# Tích hợp các nền tảng tổng hợp và nhà hàng dịch vụ nhanh (Quick Service Restaurants) với kiến trúc không máy chủ (serverless) của AWS

tác giả Mike Gomez , Salman Ahmed và Sunaina Karve vào 30 THÁNG 4 NĂM 2025 trong [Amazon API Gateway](https://aws.amazon.com/blogs/compute/category/application-services/amazon-api-gateway-application-services/), [Amazon DynamoDB](https://aws.amazon.com/blogs/compute/category/database/amazon-dynamodb/), [Amazon EventBridge](https://aws.amazon.com/blogs/compute/category/application-integration/amazon-eventbridge/), [AWS Lambda](https://aws.amazon.com/blogs/compute/category/compute/aws-lambda/), [AWS Step Functions](https://aws.amazon.com/blogs/compute/category/application-services/aws-step-functions/), [Intermediate (200)](https://aws.amazon.com/blogs/compute/category/learning-levels/intermediate-200/), [Serverless](https://aws.amazon.com/blogs/compute/category/serverless/) [Permalink](https://aws.amazon.com/blogs/compute/integrating-aggregators-and-quick-service-restaurants-with-aws-serverless-architectures/)  [Chia sẻ](https://aws.amazon.com/vi/blogs/compute/integrating-aggregators-and-quick-service-restaurants-with-aws-serverless-architectures/#)

Bài viết này hướng dẫn cách dùng các dịch vụ serverless của AWS, như [Amazon EventBridge](https://aws.amazon.com/eventbridge/) và [AWS Lambda](https://aws.amazon.com/lambda/), để xây dựng kết nối giữa các QSR và các nền tảng đặt món, giao đồ ăn trực tuyến. Các aggregator giúp QSR mở rộng tệp khách hàng, bổ sung tùy chọn giao hàng và thúc đẩy tăng trưởng.

## **Tổng quan về QSR**

QSR ưu tiên tốc độ và sự thuận tiện, cung cấp thực đơn gọn nhẹ. Để đáp ứng kỳ vọng ngày càng cao của người tiêu dùng, QSR có thể tích hợp API với các nền tảng tổng hợp bên thứ ba. Sự kết hợp này cho phép QSR mở rộng chức năng, thêm các phương thức thanh toán đa dạng và dịch vụ giao hàng – những tính năng đã trở thành tiêu chuẩn trong phân khúc này.

Ở hậu trường, API điều phối tương tác giữa aggregator và QSR để đảm bảo trải nghiệm đặt hàng – giao hàng nhất quán.

Mục tiêu kinh doanh của QSR:

* Đảm bảo trải nghiệm đặt hàng và giao hàng nhất quán
* Cung cấp thực đơn cá nhân hóa
* Giữ chân khách hàng quay lại
* Giảm hủy đơn giao hàng của bên thứ ba do thiếu tùy chọn cá nhân hóa

Bài viết bắt đầu với kiến trúc đơn giản và dần bổ sung thành phần để giải quyết các thách thức kiến trúc.

## **Kiến ​​​​trúc**

Một chuỗi nhà hàng địa phương muốn mở rộng đã tìm đến bạn – kiến trúc sư giải pháp – để thiết kế kiến trúc tích hợp tối ưu, đáp ứng yêu cầu kỹ thuật, tinh giản vận hành và nâng cao trải nghiệm khách hàng.

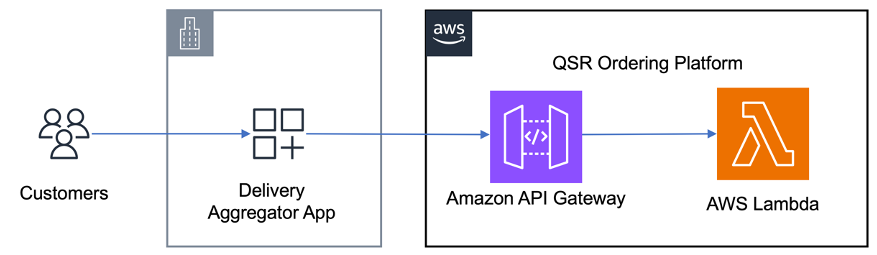
Trung tâm của tích hợp là [Amazon API Gateway](https://aws.amazon.com/api-gateway/), nhận đơn hàng từ nhiều aggregator. API Gateway đóng vai trò “cửa trước”, kết nối QSR với khách hàng để xử lý đơn động.

Phía sau là các hàm Lambda, có nhiệm vụ xác thực đơn và giao tiếp an toàn với aggregator. Lambda tự động co giãn theo lưu lượng, tối ưu tài nguyên và chi phí.

**Luồng đặt hàng**

Các bước sau đây phác thảo quá trình tích hợp không cần máy chủ giữa API Gateway và các chức năng Lambda, như thể hiện trong hình sau:

* Khách hàng có thể đặt hàng thông qua các đơn vị tổng hợp giao đồ ăn hoặc hệ thống đặt hàng riêng của doanh nghiệp.
* Yêu cầu đặt hàng được gửi đến API Gateway.



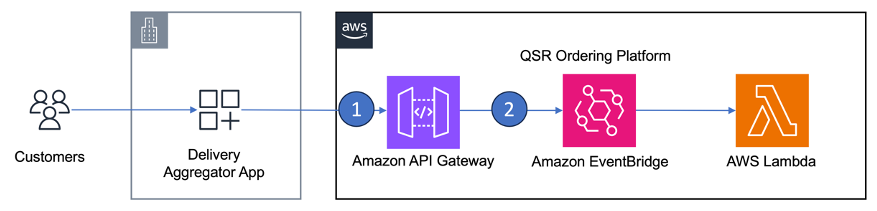
Mô hình này phù hợp với tích hợp nhỏ, đơn giản. Để mở rộng cho lưu lượng cao, nên dùng tích hợp bất đồng bộ để giảm phụ thuộc giữa API và Lambda.

**Luồng định tuyến đơn**

Các bước sau đây phác thảo tích hợp không máy chủ trong đó API Gateway kết nối với các chức năng Lambda thông qua [Amazon EventBridge](https://aws.amazon.com/eventbridge/) dưới dạng dịch vụ định tuyến sự kiện, như thể hiện trong hình sau:

1. API Gateway nhận được yêu cầu đặt hàng.
2. API Gateway định tuyến yêu cầu đặt hàng của khách hàng đến bus EventBridge để xử lý.

EventBridge định tuyến các sự kiện (ví dụ như thay đổi trạng thái đơn hàng) đến các hàm Lambda, đảm bảo khả năng phục hồi trong quá trình gián đoạn dịch vụ. Điều này loại bỏ việc xử lý lỗi thủ công và giữ cho QSR và trình tổng hợp được đồng bộ hóa.



EventBridge cung cấp các khả năng thiết yếu sau:

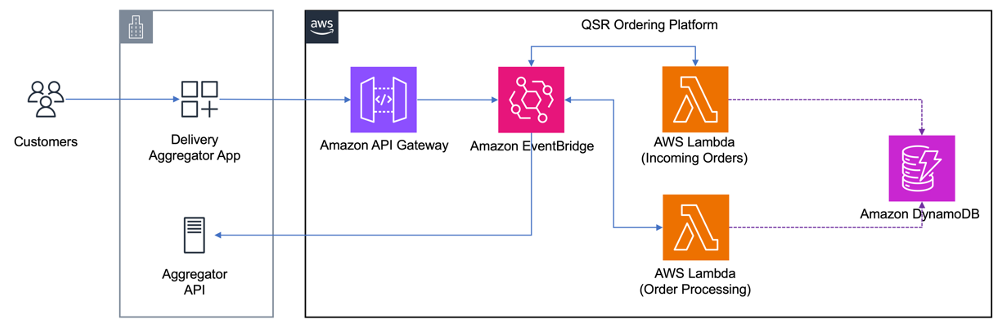
* EventBridge nhận các sự kiện được kích hoạt bởi nhiều hành động khác nhau, chẳng hạn như đơn hàng mới hoặc cập nhật menu.
* Nó định tuyến các sự kiện đến các hàm Lambda có liên quan, khởi tạo các hành động thích hợp.
* EventBridge hỗ trợ [phát lại sự kiện](https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-archive.html) , cho phép khôi phục từ các sự cố triển khai Lambda hoặc lỗi chức năng. Tính năng này cho phép duy trì hoạt động kinh doanh bằng cách lưu trữ các sự kiện trong thời gian gián đoạn dịch vụ và tự động tiếp tục xử lý khi hệ thống ổn định.

Để duy trì lịch sử đơn hàng và cho phép truy xuất dữ liệu nhanh, hệ thống cần một cơ sở dữ liệu có hiệu suất cao. [Amazon DynamoDB](https://aws.amazon.com/dynamodb/), một dịch vụ cơ sở dữ liệu NoSQL không máy chủ, đáp ứng các yêu cầu này bằng cách lưu trữ và quản lý hiệu quả thông tin đơn hàng và siêu dữ liệu. Chức năng Lambda xử lý đơn hàng tương tác với DynamoDB để duy trì thông tin chi tiết về đơn hàng. Phương pháp này cho phép xử lý không đồng bộ dữ liệu được lưu trữ bởi các quy trình phụ trợ khác. Giải pháp cơ sở dữ liệu cung cấp khả năng mở rộng và khả năng phản hồi cần thiết để xử lý khối lượng đơn hàng ngày càng tăng trong khi vẫn duy trì hiệu suất nhất quán, tách biệt việc tiếp nhận đơn hàng khỏi các bước xử lý tiếp theo.

**Quy trình xử lý đơn hàng**

Các bước sau đây phác thảo quy trình xử lý đơn hàng như thể hiện trong hình sau:

* Hàm Lambda xử lý đơn hàng sẽ xác thực đơn hàng và cập nhật cơ sở dữ liệu DynamoDB bằng thông tin chi tiết đơn hàng mới.
* Chức năng này xuất bản các sự kiện lỗi tới EventBridge, cho phép xử lý hạ lưu để xử lý lỗi và logic thử lại. Các sự kiện này có thể kích hoạt nhiều chức năng Lambda được thiết kế để quản lý các tình huống lỗi cụ thể và quy trình phục hồi.



**Các mẫu triển khai EventBridge: phương pháp tiếp cận bus đơn hoặc bus kép**

EventBridge cung cấp nhiều phương pháp tiếp cận cho cấu trúc bus sự kiện. Kiến trúc sư có thể chọn sử dụng một [bus sự kiện duy nhất](https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-what-is-how-it-works-concepts.html) với các mẫu sự kiện riêng biệt dựa trên trạng thái đơn hàng hoặc triển khai chiến lược [nhiều bus](https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-what-is-how-it-works-concepts.html).

Phương pháp tiếp cận bus đơn sử dụng một bus sự kiện cho tất cả các sự kiện với các mẫu quy tắc định tuyến dựa trên trạng thái đơn hàng. Ví dụ, các quy tắc sẽ khớp với các trạng thái cụ thể (ví dụ: "mới" hoặc "đã xử lý") để kích hoạt các hàm Lambda phù hợp. Mặc dù về mặt kiến ​​trúc, phương pháp này đơn giản, nhưng cần quản lý cẩn thận lược đồ sự kiện để tránh các lỗi tiềm ẩn. Tuy nhiên, phương pháp tiếp cận bus đơn yêu cầu xử lý cẩn thận để ngăn chặn quá trình xử lý đệ quy, trong đó các thông báo kích hoạt các thông báo bổ sung trong một vòng lặp vô tận.

Ngoài ra, phương pháp đa bus, tách biệt việc đặt lệnh và xử lý trên các bus khác nhau, có hiệu quả ngăn ngừa các vấn đề về vòng lặp và đệ quy. Phương pháp này cung cấp khả năng tách biệt các giao dịch tốt hơn, mặc dù có thiết lập phức tạp hơn một chút.

EventBridge có thể nhắm mục tiêu trực tiếp đến các dịch vụ bên ngoài bằng cách sử dụng tùy chọn [API đích](https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-api-destinations.html), loại bỏ nhu cầu sử dụng các chức năng Lambda cho tích hợp của bên thứ ba.

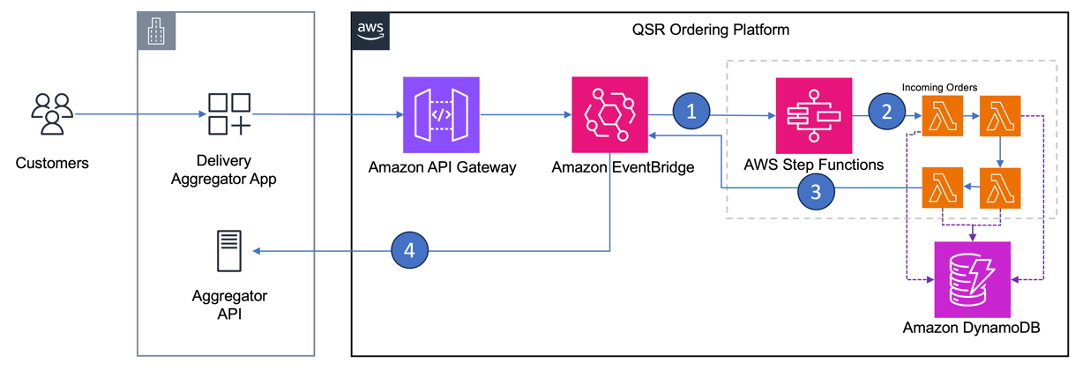
**Điều phối xử lý đơn hàng**

Trong các hệ thống xử lý đơn hàng phức tạp cho QSR, việc quản lý nhiều hàm Lambda phụ thuộc lẫn nhau có thể trở nên khó khăn, có khả năng dẫn đến mã phức tạp và kiến ​​trúc khó bảo trì. Để giải quyết vấn đề này, [AWS Step Functions](https://aws.amazon.com/step-functions/) có thể được giới thiệu như một lớp phối hợp.

Step Functions hoạt động như một điều phối viên trung tâm cho logic kinh doanh cần thiết trong luồng đơn hàng QSR. Dịch vụ này quản lý tiến trình của các hoạt động trong quy trình xử lý đơn hàng, do đó phối hợp hiệu quả các nhiệm vụ như chuẩn bị bếp và hậu cần giao hàng. Việc xác định và quản lý các quy trình làm việc phức tạp cho phép Step Functions tối ưu hóa hiệu quả chung của các hoạt động QSR, cung cấp một giải pháp có cấu trúc và thích ứng. Sự phối hợp này nâng cao khả năng xử lý xử lý động của nhà hàng, đạt được sự tích hợp mượt mà và phản hồi nhanh với các dịch vụ giao hàng trong khi hợp lý hóa kiến ​​trúc cơ bản.

Các bước sau đây phác thảo quá trình sắp xếp xử lý đơn hàng, như thể hiện trong hình sau:

* Quá trình xử lý đơn hàng kích hoạt hàm Lambda tương ứng, hàm này cập nhật dữ liệu đơn hàng trong cơ sở dữ liệu DynamoDB.
* Thứ tự cập nhật sẽ khả dụng cho các hàm Lambda tiếp theo xử lý nhiều logic kinh doanh hơn được thực hiện bởi các hàm Lambda tiếp theo.



Trong kiến ​​trúc EventBridge đa bus, luồng quy trình như sau:

1. Bus EventBridge đầu tiên nhận sự kiện lệnh ban đầu và định tuyến nó đến quy trình làm việc Step Functions.
2. Quy trình làm việc Step Functions sắp xếp quá trình xử lý đơn hàng, phối hợp nhiều tác vụ và kiểm tra khác nhau.
3. Khi hoàn tất, quy trình làm việc Step Functions sẽ phát ra sự kiện có kết quả xử lý tới bus EventBridge thứ hai.
4. Dựa trên kết quả từ quy trình làm việc Step Function, bus thứ hai này chứa một quy tắc kích hoạt Aggregator API làm đích đến của API.

**Quy trình làm việc tương tác của người dùng**

Khi khách hàng đặt hàng, phải có cách để xác nhận hoặc thông báo cho họ khi đơn hàng đã sẵn sàng. Vì mục đích này, bạn có thể sử dụng dịch vụ [AWS End User Messaging](https://aws.amazon.com/end-user-messaging/) để đẩy thông báo hoàn tất đơn hàng và các ưu đãi mới cho khách hàng.

Phân tích dữ liệu khách hàng và sở thích cá nhân cho phép sử dụng [Amazon Personalize để đưa ra](https://aws.amazon.com/personalize/) [các khuyến mãi](https://aws.amazon.com/blogs/machine-learning/implement-real-time-personalized-recommendations-using-amazon-personalize/) và đề xuất được cá nhân hóa.

Amazon Personalize có thể phân tích dữ liệu đơn hàng trong quá khứ để nâng cao trải nghiệm của người dùng thông qua các đề xuất được cá nhân hóa, chẳng hạn như thời gian giao hàng tối ưu, các mục menu ưa thích và [các chương trình khuyến mãi](https://docs.aws.amazon.com/personalize/latest/dg/promoting-items.html) phù hợp dựa trên từng kiểu đặt hàng.

## **Phần kết luận**

Bài đăng này cho thấy cách sử dụng các dịch vụ không máy chủ của AWS để xây dựng nền tảng cho quá trình xử lý đơn hàng của bạn mà không phải lo lắng về việc quản lý cơ sở hạ tầng cơ bản. Các dịch vụ không máy chủ bao gồm Amazon API Gateway, AWS Lambda, Amazon EventBridge, AWS Step Functions, AWS End User Messaging và Amazon Personalize.

Bài đăng này là phần giới thiệu ngắn gọn về kiến ​​trúc hướng sự kiện tập trung vào tích hợp hệ thống đặt hàng nội bộ với các đơn vị tổng hợp giao hàng và nền tảng đặt hàng của bên thứ ba. Điều này có thể giúp mở rộng cơ sở người dùng và là yếu tố chính trong sự phát triển của nhiều QSR. Việc làm cho trải nghiệm đặt hàng, mang đi và giao hàng hiệu quả hơn sẽ dẫn đến tăng trưởng doanh thu, giảm tình trạng bỏ đơn hàng cũng như tăng tỷ lệ giữ chân khách hàng thường xuyên và lòng trung thành với thương hiệu.

Để biết thêm tài nguyên học tập không có máy chủ, hãy truy cập [Serverless Land](https://serverlessland.com/). Để tìm thêm các mẫu, hãy truy cập trực tiếp vào [Serverless Patterns Collection](https://serverlessland.com/patterns).