Lý thuyết:

1. Tìm hiểu API là gì

API là cụm viết tắt của Giao diện chương trình ứng dụng. Trong ngữ cảnh API, từ “Ứng dụng” đề cập đến mọi phần mềm có chức năng riêng biệt. Giao diện có thể được xem là một hợp đồng dịch vụ giữa 2 ứng dụng. Hợp đồng này xác định cách thức hai ứng dụng giao tiếp với nhau thông qua các yêu cầu và phản hồi. Tài liệu API của các ứng dụng này chứa thông tin về cách nhà phát triển xây dựng cấu trúc cho những yêu cầu và phản hồi đó.

1. Restful API là gì

REST: là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE … đến một URL để xử lý dữ liệu.  
  
REST hoạt động dựa chủ yếu trên phương thức CRUD ( Create, Read, Update, Delete) tương đương với 4 giao thức HTTP: POST, GET, PUT, DELETE.  
  
RESTful API: là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng Web (như thiết kế Web services), để tiện cho việc quản lý các resource. Nó chú trọng vào resource hệ thống (như: tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái resource được định dạng và được truyền tải qua HTTP.  
  
REST là viết tắt của cụm từ Representational State Transfer và các ứng dụng sử dụng kiểu thiết kế REST thì được gọi là RESTful (-ful là tiếp vị ngữ giống như beauty và beautiful). Có thể sử dụng thuật ngữ REST thay cho RESTful và ngược lại.

1. **Các phương thức của API**

API được xây dựng trên chính 2 thành phần: Request và Reponse

**Về request**

Một cái request đúng chuẩn cần có 4 thứ:

1. **URL**  
   URL là địa chỉ duy nhất cho 1 request, thường là đường dẫn tới một hàm xử lí logic.
2. **Method**  
   HTTP request có tất cả 9 loại method , 2 loại được sử dụng phổ biến nhất là GET và POST

* GET: Sử dụng để lấy thông tin từ server theo URI đã cung cấp.
* HEAD: Giống với GET nhưng response trả về không có body, chỉ có header.
* POST: Gửi thông tin tới sever thông qua các parameters HTTP.
* PUT: Ghi đè tất cả thông tin của đối tượng với những gì được gửi lên.
* PATCH: Ghi đè các thông tin được thay đổi của đối tượng.
* DELETE: Xóa resource trên server.
* CONNECT: Thiết lập một kết nối tới server theo URI.
* OPTIONS: Mô tả các tùy chọn giao tiếp cho resource.
* TRACE: Thực hiện một bài test loop-back theo đường dẫn đến resource.

1. **Headers**  
   Là nơi chứa các thông tin cần thiết của 1 request nhưng end-users không biết có sự tồn tại của nó. Ví dụ: độ dài của request body, thời gian gửi request, loại thiết bị đang sử dụng, loại định dạng cái response mà client có đọc được…
2. **Body**  
   Là nơi chứa thông tin mà client sẽ điền.

**Về response**

Sau khi nhận được request từ phía client, server sẽ xử lý cái request đó và gửi ngược lại cho client 1 cái response. Cấu trúc của 1 response tương đối giống phần request nhưng Status code sẽ thay thế cho URL và Method. Tóm lại, nó có cầu trúc 3 phần:

* Status code
* Headers
* Body  
  Phần Header và body tương đối giống với request.

1. Status code là gì, các status code chuẩn …

Các HTTP status code là chuẩn để server trả về. HTTP status code giúp xác định request thành công hay không, nếu thất bại thì nguyên nhân là gì.

Ví dụ: bạn hay gặp nhất đó là mã HTTP status code = [404](https://stackjava.com/404) khi gửi request/truy cập 1 đường link không tồn tại.

Mục đích sinh ra để debug

1. Thành thạo thao tác đọc ghi file json
2. API endpoint là gì?
3. Mô hình client-server.
4. Thư viện requests (dành cho client)
5. Postman (giả lập môi trường API)
6. fastAPI là gì, cách sử dụng

Bất đồng bộ: không chờ đợi. (fastAPI)

Đồng bộ: chờ đợi. (Flask)

Nội dung bài tập lớn: Làm được máy chủ web phim (backend).

API GET /category/phim\_id … (API trả về thông tin của phim, link video)

CRUD (create, update, delete, get) API