

ВЕЛИКОТЪРНОВСКИ УНИВЕРСИТЕТ
„СВ. СВ. КИРИЛ И МЕТОДИЙ“



Курсова работа
по
Уеб програмиране с Java

Изготвил: Яница Красиминова Коеджикова

Специалност: Софтуерно инженерство

Фак. №: 20042

Курс: Четвърти

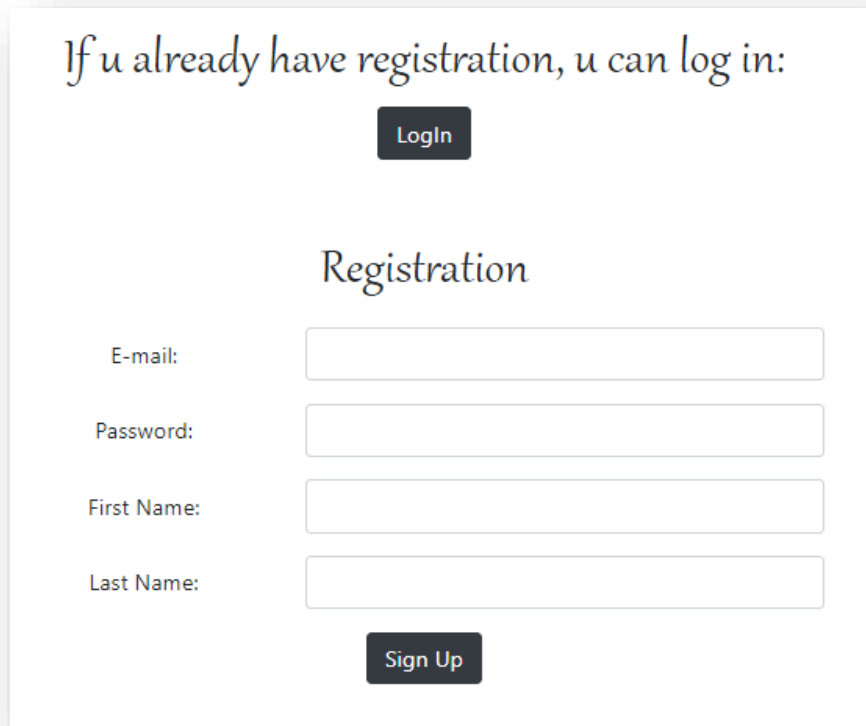
Група: Първа

Проверил: доц. д-р Тодор Йорданов Тодоров

I. Относно курсовата работа

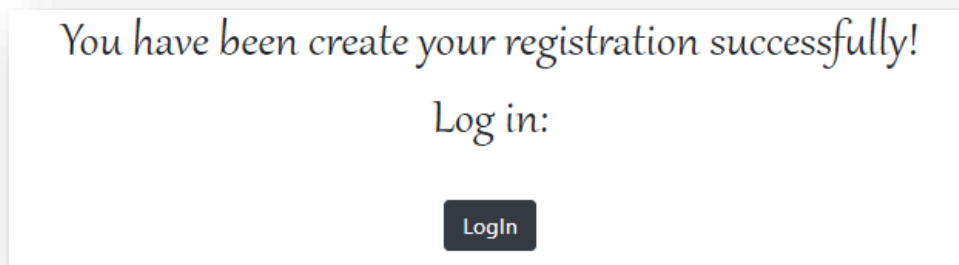
Курсовата работа уеб сайт на тема книги. Проекта е качен в github.com, линка е <https://github.com/YanitsaKoedzhikova/Java-Project>.

- При стартиране на проекта се зарежда страницата „http://localhost:8080/“ , където потребителят може да се регистрира или да влезе във вече съществуващ профил.



The screenshot shows a web page for user registration. At the top, it says "If u already have registration, u can log in:" followed by a dark button labeled "LogIn". Below this is the heading "Registration". There are four input fields: "E-mail:", "Password:", "First Name:", and "Last Name:". At the bottom of the form is a dark button labeled "Sign Up".

- При регистрация, потребителят трябва да попълни полета за имейл, парола, име и фамилия. Регистрацията става чрез бутона „Sign Up“;
- След натискане на бутона „Sign Up“, ако всички полета са вярно попълнени потребителят се праща на страницата за успешно регистрирани потребители.



The screenshot shows a confirmation message: "You have been create your registration successfully!". Below the message is the text "Log in:" followed by a dark button labeled "LogIn".

- След успешна регистрация, потребителят може да избере бутона “LogIn”. Тогава потребителят се праща на страницата “LogIn”;



Please sign in

qnicaa395@abv.bg

.....

Sign in

- На “LogIn” страницата потребителят трябва да въведе имейл и парола. При вярно попълнени полета и натискане на бутона “Sign in”, потребителят успешно влиза в профила си;
- При успешно влизане в профила потребителят се препраща на страницата “Home”.
- След като потребителят вече е в профила си, той може да достъпно страницата users където, може да се види таблица с всички регистрирани потребители;

Home Users Sign Out

Welcome Yanitsa Koedzhikova

List of all users

E-mail	First Name	Last Name
qnicaa395@abv.bg	Yanitsa	Koedzhikova
test@abv.bg	testname	testname

- В Номе страницата, потребителят може да види таблица с всички книги;
- Възможно е добавянето на нови книги от бутона “Create book”;
- Възможно е редактирането на всяка от книгите от бутона “Edit”;
- Възможно е изтриването на всяка от книгите от бутона “Delete”;
- Възможно е търсенето по име, автор или жанр в полето “Search” и натискане на бутона “Search”. Възможно е зачистване на търсенето, чрез бутона “Clear”;

Search:

Book	Author	Genre	Edit	Delete
The Witcher	Andrzej Sapkowski	Fantasy	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
Pet Sematary	Stephen King	Horror	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
Wuthering Heights	Emily Brontë	Tragedy	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>

- При натискане на бутона “Create book”, потребителят отива да страницата “Create”

Home Users Sign Out

Create new books

Book Name

Book Author

Book Genre

- За създаване на нова книга, потребителят трябва да попълни полетата “Book Name”, “Book Author” и “Book Genre”. При натискане на бутона “Create” книгата се добавя и потребителят се препраща на страницата Home;
- При избиране на бутона “Edit”, на която и да е от книгите, потребителят отива на страницата „Update”;





- За запазване на редактираните полета, потребителят трябва да избере бутона “Update”, след което потребителят се препраща с страницата “Home”;

II. База данни

Базата данни е създадена в phpMyAdmin с име „bookproject” и таблици `tbl_book`, `users` и автоматично генерираната таблица `hibernate_sequence`.

Таблица	Действие	Редове	Тип	Коляция	Размер	Загубено място
<input type="checkbox"/> <code>hibernate_sequence</code>	★ Прелистване Структура Търсене Вмъкване Изчистване Изтриване	1	InnoDB	utf16_unicode_ci	16.0 KB	-
<input type="checkbox"/> <code>tbl_book</code>	★ Прелистване Структура Търсене Вмъкване Изчистване Изтриване	3	InnoDB	utf16_unicode_ci	16.0 KB	-
<input type="checkbox"/> <code>users</code>	★ Прелистване Структура Търсене Вмъкване Изчистване Изтриване	2	InnoDB	utf16_unicode_ci	32.0 KB	-
3 таблици	Сума	6	InnoDB	utf16_unicode_ci	64.0 KB	0 B

- Таблицата `tbl_book`, служи за съхраняване на книгите. Таблицата е генерирана от `Book.java`, благодарение на `@Entity`, което информира Hibernate да създаде таблица от класа. Таблицата съдържа полетата: `book_id`, `book_name`,
- `book_author`, `book_genre`;

				book_id	book_author	book_genre	book_name
<input type="checkbox"/>	 Редакция	 Копиране	 Изтриване	4	Andrzej Sapkowski	Fantasy	The Witcher
<input type="checkbox"/>	 Редакция	 Копиране	 Изтриване	6	Stephen King	Horror	Pet Sematary
<input type="checkbox"/>	 Редакция	 Копиране	 Изтриване	8	Emily Brontë	Tragedy	Wuthering Heights

- Таблицата users, служи за съхраняване на регистрираните потребители. Таблицата е генерирана от User.java, благодарение на @Entity, което информира Hibernate да създаде таблица от класа. Таблицата съдържа полетата: id, email, first_name, last_name, password;

				id	email	first_name	last_name	password
<input type="checkbox"/>	Редакция	Копиране	Изтриване	8	qnicaa395@abv.bg	Yanitsa	Koedzhikova	\$2a\$10\$T059j235EN8KgJOvb5MVPegspGo/bmfOVtQqyN31jAb...
<input type="checkbox"/>	Редакция	Копиране	Изтриване	9	test@abv.bg	testname	testname	\$2a\$10\$nce6jBGHFD6vzSZKasK1AOf1mDW.qVdhsnn8Zl/8E8M...

☐ Маркиране всички
 Когато има отметка:
 Редакция
 Копиране
 Изтриване
 Експорт

- Базата данни е експортната и качена в архивирания файл.

III. Реализация на проекта

За създаването на курсовия проект е използвана технологията Spring Framework, средата за разработка е Spring Tool Suite 4.

За стартирането на проекта е нужен хамър със стартирани apache and MySQL за връзка с базата данни създадена в phpMyAdminе.

Използва се maven за да може spring boot да конфигурира всички dependencies автоматично. Това се случва във файла pom.xml.

Базата данни се свързва с проекта в файла application.properties.

```

1 # MySQL database connecting utility
2 spring.datasource.url=jdbc:mysql://localhost:3306/bookproject
3 spring.datasource.username=root
4 spring.datasource.password=
5
6 #JPA property utility
7 spring.jpa.hibernate.ddl-auto=update
8 spring.jpa.properties.hibernate.show_sql=true
9
10 #view pages property utility
11 spring.thymeleaf.prefix=classpath:/templates/
12 spring.thymeleaf.suffix=.html
13

```

Book.java в com.books.bookproject.model се създава за да се свърже проекта с базата данни, посочва се име на вече съществуваща таблица или записваме име за таблицата която искаме да бъде създадена от класа благодарение на @Entity, което информира Hibernate да създаде таблица. @Table: посочваме вече съществуваща таблица или записваме името на новата таблица. @GeneratedValue се използва за автоматично генериране на Primary Key/ ID. @Column се използва за указване на колоните на таблицата.

След посочване името на таблицата и нейните колони, може да се генерират get и set методите. (Десен бутон/ Source/ Generate getter and setters.)

```
@Entity
@Table(name = "tbl_book")
public class Book {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "book_id")
    private Long id;

    @Column(name = "book_name")
    private String name;

    @Column(name = "book_author")
    private String author;

    @Column(name = "book_genre")
    private String genre;
}
```

Случващото се User.java е същото както в Book.java. Създаване на таблица, и генериране на set и get методи.

BookRepository interface се създава в com.books.bookproject.repository. BookRepository extends JpaRepository. JpaRepository осигурява CRUD операциите за Book model class и работи с Long ID type за primary key на приложението. В BookRepository, декларираме и заявката за търсенето на книги в Home страницата.

```
package com.books.bookproject.repository;

import java.util.List;

public interface BookRepository extends JpaRepository<Book, Long> {

    @Query("SELECT p FROM Book p WHERE CONCAT(p.name, ' ', p.author, ' ', p.genre) LIKE %?1%")
    public List<Book> search(String keyword);

}
```

Случващото се в UserRepository е подобно. Разликата е в това, че за потребителите нямаме търсачка, поради което и не декларираме заявка за търсене. В User Repository дефинираме метода findByEmail (), който връща един потребителски обект въз основа на имейл.

BookService в com.books.bookproject.service съдържа бизнес логиката за create, read, update и delete заявките. Autowired е използван за object injection.

```
public List<Book> listAll(String keyword) {  
    if (keyword != null) {  
        return repository.search(keyword);  
    }  
    return repository.findAll();  
}
```

Тук използваме метода findAll, за изкарване на всички съществуващи книги. Search се използва за търсене измежду тези книги по зададена ключова дума в Search полето в Home страницата.

```
public void create(Book book) {  
    repository.save(book);  
}
```

Методът save се използва създаване и редактиране на книга.

```
public Book updateid(Long id) {  
    return repository.findById(id).get();  
}
```

Методът findById се използва за вземане на определен елемент според стойността на Id-а му. (При редакция вземаме елемента по Id, защото Id-а винаги е уникален).

```
public void delete(Long id) {  
    repository.deleteById(id);  
}
```

Методът deleteById се използва за изтриване. Отново е “byId“, защото когато изтриваме елемент го вземаме по Id му, защото Id е уникален.

В `com.books.bookproject.controller`, създаваме класа `BookController`, който действа като Spring MVC controller за `create`, `update` и `delete` заявки.

С `@RequestMapping` указваме URL-а, а на `return` казваме кое view искаме да бъде отворено след обработването на заявката.

Тези view-та се създават в `src/main/resources/templates`.

`CustomUserDetails` е клас от подтип `UserDetails` (дефиниран от Spring Security), който представлява потребител за удостоверяване.

В `CustomUserDetailsService` Spring Security извиква методът `loadUserByUsername()` за да удостовери потребителя и ако успее, се създава нов обект от тип `CustomUserDetails`, който представлява удостоверения потребител.

В `WebSecurityConfig` в методът `configure()`, задаваме правилото, че потребителят трябва да си създаде профил и да влезе в него за да види страницата `users`.

Също така конфигурираме страницата за вход по подразбиране (генерирана от Spring Security) с полета имейл и парола. Тук задаваме и потребителя да бъде пренасочен към home страницата след успешно влизане.

```
@Override
protected void configure(HttpSecurity http) throws Exception {
    http.authorizeRequests()
        .antMatchers("/users").authenticated()
        .anyRequest().permitAll()
        .and()
        .formLogin()
            .usernameParameter("email")
            .defaultSuccessUrl("/index")
            .permitAll()
        .and()
        .logout().logoutSuccessUrl("/").permitAll();
}
```