Google Gemini Chatbot(AI-Chemist) - Project Documentation

1. Introduction

This project is a **chatbot powered by Google Gemini API**, deployed using **Streamlit**. The chatbot can process text and image inputs, analyze documents (PDFs), and generate responses based on user queries.

2. Features

- **Text-based query support**
- **Image-based query support**
- **PDF document analysis for reference**
- **Streamlit web app for interaction**
- **Deployed on Streamlit Cloud**

3. Technologies Used

- **Python 3.11**
- **Streamlit** (Frontend UI)
- **Google Gemini API** (LLM backend)
- **PyMuPDF (fitz)** (For PDF processing)
- **OpenCV** (For image processing)
- **Pillow (PIL)** (Image handling)

4. Installation & Setup

4.1 Local Setup

1. Clone the repository:
2. `git clone https://github.com/rashidpatel04/AI-Chemist`
3. Create a virtual environment and activate it:

4. `python -m venv venv`
   `source venv/bin/activate  # On Windows: venv\Scripts\activate`
5. Install dependencies:
6. `pip install -r requirements.txt`
7. Set the API Key:
8. `export GOOGLE_API_KEY='your-api-key'  # On Windows: set GOOGLE_API_KEY="your-api-key"`
9. Run the Streamlit app:
10. `streamlit run app.py`

4.2 Deployment on Streamlit Cloud

1. Push the project to GitHub.
2. Go to **Streamlit Community Cloud**.
3. Click **Deploy an app** → Select your repository.
4. Set **GOOGLE_API_KEY** in Streamlit secrets.
5. Click **Deploy**.

5. Project Structure

```
 google_gemini_chatbot: AI-Chemist
├── app.py            # Main Streamlit app
├── requirements.txt  # Python dependencies
├── README.md         # Project documentation
├── AIC.png           # Images
```

6. Code Overview

6.1 `app.py` (Main App Logic)

- Initializes Streamlit UI.
- Accepts user input (text, image, or PDF).
- Calls `get_gemini_response()` to process input.
- Displays chatbot responses.

7. API Integration

Uses **Google Gemini API** to generate responses:

```
import google.generativeai as genai

def get_gemini_response(input_text, pdf_content=None, image=None):
    genai.configure(api_key=os.getenv("GOOGLE_API_KEY"))
    response = model.generate_content([input_text, image] if image
else input_text)
    return response.text if response else "No response received."
```

8. Debugging & Common Issues

8.1 Stuck on "Analyzing the problem..."

 Check logs on Streamlit Cloud → Fix API key issues.

8.2 No Response from API

 Increase API timeout in `get_gemini_response()`:

```
response = model.generate_content(input_text, timeout=60)
```

8.3 No Internet Access in Deployment

 Add debug check in `app.py`:

```
import requests
st.write("Internet Test:",
requests.get("https://www.google.com").status_code)
```

9. Future Enhancements

- **Add voice input support**
- **Improve UI with better response formatting**
- **Store conversation history**

10. Conclusion

This project successfully integrates **Google Gemini API** into a **Streamlit chatbot**, enabling AI-powered text, image, and document processing.

 Developed & Maintained by Rashid Patel