Introduction

This document provides an overview of the API Gateway, which acts as the entry point for managing and routing requests to various backend services.

The API Gateway offers several features to ensure efficient, secure, and scalable handling of requests, including:

- **Routing**: Directs client requests to the appropriate backend service.
- **Security**: Handles authentication, authorization, and encryption for secure data exchange.
- **Rate Limiting**: Manages traffic flow to prevent overloading services.
- **Caching**: Improves performance by caching frequently requested responses.
- **Logging & Monitoring**: Tracks request metrics and logs for analysis and troubleshooting.
- **Load Balancing**: Distributes incoming requests evenly across backend services for optimal performance.

This documentation will guide you through the process of configuring, using, and extending the API Gateway to meet your application's specific requirements.

Cluster

A **Cluster** represents a group of destination servers (or endpoints) to which the API Gateway can forward requests. It defines load balancing, health checks, and failover policies. Clusters allow the API Gateway to distribute traffic across multiple backend services, ensuring reliability and availability.

Key attributes of a Cluster:

- **Load Balancing**: Distributes traffic among multiple endpoints using a configured strategy Available Policy :
  - FirstAlphabetical
  - RoundRobin
  - LeastRequests
  - Random
  - PowerOfTwoChoices.
- **Health Checks**: Periodically checks the health of backend services, ensuring requests are only sent to operational endpoints.
- **Session Affinity**: Ensures that requests from the same client are routed to the same backend service.
- **Failover**: Redirects traffic to healthy services if one or more endpoints fail.

Example:

```
  // Clusters tell the proxy where and how to forward requests
{
  "ClusterId": "BackendCluster",
  "Destinations": {
    "Destination1": { "Address": "https://service1.example.com" },
    "Destination2": { "Address": "https://service2.example.com" }
  },
  "LoadBalancingPolicy": "RoundRobin"
}
```

Advanced Configuration:

Clusters also provide flexibility for configuring the request client used to communicate with backend services. This can be critical when interacting with services that require specific protocols, security configurations, or custom headers. Key options include the ability to:

- Define timeouts for backend requests.
- Configure retry policies to handle transient failures.
- Customize networking settings, such as connection limits and buffer sizes.

By leveraging these advanced settings, it can fine-tune how requests are handled, ensuring efficient and reliable communication with backend systems. This improves both performance and fault tolerance when interacting with backend services.

Route

A **Route** defines how client requests are matched and forwarded to a backend service through a specific cluster. Routes allow fine-grained control over request handling, including matching based on URL paths, HTTP methods, hostnames, and more.

Key attributes of a Route:

- **Path Matching**: Specifies which URL paths should trigger the route.
- **HTTP Methods**: Indicates the supported HTTP methods (e.g., GET, POST, PUT).
- **Hostnames**: Optionally matches requests based on hostnames (useful for multi-tenant environments).
- **Headers & Query Parameters**: Adds conditions for matching routes based on specific request headers or query parameters.
- **Cluster Mapping**: Links the route to a **Cluster**, where the request is forwarded to the appropriate destination.
- **Priority**: Sets the order in which routes are evaluated if multiple routes match a request.
- **Rate Limiting**: Optionally applies rate limiting rules to the route to prevent overloading backend services.

Example Route Configuration:

```
{
  "RouteId": "ProductRoute",
  "Match": {
    "Path": "/api/products/{id}",
    "Methods": [ "GET", "PUT", "DELETE" ],
    "Headers": [
      {
        "Name": "Authorization",
        "Values": [ "Bearer" ]
      }
    ]
  },
  "ClusterId": "ProductCluster"
}
```