MAD 2019 – Paper A – Answers

All answers from chatGPT & codes converted to kotlin

Android Kotlin SQLite Database | https://youtu.be/9LYn-OBO5qE
Insert, Select, Update, Delete and Display Data in RecyclerView

Question 01 - Activity Designs

a) Create an Activity called Main and design the given layout.

```
//layout
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".MainActivity">

    <!-- Username field -->
    <EditText
        android:id="@+id/username_edittext"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Username"
        android:inputType="text" />

    <!-- Password field -->
    <EditText
        android:id="@+id/password_edittext"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/username_edittext"
        android:hint="Password"
        android:inputType="textPassword" />

    <!-- Login button -->
    <Button
        android:id="@+id/login_button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/password_edittext"
        android:text="Login" />

    <!-- Registration button -->
    <Button
        android:id="@+id/registration_button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/login_button"
        android:text="Registration" />

</RelativeLayout>
```

```
//main.kt

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

b)) Create an Activity called AddMovie and design the given layout. Insert the provided image in the desktop location as a common .irnage for all movies.

```
//addmovie.kt

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle

class AddMovieActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_add_movie)
    }
}

//layout

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".AddMovieActivity">

    <!-- Movie name field -->
    <EditText
        android:id="@+id/movie_name_edittext"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Movie Name"
        android:inputType="text" />

    <!-- Movie year field -->
    <EditText
        android:id="@+id/movie_year_edittext"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/movie_name_edittext"
        android:hint="Movie Year"
        android:inputType="number" />

    <!-- Image view -->
    <ImageView
        android:id="@+id/movie_imageview"
        android:layout_width="match_parent"
        android:layout_height="200dp"
        android:layout_below="@id/movie_year_edittext"
        android:src="@drawable/movie_image" />

</RelativeLayout>
```

c) Create an Activity called MovieList and design the given layout

```
// MovieListActivity

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
```

```
class MovieListActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_movie_list)
    }
}
```

```
//activity_movie_list.xml

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".MovieListActivity">

    <!-- RecyclerView -->
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/movie_list_recyclerview"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</RelativeLayout>
```

```
// item_movie.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <!-- itemname TextView -->
    <TextView
        android:id="@+id/item_name_textview"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="18sp" />

    <!-- subitemdesc TextView -->
    <TextView
        android:id="@+id/subitem_desc_textview"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="14sp" />

</LinearLayout>
```

```
//RecyclerView.Adapter

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

class MovieAdapter(private val context: Context, private val movieList: List<Movie>) :
    RecyclerView.Adapter<MovieAdapter.ViewHolder>() {

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

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        val movie = movieList[position]
        holder.itemNameTextView.text = movie.itemName
        holder.subItemDescTextView.text = movie.subItemDesc
    }

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

    inner class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
        val itemNameTextView: TextView = itemView.findViewById(R.id.item_name_textview)
        val subItemDescTextView: TextView = itemView.findViewById(R.id.subitem_desc_textview)
    }
}
```

//MovieAdapter:

```
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView

class MovieListActivity : AppCompatActivity() {

    private lateinit var movieRecyclerView: RecyclerView
    private lateinit var movieAdapter: MovieAdapter
    private lateinit var movieList: List<Movie>

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_movie_list)

        movieRecyclerView = findViewById(R.id.movie_list_recyclerview)
        movieList = generateMovieList() // Generate some dummy movie data

        movieAdapter = MovieAdapter(this, movieList)
        movieRecyclerView.layoutManager = LinearLayoutManager(this)
        movieRecyclerView.adapter = movieAdapter
    }

    private fun generateMovieList(): List<Movie> {
        val movieList = ArrayList<Movie>()
        // Add dummy movie data
        movieList.add(Movie("Movie 1", "Description 1"))
        movieList.add(Movie("Movie 2", "Description 2"))
        movieList.add(Movie("Movie 3", "Description 3"))
        // Add more movies as needed
        return movieList
    }
}
```

d) Create an Activity called MovieOverview and design the given layout.

//overview

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.RatingBar
import android.widget.Toast

class MovieOverviewActivity : AppCompatActivity() {

    private lateinit var ratingBar: RatingBar
    private lateinit var commentEditText: EditText
    private lateinit var submitButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_movie_overview)

        ratingBar = findViewById(R.id.rating_bar)
        commentEditText = findViewById(R.id.comment_edit_text)
        submitButton = findViewById(R.id.submit_button)

        submitButton.setOnClickListener {
            val rating = ratingBar.rating
            val comment = commentEditText.text.toString()

            // Perform any necessary actions with the rating and comment

            Toast.makeText(this, "Rating: $rating, Comment: $comment", Toast.LENGTH_SHORT).show()
        }
    }
}
```

activity_movie_overview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp">

    <TextView
        android:id="@+id/rating_label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Rating:"
        android:textSize="18sp"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true" />

    <RatingBar
        android:id="@+id/rating_bar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:numStars="5"
        android:stepSize="1"
        android:rating="0"
        android:layout_toEndOf="@+id/rating_label"
        android:layout_marginStart="8dp"
        android:layout_alignBaseline="@+id/rating_label" />

    <TextView
        android:id="@+id/comment_label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Comment:"
        android:textSize="18sp"
        android:layout_below="@+id/rating_label"
        android:layout_alignParentStart="true"
        android:layout_marginTop="16dp" />

    <EditText
        android:id="@+id/comment_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textMultiLine"
        android:lines="4"
        android:maxLines="6"
        android:gravity="top"
        android:layout_below="@+id/comment_label"
        android:layout_marginTop="8dp" />

    <Button
        android:id="@+id/submit_button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Submit"
        android:textSize="18sp"
        android:layout_below="@+id/comment_edit_text"
        android:layout_marginTop="16dp" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/item_list_recyclerview"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/submit_button"
        android:layout_marginTop="16dp" />

</RelativeLayout>
```

e) Use string.xml file to include all string values / messages / labels.

```
<resources>

<string name="label_rating">Rating:</string>
<string name="label_comment">Comment:</string>
<string name="button_submit">Submit</string>
<string name="label_username">Username</string>
<string name="label_password">Password</string>

</resources>
```

Question 02 - Create the Database

a) **Create a class named DatabaseMaster to represent the tables and define column names as constant attributes of the class. Implement the class default constructor**

DatabaseMaster.kt

```
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

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

    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Define your table and column names
        private const val TABLE_NAME = "mytable"
        private const val COLUMN_ID = "_id"
        private const val COLUMN_NAME = "name"
    }

    override fun onCreate(db: SQLiteDatabase) {
        val createTableQuery = "CREATE TABLE $TABLE_NAME ($COLUMN_ID INTEGER PRIMARY KEY,
$COLUMN_NAME TEXT)"
        db.execSQL(createTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
```

```
            }
        }
```

b) Define an inner class called Users to represent users table and define f llowing columns. (3 marks) Table name - String Usemame - String Password - String UserType - String

```
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

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

    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Table and column names for the Users table
        private const val TABLE_USERS = "users"
        private const val COLUMN_USERNAME = "username"
        private const val COLUMN_PASSWORD = "password"
        private const val COLUMN_USERTYPE = "usertype"
    }

    override fun onCreate(db: SQLiteDatabase) {
        // Create the Users table
        val createUsersTableQuery = "CREATE TABLE $TABLE_USERS ($COLUMN_USERNAME TEXT, $COLUMN_PASSWORD
TEXT, $COLUMN_USERTYPE TEXT)"
        db.execSQL(createUsersTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }

    inner class Users {
        // Define the columns for the Users table as constants
        companion object {
            const val COLUMN_USERNAME = "username"
            const val COLUMN_PASSWORD = "password"
            const val COLUMN_USERTYPE = "usertype"
        }
    }
}
```

c) Define an inner class called Movie to represent movies table and define following columns. (2 marks) Table name - String Movie name - String Movie Year - Integer

```
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

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

    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Table and column names for the Users table
        private const val TABLE_USERS = "users"
        private const val COLUMN_USERNAME = "username"
        private const val COLUMN_PASSWORD = "password"
        private const val COLUMN_USERTYPE = "usertype"

        // Table and column names for the Movies table
        private const val TABLE_MOVIES = "movies"
        private const val COLUMN_MOVIE_NAME = "movie_name"
        private const val COLUMN_MOVIE_YEAR = "movie_year"
    }

    override fun onCreate(db: SQLiteDatabase) {
        // Create the Users table
        val createUsersTableQuery = "CREATE TABLE $TABLE_USERS ($COLUMN_USERNAME TEXT, $COLUMN_PASSWORD
TEXT, $COLUMN_USERTYPE TEXT)"
        db.execSQL(createUsersTableQuery)

        // Create the Movies table
        val createMoviesTableQuery = "CREATE TABLE $TABLE_MOVIES ($COLUMN_MOVIE_NAME TEXT,
$COLUMN_MOVIE_YEAR INTEGER)"
        db.execSQL(createMoviesTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }

    inner class Users {
        // Define the columns for the Users table as constants
        companion object {
          const val COLUMN_USERNAME = "username"
          const val COLUMN_PASSWORD = "password"
          const val COLUMN_USERTYPE = "usertype"
        }
    }

    inner class Movie {
        // Define the columns for the Movies table as constants
        companion object {
          const val COLUMN_MOVIE_NAME = "movie_name"
          const val COLUMN_MOVIE_YEAR = "movie_year"
        }
    }
}
```

d) Define an inner class called Comments to represent comments tabl and define following columns. (3 marks) Table name - String Movie name - String Movie rating - Integer Movie comments - String

```
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

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

  companion object {
    private const val DATABASE_NAME = "mydatabase.db"
    private const val DATABASE_VERSION = 1

    // Table and column names for the Users table
    private const val TABLE_USERS = "users"
    private const val COLUMN_USERNAME = "username"
    private const val COLUMN_PASSWORD = "password"
    private const val COLUMN_USERTYPE = "usertype"

    // Table and column names for the Movies table
    private const val TABLE_MOVIES = "movies"
    private const val COLUMN_MOVIE_NAME = "movie_name"
    private const val COLUMN_MOVIE_YEAR = "movie_year"
```

```
        // Table and column names for the Comments table
        private const val TABLE_COMMENTS = "comments"
        private const val COLUMN_COMMENT_MOVIE_NAME = "movie_name"
        private const val COLUMN_COMMENT_MOVIE_RATING = "movie_rating"
        private const val COLUMN_COMMENT_MOVIE_COMMENT = "movie_comment"
    }

    override fun onCreate(db: SQLiteDatabase) {
        // Create the Users table
        val createUsersTableQuery = "CREATE TABLE $TABLE_USERS ($COLUMN_USERNAME TEXT, $COLUMN_PASSWORD TEXT, $COLUMN_USERTYPE TEXT)"
        db.execSQL(createUsersTableQuery)

        // Create the Movies table
        val createMoviesTableQuery = "CREATE TABLE $TABLE_MOVIES ($COLUMN_MOVIE_NAME TEXT, $COLUMN_MOVIE_YEAR INTEGER)"
        db.execSQL(createMoviesTableQuery)

        // Create the Comments table
        val createCommentsTableQuery = "CREATE TABLE $TABLE_COMMENTS ($COLUMN_COMMENT_MOVIE_NAME TEXT, $COLUMN_COMMENT_MOVIE_RATING INTEGER, $COLUMN_COMMENT_MOVIE_COMMENT TEXT)"
        db.execSQL(createCommentsTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }

    inner class Users {
        // Define the columns for the Users table as constants
        companion object {
            const val COLUMN_USERNAME = "username"
            const val COLUMN_PASSWORD = "password"
            const val COLUMN_USERTYPE = "usertype"
        }
    }

    inner class Movie {
        // Define the columns for the Movies table as constants
        companion object {
            const val COLUMN_MOVIE_NAME = "movie_name"
            const val COLUMN_MOVIE_YEAR = "movie_year"
        }
    }

    inner class Comments {
        // Define the columns for the Comments table as constants
        companion object {
            const val COLUMN_COMMENT_MOVIE_NAME = "movie_name"
            const val COLUMN_COMMENT_MOVIE_RATING = "movie_rating"
            const val COLUMN_COMMENT_MOVIE_COMMENT = "movie_comment"
        }
    }
}
```

Question 03 - DBHandler Class

a) Create a java class called DBHandler and override the onCrea te ()to create the above SQLite table

```
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1
        // Table and column names for the Users table
        private const val TABLE_USERS = "users"
        private const val COLUMN_USERNAME = "username"
        private const val COLUMN_PASSWORD = "password"
        private const val COLUMN_USERTYPE = "usertype"

        // Table and column names for the Movies table
        private const val TABLE_MOVIES = "movies"
        private const val COLUMN_MOVIE_NAME = "movie_name"
        private const val COLUMN_MOVIE_YEAR = "movie_year"

        // Table and column names for the Comments table
        private const val TABLE_COMMENTS = "comments"
        private const val COLUMN_COMMENT_MOVIE_NAME = "movie_name"
        private const val COLUMN_COMMENT_MOVIE_RATING = "movie_rating"
        private const val COLUMN_COMMENT_MOVIE_COMMENT = "movie_comment"
    }

    override fun onCreate(db: SQLiteDatabase) {
        // Create the Users table
        val createUsersTableQuery = "CREATE TABLE $TABLE_USERS ($COLUMN_USERNAME TEXT, $COLUMN_PASSWORD
TEXT, $COLUMN_USERTYPE TEXT)"
        db.execSQL(createUsersTableQuery)

        // Create the Movies table
        val createMoviesTableQuery = "CREATE TABLE $TABLE_MOVIES ($COLUMN_MOVIE_NAME TEXT,
$COLUMN_MOVIE_YEAR INTEGER)"
        db.execSQL(createMoviesTableQuery)

        // Create the Comments table
        val createCommentsTableQuery = "CREATE TABLE $TABLE_COMMENTS ($COLUMN_COMMENT_MOVIE_NAME TEXT,
$COLUMN_COMMENT_MOVIE_RATING INTEGER, $COLUMN_COMMENT_MOVIE_COMMENT TEXT)"
        db.execSQL(createCommentsTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }
}
```

b) Implement a method called registerUser () to add a new user to the users table. Assume that the user name and the password to be used as parameters

```
import android.content.ContentValues
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
  // ...

  fun registerUser(username: String, password: String, userType: String): Long {
    val db = writableDatabase
    val contentValues = ContentValues()
    contentValues.put(COLUMN_USERNAME, username)
    contentValues.put(COLUMN_PASSWORD, password)
    contentValues.put(COLUMN_USERTYPE, userType)
    val result = db.insert(TABLE_USERS, null, contentValues)
    db.close()
    return result
  }

  // ...
}
```

c) Implement a method called lOc1fhUser () to login to the system and do the following. (12 marks)

(i) If the user gives user name as "admin", it will be an admin ogin.
(ii) Then this user will able to navigate to the AddMovie screei .
(iii) If the user gives another user name and password which ha e already registered to the system, the user will able to navigate to the MovieListcreen.
(iv) If the entered user name and password are not matching or ot existing in the users table, it should be displayed as "Login failed" message including the particular reason. 4

```
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
  companion object {
    private const val DATABASE_NAME = "mydatabase.db"
    private const val DATABASE_VERSION = 1

    // Table and column names for the Users table
    private const val TABLE_USERS = "users"
    private const val COLUMN_USERNAME = "username"
    private const val COLUMN_PASSWORD = "password"
    private const val COLUMN_USERTYPE = "usertype"
  }

  override fun onCreate(db: SQLiteDatabase) {
    // Create the Users table
    val createUsersTableQuery = "CREATE TABLE $TABLE_USERS ($COLUMN_USERNAME TEXT, $COLUMN_PASSWORD
TEXT, $COLUMN_USERTYPE TEXT)"
    db.execSQL(createUsersTableQuery)
  }

  override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
    // Perform necessary database upgrades or migrations
  }

  fun registerUser(username: String, password: String, userType: String) {
    val db = writableDatabase
    val values = ContentValues().apply {
      put(COLUMN_USERNAME, username)
      put(COLUMN_PASSWORD, password)
      put(COLUMN_USERTYPE, userType)
    }
    db.insert(TABLE_USERS, null, values)
    db.close()
  }

  fun loginUser(username: String, password: String): String {
    val db = readableDatabase
    val selection = "$COLUMN_USERNAME = ? AND $COLUMN_PASSWORD = ?"
    val selectionArgs = arrayOf(username, password)
    val cursor: Cursor = db.query(
      TABLE_USERS,
      null,
      selection,
      selectionArgs,
      null,
      null,
      null
    )

    val userType: String = if (username == "admin") {
      "admin"
    } else if (cursor.moveToFirst()) {
      "user"
    } else {
      "invalid"
```

```
        }

        cursor.close()
        db.close()
        return userType
    }
}
```

**d) Create three lists for movies, years and comments, and generate get methods for all of them.
(3 marks)**

```
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Table and column names for the Users table
        private const val TABLE_USERS = "users"
        private const val COLUMN_USERNAME = "username"
        private const val COLUMN_PASSWORD = "password"
        private const val COLUMN_USERTYPE = "usertype"
    }

    private val movies: MutableList<String> = mutableListOf()
    private val years: MutableList<Int> = mutableListOf()
    private val comments: MutableList<String> = mutableListOf()

    override fun onCreate(db: SQLiteDatabase) {
        // Create the Users table
        val createUsersTableQuery = "CREATE TABLE $TABLE_USERS ($COLUMN_USERNAME TEXT, $COLUMN_PASSWORD
TEXT, $COLUMN_USERTYPE TEXT)"
        db.execSQL(createUsersTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }

    fun registerUser(username: String, password: String, userType: String) {
        val db = writableDatabase
        val values = ContentValues().apply {
            put(COLUMN_USERNAME, username)
            put(COLUMN_PASSWORD, password)
            put(COLUMN_USERTYPE, userType)
        }
        db.insert(TABLE_USERS, null, values)
        db.close()
    }

    fun loginUser(username: String, password: String): String {
        val db = readableDatabase
        val selection = "$COLUMN_USERNAME = ? AND $COLUMN_PASSWORD = ?"
        val selectionArgs = arrayOf(username, password)
        val cursor: Cursor = db.query(
            TABLE_USERS,
            null,
            selection,
            selectionArgs,
            null,
            null,
            null
        )

        val userType: String = if (username == "admin") {
            "admin"
        } else if (cursor.moveToFirst()) {
            "user"
        } else {
            "invalid"
        }

        cursor.close()
        db.close()
        return userType
    }

    fun getMovies(): List<String> {
        return movies
    }

    fun getYears(): List<Int> {
        return years
    }

    fun getComments(): List<String> {
        return comments
    }
}
```

e) Implement a method called addMovie () to store movie details. Thi method must return a Boolean value based on the insert success or failure.

```
import android.content.ContentValues
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Table and column names for the Users table
        private const val TABLE_USERS = "users"
        private const val COLUMN_USERNAME = "username"
        private const val COLUMN_PASSWORD = "password"
        private const val COLUMN_USERTYPE = "usertype"

        // Table and column names for the Movies table
        private const val TABLE_MOVIES = "movies"
        private const val COLUMN_MOVIE_NAME = "movie_name"
        private const val COLUMN_MOVIE_YEAR = "movie_year"
    }

    // ...

    fun addMovie(movieName: String, movieYear: Int): Boolean {
        val db = writableDatabase
        val values = ContentValues().apply {
            put(COLUMN_MOVIE_NAME, movieName)
            put(COLUMN_MOVIE_YEAR, movieYear)
        }

        val result = db.insert(TABLE_MOVIES, null, values)
        db.close()

        return result != -1L // Returns true if insertion was successful, false otherwise
    }
    // ...
}
```

f) Implement a method called viewMoviesO to retrieve all movies' na res stored in the movies table.

```
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Table and column names for the Users table
        private const val TABLE_USERS = "users"
        private const val COLUMN_USERNAME = "username"
        private const val COLUMN_PASSWORD = "password"
        private const val COLUMN_USERTYPE = "usertype"

        // Table and column names for the Movies table
        private const val TABLE_MOVIES = "movies"
        private const val COLUMN_MOVIE_NAME = "movie_name"
        private const val COLUMN_MOVIE_YEAR = "movie_year"
        private const val COLUMN_MOVIE_COMMENT = "movie_comment"
    }

    override fun onCreate(db: SQLiteDatabase) {
        val createUsersTableQuery = "CREATE TABLE $TABLE_USERS ($COLUMN_USERNAME TEXT, $COLUMN_PASSWORD
TEXT, $COLUMN_USERTYPE TEXT)"
        db.execSQL(createUsersTableQuery)

        val createMoviesTableQuery = "CREATE TABLE $TABLE_MOVIES ($COLUMN_MOVIE_NAME TEXT,
$COLUMN_MOVIE_YEAR INTEGER, $COLUMN_MOVIE_COMMENT TEXT)"
        db.execSQL(createMoviesTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }

    fun registerUser(username: String, password: String, userType: String): Boolean {
        val db = writableDatabase
        val values = ContentValues().apply {
            put(COLUMN_USERNAME, username)
            put(COLUMN_PASSWORD, password)
            put(COLUMN_USERTYPE, userType)
        }
        val result = db.insert(TABLE_USERS, null, values)
        db.close()
        return result != -1L
    }

    fun loginUser(username: String, password: String): String {
        val db = readableDatabase
        val selection = "$COLUMN_USERNAME = ? AND $COLUMN_PASSWORD = ?"
        val selectionArgs = arrayOf(username, password)
        val cursor: Cursor = db.query(
            TABLE_USERS,
            null,
            selection,
            selectionArgs,
            null,
            null,
            null
        )

        val userType: String = if (username == "admin") {
            "admin"
        } else if (cursor.moveToFirst()) {
            "user"
        } else {
            "invalid"
        }

        cursor.close()
        db.close()
        return userType
    }

    fun addMovie(movieName: String, movieYear: Int, movieComment: String): Boolean {
        val db = writableDatabase
        val values = ContentValues().apply {
            put(COLUMN_MOVIE_NAME, movieName)
            put(COLUMN_MOVIE_YEAR, movieYear)
            put(COLUMN_MOVIE_COMMENT, movieComment)
        }
        val result = db.insert(TABLE_MOVIES, null, values)
        db.close()
        return result != -1L
    }

    fun viewMovies(): List<String> {
        val movieList = mutableListOf<String>()
        val db = readableDatabase
        val query = "SELECT $COLUMN_MOVIE_NAME FROM $TABLE_MOVIES"
        val cursor: Cursor = db.rawQuery(query, null)

        while (cursor.moveToNext()) {
            val movieName = cursor.getString(cursor.getColumnIndex(COLUMN_MOVIE_NAME))
            movieList.add(movieName)
        }

        cursor.close()
        db.close()

        return movieList
    }
}
```

g) Implement a method called insertComrnentsO to add comments' details to the comments table.

```
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Table and column names for the Movies table
        private const val TABLE_MOVIES = "movies"
        private const val COLUMN_MOVIE_NAME = "movie_name"
        private const val COLUMN_MOVIE_YEAR = "movie_year"

        // Table and column names for the Comments table
        private const val TABLE_COMMENTS = "comments"
        private const val COLUMN_COMMENT_MOVIE_NAME = "movie_name"
        private const val COLUMN_COMMENT_RATING = "movie_rating"
        private const val COLUMN_COMMENT_COMMENT = "movie_comment"
    }

    override fun onCreate(db: SQLiteDatabase) {
        val createMoviesTableQuery = "CREATE TABLE $TABLE_MOVIES ($COLUMN_MOVIE_NAME TEXT,
$COLUMN_MOVIE_YEAR INTEGER)"
        db.execSQL(createMoviesTableQuery)

        val createCommentsTableQuery = "CREATE TABLE $TABLE_COMMENTS ($COLUMN_COMMENT_MOVIE_NAME TEXT,
$COLUMN_COMMENT_RATING INTEGER, $COLUMN_COMMENT_COMMENT TEXT)"
        db.execSQL(createCommentsTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }

    fun insertComments(movieName: String, movieRating: Int, movieComment: String): Boolean {
        val db = writableDatabase
        val values = ContentValues().apply {
            put(COLUMN_COMMENT_MOVIE_NAME, movieName)
            put(COLUMN_COMMENT_RATING, movieRating)
            put(COLUMN_COMMENT_COMMENT, movieComment)
        }
        val result = db.insert(TABLE_COMMENTS, null, values)
        db.close()
        return result != -1L
    }

    fun viewMovies(): List<String> {
        val movieList = mutableListOf<String>()
        val db = readableDatabase
        val query = "SELECT $COLUMN_MOVIE_NAME FROM $TABLE_MOVIES"
        val cursor: Cursor = db.rawQuery(query, null)

        while (cursor.moveToNext()) {
            val movieName = cursor.getString(cursor.getColumnIndex(COLUMN_MOVIE_NAME))
            movieList.add(movieName)
        }

        cursor.close()
        db.close()

        return movieList
    }
}
```

h) Implement a method called viewComrnents ()to retrieve all c mments stored in the comments table.

```
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHandler(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
    companion object {
        private const val DATABASE_NAME = "mydatabase.db"
        private const val DATABASE_VERSION = 1

        // Table and column names for the Movies table
        private const val TABLE_MOVIES = "movies"
        private const val COLUMN_MOVIE_NAME = "movie_name"
        private const val COLUMN_MOVIE_YEAR = "movie_year"
        // Table and column names for the Comments table
        private const val TABLE_COMMENTS = "comments"
        private const val COLUMN_COMMENT_MOVIE_NAME = "movie_name"
        private const val COLUMN_COMMENT_RATING = "movie_rating"
        private const val COLUMN_COMMENT_COMMENT = "movie_comment"
    }

    override fun onCreate(db: SQLiteDatabase) {
        val createMoviesTableQuery = "CREATE TABLE $TABLE_MOVIES ($COLUMN_MOVIE_NAME TEXT,
$COLUMN_MOVIE_YEAR INTEGER)"
        db.execSQL(createMoviesTableQuery)

        val createCommentsTableQuery = "CREATE TABLE $TABLE_COMMENTS ($COLUMN_COMMENT_MOVIE_NAME TEXT,
$COLUMN_COMMENT_RATING INTEGER, $COLUMN_COMMENT_COMMENT TEXT)"
        db.execSQL(createCommentsTableQuery)
    }

    override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
        // Perform necessary database upgrades or migrations
    }

    fun insertComments(movieName: String, movieRating: Int, movieComment: String): Boolean {
        val db = writableDatabase
        val values = ContentValues().apply {
            put(COLUMN_COMMENT_MOVIE_NAME, movieName)
            put(COLUMN_COMMENT_RATING, movieRating)
            put(COLUMN_COMMENT_COMMENT, movieComment)
        }
        val result = db.insert(TABLE_COMMENTS, null, values)
        db.close()
        return result != -1L
    }

    fun viewMovies(): List<String> {
        val movieList = mutableListOf<String>()
        val db = readableDatabase
        val query = "SELECT $COLUMN_MOVIE_NAME FROM $TABLE_MOVIES"
        val cursor: Cursor = db.rawQuery(query, null)

        while (cursor.moveToNext()) {
            val movieName = cursor.getString(cursor.getColumnIndex(COLUMN_MOVIE_NAME))
            movieList.add(movieName)
        }

        cursor.close()
        db.close()

        return movieList
    }

    fun viewComments(): List<String> {
        val commentList = mutableListOf<String>()
        val db = readableDatabase
        val query = "SELECT * FROM $TABLE_COMMENTS"
        val cursor: Cursor = db.rawQuery(query, null)

        while (cursor.moveToNext()) {
            val movieName = cursor.getString(cursor.getColumnIndex(COLUMN_COMMENT_MOVIE_NAME))
            val movieRating = cursor.getInt(cursor.getColumnIndex(COLUMN_COMMENT_RATING))
            val movieComment = cursor.getString(cursor.getColumnIndex(COLUMN_COMMENT_COMMENT))

            val comment = "Movie: $movieName, Rating: $movieRating, Comment: $movieComment"
            commentList.add(comment)
        }

        cursor.close()
        db.close()

        return commentList
    }
}
```

Question 04 - Implementation of Activities

Implement the Main Activity as follows.
a) Call to registerUser () method implemented in DBHandler cl ss from the onClick event of the Register button. (2 marks)

```
import android.content.Intent
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        dbHandler = DBHandler(this)

        loginButton.setOnClickListener {
            val intent = Intent(this, LoginActivity::class.java)
            startActivity(intent)
        }

        registerButton.setOnClickListener {
            val username = usernameEditText.text.toString()
            val password = passwordEditText.text.toString()

            val isSuccess = dbHandler.registerUser(username, password)

            if (isSuccess) {
                // Registration successful
                // You can perform additional actions here, like displaying a success message
            } else {
                // Registration failed
                // You can display an error message to the user
            }
        }
    }
}
```

b) Call to loginUser () method implemented in DBHandler class frc m the onClick event of the Login button

```
import android.content.Intent
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        dbHandler = DBHandler(this)

        loginButton.setOnClickListener {
            val username = usernameEditText.text.toString()
            val password = passwordEditText.text.toString()

            val userType = dbHandler.loginUser(username, password)

            when (userType) {
                "admin" -> {
                    // Admin login successful
                    val intent = Intent(this, AddMovieActivity::class.java)
                    startActivity(intent)
                }
                "user" -> {
                    // User login successful
                    val intent = Intent(this, MovieListActivity::class.java)
                    startActivity(intent)
                }
                else -> {
                    // Login failed
                    // Display an error message to the user
                }
            }
        }

        registerButton.setOnClickListener {
            val username = usernameEditText.text.toString()
            val password = passwordEditText.text.toString()

            val isSuccess = dbHandler.registerUser(username, password)

            if (isSuccess) {
                // Registration successful
                // You can perform additional actions here, like displaying a success message
            } else {
                // Registration failed
                // You can display an error message to the user
            }
        }
    }
}
```

Implement the AddMovie Activity as follows.
c) Call to addMovie () method implemented in DBHandler class fr the onClick event of the Add button.

```
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_add_movie.*

class AddMovieActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_add_movie)

        dbHandler = DBHandler(this)

        addButton.setOnClickListener {
            val movieName = movieNameEditText.text.toString()
            val movieYear = movieYearEditText.text.toString().toInt()

            val isSuccess = dbHandler.addMovie(movieName, movieYear)

            if (isSuccess) {
                // Movie added successfully
                Toast.makeText(this, "Movie added successfully", Toast.LENGTH_SHORT).show()
                finish() // Close the activity
            } else {
                // Failed to add movie
                Toast.makeText(this, "Failed to add movie", Toast.LENGTH_SHORT).show()
            }
        }
    }
}
```

d) Display a message indicating success or failure of adding movie detai s

```
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_add_movie.*

class AddMovieActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_add_movie)

        dbHandler = DBHandler(this)

        addButton.setOnClickListener {
            val movieName = movieNameEditText.text.toString()
            val movieYear = movieYearEditText.text.toString().toInt()

            val isSuccess = dbHandler.addMovie(movieName, movieYear)

            if (isSuccess) {
                // Movie added successfully
                Toast.makeText(this, "Movie added successfully", Toast.LENGTH_SHORT).show()
                finish() // Close the activity
            } else {
                // Failed to add movie
                Toast.makeText(this, "Failed to add movie", Toast.LENGTH_SHORT).show()
            }
        }
    }
}
```

Implement the MovieList Activity as follows.
e) Override the onCreate () and load movie names into the moxie lis. (2 marks)

```
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import kotlinx.android.synthetic.main.activity_movie_list.*

class MovieListActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler
    private lateinit var movieAdapter: MovieAdapter

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_movie_list)

        dbHandler = DBHandler(this)

        // Load movie names into the movie list
        val movieNames = dbHandler.viewMovies()

        movieAdapter = MovieAdapter(movieNames)
        movieRecyclerView.layoutManager = LinearLayoutManager(this)
        movieRecyclerView.adapter = movieAdapter
    }
}
```

f) Implement list item click event for a selected item and pass the movie ame to MovieOverview class while displaying the particular screen. (3 marks)

```
import android.content.Intent
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import kotlinx.android.synthetic.main.activity_movie_list.*

class MovieListActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler
    private lateinit var movieAdapter: MovieAdapter

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_movie_list)

        dbHandler = DBHandler(this)

        val movieNames = dbHandler.viewMovies()

        movieAdapter = MovieAdapter(movieNames)
        movieRecyclerView.layoutManager = LinearLayoutManager(this)
        movieRecyclerView.adapter = movieAdapter

        movieAdapter.setOnItemClickListener { movieName ->
            val intent = Intent(this, MovieOverviewActivity::class.java)
            intent.putExtra("movieName", movieName)
            startActivity(intent)
        }
    }
}
```

Implement the MovieOverview Activity as follows to have an output like, below.
g) Override onCrea te () and implement the code to load available comments to the list view in the bottom of the screen and display the average rate as given in the ibove sample screen. (5 marks)

```
import android.os.Bundle
import android.widget.ArrayAdapter
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_movie_overview.*

class MovieOverviewActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler
    private lateinit var commentAdapter: ArrayAdapter<String>

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_movie_overview)

        dbHandler = DBHandler(this)

        val movieName = intent.getStringExtra("movieName")

        movieTitleTextView.text = movieName

        loadComments(movieName)

        submitButton.setOnClickListener {
            val rating = ratingBar.rating
            val comment = commentEditText.text.toString()
```

```
      val isSuccess = dbHandler.insertComment(movieName, rating, comment)

      if (isSuccess) {
        // Comment added successfully
        // You can perform additional actions here, like displaying a success message
        loadComments(movieName)
        ratingBar.rating = 0f
        commentEditText.text.clear()
      } else {
        // Comment failed to add
        // You can display an error message to the user
      }
    }
  }

  private fun loadComments(movieName: String) {
    val comments = dbHandler.viewComments(movieName)

    commentAdapter = ArrayAdapter(this, android.R.layout.simple_list_item_1, comments)
    commentsListView.adapter = commentAdapter

    val averageRating = dbHandler.calculateAverageRating(movieName)
    averageRatingTextView.text = String.format("%.1f", averageRating)
  }
}
```

h) Call to insertComrnent () method implemented in DBHandler cl: ss from the onClick event of the Submit button and send the rating and comment input by t ie user with the movie name

```
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_movie_overview.*

class MovieOverviewActivity : AppCompatActivity() {

  private lateinit var dbHandler: DBHandler

  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_movie_overview)

    dbHandler = DBHandler(this)

    val movieName = intent.getStringExtra("movieName")

    movieTitleTextView.text = movieName

    submitButton.setOnClickListener {
      val rating = ratingBar.rating
      val comment = commentEditText.text.toString()

      val isSuccess = dbHandler.insertComment(movieName, rating, comment)

      if (isSuccess) {
        // Comment added successfully
        // Display a success message
        Toast.makeText(this, "Comment added successfully", Toast.LENGTH_SHORT).show()
        ratingBar.rating = 0f
        commentEditText.text.clear()
      } else {
        // Comment failed to add
        // Display an error message
        Toast.makeText(this, "Failed to add comment", Toast.LENGTH_SHORT).show()
      }
    }
  }
}
```

i) Display a message indicating success or failure of adding review (corm lent) details. (l mark)

```
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_movie_overview.*

class MovieOverviewActivity : AppCompatActivity() {

    private lateinit var dbHandler: DBHandler

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_movie_overview)

        dbHandler = DBHandler(this)

        val movieName = intent.getStringExtra("movieName")

        movieTitleTextView.text = movieName

        submitButton.setOnClickListener {
            val rating = ratingBar.rating
            val comment = commentEditText.text.toString()

            val isSuccess = dbHandler.insertComment(movieName, rating, comment)

            if (isSuccess) {
                // Comment added successfully
                Toast.makeText(this, "Review added successfully", Toast.LENGTH_SHORT).show()
                ratingBar.rating = 0f
                commentEditText.text.clear()
            } else {
                // Failed to add comment
                Toast.makeText(this, "Failed to add review", Toast.LENGTH_SHORT).show()
            }
        }
    }
}
```