EX 6	Insert, Update, Delete, and Display operations using SQLite

Aim

To create an application to perform insert, update, delete, and display operations using SQLite in Android.

Definitions

SQLite

SQLite is a lightweight, serverless, and self-contained relational database management system (RDBMS) that stores an entire database in a single disk file. It is a C-language library embedded directly into applications rather than running as a separate service, making it ideal for mobile apps, IoT devices, and local storage. It requires zero configuration and is highly portable.

Insert, Update, Delete and Display Operations

1. INSERT (Create)
Adds new data, records, or rows into a database table or a data structure.

2. DISPLAY / SELECT (Read)
Retrieves or views existing data from a database or storage.

3. UPDATE (Modify)
Modifies or edits existing data within a database table.

4. DELETE (Remove)
Removes one or more existing records or rows from a database table.

Procedure

1. Open Android Studio IDE → File → New → New Project → specify the application name "CRUD" and company domain "com.mad.crud" → click "next" → choose Minimum SDK "API 17:Android 4.2(Jelly Bean)" → click "Next" → choose "Blank Activity" → click "next" → specify the Activity Name "MainActivity" → click "Finish".
2. Open MainActivity.java under app/java/ crud.mad.com. crud and type the following codes:

MainActivity.java

```
package crud.mad.com.crud;

import android.support.v7.app.AppCompatActivity; // Non-AndroidX
import android.app.AlertDialog;
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import android.view.View;


public class MainActivity extends AppCompatActivity {
   DatabaseHelper myDb;
   EditText editName, editDesig, editId;
   Button btnAdd, btnView, btnUpdate, btnDelete;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      myDb = new DatabaseHelper(this);

      editName = (EditText) findViewById(R.id.et_name);
      editDesig = (EditText) findViewById(R.id.et_desig);
      editId = (EditText) findViewById(R.id.et_id);
      btnAdd = (Button) findViewById(R.id.btn_add);
      btnView = (Button) findViewById(R.id.btn_view);
      btnUpdate = (Button) findViewById(R.id.btn_update);
      btnDelete = (Button) findViewById(R.id.btn_delete);

      // CREATE
      btnAdd.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            boolean inserted = myDb.insertEmployee(editName.getText().toString(),
editDesig.getText().toString());
```

```
                Toast.makeText(MainActivity.this, inserted ? "Success" : "Failed",
Toast.LENGTH_SHORT).show();
            }
        });

        // READ
        btnView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Cursor res = myDb.getAllEmployees();
                if(res.getCount() == 0) { showMsg("Error", "No records found"); return; }
                StringBuffer buffer = new StringBuffer();
                while (res.moveToNext()) {
                    buffer.append("ID: "+res.getString(0)+"\nName: "+res.getString(1)+"\nPos:
"+res.getString(2)+"\n\n");
                }
                showMsg("Employees", buffer.toString());
            }
        });

        // UPDATE
        btnUpdate.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                boolean updated = myDb.updateEmployee(editId.getText().toString(),
editName.getText().toString(), editDesig.getText().toString());
                Toast.makeText(MainActivity.this, updated ? "Updated" : "Error",
Toast.LENGTH_SHORT).show();
            }
        });

        // DELETE
        btnDelete.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Integer deleted = myDb.deleteEmployee(editId.getText().toString());
                Toast.makeText(MainActivity.this, deleted > 0 ? "Deleted" : "ID not found",
Toast.LENGTH_SHORT).show();
            }
        });
        Button btnClear = (Button) findViewById(R.id.btn_clear);
        btnClear.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Clear all EditText fields
                editName.setText("");
                editDesig.setText("");
                editId.setText("");


                Toast.makeText(MainActivity.this, "Fields Cleared",
Toast.LENGTH_SHORT).show();
      }
    });
  }

  public void showMsg(String title, String msg) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setCancelable(true);
    builder.setTitle(title).setMessage(msg).show();
  }

}
```

3. Right click on crud.mad.com. crud package → New → Java Class → Name: DatabaseHelper → ok. Type the following codes in DatabaseHelper.java,

DatabaseHelper.java

```
package crud.mad.com.crud;

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

public class DatabaseHelper extends SQLiteOpenHelper {
  public static final String DATABASE_NAME = "Company.db";
  public static final String TABLE_NAME = "employee_table";
  public static final String COL_1 = "ID";
  public static final String COL_2 = "NAME";
  public static final String COL_3 = "DESIGNATION";

  public DatabaseHelper(Context context) {
    super(context, DATABASE_NAME, null, 1);
  }

  @Override
  public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE " + TABLE_NAME + " (ID INTEGER PRIMARY
KEY AUTOINCREMENT, NAME TEXT, DESIGNATION TEXT)");
  }

  @Override
  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
    onCreate(db);
  }

  public boolean insertEmployee(String name, String desig) {
    SQLiteDatabase db = this.getWritableDatabase();
```

```
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_2, name);
        contentValues.put(COL_3, desig);
        long result = db.insert(TABLE_NAME, null, contentValues);
        return result != -1;
    }

    public Cursor getAllEmployees() {
        SQLiteDatabase db = this.getWritableDatabase();
        return db.rawQuery("SELECT * FROM " + TABLE_NAME, null);
    }

    public boolean updateEmployee(String id, String name, String desig) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_1, id);
        contentValues.put(COL_2, name);
        contentValues.put(COL_3, desig);
        db.update(TABLE_NAME, contentValues, "ID = ?", new String[]{id});
        return true;
    }

    public Integer deleteEmployee(String id) {
        SQLiteDatabase db = this.getWritableDatabase();
        return db.delete(TABLE_NAME, "ID = ?", new String[]{id});
    }
}
```

4. Open activity_main.xml under app/res/layout and type the following codes:

activity_main.xml

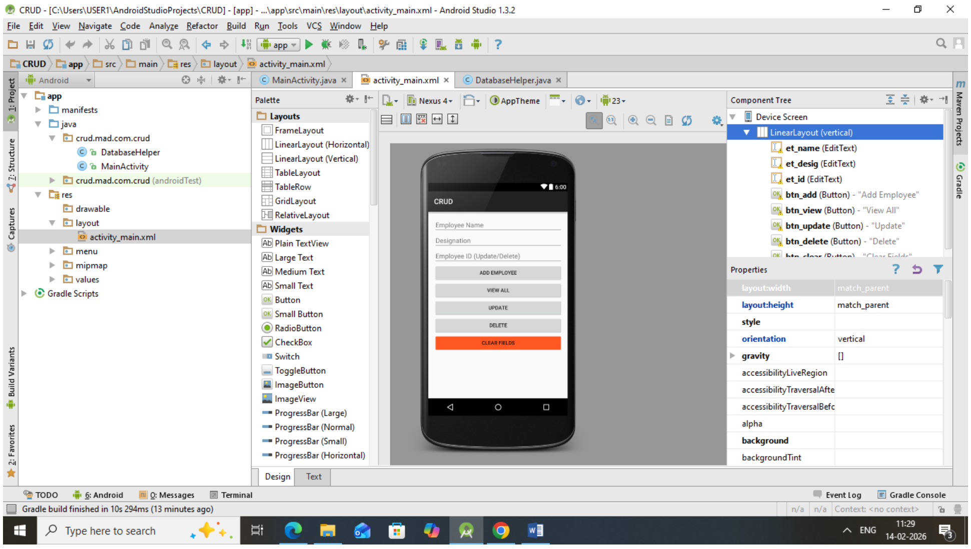
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
android:padding="16dp">

<EditText android:id="@+id/et_name" android:layout_width="match_parent"
android:layout_height="wrap_content" android:hint="Employee Name"/>
<EditText android:id="@+id/et_desig" android:layout_width="match_parent"
android:layout_height="wrap_content" android:hint="Designation"/>
<EditText android:id="@+id/et_id" android:layout_width="match_parent"
android:layout_height="wrap_content" android:hint="Employee ID (Update/Delete)"/>

<Button android:id="@+id/btn_add" android:layout_width="match_parent"
android:layout_height="wrap_content" android:text="Add Employee"/>
<Button android:id="@+id/btn_view" android:layout_width="match_parent"
android:layout_height="wrap_content" android:text="View All"/>
```

```
<Button android:id="@+id/btn_update" android:layout_width="match_parent"
android:layout_height="wrap_content" android:text="Update"/>
<Button android:id="@+id/btn_delete" android:layout_width="match_parent"
android:layout_height="wrap_content" android:text="Delete"/>
    <Button
        android:id="@+id/btn_clear"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Clear Fields"
        android:backgroundTint="#FF5722"/>
</LinearLayout>
```

5. The design of the application will be as follows:



CRUD - [C:\Users\USER1\AndroidStudioProjects\CRUD] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 1.3.2

Component Tree
- Device Screen
  - LinearLayout (vertical)
    - et_name (EditText)
    - et_desig (EditText)
    - et_id (EditText)
    - btn_add (Button) - "Add Employee"
    - btn_view (Button) - "View All"
    - btn_update (Button) - "Update"
    - btn_delete (Button) - "Delete"
    - btn_clear (Button) - "Clear Fields"

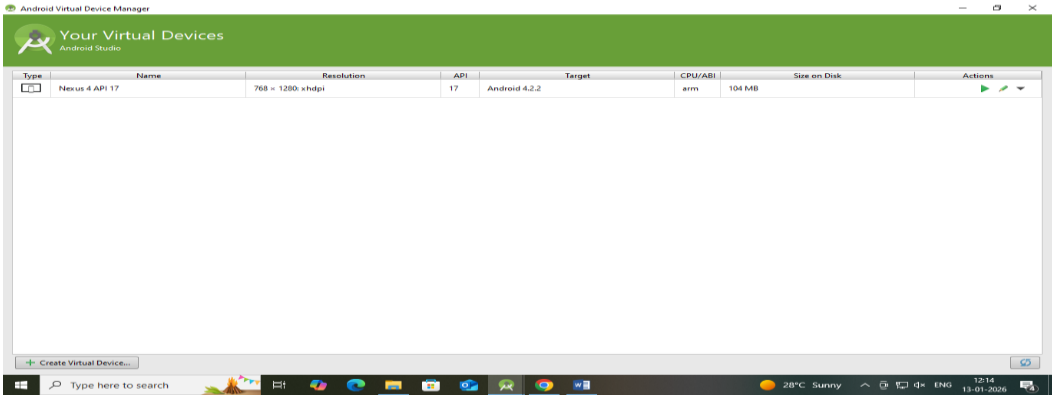
Properties

layout:width	match_parent
layout:height	match_parent
style	
orientation	vertical
gravity	[]

Gradle build finished in 10s 294ms (13 minutes ago)

6. Go to Tools → android → AVD Manager → click “+ create a virtual device” → select “phone” from category → select “Nexus 4” from the list → click “next” → select Release name: Jelly Bean, API Level: 17, ABI: armeabi-v7a, Target: Android 4.2.2 from the list → click “next” → Choose orientation “portrait” → click “finish”.

The following window will appear after configuring AVD:



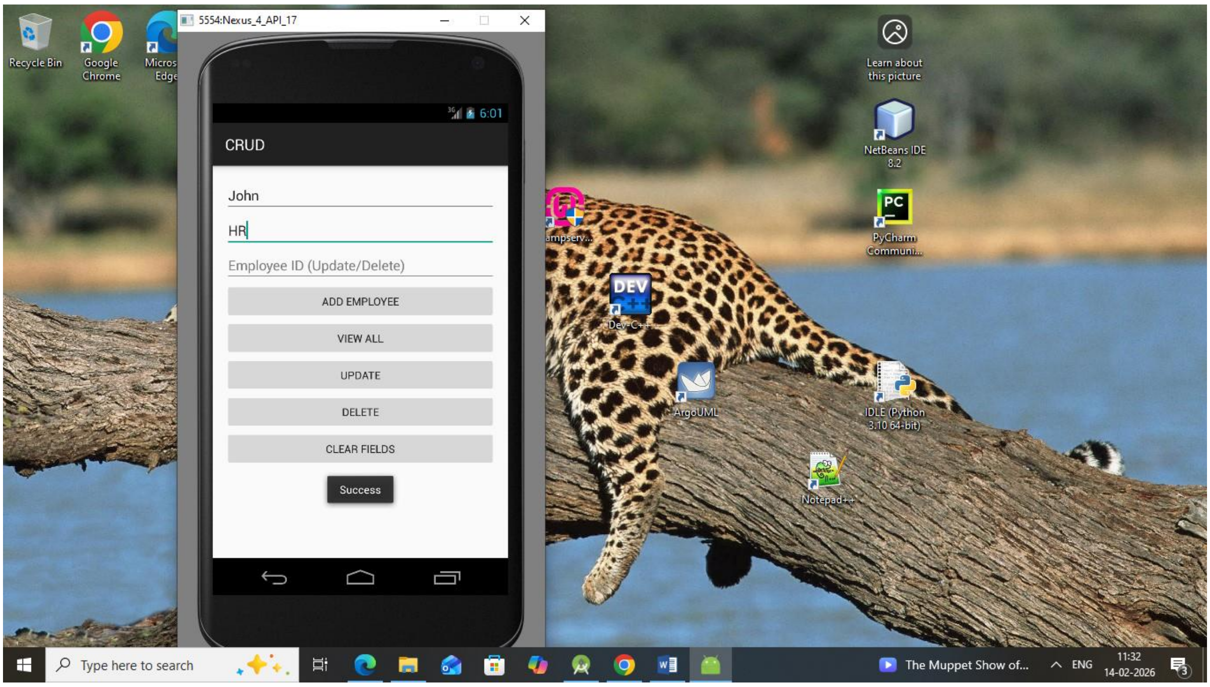
Android Virtual Device Manager

Your Virtual Devices
Android Studio

Type	Name	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	Nexus 4 API 17	768 × 1280: xhdpi	17	Android 4.2.2	arm	104 MB	

7. Click “Run app” button in the Android Studio → choose android virtual device → click “ok”.

Output



5554:Nexus_4_API_17

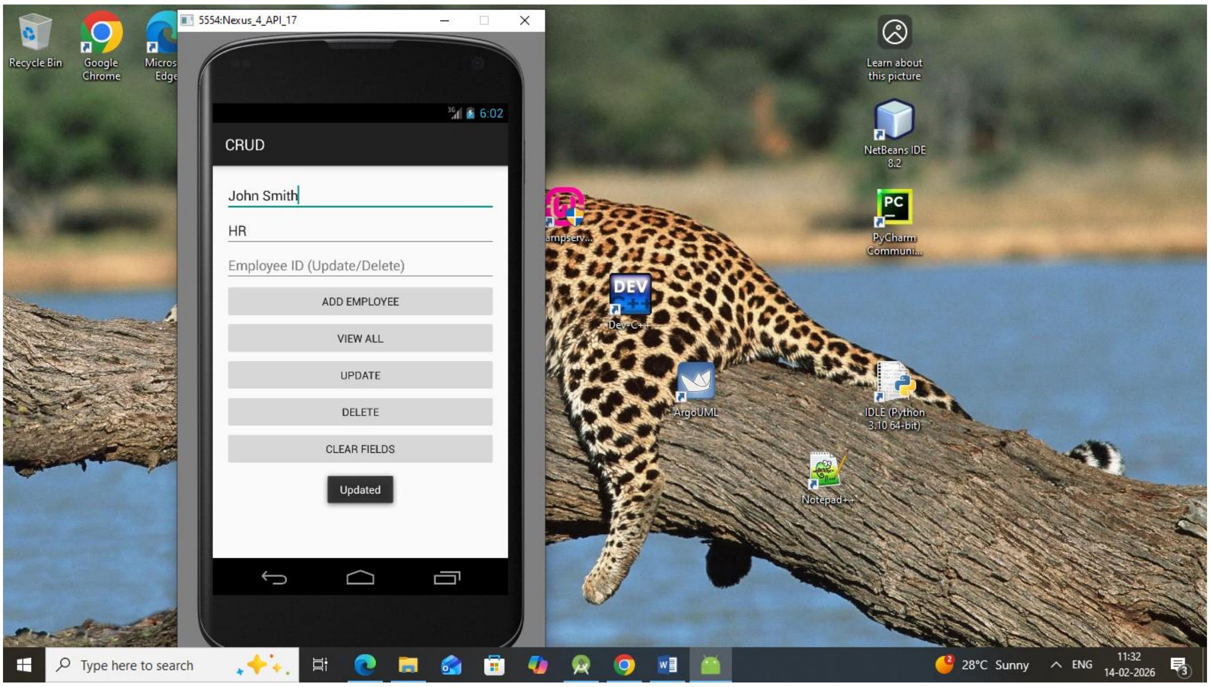
CRUD

John

HR

Employee ID (Update/Delete)

Success



5554:Nexus_4_API_17

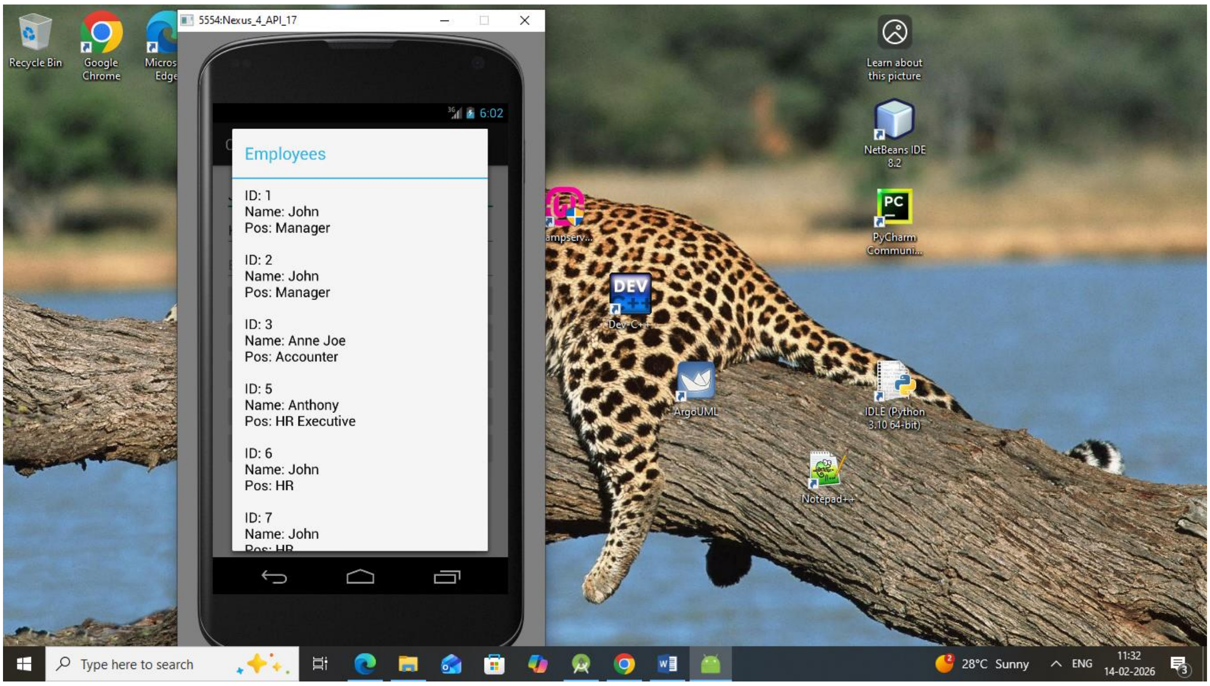
CRUD

John Smith

HR

Employee ID (Update/Delete)

Updated



5554:Nexus_4_API_17

Employees

ID: 1
Name: John
Pos: Manager

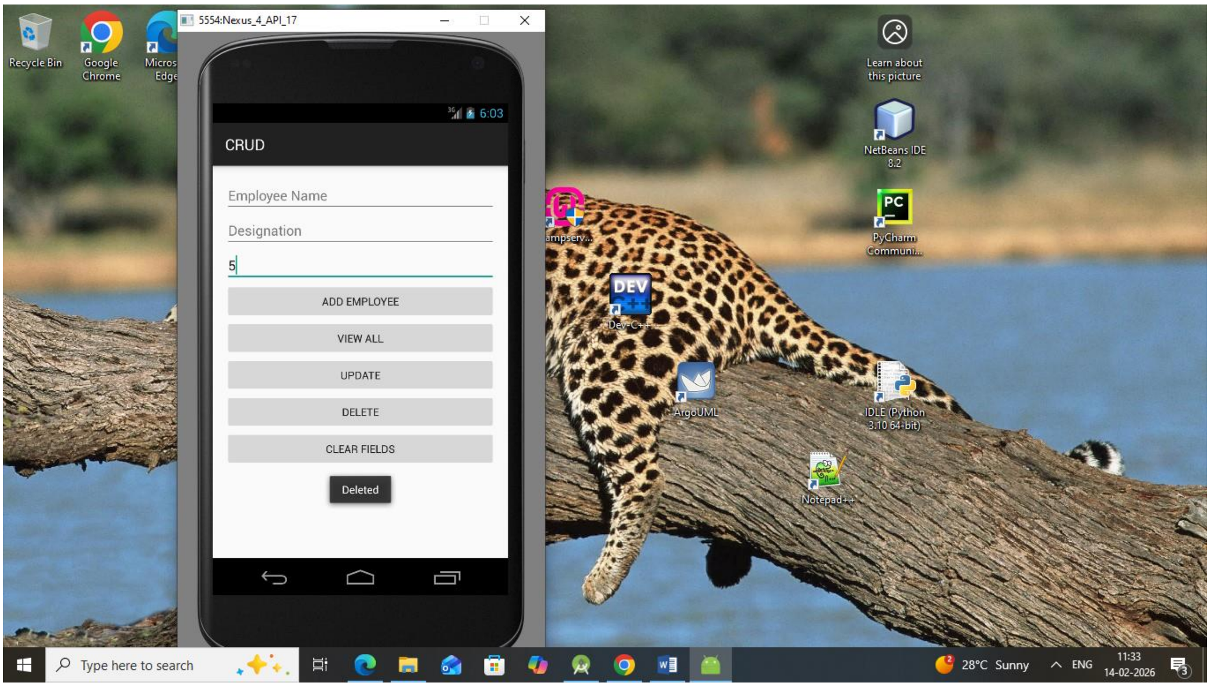
ID: 2
Name: John
Pos: Manager

ID: 3
Name: Anne Joe
Pos: Accounter

ID: 5
Name: Anthony
Pos: HR Executive

ID: 6
Name: John
Pos: HR

ID: 7
Name: John
Pos: HR



5554:Nexus_4_API_17

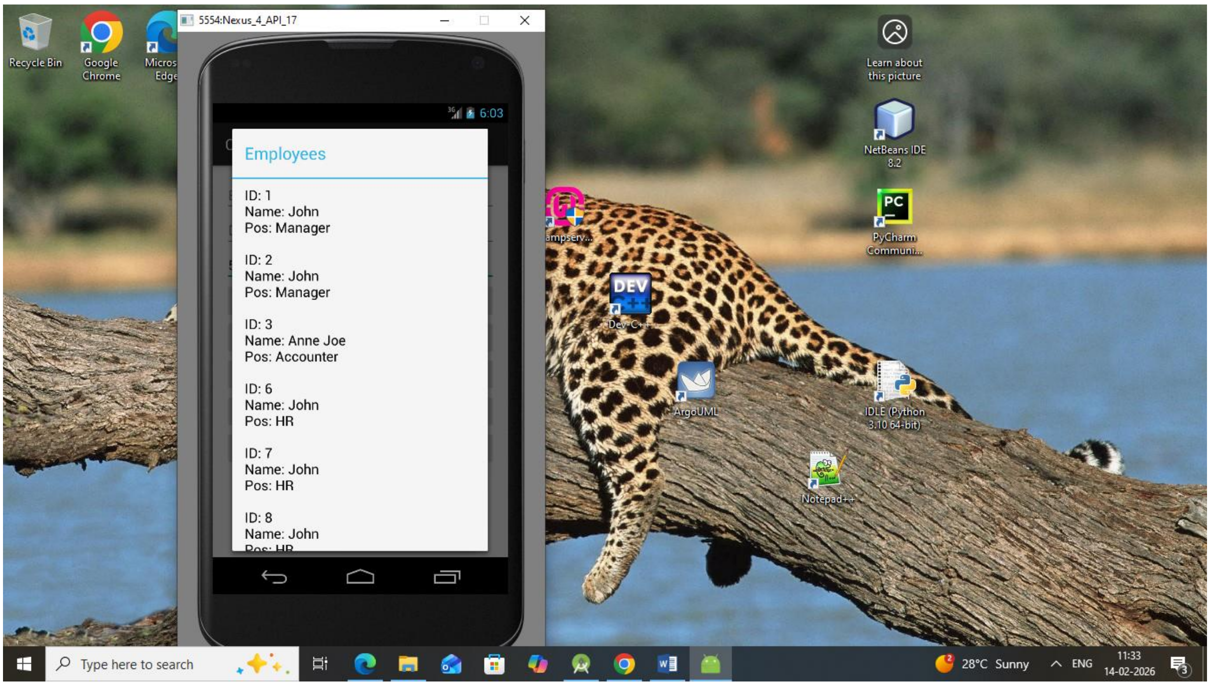
CRUD

Employee Name

Designation

5

Deleted



Employees

ID: 1
Name: John
Pos: Manager

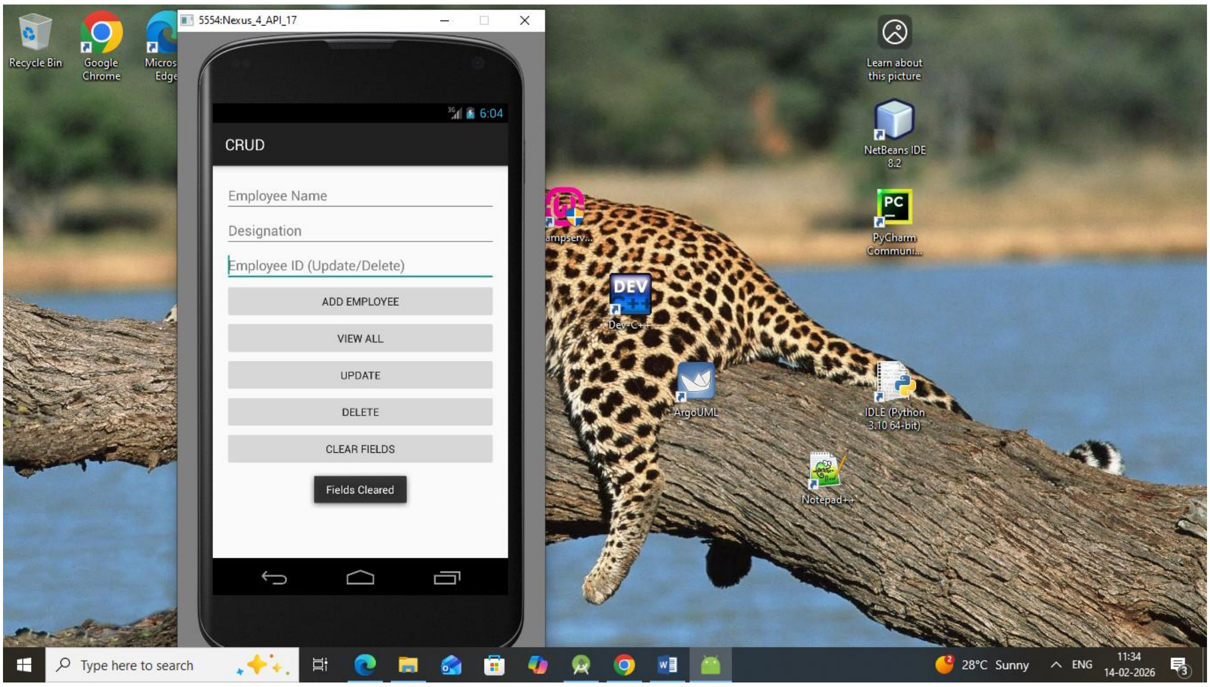
ID: 2
Name: John
Pos: Manager

ID: 3
Name: Anne Joe
Pos: Accounter

ID: 6
Name: John
Pos: HR

ID: 7
Name: John
Pos: HR

ID: 8
Name: John
Pos: HR



Fields Cleared

Result

Thus, an application to perform insert, update, delete, and display operations has been created using SQLite in Android.