Spring Security Assignments

Submitted By – Sanket Bolawmar

1. Design and develop a Spring security Hello World application by using default login form provided by spring security to secure URL access say, for example to access the content of an “admin” page, user needs to enter valid credentials. User must also logged out if successfully logged in.

Use Java Based and annotation based configuration and In-memory authentication.

**Refer Spring.Security.AssignmentQ1 file**

**Application Class:**

package com.springsecurity.assignmentQ1;

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);

}

}

**MainController Class:**

package com.springsecurity.assignmentQ1;

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);

}

}

**AppConfig Class:**

package com.springsecurity.assignmentQ1.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.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 AppConfig extends WebSecurityConfigurerAdapter {

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth

.inMemoryAuthentication()

.withUser("Sanket Bolamwar")

.password(this.passwordEncoder().encode("Pass@123"))

.roles("ADMIN");

}

@Bean

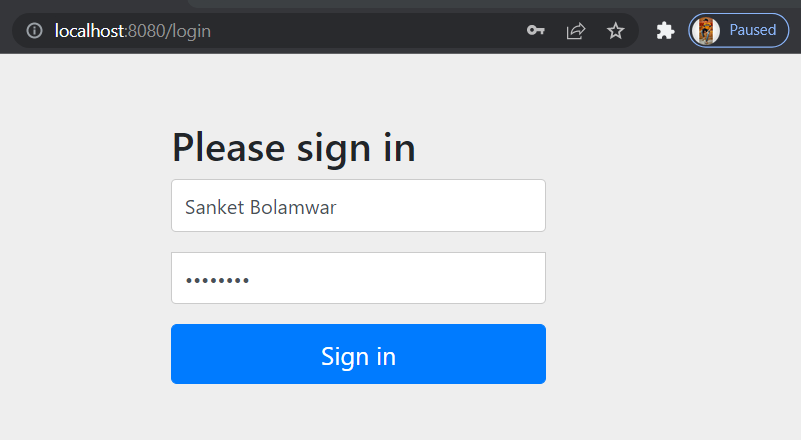
public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder(10);

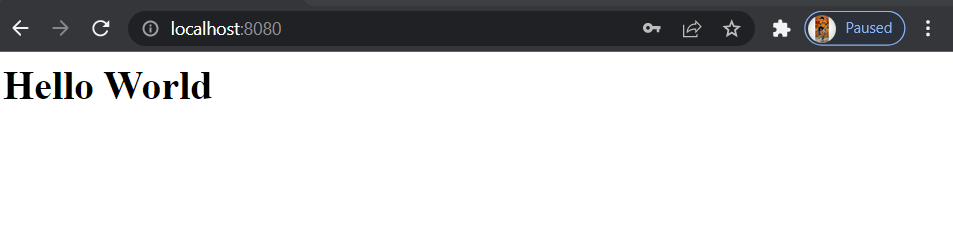
}

}

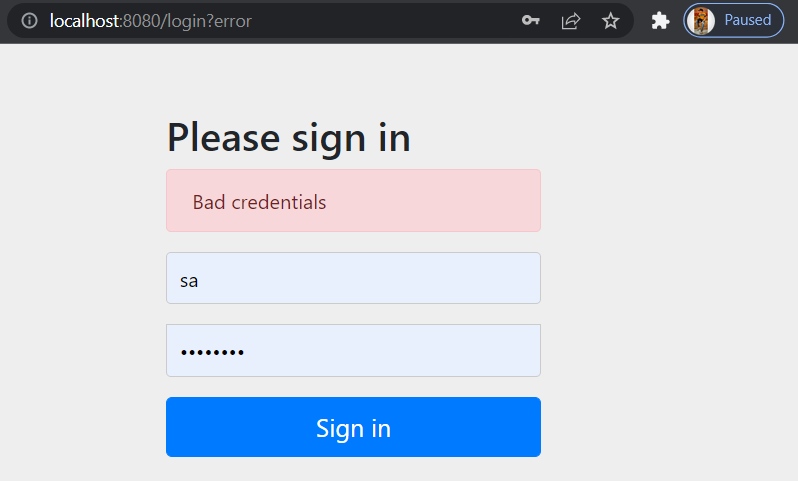
**Login Form:**

****

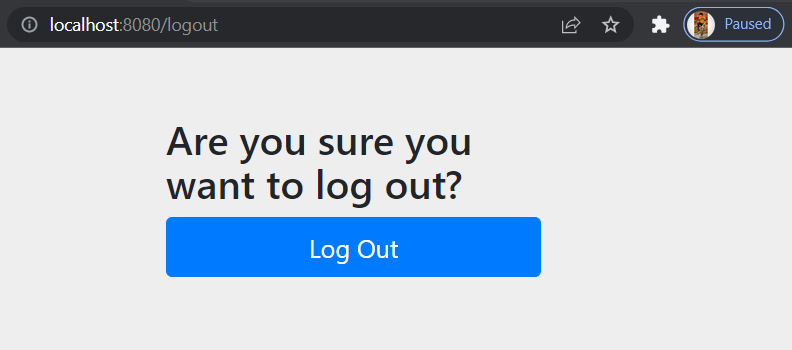
**Correct Credentials:**

****

**Wrong Credentials:**



**Logout :**

****

1. Modify the above application to use custom login form instead of default login form provided by spring security.

Use Java Based and annotation based configuration and In-memory authentication.

**Refer Spring.Security.AssignmentQ2 file**

**Application Class:**

package com.springsecurity.assignmentQ2;

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);

}

}

**MainController:**

package com.springsecurity.assignmentQ2.controller;

import org.springframework.stereotype.Controller;

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

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

import org.springframework.stereotype.Controller;

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

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!!";

}

}

**AppConfig Class:**

package com.springsecurity.assignmentQ2.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 AppConfig extends WebSecurityConfigurerAdapter {

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.authorizeRequests()

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

.anyRequest()

.authenticated()

.and()

.formLogin()

.loginPage("/signin")

.loginProcessingUrl("/dologin")

.defaultSuccessUrl("/admin")

.and().logout();

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth

.inMemoryAuthentication()

.withUser("Sanket")

.password(this.passwordEncoder().encode("Pass@123"))

.roles("ADMIN");

auth

.inMemoryAuthentication()

.withUser("Atul")

.password(this.passwordEncoder().encode("Pass@12345"))

.roles("NORMAL");

}

@Bean

public PasswordEncoder passwordEncoder() {

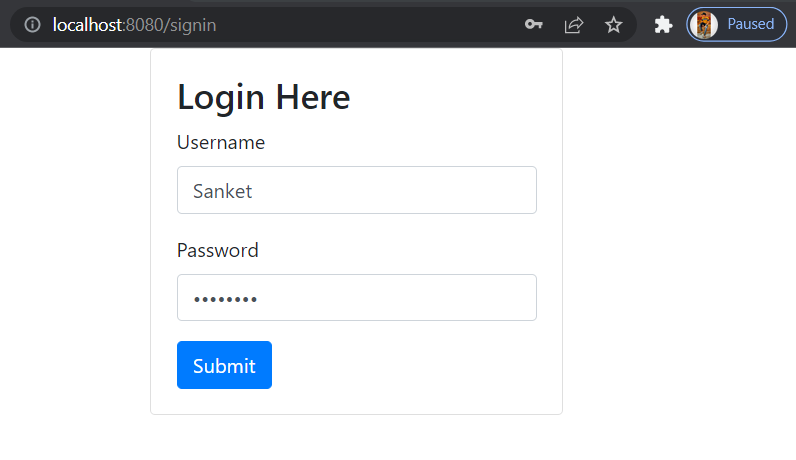
return new BCryptPasswordEncoder(10);

}

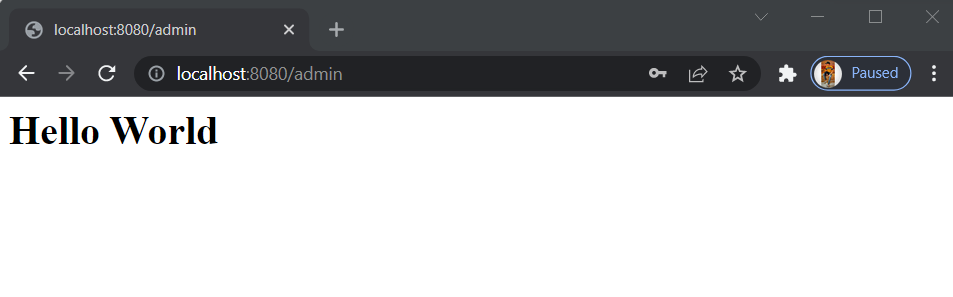
}

**Login.html file :** refer login.html resource files for used for custom login form

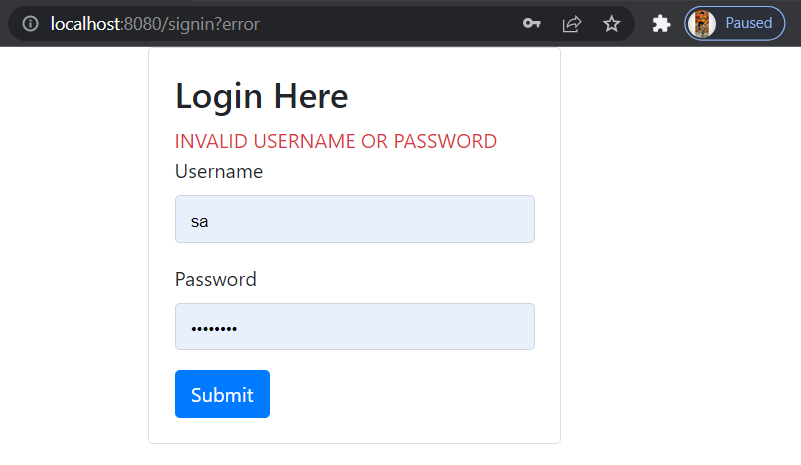
**Login Form:**

****

**Correct Credentials:**

****

**Wrong Credentials:**



1. Modify the above application and use database authentication using JDBC instead of In- memory authentication.

Use Java Based and annotation based configuration and In-memory authentication.

**Refer Spring.Security.AssignmentQ3 file**

**Application Class:**

**MainController Class:**

**AppConfig Class:**

**Login Form:**

**C**

1. Modify the above application to limit the number of login attempts.
2. Modify the above application to implement “remember me” functionality.
3. Modify the above application to secure the password by encoding it. (You may use some encryption technique)