**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA ĐIỆN TỬ VIỄN THÔNG**

**BÁO CÁO**

**LẬP TRÌNH ĐA NỀN TẢNG**

**RESTful API với HTTP và Dio**

**Sinh viên thực hiện:**

**01. Lâm Đức Duy** **Lớp: 22KTMT1**   **MSSV:**  106220214

**02. Lữ Công Minh** **Lớp: 22KTMT1**   **MSSV:**  106220223

**Người hướng dẫn:**

**TS. Nguyễn Duy Nhật Viễn**

**Đà Nẵng, 2025.**

**THUYẾT MINH**

**BÁO CÁO**

**LẬP TRÌNH ĐA NỀN TẢNG**

**RESTful API với HTTP và Dio**

BẢNG PHÂN CÔNG CÔNG VIỆC TRONG NHÓM

|  |  |  |  |
| --- | --- | --- | --- |
| STT | HỌ VÀ TÊN | NHIỆM VỤ | KHỐI LƯỢNG |
| 01 | Lâm Đức Duy | Tìm hiểu, code nội dung 1, 2 | 50% |
| 02 | Lữ Công Minh | Tìm hiểu, code nội dung 3 | 50% |

 Link code github: <https://github.com/LUCONGMINH/Ung-dung-lay-danh-sach-tu-JSONPlaceholder-API>

**Mục lục**

[**1.** **REST API là gì** 4](#_Toc212485883)

[a) HTTP là gì trong Flutter? 4](#_Toc212485884)

[b) Dio là gì trong Flutter? 4](#_Toc212485885)

[**Code lấy dữ liệu từ API :** 5](#_Toc212485886)

[**Kết quả của code** 6](#_Toc212485887)

[**Ví dụ một ứng dụng của http và dio:** 7](#_Toc212485888)

[**2.** **So sánh http vs dio về tính năng, performance** 8](#_Toc212485889)

[**3.** **Xử lý Authentication và Interceptor (Dio)** 8](#_Toc212485890)

[a) Authentication: 8](#_Toc212485891)

[**Code và kết quả về Authentication:** 9](#_Toc212485892)

[b) Interceptor 10](#_Toc212485893)

[**Code về Interceptor:** 10](#_Toc212485894)

[c) Error handling 11](#_Toc212485895)

[**Code và kết quả về Error handling** 12](#_Toc212485896)

[d) Retry mechanisms : 13](#_Toc212485897)

[**Code về Retry mechanisms** 13](#_Toc212485898)

[**TÀI LIỆU THAM KHẢO**  14](#_Toc212485899)

**NỘI DUNG**

1. **REST API là gì**

API (Application Programming Interface) : Hiểu đơn giản là một cách thức để một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác …Kiểu dữ liệu mà API trả về có thể là file JSON hoặc XML.

REST (Representational State Transfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau.

## ****HTTP là gì trong Flutter?****

* Http là **gói (package)** chính thức của Flutter dùng để **gửi yêu cầu HTTP** đến các API RESTful (GET, POST, PUT, DELETE, ...).
* Dễ dùng, nhẹ, phù hợp với ứng dụng nhỏ hoặc demo.

## ****Dio là gì trong Flutter?****

* Dio là **HTTP client mạnh mẽ hơn** thay thế cho http.
* Hỗ trợ đầy đủ các tính năng nâng cao như:
  + **Interceptors**
  + **FormData (upload file)**
  + **Timeout, cancel token**
  + **Error handling nâng cao**
  + **Retry request**
  + **Download/upload progress**

File Json của api <https://jsonplaceholder.typicode.com/posts> có dạng

A screenshot of a computer

Description automatically generated

**Code lấy dữ liệu từ API :**

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

**Kết quả của code**

A screenshot of a computer

Description automatically generated

**Ví dụ một ứng dụng của http và dio:**

**Code xoá dữ liệu**

A computer screen shot of a code

Description automatically generated

Kết quả của code sau khi xoá bài viết 2,3,4

A screenshot of a computer

Description automatically generated

1. **So sánh http vs dio về tính năng, performance**

| **Tiêu chí / Tính năng** | **HTTP** | **Dio** |
| --- | --- | --- |
| **Mức độ** | Cơ bản | Nâng cao |
| **Cách dùng** | Dễ, ngắn gọn | Linh hoạt, nhiều tùy chọn |
| **Dễ dùng** | Rất đơn giản | Cần cấu hình nhiều hơn |
| **Performance** | Tốt, nhẹ | Tốt, hỗ trợ async mạnh |
| **Interceptor** | Không có | Có (logging, auth, retry, header injection) |
| **Timeout / Cancel request** | Không hỗ trợ | Dễ thiết lập, có sẵn (CancelToken) |
| **Upload/Download progress** | Không có | Có sẵn |
| **Retry** | Thủ công | Có thể thêm RetryInterceptor |
| **Debug/Logging** | Thủ công | Có sẵn Interceptors / LogInterceptor |
| **Error handling** | Cơ bản | Chi tiết (DioException) |
| **FormData / File upload** | Thủ công | Có sẵn tiện ích |
| **Phù hợp cho** | App nhỏ, test nhanh | App lớn, nhiều API, cần bảo mật |

1. **Xử lý Authentication và Interceptor (Dio)**
   1. **Authentication**:

* Để một ứng dụng có thể hoạt động với cơ chế đăng ký, đăng nhập thì điều chúng ta cần quan tâm là Cơ chế Authentication (hay Xác thực người dùng).
* Mỗi user khi đăng ký/đăng nhập thành công thì server sẽ trả về access\_token: Định danh user nào đăng nhập. Thông thường access\_token sẽ tồn tại với thời gian khoảng 24h (hoặc ngắn hơn tùy nghiệp vụ của mỗi project). Thời gian này được gọi là Expired time.
* refresh\_token: Token dùng để lấy lại access\_token mới khi access\_token cũ hết hạn. Thời hạn tồn tại của refresh\_token sẽ dài hơn access\_token
* expired time: thời gian tồn tại của access\_token

### **Code và kết quả về Authentication:**

A screen shot of a computer code

Description automatically generated

A screenshot of a login screen

Description automatically generated A screenshot of a login screen

Description automatically generated

* 1. **Interceptor**
* Có thể hiểu như một bước tường lưới chặn các request, response của ứng dụng để cho phép kiểm tra, thêm vào header hoặc thay đổi các param của request, response.
* Nó cho phép chúng ta kiểm tra các token ứng dụng, Content-Type hoặc tự thêm các header vào request.
* Các thành phần chính của Dio Interceptor:
* onRequest(RequestOptions options): dùng để handle request trước khi gửi cho server.
* onResponse(Response response): dùng để handle reponse trước khi gửi cho client.
* onError(DioError error): handle error trước khi gửi cho client.

### **Code về Interceptor:**

A screen shot of a computer code

Description automatically generated

A computer code with text

Description automatically generated

* 1. **Error handling**
* Là cơ chế **bắt lỗi** khi request API hoặc xử lý logic trong app thất bại
* Ví dụ: server trả 500, token hết hạn (401), mạng lỗi, timeout,...

| **Loại lỗi (DioExceptionType)** | **Nguyên nhân** |
| --- | --- |
| connectionTimeout | Kết nối server quá lâu |
| sendTimeout | Gửi request quá lâu |
| receiveTimeout | Nhận response quá lâu |
| badResponse | Server trả về mã lỗi (400–500) |
| cancel | Request bị huỷ |
| unknown | Lỗi mạng, không xác định |

### **Code và kết quả về Error handling**

A computer error message

Description automatically generated

A close-up of a computer screen

Description automatically generated

A computer screen shot of a program code

Description automatically generated

A screenshot of a computer

Description automatically generated

* 1. **Retry mechanisms :**
* Khi một request bị lỗi (mất mạng, timeout, server 500, v.v.), code **tự động gửi lại request đó** thay vì để người dùng thao tác lạ

### **Code về Retry mechanisms**

A screenshot of a computer code

Description automatically generated

# **TÀI LIỆU THAM KHẢO**

[1]. **Viblo.asia**, "Dio Flutter: Tìm hiểu về Interceptor trong Dio và Triển khai cơ chế Authentication", **Viblo**, 2024.

[2]. **Dart Team**, "Pub.dev: The Dart and Flutter Package Repository", **Google**, 2018.

[3]. **Baoflutter.com**, "Kiến Thức Về Flutter", **Baoflutter**, 2024.

***Chú ý:***

Phải tuân thủ format của tài liệu tham khảo.

Nếu có tham khảo các thuyết minh, phải trích dẫn và thể hiện cụ thể trong các tài liệu tham khảo. Code của khóa trước phải upload đầy đủ code cũ lẫn code mới, trong thuyết minh phải chỉ ra được cái mới của code nhóm thực hiện. Nếu phát hiện sao chép mà không trích dẫn, không có trong mục tài liệu tham khảo thì toàn nhóm sẽ bị điểm 0.