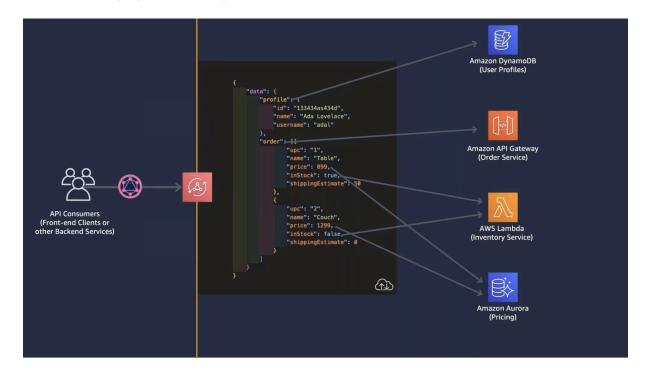


# Lecture 10 - AppSync, Amplify, other services (1h)

# AppSync

- Fully managed serverless GraphQL service
  - resolves underfetch or overfetch problems (having too much or not sufficient data in API response)
  - retrieve data from multiple backends with a single API call, even if backends are different technologies Full demo  $\rightarrow$  <u>https://youtu.be/Cs5e9vbjNdU</u>



- a really advanced GraphQL proxy for other AWS services or HTTP API
- Business use cases
  - realtime data broadcasting
  - live location as a service
  - online documents collaboration
  - ∘ IoT
  - data lakes
- · Technical and collaborative pros
  - Can be used as a connector between different microservices
  - · Resolves collaboration issues between isolated backend dev teams in a different accounts/departments
  - Simplifies work on org-wide APIs
  - Supports local mocking
  - Supports multi-auth mode with

- IAM
- Cognito
- OpenID
- API Keys

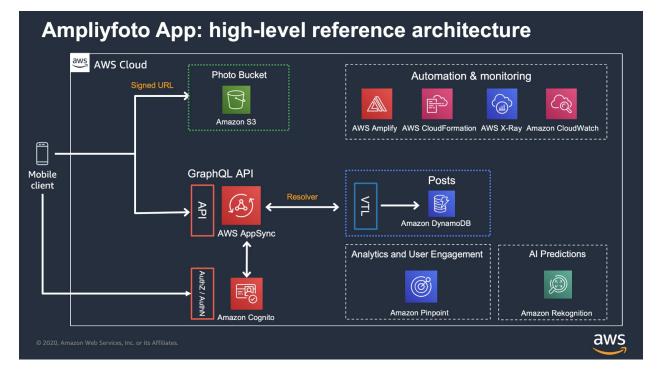
#### Amplify

· Start effortlessly

```
amplify configure
# walkthrough with aws profile setup
amplify init # init your new project
# this handles initial infra creation (roles, users, s3 buckets)
amplify add auth
# add auth capabilities to your app (creates cognito user pools and dynamodb tables)
amplify status
# check what changes to be added
amplify push
# push your changes to the undelying AWS infrastructure managed by amplify
amplifv add api
# deploy your graphql or rest api
amplify console
# open your Amplify or AWS ui in the browser
amplify update api
# redeploy your api with new settings
###
amplify delete
# seamples AWS resources cleanup
```

- · Business value
  - Scale as you grow
  - · handles offline and realtime workloads
  - · Focuses on Front-End developers and provides a set of tools to easily integrate with backends in the cloud
  - · abstracts a lots of core AWS services into a simpler concepts for the developer
- Closely integrates with AppSync (basically it manages the AppSync GraphQL APIs)
- own cli to bootstrap the apps and provision resources seamlessly
- · Amplify Studio IDE for Amplify workflows. Combines lots of services under one UI
- · Amplify Hosting provisions infrastructure for Web app hosting
- · Device Farm for testing the mobile applications on real mobile devices or desktops
- Amplify UI Components
  - Data creates DynamoDB table
  - Authentication Creates cognito User Pools
  - Storage Sets up S3
  - Functions Lambda Integration
  - GraphQL AppSync
  - Rest API Lambda + DynamoDB

- Analytics Kinesis
- Ui Library for storing UI elements (supports Figma designs)
- Amplify DataStore is capable of resolving data conflicts when the client goes offline for a while, and update that when client is back online
- Workshop → <u>https://catalog.us-east-1.prod.workshops.aws/workshops/84db0afb-0279-4d29-ae26-1609043d5bfd/en-US</u>



# Cognito

#### User pools

- · Serverless database of users for mobile or web app
- handles sign-up and login features
- supports external identity providers or SAML Federated Identities (sign-in with google, facebook etc)
- returns JWT on login
- · Integrates natively into the serverless applications in AWS

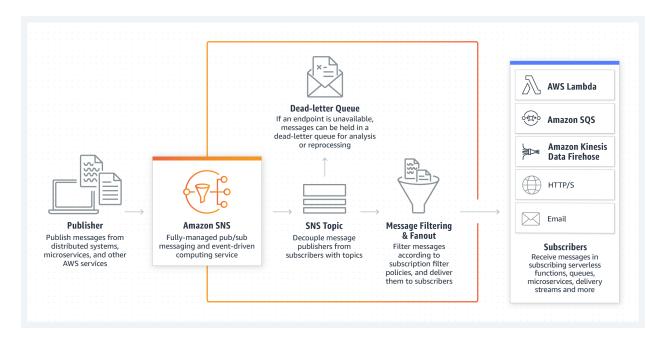
#### **Identity pools**

- · Identity pools can have IAM access to the AWS resources with roles and policies
- · users can login with OIDC, SAML, UserPools or guests

#### SNS

FAQ → <u>https://aws.amazon.com/sns/faqs/</u>

- Pub/sub
- SMS
- Mobile Push delivery

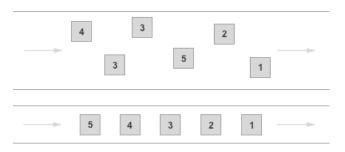


- supports multiple subscribers
- · Standard topics and FIFO topics
- often seen in pair with SQS an Lambda for fanouts

## SQS

FAQ → <u>https://aws.amazon.com/sqs/faqs/</u>

- fully managed message queue service
- serverless
- standard and FIFO queues (both have pros and cons)



- Features → <u>https://aws.amazon.com/sqs/features/</u>
- · Not a replacement for Kafka or RabbitMQ, however cover some of the use-cases

## **Kinesis**

#### **Data Streams**

 $\mathsf{FAQ} \rightarrow \underline{\mathsf{https://aws.amazon.com/kinesis/data-streams/faqs/?nc=sn\&loc=6}$ 

· high performant serverless data streaming service

- supports lambda integration for additional data processing
- · mainly used for logs and events streaming, real-time analytics or event-driven architectures
- output to lambda, kinesis firehose etc



#### **Data Firehose**

 $FAQ \rightarrow \underline{https://aws.amazon.com/kinesis/data-firehose/faqs/?nc=sn\&loc=5$ 

- · data streaming into the data lakes/warehouses
- ETL jobs
- can be used with ML
- output to Redshift/S3/Splunk

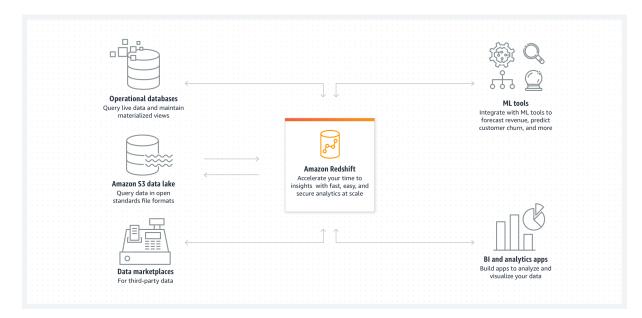
· · · · · · · · · · · · · · · · · · ·						
Input Logs, clickstreams, IOT, financial data, sales orders, and more	M AWS SDK				Amazon S3	
	WS SDK		AWS Lambda Custom Transformations		Amazon Redshift	
	Amazon Kinesis Data Streams		<sup>27</sup> Transformations			╡ <u>ੑੑੑੑ</u>
			န့္လြို့ Built-In လ္က်ိဳ Transformations		Amazon OpenSearch Service	(B III.
	Amazon Kinesis Agent				Service	
	Ayent		Transform - Optional		Amazon API Gateway	Output
	20+ AWS Services		· · · · · · · · · · · · · · · · · · ·			Analyze streaming data using interactive query
				i i i i i i i i i i i i i i i i i i i	Splunk	services (i.e., Amazon
						Athena, Amazon Redshift Spectrum), or
		]			6+ HTTP Endpoint	analytics tools
	Ingest					
		_			Load	
			Amazon Kinesis Data Firehose			
			Fully managed Ingest,			
			Transform, Load (ITL) solution			
			with no code required			

## Redshift

 $FAQ \rightarrow https://www.amazonaws.cn/en/redshift/faqs/$ 

- OLAP engine in AWS
- built on Postgres

- · data stored in columns
- super high performance can handle petabyte scale data warehouses
- Redshift Spectrum -> run queries against data lakes in S3 (not be confused with Athena which lacks the performance)



#### Batch

FAQ → <u>https://aws.amazon.com/batch/faqs/?nc=sn&loc=5</u>

- · Running batch compute jobs at scale
- · Spins up lots of EC2s or ECS containers to run the processing jobs

#### SSM

FAQ → https://aws.amazon.com/systems-manager/faq/

- allows to operate EC2s at scale
- resource grouping, maintenance windows, os patching ...
- RunCommand to run a command on thousands of EC2 at once
- · SessionsManager log in to the private instance via browser or CLI
- supports running ansible playbooks
- · ParamererStore simple credentials storage (if you don't need all the features of Secrets Manager)

## **Secrets Manager**

FAQ -> https://aws.amazon.com/secrets-manager/faqs/

- a password manager in AWS
- automatic credentials rotation
- · secrets are encrypted at rest with KMS
- IAM-granted access to the secrets
- good place to store secrets
- · slightly expensive (compared to parameters store)

• supports password generation and secrets retrieval via API/SDK

Store a new secret								
Secret type Info								
• Credentials for Amazon RDS database	Credentials for Amazon DocumentDB database	Credentials for Amazon Redshift cluster						
Credentials for other database	Other type of secret API key, OAuth token, other.							
Credentials Info								
User name								
Password								
Show password								
Encryption key Info You can encrypt using the KMS key that	Secrets Manager creates or a customer ma	anaged KMS key that you create.						
aws/secretsmanager		• C						
Add new key 🔀								
Database Info								
<b>Q</b> Search instances			<	1 >				
DB instance 🗸	DB engine $\nabla$	Status V C	reation date	$\nabla$				
No databases								
			Cancel	Next				

## KMS

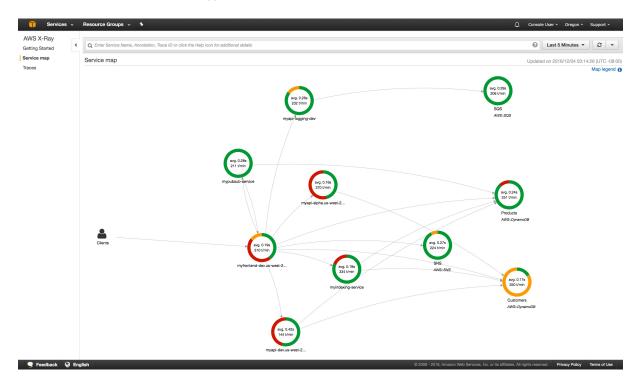
FAQ → <u>https://aws.amazon.com/kms/faqs/</u>

- fully managed key management service
- encrypts everything at rest, that can be encrypted (S3, RDS, EBS ...)
- · supports different encryption algorithms
- supports AWS managed and customer managed keys
- supports automatic keys rotation
- supports dedicated hardware with CloudHSM
- need-to-know service for the Security Specialty

# X-Ray

#### FAQ → <u>https://aws.amazon.com/xray/faqs/</u>

· collects and visualises traces between applications



• integrates in AWS AppSync

## CloudWatch

FAQ → <u>https://aws.amazon.com/cloudwatch/faqs/</u>

- fully managed alerting, logging and monitoring service
- supports detailed monitoring with up to 1sec refresh rate
- · dashboards, logs insights with SQL
- first place to check for logs and metrics
- enabled for every service
- (usually replaced by some other solution)

CloudWatch: Overview ~					Time range 1h 3h 12h 1d 3d 1w custom - Actions • 2 •		
All resources							
services summary o				Recent alarms o			
Services			1	Lambda Aggregate Duration	RDS PostGres Write Throughput 0		
Status		Alarm Insufficient OK		Milliseconds	Bytes/Second		
9 EC2				182x 8.15x	68.6k		
0 Lambda	2			88.6 Duration 2= 1000 for 1 datapoints within 1 minute	34.1k		
0 RDS	1		2	19:30 20:00 20:30 21:00 21:3	19:00 19:30 20:00 20:30 21:00 21:30 WriteThroughput		
A Kinesis		1					
DynamoDB			3	Lambda Error Count 0	CPU on App 0		
API Gateway				21.0	Percent 18.9		
© Billing				10.5 Errors >= 6 for 1 datapoints within 16 minutes	11.0 Martin My My Martin Martin Martin		
Classic ELB				0 19:00 19:30 20:00 20:30 21:00 21:30	5.00 CPUUtilization >= 5 for 1 datapoints within 1 minute 19:00 19:30 20:00 20:30 21:00 21:30		
CloudFront			-	Course .	CPUUtilization		
			,				
Default dashboard							
Custom metric 1	Custom metric 2			Custom metric 1			
Percent	Bytes			6.94			
7.20	16 Dk 16 Dk 19 00 19 30 20 00 20 30 21 50			0.04%			
				Custom metric			
				Custom metric 3			
				1 1 1			
6.80 · V				144			
19:00 19:30 20:00 20:30 21:00 21:30				Custom metrics 3			
Custom metrics 5	Custom metrics 2			Custom metric 3			
Bytes	Bytes			Count			
4.42G	22 28			145			
4.423				145			
4.423				144			
4.426				144			
4.423	19:00 19:30 20:00 2 Custom 1 Custom 2	0:30 21:00	21:30	143			
19:00 19:30 20:00 20:30 21:00 21:30	- custom 1 - Custom 2			19:00 19:30 20:00 20:30 21:00 21:30			
Cross service dashboard 0							
The cross service dashboard aggregates key metrics from each of the services in you	r account. View cross service dashboard						

# CloudTrail

- · Records API actions in the AWS account into "trails" and stores them in S3
- Think of it as a surveillance camera
- when enabled can't be disabled (only streaming to cloudwatch logs can be disabled)
- · better be enabled and sent to the logging account

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAJDPLRKLG7UEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mary_Major",
        "sessionContext": {
    "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-06-18T22:28:31Z"
           }
       },
"invokedBy": "signin.amazonaws.com"
    },
    "eventTime": "2019-06-19T00:18:31Z",
    "eventSource": "cloudtrail.amazonaws.com",
    "eventName": "StartLogging",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "signin.amazonaws.com",
    "requestParameters": {
        "name": "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-First-Trail"
    },
    "responseElements": null,
    "requestID": "ddf5140f-EXAMPLE",
    "eventID": "7116c6a1-EXAMPLE",
    "readOnly": false,
```

```
"eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
},
    ... additional entries ...
```

# ECS, ECR

- · ECS for running containerized workflows in AWS
- Task definition example

```
{
   "containerDefinitions": [
      {
         "command": [
            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top: 40px; background-color: #33
         1,
          "entryPoint": [
            "sh",
            "-C"
         1,
          "essential": true,
         "image": "httpd:2.4",
          "logConfiguration": {
             "logDriver": "awslogs",
             "options": {
                "awslogs-group" : "/ecs/fargate-task-definition",
                "awslogs-region": "us-east-1",
                "awslogs-stream-prefix": "ecs"
            }
         },
          "name": "sample-fargate-app",
          "portMappings": [
            {
                "containerPort": 80,
                "hostPort": 80,
"protocol": "tcp"
            }
         ]
     }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
"memory": "512",
   "networkMode": "awsvpc",
   "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
    },
   "requiresCompatibilities": [
       "FARGATE"
    ]
}
```

- ECR fully managed docker registry
  - lifecycle rules delete old docker images
  - vulnerability scan on upload
  - IAM-access

#### Beanstalk

- generates environments and infrastructure based on templates (usually EC2 + ELB + ASG)
- uses Cloudformation inside
- good place to start if unfamilliar with AWS
- kinda big and old, better take a deep dive on it

## EKS

- Kubernetes in AWS
- fully managed HA control plane out of the box
- pay for the control plane plus for the worker nodes
- cool to use with spot worker nodes (recommend spot.io)
- · IAM OIDC integration Pods can assume IAM roles to interact with other AWS resources
- supports Fargate worker nodes

# Elasticsearch

- Fully managed Elasticsearch and Kibana in AWS
- it's a fork of original Elastic called OpenDistro
- not all features are present, slightly lags behind the original Elastic
- · supports IAM authentication and lots of other AWS stuff (services integration, HA, VPC placement)
- good for logs storage, easily enabled log streaming from CloudWatch by Lambda

# **Q&A** session

- Topics discussion
- Sharing additional resources