Google Oauth Login

Registering Application with Google

Inorder to implement Google singin for our application we have to register our application with Google.

Steps to register application with Google:

Step1: Visit Google API Console and create a new project.

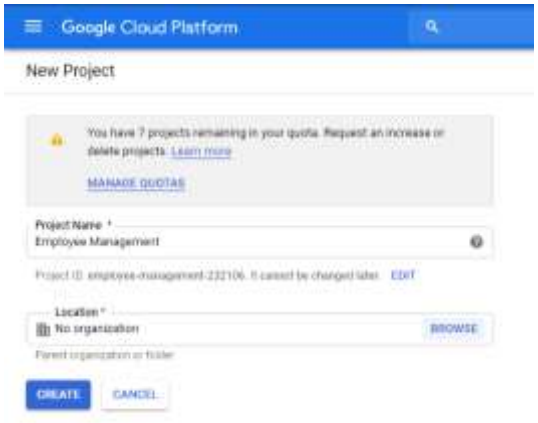

Fig: Creating a new project

Step2: Create credentials for the application created. Navigate to *credentials->create credentials->OAuth clientid->web application*

Step3: Fill the form:

Application type: web application

Name: Employee Management

Restriction. Authorized JavaScript origins: http://localhost:8080

Restriction. Authorized redirect URIs: http://localhost:8080/login

And click on create. Thus u will be prompted with ClientId and ClientSecret.

Step4: Now we have to enable few google application's for our Employee Management Application. Navigate to Library, Search for Gmail and Click on Enable. Similarly enable Analytics API and Google+ API

Add required Dependencies

Import EmployeeManagement Application perform a testrun.

Add below dependencies into pom.

```
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-core</artifactId>
</dependency>
<dependency>
	<groupId>org.springframework.security.oauth</groupId>
	<artifactId>spring-security-oauth2</artifactId>
</dependency>
```

Add Auth Server properties

```
security.oauth2.client.clientId= <CLIENT_ID>
security.oauth2.client.clientSecret= <CLIENT_SECRET>
security.oauth2.client.accessTokenUri= https://www.googleapis.com/oauth2/v4/token
security.oauth2.client.userAuthorizationUri=
https://accounts.google.com/o/oauth2/v2/auth
security.oauth2.client.clientAuthenticationScheme= form
security.oauth2.client.scope= openid, email, profile
security.oauth2.resource.userInfoUri= https://www.googleapis.com/oauth2/v3/userinfo
security.oauth2.resource.preferTokenInfo= true
```

Replace CLIENT_ID and CLIENT_SECRET with those which we got on application registration

Add index page to application

Step1: Create a 'static' folder on root of the project.

Step2: Create html file and name it as index.hmtl

Step3: Add the below tag under body section. By this way we provide a link to login via Google.

```
<a href="user"> Click here for Google Login</a>
```

Step4: Save the file.

Create web security config class

Step1: Create a class WebSecurityConfig which extends WebSecurityConfigurerAdapter.

Step2: Annotate the method with @Configuration and @EnableOAuth2Sso annotations.

Step3: Override `configure(HttpSecurity http)` method and add below lines

```
http.csrf()
	.disable()
```

```
                .antMatcher("/**")
                        .authorizeRequests()
                .antMatchers("/", "/index.html")
                        .permitAll()
                .anyRequest()
                        .authenticated();
```

Description:

- CSRF is disabled for all the incoming requests.
- Authenticating and Authorizing all the requests except "/index.html".

Step4: Save the class file and build the application.

Run and Test the application

Step1: Run the application. Project -> run as -> spring boot application

Step2: Hit http://localhost:8080 on browser, a page will be loaded with the link to login via google.

Step3: Click the link and you will be redirect to Google Login page. Login using your Google credentials.

Step4: After Successful sign in via google you will be redirected to the page that shows the Principal object which has user details.

Step5: Now try to access EmployeeManagement Requests with google authentication.

Fig: Response after google authentication