

OAuth 2.0

Single-Sign On with GitHub

Dr Harshad Prajapati
31 Dec 2022

Register our app on Authorization Server

<https://github.com/settings/applications/new>

Register a new OAuth application

Application name *

Something users will recognize and trust.

Homepage URL *

The full URL to your application homepage.

Application description

This is displayed to all users of your application.

Authorization callback URL *

Your application's callback URL. Read our [OAuth documentation](#) for more information.

☐ Enable Device Flow

Allow this OAuth App to authorize users via the Device Flow.

Read the [Device Flow documentation](#) for more information.

Register application

Cancel

sso-github



harshadbprajapati owns this application.

Transfer ownership

You can list your application in the [GitHub Marketplace](#) so that other users can discover it.

List this application in the Marketplace

0 users

Revoke all user tokens

Client ID

Client secrets

Generate a new client secret

You need a client secret to authenticate as the application to the API.


Store Client ID and Client Secret at safe location

Client ID

Client secrets

Generate a new client secret

Make sure to copy your new client secret now. You won't be able to see it again.



Client secret

✓

Added 1 minute ago by harshadbprajapati

Never used

You cannot delete the only client secret. Generate a new client secret first.

Delete

Store Client ID and Client Secret

- Client ID:



- Client Secret:



Homepage URL *

http://localhost:8080

The full URL to your application homepage.

Application description

This app is for demonstration of
SSO with GitHub server

This is displayed to all users of your application.

Authorization callback URL *

http://localhost:8080

Your application's callback URL. Read our [OAuth documentation](#) for more information.

☐ Enable Device Flow

Allow this OAuth App to authorize users via the Device Flow.

Read the [Device Flow documentation](#) for more information.

Update application

OAuth application settings x +

← → ↻ 🏠 🔒 https://github.com/settings/applications/2075... 🔗 ☆ 🧩 ⚙️ □

📧 20+ Gmail 📺 YouTube 📍 Maps 📧 0 📧 1 ⚙️ React Book - Forms 📖 How to integrate R...

☰ GitHub

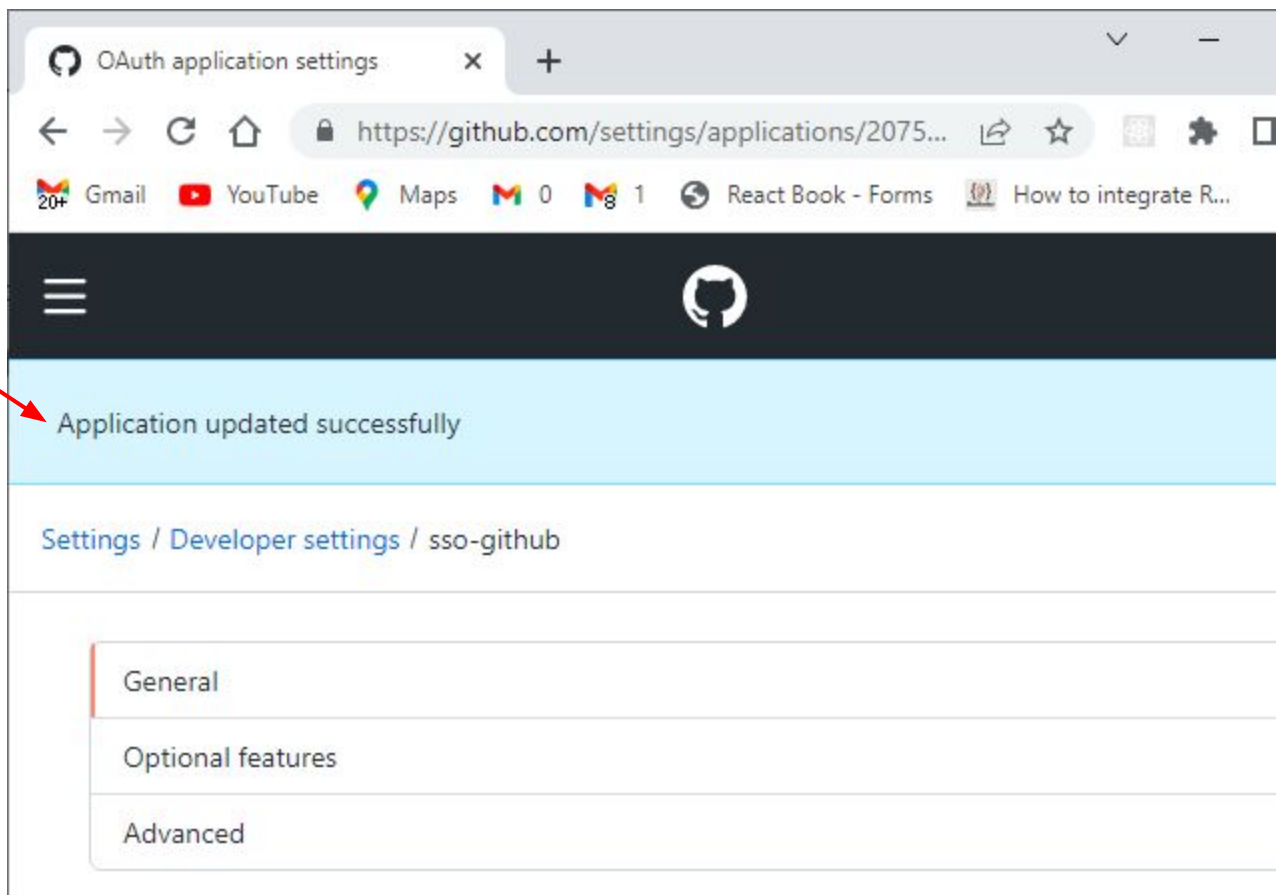
Application updated successfully

[Settings](#) / [Developer settings](#) / sso-github

General

Optional features

Advanced



Create a Spring Boot project on
Spring Initializr (<https://start.spring.io/>)

Project

☐ Gradle - Groovy

☐ Gradle - Kotlin

☒ Maven

Language

☒ Java

☐ Kotlin

☐ Groovy

Spring Boot

☐ 3.0.2 (SNAPSHOT)

☒ 3.0.1

☐ 2.7.8 (SNAPSHOT)

☐ 2.7.7

Project Metadata

Group

Artifact

Name

Description

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Security SECURITY

Highly customizable authentication and access-control framework for Spring applications.

OAuth2 Client SECURITY

Spring Boot integration for Spring Security's OAuth2/OpenID Connect client features.

GENERATE CTRL + G

EXPLORE CTRL + SPACE

SHARE...

pom.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.0.1</version>
    <relativePath/><!-- lookup parent from repository -->
  </parent>
  <groupId>com.example</groupId>
  <artifactId>sso-github</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>sso-github</name>
  <description>Project for SSO with GitHub</description>
  <properties>
    <java.version>19</java.version>
  </properties>
```

pom.xml file

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-oauth2-client</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
</dependencies>
```

pom.xml file

```
    <dependency>
      <groupId>org.springframework.security</groupId>
      <artifactId>spring-security-test</artifactId>
      <scope>test</scope>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>

</project>
```

controller/MainController.java

```
package com.example.ssogithub.controller;
```

```
import org.springframework.security.oauth2.client.authentication.OAuth2AuthenticationToken;
```

```
import org.springframework.stereotype.Controller;
```

```
import org.springframework.web.bind.annotation.GetMapping;
```

```
import java.util.logging.Logger;
```

```
@Controller
```

```
public class MainController {
```

```
    private Logger logger = Logger.getLogger(MainController.class.getName());
```

```
    @GetMapping("/")
```

```
    public String main(OAuth2AuthenticationToken token){
```

```
        logger.info("OAuth2Authentication Principal Value = " + token.getPrincipal());
```

```
        return "main.html";
```

```
    }
```

```
}
```

config/ProjectConfig.java

```
package com.example.ssogithub.config
```

```
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.oauth2.client.CommonOAuth2Provider;
import org.springframework.security.oauth2.client.registration.ClientRegistration;
import org.springframework.security.oauth2.client.registration.ClientRegistrationRepository;
import org.springframework.security.oauth2.client.registration.InMemoryClientRegistrationRepository;
import org.springframework.security.web.SecurityFilterChain;
```

```
@Configuration
```

```
@EnableWebSecurity
```

```
public class ProjectConfig {
```

```
    @Value("${sso.github.client.id}")
```

```
    private String clientId;
```

```
    @Value("${sso.github.client.secret}")
```

```
    private String clientSecret;
```

config/ProjectConfig.java

```
public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  
    //http.oauth2Login();  
    http.oauth2Login(c -> c.clientRegistrationRepository(clientRegistrationRepository()));  
  
    http.authorizeHttpRequests(  
        authz -> authz.anyRequest().authenticated()  
    );  
    return http.build();  
}  
  
private ClientRegistration clientRegistration() {  
    return CommonOAuth2Provider.GITHUB.getBuilder("github")  
        .clientId(clientId)  
        .clientSecret(clientSecret)  
        .build();  
}  
  
@Bean  
public ClientRegistrationRepository clientRegistrationRepository() {  
    var c = clientRegistration();  
    return new InMemoryClientRegistrationRepository(c);  
}
```


static/main.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>SSO with GitHub</title>
</head>
<body>
  <h1>Welcome to Single-Sign On with GitHub</h1>
</body>
</html>
```

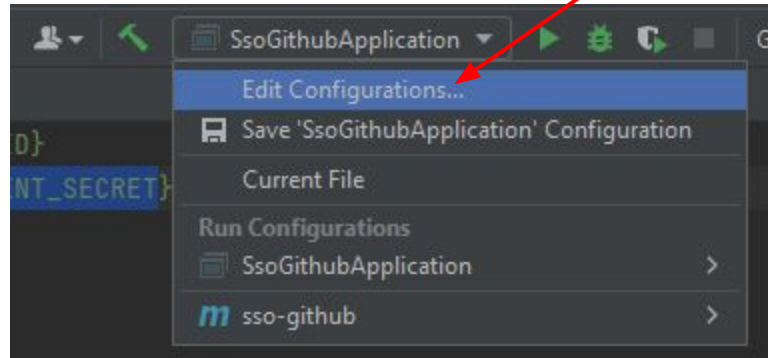
application.properties file

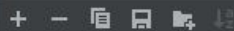
```
sso.github.client.id=${SSO_GITHUB_CLIENT_ID}
```

```
sso.github.client.secret=${SSO_GITHUB_CLIENT_SECRET}
```

Create Environment Variables

- We create two environment variables and assign them values of Client ID and Client Secret that we kept at safe place
 - SSO_GITHUB_CLIENT_ID
 - SSO_GITHUB_CLIENT_SECRET
- We can directly create environment variables in IntelliJ itself instead of creating them at OS level.





Application

SsoGithubApplication

Maven

Name: SsoGithubApplication

☐ Store as project file

Build and run

[Modify options](#) Alt+M

java 19 SDK of 'sso-github' modu

com.example.ssogithub.SsoGithubApplication

Program arguments

Press Alt for field hints

Working directory: D:\Projects\Spring Security in Action\sso-github

Environment variables: a1064706174f3;SSO_GITHUB_CLIENT_SECRET=

Separate variables with semicolon: VAR=value; VAR1=value1

Open run/debug tool window when started

Code Coverage

[Modify](#)

Packages and classes to include in coverage data

☒ com.example.ssogithub.*[Edit configuration templates...](#)

OK

Cancel

Apply

Testing our application

- Visit <http://localhost:8080>
- We get redirected to GitHub login page



Sign in to **GitHub**
to continue to **sso-github**

Username or email address

Password

[Forgot password?](#)

Sign in

Device verification



Email

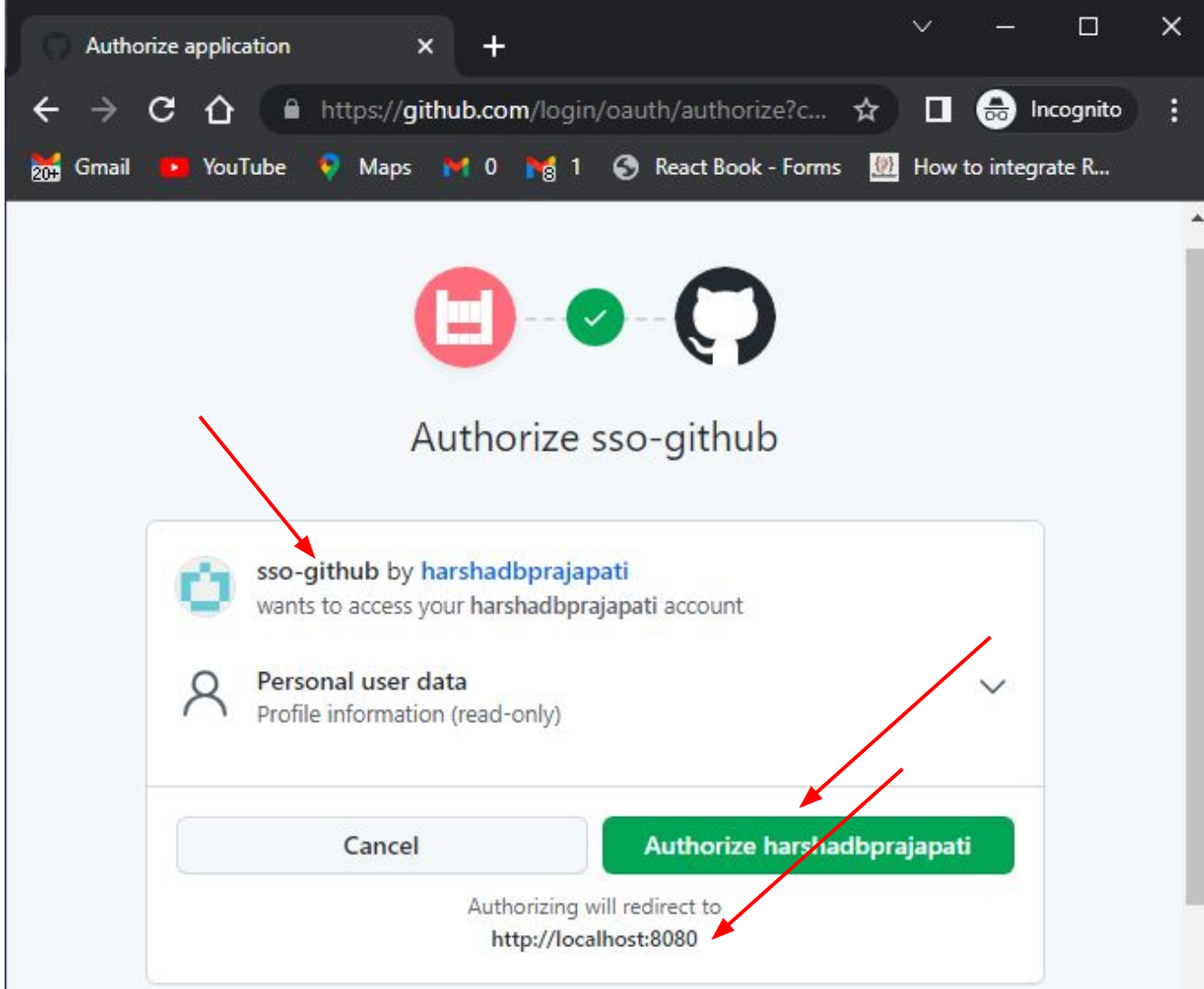
We just sent your authentication code via email to

h*****@gmail.com. The code will expire at 8:17AM IST.

Device Verification Code

Verify

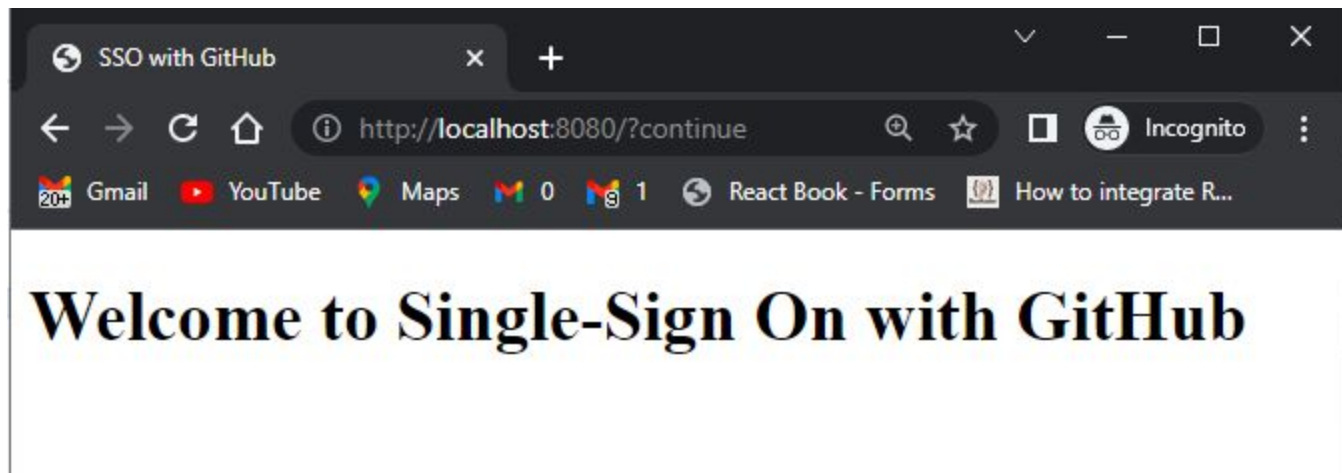
Having trouble verifying via email?

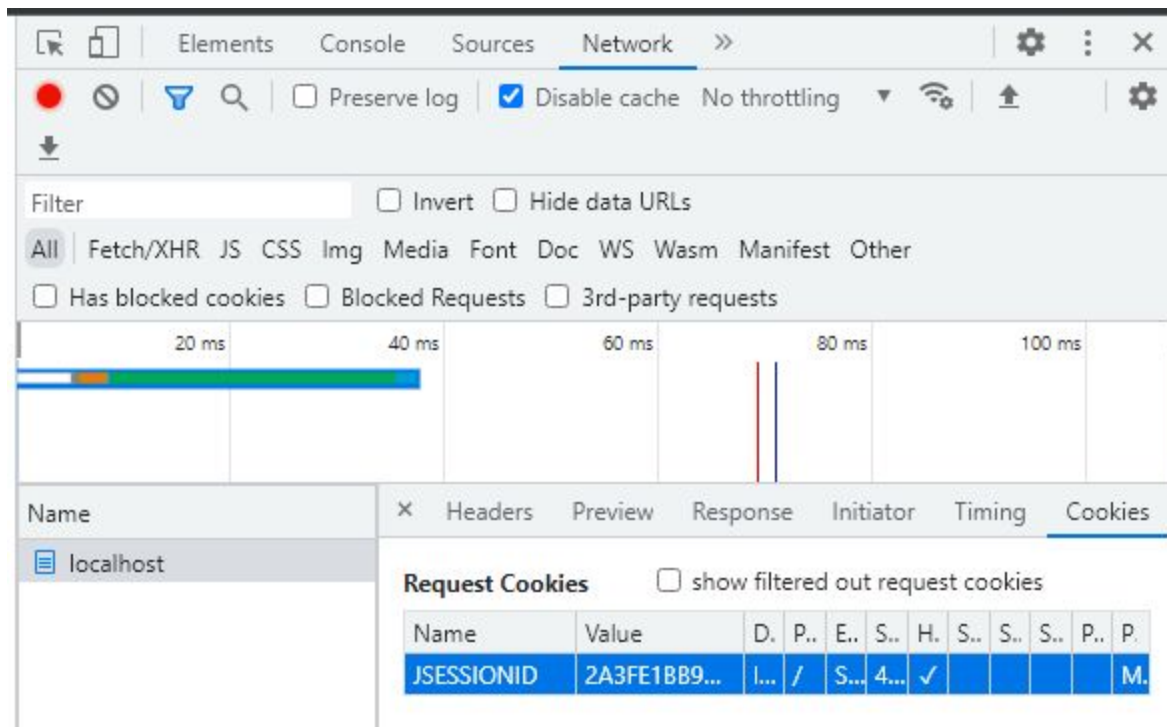




You are being redirected to the authorized application.

If your browser does not redirect you back, please [click here](#) to continue.





See server log

```
2022-12-31T07:41:14.854+05:30 INFO 8076 --- [main] c.e.ssogithub.SsogithubApplication : Started SsogithubApplication in 8.529 seconds (prod
2022-12-31T07:41:38.788+05:30 INFO 8076 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherSe
2022-12-31T07:41:38.789+05:30 INFO 8076 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2022-12-31T07:41:38.794+05:30 INFO 8076 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 3 ms
2022-12-31T07:41:39.164+05:30 WARN 8076 --- [nio-8080-exec-2] o.a.c.util.SessionIdGeneratorBase : Creation of SecureRandom instance for session ID ge
2022-12-31T07:41:39.168+05:30 WARN 8076 --- [nio-8080-exec-1] o.a.c.util.SessionIdGeneratorBase : Creation of SecureRandom instance for session ID ge
2022-12-31T07:41:55.048+05:30 INFO 8076 --- [nio-8080-exec-6] c.e.ssogithub.controller.MainController : OAuth2Authentication Principal Value = Name: [38149
```

