WORD COUNT: 1637

This presentation serves as a proposal for the automation of an academic research solution.

SLIDE 1.

Our innovative biotechnology company has successfully developed an exceptional new concept for a medical device. In order to meticulously establish its feasibility through scientific means, our esteemed research team requires an automated search solution to enhance the efficiency of the online academic research process. The following proposal outlines a comprehensive end-to-end solution meticulously crafted by our technical team, to enhance the overall research workflow significantly.

SLIDE 2.

Problem Statement

The current manual approach to academic research presents numerous substantial challenges and limitations. This outdated method proves to be excessively time-consuming, leading to delays in the efficient validation of our pioneering medical device concept. Moreover, its susceptibility to errors further exacerbates the obstacles encountered in the research process. These drawbacks underscore the pressing need for a more streamlined and accurate approach to ensure the successful validation of our revolutionary medical device.

Proposed Solution Overview

Our proposed solution is a meticulously crafted, comprehensive, and integrated system specifically designed to augment the swiftness and efficiency of academic research. Through the adept utilisation of automation technologies and astute data processing methods, our objective is to automate intricate research tasks, implement cutting-edge algorithms, and elevate the overall levels of efficiency, accuracy, and productivity. By doing so, we envisage facilitating expedited and streamlined outcomes in the academic research process.

Team

Our exceptional team of experts is diligently working to construct the essential solution required for this project, combining their specialised skills and collaborative approach to ensure a successful outcome.

SLIDE 3.

Solution Design
Our Multi-Agent System (Wooldridge, 2009) is a system composed of multiple autonomous agents that interact with each other to achieve specific goals. Our Solution Architecture presents three agents: Data Retrieval and Extraction Agent (Shah et al., 2022), Data Processing Agent, and Data Storage Agent.

The Data Retrieval and Extraction Agent is designed to retrieve and extract relevant information related to a user query. It performs several actions to accomplish this objective. Firstly, it receives a user query as input. Next, it utilises the Google Search API and Scale SERP API to search for data that is relevant to the query. The agent retrieves the search results from these APIs and then proceeds to extract pertinent information from them. This includes details such as the URL, title, summary, authors, publish date, keywords, and even the full text of articles. Additionally, the agent is equipped with error handling capabilities, allowing it to continue processing even in the presence of extraction errors. Overall, the Data Retrieval and Extraction Agent acts as a mediator between the user query and the external APIs, facilitating the retrieval and extraction of data.

The Data Processing Agent is responsible for receiving the extracted information from the Data Retrieval Agent. It proceeds to download the articles referenced by the URLs obtained during retrieval. Using the **`extract_content_v2`** function, the agent performs specific processing tasks tailored to the extracted data. Additionally, it parses the downloaded articles, extracting any additional relevant information that complements the existing dataset. The agent then modifies the extracted data as needed, which may include tasks such as joining author's names or formatting publish dates facilitated by utils.py. Finally, the agent applies any necessary processing or transformations to enhance the quality and usability of the data. Overall, the Data Processing Agent plays a pivotal role in improving the quality and usefulness of the extracted information by performing various processing tasks and transformations.

The Data Storage Agent plays a vital role in the project by receiving the processed information from the Data Processing Agent. It utilises the **`write_to_csv`** function to save the processed data in a structured format, specifically a CSV file named "**`search_results.csv.`**" This structured data includes essential details such as the URL, title, summary, authors, publish date, keywords, and the full text of articles. Moreover, the agent provides convenient API endpoints (/search) to access the processed information easily. Doing so ensures seamless utilisation and consumption of the processed data by other applications. Overall, the Data Storage Agent efficiently stores the processed information, structures it for accessibility, and facilitates its utilisation by external applications.

All agents are of the goal-based type (Shah et al., 2022) working together in a coordinated manner, exchanging information, and collaborating to achieve the common goal of retrieving and processing relevant data for storage and further use (Bansall, 2023).

SLIDES 4, 5, 6, 7.

<u>Picture 1.</u> *"This is a quick code run-through of the implementation of these search agents. I'll start with the structure of the project. It has a folder **"data"** which has a **search_results.csv** file. This is where we save all the search results offline, which is in csv, and can be uploaded to wherever we want to keep it for the saving."* (Adeniyi, 2023)

<u>Picture 2</u>. *"**env.sample** - We have the environment variables. Here we used to store all the API keys."* (Adeniyi, 2023)

<u>Picture 3.</u> *"**configs.py** - this is how we make use of the environment variables in the code base."* (Adeniyi, 2023)

<u>Picture 4.</u> *"**factories.py** - you just simply separate all the agents that are being used and what they do. So this is like an agent that calls Google to scrape data from Google, and this is the agent that calls Google Scholar. So, we have these two agents here."* (Adeniyi, 2023)

<u>Picture 5</u>. *"We have the **main.py** which is the agent that makes use of other agents by just calling them (**SearchFactory( ).get_search**) to be able to do the autonomous search at Google and autonomous search at Google Scholar. So, this agent here (**main.py**) just kind of brings together all the other search agents that do autonomous google search."* (Adeniyi, 2023)

<u>Picture 6</u>. *"This is the **models.py**. This model just comprises the data structure of our offline document."* (Adeniyi, 2023)

<u>Picture 7</u>. *"This is an offline document, it looks this way, and it stored the query string which is "how to cure cancer":*

- *the **URL** for further research*
- *the **title** of the article ,*
- *the **summary** of the article,*

- *we have the **name of the author**,*
- *the **date** is published in some cases which are available ,*
- *and we also have the **keywords**,*
- *we have the **full text**,*
- *we have the **metadata***
  So this is the high level of structure of implementation of the agents that do search." (Adeniyi, 2023)

Picture 8. *"This is the **UI** (User Interface) that researchers use to make the call. So, for example, if I say "How to treat flu at home" and make the call. Right now it calls individual agents that search Google, that search Google Scholar and agents that write this into the csv file where the such results come. So, currently I'm searching and writing this."* (Adeniyi, 2023)

Picture 9. *"If I go back and click on the **search_results.csv** file I can see that now I have "Treating flu at home" and gives me a URL which I can actually double check to make sure it is the right one and I can see "10 remedies for flu"."* (Adeniyi, 2023)

SLIDE 8.

Risk Assessment:

In our diligent approach to risk assessment, we have identified potential risks and challenges associated with the project. While acknowledging these risks, we are well-prepared to tackle them with effective mitigation strategies.

Potential Risks are:
1. Synchronous HTTP Requests.
2. Inefficient CSV Writing.
3. Lack of Error Handling.
4. Missing Exception Handling in API Endpoints.
5. Deprecated Dependencies.
6. Missing Unit Tests

We have devised a comprehensive strategy incorporating various methodologies and best practices to address these challenges effectively. Our approach includes:

1. Adoption of the Asynchronous Programming Paradigm (Mozilla, 2023).
2. Implementation of Error Handling Best Practices (Elise, 2020).
3. Application of Exception Handling Patterns (Microsoft, 2023).
4. Prudent Dependency Management.

5. Embracing Test-Driven Development (IBM, 2023).

By integrating these strategies, we are confident in our ability to address the identified risks and overcome the associated challenges. Our Multi-Agent System will be optimised for performance, efficiency, error handling, maintainability, and reliability, ensuring the successful realisation of our project goals.

SLIDE 9.

Justification and Benefits
The design of our solution identified requirements and provided several benefits:

1. **Improved Efficiency:** Multiple autonomous agents with distinct responsibilities allow for workload distribution and task parallelization.
<u>Benefits:</u> This enhances efficiency in academic research by enabling concurrent and independent operations, reducing the time needed for data retrieval, processing, and storage.

2. **Enhanced Collaboration:** Agents in the Multi-Agent System interact seamlessly, with the Data Retrieval and Extraction Agent supplying data to the Data Processing Agent and the Data Processing Agent collaborating with the Data Storage Agent for storage and structuring.
<u>Benefits:</u> This collaborative approach streamlines workflow and facilitates the achievement of research goals.

3. **Modularity and Scalability:** The Multi-Agent System design allows for independent development, modification, and replacement of agents without affecting the system.
<u>Benefits</u>: Easy maintenance, future enhancements, and scalability to handle increasing data volumes and evolving research requirements.

4. **Intelligent Data Processing:** The Data Processing Agent employs advanced algorithms to enhance data quality and usefulness.
<u>Benefits:</u> Improved research accuracy and informed decision-making based on reliable data.

5. **Structured Data Storage and Access:** The Data Storage Agent stores processed information in a structured format and provides API access.
<u>Benefits:</u> Organised data for easy retrieval, seamless integration with other applications or systems.

Conclusion

In summary, our Multi-Agent System solution represents a comprehensive and advanced approach to automating the academic research process. By leveraging the power of autonomous agents, we ensure a robust and effective system that streamlines the entire research workflow. This innovative solution not only accelerates progress but also guarantees the delivery of accurate and reliable results. Through enhanced efficiency and productivity, our Multi-Agent System empowers researchers to make significant work strides, driving scientific advancements and knowledge acquisition.

The respective references applied at the end of the file.

References:

Adeniyi, S. (2023) Explainer video script for Team, 14 July.

Bansall S. (2023) Agents in Artificial Intelligence. *Available from:* https://www.geeksforgeeks.org/agents-artificial-intelligence [Accessed 08 June 2023].

Elise J. (2020) Handling Errors in Python. *Available from:* https://betterprogramming.pub/handling-errors-in-python-9f1b32952423 [Accessed 08 June 2023].

IBM (2023) Test-driven development. *Available from:*

https://www.ibm.com/garage/method/practices/code/practice_test_driven_development/ [Accessed 08 June 2023].

Microsoft (2023) Best practices for exceptions. *Available from:*

https://learn.microsoft.com/en-us/dotnet/standard/exceptions/best-practices-for-exceptions [Accessed 08 June 2023].

Mozilla (2023) Introducing asynchronous JavaScript. *Available from:* https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Asynchronous/Introducing [Accessed 08 June 2023].

Russell, S. & Norvig, P. (2021) *Artificial Intelligence: A Modern Approach.* (4th ed). Pearson Education.

Shah, U.S. et al.(2022) Agent-Based Data Extraction in Bioinformatics. *Hindawi. Available from:* https://www.researchgate.net/publication/359500106_Agent-Based_Data_Extraction_in_Bioinformatics [Accessed 08 June 2023].

Wooldridge, M. J. (2009) *An introduction to multiagent systems.* (2nd ed). New York: John Wiley & Sons.