COMP4801 Final Year Project

Detailed Project Plan

LLM-Enhanced Cross-Platform Web Search Application
Combining AI and Traditional Search Techniques

Supervisor: Professor Heming Cui

Student: Sze Shing Fung (3035930060)

Contents

1. Background

1.1. LLM-powered Chatbots and Web Searching

Recent advancements in Large Language Models (LLMs) have led to the development of AI-powered chatbots such as ChatGPT and Claude, which are predicted by Gartner [1] to be increasingly used for information retrieval in place of traditional search engines like Google in the future. With chatbots, users can obtain a short explanation for their queries instead of having to browse through numerous search results from traditional search engines to find their answers.

However, using chatbots for information retrieval presents two significant issues. First, chatbots rely on LLMs which may have a knowledge cutoff date since their training is done in the past. For example, GPT-4o mini's knowledge cutoff date is on October 2023 [2]. LLMs with a knowledge cutoff date may struggle to provide answers based on the most current information [Fig. 1.1].

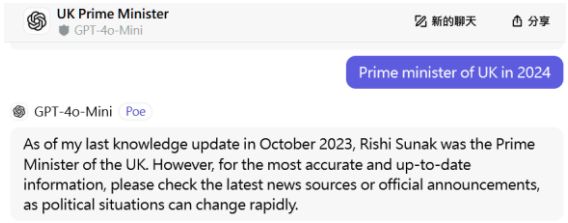

Fig. 1.1. Screenshot from Poe with GPT-4o-Mini [3], indicating it can't answer questions about current events beyond the knowledge cutoff.

Some chatbots such as Perplexity use current sources from the web and can thus avoid this problem. However, they along with other LLM-powered chatbots face the problem of hallucinations, where they can potentially provide inaccurate information to unknowing users. Yao et al. [4] found ways to construct adversarial attacks against LLMs to trigger hallucinations with an attack method reaching a success rate of 53.85% against LLaMA2-7B-chat and concluded that hallucinations can be a ”fundamental characteristic“ of LLMs. This shows that removing hallucinations from LLMs will not be a trivial task.

Moreover, the UI of chatbots such as Poe does not include the sources of their claim, potentially presenting information as the sole truth. Other chatbots including Perplexity include different websites as their sources, but still mainly present their summary as the single answer [Fig. 1.2]. This can lead users to accept the chatbot's answers as definitive truth, discouraging them from verifying the information or checking the sources.

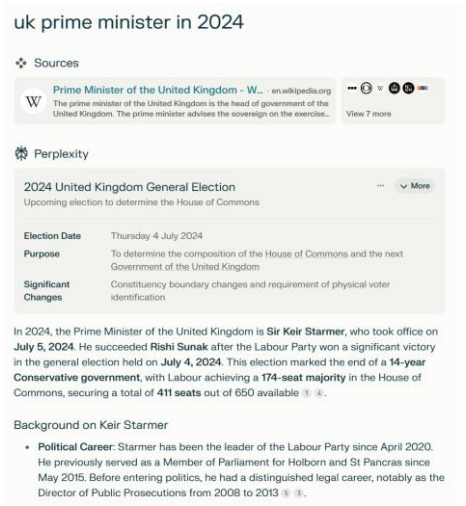

Fig. 1.2. Screenshot from Perplexity [5] showing the query, presenting the current answer and sources, but presenting it in a way that may suggest it as the sole truth.

1.2. Traditional Search Engines, SEO and Spam

Traditional search engines including Google and Bing, on the other hand, present results as a list of website links and the websites' digests. They are constantly impacted by evolving search engine optimization (SEO) techniques and spam content. A review of major search engines including Google and Bing performed from 2022 to 2023 [6] has found that the top search results are often search engine optimized and suggested that they may have lower quality. Thus, constant updates to search engines are required to combat them. For example, as of 1st October 2024, Google has performed 2 spam updates and 2 core updates for Google Search in 2024 [7]. However, the same review [6] has also found that search engine updates only have a temporary effect on reducing spam. Users may therefore have a degraded search experience in between updates.

Additionally, search engines sometimes struggle to handle complex queries and require users to dig through multiple search results to find their answers. For example, with the search query "generate multiple html and css files in vite", GPT-4o-mini provided detailed examples and allowed users to ask follow-up questions, while Google's top results contained questions on Stack Overflow and GitHub which may not suit the users' specific needs [Fig. 1.3].

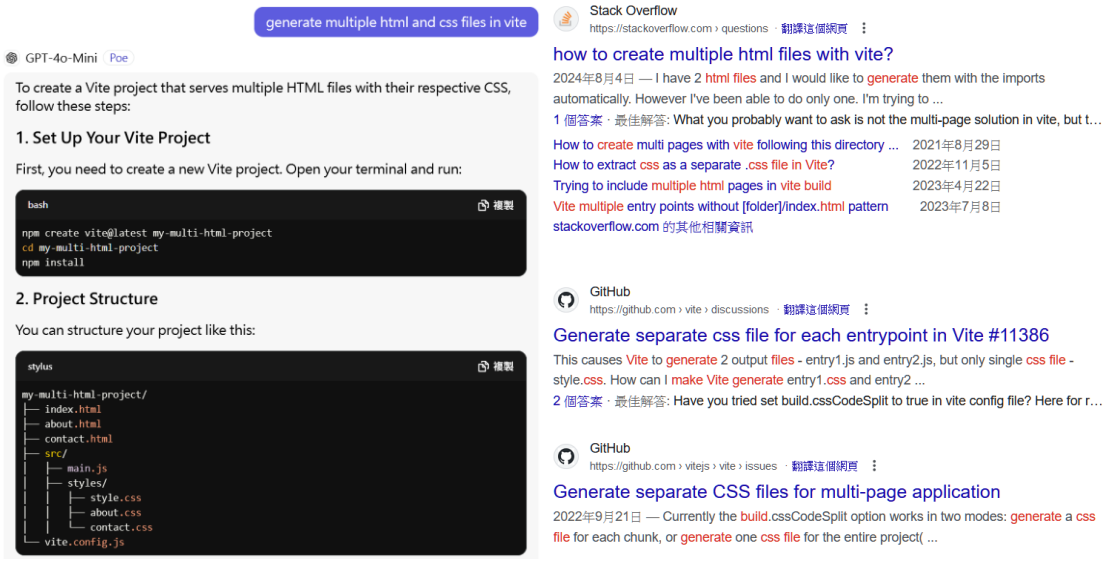

Fig 1.3. Result from Poe with GPT-4o-Mini (left) [3] and Google (right) with prompt "generate multiple html and css files in vite"

1.3. Project Introduction

Recognizing both the strengths and limitations of chatbots and search engines, this project proposes an innovative web search application. The application will enhance the search experience by reordering results to minimize spam, integrating outputs from multiple search engines, and generating concise summaries for individual results alongside a comprehensive topic overview.

The following sections of this project plan detail the objectives and scope of the project, outline the methodology and technologies for developing the application, and present a tentative project schedule.

2. Objective

2.1. Aim

This project aims to develop a cross-platform web-searching application. The proposed system will fetch search results from search engine APIs such as the Bing Web Search API and process multiple batches of search results, which will then be summarized and presented to the user in a traditional search engine format. The deliverables will include a web application, a mobile application, and a supporting backend system hosted on the cloud.

2.2. Use Case

The web search functionality can be illustrated through the following steps:

1. The user searches for a query in the frontend.
2. The backend takes the query and fetches relevant web search data via search engine APIs.
3. The backend accesses the website via the URL and parses the website's contents.
4. The backend uses the website's contents to evaluate and reorder the websites by their relevance.
5. A summary is generated for each website.
6. A general summary is generated from the top results.
7. The website URLs, their summaries and the general summary are presented to the user in the frontend.

2.3. Scope

The project will include the following systems and deliverables to support the use case:

1. Web search and parsing system

This system fetches relevant website links and extracts text content from the pages. While search engine APIs may provide summaries, they often lack detail, particularly for single-page applications where content may not load without JavaScript. Therefore, an effective parsing system is essential for retrieving the text data required for further processing by LLMs.

2. Website ranking and filtering system

This system uses LLM to reorder the website search results according to their relevance to the user query.

3. Webpage summary generation system

This system creates a summary for each website by querying LLMs with the website contents obtained from the web search and parsing system.

4. **General summary generation system**

This system retrieves the top results from the ranking system and generates a summary from them using LLMs.

5. **Backend REST APIs**

REST APIs will be developed to facilitate communication between the frontend and the backend systems.

6. **Frontend**

As of 2016, the majority of searches on Google are performed through mobile devices [8]. Furthermore, Google switched to prioritizing mobile versions of websites in indexing in 2020 [9]. These demonstrate the importance of providing a great mobile searching experience alongside a web frontend when building a web searching platform. Therefore, this project will produce both a mobile application in the form of a .ask file and a website for the users to access the web searching services.

As most users are expected to be using mobile devices, the project will take a mobile-first approach to UI design. Responsive design will be used to provide a seamless experience across devices with different screen sizes and aspect ratios.

7. **Payment system**

Considering the high running costs of using LLM services, a payment system will be implemented to cover the running costs of the application as the number of user queries grows.

8. **User authentication system**

The user authentication system supports the creation and authentication of user accounts, which are essential for the payment system and other functions such as search histories and user-defined themes.

9. **Search engine evaluations**

Evaluations will be performed to determine the best prompts and models to be used in each system. Additionally, user testing will be done to determine the effectiveness of the application in comparison with traditional search engines and existing chatbots.

2.4. Limitations

The app will focus on English searching. Multi-language support is not a focus of this project, but evaluations may be performed with other languages.

Video and image searching will not be supported in the application.

This project will not use user search history to customize users' search results.

3. Method

3.1. Frontend

Framework

This project will use Expo, a framework based on React Native, to build the frontend. For this project, using React Native comes with multiple advantages:

1. Providing a unified experience across Android, iOS and web by utilizing React Native for Web,
2. Reducing the development time required for targeting multiple platforms, and
3. Enabling integrations with native APIs instead of being limited by browser APIs.

Expo's native support for publishing websites makes it particularly suitable for this project.

Programming Language

The frontend will be developed using TypeScript instead of JavaScript. It provides type safety which helps prevent logic errors in codes. Additionally, TypeScript's type definitions for components, objects and functions can enhance the readability and maintainability of the projects' codes.

3.2. Backend

This project will implement its backend using managed cloud services. Compared to setting up local servers, this reduces the time required for maintenance and enhances the scalability of the service. It also reduces security concerns such as DDOS protection. AWS Lambda will be used to host the Express.js backend using the serverless-express module and Amazon API Gateway will be used to create REST API endpoints for frontend communication. AWS Lambda primarily charges compute costs with little to no ongoing running costs [10]. This minimizes idle costs and is particularly well-suited for this project, as the user base is expected to be initially small. As the user base grows, AWS Lambda's ability to scale seamlessly ensures the backend services can continue to run without disruption.

Additionally, a database will be required to store vital information including user details and search histories. Amazon RDS will be used as the database solution for this project. For the user authentication system, Amazon Cognito will be used to safeguard user credentials effectively.

Programming Language

The backend will also be developed in TypeScript. Besides the aforementioned advantages, sharing the same language with the frontend enables the reusing of codes and type definitions which helps reduce development time.

3.3. LLMs, Prompt Techniques and Evaluations

The project will utilize one of the available APIs for LLMs instead of considering self-

hosted models due to cost.

As prompt techniques and LLMs keep improving, the most suitable prompts and models for the project may change and should be determined through continuous evaluations. However, evaluating the performance of search engines can be challenging.

Recall and F1-score, two common information retrieval metrics, cannot be used due to the difficulty of calculating the total amount of relevant documents in the entire web given a search query [11]. Precision among the top retrieved results can be a suitable metric for complex search queries in this project, but determining the relevance of search results can be subjective. It may also fail to provide meaningful distinctions for simple and general queries such as "weather", as all the retrieved results are likely to be relevant. In addition, it does not take the rank of the search results into account.

A metric that considers the ordering of the search results is mean reciprocal rank (MRR), which can be calculated by [11]:

$$MRR = \frac{1}{|Q|}\sum_{q \in Q} \begin{cases} 0 \text{ if no relevant results} \\ (\text{Rank of first relevant result})^{-1} \text{ otherwise} \end{cases}$$

where Q is the set of the evaluated search queries.

The project will utilize a combination of precision and MRR evaluations to select the best prompts and models. In addition to the quality of the search results, the response time and cost of the LLM API calls must also be considered. Near the end of the project, a user test will be launched to evaluate the effectiveness of the application when compared to other search engines.

3.4. Software Testing

Unit tests will be implemented in both the backend and the frontend functions. After a functional prototype has been created, an integration test will be developed to ensure the software components work together seamlessly across the application after each update.

3.5. Source Control

Git will be used as the source control tool. A public repository will be hosted on GitHub for this project.

4. Tentative Schedule and Milestones

The following table shows the expected timeline, schedule and milestones of the project:

Timeline (duration from last date)	Schedule and milestones
2024-10-01	Detailed project plan
2024-10-13 (2 weeks)	Research on existing search engines and solutions
2024-10-20 (1 week)	Backend stub • Hosting service on the cloud • Fetching search data from API
2024-11-03 (2 weeks)	Backend functional prototype • Integrating LLM • Generating general summary and summary per result • Reordering search results by relevance
2024-11-17 (2 weeks)	Frontend functional prototype • Web app prototype • UI for search queries and showing results
2024-12-01 (2 weeks)	Refinements • Linking web app with backend • Implementing search history • Evaluating search effectiveness
2025-01-13 – 2025-01-17 (6 weeks)	First presentation
2025-01-26 (1 week)	Interim report
2025-02-01 (1 month)	Research and evaluations • Evaluating prompt techniques and different LLMs • Improvements based on findings
2025-03-01 (1 month)	Preparations for app release • User accounts and payment • Frontend refinements • Software testing
2025-04-01 (1 month)	Preparations for report and presentation User testing • Polls for app effectiveness
2025-04-21 (3 weeks)	Final report
2025-04-22 – 2025-04-26	Final presentation
2025-04-30	Project exhibition

Table 4.1. Tentative schedule and milestones

5. Reference

[1] “Gartner Predicts Search Engine Volume Will Drop 25% by 2026, Due to AI Chatbots and Other Virtual Agents,” Gartner. Accessed: Oct. 01, 2024. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2024-02-19-gartner-predicts-search-engine-volume-will-drop-25-percent-by-2026-due-to-ai-chatbots-and-other-virtual-agents

[2] “Introducing GPT-4o mini in the API - Announcements,” OpenAI Developer Forum. Accessed: Oct. 01, 2024. [Online]. Available: https://community.openai.com/t/introducing-gpt-4o-mini-in-the-api/871594

[3] “Poe.” Accessed: Oct. 01, 2024. [Online]. Available: https://poe.com/

[4] J.-Y. Yao, K.-P. Ning, Z.-H. Liu, M.-N. Ning, Y.-Y. Liu, and L. Yuan, “LLM Lies: Hallucinations are not Bugs, but Features as Adversarial Examples,” Aug. 04, 2024, *arXiv*: arXiv:2310.01469. Accessed: Oct. 01, 2024. [Online]. Available: http://arxiv.org/abs/2310.01469

[5] “Perplexity,” Perplexity AI. Accessed: Oct. 01, 2024. [Online]. Available: https://www.perplexity.ai/search/new?q=pending&newFrontendContextUUID=8200c74c-1369-4d77-b6af-36bfac36eab1&_rsc=1csqs

[6] J. Bevendorff, M. Wiegmann, M. Potthast, and B. Stein, “Is Google Getting Worse? A Longitudinal Investigation of SEO Spam in Search Engines,” in *Advances in Information Retrieval*, vol. 14610, N. Goharian, N. Tonellotto, Y. He, A. Lipani, G. McDonald, C. Macdonald, and I. Ounis, Eds., in Lecture Notes in Computer Science, vol. 14610. , Cham: Springer Nature Switzerland, 2024, pp. 56–71. doi: 10.1007/978-3-031-56063-7_4.

[7] “Google Search Status Dashboard.” Accessed: Oct. 01, 2024. [Online]. Available: https://status.search.google.com/products/rGHU1u87FJnkP6W2GwMi/history

[8] “Mobile-first indexing | Google Search Central Blog,” Google for Developers. Accessed: Sep. 30, 2024. [Online]. Available: https://developers.google.com/search/blog/2016/11/mobile-first-indexing

[9] “Announcing mobile first indexing for the whole web | Google Search Central Blog,” Google for Developers. Accessed: Sep. 30, 2024. [Online]. Available: https://developers.google.com/search/blog/2020/03/announcing-mobile-first-indexing-for

[10] “Serverless Function, FaaS Serverless - AWS Lambda - AWS,” Amazon Web Services, Inc. Accessed: Sep. 30, 2024. [Online]. Available: https://aws.amazon.com/lambda/

[11] P. Sirotkin, “On Search Engine Evaluation Metrics,” Feb. 10, 2013, *arXiv*: arXiv:1302.2318. doi: 10.48550/arXiv.1302.2318.