

Get Started with Spring Boot, SAML, and Okta

 javacodegeeks.com/2017/06/get-started-spring-boot-saml-okta.html

Build faster with Okta's authentication and user management API. [Register today for the free forever Developer Edition!](#)

Today I'd like to show you how build a Spring Boot application that leverages Okta's Platform API for authentication via SAML. SAML (Security Assertion Markup Language) is an XML-based standard for securely exchanging authentication and authorization information between entities—specifically between identity providers, service providers, and users. Well-known IdPs include Salesforce, Okta, OneLogin, and Shibboleth.

My Okta developer experience began a couple years ago (in December 2014) when I worked for a client that was adopting it. I was tasked with helping them decide on a web framework to use, so I built prototypes with Node, Ruby, and Spring. I documented my findings in [a blog post](#). Along the way, [I tweeted](#) my issues with Spring Boot, and [asked how to fix it](#) on Stack Overflow. I ended up figuring out the solution through trial-and-error and my findings made it into the [official Spring documentation](#). Things have changed a lot since then and now Spring Security 4.2 has support for auto-loading custom DSLs. And guess what, there's even a DSL for SAML configuration!

Ready to get started? You can follow along in with the written tutorial below, [check out the code on GitHub](#), or watch the screencast I made to walk you through the same process.

Sign Up for an Okta Developer Account

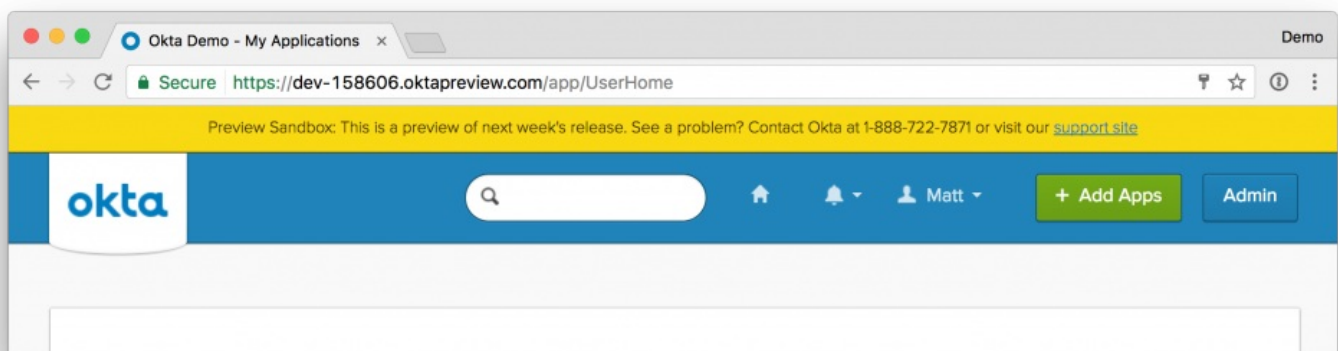
Fast forward two years, and I find myself as an Okta employee. To start developing with Okta, I created a new developer account at <https://developer.okta.com>. Make sure you take a screenshot or write down your Okta URL after you've signed up. You'll need this URL to get back to the admin console.

You'll receive an email to activate your account and change your temporary password. After completing these steps, you'll land on your dashboard with some annotations about "apps".

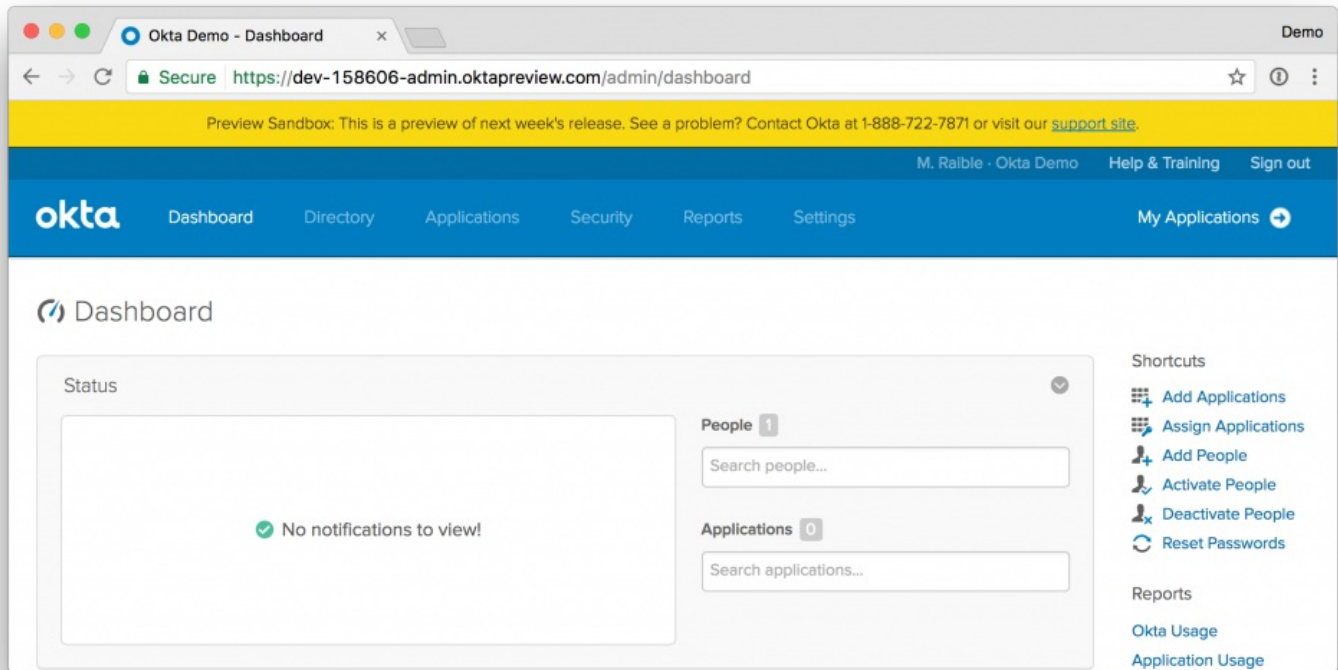
Create a SAML Application on Okta

At the time of this writing, the easiest way to create a SAML-aware Spring Boot application is to use Spring Security's [SAML DSL project](#). It contains a sample project that provides [instructions](#) for configuring Okta as a SAML provider. These instructions will likely work for you if you're experienced Spring Boot and Okta developer. If you're new to both, this "start from scratch" tutorial might work better for you.

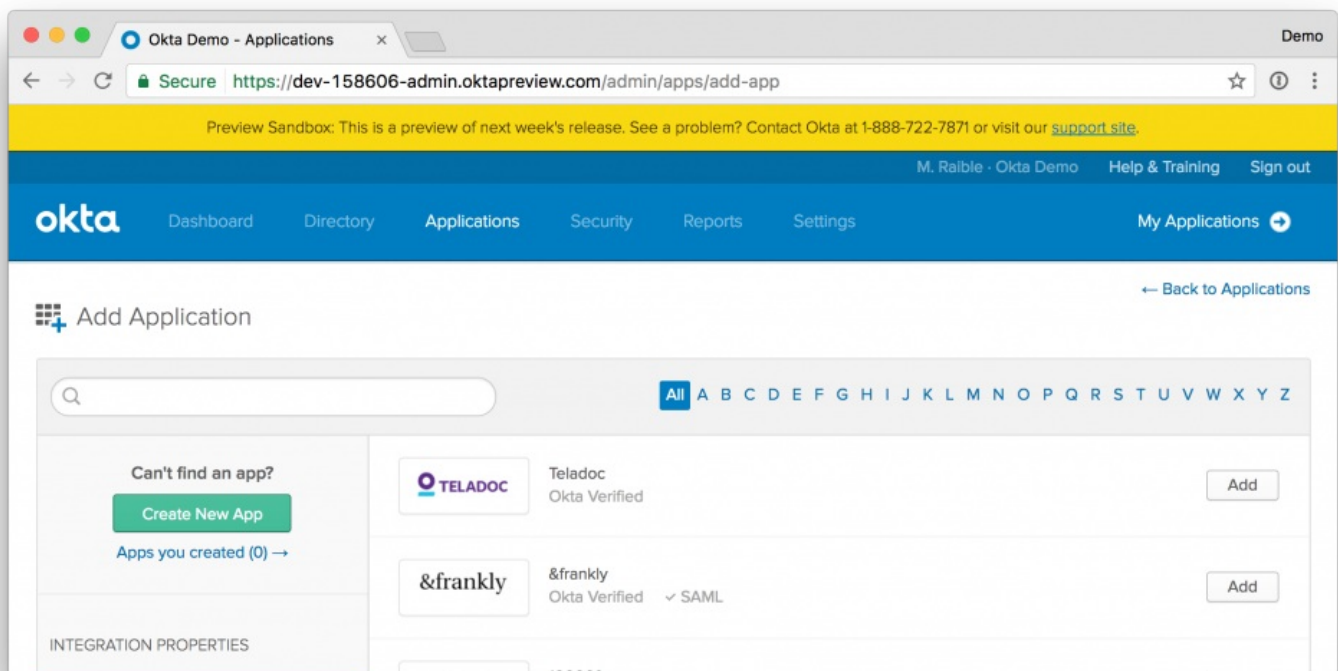
Just like I did, the first thing you'll need to do is create a developer account at <https://developer.okta.com>. After activating your account, login to it and click on the "Admin" button in the top right.



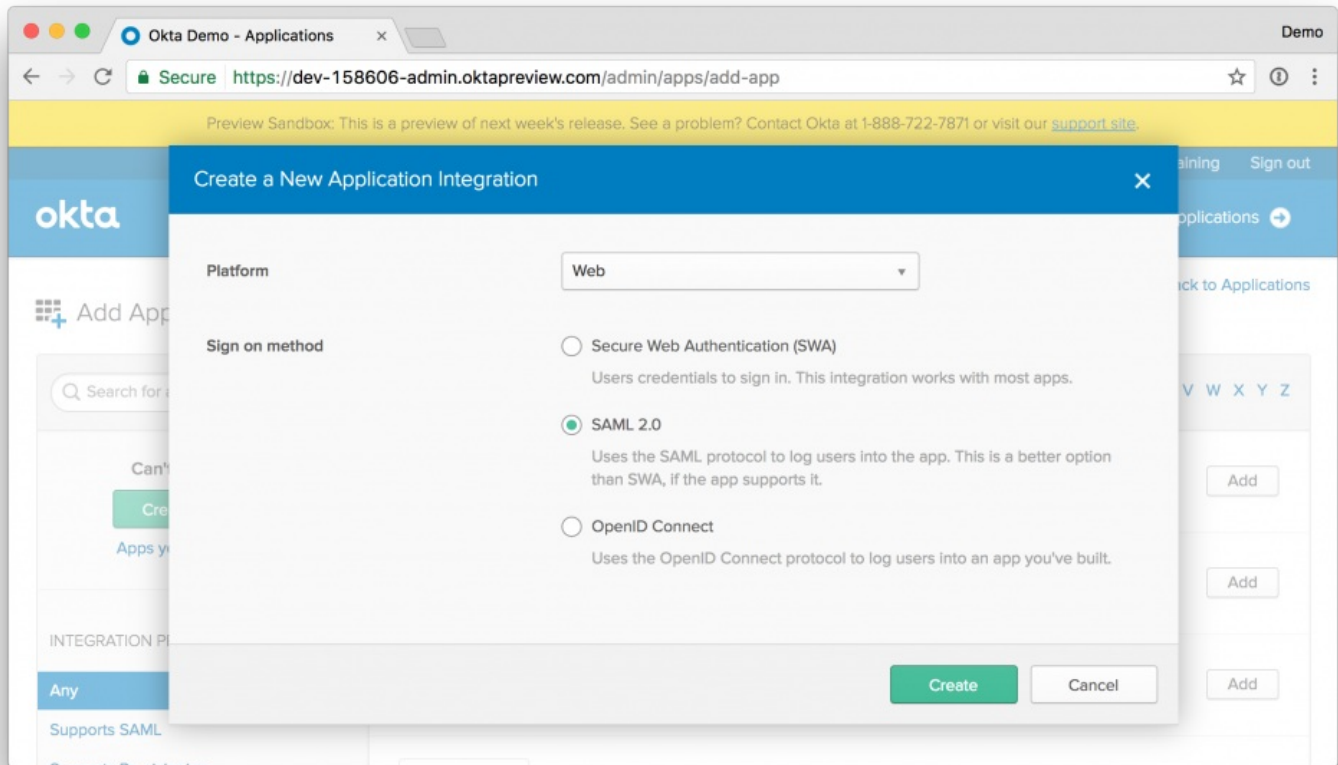
On the next screen, click "Add Applications" in the top right.



This will bring you to a screen with a “Create New App” green button on the left.



Click the button and choose “Web” for the platform and “SAML 2.0” for the sign on method.



Click the “Create” button. The next screen will prompt you for an application name. I used “Spring SAML”, but any name will work.

Okta Demo - Applications x Demo

Secure <https://dev-158606-admin.oktapreview.com/admin/apps/saml-wizard/create>

Preview Sandbox: This is a preview of next week's release. See a problem? Contact Okta at 1-888-722-7871 or visit our [support site](#).

M. Raible · Okta Demo Help & Training Sign out

okta Dashboard Directory Applications Security Reports Settings My Applications →

Create SAML Integration

1 General Settings

2 Configure SAML


3 Feedback

1 General Settings

App name

Spring SAML

App logo (optional) ?



Browse..

Upload Logo

App visibility

☐ Do not display application icon to users

☐ Do not display application icon in the Okta Mobile app

Click the “Next” button. This brings you to the second step, configuring SAML. Enter the following values:

- Single sign on URL: <https://localhost:8443/saml/SSO>
- Audience URI: <https://localhost:8443/saml/metadata>

Okta Demo - Applications x Demo

Secure <https://dev-158606-admin.oktapreview.com/admin/apps/saml-wizard/create>

Preview Sandbox: This is a preview of next week's release. See a problem? Contact Okta at 1-888-722-7871 or visit our [support site](#).

M. Raible · Okta Demo Help & Training Sign out


okta Dashboard Directory Applications Security Reports Settings

Create SAML Integration

1 General Settings 2 Configure SAML 3 Feedback


A SAML Settings


GENERAL

Single sign on URL 


☒ Use this for Recipient URL and Destination URL


☐ Allow apps to request other SSO URLs

Audience URI (SP Entity ID) 

Default RelayState 

If no value is set, a blank RelayState is sent

Name ID format 

Application username 

[Show Advanced Settings](#)

What does this form do?

This form generates the XML needed for the app's SAML request.

Where do I find the info this form needs?

The app you're trying to integrate with should have its own documentation on using SAML. You'll need to find that doc, and it should outline what information you need to specify in this form.

Okta Certificate

Import the Okta certificate to your Identity Provider if required.

[Download Okta Certificate](#)

Scroll to the bottom of the form and click "Next". This will bring you to the third step, feedback. Choose "I'm an Okta customer adding an internal app" and optionally select the App type.

Okta Demo - Applications x Demo

Secure <https://dev-158606-admin.oktapreview.com/admin/apps/saml-wizard/create>

Preview Sandbox: This is a preview of next week's release. See a problem? Contact Okta at 1-888-722-7871 or visit our [support site](#).

M. Raible · Okta Demo Help & Training Sign out

okta Dashboard Directory Applications Security Reports Settings

Create SAML Integration

1 General Settings

2 Configure SAML

3 Feedback

3 Help Okta Support understand how you configured this application

Are you a customer or partner?

☒ I'm an Okta customer adding an internal app

☐ I'm a software vendor. I'd like to integrate my app with Okta

i The optional questions below assist Okta Support in understanding your app integration.

App type **i**

☒ This is an internal app that we have created

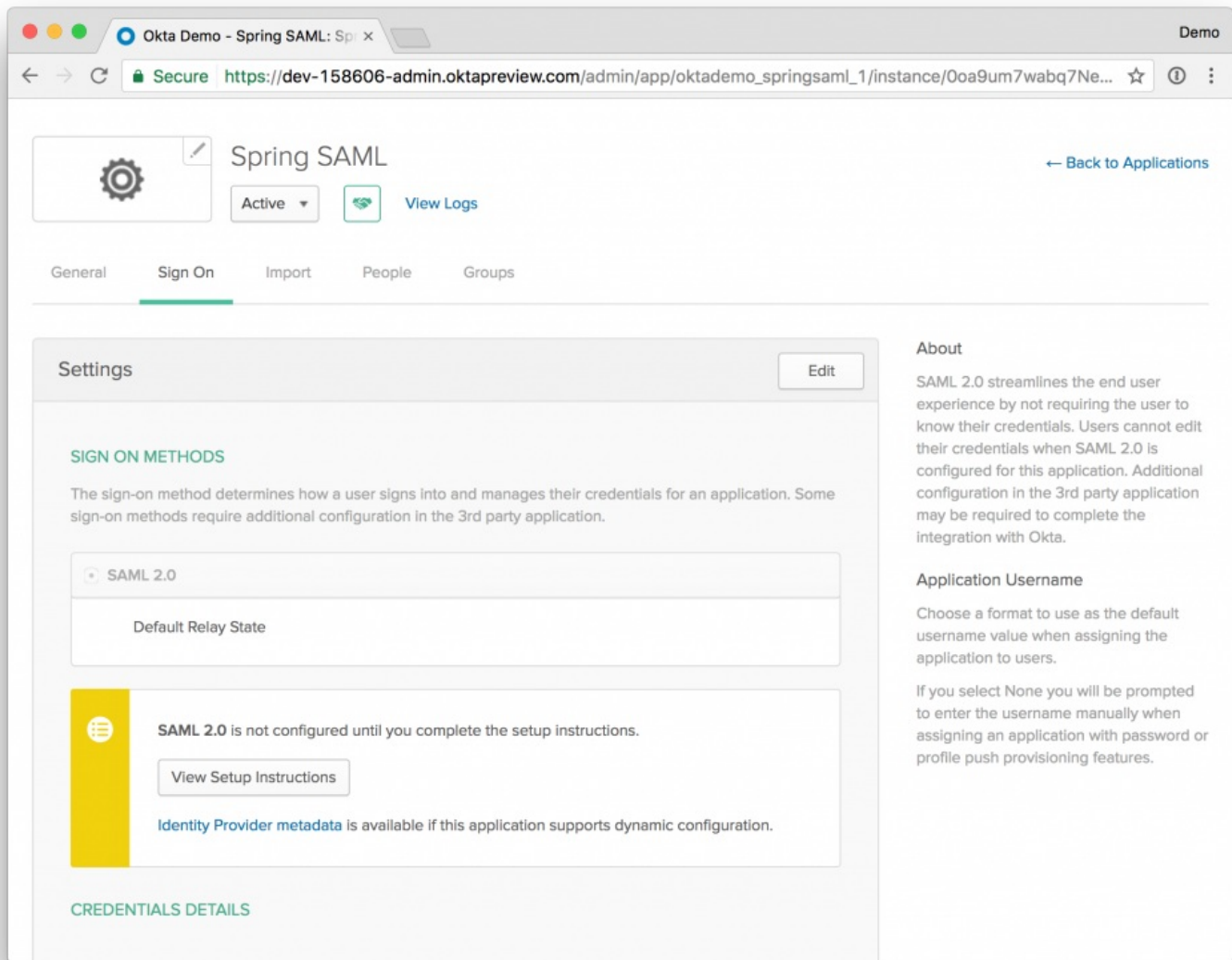
Why are you asking me this?

This form provides Okta Support with useful background information about your app. Thank you for your help—we appreciate it.

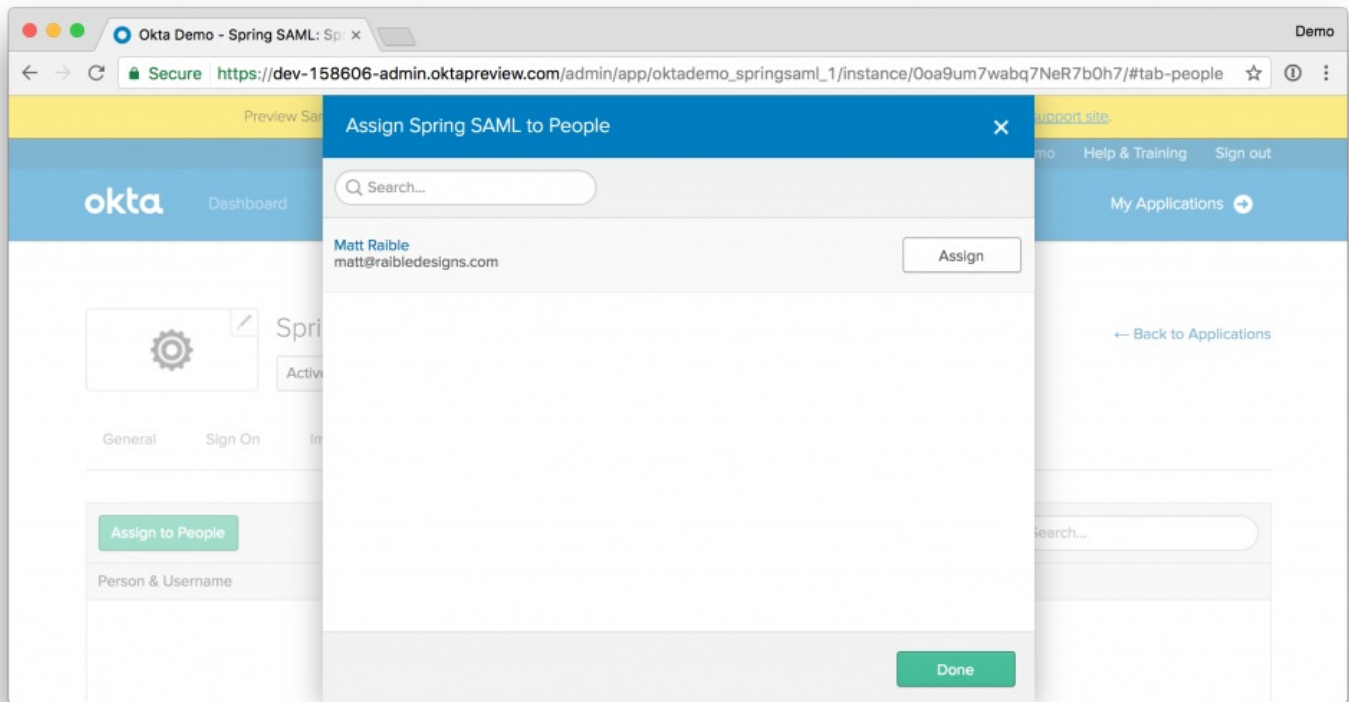
Previous

Finish

Click the “Finish” button to continue. This will bring you to the application’s “Sign On” tab which has a section with a link to your applications metadata in a yellow box. Copy the **Identity Provider metadata** link as you’ll need it to configure your Spring Boot application.



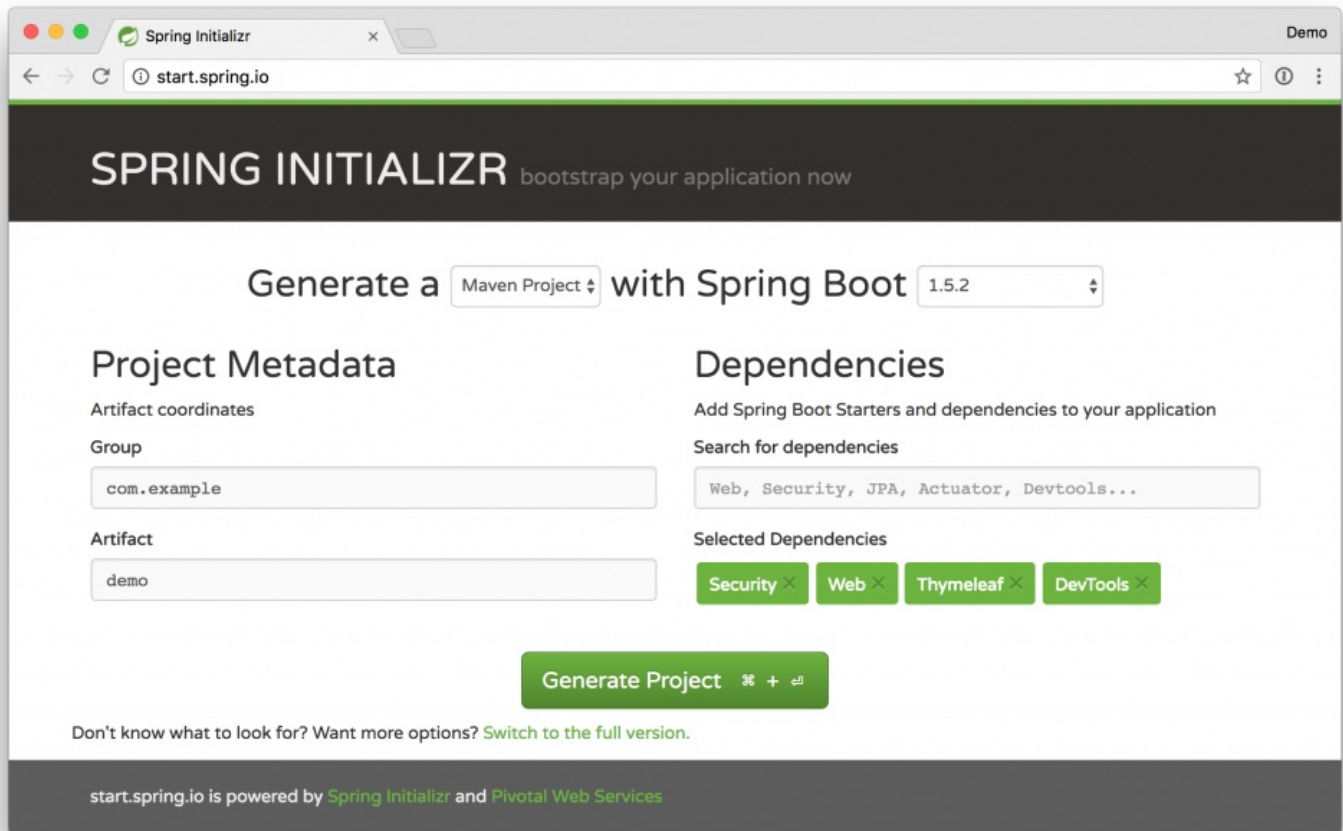
The final setup step you'll need is to assign people to the application. Click on the "People" tab and the "Assign to People" button. You'll see a list of people with your account in it.



Click the assign button, accept the default username (your email), and click the “Done” button.

Create a Spring Boot Application with SAML Support

Navigate to <https://start.spring.io> in your favorite browser and select Security, Web, Thymeleaf, and DevTools as dependencies.



Click “Generate Project”, download the generated ZIP file and open it in your favorite editor. Add the `spring-security-saml-dsl` dependency to your `pom.xml`.

```
<dependency>  
  
    <groupId>org.springframework.security.extensions</groupId>  
  
    <artifactId>spring-security-saml-dsl</artifactId>  
  
    <version>1.0.0.M3</version>  
  
</dependency>
```

You’ll also need to add the Spring Milestone repository since a milestone release is all that’s available at the time of this writing.

```
<repositories>  
  
    <repository>  
  
        <id>spring-milestones</id>  
  
        <name>Spring Milestones</name>
```

```
<url>https://repo.spring.io/libs-milestone</url>
```

```
</repository>
```

```
</repositories>
```

If you'd like to see instructions for Gradle, please view the project's [README.md](#).

In `src/main/resources/application.properties`, add the following key/value pairs. Make sure to use the "Identity Provider metadata" value you copied earlier (hint: you can find it again under the "Sign On" tab in your Okta application).

```
server.port =  
8443
```

```
server.ssl.enabled  
= true
```

```
server.ssl.key-alias spring
```

```
server.ssl.key-store =  
src/main/resources/saml/keystore.jks
```

```
server.ssl.key-store-password =  
secret
```

```
security.saml2.metadata-url = <your metadata  
url>
```

From a terminal window, navigate to the `src/main/resources` directory of your app and create a `saml` directory. Navigate into the directory and run the following command. Use "secret" when prompted for a keystore password.

```
keytool -genkey -keystore keystore.jks  
-v -alias  
spring -keyalg RSA -keysize 2048 -validity  
10000
```

The values for the rest of the questions don't matter since you're not generating a real certificate. However, you will need to answer "yes" to the following question.

```
Is CN=Unknown, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown  
correct?
```

```
[no]:
```

Create a `SecurityConfiguration.java` file in the `com.example` package.

```
package com.example;
```

```

import static org.springframework.security.extensions.saml2.config.SAMLConfigurer.saml;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.context.annotation.Configuration;

import
org.springframework.security.config.annotation.method.configuration.EnableGlobalMethodSecurity;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

@EnableWebSecurity

@Configuration

(securedEnabled
@EnableGlobalMethodSecurity= true)

WebSecurityConfigurerAdapter
public class SecurityConfiguration extends {

    @Value("${security.saml2.metadata-url}")

    String
    metadataUrl;

    @Value("${server.ssl.key-alias}")

    String
    keyAlias;

    @Value("${server.ssl.key-store-password}")

    String
    password;

```

```
@Value("${server.port}")
```

```
String  
port;
```

```
@Value("${server.ssl.key-store}")
```

```
String  
keyStoreFilePath;
```

```
@Override
```

```
                                HttpSecurity      Exception  
protected void configure(final http) throws {
```

```
    http
```

```
        .authorizeRequests()
```

```
            .antMatchers("/saml*").permitAll()
```

```
            .anyRequest().authenticated()
```

```
        .and()
```

```
        .apply(saml())
```

```
            .serviceProvider()
```

```
                .keyStore()
```

```
                    .storeFilePath("saml/keystore.jks")
```

```
                    .password(this.password)
```

```
                    .keyname(this.keyAlias)
```

```
                    .keyPassword(this.password)
```

```
                .and()
```

```

        .protocol("https")

        .hostname(String.format("%s:%s", "localhost", this.port))

        .basePath("/")

        .and()

        .identityProvider()

        .metadataFilePath(this.metadataUrl);
    }
}

```

Create an `IndexController.java` file in the same directory and use it to set the default view to `index`.

```

package com.example;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

@Controller

    IndexController
public class {

    @RequestMapping("/")

    String index()
    public {

        return "index";

    }
}

```

Since you chose Thymeleaf when creating your application, you can create a `src/main/resources/templates/index.html` and it will automatically be rendered after you sign-in. Create this file and populate it with the following HTML.

```
<!DOCTYPE html>

<html>

<head>

    <title>Spring Security SAML Example</title>

</head>

<body>

    Hello SAML!

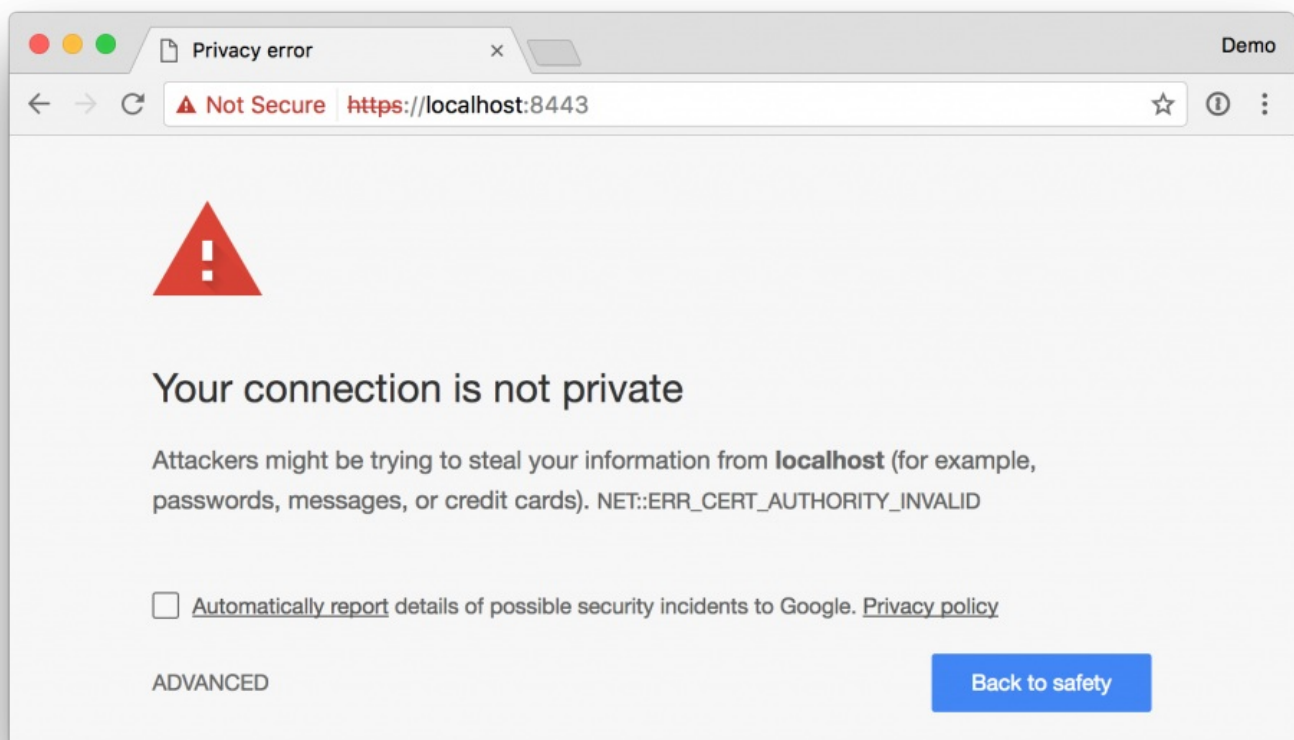
</body>

</html>
```

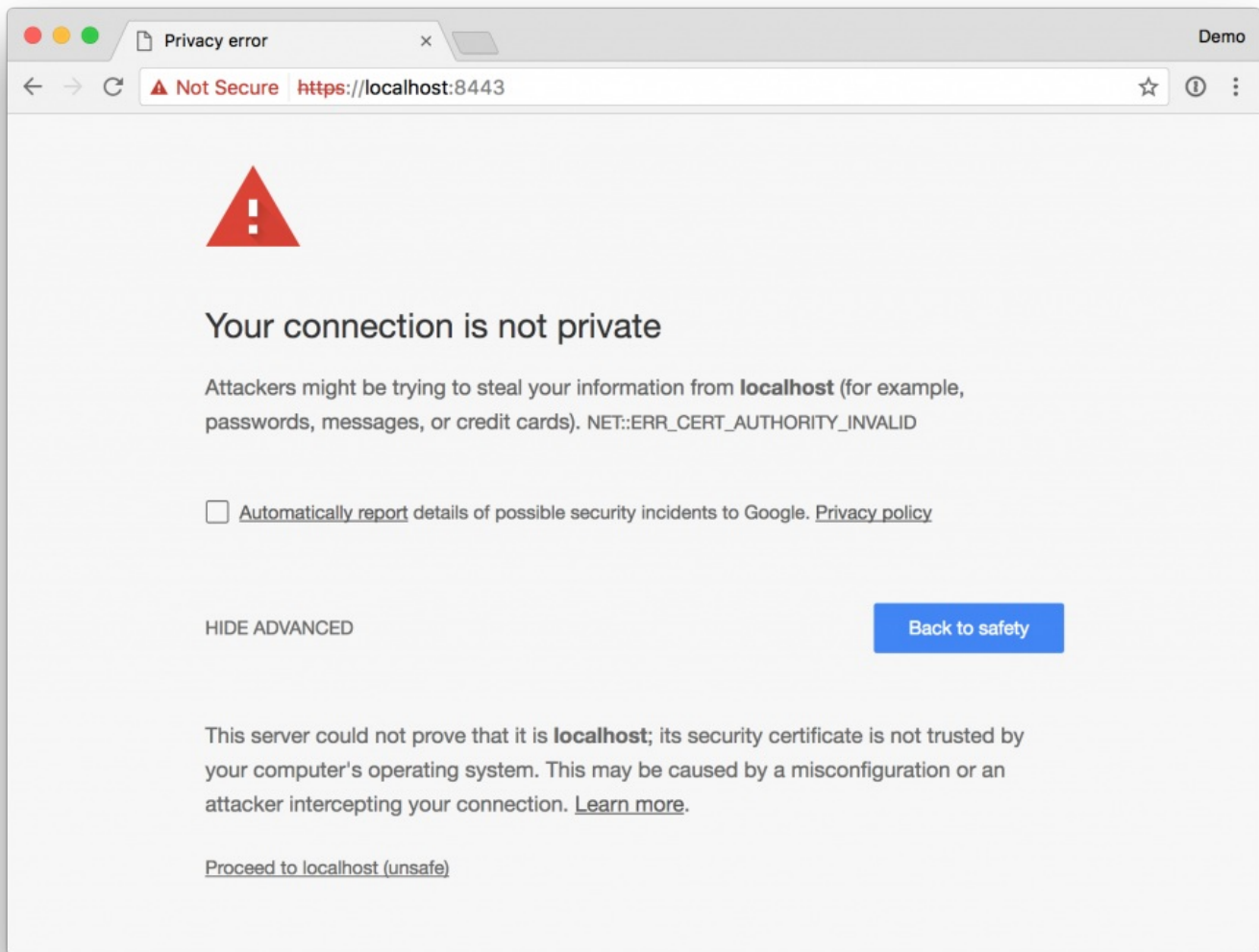
Run the App and Login with Okta

Start the app using your IDE or `mvn spring-boot:run` and navigate to <https://localhost:8443>. If you're using Chrome, you'll likely see a privacy error.

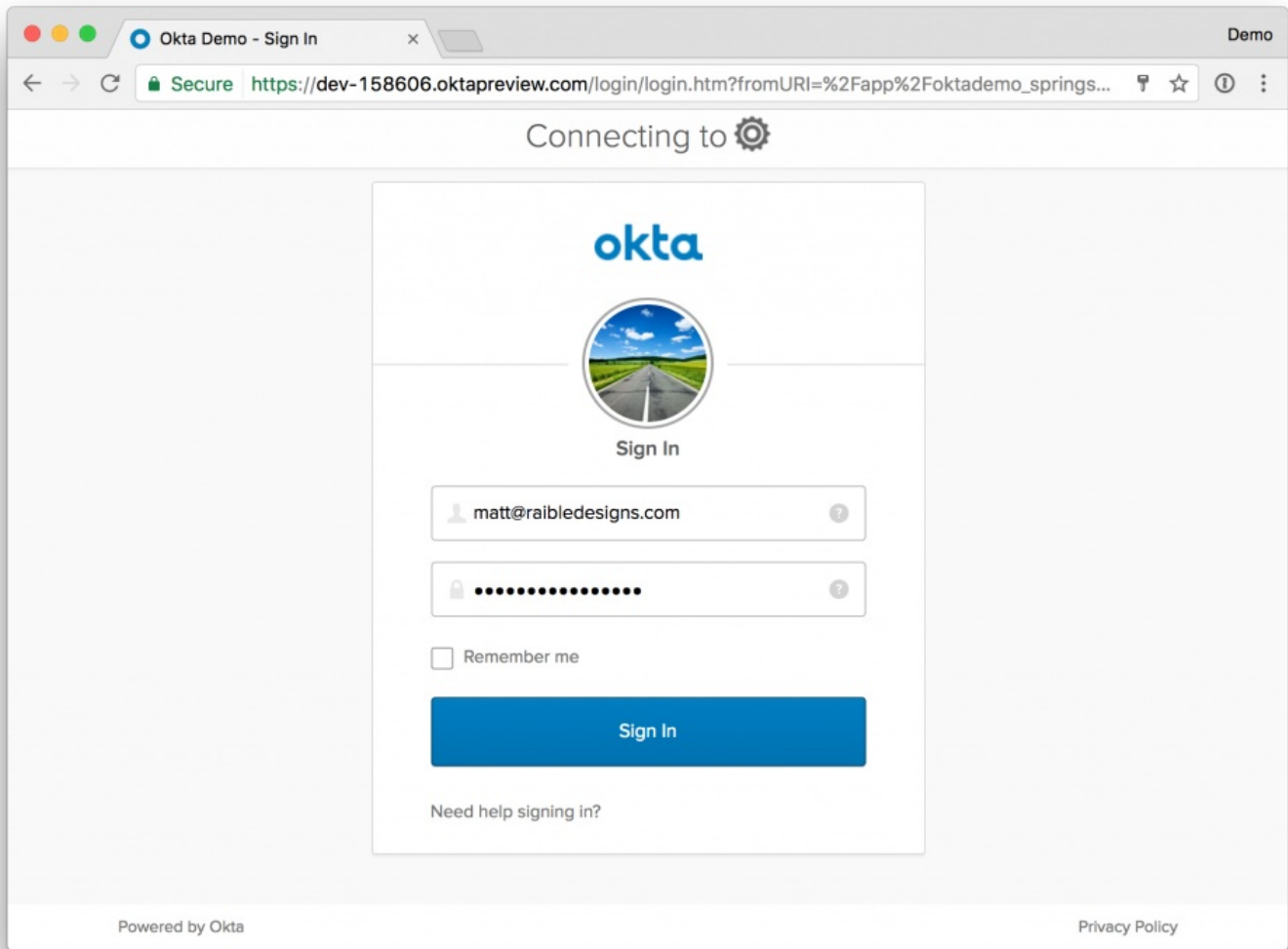
and navigate to <https://localhost:8443>. If you're using Chrome, you'll



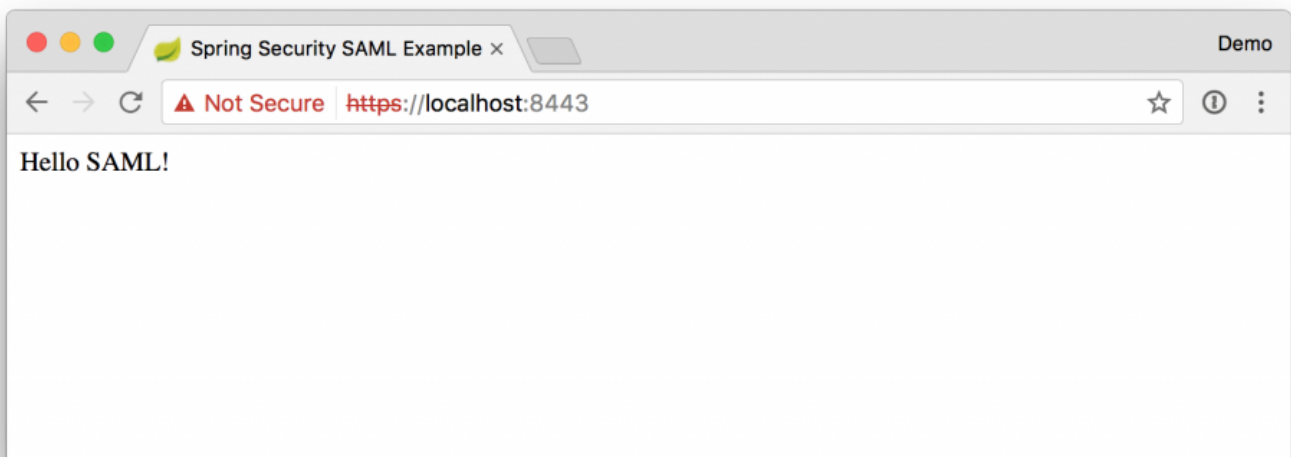
Click the “ADVANCED” link at the bottom. Then click the “proceed to localhost (unsafe)” link.



Next, you’ll be redirected to Okta to sign in and redirected back to your app. If you’re already logged in, you won’t see anything from Okta. If you sign out from Okta, you’ll see a login screen such as the one below.



After you've logged in, you should see a screen like the one below.



You can find the source code for this article at <https://github.com/oktadeveloper/okta-spring-boot-saml-example>.

Learn More

This article showed you how to create a SAML application in Okta and talk to it using Spring Boot and Spring Security's SAML extension. The SAML extension hasn't had a GA release, but hopefully will soon. I also believe it's possible to take the SAML DSL (in `SecurityConfiguration.java`) and create a Spring Boot starter that allows you to get started with SAML simply by configuring application properties.

Have questions or comments? Post your question to Stack Overflow with the “[okta](#)” or “[okta-api](#)” tag, hit me up via email at matt.raible@okta.com, or ping me on Twitter [@mraible](#). In future articles, I'll show you to to configure Spring Boot with OAuth 2.0 and Okta. Then I'll explore different techniques of authenticating with Angular and using the access token to talk to a secured Spring Boot application. Until then, happy authenticating!

Update: Thanks to [Alexey Soshin](#) for contributing a [pull request](#) to make the code in this blog post more bootiful!

Build faster with Okta's authentication and user management API. [Register today for the free forever Developer Edition!](#)