

문제를 찾아내는 것이 제 역할입니다.

기술은 그 문제를
해결하기 위한
수단입니다.

김가람 Garam Kim

kimgalam0831@gmail.com
github.com/kgr0831

김가람

Garam Kim

AI Native Software Engineer

문제를 먼저 찾고,
해결할 수단을 선택합니다.

✉ kimgalam0831@gmail.com

🐙 github.com/kgr0831

🌐 linkedin.com/in/kimgalam0831

🔗 kgr0831.tistory.com

문제를 대하는 방식

업무와 제작 과정의 불편에서 문제를 찾습니다. 기술은 그 문제를 해결하기 위한 수단입니다.

정글에서 다진 CS 기본기와 AI를 바탕으로 필요한 기술을 익히고, 처음 발견한 문제가 해결됐는지 확인합니다.



MOAI · AX Engineer

인턴 · 2026.09 – 현재

사용 기술



Unity



Next.js

AI 도구



Orca



Codex



Hermes

Website ↗

kgr0831.github.io/

문제를 풀며 쌓아온 경험

경력 · 연구

2026.09 – 현재



MOAI

인턴 · AX Engineer

AI 사용량 원장 · 관리자 기능 · 리드봇

2026.04 – 2026.08



숭실평화통일 연구원

SmartAI Lab · 학부연구생

평성시 사회 시뮬레이션 연구 참여

2026 KOSSCCHTHON



대상 · 1위

국민대 · 숭실대 · 순천향대

연합 경진대회

학력 · 교육

2026.03 – 현재



숭실대학교

AI소프트웨어학부

2025.09 – 2026.01



크래프톤 정글

SW-AI Lab 11기

PintOS · EUM 프로젝트

2023.03 – 2026.02



한국게임과학고등학교

컴퓨터게임개발과

학생부회장·학생회장 / 개발 동아리 Dash 창립·회장

어떤 문제를 찾아왔는가

- | | | | |
|----|----------------|---|---------|
| 01 | 반복 업무와 도구의 단절 | MOAI · 겐마루 · Abstraction Bot
무인 스크린골프 CMS · PaperBridge | 05-14 ↗ |
| 02 | 사용을 방해하는 품질 문제 | EUM · PintOS
평성시 사회 시뮬레이션 | 15-20 ↗ |
| 03 | 제한된 시간과 제작 인력 | 두드리 · Spell Thrower · PlanetGame · Triniacle
SpriteAIMaker · Remnants of Light / ROLAI | 21-36 ↗ |
| 04 | 새로운 서비스 수요 | Virtual Friend
음성·일러스트·기억과 사업화 구상 | 37-38 ↗ |

회사·개인·동아리의 반복 업무를 자동화

사용 환경	프로젝트	발견한 문제	자동화 범위
회사	MOAI 사용량 원장	흩어진 AI 사용 기록	사용량 수집·기록·확인
회사	MOAI 관리자 · 리드봇	작업 조회와 후속 조치의 분산	관리 API · 채팅 기반 업무 연결
동아리 · 개인	겜마루 웹 · 데일리 브리프	매일 반복되는 정보 탐색·공유	조사 → 보고서 → 보관·알림
동아리 · 팀	Abstraction Bot	대화에 묻히는 논의와 결정	수집 → 요약 → 기록
사업장	무인 스크린골프 CMS	예약·기기·결제·정산의 분리	운영 관리 · 원장 기반 자동정산
연구자	PaperBridge	논문과 AI 도구 사이의 왕복	선택 문맥 → 질문 → 근거 확인



MOAI · 업무 자동화와 관리자 기능

MOAI · 인턴 · AX Engineer · 2026.09 – 현재

흩어진 AI 사용 기록과 업무 현황을 확인하고, 채팅 요청을 실제 후속 작업으로 연결할 필요가 있었습니다.

AI 사용량 원장

☰ 흩어진 사용 기록을 한곳에

도구별 사용 내역을 수집·기록하고 확인하는 원장 운영에 참여했습니다.

관리자 화면

🔍 조회 기준과 통계를 일치

작업 검색·필터·목록 추가 조회를 구현하고, 화면의 통계가 실제 업무 상태와 맞도록 개선했습니다.

리드봇 · 개발 중

💬 채팅에서 후속 업무로

웹 채팅과 Google Chat의 요청을 업무 도구에 연결하는 봇 개발에 참여하고 있습니다.

👉 달라진 점

관리자가 조건을 바꾸거나 목록을 더 불러와도 전체 업무 현황과 현재 조회 결과를 구분해 확인할 수 있도록 개선했습니다.

기술적 챌린지 · 통계와 목록의 일관성

필터를 빠르게 바꾸면, 늦게 도착한 이전 응답이 현재 목록을 덮어쓰는 문제가 있었습니다.

사용자 선택	이전 필터 요청	→	현재 필터 요청	→	현재 조건 유지
응답 처리	현재 응답 반영	→	이전 응답 도착	→	현재 목록에 미반영

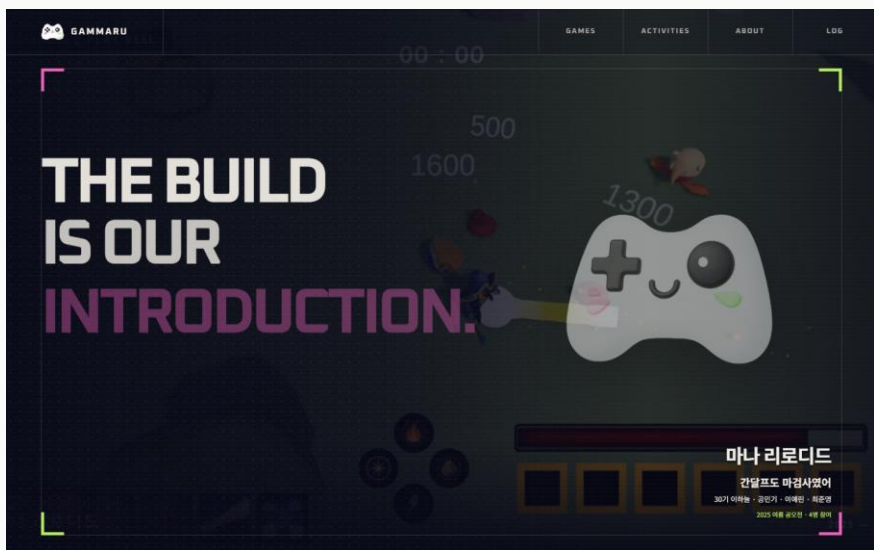
집계 ☰ 전체 통계와 목록 분리 전체·미완료·오늘 마감·기한 초과를 실제 전체 건수로 집계했습니다.	분류 ⌵ 같은 상태 기준 적용 진행 중 업무를 미완료·기한 초과 집계에도 일관되게 반영했습니다.	예외 🛡️ 실패 원인을 구분 날짜 오류·업무 대상 정보 누락·처리 실패를 구분해 안내했습니다.
---	---	--

👍

성과 · 조회 결과의 일관성

이전 요청이 늦게 도착해도 현재 선택한 조건을 유지하고, 통계와 목록에 같은 업무 분류 기준을 적용했습니다.

겐마루 웹 · 데일리 브리프



겐마루 웹 · 공개 동아리 소개와 게임 아카이브

동아리 웹 ↗ gammarupage.vercel.app/

보고서 보관함 ↗ gammarupage.vercel.app/reports

발견한 문제

매일 같은 정보를 다시 찾고 전달하는 업무

송실대 게임 제작 동아리 겐마루의 후원·지원 정보 조사가 반복됐습니다. 동아리 웹에 보고서 보관함을 연결했습니다.

본인 구현

조사 이후의 전달 과정까지 자동화

날짜별 보관·검색, Discord 로그인, 구독 승인과 알림을 구현. 개인 공모전·채용 브리프도 목적과 수신 대상을 나눠 운영.

해결 범위 · 조사 이후까지

보관·접근 권한·알림을 연결해 반복 업무 개선.

기술적 챌린지 · 반복 실행과 운영 비용

브리프 운영 흐름



운영 중 발견한 문제

같은 채널을 반복 승인

판단과 해결

이름 조회 실패와 수신 대상 변경을 구분

달라진 흐름

승인한 채널 ID가 같으면 반복 확인을 줄임

상태 확인으로 저장소 요청 증가

반복 조회를 줄이고 복구 절차를 문서화

발행·보관·알림 이후의 운영 비용까지 점검

✅ 배운 점

자동화는 결과물을 만드는 데서 끝나지 않았습니다. 보관·접근 권한·알림·운영 비용까지 이어져야 했습니다.

Abstraction Bot · 대화를 업무 기록으로

Discord 대화에 묻힌 결정과 할 일을 다시 찾는 불편을 줄이기 위해, 요약과 업무 기록을 자동화하는 도구를 만들었습니다.

Discord

메시지 수집

→ 대화 저장소

원문 저장과 조회

→ 연결된 AI 도구

대화 요약·일일 결산

→ Notion · Discord

정리와 결과 공유

기술적 챌린지 · 요약 방식의 교체

수집 봇과 요약 에이전트 분리

요약 방식을 바꿔도 대화 수집을 유지하도록 역할을 분리. 사용자가 연결한 AI 도구로 요약을 실행.

구현 성과

웹 콘솔에서 수집·연결·결산 관리

자동 결산과 수동 실행을 함께 제공하고, 원문 보존 기간과 연결 상태를 관리.

기술적 챌린지 · 대화와 명령의 구분

대화 내용을 실행 명령으로 받아들이지 않도록 AI의 도구 사용을 제한. AI가 연결되지 않으면 통계 요약을 제공하고, 원문 보존 기간도 따로 관리.

코드 · 구조 ↗

github.com/kgr0831/Abstraction-Bot

무인 스크린골프 · CMS와 자동정산

예약·기기 관제·결제·정산을 관리하는 웹 시스템

발견한 문제

예약·기기·결제 정보가 서로 분리된 운영

예약 상태와 기기 전원을 맞추고, 결제·환불 내역을 정산서까지 연결해야 하는 관리 업무.

결제 · 환불

거래 수집



정산 원장

거래별 기록



정산 · 대사

집계·차이 확인

정산 계산을 공통 도메인으로 모아

화면과 집계가 같은 원장을 사용하도록 설계.

구현 성과 · 운영과 정산의 통합

- 매장·기기·예약·회원 관리
- 예약과 기기 상태를 대조하는 작업 실행
- 중복 거래·환불·지연 도착 거래 처리
- 정산 확정·대사 차이·감사 기록 관리

Next.js 기반 CMS

운영 화면과 정산 집계가 같은 원장을 참조.

매장 관리·정산 기능 구현 및 시연. 실제 결제사·매장 기기 연결은 별도 범위.

기술적 챌린지 · 정산 금액의 근거 추적

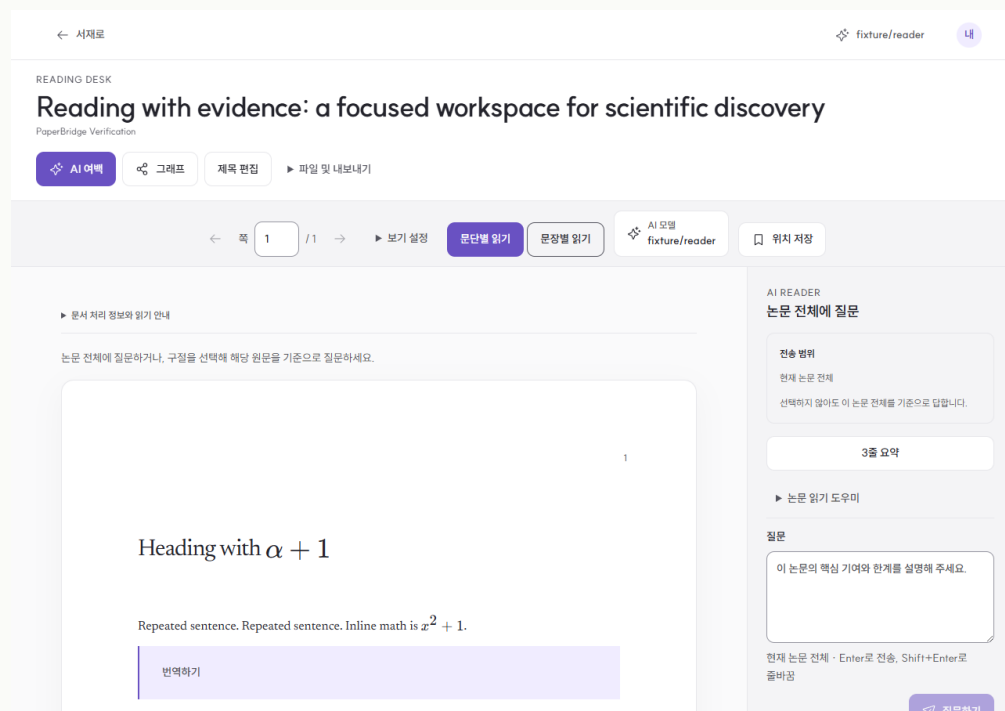
정산 합계만 수정하면 환불·중복·지연 거래가 금액을 바꾼 이유를 추적하기 어렵습니다.



발생한 상황	처리 방식	남기는 근거
환불	반대 부호의 원장 기록 추가	원거래와 환불 이력
중복 통지	거래번호로 중복 차단	같은 거래의 중복 반영 방지
마감 뒤 도착	확정 정산서는 유지하고 이월	다음 기간에 반영할 거래

매장 관리·정산 기능과 예외 처리를 구현·검증했습니다. 실제 결제사와 매장 기기의 연결은 별도 범위입니다.

PaperBridge · 논문을 읽고 근거를 남기는 공간



PaperBridge 리더 검증 화면 · 샘플 문서

발견한 문제

논문과 대화 도구 사이의 왕복

원문을 복사하고, 문맥을 다시 설명하고, 답변의 근거를 원문에서 찾는 반복.

해결 · 본인 기여

AI 연결 · 리더 상호작용 · 백엔드

AI 연결과 선택 문맥 전달, 질의 흐름을 리더 안에 구성. 인증과 연구 작업 공간을 뒷받침하는 백엔드 구현에 참여.

구현 성과 · 리더 안에서 질문과 근거 확인

구절 선택 → AI 질문 → 답변과 원문 확인

기술적 챌린지 · 리더와 AI 실행 환경 연결

외부 AI 도구로 이동할 때마다 논문의 문맥을 다시 설명해야 하는 불편을 줄였습니다.



기술적 챌린지

🔍 PC마다 다른 AI 도구의 설치 위치

지정 경로 → 시스템 검색 경로 → 기본 설치 위치 순서로 AI 도구를 찾고, 연결 실패 원인을 구분했습니다.

설계 판단

🛡️ 연결 확인과 질문 실행을 분리

AI 도구가 설치됐는지 먼저 확인하고, 사용자가 질문·문맥·모델을 검토한 뒤 실행하도록 구성했습니다.

✅ 본인 기여

AI 연결·리더 상호작용과 인증·연구 작업 공간 백엔드에 참여해, 질문과 근거 확인을 읽는 흐름 안에 연결했습니다.

EUM · AI 협업 워크스페이스



실시간 AI 번역 화상회의 · 서비스 화면

프로젝트 발표 youtube.com/watch?v=Pnjw9WitAXg

발견한 문제

회의와 그림을 함께 쓰면 화면이 느려짐

AI 번역 화상회의와 실시간 화이트보드를 함께 사용하는 협업 서비스.
동시 사용 중 화면 지연과 드로잉 품질 문제를 발견.

역할 · 5인 팀

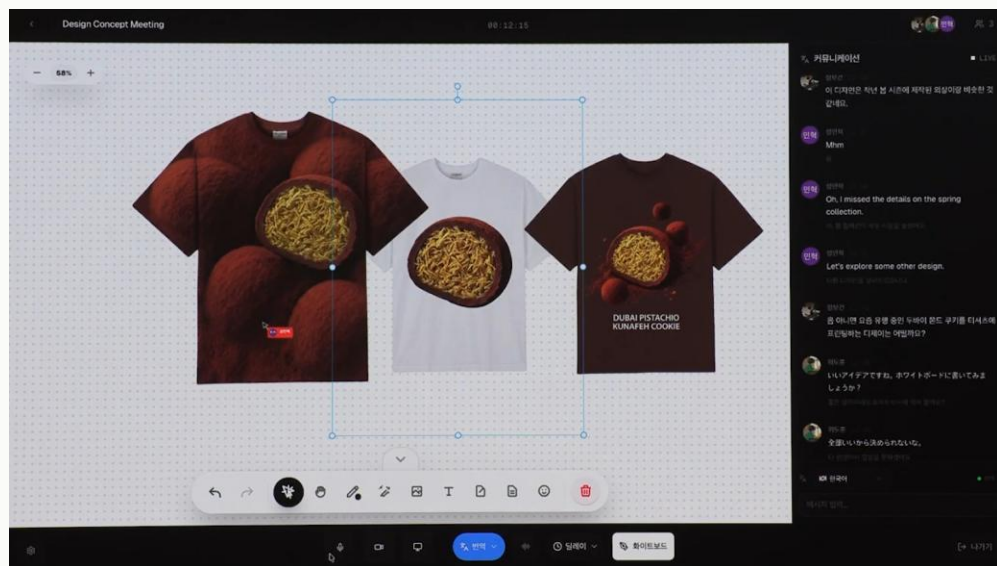
화이트보드 풀스택 개발

2025.12 – 2026.01. 드로잉 렌더러, 실시간 동기화, 포스트잇·권한 등 화이트보드 기능을 담당.

해결의 방향

렌더링 부하, 거친 선, 전송 지연의 원인을 나누고 각 문제에 맞는 처리 방식을 선택.

EUM · 화이트보드 렌더링 병목 해결



EUM · 실시간 협업 화이트보드

성과 · 평균 프레임

23.7 → 58.7 FPS

WebGPU · PixiJS v8 전환
프로젝트 당시 측정값

발견한 문제

회의 중 그림을 그리면 화면 반응이 느려짐

영상통화와 화이트보드가 같은 실행 흐름에 집중. 도형이 늘수록 CPU 부하가 커져 그리기와 화면 반응이 느려짐.

선택한 해결 수단

그래픽 연산을 GPU로 옮겨 병목을 줄임

CPU에 집중된 렌더링을 분산하기 위해 WebGPU 기반 PixiJS v8로 전환. 낯선 그래픽스 개념을 AI와 학습하고, 변경 근거를 팀에 설명.

5인 팀 · 화이트보드 담당 · 2025.12 – 2026.01

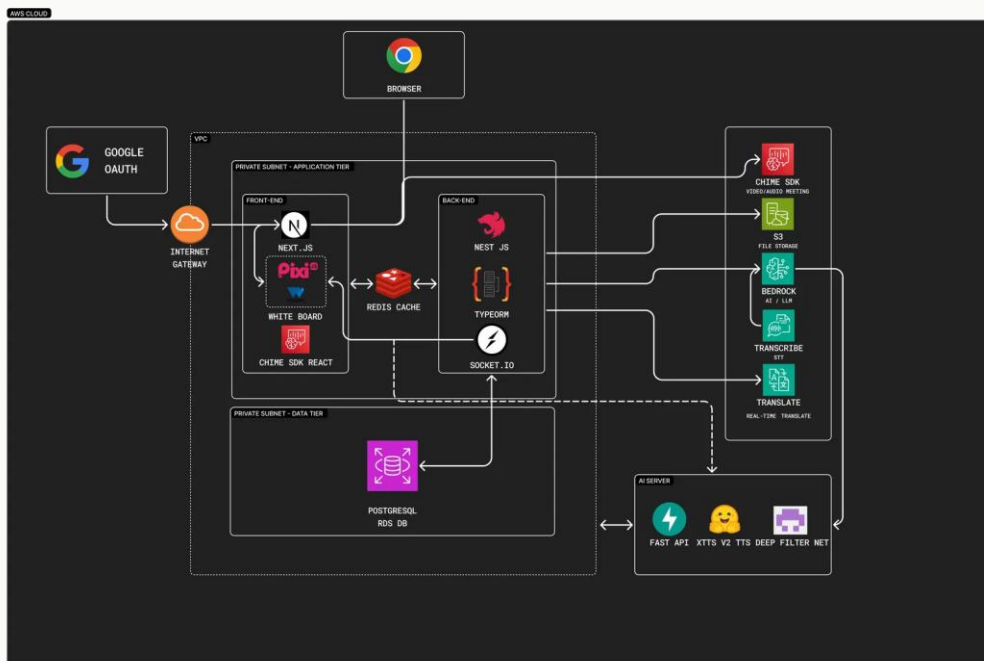
EUM · 드로잉 품질과 전송량 개선

입력·전송·표시에서 각각 개선한 문제

 One Euro Filter 미세한 떨림 보정	 Douglas-Peucker 핵심 좌표만 남김	 Bézier Curve 자연스러운 곡선 보간
사용자가 겪은 문제	적용한 방식	개선한 지점
거칠고 떨리는 선	속도에 따라 필터 강도를 조절	반응성을 유지하며 좌표의 흔들림 보정
큰 전송량과 동기화 지연	형태를 유지하는 핵심 좌표 추출	전송할 데이터 감소
빠르게 그릴 때 꺾이는 획	점 사이를 곡선으로 연결	끊겨 보이는 직선 구간 완화

선의 품질과 지연 문제에 맞춰 알고리즘을 선택했습니다. WebGPU 미지원 환경에서는 WebGL을 사용하며, 동시 편집은 나중 변경을 반영합니다.

EUM · 아키텍처 전환과 팀 설득



EUM 서비스 아키텍처 · 기존 프로젝트 자료

겪은 문제

구조 변경을 위한 팀의 합의

렌더러 교체는 일정과 기존 기능에 영향을 줌. 문제 원인과 변경 근거를 제안서·발표로 설명하고 팀과 방향을 맞춤.

트레이드오프

성능·호환성·동시 편집의 균형

WebGPU 미지원 환경은 WebGL로 처리. 좌표를 50ms 단위로 묶고 압축해 전송. 동시 편집 충돌은 나중 변경을 반영하며 자동 병합은 미구현.

기술을 선택한 기준

성능뿐 아니라 호환성과 팀의 도입 비용까지 고려.

PintOS · 가설의 오류를 찾아낸 과정

크라프톤 정글 · 4인 팀 · 스레드 스케줄링·사용자 프로그램·가상 메모리 구현

높은 우선순위 작업이 낮은 작업의 자원 해제를 기다리며 실행이 밀리는 ‘우선순위 역전’ 문제를 다뤘습니다.

가설 · Release-Trigger 실험

🔍 대기 관계를 관리하는 비용을 줄이자

스레드에 release_trigger 구조체를 두고, 스케줄링 시점에 잠금 해제와 우선순위 복구를 일괄 처리했습니다. 복잡한 기부 목록을 대신하려던 실험이었습니다.

새로 발견한 문제

🧵 기부는 한 번의 이벤트가 아닌 상태

다중·중첩 기부에서는 락마다 의존 관계가 계속 바뀌었습니다. 단일 트리거는 이를 놓쳤고, 잠금 해제도 스케줄링까지 미뤄 교착 상태와 우선순위 유실이 발생했습니다.

🔒 락별 기부 관계 추적

다중 소유·중첩 의존성 관리



🔑 해제 시 즉시 재계산

해제한 락의 기부만 제거



✅ 다중·중첩 기부 검증

경쟁 상태 해결·테스트 통과

✅ 배운 점 · 단순화가 놓친 조건을 찾기

표준 우선순위 스케줄링으로 리팩토링했습니다. 실패 원인을 추적하며 상태 추적과 잠금 해제 시점이 설계의 핵심임을 이해했습니다.

평성시 사회 시뮬레이션 · AI 연구

연구 질문

서로 다른 환경과 기억이 행동에 어떤 차이를 만들까?

직접 관찰이 어려운 평성시 사회를 가상 인물들의 상호작용으로 탐구. AI가 기억과 상황에 따라 행동하는 연구용 시뮬레이션 개발 참여.

기여 · 초기 시뮬레이션 라이브러리

에이전트·공간·월드의 공통 구조, 기억·성찰·계획 루프, 인터뷰와 이벤트·기억 내보내기 기능 구현.

기술 과제 · 에이전트의 기억·성찰·계획을 연결하고, 인터뷰와 이벤트 기록으로 행동의 맥락을 추적. 연구용 모형의 결과를 실제 사회의 예측으로 단정하지 않음.

연구 코드 · 비공개 🔒

github.com/kgr0831/SimulationProject

평성시 시뮬레이션

아동

부모

교원

관료

학교 · 시장 · 과외방 · 광장

📖 기억

사건 기록·검색



🔍 성찰

경험의 해석



🗺️ 계획

다음 행동

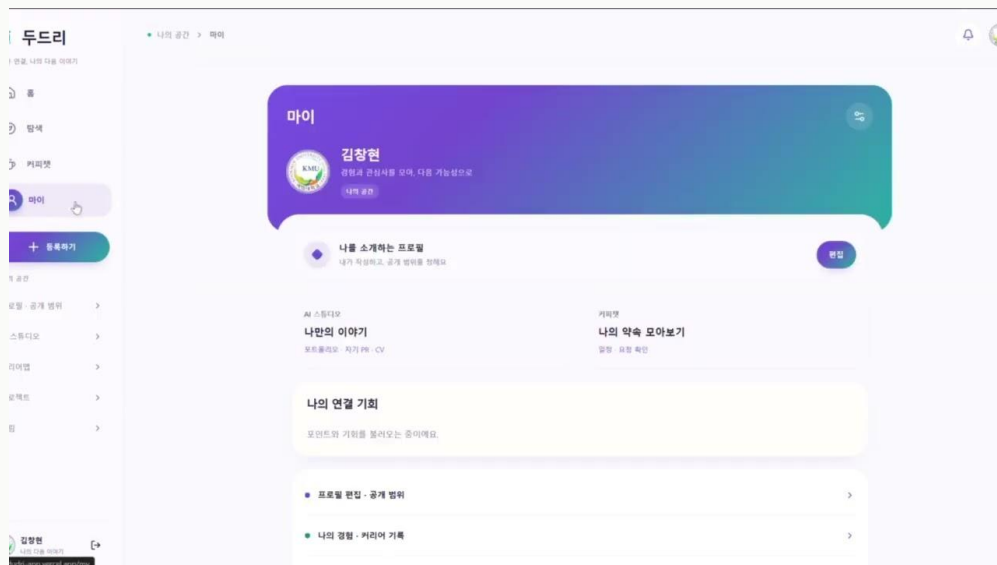
송신평화통일 연구원 · SmartAI Lab

학부연구생 · 2026.04 – 2026.08

에이전트 기반 모델링 · LLM



두드리 · 경험과 기회를 연결하는 플랫폼



두드리 소개 영상의 대표 이미지

서비스 소개 영상 [youtube.com/watch?v=vvsyQIOtiUQ](https://www.youtube.com/watch?v=vvsyQIOtiUQ)

프로젝트 코드 github.com/kgr0831/KOSSCCHTHON_2026

발견한 문제

쌓은 경험을 다음 기회로 연결하기 어려움

대학생의 경험 정리와 선후배 교류를 연결하는 성장 플랫폼. AI 자기소개 문서, 커피챗과 프로젝트 참여를 한곳에서 제공.

개발 경험

짧은 해커톤 안에서 기획과 구현 연결

AI와 명세 작성·개발·검증을 이어 여러 기능을 구현하는 워크플로우로 개발에 참여. 수상 이후에도 생성 데이터 일관성과 편집 기능을 개선.

성과

국민대·송실대·순천향대 2026 연합 경진대회 KOSSCCHTHON 대상(1위).



두드리 · AI Native 워크플로우와 대상 수상



2026 KOSSCCHTHON 대상 수상 사진

대상 · 1위

국민대 · 송실대 · 순천향대 2026 연합 경진대회 KOSSCCHTHON

짧은 해커톤 일정에서 기획을 서비스로 옮기기 위해, AI와 명세 작성·기능 구현·동작 검증을 연결했습니다.

📋 명세

작업 단위로 구체화



</> 개발

기능별 구현



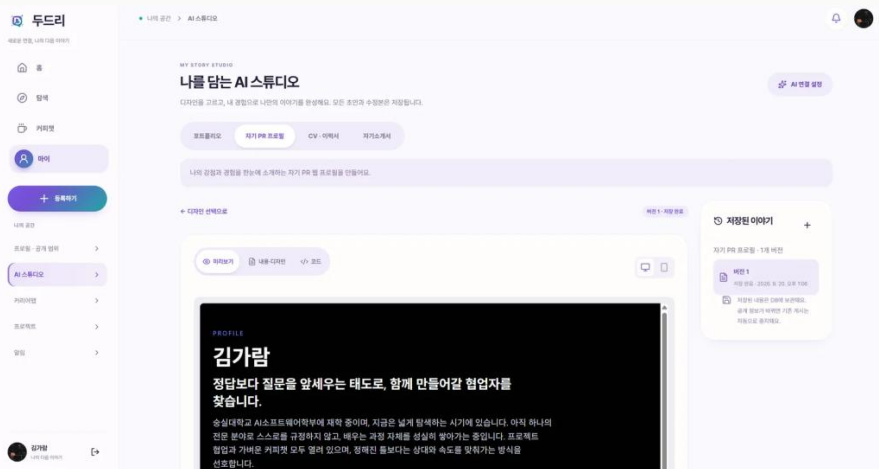
✅ 검증

동작 확인·수정

이후에도 생성 데이터의 일관성과 사용자 편집 기능을 개선했습니다.



두드리 · 경험을 문서와 프로젝트로 연결

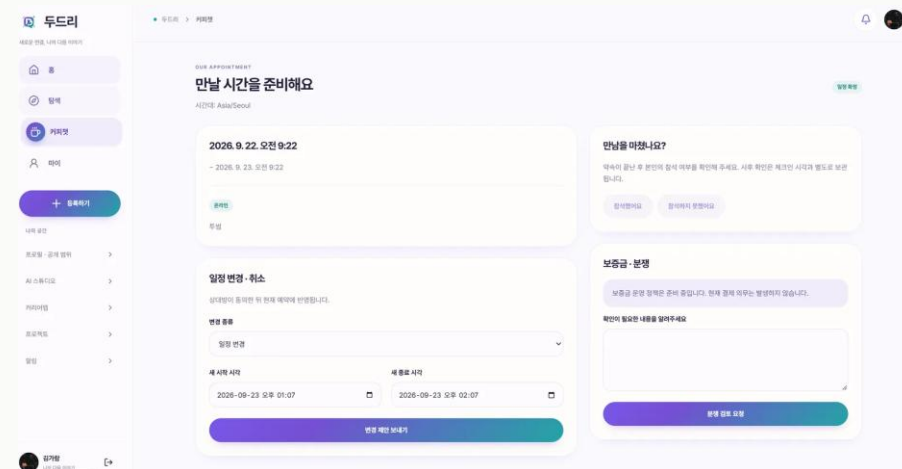


AI 스튜디오 · 경험 자료로 자기 PR 문서 제작

사용 흐름

경험을 선택하고 문서를 수정

자료 선택, AI 생성, 사용자 편집과 미리보기를 연결.



커피챗 일정 조율 · 선후배 경험 공유

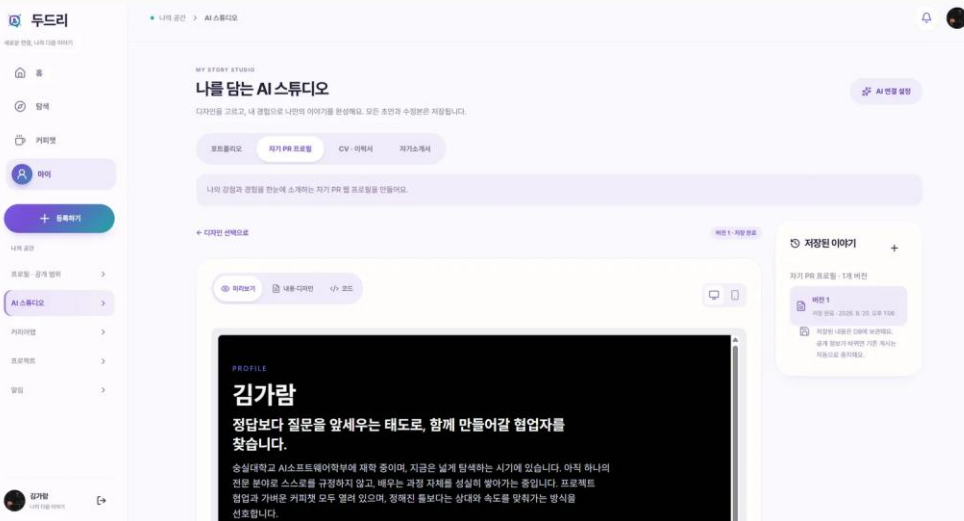
사용 흐름

커피챗으로 다음 경험을 연결

선후배와의 만남을 신청하고 시간을 조율하는 흐름.



두드리 · AI 생성 결과를 안전하게 편집



두드리 AI 스튜디오 · 후속 개발 포함

설계 원칙 · 사용자 통제권

사용자가 선택한 자료로 생성하고, 수정·게시 여부도 직접 결정. AI의 제안을 사용자가 검토할 수 있도록 설계.

문제 → 해결 · 생성 데이터의 일관성

생성 중 정보가 바뀌면 오래된 결과가 남음

생성 작업과 문서 버전을 분리. 저장 직전에 원본 변경 여부를 확인하고, 오래된 자료로 생성된 결과가 게시되지 않도록 차단.

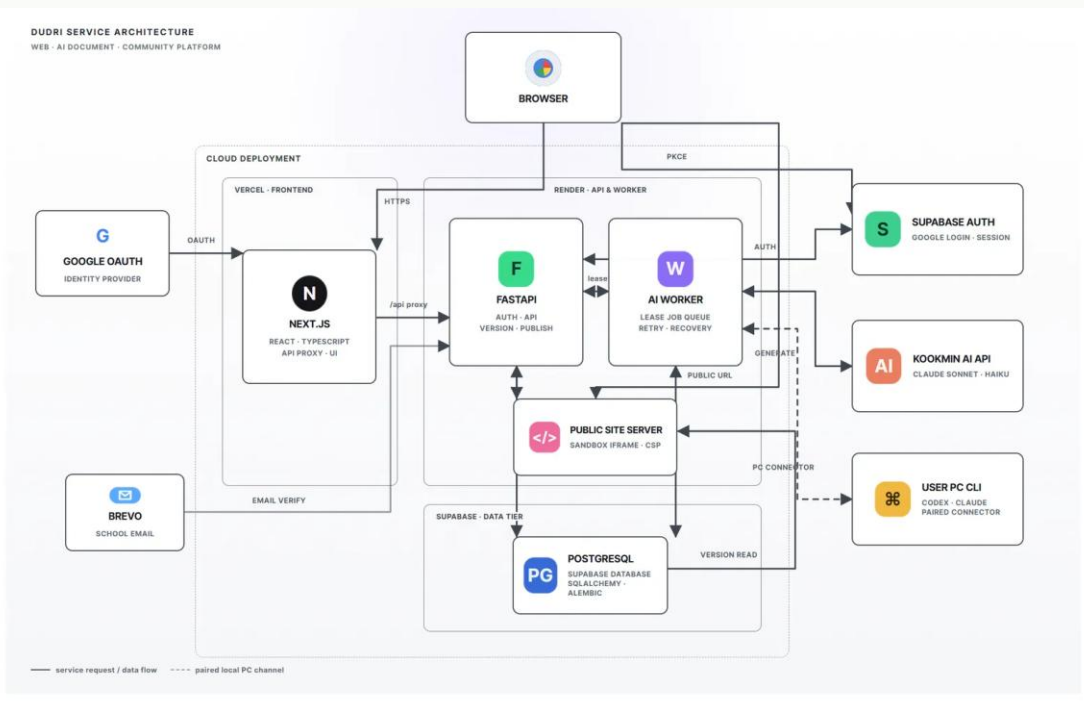
문제 → 해결 · 화면 편집과 코드 편집

화면에서 수정하면 직접 작성한 코드가 사라짐

편집 가능한 영역에 식별자를 붙여 바뀐 부분만 갱신. 사용자가 작성한 코드와 레이아웃을 보존하고 수정본은 새 버전으로 저장.

생성된 코드는 격리된 화면에서 실행하고, 접근 가능한 자원을 제한.

두드리 · 비동기 상태와 사용자 통제권



두드리 아키텍처 · 생성 작업과 게시 흐름

설계 경험

생성·수정·게시 상태를 분리

생성 중에도 자료 수정이나 취소가 가능하도록 상태를 분리. 백그라운드 작업에 실행 기한을 두고, 중단되면 다시 처리하도록 설계.

트레이드오프

편집 자유도와 안전한 공개

화면과 코드 양쪽에서 편집을 제공하되, 생성된 코드가 실행되는 영역과 접근 권한을 제한. 수정·게시 여부는 사용자가 결정.

남은 제약

무료 서버가 쉬고 있다가 깨어날 때 첫 응답이 느림. 작업 기록을 저장해 중단 후 복구할 수 있지만 최초 응답 지연은 남음.



게임을 만들며 제작 방식도 개선

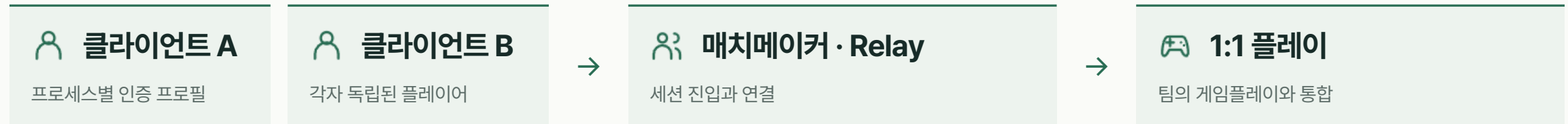
프로젝트	다룬 과제	워크플로우의 발전
Spell Thrower	게임잼 · 멀티플레이 · 매칭	작은 단위로 구현하고 팀과 합치는 경험
PlanetGame	은하·카메라·튜토리얼·연출	여러 시스템을 하나의 플레이 흐름으로 구성
Unitrio / Triniacle	전투 · 보스 AI · 제작 · 튜토리얼	기능을 나누어 개발하고 코드 검토 후 통합
SpriteAIMaker	캐릭터 · 방향별 애니메이션 · 시트	게임에 필요한 자산 제작을 도구로 연결
Remnants of Light / ROLAI	개발 요청 · Unity 실행 · 결과 확인	원격 요청부터 검증·보고까지 자동화

협업·시스템 구현·자산 제작·검증으로 넓어진 작업 범위.



Spell Thrower · 1:1 멀티플레이 게임

1:1 대전 테스트에서 두 게임 실행이 같은 인증 정보를 공유해, 서로 다른 플레이어로 접속할 수 없었습니다.



테스트 중 문제

같은 PC의 두 빌드가 인증을 공유

초기화 실패·취소 이후 재시도

구현한 처리

프로세스 ID로 익명 인증 프로필 분리

준비 작업을 다시 만들고, 나가기 시 세션 참조 정리

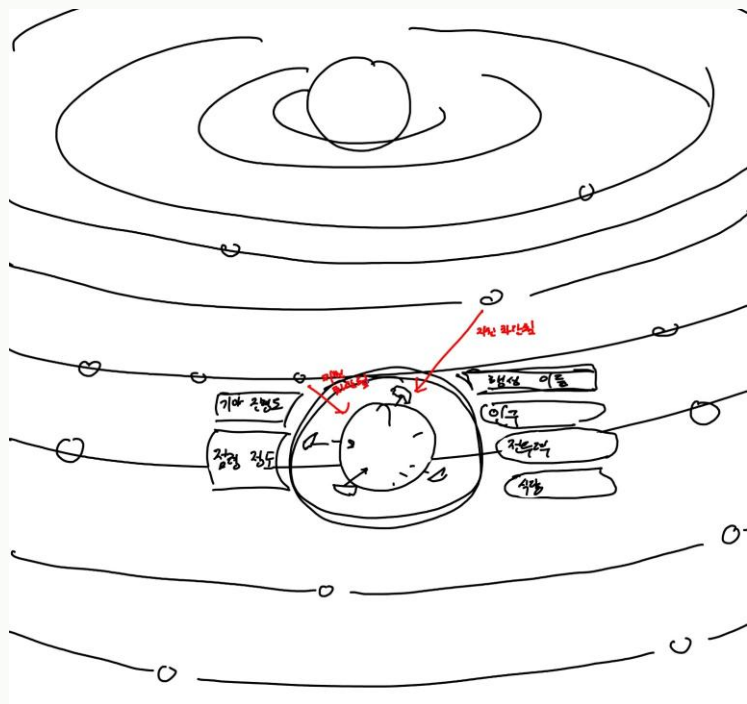
☑ 성과와 남은 과제

두 플레이어가 접속·매칭·대전하는 흐름을 팀의 게임에 연결했습니다. 시제품에서 피한 통신 모듈 충돌을 해결하고, 배포 전 암호화 통신을 복구해야 합니다.



PlanetGame · 우주 전략 게임의 플레이 흐름

은하를 탐색하고 자원·생산·점령·연구를 진행하는 Unity 3D 전략 게임



저장소에 보관된 초기 행성·UI 구상

발견한 문제

메뉴를 조작할 때 카메라도 함께 움직임

화면 조작과 카메라 입력이 충돌하고, 여러 시스템의 규칙을 처음부터 익혀야 하는 문제. UI 조작 중 카메라 입력을 차단.

본인 구현

은하 확장·카메라·튜토리얼·시네마틱

공간 탐색과 조작, 규칙 학습과 사건 연출을 연결. 튜토리얼은 한 번에 한 행동을 안내하고, 컷신 중에는 게임 진행을 멈춤.

구현 성과

복잡한 규칙을 조작 가능한 튜토리얼과 장면 전환으로 풀어내고, 건너뛰기·다시 보기까지 연결.

Unitrio / Triniacle · 팀 기반 2D 액션 RPG

5인 팀 · 팀 리더 · Unity



보스의 공격과 다음 행동이
서로 겹치는 문제.

전투와 탐험을 즐기는 2D 액션 RPG. 팀 리더로 기능을 나누어
개발하고, 담당한 전투·몬스터 행동·튜토리얼을 게임에 통합.

🔗 담당한 게임 기능

[코드](#) [기](#) **보스 행동 설계**
행동 트리로 공격과 추적 조건 구성

[코드](#) [기](#) **중립 몬스터 AI**
전투 대상의 반응과 행동 구현

[코드](#) [기](#) **전투 스킬 시스템**
플레이어의 행동 선택 확장

[코드](#) [기](#) **튜토리얼**
게임을 익히는 첫 흐름 구성

Triniacle · 보스 스킬의 실행 순서 정리

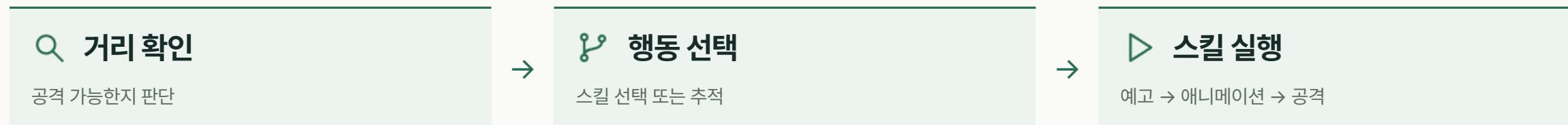
스킬 전에 쿨다운을 두자 짧은 스킬에서 다음 행동과 애니메이션이 겹치고 이동이 떨어졌습니다.

문제가 있던 순서

쿨다운 → 스킬 실행 → 다음 행동과 충돌

수정한 순서

예고 → 애니메이션·공격 → 쿨다운



✓ 본인 기여 · 팀 리더

재사용 대기 시간을 공격 뒤로 옮겨 동작이 겹치는 문제를 해결했습니다. 보스·몬스터 행동, 전투 스킬과 튜토리얼을 구현해 팀의 게임에 통합했습니다.

보스 행동 구현 ↗

github.com/kgr0831/UniTrio-Game-2026/pull/11

전투 스킬 구현 ↗

github.com/kgr0831/UniTrio-Game-2026/pull/17

SpriteAImaker · 게임용 AI 자산 제작 도구

AI가 만든 이미지를 게임에서 쓰려면 방향·움직임·프레임 크기·정렬을 함께 맞춰야 했습니다.



제작 방식의 변화

캐릭터와 애니메이션을 나눠 생성

기준 캐릭터를 확인한 뒤 움직임을 생성하고, 결과를 픽셀 편집·시트 조립으로 이어지게 했습니다.

기술적 챌린지

생성 이후의 품질을 도구에서 관리

중복 프레임, 위치 어긋남과 색상 손상을 수정하고, 방향별 생성이 끝나는 순서가 달라도 결과를 맞춰 조립했습니다.

구현 성과

AI 이미지 생성과 픽셀 편집을 연결해, 캐릭터 확인부터 프레임 가공·게임용 출력까지 한 도구에서 진행할 수 있게 했습니다.

SpriteAlMaker · 생성 결과를 엔진 규격으로

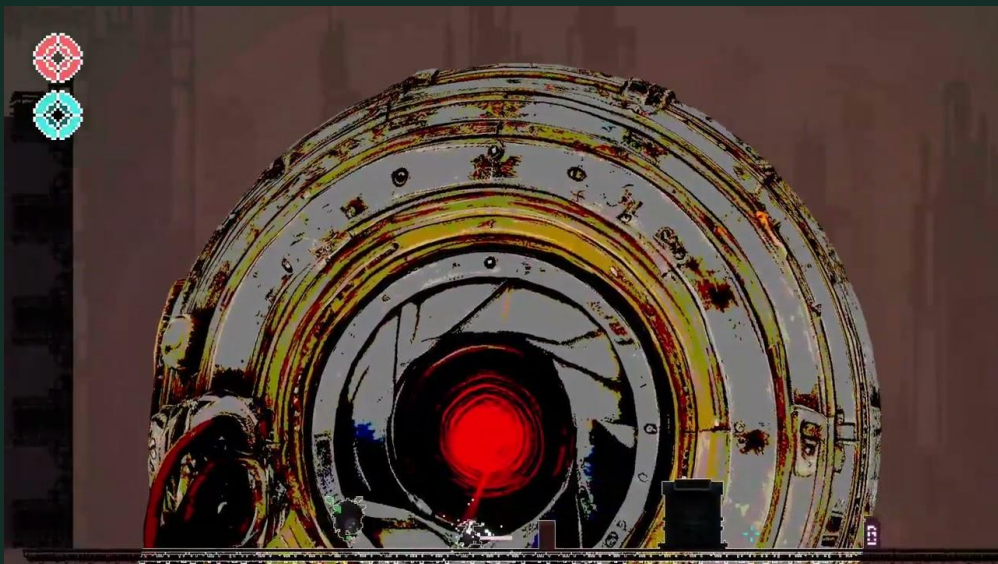
생성이 끝난 뒤에도 엔진 규격과 맞지 않는 결과가 나왔습니다. 출력 단계별로 원인을 나눠 수정했습니다.

출력 과정의 문제	수정한 처리	결과
첫 프레임 중복	중복 프레임 제거	요청한 프레임 수에 맞춤
프레임마다 다른 위치·크기	발밑 중앙을 기준으로 크기·위치 통일	동작의 기준점 정렬
색상 후처리로 픽셀 품질 손상	완성된 픽셀아트의 색상을 다시 줄이지 않음	불필요한 재처리 제거
방향·완료 시점의 불일치	방향별 생성 요청과 완료 상태를 따로 관리	4방향의 프레임을 한 시트로 조립

✔ 연결한 제작 흐름

캐릭터 확인 → 애니메이션 생성 → 프레임 가공 → Unity에서 프레임별 이미지로 분할. 생성 결과를 게임에 넣는 과정까지 연결했습니다.

Remnants of Light · 제작과 배포



Remnants of Light 플레이 영상의 대표 이미지

게임 다운로드 ↗ kgr0831.itch.io/remnants-of-light

플레이 영상 ↗ youtube.com/watch?v=WHP7L8P5Be8

프로젝트 코드 ↗

github.com/kgr0831/Remnants-of-Light

제작 중 발견한 문제

개발 PC 앞에서만 이어지는 작업

자리를 비워도 작업을 요청하고 결과를 확인할 필요가 있었음.
Discord와 AI·Unity를 연결한 게임 개발 자동화 도구 ROLAI를 구축.

프로젝트 · 1인 개발

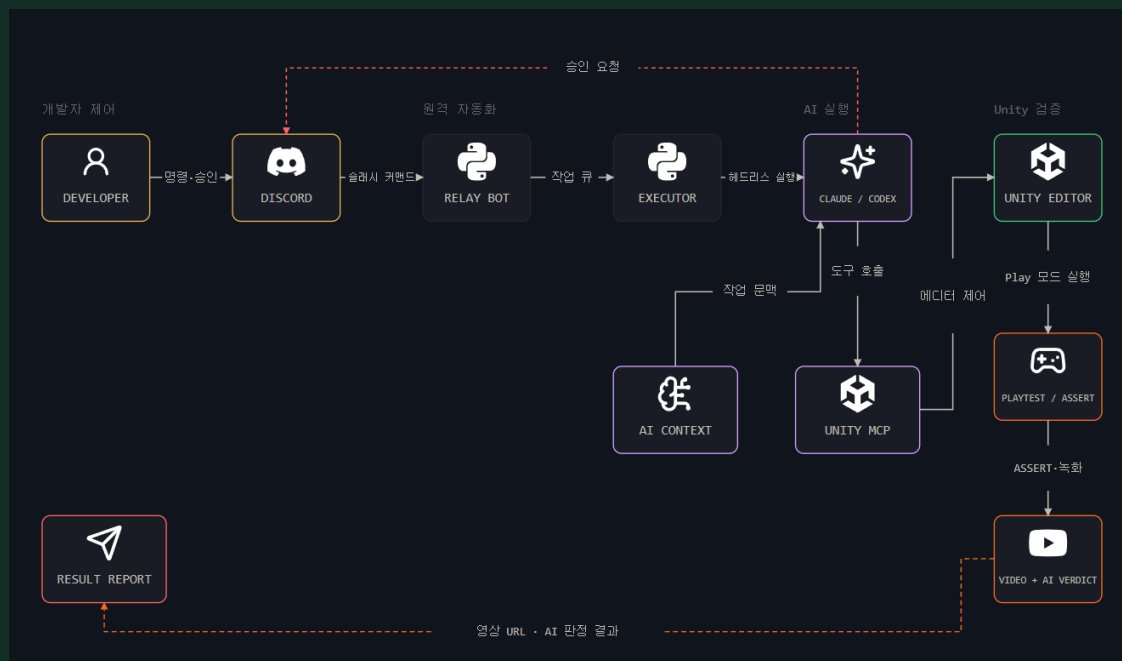
2D 액션 게임 · 첫 스테이지 데모

콤보·패링·벽 타기·대시로 방을 돌파하는 Unity 게임. 개발·기획·
아트를 맡아 Windows 데모를 itch.io에 배포.

다음 기술적 챌린지

원격으로 실행하는 데서 끝나지 않고, AI가 결과를 믿고 수정할 수
있는 검증 기준을 만드는 문제.

ROLAI · 원격 요청부터 실행과 보고까지



ROLAI의 요청·큐·실행·검증·보고 구조

작업 흐름

Discord 요청을 순차 실행

봇이 요청을 받은 순서대로 작업을 실행. AI가 Unity 편집기를 조작해 구현·실행하고, 결과를 Discord로 보고.

검증 흐름

수치 로그와 플레이 영상 대조

테스트 조건의 통과 여부를 로그로 확인하고 짧은 영상으로 실제 동작을 검토. 실패 원인에 맞춰 수정·재실행하도록 구성.

구현 성과

개발 PC 밖에서도 요청과 결과 확인을 이어갈 수 있는 원격 개발 루프 구축.

ROLAI · 같은 테스트가 매번 달라지는 문제

문제 · 조건에 따라 바뀌는 테스트 결과

같은 테스트가 실행할 때마다 달라짐

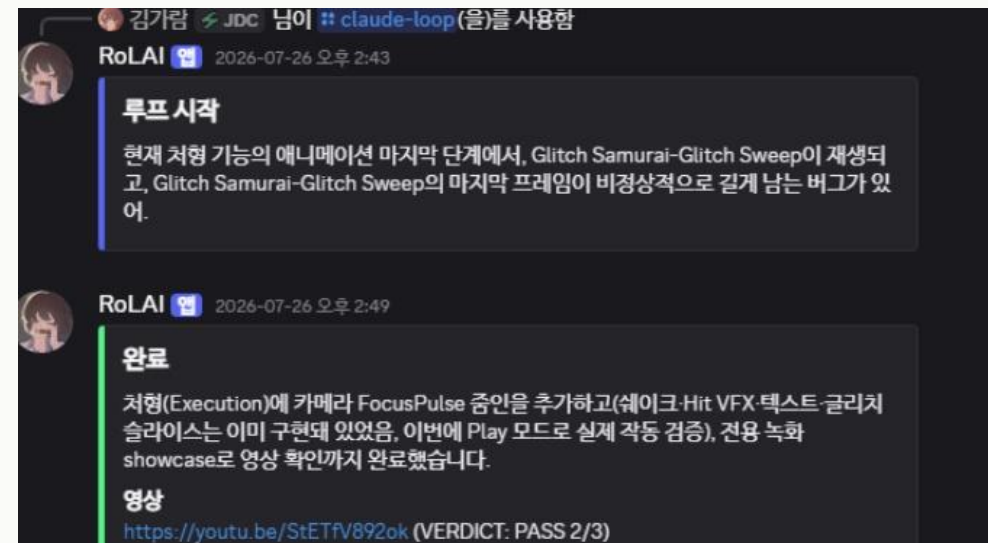
씬 상태·잔여 애니메이션·튜닝값의 영향으로 결과가 달라져, AI가 코드 결함과 테스트 환경 문제를 구분하기 어려움.

해결

📌 조건을 고정하고 실패 종류를 분리

별도 테스트 장면을 매번 새로 실행해 조건을 통일. 테스트 준비 실패와 기능 오류를 구분하고, 수정 전후의 이벤트와 수치를 대조.

트레이드오프 · 영상 검증은 정확도를 높이지만 실행 시간이 길어짐. 반복 작업은 로그로 확인하고 최종 검증에 짧은 영상을 사용.



Discord에서 작업 요청과 검증 결과를 보고받는 화면

성과 · 실패 원인을 구분하는 검증 흐름

통과 조건을 먼저 정의해 AI가 구현과 수정을 반복할 기준을 마련.

ROLAI · 자동화의 범위와 검증 비용



검증용 런타임 장면 · 기존 프로젝트 자료

배운 점

기능 설명만큼 통과 조건을 먼저 정의하는 일이 중요했음. 검증 가능한 요구사항이 반복 개발의 기준이 됨.

게임 다운로드 ↗

kgr0831.itch.io/remnants-of-light

프로젝트 코드 ↗

github.com/kgr0831/Remnants-of-Light

트레이드오프

AI 자율성과 복구 가능한 작업 범위

허용된 범위에서는 AI가 수정과 검증을 반복. 파일 삭제·변경 기록 저장·게임 장면 변경은 사용자 승인을 거쳐 잘못된 수정의 복구 범위를 관리.

트레이드오프

영상 검증의 신뢰성과 실행 시간

로그·영상·AI 판정을 함께 사용하면 확인 근거가 늘지만 실행 시간도 길어짐. 반복 확인은 빠른 로그 검증, 최종 확인은 짧은 영상으로 분리.

초기 자동화 구축 비용과 유지 부담은 남음. 검증 루프를 완전한 정답 보증으로 보지는 않음.

Virtual Friend · 캐릭터 대화 서비스 구상

WIP · 개인 프로젝트 · 사업화 구상

풀고 싶은 문제 · 대화가 이어져도 같은 캐릭터로 느낄 수 있을까?

목소리·외형·기억이 일관된 캐릭터 대화 서비스를 개발 중입니다. 자연스러운 응답과 대화의 연속성을 함께 다룹니다.

사업화 가설

캐릭터의 일관성과 대화 품질이 다시 찾을 이유가 될 수 있음. 어떤 경험을 필요로 하는지, 어떻게 유입되고 유료로 사용할 이유는 무엇인지 함께 검증할 계획.

검증할 항목: 반복 사용 · 선호 캐릭터 · 음성 경험 · 운영 비용과 지불 의향



현재 프로토타입에 포함된 캐릭터 일러스트

개발과 함께 **사용자가 찾을 이유와 사업으로 이어질 가능성을 고민.**

Virtual Friend · 구현과 다음 기술적 챌린지

영역	진행 상태	구현 · 기술 과제
대화 · 기억	프로토타입 구현	캐릭터별 대화·기억, 사진 이해와 앨범, 기록 조회·수정
음성 메시지	프로토타입 구현	캐릭터 목소리 연결, 감정·연기 태그, 음원 생성·재시도
실시간 음성 대화	개발 과제	말을 듣고 음성으로 답하는 STS 처리의 지연 감소 자연스러운 말 순서 전환과 사용자의 중간 끊기 지원
캐릭터 이미지 학습	연구·개발 과제	로컬 모델에 LoRA 추가 학습을 적용해 그림체·외형 유지 상황이 바뀌어도 같은 캐릭터로 보이는지 평가할 기준

빠른 응답, 캐릭터의 일관성, 운영 비용을
같은 서비스 경험 안에서 풀어야 하는 문제로 보고 있습니다.

문제를 찾고, 해결할 수단을 선택합니다.

문제를 찾아내는 것이 제 역할이고,
기술은 그 문제를 해결하기 위한 수단입니다.

김가람

kimgalam0831@gmail.com

github.com/kgr0831

kgr0831.github.io