2024년도 전국기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅

※ 다음 사항을 유의하여 채점하시오.

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

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	네트워크 구성	3.5		○		○	
	2	Bastion	3		○		○	
	3	S3	1.5		○		○	
	4	Control Plane	4		○		○	
	5	ECR	1.5		○		○	
	6	EKS	5.5		○		○	
	7	Deployment	1		○		○	
	8	LB	3		○		○	
	9	관계형 데이터베이스	1		○		○	
	10	S3 Presigned URL	2		○		○	
	11	CloudFront	4		○		○	
합 계			30					

2) 채점방법 및 기준

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
1과제	1	네트워크 구성	1	VPC, Subnet	0.5
			2	Routing	0.5
			3	VPC FlowLogs	2.5
	2	Bastion	1	Bastion Configurations	0.25
			2	Bastion IAM Role	0.25
			3	Bastion SG Ingress Rule Revoke	1
			4	Bastion SSH Logs	1.5
	3	S3	1	S3 Bucket Configuration	0.5
			2	Web Static	1
	4	Control Plane	1	Server Configuration	0.5
			2	Linux Account Configuration	1
			3	Security Group	0.5
			4	Role-Based Access Control	2
	5	ECR	1	Container Image	0.5
			2	Container User	1
	6	EKS	1	EKS Configuration	1.5
			2	Pod Label	1.5
			3	Calico Networking	2.5
	7	Deployment	1	Deployment Configuration	0.5
			2	Service Configuration	0.5
	8	LB	1	Load Balancer Configuration	0.5
			2	LB Routing	1
			3	Web Application	1.5
	9	관계형 데이터베이스	1	RDS Configuration	1

	10	S3 Presigned URL	1	Configuration	0.5
			2	API Request	1.5
	11	CloudFront	1	S3	2
			2	ALB	2
	총점				30

3) 채점내용

순번	사전준비
0	1) Bastion 서버에 SSH를 통해 접근합니다. (별도 명시가 없는 경우 모든 채점은 Bastion 서버에서 진행합니다.) 2) Bastion 명령어 및 권한을 확인합니다. (awscli permission, jq, curl, awscli region) 3) Bastion 비밀번호로 접속 시 Skills2024**로 접속 한지 확인합니다. 4) 제공된 img 폴더를 s3에 업로드 합니다.
	# set DistributionID read -p "DistributionID <Cloudfront_Distribution_ID>: " DistributionID # set default region of aws cli aws configure set default.region ap-northeast-2 # set default output of aws cli aws configure set default.output json

순번	채점 항목	
1-1	1-1-A (명령어 입력)	aws ec2 describe-vpcs --filter Name=tag:Name,Values=wsi-vpc --query "Vpcs[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-public-a --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-public-b --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-public-c --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-private-a --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-private-b --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-private-c --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-data-a --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-data-b --query "Subnets[0].CidrBlock" ₩ ; aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-data-c —query "Subnets[0].CidrBlock"
	1-1-A (예상 출력) **정확히 일치** **순서 중요**	"10.0.0.0/16" "10.0.11.0/24" "10.0.12.0/24" "10.0.13.0/24" "10.0.101.0/24" "10.0.102.0/24" "10.0.103.0/24" "10.0.201.0/24" "10.0.202.0/24" "10.0.203.0/24"

순번	채점 항목	
1-2	1-2-A (명령어 입력)	aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-public-rt --query "RouteTables[].Routes[]" | grep "igw-" | wc -l ₩ ; aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-private-a-rt --query "RouteTables[].Routes[].NatGatewayId" | grep "nat-" | wc -l ₩ ; aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-private-b-rt --query "RouteTables[].Routes[].NatGatewayId" | grep "nat-" | wc -l ₩ ; aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-private-c-rt --query "RouteTables[].Routes[].NatGatewayId" | grep "nat-" | wc -l ₩ ; aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-data-rt --query "RouteTables[].Routes[]" | grep -E "igw-|nat-" | wc -l
	1-2-A (예상 출력) **정확히 일치** **순서 중요**	1 1 1 1 0
1-3	1-3A (명령어 입력)	aws ec2 describe-flow-logs --query "FlowLogs[0].Tags[0].Value" ₩ ; aws ec2 describe-flow-logs --query "FlowLogs[0].LogFormat" ₩ ; LATEST_LOG_STREAM=$(aws logs describe-log-streams --log-group-name wsi-traffic-logs --order-by LastEventTime --descending --limit 1 --query "logStreams[0].logStreamName" --output text) ₩ ; aws logs get-log-events --log-group-name wsi-traffic-logs --log-stream-name $LATEST_LOG_STREAM --limit 5 --query "events[*].{message:message}" --output text
	1-3-A (예상 출력) **정확히 일치** **순서 중요**	"wsi-traffic-logs" "${region} ${vpc-id} ${action} ${instance-id}" ap-northeast-2 vpc-xxxxxxxxxxxxxxx ACCEPT - ap-northeast-2 vpc-xxxxxxxxxxxxxxx REJECT - ap-northeast-2 vpc-xxxxxxxxxxxxxxx ACCEPT i-xxxxxxxxxxxxxxx ap-northeast-2 vpc-xxxxxxxxxxxxxxx REJECT i-xxxxxxxxxxxxxxx ap-northeast-2 vpc-xxxxxxxxxxxxxxx ACCEPT - (로그 5개중에 랜덤으로 나올 수도 있고 동일 하게 나올 수 있습니다.)

순번	채점 항목	
2-1	2-1A (명령어 입력)	aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-bastion-ec2" "Name=tag:ec2,Values=bastion" --query "Reservations[0].Instances[0].InstanceType" ₩ ; BASTION_SUBNET=$(aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-bastion-ec2" "Name=tag:ec2,Values=bastion" --query "Reservations[*].Instances[*].SubnetId" --output text) ₩ ; aws ec2 describe-subnets --subnet-ids $BASTION_SUBNET --query "Subnets[*].Tags[?Key=='Name'].Value" --output text | grep wsi-public-c ₩ ; echo "1.)" | grep "1.)"; aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-bastion-ec2" "Name=tag:ec2,Values=bastion" --query "Reservations[].Instances[].PublicIpAddress" ₩ ; echo "2.)" | grep "2.)"; aws ec2 describe-addresses --query "Addresses[].PublicIp"
	2-1-A (예상 출력) **정확히 일치** **순서 중요**	"m5.large" wsi-public-c 1.) 12.xxx.xxx.123 2.) 12.xxx.xxx.123 (1번와 2번 동일한 IP 주소가 있는지 확인합니다.)
2-2	2-2A (명령어 입력)	INSTANCE_ID=$(aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-bastion-ec2" "Name=tag:ec2,Values=bastion" --query "Reservations[*].Instances[*].InstanceId" --output text) ₩ ; aws ec2 describe-instances --instance-ids $INSTANCE_ID --query "Reservations[*].Instances[*].IamInstanceProfile.Arn" --output text \| grep wsi-bastion-role ₩ ; IAM_ROLE_NAME=$(aws ec2 describe-instances --instance-ids $INSTANCE_ID --query "Reservations[*].Instances[*].IamInstanceProfile.Arn" --output text | awk -F'/' '{print $NF}') ₩ ; aws iam list-attached-role-policies --role-name $IAM_ROLE_NAME --query "AttachedPolicies[*].PolicyName" --output text
	2-2-A (예상 출력) **정확히 일치** **순서 중요**	arn:aws:iam::xxxxxxxxxxxx:instance-profile/wsi-bastion-role PowerUserAccess IAMReadOnlyAccess 또는 IAMReadOnlyAccess PowerUserAccess

<table>
<tr><th>순번</th><th colspan="2">채점 항목</th></tr>
<tr><td rowspan="2">2-3</td><td>2-3A
(명령어
입력)</td><td>aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-bastion-ec2" "Name=tag:ec2,Values=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[].{FromPort:FromPort,ToPort:ToPort,IpRanges:IpRanges}"</td></tr>
<tr><td>2-3-A
(예상 출력)
<u>정확히 일치</u>
<u>순서 중요</u></td><td>“wsi-bastion-SG”
[
{
"FromPort": 2220, → 22외에 포트 출력 값은 상관없습니다.
"ToPort": 2220, → 22외에 포트 출력 값은 상관없습니다.
"IpRanges": [
{
"CidrIp": "0.0.0.0/0" → 본인 IP로도 사용할 수도 있음
}
]
}
]</td></tr>
</table>

순번	채점 항목	
2-4	2-4A (명령어 입력)	aws logs get-log-events --log-group-name wsi-bastion-user-logs --log-stream-name wsi-bastion-stream --limit 4 --query "events[*].{message:message}" --output text --start-from-head
	2-4-A (예상 출력) **정확히 일치** **순서 중요**	{ ec2-user, ec2-user has logged out!, 2024-05-27 16:41:56 } { ec2-user, ec2-user has logged in!, 2024-05-27 16:42:01 } { ec2-user, ec2-user has logged out!, 2024-05-27 16:49:57 } { ec2-user, ec2-user has logged in!, 2024-05-27 16:50:02 } **[logged in!와 logged out! 뜨는지 확인하고, 현재 날짜 기준으로 되어 있는지 확인]** **(로그가 안뜰 경우 Bastion 서버를 로그아웃, 로그인을 합니다.)**
3-1	3-1A (명령어 입력)	aws s3 ls | grep -E "wsi-cc-data-"
	3-1-A (예상 출력) **정확히 일치** **순서 중요**	2024-05-27 17:43:54 **wsi-cc-data-<선수번호>-<영문 4자리>**
3-2	3-2-A (명령어 입력)	BUCKET=$(aws s3api list-buckets --query "Buckets[?starts_with(Name, 'wsi-cc-data-')].Name" --ou tput text) ₩ ; aws s3 ls s3://$BUCKET/frontend/index.html
	3-2-A	2024-06-19 10:17:41 6785 index.html (6785가 출력되면 정답입니다.)

순번	채점 항목	
4-1	4-1-A (명령어 입력)	aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-control-plane" --query "Reservations[0].Instances[0].InstanceType" ₩ ; CONTROL_PLANE_SUBNET=$(aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-control-plane" --query "Reservations[*].Instances[*].SubnetId" --output text) ₩ ; aws ec2 describe-subnets --subnet-ids $CONTROL_PLANE_SUBNET --query "Subnets[*].Tags[?Key=='Name'].Value" --output text \| grep wsi-private-a ₩ ; INSTANCE_ID=$(aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-control-plane" --query "Reservations[*].Instances[*].InstanceId" --output text) ₩ ; aws ec2 describe-instances --instance-ids $INSTANCE_ID --query "Reservations[*].Instances[*].IamInstanceProfile.Arn" --output text | grep wsi-control-plane-role
	4-1-A (예상 출력) **정확히 일치** **순서 중요**	c5.large wsi-private-a arn:aws:iam::xxxxxxxxx:instance-profie/wsi-control-plane-role
4-2	4-2-A (명령어 입력)	# ssh ec2-user@<control plane instance IP 주소> -p 3817 접속 합니다. awk -F: '{print $1}' /etc/passwd \| grep '^user$' awk -F: '{print $1}' /etc/passwd | grep dev # exit
	4-2-A (예상 출력) **정확히 일치** **순서 중요**	user dev

순번	채점 항목	
4-3	4-3-A (명령어 입력)	aws ec2 describe-instances --filter "Name=tag:Name,Values=wsi-control-plane" --query "Reservations[0].Instances[0].SecurityGroups[0].GroupName" ₩ ; aws ec2 describe-security-groups --filter "Name=group-name,Values=wsi-control-plane-SG" --query "SecurityGroups[0].IpPermissions[].{FromPort:FromPort,ToPort:ToPort,IpRanges:IpRanges} "

	4-3-A
	(예상 출력)
	정확히 일치
	순서 중요
		"wsi-control-plane-SG"
		[
		{
		"FromPort": 3817,
		"ToPort": 3817,
		"IpRanges": [] → 값이 없는게 정상입니다.
		}
		]

4-4	4-4-A (명령어 입력)	# ssh user@<control plane instance IP 주소> -p 3817 접속 합니다. rm ~/.aws/credentials ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text) aws sts assume-role --role-arn arn:aws:iam::${ACCOUNT_ID}:role/user --role-session-name user-session > credential.json cat credential.json

	4-4-A
	(예상 출력)
		{
		"Credentials": {
		"AccessKeyId": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",
		"SecretAccessKey": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",
		"SessionToken": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",
		"Expiration": "2024-05-30T17:23:52+00:00"
		},
		"AssumedRoleUser": {
		"AssumedRoleId": "xxxxxxxxxxxxxxxxxxxxxxx:user-session",
		"Arn": "arn:aws:sts::xxxxxxxxxxxxxxxxx:assumed-role/user/user-session"
		}
		}

순번	채점 항목	
4-4	4-4-B (명령어 입력)	AWS_ACCESS_KEY_ID=$(jq -r '.Credentials.AccessKeyId' "$assume_info") AWS_SECRET_ACCESS_KEY=$(jq -r '.Credentials.SecretAccessKey' "$assume_info") AWS_SESSION_TOKEN=$(jq -r '.Credentials.SessionToken' "$assume_info") export AWS_ACCESS_KEY_ID export AWS_SECRET_ACCESS_KEY export AWS_SESSION_TOKEN aws sts get-caller-identity aws eks update-kubeconfig --name wsi-cluster --region ap-northeast-2 kubectl get all -n skills kubectl get all kubectl delete deployment/wsi-customer-deployment –n skills unset AWS_ACCESS_KEY_ID AWS_SECRET_ACCESS_KEY AWS_SESSION_TOKEN aws eks update-kubeconfig --name wsi-cluster --region ap-northeast-2

	4-4-B
	(예상 출력)
		{
		"UserId": "xxxxxxxxxxxxxxx:user-session",
		"Account": "xxxxxxxxxxxx",
		"Arn": "arn:aws:sts::xxxxxxxxx:assumed-role/user/user-session"
		}
		
		NAME READY STATUS RESTARTS AGE
		pod/xxxxxxxxxxx 1/1 Running 0 7h6m
		<1개이상 Pod 출력이 되면 정답입니다.>
		Error from server (Forbidden): pods is forbidden: User "user" cannot list resource "pods" in API group "" in the namespace "default"
		
		Error from server (Forbidden): pods is forbidden: User "user" cannot delete resource "deployments" in API group "" in the namespace "skills"

5-1	5-1-A (명령어 입력)	# Bastion으로 다시 진행합니다. aws ecr describe-repositories --query "repositories[*].repositoryName[]"
	5-1-A (예상 출력)	["wsi-customer-ecr", "wsi-order-ecr", "wsi-product-ecr"]

순번	채점 항목	
5-2	5-2-A (명령어 입력)	# Control Plane으로 접속합니다. kubectl exec -it $(kubectl get pod -l wsi=skills -n skills --no-headers -o custom-columns=":metadata.name" | grep wsi-customer | head -n 1) -n skills -- curl -X GET "localhost:8080/healthcheck"
	5-2-A (예상 출력) **정확히 일치** **순서 중요**	{"status":"ok"}
6-1	6-1-A (명령어 입력)	# Bastion으로 진행합니다. aws eks list-clusters --query "clusters[]" ₩ ; aws eks describe-cluster --name wsi-cluster --region ap-northeast-2 --output json | jq -r '.cluster.logging.clusterLogging' ₩ ; aws eks describe-cluster --name wsi-cluster --region ap-northeast-2 --output json | jq -r '.cluster.version' ₩ ; aws eks list-nodegroups --cluster-name wsi-cluster --region ap-northeast-2 --output json | jq -r '.nodegroups'

	6-1-A
	(예상 출력)
		[
		"wsi-cluster"
		]
		[
		{
		"types": [
		"api",
		"audit",
		"authenticator",
		"controllerManager",
		"scheduler"
		],
		"enabled": true
		}
		]
		"1.29"
		[
		"wsi-addon-ng",
		"wsi-app-ng"
		]

순번	채점 항목	
6-2	6-2-A (명령어 입력)	# Control Plane에 실행합니다. kubectl get pods -l wsi=skills -n skills --no-headers | wc -l
	6-2-A (예상 출력)	6개 이상 출력

순번	채점 항목	
6-3	6-3-A (명령어 입력)	# Control Plane에 계속 진행 kubectl exec -it $(kubectl get pod -l wsi=skills -n skills --no-headers -o custom-columns=":metadata.name" | grep wsi-customer | head -n 1) -n skills -- curl -X GET "http://meister.hrdkorea.or.kr/main/main.do" --max-time 10 kubectl exec -it $(kubectl get pod -l wsi=skills -n skills --no-headers -o custom-columns=":metadata.name" | grep wsi-customer | head -n 1) -n skills -- curl -X GET "http://wsi-customer-service.skills.svc.cluster.local/healthcheck" --max-time 10
	6-3-A (예상 출력) **정확히 일치** **순서 중요**	<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN"> <HTML><HEAD> <TITLE> 302 Found </TITLE> </HEAD><BODY> <H1>Found</H1> <HR> </BODY></HTML> ({"status":"ok."}가 뜨면 오답입니다.)
7-1	7-1-A (명령어 입력)	# Control Plane에 계속 진행 kubectl get deployment -n skills | grep wsi- | wc -l
	7-1-A (예상 출력) **정확히 일치** **순서 중요**	3
7-2	7-2-A (명령어 입력)	# Control Plane에 계속 진행 kubectl get service -n skills | grep wsi- | wc -l
	7-2-A (예상 출력) **정확히 일치** **순서 중요**	3

순번	채점 항목	

8-1	8-1-A
	(명령어
	입력)
		# Bastion으로 이동합니다.
		LB_ARN=$(aws elbv2 describe-load-balancers --names wsi-alb --query 'LoadBalancers[0].LoadBalancerArn' --output text) ₩
		; TARGET_GROUP_ARNS=$(aws elbv2 describe-target-groups --load-balancer-arn "$LB_ARN" --query 'TargetGroups[*].TargetGroupArn' --output text) ₩
		; for TG_ARN in $TARGET_GROUP_ARNS; do
		echo "Target Group ARN: $TG_ARN"
		aws elbv2 describe-target-group-attributes --target-group-arn "$TG_ARN" --query 'Attributes[?Key==`load_balancing.algorithm.type`].Value' --output text
		done ₩
		; aws elbv2 describe-load-balancers --names wsi-alb --query 'LoadBalancers[0].Scheme' --output text

	8-1-A (예상 출력) **정확히 일치** **순서 중요**	Target Group ARN: arn:aws:elasticloadbalancing:ap-northeast-2:xxxxxxxxxxxx least_outstanding_requests internet-facing
8-2	8-2-A (명령어 입력)	LB_DNS=$(aws elbv2 describe-load-balancers --names wsi-alb --query 'LoadBalancers[0].DNSName' --output text) ₩ ; curl http://${LB_DNS}/ --max-time 10
	8-2-A (예상 출력)	curl: (28) Connection timed out after 10002 milliseconds <Connection timed out 출력 된다면 정답입니다.>

순번	채점 항목	
8-3	8-3-A (명령어 입력)	LB_ARN=$(aws elbv2 describe-load-balancers --names wsi-alb --query 'LoadBalancers[0].LoadBalancerArn' --output text) ₩ ; TARGET_GROUP_ARNS=$(aws elbv2 describe-target-groups --load-balancer-arn "$LB_ARN" --query 'TargetGroups[*].TargetGroupArn' --output text) ₩ ; for TG_ARN in $TARGET_GROUP_ARNS; do echo "Target Group ARN: $TG_ARN" aws elbv2 describe-target-health --target-group-arn "$TG_ARN" --query 'TargetHealthDescriptions[*].{TargetId:Target.Id,State:TargetHealth.State}' --output text done
	8-3-A (예상 출력) **정확히 일치** **순서 중요**	Target Group ARN: arn:aws:elasticloadbalancing:ap-northeast-2:xxxxxxxx healthy 10.0.102.221 healthy 10.0.101.80 Target Group ARN: arn:aws:elasticloadbalancing:ap-northeast-2:xxxxxxxx healthy 10.0.102.10 healthy 10.0.103.99 Target Group ARN: arn:aws:elasticloadbalancing:ap-northeast-2:xxxxxxxx healthy 10.0.102.4 healthy 10.0.103.238 (3개가 정상입니다.)
9-1	9-1-A (명령어 입력)	aws rds describe-db-instances --query 'DBInstances[*].{DBInstanceIdentifier:DBInstanceIdentifier,DBInstanceClass:DBInstanceClass,Engine:Engine}' --output table

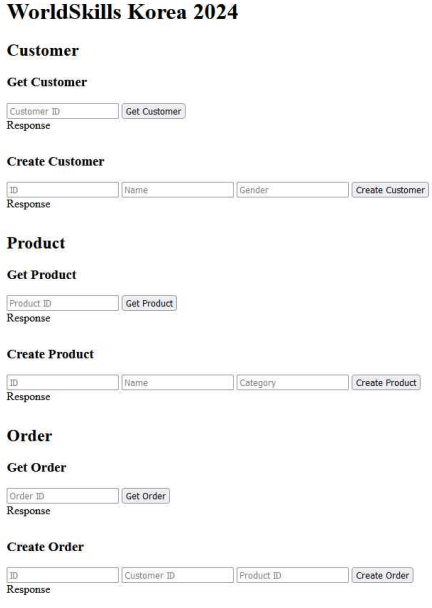
	9-1-A
	(예상 출력)
		[
		{
		"DBInstanceIdentifier": "wsi-rds-instance",
		"DBInstanceClass": "db.t3.micro",
		"Engine": "mysql",
		}
		]

순번	채점 항목	
10-1	10-1-A (명령어 입력)	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.DomainName"

	10-1-A
	(예상 출력)
	정확히 일치
	순서 중요
		[
		{
		"Key": "Name",
		"Value": "wsi-cdn"
		}
		]
		"xxxxxxxxxxxxxxxx.cloudfront.net"

순번	채점 항목	
10-2	10-2-A (명령어 입력)	curl https://<Cloudfront 배포 url>/preview?img=skills.png

	10-2-A
	(예상 출력)
	정확히 일치
	순서 중요
		{
		"uri": "xxxxxxxxxxxxxxxxxxxxxxxx/img/skills.png?xxxxxxxxxxxxxxxxxx"
		}
		3분 대기
		(링크 눌으면은 사진이 나옵니다. 3분 후에 사진이 없어져야 득점입니다.)

11-1	11-1-A (명령어 입력)	# cloudfront uri를 웹브라우저로 접속합니다. https://<CF 주소>/
	11-1-A (예상 출력) **정확히 일치** **순서 중요**	

순번	채점 항목	
11-2	11-2-A (명령어 입력)	Create Customer: 1, shawn, male Get Customer: 1 Create Product: 1, Iphone 15 Pro Max, Digital Get Product: 1 Create Order: 1, 987654321, 123456789 Get Product: 1
	11-2-A (예상 출력) **정확히 일치** **순서 중요**	**Customer** **Get Customer** 1 Get Customer { "customer": { "id": "1", "name": "shawn", "gender": "male" }, "message": "The customer is well in database." } **Create Customer** 1 shawn male Create Customer { "customer": { "id": "1", "name": "shawn", "gender": "male" }, "message": "The customer is created." } **Product** **Get Product** 1 Get Product { "product": { "id": "1", "name": "Iphone 15 Pro Max", "category": "Digital" }, "message": "The product is well in database." } **Create Product** 1 Iphone 15 Pro Max Digital Create Product { "product": { "id": "1", "name": "Iphone 15 Pro Max", "category": "Digital" }, "message": "The product is created." } **Order** **Get Order** 1 Get Order { "order": { "id": "1", "customerid": "987654321", "productid": "123456789" }, "message": "The order is well in database." } **Create Order** 1 987654321 123456789 Create Order { "order": { "id": "1", "customerid": "987654321", "productid": "123456789" }, "message": "The order is created." }