

Exercise Overview

Welcome! Thank you for interviewing at SHI AIML Team. As part of the interview process, we would like you to complete a design and coding exercise. The objective of this exercise is to assess your proficiency in designing, implementing, and integrating endpoints, as well as your ability to creatively incorporate a large language model (LLM) into a comprehensive Book Search web application. Below are the details.

Requirements for Book Search Feature

The exercise involves designing and implementing a Book Search web application, which includes both frontend and backend components. The frontend requires developing a user-friendly interface that allows users to search for books using natural language descriptions. The backend involves creating a service using Python that integrates with the OpenLibrary Web Service and a large language model (LLM) to process and enhance the book search queries and responses.

Frontend User Story

Title: As a user, I want to use a web browser-based user interface to search for books using natural language descriptions so that I can easily find books that match my interests.

Description: Develop a browser-based user interface that allows users to enter natural language descriptions to search for books. The frontend will interact with the backend book search API to fetch and display book recommendations in a user-friendly, natural language manner.

Acceptance Criteria:

- The user interface and experience should be nicely designed. It should have a good header and footer, and body. Provide proper titles. The UI/UX shall be friendly and functional.
- The user interface should have a search bar where users can enter their natural language queries. You can go beyond this requirement and implement a speech-to-text interface.
- The interface should return and display a list of book recommendations, nicely formatted and presented, based on the user's natural language query, including titles, authors, and brief descriptions, all nicely summarized using natural language, all nicely organized.
- The interface should handle loading states and display appropriate messages when no results are found or if an error occurs.
- The UI/UX design should be responsive and accessible, ensuring a good user experience on both desktop and mobile devices.

Backend User Story

Title: As a user, I want to search for books using natural language descriptions so that I can find books that match my specific interests.

Description: Implement a backend service using Python that provides a book search Web API. This API should allow users to enter natural language descriptions to search for books. The backend will integrate the OpenLibrary Web Service to fetch book data and a large language model (LLM) to process and enhance the book search query description, and the fetched book data, all using natural language.

Acceptance Criteria:

- The backend service should expose an endpoint `/search-books` that accepts a POST request with a JSON payload containing the user's natural language query.
- The service should integrate an LLM to process the natural language search query, extracting appropriate information to search for books.
- The service should call the OpenLibrary Web Service to fetch relevant book data based on the query.
- The service should integrate an LLM to process the fetched book data and generate a natural language response.
- The response should include a list of book recommendations with titles, authors, and brief descriptions, formatted in natural language.
- The service should handle errors gracefully and provide meaningful error messages to the user.
- The service should have low latency, ideally from 1-3 seconds max.
- The service should implement basic moderation and not allow for profanity to be used in the query, and if tried, a proper message should be displayed, for example, "The Book Search service is moderated, and does not allow for profanity".

General Example

The User: I'm looking for a book that's a mix of mystery and romance, set in a small town, with a strong female lead who solves a crime. Give me up to 3 recommendations.

The Book Search App:

That sounds like a fascinating combination! Here are a few book recommendations that match your description:

- **"Big Little Lies" by Liane Moriarty** - This novel weaves together mystery and romance in a small-town setting. It features strong female characters who navigate complex relationships while uncovering hidden secrets. You'll be drawn into the suspenseful plot and the intricate dynamics between the characters.
- **"The Cuckoo's Calling" by Robert Galbraith** - Written by J.K. Rowling under a pseudonym, this mystery novel introduces you to a strong female detective who tackles a high-profile case. The story blends elements of romance and mystery, making it a compelling read for fans of both genres.
- **"Still Life" by Louise Penny** - Set in the charming village of Three Pines, this mystery novel follows a female protagonist who unravels the secrets of the town while solving a crime. The book

beautifully combines the cozy atmosphere of a small town with the thrill of a mystery, all while developing a strong and relatable female lead.

These recommendations should keep you captivated! Are you interested in a particular time or period or setting for your next book search? Or maybe you'd like to explore a different genre combination?

Deliverables

Based on these deliverables, we will assess your proficiency in designing, implementing, and integrating endpoints, as well as your ability to creatively incorporate a large language model. Below are the details.

1. Timeline:

- a. You have **7 days to complete the code exercise** (from the day it was shared).
 - i. Understand the requirements and plan the implementation.
 - ii. Write the Design Spec.
 - iii. Develop both frontend and backend components.
 - iv. Integrate the OpenLibrary Web Service and the large language model (LLM).
 - v. Test the application thoroughly to ensure it meets the requirements.
 - vi. Make any necessary refinements and optimizations.
- b. Speed matters; the quicker you deliver *your complete solution*, the better (extra points).

2. Writing a detailed Design Specification:

- a. Write a detailed design specification document outlining the design of the end-to-end software. Be as detailed and complete as possible. This should include the requirements being addressed by the design spec, an overview of the design, scope, the components involved, the flow/sequences of the application, UI/UX, risks and constraints, along with clear explanatory *narratives*. Include other areas that you feel should be included. Add your alternate designs, if any, as an appendix.

3. Developing a browser-based user interface (Frontend):

- a. Design the UI and the UX. Be creative.
- b. Develop the browser-based user interface using HTML, CSS, and JavaScript (or any framework of your choice, for example, React).
- c. The UI allows users to input a book name using natural language and display the matching books retrieved from the backend, also in natural language.
- d. The web UI integrates with the backend endpoints.

4. Creating and Integrating Backend endpoints:

- a. Design the backend and flow and integrations. Be creative.
- b. Implement the backend using Python.
- c. Create an endpoint that exposes a book search Web API.
- d. This endpoint and service integrates with the OpenLibrary endpoint and calls an LLM. The backend service deals with a natural language-based request, generates natural language responses, and returns the results.

- e. Operational aspects and deployment are all important.

5. The Source Code:

- a. Provide the complete code for the frontend and backend in a GitHub repository; make it public so it is easy to access.
- b. Ensure the code is clean, well-documented, and includes comments explaining the logic and flow.

6. The Running Application:

- a. Ensure the code is running somewhere and share a link to it. This will allow us to interact with the code as we evaluate it and prepare for the interview demonstration. *Less ideal* is to provide a link to a video of the web app and service running.

Evaluation Criteria

A well-crafted design specification document is essential. The code should adhere to SOLID principles, maintain clean code standards, and demonstrate meticulous attention to detail. Delivering a comprehensive solution that meets all requirements is crucial. Additionally, speed is a key factor; the faster the delivery, the better.

- ✓ **Design Skills:** Well-written Design Specs are important in this team and have a lot of weight in the interview process. We are looking for a complete, clear and concise design document with well-explained architecture and interactions, solid and complete design, attention to details, narratives, and creativity.
- ✓ **Technical Skills:** Proficiency in Python, web development, API design and integration, and LLM integration.
- ✓ **Coding Skills:** Clean, efficient, follows best practices, and well-documented code.
- ✓ **Problem-Solving Skills:** Ability to design and implement a creative solution, handle challenges, and optimize the application.

The AIML team wishes you good luck! We look forward to seeing your solution!