Spring 5.0

Enterprise and Mobile

DE HOGESCHOOL
MET HET NETWERK

Hogeschool PXL – Dep. PXL-IT – Elfde-Liniestraat 26 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook

Security of a Web API

REST in Spring

- @RestController
- @RequestMapping
  - @GetMapping
  - @PostMapping
- Path instellingen
  - @GetMapping(“/persons/{id}”)

PXL IT

REST Security

- Spring Security!
  - Authenticatie
  - Authorisatie

REST Security

- WebSecurityConfigurerAdapter beschikbaar stellen = bean die de interface implementeert

```
@Bean
public WebSecurityConfigurer<WebSecurity>
securityConfigurer(DataSource ds) {
    return new WebSecurityConfigurerAdapter() {
     (hier overschrijven we methodes @Override)
```

PXL IT

REST Security

- In hoofdapp of in aparte file (@ComponentScan vindt deze wel)
- Authentication Manager maken:

```
@Override
protected void
configure(AuthenticationManagerBuilder auth)
throws Exception {
    …
```

PXL IT

REST Security

- Voorbeelden:
  - auth.inMemoryAuthentication
  - auth. jdbcAuthentication

    -> zie demo
  - Optioneel: eigen provider en service maken

PXL IT

Voorbeeld eigen Authentication Provider

- AuthenticationProvider extenden (AbstractUserDetailsAuthenticationProvider)

```
@Component
@AllArgsConstructor(access = PACKAGE)
@FieldDefaults(level = PRIVATE, makeFinal = true)
final class TokenAuthenticationProvider extends AbstractUserDetailsAuthenticationProvider {
  @NonNull
  UserAuthenticationService auth;

  @Override
  protected void additionalAuthenticationChecks(final UserDetails d, final UsernamePasswordAuthenticationToken auth) {
    // Nothing to do
  }

  @Override
  protected UserDetails retrieveUser(final String username, final UsernamePasswordAuthenticationToken authentication) {
    final Object token = authentication.getCredentials();
    return Optional
      .ofNullable(token)
      .map(String::valueOf)
      .flatMap(auth::findByToken)
      .orElseThrow(() -> new UsernameNotFoundException("Cannot find user with authentication token=" + token));
  }
}
```

Voorbeeld eigen UserAuthenticationService

```java
public interface UserAuthenticationService {

  /**
   * Logs in with the given {@code username} and {@code password}.
   *
   * @param username
   * @param password
   * @return an {@link Optional} of a user when login succeeds
   */
  Optional<String> login(String username, String password);

  /**
   * Finds a user by its dao-key.
   *
   * @param token user dao key
   * @return
   */
  Optional<User> findByToken(String token);

  /**
   * Logs out the given input {@code user}.
   *
   * @param user the user to logout
   */
  void logout(User user);
}
```

PXL IT

Voorbeeld eigen UserAuthenticationService

```java
@Service
@AllArgsConstructor(access = PACKAGE)
@FieldDefaults(level = PRIVATE, makeFinal = true)
final class UUIDAuthenticationService implements UserAuthenticationService {
  @NonNull
  UserCrudService users;

  @Override
  public Optional<String> login(final String username, final String password) {
    final String uuid = UUID.randomUUID().toString();
    final User user = User
      .builder()
      .id(uuid)
      .username(username)
      .password(password)
      .build();

    users.save(user);
    return Optional.of(uuid);
  }

  @Override
  public Optional<User> findByToken(final String token) { return users.find(token); }

  @Override
  public void logout(final User user) {

  }
}
```

PXL IT

Uitgewerkt voorbeeld

- https://octoperf.com/blog/2018/03/08/securing-rest-api-spring-security/#spring-mvc-controllers
  - Authentication met UUIDs en JWT
  - Alle code op GitHub

PXL IT

Authorisatie

- Twee manieren:
  - Overschrijven van een configure(HttpSecurity http) van WebSecurityConfigurer:

```
@Override
protected void configure(HttpSecurity http)
        throws Exception {
    http.csrf().disable();
    http.httpBasic();
    http.authorizeRequests()
        .antMatchers( ...antPatterns: "/orders/**")
        .hasRole("ADULT");
}
```

  - Met annotatie @Secured({“ROL1”, “ROL2”})

PXL IT

Demo

- REST beer project

Testing

```java
@ExtendWith(SpringExtension.class)
@SpringBootTest(classes = BeerApp.class, webEnvironment = SpringBootTest.WebEnvironment.RANDOM_PORT)
@DirtiesContext
public class BeerRestTest {
   @LocalServerPort
   private int port;

   TestRestTemplate restTemplate = new TestRestTemplate();

   HttpHeaders headers = new HttpHeaders();

   @Test
   public void testGetAllBeers() {
      HttpEntity<String> entity = new HttpEntity<String>( body: null, headers);

      ResponseEntity<String> response = restTemplate.exchange(
               createURLWithPort( uri: "/allbeers"),
               HttpMethod.GET, entity, String.class);

      String expected = "{id:Course1,name:Spring,description:10 Steps}";
      try {
         JSONAssert.assertEquals(expected, response.getBody(), strict: false);
      } catch (Exception e) {

      }
   }

   private String createURLWithPort(String uri) {
      return "http://localhost:" + port + uri;
   }
}
```

Demo

- http://www.springboottutorial.com/integration-testing-for-spring-boot-rest-services
- https://www.baeldung.com/spring-security-method-security
- https://www.baeldung.com/how-to-use-resttemplate-with-basic-authentication-in-spring

PXL IT

Opdracht

- Zoek uit hoe Spring Security toegepast wordt
  - AOP? Wanneer gebeurt de weaving?
  - Examen!
- Voeg nu basic web security in je Spring backend van het project

PXL IT

Messaging

- Opdracht tegen volgende week:
- Lees hoofdstuk over messaging en denk na hoe je dit gaat toepassen in het project
  - ActiveMQ message broker (http://activemq.apache.org)
  - Zoek op hoe de eventuele koppeling naar .NET kan gemaakt worden

PXL IT