NAME:- DUBEY KARAN SANJEEV
CLASS:- B.E – 4
ROLL NO:- 04
BATCH:- A

Experiment 6

Develop an application that makes use of database.

Source Code:

Dependencies

```
dependencies {
  implementation fileTree(dir: 'libs', include: ['*.jar']) implementation
  'androidx.appcompat:appcompat:1.0.2' implementation
  'androidx.constraintlayout:constraintlayout:1.1.3'
  testImplementation 'junit:junit:4.12'
  androidTestImplementation 'androidx.test.ext:junit:1.1.0'
  androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'

  //add these libraries

  //support design
  implementation "com.android.support:design:29.0.2"

  //card view
  implementation "com.android.support:cardview-v7: 29.0.2"

  //recyclerview
  implementation "com.android.support:recyclerview-v7: 29.0.2"


  //room implementation "android.arch.persistence.room:runtime: 1.1.1" annotationProcessor
  "android.arch.persistence.room:compiler: 1.1.1" testImplementation
  "android.arch.persistence.room:testing: 1.1.1"
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:padding="8dp" tools:context=".MainActivity">

  <androidx.recyclerview.widget.RecyclerView
     android:id="@+id/recyclerview_tasks" android:layout_width="match_parent"
     android:layout_height="match_parent" />
```

```
    <com.google.android.material.floatingactionbutton.FloatingActionButton
    android:id="@+id/floating_button_add"
        android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:layout_alignParentBottom="true" android:layout_alignParentRight="true"
        android:layout_margin="8dp"  android:backgroundTint="@color/colorPrimaryDark"
        android:src="@drawable/ic_add"
        android:tint="@color/colorLight" app:fabSize="normal"
        />
</RelativeLayout>
```

Recyclerview_tasks.xml

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

    <androidx.cardview.widget.CardView android:layout_width="match_parent"
        android:layout_height="wrap_content" android:layout_marginBottom="3dp">

        <LinearLayout android:layout_width="match_parent" android:layout_height="wrap_content"
            android:orientation="vertical" android:padding="7dp">

            <TextView android:id="@+id/textViewStatus" android:layout_width="match_parent"
                android:layout_height="wrap_content" android:layout_marginBottom="5dp"
                android:background="@color/colorPrimaryDark" android:text="Completed"
                android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium
                "
                android:textColor="@color/colorLight" android:textStyle="bold" />

            <TextView android:id="@+id/textViewTask" android:layout_width="match_parent"
                android:layout_height="wrap_content" android:text="Go Bring Eggs"
                android:textAppearance="@style/Base.TextAppearance.AppCompat.Headline"
                />

            <TextView android:id="@+id/textViewDesc" android:layout_width="match_parent"
                android:layout_height="wrap_content" android:text="Bring
                6 eggs from super market"
                android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />

            <TextView android:id="@+id/textViewFinishBy" android:layout_width="match_parent"
                android:layout_height="wrap_content" android:text="5pm today"

                android:textAppearance="@style/Base.Text
                Appearance.AppCompat.Medium" />
```

```
        </LinearLayout>
    </androidx.cardview.widget.CardView> </RelativeLayout>
```

Task.java package com.example.first_application;

```
  @Entity public class Task implements Serializable
  {
    @PrimaryKey(autoGenerate = true) private
int id;
    @ColumnInfo(name = "task") private String task;

    @ColumnInfo(name = "desc") private
    String desc;

    @ColumnInfo(name = "finish_by") private String finishBy;

    @ColumnInfo(name = "finished") private boolean finished;

    public int getId() { return  id;
       }
 public void setId(int id) { this.id = id;
    }

    public String getTask() { return
       task;
    }

    public void setTask(String task) { this.task =  task;
    }

    public String getDesc() { return
       desc;
    }

    public void setDesc(String desc) { this.desc =
       desc;
    }
    public String getFinishBy() { return
       finishBy;
    }


    public void setFinishBy(String finishBy) {  this.finishBy
       = finishBy;
```

```
    }

    public boolean isFinished() {  return
        finished;
    }
    public void setFinished(boolean finished) {  this.finished
        = finished;
    }
  }
```

TaskDao.java(Interface) package com.example.first_application;

```
  @Dao
  public interface TaskDao {

    @Query("SELECT * FROM task")
    List<Task> getAll();

    @Insert void
    insert(Task task);

    @Delete
    void delete(Task task);

    @Update
    void update(Task task);
}
```

Appdatabase.java package com.example.first_application;

```
  @Database(entities = {Task.class}, version = 1) public abstract class
  AppDatabase extends RoomDatabase {  public
    abstract TaskDao taskDao();
  }
```

Databaseclient.java

```
  package com.example.first_application; public
class DatabaseClient {          private Context mCtx; private
    static DatabaseClient
    mInstance;

    //our app database object private
```

```
    AppDatabase appDatabase; private
    DatabaseClient(Context mCtx) { this.mCtx =
    mCtx;


        //creating the app database with Room database builder, MyToDos is the name of the
        database appDatabase = Room.databaseBuilder(mCtx, AppDatabase.class,
        "MyToDos").build();
    }


    public static synchronized DatabaseClient getInstance(Context mCtx) { if
        (mInstance == null) { mInstance = new DatabaseClient(mCtx);
        }
        return mInstance;
    }

    public AppDatabase getAppDatabase() { return appDatabase;
    }
}
```

Addtaskactivity

```
package com.example.first_application;

public class Addtaskactivity extends AppCompatActivity { private

    EditText editTextTask, editTextDesc, editTextFinishBy;

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

        editTextTask = findViewById(R.id.editTextTask); editTextDesc =
        findViewById(R.id.editTextDesc);      editTextFinishBy       =
        findViewById(R.id.editTextFinishBy);

        findViewById(R.id.button_save).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) { saveTask();
            } });

}
```

```
private void saveTask() { final String
        sTask =editTextTask.getText().toString().trim();
        final String sDesc =editTextDesc.getText().toString().trim();
        final String sFinishBy =editTextFinishBy.getText().toString().trim();

        if  (sTask.isEmpty())  { editTextTask.setError("com.example.first_application.Task required");
            editTextTask.requestFocus();
            return;
        }

        if (sDesc.isEmpty()) { editTextDesc.setError("Desc
            required"); editTextDesc.requestFocus(); return;
        }

        if (sFinishBy.isEmpty()) { editTextFinishBy.setError("Finish by
            required"); editTextFinishBy.requestFocus(); return;
        }
        class SaveTask extends AsyncTask<Void, Void, Void> {

            @Override
            protected Void doInBackground(Void... voids) {

                //creating a task Task task = new
                Task();       task.setTask(sTask);
                task.setDesc(sDesc);
                task.setFinishBy(sFinishBy)       ;
                task.setFinished(false);


                //adding to database
                DatabaseClient.getInstance(getApplicationContext()).getAppDatabase()  .taskDao()
                        .insert(task); return null;
            }

            @Override
            protected void onPostExecute(Void aVoid) { super.onPostExecute(aVoid);
                finish(); startActivity(new Intent(getApplicationContext(),  MainActivity.class));
                Toast.makeText(getApplicationContext(), "Saved", Toast.LENGTH_LONG).show();
            }
        }
        SaveTask st = new SaveTask(); st.execute();
    }
}
```

Taskadapter.java

```
package com.example.first_application; public class TasksAdapter extends


RecyclerView.Adapter<TasksAdapter.TasksViewHolder> {
```

```
private Context mCtx;
private List<Task> taskList;

public TasksAdapter(Context mCtx, List<Task> taskList)
 { this.mCtx
    = mCtx; this.taskList
    = taskList;
}
@Override
public TasksViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(mCtx).inflate(R.layout.recyclerview_tasks, parent, false); return
    new TasksViewHolder(view);
}
@Override
public void onBindViewHolder(TasksViewHolder holder, int position) {
    Task t = taskList.get(position); holder.textViewTask.setText(t.getTask());
    holder.textViewDesc.setText(t.getDesc());
    holder.textViewFinishBy.setText(t.getFinishBy());


    if (t.isFinished()) holder.textViewStatus.setText("Completed"); else
    holder.textViewStatus.setText("Not Completed");
}

@Override
public int getItemCount() { return taskList.size();
}

class TasksViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {

    TextView textViewStatus, textViewTask, textViewDesc, textViewFinishBy; public

    TasksViewHolder(View itemView) { super(itemView);


        textViewStatus = itemView.findViewById(R.id.textViewStatus);
        textViewTask = itemView.findViewById(R.id.textViewTask); textViewDesc =
        itemView.findViewById(R.id.textViewDesc); textViewFinishBy =
        itemView.findViewById(R.id.textViewFinishBy);
        itemView.setOnClickListener(this);
    }
    @Override
    public void onClick(View view) {
        Task task = taskList.get(getAdapterPosition());

        Intent intent = new Intent(mCtx, Updatetaskactivity.class); intent.putExtra("task", task);
```

```
            mCtx.startActivity(intent);
        }
    }
}
```

MainActivity.java

```
package com.example.first_application;

public class MainActivity extends AppCompatActivity {

    private FloatingActionButton buttonAddTask; private RecyclerView
    recyclerView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);   setContentView(R.layout.activity_main);
        recyclerView =
        findViewById(R.id.recyclerview_tasks);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));

        buttonAddTask = findViewById(R.id.floating_button_add);
        buttonAddTask.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(MainActivity.this, Addtaskactivity.class); startActivity(intent);
            }
        });
        getTasks();
    }
    private void getTasks() { class GetTasks extends AsyncTask<Void,
        Void, List<Task>> {

            @Override
            protected List<Task> doInBackground(Void... voids) {
                List<Task> taskList = DatabaseClient
                        .getInstance(getApplicationContext())
                        .getAppDatabase()
                        .taskDao()



                        .getAll(); return taskList;
```

```
        }
        @Override
        protected void onPostExecute(List<Task> tasks) { super.onPostExecute(tasks);
            TasksAdapter adapter = new TasksAdapter(MainActivity.this, tasks);
            recyclerView.setAdapter(adapter);
        }
    }


    GetTasks gt = new GetTasks(); gt.execute();


  }
}
```

Updatetaskactivity

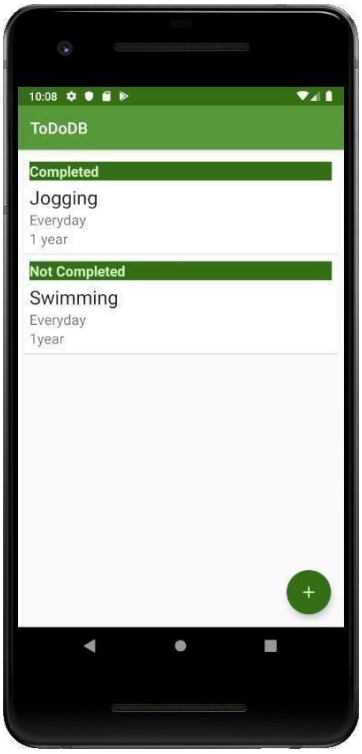
```
package com.example.first_application;

public class Updatetaskactivity extends AppCompatActivity {

  private EditText editTextTask, editTextDesc, editTextFinishBy; private CheckBox
  checkBoxFinished;

  @Override
  protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_updatetaskactivity);
    editTextTask = findViewById(R.id.editTextTask); editTextDesc =
    findViewById(R.id.editTextDesc);      editTextFinishBy       =
    findViewById(R.id.editTextFinishBy);

    checkBoxFinished = findViewById(R.id.checkBoxFinished); final Task task =

    (Task) getIntent().getSerializableExtra("task"); loadTask(task);

    findViewById(R.id.button_update).setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View view) {
        Toast.makeText(getApplicationContext(), "Clicked", Toast.LENGTH_LONG).show();
        updateTask(task);
      }
    });
```

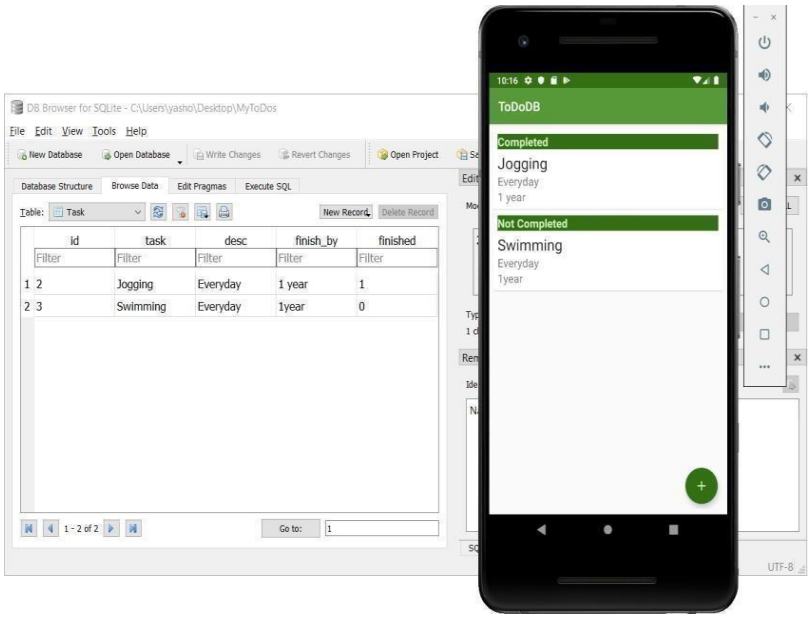
```
    findViewById(R.id.button_delete).setOnClickListener(new View.OnClickListener() {
        @Override public void
        onClick(View view) {
            AlertDialog.Builder builder = new AlertDialog.Builder(Updatetaskactivity.this);
            builder.setTitle("Are you sure?");
            builder.setPositiveButton("Yes", new DialogInterface.OnClickListener() {  @Override
                public void onClick(DialogInterface dialogInterface, int i) { deleteTask(task);
                }
            });
            builder.setNegativeButton("No", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialogInterface, int i) {


                }
            });


              }


        }
    });
AlertDialog ad = builder.create(); ad.show();

private void loadTask(Task task) { editTextTask.setText(task.getTask());
    editTextDesc.setText(task.getDesc()); editTextFinishBy.setText(task.getFinishBy());
    checkBoxFinished.setChecked(task.isFinished());
}

private void updateTask(final Task task) { final String sTask =
    editTextTask.getText().toString().trim(); final String sDesc =
    editTextDesc.getText().toString().trim(); final String sFinishBy =
    editTextFinishBy.getText().toString().trim();

    if (sTask.isEmpty()) { editTextTask.setError("Task required");
        editTextTask.requestFocus(); return;
    }


    if (sDesc.isEmpty()) { editTextDesc.setError("Desc
        required"); editTextDesc.requestFocus(); return;
    }

    if (sFinishBy.isEmpty()) { editTextFinishBy.setError("Finish by
        required"); editTextFinishBy.requestFocus(); return;
```

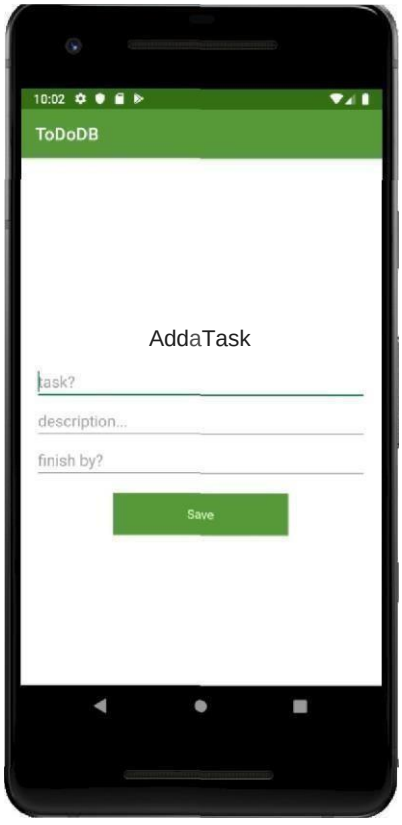
```
}

    class UpdateTask extends AsyncTask<Void, Void, Void> {
    @SuppressLint("WrongThread")@Override
       protected Void doInBackground(Void... voids) { task.setTask(sTask); task.setDesc(sDesc);
          task.setFinishBy(sFinishBy); task.setFinished(checkBoxFinished.isChecked());
          DatabaseClient.getInstance(getApplicationContext()).getAppDatabase()  .taskDao()
                .update(task); return null;
       }
       @Override
       protected void onPostExecute(Void aVoid) { super.onPostExecute(aVoid);
          Toast.makeText(getApplicationContext(), "Updated", Toast.LENGTH_LONG).show();
          finish();
          startActivity(new Intent(Updatetaskactivity.this, MainActivity.class));
       }
    }

    UpdateTask ut = new UpdateTask(); ut.execute();
  }


  private void deleteTask(final Task task) { class DeleteTask extends AsyncTask<Void,
      Void, Void> {

       @Override
       protected Void doInBackground(Void... voids) {
          DatabaseClient.getInstance(getApplicationContext()).getAppDatabase() .taskDao()


                .delete(task); return null;
       }
       @Override
       protected void onPostExecute(Void aVoid) { super.onPostExecute(aVoid);
          Toast.makeText(getApplicationContext(), "Deleted", Toast.LENGTH_LONG).show();
          finish();
          startActivity(new Intent(Updatetaskactivity.this, MainActivity.class));
       }
    }


    DeleteTask dt = new DeleteTask(); dt.execute();
```

```
    }
}
```

Output


10:08
ToDoDB
Completed
Jogging
Everyday
1 year
Not Completed
Swimming
Everyday
1year


DB Browser for SQLite - C:\Users\yasho\Desktop\MyToDos
File Edit View Tools Help
New Database
Open Database
Write Changes
Revert Changes
Open Project
Database Structure
Browse Data
Edit Pragmas
Execute SQL
Table: Task
New Record
Delete Record
id
task
desc
finish_by
finished
Filter
1 2 Jogging Everyday 1 year 1
2 3 Swimming Everyday 1year 0
1 - 2 of 2
Go to: 1
10:16
ToDoDB
Completed
Jogging
Everyday
1 year
Not Completed
Swimming
Everyday
1year
UTF-8


10:02
ToDoDB
AddaTask
task?
description...
finish by?
Save


10:07
ToDoDB
Update Task
Jogging
Everyday
1 year
Mark as finished
Update
Delete