Q.1

HelloController.java

**package** com.springsecurity.resource;

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

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

@RestController

**public** **class** HelloResource {

@GetMapping("/admin")

**public** String hello() {

**return** "Hello World!!";

}

}

HelloConfig.java

**package** com.springsecurity.config;

**import** org.springframework.context.annotation.Bean;

**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.password.~~NoOpPasswordEncoder~~;

**import** org.springframework.security.crypto.password.PasswordEncoder;

@EnableWebSecurity

**public** **class** HelloConfig **extends** WebSecurityConfigurerAdapter {

@Override

**protected** **void** configure(AuthenticationManagerBuilder auth) **throws** Exception {

auth.inMemoryAuthentication()

.withUser("admin")

.password("123")

.roles("ADMIN");

}

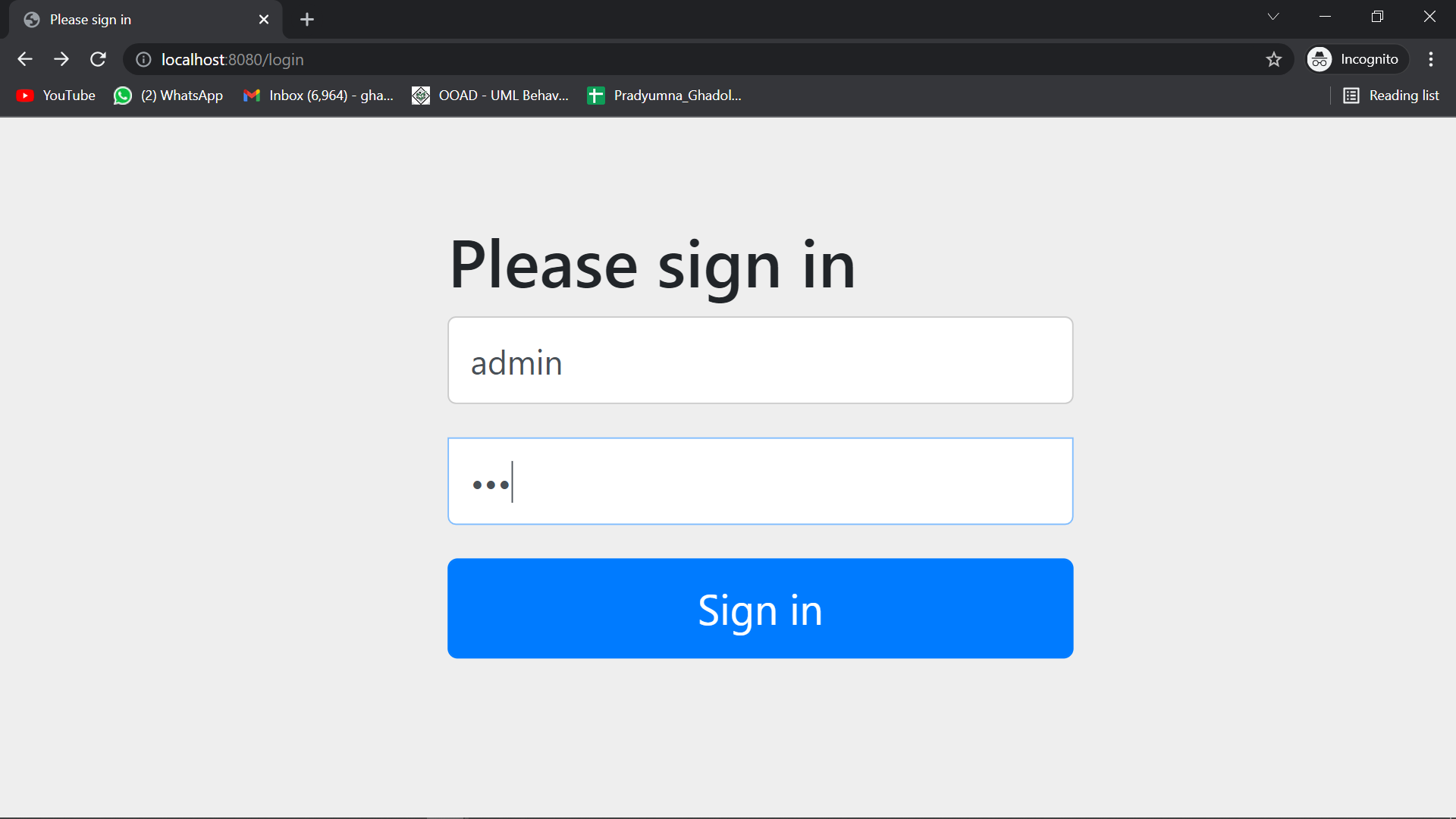
@Bean

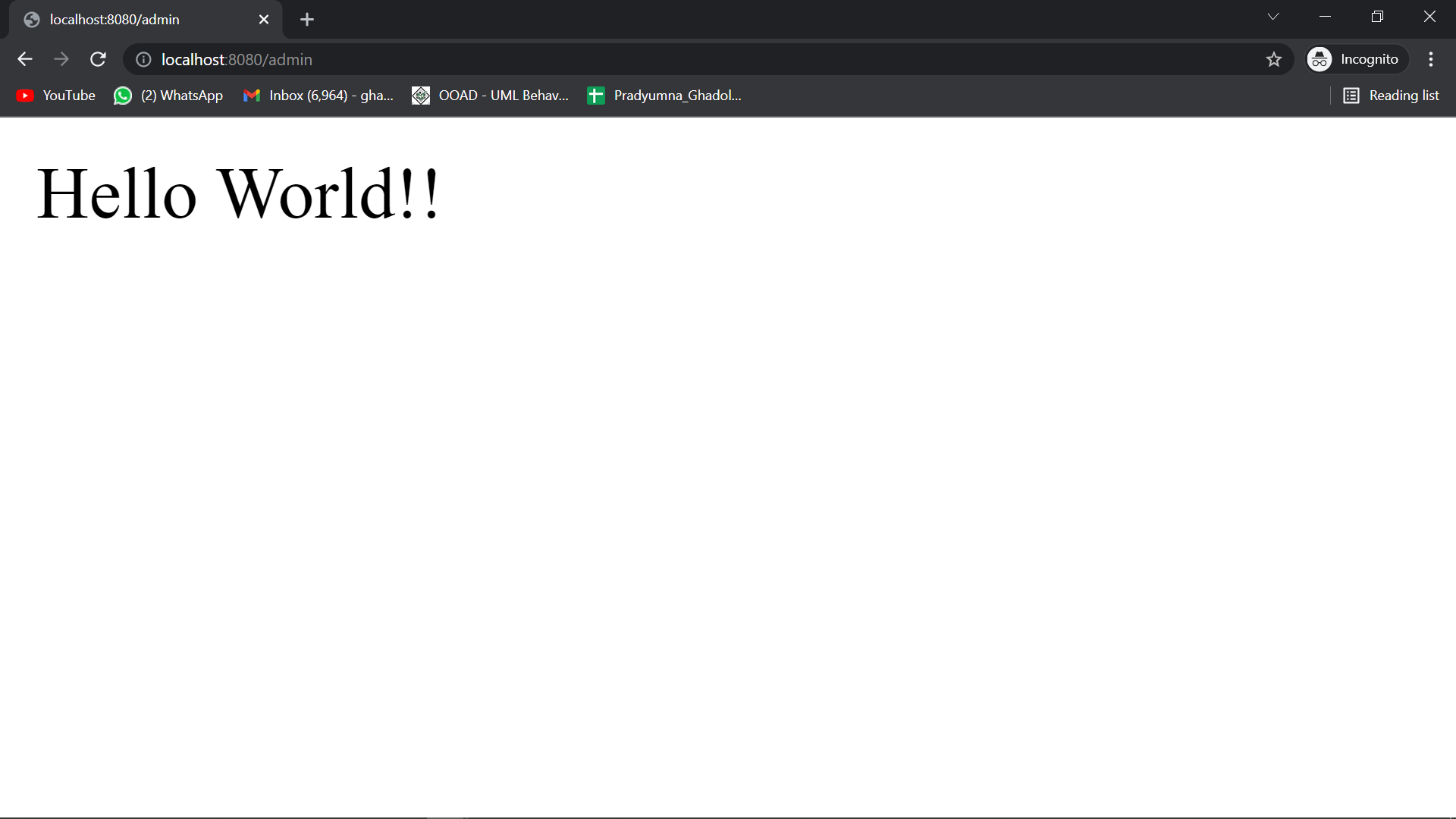
PasswordEncoder encoder() {

**return** ~~NoOpPasswordEncoder~~.~~getInstance~~();

}

}





Q.2

HelloResource.java

package com.springsecurity.resource;

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

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

@RestController

public class HelloResource {

@GetMapping("/admin")

public String hello() {

return "Hello World!!";

}

}

HelloConfig.java

package com.springsecurity.config;

import org.springframework.context.annotation.Bean;

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.password.NoOpPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

@EnableWebSecurity

public class HelloConfig extends WebSecurityConfigurerAdapter {

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.inMemoryAuthentication()

.withUser("admin")

.password("123")

.roles("ADMIN");

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.authorizeRequests()

.anyRequest().authenticated()

.and()

.formLogin()

.loginPage("/login")

.permitAll();

}

@Bean

PasswordEncoder encoder() {

return NoOpPasswordEncoder.getInstance();

}

}

LoginController.java

**package** com.springsecurity.resource;

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

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

@Controller

**public** **class** LoginController {

@GetMapping("/login")

**public** String login() {

**return** "login";

}

}

Login.html

<html xmlns:th=*"https://www.thymeleaf.org"*>

<head>

<title>Please Login</title>

</head>

<body>

<div>

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

<fieldset>

<legend>Please login</legend>

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

username and password</div>

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

have been logged out!</div>

<label for=*"username"*>Username</label> <input type=*"text"*

id=*"username"* name=*"username"*> <label for=*"password"*>Password</label>

<input type=*"password"* id=*"password"* name=*"password"*>

<div class=*"form-actions"*>

<button type=*"submit"* class=*"btn"*>Log in</button>

</div>

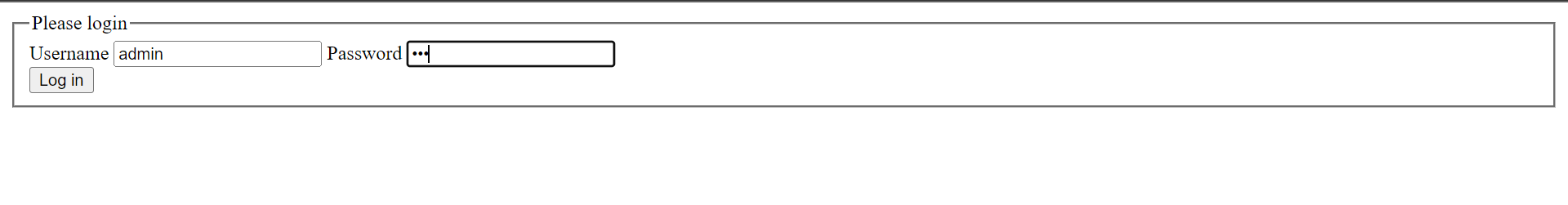
</fieldset>

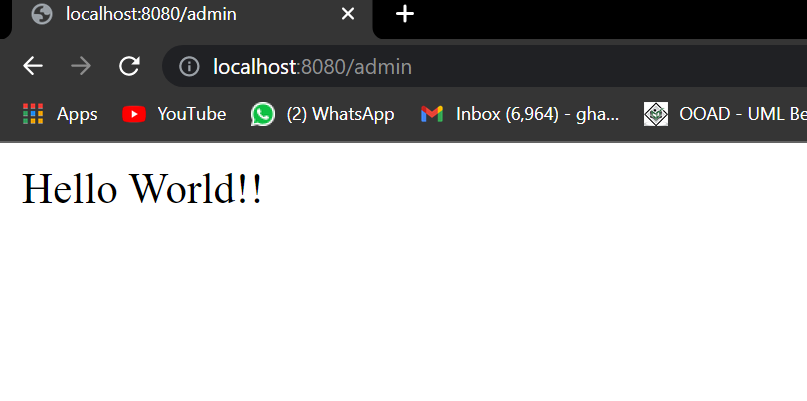
</form>

</div>

</body>

</html>





Q3.

HelloResource.java

package com.springsecurity.resource;

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

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

@RestController

public class HelloResource {

@GetMapping("/admin")

public String hello() {

return "Hello World!!";

}

@GetMapping("/user")

public String user() {

return "You cannot access helloworld";

}

}

HelloConfig.java

**package** com.springsecurity.config;

**import** javax.sql.DataSource;

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

**import** org.springframework.context.annotation.Bean;

**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.password.~~NoOpPasswordEncoder~~;

**import** org.springframework.security.crypto.password.PasswordEncoder;

@EnableWebSecurity

**public** **class** HelloConfig **extends** WebSecurityConfigurerAdapter {

@Autowired

DataSource dataSource;

@Override

**protected** **void** configure(AuthenticationManagerBuilder auth) **throws** Exception {

auth.jdbcAuthentication()

.dataSource(dataSource);

}

@Override

**protected** **void** configure(HttpSecurity http) **throws** Exception {

http.authorizeRequests()

.antMatchers("/admin").hasRole("ADMIN")

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

.anyRequest().authenticated()

.and()

.formLogin()

.loginPage("/login")

.permitAll();

}

@Bean

PasswordEncoder encoder() {

**return** ~~NoOpPasswordEncoder~~.~~getInstance~~();

}

}

LoginController.java

package com.springsecurity.resource;

import org.springframework.stereotype.Controller;

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

@Controller

public class LoginController {

@GetMapping("/login")

public String login() {

return "login";

}

}

Schema.sql

create table users(

username varchar\_ignorecase(50) not null primary key,

password varchar\_ignorecase(50) not null,

enabled boolean not null);

create table authorities (

username varchar\_ignorecase(50) not null,

authority varchar\_ignorecase(50) not null,

constraint fk\_authorities\_users foreign key(username) references users(username));

create unique index ix\_auth\_username on authorities (username,authority);

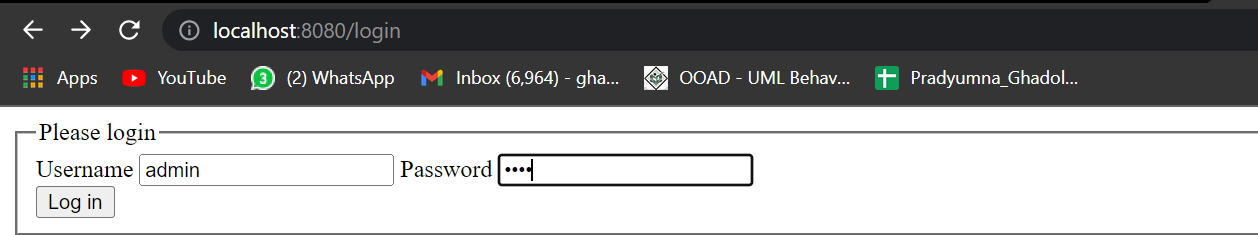
data.sql

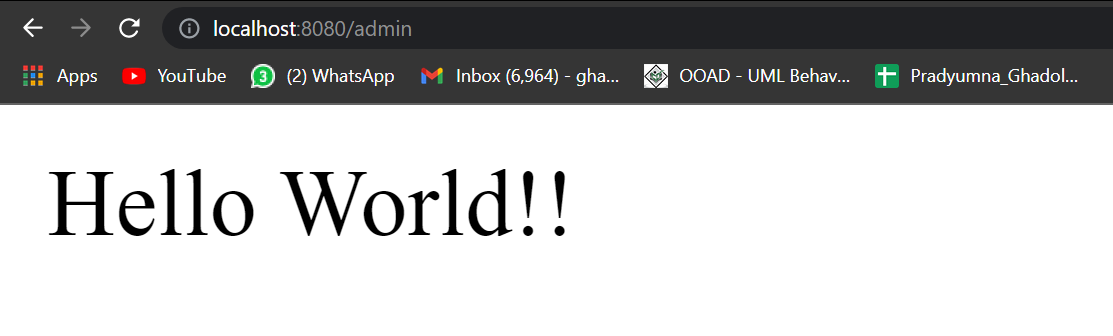
Insert into users(username, password, enabled) values('user','pass','true');

Insert into users(username, password, enabled) values('admin','pass','true');

Insert into authorities(username, authority) values('user','ROLE\_USER');

Insert into authorities(username, authority) values('admin','ROLE\_ADMIN');





Q5.

Helloconfig.java

package com.springsecurity.config;

import javax.sql.DataSource;

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

import org.springframework.context.annotation.Bean;

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.password.NoOpPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

@EnableWebSecurity

public class HelloConfig extends WebSecurityConfigurerAdapter {

@Autowired

DataSource dataSource;

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.jdbcAuthentication()

.dataSource(dataSource);

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.authorizeRequests()

.antMatchers("/admin").hasRole("ADMIN")

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

.anyRequest().authenticated()

.and()

.formLogin()

.loginPage("/login")

.permitAll()

.and()

.rememberMe().key("PRADRISHGHAD123456789")

.tokenValiditySeconds(3 \* 20 \* 60 \* 60);

}

@Bean

PasswordEncoder encoder() {

return NoOpPasswordEncoder.getInstance();

}

}

helloResource.java

package com.springsecurity.resource;

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

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

@RestController

public class HelloResource {

@GetMapping("/admin")

public String hello() {

return "Hello World!!";

}

@GetMapping("/user")

public String user() {

return "You cannot access helloworld";

}

}

Logincontroller.java

package com.springsecurity.resource;

import org.springframework.stereotype.Controller;

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

@Controller

public class LoginController {

@GetMapping("/login")

public String login() {

return "login";

}

}

Login.html

<html xmlns:th=*"https://www.thymeleaf.org"*>

<head>

<title>Please Login</title>

</head>

<body>

<div>

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

<fieldset>

<legend>Please login</legend>

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

username and password

</div>

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

have been logged out!

</div>

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

<input type=*"text"* id=*"username"* name=*"username"*>

<label for=*"password"*>Password</label>

<input type=*"password"* id=*"password"* name=*"password"*>

<p>

<input type = *"checkbox"* name= *"remember-me"* /> Remember me

</p>

<div class=*"form-actions"*>

<button type=*"submit"* class=*"btn"*>Log in</button>

</div>

</fieldset>

</form>

</div>

</body>

</html>

Application.properties

spring.h2.console.enabled=true

spring.sql.init.platform=h2

spring.datasource.url=jdbc:h2:mem:pradumnya

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>2.6.3</version>

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

</parent>

<groupId>com.springboot</groupId>

<artifactId>Assignment-SpringSecurity-Q1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Assignment-SpringSecurity-Q1</name>

<description>spring security jwt</description>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</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>

<dependency>

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

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

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

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

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

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

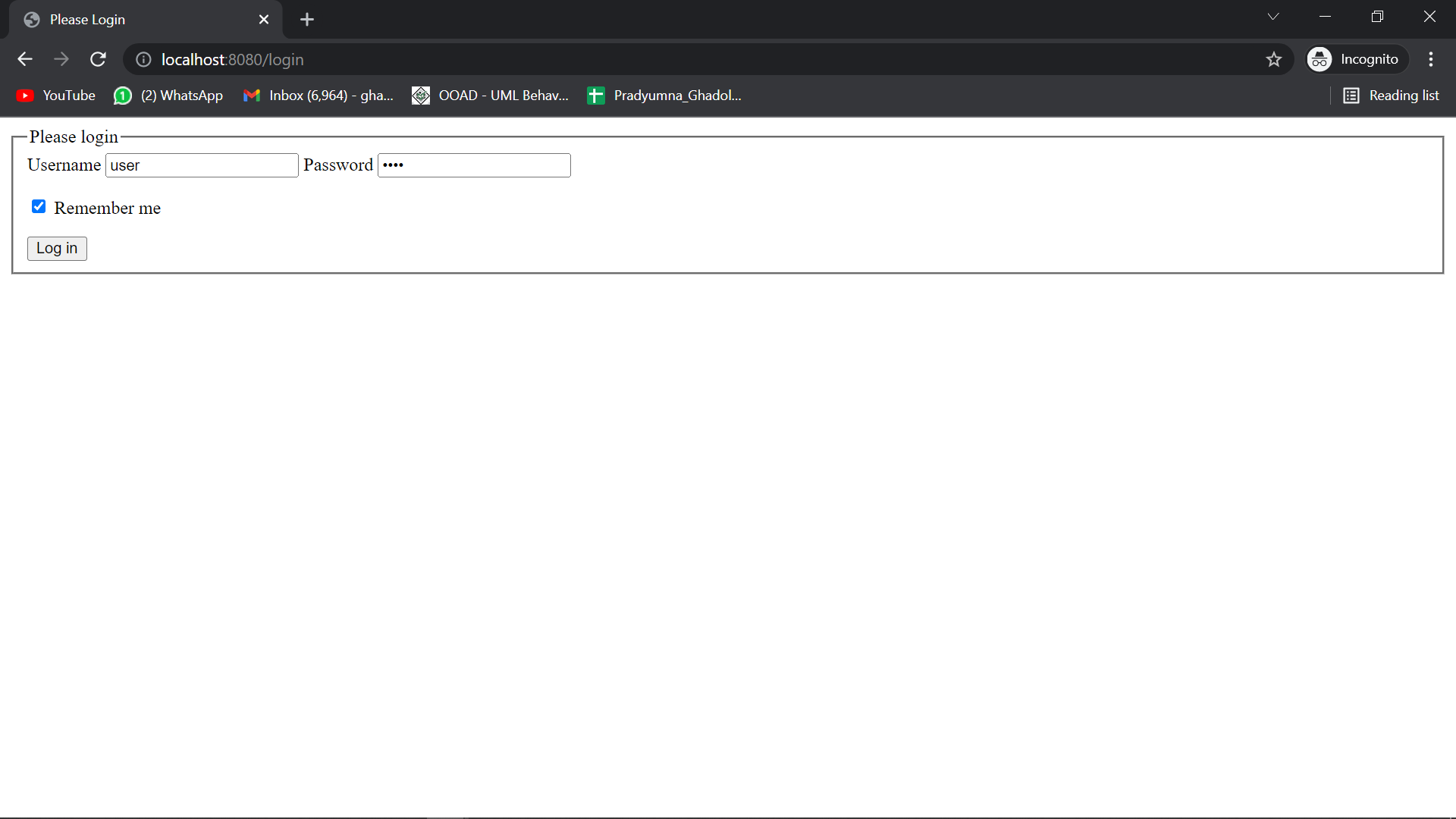
<artifactId>spring-boot-maven-plugin</artifactId>

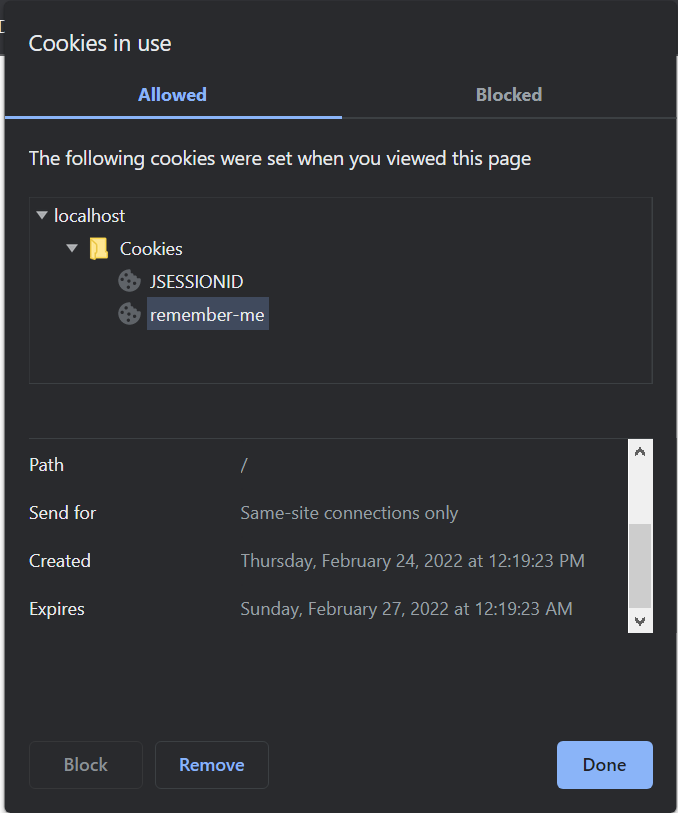
</plugin>

</plugins>

</build>

</project>





Q6.

User.java

package com.springsecurity.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "User")

public class User {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private int id;

private String userName;

private String password;

private Boolean active;

private String roles;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public Boolean getActive() {

return active;

}

public void setActive(Boolean active) {

this.active = active;

}

public String getRoles() {

return roles;

}

public void setRoles(String roles) {

this.roles = roles;

}

}

userRepository.java

package com.springsecurity.repo;

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

import com.springsecurity.model.User;

public interface UserRepository extends JpaRepository<User, Integer> {

public User findByUserName(String userName);

}

UserConfig.java

**package** com.springsecurity.config;

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

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**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** UserConfig **extends** WebSecurityConfigurerAdapter {

@Autowired

UserDetailsService userDetailsService;

@Override

**protected** **void** configure(AuthenticationManagerBuilder auth) **throws** Exception {

auth.userDetailsService(userDetailsService).passwordEncoder(passwordencoder());

}

@Override

**protected** **void** configure(HttpSecurity http) **throws** Exception {

http.authorizeRequests()

.antMatchers("/admin").hasRole("ADMIN")

.antMatchers("/user").hasAnyRole("USER","ADMIN")

.antMatchers("/").permitAll()

.and().formLogin();

}

@Bean

**public** BCryptPasswordEncoder passwordencoder() {

**return** **new** BCryptPasswordEncoder(10);

}

}

userResource.java

package com.springsecurity.controller;

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

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

@RestController

public class UserResource {

@GetMapping("/")

public String home() {

return "Hello";

}

@GetMapping("/admin")

public String admin() {

return "Hello Admin";

}

@GetMapping("/user")

public String user() {

return "Hello User";

}

}

CustomUserDetails.java

package com.springsecurity.model;

import java.util.Collection;

import java.util.HashSet;

import org.springframework.security.core.GrantedAuthority;

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

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

/\*\*

\*

\* this is used to provide user details

\*

\*/

public class CustomUserDetail implements UserDetails {

private static final long serialVersionUID = 1L;

private User user;

public CustomUserDetail(User user) {

this.user = user;

}

@Override

public Collection<? extends GrantedAuthority> getAuthorities() {

HashSet<SimpleGrantedAuthority> set = new HashSet<>();

set.add(new SimpleGrantedAuthority(this.user.getRoles()));

return set;

}

@Override

public String getPassword() {

return this.user.getPassword();

}

@Override

public String getUsername() {

return this.user.getUserName();

}

@Override

public boolean isAccountNonExpired() {

return true;

}

@Override

public boolean isAccountNonLocked() {

return true;

}

@Override

public boolean isCredentialsNonExpired() {

return true;

}

@Override

public boolean isEnabled() {

return true;

}

}

CustomerUserDetailService.java

package com.springsecurity.service;

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 org.springframework.stereotype.Service;

import com.springsecurity.model.CustomUserDetail;

import com.springsecurity.model.User;

import com.springsecurity.repo.UserRepository;

@Service

public class CustomUserDetailService implements UserDetailsService {

@Autowired

private UserRepository userRepository;

@Override

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

User user = this.userRepository.findByUserName(username);

if (user == null) {

throw new UsernameNotFoundException("NO User");

}

return new CustomUserDetail(user);

}

}

AssignmentSpringSecurityQ6Application.java

package com.springsecurity;

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

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

import com.springsecurity.model.User;

import com.springsecurity.repo.UserRepository;

@SpringBootApplication

public class AssignmentSpringSecurityQ6Application implements CommandLineRunner{

@Autowired

private UserRepository userRepository;

@Autowired

private BCryptPasswordEncoder bCryptPasswordEncoder;

public static void main(String[] args) {

SpringApplication.run(AssignmentSpringSecurityQ6Application.class, args);

}

@Override

public void run(String... args) throws Exception {

User user = new User();

user.setUserName("Pradumnya");

user.setPassword(this.bCryptPasswordEncoder.encode("123"));

user.setRoles("ROLE\_NORMAL");

userRepository.save(user);

User user2 = new User();

user2.setUserName("Rishikesh");

user2.setPassword(this.bCryptPasswordEncoder.encode("123"));

user2.setRoles("ROLE\_ADMIN");

userRepository.save(user2);

}

}

