

## 2024 2과제 전국기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅
※ 다음 사항을 유의하여 채점하십시오.		
1) AWS의 지역은 <b>과제별로 명시된 리전</b> 을 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	
제2과제	1	경북2과제3번 Elastic_stack	3.0		○		○	
	2	광주2과제3번 Collect application logs using Fluent-bit and Fluentd	3.0		○		○	
	3	대전2과제3번 EKS Observability with FluentBit	3.0		○		○	
	4	부산2과제2번 Governance	3.0		○		○	
	5	서울2과제1번 Image resizing on Edge location	3.0		○		○	
	6	서울2과제3번 IAM security maintenance system	3.0		○		○	
	7	제주2과제3번 Secure networking	3.0		○		○	
	8	충남2과제1번 Dynamic IAM Role Authorization Enforcement	3.0		○		○	
	9	ClientVPN	3.0		○		○	
	10	Keycloak	3.0		○		○	
합 계			30					

## 2) 채점방법 및 기준

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
2과제	1	경북2과제3번 Elastic_stack	1	Bastion EC2 - 인스턴스 타입 확인	0.2
			2	App EC2 - 인스턴스 타입 확인	0.2
			3	웹 어플리케이션 - API 요청 확인	0.4
			4	OpenSearch - 도메인 확인	0.4
			5	OpenSearch - 세부설정확인	0.8
			6	OpenSearch - 로그 확인	1
	2	광주2과제3번 Collect application logs using Fluent-bit and Fluentd	1	기본 배포 구성 확인	0.6
			2	중앙 집중식 로깅 확인	1.2
			3	Application Logging	1.2
	3	대전2과제3번 EKS Observability with FluentBit	1	EKS - EKS Cluster 생성 확인	0.24
			2	EKS - Namespace 확인	0.24
			3	CloudWatch Log Group 생성 확인	0.24
			4	EKS Setting - Deployment 확인	0.24
			5	EKS Setting - Sidecar Pattern 확인	0.64
			6	Application - Application 확인	0.4
			7	Application - CloudWatch Log Stream 생성 확인	0.44
			8	Application - 로그 형식 확인	0.56
	4	부산2과제2번 Governance	1	EC2 Server Configuration	0.4
			2	IAM User	0.4
			3	Trailing	0.4
			4	CloudWatch - User Login	0.4

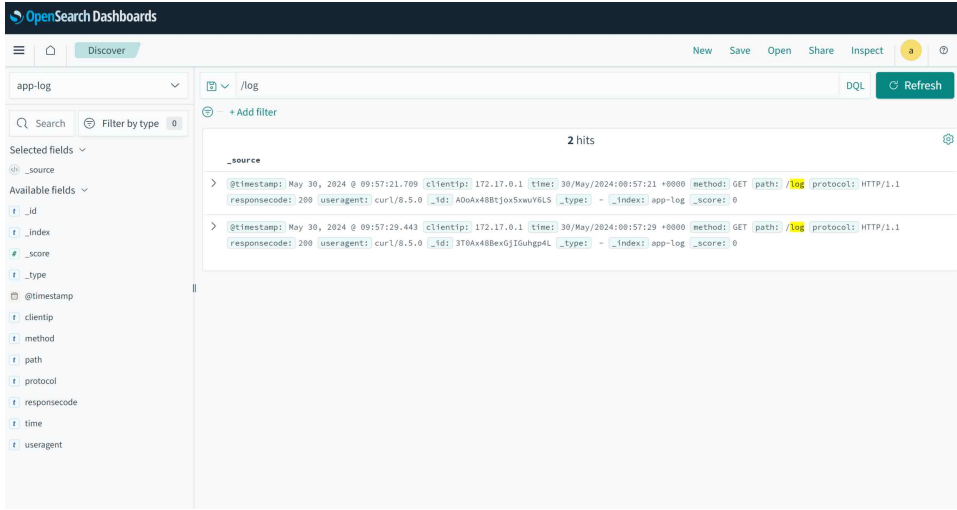
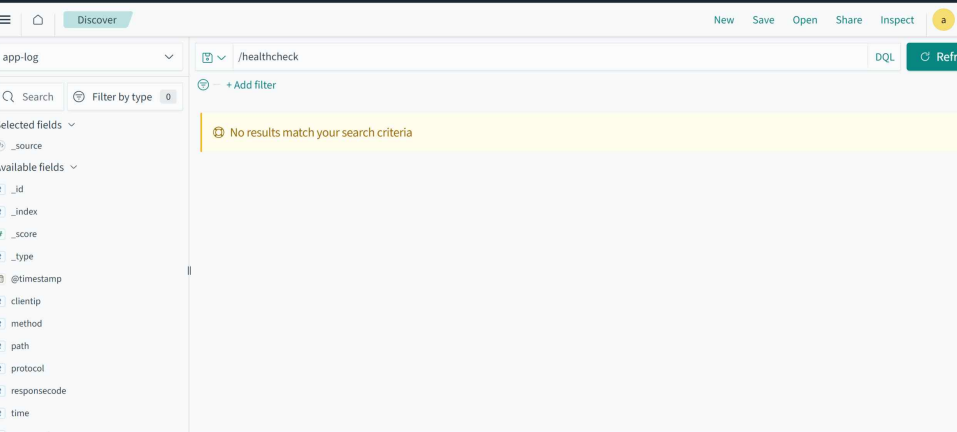
과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
2과제	4	부산2과제2번 Governance	5	Cloudwatch - Login Log	1
			6	Lambda Configuration	0.4
	5	서울2과제1번 Image resizing on Edge location	1	S3 Bucket	0.2
			2	CloudFront - Resizing	0.8
			3	CloudFront - Redirection	0.6
			4	CloudFront - Caching	0.4
			5	CloudFront - Ignore caching	0.4
			6	CloudFront - CloudFront function	0.4
			7	Lambda - Runtime	0.2
	6	서울2과제3번 IAM security maintenance system	1	tester	0.2
			2	tester admin role	0.2
			3	mfaBucketDeleteControl	1.2
			4	user_group_kr	0.2
			5	Only Seoul region	1.2
	7	제주2과제3번 Secure networking	1	S3 - S3 이름	0.12
			2	S3 - S3 접근 확인	0.64
			3	S3 - 백업 버킷 파일복사 확인	0.64
			4	SQS - S3 이벤트 발생 확인	0.64
			5	SQS - EC2 이벤트 발생 확인	0.28
			6	NAT G/W & Internet G/W 생성 여부 확인	0.4
			7	endpoint - 인스턴스 접근 서브넷 확인	0.28

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
2과제	8	충남2과제1번 Dynamic IAM Role Authorization Enforcement	1	IAM - User Name	0.16
			2	IAM - User policy	0.2
			3	IAM - instance role Name	0.16
			4	IAM - instance role policy	0.2
			5	Monitoring Service Name -CloudTrail Name	0.16
			6	Monitoring Service Name - CloudWatch Alarm Name	0.2
			7	Monitoring Service Name - CloudWtach Log Group Name	0.16
			8	Lambda - Lambda name	0.16
			9	Employee - Alarm	0.4
			10	Employee - instance role policy test	0.4
			11	Admin - Alarm	0.4
			12	Admin - instance role policy test	0.4
	9	ClientVPN	1	도메인 질의	1.5
			2	DB 연결	1.5
	10	Keycloak	1	admin 로그인	1.5
			2	dev 로그인	1.5
	총점				

### 3) 채점내용

순번	채점 항목	
1-1	1-1-A (명령어 입력)	<code>aws ec2 describe-instances --filter Name=tag:Name,Values=ws-i-bastion \</code> <code>--query "Reservations[].Instances[].InstanceType"</code>
	1-1-A (예상 출력)	[ "t3.small" ]
1-2	1-2-A (명령어 입력)	<code>aws ec2 describe-instances --filter Name=tag:Name,Values=ws-i-app \</code> <code>--query "Reservations[].Instances[].InstanceType"</code>
	1-2-A (예상 출력)	[ "t3.small" ]
1-3	1-3-A (명령어 입력)	<code>APP_PRIVATE_IP=\$(aws ec2 describe-instances --filters</code> <code>"Name=tag:Name,Values=ws-i-app" --query</code> <code>"Reservations[*].Instances[*].PrivateIpAddress" --output text)</code> <code>curl \$APP_PRIVATE_IP:5000/log</code> <code>echo ""</code> <code>curl \$APP_PRIVATE_IP:5000/healthcheck</code>
	1-3-A (예상 출력)	Log entry created status: ok
1-4	1-4-A (명령어 입력)	<code>aws opensearch list-domain-names   grep ws-i-opensearch</code>
	1-4-A (예상 출력)	"DomainName": "ws-i-opensearch"
1-5	1-5-A (명령어 입력)	<code>aws opensearch describe-domain --domain-name ws-i-opensearch --query</code> <code>"DomainStatus.ClusterConfig.[InstanceCount, DedicatedMasterCount]"</code>
	1-5-A (예상 출력)	[ 2, 3 ]

순번	채점 항목	
1-5	1-5-B (명령어 입력)	aws opensearch describe-domain --domain-name wsi-opensearch --query "DomainStatus.EngineVersion"
	1-5-B (명령어 입력)	"OpenSearch_2.13"
	1-5-C (명령어 입력)	OPENSEARCH_ENDPOINT=\$(aws opensearch describe-domain --domain-name wsi-opensearch   jq -r '.DomainStatus.Endpoint') curl -s -u admin:Password01! "https://\$OPENSEARCH_ENDPOINT/_cat/indices?index=app-log"
	1-5-C (예상 출력)	green open app-log dWxHtcnyQcGsMwhGB0FMYw 5 1 2 0 34kb 17kb 위와 같이 app-log가 포함된 출력이 있는지 확인
	1-5-D (명령어 입력)	OPENSEARCH_ENDPOINT=\$(aws opensearch describe-domain --domain-name wsi-opensearch   jq -r '.DomainStatus.Endpoint') curl -s -u admin:Password01! https://\$OPENSEARCH_ENDPOINT/app-log   jq '["app-log"].mappings.properties   keys[]'
	1-5-D (예상 출력) <b>정확히 일치하지 않아도 됨</b>	"clientip" "method" "path" "protocol" "responsecode" "time" "useragent" 위와 같이 clientip, method, path, protocol, responsecode, time, useragent가 출력되는지 확인
1-6	1-6-A (명령어 입력)	aws opensearch describe-domain --domain-name wsi-opensearch --output json   jq -r '.DomainStatus.Endpoint + "/_dashboards"'
	1-6-A (예상 출력)	대시보드 URL이 출력됩니다.
	1-6-B (대시보드 접속)	출력된 URL에 브라우저로 접속합니다. 아이디는 admin 패스워드는 Password01!로 로그인합니다. 대시보드의 discover탭에 접속합니다. index는 app-log로 지정합니다.

순번	채점 항목	
1-6	<div>1-6-B (예상 출력)</div>	<div></div> <div>로그가 출력되는지 확인합니다.</div>
	<div>1-6-C (대시보드 확인)</div>	<div>search에 /healthcheck를 입력합니다.</div>
	<div>1-6-C (예상 출력)</div>	<div></div> <div>출력되는 로그가 없는지 확인합니다.</div>




순번	채점 항목	
2-1	2-1-A (명령어 입력)	<pre>kubectl get deployment service-a -n app -o=json   jq '.spec.template.spec.containers'   grep '"image"' ₩ ; kubectl get deployment service-b -n app -o=json   jq '.spec.template.spec.containers'   grep '"image"' ₩ ; kubectl get deployment service-c -n app -o=json   jq '.spec.template.spec.containers'   grep '"image"'</pre>
	2-1-A (예상 출력)	<pre>"image": "250832144271.dkr.ecr.ap-northeast-2.amazonaws.com/service:a", "image": "fluent/fluent-bit:latest", "image": "250832144271.dkr.ecr.ap-northeast-2.amazonaws.com/service:b", "image": "fluent/fluent-bit:latest", "image": "250832144271.dkr.ecr.ap-northeast-2.amazonaws.com/service:c", "image": "fluent/fluent-bit:latest",  * 밑줄 친 부분을 제외하고 전부 동일해야 함</pre>
	2-1-B (명령어 입력)	<pre>NODE_COUNT=\$(kubectl get nodes   grep Ready   wc -l) ₩ ; DAEMONSET_COUNT=\$(kubectl get daemonsets -n fluentd   tail -n 1   awk {'print \$4'}) ₩ ; echo \$(expr \$NODE_COUNT - \$DAEMONSET_COUNT) ₩ ; kubectl describe -n fluentd daemonset fluentd   grep Image   grep fluent</pre>
	2-1-B (예상 출력) <u>정확히 일치</u>	<pre>0 Image: fluent/fluentd-kubernetes-daemonset:v1.10.3-debian-cloudwatch-1.0</pre>

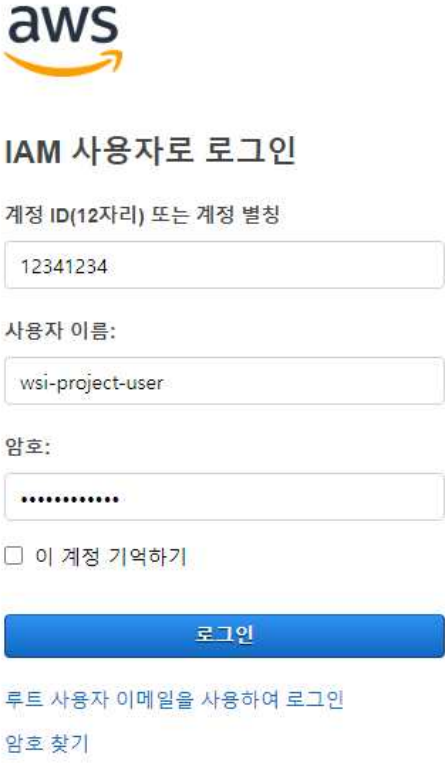
순번	채점 항목	
2-2	2-2-A (명령어 입력)	<pre>cd /tmp kubectl get daemonset fluentd -n fluentd -o yaml &gt; fluentd.yaml kubectl delete daemonset fluentd -n fluentd date kubectl exec -it -n app deployment.apps/service-a -- curl localhost:8080 &gt; /dev/null 2&gt;&amp;1 kubectl exec -it -n app deployment.apps/service-b -- curl localhost:8080 &gt; /dev/null 2&gt;&amp;1 kubectl exec -it -n app deployment.apps/service-c -- curl localhost:8080 &gt; /dev/null 2&gt;&amp;1 sleep 30; aws logs get-log-events --log-group-name /gwangju/eks/application/logs --log-stream-name service-a-logs --limit 1 --query 'events[*].message' --output json ₩ ; aws logs get-log-events --log-group-name /gwangju/eks/application/logs --log-stream-name service-b-logs --limit 1 --query 'events[*].message' --output json ₩ ; aws logs get-log-events --log-group-name /gwangju/eks/application/logs --log-stream-name service-c-logs --limit 1 --query 'events[*].message' --output json kubectl apply -f fluentd.yaml</pre>
	2-2-A (예상 출력)	<pre>daemonset.apps "fluentd" deleted Fri May 31 02:00:30 UTC 2024 [   "{w"logw":w"[2024-05-31 01:55:07,660] Service A 127.0.0.1 www"GET /WWW" 0.93ms 200w"}" ] [   "{w"logw":w"[2024-05-31 01:55:08,682] Service B 127.0.0.1 www"GET /WWW" 0.45ms 200w"}" ] [   "{w"logw":w"[2024-05-31 01:57:06,019] Service C 127.0.0.1 www"GET /WWW" 0.78ms 200w"}" ] daemonset.apps/fluentd created  * date 시간 이후의 로그가 나타나면 오답처리</pre>

순번	채점 항목	
2-3	2-3-A (명령어 입력)	<pre>date kubectl exec -it -n app deployment.apps/service-a -- curl localhost:8080 &gt; /dev/null 2&gt;&amp;1 kubectl exec -it -n app deployment.apps/service-b -- curl localhost:8080 &gt; /dev/null 2&gt;&amp;1 kubectl exec -it -n app deployment.apps/service-c -- curl localhost:8080 &gt; /dev/null 2&gt;&amp;1 sleep 30; aws logs get-log-events --log-group-name /gwangju/eks/application/logs --log-stream-name service-a-logs --limit 1 --query 'events[*].message' --output json ₩ ; aws logs get-log-events --log-group-name /gwangju/eks/application/logs --log-stream-name service-b-logs --limit 1 --query 'events[*].message' --output json ₩ ; aws logs get-log-events --log-group-name /gwangju/eks/application/logs --log-stream-name service-c-logs --limit 1 --query 'events[*].message' --output json</pre>
	2-3-A (예상 출력)	<pre>Fri May 31 01:49:42 UTC 2024 [   "{w\"logw\":w\"[2024-05-31 01:49:43,591] Service A 127.0.0.1 www\"GET /WWW\" 0.59ms 200w\"}" ] [   "{w\"logw\":w\"[2024-05-31 01:49:44,473] Service B 127.0.0.1 www\"GET /WWW\" 0.95ms 200w\"}" ] [   "{w\"logw\":w\"[2024-05-31 01:49:45,463] Service C 127.0.0.1 www\"GET /WWW\" 0.82ms 200w\"}" ]  * Service A, B, C가 한번씩 출력되어야 하며, date 명령어에 출력된 시간 이후 의 로그가 30초 내로 기록되었는지 확인</pre>

순번	채점 항목
3-1	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsi-eks-cluster” 가 출력되는지 확인합니다.</p> <pre>aws eks describe-cluster --name wsi-eks-cluster --query "cluster.name "</pre> <p>3) 아래 명령어 입력 후 “1.29” 가 출력되는지 확인합니다.</p> <pre>aws eks describe-cluster --name wsi-eks-cluster --query "cluster.version"</pre>
3-2	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsi-ns” 가 출력되는지 확인합니다.</p> <pre>kubectl get ns -o json   jq '.items[]   select(.metadata.name == "wsi-ns")   .metadata.name'</pre>
3-3	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “/wsi/eks/log/” 가 출력되는지 확인합니다.</p> <pre>aws logs describe-log-groups --query "logGroups[?contains(logGroupName, '/wsi/eks/log/')].logGroupName "</pre>
3-4	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsi-dpm” 이 출력되는지 확인합니다.</p> <pre>kubectl get deploy -n wsi-ns -o json   jq '.items[].metadata.name'</pre>
3-5	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어 입력 후 “wsi-cnt’ 와 ‘fluent-bit-cnt’ 가 출력되는지 확인합니다. (wsi-cnt 2대, fluent-bit-cnt 2대 총 4개가 존재해야 합니다.)</p> <pre>kubectl get po -n wsi-ns -o json   jq '.items[].spec.containers[].name'</pre>
3-6	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어를 입력합니다.</p> <pre>POD_NAME=\$(kubectl get po -n wsi-ns -o jsonpath='{.items[0].metadata.name}')</pre> <p>3) 아래 명령어 입력 후 “{'status': '200'} “이 출력되는지 확인합니다.</p> <pre>kubectl exec -it \$POD_NAME -n wsi-ns -- curl localhost:8080/2xx</pre> <p>3) 아래 명령어 입력 후 “{'status': '300'} “이 출력되는지 확인합니다.</p> <pre>kubectl exec -it \$POD_NAME -n wsi-ns -- curl localhost:8080/3xx</pre> <p>3) 아래 명령어 입력 후 “{'status': '400'} “이 출력되는지 확인합니다.</p> <pre>kubectl exec -it \$POD_NAME -n wsi-ns -- curl localhost:8080/4xx</pre> <p>3) 아래 명령어 입력 후 “{'status': '500'} “이 출력되는지 확인합니다.</p> <pre>kubectl exec -it \$POD_NAME -n wsi-ns -- curl localhost:8080/5xx</pre> <p>3) 아래 명령어 입력 후 “{'status': 'ok'} “이 출력되는지 확인합니다.</p> <pre>kubectl exec -it \$POD_NAME -n wsi-ns -- curl localhost:8080/healthz</pre>

순번	채점 항목
3-7	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어 입력합니다.</p> <pre>CW_LOG_STREAM_NAME=\$(aws logs describe-log-streams --log-group-name /wsi/eks/log/ W --query "logStreams[].logStreamName" --output text) POD_ID=\$(kubectl get po -n wsi-ns -o jsonpath='{.items[0].metadata.name}')</pre> <p>MATCHING_LOG_STREAM_NAME="log-\$POD_ID"</p> <p>3) 아래 명령어를 입력 후 “log-{Pod ID}” 가 출력되는지 확인합니다.</p> <p>{Pod ID}는 해당 Pod의 이름을 의미하며, 선수마다 값이 다를 수 있습니다.</p> <pre>[ "\$CW_LOG_STREAM_NAME" == "\$MATCHING_LOG_STREAM_NAME" ] &amp;&amp; aws logs describe-log-streams W --log-group-name /wsi/eks/log/ -query "logStreams[].logStreamName "</pre>
3-8	<p>1) SSH를 통해 EC2에 접근합니다.</p> <p>2) 아래 명령어를 입력합니다.</p> <pre>aws logs tail /wsi/eks/log/   tail -n 1   awk '{print substr(\$0,index(\$0,"{"))}'   jq .</pre> <p>3) 아래 사진과 같이 day, hour, ip, method, minute, month, path, port, second, statuscode, year가 존재하는지 확인합니다. (하나라도 존재하지 않을 시 틀린 것으로 간주합니다.) (Key의 위치는 달라도 상관없으며 Value 또한 차이가 있을 수 있습니다.)</p>  <pre>{   "day": "30",   "hour": "03",   "ip": "127.0.0.1",   "method": "GET",   "minute": "07",   "month": "05",   "path": "/healthz",   "port": "8080",   "second": "11",   "statuscode": "200",   "year": "2024" }</pre>

순번	채점 항목	
4-1	4-1-A (명령어 입력)	<pre>aws ec2 describe-instances --filters "Name=tag:Name,Values=ws1-bastion-ec2" --query "Reservations[*].Instances[*].[InstanceId, InstanceType, IamInstanceProfile.Arn]" --output text   awk '{print \$1,\$2,\$3}'   xargs -I {} bash -c 'INSTANCE_PROFILE_NAME=\$(echo {}   awk -F"/" '{print \\$NF}'); ROLE_NAME=\$(aws iam get-instance-profile --instance-profile-name \$INSTANCE_PROFILE_NAME --query "InstanceProfile.Roles[*].RoleName" --output text); echo "Instance ID: \$(echo {}   awk '{print \\$1}'); echo "Instance Type: \$(echo {}   awk '{print \\$2}'); aws iam list-attached-role-policies --role-name \$ROLE_NAME --query "AttachedPolicies[*].[PolicyName, PolicyArn]" --output text'</pre>
	4-1-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	<pre>Instance ID: i-xxxxxxxxxxxxxxxxxxxxxxxxxxxxx Instance Type: t3.small AdministratorAccess    arn:aws:iam::aws:policy/AdministratorAccess</pre>
4-2	4-2-A (명령어 입력)	<pre>aws iam get-user --user-name ws1-project-user --query "User.UserName" --output text \ ; aws iam list-attached-user-policies --user-name ws1-project-user --query 'AttachedPolicies[*].PolicyName' --output text &amp;&amp; aws iam list-user-policies --user-name ws1-project-user --query 'PolicyNames' --output text</pre>
	4-2-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	<pre>ws1-project-user AdministratorAccess</pre>

순번	채점 항목	
4-3	4-3-A (명령어 입력)	aws cloudtrail describe-trails --query 'trailList[?Name==`wsi-project-trail`].Name' --output json
	4-3-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	[ "wsi-project-trail" ]
4-4	4-4-A	계정에 로그아웃 후 생성했던 IAM 사용자로 로그인합니다.
	4-4-A	 <p>The screenshot shows the AWS IAM console login page. At the top is the AWS logo. Below it is the heading 'IAM 사용자로 로그인'. There are three input fields: '계정 ID(12자리) 또는 계정 별칭' with the value '12341234', '사용자 이름:' with the value 'wsi-project-user', and '암호:' with masked characters. Below the password field is a checkbox labeled '이 계정 기억하기'. At the bottom is a blue '로그인' button. Below the button are two links: '루트 사용자 이메일을 사용하여 로그인' and '암호 찾기'.</p>

순번	채점 항목	
	4-5-A (명령어 입력)	로그인 후 4분 소요 date -d "+9 hours" "+%Y-%m-%d.%H.%M.%S" echo "4분 소요" sleep 240 LOG_GROUP_NAME="wsi-project-login" aws logs describe-log-groups --log-group-name-prefix \$LOG_GROUP_NAME --query 'logGroups[?logGroupName==`wsi-project-login`].logGroupName' --output json LATEST_LOG_STREAM=\$(aws logs describe-log-streams --log-group-name \$LOG_GROUP_NAME --order-by LastEventTime --descending --limit 1 --query 'logStreams[0].logStreamName' --output text) aws logs get-log-events --log-group-name \$LOG_GROUP_NAME --log-stream-name \$LATEST_LOG_STREAM --limit 1 --query 'events[-1].{timestamp:timestamp, message:message}' --output json   jq -r '"\"(.timestamp)\\t\"(.message)\""'   while IFS=\$'\t' read -r timestamp message; do echo -e "\$(date -d @\$((timestamp / 1000 + 9 * 3600)) '+%Y-%m-%d-%H:%M:%S')\\t\$message"; done
	4-5-A (예상 출력) <u>timestamp확인하</u> <u>여 4분내에</u> <u>포함되어있는지</u> <u>확인</u>	2024-07-21.20.23.52 [ "\"wsi-project-login\"" ] 2024-07-21.20.24.51 { USER : "wsi-project-user has logged in!" } ( UTC 시간까지 득점으로 허용 )
4-6	4-6-A (명령어 입력)	aws lambda get-function --function-name wsi-project-log-function --query 'Configuration.FunctionName' --output json
	4-6-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	"wsi-project-log-function"

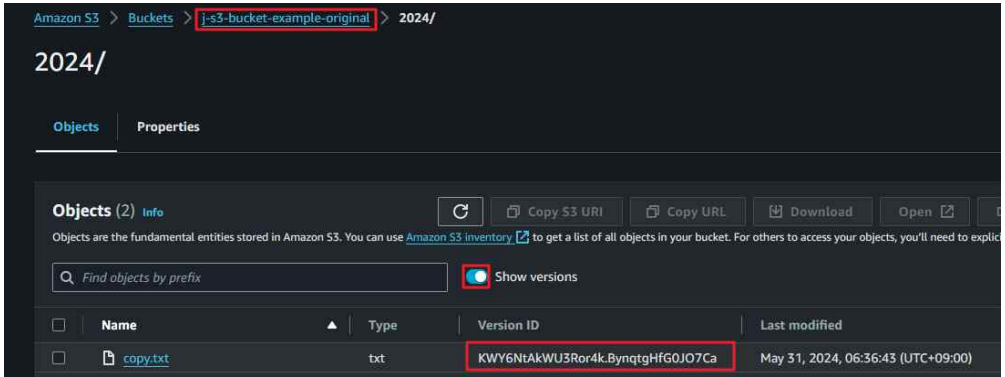
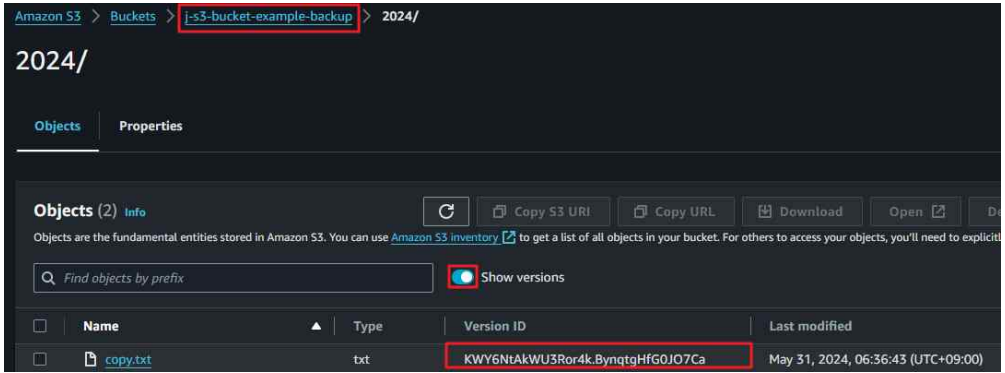


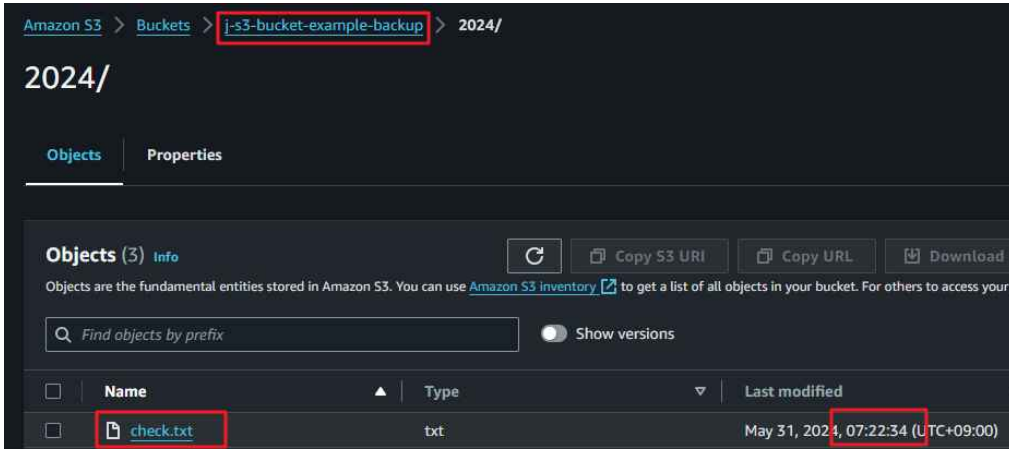
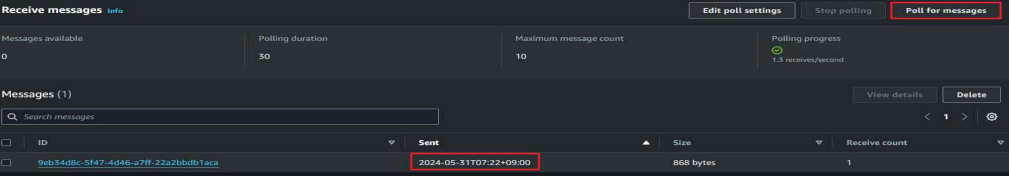
순번	채점 항목	
5-1	5-1-A (명령어 입력)	aws s3 ls s3://\${STATIC_BUCKET}/ --recursive   awk '{print \$4}'
	5-1-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	dev/ images/glass.jpg images/hamster.jpg images/library.jpg index.html
5-2	5-2-A (명령어 입력)	curl -s "https://\${CF_DOMAIN}/images/library.jpg?width=400&height=400"   file -b -   grep -oE '[0-9]+x[0-9]+'   tail -n 1
	5-2-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	400x400
5-3	5-3-A (명령어 입력)	curl -sIL -o /dev/null -w '%{url_effective}\n' https://\${CF_DOMAIN}/dev/ curl -sIL -o /dev/null -w '%{url_effective}\n' https://\${CF_DOMAIN}/
	5-3-A (예상 출력) <u>정확히 일치</u> <u>&lt;CF_DOMAIN&gt;은</u> <u>선수마다 다를 수</u> <u>있음</u>	https://<CF_DOMAIN>/index.html https://<CF_DOMAIN>/index.html
5-4	5-4-A (명령어 입력)	curl -sI "https://\${CF_DOMAIN}/images/library.jpg?width=123&height=32"   grep -i "x-cache" curl -sI "https://\${CF_DOMAIN}/images/library.jpg?width=123&height=32"   grep -i "x-cache" curl -sI "https://\${CF_DOMAIN}/images/library.jpg?width=123&height=40"   grep -i "x-cache"
	5-4-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	x-cache: Miss from cloudfront x-cache: Hit from cloudfront x-cache: Miss from cloudfront

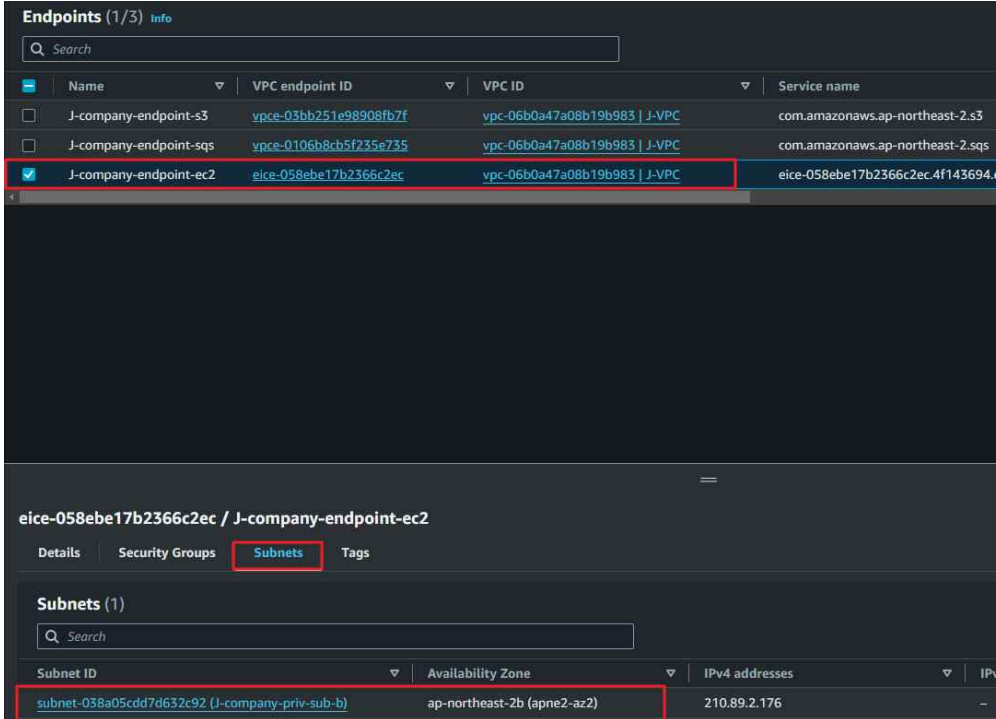
순번	채점 항목	
5-5	5-5-A (명령어 입력)	curl -sI "https://\${CF_DOMAIN}/index.html"   grep -i "x-cache" curl -sI "https://\${CF_DOMAIN}/index.html"   grep -i "x-cache"
	5-5-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	x-cache: Miss from cloudfront x-cache: Miss from cloudfront
5-6	5-6-A (명령어 입력)	aws cloudfront get-distribution-config --id \${DISTRIBUTION_ID} --query "DistributionConfig.DefaultCacheBehavior.FunctionAssociations.Quantity > \0\`    DistributionConfig.CacheBehaviors.Items[].FunctionAssociations.Quantity > \0\`" --output text
	5-6-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	True
5-7	5-7-A (명령어 입력)	aws lambda get-function-configuration --function-name \${LAMBDA_NAME} --region us-east-1 --query "Runtime" --output text
	5-7-A (예상 출력) <u>둘 중 하나만</u> <u>일치</u>	nodejs20.x python3.9

순번	채점 항목	
6-1	6-1-A (명령어 입력)	aws iam list-users --query "Users[?UserName=='tester']"   grep -i username
	6-1-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	"UserName": "tester",
6-2	6-2-A (명령어 입력)	aws iam list-attached-user-policies --user-name tester   grep AdministratorAccess
	6-2-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	"PolicyName": "AdministratorAccess", "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
6-3	6-3-A (명령어 입력)	aws iam get-policy-version --policy-arn arn:aws:iam::\$(aws sts get-caller-identity --query "Account" --output text):policy/mfaBucketDeleteControl --version-id v1 --query 'PolicyVersion.Document.Statement[0].Condition'   jq
	6-3-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	[ { "Bool": { "aws:MultiFactorAuthPresent": "false" } } ]
6-4	6-4-A (명령어 입력)	aws iam list-groups --query "Groups[?GroupName=='user_group_kr']"   grep GroupName
	6-4-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	"GroupName": "user_group_kr",

순번	채점 항목	
6-5	6-5-A (명령어 입력)	aws iam simulate-principal-policy \ --policy-source-arn arn:aws:iam::\$ACCOUNT_ID:user/tester \ --action-names ec2:DescribeInstances \ --region ap-northeast-2   jq '.EvaluationResults[0].EvalDecision'
	6-5-A (예상 출력) <u>강조색 일치</u>	"explicitDeny"
7-1	7-1-A (명령어 입력)	buckets=\$(aws s3 ls   awk '{print \$3}'   grep -E 'j-s3-bucket-[a-zA-Z0-9]{7}-(original backup)') bucket1=\$(echo "\$buckets"   grep 'original'   head -n 1) bucket2=\$(echo "\$buckets"   grep 'backup'   head -n 1) buckets=\$(aws s3 ls   awk '{print \$3}'   grep -E '^j-s3-bucket-[a-zA-Z0-9]{7}-(original backup)\$') echo "\$buckets"
	7-1-A <u>정확히 일치</u>	j-s3-bucket-<임의 7글자>-backup j-s3-bucket-<임의 7글자>-original
7-2	7-2-A (명령어 입력)	echo "test" > test.txt aws s3 cp test.txt s3://j-s3-bucket-<임의 7글자>-original/ aws s3 cp test.txt s3://j-s3-bucket-<임의 7글자>-original/2024/ aws s3 cp test.txt s3://j-s3-bucket-<임의 7글자>-backup/ sleep 80
	7-2-A <u>정확히 일치</u>	upload: ./test.txt to s3://j-s3-bucket-<임의 7글자>-original/test.txt upload: ./test.txt to s3://j-s3-bucket-<임의 7글자>-original/2024/test.txt upload: ./test.txt to s3://j-s3-bucket-<임의 7글자>-backup/test.txt
	7-2-B (명령어 입력)	aws s3 cp test.txt s3://j-s3-bucket-<임의 7글자>-backup/2024/
	7-2-B <u>정확히 일치</u>	upload failed: ./test.txt to s3://j-s3-bucket-<임의 7글자>-backup/2024/test.txt An error occurred (AccessDenied) when calling the PutObject operation: Access Denied

순번	채점 항목	
7-3	7-3-A (명령어 입력)	<pre>echo "copy" &gt; copy.txt aws s3 cp copy.txt s3://j-s3-bucket-&lt;임의 7글자&gt;-original/2024/ sleep 80</pre>
	7-3-A <u>정확히 일치</u>	<pre>upload: ./copy.txt to s3://j-s3-bucket-&lt;임의 7글자&gt;-original/2024/copy.txt</pre>
	7-3-B <u>백업 버킷에</u> <u>파일이</u> <u>올라오는지 확인</u>	<p><a href="https://ap-northeast-2.console.aws.amazon.com/s3">https://ap-northeast-2.console.aws.amazon.com/s3</a> 으로 이동을 합니다.</p> <p>j-s3-bucket-&lt;임의 7글자&gt;-backup으로 이동합니다. 2024/ 폴더로 이동 후에, copy.txt 파일이 있는 지 확인을 합니다. 있을 경우에 j-s3-bucket-&lt;임의 7글자&gt;-original 버킷과 j-s3-bucket-&lt;임의 7글자&gt;-backup의 2024/ 폴더에 들어가서 show versions를 눌러서 copy.txt Version ID가 같은 지 확인 합니다. 4분 이내로 복사가 되어야 합니다.</p>  
7-4	7-4-A (명령어 입력)	<pre>echo "check" &gt; check.txt aws s3 cp check.txt s3://j-s3-bucket-&lt;임의 7글자&gt;-original/2024/ sleep 80</pre>
	7-4-A <u>정확히 일치</u>	<pre>upload: ./check.txt to s3://j-s3-bucket-&lt;임의 7글자&gt;-original/2024/check.txt</pre>

순번	채점 항목	
7-4	<p>7-4-B <u>수동</u></p>	<p><a href="https://ap-northeast-2.console.aws.amazon.com/s3/">https://ap-northeast-2.console.aws.amazon.com/s3/</a> 으로 이동합니다.</p> <p>j-s3-bucket-example-backup에 2024/ 폴더에 check.txt 이름의 파일이 1개 인지 확인을 하고, Last Modified를 확인합니다.</p> 
	<p>7-4-C <u>Event 발생하 지 확인</u></p>	<p><a href="https://ap-northeast-2.console.aws.amazon.com/sqs/v3/home?region=ap-northeast-2#/queues">https://ap-northeast-2.console.aws.amazon.com/sqs/v3/home?region=ap-northeast-2#/queues</a> 으로 이동합니다. 만든 Queues으로 이동합니다. 이동 후에 상단 오른쪽에 Send and receive messages를 클릭 합니다.</p> <p>j-s3-bucket-example-backup에 2024폴더의 파일이 올라오면 Poll for messages를 클릭 합니다. Sent 시간이 j-s3-bucket-example-backup에 2024/ 폴더에 check.txt 의 Last Modified 하고 같아야 합니다. (1분 정도의 오차범위가 있을 수 있습니다.) ex ) Last modified 07:22 -&gt; SQS Send 07:23 O 07:21 X</p> 
7-5	<p>7-5-A <u>수동</u></p>	<p>aws sqs send-message --queue-url W <a href="https://sqs.ap-northeast-2.amazonaws.com/&lt;account-id&gt;/&lt;queue-name&gt;">https://sqs.ap-northeast-2.amazonaws.com/&lt;account-id&gt;/&lt;queue-name&gt;</a> W --message-body "Hello from EC2! "</p> <p>&lt;account-id&gt;랑 &lt;queue-name&gt; 선수가 지정한 것들로 바꿔야 합니다.</p>
	<p>7-5-A <u>정확히 일치</u></p>	<pre>{   "MD5OfMessageBody": "d6d829edde2cd621a6f20ef7e8c15c61",   "MessageId": "37c567b2-175f-4b23-9532-bc6c3d117e47" }</pre> <p>MD5OfMessageBody와 MessageId는 다를 수 있습니다.</p>

순번	채점 항목	
7-6	7-6-A <u>수동</u>	<p><a href="https://ap-northeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-northeast-2#NatGateways">https://ap-northeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-northeast-2#NatGateways</a>:</p> <p>1) 위에 URL로 이동해서 Bastion이 속한 VPC에 Nat G/W가 생성이 안되어 있는지 확인 합니다.</p> <p><a href="https://ap-northeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-northeast-2#igws">https://ap-northeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-northeast-2#igws</a>:</p> <p>2) 위에 URL로 이동해서 Bastion이 속한 VPC에 IGW가 생성이 안되어 있는지 확인 합니다.</p> <p>3) 하나라도 생성이 되어 있을 경우에 이 항목은 0점 처리 됩니다.</p>
7-7	7-7-A <u>수동</u>	<p><a href="https://ap-northeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-northeast-2#Endpoints">https://ap-northeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-northeast-2#Endpoints</a>:</p> <p>위에 URL로 이동합니다. 이동 후에 VPC endpoint ID가 eice- 로 시작하는 endpoint를 클릭 후에 Subnet을 눌러서 J-company-priv-sub-b 으로 되어 있는지 확인 합니다.</p>  <p>The screenshot shows the AWS VPC console. At the top, the 'Endpoints (1/3)' section is visible, listing three endpoints: 'J-company-endpoint-s3', 'J-company-endpoint-sqs', and 'J-company-endpoint-ec2'. The 'J-company-endpoint-ec2' endpoint is selected, and its details are shown below. The 'Subnets' tab is active, displaying a table with one subnet: 'subnet-038a05cdd7d632c92 (J-company-priv-sub-b)' in the 'ap-northeast-2b (apne2-az2)' availability zone, with an IPv4 address of '210.89.2.176'.</p>

순번	채점 항목	
8-1	8-1-A (명령어 입력)	aws iam list-users --query "Users[?UserName=='myadmin'].UserName" aws iam list-users --query "Users[?UserName=='Employee'].UserName"
	8-1-A (예상 출력) <u>정확히 일치</u>	[ "myadmin" ], [ " Employee" ]
8-2	8-2-A (명령어 입력)	aws iam list-attached-user-policies --user-name mydmin aws iam list-attached-user-policies --user-name Employee
	8-2-A (예상 출력) <u>정확히 일치</u>	{ "AttachedPolicies": [ { "PolicyName": "AdministratorAccess", "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess" } ] }, { "AttachedPolicies": [ { "PolicyName": "IAMFullAccess", "PolicyArn": "arn:aws:iam::aws:policy/IAMFullAccess" } ] }

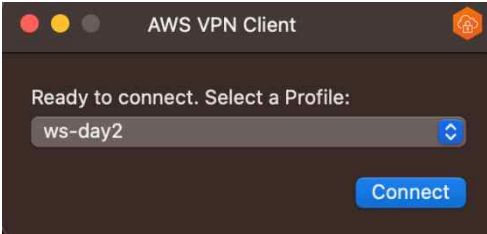




순번	채점 항목	
8-3	8-3-A (명령어 입력)	aws iam get-role --role-name wsc2024-instance-role --query 'Role.RoleName'
	8-3-A (예상 출력) <u>정확히 일치</u>	"wsc2024-instance-role"
8-4	8-4-A (명령어 입력)	aws iam list-attached-role-policies --role-name wsc2024-instance-role --query "AttachedPolicies[].PolicyArn"
	8-4-A (예상 출력) <u>정확히 일치</u>	[ "arn:aws:iam:aws:policy/AmazonSSMManagedInstanceCore" ]
8-5	8-5-A (명령어 입력)	aws cloudtrail describe-trails --trail-name-list wsc2024-CT --query "trailList[].Name"
	8-5-A (예상 출력) <u>정확히 일치</u>	[ "wsc2024-CT" ]
8-6	8-6-A (명령어 입력)	aws cloudwatch describe-alarms --alarm-name-prefix wsc2024-gvn-alarm --query 'MetricAlarms[*].AlarmName'
	8-6-A (예상 출력) <u>정확히 일치</u>	[ "wsc2024-gvn-alarm" ]
8-7	8-7-A (명령어 입력)	aws logs describe-log-groups --log-group-name-prefix wsc2024-gvn-LG --query 'logGroups[].logGroupName' --output text
	8-7-A (예상 출력) <u>정확히 일치</u>	wsc2024-gvn-LG

순번	채점 항목	
8-8	8-8-A (명령어 입력)	aws lambda list-functions --query "Functions[?FunctionName=='wsc2024-gvn-Lambda'].FunctionName"
	8-8-A (예상 출력) <u>정확히 일치</u>	[ "wsc2024-gvn-Lambda" ]
8-9	8-9-A (명령어 입력)	USERNAME="Employee" CREATED_KEYS=\$(aws iam create-access-key --user-name "\$USERNAME") ACCESS_KEY_ID=\$(echo "\$CREATED_KEYS"   jq -r '.AccessKey.AccessKeyId') SECRET_ACCESS_KEY=\$(echo "\$CREATED_KEYS"   jq -r '.AccessKey.SecretAccessKey') aws configure set aws_access_key_id "\$ACCESS_KEY_ID" aws configure set aws_secret_access_key "\$SECRET_ACCESS_KEY" sleep 10 aws iam attach-role-policy --role-name wsc2024-instance-role --policy-arn arn:aws:iam::aws:policy/AdministratorAccess rm -rf ~/.aws/* timeout 180 bash -c 'while [ "\$(aws cloudwatch describe-alarms --alarm-names "wsc2024-gvn-alarm" --query "MetricAlarms[0].StateValue" --output text)" != "ALARM" ]; do echo "Waiting for alarm to enter ALARM state..."; sleep 30; done; echo "Alarm is now in ALARM state."
	8-9-A (예상 출력) <u>밑줄 부분 일치</u>	Wating for alarm to enter ALARM state... Wating for alarm to enter ALARM state... Wating for alarm to enter ALARM state... Wating for alarm to enter ALARM state... <u>Alarm is now in ALARM state.</u>

순번	채점 항목	
8-10	8-10-A (명령어 입력)	aws iam list-attached-role-policies --role-name wsc2024-instance-role
	8-10-A (예상 출력) <u>정확히 일치</u>	{ "AttachedPolicies": [ { "PolicyName": "AmazonSSMManagedInstanceCore", "PolicyArn": "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" } ] }
8-11	8-11-A (명령어 입력)	USERNAME="Admin" CREATED_KEYS=\$(aws iam create-access-key --user-name "\$USERNAME") ACCESS_KEY_ID=\$(echo "\$CREATED_KEYS"   jq -r '.AccessKey.AccessKeyId') SECRET_ACCESS_KEY=\$(echo "\$CREATED_KEYS"   jq -r 'AccessKey.SecretAccessKey') aws configure set aws_access_key_id "\$ACCESS_KEY_ID" aws configure set aws_secret_access_key "\$SECRET_ACCESS_KEY" sleep 10 aws iam attach-role-policy --role-name wsc2024-instance-role --policy-arn arn:aws:iam::aws:policy/AdministratorAccess sleep 180 && aws cloudwatch describe-alarms --alarm-names "wsc2024-gvn-alarm" --query "MetricAlarms[0].StateValue" --output text
	8-11-A (예상 출력) <u>ALARM 상태가</u> <u>아닌지만 확인</u>	INSUFFICIENT_DATA OK

순번	채점 항목	
8-12	8-12-A (명령어 입력)	aws iam list-attached-role-policies --role-name wsc2024-instance-role
	8-12-A <u>정확히 일치</u>	<pre>{   "AttachedPolicies": [     {       "PolicyName": "AmazonSSMManagedInstanceCore",       "PolicyArn": "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"     }   ] }, {   "AttachedPolicies": [     {       "PolicyName": "AdministratorAccess",       "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"     }   ] }</pre>

순번	채점 항목
9-1	<p>아래 요소 중 어떤것이든 해당하면 비정상 방법을 이용해 풀었으므로 <b>Client VPN 항목 모두 채점하지 않습니다</b>. 기타 사항 또한 확인 하여 부정한 방법을 사용 했다고 판단하면 채점하지 않습니다.</p> <ol style="list-style-type: none"> <li>1) Client VPN 연결이 해제되어 있는지 확인합니다.</li> <li>2) 선수 Windows PC에서 아래 명령어를 입력하여 쿼리가 실패하는지 확인합니다.        &gt; nslookup db.day2.local</li> <li>3) 선수 Windows PC에서 아래 명령어를 입력하여 호스트를 찾을 수 없다는 에러로 연결에 실패하는지 확인 합니다.        &gt; ping db.day2.local</li> <li>4) 선수 Windows PC에서 아래 명령어를 입력하여 listen 되는 포트가 없는지 확인합니다.        &gt; netstat -an   findstr 5432</li> </ol> <p>1) Windows PC에서 Client VPN에 연결합니다. 반드시 ws-day2 프로필을 사용 하는지 확인 합니다.</p>  <p>2) VPN 연결 후 Windows에서 아래 명령어를 입력하여 올바른 canonical name과 IP가 반환 되는지 확인합니다.        &gt; nslookup db.day2.local</p> <p>canonical name : rds.amazonaws.com. 포함 출력        IP address : 172.26.10.1 ~ 172.26.10.254 혹은 172.26.20.1 ~ 172.26.20.254 사이 주소</p>
9-2	<ol style="list-style-type: none"> <li>1) Windows PC에서 Client VPN이 연결 되어 있는지 확인합니다. (ws-day2 프로필 사용)</li> <li>2) VPN에 연결 후 Windows에서 아래 명령어를 입력하여 DB 접근이 되는지 확인합니다.        &gt; psql -U day2_root -d day2 -h db.day2.local -p 5432</li> <li>3) 아래 명령어를 입력하여 day2 Database가 생성 되어 있는지 확인 합니다. (소문자 l)        PSQL&gt; \l</li> </ol>

순번	채점 항목
10-1	<p>아래 요소 중 어떤것이러도 해당하면 비정상 방법을 이용해 풀었으므로 <b>Keycloak 항목 모두 채점하지 않습니다</b>. 기타 사항 또한 확인하여 부정한 방법을 사용 했다고 판단하면 채점하지 않습니다.</p> <p>1) 선수가 제출한 URL을 확인해 Keycloak 서버가 선수의 계정에 있는지 확인합니다.</p>
	<p>1) 브라우저를 새창을 생성하여 과제지에 적힌 Keycloak 주소로 접속 하여 아래 비슷한 페이지가 출력 되는지 확인합니다.</p> <div data-bbox="218 631 815 963">  </div> <p>2) 계정 이름에 admin&lt;등번호&gt; 를 적어 정상적으로 로그인 되는지 확인합니다.</p> <p>3) AWS 페이지로 전환 되어 아래와 비슷한 페이지가 보이는지 확인합니다. UI는 약간 다를 수 있지만 선수 계정 번호와 admin-access role은 일치 해야 합니다.</p> <div data-bbox="225 1160 904 1451">  </div> <p>4) admin-access Role 사용하여 AWS 계정에 정상 로그인 되는지 확인 합니다. 로그 인 후 오른쪽 상단에 계정 정보를 확인 하여 계정 번호와 Role 이름을 확인합니다.</p>

순번	채점 항목
10-2	<p>아래 요소 중 어떤것이든 해당하면 비정상 방법을 이용해 풀었으므로 <b>Keycloak 항목 모두 채점하지 않습니다</b>. 기타 사항 또한 확인하여 부정한 방법을 사용 했다고 판단하면 채점하지 않습니다.</p> <p>1) 선수가 제출한 URL을 확인해 Keycloak 서버가 선수의 계정에 있는지 확인합니다.</p>
	<p>1) 브라우저를 새창을 생성하여 과제지에 적힌 Keycloak 주소로 접속하여 아래 비슷한 페이지가 출력 되는지 확인합니다.</p> <div data-bbox="217 672 815 1005" data-label="Image"> </div> <p>2) 계정 이름에 dev&lt;등번호&gt; 를 적어 정상적으로 로그인 되는지 확인합니다.</p> <p>3) AWS 페이지로 전환 되어 아래와 비슷한 페이지가 보이는지 확인합니다. UI는 약간 다를 수 있지만 선수 계정 번호와 poweruser-access role은 일치해야 합니다.</p> <div data-bbox="226 1245 1150 1639" data-label="Image"> </div> <p>4) poweruser-access Role 사용하여 AWS 계정에 정상 로그인 되는지 확인 합니다. 로그 인 후 오른쪽 상단에 계정 정보를 확인 하여 계정 번호와 Role 이름을 확인합니다.</p>