

# 전국기능경기대회 채점기준

## 1. 채점 시 유의사항

직 종 명

클라우드컴퓨팅

※ 다음 사항을 유의하여 채점하십시오.

채점하는 서버가 선수의 서버가 맞는지 확인합니다. 순서대로 채점을 진행하며 리소스 삭제시 이후 채점이 불가할 수 있으니 신중히 진행하도록 합니다.

※ 채점기준 양식은 반드시, 양식에 맞추어 작성해야 합니다.(채점사이트 입력에 필요)

2. 채점기준표

| 1) 주요항목별 배점 |          |               |    | 직 종 명 |    | 클라우드컴퓨팅   |           |    |
|-------------|----------|---------------|----|-------|----|-----------|-----------|----|
| 과제<br>번호    | 일련<br>번호 | 주요항목          | 배점 | 채점방법  |    | 채점시기      |           | 비고 |
|             |          |               |    | 독립    | 합의 | 경기<br>진행중 | 경기<br>종료후 |    |
| 제1과제        | 1        | Network       | 5  | ○     |    |           | ○         |    |
|             | 2        | Container     | 11 | ○     |    |           | ○         |    |
|             | 3        | Data Services | 5  | ○     |    |           | ○         |    |
|             | 4        | CI/CD         | 9  | ○     |    |           | ○         |    |
| 합 계         |          |               | 30 |       |    |           |           |    |

## 2) 채점방법 및 기준

| 과제<br>번호 | 일<br>련<br>번<br>호 | 주요항목          | 일련<br>번호           | 세부항목(채점방법)                      | 배점  |
|----------|------------------|---------------|--------------------|---------------------------------|-----|
| 제1과제     | 1                | Network       | 1                  | Hub VPC                         | 0.5 |
|          |                  |               | 2                  | Hub VPC Subnet                  | 0.5 |
|          |                  |               | 3                  | Hub VPC Routing Table           | 0.5 |
|          |                  |               | 4                  | Hub VPC Internet Access         | 0.5 |
|          |                  |               | 5                  | Application VPC                 | 0.5 |
|          |                  |               | 6                  | Application VPC Subnet          | 0.5 |
|          |                  |               | 7                  | Application VPC Routing Table   | 0.5 |
|          |                  |               | 8                  | Application VPC Internet Access | 0.5 |
|          |                  |               | 9                  | Transit Gateway                 | 0.5 |
|          |                  |               | 10                 | Transit Gateway Attachment      | 0.5 |
|          | 2                | Container     | 1                  | EKS Cluster                     | 0.5 |
|          |                  |               | 2                  | EKS NodeGroup                   | 0.5 |
|          |                  |               | 3                  | Change Node Name                | 1.0 |
|          |                  |               | 4                  | Cluster Local Domain            | 1.0 |
|          |                  |               | 5                  | Green ECR                       | 0.5 |
|          |                  |               | 6                  | Red ECR                         | 0.5 |
|          |                  |               | 7                  | Green container image tag       | 0.5 |
|          |                  |               | 8                  | Red container image tag         | 0.5 |
|          |                  |               | 9                  | Green Container Name            | 1.0 |
|          |                  |               | 10                 | Red Container Name              | 1.0 |
|          |                  |               | 11                 | Deployment high availability    | 1.5 |
|          |                  |               | 12                 | Application LoadBalancer        | 1.0 |
|          |                  |               | 13                 | Hub LoadBalancer                | 1.5 |
|          | 3                | Data Services | 1                  | RDS Create                      | 0.5 |
|          |                  |               | 2                  | Multi-AZ DB instance            | 0.5 |
|          |                  |               | 3                  | RDS Table                       | 1.0 |
|          |                  |               | 4                  | Data Insert test                | 1.5 |
|          |                  |               | 5                  | Bucket Create                   | 0.5 |
|          |                  |               | 6                  | Bucket File Check               | 1.0 |
|          | 4                | CI/CD         | 1                  | Green CodeBuild                 | 0.5 |
| 2        |                  |               | Red CodeBuild      | 0.5                             |     |
| 3        |                  |               | Green CodePipeline | 0.5                             |     |
| 4        |                  |               | Red CodePipeline   | 0.5                             |     |
| 5        |                  |               | Green Image Build  | 1.0                             |     |
| 6        |                  |               | Red Image Build    | 1.0                             |     |
| 7        |                  |               | Green Approval     | 1.0                             |     |
| 8        |                  |               | Red Approval       | 1.0                             |     |
| 9        |                  |               | Green Deploy       | 1.5                             |     |
| 10       |                  |               | Red Deploy         | 1.5                             |     |
| 합계       |                  |               |                    |                                 | 30  |

| 순번  | 채점 항목  |
|-----|--|
| 1-1 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “ 10.0.0.0/16”이 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-vpcs --filter Name=tag:Name,Values=wsc2025-hub-vpc --query "Vpcs[].CidrBlock"</pre>   |
| 1-2 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “ us-east-1a ” 와 “ 10.0.0.0/24 ” , “ us-east-1b ” 와 “ 10.0.1.0/24 ” 가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-subnets \   --filters "Name=tag:Name,Values=wsc2025-hub-pub-sn-a,wsc2025-hub-pub-sn-b" \   --query "Subnets[].AvailabilityZone" \   --output text</pre>   |
| 1-3 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 "wsc2025-hub-pub-rt" 로 시작하는 문구가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-route-tables --filters "Name=tag:Name,Values=wsc2025-hub-pub-rt" \   --query "RouteTables[*].{Name:Tags[?Key=='Name'] [0].Value}" --output text</pre>   |
| 1-4 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 "igw-" 로 시작하는 문구가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsc2025-hub-pub-rt \   --query "RouteTables[].Routes[].GatewayId"</pre>  |
| 1-5 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “ 172.16.0.0/16 ” 가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-vpcs --filter Name=tag:Name,Values=wsc2025-app-vpc --query "Vpcs[].CidrBlock"</pre>   |
| 1-6 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “ us-east-1a ” 에는 “ 172.16.0.0/24, 172.16.2.0/24, 172.16.4.0/24 ” 가 출력이 되고, “ us-east-1b ” 에는 172.16.1.0/24, 172.16.3.0/24, 172.16.5.0/24 ” 가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-subnets \   --filters   "Name=tag:Name,Values=wsc2025-app-pub-sn-a,wsc2025-app-pub-sn-b,wsc2025-app-priv-sn-a,wsc2025-app-priv-sn-b,wsc2025-app-db-sn-a,wsc2025-app-db-sn-b" \   --query "Subnets[*].[AvailabilityZone, CidrBlock]" \   --output text</pre> |

| 순번   | 채점 항목  |
|------|--|
| 1-7  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-app-pub-rt”, “wsc2025-app-priv-rt-a”, “wsc2025-app-priv-rt-b”, “wsc2025-app-db-rt” 가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-route-tables --filters "Name=tag:Name,Values=wsc2025-app-pub-rt,wsc2025-app-priv-rt-a,wsc2025-app-priv-rt-b,wsc2025-app-db-rt" \ --query "RouteTables[*].{Name:Tags[?Key=='Name']}[0].Value}" --output text</pre> |
| 1-8  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 "nat-" 로 시작하는 문구가 두 번 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-route-tables \ --filters "Name=tag:Name,Values=wsc2025-app-priv-rt-a,wsc2025-app-priv-rt-b" \ --query "RouteTables[*].Routes[*].[NatGatewayId]" \ --output text   grep ^nat-</pre>  |
| 1-9  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-tgw” 가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-transit-gateways \ --filters "Name=tag:Name,Values=wsc2025-tgw" \ --query "TransitGateways[*].Tags[?Key=='Name'].Value[]" \ --output text</pre>   |
| 1-10 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-app-tgat” 와 wsc2025-hub-tgat ” 가 출력되는지 확인합니다.</p> <pre>\$ aws ec2 describe-transit-gateway-attachments \ --filters "Name=tag:Name,Values=wsc2025-app-tgat,wsc2025-hub-tgat" \ --query "TransitGatewayAttachments[*].Tags[?Key=='Name'].Value[]" \ --output text</pre>  |
| 2-1  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-eks-cluster” 와 “1.32” 가 출력되는지 확인합니다.</p> <pre>\$ aws eks describe-cluster --name wsc2025-eks-cluster --query 'cluster.[name, version]' \ --output text</pre>   |

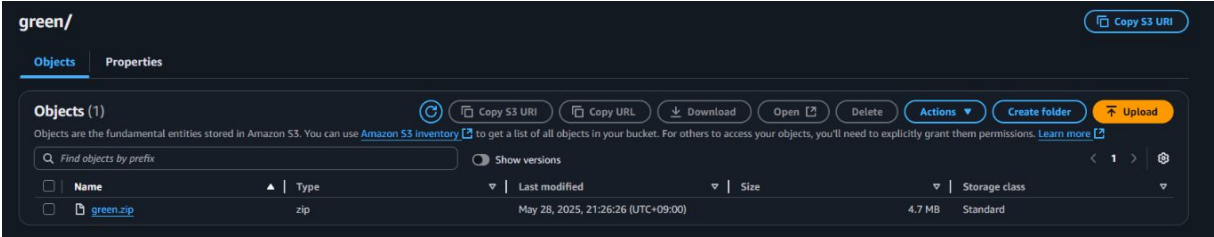
| 순번  | 채점 항목  |
|-----|--|
| 2-2 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-app-ng” 와 “t3.medium” 가 출력되는지 확인합니다.</p> <pre>\$ aws eks describe-nodegroup --cluster-name wsc2025-eks-cluster --nodegroup-name wsc2025-app-ng \ --query 'nodegroup.[nodegroupName, instanceTypes]' --output text</pre> <p>3) 아래 명령어 입력 후 “wsc2025-addon-ng” 와 “t3.medium” 가 출력되는지 확인합니다.</p> <pre>\$ aws eks describe-nodegroup --cluster-name wsc2025-eks-cluster --nodegroup-name wsc2025-addon-ng \ --query 'nodegroup.[nodegroupName, instanceTypes]' --output text</pre> |
| 2-3 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “&lt;instance-id&gt;.ec2.internal” 형태인지 확인합니다.</p> <pre>\$ kubectl get nodes -o custom-columns=NAME:.metadata.name --no-headers</pre>   |
| 2-4 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 두 개의 Address가 출력되는지 확인 합니다.</p> <pre>\$ kubectl run tmp-curl --rm -i --tty \ --image=debian --restart=Never \ -- bash -c "apt update -y &amp;&amp; apt install -y dnsutils &amp;&amp; nslookup wsc2025-green-svc.wsc2025.svc.wsc2025.local"</pre>   |
| 2-5 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “green” 가 출력되는지 확인 합니다.</p> <pre>\$ aws ecr describe-repositories \ --query "repositories[?repositoryName=='green'].repositoryName" \ --output text</pre>   |
| 2-6 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “red” 가 출력되는지 확인 합니다.</p> <pre>\$ aws ecr describe-repositories \ --query "repositories[?repositoryName=='red'].repositoryName" \ --output text</pre>   |
| 2-7 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “vx.x.x” 형식만 출력되는지 확인합니다.</p> <pre>\$ aws ecr list-images \ --repository-name green \ --query "imageIds[].imageTag" \ --output text   tr '\t' '\n'   grep -E '^v[0-9]+\.[0-9]+\.[0-9]+\$'</pre>   |

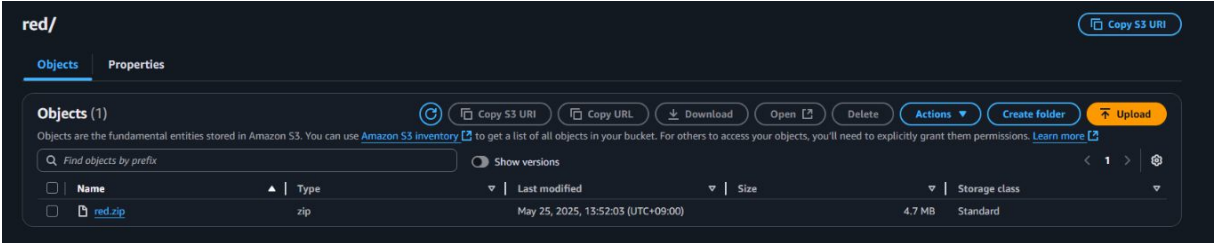
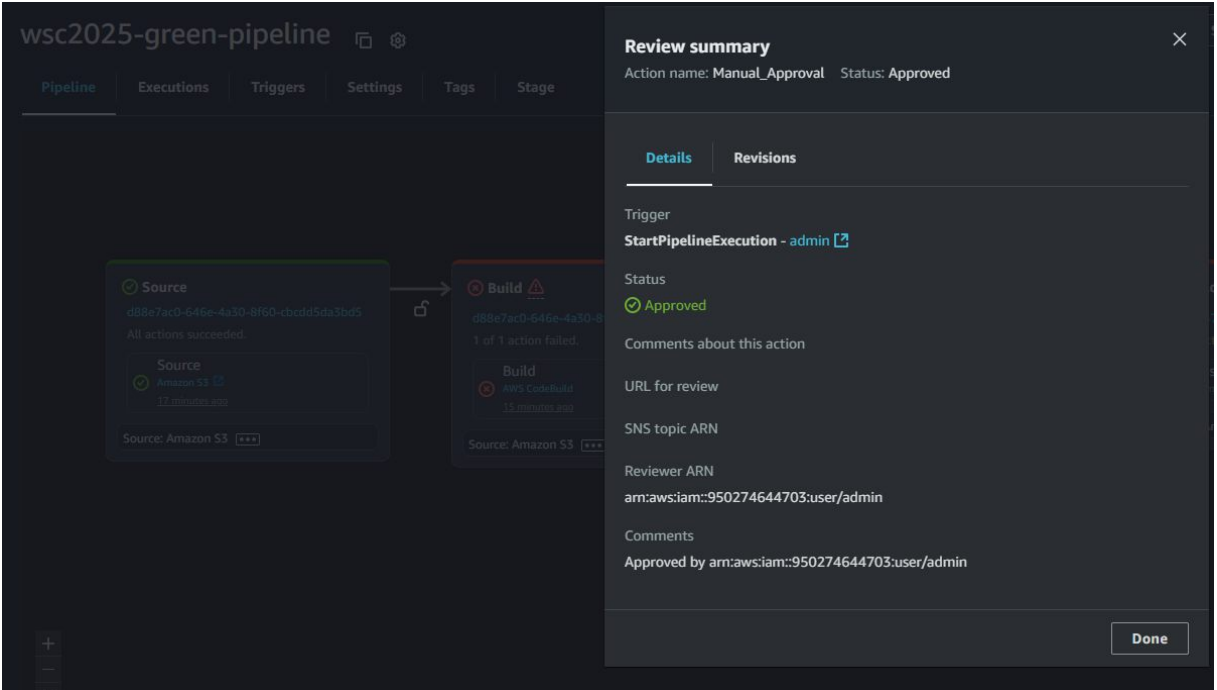
| 순번   | 채점 항목  |
|------|--|
| 2-8  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “vx.x.x” 형식만 출력되는지 확인합니다.</p> <pre>\$ aws ecr list-images \   --repository-name red \   --query "imageIds[].imageTag" \   --output text   tr '\t' '\n'   grep -E '^v[0-9]+\.[0-9]+\.[0-9]+\$'</pre>   |
| 2-9  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-green-app” 가 출력되는지 확인합니다.</p> <pre>\$ kubectl get deployment wsc2025-green-deploy -n wsc2025 -o jsonpath="{.spec.template.spec.containers[*].name}"   grep -w wsc2025-green-app</pre>  |
| 2-10 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-red-app” 가 출력되는지 확인합니다.</p> <pre>\$ kubectl get deployment wsc2025-red-deploy -n wsc2025 -o jsonpath="{.spec.template.spec.containers[*].name}"   grep -w wsc2025-red-app</pre>  |
| 2-11 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 각각 다른 가용영역이 출력 되는지 확인합니다.</p> <pre>\$ kubectl get pods -l app=wsc2025-green-deploy -n wsc2025 -o wide \     awk 'NR&gt;1 {print \$7}' \     xargs -l{} kubectl get node {} -o jsonpath="{.metadata.labels['topology\kubernetes.io/zone']}" \     sort   uniq</pre> <p>3) 아래 명령어 입력 후 각각 다른 가용영역이 출력 되는지 확인합니다.</p> <pre>\$ kubectl get pods -l app=wsc2025-red-deploy -n wsc2025 -o wide \     awk 'NR&gt;1 {print \$7}' \     xargs -l{} kubectl get node {} -o jsonpath="{.metadata.labels['topology\kubernetes.io/zone']}" \     sort   uniq</pre> |
| 2-12 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-app-alb” 와 “internal” 가 출력 되는지 확인합니다.</p> <pre>\$ aws elbv2 describe-load-balancers \   --query "LoadBalancers[?LoadBalancerName=='wsc2025-app-alb'].[LoadBalancerName, Scheme]" \   --output text</pre>   |

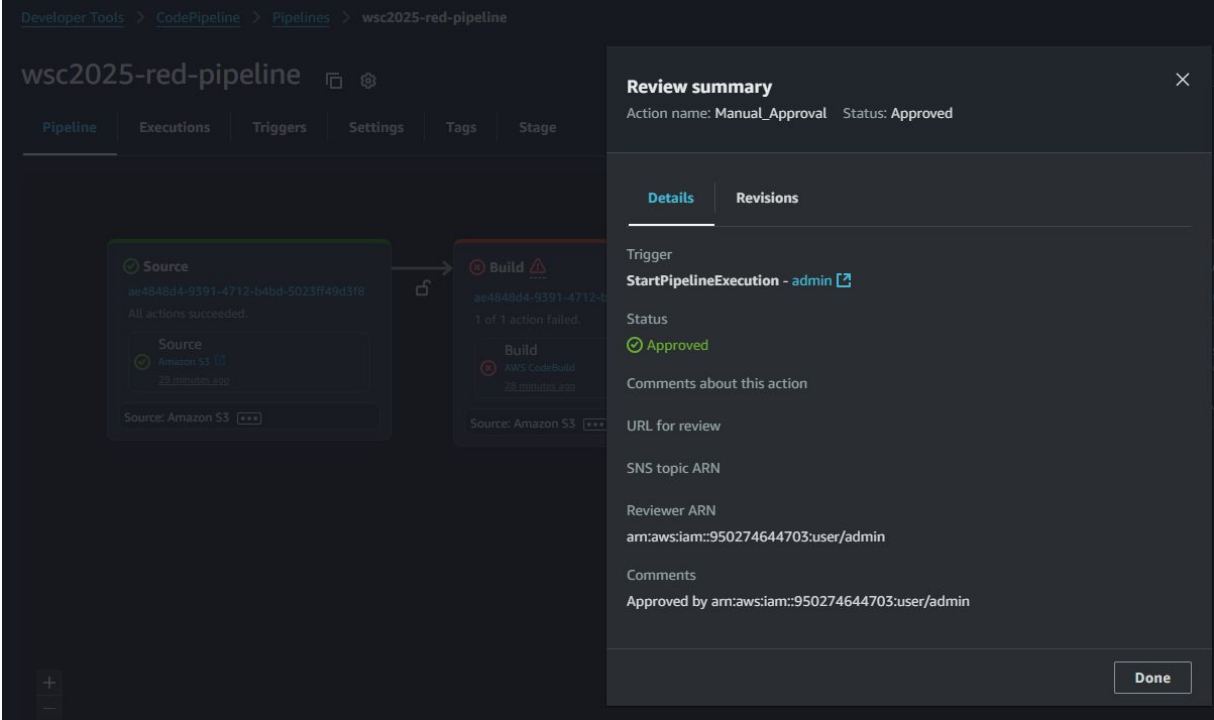
| 순번   | 채점 항목   |
|------|---|
| 2-13 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-external-nlb”와 “internet-facing”가 출력 되는지 확인합니다.</p> <pre>\$ aws elbv2 describe-load-balancers \   --query "LoadBalancers[?LoadBalancerName=='wsc2025-external-nlb'].[LoadBalancerName, Scheme]" \   --output text</pre>   |
| 3-1  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-db-instance”와 “mysql”가 출력 되는지 확인합니다.</p> <pre>\$ aws rds describe-db-instances \   --query "DBInstances[?DBInstanceIdentifier=='wsc2025-db-instance'].[DBInstanceIdentifier, Engine]" \   --output text</pre>   |
| 3-2  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “True”가 출력 되는지 확인합니다.</p> <pre>\$ aws rds describe-db-instances \   --query "DBInstances[?DBInstanceIdentifier=='wsc2025-db-instance'].MultiAZ" \   --output text</pre>  |
| 3-3  | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “green”과 “red”가 출력 되는지 확인합니다.</p> <pre>\$ export DB_ENDPOINT=\$(aws rds describe-db-instances \   --db-instance-identifier wsc2025-db-instance \   --query "DBInstances[0].Endpoint.Address" \   --output text) \$ mysql -h \$DB_ENDPOINT -u admin -pSkill53## -e "USE day1; SHOW TABLES;"</pre> |



| 순번  | 채점 항목   |
|-----|---|
| 3-4 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어를 입력 후 응답의 status가 inserted인지 확인합니다.</p> <pre>\$ export LB_ENDPOINT=\$(aws elbv2 describe-load-balancers \   --names wsc2025-external-nlb \   --query "LoadBalancers[0].DNSName" \   --output text)</pre> <pre>\$ curl --silent -X POST -H "Content-Type: application/json" \   -d '{"x": "abcd", "y": 21}' http://\${LB_ENDPOINT}/green</pre> <p>3) 위에서 출력된 id 값을 복사 후 아래 명령어 id 값에 넣었을 때<br/>     “ {"version": "1.0.0", "x": "abcd", "y": 21} ” 가 출력 되는지 확인합니다.</p> <pre>\$ curl --silent -X GET http://\${LB_ENDPOINT}/green?id=&lt;위에서 출력된 ID&gt;</pre> <p>4) 아래 명령어를 입력 후 응답의 status가 inserted인지 확인합니다.</p> <pre>\$ curl --silent -X POST -H "Content-Type: application/json" \   -d '{"name": "kim"}' http://\${LB_ENDPOINT}/red</pre> <p>5) 위에서 출력된 id 값을 복사 후 아래 명령어 id 값에 넣었을 때<br/>     “ {"name": "kim", "version": "1.0.0"} ” 가 출력 되는지 확인합니다.</p> <pre>\$ curl --silent -X GET http://\${LB_ENDPOINT}/red?id=&lt;위에서 출력된 ID&gt;</pre> |
| 3-5 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어를 입력 후 wsc2025-app-〈임의의 4자리 영문〉인지 확인합니다.</p> <pre>\$ aws s3api list-buckets \   --query "Buckets[].Name" \   --output text   tr '\t' '\n'   grep -E '^wsc2025-app-[a-zA-Z0-9]{4}\$'</pre>  |
| 3-6 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어를 입력 후<br/>     “ source/green/green.zip ” , “ source/red/red.zip ” 인지 출력되는지<br/>     확인합니다.</p> <pre>\$ export BUCKET_NAME=\$(aws s3api list-buckets --query "Buckets[].Name" --output text   tr '\t' '\n'   grep -E '^wsc2025-app-[a-zA-Z0-9]{4}\$')</pre> <pre>\$ for prefix in green red; do   aws s3api list-objects-v2 \     --bucket \$BUCKET_NAME \     --prefix source/\$prefix/ \     --query "Contents[].Key" \     --output text   echo "" done</pre>  |

| 순번  | 채점 항목   |
|-----|---|
| 4-1 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-green-build”가 출력 되는지 확인합니다.</p> <pre>\$ aws codebuild list-projects \   --query "projects[?contains(@, 'wsc2025-green-build')]" \   --output text</pre>   |
| 4-2 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-red-build”가 출력 되는지 확인합니다.</p> <pre>\$ aws codebuild list-projects \   --query "projects[?contains(@, 'wsc2025-red-build')]" \   --output text</pre>   |
| 4-3 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-green-pipeline”가 출력 되는지 확인합니다.</p> <pre>\$ aws codepipeline list-pipelines \   --query "pipelines[?name=='wsc2025-green-pipeline'].name" \   --output text</pre>  |
| 4-4 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsc2025-red-pipeline”가 출력 되는지 확인합니다.</p> <pre>\$ aws codepipeline list-pipelines \   --query "pipelines[?name=='wsc2025-red-pipeline'].name" \   --output text</pre>  |
| 4-5 | <p>1) 웹 브라우저를 사용해 AWS 콘솔에 접속합니다.</p> <p>2) ECR 콘솔에 접속하여 green repository에 v1.0.1 image를 삭제합니다.</p> <p>3) S3 콘솔에 접속하여 wsc2025-app-〈임의의 4자리 영문〉 버킷을 선택한 후, /source/green에 각각 green.zip 파일을 업로드합니다. 각 ZIP 파일에는 green_1.0.1 버전의 애플리케이션 파일을 포함하여 압축합니다.</p>  <p>3) 최대 3분 뒤에 ECR 콘솔에 접속하여 green repository에 image tag가 v1.0.1로 올라왔는지 확인합니다.</p> |

| 순번  | 채점 항목   |
|-----|---|
| 4-6 | <p>1) 웹 브라우저를 사용해 AWS 콘솔에 접속합니다.</p> <p>2) ECR 콘솔에 접속하여 red repository에 v1.0.1 image를 삭제합니다.</p> <p>3) S3 콘솔에 접속하여 wsc2025-app-〈임의의 4자리 영문〉 버킷을 선택한 후, /source/red에 각각 red.zip 파일을 업로드합니다. 각 ZIP 파일에는 red_1.0.1 버전의 애플리케이션 파일을 포함하여 압축합니다.</p>  <p>3) 최대 3분 뒤에 ECR 콘솔에 접속하여 red repository에 image tag가 v1.0.1로 올라왔는지 확인합니다.</p> |
| 4-7 | <p>1) 웹 브라우저를 사용해 AWS 콘솔에 접속합니다.</p> <p>2) Pipeline 콘솔에 접속하여 wsc2025-green-pipeline에 Approval Stage 단계에서 승인합니다.</p>    |

| 순번  | 채점 항목  |
|-----|--|
| 4-8 | <p>1) 웹 브라우저를 사용해 AWS 콘솔에 접속합니다.</p> <p>2) Pipeline 콘솔에 접속하여 wsc2025-red-pipeline에 Approval Stage 단계에서 승인합니다.</p>   |
| 4-9 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 3분 뒤 아래 명령어를 입력 후 응답의 status가 inserted인지 확인합니다.</p> <pre>\$ export LB_ENDPOINT=\$(aws elbv2 describe-load-balancers \ --names wsc2025-external-nlb \ --query "LoadBalancers[0].DNSName" \ --output text)</pre> <pre>\$ curl --silent -X POST -H "Content-Type: application/json" \ -d '{"x": "lee", "y": 19}' http://{LB_ENDPOINT}/green</pre> <p>3) 위에서 출력된 id 값을 복사 후 아래 명령어 id 값에 넣었을 때<br/> “ {"version": "1.0.1", "x": "lee", "y": 19} ” 가 출력 되는지 확인합니다.</p> <pre>\$ curl --silent -X GET http://{LB_ENDPOINT}/green?id=&lt;위에서 출력된 ID&gt;</pre> |

| 순번   | 채점 항목   |
|------|---|
| 4-10 | <p>1) SSH를 통해 Bastion 서버에 접근합니다.</p> <p>2) 3분 뒤 아래 명령어를 입력 후 응답의 status가 inserted인지 확인합니다.</p> <pre>\$ export LB_ENDPOINT=\$(aws elbv2 describe-load-balancers \   --names wsc2025-external-nlb \   --query "LoadBalancers[0].DNSName" \   --output text)</pre> <pre>\$ curl --silent -X POST -H "Content-Type: application/json" \   -d '{"name": "lee"}' http://\${LB_ENDPOINT}/red</pre> <p>4) 위에서 출력된 id 값을 복사 후 아래 명령어 id 값에 넣었을 때<br/>     “{"name": "lee", "version": "1.0.1"}”가 출력 되는지 확인합니다.</p> <pre>\$ curl --silent -X GET http://\${LB_ENDPOINT}/red?id=&lt;위에서 출력된 ID&gt;</pre> |