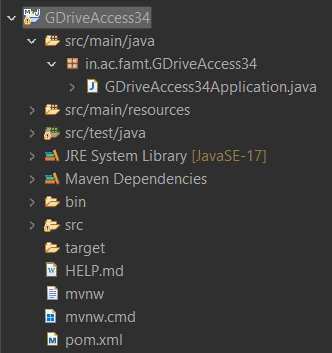
**Practical No.06**

**Implementation of Cloud Computing Services**

**Q.1 Write a java program to access the files from your Google drive account and read and write the file contents from your program.**

****

**Code:**

**GDriveAccess34Application.java**

package in.ac.famt.GDriveAccess34;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class GDriveAccess34Application {

public static void main(String[] args) {

SpringApplication.run(GDriveAccess34Application.class, args);

}

}

**GDriveAccess34ApplicationTests.java**

package in.ac.famt.GDriveAccess34;

import com.google.api.client.auth.oauth2.Credential;

import com.google.api.client.extensions.java6.auth.oauth2.AuthorizationCodeInstalledApp;

import com.google.api.client.extensions.jetty.auth.oauth2.LocalServerReceiver;

import com.google.api.client.googleapis.auth.oauth2.GoogleAuthorizationCodeFlow;

import com.google.api.client.googleapis.auth.oauth2.GoogleClientSecrets;

import com.google.api.client.googleapis.javanet.GoogleNetHttpTransport;

import com.google.api.client.http.AbstractInputStreamContent;

import com.google.api.client.http.InputStreamContent;

import com.google.api.client.http.javanet.NetHttpTransport;

import com.google.api.client.json.JsonFactory;

import com.google.api.client.json.jackson2.JacksonFactory;

import com.google.api.client.util.store.FileDataStoreFactory;

import com.google.api.services.drive.Drive;

import com.google.api.services.drive.DriveScopes;

import com.google.api.services.drive.model.File;

import com.google.api.services.drive.model.FileList;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.security.GeneralSecurityException;

import java.util.Collections;

import java.util.List;

//@SpringBootApplication

public class GDriveAccess34ApplicationTests {

private static final JsonFactory JSON\_FACTORY = JacksonFactory.getDefaultInstance();

// Directory to store user credentials for this application.

//private static final java.io.File CREDENTIALS\_FOLDER = new java.io.File(System.getProperty("user.home"), "credentials");

private static final java.io.File CREDENTIALS\_FOLDER = new java.io.File("Z:\\MCA-DSCC");

// Global instance of the scopes required by this program.

private static final List<String> SCOPES = Collections.singletonList(DriveScopes.DRIVE);

//https://developers.google.com/resources/api-libraries/documentation/drive/v2/java/latest/com/google/api/services/drive/DriveScopes.html

private static Credential getCredentials(final NetHttpTransport HTTP\_TRANSPORT) throws IOException {

java.io.File clientSecretFilePath = new java.io.File("Z:\\MCA-DSCC\\credits.json");

if (!clientSecretFilePath.exists()) {

throw new FileNotFoundException("Please copy credentials.");

}

// Load client secrets.

InputStream in = new FileInputStream(clientSecretFilePath);

GoogleClientSecrets clientSecrets = GoogleClientSecrets.load(JSON\_FACTORY, new InputStreamReader(in));

// Build flow and trigger user authorization request.

GoogleAuthorizationCodeFlow flow = new GoogleAuthorizationCodeFlow.Builder(HTTP\_TRANSPORT, JSON\_FACTORY,

clientSecrets, SCOPES).setDataStoreFactory(new FileDataStoreFactory(CREDENTIALS\_FOLDER))

.setAccessType("offline").build();

//System.out.println("Flow info - " + flow.toString());

return new AuthorizationCodeInstalledApp(flow, new LocalServerReceiver()).authorize("user");

}

public static void main(String[] args)throws IOException, GeneralSecurityException {

// 1: Build a new authorized API client service.

final NetHttpTransport HTTP\_TRANSPORT = GoogleNetHttpTransport.newTrustedTransport();

// 2: Read client\_secret.json file & create Credential object.

Credential credential = getCredentials(HTTP\_TRANSPORT);

// 3: Create Google Drive Service.

Drive service = new Drive.Builder(HTTP\_TRANSPORT, JSON\_FACTORY, credential).setApplicationName("GDrive Access").build();

System.out.println("----" + service.getApplicationName() + "----");

// Print the names and IDs for up to 10 files.

FileList result = service.files().list().setPageSize(2).setFields("nextPageToken, files(id, name)").execute();

String lastFile = "";

List<File> files = result.getFiles();

if (files == null || files.isEmpty()) {

System.out.println("No files found.");

} else {

System.out.println("Files:");

for (File file : files) {

System.out.printf("%s (%s)\n", file.getName(), file.getId());

lastFile = file.getId();

}

}

//Create Folder on Google Drive

File fileMetadata = new File();

fileMetadata.setName("MyFolderData");

fileMetadata.setMimeType("application/vnd.google-apps.folder");

//fileMetadata.setParents(folderIdParent);

File file = service.files().create(fileMetadata).setFields("id, name").execute();

if(file != null)

System.out.println("Folder Created..");

//Creating a file on GDrive

java.io.File uploadFileContent = new java.io.File("Z:\\MCA-DSCC\\textUpload.txt");

String contentType = "text/plain";

AbstractInputStreamContent uploadStreamContent = new InputStreamContent(contentType,new FileInputStream(uploadFileContent));

fileMetadata = new File();

fileMetadata.setName("MyGDriveJavaFile.txt");

file = service.files().create(fileMetadata, uploadStreamContent).setFields("id, webContentLink, webViewLink, parents").execute();

if(file != null) {

System.out.println("File Created..");

System.out.println("WebContentLink: " + file.getWebContentLink() );

System.out.println("WebViewLink: " + file.getWebViewLink() );

}

}

}

**Pom.xml**

<!-- https://mvnrepository.com/artifact/com.google.apis/google-api-services-drive -->

<dependency>

<groupId>com.google.apis</groupId>

<artifactId>google-api-services-drive</artifactId>

<version>v3-rev103-1.23.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/com.google.api-client/google-api-client -->

<dependency>

<groupId>com.google.api-client</groupId>

<artifactId>google-api-client</artifactId>

<version>1.23.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/com.google.oauth-client/google-oauth-client-jetty -->

<dependency>

<groupId>com.google.oauth-client</groupId>

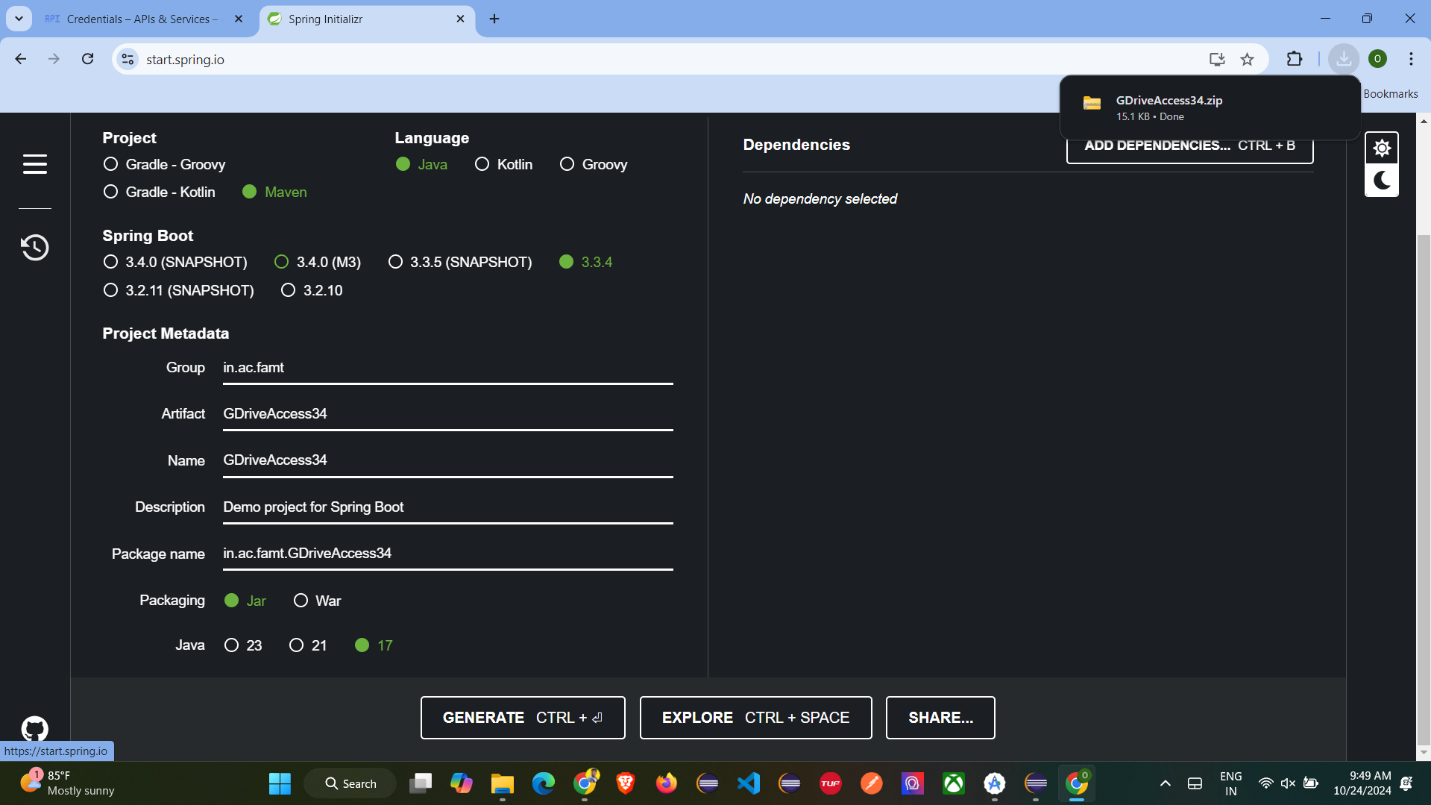
<artifactId>google-oauth-client-jetty</artifactId>

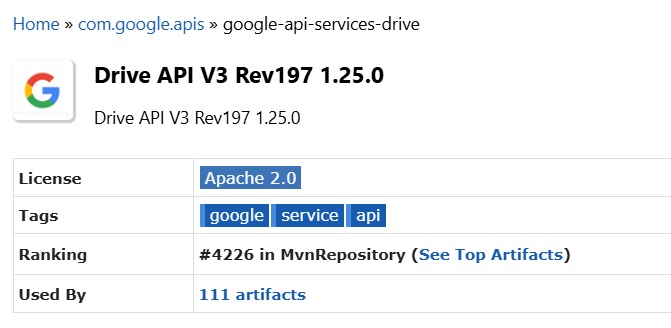
<version>1.23.0</version>

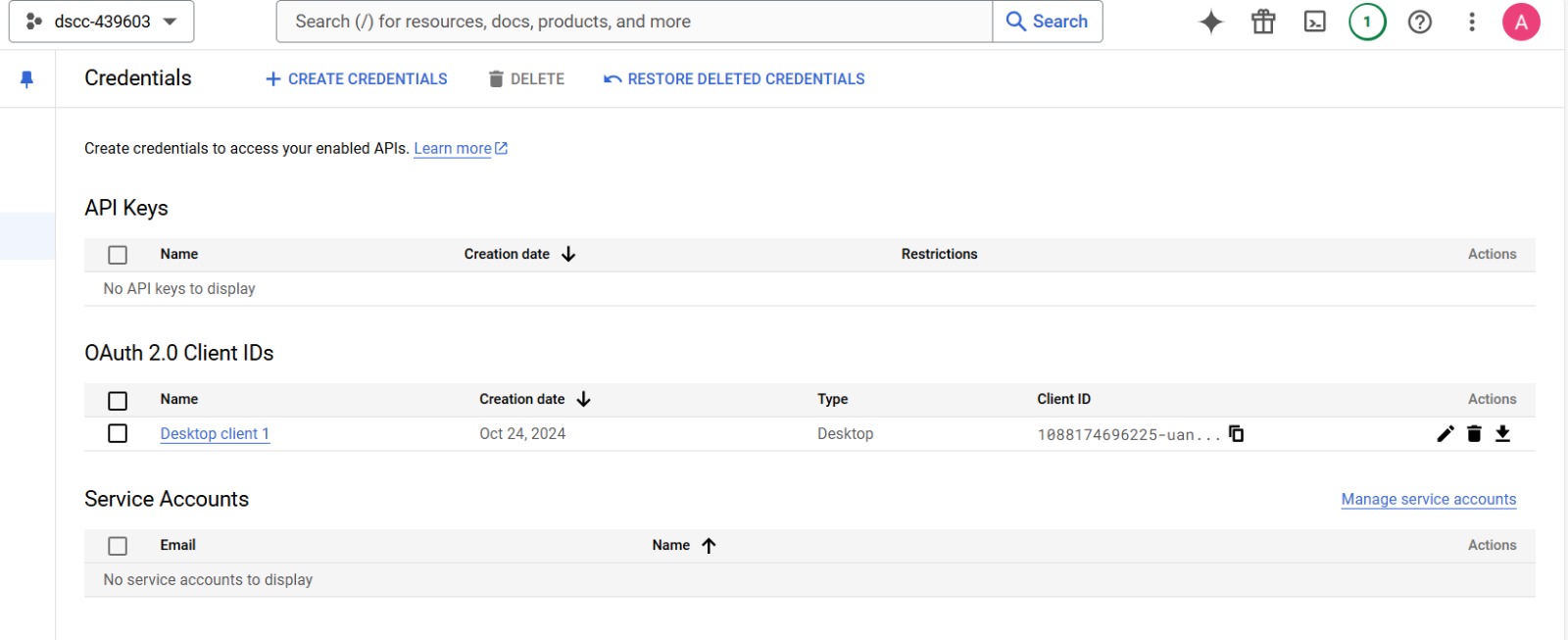
</dependency>

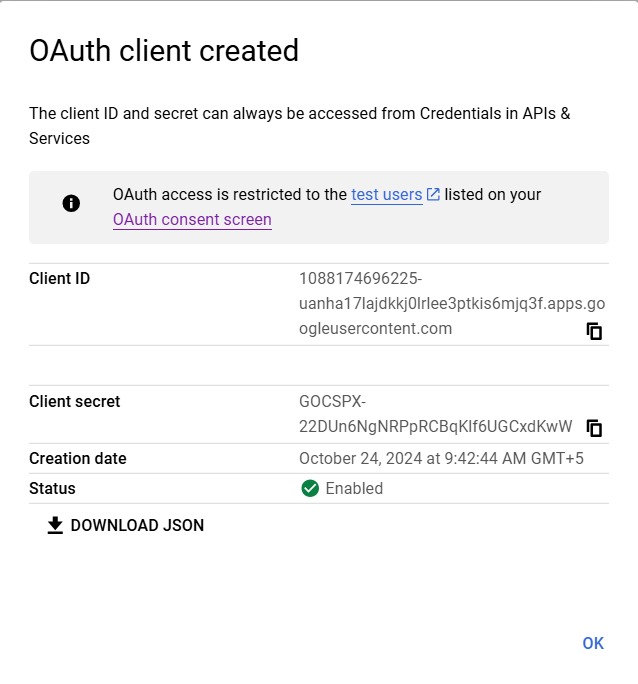
**Output:**

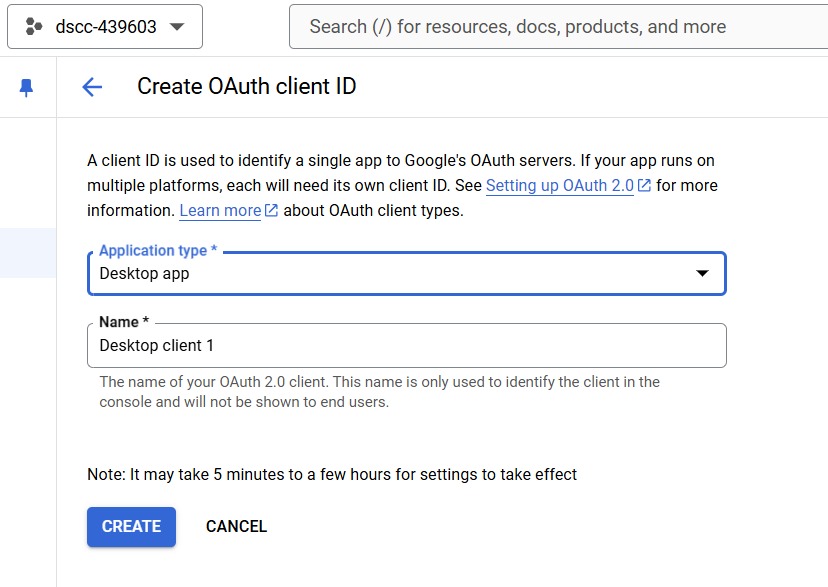
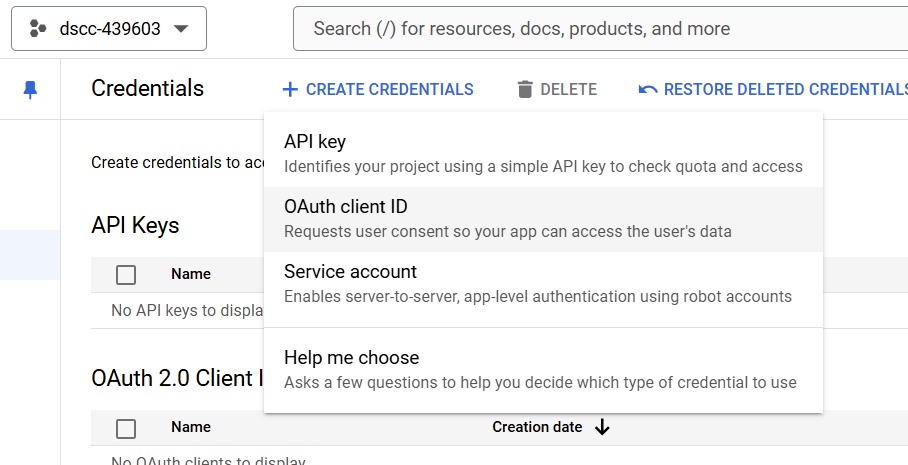
**Download Spring Boot File**

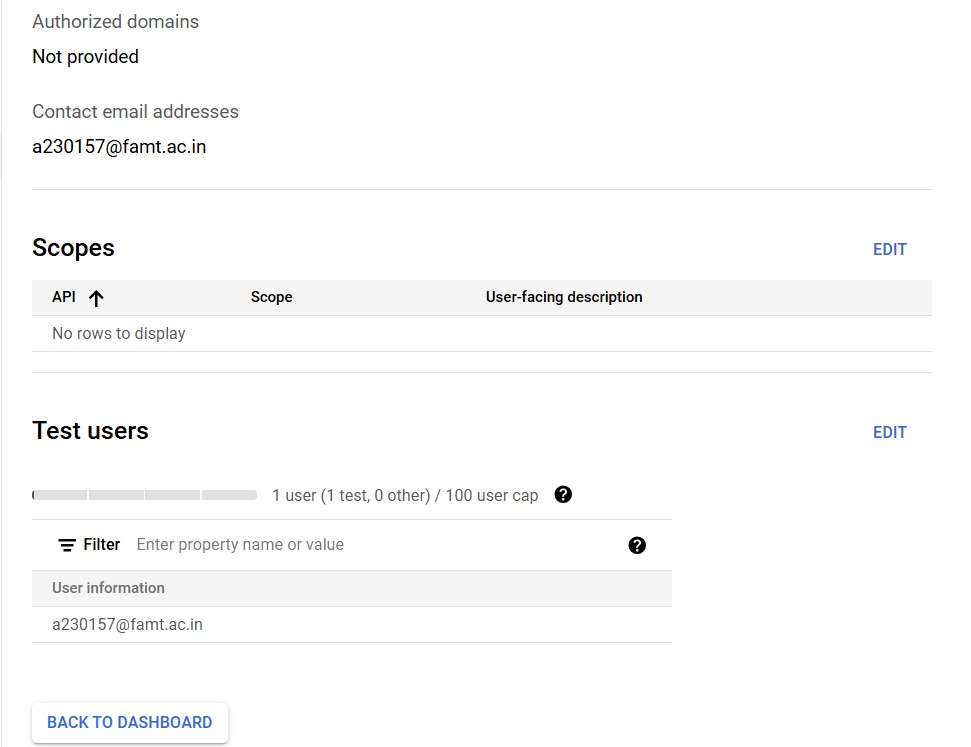
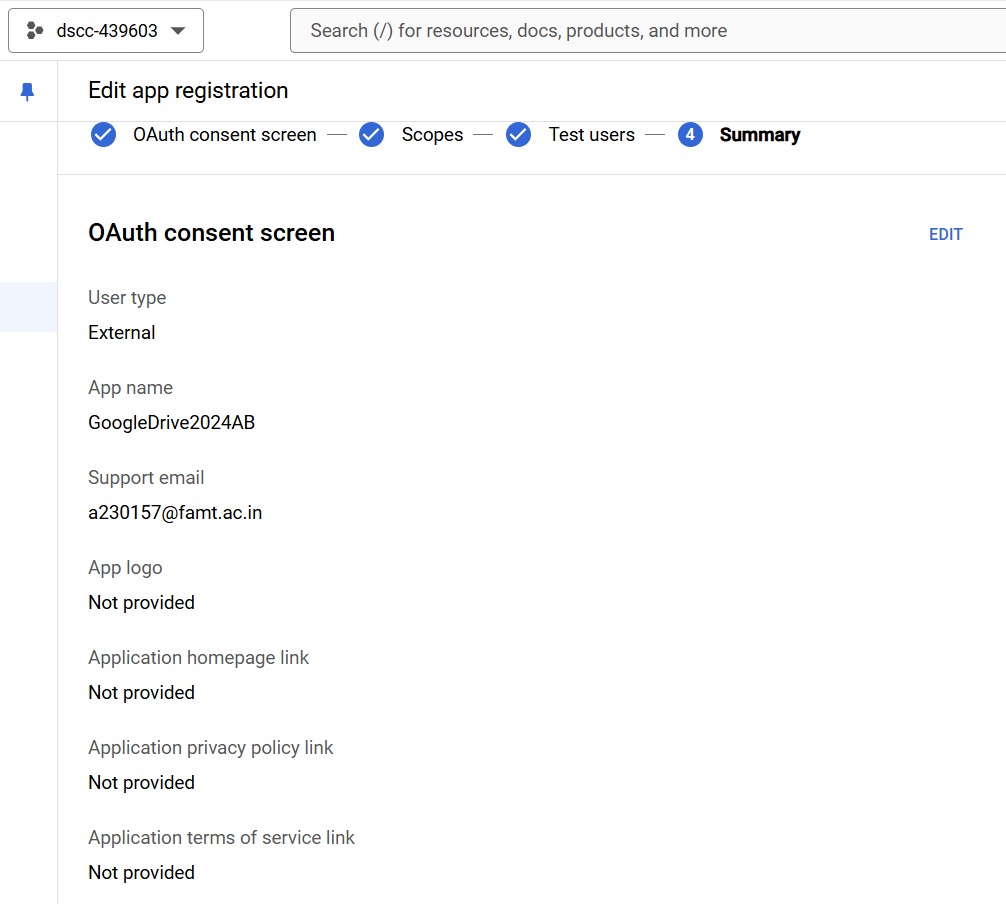


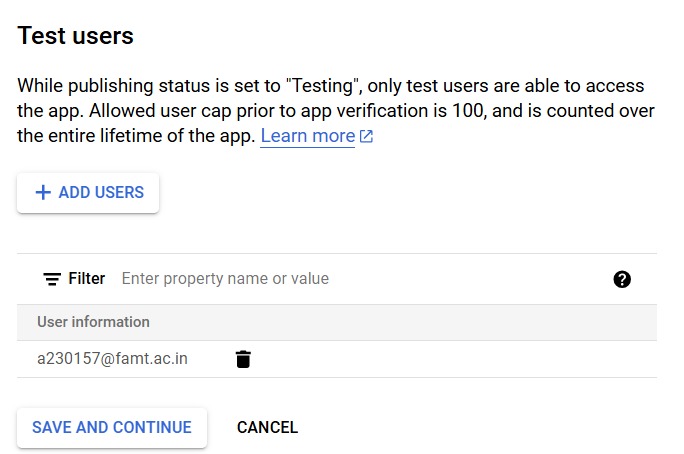
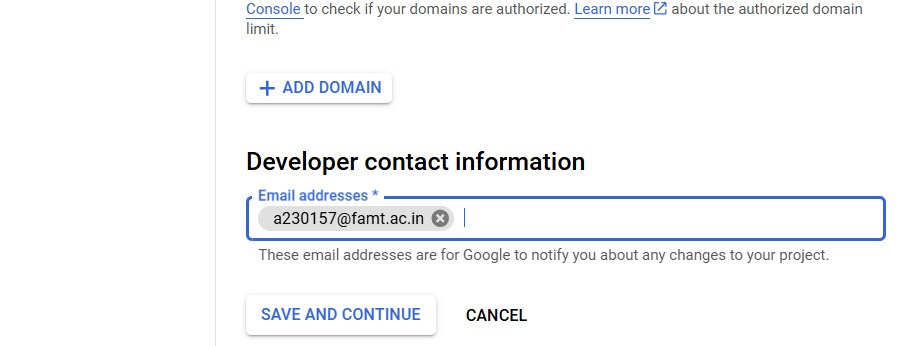
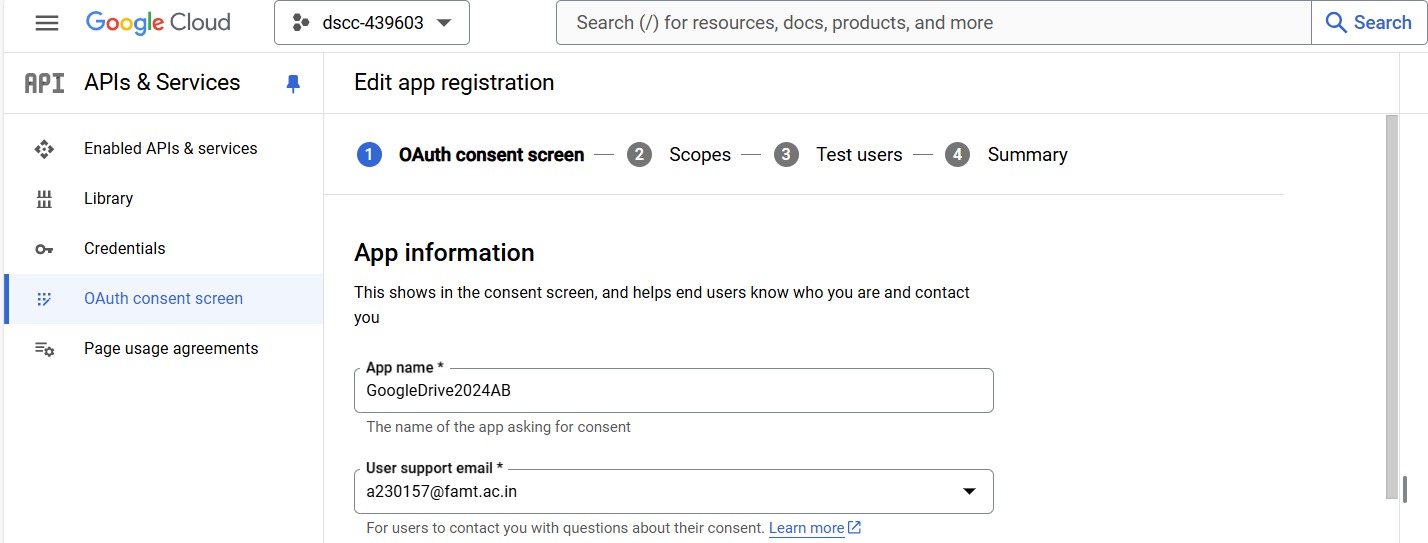


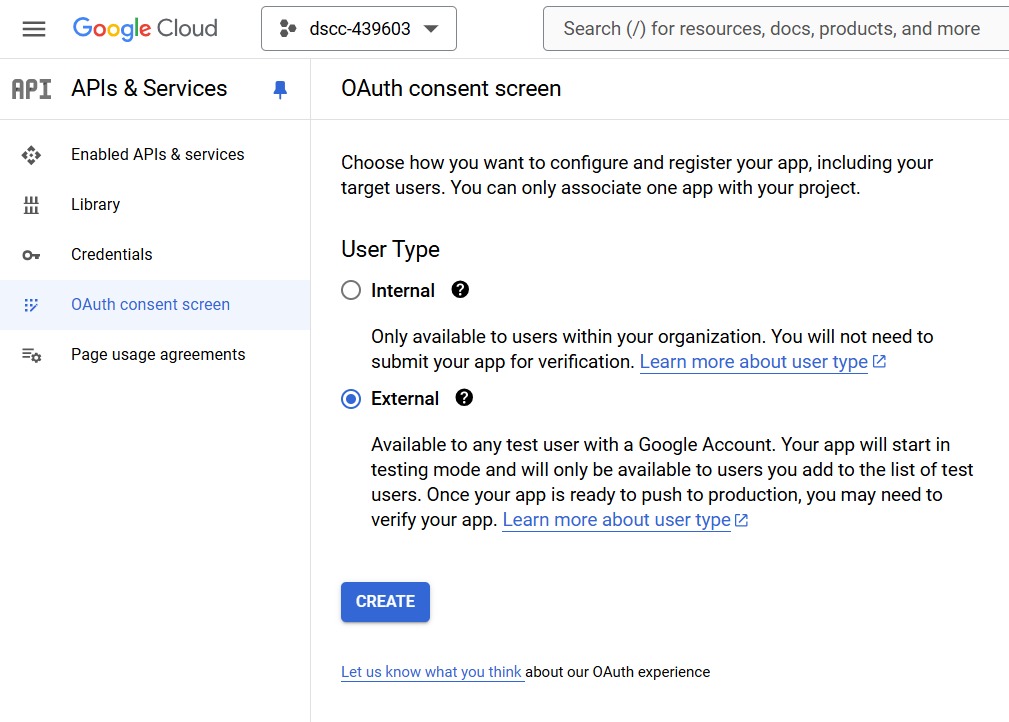
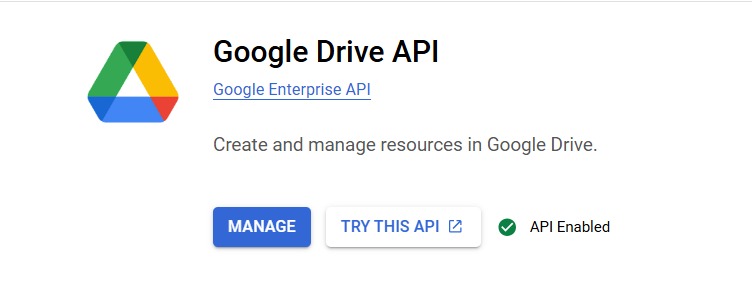
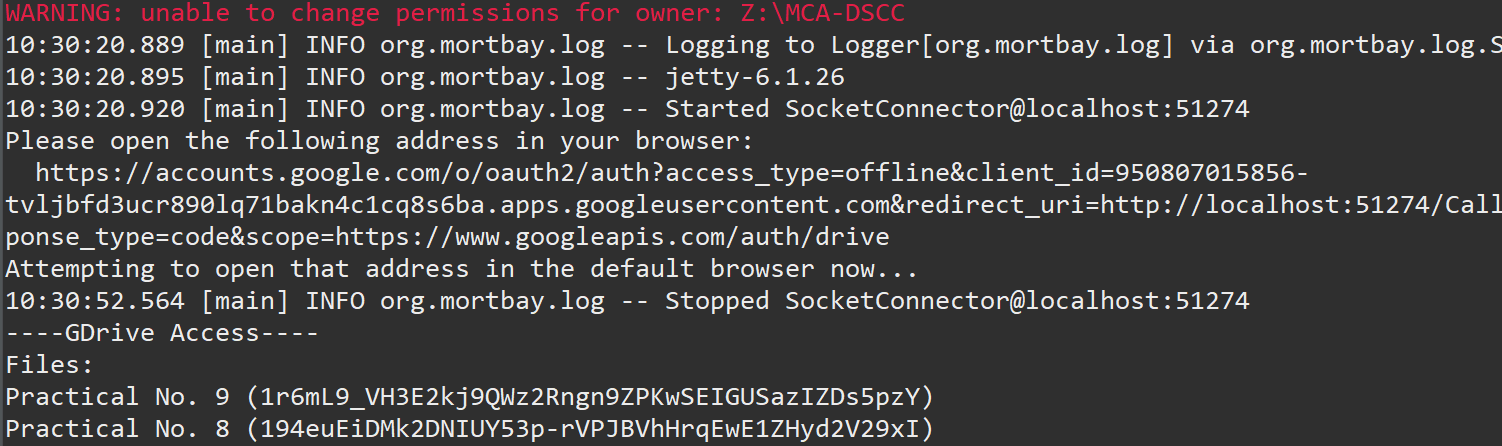


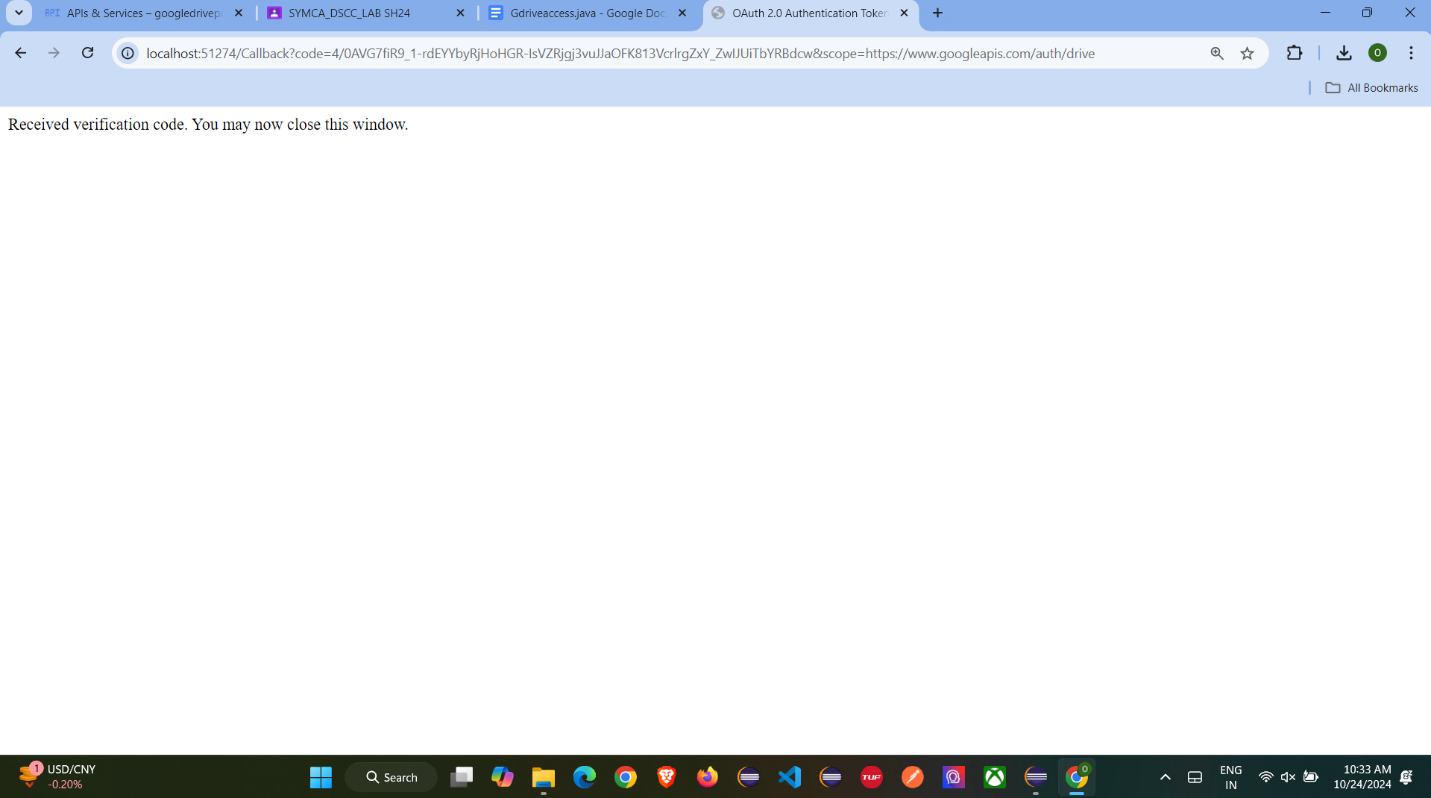


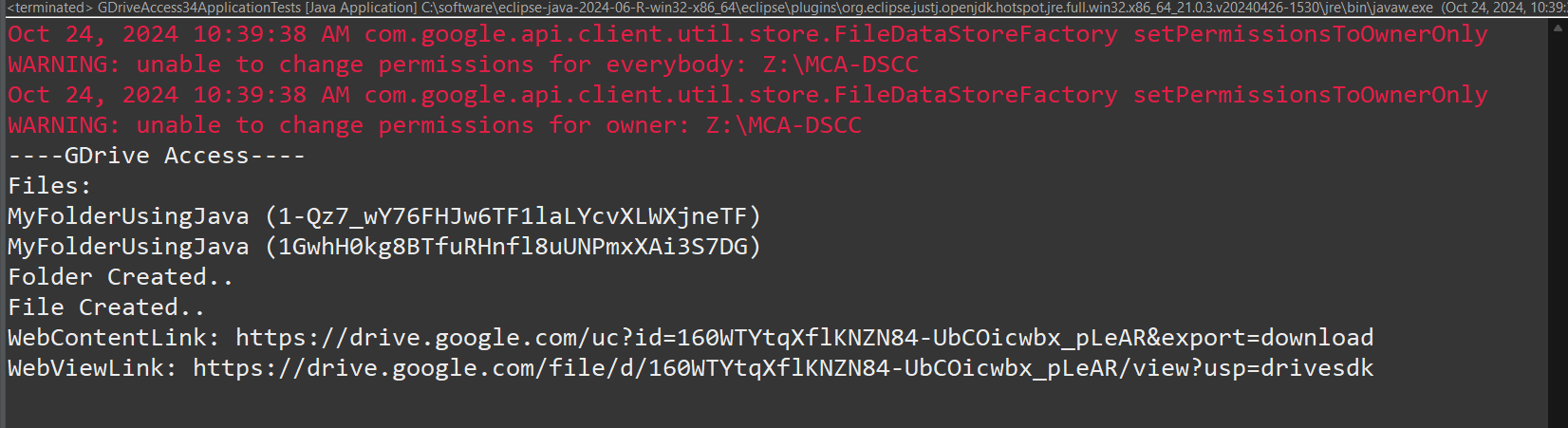
 

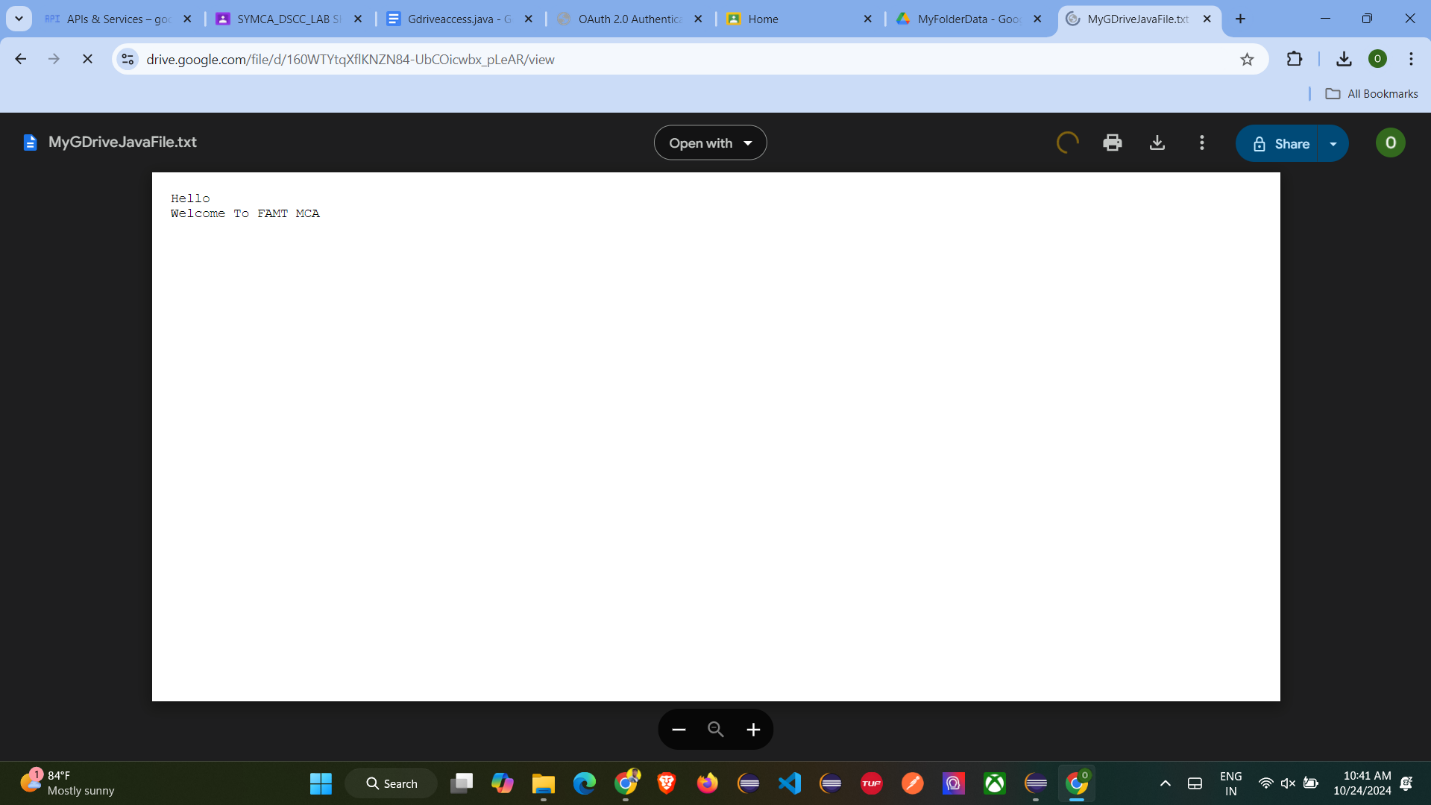
 

****

****

****