**Implicit intent**

<!-- res/layout/activity\_main.xml -->

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="16dp">

<EditText

android:id="@+id/urlEditText"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Enter URL (e.g., http://www.google.com)" />

<Button

android:id="@+id/visitButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Vissssssit"

android:layout\_below="@id/urlEditText"

android:layout\_marginTop="20dp" />

</RelativeLayout>

// MainActivity.kt

import android.content.Intent

import android.net.Uri

import android.os.Bundle

import android.widget.Button

import android.widget.EditText

import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

val urlEditText: EditText = findViewById(R.id.urlEditText)

val visitButton: Button = findViewById(R.id.visitButton)

visitButton.setOnClickListener {

val url = urlEditText.text.toString()

if (url.isNotEmpty()) {

val intent = Intent(Intent.ACTION\_VIEW)

intent.data = Uri.parse(url)

startActivity(intent)

}

}

}

}

**Explicit intent**

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout

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=".MainActivity">

<!-- URL Input -->

<EditText

android:id="@+id/e1"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:hint="Enter URL"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:layout\_marginTop="80dp"

android:layout\_marginStart="32dp"

android:layout\_marginEnd="32dp" />

<!-- Fetch Button -->

<Button

android:id="@+id/b1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Fetch"

app:layout\_constraintTop\_toBottomOf="@+id/e1"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:layout\_marginTop="24dp" />

<!-- Input A -->

<EditText

android:id="@+id/e2"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:hint="Enter A"

app:layout\_constraintTop\_toBottomOf="@id/b1"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:layout\_marginTop="40dp"

android:layout\_marginStart="32dp"

android:layout\_marginEnd="32dp" />

<!-- Input B -->

<EditText

android:id="@+id/e3"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:hint="Enter B"

app:layout\_constraintTop\_toBottomOf="@id/e2"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:layout\_marginTop="16dp"

android:layout\_marginStart="32dp"

android:layout\_marginEnd="32dp" />

<!-- Add Button -->

<Button

android:id="@+id/b2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Add"

app:layout\_constraintTop\_toBottomOf="@id/e3"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:layout\_marginTop="32dp" />

</androidx.constraintlayout.widget.ConstraintLayout>

package com.example.practice  
  
import android.content.Intent  
import android.net.Uri  
import android.os.Bundle  
import android.widget.EditText  
import android.widget.Button  
import androidx.activity.enableEdgeToEdge  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.view.ViewCompat  
import androidx.core.view.WindowInsetsCompat  
  
class MainActivity : AppCompatActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 setContentView(R.layout.*activity\_main*)  
 val e1:EditText=findViewById(R.id.*e1*)  
 val e2:EditText=findViewById(R.id.*e2*)  
 val e3:EditText=findViewById(R.id.*e3*)  
 val b1:Button=findViewById(R.id.*b1*)  
 val b2:Button=findViewById(R.id.*b2*)  
 b1.setOnClickListener**{** val ur=e1.*text*.toString()  
 val i=Intent(Intent.*ACTION\_VIEW*)  
 i.setData(Uri.parse(ur))  
 startActivity(i)  
 **}** b2.setOnClickListener**{** val s1=e2.*text*.toString().*toIntOrNull*()?:0  
 val s2=e3.*text*.toString().*toIntOrNull*()?:0  
val s=s1+s2  
 val i=Intent(this,MainActivity2::class.*java*)  
 i.putExtra("v",s)  
  
 startActivity(i)  
 **}** }  
}

**options menu**

**themes.xml**

<!-- themes.xml -->

<style name="Theme.Practice" parent="**Theme.AppCompat.Light.DarkActionBar**">

<!-- Customize your theme here. -->

</style>

**Menu/options\_menu.xml**

<?xml version="1.0" encoding="utf-8"?>  
<menu xmlns:android="http://schemas.android.com/apk/res/android">  
<item android:id="@+id/oi1" android:title="oi1"/>  
 <item android:id="@+id/oi2" android:title="oi2"/>  
</menu>

**Out of oncreate**

override fun onCreateOptionsMenu(menu: Menu?): Boolean {  
 *menuInflater*.inflate(R.menu.*option\_menu*,menu)  
 return true  
}  
  
override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 return when(item.*itemId*){  
 R.id.*oi1*->{  
 val i=Intent(this,MainActivity2::class.*java*)  
 startActivity(i)  
 true  
 }  
 else -> super.onOptionsItemSelected(item)  
 }  
}

**context menu-long press**

**1)inside Oncreate**

val tv1: TextView =findViewById(R.id.*tv1*)

registerForContextMenu(tv1) tv1->view(text)

**2)out of oncreate**

override fun onCreateContextMenu(  
 menu: ContextMenu?,  
 v: View?,  
 menuInfo: ContextMenu.ContextMenuInfo?  
) {  
 super.onCreateContextMenu(menu, v, menuInfo)  
  
 if(v?.*id* ==R.id.*tv1*){  
 menu?.setHeaderTitle("context menu")  
 menu?.add(0,v.*id*,0,"ci1")  
 menu?.add(0,v.*id*,1,"ci2")  
  
  
 }  
 }  
  
override fun onContextItemSelected(item: MenuItem): Boolean {  
  
 return when(item.*order*){  
 0->{ Toast.makeText(this,"context it1m 1",Toast.*LENGTH\_LONG*).show()  
 true  
 }  
 1->{ Toast.makeText(this,"context it1m 2",Toast.*LENGTH\_LONG*).show()  
 true  
 }  
 else->super.onContextItemSelected(item)  
 }  
}

**popup menu**

**1)create button**

**2)pop\_menu.xml**

<?xml version="1.0" encoding="utf-8"?>  
<menu xmlns:android="http://schemas.android.com/apk/res/android">  
 <item android:id="@+id/pi1" android:title="pi1"/> <item android:id="@+id/pi2" android:title="pi2"/>  
</menu>

**3)inside oncreate**

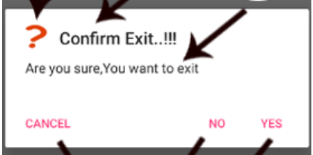
val b4:Button=findViewById(R.id.*b4*)

b4.setOnClickListener{  
 val p=PopupMenu(this,b4)  
 p.*menuInflater*.inflate(R.menu.*pop\_menu*,p.*menu*)  
 p.setOnMenuItemClickListener {item->  
 when(item.*itemId*){  
 R.id.*pi1*->{  
 val i=Intent(this,MainActivity2::class.*java*)  
 startActivity(i)  
 true  
 }  
 else ->false  
 }  
 }  
}

**Alert**

**1)backpressed**

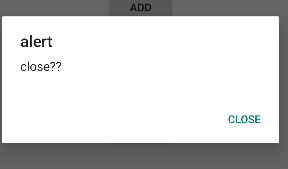
// Exit Confirmation Dialog  
override fun onBackPressed() {  
 val builder = AlertDialog.Builder(this)  
  
 builder.setTitle("Exit Confirmation")  
 .setMessage("Are you sure you want to exit?")  
 .setPositiveButton("Yes") { \_, \_ ->  
 Toast.makeText(*applicationContext*, "Exiting", Toast.*LENGTH\_SHORT*).show()  
 finish()  
 }  
 .setNegativeButton("No") { dialog, \_ ->  
 dialog.dismiss()  
 }  
 .setNeutralButton("Maybe") { \_, \_ ->  
 Toast.makeText(*applicationContext*, "Maybe later", Toast.*LENGTH\_SHORT*).show()  
 }  
 .setCancelable(true)  
 .show()  
}



**2)on clicking button**

**Inside on create**

val b5:Button=findViewById(R.id.*b5*)  
b5.setOnClickListener{  
 val b= AlertDialog.Builder(this)  
 b.setTitle("alert")  
 .setMessage("close??")  
 .setNegativeButton("close"){dialog,\_,->  
 Toast.makeText(this,"context it1m 1",Toast.*LENGTH\_LONG*).show()  
 dialog.dismiss()  
 }  
 .setCancelable(true)  
 .show()  
}



**Date and time picker**

1)

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="24dp">

<Button

android:id="@+id/dateButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Pick Date" />

<TextView

android:id="@+id/selectedDateTextView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Selected Date"

android:layout\_marginTop="8dp" />

<Button

android:id="@+id/timeButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Pick Time"

android:layout\_marginTop="24dp" />

<TextView

android:id="@+id/selectedTimeTextView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Selected Time"

android:layout\_marginTop="8dp" />

</LinearLayout>

2)

val dateButton: Button = findViewById(R.id.dateButton)

val timeButton: Button = findViewById(R.id.timeButton)

val selectedDateTextView: TextView = findViewById(R.id.selectedDateTextView)

val selectedTimeTextView: TextView = findViewById(R.id.selectedTimeTextView)

val calendar = Calendar.getInstance()

// Date Picker

dateButton.setOnClickListener {

val year = calendar.get(Calendar.YEAR)

val month = calendar.get(Calendar.MONTH)

val day = calendar.get(Calendar.DAY\_OF\_MONTH)

val datePickerDialog = DatePickerDialog(this, { \_, selectedYear, selectedMonth, selectedDay ->

val formattedDate = "$selectedDay/${selectedMonth + 1}/$selectedYear"

selectedDateTextView.text = formattedDate

}, year, month, day)

datePickerDialog.show()

}

// Time Picker

timeButton.setOnClickListener {

val hour = calendar.get(Calendar.HOUR\_OF\_DAY)

val minute = calendar.get(Calendar.MINUTE)

val timePickerDialog = TimePickerDialog(this, { \_, selectedHour, selectedMinute ->

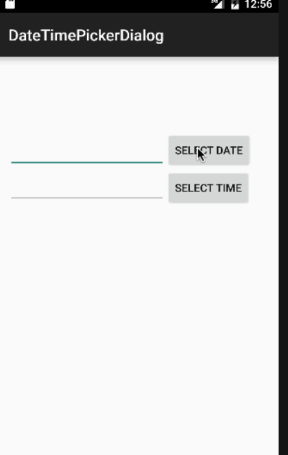
val formattedTime = String.format("%02d:%02d", selectedHour, selectedMinute)

selectedTimeTextView.text = formattedTime

}, hour, minute, true)

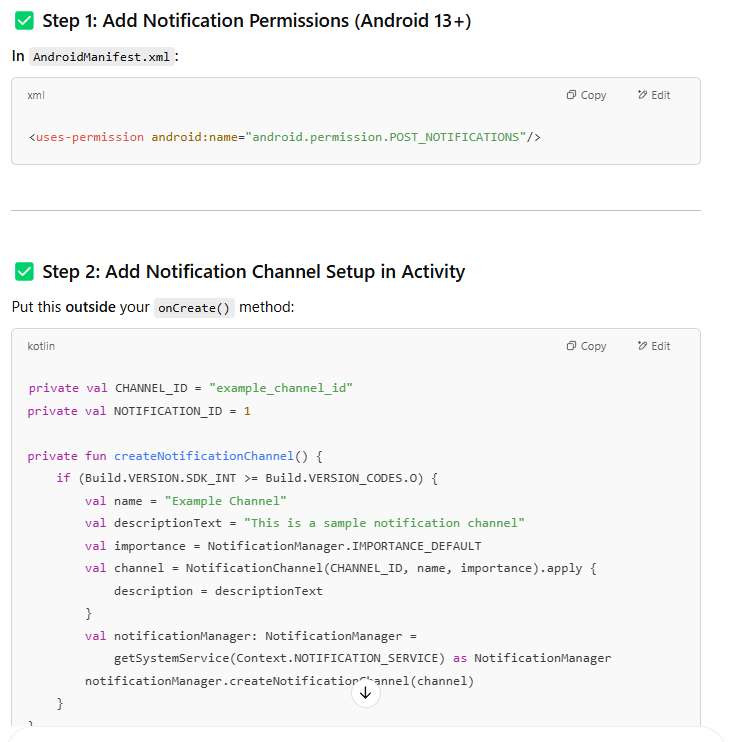
timePickerDialog.show()

}

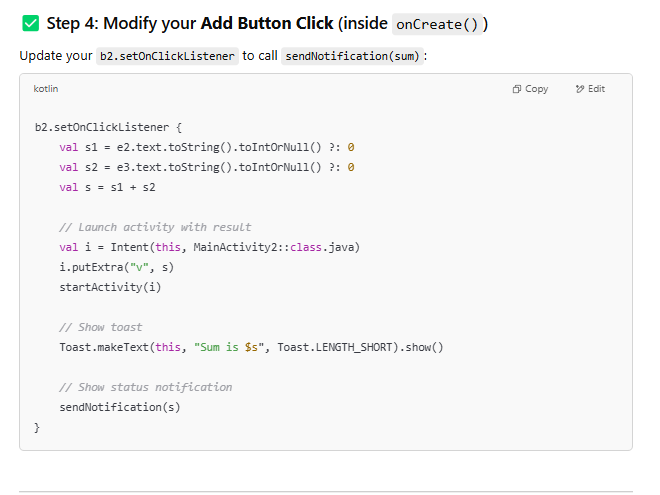


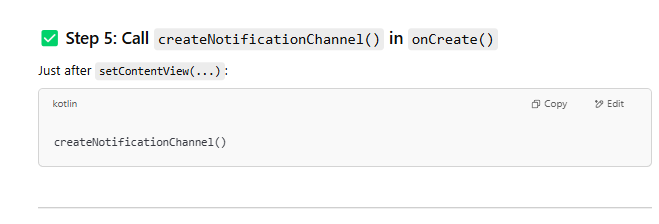
2) val tv3: TextView =findViewById(R.id.*tv3*)  
 val b6:Button=findViewById(R.id.*b6*)  
 val c= Calendar.getInstance()  
 b6.setOnClickListener**{** val year=c.get(Calendar.*YEAR*)  
 val mon=c.get(Calendar.*MONTH*)  
 val da=c.get(Calendar.*DAY\_OF\_MONTH*)  
 val d=DatePickerDialog(this,**{**\_,selectedYear,selectedMonth,selectedDay **->**val o="$selectedYear"  
 tv3.*text*=o  
 **}**,year,mon,da).show()  
 **}**

**Status notification:**









package com.example.practice

import android.app.\*

import android.content.\*

import android.net.Uri

import android.os.Build

import android.os.Bundle

import android.view.\*

import android.widget.\*

import androidx.activity.enableEdgeToEdge

import androidx.appcompat.app.AppCompatActivity

import androidx.core.app.ActivityCompat

import androidx.core.app.NotificationCompat

import androidx.core.app.NotificationManagerCompat

import com.example.practice.R

import java.util.\*

class MainActivity : AppCompatActivity() {

private val CHANNEL\_ID = "example\_channel\_id"

private val NOTIFICATION\_ID = 1

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

enableEdgeToEdge()

setContentView(R.layout.activity\_main)

// Notification channel setup

createNotificationChannel()

val e1: EditText = findViewById(R.id.e1)

val e2: EditText = findViewById(R.id.e2)

val e3: EditText = findViewById(R.id.e3)

val b1: Button = findViewById(R.id.b1)

val b2: Button = findViewById(R.id.b2)

val b4: Button = findViewById(R.id.b4)

val b5: Button = findViewById(R.id.b5)

val b6: Button = findViewById(R.id.b6)

val tv1: TextView = findViewById(R.id.tv1)

val tv3: TextView = findViewById(R.id.tv3)

// Open URL

b1.setOnClickListener {

val ur = e1.text.toString()

val i = Intent(Intent.ACTION\_VIEW)

i.data = Uri.parse(ur)

startActivity(i)

}

// Sum and Notification

b2.setOnClickListener {

val s1 = e2.text.toString().toIntOrNull() ?: 0

val s2 = e3.text.toString().toIntOrNull() ?: 0

val s = s1 + s2

Toast.makeText(this, "Sum is $s", Toast.LENGTH\_SHORT).show()

val i = Intent(this, MainActivity2::class.java)

i.putExtra("v", s)

startActivity(i)

sendNotification(s)

}

// Context Menu

registerForContextMenu(tv1)

// Popup Menu

b4.setOnClickListener {

val p = PopupMenu(this, b4)

p.menuInflater.inflate(R.menu.pop\_menu, p.menu)

p.setOnMenuItemClickListener { item ->

when (item.itemId) {

R.id.pi1 -> {

Toast.makeText(this, "Popup Item 1 clicked", Toast.LENGTH\_SHORT).show()

true

}

R.id.pi2 -> {

Toast.makeText(this, "Popup Item 2 clicked", Toast.LENGTH\_SHORT).show()

true

}

else -> false

}

}

p.show()

}

// Alert Dialog

b5.setOnClickListener {

val b = AlertDialog.Builder(this)

b.setTitle("Alert")

.setMessage("Close the app?")

.setNegativeButton("Close") { dialog, \_ ->

Toast.makeText(this, "Closed", Toast.LENGTH\_SHORT).show()

dialog.dismiss()

}

.setCancelable(true)

.show()

}

// Date Picker Dialog

b6.setOnClickListener {

val c = Calendar.getInstance()

val year = c.get(Calendar.YEAR)

val mon = c.get(Calendar.MONTH)

val da = c.get(Calendar.DAY\_OF\_MONTH)

DatePickerDialog(this, { \_, selectedYear, selectedMonth, selectedDay ->

val date = "$selectedDay/${selectedMonth + 1}/$selectedYear"

tv3.text = date

}, year, mon, da).show()

}

}

override fun onCreateOptionsMenu(menu: Menu?): Boolean {

menuInflater.inflate(R.menu.option\_menu, menu)

return true

}

override fun onOptionsItemSelected(item: MenuItem): Boolean {

return when (item.itemId) {

R.id.oi1 -> {

val i = Intent(this, MainActivity2::class.java)

startActivity(i)

true

}

else -> super.onOptionsItemSelected(item)

}

}

override fun onCreateContextMenu(

menu: ContextMenu?,

v: View?,

menuInfo: ContextMenu.ContextMenuInfo?

) {

super.onCreateContextMenu(menu, v, menuInfo)

if (v?.id == R.id.tv1) {

menu?.setHeaderTitle("Context Menu")

menu?.add(0, v.id, 0, "ci1")

menu?.add(0, v.id, 1, "ci2")

}

}

override fun onContextItemSelected(item: MenuItem): Boolean {

return when (item.order) {

0 -> {

Toast.makeText(this, "Context Item 1", Toast.LENGTH\_SHORT).show()

true

}

1 -> {

Toast.makeText(this, "Context Item 2", Toast.LENGTH\_SHORT).show()

true

}

else -> super.onContextItemSelected(item)

}

}

private fun createNotificationChannel() {

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {

val name = "Example Channel"

val descriptionText = "This is a sample notification channel"

val importance = NotificationManager.IMPORTANCE\_DEFAULT

val channel = NotificationChannel(CHANNEL\_ID, name, importance).apply {

description = descriptionText

}

val notificationManager: NotificationManager =

getSystemService(Context.NOTIFICATION\_SERVICE) as NotificationManager

notificationManager.createNotificationChannel(channel)

}

}

private fun sendNotification(sum: Int) {

val intent = Intent(this, MainActivity2::class.java)

val pendingIntent = PendingIntent.getActivity(

this, 0, intent, PendingIntent.FLAG\_IMMUTABLE

)

val builder = NotificationCompat.Builder(this, CHANNEL\_ID)

.setSmallIcon(R.drawable.ic\_launcher\_foreground)

.setContentTitle("Addition Result")

.setContentText("Sum is $sum")

.setPriority(NotificationCompat.PRIORITY\_DEFAULT)

.setContentIntent(pendingIntent)

.setAutoCancel(true)

with(NotificationManagerCompat.from(this)) {

if (ActivityCompat.checkSelfPermission(

applicationContext,

android.Manifest.permission.POST\_NOTIFICATIONS

) != PackageManager.PERMISSION\_GRANTED

) {

ActivityCompat.requestPermissions(

this@MainActivity,

arrayOf(android.Manifest.permission.POST\_NOTIFICATIONS),

101

)

return

}

notify(NOTIFICATION\_ID, builder.build())

}

}

}

**Shared preference-inside on create**

1)

val etUsername = findViewById<EditText>(R.id.etUsername)

val etPassword = findViewById<EditText>(R.id.etPassword)

val btnSave = findViewById<Button>(R.id.btnSave)

val btnView = findViewById<Button>(R.id.btnView)

val btnClear = findViewById<Button>(R.id.btnClear)

val btnDelete = findViewById<Button>(R.id.btnDelete)

val sharedPref = getSharedPreferences("UserPref", Context.MODE\_PRIVATE)

btnSave.setOnClickListener {

val editor = sharedPref.edit()

editor.putString("username", etUsername.text.toString())

editor.putString("password", etPassword.text.toString())

editor.apply()

Toast.makeText(this, "Saved Successfully", Toast.LENGTH\_SHORT).show()

}

btnView.setOnClickListener {

val username = sharedPref.getString("username", "No Data")

val password = sharedPref.getString("password", "No Data")

etUsername.setText(username)

etPassword.setText(password)

}

btnClear.setOnClickListener {

etUsername.setText("")

etPassword.setText("")

}

btnDelete.setOnClickListener {

val editor = sharedPref.edit()

editor.clear()

editor.apply()

Toast.makeText(this, "Deleted Successfully", Toast.LENGTH\_SHORT).show()

}

**2)Login validation**

val btnLogin = findViewById<Button>(R.id.btnLogin)

btnLogin.setOnClickListener {

val inputUsername = etUsername.text.toString()

val inputPassword = etPassword.text.toString()

val storedUsername = sharedPref.getString("username", null)

val storedPassword = sharedPref.getString("password", null)

if (storedUsername == null || storedPassword == null) {

Toast.makeText(this, "No credentials found. Please save first.", Toast.LENGTH\_SHORT).show()

} else if (inputUsername == storedUsername && inputPassword == storedPassword) {

Toast.makeText(this, "Login Successful", Toast.LENGTH\_SHORT).show()

} else {

Toast.makeText(this, "Invalid Credentials", Toast.LENGTH\_SHORT).show()

}

}

**Map**

**Geocoder**

**1)permission**

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

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />



val etLat = findViewById<EditText>(R.id.etLat)

val etLng = findViewById<EditText>(R.id.etLng)

val btnFetchAddress = findViewById<Button>(R.id.btnFetchAddress)

val tvAddressResult = findViewById<TextView>(R.id.tvAddressResult)

btnFetchAddress.setOnClickListener {

val lat = etLat.text.toString().toDoubleOrNull()

val lng = etLng.text.toString().toDoubleOrNull()

if (lat != null && lng != null) {

val address = getAddressFromLatLng(lat, lng)

tvAddressResult.text = address ?: "Address not found"

} else {

Toast.makeText(this, "Invalid latitude or longitude", Toast.LENGTH\_SHORT).show()

}

}

**3)Out of create()**



private fun getAddressFromLatLng(lat: Double, lng: Double): String? {

return try {

val geocoder = Geocoder(this, Locale.getDefault())

val addresses = geocoder.getFromLocation(lat, lng, 1)

if (!addresses.isNullOrEmpty()) {

val address = addresses[0]

"${address.getAddressLine(0)}, ${address.locality}, ${address.countryName}"

} else {

"No address found"

}

} catch (e: Exception) {

e.printStackTrace()

"Error fetching address"

}

}

**Location Based**

**1. AndroidManifest.xml (Permissions)**

**<uses-permission android:name="android.permission.INTERNET"/>**

**<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION"/>**

**<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION"/>**

**2. activity\_main.xml (Layout)**

**<?xml version="1.0" encoding="utf-8"?>**

**<androidx.constraintlayout.widget.ConstraintLayout**

**xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:app="http://schemas.android.com/apk/res-auto"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent">**

**<TextView**

**android:id="@+id/title"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Thiagarajar College of Engineering"**

**android:textSize="28sp"**

**android:textColor="#009688"**

**app:layout\_constraintTop\_toTopOf="parent"**

**app:layout\_constraintBottom\_toBottomOf="parent"**

**app:layout\_constraintLeft\_toLeftOf="parent"**

**app:layout\_constraintRight\_toRightOf="parent"**

**app:layout\_constraintVertical\_bias="0.08" />**

**<Button**

**android:id="@+id/show\_my\_location"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Show My Location"**

**app:layout\_constraintTop\_toBottomOf="@id/title"**

**app:layout\_constraintBottom\_toBottomOf="parent"**

**app:layout\_constraintLeft\_toLeftOf="parent"**

**app:layout\_constraintRight\_toRightOf="parent"**

**android:layout\_marginTop="24dp"/>**

**</androidx.constraintlayout.widget.ConstraintLayout>**

**3. MainActivity.kt (Kotlin Code)**

**package com.example.practice**

**import android.Manifest**

**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 com.google.android.gms.location.FusedLocationProviderClient**

**import com.google.android.gms.location.LocationServices**

**class MainActivity : AppCompatActivity() {**

**private lateinit var locationProviderClient: FusedLocationProviderClient**

**private lateinit var showLocation: Button**

**override fun onCreate(savedInstanceState: Bundle?) {**

**super.onCreate(savedInstanceState)**

**setContentView(R.layout.activity\_main)**

**// Step 2.1: Variable declarations**

**locationProviderClient = LocationServices.getFusedLocationProviderClient(this)**

**showLocation = findViewById(R.id.show\_my\_location)**

**// Step 2.4: OnClickListener**

**showLocation.setOnClickListener {**

**getYourCurrentLocation()**

**}**

**}**

**// Step 3.1: Getting current location**

**private fun getYourCurrentLocation() {**

**if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_FINE\_LOCATION)**

**!= PackageManager.PERMISSION\_GRANTED &&**

**ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_COARSE\_LOCATION)**

**!= PackageManager.PERMISSION\_GRANTED**

**) {**

**// Request permission**

**ActivityCompat.requestPermissions(**

**this,**

**arrayOf(Manifest.permission.ACCESS\_FINE\_LOCATION),**

**909**

**)**

**return**

**}**

**// Step 3.7: Fetching location**

**val task = locationProviderClient.lastLocation**

**task.addOnSuccessListener {**

**if (it != null) {**

**Toast.makeText(**

**this,**

**"Latitude: ${it.latitude} \nLongitude: ${it.longitude}",**

**Toast.LENGTH\_LONG**

**).show()**

**} else {**

**Toast.makeText(**

**this,**

**"Your location can't be displayed",**

**Toast.LENGTH\_LONG**

**).show()**

**}**

**}**

**}**

**// Optional: Handle permission result**

**override fun onRequestPermissionsResult(**

**requestCode: Int,**

**permissions: Array<out String>,**

**grantResults: IntArray**

**) {**

**super.onRequestPermissionsResult(requestCode, permissions, grantResults)**

**if (requestCode == 909 && grantResults.isNotEmpty() && grantResults[0] == PackageManager.PERMISSION\_GRANTED) {**

**getYourCurrentLocation()**

**} else {**

**Toast.makeText(this, "Permission denied", Toast.LENGTH\_SHORT).show()**

**}**

**}**

**}**