ASSIGNMENT-3

Gen AI Based Medicine Assistant

Code:

```
import streamlit as st
import openai
import requests
from io import BytesIO
from deep_translator import GoogleTranslator
import base64
import re
import os
from fpdf import FPDF
from gtts import gTTS
from docx import Document
from PIL import Image
import pytesseract


st.set_page_config(page_title=" 💊 GPT-3.5 Medicine Assistant", layout="centered")


st.markdown("""
<style>
    .main {
        background-color: #f4f6f9;
        font-family: 'Segoe UI', sans-serif;
    }
    .title {
        color: #2c3e50;
        text-align: center;
        font-size: 32px;
```

```
    }
    .highlight {
        background-color: #fffbcc;
        padding: 4px 8px;
        border-radius: 4px;
        font-weight: bold;
    }
</style>
""", unsafe_allow_html=True)

st.markdown("<div class='title'>💊 GPT-3.5 Turbo Medicine Assistant</div>", unsafe_allow_html=True)
st.markdown("Analyze a medicine strip, get human-like explanation, highlight text, translate, and listen!")

openai_api = st.text_input("🔐 OpenAI API Key", type="password")
user_lang = st.text_input("🌐 Translate to Language (e.g. en, hi, kn)", value="en")

input_type = st.radio("📷 Choose input method:", ["📷 Use Camera", "📤 Upload Image", "🌐 Paste Image URL"])

image_bytes = None
if input_type == "📷 Use Camera":
    camera_input = st.camera_input("Take a photo of the medicine strip")
    if camera_input:
        image_bytes = camera_input.read()
elif input_type == "📤 Upload Image":
    uploaded = st.file_uploader("Upload Medicine Strip", type=["jpg", "jpeg", "png"])
    if uploaded:
        image_bytes = uploaded.read()
elif input_type == "🌐 Paste Image URL":
    url = st.text_input("Paste image URL here:")
```

```
    if url:
        response = requests.get(url)
        if response.status_code == 200:
            image_bytes = response.content
            st.image(BytesIO(image_bytes), caption="Image from URL")

def extract_text_from_image(image_bytes):
    image = Image.open(BytesIO(image_bytes))
    return pytesseract.image_to_string(image)

def query_gpt35(text):
    openai.api_key = openai_api
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=[
            {"role": "user", "content": f"""Analyze the following extracted text from a medicine strip:

{text}

Step 1: Extract name, expiry, usage, dosage, food instructions, warnings in a markdown table.
Step 2: Write a bullet-point explanation in plain language. Do NOT repeat the table."""}
        ],
        max_tokens=800
    )
    return response['choices'][0]['message']['content'].strip()

def highlight_keywords(text, keywords):
    for word in keywords:
        text = re.sub(f"\\b({re.escape(word)})\\b", r"<span class='highlight'>\1</span>", text,
flags=re.IGNORECASE)
    return text
```

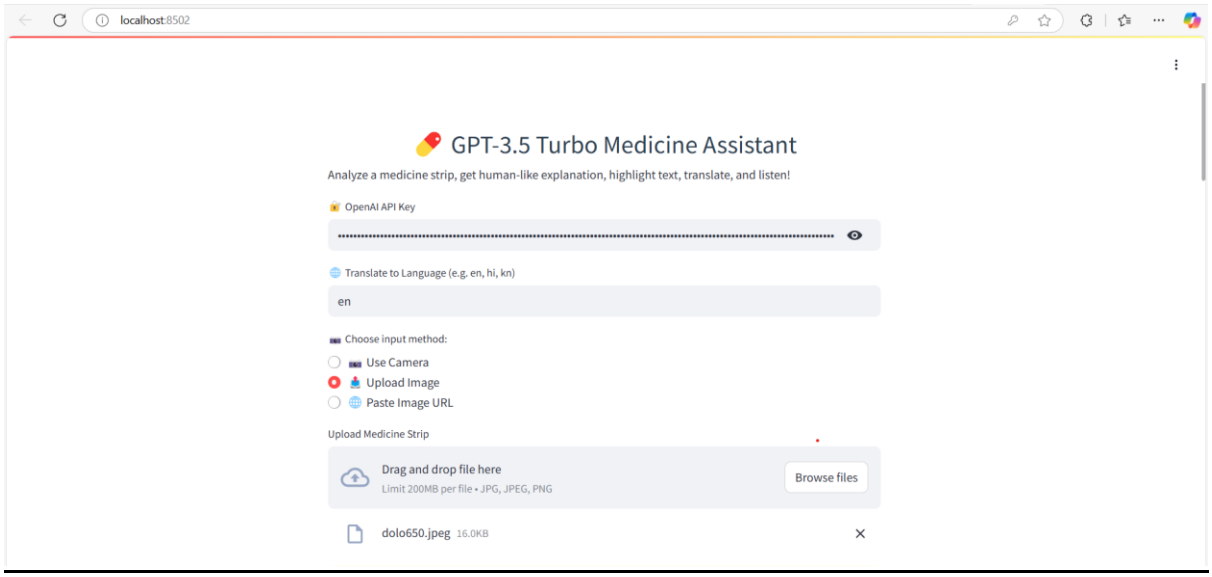
```
def create_pdf(text):
    pdf = FPDF()
    pdf.add_page()
    pdf.set_font("Arial", size=12)
    for line in text.split("\n"):
        pdf.multi_cell(0, 10, line)
    pdf_path = "/tmp/medicine_info.pdf"
    pdf.output(pdf_path)
    return pdf_path

def create_word_doc(text):
    doc = Document()
    doc.add_heading("Medicine Information", level=1)
    for line in text.split("\n"):
        doc.add_paragraph(line)
    word_path = "/tmp/medicine_info.docx"
    doc.save(word_path)
    return word_path

def speak_with_gtts(text, lang):
    tts = gTTS(text=text, lang=lang)
    audio = BytesIO()
    tts.write_to_fp(audio)
    audio.seek(0)
    return audio

if image_bytes and openai_api:
    st.image(BytesIO(image_bytes), caption="Selected Image", use_column_width=True)

    if st.button("🤖 Analyze with GPT-3.5 Turbo"):
```

```
    with st.spinner("Extracting text with OCR and analyzing with GPT-3.5..."):
        extracted_text = extract_text_from_image(image_bytes)
        gpt_output = query_gpt35(extracted_text)
        table_part, summary_part = gpt_output.split("\n\n", 1) if "\n\n" in gpt_output else (gpt_output, "")

        translated = GoogleTranslator(source="auto", target=user_lang).translate(summary_part)

        st.markdown("### 📋 Extracted Information")
        st.markdown(table_part)

        st.markdown("### ✍️ Highlight Important Terms")
        keyword_input = st.text_input("Enter keywords to highlight (comma-separated)", value="dosage,warning,side effect")
        keywords = [k.strip() for k in keyword_input.split(",") if k.strip()]
        highlighted = highlight_keywords(translated, keywords)
        st.markdown(f"<div>{highlighted}</div>", unsafe_allow_html=True)

        st.markdown("### 📎 Download Options")
        st.download_button("📜 Download as TXT", translated, file_name="medicine_info.txt")
        pdf_path = create_pdf(translated)
        with open(pdf_path, "rb") as pdf_file:
            st.download_button("📄 Download as PDF", data=pdf_file, file_name="medicine_info.pdf")
        word_path = create_word_doc(translated)
        with open(word_path, "rb") as word_file:
            st.download_button("📝 Download as Word (.docx)", data=word_file, file_name="medicine_info.docx")

        st.markdown("### 🔊 Spoken Explanation")
        audio_data = speak_with_gtts(translated, user_lang)
        st.audio(audio_data, format="audio/mp3")
```

Description:

1. **Streamlit UI**: Creates a clean web interface titled "GPT-3.5 Turbo Medicine Assistant".
2. **User Inputs**: Allows users to input OpenAI API key and choose a target translation language.
3. **Image Input Options**: Supports three image input methods — camera capture, file upload, or image URL.
4. **OCR Extraction**: Uses Tesseract OCR to extract text from the uploaded or captured medicine strip image.
5. **GPT-3.5 Analysis**: Sends the extracted text to OpenAI's GPT-3.5 Turbo to extract medicine info like name, expiry, usage, etc.
6. **Markdown Table + Summary**: GPT returns a markdown table of structured data and a simple explanation in plain English.
7. **Translation**: Translates the explanation into the user's selected language using GoogleTranslator.
8. **Keyword Highlighting**: Highlights user-specified keywords (e.g., dosage, warnings) in the translated text.
9. **Download Options**: Lets users download the result in TXT, PDF, and Word DOCX formats.
10. **Audio Output**: Converts the translated explanation into human-like audio using Google Text-to-Speech (gTTS) and plays it.

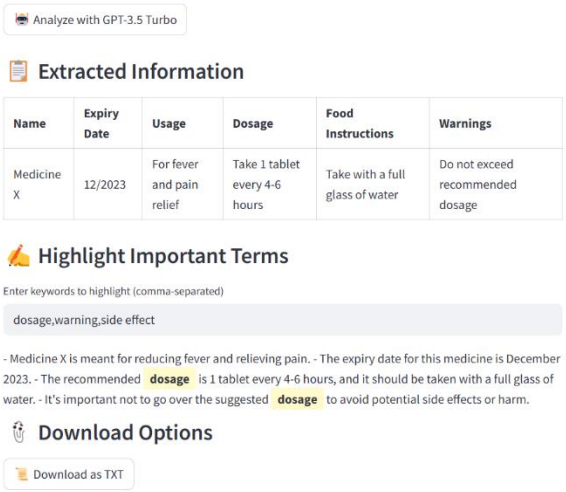
Output:


localhost:8502
GPT-3.5 Turbo Medicine Assistant
Analyze a medicine strip, get human-like explanation, highlight text, translate, and listen!
OpenAI API Key
Translate to Language (e.g. en, hi, kn)
en
Choose input method:
Use Camera
Upload Image
Paste Image URL
Upload Medicine Strip
Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG
Browse files
dolo650.jpeg 16.0KB


Deploy
Dolo-650
Paracetamol Tablets IP
Dolo-650®
डोलो-६५०
Each uncoated tablet contains :
Store in a dry & dark place,
at a temperature not exceeding 30°C.
Dosage : As directed by the Physician.
"Over dose may be injurious to Liver".

Deploy

Selected Image



Analyze with GPT-3.5 Turbo

Extracted Information

Name	Expiry Date	Usage	Dosage	Food Instructions	Warnings
Medicine X	12/2023	For fever and pain relief	Take 1 tablet every 4-6 hours	Take with a full glass of water	Do not exceed recommended dosage

Highlight Important Terms

Enter keywords to highlight (comma-separated)

dosage,warning,side effect

- Medicine X is meant for reducing fever and relieving pain. - The expiry date for this medicine is December 2023. - The recommended **dosage** is 1 tablet every 4-6 hours, and it should be taken with a full glass of water. - It's important not to go over the suggested **dosage** to avoid potential side effects or harm.

Download Options

Download as TXT