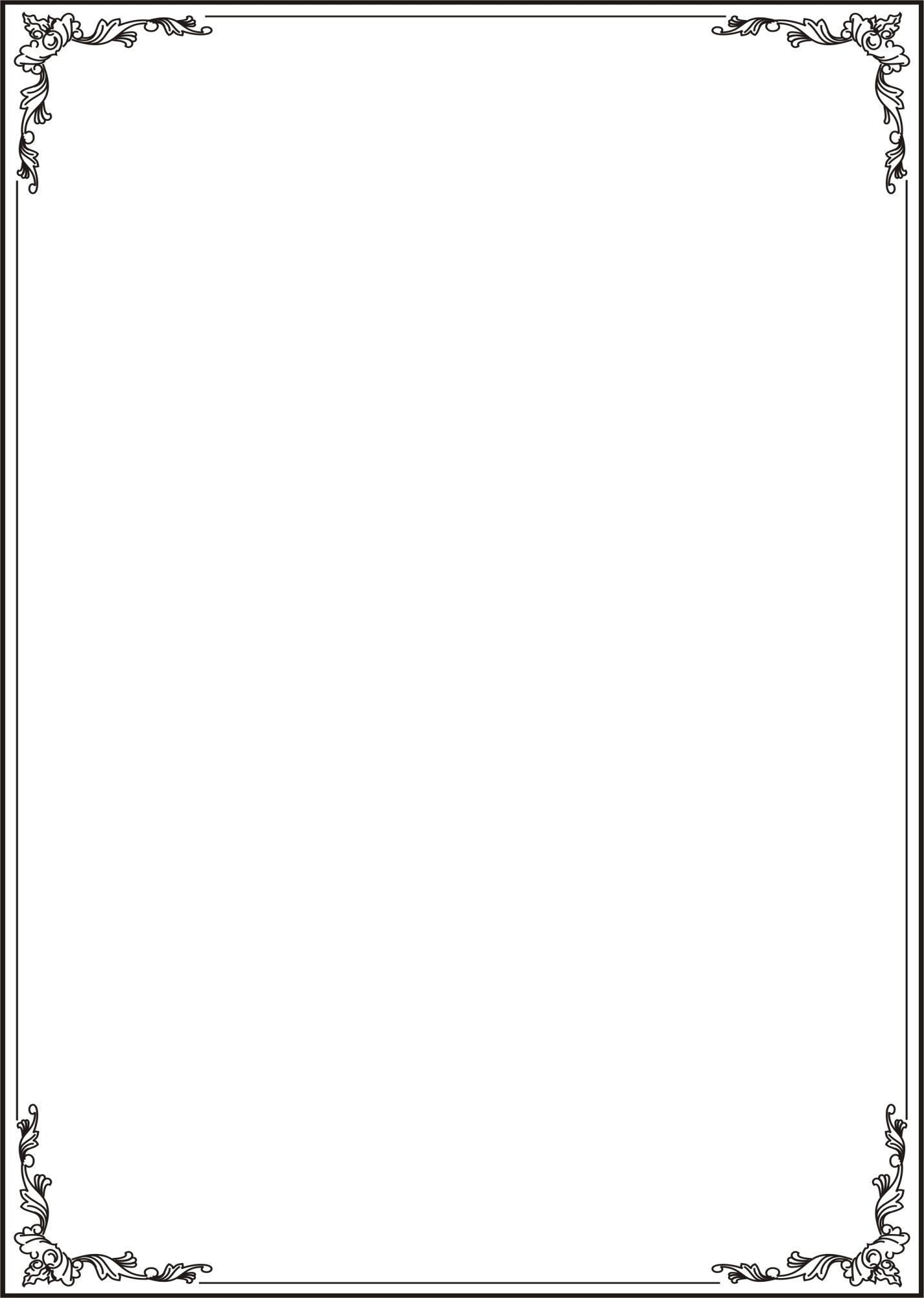
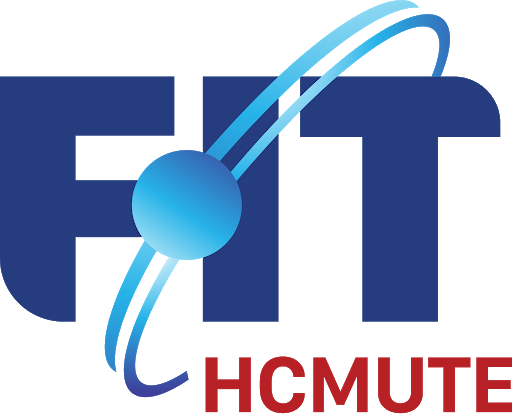
**.**

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

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN ĐIỆN TOÁN ĐÁM MÂY**

-----🙞🙜🕮🙞🙜-----

****

**ĐỀ TÀI:**

**XÂY DỰNG ỨNG DỤNG TRÊN AWS CHO PHÉP TẠO DATABASE VÀ CUNG CẤP API ĐỂ THÊM, XÓA, SỬA TRÊN DATABASE**

Khóa 2019-2022

**GVHD: TS. HUỲNH XUÂN PHỤNG**

**SVTH:**

**LÊ HẢI ĐĂNG 20110243**

**TRẦN MINH MẪN 20110301**

**TRẦN CÔNG MINH 20110677**

Khóa 2019 -2023

**KHÓA 2020-2024**

|  |  |
| --- | --- |
| **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: **Lê Hải Đăng** MSSV 1: **20110243**

Họ và tên Sinh viên 2: **Trần Minh Mẫn** MSSV 2: **20110301**

Họ và tên Sinh viên 2: **Trần Công Minh** MSSV 2: **20110677**

Ngành: **Công nghệ thông tin**

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**

Lê Hải Đăng 20110243

Trần Minh Mẫn 20110301

Trần Công Minh 20110677

# MỤC LỤC

[**LỜI CẢM ƠN**](#_heading=h.4icuj6ygolm0) **3**

[**MỤC LỤC**](#_heading=h.26in1rg) **4**

[**DANH MỤC CÁC HÌNH**](#_heading=h.1ksv4uv) **6**

[**PHẦN MỞ ĐẦU**](#_heading=h.n7wdlh6wt3ay) **8**

[**1. Tính cấp thiết của đề tài**](#_heading=h.1y810tw) **9**

[**2. Đối tượng nghiên cứu**](#_heading=h.pamn9ef1kb9) **9**

[**3. Phạm vi nghiên cứu**](#_heading=h.4i7ojhp) **9**

[**4. Kết quả dự kiến đạt được**](#_heading=h.2xcytpi) **9**

[**PHẦN NỘI DUNG**](#_heading=h.1ci93xb) **10**

[**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**](#_heading=h.147n2zr) **10**

[**1. Tổng quan về Lambda**](#_heading=h.3o7alnk) **10**

[**2. Các tính năng của Lambda**](#_heading=h.23ckvvd) **10**

[**3. Các dịch vụ liên quan đến Lambda**](#_heading=h.ihv636) **11**

[**CHƯƠNG 2: THỰC HIỆN**](#_heading=h.hhve4vulaixm) **14**

[**1. Cài đặt**](#_heading=h.41mghml) **14**

[1.1. Tạo Table trên DynamoDB](#_heading=h.2grqrue) 14

[1.2. Cài đặt Function Lamda](#_heading=h.vx1227) 15

[1.2.1. User](#_heading=h.xnrbr8kaii0r) 15

[1.2.2. Area](#_heading=h.wzgpb7tz1m10) 21

[1.2.3. Employee](#_heading=h.do6i0qh1jy0v) 26

[1.2.4. Login](#_heading=h.x86tmnkm5c1x) 33

[**2. Giao diện**](#_heading=h.rt8514uf7nl0) **33**

[2.1. Trang đăng nhập](#_heading=h.7craxcvikvjv) 33

[2.2. Trang Admin](#_heading=h.cg32d0kspjzy) 35

[2.3. Trang người quản lý](#_heading=h.1p90p4gi3jz0) 43

[**3. Các hàm gọi API tương tác với DynamoDB**](#_heading=h.3fwokq0) **49**

[3.1. Admin](#_heading=h.twt5141yw7dx) 50

[3.1.1. Người dùng](#_heading=h.tn5ej19yo96y) 50

[3.1.2. Khu vực](#_heading=h.lw6dtvg1r97b) 52

[3.2. Người quản lý](#_heading=h.uzlb5tk0kl7y) 54

[**4. Triển khai Project**](#_heading=h.1v1yuxt) **57**

[4.1. Local](#_heading=h.sv6wqastruwl) 57

[4.2. AWS](#_heading=h.cv7tekm1fxnk) 57

[**PHẦN KẾT LUẬN**](#_heading=h.3tbugp1) **57**

[**1. Kết quả đạt được**](#_heading=h.28h4qwu) **57**

[1.1. Kiến thức tìm hiểu được](#_heading=h.nmf14n) 57

[1.2. Chương trình đã làm được](#_heading=h.37m2jsg) 57

[**2. Hướng phát triển**](#_heading=h.46r0co2) **58**

# DANH MỤC CÁC HÌNH

Hình 1.1.a : Danh sách các table trong Dynamodb

Hình 1.2.1.a : Thêm user

Hình 1.2.1.b: Chỉnh sửa thông tin user

Hình 1.2.1.c: Xóa user

Hình 1.2.1.d: Xóa vĩnh viễn user

Hình 1.2.1.e: Lấy tất cả user

Hình 1.2.1.f: Lấy user theo username

Hình 1.2.2.a: Thêm khu vực

Hình 1.2.2.b: Chỉnh sửa thông tin khu vực

Hình 1.2.2.c: Xóa khu vực

Hình 1.2.2.d: Xóa vĩnh viễn khu vực

Hình 1.2.2.e: Lấy tất cả khu vực

Hình 1.2.2.f: Lấy khu vực theo id của khu vực

Hình 1.2.3.a: Thêm nhân viên

Hình 1.2.3.b: Chỉnh sửa thông tin nhân viên

Hình 1.2.3.c: Xóa nhân viên

Hình 1.2.3.d: Xóa vĩnh viễn nhân viên

Hình 1.2.3.e: Lấy tất cả nhân viên

Hình 1.2.3.f: Lấy nhân viên theo id của nhân viên

Hình1.2.4.a: Login

Hình 2.1.a: Trang đăng nhập

Hình 2.1.b: Trang chọn vai trò người dùng

Hình 2.1.c: Trang đăng nhập của Admin

Hình 2.1.d: Trang đăng nhập của người quản lý

Hình 2.2.a: Trang quản lý user

Hình 2.2.b: Trang thêm user mới

Hình 2.2.c: Trang chỉnh sửa thông tin user

Hình 2.2.d: Thông báo xóa user đã chọn

Hình 2.2.e: Thông báo xóa nhiều user

Hình 2.2.f: Thông báo xóa vĩnh viễn user

Hình 2.2.g: Thông báo khôi phục user

Hình 2.2.h: Trang quản lý khu vực

Hình 2.2.i: Trang thêm khu vực mới

Hình 2.2.j: Trang chỉnh sửa thông tin khu vực

Hình 2.2.k: Thông báo xóa khu vực đã chọn

Hình 2.2.l: Thông báo xóa nhiều khu vực

Hình 2.2.m: Thông báo xóa vĩnh viễn khu vực

Hình 2.2.n: Thông báo khôi phục khu vực

Hình 2.3.a: Trang quản lý nhân viên

Hình 2.3.b: Trang thêm nhân viên mới

Hình 2.3.c: Trang chỉnh sửa thông tin nhân viên

Hình 2.3.d: Thông báo xóa nhân viên đã chọn

Hình 2.3.e: Thông báo xóa nhiều nhân viên

Hình 2.3.f: Thông báo xóa vĩnh viễn nhân viên

Hình 2.3.g: Thông báo khôi phục nhân viên

Hình 2.3.h: Chỉnh sửa thông tin cá nhân

Hình 3.a: Các biến môi trường lưu các url

Hình 3.a: Login

Hình 3.1.1.a: Lấy tất cả người dùng và tất cả khu vực

Hình 3.1.1.b: Lấy người dùng theo username

Hình 3.1.1.c: Thêm người dùng

Hình 3.1.1.d: Chỉnh sửa thông tin người dùng

Hình 3.1.1.e: Xóa người dùng

Hình 3.1.1.f: Xóa vĩnh viễn người dùng

Hình 3.1.2.a: Lấy tất cả người dùng và tất cả khu vực

Hình 3.1.2.b: Lấy khu vực theo id

Hình 3.1.2.c: Thêm khu vực

Hình 3.1.2.d: Chỉnh sửa thông tin khu vực

Hình 3.1.2.e: Xóa khu vực

Hình 3.1.2.f: Xóa vĩnh viễn khu vực

Hình 3.2.a: Lấy người dùng theo username và khu vực theo id

Hình 3.2.b: Lấy tất cả nhân viên

Hình 3.2.c: Lấy người dùng theo username

Hình 3.2.d: Lấy nhân viên theo id

Hình 3.2.e: Thêm nhân viên

Hình 3.2.f: Chỉnh sửa thông tin nhân viên

Hình 3.2.g: Xóa nhân viên

Hình 3.2.i: Xóa vĩnh viễn nhân viên

Hình 3.2.j: Chỉnh sửa thông tin các nhân người dùng

# 

# 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, sửa, xóa các item trong DynamoDB

# 

# PHẦN NỘI DUNG

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

## Tổng quan về Lambda

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

AWS Lambda 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 Lambda cho phép lập trình viên dễ dàng kết hợp với các dịch vụ khác như S3, DynamoDB,.... vì vậy có thể dễ dàng kích hoạt các Lambda 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 Lambda cho phép lập trình viên có thể tạo các backend cho ứng dụng bằng cách sử dụng các API Lambda 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.

Bên cạnh đó còn có tính năng Lambda Function URL cho phép thêm điểm cuối HTTPS vào bất kỳ hàm Lambda nào và tùy chọn định cấu hình. Dễ hiểu hơn thì trước đây chúng ta sử dụng mô hình rất phổ biến trước đó là chúng ta có API Gateway sau đó đên Lambda rồi từ Lambda này chúng ta kế nối đến các dịch vụ khác của AWS hay database (DynamoDb, Aurora,…) nhưng thay vì như vậy chúng ta xây dựng một mô hình có Lambda và Lambda này được truy xuất thẳng từ bên ngoài không phải thông qua API Gateway.

* Sử dụng code riêng

AWS Lambda 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 Lam Dam, 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 HTTP và kích hoạt đến các Lambda 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 petabyte dữ liệu, xử lý hàng ngàn request credit 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 để tính tiền các dịch vụ hoặc các tài nguyên đã sử dụng trên AWS.

**Tại sao nên sử 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.

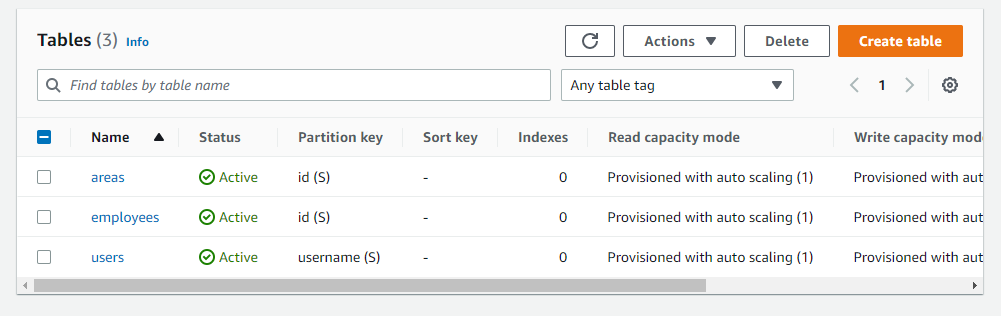
Lưu trữ các dữ liệu không liên tục: Do đặc tính là một cơ sở dữ liệu không quan hệ mà DynamoDB có đặc tính rất phù hợp với những đối tượng không có cấu trúc cố định và thay đổi theo thời gian.

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

## 1. Cài đặt

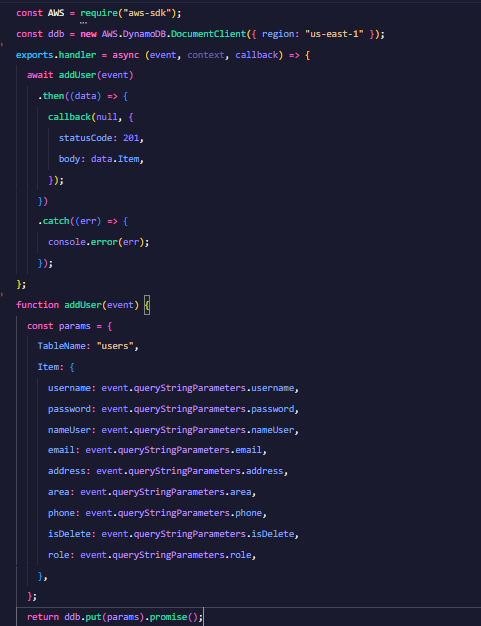
### 1.1. Tạo Table trên DynamoDB



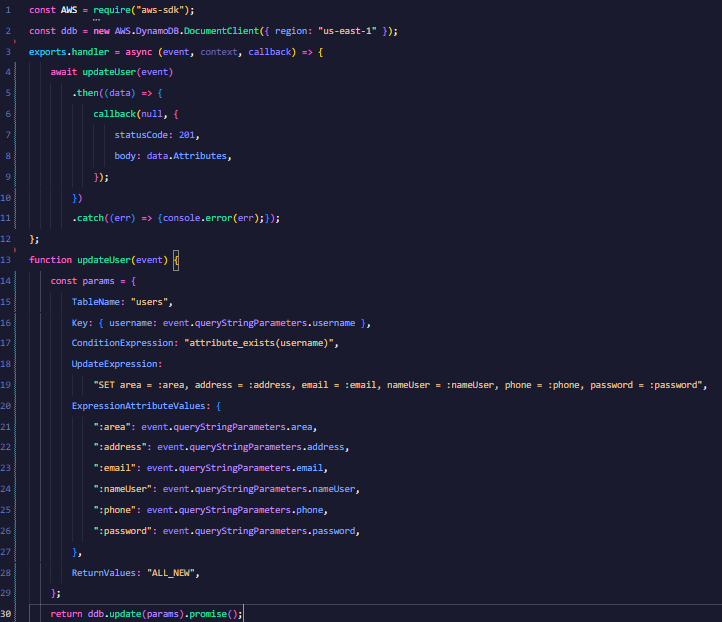
Hình 1.1.a : Danh sách các table trong Dynamodb

### 1.2. Cài đặt Function Lamda

#### 1.2.1. User



Hình 1.2.1.a : Thêm user



Hình 1.2.1.b: Chỉnh sửa thông tin user



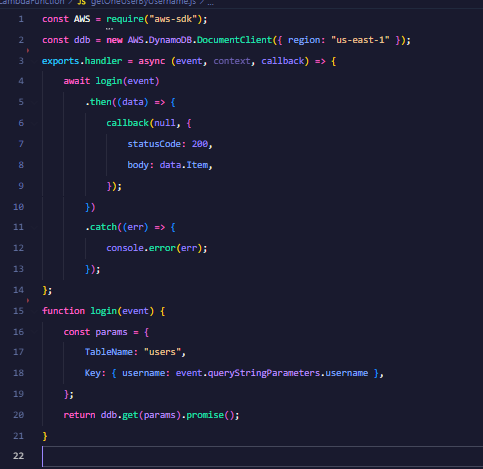
Hình 1.2.1.c: Xóa user



Hình 1.2.1.d: Xóa vĩnh viễn user

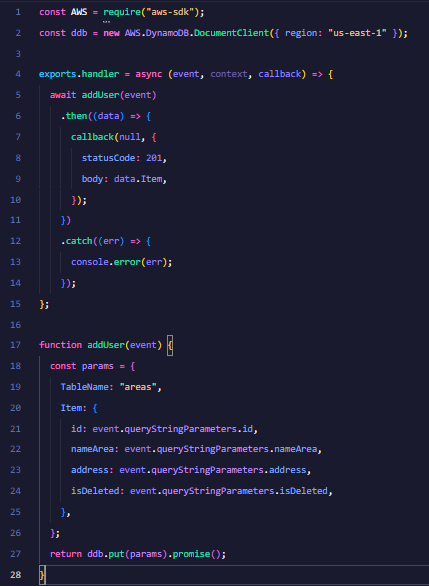


Hình 1.2.1.e: Lấy tất cả user

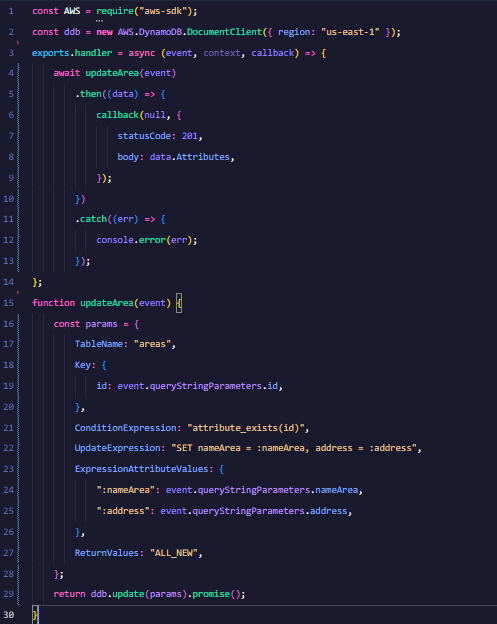


Hình 1.2.1.f: Lấy user theo username

#### 1.2.2. Area



Hình 1.2.2.a: Thêm khu vực



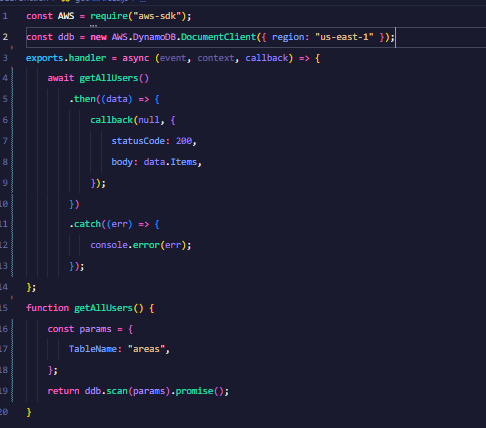
Hình 1.2.2.b: Chỉnh sửa thông tin khu vực



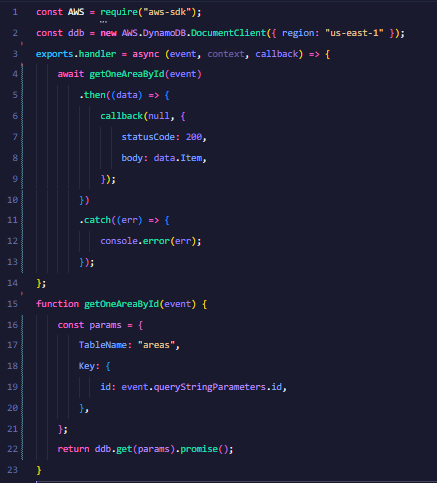
Hình 1.2.2.c: Xóa khu vực



Hình 1.2.2.d: Xóa vĩnh viễn khu vực



Hình 1.2.2.e: Lấy tất cả khu vực

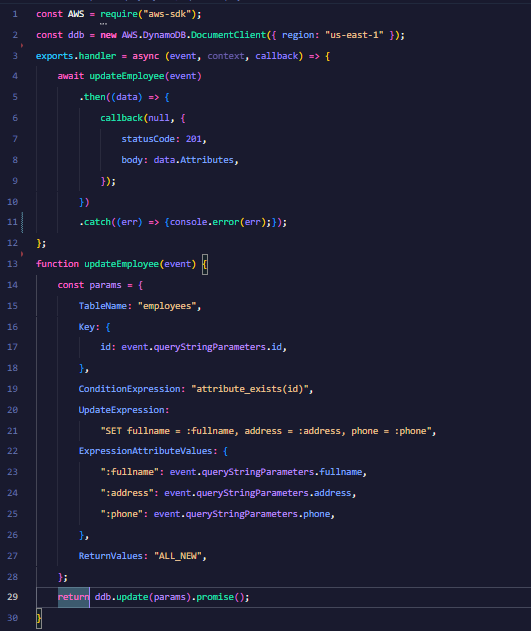


Hình 1.2.2.f: Lấy khu vực theo id của khu vực

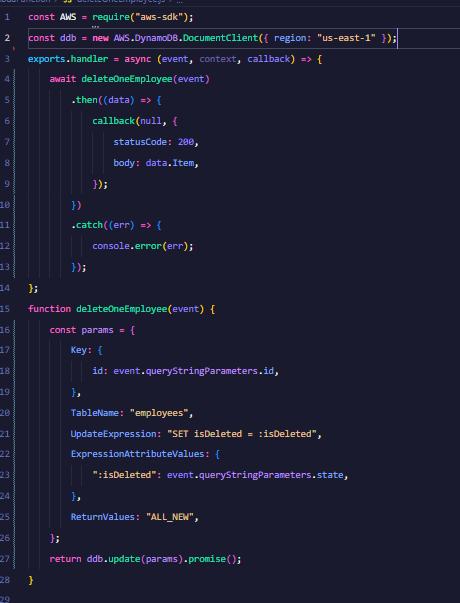
#### 1.2.3. Employee



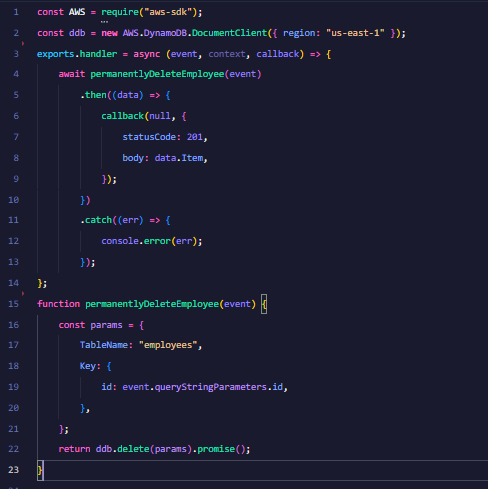
Hình 1.2.3.a: Thêm nhân viên



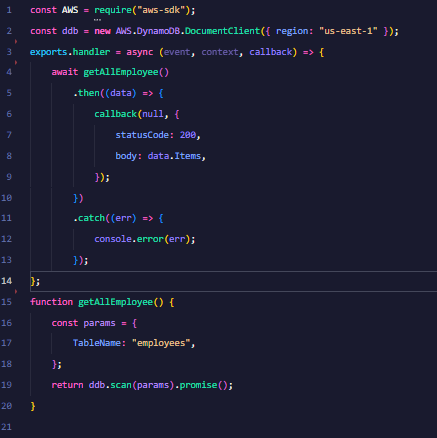
Hình 1.2.3.b: Chỉnh sửa thông tin nhân viên



Hình 1.2.3.c: Xóa nhân viên



Hình 1.2.3.d: Xóa vĩnh viễn nhân viên



Hình 1.2.3.e: Lấy tất cả nhân viên



Hình 1.2.3.f: Lấy nhân viên theo id của nhân viên

#### 1.2.4. Login



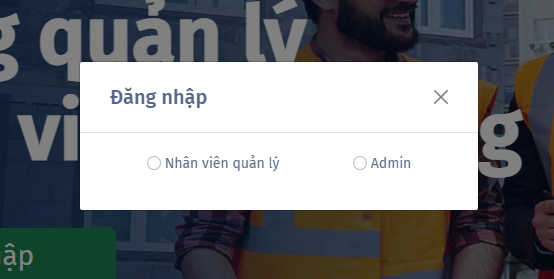
Hình1.2.4.a: Login

## 2. Giao diện

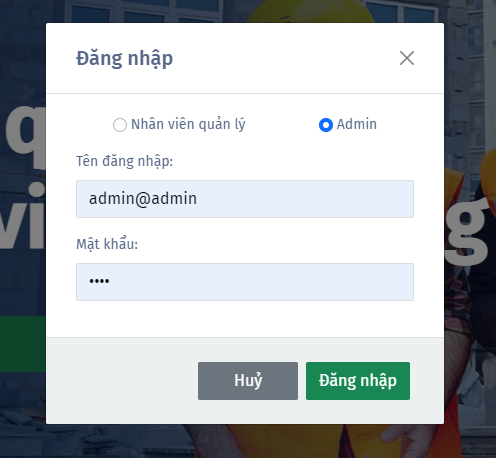
### 2.1. Trang đăng nhập



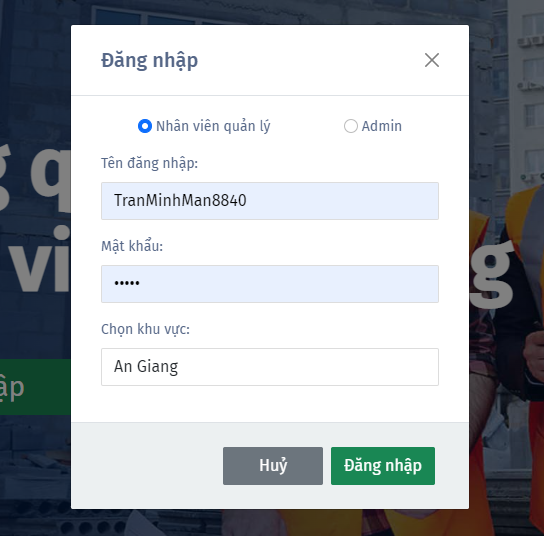
Hình 2.1.a: Trang đăng nhập



Hình 2.1.b: Trang chọn vai trò người dùng

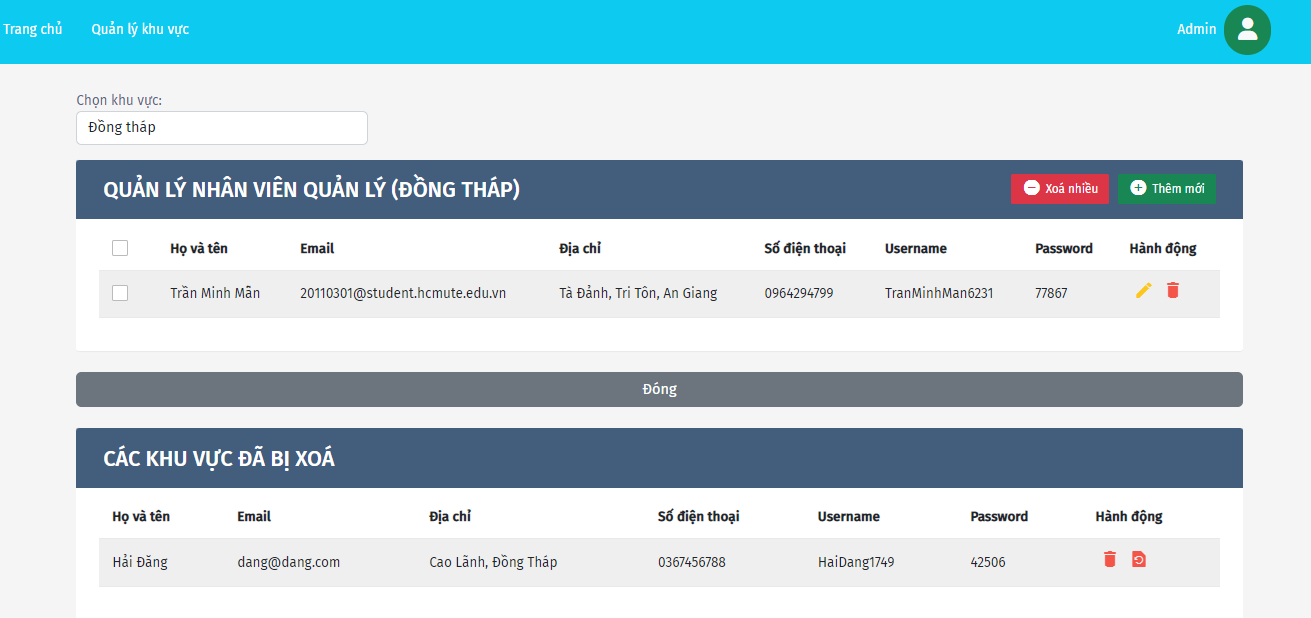


Hình 2.1.c: Trang đăng nhập của Admin

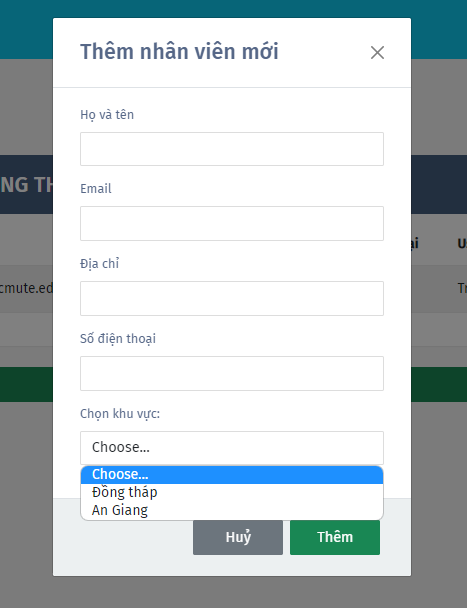


Hình 2.1.d: Trang đăng nhập của người quản lý

### 2.2. Trang Admin



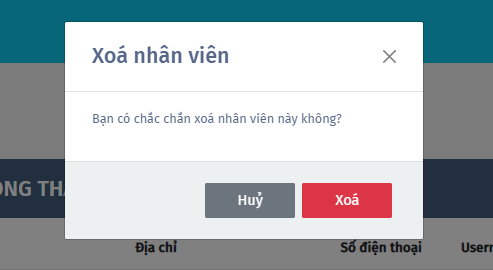
Hình 2.2.a: Trang quản lý user



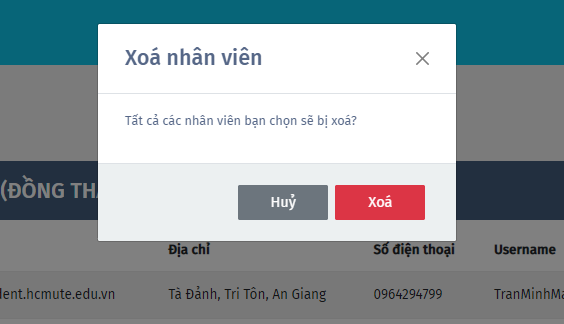
Hình 2.2.b: Trang thêm user mới



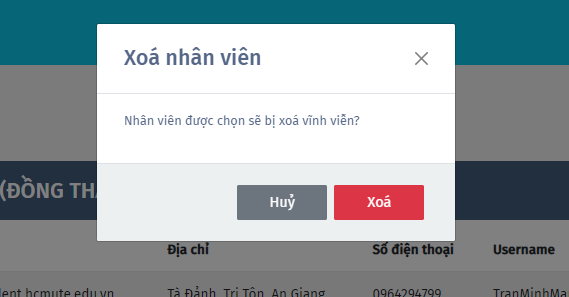
Hình 2.2.c: Trang chỉnh sửa thông tin user



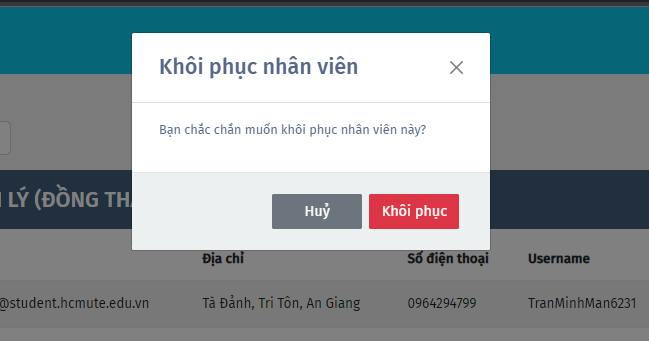
Hình 2.2.d: Thông báo xóa user đã chọn



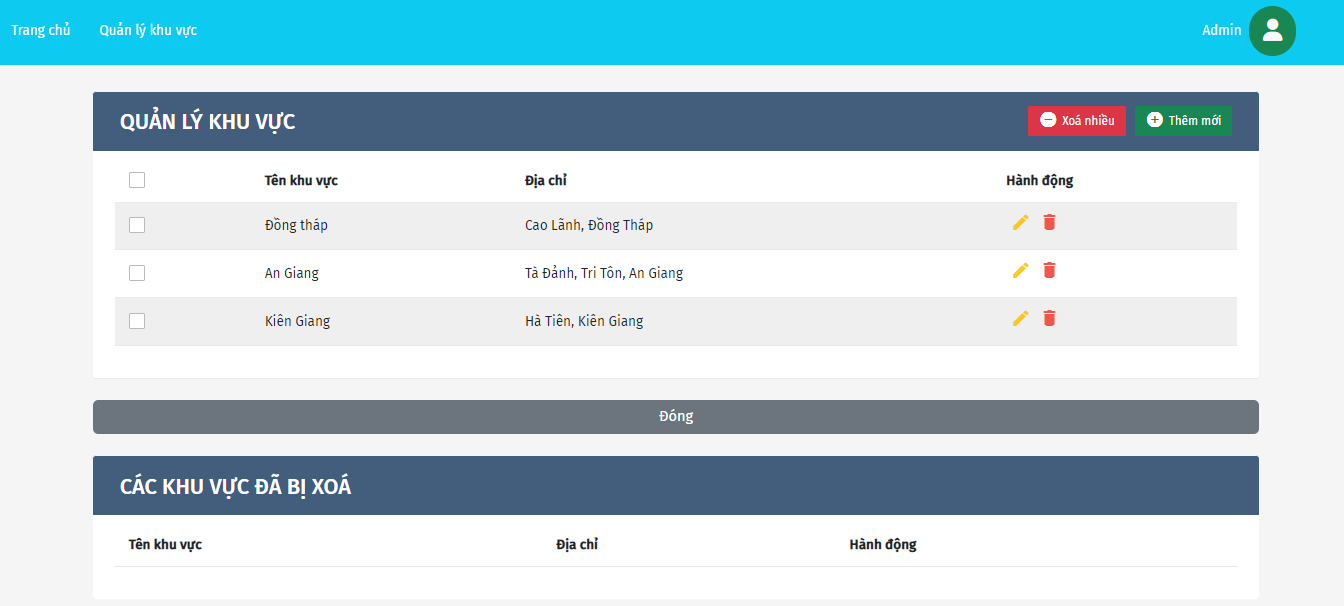
Hình 2.2.e: Thông báo xóa nhiều user



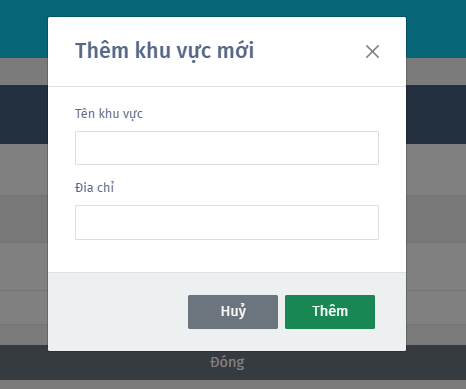
Hình 2.2.f: Thông báo xóa vĩnh viễn user



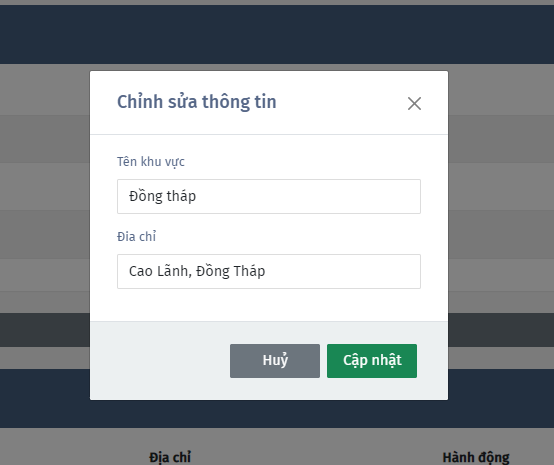
Hình 2.2.g: Thông báo khôi phục user



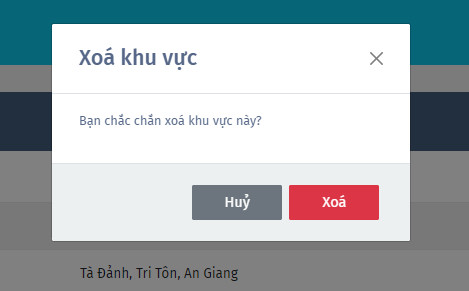
Hình 2.2.h: Trang quản lý khu vực



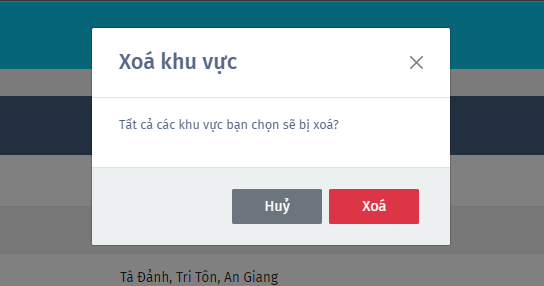
Hình 2.2.i: Trang thêm khu vực mới



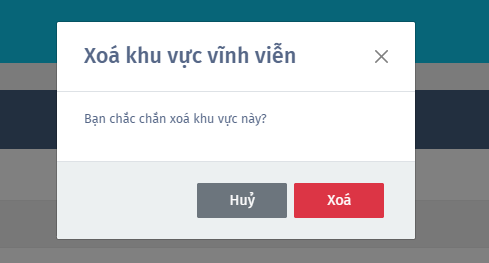
Hình 2.2.j: Trang chỉnh sửa thông tin khu vực



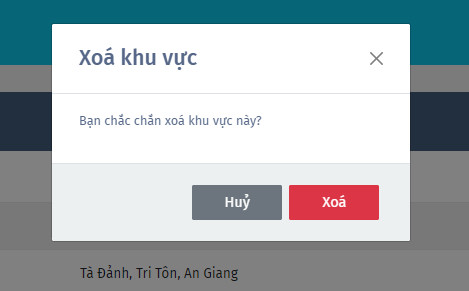
Hình 2.2.k: Thông báo xóa khu vực đã chọn



Hình 2.2.l: Thông báo xóa nhiều khu vực



Hình 2.2.m: Thông báo xóa vĩnh viễn khu vực

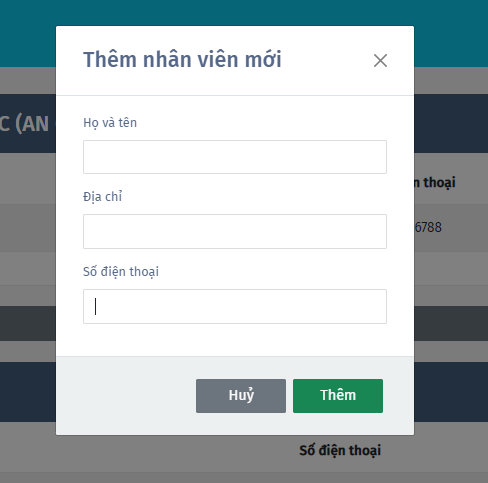


Hình 2.2.n: Thông báo khôi phục khu vực

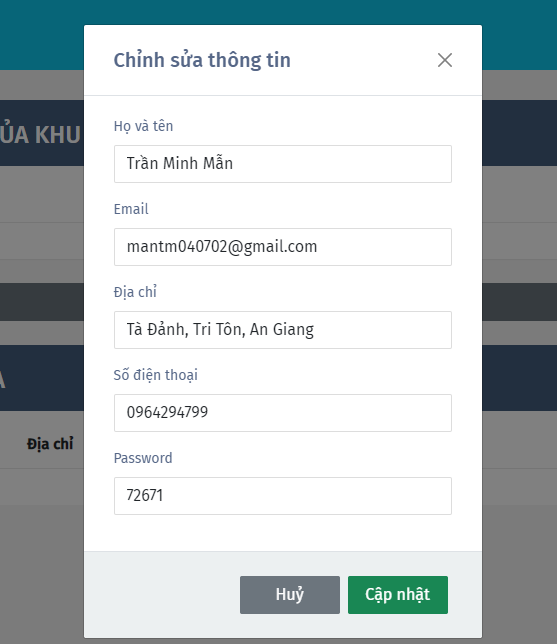
### 2.3. Trang người quản lý



Hình 2.3.a: Trang quản lý nhân viên



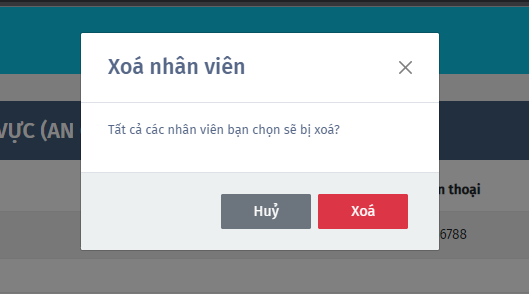
Hình 2.3.b: Trang thêm nhân viên mới



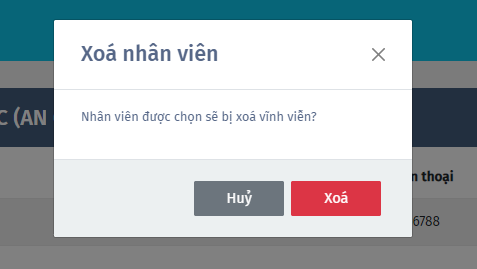
Hình 2.3.c: Trang chỉnh sửa thông tin nhân viên



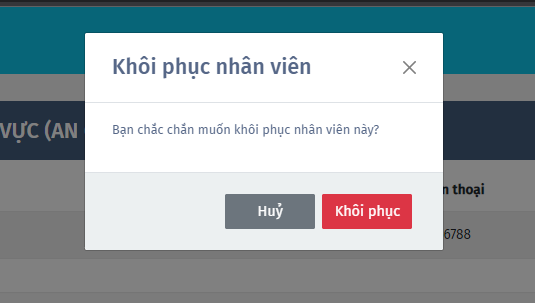
Hình 2.3.d: Thông báo xóa nhân viên đã chọn



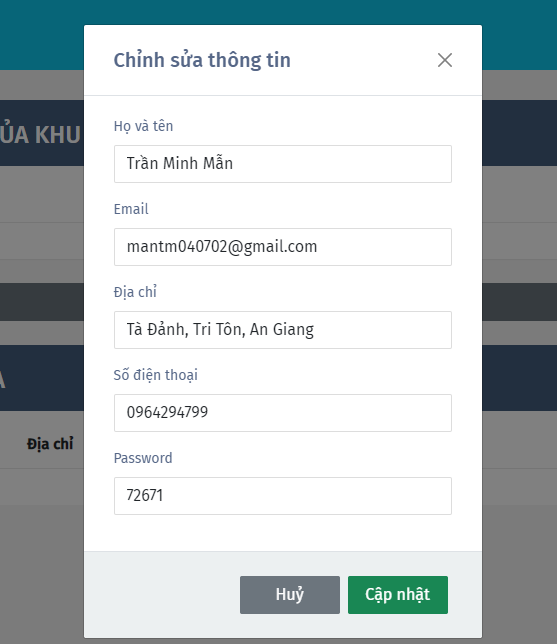
Hình 2.3.e: Thông báo xóa nhiều nhân viên



Hình 2.3.f: Thông báo xóa vĩnh viễn nhân viên

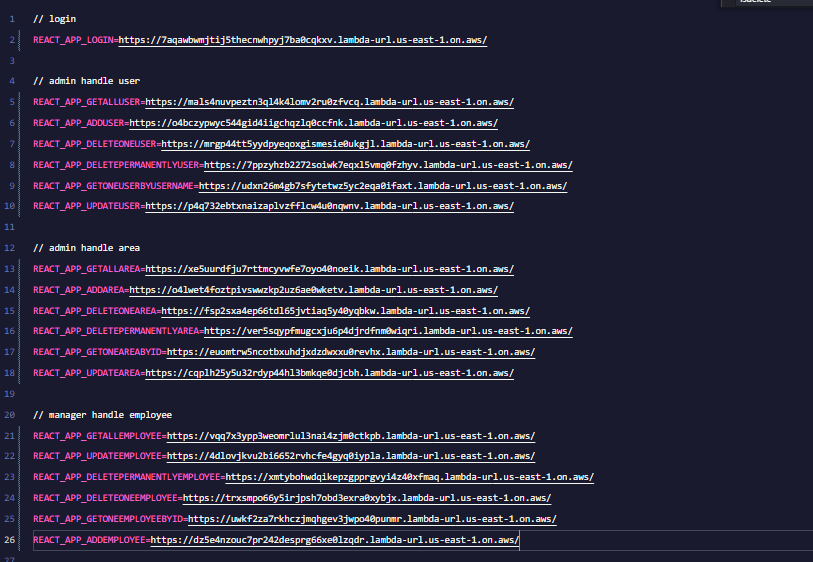


Hình 2.3.g: Thông báo khôi phục nhân viên

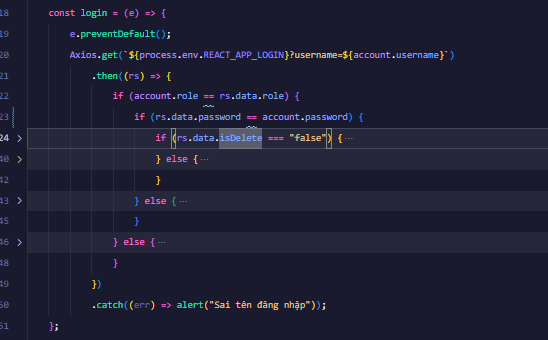


Hình 2.3.h: Chỉnh sửa thông tin cá nhân

## 3. Các hàm gọi API tương tác với DynamoDB



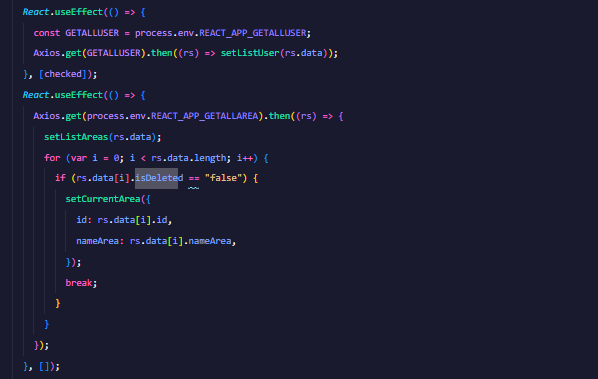
Hình 3.a: Các biến môi trường lưu các url



Hình 3.a: Login

### 3.1. Admin

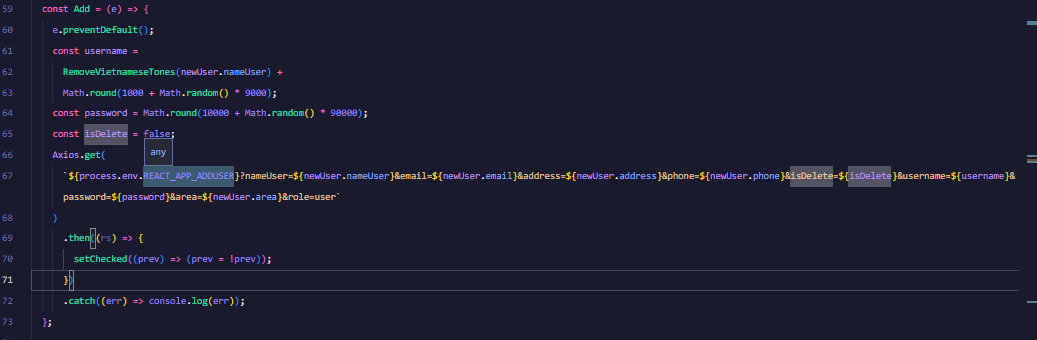
#### 3.1.1. Người dùng



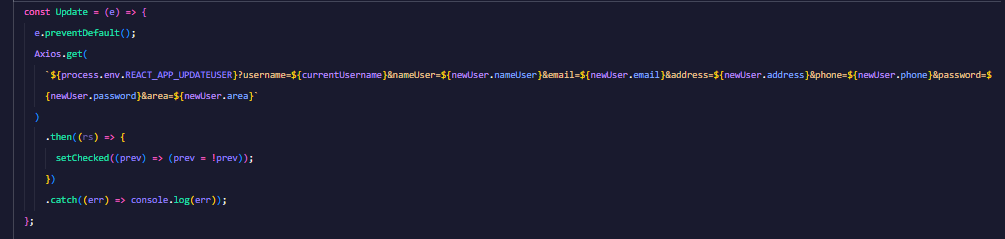
Hình 3.1.1.a: Lấy tất cả người dùng và tất cả khu vực



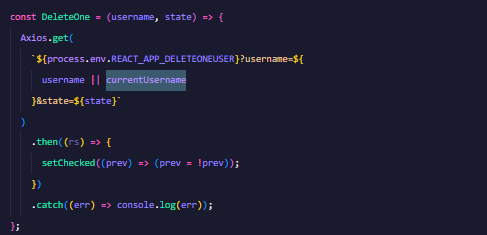
Hình 3.1.1.b: Lấy người dùng theo username



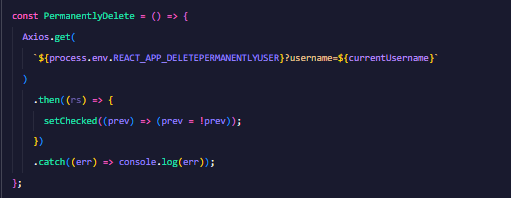
Hình 3.1.1.c: Thêm người dùng



Hình 3.1.1.d: Chỉnh sửa thông tin người dùng

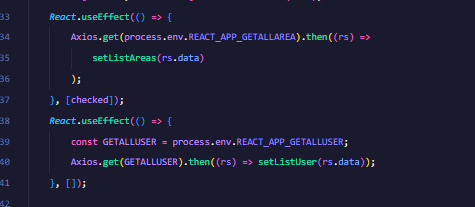


Hình 3.1.1.e: Xóa người dùng

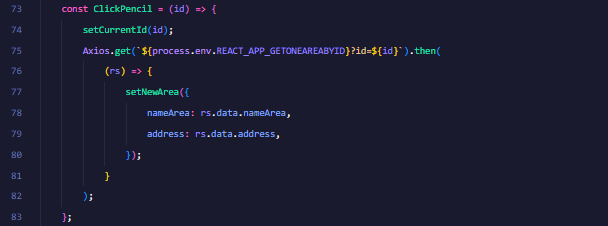


Hình 3.1.1.f: Xóa vĩnh viễn người dùng

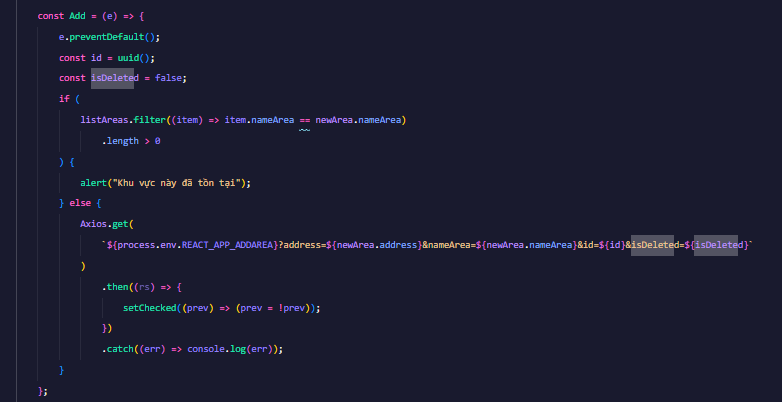
#### 3.1.2. Khu vực



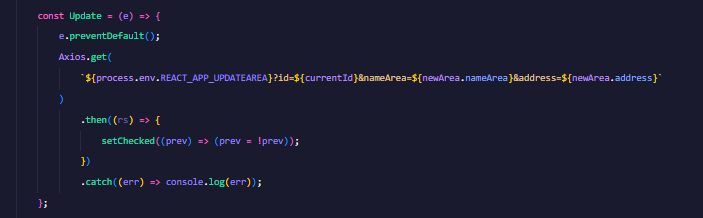
Hình 3.1.2.a: Lấy tất cả người dùng và tất cả khu vực



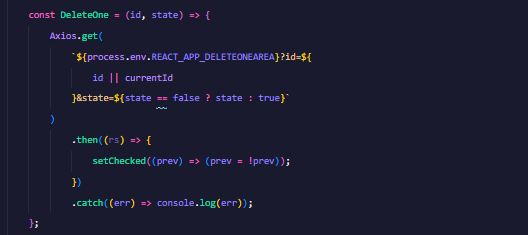
Hình 3.1.2.b: Lấy khu vực theo id



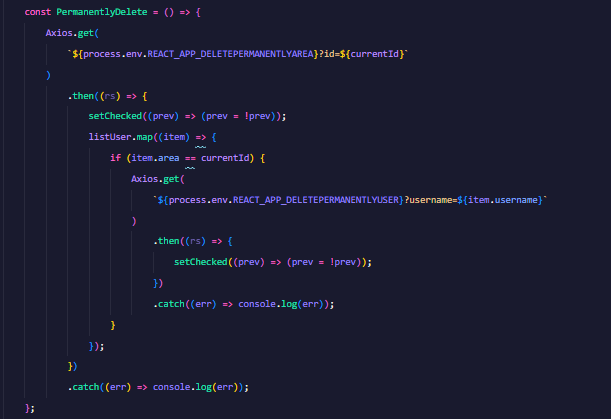
Hình 3.1.2.c: Thêm khu vực



Hình 3.1.2.d: Chỉnh sửa thông tin khu vực

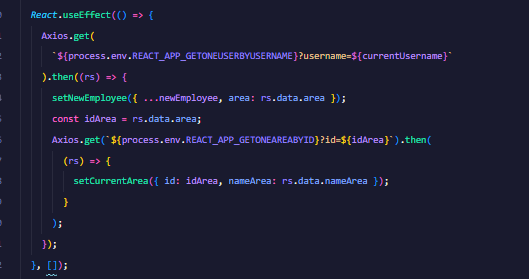


Hình 3.1.2.e: Xóa khu vực

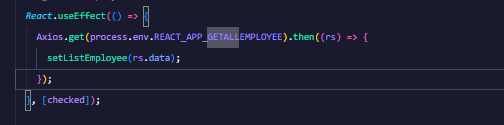


Hình 3.1.2.f: Xóa vĩnh viễn khu vực

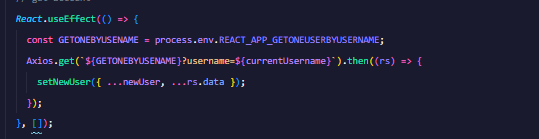
### 3.2. Người quản lý



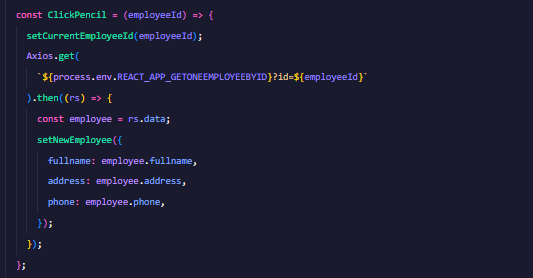
Hình 3.2.a: Lấy người dùng theo username và khu vực theo id



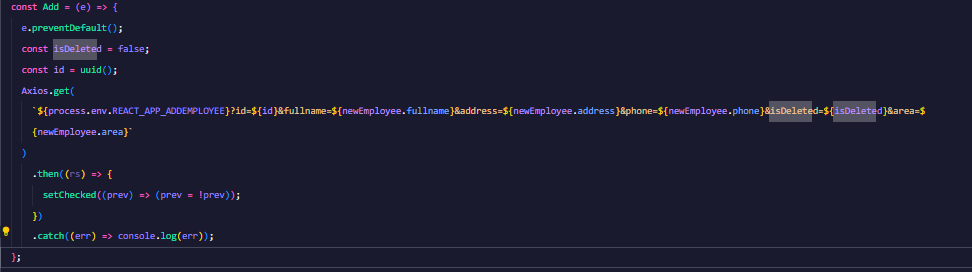
Hình 3.2.b: Lấy tất cả nhân viên



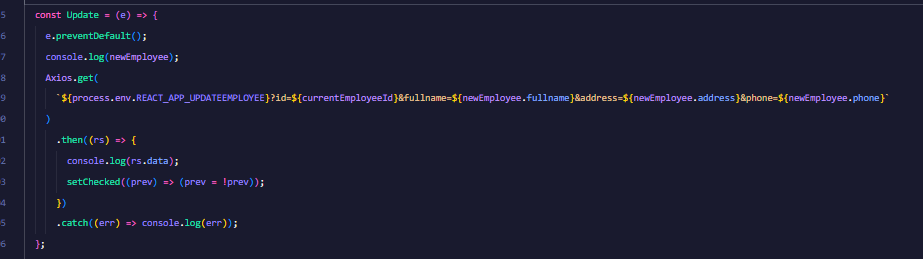
Hình 3.2.c: Lấy người dùng theo username



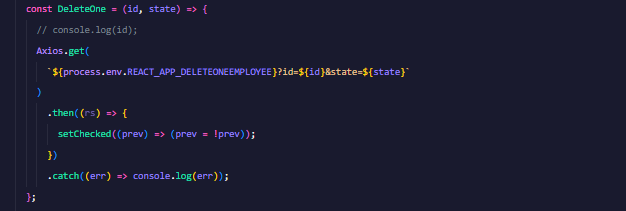
Hình 3.2.d: Lấy nhân viên theo id



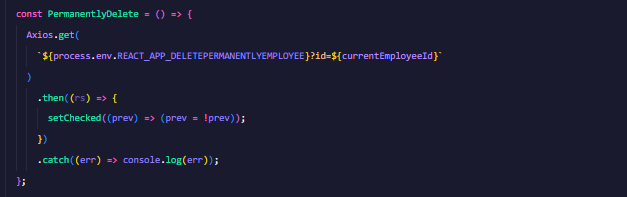
Hình 3.2.e: Thêm nhân viên



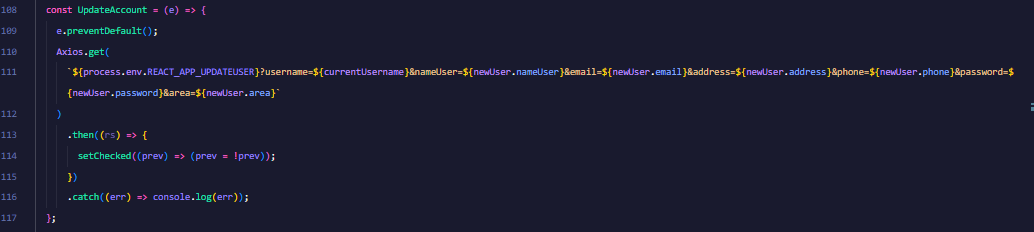
Hình 3.2.f: Chỉnh sửa thông tin nhân viên



Hình 3.2.g: Xóa nhân viên



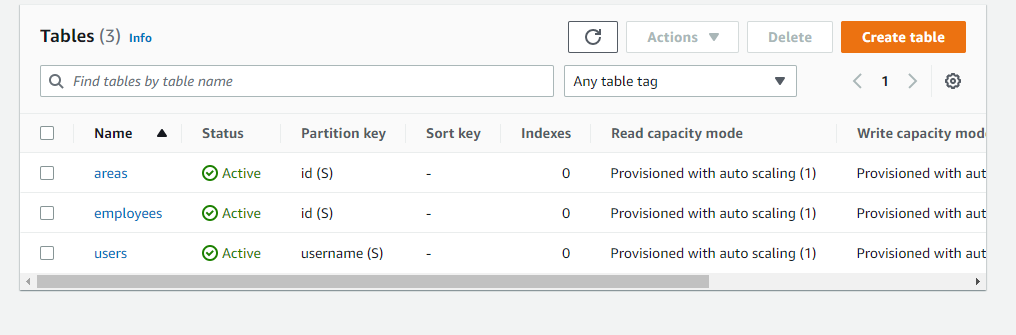
Hình 3.2.i: Xóa vĩnh viễn nhân viên



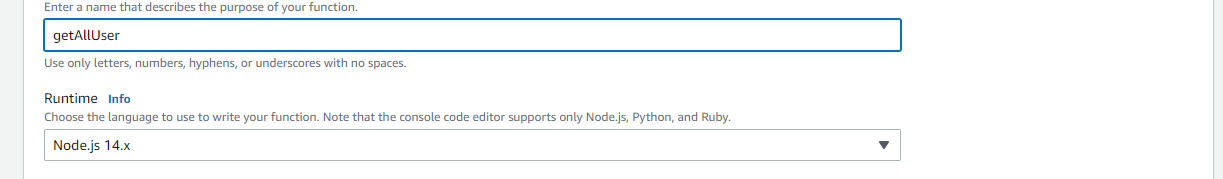
Hình 3.2.j: Chỉnh sửa thông tin các nhân người dùng

### 4. Triển khai Project

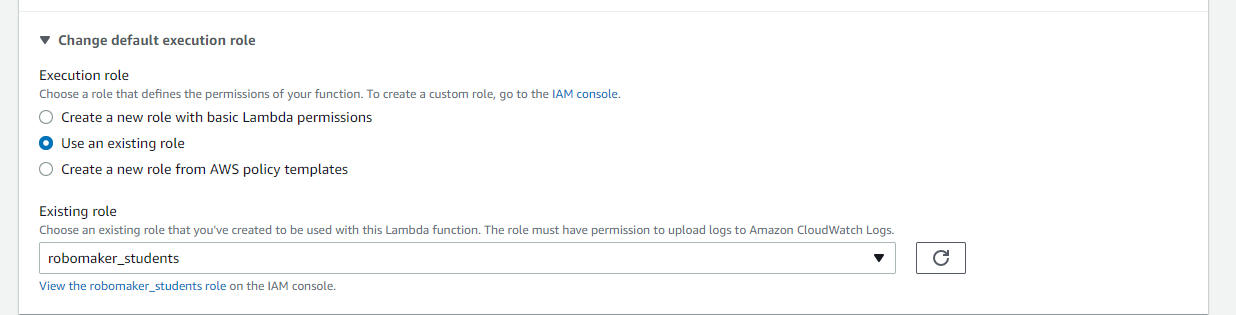
### Tạo cơ sở dữ liệu với DynamoDB



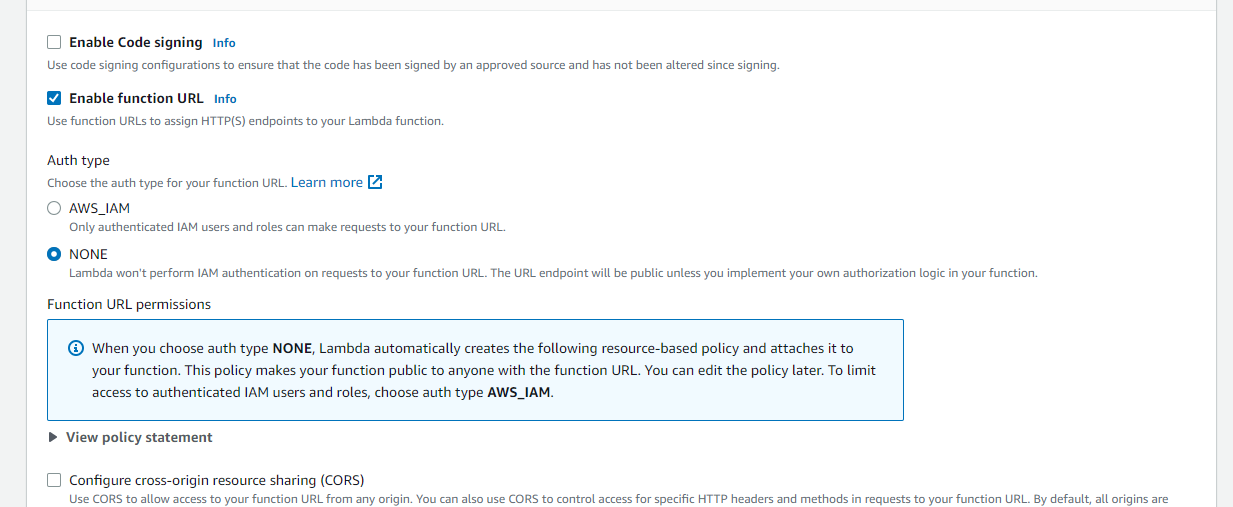
* Tạo các function Lambda
* Chọn Node.js 14 vì nhóm triển khai code với NodeJs 14



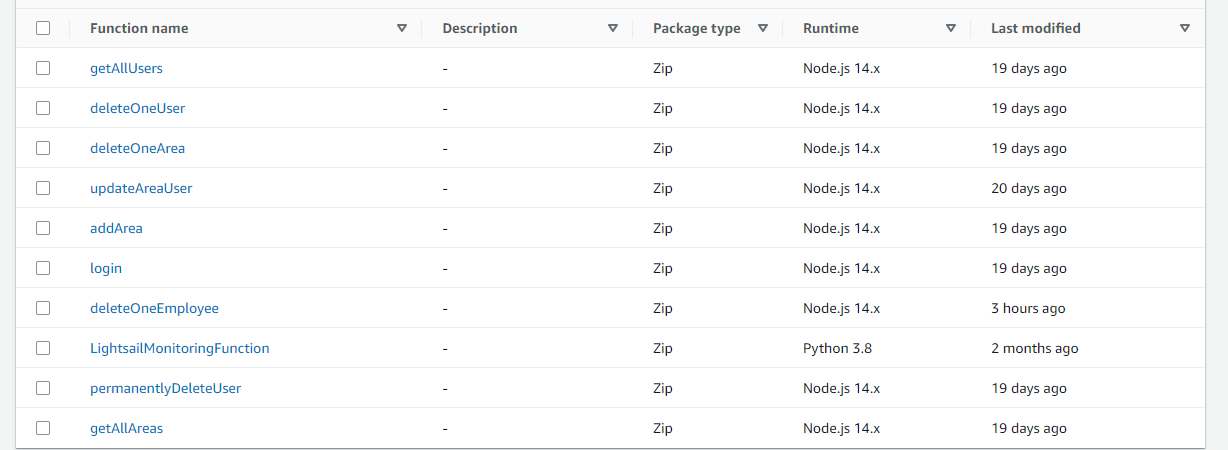
* Sử dụng các role có sẵn trong tài khoản

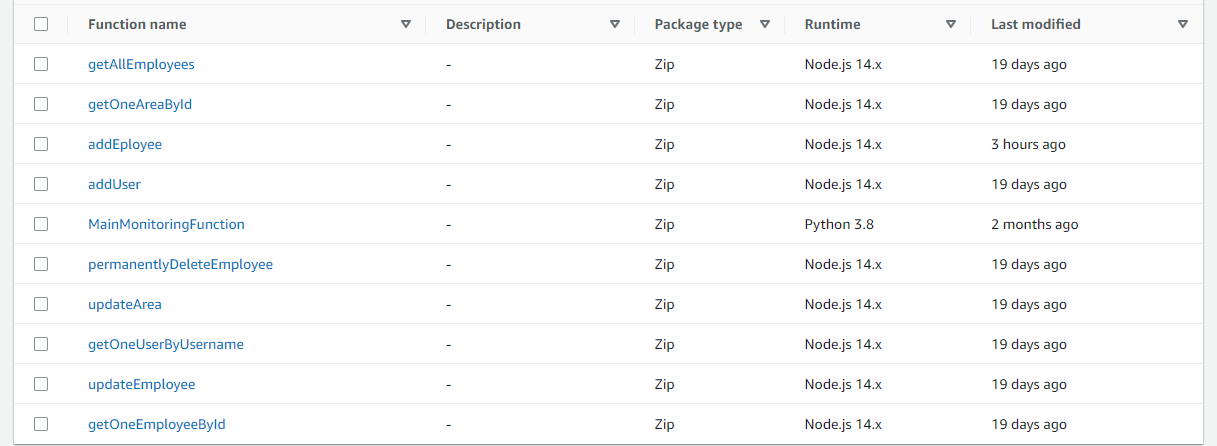


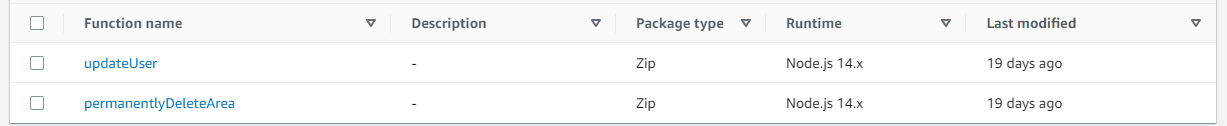
* Chọn Enable function URL để sử dụng tính năng function url của Lambda



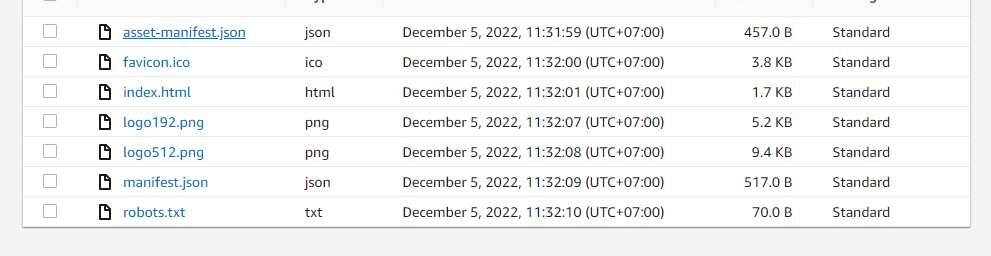
* Tương tự như trên ta tạo các function lambda cần thiết cho đề tài



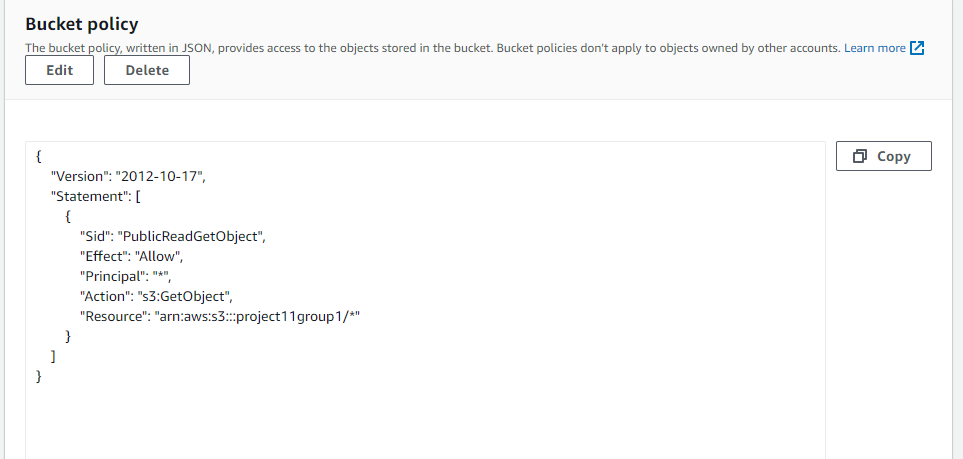




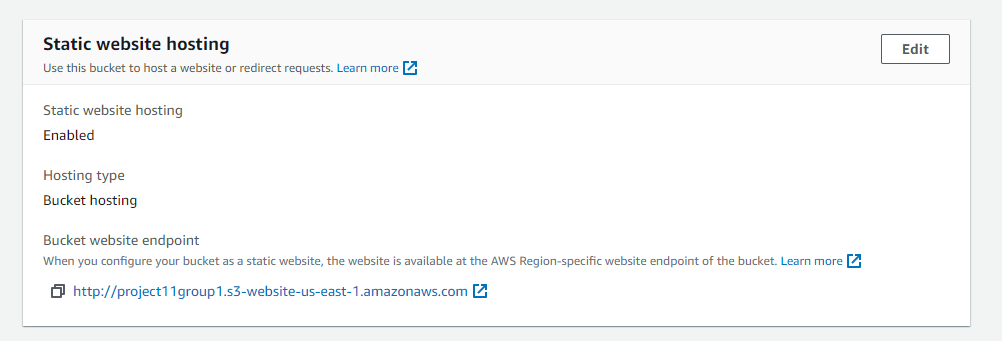
* Tải Project lên S3
* Tạo bucket
* Upload các file frontend đã build



* Vào tag Permissions -> edit Bucket policy



* Vào Properties kéo xuống phần Static website hosting để lấy đường link



# PHẦN KẾT LUẬN

## Kết quả đạt được

Sau một thời gian nghiên cứu và thực hiện đề tài “***Xây dựng 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 Lambda Function URL, DynamoDB và áp dụng kiến thức để thiết kế và xây dựng một ứng dụng 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 với Database.

### 1.2. Chương trình đã làm được

Xây dựng hoàn chỉnh một ứng dụng sử dụng các công nghệ React JS cho FrontEnd, Lambda cho BackEnd và database sử dụng DynamoDB đồng thời sử dụng các dịch dịch của AWS: S3, Lambda Function URL (NodeJS), DynamoDB với các chức năng cơ bản 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
* Sử dụng thêm các dịch vụ của AWS