

# 2024 경상북도 제59회 전국기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅
<p>※ 다음 사항을 유의하여 채점하십시오.</p> <ol style="list-style-type: none"> <li>1) AWS의 지역은 ap-northeast-2을 사용합니다.</li> <li>2) 웹페이지 접근은 크롬이나 파이어폭스를 이용합니다.</li> <li>3) 웹페이지에서 언어에 따라 문구가 다르게 보일 수 있습니다.</li> <li>4) shell에서의 명령어의 출력은 버전에 따라 조금 다를 수 있습니다.</li> <li>5) 문제지와 채점지에 있는 &lt; &gt; 는 변수입니다. 해당 부분을 변경해 입력합니다.</li> <li>6) 채점은 문항 순서대로 진행해야 합니다.</li> <li>7) 삭제된 채점자료는 되돌릴 수 없음으로 유의하여 진행하며, 이의신청까지 완료 이후 선수가 생성한 클라우드 리소스를 삭제합니다.</li> <li>8) 부분 점수가 있는 문항은 채점 항목에 부분 점수가 적혀져 있습니다.</li> <li>9) 부분 점수가 따로 없는 문항은 모두 맞아야 점수로 인정됩니다.</li> <li>10) 리소스의 정보를 읽어오는 채점항목은 기본적으로 스크립트 결과를 통해 채점을 진행하며, 만약 선수가 이의가 있다면 명령어를 직접 입력하여 확인해볼 수 있습니다.</li> <li>11) (예상 출력)은 바로 이전 (명령어 입력)의 예상 출력을 의미합니다.</li> <li>12) 채점 시에는 별도로 제공한 채점 스크립트(marking.sh)를 실행하여 채점할 수 있습니다.</li> </ol> <p>다만, 선수가 직접 입력을 원할 경우 채점기준표에 명시된 명령어 그대로 입력하여 채점할 수 있습니다.</p> <ol style="list-style-type: none"> <li>13) 배포된 채점 스크립트(marking.sh) 는 ec2-user에 최상위 경로에 위치 하도록 합니다.</li> <li>14) 모든 채점 사항은 gj2025-bastion에서 ssh 접속 후 진행합니다.</li> </ol>		

## 2. 채점기준표

1) 주요항목별 배점				직 종 명		클라우드컴퓨팅		
과제 번호	일련 번호	주요항목	배점	채점방법		채점시기		비고
				독립	합의	경기 진행중	경기 종료후	
제1과제	1	Network Configuration	1.5		○		○	
	2	Transit Gateway	1.5		○		○	
	3	Application	3.0		○		○	
	4	Security	2.75		○		○	
	5	RDBMS	2.0		○		○	
	6	Container Registry	0.25		○		○	
	7	Container	4.0		○		○	
	8	Secret management	2.0		○		○	
	9	Logging	3.0		○		○	
	10	Load Balancing	3.0		○		○	
	11	Continuous Integration	2.5		○		○	
	12	Continuous Delivery	4.5		○		○	
합 계			30					

## 2) 채점방법 및 기준

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
1과제	1	Network Configuration	1	VPC	0.5
			2	Subnet	0.5
			3	Routing Table	0.5
	2	Transit Gateway	1	Transit Gateway Configuration	1.5
	3	Application	1	Red API Test	1.5
			2	Green API Test	1.5
	4	Security	1	Network Firewall Subnet	0.25
			2	Network Firewall Configuration	0.5
			3	Network Firewall Policy	0.5
			4	Connection test - Deny	1.5
	5	RDBMS	1	RDS Configuration	1.0
			2	Proxy Configuration	1.0
	6	Container Registry	1	ECR Configuration	0.25
	7	Container	1	EKS Configuration	0.5
			2	NodeGroup Configuration	0.5
			3	Application Pods	1.5
			4	External Secrets	1.5

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
1과제	8	Secret managment	1	SecretsManager Configuration	0.75
			2	SecretsManager Security	1.25
	9	Logging	1	Red Application Logging Test	1.5
			2	Green Application Logging Test	1.5
	10	Load Balancing	1	External ELB Configuration	1.5
			2	Internal ELB Configuration	1.5
	11	Continuous Integration	1	CodeBuild Project Configuration	1.25
			2	CodePipeline Configuration	1.25
	12	Continuous Delivery	1	ArgoCD Account Configuration	0.5
			2	Blue/Green Deployment	1.0
			3	Red CI/CD Test	1.5
			4	Green CI/CD Test	1.5
	총점				30

### 3) 채점내용

순번	사전준비	
0	<p>1) gj2025-bastion 서버에 SSH를 통해 포트 2222로 접근합니다. (포트 2222 외의 모든 포트에 대한 접근은 제한되어 있습니다.)</p> <p>2) <code>rm -rf ~/.aws</code>를 진행합니다.</p> <p>3) <code>aws configure</code>를 입력하고 <code>default.region</code>을 <code>ap-northeast-2</code>으로 설정합니다.</p> <hr/> <p>위의 작업이 완료되면 “사전준비 완료! 채점 시작!” 이라는 문구가 출력됩니다.</p>	
순번	채점항목	
1-1	1-1-A (명령어 입력)	<pre>aws ec2 describe-vpcs --filter Name=tag:Name,Values=gj2025-hub-vpc --query "Vpcs[0].CidrBlock" \ ; aws ec2 describe-vpcs --filter Name=tag:Name,Values=gj2025-app-vpc --query "Vpcs[0].CidrBlock"</pre>
	1-1-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	<pre>"10.0.0.0/16" "192.168.0.0/16"</pre>

순번	채점항목	
1-2	1-2-A (명령어 입력)	<pre>aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-hub-public-subnet-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-hub-public-subnet-b --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-hub-private-subnet-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-hub-private-subnet-b --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-hub-firewall-subnet --query "Subnets[0].CidrBlock"</pre>
	1-2-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	<pre>"10.0.0.0/24" "10.0.1.0/24" "10.0.2.0/24" "10.0.3.0/24" "10.0.4.0/24"</pre>

순번	채점항목	
1-2	1-2-B (명령어 입력)	<pre>aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-app-private-subnet-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-app-private-subnet-b --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-app-data-subnet-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=gj2025-app-data-subnet-b --query "Subnets[0].CidrBlock"</pre>
	1-2-B (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	<pre>"192.168.0.0/24" "192.168.1.0/24" "192.168.2.0/24" "192.168.3.0/24"</pre>

순번	채점항목	
1-3	1-3-A (명령어 입력)	<pre> aws ec2 describe-route-tables --filters "Name=tag:Name,Values=gj2025-hub-public-rtb" --query "RouteTables[].Routes[?GatewayId != null &amp;&amp; starts_with(GatewayId, 'igw')].GatewayId" --output text \ ; aws ec2 describe-route-tables --filters "Name=tag:Name,Values=gj2025-hub-firewall-rtb" --query "RouteTables[].Routes[?NatGatewayId != null].NatGatewayId" --output text \ ; aws ec2 describe-route-tables --filters "Name=tag:Name,Values=gj2025-app-data-rtb-a" --query "RouteTables[].Associations[].SubnetId" --output text   xargs -I {} aws ec2 describe-subnets --subnet-ids {} --query "Subnets[].Tags[?Key=='Name'].Value" --output text \ ; aws ec2 describe-route-tables --filters "Name=tag:Name,Values=gj2025-app-data-rtb-b" --query "RouteTables[].Associations[].SubnetId" --output text   xargs -I {} aws ec2 describe-subnets --subnet-ids {} --query "Subnets[].Tags[?Key=='Name'].Value" --output text </pre>
	1-3-A (예상 출력) <b>순서 중요</b>	<p>“igw-” 로 시작하는 문구가 출력이 되는지 확인</p> <p>“nat-” 로 시작하는 문구가 출력이 되는지 확인</p> <p>gj2025-app-data-subnet-a &lt;- <u>정확히 일치</u></p> <p>gj2025-app-data-subnet-b &lt;- <u>정확히 일치</u></p>



순번	채점항목	
2-1	2-1-A (명령어 입력)	<pre> TGWS=\$(aws ec2 describe-transit-gateways --query "TransitGateways[*].{Name:Tags[?Key=='Name'].Value[0]}" --output json) TGW_NAMES=\$(echo \$TGWS   jq -r '.[].Name') for TGW_NAME in \$TGW_NAMES; do     echo "\$TGW_NAME"     TGW_ID=\$(aws ec2 describe-transit-gateways --filters "Name=tag:Name,Values=\$TGW_NAME" --query "TransitGateways[0].TransitGatewayId" --output text)     ATTACHMENTS=\$(aws ec2 describe-transit-gateway-attachments --filters "Name=transit-gateway-id,Values=\$TGW_ID" --query "TransitGatewayAttachments[*].{Name:Tags[?Key=='Name'].Value[0]}" --output json)     ATTACHMENT_NAMES=\$(echo \$ATTACHMENTS   jq -r '.[].Name')     for ATTACHMENT_NAME in \$ATTACHMENT_NAMES; do         echo "\$ATTACHMENT_NAME"     done done </pre>
	2-1-A (예상 출력)	<pre> gj2025-tgw &lt;- 정확히 일치 gj2025-app-tgw-attach &lt;- 정확히 일치, 순서 상관 없음 gj2025-hub-tgw-attach &lt;- 정확히 일치, 순서 상관 없음 </pre>
3-1	3-1-A (명령어 입력)	<pre> APP_EXTERNAL_NLB=\$(aws elbv2 describe-load-balancers \ --names gj2025-app-external-nlb \ --query "LoadBalancers[].DNSName" \ --output text) &amp;&amp; \ id_red=\$(curl -s -H "Content-Type: application/json" -d '{"name":"kim"}' http://\$APP_EXTERNAL_NLB/red   jq -r '.id') &amp;&amp; \ curl -s http://\$APP_EXTERNAL_NLB/red?id=\$id_red </pre>
	3-1-A (예상 출력) <u>정확히 일치</u>	<pre> {"name":"kim","version":"1.0.0"} </pre>

순번	채점항목	
3-2	3-2-A (명령어 입력)	<pre>APP_EXTERNAL_NLB=\$(aws elbv2 describe-load-balancers \ --names gj2025-app-external-nlb \ --query "LoadBalancers[].DNSName" \ --output text) &amp;&amp; \ id_green=\$(curl -s -H "Content-Type: application/json" -d '{"x":"abcd","y":21}' http://\$APP_EXTERNAL_NLB/green   jq -r '.id') &amp;&amp; \ curl -s http://\$APP_EXTERNAL_NLB/green?id=\$id_green</pre>
	3-2-A (예상 출력) <u>정확히 일치</u>	<pre>{"version":"1.0.0","x":"abcd","y":21}</pre>
4-1	4-1-A (명령어 입력)	<pre>firewall_info=\$(aws network-firewall describe-firewall --firewall-name gj2025-firewall) echo "\$firewall_info"   jq -r '.Firewall.SubnetMappings[]   "ID: \(.SubnetId)"'   while read -r subnet_info; do   subnet_id=\$(echo "\$subnet_info"   awk '{print \$2}')   subnet_name=\$(aws ec2 describe-subnets \ --subnet-ids "\$subnet_id" \ --query 'Subnets[*].Tags[?Key==`Name`].Value' \ --output text)   echo "\$subnet_name" done</pre>
	4-1-A (예상 출력) <u>정확히 일치</u>	<pre>gj2025-hub-firewall-subnet</pre>

순번	채점항목	
4-2	4-2-A (명령어 입력)	<pre>aws network-firewall describe-firewall --firewall-name gj2025-firewall --query "FirewallStatus.Status" --output text aws network-firewall describe-logging-configuration \ --firewall-name gj2025-firewall \ --query 'LoggingConfiguration.LogDestinationConfigs' \ --output json   jq -r '[]   "\(.LogType)\n\(.LogDestinationType)\n\(.LogDestination.logGroup)\n"</pre>
	4-2-A (예상 출력) <b>정확히 일치</b>	<pre>READY FLOW CloudWatchLogs /gj2025/firewall</pre>
4-3	4-3-A (명령어 입력)	<pre>aws network-firewall describe-firewall-policy --firewall-policy-name gj2025-firewall-policy --query 'FirewallPolicyResponse.{FirewallPolicyName: FirewallPolicyName}' --output text arn=\$(aws network-firewall list-rule-groups --query "RuleGroups[?contains(Name, 'gj2025-firewall-rule')].Arn" --output text) aws network-firewall describe-rule-group --rule-group-arn \$arn --query 'RuleGroup.RulesSource.RulesString' --output text   grep -q . &amp;&amp; echo "Suricata"</pre>
	4-3-A (예상 출력) <b>정확히 일치</b>	<pre>gj2025-firewall-policy Suricata</pre>

순번	채점항목	
4-4	4-4-A (명령어 입력)	<pre>kubectrl run firewall-test --image=radial/busyboxplus --restart=Never --command -- sh -c "while true; do sleep 3600; done" &gt; /dev/null 2&gt;&amp;1 sleep 3 aws ec2 describe-subnets --subnet-ids \$(aws ec2 describe-instances \ --instance-ids \$(kubectrl get node \$(kubectrl get pod firewall-test -n default -o jsonpath='{.spec.nodeName}')) \ -o jsonpath='{.spec.providerID}'   awk -F '/' '{print \$5}') \ --query 'Reservations[0].Instances[0].SubnetId' --output text) --query "Subnets[0].Tags[?Key=='Name'].Value" --output text kubectrl exec firewall-test -- curl -m 5 -sS https://ifconfig.io</pre>
	4-4-A (예상 출력)	<pre>gj2025-app-private-subnet- &lt;- 로 시작하는 문구가 출력되는지 확인 curl: (28) Operation timed out after 0 milliseconds with 0 out of 0 bytes received &lt;- 정확히 일치 command terminated with exit code 28 &lt;- 정확히 일치</pre>

순번	채점항목	
5-1	5-1-A (명령어 입력)	<pre>aws rds describe-db-instances \   --query "DBInstances[*].[DBInstanceIdentifier, DBInstanceClass, MasterUsername, Endpoint.Port, Engine, EngineVersion, join(',', EnabledCloudwatchLogsExports)]" \   --output text   tr '\t' '\n'  for id in \$(aws rds describe-db-instances --query "DBInstances[*].DBSubnetGroup.Subnets[*].SubnetIdentifier" --output text); do   aws ec2 describe-subnets --subnet-ids "\$id" --query "Subnets[0].Tags[?Key=='Name'].Value   [0]" --output text done</pre>
	5-1-A (예상 출력)	<pre>gj2025-db-instance &lt;- 정확히 일치 db.t3.medium &lt;- 정확히 일치 admin &lt;- 정확히 일치 3309 &lt;- 정확히 일치 mysql &lt;- 정확히 일치 8.0. &lt;- 로 시작하는 문구가 출력되는지 확인 audit,error,general &lt;- 정확히 일치 gj2025-app-data-subnet-a &lt;- 정확히 일치, 순서 상관 없음 gj2025-app-data-subnet-b &lt;- 정확히 일치, 순서 상관 없음</pre>
5-2	5-2-A (명령어 입력)	<pre>kubectl run proxy-test --image=mysql:8 --restart=Never -- sleep 60 &amp;&gt;/dev/null sleep 30 ENDPOINT=\$(aws rds describe-db-proxies --db-proxy-name gj2025-rds-proxy --query 'DBProxies[0].Endpoint' --output text) SUBNET_ID=\$(aws ec2 describe-instances --instance-ids \$(kubectl get node \$(kubectl get pod proxy-test -o jsonpath='{.spec.nodeName}') -o jsonpath='{.spec.providerID}'   cut -d '/' -f5) --query 'Reservations[0].Instances[0].SubnetId' --output text) aws ec2 describe-subnets --subnet-ids \$SUBNET_ID --query 'Subnets[0].Tags[?Key=='Name'].Value' --output text kubectl exec proxy-test -- mysql -h \$ENDPOINT -u admin -pSkills53#\\${%} -e 'SELECT 1' &amp;&gt;/dev/null &amp;&amp; echo True    echo False</pre>
	5-2-A (예상 출력)	<pre>gj2025-app-private-subnet- &lt;- 로 시작하는 문구가 출력되는지 확인 True &lt;- 정확히 일치</pre>

순번	채점항목	
6-1	6-1-A (명령어 입력)	aws ecr describe-repositories \ --output json \   jq -r '.repositories[]   "\"(.repositoryName)\n\"(.encryptionConfiguration.encryptionType)\"'
	6-1-A (예상 출력)	green <- 정확히 일치, 순서 상관 없음 KMS <- 정확히 일치 red <- 정확히 일치, 순서 상관 없음 KMS <- 정확히 일치
7-1	7-1-A (명령어 입력)	aws eks describe-cluster \ --name gj2025-eks-cluster \ --query 'cluster.[version, logging.clusterLogging[*].types]' \ --output text   awk 'NR==1{print \$0; next} {gsub(/\t/, "\n"); print}'
	7-1-A (예상 출력) <u>정확히 일치</u>	1.32 api audit authenticator controllerManager scheduler

순번	채점항목	
7-2	7-2-A (명령어 입력)	<pre>aws eks describe-nodegroup --cluster-name gj2025-eks-cluster --nodegroup-name gj2025-eks-addon-nodegroup --query 'nodegroup.{NodeGroupName:nodegroupName, Status:status, DesiredSize:scalingConfig.desiredSize, InstanceTypes:instanceTypes}' --output json \ ; aws eks describe-nodegroup --cluster-name gj2025-eks-cluster --nodegroup-name gj2025-eks-app-nodegroup --query 'nodegroup.{NodeGroupName:nodegroupName, Status:status, DesiredSize:scalingConfig.desiredSize, InstanceTypes:instanceTypes}' --output json \ ; aws ec2 describe-instances --instance-ids \$(aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names \$(aws eks describe-nodegroup --cluster-name gj2025-eks-cluster --nodegroup-name gj2025-eks-addon-nodegroup --query 'nodegroup.resources.autoScalingGroups[0].name' --output text) --query 'AutoScalingGroups[0].Instances[].InstanceId' --output text) --query 'Reservations[].Instances[].Tags[?Key==`Name`].Value   [])' --output text \ ; aws ec2 describe-instances --instance-ids \$(aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names \$(aws eks describe-nodegroup --cluster-name gj2025-eks-cluster --nodegroup-name gj2025-eks-app-nodegroup --query 'nodegroup.resources.autoScalingGroups[0].name' --output text) --query 'AutoScalingGroups[0].Instances[].InstanceId' --output text) --query 'Reservations[].Instances[].Tags[?Key==`Name`].Value   [])' --output text</pre>
	7-2-A (예상 출력) <u>정확히 일치</u>	<pre>{  "NodeGroupName": "gj2025-eks-addon-nodegroup",   "Status": "ACTIVE",   "DesiredSize": 2,   "InstanceTypes": [     "t3.medium"   ], } {  "NodeGroupName": "gj2025-eks-app-nodegroup",   "Status": "ACTIVE",   "DesiredSize": 2,   "InstanceTypes": [     "t3.medium"   ], }  gj2025-eks-addon-node  gj2025-eks-addon-node gj2025-eks-app-node  gj2025-eks-app-node</pre>

순번	채점항목	
7-3	7-3-A (명령어 입력)	<pre>kubectl get rollout red-rollout \ -n skills \   awk 'NR==2 {print \$1; if (\$2 == \$3 &amp;&amp; \$2 &gt; 0) print "True"; else print "False"}'</pre> <pre>kubectl get rollout green-rollout \ -n skills \   awk 'NR==2 {print \$1; if (\$2 == \$3 &amp;&amp; \$2 &gt; 0) print "True"; else print "False"}'</pre>
	7-3-A (예상 출력) <b>정확히 일치</b>	<pre>red-rollout True green-rollout True</pre>
7-4	7-4-A (명령어 입력)	<pre>kubectl get externalsecret db-secret -n skills   awk 'NR==2 {print \$(NF-1) "\n" \$NF}'</pre>
	7-4-A (예상 출력) <b>정확히 일치</b>	<pre>SecretSynced True</pre>
8-1	8-1-A (명령어 입력)	<pre>aws secretsmanager describe-secret --secret-id gj2025-eks-cluster-catalog-secret --region ap-northeast-2 --query "Name" --output text aws secretsmanager describe-secret --secret-id gj2025-github-token --region ap-northeast-2 --query "Name" --output text</pre>
	8-1-A (예상 출력) <b>정확히 일치</b>	<pre>gj2025-eks-cluster-catalog-secret gj2025-github-token</pre>



순번	채점항목	
8-2	8-2-A (명령어 입력)	<pre>for arn in \$(aws iam list-attached-role-policies --role-name \$(kubectl get sa \$(kubectl get secretstore \$(kubectl get externalsecret db-secret -n skills -o jsonpath="{.spec.secretStoreRef.name}") -n skills -o jsonpath="{.spec.provider.aws.auth.jwt.serviceAccountRef.name}") -n skills -o jsonpath="{.metadata.annotations.eks\amazonaws\.com/role-arn}"   awk -F '/' '{print \$2}') --query 'AttachedPolicies[*].PolicyArn' --output text); do aws iam get-policy-version --policy-arn \$arn --version-id \$(aws iam get-policy --policy-arn \$arn --query 'Policy.DefaultVersionId' --output text) --query 'PolicyVersion.Document.Statement[*].Resource' --output text; done   tr '\t' '\n'</pre>
	8-2-A (예상 출력) <u>개수 상관없음</u>	<pre>arn:aws:secretsmanager:ap-northeast-2:123423453456:secret:gj2025-eks-cluster-c atalog-secret-31Q2ZH arn:aws:secretsmanager:ap-northeast-2:123423453456:secret:gj2025-eks-cluster-c atalog-secret-31Q2ZH</pre> <p>* 밑줄친 부분은 다를수도 있음</p>

순번	채점항목	
9-1	9-1-A (명령어 입력)	<pre> kubectrl get daemonset red-fluent-bit -n amazon-cloudwatch -o yaml &gt; red-fluent-bit.yaml kubectrl delete daemonset red-fluent-bit -n amazon-cloudwatch &gt; /dev/null 2&gt;&amp;1 date aws logs delete-log-group --log-group-name /gj2025/app/red &gt; /dev/null 2&gt;&amp;1 kubectrl apply -f red-fluent-bit.yaml &gt; /dev/null 2&gt;&amp;1 sleep 3 EXTERNAL_NLB=\$(aws elbv2 describe-load-balancers \   --names gj2025-app-external-nlb \   --query "LoadBalancers[0].DNSName" \   --output text) curl -s -X POST -H "Content-Type: application/json" -d '{"name":"kim"}' http://\$EXTERNAL_NLB/red &gt; /dev/null 2&gt;&amp;1 sleep 3 aws logs get-log-events \   --log-group-name /gj2025/app/red \   --log-stream-name app-red-logs \   --limit 1 \   --query 'events[*].message' \   --output json   jq -r '.[0]   fromjson   .log' </pre>
	9-1-A (예상 출력)	<pre> Thu Jun 19 08:20:09 UTC 2025 2025-06-19T08:20:15.159611389Z stderr F 2025/06/19 08:20:15 [2025-06-19T08:20:15Z] POST /red from 192.168.0.99:19176 </pre> <p>* 밑줄친 부분의 경우 정확히 일치해야 한다.</p> <p>* date 명령에 출력된 시간 이후의 로그가 10초 내로 기록되었는지 확인</p>

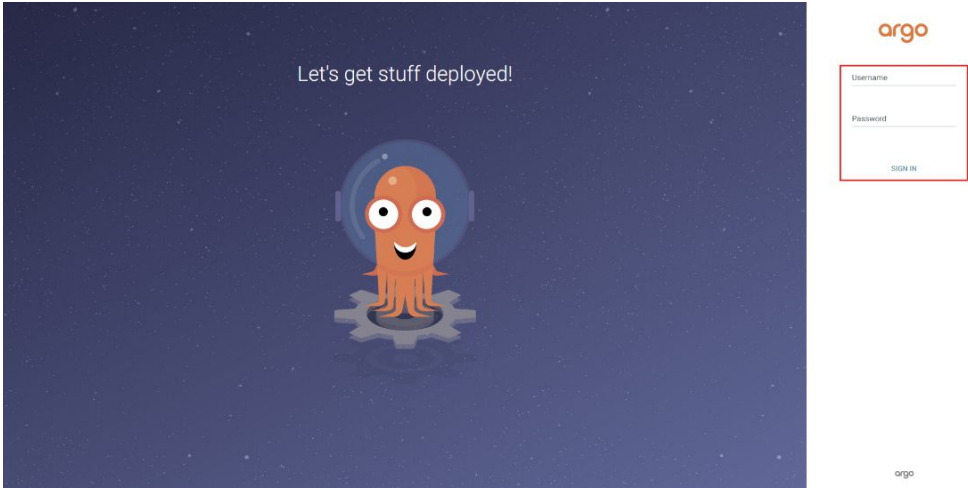
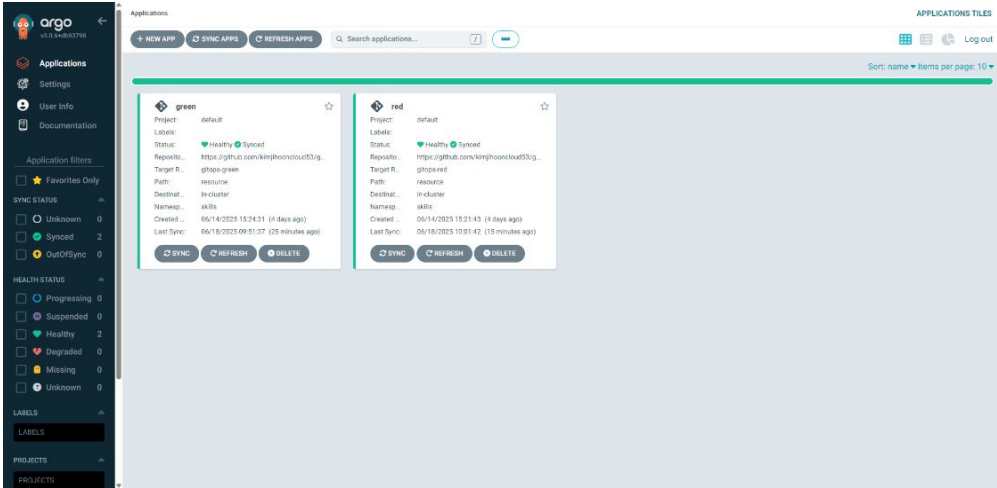
순번	채점항목	
9-2	9-2-A (명령어 입력)	<pre> kubectl get daemonset green-fluent-bit -n amazon-cloudwatch -o yaml &gt; green-fluent-bit.yaml kubectl delete daemonset green-fluent-bit -n amazon-cloudwatch &gt; /dev/null 2&gt;&amp;1 date aws logs delete-log-group --log-group-name /gj2025/app/green &gt; /dev/null 2&gt;&amp;1 kubectl apply -f green-fluent-bit.yaml &gt; /dev/null 2&gt;&amp;1 sleep 3 EXTERNAL_NLB=\$(aws elbv2 describe-load-balancers \   --names gj2025-app-external-nlb \   --query "LoadBalancers[0].DNSName" \   --output text) curl -s -X POST -H "Content-Type: application/json" -d '{"x":"abcd","y":21}' http://\$EXTERNAL_NLB/green &gt; /dev/null 2&gt;&amp;1 sleep 3 aws logs get-log-events \   --log-group-name /gj2025/app/green \   --log-stream-name app-green-logs \   --limit 1 \   --query 'events[*].message' \   --output json   jq -r '.[0]   fromjson   .log' </pre>
	9-2-A (예상 출력)	<pre> Thu Jun 19 08:23:27 UTC 2025 2025-06-19T08:23:32.684988646Z stderr F 2025/06/19 08:23:32 [2025-06-19T08:23:32Z] POST /green from 192.168.1.121:25986 </pre> <p>* 밑줄친 부분의 경우 정확히 일치해야 한다.</p> <p>* date 명령에 출력된 시간 이후의 로그가 10초 내로 기록되었는지 확인</p>

순번	채점항목	
10-1	10-1-A (명령어 입력)	<pre>aws elbv2 describe-load-balancers --names gj2025-app-external-nlb \ --query 'LoadBalancers[0].Scheme' --output text &amp;&amp; \ aws ec2 describe-tags \ --filters "Name=resource-id,Values=\$(aws elbv2 describe-load-balancers --names gj2025-app-external-nlb --query 'LoadBalancers[0].VpcId' --output text)" \ --query "Tags[?Key=='Name'].Value   [0]" --output text aws elbv2 describe-load-balancers --names gj2025-argo-external-nlb \ --query 'LoadBalancers[0].Scheme' --output text &amp;&amp; \ aws ec2 describe-tags \ --filters "Name=resource-id,Values=\$(aws elbv2 describe-load-balancers --names gj2025-argo-external-nlb --query 'LoadBalancers[0].VpcId' --output text)" \ --query "Tags[?Key=='Name'].Value   [0]" --output text</pre>
	10-1-A (예상 출력) <b><u>정확히 일치</u></b>	<pre>internet-facing gj2025-hub-vpc internet-facing gj2025-hub-vpc</pre>

순번	채점항목	
10-2	10-2-A (명령어 입력)	<pre>aws elbv2 describe-load-balancers --names gj2025-app-internal-nlb \ --query 'LoadBalancers[0].Scheme' --output text &amp;&amp; \ aws ec2 describe-tags \ --filters "Name=resource-id,Values=\$(aws elbv2 describe-load-balancers --names gj2025-app-internal-nlb --query 'LoadBalancers[0].VpcId' --output text)" \ --query "Tags[?Key=='Name'].Value   [0]" --output text aws elbv2 describe-load-balancers --names gj2025-argo-internal-nlb \ --query 'LoadBalancers[0].Scheme' --output text &amp;&amp; \ aws ec2 describe-tags \ --filters "Name=resource-id,Values=\$(aws elbv2 describe-load-balancers --names gj2025-argo-internal-nlb --query 'LoadBalancers[0].VpcId' --output text)" \ --query "Tags[?Key=='Name'].Value   [0]" --output text aws elbv2 describe-load-balancers --names gj2025-app-alb \ --query 'LoadBalancers[0].Scheme' --output text &amp;&amp; \ aws ec2 describe-tags \ --filters "Name=resource-id,Values=\$(aws elbv2 describe-load-balancers --names gj2025-app-alb --query 'LoadBalancers[0].VpcId' --output text)" \ --query "Tags[?Key=='Name'].Value   [0]" --output text</pre>
	10-2-A (예상 출력) <b><u>정확히 일치</u></b>	<pre>internal gj2025-app-vpc internal gj2025-app-vpc internal gj2025-app-vpc</pre>

순번	채점항목	
11-1	11-1-A (명령어 입력)	<pre>aws codebuild batch-get-projects --names gj2025-app-red-build --query "projects[0].[ \     source.type, \     source.location, \     source.auth.type, \     environment.environmentVariables[].type, \     logsConfig.cloudWatchLogs.groupName]" --output text   xargs -n1 aws codebuild batch-get-projects --names gj2025-app-green-build --query "projects[0].[ \     source.type, \     source.location, \     source.auth.type, \     environment.environmentVariables[].type, \     logsConfig.cloudWatchLogs.groupName]" --output text   xargs -n1</pre>
	11-1-A (예상 출력)	<pre>GITHUB https://github.com/cloud53/gj2025-repository.git SECRETS_MANAGER /gj2025/build/red SECRETS_MANAGER GITHUB https://github.com/cloud53/gj2025-repository.git SECRETS_MANAGER /gj2025/build/green SECRETS_MANAGER</pre> <p>* 밑줄친 부분은 다를수도 있음</p>

순번	채점항목	
11-2	11-2-A (명령어 입력)	<pre>aws codepipeline get-pipeline \ --name gj2025-app-red-pipeline \ --query 'pipeline.[stages[?name==`Source`].actions[0].[actionTypeId.provider, configuration.[OAuthToken != null, Repo]], stages[?name==`Build`].actions[0].configuration.ProjectName]' \ --output text   tr '\t' '\n' ; aws codepipeline get-pipeline \ --name gj2025-app-green-pipeline \ --query 'pipeline.[stages[?name==`Source`].actions[0].[actionTypeId.provider, configuration.[OAuthToken != null, Repo]], stages[?name==`Build`].actions[0].configuration.ProjectName]' \ --output text   tr '\t' '\n'</pre>
	11-2-A (예상 출력) <u>정확히 일치</u>	<pre>GitHub True gj2025-repository gj2025-app-red-build GitHub True gj2025-repository gj2025-app-green-build</pre>

순번	채점항목	
	<p>12-1-A (명령어 입력 후 작업 수행)</p>	<pre>aws elbv2 describe-load-balancers \ --names gj2025-argo-external-nlb \ --query "LoadBalancers[0].DNSName" \ --output text</pre> <p>* 위 명령어를 통해 응답받은 주소로 웹 브라우저에서 접근</p> <p>Username: admin Password: Skills53</p> <p>* 위 Username과 Password로 로그인이 가능해야 합니다.</p>
<p>12-1</p>	<p>12-1-A</p>	<div data-bbox="453 828 1426 1314">  </div> <p>* 위 화면에서 로그인을 진행해야 합니다.</p> <div data-bbox="453 1413 1455 1899">  </div> <p>* 로그인 완료 후 위와 같이 대시보드에 접속한 화면이 확인되어야 합니다.</p>



순번	채점항목	
12-2	12-2-A (명령어 실행)	<pre>kubectl argo rollouts get rollout red-rollout -n skills   egrep "Strategy" \ ; kubectl argo rollouts get rollout red-rollout -n skills   egrep "stable"   grep "Healthy"   awk {'print \$6,\$8'} kubectl argo rollouts get rollout green-rollout -n skills   egrep "Strategy" \ ; kubectl argo rollouts get rollout green-rollout -n skills   egrep "stable"   grep "Healthy"   awk {'print \$6,\$8'}</pre>
	12-2-A (예상 출력) <u>정확히 일치</u>	<pre>Strategy:      BlueGreen Healthy stable,active Strategy:      BlueGreen Healthy stable,active</pre>
12-3	12-3-A (명령어 실행)	<pre>cd /home/ec2-user/gj2025-repository git checkout app-red  * /home/ec2-user/gj2025-repository 경로에 red_1.0.1 바이너리를 불러옵니다. * /home/ec2-user/gj2025-repository 경로에 red_1.0.1 바이너리를 제외한 다른 애플리케이션 바이너리가 존재하면 안됩니다.  sudo mv red_1.0.1 red ls git add red &gt; /dev/null 2&gt;&amp;1 git commit -m "red app cicd test" &gt; /dev/null 2&gt;&amp;1 git push origin app-red &gt; /dev/null 2&gt;&amp;1 sleep 180 EXTERNAL_NLB=\$(aws elbv2 describe-load-balancers \ --names gj2025-app-external-nlb \ --query "LoadBalancers[0].DNSName" \ --output text) id=\$(curl -s -X POST -H "Content-Type: application/json" -d '{"name":"kim"}' http://\$EXTERNAL_NLB/red   jq -r '.id') curl -X GET "http://\$EXTERNAL_NLB/red?id=\$id"</pre>
	12-3-A (예상 출력)	<pre>Dockerfile  buildspec.yaml  red &lt;- <u>정확히 일치, 순서 상관 없음</u> {"name":"kim","version":"1.0.1"} &lt;- <u>정확히 일치</u></pre>

순번	채점항목	
12-4	12-4-A (명령어 실행)	<pre>cd /home/ec2-user/gj2025-repository git checkout app-green  * /home/ec2-user/gj2025-repository 경로에 green_1.0.1 바이너리를 불러옵니다. * /home/ec2-user/gj2025-repository 경로에 green_1.0.1 바이너리를 제외한 다른 red 애플리케이션 바이너리가 존재하면 안됩니다.  sudo mv green_1.0.1 green ls git add green &gt; /dev/null 2&gt;&amp;1 git commit -m "green app cicd test" &gt; /dev/null 2&gt;&amp;1 git push origin app-green &gt; /dev/null 2&gt;&amp;1 sleep 180 EXTERNAL_NLB=\$(aws elbv2 describe-load-balancers \   --names gj2025-app-external-nlb \   --query "LoadBalancers[0].DNSName" \   --output text) id=\$(curl -s -X POST -H "Content-Type: application/json" -d '{"x":"abcd","y":21}' http://\$EXTERNAL_NLB/green   jq -r '.id') curl -X GET "http://\$EXTERNAL_NLB/green?id=\$id"</pre>
	12-4-A (예상 출력)	<pre>Dockerfile buildspec.yaml green &lt;- 정확히 일치, 순서 상관 없음 {"version":"1.0.1","x":"abcd","y":21} &lt;- 정확히 일치</pre>