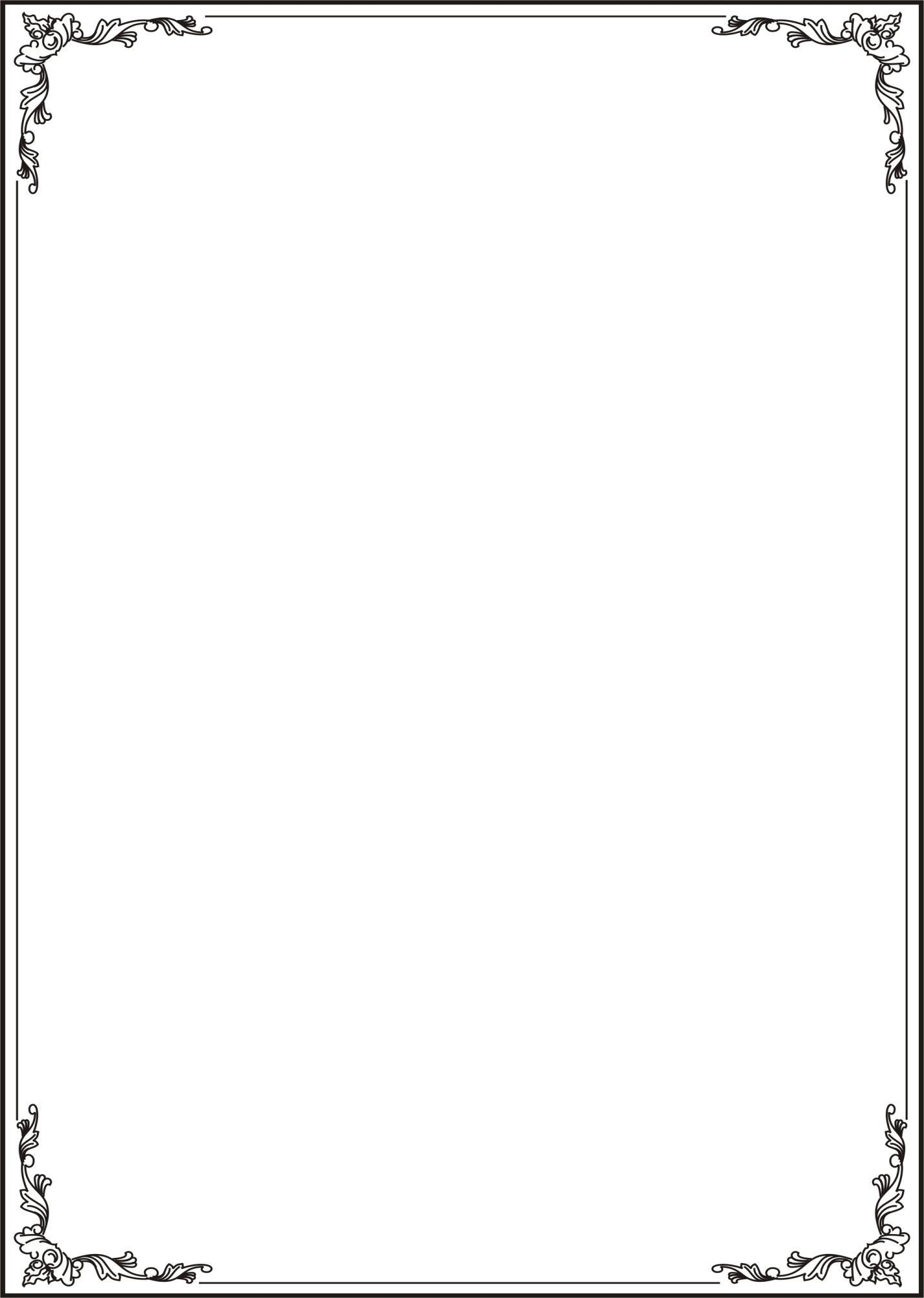
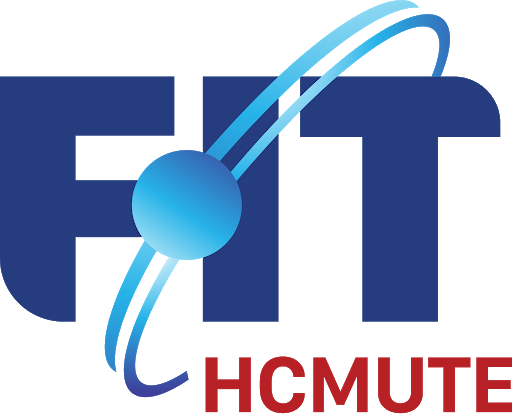
**.**

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

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN ĐIỆN TOÁN ĐÁM MÂY**

-----🙞🙜🕮🙞🙜-----

****

****



****

**KHÓA 2019-2023**

| **ĐH SƯ PHẠM KỸ THUẬT TP. HCM**  **KHOA CNTT**  \*\*\*\*\*\*\* | **CỘNG HÒA 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: **Lý Quốc Dũng** MSSV 1: **19133015**

Họ và tên Sinh viên 2: **Nguyễn Huỳnh Minh Trung** MSSV 2: **19133060**

Họ và tên Sinh viên 2: **Bùi Thị Ngân Tuyền** MSSV 2: **19133066**

Ngành: **Kỹ thuật dữ liệu**

Tên đề tài: **Xây dựng ứng dụng trên AWS cho phép tạo database và cung cấp API để thêm, xóa sửa trên database**

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

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

2. Ưu điểm:

3. Khuyết điểm:

4. Đề nghị cho bảo vệ hay không?

5. Đánh giá loại:

6. Điểm:

| *Tp. Hồ Chí Minh, ngày tháng năm 2022* | | |
| --- | --- | --- |
|  | Giáo viên hướng dẫn  *(Ký & ghi rõ họ tên)* |

# 

# **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 Ứng Dụng Trên AWS Cho Phép Tạo Database Và Cung Cấp API Để Thêm, Xóa, Sửa Trên Database.*

*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!*

***Nhóm thực hiện***



# **MỤC LỤC**

[**LỜI CẢM ƠN**](#_heading=h.30j0zll) **3**

[**MỤC LỤC**](#_heading=h.1fob9te) **4**

[**DANH MỤC CÁC HÌNH**](#_heading=h.3znysh7) **6**

[**PHẦN MỞ ĐẦU**](#_heading=h.2et92p0) **6**

[**1. Tính cấp thiết của đề tài**](#_heading=h.tyjcwt) **7**

[**2. Đối tượng nghiên cứu**](#_heading=h.3dy6vkm) **7**

[**3. Phạm vi nghiên cứu**](#_heading=h.1t3h5sf) **7**

[**4. Kết quả dự kiến đạt được**](#_heading=h.4d34og8) **7**

[**PHẦN NỘI DUNG**](#_heading=h.ztmddar5imb) **8**

[**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**](#_heading=h.17dp8vu) **8**

[**Tổng quan về Lambda**](#_heading=h.3rdcrjn) **8**

[**Các tính năng của Lambda**](#_heading=h.p70bzuuhegf) **8**

[**. Các dịch vụ liên quan đến Lambda**](#_heading=h.z6lufxy5z92x) **9**

[**Tổng quan về DynamoDB**](#_heading=h.ohmqw8sh96in) **10**

[**. Các tính năng của DynamoDB**](#_heading=h.tnk5a3epviyu) **11**

[**CHƯƠNG 2: THỰC HIỆN**](#_heading=h.26in1rg) **12**

[**2.1 Cài đặt**](#_heading=h.lnxbz9) **12**

[*2.1.1 Cài đặt Function Lamda-*](#_heading=h.35nkun2) *12*

[**CreateTable: Function dùng để tạo bảng trong DynamoDB**](#_heading=h.2yq0y55w0un0) **12**

[**2.1.2 Cài đặt Simple Queue Service:**](#_heading=h.6xpehqxgtfom) **15**

[*2.1.3 Setup Web (PHP):*](#_heading=h.44sinio) *15*

[**2.2 Triển khai Project:**](#_heading=h.2jxsxqh) **19**

[**2.2.1. Local:**](#_heading=h.8lbev8y0o0w) **19**

[**2.2.2. EC2:**](#_heading=h.fwmdfafau0sz) **21**

[**2.3 Thiết kế DynamoDB minh họa cho việc tương tác:**](#_heading=h.z337ya) **21**

[**Danh sách các bảng**](#_heading=h.3whwml4) **21**

[**PHẦN KẾT LUẬN**](#_heading=h.vryggtmqd972) **22**

[**Kết quả đạt được**](#_heading=h.2bn6wsx) **22**

[*1.1. Kiến thức tìm hiểu được*](#_heading=h.qsh70q) *22*

[*1.2. Chương trình đã làm được*](#_heading=h.3as4poj) *22*

[**Ưu điểm**](#_heading=h.1pxezwc) **22**

[**Nhược điểm**](#_heading=h.49x2ik5) **23**

[**Hướng phát triển**](#_heading=h.2p2csry) **23**

[**TÀI LIỆU THAM KHẢO**](#_heading=h.147n2zr) **24**

# **DANH MỤC CÁC HÌNH**

[**Hình 1: CreateTable** 1](#_heading=h.2grqrue)2

[**Hình 2: DeleteTable** 1](#_heading=h.vx1227)2

[**Hình 3: CreateItem** 1](#_heading=h.3fwokq0)3

[**Hình 4: UpdateItem** 1](#_heading=h.3fwokq0)4

[**Hình 5: DeleteItem** 1](#_heading=h.3fwokq0)4

[**Hình 6: Tạo SQS** 1](#_heading=h.3fwokq0)5

[**Hình 7: Source Code Web** 1](#_heading=h.3fwokq0)5

[**Hình 8: Hàm Call API** 1](#_heading=h.3fwokq0)6

[**Hình 9: Giao diện danh sách bảng** 1](#_heading=h.3fwokq0)7

[**Hình 10: Giao diện Thêm Bảng** 1](#_heading=h.3fwokq0)8

[**Hình 11: Giao diện danh sách Items** 1](#_heading=h.3fwokq0)8

[**Hình 12: Giao diện thêm/sửa bảng**](#_heading=h.3fwokq0) 19

[**Hình 13: Config SQS vào Source code PHP**](#_heading=h.3fwokq0) 20

[**Hình 14: Config API vào Source code PHP**](#_heading=h.3fwokq0) 20

[**Hình 15: Danh sách các bảng DynamoDB**](#_heading=h.3fwokq0) 21

[**Hình 16: Danh sách Items**](#_heading=h.3fwokq0) 22

# **PHẦN 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.

## **2. Đối tượng nghiên cứu**

Đối với đề tài này, đối tượng nghiên cứu là Lambda, DynamoDB. Đồng thời kết hợp với các kiến thức về Front-end và Back-end để tạo một trang web cụ thể.

## **3. Phạm vi nghiên cứu**

Đề tài này chủ yếu tập trung vào việc kết hợp sử dụng Lambda và DynamoDB nhằm thực hiện các câu truy vấn để thêm, sửa, xóa dữ liệu trên database.

## **4. Kết quả dự kiến đạt được**

* Cài đặt các giao diện
* Cài Đặt phía backend
* Thêm, xóa các table trong DynamoDB
* Thêm, sửa, xóa các item trong DynamoDB

# 

# **PHẦN NỘI DUNG**

# **CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

## **Tổng quan về Lambda**

AWS Lamda là nơi để cho lập trình viên upload các function code của mình dưới dạng các ngôn ngữ được hỗ trợ, và AWS Lamda sẽ giúp người lập trình chạy các đoạn code đó dựa trên các thư viện có sẵn của AWS, đồng thời AWS Lamda sẽ lưu lại các tiến trình thay đổi của function và lưu vào CloudWatch.

AWS Lamda được hỗ trợ và quản lý bởi đội ngũ AWS nên các function sẽ thực hiện code của lập trình viên khi cần, và có khả năng tự động mở rộng, từ một vài request đến hàng trăm ngàn request trên giây. Nhờ vào các ưu điểm này mà Lamda có thể dễ dàng sử dụng để xây dựng các ứng dụng cùng với các dịch vụ khác của Lamda như S3, DynamoDB,... với các hiệu suất và độ bảo mật cao

## **Các tính năng của Lambda**

Các tính năng chính sau đây giúp bạn phát triển các ứng dụng Lambda có khả năng mở rộng, bảo mật và dễ dàng mở rộng:

* Khả năng mở rộng

AWS Lamda cho phép lập trình viên dễ dàng kết hợp với các dịch vụ khác như S3, DyanmoDB,.... vì vậy có thể dễ dàng kích hoạt các Lamda Function, khi có các luồng dữ liệu hoặc tài nguyên có sự thay đổi giúp lập trình viên hiểu được các tiến trình của các Request

* Xây dựng Back-end

AWS Lamda cho phép lập trình viên có thể tạo các backend cho ứng dụng bằng cách sử dụng các API Lamda hoặc API endpoint của Amazon API Gateway. Lamda sẽ tiến thành xử lý các event thay vì cung cấp các event lên client, nhờ đó giúp giảm tiêu hao các tài nguyên, dễ dàng bảo trì cũng như cập nhật hơn.

* Sử dụng code riêng

AWS Lamda hỗ trợ đa dạng các ngôn ngữ cũng như cung cấp đầy đủ các API Runtime, ngoài ra AWS còn hỗ trợ lập trình viên có thể sử dụng bất kỳ thư viên, kể cả các thư viện thuần, đa dạng các framework, lập trình viên cũng có thể đóng gói các source code dưới dạng các class của Lamdam, dễ dàng chia sẻ cũng như quản lý các code trên nhiều Function khác nhau.

* Các quản lý được tự động hoàn toàn

Lamda được quản lý hoàn toàn bởi các đội ngũ AWS nên có độ sẵn sàng và khả năng chịu lỗi cao, ngoài ra lập trình viên sẽ không cần cập nhật các bản vá cũng như cập nhật dung lượng, bổ sung máy chủ mới. AWS sẽ thực hiện tất cả các hoạt động bảo trì, vá lỗi, cung cấp chức năng lưu trữ nhật ký của các quá trình Lamda thông qua Amazon CloudWatch

## **. Các dịch vụ liên quan đến Lambda**

Lambda tích hợp với các dịch vụ AWS khác để gọi các hàm dựa trên các sự kiện được chỉ định:

* API Gateway cung cấp các đường dẫn có tính bảo mật cao, có thể mở rộng với các API web định tuyến đến các request HHTP và kích hoạt đến các Lamda Function
* Đối với các dịch vụ tạo hàng đợi hoặc các luồng dữ liệu (chẳng hạn như DynamoDB và Kinesis), Lambda sẽ xem xét các hàng đợi hoặc luồng dữ liệu từ dịch vụ và gọi Function để xử lý dữ liệu của các hàng đợi hoặc luồng dữ liệu.
* Xác định các sự kiện Amazon S3 gọi một hàm Lambda để xử lý các đối tượng Amazon S3, chẳng hạn như khi một đối tượng được tạo hoặc xóa.
* Sử dụng chức năng Lambda để xử lý các thông báo Amazon SQS messages hoặc Amazon Simple Notification Service (Amazon SNS).
* Sử dụng Chức năng từng bước AWS để kết nối các chức năng Lambda với nhau thành quy trình làm việc không máy chủ được gọi là máy trạng thái.

## **Tổng quan về DynamoDB**

DynamoDB là cơ sở dữ liệu NoSQL, được lưu trữ trong SSD thực hiện nhanh và linh hoạt trên các ứng dụng khác nhau có yêu cầu độ trễ mili giây ở bất kỳ quy mô nào, những ứng dụng này có thể hoạt động trên quy mô nhỏ sau đó mở rộng ra để hỗ trợ nhiều petabye dữ liệu, xử lý hàng ngàn request cread hoặc ghi mỗi giây.

DynamoDB được quản lý đầy đủ hỗ trợ cả mô hình dữ liệu dưới dạng tài liệu và cặp value-key.

Mô hình dữ liệu rất linh hoạt và hiệu suất cao phù hợp cho nhiều loại ứng dụng, nền tảng khác nhau.

## **. Các tính năng, dịch vụ liên quan đến DynamoDB**

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.

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. **. Tại sao nên sữ dụng Dynamodb?**

Tính tự 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 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 xác định được ngưỡng trên của cơ sở dữ liệu, 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 không quan hệ mà DynamoDB có đặc tính rất phù hợp với những đối tượng không có cấu trúc cố định và thay đổi theo thời gian.

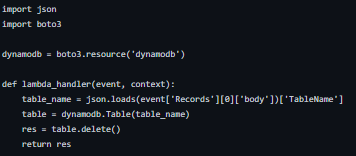
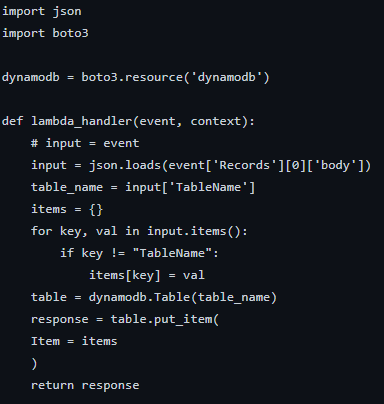
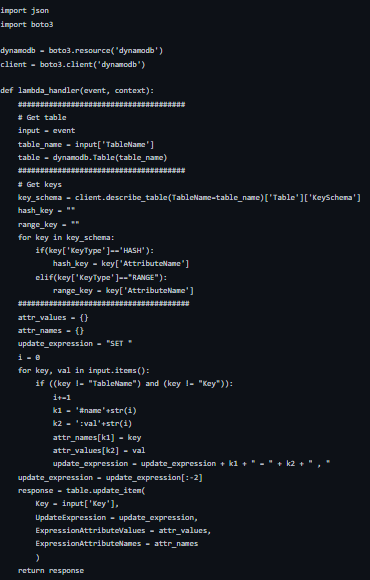
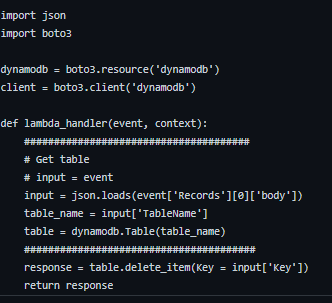
Tự động quản lý: Với mục đích bảo mật và an toàn mà dữ liệu trong DynamoDb được tự động liên tục cập nhật và backup lên cloud.

**CHƯƠNG 2: THỰC HIỆN**

## **2.1 Cài đặt**

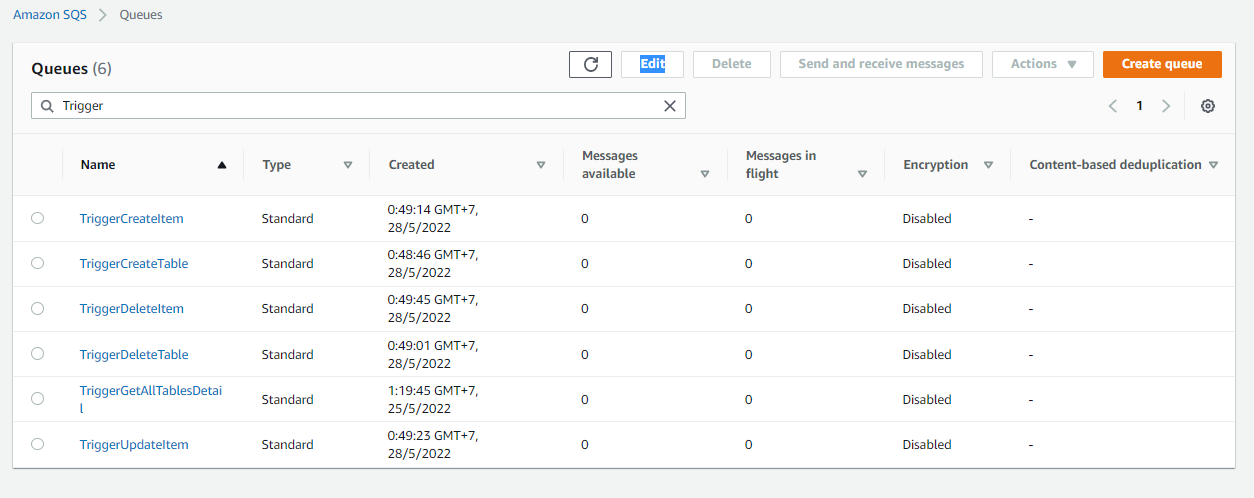
### *2.1.1 Cài đặt Function Lamda*-

#### CreateTable: Function dùng để tạo bảng trong DynamoDB

* DeleteTable: Function dùng để xóa bảng trong DynamoDB 
* CreateItem: Function dùng để tạo item trong DynamoDB
* UpdateItem: Function dùng để cập nhật các item trong DynamoDB 
* DeleteItem: Function dùng để xóa các item trong DynamoDB

### *2.1.2 Cài đặt Simple Queue Service:*

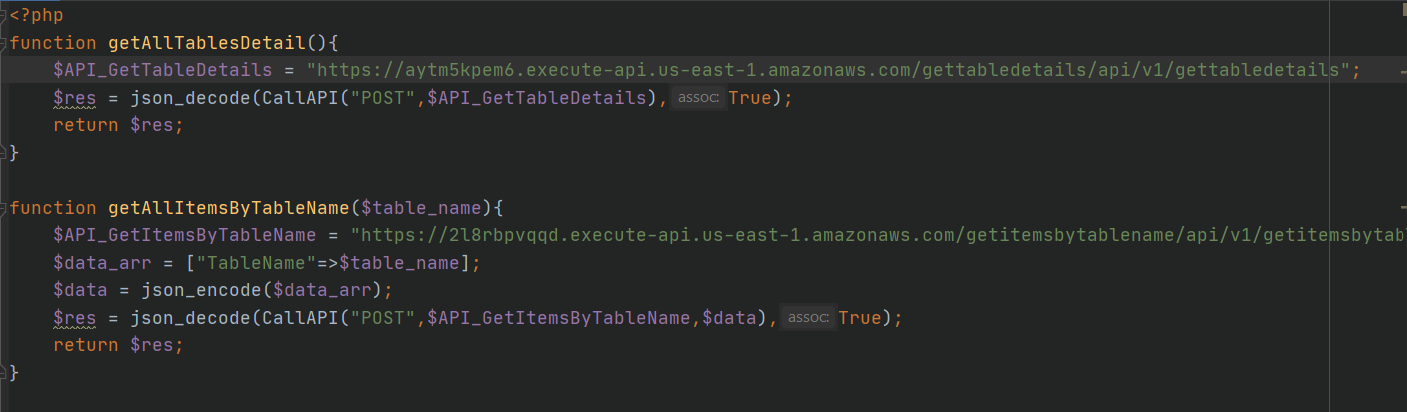
Lần Lượt tạo 5 SQS và kết nối đến các Function Lamda đã tạo:

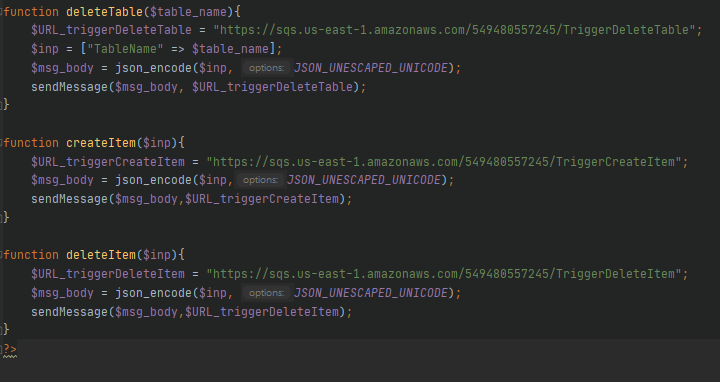
* TriggerCreateItem: trigger đến Lamda tên CreateItem
* TriggerCreateTable: trigger đến Lamda tên CreateTable
* TriggerDeleteItem: trigger đến Lamda tên DeleteItem
* TriggerDeleteTable: trigger đến Lamda tên DeleteTable
* TriggerUpdateItem: trigger đến Lamda tên UpdateItem
* 

### *2.1.3 Setup Web (PHP):*

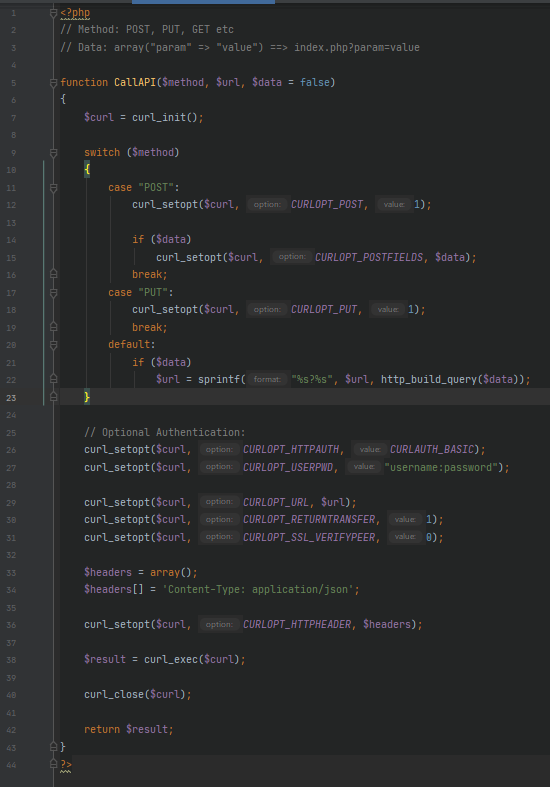
* Source Code:

**Tất cả các hàm sử dụng API để tương tác với Dynamodb**



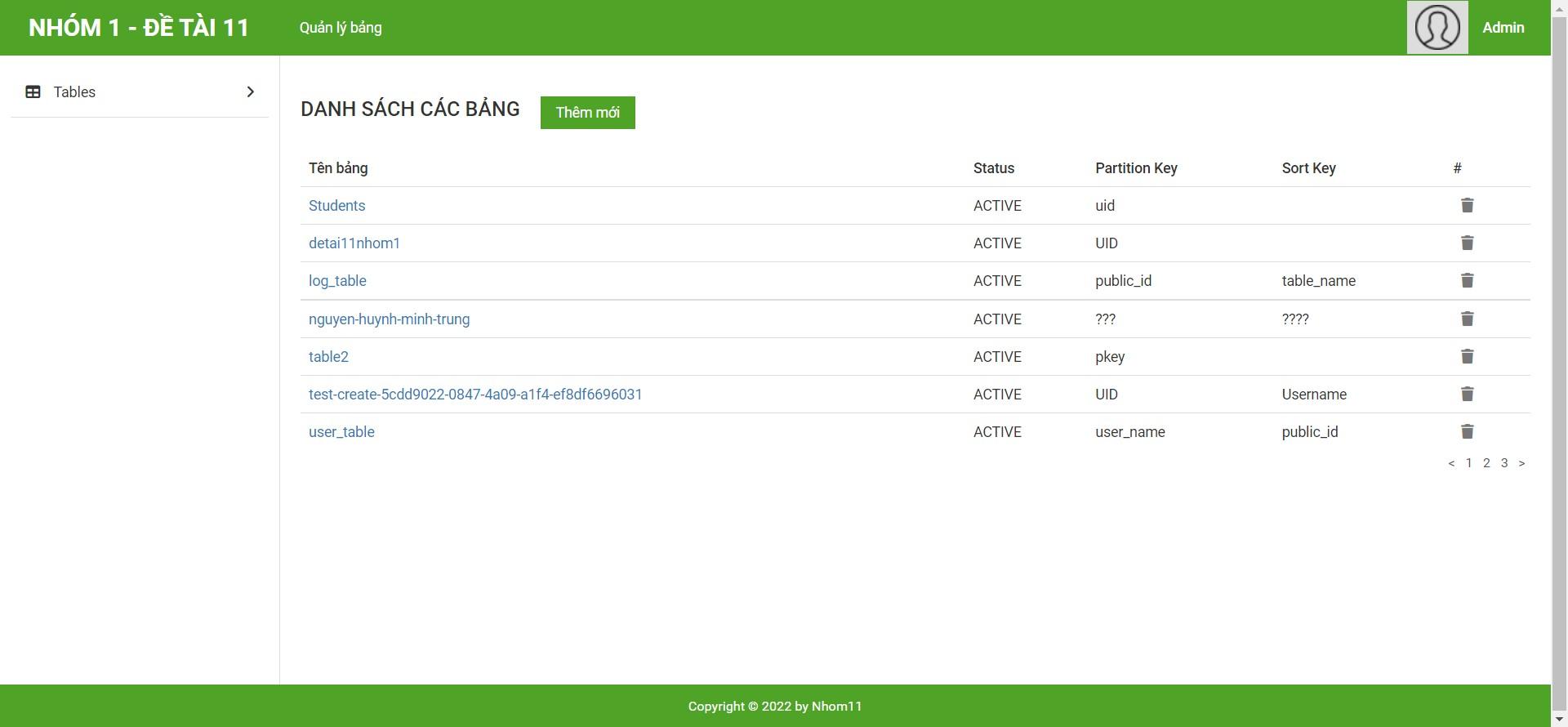


**Hàm Call API cho phép gửi request đến một API và nhận kết quả trả về của API đó**

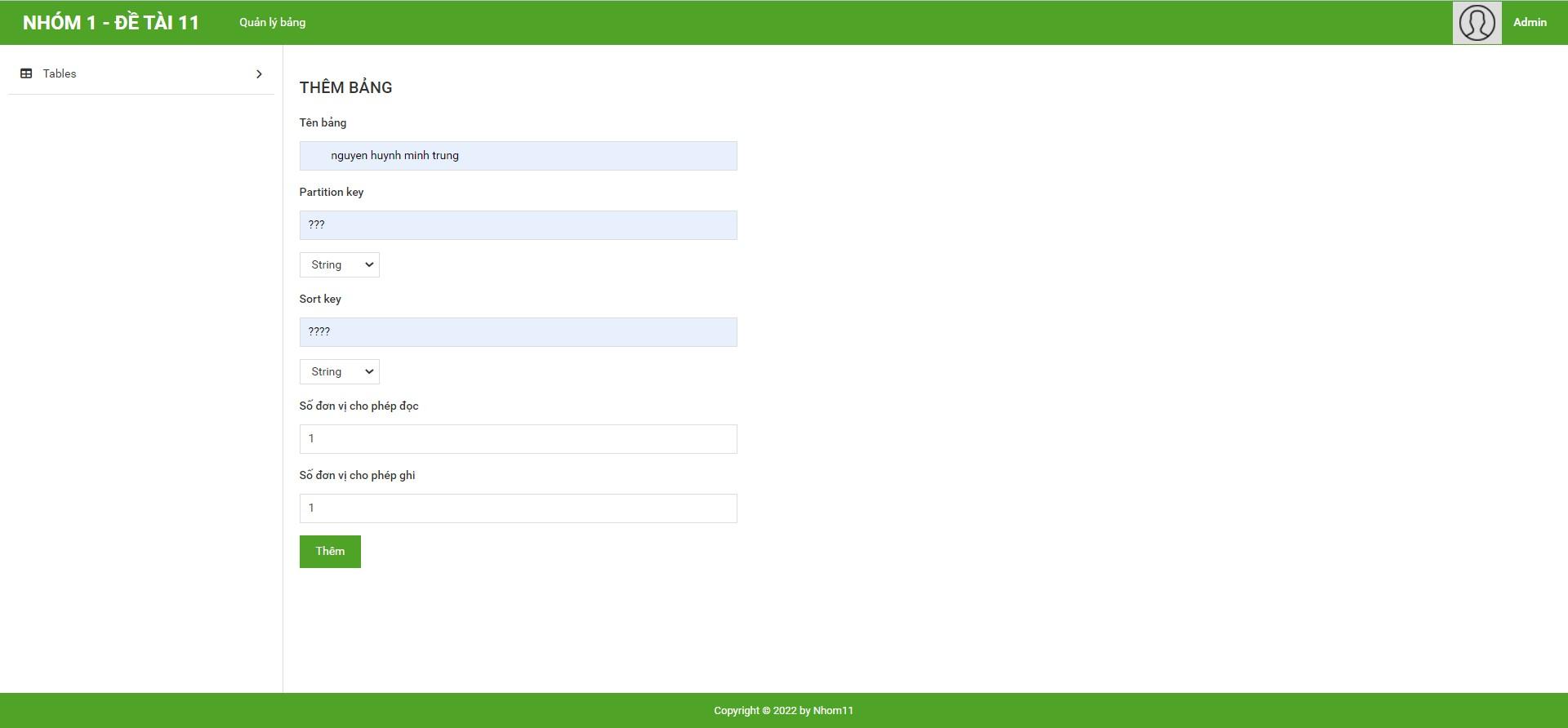
****

* Giao Diện:

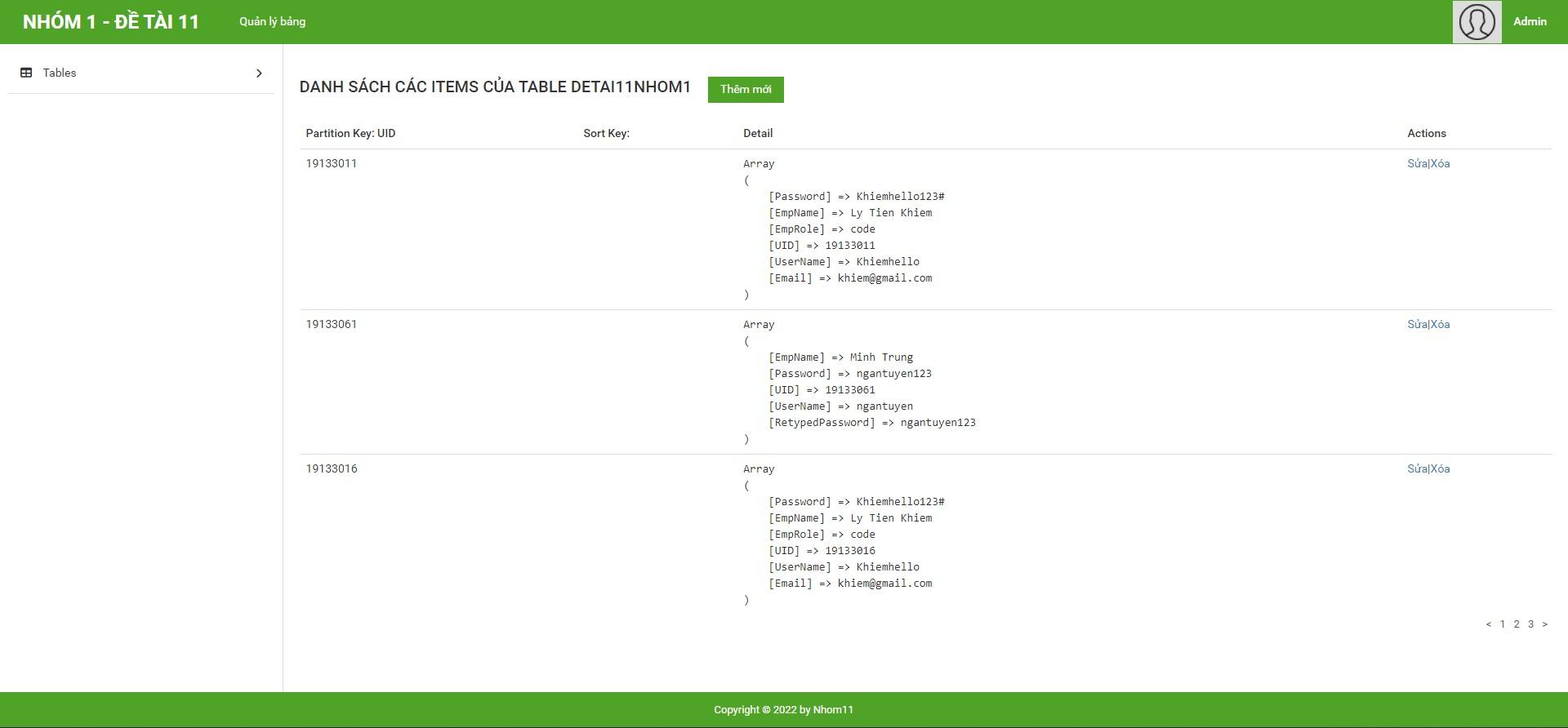
Giao diện hiển thị danh sách bảng



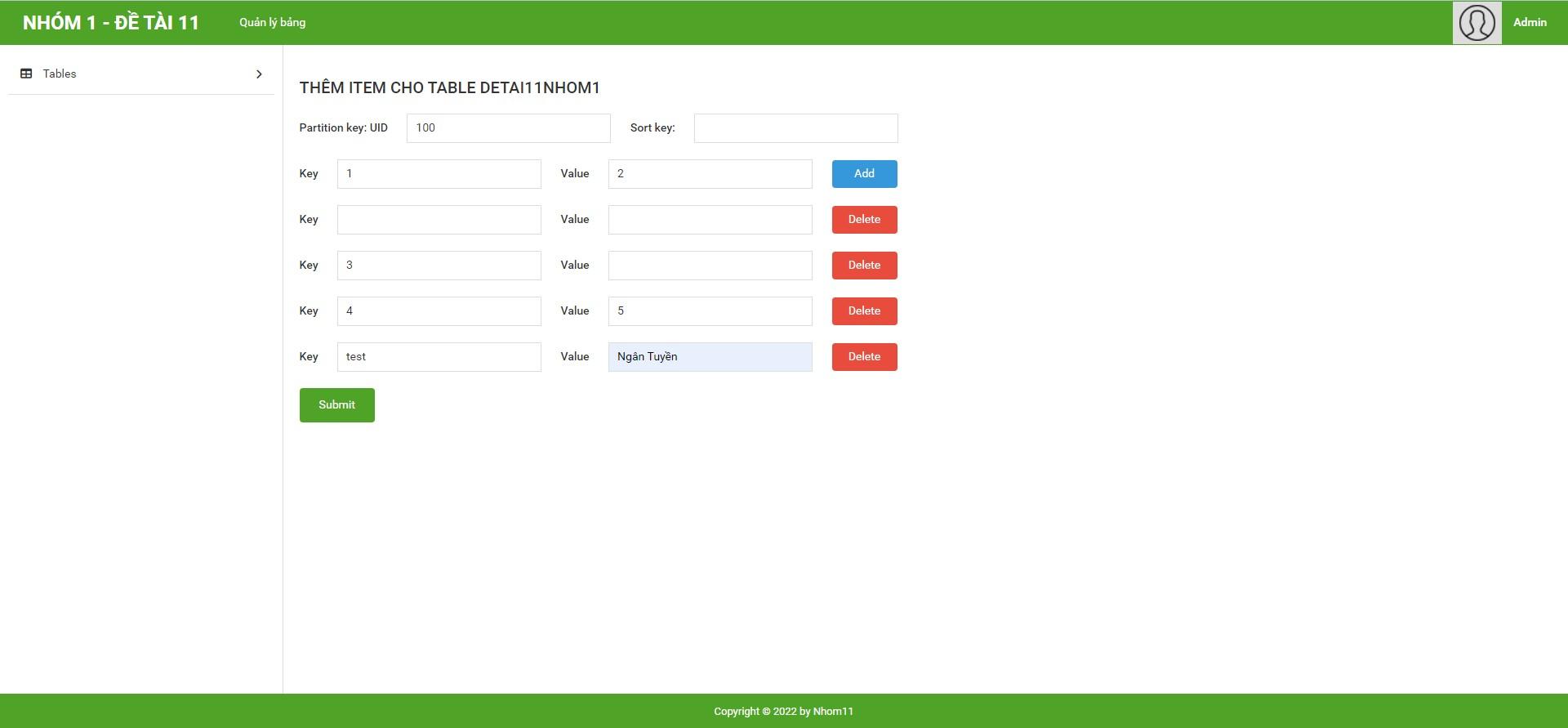
Giao diện thêm bảng



Giao diện hiển thị danh sách các Items trong bảng



Giao diện thêm/Sửa Item của bảng



## **2.2 Triển khai Project:**

### *2.2.1. Local:*

-Link video hướng dẫn: <https://youtu.be/L9aZ38QmrSo>

Bước 1: Chuẩn bị môi trường

Vào LearnLab--> start lab --> Aws Detail --> AWS CLI --> show copy bảng credentials

Vào thư mục C:\Users\tên máy\ .aws\credentials

Past AWS CLI vừa copy file credentials

Tải và cài đặt XAMPP và Git

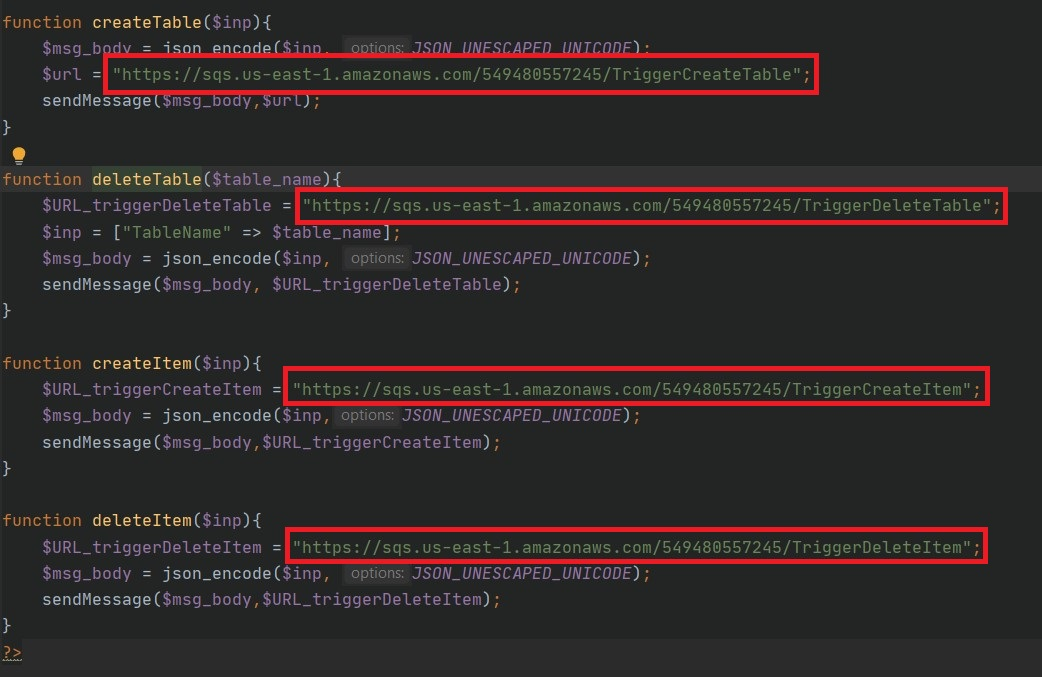
Bước 2: Cấu hình lại đồ án

git clone https://github.com/trungnguyenhuynhminh46/ProjectDetai11Nhom1v2 vào thư mục htdocs của XAMPP

**Tạo API bằng SQS:**

Vào AWS Tạo các SQS và Lamda theo file Lamda.txt

Chú ý Copy URL các SQS vừa tạo và past vào file lib/tables.php trong thư mục project để thay thế các URL của SQS

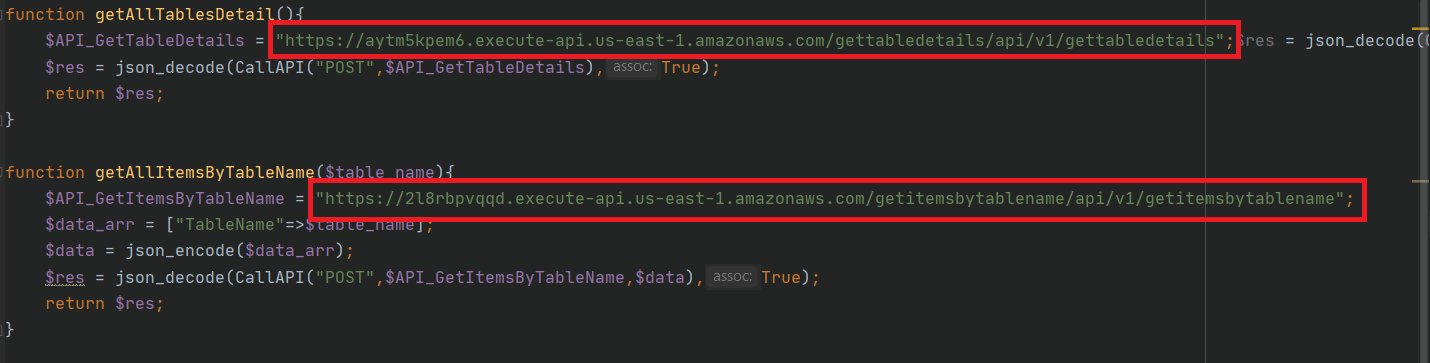


**Tạo API bằng API Gateway**

Vào AWS tạo các Lamda function theo API.txt

Làm theo video sau để tạo API cho 2 function vừa tạo: https://youtu.be/vKmmZ9Wp2-s

Sửa đường dẫn API tương tự SQS



Bước 3: Chạy

Mở XAMPP và start Apache

mở trình duyệt http://localhost:'port-apache'/ProjectDetai11Nhom1

### *2.2.2. EC2:*

* Link video hướng dẫn: <https://youtu.be/LqgAbx-NY0w>
* Các Bước thực hiện

Tạo EC2 với hệ điều hành window, mở port http và alltraffic

Tạo Elastic IP kết nối đến EC2

Connect EC2 bằng remote Desktop Connection

Trên window của EC2 vào Window Security --> Firewall & network protection --> Windows Defender Firewall Properties --> tag Public Profile --> Allow Inbound connections --> Apply--> ok

Dowload XAMPP và GIT

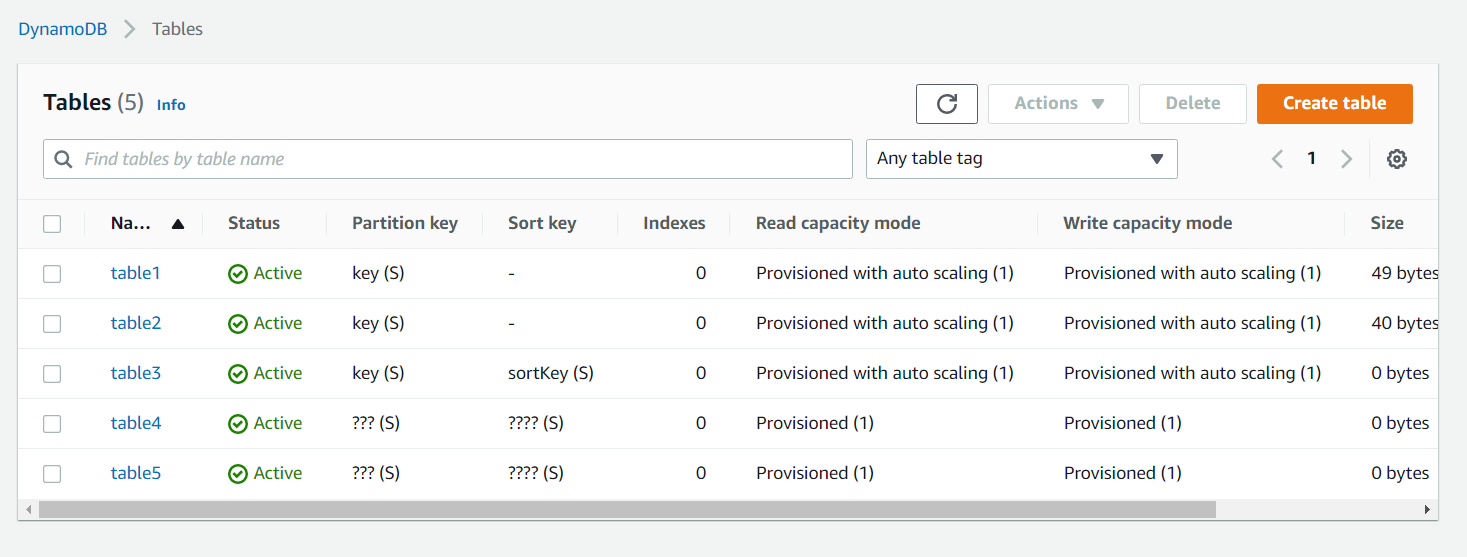
Mở git bash trong thư mục htdocs của XAMPP vừa tải về và git clone <https://github.com/trungnguyenhuynhminh46/ProjectDetai11Nhom1.git>

Copy .aws của local máy tính và past vào c/user/Administrator của máy window EC2

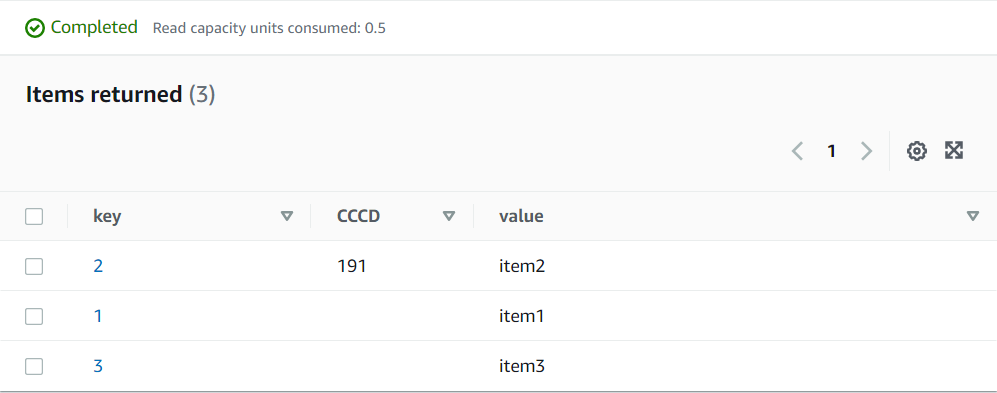
Mở XAMPP start apache và mở trình duyệt vào http://<ip máy EC2>/ProjectDetai11Nhom1 => Deploy thành công

## **2.3 Thiết kế DynamoDB minh họa cho việc tương tác:**

# Danh sách các bảng



Danh sách Items của một bảng (table1)



# **PHẦN KẾT LUẬN**

## **Kết quả đạt được**

Sau một thời gian nghiên cứu và thực hiện đề tài “***Xây dụng ứng dụng trên AWS cho phép tạo database và cung cấp API để thêm, sửa, xóa trên Database***”, nhóm chúng em đã đạt được những kết quả như sau:

### *1.1. Kiến thức tìm hiểu được*

Nắm bắt được các kiến thức cũng như những cách hoạt động của các dịch vụ Lamda Function, SQS, API, 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 các luồng Data được di chuyển, thiết kế và xây dựng ứng dụng để tương tác, thực hiện tích hợp API vào ứng dụng để tạo tương tác với Database.

### *1.2. Chương trình đã làm được*

Xây dựng hoàn chỉnh một ứng dụng tương tác với database (DynamoDB) bằng code PHP, Html, Css và các dịch vụ của AWS: SQS, Lamda Function (Python), API,DynamoDB với các chức năng cơ bản như:

* Thêm, xóa, đọc các table trong DynamoDB.
* Thêm, sửa, xóa, đọc các Item trong Table của DynamoDB.

## **Ưu điểm**

* Ứng dụng dễ tương tác, thân thiện với người sử dụng.
* luồng xử lý dữ liệu rõ ràng vì được xử lý thông qua các API
* Lưu trữ cũng như xử lý được lượng dữ liệu lớn nhanh chóng do cơ sở dữ liệu được quản lý bởi AWS.

## **Nhược điểm**

* Khó Setup.
* Các Function Lamda được theo một mẫu sẵn nên cần phải dành thời gian tìm hiểu.
* Tốn phí vì sử dụng dịch vụ của AWS và phụ thuộc phần lớn vào AWS trong vấn đề chạy code.

## **Hướng phát triển**

* Cải thiện giao diện