

TRƯỜNG ĐẠI HỌC PHENIKAA  
KHOA CÔNG NGHỆ THÔNG TIN

**BÁO CÁO HP: ĐIỆN TOÁN ĐÁM MÂY**  
**Building a Highly Available, Scalable Web**  
**Application**

21013345

Quản Trọng Hùng

21013345@st.phenikaa-uni.edu.vn

**Giảng viên hướng dẫn:** Nghiêm Việt Cường

**Khoa:** Công nghệ thông tin

**HÀ NỘI, 10/2024**

# Lời cam kết

Họ và tên nhóm sinh viên: Quân Trọng Hùng

Điện thoại liên lạc: 0367120622      Email: 21013345@st.phenikaa-uni.edu.vn

Lớp: K15-KHMT(AI&KHDL)      Hệ đào tạo: đại học chính quy

Tôi/Chúng tôi cam kết Bài tập lớn (BTL) là công trình nghiên cứu của bản thân/nhóm tôi. Các kết quả nêu trong BTL là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong BTL – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi/chúng tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

*Hà Nội, ngày 4 tháng 10 năm 2024*

Tác giả/nhóm tác giả BTL

*Họ và tên sinh viên*

## Contents

Lời cam kết .....	2
Chương 1: Giới thiệu về ứng dụng .....	5
1.1    Đặt vấn đề: .....	5
1.2    Giải pháp: .....	5
1.2.1. Lập kế hoạch và thiết kế kiến trúc: .....	5
1.2.2. Kiến trúc AWS Well-Architected Framework: .....	6
1.2.3. Bảo mật (Security): .....	6
1.2.4. Cân bằng tải (Load Balancing): .....	6
1.2.5. Xây dựng và triển khai ứng dụng web: .....	7
Chương 2: Thiết kế giải pháp điện toán đám mây (PI 6.1) .....	8
2.1 Thiết kế tổng quan .....	8
2.1.1. Biểu đồ kiến trúc triển khai trên nền tảng AWS bao gồm các thành phần chính sau: ..	8
2.1.2. Phân Tích Các Dịch Vụ AWS Được Sử Dụng Trong Dự Án .....	10
2.2 Phân tích các thành phần được triển khai .....	12
2.2.1. Network – Virtual Private Cloud (VPC) .....	12
2.2.2. Webserver – Amazon EC2 (Elastic Compute Cloud) .....	12
2.2.3. Database – Amazon RDS (Relational Database Service) .....	13
2.2.4. Load Balancer – Application Load Balancer (ALB) .....	14
Chương 3: Các giải pháp an toàn và bảo mật (PI 6.2) .....	15
3.1. Các khía cạnh an toàn và bảo mật liên quan đến ứng dụng .....	15
3.2. Giải pháp bảo mật có thể triển khai cho ứng dụng .....	15
Chương 4: Phân tích chi phí / Triển khai các tính năng nâng cao .....	17

4.1. Tóm tắt ước tính chi phí.....	17
4.2. Các phương án tiết kiệm chi phí .....	17
4.3. Tính năng nâng cao .....	18
Chương 5: Triển khai ứng dụng, kiểm thử và đánh giá (PI 2.1).....	19
5.1 Kết quả triển khai ứng dụng: .....	19
5.1.1. cloud9-scripts.yml.....	19
5.1.2. UserDataScript-phase-2.sh .....	21
5.1.3. UserDataScript-phase-3.sh .....	23
5.2. Demo.....	24
5.3. Kết luận:.....	27

# Chương 1: Giới thiệu về ứng dụng

## 1.1 Đặt vấn đề:

Trong bối cảnh chuyển đổi số và nhu cầu ngày càng tăng về các dịch vụ trực tuyến, ứng dụng web đóng vai trò quan trọng trong việc phục vụ người dùng và lưu trữ thông tin. Đặc biệt, với các ứng dụng lưu trữ dữ liệu quan trọng như hồ sơ sinh viên, tính khả dụng và hiệu suất của hệ thống là yếu tố then chốt. Các ứng dụng này thường xuyên gặp phải tình trạng quá tải trong các giai đoạn cao điểm, như mùa tuyển sinh, khi số lượng người dùng tăng đột biến.

Việc lưu trữ và quản lý ứng dụng web trên các nền tảng đám mây như AWS mang lại những lợi thế vượt trội về khả năng mở rộng, tính khả dụng cao, và hiệu suất ổn định. Một trong những thách thức lớn đối với các tổ chức hiện nay là làm sao để đảm bảo ứng dụng luôn sẵn sàng, không gặp sự cố khi số lượng yêu cầu tăng cao, đồng thời tối ưu hóa chi phí vận hành.

Vì vậy, việc triển khai một giải pháp lưu trữ ứng dụng web trên nền tảng đám mây, sử dụng các dịch vụ như Elastic Load Balancer, EC2, và RDS, sẽ giúp cải thiện trải nghiệm người dùng và tăng cường hiệu quả vận hành, đồng thời đảm bảo tính linh hoạt và bảo mật cho hệ thống. Nền tảng đám mây cũng cung cấp các công cụ tự động mở rộng tài nguyên khi cần thiết, giúp đáp ứng nhu cầu sử dụng trong các thời kỳ cao điểm mà không làm gián đoạn dịch vụ.

Chính vì vậy, giải pháp triển khai ứng dụng web trên AWS không chỉ giúp nâng cao chất lượng dịch vụ mà còn tối ưu hóa chi phí và đảm bảo hệ thống có thể mở rộng khi cần thiết.

## 1.2 Giải pháp:

### 1.2.1. Lập kế hoạch và thiết kế kiến trúc:

Kiến trúc cần đảm bảo các yếu tố:

- + High Availability (Tính khả dụng cao): Ứng dụng phải sẵn sàng 24/7, với khả năng tiếp tục hoạt động ngay cả khi một khu vực (region) hoặc một phần của cơ sở hạ tầng gặp sự cố.

- + Scalability (Khả năng mở rộng): Hệ thống có thể tự động mở rộng (scale out) hoặc thu gọn (scale in) tài nguyên dựa trên lượng yêu cầu thực tế, đặc biệt trong thời gian cao điểm.

- + Load Balancing (Cân bằng tải): Phân phối lưu lượng truy cập đều lên các tài nguyên tính toán để đảm bảo hiệu suất ổn định và tránh quá tải cho một tài nguyên duy nhất.

+ Security (Bảo mật): Hệ thống phải bảo vệ dữ liệu sinh viên, ngăn chặn các mối đe dọa an ninh, và tuân thủ các quy định bảo mật dữ liệu.

+ Performance (Hiệu suất): Đảm bảo thời gian phản hồi nhanh chóng và truy cập dữ liệu ổn định ngay cả khi có hàng nghìn người dùng cùng truy cập.

### **1.2.2. Kiến trúc AWS Well-Architected Framework:**

- Tính khả dụng cao (High Availability):

AWS Elastic Load Balancer (ELB): Sử dụng ELB để phân phối lưu lượng ứng dụng web trên nhiều EC2 instance (máy chủ ảo), giúp giảm tải và tăng khả năng đáp ứng của hệ thống.

- Khả năng mở rộng (Scalability):

+ Auto Scaling Group: Tạo nhóm Auto Scaling cho EC2 instance, tự động thêm hoặc bớt tài nguyên máy chủ ảo dựa trên lưu lượng thực tế.

+ Amazon RDS Auto Scaling: Đối với cơ sở dữ liệu, sử dụng Amazon Relational Database Service (RDS) với khả năng tự động mở rộng để đáp ứng nhu cầu lưu trữ và truy vấn dữ liệu tăng cao.

### **1.2.3. Bảo mật (Security):**

+ AWS Identity and Access Management (IAM): Cấp quyền truy cập dựa trên vai trò, đảm bảo chỉ người dùng có quyền hợp lệ mới có thể truy cập dữ liệu.

+ Amazon VPC: Mọi thành phần của ứng dụng sẽ chạy bên trong một Virtual Private Cloud (VPC), với các quy tắc kiểm soát truy cập mạng (Network ACLs) và bảo mật lớp bảo vệ (Security Groups) để bảo vệ tài nguyên khỏi các cuộc tấn công từ bên ngoài.

+ SSL/TLS Encryption: Sử dụng chứng chỉ SSL để mã hóa dữ liệu giữa người dùng và ứng dụng, đảm bảo rằng tất cả các thông tin nhạy cảm đều được bảo vệ.

### **1.2.4. Cân bằng tải (Load Balancing):**

Elastic Load Balancer (ELB): Ngoài việc tăng tính khả dụng, ELB còn giúp cân bằng tải giữa các instance, giảm thiểu nguy cơ quá tải cho một máy chủ duy nhất, và phân bổ lưu lượng dựa trên yêu cầu truy cập.

### 1.2.5. Xây dựng và triển khai ứng dụng web:

- Mã nguồn ứng dụng:

- + Backend: Viết ứng dụng với linux.

- + Frontend: Sử dụng HTML, CSS, JavaScript (React hoặc Vue.js) để phát triển giao diện người dùng.

- + Database: Sử dụng Amazon RDS với MySQL để lưu trữ hồ sơ sinh viên.

- Công cụ triển khai và giám sát:

Amazon CloudWatch: Sử dụng để giám sát tài nguyên hệ thống và tạo cảnh báo khi có sự cố, ví dụ như khi tải CPU vượt quá mức cho phép.

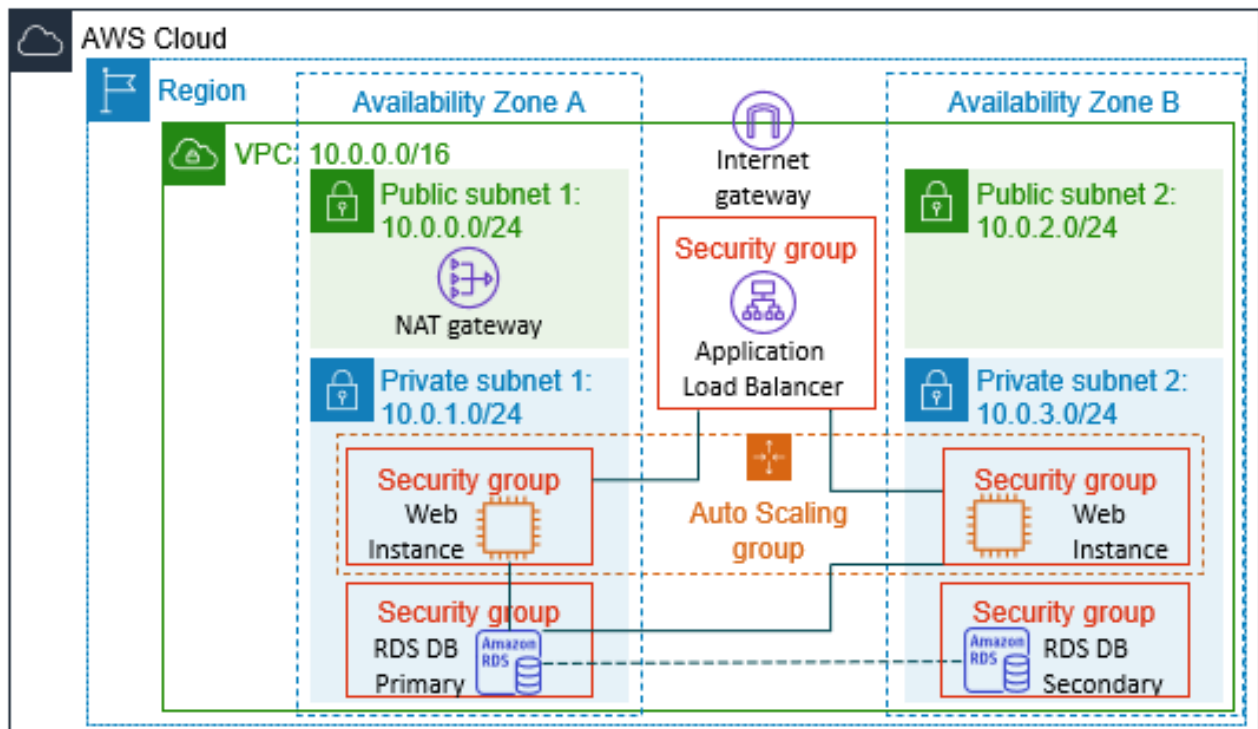
- Tối ưu hóa chi phí vận hành:

- + Chi phí: Sử dụng tính năng theo dõi và báo cáo chi phí từ AWS để điều chỉnh tài nguyên phù hợp với ngân sách của trường. Sử dụng các instance dự trữ để giảm chi phí khi không có nhu cầu cao điểm.

⇒ Kiến trúc này đáp ứng đầy đủ các yêu cầu về tính khả dụng, khả năng mở rộng, bảo mật, cân bằng tải và hiệu suất cao cho ứng dụng web của phòng tuyển sinh. AWS Cloud cung cấp nền tảng linh hoạt và đáng tin cậy để xử lý các thách thức của mùa tuyển sinh cao điểm, mang lại trải nghiệm tốt hơn cho sinh viên và đảm bảo hệ thống vận hành ổn định trong mọi hoàn cảnh.

# Chương 2: Thiết kế giải pháp điện toán đám mây (PI 6.1)

## 2.1 Thiết kế tổng quan



### 2.1.1. Biểu đồ kiến trúc triển khai trên nền tảng AWS bao gồm các thành phần chính sau:

#### 1. VPC (Virtual Private Cloud):

- Chức năng: VPC là môi trường mạng ảo cho phép người dùng định cấu hình mạng lưới theo ý muốn trên AWS. Trong biểu đồ, VPC với CIDR `10.0.0.0/16` tạo ra một không gian mạng để chứa các tài nguyên (EC2, RDS, Load Balancer, etc.).

- Nhiệm vụ: Đảm bảo các tài nguyên bên trong không gian mạng hoạt động trong môi trường an toàn và có thể tương tác với nhau thông qua subnet và các thiết lập bảo mật.



## 2. Public Subnets:

- Chức năng: Subnets này nằm trong mạng công khai (Public), tức là các tài nguyên bên trong nó có thể kết nối với Internet thông qua Internet Gateway.

- Nhiệm vụ: Trong biểu đồ, 'Public Subnet 1' và 'Public Subnet 2' chứa các thành phần như NAT Gateway và Application Load Balancer, cho phép chúng tương tác trực tiếp với Internet.

## 3. Private Subnets:

- Chức năng: Subnet này là mạng riêng, nghĩa là các tài nguyên bên trong không thể truy cập trực tiếp từ Internet.

- Nhiệm vụ: 'Private Subnet 1' và 'Private Subnet 2' chứa các EC2 Instances (Web Instances) và RDS Database. Các thành phần này không trực tiếp kết nối Internet nhưng có thể truy cập Internet thông qua NAT Gateway trong Public Subnet.

## 4. Internet Gateway:

- Chức năng: Là cổng nối giữa VPC và Internet. Tất cả các tài nguyên nằm trong subnet công khai có thể sử dụng Internet Gateway để nhận và gửi lưu lượng Internet.

- Nhiệm vụ: Giúp các thành phần như Application Load Balancer kết nối với người dùng cuối qua Internet.

## 5. NAT Gateway:

- Chức năng: NAT (Network Address Translation) Gateway cho phép các tài nguyên trong subnet riêng tư truy cập Internet mà không bị phơi bày trực tiếp.

- Nhiệm vụ: Trong biểu đồ, NAT Gateway trong 'Public Subnet 1' cho phép các máy chủ EC2 trong subnet riêng kết nối Internet để cập nhật hệ điều hành hoặc các dịch vụ cần thiết khác mà vẫn đảm bảo an toàn.

## 6. Application Load Balancer (ALB):

- Chức năng: ALB phân phối lưu lượng truy cập đến các máy chủ EC2 bên dưới trong các Availability Zones (A và B), đảm bảo cân bằng tải và độ sẵn sàng cao.

- Nhiệm vụ: ALB nhận lưu lượng từ Internet và phân phối tới các máy chủ Web Instances trong Auto Scaling Group ở cả hai Availability Zones, giúp tăng khả năng chịu tải và đảm bảo dịch vụ luôn sẵn sàng.

## 7. Auto Scaling Group:

- Chức năng: Tự động điều chỉnh số lượng máy chủ EC2 dựa trên lưu lượng hoặc nhu cầu thực tế của ứng dụng.

- Nhiệm vụ: Trong biểu đồ, Auto Scaling Group quản lý số lượng Web Instances để đảm bảo ứng dụng luôn sẵn sàng, tăng số lượng khi lưu lượng tăng cao và giảm khi lưu lượng giảm, giúp tiết kiệm chi phí.

## 8. EC2 Instances (Web Instances):

- Chức năng: EC2 (Elastic Compute Cloud) là các máy chủ ảo cung cấp sức mạnh tính toán để chạy ứng dụng web.

- Nhiệm vụ: Các Web Instances trong biểu đồ chịu trách nhiệm xử lý các yêu cầu từ người dùng, và được quản lý trong các Private Subnets để tăng cường bảo mật.

## 9. RDS (Relational Database Service):

- Chức năng: RDS cung cấp dịch vụ cơ sở dữ liệu quản lý. Biểu đồ hiển thị một hệ thống cơ sở dữ liệu dạng master-slave (Primary và Secondary) để tăng tính sẵn sàng.

- Nhiệm vụ: RDS Primary DB và Secondary DB được triển khai trong các Availability Zones khác nhau để đảm bảo tính sẵn sàng cao và dự phòng cho cơ sở dữ liệu sinh viên.

## 10. Security Groups:

- Chức năng: Security Group là một loại firewall quản lý lưu lượng vào và ra cho các tài nguyên AWS.

- Nhiệm vụ: Security Group được sử dụng để bảo vệ Web Instances và RDS DB, đảm bảo chỉ có những lưu lượng hợp lệ mới được truy cập tài nguyên.

## 2.1.2. Phân Tích Các Dịch Vụ AWS Được Sử Dụng Trong Dự Án

### 1. VPC (Virtual Private Cloud):

Tạo môi trường mạng riêng trên AWS, đảm bảo tính bảo mật và quản lý tài nguyên hiệu quả.

### 2. EC2 (Elastic Compute Cloud):

Cung cấp sức mạnh tính toán cho ứng dụng web, có thể tự động mở rộng hoặc thu nhỏ thông qua Auto Scaling Group.

### 3. RDS (Relational Database Service):

Dịch vụ cơ sở dữ liệu có quản lý, đảm bảo tính dự phòng và sẵn sàng cao với các phiên bản Primary và Secondary.

### 4. Application Load Balancer (ALB):

Cân bằng tải và phân phối lưu lượng truy cập cho các EC2 instances trong Auto Scaling Group, đảm bảo hiệu suất tốt nhất cho người dùng.

### 5. Auto Scaling:

Điều chỉnh tự động số lượng EC2 instances dựa trên nhu cầu thực tế của ứng dụng, giúp tối ưu hóa chi phí và tài nguyên.

### 6. NAT Gateway:

Cung cấp truy cập Internet cho các tài nguyên nằm trong Private Subnet mà không phơi bày địa chỉ IP ra ngoài.

### 7. Internet Gateway:

Cầu nối giữa VPC và Internet, giúp tài nguyên trong Public Subnet kết nối và tương tác với người dùng.

### 8. Security Groups:

Quản lý và kiểm soát lưu lượng vào và ra của các tài nguyên, đảm bảo tính bảo mật cao trong quá trình vận hành.

⇒ Kiến trúc này không chỉ đảm bảo hiệu năng, khả năng mở rộng và tính sẵn sàng cao mà còn đáp ứng các yêu cầu về bảo mật và quản lý chi phí một cách hiệu quả.

## 2.2 Phân tích các thành phần được triển khai

### 2.2.1. Network – Virtual Private Cloud (VPC)

- Thiết lập cấu hình:

+ VPC (Virtual Private Cloud): Được tạo để cung cấp môi trường mạng riêng biệt, bảo mật cho ứng dụng web. VPC có thể chứa nhiều subnet công khai và riêng tư để đảm bảo tính bảo mật và hiệu suất cho các thành phần khác nhau của ứng dụng.

+ Subnet: Trong VPC, các mạng con (subnets) được tạo ra để tách biệt giữa các thành phần ứng dụng.

+ Public Subnet: Được sử dụng để lưu trữ các thành phần cần truy cập từ internet, chẳng hạn như máy chủ web EC2.

+ Private Subnet: Sử dụng cho các dịch vụ cần bảo mật cao, như cơ sở dữ liệu RDS, nhằm hạn chế truy cập từ bên ngoài.

- Tại sao cấu hình này phù hợp:

Tính bảo mật: Tách biệt giữa mạng công khai và riêng tư giúp bảo vệ cơ sở dữ liệu RDS không bị truy cập trái phép từ internet, chỉ có thể truy cập từ máy chủ web.

- Hiệu suất và bảo trì:

Cấu hình này giúp dễ dàng quản lý và bảo trì hệ thống mạng với các quy tắc kiểm soát truy cập rõ ràng, cho phép kiểm soát lưu lượng dữ liệu ra/vào mạng.

### 2.2.2. Webserver – Amazon EC2 (Elastic Compute Cloud)

- Thiết lập cấu hình:

EC2 Instance: Amazon EC2 cung cấp máy ảo (instance) cho ứng dụng web. Một AMI (Amazon Machine Image) sử dụng Ubuntu được chọn để triển khai ứng dụng web.

- Loại Instance:

Tùy thuộc vào mức tải dự kiến, bạn có thể chọn loại instance như t2.micro cho POC hoặc nâng cấp lên t3.medium khi dự án mở rộng.

- Elastic IP:

Một địa chỉ IP tĩnh được gán cho EC2 instance, giúp đảm bảo ứng dụng luôn có thể truy cập từ internet với một địa chỉ cố định.

- Tại sao cấu hình này phù hợp:

+ Hiệu suất và tính sẵn sàng cao: Amazon EC2 dễ dàng mở rộng, giúp đáp ứng lưu lượng người dùng tăng cao trong thời gian tuyến sinh cao điểm. Việc sử dụng EC2 kết hợp với Auto Scaling Group đảm bảo rằng ứng dụng có thể tự động mở rộng để đáp ứng nhu cầu mà không cần ngừng hoạt động.

+ Tính linh hoạt: EC2 cung cấp khả năng tùy chỉnh cấu hình instance dễ dàng, giúp điều chỉnh theo yêu cầu cụ thể của ứng dụng web.

+ Bảo mật: Các security group cho phép kiểm soát lưu lượng dữ liệu ra vào instance, đảm bảo chỉ các cổng và địa chỉ IP cụ thể được phép truy cập.

### **2.2.3. Database – Amazon RDS (Relational Database Service)**

- Thiết lập cấu hình:

+ Amazon RDS (MySQL): RDS là dịch vụ quản lý cơ sở dữ liệu được AWS cung cấp, giúp lưu trữ và quản lý dữ liệu sinh viên.

+ Multi-AZ Deployment: Cấu hình này đảm bảo cơ sở dữ liệu có tính sẵn sàng cao, với dữ liệu được sao lưu đồng bộ tại hai Vùng khả dụng (Availability Zones).

+ Security Group: Chỉ cho phép máy chủ web trong subnet riêng tư có quyền truy cập vào cơ sở dữ liệu, ngăn chặn việc truy cập trái phép từ bên ngoài.

+ Backup & Restore: Tự động sao lưu dữ liệu hàng ngày, đảm bảo tính liên tục của dữ liệu ngay cả trong trường hợp hỏng hóc.

- Tại sao cấu hình này phù hợp:

+ Tính bảo mật và tính sẵn sàng: Việc triển khai Multi-AZ giúp đảm bảo dữ liệu được bảo vệ trong trường hợp có lỗi phần cứng hoặc sự cố mạng tại một Vùng khả dụng. Điều này giúp hệ thống luôn sẵn sàng và hoạt động ổn định.

+ Tự động hóa: RDS tự động xử lý các tác vụ như sao lưu, cập nhật phần mềm, và quản lý sao lưu cơ sở dữ liệu, giảm bớt công việc thủ công cho nhóm vận hành.

+ Khả năng mở rộng: Khi lượng người dùng tăng lên, RDS có thể mở rộng tài nguyên cơ sở dữ liệu một cách dễ dàng để đáp ứng nhu cầu.

#### **2.2.4. Load Balancer – Application Load Balancer (ALB)**

- Thiết lập cấu hình:

+ Application Load Balancer (ALB): Dịch vụ này phân phối lưu lượng giữa các máy chủ EC2, giúp cân bằng tải và đảm bảo ứng dụng không bị quá tải khi có nhiều người dùng truy cập.

+ Listener: ALB lắng nghe trên các cổng (như HTTP/HTTPS) để phân phối lưu lượng truy cập vào các instance EC2.

+ Target Group: Được cấu hình để liên kết các instance EC2 trong nhóm Auto Scaling.

- Tại sao cấu hình này phù hợp:

+ Cân bằng tải: ALB đảm bảo rằng lưu lượng truy cập đến ứng dụng web được phân phối đồng đều giữa các máy chủ web. Điều này giảm thiểu nguy cơ một máy chủ bị quá tải và dẫn đến ứng dụng chậm hoặc ngừng hoạt động.

+ Tính linh hoạt: ALB cho phép thiết lập các chính sách cân bằng tải phức tạp hơn, chẳng hạn như cân bằng theo session, cho phép điều chỉnh theo nhu cầu của người dùng.

+ Tính sẵn sàng cao: Khi kết hợp với Auto Scaling, ALB đảm bảo rằng ứng dụng có thể tự động mở rộng và cung cấp trải nghiệm người dùng ổn định.

⇒ Thiết lập trên cung cấp tính linh hoạt, hiệu suất cao, và tính bảo mật. VPC tách biệt thành subnet công khai và riêng tư để bảo vệ dữ liệu, trong khi EC2 và RDS được quản lý để đáp ứng lưu lượng truy cập cao. ALB và Auto Scaling đảm bảo ứng dụng có khả năng mở rộng và luôn sẵn sàng phục vụ người dùng, đặc biệt trong thời gian cao điểm tuyển sinh.

# Chương 3: Các giải pháp an toàn và bảo mật

## (PI 6.2)

### 3.1. Các khía cạnh an toàn và bảo mật liên quan đến ứng dụng

Ứng dụng web cần bảo vệ các thông tin nhạy cảm như dữ liệu cá nhân, dữ liệu sinh viên, và thông tin quản trị. Các khía cạnh bảo mật chính bao gồm:

- + Bảo vệ dữ liệu người dùng: Đảm bảo rằng dữ liệu cá nhân của sinh viên được mã hóa và chỉ có thể truy cập bởi những người dùng có quyền hợp lệ.

- + Xác thực và phân quyền: Đảm bảo rằng chỉ có người dùng đã được xác thực (authenticate) mới có thể truy cập vào ứng dụng, và các tài nguyên hệ thống được phân quyền đúng theo vai trò (admin, user,...).

- + Bảo vệ khỏi tấn công mạng: Ngăn chặn các cuộc tấn công như SQL injection, cross-site scripting (XSS), distributed denial of service (DDoS), và các phương pháp hack thông qua các lỗ hổng bảo mật.

- + An ninh mạng: Tạo lập một kiến trúc mạng có khả năng ngăn chặn truy cập không hợp lệ và bảo vệ các thành phần ứng dụng quan trọng khỏi bị tấn công từ bên ngoài.

### 3.2. Giải pháp bảo mật có thể triển khai cho ứng dụng

#### 1. Sử dụng Amazon VPC (Virtual Private Cloud)

Isolated Environment (Môi trường tách biệt): Tất cả các thành phần của ứng dụng sẽ được triển khai trong VPC, nơi tạo ra một mạng riêng ảo được kiểm soát hoàn toàn.

Public/Private Subnet (Mạng con công khai và riêng tư): Thiết lập các subnet riêng tư cho cơ sở dữ liệu (RDS) và mạng con công khai cho các máy chủ ứng dụng (EC2). Điều này giúp bảo vệ cơ sở dữ liệu khỏi truy cập trực tiếp từ internet, chỉ cho phép truy cập thông qua các thành phần nội bộ.

#### 2. Security Groups (Nhóm bảo mật)

Thiết lập Security Groups: AWS Security Groups đóng vai trò như một tường lửa cho các máy EC2, RDS và các dịch vụ khác. Security Groups sẽ giới hạn lưu lượng truy cập đến và đi từ

ứng dụng, chỉ cho phép lưu lượng từ các IP và cổng cụ thể. Cấu hình đúng Security Groups là rất quan trọng để ngăn chặn truy cập trái phép.

### 3. IAM (Identity and Access Management)

Phân quyền và quản lý truy cập: IAM cho phép quản lý chính xác quyền truy cập của người dùng và nhóm. Chỉ những người dùng có quyền nhất định mới có thể truy cập hoặc thao tác trên tài nguyên. Ví dụ, chỉ admin mới có quyền thay đổi cấu hình hệ thống, trong khi người dùng bình thường chỉ có thể truy cập dữ liệu mình cần.

### 4. Bảo mật cơ sở dữ liệu (RDS Security)

Sử dụng RDS IAM Authentication: Quản lý truy cập đến cơ sở dữ liệu MySQL qua việc tích hợp với IAM, điều này giúp bảo mật việc quản lý và truy cập dữ liệu.

Backup và Recovery: RDS cho phép thiết lập tự động backup và phục hồi dữ liệu để đảm bảo tính toàn vẹn của dữ liệu trong trường hợp có sự cố xảy ra.

### 5. CloudTrail và CloudWatch

Theo dõi và ghi lại các hành vi truy cập: AWS CloudTrail ghi lại tất cả các yêu cầu API và các hành động liên quan đến tài khoản AWS, giúp theo dõi và xác định hành vi bất thường. CloudWatch cung cấp công cụ giám sát hoạt động của ứng dụng và cảnh báo khi có sự cố hoặc khi đạt đến ngưỡng tài nguyên.

### 6. Auto Scaling và Load Balancing

Load Balancer bảo vệ khỏi các cuộc tấn công traffic: AWS Elastic Load Balancer giúp phân phối lưu lượng đến các máy chủ web, ngăn chặn các cuộc tấn công tắc nghẽn lưu lượng. Bên cạnh đó, Auto Scaling cho phép tự động tăng hoặc giảm số lượng máy chủ khi cần thiết, đảm bảo tính khả dụng cao.

⇒ Các giải pháp bảo mật mà AWS cung cấp cho ứng dụng web không chỉ đảm bảo tính toàn vẹn của dữ liệu và tài nguyên, mà còn giúp ứng dụng chống lại các cuộc tấn công từ mạng internet. Việc triển khai các giải pháp bảo mật như VPC, Security Groups, mã hóa dữ liệu, WAF và các biện pháp bảo mật nâng cao khác là cần thiết để bảo vệ ứng dụng trong môi trường đám mây.



# Chương 4: Phân tích chi phí / Triển khai các tính năng nâng cao

## 4.1. Tóm tắt ước tính chi phí

+ EC2 Instances: Sử dụng máy ảo (EC2) để chạy ứng dụng web. Chi phí tùy thuộc vào loại instance:

On-demand: Tính theo giờ sử dụng, chi phí khoảng \$8/tháng cho instance t2.micro.

Reserved: Giảm chi phí từ 30-70% khi cam kết dài hạn.

Spot: Tiết kiệm đến 90% nhưng dễ bị dừng nếu tài nguyên cần.

+ RDS (Cơ sở dữ liệu): Quản lý cơ sở dữ liệu MySQL với Amazon RDS, chi phí từ \$15/tháng cho instance db.t3.micro và thêm chi phí lưu trữ dữ liệu.

+ Load Balancer (Cân bằng tải): Sử dụng Elastic Load Balancer (ELB) để phân phối lưu lượng, chi phí khoảng \$20/tháng.

+ S3 (Lưu trữ): Lưu trữ dữ liệu tĩnh trên S3, chi phí \$0.023/GB/tháng cho lưu trữ tiêu chuẩn.

+ CloudWatch (Giám sát): Giám sát hệ thống và thiết lập cảnh báo, chi phí khoảng \$5/tháng cho các chỉ số cơ bản.

+ Tổng chi phí ước tính: Khoảng \$50-\$60/tháng cho ứng dụng web cơ bản.

## 4.2. Các phương án tiết kiệm chi phí

1. Reserved Instances (RI) cho EC2 và RDS: Nếu ứng dụng hoạt động dài hạn, lựa chọn RI có thể giúp tiết kiệm 30-70% chi phí so với on-demand.

2. Auto Scaling: Tận dụng Auto Scaling để tự động tăng/giảm số lượng máy chủ theo nhu cầu thực tế, tránh lãng phí tài nguyên trong các thời điểm không tải.

3. Sử dụng Spot Instances: Cho các tác vụ tính toán không yêu cầu thời gian liên tục (như batch jobs), sử dụng Spot instances có thể tiết kiệm đến 90% chi phí.

## 4.3. Tính năng nâng cao

### 1. Balancing (Cân bằng tải)

Elastic Load Balancer (ELB): AWS ELB đảm bảo phân phối đều tải đến các EC2 instances trong Auto Scaling Group, giúp duy trì tính khả dụng cao cho ứng dụng.

Lợi ích: Cân bằng tải giúp ứng dụng hoạt động ổn định và xử lý tốt hơn khi có nhiều yêu cầu từ người dùng. Nó tự động điều chỉnh lưu lượng giữa các vùng sẵn có (Availability Zones), tăng cường khả năng chịu lỗi của hệ thống.

### 2. Scaling (Tự động điều chỉnh tài nguyên)

Auto Scaling: AWS Auto Scaling tự động điều chỉnh số lượng EC2 instances theo nhu cầu thực tế của ứng dụng, dựa trên các quy tắc như CPU usage hoặc memory usage.

Lợi ích: Tiết kiệm chi phí bằng cách chỉ sử dụng tài nguyên khi cần thiết, đồng thời đảm bảo rằng ứng dụng có đủ tài nguyên để xử lý tải lớn khi có nhiều người truy cập.

### 3. Monitoring (Giám sát)

Amazon CloudWatch: Cung cấp khả năng giám sát toàn diện hệ thống và thiết lập các cảnh báo dựa trên các chỉ số quan trọng như CPU, Memory, Network traffic...

Lợi ích: CloudWatch cho phép phát hiện các vấn đề hiệu suất và bảo mật, giúp bạn tối ưu hóa hiệu quả ứng dụng và tài nguyên hệ thống.

⇒ Chi phí vận hành ứng dụng trên AWS có thể điều chỉnh linh hoạt dựa trên nhu cầu sử dụng tài nguyên và các dịch vụ. Việc lựa chọn đúng cấu hình và tận dụng các dịch vụ tiết kiệm chi phí như Reserved Instances, Auto Scaling và Spot Instances sẽ giúp tối ưu hóa chi phí mà vẫn đảm bảo hiệu suất và tính sẵn sàng của hệ thống. Đồng thời, triển khai các tính năng nâng cao như Balancing, Scaling sẽ tăng cường khả năng tự động hóa và bảo mật của ứng dụng.

# Chương 5: Triển khai ứng dụng, kiểm thử và đánh giá (PI 2.1)

## 5.1 Kết quả triển khai ứng dụng:

### 5.1.1. cloud9-scripts.yml

#Following scripts are to be run on the Cloud9 terminal

#Individual scripts are also mentioned in the Instructors Guide stepwise

Script-1:

# Use following script to create secret in secret manager for accessing your database via the web application

# Replace the values for below placeholders with the actual values you have used to configure the service

#<RDS Endpoint>

#<password>

#<username>

#<dbname>

aws secretsmanager create-secret \

--name Mydbsecret \

--description "Database secret for web app" \

```
--secret-string      "{\"user\":\"nodeapp\",\"password\":\"student12\",\"host\":\"student-  
db.c59uoqrjxnvg.us-east-1.rds.amazonaws.com\",\"db\":\"STUDENTS\"}"
```

Script-2:

```
## Load testing
```

```
#Following command installs #loadtest package to perform load testing on the application
```

```
#Prerequisites are mentioned in the resources section
```

```
npm install -g loadtest
```

```
#Following command performs load testing on the given URL. replace the URL with  
Loadbalancer (or Public IP of EC2 instance)
```

```
# Press ctrl +C to stop the script
```

```
loadtest --rps 1000 -c 500 -k http://student-alb-832979266.us-east-1.elb.amazonaws.com
```

Script -3:

```
## Migration
```

```
# Following command exports the data from existing server.
```

```
# Prerequisites - This script expects mysql-client, mysqldump to be present
```

```
# AWS Cloud9 environment already has all the prerequisites installed for running these scripts
```

# Replace the <EC2instancePrivateip> with internal IP address of the EC2 instance (CapstonePOC) created in Phase-2 earlier.

# Provide the password when prompted

```
mysqldump -h 10.0.0.8 -u nodeapp -p --databases STUDENTS > data.sql
```

#Following command imports the data into RDS database. Replace <RDSEndpoint> with the RDS Database endpoint you noted after RDS Database created in earlier steps.

#when prompted, enter password you provided during the time of database creation

```
mysql -h student-db.c59uoqrjxnvg.us-east-1.rds.amazonaws.com -u nodeapp -p STUDENTS < data.sql
```

### **5.1.2. UserdataScript-phase-2.sh**

```
#!/bin/bash -xe
```

```
apt update -y
```

```
apt install nodejs unzip wget npm mysql-server -y
```

```
#wget      https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCAP1-1-DEV/code.zip -P /home/ubuntu
```

```
wget      https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCAP1-1-79581/1-lab-capstone-project-1/code.zip -P /home/ubuntu
```

```
cd /home/ubuntu
```

```
unzip code.zip -x "resources/codebase_partner/node_modules/*"
```

```
cd resources/codebase_partner
```

```
npm install aws aws-sdk
```

```
mysql -u root -e "CREATE USER 'nodeapp' IDENTIFIED WITH mysql_native_password BY 'student12';"
```

```
mysql -u root -e "GRANT all privileges on *.* to 'nodeapp'@'%';"
```

```
mysql -u root -e "CREATE DATABASE STUDENTS;"
```

```
mysql -u root -e "USE STUDENTS; CREATE TABLE students(
```

```
    id INT NOT NULL AUTO_INCREMENT,
```

```
    name VARCHAR(255) NOT NULL,
```

```
    address VARCHAR(255) NOT NULL,
```

```
    city VARCHAR(255) NOT NULL,
```

```
    state VARCHAR(255) NOT NULL,
```

```
    email VARCHAR(255) NOT NULL,
```

```
    phone VARCHAR(100) NOT NULL,
```

```
    PRIMARY KEY ( id ));"
```

```
sed -i 's/.*/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf
```

```
systemctl enable mysql
```

```
service mysql restart
```

```
export APP_DB_HOST=$(curl http://169.254.169.254/latest/meta-data/local-ipv4)
```

```
export APP_DB_USER=nodeapp
```

```
export APP_DB_PASSWORD=student12
```

```
export APP_DB_NAME=STUDENTS
```

```
export APP_PORT=80
```

```
npm start &
```

```
echo '#!/bin/bash -xe
```

```
cd /home/ubuntu/resources/codebase_partner
```

```
export APP_PORT=80
```

```
npm start' > /etc/rc.local
```

```
chmod +x /etc/rc.local
```

### **5.1.3. UserdataScript-phase-3.sh**

```
#!/bin/bash -xe
```

```
apt update -y
```

```
apt install nodejs unzip wget npm mysql-client -y
```

```
#wget      https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCAP1-1-DEV/code.zip -P /home/ubuntu
```

```
wget      https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCAP1-1-79581/1-lab-capstone-project-1/code.zip -P /home/ubuntu
```

```
cd /home/ubuntu
```

```
unzip code.zip -x "resources/codebase_partner/node_modules/*"
```

```
cd resources/codebase_partner
```

```
npm install aws aws-sdk
```

```
export APP_PORT=80
```

```
npm start &
```

```
echo '#!/bin/bash -xe
```

```
cd /home/ubuntu/resources/codebase_partner
```

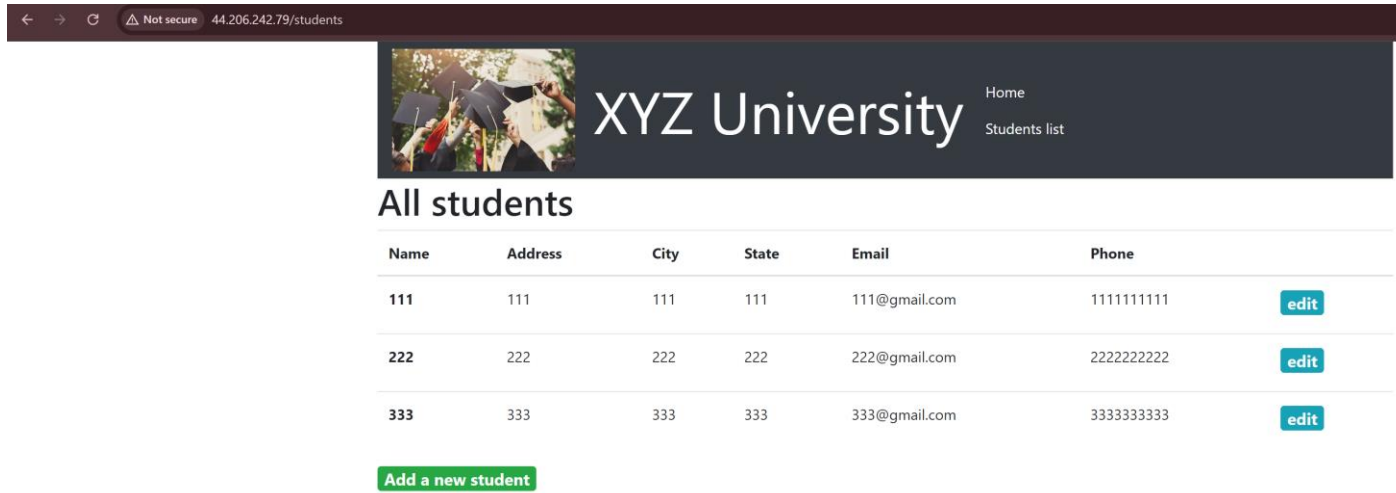
```
export APP_PORT=80
```

```
npm start' > /etc/rc.local
```

```
chmod +x /etc/rc.local
```

## 5.2. Demo

- Student 1

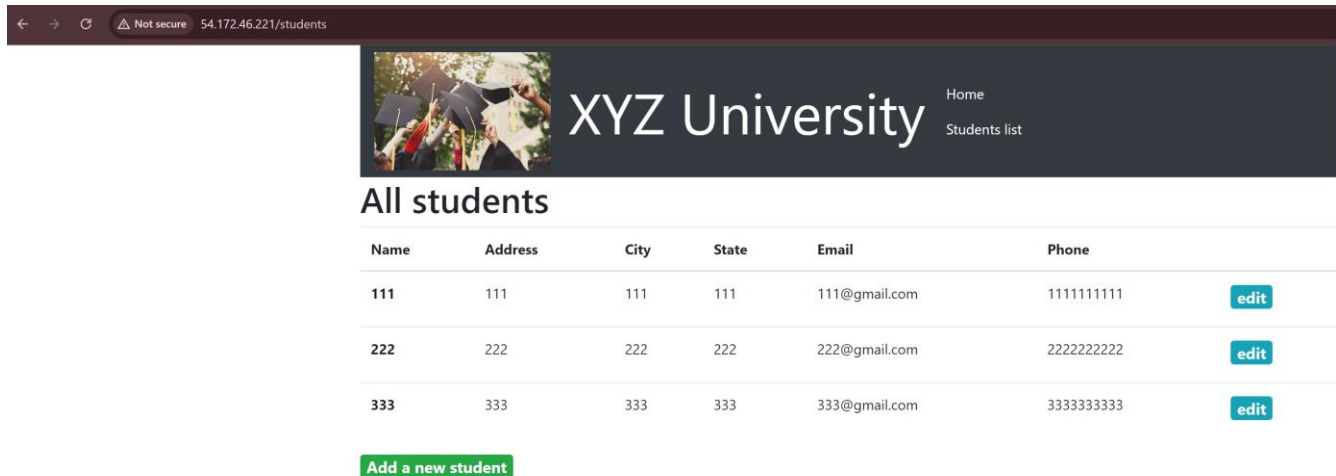


Name	Address	City	State	Email	Phone	
111	111	111	111	111@gmail.com	1111111111	<a href="#">edit</a>
222	222	222	222	222@gmail.com	2222222222	<a href="#">edit</a>
333	333	333	333	333@gmail.com	3333333333	<a href="#">edit</a>

[Add a new student](#)

*Figure 1: Student 1*

- Student 2




Name	Address	City	State	Email	Phone	
111	111	111	111	111@gmail.com	1111111111	<a href="#">edit</a>
222	222	222	222	222@gmail.com	2222222222	<a href="#">edit</a>
333	333	333	333	333@gmail.com	3333333333	<a href="#">edit</a>

[Add a new student](#)

*Figure 2: Student 2*



- Thêm Student



# XYZ University

[Home](#)  
[Students list](#)


## All students

Name	Address	City	State	Email	Phone	
111	111	111	111	111@gmail.com	1111111111	<a href="#">edit</a>
222	222	222	222	222@gmail.com	2222222222	<a href="#">edit</a>
333	333	333	333	333@gmail.com	3333333333	<a href="#">edit</a>
444	444	444	444	444@gmail.com	4444444444	<a href="#">edit</a>

Add a new student

Figure 3: Thêm Student

- Sửa Student



# XYZ University

[Home](#)  
[Students list](#)


## All students

Name	Address	City	State	Email	Phone	
111	111	111	111	111@gmail.com	1111111111	<a href="#">edit</a>
222	222	222	222	222@gmail.com	2222222222	<a href="#">edit</a>
333	333	333	333	333@gmail.com	3333333333	<a href="#">edit</a>
555	555	555	555	555@gmail.com	5555555555	<a href="#">edit</a>

Add a new student

Figure 4: Sửa Student

## - Xóa Student



# XYZ University

[Home](#)[Students list](#)

[Add a new student](#)

Figure 5: Xóa Student

## - Load Balancers

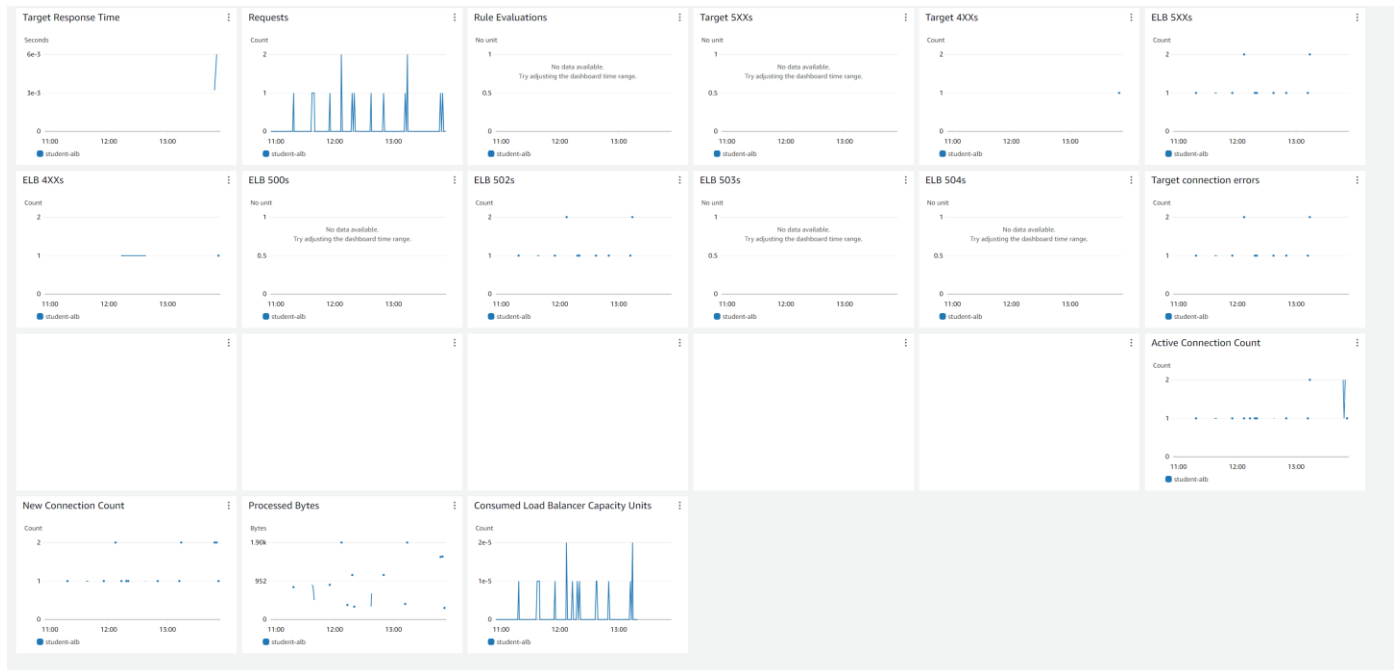
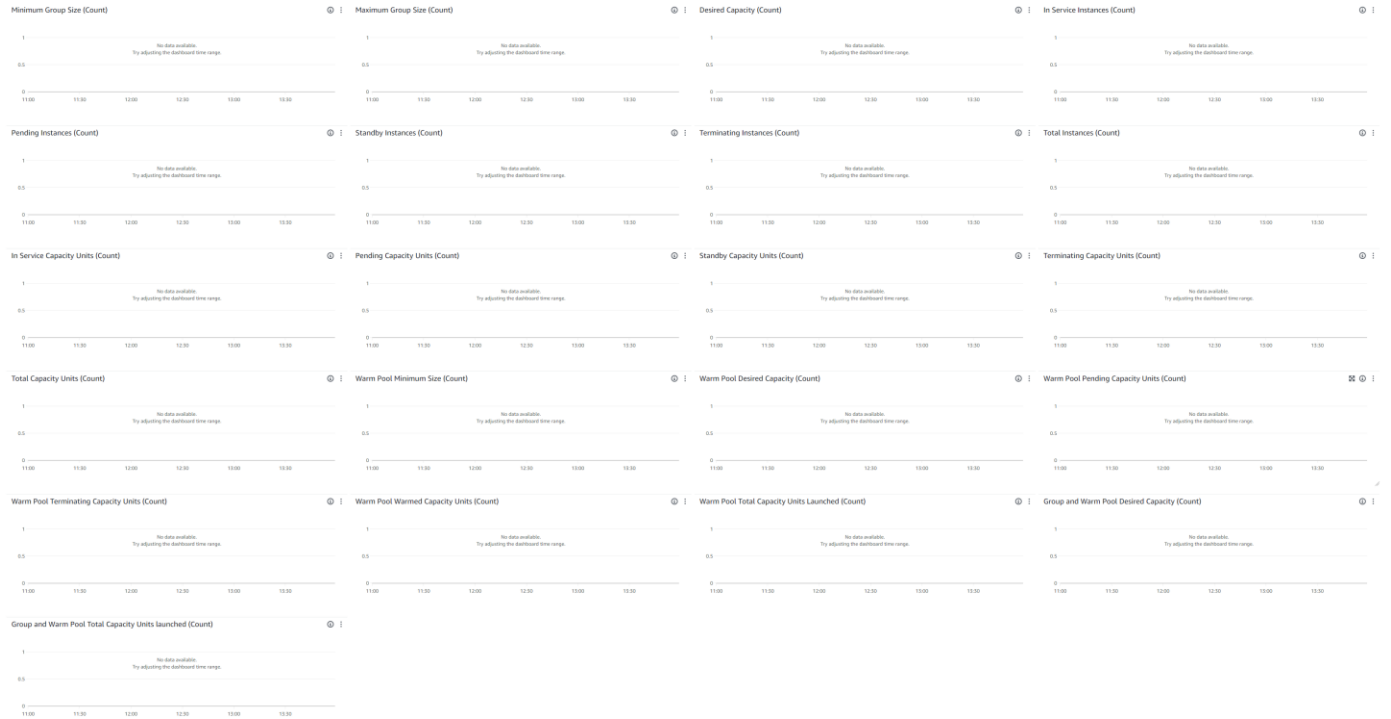


Figure 6: Load Balancers

## - Auto Scaling Group



*Figure 7: Auto Scaling Group*

### 5.3. Kết luận:

- Những tính năng của ứng dụng
- + Ứng dụng đã có thể chạy được thành công, ứng dụng khá đơn giản và dễ sử dụng.
- + Ứng dụng có đủ các chức năng cơ bản như thêm, sửa, xóa cơ sở dữ liệu.
- + Các tính năng cần thiết như bảo mật, tính khả dụng cao, khả năng mở rộng và hiệu suất đều tốt.
- + Ứng dụng có đầy đủ các tính năng chính.
- + Chi phí duy trì ứng dụng ở mức thấp.
- + Ứng dụng sử dụng rất nhiều tính năng của đám mây khá tốt.
- + Ứng dụng được bảo mật khá tốt.
- + Mọi người đều có thể sử dụng được.

- Nhược điểm của ứng dụng
  - + Ứng dụng còn đơn giản, sơ sài.
  - + Ứng dụng còn ít tính năng.
  - + Giao diện của ứng dụng đơn giản.
- Đề xuất cải tiến
  - + Nâng cao giao diện ứng dụng.
  - + Sử dụng thêm nhiều tính năng mới hơn.
  - + Giảm chi phí ứng dụng.
  - + Nâng cao ứng dụng để có thể tiếp cận với nhiều người hơn.
  - + Cải thiện hiệu suất ứng dụng.
  - + Tăng cường bảo mật.
  - + Tối ưu hóa trải nghiệm người dùng.
  - + Tích hợp hệ thống phân tích dữ liệu.