**PRACTICAL: 11**

**AIM:**

Creating an application that provides Single Sign-on (SSO) with Chrome Custom Tabs via the AppAuth library, and optionally push managed configuration to provide a user login hint.

**THEORY:**

Single sign-on: Single sign-on is a property of access control of multiple related, yet independent, software systems. With this property, a user logs in with a single ID and password to gain access to any of several related systems.

GoogleSignInAccount: Class that holds the basic account information of the signed in Google user.

GoogleSignInClient: A client for interacting with the Google Sign In API.

**CODE:**

|  |
| --- |
| **// MainActivity.java**  package com.example.pr11;  import androidx.annotation.Nullable;  import androidx.appcompat.app.AppCompatActivity;  import android.content.Intent;  import android.os.Bundle;  import android.util.Log;  import android.view.View;  import android.widget.Toast;  import com.google.android.gms.auth.api.signin.GoogleSignIn;  import com.google.android.gms.auth.api.signin.GoogleSignInAccount;  import com.google.android.gms.auth.api.signin.GoogleSignInClient;  import com.google.android.gms.auth.api.signin.GoogleSignInOptions;  import com.google.android.gms.common.SignInButton;  import com.google.android.gms.common.api.ApiException;  import com.google.android.gms.tasks.Task;  public class MainActivity extends AppCompatActivity {  private GoogleSignInClient mGoogleSignInClient;  private GoogleSignInOptions gso;  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_main);  gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT\_SIGN\_IN)  .requestEmail()  .build();  mGoogleSignInClient = GoogleSignIn.getClient(this, gso);  SignInButton signInButton = findViewById(R.id.sign\_in\_button);  signInButton.setSize(SignInButton.SIZE\_STANDARD);  signInButton.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  switch (v.getId()) {  case R.id.sign\_in\_button:  Intent signInIntent = mGoogleSignInClient.getSignInIntent();  startActivityForResult(signInIntent, 9411);  break;  }  }  });  }  @Override  protected void onStart() {  super.onStart();  GoogleSignInAccount account = GoogleSignIn.getLastSignedInAccount(this);  if ( account == null) return;  updateUI(account);  }  private void handleSignInResult(Task<GoogleSignInAccount> completedTask) {  try {  GoogleSignInAccount account = completedTask.getResult(ApiException.class);  updateUI(account);  } catch (ApiException e) {  Log.e("SSO","signInResult:failed code=" + e.getStatusCode());  }  }  private void updateUI(GoogleSignInAccount account) {  Toast.makeText(this,"Welcome " + account.getDisplayName(),Toast.LENGTH\_LONG).show();  }  @Override  protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  super.onActivityResult(requestCode, resultCode, data);  if (requestCode == 9411) {  Task<GoogleSignInAccount> task = GoogleSignIn.getSignedInAccountFromIntent(data);  handleSignInResult(task);  }  }  }  **activity\_mail.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:gravity="center"  tools:context=".MainActivity">  <com.google.android.gms.common.SignInButton  android:id="@+id/sign\_in\_button"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" />  </LinearLayout> |

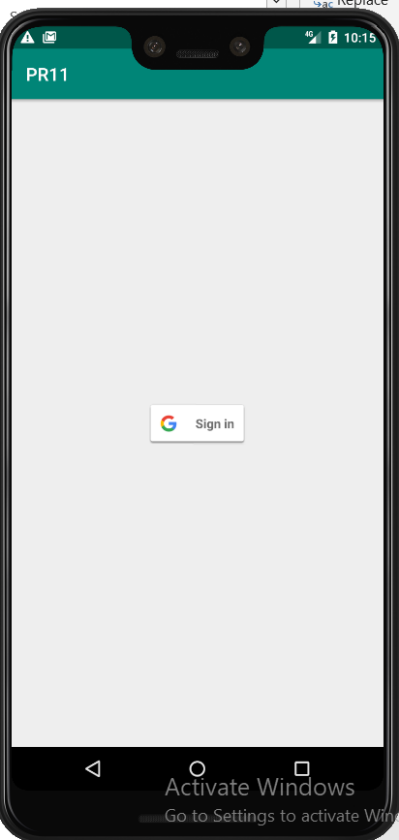
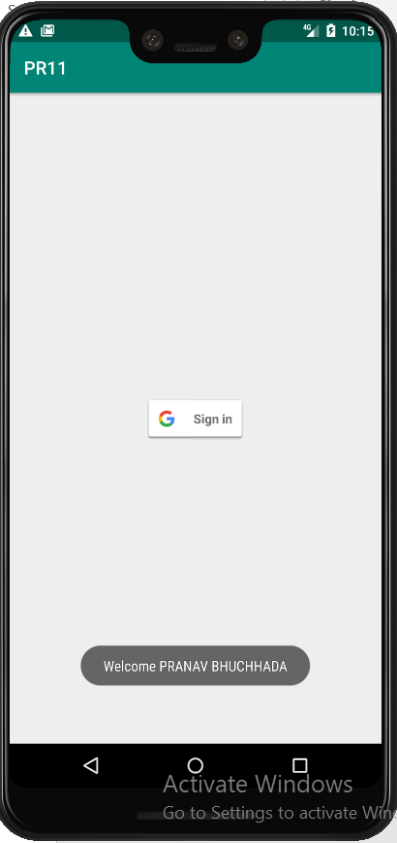
**OUTPUT:**

Figure 1 Main Activity with sign button

Figure 2 Displaying Display name of the Signed in Option

**LATEST APPLICATIONS:**

M

**LEARNING OUTCOME:**

1. Working with android different kind of layouts and resource files.
2. Working with android menu layout.