



Experiment 10

Student Name: Rohan Godha

UID: 20BCS1682

Branch: BE-CSE

Section/Group: 701/A

Semester: 6th

Subject Name: MAD Lab

Subject Code: 20CSP-356

AIM : Create an Android application for user registration that stores the user details in a database table.

Step by Step Implementaion

Step 1: Create a New Project

To create a new project in Android Studio please refer to [How to Create/Start a New Project in Android Studio](#). Note that select Java as the programming language.

Step 2: Adding permissions to access the storage in the AndroidManifest.xml file

Navigate to the app > AndroidManifest.xml and add the below code to it.

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Step 3: Working with the acTivity_main.xml file

Navigate to the app > res > layout > acTivity_main.xml and add the below code to that file.

Below is the code for the acTivity_main.xml file. `<?xml version="1.0" encoding="uZ-8"?>`

```
<LinearLayout 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:orientation="vertical"
    tools:context=".MainAcTivity">

    <!--Edit text to enter course name-->
    <EditText android:id="@+id/idEdtCourseName"
        android:layout_width="match_parent" android:layout_height="wrap_content"
        android:layout_margin="10dp" android:hint="Enter course Name" />

    <!--edit text to enter course duraTon-->
```



```
<EditText android:id="@+id/idEdtCourseDuraTon"
android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_margin="10dp" android:hint="Enter
Course DuraTon" />

<!--edit text to display course tracks-->
<EditText android:id="@+id/idEdtCourseTracks"
android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_margin="10dp" android:hint="Enter
Course Tracks" />

<!--edit text for course descripTon-->
<EditText android:id="@+id/idEdtCourseDescripTon"
android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_margin="10dp" android:hint="Enter Course DescripTon" />

<!--bu\on for adding new course-->
<Bu\on android:id="@+id/idBtnAddCourse"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="10dp" android:text="Add
Course" android:textAllCaps="false" />

</LinearLayout>
```

Step 4: CreaTng a new Java class for performing SQLite operaTons Navigate to the app > java > your app's package name > Right-click on it > New > Java class and name it as DBHandler and add the below code to it. Comments are added inside the code to understand the code in more detail.

```
//Java code import android.content.ContentValues; import android.content.Context;
import android.database.sqlite.SQLiteDatabase; import
android.database.sqlite.SQLiteOpenHelper; public class DBHandler extends
SQLiteOpenHelper {

// creaTng a constant variables for our database.
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
// below variable is for our database name.

private static final String DB_NAME = "coursedb";

// below int is our database version private static final int
DB_VERSION = 1;

// below variable is for our table name. private static final String
TABLE_NAME = "mycourses";

// below variable is for our id column.

private static final String ID_COL = "id";

// below variable is for our course name column private static
final String NAME_COL = "name";

// below variable id for our course duration column. private
static final String DURATION_COL = "duration";

// below variable for our course description column.

private static final String DESCRIPTION_COL = "description";

// below variable is for our course tracks column.

private static final String TRACKS_COL = "tracks";

// creating a constructor for our database handler. public
DBHandler(Context context) { super(context,
DB_NAME, null, DB_VERSION);
}
// below method is for creating a database by running a sqlite query @Override
public void onCreate(SQLiteDatabase db) {

    // on below line we are creating
    // an sqlite query and we are //
    defining our column names // along
    with their data types
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
String query = "CREATE TABLE " + TABLE_NAME + " ("
                + ID_COL + " INTEGER PRIMARY KEY
AUTOINCREMENT, "
                + NAME_COL + " TEXT,"
                + DURATION_COL + " TEXT,"
                + DESCRIPTION_COL + " TEXT," +
                TRACKS_COL + " TEXT)";

// at last we are calling a exec sql
// method to execute above sql query db.execSQL(query);

}
// this method is use to add new course to our sqlite database.
public void addNewCourse(String courseName, String courseDuraTon, String
courseDescripTon, String courseTracks) {

    // on below line we are creaTng a variable for // our
    sqlite database and calling writable method // as we are wriTng data in
    our database.

    SQLiteDatabase db = this.getWritableDatabase();

    // on below line we are creaTng a // variable
    for content values.

    ContentValues values = new ContentValues();

    // on below line we are passing all values // along with its key and value pair.

    values.put(NAME_COL, courseName); values.put(DURATION_COL, courseDuraTon);
    values.put(DESCRIPTION_COL, courseDescripTon); values.put(TRACKS_COL,
    courseTracks);

    // aoer adding all values we are passing // content values
    to our table.

    db.insert(TABLE_NAME, null, values);

    // at last we are closing our // database aoer adding
    database.

    db.close();
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}

@Override public void onUpgrade(SQLiteDatabase db, int
oldVersion, int newVersion) {

    // this method is called to check if the table exists already.
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME); onCreate(db);

}

}
```

Step 5: Working with the MainActivity.java file

Go to the MainActivity.java file and refer to the following code. Below is the code for the MainActivity.java file. Comments are added inside the code to understand the code in more detail.

```
import android.os.Bundle; import android.view.View; import android.widget.Button;
import android.widget.EditText;

import android.widget.Toast; import

androidx.appcompat.app.AppCompatActivity; public class

MainActivity extends AppCompatActivity { // creating variables for our

editText, button and dbHelper

private

EditText courseNameEdt, courseTracksEdt, courseDurationEdt,
courseDescriptionEdt; private Button addCourseBtn; private DBHelper dbHelper;

@Override protected void onCreate(Bundle

savedInstanceState) {

super.onCreate(savedInstanceState); setContentView(R.layout.activity_main);

// initializing all our variables.

courseNameEdt = findViewById(R.id.editCourseName); courseTracksEdt
= findViewById(R.id.editCourseTracks);
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
courseDuraTonEdt = findViewById(R.id.idEdtCourseDuraTon);
courseDescripTonEdt =
findViewById(R.id.idEdtCourseDescripTon); addCourseBtn =
findViewById(R.id.idBtnAddCourse);

// creaTng a new dbhandler class // and passing
our context to it.

dbHandler = new DBHandler(MainAcTvity.this);

// below line is to add on click listener for our add course bu\on.
addCourseBtn.setOnClickListener(new View.OnClickListener() {

    @Override public void
    onClick(View v) {

        // below line is to get data from all edit text fields. String
        courseName =
courseNameEdt.getText().toString();

        String courseTracks =
courseTracksEdt.getText().toString();

        String courseDuraTon =
courseDuraTonEdt.getText().toString();

        String courseDescripTon =
courseDescripTonEdt.getText().toString();

        // validaTng if the text fields are empty or not.
        if (courseName.isEmpty() && courseTracks.isEmpty()
&& courseDuraTon.isEmpty() && courseDescripTon.isEmpty()) {

            Toast.makeText(MainAcTvity.this, "Please enter
all the data..", Toast.LENGTH_SHORT).show(); return; }

        // on below line we are calling a method to add new // course to
sqlite data and pass all our values to it.

        dbHandler.addNewCourse(courseName,
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
courseDuraTon, courseDescripTon, courseTracks);
```

```
message.  
// after adding the data we are displaying a toast
```

```
Toast.makeText(MainAcTivity.this, "Course has been added.",  
Toast.LENGTH_SHORT).show();  
courseDuraTonEdt.setText("");  
courseTracksEdt.setText("");  
courseDescripTonEdt.setText(""); }  
});  
} }
```

OUTPUT :

The screenshot shows a mobile application interface titled "GFG App". It features a green header bar. Below the header, there are four text input fields with placeholder text: "Enter course Name", "Enter Course Duration", "Enter Course Tracks", and "Enter Course Description". At the bottom of the form, there is a green button labeled "Add Course". The status bar at the top of the screen shows the time as 10:10 AM, a data speed of 2.0KB/s, and a battery level of 37%.