Ziel: Spring Security in Betrieb nehmen

CRM Rollen:

- ch.zli.m223.CRM.role.CrmRoles { USER, ADMIN, ALL_ROLES }
- ch.zli.m223.CRM.security.SpringRoles { ROLE_PREFIX }
- ch.zli.m223.CRM.security.CrmSpringRoles { ROLE_USER, ROLE_ADMIN, ROLE_ALL }

Authentication

Verbindung User mit Spring Security (Authentication von Login-Informationen mit unserer DB)

- ch.zli.m223.CRM.security.service.impl.UserDetailsServiceImpl

Ergänzen von Maven Dependencies

«pom.xml» ⇨ Tab «pom.xml»

```
<dependency>
      <groupId>org.thymeleaf.extras</groupId>
      <artifactId>thymeleaf-extras-springsecurity4</artifactId>
</dependency>

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

Und Anpassen der Dateien:

- /CRM /src/main/resources/templates/fragments/footer.html
- /CRM/src/main/resources/templates/fragments/header.html

Security Configuration

- ch.zli.m223.CRM.security.configuration.WebSecurityConfig
- ch.zli.m223.CRM.security.configuration.AccessDeniedHandlerImpl
- benütze BCryptPasswordEncoder in User

Sicherheitsdiskussion

Absicherung nur auf Controllerebene. Neuer Controller ohne Anpassen von «WebSecurityConfig» führt zu einem Sicherheitsloch.

Austeilen alle Klassen aus «ch.zli.m223.CRM.security.web» und testen dieser Aussage. (resources-security.zip).

Annotation «@EnableGlobalMethodSecurity (jsr250Enabled = true)» in «WebSecurityConfig» und Annotation aller Servicefunktionen mit «@PermitAll», «@RolesAllowed(CrmRoles.xxx)» löst das Problem.

Neues Problem:

Huhn / Ei Problem: Admin erstellen ohne Admin-Berechtigung

- ch.zli.m223.CRM.security.util.FakeUser
- «DataInitializer»-Update mit «makeUsSuperUser()»; «try {}» «finally(..) {}», sonst Sicherheitsloch