ST-01 | FACTORY SAFETY MONITORING

Aegispi

엣지 클러스터 · AWS Cloud 기반

IoT · AI 하이브리드 실시간 관제 플랫폼

- SYSTEM OK
- 3 FACTORIES
- TEAM 일터방패 · 2026.06

INTRODUCTION | Members

일터방패 팀원 소개

Aegis-Pi 시스템 설계의 주역을 소개합니다.

팀장

김민수

@gitminsoo

역할

라즈베리파이 K3s 클러스터 설계
Management VPC 설계
배포파이프라인을 위한 EKS 클러스터 구축
온프레미스-클라우드 간 배포망 설계
IoT core를 이용한 데이터 파이프라인 설계
Bedrock을 이용한 보고서 생성 AX 전환

Github

https://github.com/gitminsoo

팀원

김종원

@Jjong-03

역할

dashboard VPC 설계
Cloud Front 를 이용한 Front end 설계
ECS를 이용한 Backend 설계
데이터 모니터링을 위한 조회 파이프라인 설계
Cognito 를 이용한 인증/인가 설계
Bedrock을 이용한 AI 챗봇 설계

Github

https://github.com/JJong-03

INDEX | AGENDA

발표 순서

01

시스템 개요

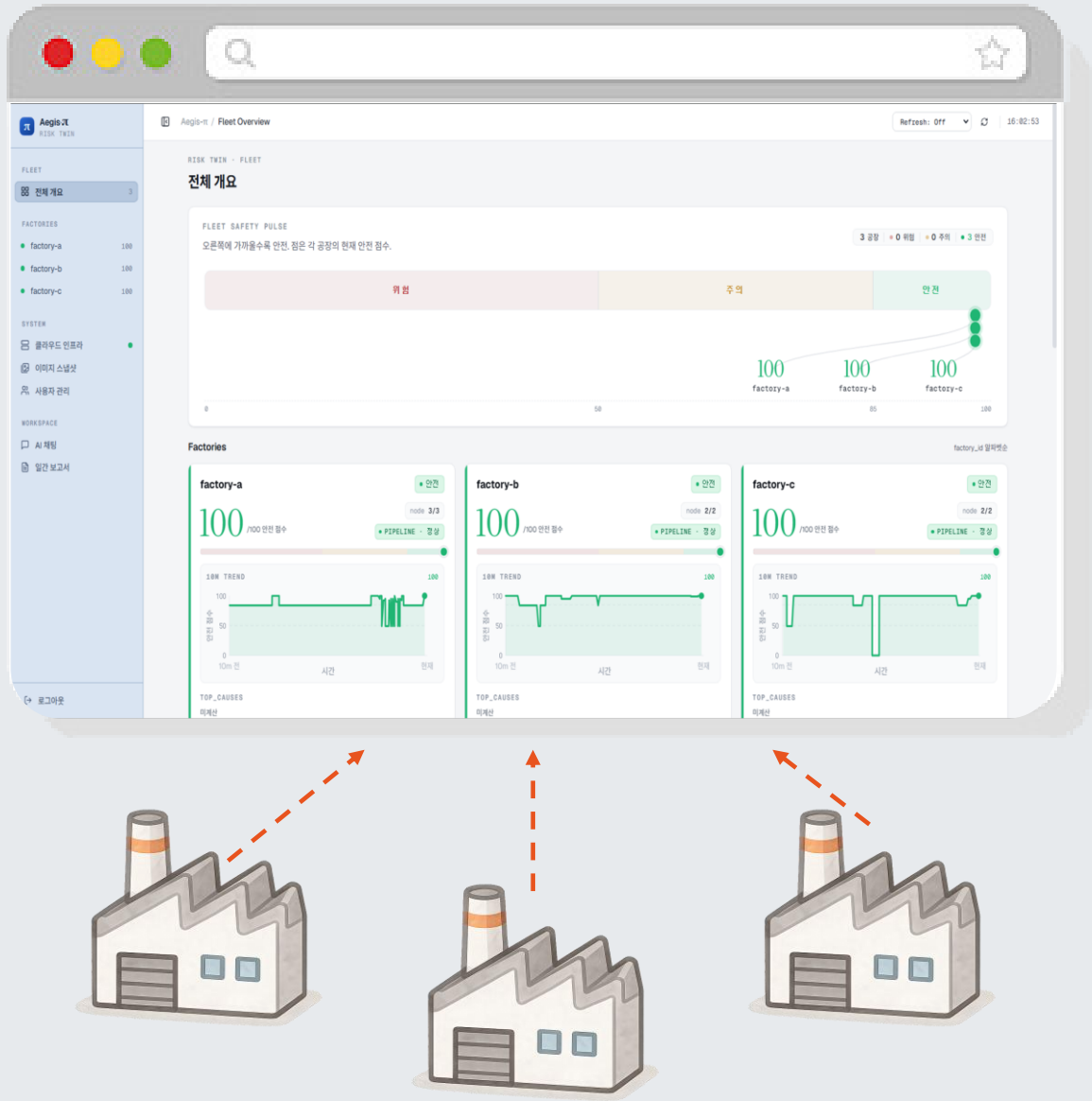
멀티 공장의 센서·카메라 데이터를 엣지에서 수집하고
AI로 위험을 탐지해 클라우드 대시보드에서 한눈에 관제합니다.

Aegis-Pi Risk Twin 개요

멀티 공장 Risk Twin 플랫폼 Aegis-Pi 를 소개합니다.

Aegis-Pi Risk Twin은 방패를 의미하는 Aegis와
Raspberry Pi의 Pi를 결합한 이름으로
공장 현장의 위험을 조기에 감지하고
중앙에서 관제하기 위한
멀티 공장 Risk Twin 플랫폼입니다.

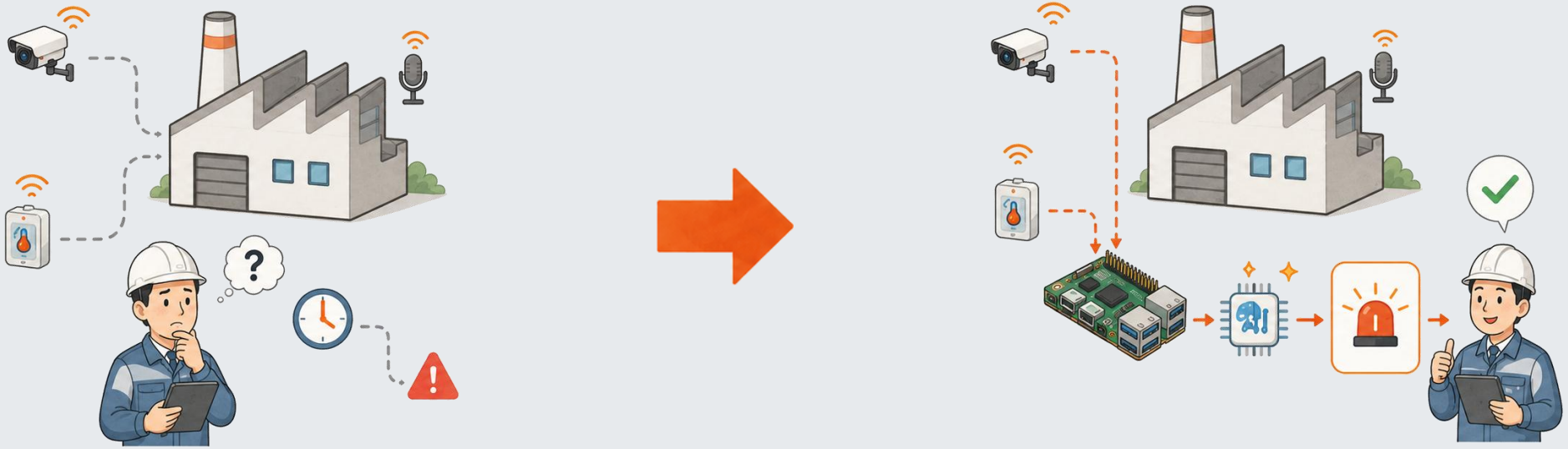
Aegis-Pi는
온프레미스 엣지 구성, **실시간 통합 대시보드**, **AX 경험**
3가지를 핵심 가치로 합니다.



OVERVIEW | ON-PREMISE EDGE

핵심 가치 - 온프레미스 엣지 구성

사람이 놓치는 순간에도 AI가 감지하고, 장애가 와도 데이터는 보존된다

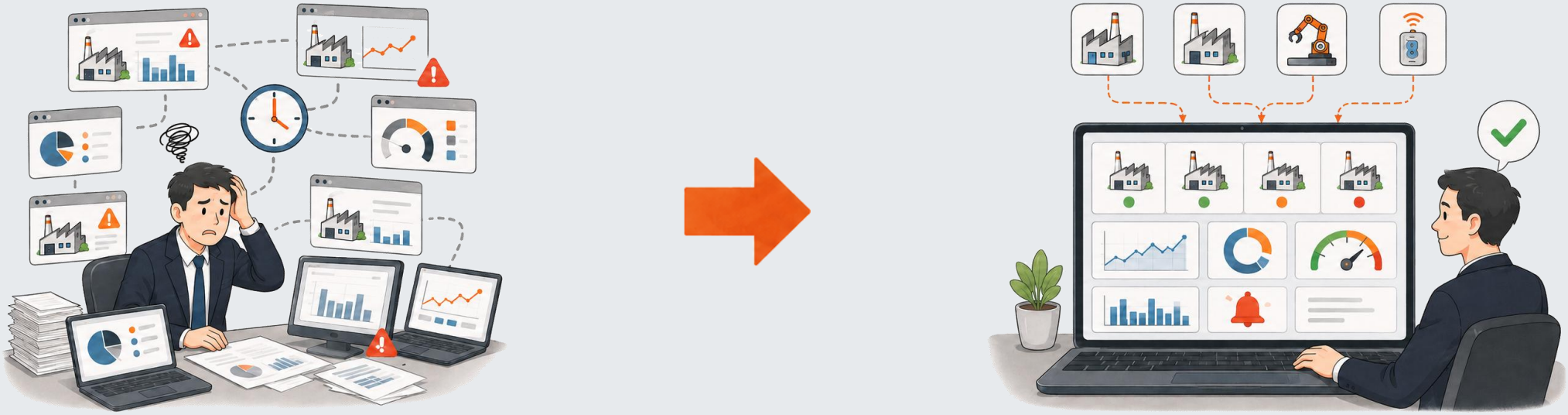


온프레미스 환경에서는 센서, 카메라, 마이크로
공장 상태를 수집하고
Raspberry Pi 기반 K3s 엣지 클러스터에서
YOLOv8n과 YAMNet 기반 AI 추론을 수행합니다.

OVERVIEW | DASHBOARD

핵심 가치 - 실시간 통합 대시보드

수치·사진·알림을 한 화면에서 실시간으로 확인



실시간 대시보드에서는 AWS 기반
데이터 수집·처리 파이프라인을 통해
공장별 위험 상태, 안전점수, 데이터 최신성,
클라우드 인프라 상태, 경보와 보고서를
한 화면에서 확인할 수 있습니다.

OVERVIEW | AX EXPIERENCE

핵심 가치 - AX 경험

AI가 매일 보고서를 만들고, 질문 하나로 원인을 찾는다

Bedrock 기반 일간 보고서 자동 생성과
공장 데이터 기반 AI Chat을 통해
수동 보고와 원인 분석 업무를 줄이고
공장 운영 전반에서 AI가 의사결정을 보조하는
AX(AI Transformation) 경험을 제공합니다.

OVERVIEW | CUSTOMER NEEDS

고객 요구사항

수동 보고와 지연 확인을 실시간 관제로 전환하는 방향

AS IS

- ✕ 수동 보고
- ✕ 보고 지연
- ✕ 확인 불가

TO BE

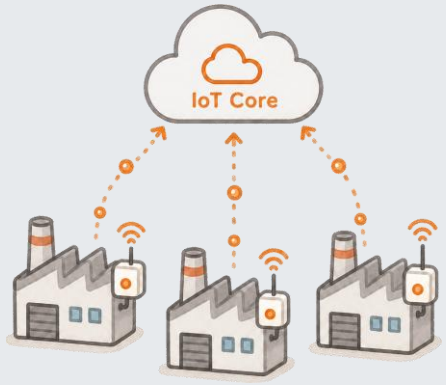
- ✓ 자동 탐지
- ✓ 중앙 수집
- ✓ 실시간 관제

OVERVIEW | DATA FLOW AT A GLANCE

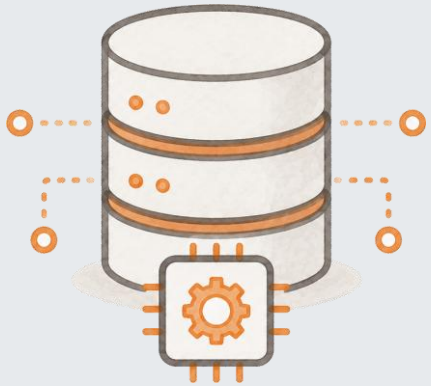
한눈에 보는 데이터 흐름

현장 탐지부터 클라우드 처리와 화면 표시까지 연결

멀티 공장
센서 + AI 탐지



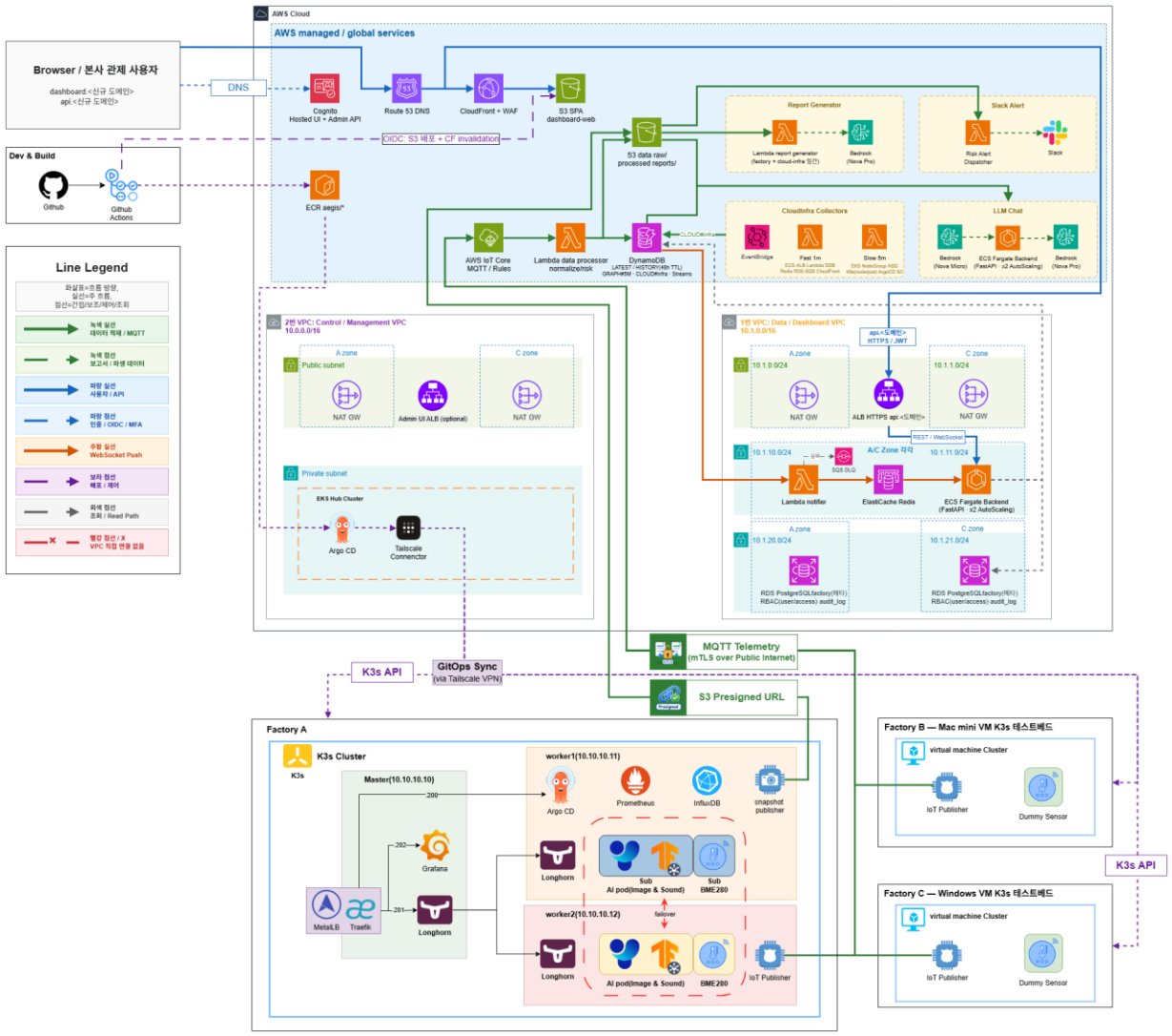
IoT Core
기반 데이터 수집



데이터
저장 및 처리

사용자
웹 대시보드

전체 시스템 아키텍처


AWS Cloud
AWS managed / global services
Browser / 본사 관제 사용자
DNS
Cognito Hosted UI + Admin API
Route 53 DNS
CloudFront + WAF
S3 SPA dashboard-web
OIDC: S3 배포 + CF invalidation
Dev & Build
Github
Github Actions
ECR aegis/*
S3 data raw/ processed reports/
Report Generator
Lambda report generator (factory + cloud-infra 일간)
Bedrock (Nova Pro)
Slack Alert
Risk Alert Dispatcher
Slack
CloudInfra Collectors
EventBridge
Fast 1m
Slow 5m
LLM Chat
Bedrock (Nova Micro)
ECS Fargate Backend (FastAPI · x2 AutoScaling)
Bedrock (Nova Pro)
AWS IoT Core MQTT / Rules
Lambda data processor normalize/risk
DynamoDB
Line Legend
2번 VPC: Control / Management VPC
10.0.0.0/16
A zone
C zone
Public subnet
NAT GW
Admin UI ALB (optional)
Private subnet
EKS Hub Cluster
Argo CD
Tailscale Connector
1번 VPC: Data / Dashboard VPC
10.1.0.0/16
api.<도메인>
HTTPS / JWT
10.1.0.0/24
10.1.1.0/24
ALB HTTPS api.<도메인>
REST / WebSocket
10.1.10.0/24
10.1.11.0/24
A/C Zone 각각
SQS DLQ
Lambda notifier
ElastiCache Redis
ECS Fargate Backend (FastAPI · x2 AutoScaling)
10.1.20.0/24
10.1.21.0/24
RDS PostgreSQL factory(메타) RBAC(user/access) audit_log
MQTT Telemetry
(mTLS over Public Internet)
S3 Presigned URL
K3s API
GitOps Sync
(via Tailscale VPN)
Factory A
K3s Cluster
K3s
Master(10.10.10.10)
200
202
201
Grafana
Longhorn
MetalLB
Traefik
worker1(10.10.10.11)
Argo CD
Prometheus
InfluxDB
snapshot publisher
Sub AI pod(Image & Sound)
Sub BME280
worker2(10.10.10.12)
failover
AI pod(Image & Sound)
BME280
IoT Publisher
Factory B — Mac mini VM K3s 테스트베드
virtual machine Cluster
IoT Publisher
Dummy Sensor
K3s API
Factory C — Windows VM K3s 테스트베드
virtual machine Cluster
IoT Publisher
Dummy Sensor

02

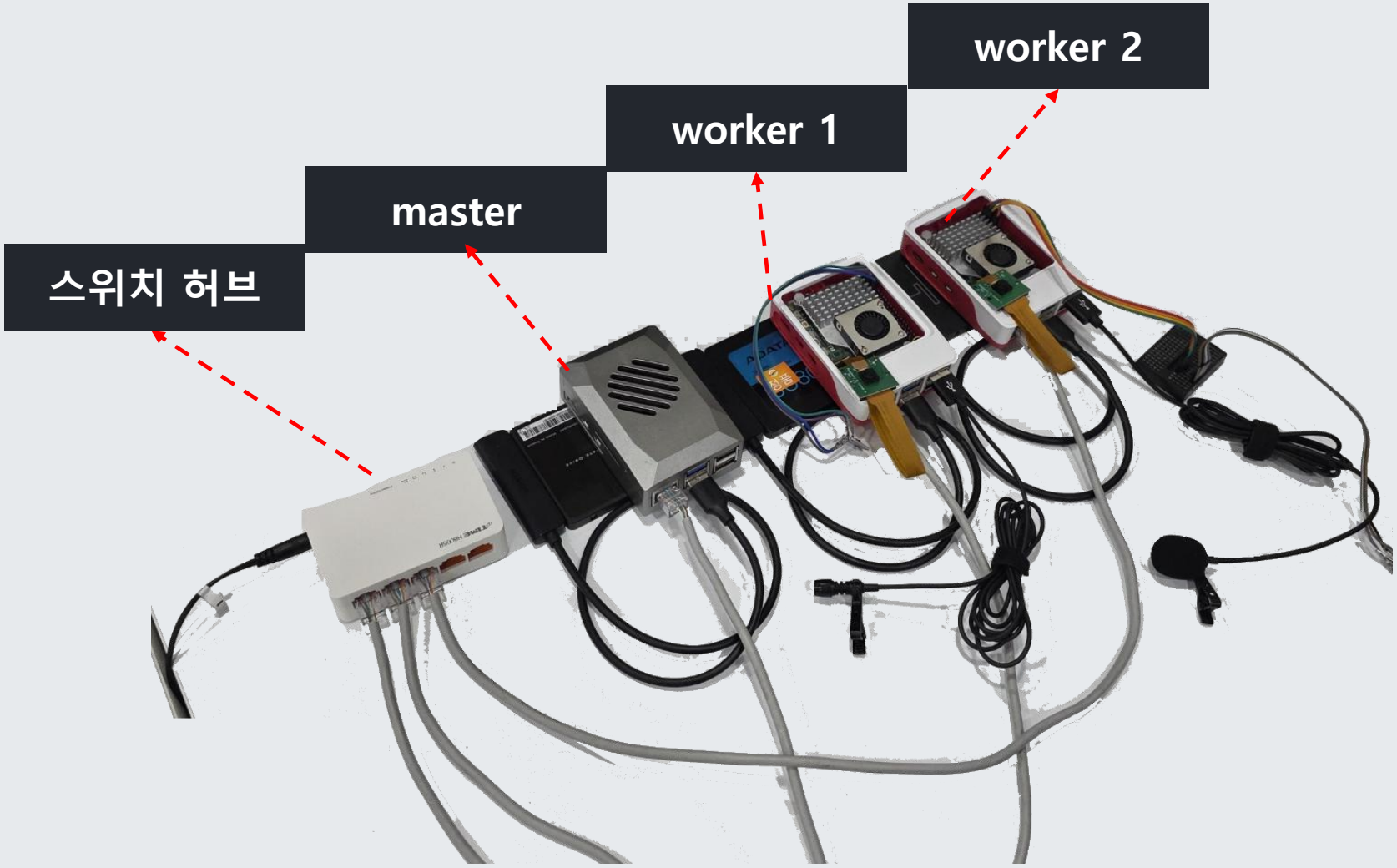
온프레미스 아키텍처 및 제어망 구성

Raspberry Pi 기반 엣지 클러스터에서 현장 데이터를 수집하고
중앙 GitOps 제어망으로 각 공장의 워크로드를 안정적으로 운영합니다.

EDGE | EDGE CLUSTER TOPOLOGY

공장 엣지 K3s 클러스터 물리 구성

Master와 Worker 노드로 구성한 현장 클러스터



EDGE | NODE SPECIFICATION

K3s 노드 사양과 저장소 구성

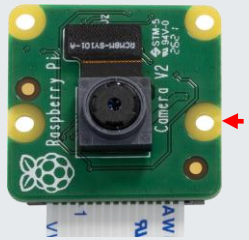
노드별 자원, 저장소, 역할 분담을 한눈에 정리

외장 SSD	
master	128 GB
worker1	128 GB
worker2	128 GB

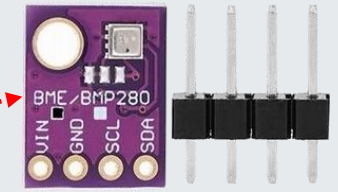
Rasberry pi 5	
master	4GB RAM
worker1	8GB RAM
worker2	8GB RAM

센서·카메라·마이크 배치

센서와 AI 입력 장치를 Worker 노드에 배치



Pi camera module	
worker1 worker2	500만 화소 YOLO v8n



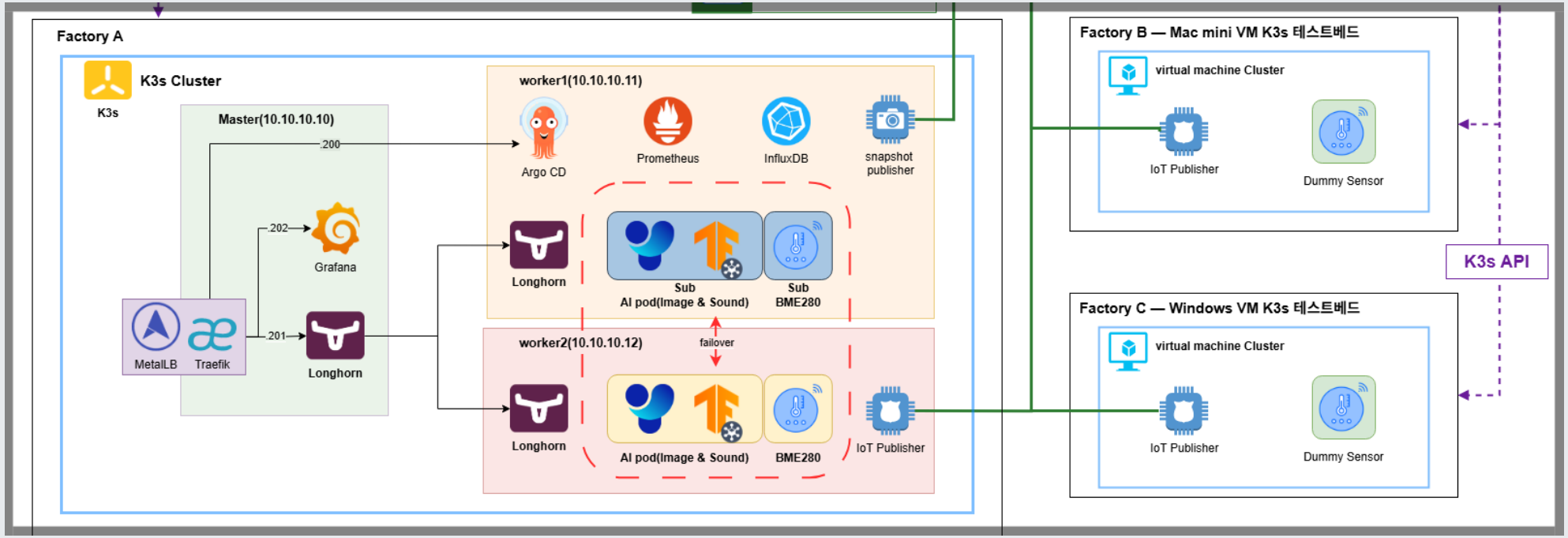
BME 280	
worker1 Worker2	온도 습도 기압

USB mic	
worker1 worker2	YAMnet

EDGE | EDGE CLUSTER ARCHITECTURE

공장 엣지 K3s 클러스터 아키텍처

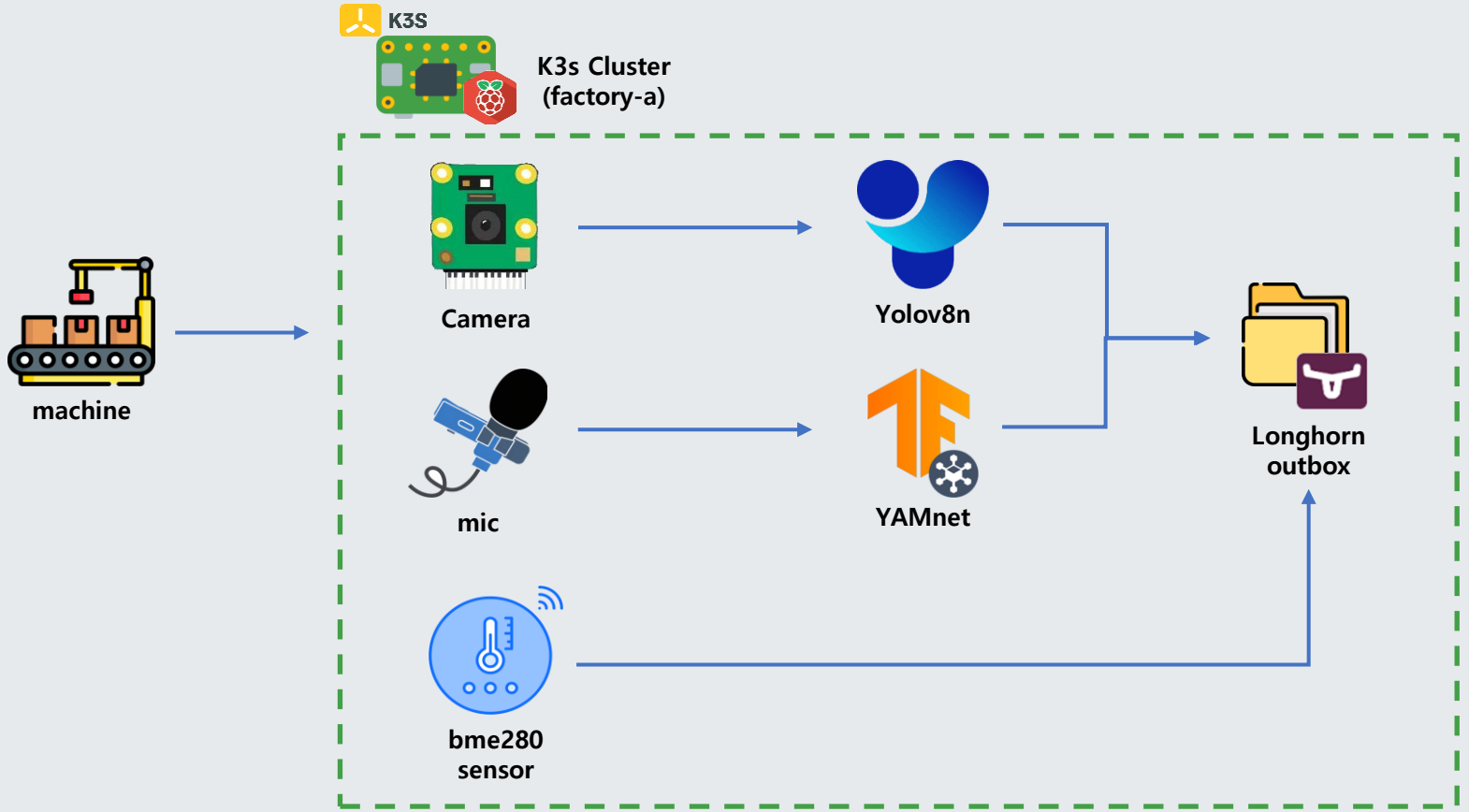
현장 장애 대응과 워크로드 재배치를 고려한 구조



EDGE | EDGE WORKLOAD FLOW

공장 엣지 K3s 워크로드 흐름

센서와 AI 결과를 로컬 버퍼로 모아 전송 준비



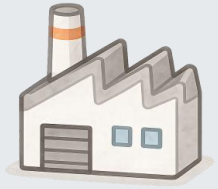
EDGE | EDGE CLUSTER FEATURES

공장 엣지 클러스터 핵심 기능

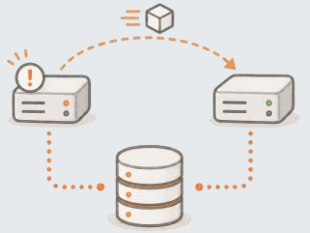
로컬 관측, AI 추론, 장애 대응 기능을 제공

Local observability

InfluxDB / Prometheus / Grafana 기반 로컬 관측
센서값 · AI 결과 · Pod/Node 상태를 현장에서 확인



Factory



Resilient Runtime

K3s scheduling + Longhorn PVC
Worker 장애 시 workload 재배치와 주요 로컬 상태 유지

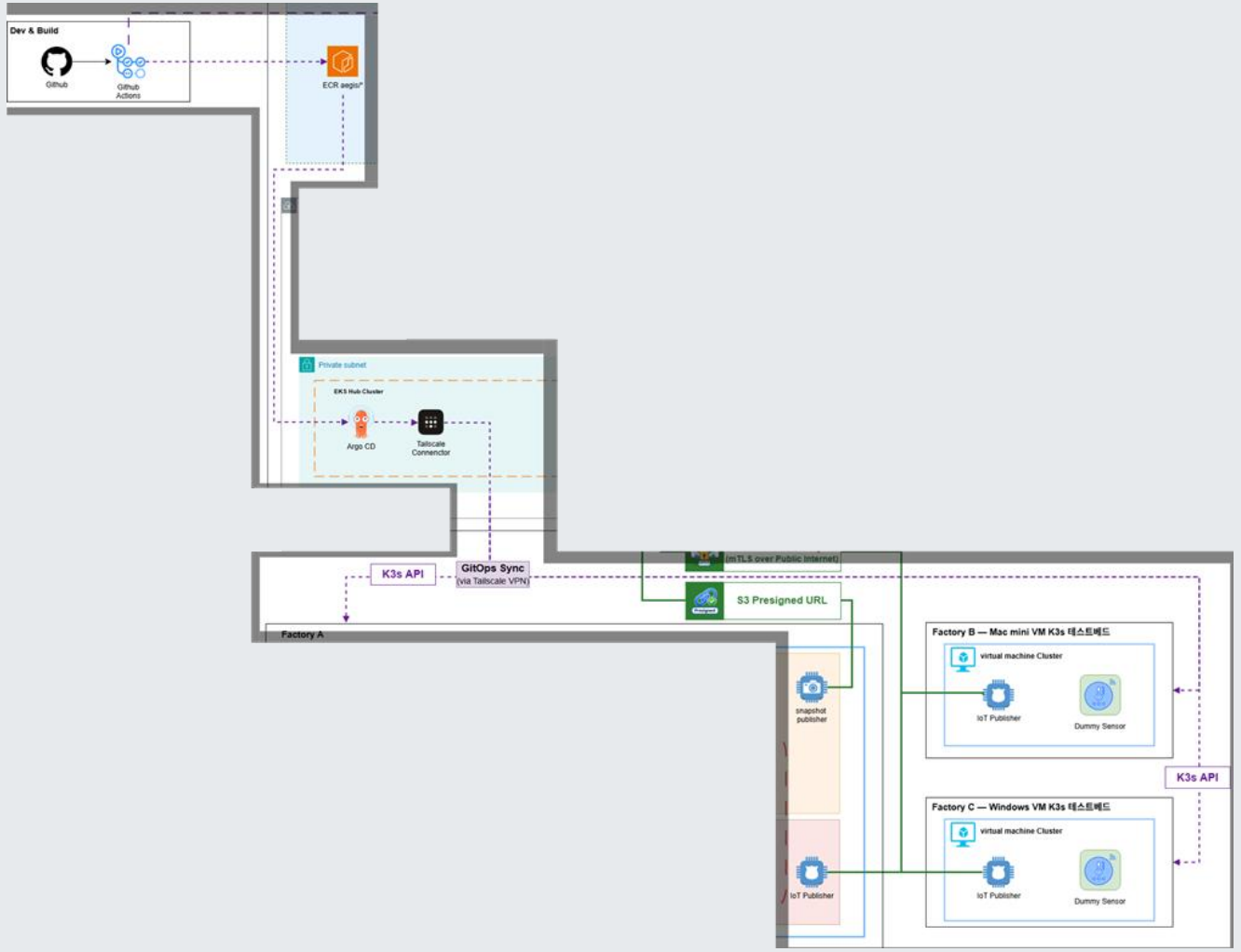
On-device AI

YOLOv8n / YAMNet을 worker node에서 직접 실행
Camera · Mic 이벤트를 현장에서 즉시 판단

EDGE | EDGE DEPLOYMENT ARCHITECTURE

공장 엣지 K3s 배포 아키텍처

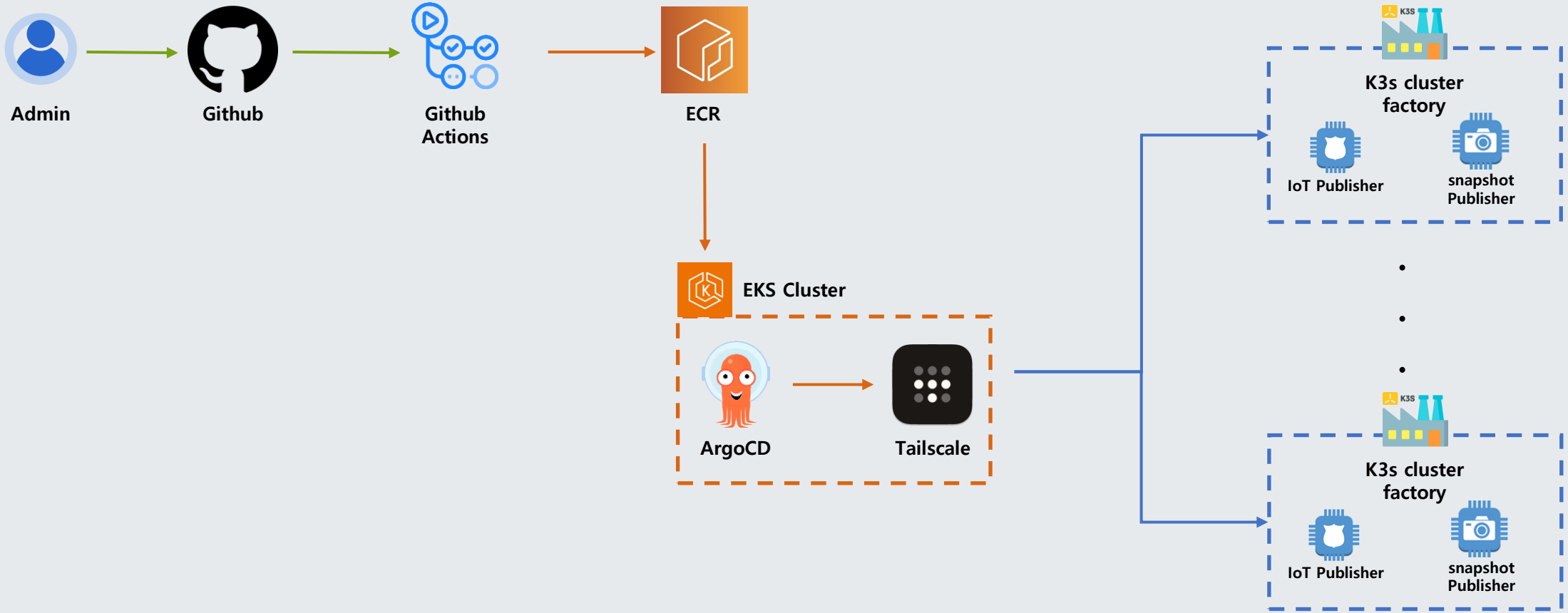
중앙 저장소 변경을 현장 워크로드로 배포하는 구조



EDGE | EDGE CLUSTER

공장 엣지 K3s 배포 흐름

코드 변경부터 이미지 빌드와 현장 반영까지 연결

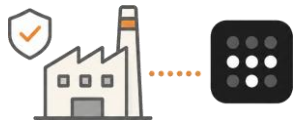


EDGE | TAILSCALE CONTROL MESH

EKS ArgoCD와 On-premise K3s 연결

공개 노출 없이 현장 클러스터를 제어하는 경로

01



Factory Tailnet 참여

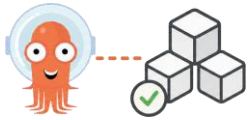
Factory master node
Tailnet 등록

02

Hub Egress 구성

Tailscale Operator
Factory별 egress 서비스

03



ArgoCD Cluster 등록

K3s cluster endpoint
Cluster Secret 등록

04

GitOps Sync 준비

Application set
Sync target

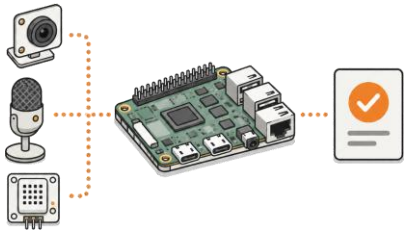
K3s API는 public open 없이 Tailnet 제어망 구성

데이터 전송 경로는 MQTT, PresignedURL로 별도 분리

EDGE | CI/CD PIPELINE

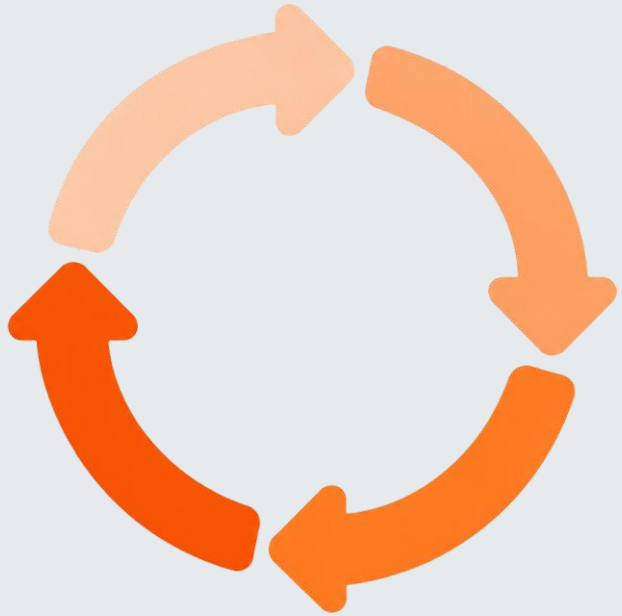
빌드 · 배포 파이프라인

코드 푸시 후 이미지 빌드와 저장 과정을 자동화



1. 현장 운영 적합성 검증

실제 센서·카메라·마이크 입력 확인
YOLOv8n / YAMNet 추론 결과 검증

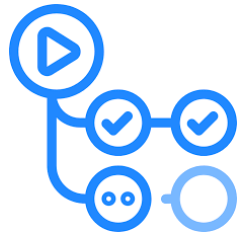


2. GitHub Push

검증된 Edge workload 코드 반영
main branch push로 workflow trigger

4. ECR Image Registry

Edge image를 ECR에 저장
sha · main · latest tag 관리



3. GitHub Actions

workflow syntax · build definition 확인
Docker build로 이미지 생성 검증

EDGE | HELM GITOPS DEPLOYMENT

Helm 기반 GitOps 배포 구성

공장별 설정을 실제 현장 리소스로 동기화하는 방식

GitOps source

Helm Chart
K3s resource template

Factory values
envs/factory-a/b/c

Image version
ECR repository · sha tag

Factory / Runtime

Deployment
IoT Publisher
snapshot Publisher

Pod
Factory별 workload

Storage
Longhorn PVC
hostPath

03

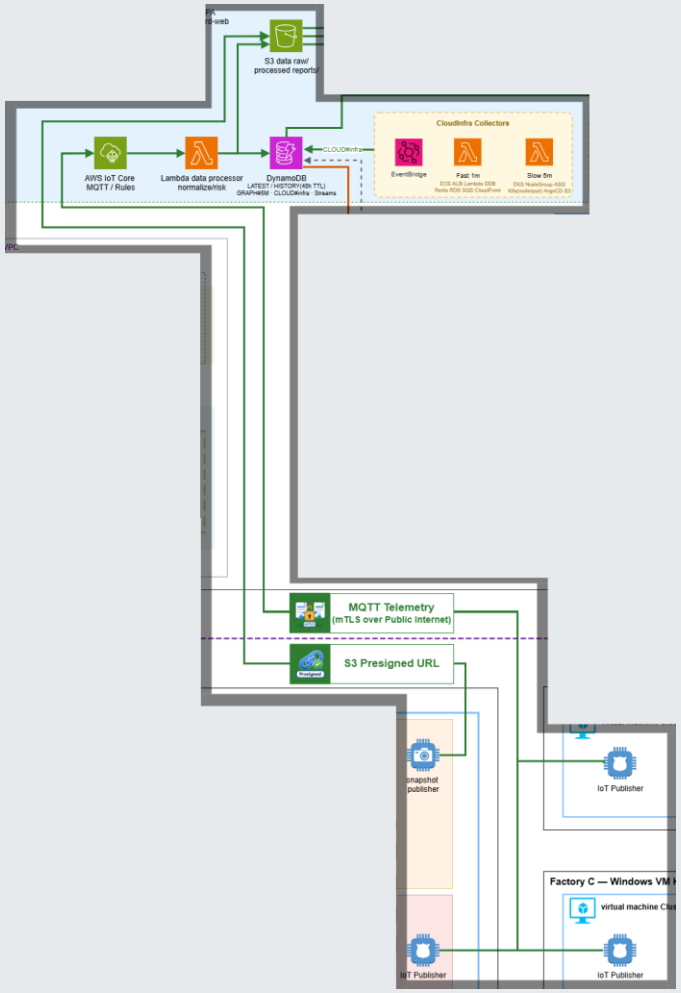
데이터 파이프라인

현장에서 생성된 센서·AI·인프라 데이터를 표준 형식으로 전송하고
클라우드에서 위험 점수와 조회 모델로 가공해 대시보드에 제공합니다.

DATA | DATA PIPELINE ARCHITECTURE

데이터 파이프라인 아키텍처

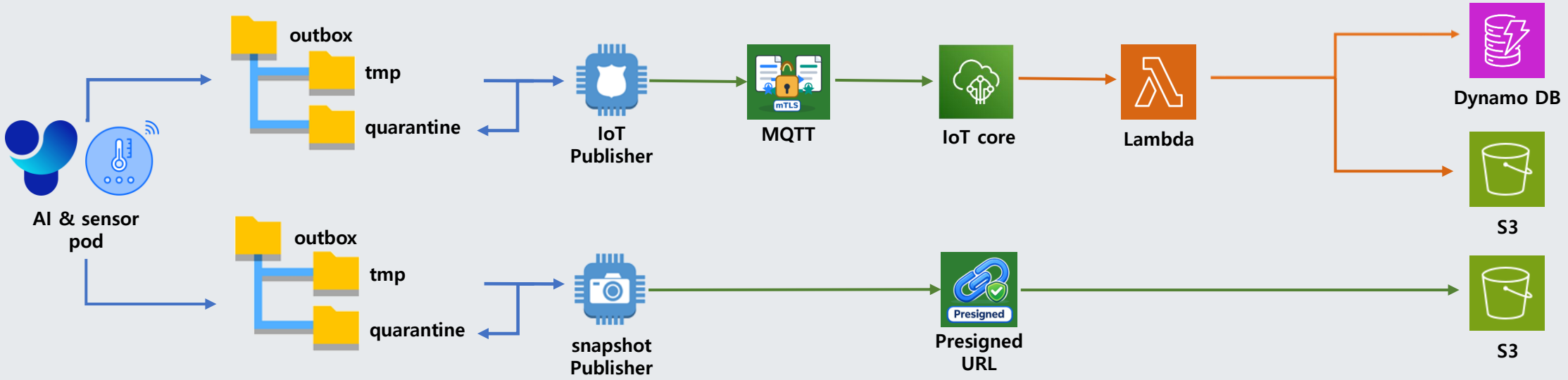
상태 데이터와 이미지 증빙을 분리해 처리하는 구조



DATA | INGESTION FLOW

데이터 파이프라인 흐름

로컬 버퍼에서 클라우드 저장소까지의 수집 경로



DATA | EDGE SOURCE TYPES

엣지 데이터 생성 · Source Type

센서, AI, 인프라 상태를 공통 형식으로 표준화

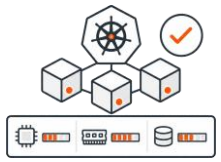
factory_state

센서 · AI 상태

3초 주기

Sensor
temperature · humidity · pressure

AI Result
fire · fall · bend · sound



infra_state

K3s · 노드 상태

20초 주기

Cluster
node ready · pod status

Resource
cpu · memory · disk

image_snapshot

이벤트 이미지 메타 데이터

이벤트 발생 시

Event Metadata
event_type · captured_at

S3 Reference
bucket · key · sha256

Presigned URL
PUT 300s · GET 900s

DATA | LOCAL OUTBOX BUFFER

Outbox 기반 로컬 버퍼

전송 실패와 재시도 상황에서도 현장 데이터를 보존

Edge workload
(factory, infra)

outbox/tmp

atomic rename

outbox

outbox/quarantine

IoT Publisher

검증 실패

JSON 오류, 필수 필드 누락,
schema 불일치

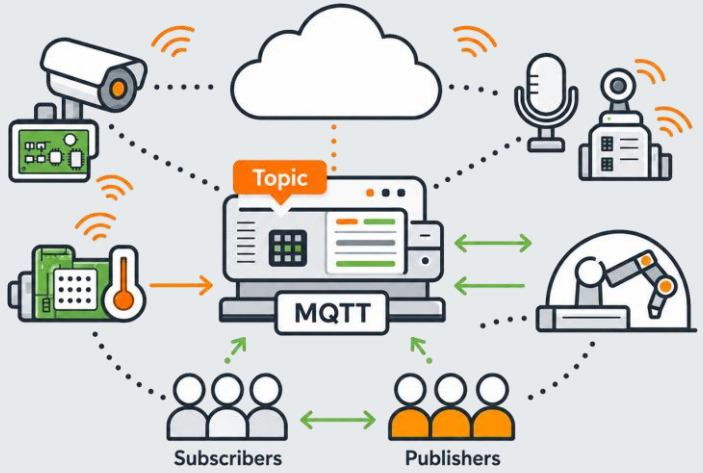
outbox 유지 → retry

Network 오류, MQTT 실패,
TLS/IoT Core 일시 장애

MQTT Publish와 IoT Core Ingestion

현장 데이터를 경량 메시징으로 클라우드에 전송

What is MQTT?



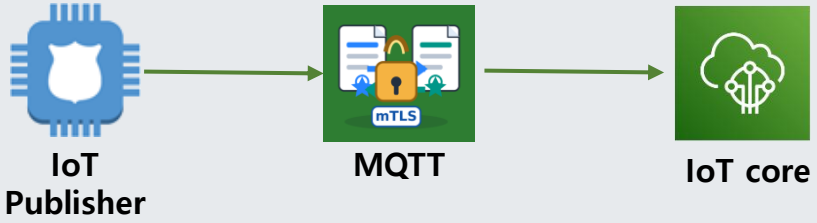
경량 Publish/Subscribe 메시징 프로토콜

Topic 기반 message routing

MQTT over TLS · mTLS 기반 보안 연결

저전력 device · 불안정한 network 환경 적합

AWS IoT Core MQTT broker로 publish/subscribe 처리



Topic 구조

aegis/{factory_id}/{source_type}

aegis/factory-a/factory_state
aegis/factory-a/infra_state
aegis/factory-a/image_snapshot
aegis/factory-b/factory_state
aegis/factory-c/infra_state

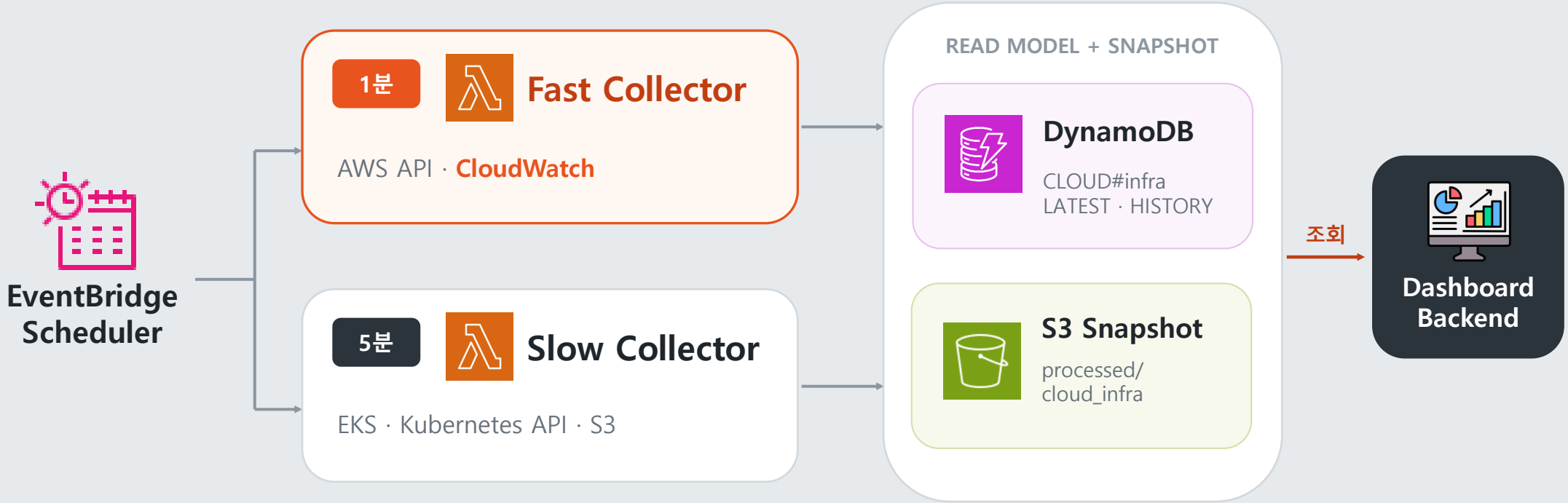
DataProcessor와 DynamoDB/S3 저장

정규화, 위험 점수 계산, 조회 모델 갱신을 수행

DATA | CLOUD INFRA COLLECTOR

클라우드 인프라 상태 수집

클라우드 구성요소 상태를 주기별로 수집해 저장



Backend = **DynamoDB Read Model 전용** · AWS API 직접 호출 없음

DATA | COLLECTION CADENCE

왜 1분과 5분으로 나눴는가

장애 민감도와 비용을 고려해 수집 주기를 분리

FAST **1분** 빠른 장애 감지

빠르게 영향이 번지는 Runtime

사용자 화면과 데이터 파이프라인 장애를 1분 단위로 먼저 감지

대상

ECS · ALB · Lambda / DDB · Redis · RDS / DLQ

출처

AWS Describe API · CloudWatch Metrics

갱신 중단 **3분** 후 stale

SLOW **5분** 관리 영역 상태

상태 변화가 느린 Management Plane

관리 plane과 저장 최신성은 5분 단위로 조회해 호출량을 절감

대상

EKS · NodeGroup · ASG / Kubernetes Node·Pod

출처

ArgoCD · S3 raw/processed 최신성

갱신 중단 **15분** 후 stale

동일한 **CLOUD#infra · LATEST** 안에서 **fast / slow** 필드만 독립 갱신

04

통합 대시보드 및 안전 관제

공장 상태·클라우드 인프라, 접근 권한, 증빙, 보고서, AI 질의를 하나의 관제 화면으로

DASHBOARD | OVERVIEW

통합 대시보드 한눈에 보기

공장·클라우드 상태와 증빙, 보고서, AI 질의를 권한 범위에 따라 통합 제공

관측

Observability

- 공장 모니터링
- 실시간 업데이트
- 클라우드 인프라 상태
- 이미지 스냅샷

AI 서비스

AI Service

- AI 채팅
- 일간 보고서

운영

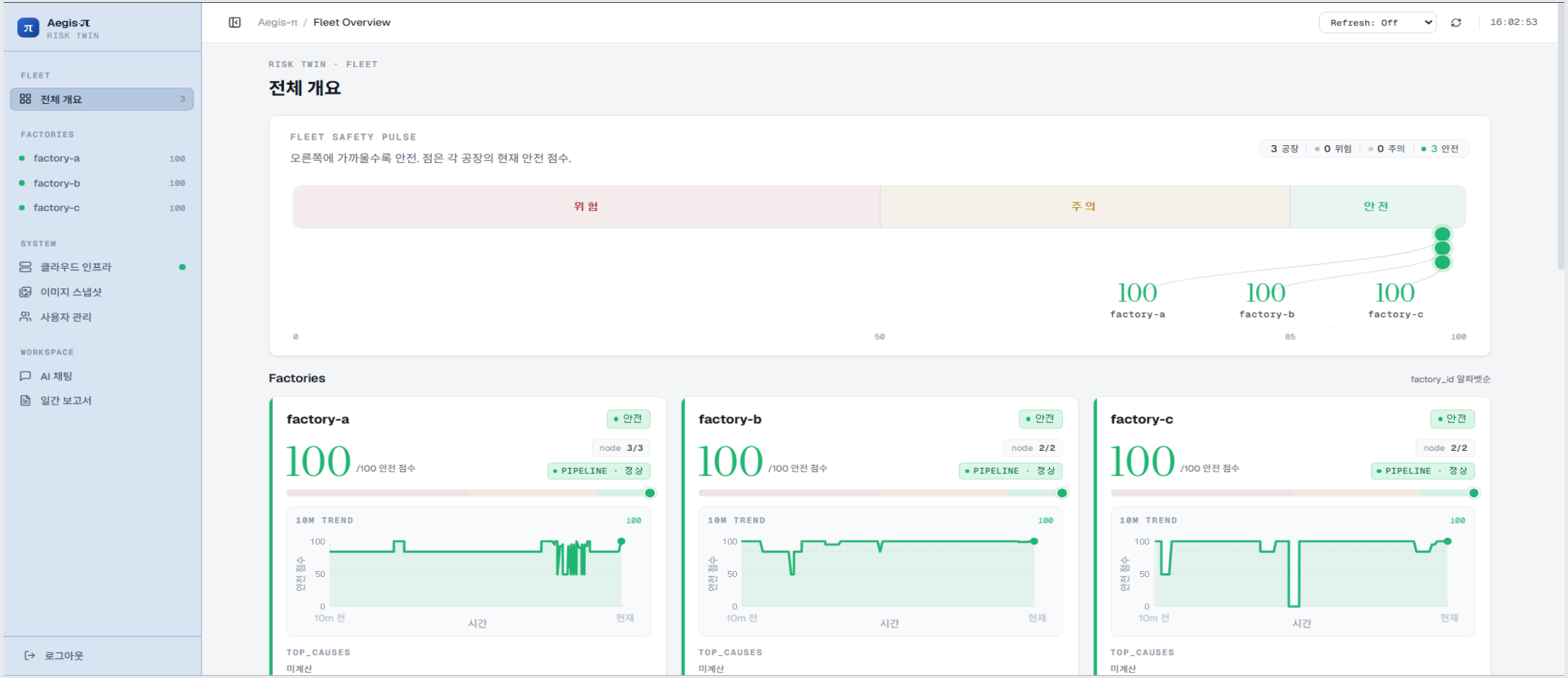
CORE BUSINESS

- 사용자 관리

DASHBOARD | FACTORY MONITORING

전체 공장 안전 상태 모니터링

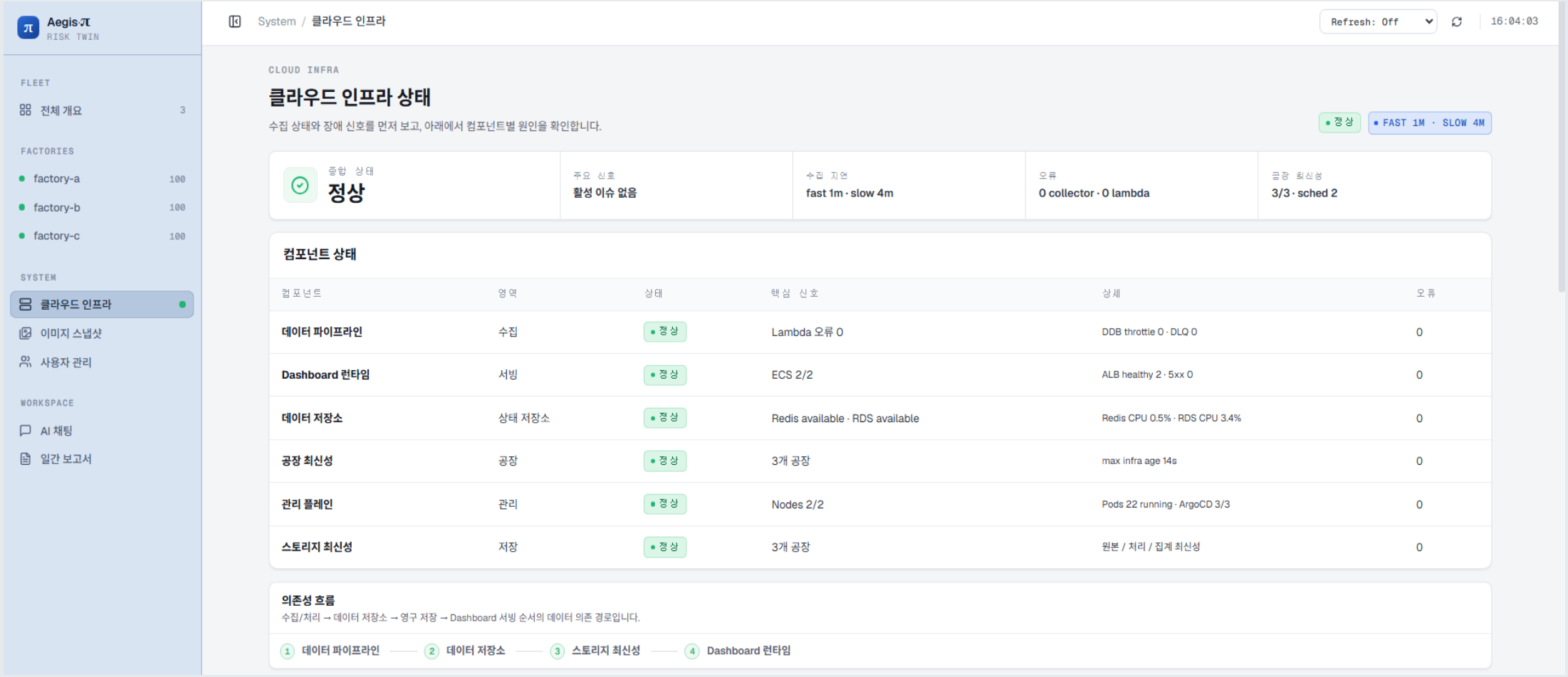
공장별 안전 점수·현재 상태·최근 추이를 한 화면에서 비교



DASHBOARD | CLOUD INFRA

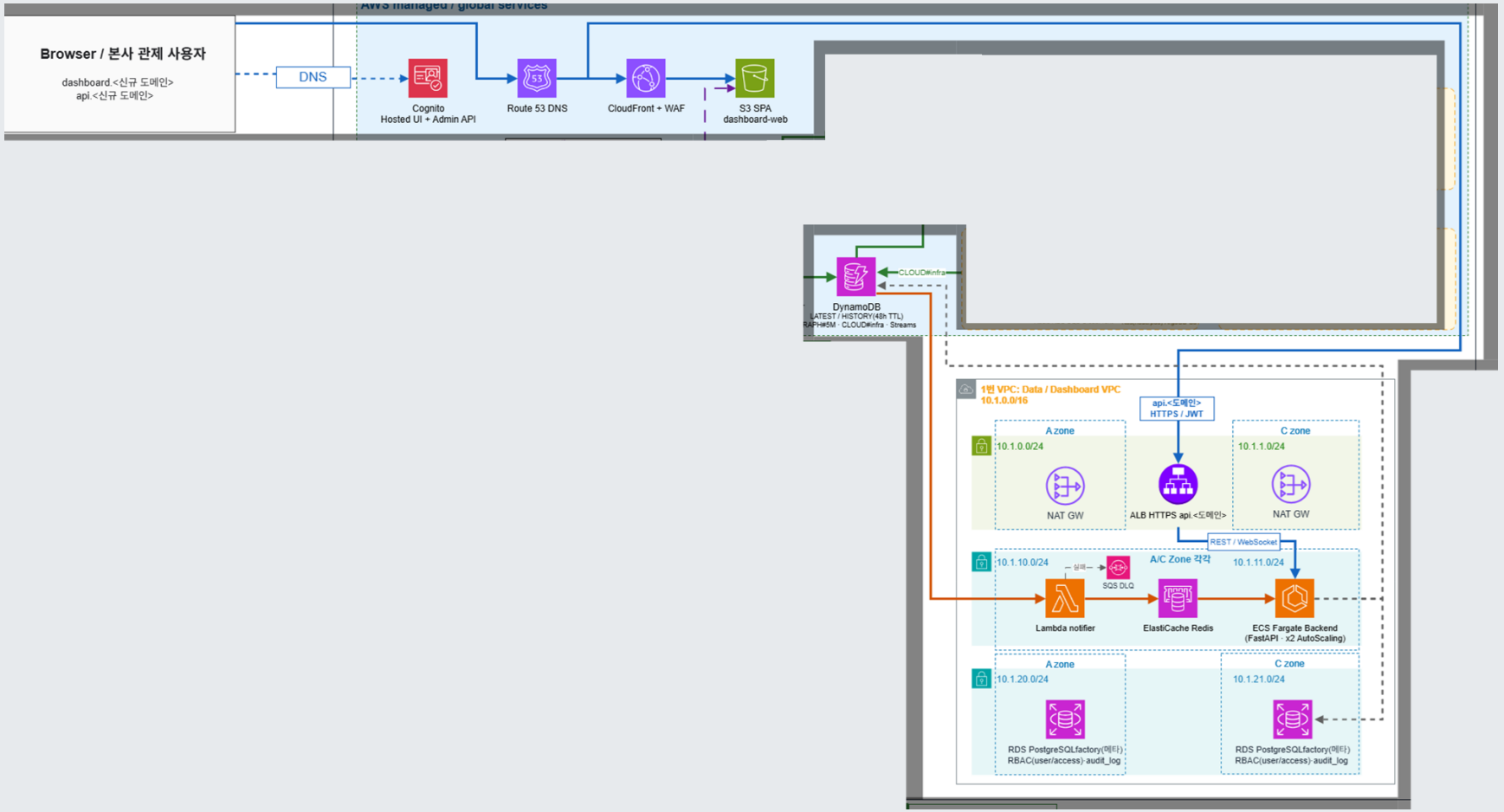
클라우드 인프라 상태 관제

수집·저장·서빙·관리 영역의 상태와 최신성을 1분·5분 수집 경로로 확인



전체 아키텍처 속 Dashboard 서비스 영역

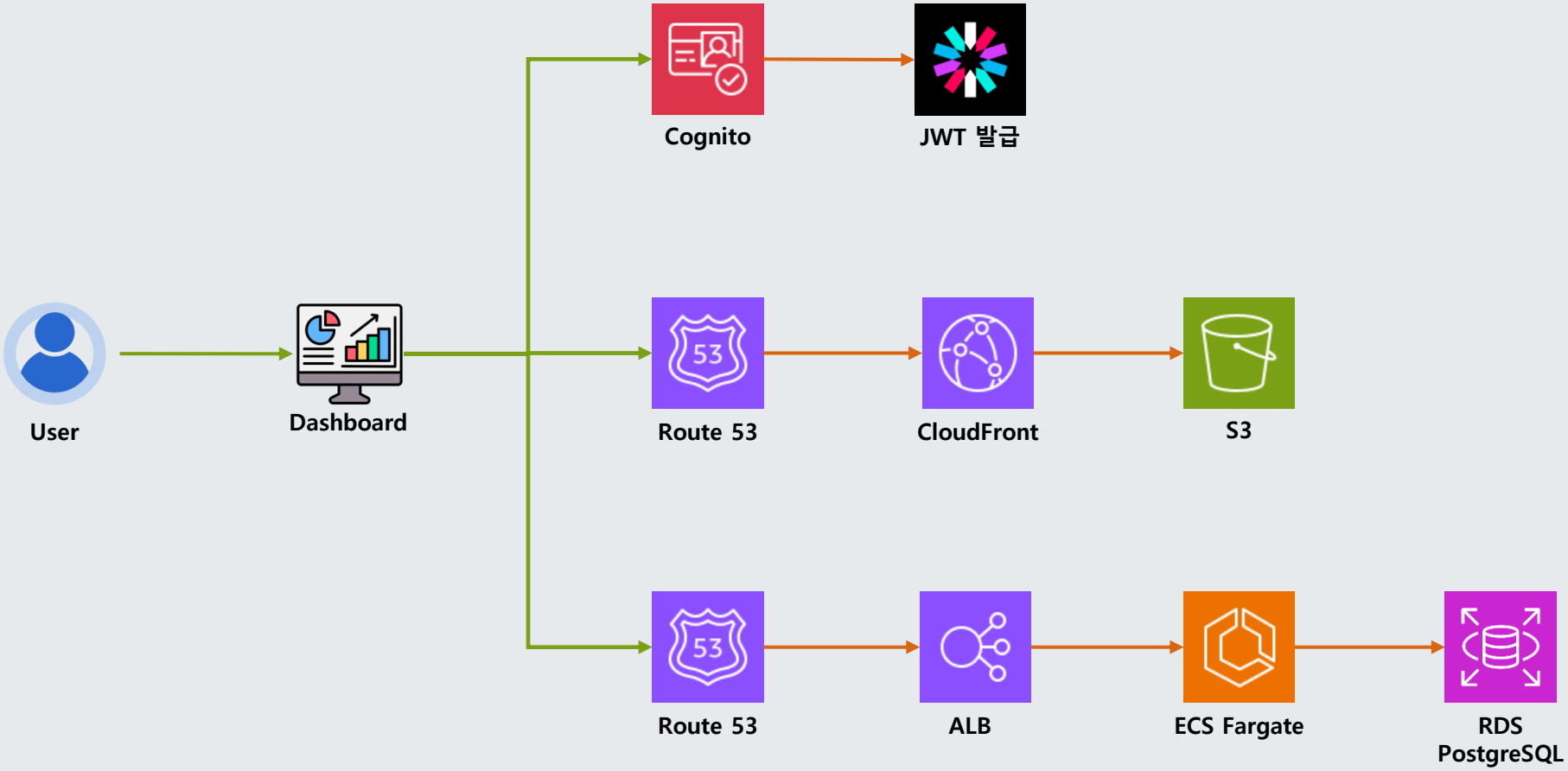
Web 전달·사용자 인증·Backend API·데이터 저장 계층을 하나의 흐름으로 연결



DASHBOARD | SERVICE FLOW

대시보드 서비스 아키텍처

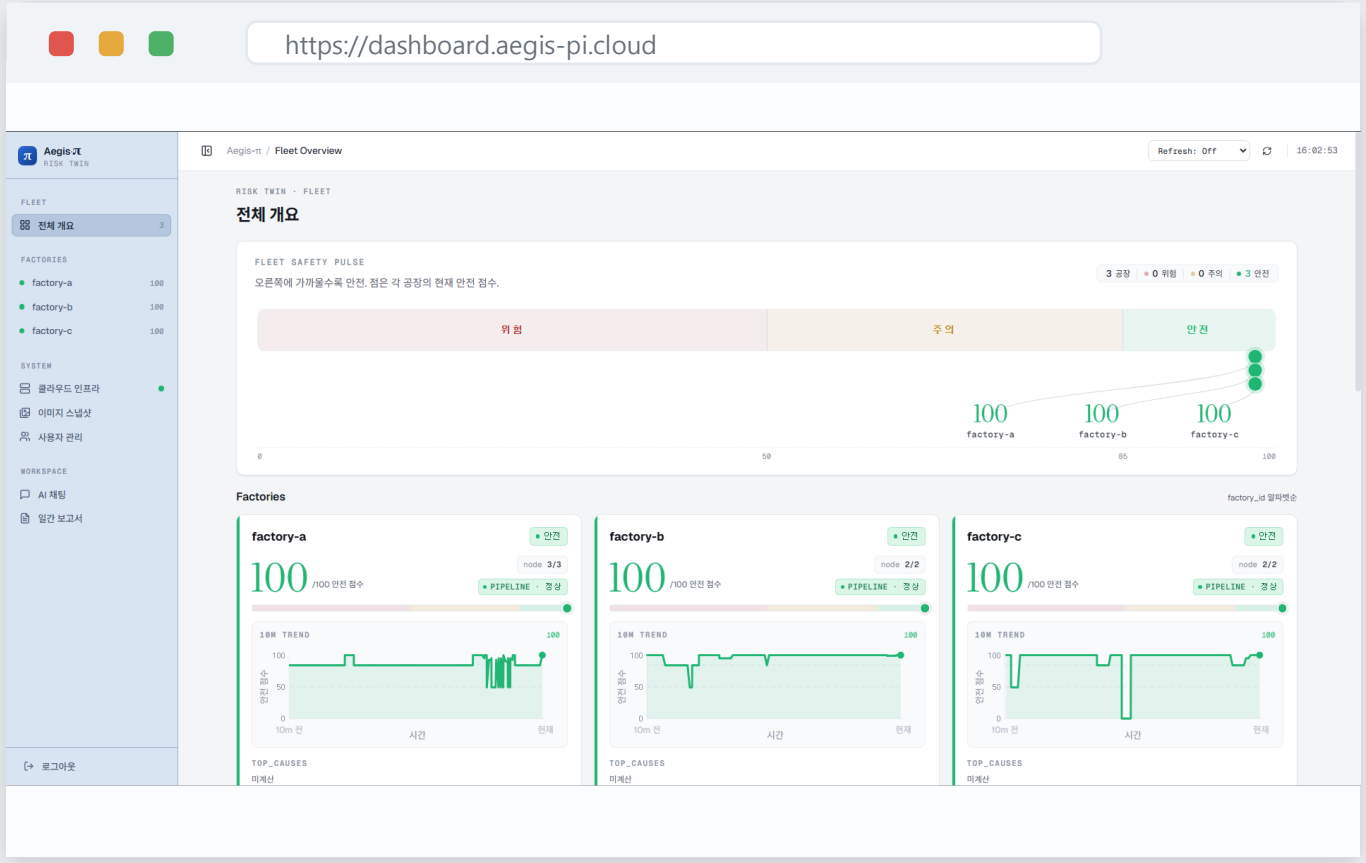
CloudFront·S3 정적 웹과 ALB·ECS API를 분리하고 Cognito JWT로 보호



DASHBOARD | FRONTEND

Frontend - React SPA 정적 배포

S3에 배포한 React SPA를 CloudFront로 제공하고 Cognito Hosted UI로 로그인



BUILT WITH

Route 53
도메인 · ACM HTTPS

CloudFront
CDN 캐싱 · 글로벌 배포

S3
React SPA 정적 호스팅

Cognito
로그인 리다이렉트 · JWT

React · Vite

GitHub Actions 자동 배포

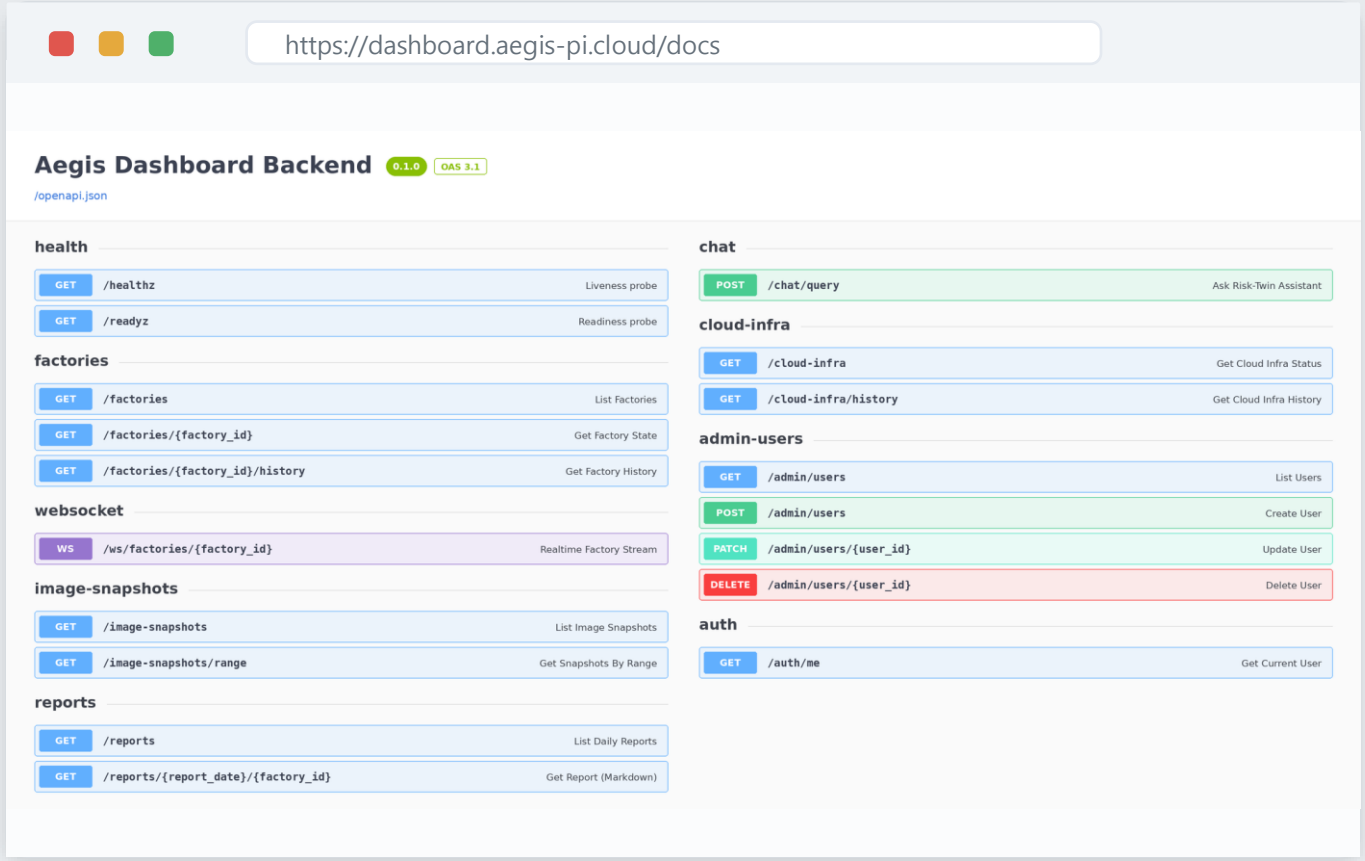
WebSocket 실시간

주요 관제 화면

DASHBOARD | BACKEND

Backend - FastAPI를 ECS Fargate로 운영

ALB 뒤의 ECS 2개 Task가 API를 처리하고 RDS·DynamoDB·S3를 역할별로 사용



BUILT WITH

ALB
HTTPS 진입점 · 트래픽 분산

ECS Fargate
컨테이너 2 Task · auto-scaling

RDS PostgreSQL
사용자 · 권한 DB

DynamoDB / S3
실시간 hot store / 문서·이미지

FastAPI · Docker · ECR

Terraform IaC

GitHub Actions CI

DASHBOARD | AUTH & RBAC

인증과 인가의 역할 분리

Cognito는 로그인·세션을, RDS는 역할·공장·시스템 접근 권한을 관리

AWS Cognito 인증

"너 누구야?"

로그인 · JWT(sub) | 신원만 책임

sub로 RDS 조회

RDS PostgreSQL 인가

"뭘 볼 수 있어?"

역할 · 공장 권한 | 사람마다 다른 화면

DASHBOARD | USER MANAGEMENT

사용자 관리 시연 - 3단계 시나리오

계정 생성·공장 권한 변경·사용자 삭제가 Cognito 인증과 RDS 인가에 함께 반영되는지 확인

1. 생성 + 첫 로그인

공장 관리자 · factory-a

공장 관리자 **계정 생성**
→ 초대 메일·임시 비밀번호 확인
→ 첫 로그인, factory-a만 노출.

2. 권한 수정

factory-b · c

공장 접근 권한을 변경
→ 보이는 **공장 범위** 새로고침 후 **변경.**

3. 삭제

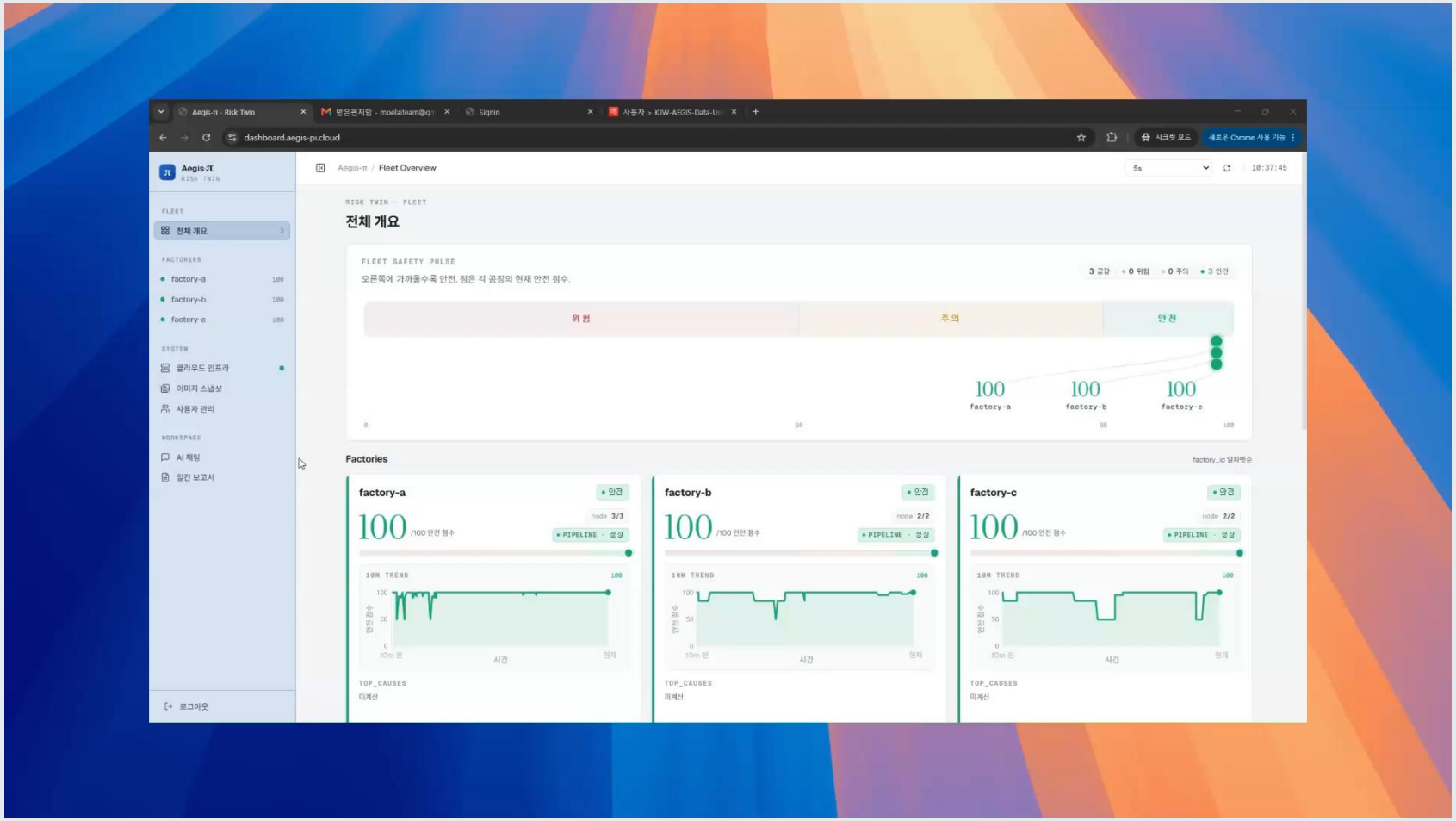
로그인 차단

Cognito 계정과 RDS **권한**을 **제거**
→ 재로그인이 **차단.**

DASHBOARD | DEMO STEP1

사용자 생성과 첫 로그인

Cognito 초대 계정으로 로그인하고 임시 비밀번호를 변경한 뒤 허용된 공장만 확인



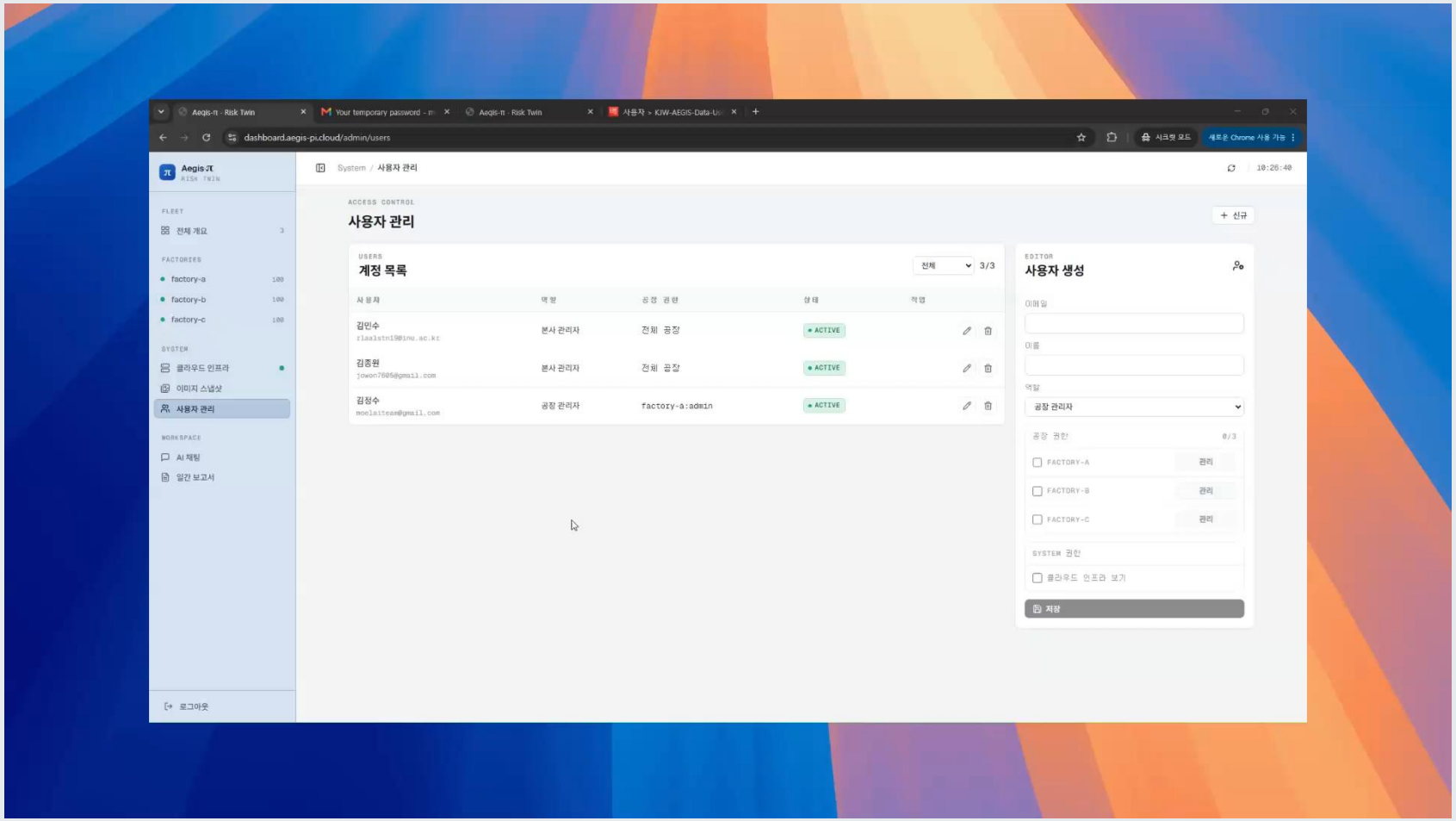
WHAT TO WATCH

공장 관리자 **factory-a** 생성
초대 메일 · 임시 비밀번호 발급
첫 로그인 시 비밀번호 변경
factory-a 만 노출

DASHBOARD | DEMO STEP2

공장 접근 권한 변경

RDS의 공장 권한을 변경하면 다음 조회부터 허용된 공장 범위만 표시



WHAT TO WATCH

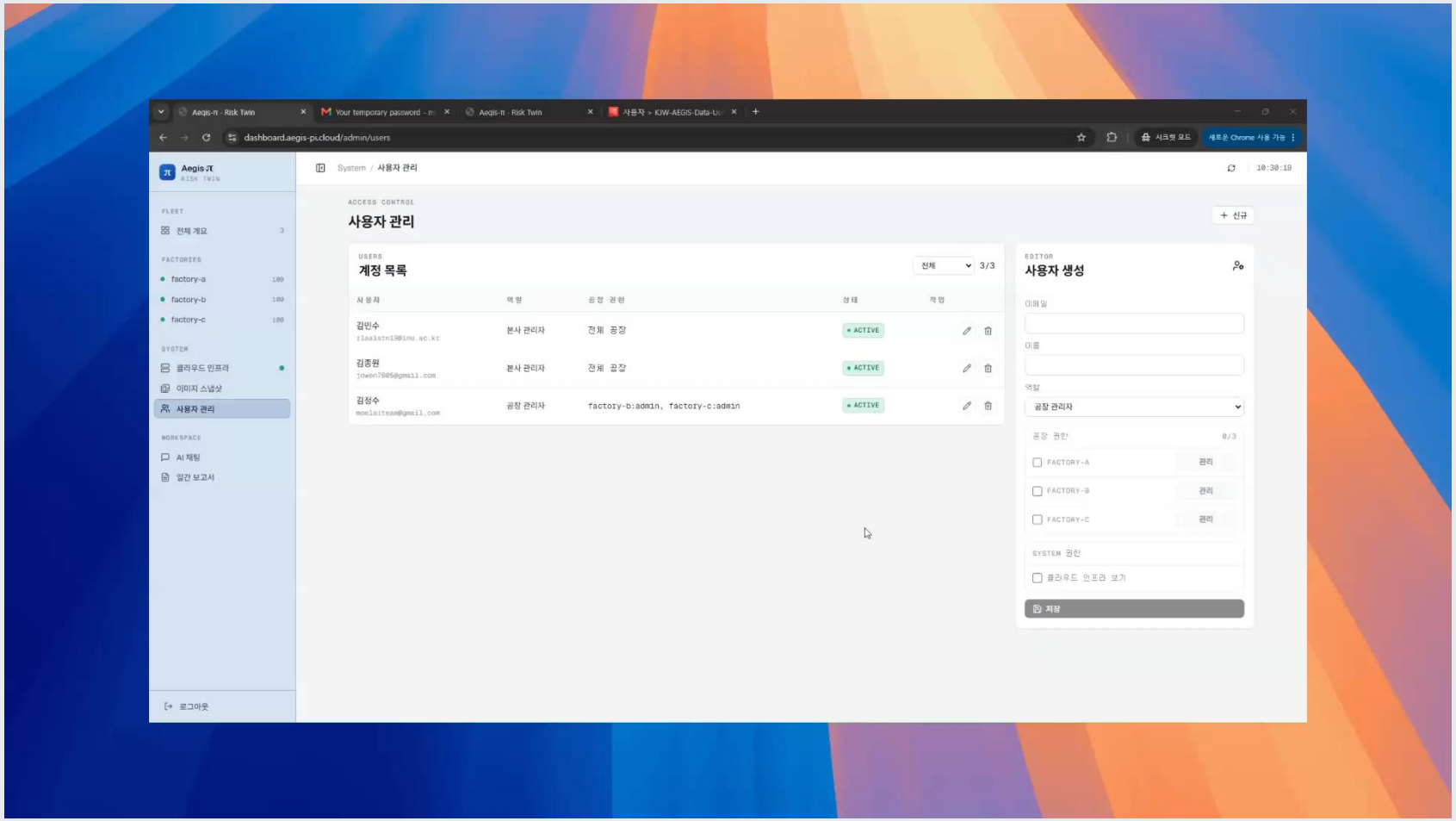
factory-a → factory-b · c

권한 변경·저장
신규 사용자 화면 새로고침
보이는 공장 범위 **다음 조회 적용**

DASHBOARD | DEMO STEP3

사용자 삭제와 로그인 차단

Cognito 계정과 RDS 권한을 함께 삭제해 이후 로그인을 차단



WHAT TO WATCH

계정 삭제

목록에서 사라짐

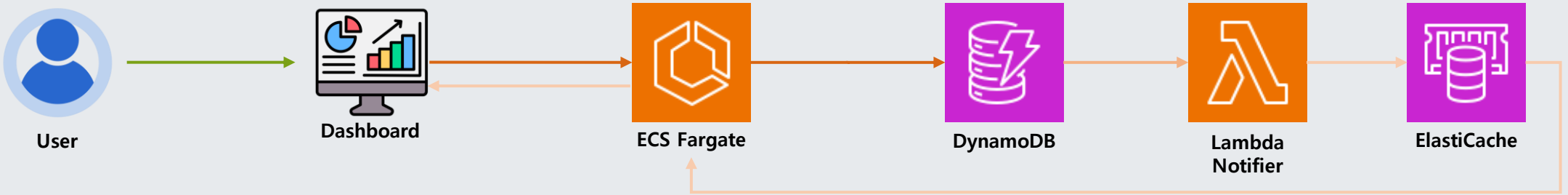
재로그인 불가

Cognito 계정 + RDS 권한 함께 제거

DASHBOARD | REALTIME

실시간 모니터링 아키텍처

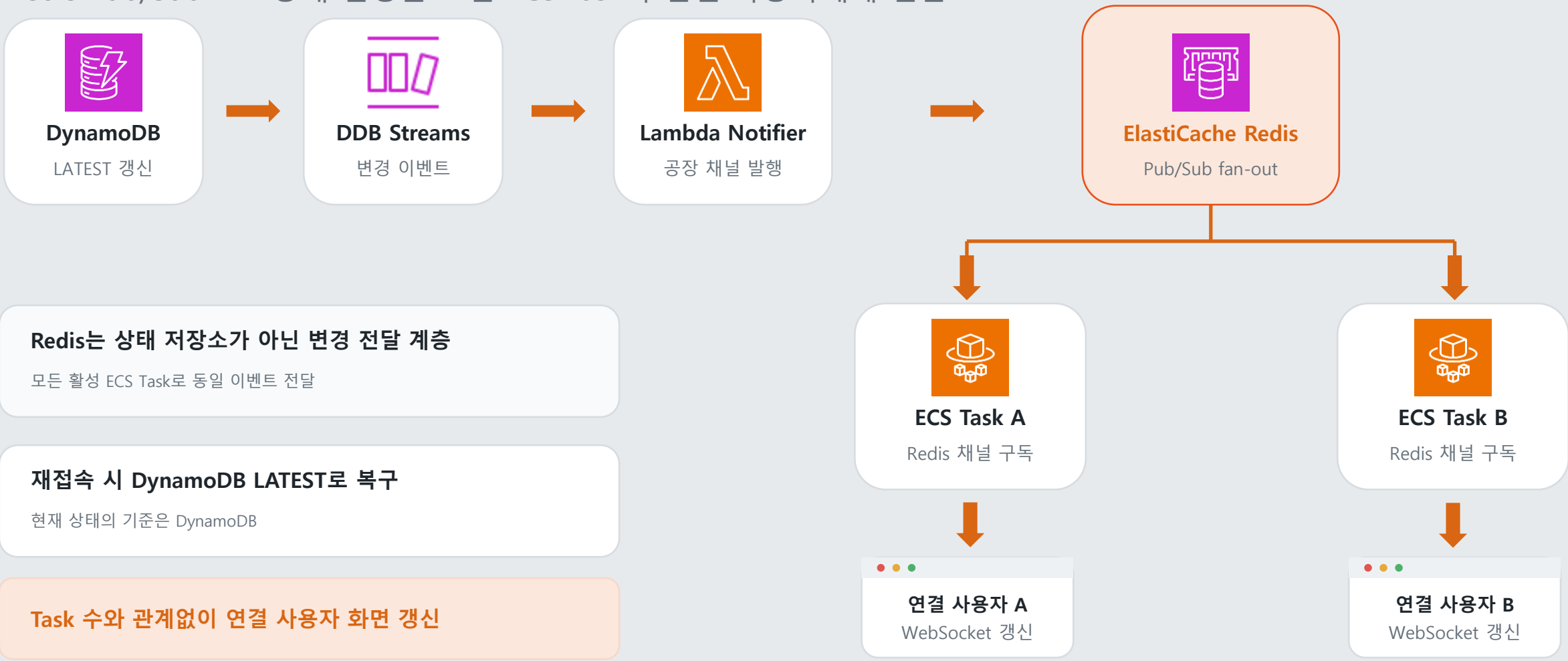
새로운 공장 상태를 감지해 여러 대시보드 화면에 자동으로 전달



DASHBOARD | REALTIME FAN-OUT

여러 Backend Task에 변경을 전달하는 방법

Redis Pub/Sub으로 상태 변경을 모든 ECS Task와 연결 사용자에게 전달



Redis는 상태 저장소가 아닌 변경 전달 계층

모든 활성 ECS Task로 동일 이벤트 전달

재접속 시 DynamoDB LATEST로 복구

현재 상태의 기준은 DynamoDB

Task 수와 관계없이 연결 사용자 화면 갱신

DASHBOARD | CLOUD ERROR

ECS Task 장애 감지와 자동 복구

Cloud Infra Collector가 실행 Task와 ALB Target 변화를 감지하고, ECS Service가 목표 Task 수를 자동 복구

1. 장애 발생

backend ECS

ECS 2대 중 1대를 일부러 중지

→ 목표 2 · 실행 1 (warning).

2. 자동 감지 · 알림

Cloud Infra

정상 화면 유지 · 장애 상태는 Warning

3. 자동 복구

ECS Service 자동 복구

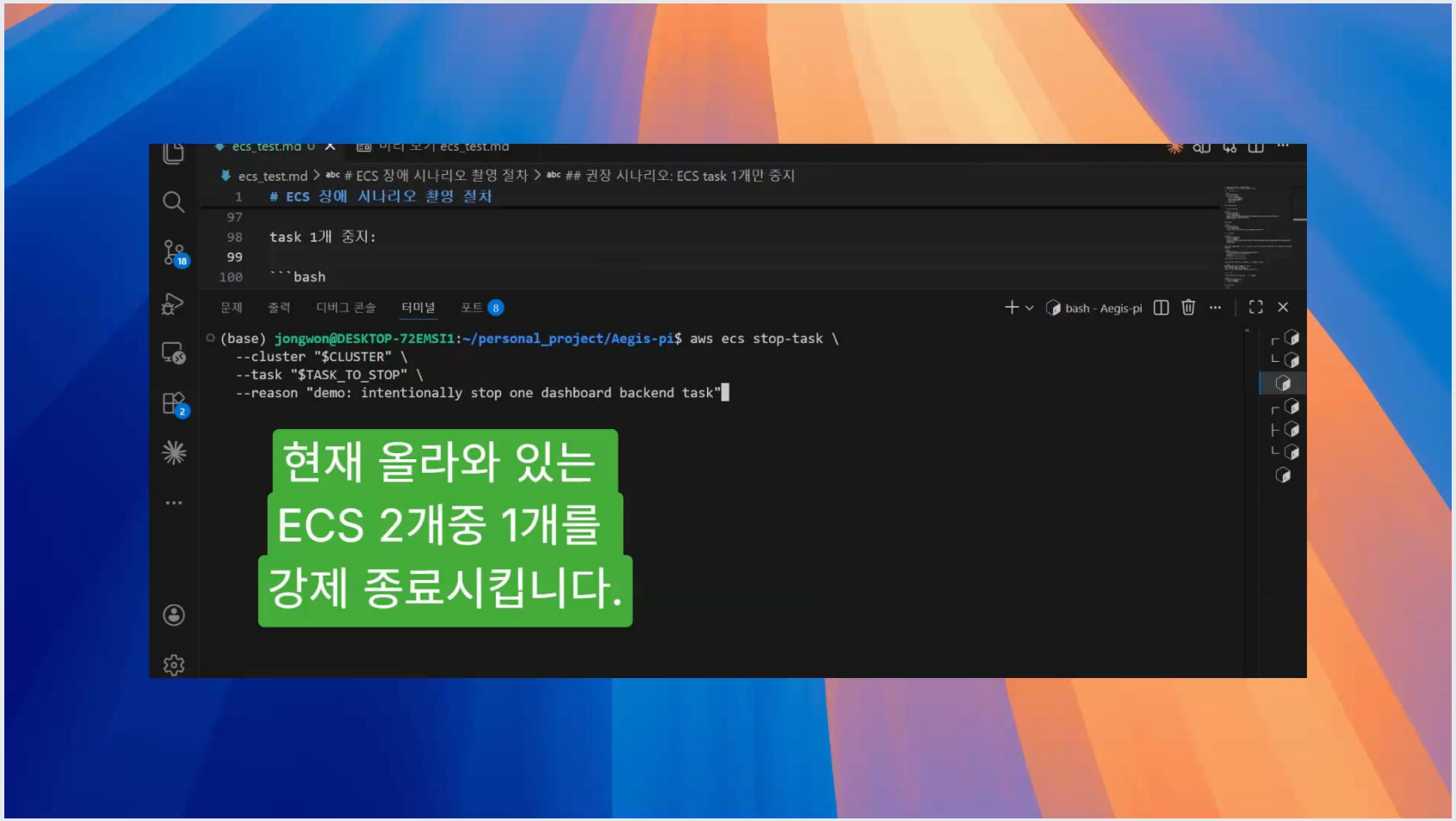
대체 Task 자동 기동

→ 정상(2/2) 복귀

DASHBOARD | DEMO

ECS 1대 중지 - 경고 반영부터 정상 복귀까지

ECS 2/2 → 1/2 경고 → 대체 Task 기동 → 2/2 정상 복귀



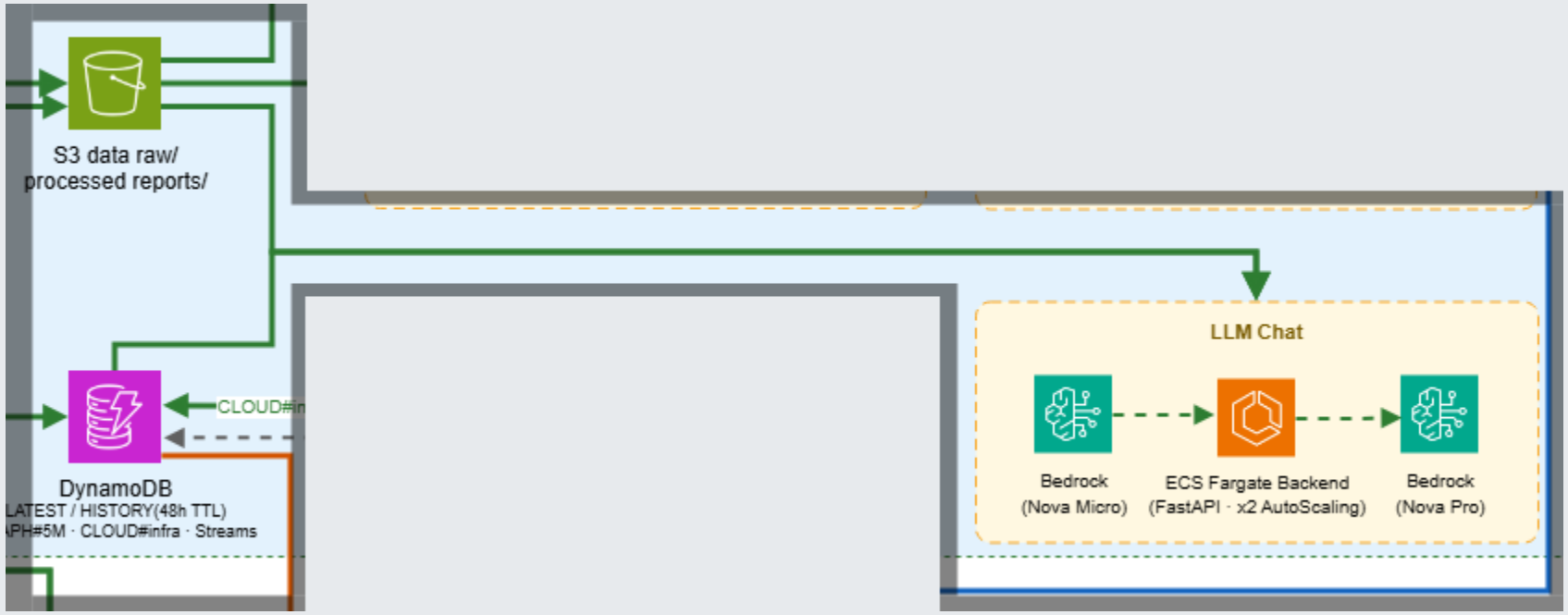
WHAT TO WATCH

- ECS 1대 **강제 중지** (1/2)
- Dashboard 상태 **주의** 반영
- ALB target 하락
- **자동 복구** → 정상 (2/2)

DASHBOARD | AI CHAT OVERVIEW

AI 채팅 - 전체 아키텍처의 데이터 질의·응답 경로

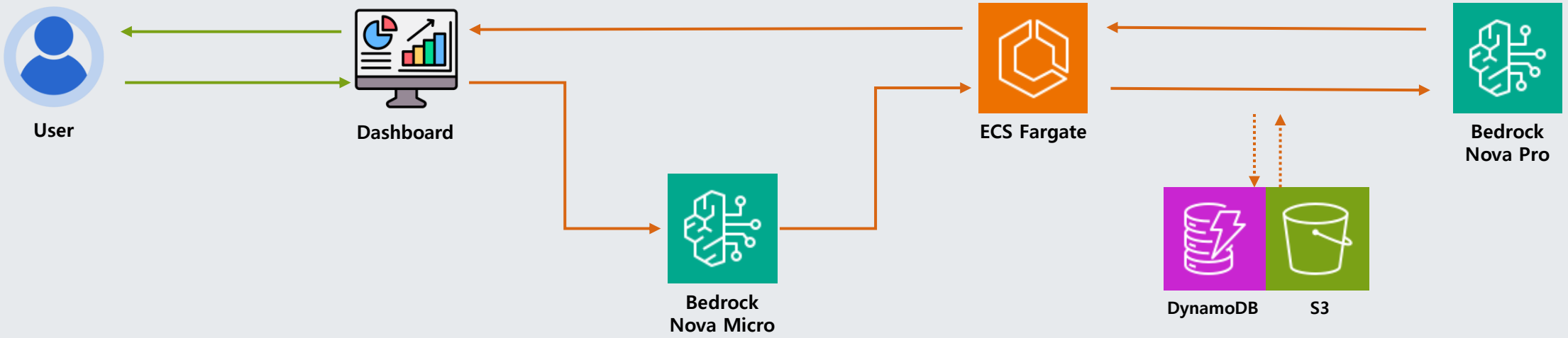
ECS Backend가 Bedrock과 DynamoDB·S3를 연결해 권한 범위 내 근거를 구성



DASHBOARD | AI CHAT DETAIL

AI 채팅 - Bedrock 2단계 처리 구조

Nova Micro가 질문을 구조화하고, Backend가 검증한 Evidence를 Nova Pro가 설명



DASHBOARD | AI CHAT

LLM은 해석하고, Backend는 검증한다

자연어 → 구조화 질의 → RBAC·데이터 검증 → Evidence(확인·추정·누락) → 근거 기반 답변

① 자연어 질문

"어제 오후 2~4시 안전 점수가 왜 떨어졌어?"

② Nova Micro

질문 → 구조화 결과

답변 생성 안 함

```
{
  "intent": "cause_analysis",
  "factory_id": "factory-a",
  "time": {
    "mode": "interval",
    "start_kst": "2026-06-09T14:00",
    "end_kst": "2026-06-09T16:00"
  }
}
```

③ FastAPI Backend

JSON 검증 · RBAC
DynamoDB · S3 조회
통계 · 스파이크 계산

실시간 hot store · 문서·이미지

④ Evidence

Confirmed

Inferred

Missing

⑤ Nova Pro

Evidence만 설명

근거 기반 답변

권한 밖 데이터 - 조회 전 차단

수치·스파이크 - Backend가 계산

확정·추정·누락 - 분리 전달

DASHBOARD | AI CHAT

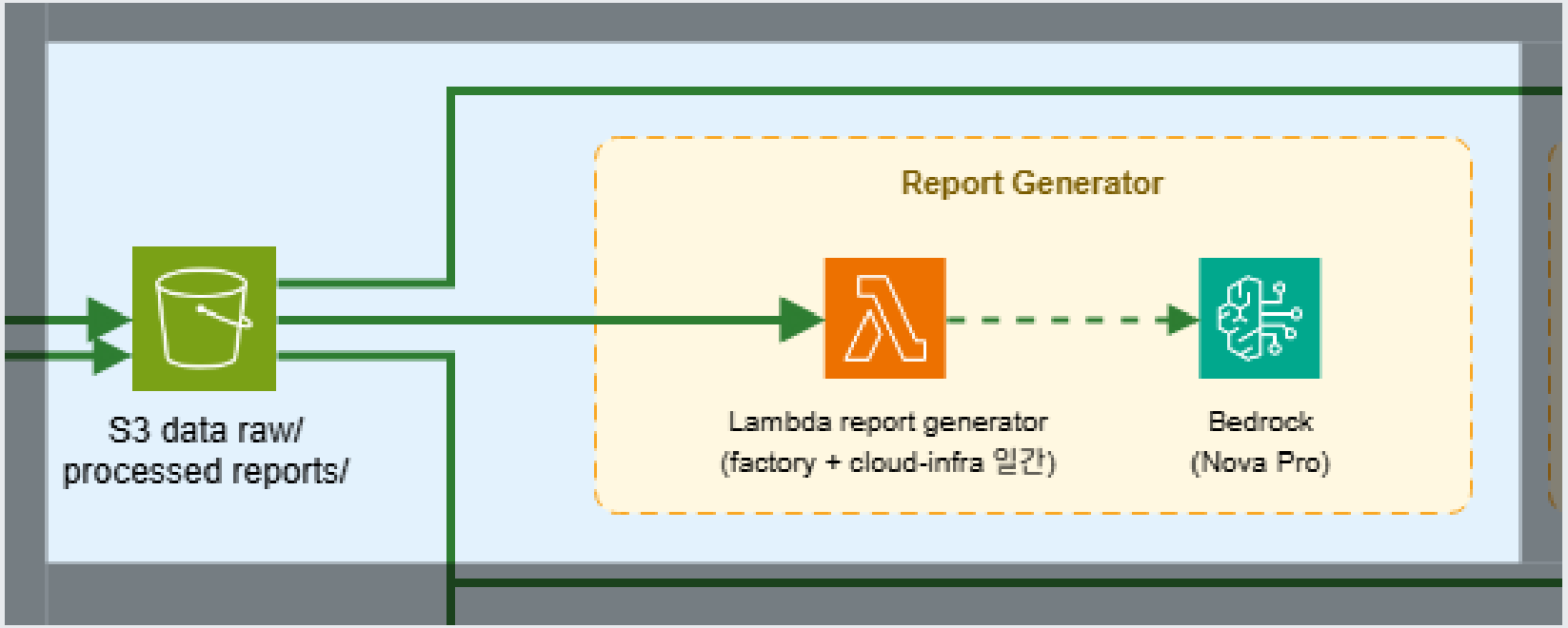
AI 채팅 - 실제 데이터 기반 관제 질의

상태·추이·원인을 자연어로 묻고 확인된 사실·추정·누락 데이터를 구분해 확인

DASHBOARD | AI REPORT OVERVIEW

AI 일간 보고서 - 자동 생성 파이프라인

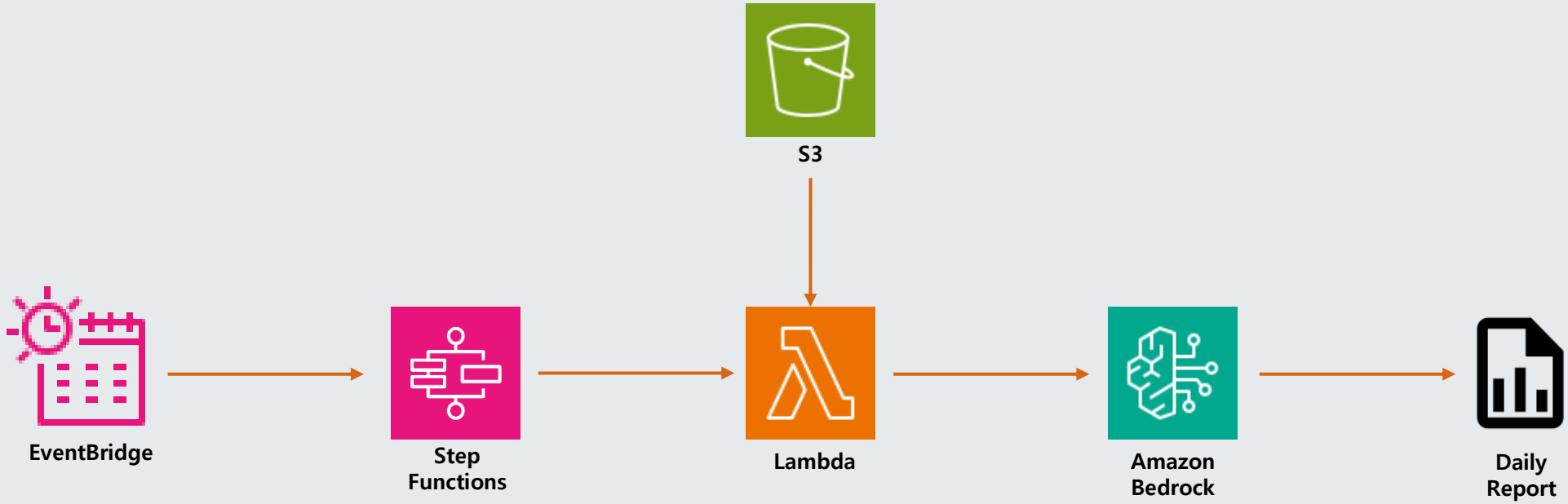
EventBridge와 Step Functions가 매일 00:30 KST에 전일 보고서를 자동 생성



DASHBOARD | REPORTING ARCHITECTURE

일간 보고서 자동 생성 아키텍처

매일 00:30 KST, 전일 데이터를 공장·시간 단위로 집계해 Markdown으로 저장



DASHBOARD | AI REPORT PRINCIPLE

수치는 코드가 정리하고, Bedrock은 설명한다

전일 S3 processed를 공장·시간 단위로 집계한 Report Context만 Bedrock에 전달

INPUT

운영 데이터

- 안전 점수 시계열
- 센서 · AI 탐지
- 인프라 상태

Raw values & signals

REPORT CONTEXT

Report Context 생성

- 시간 · 일 단위 통계
- 주요 이벤트 추출
- 위험 구간 · 데이터 한계
- 출처 · 시간 범위 명시

집계·병합 계산된 근거만 전달

OUTPUT

Markdown 보고서

- 요약
- 주요 이벤트
- 위험도 추이
- 확인 필요 항목

Bedrock 설명

수치 · 이벤트 - 코드가 계산

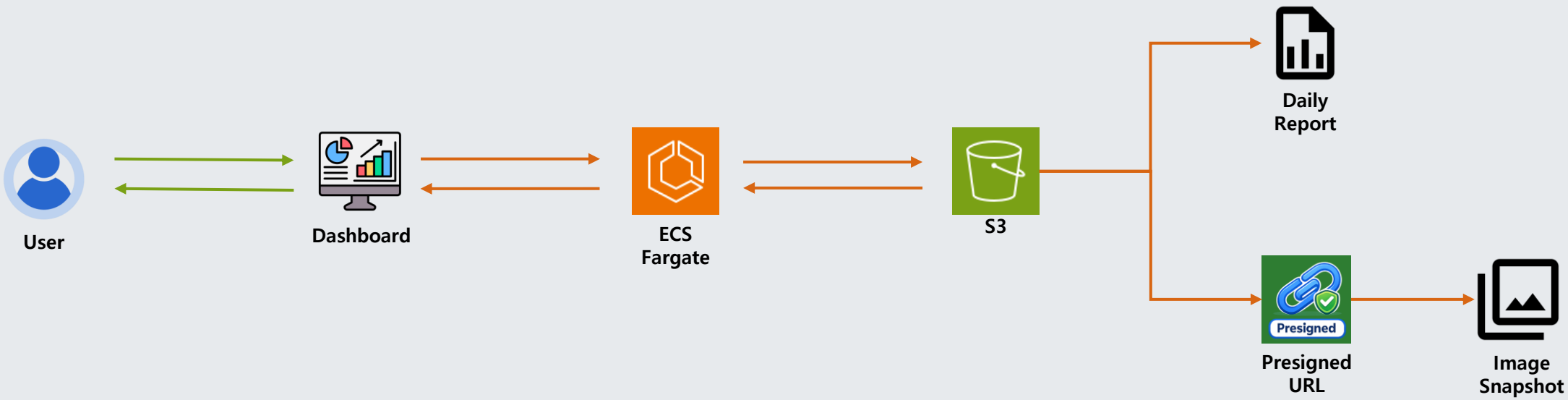
Bedrock - Context만 설명

결과 - Markdown으로 보존

DASHBOARD | S3 READ ARCHITECTURE

보고서·이미지 - 권한 기반 S3 조회 아키텍처

ECS Backend가 RBAC를 확인한 뒤 S3 콘텐츠를 Dashboard에 전달



DASHBOARD | S3 CONTENT DELIVERY

콘텐츠에 따라 전달 방식을 분리하다

보고서는 본문으로, 이미지는 Presigned URL로 안전하게 제공

DAILY REPORT

본문 반환

- Markdown 텍스트
- Backend가 S3 GetObject
- 권한 확인 후 본문 반환
- PDF 인쇄 · Word(.docx)

이유 화면 렌더링 · 문서 출력 필요

IMAGE SNAPSHOT

임시 접근

- 이미지 원본 파일
- S3 객체 목록 조회
- Backend가 Presigned URL 생성
- 15분 후 자동 만료

이유 원본 비공개 · 필요한 시간만 접근

보고서 - 공장 접근 권한

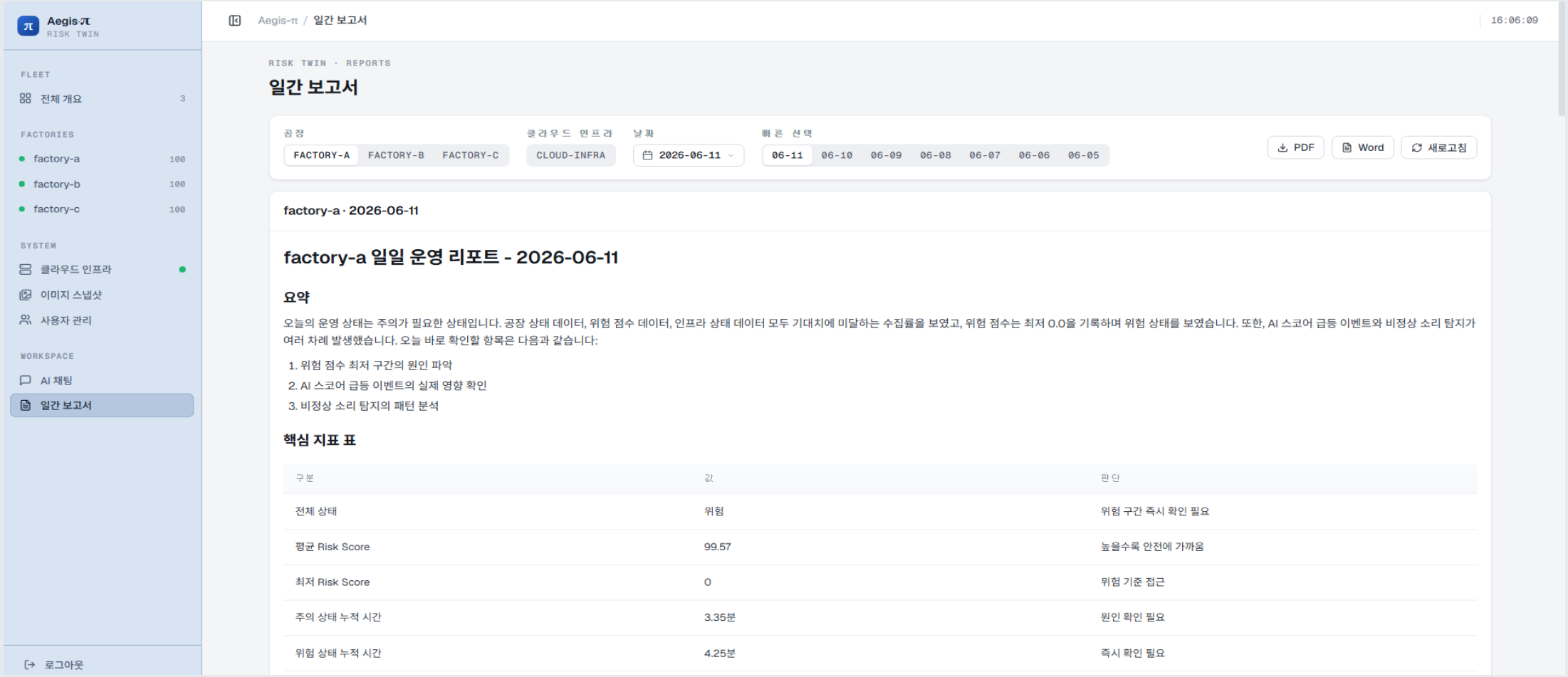
이미지 - System 조회 권한

S3 - 원본 비공개

DASHBOARD | DAILY REPORT

일간 보고서 - S3 Markdown 조회

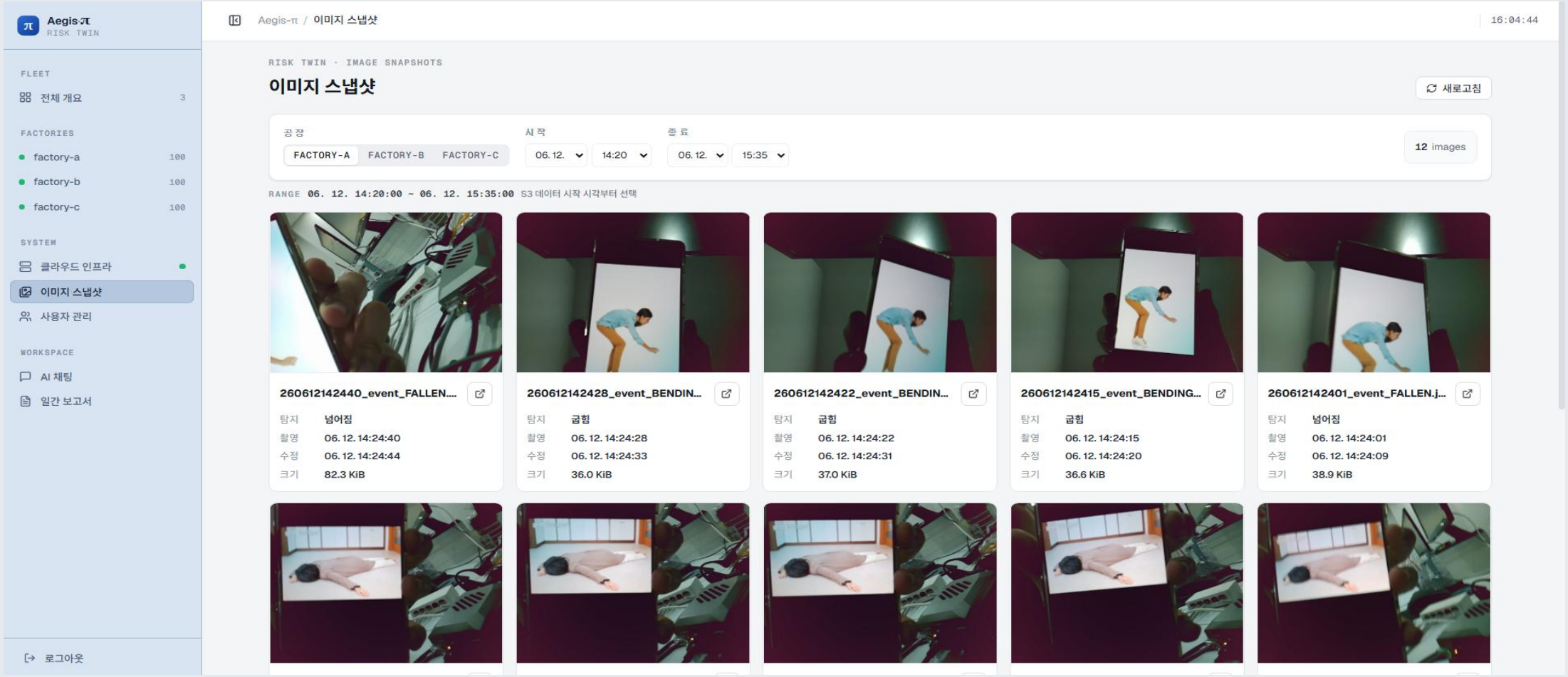
공장·날짜별 보고서를 렌더링하고 PDF 인쇄 · Word(.docx)로 내보내기



DASHBOARD | IMAGE SNAPSHOT

이미지 스냅샷 - 저장된 증빙 이미지 조회

System 조회 권한을 확인한 뒤 공장·시간 범위의 이미지를 Presigned URL로 안전하게 열람



05

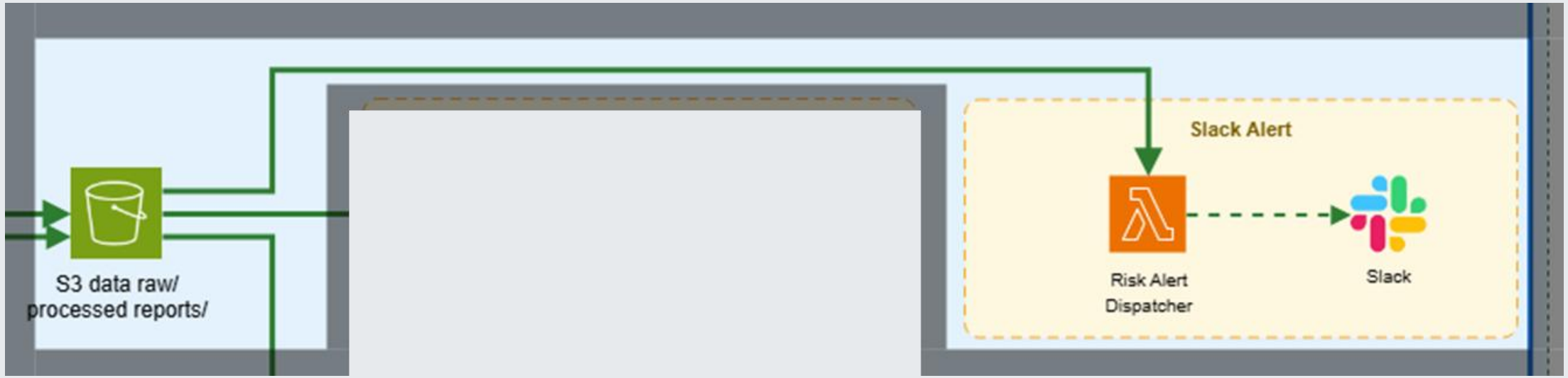
위험 판단 기준 및 알람 체계

현장도, 시스템도, 접근 권한도 - 관제에 필요한 3축을 실제 운영 상황처럼 검증합니다.

ALERT | IMAGE SNAPSHOT

전체 아키텍처 속 알람 경로

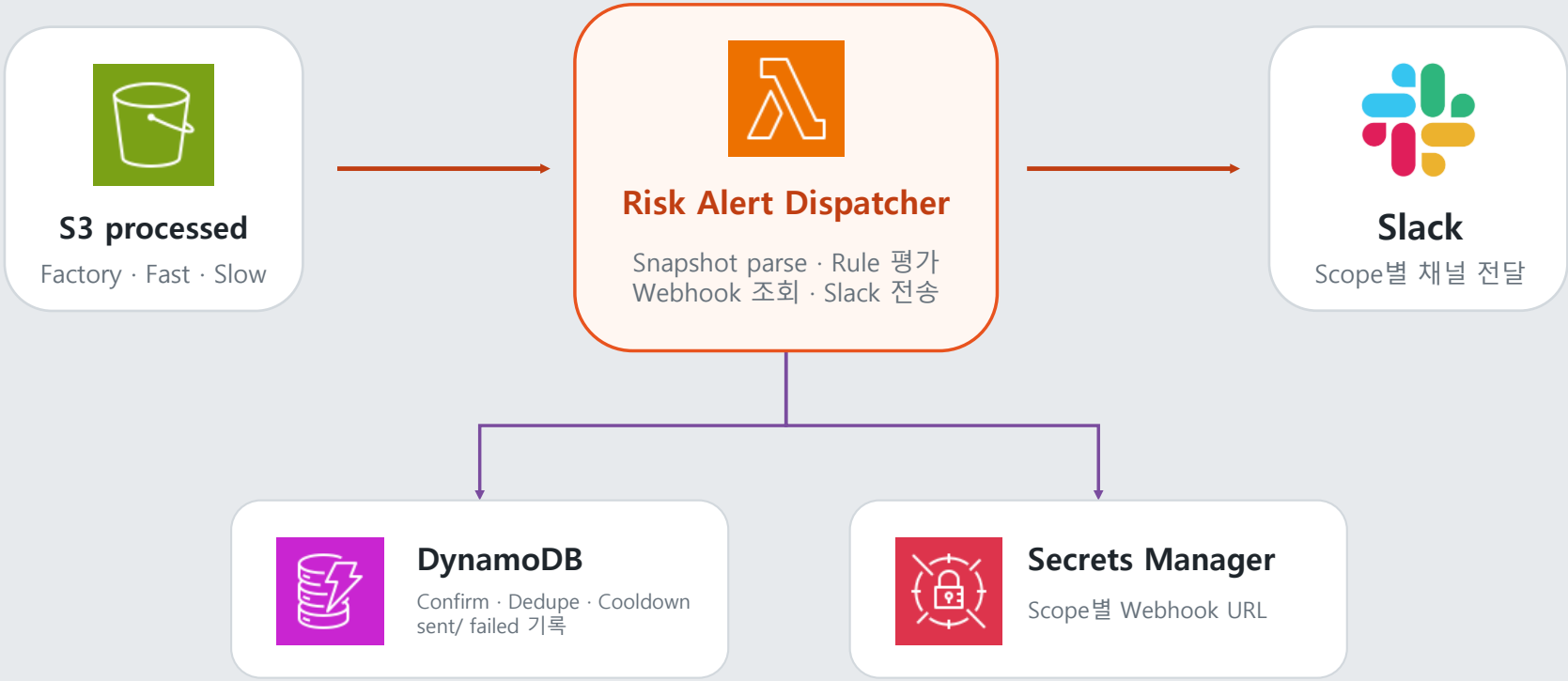
S3 processed Snapshot을 기준으로 Risk Alert Dispatcher가 Slack 알림 생성



ALERT | ALERT ARCHITECTURE

알람 처리 흐름

판단과 전송은 Lambda가 조정 — DynamoDB는 상태, Secrets Manager는 Webhook 관리



모든 조회 · 갱신 · Slack 전송은 Risk Alert Dispatcher Lambda가 조정

Factory 알람 기준과 Top Causes

위험 점수로 심각도를 판단하고, 상위 원인으로 '왜 위험한지'를 함께 전달

FACTORY ALERT

알람	판단 지표	기준
안전 주의	Safety Score	50 ≤ Score < 85
안전 위험	Safety Score	Score < 50
데이터 지연	Infra Age	60초< Age ≤ 120초
데이터 단절	Infra Age	Age > 120초 · 미수신

Danger Top Cause · Gate는 독립 Danger 알람

TOP CAUSES · 최대 5개

점수를 낮춘 원인을 severity → contribution 순으로 정렬

- **ENV** 온도 · 습도 · 기압
- **AI** 화재 · 낙상 · 굽힘 · 이상음
- **EDGE** Node · Pod · 장치 · 저장 · 네트워크
- **DATA** 데이터 최신성 · Pipeline

WEIGHTED
임계 구간에 따라 점진적 감점

GATE
위험 조건에서 Score 즉시 제한

- Weighted = 점진적 감점
- Gate = 즉시 점수 제한
- Warning 15분 / Danger 5분 Cooldown

ALERT | CLOUD FAST · 1M

Cloud Fast - 1분 알람 기준과 원인 근거

1분마다 운영 경로를 확인하고, 최근 5분 지표로 이상 원인을 판단

FAST · Alert

알람	판단 지표	기준
Lambda 오류	Errors (5m)	1건 이상
Lambda 제한	Throttles (5m)	1건 이상
DDB 제한	Read/Write Throttle	1건 이상
ALB 대상 이상	Unhealthy Host	1대 이상

Collector 실행 1분 · Metric Window 최근 5분

원인 근거 · reasons[ ]

Collector가 Section status와 판단 근거를 함께 기록

- LAMBDA — 오류 또는 Throttle 발생
- DDB — Read · Write 처리 제한
- ALB — 비정상 대상 감지
- FALLBACK — Backend · Pipeline 상태 이상

- **구체 원인 감지 → 해당 알람**
- **원인 없음 → Section 상태**

- Specific 발생 시 Generic 억제
- 일부 Warning 90초 내 2회 확인
- 수집 실패는 errors[]로 분리

ALERT | CLOUD SLOW · 5M

Cloud Slow - 5분 알람 기준과 원인 근거

5분마다 EKS · Kubernetes · ArgoCD · 저장 최신성을 점검

SLOW · 5분

알람	판단 지표	기준
EKS 상태	Cluster·NodeGroup	ACTIVE 아님
용량 부족	Healthy / Desired	Healthy < Desired
Node 이상	Ready / Total	Ready < Total
Pod 이상	Phase	Pending·Unknown / Failed
배포·저장	ArgoCD / S3	상태 이상·최신성 누락

Failed Pod는 1개부터 Critical → Danger

원인 코드 · 운영 의미

상세 상태를 운영자가 이해할 수 있는 원인으로 구분

EKS
Cluster · NodeGroup 비활성
Healthy Instance 부족

K8S · CD
Node Ready 부족 · Pod 상태 이상
ArgoCD OutOfSync · Degraded

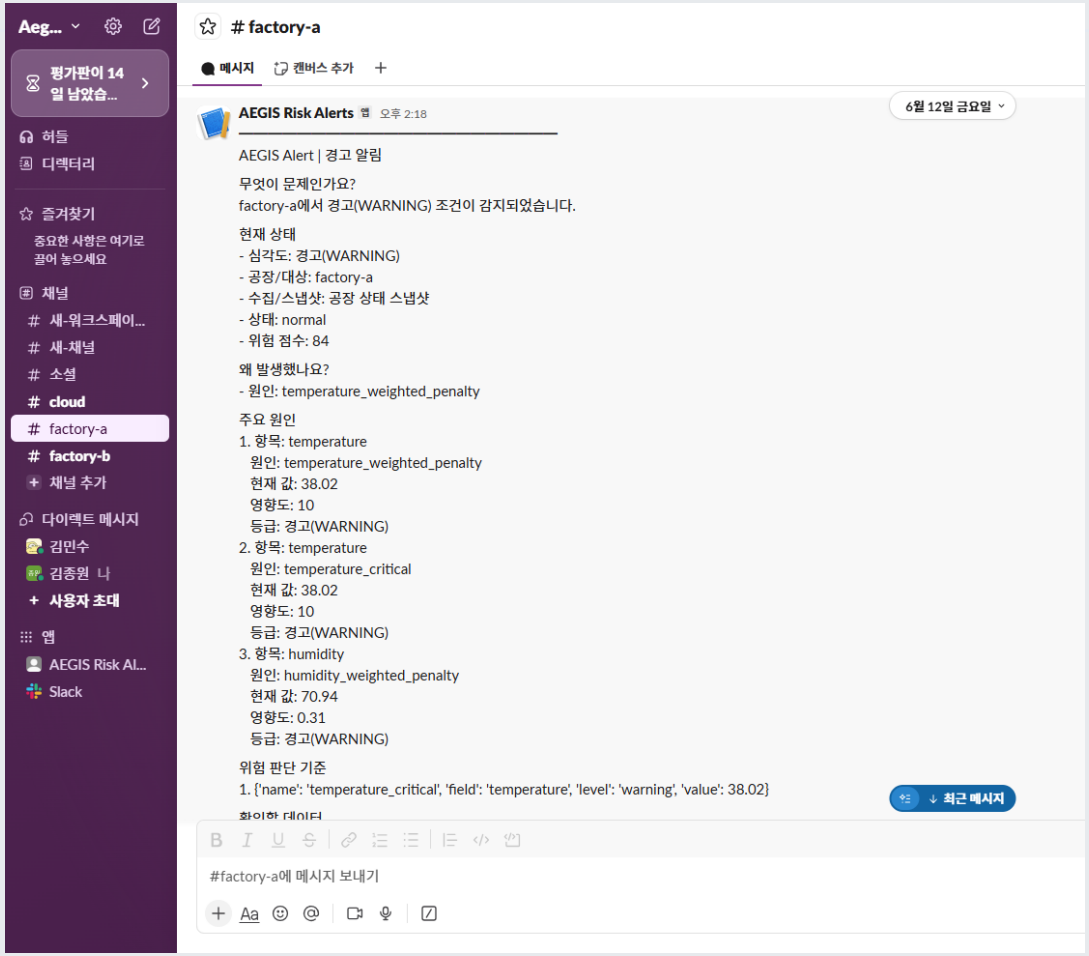
EKS
Raw · Processed 최신성 누락
5분 집계 Snapshot 미생성

- 독립 원인은 각각 유지
- 일부 Warning 450초 내 2회 확인
- Specific 발생 시 Generic 억제

ALERT | SLACK ALERT

Slack 알람 — 판단 근거까지 한 번에

심각도 · 대상 · 상태 · 주요 원인 · 원본 Snapshot을 한 메시지로 전달



ALERT | FIRE SCENARIO

Slack 알람 - 판단 근거까지 한 번에

실시간 위험 인지 → 증빙 이미지 확인 → AI 채팅 요약으로 하나의 사건을 추적

1

실시간 인지

Environment History

온도·화재 점수가 함께 오르며 안전 점수가 위험 구간에 진입합니다.

2

증빙 확인

Image Snapshot

화재 점수가 튄 시각의 스냅샷만 불러와 실제 사진으로 확인합니다.

3

AI 요약

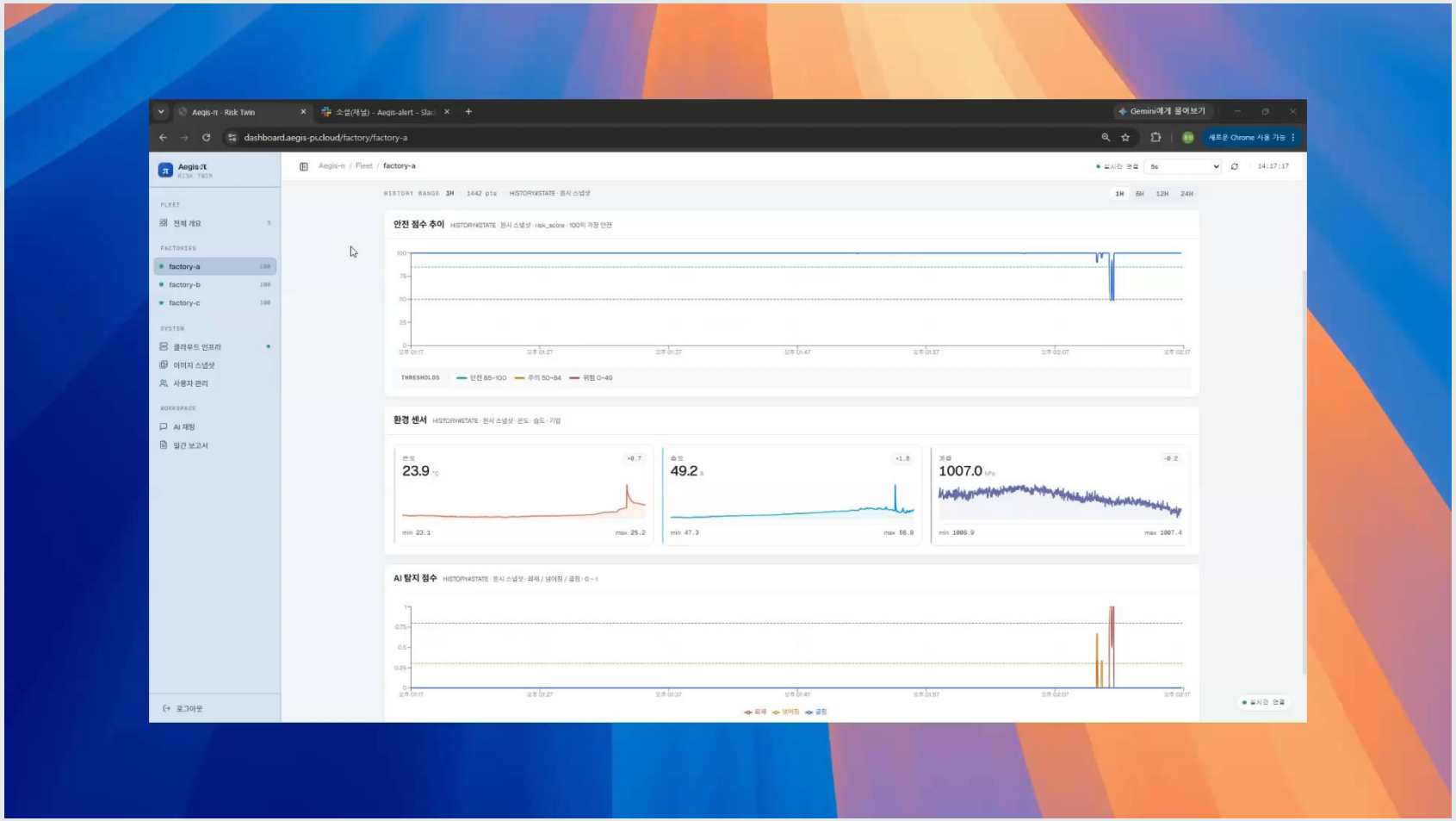
AI Chat

자연어 질의로 스파이크 31건과 증빙 사진 6장(위험 49점)을 요약합니다.

ALERT | DEMO STEP1

화재 시나리오 ① - 실시간 위험 인지

온도와 화재 감지가 함께 상승하며 안전점수가 위험으로 전환되는 과정 확인



WHAT TO WATCH

Environment **History** 1h

온도 32°C 초과

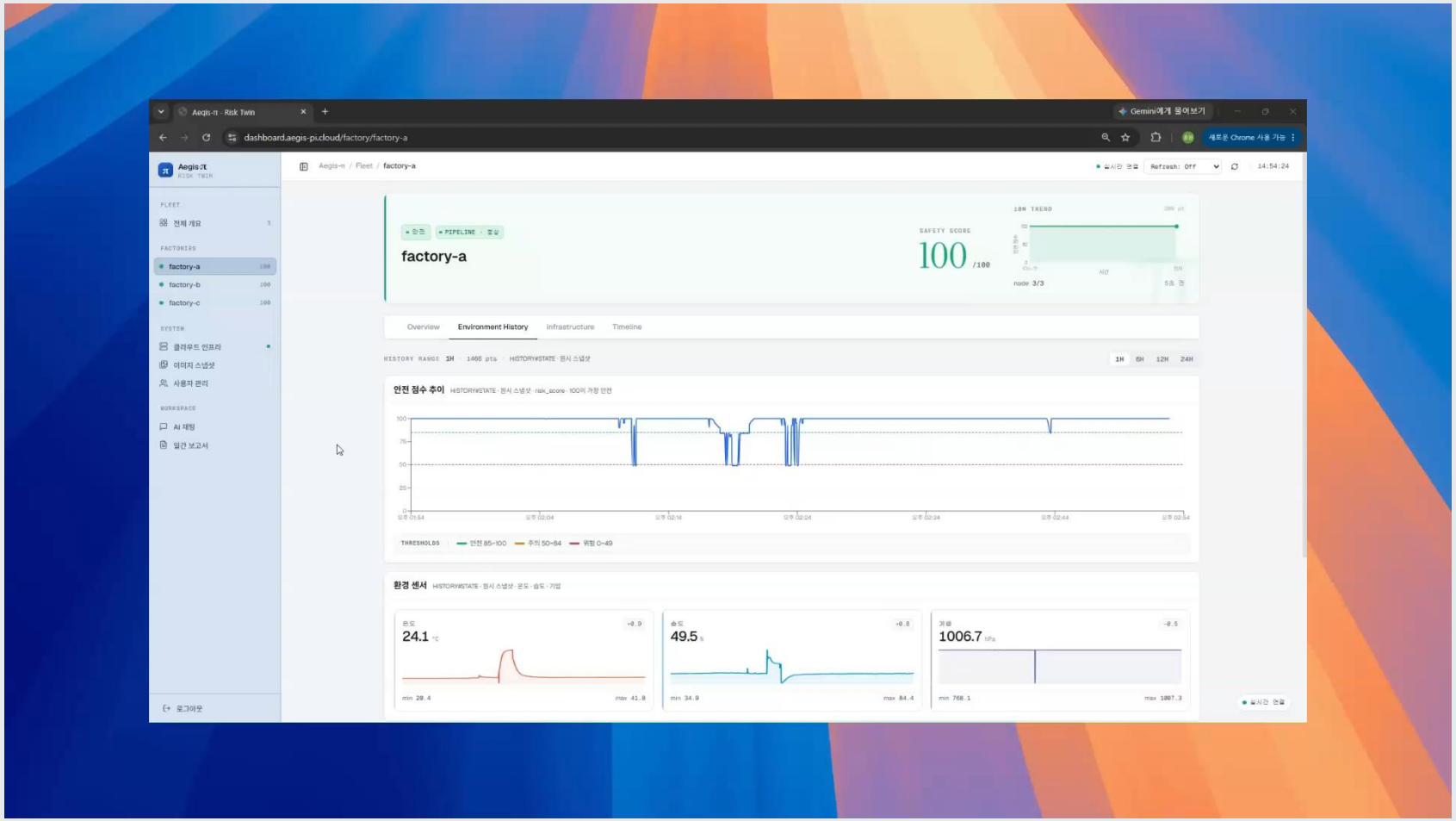
화재 AI 점수 상승

안전점수 위험 Slack 알림

ALERT | DEMO STEP2

화재 시나리오 ② - 증빙 이미지 확인

AI 화재 점수가 튄 시각을 기준으로 S3 스냅샷을 필터링해 실제 사진 확인



WHAT TO WATCH

14:19:51

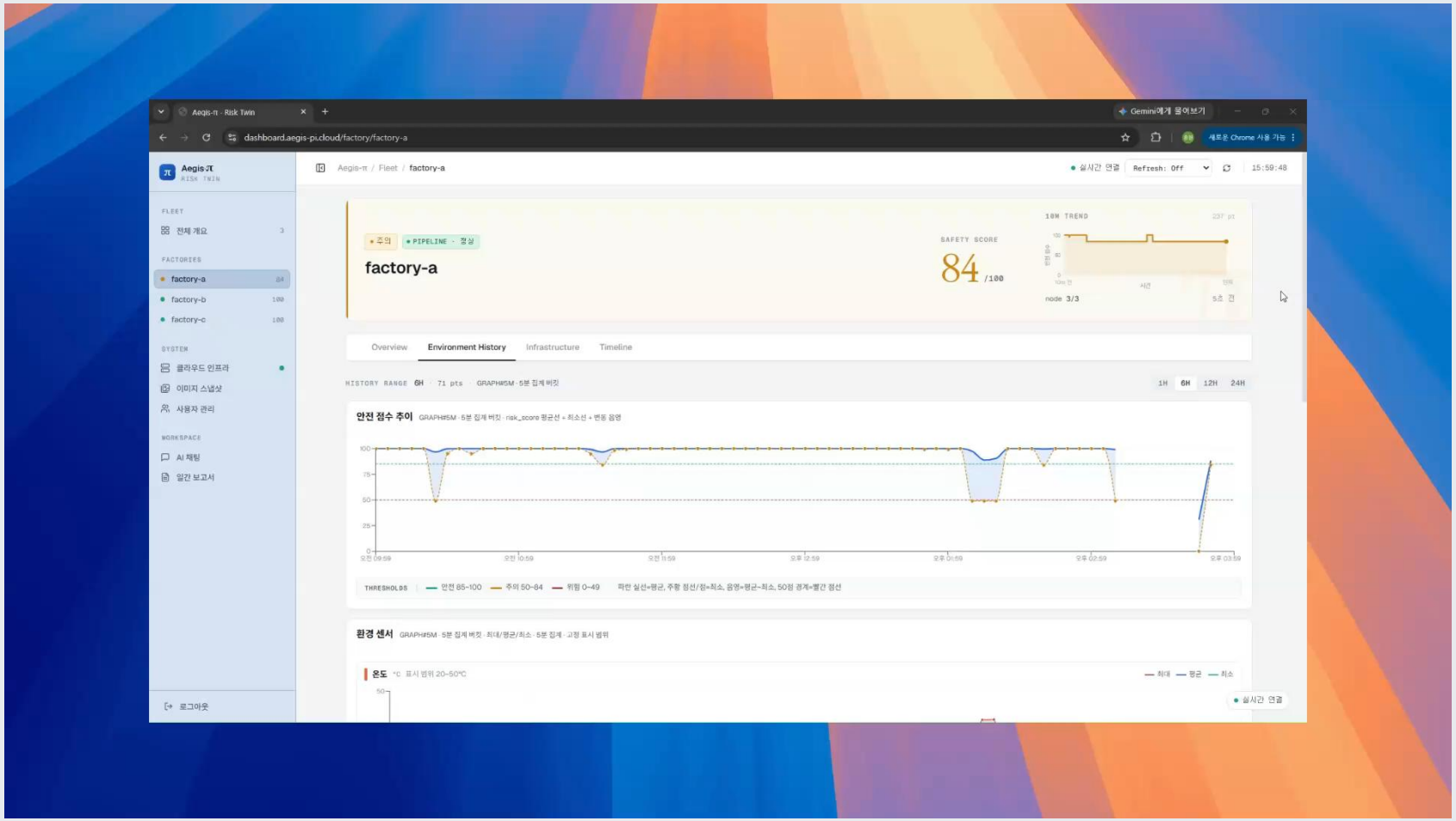
➔ **화재 AI 스파이크**

1h 그래프에서 시점 확인
조회 범위 14:15~14:20
화재 구간 사진 **증빙 확인**

ALERT | DEMO STEP3

화재 시나리오 ③ - AI 채팅으로 상황 요약

6시간 추이와 S3 증빙을 근거로 위험 구간·스파이크·사진을 자연어로 정리



WHAT TO WATCH

factory-a 질의

위험 49점

스파이크 31건

화재 증빙 사진 6장

06

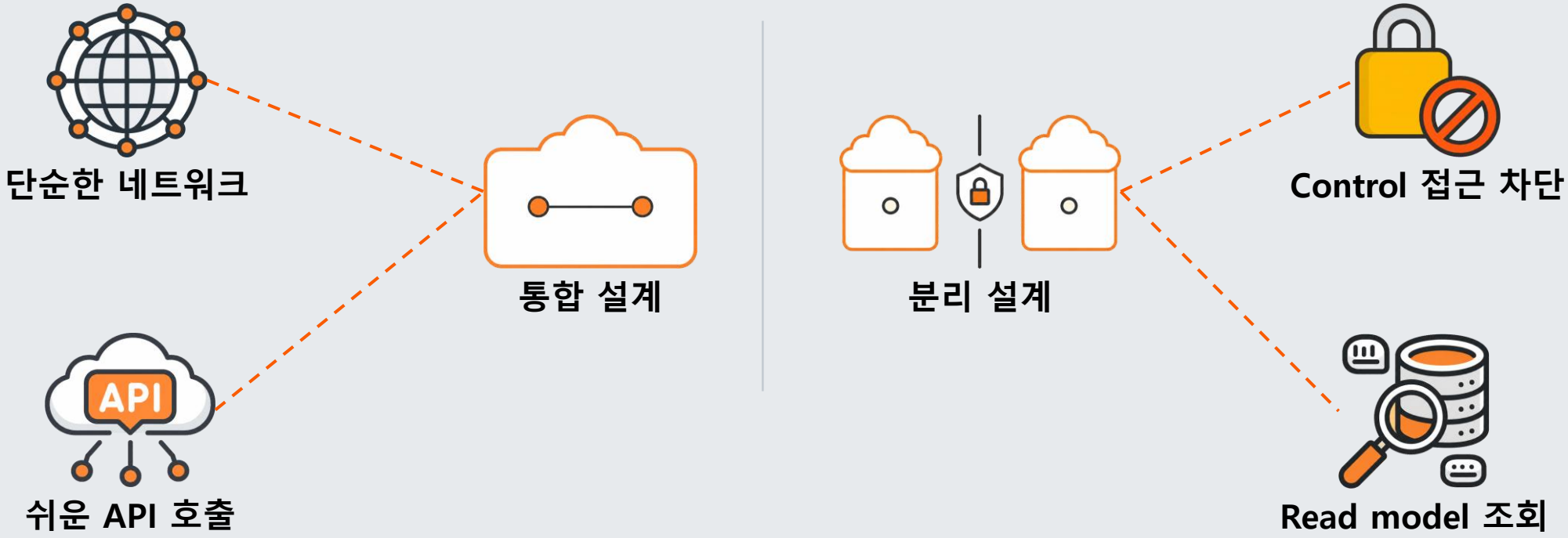
운영 결과 및 확장 방향

실제 운영 중 마주한 설계 선택과 문제 해결 과정을 정리하고
비용, 보안, 확장 관점에서 다음 개선 방향을 제시합니다.

DECISION | DASHBOARD SEPARATION

Dashboard 분리 설계 이유

제어 영역과 관제 영역의 책임과 장애 범위를 분리

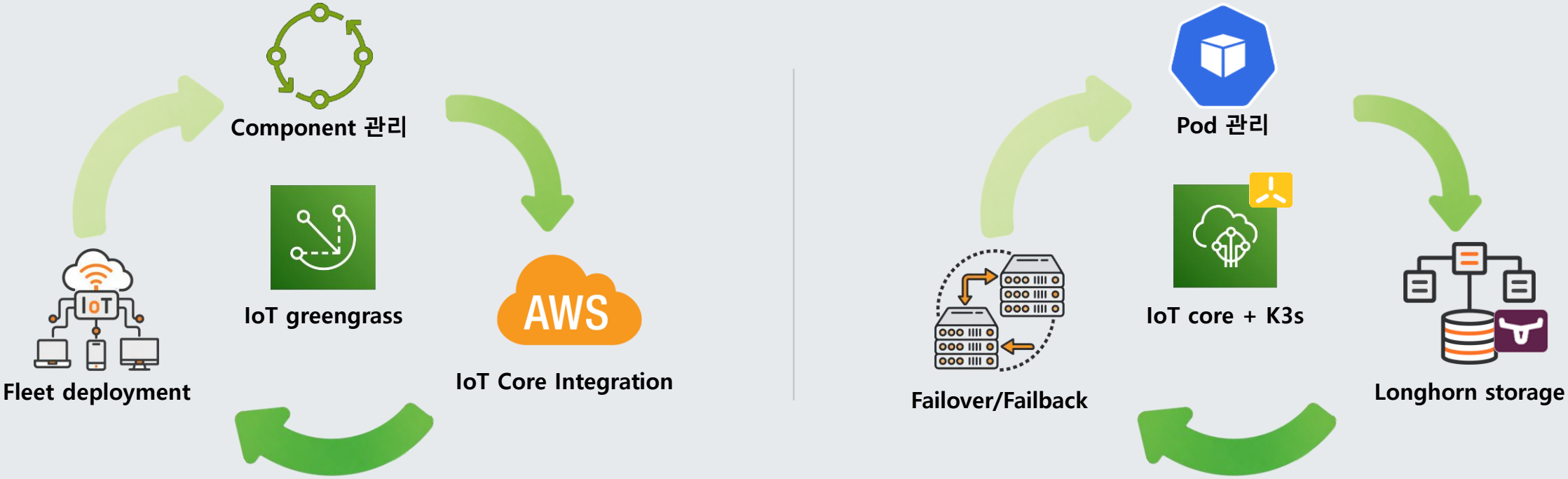


직접 연결된 Dashboard는 live 조회와 초기 구축에는 편하지만
시스템에 필요한 것은
Control Plane과 **분리된** read-only **관제 경계**입니다.

DECISION | K3S OVER GREENGRASS

AWS Greengrass 미선택 이유

현장 장애 대응과 로컬 저장소 제어를 우선 고려



Greengrass는 Edge component 배포와 fleet 관리에는 적합하지만
시스템에 필요한 것은
K3s **Pod 단위의 장애 대응**과 **로컬 저장소 제어**입니다.

DECISION | ECS FARGATE BACKEND

EKS 대신 ECS Fargate를 선택한 이유

단일 관제 API에 필요한 운영 기능만 남긴 선택

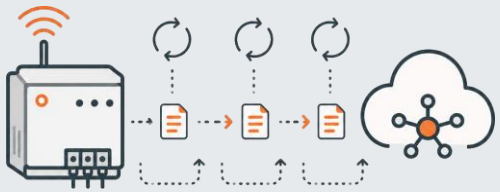
EKS는 다중 Kubernetes workload에 유리하지만

단일 FastAPI에는 ECS Fargate가 비용·배포·보안 경계를 단순화

TROUBLESHOOTING | IOT CONNECTIVITY

문제/해결 ① - IoT Fleet 연결성 지표 불일치

연결 상태와 데이터 유입 성공을 분리해 해석하는 기준



지속적 reconnect 방식

Publish	**매번 연결/해제**
Fleet	**Disconnected 가능**
판단	**데이터 장애로 오인**
Result	**장애 오탐**

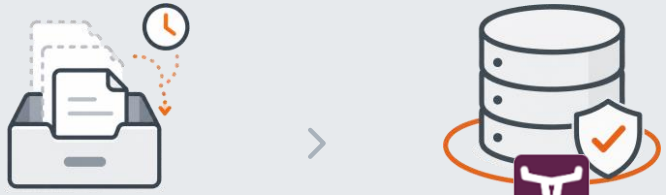
Persistent MQTT 방식

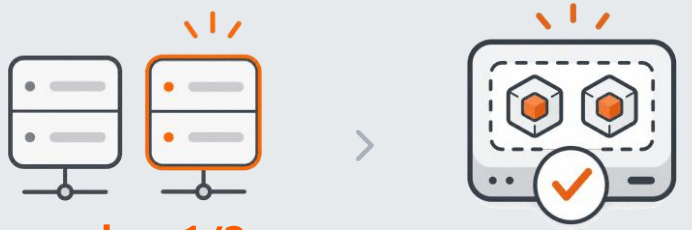
Publish	**Persistent MQTT**
Fleet	**Connected 안정화**
판단	**연결 / 유입 분리**
Result	**운영 지표 신뢰성 확보**

핵심 성과 **"Fleet connected"를 MQTT session 상태로 재정의 · 데이터 처리 성공은 S3 raw / DynamoDB LATEST로 별도 검증**

문제/해결 ② - IoT Publisher Outbox 안정화

전송 전 데이터 보존과 노드 배치 문제를 해결





전송 대기 데이터 보존

Problem	Pod 재시작, 노드 재부팅, 재스케줄링 시 publish 전 JSON이 유실될 수 있음
Action	경로를 /var/lib/aegis/outbox로 고정하고 Longhorn PVC에 마운트
Result	publish 성공 전까지 Edge local storage에 데이터 유지

Node-local 경로 정합성

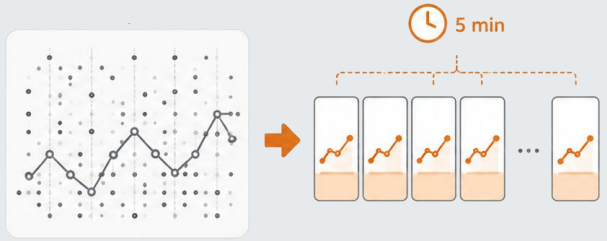
Problem	generator와 publisher가 다른 worker에 있으면 outbox 파일을 공유할 수 없음
Action	worker label 기반 node placement로 generator와 publisher를 같은 node에 배치
Result	publisher가 같은 worker의 outbox를 scan하여 데이터 갱신 경로 정상화

핵심 성과 outbox를 임시 폴더가 아닌 Edge-to-Cloud 전송 보장 계층으로 재정의

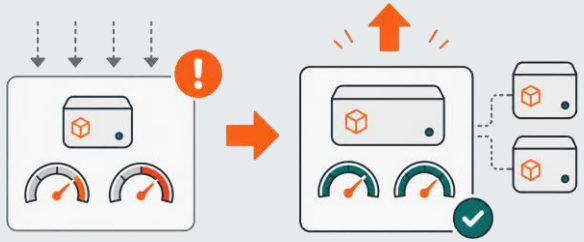
TROUBLESHOOTING | DASHBOARD PERFORMANCE

문제/해결 ③ - Dashboard 렌더링 지연

긴 구간 조회와 실시간 갱신 경로를 분리한 개선



Read model



Right-sizing

Delta Push

Problem

24h 등 긴 구간 조회 시 raw history 조회량이 커져 API 지연·504 발생

Action

GraphAggregator5m로 5분 단위 GRAPH#5M read model 생성

Result

24h 조회 수를 최대 288개 수준으로 제한
긴 window timeout 해소

Problem

동시 조회 시 작은 Fargate task의 CPU가 포화되어 응답 지연·5xx 발생

Action

Backend task를 1 vCPU / 2 GiB로 상향
min 2 warm 유지 및 autoscaling 적용

Result

동시 조회 상황의 CPU 포화와 5xx 완화

Problem

5초 polling마다 전체 series를 재조회·재파싱해 렌더링 부담 증가

Action

WebSocket 알림. since 기반 delta refresh
신규 데이터만 merge/dedupe

Result

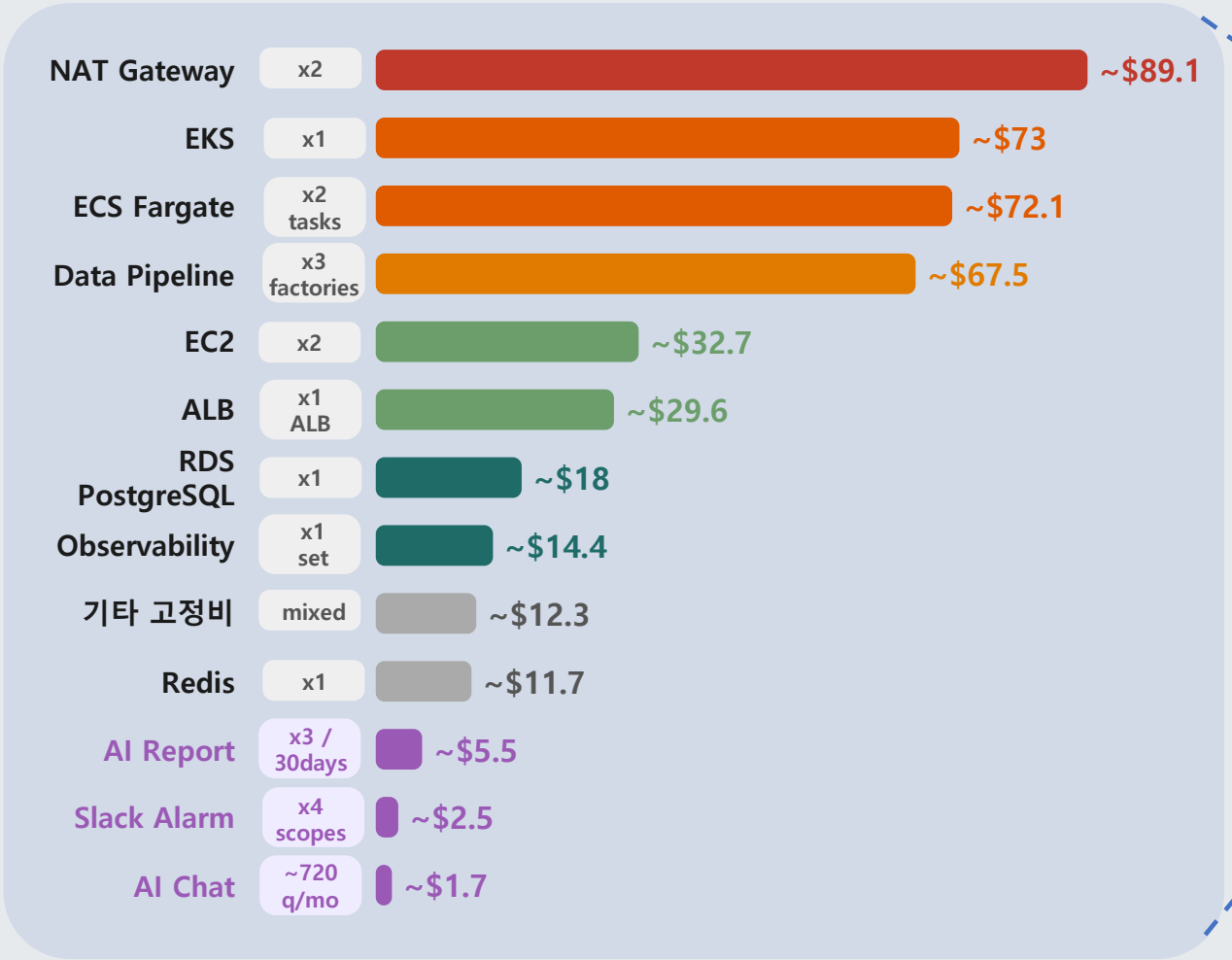
전체 재조회·재파싱 감소
실시간성은 유지, polling 의존도 감소

핵심 성과 긴 window는 집계 read model, 실시간 갱신은 delta push로 분리

COST | OPERATING STRUCTURE

운영 비용 구조

고정 운영비와 선택 기능 비용을 분리해 산정한 결과



항목	기준 금액
고정 운영비	$341 / month
공장 1개 운영	$385 / month
공장 3개 운영	$424 / month
선택 기능 포함	$434 / month
공장 1개 추가	+$19.3 ~ 26.3 / month

ROADMAP | DEVSECOPS SECURITY

보안 강화 로드맵

개발과 배포 권한을 분리해 공급망 보안을 강화

GitHub 보안 설정 강화

현재 개발 흐름 유지
저장소·workflow 권한 최소화

Private repository

Branch protection · CODEOWNERS

MFA 필수화

Actions 권한 최소화

Secret 원문 저장 금지 · SHA pinning

배포 권한과 저장소 분리

production 배포 침해로
이어지지 않도록 권한 경계 분리

Actions → test/build 중심

AWS OIDC 단기 권한

Terraform plan/apply 분리

Production deploy 수동 승인

Secret → Secrets Manager/SSM

ECR push·rollout 권한 분리

내부 통제형 DevSecOps

운영 민감도가 높아지면 CI/CD
통제를 내부 보안 경계로 이전

내부 runner · GitLab 전환 검토

GitLab Self-managed 검토

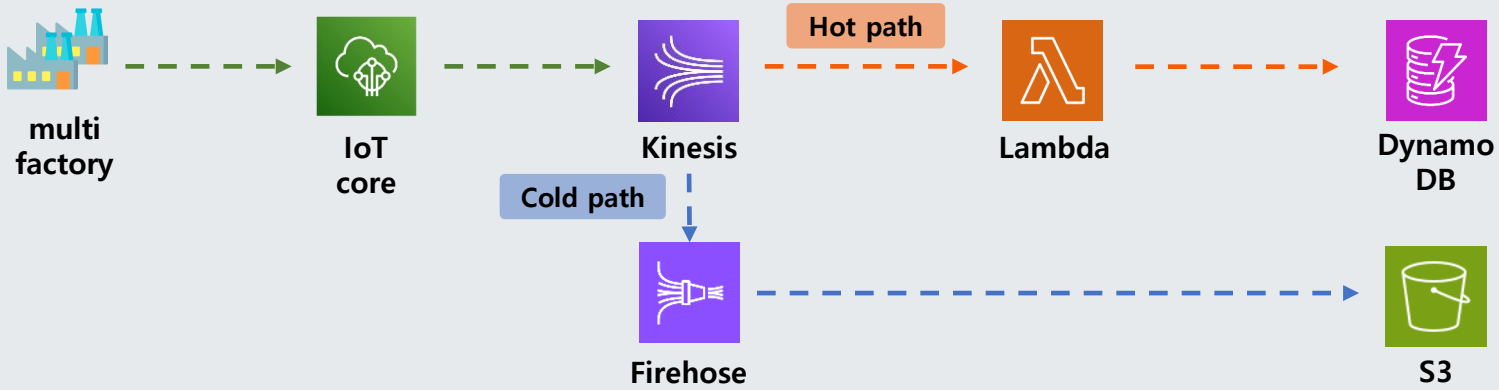
내부망 기반 runner 격리

접근 로그·권한 정책 중앙화

외부 SaaS 의존도 축소

확장 방향 - Kinesis / Firehose

클라우드 버퍼와 장기 보관 경로로 확장하는 방향



Hot Path
준실시간 Dashboard / Alert

Consumer Lambda가 stream 처리
DynamoDB read model 갱신
수 초 이내 관제 반영

Cold Path
보관 / Replay / 분석

Firehose가 batch object로 S3 적재
S3 PUT / ObjectCreated 부담 완화
Hot Path와 장애 영향 분리

Cloud-side Buffer

Edge outbox 이후에도
클라우드에서 burst를 한 번 더 흡수

Lambda / DynamoDB로 가는
순간 부하를 완화

Replay / Recovery

처리 실패나 로직 오류 시
stream 보존 구간 내 재처리 가능

장애 복구와 backfill
운영 여지를 확보

Batch Archive

Firehose가 raw telemetry를
batch object로 S3 적재

S3 small object와
GET / LIST 부담을 완화

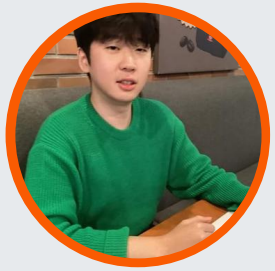
RETROSPECTIVE | TEAM VOICE

프로젝트 회고

김민수 · EDGE & CLOUD

라즈베리파이 기반 K3s 온프레미스 클러스터 구성, IoT Core를 활용한 실데이터
파이프라인 구축, EKS 내 ArgoCD GitOps CI/CD 구성을 통해
온프레미스와 클라우드를 결합한 하이브리드 클라우드 아키텍처를 설계하였습니다.

온프레미스와 클라우드를 동시에 다뤄야 하는 구조다 보니 어느 한쪽만 봐서는 문제의 원인조차
찾기 어려운 상황이 많았습니다. IoT Core 인증 구성에서 그걸 가장 크게 느꼈고
온프레미스와 클라우드 양측을 아우르는 Solution Architect로서의 경험을 쌓을 수 있었습니다.



김종원 · CLOUD & DASHBOARD

IoT, Raspberry Pi 경험을 클라우드로 확장해 멀티 공장 Risk Twin을 구현하며,
기능의 수보다 운영자의 판단을 돕는 흐름이 중요함을 배웠습니다.

매일 진행 상황과 문제를 공유하며, 피드백과 유연한 계획 조정이 완성도의 높임을 느꼈습니다.

11만 건 이상 누적된 이력으로 발생한 504 오류를 개선하며,
조회 패턴·데이터 수명·비용을 함께 고려하는 설계 역량을 키웠습니다.

ST-END | THANK YOU

엣지에서 수집하고,
클라우드에서 판단하고,
한 화면에서 관제합니다.

현장도, 시스템도, 접근 권한도 - Aegispi가 함께 지킵니다.

TEAM 일터방패 · 2026.06

● SYSTEM OK