Project Report

Development of AI Chemist App

Binil Simon

Google Cloud Generative AI

03/09/2024

Siri Ma'am

SmartInternz

Index

AI Chemist: Pioneering the Future of Chemical Science with Gemini Vision Pro

2.Category

Generative AI

3. Skills Required

Python: For backend development, API integration, and processing data inputs.

Deep Learning: To utilize the Gemini Pro AI model for generating chemical solutions and experimental recommendations.

Streamlit: For building the user interface and deploying the web-based application.

4. Abstract

AI Chemist is a pioneering mobile application designed to provide personalized chemical solutions and experimental recommendations using the advanced Gemini Pro AI model. The application processes user input, laboratory conditions, and research objectives to generate customized experimental designs, chemical synthesis pathways, and comprehensive data analysis. Through intelligent, data-driven support, AI Chemist aims to significantly enhance research efficiency and drive innovation in chemical science, empowering researchers with tailored insights and solutions for complex chemical problems

5.Project Description

Leveraging the power of the Gemini Pro AI model, this state-of-the-art mobile application delivers personalized chemical solutions and experimental strategies tailored to the unique needs of researchers. It utilizes advanced AI algorithms to analyze user inputs, evaluate laboratory conditions, and align with precise research goals. By generating optimized experimental designs, custom chemical synthesis pathways, and actionable data insights, the application enhances research efficiency and fosters innovation in chemical science. Through intelligent, data-driven guidance, it empowers scientists to accelerate discoveries and make informed decisions, driving breakthroughs in complex chemical research.

6. Project Flow

The project involves several key steps for delivering personalized solutions:

User Input Interaction:

Users interact with the app's user interface (UI) to input data relevant to their experiments.

Data Transmission to Backend:

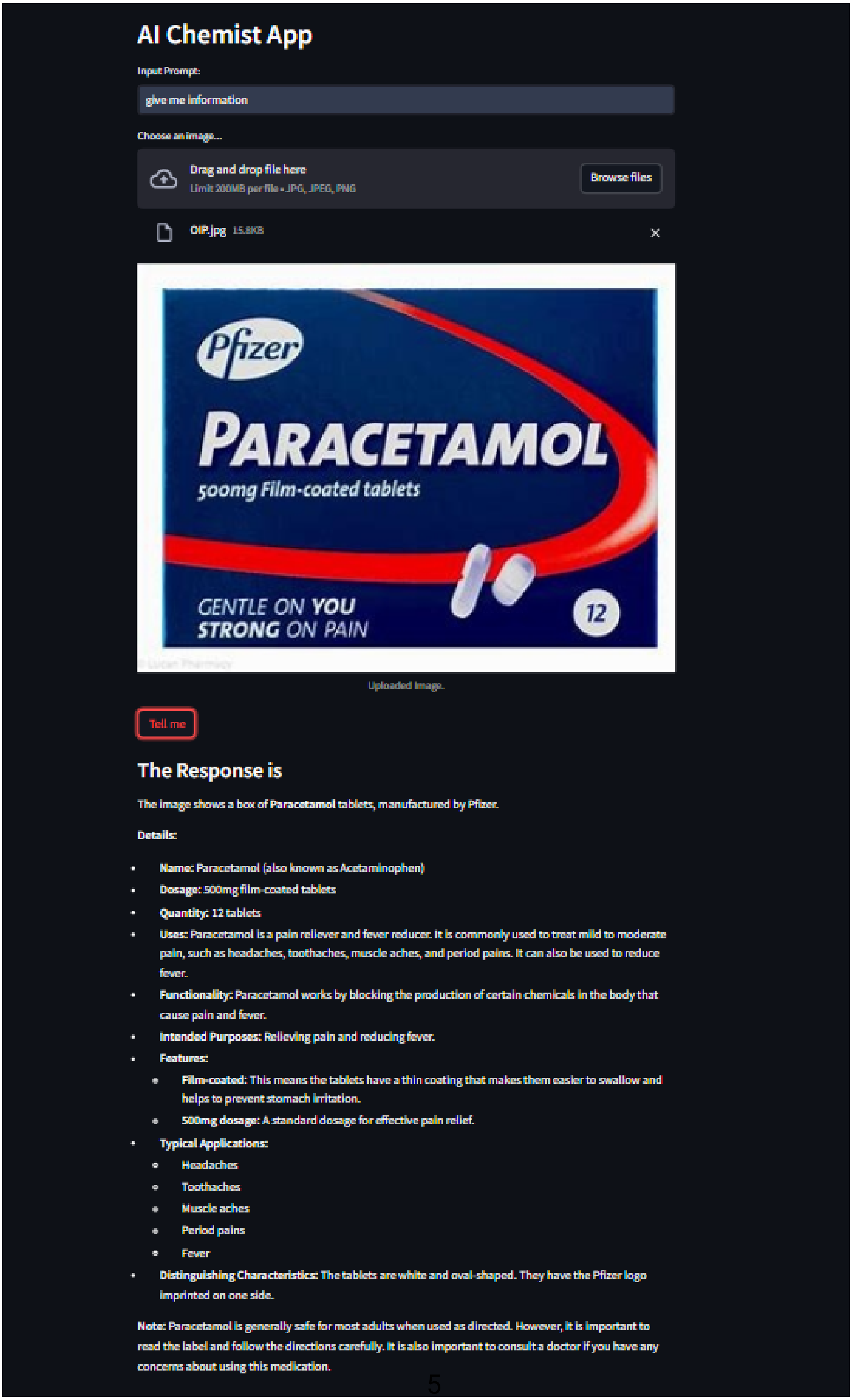
The input from the user is transmitted to the backend using Google API keys.
API Call to Gemini Pro:

An API call is made to the Gemini Pro pre-trained model to process the user input.
Output Generation:

The Gemini Pro model processes the input and generates relevant output (chemical solutions, synthesis routes, etc.).

Result Formatting:

The generated results are returned to the frontend,formatted,and displayed to user.



AI Chemist App

Input Prompt:

give me information

Choose an image...

Uploaded image.

The Response is

The image shows a box of **Paracetamol** tablets, manufactured by Pfizer.

Details:

- **Name:** Paracetamol (also known as Acetaminophen)
- **Dosage:** 500mg film-coated tablets
- **Quantity:** 12 tablets
- **Uses:** Paracetamol is a pain reliever and fever reducer. It is commonly used to treat mild to moderate pain, such as headaches, toothaches, muscle aches, and period pains. It can also be used to reduce fever.
- **Functionality:** Paracetamol works by blocking the production of certain chemicals in the body that cause pain and fever.
- **Intended Purposes:** Relieving pain and reducing fever.
- **Features:**
  - **Film-coated:** This means the tablets have a thin coating that makes them easier to swallow and helps to prevent stomach irritation.
  - **500mg dosage:** A standard dosage for effective pain relief.
- **Typical Applications:**
  - Headaches
  - Toothaches
  - Muscle aches
  - Period pains
  - Fever
- **Distinguishing Characteristics:** The tablets are white and oval-shaped. They have the Pfizer logo imprinted on one side.

Note: Paracetamol is generally safe for most adults when used as directed. However, it is important to read the label and follow the directions carefully. It is also important to consult a doctor if you have any concerns about using this medication.

Steps to Complete the Project

1. Requirements Specification
    - Create a requirements.txt file:

2.List all required libraries.
    - Install required libraries:
    - Ensure necessary libraries for AI, data processing, and UI integration are installed.

3. Initialization of Google API Key
    - Generate Google API Key:
    - Create an API key through Google Cloud.
    - Initialize Google API Key:
    - Securely store the API key and initialize it in the backend code.

4. Interfacing with Pre-trained Model
    - Load Gemini Pro Model:
    - Load the Gemini Pro pre-trained AI model for processing.
    - Implement Gemini Response Function:
    - Create a function that sends user input to the model and retrieves responses.

5.Read PDF Content:
    - Implement functionality to read and extract relevant content from

PDF documents.

6.Write a Prompt for the Model:
    - Develop structured prompts for sending input to the Gemini model for experiment recommendations.

7. Model Deployment
    - Web Framework Integration:

- Integrate the AI Chemist app with Streamlit or another web framework.
- Host the Application:
- Deploy the application on a suitable platform (e.g., Heroku, AWS) for user access.

7. Project Structure

- Images folder:
  Stores all images used in the user interface.
- .env file:
  Secures sensitive information such as the Google API key.
- app.py:
  The main application file, combining both the model logic and the Streamlit UI.
- requirements.txt:
  Lists the libraries necessary for running the project, such as streamlit, tensorflow, pandas, etc.

8. Best Practices

1. File Organization

Modular Structure: Organize the project into logical modules, separating concerns such as data processing, model handling, API calls, and frontend UI components. This makes the codebase easier to maintain, debug, and extend.

Clear Naming Conventions: Use descriptive and consistent names for files, functions, and variables. Follow widely accepted conventions like PEP 8 for Python to ensure readability and consistency.

Directory Layout:

Create separate folders for core components such as models, scripts, static (for images, CSS, JS files), templates (for UI), and utils (for helper functions).

Include a README.md for documentation and explanation of each file's role.

Use a dedicated logs/ directory to store application and debugging logs.

2. Version Control and Collaboration

Git for Version Control: Implement Git for tracking changes in the codebase. Regularly commit changes with clear, concise messages that describe the updates or fixes.

Branching Strategy: Use a branching model such as Git Flow to manage development. Create separate branches for features, bug fixes, and releases to keep the main branch stable.

Code Reviews: Set up code review processes on platforms like GitHub. Collaborators can review, comment, and suggest improvements, ensuring code quality and adherence to standards.

Pull Requests and Merging: Ensure all changes are integrated through pull requests (PRs) rather than direct commits to the main branch. This allows for peer review and discussion before merging changes.

3. Dependency Management

requirements.txt: Maintain a requirements.txt file for listing all dependencies needed for the project. Ensure that specific versions of libraries are mentioned to avoid compatibility issues.

Virtual Environments: Use virtual environments like venv or conda to isolate dependencies and avoid conflicts between different projects.

4. Documentation

In-code Documentation: Use comments and docstrings to document the functionality of functions and classes. This will help future developers (and yourself) understand the code better.

README and Setup Instructions: Provide a clear README.md with installation steps, usage guidelines, and explanations of how to set up and run the project. Include examples of API calls or UI interactions for clarity.

5. Testing and Debugging

Unit Testing: Implement unit tests for key functions and modules. Automated testing ensures that code changes do not introduce new bugs.

Logging and Error Handling: Integrate robust logging to track application performance and debug errors effectively. Ensure proper error handling, especially when interfacing with APIs and external services.

These best practices ensure the project is well-organized, easily maintainable, and scalable for future improvements or collaborations.

9. Conclusion

AI Chemist demonstrates the powerful integration of artificial intelligence in chemical sciences, utilizing the Gemini Pro model to deliver tailored, data-driven insights for experimental design. By automating complex analysis of user inputs and laboratory conditions, it optimizes chemical synthesis routes and experimental workflows, significantly improving research efficiency.

The user-friendly interface, powered by Streamlit, ensures accessibility for researchers, enabling them to harness AI without deep technical knowledge. This project sets a precedent for AI-driven innovation in chemical research, enhancing experimental accuracy, reducing time and resource consumption, and pushing the boundaries of scientific discovery.