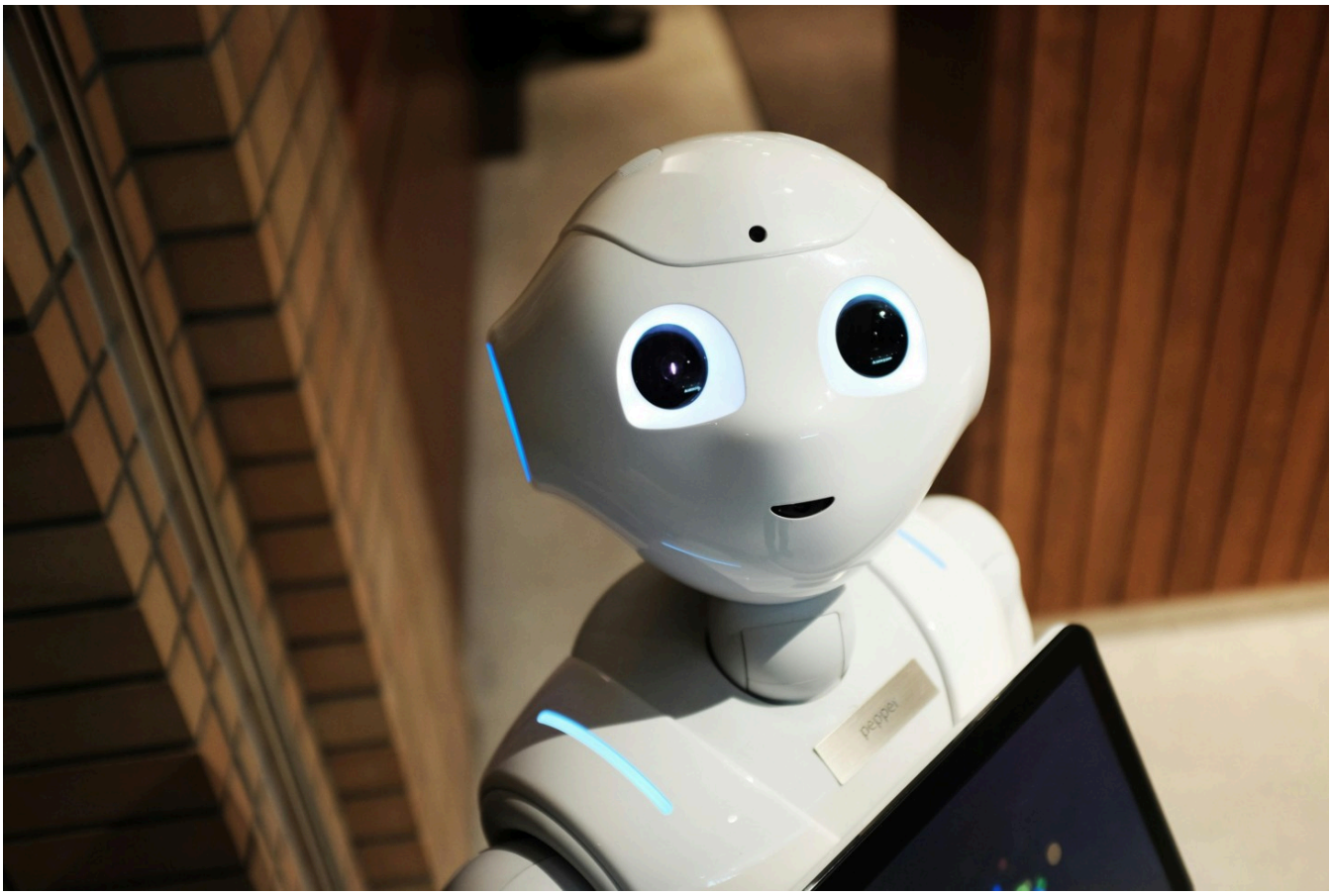
An AI *Code Reviewer*



Submit your Python code below, and I'll let you know if you are good to go!

Drop the code

```
"""Importing required packages"""

#Streamlit package
import streamlit as st
```

Submit Code

The code you provided is a good starting point for a Streamlit application that uses Google's Generative AI to review Python code. However, there are a few areas for improvement:

1. **Error Handling:** The code lacks error handling. If the Google AI API returns an error, the application will crash. You should add a `try...except` block to catch potential exceptions and display informative messages to the user.

2. **API Key Security:** Hardcoding the API key directly in the code is a security risk. It's best practice to store API keys in environment variables or a more secure configuration method.

3. **Input Validation:** The code doesn't validate the user's input. A user might enter something that's not Python code, causing the AI model to generate unexpected or erroneous results. Adding input validation will make the application more robust.

4. **Prompt Engineering:** The system prompt could be improved to be more concise and focused on the core task.

5. **Code Formatting:** The Streamlit app could benefit from slightly improved layout and styling.

Here's the improved code with the fixes and improvements mentioned above:

```
import streamlit as st
import google.generativeai as gai
import os
import sys

# Get API key from environment variable; provide a default for local development
api_key = os.environ.get("GOOGLE_GENERATIVE_AI_API_KEY")
if api_key is None:
    api_key = "YOUR_API_KEY"  # Replace with your API key for local testing.  Remo
    st.warning("Running with a hardcoded API Key.  Change this for production!")

gai.configure(api_key=api_key)

st.title("An :blue[AI] _Code Reviewer_")
st.image("Botty.jpg")  #Make sure Botty.jpg is in the same directory as your scrip

st.subheader("Submit your Python code below, and I'll let you know if you are good

user_prompt = st.text_area("Drop the code", placeholder="Paste your Python code he
button_click = st.button("Submit Code")

if button_click:
    if not user_prompt.strip():
        st.error("Please provide some Python code.")
    else:
        try:
            sys_prompt = """You are a helpful AI Python Code Reviewer. Analyze the
            model = gai.GenerativeModel(model_name="gemini-1.5-flash", system_inst
            response = model.generate_content(user_prompt)
            st.code(response.text, language="text") #Using st.code for better reaa
        except Exception as e:
            st.error(f"An error occurred: {e}")
```

Explanation of Changes:

- **API Key Management:** The API key is now retrieved from an environment variable. This is much safer than hardcoding it. A default is provided for local testing, but remember to replace it and use an environment variable in production.
- **Error Handling:** A `try...except` block handles potential errors from the API call and displays them to the user.
- **Input Validation:** A simple check is added to ensure that the user actually enters code before submitting.
- **Improved Prompt:** The system prompt is more concise and directly states the desired behavior.
- **Code Readability**: The response is displayed using `st.code` which improves readability by applying syntax highlighting to the AI's output.

Remember to replace `"YOUR_API_KEY"` with your actual API key only for local testing purposes. For production deployment, always use environment variables to store sensitive information. Make sure you also have the `Botty.jpg` image file in the same directory as your Streamlit script.