Experiment No 5

Develop GMaps application by using Linear Layout Views with different attributes

Aim: To Develop a GMaps application by using Linear Layout Views with different Attributes.

Procedure:

Step1: Create the New Project->Empty Views Activity

Step 2: Design the User Interface in the **activity_main.xml** file

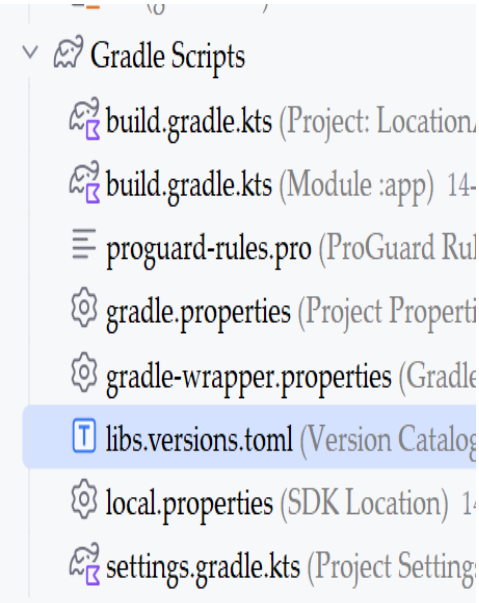
activity_main.xml

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

    <Button
        android:id="@+id/fetchLocationButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Fetch Current Location"
        android:textColor="@android:color/white"
        android:layout_marginBottom="16dp" />

    <org.osmdroid.views.MapView
        android:id="@+id/map"
        android:layout_width="match_parent"
        android:layout_height="0dp"
```

```
        android:layout_weight="1" />

</LinearLayout>
```

Note: For using GMaps API we have to configure the two step Verification Process which includes your card and Payment Gateway process even for using free trial of API. So, for that alternate I'm using OSM (Open Street Map) OSMDroid in this Application.

For Configuring OSMDroid in your application follow the Steps,



In libs.versions.toml file add this two things.

```
[versions]
osmdroid = "6.1.18"

[libraries]
osmdroid-android = { module =
"org.osmdroid:osmdroid-android",
version.ref = "osmdroid" }
```

Step 3: Add this Dependency in your Gradle Scripts build.gradle.kts

implementation(*libs.osmdroid.android*)

MainActivity.kt

```
package com.example.locationapp

import android.Manifest
import android.annotation.SuppressLint
```

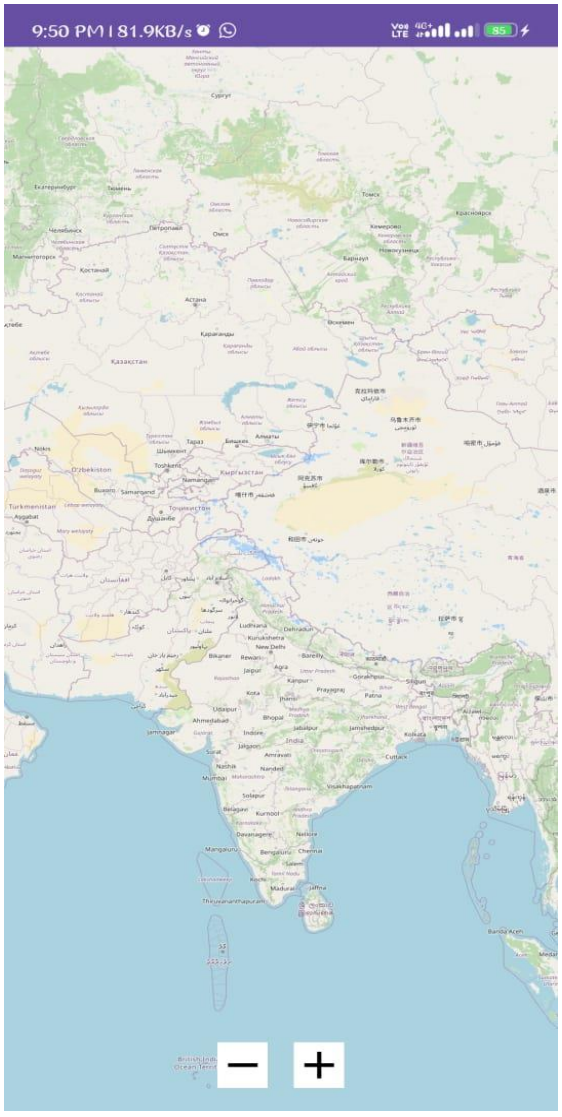
```
import android.content.pm.PackageManager
import android.os.Bundle
import android.widget.Button
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import org.osmdroid.config.Configuration
import org.osmdroid.util.GeoPoint
import org.osmdroid.views.MapView
import org.osmdroid.views.overlay.Marker
import org.osmdroid.views.overlay.mylocation.MyLocationNewOverlay
import org.osmdroid.views.overlay.mylocation.GpsMyLocationProvider
import org.osmdroid.tileprovider.tilesource.TileSourceFactory

class MainActivity : AppCompatActivity() {

    private lateinit var mapView: MapView
    private lateinit var locationOverlay: MyLocationNewOverlay
    private lateinit var fetchLocationButton: Button

    companion object {
        private const val REQUEST_LOCATION_PERMISSION = 1
    }

    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        // Load OSMDroid configuration
        Configuration.getInstance().load(this, getPreferences(MODE_PRIVATE))
        setContentView(R.layout.activity_main)

        // Initialize the MapView
        mapView = findViewById(R.id.map)
        mapView.setTileSource(TileSourceFactory.MAPNIK) // Set tile source
        mapView.setMultiTouchControls(true) // Enable multi-touch gestures

        // Set up button to fetch current location
```

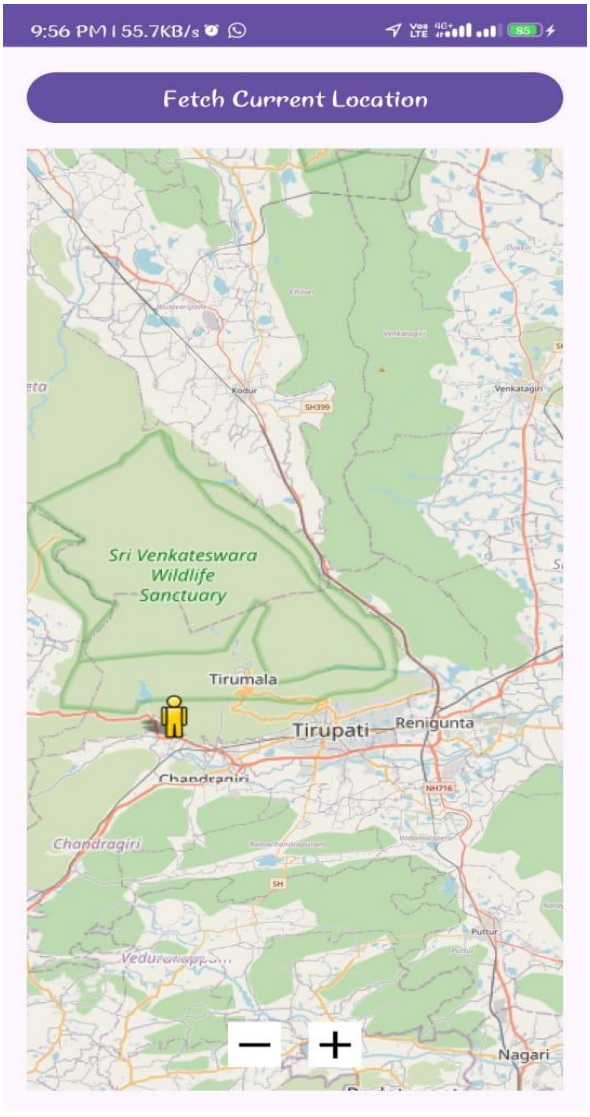
```
        fetchLocationButton = findViewById(R.id.fetchLocationButton)
        fetchLocationButton.setOnClickListener {
            fetchCurrentLocation()
        }

        // Setup location overlay
        setupLocationOverlay()
    }

    private fun setupLocationOverlay() {
        // Check for location permission
        if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED) {
            // Request location permissions if not already granted
            ActivityCompat.requestPermissions(
                this,
                arrayOf(Manifest.permission.ACCESS_FINE_LOCATION),
                REQUEST_LOCATION_PERMISSION
            )
        } else {
            // Initialize the location overlay
            locationOverlay =
MyLocationNewOverlay(GpsMyLocationProvider(this), mapView)
            locationOverlay.enableMyLocation() // Enable current location
            mapView.overlays.add(locationOverlay)
            mapView.invalidate() // Refresh the map view
        }
    }

    private fun fetchCurrentLocation() {

        val currentLocation: GeoPoint? = locationOverlay.myLocation
        if (currentLocation != null) {
            val currentGeoPoint = GeoPoint(currentLocation.latitude,
currentLocation.longitude)
            mapView.controller.setCenter(currentGeoPoint) // Center the map on the
current location
```

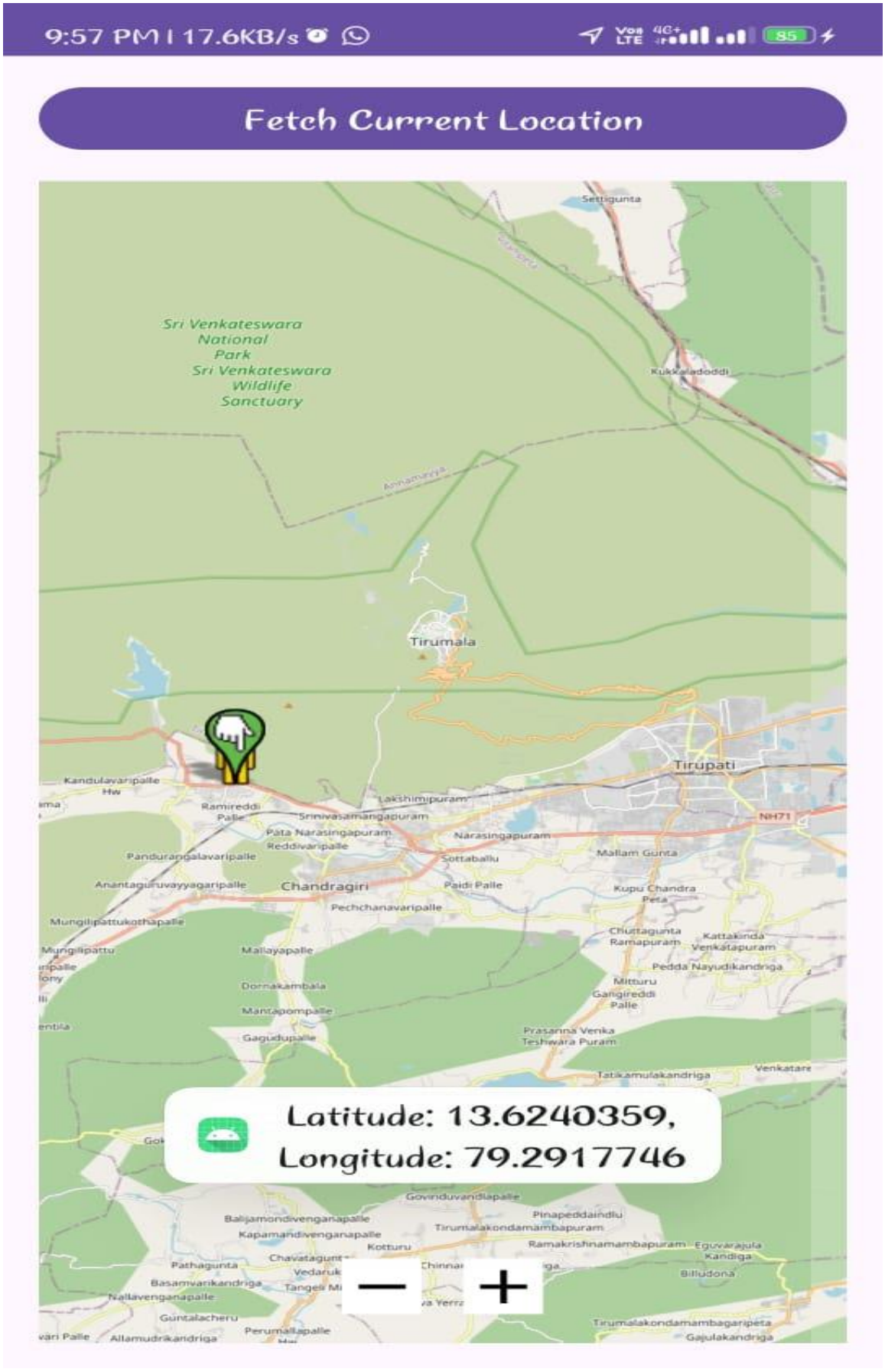
```
            Toast.makeText(this, "Latitude: ${currentLocation.latitude}, Longitude:
${currentLocation.longitude}", Toast.LENGTH_SHORT).show()


            addMarker(currentGeoPoint, "Current Location")
        } else {
            Toast.makeText(this, "Current Location: Not available",
Toast.LENGTH_SHORT).show()
        }
    }

    private fun addMarker(geoPoint: GeoPoint, title: String) {
        val marker = Marker(mapView)
        marker.position = geoPoint
        marker.setAnchor(Marker.ANCHOR_CENTER,
Marker.ANCHOR_BOTTOM)
        marker.title = title
        mapView.overlays.add(marker)
        mapView.invalidate()
    }

    override fun onRequestPermissionsResult(
        requestCode: Int,
        permissions: Array<String>,
        grantResults: IntArray
    ) {
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults)
        if (requestCode == REQUEST_LOCATION_PERMISSION) {
            if (grantResults.isNotEmpty() && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                setupLocationOverlay()
            } else {
                Toast.makeText(this, "Location permission denied",
Toast.LENGTH_SHORT).show()
            }
        }
    }
```

```
    override fun onResume() {
        super.onResume()
        mapView.onResume()
    }

    override fun onPause() {
        super.onPause()
        mapView.onPause()up
    }
}
```

Output:






9:57 PM | 17.6KB/s
Fetch Current Location
Sri Venkateswara National Park
Sri Venkateswara Wildlife Sanctuary
Tirumala
Tirupati
Chandragiri
Latitude: 13.6240359,
Longitude: 79.2917746

Experiment No 6

Develop a Program to implement a Custom Button and Handle the displayed message on the Button Press

Aim: To Develop a Program to implement a Custom Button and Handle the displayed message on the Button Press.

Procedure:

Step1: Create the New Project->Empty Views Activity

Step 2: Design the User Interface in the **activity_main.xml** file

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:id="@+id/main"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">
   <EditText
      android:id="@+id/editTextMobile"
      android:layout_width="300dp"
      android:layout_height="wrap_content"
      android:layout_below="@id/textView"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="20dp"
      android:hint="@string/enter_your_register_number"
      android:inputType="textShortMessage"
      android:textSize="16sp"
      android:padding="10dp"
      android:backgroundTint="@android:color/darker_gray"
      android:fontFamily="sans-serif" />
```

```
<TextView
    android:id="@+id/textView"
    android:layout_centerInParent="true"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="50dp"
    android:text="@string/welcome_to_mohan_babu_university"
    android:textSize="18sp"
    android:textColor="@color/black"
    android:fontFamily="sans-serif-medium" />

  <Button
    android:id="@+id/button"
    android:layout_width="200dp"
    android:layout_height="60dp"
    android:layout_below="@id/editTextMobile"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="20dp"
    android:text="@string/register"
    android:textSize="18sp"
    android:textColor="@color/black"
    android:backgroundTint="@android:color/darker_gray"
    android:fontFamily="sans-serif-medium"
    android:elevation="8dp" />

</RelativeLayout>
```

In this experiment just we are going to see how to use and implement Notification for our application. For displaying the notification there are four important concepts that we have to know.

1. NOTIFICATION CHANNEL

2. NOTIFICATION BUILDER

3. NOTIFICATION MANAGER

4. PENDING INTENT

MainActivity.java

```
///Steps for Creating the Notifications
///STEP 1: Create a Notification Channel by method and declare the variables globally that is channel id and name
///STEP 2: Create a Notification Builder check for the API version
///Check for if(CURRENT_API>=O ) && if the android version is 13 or above you have set permissions in manifest file
///STEP 3: After the builder is created we have to create the pending intent to navigate to our activity that we have created
///Finally run the application->click the home button->You will see the notification->when clicking the notification it should be navigated to the target activity

package com.example.notification

import android.annotation.SuppressLint
import android.app.NotificationChannel
import android.app.NotificationManager
import android.app.PendingIntent
import android.content.Context
import android.content.Intent
import android.content.pm.PackageManager
import android.graphics.Color
import android.os.Build
import android.os.Bundle
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.NotificationCompat
import androidx.core.app.NotificationManagerCompat

class MainActivity : AppCompatActivity() {
    val NOTIFICATION_CHANNEL_ID = "my_channel_id"
    val NOTIFICATION_CHANNEL_NAME = "my_channel_name"
    val NOTIFICATION_ID = 1
```

```
    @SuppressLint("MissingPermission")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        createNotificationChannel()
        ///Creating Pending Intent
        val intent=Intent(this,MainActivity2::class.java)

        ///...This is used for the older API versions
//        val pendingIntent=TaskStackBuilder.create(this).run {
//            addNextIntentWithParentStack(intent)
//            getPendingIntent(0,PendingIntent.FLAG_UPDATE_CURRENT)
//        }

        ///...This is used for the modern API versions
        val pendingIntent = PendingIntent.getActivity(this, 0, intent,
PendingIntent.FLAG_IMMUTABLE)

        ///NOTE: Android 13 and Higher versions require Post Notifications permission
to be granted

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.TIRAMISU) {
            if
(checkSelfPermission(android.Manifest.permission.POST_NOTIFICATIONS)
!= PackageManager.PERMISSION_GRANTED) {

requestPermissions(arrayOf(android.Manifest.permission.POST_NOTIFICATI
ONS), 1)
            }
        }

///...After the Notification is created for showing notification we have to Build the
Notification with NotificationCompat.Builder
        ///NOTIFICATION BUILDER
        val notification = NotificationCompat.Builder(this,
NOTIFICATION_CHANNEL_ID)
            .setContentTitle("MBU Id Registered Successfully")
            .setContentText("Click here to explore")
```

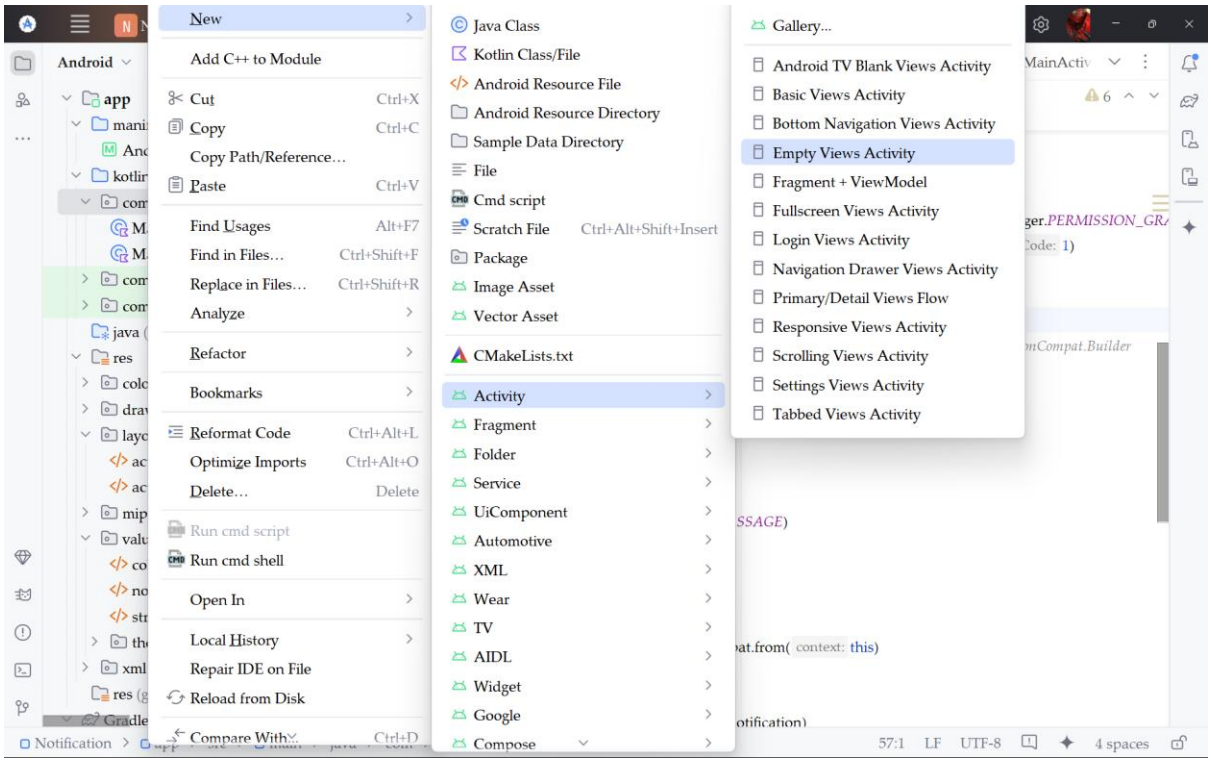
```
            .setSmallIcon(R.drawable.notification_foreground)
            .setPriority(NotificationCompat.PRIORITY_HIGH)
            .setCategory(NotificationCompat.CATEGORY_MESSAGE)
            .setContentIntent(pendingIntent)
            .setAutoCancel(true)
            .build()

        val notificationManager = NotificationManagerCompat.from(this)
        val button = findViewById<Button>(R.id.button)
        button.setOnClickListener {
            notificationManager.notify(NOTIFICATION_ID, notification)
        }
    }

    ///...This is used to create the notification channel
    @SuppressLint("ObsoleteSdkInt")
    private fun createNotificationChannel() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            val channel = NotificationChannel(
                NOTIFICATION_CHANNEL_ID,
                NOTIFICATION_CHANNEL_NAME,
                NotificationManager.IMPORTANCE_HIGH
            ).apply {
                lightColor = Color.GREEN
                enableLights(true)
            }
            val manager = getSystemService(Context.NOTIFICATION_SERVICE) as
NotificationManager
            manager.createNotificationChannel(channel)
        }
    }
}
```

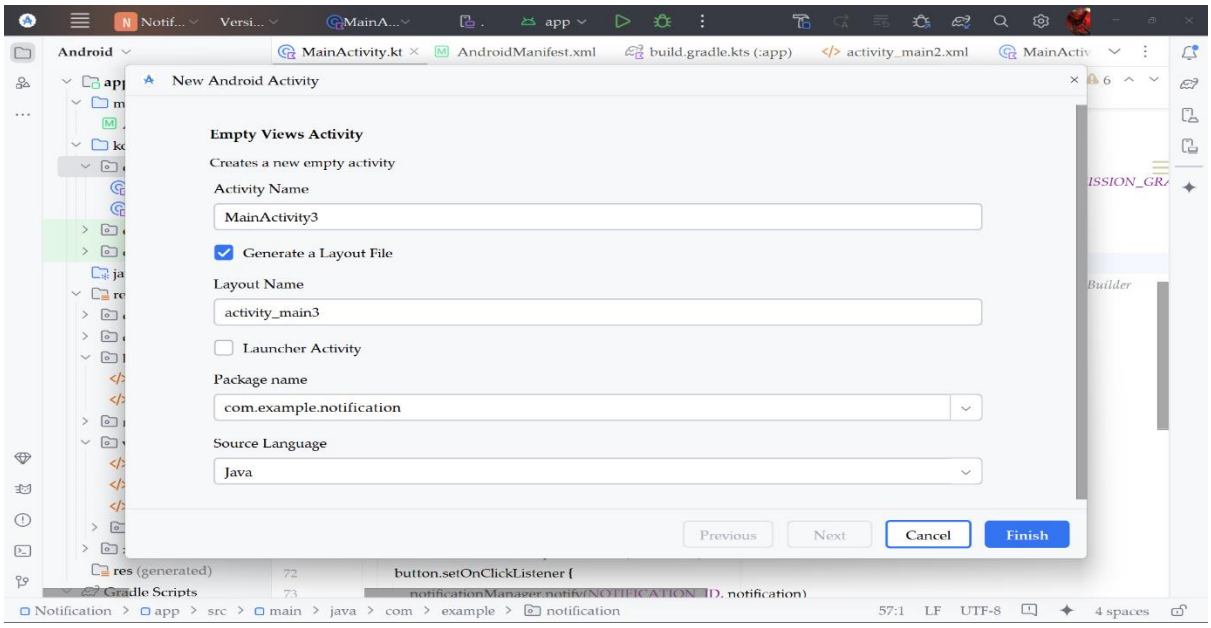
Here the first phase of the experiment is completed for creating the custom button. Simply we have designed the button and displayed a message.

NOTE: Convert the Kotlin code to Java and configure the project according to the requirements.

The second Phase is to Handling the displayed message. For this simply you can create another **Empty Views Activity by right clicking your package name**-> After that you can configure your second activity as your wish.

Note: If you are clicking the Notification it should navigate to the newly created activity.





In this experiment I'm using Web view as the Second Activity

activity_main2.xml

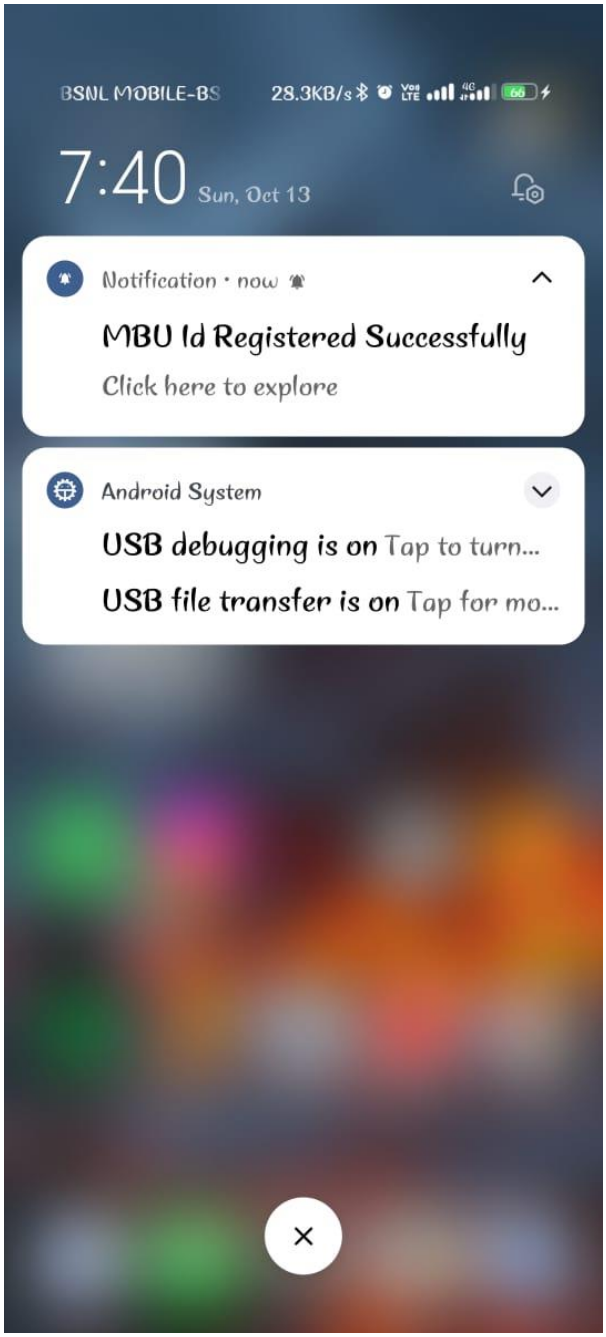
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity2">

    <WebView
        android:id="@+id/oklWebView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</RelativeLayout>
```

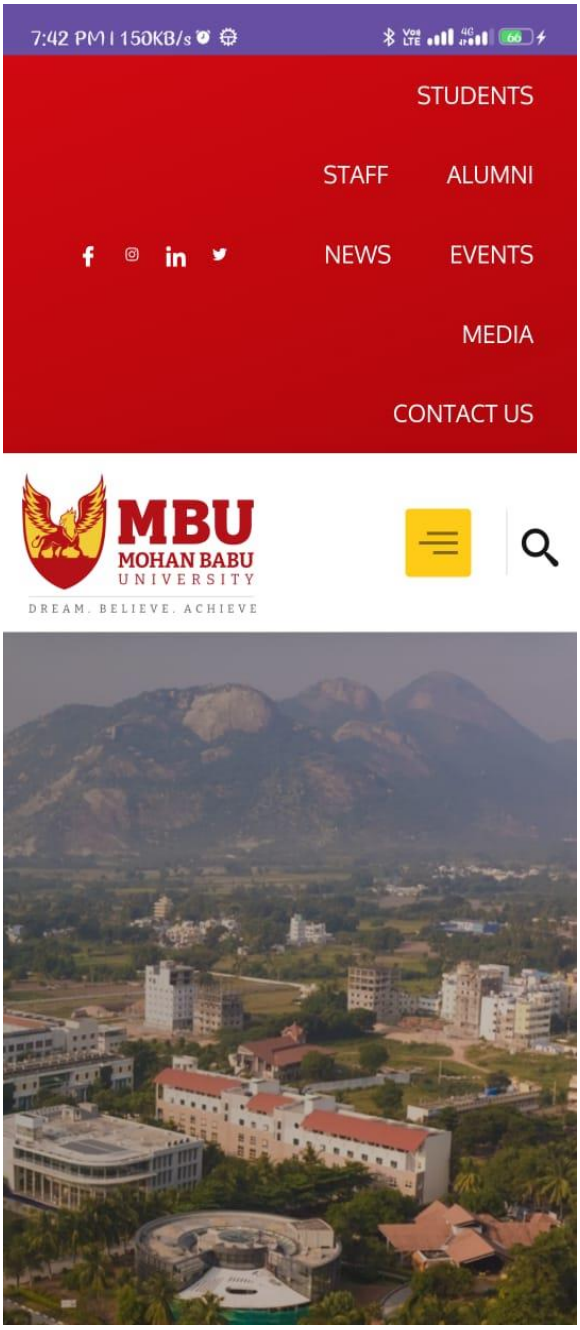
MainActivity2.kt

```
package com.example.notification

import android.annotation.SuppressLint
import android.os.Bundle
import android.webkit.WebView
import android.webkit.WebViewClient
import androidx.appcompat.app.AppCompatActivity

class MainActivity2 : AppCompatActivity() {
    @SuppressLint("SetJavaScriptEnabled")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main2)

        val url=intent.getStringExtra("url")

        val webView=findViewById<WebView>(R.id.oklWebView)
```

```
        webView.webViewClient=WebViewClient()
        webView.settings.javaScriptEnabled=true
        webView.loadUrl("https://www.mbu.asia/")
    }
}
```

OUTPUT:





RESULT:

Finally, the program for implementing a Custom Button and Handle the displayed message on the Button Press was implemented and Executed Successfully.

SREE VIDYANIKETHAN ENGINEERING COLLEGE

SreeSainath Nagar, Tirupati – 517 102

CAREER DEVELOPMENT CENTRE (CDC)

SVEC/CDC/Placement/Notification/2024 21st October 2024

NOTIFICATION

TAP Academy Campus Hiring 2025

TAP Academy is a leading software company and skilling organization dedicated to helping students upskill and stay aligned with current IT needs. We offer concept visualization through Augmented Reality-based training, a tech-enabled world-class skilling ecosystem, and the pure joy of learning.

Visit Us: TAP Academy Website

In addition to our internship and training programs, we provide unlimited placement opportunities to all selected students, helping them secure their dream jobs.

TAP Academy - Campus to Corporate Internship Drive 2025

Who Should Attend?

- **Position:** Software Development Intern
- **Qualifications:** B.Tech/MCA
- **Streams:** All branches
- **Year of Passing:** 2025
- **Percentage Criteria:** No criteria

TAP Academy Campus to Corporate Internship Drive Process:

1. Awareness session on TAP Academy and the C2C Internship, followed by a pre-placement talk and assessment.
    - Aptitude Test
    - Communication Round
    - Technical & HR Interview
2. Selected students will receive an internship offer letter on the same day.
3. The internship will be conducted in offline mode during the 8th semester.
4. TAP Academy will take full responsibility for offline training and internships, placing selected students at zero cost.
5. The internship will focus on Full Stack Web Development, with unlimited placements until students secure jobs.
6. Internship duration: 4 months.

Drive Mode: On Campus, 25th October 2025.

Registration Link: https://forms.gle/LPP9UCvDyQuZ9WoR7 (deadline: 9 am tomorrow)

(Dr. K. DELHI BABU)
Vice President – Career Development Centre

VEERARAGHAVULU CHEEKATI

Professional Title

Short and engaging pitch about yourself.

veeracheekati2003@gmail.com

8008723878

SKILLS

java | python | html5 | sql | aws | c# | JavaScript

LANGUAGES

English
Professional Working Proficiency

telugu
Full Professional Proficiency

Language
Full Professional Proficiency

INTERESTS

designing

programing

Interest

EDUCATION

BACHELOR OF TECHNOLOGY
SREE VIDYANIKETHAN ENGINEERING COLLEGE

11/2021 - Present *TIRUPATI,INDIA*

Courses
- COMPUTER SCIENCE AND DESIGN

INTERMEDIATE
VIKAS JUNIOR COLLEGE

06/2017 - 07/2021 *B N KANDRIGA,TIRUPATI*

Courses
- MPC

BOARD OF SECONDARY EDUCATION
SRINIVASA TEJA E.M SCHOOL

04/2018 - 04/2019 *B N KANDRIGA,TIRUPATI*

Courses
- SSC

WORK EXPERIENCE

Google Android Developer Virtual internship
AICTU-Eduskills

04/2024 - 06/2024

Salesforce Supported Virtual Internship Program
SmartBridge(Salesforce)

05/2023 - 07/2023

CERTIFICATES

GOOGLE IT SUPPORT Professional (Coursera)

Azure Fundamentals

person mepro

PERSONAL PROJECTS

Voice assistance using python

Facebook clone using web technology