Civil Engineering Insight Studio

Google Cloud Generative AI

MENTOR NAME : **SRI LANKA LAKSHMI NARAYANA**

FACULTY MENTOR NAME : **K.RATNA KUMARI**

TEAM ID : **LTVIP2026TMIDS38656**

MAIL : madhaveevenu226@gmail.com

ID NO : **SBAP0032744**

Project Submitted By

"PATNALA VENU MADHAVI"

Inroduction

Problem Statement: Civil engineers often face the challenge of manually describing structures based on images, which can be time-consuming and subjective. Without automated tools, generating detailed descriptions of civil engineering structures, including types, materials, dimensions, construction methods, and notable features, requires significant human effort and expertise. To address this challenge, there is a need for an efficient and reliable tool that can automatically analyze images of civil engineering structures and generate insightful descriptions, enabling engineers to make informed decisions and communicate effectively about their projects.

Scenario 1: Material Identification in Building Construction

A construction supervisor overseeing a building project utilizes the Civil Engineering Insight Studio to identify materials used in the building's construction. The supervisor uploads an image of the construction site and prompts the tool to identify construction materials such as concrete, steel, and bricks. The tool analyzes the image, identifies various materials, and provides a comprehensive list with descriptions of each material's type, quantity, and location within the structure.

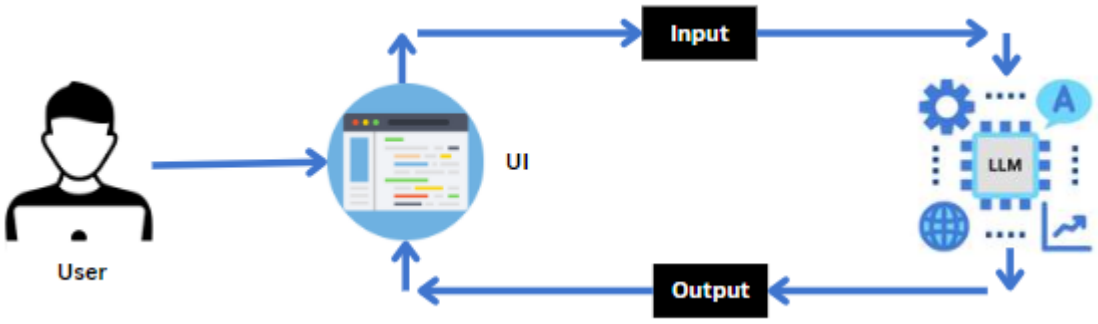
Scenario 2: Project Progress Documentation for Construction Firm

A civil engineering firm engaged in documenting ongoing construction projects utilizes the tool to streamline project documentation. Project managers upload images of construction sites and prompt the tool to document project progress, including descriptions of completed and planned structural elements. The tool analyzes the images, generates detailed documentation of completed structural elements, materials used, dimensions, construction methods, and planned phases of work, facilitating effective project management and communication within the firm.

Scenario 3: Structural Analysis of Bridge

In this scenario, a civil engineer tasked with assessing the structural integrity of a bridge uploads an image of the bridge to the Civil Engineering Insight Studio. The engineer requests an analysis of the bridge's structural components, including beams, columns, and trusses. The tool processes the image, identifies key structural elements, and generates detailed descriptions of each component's material, dimensions, and construction methods. Additionally, the tool highlights any notable features or engineering challenges observed in the structure.

Architecture



This diagram illustrates a high-level **Generative AI Workflow**, specifically focusing on how a user interacts with a Large Language Model (LLM) through a digital interface.

Here is a breakdown of the process:

The Interaction Cycle

The architecture follows a continuous loop, starting and ending with the user.

- **User:** The person initiating the request. This represents the starting point of the human-computer interaction.
- **UI (User Interface):** This is the bridge between the human and the machine (e.g., a chatbot window, a mobile app, or a web dashboard). It captures the user's intent and displays the final result.

Summary of the Loop

1. **Request:** User types into the **UI**.
2. **Processing:** The request travels as **Input** to the **LLM**.
3. **Generation:** The **LLM** computes the response.
4. **Delivery:** The generated **Output** travels back to the **UI** for the user to read.

Project Flow

- User Input: Users provide descriptions and upload images of civil engineering structures.
- Backend Processing: Input data is analyzed by specialized algorithms.
- Insight Generation: Detailed descriptions and related information are autonomously generated.
- Frontend Display: Generated insights and information are presented on the user interface.

To accomplish this, we have to complete all the activities listed below,

- **Requirements Specification**
  - Create a requirements.txt file to list the required libraries.
  - Install the required libraries
- **Initialization of Google API Key**
  - Generate Google API Key
  - Initialize Google API Key
- **Interfacing with Pre-trained Model**
  - Load the Gemini Pro pre-trained model
  - Implement a function to get gemini response
  - Implement a function to read the Image and set the image format for Gemini Pro model Input
  - Write a prompt for gemini model
- **Model Deployment**
  - Integrate with Web Framework
  - Process User Input and Generate Landmark Description
  - Run The Web Application

Prior Knowledge

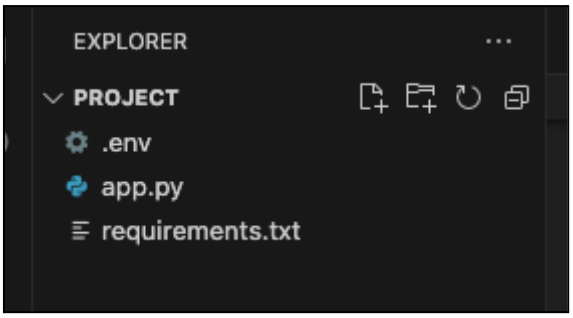
You must have the prior knowledge of the following topics to complete this project.

- Generative AI Concepts
- NLP: https://www.tutorialspoint.com/natural_language_processing/index.htm
- Generative AI: https://en.wikipedia.org/wiki/Generative_artificial_intelligence
- About Gemini: https://deepmind.google/technologies/gemini/#introduction
- Gemini API: https://ai.google.dev/gemini-api/docs/get-started/python

- Gemini Demo: https://colab.research.google.com/github/google/generative-ai-docs/blob/main/site/en/gemini-api/docs/get-started/python.ipynb
  Streamlit: https://www.geeksforgeeks.org/a-beginners-guide-to-streamlit/

Project Structure

Create the Project folder which contains application file as shown below

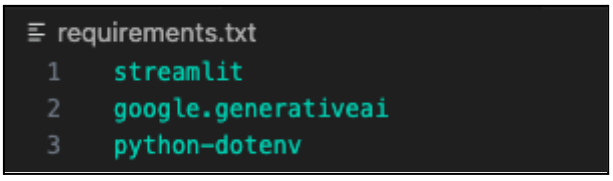


? .env contains the api key for Google Ai studio
? App.py contains the code for running the model using streamlit
? Requirements.txt has all the requirements (packages and libraries needed to install
before running the project)

Milestone 1: Requirements Specification

Specifying the required libraries in the requirements.txt file ensures seamless setup and reproducibility of the project environment, making it easier for others to replicate the development environment.

Activity 1: Create a requirements.txt file to list the required libraries.



- **Streamlit:**A Python library used for creating interactive web applications with simple Python scripts.

- **google.generativeai:**A module providing access to Google's Generative AI models for generating text, images, and other content.
- **python dotenv**:A Python library used for managing environment variables stored in a `.env` file, typically used for configuration settings.

Activity 2: Install the required libraries.

```
(base) Gururajs-MacBook-Air:Project gururajguggal$ pip install -r requirements.txt
```

- Open the terminal.
- Run the command: pip install r requirements.txt
- This command installs all the libraries listed in the requirements.txt file

Milestone 2: Initialization of Google API Key

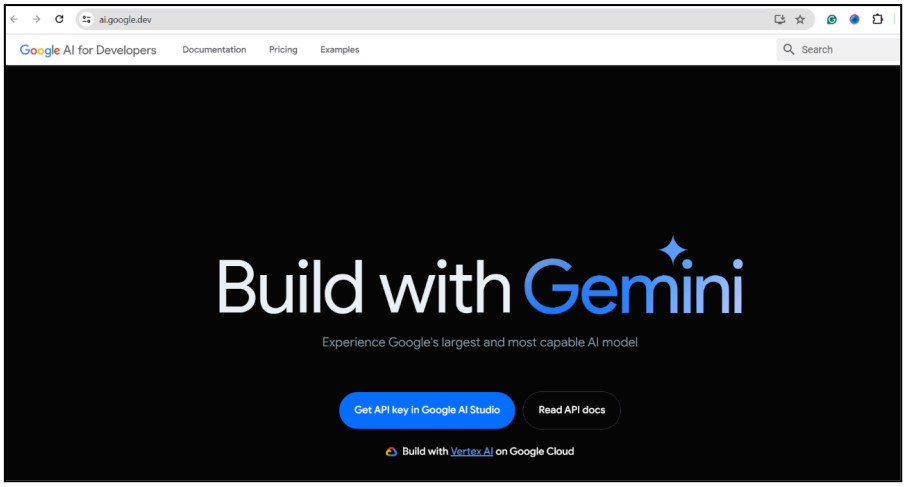
The Google API key is a secure access token provided by Google, enabling developers to authenticate and interact with various Google APIs. It acts as a form of identification, allowing users to access specific Google services and resources. This key plays a crucial role in authorizing and securing API requests, ensuring that only authorized users can access and utilize Google's services.

Activity 1: Generate Google API Key

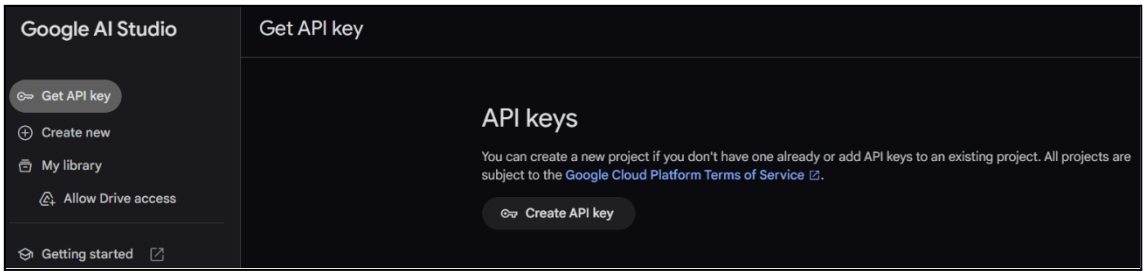
Click the provided link to access the following webpage.

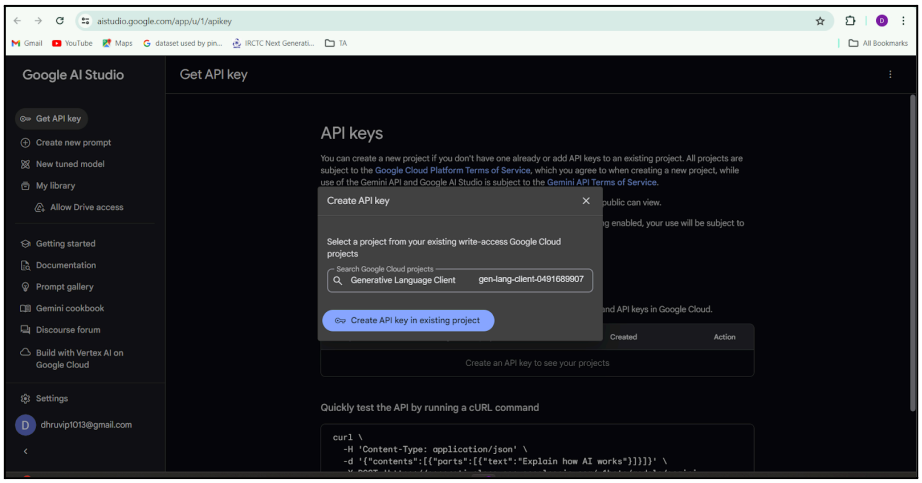
Link: https://ai.google.dev/gemini-api/docs/api-key

After signing in to your account, navigate to the 'Get an API Key' option. Clicking on this option will redirect you to another webpage as shown below.



Next, click on 'Create API Key' and choose the generative language client as the project. Then, select 'Create API key in existing project'.





Copy the newly generated API key as it is required for loading the Gemini Pro pre-trained model.

Activity 2: Initialize Google API Key

```
GOOGLE_API_KEY = "<Enter the copied Google API Key>"
```

- Create a .env file and define a variable named GOOGLE_API_KEY.
- Assign the copied Google API key to this variable.
- Paste the API key obtained from the previous steps here.

Milestone 3: Interfacing with Pre-trained Model

To interface with the pre-trained model, we'll start by creating an app.py file, which will contain both the model and Streamlit UI code.

Activity 1: Load the Gemini Pro API

```python
from dotenv import load_dotenv
import streamlit as st
import os
import google.generativeai as genai
from PIL import Image

# Load environment variables
load_dotenv()

# Configure Google Generative AI with API key
genai.configure(api_key=os.getenv("GOOGLE_API_KEY"))

```

- `dotenv`: Loads environment variables from a `.env` file into the environment.
- `streamlit`: Creates interactive web applications using Python scripts.
- `os`: Provides a way to interact with the operating system, including accessing environment variables and file paths.
- `google.generative ai`: Grants access to Google's Generative AI models for content generation.
- `PIL (Python Imaging Library)`: Allows opening, manipulating, and saving various image file formats.

```python
from dotenv import load_dotenv
import streamlit as st
import os
import google.generativeai as genai
from PIL import Image

# Load environment variables
load_dotenv()

# Configure Google Generative AI with API key
genai.configure(api_key=os.getenv("GOOGLE_API_KEY"))

```

- Load Environment Variables: Utilizes dotenv to load environment variables, ensuring sensitive data like API keys are securely managed.
- Configure Google Generative AI: Uses the configured API key from the environment variables to set up and authenticate with Google's Generative AI service, enabling content generation functionalities.

Activity 2: Implement a function to get gemini response

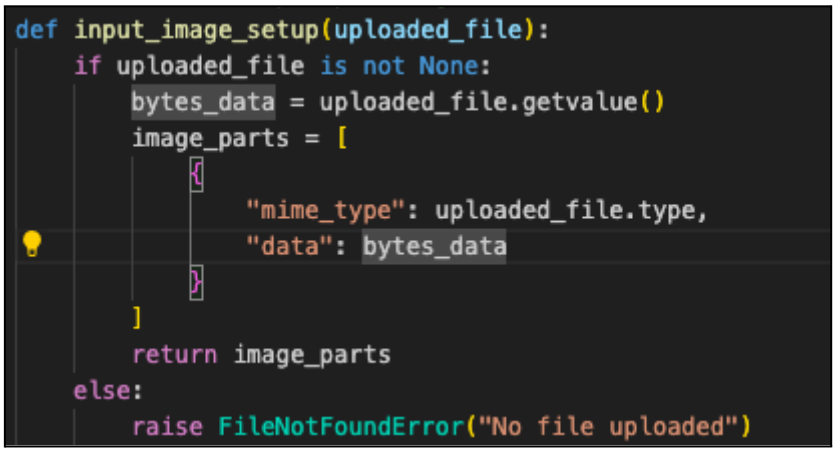
```python
# Function to get Gemini response
def get_gemini_response(input_text, image, prompt):
    model = genai.GenerativeModel('gemini-pro-vision')
    response = model.generate_content([input_text, image[0], prompt])
    return response.text
```

- Function Definition: The function `get_gemini_response` is defined to accept three parameters: `input_text`, `image`, and `prompt`.
- Model Initialization: Inside the function, a new instance of the Gemini AI model is created using `genai.GenerativeModel('gemini-pro-vision')`.
- Generate Content Call: The function then calls `model.generate_content` to generate content.
- Input Parameters: The `generate_content` method is given a list containing `input_text`, the first item from the `image` list, and `prompt`.
- Content Generation: The AI model processes these inputs to generate a detailed description of the civil engineering structure in the image.
- Extracting Text: The response from the AI model includes various data; the function specifically extracts the text part of this response.
- Return Statement: The extracted text is returned by the function.

- Purpose: This function facilitates the interaction with the Gemini AI model to generate detailed descriptions of civil engineering structures based on input text, an image, and a prompt, aiding civil engineers in analyzing and documenting structural details.

Activity 3: Implement a function to read the Image and set the image format for Gemini Pro model Input

Activity 3: Implement a function to read the Image and set the image format for Gemini Pro model Input



```
def input_image_setup(uploaded_file):
    if uploaded_file is not None:
        bytes_data = uploaded_file.getvalue()
        image_parts = [
            {
                "mime_type": uploaded_file.type,
                "data": bytes_data
            }
        ]
        return image_parts
    else:
        raise FileNotFoundError("No file uploaded")
```

- Function Definition: The function `input_image_setup` is defined to take a single parameter, `uploaded_file`.
- Check for File: The function first checks if `uploaded_file` is not `None`, ensuring that a file has been uploaded.
- Retrieve File Data: If a file is uploaded, it retrieves the file's binary data using `uploaded_file.getvalue()`.
- Prepare Image Parts: It then creates a list called `image_parts` containing a dictionary.
- Dictionary Content: The dictionary includes the MIME type of the uploaded file (`uploaded_file.type`) and the binary data (`bytes_data`).
- Return Image Parts: The function returns the `image_parts` list, which is formatted for further processing.

- Handle No File: If no file is uploaded, the function raises a `FileNotFoundError` with the message "No file uploaded".

Activity 4: Write a prompt for gemini model

```
input_prompt = """
You are a civil engineer. Please describe the structure in the image and provide details such as its type,

1. Type of structure - Description
2. Materials used - Description
```

The variable input_prompt is a single-line string designed as a prompt for a captioning AI model. It instructs the model to analyze an image .Additionally, the model is to provide a detailed breakdown of each item with its respective count

Milestone 4: Model Deployment

This milestone sets up a web application for the Civil Engineering Structure Description App. It allows users to input a text prompt and upload an image of a civil engineering structure. Upon uploading an image, the app displays the image and provides a button for users to submit their input. The app is designed to describe civil engineering structures based on user input, using the Google Generative AI model. It provides detailed descriptions of the structure, including type, materials used, dimensions, construction methods, and notable features or engineering challenges

Activity 1:Integrate with Web Framework

```
# Initialize Streamlit app
st.set_page_config(page_title="Civil Engineering Insight Studio", page_icon="🏗️")
st.header("🏗️ Civil Engineering Insight Studio")
input_text = st.text_input("📝 Input Prompt: ", key="input")
uploaded_file = st.file_uploader("🖼️ Choose an image...", type=["jpg", "jpeg", "png"])

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

submit = st.button("🚀 Describe Structure")

input_prompt = """
You are a civil engineer. Please describe the structure in the image and provide details such as its type, materials used,

1. Type of structure - Description
```

- Initialize Streamlit App with configured page title and icon.
- Header Definition: The code starts by setting the header of the Streamlit application to "Civil Engineering Insight Studio" using st.header(). This sets the title for the web app.
- Text Input Field: The st.text_input() function creates an input field where users can type a text prompt. The prompt is labeled " Input Prompt: " and is assigned the key "input" for easy reference.
- File Uploader: The st.file_uploader() function creates an interface for users to upload an image file. The accepted file types are specified as "jpg", "jpeg", and "png". The label for this uploader is "Choose an image...".
- Image Display: If a user uploads a file, the code reads the image using Image.open(uploaded_file). The uploaded image is then displayed in the web app with the caption "Uploaded Image." and is set to use the column width for display using st.image().
- Submit Button: The st.button() function creates a button labeled " Describe Structure". When this button is clicked, the code proceeds to the next steps.
- Input Prompt: The input_prompt variable contains a detailed prompt for the AI model. It instructs the model to describe the structure in the image and provide various details such as type, materials used, dimensions, construction methods, and notable features or engineering challenges.
- Button Click Handling: When the "Describe Structure" button is clicked, the code checks if an image file has been uploaded. If so, it processes the image and uses it along with the input text and prompt to generate a detailed description of the civil engineering structure.

Activity 2: Process User Input and Generate Landmark Description

This activity involves processing user input and generating a description of the landmark based on the provided information.

```python
# If submit button is clicked
if submit:
    try:
        image_data = input_image_setup(uploaded_file)
        response = get_gemini_response(input_text, image_data, input_prompt)
        st.subheader("📋 Description of the Civil Engineering Structure:")
        st.markdown(f'<div class="st-ba">{response}</div>', unsafe_allow_html=True)
    except Exception as e:
        st.error(f"⚠ Error: {str(e)}")
```

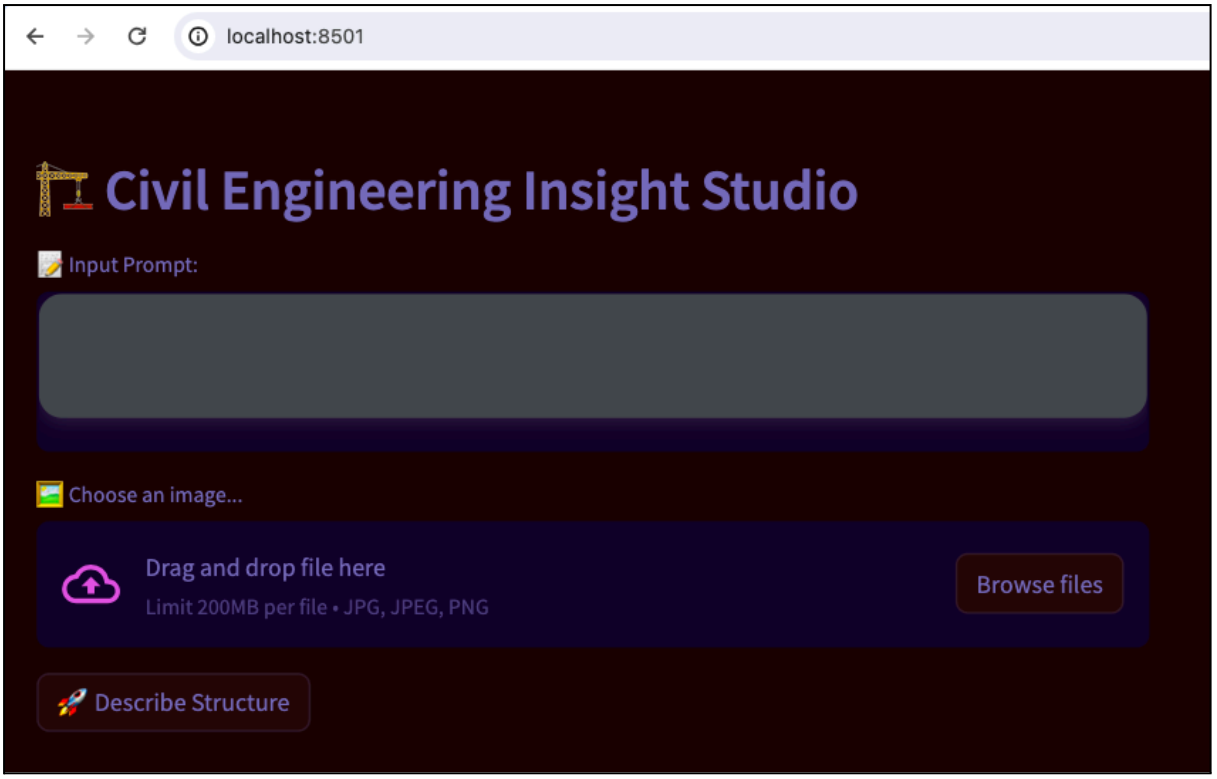
- This code is part of a web application designed for civil engineering projects.
- Users can input text and upload an image of a structure.
- The app processes this data to create a description of the structure.
- The description is then displayed on the webpage.
- If any errors occur during this process, a warning message is shown.
- Overall, the code facilitates easy understanding and analysis of civil engineering structures through user input and automated description generation.

Activity 3:Run the Web Application

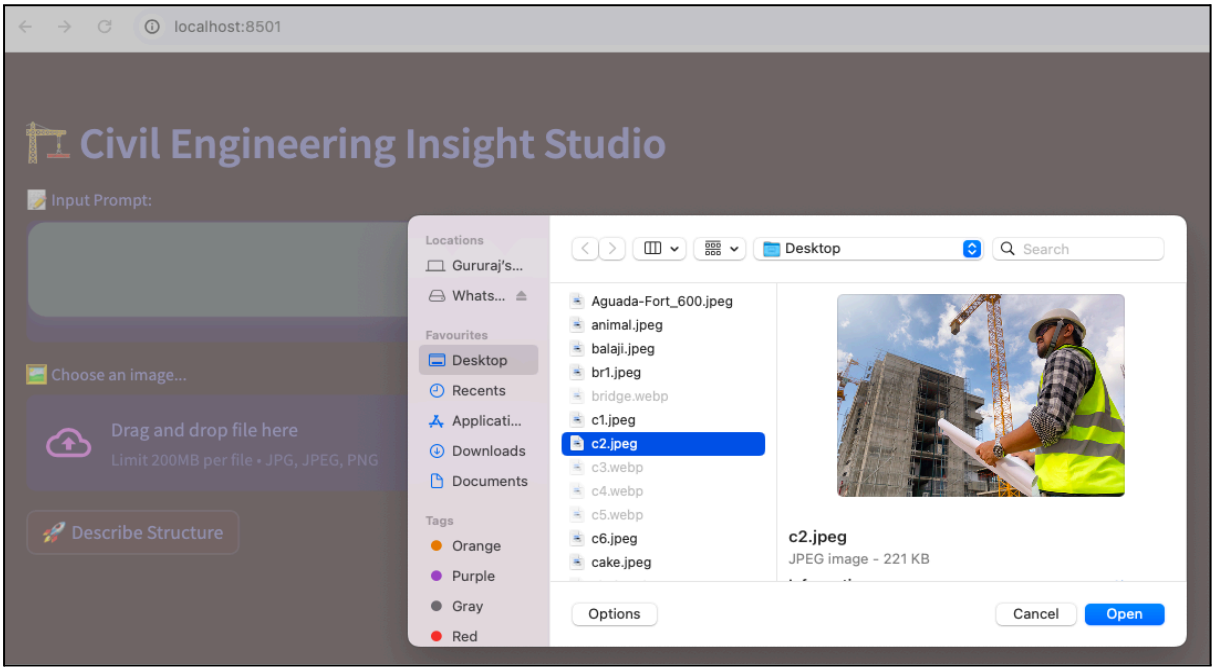
- Open the anaconda prompt from the start menu
- Navigate to the folder where your Python script is.
- Now type “streamlit run app.py” command
- Navigate to the localhost where you can view your web page

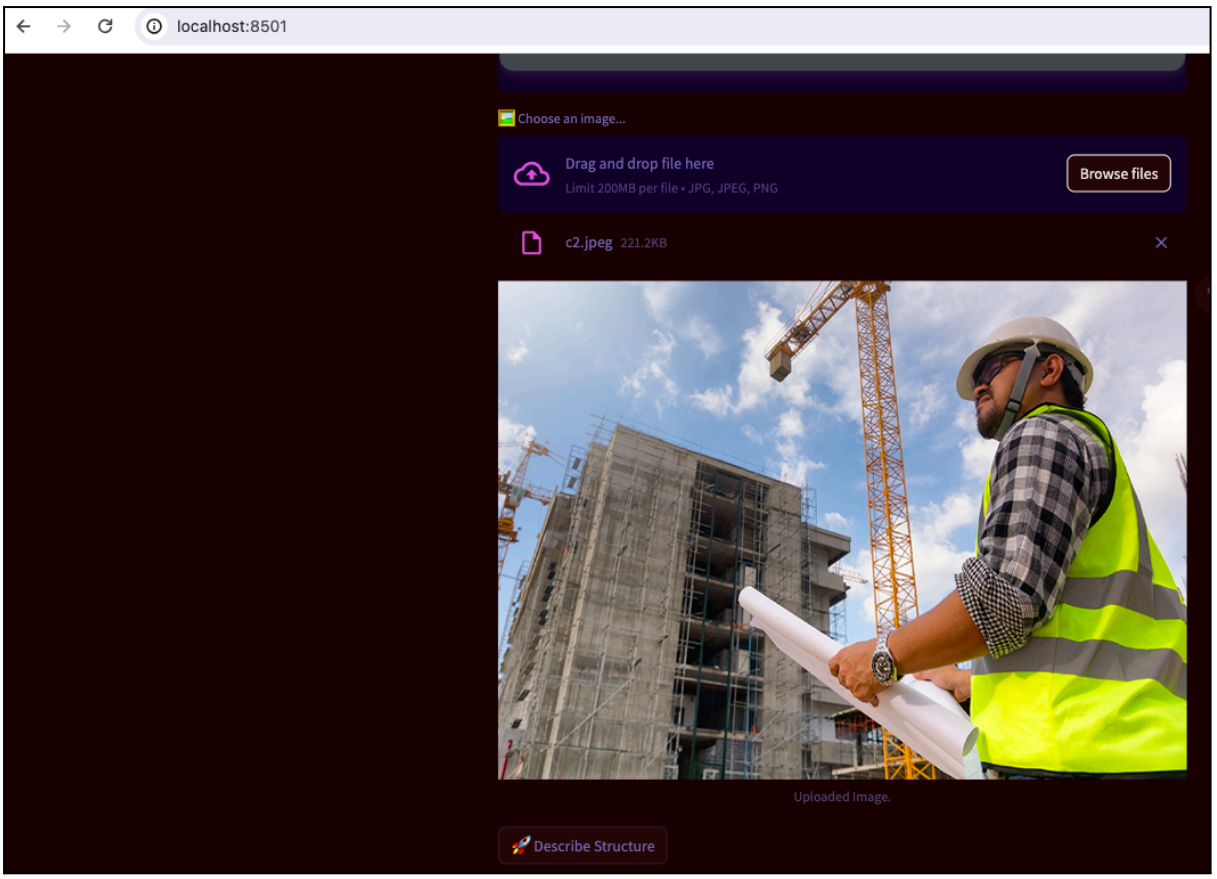
```
(base) Gururajs-MacBook-Air:Project gururajguggal$ streamlit run app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.29.143:8501

  For better performance, install the Watchdog module:
```

Now, the application will open in the web browser.

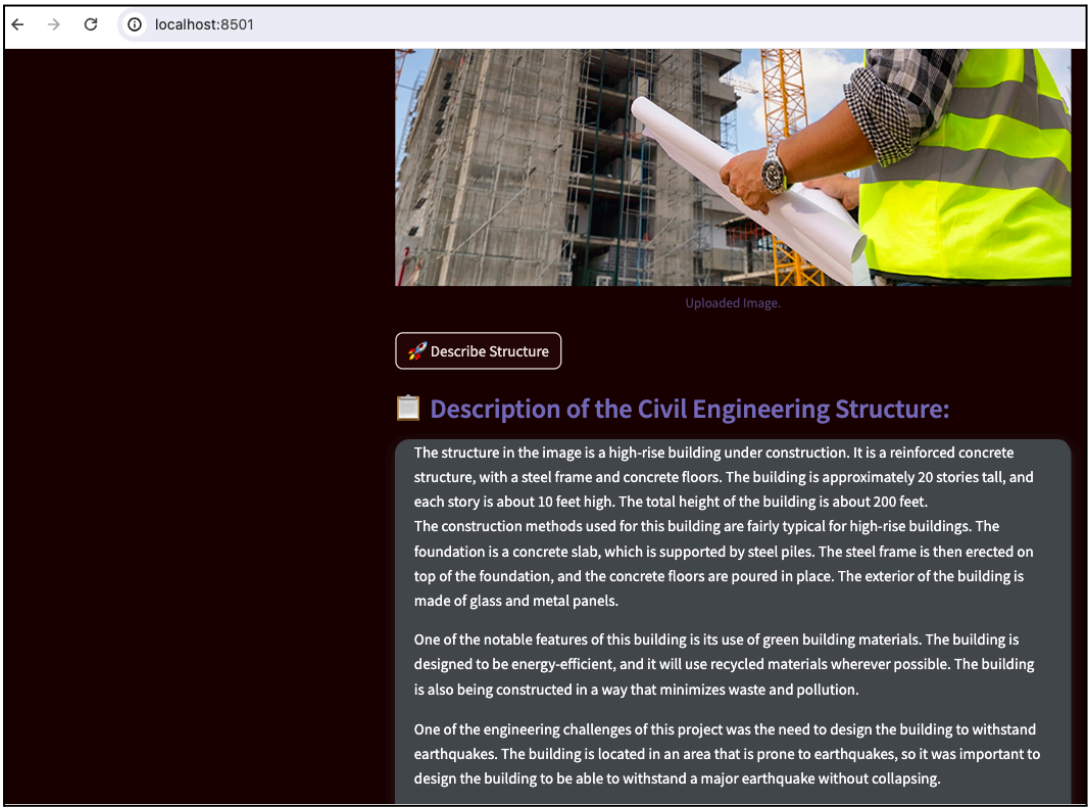

localhost:8501
Civil Engineering Insight Studio
Input Prompt:
Choose an image...
Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG
Browse files
Describe Structure

Input1:

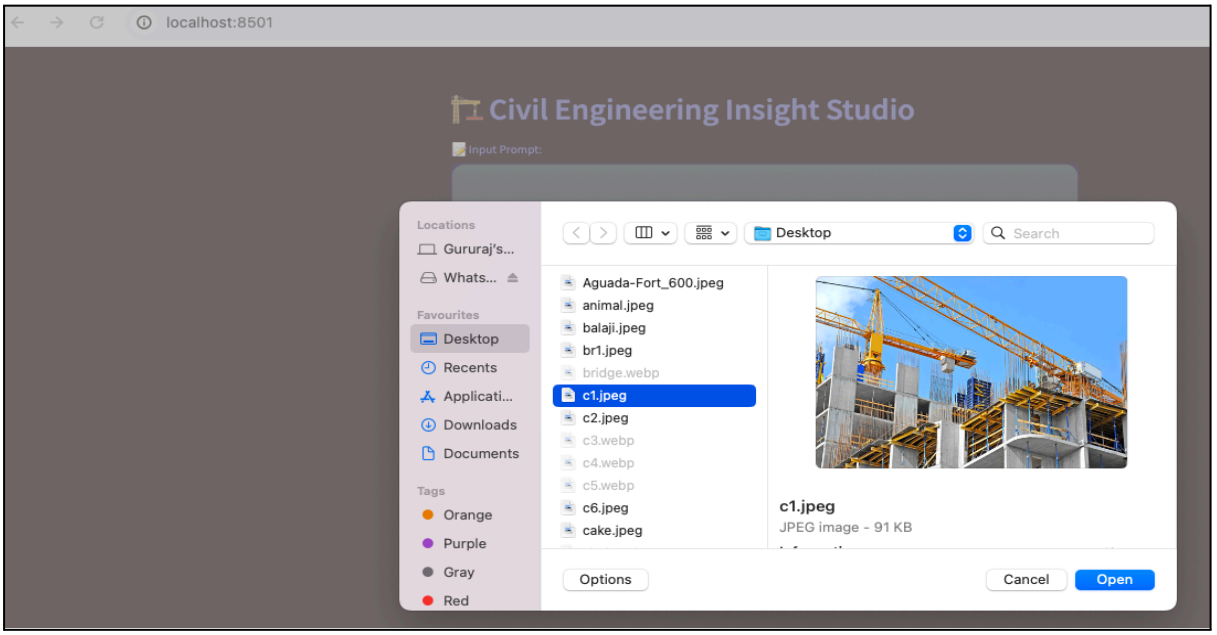

localhost:8501
Civil Engineering Insight Studio
Input Prompt:
Choose an image...
Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG
Describe Structure
Locations
Gururaj's...
Whats...
Favourites
Desktop
Recents
Applicati...
Downloads
Documents
Tags
Orange
Purple
Gray
Red
Desktop
Search
Aguada-Fort_600.jpeg
animal.jpeg
balaji.jpeg
br1.jpeg
bridge.webp
c1.jpeg
c2.jpeg
c3.webp
c4.webp
c5.webp
c6.jpeg
cake.jpeg
c2.jpeg
JPEG image - 221 KB
Options
Cancel
Open

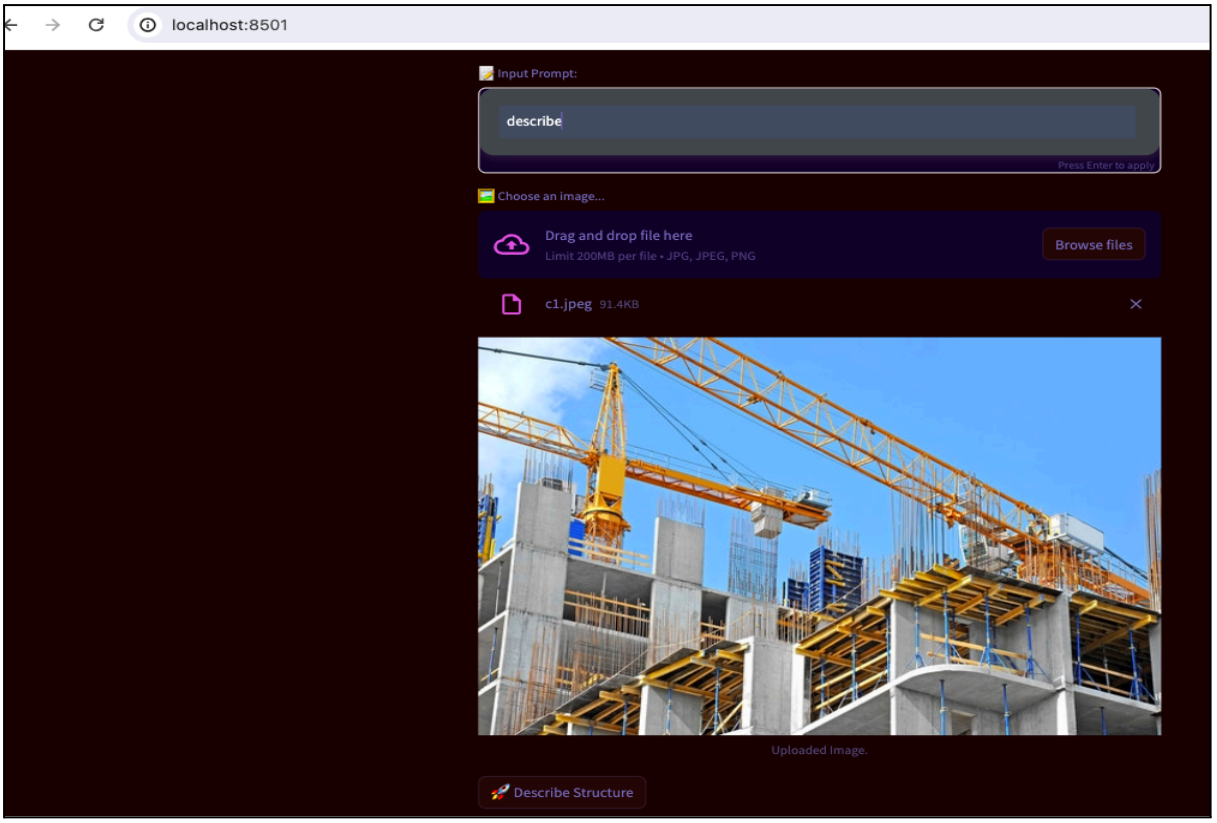

localhost:8501
Choose an image...
Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG
Browse files
c2.jpeg 221.2KB
Uploaded Image.
Describe Structure

Output:

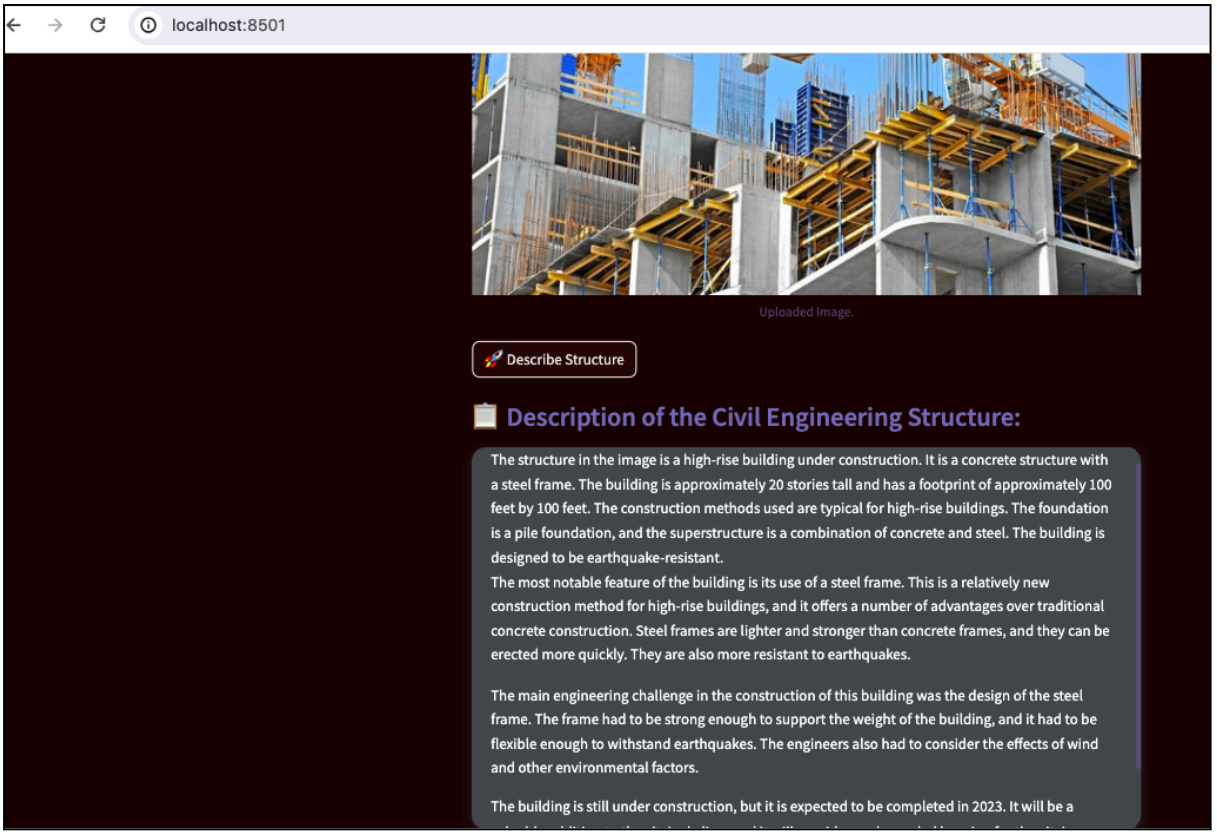

localhost:8501
Uploaded Image.
Describe Structure
Description of the Civil Engineering Structure:
The structure in the image is a high-rise building under construction. It is a reinforced concrete structure, with a steel frame and concrete floors. The building is approximately 20 stories tall, and each story is about 10 feet high. The total height of the building is about 200 feet.
The construction methods used for this building are fairly typical for high-rise buildings. The foundation is a concrete slab, which is supported by steel piles. The steel frame is then erected on top of the foundation, and the concrete floors are poured in place. The exterior of the building is made of glass and metal panels.
One of the notable features of this building is its use of green building materials. The building is designed to be energy-efficient, and it will use recycled materials wherever possible. The building is also being constructed in a way that minimizes waste and pollution.
One of the engineering challenges of this project was the need to design the building to withstand earthquakes. The building is located in an area that is prone to earthquakes, so it was important to design the building to be able to withstand a major earthquake without collapsing.

Input 2:

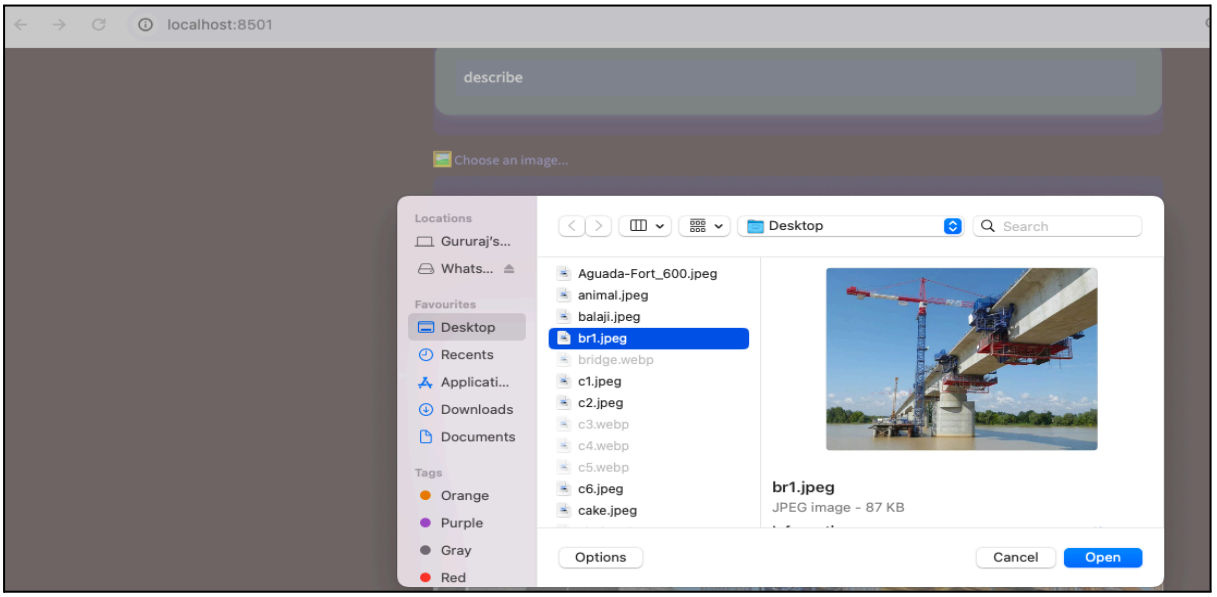


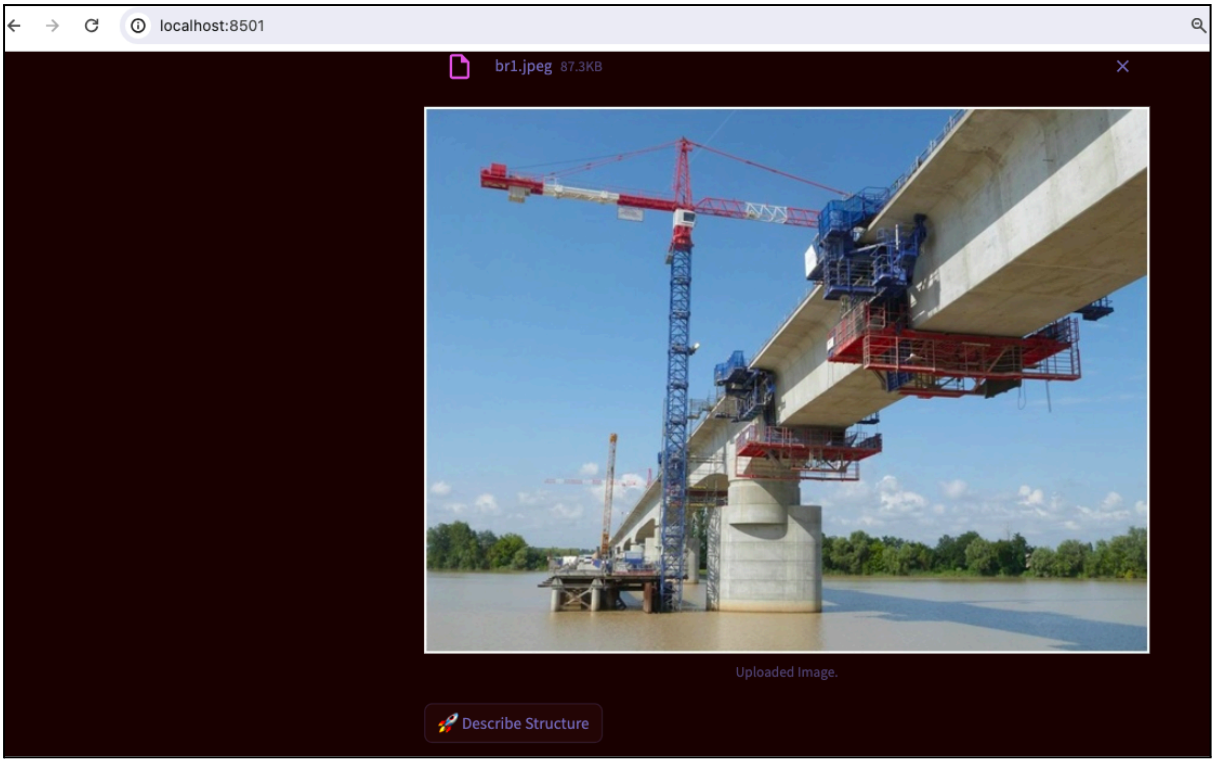


Output:

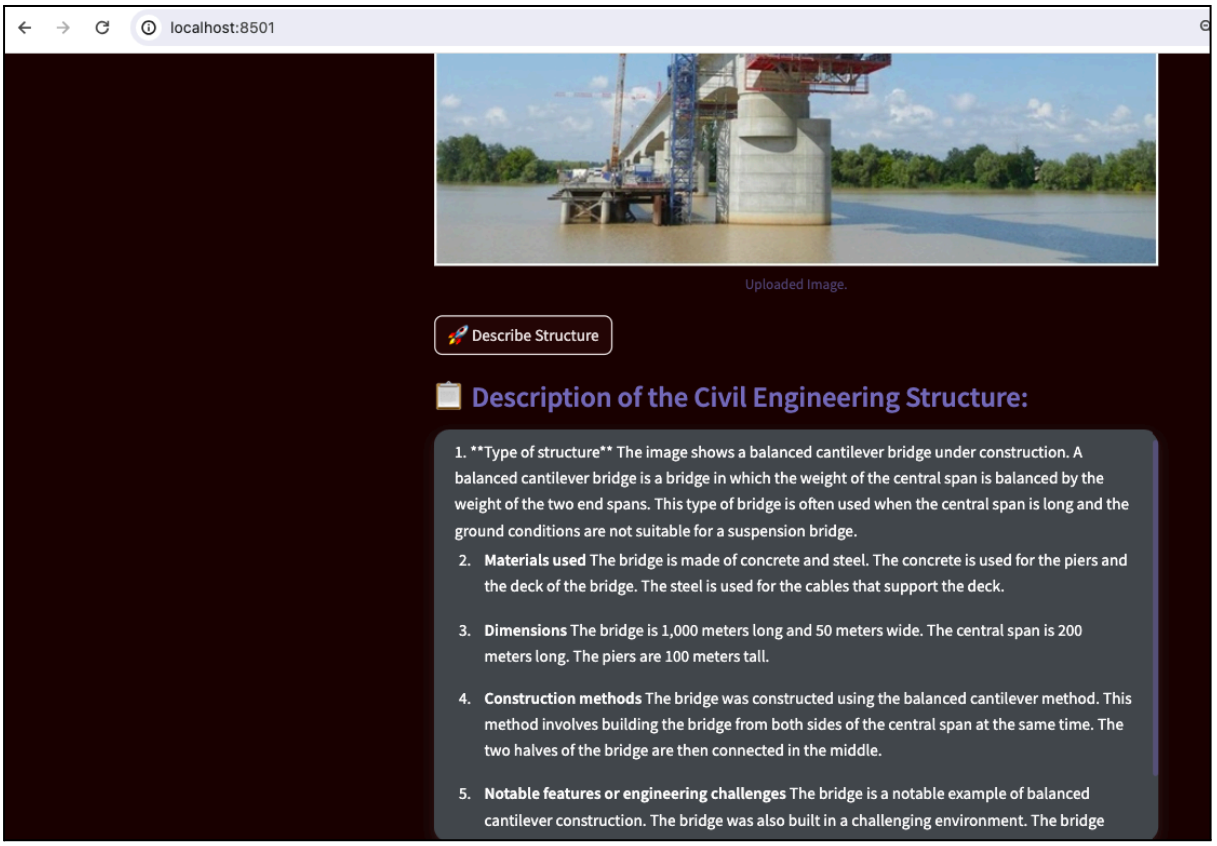

localhost:8501
Uploaded Image.
Describe Structure
Description of the Civil Engineering Structure:
The structure in the image is a high-rise building under construction. It is a concrete structure with a steel frame. The building is approximately 20 stories tall and has a footprint of approximately 100 feet by 100 feet. The construction methods used are typical for high-rise buildings. The foundation is a pile foundation, and the superstructure is a combination of concrete and steel. The building is designed to be earthquake-resistant.
The most notable feature of the building is its use of a steel frame. This is a relatively new construction method for high-rise buildings, and it offers a number of advantages over traditional concrete construction. Steel frames are lighter and stronger than concrete frames, and they can be erected more quickly. They are also more resistant to earthquakes.
The main engineering challenge in the construction of this building was the design of the steel frame. The frame had to be strong enough to support the weight of the building, and it had to be flexible enough to withstand earthquakes. The engineers also had to consider the effects of wind and other environmental factors.
The building is still under construction, but it is expected to be completed in 2023. It will be a

Input 3:


localhost:8501
describe
Choose an image...
Locations
Gururaj's...
Whats...
Favourites
Desktop
Recents
Applicati...
Downloads
Documents
Tags
Orange
Purple
Gray
Red
Desktop
Search
Aguada-Fort_600.jpeg
animal.jpeg
balaji.jpeg
br1.jpeg
bridge.webp
c1.jpeg
c2.jpeg
c3.webp
c4.webp
c5.webp
c6.jpeg
cake.jpeg
br1.jpeg
JPEG image - 87 KB
Options
Cancel
Open


localhost:8501
br1.jpeg 87.3KB
Uploaded Image.
Describe Structure

Output:


localhost:8501
Uploaded Image.
Describe Structure
Description of the Civil Engineering Structure:
1. **Type of structure** The image shows a balanced cantilever bridge under construction. A balanced cantilever bridge is a bridge in which the weight of the central span is balanced by the weight of the two end spans. This type of bridge is often used when the central span is long and the ground conditions are not suitable for a suspension bridge.
2. Materials used The bridge is made of concrete and steel. The concrete is used for the piers and the deck of the bridge. The steel is used for the cables that support the deck.
3. Dimensions The bridge is 1,000 meters long and 50 meters wide. The central span is 200 meters long. The piers are 100 meters tall.
4. Construction methods The bridge was constructed using the balanced cantilever method. This method involves building the bridge from both sides of the central span at the same time. The two halves of the bridge are then connected in the middle.
5. Notable features or engineering challenges The bridge is a notable example of balanced cantilever construction. The bridge was also built in a challenging environment. The bridge

COMMENTS

FIRST AND FOREMOST, I SINCERELY GRATITUDE TO OUR ESTEEMED INSTITUTE SRI VASAVI DEGREE COLLEGE, FOR GIVING ME THIS OPPORTUNITY TO FULFILL OUR WARM DREAM TO BECOME A GRADUATE. OUR SINCERE GRADITUDE TO OUR LONG-TERM INTERNSHIP GUIDE **SRI L LAKSHMI NARAYANA**, LECTURER DEPARTMENT OF COMPUTER SCIENCE FOR TIMELY COOPERATION AND VALUABLE SUGGESTIONS WHILE CARRYING OUT THIS INTERNSHIP.

I EXPRESS MY SINCERE THANKS AND HEARTFUL GRATITUDE TO **SRI L LAKSHMI NARAYANA**, HOD IN COMPUTER SCIENCE FOR PERMITTING ME TO DO MY PROJECT INTERNSHIP. I EXPRESS MY SINCERE THANKS AND HEARTFUL GRATITUDE TO **SRI M RAMA KRISHNA**, PRINCIPAL FOR PROVIDING A FAVOURABLE ENVIRONMENT AND SUPPORTING ME DURING THE DEVELOPMENT OF THIS INTERNSHIP.

THANK YOU,SMART BRIDGE

----PATNALA VENU MADHAVI
TEAM LEADER

THE END

SIGNATURE OF THE HOD

SIGNATURE OF THE PRINCIPAL