**Summary Report**

**Project: PubMed Paper Fetcher**

**Overview:-**

This project involves building a Python program to fetch research papers from PubMed based on a user-specified query. The program filters papers with at least one author affiliated with a pharmaceutical or biotech company and returns the results as a CSV file.

**Approach**

**1. Understanding the Requirements:**

- Fetch papers using the PubMed API.

- Filter papers based on author affiliations.

- Return results as a CSV file with specified columns.

- Implement command-line options for user interaction.

- Use Git for version control and Poetry for dependency management.

- Provide documentation in a README.md file.

**2. Planning the Program Structure:**

- Design a class `PubMedFetcher` to handle interactions with the PubMed API.

- Implement methods to fetch paper IDs, fetch paper details, filter non-academic authors, and save results to a CSV file.

- Create a command-line interface using `argparse` to handle user inputs and options.

**Methodology**

**1. Fetching Papers from PubMed:**

- Use PubMed's API to search for research papers based on the user-specified query.

- Support PubMed's full query syntax for flexibility.

**2. Filtering Papers Based on Author Affiliations:**

- Analyze author affiliations to identify non-academic authors using heuristics such as email addresses and keywords.

- List pharmaceutical/biotech companies.

**3. Returning Results as a CSV File:**

- Extract required information: PubmedID, Title, Publication Date, Non-academic Author(s), Company Affiliation(s), Corresponding Author Email.

- Format the results into a CSV file.

**4. Command-line Program Features:**

- Use `argparse` to parse command-line arguments.

- Implement options for query input, help display (`-h` or `--help`), debug information (`-d` or `--debug`), and output file specification (`-f` or `--file`).

**5. Dependency Management with Poetry:**

- Create a `pyproject.toml` file to define the project's dependencies.

- Use Poetry to manage and install dependencies.

**6. Documentation:**

- Document the code organization in a README.md file.

- Provide instructions on how to install dependencies and execute the program.

- Mention any external tools or libraries used.

**Results**

- Successfully implemented the `PubMedFetcher` class with methods to fetch and filter research papers.

- Created a command-line interface to interact with the `PubMedFetcher` class.

- Ensured the program adheres to the requirements, including fetching papers, filtering non-academic authors, and returning results as a CSV file.

- Used Poetry for dependency management and provided clear documentation in the README.md file.

**Conclusion**

The project successfully fulfills the task requirements. The Python program fetches research papers from PubMed, filters them based on author affiliations, and returns the results as a CSV file. The implementation includes command-line options for user interaction and uses Poetry for dependency management. The project is well-documented, making it easy for users to understand and execute the program.