Báo cáo về sử dụng Framework

(Framework report)

<Tên dự án>

Soạn bởi: Hoàng Hữu Phước

Mục lục

[I.Giới thiệu 3](#_Toc162476346)

[II. Các thành phần và tính năng chính 3](#_Toc162476347)

[III. Sử dụng các thành phần 3](#_Toc162476348)

[IV. Ưu điểm so với những framework khác 4](#_Toc162476349)

[V. Những hạn chế và thách thức 4](#_Toc162476350)

[VI. Tổng kết 4](#_Toc162476351)

# 

# **I.Giới thiệu**

ExpressJS là một framework mạnh mẽ và phổ biến được xây dựng trên nền tảng Node.js. Nó giúp đơn giản hóa quá trình phát triển ứng dụng web bằng cách cung cấp các công cụ và thư viện cần thiết để xử lý yêu cầu và phản hồi HTTP một cách dễ dàng.

# **II. Các thành phần và tính năng chính**

1. Routing (Định tuyến): Express cung cấp một cách dễ dàng để định tuyến các yêu cầu HTTP tới các xử lý tương ứng. Điều này giúp bạn tổ chức mã của mình thành các đường dẫn (routes) riêng biệt cho các tác vụ khác nhau trong ứng dụng.
2. Middleware: Middleware trong Express là các hàm chạy trước khi các yêu cầu đến các tác vụ xử lý chính. Middleware có thể được sử dụng để thực hiện các nhiệm vụ như xác thực người dùng, ghi nhật ký, xử lý dữ liệu đầu vào, và nhiều hơn nữa.
3. Xử lý yêu cầu và phản hồi (Request and Response Handling): Express giúp dễ dàng xử lý các yêu cầu HTTP và tạo ra các phản hồi phù hợp, cho phép bạn gửi trả lời dạng HTML, JSON, hoặc bất kỳ loại dữ liệu nào khác.
4. Xử lý lỗi (Error Handling): Express cung cấp các cơ chế để xử lý lỗi trong ứng dụng, bao gồm cả xử lý lỗi 404 (không tìm thấy) và các lỗi xử lý yêu cầu.
5. Phục vụ file tĩnh (Static Files Serving): chp phép phục vụ các tệp tĩnh như ảnh, CSS,… một cách dễ dàng

# **III. Sử dụng các thành phần**

1. Routing:

Định tuyến yêu cầu HTTP bằng cách sử dụng các phương thức định tuyến như get(), post(), put(), delete(), vv. của đối tượng app hoặc router.

1. Middleware:

Để sử dụng middleware trong dự án Express, cần chỉ định middleware bằng cách sử dụng phương thức use() của đối tượng app.

1. Xử lí yêu cầu và phản hồi:

Xử lý yêu cầu và phản hồi bằng cách sử dụng các callback được truyền vào các định tuyến và sử dụng các phương thức như res.send(), res.json(), vv. để gửi phản hồi.

1. Xử lí lỗi:

Tạo các middleware xử lý lỗi bằng cách sử dụng app.use() và bắt các lỗi bằng cách sử dụng try-catch trong các xử lý yêu cầu hoặc sử dụng next(err) để chuyển lỗi cho middleware xử lý lỗi.

1. Phục vụ file tĩnh:

Sử dụng middleware tích hợp trong Express bằng cách sử dụng express.static() và chỉ định thư mục chứa các tệp tĩnh.

# **IV. Ưu điểm so với những framework khác**

1. Đơn giản và dễ dùng: cú pháp đơn giản, dễ tiếp cận.
2. Tính linh hoạt: không áp đặt một cấu trúc cụ thể, người dùng có thể tự do điều chỉnh và xây dựng theo ý muốn, điều này giúp Express phù hợp với nhiều loại ứng dụng khác nhau.
3. Kiến trúc Middleware: cho phép thực hiện các yêu cầu trung gian một cách dễ dàng.
4. Hiệu suất cao: nhờ thiết kế đơn giản nên Express có khả năng xử lí với hiệu suất cao, khả năng mở rộng tốt, đáp ứng nhu cầu người dùng nhanh chóng.
5. Cộng đồng sử dụng rộng rãi: có nhiều tài liệu cũng như sự hỗ trợ đến từ cộng đồng giúp dễ dàng giải quyết các vướng mắc khi gặp phải.

# **V. Những hạn chế và thách thức**

1. Thiếu cấu trúc mặc định: như đã trình bày bên trên, việc không có một cấu trúc cụ thể cũng là một hạn chế của Express vì nó gây khó khăn trong việc quản lí mã nguồn trong những dự án phức tạp.
2. Khả năng mở rộng và xây dựng các ứng dụng lớn: hiệu suất của Express trong các ứng dụng lớn có thể giảm đi do khó khăn trong việc quản lí lượng lớn các module và tương tác giữa chúng.

# **VI. Tổng kết**

1. Express sẽ là Framework được sử dụng trong dự án lần này, bởi vì đây là dự án nhỏ, không cần quá nhiều sự mở rộng trong tương lai nên tránh được các hạn chế đã nêu ở trên. Hơn nữa, đây là một Framework đơn giản, dễ dàng sử dụng cho nên thời gian tìm hiểu và đưa vào sử dụng trong phát triển ứng dụng tương đối ngắn, đáp ứng được yêu cầu về thời gian.
2. Trong dự án này, Express góp phần làm tăng tốc độ phát triển, giảm thời gian nghiên cứu cũng như công sức của các thành viên.