Internship Assignment: LLM-Powered Flashcard Generator

Project Summary

You are tasked with building a lightweight yet robust Flashcard Generation Tool that utilizes Large Language Models (LLMs) to convert educational content into effective question-answer flashcards. This tool is expected to demonstrate LLM integration capabilities, simple user interaction, and clean code practices.

Objective

Design and implement a system that:

- **Ingests educational materials (e.g., textbook excerpts, lecture notes).**
- **Automatically extracts and generates relevant flashcards (Q&A format).**
- **Offers a basic UI/UX for interaction, viewing, and exporting cards.**

Input Specifications

- **Accepts raw educational content via:**
  - **File upload (.txt, .pdf)**
  - **Direct paste input**
- **Optional: Subject-type selection (e.g., Biology, History, CS) to guide prompt formatting or categorization.**

Output Requirements

- A minimum of 10–15 flashcards per input submission.
- Each flashcard must include:
  - Question (clear, concise)
  - Answer (factually correct, self-contained)
- Bonus: Auto-group flashcards under detected topic headers or sections.

Technical Guidelines

Core Stack

- Language: Python (preferred)
- Framework/UI: CLI, Streamlit, or Flask
- LLM Interface: Choose any from:
  - OpenAI GPT (e.g., gpt-3.5-turbo)
  - Hugging Face models (e.g., flan-t5, mistral)
  - Open-source LLMs (e.g., Phi, LLaMA)

Bonus Functionality (Optional, but Encouraged)

- Export Options:
  - .csv, .json, Anki, or Quizlet formats
- Flashcard Enhancements:
  - Add difficulty levels (Easy, Medium, Hard)
  - Edit functionality for user review before export
  - Multi-language support (translation of flashcards)
  - Detect and preserve structure (e.g., subheadings, chapters)

Deliverables (Due in 2–3 Days)

1. **A public GitHub repository containing:**
    - **Codebase**
    - **README.md with setup, usage instructions, and sample outputs**
2. **Sample execution:**
    - **At least one run with real-world input (e.g., textbook chapter)**
3. ***(Optional)* A short demo video (2–3 minutes)**

Evaluation Criteria

- **Successful LLM integration (API or local model)**
- **Relevance, accuracy, and formatting of flashcards**
- **Simplicity and clarity of UI/UX (even minimal is fine)**
- **Code quality:**
    - **Modularity**
    - **Readability**
    - **Reusability**
- **Bonus points for:**
    - **Feature extensions (see above)**
    - **Clean deployment (e.g., Streamlit share, Flask app hosted)**
    - **Creativity in approach or formatting**

Notes from the AI Team

- **You are not expected to build a full-scale product — focus on prototype-level functionality.**
- **Your ability to reason through the LLM prompt design and handle input edge cases will be valued.**
- **Keep the tool extensible; think about how someone else could build on top of your work.**