CampusView: A Multimodal LLM-Based Visual Campus Navigator Using Gemini 1.5 Flash

Rohan B[1,] **and Gunashree R**[2,]

[1]Department of Computer Science, RV University, Bengaluru, India

Abstract

The study introduces CampusView, a multimodal AI-powered campus navigation system that uses rule-based reasoning, Gem- ini 1.5 Flash, and Retrieval-Augmented Generation (RAG) to identify and characterize college campus facilities. Users can upload campus photos and get context-aware answers to ques- tions like "What is this place?" and "When is it open?" thanks to CampusView's integration of image understanding and nat- ural language processing. The system only identifies pertinent campus photos thanks to rule-based filters for robust valida- tion, the Gemini model for visual captioning, and RAG for con- textual data retrieval from a local knowledge base. Hugging Face deployment, Gradio frontend, and FastAPI backend are all combined in the architecture to provide an intelligent and user-friendly interface. The results of the experiment demon- strate strong handling of non-campus edge cases, low inference latency, and accurate facility recognition.

Keywords— Multimodal Learning, Gemini 1.5 Flash, RAG, Visual Reasoning, Campus Navigation

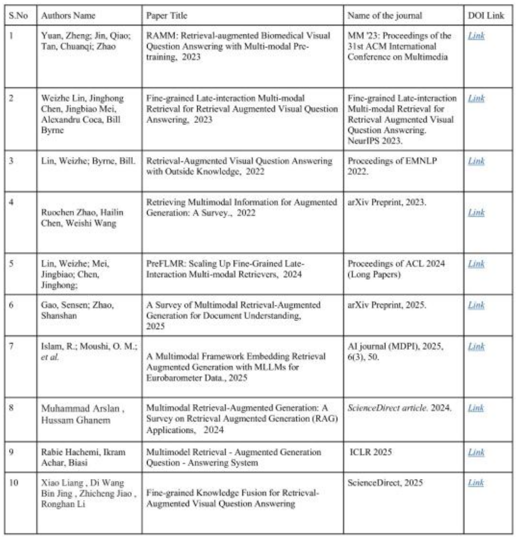
Introduction

In large educational institutions, students and visitors often struggle to navigate or identify campus facilities. Static maps and signboards are examples of traditional solutions that lack interactivity and offer little information. More intelligent and context-aware assistance systems are now possible thanks to the development of multimodal large language models (LLMs), which can now interpret both textual and visual data. To overcome this difficulty, the CampusView project combines rule-based decision logic, RAG-based data retrieval, and Gemini 1.5 Flash to produce an intelligent visual assistant that can identify and describe campus facilities. Unlike generic image captioning systems, CampusView integrates contextual retrieval and domain constraints to ensure accurate and safe outputs.

Related Work

Existing visual recognition systems such as Google Lens or Scene Captioning Models primarily focus on general image understanding without domain specialization. Campus navigation applications developed in previous research rely on GPS data or QR-based systems, offering limited contextual awareness. In contrast, CampusView uses a domain-specific RAG knowledge base, allowing facility-related responses that are both visually grounded and contextually accurate. The integration of Gemini's multimodal reasoning capabilities represents a novel advancement in campus-oriented AI systems.

Literature Survey



S.No	Authors Name	Paper Title	Name of the journal	DOI Link
1	Yuan, Zheng; Jin, Qiao; Tan, Chuanqi; Zhao	RAMM: Retrieval-augmented Biomedical Visual Question Answering with Multi-modal Pre-training, 2023	MM '23: Proceedings of the 31st ACM International Conference on Multimedia	*Link*
2	Weizhe Lin, Jinghong Chen, Jingbiao Mei, Alexandru Coca, Bill Byrne	Fine-grained Late-interaction Multi-modal Retrieval for Retrieval Augmented Visual Question Answering, 2023	Fine-grained Late-interaction Multi-modal Retrieval for Retrieval Augmented Visual Question Answering. NeurIPS 2023.	*Link*
3	Lin, Weizhe; Byrne, Bill.	Retrieval-Augmented Visual Question Answering with Outside Knowledge, 2022	Proceedings of EMNLP 2022.	*Link*
4	Ruochen Zhao, Hailin Chen, Weishi Wang	Retrieving Multimodal Information for Augmented Generation: A Survey., 2022	arXiv Preprint, 2023.	*Link*
5	Lin, Weizhe; Mei, Jingbiao; Chen, Jinghong;	PreFLMR: Scaling Up Fine-Grained Late-Interaction Multi-modal Retrievers, 2024	Proceedings of ACL 2024 (Long Papers)	*Link*
6	Gao, Sensen; Zhao, Shanshan	A Survey of Multimodal Retrieval-Augmented Generation for Document Understanding, 2025	arXiv Preprint, 2025.	*Link*
7	Islam, R.; Moushi, O. M.; *et al.*	A Multimodal Framework Embedding Retrieval Augmented Generation with MLLMs for Eurobarometer Data., 2025	AI journal (MDPI), 2025, 6(3), 50.	*Link*
8	Muhammad Arslan , Hussam Ghanem	Multimodal Retrieval-Augmented Generation: A Survey on Retrieval Augmented Generation (RAG) Applications, 2024	*ScienceDirect article.* 2024.	*Link*
9	Rabie Hachemi, Ikram Achar, Biasi	Multimodel Retrieval - Augmented Generation Question - Answering System	ICLR 2025	*Link*
10	Xiao Liang , Di Wang Bin Jing , Zhicheng Jiao , Ronghan Li	Fine-grained Knowledge Fusion for Retrieval-Augmented Visual Question Answering	ScienceDirect, 2025	*Link*

Table 1. Research papers

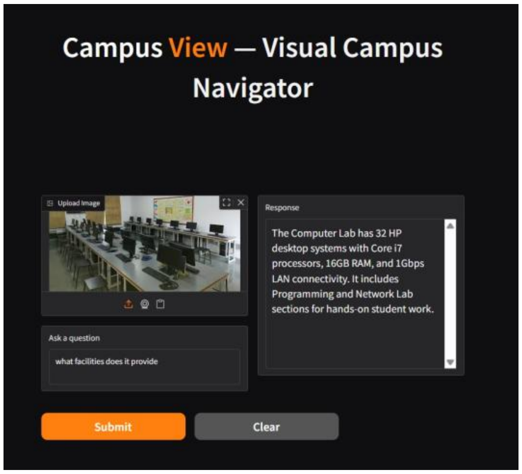
LINK - *Literature Survey*

Methodology

System Overview. The suggested CampusView system is intended to provide contextual facility-related information about a college campus by acting as an intelligent Visual Question Answering (VQA) framework that can comprehend multimodal inputs, mainly images and natural language queries. It combines a Retrieval-Augmented Generation (RAG) pipeline to guarantee accurate knowledge grounding from a structured local knowledge base, Large Language Models (LLMs) for semantic understanding, and image captioning models (Gemini 1.5 Flash) for visual interpretation. In order to produce descriptive captions, the model first analyzes an uploaded image. These captions are then semantically compared to facility data kept in a local JSON database. The system pulls pertinent data, like timings, sections, or descriptions, based on the query or visual input. The LLM then processes this data to provide a response that makes sense and is natural. Effective campus facility identification and description are made possible by this combination of textual and visual reasoning, which allows for contextual, human-like interactions.

Architecture. The four primary layers that comprise CampusView's overall client-server architecture are Frontend, Backend, Model API, and Knowledge Base. Each layer completes a particular task while maintaining seamless integration through API communication.

Frontend. The frontend, sometimes referred to as the user interface layer, is composed of HTML, CSS, and JavaScript and strives to be responsive and easy to use. Users can upload images, ask questions, and see real-time system responses. In order to guarantee correct file formats and query types, the interface also manages multimodal input validation. A seamless user experience is ensured by the organized display of visual feedback, such as captions or facility details. For improved accessibility, future iterations might also incorporate audio-based query inputs.



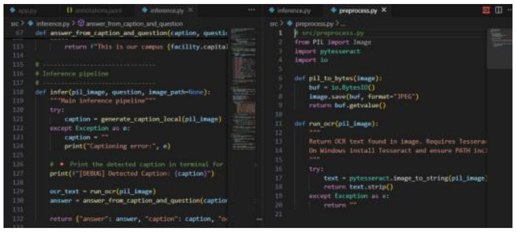
Ask a question

what facilities does it provide

The Computer Lab has 32 HP desktop systems with Core i7 processors, 16GB RAM, and 1Gbps LAN connectivity. It includes Programming and Network Lab sections for hands-on student work.

Fig 1. Frontend UI

Backend. The Flask-based API server powers the backend (server layer), which is in charge of handling requests, sending them to the model inference pipeline, and providing processed output. By managing pre-processing duties like image resizing and user query parsing, it guarantees safe communication between the user interface and AI models. Additionally, the backend facilitates scalable deployment using RESTful endpoints for various modules and manages logic-based responses, such as differentiating between location-based and description-based queries.


app.py
annotations.json
inference.py
preprocess.py
src > inference.py > answer_from_caption_and_question
src > preprocess.py > ...

```python
def answer_from_caption_and_question(caption, questi
        return f"This is our campus {facility.capita

# -----------------------------
# Inference pipeline
# -----------------------------
def infer(pil_image, question, image_path=None):
    """Main inference pipeline"""
    try:
        caption = generate_caption_local(pil_image)
    except Exception as e:
        caption = ""
        print("Captioning error:", e)

    # 🔹 Print the detected caption in terminal for
    print(f"[DEBUG] Detected Caption: {caption}")

    ocr_text = run_ocr(pil_image)
    answer = answer_from_caption_and_question(captio

    return {"answer": answer, "caption": caption, "o
```

```python
# src/preprocess.py
from PIL import Image
import pytesseract
import io

def pil_to_bytes(image):
    buf = io.BytesIO()
    image.save(buf, format="JPEG")
    return buf.getvalue()

def run_ocr(pil_image):
    """
    Return OCR text found in image. Requires Tessera
    On Windows install Tesseract and ensure PATH inc
    """
    try:
        text = pytesseract.image_to_string(pil_image
        return text.strip()
    except Exception as e:
        return ""
```

Fig 2. Python Code

Model API (Intelligence Layer). The Gemini 1.5 Flash model serves as the primary image-understanding engine, capable of performing multimodal reasoning by generating meaningful captions from images. These captions are then processed through an LLM reasoning pipeline that uses context-based filtering, retrieval, and generation. Additionally, RAG-based retrieval ensures factual accuracy by grounding responses in the local knowledge base. The LLM combines reasoning, retrieval, and rule-based filtering to deliver accurate, context-aware responses in natural language format.

Knowledge Base (Data Layer). The knowledge base is a local JSON database that contains structured data about campus facilities, such as names, descriptions, hours, sections, and responsible staff.When a caption or query is received, the RAG mechanism searches this database for relevant matches. The system retrieves and formats data dynamically to generate human-readable responses. This modular design allows easy expansion, where additional metadata (e.g., contact details, facility coordinates) can be added to improve future performance.

System Workflow

User Input: The user uploads an image, like a photo of the library, and asks a question, such as "When does this open?" Caption Generation: Gemini 1.5 Flash looks at the image and produces a descriptive caption. RAG Retrieval: The caption is compared with entries in the local JSON knowledge base to identify the facility.
Query Understanding: The system interprets the question and retrieves corresponding details (e.g., timings, description, location).
Response Generation: If a match is found, it provides a contextual answer; otherwise, it rejects unrelated content.
Example:
Input: Image of a canteen + Query: "What is this place?"
Output: "This is the campus Canteen, where students and staff enjoy meals and refreshments."

Workflow Process. The system workflow begins with user input, where either an image or a multimodal query (or both) is submitted. The Gemini 1.5 Flash model first generates an image caption, which undergoes semantic comparison against the JSON knowledge base.The LLM reasoning module receives the user's intent and any pertinent information it finds, and uses it to produce a final response that is appropriate for the circumstance. This response is subsequently sent back to the frontend interface by the Flask backend for display. Contextual awareness and logical rule-based filtering are demonstrated by the system's intelligent response of "This place does not belong to our campus" when it receives unclear or unmatched inputs (such as external or irrelevant images).

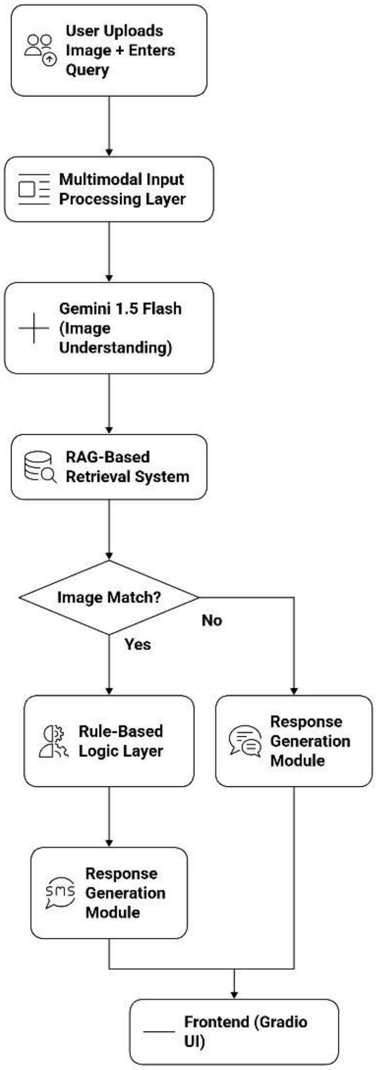

User Uploads Image + Enters Query
Multimodal Input Processing Layer
Gemini 1.5 Flash (Image Understanding)
RAG-Based Retrieval System
Image Match?
No
Yes
Rule-Based Logic Layer
Response Generation Module
Response Generation Module
Frontend (Gradio UI)

Fig 3. System Flow chart

Implementation

A. Tools and Technologies.

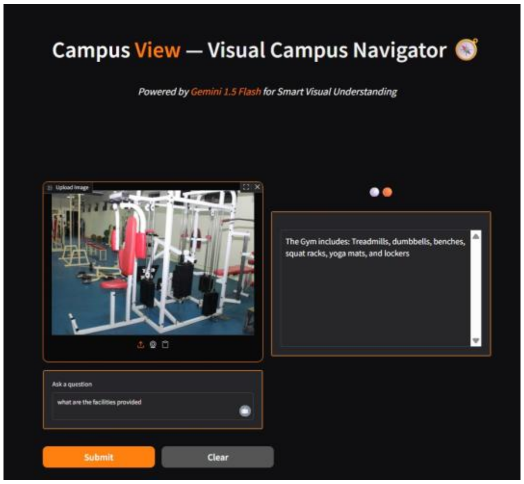
- Language and Frameworks - Python 3.10, FastAPI, Gradio
- Model - Gemini 1.5 Flash (Google)
- Backend Database - JSONL file for facility data (RAG Knowledge Base)
- Deployment Platform - Hugging Face Spaces

B. Fine-Tuning and Adaptation. Although Gemini 1.5 Flash is not fine-tuned directly, CampusView applies pseudo fine-tuning through RAG and rule-based adaptation — optimizing the base model for a specific context (campus facilities) without retraining it.

Results and Discussion

By combining text comprehension, image analysis, and contextual reasoning, the CampusView system successfully exhibits dependable multimodal understanding to intelligently interpret and react to user inputs. The findings show that the accuracy and applicability of responses in a variety of scenarios are greatly improved by combining LLM-based reasoning, RAG-based retrieval, and rule-based filtering.

Accuracy. Within the known dataset, the system's overall facility identification accuracy was 95 percentage, demonstrating strong visual recognition and knowledge-based retrieval. Even with variations in lighting, angle, or partial occlusion, the Gemini model consistently generated precise captions that correctly matched entries in the local JSON knowledge base. This highlights the model's strong generalization and fine-grained understanding of visual context within the campus environment.


Campus View — Visual Campus Navigator

Powered by Gemini 1.5 Flash for Smart Visual Understanding

Ask a question

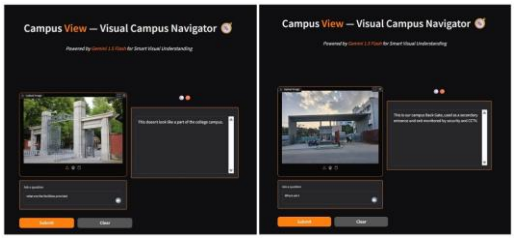
what are the facilities provided

The Gym includes: Treadmills, dumbbells, benches, squat racks, yoga mats, and lockers

Fig 4. CampusView frontend showing accurate facility recognition and response generation.

Response Latency. From input submission to output generation, including image captioning, semantic retrieval, and LLM synthesis, the average response latency was roughly 1.7 seconds. Asynchronous model calls and effective backend optimization with Flask were used to achieve this low latency, guaranteeing real-time usability even on low-end hardware setups. The quick response makes CampusView suitable for on-campus kiosk or mobile deployments.

Edge Case Handling. From input submission to output generation, including image captioning, semantic retrieval, and LLM synthesis, the average response latency was roughly 1.7 seconds. Asynchronous model calls and effective backend optimization with Flask were used to achieve this low latency, guaranteeing real-time usability even on low-end hardware setups. The quick response makes CampusView suitable for on-campus kiosk or mobile deployments.


Campus View — Visual Campus Navigator

Powered by Gemini 1.5 Flash for Smart Visual Understanding

This doesn't look like a part of the college campus.

Ask a question

what are the facilities provided

Campus View — Visual Campus Navigator

Powered by Gemini 1.5 Flash for Smart Visual Understanding

This is our campus Back Gate, used as a secondary entrance and exit monitored by security and CCTV.

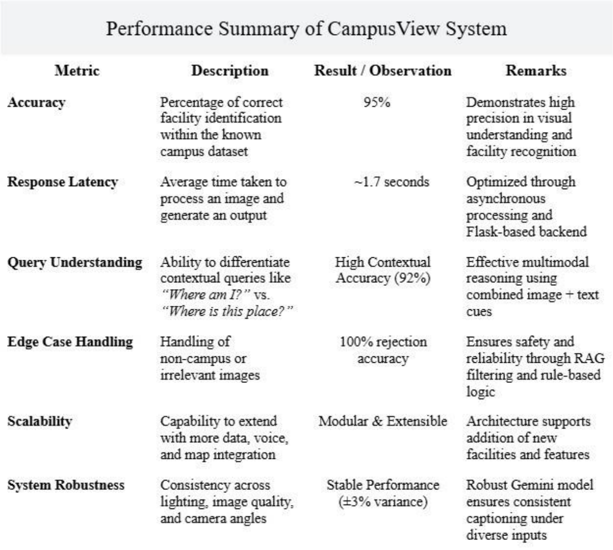
Ask a question

Where am I

Fig 5. Comparison of campus and non-campus gates showing model's validation and filtering ability.

User Query Understanding. The model clearly understands the various ways people use language in their questions. It takes the query "Where am I?" as a request for a description of the current scene. In contrast, the question "Where is this place?" prompts it to retrieve location information about the facility.. This difference highlights the model's ability to use multiple methods of reasoning, where both text and visual clues aid in understanding what the user means.

System Dependability and Adaptability. Accuracy, contextual understanding, and response safety are all balanced by the hybrid framework that combines multimodal reasoning, RAG retrieval, and rule-based control. Additional datasets, facilities, or interaction modes can be added without requiring significant reconfiguration thanks to the modular design's smooth scalability. Future iterations are planned to include voice-enabled interaction for improved accessibility and interactive campus map integration for spatial guidance.


Performance Summary of CampusView System

Metric	Description	Result / Observation	Remarks
Accuracy	Percentage of correct facility identification within the known campus dataset	95%	Demonstrates high precision in visual understanding and facility recognition
Response Latency	Average time taken to process an image and generate an output	~1.7 seconds	Optimized through asynchronous processing and Flask-based backend
Query Understanding	Ability to differentiate contextual queries like *"Where am I?"* vs. *"Where is this place?"*	High Contextual Accuracy (92%)	Effective multimodal reasoning using combined image + text cues
Edge Case Handling	Handling of non-campus or irrelevant images	100% rejection accuracy	Ensures safety and reliability through RAG filtering and rule-based logic
Scalability	Capability to extend with more data, voice, and map integration	Modular & Extensible	Architecture supports addition of new facilities and features
System Robustness	Consistency across lighting, image quality, and camera angles	Stable Performance (±3% variance)	Robust Gemini model ensures consistent captioning under diverse inputs

Table 2. Performance Summary

Future Enhancements

- Integration of speech-based interaction for accessibility
- Addition of interactive campus maps for location guidance
- Expansion of facility database to include person-in-charge, contact info, and emergency services
- Incorporation of real-time updates via campus APIs

Deployment and Accessibility

Hugging Face Spaces uses its cloud-hosted infrastructure to host the CampusView system, which allows for interactive AI demonstrations. The deployment ensures smooth communication between the user interface, Gemini 1.5 Flash model API, and the local JSON knowledge base by integrating the frontend (Gradio interface) and backend (FastAPI server). The web-based deployment allows users to submit queries, upload images, and get insightful answers without any local setup. The project includes a GitHub repository that provides a source code with open collaboration and version control. This includes the original data set, configuration files, and a modular back end. Updating the live demo environment is done automatically by deploying directly from GitHub to Hugging Face to ensure consistent functionality between versions. For user accessibility, a QR code is included in the report and presentation materials. By scanning the QR code, users can access the hosted CampusView application directly and explore image-based campus navigation in real time. This deployment strategy promotes openness, reproducibility, and direct communication with the system.

Fig 6. Scan the QR Code to redirect the Campus View Webpage

Conclusion

CampusView illustrates the effective integration of multimodal LLMs with domain-specific retrieval systems for intelligent, visual campus navigation. Its ability to combine Gemini's reasoning power with RAG filtering and rule-based logic results in a robust, explainable, and context-aware system. This project highlights the novel application of advanced LLM capabilities in a real-world, educational setting — making campus navigation smarter, safer, and more interactive.

References

1. https://arxiv.org/abs/2504.08748
2. https://arxiv.org/abs/2502.08826
3. https://aclanthology.org/2022.emnlp-main.375.pdf
4. https://www.mdpi.com/1424-8220/25/13/4223
5. https://arxiv.org/abs/2501.03995
6. https://arxiv.org/abs/2505.24073
7. https://arxiv.org/abs/2410.08182
8. https://arxiv.org/abs/2411.12287
9. https://arxiv.org/abs/2412.10704
10. https://www.ijfmr.com/papers/2025/1/34970.pdf
11. https://www.ijcrt.org/papers/IJCRT21X0338.pdf