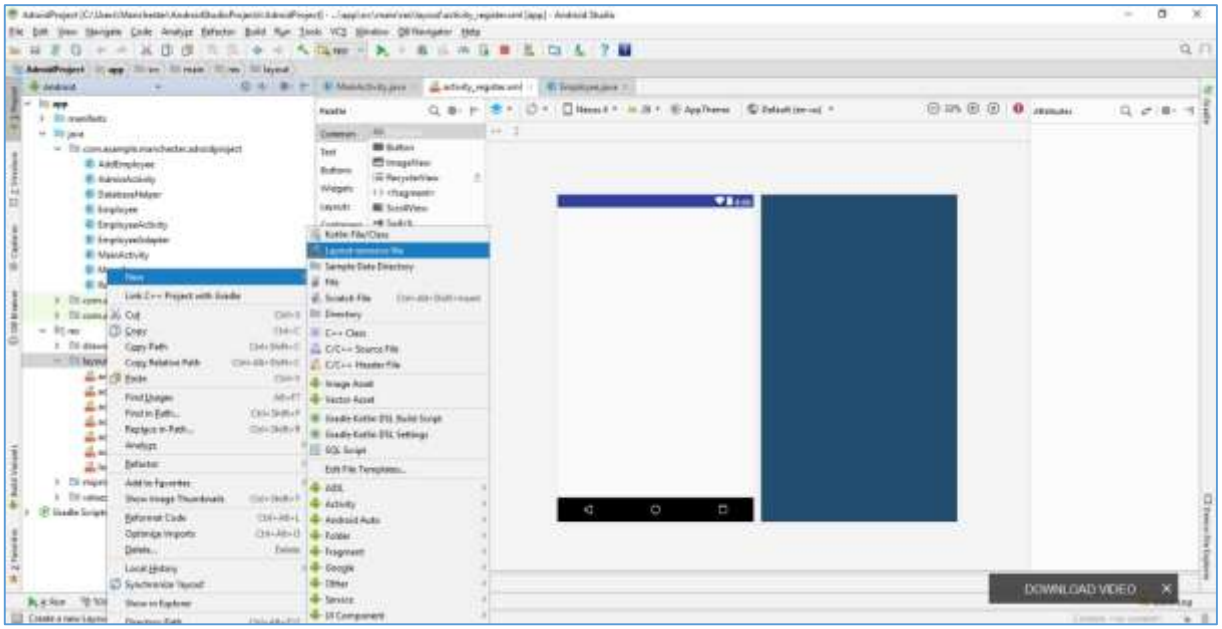
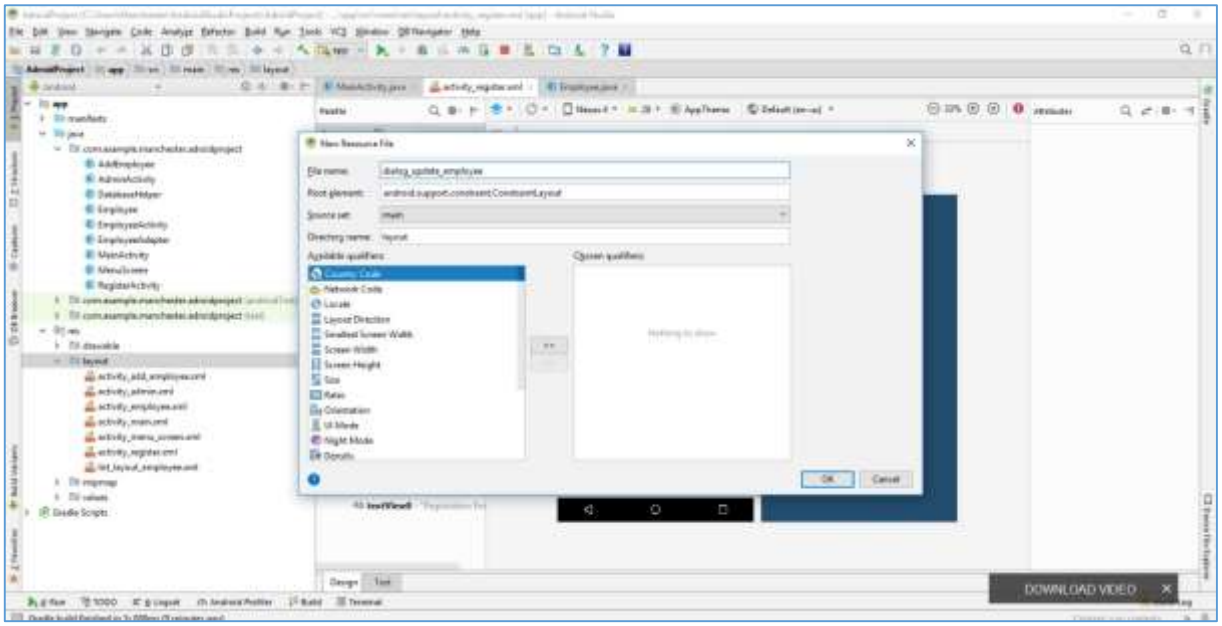
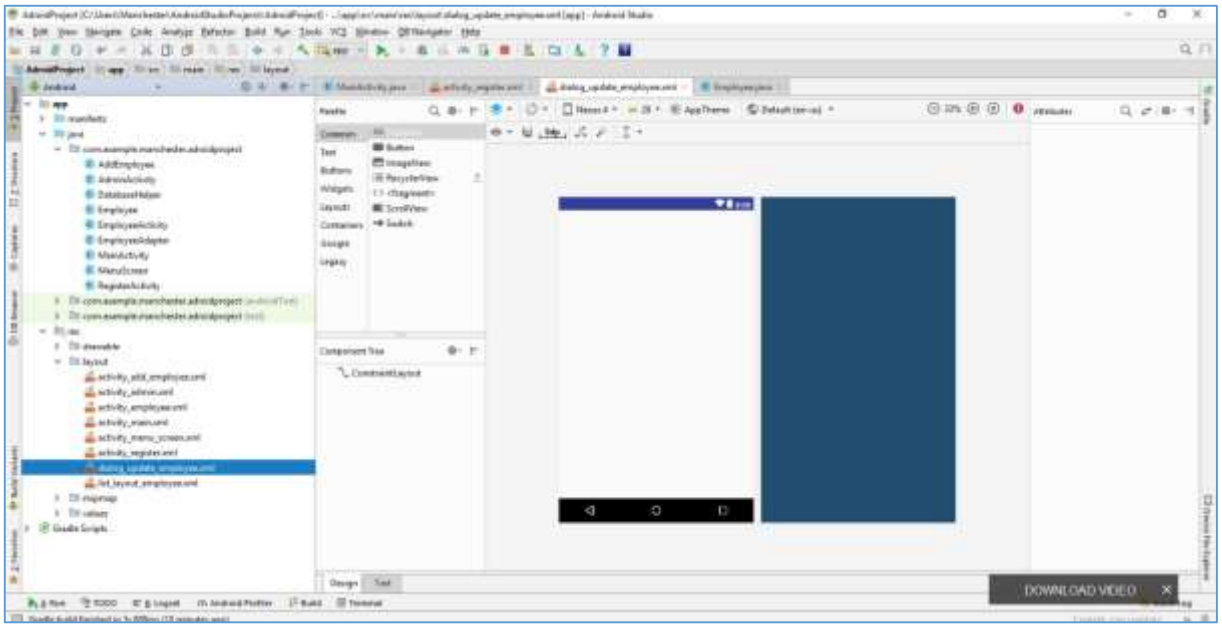
Part VIII: Update Employee Record

Step 1: Create a new layout and renamed it “dialog_update_employee.xml”



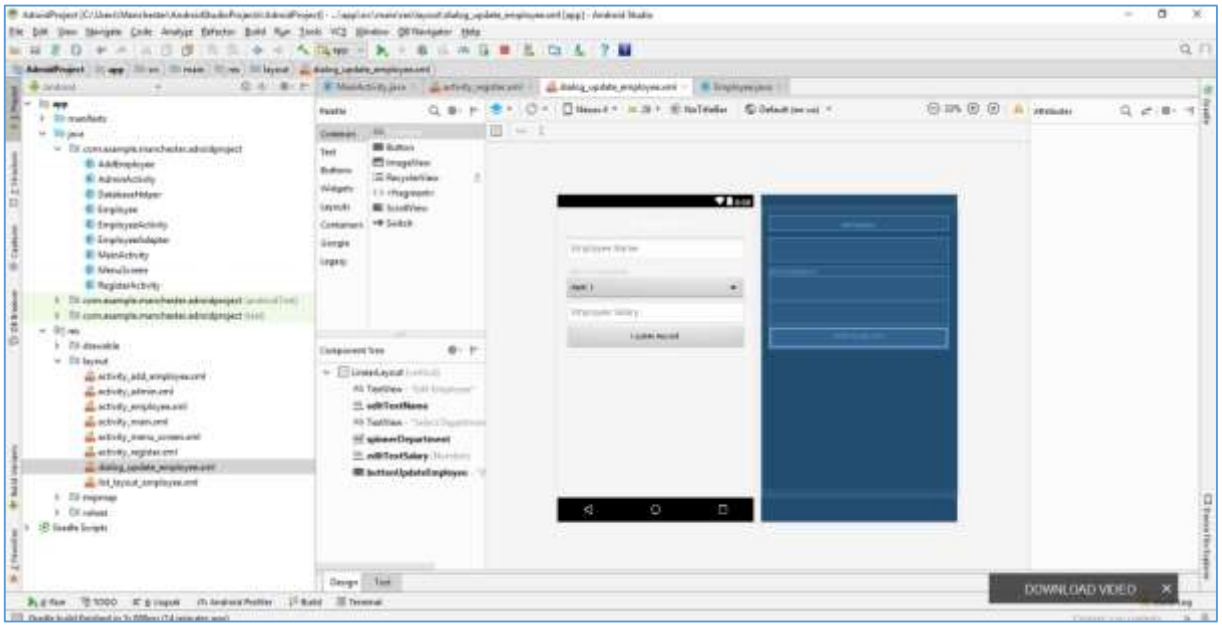




dialog_update_employee.xml

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="12dp"
        android:text="Edit Employee Record"
        android:textAlignment="center"
        android:textAppearance="@style/Base.TextAppearance.AppCompat.Large" />

    <EditText
        android:id="@+id/editTextName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Employee Name" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:paddingLeft="6dp"
        android:text="Select Department" />

    <Spinner
        android:id="@+id/spinnerDepartment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:entries="@array/departments" />

    <EditText
        android:id="@+id/editTextSalary"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:digits="0123456789"
        android:hint="Employee Salary"
```

```
        android:inputType="number" />

    <Button
        android:id="@+id/buttonUpdateEmployee"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Update Record" />
</LinearLayout>
```



Step 2: Create function *updateEmployee()* and *reloadEmployeesFromDatabase()* in file EmployeeAdapter.java

Function *updateEmployee()*

```
private void updateEmployee(final Employee employee) {
    final AlertDialog.Builder builder = new AlertDialog.Builder(mCtx);

    LayoutInflater inflater = LayoutInflater.from(mCtx);
    View view = inflater.inflate(R.layout.dialog_update_employee, null);
    builder.setView(view);


    final EditText editTextName = view.findViewById(R.id.editTextName);
    final EditText editTextSalary = view.findViewById(R.id.editTextSalary);
    final Spinner spinnerDepartment = view.findViewById(R.id.spinnerDepartment);

    editTextName.setText(employee.getName());
    editTextSalary.setText(String.valueOf(employee.getSalary()));

    final AlertDialog dialog = builder.create();
    dialog.show();

    view.findViewById(R.id.buttonUpdateEmployee).setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String name = editTextName.getText().toString().trim();
            String salary = editTextSalary.getText().toString().trim();
            String dept = spinnerDepartment.getSelectedItem().toString();
```

```
            if (name.isEmpty()) {
                editTextName.setError("Name can't be blank");
                editTextName.requestFocus();
                return;
            }

            if (salary.isEmpty()) {
                editTextSalary.setError("Salary can't be blank");
                editTextSalary.requestFocus();
                return;
            }


            String id = Integer.toString(employee.getId());
            Boolean empUpdate = db.empUpdate(id,name,dept,salary);

            Toast.makeText(mCtx, "Employee Updated", Toast.LENGTH_SHORT).show();
            reloadEmployeesFromDatabase();
            dialog.dismiss();



        }
    });
}
```

Function *reloadEmployeesFromDatabase()*

```
private void reloadEmployeesFromDatabase() {


    Cursor cursorEmployees = db.viewEmployee();

    if (cursorEmployees.moveToFirst()) {
        employeeList.clear();
        do {
            employeeList.add(new Employee(
                    cursorEmployees.getInt(0),
                    cursorEmployees.getString(1),
                    cursorEmployees.getString(2),
                    cursorEmployees.getString(3),
                    cursorEmployees.getDouble(4)
            ));
        } while (cursorEmployees.moveToNext());
    }
    cursorEmployees.close();
    notifyDataSetChanged();
}
```

EmployeeAdapter.java

```
package com.example.manchester.adroidproject;

import android.content.Context;
import android.database.Cursor;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
```

```
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import java.util.List;

public class EmployeeAdapter extends ArrayAdapter<Employee> {
    Context mCtx;
    int listLayoutRes;
    List<Employee> employeeList;
    DatabaseHelper db;

    public EmployeeAdapter(Context mCtx, int listLayoutRes, List<Employee>
employeeList, DatabaseHelper db) {
        super(mCtx, listLayoutRes, employeeList);

        this.mCtx = mCtx;
        this.listLayoutRes = listLayoutRes;
        this.employeeList = employeeList;
        this.db = db;
    }

    @NonNull
    @Override
    public View getView(int position, @Nullable View convertView, @NonNull
ViewGroup parent) {
        LayoutInflater inflater = LayoutInflater.from(mCtx);
        View view = inflater.inflate(listLayoutRes,null);

        final Employee employee = employeeList.get(position);

        TextView textViewName = view.findViewById(R.id.textViewName);
        TextView textViewDept = view.findViewById(R.id.textViewDepartment);
        TextView textViewSalary = view.findViewById(R.id.textViewSalary);
        TextView textViewJoiningDate =
view.findViewById(R.id.textViewJoiningDate);

        textViewName.setText(employee.getName());
        textViewDept.setText(employee.getDept());
        textViewSalary.setText(String.valueOf(employee.getSalary()));
        textViewJoiningDate.setText(employee.getJoiningDate());

        Button buttonEdit = view.findViewById(R.id.btnEdit);
        Button buttonDelete = view.findViewById(R.id.btnDelete);

        buttonEdit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                updateEmployee(employee);
            }
        });

        return view;
    }

    private void updateEmployee(final Employee employee) {
        final AlertDialog.Builder builder = new AlertDialog.Builder(mCtx);

        LayoutInflater inflater = LayoutInflater.from(mCtx);
        View view = inflater.inflate(R.layout.dialog_update_employee, null);
        builder.setView(view);


        final EditText editTextName = view.findViewById(R.id.editTextName);
        final EditText editTextSalary = view.findViewById(R.id.editTextSalary);
```

```
        final Spinner spinnerDepartment =
view.findViewById(R.id.spinnerDepartment);

        editTextName.setText(employee.getName());
        editTextSalary.setText(String.valueOf(employee.getSalary()));

        final AlertDialog dialog = builder.create();
        dialog.show();

        view.findViewById(R.id.buttonUpdateEmployee).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String name = editTextName.getText().toString().trim();
                String salary = editTextSalary.getText().toString().trim();
                String dept = spinnerDepartment.getSelectedItem().toString();

                if (name.isEmpty()) {
                    editTextName.setError("Name can't be blank");
                    editTextName.requestFocus();
                    return;
                }

                if (salary.isEmpty()) {
                    editTextSalary.setError("Salary can't be blank");
                    editTextSalary.requestFocus();
                    return;
                }


                String id = Integer.toString(employee.getId());
                Boolean empUpdate = db.empUpdate(id,name,dept,salary);

                Toast.makeText(mCtx, "Employee Updated",
Toast.LENGTH_SHORT).show();
                reloadEmployeesFromDatabase();
                dialog.dismiss();



            }
        });
    }


    private void reloadEmployeesFromDatabase() {


        Cursor cursorEmployees = db.viewEmployee();

        if (cursorEmployees.moveToFirst()) {
            employeeList.clear();
            do {
                employeeList.add(new Employee(
                        cursorEmployees.getInt(0),
                        cursorEmployees.getString(1),
                        cursorEmployees.getString(2),
                        cursorEmployees.getString(3),
                        cursorEmployees.getDouble(4)
                ));
            } while (cursorEmployees.moveToNext());
        }
        cursorEmployees.close();
        notifyDataSetChanged();
    }
}
```

Step 3: Create function *empUpdate()* in DatabaseHelper.java

Function *empUpdate()*

```
public Boolean empUpdate(String id, String name, String dept, String salary) {
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery("update employees set name=?, department=?,
salary=? WHERE id=?",new String[]{name,dept,salary,id});
    if (cursor.getCount()>0) return true;
    else return false;
}
```

DatabaseHelper.java

```
package com.example.manchester.adroidproject;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class DatabaseHelper extends SQLiteOpenHelper {

    private static final int DATABASE_VERSION = 1;

    public DatabaseHelper(Context context) {
        super(context, "DBEmployee",null,DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        //create table user
        db.execSQL("create table user (email text primary key, password text)");

        //creatable table employee
        db.execSQL("create table employees (id integer primary key autoincrement,
name text, department text, joiningdate datetime, salary double)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        Log.w(DatabaseHelper.class.getName(),"Upgrading database from version " +
oldVersion + " to "
                + newVersion + ",which will destroy all old data");
        db.execSQL("drop table if exists user");
        onCreate(db);
    }

    public Boolean chkemail(String a1) {
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery("select * from user where email=?", new
String[]{a1});
        if (cursor.getCount()>0) return false;
        else return true;
    }

    public Boolean insert1(String a1, String a2) {
        SQLiteDatabase db = this.getReadableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put("email",a1);
```

```
        contentValues.put("password",a2);
        long ins = db.insert("user",null,contentValues);
        if (ins==1) return true;
        else return false;
    }

    public Boolean login(String email, String password) {
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery("select * from user where email=? and
password=?", new String[]{email,password});
        if (cursor.getCount()>0) return true;
        else return false;
    }

    public Boolean addEmployee(String employeeName, String department, String
joiningdate, String salary) {
        SQLiteDatabase db = this.getReadableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put("name",employeeName);
        contentValues.put("department",department);
        contentValues.put("joiningdate",joiningdate);
        contentValues.put("salary",salary);
        long ins = db.insert("employees",null,contentValues);
        if (ins==1) return true;
        else return false;
    }

    public Cursor viewEmployee() {
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery("select * from employees",null);
        return cursor;
    }

    public Boolean empUpdate(String id, String name, String dept, String salary)
{
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery("update employees set name=?, department=?,
salary=? WHERE id=?",new String[]{name,dept,salary,id});
        if (cursor.getCount()>0) return true;
        else return false;
    }
}
```

Run and excute your program...