# Manilogix API-Gateway Documentation

## 1. Module Overview

The **API-Gateway** serves as the single entry point to all backend microservices in the Manilogix platform. It handles request routing, load balancing, CORS configuration, and centralized security checks. This module is built using **Spring Cloud Gateway** and integrates with **Eureka Discovery Server** for dynamic service registration.

## 2. Project Structure

api-gateway/  
├── src/main/java/com/manilogix/gateway/  
│ ├── config/  
│ │ └── GatewayConfig.java  
│ ├── filter/  
│ │ └── JwtAuthFilter.java  
│ └── ApiGatewayApplication.java  
└── resources/  
 ├── application.yml  
 └── bootstrap.yml (optional for config server)

## 3. Configuration (application.yml)

server:  
 port: 8080  
  
spring:  
 application:  
 name: api-gateway  
 cloud:  
 gateway:  
 discovery:  
 locator:  
 enabled: true  
 lowerCaseServiceId: true  
 routes:  
 - id: auth-service  
 uri: lb://AUTH-SERVICE  
 predicates:  
 - Path=/api/auth/\*\*  
 - id: inventory-service  
 uri: lb://INVENTORY-SERVICE  
 predicates:  
 - Path=/api/inventory/\*\*  
 - id: order-service  
 uri: lb://ORDER-SERVICE  
 predicates:  
 - Path=/api/orders/\*\*  
 - id: supplier-service  
 uri: lb://SUPPLIER-SERVICE  
 predicates:  
 - Path=/api/suppliers/\*\*  
 - id: notification-service  
 uri: lb://NOTIFICATION-SERVICE  
 predicates:  
 - Path=/api/notifications/\*\*  
  
 eureka:  
 client:  
 service-url:  
 defaultZone: http://localhost:8761/eureka/

## 4. Security Filter

### JwtAuthFilter.java

* Validates JWT from Authorization headers
* Checks roles/claims if needed (optional)
* Forwards valid requests downstream

## 5. GatewayConfig.java

* Registers custom filters (e.g., JwtAuthFilter)
* Can define global CORS policy

## 6. Responsibilities

* Route requests to appropriate microservices
* Apply centralized JWT auth filtering
* Support CORS, rate limiting, logging (can be added)
* Enable service discovery with Eureka

## 7. Dependencies

* Spring Cloud Gateway
* Spring Boot Starter Webflux
* Spring Cloud Netflix Eureka Client
* Spring Boot Configuration Processor
* Spring Security (for auth filter)
* Lombok

## 8. Notes

* Routes are dynamically mapped using service names from Eureka
* Filters can be extended for logging, tracing, throttling, etc.
* Optionally integrates with Spring Cloud Config for central configuration