### User Stories for "AI Tutor Chatbot"

#### **1. App Hosting and Performance**

* **As a user**, I want the app to run locally on my CPU or be hosted on the web running on CPU-only configurations, so that I can use it without requiring expensive hardware like GPUs.
* **Acceptance Criteria**:
  + The app can be executed locally with efficient performance on a CPU.
  + The hosted version is optimized for CPU usage, with no dependency on GPUs.

### **2. Interactive User Interface**

* **As a user**, I want an interactive and user-friendly UI to interact with the AI Tutor, so that I can perform tasks like uploading files, entering URLs, querying, and managing vector databases without technical difficulties.
* **Acceptance Criteria**:
  + The UI is built using a modern framework like **Gradio**, providing a clean, responsive design optimized for interactivity.
  + The UI includes intuitive features such as:
    - Buttons for uploading documents and entering URLs.
    - Options for resetting the temporary database.
    - Tabs or sections for "Study Guide" and "Quiz" modes.
    - Save and Load options for managing the in-memory database and chat history.
  + The app is compatible with major browsers (e.g., Chrome, Edge, Firefox).

### **3. Document and Web Data Integration**

#### **3.1 Upload and Merge Documents into Permanent Vector Database**

* **As a user**, I want to upload documents and provide web addresses for data extraction, so that I can merge the extracted content into the "permanent" vector database (general\_db or mcq\_db) stored in the working directory or in Azure Blob Storage if hosted in Azure Cloud.
* **Acceptance Criteria**:
  + Users can upload documents in supported formats (e.g., PDF, Word, PPTX) via the UI or through Python scripts.
  + Users can provide a list of URLs for web scraping or data extraction.
  + Extracted data is stored in the permanent vector database (general\_db or mcq\_db) only when explicitly chosen by the user.
  + Clear feedback is provided on the success or failure of the merge operation.

#### **3.2 Reset Temporary Vector Database**

* **As a user**, I want to reset the temporary in-memory vector database, so that I can ensure the preloaded permanent vector databases (general\_db and mcq\_db) remain unchanged while I test or explore new content.
* **Acceptance Criteria**:
  + Users can reset the temporary vector database with a single click through the UI or via Python scripts.
  + Resetting does not affect the permanent vector databases (general\_db, mcq\_db).

### **4. Querying and Contextual Responses**

#### **4.1 Generate Study Guides**

* **As a user**, I want the AI to query the general\_db vector database, so that I can receive accurate study guides tailored to my specific domain or subject queries.
* **Acceptance Criteria**:
  + AI retrieves relevant information from general\_db to generate study guides.
  + Study guides are clear, concise, and include summarized key points relevant to the query.

#### **4.2 Generate and Evaluate Quiz Questions**

* **As a user**, I want the AI to query the mcq\_db vector database, so that I can practice quiz questions generated from stored MCQ examples.
* **Acceptance Criteria**:
  + AI generates MCQs with correct answers, distractors (wrong options), and explanations for the correct options.
  + Users can evaluate their answers against the correct options and receive immediate feedback.

#### **4.3 Utilize All Vector Databases for Contextual Responses**

* **As a user**, I want the AI to combine in-memory, general\_db, and mcq\_db vector databases for contextual responses, so that I can leverage all available knowledge sources in a single interaction.
* **Acceptance Criteria**:
  + The AI combines all three databases (in-memory, general\_db, mcq\_db) to generate the best possible response.
  + Users are informed when the AI utilizes data from the in-memory database, permanent databases, or both.

#### **4.4 Fallback to Internal Knowledge Base**

* **As a user**, I want the AI to rely on its internal knowledge base if no meaningful or related context is retrieved, so that I can still have productive conversations with the AI as an expert "AI Tutor."
* **Acceptance Criteria**:
  + The AI generates meaningful responses even when database queries return no results.
  + Users are notified when the response is based solely on the AI's internal knowledge.

### **5. Temporary In-Memory Vector Database**

#### **5.1 Temporary Storage of Uploaded Data**

* **As a user**, I want to upload documents and provide web addresses for the AI to perform data extraction and store the information temporarily in the in-memory vector database, so that I can use this additional context without modifying the permanent vector databases.
* **Acceptance Criteria**:
  + Uploaded and extracted data are stored in the in-memory vector database for the duration of the session.
  + Users are notified when data is successfully added to the in-memory database.

#### **5.2 Save and Load Temporary Vector Database**

* **As a user**, I want to save the in-memory vector database along with the chat history, so that I can reload it later and continue my dialogue with the AI Tutor.
* **Acceptance Criteria**:
  + Users can save the in-memory vector database and chat history as files.
  + Users can reload saved files into the in-memory vector database to resume sessions seamlessly.

### **6. Tools for Data Extraction**

* **As a user**, I want the app to use tools like Extract\_Document and Extract\_Web for document and web data extraction, so that I can efficiently add relevant content to the vector databases.
* **Acceptance Criteria**:
  + The app integrates Extract\_Document and Extract\_Web tools for parsing content.
  + Users can choose the target vector database (general\_db, mcq\_db, or in-memory) for storing the extracted data.

### **7. User-Friendly Interface**

* **As a user**, I want a user-friendly interface with clear options for uploading files, entering URLs, resetting databases, and saving/loading sessions, so that I can interact with the AI Tutor effortlessly.
* **Acceptance Criteria**:
  + The interface includes intuitive buttons and controls for all key functionalities.
  + Error messages and confirmations are clear and concise for all actions.