4. Implement Spring Security with Authentication

Source Code:

Open pom.xml

```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>2.4.4</version>

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

</parent>

<groupId>com.project</groupId>

<artifactId>SpringSecurity</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>SpringSecurity</name>

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

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

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

```xml
        <artifactId>spring-boot-starter-data-jdbc</artifactId>

    </dependency>

    <dependency>

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

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

    </dependency>

    <dependency>

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

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

    </dependency>

    <dependency>

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

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

    </dependency>

    <dependency>

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

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

    </dependency>
```

```xml
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-thymeleaf</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
            <optional>true</optional>
        </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.apache.tomcat.embed</groupId>
            <artifactId>tomcat-embed-jasper</artifactId>
            <scope>provided</scope>
        </dependency>
    </dependencies>
```

```xml
<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>
```

Create package com.project.SpringSecurity

Create MvcConfig.java

```java
package com.project.SpringSecurity;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.servlet.config.annotation.ViewControllerRegistry;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration

public class MvcConfig implements WebMvcConfigurer{

 public void addViewControllers(ViewControllerRegistry registry) {

   registry.addViewController("/index").setViewName("index");

   registry.addViewController("/").setViewName("index");

   registry.addViewController("/login").setViewName("login");

   registry.addViewController("/welcome").setViewName("welcome");
```

```
        }

        }
```

Create SpringSecurityApplication.java

```java
package com.project.SpringSecurity;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

import com.project.SpringSecurity.repositories.UsersRepository;

@SpringBootApplication

@EnableJpaRepositories(basePackageClasses = UsersRepository.class)

public class SpringSecurityApplication {

public static void main(String[] args) {

 SpringApplication.run(SpringSecurityApplication.class, args);

}

}
```

Create WebSecurityConfig.java

```java
package com.project.SpringSecurity;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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;
```

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

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

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

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

@SuppressWarnings("deprecation")

@Configuration

@EnableWebSecurity

public class WebSecurityConfig extends WebSecurityConfigurerAdapter {

Logger logger = LoggerFactory.getLogger(WebSecurityConfig.class);

@Autowired

UserDetailsService userDetailsService;

@Bean

public PasswordEncoder getPasswordEncoder() {

return NoOpPasswordEncoder.getInstance();

}

@Autowired

public void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(userDetailsService);

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.authorizeRequests()

.antMatchers("/", "/index").permitAll()

.anyRequest().authenticated()

.and()

.formLogin()
```

```
.loginPage("/login")

.defaultSuccessUrl("/welcome")

.failureUrl("/login?error=true")

.permitAll()

.and()

.logout()

.logoutSuccessUrl("/login?logout=true")

.invalidateHttpSession(true)

.permitAll()

.and()

.csrf()

.disable();

}

}
```

Create package com.project.SpringSecurity.entities

Create MyUserDetails.java

```java
package com.project.SpringSecurity.entities;

import java.util.Collection;

import org.springframework.security.core.GrantedAuthority;

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

public class MyUserDetails implements UserDetails{

private static final long serialVersionUID = 1L;

private String userName;

private String password;

public MyUserDetails() {

}
```

```java
    public MyUserDetails(User user) {

        this.userName = user.getName();

        this.password = user.getPassword();

    }

    @Override

    public Collection<? extends GrantedAuthority> getAuthorities() {

        return null;

    }

    @Override

    public String getPassword() {

        return password;

    }

    @Override

    public String getUsername() {

        return userName;

    }

    @Override

    public boolean isAccountNonExpired() {

        return true;

    }

    @Override

    public boolean isAccountNonLocked() {

        return true;

    }

    @Override

    public boolean isCredentialsNonExpired() {
```

```java
return true;

}

@Override

public boolean isEnabled() {

return true;

}

}
```

Create User.java

```java
package com.project.SpringSecurity.entities;

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 Integer id;

private String name;

private String password;

public User() {
```

```java
super();

}

public User(String name, String password) {

super();

this.name = name;

this.password = password;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}
```

Create package com.project.SpringSecurity.repositories

Create UsersRepository.java

```java
package com.project.SpringSecurity.repositories;

import java.util.Optional;

import org.springframework.data.repository.CrudRepository;
```

```java
import org.springframework.stereotype.Repository;

import com.project.SpringSecurity.entities.User;



@Repository

public interface UsersRepository extends CrudRepository<User, Integer>{



public Optional<User> findUserByName(String name);

}
```

Create package com.project.SpringSecurity.services

Create MyUserDetailsService.java

```java
package com.project.SpringSecurity.services;

import java.util.Optional;

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

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

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

import org.springframework.stereotype.Service;



import com.project.SpringSecurity.entities.MyUserDetails;

import com.project.SpringSecurity.entities.User;

import com.project.SpringSecurity.repositories.UsersRepository;

@Service

public class MyUserDetailsService implements UserDetailsService {

@Autowired

UsersRepository userRepo;

public User GetUserByName(String name) {
```

```
Optional<User> user = userRepo.findUserByName(name);

if(!user.isPresent()) throw new RuntimeException();

return user.get();

}

@Override

public org.springframework.security.core.userdetails.UserDetails loadUserByUsername(String username)

throws UsernameNotFoundException {

return new MyUserDetails(GetUserByName(username));

}

}
```

application.properties

```
spring.jpa.hibernate.ddl-auto=update

spring.jpa.hibernate.naming-strategy=org.hibernate.cfg.ImprovedNamingStrategy

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

spring.datasource.url=jdbc:mysql://localhost:3306/mywork

spring.datasource.username=root

spring.datasource.password=password

logging.level.org.springframework.web: DEBUG

spring.thymeleaf.prefix=/WEB-INF/jsp/

spring.thymeleaf.suffix=.jsp

server.port=8080

server.error.whitelabel.enabled=false
```

Create index.jsp

```
<!DOCTYPE html>
```

```html
<html xmlns="http://www.w3.org/1999/xhtml"

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

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

xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">

<head>

<title>Greetings!</title>

</head>

<body>

<h1>Welcome!</h1>

<p>

Click <a th:href="@{/welcome}">here</a> to see a greeting.

</p>

</body>

</html>
```

Create login.jsp

```html
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="https://www.thymeleaf.org"

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

  <head>

    <title>Login</title>

  </head>

  <body>

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

      Invalid username or password.

    </div>

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

You have been logged out.

</div>

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

    <div><label> User Name : <input type="*"text"* name="*"username"*/> </label></div>

    <div><label> Password: <input type="*"password"* name="*"password"*/> </label></div>

    <div><input type="*"submit"* value="*"Sign In"*/></div>

</form>

Return to Main Page

</body>

</html>

Create welcome.jsp

<!DOCTYPE html>

<html xmlns="*"http://www.w3.org/1999/xhtml"* xmlns:th="*"https://www.thymeleaf.org"*

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

<head>

    <title>Welcome!</title>

</head>

<body>

    <h1 th:inline="*"text"*>Hello [[${#httpServletRequest.remoteUser}]]!</h1>

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

    <input type="*"submit"* value="*"Sign Out"*/>

</form>

</body>

</html>