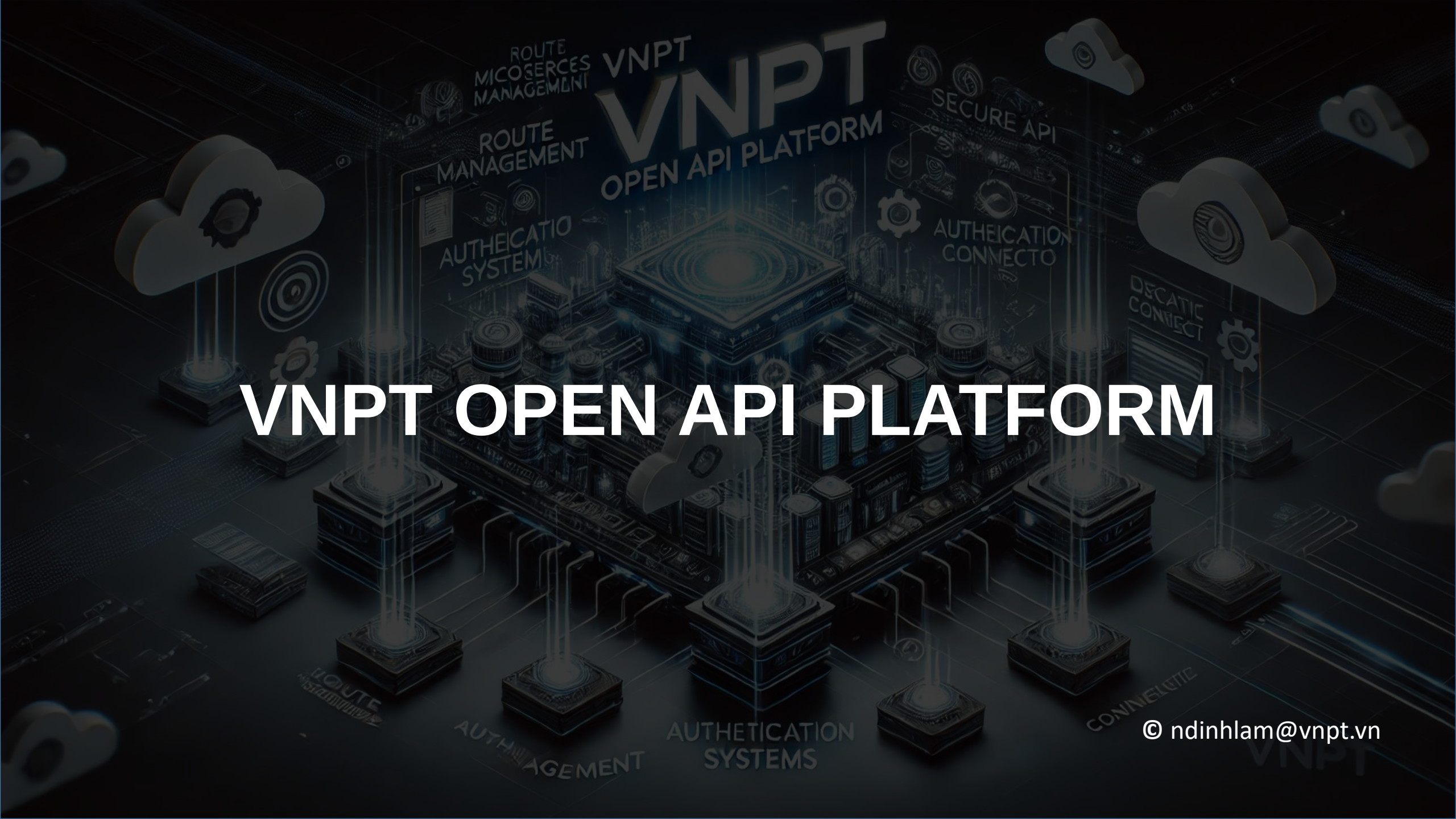




VNPT OPEN API PLATFORM

© ndinhlam@vnpt.vn

01

Giới thiệu về
VNPT Open API

02

Giải pháp công nghệ

03

Thiết kế hệ thống

04

API Gateway

05

Kết quả đạt được

06

Q & A

GIỚI THIỆU VỀ VNPT OPEN API

VNPT Open API là nền tảng quản lý, khai thác, phân phối các API dịch vụ, giúp cho các nhà phát triển dễ dàng đăng ký và sử dụng tài nguyên các API để phát triển dịch vụ.

- ✓ Cộng đồng CSP, Nhà phát triển, Xã hội hóa
- ✓ Phục vụ cho nhu cầu của đối tác triển khai dịch vụ đặc thù với VNPT
- ✓ Phục vụ cho nhu cầu tích hợp triển khai các dịch vụ nội bộ của VNPT

01

Publisher portal

Giao diện web dành cho nhà cung cấp API, là nơi cho phép nhà cung cấp đăng ký, quản lý các API.

02

Developer portal

Giao diện dành cho nhà phát triển, cung cấp môi trường để nhà phát triển đăng ký, nạp tiền, và sử dụng các API.

03

API Gateway

Quản lý và bảo mật API hiệu quả. Hỗ trợ xác thực, ủy quyền, định tuyến thông minh, cân bằng tải, và giới hạn lưu lượng (Rate Limiting, Quota) cho phép biến đổi dữ liệu...

VNPT OPEN API

Nhà phát triển API (API Developer):

Là các cá nhân hoặc tổ chức phát triển, tích hợp và sử dụng API của nhà cung cấp để xây dựng các ứng dụng và dịch vụ.

Người tiêu dùng API (API Consumer):

Là các doanh nghiệp, tổ chức hoặc người dùng cuối sử dụng ứng dụng hoặc dịch vụ do các nhà phát triển API cung cấp

Nhà cung cấp API (API Provider):

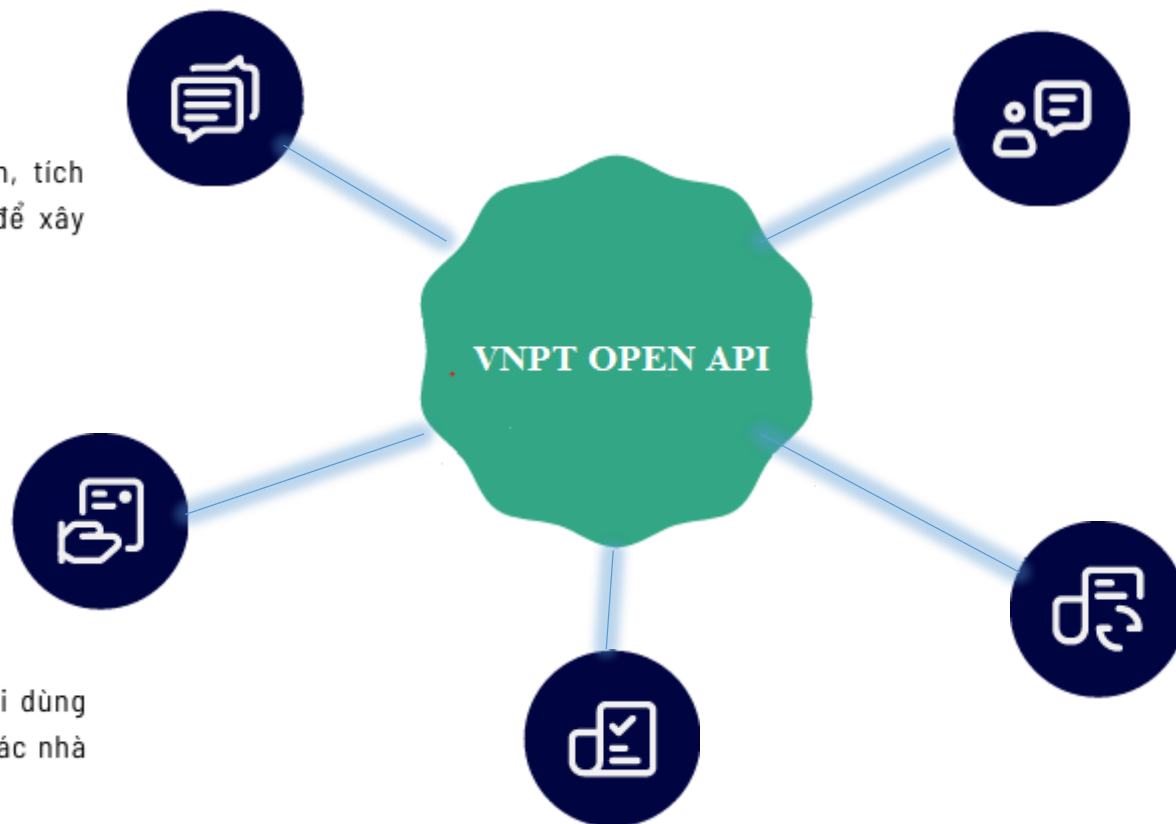
Là các nhà cung cấp dịch vụ API, thường là VNPT hoặc các doanh nghiệp có dịch vụ viễn thông, truyền thông và số hóa

Nhà quản lý nền tảng (Platform owner)

Là đơn vị xây dựng nền tảng và quản lý các APIs, nhà cung cấp APIs trong nền tảng

Đôi tác và bên thứ ba (Partners/Third-party Services):

Các doanh nghiệp đối tác hoặc bên thứ ba sử dụng hoặc tích hợp dịch vụ của Open Gateway vào hệ sinh thái của họ.

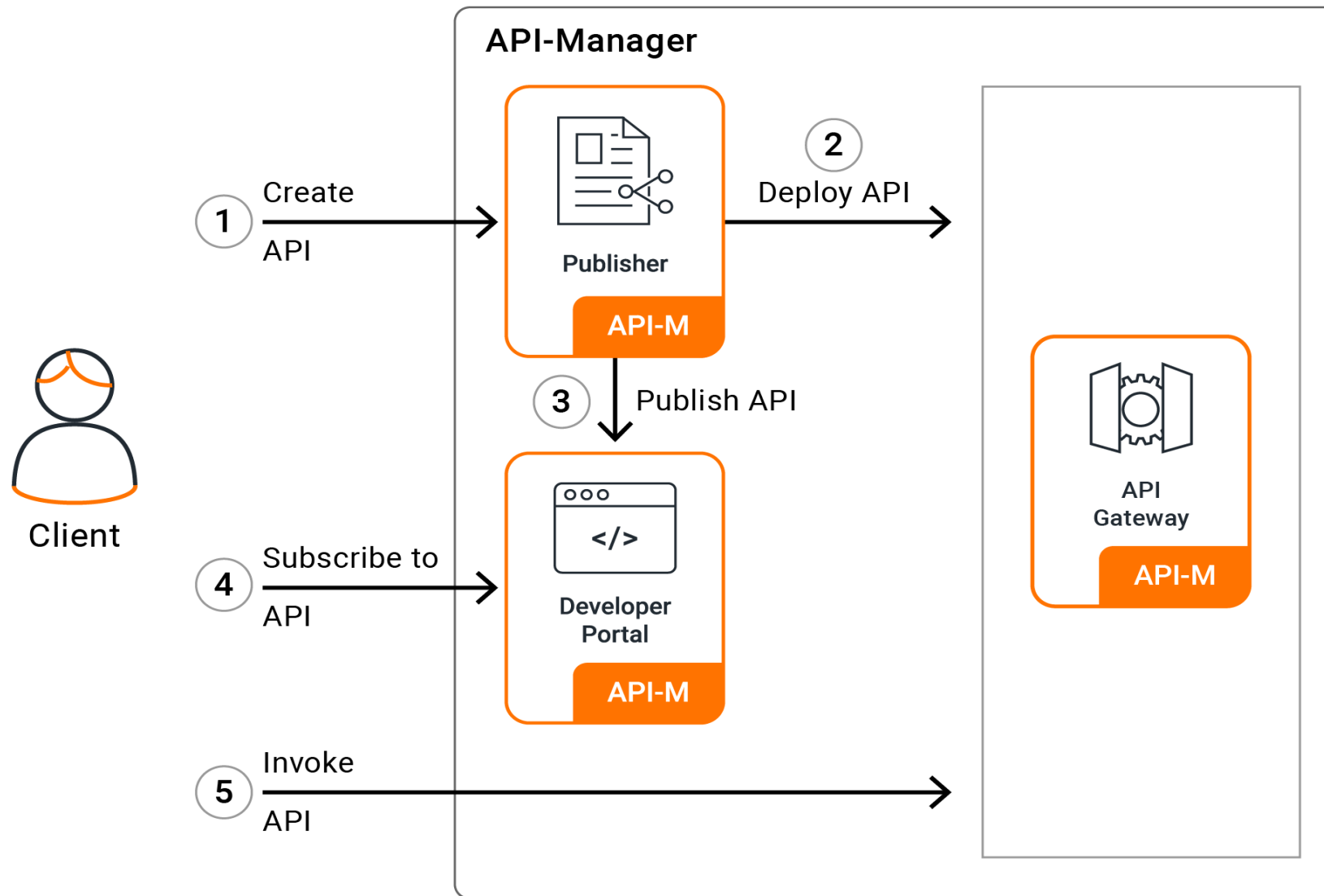




GIẢI PHÁP CÔNG NGHỆ

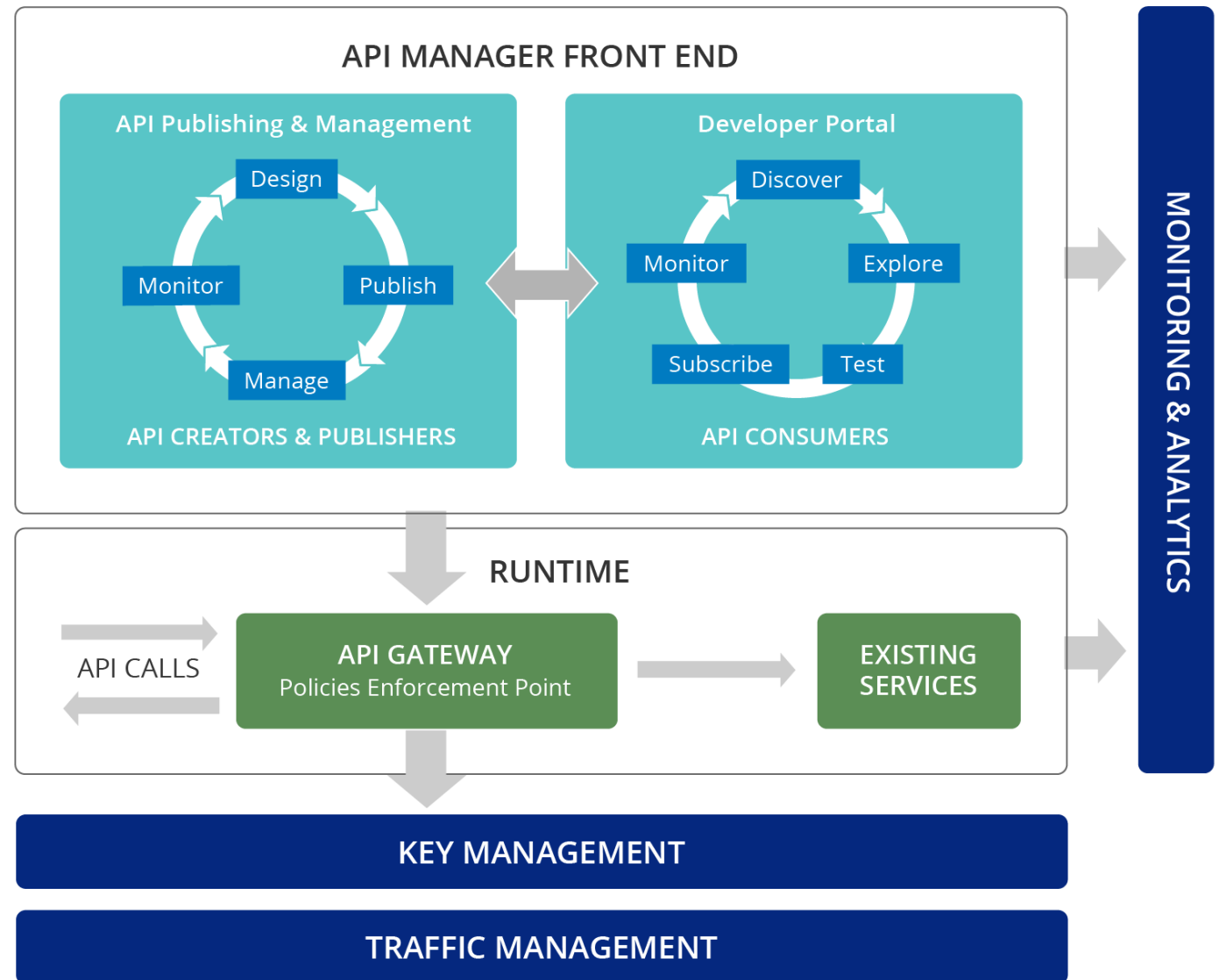


OPEN SOURCE WSO2 MANAGEMENT



Mô hình vận hành WSO2 API Management

- API Store (Developer Portal)
- API Publisher & Management
- API Gateway



ƯU ĐIỂM WSO2

- **Mã nguồn mở:** WSO2 là nền tảng mã nguồn mở, cho phép tùy chỉnh linh hoạt theo nhu cầu doanh nghiệp mà không tốn chi phí bản quyền.
- **Hệ sinh thái đầy đủ:** Bao gồm API Manager, Identity Server, Integration Server, và các công cụ khác, đáp ứng tốt các yêu cầu quản lý API và tích hợp hệ thống
- **Tích hợp mạnh mẽ:** Hỗ trợ các tiêu chuẩn như OAuth2, OpenID Connect, SOAP, REST và tích hợp tốt với các hệ thống bên thứ ba (Kafka, RabbitMQ, Databases).
- **Quản lý API hiệu quả:** Hỗ trợ đầy đủ các chức năng quản lý API như bảo mật, định tuyến, quản lý vòng đời, và phân tích hiệu suất.
- **Khả năng mở rộng và hiệu suất cao:** Có thể triển khai theo cụm (cluster) để đảm bảo khả năng mở rộng, sẵn sàng cao và xử lý lượng lớn yêu cầu.
- **Cộng đồng và tài liệu tốt:** Hỗ trợ tài liệu chi tiết và cộng đồng phát triển rộng lớn giúp giải quyết vấn đề nhanh chóng.

HẠN CHẾ WSO2

- **Giao diện trang API Developer và API Publisher (admin) còn chưa đẹp. Do dùng bộ themes WSO2 đang dùng, khó customize lại giao diện**
- **Chưa đáp ứng được các kịch bản charge phức tạp hơn, ví dụ:**
 - ✓ Khuyến mại khi gọi X request đầu tiên
 - ✓ Giá cước API thay đổi theo các bậc thang lưu lượng
- **Chưa khai báo API hoàn toàn bằng cấu hình được (Khi các API nguồn yêu cầu những tham số phức tạp, ví dụ trans_id phải tự sinh ngẫu nhiên, hoặc băm hash theo key thì vẫn phải code API kết nối trung gian)**
- **Mặc dù mã nguồn mở, các gói hỗ trợ doanh nghiệp của WSO2 khá tốn kém nếu cần hỗ trợ kỹ thuật chuyên sâu**



Spring Cloud Gateway

là một giải pháp API Gateway mạnh mẽ và linh hoạt trong hệ sinh thái Spring Cloud, được thiết kế để định tuyến các yêu cầu API và cung cấp các tính năng quản lý lưu lượng. Được phát triển dựa trên Spring Framework 5 và Project Reactor, giúp tận dụng lập trình phản ứng (Reactive Programming) để xử lý lưu lượng lớn một cách hiệu quả.

01

Định tuyến thông minh

Cho phép định tuyến yêu cầu từ client đến các dịch vụ backend dựa trên URL, cho phép cấu hình động.

02

Filter xử lý request & response

Hỗ trợ bộ lọc trước và sau để biến đổi dữ liệu yêu cầu/phản hồi, xác thực, hoặc thêm logic tùy chỉnh

03

Tích hợp Spring Ecosystem

Kết hợp dễ dàng với các công cụ như spring security, spring cloud config, các hệ thống discovery như Eureka

04

Hỗ trợ lập trình phản ứng

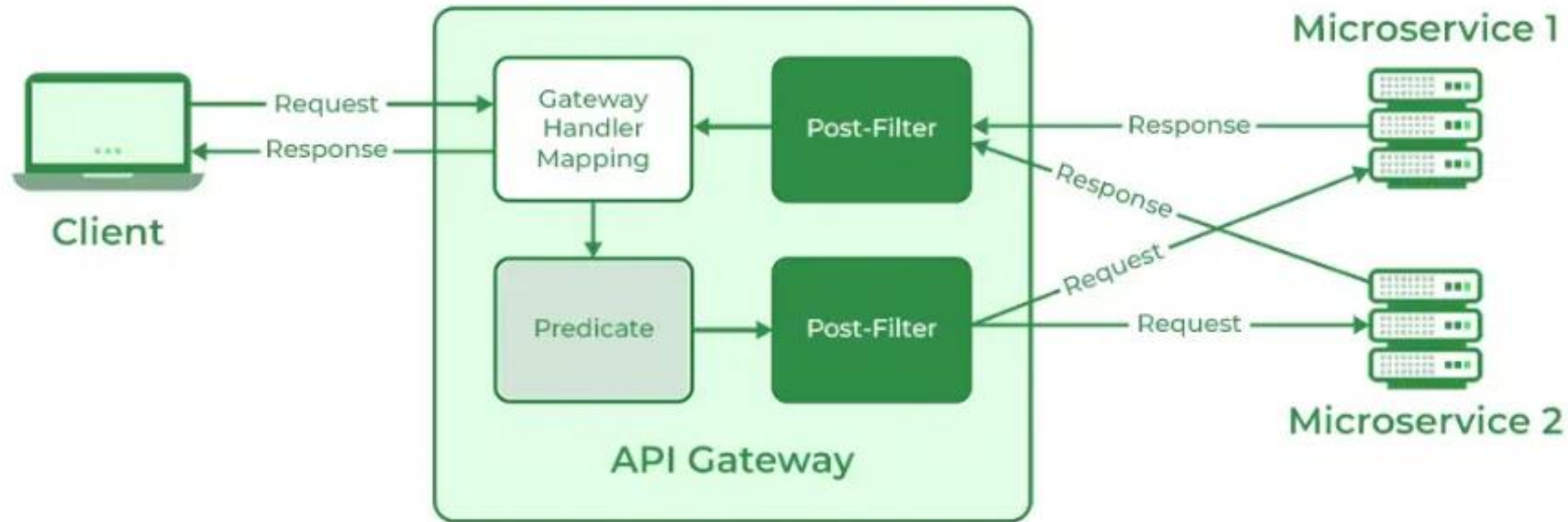
Xây dựng trên Project Reactor, giúp xử lý bất đồng bộ, tối ưu hóa hiệu suất trong môi trường tải cao

05

Tùy chỉnh dễ dàng

Dễ dàng mở rộng với các bộ lọc và logic tùy chỉnh

SPRING CLOUD GATEWAY ARCHITECTURE



01

Route

Là một thành phần quan trọng trong Spring Cloud Gateway. Nó bao gồm ID, URI đích, tiêu chí khớp (predicates), và bộ lọc (filters)

02

Predicate

Tương tự như hàm Predicate trong Java 8. Predicate được sử dụng để khớp các yêu cầu HTTP. Một route được khớp nếu Predicate trả về true.

03

Filter Chain

Là một loạt các bộ lọc được áp dụng cho các yêu cầu và phản hồi đến. Nó có thể được sử dụng cho nhiều mục đích khác nhau như xác thực, chuyển đổi yêu cầu hoặc phản hồi, và nhiều mục đích khác



THIẾT KẾ HỆ THỐNG



VNPT OPEN API



Quản lý truy cập (Access control)

Kiểm soát và quản lý truy cập hệ thống thông qua các phương thức xác thực như Token, Username and Password



Bảo mật và quyền riêng tư (Security and Privacy)

Đảm bảo bảo mật dữ liệu và quyền riêng tư cho người dùng



Ma trận phân quyền (Permission Matric)

Phân quyền người dùng dựa trên các chức năng và quyền hạn của User



Quản lý Nhà Cung Cấp APIs (API's Provider Management)

Quản lý và giám sát các APIs được cung cấp, bao gồm các tính năng như kiểm tra, bảo trì và cập nhật APIs.



Quản lý APIs (API management)

Quản lý các nhà cung cấp APIs, điều phối và theo dõi quá trình tích hợp từ các nhà cung cấp khác.



Quản lý ghi chép giao dịch (Transaction Record System)

Ghi lại và theo dõi các giao dịch hoặc yêu cầu API để đảm bảo tính minh bạch và phục vụ cho việc kiểm toán.



Mi hình tính phí (Business Model)

Định nghĩa các mô hình tính phí khi sử dụng APIs, như pay-per-use hoặc các gói thuê bao dịch vụ.

API Provider



SMS API



Data API



Topup API



Payment API

...

VNPT OPEN API



Develop Portal



Web Consumer



Admin CMS



Internal Api Gateway



Load balance



Third-party Api Gateway



Config Services



Eureka Service



Auth Service



Billing Service



Logging Service



Product Services



Report Services



Cron job Service



Redis



Oracle



Kafka

Actor



Developer



Admin CMS

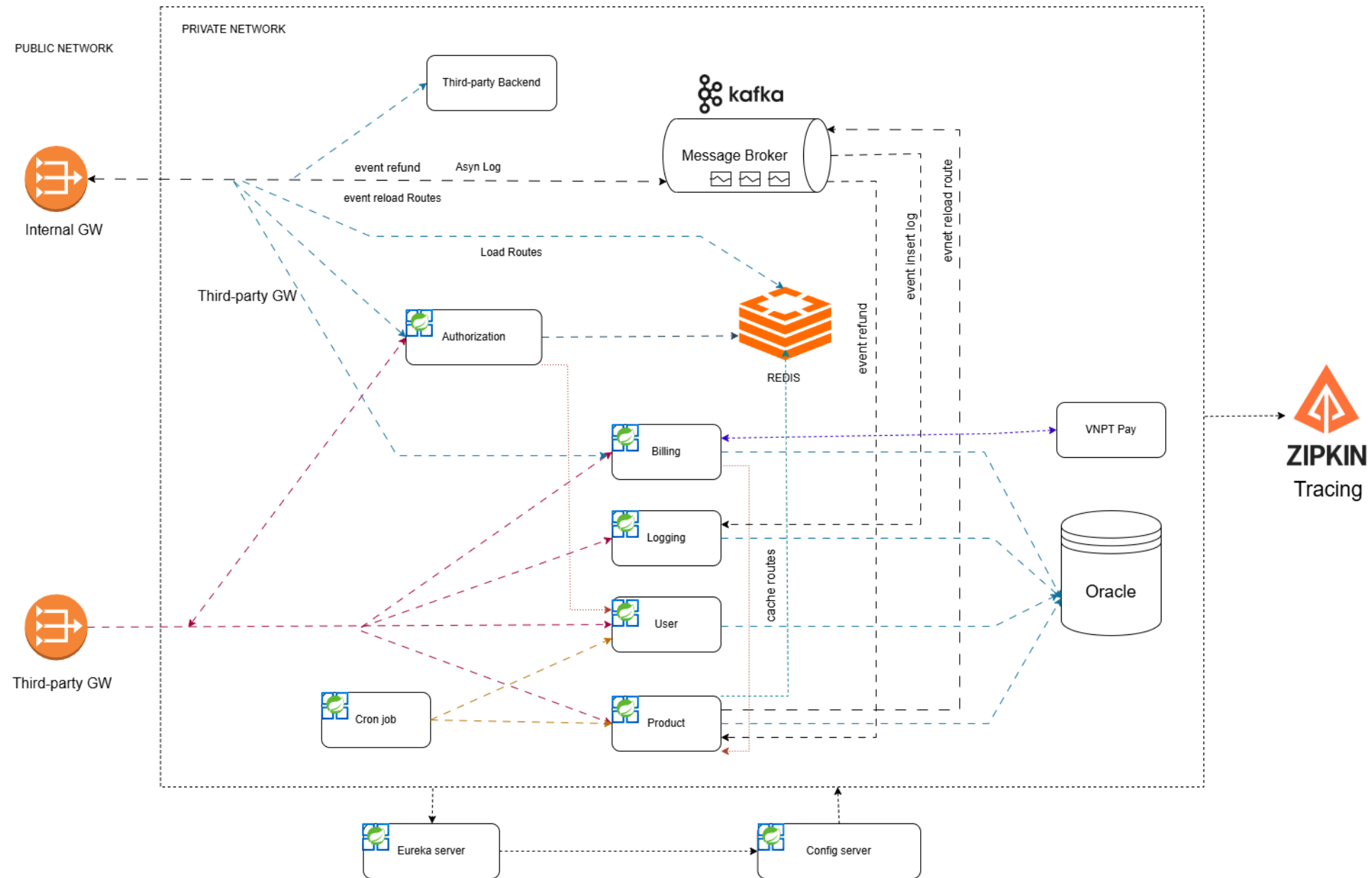


Publisher



CSKH

VNPT OPEN API ARCHITECTURE



DATABASE

TELECOM_APLAPI_LOG

P * ID	NVARCHAR2 (1000)
USER_ID	NUMBER
API_ID	NUMBER
INPUT	CLOB
OUTPUT	CLOB
CREATED_AT	DATE
STATUS	NVARCHAR2 (1000)
TRACE_ID	NVARCHAR2 (1000)
START_TIME	DATE
END_TIME	DATE
BILLING_TRANSACTION_ID	NUMBER
CLIENT_ID	VARCHAR2 (100 BYTE)
USER_EMAIL	VARCHAR2 (500 BYTE)
API_PATH	VARCHAR2 (500 BYTE)
API_ENV	VARCHAR2 (20 BYTE)

API_LOG_PK (ID)

API_LOG_PK (ID)

TELECOM_APLAPPLICATION_CLIENT_SECRET

U USER_ID	NUMBER
U ENVIROMENT	NVARCHAR2 (20)
CREATED_AT	DATE
P * CLIENT_ID	NVARCHAR2 (50)
CLIENT_SECRET	VARCHAR2 (1000 BYTE)
U API_ID	NUMBER

APPLICATION_CLIENT_PK (CLIENT_ID)

APPLICATION_CLIENT_SECRET_UK1 (ENVIROMENT, USER_ID, API_ID)

APPLICATION_CLIENT_PK (CLIENT_ID)

APPLICATION_CLIENT_SECRET_UK1 (ENVIROMENT, USER_ID, API_ID)

TELECOM_APLAPI_MODIFY_BODY_REQUEST

P * API_ID	NUMBER
BODY_SCRIPTS	CLOB
FILE_NAME	VARCHAR2 (1000 BYTE)
CREATED_AT	DATE
STATUS	NUMBER

API_MODIFY_FILTER_PK (API_ID)

API_MODIFY_FILTER_PK (API_ID)

TELECOM_APLAPI_MODIFY_BODY_RESPONSE

P * API_ID	NUMBER
BODY_SCRIPTS	CLOB
FILE_NAME	VARCHAR2 (1000 BYTE)
CREATED_AT	DATE
STATUS	NUMBER

API_MODIFY_BODY_RESPONSE_PK (API_ID)

API_MODIFY_BODY_RESPONSE_PK (API_ID)

TELECOM_APLAPI_MANAGEMENT

P * ID	NUMBER
NAME	VARCHAR2 (100 BYTE)
DESCRIPTION	VARCHAR2 (500 BYTE)
U CONTEXT	VARCHAR2 (20 BYTE)
VERSION	VARCHAR2 (20 BYTE)
PRODUCT_PATH	VARCHAR2 (1000 BYTE)
PRODUCT_ROOT	VARCHAR2 (1000 BYTE)
SANBOX_PATH	VARCHAR2 (1000 BYTE)
SANBOX_ROOT	VARCHAR2 (1000 BYTE)
AUTH_HEADER_KEY	VARCHAR2 (100 BYTE)
AUTH_HEADER_VALUE	VARCHAR2 (1000 BYTE)
METHOD	VARCHAR2 (20 BYTE)
TPS	NUMBER
SWAGGER_URL	VARCHAR2 (1000 BYTE)
PRICE_PER_REQUEST	NUMBER
STATUS	NUMBER
CREATED_AT	DATE
UPDATED_AT	DATE
OWNER	NUMBER
SANBOX_AUTH_HEAEDER_KEY	VARCHAR2 (1000 BYTE)
SANBOX_AUTH_HEAEDER_VALUE	VARCHAR2 (1000 BYTE)
API_DOCS	CLOB

API_MANAGEMENT_PK (ID)

API_MANAGEMENT_UK1 (CONTEXT)

API_MANAGEMENT_PK (ID)

API_MANAGEMENT_UK1 (CONTEXT)

TELECOM_APLBALANCE

P * USERNAME	VARCHAR2 (20 BYTE)
AMOUNT	NUMBER

BILLING_PK (USERNAME)

BILLING_PK (USERNAME)

TELECOM_APLPAY_TRANSACTION_INIT

P * ID	NUMBER
USERS_ID	NUMBER
MERCHANT_ORDER_ID	VARCHAR2 (200 BYTE)
STATUS	NUMBER (2)
CREATE_DATE	DATE

INIT_VNPTPAY_PK (ID)

INIT_VNPTPAY_PK (ID)

TELECOM_APLPAY_TRANSACTION_HIS

P * ID	NUMBER
USERS_ID	NUMBER
TYPE	NUMBER (2)
AMOUNT	NUMBER
CREATE_DATE	DATE
MERCHANT_ORDER_ID	VARCHAR2 (200 BYTE)
STATUS	NUMBER (2)
API_ID	NUMBER

PAY_TRANSACTION_HIS_PK (ID)

PAY_TRANSACTION_HIS_PK (ID)

TELECOM_APLUSERS

* EMAIL	VARCHAR2 (1000 BYTE)
PASSWORD	VARCHAR2 (1000 BYTE)
CREATED_AT	DATE
ROLE	VARCHAR2 (1000 BYTE)
P * ID	VARCHAR2 (20 BYTE)
STATUS	NUMBER

USERS_PK (ID)

USERS_EMAIL_UNIQUE (EMAIL)

USERS_PK (ID)

TELECOM_APLCUSTOMERS

P * ID	NUMBER
FULL_NAME	VARCHAR2 (1000 BYTE)
ADDRESS	VARCHAR2 (1000 BYTE)
DOB	DATE
AVATAR	VARCHAR2 (1000 BYTE)
USER_ACTION	NUMBER
PHONE	VARCHAR2 (20 BYTE)

CONSUMERS_PK (ID)

CONSUMERS_PK (ID)

TELECOM_APLROLES

P * ID	NUMBER
ROLE_NAME	VARCHAR2 (1000 BYTE)
PATH	VARCHAR2 (1000 BYTE)
PRIORITY	NUMBER
CREATED_AT	DATE
DESCRIPTION	VARCHAR2 (1000 BYTE)
USER_ACTION	NUMBER (*,0)
URI	VARCHAR2 (1000 BYTE)
METHOD	VARCHAR2 (20 BYTE)

ROLES_PK (ID)

ROLES_PK (ID)

TELECOM_APLMENUS

P * ID	NUMBER (19)
CREATED_AT	TIMESTAMP
ICON	VARCHAR2 (255 CHAR)
KEY	VARCHAR2 (255 CHAR)
PARENT_ID	NUMBER (19)
PATH	VARCHAR2 (255 CHAR)
PRIORITY	NUMBER (10)
STATUS	NUMBER (19)
TITLE	VARCHAR2 (255 CHAR)
TYPE	VARCHAR2 (255 CHAR)
USER_ACTION	NUMBER (19)

MENUS_PK (ID)

TELECOM_APLGROUP_MENUS

P * ID	NUMBER (19)
CREATED_AT	TIMESTAMP
GROUP_ID	NUMBER (19)
MENU_ID	NUMBER (19)
USER_ACTION	NUMBER (19)

GROUP_MENUS_PK (ID)

TELECOM_APLGROUPS

P * ID	NUMBER
NAME	VARCHAR2 (1000 BYTE)
CREATED_AT	DATE
STATUS	NUMBER
DESCRIPTION	VARCHAR2 (255 CHAR)
PRIORITY	NUMBER (10)
USER_ACTION	NUMBER

TABLE1_PK (ID)

TABLE1_PK (ID)

TELECOM_APLUSER_GROUPS

P * ID	NUMBER
USER_ID	NUMBER
GROUP_ID	NUMBER
CREATED_AT	DATE
USER_ACTION	NUMBER

USER_GROUP_PK (ID)

USER_GROUP_PK (ID)

TELECOM_APLGROUP_ROLES

P * ID	NUMBER
GROUP_ID	NUMBER
ROLE_ID	NUMBER
CREATED_AT	DATE
USER_ACTION	NUMBER

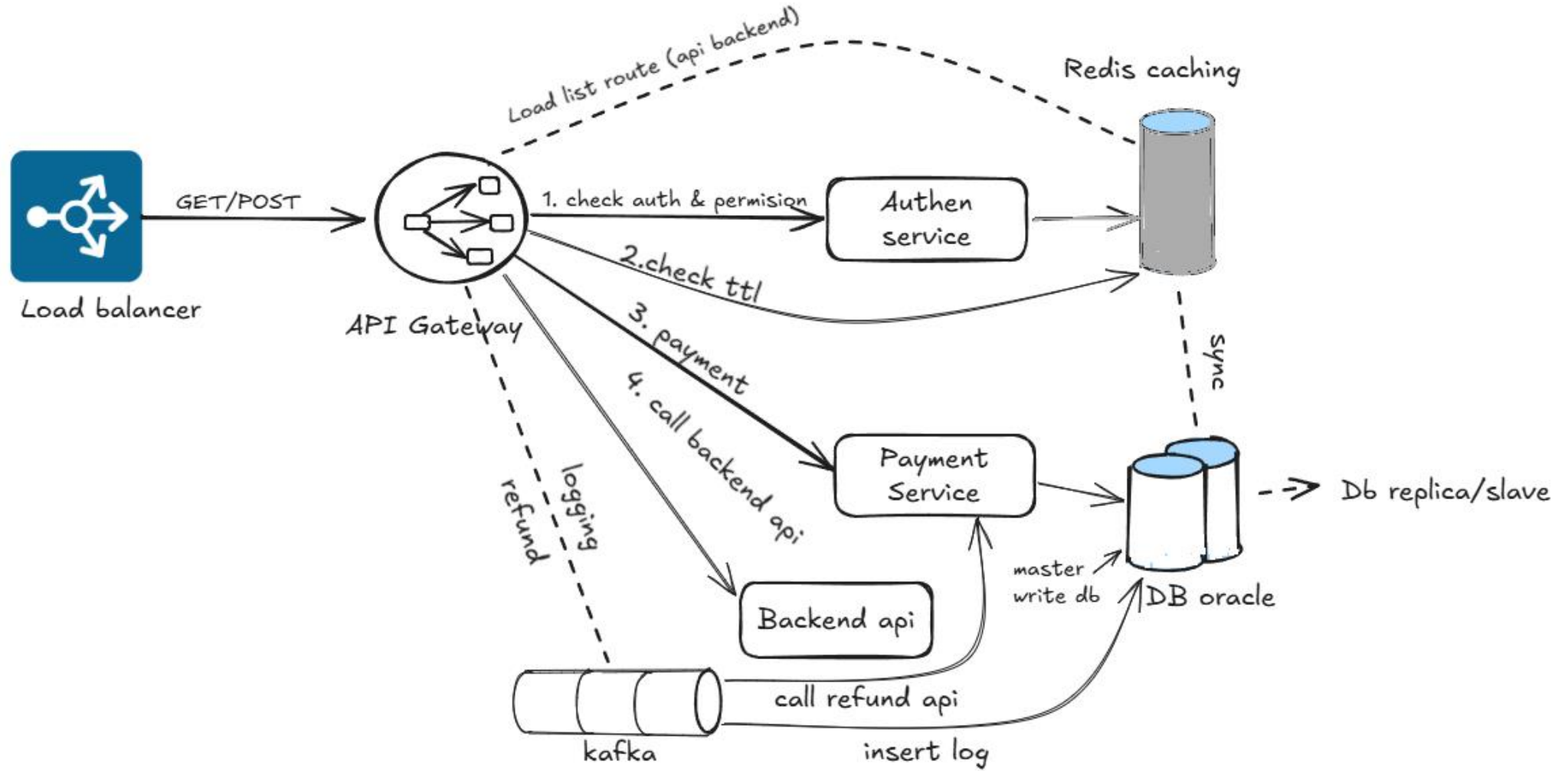
GROUP_ROLE_PK (ID)

GROUP_ROLE_PK (ID)

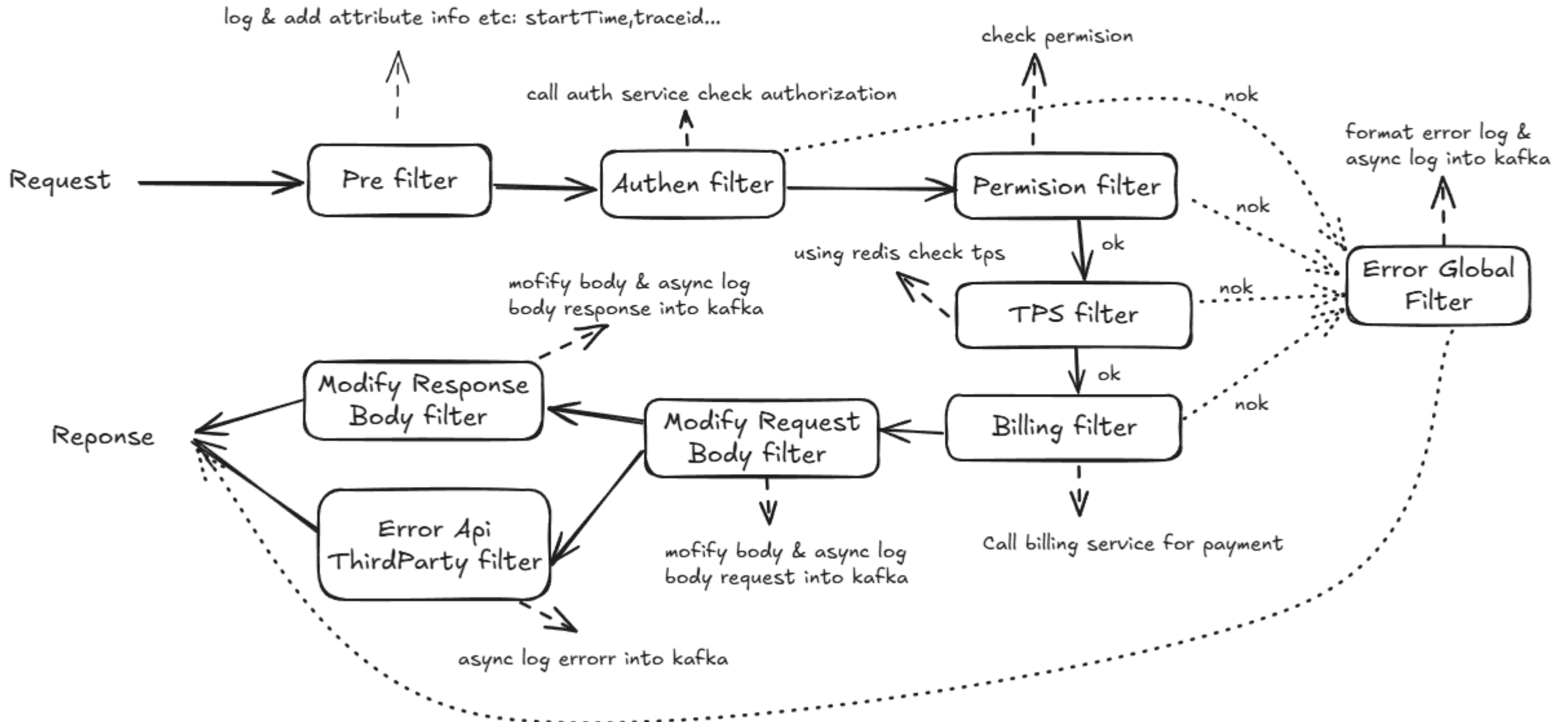
API GATEWAY



GATEWAY FLOW



GATEWAY FILTER



ADD FILTER

```
@Override
public Flux<Route> getRoutes() {
    RouteLocatorBuilder.Builder routesBuilder = routeLocatorBuilder.routes();
    return Flux.fromIterable(routeService.getAllRoutesFromRedis()) Flux<ApiRouteInfo>
        .map(apiRoute -> routesBuilder.route(apiRoute.getId(),
            predicateSpec -> setPredicateSpec(apiRoute, predicateSpec))) Flux<RouteLocatorBuilder.Builder>
        .collectList() Mono<List<RouteLocatorBuilder.Builder>>
        .flatMapMany(routes -> routesBuilder.build().getRoutes());
}

private Buildable<Route> setPredicateSpec(ApiRouteInfo apiRoute, PredicateSpec predicateSpec) {
    BooleanSpec booleanSpec = predicateSpec.path(apiRoute.getPath());
    if (!StringUtils.isEmpty(apiRoute.getMethod())) {
        booleanSpec.and()
            .method(apiRoute.getMethod());
    }
    booleanSpec.filters(f -> {
        f.setPath(apiRoute.getBackendPath()).
            filter(new RoleFilter(authenticationService)) GatewayFilterSpec
            .filter(new TPSFilter(tpsService))
            .filter(new BillingFilter(billingService))
            .filter(new ErrorAPIBackendFilter(kafkaEventProducer, objectMapper))
            .filter(new ResponseGlobalFilter(kafkaEventProducer, objectMapper))
            .modifyRequestBody(String.class, String.class,
                (exchange, originalRequestBody) -> requestResponseModifierService.modifyRequestBody(originalRequestBody, exchange))
            .modifyResponseBody(String.class, String.class,
                (exchange, originalResponseBody) -> requestResponseModifierService.modifyResponseBody(originalResponseBody, exchange))
            .metadata(Constants.API_INFO_ATTRIBUTE.API_TPS, apiRoute.getTps()) UriSpec
            .metadata(RESPONSE_TIMEOUT_ATTR, 30000)
            .metadata(CONNECT_TIMEOUT_ATTR, 30000);

        return f;
    });
    return booleanSpec.uri(apiRoute.getUri());
}
```

MODIFY REQUEST FILTER

```
@Override
public Mono<String> modifyRequestBody(String originalRequestBody, ServerWebExchange exchange) {
    try {
        Long timeStart = exchange.getAttributes().containsKey("time-start") ? Long.parseLong((String) exchange.getAttributes().get("time-start")) : 0;
        backendLogger.info("FILTER-----modifyRequestBody-----::" + (System.currentTimeMillis() - timeStart));
        exchange.getAttributes().put(Constants.API_INFO_ATTRIBUTE.API_LOG_STAGE, "insert");
        ApiLog apiLog = ResponseUtil.getAPIInfo(exchange);
        backendLogger.info(apiLog.getTraceId() + "::BACKEND-BODY::" + originalRequestBody);
        if(!Strings.isEmpty(originalRequestBody)) {
            String fileModifyRequest = null;
            if (MemoryCache.listModifyBodyRequest.containsKey(apiLog.getApiId())) {
                fileModifyRequest = MemoryCache.listModifyBodyRequest.get(apiLog.getApiId());
            } else {
                if (stringRedisTemplate.opsForHash().hasKey(Constants.REDIS_KEY.CACHE_APIS_MODIFY_REQUEST, apiLog.getApiId())) {
                    fileModifyRequest = stringRedisTemplate.opsForHash().get(Constants.REDIS_KEY.CACHE_APIS_MODIFY_REQUEST, apiLog.getApiId()).toString();
                    MemoryCache.listModifyBodyRequest.put(apiLog.getApiId(), fileModifyRequest);
                }
            }
            if (fileModifyRequest != null) {
                String fileScripts = folderGroovyRoot + apiLog.getApiId() + "/request/" + fileModifyRequest;
                Binding binding = new Binding();
                binding.setVariable("input", originalRequestBody);
                originalRequestBody = GroovyShellExecutor.shell(binding, "apild:" + apiLog.getApiId() + ":request", fileScripts);
                backendLogger.info(apiLog.getTraceId() + "::BACKEND-BODY-MODIFY::" + originalRequestBody);
            }
        }
        apiLog.setInput(originalRequestBody);
        String valueOfApiLog = objectMapper.writeValueAsString(apiLog);
        KafkaEventInfo apiLogKafkaEventInfo = new KafkaEventInfo();
        apiLogKafkaEventInfo.setEventName(KafkaTopicName.INSERT_API_LOG_EVENTS);
        apiLogKafkaEventInfo.setId(apiLog.getTraceId());
        apiLogKafkaEventInfo.setObjectJson(valueOfApiLog);
        kafkaEventProducer.sendEvent(apiLogKafkaEventInfo);
        return Mono.justOrEmpty(originalRequestBody);
    } catch (Exception e) {
        backendLogger.error("Error processing request: ", e);
        return Mono.just("Error processing request: " + e.getLocalizedMessage());
    }
}
```

MODIFY RESPONSE FILTER

```
@Override
public Mono<String> modifyResponseBody(String originalResponseBody, ServerWebExchange exchange) {
    Long timeStart = exchange.getAttributes().containsKey("time-start") ? Long.parseLong((String) exchange.getAttributes().get("time-start")) : 0;
    backendLogger.info("FILTER-----modifyResponseBody-----: "+(System.currentTimeMillis()-timeStart));
    if (exchange.getResponse().getStatusCode() != null &&
        exchange.getResponse().getStatusCode().is2xxSuccessful()) {
        try {
            exchange.getResponse().getHeaders().put("Access-Control-Allow-Origin", List.of("*"));
            ApiLog apiLog = ResponseUtil.getAPIInfo(exchange);
            backendLogger.info(apiLog.getTraceId() + "::BACKEND-RESPONSE::" + originalResponseBody);
            if(exchange.getResponse().getStatusCode().is2xxSuccessful() && !Strings.isEmpty(originalResponseBody)) {
                String fileModifyResponse = null;
                if(MemoryCache.listModifyBodyResponse.containsKey(apiLog.getApiId())){
                    fileModifyResponse = MemoryCache.listModifyBodyResponse.get(apiLog.getApiId());
                }else{
                    if(stringRedisTemplate.opsForHash().hasKey(Constants.REDIS_KEY.CACHE_APIS_MODIFY_RESPONSE, apiLog.getApiId())){
                        fileModifyResponse = stringRedisTemplate.opsForHash().get(Constants.REDIS_KEY.CACHE_APIS_MODIFY_RESPONSE, apiLog.getApiId()).toString();
                        MemoryCache.listModifyBodyResponse.put(apiLog.getApiId(), fileModifyResponse);
                    }
                }
                if (fileModifyResponse != null) {
                    String fileScripts = folderGroovyRoot + apiLog.getApiId()+ "/response/" + fileModifyResponse;
                    Binding binding = new Binding();
                    binding.setVariable( name: "output", originalResponseBody);
                    originalResponseBody = GroovyShellExecutor.shell(binding, apiId: apiLog.getApiId()+":response", fileScripts);
                    backendLogger.info(apiLog.getTraceId() + "::BACKEND-RESPONSE-MODIFY::" + originalResponseBody);
                }
            }
            apiLog.setOutput(originalResponseBody);
            String valueOfApiLog = objectMapper.writeValueAsString(apiLog);
            KafkaEventInfo apiLogKafkaEventInfo = new KafkaEventInfo();
            apiLogKafkaEventInfo.setEventName(KafkaTopicName.INSERT_API_LOG_EVENTS);
            apiLogKafkaEventInfo.setId(apiLog.getTraceId());
            apiLogKafkaEventInfo.setObjectJson(valueOfApiLog);
            kafkaEventProducer.sendEvent(apiLogKafkaEventInfo);
            return Mono.justOrEmpty(originalResponseBody);
        } catch (Exception e) {
            return Mono.just("Error processing response: " + e.getLocalizedMessage());
        }
    }
    return Mono.justOrEmpty(originalResponseBody);
}
```

GROOVY SHELL EXECUTOR

```
public class GroovyShellExecutor {

    private static final Logger logger = LoggerFactory.getLogger(GroovyShellExecutor.class);
    private static final ConcurrentHashMap<String, LinkedList<Script>> mapScripts = new ConcurrentHashMap<>();
    private static final ConcurrentHashMap<String, AtomicInteger> mapTotal = new ConcurrentHashMap<>();
    private static final int maxScript = 50000;

    public static String shell(Binding binding, String apiId, String script) throws Exception {
        Script scriptShell = null;
        try {
            scriptShell = getScript(apiId, script);
            if (scriptShell == null) throw new Exception("Can not create groovy script");
            scriptShell.setBinding(binding);
            String obj = (String) scriptShell.run();
            return obj;
        } finally {
            releaseScript(apiId, scriptShell);
        }
    }

    private static synchronized boolean releaseScript(String script, Script scriptShell) {
        if (scriptShell == null) return false;
        LinkedList<Script> linkedList = mapScripts.get(script);
        if (linkedList == null) return false;
        linkedList.add(scriptShell);
        return true;
    }

    public static void clearScripts(String apiId){
        if (mapScripts.containsKey(apiId))
            mapScripts.get(apiId).clear();
        if (mapTotal.containsKey(apiId))
            mapTotal.get(apiId).set(0);
    }

    private static synchronized Script getScript(String apiId, String fullScript) throws Exception {
        LinkedList<Script> linkedList = mapScripts.get(apiId);
        if (linkedList == null)
            mapScripts.put(apiId, new LinkedList<>());

        Script ret = mapScripts.get(apiId).poll();
        if (ret != null) {
            return ret;
        }

        AtomicInteger total = mapTotal.get(apiId);
        if (total == null)
            mapTotal.put(apiId, new AtomicInteger(initialValue: 0));
        if (mapTotal.get(apiId).get() >= maxScript) {
            logger.error("Pool " + apiId + " reach max size:" + maxScript);
            return null;
        }

        File f = new File(fullScript);
        if (!f.exists()) {
            logger.warn("File script: " + fullScript + " not found");
            return null;
        }

        ret = new GroovyShell().parse(f);
        mapTotal.get(apiId).incrementAndGet();
        logger.info("Create groovy ID"+apiId+"| " + fullScript + " pool size:" +
            mapScripts.get(apiId).size() + ", total:" + mapTotal.get(apiId).get());

        int totalScripts = 0, totalScripts2 = 0;
        Enumeration<String> keySet = mapScripts.keys();
        while (keySet.hasMoreElements()) {
            String param = keySet.nextElement();
            totalScripts += mapScripts.get(param).size();
            totalScripts2 += mapTotal.get(param).get();
        }

        return ret;
    }
}
```

KẾT QUẢ ĐẠT ĐƯỢC

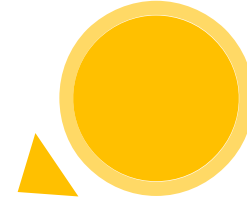


KẾT QUẢ ĐẠT ĐƯỢC

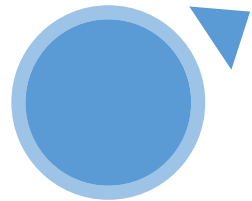
Giao diện thân thiện
người dùng



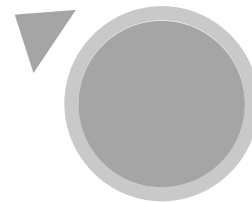
Đáp ứng được các
kịch bản charge phức tạp



Dễ dàng mở rộng



Hỗ trợ tùy biến request,
response đến các backend





WEB PUBLISHER

WEB PUBLISHER



Nguyễn Publisher

Trang chủ

PUBLISHER

Quản lý API

Tạo API

✓ Tạo API

✓ Tạo Swagger Document

✓ Code Modify Body Request

✓ Code Modify Body Response

Tên API

Test

Mô tả

Nhập mô tả

Context

M

Version

M

Thông tin Product

Test

Test

Header Key

Header Value

Thông tin Sandbox

Test

Test

Header Key

Header Value

WEB PUBLISHER



Nguyễn Publisher

Trang chủ

PUBLISHER

Quản lý API

Tạo API

Tạo API

Tạo Swagger Document

Code Modify Body Request

Code Modify Body Response

Upload

Main

Swagger Preview

Export Json

Upload

root: ""

Remove

Cập nhật và tiếp tục

Bỏ qua

- ✓ Tạo API
- ✓ Tạo Swagger Document
- ✓ Code Modify Body Request
- ✓ Code Modify Body Response

Đoạn code để biến đổi Body của Request

Code mẫu



```
import groovy.json.JsonSlurper
import groovy.json.JsonOutput

String input = binding.hasVariable("input")
? binding.getVariable("input") as String : ""

try {
    def newInput = new JsonSlurper().parseText(input)
    newInput.put("add_in", "121212")
    return JsonOutput.toJson(newInput)
} catch (Exception e) {
    return e.getMessage()
}
```

Nhập code

Lưu và tiếp tục

Bỏ qua



- ✓ Tạo API
- ✓ Tạo Swagger Document
- ✓ Code Modify Body Request
- ✓ Code Modify Body Response

Đoạn code để biến đổi Body của Response

Code mẫu



```
import groovy.json.JsonSlurper
import groovy.json.JsonOutput

String output = binding.hasVariable("output") ? binding.getVariable("output") as String : ""
try {
    def newOUTPUT = ["res":output]
    return JsonOutput.toJson(newOUTPUT)
} catch (Exception e) {
    return e.getMessage();
}
```

Nhập code

Lưu và hoàn tất

Hoàn tất



WEB DEVELOPER



Trang chủ

CONSUMER

Thông tin tài khoản

Danh sách API

API đang sử dụng

API Logging



Volodymyr Zelensky

Số điện thoại: 0822448211

Địa chỉ: Kiev, Ukraine122

Ngày sinh: 15/06/1999

Cập nhật



Volodymyr Zelensky

Tài khoản thanh toán



Tài khoản Hoạt Động

Số dư tài khoản: **₫1,022,077,417**

Nạp tiền

Lịch sử giao dịch

ID GIAO DỊCH	LOẠI GIAO DỊCH	NGÀY GIAO DỊCH	SỐ TIỀN	TRẠNG THÁI
197	Thanh toán	28/11/2024 09:11:08 AM	-₫5	Thành công
230	Thanh toán	29/11/2024 10:19:30 AM	-₫5	Thành công
241	Thanh toán	29/11/2024 10:30:21 AM	-₫5	Thành công
242	Thanh toán	29/11/2024 10:44:32 AM	-₫5	Thành công
243	Thanh toán	29/11/2024 10:45:09 AM	-₫5	Thành công
244	Thanh toán	29/11/2024 10:47:07 AM	-₫5	Thành công
245	Thanh toán	29/11/2024 10:47:10 AM	-₫5	Thành công
246	Thanh toán	29/11/2024 10:48:43 AM	-₫5	Thành công
247	Thanh toán	29/11/2024 10:48:46 AM	-₫5	Thành công
248	Thanh toán	29/11/2024 10:55:08 AM	-₫5	Thành công



Trang chủ

CONSUMER

Thông tin tài khoản

Danh sách API

API đang sử dụng

API Logging



Volodymyr Zelensky

Danh sách API

Tìm kiếm



Digishop 2

Tạo bởi: 12
Phiên bản: 1.0.0
Context: /digishop-web



API companies

Tạo bởi: 12
Phiên bản: 1.0.0
Context: /fake-json



API Digishop

Tạo bởi: 12
Phiên bản: 1.0.0
Context: /digishop-order



PetStore

Tạo bởi: 1240
Phiên bản: 1.0.0
Context: /test



test new api

Tạo bởi: 1200
Phiên bản: 1.0.0
Context: /test-contextt





Trang chủ

CONSUMER

Thông tin tài khoản

Danh sách API

API đang sử dụng

API Logging



Volodymyr Zelensky

Chi tiết

Tổng quan

Thông tin xác thực

API Console

Product keys

Cấu hình cho xác thực

Tạo Client ID và Client Secret

Thời gian hết hạn access token

3600

Thời gian hết hạn refresh token

84600

Tạo Access Token

Lệnh Curl Tạo Access Token



Trang chủ

CONSUMER

Thông tin tài khoản

Danh sách API

API đang sử dụng

API Logging



Volodymyr Zelensky

Chi tiết

Tổng quan

Thông tin xác thực

API Console

Key Type

☒ Production

Vui lòng nhập Access Token từ thông tin xác thực vào Authorize để sử dụng API

Servers

Authorize



API companies



GET

/companies





Trang chủ

CONSUMER

Thông tin tài khoản

Danh sách API

API đang sử dụng

API Logging



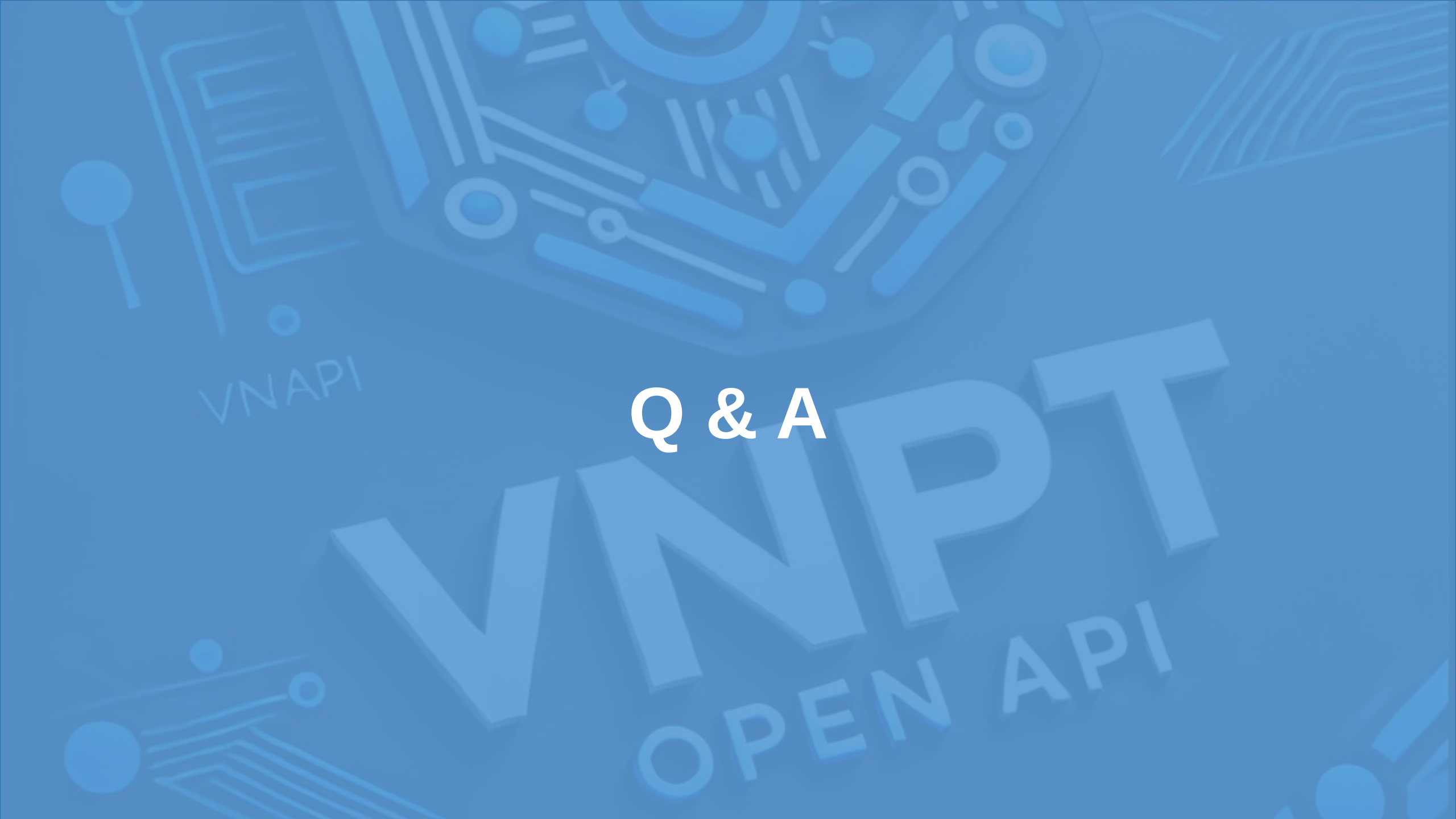
Volodymyr Zelensky

Lịch sử call API

Chọn khoảng thời gian



API ID	BẮT ĐẦU	KẾT THÚC	TRACE ID	MÔI TRƯỜNG	ĐƯỜNG DẪN	TRẠNG THÁI	SỐ TIỀN	TRẠNG THÁI THANH TOÁN
4	2024-11-28 09:11:08.0	2024-11-28 09:11:09.0	41668169-16	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:07:44.0	2024-11-29 10:07:44.0	b9a48be6-11	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:10:45.0	2024-11-29 10:10:46.0	400a1e96-2	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:11:01.0	2024-11-29 10:11:01.0	400a1e96-4	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:19:30.0	2024-11-29 10:19:31.0	e0f9dbe6-3	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:44:32.0	2024-11-29 10:44:33.0	daa9988c-3	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:45:09.0	2024-11-29 10:45:09.0	bcc1d4b6-5	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:47:07.0	2024-11-29 10:47:08.0	49e47931-1	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:47:10.0	2024-11-29 10:47:11.0	49e47931-2	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công
4	2024-11-29 10:48:43.0	2024-11-29 10:48:45.0	ec11b9e5-1	PRODUCT	/fake-json/1.0.0/companies	200 OK	-5 đ	Thành công



Q & A

VNPT
OPEN API



THANK YOU!