BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN TỐT NGHIỆP

TÌM HIỂU PLATFORM NODEJS, XÂY DỰNG MỘT WEBSITE BÁN ĐỒ BẢO HỘ LAO ĐỘNG TẠI CÔNG TY BẢO HỘ LAO ĐỘNG DƯƠNG CHÂU

(địa chỉ tại Số 30 Đường T4B, Phường Tây Thạnh, Quận Tân Phú,

Thành phố Hồ Chí Minh)

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn** | **: TS. Nguyễn Đình Hưng** |
| **Sinh viên thực hiện** | **: Nguyễn Thành Lãnh** |
| **Mã số sinh viên** | **: 61133851** |

Khánh Hòa, tháng 7 - 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN TỐT NGHIỆP**

**XÂY DỰNG WEBSITE QUẢN LÝ BỆNH NHÂN CHO BỆNH VIỆN ĐA KHOA VẠN NINH**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn** | : TS. Phạm Thị Thu Thúy |
| **Sinh viên thực hiện** | **: Nguyễn Minh Trí** |
| **Mã số sinh viên** | **: 61133202** |

Khánh Hòa, tháng 7 - 2023

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN KỸ THUẬT PHẦN MỀM



ĐỒ ÁN TỐT NGHIỆP

TÌM HIỂU PLATFORM NODEJS, XÂY DỰNG MỘT WEBSITE BÁN ĐỒ BẢO HỘ LAO ĐỘNG TẠI CÔNG TY BẢO HỘ LAO ĐỘNG DƯƠNG CHÂU

(địa chỉ tại Số 30 Đường T4B, Phường Tây Thạnh, Quận Tân Phú,

Thành phố Hồ Chí Minh)

|  |  |
| --- | --- |
| GVHD | : TS. Nguyễn Đình Hưng |
| SVTH | : Nguyễn Thành Lãnh |
| MSSV | : 61133851 |

Khánh Hòa, tháng 07 - 2023

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

PHIẾU THEO DÕI TIẾN ĐỘ VÀ ĐÁNH GIÁ ĐỒ ÁN TỐT NGHIỆP

(Dùng cho CBHD và nộp cùng báo cáo Đồ án của sinh viên)

**Tên đề tài:** TÌM HIỂU PLATFORM NODEJS, XÂY DỰNG MỘT WEBSITE BÁN ĐỒ BẢO HỘ LAO ĐỘNG TẠI CÔNG TY BẢO HỘ LAO ĐỘNG DƯƠNG CHÂU.

**Giảng viên hướng dẫn:** TS. Nguyễn Đình Hưng

**Sinh viên được hướng dẫn:** Nguyễn Thành Lãnh **Mssv**: 61133851

**Khóa:** 61 **Ngành:** Công nghệ thông tin

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Lần KT** | **Ngày** | | **Nội dung** | | | | **Nhận xét của GVHD** |
| 1 | 04/02 | | Trao đổi đề tài, viết đề cương đề tài | | | |  |
| 2 | 24/02 | | Khảo sát quy trình bán hàng của cửa hàng | | | |  |
| 3 | 05/03 | | Thiết kế hệ thống | | | |  |
| 4 | 12/03 | | Vẽ sơ đồ ERD | | | |  |
| 5 | 20/03 | | Thiết kế cơ sở dữ liệu | | | |  |
| 6 | 03/04 | | Thiết kế vài cài đặt hệ thống | | | |  |
| **Kiểm tra giữa tiến độ của Trưởng BM** | | | | | | | |
| Ngày kiểm tra:  ………………….. | | Đánh giá công việc hoàn thành: ………% | | | | Ký tên  …………………... | |
| Được tiếp tục: | | Không tiếp tục: | |
| 8 | |  | | |  |  | |
| 9 | |  | | |  |  | |
| 10 | |  | | |  |  | |
| 11 | |  | | |  |  | |
| 12 | |  | | |  |  | |
| 13 | |  | | |  |  | |
| 14 | |  | | |  |  | |
| 15 | |  | | |  |  | |

**Nhận xét chung** (sau khi sinh viên hoàn thành ĐA/KLTN):

|  |  |  |
| --- | --- | --- |
| Điểm hình thức:………/10 | Điểm nội dung:………/10 | **Điểm tổng kết**:………../10 |

|  |  |  |  |
| --- | --- | --- | --- |
| Đồng ý cho sinh viên: | | Được bảo vệ: | Không được bảo vệ: |
|  | *Khánh Hòa, ngày……tháng……năm…..* | | |
|  | **Giáo viên hướng dẫn**  *(Ký và ghi rõ họ tên)* | | |

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

PHIẾU CHẤM ĐIỂM ĐỒ ÁN TỐT NGHIỆP

(Dùng cho CBHD và nộp cùng báo cáo ĐA/KLTN của sinh viên)

**Tên đề tài:** TÌM HIỂU PLATFORM NODEJS, XÂY DỰNG MỘT WEBSITE BÁN ĐỒ BẢO HỘ LAO ĐỘNG TẠI CÔNG TY BẢO HỘ LAO ĐỘNG DƯƠNG CHÂU.

**Chuyên ngành:** Công nghệ thông tin

**Họ và tên sinh viên:** Nguyễn Thành Lãnh **Mssv:** 61133851

**Người phản biện (học hàm, học vị, họ và tên):** TS. Nguyễn Đình Hưng

**Cơ quan công tác:** Trường Đại học Nha Trang

**Phần đánh giá và cho điểm của người hướng dẫn (tính theo thang điểm 10)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tiêu chí đánh giá** | **Trọng số (%)** | **Mô tả chất lượng** | | | | **Điểm** |
| **Giỏi** | **Khá** | **Đạt yêu cầu** | **Không đạt** |
| **9 - 10** | **7 - 8** | **5 - 6** | **< 5** |
| **Xây dựng đề cương nghiên cứu** | **10** |  |  |  |  |  |
| **Tinh thần và thái độ làm việc** | **10** |  |  |  |  |  |
| **Kiến thức và kỹ năng làm việc** | **10** |  |  |  |  |  |
| **Nội dung và kết quả đạt được** | **40** |  |  |  |  |  |
| **Kỹ năng viết và trình bày báo cáo** | **30** |  |  |  |  |  |
| **ĐIỂM TỔNG** | | | | | |  |

*Ghi chú: Điểm tổng làm tròn đến 1 số lẻ.*

**Nhận xét chung** (sau khi sinh viên hoàn thành ĐA/KLTN):

|  |  |  |  |
| --- | --- | --- | --- |
| Đồng ý cho sinh viên: | | Được bảo vệ: | Không được bảo vệ: |
|  | *Khánh Hòa, ngày……tháng……năm…..* | | |
|  | **Cán bộ hướng dẫn**  *(Ký và ghi rõ họ tên)* | | |

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

PHIẾU CHẤM ĐIỂM ĐỒ ÁN TỐT NGHIỆP

(Dùng cho CBPB và nộp cùng báo cáo ĐA/KLTN của sinh viên)

**Tên đề tài:** TÌM HIỂU PLATFORM NODEJS, XÂY DỰNG MỘT WEBSITE BÁN ĐỒ BẢO HỘ LAO ĐỘNG TẠI CÔNG TY BẢO HỘ LAO ĐỘNG DƯƠNG CHÂU.

**Chuyên ngành:** Công nghệ thông tin

**Họ và tên sinh viên:** Nguyễn Thành Lãnh **MSSV:** 61133851

**Người phản biện (học hàm, học vị, họ và tên):** TS. Nguyễn Đình Hưng

**Cơ quan công tác:** Trường Đại học Nha Trang

**Phần đánh giá và cho điểm của người hướng dẫn (tính theo thang điểm 10)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tiêu chí đánh giá** | **Trọng số (%)** | **Mô tả chất lượng** | | | | **Điểm** |
| **Giỏi** | **Khá** | **Đạt yêu cầu** | **Không đạt** |
| **9 - 10** | **7 - 8** | **5 - 6** | **< 5** |
| **Xây dựng đề cương nghiên cứu** | **10** |  |  |  |  |  |
| **Tinh thần và thái độ làm việc** | **10** |  |  |  |  |  |
| **Kiến thức và kỹ năng làm việc** | **10** |  |  |  |  |  |
| **Nội dung và kết quả đạt được** | **40** |  |  |  |  |  |
| **Kỹ năng viết và trình bày báo cáo** | **30** |  |  |  |  |  |
| **ĐIỂM TỔNG** | | | | | |  |

*Ghi chú: Điểm tổng làm tròn đến 1 số lẻ.*

**Nhận xét chung:**

|  |  |  |  |
| --- | --- | --- | --- |
| Đồng ý cho sinh viên: | | Được bảo vệ: | Không được bảo vệ: |
|  | *Khánh Hòa, ngày……tháng……năm…..* | | |
|  | **Cán bộ phản biện**  *(Ký và ghi rõ họ tên)* | | |

**II. Phần nhận xét cụ thể (dựa theo phiếu chấm điểm và khung tiêu chí đánh giá theo Rubric)**

**II.1. Hình thức thuyết minh (tỉ trọng 30%)**

***\* Trình bày*** (*Rõ ràng, mạch lạc? Biểu bảng, hình vẽ trình bày rõ ràng, đúng quy cách?...)*

***\* Bố cục và lập luận*** *(Bố cục hợp lý? Tỉ trọng giữa các phần? Cơ sở lập luận?...)*

***\* Văn phòng*** *(Gọn gàng, súc tích hay rườm rà, khó hiểu? Lỗi văn phạm và chính tả?…)*

**II.2. Nội dung thuyết minh** (tỉ trọng 30%)

***\* Mục tiêu nghiên cứu*** *(Trình bày rõ ràng? Ý nghĩa khoa học và thực tiễn? Tính khả thi?...)*

***\* Tổng quan tài liệu*** *(Phân tích và đánh giá? Độ tin cậy và chất lượng nguồn tài liệu?…)*

***\* Phương pháp nghiên cứu* (***Hiện đại?**Phù hợp với mục tiêu và nội dung nghiên cứu? Mô tả? Đánh giá và so sánh với các phương pháp khác?…)*

**II.3. Kết quả nghiên cứu (**tỉ trọng 20%)

***\* Kết quả đạt được*** *(Độ**tin cậy? Tính sáng tạo? Giá trị khoa học và thực tiễn?...)*

***\* Kết luận*** *(Đáp ứng mục tiêu nghiên cứu? Quan điểm của cá nhân?....)*

**II.4. MỨC ĐỘ TRÍCH DẪN VÀ SAO CHÉP** (tỉ trọng 20%)

**\* Mức độ trích dẫn *(****Đúng quy định? Trung thực, đầy đủ, rõ ràng? Sắp xếp tài liệu tham khảo?...)*

**\* Mức độ sao chép** *(Tỉ lệ sao chép? Hình thức sao chép?...)*

**(Hình ảnh)**

**KẾT QUẢ KIỂM TRA ĐẠO VĂN BẰNG TURNITIN**

LỜI CAM ĐOAN

Tôi cam đoan rằng tôi sẽ tuân thủ mọi quy định, nguyên tắc và đạo đức của liêm chính học thuật trong quá trình học tập và nghiên cứu. Tôi sẽ không sao chép, đạo văn, hay ăn cắp tác phẩm của người khác mà sẽ làm việc chăm chỉ để sản xuất ra những công trình độc lập và đúng sự thật. Tôi sẽ tôn trọng quyền lợi và tài sản trí tuệ của người khác và không vi phạm bản quyền hoặc sử dụng trái phép thông tin của người khác. Tôi cam đoan sẽ trung thực, chính trực và không gian dối trong quá trình làm việc và đưa ra những kết luận chính xác dựa trên dữ liệu thu thập được. Tôi sẽ luôn tôn trọng đạo đức học thuật và thực hiện đúng những quy định của trường đại học và những cơ quan quản lý liên quan.

LỜI CẢM ƠN

Trước tiên, em xin cảm ơn tất cả các thầy, cô trường Đại học Nha Trang, đã nhiệt tình dạy bảo em trong khoảng bốn năm đại học. Để bây giờ em có đủ kiến thức, kỹ năng để thực hiện đề tài này, và những kiến thức quý báu từ thầy, cô sẽ là những kiến thức cốt lõi, là tiền đề cho sự phát triển sau này của em.

Em xin được gởi lời cảm ơn sâu sắc đến thầy Nguyễn Đình Hưng đã tận tình hướng dẫn, cho em những lời khuyên, và hỗ trợ em kịp thời trong khoảng thời gian thực hiện đồ án tốt nghiệp.

Vì vậy, việc thực hiện Đồ án tốt nghiệp đối với em không chỉ là nhiệm vụ để hoàn thành bốn năm đèn sách của mình, mà đây còn là dịp để em có thể hệ thống lại kiến thức, và có thể hệ thống lại kiến thức, áp dụng những kiến thức đó vào thực tế, phát triển khả năng tư duy và giải quyết một vấn đề cụ thể nào đó trong cuộc sống.

Và em cũng xin cảm ơn chân thành đến gia đình và bạn bè đã đồng hành và giúp đỡ em trong suốt khoảng thời gian vừa qua.

Trong quá trình nghiên cứu và thực hiện đồ án sẽ không thể tránh khỏi những sai sót, vì vậy những lời nhận xét của thầy, cô và bạn bè rất quý giá đối với em, em hi vọng sẽ nhận được những lời đóng góp từ thầy, cô và bạn bè để em có thể cải thiện đề tài tốt hơn.

Xin chân thành cảm ơn!

*Khánh Hòa*, ngày … tháng 5 năm 2023

Sinh viên

Nguyễn Thành Lãnh

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

Chương này trình bày các kiến thức nền tảng, tổng hợp các kiến thức cơ bản, cần thiết xoay quanh đề tài, làm tiền đề cơ sở lý luận cho các chương tiếp theo cũng như các phương pháp tiếp cận để giải quyết đề tài.

1. Thương mại điện tử
   1. Khái niệm

Thương mại điện tử, hay còn gọi là e-commerce, là một hình thức kinh doanh trực tuyến, trong đó các giao dịch mua bán hàng hóa, dịch vụ, thông tin hay tài sản khác được thực hiện qua internet hoặc các mạng máy tính khác. Thương mai điện tử cho phép người tiêu dùng mua hàng và thanh toán trực tuyến bằng các hình thức thanh toán điện tử như thẻ tín dụng, ví điện tử, chuyển khoảng trực tuyến, PayPal, … Đồng thời, thương mại điện tử cũng cho phép các nhà bán hàng tiếp cận với một lượng khách hàng mà không bị giới hạn bởi vị trí địa lý. Các trang web thương mại điện tử có thể bao gồm các cửa hàng trực tuyến, trang web đấu giá, thị trường trực tuyến, trang web tư vấn và giới thiệu sản phẩm, ….

* 1. Ưu điểm
* **Tiện lợi:** Thương mại điện tử cho phép khách hàng mua sắm và thanh toán trực tuyến mọi lúc mọi nơi, không cần phải đến các cửa hàng truyền thống. Điều này góp phần giúp tiết kiệm thời gian và công sức cho khách hàng.
* **Cung cấp thị trường lớn:** Thương mại điện tử cho phép các doanh nghiệp bán hàng trực tuyến tiếp cận với một số lượng khách hàng lớn hơn. Đồng thời, các doanh nghiệp có thể chạy quảng cáo và tiếp cận khách hàng thông qua các kênh trực tuyến như mạng xã hội
* **Tiết kiệm chi phí:** Thương mại điện tử cho phép các doanh nghiệp tiết kiệm chi phí cho việc thuê mặt bằng và chi phí hoạt động của cửa hàng truyền thống.
* **Thuận tiện cho người mua và người bán:** Thương mại điện tử cung cấp cho khách hàng nhiều lựa chọn sản phẩm, từ đó giúp khách hàng dễ dàng tìm kiếm và mua sắm sản phẩm mà họ muốn. Ngoài ra, khách hàng còn có thể đánh giá, nhận xét sản phẩm, giúp cho các doanh nghiệp nâng cao chất lượng sản phẩm và dịch vụ của mình.
  1. Nhược điểm
* **Vấn đề về an ninh mạng:** Thương mại điện tử có thể gặp phải vấn đề về an ninh thông tin, bao gồm các cuộc tấn công mạng, lừa đảo và trộm cắp thông tin cá nhân của khách hàng.
* **Khó khăn trong việc xác thực:** Việc xác thực danh tính của khách hàng trực tuyến có thể gặp nhiều khó khăn, dẫn đến rủi ro về bảo mật và độ tin cậy của thông tin.
* **Khả năng phát sinh tranh chấp:** Trong thương mại điện tử, việc phát sinh tranh chấp giữa các bên có thể khó giải quyết do sự vắng mặt của các thủ tục hành chính truyền thống.
* **Chi phí đầu tư ban đầu lớn:** Các doanh nghiệp phải đầu tư một khoản lớn để thiết lập và duy trì hệ thống thương mại điện tử. Ngoài ra, để thu hút khách hàng, các doanh nghiệp cũng phải đầu tư vào việc quảng cáo và tiếp cận khách hàng.
* **Khả năng phát sinh lỗi kỹ thuật:** Thương mại điện tử cần sự hỗ trợ của các hệ thống kỹ thuật phức tạp và liên kết giữa các hệ thống khác nhau. Do đó, khả năng phát sinh lỗi kỹ thuật là rất cao và gây khó khăn cho việc quản lý và vận hành hệ thống.
  1. Mô hình thương mại điện tử phổ biến
* **B2B (Business-to-Business):** Là mô hình thương mại điện tử giữa các doanh nghiệp trực tiếp với nhau. Các giao dịch trong mô hình này thường liên quan đến mua bán hàng hóa, dịch vụ hoặc thông tin giữa các công ty.
* **B2C (Business-to-Consumer):** Là mô hình thương mại điện tử giữa các doanh nghiệp với người tiêu dùng. Mô hình này cho phép khách hàng mua sắm trực tuyến từ các doanh nghiệp và thanh toán qua các hình thức khác nhau.
* **C2C (Consumer-to-Consumer):** Là mô hình thương mại điện tử giữa các cá nhân. Mô hình này cho phép người dùng bán và mua các sản phẩm hoặc dịch vụ từ nhau trên các trang web thương mại điện tử như eBay, Amazon, Alibaba, vv.
* **C2B (Consumer-to-Business):** Là mô hình thương mại điện tử giữa người tiêu dùng và các doanh nghiệp. Mô hình này cho phép người tiêu dùng cung cấp sản phẩm hoặc dịch vụ cho các doanh nghiệp, ví dụ như các trang web freelancer hoặc các trang web mua sắm công nghệ.
* **B2G (Business-to-Government):** Là mô hình thương mại điện tử giữa các doanh nghiệp với các tổ chức chính phủ. Mô hình này cho phép các doanh nghiệp tham gia vào các hoạt động thị trường công cộng và gửi các đề xuất, đấu thầu, hay mua sắm trực tuyến với các cơ quan chính phủ.
* **G2C (Government-to-Consumer):** Là mô hình thương mại điện tử giữa các cơ quan chính phủ và người tiêu dùng. Mô hình này cho phép người dân sử dụng các dịch vụ công cộng trực tuyến như đóng thuế, nộp hồ sơ, đăng ký và thanh toán các khoản phí.

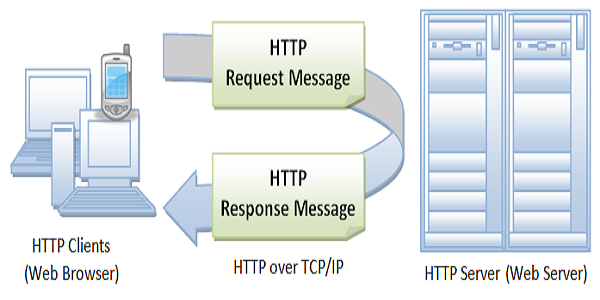
1. Giao thức HTTP
   1. Khái niệm HTTP

Giao thức HTTP (Hypertext Transfer Protocol) là một giao thức truyền tải siêu văn bản giữa bên cung cấp dịch vụ (Web server) và bên sử dụng dịch vụ (Web Client) trong mô hình Client/Server (một mô hình phổ biến hiện nay).

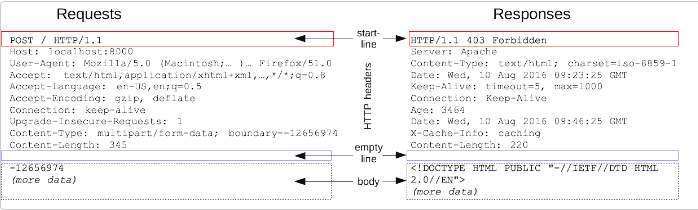
HTTP là một phần của giao thức Internet TCP/IP.

Các yêu cầu HTTP tới máy chủ, sau đó máy chủ phản hồi lại với dữ liệu được yêu cầu. Các yêu cầu HTTP bao gồm các phương thức thông dụng như GET, POST, PUT, DELETE và các phương thức khác.

HTTP sử dụng URL (Uniform Resource Locator) để xác định các tài nguyên cần truy cập và cũng sử dụng các tiêu đề để chuyển tiếp các thông tin khác nhau giữa máy khách và máy chủ, chẳng hạn như các tiêu đề yêu cầu, tiêu đề phản hồi và các tiêu đề khác.

* 1. Nguyên lý hoạt động

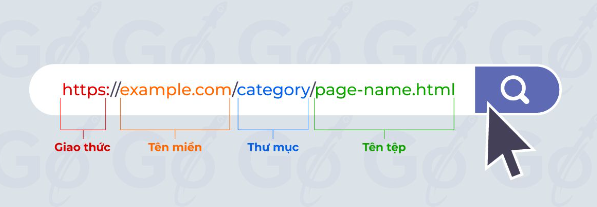
Hình 1. Nguyên lý hoạt động của HTTP

* **Cấu trúc của HTTP Request:**
* **Request line:** là dòng đầu tiên của request, chứa HTTP method, URL của resources, và version của giao được sử dụng
* **Request header:** chứa thông tin bổ sung cho request, chẳng hạn như thông tin client, content type, and cookies.
* **Blank line:** Một dòng trống để ngăn cách giữa headers và body.
* **Request body:** chứa dữ liệu được gửi từ client, chẳng hạn như dữ liệu form hoặc file.
* **Cấu trúc của HTTP Response:**
* **Status line:** là dòng đầu tiên của response, chứa version của giao thức HTTP, mã status cho biết kết quả của request, và mô tả văn bản về trạng thái code
* **Response header:** chứa thông tin bổ sung cho response, như content type và encoding
* **Blank line:** Một dòng trống để ngăn cách giữa headers và body.
* **Response body:** chứa thông tin phản hồi từ máy chủ.

Hình 2. Cấu trúc của một http request

* 1. Uniform Resource Locator (URL)

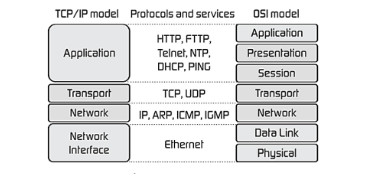
URL (Uniform Resource Locator) là định danh duy nhất của một tài nguyên trên Internet. Nó cho phép truy cập trực tiếp vào một tài nguyên thông qua một trình duyệt web hoặc bất kỳ ứng dụng nào hỗ trợ giao thức Internet.



Hình 3.Cấu trúc minh họa của một URL

URL được định dạng theo chuẩn quốc tế và bao gồm các phần sau:

* **Protocol:** Đây là phần đầu tiên của URL và chỉ định giao thức được sử dụng để truy cập tài nguyên, chẳng hạn như HTTP, HTTPS, FTP hoặc SMTP.
* **Domain:** Phần này chỉ định tên miền hoặc địa chỉ IP của máy chủ chứa tài nguyên.
* **Port:** Đây là số cổng được sử dụng để truy cập tài nguyên trên máy chủ. Nếu không được chỉ định, mặc định là cổng chuẩn của giao thức.
* **Path:** Phần này chỉ định đường dẫn đến tài nguyên trên máy chủ, bao gồm cả tên tập tin và thư mục.
* **Query string:** Phần này được sử dụng để truyền các thông tin bổ sung cho máy chủ, chẳng hạn như thông tin biểu mẫu.
* **Fragment:** Phần này chỉ định một điểm đến cụ thể trong tài liệu, thường là một phần tử HTML, để truy cập trực tiếp từ trình duyệt web.
  1. Giao thức TCP/IP



Hình 4. Mô hình kiến trúc giao thức TCP/IP và OSI seven-layer

TCP/IP là một bộ giao thức mạng được sử dụng để truyền thông giữa các thiết bị trong một mạng máy tính. TCP/IP được viết tắt từ hai giao thức chính trong bộ giao thức này là "Transmission Control Protocol" (TCP) và "Internet Protocol" (IP).

Các giao thức được chia thành các tầng, trong đó TCP/IP có 4 tầng, mỗi tầng lại sử dụng các giao thức ở tầng dưới để đạt được mục đích của mình:

* **Lớp ứng dụng (Application Layer):** Lớp này chịu trách nhiệm cho việc cung cấp các dịch vụ ứng dụng cho người dùng. Ví dụ như truyền tải email, truyền tải dữ liệu web, truyền tải file, và các dịch vụ khác.
* **Lớp Transport (Transport Layer):** Lớp này cung cấp các dịch vụ truyền tải dữ liệu đáng tin cậy và kiểm soát việc truyền tải. Lớp này sử dụng giao thức TCP (Transmission Control Protocol) hoặc UDP (User Datagram Protocol) để đảm bảo dữ liệu được truyền tải một cách đáng tin cậy.
* **Lớp Internet (Internet Layer):** Lớp này cung cấp các dịch vụ định tuyến và phân địa chỉ cho các gói dữ liệu. Lớp này sử dụng giao thức IP (Internet Protocol) để định tuyến các gói dữ liệu từ điểm xuất phát đến điểm đích.
* **Lớp Network Interface (Network Interface Layer):** Lớp này cung cấp các dịch vụ cơ bản để truyền tải các gói dữ liệu qua một đường truyền vật lý. Lớp này sử dụng các giao thức và tiêu chuẩn như Ethernet, Wi-Fi, hay Bluetooth để truyền tải dữ liệu qua một mạng vật lý.

Mỗi lớp trong mô hình TCP/IP đều cung cấp các dịch vụ khác nhau, nhưng chúng hoạt động cùng nhau để cung cấp cho người dùng các dịch vụ mạng đáng tin cậy và hiệu quả.

1. Resful API
   1. Resful API là gì?

API là viết tắt của "Application Programming Interface" (giao diện lập trình ứng dụng) là một tập các quy tắc, giao thức và công nghệ được sử dụng để cho phép các ứng dụng khác nhau trao đổi dữ liệu và tương tác với nhau.

API có nhiều loại khác nhau, bao gồm các API web, API desktop, API mobile, API phần cứng và API thư viện.

RESTful API (Representational State Transfer API) là một kiểu thiết kế API cho phép các ứng dụng trao đổi dữ liệu với nhau qua giao thức HTTP một cách đơn giản, tiêu chuẩn và linh hoạt. RESTful API được thiết kế để sử dụng các phương thức HTTP như GET, POST, PUT, DELETE để truy xuất và thao tác dữ liệu.

* 1. Lợi ích của Resful API
* **Tiêu chuẩn hóa:** RESTful API được xây dựng trên HTTP và các phương thức tiêu chuẩn của nó như GET, POST, PUT, DELETE, v.v. Điều này đảm bảo tính tiêu chuẩn của RESTful API và cho phép các ứng dụng khác nhau tương tác với nhau một cách dễ dàng và hiệu quả.
* **Tính tương thích:** RESTful API hoạt động trên nền tảng web và không yêu cầu bất kỳ công nghệ cụ thể nào. Điều này cho phép các ứng dụng khác nhau sử dụng RESTful API và tương tác với nhau một cách dễ dàng, bất kể họ sử dụng ngôn ngữ lập trình và nền tảng khác nhau.
* **Tính mở rộng:** RESTful API cho phép các ứng dụng mở rộng và phát triển dễ dàng. Các tài nguyên và phương thức của RESTful API có thể được thêm hoặc sửa đổi mà không ảnh hưởng đến các ứng dụng khác.
* **Bảo mật:** RESTful API cung cấp các cơ chế bảo mật để đảm bảo rằng chỉ các ứng dụng được ủy quyền mới có thể truy cập vào các tài nguyên của nó.
* **Dễ sử dụng:** RESTful API được thiết kế để đơn giản và dễ sử dụng. Các nhà phát triển chỉ cần sử dụng các phương thức HTTP và URI để tương tác với RESTful API, mà không cần phải tìm hiểu các giao thức phức tạp khác.
* **Tính linh hoạt:** RESTful API cho phép các ứng dụng khác nhau truy xuất và thao tác dữ liệu theo cách mà họ muốn, cho phép các ứng dụng tùy chỉnh và sử dụng dữ liệu của RESTful API một cách linh hoạt và hiệu quả.
  1. Nhược điểm của Resful API
* **Khó hiểu và phức tạp:** RESTful API có thể trở nên khó hiểu và phức tạp nếu không được thiết kế đúng cách. Các nhà phát triển cần hiểu rõ các tiêu chuẩn và hướng dẫn của RESTful API để thiết kế các API đơn giản và dễ hiểu cho các ứng dụng khác.
* **Không đảm bảo tính nhất quán:** RESTful API không đảm bảo tính nhất quán trong các tài nguyên và phương thức. Điều này có thể dẫn đến các vấn đề về độ tin cậy và hiệu suất nếu các tài nguyên và phương thức của RESTful API không được quản lý và duy trì đúng cách.
* **Thiếu tính năng:** RESTful API không hỗ trợ nhiều tính năng phức tạp như SOAP, chẳng hạn như chứng thực và ủy quyền tối đa. Điều này có thể là vấn đề đối với các ứng dụng có nhu cầu bảo mật và quản lý đối tượng phức tạp.
* **Khó khăn trong việc quản lý phiên:** RESTful API không hỗ trợ quản lý phiên, điều này có thể làm cho việc quản lý phiên và truy xuất các tài nguyên bảo mật trở nên phức tạp và khó khăn hơn.
* **Hiệu suất chậm:** RESTful API có thể làm chậm hiệu suất của ứng dụng nếu các tài nguyên của RESTful API được truy xuất nhiều lần. Điều này có thể là vấn đề đối với các ứng dụng có khối lượng truy cập lớn hoặc tài nguyên phức tạp.

1. NodeJS
   1. Giới thiệu

Node.js là một môi trường chạy mã JavaScript phía máy chủ (server-side) được xây dựng trên nền tảng Chrome V8 JavaScript engine. Nó cho phép bạn thực thi mã JavaScript bên ngoài trình duyệt web và xây dựng các ứng dụng mạng phức tạp như máy chủ web, ứng dụng web thời gian thực, ứng dụng dòng lệnh và nhiều ứng dụng mạng khác.

Node.js được phát triển bởi Ryan Dahl vào năm 2009 và từ đó đã trở thành một công cụ phổ biến cho việc phát triển các ứng dụng mạng hiệu năng cao. Một trong những điểm mạnh của Node.js là sự tập trung vào mô hình không đồng bộ (asynchronous) và sự kiện (event-driven), cho phép xử lý đa nhiệm mà không gây tắc nghẽn hệ thống.

Với Node.js, bạn có thể xây dựng các ứng dụng mạng như máy chủ web, API, ứng dụng thời gian thực, ứng dụng chat, trò chơi trực tuyến và nhiều ứng dụng khác. Nó cung cấp một hệ sinh thái mạnh mẽ với các thư viện và framework phổ biến như Express.js, Socket.io, Sequelize, Mongoose và nhiều thư viện khác để giúp việc phát triển ứng dụng dễ dàng hơn.

Node.js cũng hỗ trợ quản lý các phụ thuộc (dependencies) thông qua npm (Node Package Manager), cho phép bạn dễ dàng cài đặt và quản lý các gói thư viện của bên thứ ba. Điều này giúp bạn tận dụng lại mã nguồn mở có sẵn và phát triển ứng dụng một cách nhanh chóng và hiệu quả.

* 1. Đặc điểm của NodeJS

Một số đặc điểm quan trọng của Node.js bao gồm:

* **Hiệu năng cao:** Node.js sử dụng mô hình không đồng bộ (asynchronous) và sự kiện (event-driven) để xử lý các yêu cầu mạng, giúp tối ưu hóa hiệu năng của ứng dụng.
* **Single-threaded và non-blocking:** Node.js sử dụng mô hình xử lý đơn luồng (single-threaded) để xử lý nhiều yêu cầu mạng đồng thời mà không gây tắc nghẽn (blocking) hệ thống. Nó sử dụng các callback và Promises để xử lý các hoạt động không đồng bộ.
* **Hệ sinh thái mạnh mẽ:** Node.js có một hệ sinh thái mạnh mẽ với nhiều thư viện và framework phổ biến như Express.js, Socket.io, MongoDB, và nhiều hơn nữa. Điều này giúp việc phát triển ứng dụng dễ dàng và nhanh chóng.
* **Cross-platform:** Node.js hỗ trợ nhiều nền tảng, cho phép bạn viết mã một lần và chạy trên nhiều hệ điều hành như Windows, macOS và Linux.
* **Mã nguồn mở:** Node.js là một dự án mã nguồn mở có cộng đồng lớn và tích cực đóng góp. Điều này đảm bảo rằng nó luôn được cải thiện và bổ sung tính năng mới từ cộng đồng phát triển.
  1. Cài đặt NodeJS

Để cài đặt Node.js, bạn có thể làm theo các bước sau:

* Bước 1: Truy cập trang web chính thức của Node.js

Truy cập trang web chính thức của Node.js tại địa chỉ: https://nodejs.org

* Bước 2: Tải xuống phiên bản Node.js

Trên trang web, bạn sẽ thấy hai phiên bản Node.js: LTS (Long Term Support) và Current. Phiên bản LTS được khuyến nghị cho môi trường sản xuất, trong khi phiên bản Current chứa các tính năng mới nhất nhưng không được đảm bảo ổn định. Chọn phiên bản phù hợp với nhu cầu của bạn và nhấp vào nút "Download" để tải xuống tệp cài đặt.

* Bước 3: Chạy tệp cài đặt

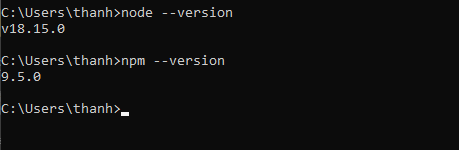
Sau khi tệp cài đặt đã được tải xuống, mở nó và bắt đầu quá trình cài đặt bằng cách nhấp đúp vào tệp cài đặt. Tiếp tục thông qua các bước cài đặt thông thường và chấp nhận các điều khoản và điều kiện.

* Bước 4: Kiểm tra cài đặt

Sau khi cài đặt hoàn tất, bạn có thể kiểm tra phiên bản Node.js và npm (Node Package Manager) bằng cách mở cửa sổ dòng lệnh (Command Prompt hoặc Terminal) và chạy các lệnh sau:

node --version

npm –version



Hình 5. Kiểm tra version nodejs

Nếu bạn nhìn thấy các phiên bản hiển thị mà không có lỗi, điều đó có nghĩa là Node.js đã được cài đặt thành công trên máy tính của bạn.

1. Express
   1. Giới thiệu

Express là một framework phát triển ứng dụng web phía máy chủ (server-side) cho Node.js. Nó là một trong những framework phổ biến và mạnh mẽ nhất để xây dựng các ứng dụng web và API đơn giản và linh hoạt.

Express giúp bạn xử lý các yêu cầu HTTP, quản lý định tuyến (routing), tạo và quản lý các middleware, và tạo các ứng dụng web phức tạp dễ dàng hơn. Nó cung cấp một cách tiếp cận tối giản và linh hoạt để xây dựng các ứng dụng web một cách nhanh chóng.

* 1. Đặc điểm

Các đặc điểm quan trọng của Express bao gồm:

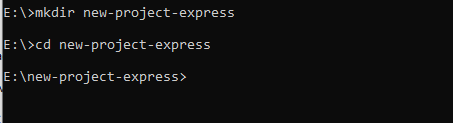
* **Định tuyến (Routing):** Express cho phép bạn xác định các định tuyến (routes) để xử lý các yêu cầu HTTP từ phía máy khách (client). Bằng cách định tuyến các URL tới các xử lý (handlers), bạn có thể điều hướng và xử lý các yêu cầu theo cách linh hoạt.
* **Middleware:** Express sử dụng khái niệm middleware để xử lý các yêu cầu HTTP trước khi đến các xử lý cuối cùng. Middleware có thể làm việc như xác thực người dùng, ghi lại nhật ký, xử lý lỗi, và thực hiện nhiều chức năng khác. Điều này giúp tổ chức mã nguồn và tái sử dụng mã một cách hiệu quả.
* **Cấu hình đơn giản:** Express có cấu hình đơn giản và dễ hiểu, giúp bạn nhanh chóng tạo ra một ứng dụng web đơn giản hoặc mở rộng nó thành một ứng dụng web phức tạp hơn. Nó không đặt ra quá nhiều quy tắc và cho phép bạn linh hoạt trong việc tổ chức mã nguồn của mình.
* **Hệ sinh thái mạnh mẽ:** Express có một hệ sinh thái phong phú với các middleware và các công cụ hỗ trợ như Express Generator giúp tạo nhanh cấu trúc dự án. Ngoài ra, nó cũng tương thích tốt với nhiều thư viện và công nghệ khác trong cộng đồng Node.js.

Vì vậy, Express được sử dụng rộng rãi để phát triển các ứng dụng web, API và dịch vụ web. Với cú pháp đơn giản, khả năng mở rộng và hệ sinh thái mạnh mẽ, Express là một lựa chọn tốt cho việc xây dựng các ứng dụng web phía máy chủ sử dụng Node.js.

* 1. Cách xây dựng một dự án Express

Để xây dựng lên một dự án bạn có thể thực hiện các bước sau:

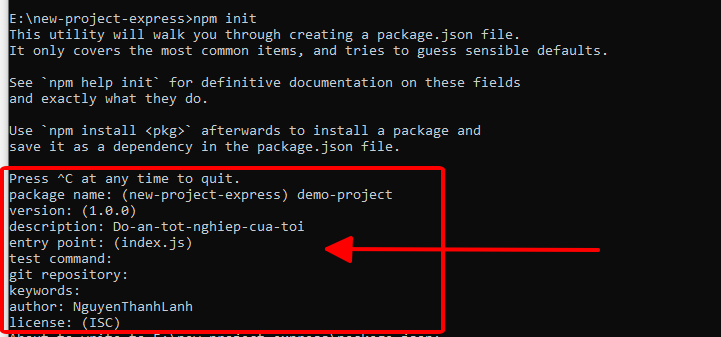
* Bước 1: Tạo thư mục dự án và di chuyển vào thư mục đó:



Hình 6.Step1\_Xây dựng dự án Express

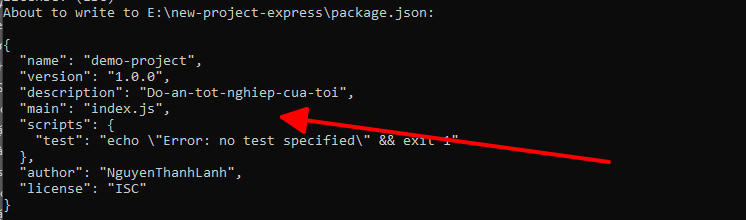
* Bước 2: Khởi tạo dự án và cài đặt Express:

Lệnh **npm init** sẽ tạo ra một tệp **package.json** cho dự án của bạn, trong đó bạn có thể xác định thông tin về dự án và các phụ thuộc của nó.



Hình 7.Step2a \_Xây dựng dự án Express

Thông tin sau khi điền sẽ được hiển thị show phía dưới.



Hình 8.Step2b \_Xây dựng dự án Express

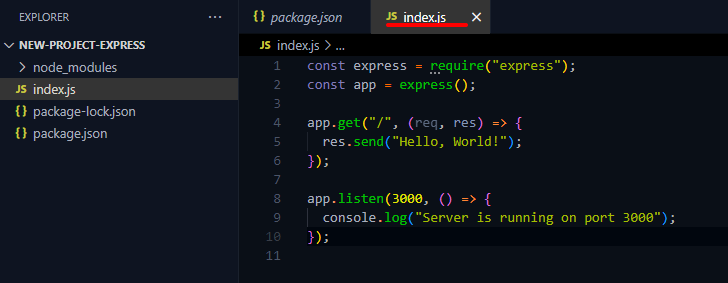
Sau đó, gõ lệnh **npm install express --save** sẽ cài đặt **Express** và lưu nó vào danh sách phụ thuộc trong **package.json.**



Hình 9.Step2c \_Xây dựng dự án Express

* Bước 3: Tạo một tệp JavaScript để viết mã ứng dụng Express:

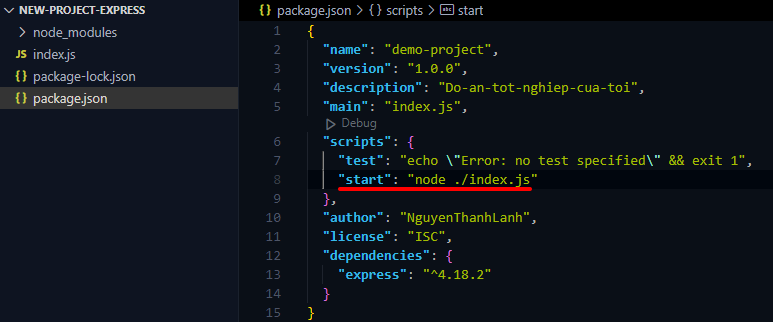
Tạo một tệp JavaScript mới (ví dụ: app.js hoặc index.js, sao cho tên trùng với thuộc tính “**main**” trong file **package.json**) trong thư mục dự án và viết mã ứng dụng Express. Dưới đây là một ví dụ đơn giản:



Hình 10.Step3a \_Xây dựng dự án Express

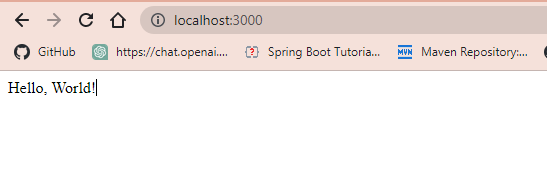
Trong ví dụ trên, chúng ta đã import Express, tạo một đối tượng ứng dụng (app), định nghĩa một đường dẫn / và xử lý yêu cầu GET trên đó. Khi truy cập vào đường dẫn gốc, nó sẽ gửi về thông báo "Hello, World!". Cuối cùng, chúng ta đã khởi động máy chủ bằng cách lắng nghe cổng 3000.

Config start với npm trong file packet.json:



Hình 11.Step3b \_Xây dựng dự án Express

Gõ lệnh npm start để khởi chạy project. Ứng dụng Express của bạn sẽ chạy trên máy chủ và lắng nghe trên cổng 3000. Bạn có thể truy cập vào ứng dụng bằng cách mở trình duyệt và điều hướng đến **http://localhost:3000**



Hình 12.Step3c\_Xây dựng dự án Express

* 1. Một số dependencies cơ bản trong Express
     1. Body-parser
        1. Giới thiệu

**“body-parser”** là một middleware phổ biến trong Node.js để xử lý dữ liệu gửi từ yêu cầu HTTP của phía máy khách (client). Nó giúp phân tích và trích xuất dữ liệu từ phần thân (body) của yêu cầu HTTP, như dữ liệu form, dữ liệu JSON hoặc dữ liệu khác được gửi lên từ trình duyệt.

Trước khi phiên bản Express 4.x, body-parser được sử dụng như một middleware mặc định trong Express để xử lý dữ liệu từ yêu cầu POST và PUT. Tuy nhiên, từ phiên bản Express 4.x trở đi, body-parser đã được chuyển thành một gói riêng biệt, và bạn cần cài đặt nó riêng để sử dụng trong dự án của mình.

* + - 1. Cài đặt

Để cài đặt body-parser, bạn có thể sử dụng npm bằng cách chạy lệnh sau:

* npm install body-parser –save

Sau khi cài đặt thành công, bạn có thể sử dụng body-parser trong mã nguồn của mình bằng cách import nó và sử dụng như một middleware trong ứng dụng Express.

Dưới đây là một ví dụ về cách sử dụng body-parser trong một ứng dụng Express:



Hình 13.Hướng dẫn sử dụng body-parser

* + 1. Cookie-parser
       1. Giới thiệu

“**cookie-parser**” là một middleware trong Node.js và Express để xử lý và phân tích các cookie gửi từ phía máy khách (client). Nó cho phép bạn đọc, ghi và xóa cookie trong ứng dụng Express của mình.

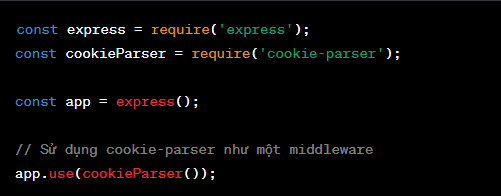
* + - 1. Cài đặt

Để sử dụng cookie-parser, bạn cần cài đặt gói cookie-parser bằng npm bằng lệnh sau:

* npm install cookie-parser –save

Sau khi cài đặt thành công, bạn có thể sử dụng cookie-parser trong mã nguồn của mình bằng cách import nó và sử dụng như một middleware trong ứng dụng Express.

Dưới đây là một ví dụ về cách sử dụng cookie-parser trong một ứng dụng Express:



Hình 14. Cách sử dụng cookie-parser.

* + 1. bcrypt.js
       1. Giới thiệu

“bcrypt.js” là một thư viện mã hóa băm dữ liệu trong Node.js. Nó cung cấp các phương thức để mã hóa và so sánh các mật khẩu dựa trên thuật toán bcrypt, một thuật toán mã hóa mạnh mẽ và an toàn cho việc lưu trữ mật khẩu.

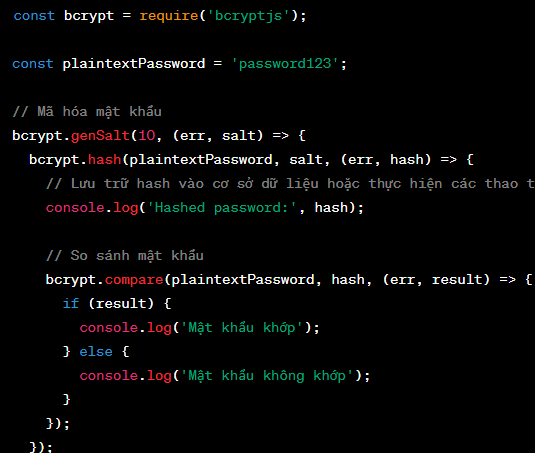
* + - 1. Cài đặt

Để sử dụng bcrypt.js, bạn cần cài đặt nó bằng npm:

* npm install bcryptjs –save

Sau khi cài đặt, bạn có thể sử dụng nó trong mã nguồn của mình bằng cách import và sử dụng các phương thức của bcrypt.js.

Dưới đây là một ví dụ về cách sử dụng bcrypt.js để mã hóa và so sánh mật khẩu:



Hình 15. Cách dùng bcrypt.js

* + 1. CORS (Cross-Origin Resource Sharing)
       1. Giới thiệu

CORS (Cross-Origin Resource Sharing) là một chính sách an ninh trong trình duyệt web để kiểm soát việc chia sẻ tài nguyên giữa các trang web từ các nguồn khác nhau (origin). Nó giúp ngăn chặn các cuộc tấn công liên trang web và bảo vệ sự riêng tư và an toàn của người dùng.

Khi một trang web yêu cầu tài nguyên từ một nguồn khác (ví dụ: yêu cầu AJAX từ một miền khác), trình duyệt sẽ thực hiện một yêu cầu CORS. Trước khi trình duyệt gửi yêu cầu thực tế, nó sẽ gửi một yêu cầu CORS preflight OPTIONS để kiểm tra xem tài nguyên có được phép chia sẻ hay không. Nếu máy chủ đáp ứng yêu cầu preflight OPTIONS với các tiêu chuẩn xác nhận, trình duyệt sẽ tiếp tục gửi yêu cầu thực tế.

Tuy nhiên, một số trình duyệt thi hành chính sách CORS khá nghiêm ngặt, chỉ cho phép các yêu cầu từ cùng một nguồn (same-origin) hoặc từ các nguồn được xác định rõ (có CORS headers phù hợp). Điều này gây ra vấn đề khi phát triển ứng dụng web phía máy khách (frontend) gửi yêu cầu đến một API phía máy chủ (backend) từ miền khác.

Để giải quyết vấn đề này, chúng ta có thể sử dụng middleware CORS trong ứng dụng Express để xác định các cài đặt liên quan đến chính sách CORS. Một trong những thư viện phổ biến để xử lý CORS trong Node.js và Express là **cors**.

* + - 1. Cài đặt

Để sử dụng cors trong ứng dụng Express, bạn cần cài đặt gói cors bằng npm:

* npm install cors –save

Sau đó, trong mã nguồn của ứng dụng Express, import và sử dụng middleware cors:

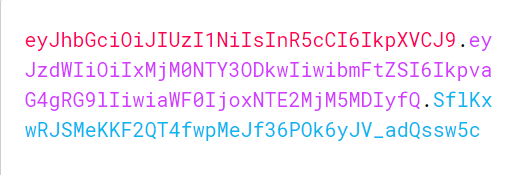


Hình 16. Cách sử dụng cors

* + 1. JWT Token
       1. Giới thiệu

JSON Web Tokens (JWT) là một tiêu chuẩn mở (RFC 7519) được sử dụng để tạo và xác thực thông tin dưới dạng một chuỗi token. JWT được sử dụng rộng rãi trong các ứng dụng web và di động để xác thực người dùng và truyền thông tin an toàn giữa các bên.

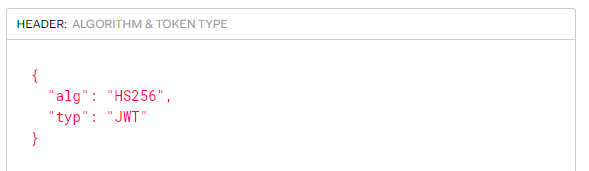
Một JWT gồm ba phần chính: Header, Payload và Signature, được ngăn cách bằng dấu chấm (.). Cấu trúc của một JWT có dạng: xxxxx.yyyyy.zzzzz.



Hình 17.Minh họa JWT Token

* **Header**: Chứa thông tin về loại token và thuật toán mã hóa/ký số hóa được sử dụng.

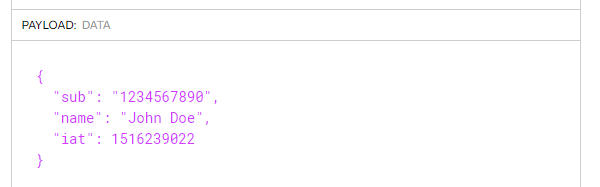
Trong đó, "alg" đại diện cho thuật toán mã hóa/ký số hóa (ví dụ: HS256 cho HMAC-SHA256), "typ" đại diện cho loại token (thường là "JWT").



Hình 18. Minh họa header JWT token

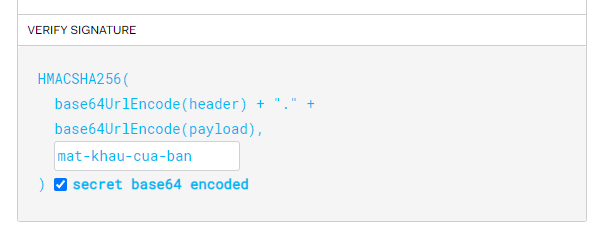
* **Payload:** Chứa thông tin (dữ liệu) mà bạn muốn chứa trong JWT.

Payload có thể chứa các thông tin như thông tin người dùng, quyền hạn, thời gian hết hạn, và các trường thông tin tùy chỉnh khác. Payload có thể là một đối tượng JSON hoặc một chuỗi base64 URL-encoded. Ví dụ:



Hình 19. Minh họa payload JWT

* **Signature:** Được tạo bằng cách ký số hóa Header và Payload bằng một khóa bí mật (hoặc khóa công khai) để đảm bảo tính toàn vẹn của JWT. Chữ ký này được thêm vào cuối chuỗi token. Ví dụ:



Hình 20.Minh họa Signature JWT

Trong đó, secretKey là khóa bí mật được sử dụng để ký số hóa. Chữ ký này được sử dụng để xác minh rằng JWT không bị chỉnh sửa trong quá trình truyền tải.

Khi một ứng dụng nhận được JWT, nó có thể kiểm tra tính toàn vẹn và xác thực JWT bằng cách xem xét Header, Payload và Signature. Nếu chữ ký được tạo bởi cùng một khóa bí mật và không bị thay đổi, JWT được coi là hợp lệ và ứng dụng có thể tin tưởng thông tin trong nó.

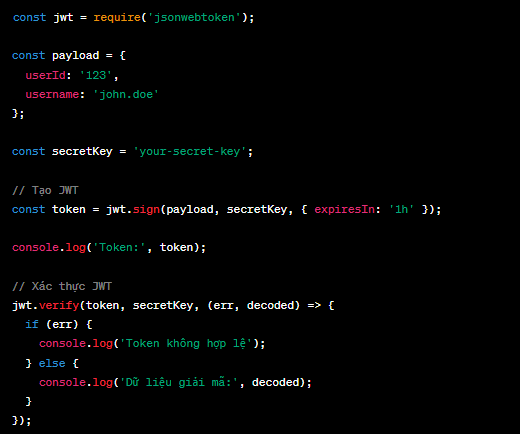
* + - 1. Cài đặt

Để sử dụng jsonwebtoken trong ứng dụng Node.js, bạn cần cài đặt nó bằng npm:

* npm install jsonwebtoken –save

Sau đó, trong mã nguồn của ứng dụng, bạn có thể import và sử dụng các phương thức của jsonwebtoken.

Dưới đây là một ví dụ về cách tạo và xác thực JWT bằng jsonwebtoken:



Hình 21. Cách dùng JWT token

* + 1. Multer
       1. Giới thiệu

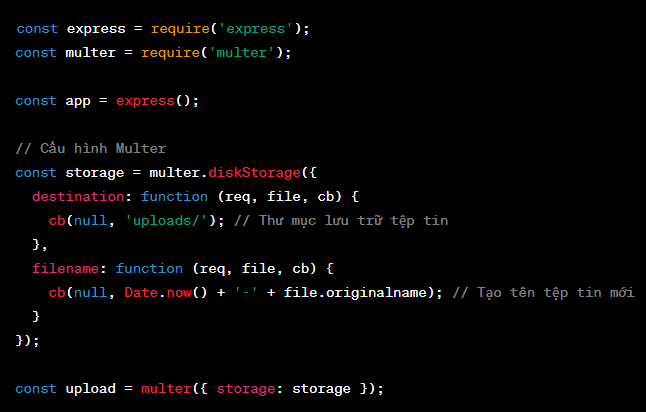
Multer là một middleware Node.js được sử dụng để xử lý tải lên (upload) các tệp tin (file) trong ứng dụng web. Nó giúp bạn dễ dàng nhận và lưu trữ các tệp tin được gửi từ client lên server.

* + - 1. Cài đặt

Để sử dụng Multer, bạn cần cài đặt nó bằng npm. Mở Terminal hoặc Command Prompt và di chuyển đến thư mục gốc của dự án Node.js, sau đó chạy lệnh sau để cài đặt Multer:

* npm install multer –save

Sau khi cài đặt thành công, bạn có thể import Multer vào mã nguồn của ứng dụng Node.js. Và để xử lý tải lên tệp tin, bạn cần cấu hình Multer và áp dụng nó như một middleware cho các tuyến (routes) trong ứng dụng của bạn. Dưới đây là một ví dụ cơ bản về cách sử dụng Multer:



Hình 22. Cách sử dụng multer trong dự án

* + 1. Nodemailer
       1. Giới thiệu

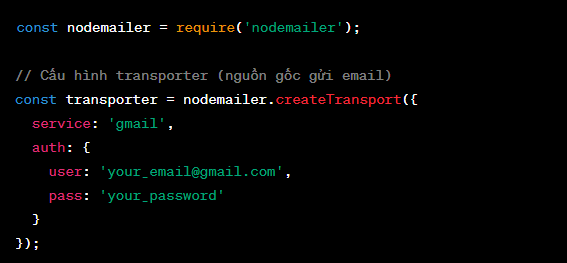
Nodemailer là một thư viện Node.js mạnh mẽ được sử dụng để gửi email từ máy chủ. Nó cung cấp các chức năng linh hoạt cho việc tạo, định dạng và gửi email trong ứng dụng của bạn.

* + - 1. Cài đặt

Để sử dụng Nodemailer, bạn cần cài đặt nó bằng npm. Mở Terminal hoặc Command Prompt và di chuyển đến thư mục gốc của dự án Node.js, sau đó chạy lệnh sau để cài đặt Nodemailer:

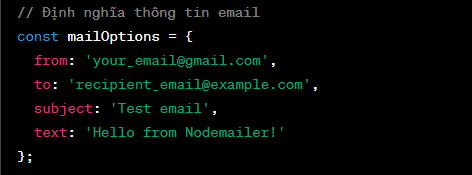
* npm install nodemailer –save

Để gửi email bằng Nodemailer, bạn cần cấu hình một tài khoản email để sử dụng làm nguồn gốc gửi. Dưới đây là một ví dụ cơ bản về cách sử dụng Nodemailer để gửi email:



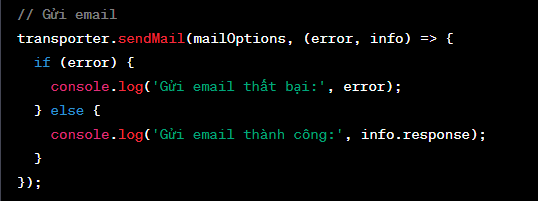
Hình 23. Nodemailer\_Cấu hình transporter nodemailer

Sau đó, bạn cần config những thông tin cần thiết của mail gởi đi.



Hình 24. Nodemailer\_Cấu hình thông tin cần thiết mail gởi

Cuối cùng, dùng phương thức sendMail() để tiến hành gởi mail.



Hình 25. Nodemailer\_Gởi mail trong nodemailer

Nodemailer cung cấp nhiều tính năng và tùy chọn để tùy chỉnh việc gửi email, bao gồm gửi email với tệp đính kèm, định dạng HTML, sử dụng template, và nhiều hơn nữa. Bạn có thể tìm hiểu thêm về Nodemailer và các tính năng khác tại trang web chính thức hoặc tài liệu của Nodemailer.

* + 1. Morgan
       1. Giới thiệu

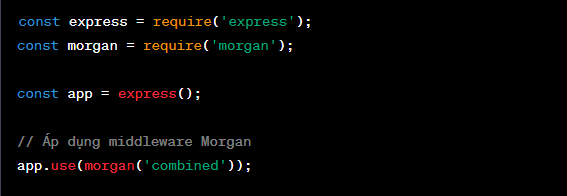
Morgan là một middleware Node.js được sử dụng để ghi lại các yêu cầu (requests) và phản hồi (responses) trong ứng dụng web. Nó cung cấp một cách dễ dàng để theo dõi các hoạt động của ứng dụng và ghi lại thông tin quan trọng như URL, phương thức HTTP, mã trạng thái và thời gian phản hồi.

* + - 1. Cài đặt

Để sử dụng Morgan, bạn cần cài đặt nó bằng npm. Mở Terminal hoặc Command Prompt và di chuyển đến thư mục gốc của dự án Node.js, sau đó chạy lệnh sau để cài đặt Morgan:

* npm install morgan –save

Bạn cần áp dụng nó như một middleware cho ứng dụng Express. Dưới đây là một ví dụ cơ bản về cách sử dụng Morgan để ghi lại các yêu cầu và phản hồi trong ứng dụng:



Hình 26. Cách dùng morgan trong dự án

Trong ví dụ trên, chúng ta đã sử dụng **morgan('combined')** để cấu hình Morgan với định dạng **'combined'**. Định dạng **'combined'** bao gồm thông tin chi tiết như địa chỉ IP của client, phương thức HTTP, URL, mã trạng thái và thời gian phản hồi.

Sau khi áp dụng middleware Morgan bằng cách sử dụng **app.use()**, mọi yêu cầu và phản hồi trong ứng dụng sẽ được ghi lại và hiển thị trong Console.

* + 1. Nodemon
       1. Giới thiệu

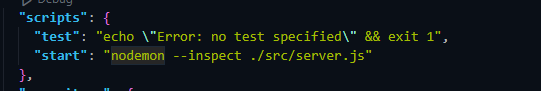
Nodemon là một công cụ giúp phát hiện các thay đổi trong mã nguồn của ứng dụng Node.js và tự động khởi động lại máy chủ mỗi khi có thay đổi. Điều này tiết kiệm thời gian và công sức khi phát triển ứng dụng, vì bạn không cần phải khởi động lại máy chủ thủ công sau mỗi lần chỉnh sửa mã.

* + - 1. Cài đặt

Để sử dụng Nodemon, bạn cần cài đặt nó toàn cục bằng npm. Mở Terminal hoặc Command Prompt và chạy lệnh sau:

* npm install -g nodemon

Bạn có thể config “start” trong thuộc tính “scripts” ở file packet.json để mỗi khi khởi động dự án, thì khởi động với nodemon



Hình 27. Cấu hình dự án với nodemon

Khi ứng dụng chạy, Nodemon sẽ theo dõi các tệp tin trong thư mục hiện tại và các thư mục con của nó. Nếu bất kỳ tệp tin nào được thay đổi, Nodemon sẽ tự động khởi động lại máy chủ để áp dụng các thay đổi mới. Bạn sẽ thấy thông báo trong Terminal khi máy chủ được khởi động lại.