

2025 오픈소스 개발자대회 결과보고서\_자유,지정과제

구 분	세 부 내 용		
팀 명	사냥단	팀 인 원 (팀장 포함)	4
부 문	학생	과제 유형	자유

□ 결과보고서

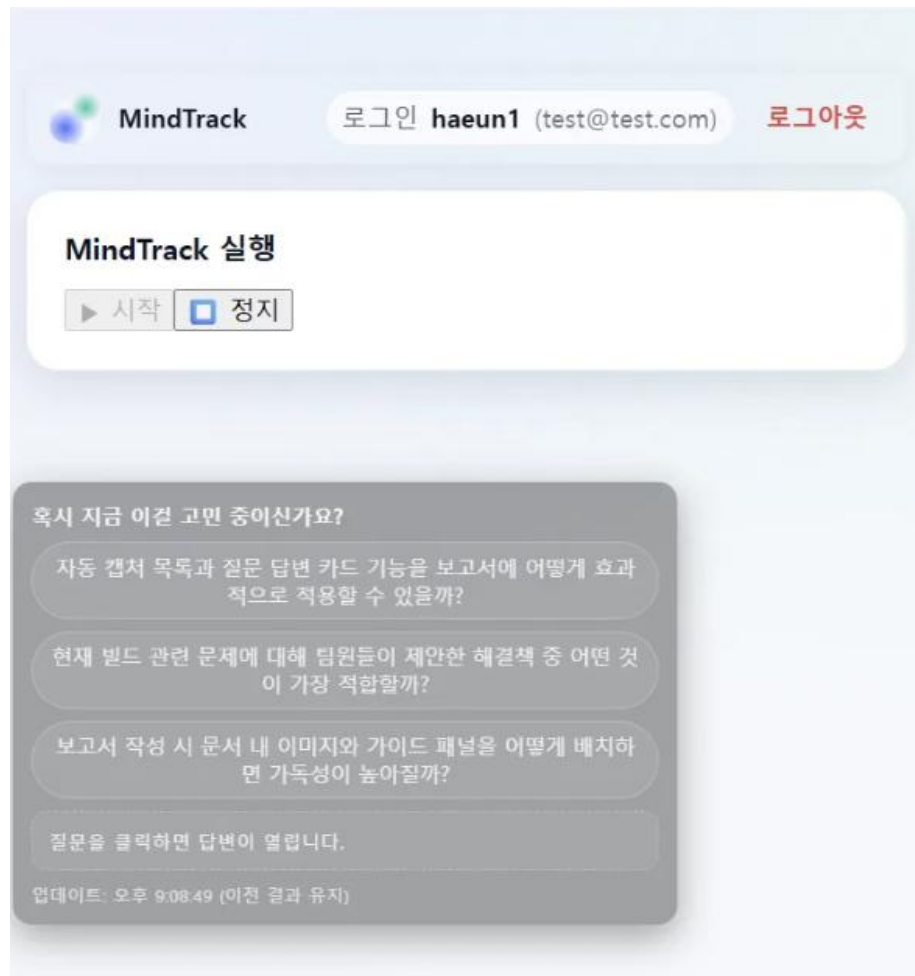
프로젝트명	능동적 화면 맥락 분석 기반 예측형 AI 에이전트
프로젝트 등록	<a href="https://github.com/4-nyang-dan">https://github.com/4-nyang-dan</a>
시연영상	<a href="https://youtu.be/lA4hnCcSAb0">https://youtu.be/lA4hnCcSAb0</a>
프로젝트 소개	단순히 묻고 답하는 수준을 넘어, 사용자의 사고 흐름을 따라가며 다음 단계를 준비하는 능동형 맥락 기반 예측 AI 에이전트
프로젝트 세부 내용	
개발배경 및 목적	<p>오늘날 GPT, Claude, Gemini, Grok과 같은 최신 대규모 언어 모델(LLM) 기반 에이전트들은 빠르게 발전했지만, 여전히 “사용자가 질문해야만 답한다”는 반응형 구조에 머물러 있다. 이는 실제 인간의 사고 흐름과 괴리가 크다. 인간은 단순히 질문-응답의 단절적 과정이 아니라, 맥락적 사고를 통해 다음 단계를 예측하고 준비한다. 그러나 기존 AI는 사용자의 입력 없이는 사고를 이어가지 못하며, 후속 문제를 선제적으로 제시하지 못하는 한계를 가진다.</p> <p>이러한 한계를 극복하기 위해 본 프로젝트는 능동적 예측형 AI 에이전트를 개발하고자 한다. 본 시스템은 사용자의 화면을 기반으로 과거 히스토리와 현재 맥락을 벡터화하여 저장한 뒤, 향후 발생할 행동과 잠재적 질문을 미리 예측하고, 나아가 답변까지 선제적으로 제안한다. 즉, 단순 반응형 AI를 넘어, 스스로 사고를 이어가고 사용자를 한 발 앞서 안내하는 미래지향적 에이전트 구현을 본 프로젝트의 목적으로 삼는다.</p>


<div>개발 환경</div> <div>(언어, Tool, 시스템 등)</div>	<ul style="list-style-type: none"> <li>- 프론트엔드: Electron, React</li> <li>- 백엔드: Spring Boot(Java), Redis, PostgreSQL</li> <li>- AI 서버: FastAPI(Python), Tesseract OCR, Microsoft Presidio(PII), OpenAI API(GPT-4.1-mini, GPT-5-mini, Openai-Embedding-Small), FAISS(Vector DB)</li> <li>- 배포/운영: Docker, (옵션) AWS S3</li> <li>- 기타: SSE(Server-Sent Events), PostgreSQL LISTEN/NOTIFY</li> </ul>
<div>시스템 구성 및 아키텍처</div>	<div> <p>The diagram illustrates the system architecture. At the top, the <b>User PC(Local)</b> contains an <b>Electron Agent</b> (with icons for periodic screen capture, JWT token and image transfer, and a 1st step: similarity judgment using pHash and SSIM) and an <b>Electron, React UI</b> (with icons for question/answer UI and login/environment settings). The User PC communicates with the <b>Back Server</b> via <b>REST API HTTP + JWT 인증</b>. The <b>Back Server</b> (containing Spring and Redis) handles <b>이미지 해시 계산</b> (image hash calculation), <b>2차 : SSIM 기반 유사도 판별</b> (2nd step: SSIM-based similarity judgment), <b>유사한 이미지 : 방문도수 증가</b> (similar images: increase visit count), <b>비유사 이미지 : 이미지 등록</b> (dissimilar images: image registration), and <b>사용자 최근 이미지 캐시 관리</b> (user recent image cache management). The Back Server sends <b>AI 처리 데이터 전송</b> (AI processing data transfer) to the <b>AI Server</b>. The <b>AI Server</b> (containing CNN, OCR+PII, and embedding services) performs <b>CNN, 클러스터링 대표 이미지 추출</b> (CNN, clustering representative image extraction), <b>OCR + PII 개인정보 제거</b> (OCR + PII personal information removal), <b>이미지 설명 추출</b> (image description extraction), <b>텍스트 임베딩 추출</b> (text embedding extraction), <b>임베딩 저장 및 유사도 검색 수행</b> (embedding storage and similarity search), <b>RAG 기반 질문 예측 및 응답 생성</b> (RAG-based question prediction and response generation), <b>질문/답변 및 마스킹 이미지 반환</b> (question/answer and masked image return), and <b>마스킹 이미지 업로드</b> (masked image upload). The AI Server also performs <b>메타데이터 업로드 (결과/설명, S3 URL)</b> (metadata upload (result/description, S3 URL)) and <b>메타데이터 조회 사용자 인증 및 저장</b> (metadata query user authentication and storage). The AI Server returns <b>결과 반환</b> (result return) to the Back Server.</p> </div> <p>위 구조는 화면 캡처 기반 맥락 인식 및 예측 파이프라인으로, <b>로컬 단말</b>에서는 SSIM과 pHash를 통해 변화가 감지된 이미지만 선별하여 전송한다. 또한, <b>백엔드 (Spring)</b>에서는 다단계 유사도 판별과 메타데이터 관리를 수행하며 이후 <b>AI 서버</b>는 CNN과 OCR, 임베딩 및 RAG 기반 처리를 통해 설명 생성과 미래 행동, 질문 예측을 수행하며, 결과는 마스킹 이미지와 함께 프론트엔드에 실시간으로 제공된다.</p>

<p><b>프로젝트 주요기능 및 구조도</b></p>	<p>본 프로젝트는 화면 캡처를 기반으로 사용자의 맥락을 인식하고 미래 행동을 예측하는 AI 에이전트 시스템이며 <b>핵심 기능</b>은 다음과 같다. <b>첫째</b>, 본 시스템은 <b>사용자의 작업 흐름을 기반으로 미래 행동을 예측</b>한다. 단순히 현재 화면만 해석하는 것이 아니라, 과거 히스토리와 맥락을 함께 분석하여 다음 단계에서 필요할 가능성이 높은 행동을 선제적으로 제안한다. <b>둘째</b>, 사용자가 <b>아직 질문하지 않은 사안에 대해서도 선제적으로 질문과 답변을 제공</b>한다. 이를 통해 사용자는 뒤따라올 문제를 미리 대비할 수 있으며, AI는 단순한 반응형 도구를 넘어 능동적으로 사고를 이어가는 동반자의 역할을 수행한다. <b>셋째</b>, 사용자가 <b>직접 입력한 질문에 대해서도 과거 맥락을 반영한 응답</b>을 제공한다. 이는 단편적인 답변이 아니라 연속적이고 심화된 사고 흐름을 지원하는 방식으로, 기존 AI 서비스와 차별화되는 특징이다.</p> <p>본 프로젝트의 개발은 <b>Top-Down 방식</b>으로 접근하였다. 먼저 전체 시스템의 목표와 구조를 정의한 뒤, 이를 구체적인 구성 요소인 프론트엔드, 백엔드, AI 서버 단계로 나누어 각 역할에 맞는 기능과 목표를 두고 개발하였다.</p> <p><b>첫째, 프론트엔드 단계</b>에서는 사용자 화면을 효율적으로 캡처하고 의미 있는 데이터만을 전송하기 위해 노력하였다. Electron과 React를 기반으로 주기적 캡처 기능을 구현하였으며, SSIM과 해시 기법을 활용하여 불필요한 중복 이미지를 제거하였다.</p> <p><b>둘째, 백엔드 단계</b>에서는 안정적이고 효율적인 데이터 관리에 초점을 맞추었다. Spring, Redis, PostgreSQL을 연동하여 다단계 유사도 검증을 수행하였고, 이를 통해 저장 공간의 낭비를 줄이며 분석 요청이 원활히 처리되도록 하였다.</p> <p><b>셋째, AI 서버 단계</b>에서는 화면 맥락 이해와 예측 기능의 구현에 주력하였다. FastAPI 기반의 분석 엔진을 구축하고 OCR, 임베딩, RAG 기법을 적용하여 화면 속 정보를</p>
---------------------------------------	--

해석한다. 그 결과 사용자의 미래 행동을 예측하고, 잠재적 질문과 답변을 선제적으로 제안할 수 있는 기반을 마련하였다.

아래는 실제 사용자의 화면을 기반으로 예측한 행동을 기반으로 생성된 질문이다.



 MindTrack

로그인

로그아웃

haeun1 (test@test.com)

MindTrack 실행

혹시 지금 이걸 고민 중이신가요?

Electron 창을 투명하게 만들 때 성능 저하나 호환성 문제는 어떻게 해결할 수 있나요?

React에서 투명 배경 컴포넌트가 다른 UI 요소와 겹칠 때 발생하는 문제를 어떻게 방지할 수 있나요?

MindTrack 분석 결과가 예상과 다른 경우, 어떤 디버깅 절차를 통해 문제 원인을 파악할 수 있나요?

...


5) 접근성 및 포커스 관리

- 투명하지만 시각적으로 보이지 않는 요소가 키보드 포커스를 차단하지 않도록 tabIndex 및 aria-hidden을 적절히 설정하세요.

- 예: 투명 레이어가 interactive하지 않으면 aria-hidden="true" 또는 tabIndex=-1로 비활성화.

6) 디버깅 팁

업데이트: 오전 11:06:16

 MindTrack

로그인

로그아웃

haeun1 (test@test.com)

MindTrack 실행

혹시 지금 이걸 고민 중이신가요?

ipcMain 핸들러 함수가 Renderer 프로세스에서 제대로 호출되는지 어떻게 확인할 수 있나요?

BrowserWindow의 투명 배경과 항상 위에 표시 설정이 충돌할 때 해결 방법은 무엇인가요?

Electron과 React 환경에서 투명 배경 컴포넌트를 구현할 때 성능 최적화 팁이 있나요?

질문을 클릭하면 답변이 열립니다.

업데이트: 오전 10:46:34 (이전 결과 유지)

<p>기대효과 및 활용분야</p>	<p>본 프로젝트의 AI를 테스트 해본 결과, 100개의 질문을 대상으로 비교 실험을 수행하였다. 기존 AI 모델은 단순히 질문이 주어졌을 때, 약 53%의 질문에만 적절히 답변하였으나, 본 소프트웨어를 적용했을 경우 사용자가 원하는 방향으로 답변한 비율이 71%로 크게 향상되었다. 특히 코드 구성상의 오류 지적, 환경 변수 설정 문제 해결, 실행 중 발생하는 에러 현상 분석 등과 같이 실제 개발 과정에서 자주 발생하는 질문에 대해 더 정밀하고 맥락적인 답변을 제공할 수 있었다.</p> <p>- 구현된 핵심 기능</p> <p>또한 구현된 기능은 크게 세 가지로 요약된다. 첫째, 사용자의 미래 행동을 예측하는 기능, 둘째, 앞으로 발생할 수 있는 질문을 선제적으로 제시하는 기능, 셋째, 과거와 현재 맥락을 반영하여 기존보다 나은 답변을 제공하는 기능이다.</p> <p>- 향후 개선 방향 및 확장 가능성</p> <p>본 프로젝트는 초기 단계에서 의미 있는 성과를 보여주었지만, 여전히 개선이 필요한 영역이 존재한다. 첫째, 화면 기반 분석 과정에서 <b>토큰 수가 과도하게 증가하는 문제</b>가 있어 이를 최소화하는 최적화가 요구된다. 둘째, 분석 및 응답 생성 과정에서 <b>일정 수준의 지연 시간이 발생</b>하므로, 처리 효율을 높이는 시스템 최적화 작업이 필요하다. 셋째, <b>MCP 및 다양한 에이전트들과의 연동</b>을 통해 더 고도화된 질의응답을 제공할 수 있도록 확장할 계획이다. 또한 현재 캐시 관리 및 시스템적 병목 현상으로 인한 한계가 일부 발견되었으며, 이러한 문제를 지속적으로 해결해 나갈 예정이다.</p>
----------------------------	--

	<p>나아가 본 프로젝트의 <b>확장 가능성</b>은 다음과 같다. 첫째, 질문에 대한 답변을 생성할 때 <b>MCP를 활용하여 더욱 정교한 질의응답 체계를</b> 마련할 수 있다. 둘째, <b>강화학습(RLHF 등)과 접목하여 사용자 맥락에 최적화된 개인화 AI</b>를 구현할 수 있다. 셋째, 현재는 데스크톱 화면을 기반으로 맥락을 추적하지만, 이를 <b>실제 세계의 데이터를 입력값으로 확장</b>할 경우 진정한 AI 비서로 진화할 수 있다. 또한, 스마트 팩토리나 스마트 시티와 같은 복잡한 환경에서 본 프로젝트의 구조를 적용하면, AI가 스스로 맥락적 메모리를 구성하고 상황에 맞는 예측과 지원을 제공할 수 있다.</p>
<p><b>기타</b> (프로젝트 추가 설명)</p>	<p>- <b>프로젝트의 혁신성 및 차별성</b></p> <p>이번 프로젝트의 차별성은 단순히 “질문에 답하는 AI”가 아니라, <b>앞으로 사용자가 무엇을 할지 먼저 생각하고 제안하는 AI</b>를 구현했다는 점이다. 화면 캡처와 맥락 분석을 통해 사용자의 흐름을 이해하고, 다음 행동과 질문을 예측하는 구조는 기존 챗봇이나 LLM 시스템에서 보기 어려운 부분이다. 한마디로, <b>수동적 응답에서 능동적 사고로 전환</b>한 것이 가장 큰 혁신이다.</p> <p>앞으로는 토큰 사용량을 줄이고 속도를 높여 더 매끄럽게 동작하도록 다듬을 예정이다. 또, MCP 같은 다양한 에이전트와 연동해 더 풍부한 예측을 보여줄 수 있도록 확장할 계획이다. 장기적으로는 단순히 화면이 아니라, 현실 세계의 데이터까지 받아서 예측할 수 있는 진짜 “AI 비서”로 발전시키는 것을 목표로 하고 있다.</p> <p><b>세부 내용이 들어간 보고서도 따로 첨부하였습니다.</b></p>

※ ‘회색 안내 문구’는 삭제하고 검정색 글씨로 자유롭게 작성