

The Complete Guide to Build Serverless Apps on AWS



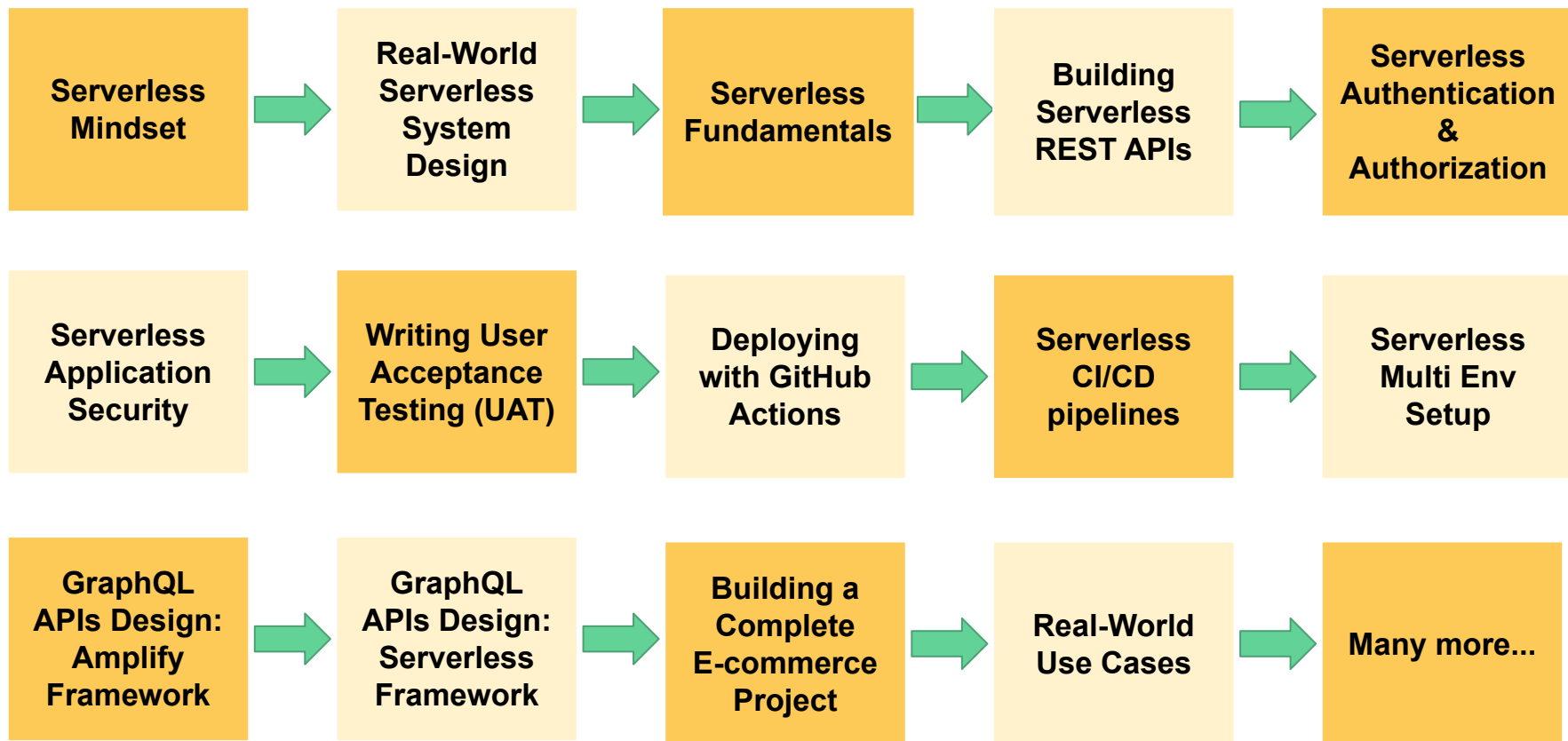
<https://www.youtube.com/c/enlearacademy>



@mjmrz

Manoj Fernando
AWS Community Hero
AWS Certified Solutions Architect - Professional

What will you learn?



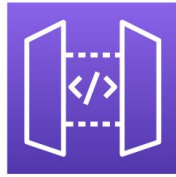
Main AWS Services



DynamoDB



Cognito



API Gateway



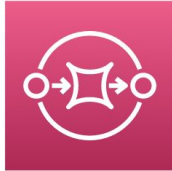
AppSync



EventBridge



Lambda



SQS



SNS



S3



Amplify

Many More...



CloudFront



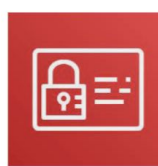
CodePipeline



CodeBuild



CloudFormation



IAM

Serverless Mindset

Serverless Mindset

- Always think about the business requirements
- Serverless-first thinking
- Using Managed Services
- Decoupled Architectures
- Architecture Complexity Vs. Code Complexity

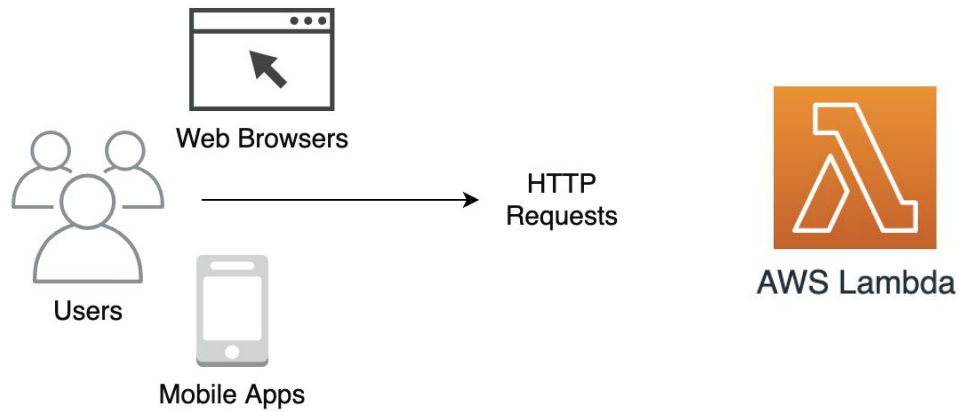
Serverless Mindset

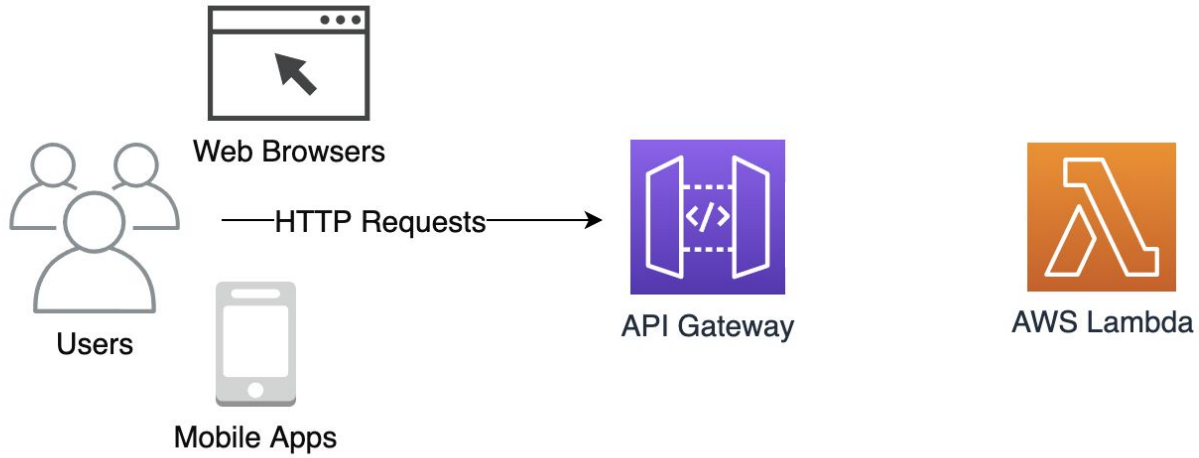
- Adding Observability
- Design for Extensibility
- Focus on Innovation
- Security at All Layers
- Use of Purpose-built databases

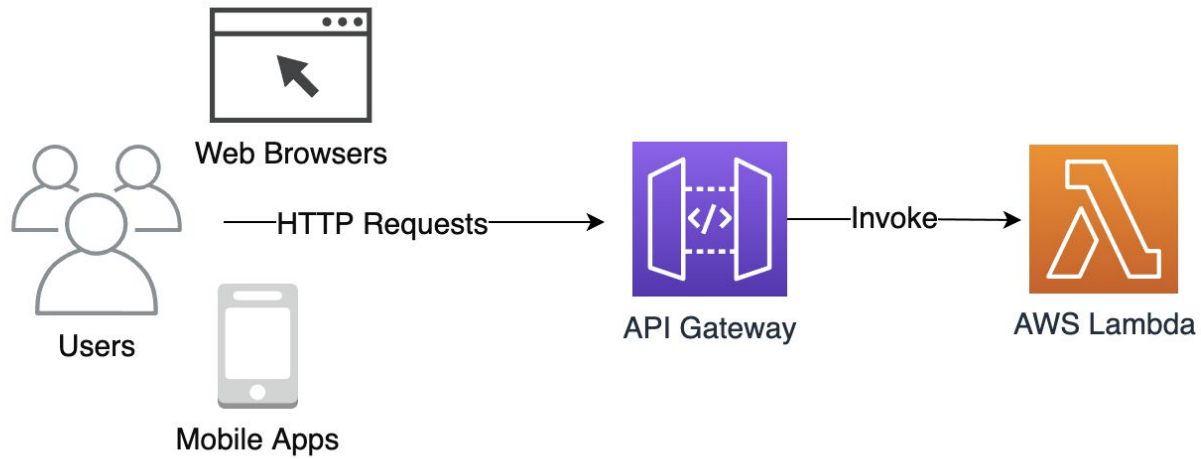
Serverless Fundamentals

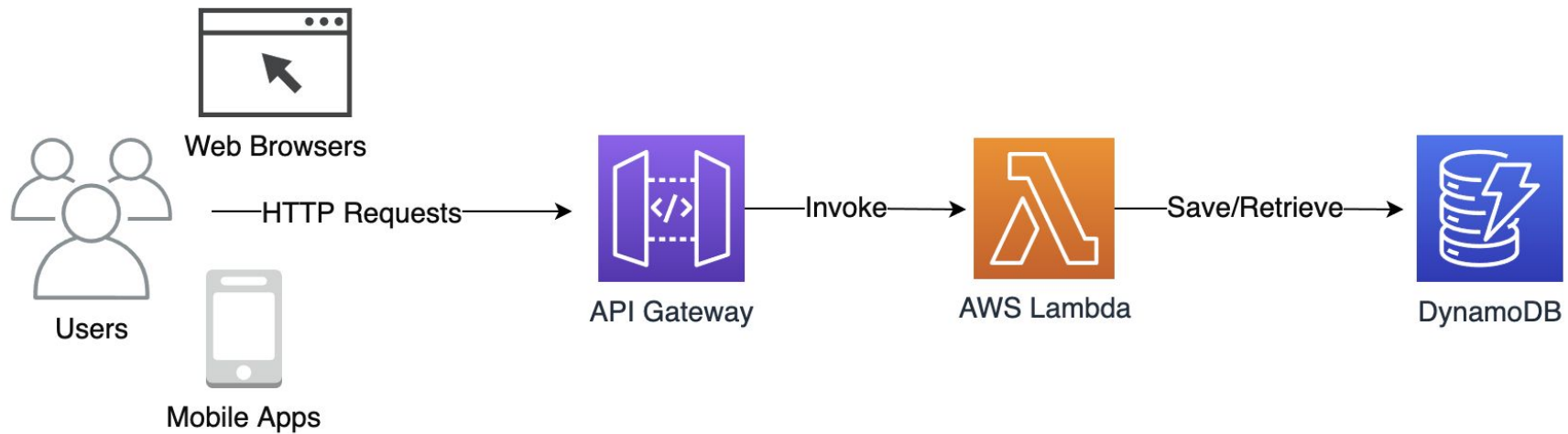


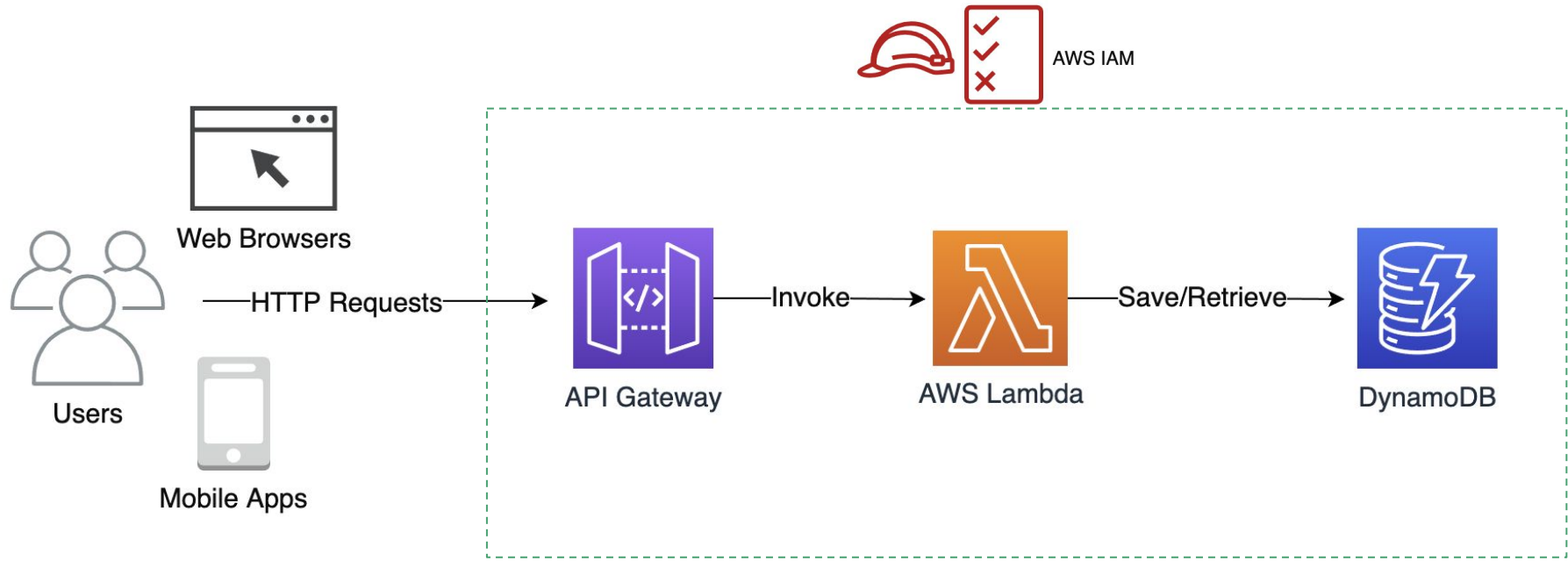
AWS Lambda

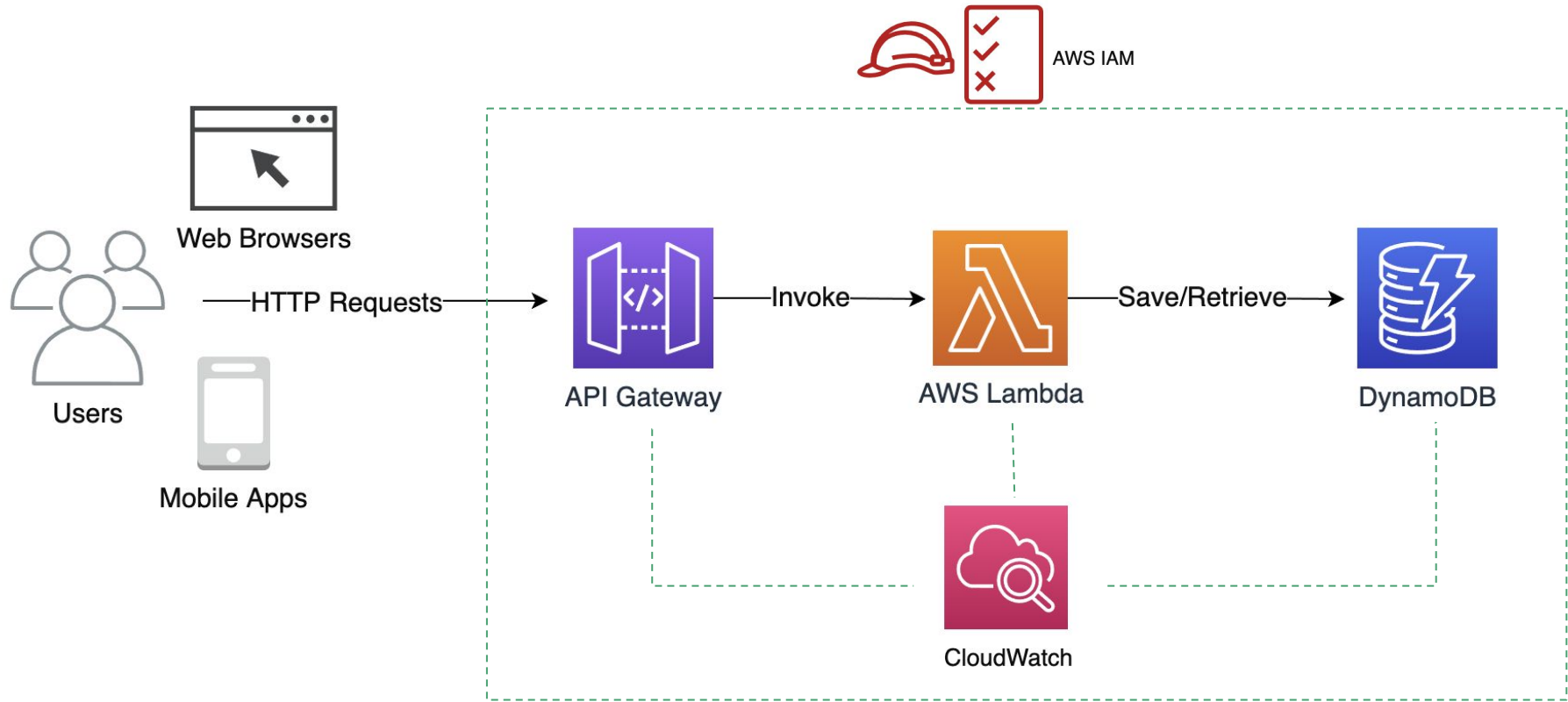












Serverless Fundamentals - AWS

Introduction to popular AWS serverless services

AWS Lambda

DynamoDB

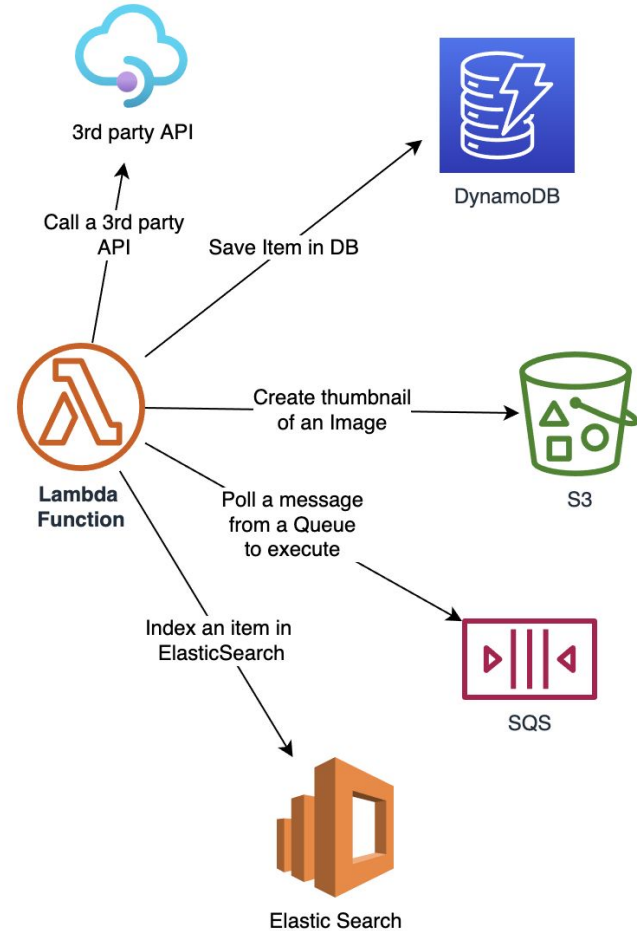
CloudWatch

IAM

API Gateway

AWS Lambda

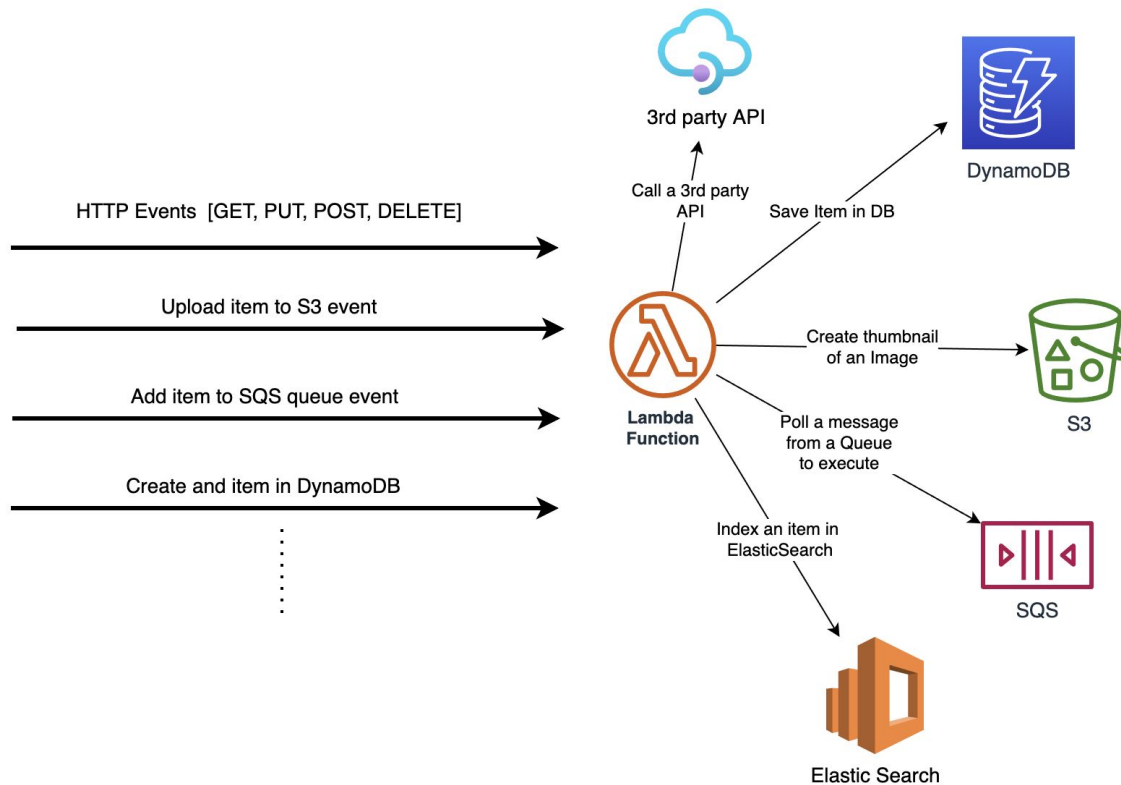
You can **execute a piece of code**
without managing a server



AWS Lambda

Event Driven:

Can respond to various **Events**



AWS Lambda

Serverless

Pay per execution

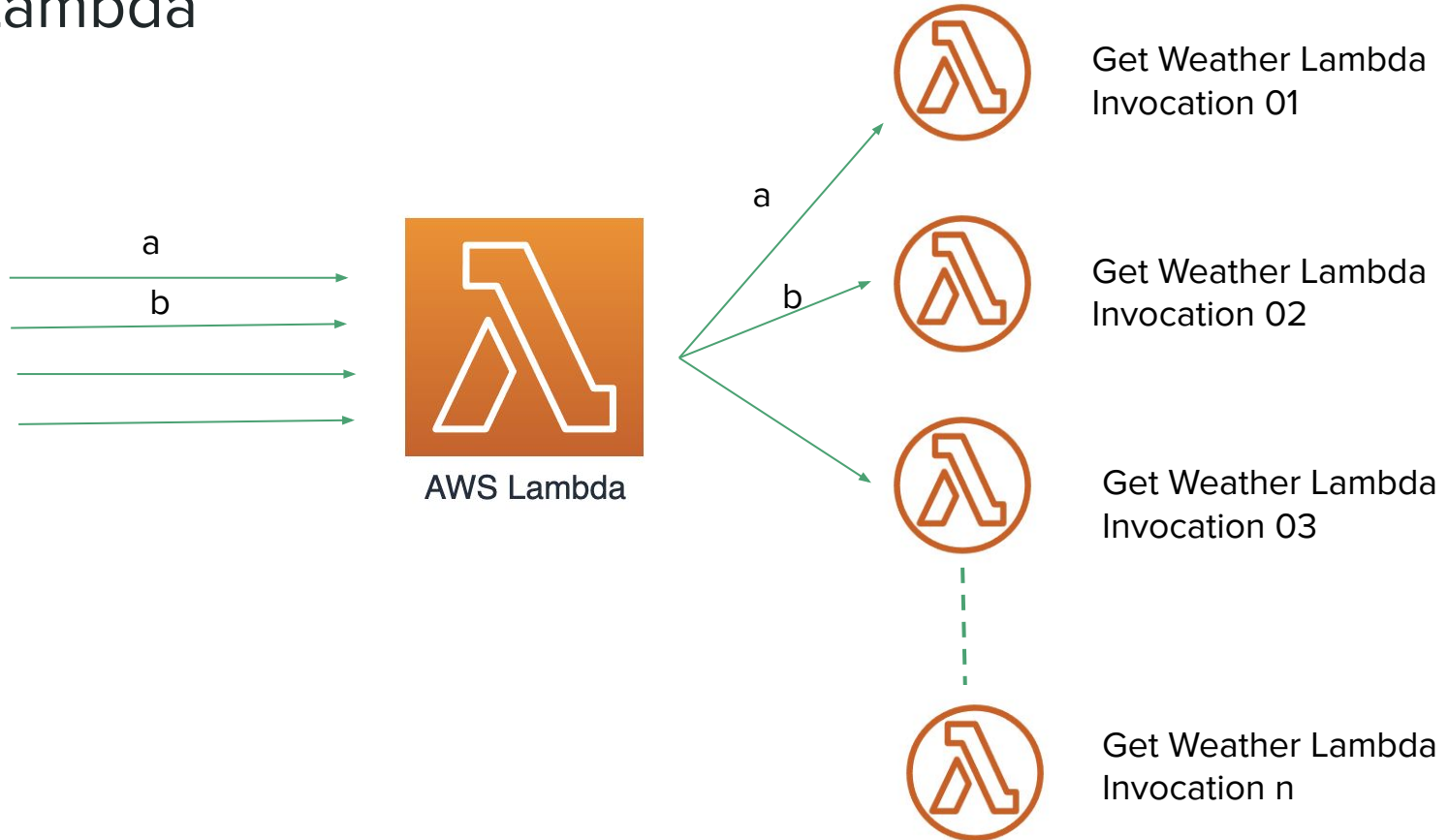
No need to manage to servers

AWS Lambda



AWS Lambda

AWS Lambda



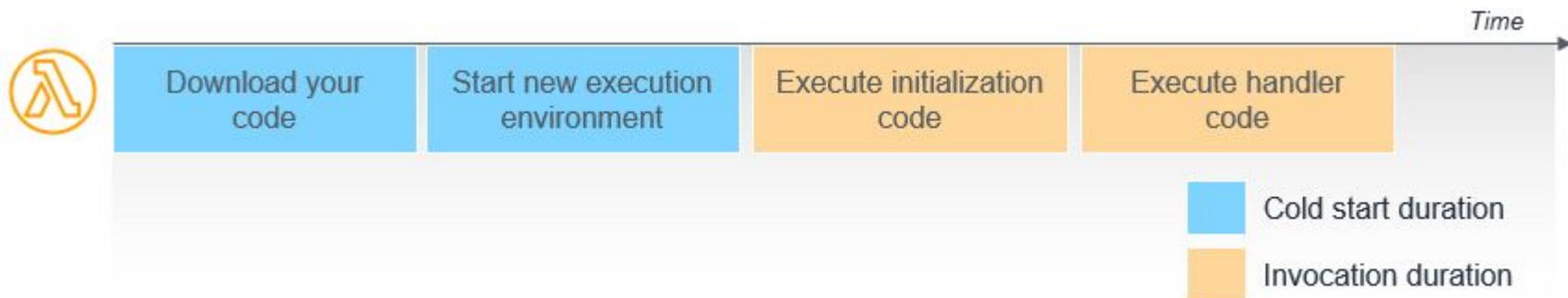
AWS Lambda

Event Driven Serverless Compute Platform

Over 200 AWS Services can trigger Lambda functions

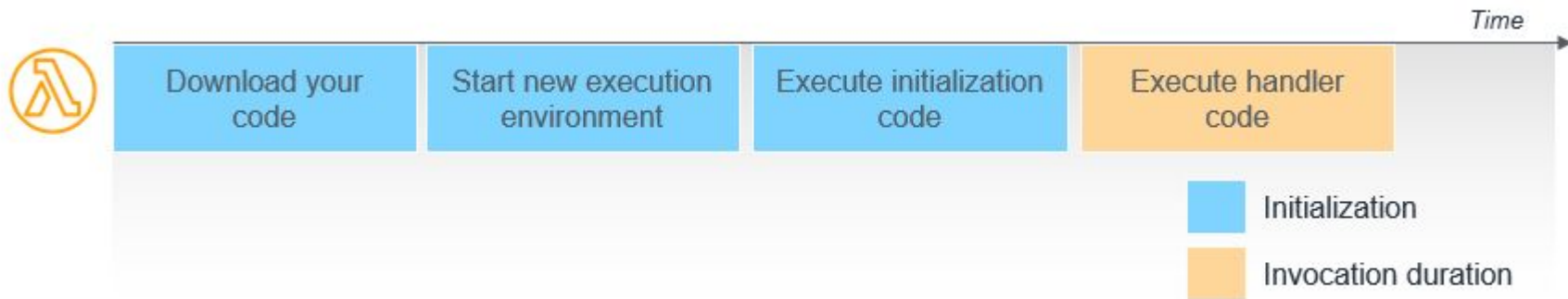
Pay only for the execution time

Lambda Cold Starts

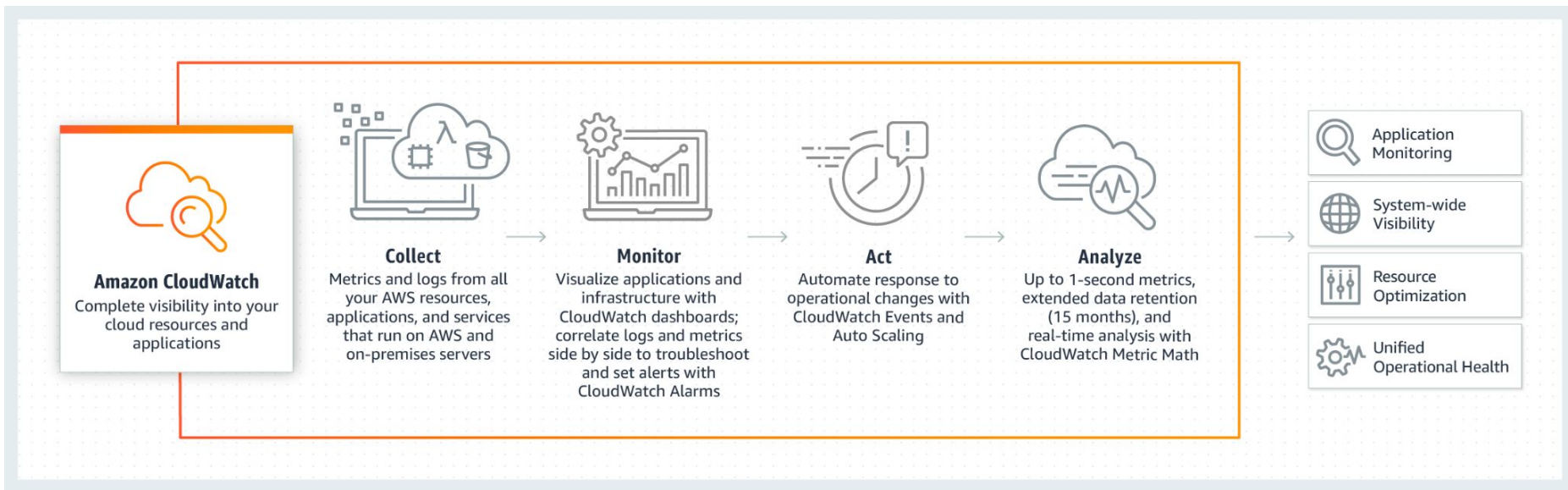


Cold starts varies from 100ms to 1 second

Reducing Cold Starts - Provision Concurrency



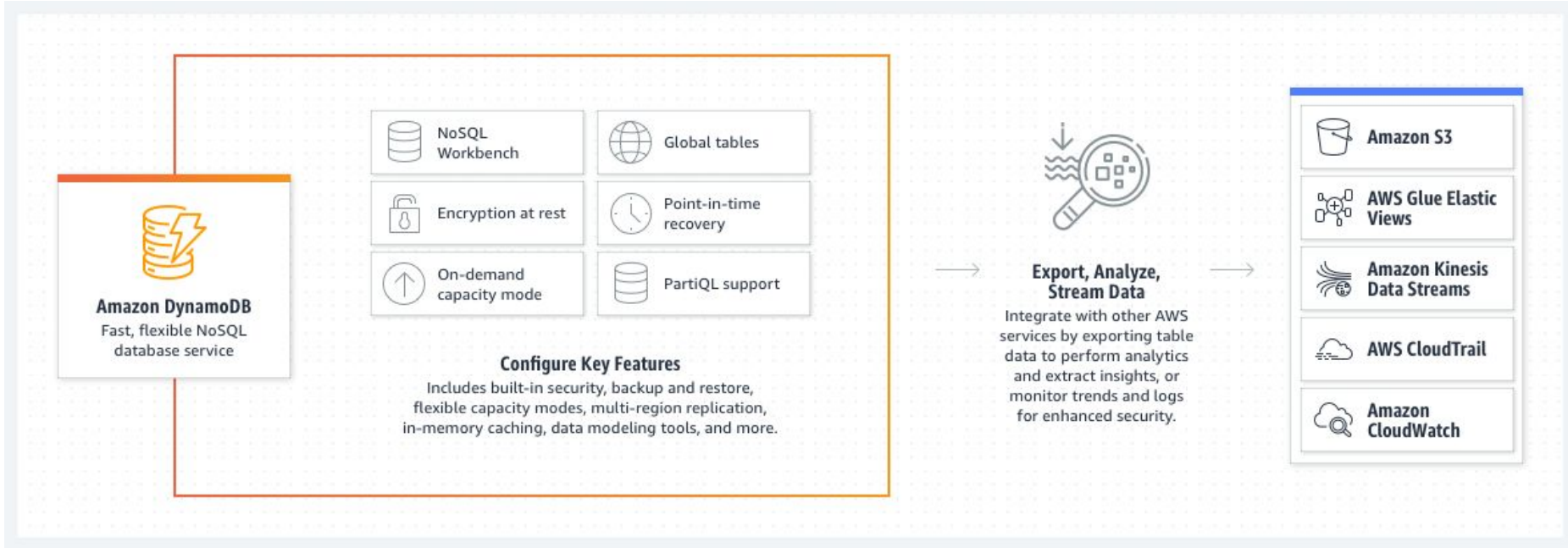
Amazon CloudWatch



Amazon API Gateway



Amazon DynamoDB



AWS IAM



AWS AppSync

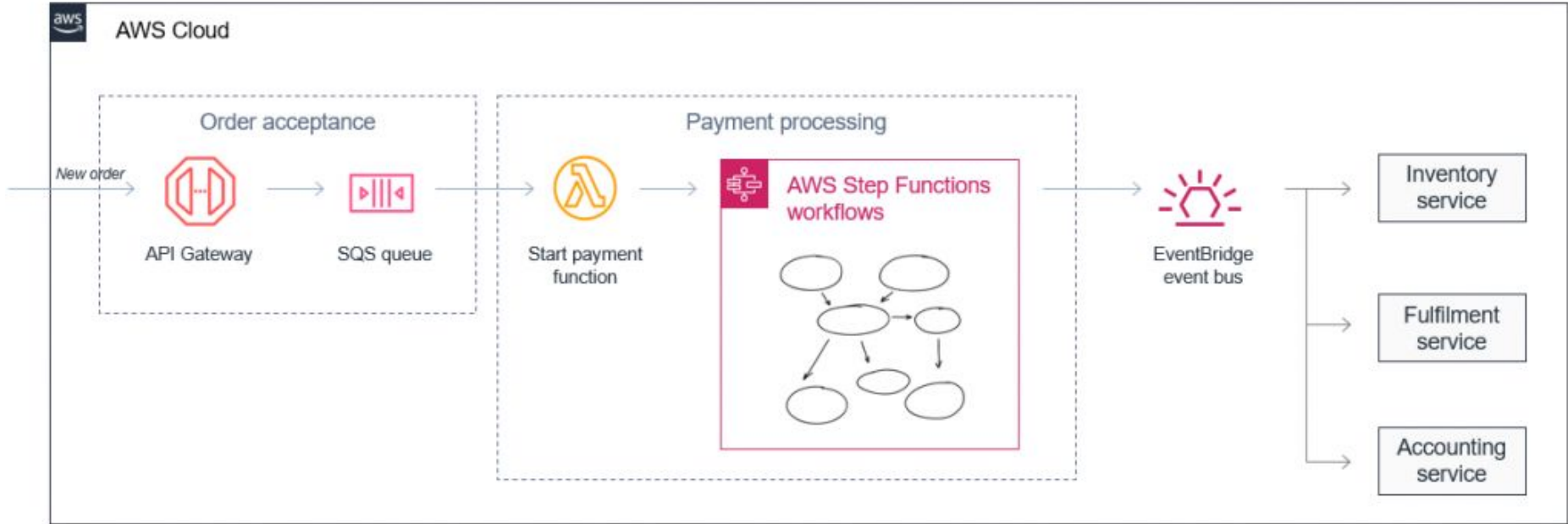
What is AWS AppSync?

- Managed GraphQL service that powers your GraphQL API
- Scalable and Multi-Az out of the box
- \$4 per million Queries/Mutation operations
- Even if AppSync has to run multiple resolvers for one Query it is considered as one operation.

Resource	Description	Default
Schema document size	The maximum size of the schema document	1 MB
Functions per pipeline resolver	The maximum number of functions per pipeline resolver	10
Throttle rate per GraphQL API	The maximum number of GraphQL queries per API per second	1,000 You can request a quota increase .
GraphQL request execution timeout	The maximum GraphQL request execution time for queries, mutations, and subscriptions	30 seconds
Evaluated resolver template size	The maximum size of the evaluated resolver template	5 MB
Request mapping template size	The maximum request mapping template size	64 KB
Response mapping template size	The maximum response mapping template size	64 KB
Iterations in a foreach loop in mapping templates	The maximum number of iterations in a <code>#foreach . . #end</code> loop in mapping templates	1000
Resolvers executed in a single request	The maximum number of resolvers that can be executed in a single request	10,000
Subscription payload size	The maximum size of the message received from	240 KB

amzn.to/2SPwKOb

Serverless Observability Challenges



Reference: AWS blog

Developing GraphQL APIs

AWS AppSync & Amplify Framework

Agenda

- Introduction to GraphQL
- Introduction to AWS AppSync
- Hands-on: Building an online store backend

What is GraphQL?

- GraphQL is a specification introduced by Facebook
- Many languages have GraphQL implementations (<https://graphql.org/code>)
- GraphQL needs a “Schema” to describe the data that clients need
- GraphQL API can aggregate data from multiple data sources easily
- A GraphQL API provides a “Smart endpoint” to the clients to query “Specific” data they need

```
{
  user {
    name
    email
    posts {
      title
      body
    }
  }
}
```

/graphql



```
{
  "user" : {
    "name" : "Manoj Fernando",
    "email": "manoj@test.com",
    "posts": [
      {
        "title" : "First Post",
        "body"  : "Hello World"
      },
      {
        "title" : "Second Post",
        "body"  : "Hello Again"
      }
    ]
  }
}
```

Why GraphQL?

- Many clients (Web, mobile, smart-watches, etc...) can use the same GraphQL API without over-fetching or under-fetching
- Support for real-time communication as part of the specification. E.g. Subscriptions
- Prevents accidental data in HTTP responses
- Support pagination without unnecessary overhead
- Offline support and Many more...

AWS AppSync

- Managed GraphQL service from AWS
- Built in **cognito group-based authorization**
- Provide highly scalable web sockets with AppSync Subscriptions
- Support many data sources from AWS (e.g. DynamoDB, Aurora Serverless, Elastic Search, AWS Lambda etc...)
- Server-side data caching capabilities
- Conflict detection and resolution
- Monitoring and logging

Demo

Designing an online book store API with AWS AppSync and Amplify Framework

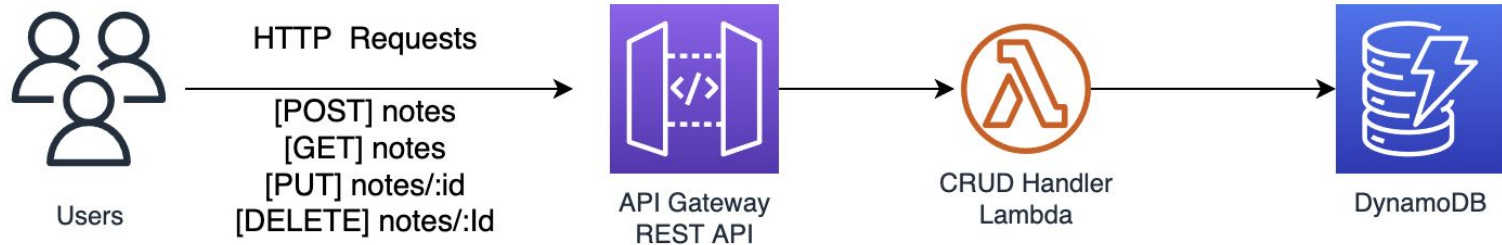
Steps

1. Create *AWS* resources
2. Create the GraphQL schema
3. Add data sources
4. Model relationships among application entities
5. Add authorization rules
6. Deploy and testing the API

Building a Serverless REST API

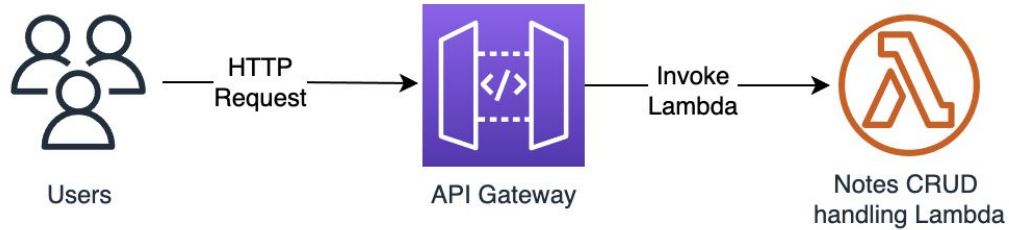
Application Architecture Overview

- REST API that handles Notes CRUD operations




Application Overview

- It's a simple API that handles Notes CRUD operations




Access Control

API Gateway Access Controlling Methods

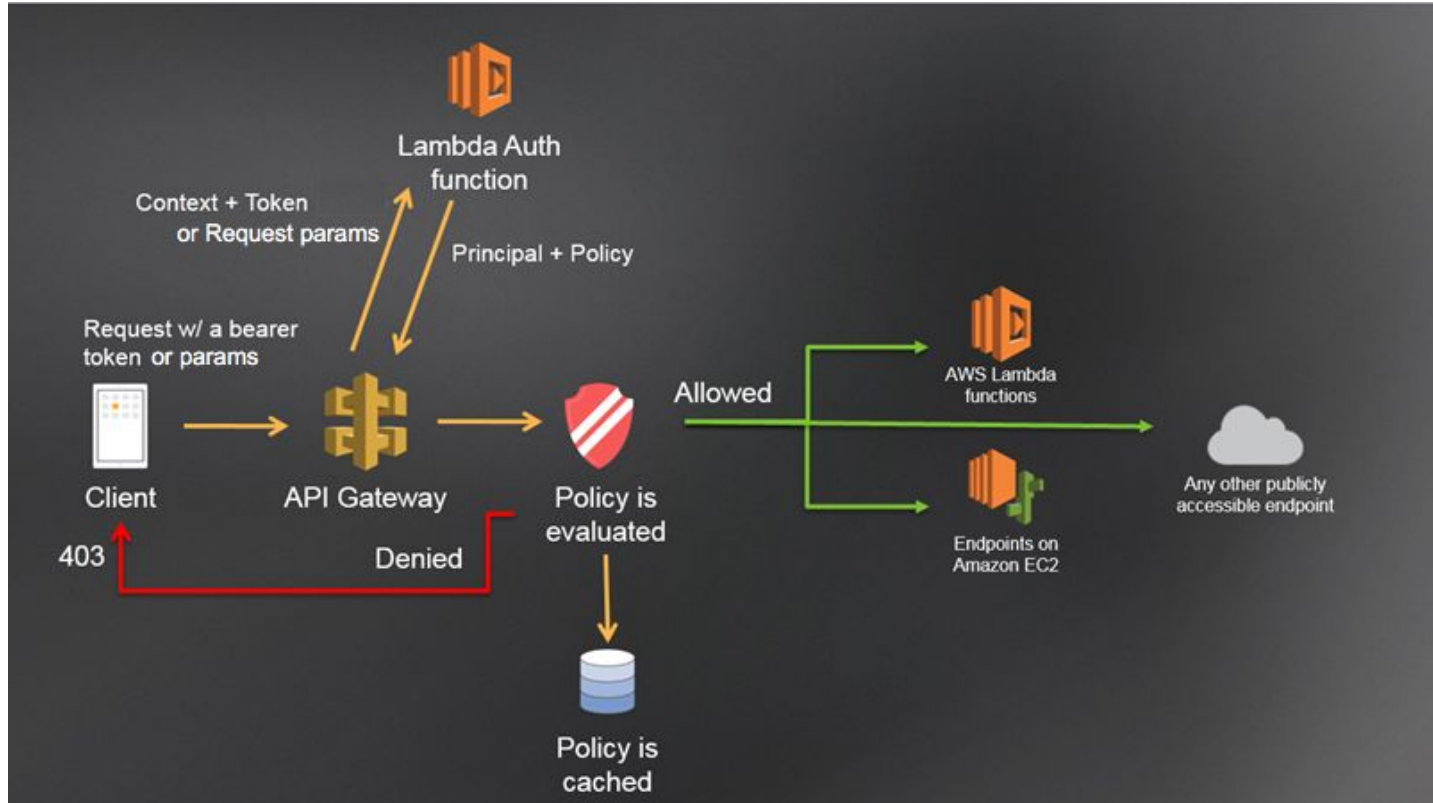
- API Key 
- Lambda Authorizer
- Cognito User Pool Authorizer
- IAM Authorizer
- Others
 - VPC Endpoints
 - API Gateway Resource Policy
 - Tagged-based Access Control

API Gateway Access Controlling Methods

- API Key
- **Lambda Authorizer** 
- Cognito User Pool Authorizer
- IAM Authorizer
- Others
 - VPC Endpoints
 - API Gateway Resource Policy
 - Tagged-based Access Control

Lambda Authorizer

Lambda Authorizer




API Key

API Key

- Not recommended for authentication and authorization
- Used to enforce throttling limits to the APIs daily quota
- Can be used with other Access Control Methods e.g. Lambda authorizer, User Pool Authorizer and IAM
- Can be used in microservices to access internal APIs (From protected environments)

API Gateway Access Controlling Methods

- API Key
- Lambda Authorizer 
- Cognito User Pool Authorizer
- IAM Authorizer

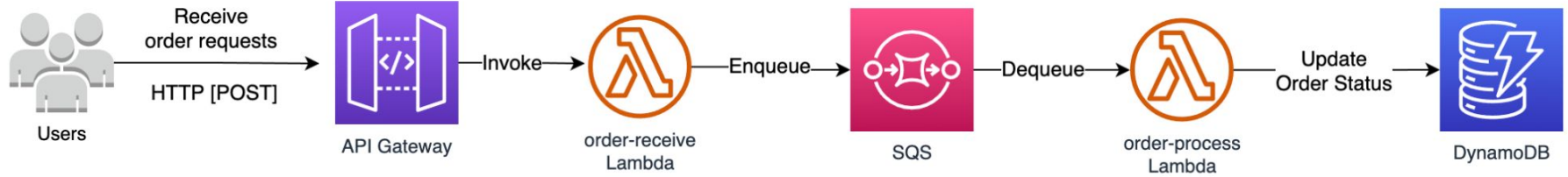
Observability for Serverless Architectures

- Serverless architectures comprise of many connected systems
- Find root-causes of production issues immediately
 - Trace the request across multiple services
 - Find errors along the way
- Identify performance bottlenecks

What are required for better observability?

- Recording custom logs
- Tracing requests across components
- Identifying application/system metrics
- Real-time alerting and dashboards
- A better tool that supports all of the above
 - AWS X-Ray
 - 3rd party tools - Lumigo

Order processing system

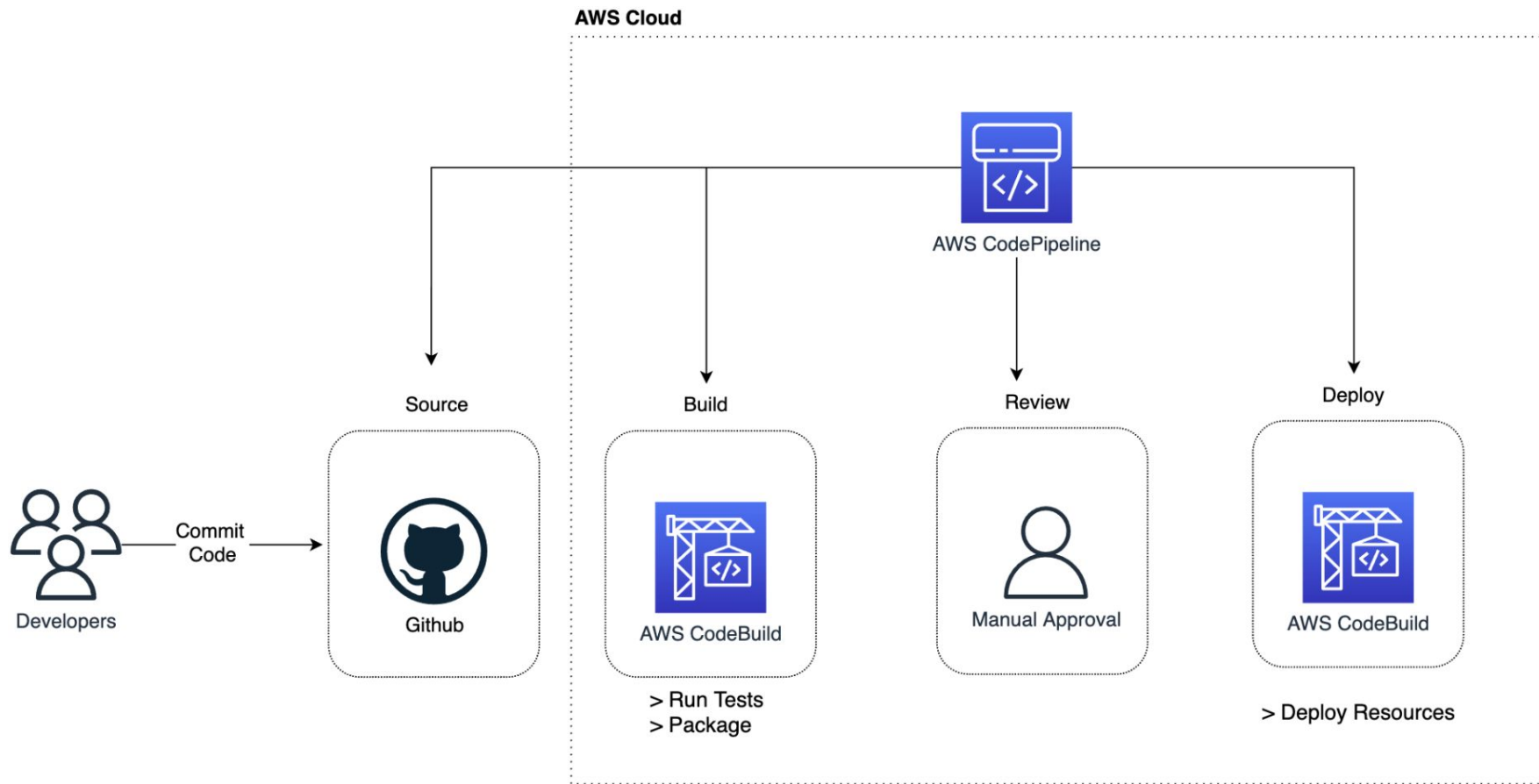


Simple CI/CD Pipeline with GitHub Actions

GitHub Actions

- CI/CD Pipeline from GitHub
- Nicely integrate with GitHub Events
- No need provision resources
- Can keep the build steps in code

Continuous Delivery Pipeline





AWS WAF

Benefits of WAF

- WAF (Web Application Firewall) helps protect Web Apps & APIs against common web exploits
 - OWASP Top 10 Vulnerabilities E.g. SQL injection, Cross Site Scripting etc..
- Create web ACL (Access Control Lists) and associate rules
- Provides automatically updating managed WAF rules
- Provides rate-based rules to block excessive traffic from IP addresses (DDoS mitigation)
- Provides rules to block IP addresses with low reputation (DDoS mitigation)
- Control bot traffic

Building an Online Bookstore

AppSync, Amplify, React & Stripe

Requirements

- Admin login
- Admin panel to add books to the bookstore
- Guest users can view books without login
- Guest users can add books to the cart
- Guest users can register and login to purchase books
- Use Stripe to handle payments

Technologies

- Frontend technologies
 - React
 - Context API
- Backend technologies
 - GraphQL API: AWS AppSync
 - AWS Amplify CLI to provision AWS resources
 - AWS Amplify React library to connect with AWS resources from the React app
 - AWS Amplify UI library to add login screens
- Payment processing:
 - Stripe API