# CS510 Project Proposal

Project Title: LLM-Assisted Open- Source Issue Summarizer and Contributor Finder

**Track: Development Track**

**Ankit Chavan**
auc3@illinois.edu

**Priyam Sahoo**
priyams3@illinois.edu

**Rohan Patil**
rspatil3@illinois.edu

**Sudarshan Shinde**
sshinde5@illinois.edu

**Sunveg Nalwar**
snalwar2@illinois.edu

**Coordinator:** Sunveg Nalwar (snalwar2@illinois.edu)

## Introduction

Open-source projects often have too many issues and not enough time or people to manage them well. Maintainers have to read through each issue, figure out what it is about, and find someone who can help, but this takes a lot of time and effort. There is no easy way to quickly match open issues with the right contributors who might already know about the code or have worked on something similar. We propose **LLM-Assisted Open-Source Issue Summarizer & Contributor Finder**, a tool designed to assist open-source maintainers by automatically analyzing open issues, extracting key details via LLMs, and matching each issue to the most relevant contributors based on historical repository data. This automation seeks to recommend potential contributors to contact who might know about the issue and changes, so as to ease the resolution of the issue quickly and request a review.

## Functions and Users

The goal is to suggest contributors who might have knowledge about the issue and can help resolve it quickly or review the changes. It will generate clear summaries of open issues, analyze recent commits, pull requests, and comments to identify active contributors with relevant expertise, rank and score these contributors based on how closely their past work relates to the issue, and either suggest or automatically assign the most suitable contributor by using the GitHub API. The main users of this tool will be open-source project maintainers, who need help organizing and managing incoming issues efficiently, and developers or contributors, who will benefit from being matched with tasks that fit their skills and interests.

## Significance

This tool can make a big difference in how open-source projects are managed. Right now, maintainers often have to spend a lot of time figuring out which issues are important and who should work on them. That process can be slow and inefficient, especially when there are many issues and contributors. By automatically summarizing issues and recommending the best contributors based on their past work, this tool helps projects run more smoothly. It not only saves time for maintainers but also helps developers get matched with tasks that fit their skills. This leads to faster bug fixes, better collaboration, and a more welcoming environment for both new and experienced contributors. Additionally, it can be especially helpful for newcomers by pointing them to the right people to contact when they run into problems, making it easier for them to get support and become active members of the community.

## Approach

The proposed system will be integrated with GitHub, leveraging the following technologies:

- **Backend:** Python will be used for server-side logic, data fetching via the GitHub API, and implementing LLM-based text analysis.
- **Frontend:** We are thinking of having a minimalistic UI using Streamlit or simply present this application as a command line tool.
- **LLM Integration:** Using OpenAI's API (or a similar model) to extract keywords and analyze the descriptions of issues.
- **IR specific technologies:** Embedding models (e.g., Sentence Transformers) or TF-IDF based approaches to compute textual similarity between issues and past contributions.

There are a few potential risks to address. Processing large amounts of repository data could be overwhelming, so the solution is to focus only on activity from the past 6 to 12 months. LLMs might misunderstand technical terms, which can be managed by fine-tuning the models with domain-specific data. Lastly, since GitHub's API might change, it's important to monitor updates and use version control to keep the integration working smoothly.

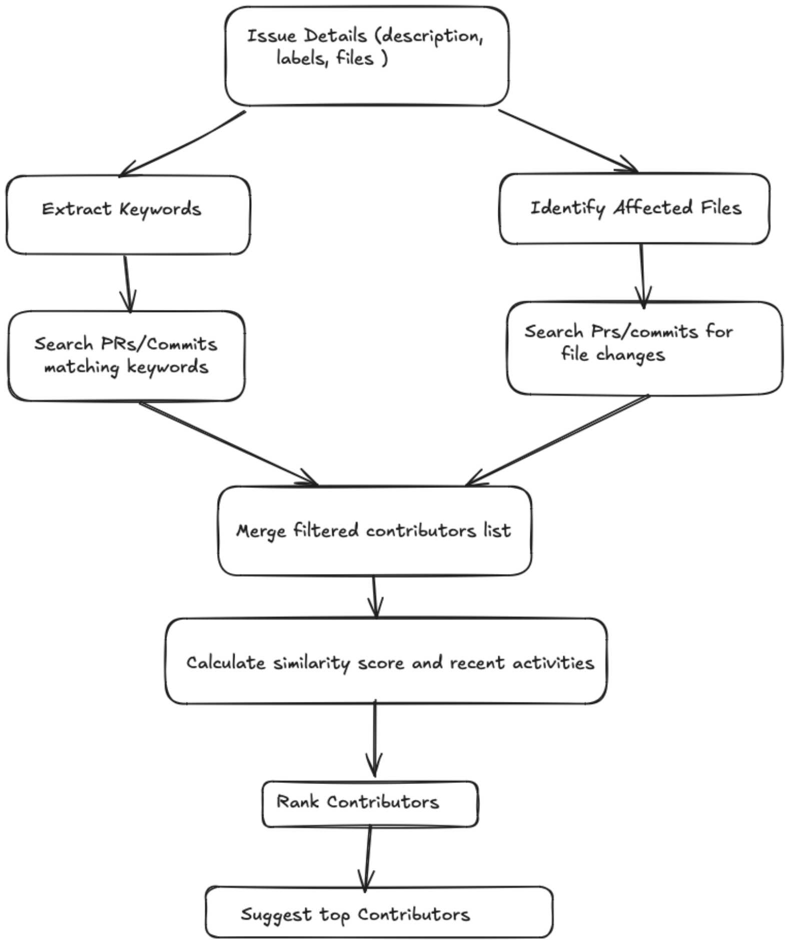
Issue Details (description, labels, files )
Extract Keywords
Identify Affected Files
Search PRs/Commits matching keywords
Search Prs/commits for file changes
Merge filtered contributors list
Calculate similarity score and recent activities
Rank Contributors
Suggest top Contributors

Figure 1: Project Workflow

## Evaluation

We plan to evaluate the tool by testing its performance on known Large-scale open source repositories. We plan to manually judge the recommendations by viewing commit history and git blame to verify them.

## Timeline

- **Week 1:** Project setup, initial design, and scoping of data sources.
- **Week 2:** Develop core modules (data fetching, LLM integration for keyword extraction).

- **Week 3:** Implement contributor matching and ranking algorithms.
- **Week 4:** Integrate with the GitHub API and develop the web interface.
- **Week 5:** Final testing, documentation, and project deployment.

## Task Division

Team roles are tentatively assigned as follows:

- **Priyam Sahoo:** Backend development, including API integration.
- **Sunveg Nalwar:** LLM integration, keyword extraction.
- **Ankit Chavan:** Frontend development and GitHub API integration.
- **Rohan Patil:** Data Processing and similarity computations.
- **Sudarshan Shinde:** Testing, evaluation, and documentation.