







focusOtter /
cdk-appsync-guests



[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)



A fully deployable solution showcasing both AppSync API access and Guest IAM access.

aws.amazon.com/blogs/mobile/secure-aws-appsync-with-api-keys-using-the-aws-cdk/


☆ 3 stars 🍴 0 forks 👁 2 watching 📈 Activity

🌐 Public repository

 main ▾

...

🔗 Branches 🏷 Tags

 **mtliendo** update content ... on Nov 9, 2022 ⌚ 11

[View code](#)

Secure AWS AppSync with API Keys using the AWS CDK

(checkout the branches for the IAM permission setup)

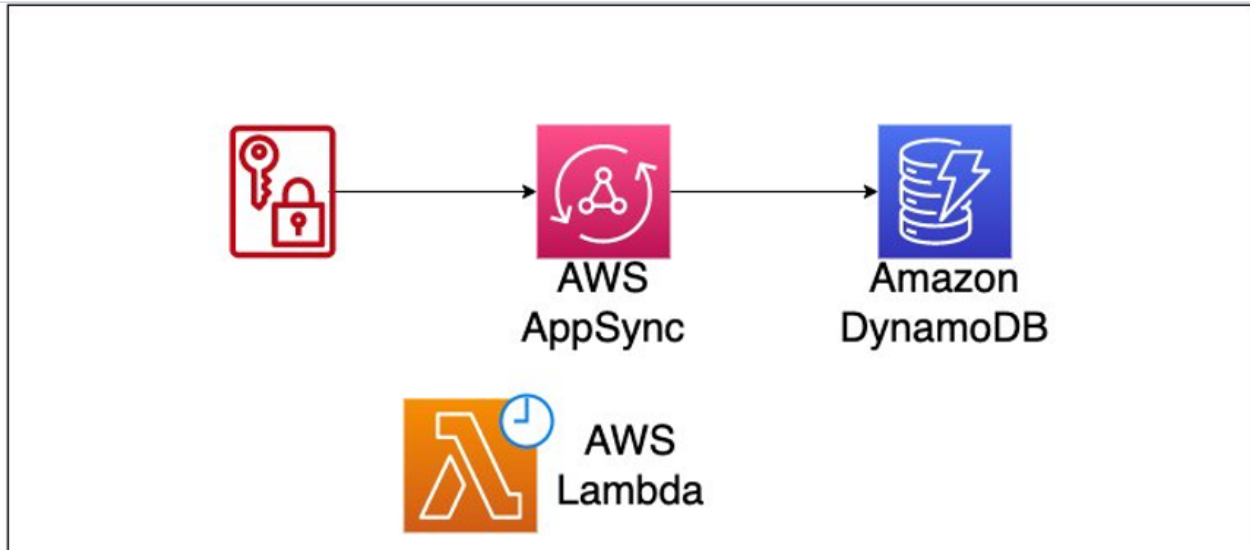
This repo walks through the steps needed to get setup an AppSync API that is protected with an API Key.

```
// valid, but simplified
const api = new GraphQLApi(this, 'User API', {
  name: 'User API',
  schema: Schema.fromAsset(path.join(__dirname, 'schema.graphql')),
  authorizationConfig: {
    defaultAuthorization: {
      authorizationType: AuthorizationType.API_KEY,
    },
  },
});
```



```
} ,  
  }
```

☰ README.md



Content Channels [↗](#)

- AWS blog post: [Secure AWS AppSync with API Keys using the AWS CDK](#)
- Focus Otter blog post: [The case for fullstack teams having a dedicated frontend and backend](#)
- Focus Otter YouTube Video: [The complete beginners guide to creating fullstack apps using the AWS CDK](#)
- TikTok: [Understanding the different parts of AWS Amplify](#)
- Copy/paste snip pic: [link to view](#)

Project Overview [↗](#)

The core of the appl

The deployed project is meant to work with a frontend (see link to frontend repo below), thereby creating a fullstack application. In addition to an AppSync API, a DynamoDB table is created to hold `User` data and a Lambda function is created to populate the table on a schedule.

On the frontend, use of the AWS Amplify JS libraries are used to connect our frontend to our backend by means of the `Amplify.configure` method (sample data configs are used):

```
Amplify.configure({  
  aws_project_region: 'us-east-1',  
  aws_appsync_graphqlEndpoint:  
    'https://c4wds3boinhrdemdnqkt5uztny.appsync-api.us-east-1.amazonaws.',  
  aws_appsync_region: 'us-east-1',  
  aws_appsync_authenticationType: 'API_KEY',  
  aws_appsync_apiKey: 'da2-ze45yo5nm5dttnsvkyoxwbbvq',  
})
```

With our frontend configured to work with our backend, and our Lambda function seeding our database, the frontend will display user data styled with the AWS [Amplify UI Components](#)

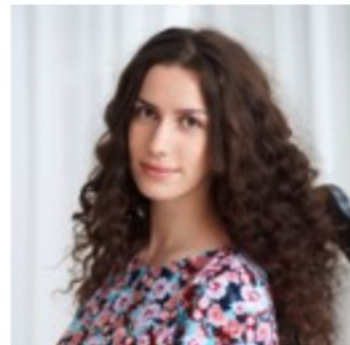
User Profile List



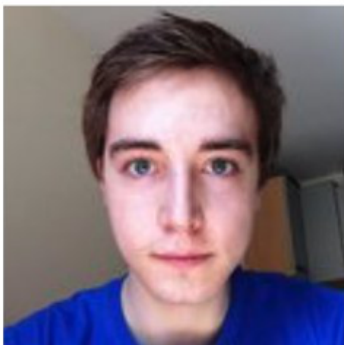
Lucas Chavarría



Geoff Ferguson



Yashika Nand



Leandro Clement



Shane Caldwell

< 1 2 >

Note the frontend repo also has a dedicated branch to show the *slight* change needed for IAM authorization.

Useful commands [↗](#)

- `npm run build` compile typescript to js
- `npm run watch` watch for changes and compile
- `npm run test` perform the jest unit tests
- `cdk deploy` deploy this stack to your default AWS account/region
- `cdk diff` compare deployed stack with current state
- `cdk synth` emits the synthesized CloudFormation template

Releases

No releases published

Packages

No packages published

Languages

