Gemini Image Analysis Application

yash.s.halwai

The Gemini Image Analysis Application is an end-to-end web application built using Streamlit, integrating Google's Gemini Pro model for generating responses based on image input and text prompts. This application allows users to upload an image and provide a textual prompt, leveraging advanced AI capabilities to analyze and generate content about the image.

Code Explanation:

1. Imports and Environment Setup:

```
from dotenv import load_dotenv

load_dotenv()  # take environment variables from .env.

import streamlit as st
import os
import pathlib
import textwrap
from PIL import Image

import google.generativeai as genai
```

- `dotenv` is imported to load environment variables from a `.env` file.
- `streamlit` is used as the main framework for building web application.
- Other standard libraries (`os`, `pathlib`, `textwrap`, `PIL.Image`) are imported for various functionalities.
- `google.generativeai` is imported specifically to interact with Google's Gemini Pro model.

2. Configure Google API Key:

```
os.getenv("GOOGLE_API_KEY")
genai.configure(api_key=os.getenv("GOOGLE_API_KEY"))
```

- Retrieves the Google API key from the environment variables stored in `.env`.
- Configures the Gemini Pro model with the obtained API key.

3. Function to Retrieve Gemini Model Response:

```
def get_gemini_response(input, image):
    model = genai.GenerativeModel('gemini-pro-vision')
    if input != "":
        response = model.generate_content([input, image])
    else:
        response = model.generate_content(image)
    return response.text
```

- Defines `get_gemini_response` function which:
  - Initializes the Gemini Pro Vision model.
  - Generates content based on provided input and/or image.
  - Returns the textual response generated by the model.

4. Streamlit Application Initialization:

```
st.set_page_config(page_title="Gemini Image Demo")
st.header("Gemini Application")
```

- Sets the Streamlit page configuration with the title "Gemini Image Demo".
- Displays a header "Gemini Application" at the beginning of the application.

5. User Input and Image Upload:

```
input = st.text_input("Input Prompt: ", key="input")
uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg", "png"])
```

- Provides a text input box for the user to enter a prompt (`input`).
  - Allows the user to upload an image (`uploaded_file`) which can be of types JPG, JPEG, or PNG.

6. Image Display:

```
image = ""
if uploaded_file is not None:
    image = Image.open(uploaded_file)
    st.image(image, caption="Uploaded Image.", use_column_width=True)
```

- Checks if an image file has been uploaded.

- Opens and displays the uploaded image using Streamlit's `st.image` function with a caption.

7. Interaction with the Model:

```
submit = st.button("Tell me about the image")

if submit:
    response = get_gemini_response(input, image)
    st.subheader("The Response is")
    st.write(response)
```

- Creates a button labeled "Tell me about the image".
- When the button is clicked (`submit` is `True`), call `get_gemini_response` with the provided input and image.
- Displays the generated response under a subheader "The Response is" using `st.subheader` and `st.write`.

Detailed Insight:

The Gemini Image Analysis Application leverages advanced AI capabilities to analyze images based on textual prompts provided by users. It integrates Google's Gemini Pro model, configured with an API key stored securely in a `.env` file. Users can upload images and enter prompts via a web interface built using Streamlit, enabling seamless interaction with sophisticated AI-powered image analysis. The application is designed to provide insights and information about uploaded images in a user-friendly and intuitive manner.

Connect With Me:

LinkedIn: https://in.linkedin.com/in/yash-sanjeev-halwai-198007203
GitHub: https://github.com/YashHalwai
CodeYAA: https://www.youtube.com/@codeyaa
Instagram: https://www.instagram.com/yash.s.halwai/?igsh=MzZrb2k2MjJ5ZnNp