1. Demonstrate different Layouts with different views in android LayoutsConstraintLayout,

RelativeLayout, TableLayout Views- Button, TextView, EditText, WebView, CheckBox, RadioButton, ToggleButton, ImageButton, RatingBar, ProgressBar, SeekBar, VideoView, DatePicker, CalendarView, Spinner

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

   <!-- ConstraintLayout -->
   <androidx.constraintlayout.widget.ConstraintLayout
android:id="@+id/constraintLayout"        android:layout_width="match_parent"
       android:layout_height="wrap_content">

       <!-- Add various views like Button, TextView, EditText, etc. -->

   </androidx.constraintlayout.widget.ConstraintLayout>

   <!-- RelativeLayout -->     <RelativeLayout
android:layout_width="match_parent"
       android:layout_height="wrap_content">

       <!-- Add various views -->

   </RelativeLayout>

   <!-- TableLayout -->     <TableLayout
android:layout_width="match_parent"
       android:layout_height="wrap_content">

       <!-- Add rows and views inside TableLayout -->

   </TableLayout>

   <!-- Other layouts can be added similarly -->

</RelativeLayout>
```

2. Write an android code to make phone call using Intent

```
// Add the CALL_PHONE permission in the manifest
// <uses-permission android:name="android.permission.CALL_PHONE" />

String phoneNumber = "tel:" + "123456789";
Intent dialIntent = new Intent(Intent.ACTION_CALL, Uri.parse(phoneNumber));

// Check for permission before making the call
if (ActivityCompat.checkSelfPermission(this, Manifest.permission.CALL_PHONE) ==
PackageManager.PERMISSION_GRANTED) {
   startActivity(dialIntent);
} else {
   // Request permission
   ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.CALL_PHONE},
1);
}
```

3. Write an android code to turn ON/OFF Bluetooth

```
BluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if (bluetoothAdapter.isEnabled()) {
// Bluetooth is enabled, turn it off
bluetoothAdapter.disable();
} else {
   // Bluetooth is disabled, turn it on
   bluetoothAdapter.enable();
}
```

4. Write an android code to turn ON /OFF the Wi-Fi

```
WifiManager = (WifiManager)
getApplicationContext().getSystemService(Context.WIFI_SERVICE);
if (wifiManager.isWifiEnabled()) {     // Wi-Fi is enabled, turn it off
   wifiManager.setWifiEnabled(false);
} else {
   // Wi-Fi is disabled, turn it on
   wifiManager.setWifiEnabled(true);
}
```

5. Design android application for login activity. Write android code to check login credentials with username = "mca" and password = "android". Display appropriate toast message to the user.

```
// Assuming you have a button and two EditTexts for username and password
Button loginButton = findViewById(R.id.loginButton);
EditText usernameEditText = findViewById(R.id.usernameEditText);
EditText passwordEditText = findViewById(R.id.passwordEditText);

loginButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String username = usernameEditText.getText().toString();
        String password = passwordEditText.getText().toString();

        if (username.equals("mca") && password.equals("android")) {
            // Correct credentials, show a success message
            Toast.makeText(getApplicationContext(), "Login Successful",
Toast.LENGTH_SHORT).show();
        } else {
            // Incorrect credentials, show an error message
            Toast.makeText(getApplicationContext(), "Invalid Credentials",
Toast.LENGTH_SHORT).show();
        }
    }
});
```

6. Create a fragment that has its own UI and enable your activities to communicate with fragments

```
import android.content.Context;
import android.os.Bundle; import
android.view.LayoutInflater; import
android.view.View; import
android.view.ViewGroup;
import androidx.fragment.app.Fragment;

public class ExampleFragment extends Fragment {
   // Define an interface for communication with the activity
public interface OnFragmentInteractionListener {        void
onButtonClicked(String message);
   }

   private OnFragmentInteractionListener mListener;
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
       // Inflate the layout for this fragment
       View = inflater.inflate(R.layout.fragment_example, container, false);

       // Assuming you have a button in the fragment layout
       View button = view.findViewById(R.id.fragmentButton);

       // Set a click listener for the button
       button.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View v) {
             // Communicate with the activity when the button is clicked
if (mListener != null) {
                mListener.onButtonClicked("Hello from Fragment!");
             }
          }
       });

       return view;
   }

   @Override
   public void onAttach(Context context) {
super.onAttach(context);
       // Ensure that the activity implements the interface
try {
          mListener = (OnFragmentInteractionListener) context;
       } catch (ClassCastException e) {
          throw new ClassCastException(context.toString() + " must implement
OnFragmentInteractionListener");
       }
   }
}
<!-- Replace this with your actual fragment layout -->
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
```

```
   android:layout_height="match_parent">

   <Button
android:id="@+id/fragmentButton"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Click me" />
   <!-- Add other UI elements as needed -->

</RelativeLayout>
public class MainActivity extends AppCompatActivity implements
ExampleFragment.OnFragmentInteractionListener {

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

       // Add the fragment to the activity
       getSupportFragmentManager().beginTransaction()
           .replace(R.id.fragmentContainer, new ExampleFragment())
           .commit();
   }

   // Implement the interface method to receive communication from the fragment
   @Override     public void onButtonClicked(String
message) {        // Handle the communication from
the fragment
       Toast.makeText(this, "Activity received: " + message, Toast.LENGTH_SHORT).show();
   }
}
<!-- Replace this with your actual activity layout -->
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
   android:layout_height="match_parent">

   <FrameLayout
android:id="@+id/fragmentContainer"
android:layout_width="match_parent"
       android:layout_height="match_parent" />

   <!-- Add other UI elements as needed -->

</RelativeLayout>
```

7. Demonstrate Array Adapter using List View to display list of fruits.

```
import android.os.Bundle; import
android.widget.ArrayAdapter; import
android.widget.ListView;
import androidx.appcompat.app.AppCompatActivity;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

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

       // Create a list of fruits
       ArrayList<String> fruitList = new ArrayList<>();
fruitList.add("Apple");        fruitList.add("Banana");
fruitList.add("Orange");
fruitList.add("Grapes");
       fruitList.add("Watermelon");

       // Create an ArrayAdapter and set it to the ListView
ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(
this,
              android.R.layout.simple_list_item_1,
fruitList
       );

       ListView = findViewById(R.id.listView);
       listView.setAdapter(arrayAdapter);
   }
}
```

8. Write an application to demonstrate Alert Dialog Box in android

```
import android.content.DialogInterface; import android.os.Bundle; import
android.view.View; import android.widget.Button; import
androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

    Button showDialogButton = findViewById(R.id.showDialogButton);

    // Set a click listener for the button
    showDialogButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        // Show the AlertDialog when the button is clicked
showAlertDialog();
      }
    });
  }

  // Method to show the AlertDialog
private void showAlertDialog() {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setTitle("Alert Dialog")
      .setMessage("This is a simple Alert Dialog.")
      .setPositiveButton("OK", new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int id) {           // Handle
positive button click (e.g., dismiss the dialog)
dialog.dismiss();
        }
      });
```

```
        // Create and show the AlertDialog
AlertDialog = builder.create();
alertDialog.show();
    }
}
```

9. Demonstrate Options Menu, Context Menu and Popup Menu in android

1. Options Menu:
res/menu/options_menu.xml:
Create a new XML file in the res/menu folder called options_menu.xml:

xml Copy
code

```
<!-- options_menu.xml -->
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/menu_item1"
android:title="Option 1"/>
    <item
        android:id="@+id/menu_item2"
        android:title="Option 2"/>
</menu>
```

MainActivity.java: java
Copy code

```
import
android.os.Bundle; import
android.view.Menu; import
android.view.MenuItem; import
android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

    // Create Options Menu
    @Override     public boolean
onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.options_menu, menu);
return true;
    }
```

```
  // Handle Options Menu item clicks
  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
switch (item.getItemId()) {            case R.id.menu_item1:
showToast("Option 1 selected");
        return true;
      case R.id.menu_item2:
showToast("Option 2 selected");
        return true;
default:
        return super.onOptionsItemSelected(item);
    }
  }

  private void showToast(String message) {
    Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
  }
}
```

2. Context Menu:
MainActivity.java: java

```
Copy code import
android.os.Bundle; import
android.view.ContextMenu; import
android.view.MenuItem; import
android.view.View; import
android.widget.TextView; import
android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

    TextView = findViewById(R.id.textView);
    registerForContextMenu(textView);
  }

  // Create Context Menu
@Override
```

```
   public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfo menuInfo) {
      getMenuInflater().inflate(R.menu.context_menu, menu);
   }

   // Handle Context Menu item clicks
   @Override
   public boolean onContextItemSelected(MenuItem item) {
      switch (item.getItemId()) {
case R.id.context_item1:
showToast("Context Option 1
selected");              return true;
case R.id.context_item2:
showToast("Context Option 2
selected");              return true;
default:
            return super.onContextItemSelected(item);
      }
   }

   private void showToast(String message) {
      Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
   }
}
```

res/menu/context_menu.xml:
Create a new XML file in the res/menu folder called context_menu.xml:

xml Copy
code

```
<!-- context_menu.xml -->
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item
      android:id="@+id/context_item1"
android:title="Context Option 1"/>
   <item
      android:id="@+id/context_item2"
      android:title="Context Option 2"/>
</menu>
```

3. Popup Menu:
MainActivity.java: java
Copy code import
android.os.Bundle; import
android.view.MenuItem; import
android.view.View; import

```
android.widget.PopupMenu; import
android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

       View = findViewById(R.id.textView);
       view.setOnClickListener(new View.OnClickListener() {
          @Override            public
void onClick(View v) {
             showPopupMenu(v);
          }
       });
   }

   // Show Popup Menu     private void
showPopupMenu(View view) {        PopupMenu
= new PopupMenu(this, view);
       popupMenu.inflate(R.menu.popup_menu);

       // Handle Popup Menu item clicks
popupMenu.setOnMenuItemClickListener(new
PopupMenu.OnMenuItemClickListener() {
          @Override
          public boolean onMenuItemClick(MenuItem item) {
             switch (item.getItemId()) {                 case
R.id.popup_item1:                    showToast("Popup
Option 1 selected");                    return true;
case R.id.popup_item2:
showToast("Popup Option 2 selected");
return true;                default:                    return
false;
             }
          }
       });

       // Show the Popup Menu
       popupMenu.show();
   }
```

```
  private void showToast(String message) {
    Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
  }
}
```

res/menu/popup_menu.xml:
Create a new XML file in the res/menu folder called popup_menu.xml:

xml Copy
code

```
<!-- popup_menu.xml -->
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item
    android:id="@+id/popup_item1"
android:title="Popup Option 1"/>
  <item
    android:id="@+id/popup_item2"
    android:title="Popup Option 2"/>
</menu>
```

In these examples:

Options Menu is created in the res/menu/options_menu.xml file. Context Menu is created in the res/menu/context_menu.xml file.
Popup Menu is created in the res/menu/popup_menu.xml file.

10. Write an application to produce Notification

1. Create a new Android Project:
Open Android Studio and create a new project with an Empty Activity.

2. Update the layout (activity_main.xml):
In the res/layout/activity_main.xml file, add a Button to trigger the notification.

xml Copy
code

```
<!-- activity_main.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:paddingLeft="16dp"
android:paddingTop="16dp"   android:paddingRight="16dp"
android:paddingBottom="16dp"
```

```
   tools:context=".MainActivity">

   <Button
android:id="@+id/notificationButton"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Show Notification"
android:layout_centerInParent="true" />
</RelativeLayout>
```

3. Update the MainActivity.java:
Add code to handle the button click and show a notification.

java
Copy code

```
// MainActivity.java import android.app.Notification;
import android.app.NotificationChannel; import
android.app.NotificationManager; import
android.content.Context; import android.os.Build;
import android.os.Bundle; import android.view.View;
import android.widget.Button; import
androidx.appcompat.app.AppCompatActivity; import
androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;

public class MainActivity extends AppCompatActivity {

   private final String CHANNEL_ID = "MyChannelID";

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

       createNotificationChannel();

       Button notificationButton = findViewById(R.id.notificationButton);
notificationButton.setOnClickListener(new View.OnClickListener() {
           @Override            public
void onClick(View v) {
               showNotification();
           }
       });
   }
   private void createNotificationChannel() {
       if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
           CharSequence name = "MyChannelName";
String description = "MyChannelDescription";
           int importance = NotificationManager.IMPORTANCE_DEFAULT;
           NotificationChannel channel = new NotificationChannel(CHANNEL_ID, name,
importance);
           channel.setDescription(description);

           NotificationManager = getSystemService(NotificationManager.class);
notificationManager.createNotificationChannel(channel);
       }
   }

   private void showNotification() {
       NotificationCompat.Builder builder = new NotificationCompat.Builder(this,
CHANNEL_ID)
               .setSmallIcon(R.drawable.ic_notification)
               .setContentTitle("My Notification")
               .setContentText("This is a sample notification.")
               .setPriority(NotificationCompat.PRIORITY_DEFAULT);

       NotificationManagerCompat notificationManager =
NotificationManagerCompat.from(this);
       notificationManager.notify(1, builder.build());
   }
}
```

4. Add Notification Icon:
Place an icon (e.g., ic_notification.png) in the res/drawable folder.

5. Run the Application:
Run your application on an Android device or emulator. Click the "Show Notification" button, and you should see a notification with the specified content.

11. Write an android application using SQLite to create table and perform CRUD operations (Example. COURSE table (ID, Name, Duration, Description), perform ADD, UPDATE, DELETE and READ operations)

1. Create a Database Helper Class:
Create a class named DatabaseHelper.java to manage your SQLite database.

java
Copy code

```
// DatabaseHelper.java import
android.content.Context; import
android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DatabaseHelper extends SQLiteOpenHelper {

   private static final String DATABASE_NAME = "courses.db";
private static final int DATABASE_VERSION = 1;

   public static final String TABLE_COURSE = "course";     public
static final String COLUMN_ID = "id";     public static final String
COLUMN_NAME = "name";     public static final String
COLUMN_DURATION = "duration";     public static final String
COLUMN_DESCRIPTION = "description";

   private static final String TABLE_CREATE =
         "CREATE TABLE " + TABLE_COURSE + " (" +
               COLUMN_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
               COLUMN_NAME + " TEXT, " +
               COLUMN_DURATION + " TEXT, " +
               COLUMN_DESCRIPTION + " TEXT);";

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

   @Override
   public void onCreate(SQLiteDatabase db) {
db.execSQL(TABLE_CREATE);
   }

   @Override
   public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
db.execSQL("DROP TABLE IF EXISTS " + TABLE_COURSE);        onCreate(db);
   }
}
```
2. Create a Data Model Class:
Create a class to represent the data model of the Course.

java
Copy code
```
// Course.java
public class Course {
   private long id;
private String name;
private String duration;
   private String description;

   // Constructor, getters, and setters
}
```

3. Create a Data Source Class:
Create a class named CourseDataSource.java to interact with the database and perform CRUD operations.

java
Copy code

```
// CourseDataSource.java import
android.content.ContentValues; import
android.content.Context; import
android.database.Cursor; import
android.database.SQLException; import
android.database.sqlite.SQLiteDatabase
;

import java.util.ArrayList;
import java.util.List;

public class CourseDataSource {

   private SQLiteDatabase database;
   private DatabaseHelper dbHelper;

   public CourseDataSource(Context context) {
      dbHelper = new DatabaseHelper(context);
   }

   public void open() throws SQLException {
      database = dbHelper.getWritableDatabase();
   }

   public void close() {
dbHelper.close();
   }

   // Add a new course
   public long addCourse(Course course) {        ContentValues values = new
ContentValues();        values.put(DatabaseHelper.COLUMN_NAME,
course.getName());        values.put(DatabaseHelper.COLUMN_DURATION,
course.getDuration());        values.put(DatabaseHelper.COLUMN_DESCRIPTION,
course.getDescription());

       return database.insert(DatabaseHelper.TABLE_COURSE, null, values);
   }

   // Update an existing course
   public int updateCourse(Course course) {        ContentValues values = new
ContentValues();        values.put(DatabaseHelper.COLUMN_NAME,
course.getName());        values.put(DatabaseHelper.COLUMN_DURATION,
course.getDuration());        values.put(DatabaseHelper.COLUMN_DESCRIPTION,
course.getDescription());

       return database.update(DatabaseHelper.TABLE_COURSE, values,
               DatabaseHelper.COLUMN_ID + " = ?", new String[]{String.valueOf(course.getId())});
   }

   // Delete a course
   public void deleteCourse(long courseId) {
       database.delete(DatabaseHelper.TABLE_COURSE,
               DatabaseHelper.COLUMN_ID + " = ?", new String[]{String.valueOf(courseId)});
   }

   // Get all courses     public List<Course>
getAllCourses() {        List<Course> courses =
new ArrayList<>();

       Cursor = database.query(DatabaseHelper.TABLE_COURSE, null,
null, null, null, null, null);

       if (cursor != null) {
cursor.moveToFirst();            while
(!cursor.isAfterLast()) {              Course =
cursorToCourse(cursor);
               courses.add(course);
               cursor.moveToNext();
           }
           cursor.close();
       }

       return courses;
   }

   private Course cursorToCourse(Cursor cursor) {
Course = new Course();
       course.setId(cursor.getLong(cursor.getColumnIndex(DatabaseHelper.COLUMN_ID)));

course.setName(cursor.getString(cursor.getColumnIndex(DatabaseHelper.COLUMN_NAME)
));

course.setDuration(cursor.getString(cursor.getColumnIndex(DatabaseHelper.COLUMN_DUR
ATION)));

course.setDescription(cursor.getString(cursor.getColumnIndex(DatabaseHelper.COLUMN_D
ESCRIPTION)));
       return course;
   }
}
```
4. Use the Database in MainActivity:
Update your MainActivity.java to interact with the database.

java
Copy code
```
// MainActivity.java import
android.os.Bundle; import
android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity; import
java.util.List;

public class MainActivity extends AppCompatActivity {

   private CourseDataSource dataSource;

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

       dataSource = new CourseDataSource(this);
       dataSource.open();

       // Example: Add a new course        Course newCourse = new
Course();        newCourse.setName("Android Development");
newCourse.setDuration("3 months");
newCourse.setDescription("Learn Android app development");
long newCourseId = dataSource.addCourse(newCourse);

       // Example: Update the course
Course courseToUpdate = new Course();
courseToUpdate.setId(newCourseId);
       courseToUpdate.setName("Updated Android Development");
       courseToUpdate.setDuration("4 months");
       courseToUpdate.setDescription("Learn advanced Android app development");
dataSource.updateCourse(courseToUpdate);

       // Example: Get all courses
       List<Course> courses = dataSource.getAllCourses();
       for (Course : courses) {
           displayCourse(course);
       }

       // Example: Delete the course
dataSource.deleteCourse(newCourseId);

       // Close the database connection
       dataSource.close();
   }

   private void displayCourse(Course course) {
       Toast.makeText(this, "Course: " + course.getName() +
               ", Duration: " + course.getDuration() +
               ", Description: " + course.getDescription(), Toast.LENGTH_SHORT).show();
   }}
```

12. Create an Android app, powered by Firebase Realtime database that supports: Adding Data to Firebase Realtime database, Retrieving Data from Firebase and Deleting data from firebase data.

```
// MyData.java
public class MyData {

    private String key; // Firebase Realtime Database key
private String data;

    public MyData() {
        // Default constructor required for Firebase
    }
    public MyData(String data) {
        this.data = data;
    }

    public String getKey() {
        return key;
    }

    public void setKey(String key) {
        this.key = key;
    }

    public String getData() {
        return data;
    }

    public void setData(String data) {
        this.data = data;
    }
}
// MainActivity.java import
android.os.Bundle; import
android.view.View; import
android.widget.Button;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private FirebaseHelper;

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

        firebaseHelper = new FirebaseHelper();

        Button addButton = findViewById(R.id.addButton);
        Button retrieveButton = findViewById(R.id.retrieveButton);
        Button deleteButton = findViewById(R.id.deleteButton);

        addButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                addData();
            }
        });

        retrieveButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                retrieveData();
            }
        });

        deleteButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                deleteData();
            }
        });
    }

    private void addData() {
        MyData = new MyData("Hello, Firebase!");
        firebaseHelper.addData(myData);
    }

    private void retrieveData() {
        firebaseHelper.retrieveData(new FirebaseHelper.DataChangeListener() {
            @Override
            public void onDataAdded(MyData myData) {
                // Handle data added to the list
            }

            @Override
            public void onDataRemoved(MyData myData) {
                // Handle data removed from the list
            }
        });
    }

    private void deleteData() {
        // Retrieve the data you want to delete and pass it to the deleteData method
        // For example, you can retrieve data in onDataAdded and delete it in onDataRemoved
    }
}
```

```
<!-- activity_main.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:paddingLeft="16dp"
android:paddingTop="16dp"     android:paddingRight="16dp"
android:paddingBottom="16dp"
   tools:context=".MainActivity">

   <Button
      android:id="@+id/addButton"
android:layout_width="wrap_content"
android:layout_height="wrap_content"        android:text="Add
Data"        android:layout_centerHorizontal="true"
      android:layout_marginBottom="16dp" />

   <Button
android:id="@+id/retrieveButton"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Retrieve Data"
android:layout_below="@id/addButton"
android:layout_centerHorizontal="true"
android:layout_marginTop="16dp"
android:layout_marginBottom="16dp" />
   <Button
android:id="@+id/deleteButton"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
      android:text="Delete Data"
android:layout_below="@id/retrieveButton"
android:layout_centerHorizontal="true"
android:layout_marginTop="16dp" />
  </RelativeLayout>
```

13. Demonstrate WebView to display the web pages in an android application

```
<!-- activity_main.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"
```

```
    tools:context=".MainActivity">

    <WebView
        android:id="@+id/webView"
android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</RelativeLayout> // MainActivity.java
import android.os.Bundle; import
android.webkit.WebView; import
android.webkit.WebViewClient;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private WebView;

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

        webView = findViewById(R.id.webView);

        // Enable JavaScript (optional, depending on the web content)
webView.getSettings().setJavaScriptEnabled(true);

        // Set a WebViewClient to handle redirects and other events within the WebView
webView.setWebViewClient(new WebViewClient());

        // Load a web page
        loadWebPage("https://www.example.com");
    }

    private void loadWebPage(String url) {
// Load the specified URL in the WebView
webView.loadUrl(url);
    }

    @Override
    public void onBackPressed() {
        // Go back in the WebView's history on back button press
if (webView.canGoBack()) {
            webView.goBack();
        } else {
            super.onBackPressed();
        }
    }
}

<!-- AndroidManifest.xml -->
<uses-permission android:name="android.permission.INTERNET" />
```

14. Write an android app to write JSON data into a file and read JSON data from created file.

```
// MainActivity.java import
android.content.Context; import
android.os.Bundle; import
android.view.View; import
android.widget.Button; import
android.widget.Toast; import
org.json.JSONException; import
org.json.JSONObject; import
java.io.BufferedReader; import
java.io.FileInputStream; import
java.io.FileOutputStream; import
java.io.IOException;
import java.io.InputStreamReader;

public class MainActivity extends AppCompatActivity {

    private static final String FILE_NAME = "json_data.json";

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

        Button writeReadButton = findViewById(R.id.writeReadButton);
writeReadButton.setOnClickListener(new View.OnClickListener() {
            @Override            public
void onClick(View v) {                //
Example JSON data
JSONObject jsonData =
createSampleJSONData();
```

```
                // Write JSON data to a file
                writeJSONToFile(jsonData);

                // Read JSON data from the file
                JSONObject readData = readJSONFromFile();

                // Display the read JSON data
if (readData != null) {
                    displayJSONData(readData);
                } else {
                    showToast("Error reading JSON data from file.");
                }
            }
        });
    }

    private JSONObject createSampleJSONData() {
        // Create a sample JSON object
JSONObject = new JSONObject();         try
{
            jsonObject.put("name", "John Doe");
jsonObject.put("age", 25);
            jsonObject.put("city", "Example City");
} catch (JSONException e) {
e.printStackTrace();
        }
        return jsonObject;
    }

    private void writeJSONToFile(JSONObject jsonData) {
        try (FileOutputStream = openFileOutput(FILE_NAME, Context.MODE_PRIVATE)) {
            fileOutputStream.write(jsonData.toString().getBytes());
showToast("JSON data written to file: " + FILE_NAME);         }
catch (IOException e) {             e.printStackTrace();
showToast("Error writing JSON data to file.");
        }
    }

    private JSONObject readJSONFromFile() {
StringBuilder = new StringBuilder();
        try (FileInputStream = openFileInput(FILE_NAME);
             InputStreamReader = new InputStreamReader(fileInputStream);
             BufferedReader = new BufferedReader(inputStreamReader)) {
            String line;
```

```
          while ((line = bufferedReader.readLine()) != null) {
stringBuilder.append(line);
          }
          return new JSONObject(stringBuilder.toString());
} catch (IOException | JSONException e) {
e.printStackTrace();
          return null;
      }
   }

   private void displayJSONData(JSONObject jsonData) {
      String displayText = "Read JSON data from file:\n" + jsonData.toString();
showToast(displayText);
   }

   private void showToast(String message) {
      Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
   }
}
```

15. Write an application to display a PDF as an image in React app using URL

```
// PDFViewer.js import
React from 'react';
import { Document, Page } from 'react-pdf';

const PDFViewer = ({ pdfUrl }) => {   const [numPages,
setNumPages] = React.useState(null);   const [pageNumber,
setPageNumber] = React.useState(1);

  const onDocumentLoadSuccess = ({ numPages }) => {
setNumPages(numPages);
  };

  return (
<div>
     <Document file={pdfUrl} onLoadSuccess={onDocumentLoadSuccess}>
       <Page pageNumber={pageNumber} />
     </Document>
     <p>
       Page {pageNumber} of {numPages}
     </p>
   </div>
  );
};

export default PDFViewer;
// App.js
import React from 'react';
import PDFViewer from './PDFViewer';

const App = () => {
  const pdfUrl = 'your_pdf_url_here';

  return (
    <div>
      <h1>PDF Viewer App</h1>
      <PDFViewer pdfUrl={pdfUrl} />
    </div>
  );
};

export default App; npm
start
```

16. Develop simple flutter application to open a browser using Android SDK

```
dependencies:   flutter:
    sdk: flutter
url_launcher: ^6.0.6 flutter
pub get
import 'package:flutter/material.dart'; import
'package:url_launcher/url_launcher.dart';

void main() {
runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
return MaterialApp(
      home: MyHomePage(),
    );
  }
}

class MyHomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
appBar: AppBar(        title:
Text('Open Browser
Example'),
      ),
      body: Center(
child: ElevatedButton(
onPressed: () {
_openBrowser();
          },
          child: Text('Open Browser'),
        ),
      ),
    );
  }

  void _openBrowser() async {
    const url = 'https://www.example.com'; // Replace with your desired URL
if (await canLaunch(url)) {       await launch(url);
    } else {
      throw 'Could not launch $url';
    }
  }
}
flutter run
```

KR an AI, ML, DL

1. Find the correlation matrix.

```
import numpy as np
import pandas as pd

# Load the Iris dataset
from sklearn.datasets import load_iris
iris = load_iris()
df = pd.DataFrame(data=np.c_[iris['data'], iris['target']],
columns=iris['feature_names'] + ['target'])

# Calculate the correlation matrix
correlation_matrix = df.corr()

# Display the correlation matrix
print(correlation_matrix)
```

2. Plot the correlation plot on dataset and visualize giving an overview of relationships among data on iris data.

```
import seaborn as sns import
matplotlib.pyplot as plt

# Plot the correlation matrix
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Plot of Iris Data') plt.show()
```

3. **Analysis of covariance: variance (ANOVA), if data have categorical variables on iris data.**

```
import statsmodels.api as sm
from statsmodels.formula.api import ols

# Fit the ANOVA model
model = ols('sepal_length ~ C(target)', data=df).fit()
# Print the ANOVA table print(sm.stats.anova_lm(model))
```

4. **Apply linear regression Model techniques to predict the data on any dataset**.

```
from sklearn.model_selection import train_test_split from
sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Split the data into features and target
X = df.drop('target', axis=1)
y = df['target']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Create and fit the linear regression model linear_model
= LinearRegression()
linear_model.fit(X_train, y_train)

# Make predictions
y_pred = linear_model.predict(X_test)

# Evaluate the model mse =
mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
```

5. **Apply logical regression Model techniques to predict the data on any dataset.**

```
 from sklearn.linear_model import LogisticRegression from
sklearn.metrics import accuracy_score, confusion_matrix

# Create and fit the logistic regression model logistic_model
= LogisticRegression()
logistic_model.fit(X_train, y_train)

# Make predictions
y_pred_logistic = logistic_model.predict(X_test)

# Evaluate the model accuracy = accuracy_score(y_test,
y_pred_logistic) conf_matrix = confusion_matrix(y_test,
y_pred_logistic)
print(f'Accuracy: {accuracy}') print(f'Confusion
Matrix:\n{conf_matrix}')
```

6. Clustering algorithms for unsupervised classification.

```
 from sklearn.cluster import KMeans

# Load only relevant features for clustering
X_clustering = df[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']]


# Fit K-Means clustering model

kmeans = KMeans(n_clusters=3, random_state=42) df['cluster']
= kmeans.fit_predict(X_clustering)

# Visualize the clusters

sns.scatterplot(x='sepal_length', y='sepal_width', hue='cluster', data=df, palette='viridis')
plt.title('Clustering of Iris Data')

plt.show()
```

7. Association algorithms for supervised classification on any dataset

```
import numpy as np import pandas as pd from sklearn.model_selection import
train_test_split from sklearn.ensemble import RandomForestClassifier from sklearn.metrics
import accuracy_score, confusion_matrix
```

```
# Load the Iris dataset from
sklearn.datasets import load_iris
iris = load_iris() df = pd.DataFrame(data=np.c_[iris['data'], iris['target']],
columns=iris['feature_names'] + ['target'])


# Split the data into features and
target X = df.drop('target', axis=1) y
= df['target']


# Split the data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Create and fit the Random Forest Classifier rf_classifier =
RandomForestClassifier(n_estimators=100, random_state=42)
rf_classifier.fit(X_train, y_train)

# Make predictions y_pred =
rf_classifier.predict(X_test)


# Evaluate the model accuracy =
accuracy_score(y_test, y_pred) conf_matrix
= confusion_matrix(y_test, y_pred)
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')
```

8. Developing and implementing Decision Tree model on the dataset

```
from sklearn.tree import DecisionTreeClassifier


# Assuming 'X_train', 'X_test', 'y_train', 'y_test' are already defined
dt_classifier = DecisionTreeClassifier(random_state=42)
dt_classifier.fit(X_train, y_train)


y_pred_dt = dt_classifier.predict(X_test)
```

9. Bayesian classification on any dataset.

```
from sklearn.naive_bayes import GaussianNB
```

# Assuming 'X_train', 'X_test', 'y_train', 'y_test' are already defined nb_classifier = GaussianNB() nb_classifier.fit(X_train, y_train)

y_pred_nb = nb_classifier.predict(X_test)

10. SVM classification on any datase

from sklearn.svm import SVC

# Assuming 'X_train', 'X_test', 'y_train', 'y_test' are already defined svm_classifier = SVC(kernel='linear', C=1, random_state=42) svm_classifier.fit(X_train, y_train)

y_pred_svm = svm_classifier.predict(X_test)

11. Text Mining algorithms on unstructured dataset

import nltk from nltk.corpus import stopwords from nltk.tokenize import word_tokenize from nltk.probability import FreqDist import matplotlib.pyplot as plt

# Sample unstructured text data
text_data = """
Text mining, also known as text data mining, is the process of deriving high-quality information from unstructured text.

This involves the use of natural language processing (NLP) and machine learning techniques to analyze and understand the patterns, trends, and insights hidden within large volumes of textual data. """

# Download NLTK resources (if not already downloaded) nltk.download('punkt') nltk.download('stopwords')

# Tokenization tokens = word_tokenize(text_data.lower()) # Convert to lowercase for consistency

# Remove stop words stop_words = set(stopwords.words('english')) filtered_tokens
= [word for word in tokens if word.isalnum() and word not in stop_words]

# Frequency distribution fdist
= FreqDist(filtered_tokens) #
Plotting the frequency
distribution fdist.plot(20,
cumulative=False) plt.show()

12. Plot the cluster data using python visualizations.

import seaborn as sns
import matplotlib.pyplot as
plt

sns.scatterplot(x='feature1', y='feature2', hue='cluster',
data=df) plt.title('Clustered Data Visualization') plt.show()

13. Creating & Visualizing Neural Network for the given data. (Use python)

import tensorflow as tf from
tensorflow import keras from
tensorflow.keras import layers

# Assuming 'X_train', 'X_test', 'y_train', 'y_test' are already defined model =
keras.Sequential([layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
layers.Dense(3, activation='softmax') # Adjust the output units based on your
classification task
])

model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
metrics=['accuracy'])

```
history = model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test))


# Visualize training history if needed
plt.plot(history.history['accuracy'], label='accuracy')
plt.plot(history.history['val_accuracy'],
label='val_accuracy') plt.xlabel('Epoch')
plt.ylabel('Accuracy') plt.legend(loc='lower right')
plt.show()
```

14. Recognize optical character using ANN.

```
import tensorflow as tf from
tensorflow.keras import layers, models
from tensorflow.keras.datasets import
mnist import matplotlib.pyplot as plt


# Load and preprocess the MNIST dataset

(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
train_images = train_images.reshape((60000, 28, 28, 1)).astype('float32') / 255
test_images = test_images.reshape((10000, 28, 28, 1)).astype('float32') / 255


# Convert labels to one-hot encoding train_labels
= tf.keras.utils.to_categorical(train_labels)
test_labels =
tf.keras.utils.to_categorical(test_labels)


# Build a simple ANN model model =
models.Sequential()
model.add(layers.Flatten(input_shape=(28, 28,
1))) model.add(layers.Dense(128,
activation='relu')) model.add(layers.Dense(10,
activation='softmax'))


# Compile the model
model.compile(optimizer='adam',

          loss='categorical_crossentropy',
metrics=['accuracy'])


# Train the model history = model.fit(train_images, train_labels, epochs=10,
batch_size=64, validation_split=0.2)
```

```
# Evaluate the model test_loss, test_acc =
model.evaluate(test_images, test_labels) print(f'Test
Accuracy: {test_acc}')
```

```
# Visualize the training history
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation
Accuracy') plt.xlabel('Epochs') plt.ylabel('Accuracy')
plt.legend() plt.show()
```

15. Write a program to implement CNN

```
import tensorflow as tf from tensorflow.keras
import layers, models, datasets import
matplotlib.pyplot as plt
```

```
# Load and preprocess the CIFAR-10 dataset

(train_images, train_labels), (test_images, test_labels) =
datasets.cifar10.load_data() train_images = train_images.astype('float32') /
255 test_images = test_images.astype('float32') / 255
```

```
# Convert labels to one-hot encoding train_labels
= tf.keras.utils.to_categorical(train_labels)
test_labels =
tf.keras.utils.to_categorical(test_labels)
```

```
# Build a simple CNN model model = models.Sequential() model.add(layers.Conv2D(32, (3,
3), activation='relu', input_shape=(32, 32, 3))) model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu')) model.add(layers.MaxPooling2D((2,
2))) model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.Flatten()) model.add(layers.Dense(64,
activation='relu')) model.add(layers.Dense(10, activation='softmax'))
```

```
# Compile the model
model.compile(optimizer='adam',
          loss='categorical_crossentropy',
metrics=['accuracy'])
```

```
# Train the model history = model.fit(train_images, train_labels,
epochs=10, batch_size=
```

16. Write a program to implement RNN

```
import tensorflow as tf from tensorflow.keras import
layers, datasets, preprocessing


# Load and preprocess the IMDB dataset max_features = 10000  #
Consider only the top 10,000 most frequent words maxlen = 500  #
Cut reviews after 500 words
(x_train, y_train), (x_test, y_test) =
datasets.imdb.load_data(num_words=max_features) x_train =
preprocessing.sequence.pad_sequences(x_train, maxlen=maxlen) x_test =
preprocessing.sequence.pad_sequences(x_test, maxlen=maxlen)


# Build a simple RNN model model =
tf.keras.Sequential()
model.add(layers.Embedding(max_features,
32)) model.add(layers.SimpleRNN(32))
model.add(layers.Dense(1,
activation='sigmoid'))


# Compile the model
model.compile(optimizer='adam',
            loss='binary_crossentropy',
metrics=['accuracy'])


# Train the model history = model.fit(x_train, y_train, epochs=5,
batch_size=64, validation_split=0.2)


# Evaluate the model test_loss, test_acc =
model.evaluate(x_test, y_test) print(f'Test
Accuracy: {test_acc}')


# Visualize the training history import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation
Accuracy') plt.xlabel('Epochs') plt.ylabel('Accuracy')
plt.legend() plt.show()
```

17. Write a program to implement GAN

```
import tensorflow as tf from
tensorflow.keras import layers, models
import matplotlib.pyplot as plt import
numpy as np


# Load and preprocess the MNIST dataset

(train_images, _), (_, _) = tf.keras.datasets.mnist.load_data()
train_images = train_images.astype('float32') / 255.0


# Define the generator model generator =
models.Sequential() generator.add(layers.Dense(128,
input_dim=100, activation='relu'))
generator.add(layers.Dense(784, activation='sigmoid'))
generator.add(layers.Reshape((28, 28, 1)))


# Define the discriminator model discriminator =
models.Sequential()
discriminator.add(layers.Flatten(input_shape=(28,
28, 1))) discriminator.add(layers.Dense(128,
activation='relu')) discriminator.add(layers.Dense(1,
activation='sigmoid'))


# Combine the generator and discriminator into a GAN model
discriminator.trainable = False  # Freeze the discriminator during GAN
training gan = models.Sequential() gan.add(generator)
gan.add(discriminator)


# Compile the discriminator discriminator.compile(optimizer='adam',
loss='binary_crossentropy', metrics=['accuracy'])


# Compile the GAN gan.compile(optimizer='adam',
loss='binary_crossentropy')
```

```
# Training the GAN def
train_gan(epochs=100, batch_size=64):
for epoch in range(epochs):
        # Generate random noise as input to the generator
noise = np.random.normal(0, 1, size=[batch_size, 100])
generated_images = generator.predict(noise)


        # Select a random batch of real images from the MNIST
dataset         idx = np.random.randint(0, train_images.shape[0],
batch_size)         real_images = train_images[idx]


        # Concatenate real and generated images to create the training set for the
discriminator         X = np.concatenate([real_images, generated_images])         y_dis =
np.zeros(2 * batch_size)         y_dis[:batch_size] = 0.9  # Label smoothing for real
images


        # Train the discriminator         d_loss =
discriminator.train_on_batch(X, y_dis)


        # Train the generator         noise =
np.random.normal(0, 1, size=[batch_size, 100])
y_gen = np.ones(batch_size)         g_loss =
gan.train_on_batch(noise, y_gen)

        # Print progress and save generated images at specific
intervals         if epoch % 10 == 0:
            print(f"Epoch {epoch}, D Loss: {d_loss[0]}, G Loss: {g_loss}")


# Train the GAN
train_gan(epochs=100,
batch_size=64)


# Generate and visualize some images
noise = np.random.normal(0, 1, size=[16,
100]) generated_images =
generator.predict(noise)


# Plot the generated images
```

```
fig, axes = plt.subplots(4, 4, figsize=(4, 4))
for i, ax in enumerate(axes.flatten()):
    ax.imshow(generated_images[i, :, :, 0],
cmap='gray')




    ax.axis('off')
plt.show()
```

18. Web scraping experiments (by using tools)

```
import requests from bs4

import BeautifulSoup


# Specify the URL you want to scrape url

= "https://example.com"


# Send an HTTP request and get the HTML content

response = requests.get(url)

html_content = response.text


# Parse the HTML content using BeautifulSoup soup

= BeautifulSoup(html_content, 'html.parser')
```

```
# Extract information from the parsed HTML

title = soup.title.text paragraphs =

soup.find_all('p')


# Print the results

print(f"Title: {title}")

print("Paragraphs:") for

p in paragraphs:

    print(p.text)


from selenium import webdriver from

selenium.webdriver.common.keys import Keys


# Specify the URL you want to scrape url

= "https://example.com"


# Set up the Selenium WebDriver (make sure you have the appropriate driver installed)

driver = webdriver.Chrome()  # You may need to download the ChromeDriver or use another
browser driver


# Open the webpage
driver.get(url)


# Extract information using Selenium

title = driver.title
```

```
element = driver.find_element_by_tag_name('p')

paragraph_text = element.text


# Print the results print(f"Title:

{title}") print(f"Paragraph:

{paragraph_text}")


# Close the browser window driver.quit()
```