Bookmark Assistant Bot: AI-Powered Bookmark Management System

Track: Development Track

Team Members:

- Xiaoke Liu (email:xiaokel4@illinois.edu)
- Congzheng Chen (email: cc186@illinois.edu)
- Qi Jing (email: qijing3@illinois.edu)
- Xiu Dong (email: xiudong2@illinois.edu)

Coordinator: Qi Jing

Functions and Users

We propose to develop a **“Bookmark Assistant Bot”**, a tool that helps users manage and retrieve their saved posts on social platforms like RedNote (小红书), Instagram, or similar apps. The envisioned tool will be a **standalone web-based application** that allows users to (1) forward useful posts to the bot, (2) automatically extract and index key information using NLP, and (3) retrieve saved content using natural language queries later. The main functions will include content ingestion (via forwarding links, screenshots, or text), intelligent tagging and summarization, and semantic search. The target users are **frequent social media users** who habitually save content but often forget or fail to revisit it.

Significance

Many users frequently save social media posts as a way to "come back later", but this rarely happens due to poor organization, lack of memory, or volume overload. Our bot addresses this common **“digital forgetfulness”** by turning saved content into searchable knowledge. It serves a real need by offering a lightweight personal memory system, increasing productivity, reducing time spent re-finding content, and promoting better knowledge organization.

Approach

We will implement the tool using **Python (backend)** with **FastAPI**, a **Gradio frontend**, and **ElasticSearch** for search capabilities. For NLP tasks such as summarization, keyword extraction, and semantic indexing, we plan to use **spaCy**, **transformers (e.g., BERT)**, and **sentence embeddings (e.g., SBERT)**. We may use **OCR (like Tesseract)** for screenshot-based input. Risks include data heterogeneity (e.g., post formats), limited API access to social media platforms, and the challenge of semantic search. To mitigate this, we’ll focus on minimal viable inputs (e.g., user copy-pastes text manually or uploads screenshots).

Evaluation
We will evaluate our tool by conducting user tests with a small group of real users. Key evaluation criteria include (1) ease of use, (2) success rate of retrieving desired content using natural queries, and (3) performance and accuracy of summarization and keyword extraction. We'll also perform internal tests for indexing correctness and search latency.

Timeline

- Week 1: Finalize system design, set up backend and frontend scaffolding
- Week 2: Implement core features (input interface, keyword extraction, tagging, search indexing)
- Week 3: Build semantic search interface and connect query to database
- Week 4: Internal testing, bug fixes, and feature refinement
- Week 5: User testing and feedback collection
- Week 6: Final report and presentation prep

Task Division

Congzheng Chen, Xiu Dong: backend development (FastAPI server, ElasticSearch integration, NLP pipeline)

Xiaoke Liu: frontend development (interface, query input, result display) and design

Qi Jing (coordinator): system integration, user testing setup, and evaluation metrics