INSTITUTE OF UNIVERSAL HIGHER STUDIES

Diploma in Information Technology

Water Bill Calculator Mobile Application

Date of Submission: 31-05-2024

R.M Gihani Madhubhashini

220100072

Table of Contents

01. Introduction

Welcome to our mobile application! The Water Bill Calculator mobile application allows you to manage and calculate your water bill easily. First, you need to register by entering a username and password. If you are already registered, simply log in. After logging in, enter your bill ID, the number of days, and the units of water consumed to calculate the total price. You can view and add new bill details.You can store the bill details you add and you can view them whenever you want and delete them if you want.To update your bill, such as changing the number of days, click on your bill details, enter the new days, and click the update button. Additionally, the app offers tips on how to save water, which you can read by clicking the "Tips" button. This makes managing your water bills simple and easy.

02. Detailed Description of Backend Implementation

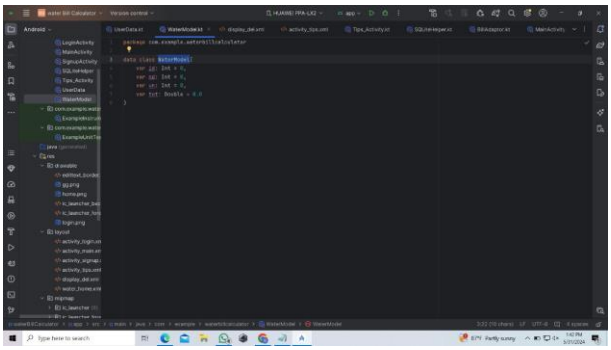
- Language: The application is developed using Kotlin.

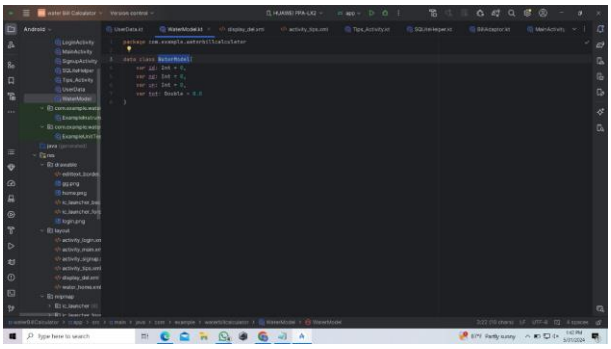
- UI Framework: XML is used for designing the user interface.

- Authentication: Firebase Authentication is used for secure user login and registration.

- Database: SQLite is used to store and retrieve bill information quickly within the app. It's lightweight and doesn't need a server, making it perfect for small to medium-sized applications.

03. Screenshots of database schema and code





Bill database

```
package com.example.waterbillcalculator


import android.annotation.SuppressLint
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class SQLDatabase(context:Context) : SQLiteOpenHelper(context,
DATABASE_NAME, null, DATABASE_VERSION) {

    companion object{
        private const val DATABASE_VERSION = 1
        private const val DATABASE_NAME = "water.db"
        private const val TBL_WATER = "tbl_water"
        private const val ID  = "id"
        private const val NO_DAYS = "nd"
        private const val UNIT = "un"
        private const val TOTAL = "tot"


    }
    override fun onCreate(p0: SQLiteDatabase?) {
        val createTblWATER = ("CREATE TABLE " + TBL_WATER + "("
                + ID + " INTEGER PRIMARY KEY," + NO_DAYS + " INTEGER," +
UNIT + " INTEGER," + TOTAL + " DOUBLE" + ")")

        p0?.execSQL(createTblWATER)
    }

    override fun onUpgrade(p0: SQLiteDatabase?, p1: Int, p2: Int) {
        p0?.execSQL("DROP TABLE IF EXISTS $TBL_WATER")
        onCreate(p0)
    }

    fun insertBill(wat: WaterModel): Long{
        val p0 = this.writableDatabase

        val contentValues = ContentValues()
        contentValues.put(ID, wat.id)
```

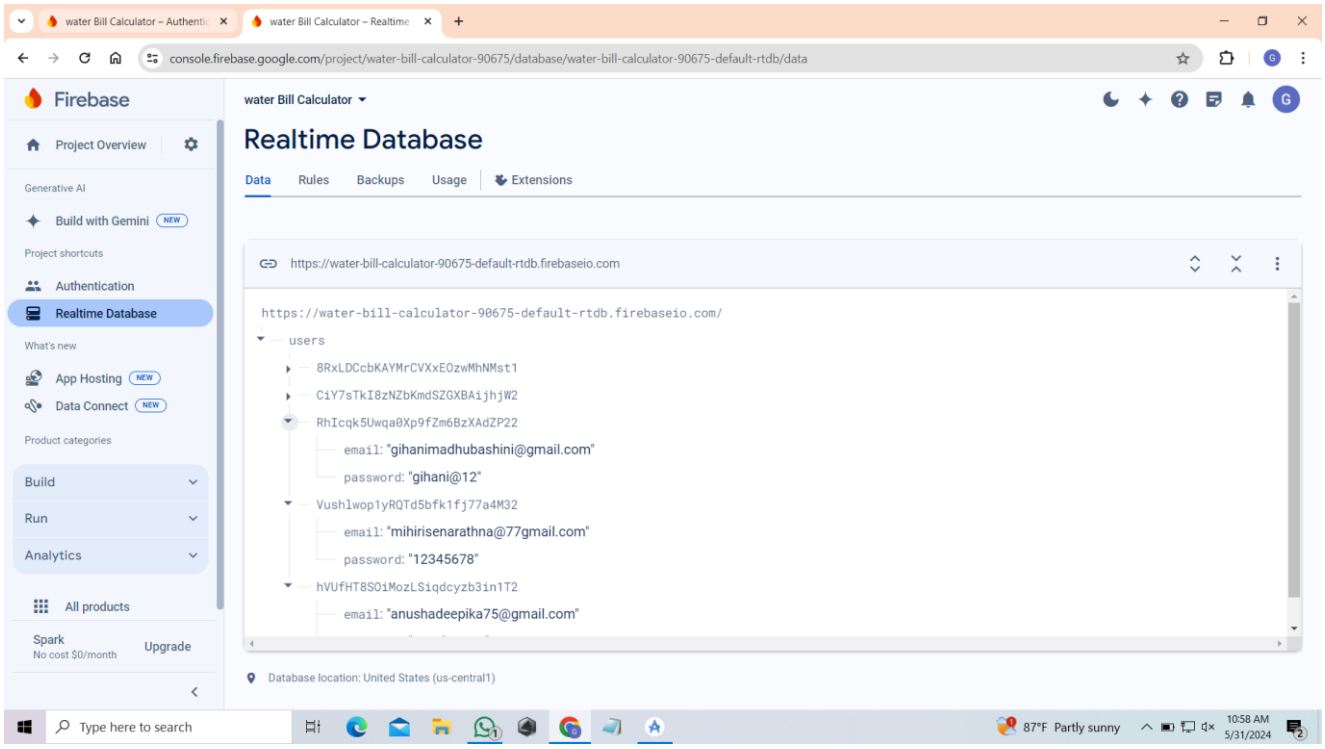
```
        contentValues.put(NO_DAYS, wat.nd)
        contentValues.put(UNIT, wat.un)
        contentValues.put(TOTAL, wat.tot)

        val success = p0.insert(TBL_WATER, null, contentValues)
        p0.close()
        return success
    }


    @SuppressLint("Range")
    fun getBill(): ArrayList<WaterModel>{
        val watList: ArrayList<WaterModel> = ArrayList()
        val selectQuery = "SELECT * FROM $TBL_WATER"
        val p0 = this.readableDatabase

        val cursor: Cursor?

        try{
            cursor = p0.rawQuery(selectQuery, null)
        } catch(e: Exception){
            e.printStackTrace()
            p0.execSQL(selectQuery)
            return ArrayList()
        }

        var id: Int
        var nd: Int
        var un: Int
        var tot: Double

        if(cursor.moveToFirst()){
            do{
                id = cursor.getInt(cursor.getColumnIndex("id"))
                nd = cursor.getInt(cursor.getColumnIndex("nd"))
                un = cursor.getInt(cursor.getColumnIndex("un"))
                tot = cursor.getDouble(cursor.getColumnIndex("tot"))

                val wat = WaterModel(id = id, nd = nd, un = un, tot = tot)
                watList.add(wat)
            } while (cursor.moveToNext())
        }

        return watList
    }

    fun updateBill(wat: WaterModel) : Int{
        val p0 = this.writableDatabase

        val contentValues = ContentValues()
        contentValues.put(ID, wat.id)
        contentValues.put(NO_DAYS, wat.nd)
        contentValues.put(UNIT, wat.un)
        contentValues.put(TOTAL, wat.tot)

        val success = p0.update(TBL_WATER, contentValues,"id=" + wat .id,
null)
        p0.close()
        return success
    }
```

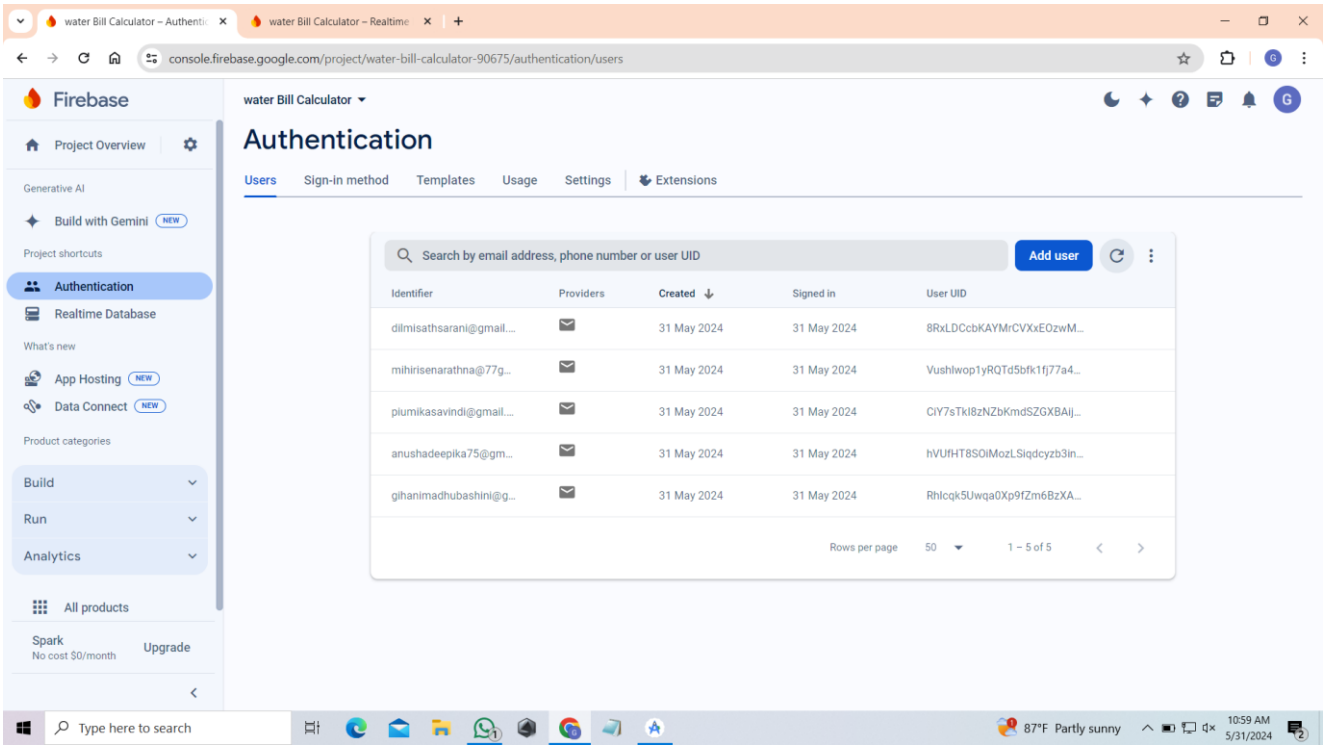
```
    fun deleteBillById(id:Int): Int{
        val p0 = this.writableDatabase

        val contentValues = ContentValues()
        contentValues.put(ID, id)

        val success = p0.delete(TBL_WATER, "id=$id", null)
        p0.close()
        return success
    }
}
```

BillAdaptor

```
package com.example.waterbillcalculator

import android.view.LayoutInflater
import android.view.ScrollCaptureCallback
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView

class BillAdaptor : RecyclerView.Adapter<BillAdaptor.BillViewHolder>(){
    private var watList: ArrayList<WaterModel> = ArrayList()
    private var onClickItem:((WaterModel) -> Unit)? = null
    private var onClickDeleteItem:((WaterModel) -> Unit)? = null

    fun addItems(items: ArrayList<WaterModel>){
        this.watList = items
        notifyDataSetChanged()
    }

    fun setOnClickItem(callback: (WaterModel)-> Unit){
        this.onClickItem = callback
    }

    fun setOnClickDeleteItem(callback: (WaterModel) -> Unit){
        this.onClickDeleteItem = callback
    }


    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int) =
BillViewHolder (
        LayoutInflater.from(parent.context).inflate(R.layout.display_del,
parent,false)
    )

    override fun onBindViewHolder(holder: BillViewHolder, position: Int) {
        val wat = watList[position]
        holder.bindView(wat)
        holder.itemView.setOnClickListener{onClickItem?.invoke(wat)}
        holder.btnDelete.setOnClickListener{onClickDeleteItem?.invoke(wat)}
    }

    override fun getItemCount(): Int {
        return watList.size
    }
```

```
    class BillViewHolder(var view: View) : RecyclerView.ViewHolder(view){
        private var id = view.findViewById<TextView>(R.id.id)
        private var nd = view.findViewById<TextView>(R.id.nd)
        private var un = view.findViewById<TextView>(R.id.un)
        private var tot = view.findViewById<TextView>(R.id.tot)
        var btnDelete = view.findViewById<TextView>(R.id.btnDelete)

        fun bindView(wat: WaterModel){
            id.text = wat.id.toString()
            nd.text = wat.nd.toString()
            un.text = wat.un.toString()
            tot.text = wat.tot.toString()
        }
    }

}
```





CRUD

```
package com.example.waterbillcalculator
import android.annotation.SuppressLint
import android.content.ContentValues.TAG
import android.content.Intent
import android.os.Bundle
import android.text.Editable
import android.text.TextWatcher
import android.util.Log
import android.widget.Button
import android.widget.EditText
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AlertDialog
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView

class MainActivity : AppCompatActivity() {
    private lateinit var id: EditText
    private lateinit var nd: EditText
    private lateinit var un: EditText
    private lateinit var tot: TextView
    private lateinit var btnAdd: Button
    private lateinit var btnView: Button
    private lateinit var btnUpdate: Button

    private var sqliteHelper = SQLDatabase(this)
```

```
    private lateinit var recyclerView: RecyclerView
    private var adaptor: BillAdaptor? = null
    private var wat:WaterModel? = null


    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.water_home)


        val btnGoToMain = findViewById<Button>(R.id.calculate_button)
        btnGoToMain.setOnClickListener {
            setContentView(R.layout.activity_main)
            id = findViewById(R.id.id)
            nd = findViewById(R.id.nd)
            un = findViewById(R.id.un)
            tot = findViewById(R.id.tot)
            btnAdd = findViewById(R.id.btn_add)
            btnView = findViewById(R.id.btn_view)
            btnUpdate = findViewById(R.id.btn_update)

            initRecycleView()

            btnAdd.setOnClickListener{ addBill() }
            btnView.setOnClickListener{ getBill() }
            btnUpdate.setOnClickListener{ updateBill() }

            adaptor?.setOnClickItem {

Toast.makeText(this,it.id.toString(),Toast.LENGTH_SHORT).show()
                id.setText(it.id.toString())
                nd.setText(it.nd.toString())
                un.setText(it.un.toString())

                wat = it
            }
            adaptor?.setOnClickDeleteItem {
                deleteBill(it.id)
            }


            un.addTextChangedListener(object: TextWatcher{
                override fun beforeTextChanged(p0: CharSequence?, p1: Int,
p2: Int, p3: Int) {}

                override fun onTextChanged(p0: CharSequence?, p1: Int, p2:
Int, p3: Int) {}

                override fun afterTextChanged(p0: Editable?) {
                    val s = ""
                    Log.i(TAG, "afterTextChanged $s")
                    computeWater()
                }

            })
        }
        val btnGoToTips = findViewById<Button>(R.id.tips)
        btnGoToTips.setOnClickListener {
            startActivity(Intent(this,Tips_Activity::class.java))
```

```
        }


    }

    private fun getBill() {
        val watList = sqliteHelper.getBill()
        Log.e("pppp", "${watList.size}")

        adaptor?.addItems(watList)
    }

    private fun addBill(){
        val id = id.text.toString().toInt()
        val nd = nd.text.toString().toInt()
        val un = un.text.toString().toInt()
        val tot = tot.text.toString().toDouble()

        val wat = WaterModel(id = id, nd = nd, un = un, tot = tot)

        val status = sqliteHelper.insertBill(wat)
        if(status > -1){
            Toast.makeText(this,"Bill Added...", Toast.LENGTH_SHORT).show()
            clearText()
            getBill()
        }else{
            Toast.makeText(this,"Record Not Saved...",
Toast.LENGTH_SHORT).show()
        }
    }
    private fun updateBill(){
        val id = id.text.toString().toInt()
        val nd = nd.text.toString().toInt()
        val un = un.text.toString().toInt()
        val tot = tot.text.toString().toDouble()

        if(id == wat?.id && nd == wat?.nd && un == wat?.un && tot ==
wat?.tot){
            Toast.makeText(this, "Record Not Changed...",
Toast.LENGTH_SHORT).show()
            return
        }

        if(wat == null) return

        val wat = WaterModel(id = wat!!.id, nd = nd, un = un, tot = tot)
        val status = sqliteHelper.updateBill(wat)
        if(status > -1){
            clearText()
            getBill()
        } else{
            Toast.makeText(this,"Update Failed", Toast.LENGTH_SHORT).show()
        }
    }

    private fun deleteBill(id:Int){
        val builder = AlertDialog.Builder(this)
        builder.setMessage("Are you sure you want to delete the
record?...")
        builder.setCancelable(true)
        builder.setPositiveButton("Yes"){dialog,_ ->
```

```
            sqliteHelper.deleteBillById(id)
            getBill()
            dialog.dismiss()
        }
        builder.setNegativeButton("No"){dialog,_ ->
            dialog.dismiss()
        }
        var alert = builder.create()
        alert.show()

    }

    private fun clearText() {
        id.setText("")
        nd.setText("")
        un.setText("")
        tot.setText("")
    }
    private fun initRecycleView(){
        recyclerView = findViewById(R.id.recycleView)
        recyclerView.layoutManager = LinearLayoutManager(this)
        adaptor = BillAdaptor()
        recyclerView.adapter = adaptor
    }

    private fun computeWater() {
        var units = if (un.text.isNotEmpty()) un.text.toString().toInt()
else 0

        var totalCost = 0.0
        val rate1 = 80.00
        val rate2 = 110.00
        val rate3 = 160.00
        val rate4 = 210.00
        val rate5 = 270.00
        val range1 = 10
        val range2 = 25
        val range3 = 40
        val range4 = 75
        val range5 = 100

        if (units <= range1) {
            totalCost = units * rate1
        } else {
            totalCost += range1 * rate1
            units -= range1

            if (units <= (range2 - range1)) {
                totalCost += units * rate2
            } else {
                totalCost += (range2 - range1) * rate2
                units -= (range2 - range1)

                if (units <= (range3 - range2)) {
                    totalCost += units * rate3
                } else {
                    totalCost += (range3 - range2) * rate3
                    units -= (range3 - range2)

                    if (units <= (range4 - range3)) {
                        totalCost += units * rate4
```

```
                    } else {
                        totalCost += (range4 - range3) * rate4
                        units -= (range4 - range3)

                        totalCost += units * rate5
                    }
                }
            }
        }

        tot.text = totalCost.toString()
    }


}
```

05. Key Functionalities and Code Snippets

Firebase Authentication

signup

```
package com.example.waterbillcalculator

import android.content.Intent
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat
import com.example.waterbillcalculator.databinding.ActivitySignupBinding
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.database.DatabaseReference
import com.google.firebase.database.FirebaseDatabase

class SignupActivity : AppCompatActivity() {

    private lateinit var binding: ActivitySignupBinding
    private lateinit var firebaseAuth: FirebaseAuth
    private lateinit var databaseReference: DatabaseReference

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivitySignupBinding.inflate(layoutInflater)
        setContentView(binding.root)

        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main))
{ v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }

        firebaseAuth = FirebaseAuth.getInstance()
        databaseReference =
FirebaseDatabase.getInstance().reference.child("users")

        binding.signupButton.setOnClickListener {
            val signupUsername = binding.signupUsername.text.toString()
            val signupPassword = binding.signupPassword.text.toString()


            if (signupUsername.isNotEmpty() && signupPassword.isNotEmpty())
{
                signUpUser(signupUsername, signupPassword)
            } else {
                Toast.makeText(this@SignupActivity, "All fields are
mandatory", Toast.LENGTH_SHORT).show()
            }
        }

        binding.loginRedirect.setOnClickListener {
            startActivity(Intent(this@SignupActivity,
```

```
LoginActivity::class.java))
            finish()
        }
    }

    private fun signUpUser(email: String, password: String) {
        firebaseAuth.createUserWithEmailAndPassword(email,
password).addOnCompleteListener(this) { task ->
            if (task.isSuccessful) {
                val userId = firebaseAuth.currentUser?.uid

                val user = User(email, password)

                userId?.let {

databaseReference.child(it).setValue(user).addOnCompleteListener { task ->
                        if (task.isSuccessful) {
                            Toast.makeText(this, "Sign Up Successful",
Toast.LENGTH_SHORT).show()
                            val intent = Intent(this,
LoginActivity::class.java)
                            startActivity(intent)
                            finish()
                        } else {
                            Toast.makeText(this, "Failed to store user
data: ${task.exception?.message}", Toast.LENGTH_SHORT).show()
                        }
                    }
                }
            } else {
                Toast.makeText(this, "Sign Up Failed:
${task.exception?.message}", Toast.LENGTH_SHORT).show()
            }
        }
    }

    data class User(val email: String, val password: String)
}
```

Login

```
package com.example.waterbillcalculator

import android.content.Intent
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat
import com.example.waterbillcalculator.databinding.ActivityLoginBinding
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.database.DatabaseReference
import com.google.firebase.database.FirebaseDatabase

class LoginActivity : AppCompatActivity() {

    private lateinit var binding: ActivityLoginBinding
    private lateinit var firebaseAuth: FirebaseAuth
    private lateinit var databaseReference: DatabaseReference
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityLoginBinding.inflate(layoutInflater)
        setContentView(binding.root)

        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main))
{ v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }

        firebaseAuth = FirebaseAuth.getInstance()
        databaseReference =
FirebaseDatabase.getInstance().reference.child("users")

        binding.loginButton.setOnClickListener {
            val loginUsername = binding.loginUsername.text.toString()
            val loginPassword = binding.loginPassword.text.toString()

            if (loginUsername.isNotEmpty() && loginPassword.isNotEmpty()) {
                loginUser(loginUsername, loginPassword)
            } else {
                Toast.makeText(this@LoginActivity, "All fields are
mandatory", Toast.LENGTH_SHORT).show()
            }
        }

        binding.signupRedirect.setOnClickListener {
            startActivity(Intent(this@LoginActivity,
SignupActivity::class.java))
            finish()
        }
    }

    private fun loginUser(email: String, password: String) {
        firebaseAuth.signInWithEmailAndPassword(email,
password).addOnCompleteListener(this) { task ->
            if (task.isSuccessful) {
                val userId = firebaseAuth.currentUser?.uid

                userId?.let {
                    databaseReference.child(it).get().addOnCompleteListener
{ task ->
                        if (task.isSuccessful) {

                            startActivity(Intent(this,
MainActivity::class.java))
                            finish()

                        } else {
                            Toast.makeText(this, "Failed to retrieve user
data: ${task.exception?.message}", Toast.LENGTH_SHORT).show()
                        }
                    }
                }
            } else {
                Toast.makeText(this, "Login Failed:
${task.exception?.message}", Toast.LENGTH_SHORT).show()
```

```
                }
            }
        }
}
```