**Spring Security Assignment2**

1)

package com.example.security11;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

    public static void main(String[] args) {

        SpringApplication.run(Application.class, args);

    }

}

//

package com.example.security11.config;

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.crypto.bcrypt.BCryptPasswordEncoder;

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

@Configuration

@EnableWebSecurity

public class MySecurityConfig extends WebSecurityConfigurerAdapter {

    @Override

    protected void configure(HttpSecurity http) throws Exception {

        http

                .authorizeRequests()

                .antMatchers("/public/\*\*").hasRole("ADMIN")

                .antMatchers("/signin").permitAll()

                .antMatchers("/users/\*\*").hasRole("NORMAL")

                .anyRequest()

                .authenticated()

                .and()

                .formLogin()

                .loginPage("/signin")

                .loginProcessingUrl("/dologin")

                .defaultSuccessUrl("/users/");

    }

    @Override

    protected void configure(AuthenticationManagerBuilder auth) throws Exception {

        auth

                .inMemoryAuthentication()

                .withUser("Himanshu")

                .password(this.passwordEncoder().encode("Himan12"))

                .roles("ADMIN");

        auth

                .inMemoryAuthentication()

                .withUser("tom")

                .password(this.passwordEncoder().encode("rider"))

                .roles("NORMAL");

    }

    @Bean

    public PasswordEncoder passwordEncoder() {

        return new BCryptPasswordEncoder(10);

    }

}

//

package com.example.security11.controller;

import com.example.security11.models.User;

import com.example.security11.services.serviceImpl;

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

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

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

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

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

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

@RestController

public class controller {

    @Autowired

    private serviceImpl Service;

    @GetMapping("/users")

    public String getAllUsers() {

        return "Hello User, Welcome to our website";

    }

    @PostMapping("/users")

    public String setUser(@RequestBody User user) {

        this.Service.addUser(user);

        return "User added successfully";

    }

    @GetMapping("/users/{username}")

    public User getUserByUserName(@PathVariable("username") String username) {

        return this.Service.getUser(username);

    }

}

//

package com.example.security11.controller;

import org.springframework.stereotype.Controller;

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

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

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

@Controller

@RequestMapping("/public")

public class HomeController {

    @GetMapping("/home")

    @ResponseBody

    public String home() {

        return "Welcome Admin, This is our home page";

    }

    @GetMapping("/login")

    @ResponseBody

    public String login() {

        return "Welcome Admin, This is our login page";

    }

    @GetMapping("/register")

    @ResponseBody

    public String register() {

        return "Welcome Admin, This is our register page";

    }

}

//

package com.example.security11.controller;

import org.springframework.stereotype.Controller;

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

@Controller

public class TestController {

    @GetMapping("/signin")

    public String signin(){

        return "login";

    }

}

//

package com.example.security11.models;

public class User {

    private String username;

    private String password;

    private String email;

    public User() {

        super();

    }

    public User(String username, String password, String email) {

        this.username = username;

        this.password = password;

        this.email = email;

    }

    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 String getEmail() {

        return email;

    }

    public void setEmail(String email) {

        this.email = email;

    }

    @Override

    public String toString() {

        return "User [email=" + email + ", password=" + password + ", username=" + username + "]";

    }

}

//

package com.example.security11.services;

import java.util.List;

import com.example.security11.models.User;

public interface service {

    public List<User> getAllUsers();

    public User getUser(String username);

    public User addUser(User user);

}

//

package com.example.security11.services;

import java.util.ArrayList;

import java.util.List;

import com.example.security11.models.User;

import org.springframework.stereotype.Service;

@Service

public class serviceImpl implements service {

    List<User> list = new ArrayList<>();

    public serviceImpl() {

        list.add(new User("Himanshu", "Himan12", "himanshuwakade@gmail.com"));

    }

    // Get All Users

    public List<User> getAllUsers() {

        return this.list;

    }

    // Get Users By Username

    public User getUser(String username) {

        return this.list.stream().filter((user) -> user.getUsername().equals(username)).findAny().orElse(null);

    }

    // To set users

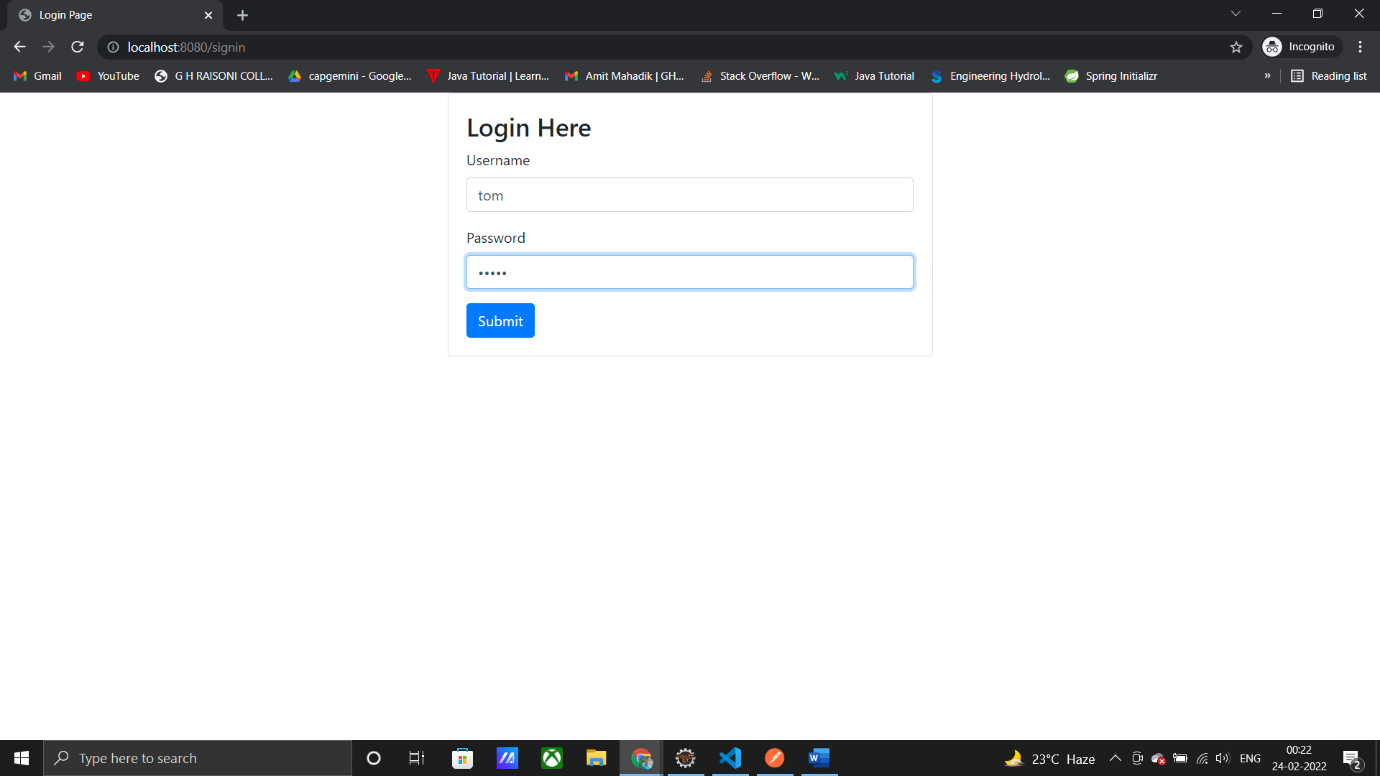
    public User addUser(User user) {

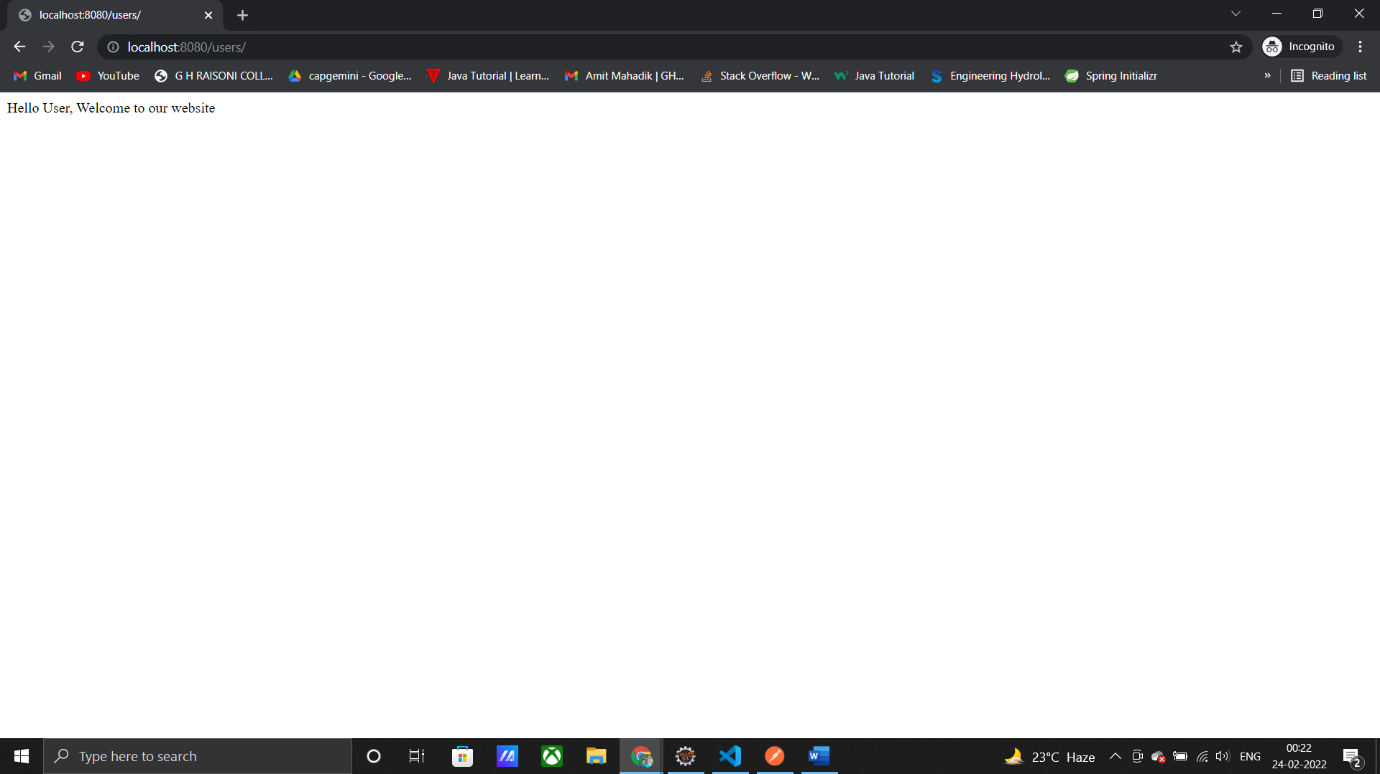
        this.list.add(user);

        return user;

    }

}





2)

package com.example.assignment2;

import com.example.assignment2.model.User;

import com.example.assignment2.repository.UserRepo;

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;

@SpringBootApplication

public class Application implements CommandLineRunner {

    @Autowired

    private UserRepo userRepo;

    @Autowired

    private BCryptPasswordEncoder bCryptPasswordEncoder;

    public static void main(String[] args) {

        SpringApplication.run(Application.class, args);

    }

    @Override

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

        User user1 = new User();

        user1.setEmail("himanshuwakade@gmail.com");

        user1.setUsername("Himanshu");

        user1.setPassword(bCryptPasswordEncoder.encode("himan07"));

        user1.setRole("ROLE\_ADMIN");

        userRepo.save(user1);

        User user2 = new User();

        user2.setEmail("rdj@gmail.com");

        user2.setUsername("Tony");

        user2.setPassword(bCryptPasswordEncoder.encode("ironman"));

        user2.setRole("ROLE\_NORMAL");

        userRepo.save(user2);

    }

}

//

package com.example.assignment2.config;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.example.assignment2.service.CustomUserDetailService;

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.AuthenticationException;

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

import org.springframework.security.web.authentication.AuthenticationFailureHandler;

@Configuration

@EnableWebSecurity

public class MyConfig extends WebSecurityConfigurerAdapter {

    static int count = 0;

    @Autowired

    private CustomUserDetailService customUserDetailService;

    @Override

    protected void configure(HttpSecurity http) throws Exception {

        http

                .authorizeRequests()

                .antMatchers("/signin").permitAll()

                .anyRequest()

                .authenticated()

                .and()

                .formLogin()

                .loginPage("/signin")

                .loginProcessingUrl("/dologin")

                .defaultSuccessUrl("/admin")

                .failureHandler(new AuthenticationFailureHandler() {

                    @Override

                    public void onAuthenticationFailure(HttpServletRequest request, HttpServletResponse response,

                            AuthenticationException exception) throws IOException, ServletException {

                        System.out.println("A user has failed to login. Error: " + exception.getMessage());

                        if (count == 3) {

                        }

                        response.sendRedirect("signin?error");

                    }

                })

                .and()

                .rememberMe().userDetailsService(customUserDetailService)

                .tokenValiditySeconds(3 \* 24 \* 60 \* 60);

    }

    @Override

    protected void configure(AuthenticationManagerBuilder auth) throws Exception {

        auth

                .userDetailsService(customUserDetailService)

                .passwordEncoder(passwordEncoder());

    }

    @Bean

    public BCryptPasswordEncoder passwordEncoder() {

        return new BCryptPasswordEncoder(10);

    }

}

//

package com.example.assignment2.controller;

import org.springframework.stereotype.Controller;

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

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

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

@Controller

public class HomeController {

    @GetMapping("/signin")

    public String login() {

        return "login.html";

    }

    @GetMapping("/admin")

    @ResponseBody

    public String hellWorld() {

        return "Hello World!!";

    }

    @PostMapping("login-fail")

    public String handleFailedLogin() {

        System.out.println("A User has failed to login");

        return "redirect:/signin?error";

    }

}

//

package com.example.assignment2.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;

public class CustomUserDetail implements UserDetails {

    private User user;

    public CustomUserDetail(User user) {

        this.user = user;

    }

    public CustomUserDetail() {

    }

    @Override

    public Collection<? extends GrantedAuthority> getAuthorities() {

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

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

        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;

    }

}

//

package com.example.assignment2.model;

import org.springframework.data.mongodb.core.mapping.Document;

@Document(collection = "user")

public class User {

    String username;

    String password;

    String email;

    String role;

    public User() {

    }

    public User(String username, String password, String email, String role) {

        this.username = username;

        this.password = password;

        this.email = email;

        this.role = role;

    }

    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 String getEmail() {

        return email;

    }

    public void setEmail(String email) {

        this.email = email;

    }

    public String getRole() {

        return role;

    }

    public void setRole(String role) {

        this.role = role;

    }

    @Override

    public String toString() {

        return "User [email=" + email + ", password=" + password + ", role=" + role + ", username=" + username + "]";

    }

}

//

package com.example.assignment2.repository;

import com.example.assignment2.model.User;

import org.springframework.data.mongodb.repository.MongoRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface UserRepo extends MongoRepository<User, String> {

    public User findByUsername(String username);

}

//

package com.example.assignment2.service;

import com.example.assignment2.model.CustomUserDetail;

import com.example.assignment2.model.User;

import com.example.assignment2.repository.UserRepo;

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;

@Service

public class CustomUserDetailService implements UserDetailsService {

    @Autowired

    private UserRepo userRepo;

    @Override

    public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

        User user = this.userRepo.findByUsername(username);

        if (user == null) {

            throw new UsernameNotFoundException("INVALID USERNAME!!");

        }

        return new CustomUserDetail(user);}}

3)

//Assignment2Application.java

package com.example.assignment2;

import com.example.assignment2.model.User;

import com.example.assignment2.repository.UserRepo;

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;

@SpringBootApplication

public class Assignment2Application implements CommandLineRunner {

    @Autowired

    private UserRepo userRepo;

    @Autowired

    private BCryptPasswordEncoder bCryptPasswordEncoder;

    public static void main(String[] args) {

        SpringApplication.run(Assignment2Application.class, args);

    }

    @Override

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

        User user1 = new User();

        user1.setEmail("himanshuwakade@gmail.com");

        user1.setUsername("Himanshu");

        user1.setPassword(bCryptPasswordEncoder.encode("himan12"));

        user1.setRole("ROLE\_ADMIN");

        userRepo.save(user1);

        User user2 = new User();

        user2.setEmail("user@gmail.com");

        user2.setUsername("yuvi");

        user2.setPassword(bCryptPasswordEncoder.encode("yuvi12"));

        user2.setRole("ROLE\_NORMAL");

        userRepo.save(user2);

    }

}

//MyConfig.java

package com.example.assignment2.config;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.example.assignment2.service.CustomUserDetailService;

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.AuthenticationException;

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

import org.springframework.security.web.authentication.AuthenticationFailureHandler;

@Configuration

@EnableWebSecurity

public class MyConfig extends WebSecurityConfigurerAdapter {

    static int count = 0;

    @Autowired

    private CustomUserDetailService customUserDetailService;

    @Override

    protected void configure(HttpSecurity http) throws Exception {

        http

                .authorizeRequests()

                .antMatchers("/signin").permitAll()

                .anyRequest()

                .authenticated()

                .and()

                .formLogin()

                .loginPage("/signin")

                .loginProcessingUrl("/dologin")

                .defaultSuccessUrl("/admin")

                .failureHandler(new AuthenticationFailureHandler() {

                    @Override

                    public void onAuthenticationFailure(HttpServletRequest request, HttpServletResponse response,

                            AuthenticationException exception) throws IOException, ServletException {

                        System.out.println("A user has failed to login. Error: " + exception.getMessage());

                        if (count == 3) {

                        }

                        response.sendRedirect("signin?error");

                    }

                })

                .and()

                .rememberMe().userDetailsService(customUserDetailService)

                .tokenValiditySeconds(3 \* 24 \* 60 \* 60);

    }

    @Override

    protected void configure(AuthenticationManagerBuilder auth) throws Exception {

        auth

                .userDetailsService(customUserDetailService)

                .passwordEncoder(passwordEncoder());

    }

    @Bean

    public BCryptPasswordEncoder passwordEncoder() {

        return new BCryptPasswordEncoder(10);

    }

}

//HomeController.java

package com.example.assignment2.controller;

import org.springframework.stereotype.Controller;

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

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

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

@Controller

public class HomeController {

    @GetMapping("/signin")

    public String login() {

        return "login.html";

    }

    @GetMapping("/admin")

    @ResponseBody

    public String hellWorld() {

        return "Hello World!!";

    }

    @PostMapping("login-fail")

    public String handleFailedLogin() {

        System.out.println("A User has failed to login");

        return "redirect:/signin?error";

    }

}

//CustomUserDetail.java

package com.example.assignment2.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;

public class CustomUserDetail implements UserDetails {

    private User user;

    public CustomUserDetail(User user) {

        this.user = user;

    }

    public CustomUserDetail() {

    }

    @Override

    public Collection<? extends GrantedAuthority> getAuthorities() {

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

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

        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;

    }

}

//User.java

package com.example.assignment2.model;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class User {

    @Id

    String username;

    String password;

    String email;

    String role;

    public User() {

    }

    public User(String username, String password, String email, String role) {

        this.username = username;

        this.password = password;

        this.email = email;

        this.role = role;

    }

    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 String getEmail() {

        return email;

    }

    public void setEmail(String email) {

        this.email = email;

    }

    public String getRole() {

        return role;

    }

    public void setRole(String role) {

        this.role = role;

    }

    @Override

    public String toString() {

        return "User [email=" + email + ", password=" + password + ", role=" + role + ", username=" + username + "]";

    }

}

//UserRepo.java

package com.example.assignment2.repository;

import com.example.assignment2.model.User;

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

import org.springframework.stereotype.Repository;

@Repository

public interface UserRepo extends JpaRepository<User, String> {

    public User findByUsername(String username);

}

//CustomUserDetailService.java

package com.example.assignment2.service;

import com.example.assignment2.model.CustomUserDetail;

import com.example.assignment2.model.User;

import com.example.assignment2.repository.UserRepo;

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;

@Service

public class CustomUserDetailService implements UserDetailsService {

    @Autowired

    private UserRepo userRepo;

    @Override

    public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

        User user = this.userRepo.findByUsername(username);

        if (user == null) {

            throw new UsernameNotFoundException("INVALID USERNAME!!");

        }

        return new CustomUserDetail(user);

    }

}

//application.properties

spring.datasource.name=security1.3

spring.datasource.url=jdbc:mysql://localhost:3306/security1.3

spring.datasource.username= root

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

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

spring.jpa.hibernate.ddl-auto=update