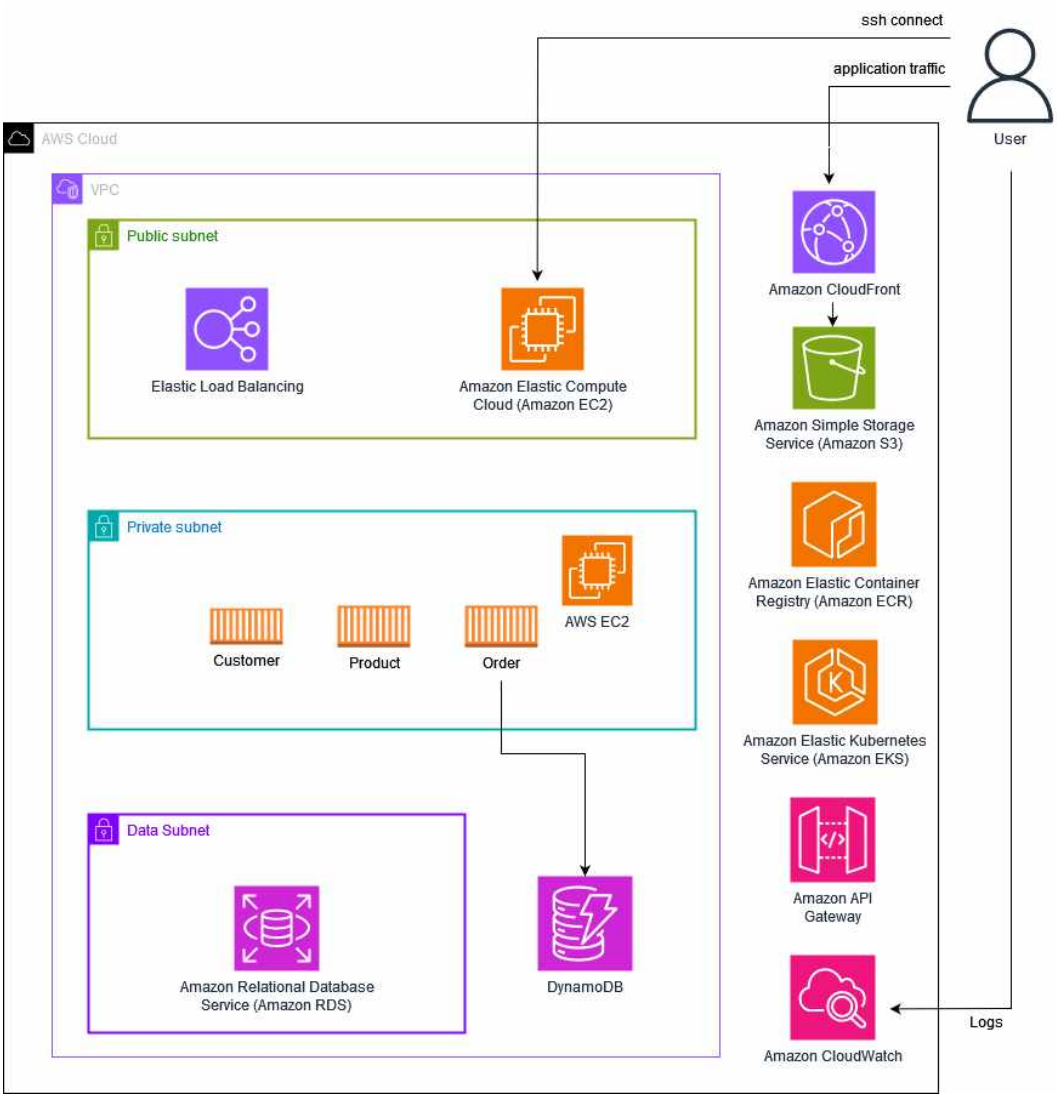
2024년도 전국기능경기대회

직 종 명	클라우드컴퓨팅	과 제 명	Web Service Provisioning	과제번호	제 1과제
경기시간	4시간	비 번 호		심사위원 확 인	(인)

1. 요구사항

당신은 J&H Company의 DevOps engineer로 제공하는 솔루션을 활용하여 웹서비스를 운영이 가능한 Infrastructure를 설계하고 구축해 운영하는 것이 목표입니다. 주어진 요구사항과 클라우드의 설계원칙인 고가용성, 확장성, 비용, 보안 등을 잘 고려하여 웹 애플리케이션이 구동할 수 있는 인프라를 구축합니다.

다이어그램



Software Stack

AWS		개발언어/프레임워크
- VPC	- RDS	- Golang / Gin
- EC2	- API Gateway	- HTML/CSS
- ELB	- CloudWatch	- Javascript
- Cloudfront	- DynamoDB	
- S3		
- ECR		
- EKS		

2. 선수 유의사항

1) 기계 및 공구 등의 사용 시 안전에 유의하시고, 필요 시 안전장비 및 복장 등을 착용하여 사고를 예방하여 주시기 바랍니다.
2) 작업 중 화상, 감전, 찰과상 등 안전사고 예방에 유의하시고, 공구나 작업도구 사용 시 안전보호구 착용 등 안전수칙을 준수하시기 바랍니다.
3) 작업 중 공구의 사용에 주의하고, 안전수칙을 준수하여 사고를 예방하여 주시기 바랍니다.
4) 경기 시작 전 가벼운 스트레칭 등으로 긴장을 풀어주시고, 작업도구의 사용 시 안전에 주의하십시오.
5) 선수의 계정에는 비용의 제한이 존재하며, 이보다 더 높게 요금이 부과될 시 계정 사용이 불가능할 수 있습니다.
6) 문제에 제시된 괄호 박스 <>는 변수를 뜻함으로 선수가 적절히 변경하여 사용해야 합니다.
7) EC2 인스턴스의 TCP 80/443 outbound는 anyopen하여 사용할 수 있도록 합니다.
8) 과제의 Bastion 서버에서 대부분의 채점이 이루어짐으로 인스턴스를 생성하지 않았거나 종료된 상태면 채점이 불가능하니 각별히 주의하도록 합니다.
9) 과제 종료 시 진행 중인 테스트를 모두 종료하여 서버에 부하가 발생하지 않도록 합니다.
10) 별도 언급이 없는 경우, ap-northeast-2 리전에 리소스를 생성하도록 합니다.
11) 모든 리소스의 이름, 태그, 변수과 변수는 대소문자를 구분합니다.
12) 불필요한 리소스를 생성한 경우, 감점의 요인이 될 수 있습니다. (e.g. VPC 추가 생성)
13) Bastion 서버로 ssh 접속시 22번 포트 외에 다른 포트를 사용하세요.
14) Bastion을 비밀번호 설정 시 비밀번호를 Skills2024** 로 설정합니다.

3. 네트워크 구성

VPC를 생성하여 클라우드 네트워킹을 구성합니다. HA를 고려하여 최소 3개의 가용 영역(AZ)을 가지도록 VPC를 설계합니다. VPC의 네트워크 트래픽을 로깅 할 수 있게 구성 하고 로그 이름과 그룹 이름은 wsi-traffic-logs로 구성합니다.

VPC 정보

- VPC CIDR : 10.0.0.0/16
- VPC Tag : Name=wsi-vpc
- Internet G/W Tag : Name=wsi-igw

Public subnet A 정보

- CIDR : 10.0.11.0/24
- Tag : Name=wsi-public-a
- 자동 IPv4 주소 할당 활성화 시킵니다
- 외부 통신 : Internet G/W 를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-public-rt

Public subnet B 정보

- CIDR : 10.0.12.0/24
- Tag : Name=wsi-public-b
- 자동 IPv4 주소 할당 활성화 시킵니다
- 외부 통신 : Internet G/W 를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-public-rt

Public subnet C 정보

- CIDR : 10.0.13.0/24
- Tag : Name=wsi-public-c
- 자동 IPv4 주소 할당 활성화 시킵니다
- 외부 통신 : Internet G/W 를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-public-rt

Private subnet A 정보

- CIDR : 10.0.101.0/24
- Tag : Name=wsi-private-a
- 외부 통신 : NAT G/W 를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-private-a-rt

Private subnet B 정보
- CIDR : 10.0.102.0/24
- Tag : Name=wsi-private-b
- 외부 통신 : NAT G/W 를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-private-b-rt

Private subnet C 정보
- CIDR : 10.0.103.0/24
- Tag : Name=wsi-private-c
- 외부 통신 : NAT G/W 를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-private-c-rt

Protected subnet A 정보
- CIDR : 10.0.201.0/24
- Tag : Name=wsi-data-a
- 외부 통신 : 인터넷 접근이 필요하지 않음
- Route table Tag : Name=wsi-protected-rt

Protected subnet B 정보
- CIDR : 10.0.202.0/24
- Tag : Name=wsi-data-b
- 외부 통신 : 인터넷 접근이 필요하지 않음
- Route table Tag : Name=wsi-protected-rt

Protected subnet C 정보
- CIDR : 10.0.203.0/24
- Tag : Name=wsi-data-c
- 외부 통신 : 인터넷 접근이 필요하지 않음
- Route table Tag : Name=wsi-protected-rt

로그 정보:
- 이름: wsi-traffic-logs
- 필터: 모든 허용
- 로그 그룹: wsi-traffic-logs
- 로그 포맷: ${region} ${vpc-id} ${action} ${instance-id}

4. Bastion

EC2를 활용해 Bastion 서버를 구성합니다. Bastion 서버는 외부에서 SSH 프로토콜만을 허용하도록 Security Group을 구성합니다. Bastion은 로그인 및 로그아웃 할 때 아래에 로그 예시와 같이 CloudWatch Logs에 저장 할 수 있게 구성합니다. Bastion은 재시작을 해도 Public IP 주소가 변경하지 않도록 구성합니다.

- Instance Type: m5.Large
- OS: Amazon Linux 2023 AMI
- Subnet: Public Subnet C
- 설치 패키지: awscli, jq, curl, git
- Tag: Name=wsi-bastion-ec2, ec2=bastion
- Security Group 이름: Name=wsi-bastion-SG
- EC2 IAM Role: IAM의 User과 Group을 관리 할수 없지만 모든 리소스에 대해 full access를 가지는 권한과 IAM 읽기 전용 권한을 붙입니다. role 이름은 wsi-bastion-role로 설정합니다.(권한은 직접 생성 하지 않습니다. 제공 되어 있는 권한만 사용.)
- 로그 포맷: { <인스턴스 사용자 이름>, <인스턴스 사용자 이름> has logged in!, <yyyy-mm-dd HH:MM:SS 날짜/한국 시간> }
- 로그 예시:
  - 로그인: { ec2-user, ec2-user has logged in!, 2024-05-22 14:18:11 }
  - 로그아웃: { ec2-user, ec2-user has logged out!, 2024-05-22 15:11:02 }
- 로그 그룹 이름: wsi-bastion-user-logs
- 로그 스트림 이름: wsi-bastion-stream

6. S3

모든 콘텐츠 파일은 /frontend/ 접두사를 가진 오브젝트 키 (예시: /frontend/index.html)로 업로드 합니다.

- S3 버킷 이름: wsi-cc-data-<선수번호>-<임의 4자리 영문>

7. Control Plane

클라우드 환경에서 운영되는 수많은 컨테이너 및 워크로드들을 일일이 관리하기에는 비효율적입니다. Kubernetes를 사용해서 수많은 컨테이너, 워크로드들을 관리 및 운영하고자 합니다. 아래의 요구사항을 참고하여 구성합니다. Amazon에서 Kubernetes를 관리하는 서비스인 EKS를 생성 및 구성하기 위해서 Control Plane 인스턴스를 생성합니다. 해당 인스턴스에서는 kubectl과 eksctl을 사용했을 때 Kubernetes에 관련된 모든 명령어가 정상 적으로 실행이 되도록 최소한의 권한을 부여합니다. user과 dev이라는 리눅스 계정을 생성합니다. 해당 인스턴스는 Bastion 인스턴스를 통해서만 SSH 접속 가능해야 합니다. user 계정은 평소에는 EKS 접근 불가 하지만, "user"라는 IAM role로 assume 후에는 kubectl을 통해서 skills라는 네임스페이스에서 조회만 가능합니다. dev 계정은 평소에는 EKS 접근 불가 하지만, "developer"라는 IAM role로 assume 후에는 kubectl

을 통해서 skills라는 네임스페이스에서 조회 및 삭제만 가능합니다. AWS Account의 Root 유저는 모든 namespace에 오브젝트를 생성, 수정, 삭제 등의 작업이 가능해야합니다. 리눅스 사용자들은 필요한 패키지(설치 패키지)들 설치 하도록 구성합니다.

- Instance Type: c5.Large
- OS: Amazon Linux 2023 AMI
- Subnet: Private Subnet A
- 설치 패키지: awscli, jq, curl, git, eksctl, kubectl
- Tag: Name=wsi-control-plane
- Security Group 이름: Name=wsi-control-plane-SG
- EC2 IAM Role 이름: wsi-control-plane-role
- SSH 포트 번호: 3817

8. ECR

어플리케이션을 컨테이너로 만든 후 쿠버네티스에 배포해야 합니다. 먼저 컨테이너 이미지를 생성한 후 ECR에 업로드하도록 합니다. ECR 타입은 Private으로 생성합니다. 컨테이너가 쿠버네티스에서 실행되어 kubectl exec 명령어로 ssh 접근 시 customer app은 customer라는 해당 유저로 실행되어야 하고, order이면 order 유저, product이면 product 유저로 실행되어 합니다. 컨테이너 이미지에서 curl와 stress 명령어를 사용할 수 있도록 패키지를 설치해 둡니다.

- Customer:
  - ECR 이름: wsi-customer-ecr
  - User 이름: customer
- Order:
  - ECR 이름: wsi-order-ecr
  - User 이름: order
- Product:
  - ECR 이름: wsi-product-ecr
  - User 이름: product

9. EKS

어플리케이션을 실해하기 위해 쿠버네티스를 사용하고자 합니다. 하지만, 클라우드 환경에 직접 쿠버네티스를 관리 및 구성하는 것이 매우 불편합니다. 따라서 AWS에서 쿠버네티스를 따로 설치 및 관리 필요 없이 편하게 사용할 수 있게 제공된 서비스인 EKS를 사용합니다. EKS 클러스터를 생성합니다. 2개의 노드그룹을 생성하고 이름은 wsi-app-ng, wsi-addon-ng으로 설정합니다. 어플리케이션은(customer, order, product) wsi-app-ng 노드그룹에 배포하고, addon(ALB Controller, Calico 등)은 wsi-addon-ng 노드그룹에 배포해야합니다. 어플리케이션 Pod에는 wsi:skills 라벨을 추가합니다. 리소스(pods, ingress 등) 들은 모두 skills라는 namespace에 존재해야 합니다.

네트워크 격리를 위해 Calico를 사용합니다. 애플리케이션 컨테이너끼리는 쿠버네티스 네트워크 내에서 서로 통신이 불가능해야 합니다. 외부로의 DNS, HTTP, HTTPS 접근은 허용합니다.

- EKS 이름: wsi-cluster
- EKS Logging: Control Plane 모두 5가지 로그가 저장되어야 합니다.
- EKS Version: 1.29
- Node Group Name: wsi-app-ng, wsi-addon-ng
- Node EC2 Instance 이름: wsi-app-instance, wsi-addon-instance
- Node Group Subnet: private subnets
- Node Group EC2 Type: t3.medium
- Node min size: 2 (워크로드가 없을시 2개로 유지되도록 해야 함)
- Node Label: "wsi/node: app", "wsi/node: addon"
- Pod Label: "wsi:skills"

10. Deployment

ECR에 업로드된 애플리케이션을 쿠버네티스 환경에 컨테이너를 관리 및 배포하기 위한 deployment와 service를 구성 및 생성합니다.

Customer:

- Deployment 이름: wsi-customer-deployment
- Service 이름: wsi-customer-service

Order:

- Deployment 이름: wsi-order-deployment
- Service 이름: wsi-order-service

Product:

- Deployment 이름: wsi-product-deployment
- Service 이름: wsi-product-service

11. LB

어플리케이션들의 Ingress 생성 시 자동으로 ALB가 생성되어야 합니다. ALB를 구성하여 외부에서 애플리케이션으로 접근 할 수 있도록 구성합니다. 보안을 위하여 로드밸런서를 통해 접근 시 주어진 /v1/* API외에 다른 호출은 403으로 응답코드를 내려 주도록 하고 "Forbidden"이라는 메시지를 반환하도록 구성합니다. 약간의 성능 저하를 감수하고 보안성을 향상시키기 위해 외부에서 CloudFront를 통해 ALB에 접근 할 수 있도록 구성합니다. ALB는 LOR 알고리즘으로 구성합니다. ALB는 CloudFront 외에는 HTTP 접근 허용하지 않도록 구성합니다.

- Ingress 이름: wsi-ingress
- Networking Facing: Internet-Facing
- Listen: HTTP 80
- Tag: Name=wsi-alb

12. 관계형 데이터베이스

애플리케이션들의 데이터 저장 및 작동을 위해 관계형 데이터베이스를 구성합니다. 애플리케이션은 MySQL 데이터베이스 및 테이블에 저장할 수 있습니다. 애플리케이션 동작 설명을 기반으로 쿼리에 지연이 생기지 않도록 테이블을 설계하고 생성하세요.

- rds id identifier: wsi-rds-instance
- instance type: db.t3.micro
- Engine: MySQL 8.0.36 (MySQL Community)
- Port: 3306
- database name: skills
- table 정보:

"customer" 테이블:

Column	Data Type	ETC
id	VARCHAR(255)	-
name	VARCHAR(255)	-
gender	VARCHAR(255)	-

"product" 테이블:

Column	Data Type	ETC
id	VARCHAR(255)	-
name	VARCHAR(255)	-
category	VARCHAR(255)	-

13. NoSQL 데이터베이스

order 애플리케이션의 데이터를 저장하기 위해서 NoSQL 데이터베이스인 DynamoDB를 구성합니다. order 애플리케이션 동작 설명을 기반으로 테이블을 설계하고 생성해야 합니다.

- name: wsi-dynamodb
- mode: on-demand

"order" 테이블:

Column	Data Type	ETC
id	String	PK
customerid	String	-
productid	String	-

14. CloudFront

CloudFront를 통하여 정적 콘텐츠 및 애플리케이션에 접근이 가능하도록 합니다. S3에 업로드 되는 정적 콘텐츠를 캐싱할 수 있도록 구성합니다. ALB로의 요청에 대해서는 캐싱하지 않고 Query String도 모두 Origin으로

전달해야 합니다. 사용자가 CloudFront에 HTTP 접근 시에도 HTTPS로 리다이렉션 되어 HTTPS로만 접근할 수 있도록 구성합니다.
Lambda@edge를 이용해서 /preview?img=<이미지 이름> 으로 호출 시 s3 presigned url을 반환하도록 합니다. presigned url은 3분 후에 만기되어야 합니다

- Origin: S3와 ALB 2개의 origin을 가지도록 구성합니다.
- Origin Path: /frontend
- Edge: 한국뿐만 아니라 전 세계의 유저가 빠른 속도로 접근할 수 있도록 구성합니다.
- Tag: Name=wsi-cdn
- 기타: 채점 시 오동작 예방으로 IPv6는 비활성화하고, 하나의 CloudFront만 생성

Path	Method	Query String
/preview	GET	?img=test.png

15. 웹 애플리케이션

product, order, customer 3개의 애플리케이션이 있습니다. 제공된 바이너리는 x86기반 EC2의 Amazon2에서 빌드하고 동작을 확인하였습니다. 애플리케이션 실행 시 바인딩되는 포트는 TCP/8080입니다. 모든 애플리케이션은 /healthcheck 경로로 상태 확인 할 수 있습니다.

- customer

Path	Method	Request Format
/v1/customer	GET	Query String
		?id=xxxxxxx
/v1/customer	POST	Request Body
		'{"id":"xxxxxx","name":"xxxxxxx","gender":"xxxxxx"}'

Environment Key	Description
MYSQL_USER	RDBMS 연결에 사용할 사용자명
MYSQL_PASSWORD	RDBMS 연결에 사용할 사용자 암호
MYSQL_HOST	RDBMS 연결에 사용할 호스트 이름
MYSQL_PORT	RDBMS 연결에 사용할 포트번호
MYSQL_DBNAME	RDBMS 연결에 사용할 데이터베이스 이름

- product

Path	Method	Request Format
/v1/product	GET	Query String
		?id=xxxxxxx

/v1/product	POST	Request Body
		'{"id":"xxxxxx","name":"xxxxxxx","category":"xxxxxx"}'

Environment Key	Description
MYSQL_USER	RDBMS 연결에 사용할 사용자명
MYSQL_PASSWORD	RDBMS 연결에 사용할 사용자 암호
MYSQL_HOST	RDBMS 연결에 사용할 호스트 이름
MYSQL_PORT	RDBMS 연결에 사용할 포트번호
MYSQL_DBNAME	RDBMS 연결에 사용할 데이터베이스 이름

- order

Path	Method	Request Format
/v1/order	GET	Query String
		?id=xxxxxxx
/v1/order	POST	Request Body
		'{"id":"xxxxxx","customerid":"xxxxxxx","productid":"xxxxxx"}'

Environment Key	Description
AWS_REGION	DynamoDB 연결에 사용할 리전 코드 (e.g. ap-northeast-2)