Softwareentwicklung (SW)
Login mit OAuth2

Prof. Dr. Alixandre Santana
alixandre.santana@oth-regensburg.de

Wintersemester
2024/2025

Lernziele

- Den Informationsfluss zwischen einem Client und einem OAuth2-Autorisierungsanbieter (wie Google) zu verstehen
- Ein Anwendungsprofil bei einem OAuth2-Anbieter zu erstellen
- Ein Login mit OAuth2 zu implementieren

Agenda

1. Der OAuth2-Architekturablauf
2. Setup mit Spring Boot und Google
3. Zugriff auf unsere Ressourcen mit OAuth2
4. Hands On

1- REST und Authentifizierung

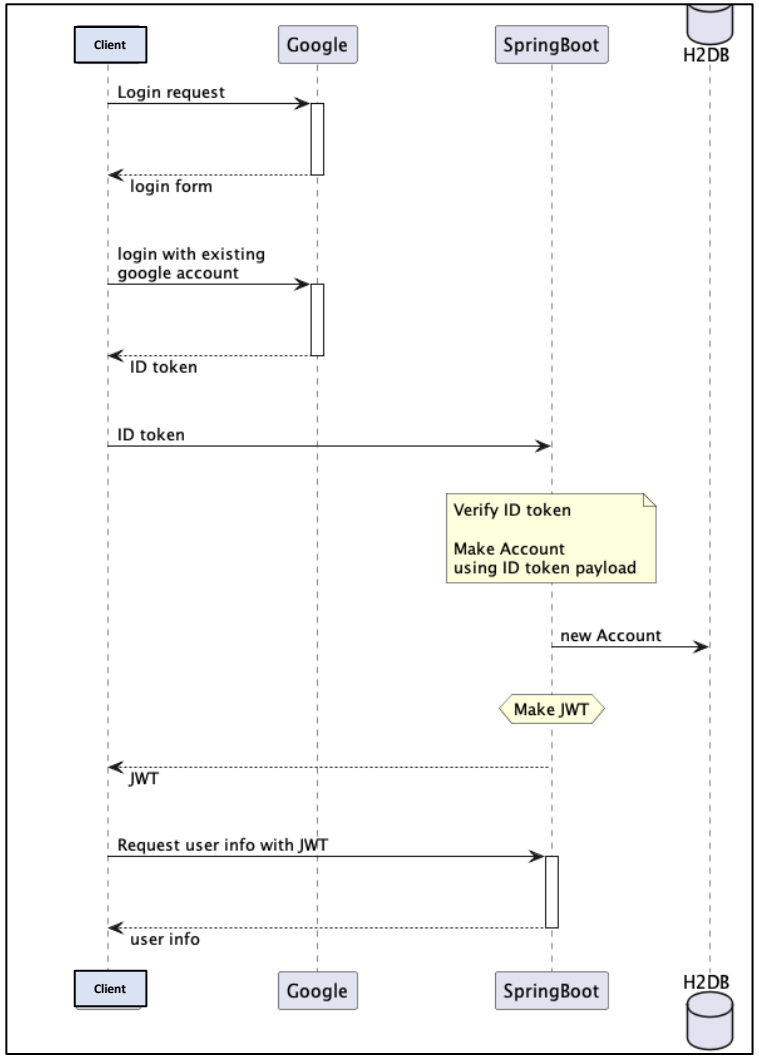
- Ist Ihre API öffentlich oder privat?
- Wie sensibel sind die Daten?

Lösungen:

- ❑ HTTPS Communication
- ❑ Authentication/Authorization (Spring Security):
- ❑ HttpBasic, HttpDigest, Token Based Authorization, OAuth

Agenda

1. Request Flow für eine OAuth2-basierte Anmeldung



https://github.com/baezzys/google-oauth2-login-spring-react-demo

Agenda

1. Der OAuth2-Architekturablauf
2. Setup mit Spring Boot und Google
3. Zugriff auf unsere Ressourcen mit OAuth2
4. Hands On

2- OAuth2 Client mit Spring Boot

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

https://www.freecodecamp.org/news/how-to-setup-jwt-authorization-and-authentication-in-spring/

2.2 - Google App Setup

Um das OAuth 2.0-Authentifizierungssystem von Google für die Anmeldung zu verwenden, müssen Sie in der Google API-Konsole ein Projekt einrichten, um OAuth 2.0-Anmeldeinformationen zu erhalten.

Mehr Info:

https://console.developers.google.com/

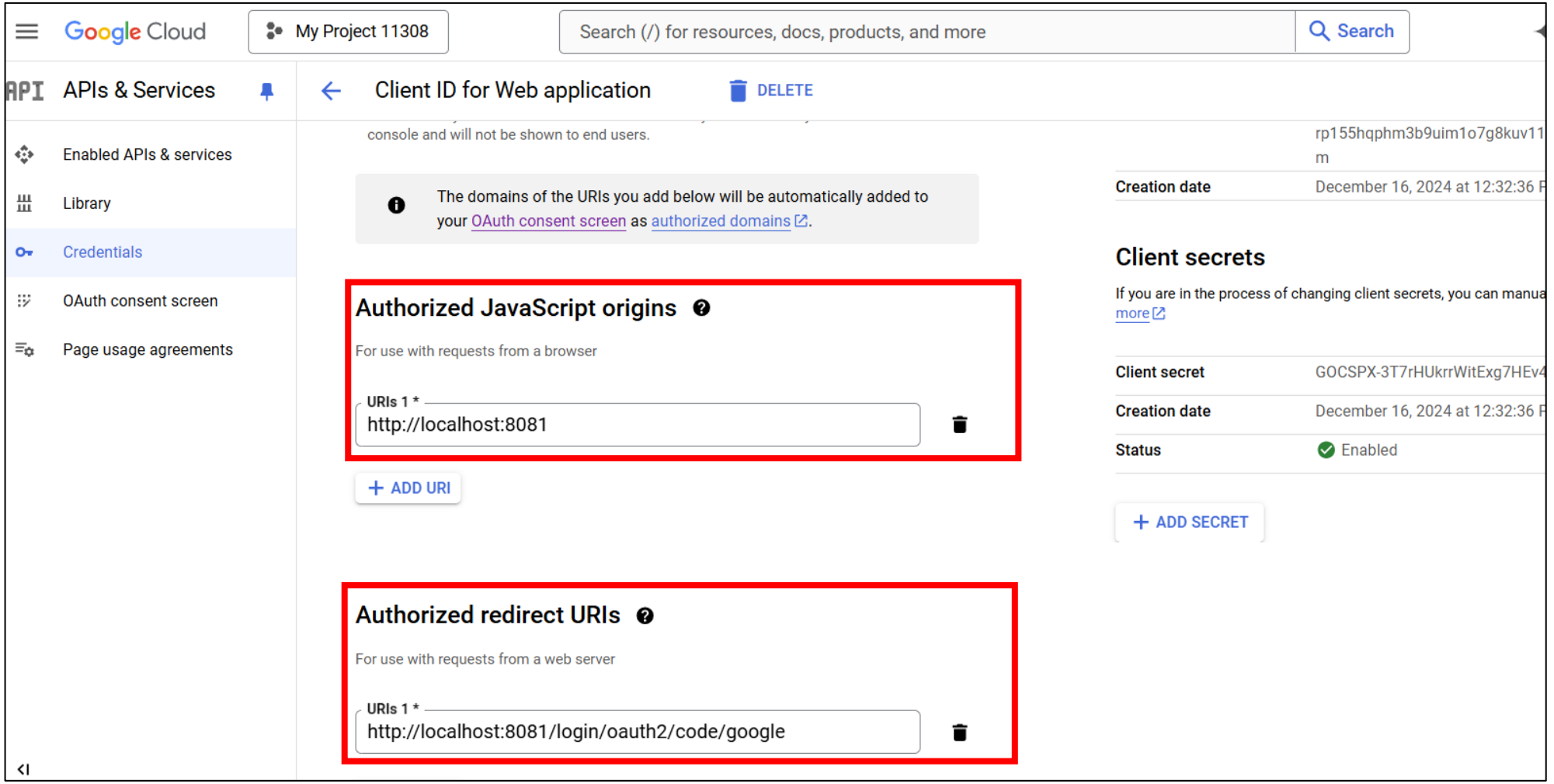
2.2- Google App Setup

Setting the redirect URI

Die Umleitungs-URI ist der Pfad in der Anwendung, zu dem der User-Agent des Endnutzers zurückgeleitet wird, nachdem er sich bei Google authentifiziert und dem OAuth-Client (der im vorherigen Schritt erstellt wurde) auf der Zustimmungsseite Zugriff gewährt hat.

Stellen Sie im Unterabschnitt „Umleitungs-URI festlegen“ im Google-Setup sicher, dass das Feld „Autorisierte Umleitungs-URIs“ auf http://localhost:8080/login/oauth2/code/google eingestellt ist.

2.2- Google App Setup



2.2- JSON Data for our APP


```
{"web":

{"client_id":"150382923487-rp155hqphm3b9uim1o7g8kuv117d2bb7.apps.googleusercontent.com",

"project_id":"pacific-apex-367315",

"auth_uri":"https://accounts.google.com/o/oauth2/auth",

"token_uri":"https://oauth2.googleapis.com/token",

"auth_provider_x509_cert_url":"https://www.googleapis.com/oauth2/v1/certs",

"client_secret":"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",

"redirect_uris":["http://localhost:8081/login/oauth2/code/google"],

"javascript_origins":["http://localhost:8081"]}}
```

2.2- Application.properties

Security Configuration

The Spring Security properties are prefixed with *spring.security.oauth2.client.registration* followed by the client name and then the name of the client property:

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

Adding these properties for at least one client will enable the *Oauth2ClientAutoConfiguration* class, which sets up all the necessary beans.

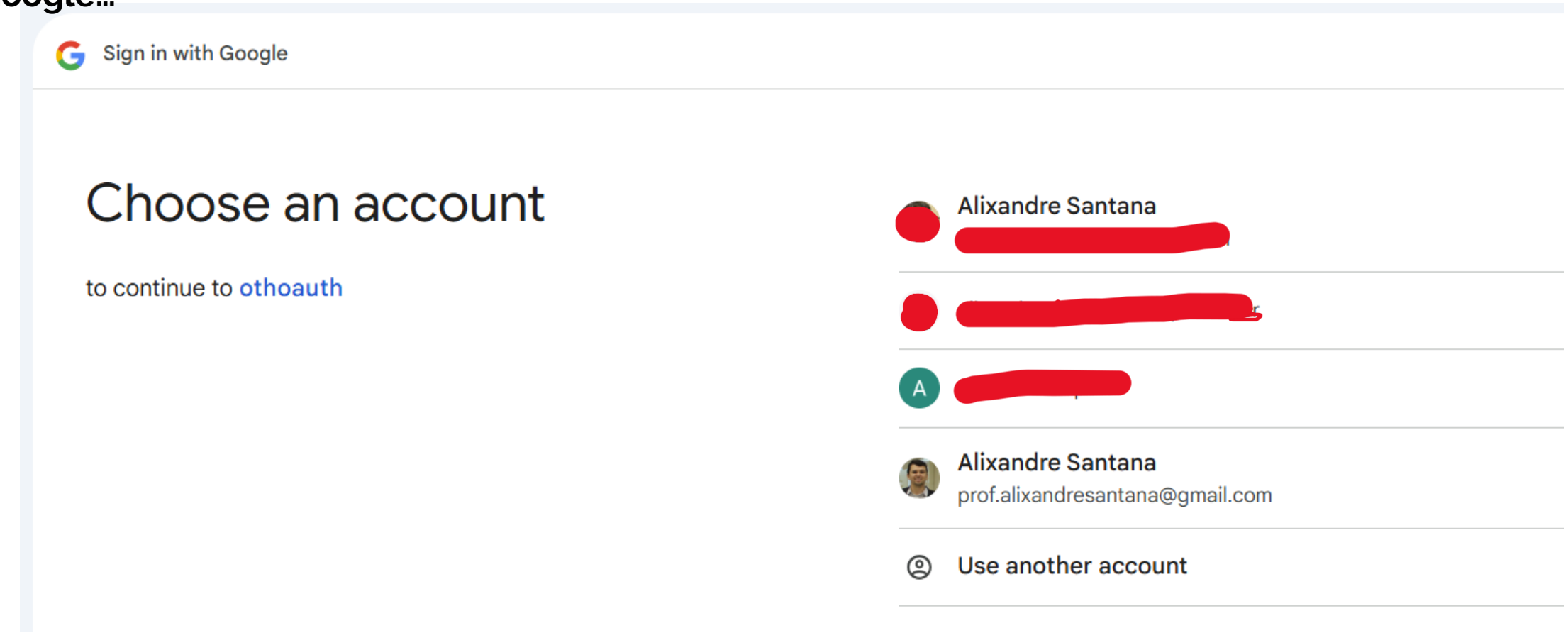
https://console.developers.google.com/

Agenda

1. Der OAuth2-Architekturablauf
2. Setup mit Spring Boot und Google
3. Zugriff auf unsere Ressourcen mit OAuth2
4. Hands On

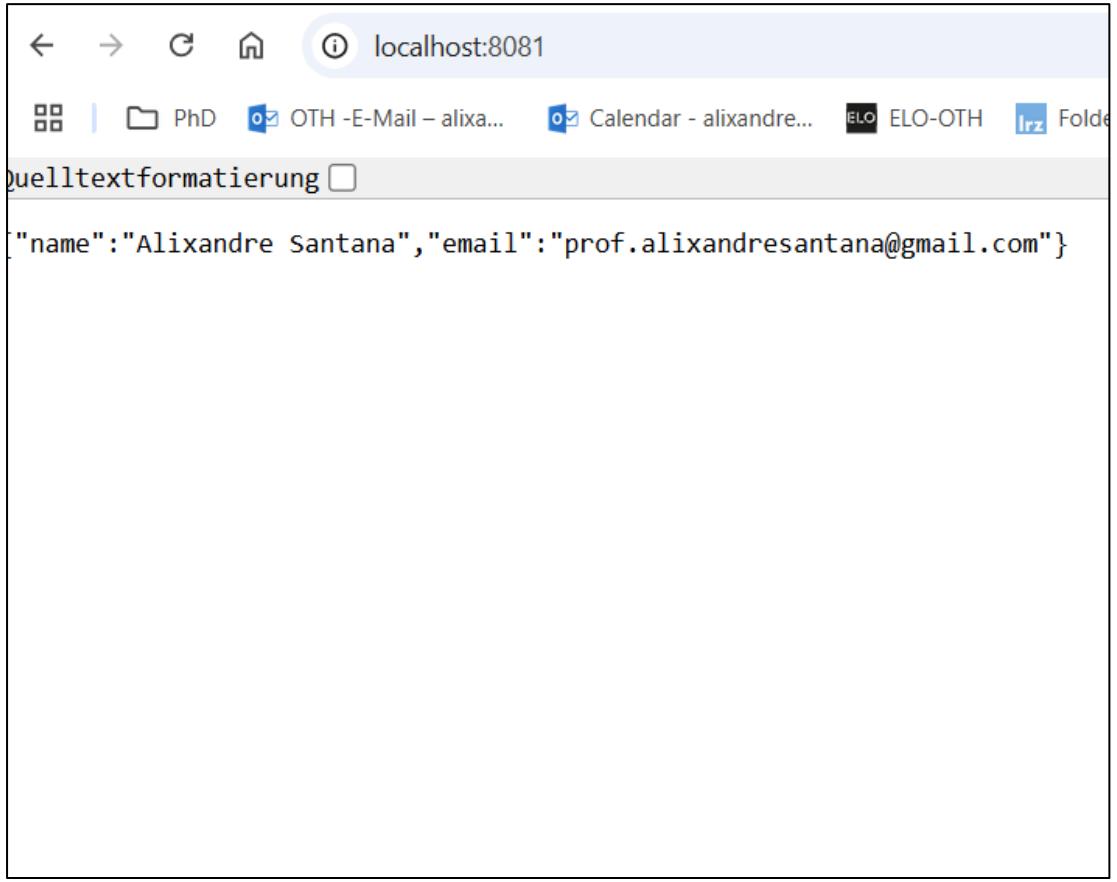
3- Accessing a protected Resource in our APP

Try to access any protected URL (ex. „/“) in your Application then you will get a login page from Google…



3- Getting the Google Credentials for the logged User

As a result, Google API gives us the user name and email back to our application....



4- Hands On

Check the project in ELO....

References

- https://spring.io/guides/tutorials/spring-boot-oauth2
- https://www.javainuse.com/webseries/spring-security-jwt/chap6#google_vignette