**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

---



**BÁO CÁO ĐỒ ÁN CHUYÊN NGHÀNH**

**ĐỀ TÀI:**

**TÌM HIỂU CÁC KỸ THUẬT CHIA MẠNG CON VÀ**

**XÂY DỰNG CHƯƠNG TRÌNH MINH HỌA TRÊN CLOUD**

**NGÀNH: AN TOÀN THÔNG TIN**

**KHÓA: 2021-2025**

**SINH VIÊN: HỒ TRUNG PHƯỚC MSSV: 2033216521**

**TP. HỒ CHÍ MINH, THÁNG … NĂM 20…**

**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

---

**TÌM HIỂU CÁC KỸ THUẬT CHIA MẠNG CON VÀ XÂY DỰNG CHƯƠNG TRÌNH MINH HỌA TRÊN CLOUD**

**HỒ TRUNG PHƯỚC MSSV: 2033216521**

GIẢNG VIÊN HƯỚNG DẪN:

*(ký tên)*

**TP. HỒ CHÍ MINH, THÁNG …. NĂM 20….**

**LỜI CAM ĐOAN**

Em xin cam kết rằng bài đồ án tốt nghiệp với đề tài “ Tìm hiểu các kỹ thuật chia mạng con và xây dựng chương trình minh họa trên Cloud ” là kết quả của quá trình nghiên cứu và làm việc độc lập của bản thân. Em không sao chép bất kỳ công trình nghiên cứu nào của người khác, và mọi thông tin sử dụng trong bài đều có nguồn gốc rõ ràng, đã được thẩm định và trích dẫn đúng quy định.

Em xin chịu hoàn toàn trách nhiệm nếu có bất kỳ phát hiện nào liên quan đến hành vi gian lận trong quá trình thực hiện đồ án này.

**TPHCM, ngày .....tháng .....năm….**

**(SV ký và ghi rõ họ tên)**

**LỜI CẢM ƠN**

Để hoàn thành tốt bài đồ án tốt nghiệp này, em xin bày tỏ lòng biết ơn chân thành đến thầy Phạm Tuấn Khiêm, người đã tận tâm hướng dẫn và hỗ trợ em trong suốt quá trình thực hiện đồ án. Thầy đã dành rất nhiều thời gian và công sức để giải đáp mọi thắc mắc, giúp em có được những kiến thức sâu sắc và hoàn thiện bài báo cáo một cách tốt nhất.

Em cũng xin gửi lời cảm ơn chân thành đến toàn thể quý thầy cô trong khoa Công Nghệ Thông Tin, trường Đại học Công Thương Thành Phố Hồ Chí Minh, những người đã tận tình giảng dạy và truyền đạt kiến thức cho em suốt những năm tháng học tập tại trường. Những kiến thức quý báu mà em tiếp thu được không chỉ là nền tảng vững chắc cho quá trình nghiên cứu mà còn là hành trang giá trị để em tự tin bước vào đời.

TPHCM , ngày ....tháng ....năm

Sinh viên thực hiện

**Mục lục**

[MỞ ĐẦU 7](#_Toc176126408)

[1. Lý do chọn đề tài 7](#_Toc176126409)

[2. Mục tiêu nghiên cứu 8](#_Toc176126410)

[3. Phạm vi nghiên cứu 8](#_Toc176126411)

[4. Phương pháp nghiên cứu 9](#_Toc176126412)

[5. Ý nghĩa của đề tài 9](#_Toc176126413)

[CHƯƠNG 1: TÌM HIỂU CÁC KỸ THUẬT CHIA MẠNG CON TRONG IPv4 10](#_Toc176126414)

[1.1 IPv4 là gì ? 10](#_Toc176126415)

[1.1.1 Ưu điểm và nhược điểm của địa chỉ IPv4 11](#_Toc176126416)

[1.1.2 Phân lớp địa chỉ IPv4 12](#_Toc176126417)

[1.1.3 Subnet Mask 14](#_Toc176126418)

[1.2 Các kỹ thuật chia mạng con 18](#_Toc176126419)

[1.2.1 Kỹ thuật FLSM 19](#_Toc176126420)

[1.2.2 Kỹ thuật CIDR 20](#_Toc176126421)

[1.2.3 Kỹ thuật VLSM 24](#_Toc176126422)

[CHƯƠNG 2: TÌM HIỂU VỀ KỸ THUẬT TRIỂN KHAI WEBSITE THÔNG QUA MÔI TRƯỜNG ĐIỆN TOÁN ĐÁM MÂY 27](#_Toc176126423)

[2.1 Khái niệm Điện toán đám mây 27](#_Toc176126424)

[2.2 Mô hình điện toán đám mây 28](#_Toc176126425)

[2.2.1 Public Cloud 28](#_Toc176126426)

[2.2.2 Private Cloud 29](#_Toc176126427)

[2.2.3 Hybrid Cloud 29](#_Toc176126428)

[2.2.4 Community Cloud 29](#_Toc176126429)

[2.3 Các loại hình dịch vụ điện toán đám mây phổ biến: SaaS, PaaS và IaaS 30](#_Toc176126430)

[2.3.1 SaaS – Phần mềm dưới dạng dịch vụ 30](#_Toc176126431)

[2.3.2 PaaS – Nền tảng dưới dạng dịch vụ 31](#_Toc176126432)

[2.3.3 IaaS – Cơ sở hạ tầng dưới dạng dịch vụ 32](#_Toc176126433)

[2.4 Triển khai Website trên Môi Trường Điện Toán Đám Mây 34](#_Toc176126434)

[CHƯƠNG 3: MINH HỌA CHƯƠNG TRÌNH CHIA MẠNG CON TRÊN WEBSITE 37](#_Toc176126435)

[3.1 Github Pages là gì ? 37](#_Toc176126436)

[3.2 Chuẩn bị mã nguồn cho ứng dụng chia mạng con 37](#_Toc176126437)

[3.2.1 Tệp index.html 38](#_Toc176126438)

[3.2.2 Giải thích cấu trúc của tệp index.html 45](#_Toc176126439)

[3.2.3 Tệp script.js 47](#_Toc176126440)

[3.2.4 Giải thích các hàm của tệp script.js 52](#_Toc176126441)

[3.3 Minh họa chương trình chia mạng con trên website bằng GitHub Pages. 60](#_Toc176126442)

[Bước 1: Tạo Kho Lưu Trữ (Repository) Mới trên GitHub 60](#_Toc176126443)

[Bước 2: Tải Tệp Lên Kho Lưu Trữ 63](#_Toc176126444)

[Bước 3: Kích Hoạt GitHub Pages 64](#_Toc176126445)

[Bước 4: Truy cập Trang Web 65](#_Toc176126446)

[Bước 5: Cập Nhật và Duy Trì Trang Web 66](#_Toc176126447)

[TÀI LIỆU KHAM KHẢO 67](#_Toc176126448)

# MỞ ĐẦU

Trong thời đại kỷ nguyên số hiện nay, sự phát triển nhanh chóng của công nghệ thông tin đã thay đổi căn bản cách thức các doanh nghiệp vận hành và cung cấp dịch vụ. Việc triển khai các dịch vụ trực tuyến, đặc biệt là các website, không chỉ đóng vai trò như một kênh giao tiếp quan trọng với khách hàng mà còn là một yếu tố quyết định sự thành công và cạnh tranh của các doanh nghiệp trong thị trường toàn cầu. Để đáp ứng nhu cầu truy cập ngày càng tăng và đảm bảo hiệu suất cao, các tổ chức đang dần chuyển sang sử dụng các giải pháp điện toán đám mây. Những giải pháp này không chỉ cung cấp sự linh hoạt và khả năng mở rộng mà còn tối ưu hóa chi phí vận hành và quản lý tài nguyên.

Cùng với sự phát triển của các dịch vụ trực tuyến, quản lý mạng máy tính hiệu quả cũng trở thành một yếu tố thiết yếu để đảm bảo hoạt động ổn định, bảo mật và tối ưu cho hệ thống. Một trong những kỹ thuật quan trọng để quản lý mạng máy tính hiệu quả là chia mạng con (subnetting) trong IPv4. Việc hiểu và áp dụng các kỹ thuật chia mạng con giúp các tổ chức tối ưu hóa việc sử dụng địa chỉ IP, tăng cường bảo mật và cải thiện hiệu suất của mạng.

## Lý do chọn đề tài

* Nhằm đáp ứng nhu cầu thực tiễn và nâng cao kiến thức chuyên sâu về các công nghệ nền tảng trong quản lý và triển khai hệ thống mạng, đề tài “Tìm hiểu các kỹ thuật chia mạng con trong IPv4 và tìm hiểu về kỹ thuật triển khai website thông qua môi trường điện toán đám mây” được lựa chọn. Mục tiêu chính của đề tài này là:
* Nắm vững: Các kiến thức cơ bản và nâng cao về mạng máy tính, đặc biệt là kỹ thuật chia mạng con (subnetting) trong IPv4, từ đó có thể tối ưu hóa tài nguyên mạng và nâng cao hiệu suất hệ thống.
* Hiểu rõ: Các khái niệm, quy trình và phương pháp triển khai website trên nền tảng điện toán đám mây. Tập trung vào các nền tảng đám mây phổ biến như AWS (Amazon Web Services), Azure (Microsoft Azure), và GCP (Google Cloud Platform), nhằm hiểu rõ các tính năng và lợi ích mà mỗi nền tảng mang lại.
* Áp dụng: Kết hợp giữa kiến thức lý thuyết và thực hành để xây dựng một hệ thống mạng và website đơn giản, nhưng hiệu quả và có khả năng mở rộng, qua đó tăng cường kỹ năng triển khai và quản lý hệ thống.

## Mục tiêu nghiên cứu

* Mục tiêu chung: Nghiên cứu và đánh giá các kỹ thuật chia mạng con trong IPv4 và phương pháp triển khai website trên nền tảng điện toán đám mây, nhằm xây dựng một hệ thống mạng và website đáp ứng các yêu cầu về hiệu năng, bảo mật, và khả năng mở rộng.
* Mục tiêu cụ thể:
* Khảo sát và phân tích các thuật toán chia mạng con phổ biến như CIDR (Classless Inter-Domain Routing) và VLSM (Variable Length Subnet Masking), so sánh ưu và nhược điểm của từng phương pháp để lựa chọn kỹ thuật phù hợp nhất với các tình huống cụ thể.
* Nghiên cứu và so sánh các mô hình triển khai website trên đám mây, xác định mô hình phù hợp nhất cho từng loại hình doanh nghiệp và yêu cầu hệ thống.
* Thực hành triển khai một website đơn giản trên nền tảng đám mây đã chọn, bao gồm việc cấu hình mạng, cài đặt ứng dụng web, và tối ưu hóa hiệu năng của hệ thống.
* Đánh giá hiệu quả của hệ thống đã triển khai dựa trên các tiêu chí về hiệu năng, bảo mật và khả năng mở rộng.

## Phạm vi nghiên cứu

* Về lý thuyết: Nghiên cứu các khái niệm cơ bản và nâng cao liên quan đến mạng máy tính, bao gồm IPv4, subnet mask, routing, và các thuật toán chia mạng con. Đồng thời, tìm hiểu các mô hình và dịch vụ điện toán đám mây, như IaaS (Infrastructure as a Service), PaaS (Platform as a Service), và SaaS (Software as a Service), cũng như các nền tảng đám mây phổ biến (AWS, Azure, GCP).
* Về thực hành: Triển khai một website đơn giản sử dụng các công nghệ web phổ biến như HTML, CSS, JavaScript, và một ngôn ngữ lập trình backend như PHP, Python, hoặc Node.js. Tập trung vào việc sử dụng các công cụ và dịch vụ hỗ trợ triển khai trên đám mây, ví dụ: AWS Management Console, Azure Portal, Terraform.
* Về công cụ: Sử dụng các công cụ và dịch vụ hỗ trợ triển khai trên đám mây, như AWS Management Console, Azure Portal, và Terraform, để quản lý và cấu hình hệ thống một cách hiệu quả.

## Phương pháp nghiên cứu

* Thu thập thông tin: Nghiên cứu các tài liệu, sách báo, bài báo khoa học, và tài liệu hướng dẫn từ các nhà cung cấp dịch vụ đám mây để thu thập các kiến thức nền tảng và cập nhật.
* Phân tích và tổng hợp: Phân tích và so sánh các thông tin thu thập được để rút ra kết luận về các kỹ thuật chia mạng con và phương pháp triển khai website trên đám mây, từ đó xác định các phương pháp và công nghệ phù hợp nhất.
* Thực hành: Triển khai một dự án thực tế nhỏ để kiểm chứng các kiến thức đã học và đánh giá hiệu quả của các kỹ thuật và phương pháp triển khai được lựa chọn.

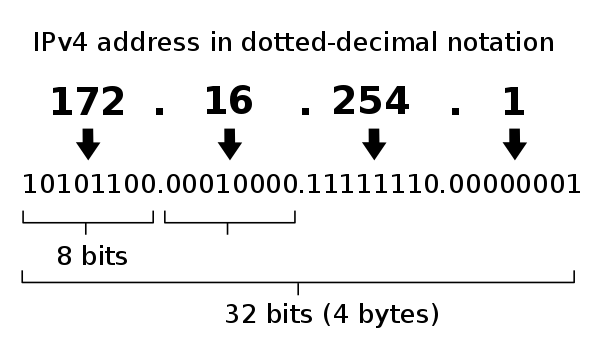
## Ý nghĩa của đề tài

* Đối với cá nhân: Đề tài này giúp rèn luyện kỹ năng nghiên cứu, phân tích, tổng hợp và khả năng áp dụng kiến thức vào thực tế, đồng thời nâng cao hiểu biết về các công nghệ mạng và đám mây.
* Đối với lĩnh vực: Đề tài góp phần nâng cao nhận thức về tầm quan trọng của việc kết hợp giữa kiến thức mạng máy tính và điện toán đám mây trong việc triển khai các ứng dụng web, từ đó thúc đẩy sự phát triển của các dịch vụ trực tuyến chất lượng cao.
* Đối với doanh nghiệp: Kết quả của đề tài có thể cung cấp nền tảng kiến thức vững chắc để doanh nghiệp xây dựng và quản lý các hệ thống web một cách hiệu quả, đáp ứng nhu cầu phát triển và cạnh tranh trên thị trường.

# CHƯƠNG 1: TÌM HIỂU CÁC KỸ THUẬT CHIA MẠNG CON TRONG IPv4

## 1.1 IPv4 là gì ?

* Địa chỉ IP là địa chỉ được dùng để định danh cho một đối tượng trên mạng. Các đối tượng này có thể là máy tính, máy in, camera, điện thoại…gọi là các thiết bị người dùng cuối “end-user devices” hay là các “host”.
* Cấu trúc địa chỉ IP gồm 2 phần: Network.Host
* Có khi người ta gọi là Net\_ID và Host\_ID, nghĩa là phần định danh cho phần Network và phần định danh cho Host. Các địa chỉ IP có cùng phần Network gọi là cùng mạng. Các địa chỉ IP trên một mạng là duy nhất.
* Ipv4 viết tắt cho Internet Protocol Version 4, dịch ra có nghĩa là giao thức Internet phiên bản thứ 4. Ipv4 đã được bộ quốc phòng Hoa Kỳ chuẩn hóa trong bản MIL-STD-1777. Giao thức Internet IP đã trải qua nhiều phiên bản khác nhau và phiên bản Ipv4 là phiên bản đầu tiên được sử dụng rộng rãi trên toàn thế giới và hiện vẫn còn đang là nòng cốt của Internet trên toàn thế giới.
* Ipv4 là giao thức mang tính hướng dữ liệu và được sử dụng cho hệ thống chuyển mạch gói. Ipv4 không quan tâm đến thứ tự truyền gói tin, cũng không đảm bảo gói tin sẽ đến đích hay là có xảy ra tình trạng lặp gói tin ở đích đến hay không. Nó chỉ có cơ chế đảm bảo tính toàn vẹn dữ liệu bằng việc sử dụng những gói kiểm tra được thiết lập đi kèm với nó.
* Địa chỉ Ipv4 là 1 địa chỉ đơn nhất đang được sử dụng bởi các thiết bị điện tử hiện nay để nhận diện và liên lạc với nhau trên Internet. Để đánh địa chỉ, Ipv4 sử dụng 32bit và chia ra làm 4 octet (mỗi octet có 8 bit = 1 byte). Dấu chấm được sử dụng để ngăn các octet với nhau.
* Để hiểu địa chỉ Ipv4 là gì có thể lấy ví dụ như sau: Giả sử ta có 1 dải số như sau: 172.16.254.1. Dải số này có thể được dùng để đặt tên cho 1 địa chỉ Ipv4 nào đó. Có thể thấy địa chỉ Ipv4 có tổng cộng 4 số và mỗi số phải nằm trong giới hạn từ 0-255.



* Các loại địa chỉ Ipv4: unicast, broadcast, multicast. Trong đó unicast là địa chỉ IP cho phép thiết bị gửi dữ liệu đến 1 nơi nhận duy nhất. Địa chỉ IP broadcast lại cho phép gửi dữ liệu đến các host trong 1 mạng con. Còn địa chỉ IP multicast cho phép thiết bị gửi dữ liệu đến 1 tập xác định trước các host.

### 1.1.1 Ưu điểm và nhược điểm của địa chỉ IPv4

* Ưu điểm của địa chỉ IPv4 là gì?
* Giao thức không có kết nối: IPv4 được thiết kế để truyền dữ liệu mà không yêu cầu thiết bị nhận phải sẵn sàng và thiết lập kết nối trước. Điều này cho phép gửi gói dữ liệu qua các đường khác nhau trong trường hợp lỗi router hoặc các vấn đề về kết nối giữa các thiết bị.
* Tạo lớp giao tiếp ảo đơn giản trên nhiều thiết bị: IPv4 cho phép tạo ra một lớp giao tiếp ảo (virtual communication layer) trên nhiều thiết bị, giúp tạo sự kết nối và truyền thông dễ dàng giữa các thiết bị trong mạng.
* Tiết kiệm bộ nhớ và dễ dàng ghi nhớ các địa chỉ: IPv4 sử dụng địa chỉ 32 bit, nên nó chiếm ít bộ nhớ hơn so với các phiên bản IPv6 hoặc các giao thức khác. Đồng thời, địa chỉ IPv4 được biểu diễn dưới dạng các số thập phân phân tách bằng dấu chấm, điều này làm cho nó dễ dàng ghi nhớ và quản lý.
* Hỗ trợ rộng rãi trên hàng triệu thiết bị: IPv4 đã được sử dụng rộng rãi trên hàng triệu thiết bị trong suốt nhiều năm. Điều này có nghĩa là nó có sự tương thích cao và được hỗ trợ bởi nhiều hệ điều hành, thiết bị mạng và ứng dụng.
* Cung cấp thư viện video và hội nghị: IPv4 cung cấp các tính năng như thư viện video và hội nghị, cho phép truyền tải và chia sẻ nội dung video trực tuyến, tổ chức các buổi họp trực tuyến, hội nghị từ xa và truyền phát trực tiếp. Điều này mang lại lợi ích và tiện ích đáng kể trong việc truyền thông và giao tiếp trực tuyến.
* Nhược điểm của địa chỉ IPv4 là gì?
* Cấu trúc thiết kế: IPv4 address sẽ có cấu trúc định tuyến phân cấp và cấu trúc định tuyến không phân cấp. Điều này sẽ dẫn đến việc mỗi router giữ một bảng thông tin định tuyến lớn, yêu cầu nhiều bộ nhớ. IPv4 cũng yêu cầu sự can thiệp nhiều từ router vào các gói dữ liệu IPv4.
* Thiếu hụt không gian địa chỉ: Vấn đề lớn nhất của IPv4 là thiếu hụt không gian địa chỉ. Với chỉ 32 bit, không gian IPv4 address chỉ có 232 địa chỉ, trong khi tốc độ phát triển của Internet ngày càng cao, tài nguyên địa chỉ IPv4 gần như cạn kiệt.
* Vấn đề này gây ra hai thách thức chính: thiếu địa chỉ, đặc biệt là trong không gian địa chỉ tầm trung (lớp B), và kích thước bảng định tuyến lớn đến mức gây hại của Internet.
* Tính bảo mật và kết nối đầu cuối: IPv4 không tích hợp bảo mật vào cấu trúc thiết kế. Bạn cần biết rằng giao thức IPv4 sẽ không hỗ trợ vấn đề mã hóa dữ liệu.
* Do đó, thay vì bảo mật lưu lượng giữa các máy chủ (host), bảo mật thường được thực hiện ở mức ứng dụng. Mặc dù có IPSec là một phương pháp bảo mật phổ biến ở lớp IP, mô hình bảo mật chính tập trung vào bảo mật lưu lượng giữa các mạng, với sự hạn chế trong việc bảo mật lưu lượng đầu cuối.
* Điều này đã thúc đẩy sự nghiên cứu và phát triển giao thức mới như IPv6 để giải quyết các nhược điểm này.

### 1.1.2 Phân lớp địa chỉ IPv4

Địa chỉ IPv4 có 32 bit, chia làm 4 phần (octet), ngăn cách nhau bởi dấu “.”, được biểu diễn dưới dạng thập phân hoặc nhị phân. Địa chỉ IPv4 được chia thành 5 lớp: A, B, C, D, E .

Trong đó:

* + - Các lớp A, B, C được dùng để gán cho các host
    - Lớp D là lớp địa chỉ multicast
    - Lớp E không dùng
* Đặc điểm của các lớp

1. Lớp A (class A):

* Dành 1 octet đầu tiên làm phần Network, 3 octet còn lại làm phần host
* Bit đầu tiên của octet đầu tiên phải là bit 0
* Một mạng lớp A có thể đánh cho 2^24 -2 = 16.777.214 (\*) host
* Như vậy: octet đầu tiên có có trị:
  + 00000000 đến 01111111 (viết dưới dạng nhị phân)
  + Hay từ 0 đến 127 (viết dưới dạng thập phân)
* *Lưu ý:*
  + Giá trị đầu tiên 00000000 khộng dùng
  + Giá trị cuối: 0111111 (127) được dùng làm địa chỉ loopback
  + Kết luận: Địa chỉ lớp A có octet đầu tiên mang giá trị 00000001 đến 01111110 (hay từ 1 đến 126)
  + Ví dụ: 10.10.10.1
  + 00001010.00001010.00001010.00000001

1. Lớp B (class B):

* Dành 2 octet đầu tiên làm phần Network, 2 octet còn lại làm phần host
* 2 Bit đầu tiên của octet đầu tiên phải là bit 10
* Một mạng lớp A có thể đánh cho 2^16 -2 = 65.534 (\*) host
* Như vậy: octet đầu tiên có có trị:
  + 10000000 đến 10111111 (viết dưới dạng nhị phân)
  + Hay từ 128 đến 191 (viết dưới dạng thập phân) Ví dụ: 172.16.10.1
  + 10101100.00010000.00001010.00000001

1. Lớp C (class C):

* Dành 3 octet đầu tiên làm phần Network, 1 octet còn lại làm phần host
* 3 Bit đầu tiên của octet đầu tiên phải là bit 110
* Một mạng lớp A có thể đánh cho 2^8 -2 = 254 (\*) host
* Như vậy: octet đầu tiên có có trị:
  + 11000000 đến 110111111 (viết dưới dạng nhị phân)
  + Hay từ 192 đến 223 (viết dưới dạng thập phân)
  + Ví dụ: 192.168.1.1
  + 11000000.10101000.00000001.00000001

1. Lớp D (class D): Địa chỉ lớp D là địa chỉ Multicast

* 4 Bit đầu tiên của octet đầu tiên phải là bit 1110
* Như vậy: octet đầu tiên có có trị:
  + 11100000 đến 111011111 (viết dưới dạng nhị phân)
  + Hay từ 224 đến 239 (viết dưới dạng thập phân) Ví dụ: 224.0.0.5
  + 1110000.00000000.00000000.00000101

1. Lớp E (class E): Chưa sử dụng

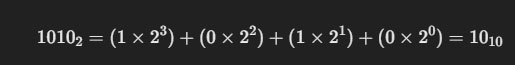
* 5 Bit đầu tiên của octet đầu tiên phải là 11110
* Trong mỗi mạng: có 2 địa chỉ không dùng để gán cho các host
* **Địa chỉ mạng (network address):**là địa chỉ mà tất cả các bit ở phần Host đề là bit 0
* **Địa chỉ Broadcast (broadcast address):**là địa chỉ mà tất cả các bit ở phần Host đề là bit 1
* **IP public và IP private**
* Dãy địa chỉ IP private (RFC 1918)
* Class A: 10.0.0.0  10.255.255.255
* Class B: 172.16.0.0 172.31.255.255
* Class C: 192.168.0.0  192.168.255.255

### 1.1.3 Subnet Mask

* Subnet Mask có chiều dài bit bằng với địa chỉ IP được dùng để chỉ ra trong một địa chỉ IP những bit nào thuộc phần *Network* và những bit nào thuộc phần *Host*. Trong đó, các bit 1 chỉ ra tương ứng các bit thuộc phần *Network* và các bit 0 chỉ ra tương ứng các bit thuộc phần *Host*.
* Subnet mask được biểu diễn dưới dạng: (1) 4 octet giống như địa chỉ IP hoặc (2) /n với n là số bit làm phần Network.
* Subnet Mask mặc định (Default Subnet Mask) cho các lớp IP như sau:
  + Class A: 255.0.0.0 hoặc /8
  + Class B: 255.255.0.0 hoặc /16
  + Class C: 255.255.255.0 hoặc /24
* Subnet là một số dạng 32 bit. Để tạo ra Subnet mask, người ta sẽ đặt host bit dưới dạng số 0 và network bit dạng số 1. Từ đó tạo ra các dãy số có dạng nhị phân là 0 và 1 để phân chia địa chỉ IP thành 2 phần, tương ứng với địa chỉ mạng và địa chỉ host.
* Địa chỉ mạng thường có số 0 và địa chỉ broadcast thường có số 255. Ngoài ra, địa chỉ IP, Subnet Mask và router sẽ có các cấu trúc riêng. Tùy từng khu vực, từng địa chỉ mà địa chỉ thể hiện sẽ có sự khác biệt. Người dùng có thể dựa vào các số này để xác định chính xác IP và route của mình.
* Khi các đơn vị bổ sung subnetworking, đơn vị cần sử dụng subnetting, sau đó chia host thành một subnet. Chính vì thế, khi tìm hiểu subnet là gì, bạn cần lưu ý, mục tiêu chính của subnet mask đó là kích hoạt quá trình subnetting
* Trong đó, subnet mask sẽ có tác dụng che địa chỉ IP của bạn dưới dạng các số 32 bit. Đây cũng là lý do vì sao Subnet Mask có từ “mask” – mặt nạ ở trong tên gọi.

1. **Số nhị phân là gì?**

* **Số nhị phân** là một hệ thống đếm được sử dụng trong lĩnh vực máy tính và toán học, dựa trên cơ số 2. Hệ thống này chỉ sử dụng hai ký tự là **0** và **1** để biểu diễn các giá trị, mỗi ký tự đại diện cho một bậc của **cơ số 2**.
* Trong hệ thống nhị phân, mỗi chữ số (hoặc bit) có thể có giá trị là 0 hoặc 1. Tuy nhiên, giá trị của mỗi chữ số trong số nhị phân được quy định bởi vị trí của nó trong chuỗi. Cụ thể, mỗi chữ số đại diện cho một lũy thừa của 2, bắt đầu từ bên phải với 2^0, sau đó là 2^1, 2^2, và cứ tiếp tục như vậy.



* Trong đó, “1010” là biểu diễn nhị phân của số thập phân 10.
* Số nhị phân được sử dụng rộng rãi trong lĩnh vực máy tính vì dễ dàng biểu diễn và thực hiện các phép toán logic như AND, OR, XOR, và NOT, cũng như các phép toán số học như cộng, trừ, nhân và chia. Đặc biệt, việc sử dụng số nhị phân là cơ sở cho việc lưu trữ và xử lý dữ liệu bên trong các thiết bị điện tử như bộ nhớ và vi xử lý.

1. **Bit trong số nhị phân là gì?**

* Trong hệ thống số nhị phân, mỗi chữ số được gọi là một bit (viết tắt của Binary Digit). Bit là đơn vị cơ bản nhất trong hệ thống số nhị phân và chỉ có thể có hai giá trị: 0 hoặc 1.
* Bit thường được sử dụng để biểu diễn trạng thái logic của các mạch điện trong các thiết bị điện tử. Mỗi bit đại diện cho một trạng thái của mạch, chẳng hạn như mạch đóng (biểu thị bằng 1) hoặc mạch mở (biểu thị bằng 0).
* Khi nhóm nhiều bit lại với nhau, chúng tạo thành các số nhị phân lớn hơn và có khả năng biểu diễn một phạm vi giá trị rộng hơn. Ví dụ, một byte là một nhóm 8 bit, có thể biểu diễn 256 giá trị khác nhau (từ 0 đến 255) trong hệ thống nhị phân.

Ví dụ:

* 10101 là số nhị phân 5 bit.
* 101 là số nhị phân 3 bit.
* 100001 là số nhị phân 6 bit.

1. **Cách chuyển đổi số nhị phân sang số thập phân và ngược lại**

Chuyển từ số nhị phân sang số thập phân:

* Bước 1: Xác định giá trị của mỗi bit nhị phân, bắt đầu từ bên phải.
* Bước 2: Nhân giá trị của mỗi bit với  2^k, trong đó k là vị trí của bit trong chuỗi (bắt đầu từ 0 cho bit cuối cùng và tăng lên 1 cho mỗi bit tiếp theo).
* Bước 3: Tổng hợp các kết quả từ bước 2.

Ví dụ: Chuyển đổi số nhị phân “1010” sang số thập phân:

1010= (1×2^3) + (0x2^2) + (1×2^1) + (0x2^0) = 10

1. **Chuyển từ số thập phân sang số nhị phân:**

* Bước 1: Chia số thập phân cho 2 và ghi lại phần dư.
* Bước 2: Lặp lại quá trình cho đến khi kết quả của phép chia bằng 0.
* Bước 3: Đọc các phần dư từ phải sang trái để tạo thành biểu diễn nhị phân.

Ví dụ: Chuyển đổi số thập phân 10 sang số nhị phân:

* 10÷2 = 5 (dư 0)
* 5÷2 = 2 (dư 1)
* 2÷2 = 1 (dư 0)
* 1÷2 = 0 (dư 1)

Do đó, số nhị phân tương ứng là “1010”.

1. **Chuyển địa chỉ IP sang số nhị phân**

Để đổi [địa chỉ IP](https://thietbimanggiare.com/dia-chi-ip/) từ dạng thập phân sang nhị phân và ngược lại, chúng ta cần hiểu rõ cấu trúc của địa chỉ IP và quy tắc chuyển đổi.

* **Chuyển từ địa chỉ IP thập phân sang nhị phân:**

Một địa chỉ IP gồm 32 bit thường được biểu diễn dưới dạng bốn phần, mỗi phần được biểu diễn bằng một con số từ 0 đến 255, dài 8 bit và phân cách bởi dấu chấm. Để chuyển từ thập phân sang nhị phân, ta chỉ cần chuyển đổi mỗi phần thập phân thành nhị phân.

Ví dụ: Đổi địa chỉ IP thập phân “192.168.1.1” sang nhị phân:

* 192÷2=96 , dư=0 , 96÷2=48, dư=0, 48÷2=24 , dư=0, 24÷2=12 ,dư=0 , 12÷2=6, dư=0, 6÷2=3, dư=0, 3÷2=1, dư=1, 1÷2=0, dư=1 do đó: 192 = 11000000
* Tương tự tính ta có:
* 168 = 10101000
* 1 = 00000001
* 1 = 00000001

**Lưu ý:** là mỗi phần của địa chỉ IP có độ dài là 8 bit nên nó phải là số nhị phân gồm 8 bit.

Do đó, địa chỉ IP “192.168.1.1” được chuyển đổi thành nhị phân là “11000000.10101000.00000001.00000001”.

* **Chuyển từ địa chỉ IP nhị phân sang thập phân:**

Để chuyển từ nhị phân sang thập phân, chúng ta chỉ cần thực hiện phép chuyển đổi ngược lại từ mỗi phần nhị phân sang thập phân.

Ví dụ: Đổi địa chỉ IP nhị phân “11000000.10101000.00000001.00000001” sang thập phân:

* 11000000 = (1×2^8) + (1×2^7) + (0x2^6) + (0x2^5) + (0x2^4) + (0x2^3) + (0x2^2) + (0x2^1) + (0x2^0) = 128 + 64 + 0 + 0 + 0 + 0 + 0 + 0 = 192
* Tương tự các phần còn lại.
* 10101000 = 168
* 00000001 = 1
* 00000001 = 1

Do đó, địa chỉ IP nhị phân “11000000.10101000.00000001.00000001” được chuyển đổi thành địa chỉ IP thập phân “192.168.1.1”.

## 1.2 Các kỹ thuật chia mạng con

* Subneting là một kỹ thuật cho phép tạo ra nhiều mạng con (subnetworks) từ một mạng lớn (major network). Với kỹ thuật này cho phép tạo ra nhiều mạng con (subnetwork) với số lượng host ít hơn, phù hợp cho nhu cầu sử dụng và tối ưu cho hệ thống.
* Để thực hiện điều này, người ta sử dụng một số bit ở phần Host-ID tham gia vào phần Network-ID.
* Ta có một số tính chất cần lưu ý như sau:
  + Nếu gọi n là số bit mượn ở phần Host để chia subneting thì số mạng con (subnetwork) có thể chia là 2n
  + Gọi m là số bit còn lại còn lại của phần host thì số host cho mỗi mạng con là 2m-2, n+m = số bit phần host của mạng ban đầu.
  + Chia mạng con giúp người quản trị chia nhỏ mạng, tối ưu việc sử dụng IP trong mạng và tối ưu việc thực hiện routing cho hệ thống thông qua các supernet (các mạng cha). Để chia mạng con (subnet), chúng ta phải mượn một số bit ở phần host ID làm network ID, các bit mượn được gọi là subnet bit, cụ thể như sau:
  + A screen shot of a computer

    Description automatically generated
  + Gọi n là số bit mượn, m là số bit host còn lại sau khi mượn.
  + Số lượng mạng con có qua số bit mượn:
  + 2n: Đối với hệ thống có hổ trợ network 0 (tất cả các hệ thống hiện tại ngày nay chúng ta dùng, các hệ điều hành đã hỗ trợ subnet 0)
  + 2n-2: đối với hệ thống không hỗ trợ subnet 0 (đối với các hệ thống cũ, các hệ điều hành không hỗ trợ subnet 0)
  + Số host trong subnet: 2m – 2 (như chúng ta đã biết trong bài về [địa chỉ IPv4](https://www.engisv.info/?p=350) thì một mạng sẽ không dùng đến địa chì đầu tiên là network và địa chỉ cuối cùng là broadcast.)
  + Bước nhảy giữa các mạng: sẽ được tính bằng công thức: 28-n. Chính là giá trị của các bit (nhị phân) mượn làm network ID lần lượt chuyển từ giá trị 0 lên giá trị một. Giá trị của bước nhảy dùng để xác định Network tiếp theo.

### 1.2.1 Kỹ thuật FLSM

* Fixed length subnet mask(FLSM) nghĩa là chia theo độ dài subnet cố định dựa trên nhu cầu đường mạng Network ID. Cách chia này sẽ ít sử dụng nhưng nó đơn giản và cơ bản .
* Chúng ta đã biết IP gồm 4byte được chia thành 3 Class và đây là netmask mặc định của từng Class.

|  |  |  |
| --- | --- | --- |
| Lớp | Dạng nhị phân | Netmask |
| A | 11111111 00000000 00000000 00000000 | 255.0.0.0 |
| B | 11111111 11111111 00000000 00000000 | 255.255.0.0 |
| C | 11111111 11111111 11111111 00000000 | 255.255.255.0 |

Chia mạng con theo phương pháp FLSM

* Để chia mạng con ta chỉ quan tâm đến *số đường mạng cần chia*. Ví dụ ta có đường mạng 192.168.1.0 /24 cần chia ra 4 đường mạng con. Các bước làm như sau:
* Phân tích cấu trúc của địa chỉ 192.168.1.0 như sau:

+ Địa chỉ NetMask: 255.255.255.0 viết tắt /24  
+ Dạng nhị phân: 11111111.11111111.11111111.00000000  
+ Network ID: 11111111.11111111.11111111  
+ HostID: 00000000

* Xác định số Bit làm Network cần mượn.  
  Ta có công thức: 2x >= m (với m là số đường mạng cần chia, x là số bit cần mượn).  
  Ở đây 2x >= 4 vậy x=2. Ta sẽ mượn 2 bit ở phần HostID làm đường mạng. Sau khi mượn xong ta sẽ có Subnet mask cho các đường mạng con.

+ Địa chỉ SubNetMask: 255.255.255.192 viết tắt /26 (do mượn thêm 2 bit từ HostID)  
+ Dạng nhị phân: 11111111.11111111.11111111.11000000  
+ Network ID: 11111111.11111111.11111111.11  
+ Host ID: 000000

Giải thích: Câu hỏi đặt ra là Subnet mask 255.255.255.192 ở đâu ra?  
Ở đây ta đang xét đường mạng 192.168.1.0 thuộc lớp C. Do đó 3 byte đầu 255.255.255 ta không bàn tới vì nó là subnet mặc định của lớp C. (vì 8bit là 1 ~ 255) mà ta chỉ phân tích byte thứ 4. Do ta mượn thêm 2 bit làm NetworkID nên ở byte thứ 4 sẽ là 11000000. Theo bảng dưới ta có giá trị 192 = 128 + 64

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bit thứ | … | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| Giá trị Bit | … | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Netmask | … | 128 | 192 | 224 | 240 | 248 | 252 | 254 | 255 |

Để xác định các đường mạng con ta cần tìm “bước nhảy” cho các mạng con: Bước nhảy k = 256 - 192 = 64 => Ta có các mạng con sau:

- Network ID: 192.168.1.0 Netmask: 255.255.255.192 (viết gọn 192.168.1.0/26)  
Khoảng IP được sử dụng: 192.168.1.1 -> 192.168.1.62  
-  Network ID: 192.168.1.64/26  
Khoảng IP được sử dụng: 192.168.1.65 -> 192.168.1.126.  
-  Network ID: 192.168.1.128/26  
Khoảng IP được sd 192.168.1.129 -> 192.168.1.192.  
- Network ID: 192.168.1.192/26  
Khoảng IP được sd 192.168.1.193 -> 192.168.1.254

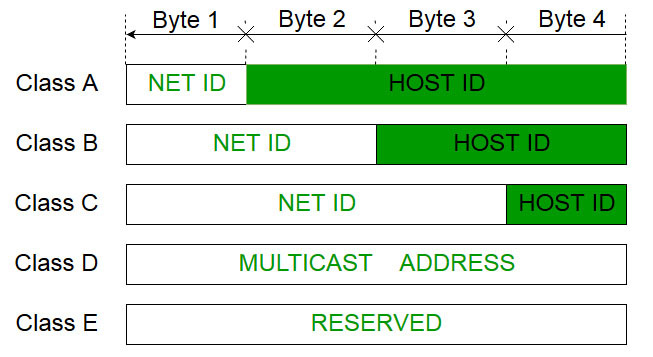
### 1.2.2 Kỹ thuật CIDR

* **CIDR** được viết tắt của **Classless Inter-Domain Routing,** nghĩa là định tuyến liên miền không phân lớp. Đây là phương pháp định tuyến trên Internet mà không dựa vào việc sử dụng các lớp địa chỉ IP mạng cố định như lớp A, B hay C. Thay vào đó mà ta có thể sử dụng Prefix Length để xác định phần mạng của địa chỉ IP. Từ đó mà ta có thể phân bổ và quản lý [địa chỉ IP](https://thietbimanggiare.com/dia-chi-ip/) một cách linh hoạt.
* A diagram of a network

  Description automatically generated

1. Tại sao CIDR ra đời?

* Trước khi CIDR xuất hiện, [Internet](https://thietbimanggiare.com/mang-internet-la-gi/) sử dụng hệ thống lớp địa chỉ IP (A, B, C). Các lớp mạng này quy định cố định số Octet được sử dụng cho phần mạng và phần host. Do đó, số địa chỉ IP có thể sử dụng trong từng lớp mạng là cố định:
  + Lớp A: Có phạm vi địa chỉ IP từ 1.0.0.0 đến 126.0.0.0.
  + Lớp B: Có phạm vi địa chỉ IP từ 128.0.0.0 đến 191.255.0.0.
  + Lớp C: Có phạm vi địa chỉ IP từ 192.0.0.0 đến 223.255.255.0.



1. Cấu trúc các lớp địa chỉ IP

* Sử dụng [các lớp địa chỉ IP](https://thietbimanggiare.com/cac-lop-dia-chi-ip/) tạo thành một nhược điểm trong việc phân bổ và quản lý địa chỉ IP. Đặc biệt khi Internet phát triển. Các lớp địa chỉ lớn lại tập trung dành cho hết các doanh nghiệp lớn, trong khi các doanh nghiệp vừa và nhỏ lại không có địa chỉ IP để sử dụng.
* Điều này dẫn tới việc lãng phí địa chỉ IP. Sự không đồng đều trong việc sử dụng các lớp mạng đã đẩy các nhà phát triển mạng tìm kiếm một phương pháp mới, linh hoạt hơn. Đó chính là lý do CIDR ra đời
* Với CIDR, các mạng được xác định bằng địa chỉ IP mạng cùng với một số lượng bit cụ thể xác định phần mạng của địa chỉ IP. Ví dụ, trong địa chỉ IP “192.168.1.0/24”, “24” đại diện cho số lượng bit trong phần mạng của địa chỉ IP. Điều này có nghĩa là 24 bit đầu tiên của địa chỉ IP được dành cho phần mạng, còn lại là phần host.
* CIDR cho phép các tổ chức có thể chia nhỏ hoặc kết hợp các phạm vi địa chỉ IP một cách linh hoạt hơn, giúp tối ưu hóa sử dụng không gian địa chỉ IP. Điều này làm giảm lãng phí và giúp tăng hiệu suất trong việc định tuyến và quản lý mạng Internet.

1. CIDR trong địa chỉ IP và Subnetting

Giờ ta sẽ hiểu sâu hơn vai trò của CIDR trong địa chỉ IP và chia [mạng con](https://thietbimanggiare.com/subnetting-mang-con/) (Subnetting)!

Ta đã biết địa chỉ IP có cấu trúc gồm 2 phần: Phần mạng và phần host.

A diagram of a number

Description automatically generated with medium confidence

* **Phần Mạng (**[**Network**](https://thietbimanggiare.com/network-la-gi/)**ID):**
  + Đây là một phần của địa chỉ IP dùng để xác định mạng mà thiết bị đó thuộc về.
  + Số lượng bit trong phần mạng được xác định bởi prefix length.
  + Prefix length là số bit liên tục từ trái sang phải của địa chỉ IP dùng để xác định mạng con.
  + Ví dụ, trong địa chỉ IP “192.168.1.0/24”, phần “192.168.1” là phần mạng, và “24” là prefix length, xác định rằng 24 bit đầu tiên của địa chỉ IP này được dành cho phần mạng.
* **Phần Host (Host ID):**
  + Đây là một phần của địa chỉ IP dùng để xác định một thiết bị cụ thể trong mạng đó.
  + Số lượng bit trong phần host được tính bằng cách trừ đi số bit dành cho phần mạng từ tổng số bit của địa chỉ IP (32 bit cho IPv4 và 128 bit cho IPv6).
  + Ví dụ, trong địa chỉ IP “192.168.1.10/24”, phần “10” là phần host, và các bit sau bit thứ 24 được dành cho phần host.
  + Subnetting là quá trình chia nhỏ một mạng lớn thành các mạng con nhỏ hơn để tối ưu hóa việc quản lý và sử dụng địa chỉ IP. Kỹ thuật này thường được sử dụng để phân bổ địa chỉ IP một cách hiệu quả và linh hoạt hơn.

Khi áp dụng CIDR, **prefix length** cũng chính là cách xác định số lượng bit dành cho phần mạng và phần host trong một địa chỉ IP. Bằng cách sử dụng prefix length, CIDR cho phép chúng ta biểu diễn một loạt các mạng con có kích thước khác nhau một cách linh hoạt và đơn giản. Điều này giúp tối ưu hóa sử dụng không gian địa chỉ IP và quản lý mạng một cách hiệu quả hơn.

1. Hiểu về Prefix Length và CIDR Notation

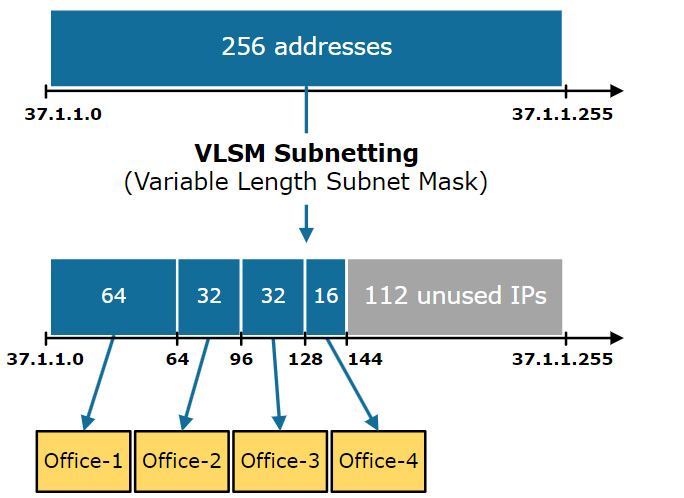
* Để biểu diễn phạm vi của một mạng IP trong CIDR, chúng ta sử dụng định dạng gọi là CIDR notation. Trong CIDR notation, mỗi mạng được biểu diễn bằng cách kết hợp địa chỉ IP mạng và prefix length, được phân tách bằng dấu gạch chéo (/).
* Prefix length xác định số lượng bit dành cho phần mạng trong địa chỉ IP. Ví dụ, trong địa chỉ IP “192.168.1.0/24”, phần “192.168.1.0” là địa chỉ IP mạng và “24” là prefix length, cho biết rằng 24 bit đầu tiên của địa chỉ IP này được dành cho phần mạng.
* Cách biểu diễn này cho phép chúng ta xác định một cách rõ ràng và tiện lợi phạm vi của một mạng IP và sử dụng nó trong việc định tuyến và quản lý mạng một cách hiệu quả.

1. CIDR trong định tuyến

* CIDR cho phép phân bổ địa chỉ IP một cách linh hoạt bằng cách sử dụng prefix length để xác định phạm vi của mỗi mạng. Các tổ chức có thể chia nhỏ hoặc kết hợp các phạm vi địa chỉ IP theo nhu cầu của họ mà không bị ràng buộc bởi các lớp mạng cố định như trong hệ thống lớp địa chỉ IP truyền thống.
* Giả sử có hai tổ chức A và B có mỗi tổ chức có một mạng con riêng của họ. Mạng của tổ chức A được gán địa chỉ IP từ 192.168.1.0 đến 192.168.1.255 và mạng của tổ chức B được gán địa chỉ IP từ 192.168.2.0 đến 192.168.2.255.
* Trước khi CIDR được sử dụng, mỗi mạng con này sẽ được biểu diễn bằng cách sử dụng các lớp mạng truyền thống và định rõ prefix length của mỗi lớp. Ví dụ, mạng của tổ chức A sẽ được xác định là lớp C với prefix length là /24 và mạng của tổ chức B cũng sẽ là lớp C với prefix length là /24.
* Tuy nhiên, khi sử dụng CIDR, cả hai mạng này có thể được biểu diễn một cách linh hoạt và hiệu quả hơn. Thay vì sử dụng prefix length /24 cho mỗi mạng, chúng ta có thể kết hợp chúng thành một phạm vi lớn hơn, ví dụ như /23. Điều này có nghĩa là cả hai mạng A và B đều có thể được biểu diễn bằng một địa chỉ IP mạng duy nhất là 192.168.0.0 với prefix length là /23.
* Khi một gói tin cần được gửi từ mạng của tổ chức A đến mạng của tổ chức B, các thiết bị định tuyến trên Internet sẽ sử dụng bảng định tuyến để quyết định đường đi tối ưu cho gói tin đó. Trong bảng định tuyến, phạm vi IP của mạng A và mạng B sẽ được biểu diễn bằng cùng một địa chỉ IP mạng (192.168.0.0/23), và các thiết bị định tuyến sẽ quyết định đường đi tối ưu dựa trên thông tin này.

### 1.2.3 Kỹ thuật VLSM

* VLSM (Variable Length Subnet Mask) là kỹ thuật chia mạng con trong đó các mạng con (subnet) của cùng một mạng ban đầu (major network) sau khi chia có chiều dài Subnet mask (số bit thuộc phần network) khác nhau.
* A diagram of a diagram

  Description automatically generatedTrong mô hình trên sử dụng subnet 172.16.14.0/24 chia thành các subnet nhỏ hơn (sub-subnet) với các subnet-mask /27 và /30.
* VLSM được sử dụng trong các giao thức định tuyến:
  + OSPF
  + IS-IS
  + EIGRP
  + RIP v2
  + Static route
* VLSM cho phép các quản trị viên mạng tạo mạng con với các Subnet Mask khác nhau. Trong ví dụ trên, VLSM có thể được sử dụng để gán mặt nạ mạng con 255.255.255.128 cho mạng hành chính (với 126 địa chỉ IP khả dụng) và 255.255.255.192 cho mạng marketing (với 62 địa chỉ IP có sẵn).
* Bằng cách này ta có thể thấy rõ ràng rằng việc sử dụng địa chỉ IP đã hiệu quả hơn mà vẫn tạo ra không gian để mở rộng thiết bị cho tương lai. 
* Cách dùng VLSM chia mạng con
* Giả sử chúng ta có một mạng lớn có địa chỉ IP là 192.168.10.0/24 và chúng ta cần phân chia nó thành các mạng con để phục vụ cho các phòng ban trong một công ty. Các yêu cầu về số lượng host trong mỗi phòng ban như sau:
  + Kinh doanh: 90 hosts
  + Phát triển: 30 hosts
  + Marketing: 15 hosts
  + IT: 10 hosts
* Bây giờ chúng ta sẽ thực hiện phân tích và áp dụng VLSM:
  + Tính toán số lượng host và subnet mask:
  + Kinh doanh: 90 hosts (2^7 – 2 = 126, với 7 bits dành cho host) => subnet mask /25 (255.255.255.128).
  + Phát triển: 30 hosts (2^5 – 2 = 30, với 5 bits dành cho host) => subnet mask /27 (255.255.255.224).
  + Marketing: 15 hosts (2^4 – 2 = 14, với 4 bits dành cho host) => subnet mask /28 (255.255.255.240).
  + IT: 10 hosts (2^4 – 2 = 14, với 4 bits dành cho host) => subnet mask /28 (255.255.255.240).
* Phân chia mạng gốc:
  + Kinh doanh: sử dụng subnet 192.168.10.0/25.
  + Phát triển: sử dụng subnet 192.168.10.128/27.
  + Marketing: sử dụng subnet 192.168.10.160/28.
  + IT: sử dụng subnet 192.168.10.176/28.
* Gán địa chỉ IP cho các thiết bị trong mạng: Bây giờ chúng ta có thể gán các địa chỉ IP tương ứng cho các thiết bị trong từng phòng ban, đảm bảo rằng mỗi thiết bị được gán một địa chỉ IP hợp lệ trong phạm vi mạng con tương ứng.
* Kiểm tra và triển khai: Kiểm tra kỹ lưỡng để đảm bảo rằng mọi thứ đều hoạt động đúng cách và triển khai cấu hình mạng được thiết kế bằng VLSM.

# CHƯƠNG 2: TÌM HIỂU VỀ KỸ THUẬT TRIỂN KHAI WEBSITE THÔNG QUA MÔI TRƯỜNG ĐIỆN TOÁN ĐÁM MÂY

## 2.1 Khái niệm Điện toán đám mây

* Điện toán đám mây (Cloud Computing), còn gọi là điện toán máy chủ ảo, mô hình cung cấp dịch vụ điện toán qua Internet, cho phép người dùng truy cập và sử dụng các tài nguyên công nghệ thông tin như máy chủ, lưu trữ, mạng, phần mềm, cơ sở dữ liệu,… mà không cần sở hữu hoặc quản lý hạ tầng vật lý. Nói cách khác, thay vì đầu tư xây dựng các trung tâm dữ liệu hoặc sử dụng một hay nhiều máy chủ vật lý (có thể nhìn thấy, chạm, nắm bằng tay), thì nay người dùng sẽ sử dụng các tài nguyên được ảo hoá thông qua môi trường Internet.
* Với các dịch vụ sẵn có trên Internet, doanh nghiệp không phải mua và duy trì hàng trăm, thậm chí hàng nghìn máy tính cũng như phần mềm. Họ chỉ cần tập trung phát triển app, phần mềm bởi đã có đơn vị cung cấp dịch vụ Cloud lo cơ sở hạ tầng và công nghệ thay họ. Bạn có thể truy cập đến bất kỳ tài nguyên nào tồn tại trong “đám mây (cloud)” tại bất kỳ thời điểm nào và từ bất kỳ đâu thông qua hệ thống Internet.
* Google Drive, Dropbox, OneDrive, iCloud,… là những ví dụ điển hình của dịch vụ điện toán đám mây (Cloud Computing). Người dùng chỉ cần đăng ký tài khoản và sử dụng dịch vụ miễn phí và trả phí theo nhu cầu của bản thân. Các tài liệu được lưu trên tài khoản “đám mây” của mình và truy cập vào sử dụng từ bất cứ vị trí nào miễn có kết nối mạng.
* Để hiểu rõ hơn về sự khác biệt giữa hình thức lưu trữ tại chỗ truyền thống (On-Premise) và lưu trữ đám mây (Cloud Computing), bạn có thể tham khảo phần so sánh giữa cách hoạt động của hai hình thức này dưới đây:
  + Lưu trữ truyền thống: On-Premise là hình thức lưu trữ phụ thuộc vào các thiết bị vật lý. Với hình thức này, doanh nghiệp cần cài đặt và chạy phần mềm trên thiết bị vật lý dựa trên khả năng kiểm soát của mình. Nhân viên có thể trực tiếp truy cập dữ liệu bằng thiết bị vật lý, kiểm soát và quản lý mọi thứ mà không có sự tham gia của bên thứ ba.
  + Lưu trữ đám mây: Là giải pháp lưu trữ dữ liệu trên mạng internet. Theo đó, người dùng sẽ được cung cấp tài nguyên hệ thống theo yêu cầu mà không cần đầu tư thiết bị hay phần cứng để lưu trữ. Mọi khách hàng đều có thể truy cập dữ liệu của mình thông qua tài khoản đã đăng ký mọi lúc, mọi nơi, không giới hạn thiết bị. Khi không còn nhu cầu sử dụng thì chỉ cần ngừng đăng ký dịch vụ là được.



## 2.2 Mô hình điện toán đám mây

* Dựa theo phương pháp triển khai,có 4 mô hình điện toán đám mây như sau:

### 2.2.1 Public Cloud

* Public Cloud là mô hình điện toán đám mây được dùng phổ biến nhất hiện nay, cho phép chứa tất cả dịch vụ, ứng dụng trên cùng một hệ thống đám mây. Theo đó, toàn bộ người dùng sẽ được dùng chung tài nguyên và nhà cung cấp sẽ có nhiệm vụ quản lý và bảo vệ tài nguyên đó.
* Khi dùng Public Cloud, người dùng sẽ không bị giới hạn thời gian và không gian lưu trữ, có thể linh động mở rộng/thu hẹp không gian tùy theo nhu cầu. Đặc biệt, dịch vụ này có thể đáp ứng tốt nhu cầu của nhiều người dùng khác nhau và chi phí thấp. Do dùng chung nên vấn đề bảo mật chưa quá cao và khó kiểm soát được dữ liệu.

### 2.2.2 Private Cloud

* Private Cloud (Đám mây riêng) là một môi trường điện toán đám mây được xây dựng và sử dụng riêng biệt bởi một tổ chức. Nó cung cấp các dịch vụ và cơ sở hạ tầng đám mây được quản lý nội bộ hoặc thông qua một nhà cung cấp dịch vụ bên thứ ba, và được truy cập qua mạng riêng (không công khai trên internet). Private Cloud cung cấp các lợi ích của đám mây như tính linh hoạt, khả năng mở rộng và tính linh hoạt trong việc sử dụng tài nguyên, nhưng đồng thời cung cấp mức độ kiểm soát và bảo mật cao hơn so với Public Cloud (Đám mây công cộng).

### 2.2.3 Hybrid Cloud

* Hybrid Cloud (Đám mây lai) là một mô hình triển khai đám mây kết hợp giữa Public Cloud (Đám mây công cộng) và Private Cloud (Đám mây riêng), hoặc các loại hình đám mây khác, nhằm tối ưu hóa khả năng quản lý, bảo mật, và chi phí của tổ chức. Mô hình này cho phép tổ chức tận dụng lợi ích của cả đám mây công cộng và đám mây riêng, cũng như linh hoạt trong việc di chuyển khối lượng công việc và dữ liệu giữa các môi trường.

### 2.2.4 Community Cloud

* Phục vụ các doanh nghiệp và cá nhân có nhu cầu chia sẻ hạ tầng và dữ liệu cho những doanh nghiệp và cá nhân khác một cách nhanh chóng.



## 2.3 Các loại hình dịch vụ điện toán đám mây phổ biến: SaaS, PaaS và IaaS

### 2.3.1 SaaS – Phần mềm dưới dạng dịch vụ

* Trong các loại dịch vụ điện toán đám mây, mô hình SaaS chính là mô hình phổ biến nhất hiện nay, có thể sử dụng rộng rãi cho mọi đối tượng.
* SaaS là viết tắt của thuật ngữ “Software as a Service”, là mô hình dịch vụ cung cấp cho người dùng quyền truy cập vào một ứng dụng hoặc phần mềm được xây dựng hoàn chỉnh. Chúng có đầy đủ chức năng và tác vụ đáp ứng các nhu cầu của người dùng (phần lớn là người dùng cuối) và giúp họ giải quyết các vấn đề cụ thể.
* Các ứng dụng SaaS được cung cấp tới người dùng dựa trên nền tảng điện toán đám mây theo một trong các dạng:
  + Ứng dụng trên máy tính.
  + Ứng dụng cho thiết bị di động.
  + Tiện ích trên trình duyệt web.



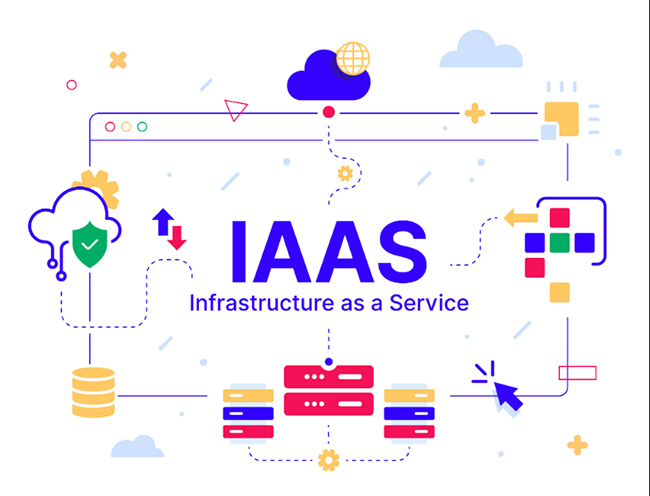
### 2.3.2 PaaS – Nền tảng dưới dạng dịch vụ

* Trong khi SaaS cung cấp một giải pháp phần mềm ứng dụng hoàn chỉnh, thì PaaS mang lại các công cụ để xây dựng, phát triển các phần mềm, ứng dụng đó mà không cần phải lo về tài nguyên máy tính.
* PaaS là viết tắt của thuật ngữ “Platform as a Service” – “Nền tảng dưới dạng dịch vụ”. Trong khi SaaS là sản phẩm hoàn chỉnh giúp giải quyết trực tiếp các vấn đề của người dùng, thì PaaS chính là một hệ sinh thái, cung cấp môi trường hoàn chỉnh để người dùng tự thiết kế, tạo dựng, phát triển, thử nghiệm, triển khai và lưu trữ các sản phẩm đó.
* Với mô hình PaaS, người dùng được nhà cung cấp trao quyền truy cập và sử dụng một “bộ kit” bao gồm:
  + Quyền truy cập máy chủ và bộ lưu trữ.
  + Cơ sở dữ liệu.
  + Công cụ phát triển: Trình chỉnh sửa mã nguồn, trình gỡ lỗi, trình biên dịch, các công cụ hỗ trợ viết/ triển khai/ gỡ lỗi/ quản lý mã…
  + Hệ điều hành, API trung gian,…

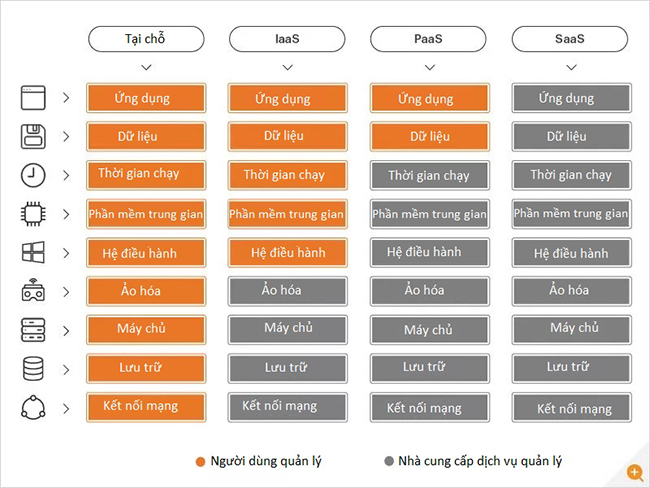


### 2.3.3 IaaS – Cơ sở hạ tầng dưới dạng dịch vụ

* Trong khi SaaS là mô hình dễ sử dụng nhất, thì IaaS là phương án tốt nhất để người dùng kiểm soát tối đa môi trường công nghệ và tự tạo sản phẩm riêng với mức chi phí tiết kiệm hơn phần mềm tại chỗ truyền thống.
* IaaS là viết tắt của thuật ngữ “Infrastructure as a Service”, là mô hình dịch vụ tạo nền tảng để triển khai công nghệ điện toán đám mây. Thông qua nhà cung cấp IaaS, người dùng có quyền truy cập qua internet để vào các tài nguyên CNTT phần cứng và cốt lõi gồm:
  + Máy chủ ảo hoặc máy chủ chuyên dụng chạy trên nền máy tính vật lý.
  + Các dịch vụ kết nối mạng.
  + Trung tâm lưu trữ dữ liệu (Data center).



* So sánh các loại hình dịch vụ của điện toán đám mây
* Việc so sánh và nắm bắt sự khác biệt của ba loại mô hình SaaS, PaaS và IaaS sẽ giúp khách hàng lựa chọn được bộ dịch vụ phù hợp nhất. Đặc biệt là với các cơ quan, doanh nghiệp, vì mỗi đơn vị đều có quy mô, định hướng, nguồn lực, thế mạnh và đối mặt với những vấn đề thách thức khác nhau, vì vậy việc áp dụng mô hình đám mây phù hợp là điều rất quan trọng.
* Sự khác biệt lớn nhất giữa IaaS, PaaS và SaaS là mức độ trách nhiệm của nhà cung cấp là nhiều hay ít.
* Giải thích theo cách dễ hiểu nhất, IaaS tương tự như việc thuê một căn hộ trống nội thất, chỉ có các tiện ích đi kèm với khu dân cư đó là có sẵn.
* Trong khi đó, PaaS tương tự như việc thuê một căn hộ có nội thất cơ bản, nhưng người thuê vẫn phải sắm sửa các đồ gia dụng và đồ dùng sinh hoạt hàng ngày.
* Và cuối cùng, việc sử dụng SaaS có thể hiểu như việc thuê một căn hộ đầy đủ mọi tiện nghi.



## 2.4 Triển khai Website trên Môi Trường Điện Toán Đám Mây

* Triển khai website trên nền tảng điện toán đám mây đang ngày càng trở nên phổ biến nhờ vào tính linh hoạt, hiệu quả và khả năng mở rộng mà nó mang lại. Thay vì đầu tư vào các máy chủ vật lý và cơ sở hạ tầng riêng, doanh nghiệp có thể thuê các tài nguyên máy tính từ các nhà cung cấp dịch vụ đám mây như AWS, Azure, GCP.

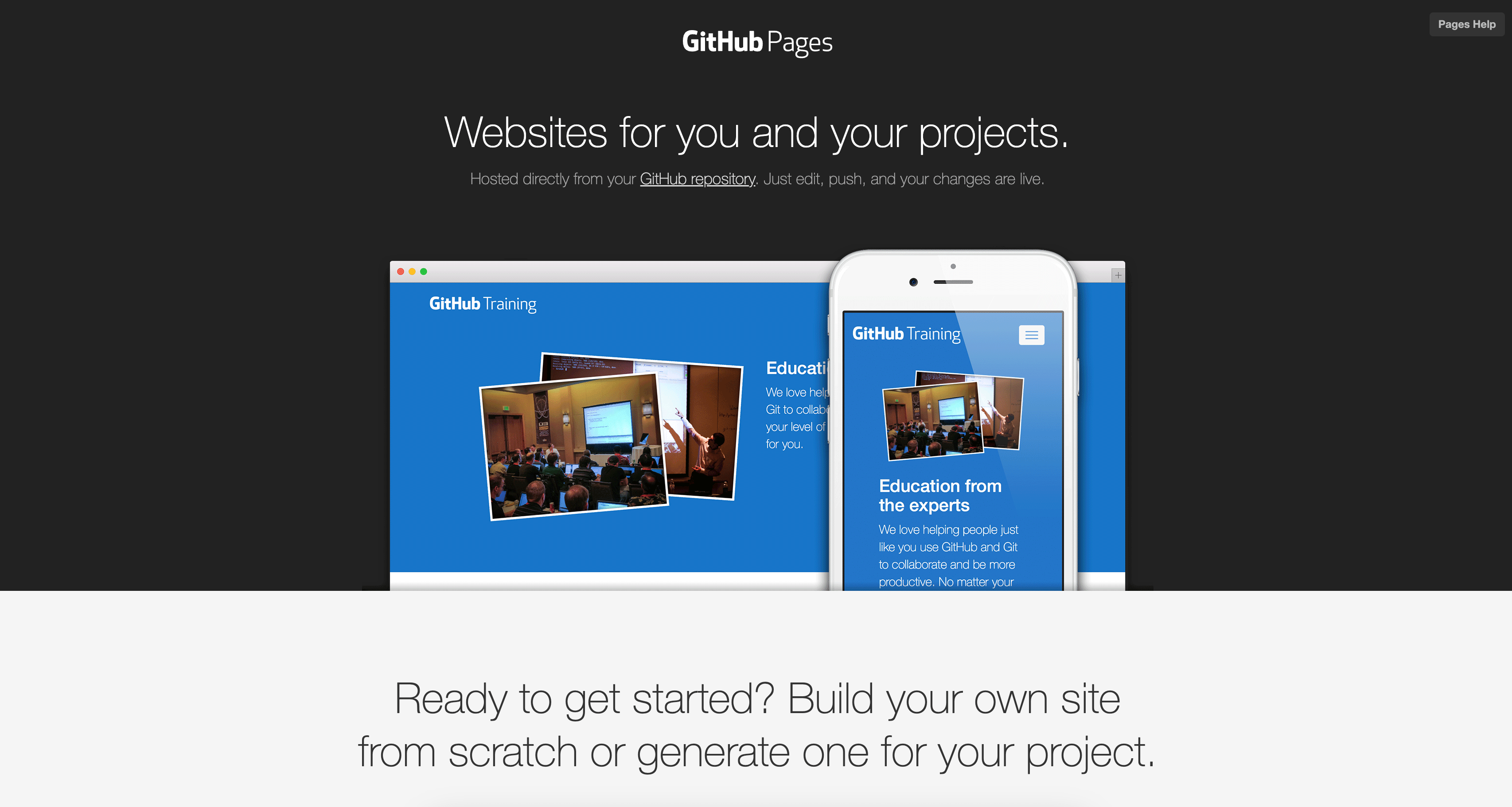
Các bước triển khai chi tiết

1. Lựa chọn nền tảng đám mây:
   * + AWS (Amazon Web Services): Nền tảng đám mây lớn nhất và toàn diện nhất, cung cấp đa dạng các dịch vụ.
     + Azure (Microsoft Azure): Tích hợp chặt chẽ với các sản phẩm của Microsoft, phù hợp cho các doanh nghiệp sử dụng hệ sinh thái Microsoft.
     + GCP (Google Cloud Platform): Nổi bật với khả năng xử lý dữ liệu lớn và trí tuệ nhân tạo.
     + Các nền tảng khác: Alibaba Cloud, DigitalOcean, ...
2. Lựa chọn dịch vụ:
   * + IaaS (Infrastructure as a Service): Cung cấp các tài nguyên hạ tầng cơ bản như máy chủ ảo, lưu trữ, mạng.
     + PaaS (Platform as a Service): Cung cấp môi trường phát triển và triển khai ứng dụng.
     + SaaS (Software as a Service): Cung cấp các ứng dụng sẵn sàng sử dụng.
3. Tạo môi trường:
   * + Tạo instance: Tạo một máy chủ ảo để chạy website.
     + Cài đặt hệ điều hành: Cài đặt hệ điều hành phù hợp (Ubuntu, CentOS, ...).
     + Cài đặt phần mềm: Cài đặt web server (Apache, Nginx), database (MySQL, PostgreSQL), ngôn ngữ lập trình (PHP, Python, ...).
4. Triển khai ứng dụng:
   * + Upload code: Upload code ứng dụng lên máy chủ.
     + Cấu hình: Cấu hình web server, database để kết nối với ứng dụng.
     + Kiểm tra: Kiểm tra xem ứng dụng có hoạt động đúng không.
5. Quản lý và bảo trì:
   * + Theo dõi: Theo dõi hiệu năng của website, sử dụng các công cụ giám sát.
     + Bảo mật: Đảm bảo an toàn cho website bằng cách cài đặt firewall, cập nhật phần mềm thường xuyên.
     + Sao lưu: Thực hiện sao lưu dữ liệu định kỳ.
6. Lợi ích của việc triển khai website trên đám mây
   * + Linh hoạt: Dễ dàng mở rộng hoặc thu nhỏ quy mô tài nguyên.
     + Tiết kiệm chi phí: Chỉ trả tiền cho những gì mình sử dụng.
     + Khả năng mở rộng: Khả năng xử lý lượng truy cập lớn.
     + Bảo mật: Các nhà cung cấp đám mây thường cung cấp các tính năng bảo mật cao.
     + Truy cập từ xa: Có thể quản lý website từ bất kỳ đâu có kết nối internet.

# CHƯƠNG 3: MINH HỌA CHƯƠNG TRÌNH CHIA MẠNG CON TRÊN WEBSITE

## 3.1 Github Pages là gì ?

* GitHub Pages là một dịch vụ cung cấp bởi GitHub để tạo trang web từ các kho lưu trữ Git. Nó cho phép người dùng tận dụng sức mạnh của Git để quản lý mã nguồn và cung cấp một cách nhanh chóng để triển khai trang web của họ. GitHub Pages hỗ trợ nhiều ngôn ngữ lập trình và cung cấp nhiều chủ đề mẫu cho người dùng lựa chọn.
* Lợi ích của việc sử dụng GitHub Pages[​](https://www.thanhnamnguyen.dev/blog/deploy-project-len-github-pages#loi-ich)
  + GitHub Pages là một dịch vụ miễn phí và dễ dàng triển khai.
  + Hỗ trợ nhiều ngôn ngữ lập trình: GitHub Pages không giới hạn ngôn ngữ lập trình. Bạn có thể sử dụng HTML, CSS, JavaScript, hoặc bất kỳ ngôn ngữ khác mà bạn thích.
  + Tích hợp dễ dàng với GitHub: GitHub Pages tích hợp chặt chẽ với GitHub, giúp bạn quản lý mã nguồn và trang web của mình tại cùng một nơi.



## 3.2 Chuẩn bị mã nguồn cho ứng dụng chia mạng con

* Tạo các tệp cần thiết
  + index.html: Tệp HTML chính sẽ chứa mã HTML và liên kết đến tệp JavaScript.
  + script.js: Tệp JavaScript sẽ chứa mã tính toán địa chỉ IP mà bạn đã viết.

### 3.2.1 Tệp index.html

<!DOCTYPE html>

<html lang="vi">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Máy Tính IP</title>

    <style>

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: #e9ecef;

            margin: 0;

            padding: 20px;

            display: flex;

            justify-content: center;

            align-items: flex-start;

            height: 100vh;

        }

        .container {

            display: flex;

            width: 100%;

            max-width: 1200px;

            background: #ffffff;

            padding: 20px;

            border-radius: 10px;

            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

        }

        .guide {

            flex: 1;

            max-width: 300px;

            margin-right: 20px;

        }

        .guide h2 {

            color: #333;

            font-size: 20px;

            margin-bottom: 15px;

        }

        .guide p {

            font-size: 14px;

            color: #555;

            line-height: 1.6;

        }

        .form-container {

            flex: 2;

            width: 100%;

            display: flex;

            flex-direction: column;

        }

        h1 {

            color: #333;

            font-size: 24px;

            margin-bottom: 20px;

        }

        .form-group {

            margin-bottom: 10px;

        }

        .form-group label {

            display: block;

            margin-bottom: 5px;

            font-weight: 600;

            color: #555;

        }

        .form-group input[type="text"],

        .form-group select {

            width: 100%;

            padding: 10px;

            font-size: 16px;

            border: 1px solid #ced4da;

            border-radius: 5px;

            box-sizing: border-box;

            background-color: #f8f9fa;

        }

        .form-group input[type="text"]:focus,

        .form-group select:focus {

            border-color: #80bdff;

            outline: none;

            background-color: #ffffff;

        }

        .btn {

            padding: 12px;

            font-size: 16px;

            background-color: #007bff;

            color: white;

            border: none;

            border-radius: 5px;

            cursor: pointer;

            transition: background-color 0.3s ease;

        }

        .btn:hover {

            background-color: #0056b3;

        }

        .result-label {

            font-size: 14px;

            color: #333;

        }

        .result-label span {

            font-weight: 500;

        }

        .oktett-container {

            display: flex;

            align-items: center;

            gap: 5px;

        }

        .oktett-container input {

            width: 60px;

            text-align: center;

        }

        .separator {

            width: 2px;

            background-color: #dee2e6;

            margin: 0 20px;

        }

        .footer-divider {

            width: 100%;

            height: 2px;

            background-color: #dee2e6;

            margin: 20px 0;

        }

        .footer {

            text-align: center;

            font-size: 16px;

            color: #333;

            background-color: #f8f9fa;

            padding: 20px;

            border-radius: 10px;

            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

        }

        .footer p {

            margin: 10px 0;

        }

        .footer p:first-child {

            font-size: 18px;

            font-weight: bold;

            color: #007bff;

        }

        .footer a {

            color: #007bff;

            text-decoration: none;

        }

        .footer a:hover {

            text-decoration: underline;

        }

    </style>

</head>

<body>

    <div class="container">

        <!-- Phần hướng dẫn sử dụng -->

        <div class="guide">

            <h2>Hướng Dẫn Sử Dụng Ứng Dụng Tính Toán IP</h2>

            <p><strong>Mục Đích:</strong></p>

            <p>Ứng dụng này giúp bạn tính toán các thông tin liên quan đến địa chỉ IP và subnet mask, bao gồm địa chỉ IP nhị phân, subnet mask, địa chỉ mạng, địa chỉ broadcast, phạm vi địa chỉ IP, và số lượng địa chỉ IP hợp lệ.</p>

            <p><strong>Các Trường Nhập Liệu:</strong></p>

            <p><strong>Octet 1, Octet 2, Octet 3, Octet 4:</strong><br>

            Nhập từng phần của địa chỉ IP (mỗi phần gọi là oktett). Địa chỉ IP có định dạng IPv4 bao gồm bốn oktett, mỗi oktett có giá trị từ 0 đến 255.<br>

            Ví dụ: Để nhập địa chỉ IP 192.168.1.1, bạn sẽ nhập 192 vào Octet 1, 168 vào Octet 2, 1 vào Octet 3, và 1 vào Octet 4.</p>

            <p><strong>Giá trị CIDR:</strong><br>

            CIDR (Classless Inter-Domain Routing) là giá trị số đại diện cho kích thước của subnet mask. Ví dụ, CIDR 24 tương ứng với subnet mask 255.255.255.0.<br>

            Chọn giá trị CIDR từ menu thả xuống. Giá trị CIDR có thể từ 1 đến 32.</p>

            <p><strong>Các Kết Quả Tính Toán:</strong></p>

            <p><strong>Địa chỉ IP nhị phân:</strong><br>

            Hiển thị địa chỉ IP của bạn dưới dạng nhị phân. Mỗi oktett sẽ được chuyển đổi thành chuỗi nhị phân 8 bit.</p>

            <p><strong>Subnet Mask:</strong><br>

            Hiển thị subnet mask tương ứng với giá trị CIDR đã chọn.</p>

            <p><strong>Subnet Mask nhị phân:</strong><br>

            Hiển thị subnet mask dưới dạng nhị phân.</p>

            <p><strong>Lớp địa chỉ IP:</strong><br>

            Xác định lớp của địa chỉ IP (Class A, B, C, D, hoặc E) dựa trên giá trị của oktett đầu tiên.</p>

            <p><strong>Địa chỉ mạng:</strong><br>

            Tính toán địa chỉ mạng dựa trên địa chỉ IP và subnet mask.</p>

            <p><strong>Địa chỉ broadcast:</strong><br>

            Tính toán địa chỉ broadcast của subnet. Địa chỉ này được sử dụng để gửi dữ liệu đến tất cả các thiết bị trong subnet.</p>

            <p><strong>Phạm vi địa chỉ IP:</strong><br>

            Hiển thị phạm vi địa chỉ IP hợp lệ trong subnet. Phạm vi bao gồm địa chỉ IP đầu tiên (địa chỉ mạng + 1) đến địa chỉ IP cuối cùng (địa chỉ broadcast - 1).</p>

            <p><strong>Số lượng địa chỉ IP hợp lệ:</strong><br>

            Hiển thị số lượng địa chỉ IP hợp lệ trong subnet (số lượng IP có thể sử dụng cho các thiết bị).</p>

            <p><strong>Cách Sử Dụng:</strong></p>

            <p>Nhập địa chỉ IP vào các trường nhập liệu Octet 1 đến Octet 4.<br>

            Chọn giá trị CIDR từ menu thả xuống.<br>

            Nhấn nút "Tính" để thực hiện các tính toán.<br>

            Xem các kết quả tính toán hiển thị bên dưới nút "Tính".</p>

            <p><strong>Lưu Ý:</strong></p>

            <p>Đảm bảo rằng các giá trị nhập vào đều nằm trong phạm vi hợp lệ (0-255) cho các oktett và CIDR từ 1 đến 32.<br>

            Nếu CIDR là 31 hoặc 32, phạm vi địa chỉ IP sẽ không có vì đây là subnet cho các liên kết điểm-điểm.</p>

        </div>

        <!-- Thanh chắn ngang giữa các phần -->

        <div class="separator"></div>

        <!-- Phần form tính toán -->

        <div class="form-container">

            <h1>Nhập địa chỉ IP</h1>

            <div class="oktett-container">

                <div class="form-group">

                    <label for="octet1">Octet 1:</label>

                    <input type="text" id="octet1" maxlength="3">

                </div>

                <div class="form-group">

                    <label for="octet2">Octet 2:</label>

                    <input type="text" id="octet2" maxlength="3">

                </div>

                <div class="form-group">

                    <label for="octet3">Octet 3:</label>

                    <input type="text" id="octet3" maxlength="3">

                </div>

                <div class="form-group">

                    <label for="octet4">Octet 4:</label>

                    <input type="text" id="octet4" maxlength="3">

                </div>

            </div>

            <div class="form-group">

                <label for="cidr">Giá trị CIDR:</label>

                <select id="cidr">

                    <option value="1">1</option>

                    <option value="2">2</option>

                    <option value="3">3</option>

                    <option value="4">4</option>

                    <option value="5">5</option>

                    <option value="6">6</option>

                    <option value="7">7</option>

                    <option value="8">8</option>

                    <option value="9">9</option>

                    <option value="10">10</option>

                    <option value="11">11</option>

                    <option value="12">12</option>

                    <option value="13">13</option>

                    <option value="14">14</option>

                    <option value="15">15</option>

                    <option value="16">16</option>

                    <option value="17">17</option>

                    <option value="18">18</option>

                    <option value="19">19</option>

                    <option value="20">20</option>

                    <option value="21">21</option>

                    <option value="22">22</option>

                    <option value="23">23</option>

                    <option value="24">24</option>

                    <option value="25">25</option>

                    <option value="26">26</option>

                    <option value="27">27</option>

                    <option value="28">28</option>

                    <option value="29">29</option>

                    <option value="30">30</option>

                    <option value="31">31</option>

                    <option value="32">32</option>

                </select>

            </div>

            <!-- Calculate button -->

            <button class="btn" id="calculateBtn">Tính</button>

            <!-- Result labels -->

            <div class="form-group result-label">

                <label>Địa chỉ IP nhị phân:</label>

                <span id="binarisIPLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Subnet mask:</label>

                <span id="maszkLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Subnet mask nhị phân:</label>

                <span id="binarisMaszkLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Lớp địa chỉ IP:</label>

                <span id="cimosztalyLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Địa chỉ mạng:</label>

                <span id="halozatiCimLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Địa chỉ broadcast:</label>

                <span id="broadcastLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Phạm vi địa chỉ IP:</label>

                <span id="ipCimtartomanyLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Số lượng địa chỉ IP hợp lệ:</label>

                <span id="ipCimekDbLabel"></span>

            </div>

            <div class="form-group result-label">

                <label>Các mạng con:</label>

                <ul id="subnetworksLabel">

                    <!-- Các mạng con sẽ được hiển thị tại đây -->

                </ul>

            </div>

            <!-- Thanh chắn ngang phía dưới cùng -->

            <div class="footer-divider"></div>

            <!-- Chú thích tác giả -->

            <div class="footer">

                <p>Hồ Trung Phước - 16/06/2003</p>

                <p>20/08/2024 - Máy tính IP</p>

                <p>Đồ Án Tốt nghiệp</p>

                <p><a href="https://drive.google.com/file/d/1soLa6dacibwWMfK1\_4MlBzppPbwJnu-t/view?usp=drive\_link" download>Tải ứng dụng tại đây</a></p>

                <p>Vui lòng không reup dưới mọi hình thức!</p>

            </div>

        </div>

    </div>

    <script src="script.js"></script>

</body>

</html>

### 3.2.2 Giải thích cấu trúc của tệp index.html

1. **Khai báo HTML cơ bản:**

<!DOCTYPE html>

<html lang="vi">

Khai báo này cho trình duyệt biết tài liệu này là một tài liệu HTML5 và sử dụng ngôn ngữ tiếng Việt (lang="vi").

1. **Phần <head>:**

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Máy Tính IP</title>

...

</head>

* + meta charset="UTF-8": Thiết lập mã hóa ký tự là UTF-8 để hỗ trợ tiếng Việt.
  + meta name="viewport": Thiết lập để trang web hiển thị tốt trên các thiết bị di động.
  + <title>: Tiêu đề trang hiển thị trên tab của trình duyệt là "Máy Tính IP".

1. **Phần <style>:**

<style>

body { ... }

.container { ... }

.guide { ... }

...

</style>

Phần này chứa các định dạng CSS cho trang web. Các lớp CSS như .container, .guide, .form-container, .form-group, .btn, v.v. được định nghĩa để tạo bố cục và thiết kế cho trang.

* + **Cấu trúc bố cục:**
    - .container: Là phần chứa chính, chứa tất cả nội dung của trang.
    - .guide: Là phần bên trái, cung cấp hướng dẫn sử dụng.
    - .separator: Là thanh ngang phân chia giữa phần hướng dẫn và phần tính toán.
    - .form-container: Là phần bên phải, chứa các biểu mẫu nhập liệu và các kết quả tính toán.
  + **Phong cách và giao diện người dùng:**
    - body: Đặt kiểu chữ, nền, và căn giữa nội dung.
    - .btn: Định dạng cho nút "Tính".
    - .footer: Định dạng cho phần chú thích tác giả ở cuối trang.

1. **Phần <body>:**
   * **Phần chứa chính (.container):**

<div class="container">

...

</div>

Đây là phần chứa tất cả nội dung của trang web bao gồm cả hướng dẫn sử dụng và phần tính toán.

* + **Hướng dẫn sử dụng (.guide):**

<div class="guide">

<h2>Hướng Dẫn Sử Dụng Ứng Dụng Tính Toán IP</h2>

<p><strong>Mục Đích:</strong></p>

...

</div>

Phần này cung cấp thông tin chi tiết về cách sử dụng ứng dụng tính toán IP, giải thích từng trường nhập liệu và kết quả đầu ra.

* + **Phần form tính toán (.form-container):**

<div class="form-container">

<h1>Nhập địa chỉ IP</h1>

<div class="oktett-container">

...

</div>

<div class="form-group">

...

</div>

<button class="btn" id="calculateBtn">Tính</button>

...

</div>

Phần này chứa các trường nhập liệu để người dùng nhập địa chỉ IP (bằng cách chia thành 4 octet) và chọn giá trị CIDR. Nó cũng chứa nút "Tính" để bắt đầu quá trình tính toán và hiển thị các kết quả đầu ra.

* + **Chú thích tác giả (.footer):**

<div class="footer">

<p>Hồ Trung Phước - 16/06/2003</p>

<p>20/08/2024 - Máy tính IP</p>

...

</div>

Phần này cung cấp thông tin về tác giả và cung cấp liên kết tải ứng dụng.

1. **Phần <script>:**

<script src="script.js"></script>

Dòng này nhúng tệp JavaScript script.js để xử lý logic tính toán các địa chỉ IP. Tệp này có thể chứa các chức năng JavaScript để thực hiện các tính toán khi người dùng nhấn nút "Tính".

### 3.2.3 Tệp script.js

// Hàm chuyển đổi số nguyên oktett thành chuỗi nhị phân 8 bit

function oktettToBinary(oktett) {

    return oktett.toString(2).padStart(8, '0');

}

// Hàm chuyển đổi nhị phân thành số nguyên

function binaryToDecimal(binaryAddress) {

    return binaryAddress.split('.').map(octet => parseInt(octet, 2));

}

// Hàm tính toán thông tin địa chỉ IP và subnet mask

function calculate() {

    // Lấy giá trị từ các trường nhập liệu

    let oktettIP1 = parseInt(document.getElementById('octet1').value);

    let oktettIP2 = parseInt(document.getElementById('octet2').value);

    let oktettIP3 = parseInt(document.getElementById('octet3').value);

    let oktettIP4 = parseInt(document.getElementById('octet4').value);

    let cidr = parseInt(document.getElementById('cidr').value);

    if (isNaN(oktettIP1) || isNaN(oktettIP2) || isNaN(oktettIP3) || isNaN(oktettIP4) || isNaN(cidr)) {

        alert('Vui lòng nhập đầy đủ và chính xác các giá trị.');

        return;

    }

    // Địa chỉ IP dưới dạng nhị phân

    let binaryIP = [oktettIP1, oktettIP2, oktettIP3, oktettIP4].map(oktettToBinary).join('.');

    document.getElementById('binarisIPLabel').textContent = "Binary IP: " + binaryIP;

    // Xác định subnet mask

    let mask = calculateSubnetMask(cidr);

    document.getElementById('maszkLabel').textContent = "Subnet Mask: " + mask;

    // Subnet mask dưới dạng nhị phân

    let binaryMask = mask.split('.').map(octet => oktettToBinary(parseInt(octet))).join('.');

    document.getElementById('binarisMaszkLabel').textContent = "Binary Subnet Mask: " + binaryMask;

    // Xác định địa chỉ mạng

    let networkAddress = calculateNetworkAddress(oktettIP1, oktettIP2, oktettIP3, oktettIP4, mask);

    document.getElementById('halozatiCimLabel').textContent = "Network Address: " + networkAddress;

    // Xác định địa chỉ broadcast

    let broadcastAddress = calculateBroadcastAddress(networkAddress, mask);

    document.getElementById('broadcastLabel').textContent = "Broadcast Address: " + broadcastAddress;

    // Xác định phạm vi địa chỉ IP

    let ipRange = calculateIPRange(networkAddress, broadcastAddress, cidr);

    document.getElementById('ipCimtartomanyLabel').textContent = "IP Range: " + ipRange;

    // Xác định số lượng địa chỉ IP hợp lệ

    let ipCount = calculateValidIPCount(cidr);

    document.getElementById('ipCimekDbLabel').textContent = "Number of IPs: " + ipCount.toLocaleString() + " địa chỉ IP";

    // Tính toán và hiển thị tất cả các mạng con

    let subnetworks = calculateSubnets(oktettIP1, oktettIP2, oktettIP3, oktettIP4, cidr);

    let subnetResults = subnetworks.map((subnet, index) => {

        let network = subnet.network.join('.');

        let broadcast = subnet.broadcast.join('.');

        let rangeStart = subnet.rangeStart.join('.');

        let rangeEnd = subnet.rangeEnd.join('.');

        return `<li>Mạng con ${index + 1}: ${network}, Broadcast: ${broadcast}, Phạm vi: ${rangeStart} - ${rangeEnd}</li>`;

    }).join('');

    document.getElementById('subnetworksLabel').innerHTML = `<ul>${subnetResults}</ul>`;

    // Xác định lớp IP

    let cimosztalyLabel = document.getElementById('cimosztalyLabel');

    if (oktettIP1 >= 1 && oktettIP1 <= 127) cimosztalyLabel.textContent = "Class A";

    else if (oktettIP1 >= 128 && oktettIP1 <= 191) cimosztalyLabel.textContent = "Class B";

    else if (oktettIP1 >= 192 && oktettIP1 <= 223) cimosztalyLabel.textContent = "Class C";

    else if (oktettIP1 >= 224 && oktettIP1 <= 239) cimosztalyLabel.textContent = "Class D";

    else if (oktettIP1 >= 240 && oktettIP1 <= 255) cimosztalyLabel.textContent = "Class E";

}

// Hàm tính toán subnet mask

function calculateSubnetMask(cidr) {

    const mask = (0xFFFFFFFF << (32 - cidr)) >>> 0;

    return [

        (mask >>> 24) & 0xFF,

        (mask >>> 16) & 0xFF,

        (mask >>> 8) & 0xFF,

        mask & 0xFF

    ].join('.');

}

// Hàm tính toán địa chỉ mạng

function calculateNetworkAddress(ip1, ip2, ip3, ip4, mask) {

    let maskArr = mask.split('.').map(Number);

    return [

        ip1 & maskArr[0],

        ip2 & maskArr[1],

        ip3 & maskArr[2],

        ip4 & maskArr[3]

    ].join('.');

}

// Hàm tính toán địa chỉ broadcast

function calculateBroadcastAddress(networkAddress, mask) {

    let networkArr = networkAddress.split('.').map(Number);

    let maskArr = mask.split('.').map(Number);

    let invertedMask = maskArr.map(octet => 255 - octet);

    return [

        networkArr[0] | invertedMask[0],

        networkArr[1] | invertedMask[1],

        networkArr[2] | invertedMask[2],

        networkArr[3] | invertedMask[3]

    ].join('.');

}

// Hàm tính phạm vi địa chỉ IP

function calculateIPRange(networkAddress, broadcastAddress, cidr) {

    if (cidr === 31 || cidr === 32) return "―";

    let [startIP1, startIP2, startIP3, startIP4] = networkAddress.split('.').map(Number);

    startIP4 += 1;

    let [endIP1, endIP2, endIP3, endIP4] = broadcastAddress.split('.').map(Number);

    endIP4 -= 1;

    return `${startIP1}.${startIP2}.${startIP3}.${startIP4} - ${endIP1}.${endIP2}.${endIP3}.${endIP4}`;

}

// Hàm tính số lượng địa chỉ IP hợp lệ

function calculateValidIPCount(cidr) {

    return cidr === 31 || cidr === 32 ? 0 : Math.pow(2, 32 - cidr) - 2;

}

// Hàm tính toán tất cả các mạng con

function calculateSubnets(ip1, ip2, ip3, ip4, cidr) {

    let baseIP = (ip1 << 24) | (ip2 << 16) | (ip3 << 8) | ip4;

    let hostBits = 32 - cidr;

    let numberOfSubnets = Math.pow(2, cidr - 24);

    let subnetSize = Math.pow(2, hostBits);

    let subnets = [];

    for (let i = 0; i < numberOfSubnets; i++) {

        let subnetIP = baseIP + (i \* subnetSize);

        let subnetStart = [

            (subnetIP >>> 24) & 0xFF,

            (subnetIP >>> 16) & 0xFF,

            (subnetIP >>> 8) & 0xFF,

            subnetIP & 0xFF

        ];

        let subnetEndIP = subnetIP + subnetSize - 1;

        let broadcastIP = [

            (subnetEndIP >>> 24) & 0xFF,

            (subnetEndIP >>> 16) & 0xFF,

            (subnetEndIP >>> 8) & 0xFF,

            subnetEndIP & 0xFF

        ];

        let rangeStart = [...subnetStart];

        rangeStart[3] += 1;

        let rangeEnd = [...broadcastIP];

        rangeEnd[3] -= 1;

        subnets.push({

            network: subnetStart,

            broadcast: broadcastIP,

            rangeStart: rangeStart,

            rangeEnd: rangeEnd

        });

    }

    return subnets;

}

// Cập nhật hàm xử lý sự kiện khi nhấn nút tính toán

document.getElementById('calculateBtn').addEventListener('click', () => {

    let oktett1 = document.getElementById('octet1').value;

    let oktett2 = document.getElementById('octet2').value;

    let oktett3 = document.getElementById('octet3').value;

    let oktett4 = document.getElementById('octet4').value;

    let cidr = document.getElementById('cidr').value;

    if (oktett1 && oktett2 && oktett3 && oktett4 && cidr) {

        calculate();

    } else {

        alert("Vui lòng nhập đầy đủ các giá trị.");

    }

});

### 3.2.4 Giải thích các hàm của tệp script.js

**1. Hàm oktettToBinary**

**Mục đích**: Chuyển đổi số nguyên oktett (từ 0 đến 255) thành chuỗi nhị phân 8 bit.

function oktettToBinary(oktett) {

return oktett.toString(2).padStart(8, '0');

}

**Giải thích**:

* oktett.toString(2) chuyển đổi số nguyên oktett thành chuỗi nhị phân.
* .padStart(8, '0') đảm bảo chuỗi nhị phân có độ dài 8 ký tự, bằng cách thêm số 0 ở đầu nếu cần.

**Công thức**: Binary=oktett.toString(2).padStart(8, ’0’)\

**2. Hàm binaryToDecimal**

**Mục đích**: Chuyển đổi địa chỉ IP từ dạng nhị phân thành số nguyên.

function binaryToDecimal(binaryAddress) {

return binaryAddress.split('.').map(octet => parseInt(octet, 2));

}

**Giải thích**:

* binaryAddress.split('.') chia chuỗi nhị phân thành các phần oktett.
* map(octet => parseInt(octet, 2)) chuyển mỗi phần nhị phân thành số nguyên.

**Công thức**: Decimal=parseInt(binaryAddress, 2)

**3. Hàm calculate**

**Mục đích**: Tính toán các thông tin địa chỉ IP và subnet mask.

function calculate() {

// ...

// Địa chỉ IP dưới dạng nhị phân

let binaryIP = [oktettIP1, oktettIP2, oktettIP3, oktettIP4].map(oktettToBinary).join('.');

// Xác định subnet mask

let mask = calculateSubnetMask(cidr);

// Subnet mask dưới dạng nhị phân

let binaryMask = mask.split('.').map(octet => oktettToBinary(parseInt(octet))).join('.');

// Xác định địa chỉ mạng

let networkAddress = calculateNetworkAddress(oktettIP1, oktettIP2, oktettIP3, oktettIP4, mask);

// Xác định địa chỉ broadcast

let broadcastAddress = calculateBroadcastAddress(networkAddress, mask);

// Xác định phạm vi địa chỉ IP

let ipRange = calculateIPRange(networkAddress, broadcastAddress, cidr);

// Xác định số lượng địa chỉ IP hợp lệ

let ipCount = calculateValidIPCount(cidr);

// Tính toán và hiển thị tất cả các mạng con

let subnetworks = calculateSubnets(oktettIP1, oktettIP2, oktettIP3, oktettIP4, cidr);

// Xác định lớp IP

let cimosztalyLabel = document.getElementById('cimosztalyLabel');

if (oktettIP1 >= 1 && oktettIP1 <= 127) cimosztalyLabel.textContent = "Class A";

else if (oktettIP1 >= 128 && oktettIP1 <= 191) cimosztalyLabel.textContent = "Class B";

else if (oktettIP1 >= 192 && oktettIP1 <= 223) cimosztalyLabel.textContent = "Class C";

else if (oktettIP1 >= 224 && oktettIP1 <= 239) cimosztalyLabel.textContent = "Class D";

else if (oktettIP1 >= 240 && oktettIP1 <= 255) cimosztalyLabel.textContent = "Class E";

}

**Giải thích**:

* Hàm này gọi các hàm tính toán khác để xác định các thông tin như địa chỉ IP nhị phân, subnet mask, địa chỉ mạng, địa chỉ broadcast, phạm vi IP, số lượng IP hợp lệ và các mạng con.

**4. Hàm calculateSubnetMask**

**Mục đích**: Tính toán subnet mask dựa trên CIDR.

function calculateSubnetMask(cidr) {

const mask = (0xFFFFFFFF << (32 - cidr)) >>> 0;

return [

(mask >>> 24) & 0xFF,

(mask >>> 16) & 0xFF,

(mask >>> 8) & 0xFF,

mask & 0xFF

].join('.');

}

**Giải thích**:

* 0xFFFFFFFF là giá trị nhị phân 32 bit tất cả 1.
* << (32 - cidr) dịch bit sang trái để tạo subnet mask.
* >>> 0 đảm bảo giá trị là số dương.
* Chia giá trị mask thành các oktett và chuyển đổi thành định dạng IP.

**Công thức**: Mask=(0xFFFFFFFF<<(32−CIDR))&0xFFFFFFFF

**5. Hàm calculateNetworkAddress**

**Mục đích**: Tính toán địa chỉ mạng bằng cách AND địa chỉ IP với subnet mask.

function calculateNetworkAddress(ip1, ip2, ip3, ip4, mask) {

let maskArr = mask.split('.').map(Number);

return [

ip1 & maskArr[0],

ip2 & maskArr[1],

ip3 & maskArr[2],

ip4 & maskArr[3]

].join('.');

}

**Giải thích**:

* Tách subnet mask thành các phần oktett.
* Thực hiện phép AND bitwise giữa địa chỉ IP và subnet mask.

**Công thức**: Network Address=IP&Mask

**6. Hàm calculateBroadcastAddress**

**Mục đích**: Tính toán địa chỉ broadcast từ địa chỉ mạng và subnet mask.

function calculateBroadcastAddress(networkAddress, mask) {

let networkArr = networkAddress.split('.').map(Number);

let maskArr = mask.split('.').map(Number);

let invertedMask = maskArr.map(octet => 255 - octet);

return [

networkArr[0] | invertedMask[0],

networkArr[1] | invertedMask[1],

networkArr[2] | invertedMask[2],

networkArr[3] | invertedMask[3]

].join('.');

}

**Giải thích**:

* Tách địa chỉ mạng và subnet mask thành các phần oktett.
* Invert subnet mask và thực hiện phép OR bitwise với địa chỉ mạng.

**Công thức**: Broadcast Address=Network Address ∣ (Inverted Mask)

**7. Hàm calculateIPRange**

**Mục đích**: Xác định phạm vi địa chỉ IP hợp lệ trong mạng con.

function calculateIPRange(networkAddress, broadcastAddress, cidr) {

if (cidr === 31 || cidr === 32) return "―";

let [startIP1, startIP2, startIP3, startIP4] = networkAddress.split('.').map(Number);

startIP4 += 1;

let [endIP1, endIP2, endIP3, endIP4] = broadcastAddress.split('.').map(Number);

endIP4 -= 1;

return `${startIP1}.${startIP2}.${startIP3}.${startIP4} - ${endIP1}.${endIP2}.${endIP3}.${endIP4}`;

}

**Giải thích**:

* Tăng giá trị oktett cuối cùng của địa chỉ mạng lên 1 để tính địa chỉ đầu của phạm vi.
* Giảm giá trị oktett cuối cùng của địa chỉ broadcast xuống 1 để tính địa chỉ cuối của phạm vi.

**Công thức**: IP Range=(Network Address+1)−(Broadcast Address−1)

**8. Hàm calculateValidIPCount**

**Mục đích**: Tính số lượng địa chỉ IP hợp lệ trong một mạng con.

function calculateValidIPCount(cidr) {

return cidr === 31 || cidr === 32 ? 0 : Math.pow(2, 32 - cidr) - 2;

}

**Giải thích**:

* Đối với CIDR 31 hoặc 32, không có địa chỉ hợp lệ vì địa chỉ mạng và broadcast không sử dụng được.
* Còn lại, số lượng địa chỉ hợp lệ là tổng số địa chỉ trong mạng con trừ đi 2 (cho địa chỉ mạng và broadcast).

**Công thức**: Valid IP Count=2^32−CIDR−2

**9. Hàm calculateSubnets**

**Mục đích**: Tính toán tất cả các mạng con từ địa chỉ IP gốc và CIDR.

function calculateSubnets(ip1, ip2, ip3, ip4, cidr) {

let baseIP = (ip1 << 24) | (ip2 << 16) | (ip3 << 8) | ip4;

let hostBits = 32 - cidr;

let numberOfSubnets = Math.pow(2, cidr - 24);

let subnetSize = Math.pow(2, hostBits);

let subnets = [];

for (let i = 0; i < numberOfSubnets; i++) {

let subnetIP = baseIP + (i \* subnetSize);

let subnetStart = [

(subnetIP >>> 24) & 0xFF,

(subnetIP >>> 16) & 0xFF,

(subnetIP >>> 8) & 0xFF,

subnetIP & 0xFF

];

let subnetEndIP = subnetIP + subnetSize - 1;

let broadcastIP = [

(subnetEndIP >>> 24) & 0xFF,

(subnetEndIP >>> 16) & 0xFF,

(subnetEndIP >>> 8) & 0xFF,

subnetEndIP & 0xFF

];

let rangeStart = [...subnetStart];

rangeStart[3] += 1;

let rangeEnd = [...broadcastIP];

rangeEnd[3] -= 1;

subnets.push({

network: subnetStart,

broadcast: broadcastIP,

rangeStart: rangeStart,

rangeEnd: rangeEnd

});

}

return subnets;

}

**Giải thích**:

* Chuyển đổi địa chỉ IP thành giá trị số nguyên 32 bit.
* Tính số lượng subnet và kích thước của mỗi subnet.
* Lặp qua các subnet, tính toán địa chỉ mạng, broadcast, và phạm vi cho mỗi subnet.

**Công thức**: Subnet Size=2(32−CIDR)

## 3.3 Minh họa chương trình chia mạng con trên website bằng GitHub Pages.

### Bước 1: Tạo Kho Lưu Trữ (Repository) Mới trên GitHub

* Đăng nhập vào tài khoản GitHub của bạn.
* Tạo một kho lưu trữ mới bằng cách nhấp vào dấu "+" ở góc trên cùng bên phải và chọn "New repository". A screenshot of a computer

  Description automatically generated
* Đặt tên cho kho lưu trữ, ví dụ: Đồ án tốt ngiệp, Chọn quyền truy cập là "Public" để mọi người có thể xem trang của bạn. A screenshot of a computer

  Description automatically generated
* Nhấn vào nút "Create repository" để tạo kho lưu trữ. A screenshot of a computer

  Description automatically generated

### Bước 2: Tải Tệp Lên Kho Lưu Trữ

* Sau khi kho lưu trữ đã được tạo, nhấp vào "Add file" và chọn "Upload files". A screenshot of a computer

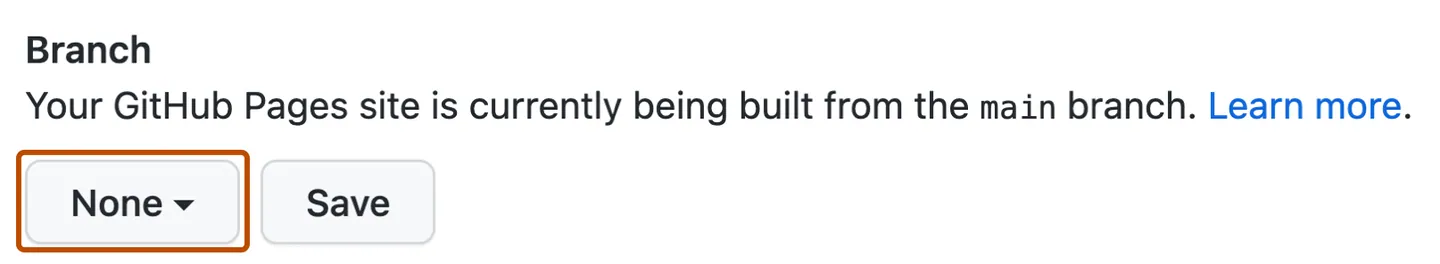
  Description automatically generated
* Kéo và thả các tệp index.html, styles.css, và script.js từ máy tính của bạn vào trang GitHub, Nhấn vào nút "Commit changes" để tải tệp lên kho lưu trữ. A screenshot of a computer

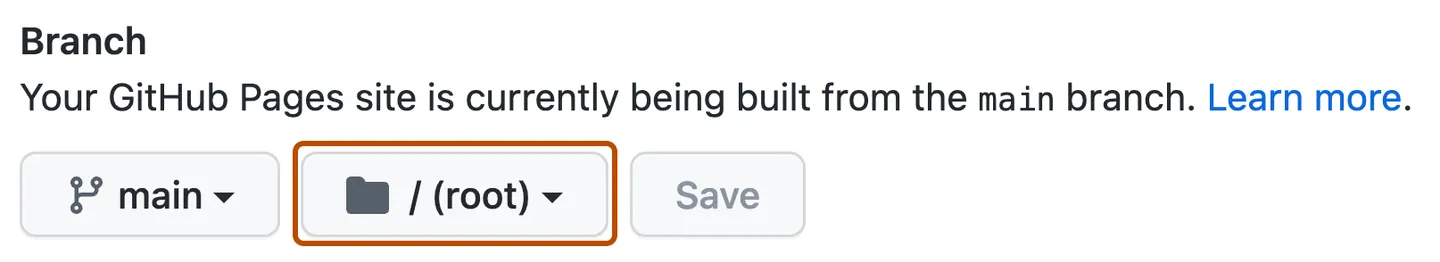
  Description automatically generated

### Bước 3: Kích Hoạt GitHub Pages

* Truy cập trang chính của kho lưu trữ của bạn.
* Nhấp vào "Settings" (cài đặt) ở phía trên của kho lưu trữ. A screenshot of a computer

  Description automatically generated
* Trong mục "Code and automation", tìm và nhấp vào "Pages". A screenshot of a computer

  Description automatically generated
* Trong phần "Branch", chọn main (hoặc master, tùy vào thiết lập kho lưu trữ của bạn) từ menu thả xuống, và đảm bảo thư mục là / (root).



* Nhấn "Save" để kích hoạt GitHub Pages.

### Bước 4: Truy cập Trang Web

* Sau khi kích hoạt GitHub Pages, bạn sẽ thấy một URL hiển thị trên trang "Pages". URL này sẽ có dạng: https://hophuocbk1.github.io/DoAnTotNgiep/A close up of a text

  Description automatically generated
* Nhấp vào liên kết này hoặc sao chép và dán vào trình duyệt để truy cập trang web mà bạn vừa tạo. A screenshot of a computer

  Description automatically generated

### Bước 5: Cập Nhật và Duy Trì Trang Web

* Nếu bạn cần cập nhật nội dung của trang web, chỉ cần sửa đổi các tệp và tải chúng lên lại kho lưu trữ.
* Mỗi khi bạn tải lên một phiên bản mới của tệp HTML, CSS, hoặc JavaScript, GitHub Pages sẽ tự động cập nhật trang web của bạn.

# TÀI LIỆU KHAM KHẢO

1. <https://viettelidc.com.vn/tin-tuc/ipv4-la-gi-tong-hop-kien-thuc-tu-a-z>
2. <https://vinahost.vn/ipv4-la-gi/>
3. <https://bkhost.vn/blog/ipv4/>
4. <https://vngcloud.vn/vi/blog/tong-quan-5-mo-hinh-trien-khai-dien-toan-dam-may-pho-bien>
5. <https://bizflycloud.vn/tin-tuc/tim-hieu-ve-dien-toan-dam-may-cloud-computing-193.htm>
6. <https://onesme.vn/blog/san-pham/mo-hinh-dich-vu-dien-toan-dam-may.html>
7. <https://chatgpt.com/>
8. <https://business19.chuanmen.edu.vn/2012/03/flsm-chia-mang-con-subnet-mask.html>
9. <https://hostingviet.vn/cidr-la-gi-khai-niem-classless-inter-domain-routing>
10. <https://mona.media/cidr-la-gi/>
11. <https://kdata.vn/tin-tuc/github-pages-va-cach-luu-tru-trang-web-mien-phi-nhu-the-nao>
12. <https://www.thanhnamnguyen.dev/blog/deploy-project-len-github-pages/>