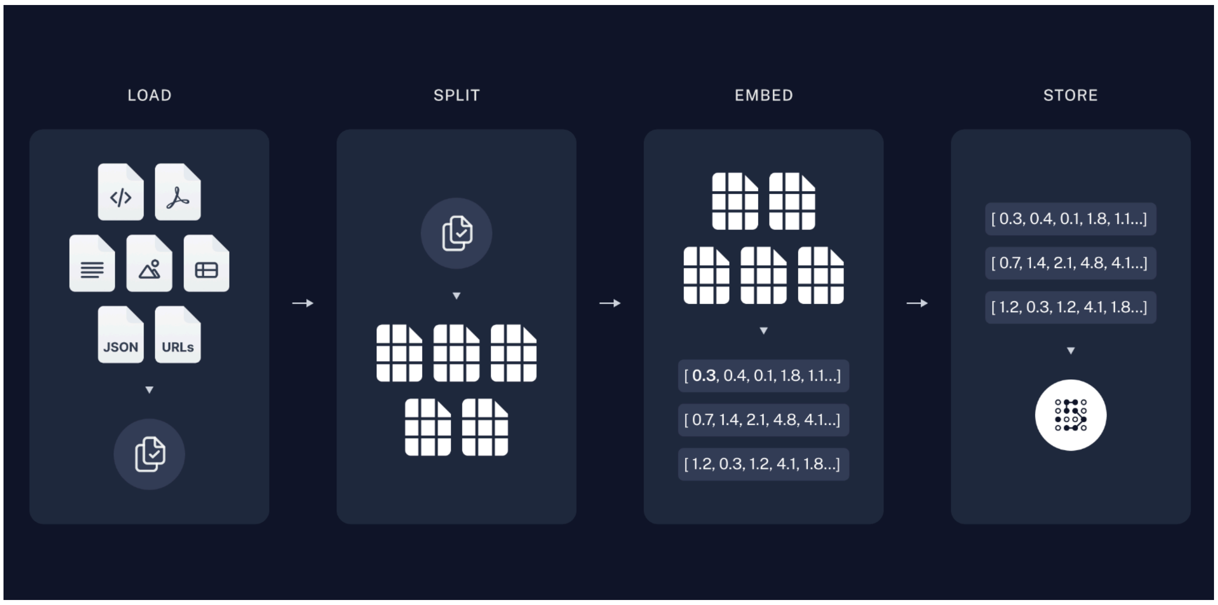
Repo Analysis Chatbot

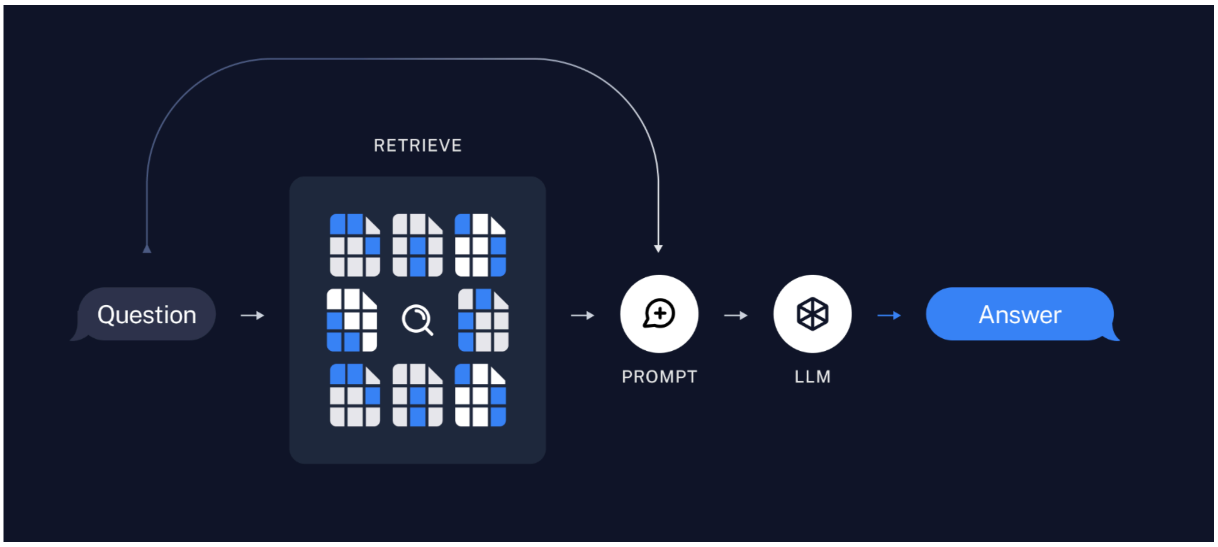
MVP & Project Plan

Shai Ballali	207380379
Yarin Barnes	318851300
Noy Keren	313307316

Introduction

The Repo Analysis Chatbot is an innovative tool designed to revolutionize the way developers interact with and understand codebases on GitHub. Utilizing the power of OpenAI's LLMs and LangChain framework, this chatbot simplifies the complex task of code analysis by providing real-time, conversational insights into any repository. With features like RAG for data ingestion and a community-driven "marketplace" of repositories, it fosters a collaborative environment that enhances productivity and knowledge sharing among developers. Whether for code quality assessment, security vulnerability identification, or simply navigating through a new codebase, the Repo Analysis Chatbot offers a seamless and intuitive solution to meet the diverse needs of the modern development landscape.





1. Identify And Understand The Business Needs

In developing our innovative chatbot, we have pinpointed and addressed a critical need within the software development community: the necessity for an intuitive, efficient means of navigating and comprehending complex codebases.

By leveraging the OpenAI API in conjunction with LangChain and the Retrieval-Augmented Generation (RAG) technique, our solution transcends traditional barriers to understanding source code.

This tool is not just a navigational aid but a comprehensive knowledge resource, enabling developers to conduct in-depth Q&A sessions about specific GitHub repositories, thus significantly enhancing their grasp of the code's functionality and structure. Additionally, our chatbot facilitates the automatic documentation of code, streamlining what is often a time-consuming process for developers.

This project is designed to serve as a pivotal tool for developers seeking to accelerate their workflow, enhance collaboration, and deepen their understanding of software projects, thereby addressing a fundamental business need for efficiency and clarity in software development processes.

Gone are the days of laboriously copying and pasting class by class to enable the chatbot to grasp the overall context. With Repo Analysis Chatbot, all you need to do is simply input the repository URL, and the task is complete.

2. Find The Opportunities

The Repo Analysis Chatbot opens up a world of possibilities for a diverse array of users, extending far beyond traditional developer-centric tools. **For developers**, it offers an unparalleled opportunity to dive deep into the intricacies of their code, uncovering insights that can lead to enhanced test coverage, optimized code quality, and a more holistic understanding of the codebase with minimal effort. This facilitates a more efficient debugging process and fosters a culture of continuous improvement and learning within development teams.

On the other hand, **non-technical stakeholders such as Product Managers and QA professionals** stand to gain a new level of clarity and insight into the technical landscape. Product Managers can quickly verify the alignment of code with business requirements and strategic goals, enabling more informed decision-making. QA professionals, through an intuitive Q&A interface, can dissect complex workflows and identify potential bottlenecks or areas for improvement, ensuring a more robust end product.

Moreover, the Repo Analysis Chatbot demystifies the technical details of software projects **for end users and other non-developer roles**, making the underlying technology more accessible and understandable. This transparency can foster stronger trust and engagement between users and the development team, contributing to a more inclusive and collaborative software development ecosystem.

3. Decide What Features to Build

- **RAG - Data Ingestion and Transformation:**
  At the heart of our chatbot's functionality is the Retrieval-Augmented Generation (RAG) feature, pivotal for ingesting data from specified GitHub repositories. This process involves transforming raw code and related documentation into a structured format, suitable for storage in a vector database. By embedding this data into a vector store, RAG enables the chatbot to retrieve relevant information efficiently when responding to user queries, ensuring responses are both accurate and contextually relevant.

- **Codebase Q&A via Chat Interface:**
  The Codebase Q&A feature represents the interactive heart of our chatbot, providing users with a conversational interface to explore and understand the codebase. This feature is underpinned by our ingested and transformed LLM (Large Language Model), which processes user queries in natural language. By prompting questions to our enriched LLM, users can delve into the intricacies of the code, receive explanations about specific functions, and gain insights into coding patterns and best practices, all through an intuitive chat interface.

- **Authentication with OpenAI using API Key:**
  Secure access to the chatbot's capabilities is ensured through authentication using an OpenAI API key. This process guarantees that only authorized users can initiate data ingestion processes and engage with the chatbot.

- **4. Social and Cooperative Feature:**
  Emphasizing the power of community and collaboration, our chatbot is designed to be inherently social and cooperative. Once a user initiates the scanning of a repository, that repository becomes accessible to the broader user base. This innovative approach not only fosters a sense of community among users but also optimizes resource utilization by eliminating the need for redundant data ingestion, thereby significantly reducing API usage costs.

- **Repository Marketplace:**
  Building on the social and cooperative nature of the project, we have introduced a "marketplace" feature that showcases the repositories currently embedded within our system. Users can browse this marketplace to discover and select repositories of interest, initiating inquiries and discussions around these pre-scanned codebases. This feature not only enhances user engagement by providing immediate access to a wide range of repositories but also leverages the collective intelligence of the user community to enrich the overall knowledge base of the chatbot.

4. Competition

While there are several open-source projects available on the internet and main well-known LLMs like GPT and Bard, we currently face no significant direct competitors in the market.

Our advantages:

- **Unique Capability:** Our tool specializes in extracting data from specific repositories, a feature not inherent in general LLMs.
- **Simplicity:** The tool is designed for ease of use; users can quickly understand a repository by simply entering its URL.
- **Community Power:** Once a repository is analyzed, its data becomes accessible to all users, fostering a collaborative environment.
- **Resource Efficiency:** This community-driven approach reduces redundant analyses, saving on computational resources and costs.

5. Future Optimization

- **Diversifying SCM Support**
  We will be able to incorporate support for alternative SCM (source control management) tools like Bitbucket & Gitlab.

- **Private Repository Scanning**
  Users will have the capability to conduct scans on private repositories and without the necessity of sharing them with the broader community, respecting individual privacy preferences.

- **Implementation of Predefined Tested Prompts**
  We plan to introduce predefined prompts to quickly gather essential information, enhancing the chatbot's efficiency. These will cover:

  - **Code Quality Metrics:** To evaluate and improve code standards.
  - **Security Vulnerability Identification:** For uncovering and mitigating potential security issues.
  - **Tests:** To assess test coverage and effectiveness.
  - **Documentation:** To ensure comprehensive and understandable documentation.

6. Roles & Responsibilities

Yarin - Overseeing the training and development of the Large Language Model.

Shai - Managing all backend aspects.

Noy - Accountable for UX/UI design, frontend development, and Quality Assurance.

7. Platform For Development

The Repo Analysis Chatbot leverages a streamlined tech stack for optimal performance and scalability. The core functionality is powered by **OpenAI** for LLM generation and **LangChain** for data integration, enriching the chatbot's responses. The project is structured within an **NX monorepo**, ensuring cohesive code management across the platform.

The frontend is developed using **Next.js** in **TypeScript**, offering a robust, type-safe user experience. For the backend, the choice is between **Java Spring Boot** for its rapid development capabilities and **Node.js** with **NestJS** for a TypeScript-friendly, scalable architecture.

Additionally, **Supabase** is employed as the database and vector store solution, providing a versatile backend with real-time capabilities and efficient data handling. This combination of technologies ensures that the chatbot is not only responsive and user-friendly but also highly adaptable for future expansions.

8. User Flow

Authentication with OpenAI API Key: Users authenticate by providing their OpenAI API key, which is required to access the chatbot's features.

Repository Input: Users either input the URL of a GitHub repository they wish to analyze or select from a list of repositories already available in the system's "marketplace".

Data Ingestion and Transformation: For new repositories, the chatbot uses the RAG process to ingest data from the repository, transforming and storing it in a queryable format.

Interactive Query Interface: Users interact with the chatbot through a chat interface, where they can ask questions about the repository's codebase in natural language.

Insightful Response Generation: Leveraging the ingested data and the capabilities of the LLM, the chatbot provides detailed responses to user queries, offering insights and explanations related to the codebase.

Community Knowledge Sharing: Once a repository's data is ingested, it becomes accessible to the broader user community, allowing others to benefit from the same insights without the need for re-scanning, thus fostering a collaborative environment.
