[2CEIT5PE18: MOBILE APPLICATION DEVELOPMENT]

Practical: 7

AIM-Develop an Android application that retrieves person data in JSON format from an internet API and stores the retrieved data in an SQLite database.

Submitted By: KANSAGARA KRISH
Enrollment number: 23012011026

Department of Computer Engineering/Information Technology

Practical-7

MainActivity.kt

```
package com.example.mad_23012011026_practical7

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.util.Log
import android.widget.Toast
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import
com.google.android.material.floatingactionbutton.FloatingActionBut
ton
import kotlinx.coroutines.CoroutineScope
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.launch
import kotlinx.coroutines.withContext
import org.json.JSONArray

class MainActivity : AppCompatActivity() {

    private lateinit var dbHelper: DatabaseHelper
    private lateinit var adapter: PersonAdapter
    private val tag = "MainActivity"

    private lateinit var recyclerView: RecyclerView
    private lateinit var fabRefresh: FloatingActionButton

    override fun onCreate(savedInstanceState: Bundle?)
        { super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        recyclerView = findViewById(R.id.recyclerView)
        fabRefresh = findViewById(R.id.fabRefresh)
        dbHelper = DatabaseHelper(this)

        recyclerView.layoutManager = LinearLayoutManager(this)
        adapter = PersonAdapter(mutableListOf())
        recyclerView.adapter = adapter

        loadFromDb()
        fabRefresh.setOnClickListener
            { fetchAndSaveData()
        }
    }

    private fun loadFromDb() {
        val list = dbHelper.allPersons
        adapter.updateItems(list)
    }

    private fun fetchAndSaveData()
        { CoroutineScope(Dispatchers.IO).launch {
            try {
                val        url        =        "https://api.json-
generator.com/templates/x1QbX-JkAS57/data"
                val                    token                    =
"mm53tej3fypzbc4exljs9adhqmo4oj6eqkqg5656"

                val   result   =   HttpRequest.makeServiceCall(url,
token)

                if (!result.isNullOrEmpty())
                    { parseAndSave(result)
                    withContext(Dispatchers.Main) {
                        loadFromDb()
                        Toast.makeText(this@MainActivity,     "Data
refreshed", Toast.LENGTH_SHORT).show()
                    }
                } else {
                    withContext(Dispatchers.Main) {
                        Toast.makeText(this@MainActivity, "No data
from server", Toast.LENGTH_SHORT).show()
                    }
                }

            } catch (e: Exception) {
                Log.e("MainActivity",  "Exception  fetching  data:
${e.message}")
                withContext(Dispatchers.Main)
                    { Toast.makeText(this@MainActivity,    "Error:
${e.message}", Toast.LENGTH_SHORT).show()
```

```
                }
            }
        }
    }


    private fun parseAndSave(jsonStr: String)
        { try {
            val arr = JSONArray(jsonStr)
            val db = dbHelper
            for (i in 0 until arr.length())
                { val obj =
                arr.getJSONObject(i) val
                person = Person() person.id =
                obj.getString("id")
                person.emailId = obj.optString("email", "")
                person.phoneNo = obj.optString("phone", "")
                if (obj.has("profile")) {
                    val prof = obj.getJSONObject("profile")
                    person.name = prof.optString("name", "")
                    person.address = prof.optString("address", "")
                }
                db.insertPerson(person)
            }
        } catch (e: Exception) {
            Log.e(tag, "parseAndSave error: ${e.message}")
        }
    }
}
```

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"
    android:fitsSystemWindows="true"
    tools:context=".MainActivity">

    <TextView
```

```
        android:id="@+id/title"
        android:text="SQLite and JSON Practical"
        android:textSize="24sp"
        android:textStyle="bold"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        android:layout_margin="16dp" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginHorizontal="16dp"
        android:layout_marginTop="12dp"
        app:layout_constraintTop_toBottomOf="@id/title"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintBottom_toBottomOf="parent" />


<com.google.android.material.floatingactionbutton.FloatingActionBu
tton
        android:id="@+id/fabRefresh"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:contentDescription="Refresh"
        android:src="@android:drawable/ic_popup_sync"
        android:tint="@android:color/white"
        android:layout_margin="16dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

item_person.xml

```
<?xml version="1.0" encoding="utf-8"?>
<com.google.android.material.card.MaterialCardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="8dp"
    app:cardCornerRadius="8dp"
    app:cardElevation="4dp">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="12dp">

        <ImageView
            android:id="@+id/avatar"
            android:layout_width="56dp"
            android:layout_height="wrap_content"
            android:src="@android:drawable/sym_def_app_icon"
            android:scaleType="centerCrop"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintBottom_toBottomOf="parent" />

        <TextView
            android:id="@+id/nameText"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Name"
            android:textSize="18sp"
            android:textStyle="bold"
            android:layout_marginStart="10dp"
            app:layout_constraintStart_toEndOf="@id/avatar"
            app:layout_constraintTop_toTopOf="@id/avatar"
            app:layout_constraintEnd_toStartOf="@id/deleteBtn" />

        <TextView
            android:id="@+id/phoneText"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Phone"
            app:layout_constraintStart_toStartOf="@id/nameText"
            app:layout_constraintTop_toBottomOf="@id/nameText"
            app:layout_constraintEnd_toStartOf="@id/deleteBtn" />
        <TextView
            android:id="@+id/emailText"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Email"
            app:layout_constraintStart_toStartOf="@id/nameText"
            app:layout_constraintTop_toBottomOf="@id/phoneText"
            app:layout_constraintEnd_toStartOf="@id/deleteBtn" />

        <TextView
            android:id="@+id/addressText"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Address"
            app:layout_constraintStart_toStartOf="@id/nameText"
            app:layout_constraintTop_toBottomOf="@id/emailText"
            app:layout_constraintEnd_toStartOf="@id/deleteBtn" />

        <ImageButton
            android:id="@+id/deleteBtn"
            android:layout_width="36dp"
            android:layout_height="36dp"
            android:background="#fd006a"
            android:src="@drawable/baseline_delete_24"
            app:tint="@android:color/white"
            android:contentDescription="Delete"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintBottom_toBottomOf="parent" />

    </androidx.constraintlayout.widget.ConstraintLayout>

</com.google.android.material.card.MaterialCardView>
```

Person.kt

```
package com.example.mad_23012011026_practical7

import java.io.Serializable

data class Person(
    var id: String = "",
    var name: String = "",
    var emailId: String = "",
    var phoneNo: String = "",
    var address: String = ""
) : Serializable
```

PersonAdapter.kt

```
package com.example.mad_23012011026_practical7

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

class PersonAdapter(private val items: MutableList<Person>) :
    RecyclerView.Adapter<PersonAdapter.PersonViewHolder>() {

    class          PersonViewHolder(view:          View)          :
RecyclerView.ViewHolder(view) {
        val nameText: TextView = view.findViewById(R.id.nameText)
        val           phoneText:           TextView           =
view.findViewById(R.id.phoneText)
        val           emailText:           TextView           =
view.findViewById(R.id.emailText)
        val           addressText:           TextView           =
view.findViewById(R.id.addressText)
        val           deleteBtn:           ImageButton           =
view.findViewById(R.id.deleteBtn)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType:
Int): PersonViewHolder {
        val                            v                          =
LayoutInflater.from(parent.context).inflate(R.layout.item_person,
parent, false)
        return PersonViewHolder(v)
    }

    override   fun   onBindViewHolder(holder:   PersonViewHolder,
position: Int) {
        val p = items[position]
        holder.nameText.text = p.name
        holder.phoneText.text = p.phoneNo
        holder.emailText.text = p.emailId
        holder.addressText.text = p.address

        holder.deleteBtn.setOnClickListener { v ->
            val ctx = v.context
            val db = DatabaseHelper(ctx)
            db.deletePerson(p)
            items.removeAt(position)
            notifyItemRemoved(position)
            notifyItemRangeChanged(position, items.size)
        }


    }

    override fun getItemCount(): Int = items.size

    fun updateItems(newList: List<Person>) {
        items.clear()
        items.addAll(newList)
        notifyDataSetChanged()
    }
}
```

PersonDbTableData.kt

```
package com.example.mad_23012011026_practical7

object PersonDbTableData {

    const val TABLE_PERSONS = "persons"
    const val KEY_ID = "id"
    const val KEY_NAME = "name"
    const val KEY_EMAIL = "email"
    const val KEY_PHONE = "phone"
    const val KEY_ADDRESS = "address"

    val CREATE_TABLE = ("CREATE TABLE $TABLE_PERSONS("
            + "$KEY_ID TEXT PRIMARY KEY,"
            + "$KEY_NAME TEXT,"
            + "$KEY_EMAIL TEXT,"
            + "$KEY_PHONE TEXT,"
            + "$KEY_ADDRESS TEXT)"
            )
}
```

HttpRequest.kt

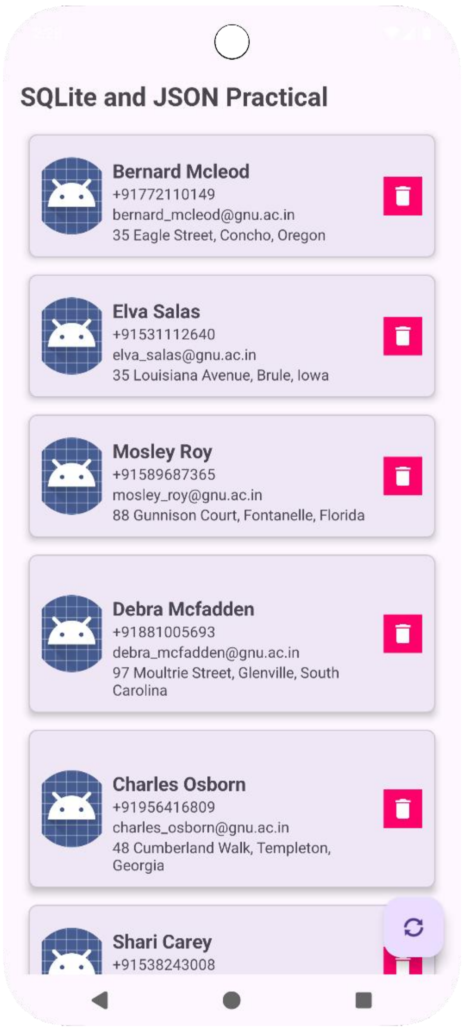
```
package com.example.mad_23012011026_practical7

import android.util.Log
import java.io.BufferedInputStream
import java.io.BufferedReader
import java.io.InputStreamReader
import java.net.HttpURLConnection
import java.net.MalformedURLException
import java.net.ProtocolException
import java.net.URL

object HttpRequest {
    private const val TAG = "HttpRequest"

    fun makeServiceCall(reqUrl: String?, token: String? = null): 
String? {
        var response: String? = null
        try {
            val url = URL(reqUrl)
            val conn = url.openConnection() as HttpURLConnection
            if (token != null) {
                conn.setRequestProperty("Authorization",    "Bearer 
$token")
                conn.setRequestProperty("Content-Type", 
"application/json")
            }
            conn.requestMethod = "GET"
            conn.connectTimeout = 15000
            conn.readTimeout = 15000

            val input = BufferedInputStream(conn.inputStream)
            response = convertStreamToString(input)
            input.close()
            conn.disconnect()
        } catch (e: MalformedURLException) {
            Log.e(TAG, "MalformedURLException: " + e.message)
        } catch (e: ProtocolException) {
            Log.e(TAG, "ProtocolException: " + e.message)
        } catch (e: java.io.IOException)
            { Log.e(TAG, "IOException: " +
            e.message)
        } catch (e: Exception) {
            Log.e(TAG, "Exception: " + e.message)
        }
        return response
    }

    private fun convertStreamToString(input: java.io.InputStream):
String {
        val reader = BufferedReader(InputStreamReader(input))
        val sb = StringBuilder()
        var line: String?
        while (true) {
            line = reader.readLine() ?: break
            sb.append(line)
        }
        return sb.toString()
    }
}
```

DatabaseHelper.kt

```
package com.example.mad_23012011026_practical7

import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper
import java.util.ArrayList

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

    companion object {
        private const val DATABASE_VERSION = 1
        private const val DATABASE_NAME = "persons_db"
    }
    override fun onCreate(db: SQLiteDatabase) {
        db.execSQL(PersonDbTableData.CREATE_TABLE)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int,
newVersion: Int) {
        db.execSQL("DROP      TABLE      IF      EXISTS      "      +
PersonDbTableData.TABLE_PERSONS)
        onCreate(db)
    }

    fun insertPerson(person: Person): Long
        { val db = writableDatabase
        val values = ContentValues()
        values.put(PersonDbTableData.KEY_ID, person.id)
        values.put(PersonDbTableData.KEY_NAME, person.name)
        values.put(PersonDbTableData.KEY_EMAIL, person.emailId)
        values.put(PersonDbTableData.KEY_PHONE, person.phoneNo)
        values.put(PersonDbTableData.KEY_ADDRESS, person.address)
        val                          id                          =
db.insertWithOnConflict(PersonDbTableData.TABLE_PERSONS,      null,
values, SQLiteDatabase.CONFLICT_REPLACE)
        db.close()
        return id
    }

    private fun getPersonFromCursor(cursor: Cursor): Person
        { val person = Person()
        person.id                                                  =
cursor.getString(cursor.getColumnIndexOrThrow(PersonDbTableData.KE
Y_ID))
        person.name                                                =
cursor.getString(cursor.getColumnIndexOrThrow(PersonDbTableData.KE
Y_NAME))
        person.emailId                                             =
cursor.getString(cursor.getColumnIndexOrThrow(PersonDbTableData.KE
Y_EMAIL))
        person.phoneNo                                             =
cursor.getString(cursor.getColumnIndexOrThrow(PersonDbTableData.KE
Y_PHONE))
```

```
        person.address                                                 =
cursor.getString(cursor.getColumnIndexOrThrow(PersonDbTableData.KE
Y_ADDRESS))
        return person
    }

    val allPersons: ArrayList<Person>
        get() {
            val list = ArrayList<Person>()
            val    selectQuery    =    "SELECT    *    FROM    "    +
PersonDbTableData.TABLE_PERSONS
            val db = readableDatabase
            val c = db.rawQuery(selectQuery, null)
            if (c.moveToFirst()) {
                do {
                    val p = getPersonFromCursor(c)
                    list.add(p)
                } while (c.moveToNext())
            }
            c.close()
            db.close()
            return list
        }

    fun deletePerson(person: Person): Int
        { val db = writableDatabase
        val rows = db.delete(PersonDbTableData.TABLE_PERSONS,
            "${PersonDbTableData.KEY_ID}=?",
            arrayOf(person.id))
        db.close()
        return rows
    }
}
```

Output



SQLite and JSON Practical

Bernard Mcleod
+91772110149
bernard_mcleod@gnu.ac.in
35 Eagle Street, Concho, Oregon

Elva Salas
+91531112640
elva_salas@gnu.ac.in
35 Louisiana Avenue, Brule, Iowa

Mosley Roy
+91589687365
mosley_roy@gnu.ac.in
88 Gunnison Court, Fontanelle, Florida

Debra Mcfadden
+91881005693
debra_mcfadden@gnu.ac.in
97 Moultrie Street, Glenville, South Carolina

Charles Osborn
+91956416809
charles_osborn@gnu.ac.in
48 Cumberland Walk, Templeton, Georgia

Shari Carey
+91538243008