

**Втора домашна задача**  
**Информациска безбедност**  
**Љубомир Давитков 216092**  
**Автентикација**

Системот за автентикација за домашна работа е веб-апликација изградена со помош на Java, Spring MVC, Spring Boot, Spring Data JPA, PostgreSQL, HTML, CSS и Thymeleaf.

**Користени технологии:**

Java 17: Основниот програмски јазик за апликацијата.

Spring Boot 3.1.4: Го поедноставува поставувањето и развојот на пролетните апликации.

Spring MVC: Управува со веб-слојот на апликацијата.

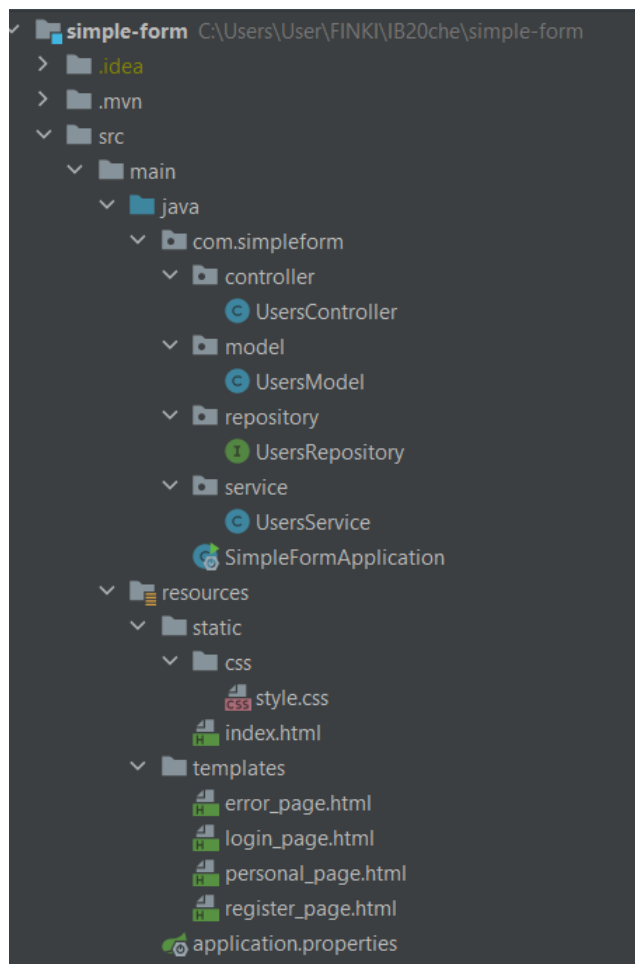
Spring Data JPA: Го поедноставува пристапот до податоци со користење на Java Persistence API (JPA).

PostgreSQL: Служи како релациона база на податоци за складирање на податоци од апликацијата.

HTML и Thymeleaf: Се користи за прикажување динамични веб-страници и шаблони.

CSS: Го подобрува визуелниот стил на апликацијата.

**Структурата на апликацијата:**



### Контролерот на апликацијата:

```
Ljubomir
@GetMapping("/register")
public String getRegisterPage(Model model) {
    model.addAttribute(attributeName: "registerRequest", new UsersModel());
    return "register_page";
}
```

Се справува со барањата GET за крајната точка „/register“.

Го подготвува моделот со празен UsersModel и го враќа шаблонот „register\_page“.

```
Ljubomir
@GetMapping("/login")
public String getLoginPage(Model model) {
    model.addAttribute(attributeName: "loginRequest", new UsersModel());
    return "login_page";
}
```

Се справува со барањата GET за крајната точка „/login“.

Го подготвува моделот со празен UsersModel и го враќа шаблонот „login\_page“.

```
1 Ljubomir
2 @PostMapping("/register")
3 public String register (@ModelAttribute UsersModel usersModel){
4     System.out.println("register request: " + usersModel);
5     UsersModel registeredUser = userService.registerUser(usersModel.getLogin(), usersModel.getPassword(), usersModel.getEmail());
6     return registeredUser == null ? "error_page" : "redirect:/login";
7 }
8 }
```

Се справува со барањата POST за крајната точка „/register“.

Регистрира корисник користејќи го дадениот UsersModel.

Се пренасочува на „/login“ доколку регистрацијата е успешна; во спротивно, се пренасочува на „error\_page“.

```
1 Ljubomir
2 @PostMapping("/login")
3 public String login (@ModelAttribute UsersModel usersModel, Model model){
4     System.out.println("login request: " + usersModel);
5     UsersModel authenticated = userService.authenticate(usersModel.getLogin(), usersModel.getPassword());
6     if (authenticated != null){
7         model.addAttribute("attributeName: userLogin", authenticated.getLogin());
8         return "personal_page";
9     } else {
10         return "error_page";
11     }
12 }
13 }
```

Се справува со барањата POST за крајната точка „/login“.

Го автентифицира корисникот користејќи го дадениот UsersModel.

Пренасочува кон „personal\_page“ со автентифицирано најавување на корисникот доколку е успешно; во спротивно, се пренасочува на „error\_page“.

**Service делот на апликацијата:**

```
1 usage Ljubomir
2 public UsersModel registerUser(String login, String password, String email) {
3     if (login == null || password == null) {
4         return null;
5     } else {
6         if (usersRepository.findFirstByLogin(login).isPresent()) {
7             System.out.println("Duplicate login");
8             return null;
9         }
10        UsersModel usersModel = new UsersModel();
11        usersModel.setLogin(login);
12        usersModel.setPassword(password);
13        usersModel.setEmail(email);
14        return usersRepository.save(usersModel);
15    }
16 }
```

Регистрира нов корисник со наведеното најавување, лозинка и е-пошта.

Ги потврдува влезните параметри и проверува дали има дупликат најавувања.

Го враќа регистрираниот UsersModel доколку е успешен, или е нулатен во случај на неуспех на валидацијата или дупликат најава.

```
1 usage  👤 Ljubomir
public UserModel authenticate(String login, String password) {
    return usersRepository.findByLoginAndPassword(login, password).orElse( other: null);
}
```

Го автентифицира корисникот со наведеното најавување и лозинка.

Го бара корисничкото складиште за соодветен корисник.

Го враќа автентифицираниот UsersModel ако е успешен или нула ако автентификацијата не успее.

**Repository делот на апликацијата:**

```
3 usages  👤 Ljubomir
public interface UsersRepository extends JpaRepository<UsersModel, Integer> {

    1 usage  👤 Ljubomir
    Optional<UsersModel> findByLoginAndPassword(String login, String password);

    1 usage  👤 Ljubomir
    Optional<UsersModel> findFirstByLogin(String login);
}
```

UsersRepository е интерфејс за складиште за пролетни податоци JPA одговорен за ракување со операциите на базата на податоци поврзани со ентитетот UsersModel.