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: AppAuth library:

AppAuth is a client SDK for native apps to authenticate and authorize end-users using OAuth 2.0 and OpenID Connect. It strives to directly map the requests and responses of those specifications, while following the idiomatic style of the implementation language.

Google Single Sign On (SSO):

Google Sign-In manages the OAuth 2.0 flow and token lifecycle, simplifying your integration with Google APIs. A user always has the option to revoke access to an application at any time.

CODE:

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: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>
```

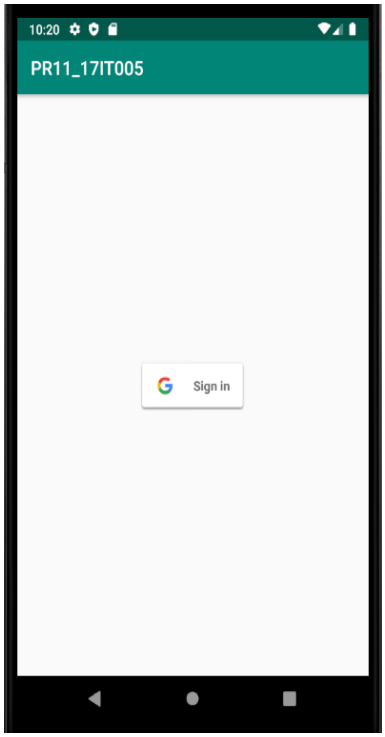
MainActivity.java

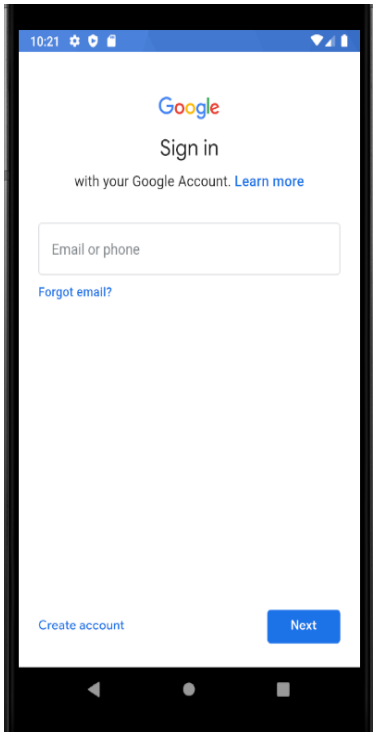
```
package com.example.pr11_17it005;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Toast;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import com.google.android.gms.auth.api.signin.GoogleSignIn;
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;
```

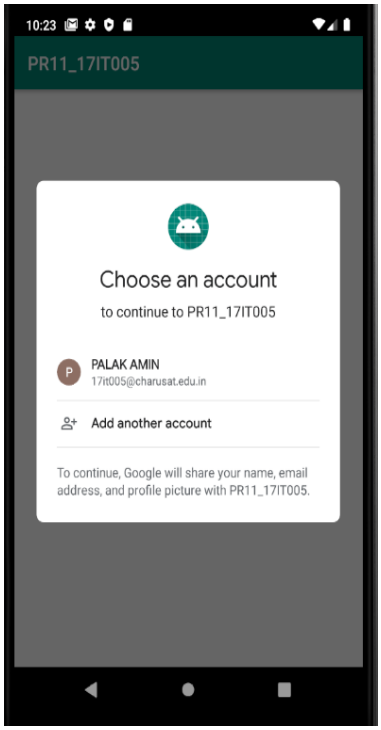
```
public class MainActivity extends AppCompatActivity {
   private GoogleSignInClient mGoogleSignInClient;
   private GoogleSignInOptions gso;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      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);
        }
    }
}
```

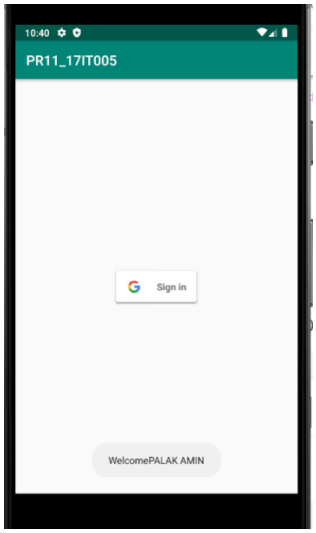
OUTPUT:







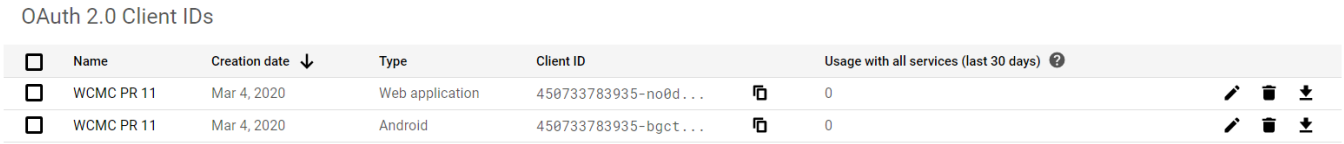
Second image asks to sign in when we first time login into some new account, and when we open the app next time we have option as in third picture.



Once we are logged in, the toast message appears.

LATEST APPLICATIONS: Almost every application uses google sign on module in their application. Whenever we authenticate ourselves using google or facebook we use the feature. All the famous applications like Edmodo, coursera, codecademy, etc. use the same modules.

LEARNING OUTCOME: Here to implement google sign on feature we need to include appAuth dependency. Also once we are done with the coding part, we need to create a client o AppAuth for our application for it to be able to provide login through console.developers.google.com.



This way we include client id and only after that we are able to log into our account through this application.