Project Brief: AI Powered Work Hours Calculator

Project Title: AI Powered Work Hours Calculator

Project Overview:

The AI Powered Work Hours Calculator aims to provide users with a flexible and intuitive tool to calculate their work hours. Users can input their work hours in any format through a text box. The application will process the input using an AI chatbot (likely utilising the Gemini API, but other APIs can be considered) to provide a detailed breakdown of work hours, including total hours worked and any overtime or undertime. Additionally, the application will feature a "calculator" section that performs time-based arithmetic operations, making it easy for users to calculate differences between hours.

Objectives:

1. Develop a user-friendly interface for inputting work hours in various formats.
2. Implement an AI chatbot to process and interpret user inputs.
3. Provide a detailed breakdown of work hours, including total hours, overtime, and undertime.
4. Create a time-based calculator for performing arithmetic operations on hours.
5. Ensure the system supports flexible and varied user inputs efficiently. .

Key Features:

1. **Text Box for Work Hours:** Allow users to input their work hours in any format, such as natural language descriptions or structured formats.
2. **AI Chatbot:** Utilize an AI chatbot (Gemini API or other suitable APIs) to process and interpret user inputs.
3. **Work Hours Breakdown:** Provide a comprehensive breakdown of total hours worked, overtime, and undertime across all days mentioned.
4. **Calculator Section:** Implement a time-based calculator that performs operations on hours, such as calculating the difference between two times.

Technical Specifications:

1. **Programming Language:** Python
2. **Frameworks and Libraries:**
    - Flask or Django for the web application
    - NLTK or SpaCy for natural language processing
    - Dateutil for parsing date and time information
    - Gemini API or other suitable APIs for AI chatbot functionality

3. **Database:** SQLlite for storing user inputs and results
4. **Deployment:** Docker for containerisation and deployment on cloud services like AWS or Azure
5. **Version Control:** Git for source code management

Expected Outcomes:

- A fully functional AI Version of a Work Hours Calculator application.
- Improved accuracy and flexibility in calculating work hours and identifying overtime/undertime.
- Enhanced user experience through an intuitive and versatile interface.

Risks and Mitigations:

1. **Data Privacy:** Ensure all user data is securely stored and accessed. Implement encryption and access control measures.
2. **Performance:** Optimize the AI chatbot and parsing logic to handle varied user inputs efficiently.
3. **Integration Challenges:** Test the integration with different APIs and ensure seamless functionality.