Leveraging serverless in full-stack development

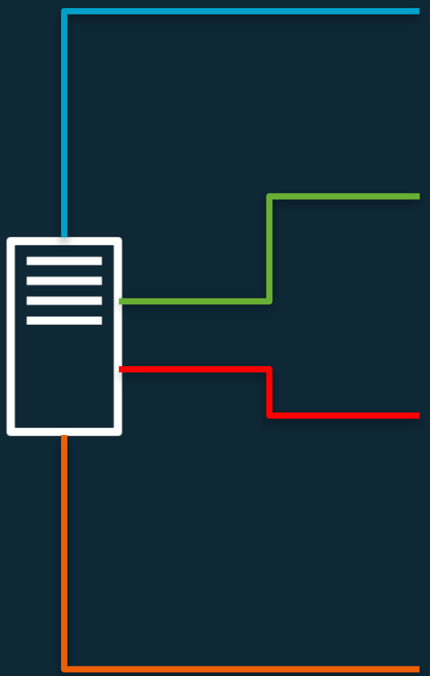
Eric Johnson – Sr. Developer Advocate – Serverless, AWS
@edjgeek

Who am I?

- @edjgeek
- Husband and father of 5
- Senior Developer Advocate – Serverless, AWS
- Serverless tooling and automation nerd
- Solutions & software architect (> 25 years)
- Music lover
- Pizza lover (without pineapple)
- Pusher of #ServerlessForEveryone

Application architecture journey

The single server architecture



Many application architectures start here.

Proxy/Firewall: NGINX, Apache, F5, Zeus

Server: IIS, Apache, Tomcat

Code: Node, .Net, Ruby, Java, Go, Python

Database: SQL, MySQL, PostgreSQL, Mongo, Cassandra

The tiered architecture

We then break applications into tiers based on functionality.

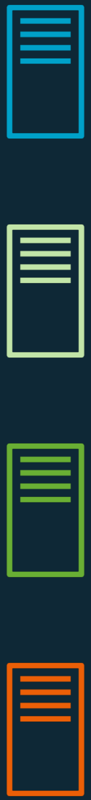
Proxy/Firewall

Server

Database

The tiered architecture

Even separating the client and backend.


Proxy/Firewall

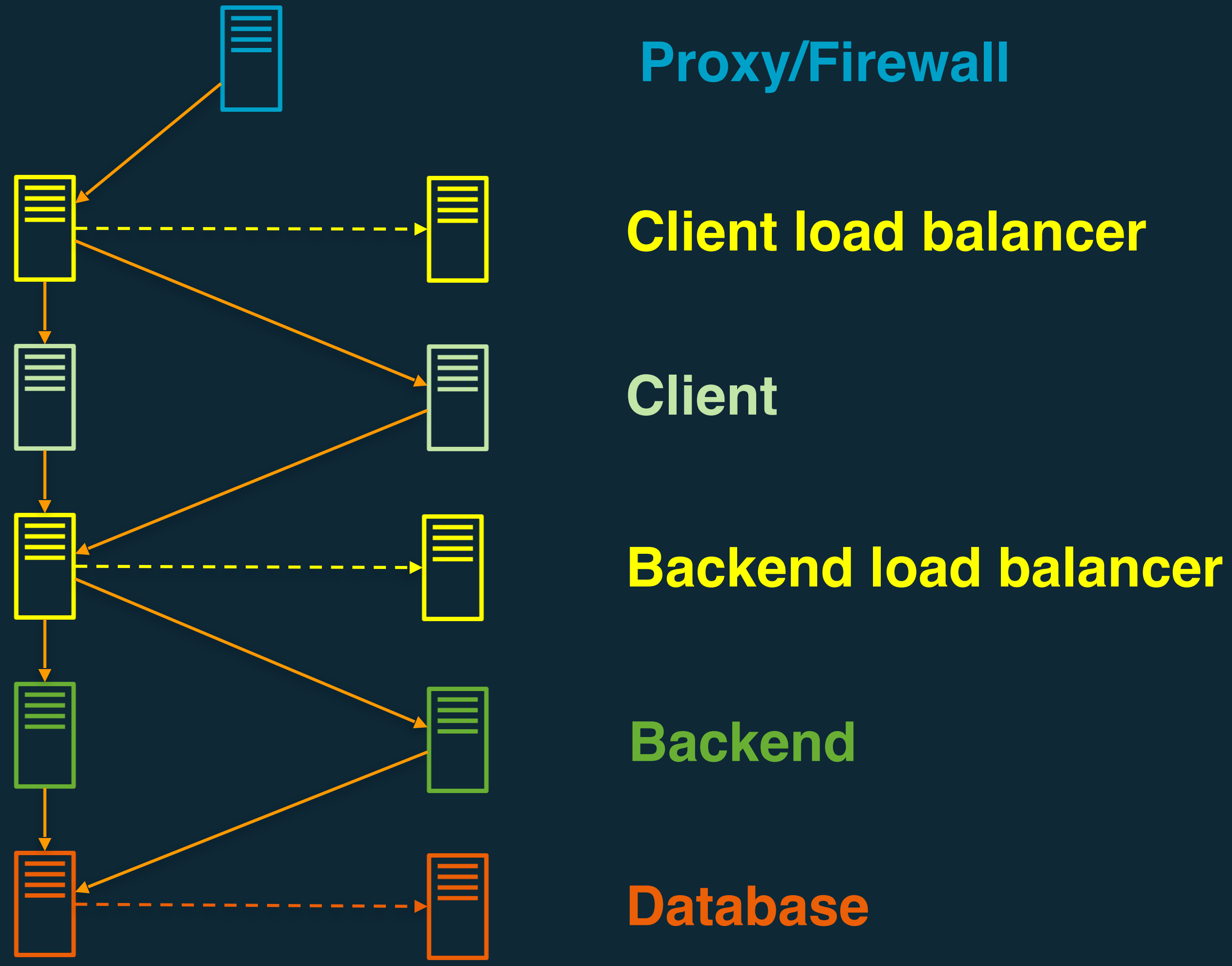
Client

Backend

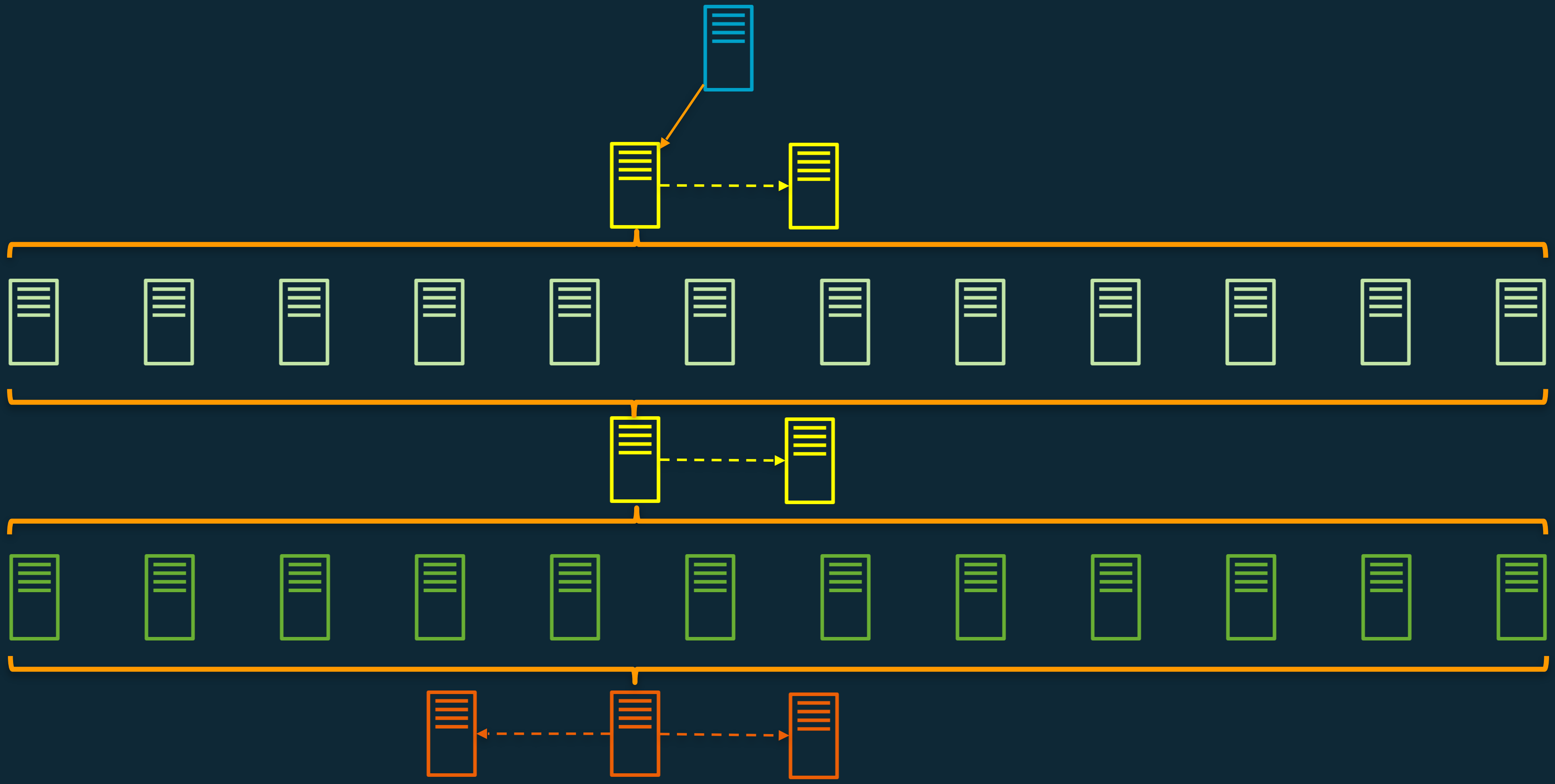
Database

The redundant tiered architecture

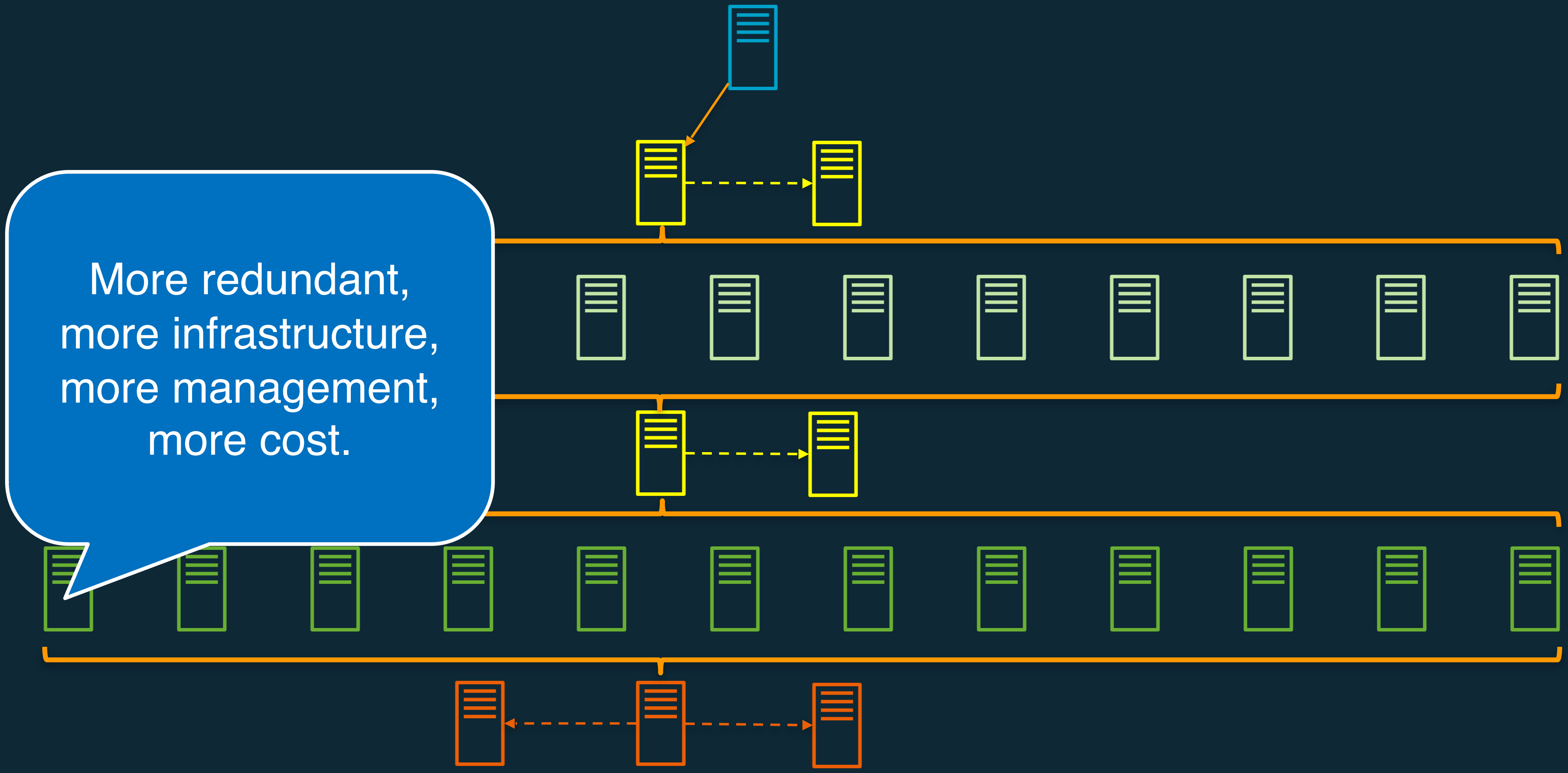
As applications scale, we build in redundancy.



The “more” redundant tiered architecture

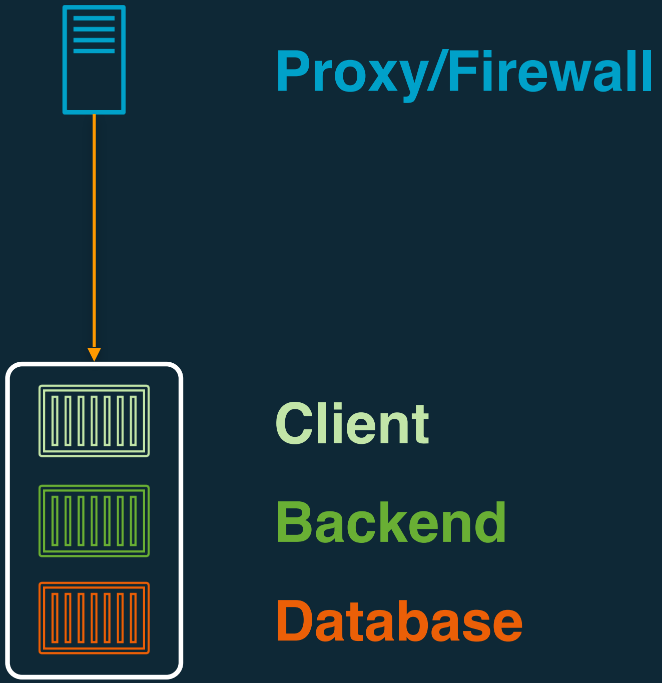


The “more” redundant tiered architecture

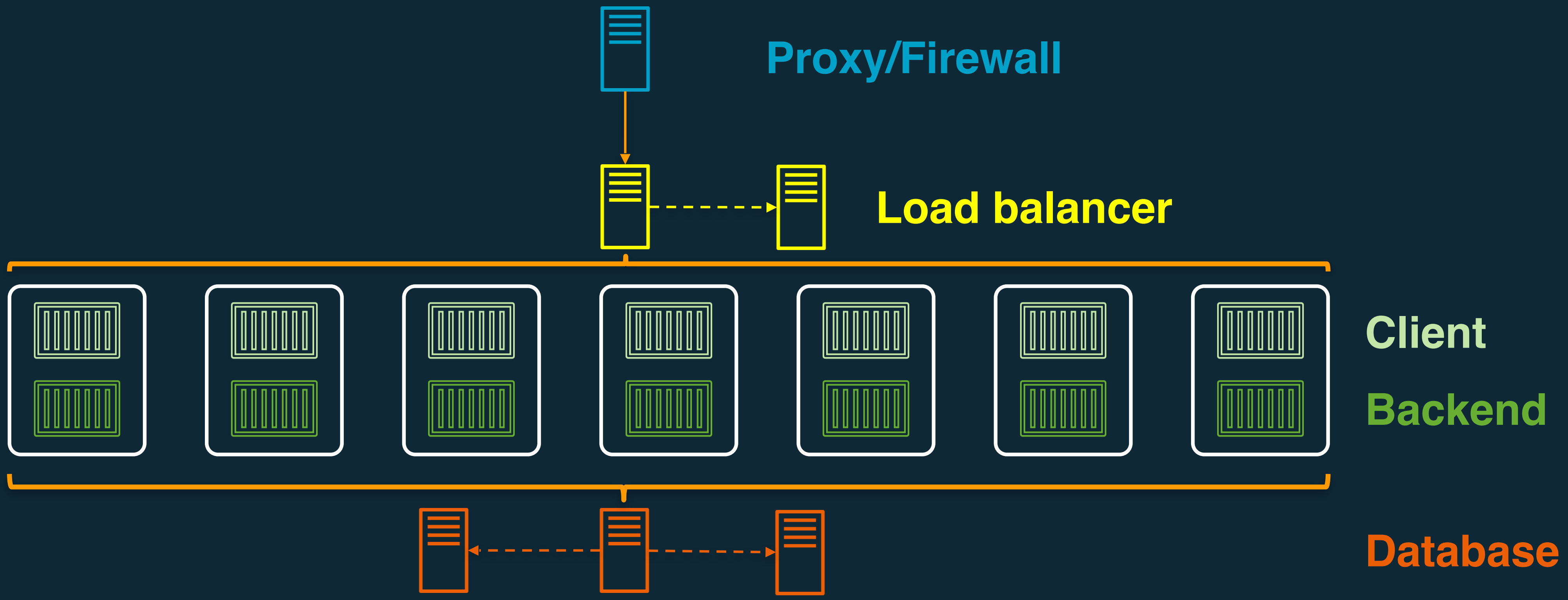

More redundant, more infrastructure, more management, more cost.

The containers architecture

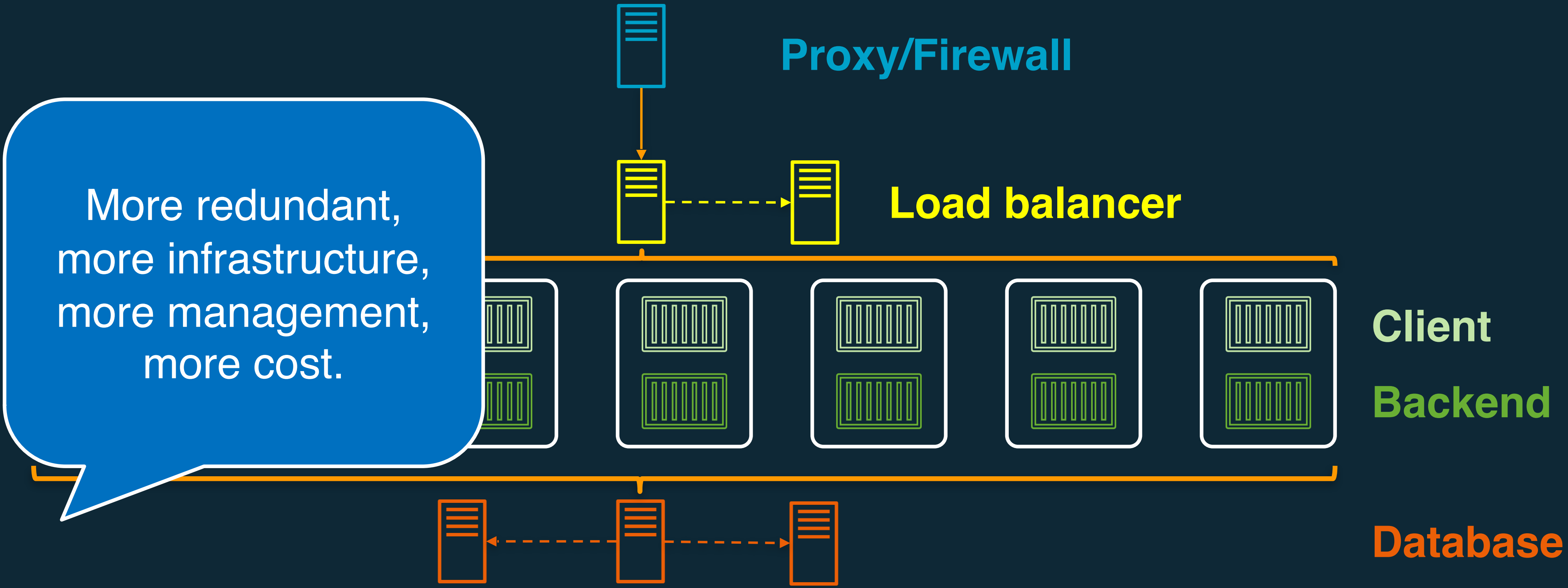
But what about containers?



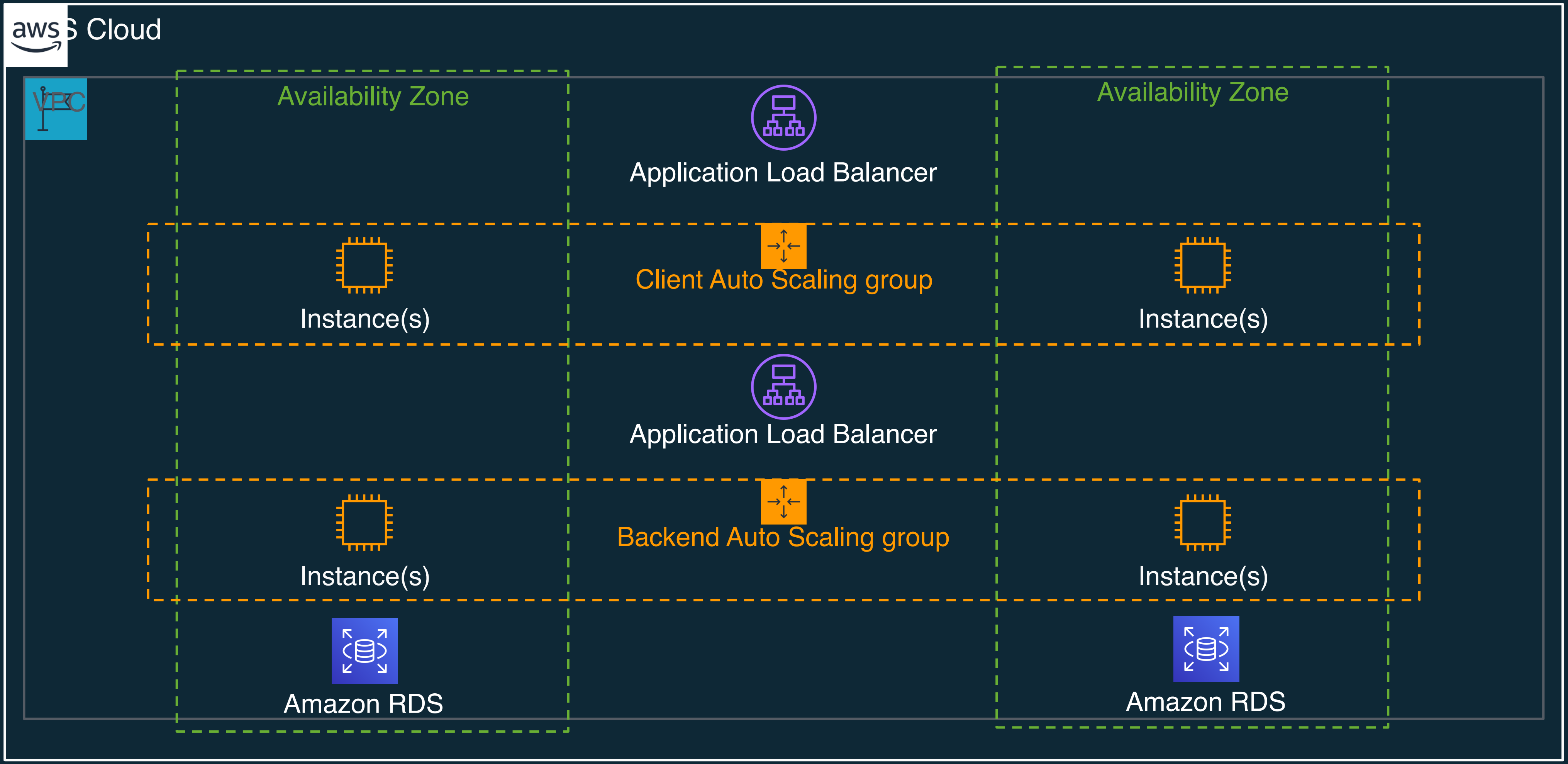
The "more" redundant containers architecture



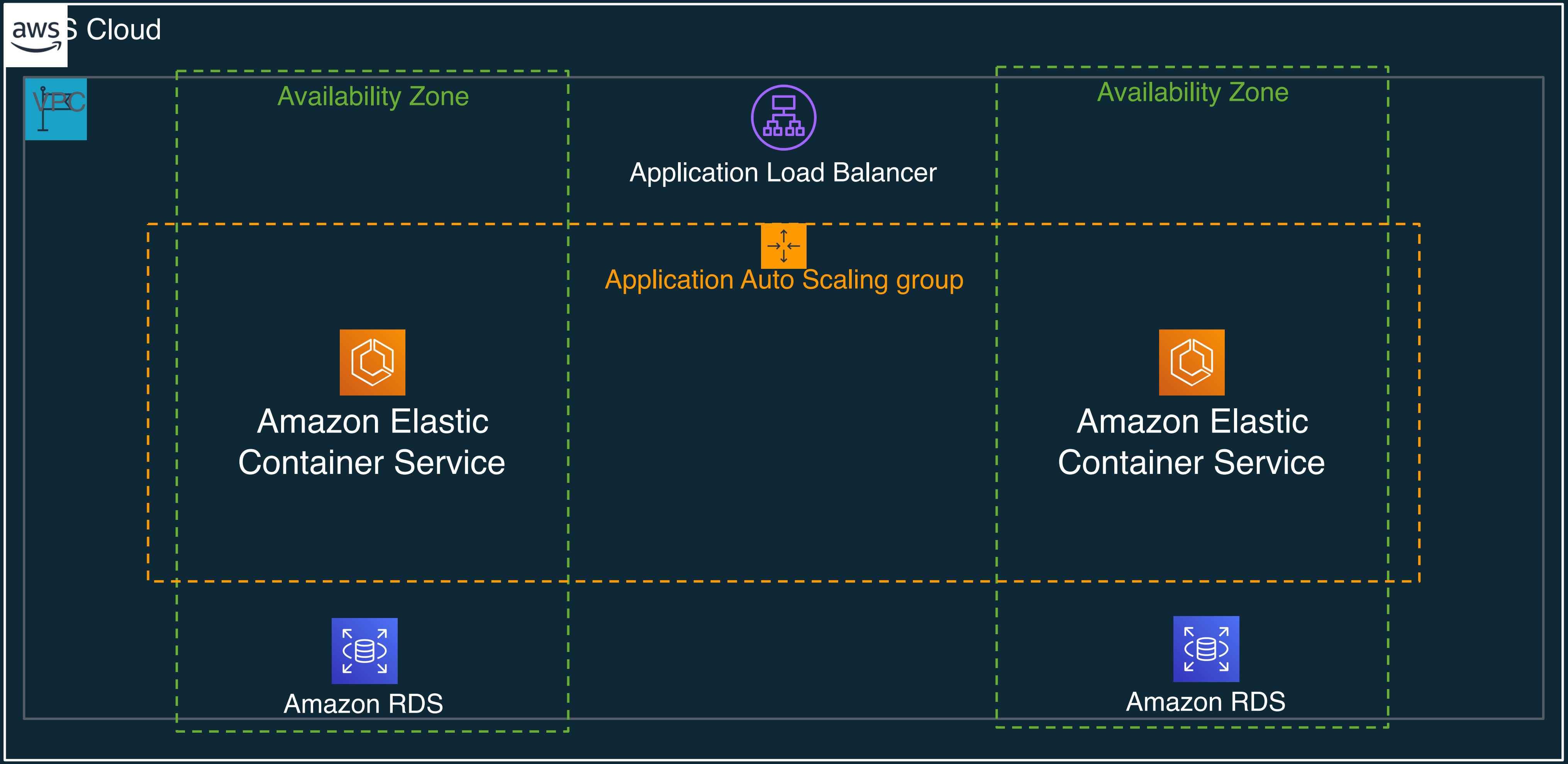
The "more" redundant containers architecture


More redundant, more infrastructure, more management, more cost.
Proxy/Firewall
Load balancer
Client
Backend
Database

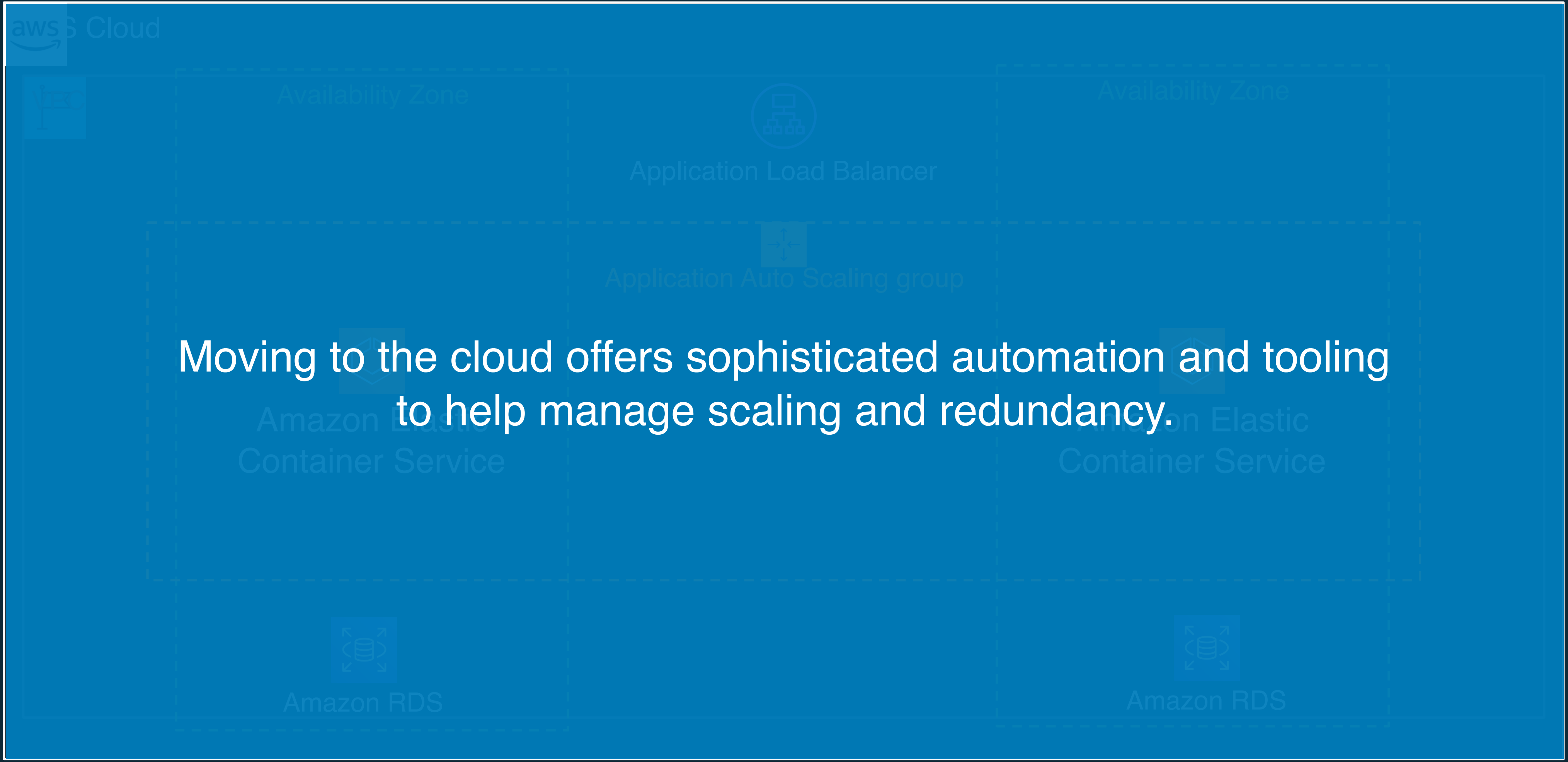
Moving to the cloud



Moving containers to the cloud


aws AWS Cloud
VPC
Availability Zone
Application Load Balancer
Availability Zone
Application Auto Scaling group
Amazon Elastic Container Service
Amazon Elastic Container Service
Amazon RDS
Amazon RDS

Moving containers to the cloud


AWS Cloud
Availability Zone
Application Load Balancer
Availability Zone
Application Auto Scaling group
Amazon Elastic Container Service
Amazon Elastic Container Service
Amazon RDS
Amazon RDS

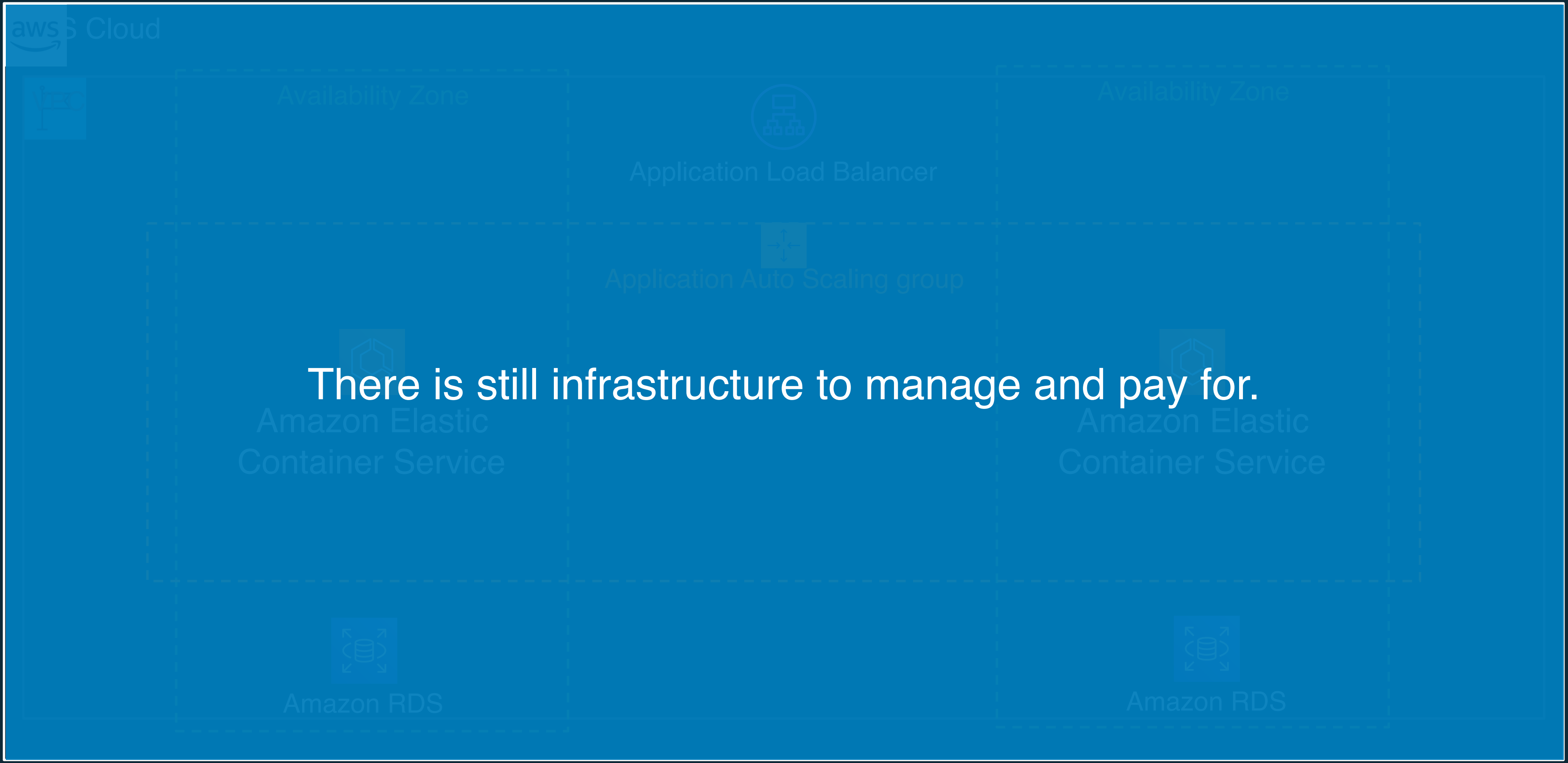
Moving to the cloud offers sophisticated automation and tooling to help manage scaling and redundancy.

Moving containers to the cloud

AWS Cloud
Availability Zone
Application Load Balancer
Availability Zone
Application Auto Scaling group
Amazon Elastic Container Service
Amazon Elastic Container Service
Amazon RDS
Amazon RDS

But…

Moving containers to the cloud


AWS Cloud
Availability Zone
Availability Zone
Application Load Balancer
Application Auto Scaling group
Amazon Elastic Container Service
Amazon Elastic Container Service
Amazon RDS
Amazon RDS

There is still infrastructure to manage and pay for.

Hello serverless!

What is serverless?

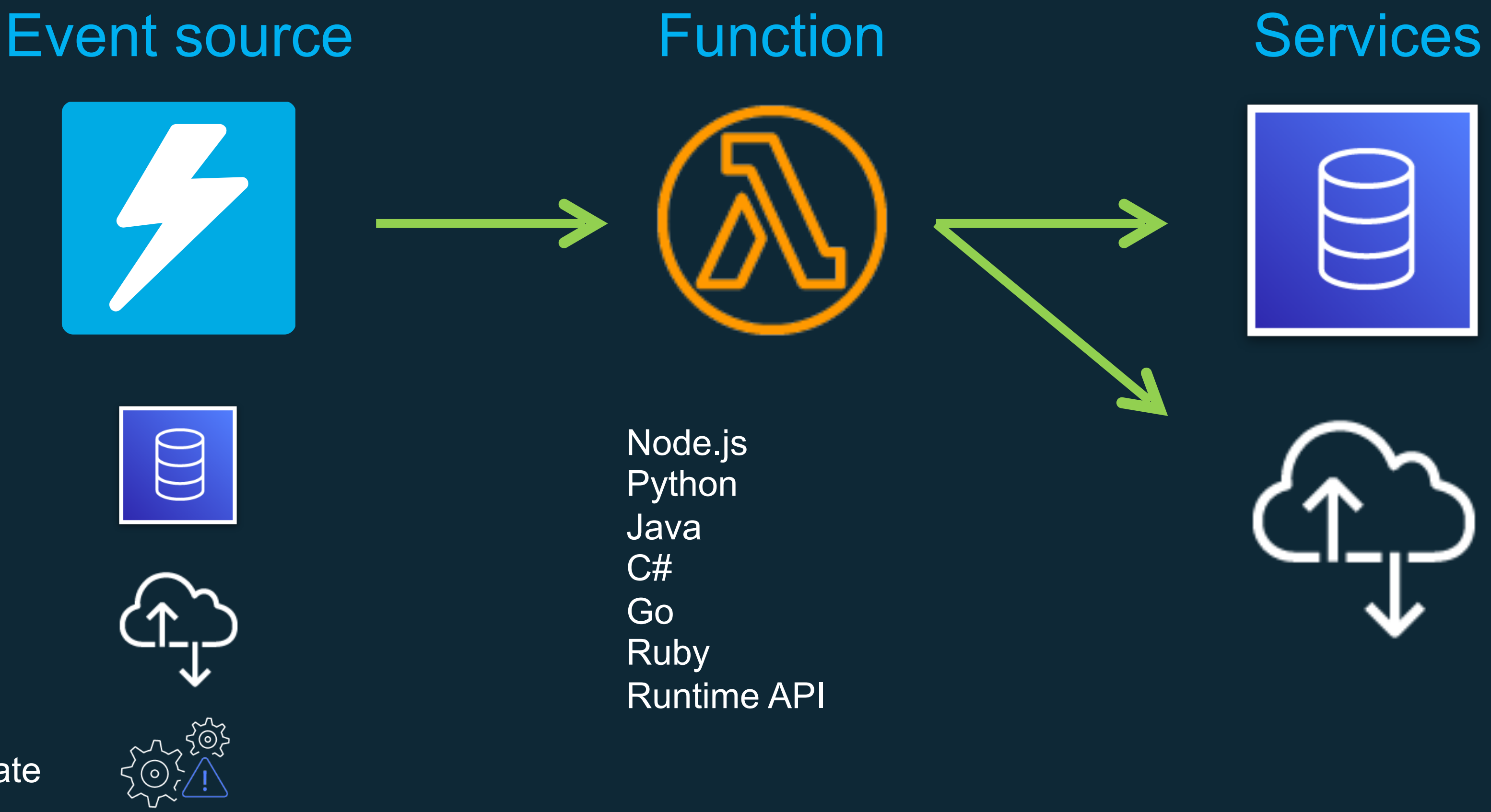
No infrastructure provisioning, no management

Automatic scaling

Pay for value

Highly available and secure

Serverless applications



Common AWS Lambda use cases

Web Apps

- Static websites
- Complex web apps
- Packages for Flask and Express

Backends

- Apps & services
- Mobile
- IoT

Data Processing

- Real time
- MapReduce
- Batch

Chatbots

- Powering chatbot logic

Amazon Alexa

- Powering voice-enabled apps
- Alexa Skills Kit

IT Automation

- Policy engines
- Extending AWS services
- Infrastructure management

Common AWS Lambda use cases

Web Apps

- Static websites
- Complex web apps
- Packages for Flask and Express

Backends

- Apps & services
- Mobile
- IoT

Data Processing

- Real time
- MapReduce
- Batch

xa

ps

Kit

Skills

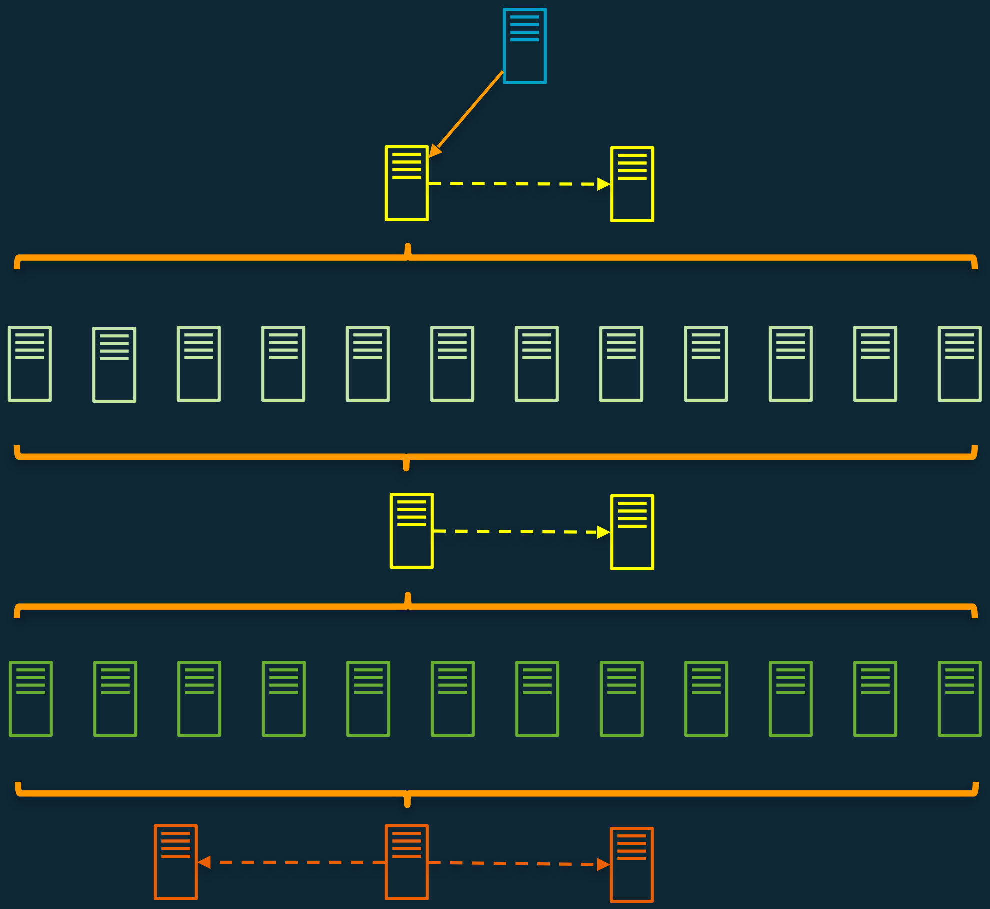
IT Automation

- Policy engines
- Extending AWS services
- Infrastructure management

I will be spending most of the time in the Web Apps and Backend area.

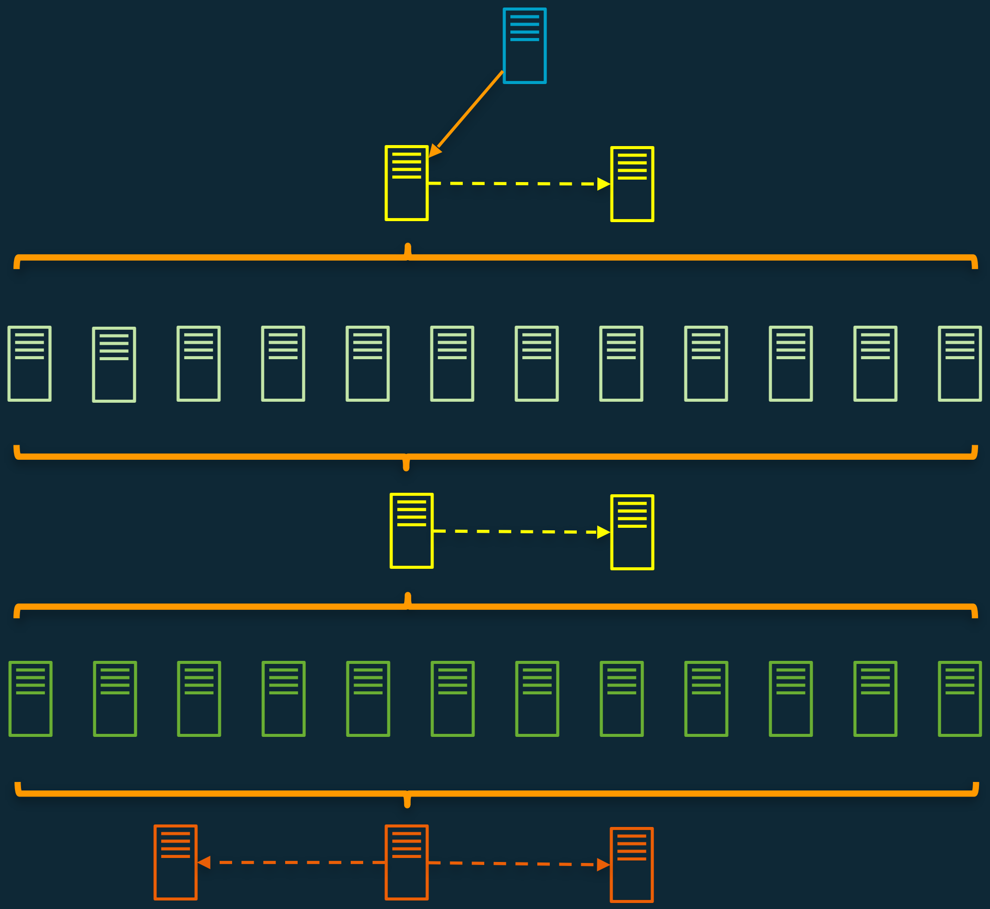
Serverless in full-stack development?

Moving to serverless – client



Assumption: Client is separate from backend.

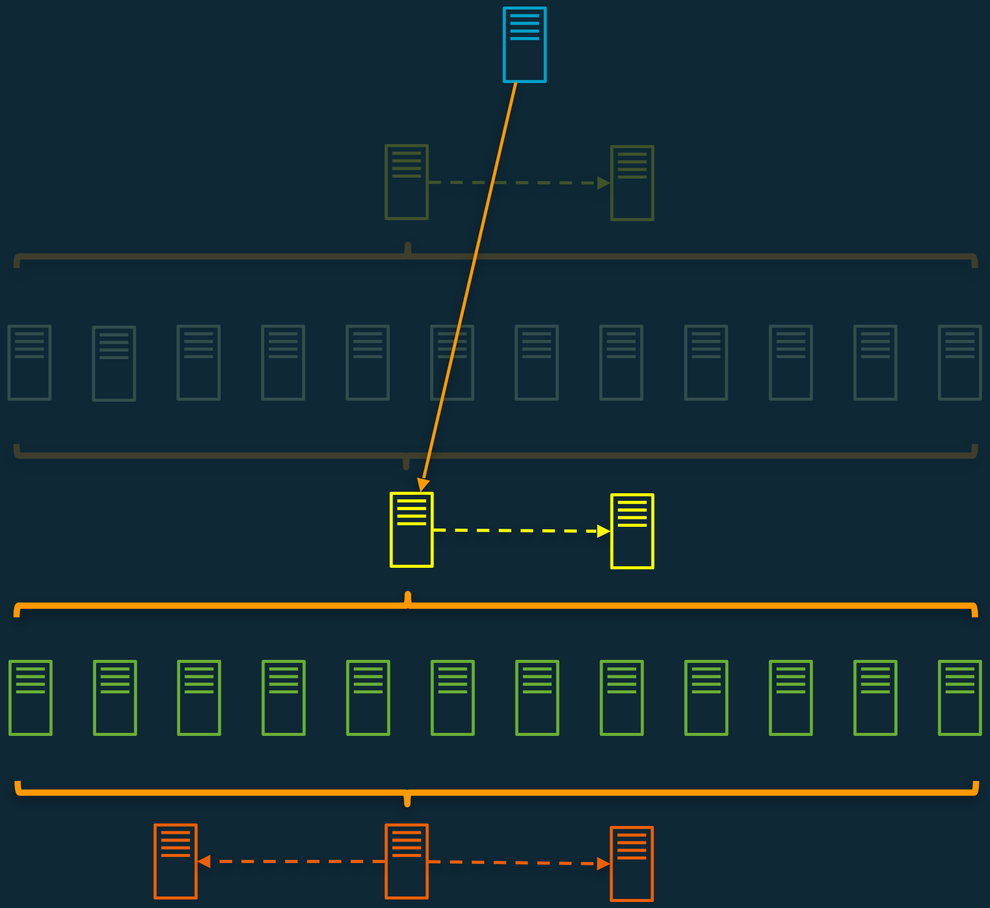
Moving to serverless – client



Assumption: Client is separate from backend.

If not, don't worry, I'll cover that too.

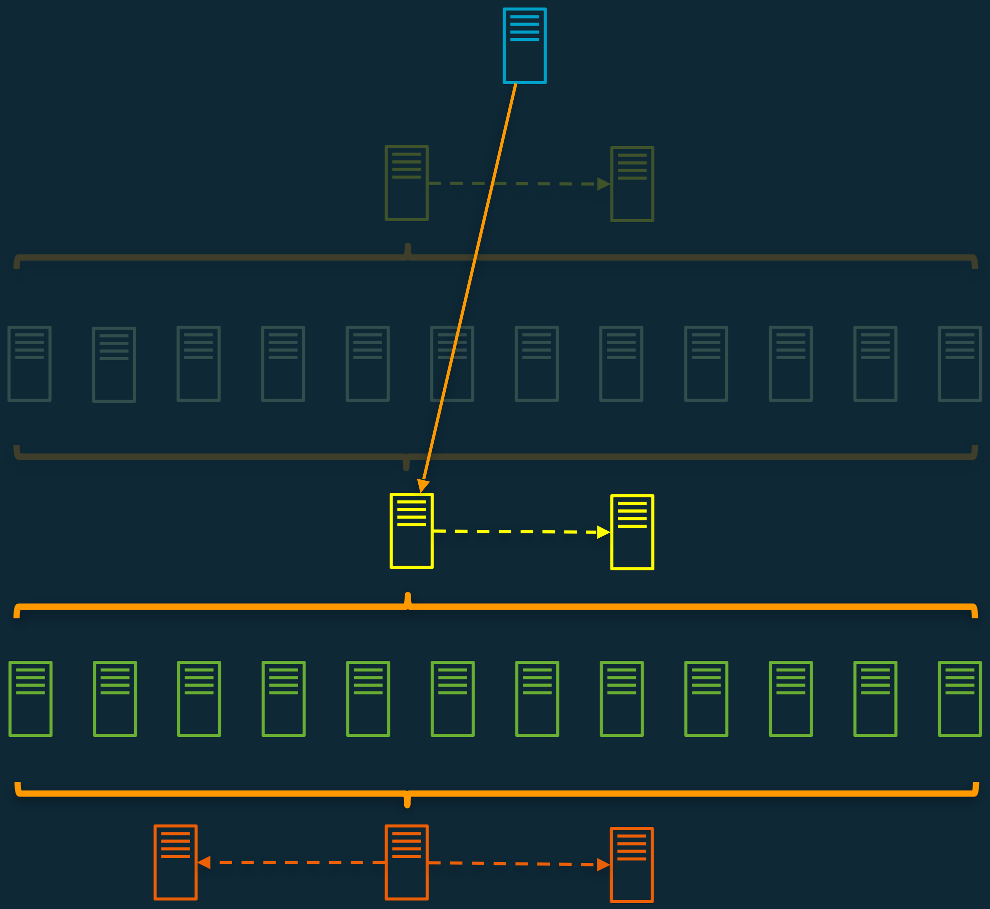
Moving to serverless – client



Amazon Simple Storage Service

- Hosting for http and https
- Object versioning
- Bucket policy, ACL, and IAM security controls.
- 99.999999999% (11 9s) data durability
- Custom domains
- Domain redirects

Moving to serverless – client

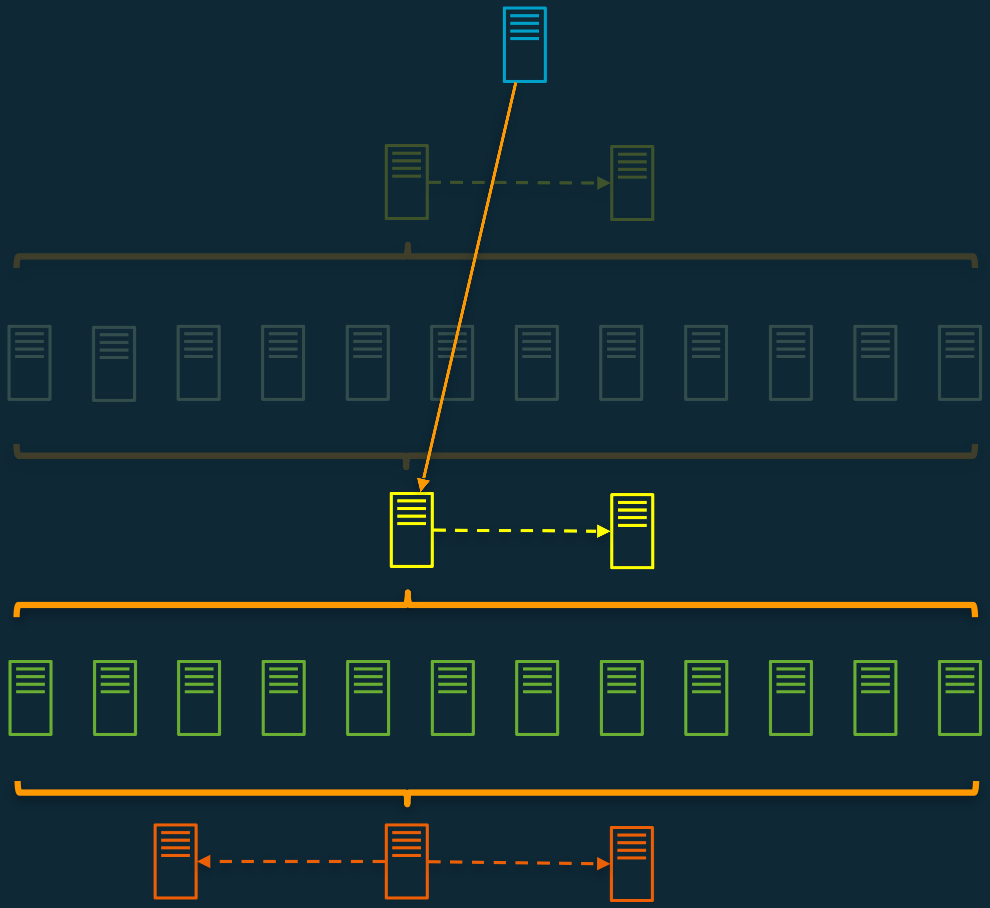


Amazon S3
Lambda@edge

AWS Amplify Console

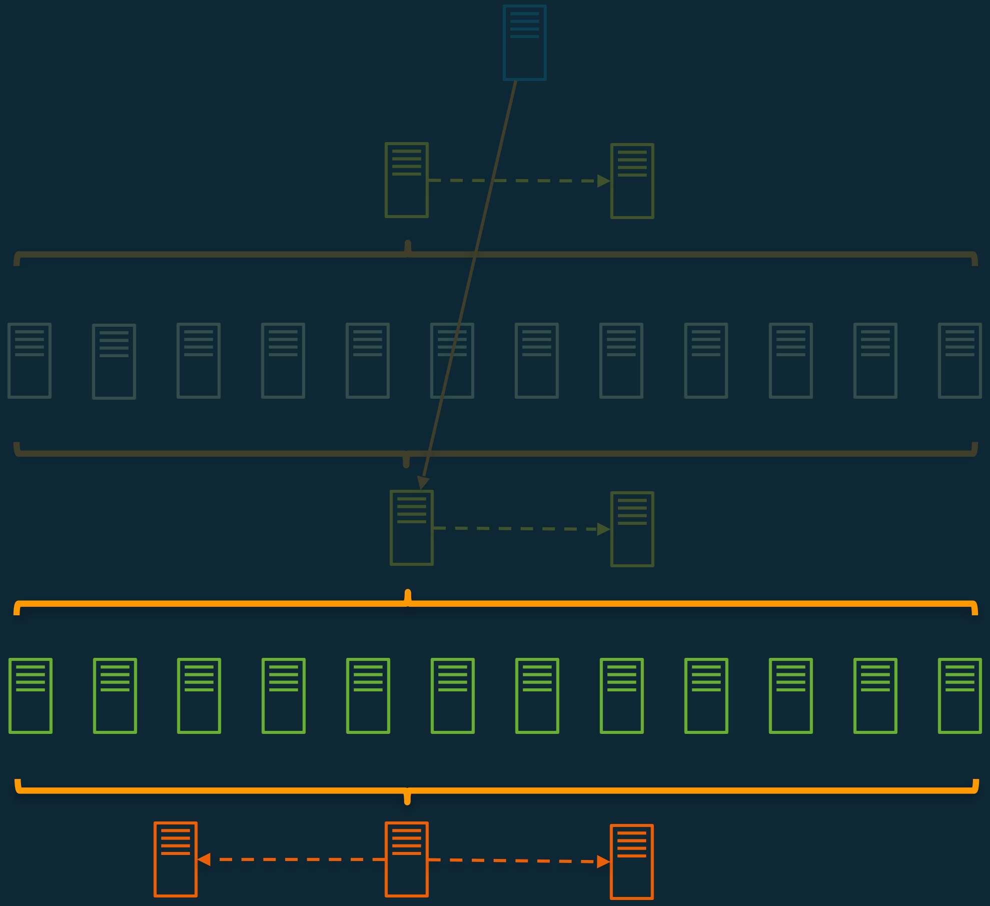
- Powered by Lambda@edge and Amazon S3
- CI/CD
- Build configurations
- Feature branch deployments
- Global availability (CDN)
- Basic password protection

Moving to serverless – API



The proxy/firewall now points directly at the backend.

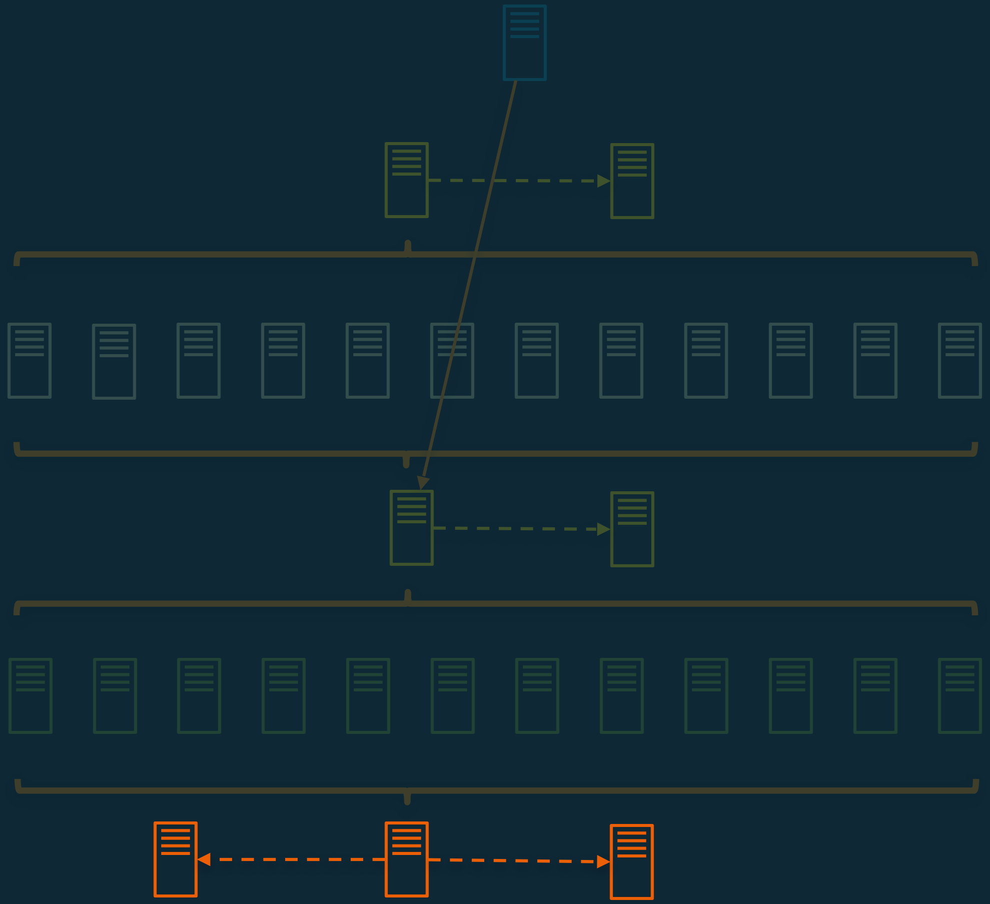
Moving to serverless – API



Amazon API Gateway

- Load balancing and redundancy built in
- REST, WebSocket
- 10k+ requests per second (soft limit)
- Private, regional or global options
- Application authentication and authorization
- SSL offloading
- Custom domains
- API user keys and throttling
- Data validation
- Resource Policy, IAM security

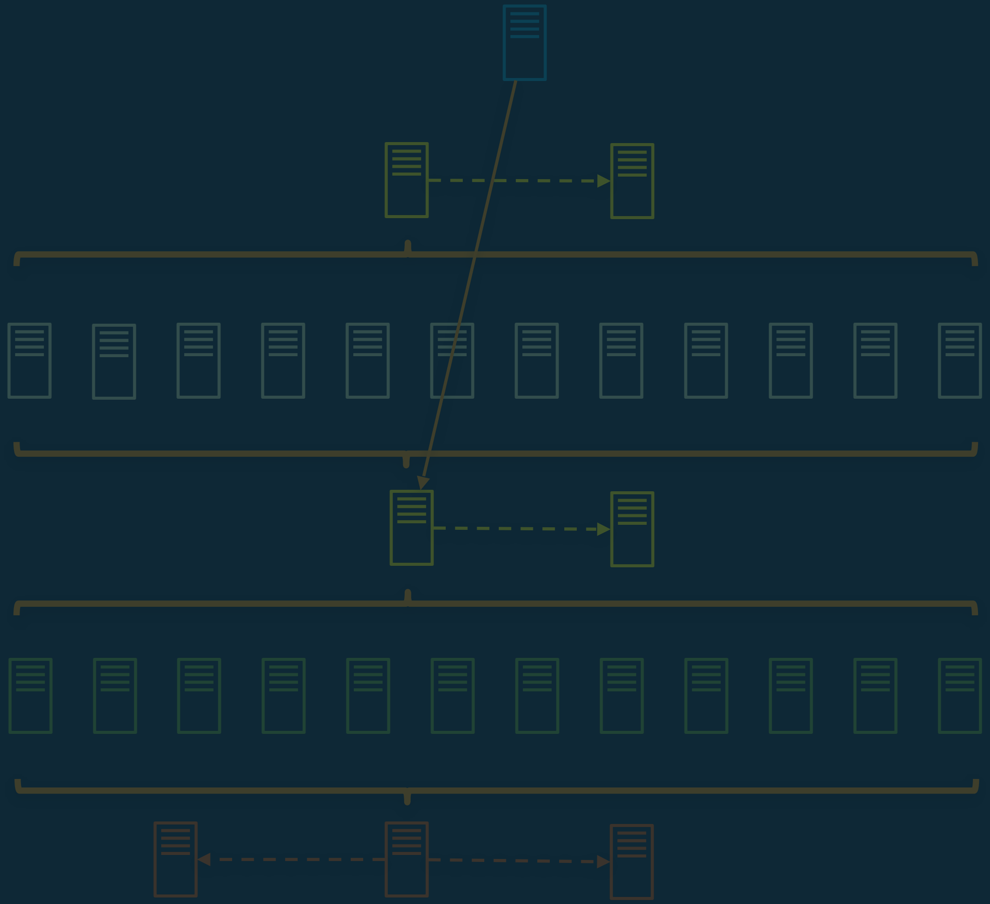
Moving to serverless – compute



AWS Lambda

- Supports Node, Python, Go, Ruby, Java, and .NetCore natively
- Custom runtime for all other languages
- Triggered by many internal and external events
- 1k concurrent invocations (soft limit)
- Triggered synchronously and asynchronously

Moving to serverless – database



Amazon Aurora

Amazon DocumentDB (with MongoDB compatibility)

Amazon DynamoDB

Amazon Redshift

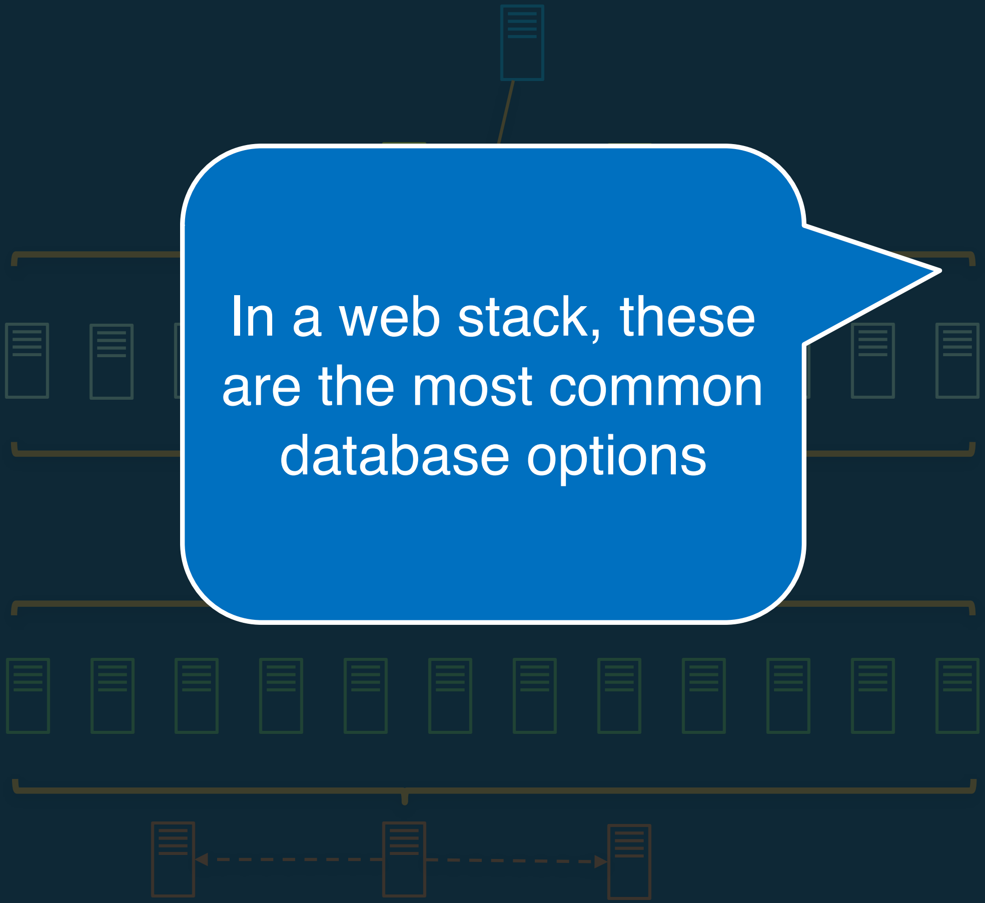
Amazon RDS

Amazon Quantum Ledger Database

Amazon Neptune

Amazon Timestream

Moving to serverless – database



In a web stack, these are the most common database options

Amazon Aurora

Amazon DocumentDB (with MongoDB compatibility)

Amazon DynamoDB

Amazon Redshift

Amazon RDS

Amazon Quantum Ledger Database

Amazon Neptune

Amazon Timestream

Moving to serverless – database

Relational Databases

Amazon Aurora Serverless

- Serverless
- Highly redundant
- Highly available
- Auto scaling
- PostgreSQL
- MySQL

Amazon RDS

- Simple scaling
- Engine native replication
- Microsoft SQL
- MySQL
- PostgreSQL
- MariaDB
- Oracle

NoSQL Databases

Amazon DynamoDB

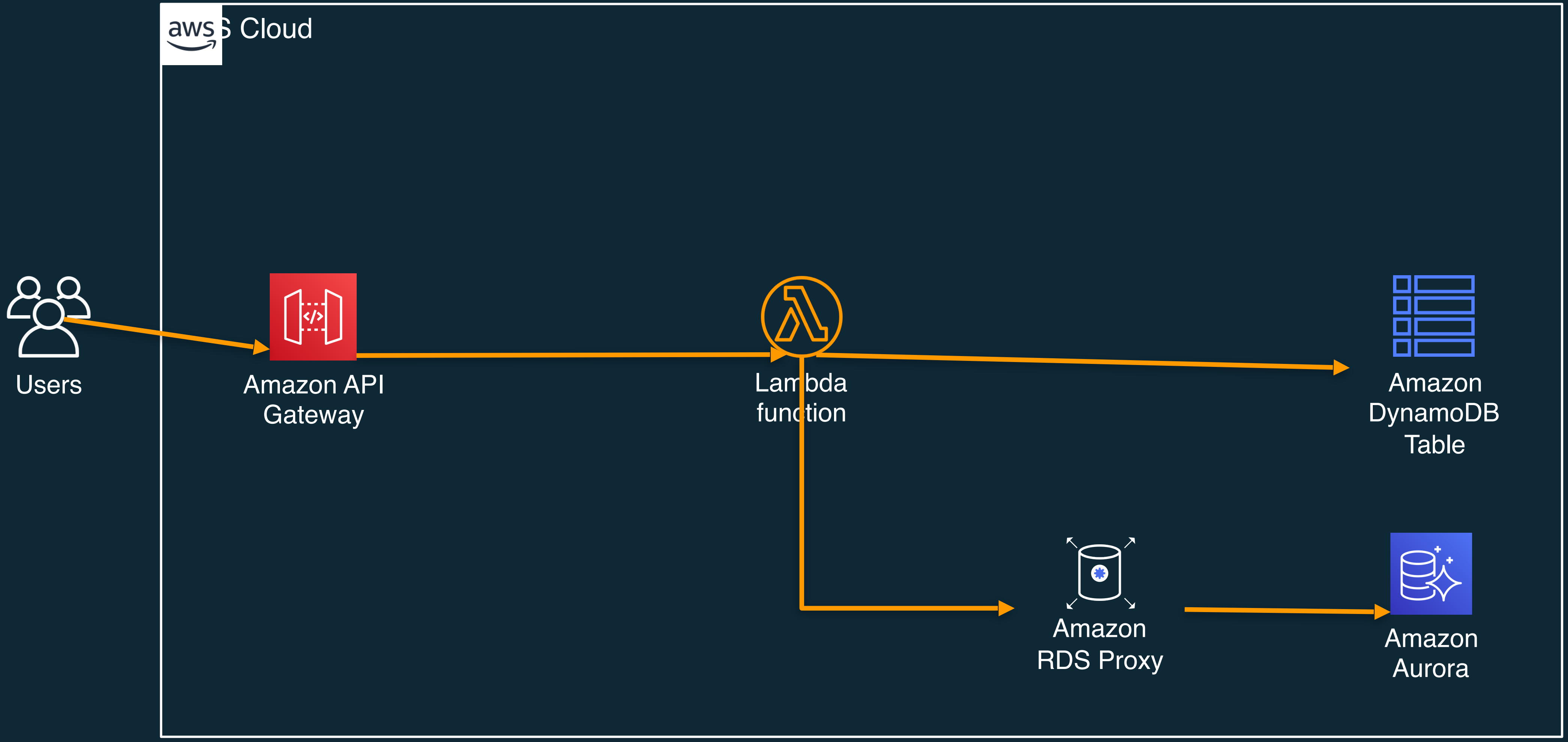
- Serverless
- Highly redundant
- Highly available
- Global tables
- High input and output

Amazon DocumentDB

- Highly redundant
- Highly available
- MongoDB Compatible

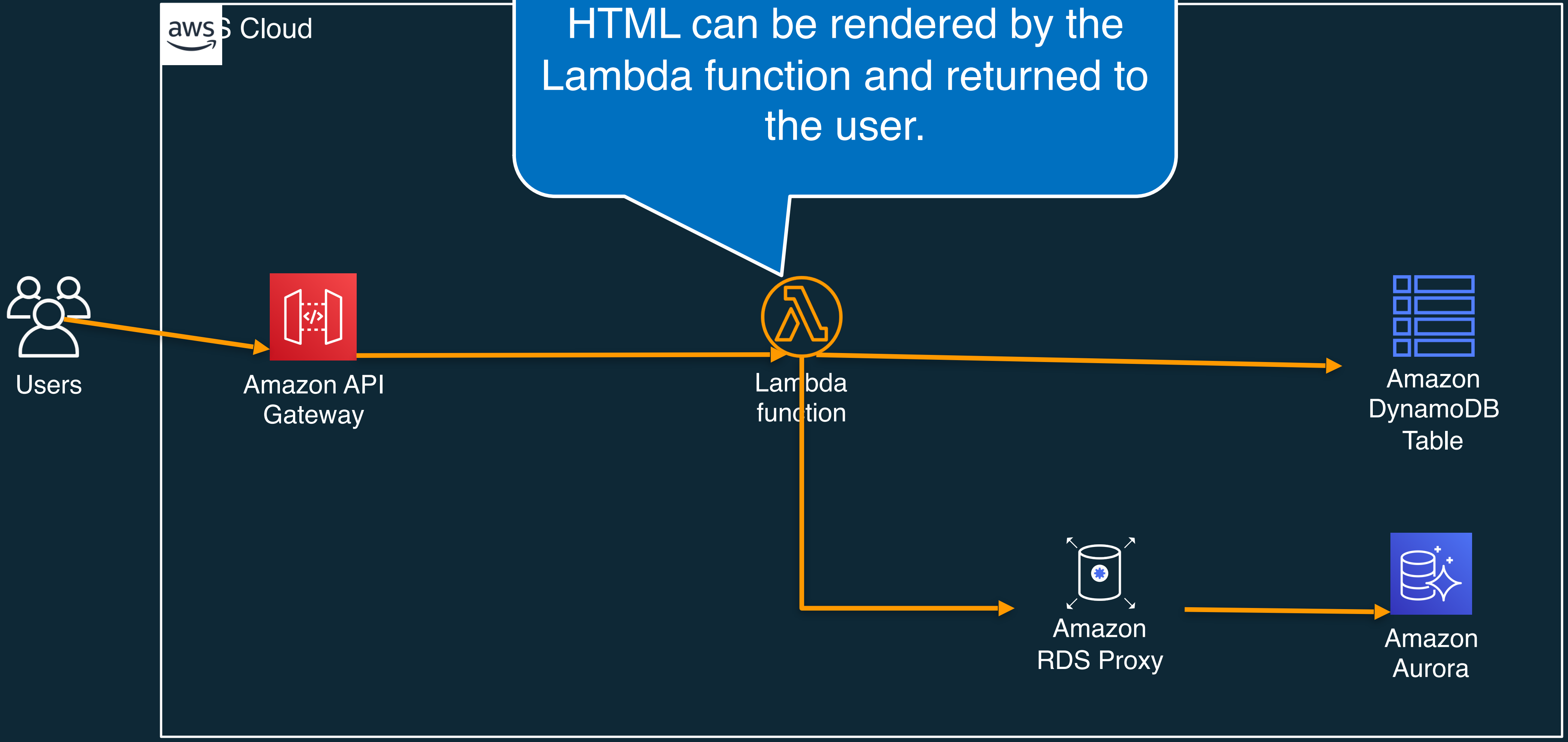
Putting it all together

A basic website

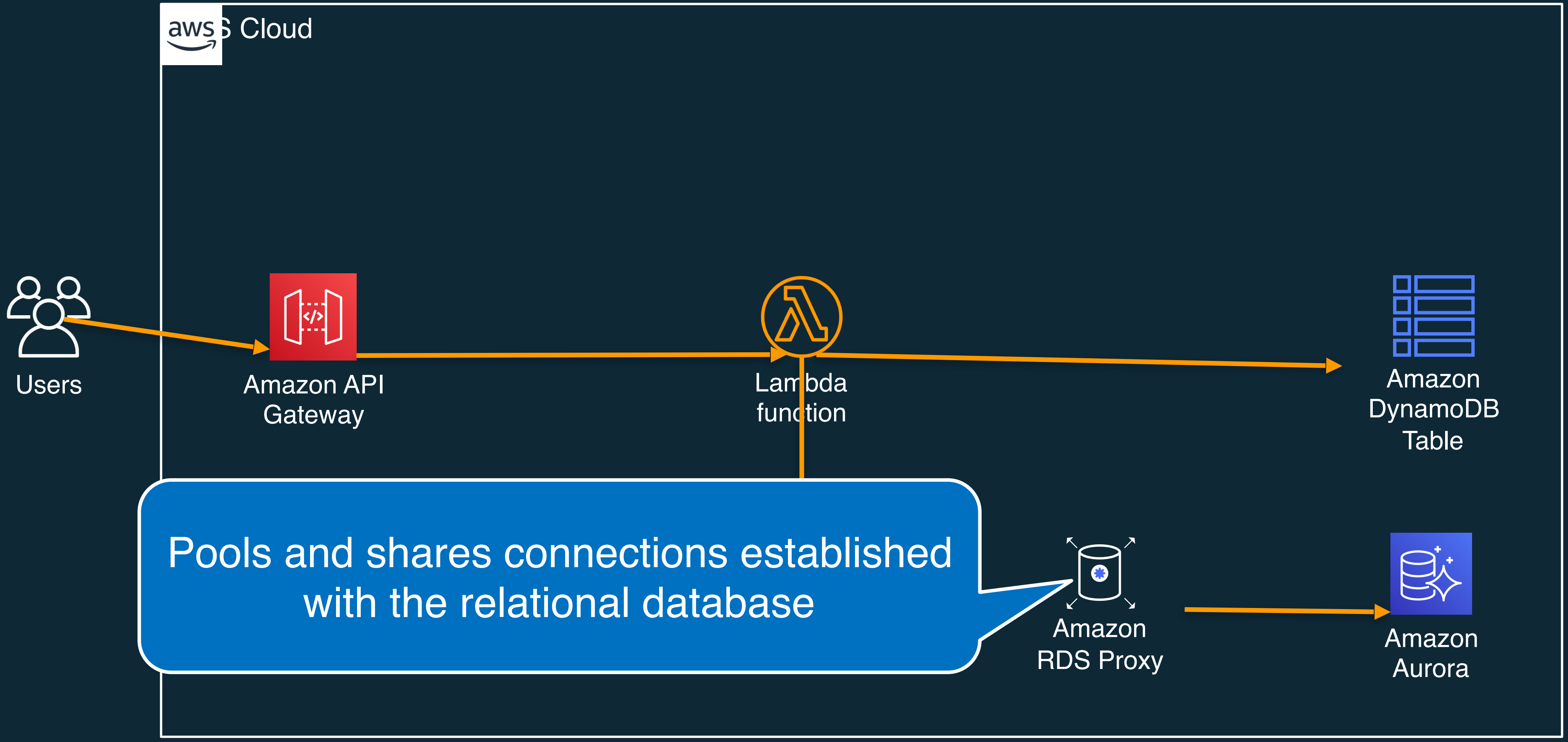


A basic website

HTML can be rendered by the Lambda function and returned to the user.

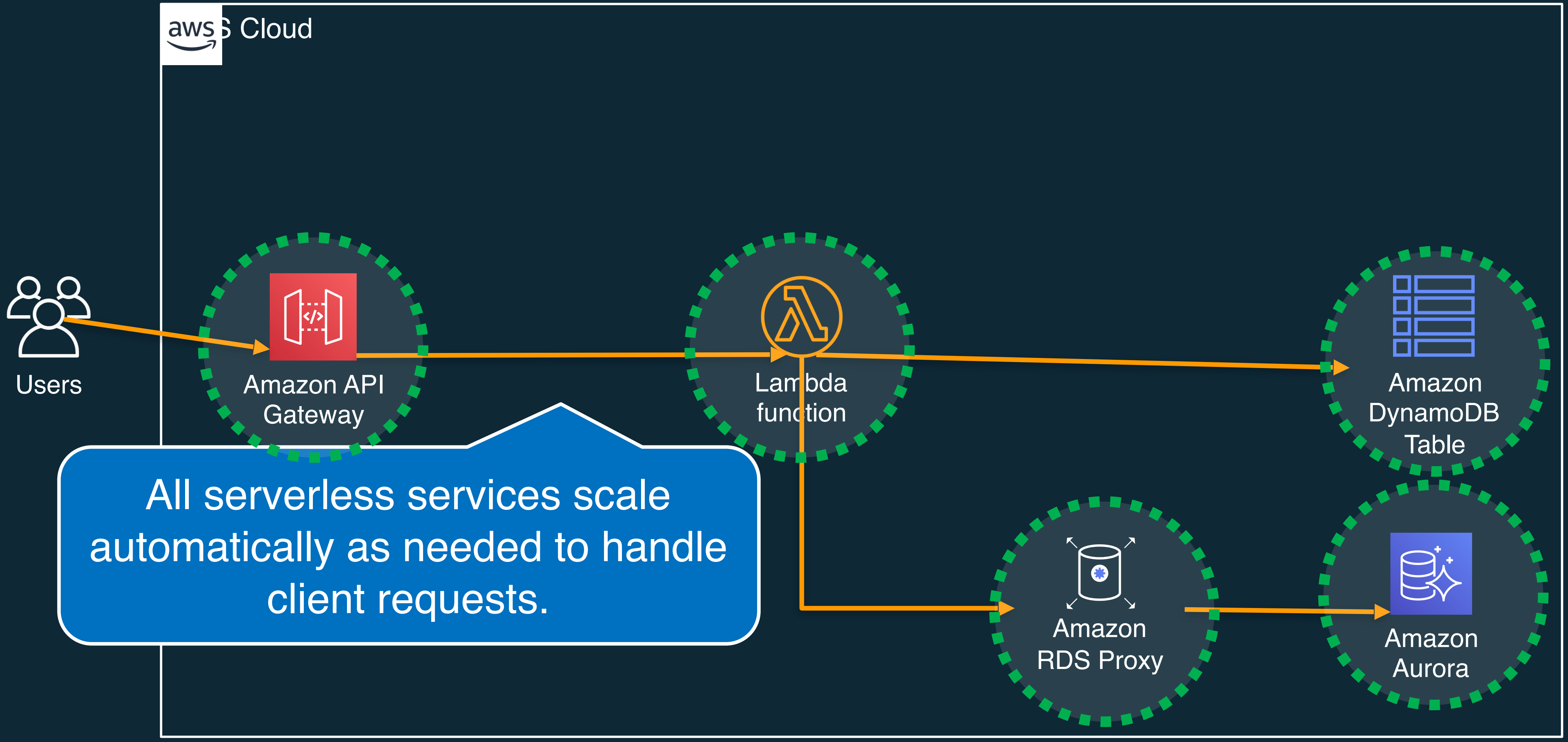


A basic website



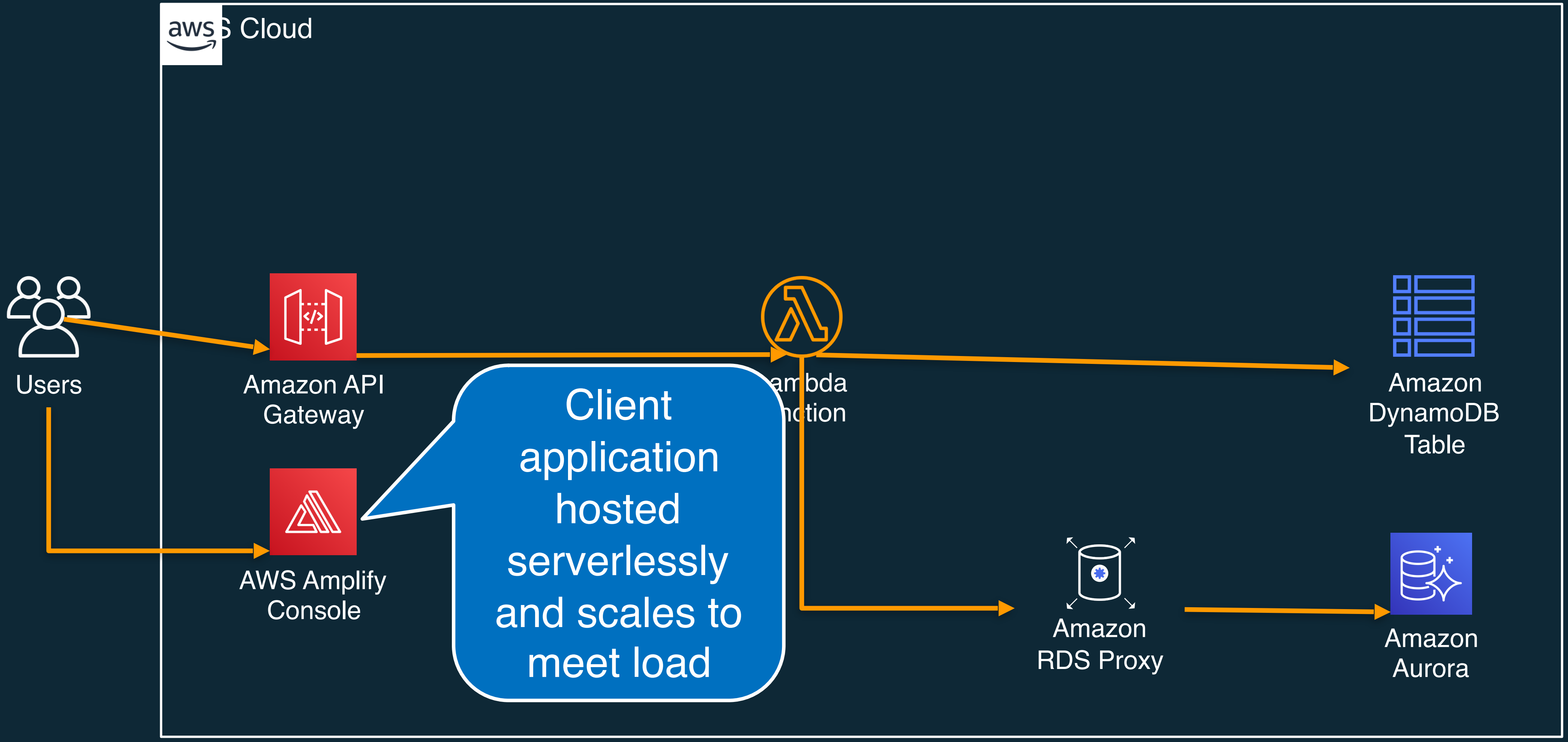
Pools and shares connections established with the relational database

A basic website

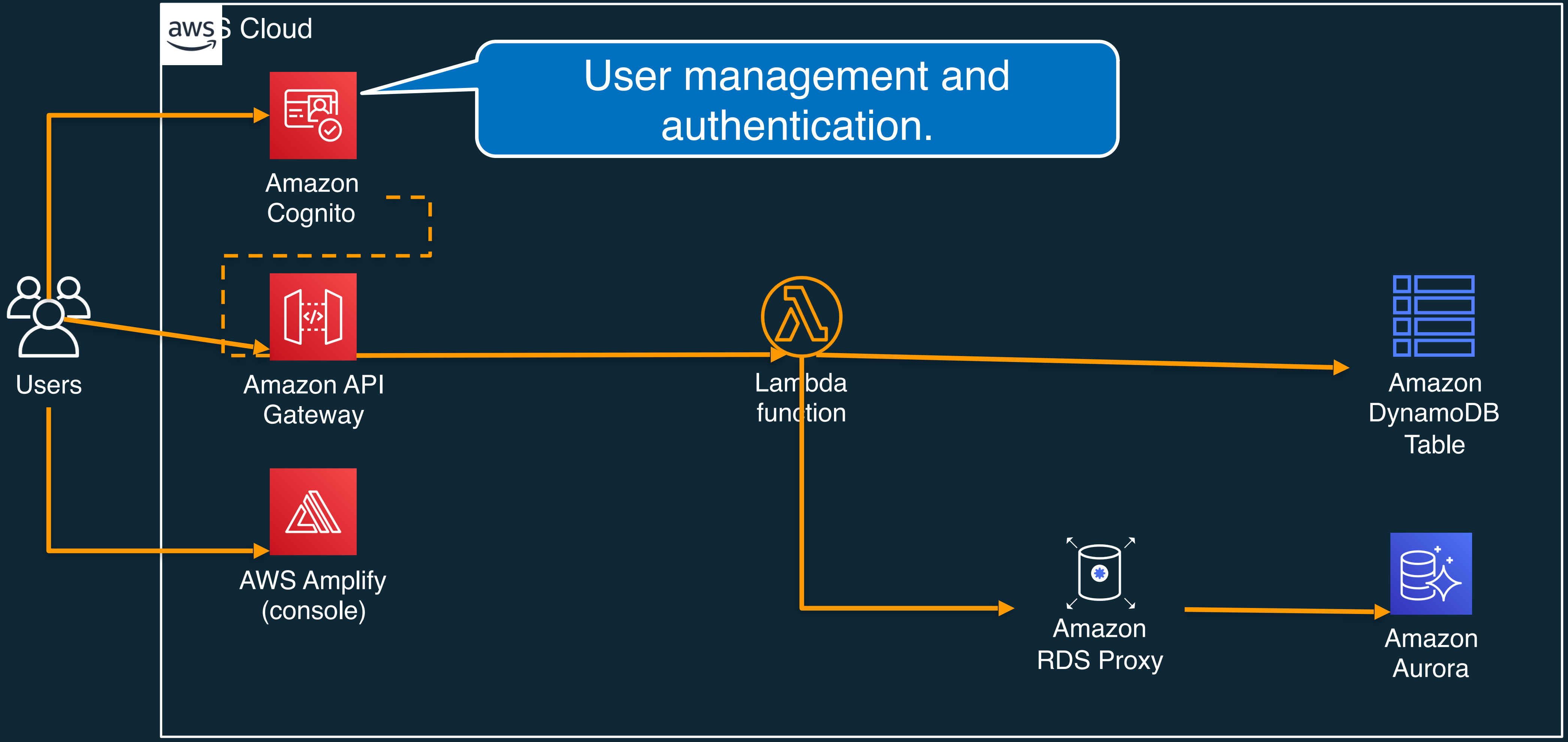


All serverless services scale automatically as needed to handle client requests.

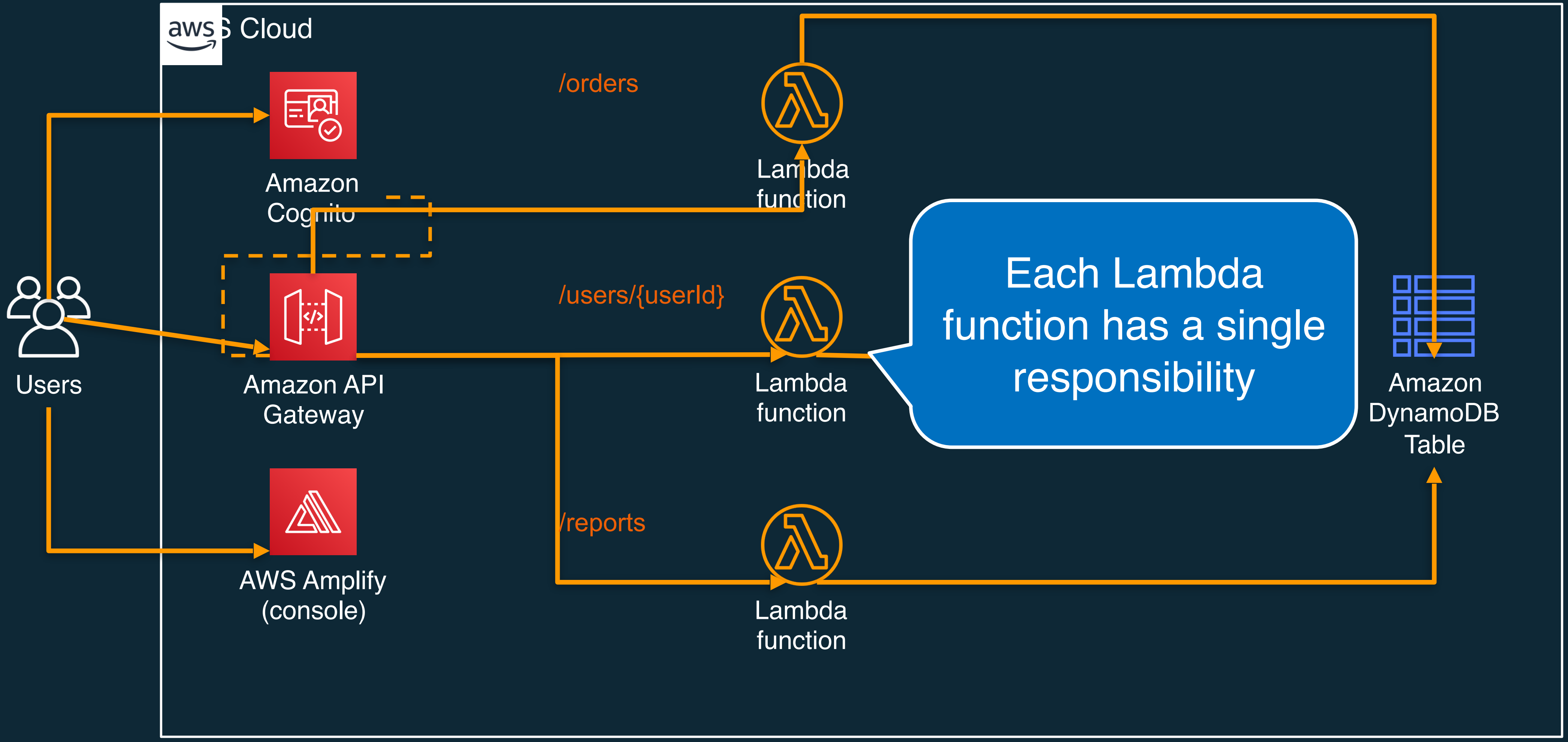
A basic client/backend website



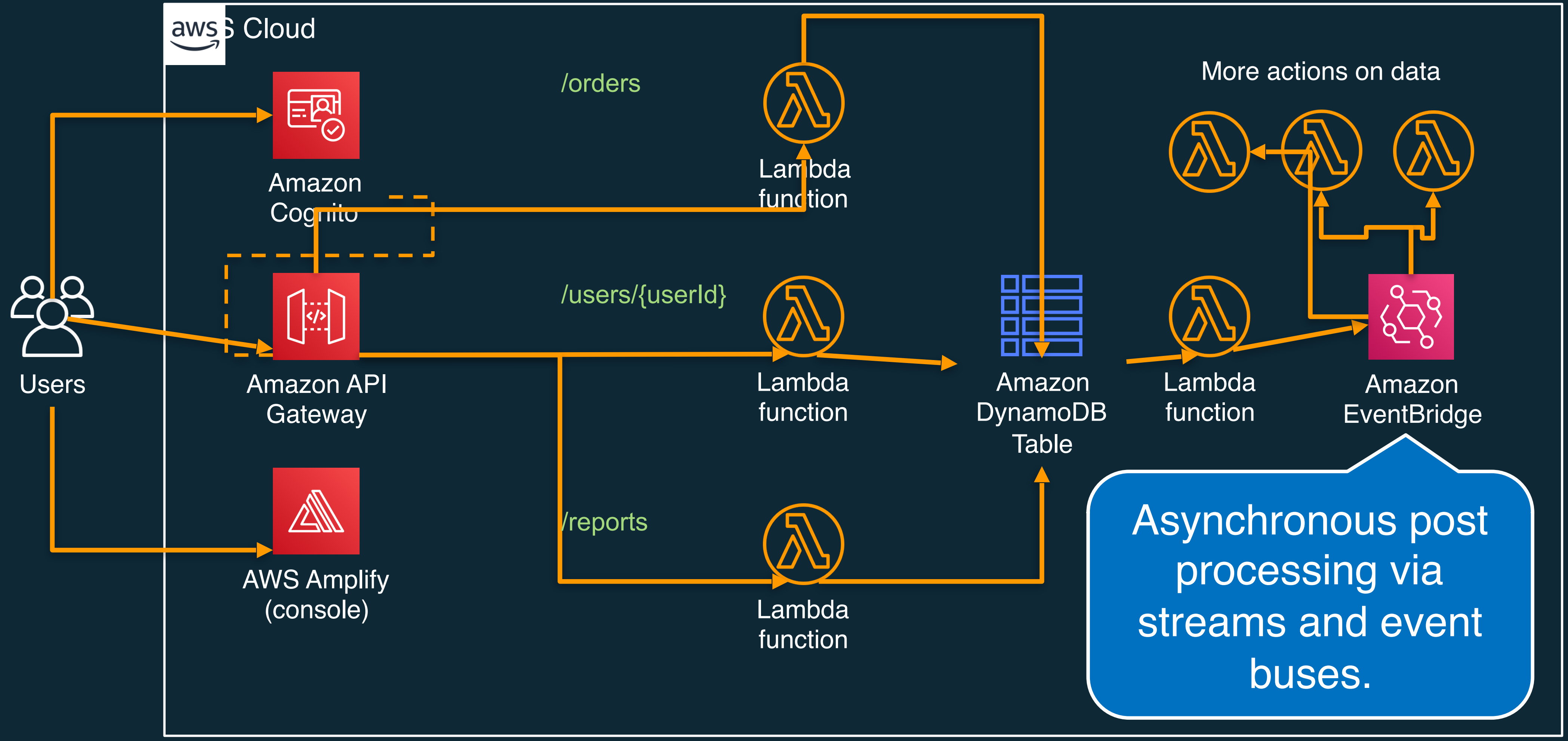
A basic secure client/backend website



A microservice website



A microservice website



Asynchronous post processing via streams and event buses.

Getting started

START WITH A FRAMEWORK

START WITH A FRAMEWORK START WITH A FRAMEWORK

START WITH A FRAMEWORK START WITH A FRAMEWORK

START WITH A FRAMEWORK

START WITH A FRAMEWORK!

START WITH A FRAMEWORK

START WITH A FRAMEWORK START WITH A FRAME
WORK START WITH A FRAMEWORK START WITH
A FRAMEWORK START WITH A FRAMEWORK
START WITH A FRAMEWORK START WITH A FR

START WITH A FRAMEWORK

START WITH A FRAMEWORK

START WITH A FRAMEWORK

START WITH A FRAMEWORK

START WITH A FRAMEWORK

START WITH A FRAMEWORK

START WITH A FRAMEWORK

START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK START WITH A FRAMEWORK

!

Frameworks: AWS SAM

Serverless Application Model

SAM comes in two parts

SAM comes in *2* parts

SAM templates

Using shorthand syntax to express resources and event source mappings, it provides infrastructure as code (IaC) for serverless applications.

SAM CLI

Provides tooling for local development, debugging, build, packaging, and deployment for serverless applications

https://aws.amazon.com/serverless/sam/

Frameworks: AWS Amplify

- Designed for front-end developers
- Allows creation of full-stack serverless infrastructure
- Uses SAM for backend management
- Helps manage front end code and backend connectivity for:
    - Analytics
    - APIs
    - AR/VR
    - Auth
    - DataStore
    - PubSub
    - And more

https://aws-amplify.github.io/docs/

Frameworks: Third Party

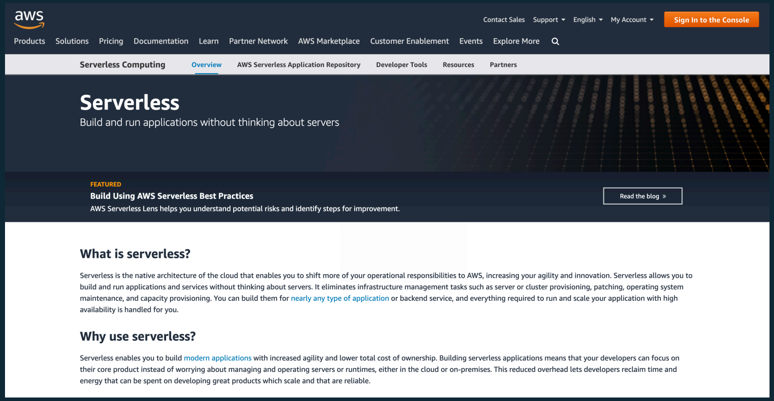
Claudia.js

serverless

APEX
serverless architecture

SPARTA

Final resources



Serverless

Build and run applications without thinking about servers

FEATURED

Build Using AWS Serverless Best Practices

AWS Serverless Lens helps you understand potential risks and identify steps for improvement.

What is serverless?

Serverless is the native architecture of the cloud that enables you to shift more of your operational responsibilities to AWS, increasing your agility and innovation. Serverless allows you to build and run applications and services without thinking about servers. It eliminates infrastructure management tasks such as server or cluster provisioning, patching, operating system maintenance, and capacity provisioning. You can build them for nearly any type of application or backend service, and everything required to run and scale your application with high availability is handled for you.

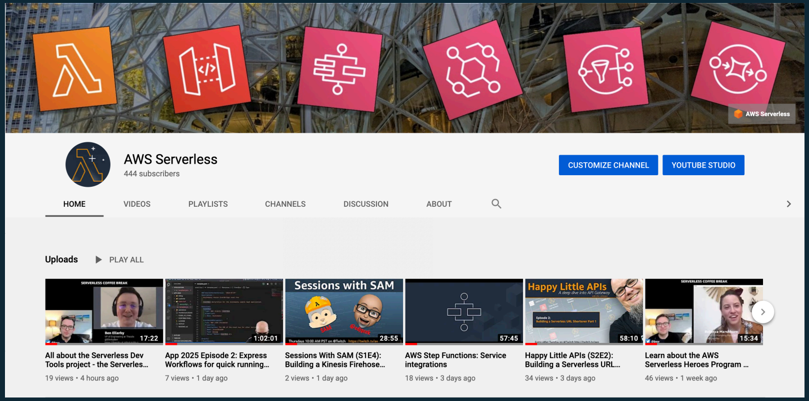
Why use serverless?

Serverless enables you to build modern applications with increased agility and lower total cost of ownership. Building serverless applications means that your developers can focus on their core product instead of worrying about managing and operating servers or runtimes, either in the cloud or on-premises. This reduced overhead lets developers reclaim time and energy that can be spent on developing great products which scale and that are reliable.

AWS Serverless

slip.link/aws-serverless

Final resources



AWS Serverless
444 subscribers

Uploads

All about the Serverless Dev Tools project - the Serverles…
19 views • 4 hours ago

App 2025 Episode 2: Express Workflows for quick running…
7 views • 1 day ago

Sessions With SAM (S1E4): Building a Kinesis Firehose…
2 views • 1 day ago

AWS Step Functions: Service integrations
18 views • 3 days ago

Happy Little APIs (S2E2): Building a Serverless URL…
34 views • 3 days ago

Learn about the AWS Serverless Heroes Program …
46 views • 1 week ago

AWS Serverless YouTube Channel

slip.link/serverless

Eric Johnson

@edjgeek

Image Source: https://pixabay.com/illustrations/thank-you-polaroid-letters-2490552/