

# 2024년도 전국기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅
<p>※ 다음 사항을 유의하여 채점하십시오.</p> <p>1) AWS의 지역은 ap-northeast-2을 사용합니다.</p> <p>2) 웹페이지 접근은 크롬이나 파이어폭스를 이용합니다.</p> <p>3) 웹페이지에서 언어에 따라 문구가 다르게 보일 수 있습니다.</p> <p>4) shell에서의 명령어의 출력은 버전에 따라 조금 다를 수 있습니다.</p> <p>5) 문제지와 채점지에 있는 ◇ 는 변수입니다. 해당 부분을 변경해 입력합니다.</p> <p>6) 채점은 문항 순서대로 진행해야 합니다.</p> <p>7) 삭제된 채점자료는 되돌릴 수 없음으로 유의하여 진행하며, 이의신청까지 완료 이후 선수가 생성한 클라우드 리소스를 삭제합니다.</p> <p>8) 부분 점수가 있는 문항은 채점 항목에 부분 점수가 적혀져 있습니다.</p> <p>9) 부분 점수가 따로 없는 문항은 모두 맞아야 점수로 인정됩니다.</p> <p>10) 리소스의 정보를 읽어오는 채점항목은 기본적으로 스크립트 결과를 통해 채점을 진행하며, 만약 선수가 이의가 있다면 명령어를 직접 입력하여 확인해 볼 수 있습니다.</p> <p>11) [ ] 기호는 채점에 영향을 주지 않습니다.</p> <p>12) 명령어 입력 Box 안의 명령줄은 한 줄 명령어입니다. 별도의 지시가 없으면 수정 없이 박스 안의 전체 내용을 복사하고 쉘에 붙여넣어 명령을 실행합니다.</p> <p>13) (예상 출력)은 바로 이전 (명령어 입력)의 예상 출력을 의미합니다.</p>		

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	Networking	3.2		○		○	
	2	Bastion	1.6		○		○	
	3	DynamoDB	1.6		○		○	
	4	RDS	1.8		○		○	
	5	ECR	1.6		○		○	
	6	EKS	6.2		○		○	
	7	Load Balancer	1.0		○		○	
	8	S3	2.4		○		○	
	9	CloudFront	2.4		○		○	
	10	App Service	5.6		○		○	
	11	Logging	1.8		○		○	
	12	Monitoring	0.8		○		○	
합 계			30					

## 2) 채점방법 및 기준

( 경기종료 후 채점 )

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
제1과제	1	Networking	1	VPC & Subnet	0.8
			2	Gateway & Route Table	0.8
			3	S3 & DynamoDB Endpoint	0.8
			4	VPC Flow Logs	0.8
	2	Bastion	1	Instance Config	0.8
			2	Security Group	0.8
	3	DynamoDB	1	Database Config	0.8
			2	Backup	0.8
	4	RDS	1	Database Config	0.8
			2	Security Group	1.0
	5	ECR	1	Repository Config	0.8
			2	Replication	0.8
	6	EKS	1	Cluster Config	0.8
			2	Addon Nodegroup Config	0.8
			3	App Nodegroup Config	0.8
			4	App Fargate Config	0.8
			5	Metadata Access	1.0
			6	Secret Rotation	1.0
			7	QoS Class	1.0
	7	Load Balancer	1	Access Timeout	1.0
	8	S3	1	Encryption	0.8
			2	Bucket Policy	0.8
			3	Replication	0.8
	9	CloudFront	1	CDN Config	0.8
			2	Static Caching	0.8
			3	HTTPS Redirect	0.8
	10	App Service	1	Customer Service Test	1.2
			2	Product Service Test	1.2
			3	Order Service Test	1.2
			4	Origin Group Test	1.0
			5	Service Unavailable Test	1.0
	11	Logging	1	Basic Logging Test	0.8
			2	Health Check Exclude Test	1.0
	12	Monitoring	1	Container Insights	0.8
					30

### 3) 채점내용

순번	사전준비
0	<p>1) Bastion 서버에 SSH를 통해 접근합니다. (별도 명시가 없는 경우 모든 채점은 Bastion 서버에서 진행합니다.)</p> <p>2) Bastion 명령어 및 권한을 확인합니다. (awscli permission, jq, curl, awscli region)</p> <p>3) marking 스크립트들을 /root/marking에 다운로드 합니다.</p> <p>4) /root/marking 경로에서 스크립트를 실행합니다. 실행 결과를 기반으로 채점을 진행하되 선수가 이의를 제기할 경우 수동으로 채점을 진행할 수 있도록 합니다.</p> <p>5) 채점을 진행하는 Bastion 서버의 쉘을 초기 실행할 때 다음 명령어를 실행하여 환경 변수를 초기화합니다. <b>(채점 스크립트로 진행 시 생략)</b></p> <p>6) marking2.sh는 인프라에 영향이 있는 스크립트입니다. 반드시 marking1.sh를 실행한 후, 진행해야 합니다.</p>
	<pre>export DistributionID="<u>&lt;Cloudfront_Distribution_ID&gt;</u>" export AP_BUCKET="ap-wsi-static-<u>&lt;4words&gt;</u>" export US_BUCKET="us-wsi-static-<u>&lt;4words&gt;</u>" export CF_DOMAIN=\$(aws cloudfront get-distribution --id \${DistributionID} --query "Distribution.DomainName" --output text)</pre>
	<p>6) 채점을 진행하기 전에 다음 명령어를 수행하여 채점 진행을 위한 사전 작업을 진행합니다. <b>(채점 스크립트로 진행 시 생략)</b></p> <pre># set default region of aws cli aws configure set default.region ap-northeast-2 # set default output of aws cli aws configure set output json # clear CDN cache (perform CloudFront invalidation) export InvalidationID=\$(aws cloudfront create-invalidation --distribution-id \${DistributionID} --paths "/" --query "Invalidation.Id" --output text) aws cloudfront wait invalidation-completed --distribution-id \${DistributionID} --id \${InvalidationID}</pre>

순번	채점 항목	
1-1	1-1-A (명령어 입력)	<pre>aws ec2 describe-vpcs --filter Name=tag:Name,Values=ws1-vpc --query "Vpcs[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=ws1-app-a --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=ws1-app-b --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=ws1-public-a --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=ws1-public-b --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=ws1-data-a --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=ws1-data-b --query "Subnets[0].CidrBlock"</pre>
	1-1-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	<pre>"10.1.0.0/16" "10.1.0.0/24" "10.1.1.0/24" "10.1.2.0/24" "10.1.3.0/24" "10.1.4.0/24" "10.1.5.0/24"</pre>
1-2	1-2-A (명령어 입력)	<pre>aws ec2 describe-route-tables --filter Name=tag:Name,Values=ws1-app-a-rt --query "RouteTables[].Routes[].NatGatewayId"   grep "nat-"   wc -l ₩ ; aws ec2 describe-route-tables --filter Name=tag:Name,Values=ws1-app-b-rt --query "RouteTables[].Routes[].NatGatewayId"   grep "nat-"   wc -l ₩ ; aws ec2 describe-route-tables --filter Name=tag:Name,Values=ws1-public-rt --query "RouteTables[].Routes[]"   grep "igw-"   wc -l ₩ ; aws ec2 describe-route-tables --filter Name=tag:Name,Values=ws1-data-rt --query "RouteTables[].Routes[]"   grep -E "igw- nat-"   wc -l</pre>
	1-2-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	<pre>1 1 1 0</pre>

순번	채점 항목	
1-3	1-3-A (명령어 입력)	aws ec2 describe-vpc-endpoints --query "VpcEndpoints[].ServiceName"
	1-3-A (예상 출력) <u>정확히 일치</u>	[ "com.amazonaws.ap-northeast-2.dynamodb", "com.amazonaws.ap-northeast-2.s3" ]
1-4	1-4-A (명령어 입력)	aws ec2 describe-flow-logs --query "FlowLogs[].LogGroupName"
	1-4-A (예상 출력) <u>정확히 일치</u>	[ "/aws/vpc/wsi-vpc" ]
2-1	2-1-A (명령어 입력)	aws ec2 describe-instances --filter Name=tag:Name,Values=ws-bastion --query "Reservations[0].Instances[0].InstanceType" aws ec2 describe-instances --filter Name=tag:Name,Values=ws-bastion --query "Reservations[0].Instances[0].IamInstanceProfile.Arn" --output text   cut -d '/' -f 2
	2-1-A (예상 출력) <u>정확히 일치</u>	"t3.small" ws-bastion-role

순번	채점 항목	
2-2	2-3-A (명령어 입력)	aws ec2 describe-instances --filter Name=tag:Name,Values=wsi-bastion --query "Reservations[0].Instances[0].SecurityGroups[0].GroupName" ₩ ; aws ec2 describe-security-groups --filter Name=group-name,Values=wsi-bastion-sg --query "SecurityGroups[0].IpPermissions[0].{FromPort:FromPort,ToPort:ToPort,IpRanges:IpRanges}"
	2-3-A (예상 출력) <u>정확히 일치</u> <u>Description</u> 무시	"wsi-bastion-sg" [ { "ToPort": 4272, "FromPort": 4272, "IpRanges": [ { "CidrIp": "0.0.0.0/0" → 단일 IP만 허용할 수도 있음 } ] } ]
3-1	3-1-A (명령어 입력)	aws dynamodb describe-table --table-name order --query "Table.{KeySchema:KeySchema[?KeyType=='HASH'],SSEType:SSEDescription.SSEType,BillingMode:BillingModeSummary.BillingMode}"
	3-1-A (예상 출력) <u>정확히 일치</u>	{ "KeySchema": [ { "AttributeName": "id", "KeyType": "HASH" } ], "SSEType": "KMS", "BillingMode": "PAY_PER_REQUEST" }

순번	채점 항목	
3-2	3-3-A (명령어 입력)	aws dynamodb describe-continuous-backups --table-name order --query "ContinuousBackupsDescription.ContinuousBackupsStatus"
	3-3-A (예상 출력) <u>정확히 일치</u>	"ENABLED"
4-1	4-1-A (명령어 입력)	aws rds describe-db-instances --db-instance-identifier wsi-rds-mysql --query "DBInstances[0].{Engine:Engine,MultiAZ:MultiAZ,DBInstanceStatus:DBInstanceStatus,DBInstanceClass:DBInstanceClass,StorageEncrypted:StorageEncrypted,EnabledCloudwatchLogsExports:EnabledCloudwatchLogsExports,Port:Endpoint.Port}"
	4-1-A (예상 출력) <u>정확히 일치</u>	{ "Engine": "mysql", "MultiAZ": true, "DBInstanceStatus": "available", "DBInstanceClass": "db.m5.xlarge", "StorageEncrypted": true, "EnabledCloudwatchLogsExports": [ "audit", "error", "general", "slowquery" ], "Port": 3310 [3306을 제외한 아무 숫자] }
4-2	4-2-A (명령어 입력)	aws ec2 describe-security-groups --group-ids \$(aws rds describe-db-instances --db-instance-identifier wsi-rds-mysql --query "DBInstances[0].VpcSecurityGroups[0].VpcSecurityGroupId" --output text) --query "SecurityGroups[0].IpPermissions[0].UserIdGroupPairs[0].GroupId"
	4-2-A (예상 출력)	[ "sg-0951e9f7602cc73e0" ]  (여러 개가 나와도 무관하나 모두 "sg-"로 시작하는 형태여야 함)



순번	채점 항목	
5-1	5-1-A (명령어 입력)	aws ecr describe-repositories --repository-names "customer" "product" "order" --query "repositories[].{imageTagMutability:imageTagMutability,scanOnPush:imageScanning Configuration.scanOnPush,encryptionConfiguration:encryptionConfiguration.encryp tionType}"
	5-1-A (예상 출력) <u>정확히 일치</u>	[ { "imageTagMutability": "IMMUTABLE", "scanOnPush": true, "encryptionConfiguration": "KMS" }, { "imageTagMutability": "IMMUTABLE", "scanOnPush": true, "encryptionConfiguration": "KMS" }, { "imageTagMutability": "IMMUTABLE", "scanOnPush": true, "encryptionConfiguration": "KMS" } ]
5-2	5-2-A (명령어 입력)	aws ecr describe-registry --query "replicationConfiguration.rules[0].destinations[0].region"
	5-2-A (예상 출력) <u>정확히 일치</u>	"us-east-1"

순번	채점 항목	
6-1	6-1-A (명령어 입력)	<pre>aws eks describe-cluster --name wsi-eks-cluster --query "cluster.{version:version,endpointPublicAccess:resourcesVpcConfig.endpointPublicAccess,endpointPrivateAccess:resourcesVpcConfig.endpointPrivateAccess,logging:logging,encryption:encryptionConfig[0].resources}"</pre>
	6-1-A (예상 출력) <u>정확히 일치</u>	<pre>{   "version": "1.29",   "endpointPublicAccess": false,   "endpointPrivateAccess": true,   "logging": {     "clusterLogging": [       {         "types": [           "api",           "audit",           "authenticator",           "controllerManager",           "scheduler"         ],         "enabled": true       }     ],     "encryption": [       "secrets"     ]   } }</pre>

순번	채점 항목	
6-2	6-2-A (명령어 입력)	<pre>aws eks describe-nodegroup --cluster-name wsi-eks-cluster --nodegroup-name wsi-addon-nodegroup --query "nodegroup.{instanceType:instanceTypes[0],amiType:amiType}"; kubectl get no -l "eks.amazonaws.com/nodegroup=wsi-addon-nodegroup" --output json   jq ".items[].metadata.labels   .W"eks.amazonaws.com/nodegroupW" + W" W" + .W"topology.kubernetes.io/zoneW"";</pre>
	6-2-A (예상 출력) <u>정확히 일치</u>	<pre>{   "instanceType": "t4g.large",   "amiType": "BOTTLEROCKET_ARM_64" } "wsi-addon-nodegroup ap-northeast-2a" "wsi-addon-nodegroup ap-northeast-2b"</pre>
6-3	6-3-A (명령어 입력)	<pre>aws eks describe-nodegroup --cluster-name wsi-eks-cluster --nodegroup-name wsi-app-nodegroup --query "nodegroup.{instanceType:instanceTypes[0],amiType:amiType}"; kubectl get no -l "eks.amazonaws.com/nodegroup=wsi-app-nodegroup" --output json   jq ".items[].metadata.labels   .W"eks.amazonaws.com/nodegroupW" + W" W" + .W"topology.kubernetes.io/zoneW"";</pre>
	6-3-A (예상 출력) <u>정확히 일치</u>	<pre>{   "instanceType": "m5.xlarge",   "amiType": "BOTTLEROCKET_x86_64" } "wsi-app-nodegroup ap-northeast-2a" "wsi-app-nodegroup ap-northeast-2b"</pre>

순번	채점 항목	
6-4	6-4-A (명령어 입력)	aws eks describe-fargate-profile --cluster-name wsi-eks-cluster --fargate-profile-name wsi-app-fargate --query "fargateProfile.{namespace:selectors[0].namespace,status:status}"
	6-4-A (예상 출력) <u>정확히 일치</u>	{ "namespace": "wsi", "status": "ACTIVE" }
6-5	6-5-A (명령어 입력)	kubectrl exec -n wsi -it \$(kubectrl get pods -n wsi --no-headers -o custom-columns=":metadata.name"   grep customer   head -n 1) -- curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600" --max-time 10
	6-5-A (예상 출력)	curl: (28) Operation timed out after 10002 milliseconds with 0 bytes received (“Operation timed out” 메시지가 나와야 함)
6-6	6-6-A (명령어 입력)	aws secretsmanager rotate-secret --secret-id \$(aws secretsmanager list-secrets --query 'SecretList[?starts_with(Name, `rds!`)].Name' --output text) sleep 2m aws secretsmanager get-secret-value --secret-id \$(aws secretsmanager list-secrets --query 'SecretList[?starts_with(Name, `rds!`)].Name' --output text) --query "SecretString" --output text   jq -r .password kubectrl exec -n wsi -it \$(kubectrl get pods -n wsi --no-headers -o custom-columns=":metadata.name"   grep customer   head -n 1) -- /bin/sh -c 'echo \$MYSQL_PASSWORD' kubectrl exec -n wsi -it \$(kubectrl get pods -n wsi --no-headers -o custom-columns=":metadata.name"   grep product   head -n 1) -- /bin/sh -c 'echo \$MYSQL_PASSWORD'
	6-6-A (예상 출력)	(마지막 3줄이 모두 일치해야 함)

순번	채점 항목	
6-7	6-7-A (명령어 입력)	kubectl describe pod -n wsi \$(kubectl get pods -n wsi --no-headers -o custom-columns=":metadata.name"   grep customer   head -n 1)   grep QoS
	6-7-A (예상 출력) <b>정확히 일치</b>	QoS Class: Guaranteed
7-1	7-1-A (명령어 입력)	curl \$(aws elbv2 describe-load-balancers --query "LoadBalancers[0].DNSName" --output text) --max-time 10
	7-1-A (예상 출력)	curl: (28) Connection timed out after 10002 milliseconds ("Connection timed out" 메시지가 나와야 함)
8-1	8-1-A (명령어 입력)	aws s3api get-bucket-encryption --bucket \$AP_BUCKET --query "ServerSideEncryptionConfiguration.Rules[0].ApplyServerSideEncryptionByDefault.SSEAlgorithm"
	8-1-A (예상 출력) <b>정확히 일치</b>	aws s3api get-bucket-encryption --bucket \$US_BUCKET --query "ServerSideEncryptionConfiguration.Rules[0].ApplyServerSideEncryptionByDefault.SSEAlgorithm" "aws:kms" "aws:kms"
8-2	8-2-A (명령어 입력)	aws s3api get-bucket-policy --bucket \$AP_BUCKET --query "Policy" --output text   jq .Statement[].Principal.Service aws s3api get-bucket-policy --bucket \$US_BUCKET --query "Policy" --output text   jq .Statement[].Principal.Service
	8-2-A (예상 출력) <b>정확히 일치</b>	"cloudfront.amazonaws.com" "cloudfront.amazonaws.com"

순번	채점 항목	
8-3	8-3-A (명령어 입력)	<pre> echo "WorldSkills" &gt; sample.txt aws s3api put-object --bucket \$AP_BUCKET --key sample.txt --body sample.txt sleep 30 aws s3api get-object --bucket \$US_BUCKET --key sample.txt replication.txt cat replication.txt </pre>
	8-3-A (예상 출력)	<pre> WorldSkills (마지막 줄) </pre>
9-1	9-1-A (명령어 입력)	<pre> aws cloudfront list-tags-for-resource --resource "arn:aws:cloudfront::\$(aws sts get-caller-identity --query Account --output text):distribution/\${DistributionID}" --query "Tags.Items[?Key=='Name']"; aws cloudfront get-distribution --id \${DistributionID} --query "Distribution.DistributionConfig.{PriceClass:PriceClass,IsIPV6Enabled:IsIPV6Enabled}" </pre>
	9-1-A (예상 출력)	<pre> [   {     "Key": "Name",     "Value": "wsi-cdn"   } ] {   "PriceClass": "PriceClass_All",   "IsIPV6Enabled": false } </pre>

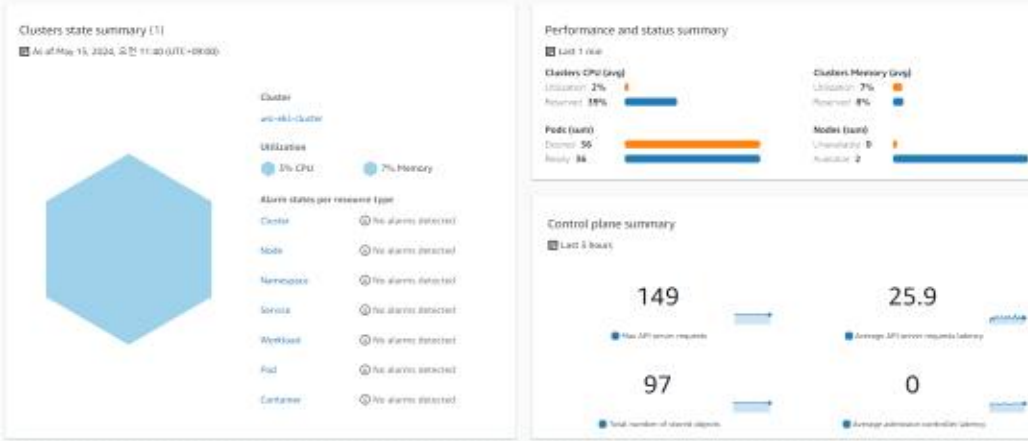

순번	채점 항목	
9-2	9-2-A (명령어 입력)	<pre>curl --silent -i -X GET --max-time 5 -w "%{http_code}%n" https://\${CF_DOMAIN}/index.html   grep -iE "x-cache: ^200\$"</pre>
	9-2-A (예상 출력) <b>정확히 일치</b>	<pre>x-cache: Hit from cloudfront 200</pre> <p>(최대 5회까지 시도 가능)</p>
9-3	9-3-A (명령어 입력)	<pre>curl --silent -i -X GET --max-time 5 -w "%{http_code}%n" http://\${CF_DOMAIN}/index.html   grep -iE "x-cache: ^301\$"</pre>
	9-3-A (예상 출력) <b>정확히 일치</b>	<pre>X-Cache: Redirect from cloudfront 301</pre>
10-1	10-1-A (명령어 입력 & 브라우저 접속)	<ol style="list-style-type: none"> <li>1. echo \${CF_DOMAIN}/index.html</li> <li>2. 브라우저로 출력값 URL에 접속</li> <li>3. Customer - POST에서 (ID : wsi-customer, Name : example, Gender : male) 입력 후 Save 버튼 클릭</li> <li>4. Customer - GET에서 (ID : wsi-customer) 입력 후 Load 버튼 클릭</li> </ol>
	10-1-A (예상 출력) <b>정확히 일치</b>	<p>다음과 같은 화면 확인</p> 

순번	채점 항목	
10-2	10-2-A (브라우저 접속)	1. 브라우저 접속 유지 (새로고침 필요) 2. Product - POST에서 (ID : wsi-product, Name : example, Category : cloud) 입력 후 Save 버튼 클릭 3. Product - GET에서 (ID : wsi-product) 입력 후 Load 버튼 클릭
	10-2-A (예상 출력) <u>정확히 일치</u>	다음과 같은 화면 확인 
10-3	10-3-A (브라우저 접속)	1. 브라우저 접속 유지 (새로고침 필요) 2. Order - POST에서 (ID : wsi-order, Customer ID : wsi-customer, Product ID : wsi-product) 입력 후 Save 버튼 클릭 3. Order - GET에서 (ID : wsi-order) 입력 후 Load 버튼 클릭
	10-3-A (예상 출력) <u>정확히 일치</u>	다음과 같은 화면 확인 



순번	채점 항목	
10-4	10-4-A (명령어 입력)	<pre>aws s3 rm s3://\$AP_BUCKET/index.html export InvalidationID=\$(aws cloudfront create-invalidation --distribution-id \${DistributionID} --paths "/index.html" --query "Invalidation.Id" --output text) aws cloudfront wait invalidation-completed --distribution-id \${DistributionID} --id \${InvalidationID} curl --silent -o /dev/null -X GET --max-time 5 -w "%{http_code}\n" https://\${CF_DOMAIN}/index.html</pre>
	10-4-A (예상 출력) <u>정확히 일치</u>	200 (마지막 줄)
10-5	10-5-A (명령어 입력)	<pre>aws s3 rm s3://\$US_BUCKET/index.html export InvalidationID=\$(aws cloudfront create-invalidation --distribution-id \${DistributionID} --paths "/index.html" --query "Invalidation.Id" --output text) aws cloudfront wait invalidation-completed --distribution-id \${DistributionID} --id \${InvalidationID} curl --silent -o /dev/null -X GET --max-time 5 -w "%{http_code}\n" https://\${CF_DOMAIN}/index.html</pre>
	10-5-A (예상 출력) <u>정확히 일치</u>	503 (마지막 줄)

순번	채점 항목	
11-1	11-1-A (명령어 입력)	<pre>curl --silent --output /dev/null "https://\${CF_DOMAIN}/v1/customer?id=worldskillstest" curl --silent --output /dev/null "https://\${CF_DOMAIN}/v1/product?id=worldskillstest" curl --silent --output /dev/null "https://\${CF_DOMAIN}/v1/order?id=worldskillstest" sleep 1m aws logs filter-log-events --log-group-name /wsi/webapp/customer --filter-pattern "/v1/customer?id=worldskillstest"   jq ".events   length" aws logs filter-log-events --log-group-name /wsi/webapp/product --filter-pattern "/v1/product?id=worldskillstest"   jq ".events   length" aws logs filter-log-events --log-group-name /wsi/webapp/order --filter-pattern "/v1/order?id=worldskillstest"   jq ".events   length"</pre>
	11-1-A (예상 출력) <u>정확히 일치</u>	1 1 1
11-2	11-2-A (명령어 입력)	<pre>aws logs filter-log-events --log-group-name /wsi/webapp/customer --filter-pattern "/healthcheck"   jq ".events   length" aws logs filter-log-events --log-group-name /wsi/webapp/product --filter-pattern "/healthcheck"   jq ".events   length" aws logs filter-log-events --log-group-name /wsi/webapp/order --filter-pattern "/healthcheck"   jq ".events   length"</pre>
	11-2-A (예상 출력) <u>정확히 일치</u>	0 0 0

순번	채점 항목	
12-1	12-1-A (콘솔 접속)	1. CloudWatch 서비스 콘솔에 접속합니다. 2. 좌측 메뉴 중 인사이트 > Container Insights를 클릭합니다,
	12-1-A (예상 출력) <u>정확히 일치</u>	<p>다음과 같이 여러 수치가 정확히 보이는 화면이어야 합니다.</p> <p>&lt;알맞은 예시&gt;</p>  <p>The screenshot shows the 'Clusters state summary' for 'api-eks-cluster' with a large blue hexagon icon. The 'Performance and status summary' section shows 'Clusters CPU (avg)' utilization at 2% (15% reserved) and 'Clusters Memory (avg)' utilization at 7% (8% reserved). The 'Control plane summary' shows 'Max API server requests' at 149 and 'Average API server requests latency' at 25.9ms. The 'Pods (sum)' section shows 56 desired and 34 ready pods. The 'Nodes (sum)' section shows 0 unavailable nodes and 2 ready nodes.</p> <p>&lt;맞지 않은 예시&gt;</p>  <p>The screenshot shows the same console but with placeholder values. The 'Clusters CPU (avg)' section shows utilization at 0% and reserved at 0%. The 'Clusters Memory (avg)' section shows utilization at 0% and reserved at 0%. The 'Control plane summary' shows 'Max API server requests' at 0 and 'Average API server requests latency' at 0. The 'Pods (sum)' section shows 0 desired and 0 ready pods. The 'Nodes (sum)' section shows 0 unavailable and 0 ready nodes.</p>