ITAI 2373

Group 5 Final Journal Entry

Franck Kolontchang - Iman Haamid - Kimberly Navarrete - Marvin Azuogu

Github Link:
https://github.com/imid12/miniature-eureka-Group5/tree/main/ITAI2373-NewsBot_2.0-Final

Today's journal entry is all about mapping out the core architecture of my NewsBot 2.0. This reflection is crucial for building a robust and intelligent system that can not only deliver news but also provide meaningful insights. Let's break down the plan.We need to analyze the plan.

Reflecting on NewsBot 2.0

Building a next-generation news bot requires a thoughtful approach to system architecture, data processing, and user interaction. This journal entry serves as a reflection on the core components and key challenges ahead.

---

1. System Architecture & Data Flow

To function effectively, NewsBot 2.0 needs a modular and scalable architecture. I've identified four main components:

- **Data Ingestion Module:** This is the entry point, responsible for fetching raw articles from various sources.
- **NLP Pipeline:** The heart of the bot, this module will process and enrich the article data.
- **Database:** A central repository to store raw articles, processed data (entities, topics, sentiment), and user interaction logs.
- **User Interface (UI) / API:** The external-facing component that handles user queries and delivers responses.

Data will flow in a linear path. First, the **Data Ingestion Module** will pull articles and pass them to the **NLP Pipeline**. The pipeline will then perform a series of operations (like topic modeling, sentiment analysis, and entity extraction) before storing the final, enriched data in the **Database**. Finally, when a user asks a question, the **UI/API** will query the database to retrieve and format the most relevant information.

The code will be organized into distinct modules, one for each component, to promote reusability and maintainability. Components will communicate via well-defined interfaces or message queues, and a central configuration file will manage all settings, from API keys to database connection strings.

---

2. External Services and Data Processing

To build a robust NewsBot 2.0, we will need several external services:

- **News APIs:** To access a wide range of news sources.
- **LLM APIs:** For advanced NLP tasks like summarization, entity extraction, and sentiment analysis. This is a crucial upgrade from NewsBot 1.0.
- **Translation APIs:** To handle multilingual content.

When handling multiple categories per article, the NLP pipeline will assign a probability score to each potential category. This will allow the bot to understand that an article might be primarily about "Technology" but also have elements of "Business." For topic modeling, we will need to preprocess text by removing stopwords and performing stemming before feeding it to the model. We'll experiment with different numbers of topics and use metrics like perplexity to find the optimal number. We can evaluate the quality of a topic by manually inspecting the most common words and articles associated with it.

---

3. User Interaction and Content Generation

The new bot must provide a seamless and intelligent user experience. A key challenge is generating high-quality summaries. What makes a good summary can depend on the type of news. For breaking news, a short, factual summary is best. For feature articles, a more comprehensive, analytical summary is more appropriate. The bot will need to use an LLM with specific prompt engineering to tailor the summary style.

Handling ambiguous user queries is a significant challenge. The bot will need to maintain a conversational context, remembering previous questions and answers to infer the user's intent. For example, if a user asks about "the market," and then "what about tomorrow?", the bot needs to understand "tomorrow" in the context of "the market."

4. Technical and Operational Concerns

Building a complex, integrated system like this comes with potential pitfalls. Efficient communication between components is vital. Using a message-based system will help decouple components and prevent bottlenecks. Error handling must be robust; what happens if an API call fails or a network connection drops? The system should be designed with retries and fallbacks.

Testing will be a critical phase, especially for the complex integrated functionality. We'll need a combination of unit tests for individual modules and end-to-end integration tests to ensure all components work together as intended. Performance bottlenecks are most likely to occur in the **NLP Pipeline**, as processing large volumes of text can be computationally intensive. Caching results and using optimized algorithms will be essential.

NewsBot 2.0 is a news assistant with a modular design, composed of four main parts:

- **Data Ingestion Module:** This component pulls in news articles from various sources.
- **NLP Pipeline:** The core of the bot, this module processes articles to extract key information like topics, sentiment, and entities.
- **Database:** A central hub where all raw and processed data is stored.
- **User Interface (UI):** The part of the system that interacts with users and answers their questions.

The bot will use external services like **LLM APIs** for advanced features such as summarizing articles and handling complex user queries. It will also be designed to manage ambiguous questions by maintaining conversational context and will use robust error handling to manage issues like failed API calls. The entire system will be built to be scalable and will undergo rigorous testing to ensure all parts work together smoothly.