**Code for User Login**

Maven Dependencies

<?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>2.3.0.RELEASE</version>

<relativePath />

<!-- lookup parent from repository -->

</parent>

<groupId>net.javaguides</groupId>

<artifactId>registration-login-spring-boot-security-thymeleaf</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>registration-login-spring-boot-security-thymeleaf</name>

<description>Demo project for Spring Boot Thymeleaf and Hibernate </description>

<properties>

<java.version>1.8</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.thymeleaf.extras</groupId>

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

</dependency>

<dependency>

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

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

<exclusions>

<exclusion>

<groupId>org.junit.vintage</groupId>

<artifactId>junit-vintage-engine</artifactId>

</exclusion>

</exclusions>

</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>

**MySQL Database Configuration**

Create a database with the name "demo" in the MySQL database server.

We’ll need to configure MySQL database URL, username, and password so that Spring can establish a connection with the database on startup.

Open application.properties and add following MySQL database configuration:

## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)

spring.datasource.url = jdbc:mysql://localhost:3306/demo?useSSL=false

spring.datasource.username = root

spring.datasource.password = root

## Hibernate Properties

# The SQL dialect makes Hibernate generate better SQL for the chosen database

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

# Hibernate ddl auto (create, create-drop, validate, update)

spring.jpa.hibernate.ddl-auto = update

logging.level.org.hibernate.SQL=DEBUG

logging.level.org.hibernate.type=TRACE

**Model Layer – Creation of JPA entities**

User

package net.javaguides.springboot.model;

import java.util.Collection;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinTable;

import javax.persistence.ManyToMany;

import javax.persistence.Table;

import javax.persistence.UniqueConstraint;

import javax.persistence.JoinColumn;

@Entity

@Table(name = "user", uniqueConstraints = @UniqueConstraint(columnNames = "email"))

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "first\_name")

private String firstName;

@Column(name = "last\_name")

private String lastName;

private String email;

private String password;

@ManyToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)

@JoinTable(

name = "users\_roles",

joinColumns = @JoinColumn(

name = "user\_id", referencedColumnName = "id"),

inverseJoinColumns = @JoinColumn(

name = "role\_id", referencedColumnName = "id"))

private Collection < Role > roles;

public User() {

}

public User(String firstName, String lastName, String email, String password, Collection < Role > roles) {

super();

this.firstName = firstName;

this.lastName = lastName;

this.email = email;

this.password = password;

this.roles = roles;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public Collection < Role > getRoles() {

return roles;

}

public void setRoles(Collection < Role > roles) {

this.roles = roles;

}

}

## **Role**

package net.javaguides.springboot.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "role")

public class Role {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

public Role() {

}

public Role(String name) {

super();

this.name = name;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

**Sprint Data JPA Repository**

package net.javaguides.springboot.repository;

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

import org.springframework.stereotype.Repository;

import net.javaguides.springboot.model.User;

@Repository

public interface UserRepository extends JpaRepository<User, Long>{

User findByEmail(String email);

}

**DTO – User Registration DTO**

package net.javaguides.springboot.web.dto;

public class UserRegistrationDto {

private String firstName;

private String lastName;

private String email;

private String password;

public UserRegistrationDto() {

}

public UserRegistrationDto(String firstName, String lastName, String email, String password) {

super();

this.firstName = firstName;

this.lastName = lastName;

this.email = email;

this.password = password;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

Service Layer

User Service

package net.javaguides.springboot.service;

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

import net.javaguides.springboot.model.User;

import net.javaguides.springboot.web.dto.UserRegistrationDto;

public interface UserService extends UserDetailsService{

User save(UserRegistrationDto registrationDto);

}

UserServiceImpl

package net.javaguides.springboot.service;

import java.util.Arrays;

import java.util.Collection;

import java.util.stream.Collectors;

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

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

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

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

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

import org.springframework.stereotype.Service;

import net.javaguides.springboot.model.Role;

import net.javaguides.springboot.model.User;

import net.javaguides.springboot.repository.UserRepository;

import net.javaguides.springboot.web.dto.UserRegistrationDto;

@Service

public class UserServiceImpl implements UserService {

private UserRepository userRepository;

@Autowired

private BCryptPasswordEncoder passwordEncoder;

public UserServiceImpl(UserRepository userRepository) {

super();

this.userRepository = userRepository;

}

@Override

public User save(UserRegistrationDto registrationDto) {

User user = new User(registrationDto.getFirstName(),

registrationDto.getLastName(), registrationDto.getEmail(),

passwordEncoder.encode(registrationDto.getPassword()), Arrays.asList(new Role("ROLE\_USER")));

return userRepository.save(user);

}

@Override

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

User user = userRepository.findByEmail(username);

if (user == null) {

throw new UsernameNotFoundException("Invalid username or password.");

}

return new org.springframework.security.core.userdetails.User(user.getEmail(), user.getPassword(), mapRolesToAuthorities(user.getRoles()));

}

private Collection << ? extends GrantedAuthority > mapRolesToAuthorities(Collection < Role > roles) {

return roles.stream().map(role - > new SimpleGrantedAuthority(role.getName())).collect(Collectors.toList());

}

}

**Spring Security Configuration**

package net.javaguides.springboot.config;

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

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.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.web.util.matcher.AntPathRequestMatcher;

import net.javaguides.springboot.service.UserService;

@Configuration

@EnableWebSecurity

public class SecurityConfiguration extends WebSecurityConfigurerAdapter {

@Autowired

private UserService userService;

@Bean

public BCryptPasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.authorizeRequests().antMatchers(

"/registration\*\*",

"/js/\*\*",

"/css/\*\*",

"/img/\*\*").permitAll()

.anyRequest().authenticated()

.and()

.formLogin()

.loginPage("/login")

.permitAll()

.and()

.logout()

.invalidateHttpSession(true)

.clearAuthentication(true)

.logoutRequestMatcher(new AntPathRequestMatcher("/logout"))

.logoutSuccessUrl("/login?logout")

.permitAll();

}

@Bean

public DaoAuthenticationProvider authenticationProvider() {

DaoAuthenticationProvider auth = new DaoAuthenticationProvider();

auth.setUserDetailsService(userService);

auth.setPasswordEncoder(passwordEncoder());

return auth;

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.authenticationProvider(authenticationProvider());

}

}

**User Registration Controller**

package net.javaguides.springboot.web;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

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

import net.javaguides.springboot.service.UserService;

import net.javaguides.springboot.web.dto.UserRegistrationDto;

@Controller

@RequestMapping("/registration")

public class UserRegistrationController {

private UserService userService;

public UserRegistrationController(UserService userService) {

super();

this.userService = userService;

}

@ModelAttribute("user")

public UserRegistrationDto userRegistrationDto() {

return new UserRegistrationDto();

}

@GetMapping

public String showRegistrationForm() {

return "registration";

}

@PostMapping

public String registerUserAccount(@ModelAttribute("user") UserRegistrationDto registrationDto) {

userService.save(registrationDto);

return "redirect:/registration?success";

}

}

**Main Controller**

package net.javaguides.springboot.web;

import org.springframework.stereotype.Controller;

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

@Controller

public class MainController {

@GetMapping("/login")

public String login() {

return "login";

}

@GetMapping("/")

public String home() {

return "index";

}

}

**Registration Page – registration.html**

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Registration</title>

<link rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"

integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"

crossorigin="anonymous">

</head>

<body>

<!-- create navigation bar ( header) -->

<nav class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle collapsed"

data-toggle="collapse" data-target="#navbar" aria-expanded="false"

aria-controls="navbar">

<span class="sr-only">Toggle navigation</span> <span

class="icon-bar"></span> <span class="icon-bar"></span> <span

class="icon-bar"></span>

</button>

<a class="navbar-brand" href="#" th:href="@{/}">Registration and

Login Module</a>

</div>

</div>

</nav>

<br>

<br>

<!-- Create HTML registration form -->

<div class="container">

<div class="row">

<div class="col-md-6 col-md-offset-3">

<!-- success message -->

<div th:if="${param.success}">

<div class="alert alert-info">You've successfully registered

to our awesome app!</div>

</div>

<h1>Registration</h1>

<form th:action="@{/registration}" method="post" th:object="${user}">

<div class="form-group">

<label class="control-label" for="firstName"> First Name </label>

<input id="firstName" class="form-control" th:field="\*{firstName}"

required autofocus="autofocus" />

</div>

<div class="form-group">

<label class="control-label" for="lastName"> Last Name </label> <input

id="lastName" class="form-control" th:field="\*{lastName}"

required autofocus="autofocus" />

</div>

<div class="form-group">

<label class="control-label" for="email"> Email </label> <input

id="email" class="form-control" th:field="\*{email}" required

autofocus="autofocus" />

</div>

<div class="form-group">

<label class="control-label" for="password"> Password </label> <input

id="password" class="form-control" type="password"

th:field="\*{password}" required autofocus="autofocus" />

</div>

<div class="form-group">

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

<span>Already registered? <a href="/" th:href="@{/login}">Login

here</a></span>

</div>

</form>

</div>

</div>

</div>

</body>

</html>

**Login Page – login.html**

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Registration and Login App</title>

<link rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"

integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"

crossorigin="anonymous">

</head>

<body>

<!-- create navigation bar ( header) -->

<nav class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle collapsed"

data-toggle="collapse" data-target="#navbar" aria-expanded="false"

aria-controls="navbar">

<span class="sr-only">Toggle navigation</span> <span

class="icon-bar"></span> <span class="icon-bar"></span> <span

class="icon-bar"></span>

</button>

<a class="navbar-brand" href="#" th:href="@{/}">Registration and

Login Module</a>

</div>

</div>

</nav>

<br>

<br>

<div class = "container">

<div class = "row">

<div class = "col-md-6 col-md-offset-3">

<h1> User Login Page </h1>

<form th:action="@{/login}" method="post">

<!-- error message -->

<div th:if="${param.error}">

<div class="alert alert-danger">Invalid username or

password.</div>

</div>

<!-- logout message -->

<div th:if="${param.logout}">

<div class="alert alert-info">You have been logged out.</div>

</div>

<div class = "form-group">

<label for ="username"> Username </label> :

<input type="text" class = "form-control" id ="username" name = "username"

placeholder="Enter Email ID" autofocus="autofocus">

</div>

<div class="form-group">

<label for="password">Password</label>: <input type="password"

id="password" name="password" class="form-control"

placeholder="Enter Password" />

</div>

<div class="form-group">

<div class="row">

<div class="col-sm-6 col-sm-offset-3">

<input type="submit" name="login-submit" id="login-submit"

class="form-control btn btn-primary" value="Log In" />

</div>

</div>

</div>

</form>

<div class="form-group">

<span>New user? <a href="/" th:href="@{/registration}">Register

here</a></span>

</div>

</div>

</div>

</div>

</body>

</html>

**Home Page – index.html**

<!DOCTYPE html>

<html xmlns:th="http://www.thymeleaf.org"

xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity3">

<head>

<meta charset="ISO-8859-1">

<title>Registration and Login App</title>

<link rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"

integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"

crossorigin="anonymous">

</head>

<body>

<!-- create navigation bar ( header) -->

<nav class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle collapsed"

data-toggle="collapse" data-target="#navbar" aria-expanded="false"

aria-controls="navbar">

<span class="sr-only">Toggle navigation</span> <span

class="icon-bar"></span> <span class="icon-bar"></span> <span

class="icon-bar"></span>

</button>

<a class="navbar-brand" href="#" th:href="@{/}">Registration and

Login Module</a>

</div>

<div id="navbar" class="collapse navbar-collapse">

<ul class="nav navbar-nav">

<li sec:authorize="isAuthenticated()"><a th:href="@{/logout}">Logout</a></li>

</ul>

</div>

</div>

</nav>

<br>

<br>

<div class="container">

<h1>Registration and Login with Spring Boot, Spring Security,

Thymeleaf, Hibernate and MySQL</h1>

Welcome <span sec:authentication="principal.username"> User</span>

</div>

</body>

</html>

**Code for Student Database**

**Maven Dependencies – Code for 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>3.0.2</version>

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

</parent>

<groupId>com.example</groupId>

<artifactId>CrudDemoApplication</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>CrudDemoApplication</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>17</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-thymeleaf</artifactId>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

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

<artifactId>spring-boot-starter-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>

**Create JPA Entity - Student.java**

**package** com.example.CrudDemoApplication.model;

**import** jakarta.persistence.Column;

**import** jakarta.persistence.Entity;

**import** jakarta.persistence.GeneratedValue;

**import** jakarta.persistence.GenerationType;

**import** jakarta.persistence.Id;

**import** jakarta.persistence.Table;

@Entity

@Table(name="students")

**public** **class** Student {

@Id

@GeneratedValue(strategy=GenerationType.***IDENTITY***)

**private** **long** id;

@Column(name= "first\_name")

**private** String firstName;

@Column(name="last\_name")

**private** String lastName;

**private** String email;

**public** **long** getId() {

**return** id;

}

**public** **void** setId(**long** id) {

**this**.id = id;

}

**public** String getFirstName() {

**return** firstName;

}

**public** **void** setFirstName(String firstName) {

**this**.firstName = firstName;

}

**public** String getLastName() {

**return** lastName;

}

**public** **void** setLastName(String lastName) {

**this**.lastName = lastName;

}

**public** String getcourse() {

**return** course;

}

**public** **void** setcourse(String course) {

**this**.course = course;

}

**public** String getcountry() {

**return** country;

}

**public** **void** setcountry(String country) {

**this**.country = country;

}

}

**Backend Changes –**

**StudentRepository.java**

**package** com.example.CrudDemoApplication.repository;

**import** org.springframework.stereotype.Repository;

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

**import** com.example.CrudDemoApplication.model.Employee;

@Repository

**public** **interface** StudentRepository **extends** JpaRepository<Student,Long>{

}

**StudentService.java**

**package** com.example.CrudDemoApplication.service;

**import** java.util.List;

**import** com.example.CrudDemoApplication.model.Employee;

**public** **interface** StudentService {

List<Student>getAllStudents();

**void** saveStudent(Student student);

Employee getStudentById(**long** id);

**void** deleteStudentById(**long** id);

}

**StudentServiceImpl.java**

**package** com.example.CrudDemoApplication.service;

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

**import** org.springframework.stereotype.Service;

**import** com.example.CrudDemoApplication.model.Employee;

**import** com.example.CrudDemoApplication.repository.EmployeeRepository;

**import** java.util.List;

**import** java.util.Optional;

@Service

**public** **class** StudentServiceImpl **implements** StudentService {

@Autowired

**private** StudentRepository studentRepository;

@Override

**public** List<Student> getAllStudents(){

**return** studentRepository.findAll();

}

@Override

**public** **void** savestudent(Student student) {

**this**.studentRepository.save(student);

}

**public** Employee getStudentById(**long** id) {

Optional < Employee > optional = studentRepository.findById(id);

Student student = **null**;

**if** (optional.isPresent()) {

student = optional.get();

} **else** {

**throw** **new** RuntimeException(" Student not found for id :: " + id);

}

**return** employee;

}

@Override

**public** **void** deleteStudentById(**long** id) {

**this**.studentRepository.deleteById(id);

}

}

**StudentController.java**

**package** com.example.CrudDemoApplication.controller;

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

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

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

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

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

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

**import** com.example.CrudDemoApplication.model.Employee;

**import** com.example.CrudDemoApplication.service.EmployeeService;

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

@Controller

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

@Autowired

**private** StudentService studentService;

@GetMapping("/")

**public** String viewHomePage(Model model) {

model.addAttribute("listStudents" ,studentService.getAllStudents());

**return** "index";

}

@GetMapping("/showNewStudentForm")

**public** String showNewStudentForm(Model model) {

Student student = **new** Student();

model.addAttribute("student", student);

**return** "new\_student";

}

@PostMapping("/saveStudent")

**public** String saveStudent(@ModelAttribute("student")Student student) {

studentService.saveStudent(student);

**return** "redirect:/";

}

@GetMapping("/showFormForUpdate/{id}")

**public** String showFormForUpdate(@PathVariable(value = "id") **long** id, Model model) {

// get student from the service

Student student = studentService.getStudentById(id);

// set employee as a model attribute to pre-populate the form

model.addAttribute("student", student);

**return** "update\_student";

}

@GetMapping("/deleteStudent/{id}")

**public** String deleteStudent(@PathVariable(value = "id") **long** id) {

// call delete student method

**this**.studentService.deleteStudentById(id);

**return** "redirect:/";

}

}

**Front End Part**

<!DOCTYPE html>

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

<head>

<meta charset=*"ISO-8859-1"*>

<title>Student Management System</title>

<link rel=*"stylesheet"* href=*"https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css"*

integrity=*"sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"*

crossorigin=*"anonymous"*>

</head>

<body>

<div class=*"container my-2"*>

<h1> Students List</h1>

<a th:href = *"@{/showNewStudentForm}"* class=*"btn btn-primary btn-sm mb-3"*> Add Student</a>

<br> <br>

<table border=*"1"* class=*"table table-striped table-responsive-md"*>

<thead>

<tr>

<th> First Name </th>

<br>

<th> Last Name </th>

<br>

<th> Course </th>

<br>

<th>Country </th>

<br>

<th>Actions</th>

<br>

</tr>

</thead>

<tbody>

<tr th:each=*"student : ${listStudents}"*>

<td th:text= *"${student.firstName}"*></td>

<td th:text= *"${student.lastName}"*></td>

<td th:text= *"${student.course}"*></td>

<td th:text= *"${student.country}"*></td>

<td><a th:href=*"@{/showFormForUpdate/{id}(id=${student.id})}"* class=*"btn btn-primary"*>Update</a>

<a th:href=*"@{/deleteStudent/{id}(id=${student.id})}"* class=*"btn btn-danger"*>Delete</a>

</td>

</tr>

</tbody>

</table>

</div>

</body>

</html>

**Create new\_student.html**

<!DOCTYPE html>

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

<head>

<meta charset=*"ISO-8859-1"*>

<title>Student Management System</title>

<link rel=*"stylesheet"* href=*"https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css"*

integrity=*"sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"*

crossorigin=*"anonymous"*>

</head>

<body>

<div class=*"container"*>

<h1> Student Management System</h1>

<hr>

<h2> Save Student</h2>

<form action= *"#"* th:action=*"@{/saveStudent}"* th:object=*"${student}"* method=*"POST"*>

<input type =*"text"* th:field = *"\*{firstName}"*

placeholder= *"Student First Name"* class=*"form-control mb-4 col-4"*>

<input type =*"text"* th:field = *"\*{lastName}"*

placeholder= *"Employee Last Name"* class=*"form-control mb-4 col-4"*>

<input type =*"text"* th:field = *"\*{course}"*

placeholder= *"Student Course"* class=*"form-control mb-4 col-4"*>

<input type =*"text"* th:field = *"\*{country}"*

placeholder= *"Student Country"* class=*"form-control mb-4 col-4"*>

<button type=*"submit"* class=*"btn btn-info col-2"*>Save Student</button>

</form>

</div>

</body>

</html>

**Create update\_student.html**

<!DOCTYPE html>

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

<head>

<meta charset=*"ISO-8859-1"*>

<title>Student Management System</title>

<link rel=*"stylesheet"* href=*"https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css"*

integrity=*"sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"*

crossorigin=*"anonymous"*>

</head>

<body>

<div class=*"container"*>

<h1> Student Management System</h1>

<hr>

<h2> Update Student</h2>

<form action= *"#"* th:action=*"@{/saveStudent}"* th:object=*"${student}"* method=*"POST"*>

<input type=*"hidden"* th:field=*"\*{id}"* />

<input type=*"text"* th:field=*"\*{firstName}"* class=*"form-control mb-4 col-4"*>

<input type=*"text"* th:field=*"\*{lastName}"* class=*"form-control mb-4 col-4"*>

<input type=*"text"* th:field=*"\*{course}"* class=*"form-control mb-4 col-4"*>

<input type=*"text"* th:field=*"\*{country}"* class=*"form-control mb-4 col-4"*>

<button type=*"submit"* class=*"btn btn-info col-2"*> Update Student</button>

</form>

<hr>

<a th:href=*"@{/}"*> Back to Student List</a>

</div>

</body>

</html>