

FINAL REPORT

Civil Engineering Insight Studio

1. INTRODUCTION

1.1 Project Overview

Civil Engineering Insight Studio is a web-based Generative AI application designed to automate structural analysis and documentation from construction site images. The system leverages Google Gemini Vision API to analyze images and generate structured engineering reports.

The application allows users to:

- Upload construction site images
- Enter structural analysis request
- Generate structured engineering report
- Identify materials and structural components
- View formatted AI-generated output

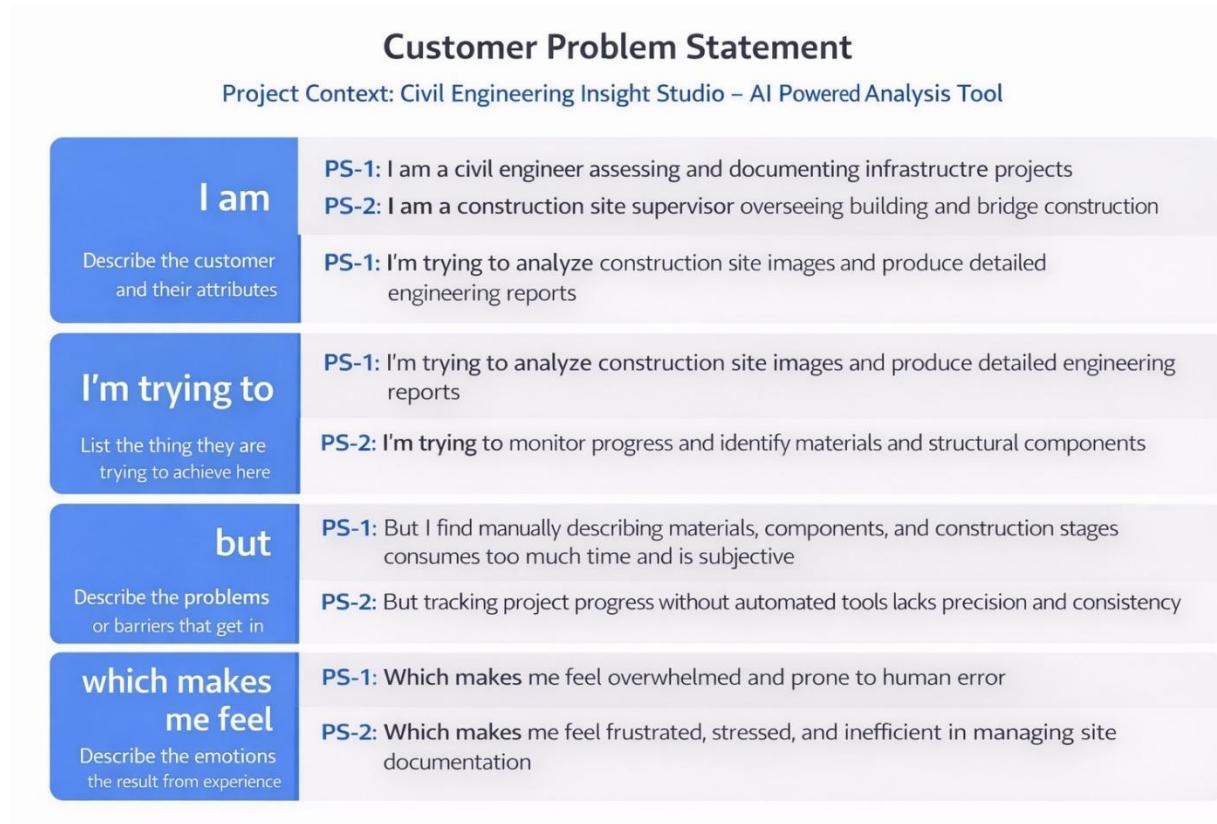
1.2 Purpose

- The purpose of this project is to reduce manual documentation effort in civil engineering projects, provide AI-assisted structural analysis, and demonstrate real-world integration of multimodal Generative AI within a deployable web application

2. IDEATION PHASE

2.1 Problem Statement

Civil engineers and construction supervisors spend significant time manually analyzing construction images and preparing structured reports. Traditional documentation methods are time-consuming and subjective.



2.2 Empathy Map Canvas

Target User: Civil Engineer / Site Supervisor

Pain Points:

- Time-consuming manual analysis
- Risk of missing structural details
- Inconsistent reporting format
- Heavy workload under tight deadlines

2.3 Brainstorming

Among multiple AI-based ideas, the AI-powered civil structure analysis system was selected due to its strong industry relevance, technical feasibility, and clear problem-solution alignment.

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

⌚ 10 minutes to prepare
⌚ 1 hour to collaborate
👤 2-8 people recommended

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
⌚ 10 minutes

1 Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
⌚ 5 minutes

A Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

C Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.

[Open article →](#)

PROBLEM
How might we [your problem statement]?

Key rules of brainstorming
To run a smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

3. REQUIREMENT ANALYSIS

3.1 Functional Requirements

- Accept construction image upload
- Accept structural analysis request
- Generate structured report via Gemini API
- Display formatted engineering output
- Handle API errors gracefully

3.2 Non-Functional Requirements

- Usability – Simple Streamlit interface
- Performance – Response within 3–10 seconds
- Reliability – No crash during API failure
- Security – API key stored securely using environment variables
- Scalability – Cloud deployable stateless architecture

3.3 Technology Stack

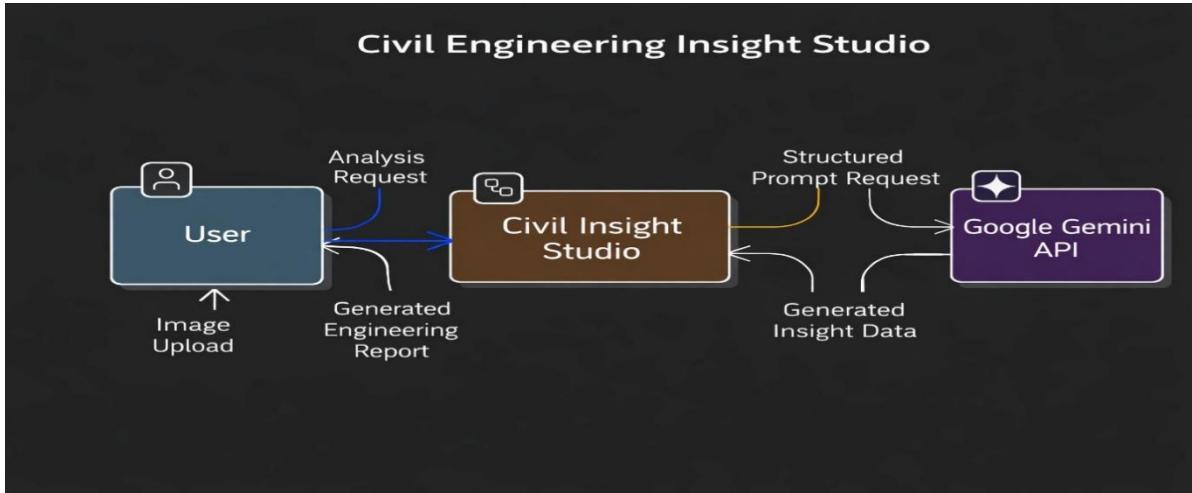
Frontend: Streamlit

Backend: Python

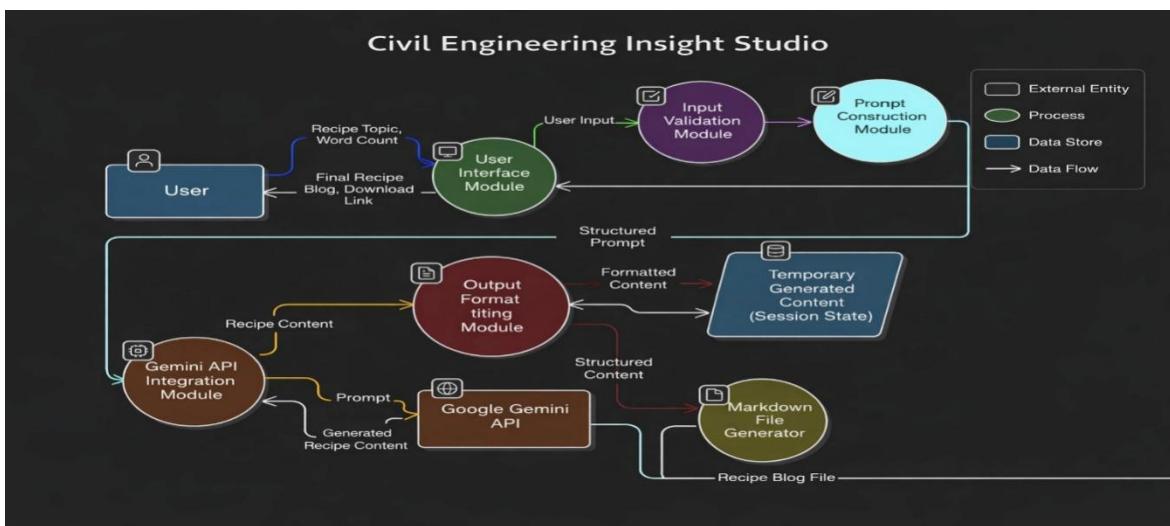
AI Model: Google Gemini Vision API

Deployment: Local / Streamlit Cloud

DFD Level 0 (Industry Standard)



DFD Level 1 (Industry Standard)



The system follows a stateless architecture with no persistent database.

4. PROJECT DESIGN

4.1 Problem-Solution Fit

Problem: Manual structural documentation is slow and inconsistent.

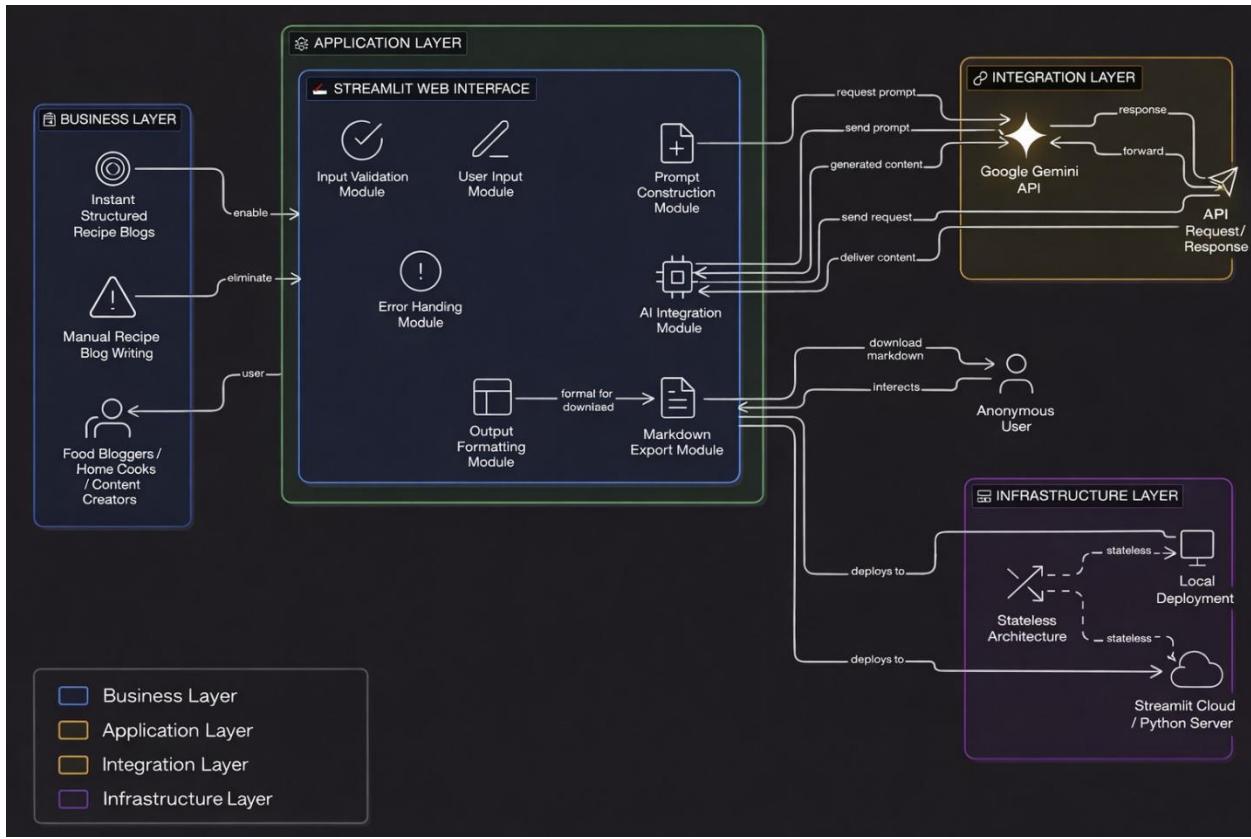
Solution: AI-powered image analysis generating structured engineering reports instantly.

<p>1. CUSTOMER SEGMENT(S) Who is your customer? I.e. working parents of 0-5 y.o. kids</p>	<p>6. CUSTOMER CONSTRAINTS What constraints prevent your customers from taking action or limit their choices of solutions? I.e. spending power, budget, no cash, network connection, available devices.</p>	<p>5. AVAILABLE SOLUTIONS Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? I.e. pen and paper is an alternative to digital notetaking</p>
<p>2. JOBS-TO-BE-DONE / PROBLEMS Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one, explore different sides.</p>	<p>9. PROBLEM ROOT CAUSE What is the real reason that this problem exists? What is the back story behind the need to do this job? I.e. customers have to do it because of the change in regulations.</p>	<p>7. BEHAVIOUR What does your customer do to address the problem and get the job done? I.e. directly related: find the right solar panel installer, calculate usage and benefits; Indirectly associated: customers spend free time on volunteering work (I.e. Greenpeace)</p>
<p>3. TRIGGERS What triggers customers to act? I.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.</p>	<p>10. YOUR SOLUTION If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.</p>	<p>8. CHANNELS OF BEHAVIOUR 8.1 ONLINE What kind of actions do customers take online? Extract online channels from #7</p> <p>8.2 OFFLINE What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.</p>

4.2 Solution Architecture

The system follows a layered architecture:

- Client Layer – Civil Engineer (Web User)
- Presentation Layer – Streamlit Interface
- Application Layer – Python Logic + Prompt Engineering
- Integration Layer – Google Gemini Vision API
- Infrastructure Layer – Local/Cloud Hosting



5. PROJECT PLANNING & SCHEDULING

Development was completed in two sprints covering UI development, Gemini API integration, structured report formatting, validation, and testing.

Average team velocity: 15–19 Story Points per sprint.

6. FUNCTIONAL AND PERFORMANCE TESTING

Functional tests validated image upload, API integration, and report generation.

Performance tests confirmed average response time within 3–10 seconds depending on API latency.

System remained stable during UAT.

7. RESULTS

The system successfully generated structured engineering reports including:

- Structure type identification
- Materials detection
- Structural components (beams, columns, slabs)
- Construction stage observations

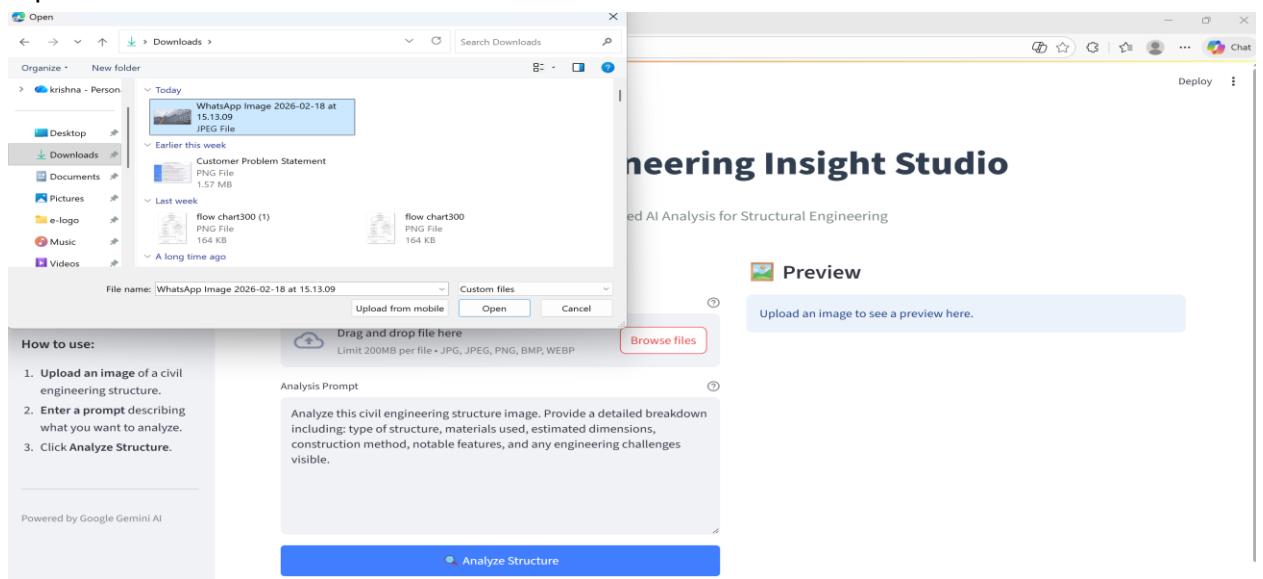
Outputs demonstrated consistent formatting and professional documentation quality.

7.1 Output Screenshots

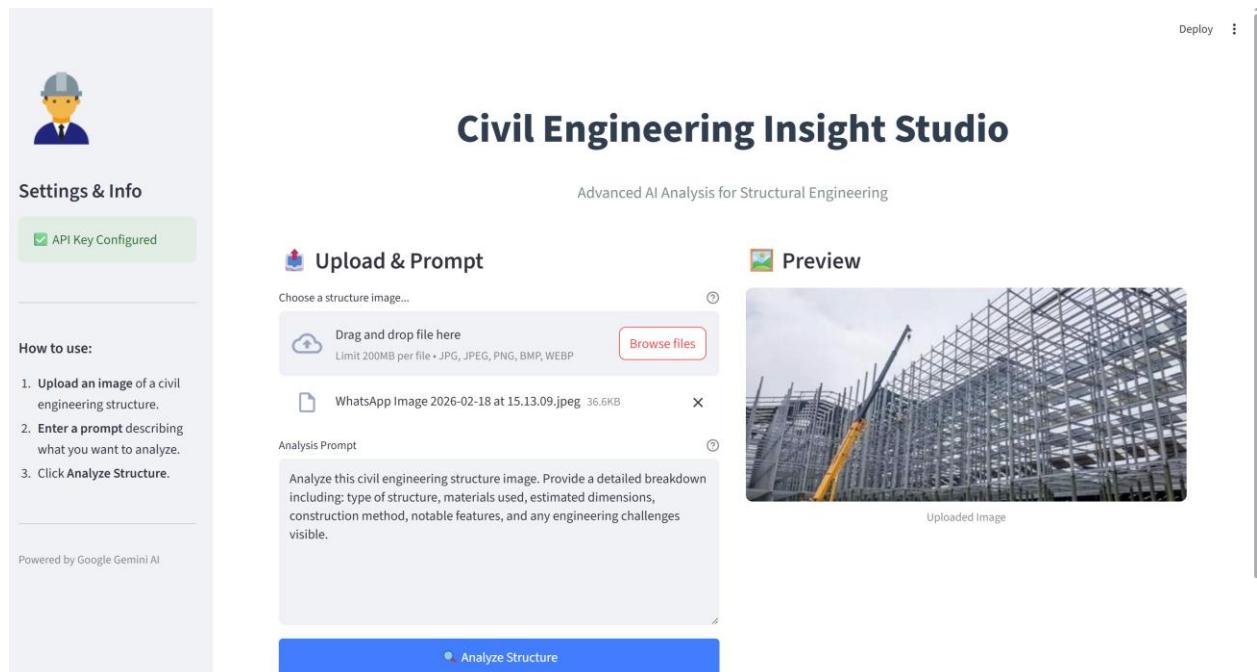
- Main interface screen

The screenshot displays the main interface of the Civil Engineering Insight Studio. On the left, there is a sidebar titled "Settings & Info" which includes a "API Key Configured" status indicator. Below this is a "How to use:" section with three numbered steps: 1. Upload an image of a civil engineering structure, 2. Enter a prompt describing what you want to analyze, and 3. Click Analyze Structure. At the bottom of the sidebar, it says "Powered by Google Gemini AI". The main content area is titled "Civil Engineering Insight Studio" and "Advanced AI Analysis for Structural Engineering". It features two main sections: "Upload & Prompt" and "Preview". The "Upload & Prompt" section contains a "Choose a structure image..." input field with a "Drag and drop file here" button and a "Browse files" button. Below this is an "Analysis Prompt" input field with placeholder text: "Analyze this civil engineering structure image. Provide a detailed breakdown including: type of structure, materials used, estimated dimensions, construction method, notable features, and any engineering challenges visible." A large blue "Analyze Structure" button is located at the bottom of this section. To the right, there is a "Preview" section with a placeholder text "Upload an image to see a preview here." and a small camera icon.

- **Input section**



- **Upload generation**



- Generated output

Upload & Prompt

Choose a structure image...

+ Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG, BMP, WEBP

WhatsApp Image 2026-02-18 at 15.13.09.jpeg 36.6KB ×

Analysis Prompt

Analyze this civil engineering structure image. Provide a detailed breakdown including: type of structure, materials used, estimated dimensions, construction method, notable features, and any engineering challenges visible.

● Analyze Structure

⌚ ⚙️ Analyzing structure structure... This may take a moment.

Preview

Uploaded Image

- Markdown download confirmation

CIVIL ENGINEERING AI

Advanced AI Analysis for Structural Engineering

Downloads

structural_analysis_report.md	Open file	⋮
See more		

Upload & Prompt

Choose a structure image...

+ Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG, BMP, WEBP

WhatsApp Image 2026-02-18 at 15.13.09.jpeg 36.6KB ×

Analysis Prompt

Analyze this civil engineering structure image. Provide a detailed breakdown including: type of structure, materials used, estimated dimensions, construction method, notable features, and any engineering challenges visible.

● Analyze Structure

Preview

Uploaded Image

• Output Summary

steel members.

- **Bolted/Welded Connections:** Workers would be securing the steel members using bolted connections (most common for primary field connections due to speed and efficiency) and/or welding.
- **Sequential Phasing:** The construction appears to be progressing in phases, with some sections of the building frame more advanced than others.
- **Working at Height:** Workers are visible on the steel frame, indicating the use of personal protective equipment (PPE) like safety harnesses, and likely temporary access platforms or mobile elevated work platforms (MEWPs) for assembly.

Deploy ⋮

5. Notable Features:

- **Modular Grid Design:** The structure exhibits a highly organized and repetitive grid pattern of columns and beams, which is typical for efficient fabrication, erection, and future space utilization in industrial or commercial buildings.
- **Open Framework:** The structure is currently an open framework, meaning external cladding (walls) and roofing have not yet been installed.
- **Construction in Progress:** The presence of the crane and the unfinished nature of the building clearly indicate an active construction site.
- **Large Spans:** The significant distances between columns suggest the design prioritizes large, unobstructed interior spaces, common for warehousing, manufacturing, or large open-plan offices.

6. Engineering Challenges Visible (or Implied):

- **Stability during Erection:** Erecting such a tall and open steel frame requires careful planning for temporary stability against wind loads, especially before all bracing elements and floor diaphragms are fully connected.
- **Crane Operations and Logistics:** Coordinating the heavy lifts of large steel members, ensuring crane capacity, reach, and stability are adequate for the loads and heights involved.
- **Connection Design and Fabrication Tolerance:** Ensuring that thousands of connections are designed correctly, fabricated accurately, and installed precisely to transfer all structural loads (gravity, wind, seismic). Misalignment can cause significant construction delays.
- **Wind Loading:** The tall, open structure presents a large surface area for wind to act upon during construction, necessitating temporary bracing and careful structural analysis.
- **Worker Safety:** Working at these heights with heavy materials presents significant safety challenges, requiring strict adherence to safety protocols and equipment.
- **Foundation Design (Implied):** For a structure of this scale, the foundation system would need to be robust, considering soil conditions and the substantial loads transferred from the steel frame.

 Download Report

8. ADVANTAGES & DISADVANTAGES

Advantages

- Significant time savings
- Automated structured documentation
- Industry-relevant AI integration
- Cloud deployable architecture

Disadvantages

- Dependent on external AI API
- Requires internet connection

- API latency may vary

9. CONCLUSION

Civil Engineering Insight Studio demonstrates practical integration of multimodal Generative AI into civil engineering workflows. The system reduces manual documentation effort, improves reporting accuracy, and validates scalable AI-powered automation in the construction domain.

10. FUTURE SCOPE

- AI-based risk detection with bounding boxes
- Progress comparison between multiple images
- PDF report export feature
- User authentication & project history
- Cloud auto-scaling deployment

11. APPENDIX

Source Code (app.py):

```
from dotenv import load_dotenv  
  
import streamlit as st  
  
import os  
  
import google.generativeai as genai  
  
from PIL import Image  
  
import io  
  
  
# Load environment variables from .env file  
  
load_dotenv()
```

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

```
genai.configure(api_key=api_key)
```

```
def get_gemini_response(input_text, image, prompt):
```

```
"""
```

Function to get response from Gemini model.

Args:

input_text: User input text

image: List containing image data in the format required by Gemini

prompt: The prompt to send to the model

Returns:

str: The text response from the model

```
"""
```

```
model = genai.GenerativeModel('gemini-2.5-flash')
```

```
try:
```

```
    response = model.generate_content([input_text, image[0], prompt])
```

```
    return response.text
```

```
except Exception as e:
```

```
    return f"Error generating content: {e}"
```

```
def input_image_setup(uploaded_file):
```

```
"""
```

Function to read the uploaded image and format it for Gemini Pro model.

Args:

uploaded_file: The uploaded file object from Streamlit

Returns:

list: A list containing image data formatted for Gemini API

Raises:

FileNotFoundException: If no file is uploaded

""""

if uploaded_file is not None:

Read the file's binary data

bytes_data = uploaded_file.getvalue()

Create image parts in the required format

image_parts = [

{

 "mime_type": uploaded_file.type,

 "data": bytes_data,

}

]

return image_parts

else:

 raise FileNotFoundError("No file uploaded")

```
def setup_page():

    st.set_page_config(page_title="Civil Engineering Insight Studio", layout="wide",
page_icon="💡")  
  
# Custom CSS for styling  
st.markdown("""  
  
    <style>  
  
        .main {  
  
            background-color: #f8f9fa;  
        }  
  
        .stButton>button {  
  
            width: 100%;  
  
            background-color: #007bff;  
            color: white;  
  
            border-radius: 5px;  
            font-weight: bold;  
            border: none;  
            padding: 10px;  
        }  
  
        .stButton>button:hover {  
  
            background-color: #0056b3;  
            color: white;  
        }  
  
        .header-title {  
  
            text-align: center;  
            font-size: 3em;  
        }  
    </style>  
""")
```

```
        font-weight: 800;
        color: #2c3e50;
        margin-bottom: 0.5em;
    }

.sub-header {
    text-align: center;
    font-size: 1.2em;
    color: #7f8c8d;
    margin-bottom: 2em;
}

.feature-card {
    background-color: white;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 4px 6px rgba(0,0,0,0.1);
}

</style>

"""", unsafe_allow_html=True)

def main():
    """Main function to run the Streamlit application."""
    setup_page()

    # Sidebar for configuration and info
    with st.sidebar:
        st.image("https://img.icons8.com/color/96/000000/engineer.png", width=100)
```

```
st.title("Settings & Info")

if not api_key:
    st.error("⚠️ API Key Missing")
    st.info("Please add GOOGLE_API_KEY to your .env file.")

else:
    st.success("✅ API Key Configured")

st.markdown("---")
st.subheader("How to use:")
st.markdown("""
1. **Upload an image** of a civil engineering structure.
2. **Enter a prompt** describing what you want to analyze.
3. Click **Analyze Structure**.
""")  

st.markdown("---")
st.caption("Powered by Google Gemini AI")
```

Main Content Area

```
st.markdown('<div class="header-title">Civil Engineering Insight Studio</div>',
unsafe_allow_html=True)

st.markdown('<div class="sub-header">Advanced AI Analysis for Structural
Engineering</div>', unsafe_allow_html=True)
```

```
col1, col2 = st.columns([1, 1], gap="medium")
```

```
with col1:  
    st.markdown("### 📸 Upload & Prompt")  
    uploaded_file = st.file_uploader(  
        "Choose a structure image...",  
        type=["jpg", "jpeg", "png", "bmp", "webp"],  
        help="Supported formats: JPG, PNG, WEBP"  
    )  
  
    input_prompt = st.text_area(  
        "Analysis Prompt",  
        value="Analyze this civil engineering structure image. Provide a detailed breakdown  
        including: type of structure, materials used, estimated dimensions, construction method,  
        notable features, and any engineering challenges visible.",  
        height=200,  
        help="This prompt guides the AI analysis."  
    )  
  
    analyze_button = st.button("🔍 Analyze Structure", type="primary")  
  
with col2:  
    st.markdown("### 🖼 Preview")  
    if uploaded_file is not None:  
        image = Image.open(uploaded_file)  
        st.image(image, caption="Uploaded Image", use_container_width=True)  
    else:  
        st.info("Upload an image to see a preview here.")
```

```
# Analysis Results Section (Full Width below columns)

if analyze_button and uploaded_file:

    if not api_key:

        st.error("Please configure your API key first.")

    elif input_prompt.strip() == "":

        st.warning("Please enter a prompt.")

    else:

        with st.spinner("🚀 Analyzing structure structure... This may take a moment."):

            try:

                image_data = input_image_setup(uploaded_file)

                response = get_gemini_response(input_text="Analyze the following image:",
                                                image=image_data, prompt=input_prompt)

                st.markdown("---")

                st.subheader("📊 Analysis Report")

                st.markdown(response)

# Download button

st.download_button(

    label="📥 Download Report",

    data=response,

    file_name="structural_analysis_report.md",

    mime="text/markdown"

)
```

```
except FileNotFoundError as e:  
    st.error(f"File Error: {e}")  
  
except Exception as e:  
    st.error(f"Analysis Failed: {str(e)}")  
  
  
if __name__ == "__main__":  
    main()
```

GitHub Repository

<https://github.com/nandinioduri78/Civil-Engineering-Insight-Studio>

Project Demo Link

<https://drive.google.com/file/d/1CvoY4tUqxvjuFWgRchyF8DZicNLf71v8/view?usp=sharing>