**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

**BÁO CÁO ĐỒ ÁN CUỐI KỲ**

**MÔN: MẬT MÃ HỌC**

**AUTHENTICATION AND AUTHORIZATION IN MICROSERVICE ARCHITECTURE**

**TP. HỒ CHÍ MINH, NĂM 2023**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

**BÁO CÁO ĐỒ ÁN CUỐI KỲ**

**MÔN: MẬT MÃ HỌC**

**AUTHENTICATION AND AUTHORIZATION IN MICROSERVICE ARCHITECTURE**

**GIẢNG VIÊN HƯỚNG DẪN**

**TS. NGUYỄN NGỌC TỰ**

**SINH VIÊN THỰC HIỆN**

**NGUYỄN NHẬT QUÂN - 21522497**

**PHẠM NGUYỄN HẢI ANH - 21520586**

**TP. HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Lời đầu tiên nhóm 1 xin gửi cảm ơn tới thầy Nguyễn Ngọc Tự, người đã tận tình giảng dạy, giúp đỡ nhóm trong suốt quá trình thực hiện đề tài. Mỗi bài giảng của thầy đều giúp nhóm chúng em hiểu biết hơn về những vấn đề xung quanh và từ đó có thể đưa ra những vận dụng thực tiễn trong cuộc sống. Cảm ơn những chia sẻ vô cùng quý báu từ thầy, nhờ đó mà nhóm đã có thêm nhiều kiến thức và kỹ năng cần thiết để thực hiện đồ án lần này.

Trong quá trình thực hiện đồ án, bằng những kiến thức và kỹ năng của mình, nhóm 1 đã rất cố gắng và nỗ lực hết mình để hoàn thành thật tốt đề tài đã chọn. Tuy nhiên, vì kinh nghiệm còn hạn chế nên nhóm không thể tránh khỏi những sai sót. Rất mong thầy thông cảm và chia sẻ những góp ý để nhóm có thể chỉnh sửa và hoàn thiện đề tài hơn trong tương lai. Chúc thầy luôn nhiều sức khỏe và tiếp tục chia sẻ thêm kiến thức bổ ích như vậy đến với các bạn sinh viên.

Nhóm 1 xin chân thành cảm ơn!

**Nhóm thực hiện.**

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC TỪ VIẾT TẮT

JWT: Json Web Token

OT: Opaque Token

TLS: Transport Layer Securty

# **TỔNG** QUAN ĐỀ TÀI

## Giới thiệu:

* Tên đề tài: Authentication and authorization in microservice architecture.
* Tên môn học: Mật mã học (NT219.N21.ANTN)
* Nhóm sinh viên thực hiện - Nhóm 1 bao gồm 2 thành viên:
* Nguyễn Nhật Quân - 21522497
* Phạm Nguyễn Hải Anh - 21520586
* Phân chia công việc:

|  |  |  |
| --- | --- | --- |
| STT | Nội dung công việc | Thành viên thực hiện |
| 1 | Lựa chọn đề tài, tìm kiếm bài báo tham khảo | Cả nhóm |
| 2 | Tìm hiểu về ngữ cảnh và JWT | Hải Anh |
| 3 | Tìm hiểu chung và code microservice, JWT, chứng chỉ | Nhật Quân |
| 4 | Góp ý thực hiện, deploy | Hải Anh |
| 5 | Làm slide thuyết trình | Hải Anh |
| 6 | Viết report cuối kỳ | Cả nhóm |

## Ngữ cảnh ứng dụng của đề tài

* Trong thời đại công nghệ hiện đại, việc phát triển các ứng dụng phần mềm phức tạp đang đặt ra nhiều thách thức đáng kể. Kiến trúc microservice đã nổi lên như một giải pháp linh hoạt và mạnh mẽ, giúp giải quyết các vấn đề này và mang đến sự linh hoạt, mở rộng và bảo trì dễ dàng hơn. Mô hình microservice architecture tạo ra một hệ thống linh hoạt và dễ mở rộng, cho phép các nhóm phát triển tập trung vào từng phần nhỏ của ứng dụng mà không ảnh hưởng đến các thành phần khác.



Hình 1: Minh họa cho microservice architecture

## Các bên liên quan

### Sơ đồ hệ thống ứng dụng microservice

A picture containing text, screenshot, diagram, font

Description automatically generated

Hình 2: Sơ đồ các bên liên quan trong mô hình microservice architecture

### Chi tiết các bên liên quan.

* Người dùng cuối (users): Truy cập tới server từ máy tính và điện thoại
* API Gateway: Chuyển các resquest của người dùng đến từng microservice phù hợp, trả response lại cho người dùng từ các microservices.
* Auth service: Xác thực người dùng qua username và password nhận được từ API Gateway sau đó trả về token để người dùng truy cập vào các tài nguyên của hệ thống.
* Task services: các microservices làm từng nhiệm vụ riêng biệt được người dùng yêu cầu qua token.
* Certificate authority (CA): bên thứ 3 cung cấp certificate cho hệ thống.
* Attacker.

## Các giả thuyết đặt ra và các yêu cầu về bảo mật

### Các giải thuyết đặt ra

* Server được đặt trên cloud, không nên tin tưởng hoàn toàn (semi-trusted).
* Kẻ tấn công có thể truy cập vào hệ thống và có thể can thiệp đường truyền.

### Các yêu cầu về bảo mật

* Xác thực và phân quyền: xác thực chính xác người dùng của hệ thống và phần quyền cho phép truy cập vào đúng tài nguyên của họ.
* Bảo mật đường truyền: bảo vệ dữ liệu được truyền đi bằng TLS.

## Các mối đe dọa

### Tấn công giả mạo (Spoofing attack)

* Tấn công qua các dịch vụ ủy quyền: Trong một môi trường microservice, các dịch vụ có thể sử dụng các phương pháp ủy quyền để xác thực và truy cập vào các tài nguyên khác. Kẻ tấn công có thể giả mạo một dịch vụ ủy quyền và gửi yêu cầu giả mạo để lừa đảo người dùng cung cấp thông tin xác thực hoặc token truy cập.
* Tấn công qua các API gateway: API gateway thường được sử dụng trong kiến trúc microservice để quản lý và bảo vệ các API. Kẻ tấn công có thể tạo ra một trang web giả mạo với mục đích lừa đảo người dùng truy cập vào một API giả mạo thông qua API gateway. Điều này có thể dẫn đến việc tiết lộ thông tin xác thực hoặc tấn công vào các dịch vụ phía sau API gateway.
* Tấn công thông qua hệ thống quản lý người dùng: Trong một môi trường microservice, việc quản lý và xác thực người dùng thường được thực hiện thông qua một hệ thống quản lý người dùng chung. Kẻ tấn công có thể tạo ra một trang đăng nhập giả mạo, với mục đích lừa đảo người dùng nhập thông tin đăng nhập của họ. Khi người dùng cung cấp thông tin đăng nhập, kẻ tấn công có thể sử dụng nó để truy cập trái phép vào các dịch vụ microservice.

### Man-in-the-middle attack (MITM)

* Tấn công giữa các microservice: Kẻ tấn công có thể can thiệp vào giao tiếp giữa các microservice. Điều này có thể dẫn đến việc chặn, thay đổi hoặc đánh cắp dữ liệu gửi giữa các microservice. Kẻ tấn công có thể sử dụng thông tin đánh cắp để thực hiện các hành động không an toàn hoặc tiếp tục tấn công vào hệ thống.
* Tấn công giữa microservice và người dùng: Kẻ tấn công có thể can thiệp vào quá trình giao tiếp giữa một microservice và người dùng cuối. Khi người dùng gửi yêu cầu đến microservice, kẻ tấn công có thể đọc, thay đổi hoặc chèn thêm thông tin vào yêu cầu và phản hồi. Điều này có thể dẫn đến việc lừa đảo người dùng, đánh cắp thông tin cá nhân hoặc thực hiện các hành động không an toàn.

# HƯỚNG NGHIÊN CỨU CHO ĐỒ ÁN

## Giải pháp đề tài

### Sử dụng JWT (dynamic token) để xác thực và phân quyền cho người dùng

* JWT là một phương tiện đại diện cho các yêu cầu chuyển giao giữa hai bên Client – Server, các thông tin trong chuỗi JWT được định dạng bằng JSON. Trong đó chuỗi Token phải có 3 phần là header, phần payload và phần signature được ngăn bằng dấu “.”



Hình 3: Thành phần của json web token

* Header gồm 2 thành phần chính:
  + typ - Loại token (mặc định là JWT - Thông tin này cho biết đây là một Token JWT).
  + alg - Thuật toán đã dùng để mã hóa (SHA256, SHA3-384…).
  + Sau đó header được encode bằng base64.

{

"alg": "HS256",

"typ": "JWT"

}

* Phần payload chứa các thông tin muốn đặt trong chuỗi token:
  + iss: thông tin về microservice tạo ra token.
  + id: user id của người dùng muốn tạo ra token.
  + exp: thời gian hết hạn của token (được đặt là sau 10 phút).
  + admin: quyền admin.
  + Sau đó payload được encode bằng base64.

A picture containing text, screenshot, font

Description automatically generated

Hình 4: Payload của JWT

* Signature: Phần chử ký này sẽ được tạo ra bằng cách mã hóa phần header, payload kèm theo một chuỗi secret (khóa bí mật).

data = base64Encode( header ) + "." + base64Encode( payload )

signature = Hash( data, secret );

* Sau khi Auth service tạo ra JWT (có hiệu lực trong 10 phút) sẽ cùng với API gateway giúp người dùng truy cập vào hệ thống.

A picture containing text, screenshot

Description automatically generated

Hình 5: Sơ đồ giải pháp cho microservice architecture

* API gateway: sau khi nhận được JWT từ Auth service, lập tức chuyển sang opaque token (OT) và lưu lại cặp OT và JWT để cho người dùng sử dụng và truy cập vào các tài nguyên của họ dựa trên user id sau khi validate.

### Sử dụng TLS để mã hóa đường truyền

* TLS (Transport Layer Security) – Bảo mật tầng vận chuyển. Đây là một giao thức mật mã cung cấp bảo mật đầu cuối cho dữ liệu được gửi giữa các ứng dụng qua Internet. TLS được biết đến chủ yếu thông qua việc sử dụng trong duyệt web an toàn với chuẩn HTTPS.

A diagram of a server

Description automatically generated with low confidence

Hình 6: Mô hình giao tiếp TLS

* Certificate được lấy từ Zerossl, với thuật toán ký là sha384ECDSA, thuật toán băm sha384.

Screens screenshot of a computer

Description automatically generated with medium confidence

Hình 7: Zerossl certificate

# NGỮ CẢNH ỨNG DỤNG VÀ TRIỂN KHAI THỬ NGHIỆM

## Kịch bản triển khai

### Chuẩn bị

* Platform triển khai container: Docker
* Thư viện sử dụng:
  + PyJWT: Thư viện của Python cho phép encode và decode JWT
* Môi trường của Server theo dạng Serverless:
  + Amzaon Linux phiên bản 2 (CentOS) của AWS EC2
* Hệ quản trị cơ sở dữ liệu: MySQL
* Các client:
  + 1 máy Windows (Wins 11)
    - Cài đặt công cụ curl của Linux
  + 1 máy Ubuntu (phiên bản 20.04 LTS)
  + 1 điện thoại (Iphone 11)
    - Cài đặt ứng dụng API Tester

### Kịch bản triển khai

* Client là những thiết bị sử dụng hệ điều hành Windows, Linux và IOS
* Mô hình triển khai các microservice trên docker:

A diagram of a service

Description automatically generated with low confidence

*Mô hình các microservice trên docker*

* + Gateway: chuyển tiếp các luồng dữ liệu giữa người dùng và các microservice liên quan, bao gồm auth, task service và database. Chuyển đổi giữa JWT và opaque token.

A screenshot of a computer screen

Description automatically generated with medium confidence

*Code gateway*

* + Auth service: có chức năng đăng ký người dùng mới, tạo JWT khi đăng nhập và quản lý chúng.

A screen shot of a computer screen

Description automatically generated with low confidence

A screen shot of a computer screen

Description automatically generated with low confidence

*Hình 8+9: Code auth service*

* + Task service: tạo ra các ghi chú

A screenshot of a computer screen

Description automatically generated with medium confidence

*Hình 10: code task service*

* + Database service: tạo ra mỗi database ứng với từng service để lưu lại thông tin tương ứng.

A screenshot of a computer

Description automatically generated

*Hình 11: sql script for database*

* Đóng gói các chương trình trên thành container rồi đẩy lên docker hub.

A screenshot of a computer

Description automatically generated with medium confidence

* Trên Server, kéo những image từ Docker hub. Tạo một mạng docker loại bridge bằng lệnh “docker network create --driver bridge --subnet 172.0.0.1/19 my-nw” và thực thi các lệnh của docker để chạy các container.

A computer screen with white text

Description automatically generated with low confidence

*Chạy container MySQL database*

A computer screen with white text

Description automatically generated with low confidence

*Chạy container task service*

A screen shot of a computer

Description automatically generated with medium confidence

*Chạy container gateway*

*A computer screen with white text

Description automatically generated with low confidence*

*Chạy container auth*

* Server thực thi Docker cho mô hình các microservice là 1 AWS EC2 instance. Server này được cấu hình để chạy cân bằng tải (load balancing) với replica=2 bằng dịch vụ AWS ELB. Con load balancer sẽ được gắn tên miền “mmhmicro.online” kèm với certificate đã ký được từ ZeroSSL.
  + Import chứng chỉ đã được ký cho tên miền “mmhmicro.online” vào AWS Certificate Manager.

A screenshot of a computer

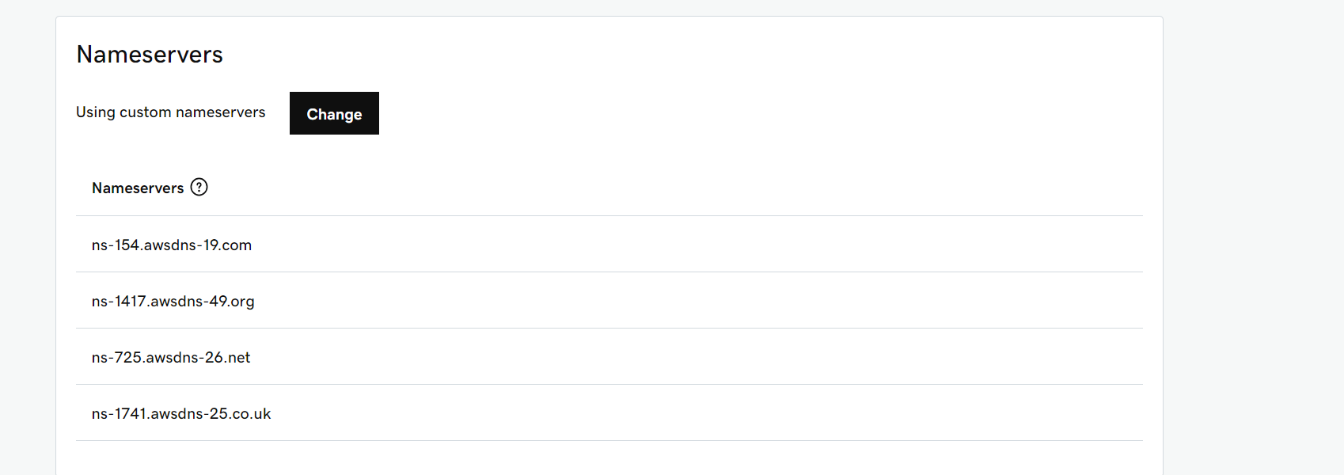
Description automatically generated with medium confidence

*Hình 13: Nhập chứng chỉ đã được ký cho ACM*

* + Tạo hosted zone cho domain trên AWS 53. Khi đó AWS sẽ cung cấp tên của 4 name server. Nhập chúng vào nơi mua domain.

A screenshot of a computer

Description automatically generated with medium confidence



*Hình 14+15: Cấu hình DNS nameserver cho domain*

* + Tạo Load Balancer và Target group cho server.
    - Con load balancer này chứa certificate import từ ACM. Giao thức lắng nghe từ internet là HTTPS, giao thức giao tiếp với server là HTTP.

A screenshot of a computer

Description automatically generated with medium confidence

*Hình 16: Thông tin Load balancer*

* + - Target group là nơi chứa những server. Load balancer sẽ luân phiên chuyển tiếp request tới những server có tình trạng tốt.

A screenshot of a computer

Description automatically generated

*Hình 17: Thông tin target group*

* + Tại AWS Route 53, tạo DNS record type A để chuyển từ “mmhmicro.online” sang tên DNS của con load balancer.

A screenshot of a computer

Description automatically generated with medium confidence

*Hình 18: Chuyển từ tên miền sang tên DNS của load balancer*

A screenshot of a computer

Description automatically generated

*Hình 19: Mô hình triển khai trên AWS*

## Kết quả đạt được:

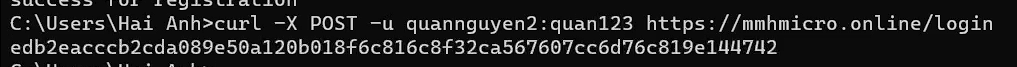
Về bản chất, client chỉ cần 1 công cụ tạo ra HTTP request để gọi đến api của server.

* Trên Windows Client, sử dụng công cụ curl để tương tác với server:
  + Đăng ký tài khoản:
    - Option -X: tên phương thức
    - Option -u: thông tin tài khoản, mật khảu



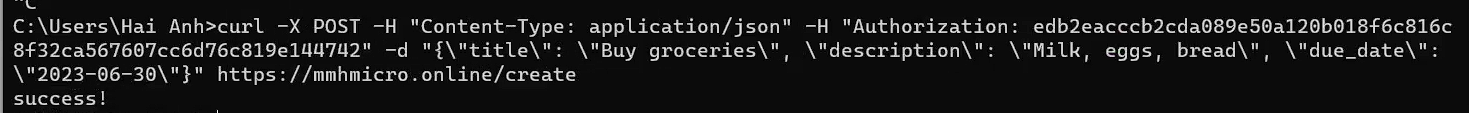
*Đăng ký người dùng*

* Đăng nhập, server sẽ trả về 1 opaque token có hiệu lực trong 10p:



*Đăng nhập*

* Sử dụng token vừa cấp để thao tác trên task service (tạo ghi chú):
  + Option -H: thêm trường header
    - Authorization: <opaque-token>
  + Option -d: thêm dữ liệu vào server trong phương thức POST



*Thao tác trên server*

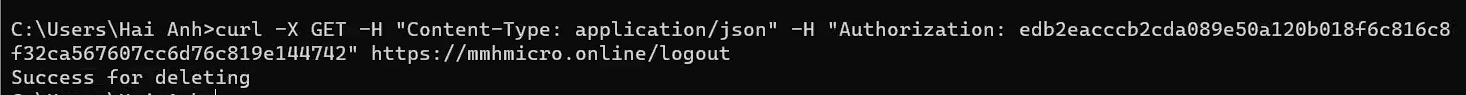
* Sử dụng token này để lấy ghi chú vừa ghi:

A computer screen with white text

Description automatically generated with low confidence

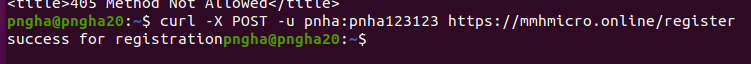
*Thao tác trên server*

* Đăng xuất. Server sẽ xóa dynamic token và JWT:



*Đăng xuất*

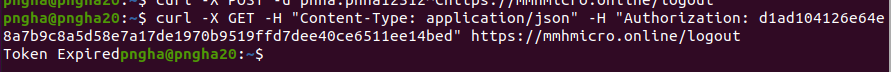
* Tương tự trên Ubuntu Client:
  + Đăng ký:



* + Đăng nhập:



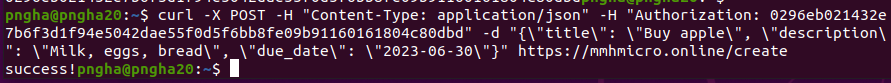
* + Thao tác với server khi token hết hạn:



* + Cần đăng nhập lại:



* + Tạo ghi chú trên server:

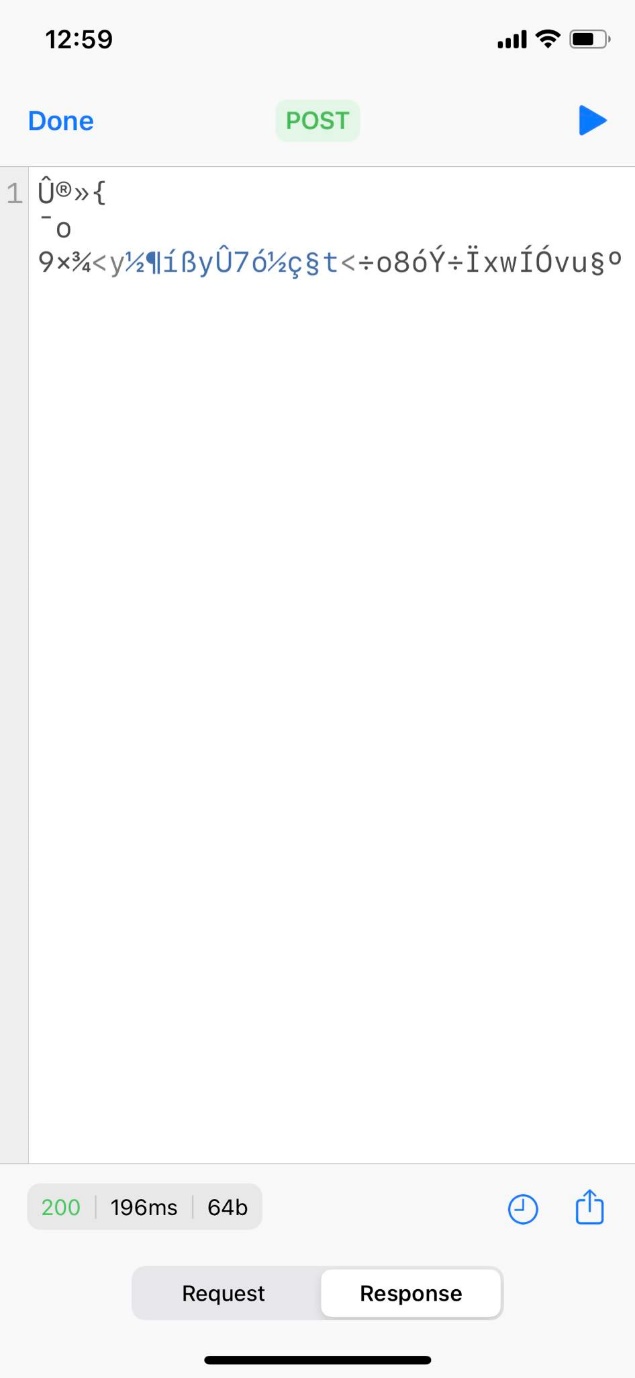


* Trên điện thoại:
  + Đăng nhập tài khoản đã tạo trên máy tính Linux

Screens screenshot of a phone

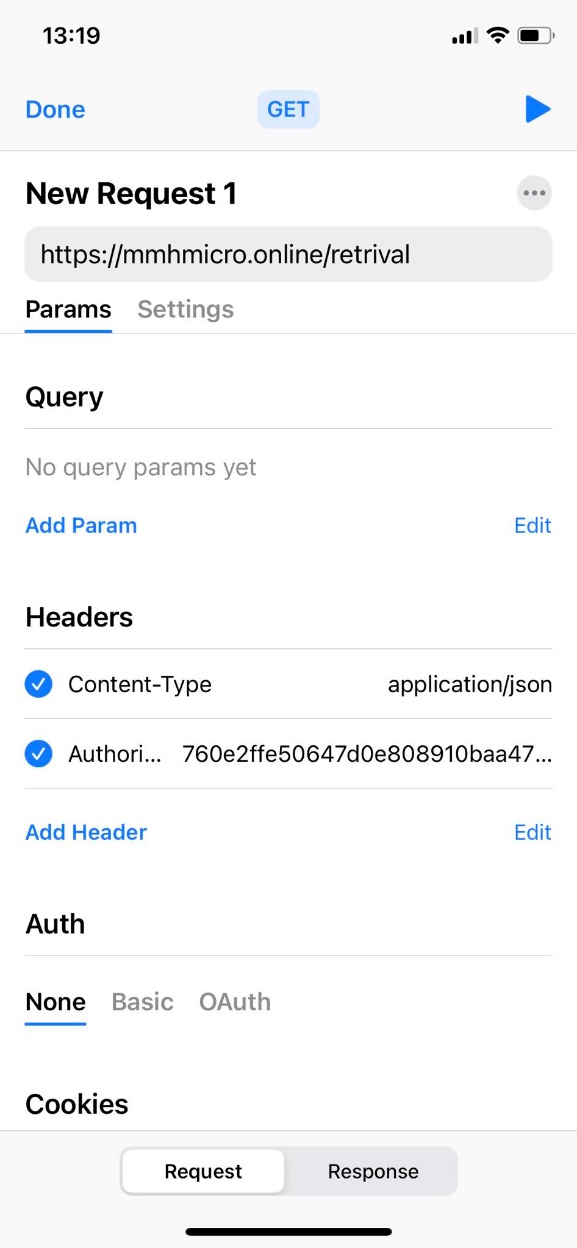
Description automatically generated with low confidence

*Tạo request trên điện thoại tới server*

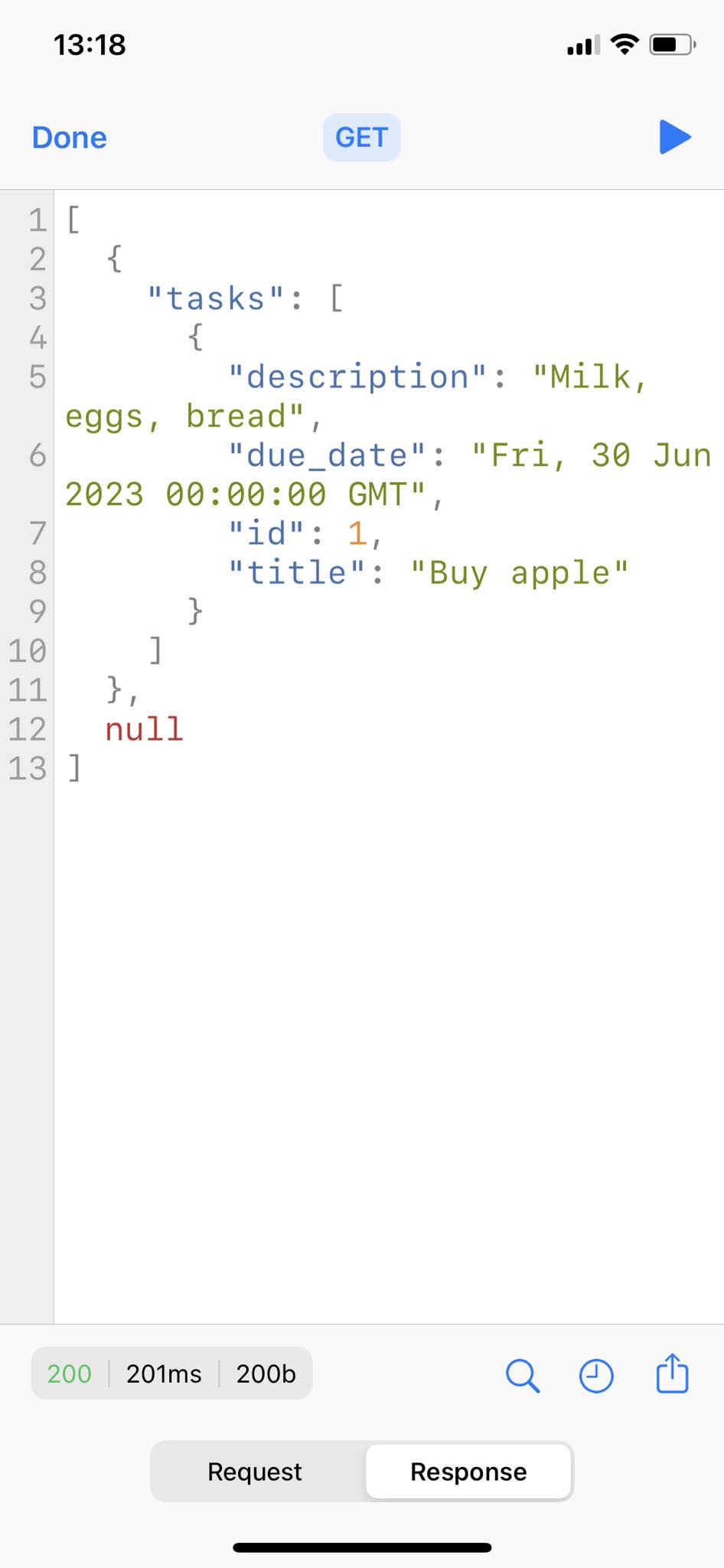


*Server trả opaque*

* + Tương tác với server:
    - Thêm trường header “Authorization: <opaque-token>”



*Client gửi request*



*Server trả response*

TÀI LIỆU THAM KHẢO

1. M. G. de Almeida and E. D. Canedo, “Authentication and Authorization in Microservices Architecture: A Systematic Literature Review,” *Appl. Sci.-Basel*, vol. 12, no. 6, p. 3023, Mar. 2022, doi: [10.3390/app12063023](https://doi.org/10.3390/app12063023).
2. X. He and X. Yang, “Authentication and Authorization of End User in Microservice Architecture,” *J. Phys.: Conf. Ser.*, vol. 910, no. 1, p. 012060, Oct. 2017, doi: [10.1088/1742-6596/910/1/012060](https://doi.org/10.1088/1742-6596/910/1/012060).
3. A. Banati, E. Kail, K. Karoczkai, and M. Kozlovszky, “Authentication and Authorization Orchestrator for microservice-based software architectures,” in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (mipro)*, K. Skala, M. Koricic, T. G. Grbac, M. CicinSain, V. Sruk, S. Ribaric, S. Gros, B. Vrdoljak, M. Mauher, E. Tijan, P. Pale, and M. Janjic, Eds., New York: Ieee, 2018, pp. 1180–1184. Accessed: Apr. 04, 2023. [Online]. Available: <https://www.webofscience.com/wos/woscc/full-record/WOS:000630901400203>
4. R. Melton, “Securing a Cloud-Native C2 Architecture Using SSO and JWT,” in *2021 Ieee Aerospace Conference (aeroconf 2021)*, New York: Ieee, 2021. doi: [10.1109/AERO50100.2021.9438218](https://doi.org/10.1109/AERO50100.2021.9438218).