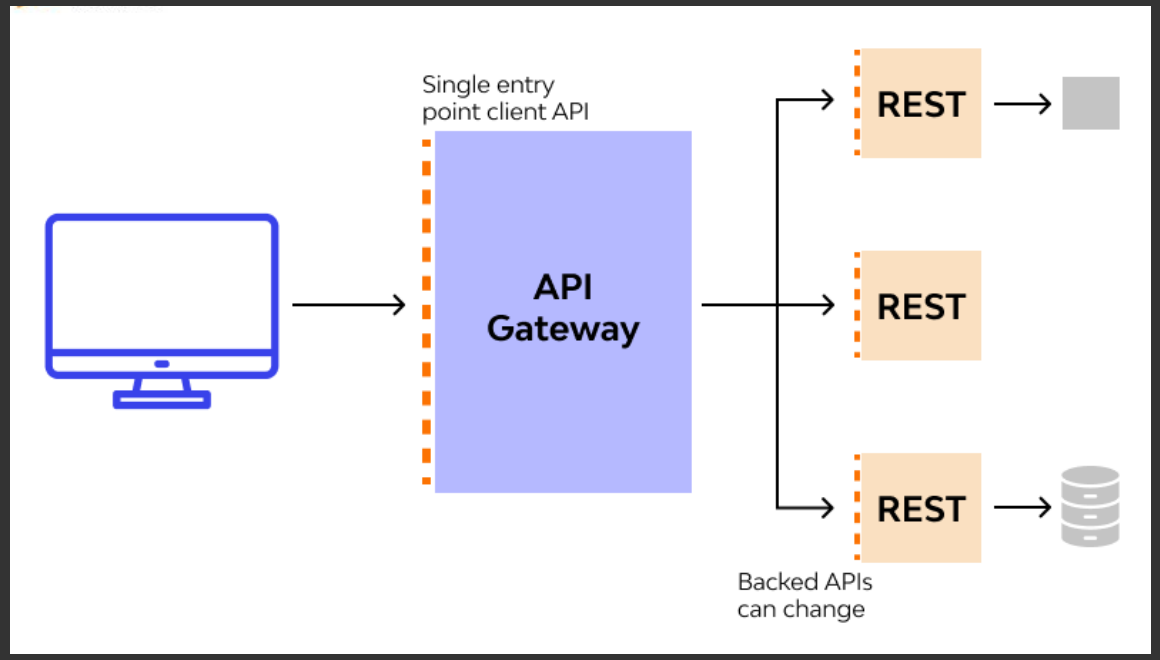
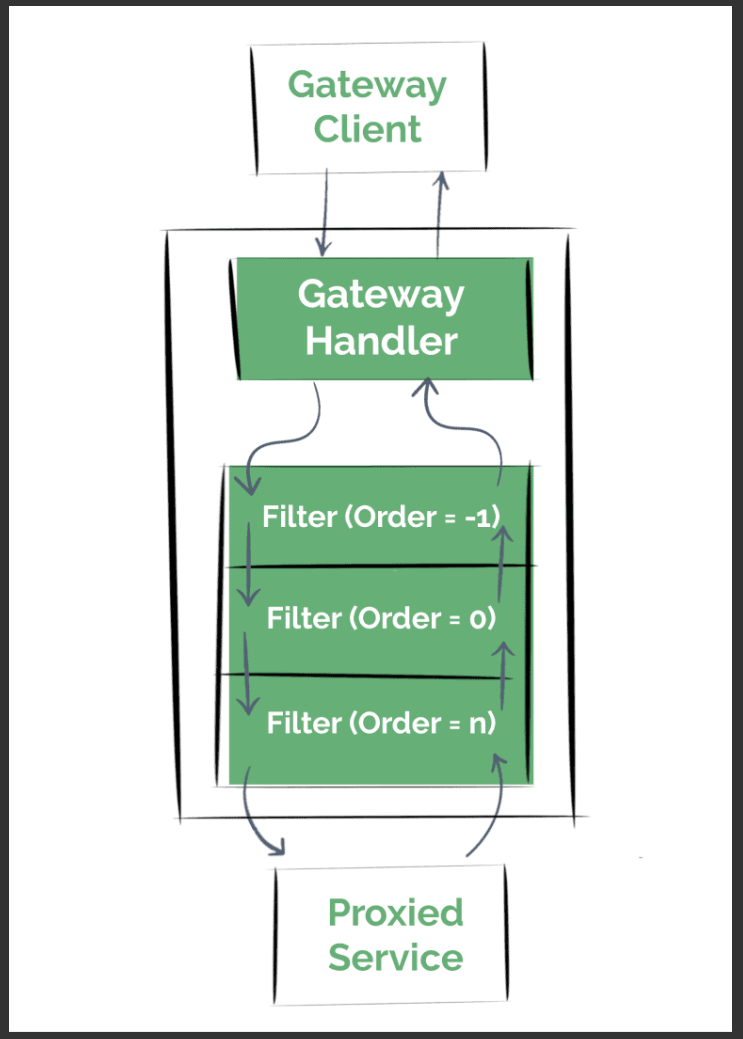
CODING SHUTTLE

Microservice

Introduction to the API Gateway Filters

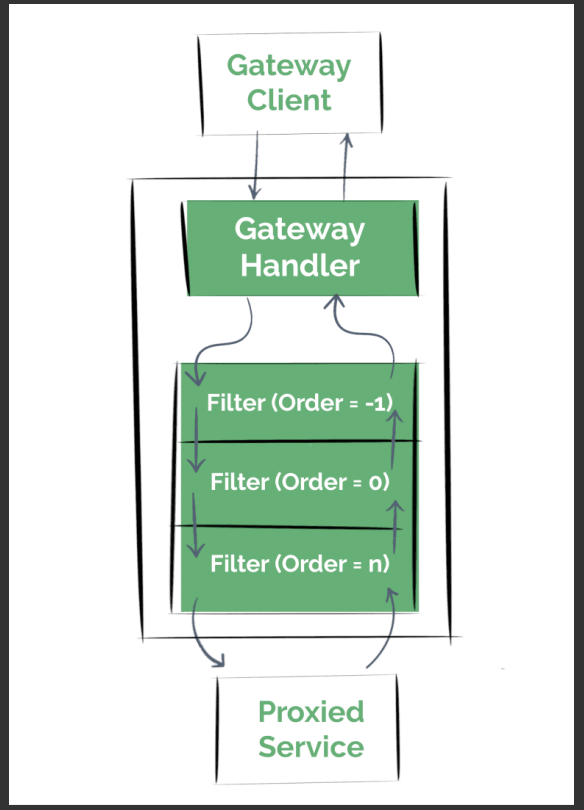




API Gateway Filters

API Gateway filters are used to intercept, modify, and enhance requests and responses that pass through an API Gateway. They allow you to apply common cross-cutting concerns (such as authentication, logging, rate limiting, and transformation) at a centralized entry point before routing requests to microservices. There are two types of filters:

1. Global Filters
2. Route specific Filters



Global Filter

```
@Component
public class FirstPreLastPostGlobalFilter
  implements GlobalFilter, Ordered {

    final Logger logger =
      LoggerFactory.getLogger(FirstPreLastPostGlobalFilter.class);

    @Override
    public Mono<Void> filter(ServerWebExchange exchange,
      GatewayFilterChain chain) {
        logger.info("First Pre Global Filter");
        return chain.filter(exchange)
          .then(Mono.fromRunnable(() -> {
              logger.info("Last Post Global Filter");
            }));
    }

    @Override
    public int getOrder() {
        return -1;
    }
}
```

All we have to do to create a custom global filter is to implement the Spring Cloud Gateway `GlobalFilter` interface, and add it to the context as a bean.

Route Specific Filters

Global filters are quite useful, but we often need to execute fine-grained custom Gateway filter operations that apply to only some routes.2. Route specific Filters.

To implement a `GatewayFilter`, we'll have to extend from the `AbstractGatewayFilterFactory` class provided by Spring Cloud Gateway.