



# Serverless Architecture 101

Ellipsis Tech Series 2020 Hackathon  
Pre-Hackathon Workshop

Glendon Thaiw Yong Neng  
3 September 2020



# Introduction



## Glendon Thaiw Yong Neng

**Solutions Architect Intern**, ASEAN Digital Customers & Region Expansion at AWS

Renaissance Engineering Programme, Nanyang Technological University



# Workshop Overview

- Serverless 101
- Case Study – Realtor.com
- Hands-on Project with AWS
- Q&A

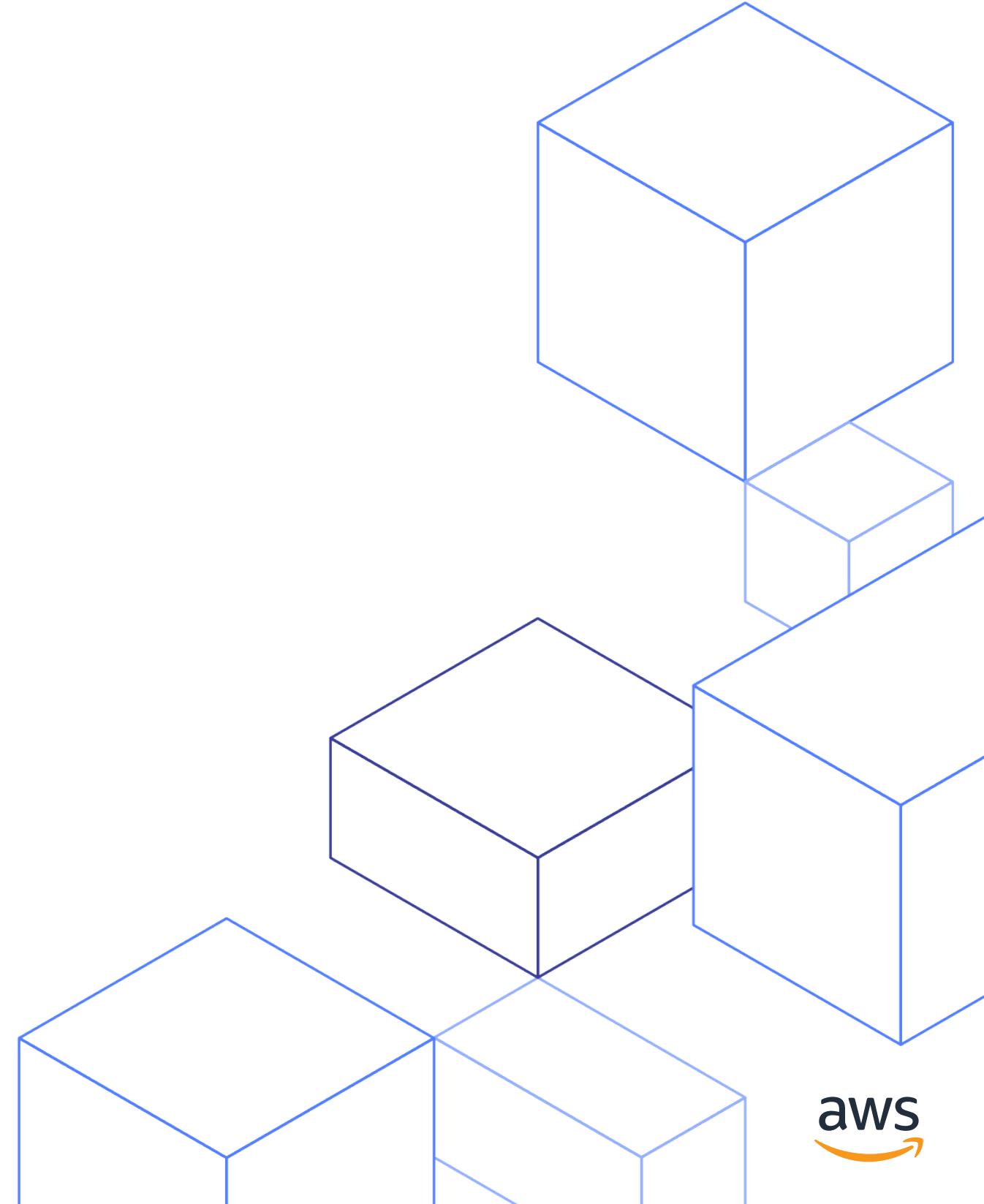
Resources (code snippets and workshop slide deck) available at:

<https://github.com/glendont/aws-serverless-workshop>

*\*\* Please direct your technical issues to Okkar Min through direct message*

# Serverless 101

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



# What is serverless?



 **Rackspace**

**Data Centers**

**In 2006,  
EC2 is introduced**



Amazon EC2

**Infrastructure  
As a service**

**In 2011,  
EB is introduced**



Elastic BeanStalk

**Platform  
As a service**

**In 2014,  
Lambda is introduced**



AWS Lambda

**Serverless  
*“Function-as-a-service”***

# What is serverless?



Data Centers

Coffee-machine-as-a-Service



Infrastructure  
As a service

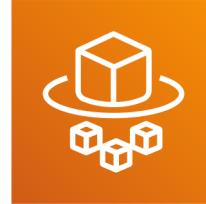
Platform  
As a service

Coffee-as-a-service



Serverless  
*"Function-as-a-service"*

# Why use serverless?



**No infrastructure provisioning,  
no server management**



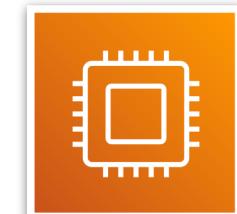
**Automatic, Flexible Scaling**

---

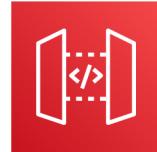
**Pay for Value**



**Highly available and secure**



# AWS Serverless Services



**Amazon API Gateway**



**Amazon Fargate**



**Amazon Lambda**



**Amazon EventBridge**



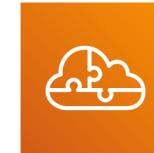
**Amazon Simple Notification Service (SNS)**



**Amazon Step Functions**



**Amazon Simple Queue Service (SQS)**

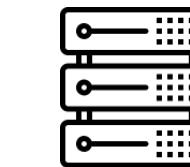
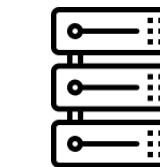
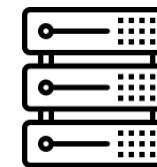


**Amazon Serverless Application Model (SAM)**

# Three-tier application architecture



**Web servers**  
Presentation

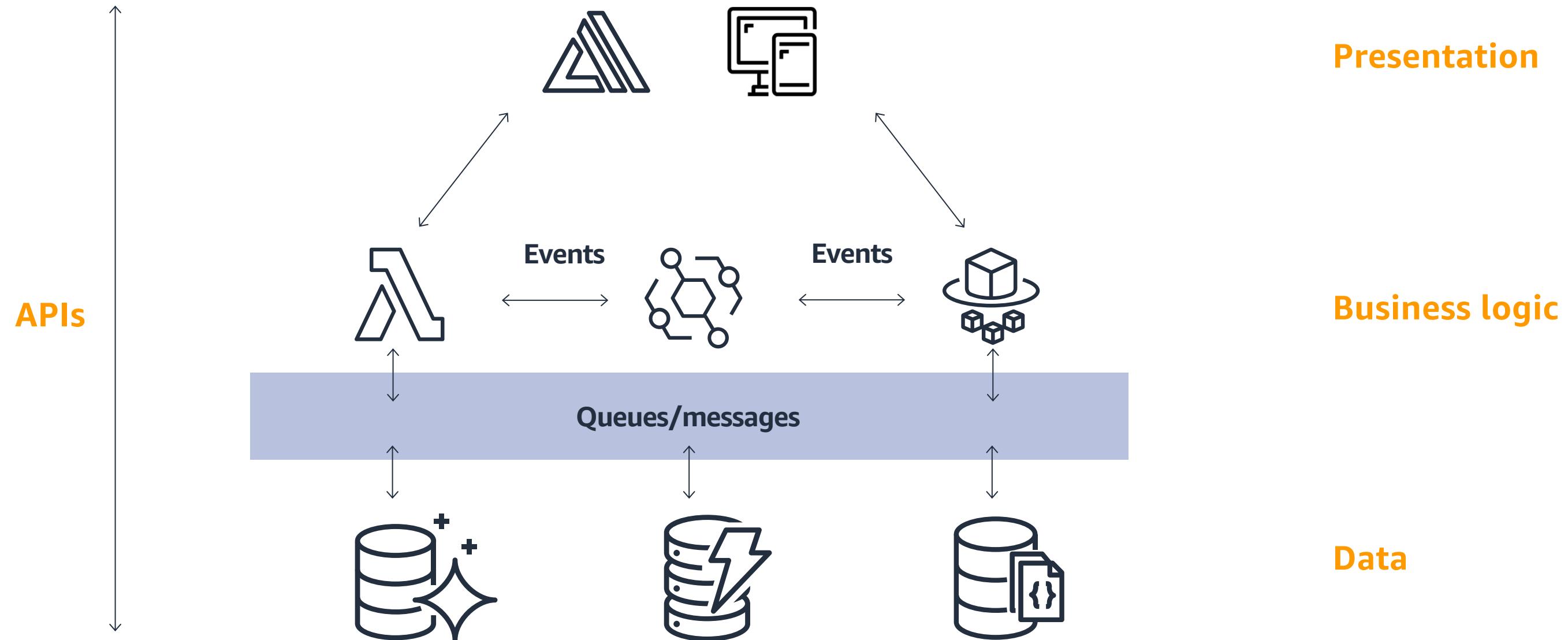


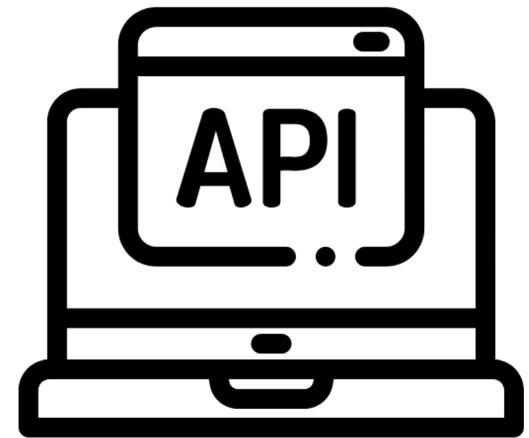
**Application  
servers**  
Business



**Database servers**  
Data layer

# A modern three-tier web application architecture



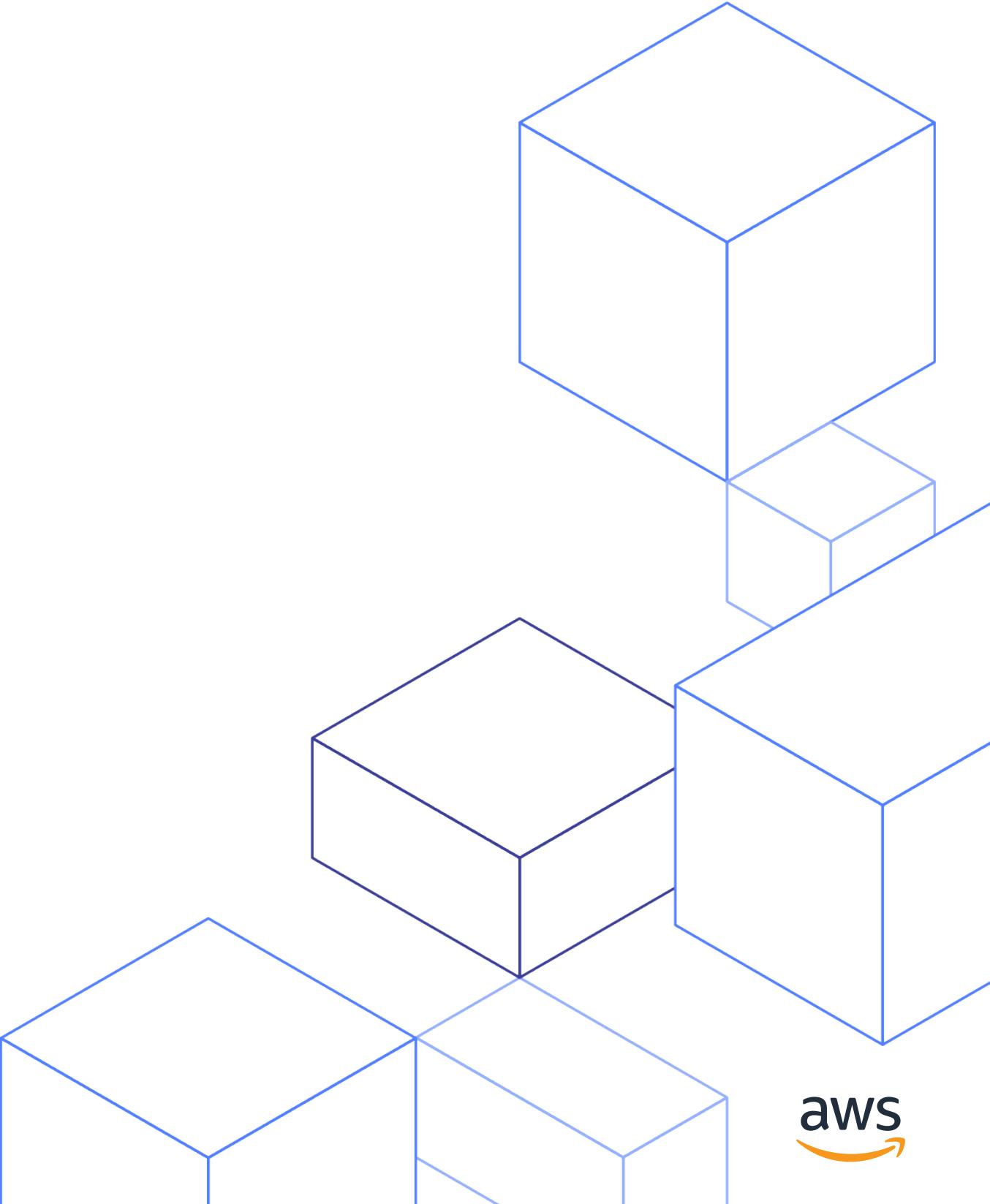


APIs are the front door of  
microservices

# Case Study

## Realtor.com

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

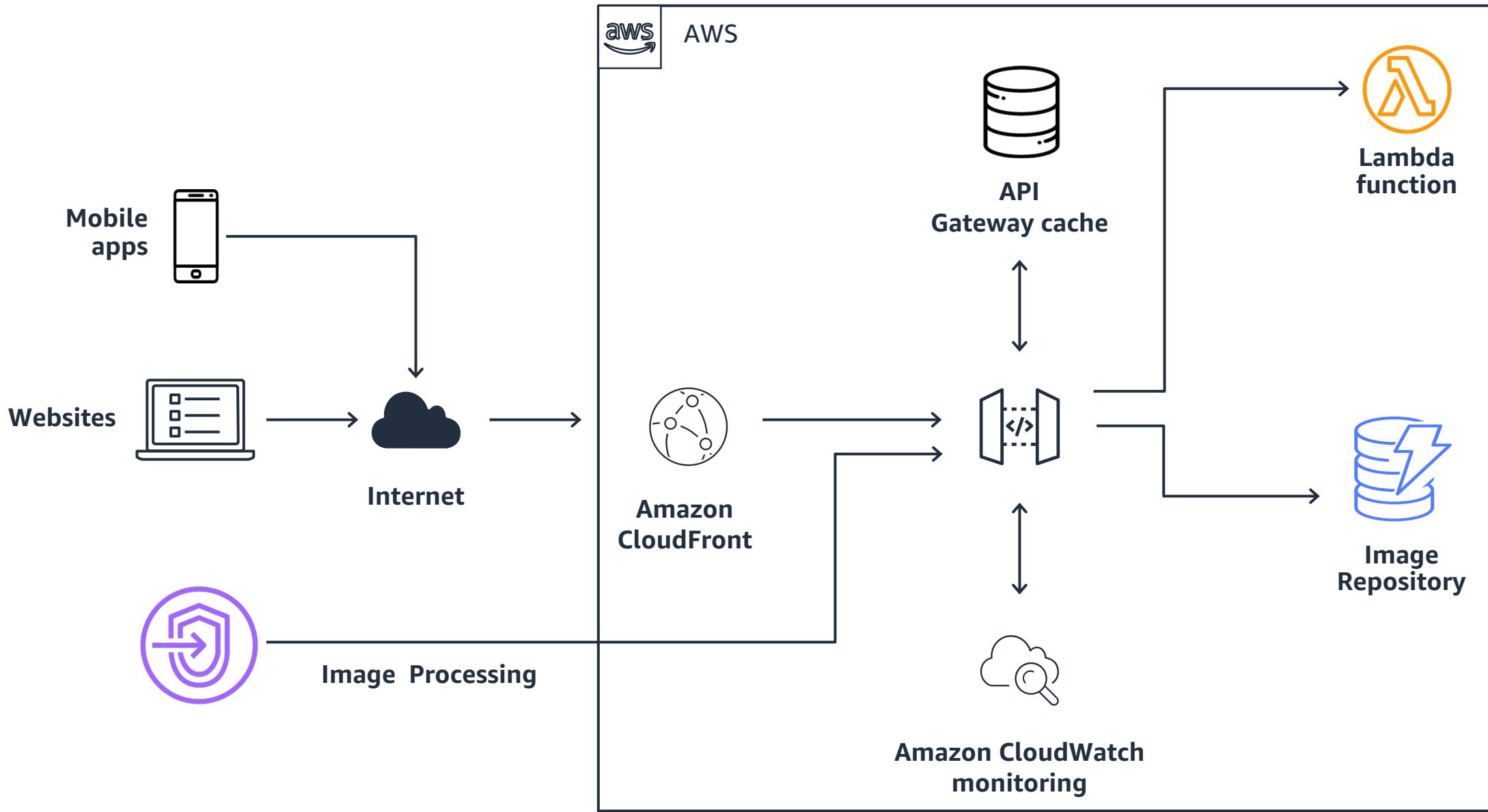


# Case Study: Realtor.com uses APIs between services

The screenshot shows the Realtor.com homepage. At the top, there's a navigation bar with links for Buy, Sell, Rent, Mortgage, Find Realtors®, My Home, and News & Insights. On the right side of the header are Log In, Sign up, and Advertise buttons. Below the header, a message reads "Help for planning your next steps post COVID-19. Visit site". The main banner features a large American flag and the text "Let's find a home that's perfect for you." Below the banner is a search bar with tabs for BUY (which is selected), RENT, PRE-APPROVAL, JUST SOLD, and HOME VALUE. The search bar has fields for "Address, School, City, Zip or Neighborhood" and a red "Search" button. The background of the page is a photograph of a house with a porch and a flag. Below the banner, there's a section titled "New listings in San Francisco, CA" with a link to "View All 433 New Listings". This section displays four new listing cards:

- Listing for Sale \$999,000** (NEW - 2 HOURS AGO)  
1 bed 2 bath  
2655 Bush St Apt 123, San Francisco, CA ...
- House for Sale \$1,700,000** (NEW - 2 HOURS AGO)  
3 bed 1 bath 2,220 sqft  
730 3rd Ave, San Francisco, CA 94118
- Condo \$1,198,000** (NEW - 3 HOURS AGO)  
2 bed 2 bath 985 sqft  
201 Sansome St Unit 602, San Francisco, CA 94104
- Condo \$1,249,000** (NEW - 3 HOURS AGO)  
3 bed 2 bath 1,600 sqft  
740 Great Hwy Apt 3, San Francisco, CA 94118

# Case Study: Realtor.com uses APIs between services

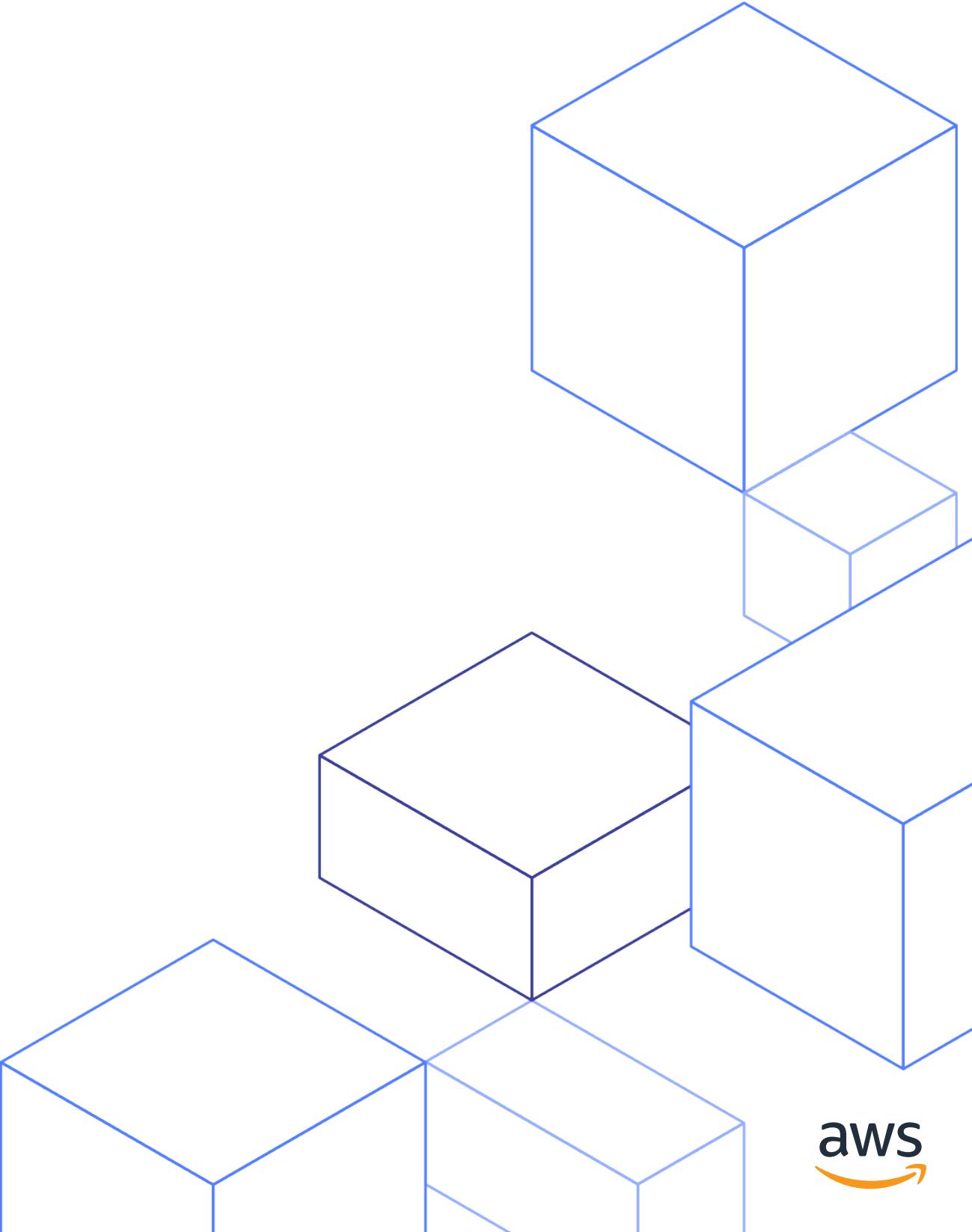


**"We process 800 million images per day through Amazon API Gateway..."**

Kuntal Shah,  
SVP Engineering,  
Realtor.com

**realtor.com®**

# Hands-on project with AWS



# Serverless Web Application

## WildRydes

- Ride Hailing Web Application

### Core Functionalities

- Account creation and user authentication
- Request a ride on the app by submitting a pick-up location
- Data persistence on a database to store records and transactions



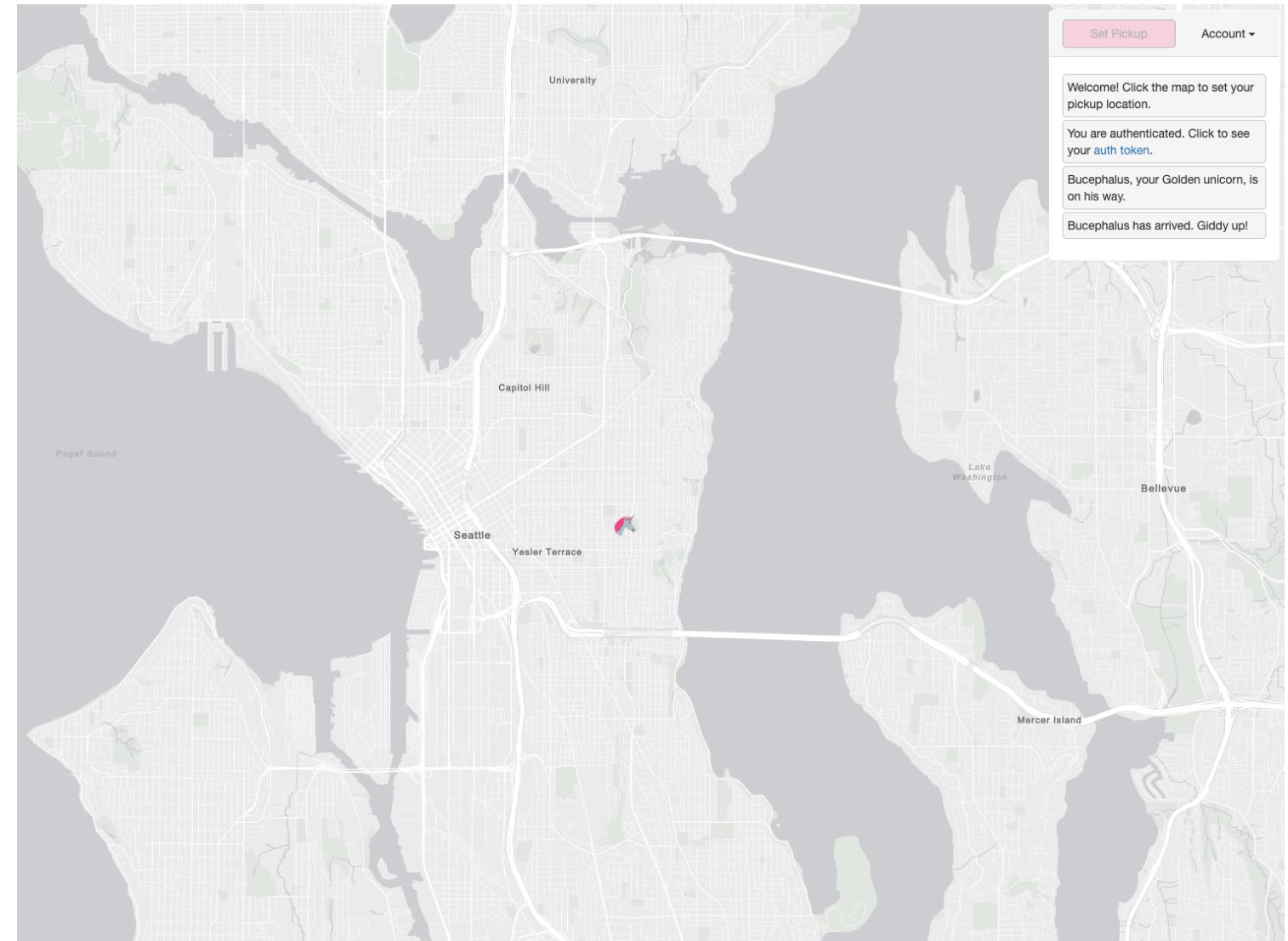
# Serverless Web Application

## WildRydes

- Ride Hailing Web Application

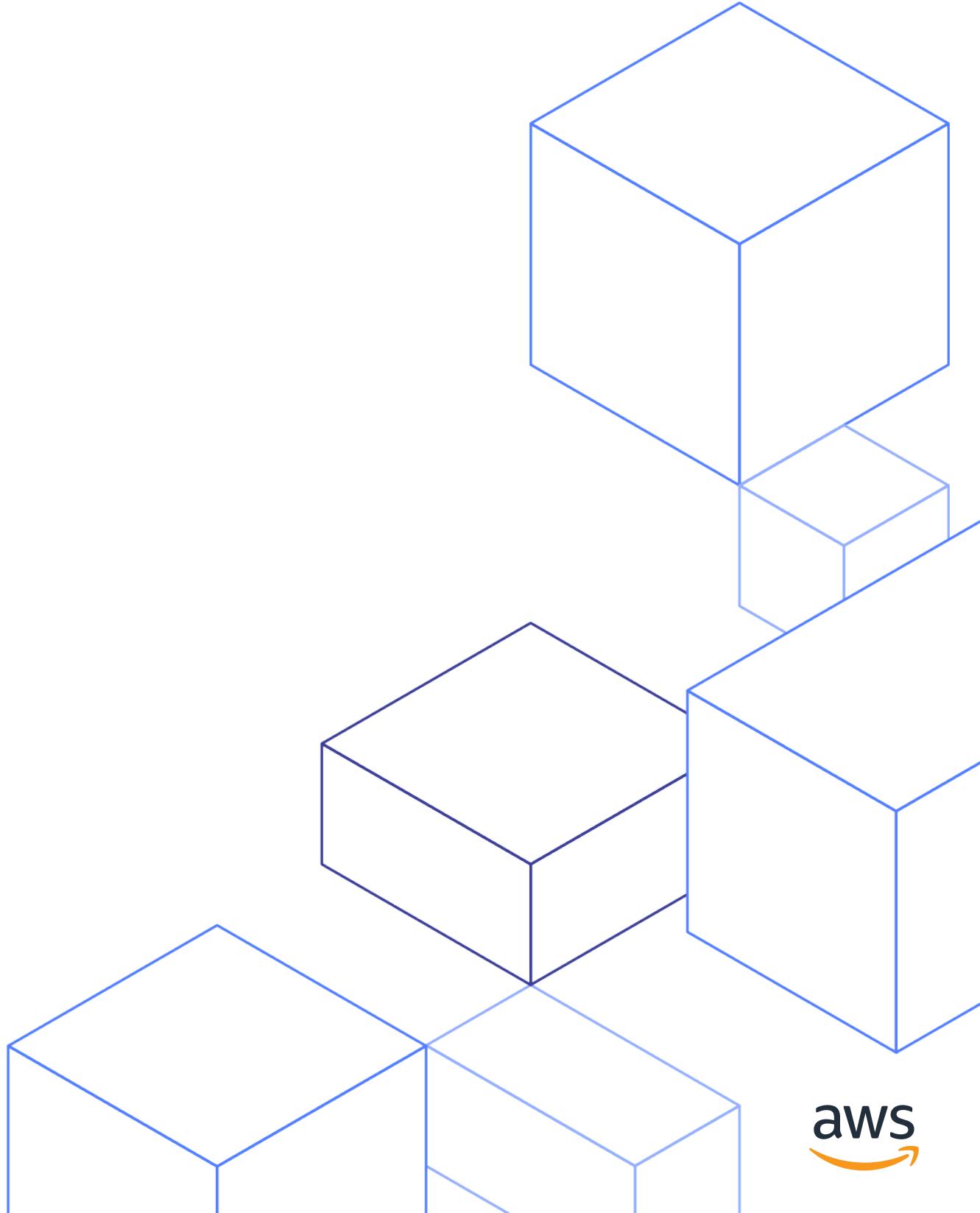
### Core Functionalities

- Account creation and user authentication
- Request a ride on the app by submitting a pick-up location
- Data persistence on a database to store records and transactions

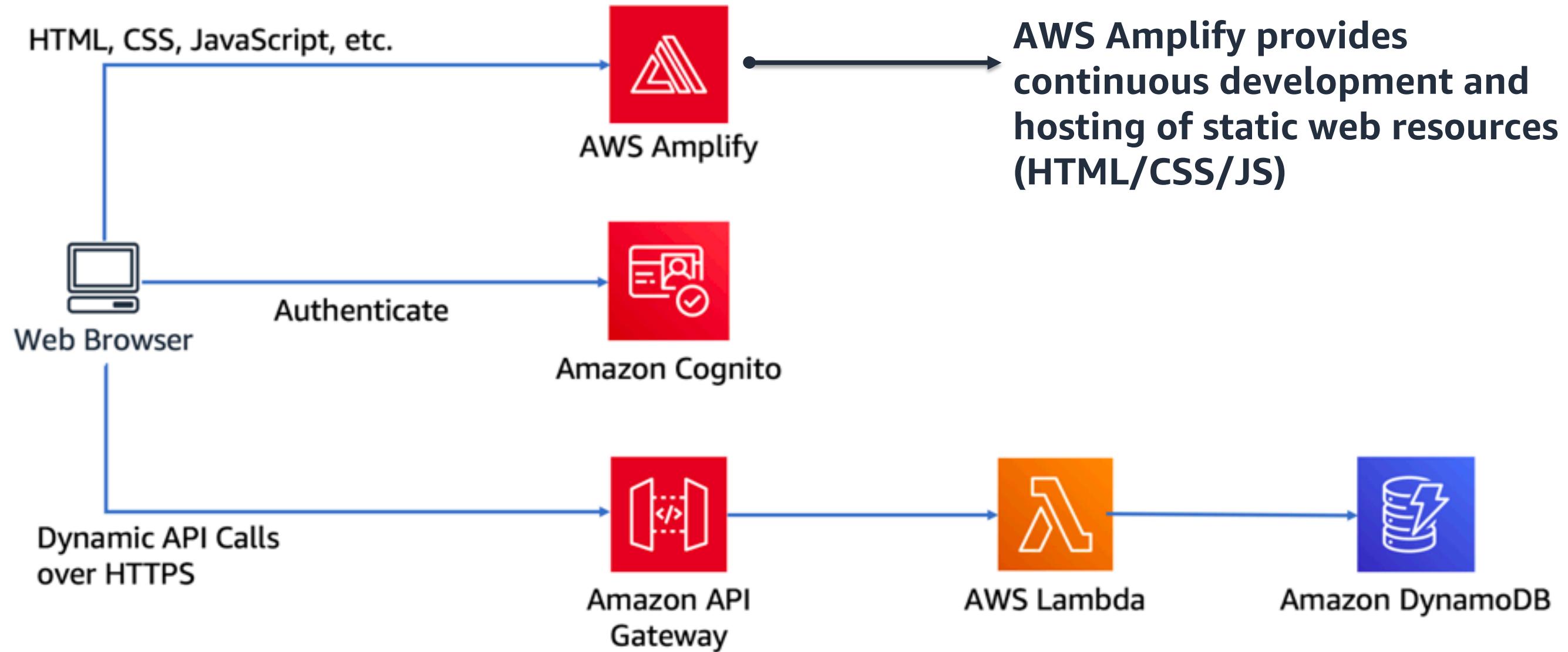


# App Demo

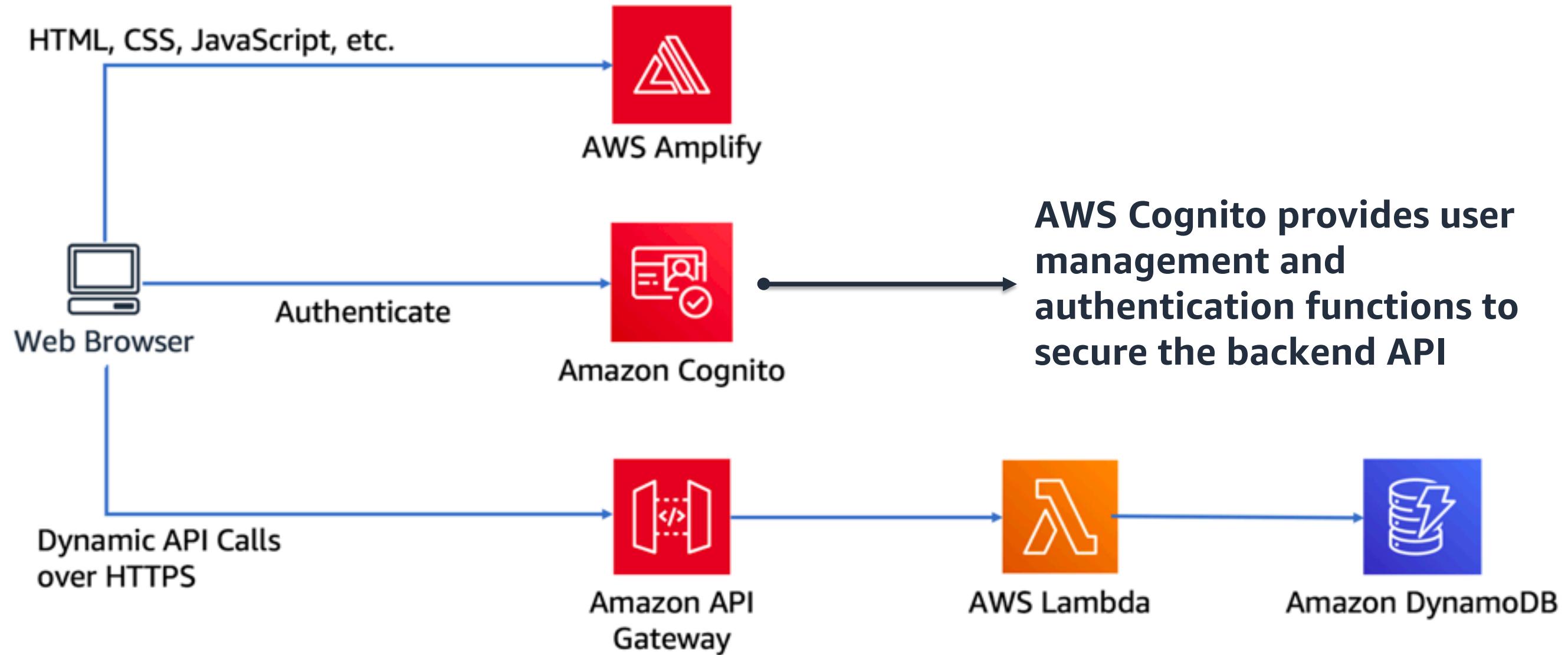
© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



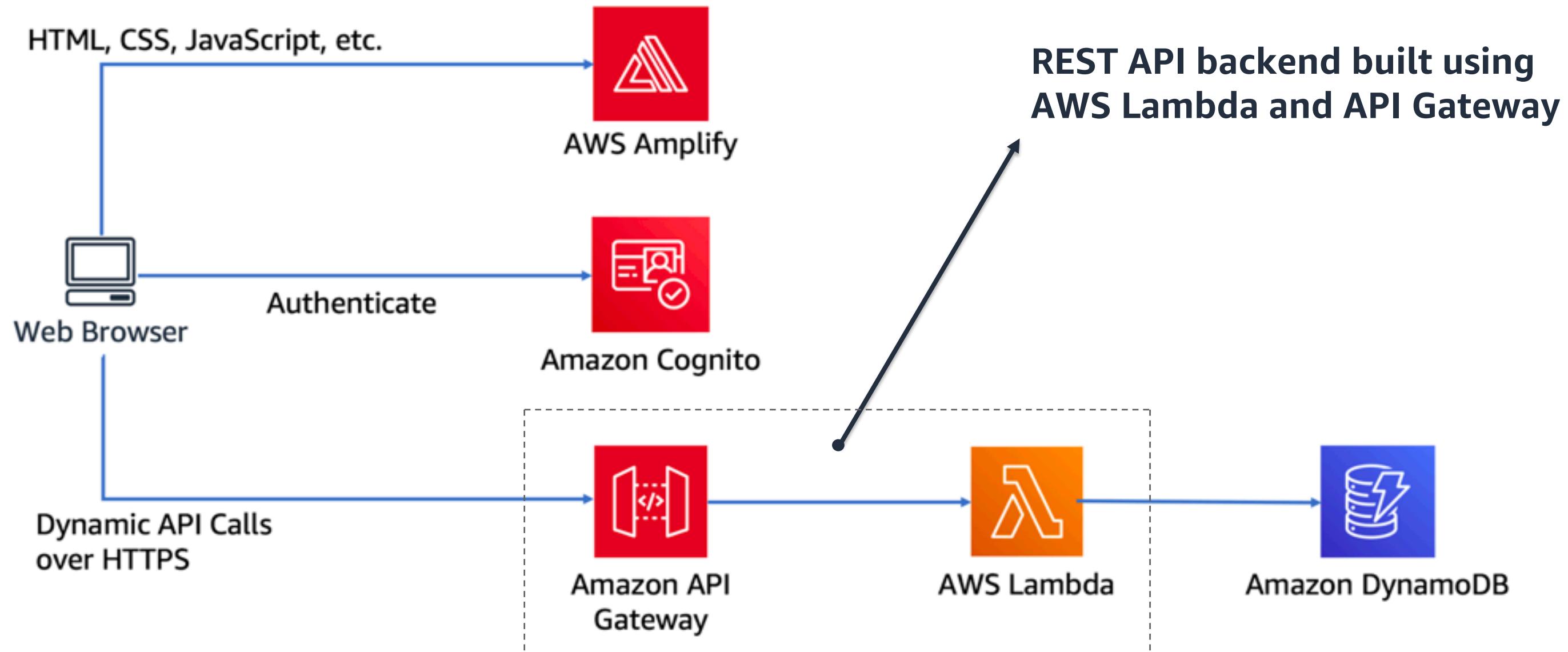
# WildRydes Architecture Diagram



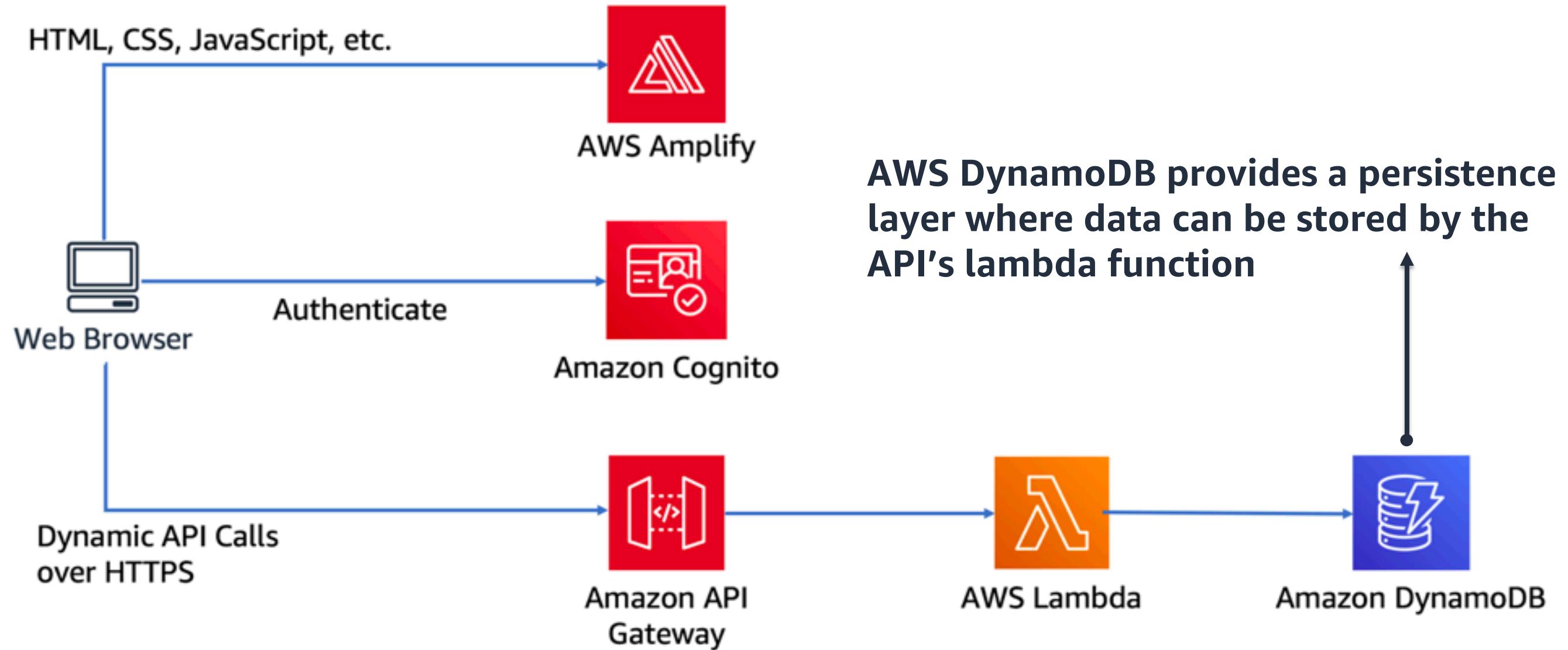
# WildRydes Architecture Diagram



# WildRydes Architecture Diagram

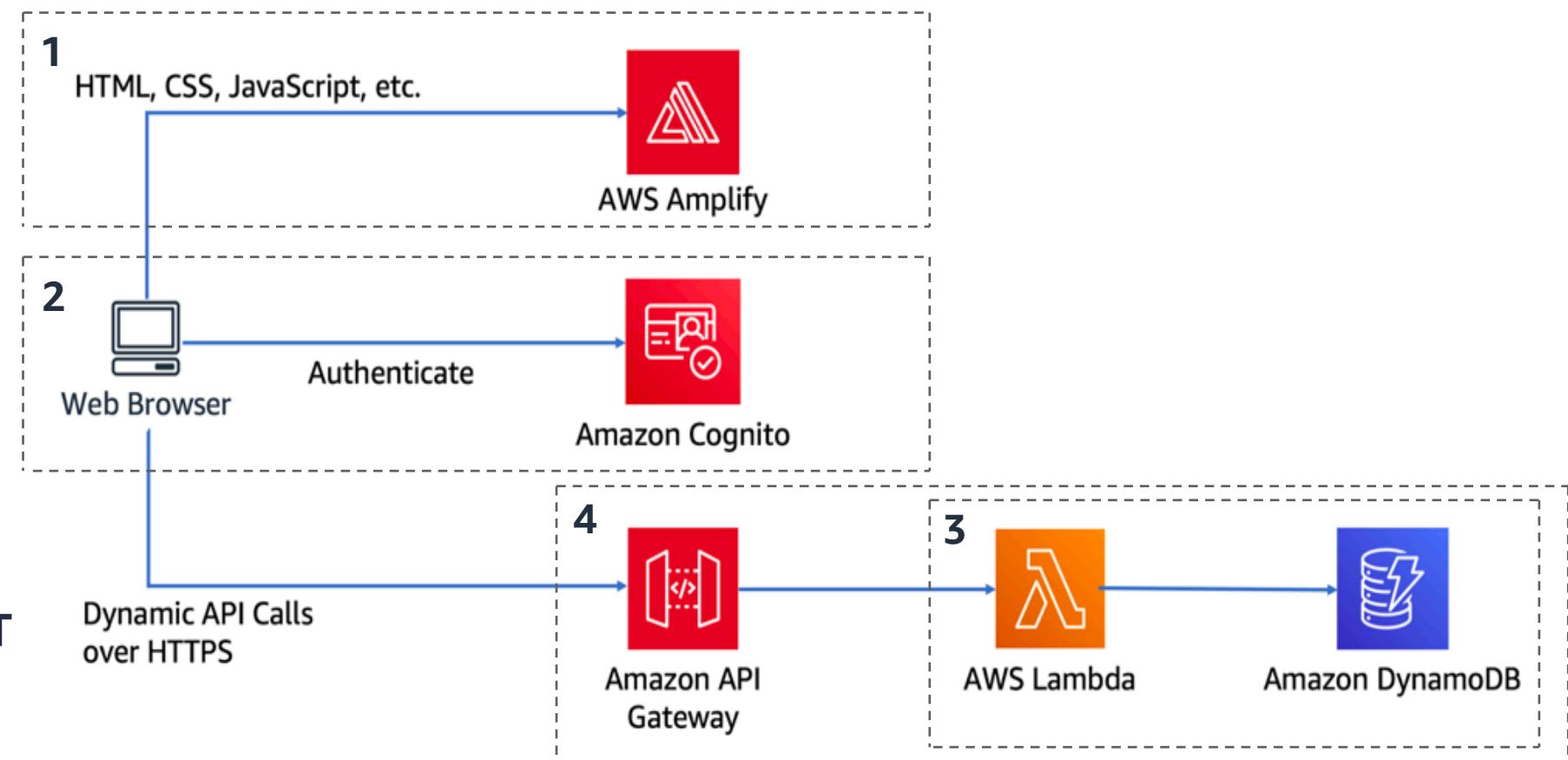


# WildRydes Architecture Diagram

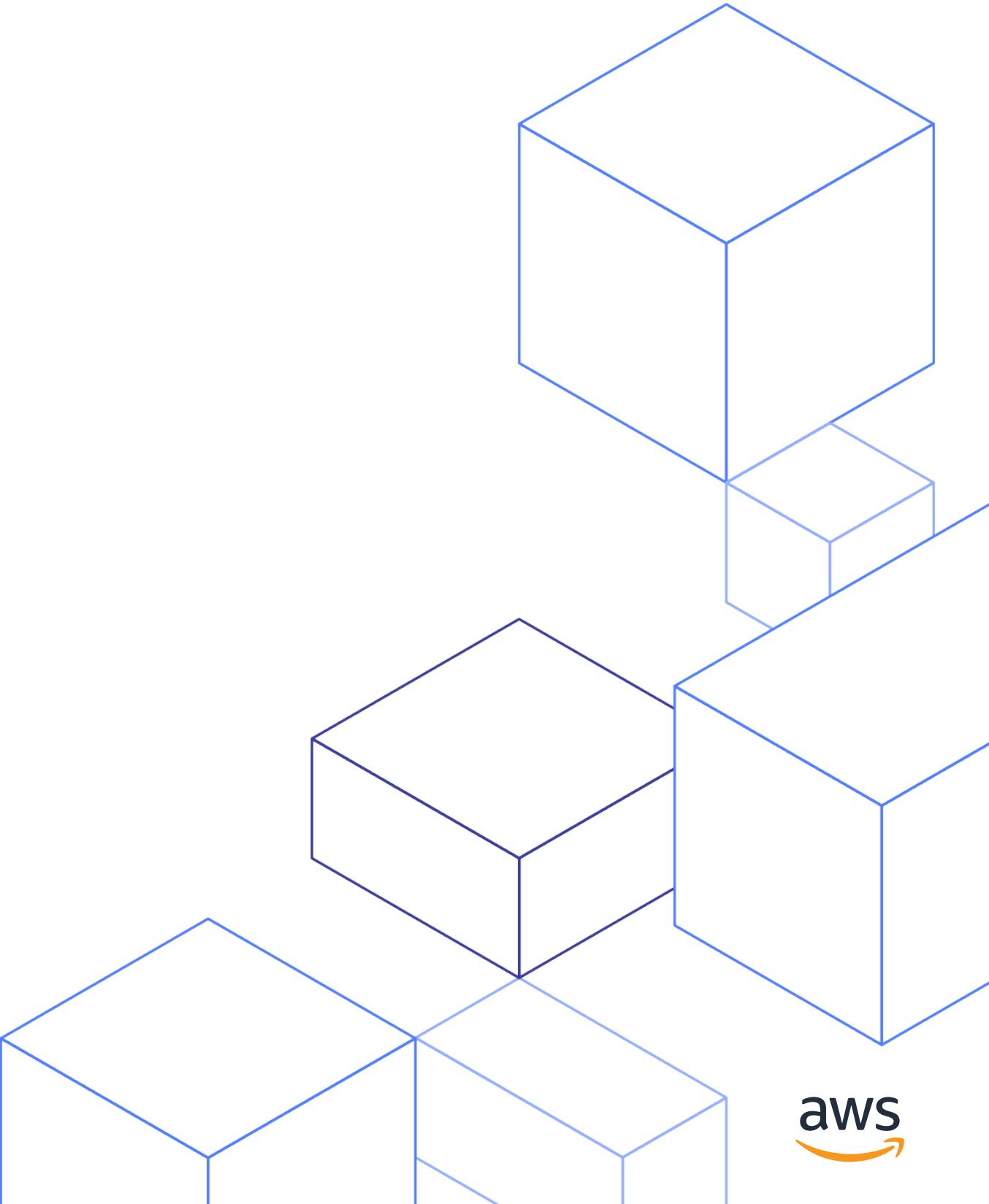


# Built Sequence

- 1) Host front-end static site on AWS Amplify**
- 2) Set up user authentication with AWS Cognito**
- 3) Build a serverless backend with AWS Lambda and DynamoDB**
- 4) Deploy AWS Lambda function as a REST API with API Gateway**



# Let's get buildin'!



# Q&A

Glendon Thaiw Yong Neng

Resources (code snippets and workshop slide deck) available at:  
<https://github.com/glendont/aws-serverless-workshop>

[https://docs.google.com/forms/d/e/1FAIpQLSdpRfVhLYOnzggHASXlbOjkZoYi4t72qrACiPvwuSBxBVHOiw/viewform?usp=sf\\_link](https://docs.google.com/forms/d/e/1FAIpQLSdpRfVhLYOnzggHASXlbOjkZoYi4t72qrACiPvwuSBxBVHOiw/viewform?usp=sf_link)