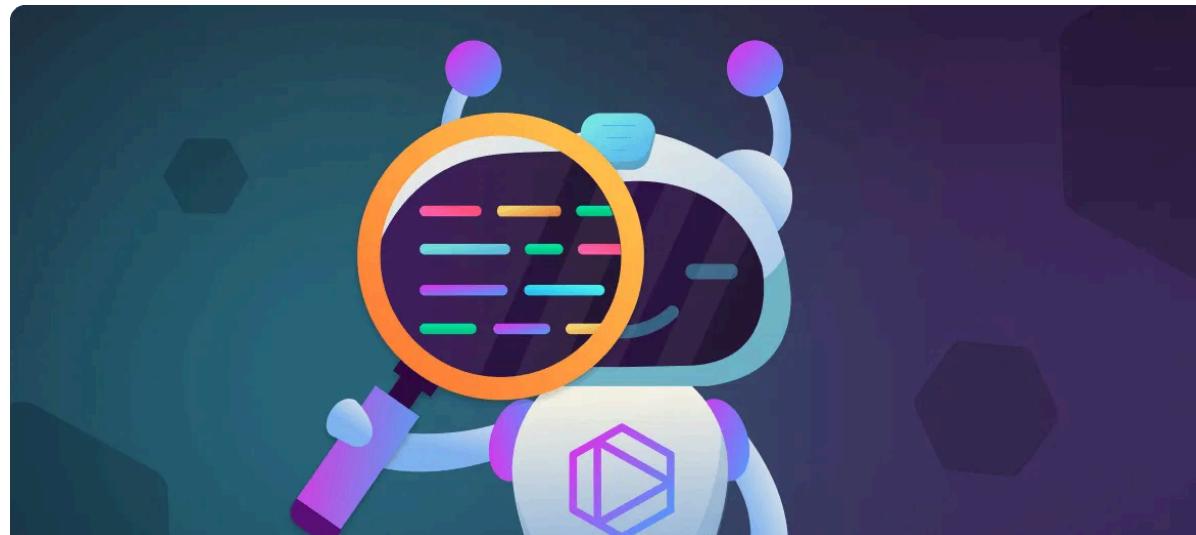


AI 페어 프로그래머

실시간 AI 코드 관리 시스템



프로젝트 개요 및 비전

프로젝트 목표

개발자가 코드를 작성하는 동안 실시간으로 지능적인 피드백과 제안을 제공하는 "AI 페어 프로그래머"를 개발하는 것을 목표로 합니다.

핵심 비전

개발자의 작업 흐름을 방해하지 않으면서도 코드 품질 향상, 잠재적 오류 사전 방지, 코드와 문서의 동기화 부담 제거를 통해 개발 생산성을 극대화하는 '지능형 코딩 동반자'를 구현하는 것입니다.



핵심 기능 (1)

API 일관성 실시간 검사

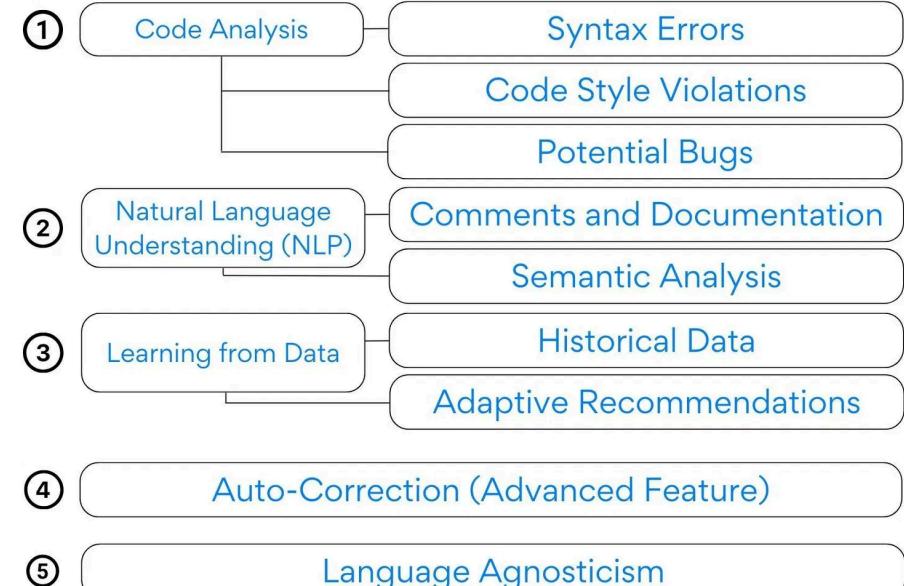
함수나 메소드의 시그니처(매개변수, 반환 타입 등) 변경 시, 해당 함수를 호출하는 모든 코드의 일관성을 실시간으로 분석하고 불일치 항목을 즉시 경고합니다.

코드 변경에 따른 실시간 문서 자동 업데이트

코드의 구조적 변경(매개변수 이름 변경, 타입 추가, 반환값 수정 등)이 발생했을 때, 이와 관련된 주석(Docstring, Javadoc 등)을 AI가 자동으로 수정하고 업데이트합니다.

- 개발자는 더 이상 코드와 문서의 동기화를 위해 수동으로 작업할 필요가 없습니다.
- 문서는 항상 코드와 완벽하게 일치된 최신 상태를 유지합니다.

How do AI code reviewers work?



핵심 기능 (2)

AI 기반 유사 코드 및 모범 사례 추천

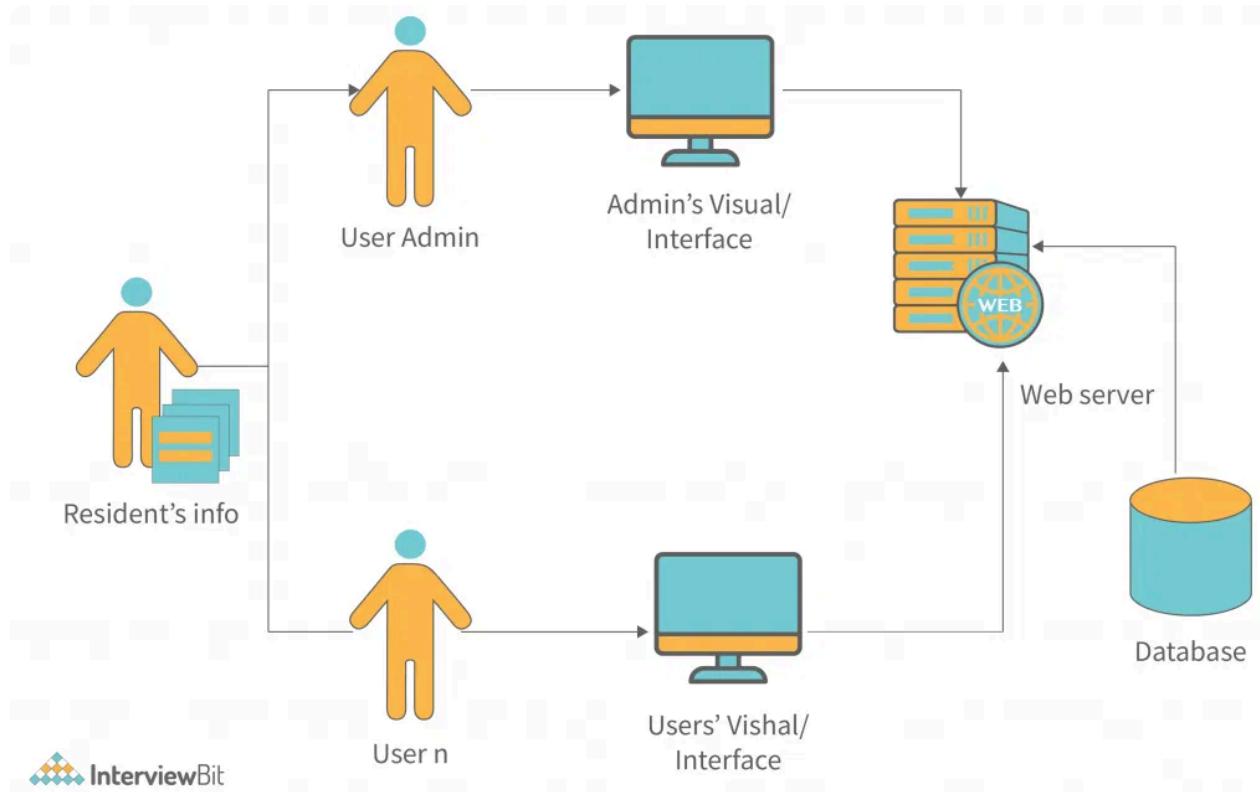
개발자가 작성한 코드의 의도와 구조를 AI가 분석하여, 전 세계 고품질 오픈소스 코드 기반의 유사 기능 구현 사례나 더 효율적인 코드 패턴을 추천합니다. 이를 통해 코드의 품질을 상향 평준화하고, 새로운 기술 습득을 돋습니다.

직관적인 IDE 통합 경험

분석 및 추천 결과를 VS Code, IntelliJ 등 주요 IDE(통합 개발 환경)에 직접 표시합니다. 오류가 의심되는 코드에는 밑줄을 긋고, 마우스를 올리면 상세 설명을 툴팁으로 보여주며, 추천 코드는 별도의 패널에 제시하여 개발자가 자연스럽게 정보를 인지하고 활용할 수 있도록 합니다.



시스템 아키텍처



실시간 코드 감지 엔진

지정된 프로젝트 폴더 내의 코드 파일 변경을 실시간으로 감지하여 분석 시스템에 신호를 전달합니다.

지능형 코드 분석기

변경 코드를 전달받아 구문 분석, AI 모델 추론 등을 통해 문제점을 진단하고 개선안을 도출하는 시스템의 두뇌입니다.

IDE 통합 인터페이스

분석 결과를 개발자가 사용하는 IDE에 시각적으로 표현하는 사용자 접점입니다.

기술 구현 방안 (1)

실시간 코드 감지 엔진 (File System Watcher)

지정된 워크스페이스 내에서 타겟 파일의 생성, 수정, 삭제 이벤트를 감지하여 분석 엔진에 전달합니다.

주요 기술: Python watchdog 라이브러리

지능형 코드 분석기 (Intelligent Analyzer)

실제 코드 분석 로직이 수행되는 핵심 두뇌로, AST 파싱, 규칙 기반 분석, AI 모델 추론 등을 담당합니다.

주요 기술:

- AST(추상 구문 트리) 파싱 라이브러리 (예: tree-sitter)
- 코드 생성 및 수정 모듈 (주석 부분 수정 및 재생성 포함)
- 코드 임베딩 AI 모델 (예: CodeBERT, GraphCodeBERT)
- 고속 벡터 검색 엔진 (예: FAISS, ScaNN)

아키텍처 노트

IDE 확장 프로그램(클라이언트)과 분석 엔진(서버)은 **LSP(Language Server Protocol)**를 통해 통신하여 시스템을 IDE 독립적으로 만듭니다.

AI-Powered Code Documentation And Analysis



기술 구현 방안 (2)

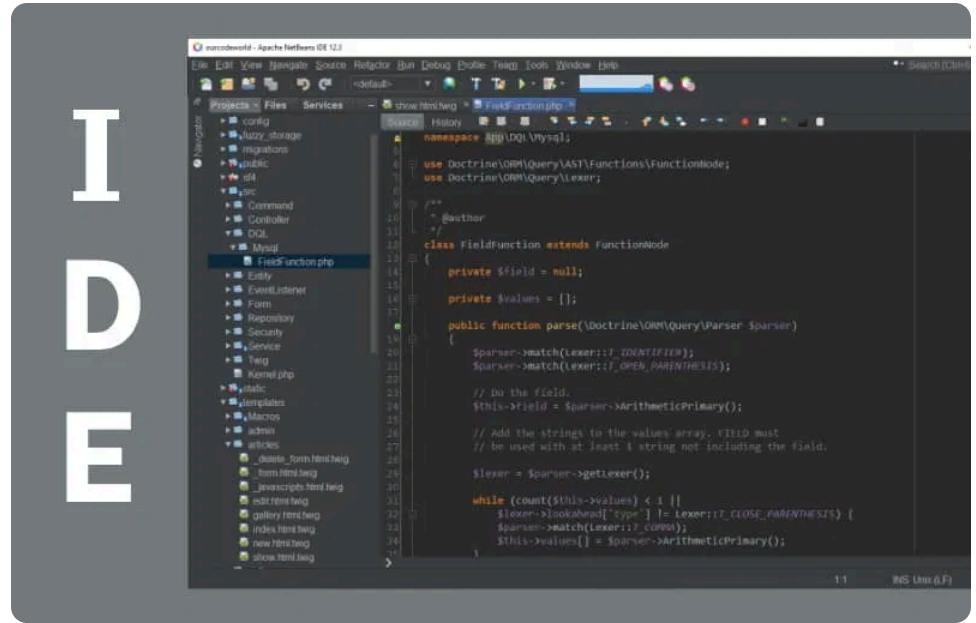
IDE 통합 인터페이스 (VS Code Extension 기준)

분석 서버로부터 받은 정보를 사용자에게 시각적으로 제공하고 상호작용합니다.

주요 기술: vscode-languageclient 라이브러리, Webview API

구현 기능

- 진단 정보 표시:** 서버로부터 `textDocument/publishDiagnostics` 알림을 받아 코드의 문제 지점에 빨간색/노란색 물결 밑줄을 표시합니다.
- 자동 업데이트 적용:** 서버로부터 `workspace/applyEdit` 요청을 받아 `vscode.workspace.applyEdit` API를 통해 문서 자동 업데이트를 수행합니다.
- 추천 코드 패널:** 서버로부터 '추천 코드' 목록을 받아 `vscode.window.createWebviewPanel`을 사용하여 IDE 측면에 별도의 패널을 생성하고 구문 강조 처리하여 렌더링합니다.



기대 효과 및 개발 로드맵

기대 효과

- 개발 생산성 향상:** 반복적인 오류 수정, 코드 검색, 문서 수정 시간을 획기적으로 단축하여 핵심 로직 개발에 더 집중할 수 있습니다.
- 코드 품질 개선:** 잠재적인 버그를 조기에 발견하고, 검증된 모범 사례를 적용하여 견고하고 유지보수가 용이한 코드를 작성하도록 유도합니다.
- 문서의 최신성 및 신뢰도 확보:** 코드 변경이 문서에 즉시 자동 반영되므로, 문서는 항상 최신 상태를 유지하며 팀원들이 신뢰할 수 있는 정보 소스가 됩니다.
- 팀의 개발 표준화:** 프로젝트 내 코드 스타일과 API 사용의 일관성을 유지하고, 팀원 간의 지식 격차를 줄이는 데 기여합니다.
- 신규 개발자의 빠른 적응 지원:** 항상 정확한 문서를 통해 프로젝트의 코드 베이스와 모범 사례를 빠르게 학습할 수 있도록 돕습니다.

개발 로드맵

- 1단계 (MVP):** 대상 언어: Python, 기능: 실시간 코드 감지, AST 파싱, API 일관성 검사 및 IDE 진단 표시, 플랫폼: VS Code
- 2단계 (핵심 기능 완성):** 기능: 문서 자동 업데이트 기능 구현, AI 추천을 위한 인프라 구축 및 기본 추천 기능 구현, 대상 언어: JavaScript/TypeScript 추가
- 3단계 (고도화 및 확장):** 기능: AI 모델 파인튜닝, 보안 취약점 분석 등 신규 기능 추가, 플랫폼: JetBrains, VS Code, Codium 등 여러 플랫폼들 (+웹 포함)

