor.moveToNext());
    }
    cursor.close();
}

private void addEmergencyContact() {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Add Emergency Contact");
    EditText inputName = new EditText(this);
    inputName.setHint("Enter Name");
    EditText inputPhone = new EditText(this);
    inputPhone.setHint("Enter Phone Number");
    LinearLayout layout = new LinearLayout(this);
    layout.setOrientation(LinearLayout.VERTICAL);
    layout.addView(inputName);
    layout.addView(inputPhone);
    builder.setView(layout);
    builder.setPositiveButton("Save", (dialog, which) -> {
        String name = inputName.getText().toString().trim();
        String phone = inputPhone.getText().toString().trim();
        if (name.isEmpty() || phone.isEmpty()) {
            Toast.makeText(this, "Please fill out all fields.", Toast.LENGTH_SHORT).show();
            return;
        }
        ContentValues values = new ContentValues();
        values.put("name", name);
        values.put("phone", phone);
        db.insert("Contacts", null, values);
        addContactToUI(name, phone);
    });
    builder.setNegativeButton("Cancel", (dialog, which) -> dialog.cancel());
    builder.create().show();
}

```

```

}

private void addContactToUI(String name, String phone) {
    LinearLayout contactItem = new LinearLayout(this);
    contactItem.setOrientation(LinearLayout.HORIZONTAL);
    TextView contactText = new TextView(this);
    contactText.setText(String.format("%s: %s", name, phone));
    contactText.setLayoutParams(new LinearLayout.LayoutParams(
        0, LinearLayout.LayoutParams.WRAP_CONTENT, 1));
    Button deleteButton = new Button(this);
    deleteButton.setText("Delete");
    deleteButton.setOnClickListener(v -> {
        db.delete("Contacts", "phone=?", new String[]{phone});
        contactListLayout.removeView(contactItem);
    });
    contactItem.addView(contactText);
    contactItem.addView(deleteButton);
    contactListLayout.addView(contactItem);
}

private void sendEmergencySMS() {
    Cursor cursor = db.rawQuery("SELECT phone FROM Contacts", null);
    if (cursor.moveToFirst()) {
        do {
            String phone = cursor.getString(0);
            try {
                SmsManager smsManager = SmsManager.getDefault();
                smsManager.sendTextMessage(phone, null, "Emergency Alert! Please check my
location.", null, null);
            } catch (Exception e) {
                Toast.makeText(this, "Failed to send alert to " + phone,
Toast.LENGTH_SHORT).show();
            }
        }
    }
}

```



```
} while (cursor.moveToNext());
```

For login

The LoginActivity class is an integral component of the WomenSafetyApp, designed to handle user login functionality. Upon launch, it initializes a DBHelper instance to interact with the local SQLite database. The login process begins when the user clicks the "Login" button, which retrieves the username and password entered in the respective fields. These credentials are then validated by the validateUser() method from the DBHelper class. If the credentials match a stored user record, a success message is shown, and the user is redirected to the MainActivity, with the username passed as an extra in the Intent. If the credentials are invalid, an error message is displayed. Additionally, the activity provides a "Register" button, which opens the RegisterActivity, allowing users to sign up for a new account. This activity plays a crucial role in managing user authentication and directing users to the next steps in the application.

```
package com.example.womensafetyapp;

import android.content.Intent;
import android.os.Bundle;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class LoginActivity extends AppCompatActivity {
    DBHelper dbHelper;

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

```

dbHelper = new DBHelper(this);

findViewById(R.id.btnLogin).setOnClickListener(v -> {

    String username = ((EditText)
findViewById(R.id.editUsername)).getText().toString();

    String password = ((EditText)
findViewById(R.id.editPassword)).getText().toString();

    if (dbHelper.validateUser(username, password)) {

        Toast.makeText(this, "Login Successful!", Toast.LENGTH_SHORT).show();

        Intent intent = new Intent(LoginActivity.this, MainActivity.class);

        intent.putExtra("USERNAME", username); // Add username to the Intent

        startActivity(intent);

        finish();

    } else {

        Toast.makeText(this, "Invalid Credentials", Toast.LENGTH_SHORT).show();

    }

});

findViewById(R.id.btnRegister).setOnClickListener(v -> {

    startActivity(new Intent(LoginActivity.this, RegisterActivity.class));

});

}

}

```

For registration

The RegisterActivity class is a crucial part of the WomenSafetyApp, responsible for handling user registration. Upon its creation, it sets the content view to the activity_register.xml layout, which is assumed to include fields for the user to enter their username and password. The activity initializes a DBHelper instance for interacting with the SQLite database. When the user clicks the "Register" button, the activity retrieves the entered username and password from the respective EditText fields. It then attempts to add the new user to the database using the addUser() method from DBHelper. If the user is successfully registered, a Toast message informs the user of the successful registration, and the activity is closed, returning the user to the previous screen (usually the LoginActivity). If there is an error during registration, a Toast message is displayed, alerting the user to the issue. This activity ensures a smooth user registration process for the app.

```
package com.example.womensafetyapp;

import android.os.Bundle;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import com.example.womensafetyapp.R;

public class RegisterActivity extends AppCompatActivity {
    DBHelper dbHelper;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        dbHelper = new DBHelper(this);

        findViewById(R.id.btnRegister).setOnClickListener(v -> {
            String username = ((EditText)
            findViewById(R.id.editUsername)).getText().toString();

            String password = ((EditText)
            findViewById(R.id.editPassword)).getText().toString();

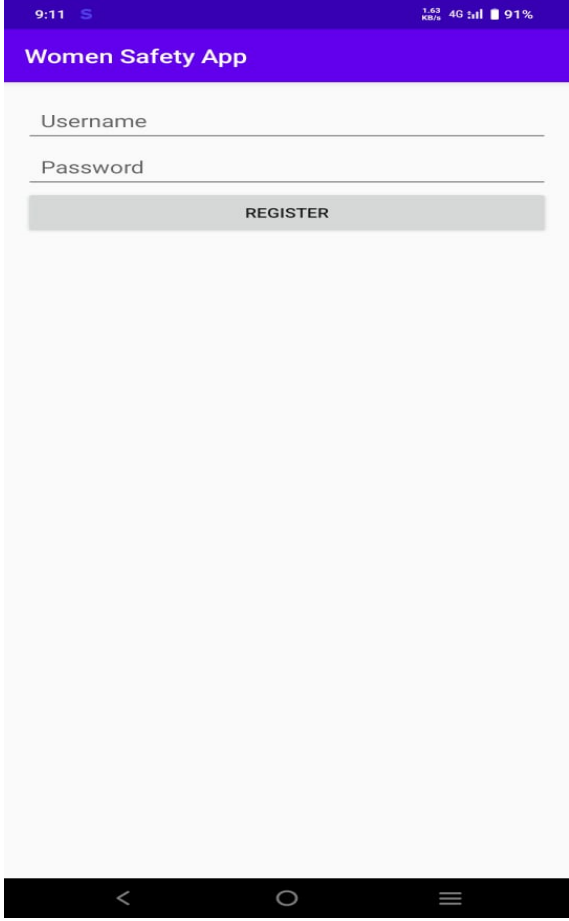
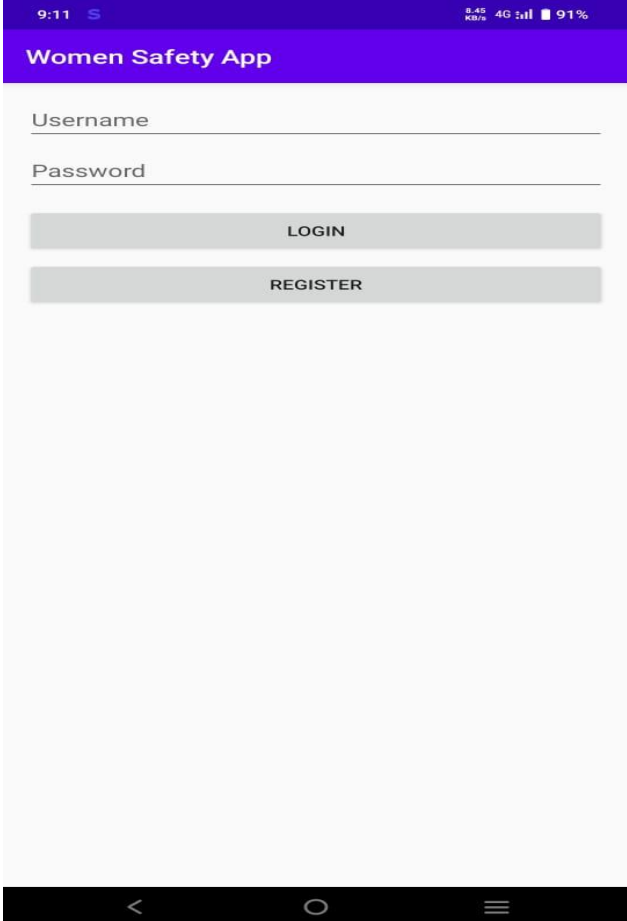
            if (dbHelper.addUser(username, password)) {
                Toast.makeText(this, "Registration Successful!", Toast.LENGTH_SHORT).show();
                finish();
            } else {
                Toast.makeText(this, "Error occurred!", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

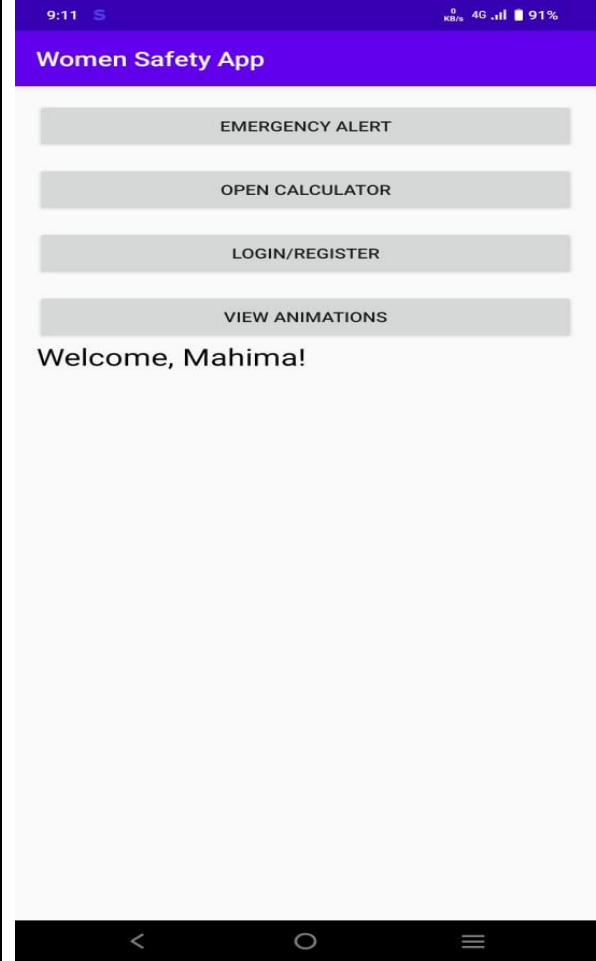

Androidmanifest.xml

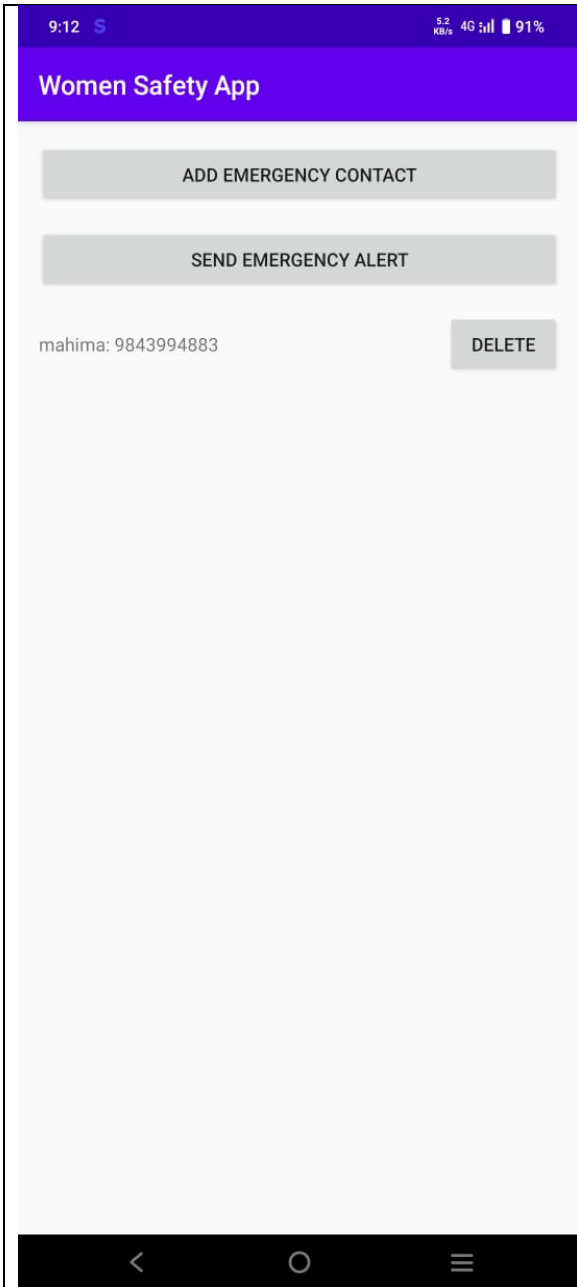
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <!-- Permissions -->
    <uses-permission android:name="android.permission.SEND_SMS" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"
/>
    <uses-permission android:name="android.permission.SEND_SMS" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.WRITE_CONTACTS" />
    <!-- Features -->
    <uses-feature android:name="android.hardware.telephony" android:required="false" />
    <uses-feature android:name="android.hardware.location.gps" android:required="false"
/>
    <application
        android:allowBackup="true"
        android:label="Women Safety App"
        android:theme="@style/Theme.WomenSafetyApp">
        <!-- Activities -->
        <activity android:name=".LoginActivity" android:exported="true" />
        <activity android:name=".RegisterActivity" android:exported="true" />
        <activity android:name=".EmergencyActivity" android:exported="true" />
        <activity android:name=".CalculatorActivity" android:exported="true" />
        <activity android:name=".AnimationActivity" android:exported="true" />
        <activity android:name=".MainActivity" android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
```

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

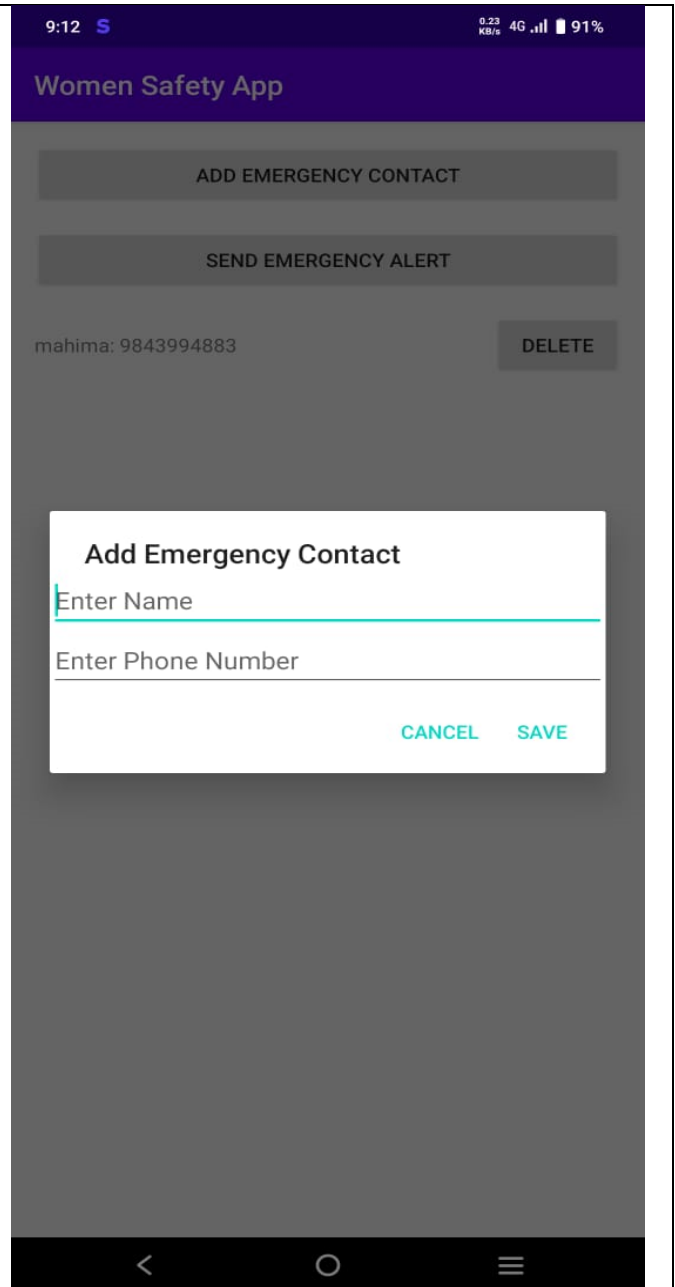
OUTPUT:

	
REGISTER	LOGIN

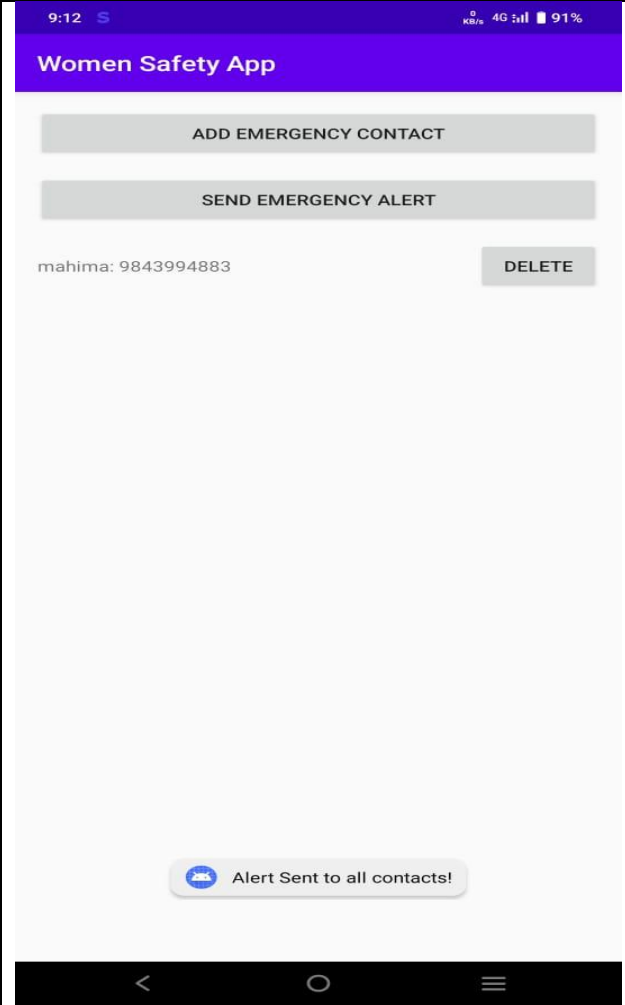
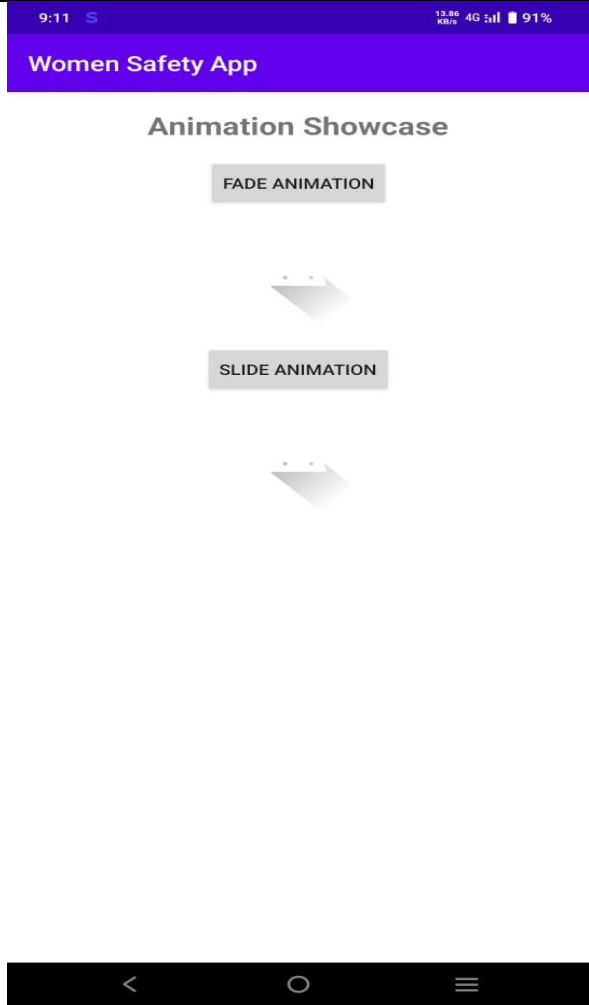
	
<p>HOME</p>	<p>CALCULATOR</p>

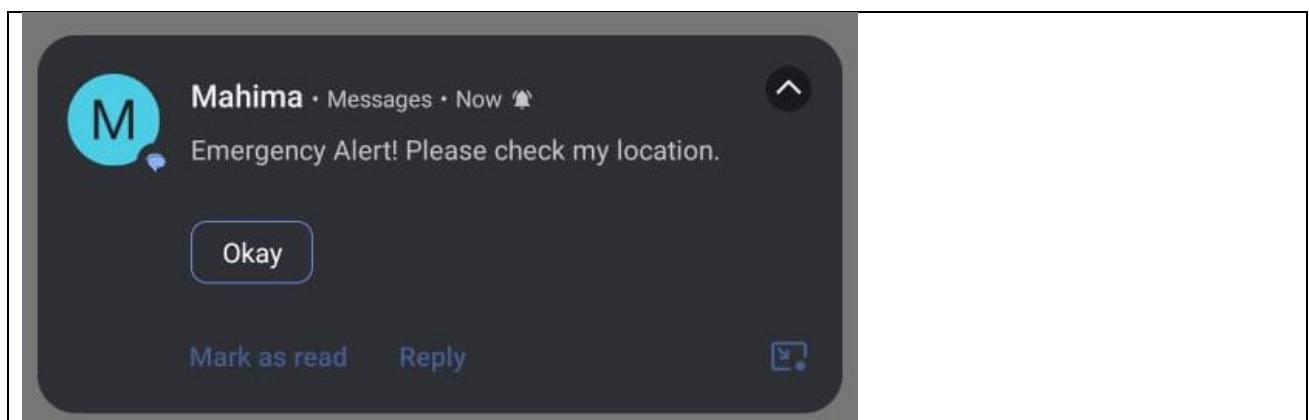


EMERGENCY MAIN



ADDING CONTACTS

	
ALERT SENT	ANIMATIONIMPLEMENTATION



RESULT

☐ **Enhanced Safety with Emergency Support:**

- The app ensures immediate assistance during emergencies through the Emergency SMS Alert feature.
- Real-time location sharing via SMS aids in prompt help from designated contacts.

☐ **Versatility with Everyday Utilities:**

- The inclusion of a Calculator encourages regular usage of the app, making it an integral tool beyond emergencies.
- An Animation Button adds an interactive and enjoyable user experience, fostering engagement.

☐ **User Authentication and Customization:**

- The secure Login and Registration system safeguards sensitive features and personal data.
- Users can customize emergency contacts and settings, enhancing the app's usability.

☐ **Technological Highlights:**

- Developed using Android Studio, leveraging its advanced tools for seamless design and robust development.
- Integration of SQLite ensures secure, efficient, and lightweight data management.

☐ **Innovative Problem-Solving:**

- Addresses a critical societal need by merging safety and practicality into a single, user-friendly mobile solution.
- Sets a precedent for leveraging mobile technology to empower women and improve safety.

☐ **Scalability and Adaptability:**

- The modular design using Android packages (e.g., SmsManager, LocationManager, Animation) enables future enhancements and customization.
- SQLite's lightweight and efficient database handling makes the app suitable for wider adoption.