Scaffolding and Deploying Serverless CRUD APIs

Fernando Medina Corey
DATA ENGINEER

@fmc_sea www.fernandomc.com

Outline

Serverless CRUD APIs On AWS

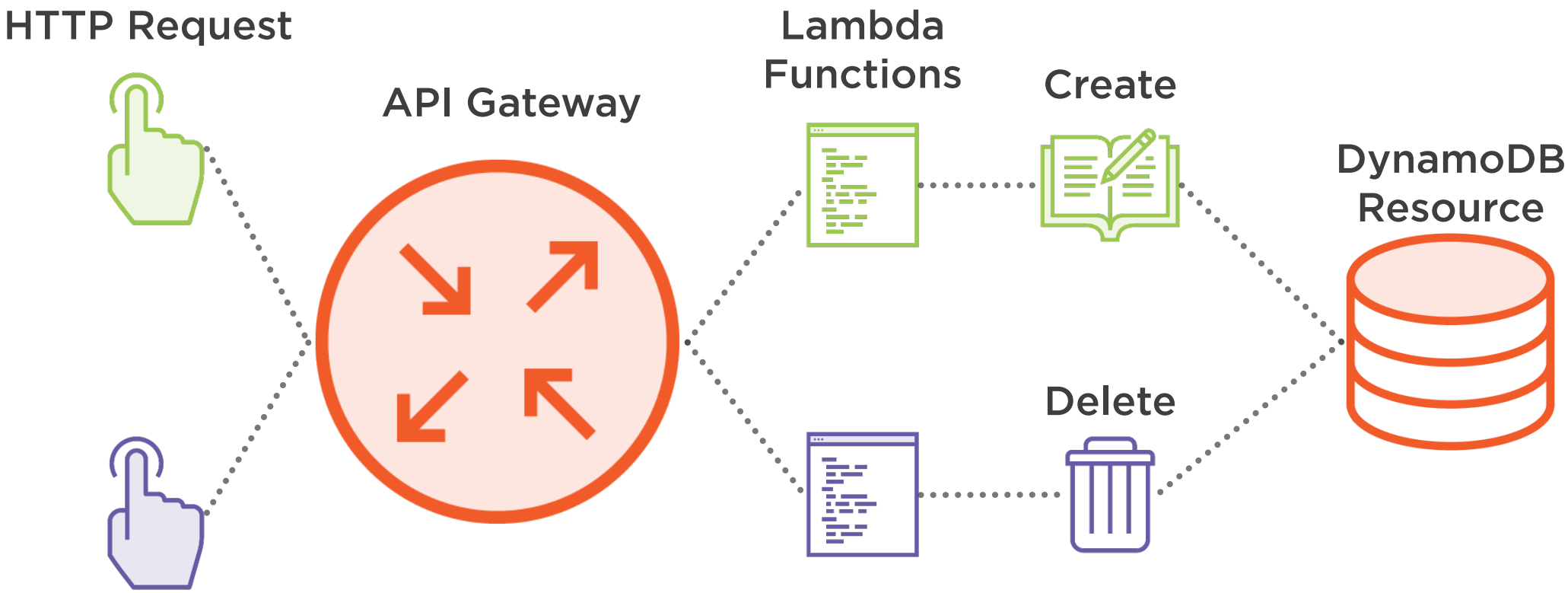
- API Gateway CRUD endpoints
- Serverless databases (DynamoDB)
- Serverless resource integration

Project – Serverless Pet Database

Testing & Debugging

- Tools & strategies

Serverless CRUD Operations on AWS



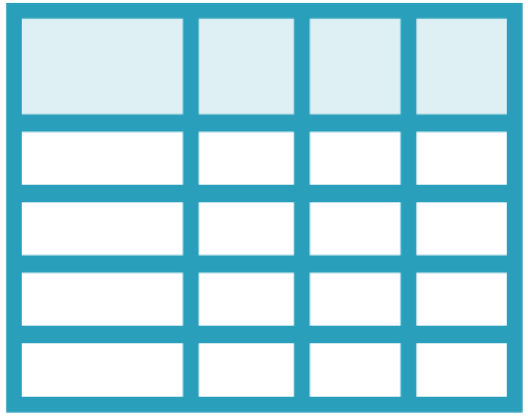
What is DynamoDB?

DynamoDB

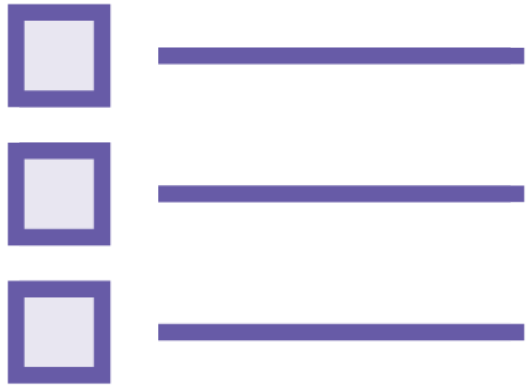
"Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability."

- docs.aws.amazon.com

DynamoDB Structure



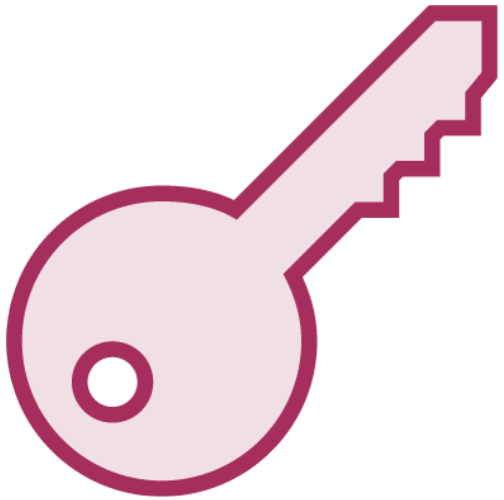
Tables



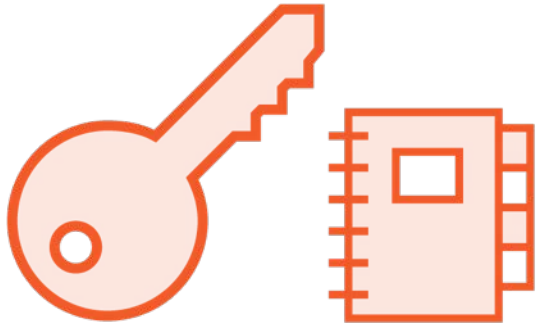
Items

Attributes

DynamoDB Keys



Simple Primary Key
(Partition key only)



Composite Primary Key
(Partition & sort key)

DynamoDB Features

Auto Scaling

Provisioned Throughput

Reserved Capacity

HIPPA Compliance

DynamoDB - Common Gotchas

NoSQL isn't relational

(Table scans are expensive)

Design decisions matter

(Table keys are immutable)

Picking uniform partition keys

(Hot keys at scale cause issues)

DynamoDB Pricing

Writes
($0.47/'WCU'/mo)
25 WCUs/mo - Free

Reads
($0.09/'RCU'/mo)
25 RCUs/mo - Free

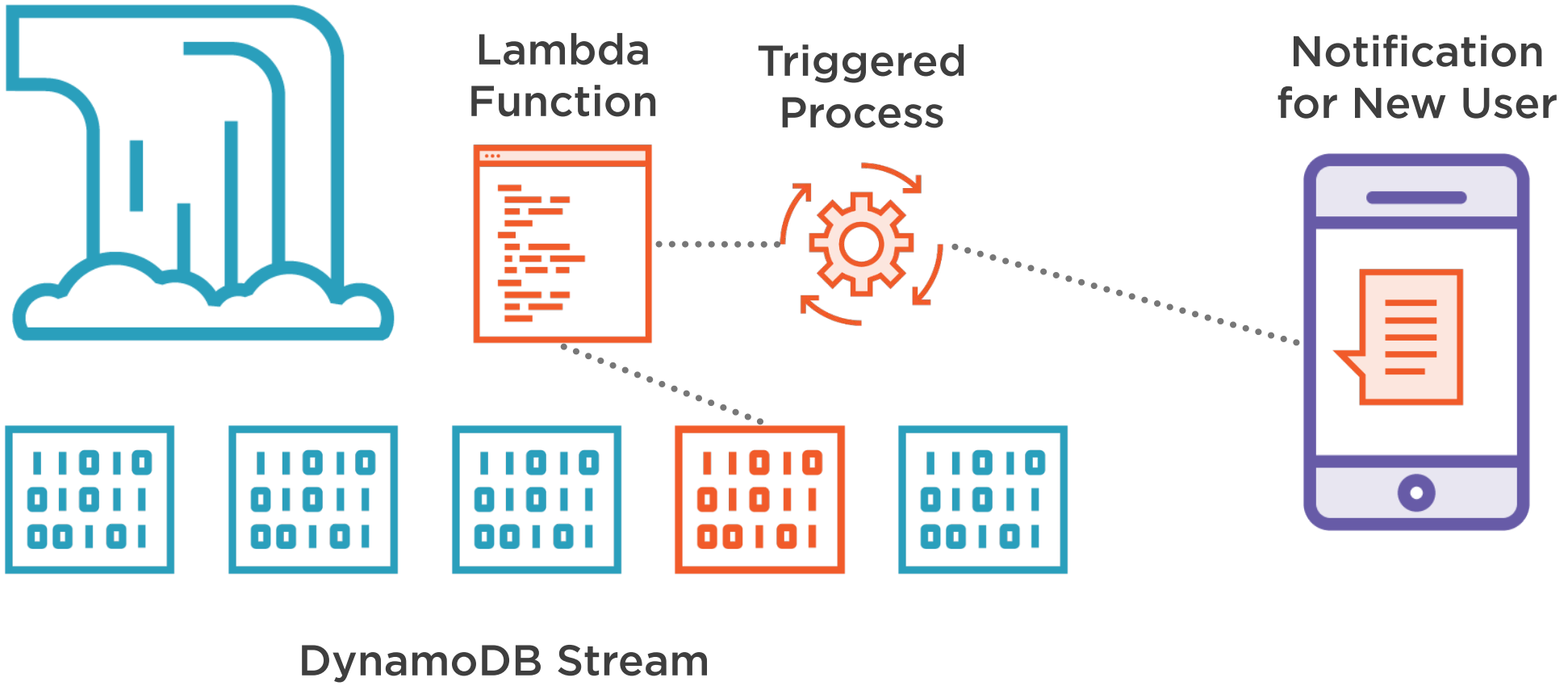
Storage
($0.25/GB/mo)
25 GB/mo - Free

DynamoDB Read Consistency

Eventually Consistent
(Default, cheaper)

Strongly Consistent
(More RCU intensive & more $$)

DynamoDB Streams



Key DynamoDB Takeaways

Free Tier – 200 million request/mo & 25GB storage

Use strong read consistency when appropriate

Design your tables with the future in mind

Take advantage of Dynamo streams

Serverless Configuration – Resources

```
service: serverless-node-rest-api

provider:
  name: aws
  runtime: nodejs6.10
  iamRoleStatements:
    - Effect: Allow
      Action:
        - dynamodb:GetItem
        - dynamodb:PutItem
        - dynamodb:OtherPermissions
      Resource: "arn:aws:dynamodb:oversimplified:PetTable"
```

Serverless Configuration – Resources

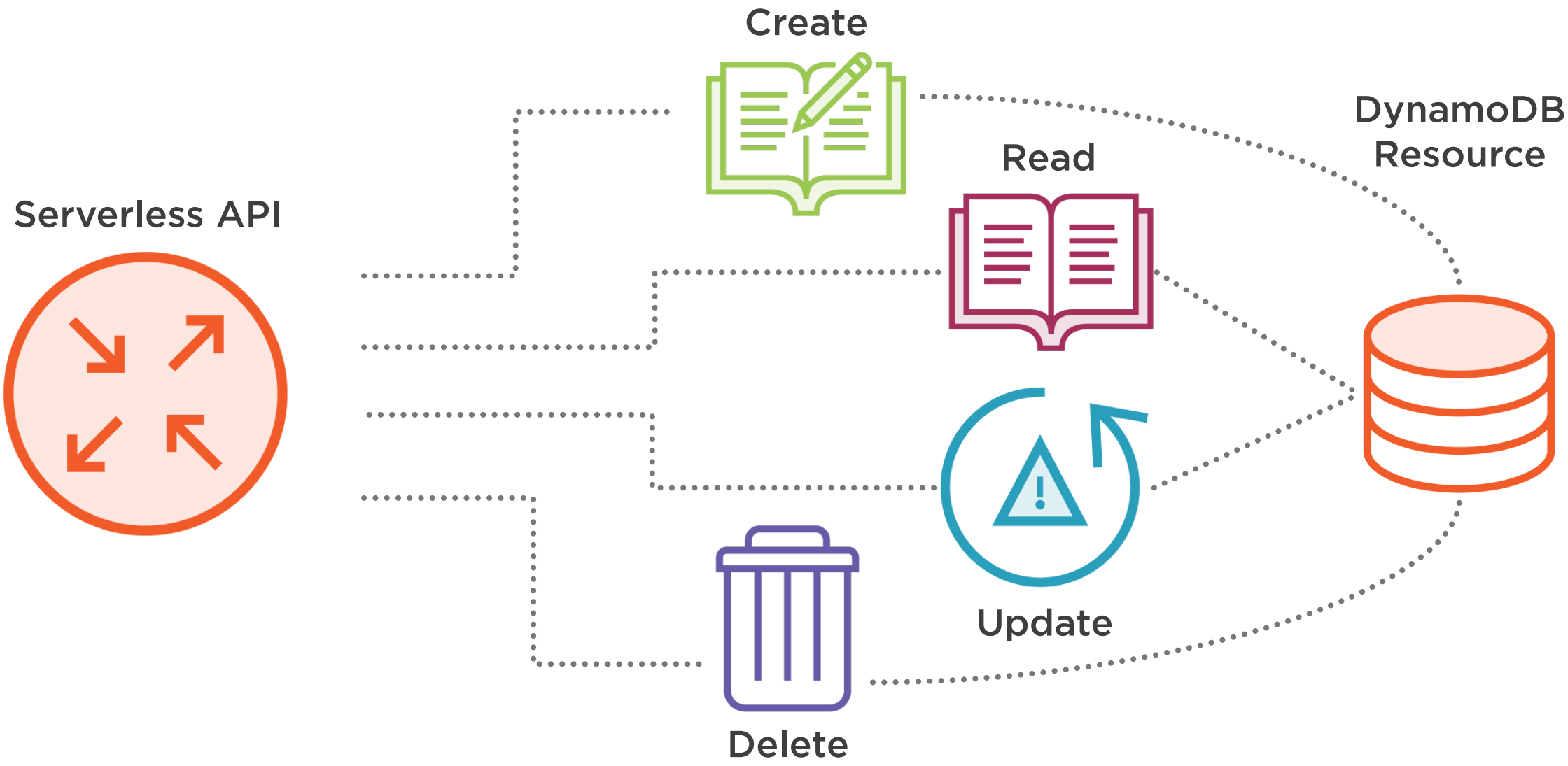
```
# in serverless.yml

resources:
  Resources:
    PetTable:
      Type: AWS::DynamoDB::Table
      DeletionPolicy: Retain
      Properties:
        # Continued
```

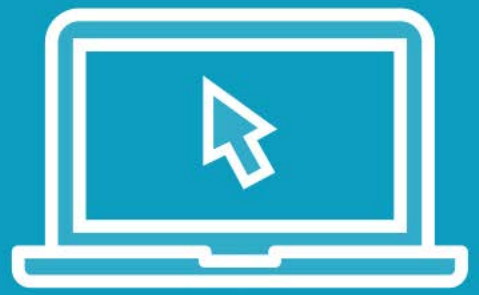
Serverless Configuration – Resources

```
Properties:
  AttributeDefinitions:
    -
      AttributeName: id
      AttributeType: S
  KeySchema:
    -
      AttributeName: id
      KeyType: HASH
  ProvisionedThroughput:
    ReadCapacityUnits: 1
    WriteCapacityUnits: 1
  TableName: PetTable  # Same as in IAMRoleStatement
```

Demo - Serverless CRUD API



Demo

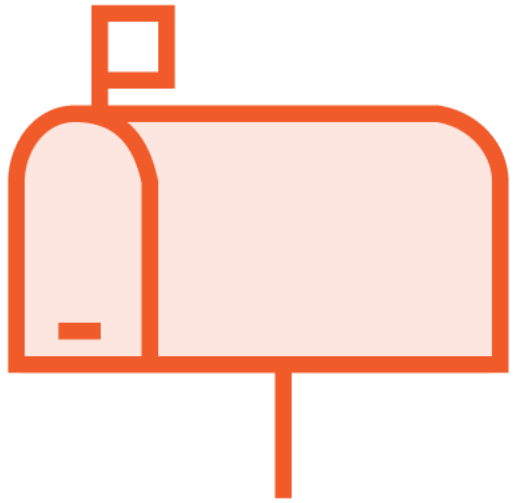


Creating a Serverless Pet Database

- Starting our new project
  - Clone example project
  - Configuration and code review
  - Install requirements
  - Deploy project
- Test endpoints

Serverless Debugging Tools

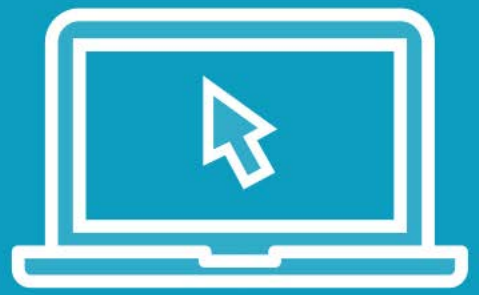
CloudWatch Logs



Postman

Serverless GitHub Issues

Demo



Debugging a Serverless Project

- Modifications to our CRUD API
  - Checking the request/response
    - Curl
    - Postman
  - CloudWatch logs

Review

Serverless Concepts

- Multiple-endpoint Serverless APIs
- Serverless databases

Serverless Skills

- Implemented a CRUD API
  - Lambda, API gateway, DynamoDB
- Deployed a full-stack Serverless App
- Testing & debugging tools
  - CloudWatch, Curl, Postman, GitHub

Coming Up

- Multi-platform Serverless & “Where to go next.”