Onboarding Code Companion: A Vectorized AI Assistant for Private Repositories

DISSERTATION

Submitted in partial fulfillment of the requirements of the

Degree: MTech in DATA SCIENCE AND ENGINEERING

By

RENDUCHINTALA PHANI TEJA
2023DA04339

Under the supervision of

Shyamkumar Jha, Lead Software Engineer

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
Pilani (Rajasthan) INDIA

May 2025

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
SECOND SEMESTER 2024-25

DSECLZG628T DISSERTATION

Dissertation Title: Onboarding Code Companion: A Vectorized AI Assistant for Private Repositories

Name of Supervisor: Shyamkumar Jha

Name of Student: Renduchintala Phani Teja

ID No. of Student: 2023DA04339

Courses Relevant for the Project & Corresponding Semester:

1. Artificial Computer Intelligence (II semester)
2. Machine Learning (II semester)
3. Natural Language Processing (III semester)
4. Deep Learning (III semester)

Abstract

In software development, understanding extensive and complex private codebases presents significant challenges, especially when dealing with multiple interconnected business components and numerous solution files. Developers, whether new to a project or tasked with extending existing functionalities, frequently encounter difficulties in navigating and effectively understanding these extensive structures, despite the availability of documentation. Existing solutions, such as Microsoft's Copilot, have limitations in context-awareness, primarily restricted by workspace boundaries, rendering them inadequate for comprehensively handling segmented repositories composed of multiple interlinked components.

This dissertation aims to develop the "Onboarding Code Companion," a proposed advanced AI assistant designed explicitly to streamline developer onboarding and improve understanding of large-scale private repositories. The planned approach involves utilizing cutting-edge embedding techniques from OpenAI and Hugging Face to semantically encode source code and associated documentation, creating structured and context-rich vector representations. These representations will be systematically stored in vector databases such as ChromaDB or FAISS, organized clearly by repository identifiers to facilitate accurate and contextually relevant retrieval.

The backend infrastructure will leverage FastAPI within a scalable microservices architecture. Real-time webhook events, activated upon code merges or updates, will trigger targeted, incremental updates to embeddings. This strategy ensures efficient processing by focusing solely on modified or newly introduced files, thus maintaining responsiveness and minimizing unnecessary computational overhead.

To provide an intuitive user experience, a custom Visual Studio Code extension using ReactJS will be developed, offering a user-friendly, chat-based interface. Developers will be able to submit queries in natural language directly through this interface, leveraging Retrieval-Augmented Generation (RAG) to retrieve relevant embeddings and generate precise, contextually informed responses via OpenAI's GPT-4. Additionally, integration with LangChain will manage conversational context effectively, enhancing the relevance and accuracy of interactions.

The anticipated outcomes include significant improvements in developer productivity, substantial reductions in onboarding time, and precise retrieval of information from continuously evolving codebases. By combining advanced natural language processing, deep learning methodologies, and semantic retrieval techniques, the "Onboarding Code Companion" aims to contribute meaningfully to the development of intelligent assistance tools, addressing vital comprehension and navigation challenges within modern software engineering contexts.

Key Words: Generative AI, Embeddings, Information Retrieval, ChromaDB, FAISS, LangChain, GitHub Webhooks, FastAPI, ReactJS, Visual Studio Code Extension, Retrieval-Augmented Generation, Software Onboarding Assistant

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
II SEMESTER 24-25
DSECLZG628T DISSERTATION
Dissertation Outline (Abstract)

BITS ID No. 2023DA04339

Name of Student: Renduchintala Phani Teja

Name of Supervisor: Shyamkumar Jha

Designation of Supervisor: Lead Software Engineer

Qualification and Experience: Bachelor's in engineering in Computer Science, 8+ years of experience

Official E- mail ID of Supervisor: shyamkumar.jha@sciex.com

Topic of Dissertation: Context-Aware AI Assistants for Software Developer Onboarding

R. Phaniteja SK Jha

(Signature of Student) (Signature of Supervisor)

Date: 24/5/2025 Date:24/5/2025

Project Work Title:

Onboarding Code Companion: A Vectorized AI Assistant for Private Repositories

1. Purpose:

The purpose of this project is to develop an intelligent, context-sensitive onboarding assistant aimed at improving developer productivity and comprehension when interacting with large-scale, private code repositories. This assistant will facilitate efficient navigation and understanding through advanced embedding technologies and natural language querying, significantly enhancing developer onboarding processes and productivity.

2. Expected Outcome:

- An operational, scalable AI system capable of contextually understanding and assisting with complex private code repositories.
- A backend microservices architecture integrated with embedding generation and real-time webhook updates.
- A functional Visual Studio Code extension with an intuitive chat interface.
- Comprehensive documentation and performance evaluation of the developed system.

3. Literature Review:

[A] Vaswani et al. (2017) - "Attention Is All You Need": Established the transformer architecture foundational for embedding and generative models used in NLP and information retrieval.

[B] Semantic Web Technologies Journal: Discusses semantic retrieval mechanisms and embedding-based representation methods that enhance the relevance of information retrieval systems.

[C] LangChain Documentation: Provides guidelines and technical insights into managing conversational AI, memory, and context within AI-driven assistant frameworks.

[D] ChromaDB and FAISS Documentation: Highlight essential best practices for storing, retrieving, and managing embeddings efficiently within vector databases.

4. Existing Process:

Currently, developer onboarding and navigation of complex codebases rely heavily on traditional documentation, mentorship, and limited tools like Microsoft's Copilot. These methods are inefficient and often inadequate due to their limited contextual coverage and inability to handle vast, fragmented repositories effectively.

5. Limitations:

- Existing tools lack comprehensive context-awareness, leading to inadequate or misleading assistance.
- Manual onboarding methods are time-consuming and prone to human error.
- Current tools fail to provide incremental real-time context updates aligned with repository changes.

6. Justification for Methodology:

The use of advanced embedding technologies combined with Retrieval-Augmented Generation (RAG) and a microservices-based scalable backend ensures real-time accuracy, relevance, and efficiency. FastAPI and AWS-based microservices provide scalability and responsiveness. This methodology allows incremental updates and targeted retrieval of context, making it ideal for large, evolving codebases.

7. Project Work Methodology:

Phase 1: Literature Review and Requirement Analysis – Conduct comprehensive literature research and define detailed system requirements.

Phase 2: Architecture Design – Design a scalable microservices architecture, embedding workflow, and VS Code extension interface.

Phase 3: Backend Development – Implement microservices, authentication, repository access validation, and initial embedding generation pipeline.

Phase 4: Embedding Updates via Webhooks – Develop webhook-triggered incremental embedding updates.

Phase 5: Frontend Development – Build a ReactJS-based VS Code extension with a chat interface.

Phase 6: Generative AI Integration – Integrate GPT-4 and LangChain for accurate retrieval and response generation.

Phase 7: Comprehensive Integration and Testing – Perform thorough integration testing, debugging, and optimization.

Phase 8: Documentation and Deployment – Prepare comprehensive documentation, optimize the system, and make deployment-ready.

8. Benefits Derivable from the Work:

- Significant reduction in developer onboarding time.
- Enhanced productivity and accuracy in understanding large and complex code repositories.
- Real-time contextual assistance improving daily developer tasks.
- Scalable solution adaptable to various organizational repositories and environments.

9. Additional Details:

- The system architecture can easily integrate additional AI models or embedding techniques in the future.
- Potential integration with other developer tools beyond Visual Studio Code.
- Extensible design allowing further enhancements like real-time code review assistance and automated documentation generation.

1. **Broad Area of Work:** Artificial Intelligence in Software Engineering

2. Objectives

The objectives of my project are as follows:

- To develop an AI-powered onboarding assistant for navigating extensive private code repositories.
- To implement efficient embedding and retrieval methodologies for accurate code and documentation understanding.
- To build a scalable microservices backend integrated with real-time webhook-driven updates.
- To create a user-friendly interface using a Visual Studio Code extension.
- To evaluate improvements in developer productivity and code comprehension.

3. Scope of Work

- The scope of this dissertation is to design, implement, and evaluate an advanced AI system capable of assisting developers in comprehensively understanding and navigating complex private code repositories. The project will involve developing backend services, embedding generation and retrieval mechanisms, integration with webhook systems, and creating an interactive user interface via a Visual Studio Code extension.

4. Detailed Plan of Work

Serial Number of Task/Phases	**Tasks or subtasks to be done** (be precise and specific)	**Start Date-End Date**	Planned duration in weeks	Specific Deliverable in terms of the project
1	Project kickoff, finalize scope, gather requirements	03 May – 09 May	1	Problem statement and requirement documentation
2	Literature review and architecture brainstorming	10 May – 16 May	1	Literature survey and initial architecture notes
3	Finalize system design and tech stack	17 May – 23 May	1	Architecture document

				and finalized tech stack
4	Backend setup: FastAPI skeleton and AWS infra provisioning	24 May – 30 May	1	Functional FastAPI backend and infrastructure setup
5	Implement repo integration and access validation	31 May – 06 Jun	1	GitHub PAT/OAuth integration and repo access validation (MVP 1)
6	Build embedding generation service + parser	07 Jun – 13 Jun	1	Parser and embedding pipeline for .py/.cs/.docx/.yaml files
7	Setup ChromaDB/FAISS and enable vector search	14 Jun – 20 Jun	1	Vector DB configured with upsert/search logic
8	Create webhook listener for incremental updates	21 Jun – 27 Jun	1	GitHub webhook working and triggering updates (MVP 2)
9	Develop VS Code extension UI skeleton and backend connectivity	28 Jun – 04 Jul	1	VS Code extension basic interface connected to backend
10	Chat interface and prompt flow integration	05 Jul – 11 Jul	1	Chat system with query submission (MVP 3)
11	Integrate OpenAI GPT-4 and LangChain for RAG	12 Jul – 18 Jul	1	Retrieval-Augmented Generation working end-to-end
12	Add session memory and history using LangChain	19 Jul – 25 Jul	1	Conversational memory and state

				retention logic
13	Conduct full integration and bug fixing	26 Jul – 01 Aug	1	Stable integration build ready for testing
14	Performance tuning, logs, error handling	02 Aug – 08 Aug	1	Optimized APIs, monitoring, and exception handling
15	Prepare user manual, architecture doc, testing reports	09 Aug – 15 Aug	1	Final documentation including user guide and technical details
16	Final review, dissertation write-up, submission, and viva prep	16 Aug – 22 Aug	1	Final dissertation submission and presentation-ready system

5. Literature References

The following are referred journals from the preliminary literature review.

- Vaswani, Ashish, et al. "Attention Is All You Need." Advances in Neural Information Processing Systems (2017).
- Semantic Web Technologies: Trends and Research in Ontology-based Systems. Journal of Semantic Web and Information Systems.
- LangChain Documentation. Available at: https://docs.langchain.com/
- ChromaDB and FAISS Documentation. Available at: https://docs.trychroma.com/, https://faiss.ai/
- ReactJS and Visual Studio Code Extension Guidelines. Available at: https://code.visualstudio.com/api
- OpenAI and Hugging Face Embeddings Technical Specifications and Documentation.

Supervisor's Rating of the Technical Quality of this Dissertation Outline

EXCELLENT / GOOD / FAIR/ POOR (Please specify): EXCELLENT

Supervisor's suggestions and remarks about the outline (if applicable).

Date: 24/5/2025

SKJha

(Signature of Supervisor)

Name of the supervisor: Shyamkumar Jha

Email Id of Supervisor: shyamkumar.jha@sciex.com

Mob # of supervisor: +919738779745