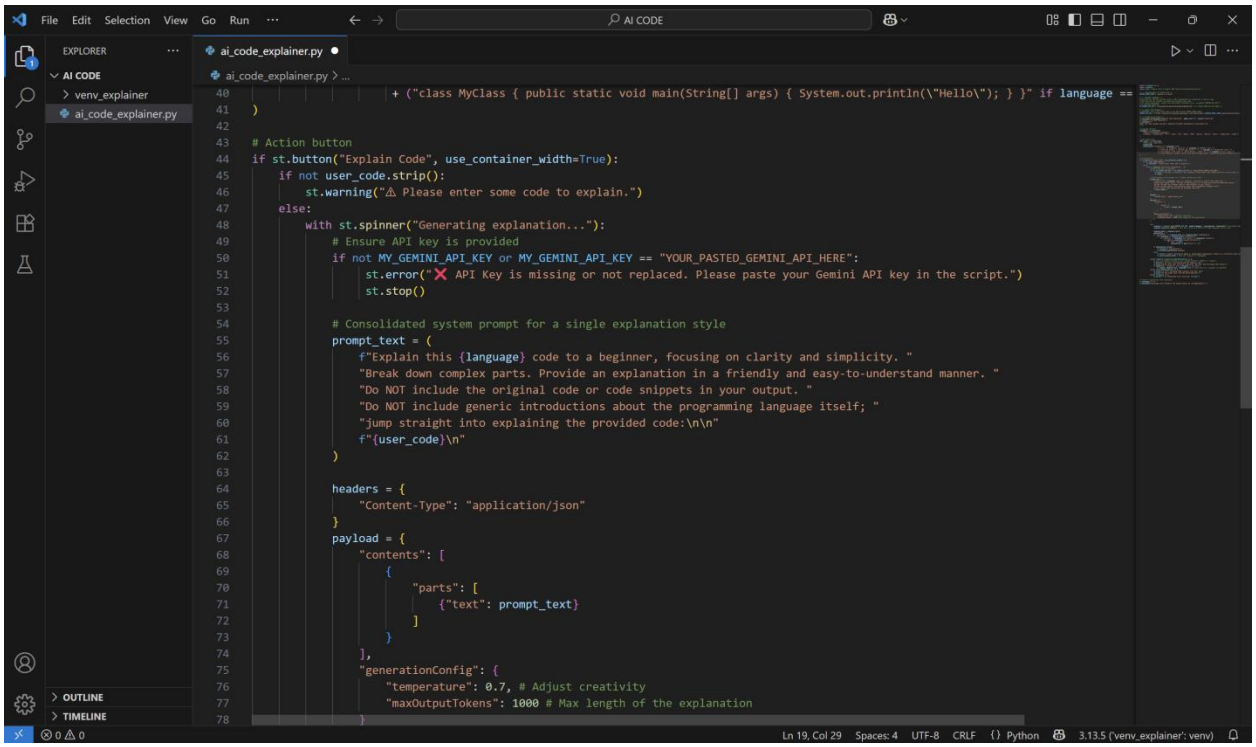
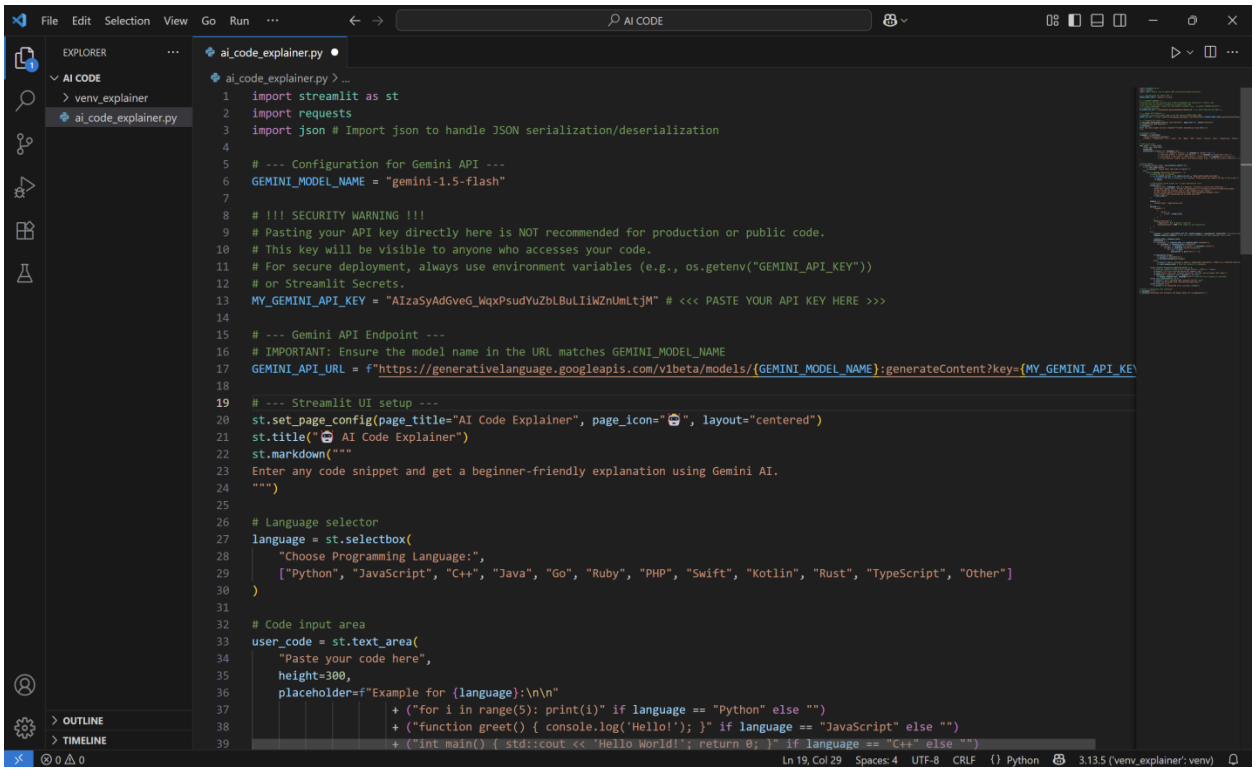
**Name** :- B.Tejaswini

**Clg Name :-** VIT (vellore)

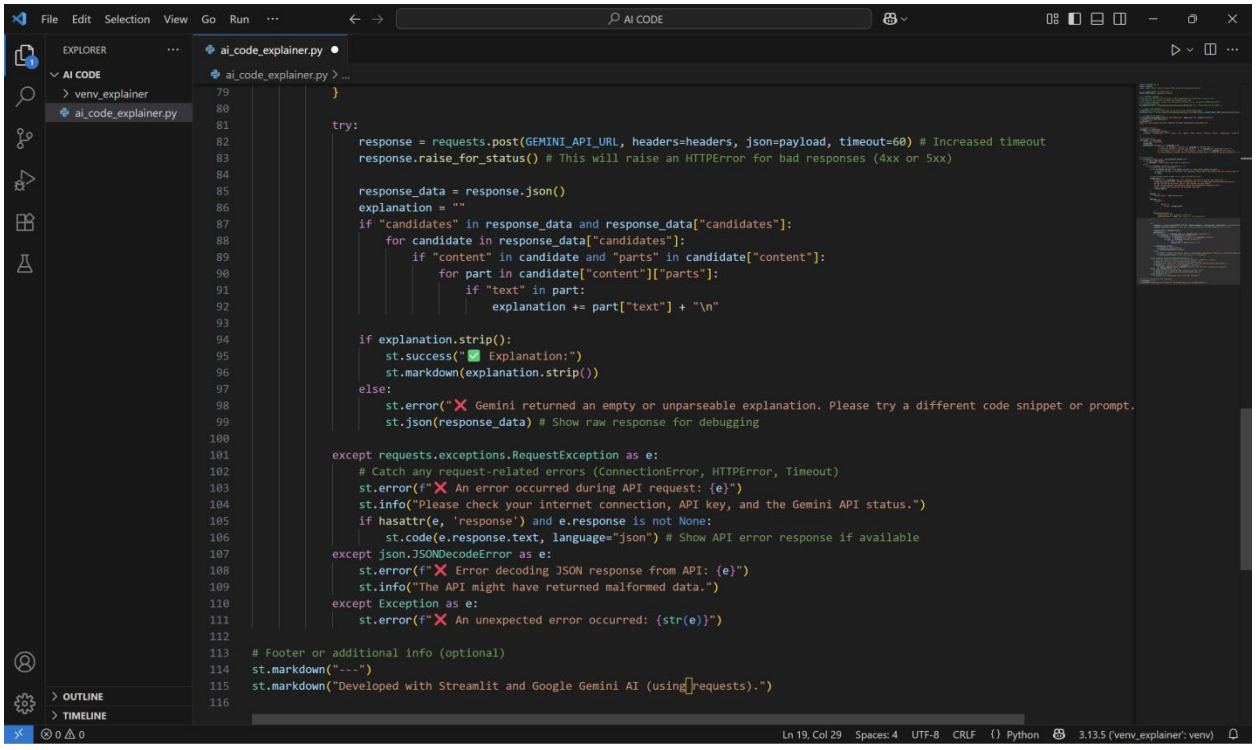
# AI Code Explainer

**Code :-**



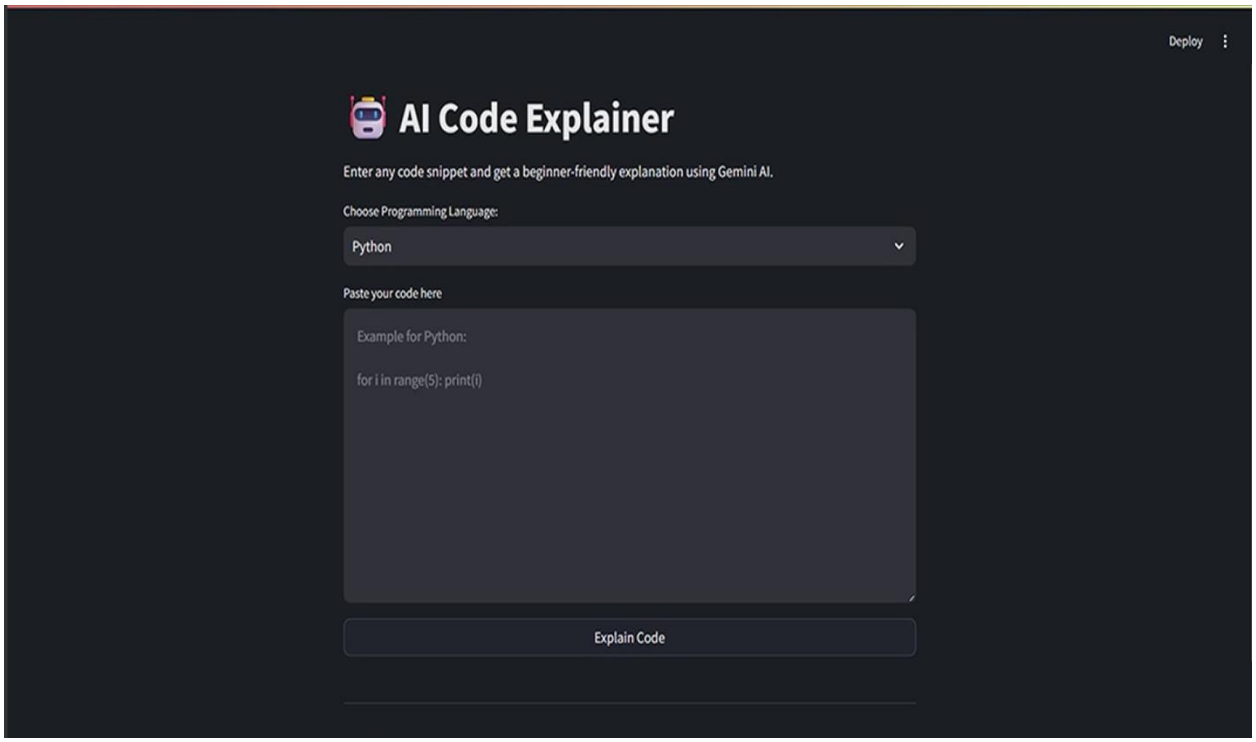


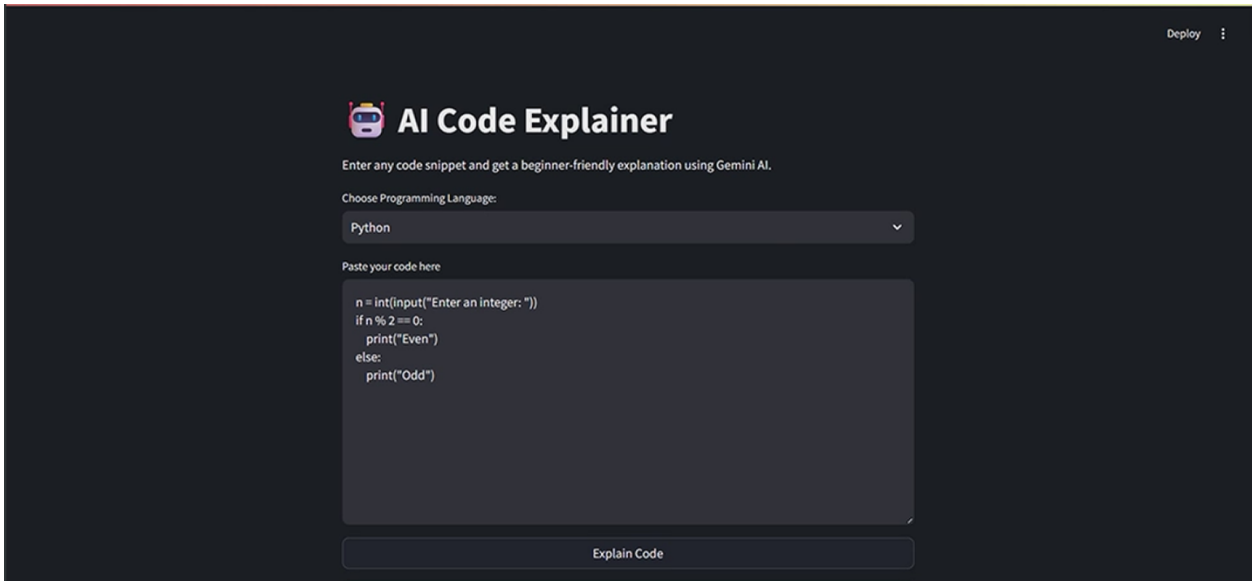
```python
import streamlit as st
import requests
import json # Import json to handle JSON serialization/deserialization

# --- Configuration for Gemini API ---
GEMINI_MODEL_NAME = "gemini-1.5-flash"

# !!! SECURITY WARNING !!!
# Pasting your API key directly here is NOT recommended for production or public code.
# This key will be visible to anyone who accesses your code.
# For secure deployment, always use environment variables (e.g., os.getenv("GEMINI_API_KEY"))
# or Streamlit Secrets.
MY_GEMINI_API_KEY = "AIzaSyAdGveG_WqxPsudYuZbLBuLIiWZnUmLtjM" # <<< PASTE YOUR API KEY HERE >>>

# --- Gemini API Endpoint ---
# IMPORTANT: Ensure the model name in the URL matches GEMINI_MODEL_NAME
GEMINI_API_URL = f"https://generativelanguage.googleapis.com/v1beta/models/{GEMINI_MODEL_NAME}:generateContent?key={MY_GEMINI_API_KE\

# --- Streamlit UI setup ---
st.set_page_config(page_title="AI Code Explainer", page_icon="🤖", layout="centered")
st.title("🤖 AI Code Explainer")
st.markdown("""
Enter any code snippet and get a beginner-friendly explanation using Gemini AI.
""")

# Language selector
language = st.selectbox(
    "Choose Programming Language:",
    ["Python", "JavaScript", "C++", "Java", "Go", "Ruby", "PHP", "Swift", "Kotlin", "Rust", "TypeScript", "Other"]
)

# Code input area
user_code = st.text_area(
    "Paste your code here",
    height=300,
    placeholder=f"Example for {language}:\n\n"
                + ("for i in range(5): print(i)" if language == "Python" else "")
                + ("function greet() { console.log('Hello!'); }" if language == "JavaScript" else "")
                + ("int main() { std::cout << 'Hello World!'; return 0; }" if language == "C++" else "")
```

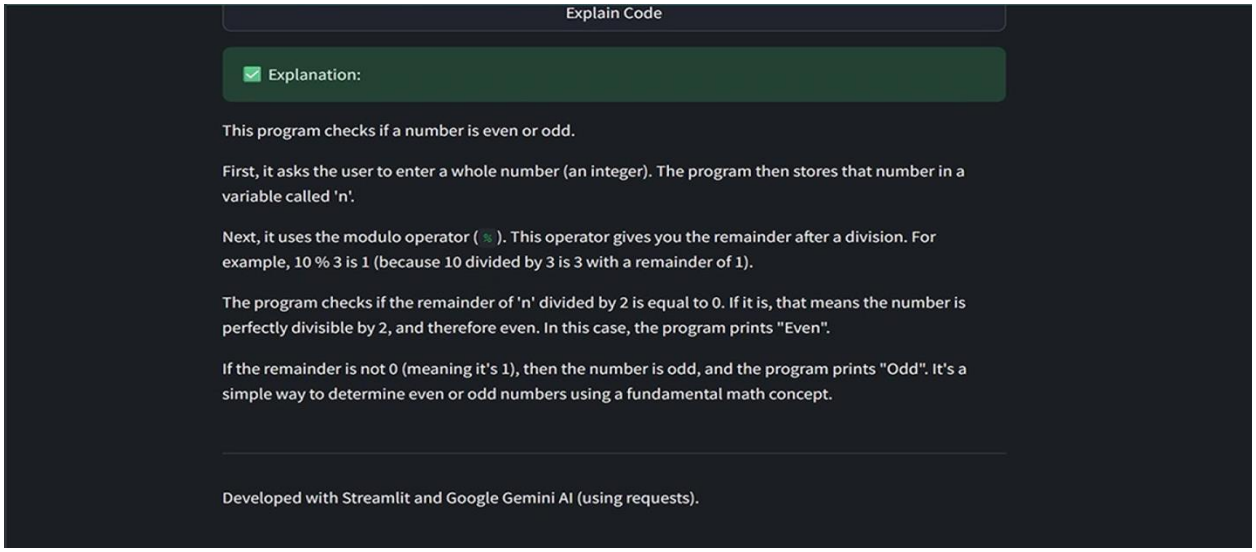


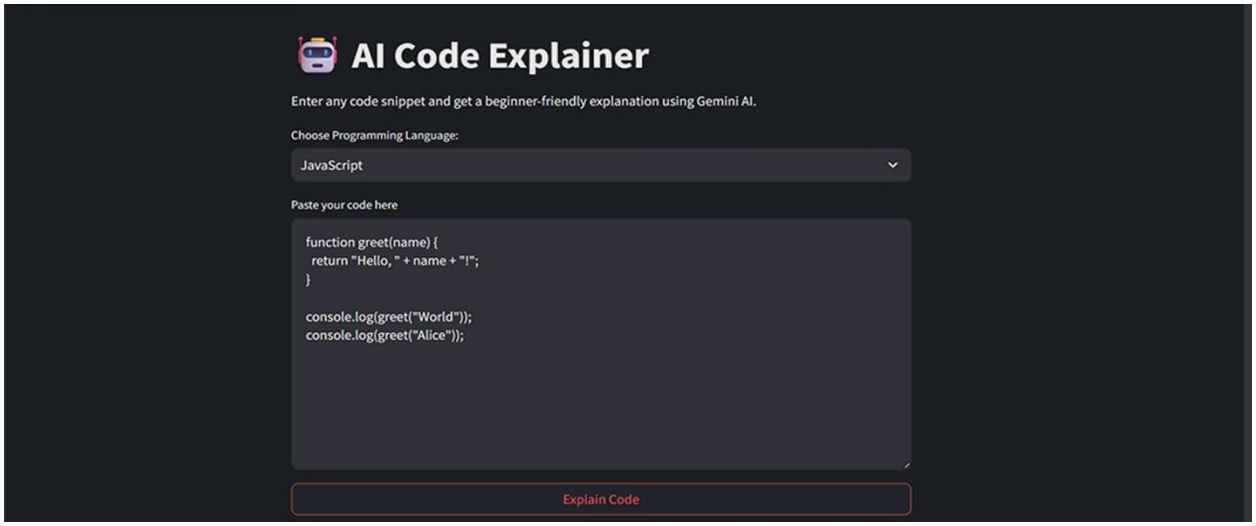
```python
            }

        try:
            response = requests.post(GEMINI_API_URL, headers=headers, json=payload, timeout=60) # Increased timeout
            response.raise_for_status() # This will raise an HTTPError for bad responses (4xx or 5xx)

            response_data = response.json()
            explanation = ""
            if "candidates" in response_data and response_data["candidates"]:
                for candidate in response_data["candidates"]:
                    if "content" in candidate and "parts" in candidate["content"]:
                        for part in candidate["content"]["parts"]:
                            if "text" in part:
                                explanation += part["text"] + "\n"

            if explanation.strip():
                st.success("✅ Explanation:")
                st.markdown(explanation.strip())
            else:
                st.error("❌ Gemini returned an empty or unparseable explanation. Please try a different code snippet or prompt.")
                st.json(response_data) # Show raw response for debugging

        except requests.exceptions.RequestException as e:
            # Catch any request-related errors (ConnectionError, HTTPError, Timeout)
            st.error(f"❌ An error occurred during API request: {e}")
            st.info("Please check your internet connection, API key, and the Gemini API status.")
            if hasattr(e, 'response') and e.response is not None:
                st.code(e.response.text, language="json") # Show API error response if available
        except json.JSONDecodeError as e:
            st.error(f"❌ Error decoding JSON response from API: {e}")
            st.info("The API might have returned malformed data.")
        except Exception as e:
            st.error(f"❌ An unexpected error occurred: {str(e)}")

# Footer or additional info (optional)
st.markdown("---")
st.markdown("Developed with Streamlit and Google Gemini AI (using requests).")
```

Output:-


Deploy
AI Code Explainer
Enter any code snippet and get a beginner-friendly explanation using Gemini AI.
Choose Programming Language:
Python
Paste your code here
Example for Python:
for i in range(5): print(i)
Explain Code


Deploy
AI Code Explainer
Enter any code snippet and get a beginner-friendly explanation using Gemini AI.
Choose Programming Language:
Python
Paste your code here
n = int(input("Enter an integer: "))
if n % 2 == 0:
print("Even")
else:
print("Odd")
Explain Code


Explain Code
Explanation:
This program checks if a number is even or odd.
First, it asks the user to enter a whole number (an integer). The program then stores that number in a variable called 'n'.
Next, it uses the modulo operator ( % ). This operator gives you the remainder after a division. For example, 10 % 3 is 1 (because 10 divided by 3 is 3 with a remainder of 1).
The program checks if the remainder of 'n' divided by 2 is equal to 0. If it is, that means the number is perfectly divisible by 2, and therefore even. In this case, the program prints "Even".
If the remainder is not 0 (meaning it's 1), then the number is odd, and the program prints "Odd". It's a simple way to determine even or odd numbers using a fundamental math concept.
Developed with Streamlit and Google Gemini AI (using requests).


AI Code Explainer
Enter any code snippet and get a beginner-friendly explanation using Gemini AI.
Choose Programming Language:
JavaScript
Paste your code here
function greet(name) {
return "Hello, " + name + "!";
}
console.log(greet("World"));
console.log(greet("Alice"));
Explain Code

✅ Explanation:

Imagine you have a little machine that takes a name as input and says hello to that name. That's what the code does!

It defines a "recipe" (called a function) named `greet`. This recipe has one step: it takes whatever name you give it and creates a greeting like "Hello, [name]!". For example, if you give it "World", it makes "Hello, World!".

Then, the code uses this recipe twice. First, it feeds the name "World" to the `greet` recipe, and the resulting "Hello, World!" is shown on the screen. Second, it uses the name "Alice", resulting in "Hello, Alice!" being displayed on the screen. It's like using a stamp to make two different greeting cards.

---

Developed with Streamlit and Google Gemini AI (using requests).

✅ Explanation:

Imagine you have a little machine that takes a name as input and says hello to that name. That's what the code does!

It defines a "recipe" (called a function) named `greet`. This recipe has one step: it takes whatever name you give it and creates a greeting like "Hello, [name]!". For example, if you give it "World", it makes "Hello, World!".

Then, the code uses this recipe twice. First, it feeds the name "World" to the `greet` recipe, and the resulting "Hello, World!" is shown on the screen. Second, it uses the name "Alice", resulting in "Hello, Alice!" being displayed on the screen. It's like using a stamp to make two different greeting cards.

---

Developed with Streamlit and Google Gemini AI (using requests).