

전국기능경기대회 채점기준

1. 채점 시 유의사항

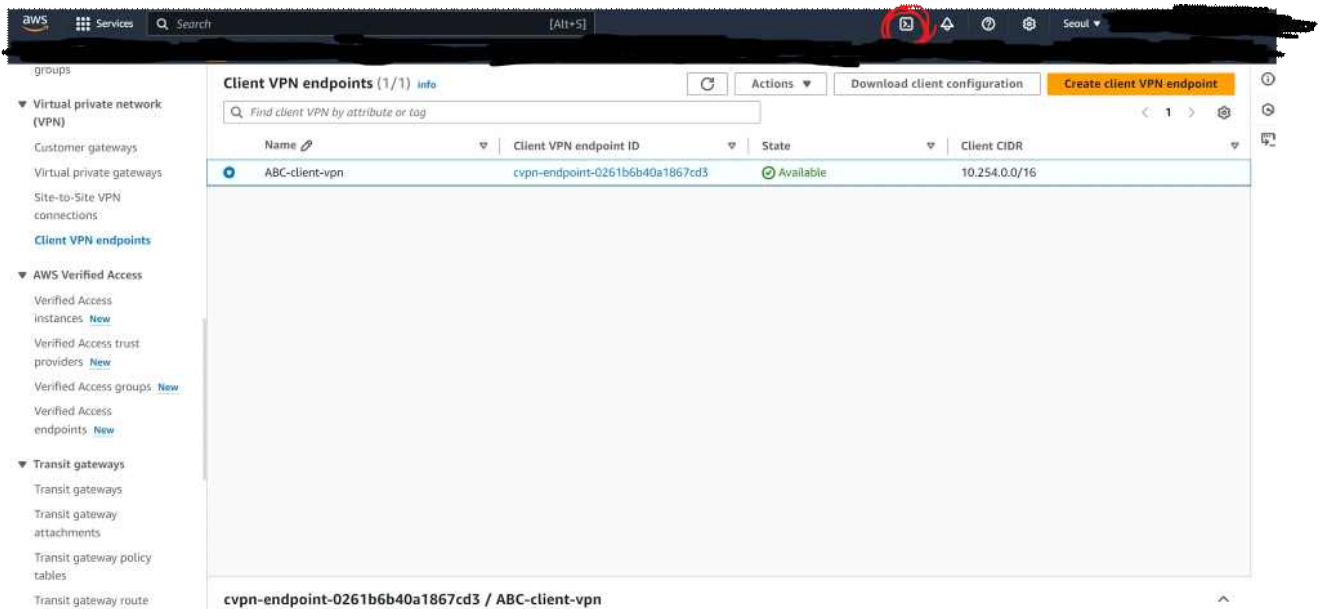
직 종 명

클라우드컴퓨팅

※ 다음 사항을 유의하여 채점하시오.

채점하는 서버가 선수의 서버가 맞는지 확인합니다.

채점시 명령어 입력은 아래 사진과 같이 CloudShell을 이용할 수 있습니다.



※ 채점기준 양식은 반드시, 양식에 맞추어 작성해야 합니다.(채점사이트 입력에 필요)

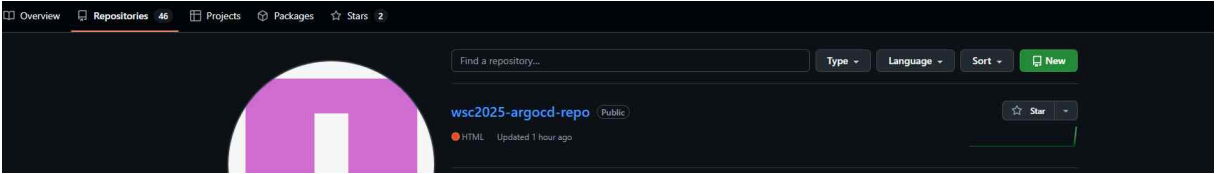
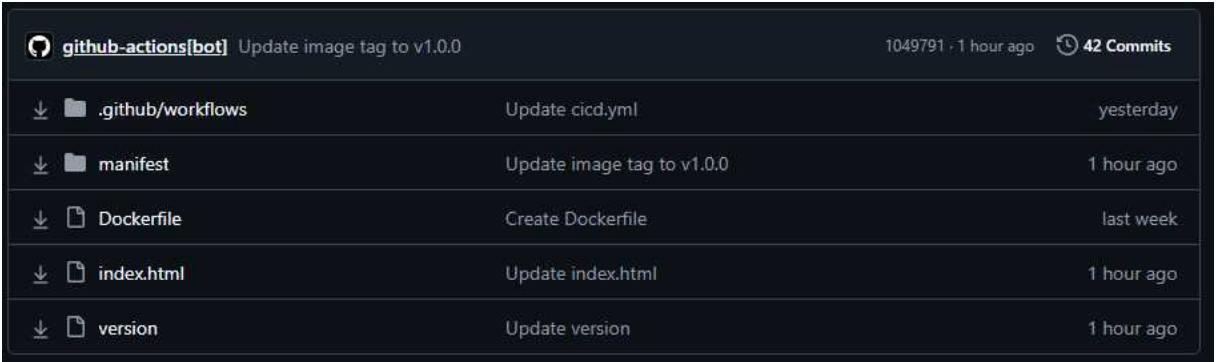
2. 채점기준표


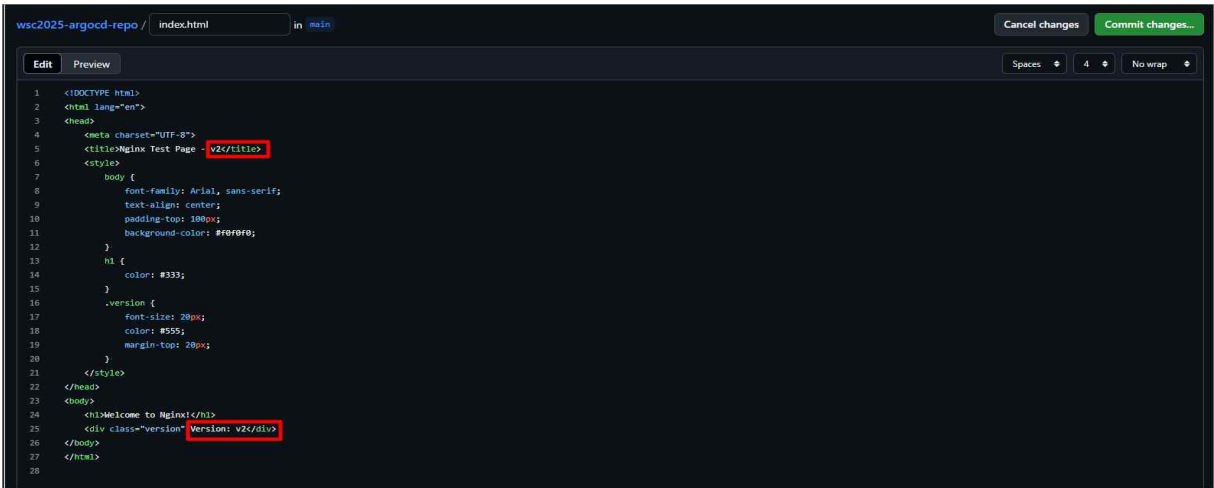
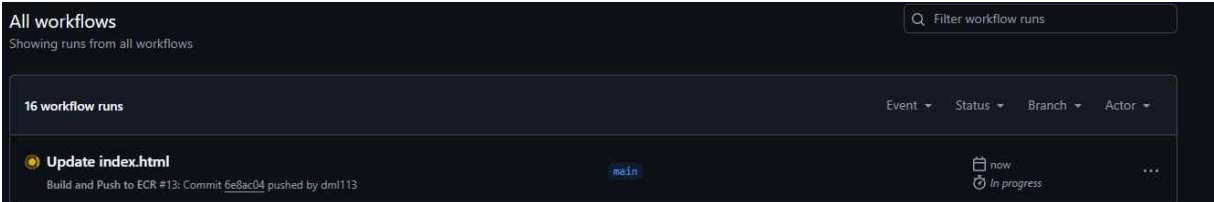
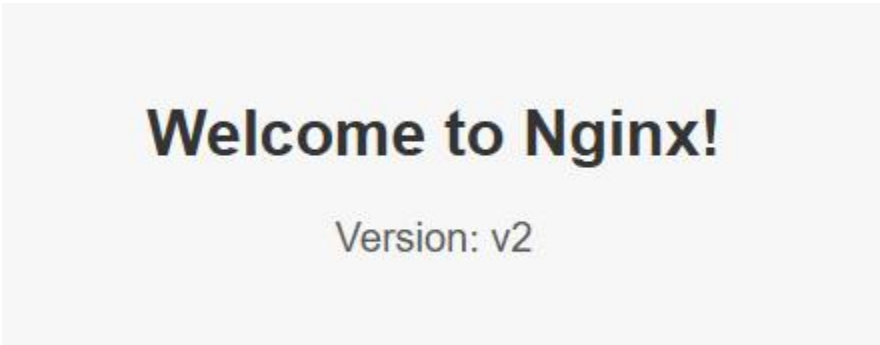
1) 주요항목별 배점			직 종 명		클라우드컴퓨팅			
과제 번호	일련 번호	주요항목	배점	채점방법		채점시기		비고
				독립	합의	경기 진행중	경기 종료후	
제2과제	1	CI/CD	7.5	○			○	
	2	Storage data protect	7.5	○			○	
	3	WAF	7.5	○			○	
	4	Secure networking	7.5	○			○	
합 계			30					

2) 채점방법 및 기준

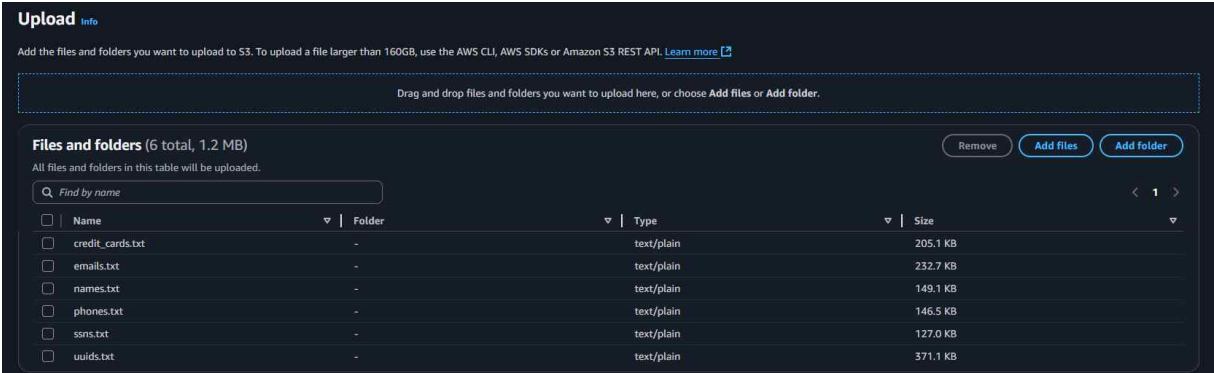
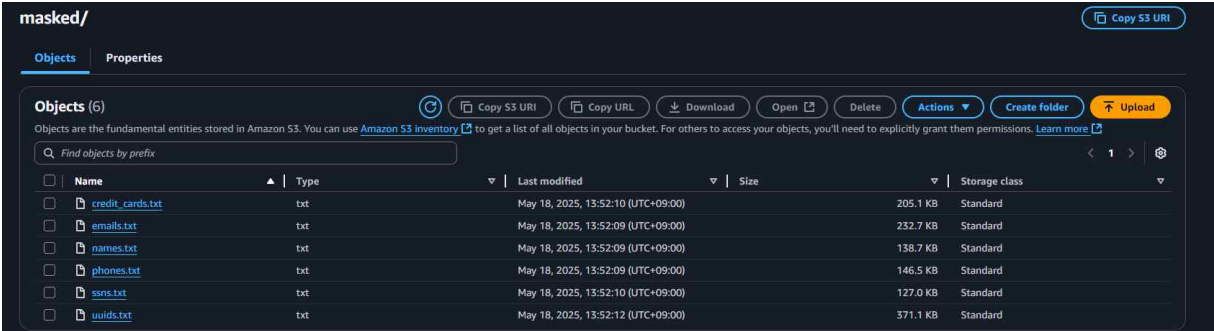
과제 번호	일 련 번 호	주요항목	일련 번호	세부항목(채점방법)	배점
제2과제	1	CI/CD	1	ECR Configure	0.5
			2	ECR Image Tag	0.5
			3	EKS Cluster Configure	0.5
			4	EKS Node Configure	0.5
			5	Pod Create	0.5
			6	ELB Create	0.5
			7	ELB Endpoint Check	0.5
			8	Github Repository Configure	0.5
			9	Github Action Cofigure	1.5
			10	ArgoCD Deploy Test	1.5
			11	ArgoCD SSOT	0.5
	2	Storage data protect	1	S3 Create	0.5
			2	Macie Job Create	0.5
			3	Lambda Create	0.5
			4	Lambda Trigger	1.5
			5	File masking check	1.5
			6	Macie Sensor Test	1.5
			7	Masking Test	1.5
	3	WAF	1	App Server Configure	1.5
			2	ELB Configure	1.5
			3	WAF Configure	1.5
			4	login API Attach	1.5
			5	lookup API Attach	1.5
	4	Secure networking	1	VPC Create	0.5
			2	Subnet Create	0.5
			3	Bastion Create	0.5
			4	Network Firewall Configure	0.5
			5	Network Firewall Stateless rule	1.25
			6	Network Firewall Stateful rule	1.25
			7	Bastion Ping Test	1.5
			8	Bastion DNS Query Test	1.5
합계					28.5

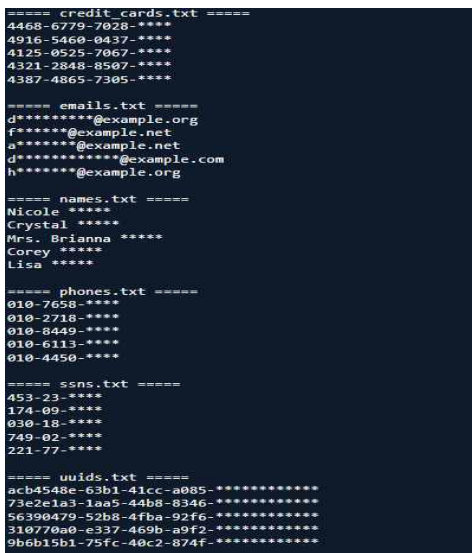
순번	채점 항목
1-0	1) 웹브라우저로 선수의 AWS 계정에 로그인 후 ap-southeast-1 Region으로 접속합니다.
1-1	1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다. 2) 아래 명령어를 입력 합니다. \$ aws ecr describe-repositories --repository-names app-repo --query 'repositories[0].repositoryName' --output text 3) 출력되는 결과가 "app-repo" 인지 확인 합니다.
1-2	1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다. 2) 아래 명령어를 입력 합니다. \$ aws ecr list-images --repository-name app-repo --query 'imageIds[?imageTag!=`null`].imageTag' --output text 3) 출력되는 결과가 "vX.X.X" 인지 확인 합니다. (ex. v1.0.0)
1-3	1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다. 2) 아래 명령어를 입력 합니다. \$ aws eks describe-cluster --name wsc2025-cluster --query 'cluster.[name, version]' --output text 3) 출력되는 결과가 "wsc2025-cluster 1.32" 인지 확인 합니다.
1-4	1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다. 2) 아래 명령어를 입력 합니다. \$ aws eks describe-nodegroup --cluster-name wsc2025-cluster --nodegroup-name wsc2025-app-ng W --query 'nodegroup.[nodegroupName, instanceTypes]' --output text \$ aws ec2 describe-instances W --filters "Name=tag:eks:cluster-name,Values=wsc2025-cluster" W --query 'Reservations[*].Instances[*].[InstanceId,InstanceType,Tags[?Key==`eks:wsc2025-app-ng`].Value [0]]' W --output text 3) 출력되는 결과가 "wsc2025-app-ng", "t3.medium"인지 확인합니다.
1-5	1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다. 2) 아래 명령어를 입력 합니다. \$ kubectl get deploy -n wsc2025 3) 출력되는 결과가 출력되고, READY 상태인지 확인합니다.
1-6	1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다. 2) 아래 명령어를 입력 합니다. \$ aws elbv2 describe-load-balancers W --query "LoadBalancers[?LoadBalancerName=='wsc2025-cicd-alb'].LoadBalancerName" W --output text 3) 출력되는 결과가 "wsc2025-cicd-alb"인지 확인합니다.

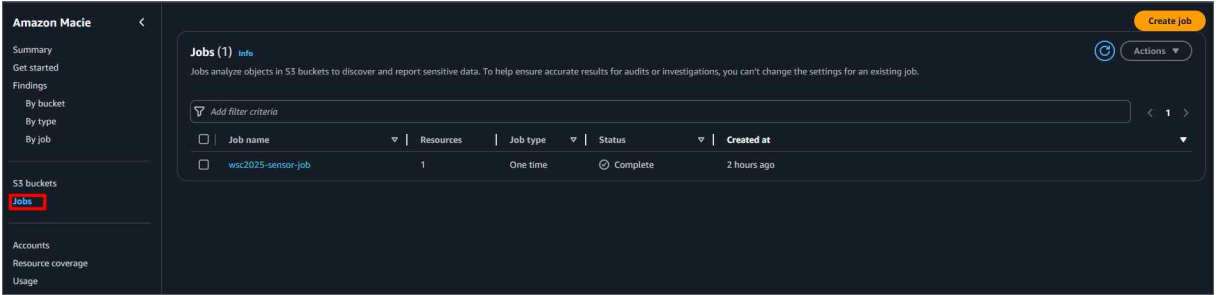
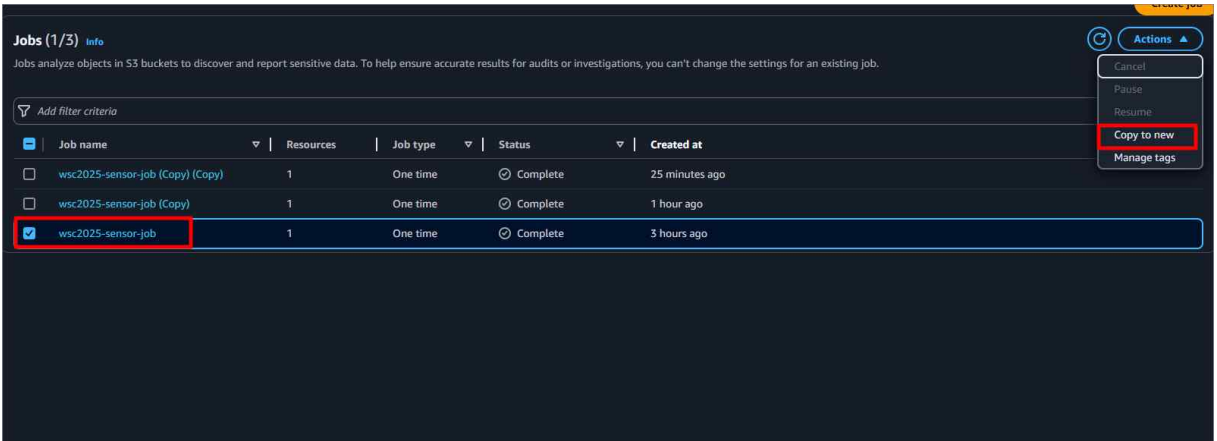
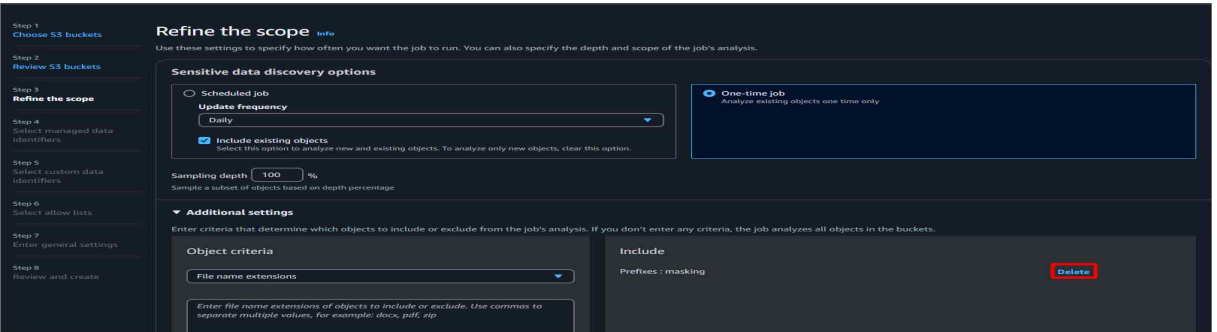
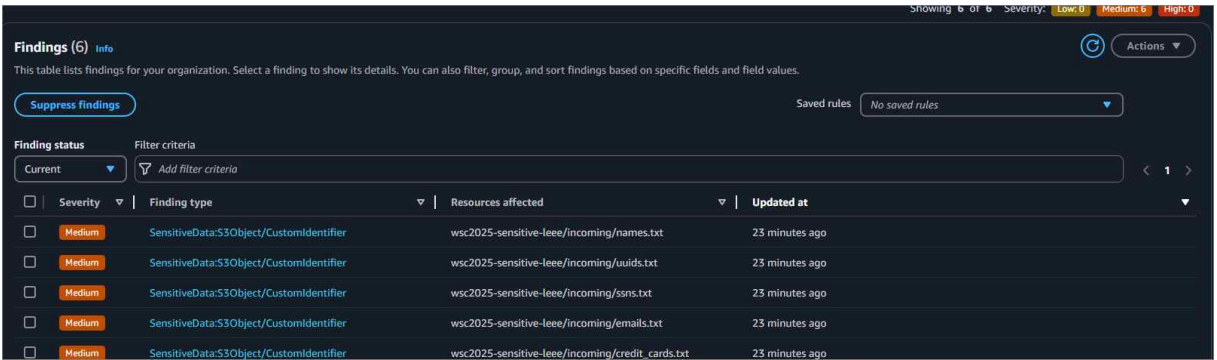
순번	채점 항목
1-7	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다.</p> <p>2) 아래 명령어를 입력 합니다.</p> <pre>\$ ENDPOINT=\$(aws elbv2 describe-load-balancers --query "LoadBalancers[?LoadBalancerName=='wsc2025-cicd-alb'].DNSName" --output text)</pre> <pre>\$ curl http://\$ENDPOINT/</pre> <p>3) 출력되는 결과에 “Version: v1” 이 포함되어 있는지 확인합니다.</p>
1-8	<p>1) 선수의 Github에 접근하여 wsc2025-argocd-repo라는 Repository가 생성되어 있고, Public 형태인지 확인합니다.</p>  <p>2) 아래 그림처럼 최상위 폴더에 index.html과 version 파일이 존재하는지 확인하며, manifest 폴더가 존재하는지 확인합니다.</p> 

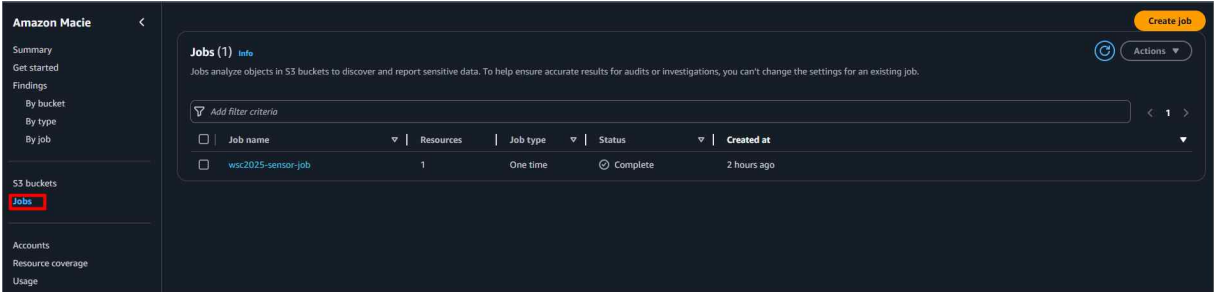
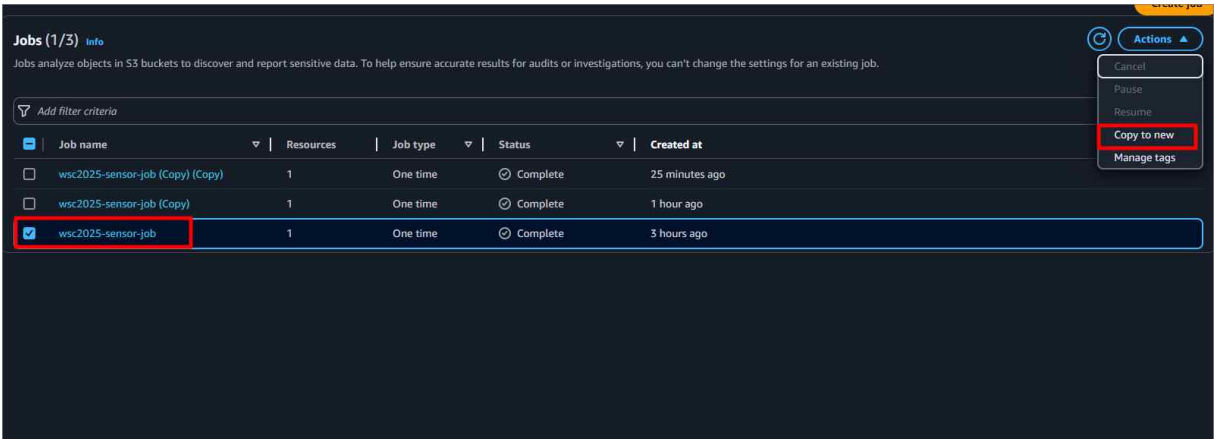
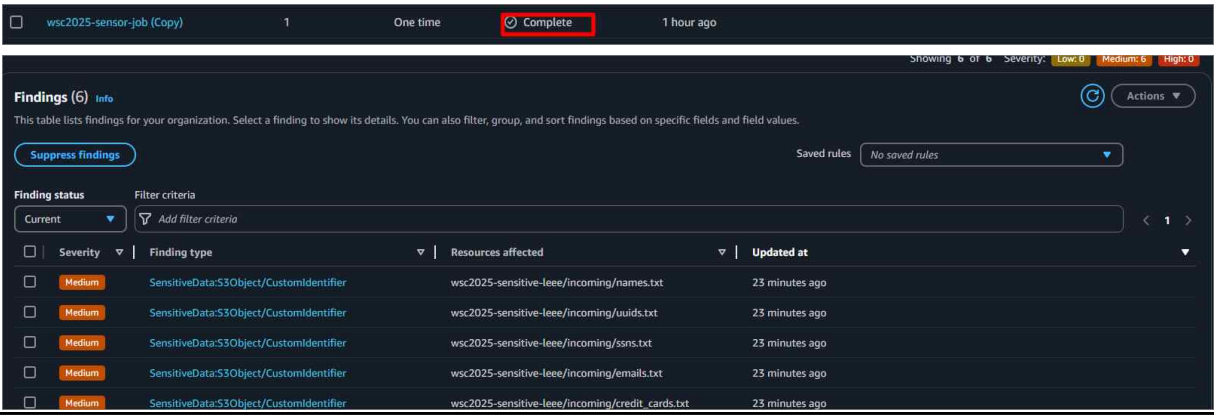
순번	채점 항목
1-9	<p>1) Github Repository에 접근합니다.</p> <p>2) Github Repository에 존재하는 version에 내용을 v1.0.1로 변경합니다.</p>  <p>3) index.html의 v1 부분을 전부 v2로 변경합니다.</p>  <p>4) Github Action에서 workflow가 동작하는 것을 확인</p> 
1-10	<p>1) 1-9 항목 채점이 끝나고, 5분 뒤에 wsc2025-cicd-alb라는 이름을 가진 LoadBalancer에 접근하여, Version: v2로 변경되었는지 확인</p> 

순번	채점 항목
1-11	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다.</p> <p>2) 아래 명령어를 입력합니다.</p> <pre>\$ kubectl set image deployment/nginx-deploy nginx=nginx -n wsc2025</pre> <pre>\$ kubectl get deployment nginx-deploy -o jsonpath='{.spec.template.spec.containers[0].image}' -n wsc2025</pre> <p>3) app-repo ECR에 존재하는 Image가 출력 되는지 확인합니다.</p>
2-0	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 ap-northeast-2 Region으로 접속합니다.</p>
2-1	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다.</p> <p>2) 아래 명령어를 입력합니다.</p> <pre>\$ aws s3api list-buckets --query "Buckets[?starts_with(Name, 'wsc2025-sensitive-')].Name" --output text</pre> <p>3) wsc2025-sensitive-<임의의 4자리 영문>가 출력되는지 확인합니다.</p>
2-2	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다.</p> <p>2) 아래 명령어를 입력합니다.</p> <pre>\$ aws macie2 list-classification-jobs W --query "items[?name=='wsc2025-sensor-job'].name" W --output text</pre> <p>3) wsc2025-sensor-job가 출력 되는지 확인합니다.</p>
2-3	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다.</p> <p>2) 아래 명령어를 입력합니다.</p> <pre>\$ aws lambda list-functions W --query "Functions[?FunctionName=='wsc2025-masking-start'].FunctionName" W --output text</pre> <p>3) wsc2025-masking-start가 출력 되는지 확인합니다</p>

순번	채점 항목
2-4	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 S3에 접속합니다.</p> <p>2) 선수가 생성한 wsc2025-sensitive-<임의의 4개의 영문>를 접근합니다.</p> <p>3) incoming prefix에 배포파일을 재업로드합니다.</p>  <p>4) 최대 1분 정도 기다릴 수 있으며, masked prefix에 incoming prefix에 업로드한 파일들이 똑같이 생성되었는지 확인</p> 

순번	채점 항목
2-5	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 CloudShell을 실행합니다.</p> <p>2) 아래 명령어를 입력합니다.</p> <pre>\$ aws s3 cp s3://wsc2025-sensitive-<임의의 4자리 영문>/masked/ ./masked/ --recursive</pre> <pre>\$ cd masked/</pre> <p>3) 아래 명령어를 입력하여 파일들이 마스킹이 되었는지 확인합니다.</p> <pre>\$ for f in *; do echo "==== \$f =====" head -n 5 "\$f" echo done</pre> <p>예)</p>  <pre>==== credit_cards.txt ==== 4468-6779-7028-**** 4916-5460-0437-**** 4125-9525-7067-**** 4321-2848-8507-**** 4387-4865-7305-**** ==== emails.txt ==== d*****@example.org f*****@example.net a*****@example.net j*****@example.com h*****@example.org ==== names.txt ==== Nicole ***** Crystal ***** Mrs. Brianna ***** Corey ***** Lisa ***** ==== phones.txt ==== 010-7658-**** 010-2718-**** 010-8449-**** 010-6113-**** 010-4450-**** ==== sns.txt ==== 453-23-**** 174-09-**** 030-18-**** 749-02-**** 221-77-**** ==== uuids.txt ==== acb4548e-63b1-41cc-a085-***** 73e2e1a3-1aa5-44b8-8346-***** 56390479-52b8-4fba-92f6-***** 310770a0-e337-469b-a9f2-***** 9b6b15b1-75fc-40c2-874f-*****</pre>

순번	채점 항목
	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 AWS Macie Service에 접근합니다.</p>  <p>2) wsc2025-sensor-job을 복사 후 scope의 additional settings의 모든 것을 제거한 후 다시 생성합니다. (job의 name은 선수 임의로 생성합니다.)</p>  <p>2-6</p>  <p>3) 생성한 Job의 status가 Complete 상태인지 확인 후 Finding에서 아래와 같이 6개의 파일들이 전부 위험 상태로 존재하는지 확인</p> 

순번	채점 항목
2-7	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 AWS Macie Service에 접근합니다.</p>  <p>2) wsc2025-sensor-job을 복사 후 변경사항 없이 Next 클릭하여 생성합니다. (job의 name은 선수 임의로 생성합니다.)</p>  <p>3) 복사한 Job의 status가 Complete 상태인지 확인 후 Finding에서 감지되지 않았는지 확인합니다. (Findings의 개수가 늘어나지 않아야 합니다.)</p> 
3-0	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 us-east-1 Region으로 접속합니다.</p>

순번	채점 항목
3-1	<p>1) Default VPC에 배포된 Bastion의 shell에 접근합니다.</p> <p>2) 아래 명령어를 입력해 “wsc2025-app-server” 이 출력 되는지 확인합니다.</p> <pre>\$ aws ec2 describe-instances W --filters "Name=tag:Name,Values=wsc2025-app-server" W --query "Reservations[*].Instances[*].Tags[?Key=='Name'].Value[]" W --output text</pre> <p>3) 아래 명령어를 입력해 403 Error 코드가 뜨지 않는지 확인합니다.</p> <pre>\$ IP=\$(aws ec2 describe-instances W --filters "Name=tag:Name,Values=wsc2025-app-server" W --query "Reservations[*].Instances[*].PublicIpAddress" W --output text) \$ curl "http://\$IP:5000" \$ curl "http://\$IP:5000/login?name=admin'--&secret=anything" \$ curl "http://\$IP:5000/lookup?id=id%3D1+OR+1%3D1"</pre>
3-2	<p>1) Default VPC에 배포된 Bastion의 shell에 접근합니다.</p> <p>2) 아래 명령어를 입력해 wsc2025-waf-alb, application, internet-facing가 출력 되는지 확인합니다.</p> <pre>\$ aws elbv2 describe-load-balancers W --names wsc2025-waf-alb W --query "LoadBalancers[*].[LoadBalancerName, Type, Scheme]" W --output text</pre>
3-3	<p>1) Default VPC에 배포된 Bastion의 shell에 접근합니다.</p> <p>2) 아래 명령어를 입력해 wsc2025-waf가 출력되는지 확인합니다.</p> <pre>\$ aws wafv2 list-web-acls W --scope REGIONAL W --query "WebACLs[?Name=='wsc2025-waf'].Name" W --output text</pre>
3-4	<p>1) Default VPC에 배포된 Bastion의 shell에 접근합니다.</p> <p>2) 아래 명령어를 입력했을 때 403 Error 코드가 출력 되는지 확인합니다.</p> <pre>\$ export ALB_ENDPOINT=\$(aws elbv2 describe-load-balancers W --names wsc2025-waf-alb W --query "LoadBalancers[*].DNSName" W --output text) \$ curl "http://\$ALB_ENDPOINT/login?name=admin'--&secret=anything"</pre>

순번	채점 항목
3-5	<p>1) Default VPC에 배포된 Bastion의 shell에 접근합니다.</p> <p>2) 아래 명령어를 입력했을 때 403 Error 코드가 출력 되는지 확인합니다.</p> <pre>\$ export ALB_ENDPOINT=\$(aws elbv2 describe-load-balancers W --names wsc2025-waf-alb W --query "LoadBalancers[*].DNSName" W --output text) \$ curl "http://\$ALB_ENDPOINT/lookup?id=id%3D1+OR+1%3D1"</pre>
4-0	<p>1) 웹브라우저로 선수의 AWS 계정에 로그인 후 eu-west-1 Region으로 접속합니다.</p>
4-1	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력해 172.16.0.0/16와 10.0.0.0/16가 출력 되는지 확인합니다.</p> <pre>\$ aws ec2 describe-vpcs W --filters "Name=tag:Name,Values=wsc2025-egress-vpc,wsc2025-app-vpc" W --query "Vpcs[*].CidrBlock" W --output text</pre>
4-2	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력해 순서 상관없이 10.0.4.0/24, 172.16.1.0/24, 172.16.0.0/24, 10.0.0.0/24, 10.0.1.0/24, 10.0.3.0/24, 10.0.2.0/24, 10.0.5.0/24가 출력 되는지 확인합니다.</p> <pre>\$ aws ec2 describe-subnets W --filters "Name=tag:Name,Values=wsc2025-egress-public-sn-a,wsc2025-egress-public-sn-b,wsc2025-egress-peering-sn-a,wsc2025-egress-peering-sn-b,wsc2025-egress-firewall-sn-a,wsc2025-egress-firewall-sn-b,wsc2025-app-private-sn-a,wsc2025-app-private-sn-b" W --query "Subnets[*].CidrBlock" W --output text</pre>
4-3	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력해 app-bastion이 출력 되는지 확인합니다.</p> <pre>\$ aws ec2 describe-instances W --filters "Name=tag:Name,Values=app-bastion" W --query "Reservations[*].Instances[*].Tags[?Key=='Name'].Value [0]" W --output text</pre>
4-4	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력해 “wsc2025-firewall” 와 두 개의 subnet이 출력 되는지 확인합니다.</p> <pre>\$ aws network-firewall describe-firewall W --firewall-name wsc2025-firewall W --query "[Firewall.FirewallName, Firewall.SubnetMappings[*].SubnetId]" W --output text</pre>

순번	채점 항목
4-5	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력했을 때 하나 이상의 출력값이 출력 되는지 확인합니다.</p> <pre>\$ aws network-firewall list-rule-groups W --type STATEFUL W --query "RuleGroups[*].Arn" W --output text \$ aws network-firewall list-rule-groups W --type STATELESS W --query "RuleGroups[*].Arn" W --output text</pre>
4-6	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력했을 때 하나 이상의 출력값이 출력 되는지 확인합니다.</p> <pre>\$ aws network-firewall list-rule-groups W --type STATEFUL W --query "RuleGroups[*].Arn" W --output text \$ aws network-firewall list-rule-groups W --type STATELESS W --query "RuleGroups[*].Arn" W --output text</pre>
4-7	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력했을 때 Ping이 정상적으로 가지 않은지 확인합니다.</p> <pre>\$ ping 1.1.1.1 -c 5</pre>
4-8	<p>1) App VPC에 배포된 Bastion의 System Manager에 접근합니다.</p> <p>2) 아래 명령어를 입력해 ;; communications error to 8.8.8.8#53: timed out가 출력되는지 확인합니다.</p> <pre>\$ nslookup google.com 8.8.8.8</pre>