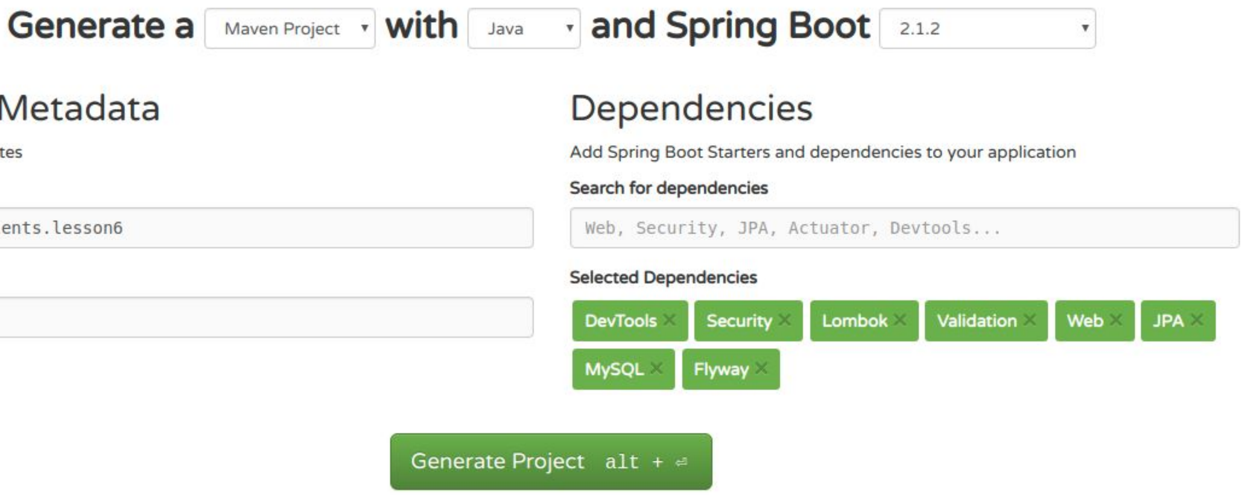
JAVA II

JPA & introduction to spring security & internationalization (i18n)

Before warm up :)

Go to: https://start.spring.io/

Generate project as an image below:



Generate a Maven Project with Java and Spring Boot 2.1.2

Project Metadata

Artifact coordinates

Group: lt.baltictalents.lesson6

Artifact: lesson6

Dependencies

Add Spring Boot Starters and dependencies to your application

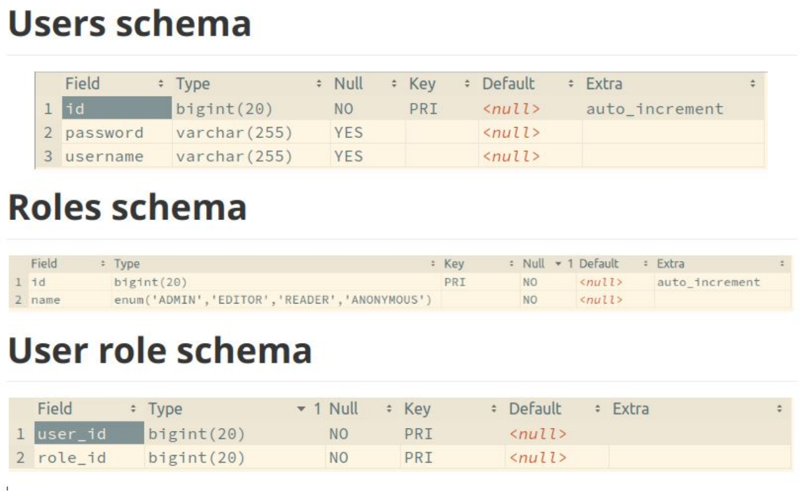
Search for dependencies: Web, Security, JPA, Actuator, Devtools...

Selected Dependencies: DevTools, Security, Lombok, Validation, Web, JPA, MySQL, Flyway

Import project to your IDE or text editor

Let’s create some tables with relations :)

Create databases with one to one (many to many) relation by schema provided below



Users schema

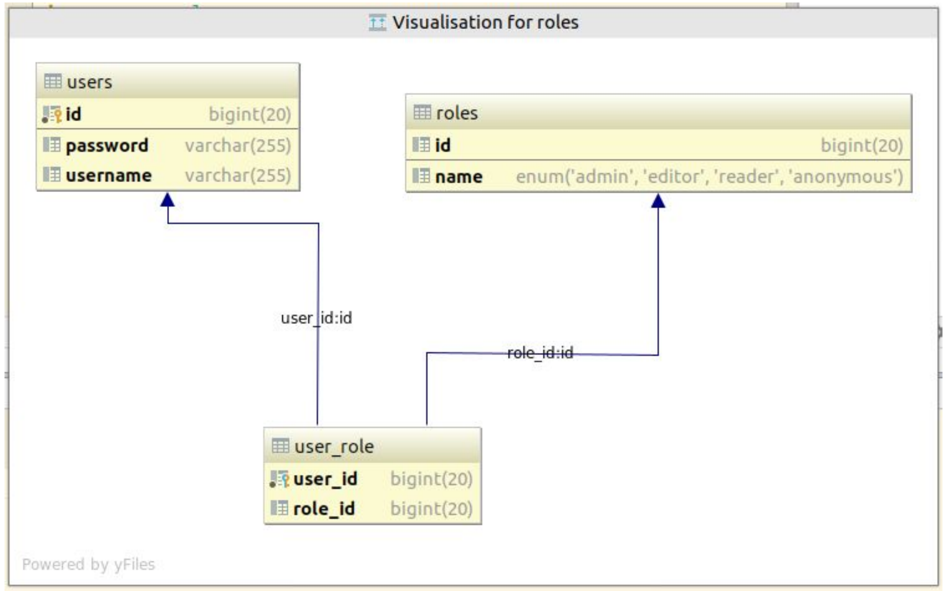
	Field	Type	Null	Key	Default	Extra
1	id	bigint(20)	NO	PRI	<null>	auto_increment
2	password	varchar(255)	YES		<null>	
3	username	varchar(255)	YES		<null>	

Roles schema

	Field	Type	Key	Null	Default	Extra
1	id	bigint(20)	PRI	NO	<null>	auto_increment
2	name	enum('ADMIN','EDITOR','READER','ANONYMOUS')		NO	<null>	

User role schema

	Field	Type	Null	Key	Default	Extra
1	user_id	bigint(20)	NO	PRI	<null>	
2	role_id	bigint(20)	NO	PRI	<null>	



What is data migration and seeding?

Data migration:
In software engineering, **schema migration** (also **database migration**, **database change management**]) refers to the management of incremental, reversible changes to relational database schemas. A schema migration is performed on a database whenever it is necessary to update or revert that database's schema to some newer or older version.

Data seeding:

Is the initial seeding of a database with data.

Seeding a database is a process in which an initial set of data is provided to a database when it is being installed .

It is especially useful when we want to populate the database with data we want to develop in future.

This is often an automated process that is executed upon the initial setup of an application.

Migration can be done with flyway util

Spring Boot comes with out-of-the-box integration for Flyway.

The migrations are scripts in the form `V<VERSION>__<NAME>.sql` (with `<VERSION>` an underscore-separated version, such as ‘1’ or ‘2_1’). By default, they are in a folder called `classpath:db/migration`, but you can modify that location by setting `spring.flyway.locations`. This is a comma-separated list of one or more `classpath:` or `filesystem:` locations. For example, the following configuration would search for scripts in both the default classpath location and the `/opt/migration` directory:

```
spring.flyway.locations=classpath:db/migration,filesystem:/opt/migration
```

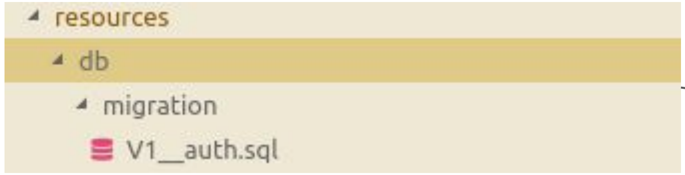
You can also add a special `{vendor}` placeholder to use vendor-specific scripts. Assume the following:

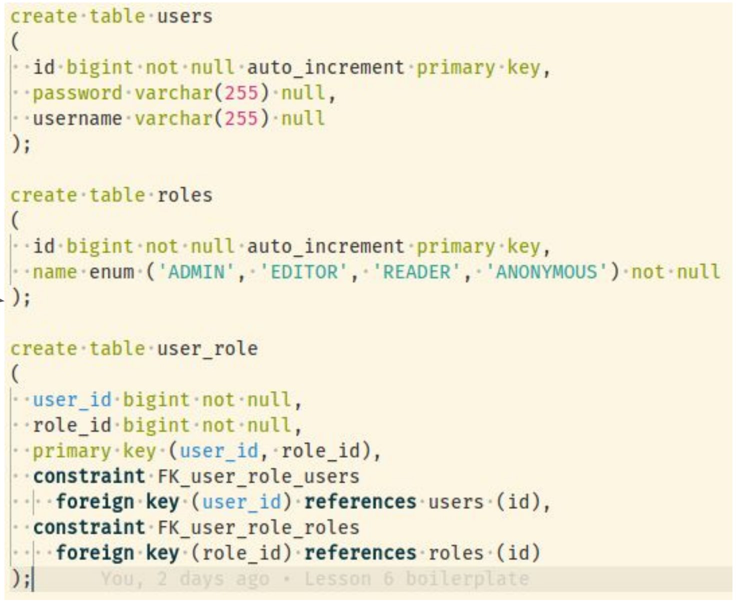
```
spring.flyway.locations=classpath:db/migration/{vendor}
```

Rather than using `db/migration`, the preceding configuration sets the folder to use according to the type of the database (such as `db/migration/mysql` for MySQL). The list of supported databases is available in DatabaseDriver.

Write migration files for saving data about application users.

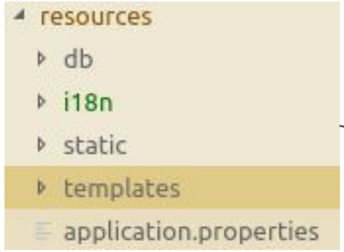
src/main/resources/db/migration/V1__auth.sql

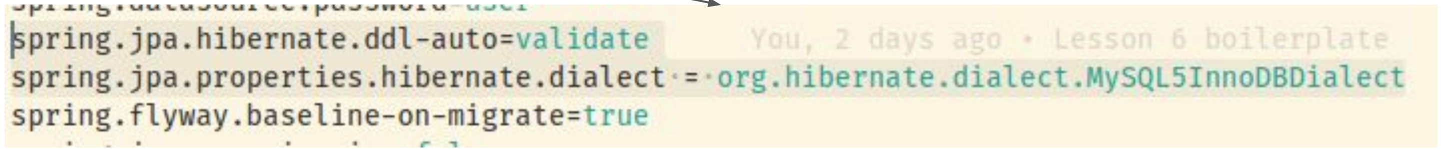




```sql
create table users
(
  id bigint not null auto_increment primary key,
  password varchar(255) null,
  username varchar(255) null
);

create table roles
(
  id bigint not null auto_increment primary key,
  name enum ('ADMIN', 'EDITOR', 'READER', 'ANONYMOUS') not null
);

create table user_role
(
  user_id bigint not null,
  role_id bigint not null,
  primary key (user_id, role_id),
  constraint FK_user_role_users
    foreign key (user_id) references users (id),
  constraint FK_user_role_roles
    foreign key (role_id) references roles (id)
);      You, 2 days ago • Lesson 6 boilerplate
```

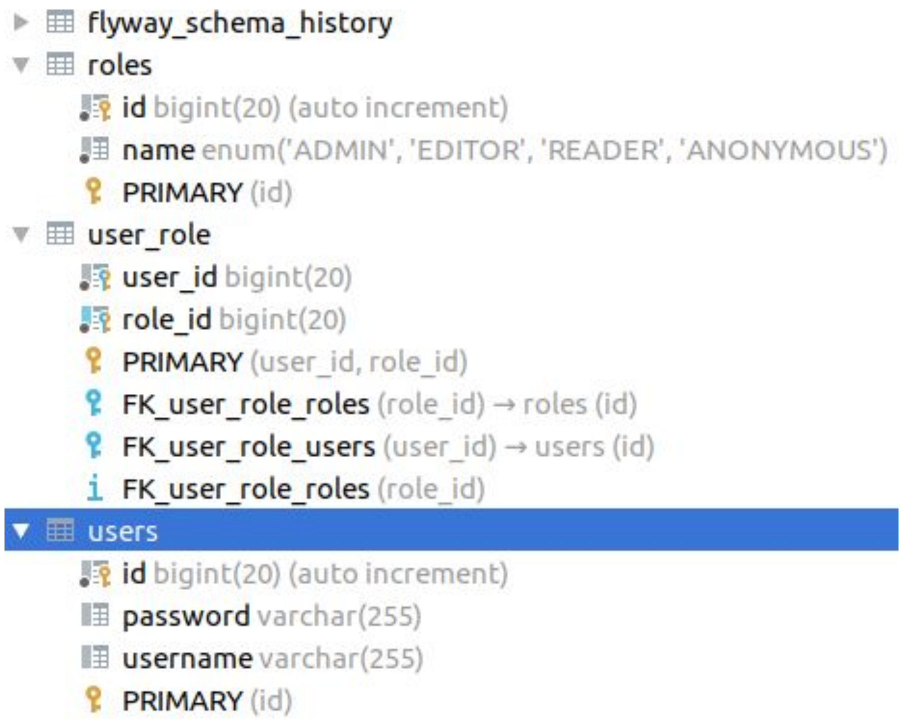
src/main/resources/application.properties





```properties
spring.jpa.hibernate.ddl-auto=validate      You, 2 days ago • Lesson 6 boilerplate
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect
spring.flyway.baseline-on-migrate=true
```

On application start up new tables will be created if their doesn’t exist.



```
▶ flyway_schema_history
▼ roles
    id bigint(20) (auto increment)
    name enum('ADMIN', 'EDITOR', 'READER', 'ANONYMOUS')
    PRIMARY (id)
▼ user_role
    user_id bigint(20)
    role_id bigint(20)
    PRIMARY (user_id, role_id)
    FK_user_role_roles (role_id) → roles (id)
    FK_user_role_users (user_id) → users (id)
    FK_user_role_roles (role_id)
▼ users
    id bigint(20) (auto increment)
    password varchar(255)
    username varchar(255)
    PRIMARY (id)
```

How to describe database relations with JPA Data?

- @ManyToMany
- @JoinTable
- @JoinColumn
- @Enumarated

Create two models Users and Roles that have many to many relation

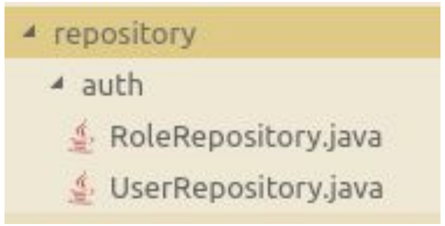
https://docs.oracle.com/javaee/7/api/javax/persistence/ManyToMany.html
https://hellokoding.com/jpa-many-to-many-relationship-mapping-example-with-spring-boot-maven-and-mysql/

Create UserRepository and RoleRepository

Create a UserRepository that extends JpaRepository and have findByUsername abstract method

Create a RoleRepository that extends JpaRepository

src/main/java/lt/baltictalents/lesson6/repository

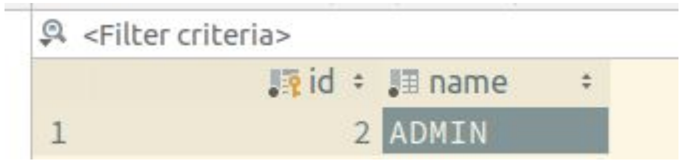


```
repository
  auth
    RoleRepository.java
    UserRepository.java
```

Create UserRestController.java (@RestController) that will return users with roles.

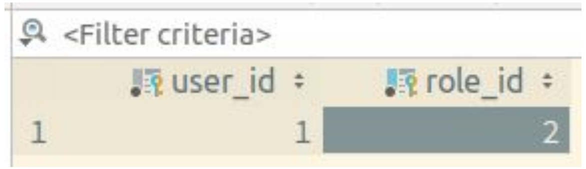
```java
package lt.baltictalents.lessons678.http.contollers;

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

@RestController
public class UsersRestController {
   @Autowired
   UserRepository userRepository;

   @GetMapping("/users")
   public List<UserPayload> getUsers() {
        ...
   }
}
```

Populate users and roles tables with faked data



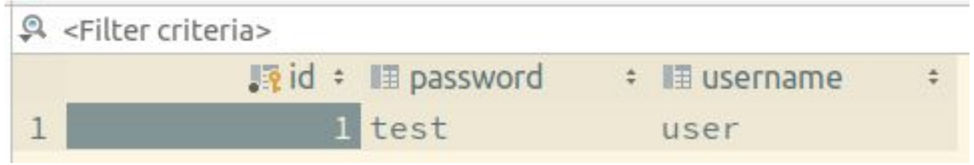
<Filter criteria>

	id	name
1	2	ADMIN



<Filter criteria>

	user_id	role_id
1	1	2

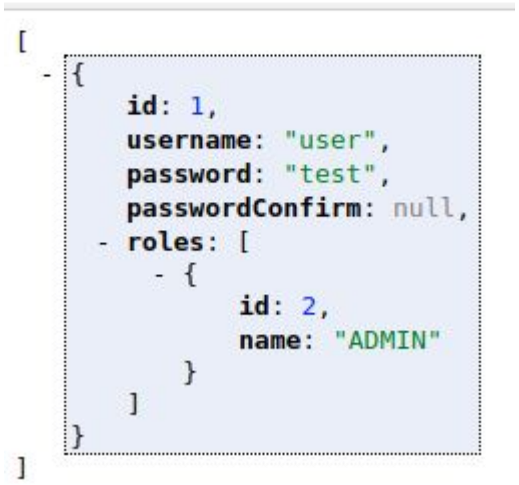


<Filter criteria>

	id	password	username
1	1	test	user

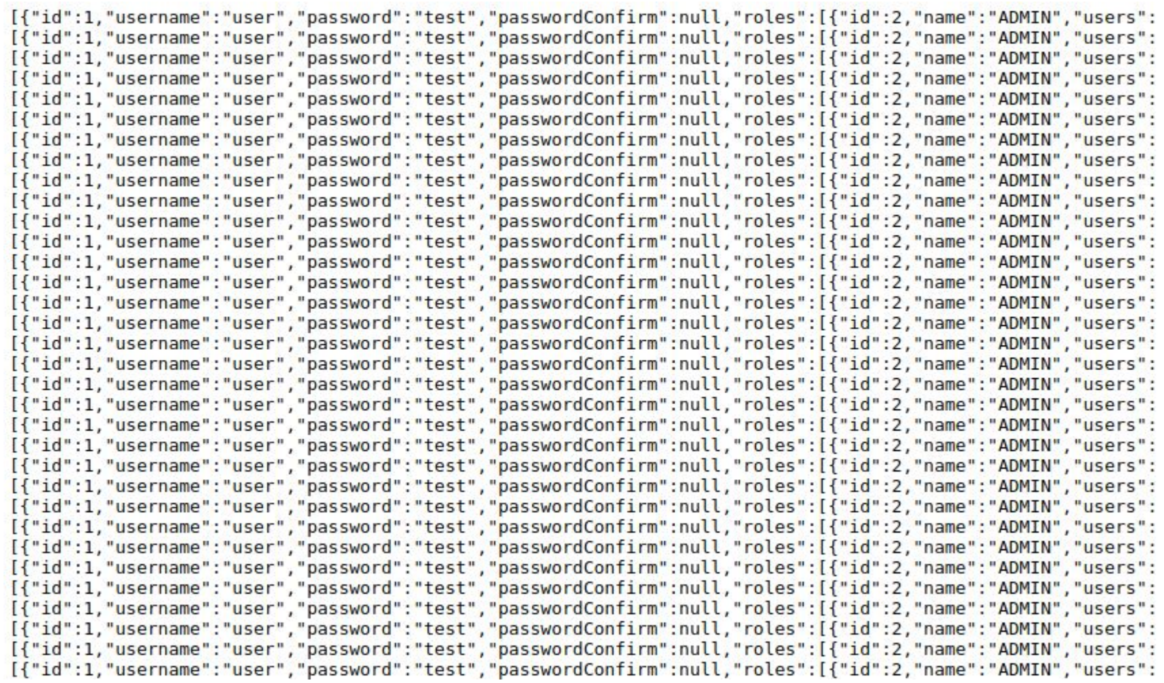
Result?

Why you don’t see this?



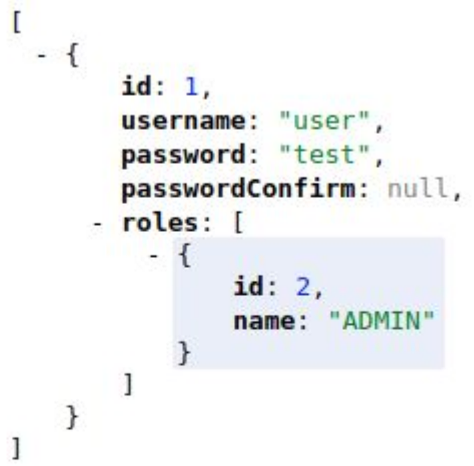
```
[
  - {
      id: 1,
      username: "user",
      password: "test",
      passwordConfirm: null,
    - roles: [
        - {
              id: 2,
              name: "ADMIN"
          }
      ]
  }
]
```

Why you see this?

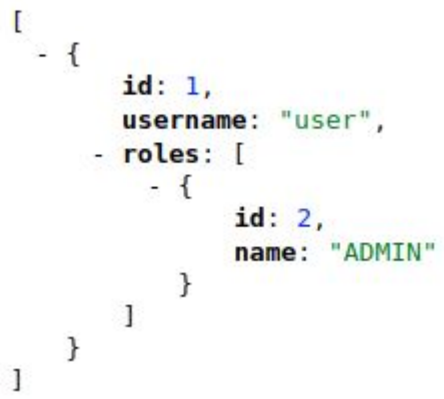


```
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
[{"id":1,"username":"user","password":"test","passwordConfirm":null,"roles":[{"id":2,"name":"ADMIN","users":
```

How to hide some fields in your users rest controller output



```
[
  - {
        id: 1,
        username: "user",
        password: "test",
        passwordConfirm: null,
      - roles: [
          - {
                id: 2,
                name: "ADMIN"
            }
        ]
    }
]
```



```
[
  - {
        id: 1,
        username: "user",
      - roles: [
          - {
                id: 2,
                name: "ADMIN"
            }
        ]
    }
]
```

Let’s create UserService and UserDetailsService

What is Spring security?

Spring Security is a powerful and highly customizable authentication and access-control framework. It is the de-facto standard for securing Spring-based applications.

Spring Security is a framework that focuses on providing both authentication and authorization to Java applications. Like all Spring projects, the real power of Spring Security is found in how easily it can be extended to meet custom requirements

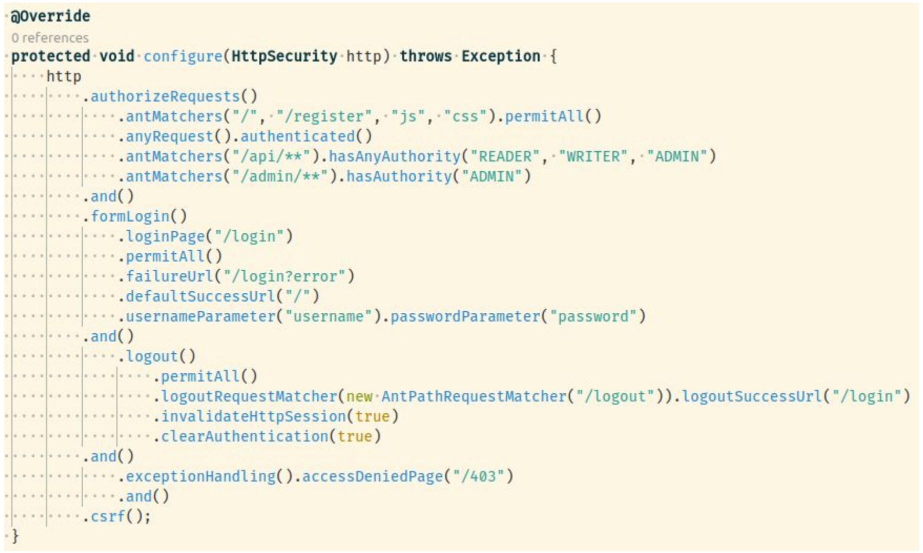
Features:

- Comprehensive and extensible support for both Authentication and Authorization
- Protection against attacks like session fixation, clickjacking, cross site request forgery, etc
- Servlet API integration
- Optional integration with Spring Web MVC
- Much more…

https://docs.spring.io/spring-security/site/docs/current/reference/html/jc.html

Basic spring security configuration

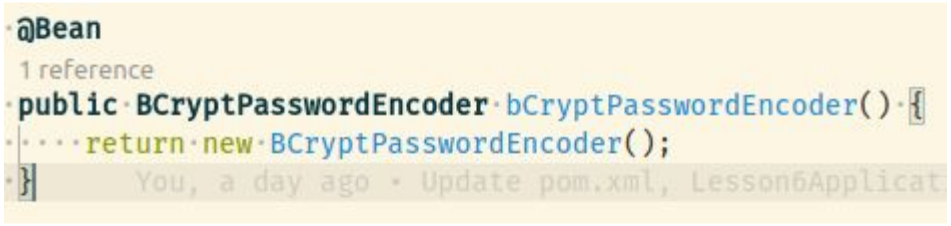
- Create security config java file that will extends `WebSecurityConfigurerAdapter`
- Annotated with @Configuration, @EnableWebSecurity
- Override configure method where main authentication logic is



```
@Override
0 references
protected void configure(HttpSecurity http) throws Exception {
    http
        .authorizeRequests()
            .antMatchers("/", "/register", "js", "css").permitAll()
            .anyRequest().authenticated()
            .antMatchers("/api/**").hasAnyAuthority("READER", "WRITER", "ADMIN")
            .antMatchers("/admin/**").hasAuthority("ADMIN")
        .and()
        .formLogin()
            .loginPage("/login")
            .permitAll()
            .failureUrl("/login?error")
            .defaultSuccessUrl("/")
            .usernameParameter("username").passwordParameter("password")
        .and()
            .logout()
                .permitAll()
                .logoutRequestMatcher(new AntPathRequestMatcher("/logout")).logoutSuccessUrl("/login")
                .invalidateHttpSession(true)
                .clearAuthentication(true)
        .and()
            .exceptionHandling().accessDeniedPage("/403")
            .and()
        .csrf();
}
```

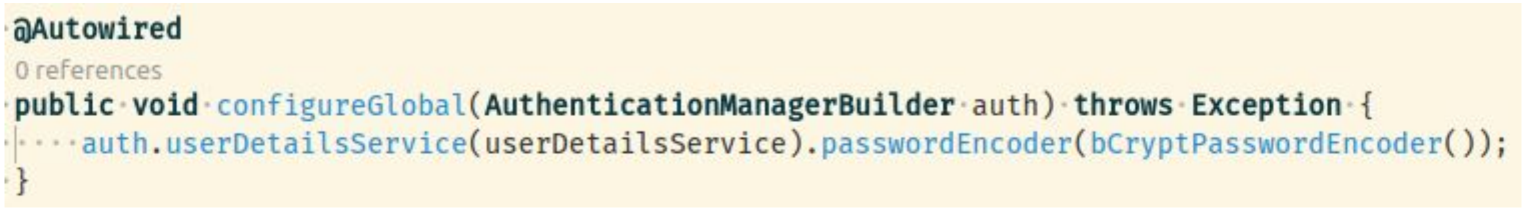
Extend spring security to meet our needs

- Set up password encoder



```
@Bean
1 reference
public BCryptPasswordEncoder bCryptPasswordEncoder() {
    return new BCryptPasswordEncoder();
}       You, a day ago • Update pom.xml, Lesson6Applicat
```

- Set up userDetailsService to store user data in database

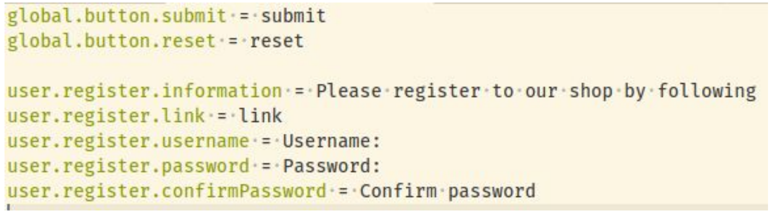


```
@Autowired
0 references
public void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {
    auth.userDetailsService(userDetailsService).passwordEncoder(bCryptPasswordEncoder());
}
```

Spring boot internationalization

```java
@Configuration
@EnableWebMvc
public class WebMvcConfig implements WebMvcConfigurer {
   @Bean
   public LocaleResolver localeResolver() {
       SessionLocaleResolver slr = new SessionLocaleResolver();
       slr.setDefaultLocale(Locale.ENGLISH);
       return slr;
   }

   @Bean
   public LocaleChangeInterceptor localeChangeInterceptor() {
       LocaleChangeInterceptor lci = new LocaleChangeInterceptor();
       lci.setParamName("lang");
       return lci;
   }

   @Bean
   public MessageSource messageSource() {
       ResourceBundleMessageSource messageSource = new ResourceBundleMessageSource();
       messageSource.setBasename("i18n/messages");
       messageSource.setDefaultEncoding("UTF-8");
       return messageSource;
   }

   @Override
   public void addInterceptors(InterceptorRegistry registry) {
       registry.addInterceptor(localeChangeInterceptor());
   }
}
```

src/main/resources/i18n/messages_en.properties



```properties
global.button.submit = submit
global.button.reset = reset

user.register.information = Please register to our shop by following
user.register.link = link
user.register.username = Username:
user.register.password = Password:
user.register.confirmPassword = Confirm password
```

Register and login page creation

- Let’s create register page that will let for us to create users;
- Let’s create login page that will let for us to create users;
- Let’s create accounts page that will list all users with roles;

Models data validation

Validation is done by using validation constraints:

```
javax.validation.constraints.*
import org.hibernate.validator.constraints.*
```

@NotBlank @Size

@NotNull @Lenght

@ScriptAssert @Region

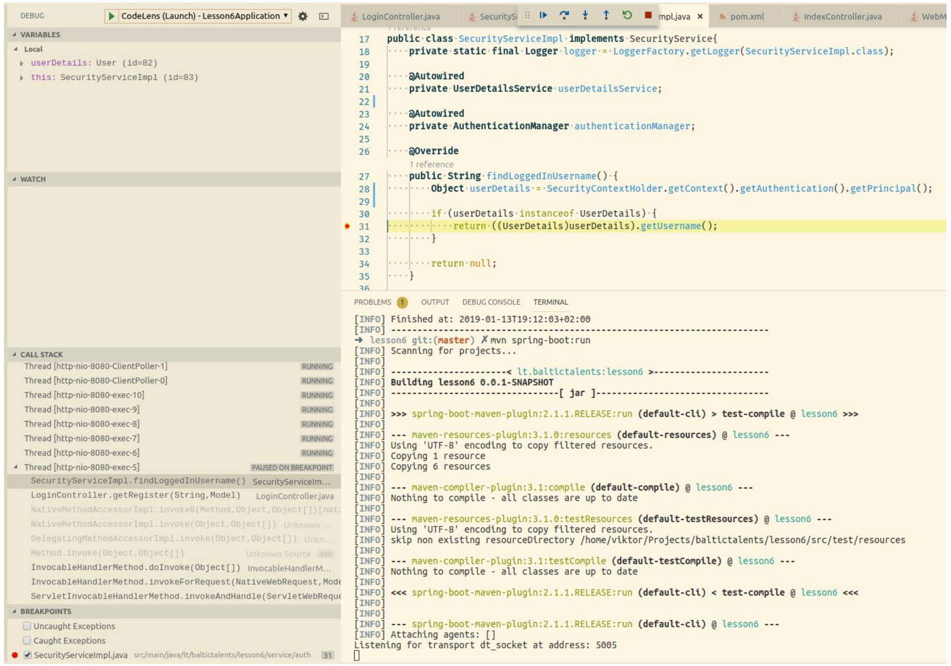
and more.

How to debug spring boot application

- Add additional debug configuration for maven / gradle

```
<configuration>
  <jvmArguments>
     -Xdebug
-Xrunjdwp:transport=dt_socket,server=y,suspend=y,address=5005
  </jvmArguments>
</configuration>
```

- Run application (e.g. mvn spring-boot:run) and attach debug in IDE



```
userDetails: User (id=82)
this: SecurityServiceImpl (id=83)
```

```java
public class SecurityServiceImpl implements SecurityService{
    private static final Logger logger = LoggerFactory.getLogger(SecurityServiceImpl.class);

    @Autowired
    private UserDetailsService userDetailsService;

    @Autowired
    private AuthenticationManager authenticationManager;

    @Override
    1 reference
    public String findLoggedInUsername() {
        Object userDetails = SecurityContextHolder.getContext().getAuthentication().getPrincipal();

        if (userDetails instanceof UserDetails) {
            return ((UserDetails)userDetails).getUsername();
        }

        return null;
    }
```

```
Thread [http-nio-8080-ClientPoller-1]        RUNNING
Thread [http-nio-8080-ClientPoller-0]        RUNNING
Thread [http-nio-8080-exec-10]               RUNNING
Thread [http-nio-8080-exec-9]                RUNNING
Thread [http-nio-8080-exec-8]                RUNNING
Thread [http-nio-8080-exec-7]                RUNNING
Thread [http-nio-8080-exec-6]                RUNNING
Thread [http-nio-8080-exec-5]                PAUSED ON BREAKPOINT
  SecurityServiceImpl.findLoggedInUsername()  SecurityServiceIm...
  LoginController.getRegister(String,Model)   LoginController.java
  NativeMethodAccessorImpl.invoke0(Method,Object,Object[])[nati
  NativeMethodAccessorImpl.invoke(Object,Object[])  Unknown ...
  DelegatingMethodAccessorImpl.invoke(Object,Object[])  Unkn...
  Method.invoke(Object,Object[])              Unknown Source 498
  InvocableHandlerMethod.doInvoke(Object[])   InvocableHandlerM...
  InvocableHandlerMethod.invokeForRequest(NativeWebRequest,Mode
  ServletInvocableHandlerMethod.invokeAndHandle(ServletWebReque
```

- [ ] Uncaught Exceptions
- [ ] Caught Exceptions
- [x] SecurityServiceImpl.java src/main/java/lt/baltictalents/lesson6/service/auth 31

```
[INFO] Finished at: 2019-01-13T19:12:03+02:00
[INFO] ------------------------------------------------------------------------
➜ lesson6 git:(master) ✗ mvn spring-boot:run
[INFO] Scanning for projects...
[INFO]
[INFO] ---------------------< lt.baltictalents:lesson6 >----------------------
[INFO] Building lesson6 0.0.1-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] >>> spring-boot-maven-plugin:2.1.1.RELEASE:run (default-cli) > test-compile @ lesson6 >>>
[INFO]
[INFO] --- maven-resources-plugin:3.1.0:resources (default-resources) @ lesson6 ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO] Copying 6 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ lesson6 ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:3.1.0:testResources (default-testResources) @ lesson6 ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/viktor/Projects/baltictalents/lesson6/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ lesson6 ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] <<< spring-boot-maven-plugin:2.1.1.RELEASE:run (default-cli) < test-compile @ lesson6 <<<
[INFO]
[INFO]
[INFO] --- spring-boot-maven-plugin:2.1.1.RELEASE:run (default-cli) @ lesson6 ---
[INFO] Attaching agents: []
Listening for transport dt_socket at address: 5005
```

References

https://www.baeldung.com/spring-security-create-new-custom-security-expression
https://www.baeldung.com/spring-security-expressions-basic
https://github.com/eugenp/tutorials/tree/master/spring-boot/src/main/java/com/baeldung/internationalization
https://www.baeldung.com/spring-boot-internationalization
https://www.baeldung.com/spring-security-login
https://www.thymeleaf.org/doc/articles/springmvcaccessdata.html
https://spring.io/guides/gs/handling-form-submission/
https://marcin-chwedczuk.github.io/hello-hibernate-validator