KTB클라우드 구축 보고서 문서관리지침

Ver. 1.0

개 정 이 력

버전	작성일	변경사유[1]	변경내용[2]	작성자	승인자
1.0	250327	최초작성	최초 작성	jacky	

[1] 변경 사유: 변경 내용이 이전 문서에 대해 최초작성/승인/추가/수정/삭제 중 선택 기입

[2] 변경 내용: 변경이 발생되는 위치와 변경 내용을 자세히 기록(장.절과 변경 내용을 기술한다.)

목 차

1. 개요

1.1 목적

이 KTB 클라우드 구성보고서는 여러가지 클라우드 인프라 리소스를 실제로 만들어보는 실습을 통해 구성된 요소들을 정리한 것입니다.

프로젝트 구조

프로젝트는 다음과 같이 단계별로 구성되어 있습니다:

1. 01-base-infra: 기본 인프라 구성
   - VPC, 서브넷, 라우팅 테이블 등 네트워크 인프라
   - 보안 그룹, IAM 역할 등 보안 관련 리소스
   - Secrets Manager 통합
1. 02-database: 데이터베이스 레이어
   - RDS MySQL 인스턴스 구성
   - 데이터베이스 연결 정보 관리
1. 03-compute: 컴퓨트 리소스
   - EKS 클러스터 (Kubernetes v1.31)
   - 노드 그룹 구성 (t3.medium 타입, 2-3개 노드)
   - ArgoCD 및 Jenkins 설치 및 구성
1. 04-cicd-apps: CI/CD 및 애플리케이션 배포
   - ECR 저장소 관리
   - ArgoCD를 통한 앱 배포 (App of Apps 패턴)
   - 도메인 및 인그레스 구성

주요 기술 스택

인프라 및 오케스트레이션

- AWS 클라우드 서비스: VPC, EKS, RDS, ECR, Route53, ALB
- Kubernetes: 컨테이너 오케스트레이션
- Terraform: 인프라 코드화(IaC)

CI/CD 및 GitOps

- Jenkins: CI 파이프라인 (백엔드 빌드 및 ECR 배포)
- ArgoCD: CD 및 GitOps 구현 (Kubernetes 매니페스트 동기화)
- GitHub: 소스 코드 및 매니페스트 관리

애플리케이션 아키텍처

- 프론트엔드: SPA 웹 애플리케이션 (tier3-frontend)
- 백엔드: API 서버 (tier3-backend)
- 데이터베이스: MySQL (RDS)

동작 방식

인프라 프로비저닝

1. Terraform을 사용하여 단계별로 인프라 구축 (01 → 02 → 03 → 04)
1. 각 단계는 이전 단계의 원격 상태(S3 버킷에 저장)를 참조하여 의존성 관리

CI/CD 파이프라인

1. 개발자가 코드를 GitHub 레포지토리에 푸시
1. Jenkins가 GitHub 폴링을 통해 변경 감지 (3분마다 체크)
1. Jenkins 파이프라인 실행:
    - 코드 빌드 및 테스트
    - Docker 이미지 생성
    - ECR에 이미지 푸시
1. ArgoCD가 Kubernetes 매니페스트 변경 감지
1. ArgoCD가 Kubernetes 클러스터에 애플리케이션 배포

네트워크 흐름

1. 사용자 요청 → Route53 → ALB → Kubernetes Ingress
1. 프론트엔드 → 백엔드 API → 데이터베이스

도메인 및 접근성

- 메인 도메인: `mydairy.my`
- ArgoCD: `argocd.mydairy.my`
- Jenkins: `jenkins.mydairy.my`
- 프론트엔드: `mydairy.my`

보안 및 관리 측면

- IAM 역할 및 IRSA(IAM Roles for Service Accounts) 구현
- 보안 그룹을 통한 네트워크 접근 제어

- Secrets Manager를 통한 보안 정보 관리
- 태그 기반 리소스 관리

상세 구현 특징

- 고가용성: 다중 가용 영역 구성 (ap-northeast-2a, ap-northeast-2c)
- 자동 확장: 노드 그룹 Auto Scaling 설정 (min 2, max 3)
- 관측 가능성: CloudWatch 통합

2. GitOps: 모든 인프라 및 애플리케이션 상태가 Git 레포지토리에 정의됨

이 프로젝트는 현대적인 클라우드 네이티브 아키텍처를 따르며, 인프라를 코드로 관리하여 재현 가능하고 안정적인 배포를 가능하게 합니다

2.1 범위

OOO 시스템 관련 업무는 운영자 및 개발자의 관점에서 시스템 유지보수에 필요한 내용을 기술하고, 타 시스템 연동과 관련된 배치 업무 및 시스템 운영작업에 필요한 작업수행 절차와 방법 위주로 지침서를 구성한다.

구분	내용	설명
시스템 환경관리	시스템 구조 정의	운영서버의 하드웨어 및 WEB 환경 구성에 필요한 시스템 소프트웨어 사양을 기술하고, 프로그램 설치에 관계되는 Directory 및 사용자 계정 등을 설명한다.
	WEB 서버 관리	WEB 서버 설치, 기동 및 정지방법, 장애발생에 대비한 응급조치 요령을 파악한다.
시스템 운영관리	WEB 유지보수 관리	WEB 프로그램 유지보수를 위한 개발자 환경 설치, 프로그램 변경절차 및 작업방법을 설명한다.
시스템 백업 및 복구	시스템 백업	소스 프로그램 및 D/B 객체를 백업 대상으로 한 백업 수행주기 및 방법을 정의한다.
	복구	시스템 장애 또는 오류발생시 백업된 자료를 복구하기 위한 작업절차 및 방법을 기술한다.
시스템 표준 정의	Naming 표준	시스템 환경설정 및 프로그램 개발에 필요한 요소들의 표준적인 Naming Rule을 정리한다.
	표준	개발표준에 따라 설계한 프로젝트 표준을 참고하여 시스템 유지보수에 활용한다.
기타	WEB 서버 운영 Reference Guide	WEB 서버 운영에 필요한 참조자료 및 관련 사이트 정보를 리스트 형식으로 정리한다.

[표 11] 범위

3. 시스템 환경 관리

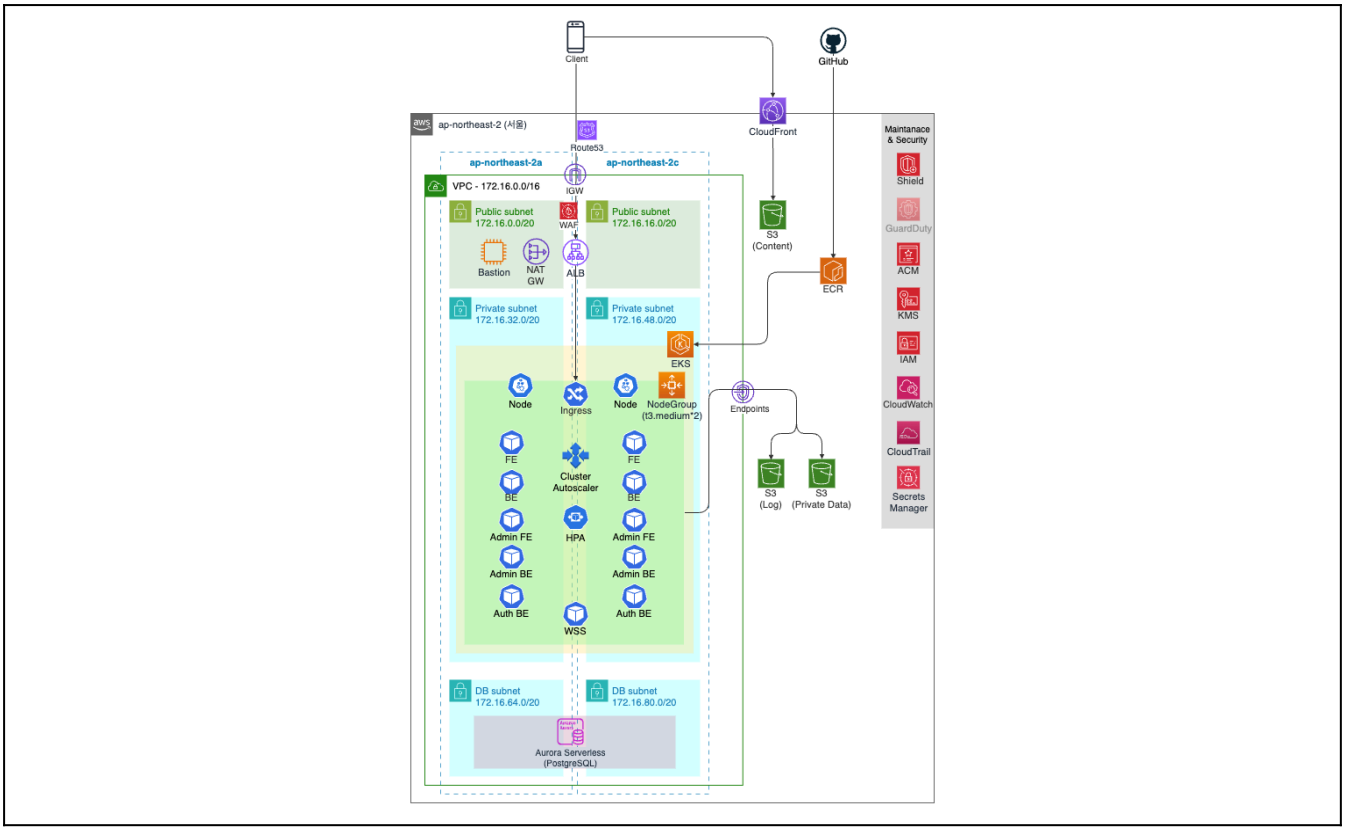
3.1 시스템 구성 정의

OOO 플랫폼은AWS Cloud 인프라 내에 구성되어 있으며, Shared, DEV, PROD 세개의 VPC로 구축되어있어 별도 필요한 구성 외 완전 격리된 환경을 제공한다.

3.1.1 Account 구성

- 단일 IAM 계정으로 구성

3.1.2 시스템 Architecture



[그림 22] 시스템 Architecture구성도

- 클라우드의 경우 최대한 AWS서비스를 활용하여 구축합니다.
- 테라폼 apply 후 백엔드/프론트엔드 ci를 마치면 정상 동작합니다. 다른 과정은 필요 없습니다.
-

3.1.3 운영시스템 인프라 사양

항 목(서버명)	수량	세부항목	사양 (인스턴스타입)
	1	vCPU (2core) / 메모리 (4GB)	c5.large
		디스크(100GB)	
	1	vCPU (4core) / 메모리 (4GB)	c5.large
		디스크(100GB)	
	1	vCPU (2core) / 메모리 (8GB)	m5.large
		디스크(100GB)	
	1	vCPU(4core) / 메모리 (16GB)	m5.xlarge
		디스크(100GB)	
	1	vCPU(2core) / 메모리 (8GB)	m5.large
		디스크(100GB)	

[표 21] 하드웨어 사양

2.1.3.1 운영 시스템 관련 디렉토리

서버 명	Mount Point	용도
	/home/allt/www	소스 폴더
	/var/log/httpd	웹 로그 폴더
	/etc/httpd/conf	Apache 컨피스 폴더
	/etc/httpd/conf/httpd.conf	Apache 설정
	/opt/codedeploy-agent	Code Deploy Agent
	/var/log/aws/codedeploy-agent	Code Deploy Agent Log
	/opt/codedeploy-agent/deployment-root /deployment-logs	AWS Code Deploy Log
	/opt/aws/amazon-cloudwatch-agent	AWS CloudWatch Agent
	/opt/aws/amazon-cloudwatch-agent/log s	AWS CloudWatch Agent Log
서버 명	**Mount Point**	**용도**
	/home/ubuntu/build	소스 폴더
	/opt/codedeploy-agent	Code Deploy Agent
	/var/log/aws/codedeploy-agent	Code Deploy Agent Log
	/opt/codedeploy-agent/deployment-root /deployment-logs	AWS Code Deploy Log
	/opt/aws/amazon-cloudwatch-agent	AWS CloudWatch Agent
	/opt/aws/amazon-cloudwatch-agent/log s	AWS CloudWatch Agent Log
서버 명	**Mount Point**	**용도**
	/home/allt/saleson-api	소스 폴더

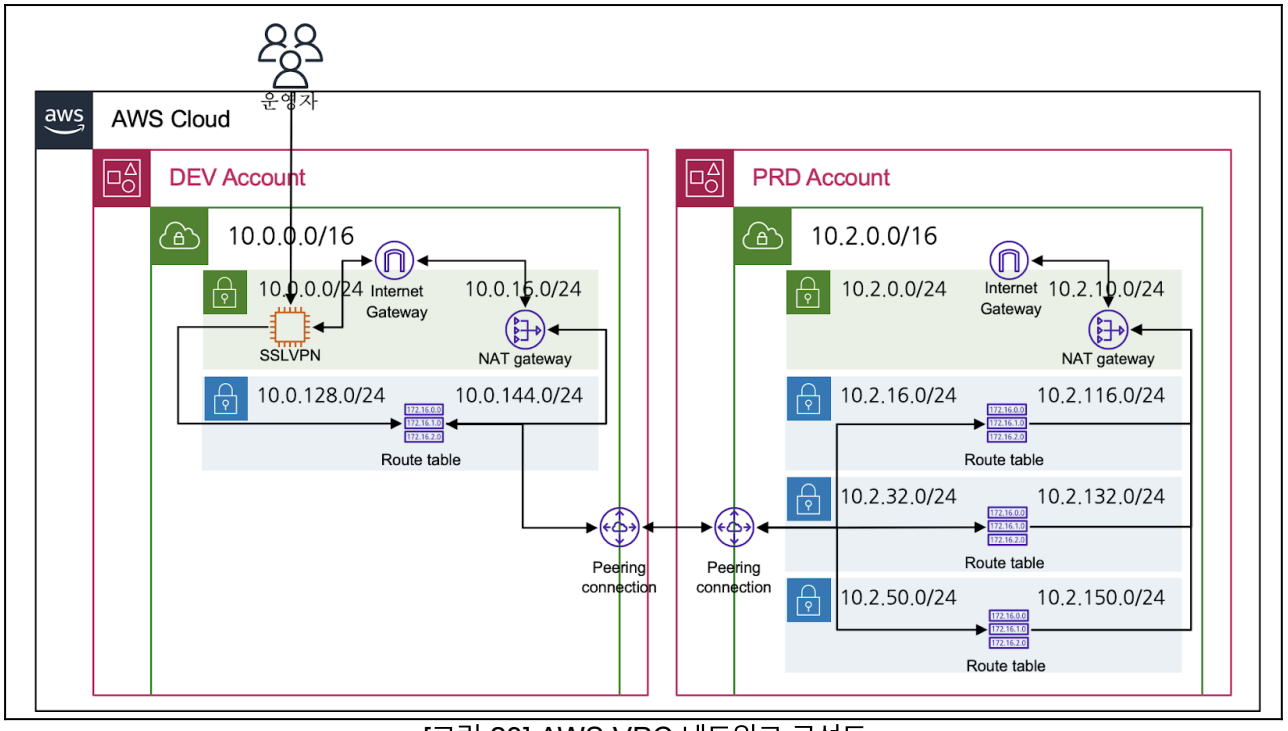
	/home/allt	Tomcat start.sh / shutdown.sh
	/home/alltm/logs	Tomcat 로그 폴더
	/opt/codedeploy-agent	Code Deploy Agent
	/var/log/aws/codedeploy-agent	Code Deploy Agent Log
	/opt/codedeploy-agent/deployment-root/deployment-logs	AWS Code Deploy Log
	/opt/aws/amazon-cloudwatch-agent	AWS CloudWatch Agent
	/opt/aws/amazon-cloudwatch-agent/logs	AWS CloudWatch Agent Log
서버 명	**Mount Point**	**용도**
	/home/allt/saleson-app	소스 폴더
	/home/allt/tomcat	톰캣 폴더
	/home/allt/tomcat/logs	톰캣 로드 폴더
	/opt/codedeploy-agent	Code Deploy Agent
	/var/log/aws/codedeploy-agent	Code Deploy Agent Log
	/opt/codedeploy-agent/deployment-root/deployment-logs	AWS Code Deploy Log
	/opt/aws/amazon-cloudwatch-agent	AWS CloudWatch Agent
	/opt/aws/amazon-cloudwatch-agent/logs	AWS CloudWatch Agent Log
	/home/allt/ALLT_MIDDLEWARE_SOKETIO	소스 폴더
	/opt/codedeploy-agent	Code Deploy Agent
	/var/log/aws/codedeploy-agent	Code Deploy Agent Log
	/opt/codedeploy-agent/deployment-root/deployment-logs	AWS Code Deploy Log
	/opt/aws/amazon-cloudwatch-agent	AWS CloudWatch Agent
	/opt/aws/amazon-cloudwatch-agent/logs	AWS CloudWatch Agent Log

[표 22] 운영 시스템 관련 디렉토리

3.1.4 네트워크 구성

OOO플랫폼 시스템에서 사용하는 네트워크 구성 및 IP정보는 다음과 같다.

2.1.4.1 VPC subnet IP CIDR 및 Peering 구성



[그림 23] AWS VPC 네트워크 구성도

- Network의 경우 4Tier 구성으로 보안성을 확보한다.
- Public Subnet의 경우 OpenVPN , NATGW, InternetGW, Loadbalancer만 구성한다.
- WEB Private의 경우 Front-End, WAS Private의 경우 Back-End, DB Private의 경우 DBMS만 구성한다.
- WEB/WAS Private 의 서브넷의 경우 NATG를 통해서만 공용망과 연결 된다.
-DB Private의 경우 공용망과 단절되며, 아마존 Local망에서만 접근 된다.
- 공용망통신은 OpenVPN을 통해 암호화 통신으로 이루어지며 Peering을 통해 라우팅 된다.

2.1.4.2 Security Group List (방화벽 규칙)

보안 그룹 규칙 ID	IP 버전	유형	프로토콜	포트 범위	소스	설명	비고
sg-03a1d448cc0ede277	IPv4	SSH	TCP	22	0.0.0.0		-
sg-03a1d448cc0ede277	IPv4	SSH	TCP	80	0.0.0.0		-
sg-03a1d448cc0ede277	IPv4	SSH	TCP	443	0.0.0.0		-
sg-0495b842fb4ea2fd6	IPv4	SSH	TCP	22	0.0.0.0		-
sg-0495b842fb4ea2fd6	IPv4	SSH	TCP	9080	0.0.0.0		-
sg-0495b842fb4ea2fd6	IPv4	SSH	TCP	8080	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	22	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	80	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	8080	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	22	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	80	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	8080	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	22	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	80	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	5601	0.0.0.0		-
sg-03904ed414187b7f2	IPv4	SSH	TCP	9090	0.0.0.0		-
sg-04006d8f6c6aea56b	IPv4		TCP	3306	10.0.0.0/16		-

[그림 24] AWS Security Group 정책표

- 각 인스턴스 별로 각각의 Security Group(방화벽)과 매칭 된다.
- Security Grou(방화벽)의 경우 사용하는 포트만 오픈 되어 있다.

2.1.4.3 시스템 네트워크 구성

서비스	Host Name	IP
EC2		
EC2		
EC2		
EC2		
EC2		
LoadBalancer		
LoadBalancer		
LoadBalancer		
RDS		

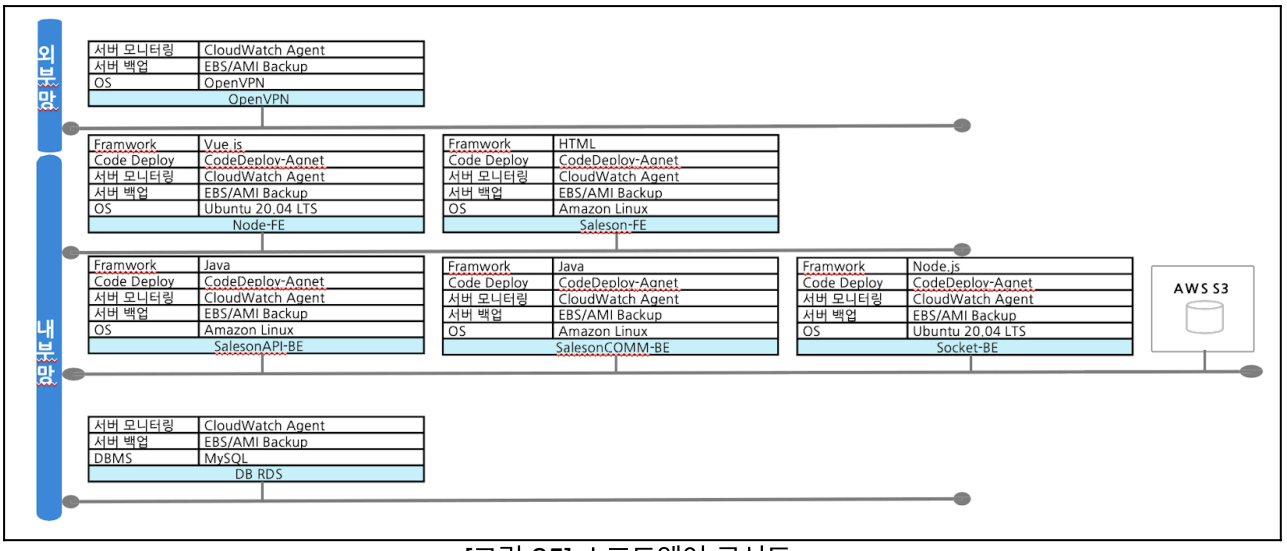
[표 23] AWS Resource Interfaces 구성

2.1.4.4 서비스 접속 URL

항목	URL	Port
1	www.kakao.com	80
2	backend.kakao.com	8080
3		8080
4		9080

[표 24] 서비스 URL 및 Port 정보

3.1.5 소프트웨어 구성도



[그림 25] 소프트웨어 구성도

- SSL인증서는 ACM에 등록되어 있으며, ALB에 적용되어 HTTPS 통신으로 처리됨.
- 각 서버 별 로그 및 홈페이지 이미지의 경우 S3를 서버에 마운트 하여 저장함(논의필요).

2.1.6 Application Server 운영 일반

- SalesonAPI-BE 서버 시작/종료

- SSH 접속
: alltmro 계정으로 접속한다.

- Tomcat Start 명령어
: sh ./startup.sh

- Tomcat Stop 명령어
: sh ./shutdown.sh

- process 확인 명령어
: ps –ef | grep tomcat

- SalesonAPI-BE Log 정보 조회

- SalesonAPI-BE Log 보기 명령어
: tail -n50 -f /home/allt/logs/saleon-api.log

- SalesonManager-BE 서버 시작/종료

- SSH 접속
: alltmro 계정으로 접속한다.

- Tomcat Start 명령어
: cd /home/allt/tomcat/bin
: sh ./startup.sh

- Tomcat Stop 명령어
: cd /home/allt/tomcat/bin
: sh ./shutdown.sh

- process 확인 명령어
: ps –ef | grep tomcat

- SalesonManager-BE Log 정보 조회

- SalesonAPI-BE Log 보기 명령어
: cd **/home/allt/tomcat/logs**
: tail -n50 -f /home/allt/tomcat/logs/catalina.2023-01-02.log(해당 년월일)
: tail -n50 -f /home/allt/tomcat/logs/localhost_access.2023-01-02.txt(해당 년월일)

- Saleson-FE 서버 시작/종료

- SSH 접속
: alltmro 계정으로 접속한다.

- Apache Start 명령어
: System start httpd

- Apache Stop 명령어
: System Stop httpd

- process 확인 명령어
: ps –ef | grep httpd

- Saleson-FE Log 정보 조회

- Saleson-FE Log 보기 명령어
: cd /var/log/httpd
: tail -n50 -f /var/log/httpd/access_log
: tail -n50 -f /var/log/httpd/error_log

- Node-FE 서버 시작/종료

- SSH 접속
: alltmro 계정으로 접속한다.

- NPM Start 명령어
: cd /home/ubuntu/build
: npm run serve

- process 확인 명령어
: ps -ef | grep node

- Socket-BE 서버 시작/종료

- SSH 접속
: alltmro 계정으로 접속한다.

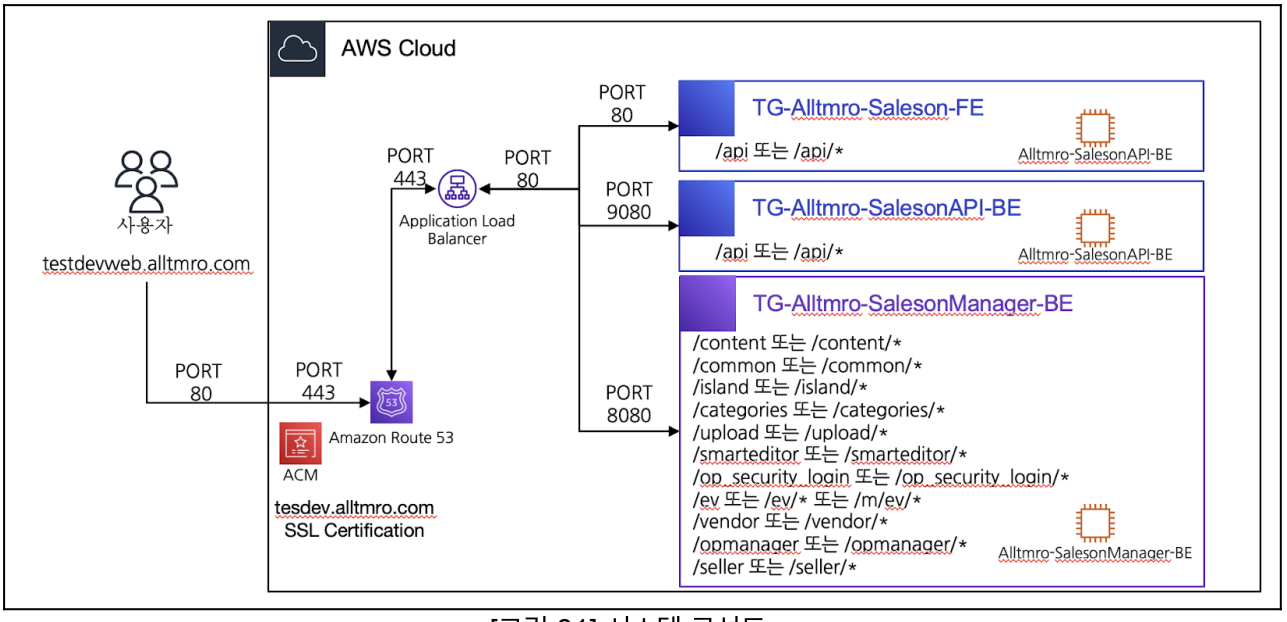
- NPM Start 명령어
: cd /home/allt/ALLT_MIDDLEWARE_SOKETIO
: node index.js

- NPM Stop 명령어
: cd /home/allt/ALLT_MIDDLEWARE_SOKETIO
: sh ./node.sh

- process 확인 명령어
: ps -ef | grep node

4. 시스템 운영 관리

4.1 Application Flow (Application Loadbalaner Forwarding)구성도



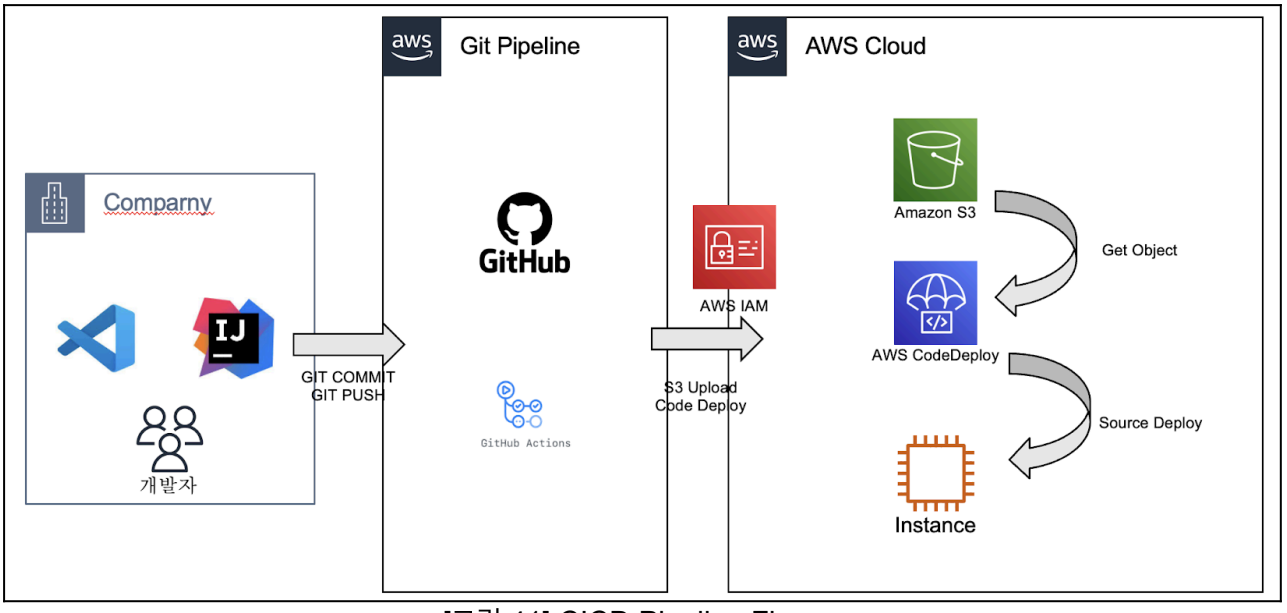
[그림 31] 시스템 구성도

5. CICD 운영 관리

5.1 CICD Repository

서비스	Host Name	EndPoint
Github		
Github		
Github		
Github		
Github		
Github		
Github		
S3		
S3		
S3		
S3		
S3		
S3		
S3		
CodeDeploy		
CodeDeploy		
CodeDeploy		
CodeDeploy		
CodeDeploy		

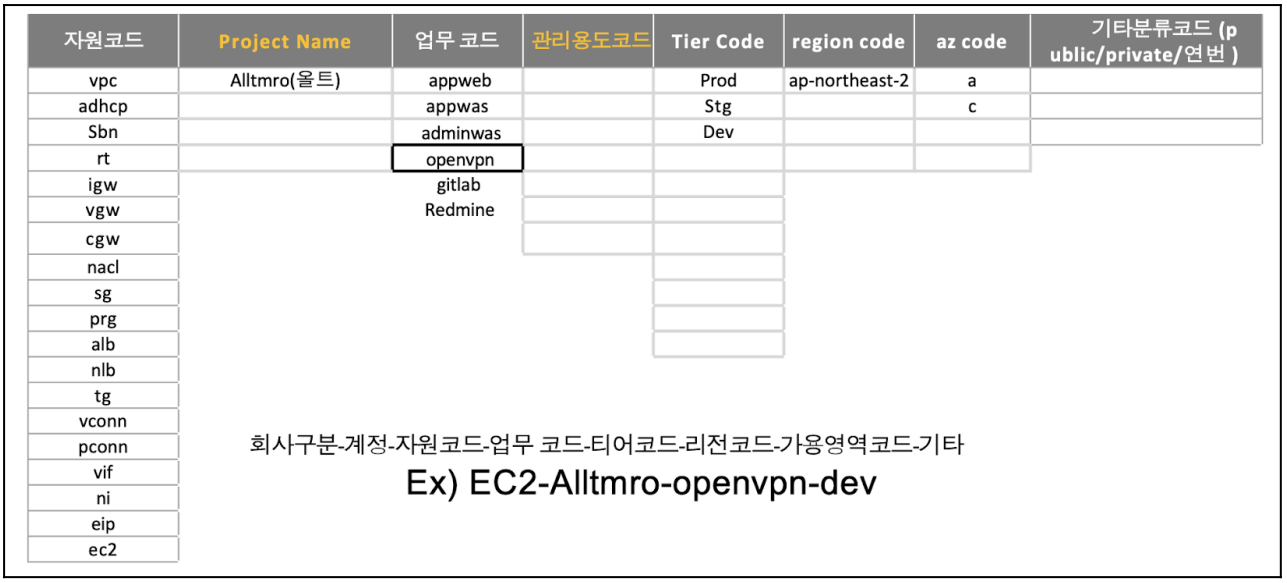
5.2 S3 + Codedeploy



[그림 41] CICD Pipeline Flow

6. 시스템 표준

6.1 AWS Resources Tagging 표준



자원코드	Project Name	업무 코드	관리용도코드	Tier Code	region code	az code	기타분류코드 (public/private/연번)
vpc	Alltmro(올트)	appweb		Prod	ap-northeast-2	a	
adhcp		appwas		Stg		c	
Sbn		adminwas		Dev			
rt		openvpn					
igw		gitlab					
vgw		Redmine					
cgw							
nacl							
sg							
prg							
alb							
nlb							
tg							
vconn							
pconn							
vif							
ni							
eip							
ec2							

[그림 61] AWS Resources Tagging

- Tag는 고객사나 AWS가 AWS 리소스에 할당하는 레이블로, Key - Value로 구성됨
- Tag의 Key와 Value는 대소문자를 구분함

Tag 사용 목적	설명
AWS console	AWS 관리 콘솔에서 태그를 사용하여 AWS 자원들을 조직화 자원들과 함께 표시되도록 태그를 구성하고 태그별로 검색 및 필터링 가능
자동화	자동화 태그는 자동화된 작업을 선택 또는 해제하거나 소스 배포 등의 자원을 식별하는데 사용함.
백업	백업 정책 설정 시 특정 서버 단위로 Tag를 통해 적합한 자원만 적용하여 실행하는데 사용

[표 61] Tag 사용 목적

-Tagging 규칙

Tag		
Name	리소스 명	EC2-alltmro-openvpn-prod...
Owner	리소스를 관리하는 주체가 되는 팀	dev_infar_alltmro
System	AWS 리소스에 의해 운영되는 서비스	ec2, vpc, subnet...
Env	리소스 배포 환경	dev, stg, prod
Backup	백업 리소스 특정 명	front-end, back-end, admin, etc

[표 62] Tagging 규칙