

2025년도 전국기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅
<p>※ 다음 사항을 유의하여 채점하시오.</p> <ol style="list-style-type: none"> 1) AWS 지역은 각 Module에서 명시된 Region사용합니다. 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 채점 자료는 되돌릴 수 없음으로 유의하여 진행하며, 이의 신청까지 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 붙여 넣어 명령을 실행합니다. 13) 채점시 명령어 입력은 CloudShell을 이용할 수 있습니다. 		

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	NoSQL	7.5	○			○	
	2	CDN	7.5	○			○	
	3	Monitoring	7.5	○			○	
	4	WAF	7.5	○			○	
합 계			30					

2) 채점방법 및 기준

과제 번호	일 련 번 호	주요항목	일련 번호	세부항목(채점방법)	배점
제2과제	1	NoSQL	1	DynamoDB 생성 및 설정 확인	1.5
			2	Lambda 생성 확인	1.5
			3	Application 확인1	1.5
			4	Application 확인 2	1.5
			5	DynamoDB Conflict Resolver 작동 확인	1.5
	2	CDN	1	S3, Lambda, CloudFront 생성 확인	1
			2	CloudFront Function확인	1
			3	Lambda@Edge 확인	1
			4	CDN 확인1	1.5
			5	CDN 확인2	1.5
			6	CDN 확인3	1.5
	3	Monitoring	1	VPC 구성	0.25
			2	Subnet 구성	0.25
			3	ECS 구성	1.5
			4	Cloudwatch Dashboard 구성	0.5
			5	Success Widget 확인	1.5
			6	Fail Widget 확인	1.5
			7	Percentile Widget 확인	0.5
			8	Log Insights 확인	1.5

과제 번호	일 련 번 호	주요항목	일련 번호	세부항목(채점방법)	배점
제2과제	4	WAF	1	EC2 Configure	0.5
			2	ELB Configure	0.5
			3	WAF Configure	0.5
			4	Application Test	1.5
			5	Internal Recon Attack	1.5
			6	Billion Laughs Attack	1.5
			7	Extracting Local Files Attack	1.5
합계					30

3) 채점 내용

순번	채점 항목
0	<p>1) 선수가 구성한 Bastion에 접근합니다.</p> <p>2) root로 접근합니다.</p> <p>3) 아래 파일들을 CloudShell의 /root/markings 디렉토리로 복사합니다.</p> <ul style="list-style-type: none">- mark.sh <p>4) /root/markings 경로에서 스크립트를 실행합니다. 실행 결과를 기반으로 채점을 진행하되 선수가 이의를 제기할 경우 수동으로 채점을 진행할 수 있도록 합니다.</p> <p>5) 채점을 진행하기 전에 다음 명령어를 수행하여 채점 진행을 위한 사전 작업을 조건에 맞게 진행합니다.</p> <p>(채점 스크립트도 진행 시 생략)</p>
	<pre># set default region of aws cli with NoSQL aws configure set default.region ap-northeast-2 # set default region of aws cli with CDN aws configure set default.region us-east-1 # set default region of aws cli with Monitoring aws configure set default.region ap-northeast-1 # set default region of aws cli with WAF aws configure set default.region us-west-1 # set default output of aws cli aws configure set default.output json</pre>

순번	채점 항목	
1-1	1-1-A (명령어 입력)	<pre>aws dynamodb describe-table --table-name account-table --query "Table.TableName" --output text aws dynamodb describe-table --table-name account-table --query "Table.Replicas[].RegionName" --output text aws dynamodb describe-continuous-backups --table-name account-table --query "ContinuousBackupsDescription.PointInTimeRecoveryDescription.PointInTimeR ecoveryStatus" --output text</pre>
	1-1-A <u>정확히 일치</u>	<pre>account-table eu-central-1 ENABLED</pre>
1-2	1-2-A (명령어 입력)	<pre>aws lambda get-function --function-name account-conflict-resolver --query "Configuration.FunctionName" --output text</pre>
	1-2-A <u>정확히 일치</u>	<pre>account-conflict-resolver</pre>

순번	채점 항목	
1-3	1-3-A (명령어 입력)	<pre> APP_SRV_PUBLIC_IP=\$(aws ec2 describe-instances --filter Name=tag:Name,Values=account-app-ec2 --query "Reservations[].Instances[].PublicIpAddress" --output text) ACCOUNT_ID_V1="id-\$RANDOM" ACCOUNT_ID_V2="id-\$RANDOM" BALANCE1=\$(shuf -i 1-10000 -n 1) BALANCE2=\$(shuf -i 1-10000 -n 1) CURRENCY1=\$(shuf -e USD EUR KRW -n 1) CURRENCY2=\$CURRENCY1 curl -s -X POST "\$APP_SRV_PUBLIC_IP:8080/create_account" \ -H "Content-Type: application/json" \ -d "{\"account_id\": \"\$ACCOUNT_ID_V1\", \"balance\": \$BALANCE1, \"currency\": \"\$CURRENCY1\"}" curl -s -X POST "\$APP_SRV_PUBLIC_IP:8080/create_account" \ -H "Content-Type: application/json" \ -d "{\"account_id\": \"\$ACCOUNT_ID_V2\", \"balance\": \$BALANCE2, \"currency\": \"\$CURRENCY2\"}" </pre>
	1-3-A <u>부분 일치</u>	<pre> (빨간색 부분 제외 전부 일치해야 득점) { "message": "Account id-30565 created." } { "message": "Account id-29923 created." } </pre>

1-4	1-4-A (명령어 입력)	<pre>aws dynamodb get-item \ --table-name account-table \ --region ap-northeast-2 \ --key "{\"account_id\": {\"S\": \"\$ACCOUNT_ID_V1\"}}\" --query '{account_id: Item.account_id.S, balance: Item.balance.N, currency: Item.currency.S}'</pre> <pre>aws dynamodb get-item \ --table-name account-table \ --region ap-northeast-2 \ --key "{\"account_id\": {\"S\": \"\$ACCOUNT_ID_V2\"}}\" --query '{account_id: Item.account_id.S, balance: Item.balance.N, currency: Item.currency.S}'</pre>
	1-4-A <u>부분 일치</u>	<p>(account_id, balance, currency 존재해야 득점)</p> <pre>{ "account_id": "id-30565", "balance": "9593", "currency": "EUR" }</pre> <pre>{ "account_id": "id-29923", "balance": "7307", "currency": "EUR" }</pre>

1-5	1-5-A (명령어 입력)	<pre> ACCOUNT_ID_V3="id-\$RANDOM" BALANCE1=\$(shuf -i 1-10000 -n 1) BALANCE2=\$(shuf -i 1-10000 -n 1) CURRENCY=\$(shuf -e USD EUR KRW -n 1) echo "ap-northeast-2 balance: \$BALANCE1" echo "eu-central-1 balance: \$BALANCE2" aws dynamodb update-item \ --table-name account-table \ --region ap-northeast-2 \ --key "{\"account_id\": {\"S\": \"\$ACCOUNT_ID_V3\"}}" \ --update-expression "SET balance = :b, currency = :c" \ --expression-attribute-values "{\"b\": {\"N\": \"\$BALANCE1\"}, \"c\": {\"S\": \"\$CURRENCY\"}}" aws dynamodb update-item \ --table-name account-table \ --region eu-central-1 \ --key "{\"account_id\": {\"S\": \"\$ACCOUNT_ID_V3\"}}" \ --update-expression "SET balance = :b, currency = :c" \ --expression-attribute-values "{\"b\": {\"N\": \"\$BALANCE2\"}, \"c\": {\"S\": \"\$CURRENCY\"}}" sleep 30 aws dynamodb get-item \ --table-name account-table \ --region ap-northeast-2 \ --key "{\"account_id\": {\"S\": \"\$ACCOUNT_ID_V3\"}}" --query '{"account_id": Item.account_id.S, balance: Item.balance.N, currency: Item.currency.S}' aws dynamodb get-item \ --table-name account-table \ --region eu-central-1 \ --key "{\"account_id\": {\"S\": \"\$ACCOUNT_ID_V3\"}}" --query '{"account_id": Item.account_id.S, balance: Item.balance.N, currency: Item.currency.S}' </pre>
-----	-------------------	--

1-5	1-5-A <u>부분 일치</u>	<p>(아래 조건이 일치해야 득점 1. account_id, balance, currency 존재 2. ap-northeast-2 balance 또는 eu-central-1 balance 값 중 하나가 조회한 DynamoDB Item 값 중 balance 값과 동일해야함)</p> <p>ap-northeast-2 balance: 2212 eu-central-1 balance: 4509</p> <pre>{ "account_id": "id-29964", "balance": "4509", "currency": "KRW" } { "account_id": "id-29964", "balance": "4509", "currency": "KRW" }</pre>
-----	-----------------------	---

순번	채점 항목	
2-1	2-1-A (명령어 입력)	<pre>S3_BUCKET_NAME=\$(aws s3api list-buckets --query "Buckets[?starts_with(Name, 'web-drm-bucket')].Name" --output text) aws s3 ls \$S3_BUCKET_NAME/ --recursive awk '{print \$4}'</pre>
	2-1-A (예상 출력) <u>부분 일치</u> <u>순서 중요</u> (배점 0.3점)	<pre>media/cloud.mp4 media/fire.mp4 media/mountain.mp4 media/rain.mp4 media/sea.mp4</pre>
	2-1-B (명령어 입력)	<pre>aws lambda list-functions --query "Functions[?FunctionName=='web-drm-function'].FunctionName" --output text aws lambda get-function-configuration --function-name web-drm-function --region us-east-1 --query "Runtime" --output text</pre>
	2-1-B <u>정확히 일치</u> (배점 0.3점)	<pre>web-drm-function python3.13</pre>
	2-1-C (명령어 입력)	<pre>aws resourcegroupstaggingapi get-resources --tag-filters Key=Name,Values=web-cdn --resource-type-filters 'cloudfront' --region us-east-1 --query "ResourceTagMappingList[0].ResourceARN" --output text sed 's:./::' xargs -l {} aws cloudfront get-distribution --id {} --query "Distribution.DomainName" --output text</pre>
	2-1-C (예상 출력) <u>cloudfront.net</u> <u>존재 확인</u> (배점 0.4점)	<pre>d6m4ppgylgncs.cloudfront.net</pre>

순번	채점 항목	
2-2	2-2-A (명령어 입력)	<pre> CLOUDFRONT_DISTRIBUTION_ID=\$(aws resourcegroupstaggingapi get-resources --tag-filters Key=Name,Values=web-cdn --resource-type-filters 'cloudfront' --region us-east-1 --query "ResourceTagMappingList[0].ResourceARN" --output text sed 's:./:/:') aws cloudfront get-distribution --id \$CLOUDFRONT_DISTRIBUTION_ID --query "Distribution.DistributionConfig.DefaultCacheBehavior.FunctionAssociations.Items[?contains(FunctionARN, 'web-cdn-function')].EventType" --output text </pre>
	2-2-A <u>정확히 일치</u>	viewer-request
2-3	2-3-A (명령어 입력)	<pre> CLOUDFRONT_DISTRIBUTION_ID=\$(aws resourcegroupstaggingapi get-resources --tag-filters Key=Name,Values=web-cdn --resource-type-filters 'cloudfront' --region us-east-1 --query "ResourceTagMappingList[0].ResourceARN" --output text sed 's:./:/:') aws cloudfront get-distribution --id \$CLOUDFRONT_DISTRIBUTION_ID --query "Distribution.DistributionConfig.DefaultCacheBehavior.LambdaFunctionAssociations.Items[?contains(LambdaFunctionARN, 'web-drm-function')].EventType" --output text </pre>
	2-3-A <u>정확히 일치</u>	origin-request

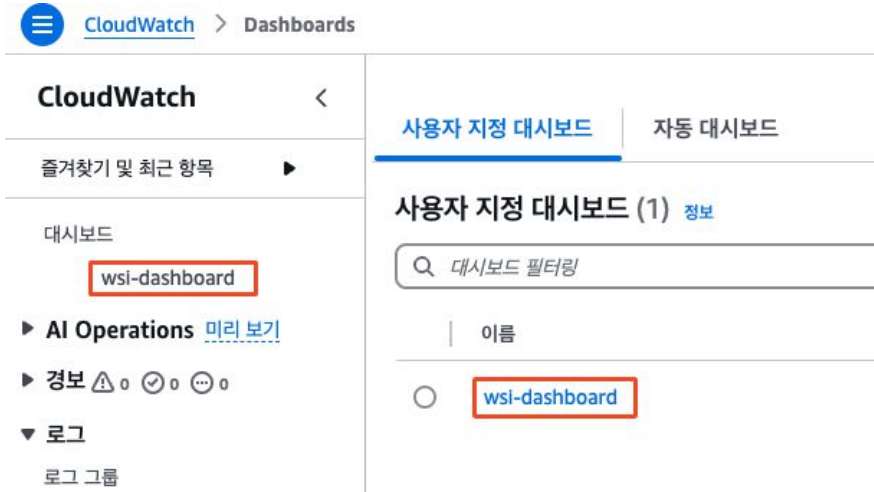
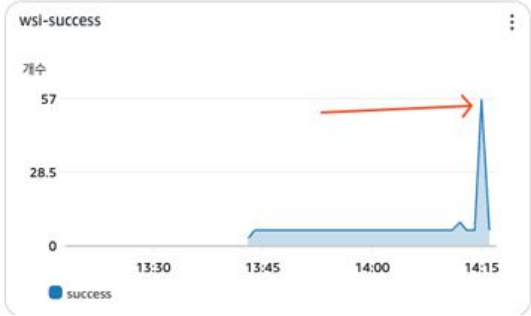

2-4	2-4-A (명령어 입력)	<pre>CLLOUDFRONT_DNS=\$(aws resourcegroupstaggingapi get-resources --tag-filters Key=Name,Values=web-cdn --resource-type-filters 'cloudfront' --region us-east-1 --query "ResourceTagMappingList[0].ResourceARN" --output text sed 's:.*/:::' xargs -l {} aws cloudfront get-distribution --id {} --query "Distribution.DomainName" --output text) curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/cloud.mp4" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/fire.mp4" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/mountain.mp4" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/rain.mp4" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/sea.mp4"</pre>
	2-4-A <u>정확히 일치</u>	403 403 403 403 403

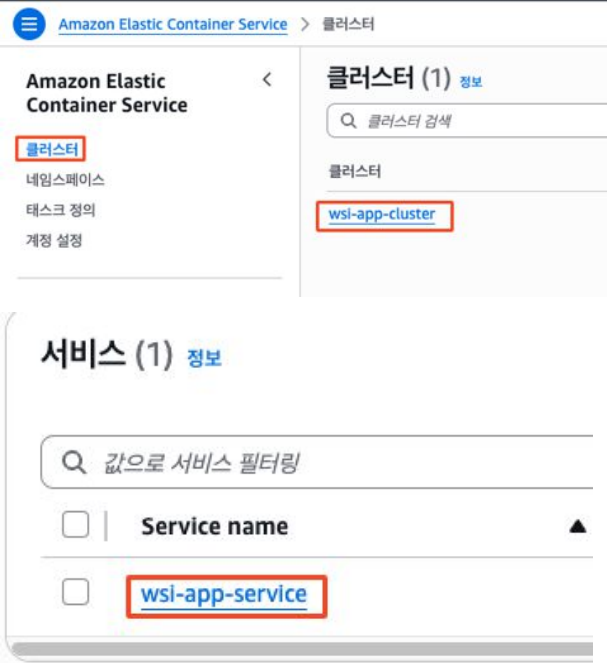


2-5	2-5-A (명령어 입력)	<pre>CLLOUDFRONT_DNS=\$(aws resourcegroupstaggingapi get-resources --tag-filters Key=Name,Values=web-cdn --resource-type-filters 'cloudfront' --region us-east-1 --query "ResourceTagMappingList[0].ResourceARN" --output text sed 's:.*/:::' xargs -l {} aws cloudfront get-distribution --id {} --query "Distribution.DomainName" --output text) curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/cloud.mp4?token=drm-it" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/fire.mp4?token=drm-it" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/mountain.mp4?token=drm-it" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/rain.mp4?token=drm-it" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLLOUDFRONT_DNS/media/sea.mp4?token=drm-it"</pre>
	2-5-A <u>정확히 일치</u>	403 403 403 403 403

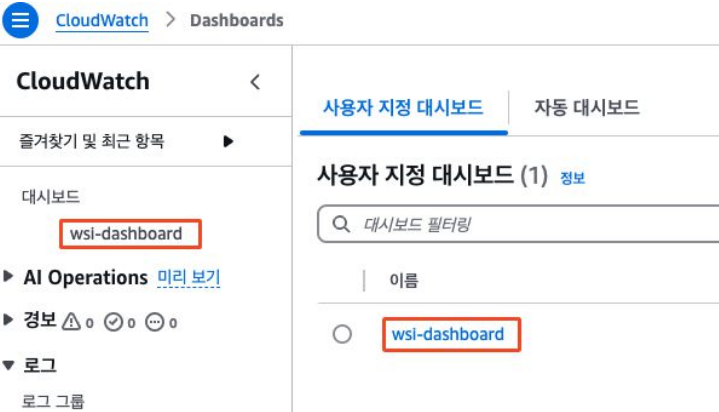
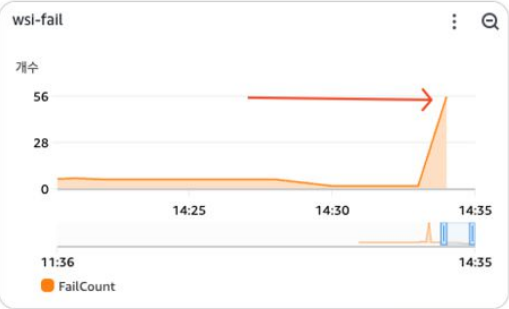

2-6	<p>2-6-A (명령어 입력)</p>	<pre> CLOUDFRONT_DNS=\$(aws resourcegroupstaggingapi get-resources --tag-filters Key=Name,Values=web-cdn --resource-type-filters 'cloudfront' --region us-east-1 --query "ResourceTagMappingList[0].ResourceARN" --output text sed 's:.*/:::' xargs -l {} aws cloudfront get-distribution --id {} --query "Distribution.DomainName" --output text) curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLOUDFRONT_DNS/media/cloud.mp4?token=drm-cloud" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLOUDFRONT_DNS/media/fire.mp4?token=drm-cloud" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLOUDFRONT_DNS/media/mountain.mp4?token=drm-cloud" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLOUDFRONT_DNS/media/rain.mp4?token=drm-cloud" curl -s -o /dev/null -w "%{http_code}\n" "https://\$CLOUDFRONT_DNS/media/sea.mp4?token=drm-cloud" </pre>
	<p>2-6-A <u>정확히 일치</u></p>	<p>200</p> <p>200</p> <p>200</p> <p>200</p> <p>200</p>

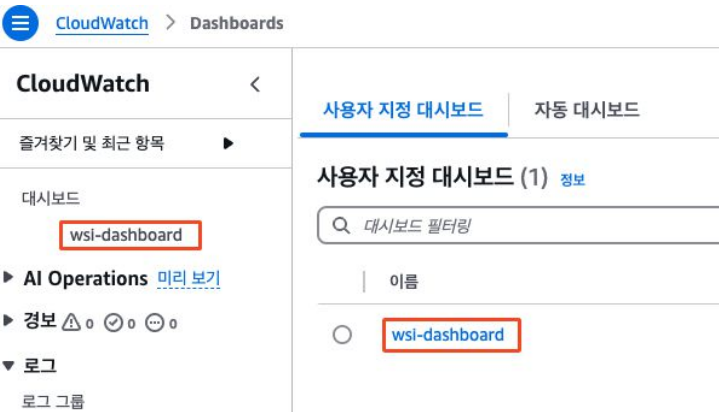
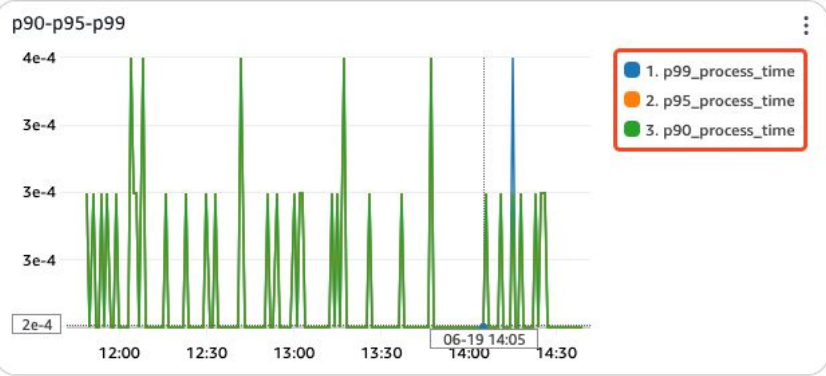
순번	채점 항목	
3-1	3-1-A (명령어 입력)	<pre>aws ec2 describe-vpcs \ --filters "Name=tag:Name,Values=ws-i-vpc" \ --query "Vpcs[*].{Name:Tags[?Key=='Name']}[[0].Value}" \ --output table</pre>
	3-1-A (예상 출력) <u>정확히 일치</u>	<pre>----- DescribeVpcs +-----+ Name +-----+ ws-i-vpc +-----+</pre>
3-2	3-2-A (명령어 입력)	<pre>filters=() for name in "\${SUBNET_NAMES[@]"; do filters+=("Name=tag:Name,Values=\$name") done aws ec2 describe-subnets \ --filters "\${filters[@]}" \ --query "Subnets[].{Name:Tags[?Key=='Name']}[[0].Value}" \ --output table</pre>
	3-2-A <u>순서 상관 없이 출력</u>	<pre>----- DescribeSubnets +-----+ Name +-----+ ws-i-pub-sn-a ws-i-priv-sn-c ws-i-pub-sn-c ws-i-priv-sn-a +-----+</pre>

순번	채점 항목	
3-3	3-3-A (명령어 입력)	<pre>aws ecs list-clusters \ --query "clusterArns[?contains(@, '\${wsi-app-cluster}')]\" \ --output text \ xargs -n1 basename \ jq -R -s 'split("\n")[:-1] map({Name: .})'</pre>
	3-3-A (예상 출력) <u>정확히 일치</u>	<pre>[{ "Name": "wsi-app-cluster" }]</pre>
3-4	3-4-A (명령어 입력)	<pre>aws cloudwatch list-dashboards \ --region "ap-northeast-1" \ --query "DashboardEntries[?DashboardName=='wsi-dashboard'] . [DashboardName]" \ --output table</pre>
	3-4-A <u>정확히 일치</u>	<pre>----- ListDashboards +-----+ wsi—dashboard +-----+</pre>

순번	채점 항목	
3-5	3-5-A (명령어 입력)	<pre>for i in {1..50} do curl -X GET http://<접근가능한 엔드포인트>/healthcheck done</pre>
	3-5-B <u>브라우저에서 진행</u>	<p>CloudWatch DashBoard로 이동 후 wsi-dashboard 클릭</p> 
	3-5-B <u>결과 확인</u>	<p>아래와 같이 5분내에 wsi-success Widget이 개수가 50 이상인지 확인</p>  <p>아래와 같이 wsi-SLI이 시스템 가용률이 100인지 확인</p> 

순번	채점 항목	
3-6	<p>3-6-A</p> <p><u>브라우저에서 실행</u></p>	<p>ECS로 이동 후 “wsi-app-cluster” page로 이동 후 wsi-app-service로 이동</p>  <p>구성 및 네트워킹 탭 클릭 후 네트워크 구성 항목에 보안 그룹 ID 클릭</p>  <p>아래와 같이 모든 인바운드 규칙 삭제</p> 
	<p>3-6-B</p> <p>(명령어 입력)</p>	<pre>for i in {1..50} do curl -X GET http://<접근가능한 엔드포인트>/healthcheck done</pre>

순번	채점 항목	
3-6	3-6-B (명령어 입력)	<p>Cloudwatch Dashboard로 이동 후 wsi-dashboard 클릭</p> 
	3-6-C <u>결과 확인</u>	<p>아래와 같이 5분내에 wsi-fail Widget이 개수가 50 이상인지 확인</p>  <p>아래와 같이 wsi-SLI이 시스템 가용률이 50(±1)인지 확인</p> 

순번	채점 항목	
	<p>3-7-A <u>브라우저에서 실행</u></p>	<p>Cloudwatch Dashboard로 이동 후 wsi-dashboard 클릭</p> 
<p>3-7</p>	<p>3-7-A <u>결과 확인</u> (순서 무관)</p>	<p>아래와 같이 p90-p95-99가 아래와 같이 출력되었는지 확인</p> 

순번	채점 항목																																													
3-8	3-8-A 브라우저에서 실행	<div>Cloudwatch log Insights로 이동 후 wsi-query 불러오기</div> <div><div><div>▼ 로그</div><div>로그 그룹</div><div>로그 이상 항목</div><div>라이브 테일</div><div>Logs Insights <small>신규</small></div><div>Contributor Insights</div></div><div><div>쿼리</div><div>저장된 쿼리 <small>자세히 알아보기</small></div><div><div>Q wsi</div><div>×</div></div><div>쿼리 생성</div><div>wsi-query - CWLI</div></div></div> <div>query 실행(추가 행동 없이 바로 쿼리 실행 버튼 클릭)</div> <div><div>로그 그룹 선택 기준</div><div>선택 기준</div><div><div>로그 그룹 이름</div><div>최대 50 로그 그룹 선택</div></div><div><div>/ecs/wsi-app-fargate-task</div><div>모두 지우기</div></div><div>Query generator</div><div><div>쿼리 실행</div><div>취소</div><div>저장</div><div>작업 ▼</div><div>기록</div></div><div>Logs Insights QL 쿼리는 최대 60분 동안 실행할 수 있습니다.</div></div>																																												
	3-8-A 결과 확인	<div>가장 최근 로그상 시간과 현재 시간과 ±□□초 이내 시간의 테이블 형태로 출력되었는지 확인 (파란 박스의 값이 달라도 됨)</div> <table><tr><th>#</th><th>date</th><th>time</th><th>src_ip</th><th>dst_ip</th><th>method</th><th>path</th><th>status</th><th>duration</th></tr><tr><td>▶ 1</td><td>2025/06/19</td><td>14:49:08</td><td>127.0.0.1</td><td>10.0.20.34</td><td>GET</td><td>/hello</td><td>404</td><td>0.0002</td></tr><tr><td>▶ 2</td><td>2025/06/19</td><td>14:48:38</td><td>127.0.0.1</td><td>10.0.20.34</td><td>GET</td><td>/hello</td><td>404</td><td>0.0003</td></tr><tr><td>▶ 3</td><td>2025/06/19</td><td>14:47:38</td><td>127.0.0.1</td><td>10.0.20.34</td><td>GET</td><td>/hello</td><td>404</td><td>0.0002</td></tr><tr><td>▶ 4</td><td>2025/06/19</td><td>14:47:08</td><td>127.0.0.1</td><td>10.0.20.34</td><td>GET</td><td>/hello</td><td>404</td><td>0.0002</td></tr></table>	#	date	time	src_ip	dst_ip	method	path	status	duration	▶ 1	2025/06/19	14:49:08	127.0.0.1	10.0.20.34	GET	/hello	404	0.0002	▶ 2	2025/06/19	14:48:38	127.0.0.1	10.0.20.34	GET	/hello	404	0.0003	▶ 3	2025/06/19	14:47:38	127.0.0.1	10.0.20.34	GET	/hello	404	0.0002	▶ 4	2025/06/19	14:47:08	127.0.0.1	10.0.20.34	GET	/hello	404
#	date	time	src_ip	dst_ip	method	path	status	duration																																						
▶ 1	2025/06/19	14:49:08	127.0.0.1	10.0.20.34	GET	/hello	404	0.0002																																						
▶ 2	2025/06/19	14:48:38	127.0.0.1	10.0.20.34	GET	/hello	404	0.0003																																						
▶ 3	2025/06/19	14:47:38	127.0.0.1	10.0.20.34	GET	/hello	404	0.0002																																						
▶ 4	2025/06/19	14:47:08	127.0.0.1	10.0.20.34	GET	/hello	404	0.0002																																						

순번	채점 항목	
4-1	4-1-A (명령어 입력)	aws ec2 describe-instances \ --filters "Name=tag:Name,Values=xxe-server" \ --query "Reservations[].Instances[0].Name:Tags[?Key=='Name'] [0].Value, Type: InstanceType}" \ --output text
	4-1-A <u>정확히 일치</u>	"xe-server" "t3.micro"
4-2	4-2-A (명령어 입력)	aws elbv2 describe-load-balancers \ --query "LoadBalancers[?contains(LoadBalancerName, 'xe-alb')].[LoadBalancerName, Type]" \ --output text
	4-2-A <u>정확히 일치</u>	"xe-alb" "application"
4-3	4-3-A (명령어 입력)	aws wafv2 list-web-acls \ --scope REGIONAL \ --query "WebACLs[?Name=='xe-waf'].Name" \ --output text
	4-3-A <u>정확히 일치</u>	"xe-waf "
4-4	4-4-A (명령어 입력)	ENDPOINT=\$(aws elbv2 describe-load-balancers \ --names xe-alb \ --query "LoadBalancers[0].DNSName" \ --output text) curl -X POST http://\$ENDPOINT/parse \ -H "Content-Type: application/x-www-form-urlencoded" \ --data-urlencode '\$xml=<?xml version="1.0"?>\n<user>\n<username>testuser</username>\n<email>test@example.com</email>\n</user>'
4-4	4-4-A <u>정확히 일치</u>	<pre><user><username>testuser</username><email>test@example.com</email></user></pre>

순번	채점 항목	
4-5	4-5-A (명령어 입력)	<pre> ENDPOINT=\$(aws elbv2 describe-load-balancers \ --names xxe-alb \ --query "LoadBalancers[0].DNSName" \ --output text) curl -X POST http://\$ENDPOINT/parse \ -H "Content-Type: application/x-www-form-urlencoded" \ --data-urlencode '\$xml=<?xml version="1.0"?>\n<!DOCTYPE root [\n<!ENTITY xxe SYSTEM "http://169.254.169.254/latest/meta-data/iam/">\n]>\n<root><data>&xxe;</d ata></root>' </pre>
	4-5-A <u>정확히 일치</u>	403 Forbidden error
4-6	4-6-A (명령어 입력)	<pre> ENDPOINT=\$(aws elbv2 describe-load-balancers \ --names xxe-alb \ --query "LoadBalancers[0].DNSName" \ --output text) curl -X POST http://\$ENDPOINT/parse \ -H "Content-Type: application/x-www-form-urlencoded" \ --data-urlencode '\$xml=<?xml version="1.0"?>\n<!DOCTYPE lolz [\n<!ENTITY lol "lol">\n<!ENTITY lol1 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">\n<!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;">\n<!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;">\n]>\n<roo t><boom>&lol3;</boom></root>' </pre>
	4-6-A <u>정확히 일치</u>	403 Forbidden error

순번	채점 항목	
4-7	4-7-A (명령어 입력)	<pre> ENDPOINT=\$(aws elbv2 describe-load-balancers \ --names xxe-alb \ --query "LoadBalancers[0].DNSName" \ --output text) curl -X POST http://\$ENDPOINT/parse \ -H "Content-Type: application/x-www-form-urlencoded" \ --data-urlencode '\$xml=<?xml version="1.0"?>\n<!DOCTYPE root [\n<!ENTITY xxe SYSTEM "file:///etc/passwd">\n]>\n<root><data>&xxe;</data></root> '</pre>
	4-7-A (예상 출력)	403 Forbidden error