

# 2024 경상북도 제59회 전국기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅
<p>※ 다음 사항을 유의하여 채점하시오.</p> <ol style="list-style-type: none"> <li>1) AWS의 지역은 us-east-1을 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(wsc.sh)를 실행하여 채점할 수 있습니다. 다만, 선수가 직접 입력을 원할 경우 채점기준표에 명시된 명령어 그대로 입력하여 채점할 수 있습니다. 채점 스크립트는 root 경로에 지정하도록 합니다.</li> <li>13) 배포된 채점 스크립트(wsc.sh) 는 ec2-user에 최상위 경로에 위치 하도록 합니다.</li> <li>14) 모든 채점 사항은 wsc2024-bastion-ec2에서 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	4.0		○		○	
	2	Transit Between VPC	0.5		○		○	
	3	Bastion Server	1.0		○		○	
	4	Application Access Control	4.5		○		○	
	5	RDBMS	2.0		○		○	
	6	NoSQL	0.5		○		○	
	7	Container Registry	0.5		○		○	
	8	Container Orchestration	2.5		○		○	
	9	Load Balancer	2.0		○		○	
	10	Static Page	2.0		○		○	
	11	CDN	10.5		○		○	
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
			4	Flow Logs	0.5
			5	VPC Endpoint	0.5
			6	Endpoint Preparation Process	1.5
			7	Bastion Access to ECR	1.5
	2	Transit Between VPC	1	Transit Gateway Configure	0.5
	3	Bastion Server	1	Bastion Configure	0.5
			2	Bastion Security	0.5
	4	Application Access Control	1	VPC Lattice Configure	1.5
			2	Healthcheck	1.5
			3	Healthcheck Access	1.5
	5	RDBMS	1	RDS Configure	0.5
			2	DB RollBack	0.5
	6	NoSQL	1	Table Configure	0.5
	7	Container Registry	1	ECR Configure	0.5
	8	Container Orchestration	1	EKS Configure	0.5
			2	EKS KMS Encryption	0.5
			3	DB Application Node Configure	0.5
			4	Other Application Node Configure	0.5
			5	Application Pods	0.5

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
1과제	9	Ingress	1	ALB Configure	0.5
			2	Customer API Test	0.5
			3	Order API Test	0.5
			4	Order API Test	0.5
	10	Static Page	1	S3 Bucket Configure	0.5
			2	S3 Objects	0.5
			3	S3 Access	0.5
	11	CDN	1	CloudFront Configure	1.5
			2	Redirect HTTPS	1.5
			3	Static Page Test	1.5
			4	S3 Caching	1.5
			5	Customer API Test	1.5
			6	Product API Test	1.5
			7	Order API Test	1.5
	총점				30

3) 채점내용

순번	사전준비	
0	1) wsc2024-bastion-ec2 서버에 SSH를 통해 접근합니다. 2) rm -rf ~/.aws를 진행합니다. 3) aws configure를 입력하고 default.region을 us-east-1으로 설정합니다.	
	위의 작업이 완료되면 "사전준비 완료! 채점 시작!" 이라는 문구가 출력됩니다.	
순번	채점항목	
1-1	1-1-A (명령어 입력)	aws ec2 describe-vpcs --filter Name=tag:Name,Values=wsc2024-ma-vpc --query "Vpcs[0].CidrBlock" \ ; aws ec2 describe-vpcs --filter Name=tag:Name,Values=wsc2024-prod-vpc --query "Vpcs[0].CidrBlock" \ ; aws ec2 describe-vpcs --filter Name=tag:Name,Values=wsc2024-storage-vpc --query "Vpcs[0].CidrBlock"
	1-1-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	"10.0.0.0/16" "172.16.0.0/16" "192.168.0.0/16"

순번	채점항목	
1-2	1-2-A (명령어 입력)	<pre>aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-ma-mgmt-sn-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-ma-mgmt-sn-b --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-prod-load-sn-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-prod-load-sn-b --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-prod-app-sn-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-prod-app-sn-b --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-storage-db-sn-a --query "Subnets[0].CidrBlock" \ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsc2024-storage-db-sn-b --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" "172.16.0.0/24" "172.16.1.0/24" "172.16.2.0/24" "172.16.3.0/24" "192.168.0.0/24" "192.168.1.0/24"</pre>

순번	채점항목	
1-3	1-3-A (명령어 입력)	<pre> aws ec2 describe-route-tables --filters "Name=tag:Name,Values=wsc2024-ma-mgmt-rt" --query "RouteTables[].Routes[?GatewayId != null &amp;&amp; starts_with(GatewayId, 'igw')].GatewayId" --output text \ ; aws ec2 describe-route-tables --filters "Name=tag:Name,Values=wsc2024-prod-load-rt" --query "RouteTables[].Routes[?GatewayId != null &amp;&amp; starts_with(GatewayId, 'igw')].GatewayId" --output text \ ; aws ec2 describe-route-tables --filters "Name=tag:Name,Values=wsc2024-prod-app-rt-a" --query "RouteTables[].Routes[?NatGatewayId != null].NatGatewayId" --output text \ ; aws ec2 describe-route-tables --filters "Name=tag:Name,Values=wsc2024-prod-app-rt-b" --query "RouteTables[].Routes[?NatGatewayId != null].NatGatewayId" --output text \ ; aws ec2 describe-route-tables --filters "Name=tag:Name,Values=wsc2024-storage-db-rt-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=wsc2024-storage-db-rt-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) <u>순서 중요</u>	<pre> "igw-" <u>로 시작하는 문구가 출력이 되는지 확인</u> "igw-" <u>로 시작하는 문구가 출력이 되는지 확인</u> "nat-" <u>로 시작하는 문구가 출력이 되는지 확인</u> "nat-" <u>로 시작하는 문구가 출력이 되는지 확인</u> wsc2024-storage-db-sn-a &lt;- <u>정확히 일치</u> wsc2024-storage-db-sn-b &lt;- <u>정확히 일치</u> </pre>

순번	채점항목	
1-4	1-4-A (명령어 입력)	<pre>VPC_ID=\$(aws ec2 describe-vpcs --filters "Name=tag:Name,Values=wsc2024-ma-vpc" --query "Vpcs[*].VpcId" --output text) aws ec2 describe-flow-logs --filter "Name=resource-id,Values=\$VPC_ID" --query "FlowLogs[*].FlowLogId" --output text</pre>
	1-4-A (예상 출력)	"fl-" <u>로 시작하는 문구가 출력이 되는지 확인</u>
1-5	1-5-A (명령어 입력)	aws ec2 describe-vpc-endpoints --query "VpcEndpoints[].ServiceName"
	1-5-A (예상 출력) <u>ecr.dkr , s3</u> <u>존재 여부 확인</u>	<pre>[     "com.amazonaws.us-east-1.s3",     "com.amazonaws.us-east-1.ecr.dkr", ]</pre>
1-6	1-6-A (명령어 입력)	<pre>POLICY_ARNS=\$(aws iam list-attached-role-policies --role-name wsc2024-bastion-role --query "AttachedPolicies[].PolicyArn" --output text) for POLICY_ARN in \$POLICY_ARNS; do     POLICY_VERSION=\$(aws iam get-policy --policy-arn \$POLICY_ARN --query "Policy.DefaultVersionId" --output text)     POLICY_DOCUMENT=\$(aws iam get-policy-version --policy-arn \$POLICY_ARN --version-id \$POLICY_VERSION --query "PolicyVersion.Document" --output json)     echo "\$POLICY_DOCUMENT" done</pre>
	1-6-A (예상 출력) <u>정확히 일치</u> <u>이 외 JSON</u> <u>출력되면 안 됨</u>	<pre>{     "Version": "2012-10-17",     "Statement": [         {             "Effect": "Allow",             "Action": "*",             "Resource": "*"         }     ] }</pre>



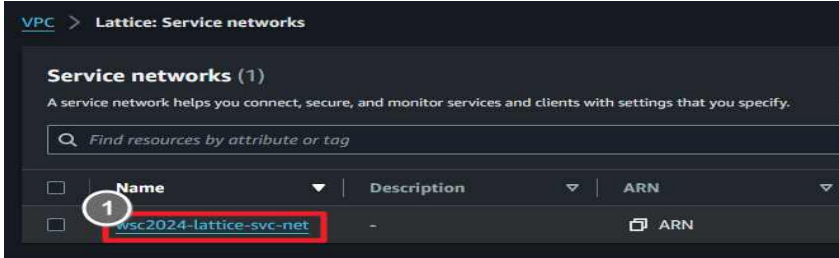
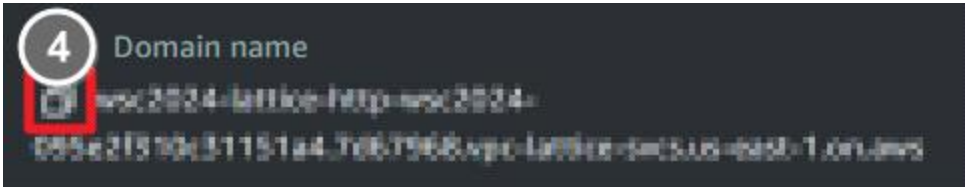
순번	채점항목	
1-6	1-6-B (명령어 입력)	<pre>export BUCKET_NAME="tesfsdfklsqwerlk sdf" export REGION="us-east-1" export FILE_NAME="test_upload.txt" export DOWNLOADED_FILE_NAME="downloaded_test_upload.txt" aws s3api create-bucket --bucket \$BUCKET_NAME --region \$REGION &gt; /dev/null 2&gt;&amp;1 echo "This is a test file for S3 upload and download." &gt; \$FILE_NAME aws s3 cp \$FILE_NAME s3://\$BUCKET_NAME/ &gt; /dev/null 2&gt;&amp;1 aws s3 cp s3://\$BUCKET_NAME/\$FILE_NAME \$DOWNLOADED_FILE_NAME</pre>
	1-6-B (예상 출력) <u>정확히 일치</u>	<pre>download: s3://tesfsdfklsqwerlk sdf/test_upload.txt to ./downloaded_test_upload.txt delete: s3://tesfsdfklsqwerlk sdf/test_upload.txt</pre>
1-7	1-7-A (명령어 입력) <u>(1-6 오답 시 진행 X)</u>	<pre>AWS_REGION=\$(aws configure get region) ACCOUNT_ID=\$(aws sts get-caller-identity --query Account --output text) docker rmi -f \$(docker images) 2&gt;/dev/null \ ; aws ecr get-login-password --region "\$AWS_REGION"   docker login --username AWS --password-stdin "\$ACCOUNT_ID.dkr.ecr.\$AWS_REGION.amazonaws.com" &gt; /dev/null 2&gt;&amp;1 \ ; docker pull \$ACCOUNT_ID.dkr.ecr.\$AWS_REGION.amazonaws.com/customer-repo:latest  aws s3 rm s3://\$BUCKET_NAME/\$FILE_NAME \ ; aws s3api delete-bucket --bucket \$BUCKET_NAME --region \$REGION</pre>
	1-7-A (예상 출력) <u>(1-6 오답 시 진행 X)</u> <u>AccessDenied</u> <u>출력 확인</u>	<pre>error pulling image configuration: download failed after attempts=1: denied: &lt;?xml version="1.0" encoding="UTF-8"?&gt; &lt;Error&gt; &lt;Code&gt;AccessDenied&lt;/Code&gt; &lt;Message&gt;Access Denied&lt;/Message&gt; &lt;RequestId&gt;D266MTTR9A8Y6X AQ&lt;/RequestId&gt; &lt;HostId&gt;oNG QmQ4OWAyNe/nB9X2st34y5tQWJILL/9mpwK56unHhOR2izzuNUByqiNIL10Jtw3vp MJg+vdQ=&lt;/HostId&gt; &lt;/Error&gt;</pre>

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     ROUTE_TABLES=\$(aws ec2 describe-transit-gateway-route-tables --filters "Name=transit-gateway-id,Values=\$TGW_ID" --query "TransitGatewayRouteTables[*].{Name:Tags[?Key=='Name'].Value [0]}" --output json)     ROUTE_TABLE_NAMES=\$(echo \$ROUTE_TABLES   jq -r '.[].Name')     for ROUTE_TABLE_NAME in \$ROUTE_TABLE_NAMES: do         echo "\$ROUTE_TABLE_NAME"     done done </pre>
	2-1-A (예상 출력) <b>정확히 일치</b> <b>순서 상관 없음</b>	<pre> wsc2024-vpc-tgw wsc2024-ma-tgw-attach wsc2024-prod-tgw-attach wsc2024-storage-tgw-attach wsc2024-ma-tgw-rt wsc2024-prod-tgw-rt wsc2024-storage-tgw-rt </pre>

순번	채점항목	
3-1	3-1-A (명령어 입력)	<pre> INSTANCE_NAME_TAG="wsc2024-bastion-ec2" INSTANCE_ID=\$(aws ec2 describe-instances --filters "Name=tag:Name,Values=\$INSTANCE_NAME_TAG" --query "Reservations[0].Instances[0].InstanceId" --output text) AMI_ID=\$(aws ec2 describe-instances --instance-ids "\$INSTANCE_ID" --query "Reservations[0].Instances[0].ImageId" --output text) AMI_DESCRIPTION=\$(aws ec2 describe-images --image-ids "\$AMI_ID" --query "Images[0].Description" --output text) INSTANCE_TYPE=\$(aws ec2 describe-instances --instance-ids "\$INSTANCE_ID" --query "Reservations[0].Instances[0].InstanceType" --output text) echo "\$AMI_DESCRIPTION" echo "\$INSTANCE_TYPE" </pre>
	3-1-A (예상 출력)	<p>"Amazon Linux 2023 AMI" <u>로 시작하는 문구가 출력이 되는지 확인</u></p> <p>t3.small &lt;- <u>정확히 일치</u></p>
3-2	3-2-A (명령어 입력)	<pre> aws ec2 describe-security-groups --filter Name=group-name,Values=wsc2024-bastion-sg --query "SecurityGroups[0].IpPermissions[0].{FromPort:FromPort,ToPort:ToPort,IpRanges:IpRa nges}" </pre>
	3-2-A (예상 출력) <u>정확히 일치</u>	<pre> [   {     "FromPort": 28282,     "ToPort": 28282,     "IpRanges": [       {         "CidrIp": "0.0.0.0/0" &lt;- <u>단일 IP만 허용할 수도 있음</u>       }     ]   } ] </pre>

순번	채점항목	
3-2	3-2-B (명령어 입력)	<pre> INSTANCE_NAME_TAG="wsc2024-bastion-ec2" INSTANCE_DESC=\$(aws ec2 describe-instances --filters "Name=tag:Name,Values=\$INSTANCE_NAME_TAG" --query "Reservations[0].Instances[0]" --output json) IAM_INSTANCE_PROFILE_ARN=\$(echo \$INSTANCE_DESC   jq -r '.IamInstanceProfile.Arn') ROLE_NAME=\$(aws iam get-instance-profile --instance-profile-name \$(echo \$IAM_INSTANCE_PROFILE_ARN   awk -F'/' '{print \$NF}') --query "InstanceProfile.Roles[0].RoleName" --output text) ROLE_POLICIES=\$(aws iam list-attached-role-policies --role-name "\$ROLE_NAME" --query "AttachedPolicies[].PolicyName" --output text) echo "\$ROLE_POLICIES" </pre>
	3-2-B (예상 출력) <u>정확히 일치</u>	AdministratorAccess
4-1	4-1-A (명령어 입력)	<pre> aws vpc-lattice list-service-networks --query "items[?name=='wsc2024-lattice-svc-net'].name" --output text SERVICE_NETWORK_ID=\$(aws vpc-lattice list-service-networks --query "items[?name=='wsc2024-lattice-svc-net'].id" --output text) SVC_ASSOCIATION=\$(aws vpc-lattice list-service-network-service-associations --service-network-identifier "\$SERVICE_NETWORK_ID" --query items[*].id --output text) VPC_ASSOCIATION=\$(aws vpc-lattice list-service-network-vpc-associations --service-network-identifier "\$SERVICE_NETWORK_ID" --query 'items[*].id' --output text) echo "\$SVC_ASSOCIATION" echo "\$VPC_ASSOCIATION" </pre>
	4-1-A (예상 출력) <u>순서 상관 있음</u>	<pre> wsc2024-lattice-svc-net &lt;- <u>정확히 일치</u> "snsa-" <u>로 시작하는 문구가 출력이 되는지 확인</u> "snva-" <u>로 시작하는 문구가 출력이 되는지 확인</u> </pre>

순번	채점항목	
4-2	4-2-A (명령어 입력)	<pre> TARGET_GROUP_ID=\$(aws vpc-lattice list-target-groups --target-group-type IP   jq -r '.items[].id') aws vpc-lattice list-targets --target-group-identifier "\$TARGET_GROUP_ID" </pre>
	4-2-A (예상 출력) <u>status가</u> <u>HEALTH인지</u> <u>확인 , 복수</u> <u>가능</u>	<pre> {   "items": [     {       "id": "172.16.3.145",       "port": 8080,       "status": "HEALTHY"     }   ] } </pre>
4-3	4-3-A (명령어 입력)	<pre> SERVICE_NETWORK_ID=\$(aws vpc-lattice list-service-networks --query "items[?name=='wsc2024-lattice-svc-net'].id" --output text) aws vpc-lattice list-service-network-service-associations --service-network-identifier "\$SERVICE_NETWORK_ID" --query items[*].id --output text </pre>
	4-3-A (예상 출력)	"snsa-" <u>로 시작하는 문구 복사</u>

순번	채점항목	
4-3	<p>4-3-B (콘솔 접근)</p>	<p>1) wsc2024-lattice-svc-net 이라는 name을 가진 Service networks에 접속</p>  <p>2) 4-3-A에서 복사한 ID에 접속</p>  <p>3) Domain Name을 복사</p>  <p>4) curl http://&lt;4-3-B-3 항목에서 복사한 Domain&gt;/healthcheck</p>
	<p>4-3-B (예상 출력) <u>정확히 일치</u></p>	<pre>{"status": "ok."}</pre>

순번	채점항목	
5-1	5-1-A (명령어 입력)	<pre>aws rds describe-db-clusters --db-cluster-identifier wsc2024-db-cluster --query 'DBClusters[0].EngineVersion' --output text \ ; aws rds describe-db-clusters --db-cluster-identifier wsc2024-db-cluster --query 'DBClusters[0].MasterUsername' --output text \ ; aws rds describe-db-instances --query "DBInstances[?DBClusterIdentifier=='wsc2024-db-cluster'].DBInstanceClass" --output text \</pre>
	5-1-A (예상 출력)	<p>"8.0.mysql_aurora" 로 시작하는 문구가 출력이 되는지 확인</p> <p>admin &lt;- <u>정확히 일치</u></p> <p>db.t3.medium db.t3.medium &lt;- <u>정확히 일치</u></p>
5-2	5-2-A (명령어 입력)	<pre>aws rds describe-db-clusters --db-cluster-identifier wsc2024-db-cluster --query "DBClusters[0].BacktrackWindow" --output text</pre>
	5-2-A (예상 출력) <u>정확히 일치</u>	14400

순번	채점항목	
6-1	6-1-A (명령어 입력)	aws dynamodb describe-table --table-name order --query 'Table.KeySchema[?KeyType == `HASH`].AttributeName' --output text
	6-1-A (예상 출력) <u>정확히 일치</u>	id
7-1	7-1-A (명령어 입력)	aws ecr describe-repositories --query 'repositories[*].repositoryName' --output text
	7-1-A (예상 출력) <u>정확히 일치</u>	customer-repo    product-repo    order-repo
8-1	8-1-A (명령어 입력)	aws eks describe-cluster --name wsc2024-eks-cluster --query 'cluster.version' --output text \ ; aws eks describe-cluster --name wsc2024-eks-cluster --query 'cluster.logging.clusterLogging[].types'   jq . --output text
	8-1-A (예상 출력) <u>정확히 일치</u>	1.29 [ [ "api", "audit", "authenticator", "controllerManager", "scheduler" ] ]



순번	채점항목	
8-2	8-2-A (명령어 입력)	aws eks describe-cluster --name wsc2024-eks-cluster --query "cluster.encryptedConfig[].provider.keyArn" --output text
	8-2-A (예상 출력)	"arn:aws:kms:us-east-1" <u>로 시작하는 문구가 출력이 되는지 확인</u>
8-3	8-3-A (명령어 입력)	kubectl get node -l app=db -o json   jq -r '.items[].metadata.labels["eks.amazonaws.com/nodegroup"]' kubectl get nodes -l app=db -o json   jq -r '.items[].metadata.name' kubectl get nodes -l app=db -o json   jq -r '.items[]   .metadata.labels["beta.kubernetes.io/instance-type"]'
	8-3-A (예상 출력) <u>정확히 일치 ,</u> <u>복수로 출력되어야</u> <u>정답</u>	wsc2024-db-application-ng wsc2024-db-application-ng <u>"ip-" 로 시작하는 문구가 출력이 되는지 확인</u> <u>"ip-" 로 시작하는 문구가 출력이 되는지 확인</u> t3.medium t3.medium
8-4	8-4-A (명령어 입력)	kubectl get node -l app=other -o json   jq -r '.items[].metadata.labels["eks.amazonaws.com/nodegroup"]' kubectl get nodes -l app=other -o json   jq -r '.items[].metadata.name' kubectl get nodes -l app=other -o json   jq -r '.items[]   .metadata.labels["beta.kubernetes.io/instance-type"]'
	8-4-A (예상 출력)	wsc2024-other-ng wsc2024-other-ng <u>"ip-" 로 시작하는 문구가 출력이 되는지 확인</u> <u>"ip-" 로 시작하는 문구가 출력이 되는지 확인</u> t3.medium t3.medium

순번	채점항목	
8-5	8-5-A (명령어 입력)	kubectl get deploy -n wsc2024
	8-5-A (예상 출력) <u>정확히 일치</u>	<pre> === Container Orchestration : 8-5-A (Application Pods) === NAME                READY    UP-TO-DATE    AVAILABLE    AGE customer-deploy      2/2      2              2            24h order-deploy         2/2      2              2            24h product-deploy       2/2      2              2            24h </pre>
9-1	9-1-A (명령어 입력)	aws elbv2 describe-load-balancers --names wsc2024-alb --query "LoadBalancers[0].Scheme" --output text aws elbv2 describe-load-balancers --names wsc2024-alb --query "LoadBalancers[0].Type" --output text
	9-1-A (예상 출력) <u>정확히 일치</u>	internet-facing application
9-2	9-2-A (명령어 입력)	LBDNS=\$(aws elbv2 describe-load-balancers --names wsc2024-alb --query "LoadBalancers[0].DNSName" --output text) curl http://\$LBDNS/v1/customer -X POST -H 'Content-Type: application/json' -d '{"id": "3101", "name": "Lee", "gender": "18"}' echo "-"
	9-2-A (예상 출력) <u>정확히 일치</u>	{"customer":{"id":"3101","name":"Lee","gender":"18"},"message":"The customer is created."}

순번	채점항목	
9-3	9-3-A (명령어 입력)	LBDNS=\$(aws elbv2 describe-load-balancers --names wsc2024-alb --query "LoadBalancers[].DNSName" --output text) curl http://\$LBDNS/v1/product -X POST -H 'Content-Type: application/json' -d '{"id": "3201", "name": "kim", "category": "stduent"}' echo "-"
	9-3-A (예상 출력) <u>정확히 일치</u>	{"product":{"id":"3201","name":"kim","category":"stduent"},"message":"The product is created."}
9-4	9-4-A (명령어 입력)	LBDNS=\$(aws elbv2 describe-load-balancers --names wsc2024-alb --query "LoadBalancers[].DNSName" --output text) curl http://\$LBDNS/v1/order -X POST -H 'Content-Type: application/json' -d '{"id": "3301", "customerid": "3101", "productid": "3201"}' echo "-"
	9-4-A (예상 출력) <u>정확히 일치</u>	{"order":{"id":"3301","customerid":"3101","productid":"3201"},"message":"The order is created."}
10-1	10-1-A (명령어 입력)	aws s3 ls
	10-1-A (예상 출력)	2024-05-29 01:45:49 wsc2024-s3-static-zfff <u>wsc2024-s3-static-&lt;4자리 영문&gt; 이 출력되는지 확인</u>

순번	채점항목	
10-2	10-2-A (명령어 입력)	<pre>for bucket in \$(aws s3api list-buckets --query "Buckets[?starts_with(Name, 'wsc2024-s3-static')].Name" --output text); do aws s3 ls "s3://\$bucket" --recursive done</pre>
	10-2-A (예상 출력)	2024-05-30 02:19:44      10004 index.html <u><b>index.html 만 출력되는지 확인</b></u>
10-3	10-3-A (명령어 입력)	<pre>BUCKET_NAME=\$(aws s3api list-buckets --query "Buckets[?starts_with(Name, 'wsc2024-s3-static')].Name" --output text) curl https://s3.us-east-1.amazonaws.com/\$BUCKET_NAME/index.html</pre>
	10-3-A (예상 출력) <u><b>AccessDenied</b></u> <u><b>출력 확인</b></u>	<pre>&lt;?xml version="1.0" encoding="UTF-8"?&gt; &lt;Error&gt; &lt;Code&gt;AccessDenied&lt;/Code&gt; &lt;Message&gt; Access Denied&lt;/Message&gt; &lt;RequestId&gt;61ZYXR7KRGYYQV0F&lt;/RequestId&gt; &lt;HostId&gt;1DV S1AnW17Q1FBRzgyk37tJ36ONhVMkjin8M4A+mTm02SW2krmxIZ2uVIs5A25rYCTms R8OG+A+I=&lt;/HostId&gt;&lt;/Error&gt;</pre>

순번	채점항목	
11-1	11-1-A (명령어 입력)	aws cloudfront list-distributions --query "DistributionList.Items[].Origins.Items[].DomainName" --output text aws cloudfront list-distributions --query "DistributionList.Items[].IsIPv6Enabled" --output text ID=\$(aws cloudfront list-distributions --query "DistributionList.Items[].Id" --output text) aws cloudfront get-distribution-config --id \$ID --query 'DistributionConfig.PriceClass' --output text
	11-1-A (예상 출력)	"wsc2024-s3-static-" <u>로 시작하는 문구가 출력이 되는지 확인</u> "wsc2024-alb-" <u>로 시작하는 문구가 출력이 되는지 확인</u> False <- <u>정확히 일치</u> PriceClass_All <- <u>정확히 일치</u>
11-2	11-2-A (명령어 입력)	aws cloudfront list-distributions --query "DistributionList.Items[].DefaultCacheBehavior.ViewerProtocolPolicy" --output text aws cloudfront list-distributions --query "DistributionList.Items[].CacheBehaviors.Items[].ViewerProtocolPolicy" --output text
	11-2-A (예상 출력) <u>정확히 일치</u>	redirect-to-https redirect-to-https
11-3	11-3-A (명령어 입력)	DOMAIN=\$(aws cloudfront list-distributions --query "DistributionList.Items[].DomainName" --output text) curl https://\$DOMAIN 2>/dev/null   grep -oP '(?<=<h1>).*?(?=</h1>)'
	11-3-A (예상 출력) <u>정확히 일치</u>	Welcome to Cloud Computing

순번	채점항목	
11-4	11-4-A (명령어 입력) <b>한 번만 입력</b>	DOMAIN=\$(aws cloudfront list-distributions --query "DistributionList.Items[].DomainName" --output text) curl -s -I https://\$DOMAIN   grep -i x-cache
	11-4-A (예상 출력) <b>정확히 일치</b>	x-cache: Hit from cloudfront
11-5	11-5-A (명령어 입력)	DOMAIN=\$(aws cloudfront list-distributions --query "DistributionList.Items[].DomainName" --output text) curl https://\$DOMAIN/v1/customer?id=3101 echo "-"
	11-5-A (예상 출력) <b>정확히 일치</b>	{ "customer":{"id":"3101","name":"Lee","gender":"18"},"message":"The customer is well in database."}-
11-6	11-6-A (명령어 입력)	DOMAIN=\$(aws cloudfront list-distributions --query "DistributionList.Items[].DomainName" --output text) curl https://\$DOMAIN/v1/product?id=3201 echo "-"
	11-6-A (예상 출력) <b>정확히 일치</b>	{ "product":{"id":"3201","name":"kim","category":"stduent"},"message":"The product is well in database."}-
11-7	11-7-A (명령어 입력)	DOMAIN=\$(aws cloudfront list-distributions --query "DistributionList.Items[].DomainName" --output text) curl https://\$DOMAIN/v1/order?id=3301 echo "-"
	11-7-A (예상 출력) <b>정확히 일치</b>	{ "order":{"id":"3301","customerid":"3101","productid":"3201"},"message":"The order is well in database."}-