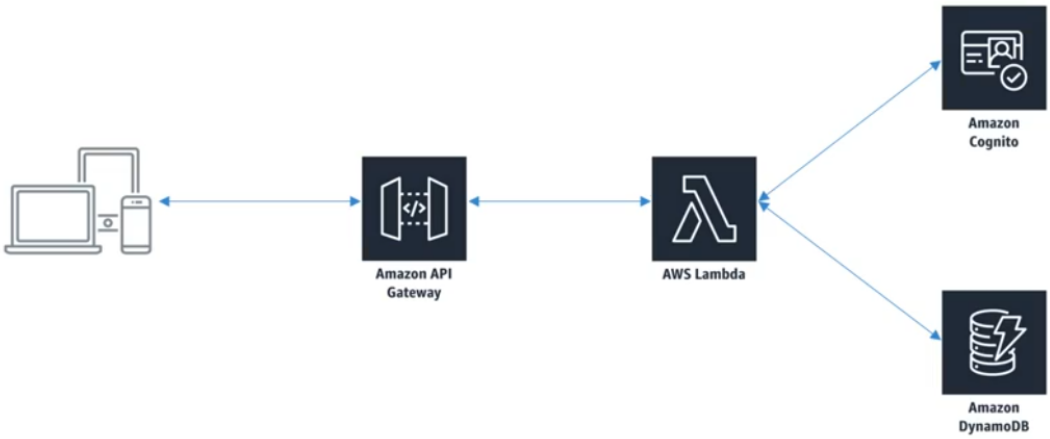
lambda

What is Lambda?

- A way to run code snippets "in the cloud"
  - Serverless
  - Continuous scaling
- Often used to process data as it's moved around

lambda is the glue between other services some services are not talk to each other

Example: Serverless Website



this images show the services example : it was chat retrieve data application when user login api gateway take the api and send to the lambda then lambda detect user id and go to the dynmodb pick the user data and back to api gateway show the users data

Example:
Order history app

Example:
Transaction rate alarm

Why not just run a server?

- Server management (patches, monitoring, hardware failures, etc.)
- Servers can be cheap, but scaling gets expensive really fast
- You don't pay for processing time you don't use
- Easier to split up development between front-end and back-end

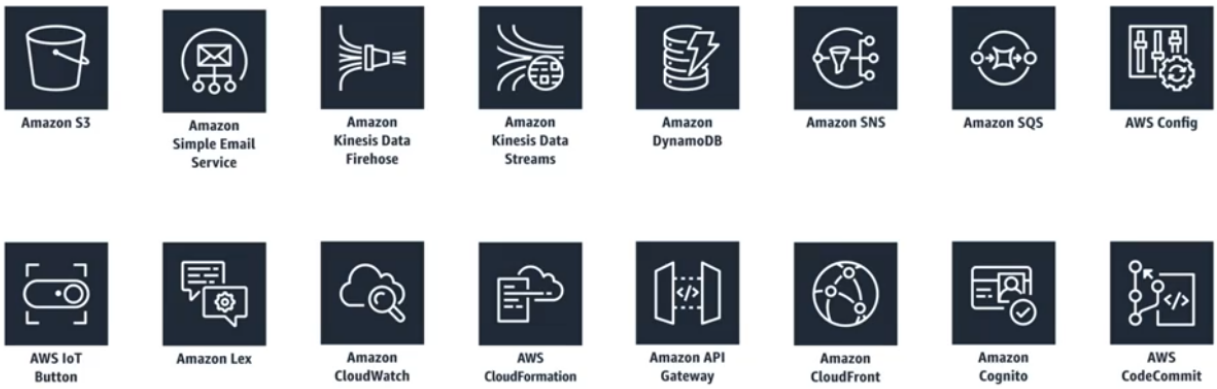
Main uses of Lambda

- Real-time file processing
- Real-time stream processing
- ETL
- Cron replacement
- Process AWS events

Supported languages

- Node.js
- Python
- Java
- C#
- Go
- Powershell
- Ruby

Lambda triggers



Lambda and Amazon Elasticsearch Service

Lambda and Data Pipeline

Data pipeline can be scheduled with preconditions to check S3

But Lambda lets you activate it at random times.

Lambda and Data Pipeline

Data pipeline can be scheduled with preconditions to check S3

But Lambda lets you activate it at random times.

Lambda + Kinesis

- Your Lambda code receives an event with a **batch** of stream records
  - You specify a batch size when setting up the trigger (up to 10,000 records)
  - Too large a batch size can cause timeouts!
  - Batches may also be split beyond Lambda's payload limit (6 MB)
- Lambda will retry the batch until it succeeds or the data expires
  - This can stall the shard if you don't handle errors properly
  - Use more shards to ensure processing isn't totally held up by errors
- Lambda processes shard data synchronously