Project Brief: AI Apps Challenge General Integrated Chatbot

Project Title: General AI Integrated Chatbot

Project Overview:

The **General AI Integrated Chatbot** is the twelfth and final project of the 2025 AI Apps challenge, designed as a **unified conversational interface** that ties together all previous applications. It provides a **chatbot-style UI** where users can ask general queries (with prompt modes such as *code*, *writing*, *Q&A*, etc.) and also request tasks that map directly to the functionality of the earlier 11 apps.

For example, a user could ask the chatbot to "summarise this document" (calling the **AI Doc Summariser**), "generate a cover letter" (calling the **AI Cover Letter Writer**), or "calculate work hours" (calling the **AI Work Hours Calculator**). The chatbot acts as a **central hub**, routing queries to the appropriate app functions and returning results seamlessly.

Objectives:

1. Build a chatbot interface capable of handling general queries and specialised tasks.
2. Implement prompt modes (e.g. code, writing, Q&A) to tailor responses.
3. Integrate all 11 previous apps so their functions can be called directly from the chatbot.
4. Provide a clean directory structure with the chatbot in the root and helper functions in the 12.December_ai-chatbot folder.
5. Ensure scalable design, allowing future apps or functions to be added easily.
6. Deliver a capstone project that demonstrates the cumulative power of the AI Apps Challenge.

Key Features:

1. **Chatbot Interface:** Conversational UI for general queries.
2. **Prompt Modes:** Switch between *code*, *writing*, *Q&A*, and other tailored modes.
3. **App Integration:** Calls functions from all 11 previous apps.
4. **Unified Workflow:** Users can access any app's functionality without leaving the chatbot.
5. **Directory Structure:** Root app + December app folder for helper functions and chatbot logic.
6. **Extensible Design:** Easy to add new apps or prompt modes in the future.

Technical Specifications:

1. **Programming Language:** Python
2. **Frameworks and Libraries:**
    - Flask: Web application framework.
    - PyPDF2 + python-docx: For PDF/Word Doc handling and text extraction.
    - Gemini: For general query handling and prompt modes.

3. **Database:** SQLlite for storing users, chatbot sessions and mode preferences.
4. **Deployment:** Docker containerization and cloud service deployment (e.g., AWS or Azure).
5. **Version Control:** Git for source code management.

Expected Outcomes:

1. A capstone chatbot that unifies all 11 apps into one interface.
2. A flexible system that can handle both general queries and specialised tasks.
3. A demonstration of how modular AI apps can be integrated into a single workflow.
4. A polished final project that showcases the full scope of the AI Apps Challenge.

Risks and Mitigations:

1. **Integration Complexity:** Use modular imports and consistent function signatures.
2. **Performance Bottlenecks:** Optimise routing logic and cache results where possible.
3. **Mode Confusion:** Provide clear UI indicators for active prompt mode.
4. **Directory Management:** Keep helper functions isolated in December folder to avoid clutter.