I CHAPTER

ABSTRACT

In an era where efficient resource management is crucial, "Thuli" emerges as a pioneering Android application designed to revolutionize water supply systems. Thuli serves as a robust platform connecting water suppliers and consumers, facilitating seamless communication and resource optimization. This innovative solution leverages modern technology to address the challenges associated with water distribution by providing real-time updates and notifications to users. Thuli offers a user-friendly interface for both water suppliers and consumers. Water suppliers can efficiently manage and monitor the water distribution network, track supply levels, and identify potential issues. On the consumer side, the app keeps users informed about scheduled water supply, outage notifications, and any relevant announcements.

Thuli ensures that consumers receive timely updates about their water supply status, helping them plan and manage their water usage more effectively. The application sends automated notifications for scheduled maintenance, unexpected outages, or any critical information related to water supply. Usage Monitoring: Thuli allows consumers to track their water consumption patterns, encouraging responsible water usage and conservation efforts. Thuli allows consumers to track their water consumption patterns, encouraging responsible water usage and conservation efforts. Water suppliers can leverage data analytics tools within the app to identify trends, optimize distribution routes, and proactively address potential issues. The intuitive design of Thuli ensures accessibility for users of all backgrounds, promoting widespread adoption and usability. The intuitive design of Thuli ensures accessibility for users of all backgrounds, promoting widespread adoption and usability.

II CHAPTER

SYSTEM SPECIFICATIONS

HARDWARE SPECIFICATION:

Processor	: Octa-core Max 2.3GHz
Ram	: 6.0 GB
Hard Disk	: 128 GB
Device	: Phone
Connectivity	: Internet

SOFTWARE SPECIFICATION:

Front End	: XML
Back End	: JAVA
Operating System	: Android
Mobile Version	: Android 13
IDE	: Android Studio

III CHAPTER

SOFTWARE DESCRIPTION

JAVA

Java is the primary programming language for Android app development using Android Studio. It serves as the foundation for writing the core logic of applications, encompassing syntax rules, object-oriented programming principles, and control flow structures. Android Studio, the official integrated development environment (IDE) for Android, is built on IntelliJ IDEA and utilizes Java as the main language for app development. The project structure is organized through the Gradle build system, where Java code is responsible for defining classes, methods, and variables.

Android development heavily relies on the Android SDK, which provides a set of APIs and libraries for accessing device features such as the camera, sensors, and network connectivity. While XML is used to define user interfaces, Java code interacts with these layouts to handle UI components and user interactions. Java is instrumental in implementing event listeners to respond to user actions, such as button clicks or gestures.

Concurrency is managed through Java's threading mechanisms, crucial for tasks like background processing. Networking functionalities, including making HTTP requests, are implemented in Java, either through native classes like HttpURL Connection or third-party libraries like Retrofit. For local data storage, Java is used in conjunction with SQLite databases.

Java code is employed for security-related tasks, including declaring and requesting permissions to access sensitive device features. Unit testing is facilitated through Java's support for testing individual units of code, often using tools like JUnit in Android Studio. Java is also involved in ensuring version compatibility by specifying minimum and target API levels.

XML

In Android Studio, XML (eXtensible Markup Language) plays a fundamental role in defining the layout and appearance of user interfaces. XML is used to articulate the structure of the user interface, outlining the arrangement of widgets and layout containers that constitute the app's visual components. Each UI element, known as a widget, is represented by a corresponding XML tag, where attributes within the tag define properties such as size, text appearance, color, padding, and margins. Common layout containers, such as LinearLayout, RelativeLayout, FrameLayout, and ConstraintLayout, are nested using XML to organize and position widgets in specific arrangements.

XML also serves as a medium for storing resource values, encompassing strings, dimensions, colors, and styles, in dedicated resource files. Additionally, it is employed to define drawable resources, including shapes, gradients, and drawables for images, housed in the res/drawable directory. Animation sequences for UI elements are expressed in XML, allowing for visual effects like translations, rotations, and fades to enhance the user experience.

Moreover, XML is integral to data binding, where it works in tandem with code to create dynamic and reactive UIs by binding UI elements directly to data sources. The language is also employed in defining styles and themes to ensure a consistent visual design across various UI elements. For internationalization (i18n) and localization (l10n) purposes, XML is used to store translated strings in resource files.

In Android development, understanding how to structure and utilize XML is crucial for designing responsive and visually cohesive user interfaces. The XML files are typically organized in specific directories within the res folder, such as res/layout for layouts, res/values for resource values, and other appropriate directories for specific resources, facilitating a systematic approach to app development in Android Studio.

FIREBASE

Firebase is a comprehensive platform provided by Google to facilitate mobile and web application development, and integrating it with Android Studio offers a range of powerful features. One key aspect is Firebase Authentication, which enables seamless integration of user authentication methods like email/password, Google Sign-In, and Facebook Login. This functionality simplifies the implementation of user sign-up, sign-in, and password recovery processes in your Android app.

For real-time data management, Firebase offers two main services. Firebase Realtime Database is a NoSQL cloud database structured as a JSON tree, ideal for applications requiring real-time updates, such as chat applications. On the other hand, Cloud Firestore is a more flexible NoSQL database, supporting richer queries and hierarchical data structures. It provides an effective solution for managing and synchronizing data in real-time across multiple connected devices.

Firebase also offers Cloud Storage, allowing you to store and serve user-generated content, such as images and videos, directly from Google Cloud Storage. This is particularly useful for handling media files in your Android application.

Additionally, Firebase provides services like Cloud Functions, Authentication, Cloud Messaging (FCM), Hosting, and more, allowing developers to build scalable and feature-rich applications. Integrating Firebase services into your Android Studio project is streamlined through the Firebase Assistant, making it relatively straightforward to set up and manage Firebase features within your app. Overall, Firebase serves as a robust backend solution, reducing the need for developers to manage complex server infrastructure and allowing them to focus more on building great user experiences.

IV CHAPTER

PROJECT AND MODULE DESCRIPTION

Project Description:

The Thuli Android App project is an innovative solution developed exclusively within the Android Studio environment, with the primary objective of optimizing water supply management for increased efficiency and effectiveness. By capitalizing on the capabilities of Android technology, the application focuses on real-time monitoring and control within existing water supply infrastructure. Specialized features include the detection of leakages and abnormalities, with the app providing immediate alerts to water authorities. This functionality not only minimizes water loss but also enhances the overall responsiveness of the water supply system.

In addition to real-time monitoring, Thuli Android App incorporates basic predictive analytics capabilities to anticipate potential fluctuations in water demand. By analyzing historical data and user patterns, the application assists water utilities in optimizing resource allocation and planning for efficient water distribution. The predictive modeling feature serves as a proactive measure to prevent shortages, ensuring a consistent and reliable water supply.

A key aspect of the project is its user-centric approach, manifesting in a user-friendly Android interface. This interface empowers end-users to actively engage in water conservation efforts by providing a personalized dashboard to monitor water usage, receive conservation tips, and report issues directly to water management authorities. This direct communication channel fosters a sense of responsibility among community members, contributing to a culture of sustainable water consumption practices.

Objectives:

- Real-time Messaging and Updates
- User-Friendly Interface for Updates
- Community Reporting
- Localized Information
- Water Conservation Tips
- Notification System
- User Engagement Analytics
- Accessibility and Inclusivity

List of Activity:

- User Authentication
- Dashboard
- Messaging and Alerts
- Localization
- Water Conservation Tips
- Notification System
- User Engagement Analytics
- User Profile
- Accessibility
- Integration

List of Activity:

User Authentication Activity:

- Implement user authentication to secure access to the application.
- Include features for user registration and login.

Admin Authentication Activity:

- Implement admin authentication to secure access to the application.
- Include features for admin registration and login.

Dashboard Activity:

- Design a user-friendly dashboard for end-users.
- Display real-time information on water supply status, consumption, and relevant alerts.

Messaging and Alerts Activity:

- Develop a module for receiving real-time messages and alerts from water management authorities.
- Enable push notifications for timely updates.

Water Conservation Tips Activity:

- Create a module to offer water conservation tips and practices.
- Include educational content to promote sustainable water usage.

Notification System Activity:

- Implement a module for managing notifications.
- Allow users to customize notification preferences.

User Engagement Analytics Activity:

- Integrate analytics tools to track user engagement.
- Collect data on user interactions with messages, updates, and features.

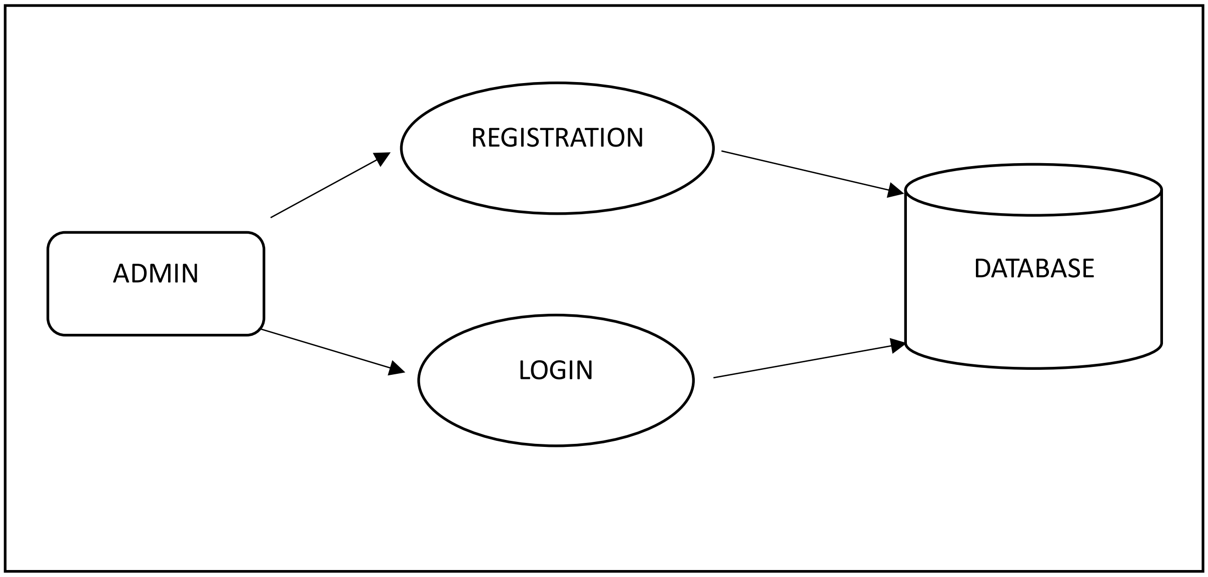
User Profile Activity:

- Implement a module for users to manage their profiles.
- Include options to update personal information and preferences.

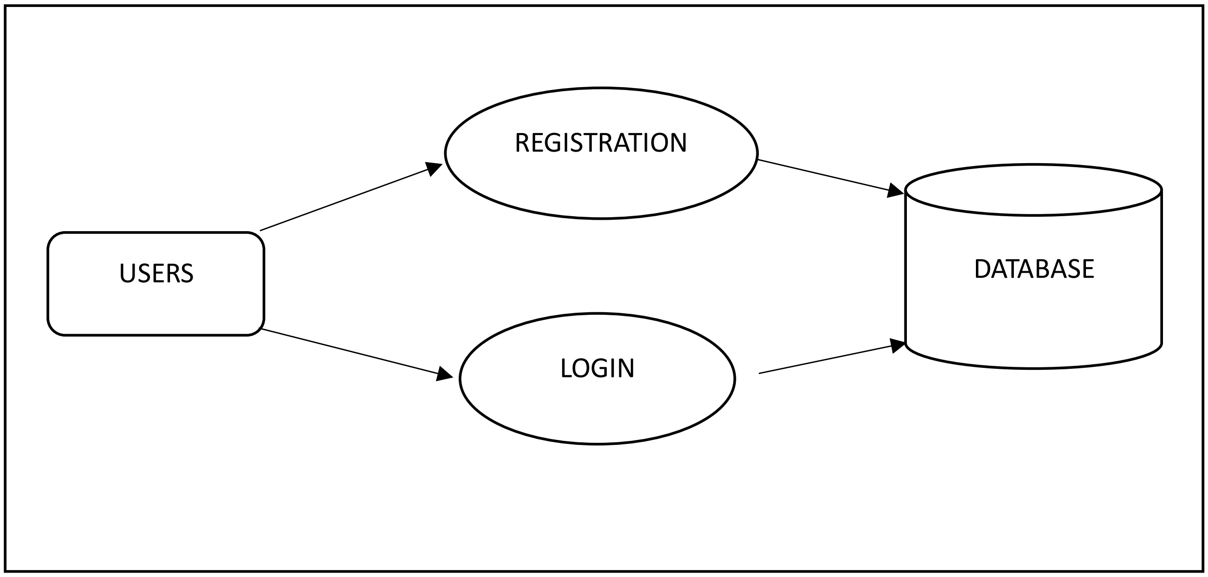
IV CHAPTER

DATA FLOW DIAGRAM:

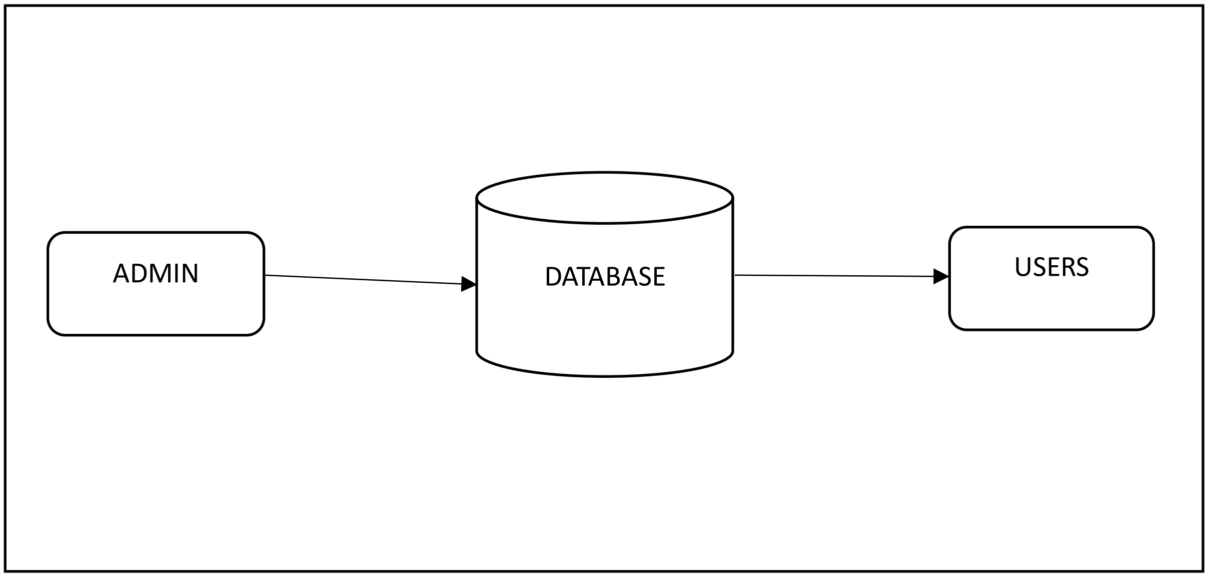
Level 0



Level 1



Level 2



IV CHAPTER

SYSTEM TESTING

Testing is the process of evaluating a system or its components with the intent to find whether it satisfies the specified requirements or not. Software testing can be stated as the process of verifying and validating that a software or application is bug free, meets the technical requirements as guided by it's design and development and meets the user requirements effectively and efficiently with handling all the exceptional and boundary cases.

TYPES OF TESTING:

Unit Testing:

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These unit shave specific behavior. The test done on these units of code is called unit test. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results. Functional and reliability testing in an Engineering Environment. Producing tests for the behavior of components (nodes and vertices) of a product to ensure their correct behavior prior to system integration.

Integration Testing:

Testing in which modules are combined and tested as a group. Modules are typically code modules, individual applications, source and destination applications on a network, etc. Integration Testing follows unit testing and precedes system testing. Testing after the product is code complete. Betas are often widely distributed or even distributed to the public at large in hopes that they will buy the final product when it is release.

System Testing:

After a system has been verified, it needs to be thoroughly tested to ensure that every component of the system is performing in accordance with the specific requirements and that it is operating as it should include when the wrong functions are requested or the wrong data is introduced.

Testing measures consist of developing a set of test criteria either for the entire system or for specific hardware, software and communications components. For an important and sensitive system such as an electronic voting system, a structured system testing program may be established to ensure that all aspects of the system are thoroughly tested.

Testing measures that could be followed include:

- Applying functional tests to determine whether the test criteria have been met
- Applying qualitative assessments to determine whether the test criteria have been met.
- Conducting tests in "laboratory" conditions and conducting tests in a variety of "real life" conditions. Conducting tests over an extended period of time to ensure systems can perform consistently. Conducting "load tests", simulating as close as possible likely conditions while using or exceeding the amounts of data that can be expected to be handled in an actual situation.

Test measures for hardware may include:

- Applying "non operating" tests to ensure that equipment can stand up to expected levels of physical handling.
- Testing "hard wired" code in hardware (firmware) to ensure its logical correctness and that appropriate standards are followed.

Tests for software components also include:

- Testing all programs to ensure its logical correctness and that appropriate design, development and implementation standards have been followed.
- Conducting "load tests", simulating as close as possible a variety of "real life" conditions using or exceeding the amounts of data that could be expected in an actual situation.
- Verifying that integrity of data is maintained throughout its required manipulation.

APPENDIX

APPENDIX A – SAMPLE SOURCE CODE:

SOURCE CODE

Main Activity.java

```
package com.example.vasa_app;
import android.content.Intent;
import android.os.Bundle;
import android.view.animation.Animation;
import androidx.appcompat.app.AppCompatActivity;
import java.util.Timer;
import java.util.TimerTask;
public class  MainActivity extends AppCompatActivity {
    Timer time;
    Animation ani;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        time = new Timer();
        time.schedule(new TimerTask() {
            @Override
            public void run() {
                Intent inte = new Intent(MainActivity.this, ac.class);
                startActivity(inte);
                finish();
            }
        }, 5000);
    }
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="200dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">
        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:fontFamily="@font/baloo_thambi"
            android:text="THULI"
            android:textSize="130dp"
            android:gravity="center"
            android:textColor="#1CA3EC"
            android:foregroundGravity="center"
            android:layout_gravity="center"
            android:textStyle="bold" />
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

Alog.java

```
package com.example.vasa_app;
import android.annotation.SuppressLint;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.FirebaseDatabase;
public class alog extends AppCompatActivity {
    LinearLayout l1;
    EditText au,ap,amn;
    TextView reg,fot;
    String aids,apass,amns;
    Button b1;
    Intent inte;
    DatabaseReference ref;
    String passs;
    String adsss;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_alog);
        ref = FirebaseDatabase.getInstance().getReferenceFromUrl("https://thuli-firebase-default-rtdb.firebaseio.com/");
        l1 = (LinearLayout) findViewById(R.id.ad_log_in);
        au = (EditText) findViewById(R.id.ad_uu);
        ap = (EditText) findViewById(R.id.ad_upass);
        amn = (EditText) findViewById(R.id.ad_mn);
        b1 = (Button) findViewById(R.id.ad_signin);
        reg = (TextView) findViewById(R.id.adtext4);
        fot = (TextView) findViewById(R.id.adtext2);
        aids = au.getText().toString().trim();
        apass = ap.getText().toString().trim();
        amns = amn.getText().toString().trim();
        l1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                l1.setBackground(getDrawable(R.drawable.ac_lin_bor1));
                l1.setScaleX(1.001F);
                l1.setScaleY(1.001F);
                                  }
```

```
au.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        au.setBackground(getDrawable(R.drawable.edittext1));
        au.setScaleX(1.001F);
        au.setScaleY(1.001F);
    }
amn.setOnClickListener(new View.OnClickListener() {
                          @SuppressLint("UseCompatLoadingForDrawables")
                          @Override
                          public void onClick(View v) {
                              amn.setBackground(getDrawable(R.drawable.edittext1));
                              amn.setScaleX(1.001F);
                              amn.setScaleY(1.001F);
                          }
ap.setOnClickListener(new View.OnClickListener() {
    @SuppressLint("UseCompatLoadingForDrawables")
    @Override
    public void onClick(View v) {
        ap.setBackground(getDrawable(R.drawable.edittext1));
        ap.setScaleX(1.001F);
        ap.setScaleY(1.001F);
    }
b1.setOnClickListener(new View.OnClickListener() {
    @SuppressLint("UseCompatLoadingForDrawables")
    @Override
    public void onClick(View v) {
        b1.setTextColor(Color.WHITE);
        b1.setScaleX(1.0001F);
        b1.setScaleY(1.0001F);
        b1.setBackground(getDrawable(R.drawable.ac_lin_but1));
        if (au.length() == 0) {
            au.setError("Enter admin id");
            au.setBackground(getDrawable(R.drawable.edittext2));
        }
        else if (ap.length() == 0) {
            ap.setError("Enter passsword");
```

```
                    ap.setBackground(getDrawable(R.drawable.edittext2));
                } else if (amn.length() == 0) {
                    amn.setError("Enter mobile number");
                    amn.setBackground(getDrawable(R.drawable.edittext2));
                }
                else if (amn.length()!=10) {
                    amn.setError("Mobile number must be in 10 digits");
                    amn.setBackground(getDrawable(R.drawable.edittext2));
                }
                else if (ap.length()!=8) {
                    ap.setError("Pasword must be in 8 digits");
                    ap.setBackground(getDrawable(R.drawable.edittext2));
                }
                else {
                    aids = au.getText().toString().trim();
                    apass = ap.getText().toString().trim();
                    amns = amn.getText().toString().trim();
                    ref.child("ad_reg_db").addListenerForSingleValueEvent(new
ValueEventListener() {
                        @Override
                        public void onDataChange(@NonNull DataSnapshot snapshot) {
                            if (snapshot.hasChild(amns)) {
                                passs =
snapshot.child(amns).child("password").getValue(String.class);
                                adsss =
snapshot.child(amns).child("admin_id").getValue(String.class);
                                if (!apass.equals(passs)) {
                                    Toast.makeText(alog.this, "Enter Valid Password",
Toast.LENGTH_LONG).show();
                                } else if (!aids.equals(adsss)) {
                                    Toast.makeText(alog.this, "Enter Valid Admin_Id",
Toast.LENGTH_LONG).show();
                                } else {
                                    if (aids.equals(adsss) && apass.equals(passs)) {
                                        Toast.makeText(alog.this, "Successfully Logged In",
Toast.LENGTH_LONG).show();
```

```
                        String fnameDB =
snapshot.child(amns).child("First_name").getValue(String.class);
                        String lnameDB =
snapshot.child(amns).child("Last_Name").getValue(String.class);
                        String dobDB =
snapshot.child(amns).child("Date_of_birth").getValue(String.class);
                        String mobDB =
snapshot.child(amns).child("mobile_number").getValue(String.class);
                        String eidDB =
snapshot.child(amns).child("Email_Id").getValue(String.class);
                        String natDB =
snapshot.child(amns).child("National").getValue(String.class);
                        String staDB =
snapshot.child(amns).child("State").getValue(String.class);
                        String disDB =
snapshot.child(amns).child("District").getValue(String.class);
                        String taDB =
snapshot.child(amns).child("Taluk").getValue(String.class);
                        String viDB =
snapshot.child(amns).child("Village").getValue(String.class);
                        String areaDB =
snapshot.child(amns).child("Location").getValue(String.class);
                        String pcDB =
snapshot.child(amns).child("Postal_code").getValue(String.class);
                        inte = new Intent(alog.this,ad_change.class);
                        inte.putExtra("fname",fnameDB);
                        inte.putExtra("lname",lnameDB);
                        inte.putExtra("dob",dobDB);
                        inte.putExtra("mob",mobDB);
                        inte.putExtra("eid",eidDB);
                        inte.putExtra("nat",natDB);
                        inte.putExtra("sta",staDB);
                        inte.putExtra("dis",disDB);
                        inte.putExtra("taluk",taDB);
                        inte.putExtra("village",viDB);
                        inte.putExtra("ad_id",adsss);
                        inte.putExtra("ad_pass",apass);
```

```
                                    inte.putExtra("area",areaDB);
                                    inte.putExtra("pc",pcDB);
                                    startActivity(inte);
                                    finish();
                              } else {
                                Toast.makeText(alog.this, "Enter Valid Mobile Number",
Toast.LENGTH_LONG).show();
                              }
                          }
                          @Override
                          public void onCancelled(@NonNull DatabaseError error) {
         });
         reg.setOnClickListener(new View.OnClickListener() {
                                 @Override
                                 public void onClick(View v) {
                                    inte = new Intent(alog.this,ad_reg.class);
                                    startActivity(inte);
                                 }
                              }
```

Activity_alog.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/ad_log"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:isScrollContainer="false"
   android:scrollbars="vertical"
   tools:context=".alog">
   <LinearLayout
      android:id="@+id/lin"
      android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:orientation="vertical"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent">
<LinearLayout
  android:id="@+id/ad_log_in"
  android:layout_width="375dp"
  android:layout_height="568dp"
  android:layout_gravity="center_horizontal"
  android:layout_marginTop="150dp"
  android:background="@drawable/ac_lin_bor"
  android:orientation="vertical">
  <ImageView
    android:id="@+id/adimg"
    android:layout_width="94dp"
    android:layout_height="110dp"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="5dp"
    android:clickable="false"
    app:srcCompat="@drawable/admin"
    android:contentDescription="TODO" />
  <TextView
    android:id="@+id/adtext1"
    android:layout_width="243dp"
    android:layout_height="35dp"
    android:layout_gravity="center_horizontal"
    android:layout_marginLeft="4dp"
    android:layout_marginTop="1dp"
    android:fontFamily="sans-serif"
    android:text="Admin Sign In Panel"
    android:textAllCaps="false"
    android:textAppearance="@style/TextAppearance.AppCompat.Display1"
    android:textColor="#000000"
    android:textSize="25sp"
    android:textStyle="bold" />
<EditText
    android:id="@+id/ad_uu"
    android:layout_width="330dp"
    android:layout_height="60dp"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="20dp"
    android:background="@drawable/edittext"
    android:cursorVisible="true"
    android:drawableLeft="@drawable/baseline_person_24"
    android:drawablePadding="10dp"
    android:ems="10"
    android:hint="Admin Id"
    android:inputType="text"
    android:onClick="ad_uuon"
    android:textColorHint="#757373"
    android:textSize="20dp"
    android:autofillHints="" />
<EditText
    android:id="@+id/ad_upass"
    android:layout_width="330dp"
    android:layout_height="60dp"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="15dp"
    android:background="@drawable/edittext"
    android:cursorVisible="true"
    android:drawableLeft="@drawable/baseline_lock_241"
    android:drawablePadding="10dp"
    android:ems="10"
    android:hint="Password"
    android:inputType="text|textPassword"
    android:onClick="ad_upasson"
    android:singleLine="false"
    android:textColorHint="#757373"
```

```
                android:textSize="20dp" />
            <Button
                android:id="@+id/ad_signin"
                android:layout_width="330dp"
                android:layout_height="wrap_content"
                android:layout_gravity="center_horizontal"
                android:layout_marginTop="20dp"
                android:background="@drawable/ac_lin_but"
                android:drawableRight="@drawable/baseline_login_white"
                android:drawablePadding="0dp"
                android:onClick="ad_signin_on"
                android:padding="10dp"
                android:paddingLeft="10dp"
                android:paddingTop="10dp"
                android:paddingRight="10dp"
                android:paddingBottom="10dp"
                android:text="  SIGN IN"
                android:textColor="#ffffff"
                android:textSize="20dp"
                android:textStyle="bold" />
        </LinearLayout>
        <FrameLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/other_activity_fr"/>
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

Ad_reg.java

```
package com.example.vasa_app;

import android.annotation.SuppressLint;

import android.content.Intent;
import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Bundle;
import android.util.Patterns;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
public class ad_reg extends AppCompatActivity {
   String selectn, selects, selectd, selectt, selectv;
   public Spinner nspinner, sspinner, dspinner, tspinner, vspinner;
   ArrayAdapter<CharSequence> narr, sarr, darr, tarr, varr;
   EditText fn, ln, age, eid, dn, pc;
   public String fns,lns,ages,mns,eids,pwds,cpwds,dns,pcs,males,females,otherss;
   Button req;
   private Uri imguri;
   private Bitmap imgstore;
```

```
    private static final int PICK_IMAGE_REQUEST = 1;

    DatabaseReference refer,ref ;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_ad_reg);

        refer = FirebaseDatabase.getInstance().getReferenceFromUrl("https://thuli-
firebase-default-rtdb.firebaseio.com/");

        ref = FirebaseDatabase.getInstance().getReferenceFromUrl("https://thuli-
firebase-default-rtdb.firebaseio.com/");

        nspinner = findViewById(R.id.ad_reg_spin_nat);

        narr = ArrayAdapter.createFromResource(this, R.array.national,
R.layout.ad_spinner_na);

narr.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_ite
m);

        nspinner.setAdapter(narr);

        nspinner.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {

            @Override

            public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {

                sspinner = findViewById(R.id.ad_reg_spin_sta);

                selectn = nspinner.getSelectedItem().toString();

                int parentId = parent.getId();

                if (parentId == R.id.ad_reg_spin_nat) {

                    switch (selectn) {
```

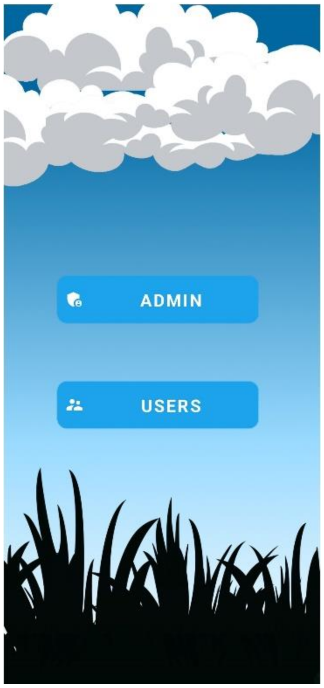
```
              @Override
              public void onNothingSelected(AdapterView<?> parent) {
        @Override
        public void onNothingSelected(AdapterView<?> parent) {
  @Override
  public void onNothingSelected(AdapterView<?> parent) {
  }
});
fn = (EditText) findViewById(R.id.ad_reg_fn);
ln = (EditText) findViewById(R.id.ad_reg_ln);
age = (EditText) findViewById(R.id.ad_reg_age);
mn = (EditText) findViewById(R.id.ad_reg_mn);
eid = (EditText) findViewById(R.id.ad_reg_em);
dn = (EditText) findViewById(R.id.ad_reg_add);
pc = (EditText) findViewById(R.id.ad_reg_pc);
req = (Button) findViewById(R.id.ad_reg_btn);
img = (ImageView) findViewById(R.id.imageView);
req.setOnClickListener(new View.OnClickListener(){
  @SuppressLint("UseCompatLoadingForDrawables")
  @Override
  public void onClick(View view){
    fns = fn.getText().toString();
    lns = ln.getText().toString();
```

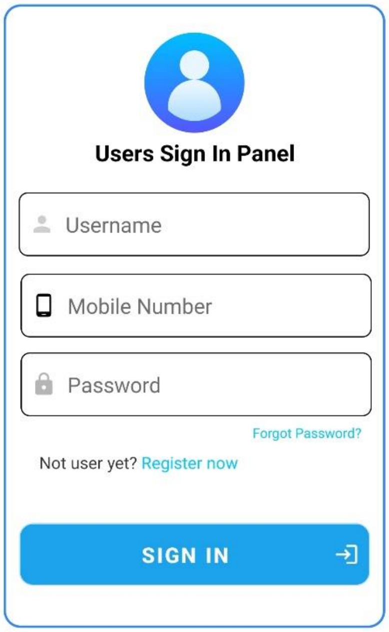
```
ages = age.getText().toString();
mns = mn.getText().toString();
eids = eid.getText().toString();
dns = dn.getText().toString();
pcs  = pc.getText().toString();
selectn = nspinner.getSelectedItem().toString();
selects = sspinner.getSelectedItem().toString();
selectd = dspinner.getSelectedItem().toString();
selectt = tspinner.getSelectedItem().toString();
selectv = vspinner.getSelectedItem().toString();
if(fn.length() == 0){
    fn.setError("Enter Your First Name");
    fn.setBackground(getDrawable(R.drawable.error_edit1));
}
else if(ln.length() == 0){
    ln.setError("Enter Your Last Name");
    ln.setBackground(getDrawable(R.drawable.error_edit1));
}
else if(age.length() == 0){
    age.setError("Enter Your Age");
    age.setBackground(getDrawable(R.drawable.error_edit1));
}
else if(mn.length() == 0){
    mn.setError("Enter Your Mobile Number");
```

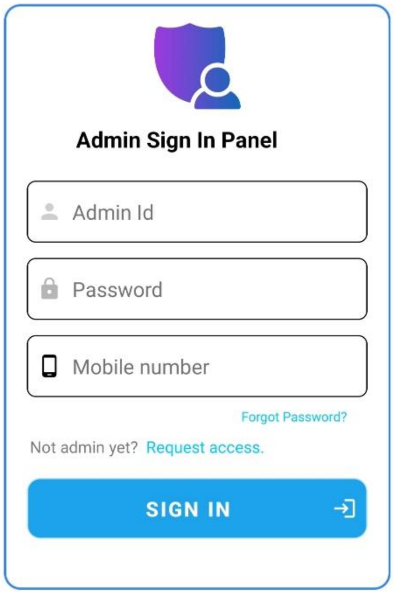
```
   mn.setBackground(getDrawable(R.drawable.error_edit1));
}
else if(mn.length() != 10){
   mn.setError("Enter Your Mobile Number in 10 Digits");
   mn.setBackground(getDrawable(R.drawable.error_edit1));
}
else if(eid.length() == 0){
   eid.setError("Enter Your Email-Id");
   eid.setBackground(getDrawable(R.drawable.error_edit1));
}
else if (!Patterns.EMAIL_ADDRESS.matcher(eids).matches()){
   eid.setError("Enter valid Email-Id");
   eid.setBackground(getDrawable(R.drawable.error_edit1));
}
else if(dn.length() == 0){
   dn.setError("Enter Your Local Address");
   dn.setBackground(getDrawable(R.drawable.error_edit1));
}
else if(pc.length() == 0){
   pc.setError("Enter Your Postal Code");
   pc.setBackground(getDrawable(R.drawable.error_edit1));
}
else{
   if (!selectv.equals("Select Your Village")){
```

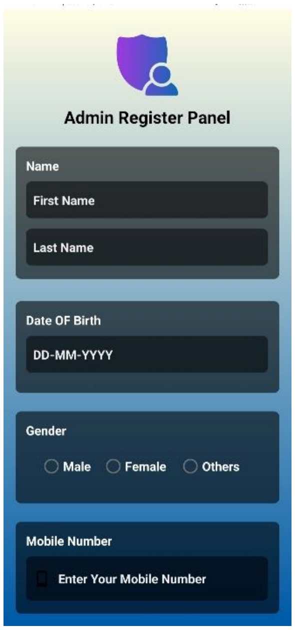
```
                   @Override
                   public void onCancelled(@NonNull DatabaseError error) {


                   }
                });



                ref.child("ad_dis_db").addListenerForSingleValueEvent(new
ValueEventListener() {
                   @Override
                   public void onDataChange(@NonNull DataSnapshot snapshot) {

refer.child("ad_dis_db").child(selectd).child("First_name").setValue(fns);

refer.child("ad_dis_db").child(selectd).child("Last_Name").setValue(lns);

refer.child("ad_dis_db").child(selectd).child("Date_of_birth").setValue(ages);

refer.child("ad_dis_db").child(selectd).child("mobile_number").setValue(mns);

refer.child("ad_dis_db").child(selectd).child("Email_Id").setValue(eids);

refer.child("ad_dis_db").child(selectd).child("National").setValue(selectn);

refer.child("ad_dis_db").child(selectd).child("State").setValue(selects);
                    });   }}}
```

APPANDIX A – SAMPLE SCREENSHOTS









Admin Register Panel

Name

First Name

Last Name

Date OF Birth

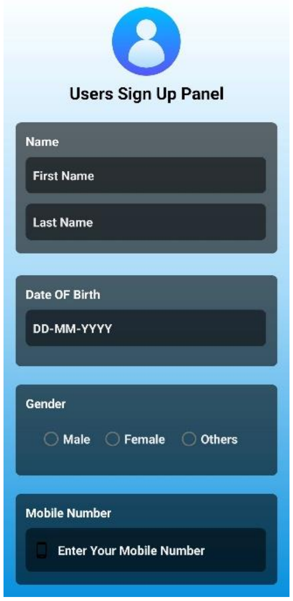
DD-MM-YYYY

Gender

○ Male ○ Female ○ Others

Mobile Number

Enter Your Mobile Number



Users Sign Up Panel

Name

First Name

Last Name

Date OF Birth

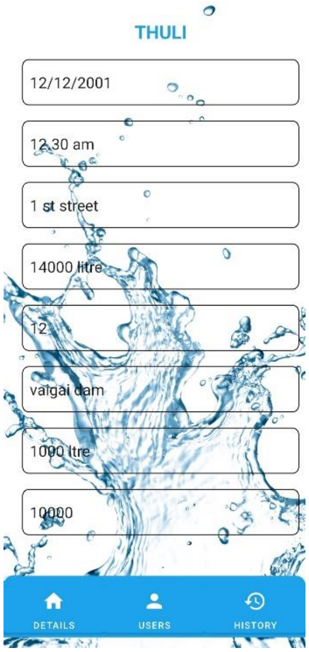
DD-MM-YYYY

Gender

○ Male ○ Female ○ Others

Mobile Number

Enter Your Mobile Number



THULI

12/12/2001

12.30 am

1 st street

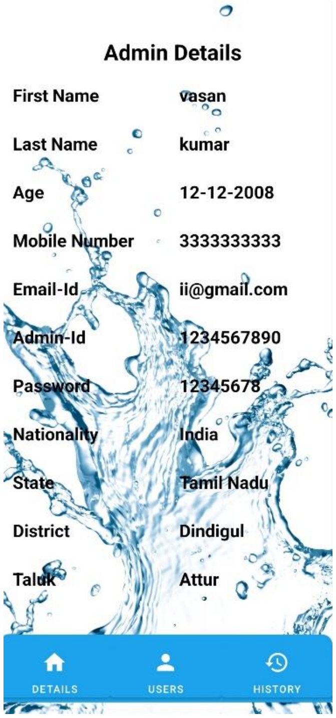
14000 litre

12

vaigai dam

1000 ltre

10000



Admin Details

First Name	vasan
Last Name	kumar
Age	12-12-2008
Mobile Number	3333333333
Email-Id	ii@gmail.com
Admin-Id	1234567890
Password	12345678
Nationality	India
State	Tamil Nadu
District	Dindigul
Taluk	Attur

CONCLUSION

In conclusion, Thuli stands as a groundbreaking solution in the realm of water supply management, seamlessly bridging the gap between providers and consumers. By harnessing the power of real-time updates and smart notifications, Thuli empowers users to take control of their water usage, plan effectively, and respond to unexpected changes. The application's focus on usage monitoring not only promotes individual responsibility but also contributes to the larger goal of sustainable water management.

BIBLIOGRAPHY

REFERENCE WEBSITE

https://www.w3schools.com/

https://www.tutorialspoint.com/

https://developer.android.com/

https://chat.openai.com/

https://stackoverflow.com/