PLANTPULSE · ARCHITECTURE OVERVIEW

# 네 제품이 맞물리는 방식
# 한 벌로 보는 전체 구조.

현장의 신호가 어디를 지나 화면에 닿는지, 같은 일을 할 수 있는 자리가 여럿일 때 무엇을 어디서 하는지, 끊겼을 때 무엇이 남는지를 **여덟 쪽**에 정리했습니다.

**4**
제품 — 엣지 · 플랫폼 · AI · 스튜디오

**1**
데이터 원장 — 한 곳에 모입니다

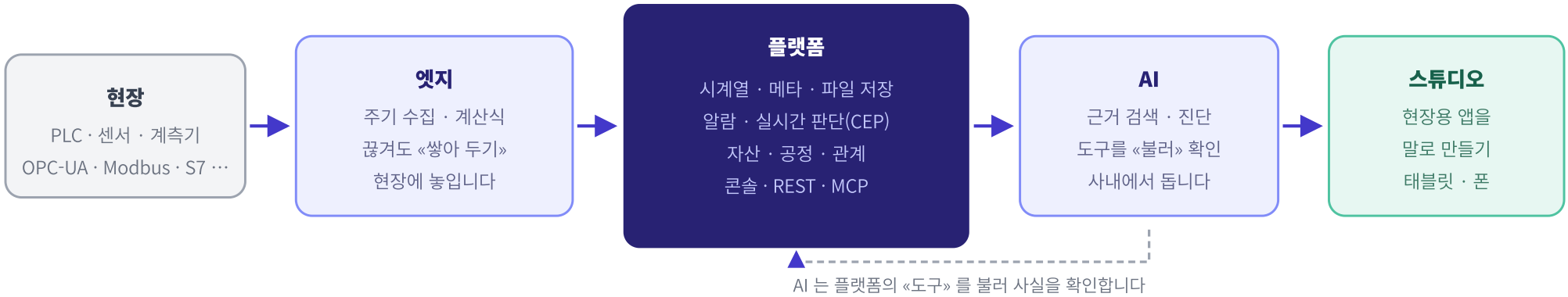
**0**
인터넷 요구 — 폐쇄망에서 그대로

**8**
쪽 · 2026.09

01 · 한 장으로

# 신호가 화면에 닿기까지 — 네 제품이 하는 일

한 문장으로 줄이면 이렇습니다 — **엣지가 모으고, 플랫폼이 저장하고 판단하고, AI 가 설명하고, 스튜디오가 화면으로 만듭니다.** 넷은 각자 따로 설치할 수 있지만 **데이터 원장은 하나**입니다.




**따로 설치되고, 따로 죽습니다**

넷은 각각 다른 상자에 설치됩니다. **하나가 멈춰도 나머지가 자기 일을 계속**합니다 — 플랫폼이 안 보여도 엣지는 계속 모으고, AI 가 없어도 콘솔은 그대로 돕니다.

**바깥으로 나가는 것이 없습니다**

넷 다 **고객 장비에서** 돕니다. 인터넷 요구가 없어 **폐쇄망에서 그대로** 서고, 모델도 데이터도 사내에 머뭅니다.

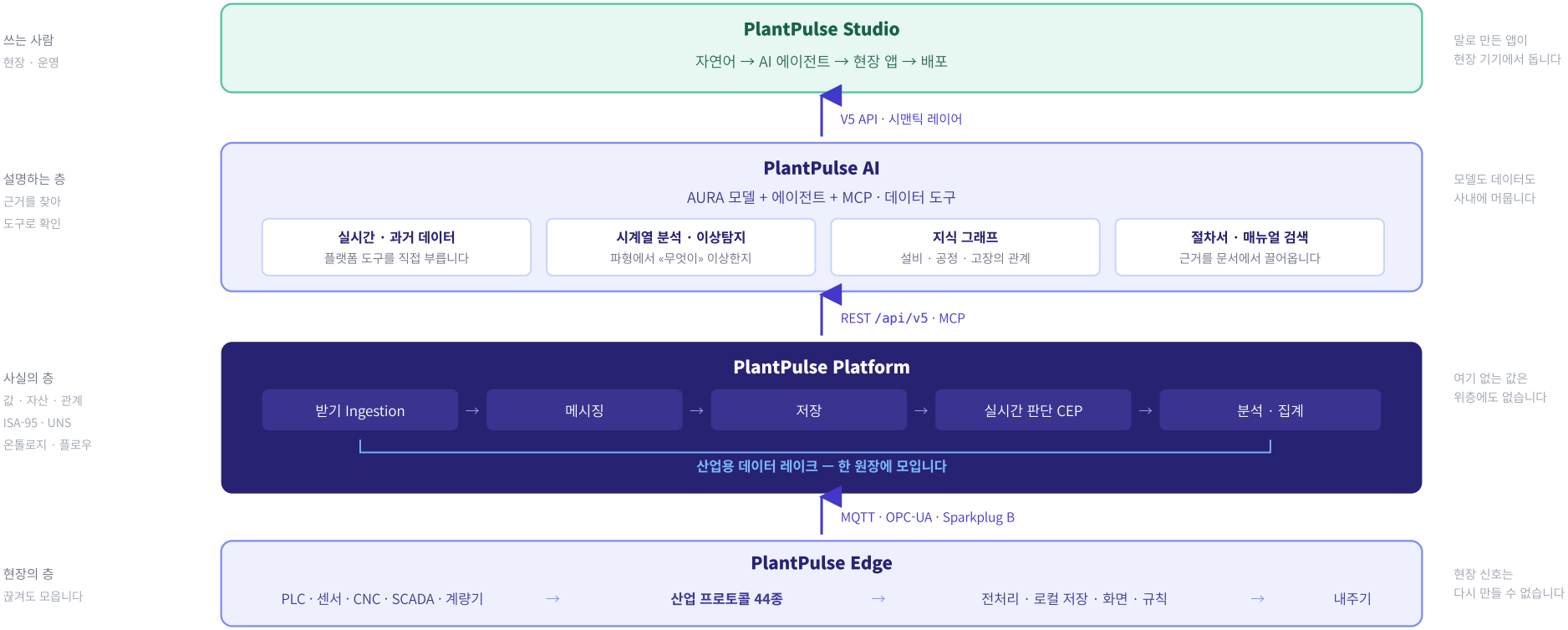
## 목차


02 스택 — 네 층, 아래가 위를 떠받친다 3
03 전체 아키텍처 — 구성요소와 그 사이의 길 4
04 기술 스택 — 무엇으로 만들었나 5
05 데이터가 흐르는 길 6
06 제품 경계 — 같은 일을 할 자리가 여럿일 때 7
07 끊기면 어떻게 되나 8
08 망 · 포트 · 방화벽 9
09 보안 · 계정이 지나는 지점 10
10 도입 순서 · 더 읽을 곳 11

02 · 스택

# 네 층 — 아래가 위를 떠받칩니다

현장에서 시작해 화면으로 끝납니다. **각 층은 아래층이 내준 것만** 쓰고, 층 사이를 잇는 것은 **표준 프로토콜과 공개된 API** 입니다 — 그래서 한 층을 갈아도 나머지가 버팁니다.




**층을 나눈 값어치는 «갈아끼울 수 있다» 는 것입니다.** 엣지만 먼저 깔아 현장 데이터를 모으다가 나중에 플랫폼을 붙여도 되고, AI 없이 플랫폼만 쓰다가 뒤에 얹어도 됩니다. **층 사이가 표준 프로토콜과 공개 API** 라서 가능한 일입니다 — 한 회사 제품끼리만 통하는 길로 이었다면 그 순서를 고를 수 없습니다.

03 · 전체 아키텍처

# 구성요소와 그 사이의 길

굵은 화살표가 **현장 데이터**가 가는 길입니다. 점선은 **물어보는 방향**(질의·도구 호출)이고, 괄호 안은 포트입니다. **회색은 사내망 안에서만** 쓰는 자리입니다.




**저장을 셋으로 나눈 이유.** 값은 끝없이 들어오고 설정은 거의 안 바뀌며 파일은 크기만 큽니다 — **성질이 다른 셋을 한 곳에 넣으면** 하나를 위한 선택이 나머지를 망칩니다. 대신 **입구는 하나**로 두어, 쓰는 쪽은 셋을 구별하지 않아도 됩니다.

04 · 기술 스택

# 무엇으로 만들었나 — 그리고 왜 그것인가

검증된 오픈소스를 **고르고 · 손보고 · 한 벌로 묶었습니다.** 직접 만든 것은 **산업 현장에서만 생기는 요구** 때문이고, 그렇지 않은 자리는 업계 표준을 그대로 씁니다.

## 엣지 — 현장에서 혼자 버티는 층

| | |
|---|---|
| **프로토콜** | **산업 프로토콜 44종** — OPC-UA · Modbus · MELSEC · S7 · LS · EIP · PROFINET · IEC 61850 · DNP3 · HART · FINS · BACnet · CANopen · M-Bus · SECS/GEM · FOCAS · MTConnect · ROS 2 … |
| **로컬 저장** | 끊긴 동안의 값을 **박스 안에** 쌓아 둡니다 |
| **현장 로직** | 계산식 · 규칙 · **Node-RED** 플로우 |
| **내주기** | OPC-UA 서버 · MQTT · **Sparkplug B** · REST · MCP |

## 플랫폼 — 데이터 원장

| | |
|---|---|
| **시계열** | **Aster TSDB** — Apache Cassandra 계열을 **KOPENS 가 유지**합니다. 끝없이 들어오는 쓰기를 견디는 쪽으로 다듬었습니다 |
| **메타 · 설정** | **PostgreSQL** — 자산 · 사용자 · 설정처럼 **정확히 한 번** 맞아야 하는 것 |
| **파일 · 이미지** | **S3 호환** 오브젝트 스토어 |
| **메시징** | **Aster MQTT** · **Apache Kafka** — 현장에서 오는 길과 내부에서 흐르는 길 |
| **실시간 판단** | **Aster CEP** — 흐르는 값 위에서 규칙을 겁니다 |
| **분석 · 배치** | **Apache Spark** · **Kyuubi** · **Gravitino**(카탈로그) · **Kestra**(워크플로) · **Temporal** |
| **캐시 · 대시보드** | **Valkey** · **Grafana** |

## AI — 사내에서 도는 층

| | |
|---|---|
| **모델** | **AURA** — 진단 · 검색 · 정밀 정렬. **KOPENS 가 데이터 설계부터 학습 · 평가 · 양자화까지** 직접 합니다 |
| **서빙** | 사내 GPU 위의 추론 엔진 + **OpenAI 호환 게이트웨이** — 기존 클라이언트를 그대로 씁니다 |
| **근거 검색** | 벡터 검색 + 재정렬. 절차서 · 규격 · 고장 이력이 대상입니다 |
| **지식 그래프** | 설비 · 공정 · 고장의 관계를 그래프로 |
| **도구 연결** | **MCP** — 플랫폼과 엣지의 도구를 그대로 부릅니다 |
| **관측** | 도구 실행 원장 · 트레이스 · 차단기 |

## 스튜디오 — 말로 만드는 층

| | |
|---|---|
| **앱 생성** | 자연어 → 에이전트 → **현장 앱**. 만든 앱은 태블릿·폰에서 바로 열립니다 |
| **저장** | **PostgreSQL** — 스택에 함께 옵니다 |
| **데이터** | 플랫폼 **V5 API** 로 읽습니다 — 별도 연결을 만들지 않습니다 |

## 따르는 표준

| | |
|---|---|
| **모델 · 구조** | **ISA-95** 계층 · **UNS**(통합 네임스페이스) · 온톨로지 관계 |
| **통신** | **OPC-UA** · **MQTT** · **Sparkplug B** · Modbus 외 41종 |
| **연동** | **OpenAPI** · **MCP** · OpenAI 호환 엔드포인트 |

**왜 어떤 것은 직접 만들었나.** 시계열 · 실시간 판단 · 메시징 셋은 **현장의 요구가 일반 제품과 다릅니다** — 초 단위로 끝없이 들어오고, 끊겼다 몰려오고, 순서가 뒤집히면 안 됩니다. 그래서 그 셋은 **오픈소스를 들여와 우리가 유지**합니다. 나머지는 **업계에서 검증된 것을 그대로** 씁니다 — 고칠 이유가 없는 자리를 고치면 **유지할 것만 늘어납니다.**

05 · 흐르는 길

# 센서 한 점이 화면에 닿기까지

값 하나가 지나는 자리를 순서대로 적었습니다. **어디서 지연이 생기고 어디서 쌓이는지**가 이 표에 다 들어 있습니다.

| 단계 | 어디 | 무엇이 일어나나 |
|---|---|---|
| ① **읽기** | 현장 → 엣지 | 설정한 **주기**로 PLC·계측기를 읽습니다. 프로토콜은 OPC-UA · Modbus · MELSEC · S7 · LS · EIP · HTTP 등 |
| ② **다듬기** | 엣지 | 계산식으로 **파생 값**을 만들고, 품질을 표시합니다 — **값이 0 인 것과 못 읽은 것**을 여기서부터 가릅니다 |
| ③ **보내기** | 엣지 → 플랫폼 | MQTT(1883 · TLS 8883) 또는 플랫폼 REST. SparkplugB 도 씁니다 |
| ④ **받기** | 플랫폼 | 메시징으로 받아 **태그 단위로 순서를 지켜** 처리합니다 |
| ⑤ **쌓기** | 플랫폼 저장소 | **시계열**은 Aster TSDB(9042) · **메타·설정**은 PostgreSQL(5432) · **파일·이미지**는 오브젝트 스토어(9000) |
| ⑥ **판단** | 플랫폼 | 알람 규칙과 **실시간 이벤트 처리(CEP, 7400)**. 집계·배치는 따로 돕니다 |
| ⑦ **꺼내기** | 플랫폼 | 시계열 질의 엔진(7800) · REST( `/api/v5` ) · 콘솔 화면 |
| ⑧ **설명** | AI | 문서·이력에서 근거를 찾고, **플랫폼 도구를 «불러»** 사실을 확인한 뒤 답합니다 |
| ⑨ **쓰기** | 스튜디오 | 현장이 쓸 화면을 만듭니다 — 태블릿·폰에서 열립니다 |

### 반대 방향 — 사람이 «내리는» 것

| | |
|---|---|
| **설비 명령** | 콘솔·API 에서 내려 엣지를 거쳐 PLC 로 갑니다. **읽기와 반대 방향**이라 통제도 반대로 걸립니다 |
| **AI 의 쓰기** | AI 가 값을 바꾸는 도구를 부를 때는 **사람 승인**을 지납니다. 승인이 없는 경로에는 **쓰기 도구를 열지 않습니다** |
| **설정 내려보내기** | 태그·연결 정의는 플랫폼에서 관리하고 엣지가 받아 갑니다 |

> **「0」과 「못 읽었다」를 처음부터 가릅니다.** 수집이 끊긴 것을 값 0 으로 적으면 **아무도 들여다보지 않은 설비가 「정상」으로 보고**됩니다. 엣지에서 품질을 표시하고, 저장·화면·AI 가 그 표시를 **끝까지 들고 갑니다.**

06 · 제품 경계

# 같은 일을 할 수 있는 자리가 여럿일 때

도입 검토에서 가장 자주 나오는 질문이 **「그건 어디서 하나요」**입니다. 넷이 겹쳐 보이는 자리를 골라 **무엇을 기준으로 고르는지**를 적었습니다.

| 하려는 일 | 할 수 있는 자리 | 무엇으로 고르나 |
|---|---|---|
| **값 가공 · 파생** | **엣지 계산식** / 플랫폼 처리 | **현장에서 바로 써야 하면 엣지**(끊겨도 돕니다). 여러 설비를 가로질러 보거나 보존이 필요하면 플랫폼 |
| **이상 감지** | **플랫폼 알람 · CEP** / AI | 규칙이 분명하면 **알람과 CEP**. 「왜 그런가」를 문장으로 받아야 하면 AI |
| **화면 만들기** | **플랫폼 콘솔** / 스튜디오 | 운영자가 쓰는 표준 화면은 콘솔. **현장 사람이 쓰는 전용 앱**은 스튜디오 |
| **문서·이력에서 찾기** | **AI 검색** | 절차서 · 규격 · 고장 이력이 대상일 때. 태그 값 조회는 플랫폼 쪽이 빠릅니다 |
| **외부 시스템 연동** | 플랫폼 REST · MCP / 엣지 REST · MCP | **전사 데이터가 필요하면 플랫폼, 그 박스 하나면 엣지**. 엣지는 플랫폼과 끊겨도 답합니다 |
| **AI 어시스턴트 붙이기** | 플랫폼 MCP / AI / 엣지 MCP | 자산·알람·온톨로지까지 보려면 플랫폼(도구 22종). 현장 박스 하나만 보면 엣지(읽기 6종) |

### 경계를 가르는 기준 셋

① **지연** — 현장에서 바로 반응해야 하는가, 몇 초 뒤라도 되는가. ② **보존** — 지나가면 그만인가, 몇 년을 두고 다시 보는가. ③ **권한** — 그 자리에서 값을 «쓸» 수 있어야 하는가.

### 둘 다 되는 것은 «둘 다» 두지 않습니다

같은 계산을 엣지와 플랫폼 양쪽에 두면 **두 값이 갈립니다.** 그리고 갈린 뒤에는 어느 쪽이 맞는지 아무도 모릅니다 — **한 곳을 정하고 나머지는 그것을 읽습니다.**

## 제품이 각각 «혼자서» 할 수 있는 것

| | |
|---|---|
| **엣지 단독** | 수집 · 계산 · 로컬 화면 · OPC-UA 서버로 내주기 · REST/MCP 응답. **플랫폼 없이도 섭니다** |
| **플랫폼 단독** | 저장 · 알람 · 콘솔 · 분석 · API. 엣지 없이 **외부 시스템이 직접 넣을** 수도 있습니다 |
| **AI** | 플랫폼이 있어야 «사실»을 확인합니다 — 도구로 값을 읽어 오기 때문입니다 |
| **스튜디오** | 플랫폼의 API 키와 AI 게이트웨이를 **둘 다** 봅니다(설치가 그 둘을 먼저 확인합니다) |

07 · 끊기면

# 선이 끊겼을 때 무엇이 남나

공장 망은 끊깁니다. 그래서 **끊긴 동안 무엇이 계속되고, 붙은 뒤 무엇이 따라오는지**를 설계에 넣었습니다.

| 끊긴 구간 | 그동안 | 붙은 뒤 |
|---|---|---|
| **엣지 ↔ 플랫폼** | **수집은 계속됩니다.** 엣지가 로컬에 쌓아 둡니다 | **쌓아 둔 것을 보냅니다** — 빈 구간이 메워집니다 |
| **엣지 ↔ PLC** | 그 연결만 끊깁니다. 다른 연결은 그대로 돕니다 | 자동으로 다시 붙습니다. 끊긴 구간은 **「못 읽음」으로 남습니다** — 0 으로 채우지 않습니다 |
| **AI ↔ 플랫폼** | AI 가 **사실 확인을 못 합니다.** 그래서 지어내지 않고 **그렇다고 말합니다** | 바로 정상화됩니다 — AI 는 상태를 들고 있지 않습니다 |
| **스튜디오 ↔ 플랫폼** | 만들어 둔 앱은 뜨지만 **현장 값이 안 옵니다** | 바로 정상화됩니다 |

**엣지의 «쌓아 두기» 가 이 구조의 핵심입니다**

현장 신호는 **다시 만들 수 없습니다.** 그 순간을 놓치면 영영 없습니다 — 그래서 보내는 쪽이 아니라 **모으는 쪽에** 저장을 둡니다. 플랫폼이 몇 시간 없어도 **그 시간의 데이터가 빈칸으로 남지 않습니다.**

**이중화 — 쓰기 권한은 «한 쪽만»**

엣지를 둘로 둘 때, **PLC 에 쓰는 권한은 한 쪽만** 가집니다. 둘 다 가지면 같은 설비에 두 곳에서 씁니다. 그래서 권한 이동은 **내리고 → 확인하고 → 올리는** 순서로 합니다.

## 막히면 어떻게 보이나 — 운영자가 먼저 볼 곳

| | |
|---|---|
| **엣지 상태** | 엣지 콘솔의 연결 목록과 **건강 상태**(구성요소별 UP/DOWN) |
| **플랫폼 유입** | 콘솔의 **모니터링** — 초당 유입 · 큐 · 저장소 상태를 한 화면에서 봅니다 |
| **전송 상태** | 엣지의 전송 현황 — **밀린 양**이 보입니다 |

**「영향 없음」과 「모름」은 다릅니다.** 설비 사이의 흐름을 그려 두지 않으면 「이 설비가 서면 어디까지 멈추나」에 **빈 답**이 옵니다. 그것은 영향이 없다는 뜻이 아니라 **그려 둔 것이 없다는 뜻**입니다 — 그래서 두 상태를 화면과 API 가 구별해 알려 줍니다.

08 · 망과 포트

# 무엇이 어디로 붙나 — 한 장으로

방화벽을 설계할 때 필요한 것만 모았습니다. **밖에 열어야 하는 것**과 **안에서만 쓰는 것**을 갈라 두었습니다.

## 사람이 쓰는 입구

| 제품 | 포트 | 무엇 |
|---|---|---|
| **플랫폼** | **80 · 443** | 콘솔과 REST API. **웹 화면과 API 가 같은 주소**입니다 |
| **엣지** | **80 · 443** | 엣지 콘솔과 `/api/v1`. 외부 호출은 **HTTPS 를 강제**합니다 |
| **AI** | **80 · 443** | 채팅 화면 — **유일한 외부 노출**입니다 |
| **스튜디오** | **5171** | 만든 앱이 열리는 자리. **막으면 프리뷰와 배포앱이 안 열립니다** |

## 제품 사이가 쓰는 길

| | |
|---|---|
| **엣지 → 플랫폼** | **1883**(MQTT) 또는 **8883**(TLS), 그리고 플랫폼 **443** |
| **AI → 플랫폼** | 플랫폼 **443** — REST 와 MCP 를 같은 입구로 씁니다 |
| **스튜디오 → 플랫폼** | 플랫폼 **443**(API 키) |
| **스튜디오·AI → 모델** | 사내 모델 게이트웨이(**4000**). **바깥으로 나가지 않습니다** |
| **외부 SCADA → 엣지** | **12000 · 12443** — 엣지가 OPC-UA 서버로 내줄 때만 |

🔴 **설치 직후 상태는 «넓게 열림» 입니다 — 좁히는 것이 운영자의 몫입니다.** 설치기는 동작에 필요한 포트를 방화벽에 등록합니다. 그중 **데이터베이스 · 메시징 · 내부 리스너**는 대개 **사내망 안에서만** 필요합니다. 도입 때 한 번 훑어 **밖에 열어 둘 이유가 있는 것만** 남기십시오 — 제품별 문서에 **포트마다 「밖에 열어 둘 이유가 있나」**를 적어 두었습니다.

## 폐쇄망

넷 다 **인터넷 요구가 없습니다.** 설치는 오프라인 번들로 하고, 모델도 사내에서 돕니다. 바깥으로 나가는 통신이 없다는 것은 **망 설계에서 가장 먼저 확인되는 조건**이라 여기 적어 둡니다.

09 · 보안

# 계정과 키가 지나는 지점

어디서 무엇으로 인증하는지, 그리고 **도입 때 반드시 바꿔야 하는 것**을 모았습니다.

| 구간 | 무엇으로 | 알아 둘 것 |
|---|---|---|
| **사람 → 플랫폼** | 콘솔 로그인 · 역할 | 역할에 따라 쓰기 권한이 갈립니다 |
| **프로그램 → 플랫폼** | 발급 토큰을 **키로 교환** | **성공 판정이 HTTP 200 이 아닙니다** — 응답의 상태와 키를 함께 봅니다. 그리고 **503 은 「키가 틀렸다」가 아닙니다** |
| **프로그램 → 엣지** | `X-API-Key` 또는 Bearer | 기본이 **HTTPS 강제**라, 평문으로 부르면 키가 맞아도 거부됩니다 |
| **AI → 플랫폼** | 플랫폼 API 키 | 도구가 **읽기 위주**입니다. 값을 바꾸는 도구는 **사람 승인**을 지납니다 |
| **→ 모델 게이트웨이** | 게이트웨이 키 | 스튜디오는 **설치할 때** 이 키와 모델 이름을 실제로 호출해 확인합니다 |

### 도입 때 반드시 바꾸는 것

제품마다 **초기 비밀번호**가 있고, 문서에 그대로 적혀 있습니다. **숨기지 않는 이유**는 숨겨도 설치본에 들어 있고, 적어 두어야 **바꾸라고 말할 수 있기** 때문입니다 — 제품별 「초기 비밀번호 변경」 문서가 커미셔닝 필수 절차로 있습니다.

### 거부는 «설치 전» 에 나는 편이 낫습니다

스튜디오 설치는 **도커를 깔기 전에** 받은 자격증명으로 실제 호출을 해 봅니다. 틀린 키면 **그 자리에서 멈춥니다** — 「설치는 끝났는데 첫 사용에서 전부 실패」를 막습니다. 다만 **서버가 답을 못 낸 경우(5xx)는 멈추지 않습니다**. 점검 중인 서버를 「틀린 키」로 몰지 않기 위해서입니다.

## 감사와 기록

| | |
|---|---|
| **플랫폼** | API 호출과 변경이 감사에 남습니다. **AI 가 알람을 확인 처리**할 때도 사유를 남깁니다 |
| **엣지** | REST 호출이 접근 로그에 남고, **쓰기 권한 변경은 별도로** 기록됩니다 |
| **AI** | **도구 실행 원장**이 따로 있습니다 — 무엇을 어떤 인자로 불렀는지 |

10 · 도입 순서

# 무엇부터 깔고, 어디서 더 읽나

넷을 한꺼번에 깔 필요가 없습니다. **앞이 뒤의 전제**라서 순서가 곧 의존 관계입니다.

| 순서 | 무엇 | 끝났다고 볼 수 있는 지점 |
|---|---|---|
| 1 | **플랫폼** | 콘솔에 로그인되고 **초기 비밀번호를 바꿨다**. 저장소가 전부 올라왔다 |
| 2 | **엣지** | 플랫폼에 등록되고, **연결 하나가 값을 올린다**. 끊었다 붙여 **쌓인 것이 따라오는지**까지 본다 |
| 3 | 자산 · 태그 · 흐름 | 설비 계층이 서고, **공정 흐름을 그려 두었다** — 하류 영향 조회의 전제입니다 |
| 4 | 알람 · 화면 | 운영자가 **매일 보는 화면**이 정해졌다 |
| 5 | **AI** | 모델 게이트웨이가 서고, **플랫폼 도구를 부를 수 있다** |
| 6 | **스튜디오** | 현장용 앱 하나가 **태블릿에서 열린다** |

**2번에서 «끊었다 붙이는» 시험을 꼭 하십시오.** 평소에는 드러나지 않고 **정작 필요할 때 드러나는** 기능입니다. 도입 검토 단계에서 한 번 확인해 두면, 나중에 「그때 데이터가 왜 비었나」를 따질 일이 없습니다.

## 제품별 문서

| | | | |
|---|---|---|---|
| **플랫폼** | docs.plantpulse.io /ko/platform | **데이터레이크 기술 백서** | /ko/platform/whitepaper |
| **엣지** | docs.plantpulse.io /ko/edge | **AURA 모델 기술 백서** | /ko/ai/overview/whitepaper |
| **AI** | docs.plantpulse.io /ko/ai | **공정 다이어그램** | /ko/platform/user-guide/process |
| **스튜디오** | docs.plantpulse.io /ko/studio | **MCP 서버** | /ko/platform/api/mcp · /ko/edge/api/mcp |

## 문의

| | |
|---|---|
| **전화 · 이메일** | 1588-5268 · webmaster@kopens.com |
| **문의 · 데모** | kopens.com/ko/contact |

㈜한국오픈솔루션 · KOPENS, Inc. © 2026 KOPENS. 플랜트펄스®는 ㈜한국오픈솔루션의 등록상표입니다. 본 문서 Rev. 2026.09.