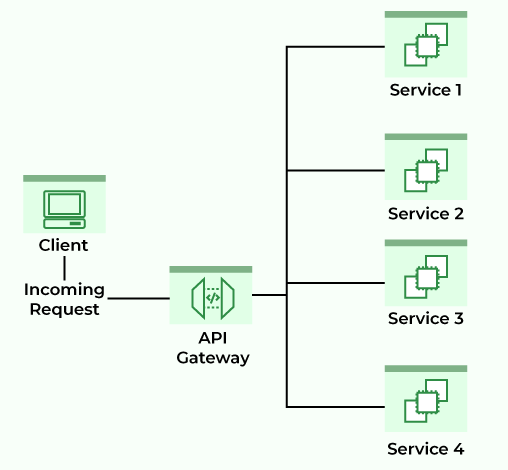
API Gateway:

**Why Implement API Gateway in Microservices?**

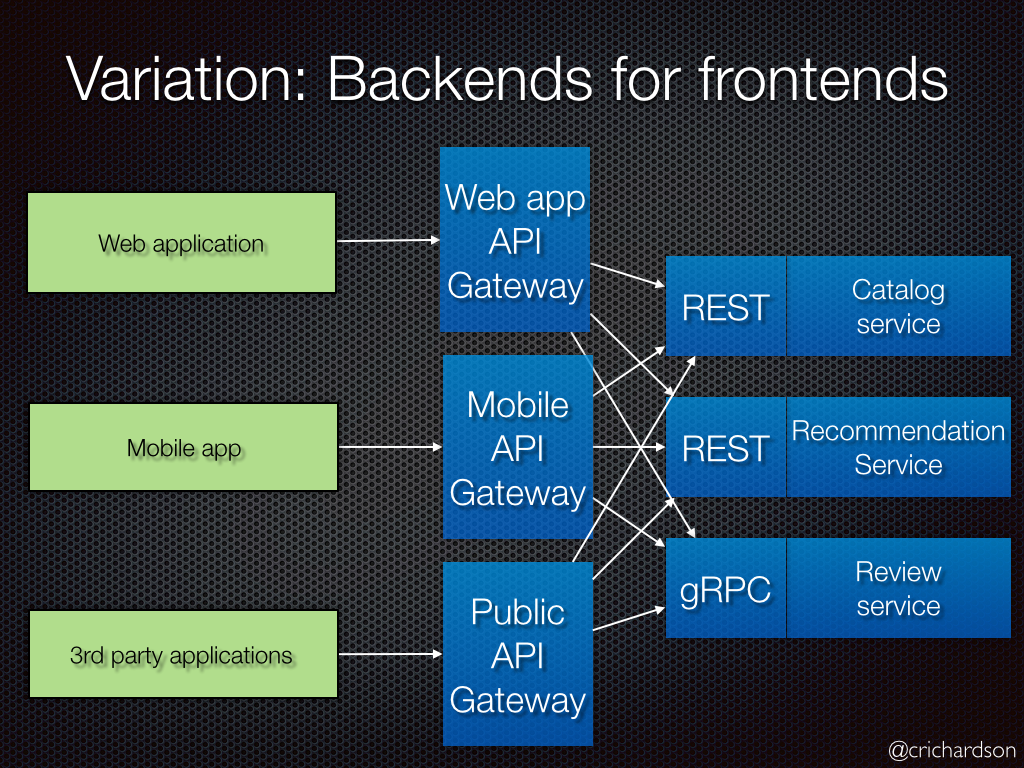
An API gateway simplifies the communication between a client and a service, whether that be between a user’s web browser and a server, or between a frontend application and the backend application that it relies on. **The main purpose of integrating the API gateway in microservice communication is, API Gateway acts as a single entry point to access services**. We will see the whole implementation in the example below. As of now please refer to the below image to get an idea of how the API gateway works.



**What is Spring Cloud Gateway?**

Spring Cloud Gateway provides a library for making API gateways on top of Spring and Java. It provides a flexible way of routing requests based on a number of criteria, as well as focuses on cross-cutting problems like security, resiliency, and monitoring. Some of the important features of Spring Cloud Gateway are:

* It is Built on Spring Framework 5, Project Reactor, and Spring Boot 2.0
* You can integrate Circuit Breaker with Spring Cloud Gateway
* You can integrate Spring Cloud DiscoveryClient
* Predicates and filters are specific to routes
* Path Rewriting
* It is able to match routes on any request attribute, etc.



Pom.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>3.1.7</version>

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

</parent>

<groupId>com.example</groupId>

<artifactId>gateway</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>gateway</name>

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

<properties>

<java.version>17</java.version>

<spring-cloud.version>2022.0.4</spring-cloud.version>

</properties>

<dependencies>

<dependency>

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

<artifactId>spring-boot-devtools</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-gateway</artifactId>

<version>3.1.3</version>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-gateway-mvc</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-netflix-eureka-client</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-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>${spring-cloud.version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<plugins>

<plugin>

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

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

<configuration>

<image>

<builder>paketobuildpacks/builder-jammy-base:latest</builder>

</image>

</configuration>

</plugin>

</plugins>

</build>

</project>

application.yml

server:

port: 8888

eureka:

client:

service-url:

defaultZone: http://localhost:8761/eureka

spring:

main:

web-application-type: reactive

application:

name: gatewayapp

cloud:

gateway:

discovery:

locator:

enabled: true

routes:

- id: producer

uri: http://localhost:8081/

predicates:

- Path=/producer/\*\*

- id: consumer

uri: http://localhost:8083/

predicates:

- Path=/consumer/\*\*

#uri: http://localhost:8081/

#uri: http://localhost:8083/

SpringApplication:

package com.example.adpgateway;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication

@EnableDiscoveryClient

public class AdpGatewayApplication {

public static void main(String[] args) {

SpringApplication.run(AdpGatewayApplication.class, args);

}

}

package com.example.adpgateway;

import org.springframework.cloud.gateway.route.RouteLocator;

import org.springframework.cloud.gateway.route.builder.RouteLocatorBuilder;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

//@Configuration

public class SpringCloudConfiguration {

// @Bean

public RouteLocator gatewayRoutes(RouteLocatorBuilder builder) {

return builder.routes()

.route("producerapp", r -> r.path("/producer/\*\*")

.uri("lb://prodcuerapp")

)

.route("consumerapp", r -> r.path("/consumer/\*\*")

.uri("lb://consumerapp")

)

.build();

}

}