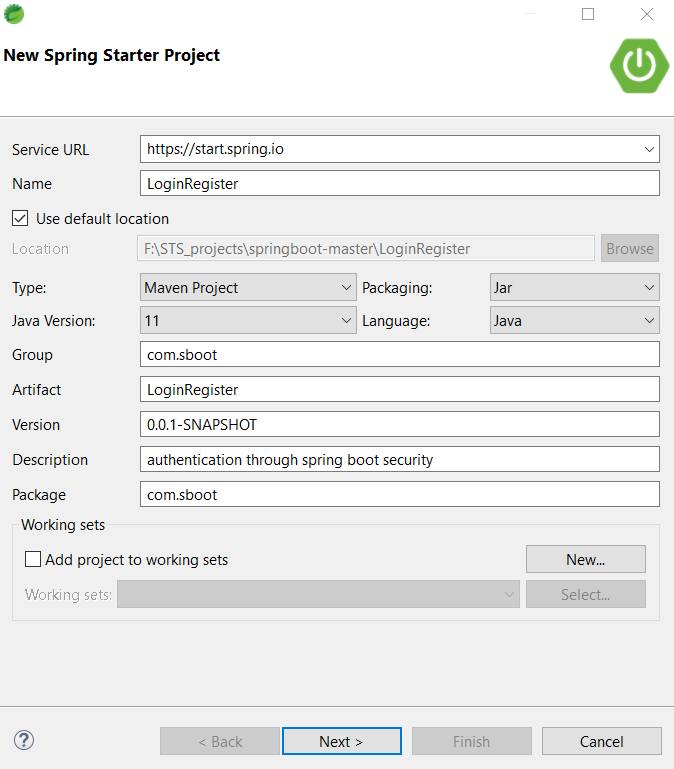
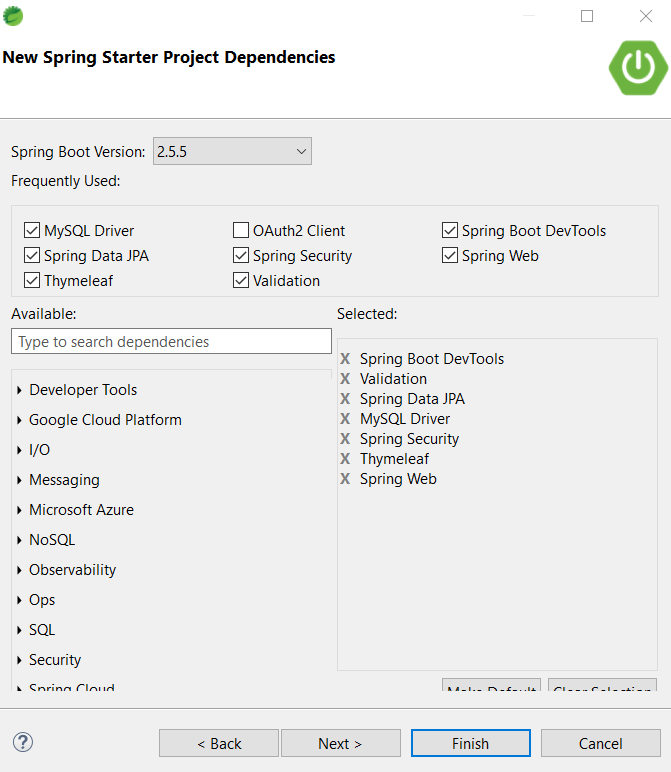
Implement Login and Register in Spring Boot using thymeleaf





Pom.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.5.5</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.sboot</groupId>

<artifactId>LoginRegister</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>LoginRegister</name>

<description>authentication through spring boot security</description>

<properties>

<java.version>11</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-validation</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.thymeleaf.extras</groupId>

<artifactId>thymeleaf-extras-springsecurity5</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

application.properties

server.port=8899

spring.datasource.name=ds

spring.datasource.url=jdbc:mysql://localhost:3306/loginregister?

spring.datasource.username=root

spring.datasource.password=Lufa@7917

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect

spring.jpa.hibernate.ddl-auto=update

spring.application.name = LoginRegister

Main Application File :

package com.sboot;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class LoginRegisterApplication {

public static void main(String[] args) {

SpringApplication.run(LoginRegisterApplication.class, args);

} }

Student Model :

package com.sboot.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import javax.validation.constraints.NotBlank;

@Entity

@Table(name="Student")

public class Student {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

private int stuid;

@NotBlank(message="Srudent Name field is required !!")

private String stuname;

@NotBlank(message="Student Email field is required !!")

private String stuemail;

@NotBlank(message="Password field is required !!")

private String password;

private String imgurl;

private String role;

public int getStuid() {

return stuid;

}

public void setStuid(int stuid) {

this.stuid = stuid;

}

public String getStuname() {

return stuname;

}

public void setStuname(String stuname) {

this.stuname = stuname;

}

public String getStuemail() {

return stuemail;

}

public void setStuemail(String stuemail) {

this.stuemail = stuemail;

}

public String getImgurl() {

return imgurl;

}

public void setImgurl(String imgurl) {

this.imgurl = imgurl;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public Student(int stuid, @NotBlank(message = "Srudent Name field is required !!") String stuname,

@NotBlank(message = "Student Email field is required !!") String stuemail,

@NotBlank(message = "Password field is required !!") String password, String imgurl, String role) {

super();

this.stuid = stuid;

this.stuname = stuname;

this.stuemail = stuemail;

this.password = password;

this.imgurl = imgurl;

this.role = role;

}

public Student() {

super();

// TODO Auto-generated constructor stub

}

@Override

public String toString() {

return "Student [stuid=" + stuid + ", stuname=" + stuname + ", stuemail=" + stuemail + ", password=" + password

+ ", imgurl=" + imgurl + ", role=" + role + "]";

}

}

StudentDao interface :

package com.sboot.dao;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.query.Param;

import com.sboot.model.Student;

public interface StudentDao extends JpaRepository<Student, Integer> {

@Query("select s from Student s where s.stuname = :stuname")

**public** Student getStudentByStuName(@Param("stuname") String stuname);

}

CustomStudentDetails

package com.sboot.config;

import java.util.Collection;

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.userdetails.UserDetails;

import com.sboot.model.Student;

public class CustomStudentDetails implements UserDetails {

private Student student;

@Override

public Collection<? extends GrantedAuthority> getAuthorities() {

SimpleGrantedAuthority simpleGrantedAuthority = **new** SimpleGrantedAuthority(student.getRole());

**return** List.*of*(simpleGrantedAuthority);

}

@Override

public String getPassword() {

return student.getPassword();

}

@Override

public String getUsername() {

return student.getStuname();

}

@Override

public boolean isAccountNonExpired() {

return true;

}

@Override

public boolean isAccountNonLocked() {

return true;

}

@Override

public boolean isCredentialsNonExpired() {

return true;

}

@Override

public boolean isEnabled() {

return true;

}

}

CustomStudentDeatilsService

**package** com.sboot.config;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.security.core.userdetails.UserDetails;

**import** org.springframework.security.core.userdetails.UserDetailsService;

**import** org.springframework.security.core.userdetails.UsernameNotFoundException;

**import** com.sboot.dao.StudentDao;

**import** com.sboot.model.Student;

**public** **class** CustomStudentDetailsService **implements** UserDetailsService {

@Autowired

**private** StudentDao studentDao;

@Override

**public** UserDetails loadUserByUsername(String username) **throws** UsernameNotFoundException {

Student student = **this**.studentDao.getStudentByStudentName(username);

**if**(student == **null**) {

**throw** **new** UsernameNotFoundException("Student is not found");

}

CustomStudentDetails customStudentDetails= **new** CustomStudentDetails(student);

**return** customStudentDetails;

}

}

SpringSecurityConfig

package com.sboot.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.authentication.dao.DaoAuthenticationProvider;

import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@Configuration

@EnableWebSecurity

public class SpringSecurityConfig extends WebSecurityConfigurerAdapter {

@Bean

public UserDetailsService getUserDetailsService() {

return new CustomStudentDetailsService();

}

@Bean

public BCryptPasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Bean

public DaoAuthenticationProvider authenticationProvider() {

DaoAuthenticationProvider daoAuthenticationProvider = new DaoAuthenticationProvider();

daoAuthenticationProvider.setUserDetailsService(this.getUserDetailsService());

daoAuthenticationProvider.setPasswordEncoder(passwordEncoder());

return daoAuthenticationProvider;

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.authenticationProvider(authenticationProvider());

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.authorizeRequests()

.antMatchers("/user/\*\*").hasRole("USER")

.antMatchers("/\*\*").permitAll().and().formLogin()

.loginPage("/signin")

.defaultSuccessUrl("/user/index")

.and().csrf().disable();

}

}

SpringSecurityConfig

package com.sboot.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.authentication.dao.DaoAuthenticationProvider;

import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@Configuration

@EnableWebSecurity

public class SpringSecurityConfig extends WebSecurityConfigurerAdapter {

@Bean

public UserDetailsService getUserDetailsService() {

return new CustomStudentDetailsService();

}

@Bean

public BCryptPasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Bean

public DaoAuthenticationProvider authenticationProvider() {

DaoAuthenticationProvider daoAuthenticationProvider = new DaoAuthenticationProvider();

daoAuthenticationProvider.setUserDetailsService(this.getUserDetailsService());

daoAuthenticationProvider.setPasswordEncoder(passwordEncoder());

return daoAuthenticationProvider;

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.authenticationProvider(authenticationProvider());

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.authorizeRequests()

.antMatchers("/user/\*\*").hasRole("USER")

.antMatchers("/\*\*").permitAll().and().formLogin()

.loginPage("/signin")

.defaultSuccessUrl("/user/index")

.and().csrf().disable();

}

}

Home Controller :

package com.sboot.controller;

import java.io.File;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.nio.file.StandardCopyOption;

import javax.servlet.http.HttpSession;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.core.io.ClassPathResource;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

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

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

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

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

import org.springframework.web.multipart.MultipartFile;

import com.sboot.dao.StudentDao;

import com.sboot.helper.Message;

import com.sboot.model.Student;

@Controller

public class HomeController {

@Autowired

private StudentDao studentDao;

@Autowired

private BCryptPasswordEncoder bCryptPasswordEncoder;

@RequestMapping("/signin")

public String signin(Model m) {

m.addAttribute("title", "Login");

return "signin";

}

@RequestMapping("/register")

public String signup(Model m) {

m.addAttribute("title", "Register");

m.addAttribute("student", new Student());

return "register";

}

// processing of register

@RequestMapping(value="/doregister", method = RequestMethod.POST)

public String processofregister(@Valid @ModelAttribute("student") Student student,BindingResult result,

@RequestParam("profileimg") MultipartFile file, Model m , HttpSession session) {

try {

if(result.hasErrors()) {

m.addAttribute("student",student);

return "register";

}

if(file.isEmpty()) {

System.out.println("file is empty");

student.setImgurl("profile.png");

}

else {

student.setImgurl(file.getOriginalFilename());

File savefile = new ClassPathResource("static/img").getFile();

Path path = Paths.get(savefile.getAbsolutePath()+File.separator+file.getOriginalFilename());

Files.copy(file.getInputStream(),path,StandardCopyOption.REPLACE\_EXISTING);

System.out.println("Image is uploaded ");

}

student.setRole("ROLE\_USER");

student.setPassword(bCryptPasswordEncoder.encode(student.getPassword()));

Student stu = this.studentDao.save(student);

m.addAttribute("student", new Student());

session.setAttribute("message", new Message("alert-success","Successfully register !! "));

} catch (Exception e) {

e.printStackTrace();

m.addAttribute("student",student);

session.setAttribute("message", new Message("alert-danger","Something went wrong !! " + e.getMessage()));

}

return "register";

}

}

Student Controller

**package** com.sboot.controller;

**import** java.security.Principal;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.Model;

**import** org.springframework.web.bind.annotation.ModelAttribute;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** com.sboot.dao.StudentDao;

**import** com.sboot.model.Student;

@Controller

@RequestMapping("/user")

**public** **class** StudentController {

@Autowired

**private** StudentDao studentDao;

@ModelAttribute

**public** **void** commondata(Model model , Principal principal) {

String stuname = principal.getName();

Student student = studentDao.getStudentByStuName(stuname);

model.addAttribute("student", student);

System.***out***.println(student);

}

@RequestMapping("/index")

**public** String studentindex(Model model , Principal principal) {

String stuname = principal.getName();

model.addAttribute("title","Dashboard");

**return** "index";

}

}

Message

**package** com.sboot.helper;

**public** **class** Message {

**private** String type;

**private** String content;

**public** String getType() {

**return** type;

}

**public** **void** setType(String type) {

**this**.type = type;

}

**public** String getContent() {

**return** content;

}

**public** **void** setContent(String content) {

**this**.content = content;

}

**public** Message(String type, String content) {

**super**();

**this**.type = type;

**this**.content = content;

}

**public** Message() {

**super**();

// **TODO** Auto-generated constructor stub

}

@Override

**public** String toString() {

**return** "Message [type=" + type + ", content=" + content + "]";

}

}

Index.html

<!doctype html>

<html lang=*"en"* xmlns:th=*"http://www.thymeleaf.org"* >

<head>

<meta charset=*"utf-8"*>

<meta content=*"width=device-width, initial-scale=1.0"* name=*"viewport"*>

<link href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"* rel=*"stylesheet"* integrity=*"sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"* crossorigin=*"anonymous"*>

<title th:text = *"${title}"*></title>

<meta content=*""* name=*"description"*>

<meta content=*""* name=*"keywords"*>

</head>

<body>

<div class=*"container"*>

<div class=*"row"*>

<div class=*"col-md-4 offset-md-4"*>

<div class=*"login-form bg-light mt-4 p-4 text-center"* >

<h2 th:text=*"'Welcome, ' + ${student.stuname}"*></h2>

<img th:src=*"@{'/img/'+${student.imgurl}}"* alt=*"profile photo"* class=*"my\_profile rounded-circle"* style="width:*200px*; height:*200px*;">

<div class=*"col-12 mt-4"*>

<a href=*"/logout"* class=*"btn btn-success "*>Logout</a>

</div>

<hr class=*"mt-4"*>

</div>

</div>

</div>

</div>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"* crossorigin=*"anonymous"*></script>

</body>

</html>

signin.html

<!doctype html>

<html lang=*"en"* xmlns:th=*"http://www.thymeleaf.org"* >

<head>

<meta charset=*"utf-8"*>

<meta content=*"width=device-width, initial-scale=1.0"* name=*"viewport"*>

<link href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"* rel=*"stylesheet"* integrity=*"sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"* crossorigin=*"anonymous"*>

<title th:text = *"${title}"*></title>

<meta content=*""* name=*"description"*>

<meta content=*""* name=*"keywords"*>

</head>

<body>

<div class=*"container"*>

<div class=*"row"*>

<div class=*"col-md-4 offset-md-4"*>

<div class=*"login-form bg-light mt-4 p-4"*>

<form th:action=*"@{/signin}"* method=*"post"* class=*"row g-3"*>

<div th:if=*"${param.error}"* class=*"alert alert-danger"*>

Invalid username and password</div>

<div th:if=*"${param.logout}"* class=*"alert alert-success"*>

You have been logout.</div>

<h4>Login Here</h4>

<div class=*"col-12"*>

<label>Username</label>

<input type=*"text"* name=*"username"* class=*"form-control"* placeholder=*"Username"*>

</div>

<div class=*"col-12"*>

<label>Password</label>

<input type=*"password"* name=*"password"* class=*"form-control"* placeholder=*"Password"*>

</div>

<div class=*"col-12"*>

<button type=*"submit"* class=*"btn btn-dark "*>Login</button>

</div>

</form>

<hr class=*"mt-4"*>

<div class=*"col-12"*>

<p class=*"text-center mb-0"*>Have not account yet? <a href=*"/register"*>Signup</a></p>

</div>

</div>

</div>

</div>

</div>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"* crossorigin=*"anonymous"*></script>

</body>

</html>

register.html

<!doctype html>

<html lang=*"en"* xmlns:th=*"http://www.thymeleaf.org"* >

<head>

<meta charset=*"utf-8"*>

<meta content=*"width=device-width, initial-scale=1.0"* name=*"viewport"*>

<link href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"* rel=*"stylesheet"* integrity=*"sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"* crossorigin=*"anonymous"*>

<title th:text = *"${title}"*></title>

<meta content=*""* name=*"description"*>

<meta content=*""* name=*"keywords"*>

</head>

<body>

<div class=*"container"*>

<div class=*"row"*>

<div class=*"col-md-4 offset-md-4"*>

<div class=*"login-form bg-light mt-4 p-4"*>

<div th:if=*"${session.message}"* th:classappend=*"${session.message.type}"* class=*"alert"* role=*"alert"*>

<p th:text=*"${session.message.content}"*></p>

<th:block th:text=*"${#session.removeAttribute('message')}"*></th:block>

</div>

<form novalidate=*"novalidate"* th:action=*"@{/doregister}"* method=*"post"* class=*"row g-3"*enctype=*"multipart/form-data"* th:object=*"${student}"*>

<h4>Register Here</h4>

<div class=*"col-12"*>

<label>Name</label>

<input type=*"text"* name=*"stuname"* class=*"form-control"*

th:classappend=*"${#fields.hasErrors('stuname') ? 'is-invalid' : '' }"*

id=*"name"* placeholder=*"Enter your student name"* required>

<div class=*"invalid-feedback"* th:each=*"e:${#fields.errors('stuname')}"* th:text=*"${e}"*>

</div>

</div>

<div class=*"col-12"*>

<label>Password</label>

<input type=*"password"* name=*"password"* class=*"form-control"*

th:classappend=*"${#fields.hasErrors('password') ? 'is-invalid' : '' }"*

id=*"password"* placeholder=*"Enter your student password"* required>

<div class=*"invalid-feedback"* th:each=*"e:${#fields.errors('password')}"* th:text=*"${e}"*>

</div>

</div>

<div class=*"col-12"*>

<label>Email</label>

<input type=*"email"* name=*"stuemail"* class=*"form-control"*

th:classappend=*"${#fields.hasErrors('stuemail') ? 'is-invalid' : '' }"*

id=*"stuemail"* placeholder=*"Enter your student email"* required>

<div class=*"invalid-feedback"* th:each=*"e:${#fields.errors('stuemail')}"* th:text=*"${e}"*>

</div>

</div>

<div class=*"col-12"*>

<label>Profile</label>

<input type=*"file"* name=*"profileimg"* class=*"form-control"*

id=*"imgurl"* placeholder=*"upload file"* required>

</div>

<div class=*"col-12"*>

<button type=*"submit"* class=*"btn btn-dark "*>Register</button>

</div>

</form>

<hr class=*"mt-4"*>

<div class=*"col-12"*>

<p class=*"text-center mb-0"*>Already Have account ? <a href=*"/signin"*>Login</a></p>

</div>

</div>

</div>

</div>

</div>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"* crossorigin=*"anonymous"*></script>

</body>

</html>

