**Sample exercises on Centralized Authentication and SSO with Spring Boot 3 and Spring Cloud**

**Exercise 1: Implementing Centralized Authentication with OAuth 2.1/OIDC**

package com.example.oauth2\_central\_auth.controller;

import java.security.Principal;

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

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

@RestController

public class UserController {

@GetMapping("/user")

public Principal user(Principal principal) {

return principal;

}

@GetMapping("/")

public String home(Principal principal) {

if (principal != null) {

return "Welcome, " + principal.getName() + "! You are logged in.";

}

return "Welcome! Please log in.";

}

}

package com.example.oauth2\_central\_auth;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Oauth2CentralAuthApplication {

public static void main(String[] args) {

SpringApplication.run(Oauth2CentralAuthApplication.class, args);

}

}

package com.example.oauth2\_central\_auth;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.web.SecurityFilterChain;

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {

http

.authorizeHttpRequests(authorizeRequests ->

authorizeRequests

.anyRequest().authenticated() // All requests require authentication

)

.oauth2Login(oauth2Login ->

oauth2Login

.defaultSuccessUrl("/", true) // Redirect to root after successful login

.failureUrl("/login?error") // Redirect to /login?error on failure

)

.formLogin(form -> form.disable()); // Disable form login if you only want OAuth2

return http.build();

}

}

spring:

security:

oauth2:

client:

registration:

my-client:

client-id: 387690386297-5fqq3efnbffl7m03qdlhrr6bt4sokf6c.apps.googleusercontent.com

client-secret: GOCSPX-xxMvOd\_xGqzvc0KEDMfUZcFtJ\_DD scope: openid, profile, email

authorization-grant-type: authorization\_code

redirect-uri: "{baseUrl}/login/oauth2/code/{registrationId}"

provider:

my-client:

authorization-uri: https://accounts.google.com/o/oauth2/auth

token-uri: https://oauth2.googleapis.com/token

user-info-uri: https://openidconnect.googleapis.com/v1/userinfo

user-name-attribute: sub

**Sample Hands-on Exercises on Edge Services and API Gateway with Spring Boot 3 and Spring Cloud**

**Exercise 1: Implementing Edge Services for Routing and Filtering**

package com.example.edge\_service;

import org.springframework.cloud.gateway.filter.GlobalFilter;

import org.springframework.core.annotation.Order;

import org.springframework.stereotype.Component;

import org.springframework.web.server.ServerWebExchange;

import reactor.core.publisher.Mono;

import org.springframework.cloud.gateway.filter.GatewayFilterChain;

@Component

@Order(1)

public class LoggingFilter implements GlobalFilter {

@Override

public Mono<Void> filter(ServerWebExchange exchange, GatewayFilterChain chain) {

System.out.println("Request: " + exchange.getRequest().getURI());

return chain.filter(exchange);

}

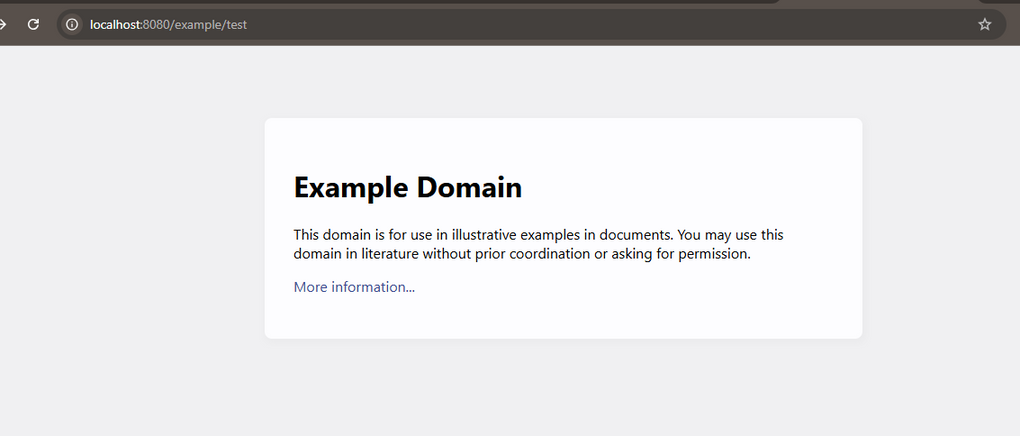
}

spring.application.name=edge-service

spring.cloud.gateway.server.webflux.routes[0].id=example\_route

spring.cloud.gateway.server.webflux.routes[0].uri=http://example.org

spring.cloud.gateway.server.webflux.routes[0].predicates[0]=Path=/example/\*\*



**Exercise 2: Load Balancing in an API Gateway**

package com.example.edge\_service;

import org.springframework.beans.factory.ObjectProvider;

import org.springframework.cloud.loadbalancer.core.RandomLoadBalancer;

import org.springframework.cloud.loadbalancer.core.ReactorServiceInstanceLoadBalancer;

import org.springframework.cloud.loadbalancer.core.ServiceInstanceListSupplier;

import org.springframework.cloud.loadbalancer.support.LoadBalancerClientFactory;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.core.env.Environment;

@Configuration

public class LoadBalancerConfiguration {

@Bean

public ReactorServiceInstanceLoadBalancer randomLoadBalancer(

ObjectProvider<ServiceInstanceListSupplier> serviceInstanceListSupplierProvider,

LoadBalancerClientFactory loadBalancerClientFactory,

Environment environment) {

return new RandomLoadBalancer(

serviceInstanceListSupplierProvider,

loadBalancerClientFactory.getName(environment)

);

}

}

spring.application.name=edge-service

spring.cloud.gateway.server.webflux.routes[0].id=example\_route

spring.cloud.gateway.server.webflux.routes[0].uri=http://example.org

spring.cloud.gateway.server.webflux.routes[0].predicates[0]=Path=/example/\*\*

spring.cloud.gateway.server.webflux.routes[1].id=load\_balanced\_route

spring.cloud.gateway.server.webflux.routes[1].uri=lb://example-service

spring.cloud.gateway.server.webflux.routes[1].predicates[0]=Path=/loadbalanced/\*\*

package com.example.service\_instance\_1;

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

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

@RestController

public class HelloController {

@GetMapping("/hello")

public String hello() {

return "Hello from instance 1";

}

}

package com.example.service\_instance\_2;

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

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

@RestController

public class HelloController {

@GetMapping("/hello")

public String hello() {

return "Hello from instance 2";

}

}   
server:

  port: 8080

spring:

  application:

    name: edge-service

  cloud:

    gateway:

      routes:

        - id: example\_route

          uri: http://example.org

          predicates:

            - Path=/example/\*\*

        - id: load\_balanced\_route

          uri: lb://example-service

          predicates:

            - Path=/loadbalanced/\*\*



