

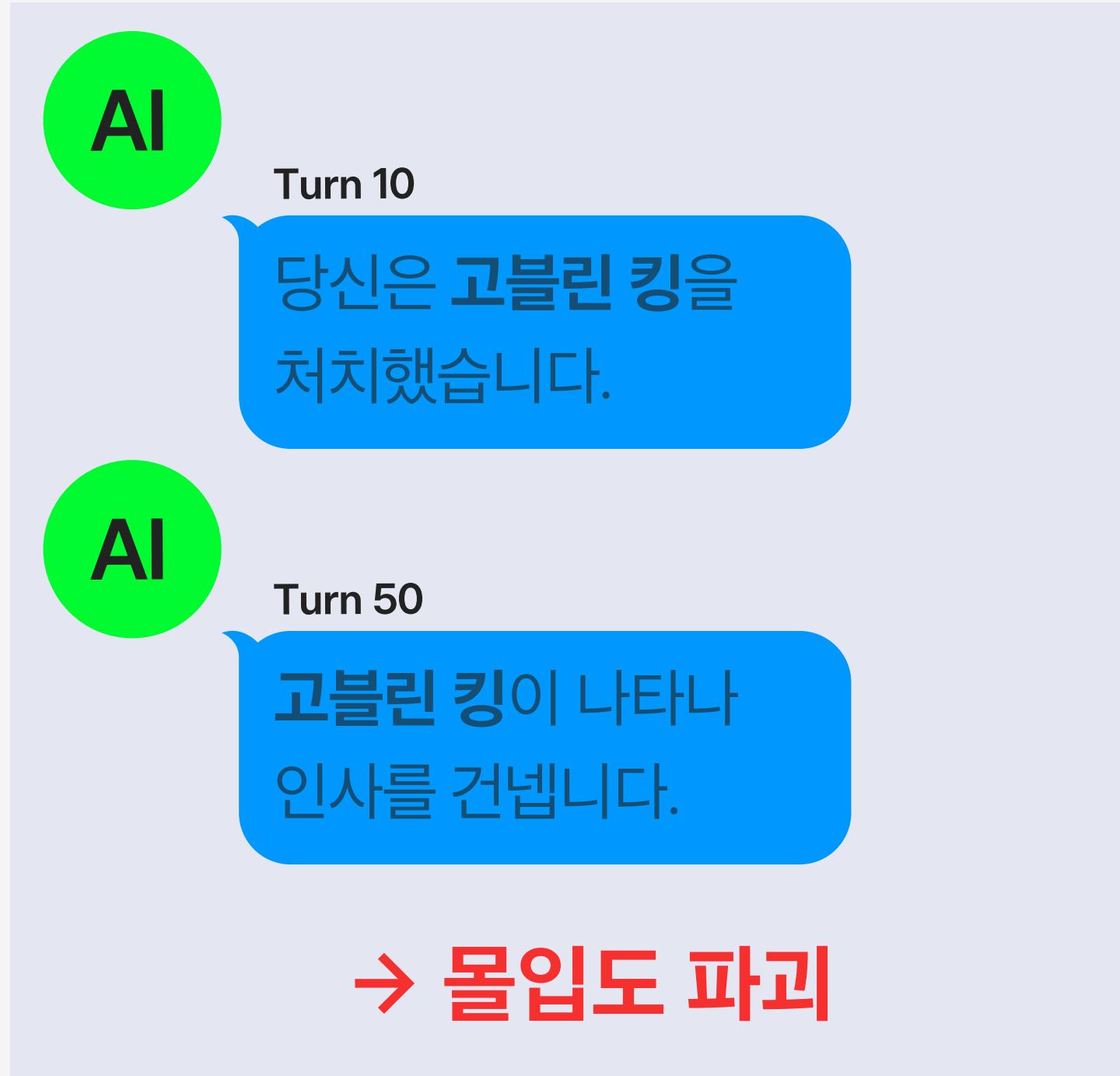
Project YEOUL

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

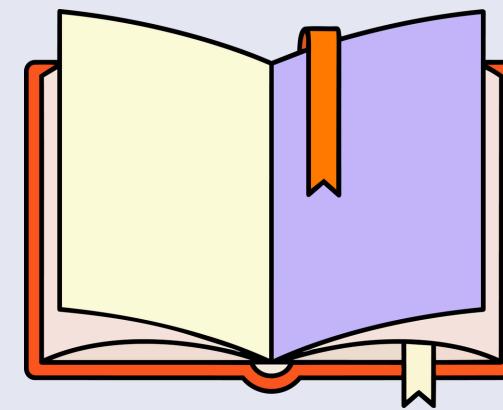
- 대화가 길어질수록 초반 설정을 망각함
- # 맥락 상실 # 장기 기억력 부재 # 일관성 결여

Point 2

환각 현상

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

✓ 왜 '여울'인가? - 멀티 에이전트 조율 엔진

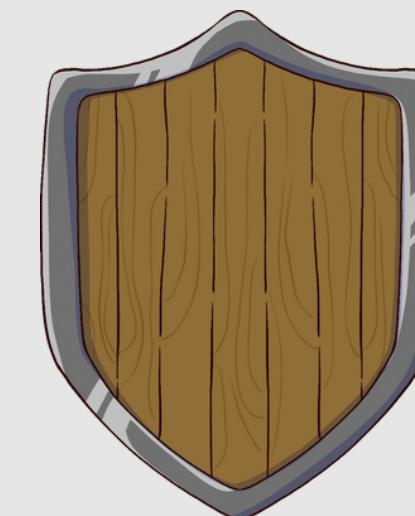


Narrator Agent

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



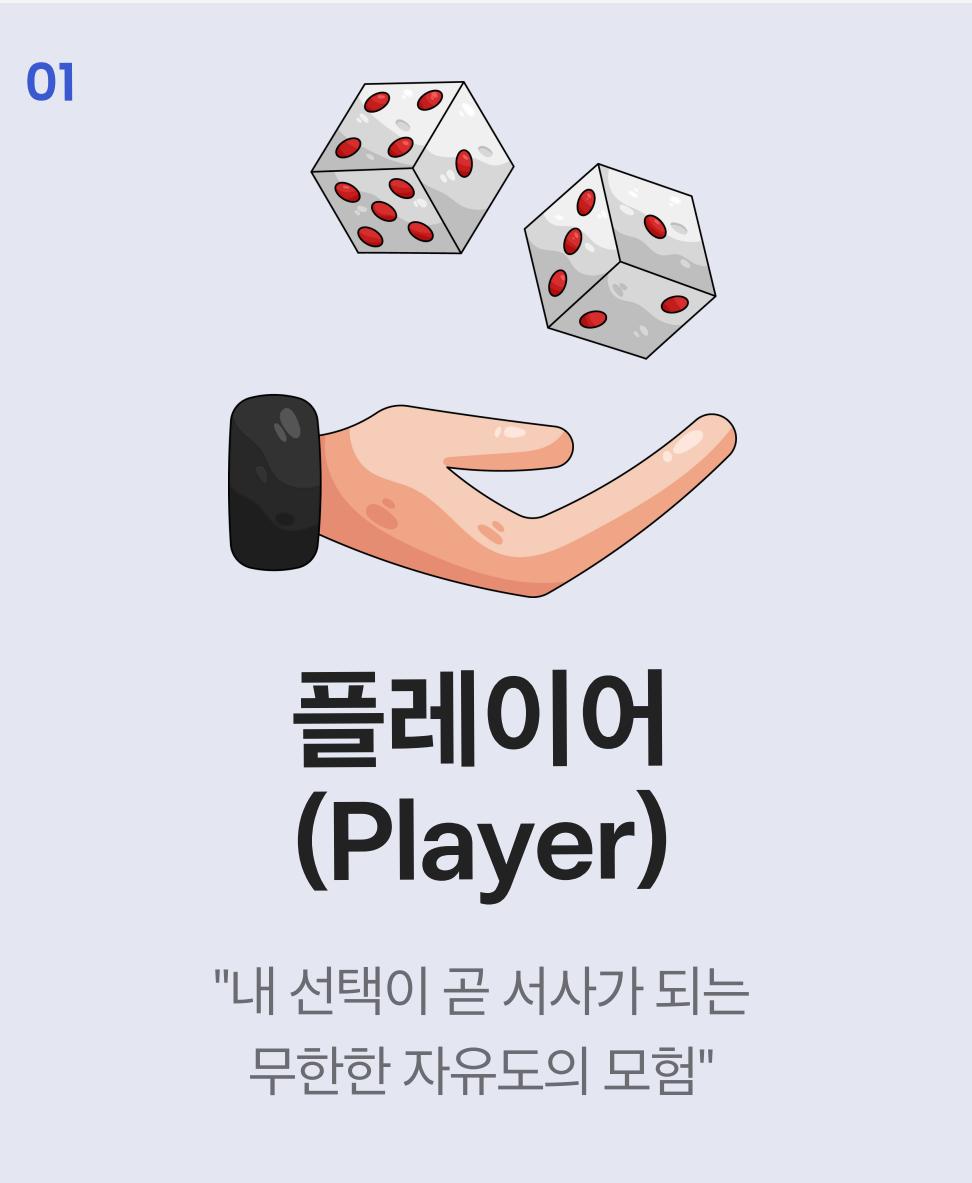
“



"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

2025.12.20



2025.12.16

핵심 엔진 및 빌더 개발

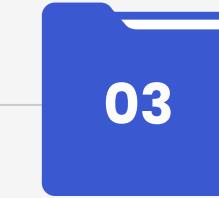
LangGraph 기반 워크플로우 설계
노드 GUI 시나리오 빌더 개발



서비스 고도화 및 최적화

SSE 실시간 스트리밍 구축
Mermaid.js 연동 서사 시각화 고도화

현재



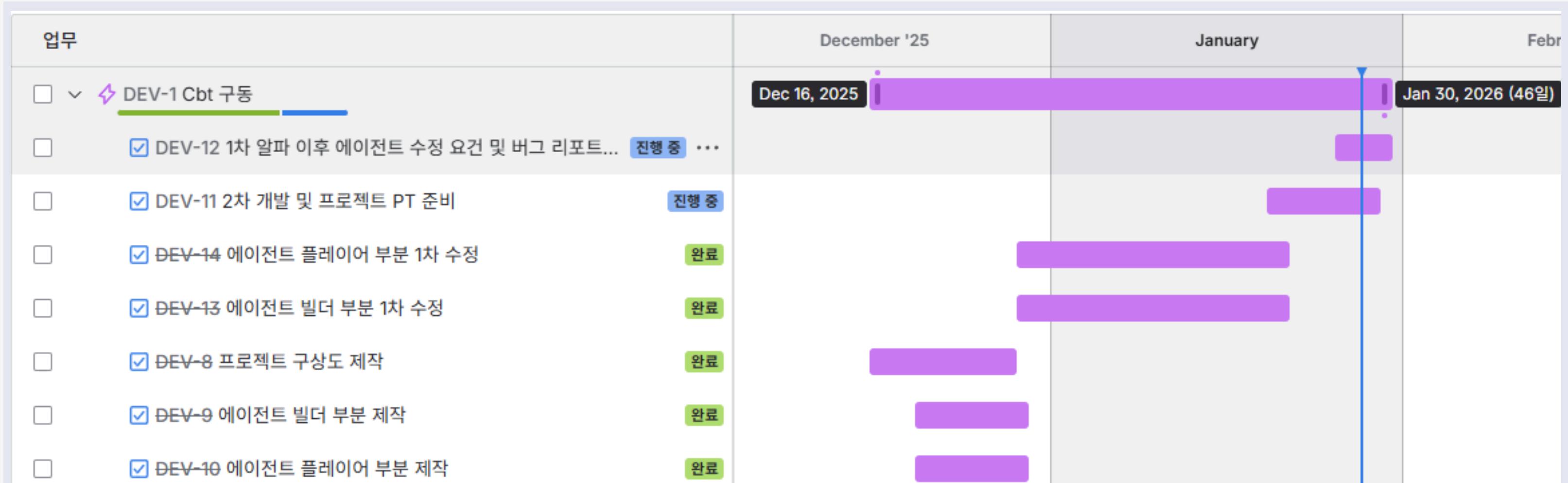
2026.01.20

배포 및 실서비스 운영

Railway 클라우드 실서비스 배포
CI/CD 파이프라인 자동화 구축



✓ 상세 일정 및 마일스톤



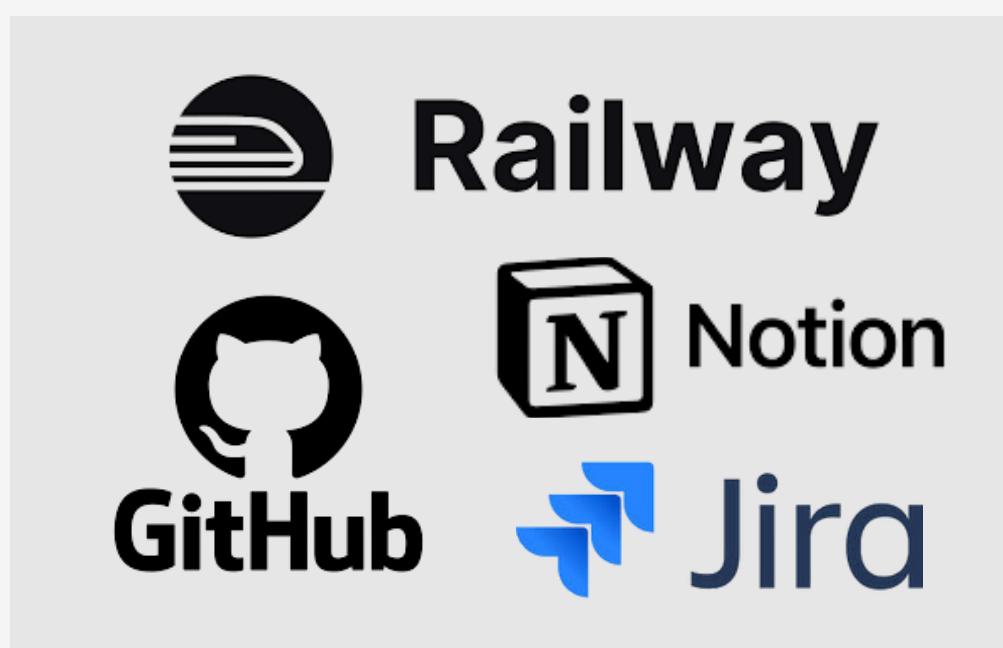
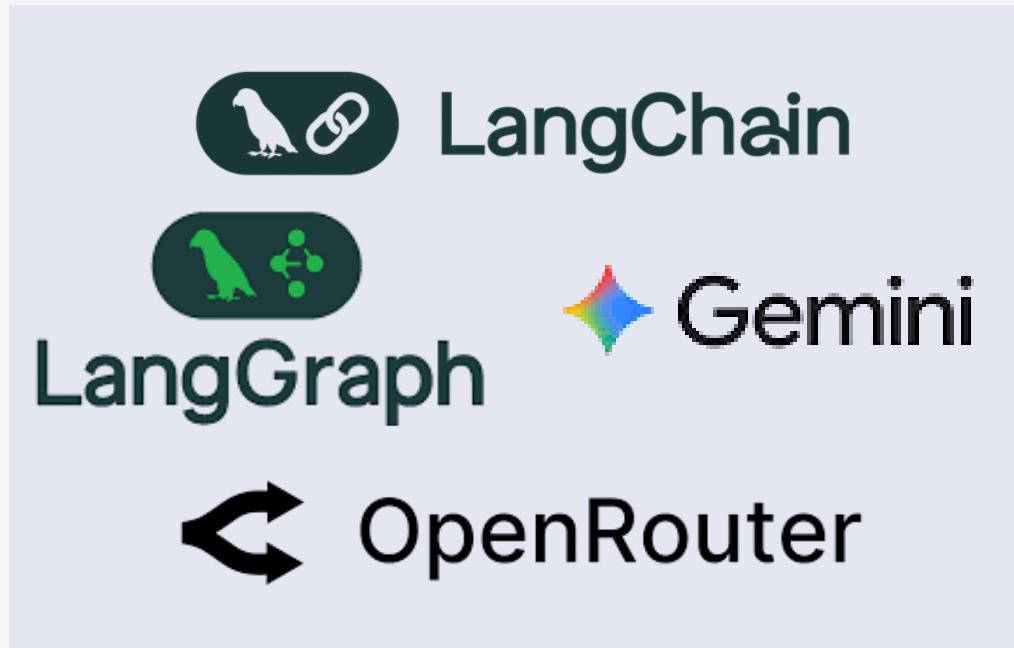
1. 기획/설계: 아키텍처 및 시나리오 규격 설계 완료

3. 엔진 고도화: LangGraph 워크플로우 및 시각화 고도화 완료

2. 에이전트 개발: 빌더/플레이어 핵심 모듈 1차 개발 완료

4. 운영/안정화: CBT 구동 및 버그 리포트 기반 최종 튜닝 (진행 중)

✓ 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

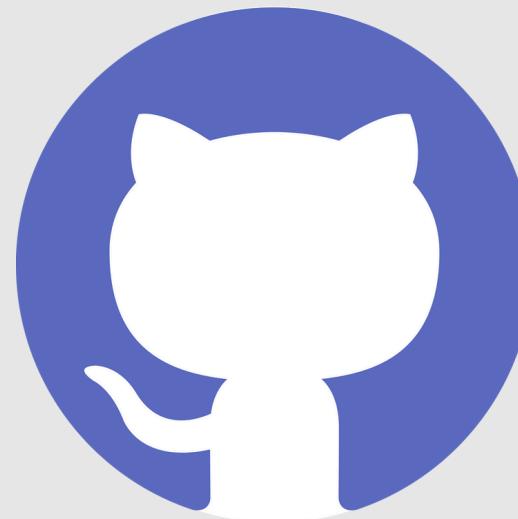
01



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

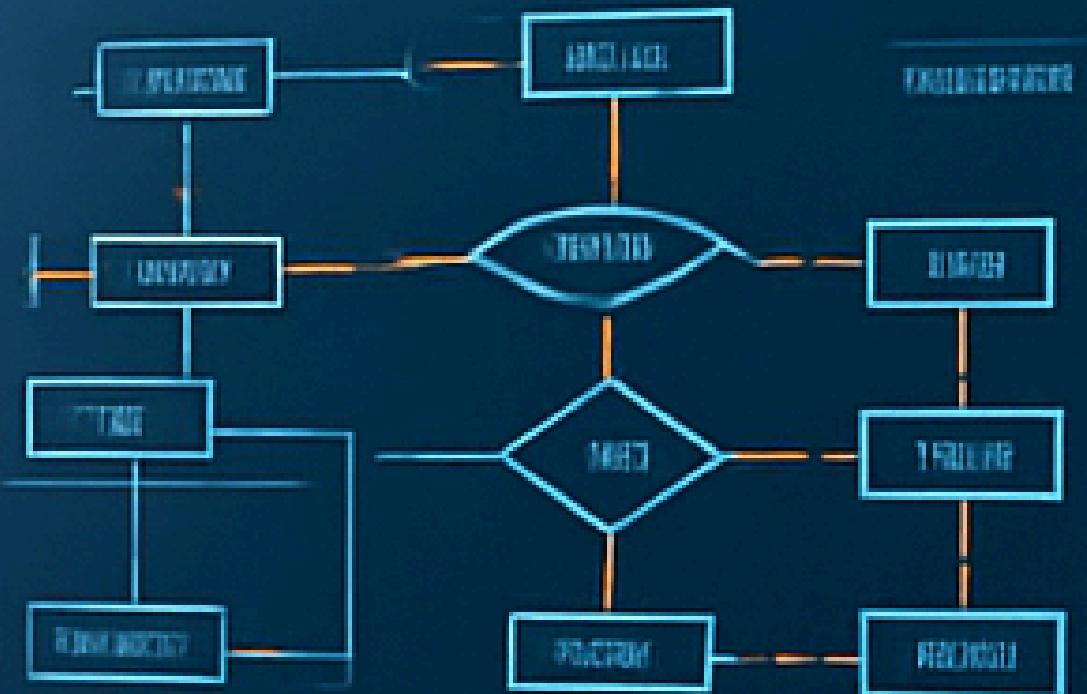
03



Notion
(Knowledge Sync)

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

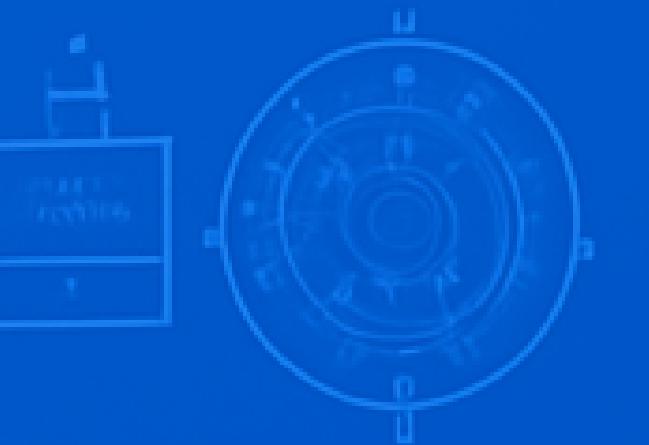
DATA SCHEMA



SCENARIO GRAPH



CURRENT STATUS



03

데이터 및 상태 설계

비정형 서사를 완벽하게 제어하는
정형 데이터 관리 전략

INTELLIGENT DATA BUILDER

✓ 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."

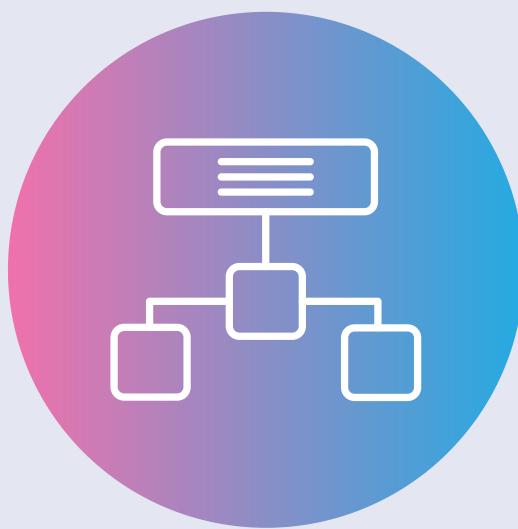
```
{  
  "scenario": {  
    "title": "데이터 말소 작전: 랩 03",  
    "npcs": [{"name": "G-72", "isEnemy": false, "...": "..."}],  
             // AI가 아닌 '시스템'이 적대 여부 결정  
    "scenes": [{"scene_id": "Scene-1",  
               "trigger": "[관리자 키카드] 소지 시에만 이동 가능",  
               // 서사 분기를 결정하는 핵심 데이터  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



Strict Schema Strategy: 비정형 서사를 정형 데이터로 제어

✓ Scenario Schema 설계

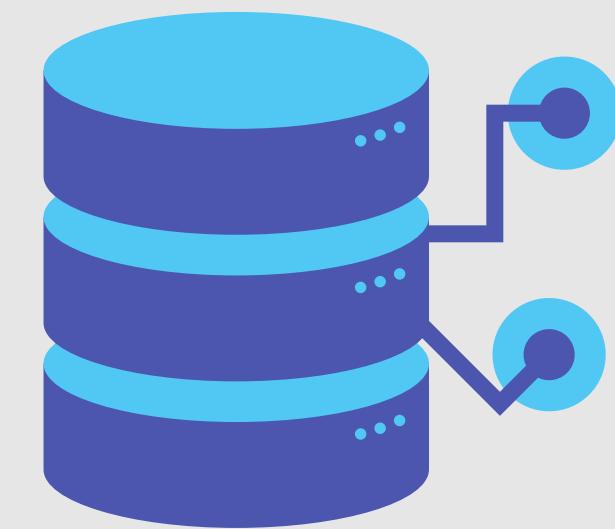
01



Data Hierarchy

Root 모델 중심의
계층적 시나리오 구조화

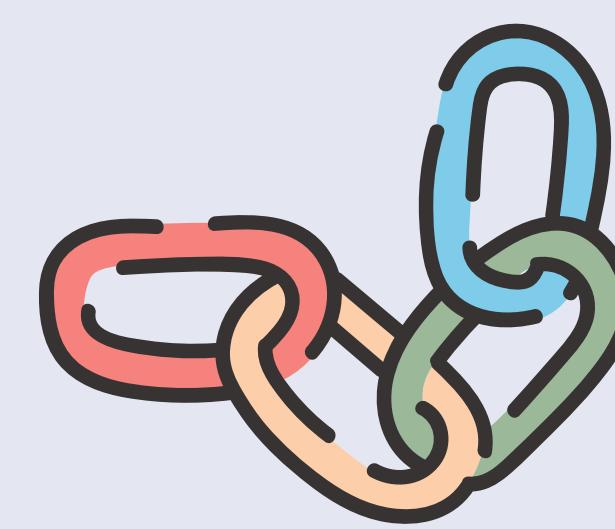
02



World State

HP, 골드 등 실시간 변동 수치
데이터 관리

03

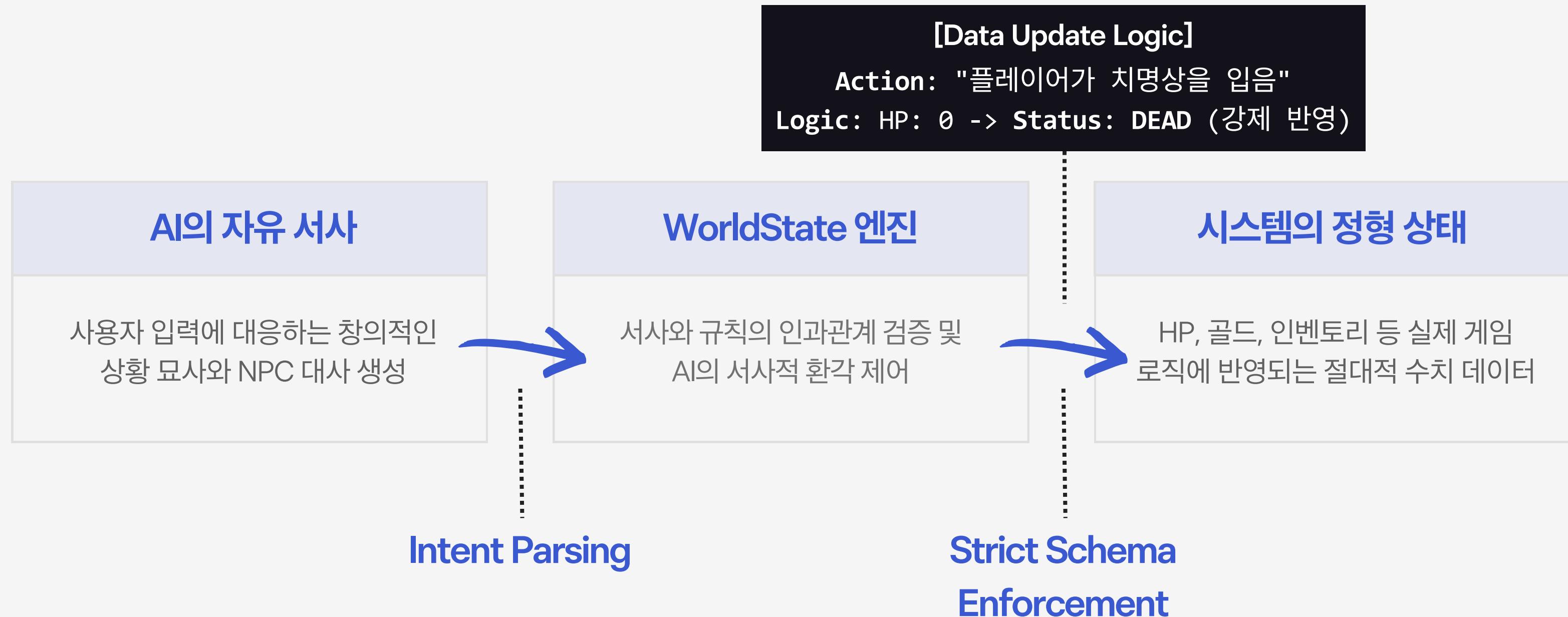


Logic Chain

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

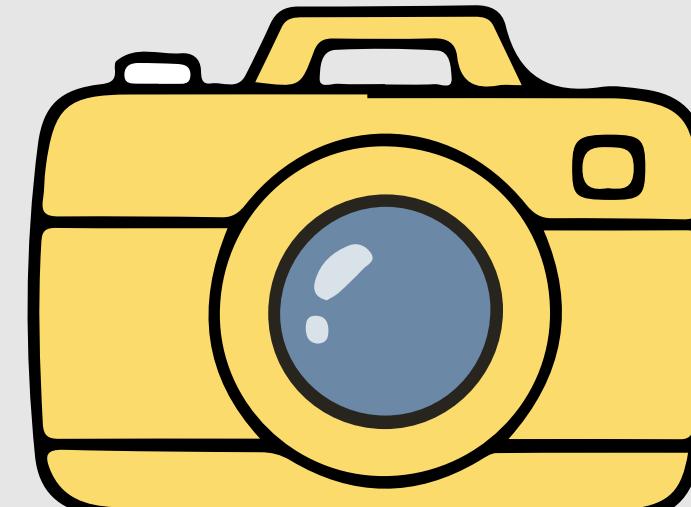
01



Hybrid Storage

PostgreSQL JSONB를 활용한 정형/비정형 데이터 통합 관리

02



State Snapshot

모든 행동 직후 실시간 스냅샷 기록으로 0% 데이터 유실 실현

03

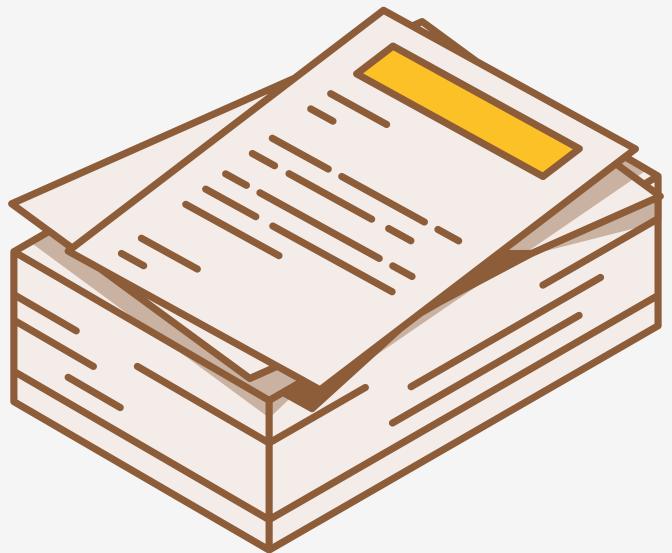


Integrity Check

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



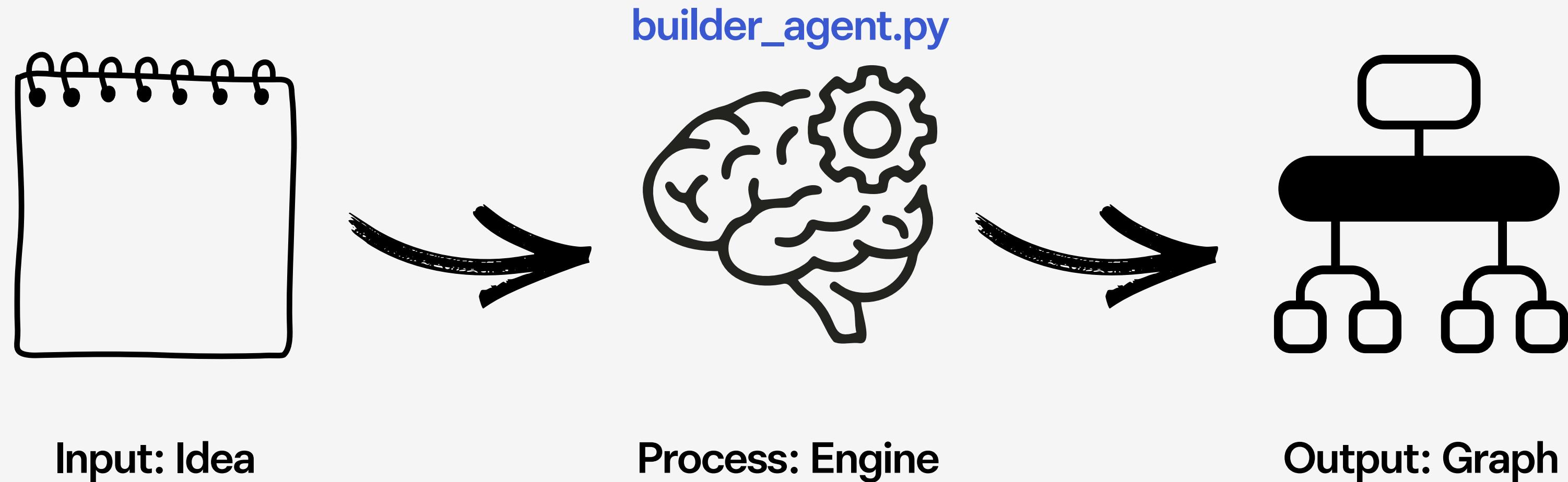
Qdrant (Vector DB)



최적화된 컨텍스트

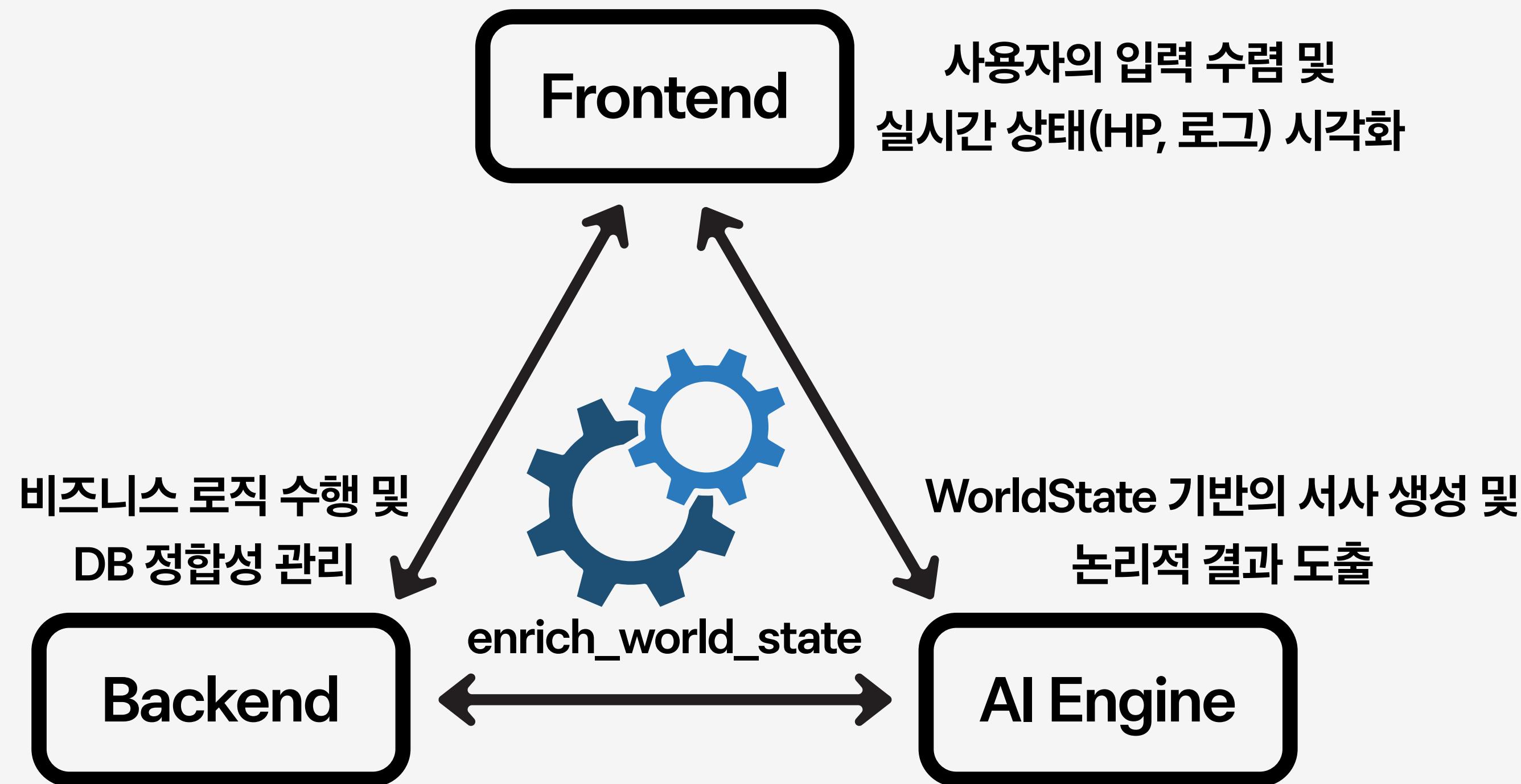
불필요한 노이즈는 제거하고, LLM에는 오직 핵심 지식만을 전달

✓ 데이터 변환 로직



비정형 아이디어를 정교한 서사 그래프로 치환하는 지능형 데이터 빌더

✓ 데이터 동기화 프로세스

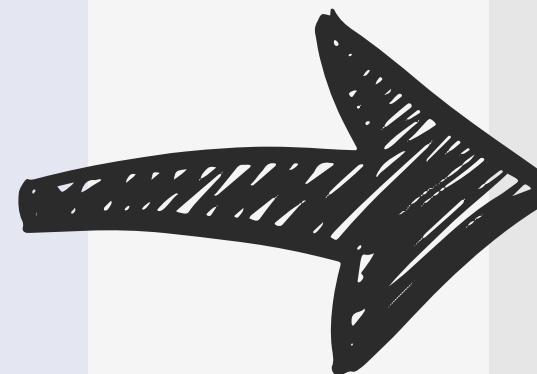


✓ 서사 컨텍스트 주입

01

시스템 데이터 (JSON)

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



Context Injection

02

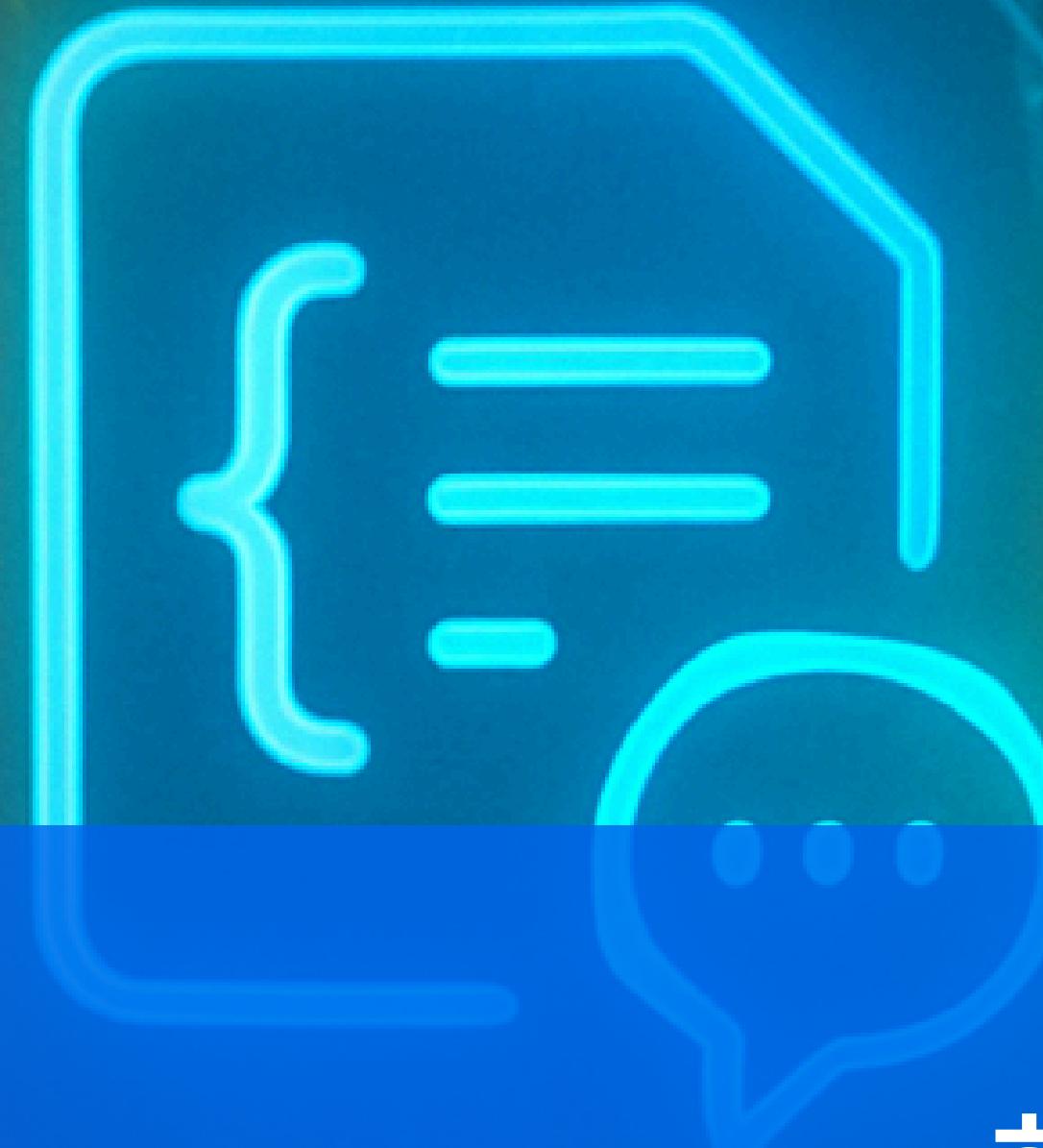
AI 마스터의 서사

"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

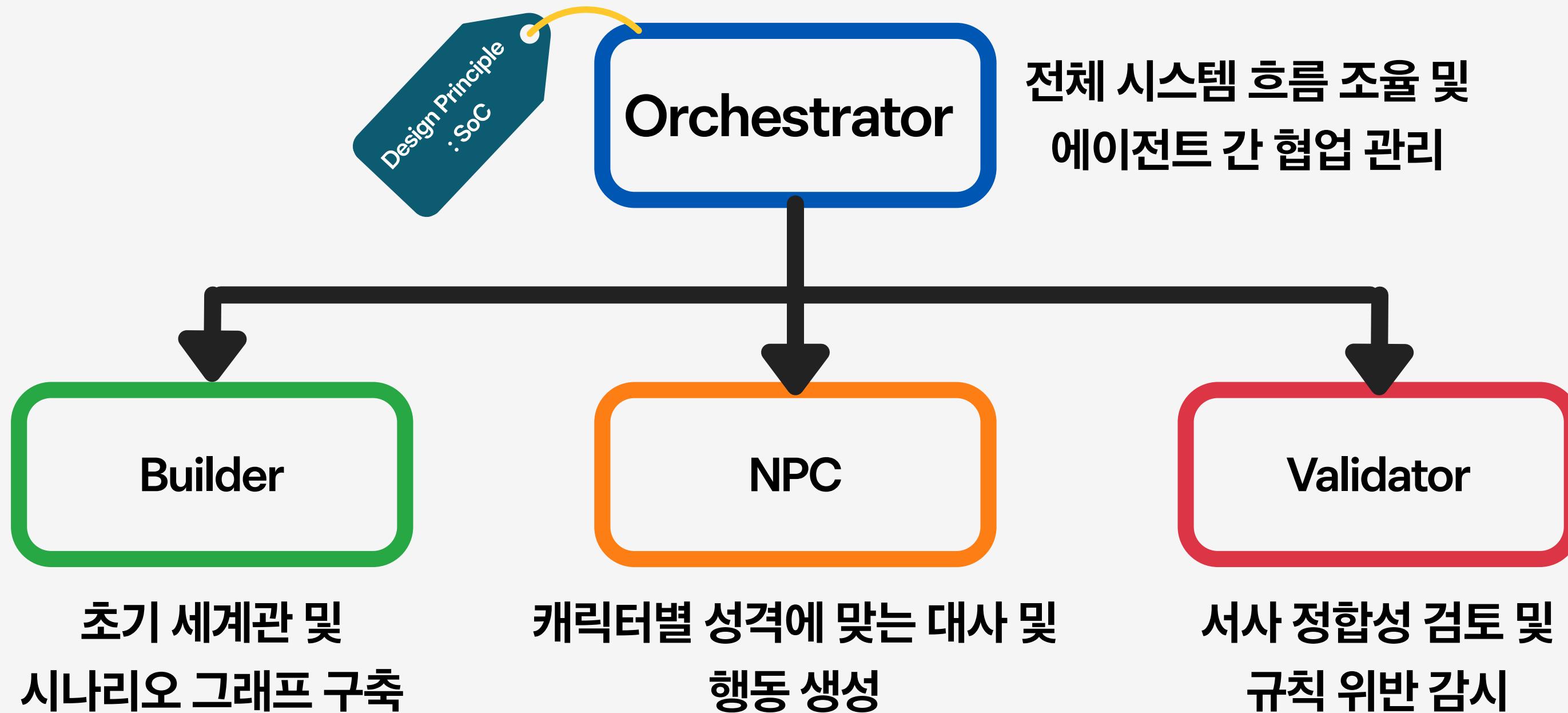
04



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.

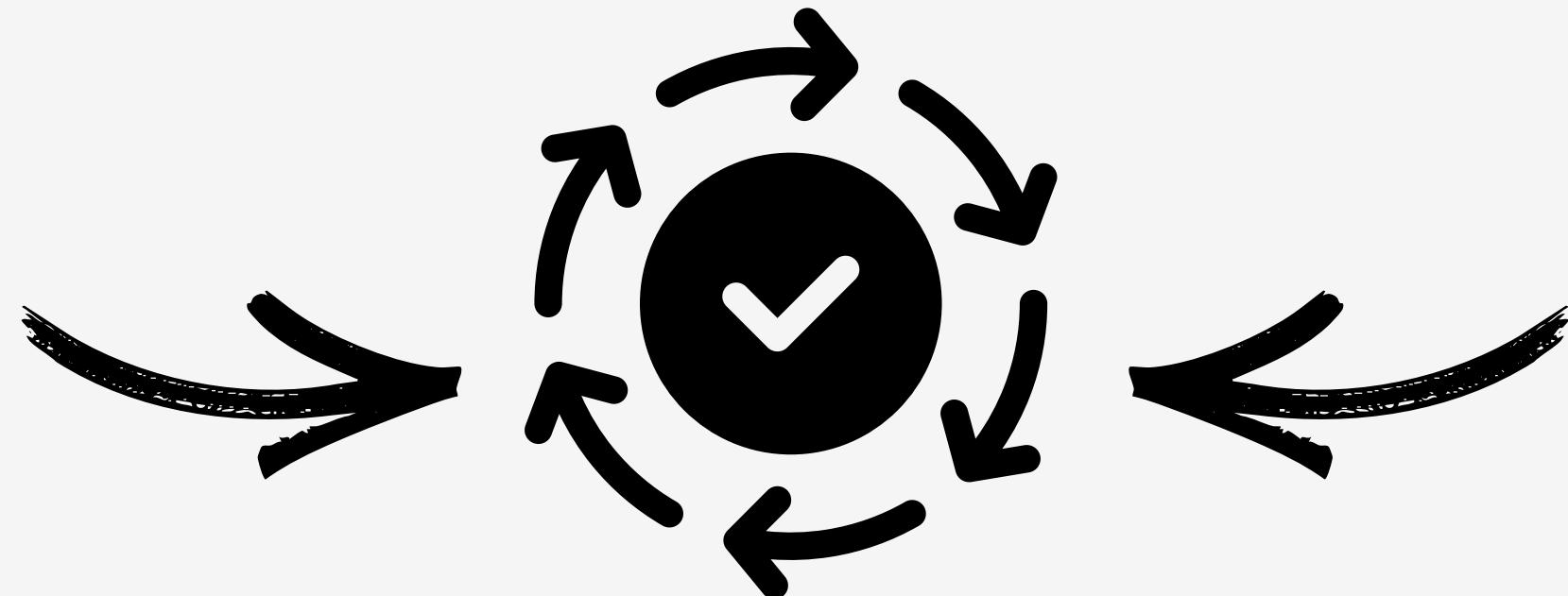
4 컨트롤 타워

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

✓ NPC & Narrator 에이전트



Narrator
(전지적 서사 가이드)



일관성 유지



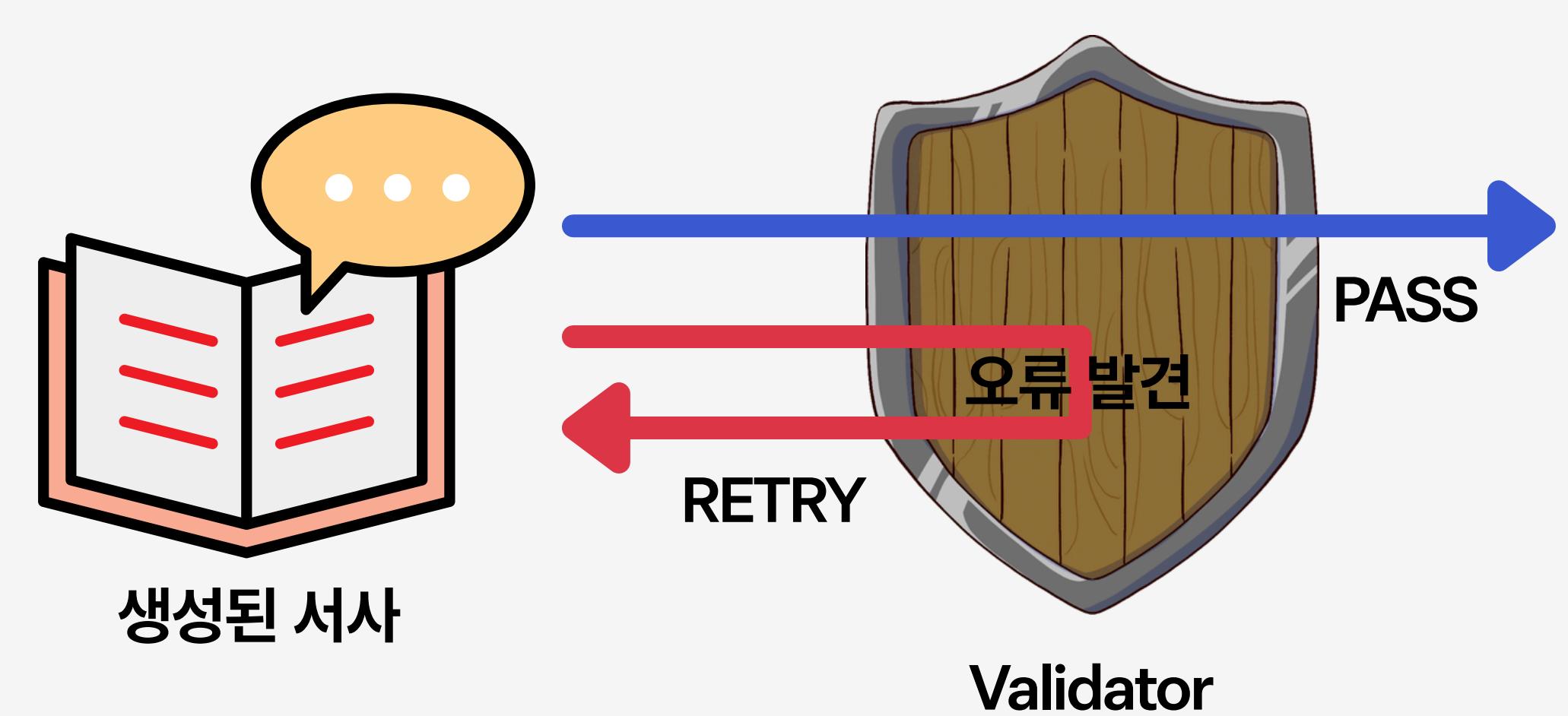
NPC 에이전트
(페르소나 구현)

전지적 묘사와 개성 있는 대사의 완벽한 분리, 체계적인 내러티브 전달

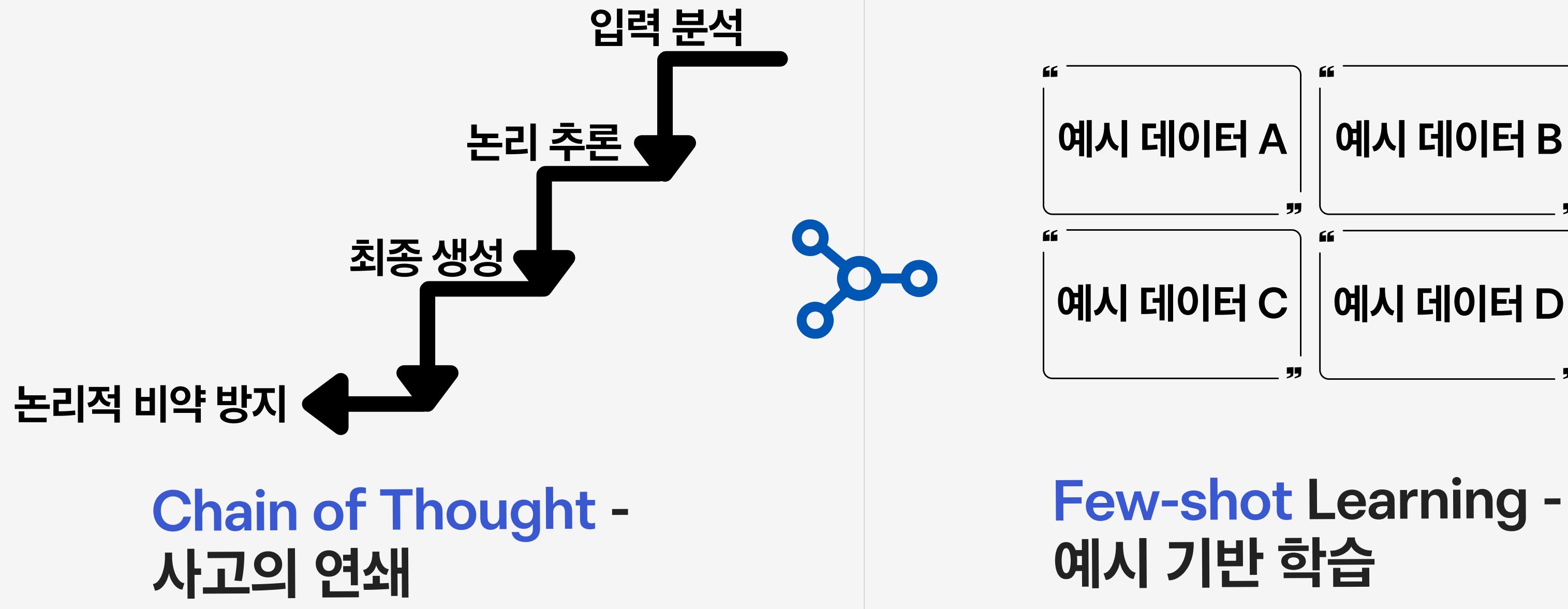
✓ Validator 에이전트



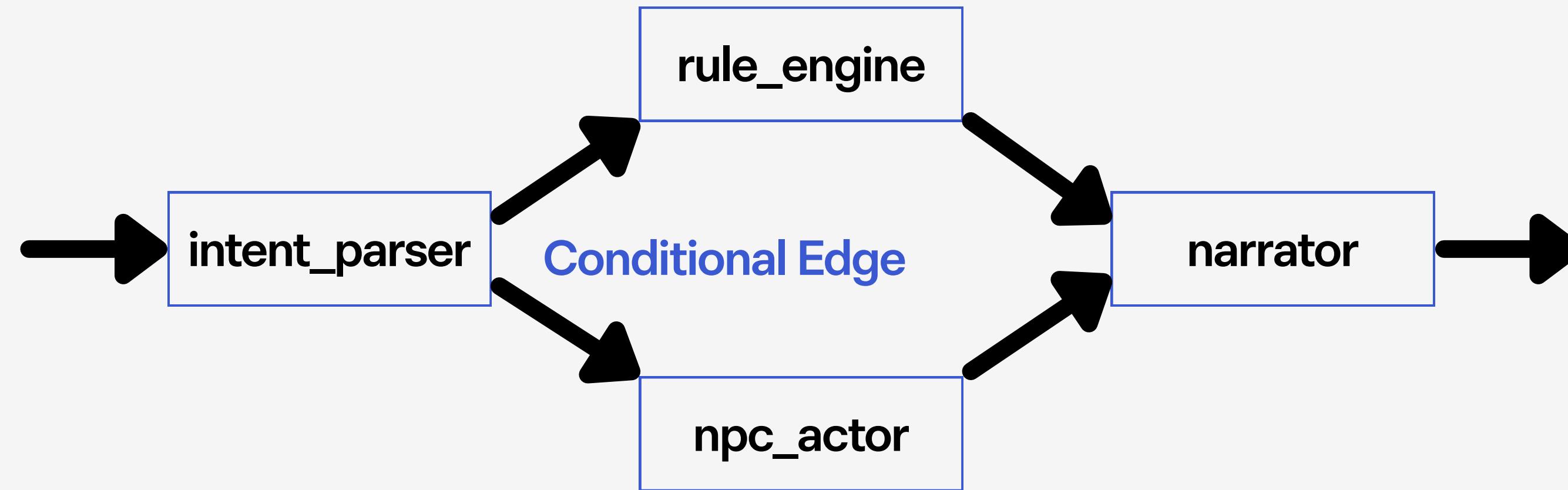
설정 오류 및
서사 붕괴
원천 차단 엔진



✓ 프롬프트 엔지니어링



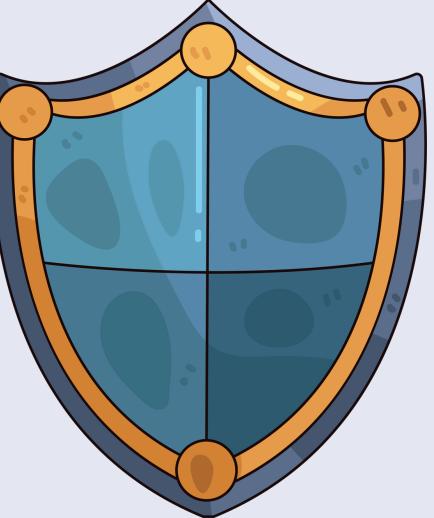
✓ 에이전트 간 협업 프로세스



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

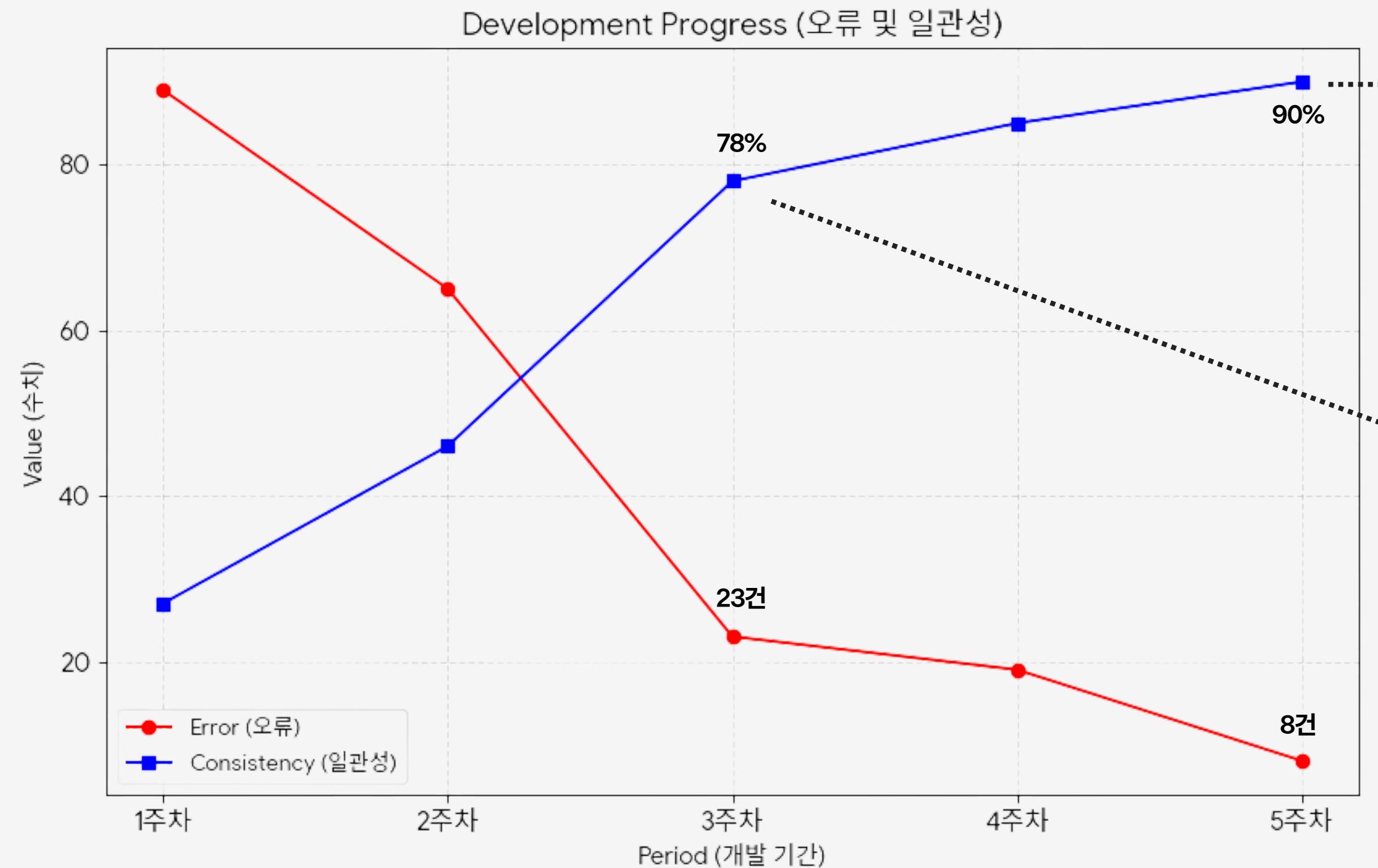
03



Loop Verification

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

✓ 품질 평가 지표

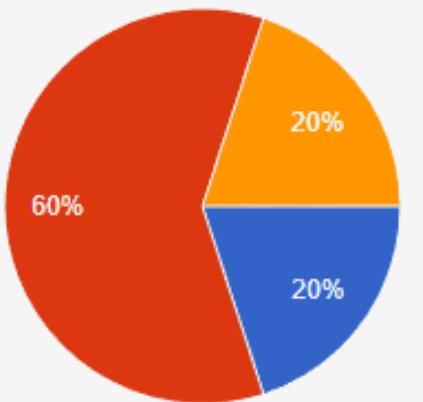


실서비스 수준의
정합성 확보

Validator 에이전트
도입 후 급격한 안정화

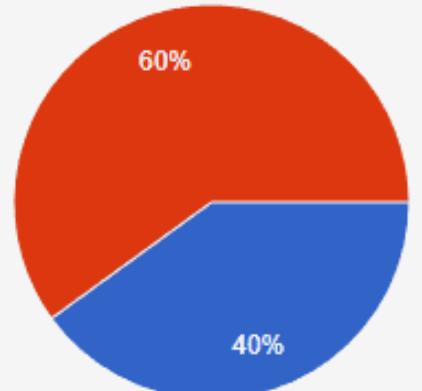
✓ 유저 평가

전반적인 평가



- 5 (정말 좋았다.)
- 4 (나쁘지 않았다.)
- 3 (그저 그랬다.)
- 2 (생각보다 별로였다.)
- 1 (기대 이하였다.)

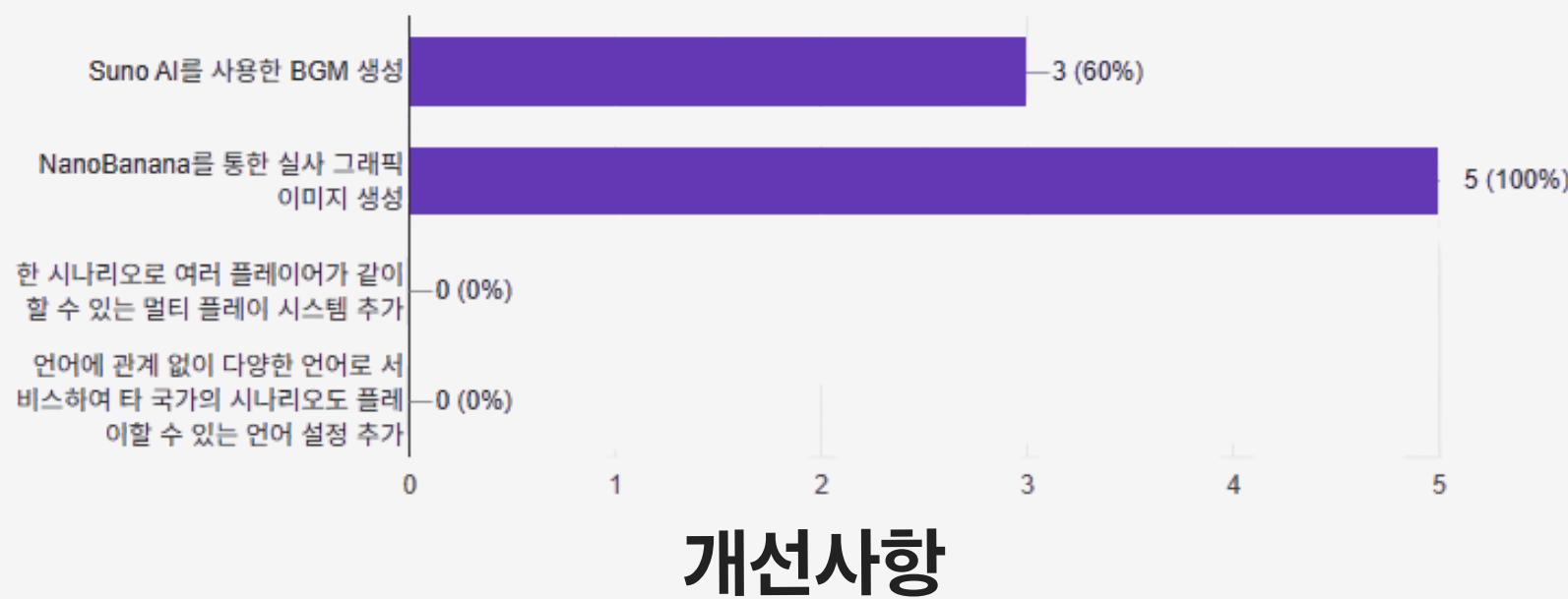
출시 후 이용할 의향



- 5 (출시된다면 할 것이다.)
- 4 (출시된다면 할 의향은 있다.)
- 3 (출시되어도 생각해 보겠다.)
- 2 (출시되어도 바로 할 의향은 없다.)
- 1 (출시되어도 할 의향은 없다.)

유저 개선 요청 사항

- 게임 플레이 도중 플레이어 행동으로 시나리오가 초반으로 이어져 다시 처음부터 하는 경우가 있었습니다. 선택지가 있었으면 좋을 것 같습니다.
- 다양한 컨텐츠가 나오면 더 좋겠습니다.
- 플레이 및 빌드 과정이 어렵습니다.



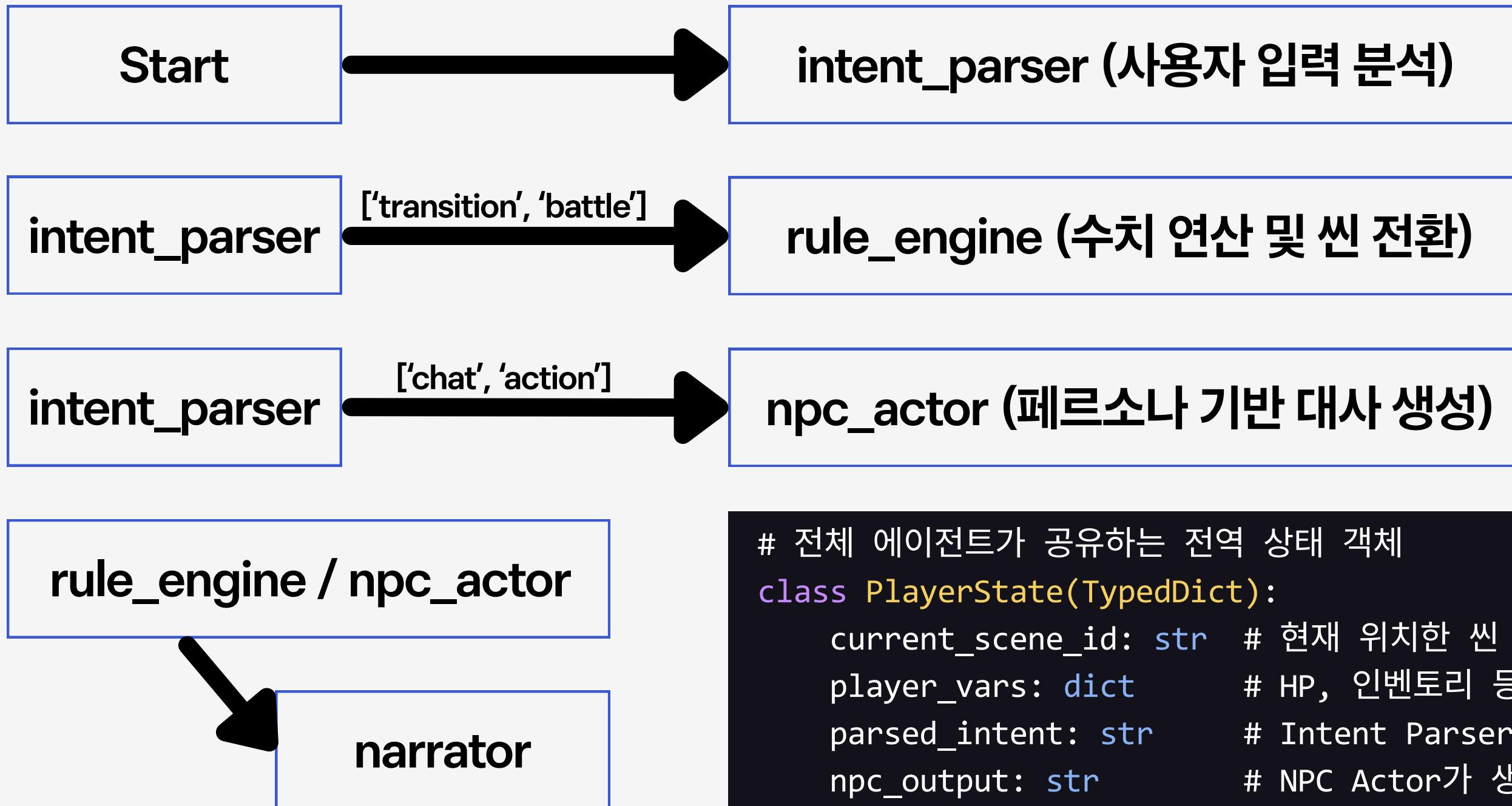
05

시스템 로직 및 내러티브 엔진

설계를 넘어 실시간으로 작동하는
서사 알고리즘과 판정 시스템

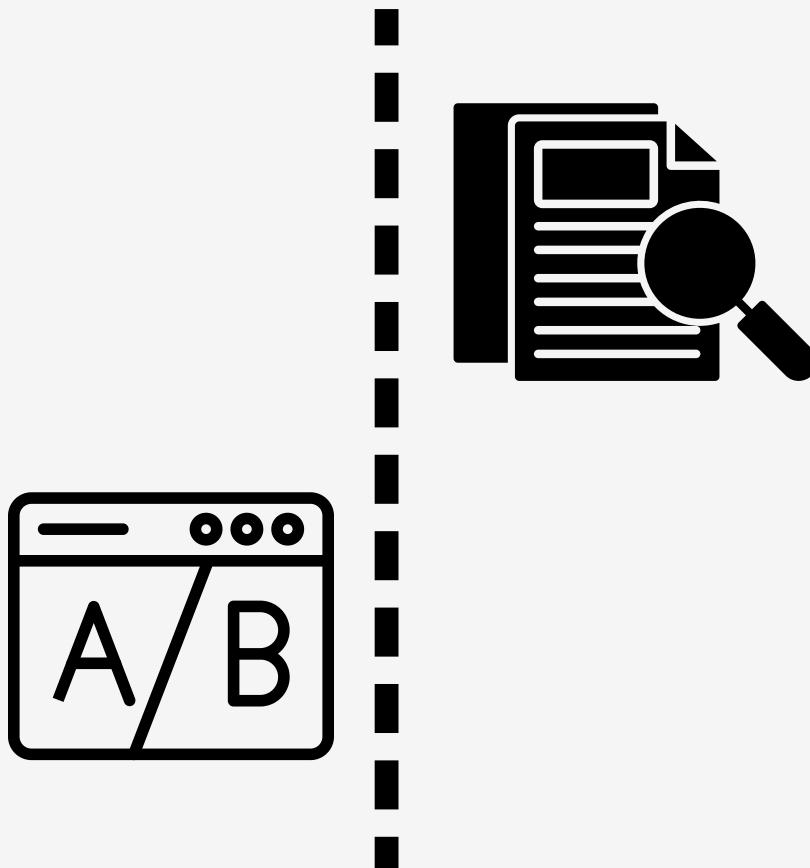


✓ LangGraph 기반 워크플로우



✓ Intent Parser

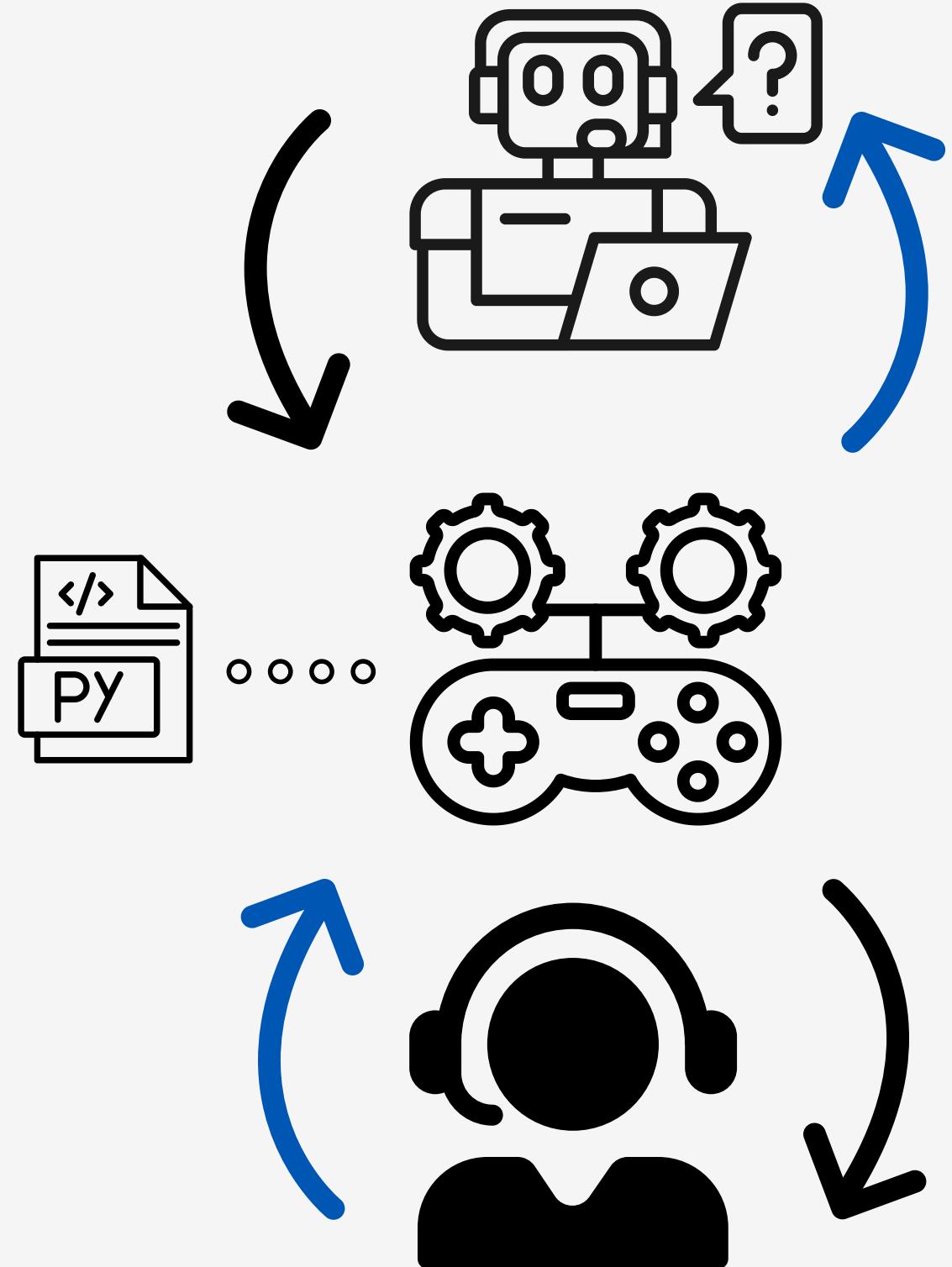
칼을 휘두른다.



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

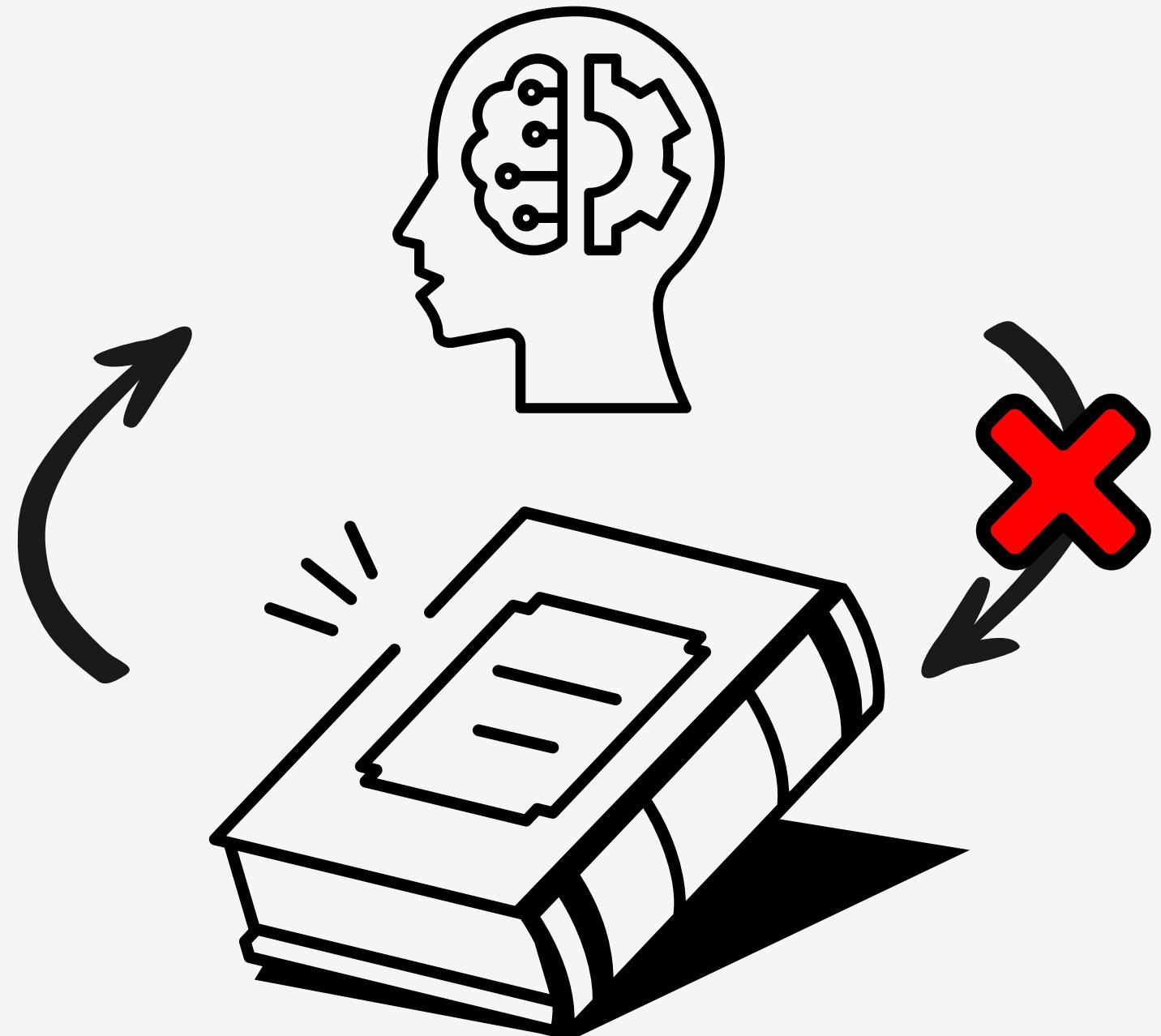
"action": "Attack"

✓ 게임 규칙 엔진



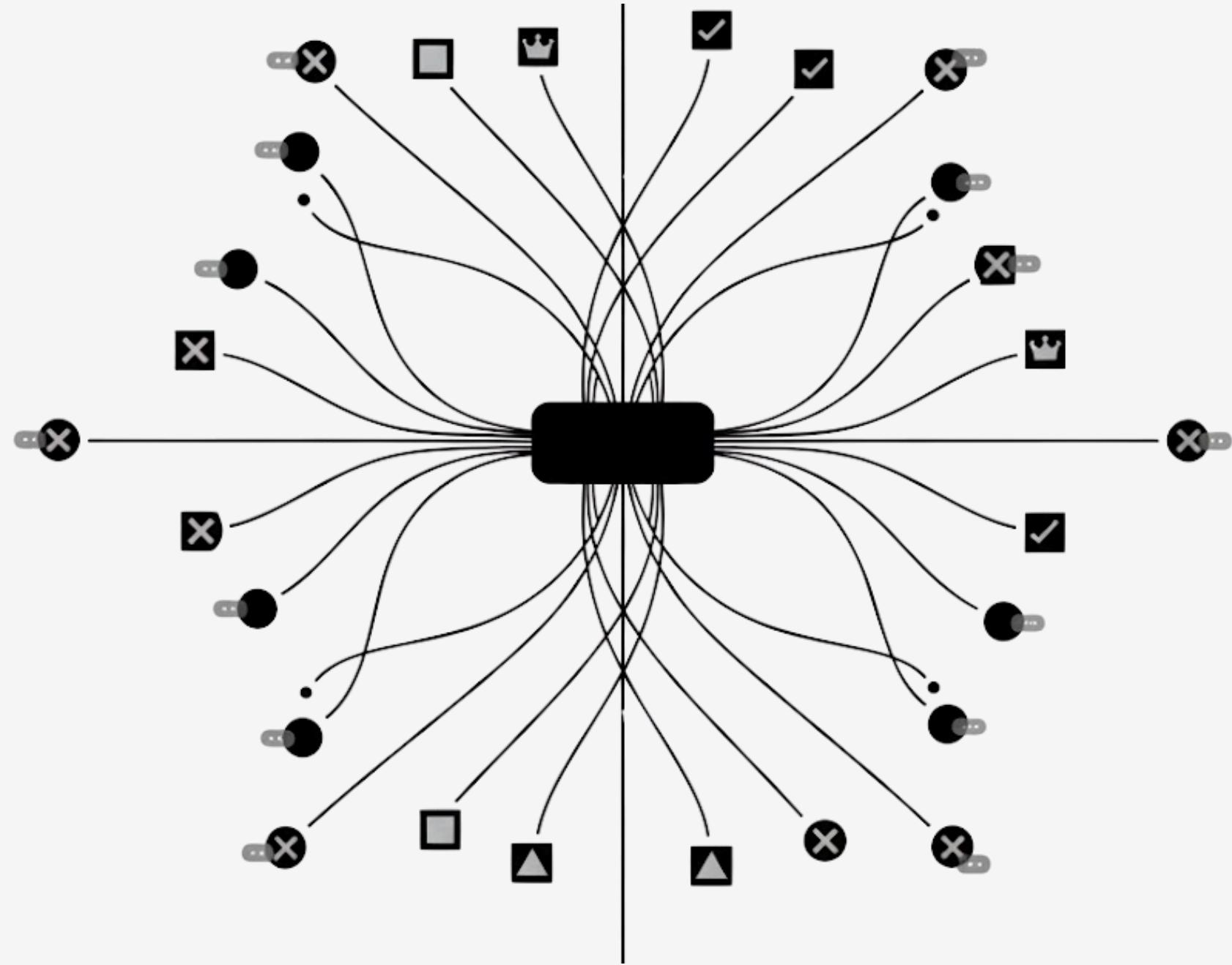
AI의 상상이 아닌 **코드의 논리**로 결정되는,
환각 없는 공정한 게임 마스터링

✓ World State Manager



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

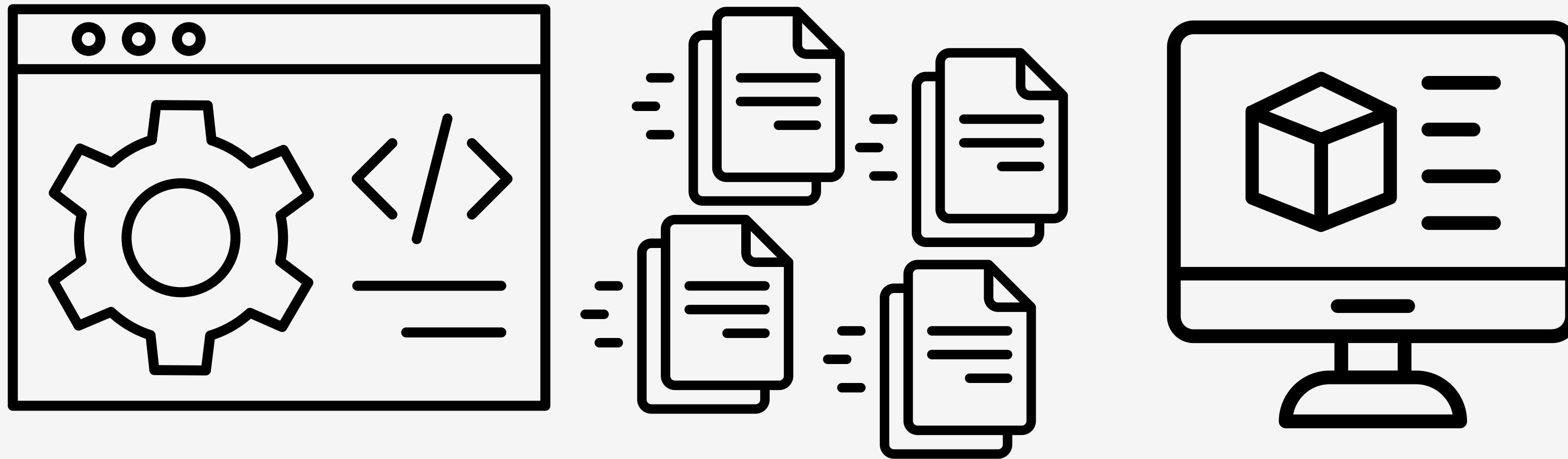
✓ 시나리오 브랜칭



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

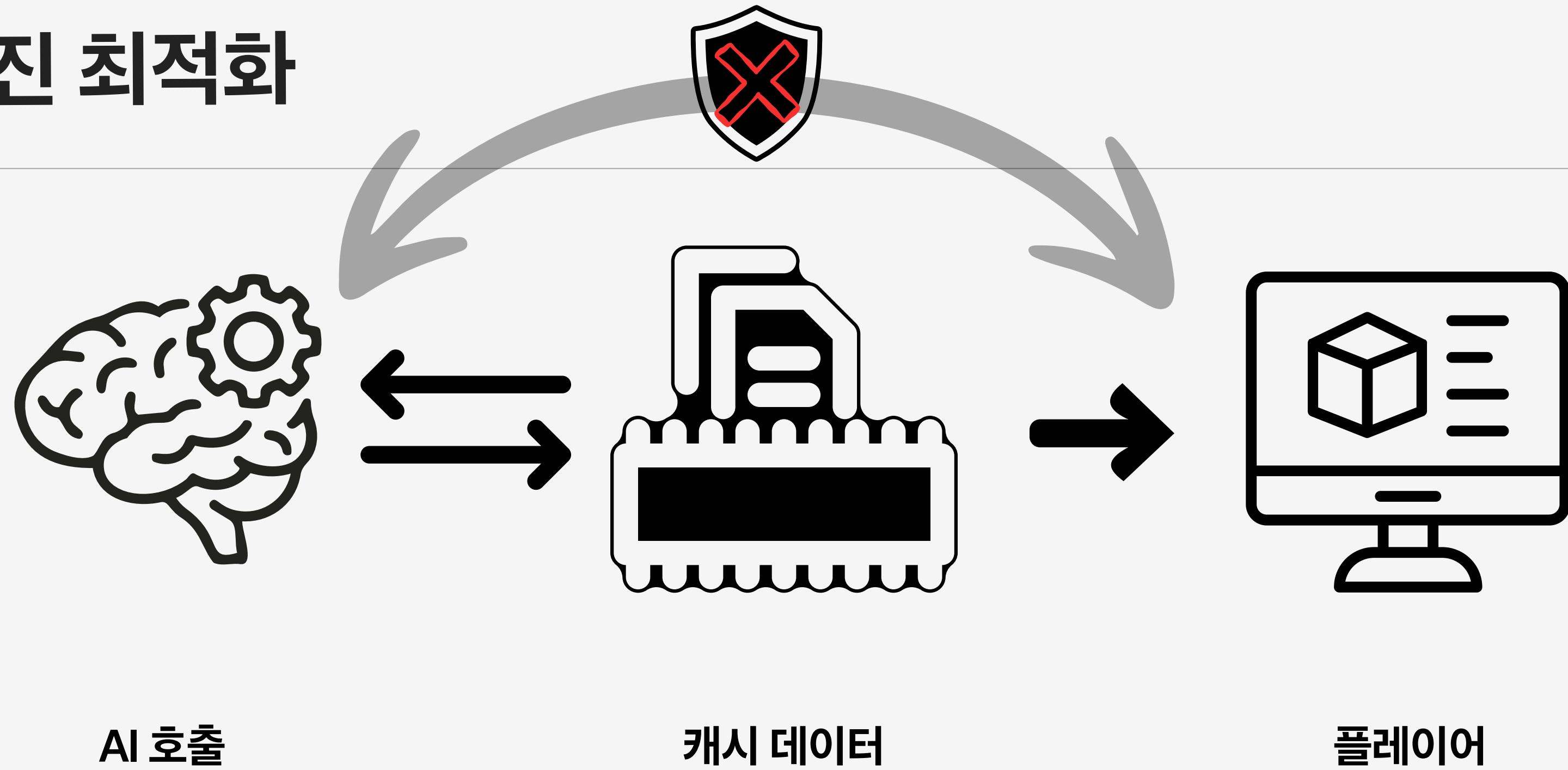
✓ 실시간 스트리밍



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



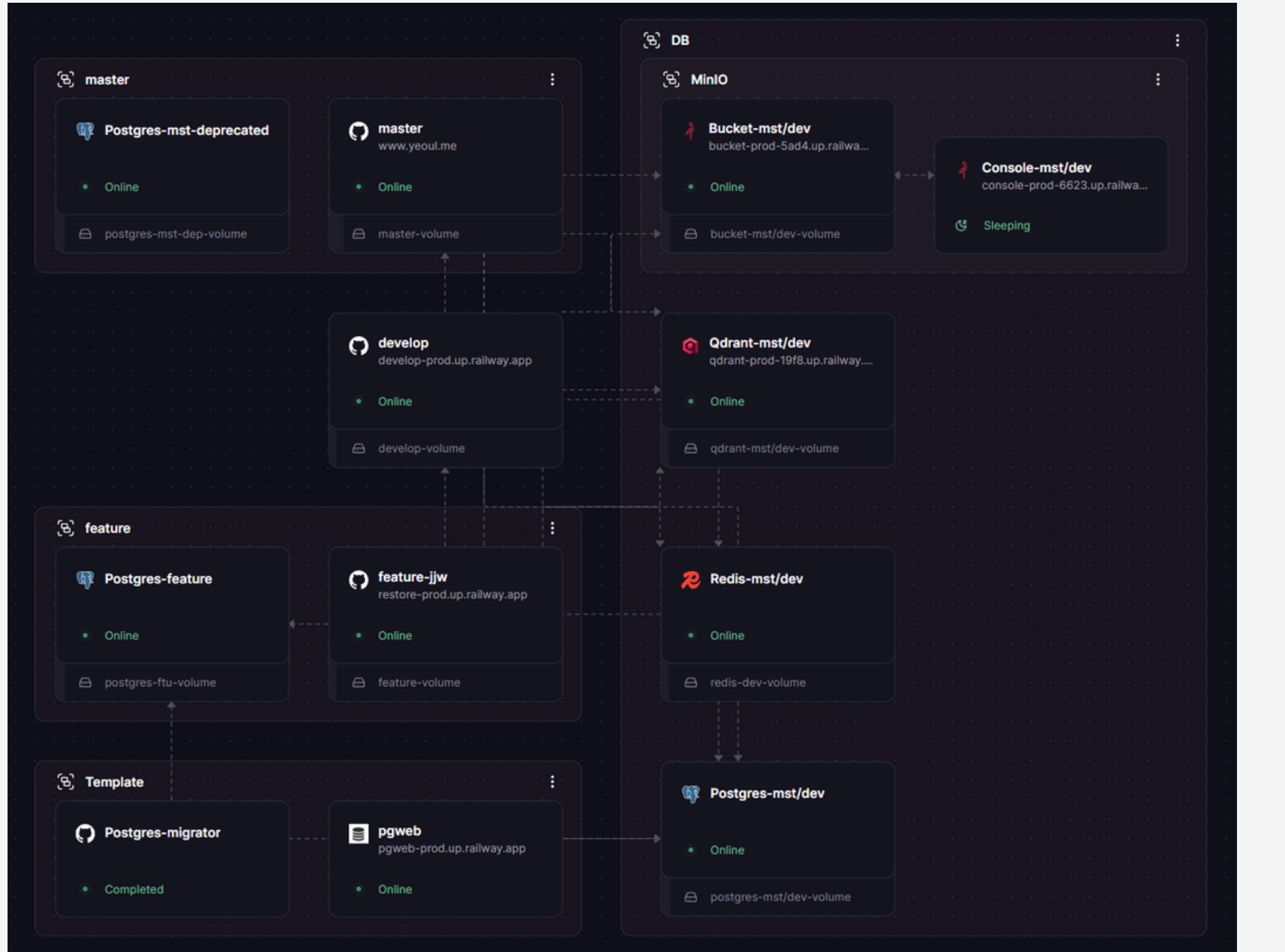
시스템 아키텍처

INFRASRUCTURE
DEPLOYMENT

안정적인 서비스 제공을 위한

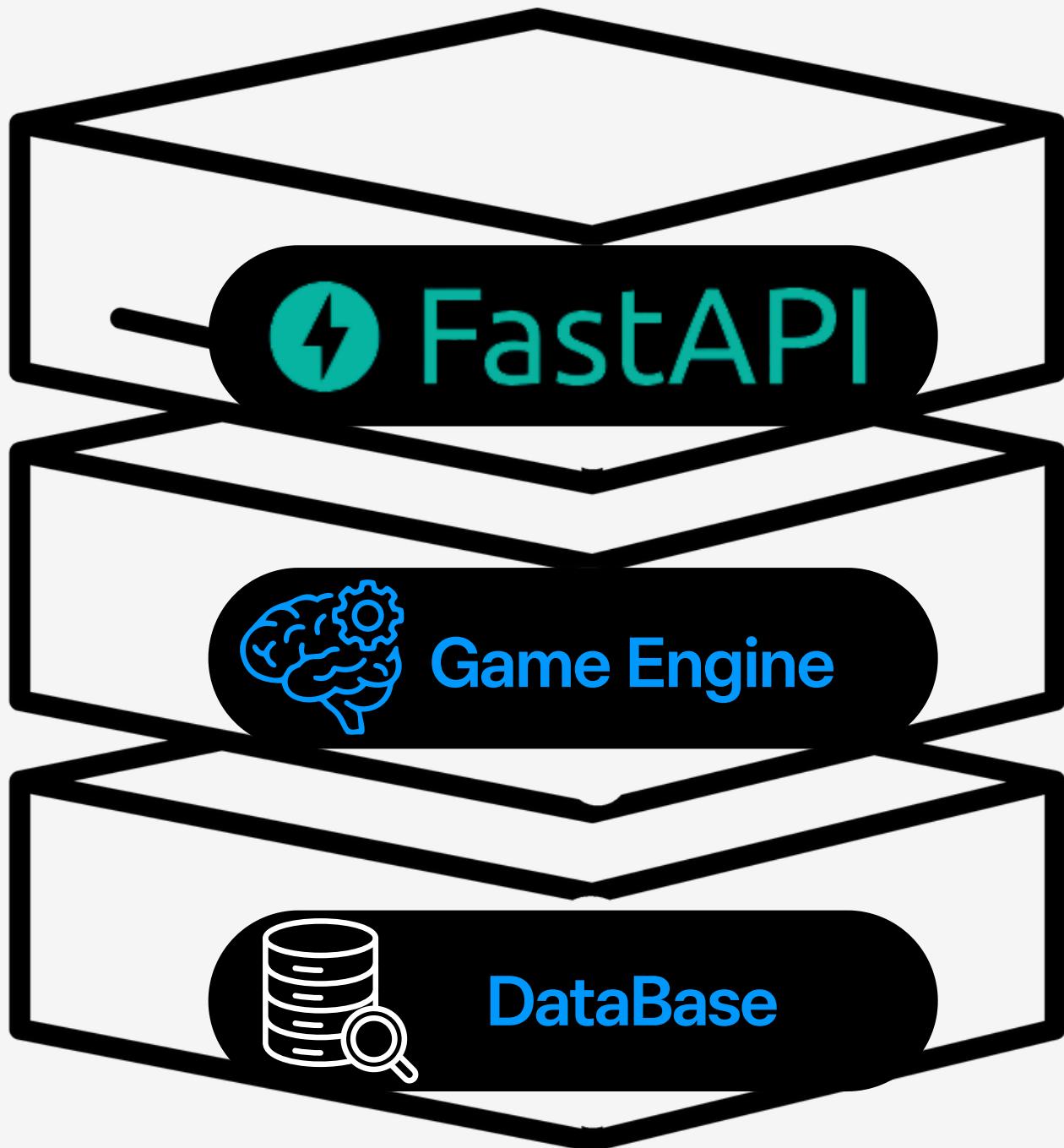
Docker 기반의 클라우드 인프라 구축

✓ 전체 서비스 인프라 구성도



Docker와 Railway로 구현한
안정적이고 효율적인
클라우드 네이티브 인프라

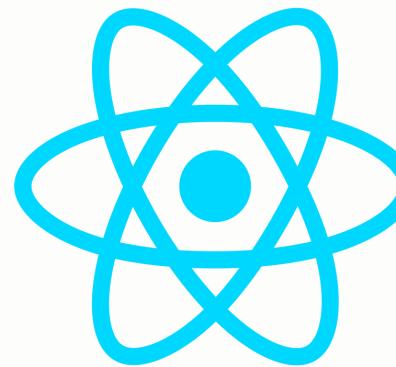
✓ 백엔드 아키텍처



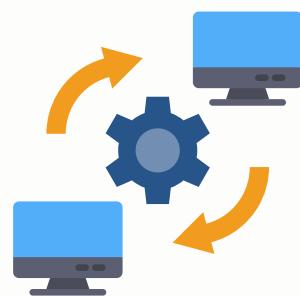
FastAPI와 레이어드 아키텍처로 완성한
고성능·확장형 백엔드 시스템

✓ 프론트엔드 아키텍처

기술 스택 & 성능



Jinja2 템플릿 + React 활용



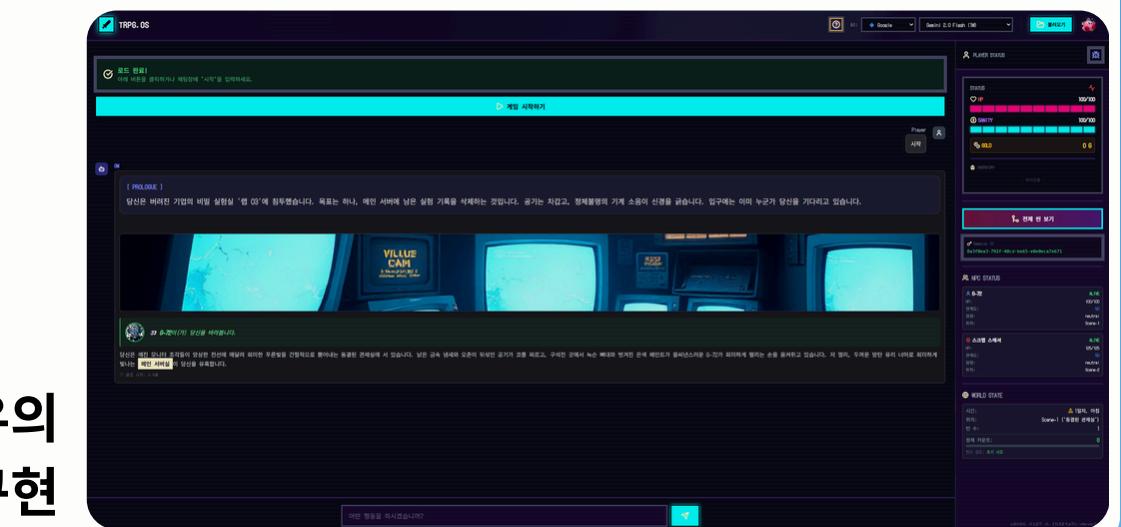
동기/비동기 인터페이스 구축

디자인 & 분위기



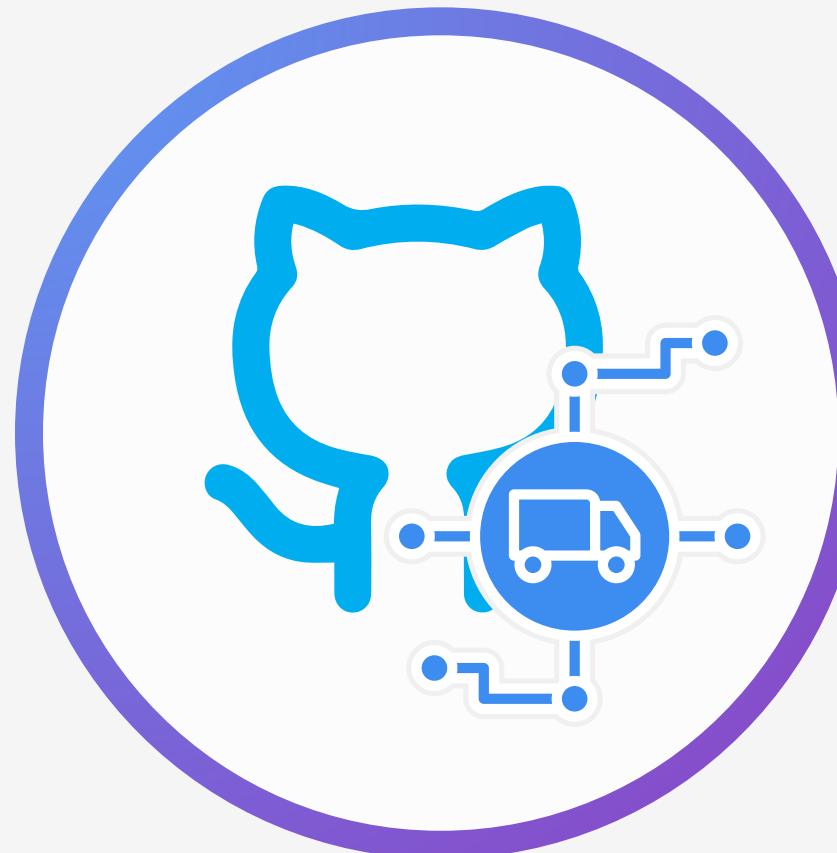
픽셀 아트 스타일 & 다크 모드 적용

TRPG 특유의
몰입감 있는 분위기 구현



✓ 인프라 배포

자동 CI/CD 파이프 라인



GitHub 코드 푸시 시
자동 빌드 및 배포

Railway Paas 플랫폼



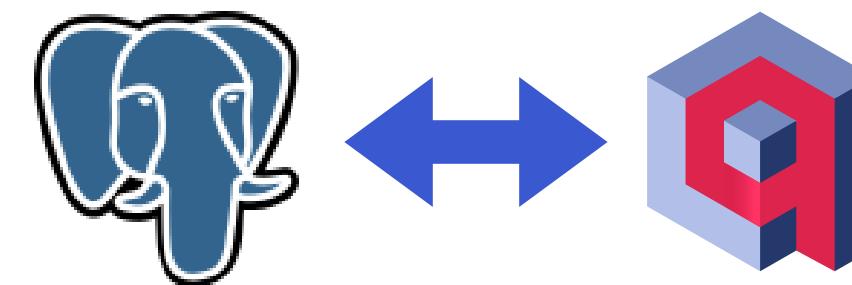
서버관리 없는 완전 관리형
서비스 및 안정적 운영

✓ 데이터베이스 구성

메타 데이터와 벡터지식의 하이브리드 구조 및 스키마 자동관리

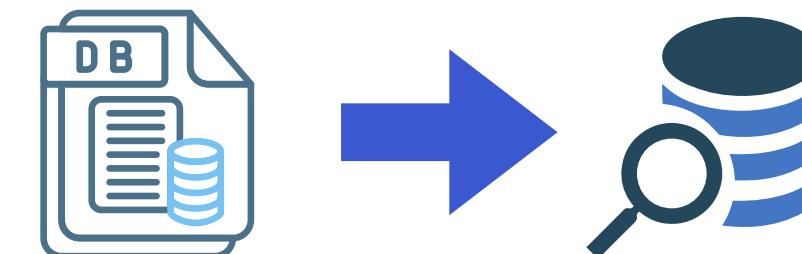
저장소	데이터 유형	주요 데이터 예시
 PostgreSQL (관계형 DB)	정형 메타 데이터	유저/캐릭터 정보, 게임진행 상태, 배경설정
 Qdrant (벡터 DB)	비정형/벡터 지식	룰북 임베딩, NPC 대화 기록, 배경정보 청크

하이브리드 저장소 전략



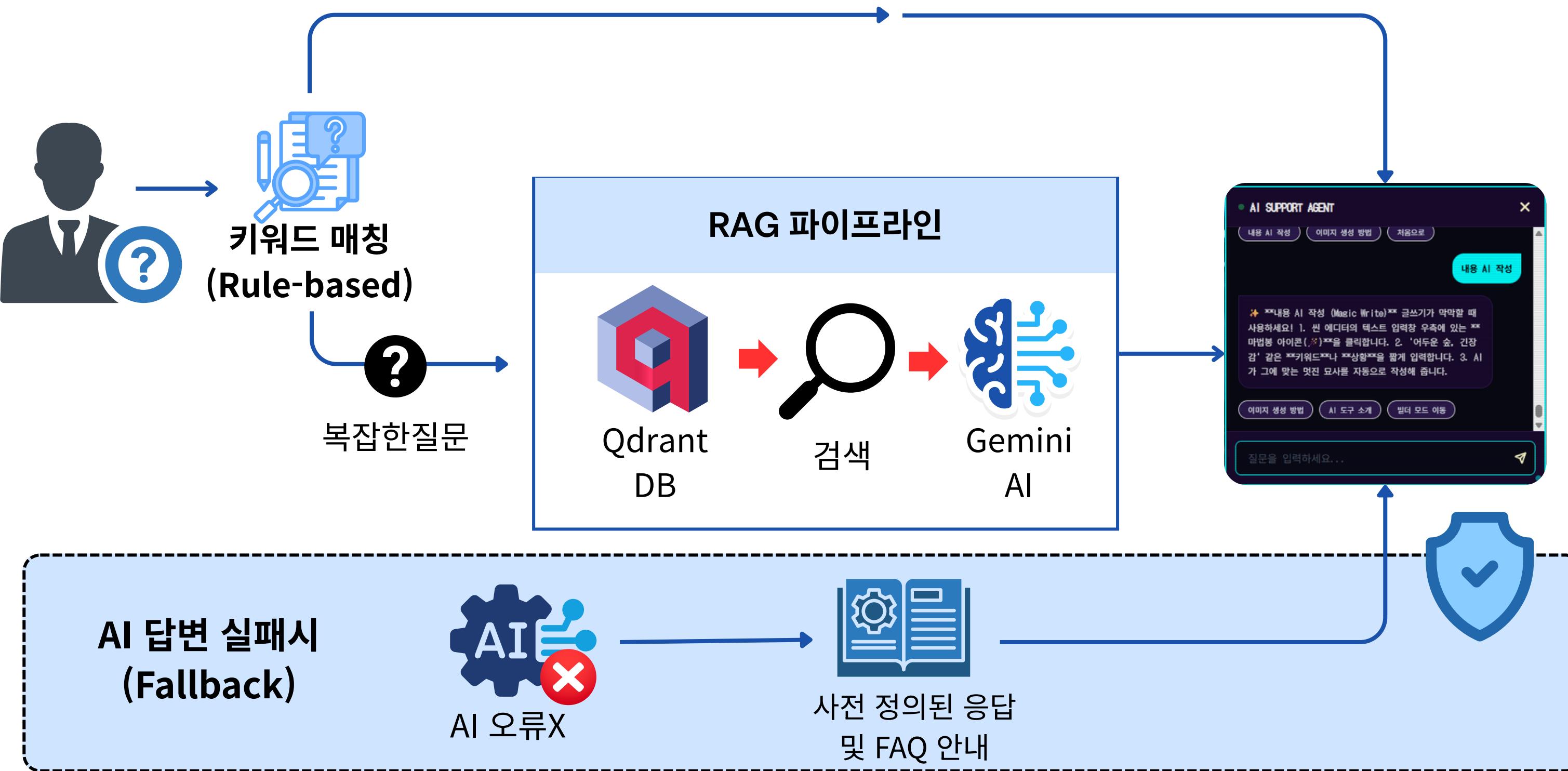
데이터 특성에 따른 최적화된 분산 저장

스키마 자동 관리 (`init_db.py`)



빌드시 스키마 자동 생성 및 갱신

✓ 지능형 가이드 챗봇



✓ 보안 및 인증

DDoS 방어 및 DNS 보호(네트워크)



Cloudflare 네임서버 이전으로
악성 트래픽 필터링 및
WAF 적용



민감 정보 환경 변수 격리 (서버)



API 키(Gemini,DB등)를
소스 코드에서 완전 분리하여
실서비스 보안 강화



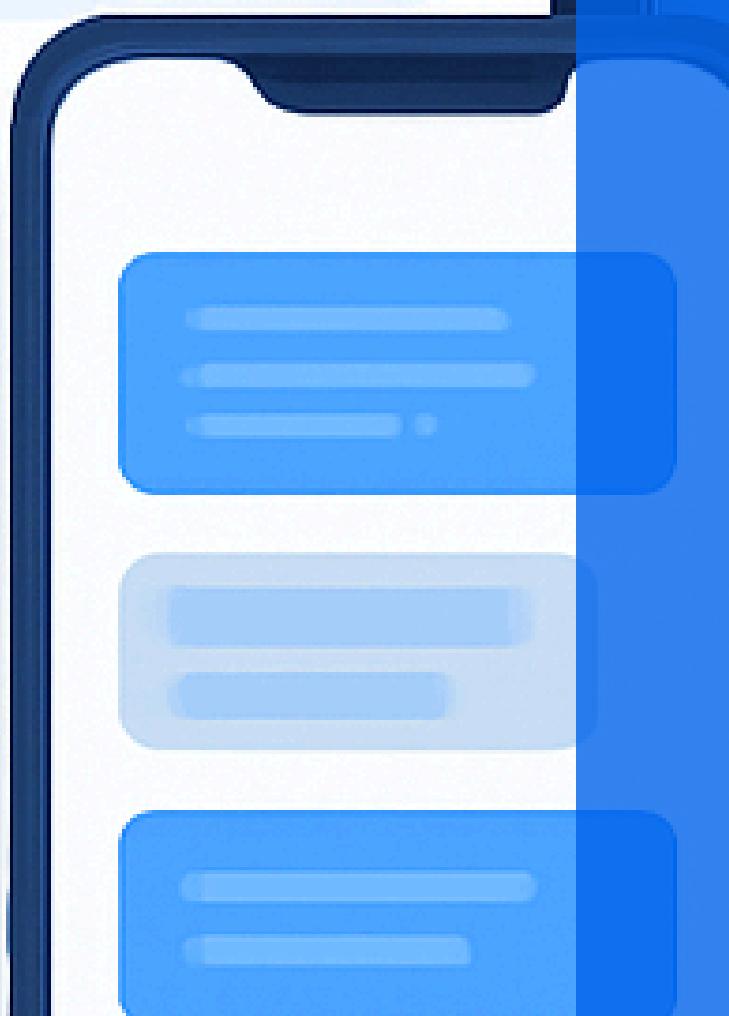
안전한 인증 및 세션 관리(애플리케이션)



구글/카카오 소셜 로그인 및
토큰 기반 세션으로
비밀번호 유출 원천 차단



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

Trouble 1

서버 타임아웃

LLM 호출에만 의존하면
호출 시간이 길어지거나
서버가 닫히는 문제 발생

비동기 및 스트리밍으로
응답 신뢰성을 높여
타임아웃을 원천 봉쇄

Trouble 2

AI 환각 억제

LLM이 모든 요소를 조정하면
환각이 발생하거나
수치가 조정되지 않는 현상 발생

WorldState로 요소를 관리하여
논리적 일관성을 통일
서사보다 데이터가 상위 개념

Trouble 3

웹 연동 보안

도메인을 구매하여
Railway 환경에서 구성하니
실제 봇을 통한 탈취 로그가 발견

CloudFlare 및 환경 변수
개별 저장, OAuth 2.0 도입으로
최소한의 보안성을 유지

✓ 트러블슈팅 2

Trouble 4

미숙한 협업체계

깃 허브 CI/CD로 버전을 관리
그러나 실제 협업 시
충돌 및 코드 삭제 등이 발생

커밋 문구에 수정한 파일 및
수정 방향성 등을 작성하도록 지시

Trouble 5

부족한 상상력

프로젝트 진행 중 새로운 기능이나
실제 필요한 기능을 추가할 때
신선한 시각이 부족

Closed Beta Test를 자체적으로
진행하여 구글 품으로 실제 플레이
경험을 토대로 개선 방향 수집

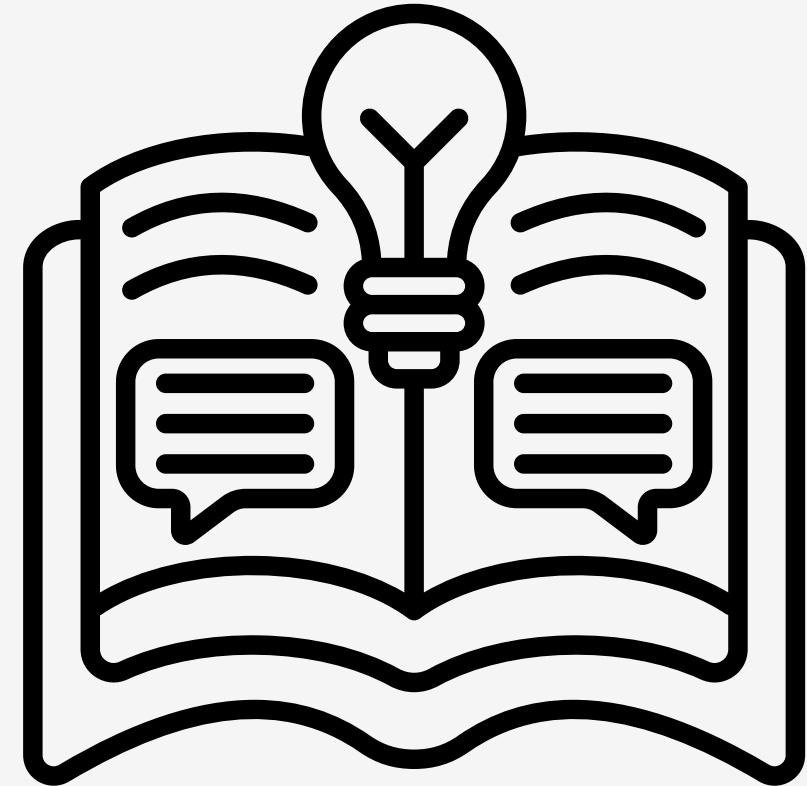
Trouble 6

디버깅 로그 강화

Railway의 기본 로그만으로는
비동기 AI 로직의 에러 추적이 어려워
전역 커스텀 로거를 도입

상세 디버깅 체계를 구축하여
배포 환경에서도 실시간으로 에러를
파악하고 대응 시간을 획기적으로 단축

✓ 프로젝트 성과 요약



서사 유지력 및 기술적 무결성 확보

RAG, WorldState, SSE 스트리밍,
비동기 등으로 서사 유지력과 기술적 무결성을
확보해 실제 서비스가 가능할 정도의 인프라
안정성 확보



창작 문턱의 파괴

복잡한 코딩이나 룰북 숙지 없이, 오직 텍스트
입력만으로 고품질 TRPG 시나리오를 자동 생
성하는 환경을 구축



플랫폼 확장성 확인

단순 게임을 넘어 누구나 자신만의 세계관을
공유하고 플레이할 수 있는
콘텐츠 플랫폼으로서의 가능성은 확인

✓ 한계점 분석

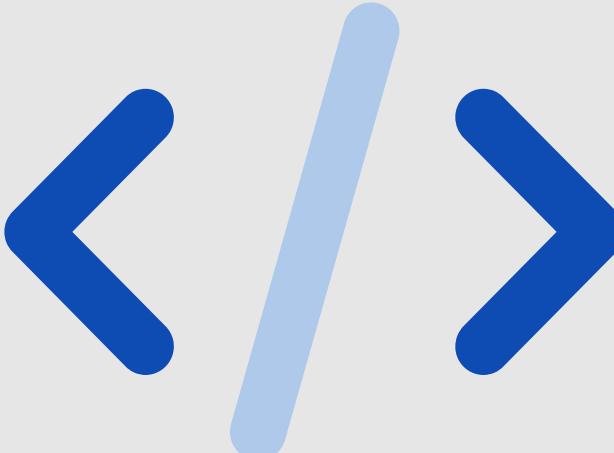
01



운영/비용적 한계

고성능 LLM 호출에 따른 API 비용 부담 및 서비스 스케일업 시 운영 효율성 확보 필요

02



기술적 구현의 한계

현재 전투 로직이 기초적인 수준이며, 복잡한 TRPG 룰북의 물리 법칙을 100% 반영하기엔 미흡함

03



데이터 의존성

RAG가 참조하는 세계관 데이터의 양과 질에 따라 AI 답변의 퀄리티 편차가 발생하는 문제 존재

✓ 서비스 고도화 1 - 멀티플레이 확장

01



멀티 유저 세션

싱글 플레이를 넘어 다수의 사용자가
하나의 시나리오에 동시 접속하는 환경 구축

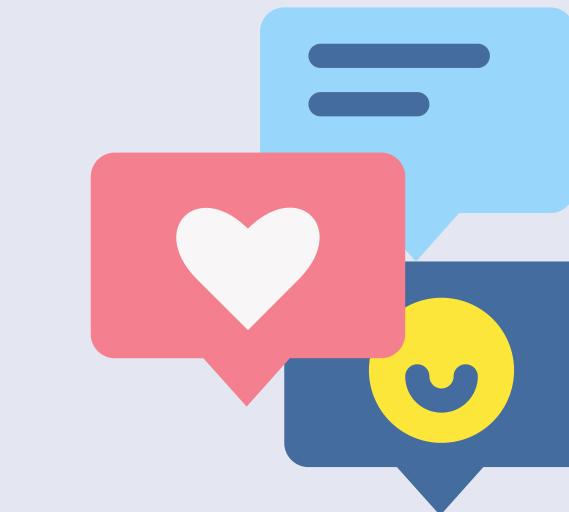
02



실시간 턴 제어

여러 명의 행동이 충돌하지 않도록
AI 마스터가 주도하는 정교한 턴 관리 로직 도입

03



협동 서사 상호작용

한 명의 선택이 파티 전체의 운명을
결정하는 다대일 서사 구조 구현

혼자 하는 모험에서 우리를 위한 서사로: 실시간 멀티플레이 파티 시스템 구축

✓ 서비스 고도화 2 - 멀티미디어 확장

01



상황 맞춤형 BGM

서사의 분위기(긴박, 평화 등)를 AI가 분석하여 배경음악 실시간 생성 및 재생

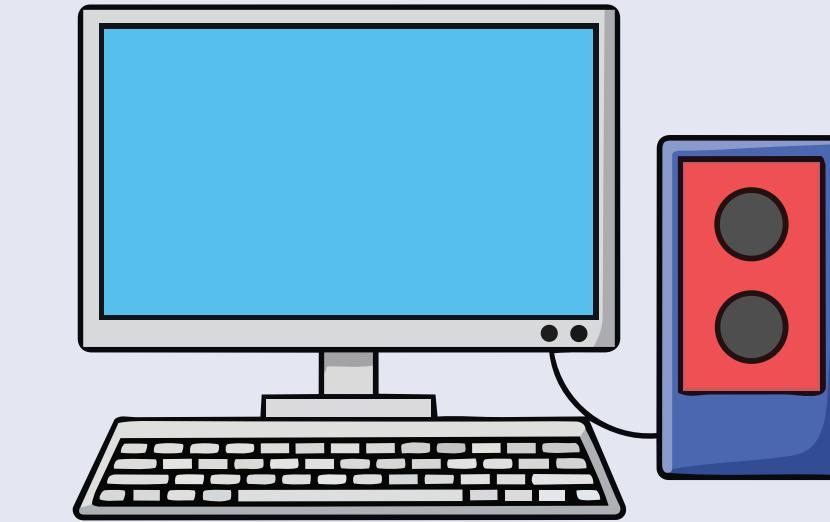
02



음성 합성(TTS)

NPC와 내레이터의 페르소나를 살린 고품질 음성을 입혀 청각적 몰입감 제공

03



입체적 서사 경험

시각(텍스트/이미지)과 청각(BGM/TTS)이 결합된 멀티미디어 환경 구축

텍스트를 넘어 전율로: AI 기반 BGM과 TTS가 완성하는 궁극의 몰입형 서사

Project YEOUL

「AI - Story - Human」

“Thank You & Q&A”