# Project Proposal: Automated Code Analysis and Diagram Generation System

## Executive Summary

Our proposed system aims to streamline the process of onboarding new developers and facilitating project handovers by providing an automated solution for analyzing and visualizing codebases. Utilizing advanced AI algorithms, specifically transformer models, the system will analyze Python codebases integrated with FastAPI and MongoDB, automatically generating comprehensive flowcharts that depict the interactions and dependencies within the code. This visualization will enhance understanding and reduce the learning curve for new team members or during handovers.

## Objectives

1. Automate Code Analysis: Use state-of-the-art AI to analyze Python code structured with FastAPI and MongoDB, identifying all API endpoints and tracing through each function systematically.

2. Enhance Codebase Understanding: Generate detailed flowcharts that visually map out the function calls and interactions within the code, making complex systems easier to understand at a glance.

3. Streamline Onboarding and Handovers: Provide a tool that makes it easier for new developers to understand the system architecture and code interactions without needing to manually trace through the code.

## Technical Approach

1. Code Upload and Management:  
- Frontend: Develop a Flask-based web interface where users can upload the zip file of the Python codebase.  
- Backend Processing: Utilize FastAPI to handle file uploads, downloads, and the orchestration of code analysis tasks.

2. Code Unzipping and Preparation: Automatically extract the uploaded zip file on the backend to prepare the codebase for analysis.

3. Code Analysis:  
- AI Algorithm: Implement transformer-based models to parse and analyze the code. The analysis will start from each API endpoint and cover each function recursively.  
- Function Mapping: Identify and merge common functions to optimize the understanding and visualization of the codebase.

4. Diagram Generation and Delivery:  
- Flowchart Generation: After analysis, the system will automatically generate flowcharts representing the codebase structure.  
- Output Delivery: Flowcharts will be sent back to the target FastAPI backend server, where they can be accessed by the frontend for user viewing.

## Expected Outcomes

Reduced Onboarding Time: New developers can quickly get up to speed by studying the generated diagrams.  
Improved Code Maintenance: Easier identification of code redundancies and dependencies facilitates better code maintenance and scalability.  
Enhanced Documentation: Automatically generated diagrams serve as an up-to-date documentation resource, assisting in both development and code review processes.

## Risk Analysis

Technical Challenges: Ensuring the AI algorithm accurately interprets and analyzes complex code structures.  
User Adoption: Encouraging adoption within existing projects and integrating feedback into system improvements.

## Conclusion

This project represents a significant step forward in leveraging AI for software development practices, particularly in enhancing the accessibility and understandability of complex codebases for new developers and during project transitions. With careful implementation and ongoing support, this system promises to be a valuable tool in software development and project management.