

저비용으로 서버리스 웹서비스 만들기

김한성

간단 자기 소개

- 이름 : 김한성 (gnidoc327)
- 소속 : SKT Cloud Data Solution팀
- 업무 : Meta Data Discovery Tool 개발

목차

- 서비스 소개 - Metadata Discover Tool
- 서버리스를 택한 배경
- 서버리스를 도입하면서 고민했던 부분
- 적용된 아키텍처 소개
 - frontend
 - backend
- 가격비교 : 서버리스 vs IaaS
- 운영 중 발견한 문제(단점)
- Q&A

서비스 소개

Metadata Discover Tool

Metadata Discovery Tool

- Metadata
 - 데이터에 대한 데이터
 - 데이터에 관한 구조화된 데이터
 - 다른 데이터를 설명해 주는 데이터

3	5	2.5	2
아니오	예	아니요	아니오
2000이하	2000원	2900	0
남	여	남	남
31	32	31	33
자녀와함께	자녀와함께	1인	부부

Metadata Discovery Tool

- Metadata
 - 데이터에 대한 데이터
 - 데이터에 관한 구조화된 데이터
 - 다른 데이터를 설명해 주는 데이터

	유저1	유저2	유저3	유저4
추천 의향	3	5	2.5	2
가입의향	아니오	예	아니요	아니오
유료서비스 적정요금	2000이하	2000원	2900	0
성별	남	여	남	남
연령	31	32	31	33
가족유형	자녀와함께	자녀와함께	1인	부부

Metadata Discovery Tool

- Metadata
 - 데이터에 대한 데이터
 - 데이터에 관한 구조화된 데이터
 - 다른 데이터를 설명해 주는 데이터

	유저1	유저2	유저3	유저4
추천 의향	3	5	2.5	2
가입의향	아니오	예	아니요	아니오
유료서비스 적정요금	2000이하	2000원	2900	0
성별	남	여	남	남
연령	31	32	31	33
가족유형	자녀와함께	자녀와함께	1인	부부



DATA HUB
Datasets
metadata

Filters

Data Origin

☐ Prod (11068)
☐ Ei (2686)
☐ Corp (244)

Platform

☐ Hive (8333)
☐ Hdfs (5070)
☐ Mysql (374)
☐ Kafka (61)
☐ Dalids (50)
☐ Espresso (30)
☐ Pinot (26)
☐ Teradata (15)

Datasets

Showing 1 - 10 of 13998 results

u_metrics.metadata_drift_v2_union

Data Origin

PROD

Platform

hive

METRICS.MetadataLineageEvent_v1

Data Origin

PROD

Platform

kafka

TRACKING.MetadataChangeEvent_v2

Data Origin

PROD

Platform

kafka

/data/service_column/metadata_lineageevent_v1/final

Data Origin

PROD

Platform

hdfs

/data/service/MetadataLineageEvent_v1

DATA HUB
Datasets
Search for datasets...

Datasets > hdfs > jobs > metrics > ump_v2 > metrics > basic_qualified_applies > basic_qualified_applies

Schema
Status
ACL Access
Ownership
Compliance
Relationships
Health
Docs

Downstream

Dataset	All Types	Platform	Actor	Last Modified
u_metrics.basic_qualified_applies__basic_qualified_applies_metrics	MANAGE	Hive	umrll:multiProduct:wherehows-samza-hive	Yesterday at 6:18 PM

Lineage (Beta)

Up 1 Level

foundation_core_entity_mp.dim_position

Platform

dalids

Type

TRANSFORMED

Actor

umrll:multiProduct:ump

Children

1

Parents

-

u_ifayad.all_title_similarity_scores

Platform

hive

Type

TRANSFORMED

Actor

umrll:multiProduct:ump

Children

1

Parents

-

DATA HUB
Datasets
Search for datasets...

Datasets > kafka > metrics > MetadataChangeEvent_v4

Metrics.MetadataChangeEvent_v4

Fabric: Prod

100%

Health Score

Schema
Status
ACL Access
Ownership
Compliance
Relationships
Health
Docs

View as table

View as JSON

Last modified: 06/22/2019, 1:21:57 am

Column	Data Type	Default Comments
auditHeader[type=com.linkedin.events.KafkaAuditHeader].time	Long	The time at which the event was emitted into kafka.
auditHeader[type=com.linkedin.events.KafkaAuditHeader].server	String	The fully qualified name of the host from which the event is being emitted.
auditHeader[type=com.linkedin.events.KafkaAuditHeader].instance [type=string]	String	The instance on the server from which the event is being emitted. e.g. i001
auditHeader[type=com.linkedin.events.KafkaAuditHeader].appName	String	The name of the application from which the event is being emitted. see go/appname
auditHeader[type=com.linkedin.events.KafkaAuditHeader].messageId	Fixed	A unique identifier for the message

DATA HUB
Datasets
Search for datasets...

Datasets > kafka > metrics > MetadataChangeEvent_v4

Metrics.MetadataChangeEvent_v4

Fabric: Prod

100%

Health Score

Schema
Status
ACL Access
Ownership
Compliance
Relationships
Health
Docs

Owners

Please add at least 2 owners

Last modified: 2 months ago by aberger

LDAP Username	Full name	ID Type	Ownership Type
aberger	Arianne Berger	User	Data owner
dkealoha	David Kealoha	User	Data owner
nsewart	Nosizwe Stewart	User	Data owner

서버리스를 택한 배경

왜 서버리스(Serverless)?

1. 비용절감

- a. 사용자가 적은 사내서비스
- b. 다뤄야되는 데이터양은 많은데 동접자는 매우 적음

2. 완전관리형 서비스

- a. 보장된 HA, 무중단 배포
- b. 개발/운영비용 절감 - 모니터링, 알림 등등
- c. 간단한 MSA 구조

3. 클라우드 최적화

- a. 기본으로 제공되는 문서
- b. DevOps(code-base) - terraform, zappa
- c. Support Plan
- d. 보안...?

서버리스를 도입하면서 고민했던 부분

1. 100%까지는 불가능한 서버리스

- 서버(EC2)가 필요한 영역
 - batch
 - 얼마나 오래 걸릴지 모르는 작업 - 통계, 추천 모델 등
 - 1회성 작업
 - OS 의존성
 - shell script
 - 윈도우는 이런거 안되지?
 - 맥북에서는 될줄 알았냐? - **unix-like**의 함정
 - 윈도우에서 배포하니 실행이 안된다... - 그리워지는 JVM
 - intel vs arm
 - jupyter notebook
 - sagemaker는 너무 비싸요
 - 맥북에서 돌리면 너무 느려요



IT WORKS
ON MY MACHINE

2. Vendor lock-in

특정 **vendor**의 독자적 기술에

크게 의존한 제품, 서비스, 시스템 등을 채용했을 때

다른 **vendor**가 제공하는

동종 제품, 서비스, 시스템 등으로 갈아타기 어렵게 되는 현상

2. Vendor lock-in

특정 **vendor**의 독자적 기술에

크게 의존한 제품, 서비스, 시스템 등을 채용했을 때

다른 **vendor**가 제공하는

동종 제품, 서비스, 시스템 등으로 갈아타기 어렵게 되는 현상



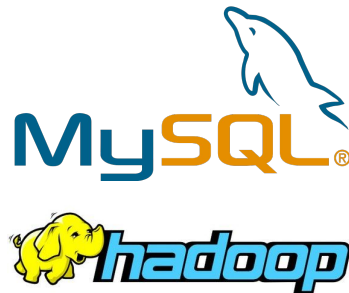
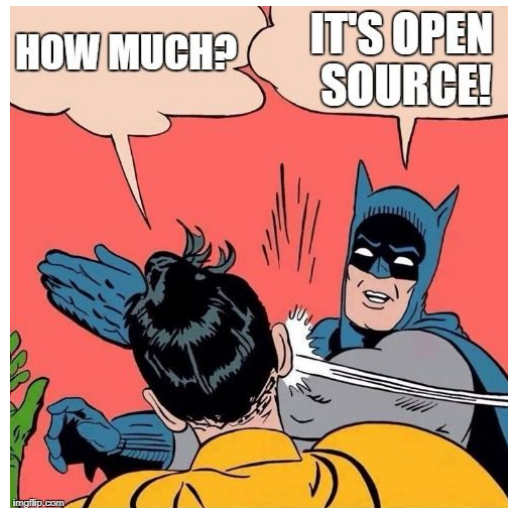
2. Vendor lock-in

특정 vendor의 독자적 기술에

크게 의존한 제품, 서비스, 시스템 등을 채용했을 때

다른 vendor가 제공하는

동종 제품, 서비스, 시스템 등으로 갈아타기 어렵게 되는 현상



3. K8S을 쓰고 싶다...

- 완전관리형 서비스 = 내가 관리할 수 없음
 - auto-scaling : 아침엔 100명, 점심엔 1만명, 저녁엔 0명
 - lambda : cold start
- AWS에서 비슷한걸 제공하지만 아쉬운 IaaS서비스
 - Code Series = Gitlab, Jenkin, Nexus
 - dynamo/documentdb = mongodb
 - chatbot = slackbot
 - Open Search = Elastic Search
 - wordpress



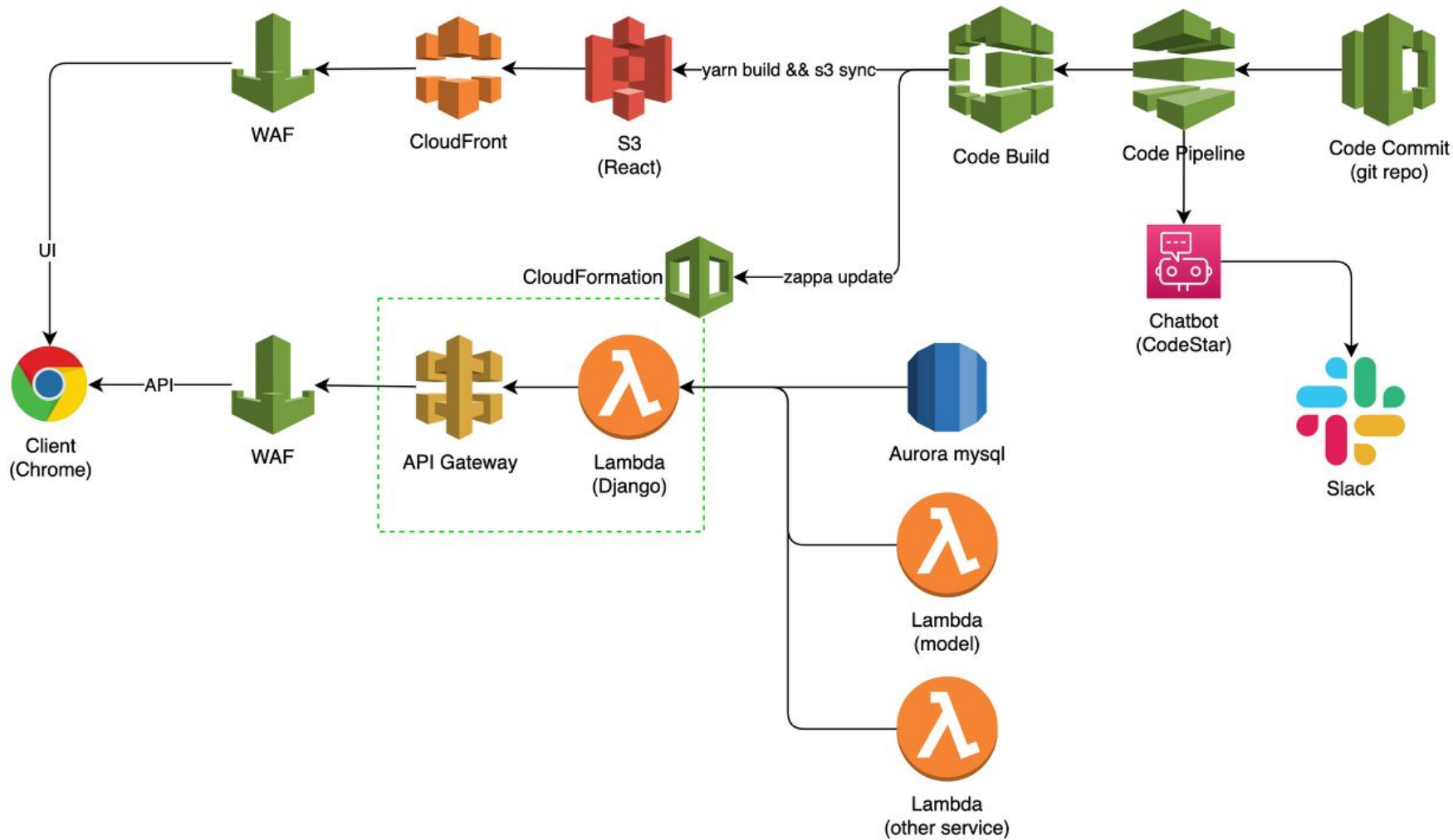
4. AWS 장애

- 답이 없...
- 종교가 없어도 모두가 기도하는 순간

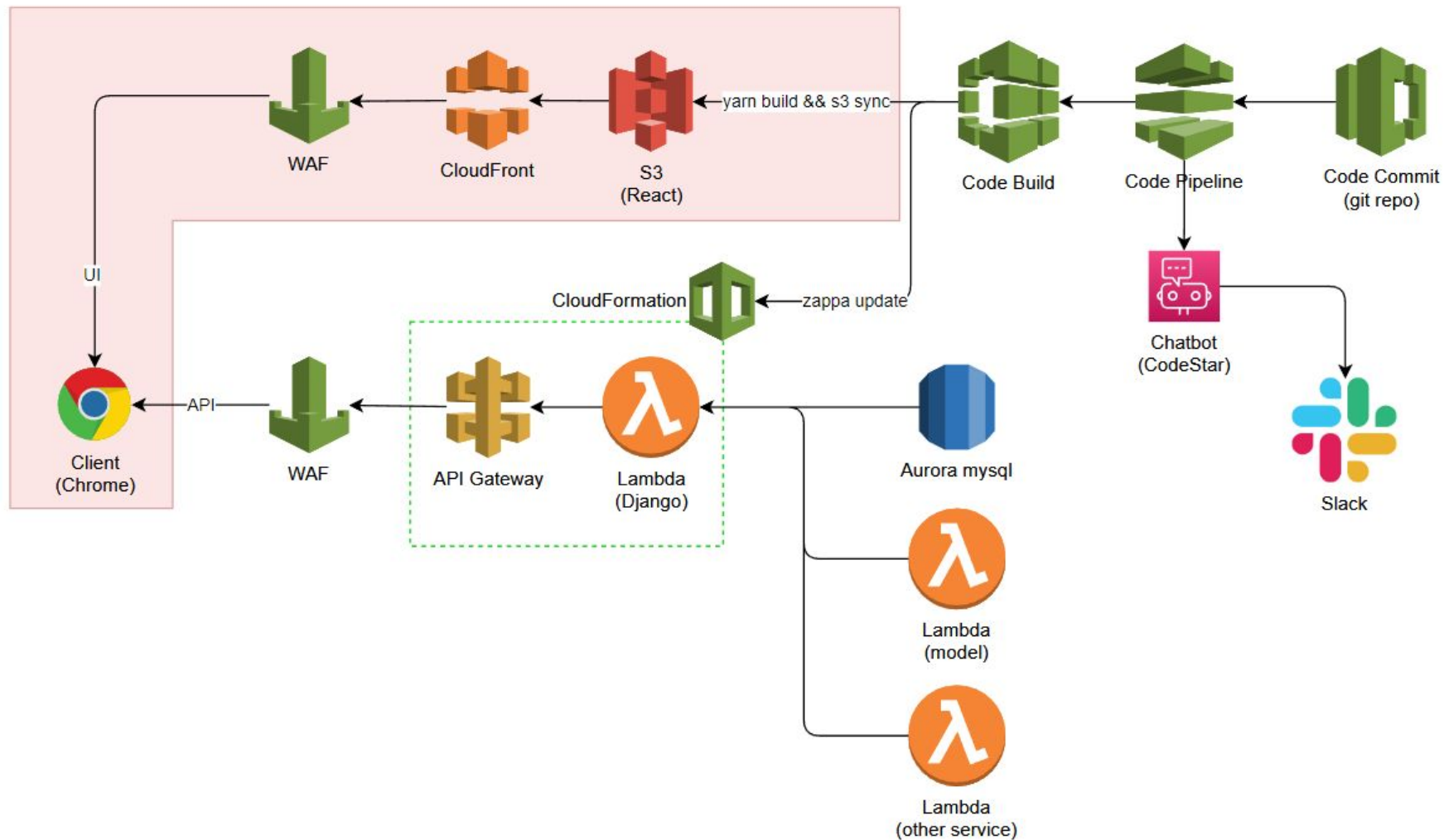


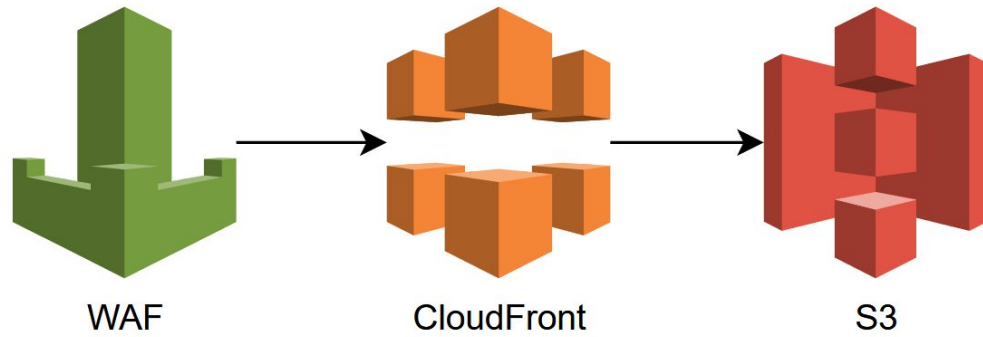
나만 장애난거 아니니깐 괜찮겠...(이걸 우리 팀장님이 안보시기를...)

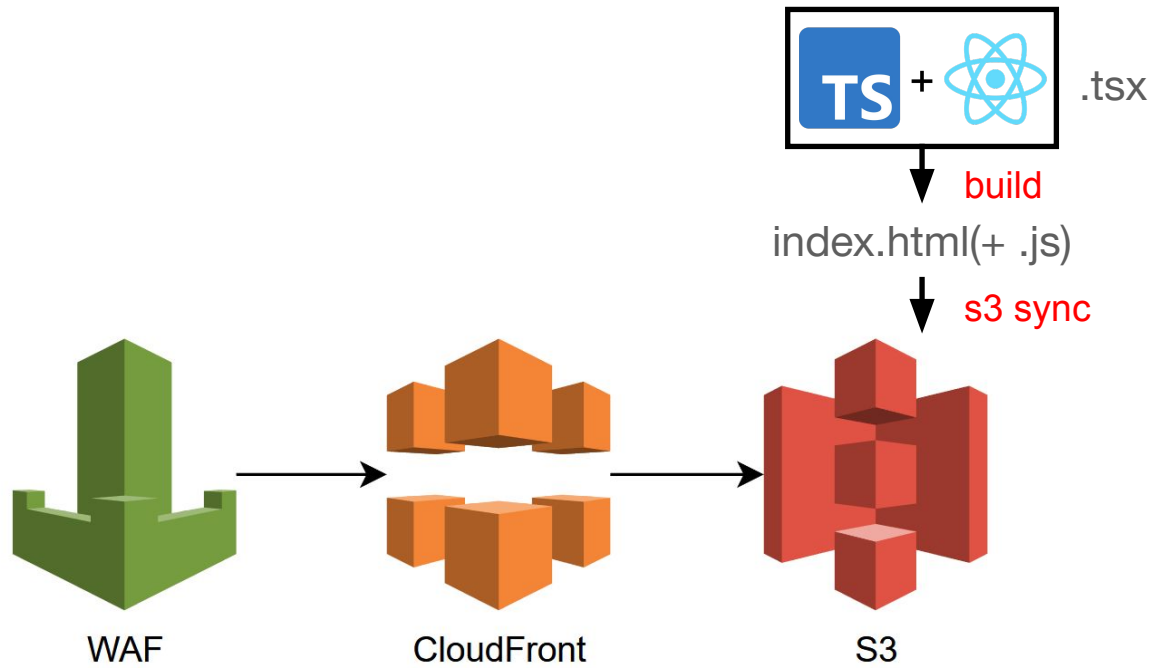
적용된 아키텍처 소개



Frontend



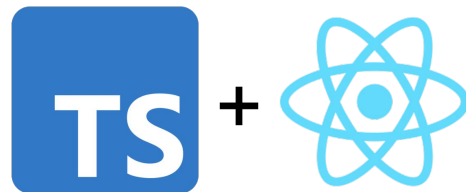




Frontend 기술 스택

- Typescript + React

- ant design : UI
- redux : state
- code : prettier, eslint
- util : lodash.js
- API client : axios
- URL path : react router



Ant Design



ESLint



Prettier



REACT ROUTER



Redux

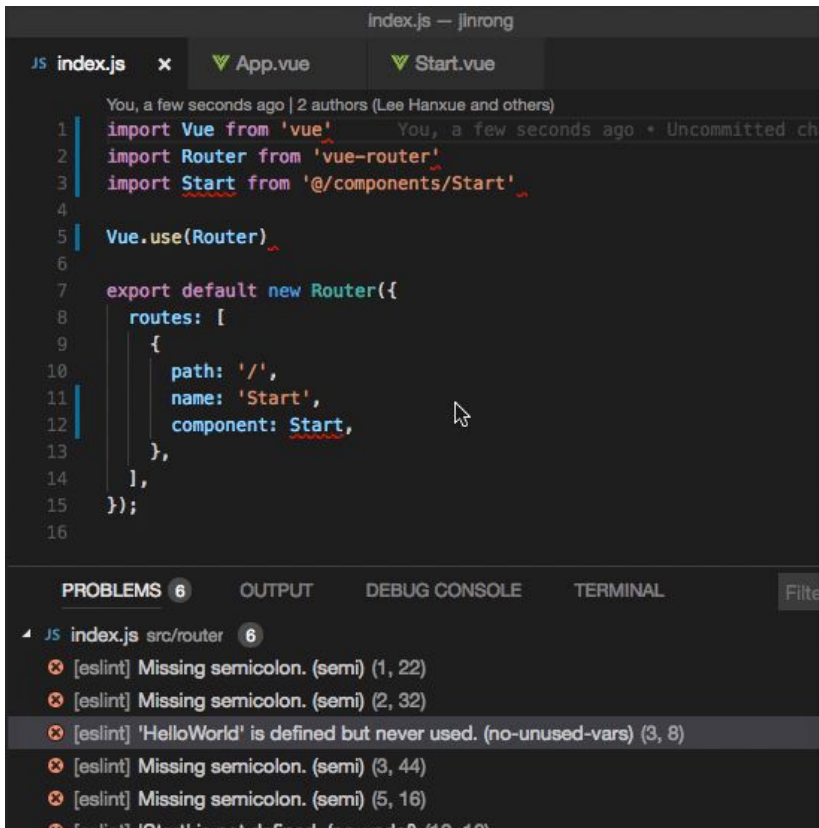
AXIOS

Lo



awesome
web react

eslint, prettier

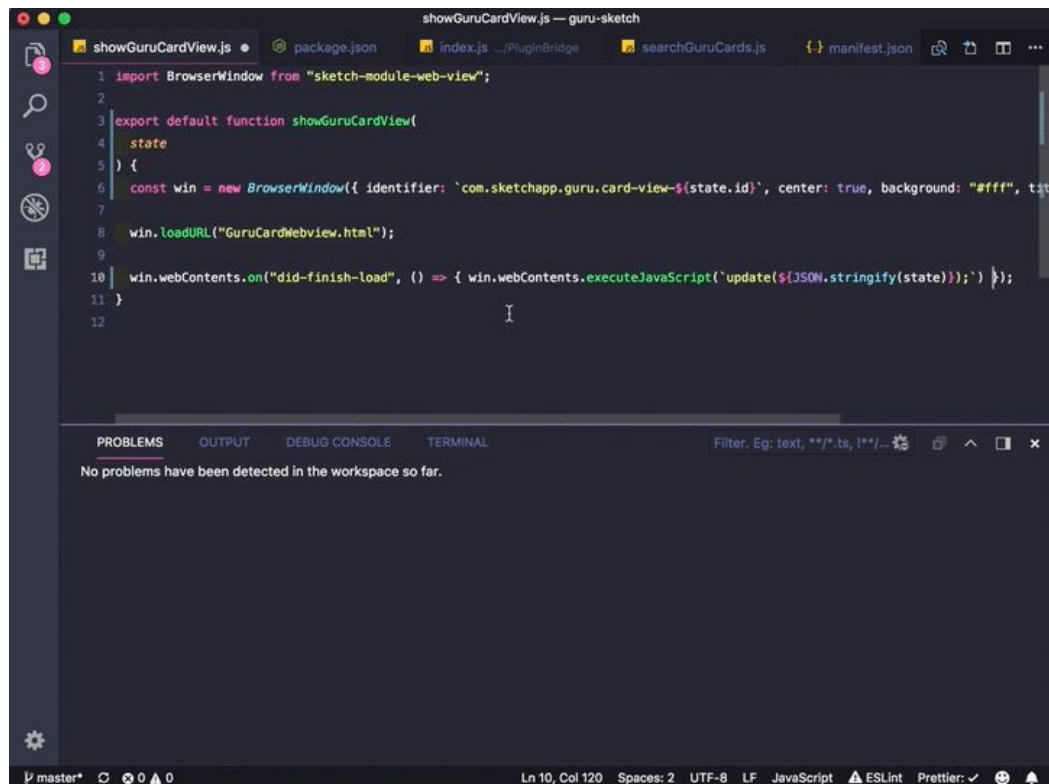


The screenshot shows a VS Code editor window with a file named `index.js` open. The code is a Vue.js application using `Vue` and `Vue Router`. The code is as follows:

```
1 import Vue from 'vue'
2 import Router from 'vue-router'
3 import Start from '@components/Start'
4
5 Vue.use(Router)
6
7 export default new Router({
8   routes: [
9     {
10      path: '/',
11      name: 'Start',
12      component: Start,
13    },
14  ],
15 });
```

Below the code editor, the **PROBLEMS** panel is visible, showing 6 errors detected by ESLint:

- [eslint] Missing semicolon. (semi) (1, 22)
- [eslint] Missing semicolon. (semi) (2, 32)
- [eslint] 'HelloWorld' is defined but never used. (no-unused-vars) (3, 8)
- [eslint] Missing semicolon. (semi) (3, 44)
- [eslint] Missing semicolon. (semi) (5, 16)
- [eslint] 'Start' is not defined. (no-undef) (10, 10)



The screenshot shows a VS Code editor window with a file named `showGuruCardView.js` open. The code is a JavaScript module using `BrowserWindow` from `sketch-module-web-view`. The code is as follows:

```
1 import BrowserWindow from "sketch-module-web-view";
2
3 export default function showGuruCardView(
4   state
5 ) {
6   const win = new BrowserWindow({ identifier: `com.sketchapp.guru.card-view-${state.id}`, center: true, background: "#fff", title: "Guru Card View" });
7   win.loadURL("GuruCardWebView.html");
8
9   win.webContents.on("did-finish-load", () => { win.webContents.executeJavaScript(`update(${JSON.stringify(state)})`); });
10 }
11
12
```

Below the code editor, the **PROBLEMS** panel is visible, showing no errors detected by ESLint.

At the bottom of the VS Code window, the status bar shows the following information: `master*`, `0` errors, `0` warnings, `0` info, `0` hints, `Ln 10, Col 120`, `Spaces: 2`, `UTF-8`, `LF`, `JavaScript`, `ESLint`, `Prettier`, and a checkmark icon.

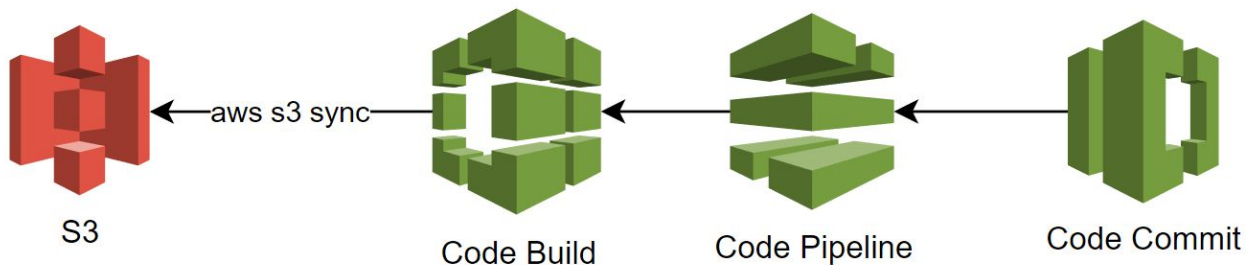
개발부터 배포 과정(CI/CD)

- 개발

코드 작성 → 실행/테스트 → Lint → Formatter → Test → **git push**

- 배포(in codebuild)

Lint → Test → Build → **Deploy**

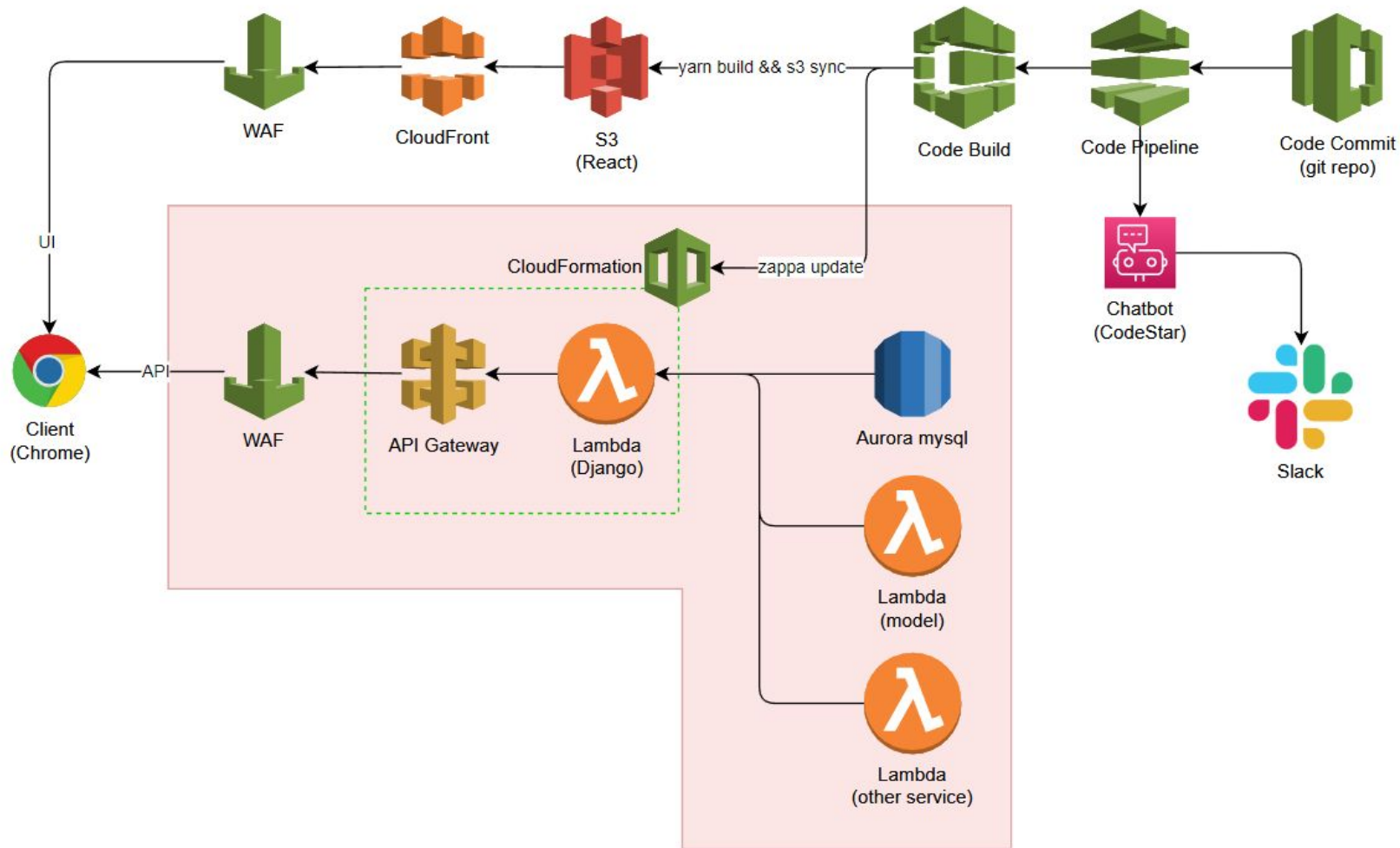


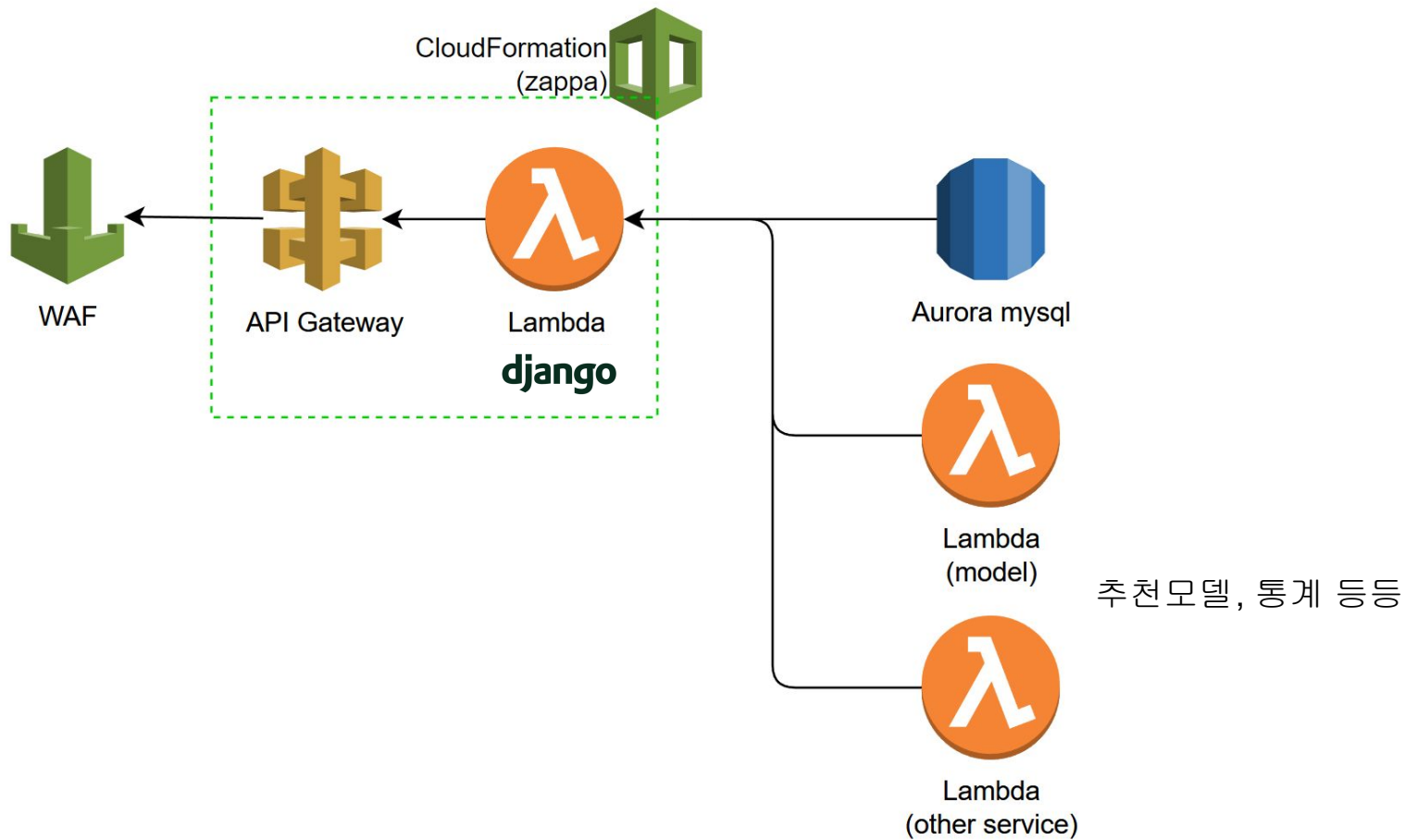
왜 jekyll, hexo는 안 썼나요?

- 플러그인 의존성
- CLI로 페이지 생성 - 페이지별로 html 파일 생성
- 로그인, 글작성, 유틸과 같은 공통 모듈을 사용해야되는 경우 구조가 난잡해짐
 - 게시글마다 html 파일이 생성되는데 게시글마다 댓글 기능? => 공통 모듈 필요
 - 페이지마다 공통 모듈을 import => 의존성 지옥
 - jquery 사용이 필요 => 유지보수 지옥
- 생성된 페이지 수정시 번거로움(수정 후 재생성 + CLI)
- 필요한 라이브러리, 데이터 연계의 어려움
- 기본적으로 블로그 프레임워크라 커스텀이 많이 필요
 - 생각보다 원하는 형태의 플러그인(오픈소스)가 잘 없음

=> React를 쓰게된 이유...

Backend





Backend 기술 스택

django

- Django
 - swagger : drf-yasg
 - code : pylint, black, coverage.py
 - restfull : rest framework
 - login : Simple JWT
 - deploy : zappa



Zappa

- <https://github.com/zappa/Zappa>
- CloudFormation + boto3 기반의
매우 간단한 serverless 빌드/배포용 패키지
- Flask / Django 지원
- CloudWatch Log 확인 가능(Tailing Logs)

```
{
    // The name of your stage
    "dev": {
        // The name of your S3 bucket
        "s3_bucket": "lambda",

        // The modular python path to your WSGI application function.
        // In Flask and Bottle, this is your 'app' object.
        // Flask (your_module.py):
        // app = Flask()
        // Bottle (your_module.py):
        // app = bottle.default_app()
        "app_function": "your_module.app"
    }
}
```

```
▶ zappa tail dev --since 1h
Calling tail for stage dev..
[1616661338843] [DEBUG] 2021-03-25T08:35:38.843Z 68b0ddd3-1239-48a8-8ece-52606f88f256 Zappa Event: {'time': '2021-03-25T08:34:52Z', 'detail-type': 'Scheduled Event', 'source': 'aws.events', 'account': '524500027903', 'region': 'ap-northeast-2', 'detail': {}, 'version': '0', 'resources': ['arn:aws:events:ap-northeast-2:524500027903:rule/data-weaver-dev-zappa-keep-warm-handler.keep_warm_callback'], 'id': 'bd043ad3-af33-f678-a42b-e5bb16765db8', 'kwargs': {}}
[1616661338844] [DEBUG] 2021-03-25T08:35:38.843Z 68b0ddd3-1239-48a8-8ece-52606f88f256 Zappa Event: {}
[1616661579281] [DEBUG] 2021-03-25T08:39:39.279Z a10ab377-b395-4776-b2da-6db349a881f1 Zappa Event: {'time': '2021-03-25T08:38:52Z', 'detail-type': 'Scheduled Event', 'source': 'aws.events', 'account': '524500027903', 'region': 'ap-northeast-2', 'detail': {}, 'version': '0', 'resources': ['arn:aws:events:ap-northeast-2:524500027903:rule/data-weaver-dev-zappa-keep-warm-handler.keep_warm_callback'], 'id': 'de9f08be-8c75-79e5-e99a-808c246e01d2', 'kwargs': {}}
```


Zappa

- zappa cli를 사용하여
프로젝트 생성/빌드/배포가 자유로움
 - 생성 : `zappa init`
 - 배포 : `zappa deploy [env name]`
 - 업데이트 : `zappa update [env name]`
 - 로그 확인 : `zappa tail [env name]`
- CloudFormation 기반이지만 별도의 json 포맷으로 관리
- Serverless 에서 고려해야될 요소를 설정 가능
 - Lambda Cold Start : Provisioned Concurrency X / CloudWatch events O
 - AWS X-Ray : 분산 추적 시스템

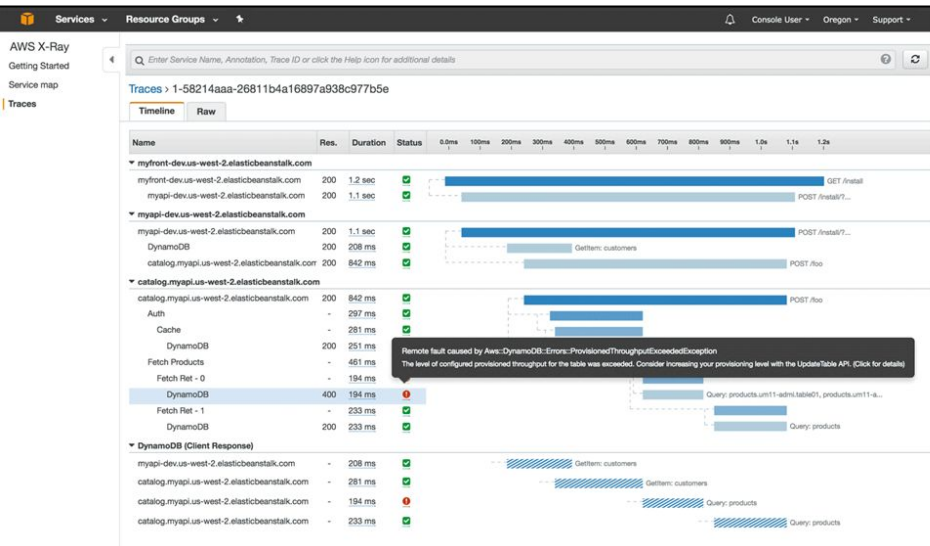
CloudFormation 문제

- Script가 매우 복잡함
 - 유지보수 어려움
 - 재사용 어려움
 - 템플릿을 s3에 올려야 사용 가능
- 가독성이 매우 떨어짐
 - 템플릿이 yaml / json 포맷이지만 요소 간의 의존성이 커서 human error 발생할 가능성이 높음
- 로컬 테스트 시에도 s3에 템플릿이 올라가야함
 - 로컬 테스트 시 lambda docker image를 통해서 동작하기 때문에 PC 리소스를 매우 많이 사용함

Terraform 문제

- CloudFormation에 비해 지원이 완벽하지 않음
 - AWS에서만 지원되는 일부 기능에 대해서 사용이 불가
 - X-RAY
- Lambda에 올라갈 코드와 Terraform 코드를 함께 관리하기 어려움
 - 코드만 업데이트하고 싶은데 인프라도 함께 반영
 - 코드는 바뀌었는데 Terraform apply에서 에러 발생해서 Rollback
 - DevOps도 분리가 필요!
- 배포시 상대적으로 느린 속도
 - 모듈화가 강제화됨
- HCL syntax
- 매우.. 사소한 버그들

Lambda X-RAY

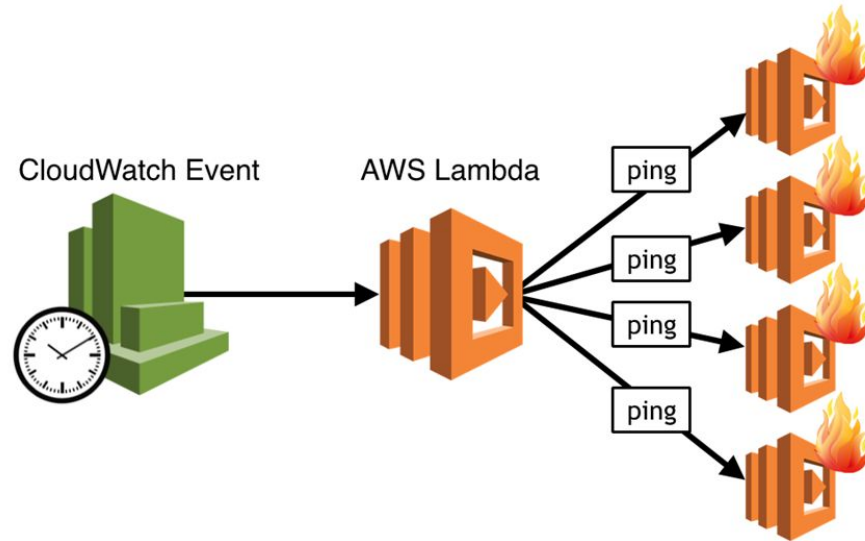
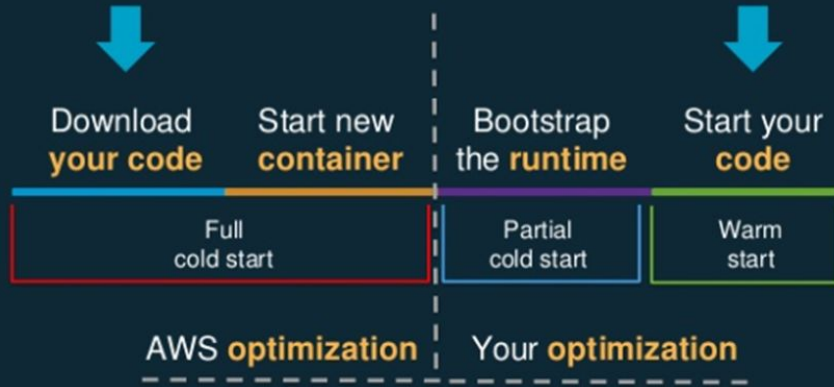


Service map



Lambda Cold Start

The function lifecycle



```
"keep_warm": true, // Create CloudWatch events to keep the server warm. Default true.  
"keep_warm_expression": "rate(4 minutes)", // How often to execute the keep-warm, in
```

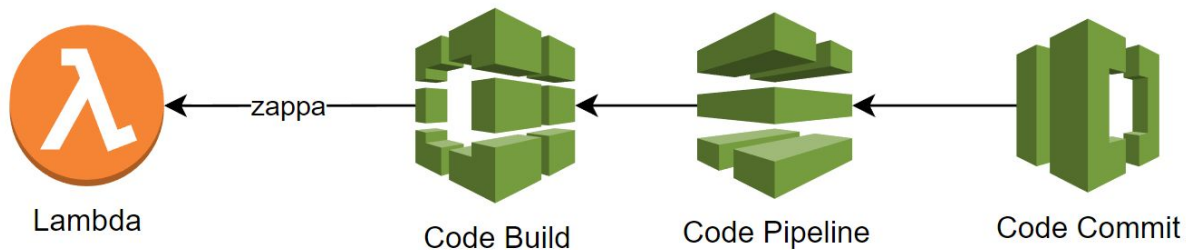
개발부터 배포 과정(CI/CD)

- 개발

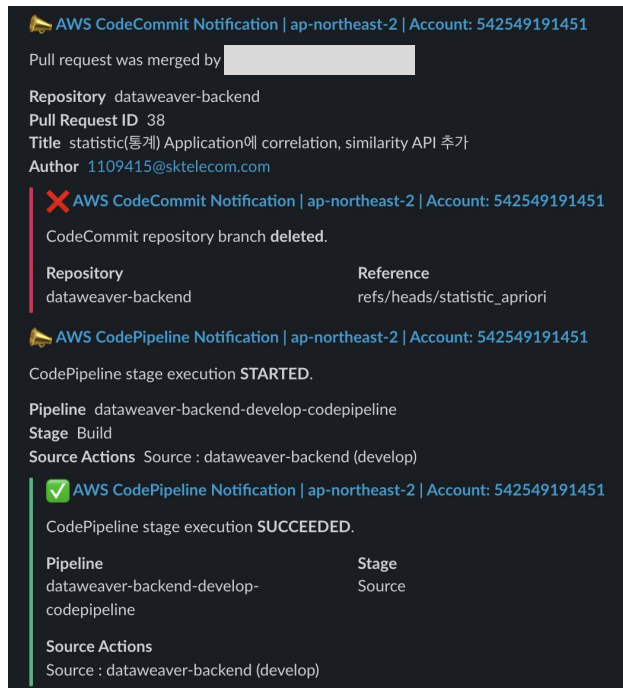
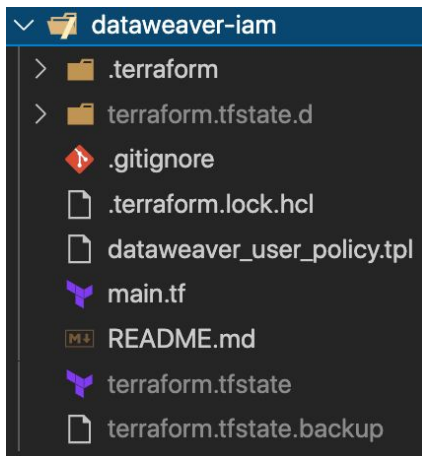
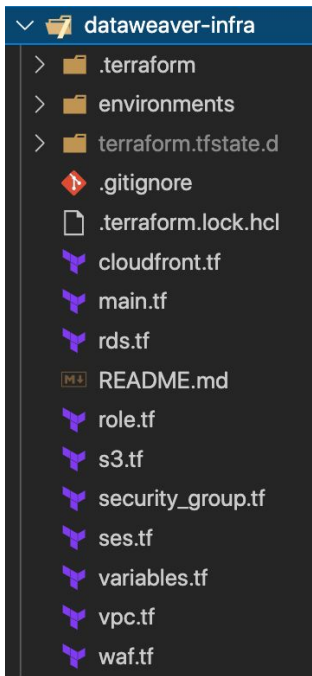
코드 작성 → 실행/테스트 → Lint → Formatter → Test → **git push**

- 배포(in codebuild)

Lint → Test → Build → **Deploy**



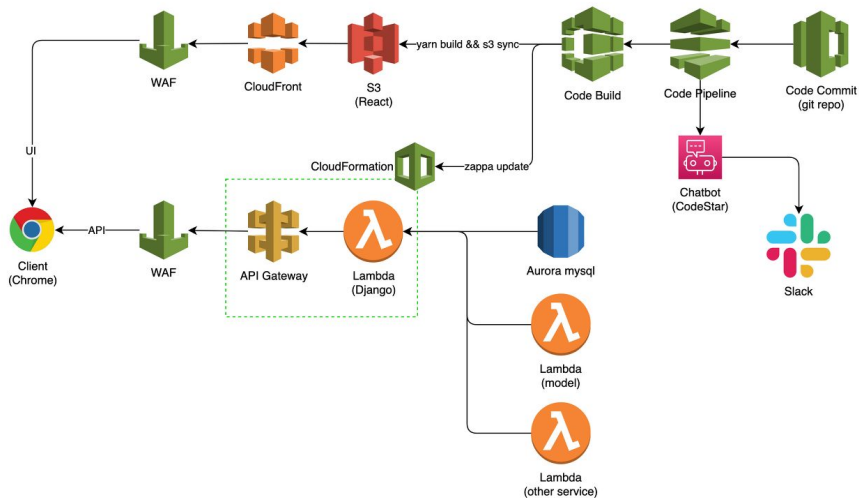
Infra 관련 - terraform, slack



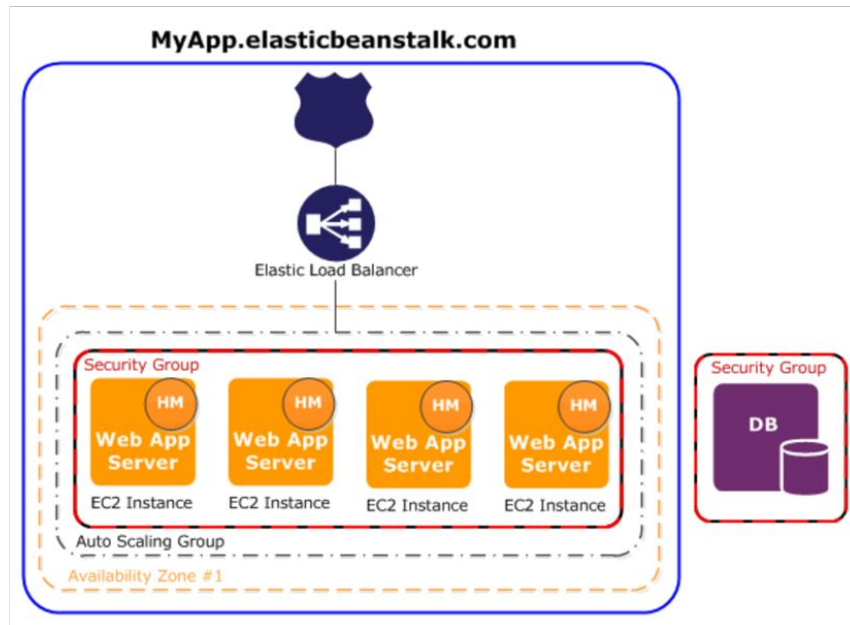
가격 비교 서버리스 vs IaaS

EC2와 비교

Serverless



EC2(Elastic Beanstalk)



EC2와 비교

Serverless

- API Gateway
- Lambda
- CloudFormation



EC2(Elastic Beanstalk)

- ELB
- EC2
- Elastic Beanstalk

가격 비교

분류	서비스	성능/타입	비용	갯수	비고
Serverless	API Gateway	HTTP API	한달에 100만 <u>요청시</u> \$2.46	100만	한달에 100만번 <u>호출</u> = 1분에 24번 호출
	Lambda	256MB	한달 \$21.03		평균 요청 size = 1MB 1요청당 5초간 실행된다고 가정
Elastic Beanstalk	EC2	t3.small (2 core, 2GiB)	한달 \$39	2대	HA 고려

계산기 : <https://calculator.aws/#/>

가격 비교 - 약 3.3배 차이

분류	서비스	성능/타입	비용	갯수	비고
Serverless	API Gateway	HTTP API	한달에 100만 <u>요청시</u> \$2.46	100만	한달에 100만번 <u>호출</u> = 1분에 24번 호출
	Lambda	256MB	한달 \$21.03		평균 요청 size = 1MB 1요청당 5초간 실행된다고 가정
Elastic Beanstalk	EC2	t3.small (2 core, 2GiB)	한달 \$39	2대	HA 고려

계산기 : <https://calculator.aws/#/>

그래서 얼마 나오고 있는데?

~~일단 제 월급보단 적네요~~

하루 접속자 100명 이하

RDS 제외하고 10~20만원 사이로 사용 중

Details

AWS Service Charges **\$1,010.07**

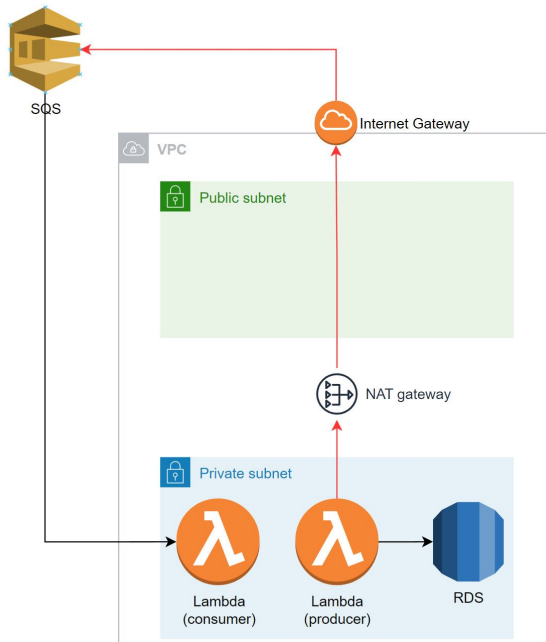
‣ API Gateway	\$0.02
‣ CloudFront	\$0.11
‣ CloudTrail	\$0.00
‣ CloudWatch	\$41.49
‣ CloudWatch Events	\$0.00
‣ CodeBuild	\$1.44
‣ CodeCommit	\$3.00
‣ CodePipeline	\$4.00
‣ Compute Optimizer	\$0.24
‣ Config	\$3.66
‣ Cost Explorer	\$0.92
‣ Data Transfer	\$0.47
‣ Direct Connect	\$0.17
‣ DynamoDB	\$2.51
‣ Elastic Compute Cloud	\$438.13
‣ Glue	\$0.06
‣ GuardDuty	\$33.63
‣ Key Management Service	\$0.35
‣ Lambda	\$88.84
‣ OpenSearch Service	\$333.86
‣ Relational Database Service	\$2.58
‣ Secrets Manager	\$1.27
‣ Simple Email Service	\$0.00
‣ Simple Notification Service	\$0.42
‣ Simple Queue Service	\$0.97
‣ Simple Storage Service	\$3.87
‣ Virtual Private Cloud	\$19.06
‣ WAF	\$29.00

운영 중 발견한 문제(단점)

문제 및 단점

- Custom VPC에서 Lambda 사용시 직접 ENI 삭제가 불가능함
 - <https://aws.amazon.com/ko/premiumsupport/knowledge-center/lambda-eni-find-delete/>
 - ENI는 EC2요금으로 발생해서 요금계산시 헛갈리게 함
- 가끔 CloudFormation이 실패하는 경우가 있음
 - 직접 zappa update 해주면 잘됨.. = Lambda deploy status 문제
- Terraform과 같이 사용시 모듈화할때 경계가 불분명함
 - EX) Cloudwatch에 뭔가 생겼는데 이게 zappa가 만든건가..
 - zappa가 만드는데 뭔지 다 알기 어려움
- zappa 관련 IAM policy 관리의 어려움
 - 개발/운영자 간의 권한 차이가 거의 사라짐
 - 어쩌다보니 devops..
- 구조를 알면 별거 없지만 모르면...
 - 로그 어디서봐요?
 - lambda가 왜 private subnet에 있어요?

SQS in Custom VPC



NAT를 통한 데이터 전송 요금 발생

NAT 게이트웨이 요금

VPC에 NAT 게이트웨이를 생성하는 경우 게이트웨이를 프로비저닝하고 사용하는 각 'NAT 게이트웨이 시간'에 대한 요금이 부과됩니다. 데이터 처리 요금은 트래픽 소스나 대상과 관계없이 NAT 게이트웨이를 통해 처리된 각 기가바이트에 적용됩니다. 1시간 미만의 각 NAT 게이트웨이 사용 시간은 1시간으로 청구됩니다. 또한 NAT 게이트웨이를 통해 전송된 모든 데이터에 대한 표준 AWS 데이터 전송 요금도 발생합니다. NAT 게이트웨이에 대한 요금이 청구되지 않도록 하려면 AWS 관리 콘솔, 명령줄 인터페이스 또는 API를 사용하여 NAT 게이트웨이를 삭제하면 됩니다.

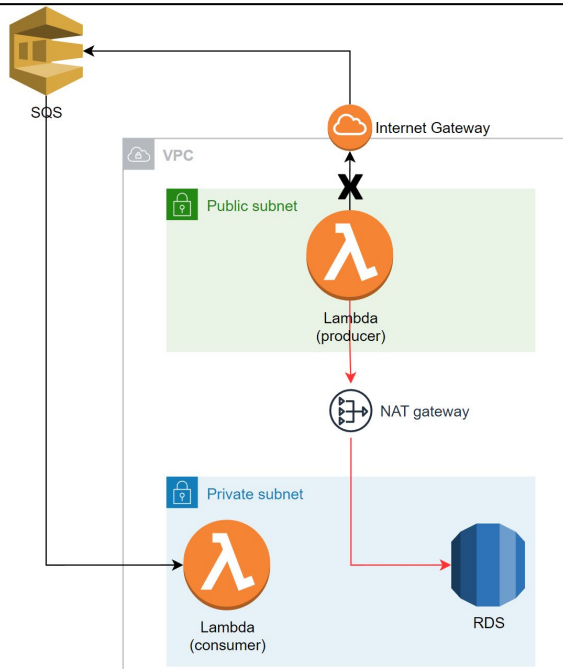
리전: 아시아 태평양(서울) *

NAT 게이트웨이당 요금(USD/시간)

0.059 USD

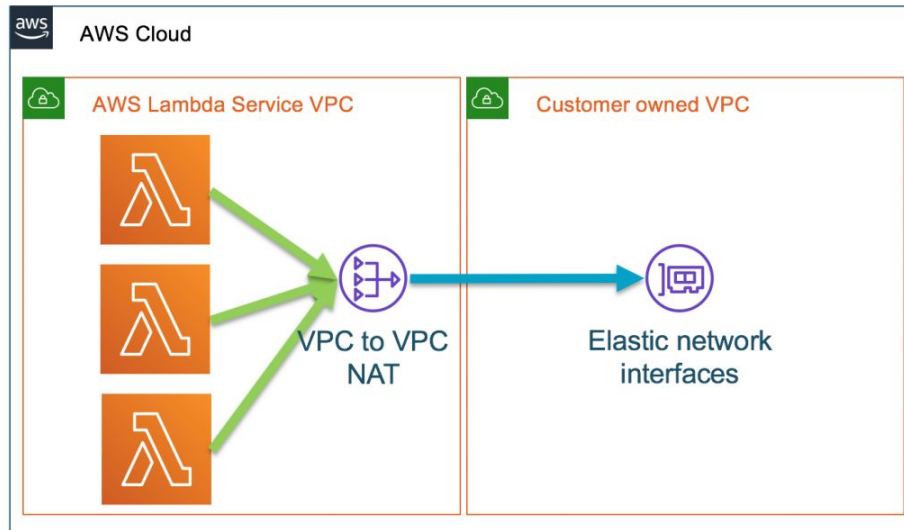
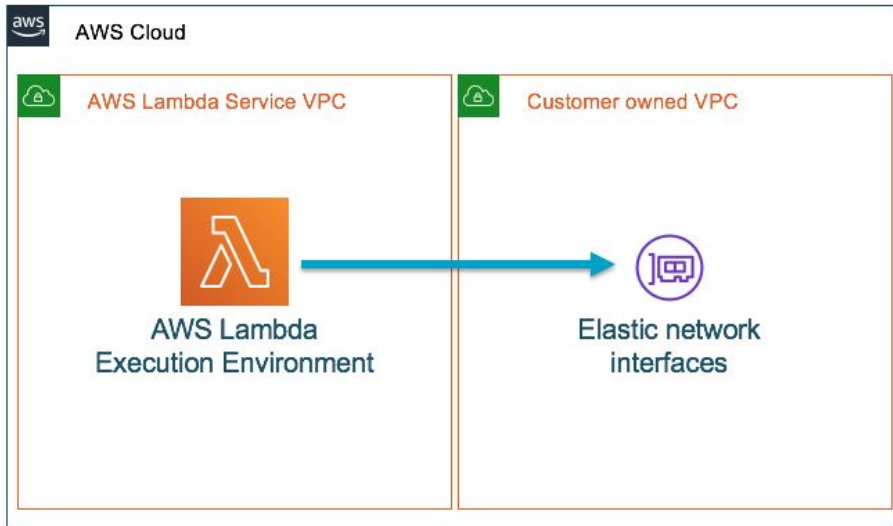
처리된 데이터 GB당 요금(USD)

0.059 USD



Lambda에는 Public IP가 없어서 통신 불가

Lambda in Custom VPC



- Custom VPC에 할당된 Lambda(instance)는 ENI(Elastic Network Interface)가 생성됨
- ENI는 기본적으로 Private IP
- EIP는 할당이 불가능하고 IP를 부여하기 위해선 NAT 사용이 필수
- EC2와 Lambda의 네트워크의 차이점(EIP 할당 여부)

Q&A

감사합니다