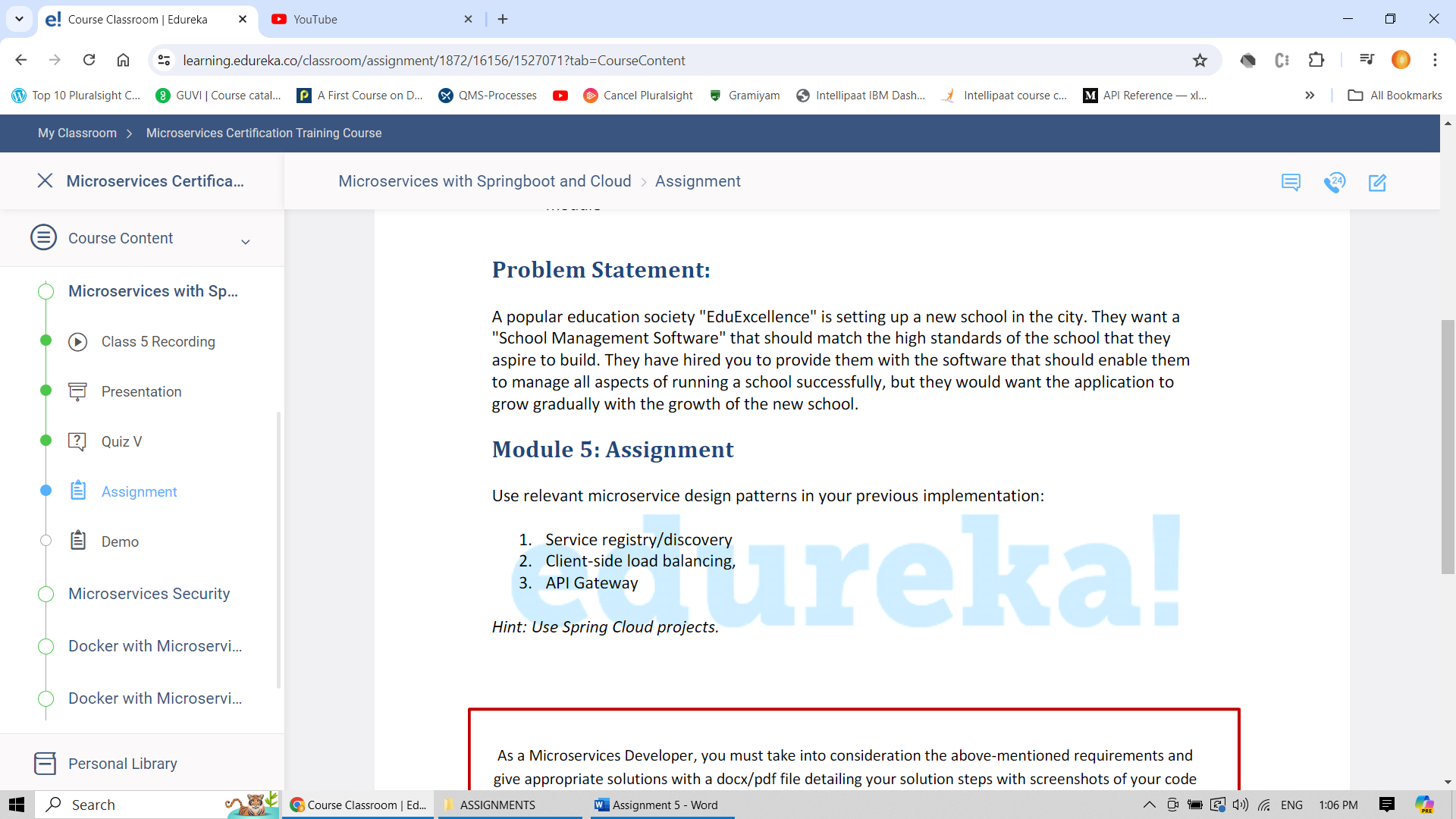
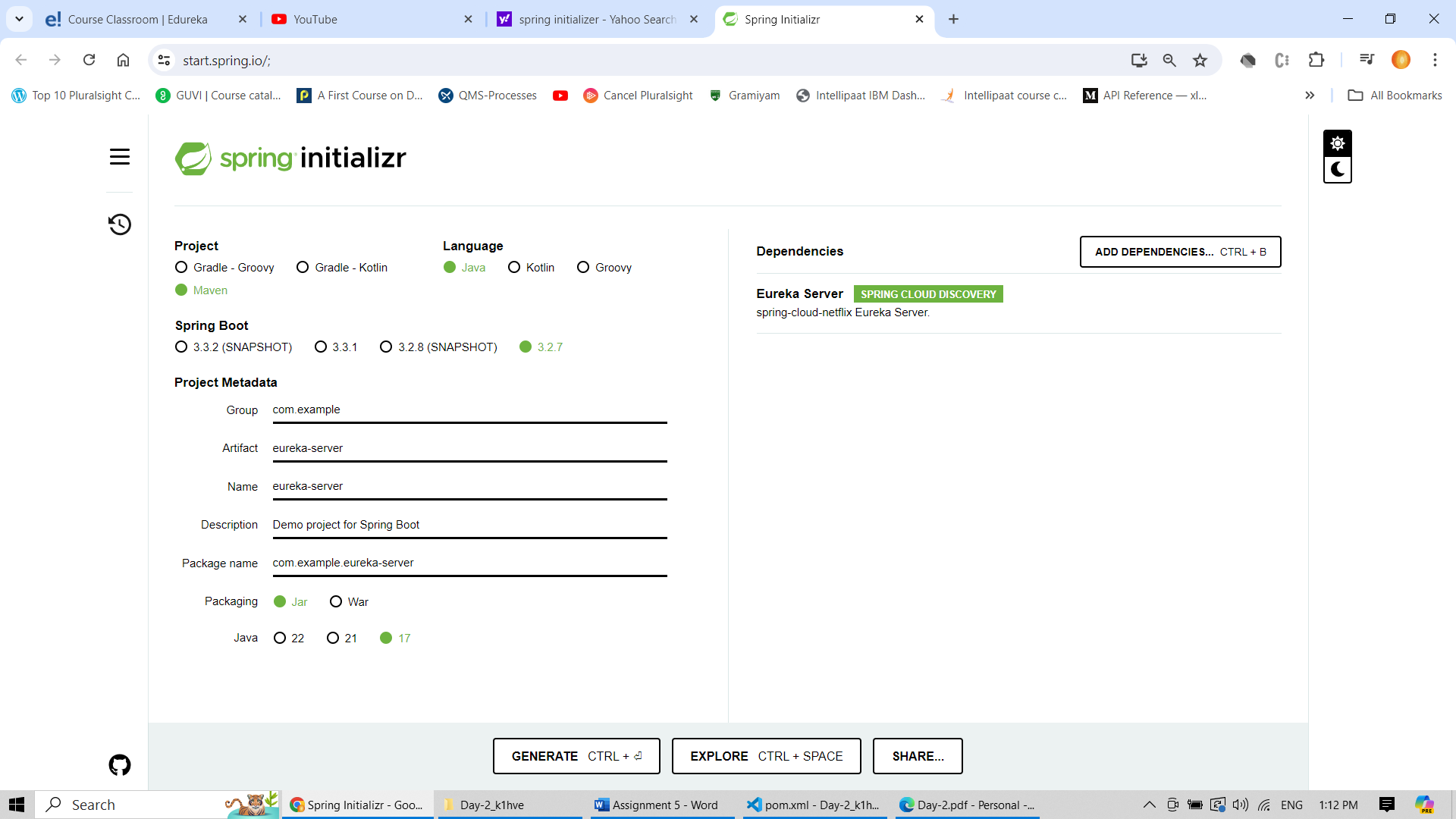
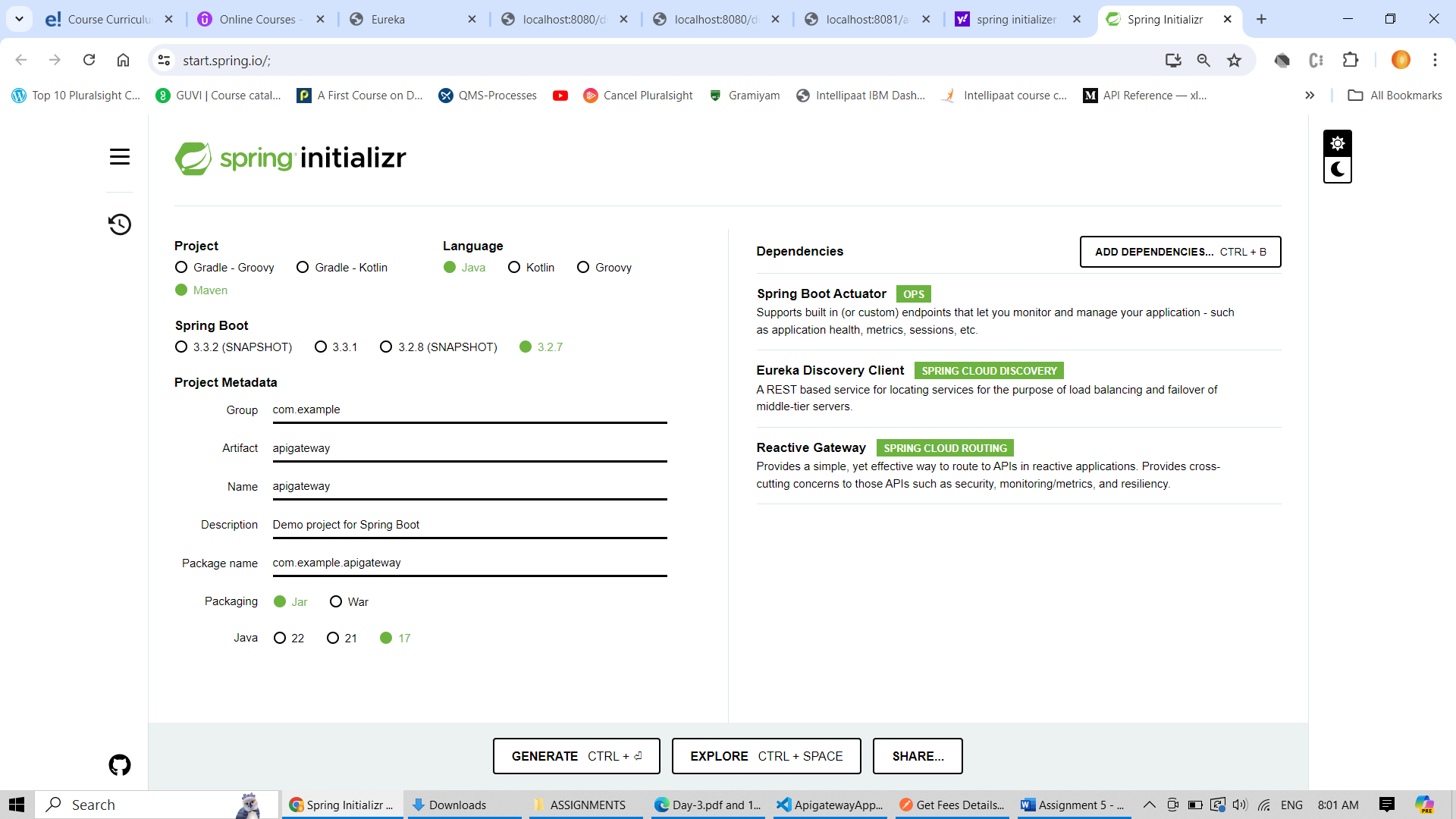
Problem 5:



Step 1: Dependencies for Eureka Server / Service Registry and Discovery



Step 2: Dependencies for apigateway



Step 3: feesms , studentms has already included eureka client lib and hence no change in ms. Code to call feesms api from studentms

WebClientConfig

package com.example.studentms.configuration;

import org.springframework.cloud.client.loadbalancer.LoadBalanced;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.reactive.function.client.WebClient;

@Configuration

public class WebClientConfig {

    @Bean

    @LoadBalanced // Load balance on web client

    public WebClient.Builder builder() {

        return WebClient.builder().baseUrl("http://feesms");

    }

    @Bean

    public WebClient webClient(WebClient.Builder builder) {

        return builder.build();

    }

}

FeesmsClientResource

package com.example.studentms.controller;

import io.github.resilience4j.circuitbreaker.CallNotPermittedException;

import io.github.resilience4j.circuitbreaker.annotation.CircuitBreaker;

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.RestController;

import org.springframework.web.reactive.function.client.WebClient;

import reactor.core.publisher.Mono;

@RestController // MRI write complete class

public class FeesmsClientResource {

    @Autowired

    private WebClient webClient;

    @GetMapping("/client-fees/student/{studentId}")

    @CircuitBreaker(name = "feesmsclient", fallbackMethod = "feesmsfallback")

    public Mono<Object> getFeesFromFeesms(@PathVariable("studentId") Integer studentId) {

        System.out.println("About to call feesms from studentms");

        //Partial uri given in WebClientConfig.java and hence give only the method name here.

        return webClient.get()

                        .uri(uriBuilder -> uriBuilder.path("/fees/student/{studentId}")

                                                      .build(studentId))

                        .retrieve()

                        .bodyToMono(Object.class);

    }

    /\*\*

     \* This method is called when circuit breaker is in closed state but call failed

     \* @param exception

     \* @return

     \*/

    private Mono feesmsfallback(Exception exception) {

        System.out.println("\*\*\*\*\* Exception occured \*\*\*\*\*");

        exception.printStackTrace();

        return Mono.just("Exception occured"); // return just static string

    }

    /\*\*

     \* This method is called when circuit breaker is in open state

     \* @param exception

     \* @return

     \*/

    private Mono feesmsfallback(CallNotPermittedException exception) {

        System.out.println("\*\*\*\*\* CallNotPermittedException occured \*\*\*\*\*");

        exception.printStackTrace();

        return Mono.just("CallNotPermittedException occured");

    }

}

Circuit breaker related changes in application.properties

spring.application.name=studentms

server.port=8081

eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka

management.health.circuitbreakers.enabled=true

management.endpoints.web.exposure.include=\*

management.endpoint.health.show-details=always

logging.level.io.github.resilience4j=debug

resilience4j.circuitbreaker.configs.default.slidingWindowSize=8

resilience4j.circuitbreaker.configs.default.permittedNumberOfCallsInHalfOpenState=2

resilience4j.circuitbreaker.configs.default.slidingWindowType=COUNT\_BASED

resilience4j.circuitbreaker.configs.default.waitDurationInOpenState=30

resilience4j.circuitbreaker.configs.default.failureRateThreshold=50

resilience4j.circuitbreaker.configs.default.minimumNumberOfCalls=4

resilience4j.circuitbreaker.instances.feesmsclient.registerHealthIndicator=true

resilience4j.circuitbreaker.instances.feesmsclient.eventConsumerBufferSize=10

resilience4j.circuitbreaker.instances.feesmsclient.failureRateThreshold=50

resilience4j.circuitbreaker.instances.feesmsclient.minimumNumberOfCalls=5

resilience4j.circuitbreaker.instances.feesmsclient.automatic-transition-from-open-to-half-open-enabled=true

resilience4j.circuitbreaker.instances.feesmsclient.waitDurationInOpenState=20s

resilience4j.circuitbreaker.instances.feesmsclient.permittedNumberOfCallsInHalfOpenState=3

resilience4j.circuitbreaker.instances.feesmsclient.slidingWindowSize=10

resilience4j.circuitbreaker.instances.feesmsclient.slidingWindowType=COUNT\_BASED

Step 4: Eureka server related code changes

package com.example.eureka\_server;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;

@SpringBootApplication

@EnableEurekaServer

public class EurekaServerApplication {

    public static void main(String[] args) {

        SpringApplication.run(EurekaServerApplication.class, args);

    }

}

Application.properties

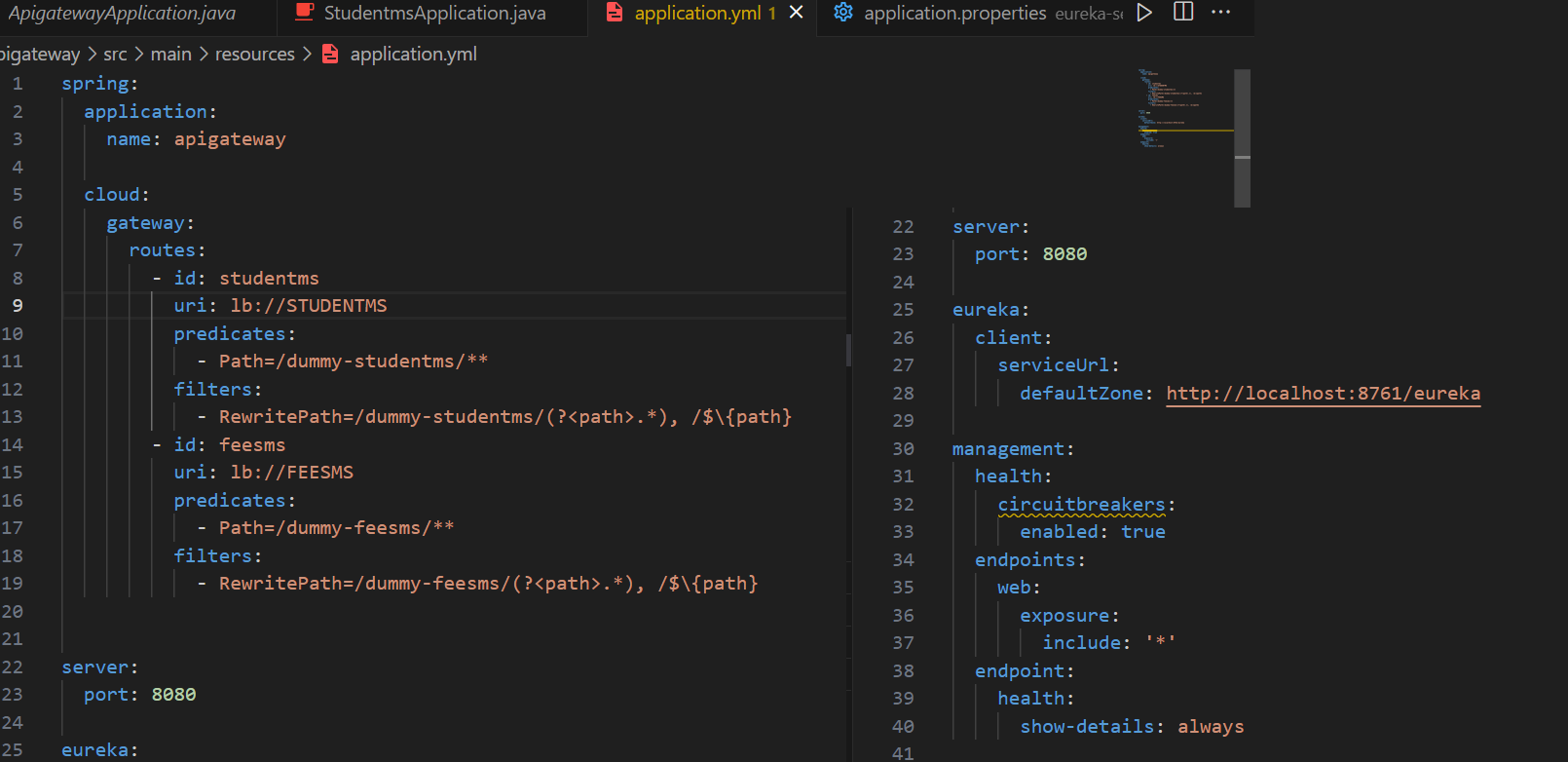
spring.application.name=eureka-server

server.port=8761

eureka.client.register-with-eureka=false

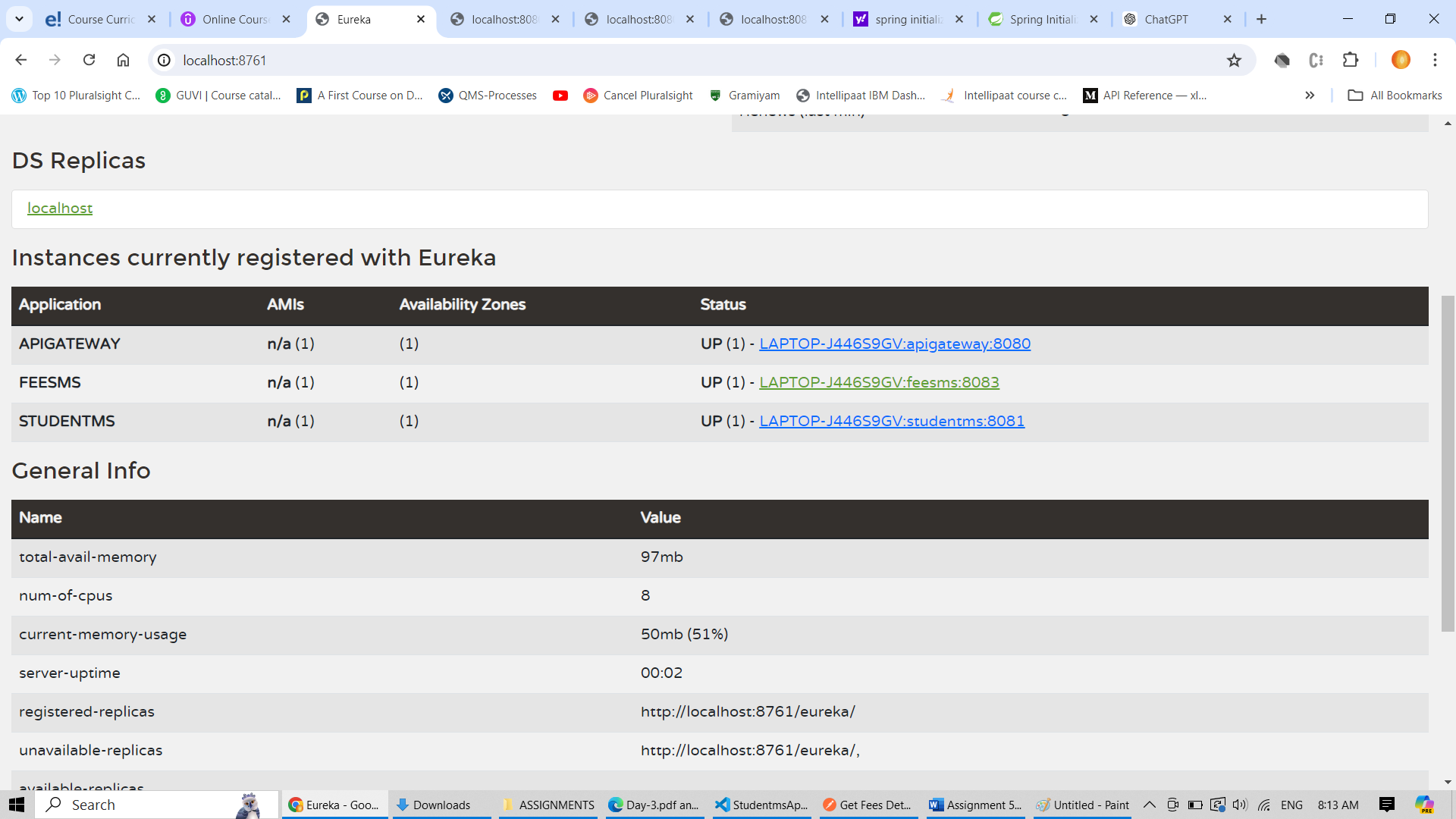
eureka.client.fetch-registry=false

Step 4: API gateway related changes

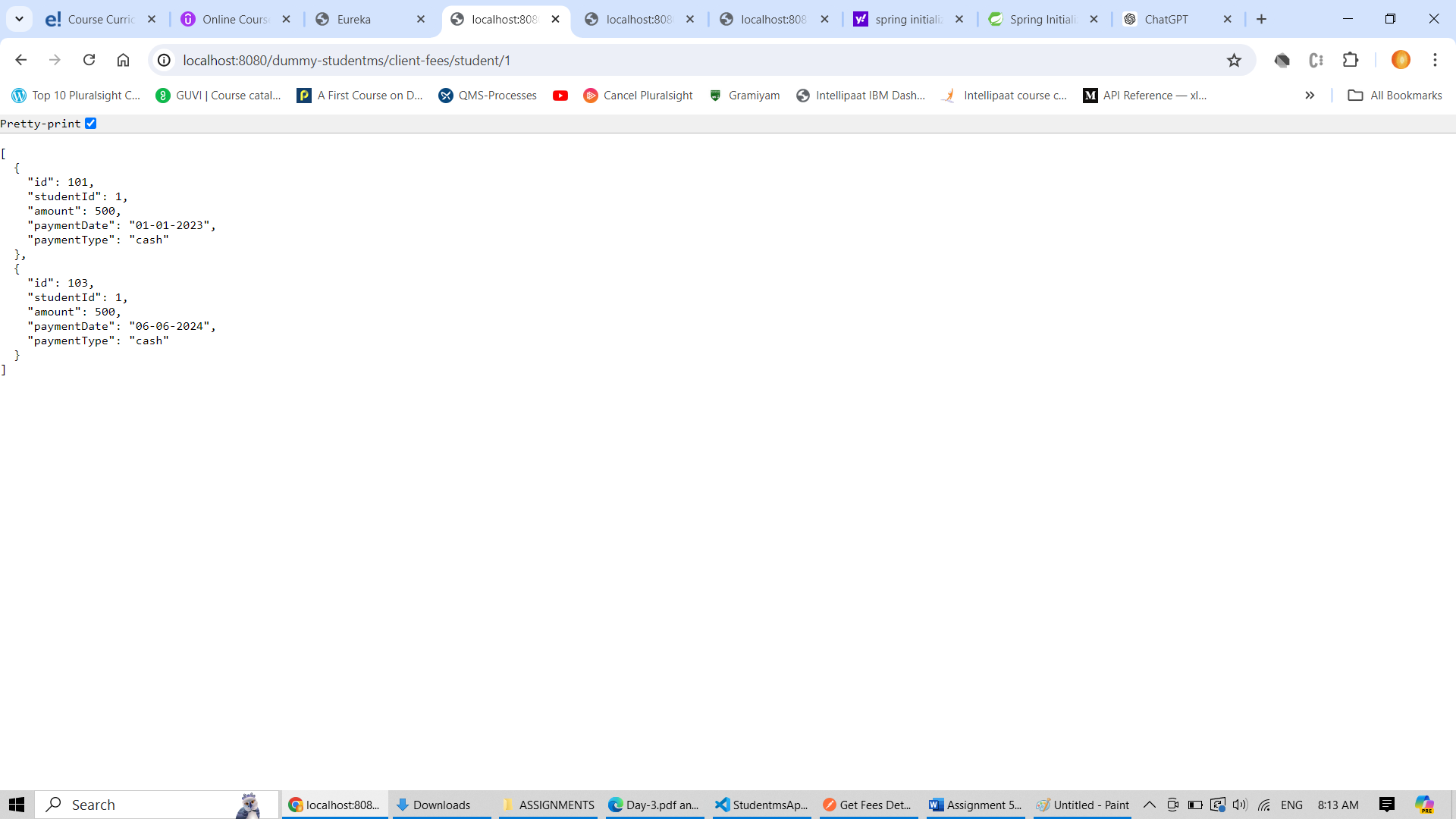


Output screenshots

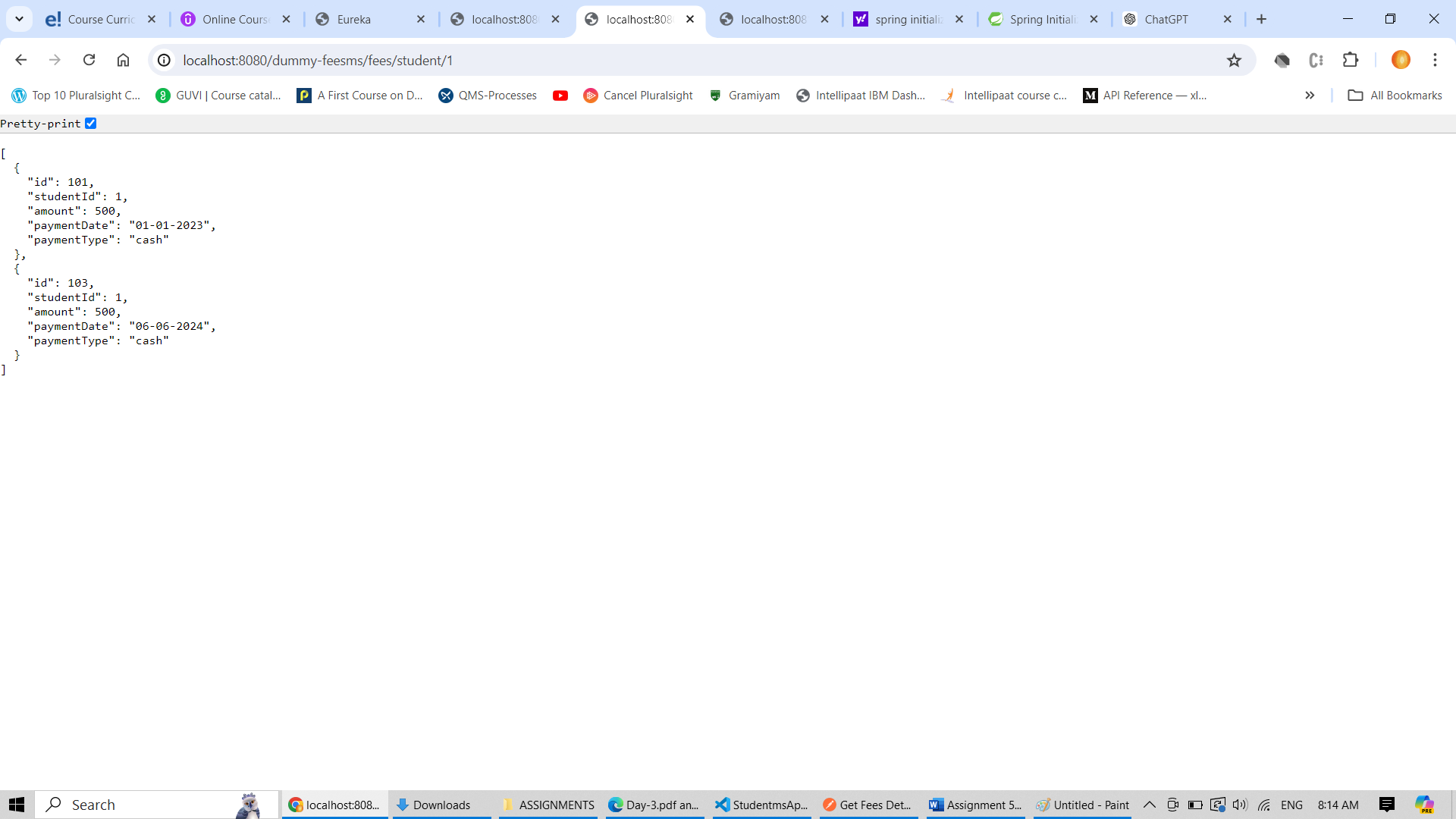
Service discovery could able to identify apigateway, studentms, feesms



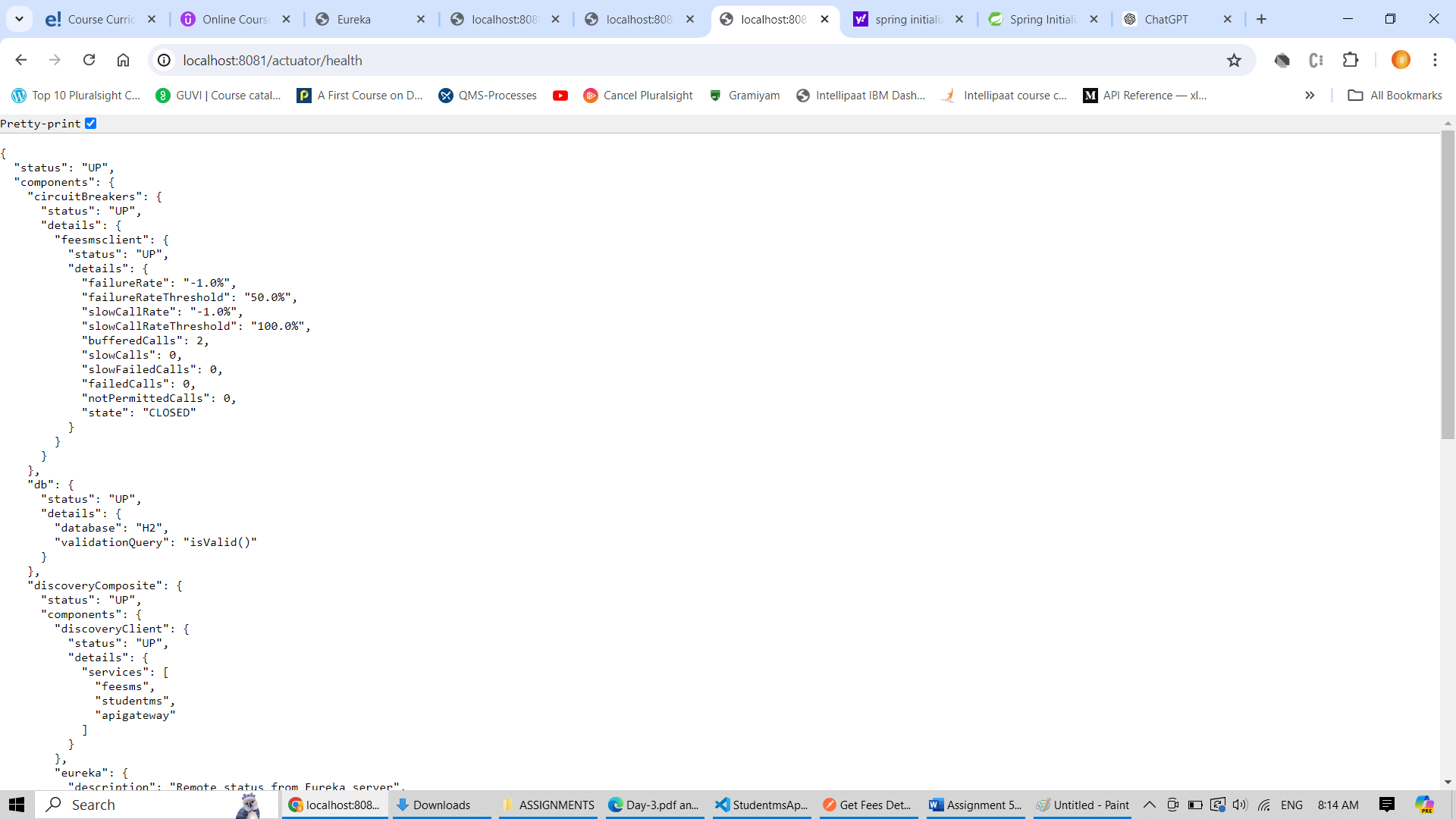
Accessing feesms from studentms using apigateway routes



Accessing feesms using apigateway routes



Accessing health status of apigateway



Accessing health status of circuit breaker which is part of client ms “studentms”

