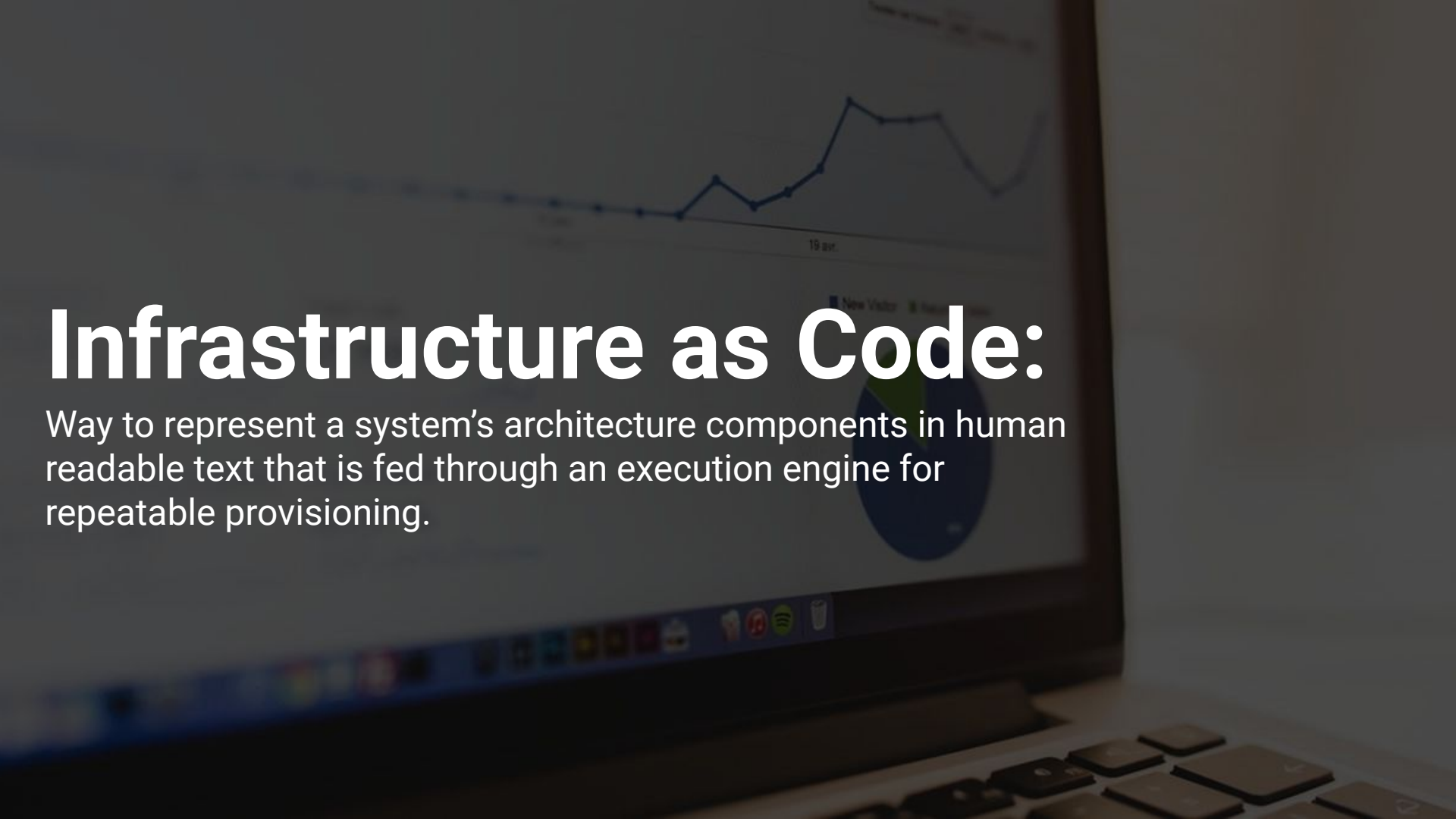
Modern Infrastructure as Code

AWS Cloud Development Kit (CDK)



Infrastructure as Code:

Way to represent a system’s architecture components in human readable text that is fed through an execution engine for repeatable provisioning.

Why Infrastructure as Code (IaC)?

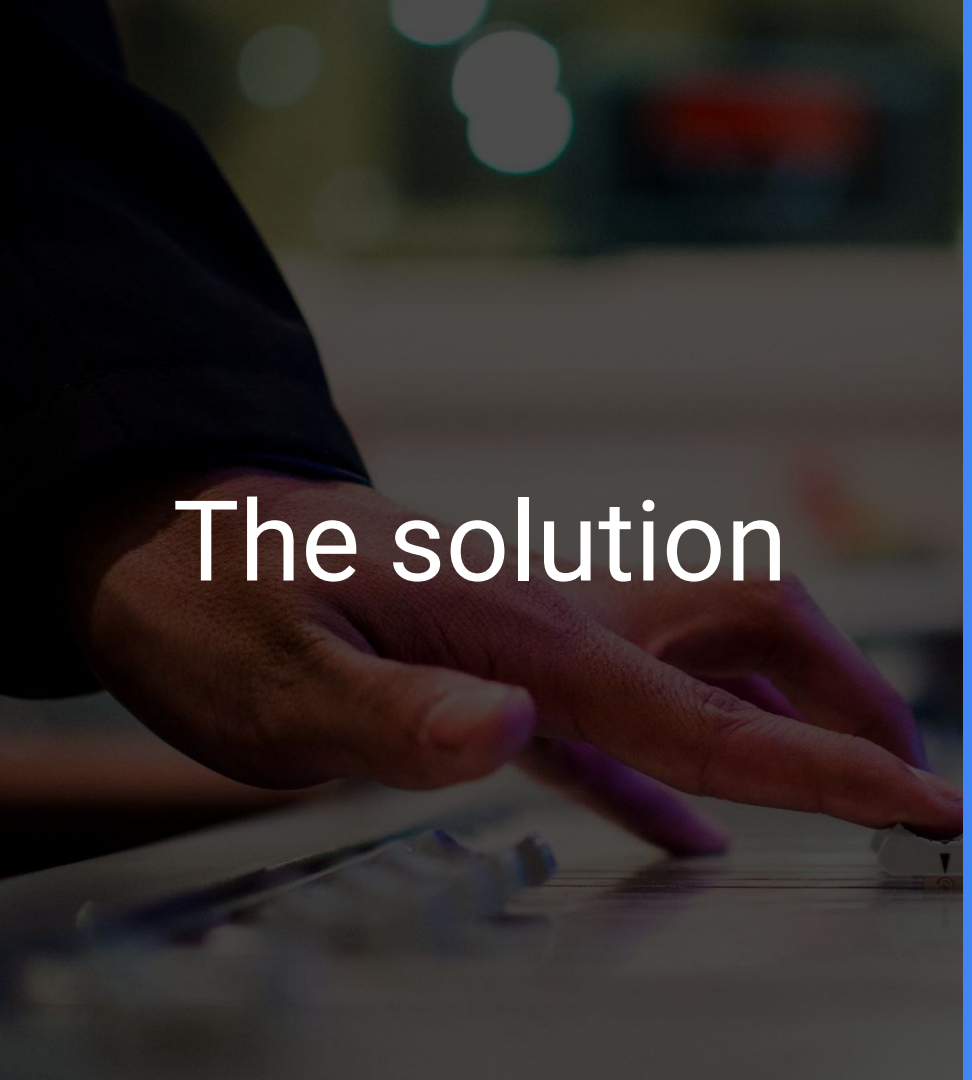
Draws from Software Engineering best practices

- Executable set of instructions as source of truth & documentation
- Consistency via automation
- Accelerated environment provisioning
- Version controlled, auditable and peer reviewed

The problem

Many technologies have successfully executed on the promises of IaC like Terraform, Ansible, Chef, Puppet, AWS Cloud Formation but …

All require a Domain Specific Language (DSL) and specialized skills



The solution

Cloud Development Kit

- Abstraction over Cloud Formation with interfaces in common high-level programming languages

AWS CDK

Technology Components

NodeJS based CLI utility

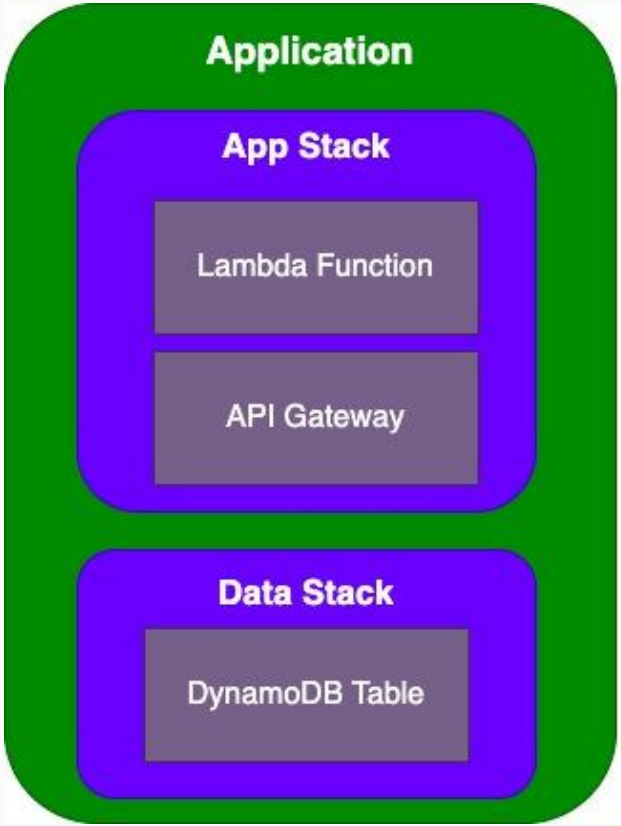
- NodeJS required regardless of language implementation
- Initialize CDK projects
- Synthesize CDK project to Cloud Formation Stacks
- Deploy synthesized stacks

Construct library (TypeScript, C#, Java, Python, Go)

- OO classes representing cloud services & resources
- Typescript as base implementation then translated to other languages via JSII

AWS IaC Abstraction Hierarchy

AWS CDK Class Relationships



Getting Started with CDK

Step 1

Install CDK CLI

```
npm install -g aws-cdk
```

Step 2

Initialize CDK App

```
cdk init app -l typescript
```

Step 3

Define resources as constructs

```
new s3.Bucket(this, "MyBucket");
```

Step 4

Define resources as constructs

```
new S3Bucket(this, "MyBucket");
```

Step 5

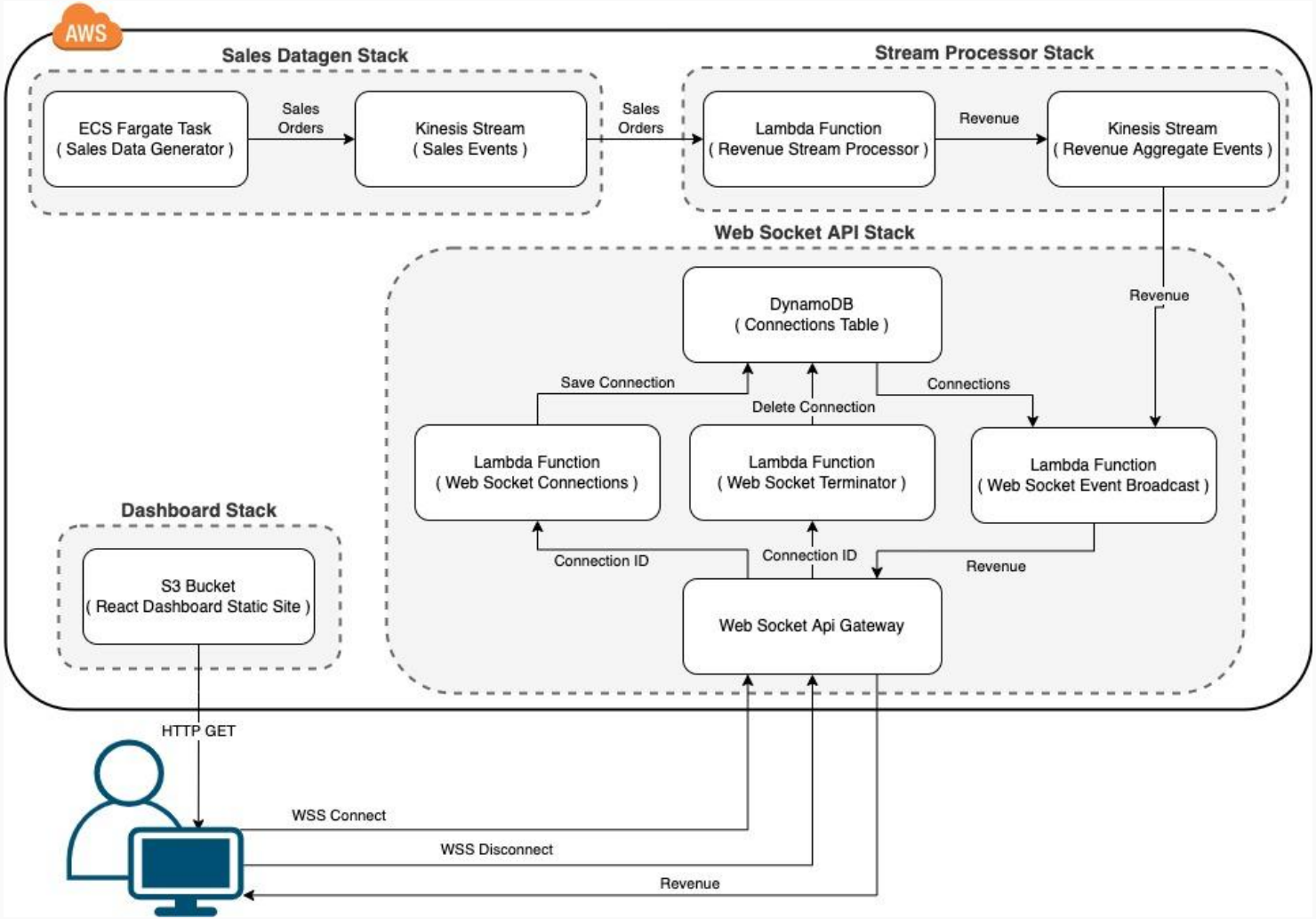
Synthesize to Cloud Formation Stack(s)

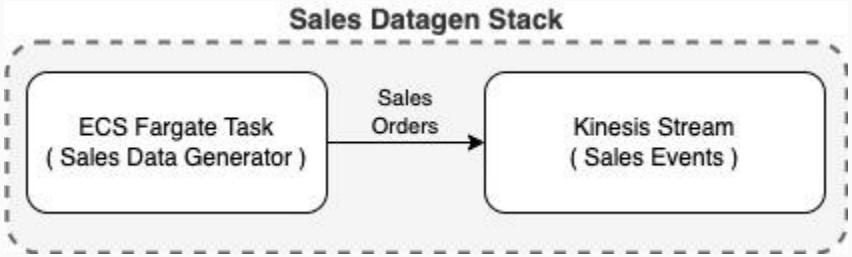
```
cdk synth
```

Step 6

Deploy stacks

```
cdk deploy
```


AWS
Sales Datagen Stack
ECS Fargate Task
(Sales Data Generator)
Sales
Orders
Kinesis Stream
(Sales Events)
Sales
Orders
Stream Processor Stack
Lambda Function
(Revenue Stream Processor)
Revenue
Kinesis Stream
(Revenue Aggregate Events)
Web Socket API Stack
DynamoDB
(Connections Table)
Revenue
Save Connection
Delete Connection
Connections
Lambda Function
(Web Socket Connections)
Lambda Function
(Web Socket Terminator)
Lambda Function
(Web Socket Event Broadcast)
Connection ID
Connection ID
Revenue
Web Socket Api Gateway
Dashboard Stack
S3 Bucket
(React Dashboard Static Site)
HTTP GET
WSS Connect
WSS Disconnect
Revenue

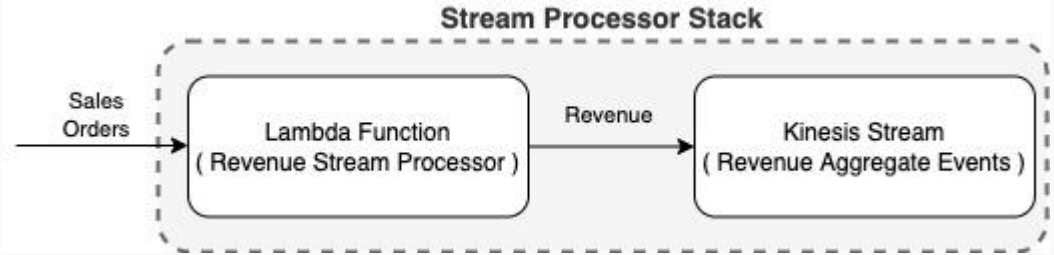


Constructs

- Stream (L2)
- Cluster (L2)

Deployed Resources

- Kinesis Stream
- VPC and Route Table
- 2 Public Subnets with Internet Gateway
- 2 Private Subnets with Nat Gateways
- Fargate Cluster
- ECS Fargate Task Definition
- ECS Fargate Service
- ECS Task with IAM Roles & Policies
- ECR Container Image
- CloudWatch Log Groups

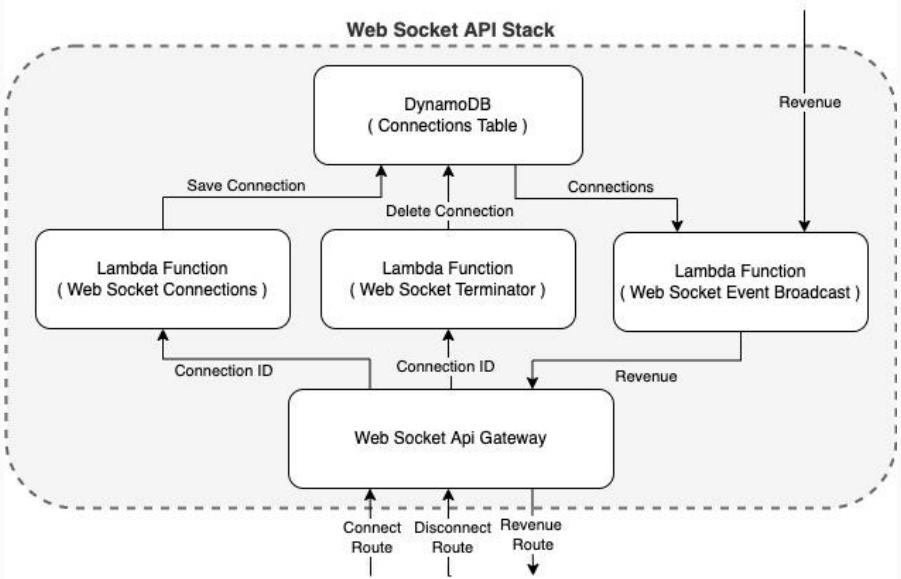


Constructs

- Stream (L2)
- NodejsFunction (L3)

Deployed Resources

- Kinesis Stream
- Lambda Function with IAM Role & Policy
- Lambda / Kinesis Event Source Mapping

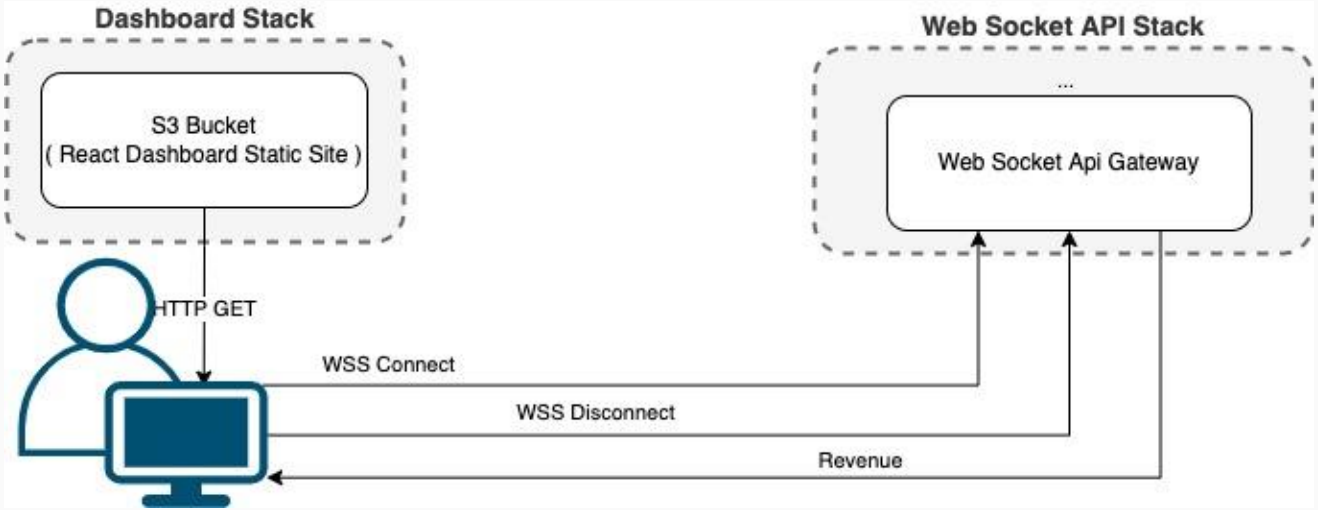


Constructs

- Table (L2)
- NodejsFunction (L3)
- WebsocketApi (Custom L2)
  - CfnApi
  - CfnDeployment
  - CfnStage
  - CfnIntegration
  - CfnRoute

Deployed Resources

- Dynamo DB Table
- Lambda Function with IAM Role & Policy (3)
- Lambda / Kinesis Event Source Mapping
- Api Gateway
- Api Gateway Deployment
- Api Gateway Route
- Api Gateway Integration



Constructs

- Bucket (L2)
- BucketDeployment (L3)

Deployed Resources

- S3 Bucket
- Lambda Function for Deploying Static Site

Q & A / Thank You!

Contact Info

https://www.linkedin.com/in/adammcquistan/

adam.mcquistan@thecodinginterface.com

Slides and Demo Code

https://github.com/amcquistan/nebcode-cdk

AWS CDK Docs

https://docs.aws.amazon.com/cdk/api/v2/