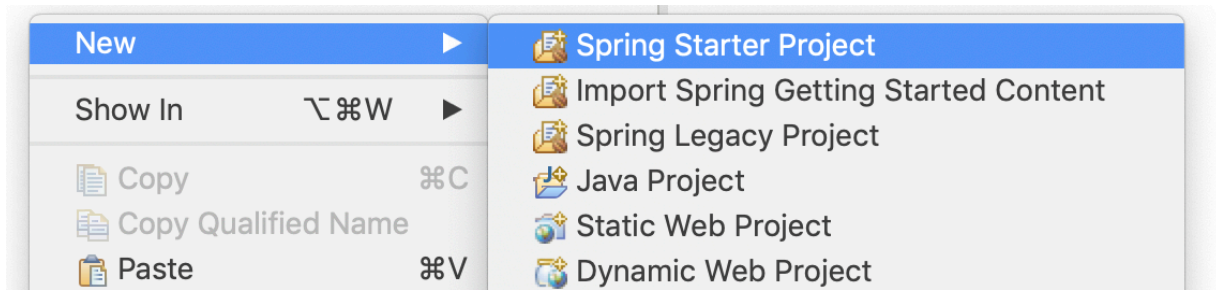
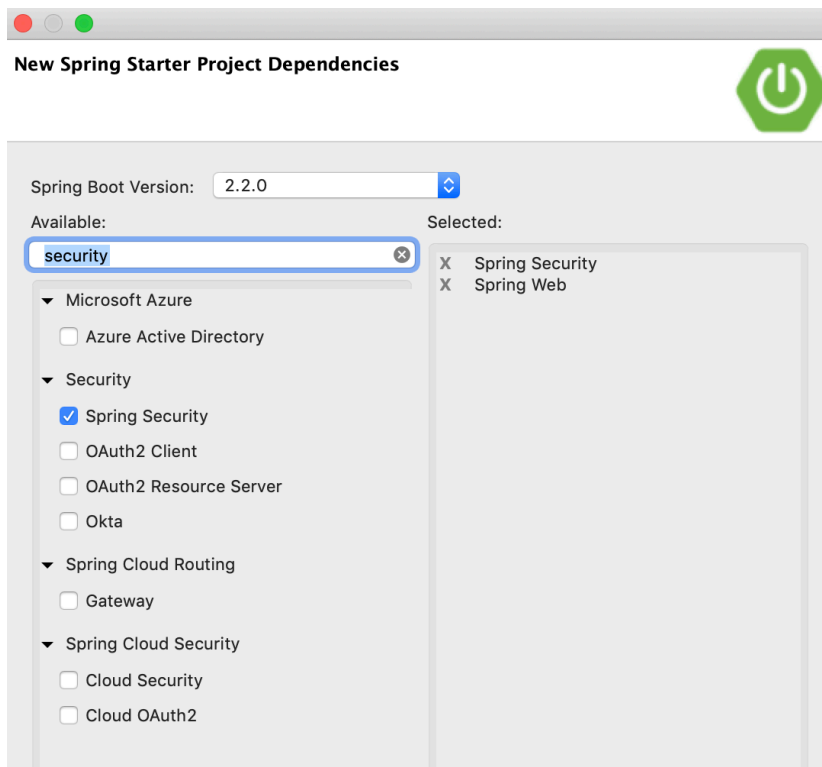


Create a Spring Boot Application to Authenticate using OAuth via Google Provider

1. Create a Spring Boot application using Spring Boot Starter



2. Add **Web & Security** modules



3. Add the following dependencies in **pom.xml** for **OAuth2** integration

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-oauth2-client</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-oauth2-jose</artifactId>
</dependency>

<!-- Other DEPENDENCIES -->
<dependency>
    <groupId>org.apache.tomcat.embed</groupId>
    <artifactId>tomcat-embed-jasper</artifactId>
</dependency>
```

4. Add the following properties in **application.properties** and observe the **MAGIC** when you RUN the APP

```
#Google app details
spring.security.oauth2.client.registration.google.client-id=<your client id>
spring.security.oauth2.client.registration.google.client-secret=<your client secret>

#Facebook app details
spring.security.oauth2.client.registration.facebook.client-id=<your client id>
spring.security.oauth2.client.registration.facebook.client-secret=<your client secret>

#Github app details
spring.security.oauth2.client.registration.github.client-id=<your client id>
spring.security.oauth2.client.registration.github.client-secret=<your client secret>

#Spring MVC details
spring.mvc.view.prefix: /WEB-INF/views/
spring.mvc.view.suffix: .jsp
spring.mvc.static-path-pattern=/resources/**
```

5. RUN the APP

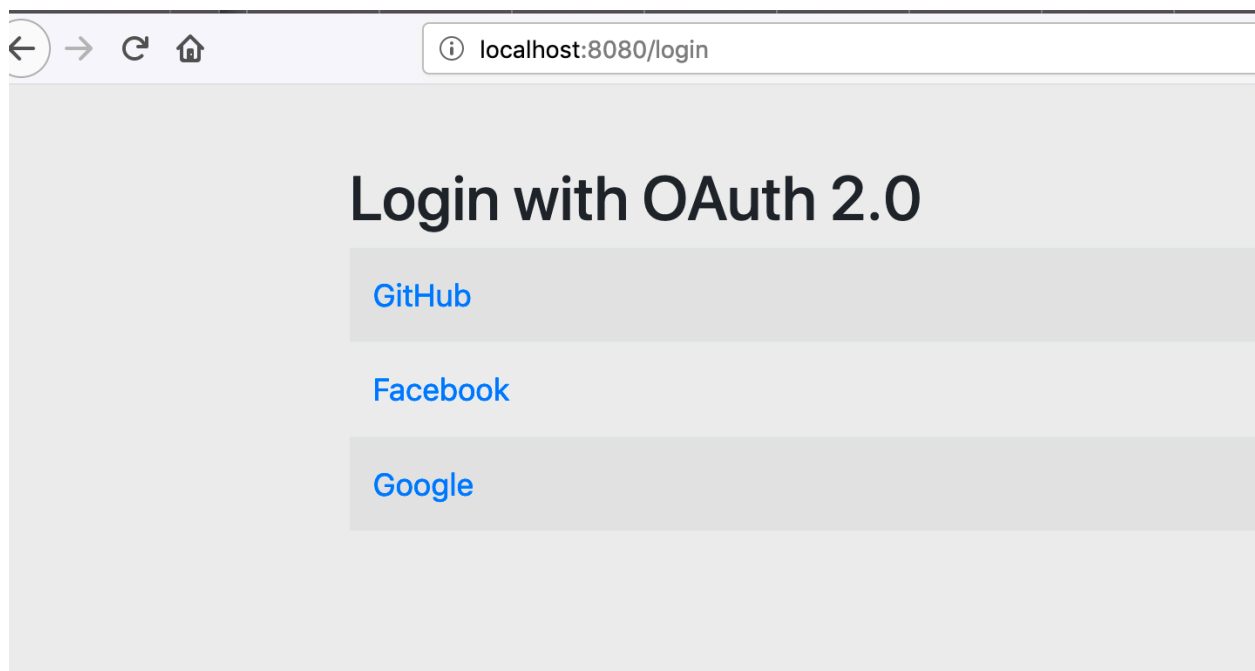
SpringBoot0authGoogleDemoApplication.java

→ **Run As**

→ **Spring Boot App**

Here, we declare **two properties** for each provider. We will be implementing the **Google provider**, but you can add any number of providers.

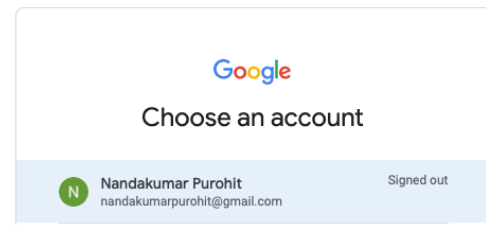
Just adding these properties will create more magic, and your login page will suddenly change to the following:



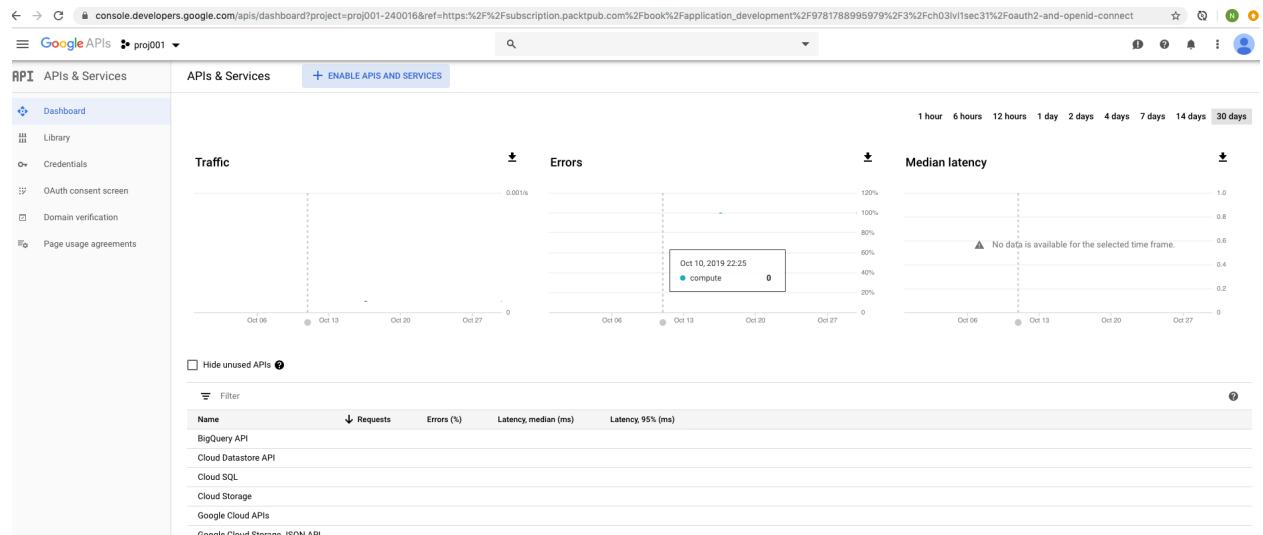
Google Provider setup

6. We will be using Google as our provider in this example. Navigate to <https://console.developers.google.com/> and perform the following steps:

← → ↻ accounts.google.com/ServiceLogin/signinchooser?service=cloudconsole&passive=1209600&osid=1&continue=https%3A%2F%2Fconsole.developers.google.cc



7. Login with your google credentials and you will see the following google console page



8. Click on Project Name next to Google APIs icon



9. Click on **NEW PROJECT** button

Select a project

NEW PROJECT

Search projects and folders

RECENT ALL

Name	ID
✓ proj001 ?	proj001-240016
test-project-001 ?	test-project-001-234600
sb-deploy ?	sb-deploy

10. Enter the Project Name and click on **CREATE** button

New Project



You have 22 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)

[MANAGE QUOTAS](#)

Project name *

springboot-withoauth



Project ID: springboot-withoauth. It cannot be changed later. [EDIT](#)

Location *



No organisation

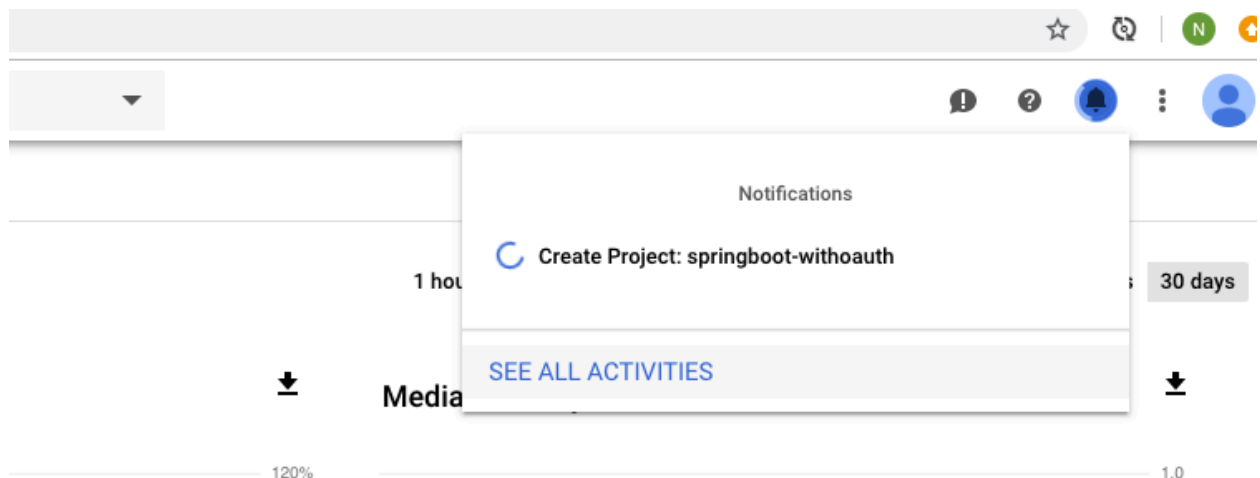
[BROWSE](#)

Parent organisation or folder

CREATE

CANCEL

Project creation in progress



The screenshot shows a web application interface. At the top, there is a navigation bar with a star icon, a refresh icon, and a user profile icon. Below the navigation bar, there is a dropdown menu. In the center, a notification box is displayed with the title "Notifications". The notification content says "Create Project: springboot-withoauth" with a circular progress indicator. Below the notification, there is a button labeled "SEE ALL ACTIVITIES". The background of the application shows a table with columns for "Media" and "30 days".

11. Click on the Project Name next to Google API link and select the newly created Project **springboot-withoauth**:

Select a project

NEW PROJECT

RECENT

ALL

Name	ID
springboot-withoauth ?	springboot-withoauth
proj001 ?	proj001-240016
test-project-001 ?	test-project-001-234600
sb-deploy ?	sb-deploy

12. You will see this default screen for the new project

Google APIs springboot-withoauth ▼

APIs & Services

Dashboard

Library

Credentials

OAuth consent screen

Domain verification

Page usage agreements

APIs & Services

[+ ENABLE APIS AND SERVICES](#)

You don't have any APIs available to use yet. To get started, click 'Enable APIs and services' or go to the [API library](#)

13. Click on the **Credentials** link on the side menu

APIs
Credentials

You need credentials to access APIs. [Enable the APIs that you plan to use](#) and then create the credentials that they require. Depending on the API, you need an API key, a service account or an OAuth 2.0 client ID. For more information, see the [authentication documentation](#).

Create credentials ▾

APIs
Credentials

You need credentials to access APIs. [Enable the APIs that you plan to use](#) and then create the credentials that they require. Depending on the API, you need an API key, a service account or an OAuth 2.0 client ID. For more information, see the [authentication documentation](#).

Create credentials ▾

API key
Identifies your project using a simple API key to check quota and access.

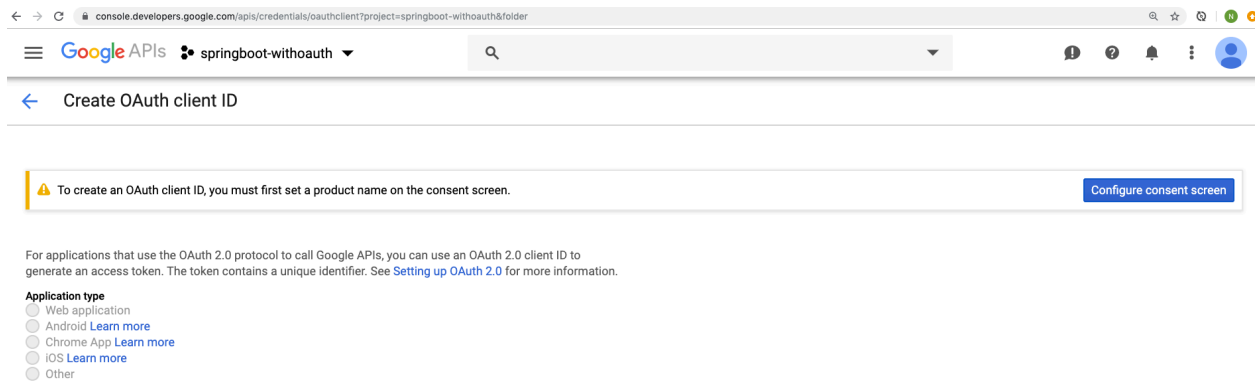
OAuth client ID
Requests user consent so your app can access the user's data.

Service account key
Enables server-to-server, app-level authentication using robot accounts.

Help me choose
Asks a few questions to help you decide which type of credential to use

14. From the drop-down menu, click on **OAuth client ID**. This will navigate you to the page shown in the following screenshot. Please note that the **Application type** radio group will be disabled at this stage:

Click on the **Configure Consent Screen** button on the following screen



15. Enter the relevant details (*leave the optional fields out while filling in the form*), as shown in the preceding figure, and click on the **Save** button.

← → ↻ console.developers.google.com/apis/credentials/consent?createClient&project=springboot-withoauth&folder=&duration=P1D

Google APIs •.springboot-withoauth 🔍

APIs & Services


- Dashboard
- Library
- Credentials
- OAuth consent screen**
- Domain verification
- Page usage agreements

OAuth consent screen

springboot-withoauth

Application logo ⓘ
An image on the consent screen that will help users recognise your app

Local file for upload Browse



Support email ⓘ
Shown on the consent screen for user support

nandakumarpurohit@gmail.com

Scopes for Google APIs
Scopes allow your application to access your user's private data. [Learn more](#)
If you add a sensitive scope, such as scopes that give you full access to Calendar or Drive, Google will verify your consent screen before it's published.

email

profile

openid

Add scope

Authorised domains ⓘ
To protect you and your users, Google only allows applications that authenticate using OAuth to use Authorized Domains. Your applications' links must be hosted on Authorized Domains. [Learn more](#)

example.com

Type in the domain and press Enter to add it

Application Homepage link
Shown on the consent screen. Must be hosted on an Authorized Domain.

https:// or http://

Application Privacy Policy link
Shown on the consent screen. Must be hosted on an Authorized Domain.

https:// or http://

Application Terms of Service link (Optional)
Shown on the consent screen. Must be hosted on an Authorized Domain.

https:// or http://

Save Submit for verification Cancel

Verification is required if your app is marked as **Public** and at least one of the following is true:

- Your app uses a sensitive and/or restricted scope
- Your app displays an icon on its OAuth consent screen
- Your app has a large number of authorised domains
- You have made changes to a previously verified OAuth consent screen

The verification process may take up to several weeks, and you will receive email updates as it progresses. [Learn more](#) about verification.

Before your consent screen and application are verified by Google, you can still test your application with limitations. [Learn more](#) about how your app will behave before it's verified.

[Let us know what you think about our OAuth experience.](#)

OAuth grant limits

Token grant rate
Your current per minute token grant rate limit is 100 grants per minute. The per minute token grant rate resets every minute. Your current per day token grant rate limit is 10,000 grants per day. The per day token grant rate resets every day.

[Raise limit](#)

1 h 6 h 1 d 7 d 30 d

No data for this time interval

16. Enter the relevant details (leave the optional fields out while filling in the form), as shown in the preceding figure, and click on the **Save** button.

You will be navigated back to the page shown in the following figure.

This time the **Application Type** is enabled.

Name = *Spring Boot Google OAuth Client*

Authorised JavaScript origins = *http://localhost:8080*

Authorised redirect URIs = *http://localhost:8080/login/oauth/code/google*

17. Then, click on **Create** button

← Create OAuth client ID

For applications that use the OAuth 2.0 protocol to call Google APIs, you can use an OAuth 2.0 client ID to generate an access token. The token contains a unique identifier. See [Setting up OAuth 2.0](#) for more information.

Application type

- ☒ Web application
- ☐ Android [Learn more](#)
- ☐ Chrome App [Learn more](#)
- ☐ iOS [Learn more](#)
- ☐ Other

Name ?

Spring Boot Google OAuth Client

Restrictions

Enter JavaScript origins, redirect URIs or both [Learn more](#)

Origins and redirect domains must be added to the list of authorised domains in the [OAuth consent settings](#).

Authorised JavaScript origins

For use with requests from a browser. This is the origin URI of the client application. It cannot contain a wildcard (https://*.example.com) or a path (<https://example.com/subdir>). If you're using a non-standard port, you must include it in the origin URI.

<http://localhost:8080>



<https://www.example.com>

Type in the domain and press Enter to add it

Authorised redirect URIs

For use with requests from a web server. This is the path in your application that users are redirected to after they have authenticated with Google. The path will be appended with the authorisation code for access. Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

<http://localhost:8080/login/oauth2/code/google>



<https://www.example.com>

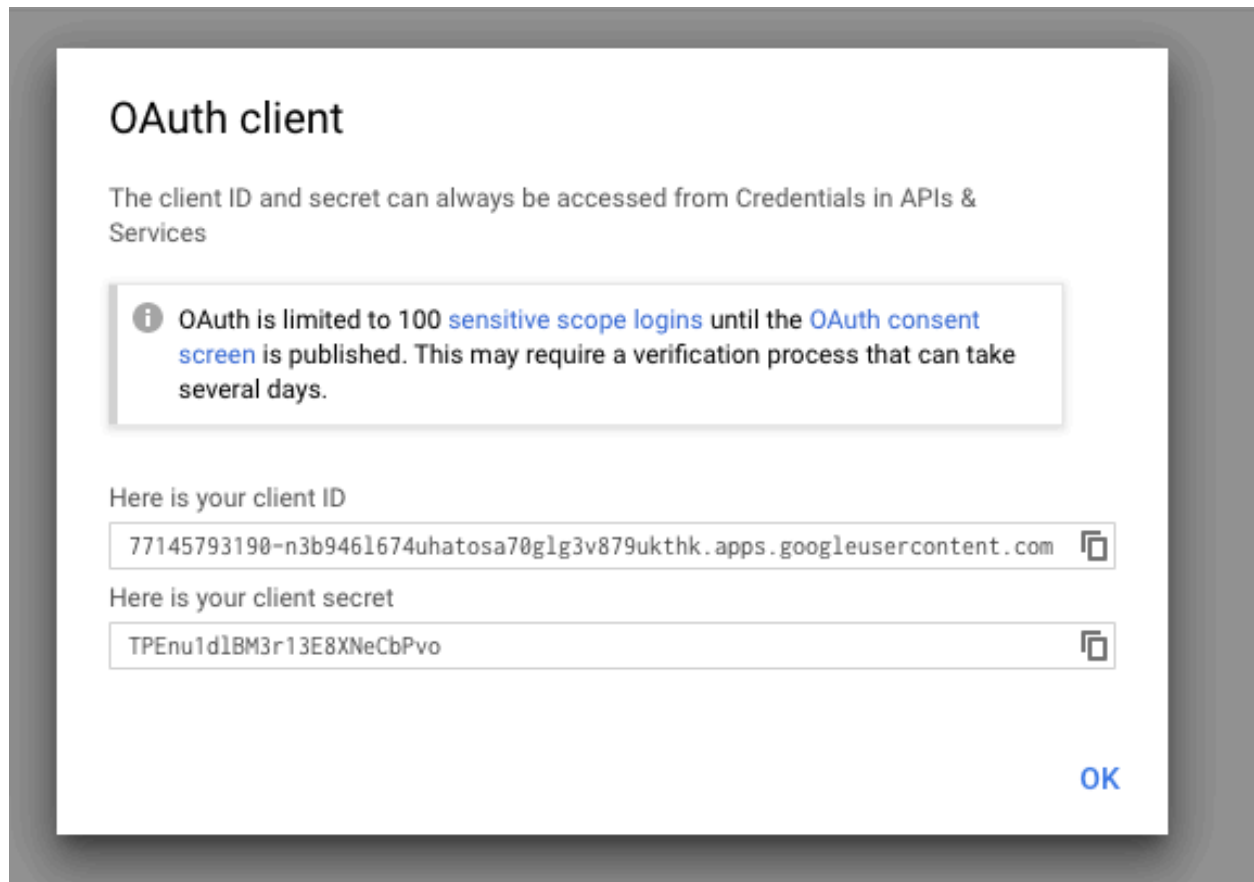
Type in the domain and press Enter to add it

Create

Cancel

18. You will see a following screen with a confirmation on OAuth **Client ID** and **Client Secret** code generation.

Click on **OK**



The screenshot shows the Google APIs console interface. The top navigation bar includes the Google APIs logo, the project name 'springboot-withoauth', a search bar, and user profile icons. The left sidebar contains a menu with 'APIs & Services' selected, and sub-items: Dashboard, Library, Credentials (highlighted), OAuth consent screen, Domain verification, and Page usage agreements. The main content area is titled 'Credentials' and features a 'Create credentials' button and a 'Delete' button. Below this, a message states: 'Create credentials to access your enabled APIs. For more information, see the [authentication documentation](#).' Under the heading 'OAuth 2.0 client IDs', there is a table with one entry:

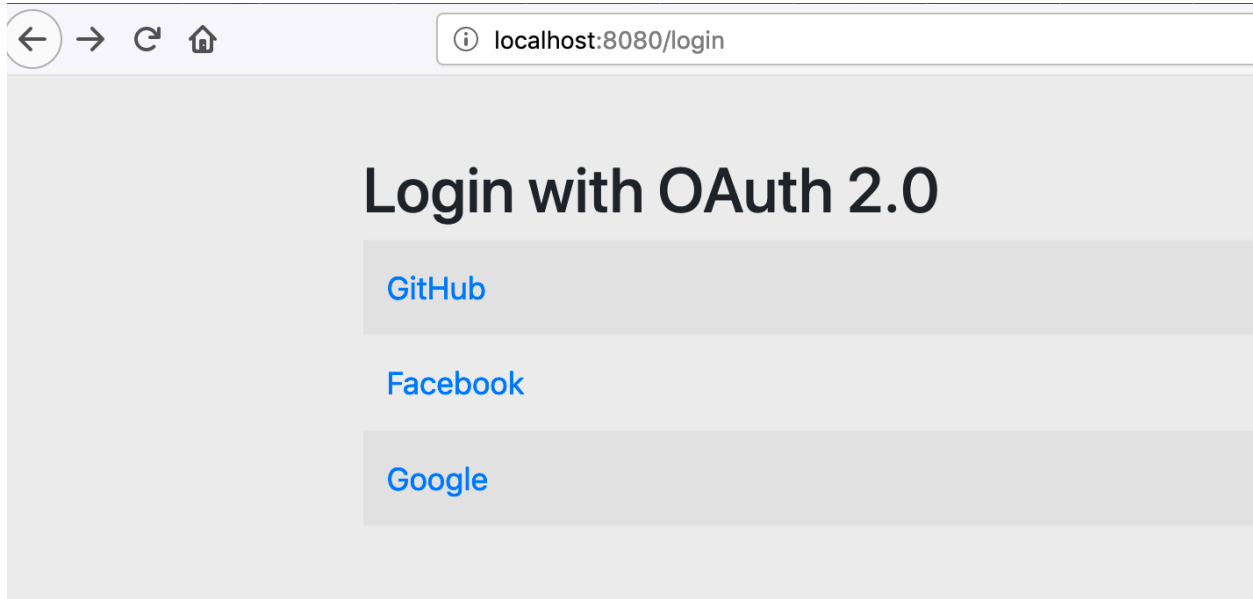
<input type="checkbox"/>	Name	Creation date	Type	Client ID	
<input type="checkbox"/>	Spring Boot Google OAuth Client	29 Oct 2019	Web application	77145793190-n3b9461674uhatos70glg3v879ukthk.apps.googleusercontent.com	

19 Copy and Paste these credentials in your **application.properties** for **Google** account

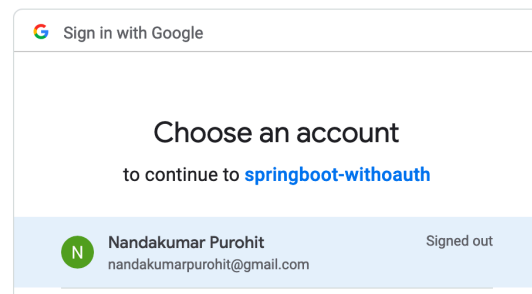
#Google app details

```
spring.security.oauth2.client.registration.google.client-id=<your-generated-client-id>
spring.security.oauth2.client.registration.google.client-secret=<your-generated-client-secret>
```

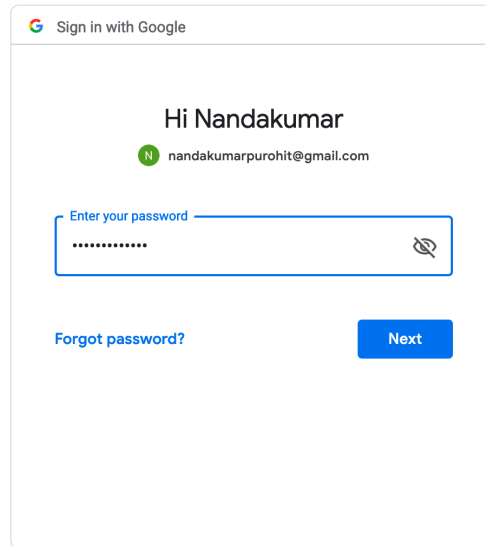
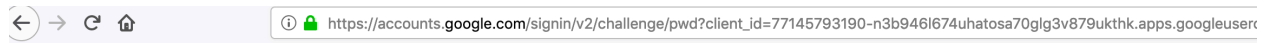
20. RUN the APP
Click on **Google** link



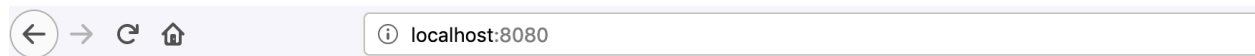
21. You will be redirected to Google login page



22. Login with your Google credentials



23. Observe that you get the following response



Spring Security OAuth and OIDC Google Sample

Welcome 118164094545438500238 into Spring Boot LDAP managed user using BASIC authentication.

