URL Shortener on AWS

Rachel Chen

Very easy to set up!

Follow along here:

https://s.anti.works/guide

Just kidding 😁

Github Repo:

https://github.com/zllovesuki/shortener-on-lambda

Alternatively:

https://s.anti.works/gh

Why?

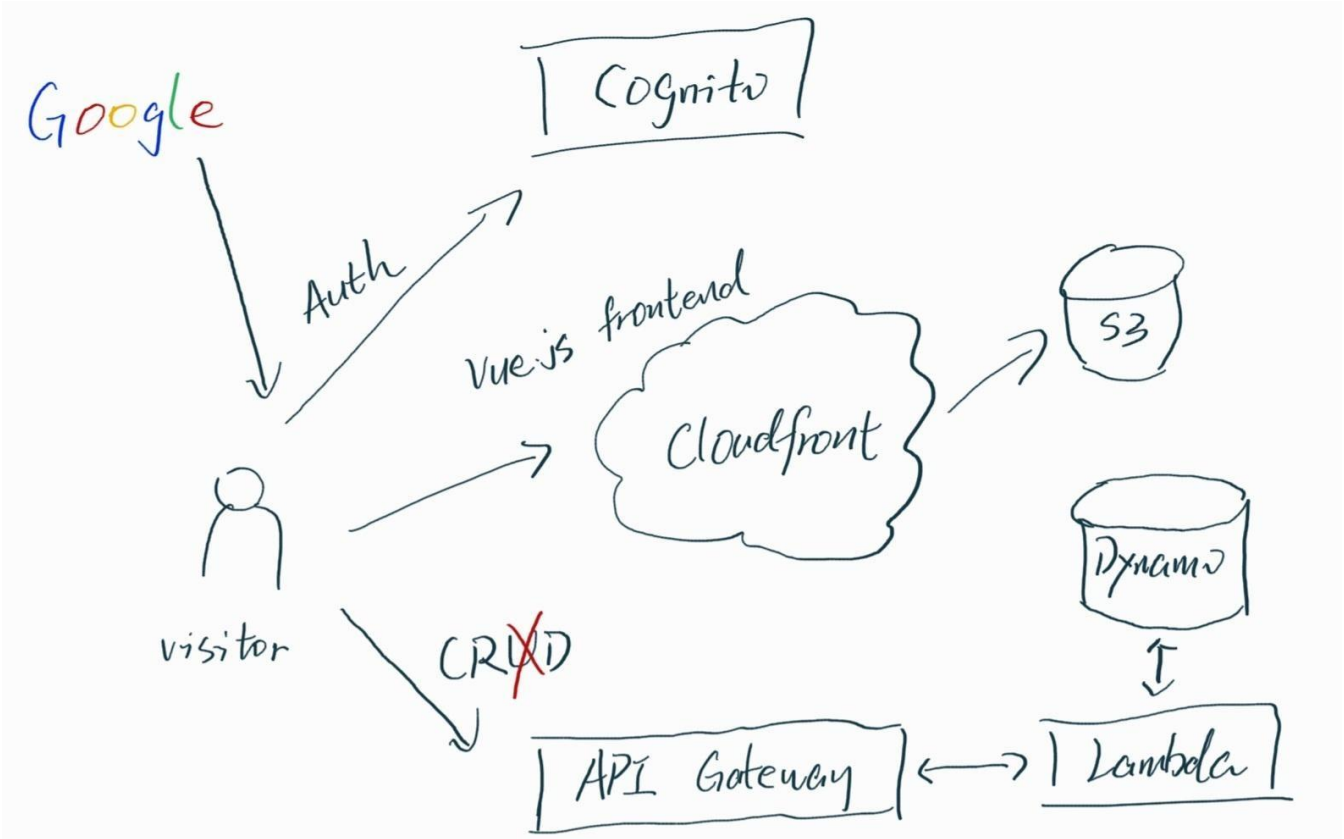
- Create - Get a shorter URL from your very very long URL
  - Example: Turning https://arstechnica.com/gadgets/2021/10/windows-11-the-ars-technica-review/ into https://s.anti.works/fhe92o1w
- Read - Redirect to your original URL
- Update - 

- Delete - When you regret having it

How?

- Frontend - Vue.js on S3 via Cloudfront
- Backend - Lambda with Serverless Framework via API Gateway
- Persistence - DynamoDB
- Authentication - Cognito with Google

No like actually how



Dynamo you say?

- shortStr (the fhe92o1w part of s.anti.works/fhe92o1w)
- userId: the email of the authenticated user
- longURL: as the name suggested 🤔
- createdAt: unix timestamp in milliseconds of when the entry was created

Primary Key: shortStr

Global Secondary Index: userId (HASH) + createdAt (RANGE) for pagination

Frontend

- Two components: Login & Main
- Login: When no JWT token is found locally, or the token is expired
  - On click: Redirect to Cognito
  - On callback: Read token off window.location.hash, and save to local storage (and vuex)
- Main: Main UI for creating a short URL, list of all of your URLs, and infinite scroll
  - POST with token in Authorization header to create short URL
  - GET with token, along with reference timestamp, to fetch upto 10 entries at a time
  - DELETE with token to remove the URL that belongs to you

Backend

- Extensive use of Serverless Framework to publish Lambda functions, and setup DynamoDB/IAM/Cognito App Client/API Gateway CORS/Custom Domain
- Communicate with DynamoDB with AWS SDK
- Authentication is done on API Gateway with Authorizer
  - Only allow @usfca.edu emails to create URLs
- GET /{shortId} does not require authentication
  - How else are you going to share the URL to your friends and family and rick roll them?

Demo

Was it difficult?

- DynamoDB: GlobalSecondaryIndex vs LocalSecondaryIndex
- Serverless Framework: Basically CloudFormation in different clothes
- CORS: Lambda function also needs to set headers to allow CORS, on top of API Gateway responding to OPTIONS
- Documentation?

Conclusion

- If you are going to use Lambda, use a framework (like Serverless)
- If you are going to use API Gateway, use a framework (like Serverless)
- If you are going to use AWS, hire someone else to do it
    - Or spend a night reading documentations that lead to nowhere