

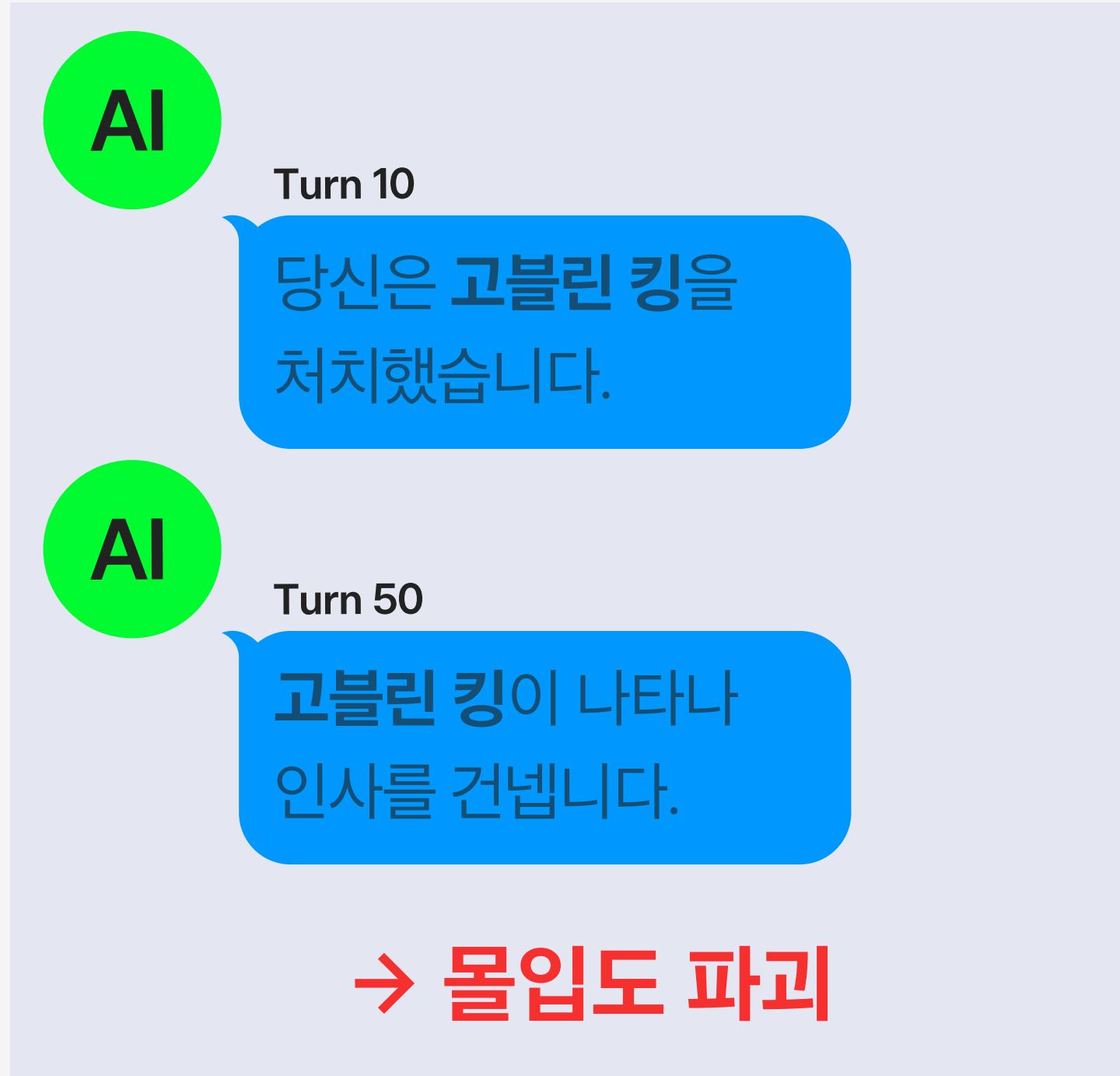
Project YEOUL

Multi-Agent AI Platform for Branching Narrative Orchestration

팀명	Team GARAM
팀원	김세영(TL), 안재현, 정진웅



✓ 기존 생성형 AI 서사의 치명적 한계



Point 1

서사 붕괴

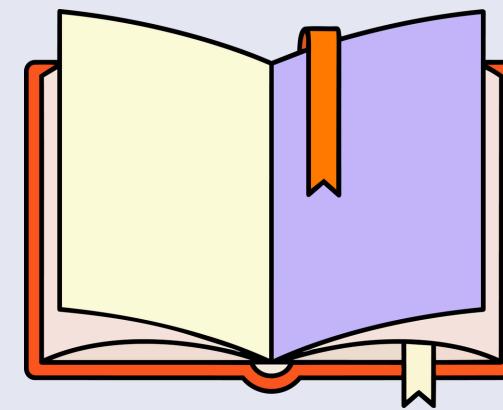
- 대화가 길어질수록 초반 설정을 망각함
- # 맥락 상실 # 장기 기억력 부재 # 일관성 결여

Point 2

환각 현상

- 세계관 규칙을 무시하고 논리적 모순 발생
- # 죽은 NPC의 부활 # 존재하지 않는 아이템 언급

✓ 왜 '여울'인가? - 멀티 에이전트 조율 엔진

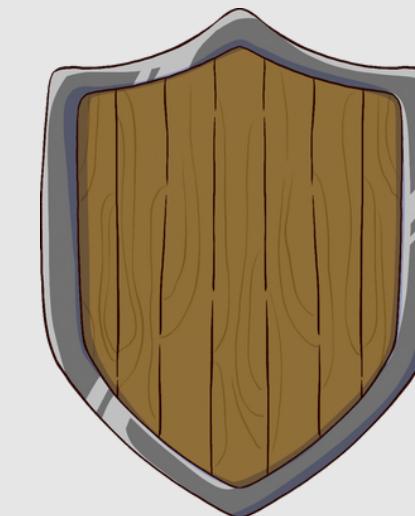


Narrator Agent

전지적 상황 묘사 및 페르소나 유지

Validator Agent

실시간 서사 정합성 및 오류 검수



World Manager

규칙 기반의 '절대적 진실' 수치 관리

Orchestration

상태(State) 기반의 유기적 협업 제어



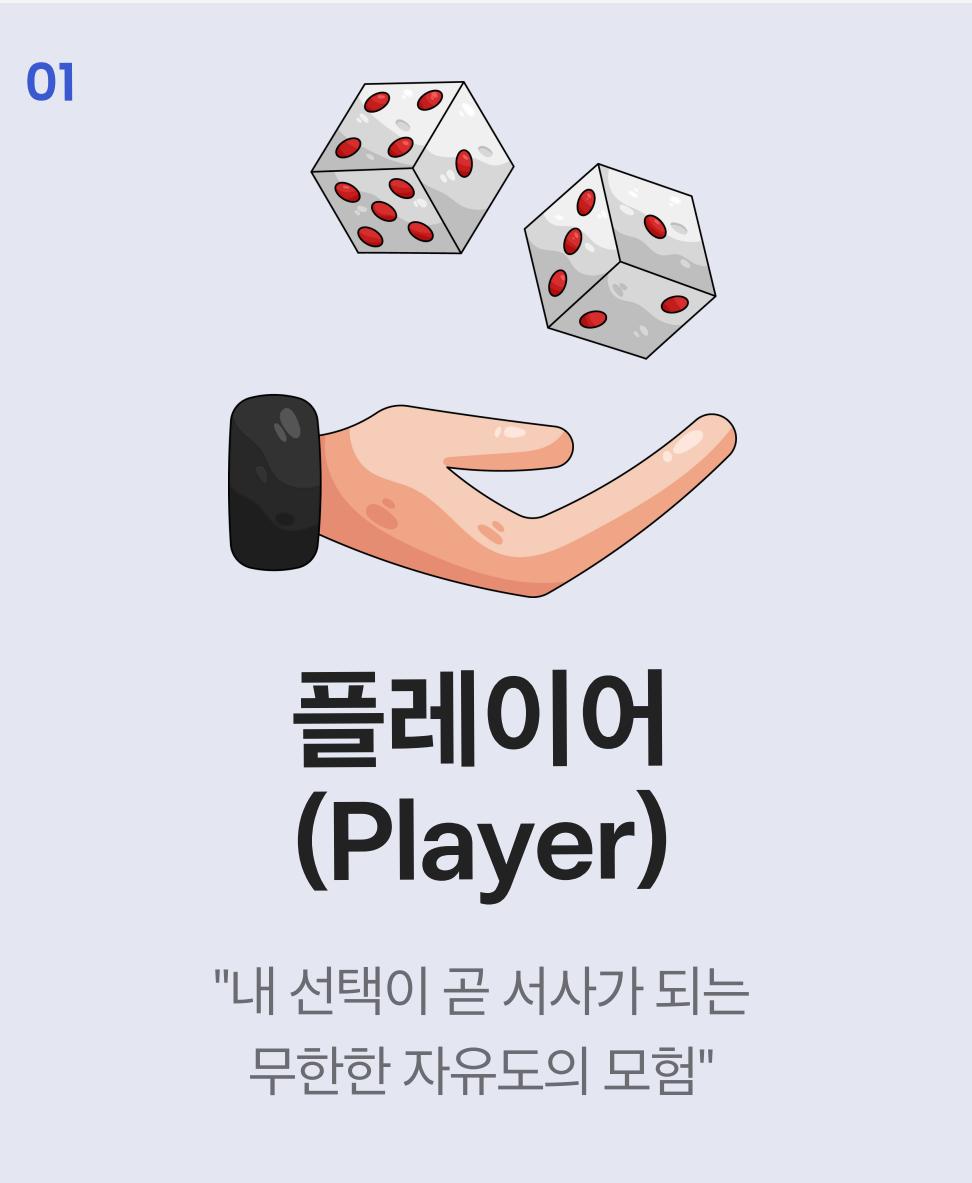
“



"상상을 서사로, 선택을 현실로"

완벽한 서사 생성 및 플레이를 위한
멀티 에이전트 프로젝트

✓ 타겟 유저



✓ 프로젝트 목표

01

정성적 목표

- **AI를 넘어선 몰입감:** 단순히 기계와 대화하는 것이 아닌, 실제 던전 마스터(DM)와 함께 모험하는 듯한 깊은 **서사적 몰입**을 제공합니다.
- **무결한 세계관 유지:** 멀티 에이전트의 상호 검증을 통해 설정 붕괴와 환각 현상이 없는 일관된 이야기를 보장합니다.

02

정량적 목표

- **실시간 응답 속도:** LLM의 스트리밍(SSE) 최적화를 통해 턴당 평균 **응답 속도 3초 이내**를 유지하여 쾌적한 플레이 환경을 구축합니다.
- **서사 오류 발생률 0% 지향:** Validator 에이전트의 자동 검수 루프를 통해 시나리오 **논리 모순 발생 건수를 제로화**하는 것을 목표로 합니다.
- **데이터 정합성 100%:** WorldState 엔진을 통해 캐릭터의 **상태** 및 아이템 **수치가 서사와 100% 일치**하도록 강제합니다.

INDEX

목차페이지

01	프로젝트 배경 및 팀 소개	01p
02	개발 계획 및 기술 스택	09p
03	서사 데이터 및 상태 관리 설계	16p
04	멀티 에이전트 아키텍처	25p
05	시스템 로직 및 내러티브 엔진	34p
06	시스템 아키텍처	42p
07	서비스 구현 및 시연	49p
08	트러블슈팅 및 향후 계획	56p



김세영



안재현



정진웅

Architecture & Scenario Design

- 전체 아키텍처 구상 및 멀티 에이전트 시스템 기획
- 시나리오 빌드 에이전트 및 노드 GUI 기반 프론트엔드 개발
- 실시간 토큰 소모 계산 로직 개선 및 CBT 모집·관리 총괄

Engine & Infrastructure

- LangGraph 워크플로우 및 계층형 Intent Parser 정합성 로직 구현
- SSE 실시간 스트리밍 및 WorldState 수치 데이터 동기화 프로세스 구축
- Railway 클라우드 배포 및 GitHub Actions 기반 CI/CD 파이프라인 총괄

Auth & Service Optimization

- OAuth 2.0 기반 인증 시스템 구축 및 소셜 로그인 세션 안정화
- 메인·마이페이지 반응형 UI/UX 및 인터랙티브 레이아웃 구현
- 시나리오 랭킹 알고리즘 설계 및 실시간 조회수 시스템 개발



02



개발 계획

기획부터 실서비스 배포까지,
'여울'이 탄생하기 위한
기술적 여정과 협업의 기록

✓ 전체 프로젝트 로드맵

기획 및 아키텍처 설계

멀티 에이전트 아키텍처 구상
시나리오 데이터 규격 설계

2025.12



2025.11 ~ 12



핵심 엔진 및 빌더 개발

LangGraph 기반 워크플로우 설계
노드 GUI 시나리오 빌더 개발

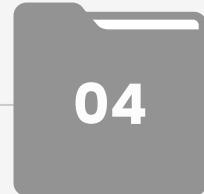
서비스 고도화 및 최적화

SSE 실시간 스트리밍 구축
Mermaid.js 연동 서사 시각화 고도화

현재



2026.01



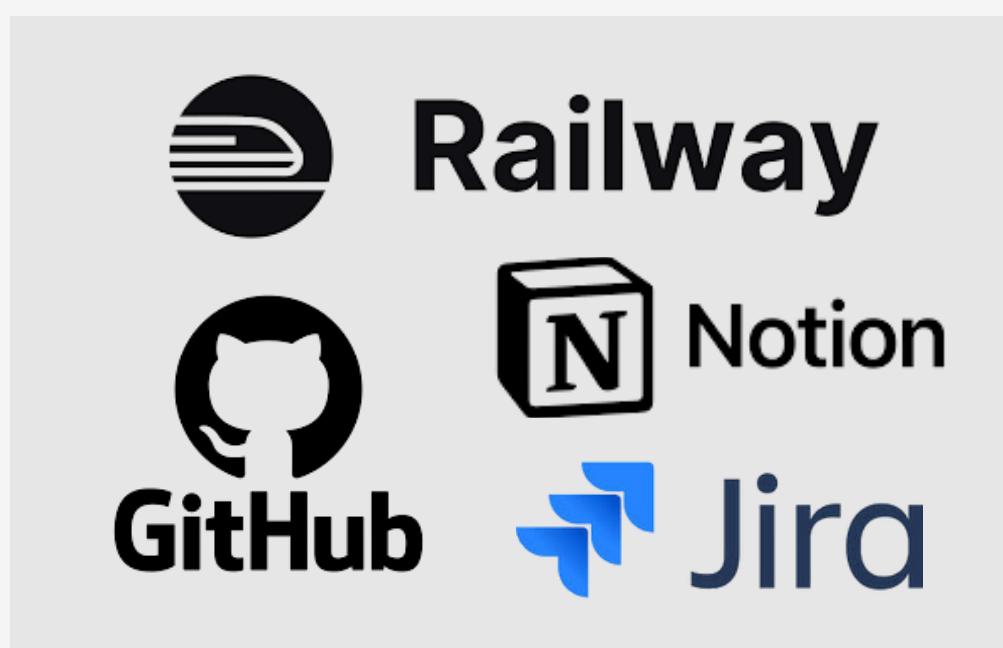
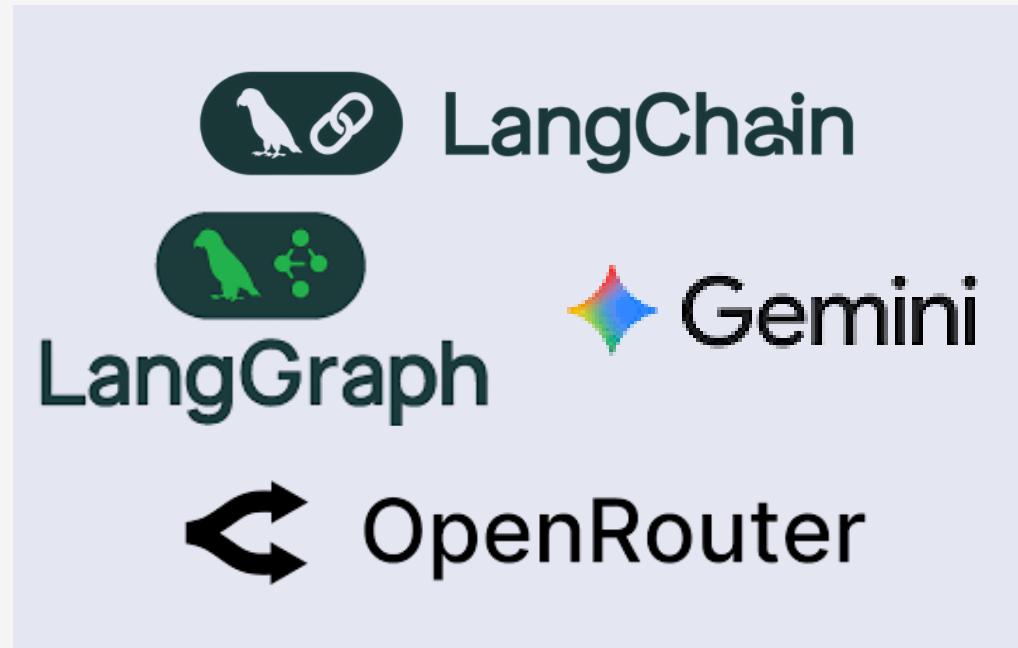
배포 및 실서비스 운영

Railway 클라우드 실서비스 배포
CI/CD 파이프라인 자동화 구축

✓ 상세 일정 및 마일스톤

간트차트

✓ 기술 스택



✓ 기술 선정 사유 1 (FastAPI)

“

고성능 비동기와
실시간 스트리밍을
위한 최적의
AI 생태계 접점

Python Ecosystem:

LangChain, OpenAI 등 최신 AI 라이브러리와의 완벽한 연동

Async Performance:

async 기반 비동기 처리로 LLM 생성 대기 시간 효율적 관리

Real-time Streaming:

SSE 지원을 통한 끊김 없는 텍스트 출력 구현

✓ 기술 선정 사유 2 (LangGraph)

“

상태 기반의 순환과
분기 제어로 완성한
정교한 서사 조율

Beyond Linear:

단순 선형 구조를 넘어 복잡한 서사 분기와 순환 로직 구현

Async Performance:

전역 상태를 기반으로 턴제 TRPG의 정교한 규칙 관리 및 유지

Real-time Streaming:

플레이어 행동에 따른 서사 반복 및 특정 시점 회귀 완벽 제어

✓ 협업 프로세스

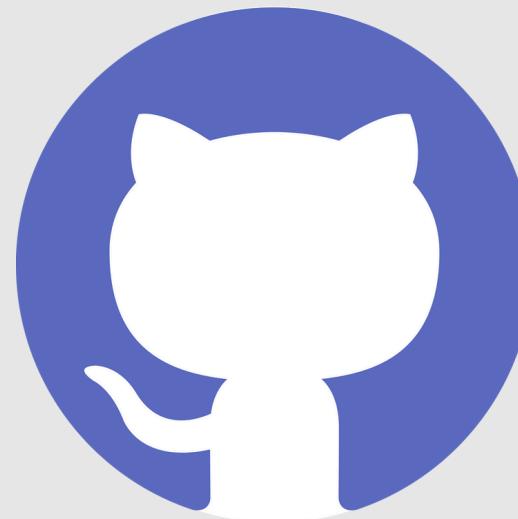
01



Jira
(Agile Workflow)

마일스톤 중심의 티켓 시스템으로
전체 공정률 실시간 추적

02



Git
(Code Integrity)

기능별 독립 브랜치 전략과 리뷰로
시스템 전반의 코드 정합성 유지

03



Notion
(Knowledge Sync)

개발 로그 및 API 명세를 공유하여
팀원 간 기술적 간극을 근본적으로 제거



03

데이터 및 상태 설계

비정형 서사를 완벽하게 제어하는
정형 데이터 관리 전략

✓ 데이터 설계 개요

"플레이어는 버려진 기업의 비밀 실험실 '랩 03'에 침투하여 메인 서버의 실험 기록을 삭제해야 합니다. 낡은 안드로이드 G-72와 육중한 보안 로봇 스크랩 스매셔의 위협을 헤쳐나가며 임무를 완수해야 합니다. 공기는 차갑고, 정체불명의 기계 소음이 신경을 긁는 가운데 입구에서 G-72가 당신을 기다립니다."

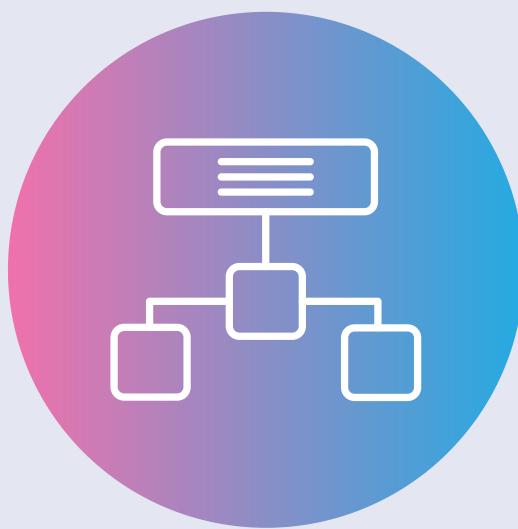
```
{  
    "scenario": {  
        "title": "데이터 말소 작전: 랩 03",  
        "npcs": [{"name": "G-72", "isEnemy": false, "...": "..."}],  
                // AI가 아닌 '시스템'이 적대 여부 결정  
        "scenes": [{"scene_id": "Scene-1",  
                    "trigger": "[관리자 키카드] 소지 시에만 이동 가능",  
                // 서사 분기를 결정하는 핵심 데이터  
                    ...  
                }]}  
}
```



Strict Schema Strategy: 비정형 서사를 정형 데이터로 제어

✓ Scenario Schema 설계

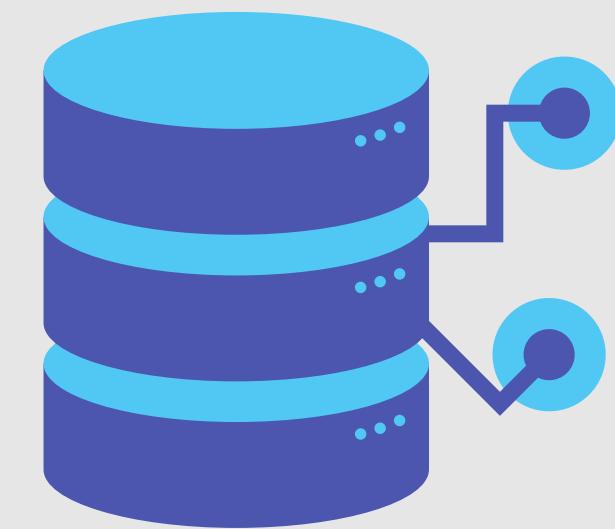
01



Data Hierarchy

Root 모델 중심의
계층적 시나리오 구조화

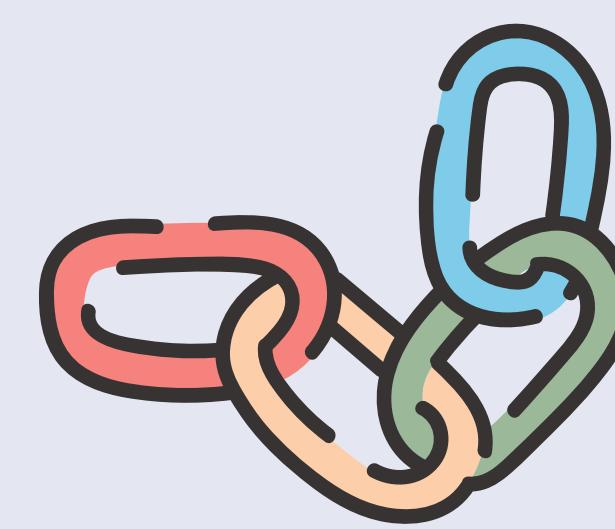
02



World State

HP, 골드 등 실시간 변동 수치
데이터 관리

03

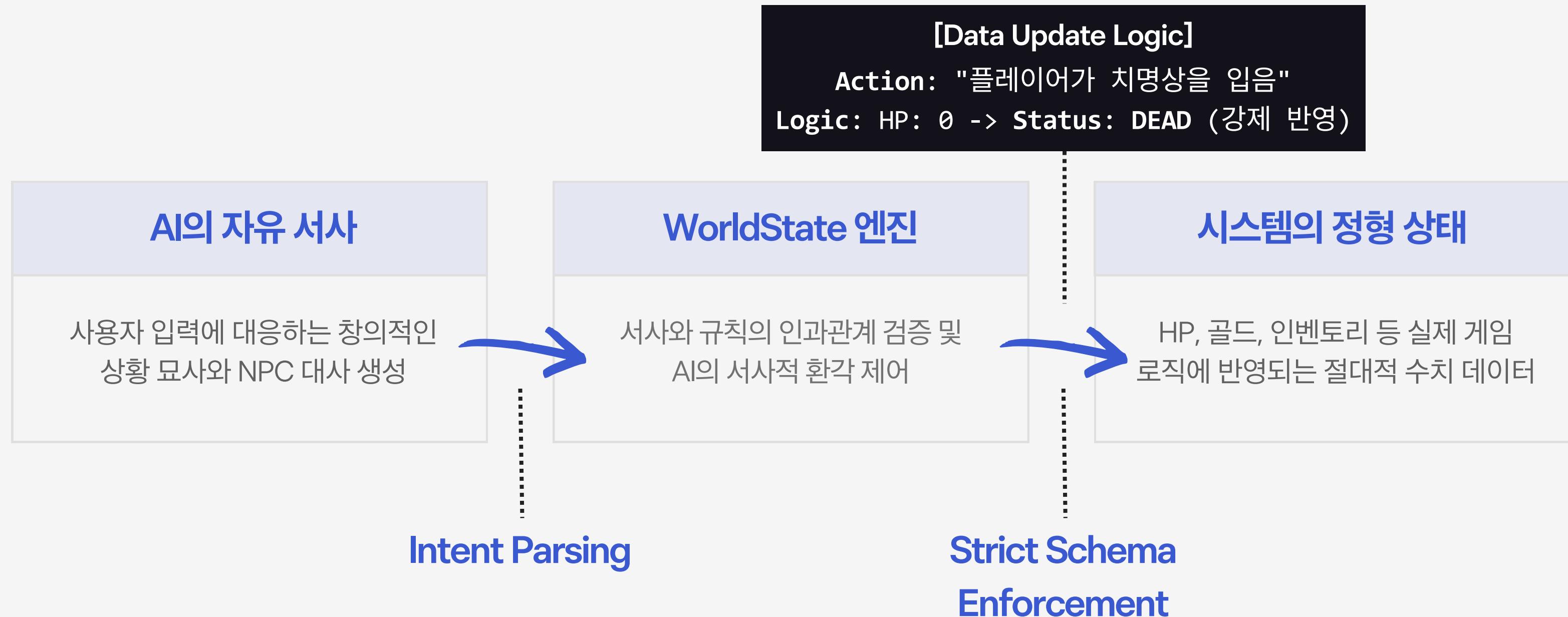


Logic Chain

Condition -> Action -> Effect
인과관계 규격화

Pydantic 기반 객체 모델링으로 구현한 타입 안정적 서사 규격

✓ WorldState 엔진 설계



서사는 AI가 묘사하지만, 규칙과 수치는 코드가 지배한다

✓ 데이터 영속화 전략

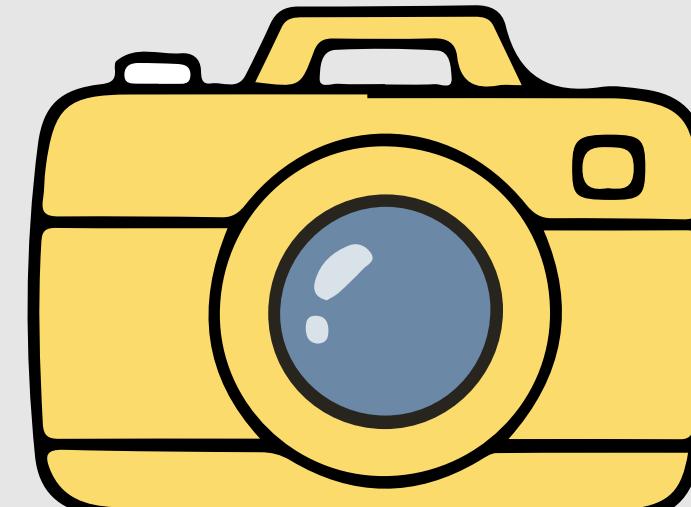
01



Hybrid Storage

PostgreSQL JSONB를 활용한 정형/비정형 데이터 통합 관리

02



State Snapshot

모든 행동 직후 실시간 스냅샷 기록으로 0% 데이터 유실 실현

03

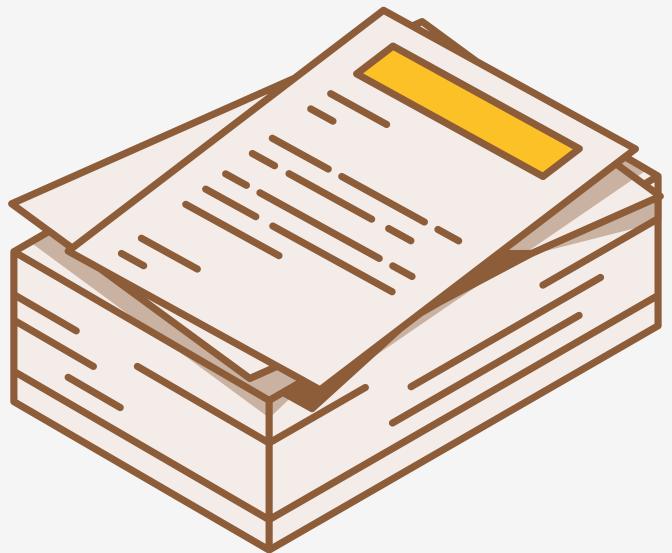


Integrity Check

DB-객체 간 1:1 매칭 및 로드 시 실시간 스키마 검증 수행

실시간 스냅샷과 엄격한 스키마 검증으로 구축한 끊김 없는 게임 환경

✓ RAG 지식베이스 구축



방대한 세계관 데이터



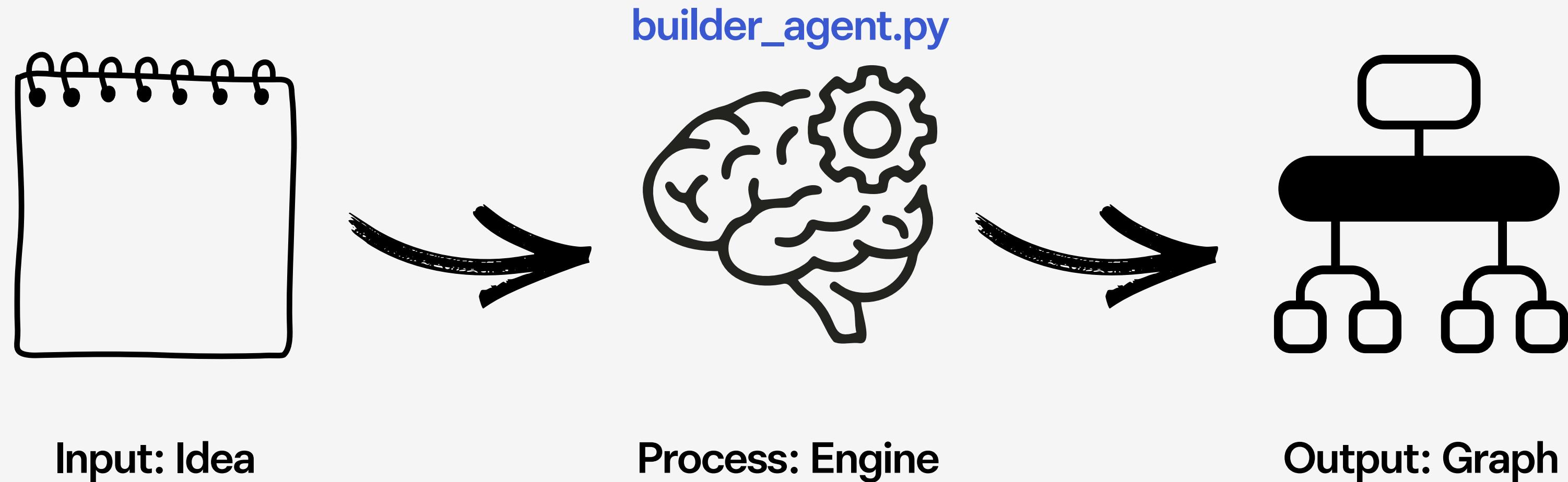
Qdrant (Vector DB)



최적화된 컨텍스트

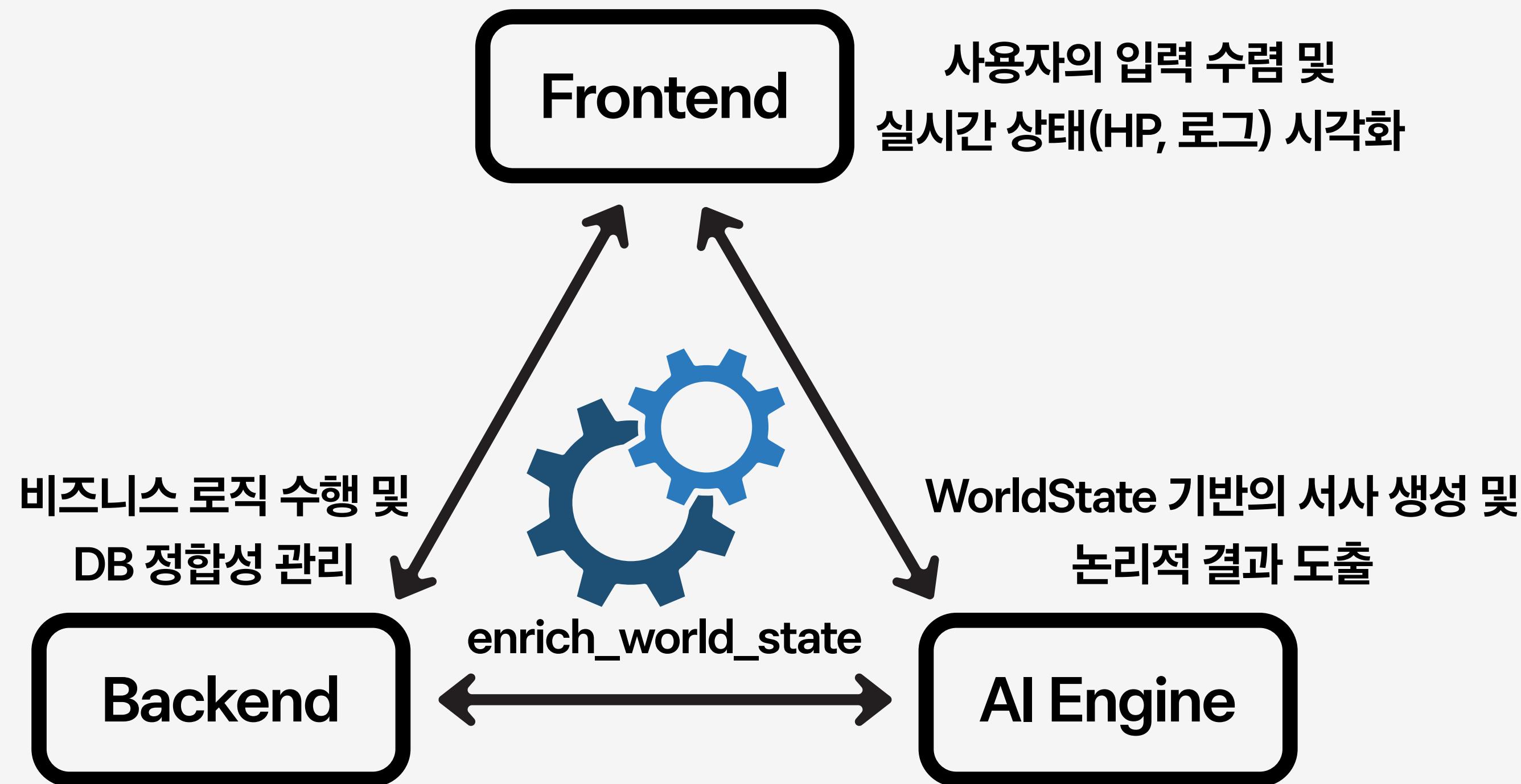
불필요한 노이즈는 제거하고, LLM에는 오직 핵심 지식만을 전달

✓ 데이터 변환 로직



비정형 아이디어를 정교한 서사 그래프로 치환하는 지능형 데이터 빌더

✓ 데이터 동기화 프로세스

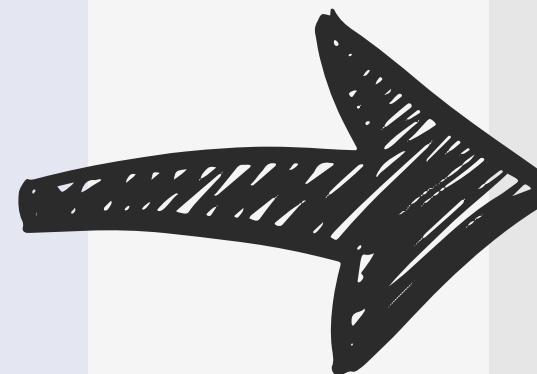


✓ 서사 컨텍스트 주입

01

시스템 데이터 (JSON)

```
{  
  "world_state": {  
    "hp": 10,  
    "max_hp": 100,  
    "location": "불타는 성곽",  
    "status": ["중독됨"]}  
}
```



Context Injection

02

AI 마스터의 서사

"매캐한 연기가 자욱한 **불타는 성곽**의 중심에서
당신은 무거운 몸을 이끌고 비틀거립니다.

전체 100의 생명력 중 단 **10만**이 남은 절체절명의
위기, 설상가상으로 온몸에 퍼진 **중독**의 고통이
당신의 감각을 마비시키기 시작합니다."

데이터를 '절대적 진실'로 규정하여 AI의 **환각(Hallucination)**을 원천 차단

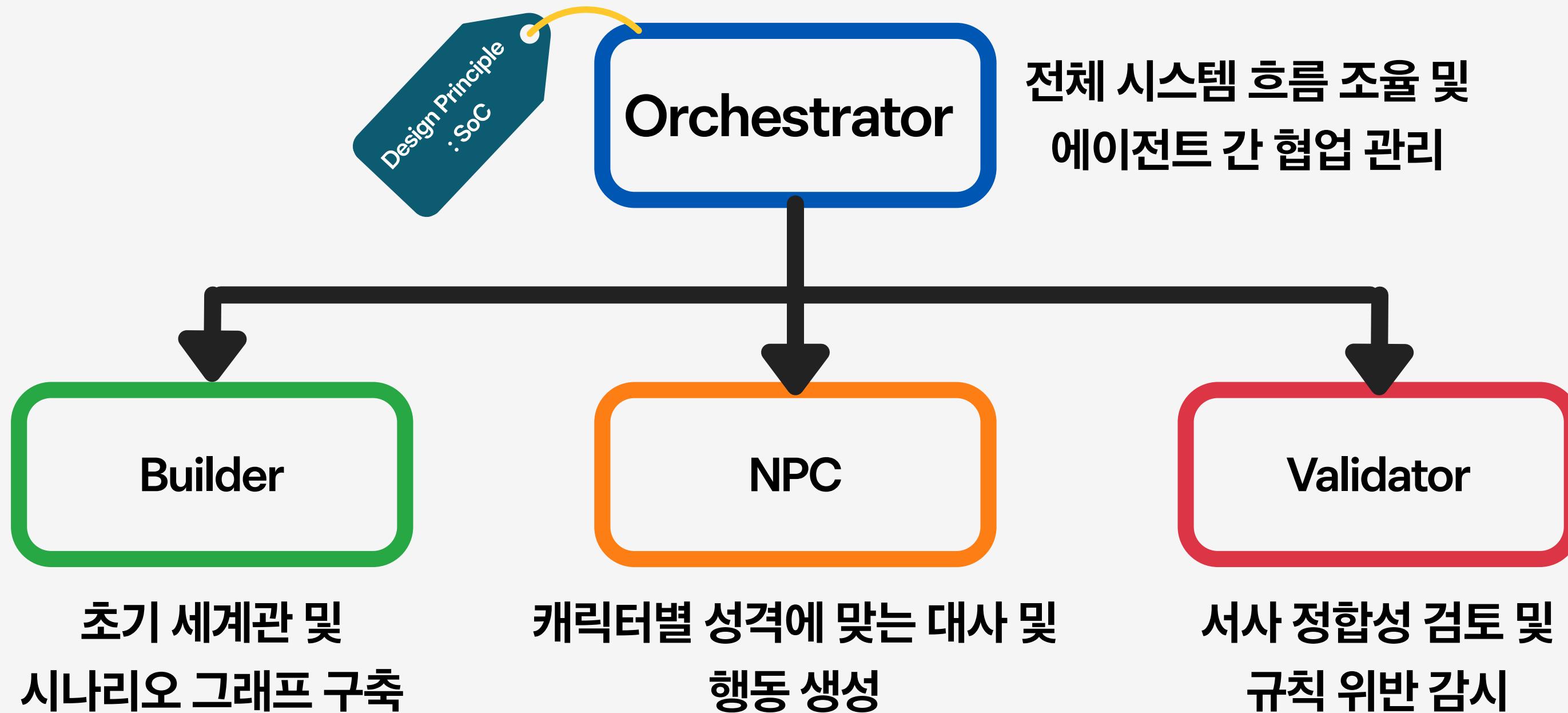


04

핵심 에이전트

시나리오 설계, 서사 전개, 정합성 검증을
자율적으로 수행하는 세 가지 핵심 AI 에이전트의
협업 체계

✓ 에이전트 아키텍처 개요



✓ Orchestrator 에이전트

1 메인 그래프 운영

game_engine.py에 구현된 메인 그래프 로직이 시스템의 전체 워크플로우를 실시간으로 제어합니다.

2 지능형 업무 할당

사용자의 입력과 현재 게임의 WorldState를 분석하여, 서사·NPC·검증 등 최적의 에이전트에게 업무를 배분합니다.

3 결과 취합 및 전달

각 에이전트로부터 전달받은 파편화된 데이터를 하나의 완성된 시나리오 패키지로 취합하여 클라이언트에 최종 전달합니다.

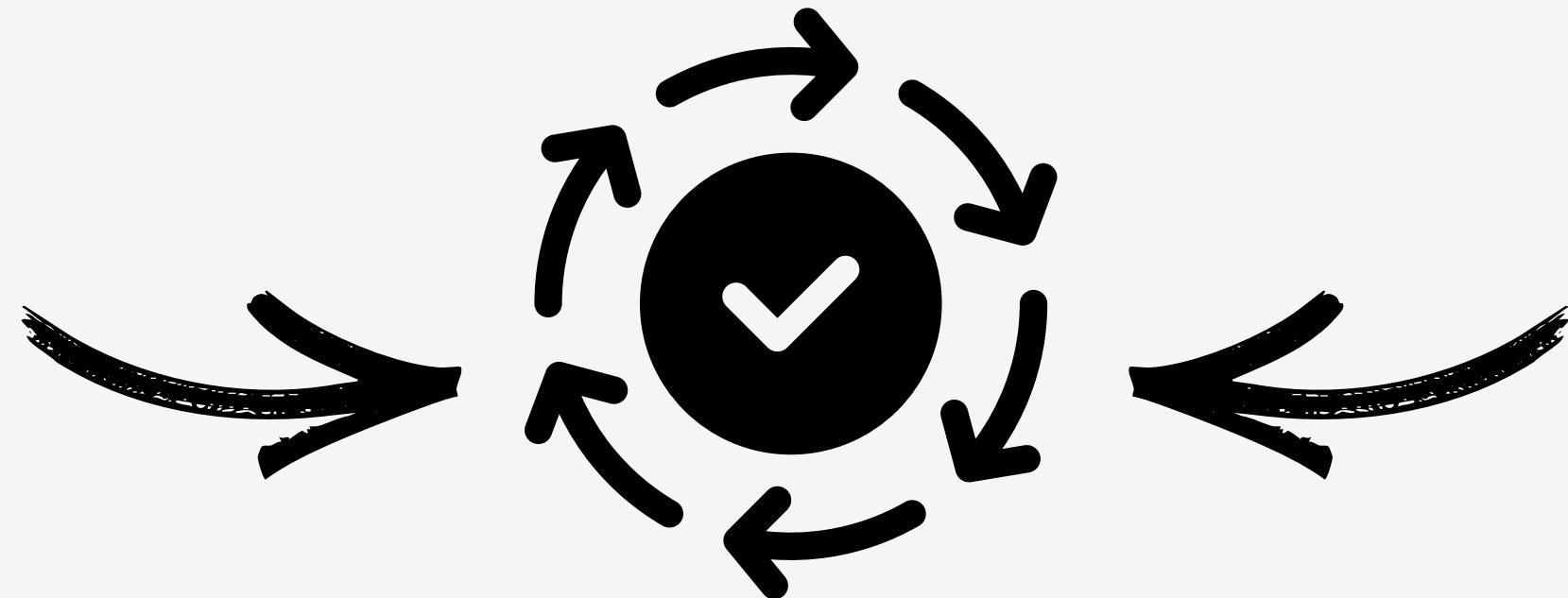
4 컨트롤 타워

시스템 전반의 상태를 유지하며, 에이전트 간의 통신이 끊김 없이 유기적으로 흐르도록 조율하는 가교 역할을 수행합니다.

✓ NPC & Narrator 에이전트



Narrator
(전지적 서사 가이드)



일관성 유지



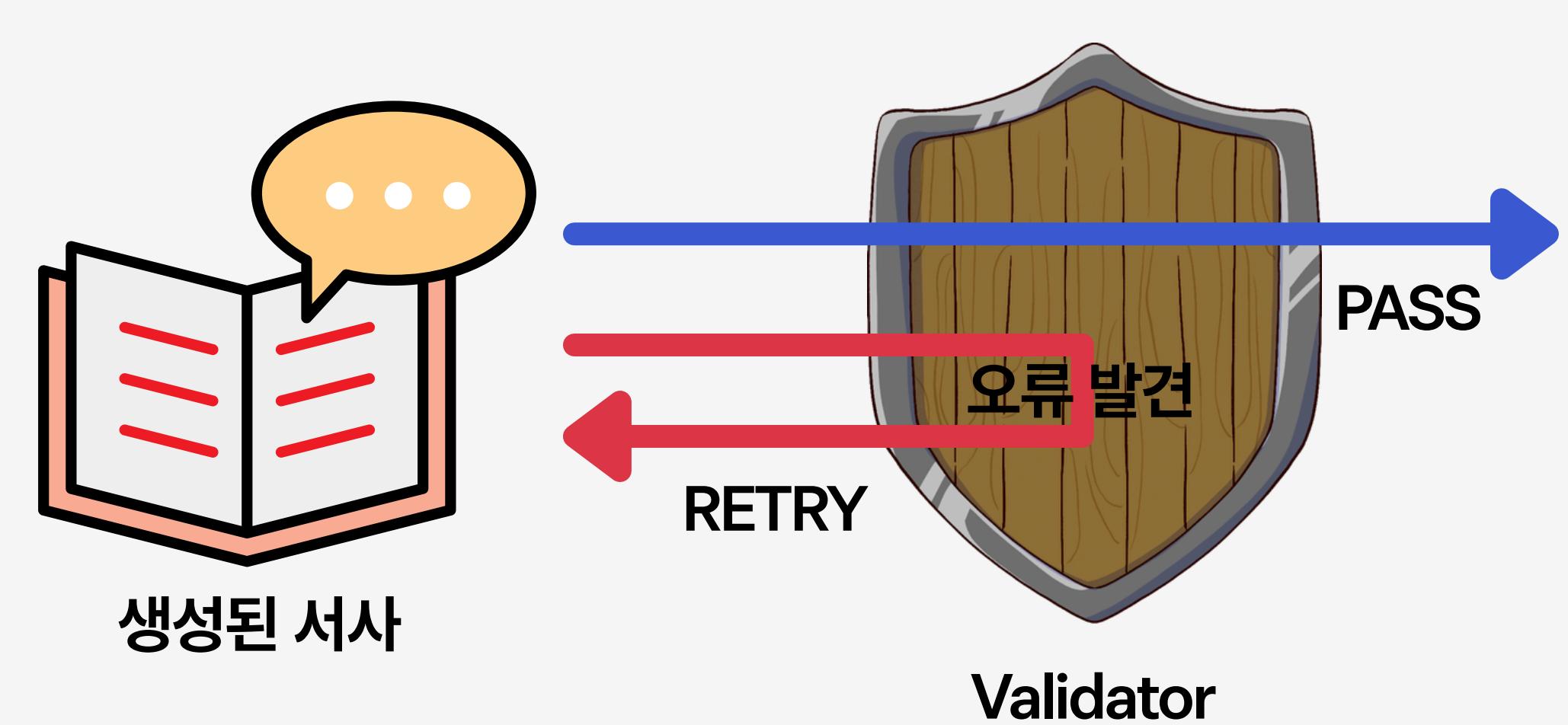
NPC 에이전트
(페르소나 구현)

전지적 묘사와 개성 있는 대사의 완벽한 분리, 체계적인 내러티브 전달

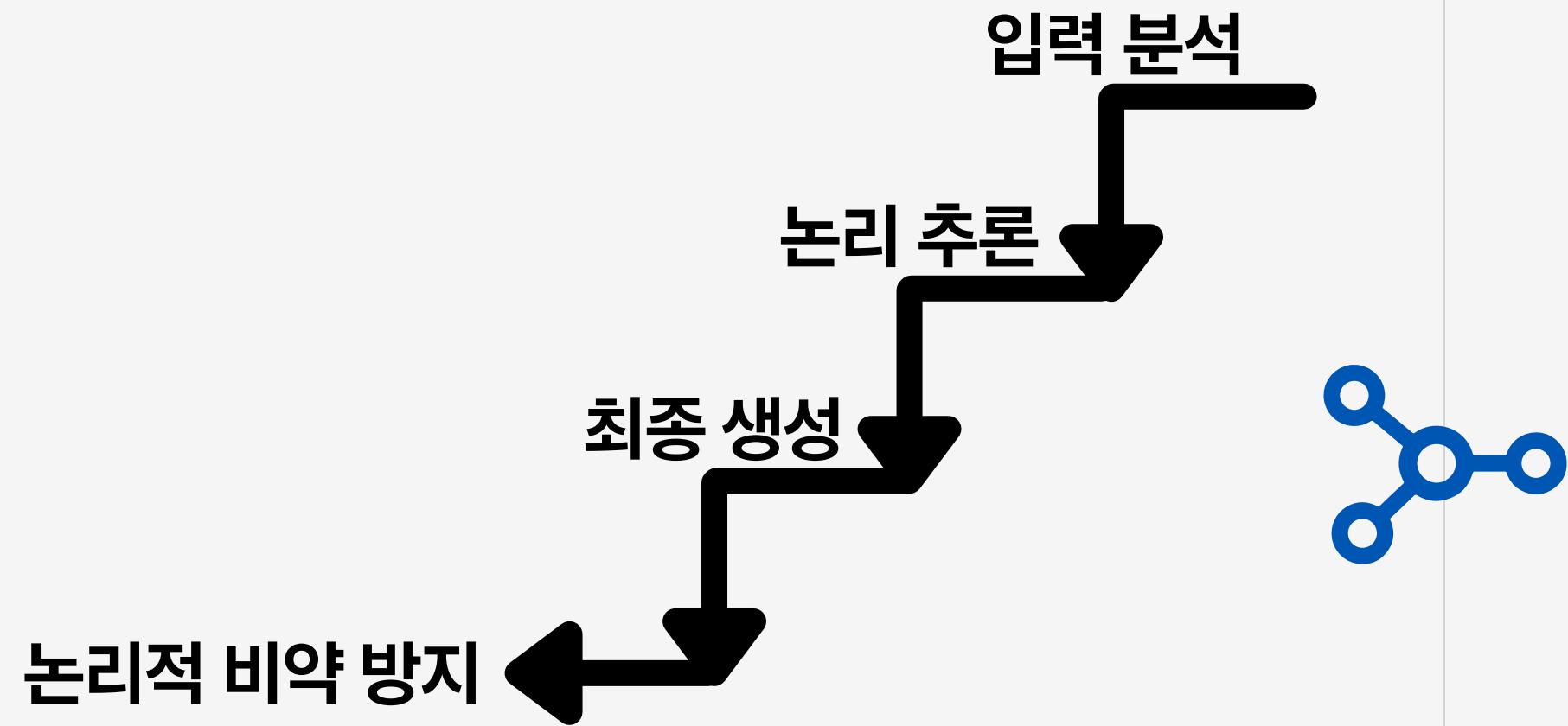
✓ Validator 에이전트



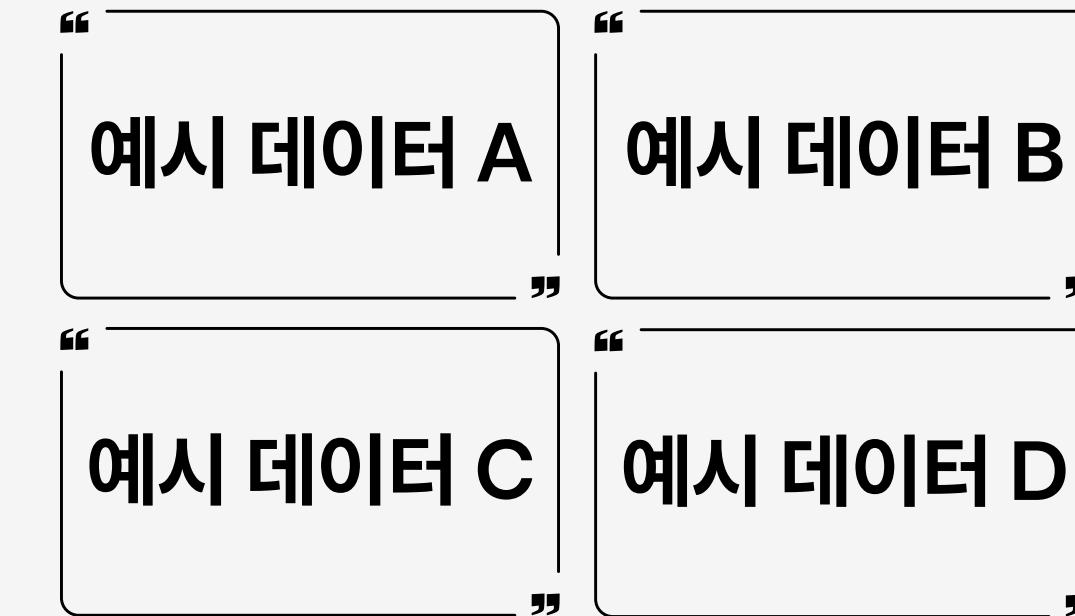
설정 오류 및
서사 붕괴
원천 차단 엔진



✓ 프롬프트 엔지니어링

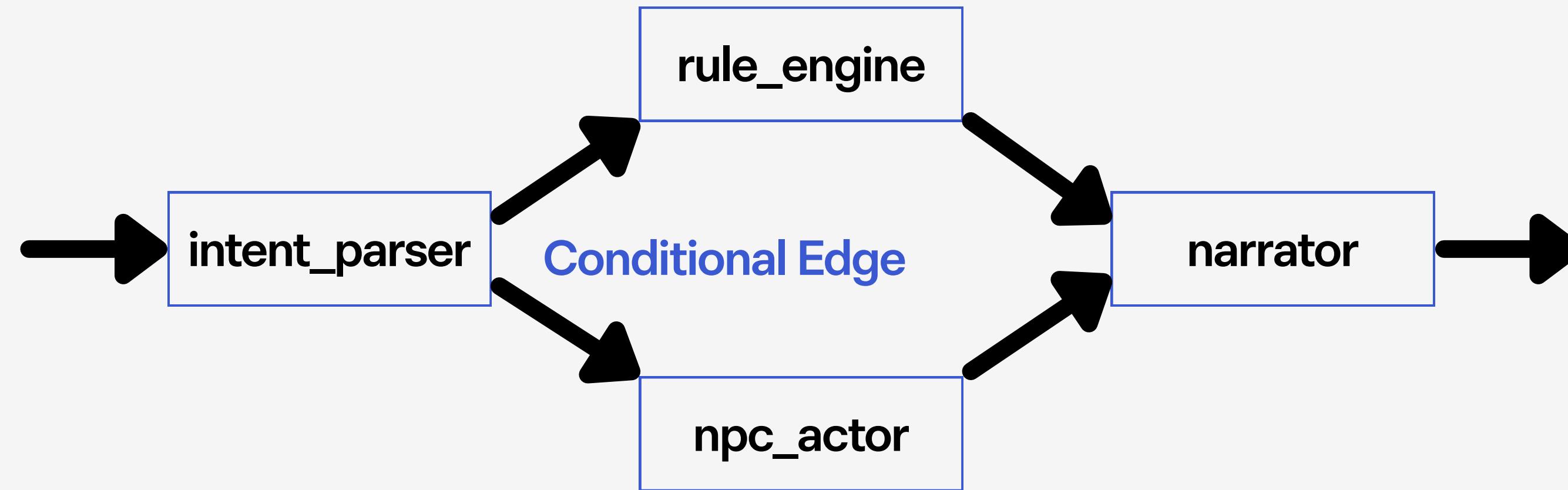


Chain of Thought -
사고의 연쇄



Few-shot Learning -
예시 기반 학습

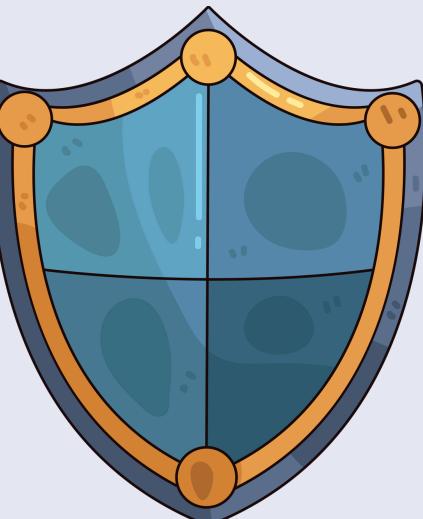
✓ 에이전트 간 협업 프로세스



코드 기반의 엄격한 라우팅으로 구현한 에이전트 간 유기적 협업 파이프라인

✓ 환각 억제 전략

01



Schema Validation

AI 출력을 규격화된 객체로 강제 변환하여
비정형 데이터 오류 원천 차단

02



WorldState Grounding

엔진에 정의된 절대적 수치 데이터를
기반으로 AI의 서사적 모순 실시간 검증

03



Loop Verification

LangGraph의 순환 구조를 활용해
오류 발견 시 즉시 재수행 및 보정 로직 가동

✓ 품질 평가 지표

정합성 및 페르소나 모니터링: 생성된 서사의 논리적 정합성과 캐릭터 고유의 페르소나 유지력을 측정하기 위해 지속적으로 모니터링했음을 명시해라.

데이터 기반 프롬프트 튜닝: 모니터링에서 수집된 실제 응답 데이터를 분석해서 프롬프트의 세부 파라미터를 미세 조정(Tuning)했다고 써라.

피드백 루프 구축: 오류 발생 지점을 역추적해서 Validator 에이전트의 검증 규칙을 강화하는 선순환 구조를 강조해라.

서비스 품질 고도화: 반복적인 테스트와 튜닝을 통해 시스템의 신뢰도를 확보하고 사용자 만족도를 극대화했다는 점을 박아라.

05

시스템 로직 및 내러티브 엔진

설계를 넘어 실시간으로 작동하는
서사 알고리즘과 판정 시스템

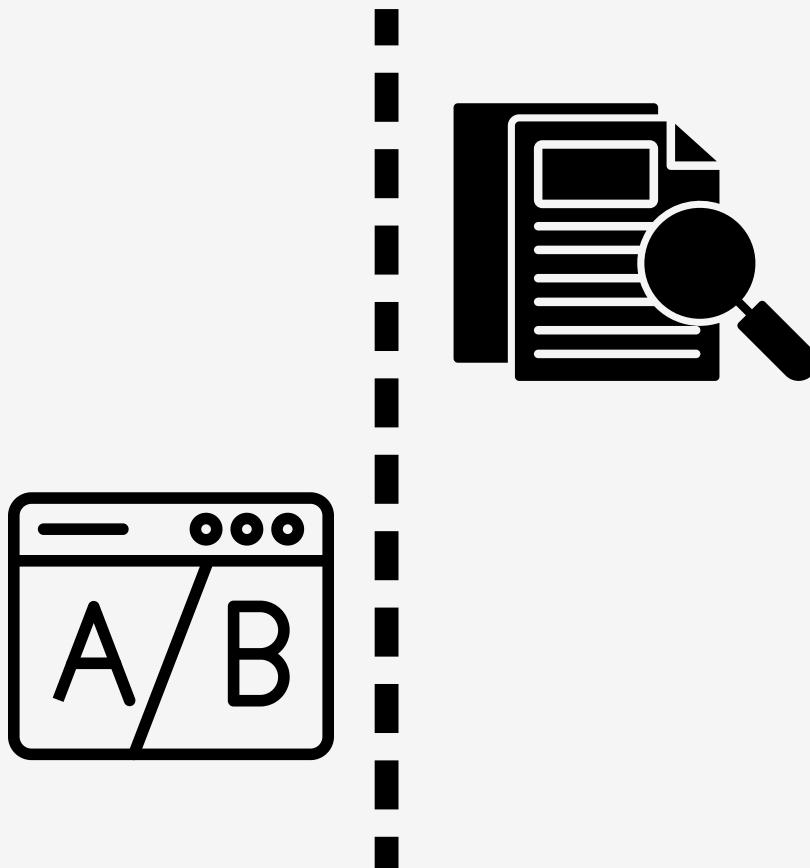


✓ LangGraph 기반 워크플로우

슬라이드 31과 중복?

✓ Intent Parser

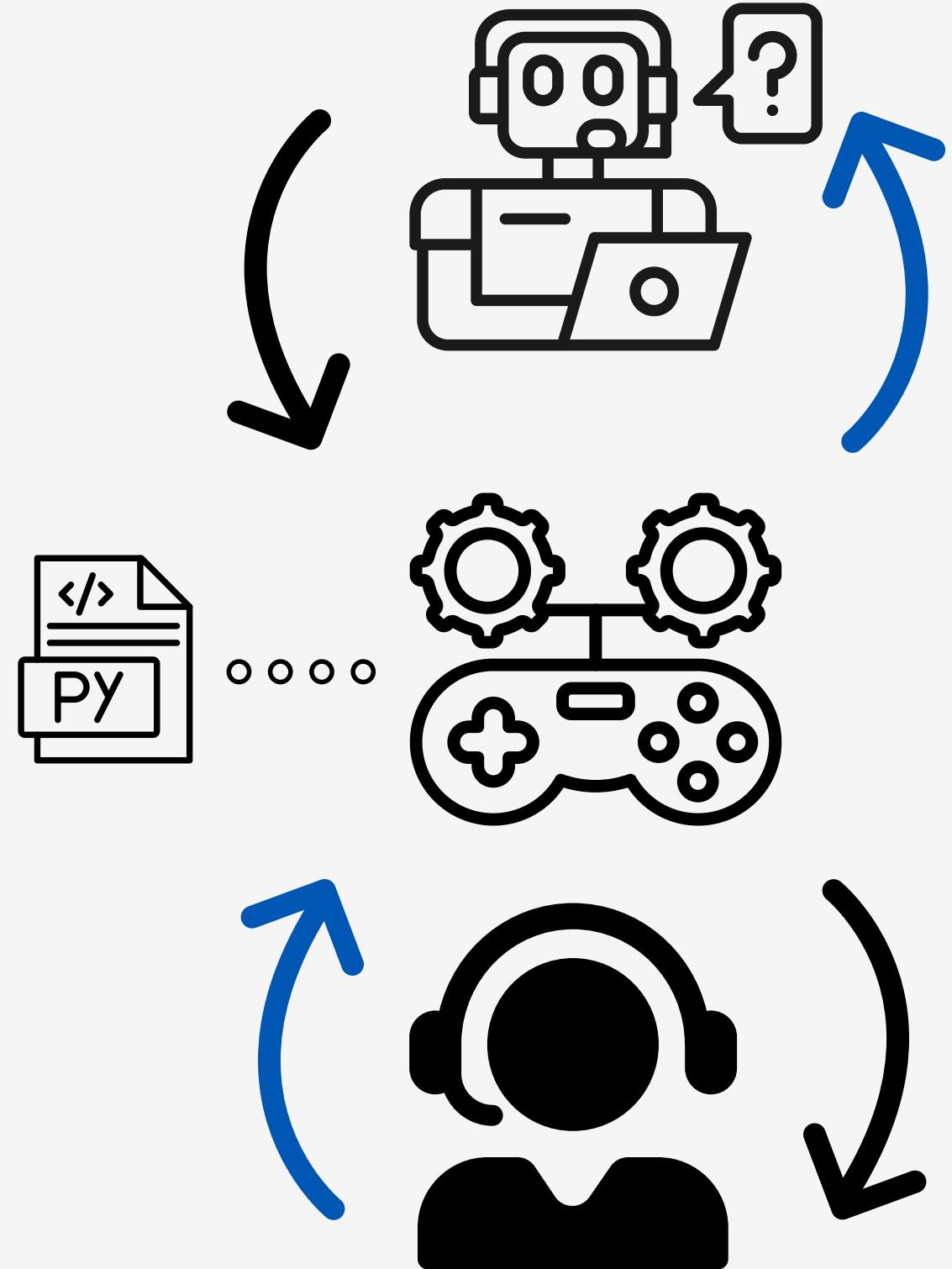
칼을 휘두른다.



모호한 자연어를 정교한 시스템 액션으로
치환하는 **지능형 의도 분석 파이프라인**

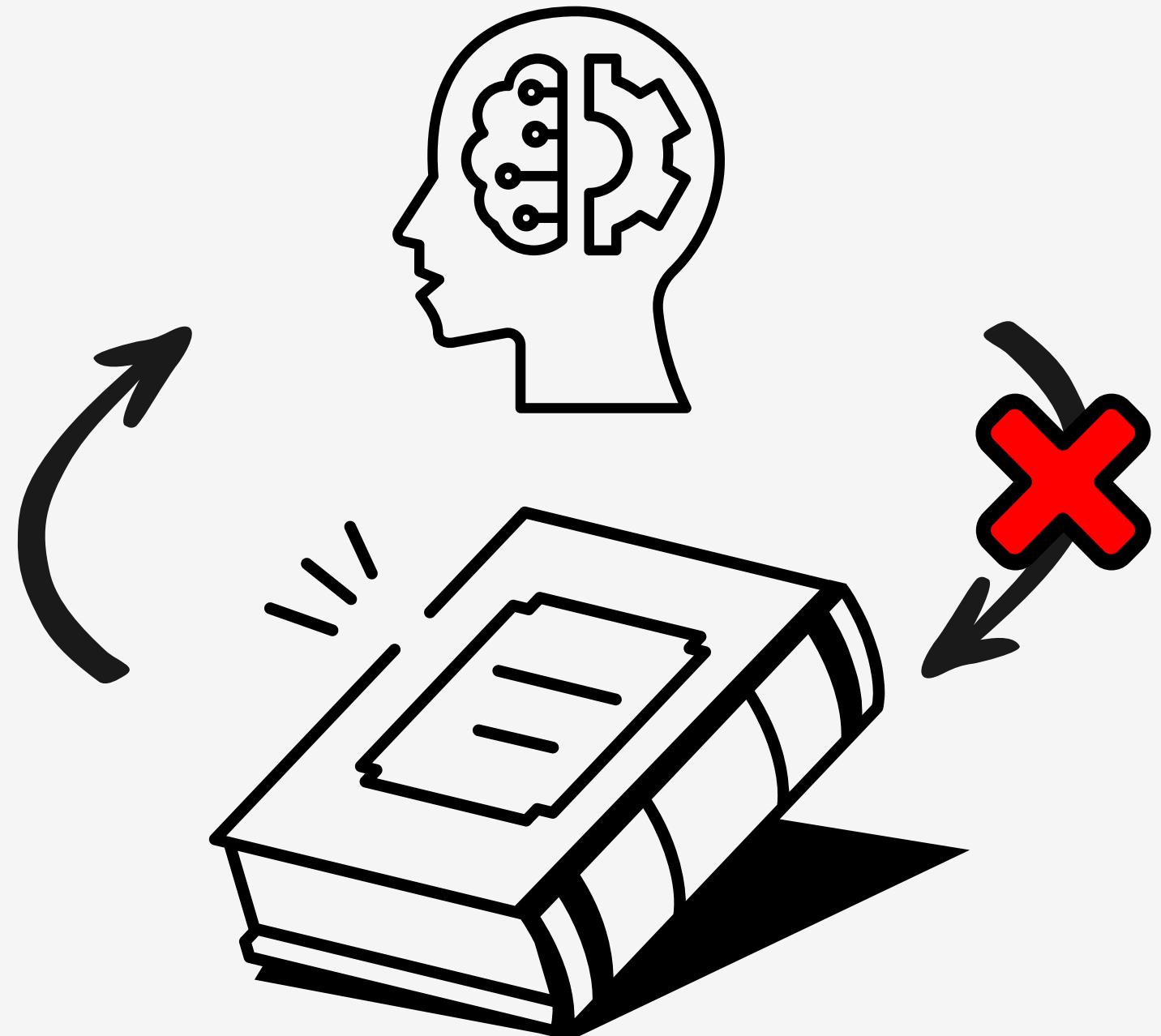
"action": "Attack"

✓ 게임 규칙 엔진



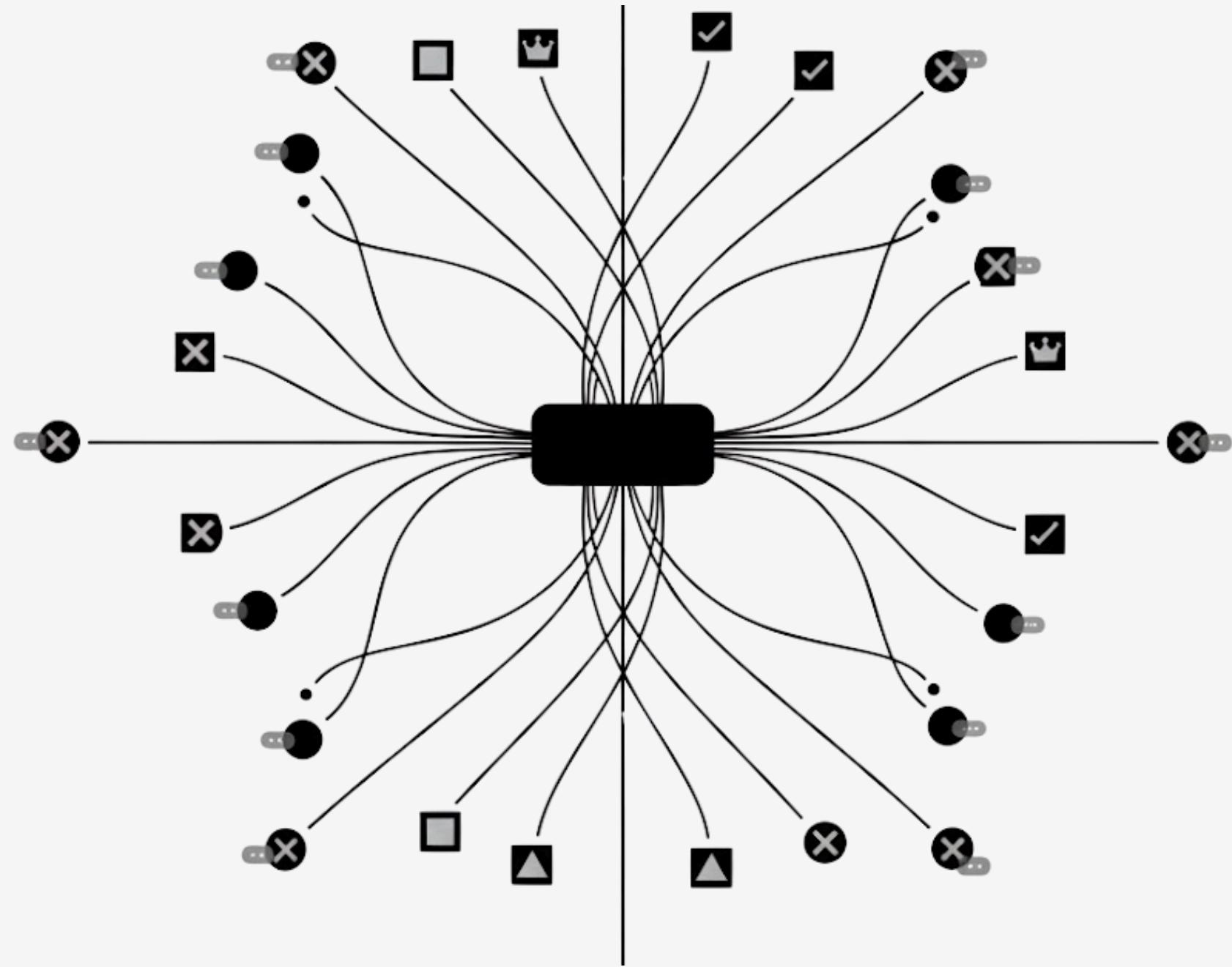
AI의 상상이 아닌 **코드의 논리**로 결정되는,
환각 없는 공정한 게임 마스터링

✓ World State Manager



텍스트는 AI가 묘사하지만,
생사와 규칙은 **코드가 지배**하는 철저한 상태 관리

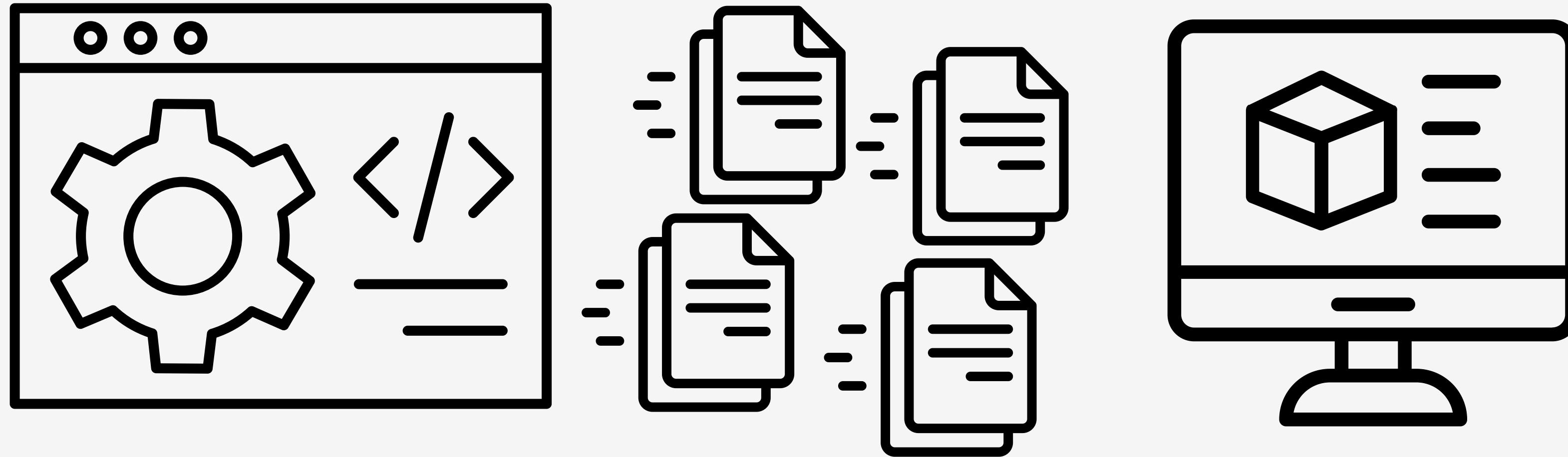
✓ 시나리오 브랜칭



고정된 결말을 거부하고
플레이어의 선택으로 완성되는
실시간 동적 분기 엔진

- 실시간 분기 생성
- 동적 씬 매핑 (Dynamic Mapping)
- 무한한 서사 확장성
- 조건부 엣지(Conditional Edge) 활용

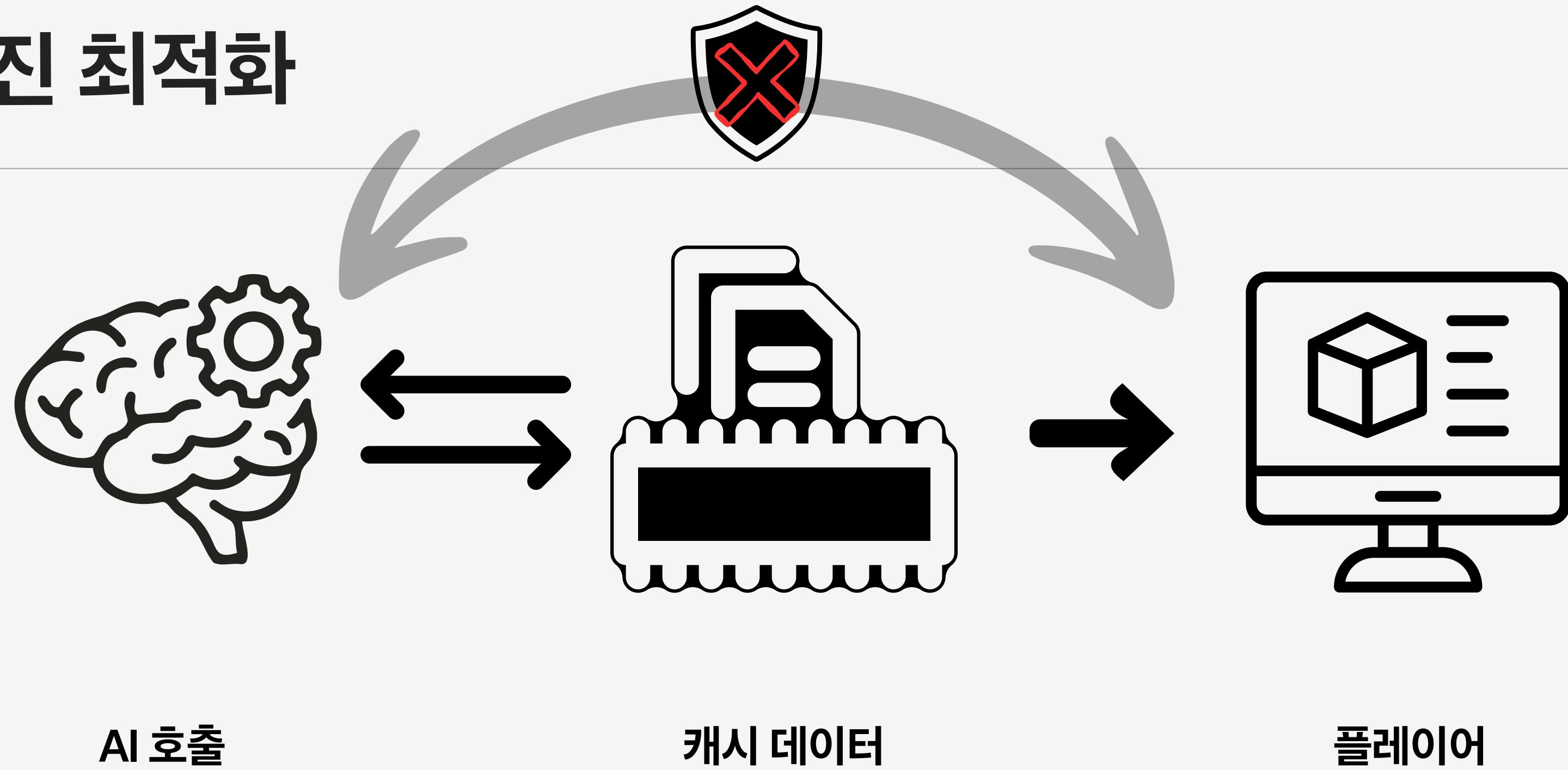
✓ 실시간 스트리밍



routes/game.py

기다림 없는 몰입: 실시간 SSE 스트리밍으로 구현하는 생동감 넘치는 서사 전개

✓ 엔진 최적화



비용은 낮추고 속도는 높이는, 캐싱 기반의 고성능 엔진 최적화 아키텍처

06

SECURITY



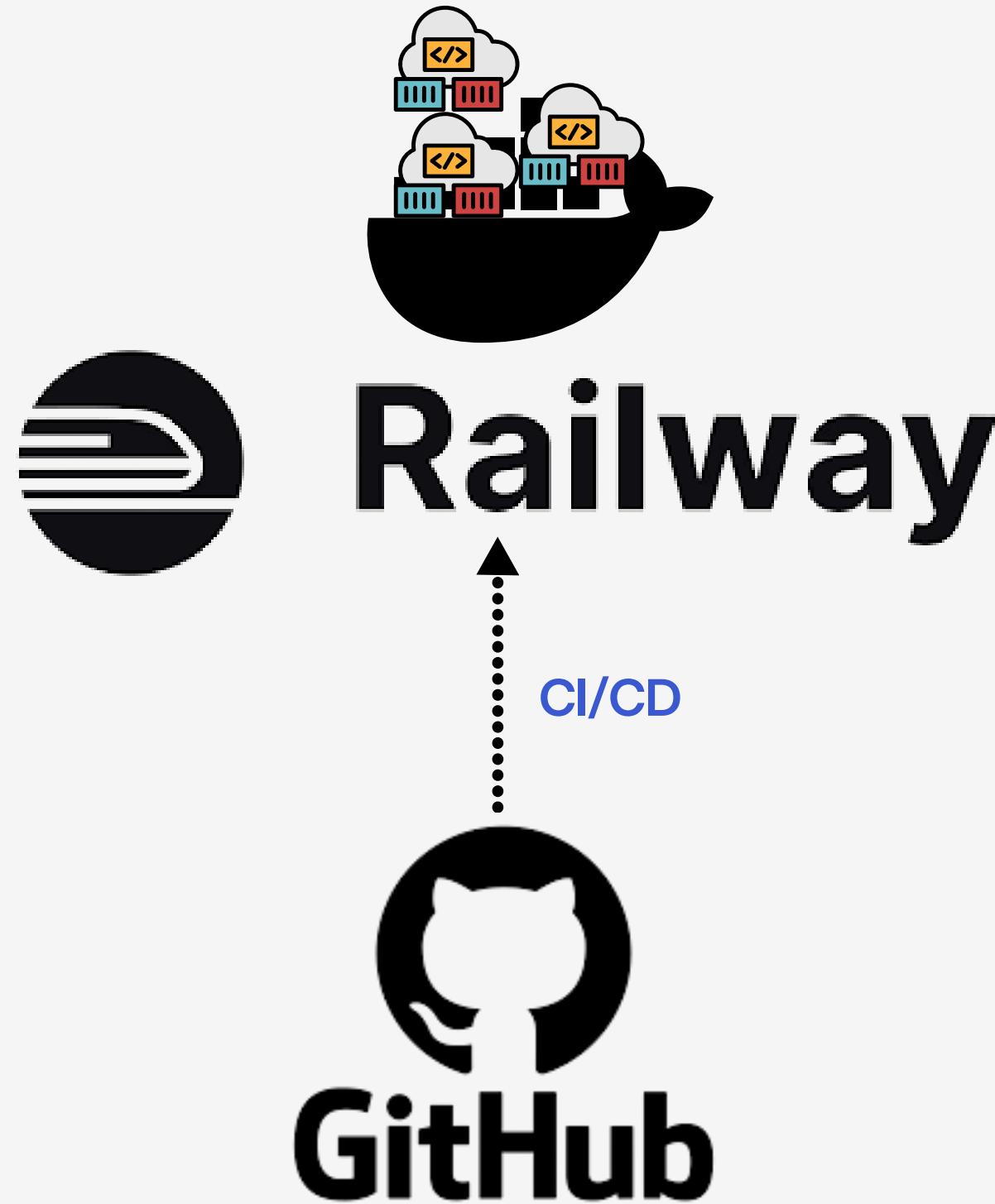
시스템 아키텍처

INFRASRUCTURE
DEPLOYMENT

안정적인 서비스 제공을 위한

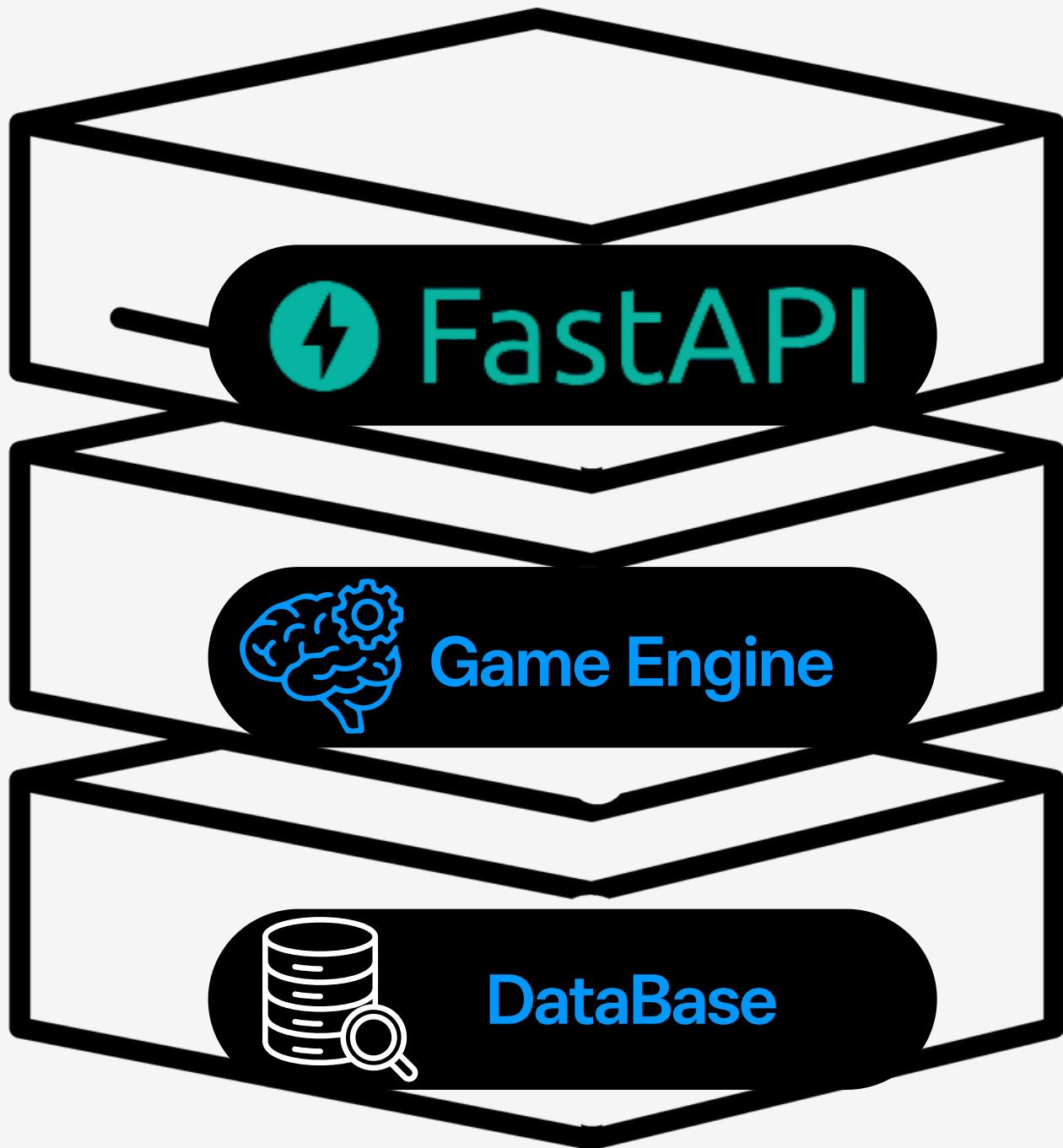
Docker 기반의 클라우드 인프라 구축

✓ 전체 서비스 인프라 구성도



Docker와 Railway로 구현한
안정적이고 효율적인 클라우드 네이티브 인프라

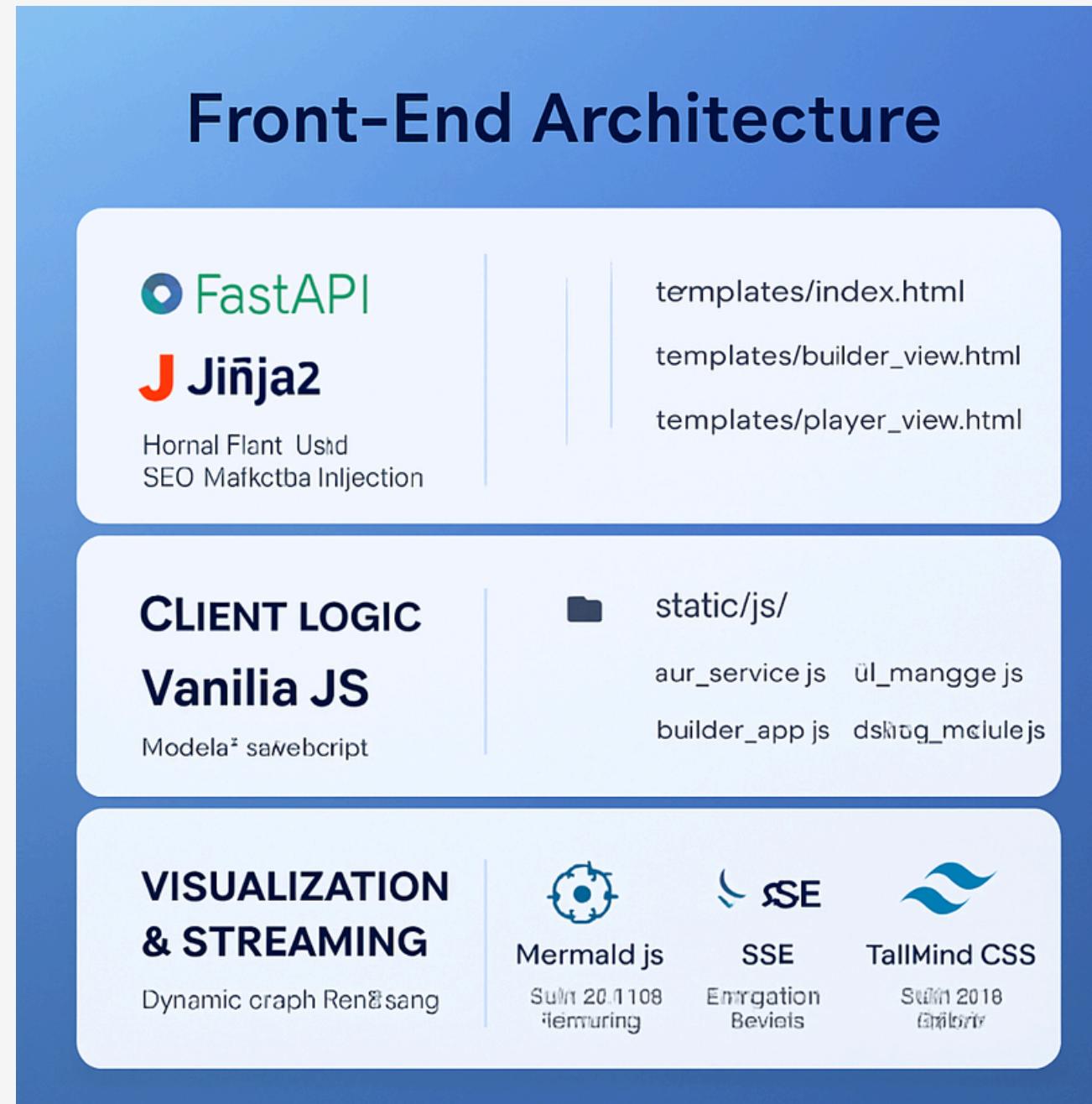
✓ 백엔드 아키텍처



FastAPI와 레이어드 아키텍처로 완성한
고성능·확장형 백엔드 시스템

✓ 프론트엔드 아키텍처

메인타이틀에 대한 세부 설명을 입력해 주세요.



Layer 1: View Rendering (FastAPI + Jinja2)

- 역할: 초기 HTML 로드, SEO 최적화, 메타데이터 주입
- 파일: templates/index.html, builder_view.html, player_view.html

Layer 2: Client Logic (Modular Vanilla JS)

- 역할: DOM 조작, 이벤트 핸들링, 상태 관리 (No Framework Overhead)
- 구조: static/js/
 - Core: api_service.js (통신), ui_manager.js (화면 제어)
 - Modules: builder_app.js (제작), debug_module.js (디버깅)

Layer 3: Visualization & Streaming (External Libs)

- Mermaid.js: 실시간 서사 구조 시각화 (동적 그래프 렌더링)
- SSE (Server-Sent Events): LLM 토큰 스트리밍(파이밍 효과 구현)
- Tailwind CSS: 유틸리티 퍼스트 스타일링 (빠른 렌더링)

✓ 프론트엔드 아키텍처

메인타이틀에 대한 세부 설명을 입력해 주세요.

파운데이션 & 라우팅

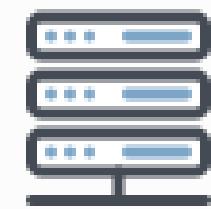
Rendering & Strategy



FastAPI



Jinja2

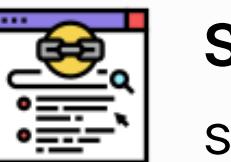


Routing:
Server-Side
Routing

데이터 & 상태 매니지먼트(관리)



Data Fetching:
Native Fetch API



State Strategy:
server- Authoritative



Real-time:
SSE(Server-Sent)

애플리케이션 구조



Module System:
ES6 Modules
(Vanilla JS)



Separation:
Templates / Styles /
Scripts

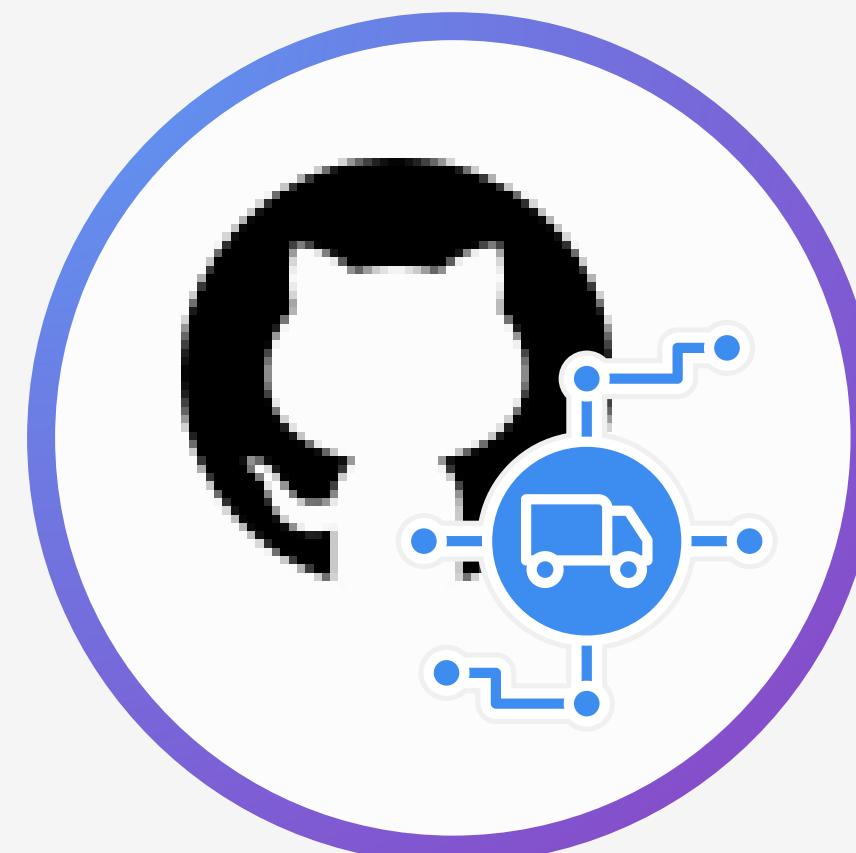


Visualization:
Mermaid.js

✓ 인프라 배포

메인타이틀에 대한 세부 설명을 입력해 주세요.

자동 CI/CD 파이프 라인



GitHub 코드 푸시 시
자동 빌드 및 배포

코드형 인프라 관리(IaC)



railway.json 및 Procfile로
배포 환경 코드로 정의

Railway Paas 플랫폼



서버관리 없는 완전 관리형
서비스 및 안정적 운영

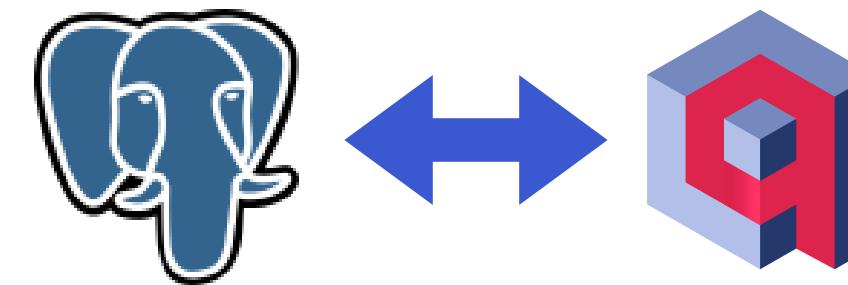
✓ 데이터베이스 구성

메타데이터에 대한 세부 설명을 입력해 주세요.

메타 데이터와 벡터지식의 하이브리드 구조 및 스키마 자동관리

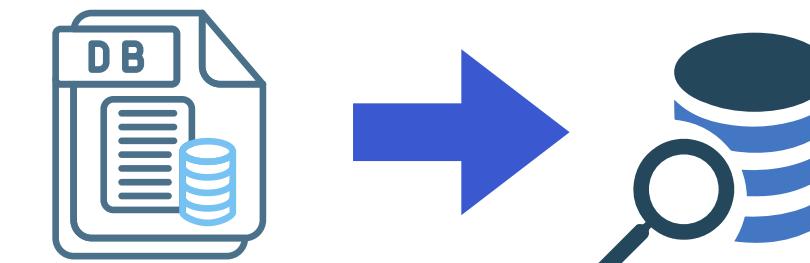
저장소	데이터 유형	주요 데이터 예시
 PostgreSQL (관계형 DB)	정형 메타 데이터	유저/캐릭터 정보, 게임진행 상태, 설정월드
 Qdrant (벡터 DB)	비정형/벡터 지식	룰북 임베딩, NPC 대화 기록, 월드정보 청크

하이브리드 저장소 전략



데이터 특성에 따른 최적화된 분산 저장

스키마 자동 관리 (init_db.py)



앱 실행 시 스키마 자동 생성 및 갱신

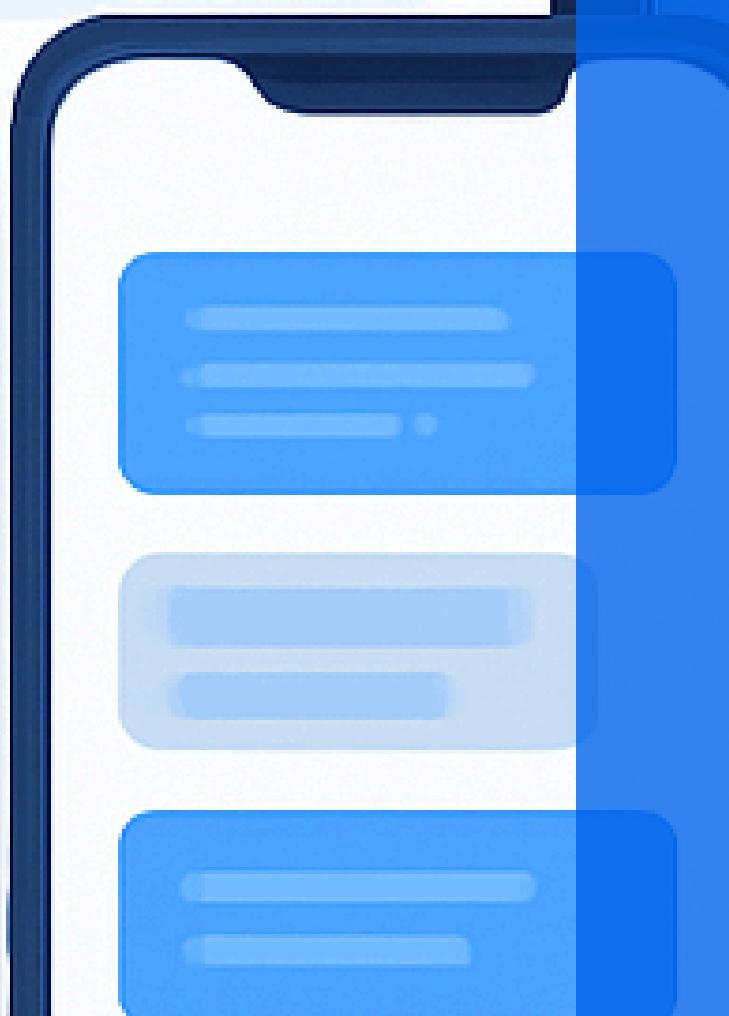
DDoS 시치

✓ 보안 및 인증

메인타이틀에 대한 세부 설명을 입력해 주세요.

고객군	주요특징	핵심니즈	전략적 접근법
A 타깃군	20-30대 디지털 네이티브	빠른 정보 습득, 소셜 미디어 활동	SNS 중심의 빠르고 트렌디한 정보 제공
B 타깃군	30-40대 직장인	신뢰도, 효율적 의사결정	전문 콘텐츠, 객관적 정보 제공
C 타깃군	40-50대 안정추구형	브랜드 충성도, 가격 대비 가치	충성 고객 프로그램 확대, 맞춤형 혜택
D 타깃군	50-60대 중장년층	간편함, 명확한 정보 전달	접근성 높고 사용 편의성 강조한 서비스
E 타깃군	10대 후반~20대 초반 청년층	최신 트렌드, 창의적 경험 추구	참여형 이벤트, 체험 중심 콘텐츠 제공

Dashboard



✓ 실시간 서비스 시연

LIVE DEMO

[프로젝트 예술: 실시간 서비스 시연]

08



회고 및 향후 계획

시련을 통해 완성된 '여울',
그리고 우리가 그려갈 내러티브의 미래

✓ 트러블슈팅 1

Card 01

브랜드 소개용 슬라이드

이 카드는 브랜드나 조직을 처음 소개할 때 사용하는 레이아웃입니다. 로고, 브랜드 철학, 설립 연도, 비전 등을 간략하게 정리해 한눈에 브랜드의 정체성을 전달할 수 있습니다.

Card 02

핵심 서비스 안내 카드

기업이 제공하는 주요 서비스나 제품을 간략히 소개하는 데 적합한 카드입니다. 한 장의 카드에 하나의 서비스만 집중해서 담고, 아이콘이나 일러스트를 활용하여 직관성을 높입니다.

Card 03

정보형 콘텐츠 하이라이트

정보 콘텐츠나 교육 자료의 핵심만 뽑아서 요약하는 카드 구성입니다. [콘텐츠 마케팅의 3가지 전략]처럼 주제를 간단하게 정리하고, 핵심 문장 또는 키워드를 강조하여 구성합니다.

✓ 트러블슈팅 2

Card 01

브랜드 소개용 슬라이드

이 카드는 브랜드나 조직을 처음 소개할 때 사용하는 레이아웃입니다. 로고, 브랜드 철학, 설립 연도, 비전 등을 간략하게 정리해 한눈에 브랜드의 정체성을 전달할 수 있습니다.

Card 02

핵심 서비스 안내 카드

기업이 제공하는 주요 서비스나 제품을 간략히 소개하는 데 적합한 카드입니다. 한 장의 카드에 하나의 서비스만 집중해서 담고, 아이콘이나 일러스트를 활용하여 직관성을 높입니다.

Card 03

정보형 콘텐츠 하이라이트

정보 콘텐츠나 교육 자료의 핵심만 뽑아서 요약하는 카드 구성입니다. [콘텐츠 마케팅의 3가지 전략]처럼 주제를 간단하게 정리하고, 핵심 문장 또는 키워드를 강조하여 구성합니다.

✓ 프로젝트 성과 요약



이미지 강조형 정보 배열

하나의 슬라이드에 여러 이미지를 나열하여 다양한 정보를 한눈에 보여줄 수 있는 레이아웃입니다.

병렬 구조 정보 나열

세 개의 박스가 좌우로 균등하게 배치되어 있어, 항목별 비교나 연속된 흐름 설명에 적합한 레이아웃입니다.

설명 중심 캡션 구성

사진 아래에 제목과 함께 짧은 설명을 추가해, 시각적 요소와 텍스트 정보를 조화롭게 전달합니다.

✓ 한계점 분석

리스트형 콘텐츠 정리 레이아웃은 정보의 구조를 명확히 하고, 각 항목을 구분하여 보기 쉽게 정렬하는 데 매우 적합한 방식입니다. 이 레이아웃은 복잡하거나 많은 내용을 간결하게 요약할 때 유용하며, 발표자와 청중 모두에게 정보 전달 효율을 높여줍니다.

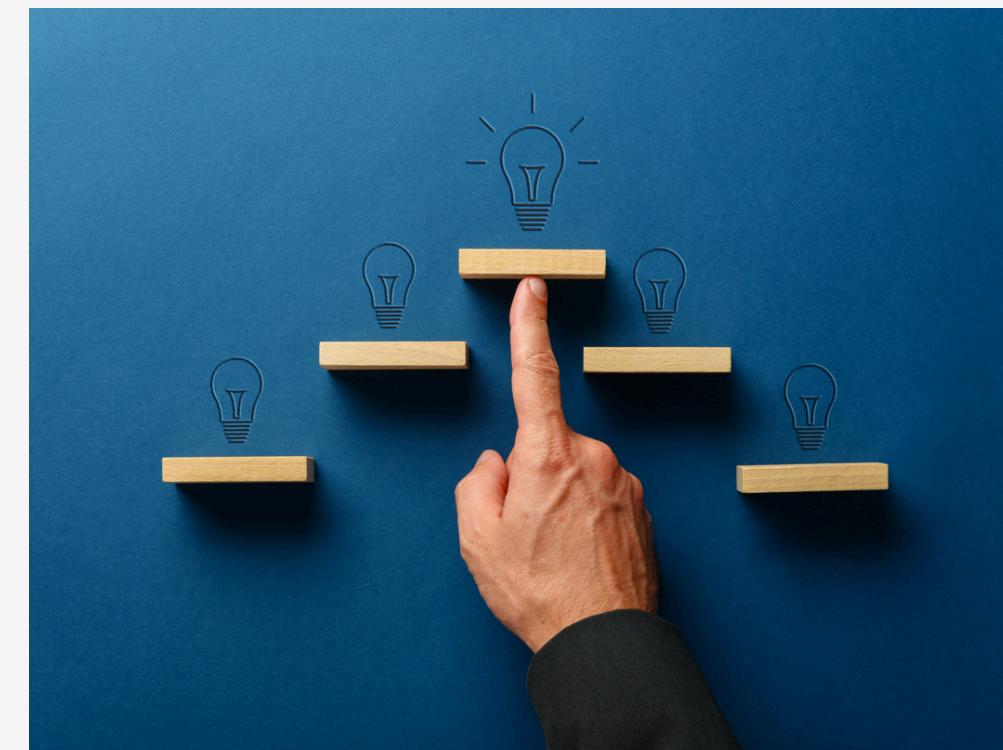
- ✓ 각 항목은 **하나의 주제**에만 집중하여 정보를 쪼개어 전달합니다.
- ✓ 체크아이콘, 숫자, 화살표 등으로 정보의 흐름을 **시각적으로 표현**합니다.
- ✓ 동일한 간격, 균형 잡힌 정렬로 **보기 좋은 구조**를 만듭니다.

✓ 서비스 고도화 1



이미지 강조형 정보 배열

하나의 슬라이드에 여러 이미지를 나열하여 다양한 정보를 한눈에 보여줄 수 있는 레이아웃입니다.



병렬 구조 정보 나열

세 개의 박스가 좌우로 균등하게 배치되어 있어, 항목별 비교나 연속된 흐름 설명에 적합한 레이아웃입니다.



설명 중심 캡션 구성

사진 아래에 제목과 함께 짧은 설명을 추가해, 시각적 요소와 텍스트 정보를 조화롭게 전달합니다.

✓ 서비스 고도화 1



이미지 강조형 정보 배열

하나의 슬라이드에 여러 이미지를 나열하여 다양한 정보를 한눈에 보여줄 수 있는 레이아웃입니다.



병렬 구조 정보 나열

세 개의 박스가 좌우로 균등하게 배치되어 있어, 항목별 비교나 연속된 흐름 설명에 적합한 레이아웃입니다.



설명 중심 캡션 구성

사진 아래에 제목과 함께 짧은 설명을 추가해, 시각적 요소와 텍스트 정보를 조화롭게 전달합니다.

출처

Q&A

본 프레젠테이션에 대해 문의할 점이 있으시면 언제든 문의주세요.

소속	마케팅 1팀
성명	이수진

기사합니다

본 프레젠테이션에 대해 문의할 점이 있으시면 언제든 문의주세요.

소속	마케팅 1팀
성명	이수진