Serverless Architecture Discussion

MyTodoList

We want to make a mobile todo with REST API with HTTPS using serverless architecture. User should be able to interact with their own folder in S3. Users can write and read to-dos, but mostly read them. Database should scale and have high read throughput.

How do we do it?

API Gateway will be used for exposing endpoint. Behind it will invoke AWS lambda, and the lambda will be retrieving data from DynamoDB.

To add authentication we can use Amazon Cognito along with API Gateway to do verification.

To give user access to the S3 bucket, we will generate temporary credentials using Amazon Cognito then the client can use that temporary credentials to store/retrieve files from the S3 bucket.

To handle high read throughput for DynamoDB we can add DAX (Dyanmo Accelerator) so frequently access data are stored in the cache. Caching for REST request can be done at API gateway level as well.

Nothing is managed by us, it is all serverless.

MyBlog.com

This website should scale globally, rarely write blogs but read by many

Some websites is purely static files, the rest is a dynamic REST API. Caching should be implemented where possible, and new users should receive welcome email. Photo uploaded to the blog should generate a thumbnail.

CloudFront fronted S3 to expose the S3 files globally. Add origin access control so that the bucket can only be accessed by the CloudFront.

To add RESTful API you add API gateway, then invoke AWS lambda, that read from DynamoDB. Add DAX as well.

To give welcome email, you can use Dynamo stream, and trigger a lambda function to sent email using Simple Email Service for the welcome email.

You can also trigger the thumbnail creation by a lambda whenever the file is uploaded to S3 bucket.

Microservice Architecture

Synchronous pattern: API Gateway, load balancer, this allows you to make explicit call to the microservice

Asynchronous pattern: SQS, Kinesis, SNS, Lambda triggers, the invocation to the microservice will be done at a later time

There are challenges with microservices, but some of them can be solved by using serverless patterns.

Software updates offloading

If your EC2 instances host software updates, then there will be a sudden surge in traffic when people want to get a software updates. We want to optimize our cost and CPU how do we do it?

We just have to put CloudFront in front it, we don't need to change our architecture at all. CloudFront will scale for us, ASG don't have to scale as much. This is the easiest solution. CloudFront can make our application more scalable and cheaper since CloudFront cache the updates for us, so the request don't have to go to the EC2 instances as often.

Revision #1
Created 24 February 2023 16:21:31 by Tamarine
Updated 24 February 2023 18:59:24 by Tamarine