Jenkins:

pipeline {

agent any

stages {

stage('Checkout') {

steps {

checkout scm

}

}

stage('Build') {

steps {

sh 'mvn clean install'

}

}

stage('Deploy') {

steps {

// Add deployment steps here, such as deploying to a server

}

}

}

post {

success {

// Notify or perform additional actions on success

}

failure {

// Notify or perform additional actions on failure

}

}

}

Login.component.html:

<!-- login.component.html -->

<div class="login-container">

<h2>Login</h2>

<form (ngSubmit)="login()">

<div class="form-group">

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

<input type="text" id="username" name="username" [(ngModel)]="username" required>

</div>

<div class="form-group">

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

<input type="password" id="password" name="password" [(ngModel)]="password" required>

</div>

<button type="submit">Login</button>

</form>

</div>

auth.service.ts

import { Injectable } from '@angular/core';

import { HttpClient } from '@angular/common/http';

@Injectable({

providedIn: 'root'

})

export class AuthService {

constructor(private http: HttpClient) { }

login(username: string, password: string) {

// Send a POST request to your Spring Security authentication endpoint

return this.http.post('/api/login', { username, password });

}

logout() {

// Implement logout logic here

}

}

login.component.ts

import { Component } from '@angular/core';

import { AuthService } from './auth.service';

@Component({

selector: 'app-login',

template: `

<form (ngSubmit)="login()">

<input [(ngModel)]="username" type="text" placeholder="Username">

<input [(ngModel)]="password" type="password" placeholder="Password">

<button type="submit">Login</button>

</form>

`

})

export class LoginComponent {

username: string = '';

password: string = '';

constructor(private authService: AuthService) {}

login() {

this.authService.login(this.username, this.password).subscribe(

response => {

// Handle successful login

},

error => {

// Handle login error

}

);

}

}

auth.guard.ts

import { Injectable } from '@angular/core';

import { CanActivate, Router } from '@angular/router';

@Injectable({

providedIn: 'root'

})

export class AuthGuard implements CanActivate {

constructor(private router: Router, private authService: AuthService) {}

canActivate(): boolean {

if (this.authService.isAuthenticated()) {

return true;

} else {

this.router.navigate(['/login']);

return false;

}

}

}

SecurityConfig.java

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter {

@Autowired

public void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {

auth.inMemoryAuthentication()

.withUser("user").password(passwordEncoder().encode("password")).roles("USER");

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.authorizeRequests()

.antMatchers("/api/login").permitAll()

.anyRequest().authenticated()

.and()

.formLogin()

.loginPage("/login")

.permitAll()

.and()

.logout()

.permitAll();

}

}