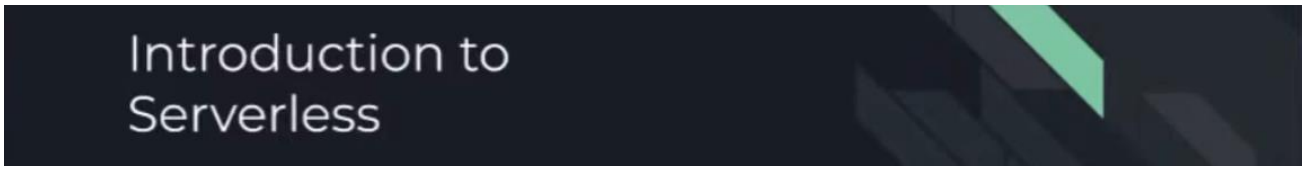
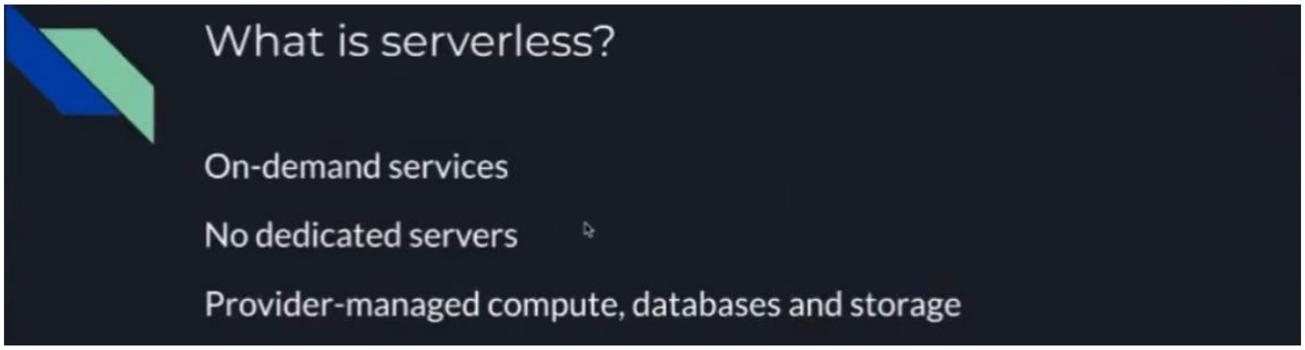


Introduction to Serverless

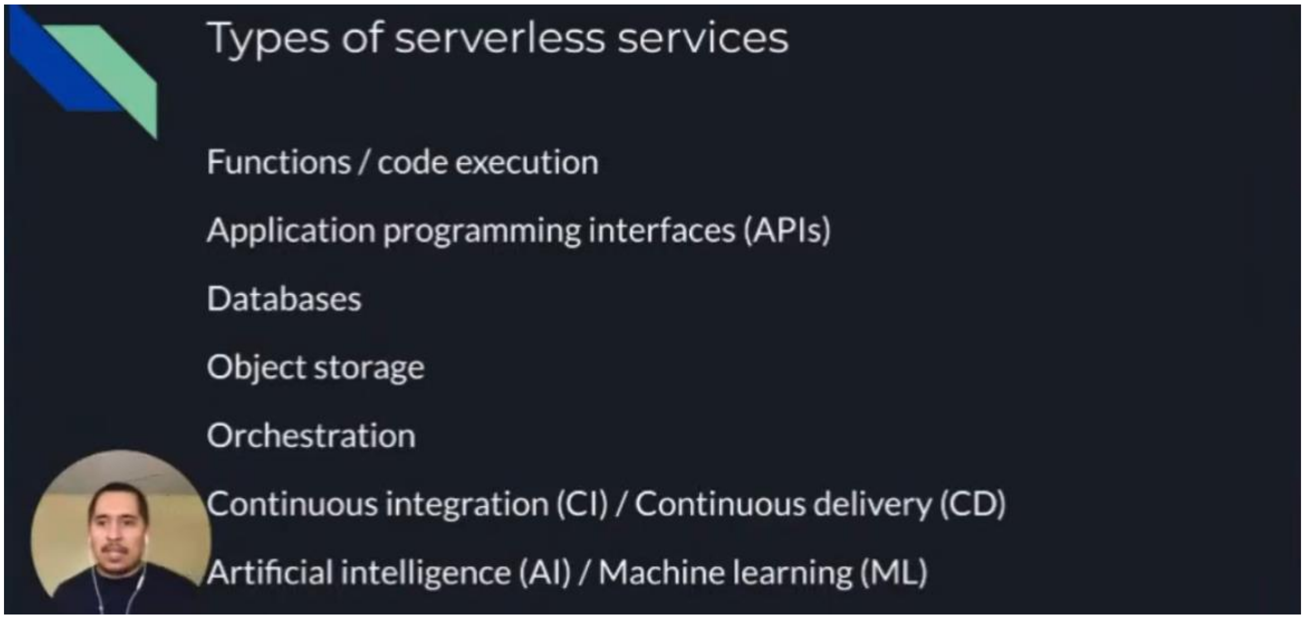


What is serverless?

On-demand services

No dedicated servers

Provider-managed compute, databases and storage



Types of serverless services

Functions / code execution

Application programming interfaces (APIs)

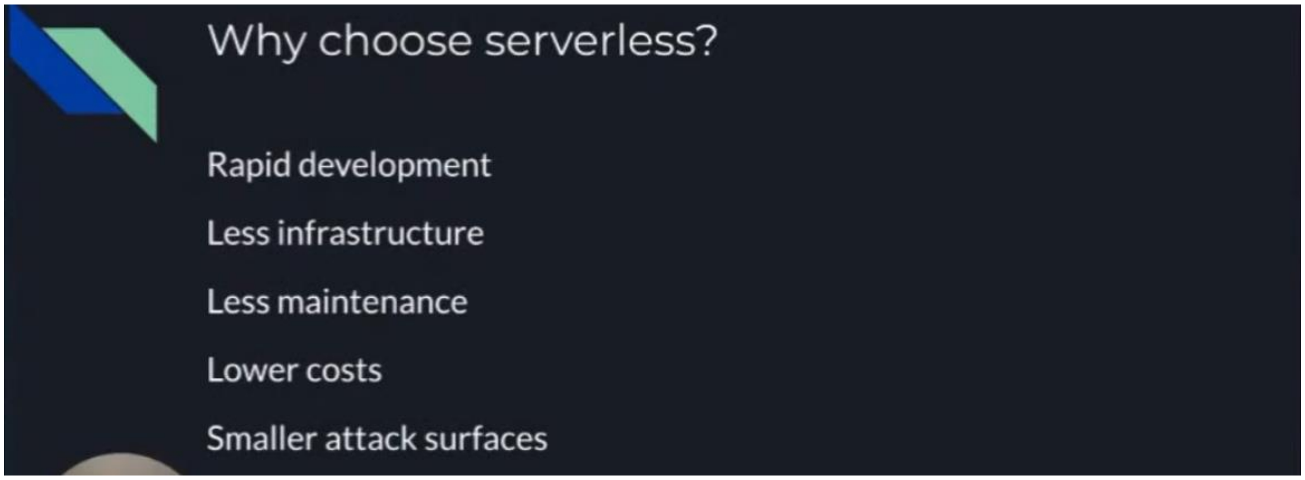
Databases

Object storage

Orchestration

Continuous integration (CI) / Continuous delivery (CD)

Artificial intelligence (AI) / Machine learning (ML)



Why choose serverless?

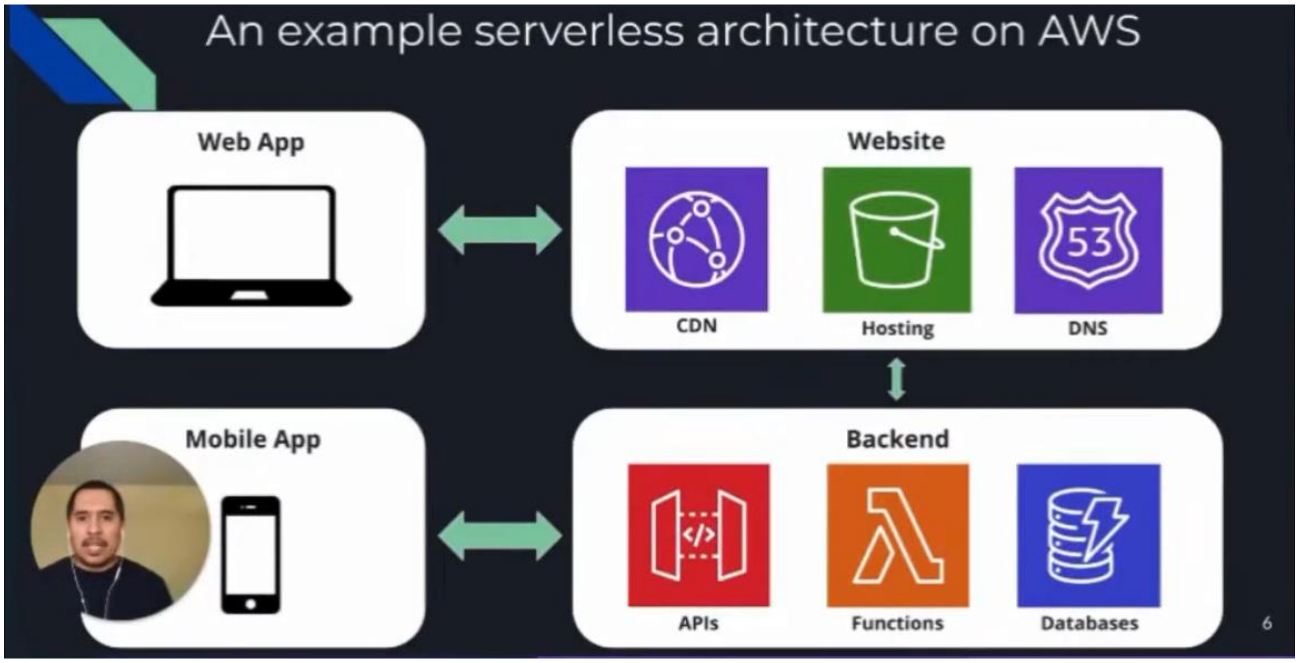
Rapid development

Less infrastructure

Less maintenance

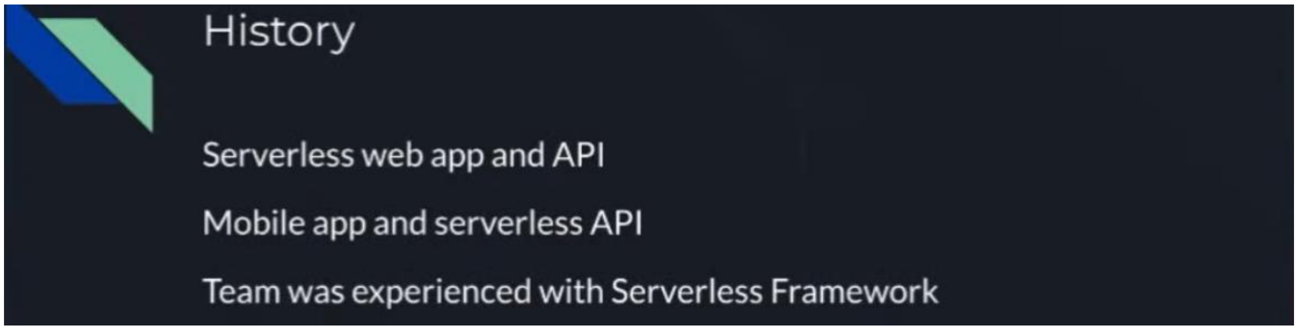
Lower costs

Smaller attack surfaces



An example serverless architecture on AWS

Using Serverless Framework

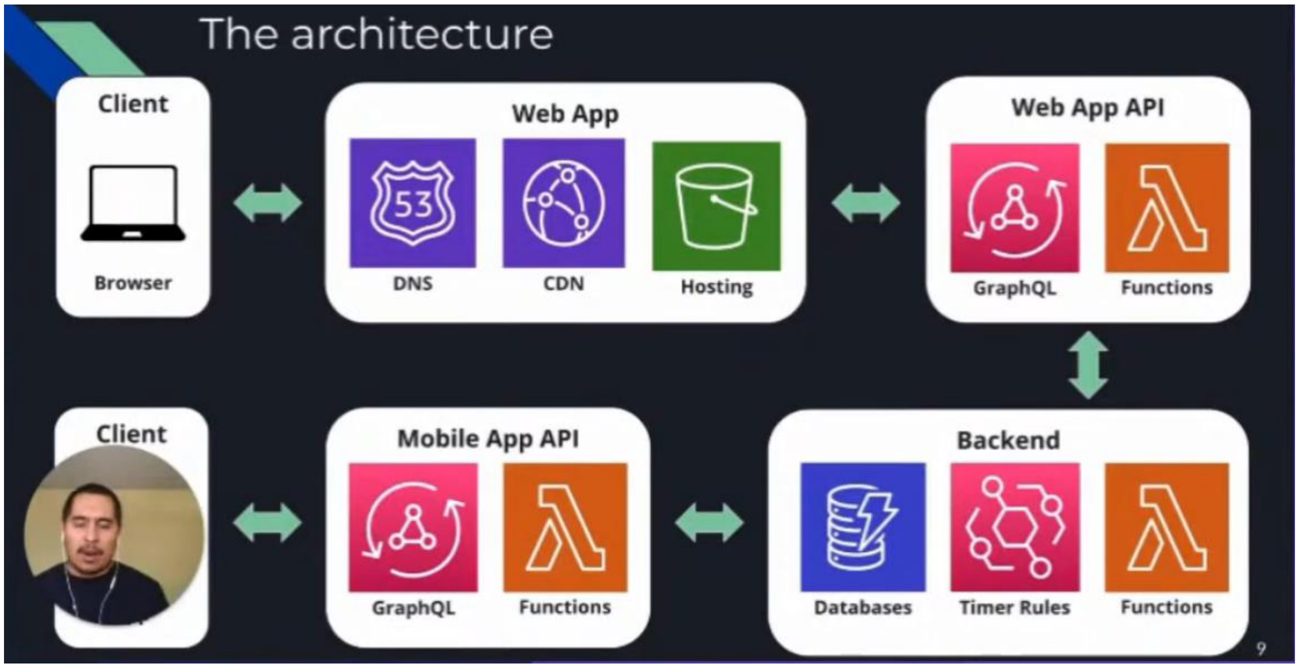


History

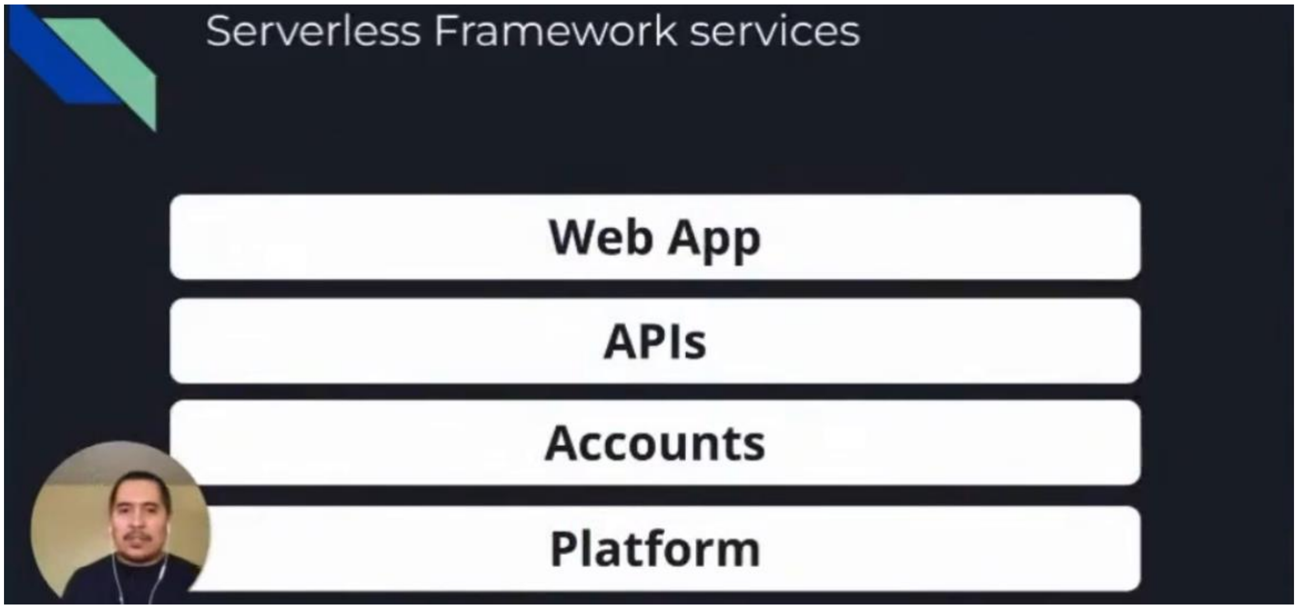
Serverless web app and API

Mobile app and serverless API

Team was experienced with Serverless Framework

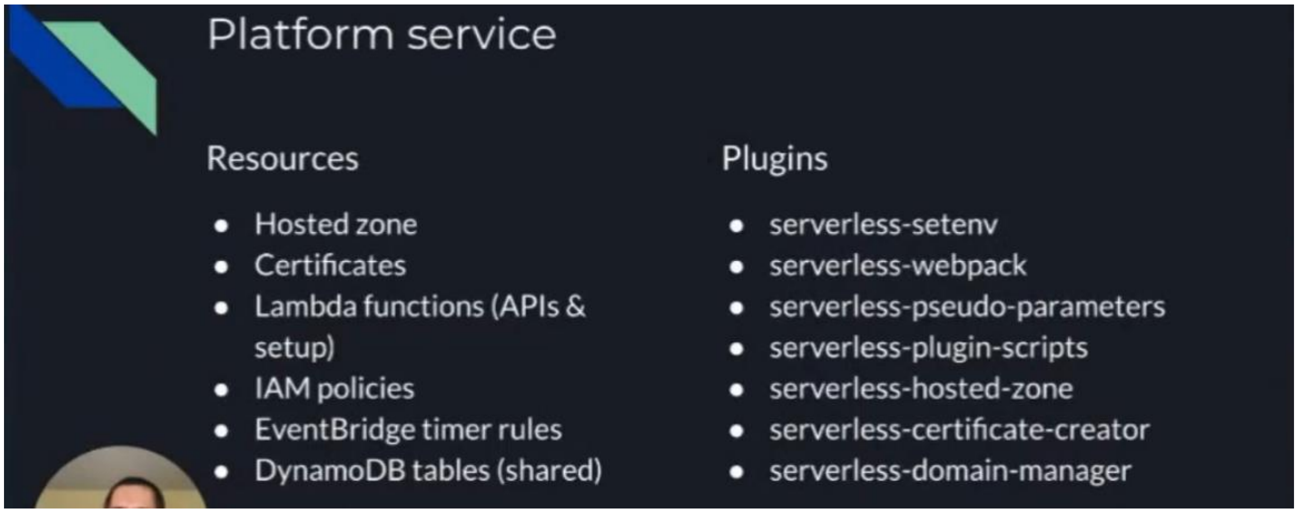


The architecture



Serverless Framework services

- Web App
- APIs
- Accounts
- Platform



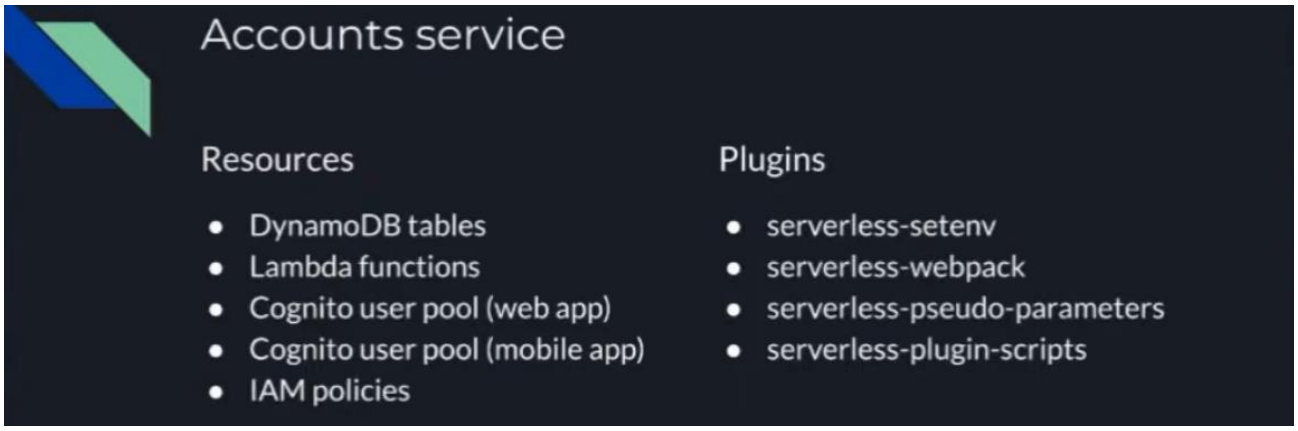
Platform service

Resources

- Hosted zone
- Certificates
- Lambda functions (APIs & setup)
- IAM policies
- EventBridge timer rules
- DynamoDB tables (shared)

Plugins

- serverless-setenv
- serverless-webpack
- serverless-pseudo-parameters
- serverless-plugin-scripts
- serverless-hosted-zone
- serverless-certificate-creator
- serverless-domain-manager



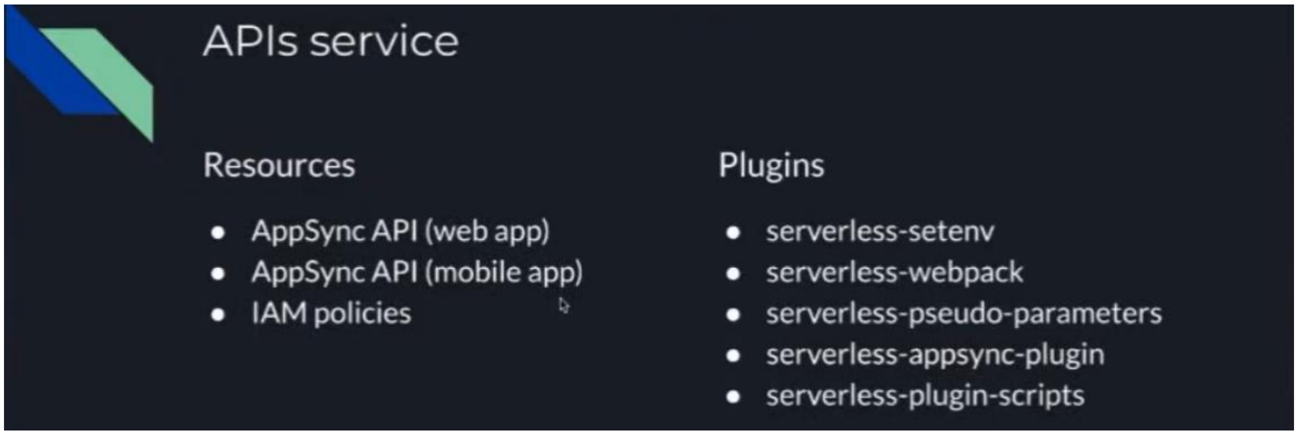
Accounts service

Resources

- DynamoDB tables
- Lambda functions
- Cognito user pool (web app)
- Cognito user pool (mobile app)
- IAM policies

Plugins

- serverless-setenv
- serverless-webpack
- serverless-pseudo-parameters
- serverless-plugin-scripts



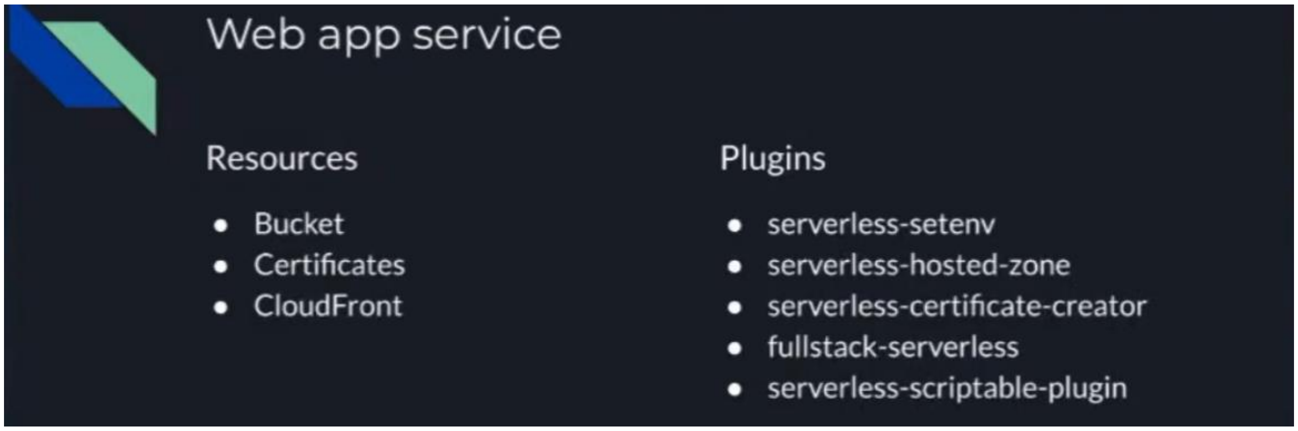
APIs service

Resources

- AppSync API (web app)
- AppSync API (mobile app)
- IAM policies

Plugins

- serverless-setenv
- serverless-webpack
- serverless-pseudo-parameters
- serverless-appsync-plugin
- serverless-plugin-scripts



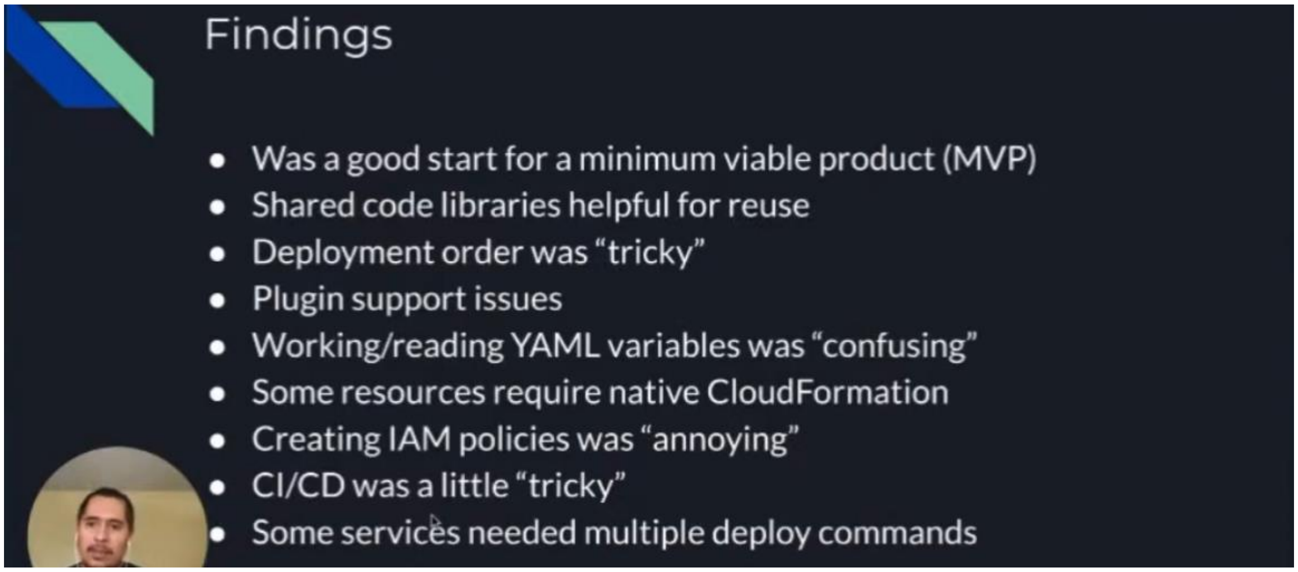
Web app service

Resources

- Bucket
- Certificates
- CloudFront

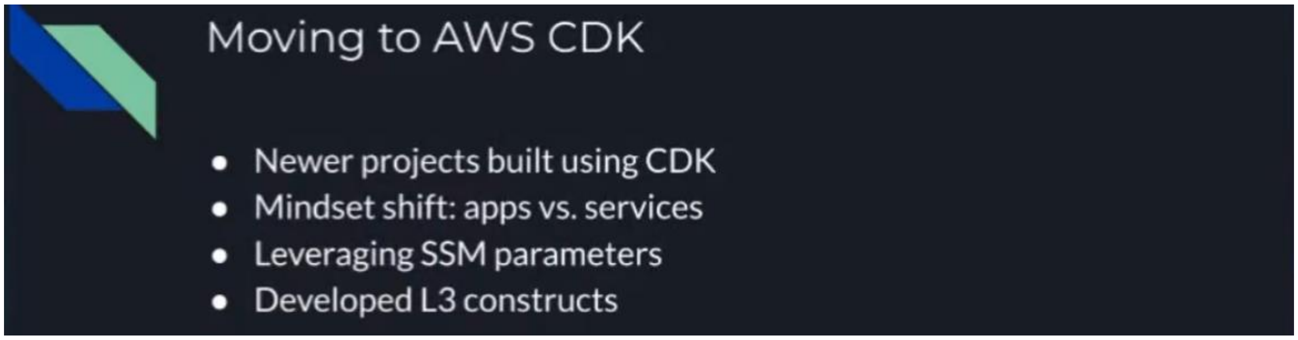
Plugins

- serverless-setenv
- serverless-hosted-zone
- serverless-certificate-creator
- fullstack-serverless
- serverless-scriptable-plugin



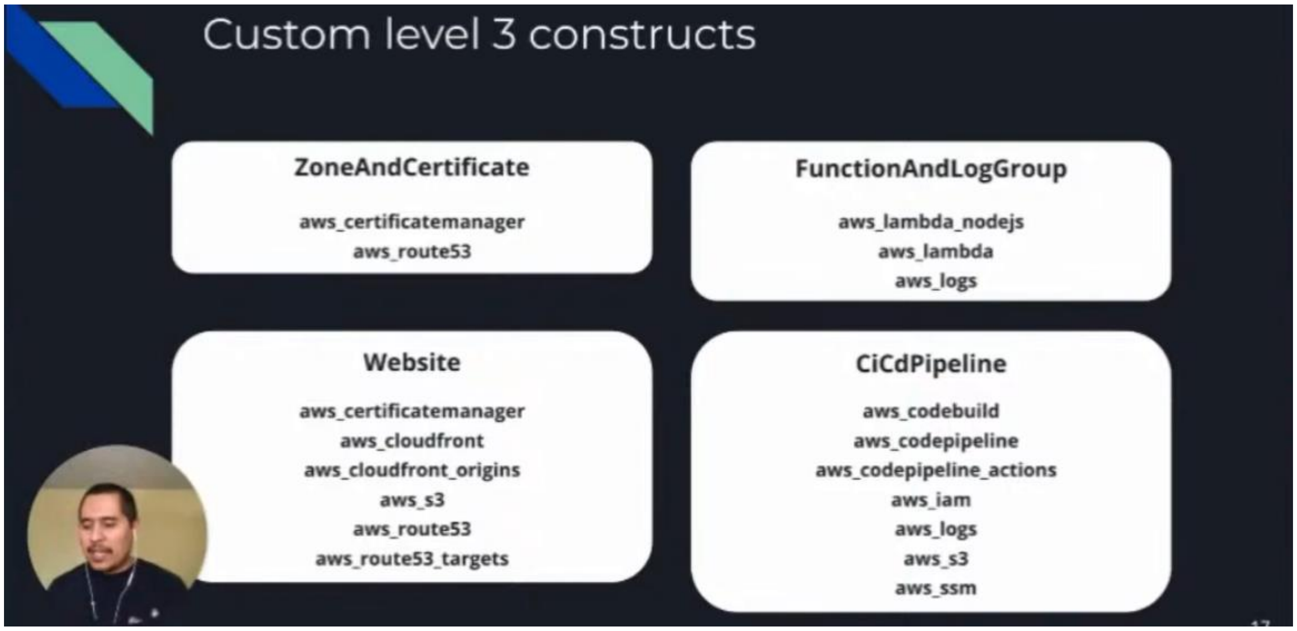
Findings

- Was a good start for a minimum viable product (MVP)
- Shared code libraries helpful for reuse
- Deployment order was "tricky"
- Plugin support issues
- Working/reading YAML variables was "confusing"
- Some resources require native CloudFormation
- Creating IAM policies was "annoying"
- CI/CD was a little "tricky"
- Some services needed multiple deploy commands



Moving to AWS CDK

- Newer projects built using CDK
- Mindset shift: apps vs. services
- Leveraging SSM parameters
- Developed L3 constructs



Custom level 3 constructs

ZoneAndCertificate

aws_certificatemanager
aws_route53

FunctionAndLogGroup

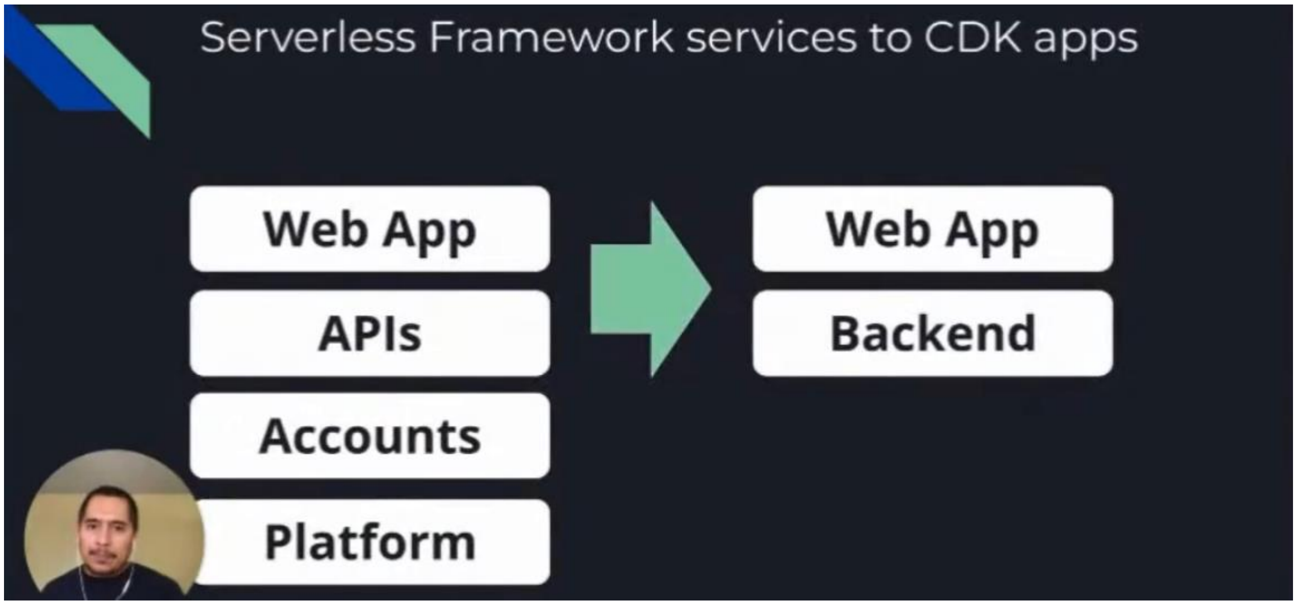
aws_lambda_nodejs
aws_lambda
aws_logs

Website

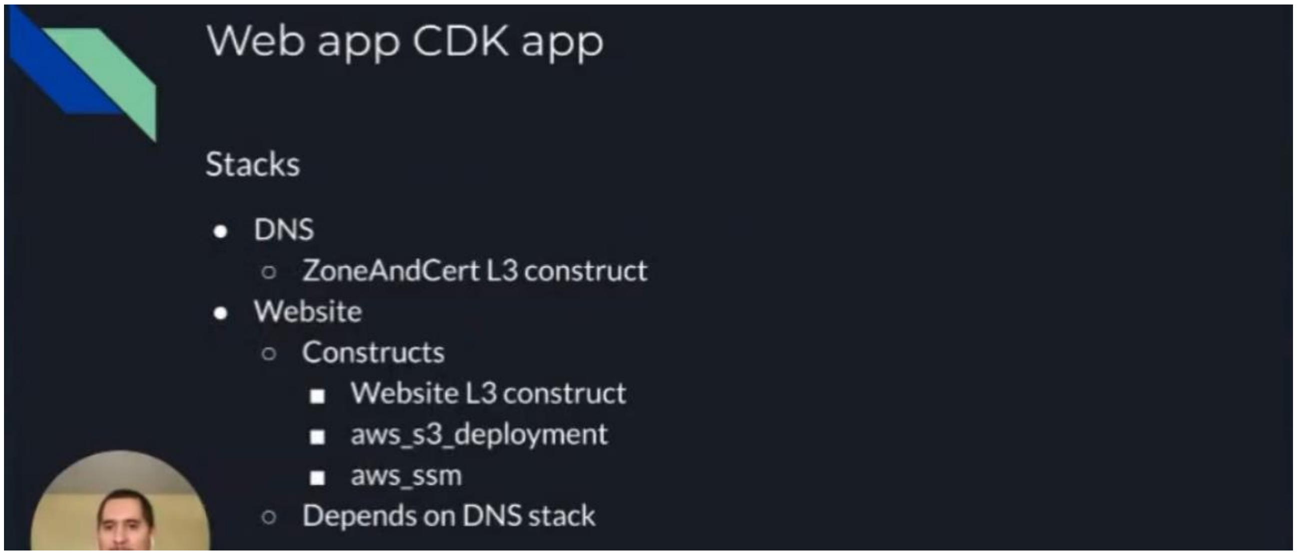
aws_certificatemanager
aws_cloudfront
aws_cloudfront_origins
aws_s3
aws_route53
aws_route53_targets

CiCdPipeline

aws_codebuild
aws_codepipeline
aws_codepipeline_actions
aws_iam
aws_logs
aws_s3
aws_ssm



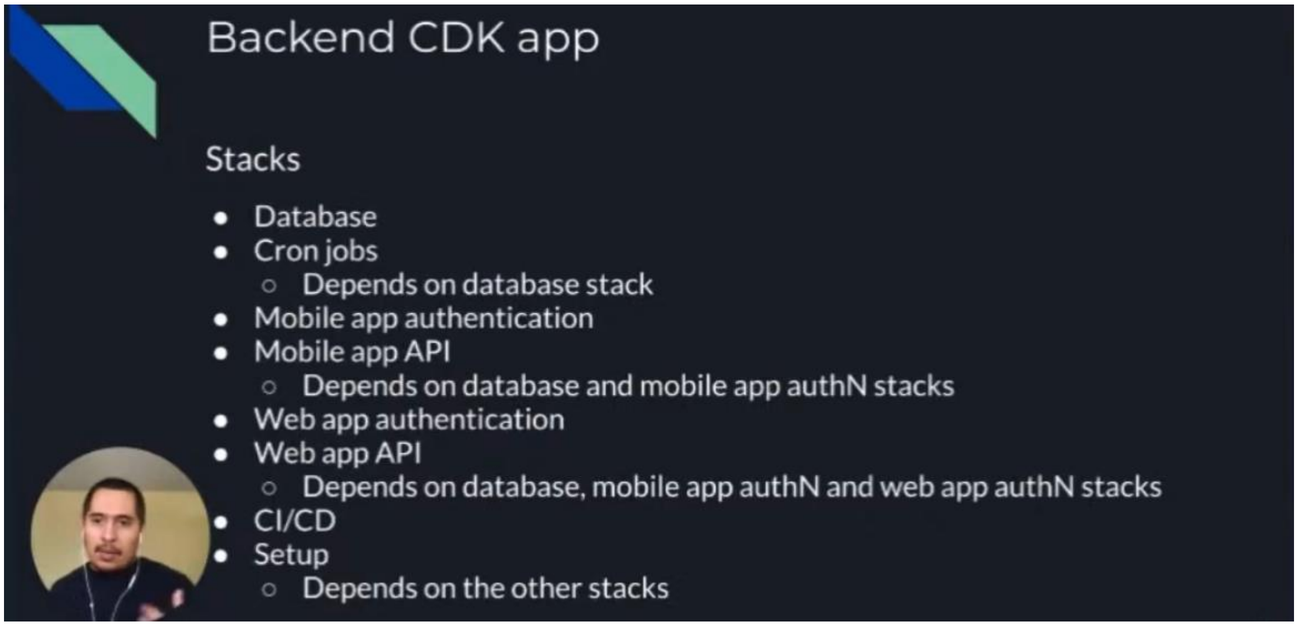
Serverless Framework services to CDK apps



Web app CDK app

Stacks

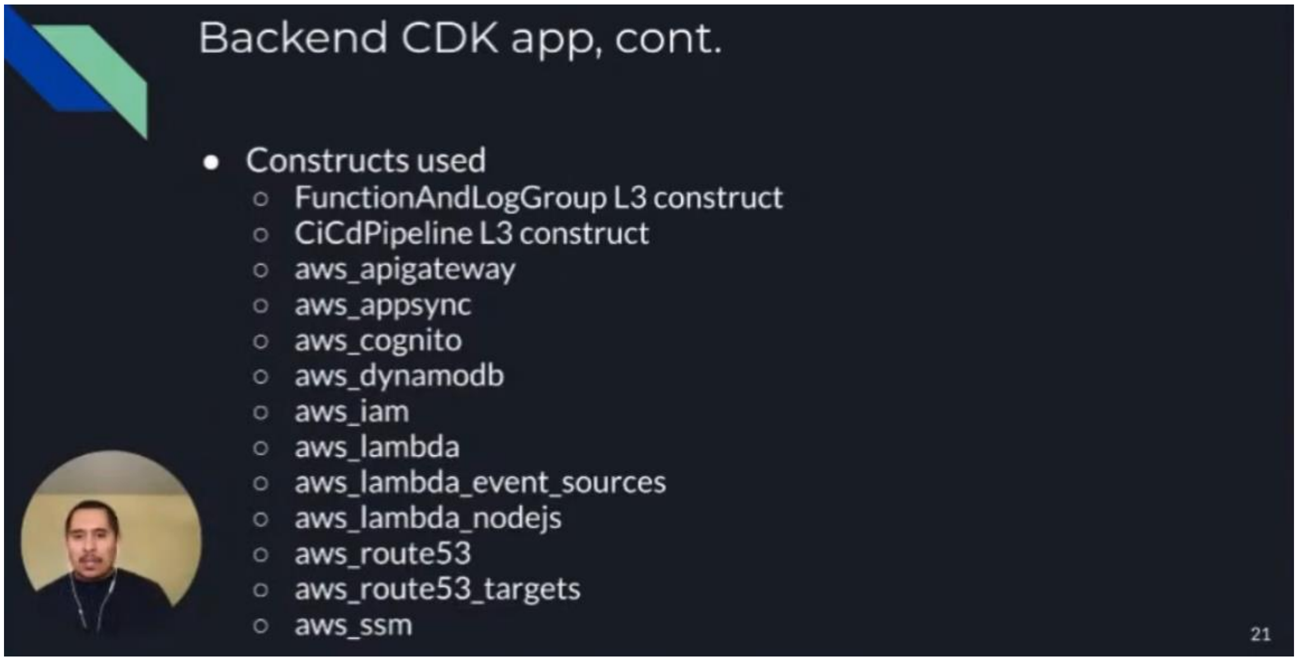
- DNS
  - ZoneAndCert L3 construct
- Website
  - Constructs
    - Website L3 construct
    - aws_s3_deployment
    - aws_ssm
  - Depends on DNS stack



Backend CDK app

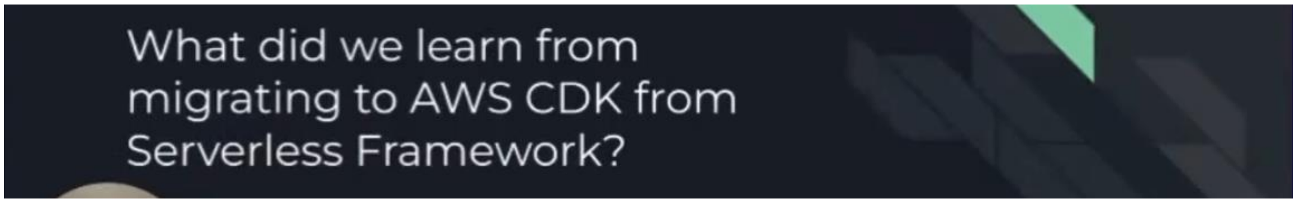
Stacks

- Database
- Cron jobs
  - Depends on database stack
- Mobile app authentication
- Mobile app API
  - Depends on database and mobile app authN stacks
- Web app authentication
- Web app API
  - Depends on database, mobile app authN and web app authN stacks
- CI/CD
- Setup
  - Depends on the other stacks

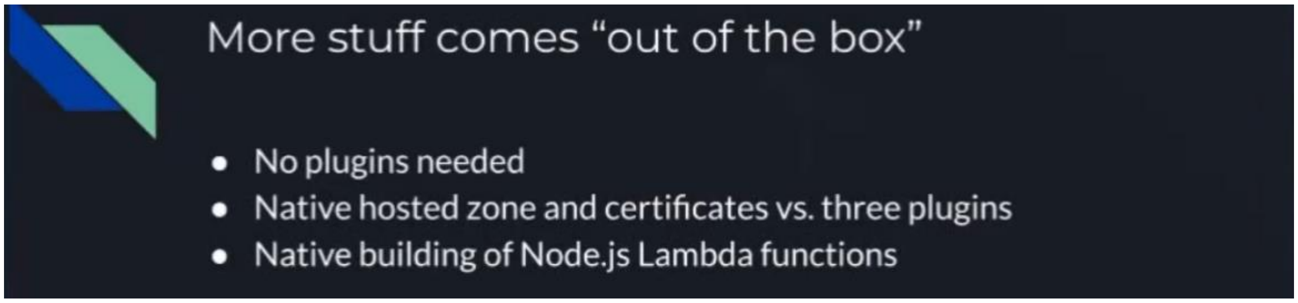


Backend CDK app, cont.

- Constructs used
  - FunctionAndLogGroup L3 construct
  - CiCdPipeline L3 construct
  - aws_apigateway
  - aws_appsync
  - aws_cognito
  - aws_dynamodb
  - aws_iam
  - aws_lambda
  - aws_lambda_event_sources
  - aws_lambda_nodejs
  - aws_route53
  - aws_route53_targets
  - aws_ssm

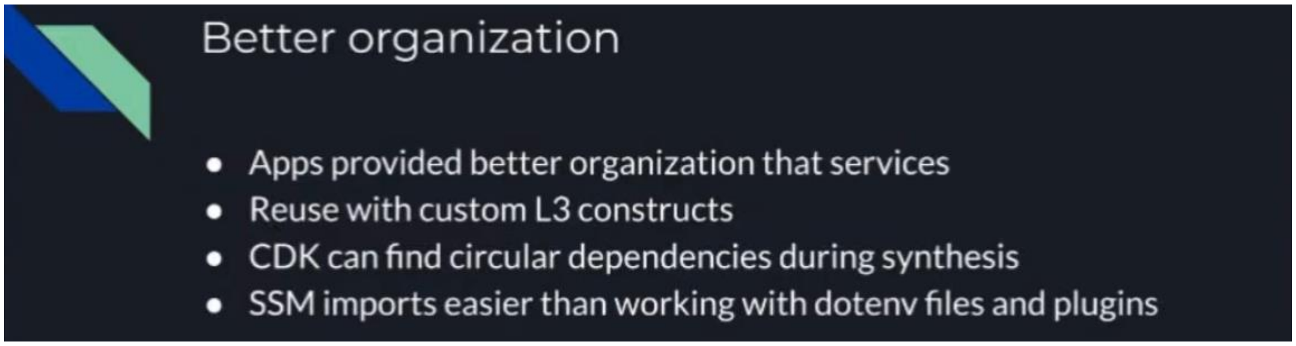


What did we learn from migrating to AWS CDK from Serverless Framework?



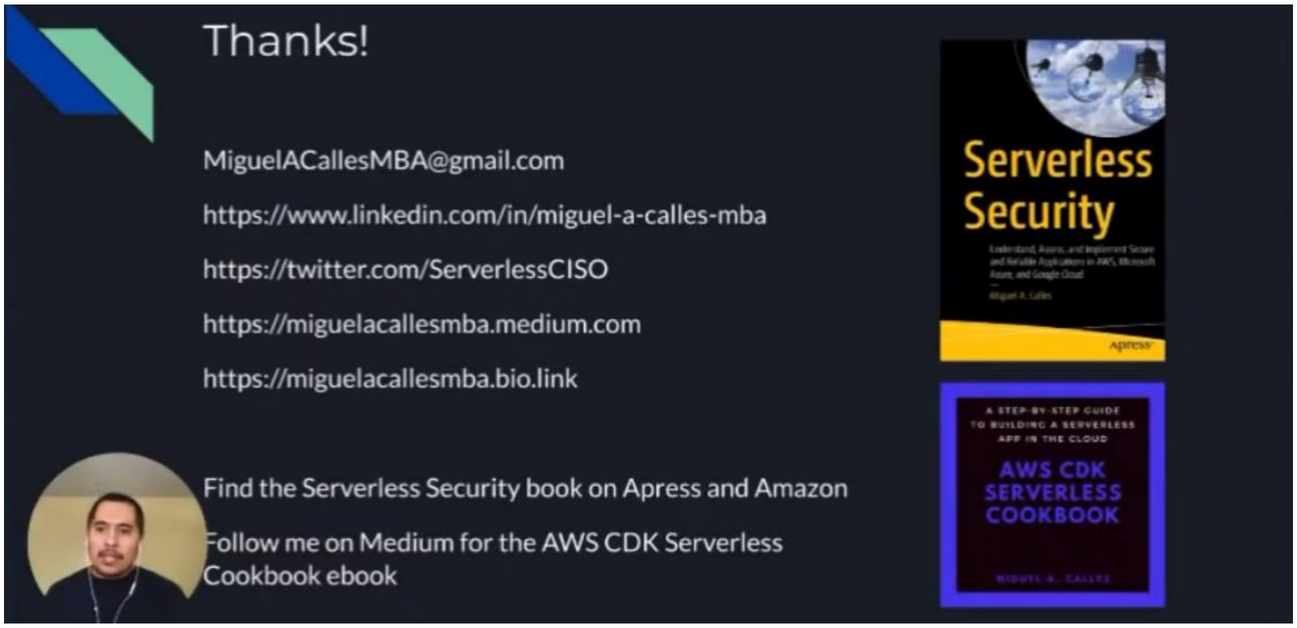
More stuff comes "out of the box"

- No plugins needed
- Native hosted zone and certificates vs. three plugins
- Native building of Node.js Lambda functions



Better organization

- Apps provided better organization that services
- Reuse with custom L3 constructs
- CDK can find circular dependencies during synthesis
- SSM imports easier than working with dotenv files and plugins



Thanks!

MiguelACallesMBA@gmail.com

https://www.linkedin.com/in/miguel-a-calles-mba

https://twitter.com/ServerlessCISO

https://miguelacallesmba.medium.com

https://miguelacallesmba.bio.link

Find the Serverless Security book on Apress and Amazon

Follow me on Medium for the AWS CDK Serverless Cookbook ebook

Serverless Security
Apress
A STEP-BY-STEP GUIDE TO BUILDING A SERVERLESS APP IN THE CLOUD
AWS CDK SERVERLESS COOKBOOK
MIGUEL A. CALLES