Chat Transcript: Using LLMs for PubMed Tool Development

1. Initial Code Assistance

User:
I need a Python script to fetch PubMed articles based on a search query. It should filter articles with non-academic authors and output the data in a CSV format.

LLM:
Here's a script outline:

1. Use `requests` for API calls to PubMed's E-utilities.
2. Parse XML responses with `xml.etree.ElementTree`.
3. Identify non-academic authors by analyzing affiliations.
4. Output data with `csv.DictWriter`.

(*Followed by an initial draft of the code.*)

2. Debugging and Enhancements

User:
The affiliations filtering isn't working. How do I identify non-academic authors better?

LLM:
Use keyword matching for academic terms like "university" and "institute." Incorporate lowercasing and fuzzy matching for robustness. Add a function `is_non_academic()` to handle this.

(*Function provided and integrated into the script.*)

User:
How do I extract corresponding author emails from the affiliations?

LLM:
Parse affiliation text for tokens containing "@" using simple string operations. Here's a helper function:

```
python
Copy code
def get_corresponding_author_email(paper):
    for author in paper.findall(".//Author"):
        if author.find("AffiliationInfo/Affiliation") is not None:
            email = next((word for word in
author.find("AffiliationInfo/Affiliation").text.split() if "@" in word),
None)
            if email:
                return email.strip(".")
    return ""
```

(*This was added to the script.*)

---

3. Refinement and Validation

User:
The publication dates are inconsistent. Can we normalize them to `YYYY-MM-DD` format?

LLM:
Extract date parts (`Year`, `Month`, `Day`). Convert month names to numeric values with `datetime.strptime()`. Handle missing parts gracefully.

(*A solution was provided, implemented, and debugged together.*)

---

4. Conversion and Presentation

User:
Can you convert this Python script into a Jupyter Notebook?

LLM:
Here's a structured Jupyter Notebook version with Markdown cells for explanations, Python code blocks for each function, and a final block for execution.

(*Generated a* `.ipynb` *version, integrating it seamlessly.*)

User:
Can you summarize the approach and methodology for a report?

LLM:
(*Provided a detailed report outlining objectives, approach, methodology, challenges, and expected results.*)

User:
Make a transcript documenting how I used the LLM.

LLM:
(*Compiled this transcript to summarize interactions, showing how LLMs facilitated development and refinement.*)