2023 지방기능경기대회 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) 채점 진행 전 환경 셋업을 위해 다음 사항을 확인해야 합니다.
   - Bastion에 SSH로 접근 가능한지 확인합니다.
   - Bastion에서 awscli, curl이 설치되었는지 확인합니다.
   - Bastion에서 IAM Role이 맵핑되어 awscli로 AWS 모든 리소스에 접근 가능한지 확인합니다.
   - aws sts get-caller-identity 명령을 통해 선수의 계정이 아닌 다른 계정에 접근하고 있는지 확인합니다. 만약, 다른 계정이라면 부정행위를 의심할 수 있습니다.
6) 문제지와 채점지에 있는 <> 는 변수입니다. 해당 부분을 변경해 입력합니다.
7) 채점은 문항 순서대로 진행해야 합니다.
8) 삭제된 내용은 되돌릴 수 없음으로 유의하여 채점을 진행합니다.
9) 이의신청까지 종료된 이후 선수가 생성한 클라우드 리소스를 삭제합니다.
10) 부분 점수가 있는 문항은 채점 항목에 부분 점수가 적혀져 있습니다.
11) 부분 점수가 따로 없는 문항은 전체 다 맞아야 점수로 인정됩니다.
12) 채점 전 채점환경 구성을 위해 ~/.aws/config에 아래 내용이 추가되도록 합니다.

```
/////
[default]
region = ap-northeast-2
output = json
/////
```

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	네트워킹	7.5		○		○	
	2	Bastion	3		○		○	
	3	ECR	6		○		○	
	4	ECS	16.5		○		○	
	5	ALB	7.5		○		○	
	6	Target Group	9		○		○	
	7	CloudFront	10.5		○		○	
합 계			60					

2) 채점방법 및 기준

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
제1과제	1	네트워킹	1	VPC	1.5
			2	Subnet	1.5
			3	IGW	1.5
			4	NAT	1.5
			5	HA	1.5
	2	Bastion	1	EC2 Instance	1.5
			2	EIP	1.5
	3	ECR	1	ECR Creation	3
			2	ECR Scan	3
	4	ECS	1	ECS Cluster	1.5
			2	ECS Task Definition	4.5
			3	ECS Service	3
			4	ECS Task	1.5
			5	ECS HA	3
			6	ECS Security	3
	5	ALB	1	ALB Creation	3
			2	ALB Listener	3
			3	ALB Security	1.5
	6	Target Group	1	Target Group Creation	3
			2	Target Group Healty	3
			3	Target Group HA	3
	7	CloudFront	1	CloudFront Creation	3
			2	CloudFront Connect	3
			3	CloudFront Cache	3
			4	CloudFront Edge	1.5
	총점				60

3) 채점 내용

순번	채점 항목
1-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "10.0.0.0/16"이 출력되는지 확인합니다. aws ec2 describe-vpcs --filter Name=tag:Name,Values=wsi-vpc --query "Vpcs[].CidrBlock"
1-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력 후 "10.0.1.0/24"이 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-public-a --query ₩ "Subnets[].CidrBlock" 3) 아래 명령어를 입력 후 "10.0.2.0/24"이 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-public-b --query ₩ "Subnets[].CidrBlock" 4) 아래 명령어를 입력 후 "10.0.3.0/24"이 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-private-a --query ₩ "Subnets[].CidrBlock" 5) 아래 명령어를 입력 후 "10.0.4.0/24"이 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-private-b --query ₩ "Subnets[].CidrBlock"
1-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "igw-" 로 시작하는 문구가 출력되는지 확인합니다. aws ec2 describe-internet-gateways --filter Name=tag:Name,Values=wsi-igw ₩ --query "InternetGateways[].InternetGatewayId" 3) 아래 명령어 입력 후 "igw-" 로 시작하는 문구가 출력되는지 확인합니다. aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-public-rtb ₩ --query "RouteTables[].Routes[].GatewayId"
1-4	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "nat-" 로 시작하는 문구가 출력되는지 확인합니다. aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-private-rtb-a ₩ --query "RouteTables[].Routes[].NatGatewayId" 3) 아래 명령어 입력 후 "nat-" 로 시작하는 문구가 출력되는지 확인하고 2)의 결과와 다른 ID를 갖는지 확인합니다. aws ec2 describe-route-tables --filter Name=tag:Name,Values=wsi-private-rtb-b ₩ --query "RouteTables[].Routes[].NatGatewayId"

순번	채점 항목
1-5	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "ap-northeast-2a"가 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-public-a --query ₩ "Subnets[].AvailabilityZone" 3) 아래 명령어 입력 후 "ap-northeast-2b"가 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-public-b --query ₩ "Subnets[].AvailabilityZone" 4) 아래 명령어 입력 후 "ap-northeast-2a"가 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-private-a --query ₩ "Subnets[].AvailabilityZone" 5) 아래 명령어 입력 후 "ap-northeast-2b"가 출력되는지 확인합니다. aws ec2 describe-subnets --filter Name=tag:Name,Values=wsi-private-b --query ₩ "Subnets[].AvailabilityZone"
2-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력하여 "i-"로 시작하는 문구를 받아오는지 확인합니다. aws ec2 describe-instances --filter Name=tag:Name,Values=wsi-bastion ₩ --query "Reservations[].Instances[].InstanceId"
2-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력해 나오는 IP를 기록합니다. aws ec2 describe-instances --filter Name=tag:Name,Values=wsi-bastion ₩ --query "Reservations[].Instances[].PublicIpAddress" 3) 아래 명령어 입력 후 출력되는 IP 리스트 중에 2)번에서 출력된 IP가 있는지 확인합니다. aws ec2 describe-addresses --query "Addresses[].PublicIp"
3-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "latest"가 출력되는지 확인합니다. (1.5점) aws ecr describe-images --repository-name wsi-about --query "imageDetails[].imageTags[]" 3) 아래 명령어 입력 후 "latest"가 출력되는지 확인합니다. (1.5점) aws ecr describe-images --repository-name wsi-projects --query "imageDetails[].imageTags[]"

순번	채점 항목
3-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "CRITICAL", "HIGH", "MEDIUM", "LOW" 중 어떤 것도 출력되지 않는지 확인합니다. (1.5점) aws ecr describe-image-scan-findings --repository-name wsi-about ₩ --image-id imageTag=latest --query "imageScanFindings.findingSeverityCounts" 3) 아래 명령어 입력 후 "CRITICAL", "HIGH", "MEDIUM", "LOW" 중 어떤 것도 출력되지 않는지 확인합니다. (1.5점) aws ecr describe-image-scan-findings --repository-name wsi-projects ₩ --image-id imageTag=latest --query "imageScanFindings.findingSeverityCounts"
4-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-ecs"가 출력되는지 확인합니다. aws ecs describe-clusters --cluster wsi-ecs --query "clusters[].clusterName" 3) 아래 명령어 입력 후 "FARGATE", "FARGATE_SPOT"이 출력되는지 확인합니다. aws ecs describe-clusters --cluster wsi-ecs --query "clusters[].capacityProviders[]"
4-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 5000이 출력되는지 확인합니다. (1점) aws ecs describe-task-definition --task-definition about-task-def ₩ --query "taskDefinition.containerDefinitions[].portMappings[].containerPort" 3) 아래 명령어 입력 후 "[계정 ID].dkr.ecr.ap-northeast-2.amazonaws.com/wsi-about:latest"가 출력되는지 확인합니다. (1.25점) aws ecs describe-task-definition --task-definition about-task-def ₩ --query "taskDefinition.containerDefinitions[].image" 4) 아래 명령어 입력 후 5000이 출력되는지 확인합니다. (1점) aws ecs describe-task-definition --task-definition projects-task-def ₩ --query "taskDefinition.containerDefinitions[].portMappings[].containerPort" 5) 아래 명령어 입력 후 "[계정 ID].dkr.ecr.ap-northeast-2.amazonaws.com/wsi-projects:latest"가 출력되는지 확인합니다. (1.25점) aws ecs describe-task-definition --task-definition projects-task-def ₩ --query "taskDefinition.containerDefinitions[].image"

순번	채점 항목
4-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "DISABLED"가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-about-svc ₩ --query "services[].networkConfiguration.awsvpcConfiguration.assignPublicIp" 3) 아래 명령어 입력 후 "FARGATE"가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-about-svc ₩ --query "services[].launchType" 4) 아래 명령어 입력 후 "ACTIVE"가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-about-svc --query "services[].status" 5) 아래 명령어 입력 후 "DISABLED"가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-projects-svc ₩ --query "services[].networkConfiguration.awsvpcConfiguration.assignPublicIp" 6) 아래 명령어 입력 후 "FARGATE"가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-projects-svc ₩ --query "services[].launchType" 7) 아래 명령어 입력 후 "ACTIVE"가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-projects-svc --query "services[].status"
4-4	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 2 이상의 숫자가 출력되는지 확인합니다. aws ecs describe-clusters --cluster wsi-ecs --query "clusters[].runningTasksCount"

순번	채점 항목
4-5	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 4 이상의 숫자가 출력되는지 확인합니다. (1점) aws ecs describe-clusters --cluster wsi-ecs --query "clusters[].runningTasksCount" 3) 아래 명령어 입력 후 "subnet-"로 시작하는 문자열이 2개 이상 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-about-svc ₩ --query "services[].networkConfiguration.awsvpcConfiguration[].subnets[]" 4) 아래 명령어 입력 후 2 이상의 숫자가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-about-svc ₩ --query "services[].runningCount" 5) 아래 명령어 입력 후 "subnet-"로 시작하는 문자열이 2개 이상 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-projects-svc ₩ --query "services[].networkConfiguration.awsvpcConfiguration[].subnets[]" 6) 아래 명령어 입력 후 2 이상의 숫자가 출력되는지 확인합니다. (0.5점) aws ecs describe-services --cluster wsi-ecs --services wsi-projects-svc ₩ --query "services[].runningCount"
4-6	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 출력되는 문자열을 기록합니다. aws elbv2 describe-load-balancers --query "LoadBalancers[].SecurityGroups[]" 3) 아래 명령어 입력후 2)에서 기록한 문자열과 일치하는지 확인합니다. (1.5점) aws ec2 describe-security-groups ₩ --query "SecurityGroups[?GroupName=='wsi-about-sg'].IpPermissions[].UserIdGroupPairs[].GroupId" 4) 아래 명령어 입력후 2)에서 기록한 문자열과 일치하는지 확인합니다. (1.5점) aws ec2 describe-security-groups --query ₩ "SecurityGroups[?GroupName=='wsi-projects-sg'].IpPermissions[].UserIdGroupPairs[].GroupId"

순번	채점 항목
5-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-alb"가 출력되는지 확인합니다. (0.5점) aws elbv2 describe-load-balancers ₩ --query "LoadBalancers[?LoadBalancerName=='wsi-alb'].LoadBalancerName" 3) 아래 명령어 입력 후 "internet-facing"이 출력되는지 확인합니다. (0.5점) aws elbv2 describe-load-balancers ₩ --query "LoadBalancers[?LoadBalancerName=='wsi-alb'].Scheme" 4) 아래 명령어 입력 후 "application"이 출력되는지 확인합니다. (0.5점) aws elbv2 describe-load-balancers ₩ --query "LoadBalancers[?LoadBalancerName=='wsi-alb'].Type" 5) 아래 명령어 입력 후 "ap-northeast-2a", "ap-northeast-2b"가 출력되는지 확인합니다. (1.5점) aws elbv2 describe-load-balancers ₩ --query "LoadBalancers[?LoadBalancerName=='wsi-alb'].AvailabilityZones[].ZoneName"
5-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. alb_arn=`aws elbv2 describe-load-balancers ₩ --query "LoadBalancers[?LoadBalancerName=='wsi-alb'].LoadBalancerArn" --output text` 3) 아래 명령어 입력 후 "HTTP"가 출력되는지 확인합니다. (1.5점) aws elbv2 describe-listeners --load-balancer-arn $alb_arn --query "Listeners[].Protocol" 4) 아래 명령어 입력 후 "80"이 출력되는지 확인합니다. (1.5점) aws elbv2 describe-listeners --load-balancer-arn $alb_arn --query "Listeners[].Port"
5-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. alb_dns=`aws elbv2 describe-load-balancers ₩ --query "LoadBalancers[?LoadBalancerName=='wsi-alb'].DNSName" --output text` 3) 아래 명령어 입력 후 "403 Forbidden"이 출력되는지 확인합니다. curl $alb_dns

순번	채점 항목
6-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "/about"이 출력되는지 확인합니다. (1.5점) aws elbv2 describe-target-groups --names wsi-about-tg ₩ --query "TargetGroups[].HealthCheckPath" 3) 아래 명령어 입력 후 "/projects"가 출력되는지 확인합니다. (1.5점) aws elbv2 describe-target-groups --names wsi-projects-tg ₩ --query "TargetGroups[].HealthCheckPath"
6-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. about_tg_arn=`aws elbv2 describe-target-groups --names wsi-about-tg ₩ --query "TargetGroups[].TargetGroupArn" --output text` projects_tg_arn=`aws elbv2 describe-target-groups --names wsi-projects-tg ₩ --query "TargetGroups[].TargetGroupArn" --output text` 3) 아래 명령어 입력 후 "healty"가 출력되는지 확인합니다. (1.5점) aws elbv2 describe-target-health --target-group-arn $about_tg_arn ₩ --query "TargetHealthDescriptions[].TargetHealth.State" 4) 아래 명령어 입력 후 "healty"가 출력되는지 확인합니다. (1.5점) aws elbv2 describe-target-health --target-group-arn $projects_tg_arn ₩ --query "TargetHealthDescriptions[].TargetHealth.State"
6-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. about_tg_arn=`aws elbv2 describe-target-groups --names wsi-about-tg ₩ --query "TargetGroups[].TargetGroupArn" --output text` projects_tg_arn=`aws elbv2 describe-target-groups --names wsi-projects-tg ₩ --query "TargetGroups[].TargetGroupArn" --output text` 3) 아래 명령어 입력 후 "ap-northeast-2a", "ap-northeast-2b"가 출력되는지 확인합니다. (1.5점) aws elbv2 describe-target-health --target-group-arn $about_tg_arn ₩ --query "TargetHealthDescriptions[].Target.AvailabilityZone" 4) 아래 명령어 입력 후 "ap-northeast-2a", "ap-northeast-2b"가 출력되는지 확인합니다. (1.5점) aws elbv2 describe-target-health --target-group-arn $projects_tg_arn ₩ --query "TargetHealthDescriptions[].Target.AvailabilityZone"

순번	채점 항목
7-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 약 14자리의 영문과 숫자가 섞인 ID가 출력되는지 확인합니다. (1.5점) aws cloudfront list-distributions --query "DistributionList.Items[].Id" 3) 아래 명령어 입력 후 "wsi-alb"로 시작하는 문자열이 출력되는지 확인합니다. (1.5점) aws cloudfront list-distributions --query "DistributionList.Items[].Origins.Items[].DomainName"
7-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. cloudfront_url=`aws cloudfront list-distributions --query "DistributionList.Items[].DomainName" ₩ --output text` 3) 아래 명령어 입력 후 "The about page"가 출력되는지 확인합니다. (1.5점) curl $cloudfront_url/about 4) 아래 명령어 입력 후 "The projects page"가 출력되는지 확인합니다. (1.5점) curl $cloudfront_url/projects
7-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "X-Cache: Hit from cloudfront"가 출력되는지 확인합니다. (1.5점) curl -I -s $cloudfront_url/about | grep X-Cache 3) 아래 명령어 입력 후 "X-Cache: Miss from cloudfront"가 출력되는지 확인합니다. (1.5점) curl -I -s $cloudfront_url/projects | grep X-Cache
7-4	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. cloudfront_id=`aws cloudfront list-distributions --query "DistributionList.Items[].Id" --output text` 3) 아래 명령어 입력 후 "PriceClass_All"이 출력되는지 확인합니다. aws cloudfront get-distribution-config --id $cloudfront_id --query "DistributionConfig.PriceClass"