Bài 12. Một số dịch vụ cao cấp của AWS

Nội dung

- Serverless và dịch vụ Lambda
- Microservice và dịch vụ Kuberness

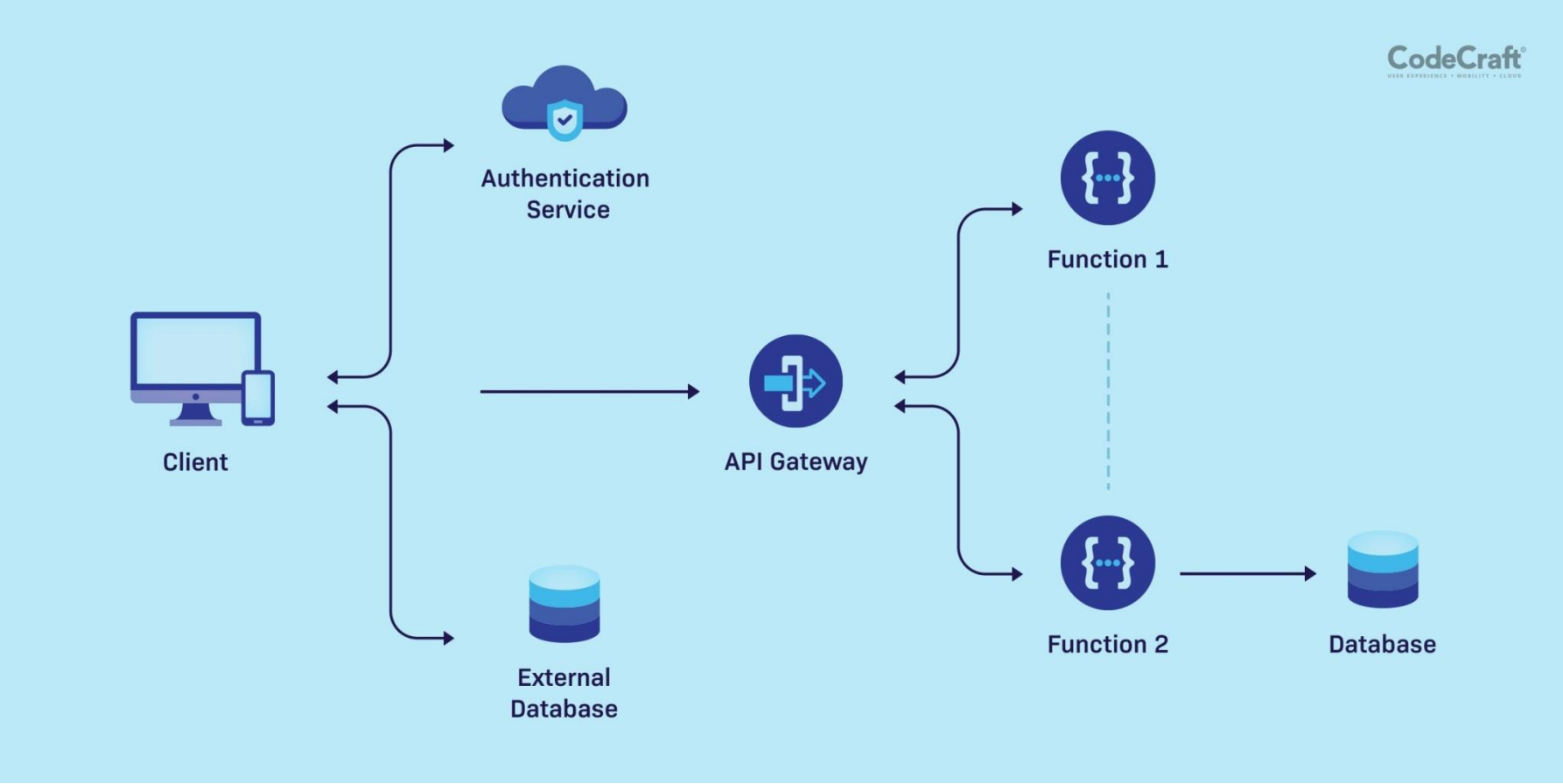
Serverless là gì?

- **Serverless là** môi trường, nền tảng thực thi ứng dụng và dịch vụ mà không phải quan tâm đến máy chủ.
- Điều quan trọng và khác biệt nhất trong serverless là bạn **chỉ trả tiền khi và chỉ những phần bạn sử dụng**.

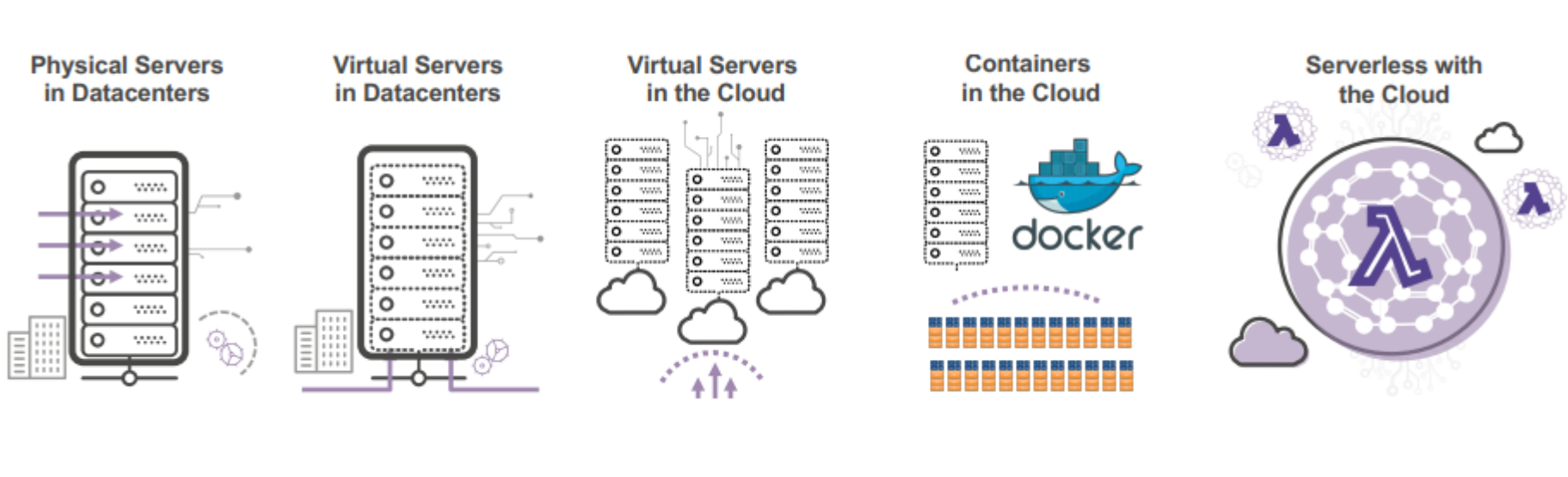
- **BaaS – Backend as a Service:**
  - *"Ở mô hình này, phần lớn code logic của chúng ta sẽ chuyển về xử lý ở phía Frontend. Còn Backend thì sử dụng các API có sẵn của bên thứ ba."*

- **FaaS – Function as a Service:**
  - Tự viết các API cho mục đích của mình, và triển khai chúng lên Server.
  - Nhưng thay vì triển khai theo mô hình client-server thông thường là phải thuê Server rồi deploy code lên đó, thì chúng ta lại **deploy code dưới dạng các Function (Function as a Service)** và các function này có thể gọi dưới dạng **RestAPI**.
  - Với mô hình **FaaS** này, chúng ta sẽ chỉ cần viết code thôi, không cần quan tâm việc server và code được lưu trữ ở đâu, các nhà cung cấp Cloud Computing bên thứ ba sẽ tự quản lý phần này.

Kiến trúc serverless



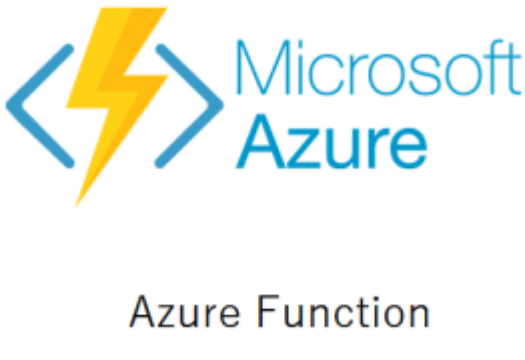
Quá trình phát triển



Ưu, nhược điểm

- Ưu điểm:
  - Người phát triển ứng dụng không lo việc quản lý và vận hành máy chủ.
  - Tiết kiệm chi phí thuê máy chủ.
  - Dễ dàng mở rộng.
  - Độ sẵn sàng cao.
- Nhược điểm:
  - Việc sử dụng ứng dụng có độ trễ nhất định
  - Việc giám sát và sửa lỗi ứng dụng khó khăn
  - Giới hạn về bộ nhớ và thời gian của nhà cung cấp dịch vụ đối với ứng dụng.
  - Phụ thuộc nhà cung cấp dịch vụ.
  - Chi phí ngầm.
  - Tốn thời gian cho việc tìm hiểu các công cụ liên quan.

Một số dịch vụ

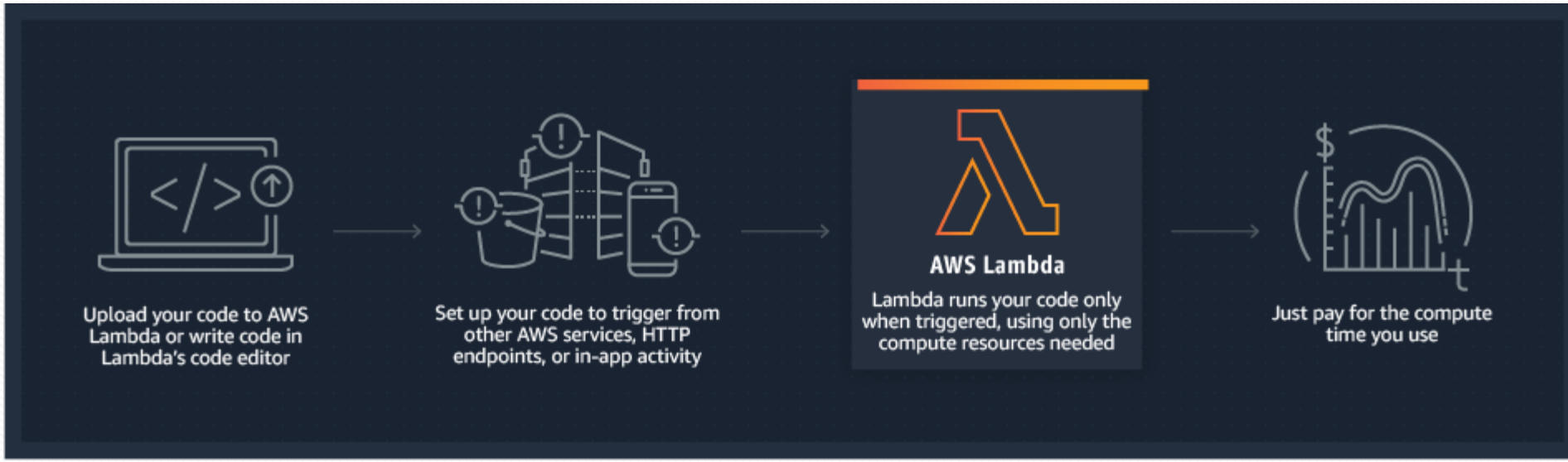


Tính năng	AWS Lambda	Google Cloud	Azure Functions
Khả năng mở rộng	Tự động	Tự động	Bằng tay hoặc theo plan đặt trước
Số Function tối đa	Không giới hạn	1000 trên 1 project	Không giới hạn
Xử lí đồng thời	1000 trên 1 account 1 region (soft limit)	Không giới hạn	Không giới hạn
Thời gian xử lí tối đa	300 sec (5 min)	540 seconds (9 minutes)	300 sec (5 min)
Ngôn ngữ	JavaScript, Java, C#, and Python	Only JavaScript	C#, JavaScript, F#, Python, Batch, PHP, PowerShell

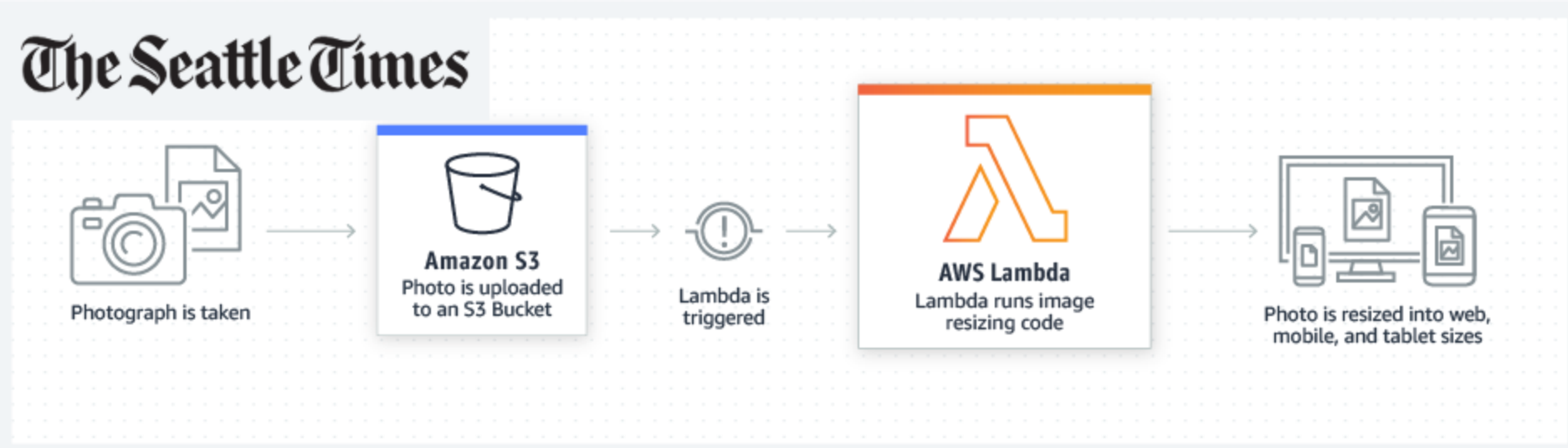
AWS Lambda

- AWS Lambda là dịch vụ của AWS cho phép chạy mã mà không cần cung cấp hay quản lý máy chủ.
- Với Lambda, bạn có thể chạy mã cho gần như toàn bộ các loại ứng dụng hay dịch vụ backend – tất cả đều không cần quản trị.
- Chỉ cần tải đoạn mã của bạn lên và Lambda sẽ lo hết những gì cần làm để chạy và mở rộng mã của bạn với mức độ có sẵn cao.
- Bạn có thể thiết lập để mã của bạn tự động kích hoạt từ các dịch vụ AWS khác, hoặc gọi trực tiếp từ bất cứ ứng dụng web hay di động nào.

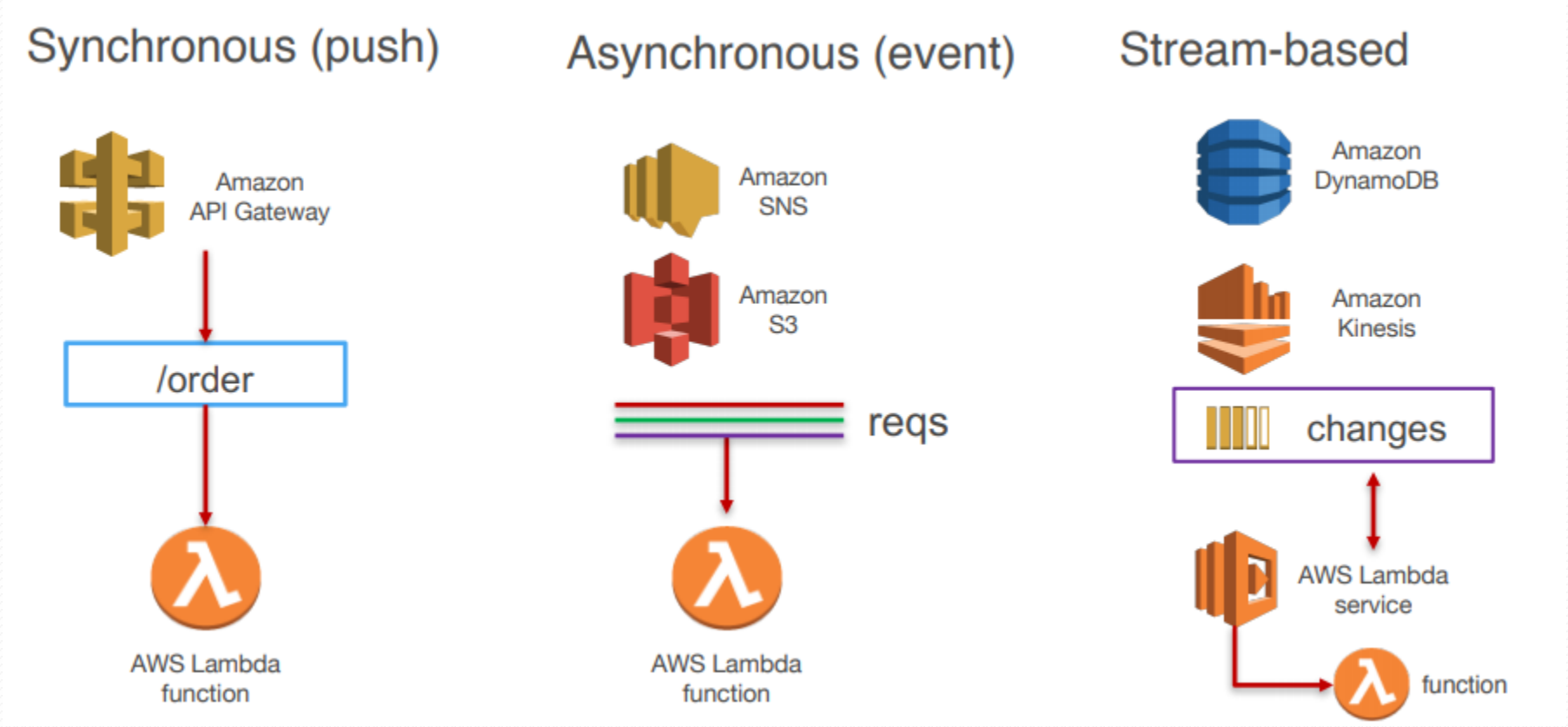
Cách thức hoạt động



Ví dụ


The Seattle Times
Photograph is taken
Amazon S3
Photo is uploaded to an S3 Bucket
Lambda is triggered
AWS Lambda
Lambda runs image resizing code
Photo is resized into web, mobile, and tablet sizes

Mô hình thực thi

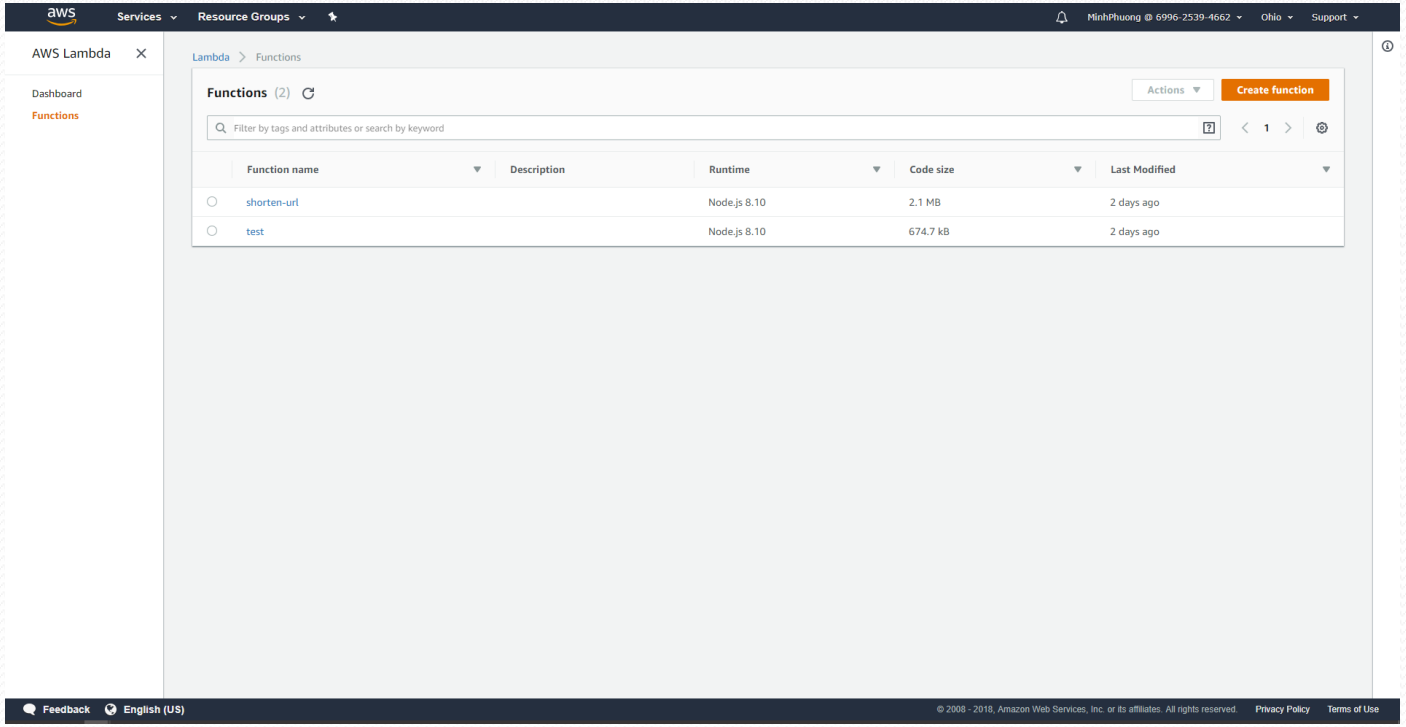


Hoạt động của Lambda

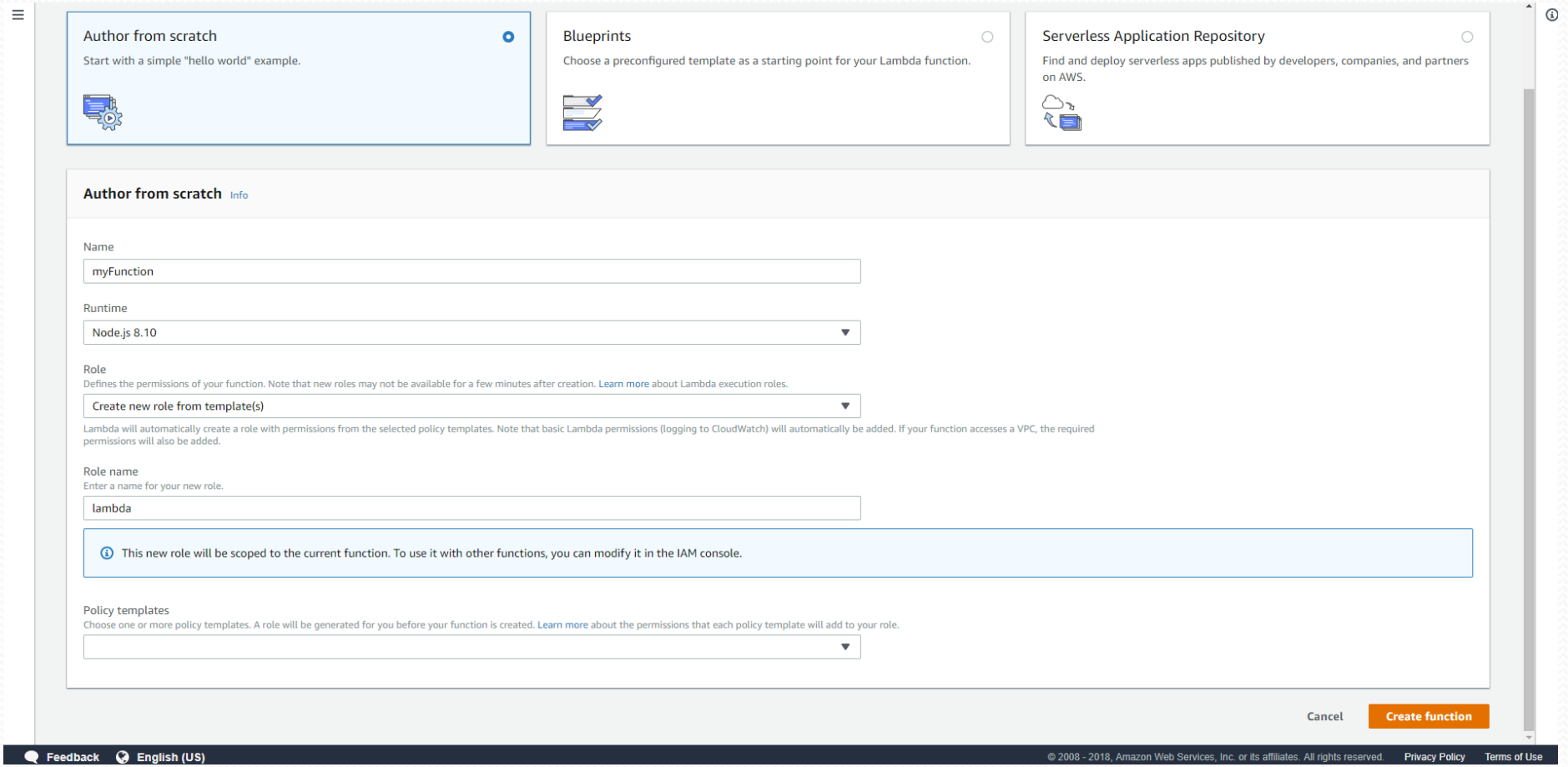
- Các thành phần chính: Lambda Function và Event Source
  - Event Source đưa ra các sự kiện
  - Lambda function: upload code lên Lambda.
  - Tính toán tài nguyên cho hệ thống là con số ước lượng mà bạn đưa ra mà function của mình cần dùng.
- Gọi Lambda function tự động: sử dụng Event Sources

Tạo ứng dụng Lambda

- Chọn mục Lambda trong nhóm dịch vụ Compute

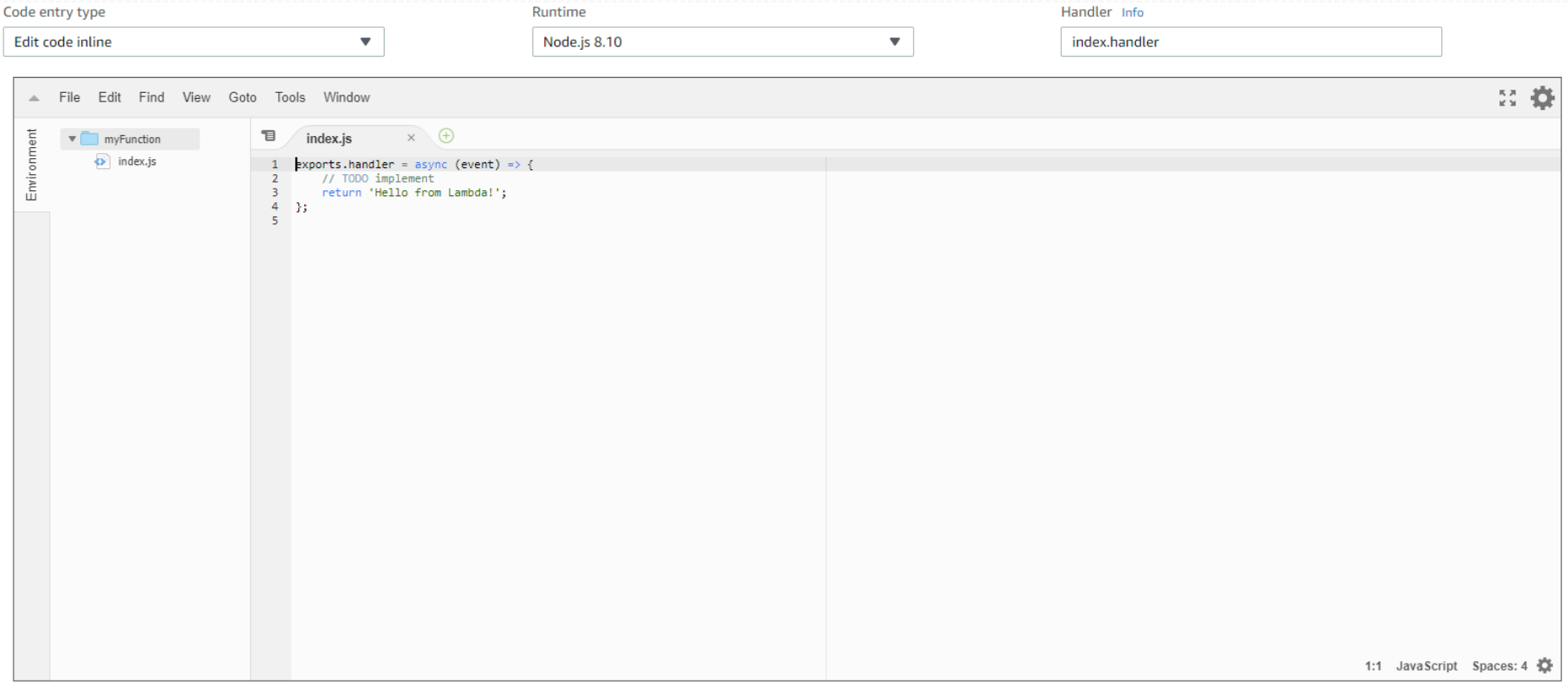


- Chọn vào Create function để tạo 1 Lambda function mới:



- Chọn ***Author from scratch*** để tạo 1 function rỗng
- Thiết lập các thông số:
  - ***Name*** là tên Lambda function,
  - ***Runtime*** là môi trường chạy code (Nodejs,...)
  - ***Role*** là vai trò tạo khởi tạo hàm,
  - ***Policy templates*** là tùy chọn sử dụng thêm các AWS mà Amazon cung cấp, sử dụng API Gateway.

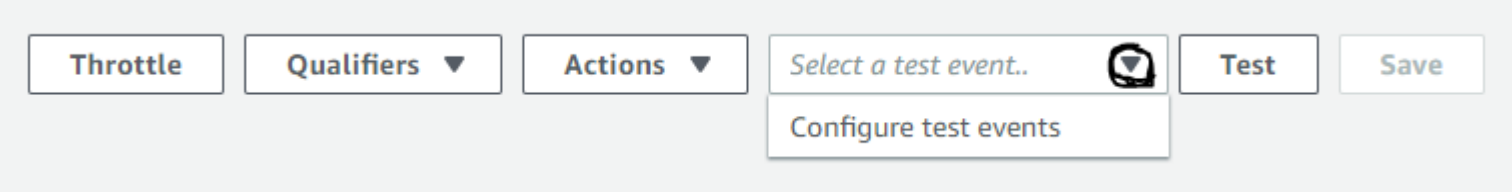
- Cửa sổ soạn thảo code



- Soạn code hoặc upload code đã có trên máy cá nhân.

Test Lambda function

- Chọn **Select a test event.. -> Configure test events**



Configure test event

A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.

- (•) Create new test event
- ( ) Edit saved test events

Event template

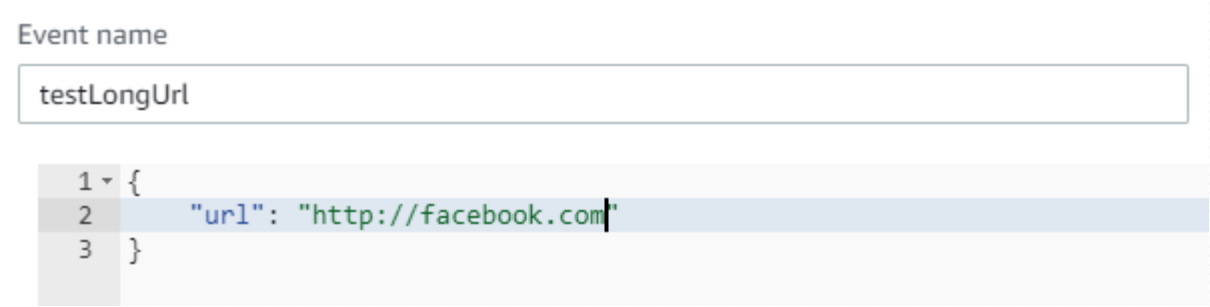
Hello World

Event name

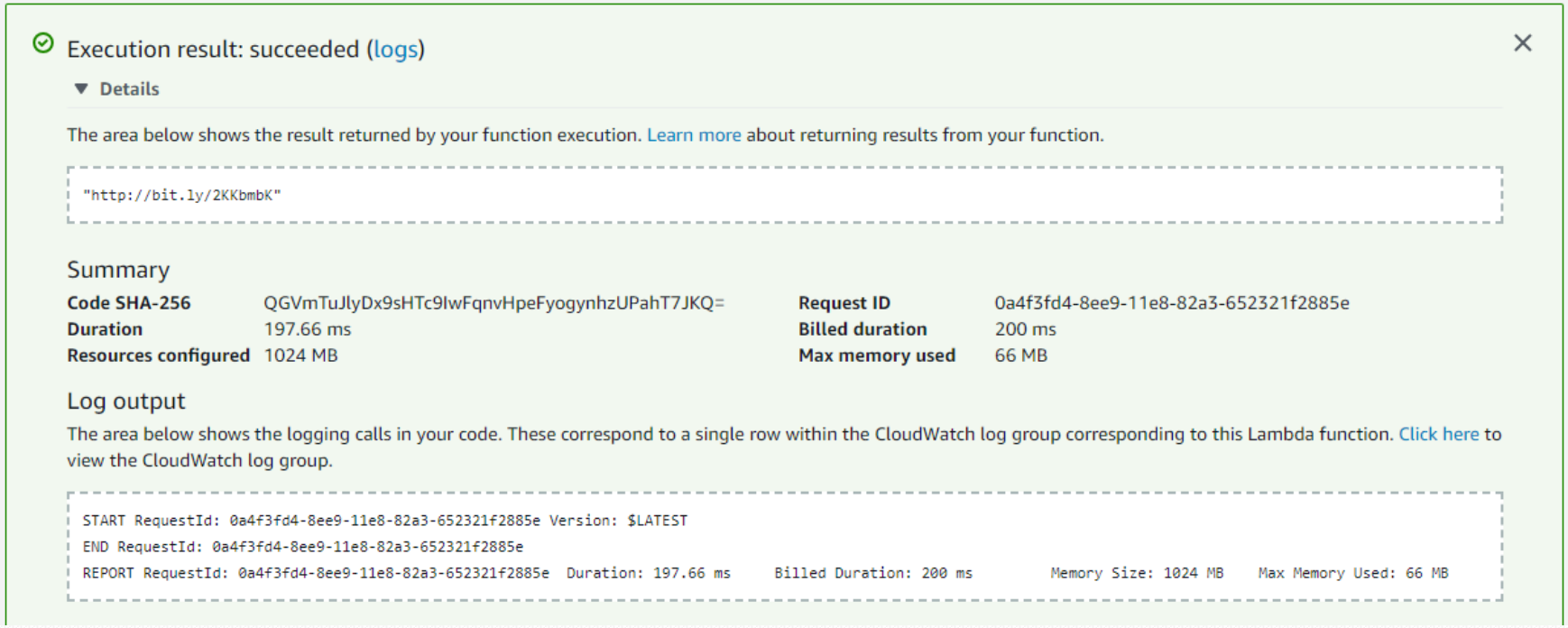
MyEventName

```
{
  "key3": "value3",
  "key2": "value2",
  "key1": "value1"
}
```

- Vì function của chúng ta nhận sự kiện *event.url* nên ta sẽ viết test như sau.

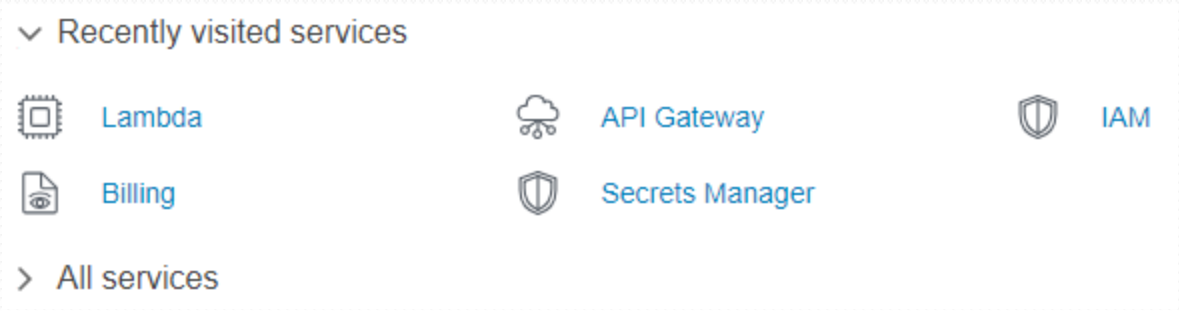


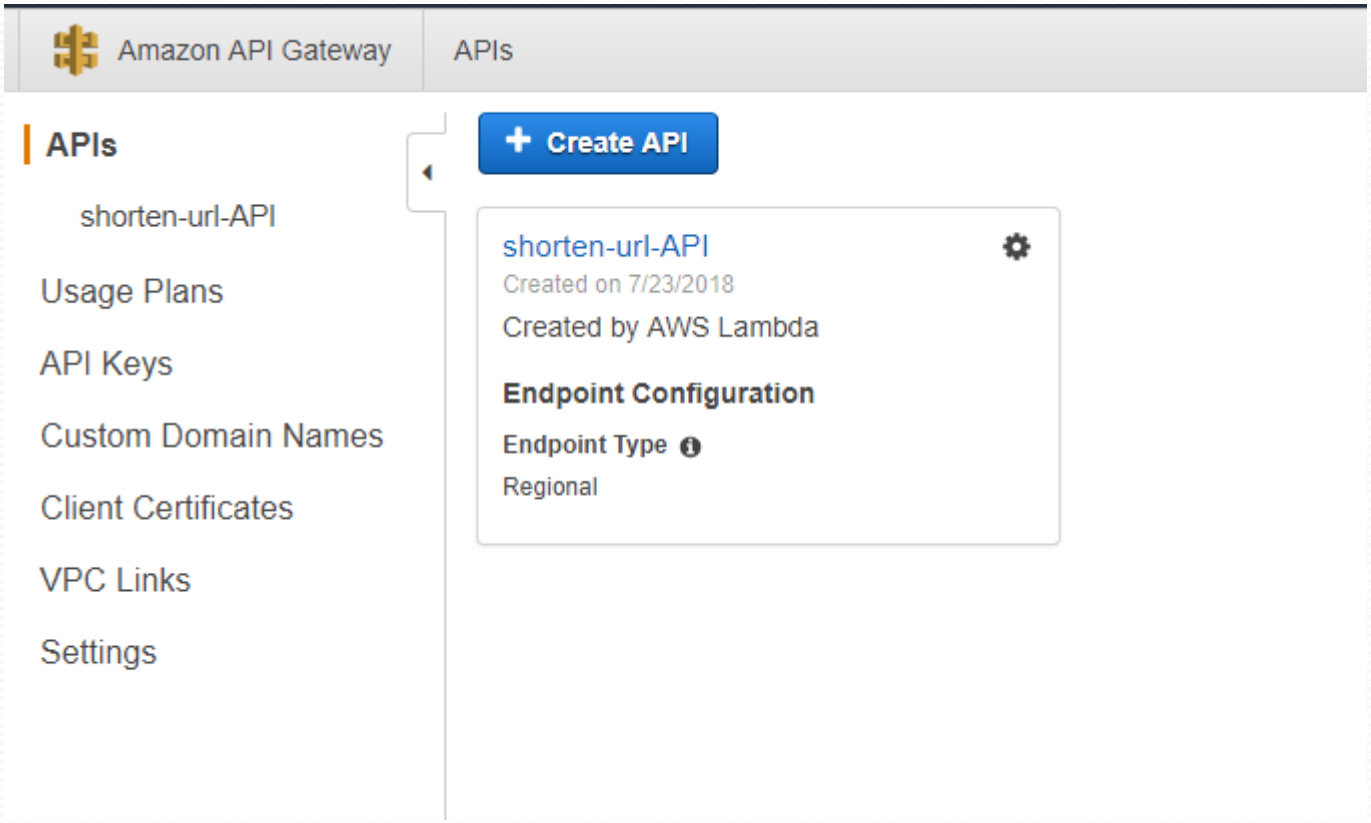
- Sau đó Save lại và nhấn chọn ***Test***, nó sẽ trả về kết quả là link đã rút gọn nhờ bitly:



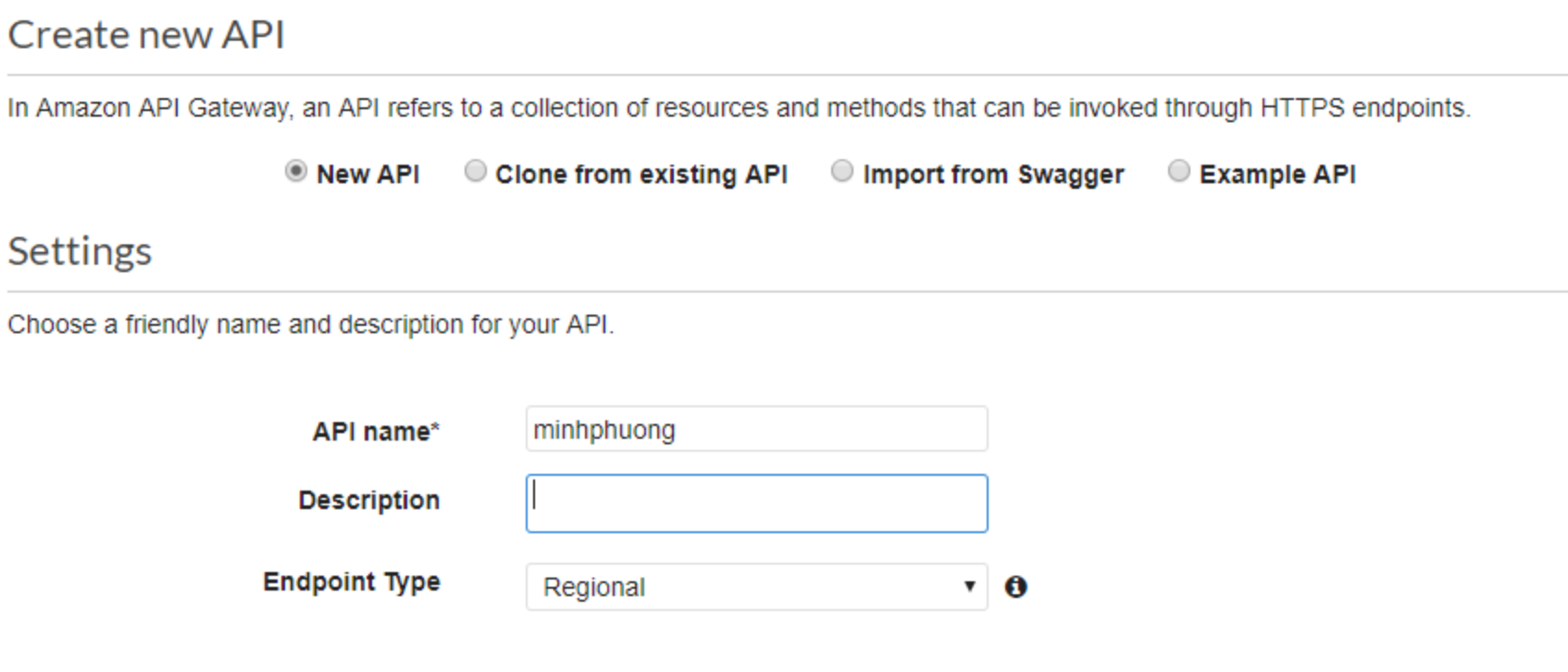
Tạo API endpoint cho Lambda function

- Vào trang ***API Gateway*** ở trang chủ của AWS

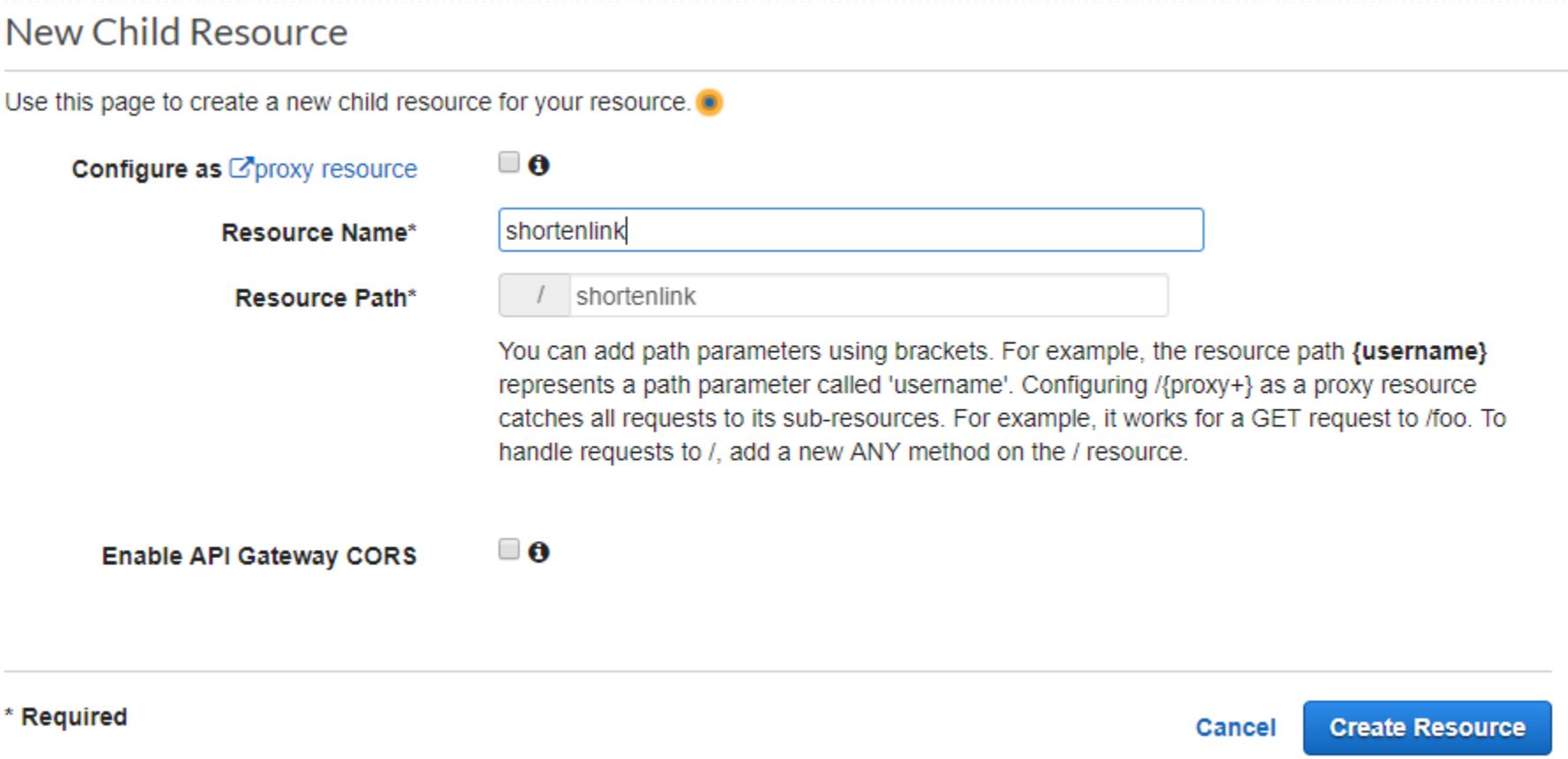



Amazon API Gateway
APIs
APIs
shorten-url-API
Usage Plans
API Keys
Custom Domain Names
Client Certificates
VPC Links
Settings
Create API
shorten-url-API
Created on 7/23/2018
Created by AWS Lambda
Endpoint Configuration
Endpoint Type
Regional

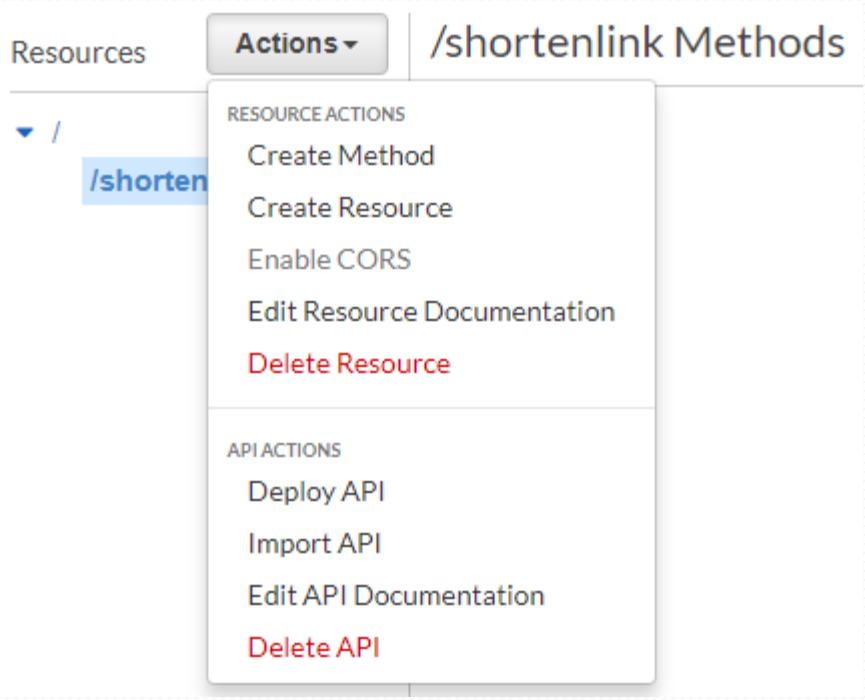
- Chọn ***Create API*** và tiến hành set name cho nó:



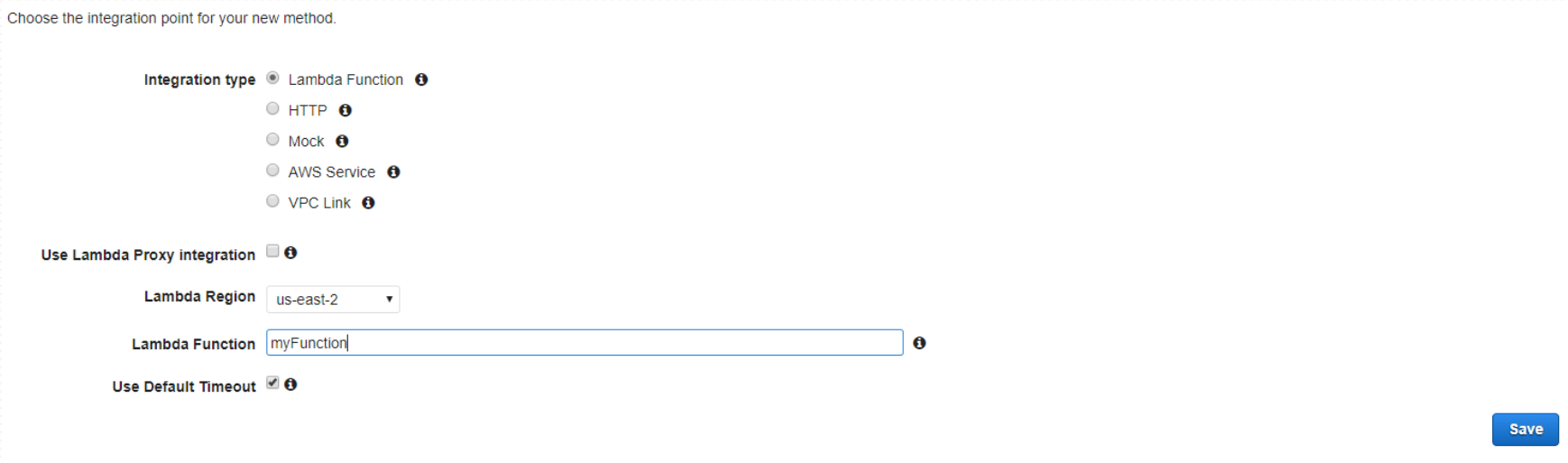
- Add resource:



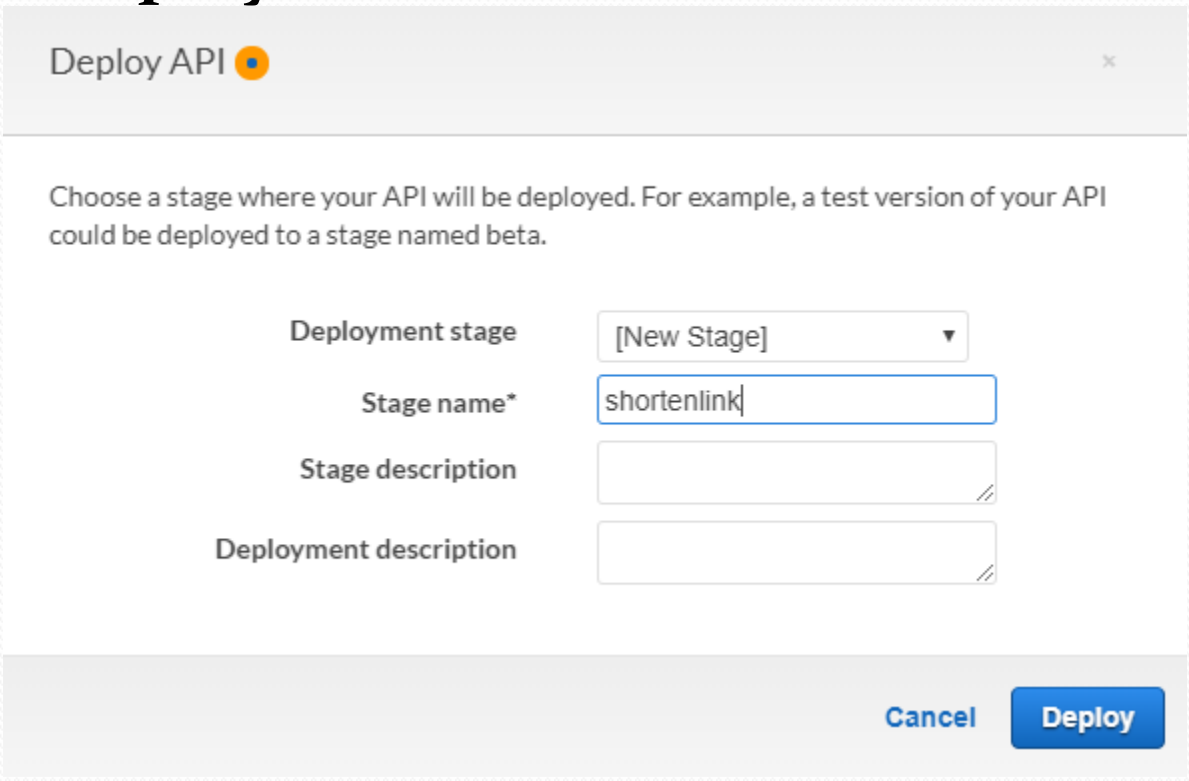
- Add methods cho resource :



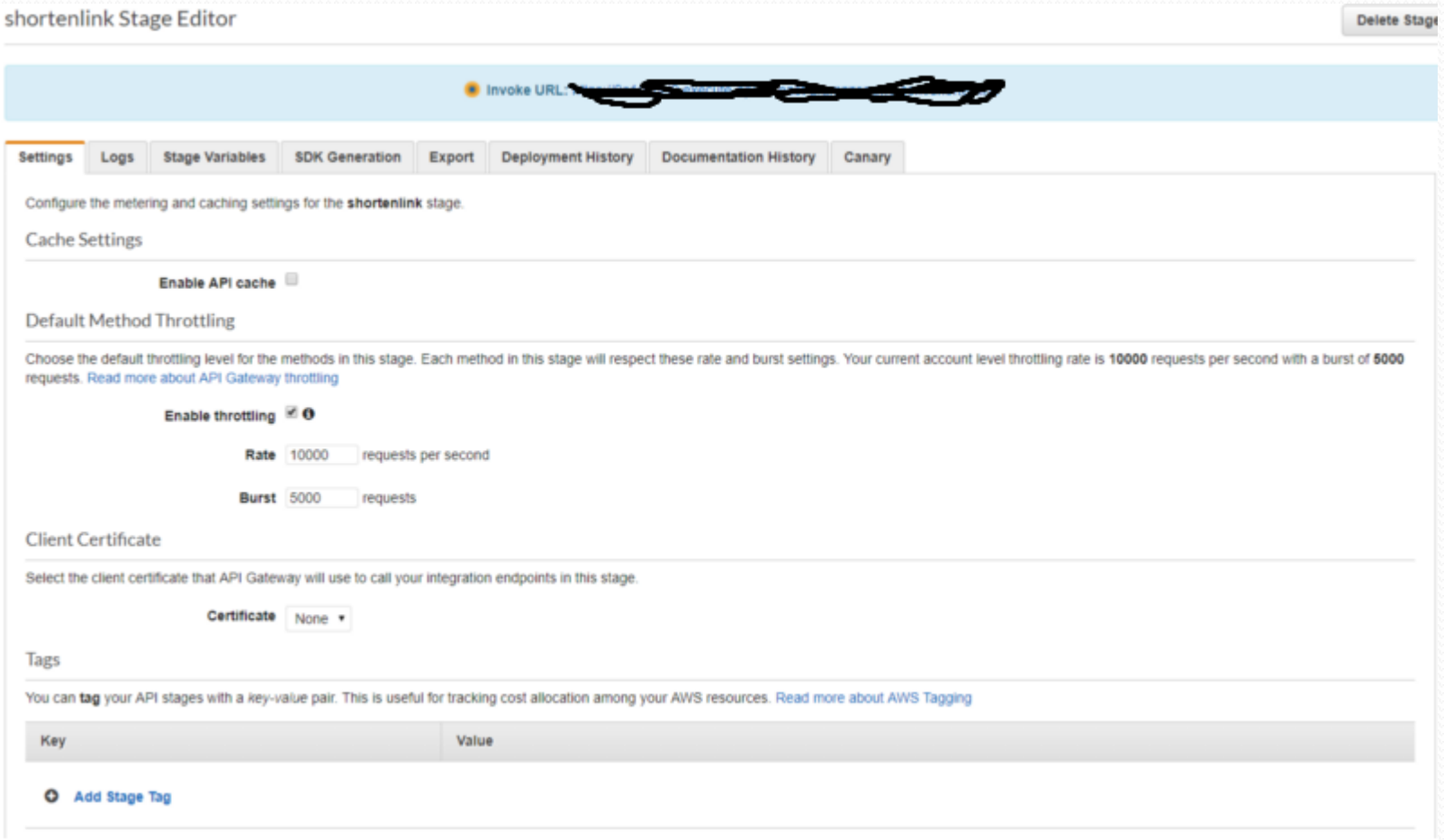
- Chọn ***Create Method*** để thêm phương thức request



- Trong bảng ***Action*** chọn ***Deploy API***, chọn ***New stage*** rồi nhập tên stage bất kì trong ô ***Stage name*** rồi nhấn ***Deploy***:

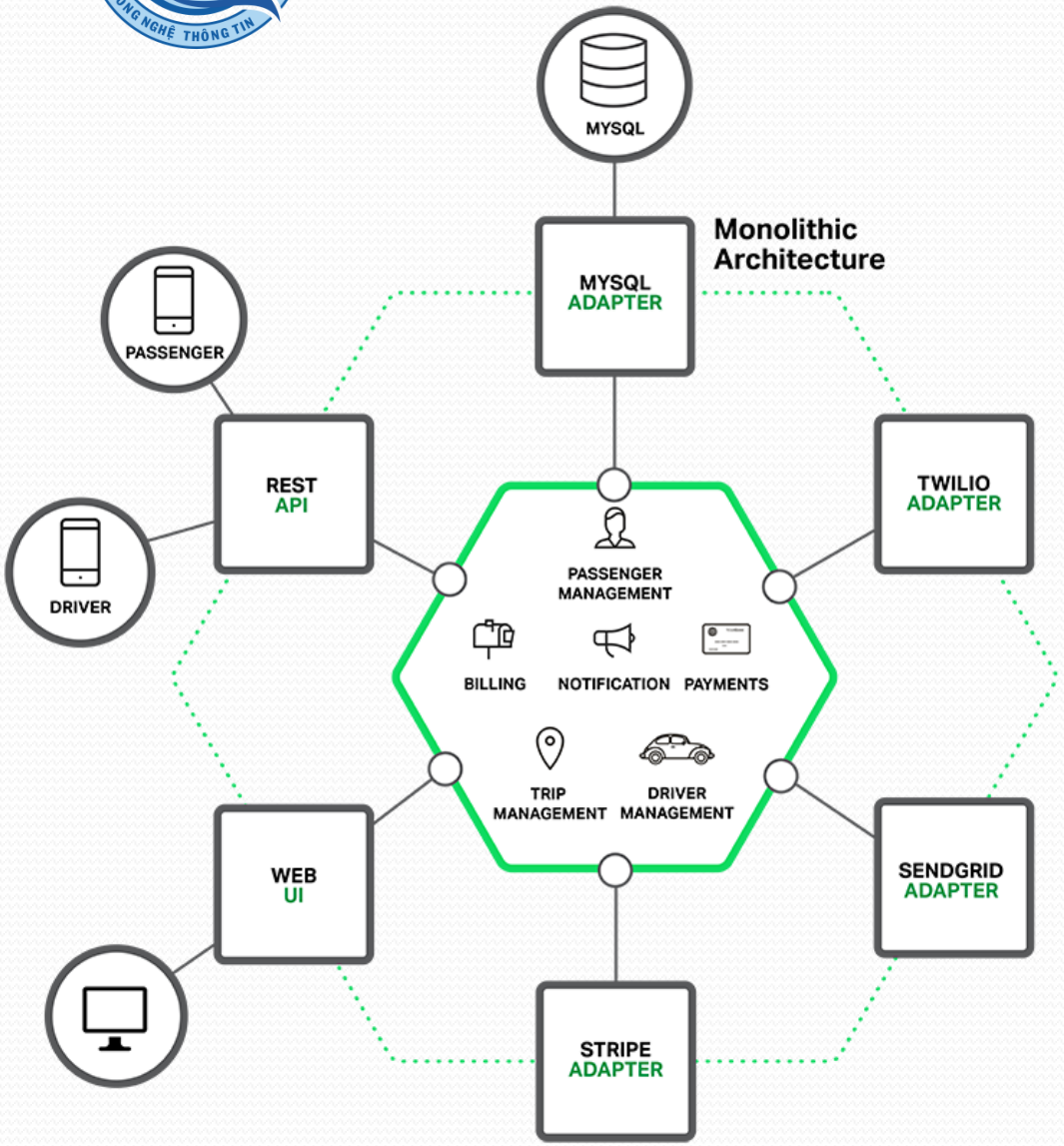


- Sau đó bạn sẽ nhận được 1 trang quản lý api của mình.
- Vào Endpoint để lấy URL

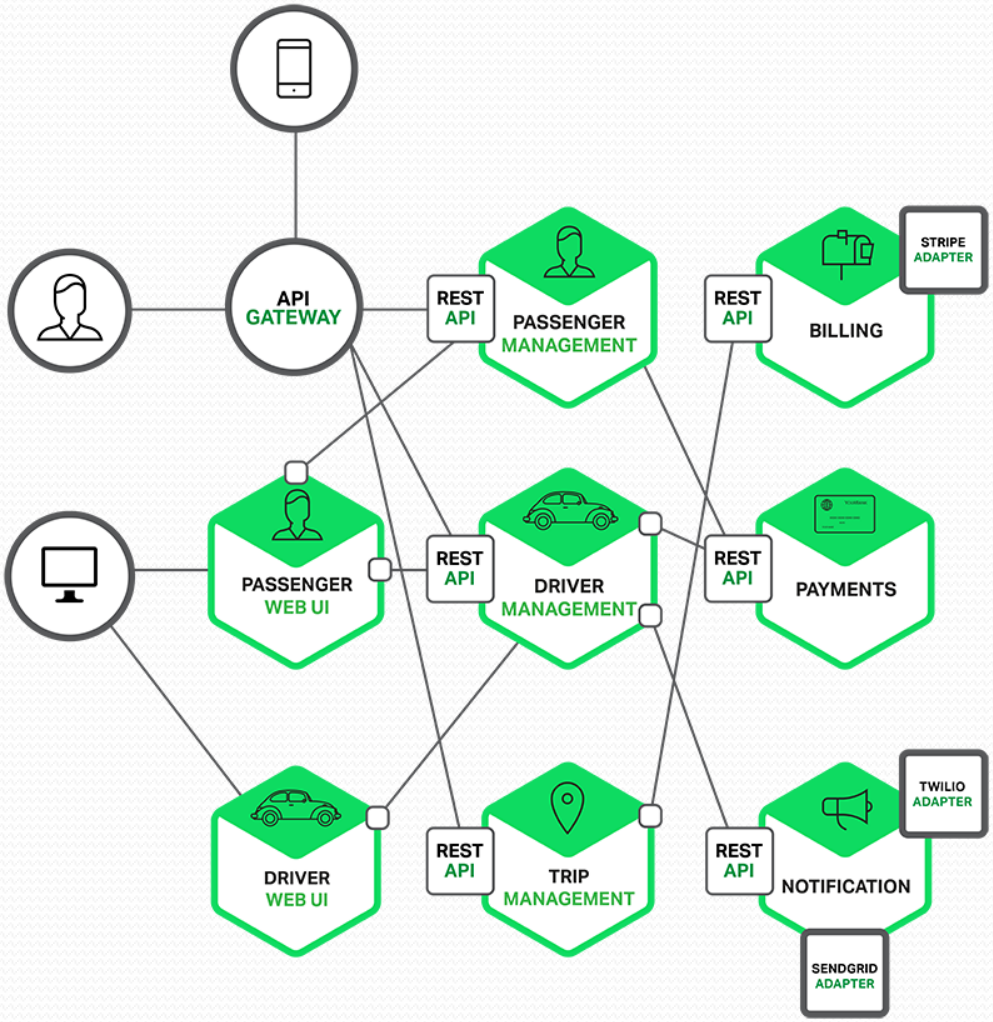


Microservices và dịch vụ Kuberness

Kiến trúc một khối (Monolithic Architecture)



- **Kiến trúc dịch vụ nhỏ (Microservices Architecture)**



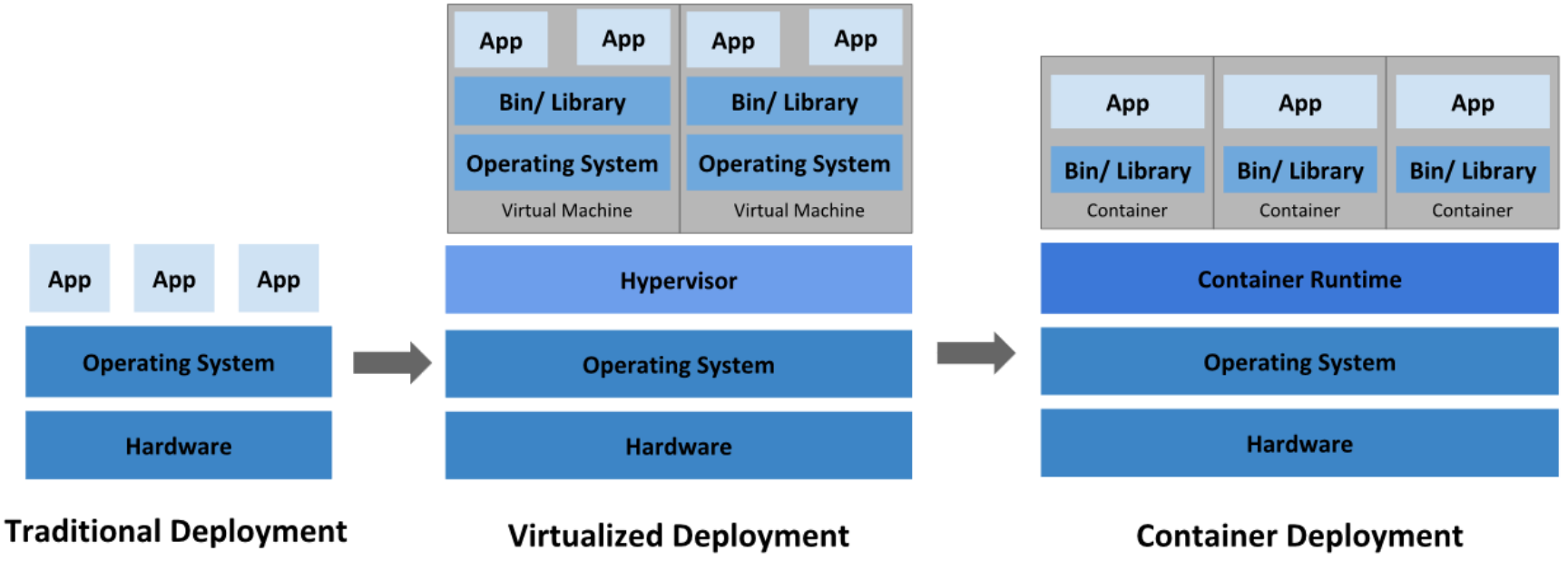
Ưu nhược điểm của Microservices

- **Ưu điểm**
- Cho phép dễ dàng triển khai các ứng dụng lớn, phức tạp:
  - Cải thiện khả năng bảo trì - mỗi service tương đối nhỏ do đó dễ hiểu và thay đổi hơn
  - Khả năng testing dễ dàng hơn - các services nhỏ hơn và nhanh hơn để test
  - Khả năng triển khai tốt hơn - các services có thể được triển khai độc lập
  - Cho phép các services được phát triển bởi những team khác nhau.
  - Giảm thiểu rủi ro: Nếu có lỗi trong một service thì chỉ có service đó bị ảnh hưởng.
  - Dễ dàng thay đổi sử dụng các công nghệ mới

- **Nhược điểm**
- Các nhà phát triển phải đối phó với sự phức tạp của việc tạo ra một hệ thống phân tán:
  - Cần cài đặt việc truyền thông giữa các inter-services
  - Xử lý lỗi rất phức tạp vì một luồng xử lý cần đi qua nhiều services
  - Việc thực hiện các requests trải rộng trên nhiều services khó khăn hơn
  - Khó khăn trong việc đảm bảo toàn vẹn CSDL nếu triển khai theo kiến trúc cơ sở dữ liệu phân vùng
  - Triển khai và quản lý microservices nếu làm thủ công theo cách đã làm với ứng dụng một khối phức tạp hơn rất nhiều
  - Phải xử lý sự cố khi kết nối chậm, lỗi khi thông điệp không gửi được hoặc thông điệp gửi đến nhiều đích đến vào các thời điểm khác nhau

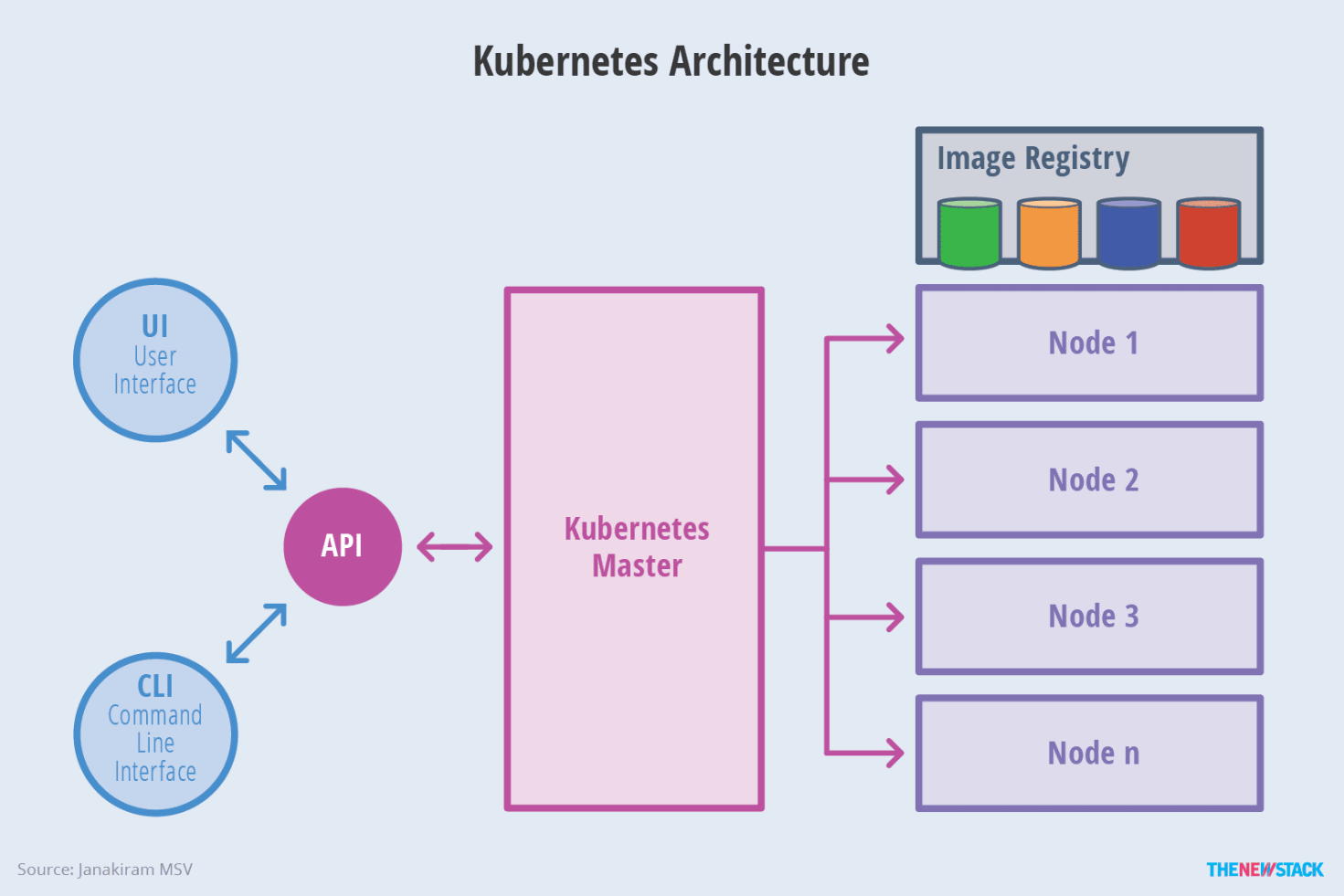
Kubernetes

- Quá trình phát triển mô hình phát triển ứng dụng



- Kubernetes là một mã nguồn mở được dùng để tự động triển khai hệ thống, scaling, quản lý các container.

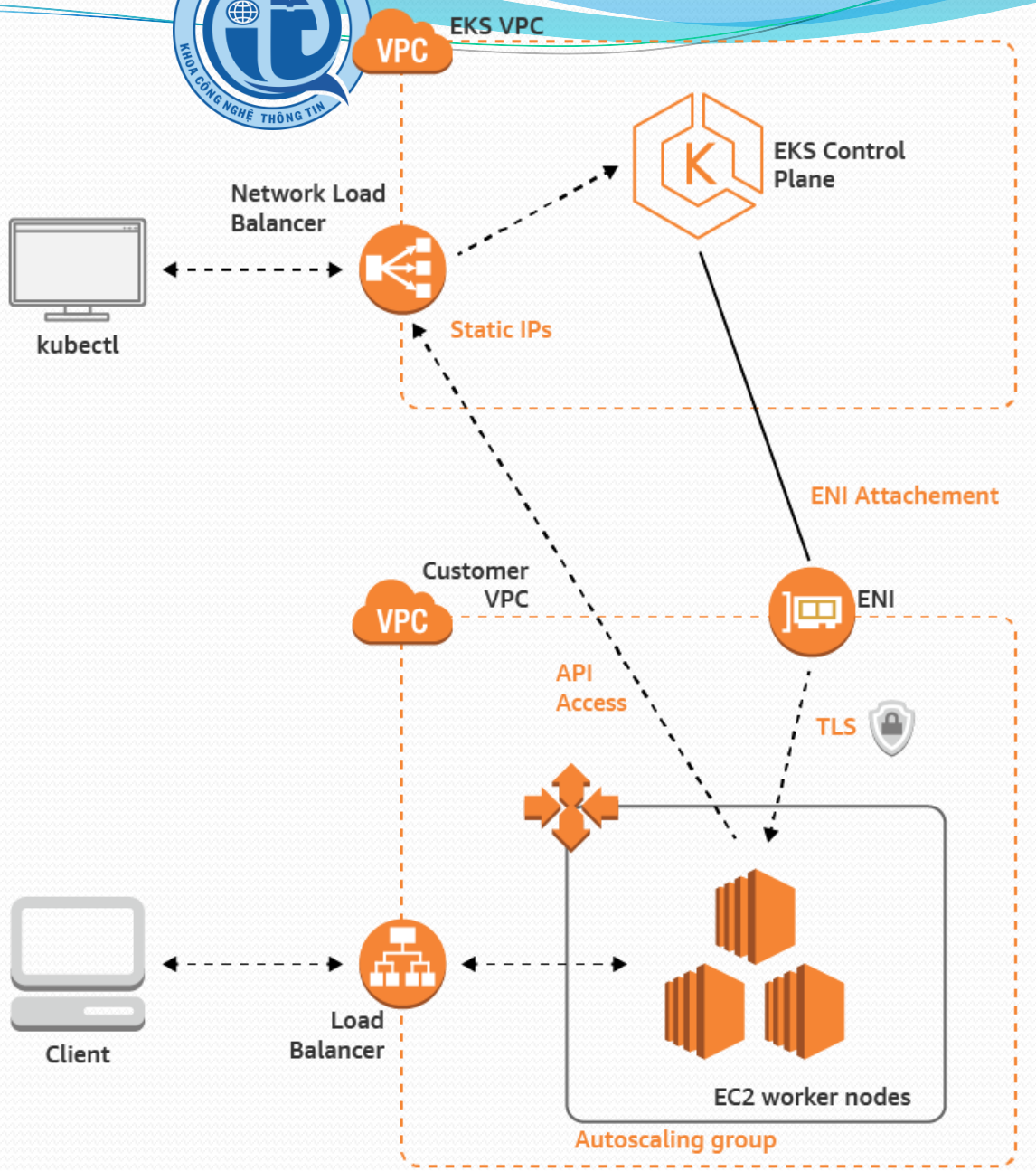
Kiến trúc Kubernetes



Amazon Elastic Kubernetes Service

- Amazon Elastic Kubernetes Service (Amazon EKS) là một dịch vụ Kubernetes được quản lý hoàn toàn.
- AWS EKS cho phép bạn deploy, run Kubernetes trên AWS.
- AWS EKS sẽ deploy Kubernetes master, rồi add các worker node vào cluster.
- Sau đó bạn có thể dùng Kubernetes command để kết nối và thao tác với Kubernetes.

- Mô hình ứng dụng triển khai trên Kubernetes



Tổng kết

- Serverless và Kubernetes là những dịch vụ cho các ứng dụng lớn, phức tạp.
- Điện toán đám mây là môi trường thuận lợi cho việc phát triển những dịch vụ này.