# Notion-GitHub Agent Project Documentation

## Project Overview

This project involved building and demonstrating an AI Agent that connects Notion, GitHub, and OpenAI to provide intelligent answers using a single codebase. The system performs three main tasks — data retrieval, reasoning, and response generation — through a smooth, automated flow.

## How I Built the Agent

The agent works on a simple flow: Query → Retrieve → Reason → Respond, and it was developed inside one folder named notion\_git\_agent with four main Python files:

1. .env – Stores all the secret keys (Notion, GitHub, and OpenAI) safely using python-dotenv.  
2. notion\_connector.py – Connects to Notion’s API and searches the workspace for relevant documents.  
3. git\_connector.py – Connects to GitHub’s API to find matching code or files in the repository.  
4. reasoning\_engine.py – Sends the gathered information to OpenAI’s model (GPT-4o-mini) for reasoning and answer generation.  
5. main\_agent.py – Controls the complete flow, loads secrets, runs all the components, and prints the final answer with the data sources.

## How the OpenAI API Key Was Used

The OpenAI API Key powers the reasoning engine. After retrieving content from Notion and GitHub, the agent uses this key to send the data to OpenAI’s model, which understands the context and produces a clear, meaningful answer.

## How Relevance Was Determined

The agent checks relevance in two ways:  
1. Keyword Matching – Finds documents or files containing the user’s query.  
2. Contextual Understanding – The OpenAI model compares the information from both Notion and GitHub and links related content intelligently (for example, connecting “API v2” in Notion with “api/v2/handler.py” in GitHub).

## Main Challenge and How I Solved It

The biggest challenge was handling three different API authentications — Notion, GitHub, and OpenAI — securely and efficiently.  
To solve this:  
- I stored all API keys in a .env file for better security.  
- I used modular files for each connector to keep the code clean and organized.  
- I added validation checks in main\_agent.py to ensure all keys loaded before running any API call.

## Additional Debugging Step

At one point, my main\_agent.py file ran without any output. After researching and debugging, I created three helper scripts — tools/debug.py, tools/inspect\_file.py, and tools/inspect.py — which finally resolved the issue and made the agent run successfully.

Note: After integrating the data into the GitHub repository and Notion workspace, we will be able to seamlessly retrieve precise responses to any queries or tasks we intended to perform.