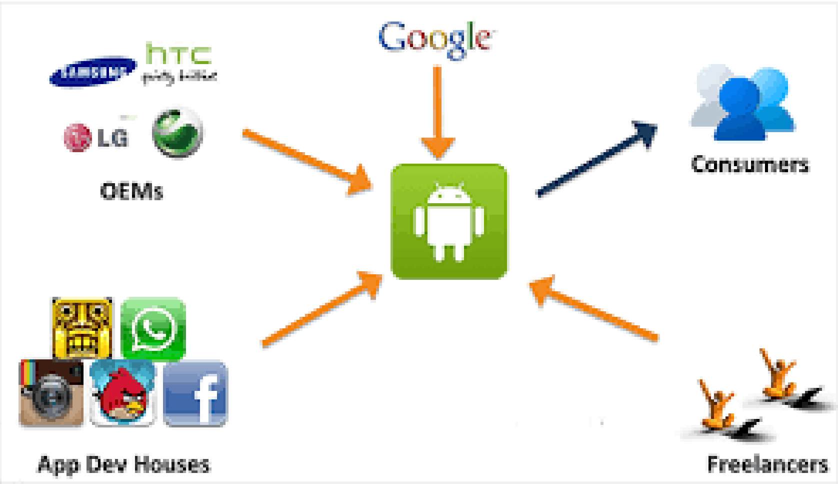
MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

SUMMER – 2023 EXAMINATION
Model Answer – Only for the Use of RAC Assessors

Subject Name: **Mobile Application Development** **Subject Code:** 22617

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.
8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1		**Attempt any FIVE of the following:**	**10 M**
	a)	**State Android ECO System.**	**2 M**
	Ans	**Android Ecosystem**  Google SAMSUNG htc LG OEMs Consumers App Dev Houses Freelancers • **Ecosystem** in Market terminology refers to the inter-dependence between demand and supply.	Any 4 points 2 M

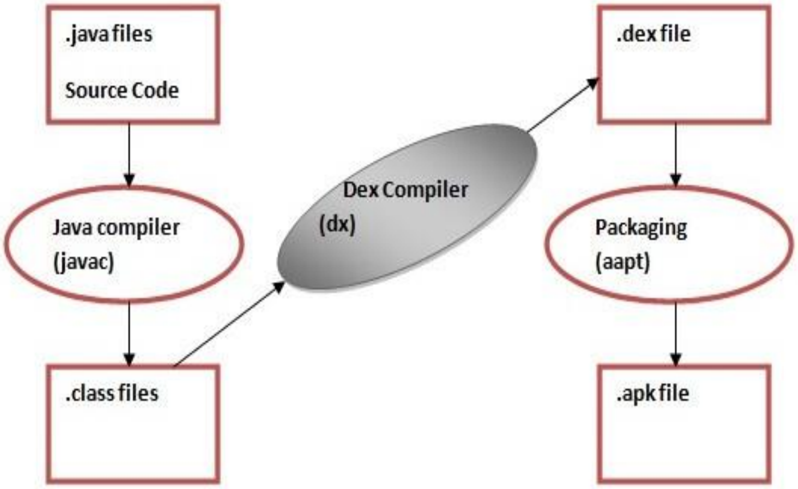
		• In the **Android ecosystem** this translates to inter-dependence between users, developers, and equipment makers. One cannot exist without the other: ▪ **Google** develops android ▪ **Users** buy devices and applications ▪ **Original Equipment makers** sell devices, sometimes bundled with applications ▪ **Developers** buy devices, then make and sell applications ▪ **Freelance Android Developer** developers have the skills to contribute to the ecosystem for android development , they are who creates their own applications and published them on googles play store.	
	b)	**List various tools for android application development**	**2 M**
	Ans	• Android Studio • ADB (Android Debug Bridge) • AVD Manager • Eclipse • Fabric • FlowUp • GameMaker: Studio • Genymotion • Gradle IntelliJ IDEA	Any 4 tools 2 M
	c)	**List various layouts used in android UI design**.	**2 M**
	Ans	• Linear Layout • Absolute Layout • Frame Layout • Table Layout • Relative Layout	Any 4 layouts 2 M
	d)	**Name any four attributes of Edit Text control**.	**2 M**
	Ans	android:id android: gravity android: text android: hint android: textColor android: textSize android: textStyle android: background	Any 4 attributes 2 M
	e)	**State the use of fragments in android App development.**	**2 M**
	Ans	Android Fragment is the part of activity, it is also known as sub-activity. There can be more than one fragment in an activity. Fragments represent multiple screen inside one activity.	Explanation 2 M

		We can create Fragments by extending Fragment class or by inserting a Fragment into our Activity layout by declaring the Fragment in the activity's layout file, as a <fragment> element. Fragments were added in Honeycomb version of Android i.e API version 11. We can add, replace or remove Fragment's in an Activity while the activity is running. Fragment can be used in multiple activities. We can also combine multiple Fragments in a single activity to build a multi-plane UI. We can only show a single Activity on the screen at one given point of time so we were not able to divide the screen and control different parts separately. With the help of Fragment's we can divide the screens in different parts and controls different parts separately	
	f)	**Define SMS service in android application development.**	**2 M**

	Ans	**SMS**
		• In Android, you can use SmsManager API or devices Built-in SMS application to send SMS's
		• Android SMS is stored in PDU (protocol description unit) format
		• SmsManager class takes care of sending the SMS message.
		• We just need to get an instance of it and send the SMS message.
		• We need to add permission to SEND_SMS in the Android manifest file.
		`SmsManager smsManager = SmsManager.getDefault();`
		`smsManager.sendTextMessage("phoneNo", null, "sms message", null, null);`
			Any
			4 points
			2 M

	g)	**List different types of sensors used in android.**	**2 M**
	Ans	The android platform supports three broad categories of sensors. • **Motion Sensors** These are used to measure acceleration forces and rotational forces along with three axes. • **Environmental sensors** These are used to measure the environmental changes such as temperature, humidity etc. • **Position sensors** These are used to measure the physical position of device.	2 M for List

2.		**Attempt any <u>THREE</u> of the following:**	**12 M**
	a)	**Describe android and importance of OHA.**	**4 M**
	Ans	**Android** Android is an open source and Linux-based Operating System .It is designed primarily for touch screens mobile devices such as smartphones and tablet computers. Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android. Android was developed by the *Open Handset Alliance*, led by Google, and other companies. **OHA** • The Open Handset Alliance (OHA) is a business alliance that was created for the purpose of developing open mobile device standards. • The OHA has approximately 80 member companies, including HTC, Dell, Intel, Motorola, Qualcomm and Google. **Importance of OHA** • **Lower overall handset costs:** Opens up resources, which facilitates the focus on creating innovative applications, solutions and services. • **Developer-friendly environment:** In the open-source community, developers share notes to expedite application development. • **Post-development:** Provides an ideal channel for application marketing and distribution.	Explain android 2 M Importance 2 M
	b)	**Explain Dalvik Virtual Machine and state its importance.**	**4 M**
	Ans	The **Dalvik Virtual Machine (DVM)** is an android virtual machine optimized for mobile devices. Dalvik VM is also a virtual machine that is highly optimized for mobile devices. Thus, it provides all the three things, that are memory management, high performance as well as battery life. It is strictly developed for Android mobile phones.	Explain 2 M Importance 2 M

		 .java files Source Code Java compiler (javac) .class files Dex Compiler (dx) .dex file Packaging (aapt) .apk file The Dex compiler converts the class files into the .dex file that run on the Dalvik VM. Multiple class files are converted into one dex file. The **javac tool** compiles the java source file into the class file. The **dx tool** takes all the class files of your application and generates a single .dex file. It is a platform-specific tool. The Android Assets Packaging Tool (aapt) handles the packaging process.	
	c)	**Describe the process of getting the map API key.**	**4 M**
	Ans	**Creating API keys** The API key is a unique identifier that authenticates requests associated with your project for usage and billing purposes. You must have at least one API key associated with your project. 1. Browse the site on your browser. https://console. developers. google.com/project 2. Login with your google account. 3. Create a new project by clicking on Create Project option. 4. Add your project name and organization name in the fields present on the screen. 5. Now click on APIs and Services. 6. Enable APIs and services. 7. Select Google maps Android API 8. To create an API key for Maps click on Create credentials option and then select the API key option Click on the API key option to generate your API key. After clicking on this option your API key will be generated	Correct Steps 4 marks
	d)	**Explain multimedia framework in android.**	**4 M**

Ans **Multimedia Framework**

Proper Explanation 4 marks

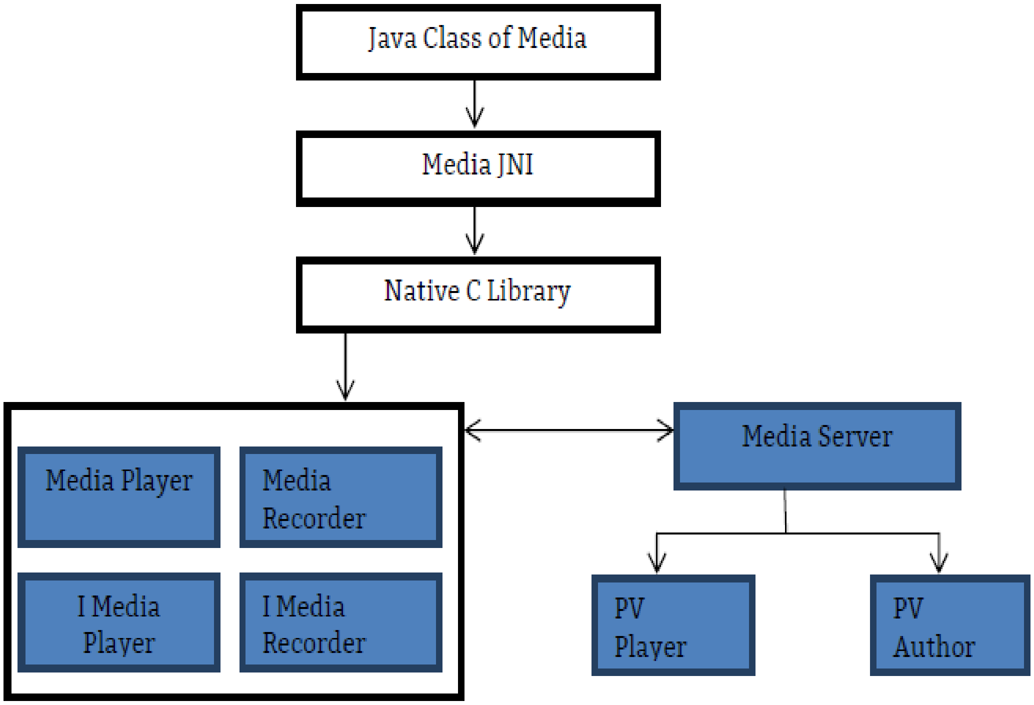


Fig. 5.6: Android Multimedia Framework Architecture

- The android multimedia system includes multimedia applications, multimedia framework, OpenCore engine and hardware abstract for audio/video input/output devices.
- The goal of the android multimedia framework is to provide a consistent interface for Java services.
- The multimedia framework consists of several core dynamic libraries such as libmediajni, libmedia, libmediaplayservice and so on
- Java classes call the Native C library Libmedia through Java JNI (Java Native Interface).
- Libmedia library communicates with Media Server guard process through Android's Binder IPC (inter process communication) mechanism.
- Media Server process creates the corresponding multimedia service according to the Java multimedia applications. The whole communication between Libmedia and Media Server forms a Client/Server model.
- In Media Server guard process, it calls OpenCore multimedia engine to realize the specific multimedia processing functions. And the OpenCore engine refers to the PVPlayer and PVAuthor.

3. **Attempt any THREE of the following:** **12 M**

a) **Describe various installation steps of android studio and its environment.** **4 M**

Ans **Step 1:**

Correct

		Go to Android https://developer.android.com/studio to get the Android Studio executable or zip file. **Step 2:** ● Click on the Download Android Studio Button. ● Click on the "I have read and agree with the above terms and conditions" checkbox followed by the download button ● Click on the Save file button in the appeared prompt box and the file will start downloading. **Step 3:** After the downloading has finished, open the file from downloads and will prompt the following dialog box. Click on next. In the next prompt, it'll ask for a path for installation. Choose a path and hit next. **Step 4:** It will start the installation, and once it is completed, it will be like the image shown below. **Step 5:** Once "Finish" is clicked, it will ask whether the previous settings need to be imported [if the android studio had been installed earlier], or not. It is better to choose the 'Don't import Settings option'. Click the OK button. **Step 6:** This will start the Android Studio. Meanwhile, it will be finding the available SDK components. **Step 7:** After it has found the SDK components, it will redirect to the Welcome dialog box. Choose Standard and click on Next. Now choose the theme, whether the Light theme or the Dark one. The light one is called the IntelliJ theme whereas the dark theme is called Dracula. Choose as required. Click on the Next button. **Step 8:** Now it is time to download the SDK components. Click on Finish. Components begin to download let it complete. The Android Studio has been successfully configured. Now it's time to launch and build apps. Click on the Finish button to launch it. **Step 9:** Click on Start a new Android Studio project to build a new app.	steps 4 marks
	b)	**Explain Gridview with its attributes with suitable example.**	**4 M**

Ans

GridView :

Android GridView shows items in two-dimensional scrolling grid (rows & columns) and the grid items are not necessarily predetermined but they automatically inserted to the layout using a ListAdapter.

GridView Attributes

Following are the important attributes specific to GridView –

Sr.No	Attribute & Description
1	**android:id** This is the ID which uniquely identifies the layout.
2	**android:columnWidth** This specifies the fixed width for each column. This could be in px, dp, sp, in, or mm.
3	**android:gravity** Specifies the gravity within each cell. Possible values are top, bottom, left, right, center, center_vertical, center_horizontal etc.
4	**android:horizontalSpacing** Defines the default horizontal spacing between columns. This could be in px, dp, sp, in, or mm.
5	**android:numColumns** Defines how many columns to show. May be an integer value, such as "100" or auto_fit which means display as many columns as possible to fill the available space.
6	**android:stretchMode** Defines how columns should stretch to fill the available empty space, if any. This must be either of the values − • none − Stretching is disabled. • spacingWidth − The spacing between each column is stretched. • columnWidth − Each column is stretched equally. • spacingWidthUniform − The spacing between each column is uniformly stretched..

(1 M for explanation of GridView,1 M for explaining attributes, 2 M example)

[any two attributes should be considered for 1 M, any valid example of GridView for 2 M]

7	**android:verticalSpacing** Defines the default vertical spacing between rows. This could be in px, dp, sp, in, or mm.

activity_main.xml Code :

```
<?xml version="1.0" encoding="utf-8"?>
<GridView  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:id="@+id/gridview"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:columnWidth="90dp"
android:gravity="center"
android:horizontalSpacing="10dp"
android:numColumns="auto_fit"
android:stretchMode="columnWidth"
android:verticalSpacing="10dp"
tools:context=".MainActivity">
</GridView>
```

activity_listview.xml code :

```
<?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">
<Button
android:id="@+id/btn"
android:layout_width="fill_parent"
```

```
android:layout_height="wrap_content"
android:layout_gravity="center" />
</LinearLayout>
```

MainActivity.java

```
package com.example.myapplication.gridviewbuttons;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.GridView;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
GridView gridview;
String arr[] = new String[15];
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
gridview = findViewById(R.id.gridview);
for (int i = 0; i < 15; i++) {
arr[i] = Integer.toString(i + 1);
}
ArrayAdapter<String> ad = new ArrayAdapter<String>(this, R.layout.activity_listview,
R.id.btn, arr);
gridview.setAdapter(ad);
}
}
```

	c)	**Explain text to speech conversion technique in android**	**4 M**
	Ans	Text to Speech converts the text written on the screen to speech like you have written “Hello World” on the screen and when you press the button it will speak “Hello World”. Text-to-speech is commonly used as an accessibility feature to help people who have trouble reading	Proper explanation

on-screen text, but it’s also convenient for those who want to be read too. This feature has come out to be a very common and useful feature for the users.

4 marks

In android, by using TextToSpeech class we can easily convert our text into voice and it supports different types of speaking languages. We can choose the speaking language based on our requirements in the android application.

The android TextToSpeech instance can only be used to synthesize text once it has completed its initialization so implement TextToSpeech.

OnInitListener to notify the completion of initialization. During the initialization, we can set the audio pitch rate, audio speed, type of language to speak, etc. based on our requirements.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
        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:orientation="vertical"
        android:layout_margin="30dp"
        tools:context=".MainActivity">

        <EditText
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/Text"
                android:layout_marginBottom="20dp"
                android:hint="Enter your  text"
                android:gravity="center"
                android:textSize="16dp"/>
```

```
    <Button
            android:layout_width="wrap_content"
            android:id="@+id/btnText"
            android:layout_height="wrap_content"
            android:text="Click"
            android:layout_gravity="center"/>


    <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="70dp"
            android:gravity="center_horizontal"
            android:text="MobileApplicationDevelopment"
            android:textSize="36sp" />

</LinearLayout>
```

MainActivity.java

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.speech.tts.TextToSpeech;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import java.util.Locale;
```

```
public class MainActivity extends AppCompatActivity {


        EditText Text;
        Button btnText;
        TextToSpeech textToSpeech;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);

                Text = findViewById(R.id.Text);
                btnText = findViewById(R.id.btnText);

                textToSpeech = new TextToSpeech(getApplicationContext(), new
TextToSpeech.OnInitListener() {
                        @Override
                        public void onInit(int i) {
                                if(i!=TextToSpeech.ERROR){
                                        // To Choose language of speech
                                        textToSpeech.setLanguage(Locale.UK);
                                }
                        }
                });

                btnText.setOnClickListener(new View.OnClickListener() {
                        @Override
                        public void onClick(View view) {

        textToSpeech.speak(Text.getText().toString(),TextToSpeech.QUEUE_FLUSH,null);
                        }
```

```
                    });


          }

}
```

d) **Describe steps for deploying android application on Google Play Store.** **4 M**

Ans

Step 1: Create a Developer Account
Before you can publish any app on Google Play, you need to create a Developer Account. You can easily sign up for one using your existing Google Account. You'll need to pay a one-time registration fee of $25 using your international credit or debit card. It can take up to 48 hours for your registration to be fully processed.

Correct steps 4 marks

Step 2: Plan to Sell? Link Your Merchant Account
If you want to publish a paid app or plan to sell in-app purchases, you need to create a payments center profile, i.e. a merchant account. A merchant account will let you manage your app sales and monthly payouts, as well as analyze your sales reports right in your Play Console.

Step 3: Create an App
Now you have create an application by clicking on 'Create Application'. Here you have to select your app's default language from the drop-down menu and then type in a title for your app. The title of your app will show on Google Play after you've published.

Step 4: Prepare Store Listing
Before you can publish your app, you need to prepare its store listing. These are all the details that will show up to customers on your app's listing on Google Play. You not necessarily complete it at once , you can always save a draft and revisit it later when you're ready to publish.
The information required for your store listing is divided into several categories such as Product Details containing title, short and full description of the app, Your app's title and description should be written with a great user experience in mind. Use the right keywords, but don't overdo it. Make sure your app doesn't come across as spam-y or promotional, or it will risk getting suspended on the Play Store.
Graphic Assets where you can add screenshots, images, videos, promotional graphics, and icons that showcase your app's features and functionality.
Languages & Translations, Categorization where in category can be selected to which your app belong to. Contact Details , Privacy Policy for apps that request access to sensitive user data or permissions, you need to enter a comprehensive privacy policy that effectively discloses how your app collects, uses, and shares that data.

Step 5: Upload APK to an App Release
Finally upload your app, by uploading APK file. Before you upload APK, you need to create an app release. You need to select the type of release you want to upload your first app version to. You can choose between an internal test, a closed test, an open test, and a

production release. The first three releases allow you to test out your app among a select group of users before you make it go live for everyone to access.
This is a safer option because you can analyze the test results and optimize or fix your app accordingly if you need to before rolling it out to all users.
Once you create a production release, your uploaded app version will become accessible to everyone in the countries you choose to distribute it in and click on ‘Create release.’

Step 6: Provide an Appropriate Content Rating
If you don’t assign a rating to your app, it will be listed as ‘Unrated’. Apps that are ‘Unrated’ may get removed from Google Play.
To rate your app, you need to fill out a content rating questionnaire An appropriate content rating will also help you get to the right audience, which will eventually improve your engagement rates.

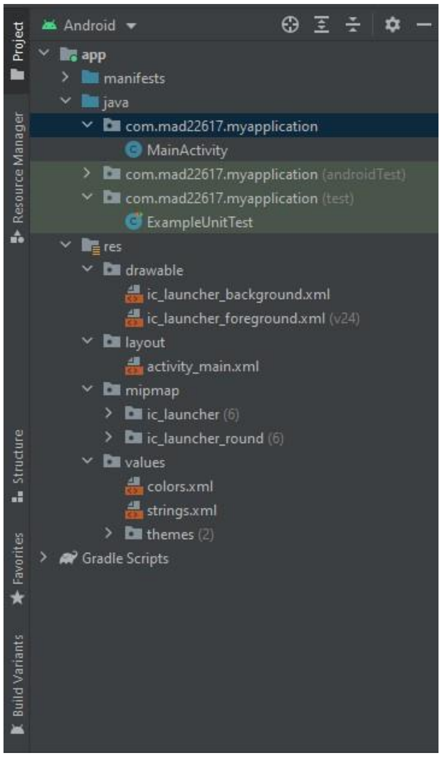
Step 7: Set Up Pricing & Distribution
Before you can fill out the details required in this step, you need to determine your app’s monetization strategy. Once you know how your app is going to make money, you can go ahead and set up your app as free or paid.
You can always change your app from paid to free later, but you cannot change a free app to paid. For that, you’ll need to create a new app and set its price.

Step 8: Rollout Release to Publish Your App
The final step involves reviewing and rolling out your release after making sure you’ve taken care of everything else.
Before you review and rollout your release, make sure the store listing, content rating, and pricing and distribution sections of your app each have a green check mark next to them.
Once you’re sure about the correctness of the details, select your app and navigate to ‘Release management’ – ‘App releases.’ You can always opt for reviews by clicking on ‘Review’ to be taken to the ‘Review and rollout release’ screen. Here, you can see if there are any issues or warnings you might have missed out on.
Finally, select ‘Confirm rollout.’ This will also publish your app to all users in your target countries on Google Play.

4.		**Attempt any <u>THREE</u> of the following:**	**12 M**
	a)	**Describe directory structure and its components.**	**4 M**
	Ans	The android project contains different types of app modules, source code files, and resource files. 1. Manifests Folder 2. Java Folder 3. res (Resources) Folder	1 M for listing of directory structure , 3 M for explanation)

- Drawable Folder
- Layout Folder
- Mipmap Folder
- Values Folder

4. Gradle Scripts



```
app
  manifests
  java
    com.mad22617.myapplication
      MainActivity
    com.mad22617.myapplication (androidTest)
    com.mad22617.myapplication (test)
      ExampleUnitTest
  res
    drawable
      ic_launcher_background.xml
      ic_launcher_foreground.xml (v24)
    layout
      activity_main.xml
    mipmap
      ic_launcher (6)
      ic_launcher_round (6)
    values
      colors.xml
      strings.xml
      themes (2)
Gradle Scripts
```

Manifests Folder

Manifests folder contains AndroidManifest.xml for creating our android application. This file contains information about our application such as the Android version, metadata, states package for Kotlin file, and other application components. It acts as an intermediator between android OS and our application.

Java folder

The Java folder contains all the java source code (.java) files that we create during the app development, including other Test files. If we create any new project using Java, by default the class file MainActivity.java will be created.

Resource (res) folder

The resource folder is the most important folder because it contains all the non-code sources like images, XML layouts, and UI strings for our android application.

res/drawable folder

It contains the different types of images used for the development of the application. We need to add all the images in a drawable folder for the application development.

res/layout folder

The layout folder contains all XML layout files which we used to define the user interface of our application. It contains the activity_main.xml file

res/mipmap folder

This folder contains launcher.xml files to define icons that are used to show on the home screen. It contains different density types of icons depending upon the size of the device such as hdpi, mdpi, xhdpi.

res/values folder

Values folder contains a number of XML files like strings, dimensions, colors, and style definitions. One of the most important files is the strings.xml file which contains the resources.

Gradle Scripts folder

Gradle means automated build system and it contains a number of files that are used to define a build configuration that can be applied to all modules in our application. In build.gradle (Project) there are buildscripts and in build.gradle (Module) plugins and implementations are used to build configurations that can be applied to all our application modules.

b) Develop an android application for Date and Time Picker. **4 M**

Ans **activity_main.xml**

(2M for Date Picker and 2M for Time Picker)

```
<?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"
tools:context=".MainActivity">
<EditText
android:layout_width="200dp"
android:layout_height="wrap_content"
android:id="@+id/in_time"
android:layout_alignParentLeft="true"
```

```
android:layout_alignParentStart="true" />
<Button
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="SELECT TIME"
android:id="@+id/btn_time"
android:layout_below="@+id/in_time"/>
</RelativeLayout>
```

MainActivity.java

```
package com.example.myapplication.timepickerwithspinnermode;
import android.app.TimePickerDialog;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TimePicker;
import androidx.appcompat.app.AppCompatActivity;
import java.util.Calendar;
public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
Button btnTimePicker;
EditText txtTime;
private int mHour, mMinute;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
btnTimePicker=(Button)findViewById(R.id.btn_time);
txtTime=(EditText)findViewById(R.id.in_time);
btnTimePicker.setOnClickListener(this);
}
@Override
public void onClick(View v) {
if (v == btnTimePicker) {
// Get Current Time
final Calendar c = Calendar.getInstance();
mHour = c.get(Calendar.HOUR_OF_DAY);
mMinute = c.get(Calendar.MINUTE);
// Launch Time Picker Dialog
TimePickerDialog timePickerDialog = new TimePickerDialog(this,
new TimePickerDialog.OnTimeSetListener() {
@Override
public void onTimeSet(TimePicker view, int hourOfDay,
int minute) {
txtTime.setText(hourOfDay + ":" + minute);
}
}, mHour, mMinute, false);
timePickerDialog.show();
}
}
}
```

	c)	**Explain property animation method to animate the properties of view object with example**.	**4 M**
	Ans	A property animation changes a property's (a field in an object) value over a specified length of time. To animate something, you specify the object property that you want to animate, such as an object's position on the screen, how long you want to animate it for, and what values you want to animate between. The property animation system lets you define the following characteristics of an animation: **Duration:** You can specify the duration of an animation. The default length is 300 ms. **Time interpolation:** You can specify how the values for the property are calculated as a	(2 M for explaining property animation method, 2 M for example)

function of the animation's current elapsed time.

Repeat count and behavior: You can specify whether or not to have an animation repeat when it reaches the end of a duration and how many times to repeat the animation. You can also specify whether you want the animation to play back in reverse. Setting it to reverse plays the animation forwards then backwards repeatedly, until the number of repeats is reached.

Animator sets: You can group animations into logical sets that play together or sequentially or after specified delays.

Frame refresh delay: You can specify how often to refresh frames of your animation. The default is set to refresh every 10 ms, but the speed in which your application can refresh frames is ultimately dependent on how busy the system is overall and how fast the system can service the underlying timer.

Strings.xml

```
<resources>
        <string name="app_name">Animation</string>
        <string name="blink">BLINK</string>
        <string name="fade">FADE</string>
        <string name="move">MOVE</string>
</resources>
```

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"
        tools:context=".MainActivity">

        <ImageView
        android:id="@+id/imageview"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="40dp"
        android:contentDescription="@string/app_name"
        android:src="@drawable/image" />

    <LinearLayout
        android:id="@+id/linear1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/imageview"
        android:layout_marginTop="30dp"
        android:orientation="horizontal"
        android:weightSum="3">



        <Button
            android:id="@+id/BTNblink"
            style="@style/TextAppearance.AppCompat.Widget.Button"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            android:layout_weight="1"
            android:padding="3dp"
            android:text="@string/blink"
            android:textColor="@color/white" />
```

```
                <Button
                    android:id="@+id/BTNfade"
                    style="@style/TextAppearance.AppCompat.Widget.Button"
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_margin="10dp"
                    android:layout_weight="1"
                    android:padding="3dp"
                    android:text="@string/fade"
                    android:textColor="@color/white" />


                <Button
                    android:id="@+id/BTNmove"
                    style="@style/TextAppearance.AppCompat.Widget.Button"
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_margin="10dp"
                    android:layout_weight="1"
                    android:padding="3dp"
                    android:text="@string/move"
                    android:textColor="@color/white" />

        </LinearLayout>

</RelativeLayout>
```

1) Blink Animation

```
    <?xml version="1.0" encoding="utf-8"?>
    <set xmlns:android="http://schemas.android.com/apk/res/android">
            <alpha android:fromAlpha="0.0"
                    android:toAlpha="1.0"
                    android:interpolator="@android:anim/accelerate_interpolator"
                    android:duration="500"
                    android:repeatMode="reverse"
                    android:repeatCount="infinite"/>
    </set>
```

2) Fade Animation

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/accelerate_interpolator">
    <alpha
            android:duration="1000"
            android:fromAlpha="0"
            android:toAlpha="1" />
    <alpha
            android:duration="1000"
            android:fromAlpha="1"
            android:startOffset="2000"
            android:toAlpha="0" />
</set>
```

3) Move Animation

```
    <?xml version="1.0" encoding="utf-8"?>
    <set
            xmlns:android="http://schemas.android.com/apk/res/android"
            android:interpolator="@android:anim/linear_interpolator"
            android:fillAfter="true">
    <translate
        android:fromXDelta="0%p"
        android:toXDelta="75%p"
        android:duration="700" />
</set>
```

MainActivity.java

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

    ImageView imageView;
    Button blinkBTN, fadeBTN, moveBTN;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        imageView = findViewById(R.id.imageview);
        blinkBTN = findViewById(R.id.BTNblink);
        fadeBTN = findViewById(R.id.BTNfade);
        moveBTN = findViewById(R.id.BTNmove);
```

```
                blinkBTN.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        // To add blink animation
                        Animation animation =
        AnimationUtils.loadAnimation(getApplicationContext(), R.anim.blink_animation);
                        imageView.startAnimation(animation);
                    }
                });

                fadeBTN.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        // To add fade animation
                        Animation animation =
        AnimationUtils.loadAnimation(getApplicationContext(), R.anim.fade_animation);
                        imageView.startAnimation(animation);
                    }
                });
                moveBTN.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        // To add move animation
                        Animation animation =
        AnimationUtils.loadAnimation(getApplicationContext(), R.anim.move_animation);
                        imageView.startAnimation(animation);
                    }
                });
  }
}
```

d) Describe permissions required for android application development. **4 M**

Ans

(2 Marks for two permission explanation)

The Android security model is primarily based on a sandbox and permission mechanism. Each application is running in a specific Dalvik virtual machine with a unique user ID assigned to it, which means the application code runs in isolation from the code of all other applications. Therefore, one application has not granted access to other applications' files.

Android application has been signed with a certificate with a private key Know the owner of the application is unique. This allows the author of the application will be identified if needed. When an application is installed in the phone is assigned a user ID, thus avoiding it from affecting it other applications by creating a sandbox for it. This user ID is permanent on which devices and applications with the same user ID are allowed to run in a single process. This is a way to ensure that a malicious application has Cannot access / compromise the data of the genuine application. It is mandatory for an application to list all the resources it will Access during installation. Terms are required of an application, in the installation process should be user-based or interactive Check with the signature of the application.

Declaring and Using Permissions

The purpose of a permission is to protect the privacy of an Android user. Android apps must request permission to access sensitive user data (such as contacts and SMS), as well as certain system features (such as camera and internet). Depending on the feature, the system might grant the permission automatically or might prompt the user to approve the request.

Permissions are divided into several protection levels. The protection level affects whether runtime permission requests are required. There are three protection levels that affect third-party apps: normal, signature, and dangerous permissions.

Normal permissions cover areas where your app needs to access data or resources outside the app's sandbox, but where there's very little risk to the user's privacy or the operation of other apps. For example, permission to set the time zone is a normal permission. If an app declares in its manifest that it needs a normal permission, the system automatically grants the app that permission at install time. The system doesn't prompt the user to grant normal permissions, and users cannot revoke these permissions.

Signature permissions: The system grants these app permissions at install time, but only when the app that attempts to use permission is signed by the same certificate as the app that defines the permission.

Dangerous permissions: Dangerous permissions cover areas where the app wants data or resources that involve the user's private information, or could potentially affect the user's stored data or the operation of other apps. For example, the ability to read the user's contacts is a dangerous permission. If an app declares that it needs a dangerous permission, the user must explicitly grant the permission to the app. Until the user approves the permission, your app cannot provide functionality that depends on that permission. To use a dangerous permission, your app must prompt the user to grant permission at runtime. For more details about how the user is prompted, see Request prompt for dangerous permission.

	e)	**Develop an android application to show current location of an user's car**	**4 M**
	Ans	**activity_maps.xml**	(2 M for xml code, 1 M java code, 1 M for permissions)

activity_maps.xml

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.com.mapexample.MapsActivity" />
```

MapsActivity.java

```
import android.os.Build;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;

import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.location.LocationServices;

import android.location.Location;
import android.Manifest;
import android.content.pm.PackageManager;

import android.support.v4.content.ContextCompat;

import com.google.android.gms.common.ConnectionResult;

import com.google.android.gms.location.LocationListener;

import com.google.android.gms.location.LocationRequest;



public class MapsActivity extends FragmentActivity implements OnMapReadyCallback,

        LocationListener,GoogleApiClient.ConnectionCallbacks,

        GoogleApiClient.OnConnectionFailedListener{


    private GoogleMap mMap;

    Location mLastLocation;

    Marker mCurrLocationMarker;

    GoogleApiClient mGoogleApiClient;

    LocationRequest mLocationRequest;


    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_maps);

        // Obtain the SupportMapFragment and get notified when the map is ready to be used.

        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()

                .findFragmentById(R.id.map);

        mapFragment.getMapAsync(this);


    }


    @Override
```

```
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        if (ContextCompat.checkSelfPermission(this,
                Manifest.permission.ACCESS_FINE_LOCATION)
                == PackageManager.PERMISSION_GRANTED) {
            buildGoogleApiClient();
            mMap.setMyLocationEnabled(true);
        }
    }
    else {
        buildGoogleApiClient();
        mMap.setMyLocationEnabled(true);
    }

}
protected synchronized void buildGoogleApiClient() {
    mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addConnectionCallbacks(this)
            .addOnConnectionFailedListener(this)
            .addApi(LocationServices.API).build();
    mGoogleApiClient.connect();
}

@Override
public void onConnected(Bundle bundle) {

    mLocationRequest = new LocationRequest();
```

```
        mLocationRequest.setInterval(1000);

        mLocationRequest.setFastestInterval(1000);


mLocationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY)
;

       if (ContextCompat.checkSelfPermission(this,

               Manifest.permission.ACCESS_FINE_LOCATION)

               == PackageManager.PERMISSION_GRANTED) {

           LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
mLocationRequest, this);

       }


    }

@Override

    public void onConnectionSuspended(int i) {


    }


    @Override

    public void onLocationChanged(Location location) {


        mLastLocation = location;

        if (mCurrLocationMarker != null) {

            mCurrLocationMarker.remove();

        }

        //Place current location marker

        LatLng latLng = new LatLng(location.getLatitude(), location.getLongitude());

        MarkerOptions markerOptions = new MarkerOptions();

        markerOptions.position(latLng);

        markerOptions.title("Current Position");

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_GREEN));
```

```
       mCurrLocationMarker = mMap.addMarker(markerOptions);


        //move map camera
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
        mMap.animateCamera(CameraUpdateFactory.zoomTo(11));


        //stop location updates
        if (mGoogleApiClient != null) {
          LocationServices.FusedLocationApi.removeLocationUpdates(mGoogleApiClient,
this);
        }
      }
      @Override
    public void onConnectionFailed(ConnectionResult connectionResult) {
      }
  }
```

Add the following user-permission in AndroidManifest.xml file.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

Note: only the permission line can be written , no entire code is required for manifest file.

5.		**Attempt any TWO of the following:**	**12 M**
	a)	**Design a employee registration form using UI component**.	**6 M**
	Ans	**activity_main.xml**	(Any Correct Design - XML file: 6M)

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

```
<TextView
   android:text="Employee Registration Form"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignParentTop="true"
   android:layout_centerHorizontal="true"
   android:id="@+id/textView"
   android:gravity="center"
   android:textSize="20dp"
   android:textColor="#000000"/>

<EditText
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:hint="ID"
   android:id="@+id/editid"
   android:layout_below="@+id/textView"/>

<EditText
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:hint="Name"
   android:id="@+id/editname"
   android:layout_below="@+id/editid"/>

<EditText
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:hint="Mobile No."
   android:id="@+id/editmobile"
   android:layout_below="@+id/editname"/>

<EditText
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:hint="Address"
   android:lines="3"
   android:id="@+id/editaddress"
   android:layout_below="@+id/editmobile"/>

<EditText
   android:layout_width="fill_parent"
```

```
        android:layout_height="wrap_content"
        android:hint="Pin Code"
        android:id="@+id/editpincode"
        android:layout_below="@+id/editaddress"/>

    <Button
        android:text="Submit Details"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editpincode"
        android:layout_centerHorizontal="true"
        android:id="@+id/button" />
</RelativeLayout>
```

b) **Develop an android application for taking student feedback with database connectivity.** **6 M**

Ans

(Any correct code can be consider

3 Marks for XML file and 3 marks for Java file)

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Student Feedback Form" />

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

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Roll No."
        android:id="@+id/editrollno"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Class"
        android:id="@+id/editclass"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter your Feedback"
        android:lines="3"
        android:id="@+id/editfeedback"/>

    <Button
        android:text="Submit Feedback"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:id="@+id/button" />
</LinearLayout>
```

MapsActivity.java

```
package com.example.feedback;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    SQLiteDatabase sqLiteDatabaseObj;
    Button submitBtn;
    EditText std_name, std_rollno, std_class, std_feedback;
    String sname, srollno, sclass, sfeedback, sql_query;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        submitBtn = (Button)findViewById(R.id.button);
        std_name = (EditText)findViewById(R.id.editname);
        std_rollno  = (EditText)findViewById(R.id.editrollno);
        std_class  = (EditText)findViewById(R.id.editclass);
        std_class  = (EditText)findViewById(R.id.editfeedback);

        submitBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                sqLiteDatabaseObj = openOrCreateDatabase("FeedbaseDataBase",
Context.MODE_PRIVATE, null);
                sqLiteDatabaseObj.execSQL("CREATE TABLE IF NOT EXISTS
Student(id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, name
VARCHAR, rollno VARCHAR, class VARCHAR, feedback VARCHAR);");
                sname = std_name.getText().toString();
                srollno = std_rollno.getText().toString() ;
                sclass = std_class.getText().toString();
                sfeedback = std_class.getText().toString();
                sql_query = "INSERT INTO Student (name, rollno, class, feedback)
VALUES('"+sname+"', '"+srollno+"', '"+sclass+"', '"+sfeedback+"')";
                sqLiteDatabaseObj.execSQL(sql_query);
                Toast.makeText(getApplicationContext(), "Feedback Submitted
Successfully", Toast.LENGTH_LONG).show();
            }
        });

    }
}
```

	c)	**Explain Geocoding and Reverse Geocoding with suitable example.**	**6 M**
	Ans	Geocoding is the process of transforming a street address or other description of a location into a (latitude, longitude) coordinate. Reverse geocoding is the process of transforming a (latitude, longitude) coordinate into a (partial) address. The amount of detail in a reverse geocoded location description may vary, for example one might contain the full street address of the closest building, while another might contain only a city name and postal code.	(Geocoding , Reverse Geocoding Explanation : 3 M, Example : 3 M)

The Geocoder class is used for handling geocoding and reverse geocoding.

activity_maps.xml

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.com.mapexample.MapsActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <EditText
            android:layout_width="248dp"
            android:layout_height="wrap_content"
            android:id="@+id/editText"
            android:hint="Search Location" />

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:onClick="searchLocation"
            android:text="Search" />

    </LinearLayout>

</fragment>
```

AndroidManifest.xml

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

MapsActivity.java

```
package example.com.mapexample;

import android.location.Address;
import android.location.Geocoder;
import android.os.Build;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;

import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.location.LocationServices;

import android.location.Location;
import android.Manifest;
import android.content.pm.PackageManager;
import android.support.v4.content.ContextCompat;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.location.LocationListener;
import com.google.android.gms.location.LocationRequest;

import java.io.IOException;
import java.util.List;

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback,
        LocationListener,GoogleApiClient.ConnectionCallbacks,
        GoogleApiClient.OnConnectionFailedListener{

    private GoogleMap mMap;
    Location mLastLocation;
    Marker mCurrLocationMarker;
    GoogleApiClient mGoogleApiClient;
    LocationRequest mLocationRequest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be
used.
  SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()

                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (ContextCompat.checkSelfPermission(this,
                    Manifest.permission.ACCESS_FINE_LOCATION)
                    == PackageManager.PERMISSION_GRANTED) {
                buildGoogleApiClient();
                mMap.setMyLocationEnabled(true);
            }
        }
        else {
            buildGoogleApiClient();
            mMap.setMyLocationEnabled(true);
        }

    }
    protected synchronized void buildGoogleApiClient() {
        mGoogleApiClient = new GoogleApiClient.Builder(this)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .addApi(LocationServices.API).build();
        mGoogleApiClient.connect();
    }

    @Override
    public void onConnected(Bundle bundle) {

        mLocationRequest = new LocationRequest();
        mLocationRequest.setInterval(1000);
        mLocationRequest.setFastestInterval(1000);

mLocationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY);
        if (ContextCompat.checkSelfPermission(this,
                Manifest.permission.ACCESS_FINE_LOCATION)
                == PackageManager.PERMISSION_GRANTED) {

LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
mLocationRequest, this);
        }

    }

    @Override
    public void onConnectionSuspended(int i) {
    }

    @Override
    public void onLocationChanged(Location location) {

        mLastLocation = location;
        if (mCurrLocationMarker != null) {
            mCurrLocationMarker.remove();
        }
        //Place current location marker
        LatLng latLng = new LatLng(location.getLatitude(), location.getLongitude());
        MarkerOptions markerOptions = new MarkerOptions();
        markerOptions.position(latLng);
        markerOptions.title("Current Position");

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_GREEN));
       mCurrLocationMarker = mMap.addMarker(markerOptions);

        //move map camera
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
        mMap.animateCamera(CameraUpdateFactory.zoomTo(11));

        //stop location updates
        if (mGoogleApiClient != null) {

LocationServices.FusedLocationApi.removeLocationUpdates(mGoogleApiClient, this);
        }

    }

    @Override
    public void onConnectionFailed(ConnectionResult connectionResult) {

    }

    public void searchLocation(View view) {
        EditText locationSearch = (EditText) findViewById(R.id.editText);
        String location = locationSearch.getText().toString();
        List<Address> addressList = null;

        if (location != null || !location.equals("")) {
            Geocoder geocoder = new Geocoder(this);
            try {
                addressList = geocoder.getFromLocationName(location, 1);

            } catch (IOException e) {
                e.printStackTrace();
            }
```

```
                Address address = addressList.get(0);
                LatLng latLng = new LatLng(address.getLatitude(), address.getLongitude());
                mMap.addMarker(new MarkerOptions().position(latLng).title(location));
                mMap.animateCamera(CameraUpdateFactory.newLatLng(latLng));
                Toast.makeText(getApplicationContext(),address.getLatitude()+"
      "+address.getLongitude(),Toast.LENGTH_LONG).show();
            }
        }

    }
```

6.		**Attempt any TWO of the following:**	**12 M**
	a)	**Design an android application to show the list of paired devices by Bluetooth.**	**6 M**
	Ans	**activity_main.xml**	**Layout file : 2M** **Java File : 3M** **Manifest file : 1M**

```
<?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"
   tools:context=".MainActivity"
   android:transitionGroup="true">
   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="List all Paired devices"
      android:onClick="list"
      android:id="@+id/button1"/>

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
     android:text="Paired devices:"
      android:id="@+id/textView1"
      android:textColor="#ff34ff06"
      android:textSize="25dp"
      android:layout_below="@+id/button1" />

   <ListView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/listView"
```

```
                android:layout_alignParentBottom="true"
                android:layout_below="@+id/textView1" />

        </RelativeLayout>
```

AndroidManifest.xml

```
    <?xml version="1.0" encoding="utf-8"?>
    <manifest xmlns:androclass="http://schemas.android.com/apk/res/android"
       package="com.example.bluetooth"
       android:versionCode="1"
       android:versionName="1.0" >

       <uses-sdk
          android:minSdkVersion="8"
          android:targetSdkVersion="16" />
       <uses-permission android:name="android.permission.BLUETOOTH" />
     <uses-
    permission android:name="android.permission.BLUETOOTH_ADMIN" />
       <application
          android:allowBackup="true"
          android:icon="@drawable/ic_launcher"
          android:label="@string/app_name"
          android:theme="@style/AppTheme" >
    <activity
             android:name=" in.org.msbte.bluetooth.MainActivity"
             android:label="@string/app_name" >
             <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                  <category android:name="android.intent.category.LAUNCHER" />
             </intent-filter>
          </activity>
       </application>
</manifest>
```

MainActivity.java

```
package in.org.msbte.bluetooth;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;

import android.content.Intent;
import android.view.View;

import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;

import android.widget.Toast;
import java.util.ArrayList;
import java.util.Set;

public class MainActivity extends AppCompatActivity {
  Button b1;
  private BluetoothAdapter BA;
  private Set<BluetoothDevice>pairedDevices;
  ListView lv;

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

    b1 = (Button) findViewById(R.id.button1);

    BA = BluetoothAdapter.getDefaultAdapter();
    lv = (ListView)findViewById(R.id.listView);
  }

  public void list(View v){
    pairedDevices = BA.getBondedDevices();

    ArrayList list = new ArrayList();

    for(BluetoothDevice bt : pairedDevices) list.add(bt.getName());
    Toast.makeText(getApplicationContext(), "Showing Paired
Devices",Toast.LENGTH_SHORT).show();

    final ArrayAdapter adapter = new
ArrayAdapter(this,android.R.layout.simple_list_item_1, list);

    lv.setAdapter(adapter);
  }
}
```

b) **Develop an android application for sending Short Message Service (SMS).** **6 M**

Ans

activity_main.xml

(XML file 3 marks Java file 3 Marks)

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

```
    <TextView
        android:id="@+id/fstTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:layout_marginTop="150dp"
        android:text="Mobile No" />
    <EditText
        android:id="@+id/mblTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:ems="10"/>

    <TextView
        android:id="@+id/secTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Message"
        android:layout_marginLeft="100dp" />
    <EditText
        android:id="@+id/msgTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:ems="10" />
    <Button
        android:id="@+id/btnSend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:text="Send SMS" />
</LinearLayout>
```

MainActivity.java

```
package in.org.msbte.sendsmsexample;
import android.content.Intent;
import android.net.Uri;
import android.provider.Telephony;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
public class MainActivity extends AppCompatActivity {

    private EditText txtMobile;
    private EditText txtMessage;
    private Button btnSms;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtMobile = (EditText)findViewById(R.id.mblTxt);
        txtMessage = (EditText)findViewById(R.id.msgTxt);
        btnSms = (Button)findViewById(R.id.btnSend);
        btnSms.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try{
                    SmsManager smgr = SmsManager.getDefault();

smgr.sendTextMessage(txtMobile.getText().toString(),null,txtMessage.getText().toString(),null,null);
                    Toast.makeText(MainActivity.this, "SMS Sent Successfully",
Toast.LENGTH_SHORT).show();
                }
                catch (Exception e){
                    Toast.makeText(MainActivity.this, "SMS Failed to Send, Please try again",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

c) **Explain how linear and frame layout is used to design an android application with suitable example.** **6 M**

Ans

LinearLayout

(2 M : For each layout explanation, 1 M for each layout example)

- Android LinearLayout is a view group that aligns all children in either vertically or horizontally.
- Linear layout in Android allow us to arrange components horizontally in a single column or
  vertically in a single row.
- Vertically or horizontally direction depends on attribute android: orientation.
- Linear layout is simple and easy to use, it creates a scroll bar if the length of the window exceeds the length of the screen.
- Linear Layout are one of the simplest and common type of layouts used by Android developers to keep controls within their interfaces. The linear layout works as much as its name implies, it organizes the controls either a vertical or horizontal pattern.
- When the layout’s orientation is set to vertical, all child controls within

organized in a single column, and when the layout's orientation is set to horizontal, all child controls within in single row.

Example

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutandroid:layout_width="368dp"
    android:layout_height="495dp"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    tools:layout_editor_absoluteX="8dp"
    tools:layout_editor_absoluteY="8dp"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <Button
      android:id="@+id/button5"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Button1" />
    <Button
      android:id="@+id/button6"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Button2" />
    <Button
      android:id="@+id/button7"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Button3" />
    <Button
      android:id="@+id/button8"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Button4" />
</LinearLayout>
```

Frame Layout

- Frame Layout is designed to block out an area on the screen to display a single item. Generally, FrameLayout should be used to hold a single child view, because it can be difficult to organize child views in a way that's scalable to different screen sizes without the children overlapping each other.

- Frame layouts are one of the simplest layout types used to organize controls within the user interface of an Android application. The purpose of FrameLayout is to allocate an area of screen.
- Frame layouts are one of the most efficient types of layouts used by Android developers to organize view controls. They are used less often than some other layouts, simply because they are generally used to display only one view, or views which overlap.
- The frame layout is often used as a container layout, as it generally only has a single child view (often another layout, used to organize more than one view).

Example

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayoutxmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/framelayout"
    android:layout_width="200dp"
    android:layout_height="300dp"
    tools:context=".MainActivity">
    <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="90dp"
            android:layout_marginLeft="20dp"
            android:text="Button"/>
    <TextView
            android:layout_width="100dp"
            android:layout_height="50dp"
            android:textSize="20sp"
            android:layout_marginTop="20dp"
            android:layout_marginLeft="20dp"
            android:background="@color/colorPrimary"
            android:textColor="#fff"
            android:text="I am TextView" />
 </FrameLayout
```