BDRPC187**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**BÁO CÁO CUỐI KỲ**

**MÔN HỌC: ĐIỆN TOÁN ĐÁM MÂY**

*ĐỀ TÀI*

**XÂY DỰNG VPC VÀ CHẠY WEBSERVER TRÊN AWS**

**

**GVHD:** TS. Huỳnh Xuân Phụng

**(**Nhóm 4 – STT 58 – Lớp MOOC 09)

**Sinh viên thực hiện:**

Nguyễn Minh Sơn (MSSV: 20110713)

Nguyễn Đức Thành (MSSV: 20110307)

Nguyễn Duy Đăng (MSSV: 20110632)

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP.HCM KHOA CNTT**  \*\*\*\*\*\*\* | **XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh Phúc**  \*\*\*\*\*\*\* |

# **PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

Họ và tên Sinh viên 1: Nguyễn Minh Sơn MSSV: 20110713

Họ và tên Sinh viên 2: Nguyễn Đức Thành MSSV: 20110307

Họ và tên Sinh viên 3: Nguyễn Duy Đăng MSSV: 20110632

Ngành: Công nghệ thông tin

Tên đề tài: Xây dựng VPC và chạy webserver trên AWS

Họ và tên Giáo viên hướng dẫn: TS. Huỳnh Xuân Phụng

**NHẬN XÉT**

Về nội dung đề tài khối lượng thực hiện:

..................................................................................................................................................

..................................................................................................................................................

..................................................................................................................................................

..................................................................................................................................................

1. Ưu điểm:

..................................................................................................................................................

.................................................................................................................................................

..................................................................................................................................................

..................................................................................................................................................

1. Khuyết điểm

..................................................................................................................................................

..................................................................................................................................................

..................................................................................................................................................

..................................................................................................................................................

1. Đề nghị cho bảo vệ hay không?
2. Đánh giá loại:
3. Điểm:

Tp*. Hồ Chí Minh, ngày tháng năm 2021*

Giáo viên hướng dẫn

*(Ký & ghi rõ họ tên)*

# 

# **MỤC LỤC**

[**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN** 1](#_Toc120998415)

[**MỤC LỤC** 2](#_Toc120998416)

[**LỜI CẢM ƠN** 4](#_Toc120998417)

[**PHÂN CÔNG** 5](#_Toc120998418)

[**PHẦN 1: MỞ ĐẦU** 6](#_Toc120998419)

[**PHẦN 2: NỘI DUNG** 7](#_Toc120998420)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 7](#_Toc120998421)

[**1.1.** **Amazon Virtual Private Cloud (VPC)** 7](#_Toc120998422)

[**1.2.** **Amazon DynamoDB** 8](#_Toc120998423)

[**1.3.** **Amazon Elastic Compute Cloud (EC2)** 10](#_Toc120998424)

[**1.4.** **Amazon Simple Storage Service (S3)** 12](#_Toc120998425)

[CHƯƠNG 2: THIẾT KẾ VÀ CÀI ĐẶT 13](#_Toc120998426)

[**2.1.** **Phân tích yêu cầu** 13](#_Toc120998427)

[**2.1.1.** **Chức năng của website** 13](#_Toc120998428)

[**2.1.2.** **Thiết kế Cơ sở dữ liệu NoSQL** 13](#_Toc120998429)

[**2.2.** **Cài đặt chương trình** 14](#_Toc120998430)

[**2.2.1.** **Xây dựng Website Quản lý thông tin sinh viên** 14](#_Toc120998431)

[**2.2.2. Cấu trúc thư mục của website 14**](#_Toc120998432)

[**2.2.3. Giao diện Website 15**](#_Toc120998433)

[**2.2.4.** **Xây dựng dịch vụ tương tác với Amazon DynamoDB** 20](#_Toc120998434)

[**2.2.5.** **Xây dựng dịch vụ tương tác với Amazon S3** 28](#_Toc120998435)

[**2.2.** **Chạy project trên AWS** 29](#_Toc120998436)

[**2.3.1.** **Xây dựng VPC** 29](#_Toc120998437)

[**2.3.2.** **Tạo EC2** **instance** 32](#_Toc120998438)

[**2.3.3.** **Tạo bucket S3** 34](#_Toc120998439)

[**2.3.4.** **Tạo table DynamoDB và dữ liệu ban đầu** 34](#_Toc120998440)

[**2.3.5.** **Deploy WAR file** 35](#_Toc120998441)

[**2.3.6.** **Kiểm tra website** 35](#_Toc120998442)

[CHƯƠNG 3: KẾT LUẬN 36](#_Toc120998443)

[TÀI LIỆU THAM KHẢO 37](#_Toc120998444)

# **LỜI CẢM ƠN**

“Lời đầu tiên nhóm xin phép được gửi lời cảm ơn chân thành đến Khoa Công Nghệ Thông Tin – Trường Đại Học Sư Phạm Kỹ Thuật Thành Phố Hồ Chí Minh đã tạo điều kiện cho nhóm chúng em được học tập, phát triển nền tảng kiến thức sâu sắc và thực hiện đề tài này.

Bên cạnh đó chúng em xin gửi lời cảm ơn đến thầy Huỳnh Xuân Phụng đã chỉ dạy những kiến thức bổ ích của bộ môn Điện Toán Đám Mây và hướng dẫn chúng em thực hiện đề tài **Xây dựng VPC và chạy webserver trên AWS**.

Chúng em xin cảm ơn thầy vì thời gian qua đã hướng dẫn và chỉ dạy chúng em nhiệt tình giúp chúng em nắm được những kiến thức bổ ích của môn học này.

Tuy nhiên lượng kiến thức là vô tận nên chúng em đã cố gắng để có thể hoàn thành đề tài này, nên khó tránh khỏi thiếu sót chúng em mong thầy có thể thông cảm. Chúng em mong nhận được sự góp ý của thầy để có thêm kinh nghiệm để có thể hoàn thiện đề tài hơn.”

Chúng em xin chân thành cảm ơn!

**PHÂN CÔNG**

**Phân công thực hiện xây dựng website quản lý thông tin sinh viên**

**Link github:** [**https://github.com/NMS1010/Student\_Management-Cloud\_Computing**](https://github.com/NMS1010/Student_Management-Cloud_Computing)

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Công việc** |
| Nguyễn Minh Sơn | 20110713 | * Quản lý khoa * Quản lý giảng viên * Quản lý điểm * Đăng nhập |
| Nguyễn Đức Thành | 20110307 | * Quản lý sinh viên * Quản lý lớp học phần * Quản lý vai trò hệ thống |
| Nguyễn Duy Đăng | 20110632 | * Quản lý sinh viên * Quản lý lớp sinh viên * Quản lý học phần |

**PHẦN 1: MỞ ĐẦU**

1. **Tính cấp thiết của đề tài**

Cơ sở dữ liệu (database) nói chung và cơ sở dữ liệu không quan hệ (noSQL database) nói riêng là một thành phần không thể thiếu trong việc tạo nên các phần mềm ứng dụng hoặc website có như cầu lưu trữ thông tin có thể hoặc không do người dùng nhập vào. Nhờ chúng mà dữ liệu với những insight quan trọng có thể được lưu trữ một cách dễ dàng. Vì vậy mà việc tạo nên một hệ thống có khả năng làm cho việc tương tác đến cơ sở dữ liệu của các lập trình viên trở nên dễ dàng hơn là một việc cấp thiết cần phải được thực hiện.

1. **Đối tượng nghiên cứu**

Đối với đề tài này, đối tượng nghiên cứu là VPC, DynamoDB, EC2. Đồng thời kết hợp với các kiến thức về Front-end và Back-end để tạo một trang web Quản lý thông tin sinh viên.

1. **Phạm vi nghiên cứu**

Đề tài này chủ yếu tập trung vào việc kết hợp sử dụng VPC và EC2 nhằm thực hiện với chạy webserver, đồng thời sử dụng DynamoDB để làm Cơ sở dữ liệu

1. **Kết quả dự kiến đạt được**

- Cài đặt giao diện website Quản lý thông tin sinh viên

- Cài đặt phía backend

- Xây dựng VPC với 2 subnet: private và public

- Xây dựng máy ảo EC2 nằm trong VPC đã tạo để host website

- Thêm, sửa, xóa các item và các table trong DynamoDB

**PHẦN 2: NỘI DUNG**

**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

* 1. **Amazon Virtual Private Cloud (VPC)**
     1. **Tổng quan:**

Amazon Virtual Private Cloud (Amazon VPC) là dịch vụ cho phép bạn khởi chạy các tài nguyên AWS trong mạng ảo cô lập theo logic mà bạn xác định. Bạn có toàn quyền kiểm soát môi trường mạng ảo của mình, bao gồm lựa chọn dải địa chỉ IP, tạo các mạng con, cấu hình các bảng định tuyến và cổng kết nối mạng. Bạn có thể dùng cả IPv4 và IPv6 cho hầu hết các tài nguyên trong đám mây riêng ảo, giúp bảo mật nghiêm ngặt và truy cập dễ dàng các tài nguyên cũng như ứng dụng.

Là một trong các dịch vụ nền tảng của AWS, Amazon VPC sẽ giúp bạn dễ dàng tùy chỉnh cấu hình mạng của VPC. Bạn có thể tạo một mạng con công khai cho các máy chủ web có quyền truy cập internet. Dịch vụ này cũng cho phép bạn đặt các hệ thống backend, như máy chủ ứng dụng hoặc cơ sở dữ liệu, trong mạng con riêng tư không có quyền truy cập internet. Với Amazon VPC, bạn có thể sử dụng nhiều lớp bảo mật, bao gồm các nhóm bảo mật và danh sách kiểm soát truy cập mạng, để giúp kiểm soát quyền truy cập vào các phiên bản Amazon EC2 trong mỗi mạng con

* + 1. **Các tính năng**

Các tính năng dùng để cấu hình VPC

* **VPC**: là một mạng ảo gần giống với mạng truyền thống mà ta sẽ sử dụng nó trong trung tâm dữ liệu của mình. Sau khi tạo mới một VPC, ta có thể tạo các subnets
* **Subnets**: Một subnet là một dãi địa chỉ IP trong VPC ta tạo. Một subnet bắt buộc phải ở trong một AZ. Sau khi thêm subnets, bạn có thể triển khai tai nguyên lên VPC
* **Địa chỉ IP**: ta có thể gán địa chỉ IPv4 và IPv6 cho VPC hoặc subnet mình tạo.Nếu ta có các địa chỉ IPv4, IPv6 công khai của riêng mình, ta có thể cấp phát chúng cho các tài nguyên trong VPC như EC2 instance, NAT gateways, Network Load Balancers
* **Routing**: Cung cấp bảng Route để xác định nơi các lưu lượng mạng từ các subnet hoặc gateway hướng tới
* **Gatewates và endpoints**: Một gateway giúp kết nối VPC tới một mạng khác ngoài VPC. Ví dụ, ta dùng một internet gateway để kết nối VPC của mình tới internet. Dùng **VPC endpoint** để kết nối các dịch vụ AWS riêng tư mà không cần đến **internet gateway** hoặc thiết bị **NAT**
* **Peering connections:** sử dụng kết nối VPC peering để định tuyến lưu lượng giữa các tài nguyên trong 2 VPCs
* **Traffic Mirroring:** Copy lưu lượng mạng từ các network interfaces và gửi tới các thiết bị giám sát và bảo mật để phân tích các gói tin
* **Transit gateways:** Sử dụng một transit gateway (hoạt động giống một hub trung tâm) để định tuyến lưu lượng mạng giữa các VPC, kết nối VPN và các kết nối AWS Direct Connect
* **VPC Flow Logs:** Flow log dùng để ghi lại thông tin về các địa chi IP truy cập đến hoặc đi từ các network interface trong VPC ta tạo
  + 1. **Các dịch vụ liên quan**
* **AWS PrivateLink:** giúp thiết lập kết nối riêng tư giữa các VPC và các dịch vụ AWS được hỗ trỡ, các dịch vụ có trong tài khoản AWS và các dịch vụ được hỗ trợ trong AWS Marketplace. Ta không cần phải dùng internet gateway, thiết bị NAT, kết nối AWS Direct Connect hoặc AWS site-to-site VPN để giao tiếp với các dịch vụ từ VPC của mình
* **AWS Network Firewall:** AWS Network Firewall là một dịch vụ tường lửa mạng, mang tính trạng thái, được quản lý và là dịch vụ giúp phát hiện, ngăn chặn xâm nhậpbất hợp pháp vào VPC bạn đã tạo. Nó giúp ta lọc được lưu lượng mạng truy cập ở ngoại vi VPC, bao gồm cả lưu lượng truy cập đến và đi từ internet gateway, NAT gateway hoặc VPN..
* **Route 53 Resolver DNS Firewall:** dùng để lọc và điều chỉnh lưu lượng DNS gửi đi ra ngoài cho VPC của bạn. Để làm được, ta phải tạo các rule trong DNS Firewall. DNS Firewall cung cấp khả năng bảo vệ cho các yêu cầu DNS gửi đi từ các VPC
  1. **Amazon DynamoDB**
     1. **Tổng quan**

DynamoDB là môt dịch vụ cơ sở dữ liệu NoSQL có khả năng đáp ứng hiệu suất cao và nhanh kèm theo khả năng mở rộng. Nếu bạn là một nhà phát triển, bạn có thể sử dụng DynamoDB để tạo ra một bảng có khả năng lưu trữ và truy xuất bất kỳ số lượng dữ liệu, mà vẫn có thể phục vụ cho bất kỳ mức độ request traffic. DynamoDB được quản lý đầy đủ, hỗ trợ cả mô hình dữ liệu dưới dạng tài liệu và cặp value-key.

DynamoDB tự động phân tán dữ liệu và traffic của một bảng ra một số lượng server vừa đủ để có thể xử lý request capacity đặt ra bởi khách hàng và lượng dữ liệu lưu trữ, đồng thời đảm bảo hiệu suất nhanh và đồng nhất. Tất cả dữ liệu được lưu trữ trên SSD và tự động được sao chép ra các vùng sẵn sàng (Availability Zones) trong một khu vực (Region) để cung cấp độ sẵn sàng cao và độ bền của dữ liệu (high availablity and data durability)

* + 1. **Các chức năng, dịch vụ liên quan**

**AWS Glue Views** cho phép các lập trình viên tạo nên các giao diện trực quan cho phép dễ dàng tìm kiếm, kết hợp, nhân bản giữa nhiều kho dữ liệu khác nhau theo thời gian thực mà không cần phải tự viết code tùy biến, dữ liệu tổng hợp có thể được chuyển đến các dịch vụ khác như: Redshift, S3, Aura … phục vụ nhiều mục đích khác nhau.

**AWS Database Migration Service (AWS DMS)** cho phép chuyển dữ liệu trong cơ sở dữ liệu quan hệ hoặc MongoDB sang DynamoDB

**PartiSQL** là một ngôn ngữ truy vấn tương thích với SQL, có thể liên kết với nhiều cơ sở dữ liệu với nhiều định dạng dữ liệu khác nhau như dữ liệu có cấu trúc, dữ liệu bán cấu trúc, dữ liệu dạng lưới. Ngôn ngữ truy vấn này được áp dụng trong nhiều dịch vụ của AWS, trong đó có Dynamodb.

**Amazon Kinesis** được sử dụng để tổng hợp và xử lý các luồng dữ liệu lớn theo thời gian thực. Các dữ liệu này được lưu trong các data record và có thể được sử dụng cho nhiều mục đích khác nhau: tạo thông báo, tạo các bảng báo cáo, các bảng giá có giá trị thay đổi theo thời gian,…

Công cụ tính giá tiền AWS dùng dể tính tiền các dịch vụ hoặc các tài nguyên đã sử dụng trên AWS

* + 1. **Ưu điểm của DynamoDB**

Khả năng mở rộng: Chắc hẳn tất cả các lập trình viên đều biết rằng khi một trang web hoặc ứng dụng đưa vào thực tế một thời gian thì nó sẽ phát triển đến mức mà luồng dữ liệu ra vào hệ thống sẽ lớn mức gần như không điểm soát được. Đối với cơ sở dữ liệu có quan hệ (SQL Database) thì việc nâng cấp, mở rộng cơ sở dữ liệu sẽ rất khó thực hiện bởi tồn tại các ràng buộc giữa các bảng, tốn kém và đi kèm với nhiều rủi ro, do hầu hết dữ liệu được lưu trữ là dữ liệu của khách hàng. Còn nếu cài đặt cơ sở dữ liệu quá lớn so với nhu cầu sẽ dễ đến hệ quả là tốn những chi phí không cần thiết. Bài toán này đã được DynamoDB giải quyết bằng chức năng tự scale theo nhu cầu sử dụng, từ đó mà cân bằng được bài toán hiệu suất - chi phí.

Tính phân quyền: Khi cơ sở dữ liệu phát triển đến một mức độ nào đó thì việc cơ sở dữ liệu chỉ do duy nhất một lập trình viên quản lý là việc bất khả thi. **DynamoDB** cho phép phân quyền cho các lập trình, cho phép mỗi người có quyền truy cập vào một/ một số phân vùng nhất định.

Thời gian sống: **DynamoDB** cho phép lập trình viên cấu hình thời gian tồn tại của một table, khi thời gian tồn tại này hết thì table sẽ bị hủy. Việc này cho phép lập trình viên tự động hóa việc xóa dữ liệu thay vì xử lý bằng tay. Tính năng này cũng giúp tiết kiệm dung lượng sử dụng và chi phí.

Lưu trữ các dữ liệu không liên tục: Do đặc tính là một cơ sở dữ liệu NoSQL mà **DynamoDB** có đặc tính rất phù hợp với những đối tượng không có cấu trúc cố định và thay đổi theo thời gian.

Tự động quản lý: Với mục đích bảo mật và an toàn mà dữ liệu trong **DynamoDB** được tự động liên tục cập nhật và backup lên cloud.

* 1. **Amazon Elastic Compute Cloud (EC2)**
     1. **Tổng quan**

Amazon Elastic Compute Cloud (Amazon EC2) là một cơ sở hạ tầng điện toán đám mây được cung cấp bởi Amazon Web Services (AWS) giúp cung cấp tài nguyên máy tính ảo hoá theo yêu cầu.

Amazon EC2 cung cấp các ứng dụng máy tính ảo hoá có thể mở rộng về khả năng xử lý cùng các thành phần phần cứng ảo như bộ nhớ máy tính (ram), vi xử lý, linh hoạt trong việc lựa chọn các phân vùng lưu trữ dữ liệu ở các nền tảng khác nhau và sự an toàn trong quản lý dịch vụ bởi kiến trúc ảo hoá đám mây mạnh mẽ của AWS.

Amazon EC2 sẽ cung cấp một hoặc máy chủ ảo có thể kết hợp với nhau để dễ dàng triển khai ứng dụng nhanh nhất và đảm bảo tính sẵn sàng cao nhất. Thậm chí về mặt thanh toán bạn dễ dàng biết được các mức chi phí cần thanh toán dựa trên thông tin tài nguyên bạn sử dụng

* + 1. **Chức năng**
* Cung cấp môi trường máy tính ảo, còn gọi là instance
* Cung cấp sẵn các mẫu hệ điều hành cho instance (còn gọi là Amazon Machine Images - AMIs), gồm các gói cài đặt cần thiết cho server
* Cho phép tuỳ chỉnh, thiết lập cấu hình instance của bạn: CPU, bộ nhớ, lưu trữ…
* Bảo mật thông tin cho instance thông qua việc dùng key – pairs (AWS lưu trữ public key, người dùng lưu trữ private key)
* Lưu trữ tạm thời dữ liệu đã bị xóa sau khi ta thực hiện stop, hibernate hoặc terminate instance
* Cung cấp tường lửa cho phép bạn chỉ định giao thức, cổng dịch vụ và các nguồn IP được phép truy cập instance của bạn
* Dùng địa chỉ IPv4 tĩnh, hay gọi là Elastic IP address
  + 1. **Dịch vụ liên quan**

**Amazon EC2 Auto Scaling** giúp đảm bảo rằng luôn có chính xác, hợp lý số lượng EC2 instances để xử lý tải cho các ứng dụng. Ta có thể tạo các Auto Scaling groups – nhóm các EC2 instances. Bạn có thể chỉ định giới hạn số lượng instance trong mỗi nhóm và Amazon EC2 Auto Scaling luôn đảm bảo rằng số lượng instance luôn là phù hợp nhất. Nó cũng có thể tự động chạy hoặc huỷ các instance tuỳ theo yêu cầu đến ứng dụng tăng hay giảm

**AWS CloudFormation l**à dịch vụ giúp bạn mô hình hoá và thiết lập các tài nguyên AWS nhanh chóng để giúp tốn ít thời gian cho việc quản lý, thay vào đó tập trung vào các ứng dụng bạn đang chạy trên AWS

**AWS Elastic Beanstalk** là dịch vụ giúp bạn có thể triển khai nhanh chóng và quản lý các ứng dụng trong AWS Cloud mà không phải có kiến thức chuyên sâu về Cơ sở hạ tầng để chạy các ứng dụng đó. Nó giúp giảm sự phức tạp trong sự quản lý nhưng không hạn chế sự lựa chọn hoặc kiểm soát. Có thể nói, bạn chỉ cần upload ứng dụng của mình, Elastic Beanstalk sẽ tự động xử lý chi tiết về việc cung cấp dung lượng, cân bằng tải, mở rộng và giám sát tình trạng ứng dụng mà bạn không cần phải làm gì nhiều

**AWS OpsWorks** là dịch vụ quản lý cấu hình giúp bạn cấu hình và khởi chạy các ứng dụng trong các Cloud doanh nghiệp bằng cách sử dụng Puppet hoặc Chef

**Elastic Load Balacing** giúp tự động phân phối các lưu lượng truy cập đến cho nhiều mục tiêu như EC2 instance, container và địa chỉ IP, trong một hoặc nhiều AZ. Ngoài ra, nó còn giúp giám sát tình trạng mục tiêu đã đăng kí, chỉ định tuyến lưu lượng tới các mục tiêu có tình trạng ổn định. Hơn nữa, nó còn giúp tự động mở rộng khả năng cân bằng tải để đáp ứng các lưu lượng truy cập đến (nhiều hoặc ít truy cập tới)

**Amazon Relational Database Service (Amazon RDS)** là một dịch vụ web cho phép ta dễ dàng cài đặt, khởi chạy và mở rộng một cơ sở dữ liệu quan hệ trong AWS Cloud. Nó cung cấp khả năng thay đổi kích thước, hiệu quả về chi phí cho Cơ sở dữ liệu quan hệ và cung cấp các thao tác để quản trị cơ sở dữ liệu đó.

**Amazon Elastic Container Service (Amazon ECS)** được gọi tắt là ECS là một service quản lý container có tính scale cao và nhanh. Dễ dàng run, stop, hay quản lý docker container ở trong một cluster. Bạn có thể host một serverless infrastructure bằng cách chạy service hay task sử dụng Fragate launch type. Amazon ECScho phép bạn launch hay stop container-based thông qua API, cho phép bạn lấy state của cluster từ centralized service cũng như cho phép access tới các feature giống như EC2. Bạn có thể sử dụng Amazon ECS để cài đặt contaner thông qua cluster và dựa và nguồn tài nguyên mà bạn cần, chính sách độc lập hay khả năng thay đổi. Amazon ECS sẽ giúp bạn không phải operate việc quản lý cluster hay cấu hình system hay quản lý scaling infrastructure.

**Amazon CloudWatch** là một dịch vụ giúp giám sát, tổng hợp, phân tích dữ liệu, nguồn tài nguyên chạy trên AWS. Dịch vụ này giúp cung cấp thông tin thực tiễn một cách realtime, cho phép giám sát các vùng nhớ của ứng dụng, cơ sở hạ tầng và dịch vụ ví dụ như Ram, Disk,... và sử dụng cảnh báo, tự động hành động; hỗ trợ việc tối ưu hóa hiệu suất ứng dụng, quản lý sử dụng tài nguyên và hiểu rõ tình trạng hoạt động của toàn hệ thống.

**Amazon GuardDuty** sử dụng học máy (machine learning) để tìm kiếm hoạt động độc hại trong môi trường AWS của bạn. Nó kết hợp CloudTrail event logs, VPC flow logs, S3 event logs và DNS logs để liên tục theo dõi và phân tích tất cả hoạt động. GuardDuty xác định các vấn đề như leo thang đặc quyền, thông tin đăng nhập bị lộ và giao tiếp với các miền và địa chỉ IP độc hại. Nó cũng có thể phát hiện khi các EC2 instance của bạn đang phân phối phần mềm độc hại hoặc khai thác bitcoin. Ngoài ra, GuardDuty có thể phát hiện sự bất thường trong các mẫu truy cập của bạn, chẳng hạn như lệnh gọi API ở các vùng mới. Định giá dựa trên lượng dữ liệu được phân tích, vì vậy chi phí sẽ tăng tuyến tính khi môi trường AWS của bạn phát triển.

* 1. **Amazon Simple Storage Service (S3)**
     1. **Tổng quan**

Là dịch vụ đám mây lưu trữ do đó bạn có thể tải lên các tệp, các tài liệu, các dữ liệu tải về của người dùng hoặc các bản sao lưu

Amazon S3 lưu trữ dữ liệu như các object trong các bucket. Một object gồm 1 file và metadata mô tả cho file (tùy chọn).

Để lưu 1 object trong Amazon S3, bạn tải file lên 1 bucket. Khi đã tải file, bạn có thể gán quyền cho đối tượng cũng như bổ sung metadata.

Bucket là các thùng chứa cho các object. Bạn có thể tạo 1 hay nhiều bucket nhưng mỗi bucket sẽ có tên riêng biệt, không trùng nhau. Với mỗi bucket, bạn có thể điều khiển việc truy xuất đến nó (ai có thể tạo, xóa và xem các object trong bucket), xem nhật ký truy xuất đến bucket và đến các object bên trong, cũng như chọn region mà Amazon S3 sẽ lưu bucket và nội dung trong nó.

* + 1. **Chức năng**
* Lưu trữ các tệp, tài liệu hoặc bất cứ loại dữ liệu nào của người dùng
* Truy xuất dữ liệu
* Có khả năng mở rộng và độ linh hoạt cao
* Bảo mật dữ liệu được lưu và phân quyền truy cập dữ liệu
  1. **VPC Endpoints**
     1. **Khái niệm**
* VPC Endpoint đóng vai trò là cầu nối, và là kết nối riêng biệt để các instance trong VPC có thể giao tiếp với các dịch vụ khác trên AWS mà không cần đi qua Internet, NAT, VPN, hay AWS Direct Connect.
* VPC Endpoint sẽ có một ID định danh để có thể điều hướng traffic từ VPC đến service. Tất cả các traffic đi từ VPC đến AWS Service đều không đi ra khỏi AWS network.
  + 1. **Các loại endpoint**
* **Endpoint service**: ta có một application được chạy bên trong VPC và thiết lập một endpoint để kết nối với ứng dụng đó. Các AWS Account, dịch vụ khác có thể tạo kết nối từ VPC của chúng đến endpoint của chúng ta.
* **Endpoint**: 2 loại endpoint thông dụng
  + **Gateway endpoint**: một gateway endpoint được coi là một cánh cổng mà sẽ được khai báo trong bảng định tuyền, chỉ định để các gói tin đi qua để truy cập đến một số dịch vụ được support bởi AWS như S3, DynamoDB.
  + **Interface endpoint**: là một ENI với một địa chỉ IP nội bộ từ dải IP của mạng con, đóng vai trò là điểm đầu vào cho các gói tin đến một số dịch vụ được hỗ trợ. Nó cho phép bạn truy cập một cách bảo mật bằng cách sử dụng các địa chỉ IP nội bộ.

## CHƯƠNG 2: THIẾT KẾ VÀ CÀI ĐẶT

1. **Phân tích yêu cầu**

Yêu cầu của đề tài:

* Xây dựng VPC: 1 public subnet + 1 private subnet
* Khởi tạo 1 EC2 (cài đặt web server) + 1 DynamoDB
* Website quản lý thông tin sinh viên (Giao diện + Xử lý + Database đơn giản)

1. **Chức năng của website**

**-** Website quản lý sinh viên gồm các chức năng CRUD

* Quản lý sinh viên
* Quản lý điểm
* Quản lý giảng viên
* Quản lý khoa
* Quản lý lớp sinh viên
* Quản lý học phần
* Quản lý lớp học phần
* Quản lý tài khoản
* Quản lý vai trò

- Các role trên website:

* Người quản trị
* Giảng viên

- Phân chia chức năng theo role

* Người quản trị: Tất cả chức năng của website quản lý sinh viên
* Giảng viên: Quản lí điểm các nhóm lớp giảng viên đó dạy và quản lý thông tin của chính giảng viên đó

1. **Thiết kế Cơ sở dữ liệu NoSQL**

**faculty**(facultyId, facultyName, image, deleted)

**student\_class**(studentClassId, studentClassName, facultyId, deleted)

**student**(studentId, studentClassId, studentName, dob, address, gender, phone, image, deleted)

**subject**(subjectId, subjectName, creditsNo, periodsNo, deleted)

**grade**(studentId, subjectGroupId, middleGrade, finalGrade, totalGrade, deleted)

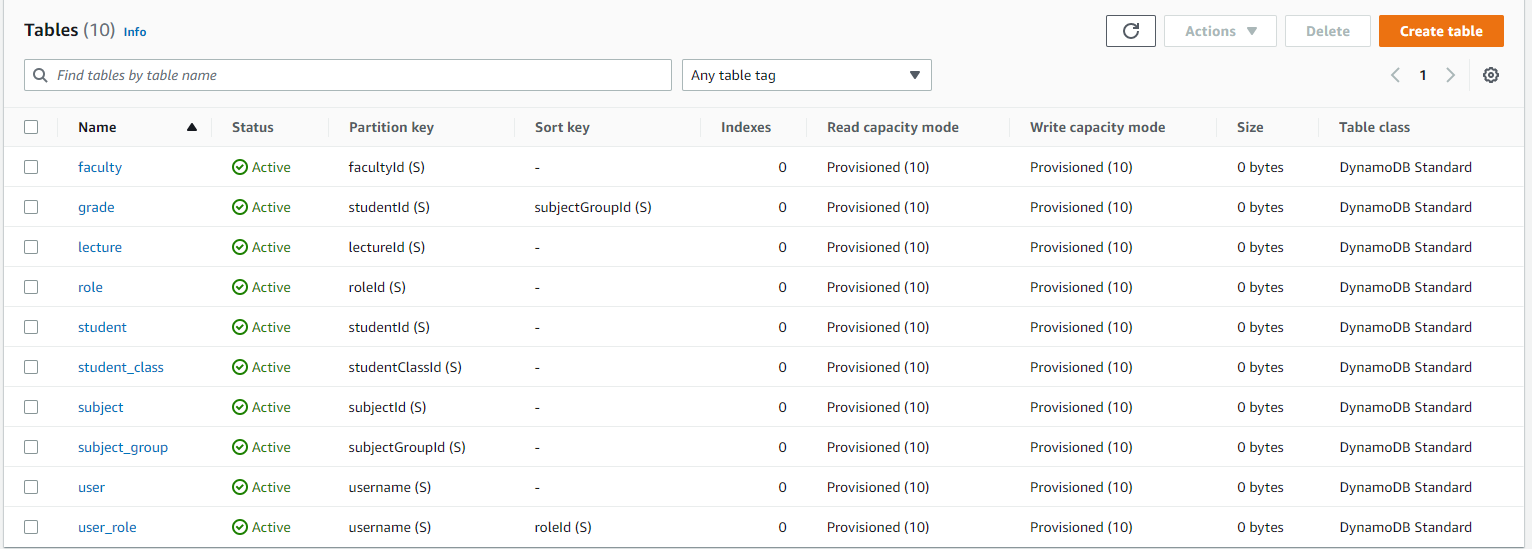
**lecture**(lectureId, lectureName, dob, address, gender, phone, image, deleted)

**subject\_group**(subjectGroupId, subjectGroupName, subjectId, lectureId, deleted)

**user**(username, password, lectureId, deleted)

**role**(roleId, roleName, deleted)

**user\_role**(userName, roleId)

****

1. **Cài đặt chương trình**
2. **Xây dựng Website Quản lý thông tin sinh viên**

Website sử dụng công nghệ JSP & Servlet và cài đặt gói AWS SDK

### 2.2.2. Cấu trúc thư mục của website

* **Common/interfaces**: Chứa các interface chung cho thao tác CRUD
* **Controllers**: phần controller để tiếp nhận các request và gọi các Service tương ứng để xử lý
* **Filter**: dùng để lọc các request cho mục đích xác thực
* **Models**:
  + **Entities**: các lớp thực thể sử dụng trong hệ thống
  + **Repositories**: triển khai các phương thức CRUD thao tác với cơ sở dữ liệu DynamoDB
  + **Services**: triển khai các dịch vụ để tương tác với **repositories** và các dịch vụ của AWS như DynamoDB và S3
  + **View\_models**: chứa các class định nghĩa CreateRequest (yêu cầu tạo đối tượng), UpdateRequest (yêu cầu cập nhật đối tượng) và ViewModel (lớp dữ liệu trả về cho view)
* **Utils**: chứa các class sử dụng chung

Text

Description automatically generated

### 2.2.3. Giao diện Website

Trang Login

**Graphical user interface

Description automatically generated**

Trang Quản lý Khoa

**Graphical user interface, text, application

Description automatically generated**

Trang Quản lý Giảng viên

**Graphical user interface, application

Description automatically generated**

Trang Quản lý vai trò hệ thống

**Graphical user interface, text, application

Description automatically generated**

Trang Quản lý Học phần

**Graphical user interface, text, application

Description automatically generated**

Trang Quản lý Lớp sinh viên

**Graphical user interface, application

Description automatically generated**

Trang Quản lý Sinh viên

**Graphical user interface, application

Description automatically generated**

Trang Quản lý Lớp học phần

**Graphical user interface, application

Description automatically generated**

Trang Quản lý Tài khoản

**Graphical user interface, application, Teams

Description automatically generated**

Trang Quản lý Điểm lớp học phần

**Graphical user interface, application

Description automatically generated**

* + 1. **Xây dựng dịch vụ tương tác với Amazon DynamoDB**

Hàm tạo các table trong DynamoDB

Text

Description automatically generated

Trước khi tạo mới, ta xoá hết các table sẵn có trong DynamoDB bằng hàm **DropAllTable**

Text

Description automatically generated

Sau đó ta thực hiện tạo mới các table bằng hàm **GenerateTable,** hàm này bao gồm các hàm tạo từng bảng như phần mô tả Cơ sở dữ liệu

Text

Description automatically generated with medium confidence

Các hàm tạo bảng tương ứng

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Tiếp theo ta khởi tạo dữ liệu ban đầu để có tài khoản đăng nhập vào trang quản lý bằng hàm **InitialData**

Sau đó, ta triển khai dịch vụ DynamoDB thông qua các phương thức trong gói AWS SDK bằng class **AmazonDynamoDBService.java** trong project để có thể thao tác với DynamoDB (thêm, sửa, xoá) bằng ngôn ngữ Java

Text

Description automatically generated

Hàm **getDynamoDB** để lấy đối tượng DynamoDB (dùng để tạo các truy vấn tới CSDL DynamoDB) từ đối tượng **client** đã được tạo kết nối thông qua AWS Credentials

Các hàm thực hiện thao táo CRUD trên table của DynamoDB

* **Insert**

Text

Description automatically generated

* **Update**

Text

Description automatically generated

* **Delete**

Text

Description automatically generated

* **GetItem**

Text

Description automatically generated

* **Scan**

Text

Description automatically generated

* + 1. **Xây dựng dịch vụ tương tác với Amazon S3**

Ta triển khai dịch vụ Amazon S3 thông qua gói AWS SDK bằng class **AmazonS3Service.java** trong project

Text

Description automatically generated

Hàm **uploadFile** để upload một file lên S3 từ máy và lấy URL file để lưu vào CSDL

* 1. **Chạy project trên AWS**

Clone project theo link github: <https://github.com/NMS1010/Student_Management-Cloud_Computing>

* + 1. **Xây dựng VPC**

**Graphical user interface, text, application, email

Description automatically generated**

Địa chỉ mạng IPv4 cho VPC: **10.0.0.0/16**

Ta đặt 2 subnet trên 1 AZ

**Graphical user interface, text, application, email

Description automatically generated**

1 Public Subnet có địa chỉ mạng: **10.0.0.0/24**

1 Private Subnet có địa chỉ mạng: **10.0.1.0/24**

**Graphical user interface, text, application

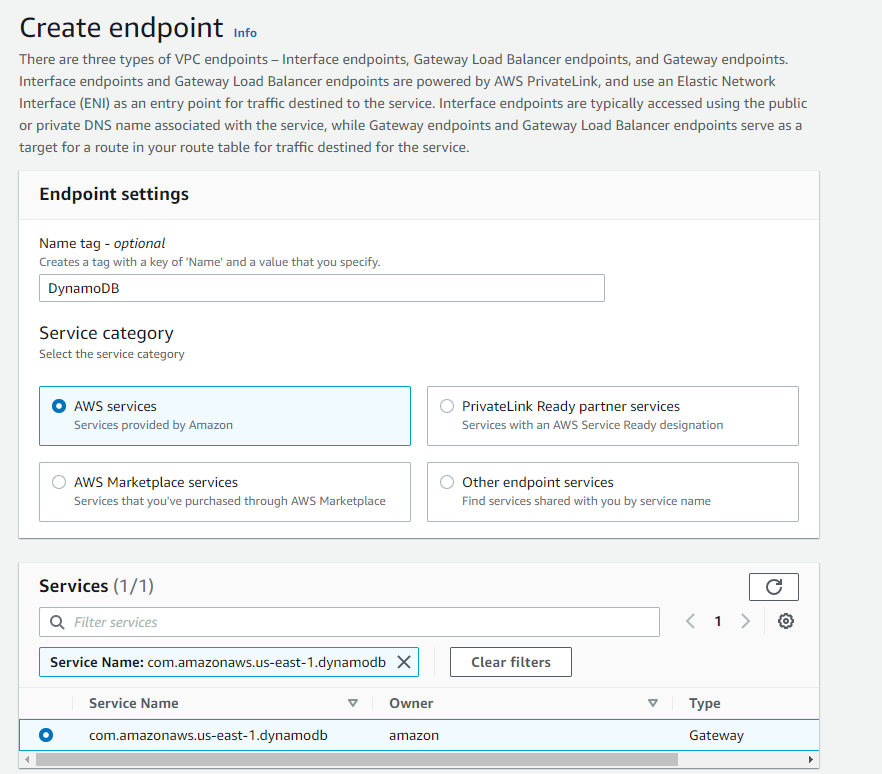
Description automatically generated**

Tiếp theo, tạo một **Security Group** (coi như một tường lửa ảo) giúp lọc các traffic đến VPC

* Đặt tên: Web Security Group
* Inbound rules: HTTP, HTTPs, All Traffic
* Source: Anywhere – IPv4

Ta tạo một VPC endpoint cho DynamoDB bằng cách:

* Chọn vào mục **Endpoint** bên phần Dashboard
* Chọn Create Endpoint
* Đặt tên, chọn dịch vụ DynamoDB, chọn VPC ta vừa tạo: project-vpc
* Chọn route table và quyền truy cập bằng policy
* Nhấn Create Endpoint



Graphical user interface, text, application, email

Description automatically generated

* + 1. **Tạo EC2** **instance**
* Tạo một EC2 instance với tên là **Project Web Server** và hệ điều hành **Amazon Linux 2**
* Đến phần Network Settings, chọn Edit
  + Chọn VPC: **project-vpc**
  + Chọn Subnet: **public**
  + Chọn **Select existing security group**: **Web Security Group**

**Graphical user interface, text, application, email

Description automatically generated**

* Sau khi EC2 instance vừa tạo được khởi chạy, ta tiến hành kết nối đến máy ảo EC2
* Ta truy cập **user root** trên máy ảo bằng lệnh: **sudo su –** và chạy lệnh **sudo yum update** để tiến hành cập nhật các package trên Hệ điều hành
* Cài đặt **Apache Tomcat 9** trên máy ảo EC2 để có thể chạy ứng dụng java web:
  + Chuyển đến thư mục opt: **cd /opt**
  + Download file apache tomcat: **wget** <https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.69/bin/apache-tomcat-9.0.69.tar.gz>
  + Giải nén file tar.gz: **tar -xvzf /opt/apache-tomcat-9.0.69.tar.gz**
* Chuyển đến thư mục: **cd apache-tomcat-9.0.69/bin**
* Cài đặt Java: **yum install java-1.8.0**
* Để có thể truy cập được Manage App của Tomcat Server, ta cần thực hiện
  + Tìm các file content.xml : **find / -name context.xml**
  + Mở file bằng lệnh **vim** và comment các thẻ **valve** trong từng file
    - **vim /opt/apache-tomcat-9.0.69/webapps/host-manager/META-INF/context.xml**
    - **vim /opt/apache-tomcat-9.0.69/webapps/manager/META-INF/context.xml**
  + Thêm các username và role sau đây vào file bằng lệnh (lưu ý đang ở thư mục bin): **vim ../conf/tomcat-users.xml**

**<role rolename="manager-gui"/>**

**<role rolename="manager-script"/>**

**<role rolename="manager-jmx"/>**

**<role rolename="manager-status"/>**

**<user username="admin" password="admin" roles="manager-gui, manager-script, manager-jmx, manager-status"/>**

**<user username="deployer" password="deployer" roles="manager-script"/>**

**<user username="tomcat" password="s3cret" roles="manager-gui"/>**

* Ta đã cài đặt xong Tomcat Server trên EC2
  + 1. **Tạo bucket S3**
* Truy cập dịch vụ S3 trên AWS
* Chọn Create Bucket
* Đặt tên bucket: **studentms**
* Trong khi tạo, chọn **ACLs enabled**, bỏ chọn **Block all public access**, tick vào ô **I acknowledge that the current settings might result...** để cho phép truy cập link các file được lưu trong S3 từ internet
* Sau khi tạo thành công, chọn bucket mới tạo, chọn tab Permission -> Bucket Policy -> Edit -> Copy Policy sau và dán vào ô trống -> Save

**{ "Version":"2012-10-17", "Statement":[ { "Sid":"PublicReadGetObject", "Effect":"Allow", "Principal":"\*", "Action":[ "s3:GetObject" ], "Resource":[ "arn:aws:s3:::studentms/\*" ] } ] }**

* + 1. **Tạo table DynamoDB và dữ liệu ban đầu**

Ta sẽ tạo table DynamoDB từ máy local thông qua Credentials của account AWS

* Đầu tiên, ta cần cài đặt AWS CLI
* Sau khi cài xong, ta mở CMD và chạy lệnh **aws configure**
* Tiếp theo, nhập lần lượt **Access key**, **Secret access key**, **region: 'us-east-1'**, **output format: 'json'** được lấy từ tài khoản Learner Lab sau khi đã Start Lab
* Chạy lệnh **aws configure set aws\_session\_token ‘{value}’** với **{value}** là token của phiên đăng nhập hiện tại mà tài khoản AWS đang chạy
* Mở project đã clone bằng **IntelliJ**, vào thư mục **src/main/java/utils** và chạy file **DynamoDBUtils.java** để tiến hành tạo các table trên **Amazon DynamoDB** và thêm dữ liệu ban đầu
  + 1. **Deploy WAR file**
* Kết nối tới máy ảo EC2, ta chuyển sang quyền root: **sudo su -**
* Ta chạy lệnh **aws configure** để chỉ định credentials của tài khoản AWS cho máy ảo EC2 và chỉ định session token bằng lệnh **aws configure set aws\_session\_token {value}** ({value} là token của phiên đăng nhập hiện tại)
* Ta kiểm tra lại xem đã config thành công chưa bằng lệnh: **aws sts get-caller-identity**
* Ta gõ lệnh: **cd apache-tomcat-9.0.69/bin**
* Gõ lệnh: **./startup.sh** để khởi chạy **Tomcat server**
* Copy địa chỉ public DNS của máy ảo EC2 và truy cập vào bằng cổng 8080 (truy cập bằng giao thức HTTP)
* Sau khi truy cập trang **Tomcat server**, chọn **Manager App**
* Đăng nhập với **username/password** là **admin/admin**
* Tiếp theo, ta chọn file WAR đã tạo từ project và deploy lên **Tomcat Server** trong mục **WAR file to deploy**
* Ta đợi quá trình deploy hoàn tất thì đã có thể truy cập được vào Website từ mục **Applications** của Tomcat Server
  + 1. **Kiểm tra website**
* Khởi chạy website theo địa chỉ trang web ở bước **2.3.5**
* Đăng nhập với vai trò người quản trị bằng tài khoản/mật khẩu: **admin/admin**
* Thực hiện các thao tác CRUD để kiểm tra các chức năng

## CHƯƠNG 3: KẾT LUẬN

1. **Kết quả đạt được**

Sau một thời gian nghiên cứu và thực hiện đề tài “**Xây dựng VPC và chạy webserver trên AWS**”, nhóm chúng em đã đạt được những kết quả như sau:

* Kiến thức tìm hiểu được
  + Nắm bắt được các kiến thức cũng như những cách hoạt động của các dịch vụ VPC, EC2, S3, DynamoDB và áp dụng kiến thức để thiết kế và xây dựng một mô hình tương tác với database.
  + Nắm bắt được mô hình hoạt động, cách tạo một VPC, cách phân chia subnet
  + Biết cách deploy website lên EC2
  + Biết dùng S3 để lưu trữ ảnh
  + Biết sử dụng AWS SDK bằng Java
* Chương trình đã làm được: Xây dựng hoàn chỉnh một website quản lý sinh viên đơn giản bằng Java, HTML, CSS, JS và các dịch vụ của AWS: VPC, EC2, S3, DynamoDB với các chức năng cơ bản như:
  + CRUD trên website
  + Tạo các table trong DynamoDB.
  + Thêm, sửa, xóa, đọc các Item trong table của DynamoDB.

1. **Ưu điểm**

* Website với giao diện đơn giản, dễ tương tác
* Tương tác nhanh vì DynamoDB là CSDL dạng NoSQL

1. **Nhược điểm**

* Các bước cài đặt và deploy lên EC2 khó
* Database đơn giản
* Giao diện không được đẹp mắt
* Tốn phí vì sử dụng dịch vụ của AWS và phụ thuộc phần lớn vào AWS trong vấn đề chạy code

**TÀI LIỆU THAM KHẢO**

1. Amazon EC2: <https://docs.aws.amazon.com/ec2/index.html>
2. Amazon VPC: <https://docs.aws.amazon.com/vpc/>
3. Amazon DynamoDB: <https://docs.aws.amazon.com/dynamodb/>
4. Amazon S3: <https://docs.aws.amazon.com/s3/index.html>
5. Link github tham khảo: <https://github.com/tranquoctuan2001/dynamodb-vpc-nodejs>