

Федеральное государственное автономное  
образовательное учреждение  
высшего образования  
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №6**

Конфигурация Spring Security

тема

Преподаватель

\_\_\_\_\_

подпись, дата

А.С. Черниговский

инициалы, фамилия

Студент КИ18-17/16 031830504

номер группы, зачетной книжки

\_\_\_\_\_

подпись, дата

Е.В. Железкин

инициалы, фамилия

Красноярск 2020

## 1 Цель работы

Познакомиться с настройкой безопасности в Spring.

## 2 Задача работы

Взять практическое задание №5 и добавить следующий функционал:

1) Добавить простейшую страницу регистрации. Пользователь вводит свои логин и пароль, и данная информация вносится в базу данных, пользователю присваивается роль пользователя (User) приложения.

2) Добавить простейшую форму аутентификации. Форма должна быть создана студентом, а не автоматически сгенерированной Spring.

3) В приложении должен быть предусмотрен пользователь — администратор, с, отличной от роли User, ролью администратора (Admin).

4) Разграничить уровни доступа к страницам приложения. Пользователь (User) имеет доступ только к страницам просмотра всех записей и запросов.

Администратор (Admin) имеет возможность добавлять, редактировать и удалять записи.

5) Информация о пользователях и их ролях должна храниться в базе данных. Способ хранения — на усмотрение студента.

6) Предусмотреть возможность пользователю выйти из приложения (logout).

7) Продемонстрировать умение настраивать безопасность на уровне представлений. Для этого реализуйте приветствие пользователя после его входа и отображение элемента на основе его роли.

*Вариант 9 - Книга*

### 3 Ход работы

1) Официальный мануал, взятый за основу данной работы [https://spring.io/guides/gs/securing-web/#run\\_the\\_app](https://spring.io/guides/gs/securing-web/#run_the_app) Добавлена конфигурация безопасности проекта (модуль безопасности, рисунок 1):

```
43      @Override
44      @
45      {
46          //      auth.inMemoryAuthentication()
47          //          .withUser("u")
48          //          .password("{noop}123")
49          //          .authorities("USER");
50
51          auth.jdbcAuthentication()
52              .dataSource(dataSource)
53              .usersByUsernameQuery(
54                  "select username, password, 'true' from users " +
55                  "where username=?"
56              )
57              .authoritiesByUsernameQuery(
58                  "select username, role from user_aut " +
59                  "where username=?");
60
61      @Override
62      @
63      {
64          http
65              .authorizeRequests() ExpressionUrlAuthorizationConfigurer<H>.ExpressionInterceptUrlRegistry
66                  .antMatchers(...antPatterns: "/", "/registration").permitAll()
67                  .antMatchers(...antPatterns: "/books").hasAnyAuthority(...authorities: "USER", "ADMIN")
68                  .antMatchers("/books/new").hasAuthority("ADMIN")
69                  .antMatchers("/books/**/edit").hasAuthority("ADMIN")
70                  .anyRequest().authenticated()
71              .and() HttpSecurity
72                  .formLogin() FormLoginConfigurer<HttpSecurity>
73                  .loginPage("/login")
74                  .permitAll()
75              .and() HttpSecurity
76                  .logout() LogoutConfigurer<HttpSecurity>
77                  .permitAll();
78      }
```

Рисунок 1 – Часть конфигурации (модуль security), SpringSecurityConfig.java

2) Реализован контроллер пользователя(доступа) (рисунок 2):

```

@Controller
@RequestMapping
public class UserController
{
    private final UserServiceImpl userService;
    private final BookServiceImpl bookService;

    public UserController(@Qualifier("userService") UserServiceImpl userService, @Qualifier("bookService") BookService bookService)
    {
        this.userService = userService;
        this.bookService = bookService;
    }

    @GetMapping("/profile")
    public String getUserProfile(@AuthenticationPrincipal org.springframework.security.core.userdetails.User user,
                                Model model)
    {
        model.addAttribute(s: "user", userService.findByUsername(user.getUsername()).orElse(new User()));
        model.addAttribute(s: "springUser", user);
        boolean isAdmin = Arrays.asList(user.getAuthorities().toArray())
            .contains(new SimpleGrantedAuthority( role: "ADMIN"));
        model.addAttribute(s: "isAdmin", isAdmin);
        model.addAttribute(s: "count", bookService.count());
        return "security/profile";
    }
}

```

Рисунок 2 – Часть контроллера пользователя, UserController.java

3) Созданы и настроены формы аутентификации (рисунок 3):

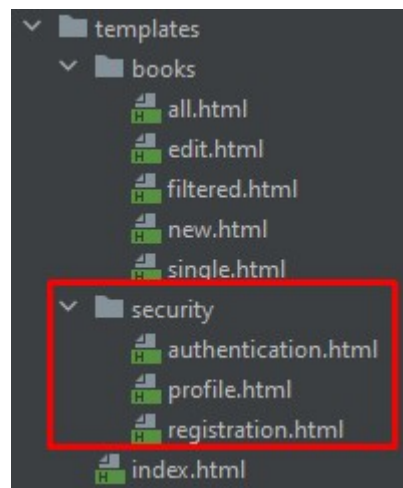


Рисунок 3 – Часть структуры проекта, отвечающей за представление (модуль view)

4) Интегрированы модули для работы с сущностью(книгой) из предыдущих работ (рисунок 4):

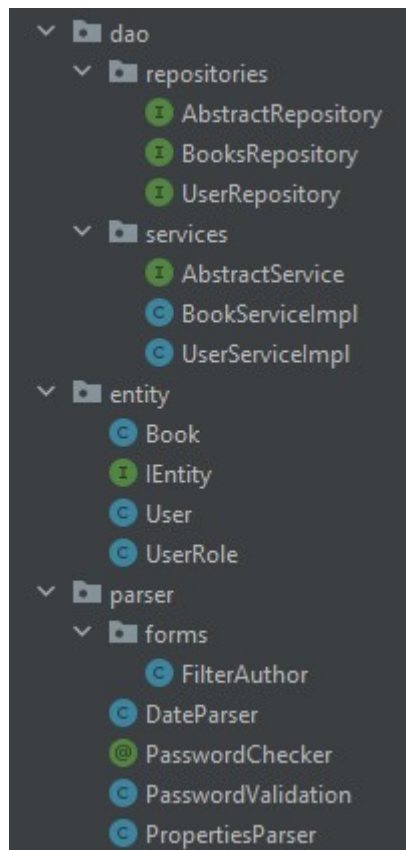


Рисунок 4 – Часть структуры проекта (модуль model) WebApp.java

5) Запущенное приложение (рисунки 5-8):

## Регистрация:

Имя пользователя:

Такой никнейм уже занят!

Пароль:

Пароли не совпадают!

Повтор пароля:

Биография:

[Main](#)

Рисунок 5 – Проверка на регистрации

## Регистрация:

Имя пользователя:   
**Некорректное имя пользователя (2-30 символов)**  
Пароль:   
**Некорректная длина пароля**  
Повтор пароля:   
Биография:

---

[Main](#)

Рисунок 6 – Проверка на регистрации (2)

## Ваш профиль:

**Имя пользователя: гыук322**

Ваши роли:  
USER;

Ваше био: www

## Книги

Количество книг в БД: 6

- [Просмотр книг](#)

---

[Main](#)

Рисунок 7 – Авторизация под пользователем, имеющем роль «USER»

# Ваш профиль:

Имя пользователя: admin

Ваши роли:  
ADMIN;

Памятка для админа (123456789)

Ваше био: Admin

## Книги

Количество книг в БД: 6

- [Просмотр книг](#)
- [Добавить книгу](#)

Выйти

[Main](#)

Рисунок 8 – Авторизация под пользователем, имеющем роль «ADMIN»

### 4 Вывод

В ходе данной лабораторной работы были изучены основы работы с настройкой безопасности в Spring.