**Documentation for Gemini-Decode Application**

**Overview**

The **Gemini Decode** application is a Streamlit-based tool designed to extract information from multilingual documents. Utilizing the Gemini Pro AI model, it allows users to upload document images and ask questions regarding their content. This application efficiently bridges language barriers, making information retrieval straightforward and accessible.

**For Creating this we need three files which are as follows:**  
A. .env   
B. app.py

c. requirements.txt  
  
In .env file you need to write your GOOGLE\_API\_KEY which can be access from the link below  
[LINK TO GET GOOGLE\_API\_KEY.](https://ai.google.dev/gemini-api/docs/api-key)

In requirement.txt file we’ll write all the necessary library of python which are as follows:

streamlit

google-generativeai

python-dotenv

langchain

PyPDF2

chromadb

faiss-cpu

streamlit-extras

Pillow

Along with that also ensure that you have python and pip instal in your system, if not then installed them first.

**For installing all the library need to open terminal and do as below:-  
Run the command: pip install -r requirements.txt**

**After this just just save the both the file .env and requirements.txt.**

**In .env file write the api key as given below:-  
GOOGLE\_API\_KEY=”your key value from the link ”**

**NOW LET’S START THE CODE STRUCTURE OF OUR GEMINI-DECODE APPLICATION:-  
  
IMPORT ALL IMPORTANT LIBRARY:**  
from dotenv import load\_dotenv

load\_dotenv()  # load all the **environment variables:**

import streamlit as st

import os

import google.generativeai as genai

from PIL import Image

genai.configure(api\_key=os.getenv("GOOGLE\_API\_KEY"))

**## Function to load Google Gemini Pro Vision API And get response:**

def get\_gemini\_response(input, image, prompt):

    model = genai.GenerativeModel('gemini-1.5-flash')  # Updated model name

    response = model.generate\_content([input, image[0], prompt])

    return response.text

**def input\_image\_setup(uploaded\_file):**

**# Check if a file has been uploaded**

    if uploaded\_file is not None:

        # Read the file into bytes

        bytes\_data = uploaded\_file.getvalue()

        image\_parts = [

            {

                "mime\_type": uploaded\_file.type,  # Get the mime type of the uploaded file

                "data": bytes\_data

            }

        ]

        return image\_parts

    else:

        raise FileNotFoundError("No file uploaded")

## Initialize our streamlit app:

input\_prompt = """

You are an expert pharmaceutical/Chemist where you need to see the tablets from the image,

and also provide the details of every drug/tablet items with the below format:

Examine the image carefully and identify the tablets depicted.

Describe the uses and functionalities of each tablet shown in the image.

Provide information on the intended purposes, features, and typical applications of the tablets.

If possible, include any notable specifications or distinguishing characteristics of each tablet.

Ensure clarity and conciseness in your descriptions, focusing on key details and distinguishing factors.

"""

**## Initialize Streamlit app:**

st.set\_page\_config(page\_title="AI Chemist App")

st.header("AI Chemist App")

input\_text = st.text\_input("Input Prompt: ", key="input")

uploaded\_file = st.file\_uploader("Choose an image ...", type=["jpg", "jpeg", "png"])

image = ""

if uploaded\_file is not None:

    image = Image.open(uploaded\_file)

    st.image(image, caption="Uploaded Image.", use\_column\_width=True)

submit = st.button("Tell me")

**## If submit button is clicked if submit:**

    image\_data = input\_image\_setup(uploaded\_file)

    response = get\_gemini\_response(input\_prompt, image\_data, input\_text)

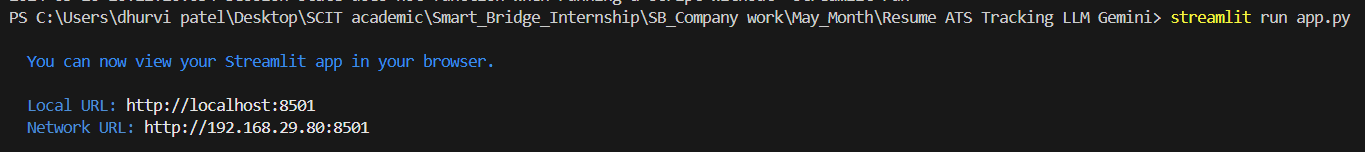
    st.subheader("The Response is")

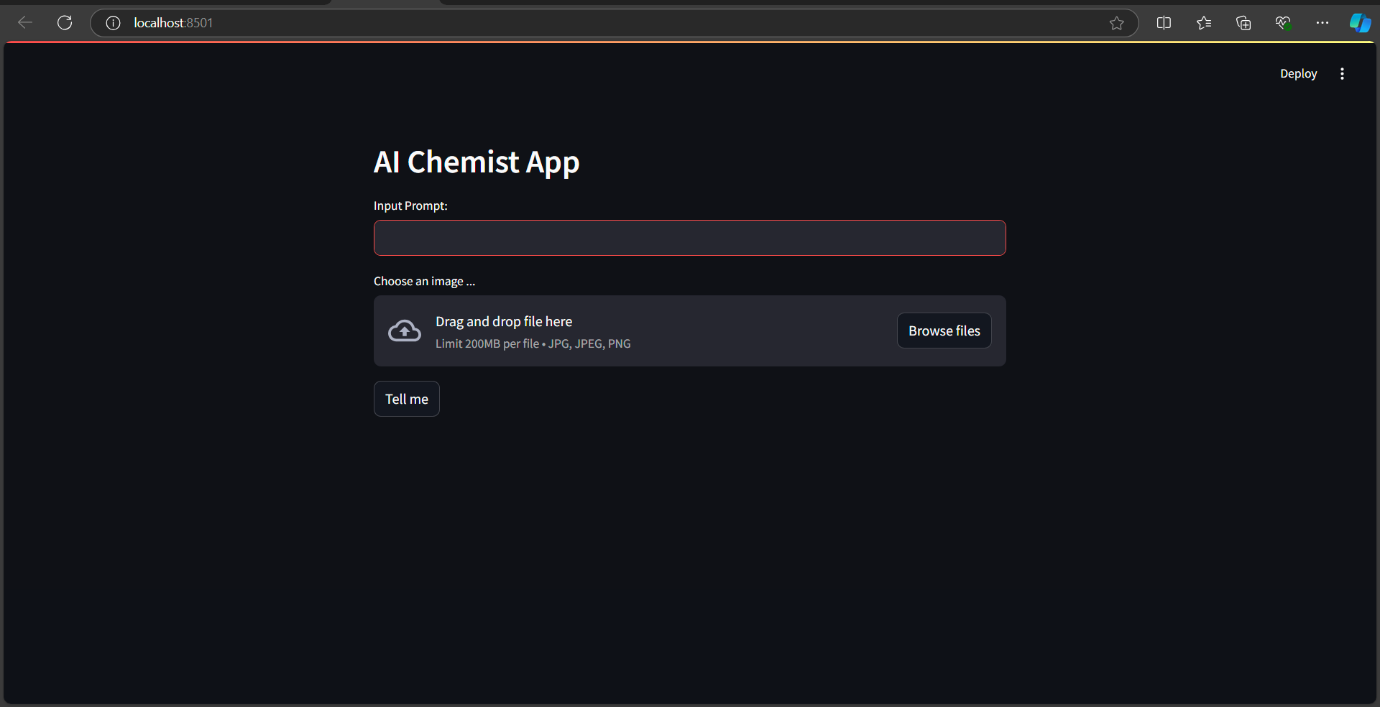
    st.write(response)

**Running the Application**

streamlit run app.py

NOW AS THE APPLICATION START IT WILL BE ON THE LOCALHOST OR LOCALSERVER   
YOU WILL GET THESE TWO LINK IN THE TERMINAL:

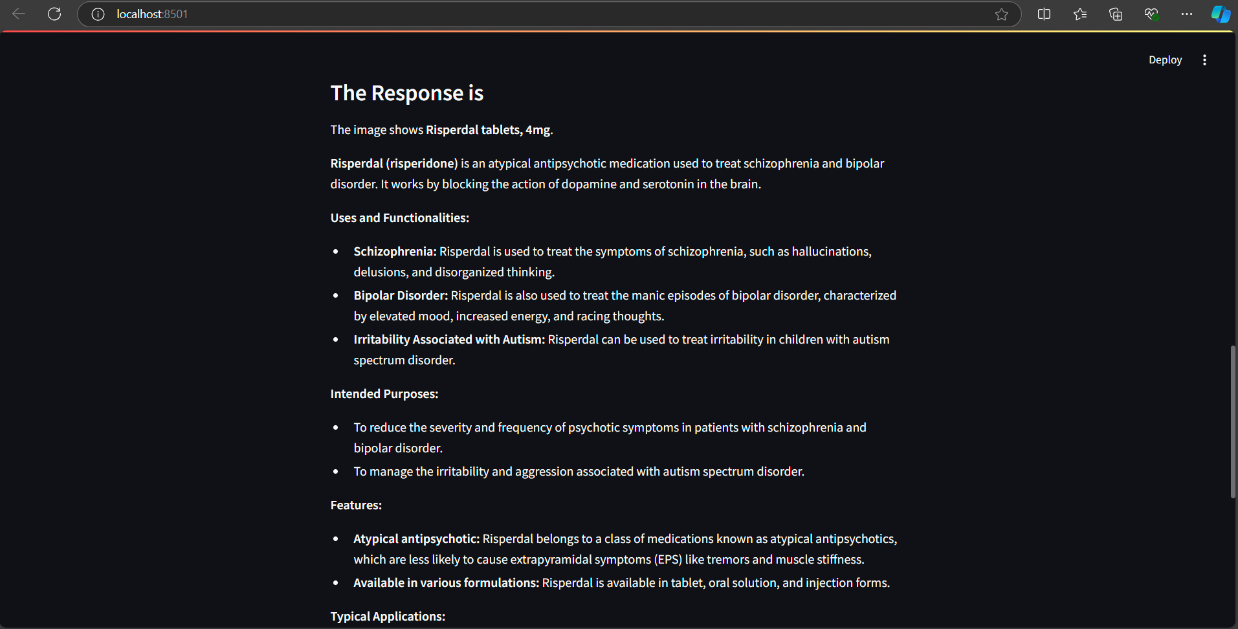


**On clicking the Local URL, you will be able to use our Gemini Decode Application  
It will look like this:**  


**Now Choose an image as we will choose two images in which one is of Risperdal\_tablets and another image is of Stimulation 60 mg Then in input field write : What is the use of this medicine ? For both image use same input and then you will get output after clicking on submit button**

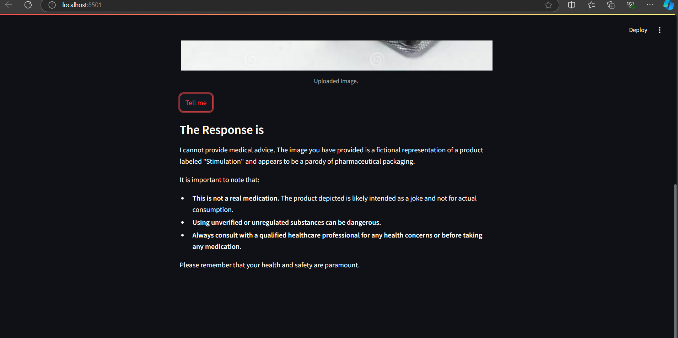
A. Risperdal\_tablets  
 *The input is*   
 

*And its output will be as :*



b. stimulation  
 The input is :



The output will be as :   
 

And that’s how this application is work .  
  
**Conclusion:**

**The Gemini Decode application provides an intuitive interface for extracting information from documents using AI. With its ability to handle multilingual content, it is an invaluable tool for enhancing productivity and decision-making.**