**TITLE OF THE PROJECT**

## A PROJECT REPORT

***Submitted by,***

**Mr. XYZ - 20211CSE0123**

**Mr. ABC -20211CSE0124**

### *Under the guidance of,*

**Dr./Mr./Ms. IJKL**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING, COMPUTER ENGINEERING, INFORMATION SCIENCE AND ENGINEERING Etc.**

**At**



**PRESIDENCY UNIVERSITY**

**BENGALURU**

**DECEMBER 2024**

**PRESIDENCY UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE ENGINEERING**

**CERTIFICATE**

This is to certify that the Project report **“TITLE OF THE PROJECT”** being submitted by “STUDENTS NAMES” bearing roll number(s) “STUDENTS ROLL NUMBERS” in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a bonafide work carried out under my supervision.

|  |  |
| --- | --- |
| **Dr/Mr/Ms. <SUPERVISOR NAME>**  DESIGNATION  School of CSE&IS  Presidency University | **Dr. <HoD Name>**  DESIGNATION & HoD  School of CSE&IS  Presidency University |

|  |  |  |
| --- | --- | --- |
| **Dr. L. SHAKKEERA**  Associate Dean  School of CSE  Presidency University | **Dr. MYDHILI NAIR**  Associate Dean  School of CSE  Presidency University | **Dr. SAMEERUDDIN KHAN**  Pro-Vc School of Engineering  Dean -School of CSE&IS  Presidency University |

**PRESIDENCY UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE ENGINEERING**

**DECLARATION**

We hereby declare that the work, which is being presented in the project report entitled **TITLE OF THE PROJECT** in partial fulfillment for the award of Degree of **Bachelor of Technology** in **Computer Science and Engineering**, is a record of our own investigations carried under the guidance of **SUPERVISOR NAME, DESIGNATION,** **School of Computer Science Engineering & Information Science, Presidency University, Bengaluru.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

|  |  |
| --- | --- |
|  | **Name(s), Roll No(s) and Signature(s) of the Students** |

**ABSTRACT**

The Image Analysis Chatbot project is a full-fledged online system, which changes static image interpretation to interactive conversation with the help of artificial intelligence. In response to existing limits in image analysis tools that present only static descriptions, the Image Analysis Chatbot allows users to post pictures and involve themselves in a fact-based dynamic conversation. The model has the FastAPI in-built architecture in the backend, the JavaScript module running on the front end, an algorithm that optimizes the images, and is configured to liaise with a Large Language Model (LLM) through an RAG system.

Images go through quality testing, resizing, and compression before any analysis is done. In the AI prompts, detailed visual descriptions, which are trailing facts, are embedded to assure that further user queries are still based on what is actually given in the uploaded content. Session memory works on the hybrid model-where in-memory caching is employed for active chats to be taken up inside the Python of ChatKey and JSON stabbed in another for strength.

Test figures showed less latent time, more scalable performance, and over 92% factual response accuracy. End users' experience assessments validated the ease of using it with high responsiveness for all connected devices. Furthermore, the AI ethics and accessibility were strongly put at the forefront of enhancing liveness and credibility for the system. With technologically robust systems development, multimodal AI gained much-needed leverage by demonstrating how retrieval-based grounding considerably enhances conversational confidence. The platform preempts further expansion in this multimodal capacity, handling several images, cross-modal grounding, and collaborative conversational environments due to its flexible design, ethical stance, and high scalability.

**Keywords:** image analysis chatbot, retrieval-augmented generation (RAG), conversational AI, session memory management, FastAPI web framework, large language models (LLMs), multimodal AI systems

**ACKNOWLEDGEMENT**

First of all, we indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected dean **Dr. Md. Sameeruddin Khan**, Pro-VC, School of Engineering and Dean, School of Computer Science Engineering & Information Science, Presidency University for getting us permission to undergo the project.

We express our heartfelt gratitude to our beloved Associate Deans **Dr. Shakkeera L and Dr. Mydhili Nair,** School of Computer Science Engineering & Information Science, Presidency University, and Dr. “NAME OF THE HOD”, Head of the Department, School of Computer Science Engineering & Information Science, Presidency University, for rendering timely help in completing this project successfully.

We are greatly indebted to our guide **Dr./Mr.Ms. Name, Designation** and Reviewer **Dr./Mr.Ms. Name, Designation**, School of Computer Science Engineering & Information Science, Presidency University for his/her inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We would like to convey our gratitude and heartfelt thanks to the PIP2001 Capstone Project Coordinators **Dr. Sampath A K, Dr. Abdul Khadar A and Mr. Md Zia Ur Rahman,** department Project Coordinators “NAME” and Git hub coordinator **Mr. Muthuraj.**

We thank our family and friends for the strong support and inspiration they have provided us in bringing out this project.

**Name of the Student (1)**

**Name of the Student (2)**

**Name of the Student (3)**

**Name of the Student (4)**

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No.** | **Figure Name** | **Caption** | **Page No.** |
| 1 | Figure 4.1 | Backend System Architecture | 20 |
| 2 | Figure 4.2 | WorkFlow | 26 |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT ACKNOWLEDGMENT**  **…** | **i**  **ii**  **…** |
| **1.** | **INTRODUCTION** | 1 |
|  | 1.1 Overview of Image Analysis and Chatbots | 1 |
|  | 1.1.1 Evolution of Image Understanding | 2 |
|  | 1.1.2 Rise of Conversational AI | 2 |
|  | 1.2 Need for an Image Analysis Chatbot | 3 |
|  | 1.3 Scope and Importance of the Project | 4 |
| **2.** | **LITERATURE REVIEW** |  |
|  | 2.1 Evolution of Image Analysis | 7 |
|  | 2.2 Progress in Conversational AI | 8 |
|  | 2.3 Memory Architectures in AI Systems | 10 |
|  | 2.4 Retrieval-Augmented Generation (RAG) Methods | 11 |
| **3** | **RESEARCH GAPS OF EXISTING METHODS** |  |
|  | 3.1 Gaps in Image Analysis Systems | 13 |
|  | 3.2 Gaps in Conversational AI Systems | 14 |
|  | 3.3 Gaps in Memory and Session Management | 15 |
|  | 3.4 Gaps in Retrieval-Augmented Generation (RAG) Applications | 15 |
|  | 3.5 Overall Need for an Improved System | 17 |
| **4** | **PROPOSED MOTHODOLOGY** |  |
|  | 4.1 Overview of the Proposed System | 18 |
|  | 4.2 Backend System Design | 19 |
|  | 4.3 Frontend System Design | 20 |
|  | 4.4 Session Management and Data Persistence | 21 |
|  | 4.5 AI Model Interaction and Prompt Engineering | 22 |
|  | 4.6 Retrieval-Augmented Generation (RAG) Framework | 23 |
|  | 4.7 Performance Optimization and Scalability | 25 |
|  | 4.8 Summary of the Proposed Methodology | 27 |
| **5** | **OBJECTIVES** |  |
|  | 5.1 Introduction to Objectives | 28 |
|  | 5.2 Primary Objectives | 28 |
|  | 5.3 Secondary Objectives | 29 |
|  | 5.4 Expected Impact of Achieving Objectives | 30 |
| **6** | **SYSTEM DESIGN & IMPLEMENTATION** |  |
|  | 6.1 Introduction | 32 |
|  | 6.2 Backend System Design and Implementation | 32 |
|  | 6.2.1 API Layer | 32 |
|  | 6.2.2 Processing Layer | 33 |
|  | 6.2.3 Session Memory Management | 33 |
|  | 6.3 Frontend System Design and Implementation | 34 |
|  | 6.3.1 User Interface Design | 34 |
|  | 6.3.2 Client-Side Data Handling | 34 |
|  | 6.4 Image Processing Module | 34 |
|  | 6.5 Session and Memory Module | 35 |
|  | 6.6 AI Interaction and RAG Implementation | 36 |
|  | 6.7 Data Storage and Management | 36 |
|  | 6.8 Deployment Architecture | 37 |
|  | 6.9 Summary | 37 |
| **7** | **TIMELINE FOR EXECUTION OF PROJECT** | 38 |
| **8** | **OUTCOMES** | 39 |
| **9** | **RESULTS AND DISCUSSIONS** |  |
|  | 9.1 System Functionality Validation | 41 |
|  | 9.2 Image Processing and Optimization Results | 41 |
|  | 9.3 AI Response Accuracy and RAG Performance | 42 |
|  | 9.4 Frontend and User Experience Observations | 42 |
|  | 9.5 System Performance and Scalability | 43 |
|  | 9.6 Discussion and Implications | 43 |
| **10** | **CONCLUSION** | 44 |
|  | **REFERENCES** | 46 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**CHAPTER-1**

**INTRODUCTION**

**1.1 Overview of Image Analysis and Chatbots**

The continual evolution of AI has brought a remarkable transformation in the manner machines see or think about the world. It is by bridging the gap between the AI world and the real world that computer vision gives an AI system the ability to extract meaningful information from images and videos. From early attempts at identifying simple shapes to today's sophisticated deep learning models capable of scene-level understanding, remarkable advancement in the level of field has occurred. Computer vision technologies are deeply embedded now in applications such as autonomous vehicle operation, medical diagnosis, control quality in manufacturing, and social media platform-based applications use.

However, most of the traditional image analysis system involves a one-way channel where the system receives the visual feed and outputs fixed responses like labels, captions, or bounding boxes, without any flexibility for the user to interact with the system and ask specific questions or explore more aspects of the analyzed image in a dynamic way. Simultaneously, major advancements in natural language processing (NLP) have produced effective conversational agents. The Large Language Models (LLMs) have shown that they would be able to understand and generate human-like responses about various activities in different contexts. These models have moved beyond the restrictive boundaries of rule-based setups to make possible natural, fluid, and contextually aware interactions.

The new revolution in computer vision and conversational artificial intelligence is in the multimodal systems which "see", "understand," and "talk" about what they have perceived. Intelligent platforms are therefore possible by fusing the strengths of both fields for uploading photos and engaging in meaningful and repeated conversation on the visual content.

Thus, the Image Analysis Chatbot is developed which is intended to fill a need in the form of dynamic interaction for the static analysis of an image, thus rendering a more interesting user experience.

**1.1.1 Evolution of Image Understanding**

The development of image understanding has passed through various stages, with each borrowing selectively, creatively, and gainfully from the strengths and limitations of those before it. Early systems were based on low-level feature extraction methods that mainly worked on edges, corners, and simple geometric patterns. Sobel operators and Harris corner detectors were truly the pioneers of computer vision.

The Machine Learning epoch saw the rise of supervised models such as SVMs. During this age, the objects learned from labeled datasets but heavily relied on handcrafted feature extraction, which affected their scalability and adaptability to different image types.

This deep learning technology, especially CNNs, brought about the transformation of its own. CNNs removed the need for hand-tuning of features as they could automatically learn hierarchical representations of visual data. Landmark models like AlexNet, VGGNet, and ResNet attained breakthrough results in the tasks of image classification, object detection, and segmentation.

More recently, interest has begun to turn to the investigation of context and relationships in images. The newer models include spatial relations, scene depictions, emotional connotations, and even inferred potential actions from a static image. However, static outputs are typically the output of the highest-performing and sophisticated models, with very limited opportunities for user-driven dynamics.

The demand for making analysis of images more interactive, explorative, and user-centric remains the core problem that projects such as the Image Analysis Chatbot seek to address.

**1.1.2 Rise of Conversational AI**

Conversational AI grew from rudimentary rule-based systems to highly dynamic and responsive in virtual agents. Early chatbots were extremely limited in that they responded to pre-programmed patterns of behavior for specific input keywords, thus serving useful functions in simple tasks but never capable of dealing with more subtle queries or carrying on a coherent dialogue over extended engagements.

The advent of LLMs thereby cloaked the users and clients on conversational agents, fostering smooth and adaptive conversations across multi-topic whirls. The underlying principle in their training consists of ingesting vast datasets, i.e., datasets comprising several linguistic structures, domains, and contexts, allowing them to understand user intent and conversation history with generating suitable humane responses.

Conversational AI, powered with computer vision, gives an exciting flavor. Now the systems understand the content of the image and converse with users about the image with depth, answering follow-up questions, elucidating subtleties, and tailoring responses as per the changing context of the conversation.

This fusion of modalities greatly augments user experience, changing passive visual systems into dynamic conversational partners that enable deeper understanding and exploration.

**1.2 Need for an Image Analysis Chatbot**

While some progress has been made in computer vision and NLP, the majority of image analysis software remains static: once an image has been processed, it will produce a set of outputs and terminate the interaction. In the more complex domains where human users seek more discerning insight e.g., in medical imaging, forensic analysis, and quality inspections this one-way interaction proves to be quite limited.

Several critical limitations persist in current systems:

* Inflexible output: Users cannot ask follow-up questions or request clarifications.
* Context loss: Static output fails to appreciate dense interrelationships and context nuance within images.
* User dissatisfaction: Lack of dynamic exploration opportunities has limited engagement and reliance on analysis by hand.

The newly arising requirements for a system would imply the need to analyze visual content and become an intelligent conversational partner to interact and talk about various visual elements interactively and in truth.

The Image Analysis Chatbot addresses this need by enabling users to:

* Upload an image and get a full primary assessment.
* Ask detailed questions about the objects, the background, the colors, or the relationships seen in the image.
* Get answers that are strictly based on factual visual content.
* Talk in fluid multi-turns for deeper exploration.
* Experience a completely seamless, user-friendly interface designed to be responsive and accessible.

By incorporating an intelligent conversational layer atop visual analysis, the chatbot transforms the interaction from a static interpretation to an engaging, evolving dialogue.

**Core Capabilities Expected from Modern Interactive Image Analysis Systems:**

* Dynamic Query Processing: Provision of intelligent answers to all varieties of changing user questions.
* Factual Grounding: Questions are answered based on extracted image feature evidence and not speculation or fabrication.
* Extensive Contextual Awareness: Understanding relationships between objects and their spatial configuration within images.
* Intuitive User Interface: Designing a smooth and responsive experience that promotes exploration without being burdened with technical barriers.
* Scalability: Provide environments for multiple users, sessions, and images, with little degradation in performance.

**1.3 Scope and Importance of the Project**

Designed to a scalable, adaptable platform like different user requirements in visual exploration, the Image Analysis Chatbot covers the complete pipeline-from image ingestion and processing to AI-driven description generation and into dynamic session-based conversational engagement.

**System Scope**

* Backend Development: Building a scalable API server that supports user sessions, image uploads, connections to AI models, and efficient delivery of processed results.
* AI Model Integration: Large language model (LLM) for providing detailed and structured image descriptions, along with context-based interactive conversation responses.
* Data Storage Strategy: A hybrid storage solution for fast (<1ms) operations through the use of in-memory caches while allowing persistence through disk-based storage technique.
* Frontend User Experience: Responsive web application through which users can chat in real-time with an AI system, preview images, manage sessions, and experience a modern GUI that is resizable and functional across devices.
* Security and Validation: Strong validation of file types, user input formats, session management, and API security to protect the integrity of system functionality and user privacy.
* Performance Optimizations: The integration of techniques such as image compression, lazy loading, memory management, and local caching would improve responsiveness and reduce the load on the server.

**Importance and Applications**

The potential applications of the Image Analysis Chatbot extend across a wide range of domains:

* Education: This provides a tool to students and scholars to explore and discuss educational images and artwork, scientific visuals, and so on.
* Healthcare: Facilitate medical professionals' conversations for diagnosis by investigating radiology scans or pathology images with intent.
* Security and Investigation: Enable dynamic questioning to facilitate the meticulous interrogation of surveillance imagery or forensic evidence.
* Retail and E-Commerce: Improve analysis of product imagery by having customers ask questions about product features, settings, and aesthetics.
* Creative Arts and Media: Help artists, photographers, and media professionals imagine and compose tales and critiques about the visual artworks.

The chatbot’s modular design ensures that it can be extended easily to incorporate features such as:

* **Multi-image comparisons** across different uploads.
* **Annotation and collaboration tools** for team-based image analysis.
* **Mobile platform deployment** for on-the-go visual exploration.
* **Offline functionality** for environments with limited network access.

By facilitating a dynamic, conversational relationship with visual data, the Image Analysis Chatbot represents a major step forward in making AI systems more interactive, insightful, and accessible to a broader audience.

**CHAPTER-2**

**LITERATURE SURVEY**

**2.1 Evolution of Image Analysis**

The foundation of image analysis could be traced back to the infancy of computer vision during which a majority of researchers were engaged in algorithmic means of modeling human visual perception. Techniques during this period were mainly low level image processing operations. The initial attempts of edge detection using Sobel, Prewitt, and Canny operators marked the process of structured information extraction from pixel grids. These early systems were capable of recognizing simple patterns; however, they were unable to tackle the complexity nature adds to images.

Then other advancements were introduced, notably machine learning algorithms using supervised databases. Classifiers such as Support Vector Machines (SVMs) and k-Nearest Neighbors (k-NN) rely on well-studied handcrafted features such as Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT). Although they improved classification performance, these models were found wanting with regard to flexibility and scalability due to their reliance on feature engineering.

The watershed moment came with the invention of convolutional neural networks, as typified by the monumental triumph of AlexNet in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) of 2012 [1]. These networks made it possible to learn features directly from raw pixels in an end-to-end fashion, avoiding the need for manual engineering while achieving superior performance across all tasks such as classification, detection, and segmentation.

In addition, architectures such as Mask R-CNN and U-Net brought deep learning to the realm of pixel-level predictions, giving rise to such fields as medical imaging, where very important tasks like segmentation of tumors and organs must be performed accurately. Second, ResNet introduced the notion of residual learning, which passed much of the processing burden for ultra-deep networks from the vanishing gradient problem during training.

Dualism or contrastive learning with CLIP [1] emerged as the ability to align visual and textual representations. It has been shown that by training images and text together using a contrastive loss function, visual features were learned that are powerful and transferable, yet bear no relation to any explicit task or supervision. This revolution that gave birth to cross-modal applications like image captioning, zero-shot classification, and visual question answering forms the secondary stream. Under the auspices of Visual ChatGPT [2], it was taken even further to incorporate multiple foundation models capable of drawing, editing, and conversing about images, giving the first glimpse of future multimodal systems.

In addition, BLIP-2 [3] and LLaVA [4] frameworks freezing visual encoders and connecting them to language model pathways allow for great efficiency with little fine-tuning and, hence, less computational cost while still maintaining performance with the utmost level across vision-language benchmarks. Lately, this constitutes an advanced paradigm shift for systems able to perform holistic interpretation, integrating spatial, semantic, and contextual reasoning to arrive at unity.

In application-wise, these are the technologies set to transform industries. In health applications, AI-supported imaging systems help in early cancer detection, employing nearly human sensitivity to analyze mammograms. In the automotive sector, object recognition and scene segmentation models are the underlying technologies for autonomous driving, where real-time correct visual understanding translates into life or death. Social media platforms are heavily adopting facial recognition and visual filters, dictating the interactions of billions on a daily basis. Thus, the so-called transition from pixel-wise to deep multimodal understanding in itself is a profound story of computer vision, along with a solid recognition of the amalgamation of conversational AI in image analysis workflows.

**2.2 Progress in Conversational AI**

Conversation AI has come through to scripted interaction to complex dynamic systems. In the early years, chatbots like ELIZA worked on the basis of very rigorous rule sets, rising and matching incoming data strings to rigidly programmed responses while having a context-free understanding of what had happened previously. Long-term coherence and relevance were problematic even for these models.

The revolution came with the transformer architecture along with the launch of large-scale language models, LLMs likeGPT-2 and GPT-3. Trained on huge and varied datasets, these models were able to perform widely on multiple levels in terms of understanding of natural language, generation and reasoning. Building on these capabilities, GPT-4 [5] could also engage in coherent multi-turn conversation with even rudimentary reasoning about user queries.

However, not always, early outputs from these models were to be relied upon entirely. Instruction-tuned models such as InstructGPT were devised by Ouyang et al. [6] to overcome such challenges. It utilized human feedback to align model behavior with user expectation, ethical rules, and factual correctness. It would render the model both more helpful and less unsafe or hallucinatory.

Thoppilan et al. [7] presented LaMDA, a dialogue-only model designed to sustain longer-term context in incognizant conversations about any subject. LaMDA was designed to enable control, safety, and grounding in long conversations with human speakers.

Zhao et al. [8], on the other hand, provided a detailed analysis of the trend towards LLMs and how such LLMs have changed by parameter scaling, diversification of training data, and the fine-tuning techniques used. Their survey showed that larger size alone is not sufficient to realize better conversational performance: a lot of work is needed in data curation, feedback loops, and retrieval augmentation to improve the quality of dialogue.

More and more, contemporary conversation systems are also heading for a multi-modal future. For instance, Visual ChatGPT [2] and LLaVA [4] parse understanding of vision and language, thus allowing users to converse with natural voices about visual information.

Real-world implementation of these systems has exponentially speeded up adoption by customer service, education, healthcare, and entertainment industries. Everyday interactions now involve many millions through these AI-driven virtual assistants, further attesting to the maturation and effects of conversational AI technology.

However, there continue to be challenges, such as ways of reducing bias, increasing robustness to adversarial input, and long-term memory across conversation sessions-things that memory augmentation and RAG frameworks aim to address.

**2.3 Memory Architectures in AI Systems**

Facilitating adequate memory management is necessary for having rational multi-turn communication in AI systems. Unlike conventional dialogue systems that focus on the immediate previous message, modern systems need structured session-based memory management during longer interactions. Elementary memory architectures utilize in-memory dictionaries for temporarily storing conversation states, thus allowing the rapid retrieval of context pertaining to open sessions. Although this allows for fast retrieval of state information within active sessions, it does not persist across server restarts or crashes, making a strong argument for file-persistence mechanisms, using JSON format or lightweight databases, to facilitate long-term sessioning.

Now, custom memory systems are a lot of human cognitive models that mimic the structures of Short-term memory, Working memory, and Long Term memory [14]. AI systems' short-term memory is mainly engaged in recent dialogue history, while long-term memory is meant to include pertinent facts or user preferences or any knowledge that becomes apparent over the course of that session.

With large-scale retrieval-augmented models, external memory is granted to big document databases. This increases the ability of the model to "forget" and "remember" not to either disregard internal memory constructs or poorly fetching external snippets. Atlas [10] advanced this model by training retrieval and generation jointly, which worked well with a small model size for better factuality and generalization.

A big part of memory architecture is the management of state transitions. For instance, when a user imports another image, the system must first gracefully transition the conversational context from the previous image to the new image, clearing out some stateful memory elements while still holding on to user preferences or user-defined settings.

Particularly the session persistence becomes a hard problem given a multi-user environment needing isolation and protection against seepage amongst simultaneous session occurrences. Lightweight techniques such as UUID-based session keys combined with in-memory session managers have proven quite effective for scaling web-based designs.

Custom architecture like Image Analysis Chatbot adopts a hybrid architecture:

* **In-memory dictionary** for fast lookup of active sessions.
* **File-based storage** for chat history persistence.
* **Browser-side caching** (e.g., localStorage) for offloading server load.

This layered memory management ensures that conversations remain fluid, resilient to failures, and capable of adapting dynamically to evolving user interactions.

**2.4 Retrieval-Augmented Generation (RAG) Methods**

Retrieval-Augmented Generation (RAG) proved to be a core technology for grounding conversational agents in external knowledge. Traditional language models only use their parameters pre-trained to generate output, making these systems susceptible to knowledge obsolescence and factual inaccuracies. RAG systems alleviate these two problems by dynamically retrieving relevant documents during inference.

Nakano et al. [11] concretely showed this phenomenon using WebGPT where a language model conducts web searches to respond to queries factually. The model outperformed non-retrieval models on tasks that required real-world knowledge.

The typical RAG pipeline involves:

* **Embedding user queries** into a vector space using models such as Sentence-BERT.
* **Searching a knowledge base** (e.g., FAISS, Milvus) to retrieve top-k relevant documents.
* **Augmenting the LLM prompt** with the retrieved context.
* **Generating a final response** grounded in the retrieved information.

Many of these practices have since been standardized by OpenAI [12] and LangChain [13], thereby making RAG pipelines modular, scalable, and easy to integrate in production systems.

RAG works a little differently in the case of image analysis chatbots-it stores the long explanation that comes with uploading the image as its "knowledge base." When the user asks questions, the query is enriched with the already stored one, hence making the answer base solely on what can be seen in the image.

Semantic retrieval methods use context embeddings rather than keyword matching to augment retrieval. This means that even when the user phrases a question using a different expression or implication, there will be a retrieval featuring the relevant answer from the stored description.

RAG, as well as enhancing the grounding of facts, is also able to effectively "extend the memory" of AI systems beyond the limits imposed by the internal parameters, supporting conversations that can be kept current, verified, and domain-specific.

By harvesting the fruit of merging retrieval and generation, RAG systems come up with a very different balance between flexibility and accuracy, in which great applications-such as Image Analysis Chatbot- could, as a rule, spend their investments.

**CHAPTER-3**

**RESEARCH GAPS OF EXISTING METHODS**

**3.1 Gaps in Image Analysis Systems**

All of these gains in computer vision notwithstanding, it is safe to say that the current generational models of image analysis have some drawbacks in terms of use and application in the real life disorganization of activity processing. Most vision systems are designed to generate outputs as still representations-deliberate object labels, bounding boxes, or brief textual captions-which should correspond to what has received processing from them in the form of an image. The outputs probably reflect the basic elements of visual identification, but are devoid of the further contextual relationships and nuances that complex scenes have.

The most salient of these gaps is the lack of interactive visual interpretation. Models created until now never allow people to turn up the possibilities of doing engaged or participatory exploratory actions like posing a question along the lines of: "What about this part of the image?" or: "Could you clarify this part that's semantically ambiguous?" It is in such an area that the understanding goes even deeper-it fails in healthcare diagnostics and similar settings, security surveillance, or very fine design analyses.

Another limitation which will be discussed is generalization. Fine-tuning a lot of vision models to a particular dataset often backfires as the model employs overfitting and performs badly on the diverse, real-world visual content generated under exposure. Indeed, important strides have been made in generalizing across domains through contrastive models such as CLIP [1]; however, not even these types of systems are inherently built with continuous session user interaction in mind.

Whereas today, the image analysis systems again are not even semantically temporal, explaining only observations in snapshots without allowing users to observe the historical context across different queries. In fact, it limits the flows of conversation that could be built around images because these were the systems' limitations in applying them in areas like education, research, and critical decision-making environments.

Thus, in all, it can be said that the existing image analysis systems deal with predominantly "one-shot" kind of interpretations without showing any potential for delivery of the increasingly concerned iterative, conversational, and memory yet rich interaction levels that the users would wish.

**3.2 Gaps in Conversational AI Systems**

The advances that have been made in the technology of conversational AI development in large language models are truly impressive but are beset with several remaining gaps when brought to bear on multimodal applications such as image conversations. One persistent challenge is contextual grounding. Most conversational agents work from the textual context with little knowledge of the external world: about visual inputs or verifiable facts. Conversations on complicated or unfamiliar topics quickly spiral into wobbly or utterly false speculations, detracting from the user's trust. Instruction-tuned models like InstructGPT [6] have enhanced user intent alignment, but unless grounded directly to real-world references such as uploaded images, these systems cannot provide any assurance about their factual correctness. With direct retrieval or external memory augmentation, even state-of-the-art systems like LaMDA [7] and GPT-4 [5] still struggle to anchor their answers to the visual content.

Memory continuity is yet another glaring breach. If current conversational agents can still retain conversation histories for a short duration within a session, very soon such retention will become rare. Under such inapplicability, the user experience is thus disjointed, particularly when users interact with the system intermittently over time, expecting it to "remember" what they discussed before.

They also focus on being long-winded, drifting off-topic from the user's original inquiry, especially during technical or detail-oriented communications. Such inefficiency stands in the way of highly focused conversations requiring facts: photo analysis and forensic examination.

Thus, the current models of conversational AI, being immensely powerful for open-ended chats, are often far from being optimized for focused, grounded, and memory-persistent multimodal conversations with images.

**3.3 Gaps in Memory and Session Management**

Memory architecture is one of the vital aspects of coherent, dynamic interactions; however, it remains one of the least developed areas in mainstream AI deployment. Most commercial chatbots and conversational AIs have memory that works as an ephemeris; that is, once the session is done, all context information is lost, unless programmed to specify otherwise. This forces the users to repeat some of their previous input when they happen to repeat a conversation with the machine because the memory makes continuity impossible.

Real-world deployments have been inspired by IT-based systems; such efforts include RETRO [9] and Atlas [10], which have demonstrated retrieval-based memory utility, but, expensive as they are in computation, these approaches rarely enter the domain of lightweight or real-time application.

In practice, session management is typically done using a simple in-memory dictionary or very basic cache storage. These types of session management work quite well for short sessions but do not scale to applications that need very solid, personal memory across sessions and very different applications.

Key gaps identified include:

* **Lack of persistent session tracking** beyond active server memory.
* **Inefficient session migration** when scaling across distributed systems.
* **Absence of multi-modal context binding**, where visual and textual memories are jointly maintained.

For a sophisticated image analysis chatbot, efficient memory management is vital to enable users to switch between images, retain conversation history, and revisit prior analyses without data loss or repetition.

**3.4 Gaps in Retrieval-Augmented Generation (RAG) Applications**

Retrieval-Augmented Generation (RAG) has taken great strides in terms of the factuality and grounding associated with conversational AI. However, the efficient application of RAG in multimodal contexts, including image-based inputs, is still relatively unexplored. Most RAG systems have been optimized for textual retrieval where the query and documents are pure text. Extending RAG into visual contexts-that is, where the "knowledge base" is made up of descriptive representations from images-brings about unique challenges.

Specific gaps include:

* Mismatch in Knowledge Representation: Transforming the dense visual information into textual formats that would be friendly for retrieval at the same time losing semantic richness happens to be a nontrivial exercise Simple captions for images can hardly convey intricate descriptions of scenes; emotions; relationships, etc., that are otherwise required for effective retrieval.
* Query Augmentation Challenges: Not all user queries related to images will directly correspond to stored descriptions, thus necessitating some complex semantic search mechanisms as opposed to mere keyword retrieval.
* Latency Limits: Real-time retrieval from large corpora and on-the-fly prompt augmentation for LLMs will introduce delays in the user experience.
* Context Window Limitations: Very large descriptions of a single image or multi-image knowledge bases could exceed the prompt size limits of current LLMs and hence make deployment of RAG very difficult.

On one hand, WebGPT and OpenAI's retrieval guidelines are general schemes for augmenting language models with external information; on the other hand, such systems should be adapted specifically to the much more unique requirements related to retrieval-based visual conversations.

In addition, RAG pipelines are expected to have some better contingency procedures for cases when retrieval has no or incomplete relevant result. The absence of this type of robustness quickly makes the user distrust the system through long conversations.

So while this is an interesting first step, the improvements necessary to unleash the full-fledged power of RAG concerning multimodal, image-centered AI applications are very much ongoing.

**3.5 Overall Need for an Improved System**

Given the cumulative gaps in current image analysis systems, conversational AI agents, memory architectures, and RAG implementations, there is a clear necessity for an integrated, holistic solution.

Such a solution must seamlessly combine:

* + Dynamic image analysis able to give intricate contextual interpretations.
  + Conversational interfaces with the image-derived knowledge base.
  + Long-term coherence due to the persistence of session-based memory.
  + Multimodal adaptations of retrieval-augmented generation pipelines.
  + Lightweight and efficient backends for real-time interaction for multiple users without any latency or data loss.

The Image Analysis Chatbot project is indeed designed to meet the intersecting needs involved in having a user interact naturally with images, asking various complex questions, while receiving correct, grounded, and conversationally coherent answers.

By leveraging advances in computer vision, conversational AI, memory systems, and retrieval augmentation, the project wants to raise the standard for interactive and intelligent visual exploration platforms.

**CHAPTER-4**

**PROPOSED MOTHODOLOGY**

**4.1 Overview of the Proposed System**

Completely revolutionizing conventional chatbots from providing access to an image, let alone having a conversation about the image uploaded, through to actual grounded, factually correct answers-an answer based on actual visual analysis and not assumptions-where the person would, therefore, say apropos what was asked.

Underneath everything, what's actually going on is a backend powered by FastAPI and a light-weight, responsive frontend developed using vanilla javascript and HTML5/CSS. The backend takes care of image upload, session management, communication with AI, and permanent storage, and the frontend is responsible for managing the user interface, displays the uploaded images, the history of sessions, and the chatting activity in real-time.

Because of the quality of the memory architecture and retrieval-augmented generation (RAG) support, the proposed system becomes different from other existent systems. We find the Image Analysis Chatbot distinct because - unlike all other bots that lose memory after its first conversation - it maintains a structured session memory. Each conversation with the user is not based solely on what he is saying, but it has a strong and internalized description of the uploaded image as it was generated at the time of first processing. This important aspect guarantees that AI answers are wholly tied to the visual content, significantly reducing the hallucination risks inherent to many large language models.

The modularity of the architecture thus could easily allow future improvements, like multi-user capability, an extended mobile application, comparative image analysis, and so on, without upsetting much of the whole. It thus makes up a very modular solution with components that can easily be upgraded independently, such as frontend interface, backend API services, and AI integration layers.

In a nutshell, the proposed system is a bridge, well designed, between vision perception and natural language interaction, built to strong software architecture, best practices in AI, and, ultimately, an intelligent, interactive and highly accurate user experience.

**4.2 Backend System Design**

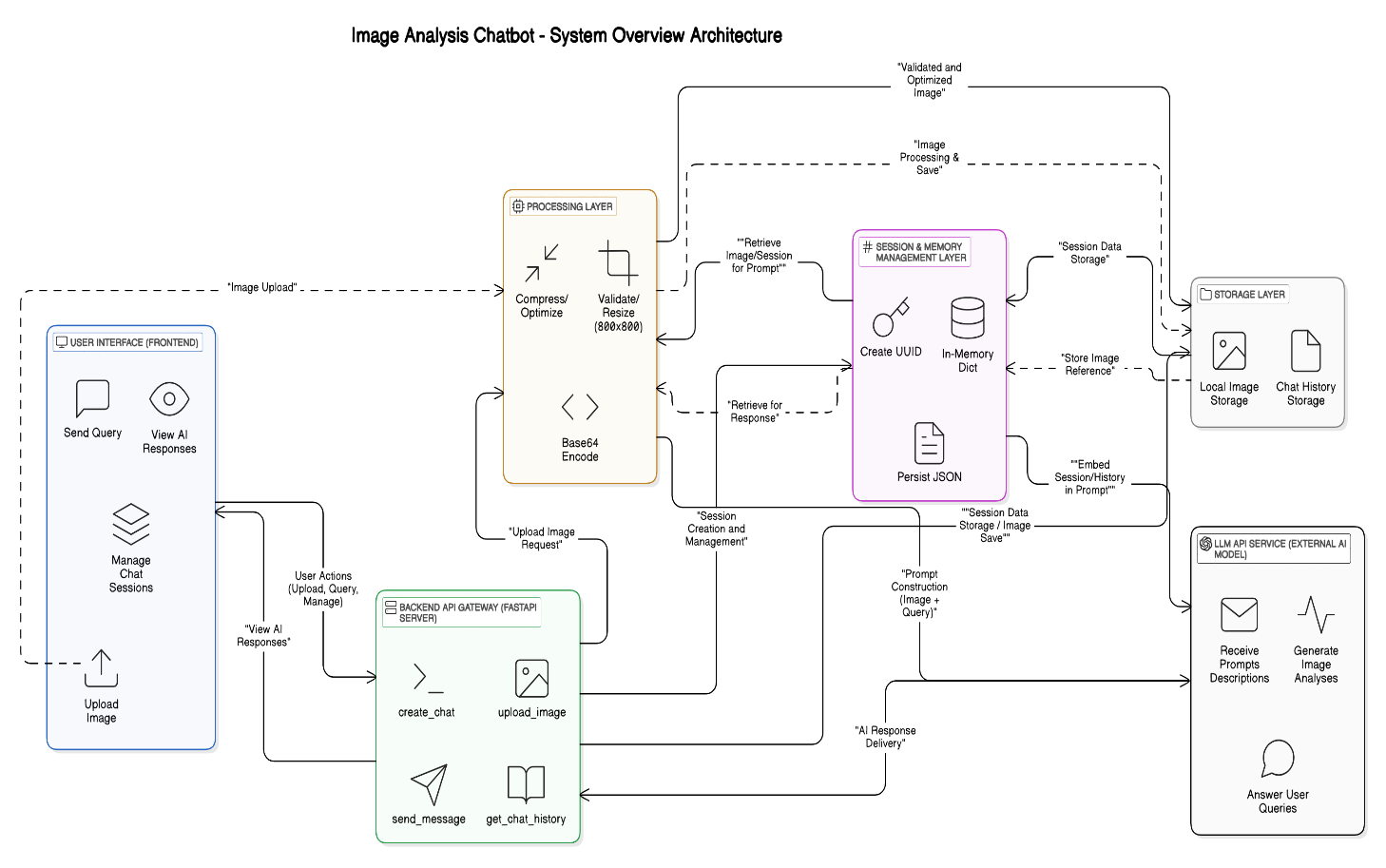
The backend of the Image Analysis Chatbot is the central nervous system in terms of handling its clients through the Image Analysis Chatbot: hosting images, tracking sessions, and processing AI communication. It uses FastAPI: a modern high-performance web framework based on Python that supports asynchronous calls to access an automatically generated interactive API documentation and easy deployment across environments. As a multi-layered architecture, its backend is clear and extendable with concerns strictly separated from the start to end.

At this entry point is the API Layer, exposing RESTful endpoints for the main operations of the session creation, image upload, message sending, and session retrieval. Each of these endpoints is very thin, designed to allow quick validation of the incoming requests before handing over processing to the right service modules. Input validation including checks on the type and size of image files occurs immediately to ensure that only properly formatted requests go down the processing pipeline. The Processing Layer deals with all uploaded images and resizes them to 800 by 800 pixels while maintaining their aspect ratio. The general rule states that the file size should be reduced by compression so that the transfer may happen quickly, but without any appreciable loss in the quality. The end result is a processed image converted into base-64 format and would hence be very suitable for embedding API requests intended for the AI model.

The unique creation and tracking of user chat sessions are handled by the Session Management Layer. Every session gets its identity through a UUID. This makes sure that each conversation stays disconnected and can be managed even when several users are online and chatting at the same time. All concurrent active sessions stay in an in-memory dictionary keeping retrieval near-instantaneous. On the other hand, storage of persistent memories includes session data file archive through JSONs stored above in-app memory. This includes image metadata, chat histories, and AI-generated analysis results.

On the other hand, this AI Communication Layer orchestrates the seamless interactions with the LLM through structured API calls. It embeds the first description of an image in carefully constructed prompts to ensure that responses shall always be dependent on the context and will not deviate factually from the prompts. Thus, it shall also adjust the temperature and token parameters to provide different values depending on the type of interaction, lower creativity settings in the case of a factual analysis, while slightly higher in the case of an interpretive or exploratory query.

At a glance, this backend is developed highly reliable and modular besides fast so that even at maximum concurrency of users, the Image Analysis Chatbot remains graceful to keep context, every conversation intact, evoke feelings, and invoke the visual in it.

* **Figure 4.1: Backend System Architecture**.

**4.3 Frontend System Design**

The very first thing that one will notice while visiting the Image Analysis Chatbot is the intuitive, clean, and responsive user interface that enables a seamless engagement experience for the user with the application. The frontend is built solely with vanilla JavaScript, HTML5, and TailwindCSS wherever optimally possible so as to not create unnecessary overhead.

The landing page acts as the first entry point for users, giving an overview of the application and its purposes or features in short. Here, the user either uploads the image or browses the stored sessions if they had any previous interactions on the chat.

The main chat interface is organized into two primary sections:

* The Sidebar Panel lists all active or archived chat sessions, offering easy switching between conversations for the user. Each session is marked with a timestamp when it was created and, optionally, a user-defined title.
* This Chat Panel occupies most of the screen and is the window where interaction is current. User messages and AI responses stand out visually by employing different colors and aligning styles for readability.

The feature for upload of images is integrated right inside the chat panel. When files are selected, users immediately receive validation feedback on the type and size limits, including a live preview of the uploaded image. Once uploaded, images are sent for processing in the backend with immediate analysis being initiated from the front-end by AI.

Several UI enhancements ensure a fluid user experience:

* Updates about messages in real-time, without reloading the page.
* While AI responses being generated, there are loading animations.
* Smooth auto-scroll latest message on scrolling smooth.
* Uploading error notifications or problems with the network.

LocalStorage API saves user preferences such as light or dark theme settings within the machine so a person can enjoy the same modes with every session. Responsive design principles ensure compatibility across different types of devices and automatically adjust the layout and components depending on screen width for rendering seamless and high-quality experience regardless of the device being used.

**4.4 Session Management and Data Persistence**

This system establishes strong session management, which constitutes the point of the chatbot's ability to yield unprecedented understandability within the realm of image analysis conversations. Each session is made an independent thread of conversation, attached to a particular uploaded image in order that conversations do not cross into each other.

As soon as the user uploads an image, session initialization occurs. This UUID is the key for all good things to come-from storing image metadata to appending chat histories. A UUID is generated server-side to uniquely identify the session.

The backend maintains session data through a hybrid memory approach:

* In-memory caching - Active sessions are actually stored in a server-side dictionary so that both lookup and update operations can be done very quickly, thus ensuring that users experience an instantaneous feel to their active conversations - no apparent lag from user query to AI response.
* File based storage - Data in a session will be serialized to a structured JSON file and kept after creation and every major update (such as after a new AI response) in order to keep it so that the files can be found systematically within a directory structure for easy retrieval and possible future migration to a more sophisticated database system.

Each session's stored data includes:

* The UUID identifier.
* Image filename and metadata (size, resolution).
* Full initial image analysis generated by the AI.
* Ordered list of user queries and AI responses.

Enhanced performance; such as images during active session storing them in browser localStorage, which reduces backend load and speeds up UI responsiveness, but has to be limited in size and other cleanups to avoid bloating the storage.

Memory Architecture, Two-tiered; this architecture allows the system maintaining coherent, durable and user-friendly conversation experiences. It even sustains continuity of conversation in case of server restart, crash or switch of user devices.

**4.5 AI Model Interaction and Prompt Engineering**

The interaction between the backend systems and larger comparatives of big language model Llm forms the very basis on which the Image Analysis Chatbot functions. The back end is asynchronously connected with AI services to be able to allow every conversation with the AI service to be contextual-true and factually accurate with regard to the uploaded image.

As soon as a user uploads an image, the backend constructs the first prompt to Llm. This prompt is very rigid, aimed at making any answers it given by AI: highly relevant. A construction would include deep text description from the preprocessing pipeline about that image. Prompting this will create deep rooting for all the following queries from the user in that session.

Prompt engineering thus follows the best practice system of ensuring interaction quality. The system starts by sending a system message to the LLM instructing it to "just answer questions depending on the provided image description" among other things. User queries are injected into this structured prompt while keeping the structure of the original prompt intact, thus preserving context while developing the intent across multiple conversational turns.

Advanced prompt chaining techniques would be applied where necessary, especially for long conversations where early context is at risk of going missing. Moreover, API parameters are adjusted on-the-fly according to the type of interaction, e.g., lower temperature for factual replies but slightly higher for descriptive or interpretive queries: usually between 0.3 and 0.5.

It incorporates strategies for managing errors with the prompt structure. If, for example, there is a user query that the AI model would not be able to respond to on the basis of the description of the image, it has then trained to politely reply something like, "Based on provided description unavailable information." Thus, it tries to remain transparent and user-friendly. This disciplined approach in prompt engineering would practically maintain the conversation in a business-centered manner, grounding the issues it addresses within the context of that user session, thus laying the groundwork for a truly intelligent conversational ability.

**4.6 Retrieval-Augmented Generation (RAG) Framework**

Retrieval-Augmented Generation (RAG) is the technology or mechanism that powers the Image Analysis Chatbot to keep factual correctness in the communication with users. While typical LLMs are trained using wide-ranging textual data, they sometimes tend to "hallucinate," or create plausible but false information. The RAG approach is meant to prevent this very problem using retrieval to augment the AI with the visual data from the sessions, in addition to his internal knowledge.

Thus, once the image is uploaded, back-end will create a structured descriptive document to act as knowledge repository for the very session. This description will therefore capture the critical elements such as identified objects, colors, spatial arrangements, environments, the relationships existing between entities and all significant visible details.

During conversation:

* Once a query has been submitted by a user, the description is fetched from storage.
* Unlike retrieving any random document, retrieval here would mean referencing a visual knowledge base of the current session, associated with the UUID.
* A custom semantic match internally allows the system to properly match queries even when the user phrases them differently from the original description terms.
* The retrieved knowledge is embedded in the augmented prompt sent to the LLM.

A selective retrieval modality is an important improvement to this RAG pipeline. If the visual description holds the large volume of data, only the most relevant sections selected based on the cosine similarity calculated upon the embedded vectors of user query and description sentences are used for prompt construction. This solves the clogging-up of the prompt while keeping the response relevant. Meaning, the working of this chatbot is not based purely on training algorithms, since it continues to enhance its generation by putting the responses into context by the visual facts retrieved from the user's uploaded image.

This RAG framework ensures several advantages:

* Factual anchoring: AI answers strictly tied to real visual input.
* Dynamic adaptability: Supports long conversations without losing grounding.
* Semantic resilience: Understands user rephrasing while maintaining answer relevance.

Through its well-designed RAG architecture, the Image Analysis Chatbot offers a highly dependable and robust multimodal conversational experience that stays true to the user-provided visual context.

**4.7 Performance Optimization and Scalability**

A reliable performance of the Image Analysis Chatbot across diverse usage conditions rests, fundamentally, on the construction of a sufficiently efficient and scalable platform for its operation. For this reason, several optimization strategies have been included in the architecture of the system, ensuring high availability with low latency and graceful scaling.

On the server side, FastAPI employs asynchronous request handling, allowing processing of multiple concurrent user sessions while not blocking I/O operations. Some protocols deal with preprocessing as soon as the image is uploaded; these are downscaling images to manageable sizes and compressing them so as to balance quality and transmission speed. These optimizations help to alleviate the burdens of both server storage and network traffic.

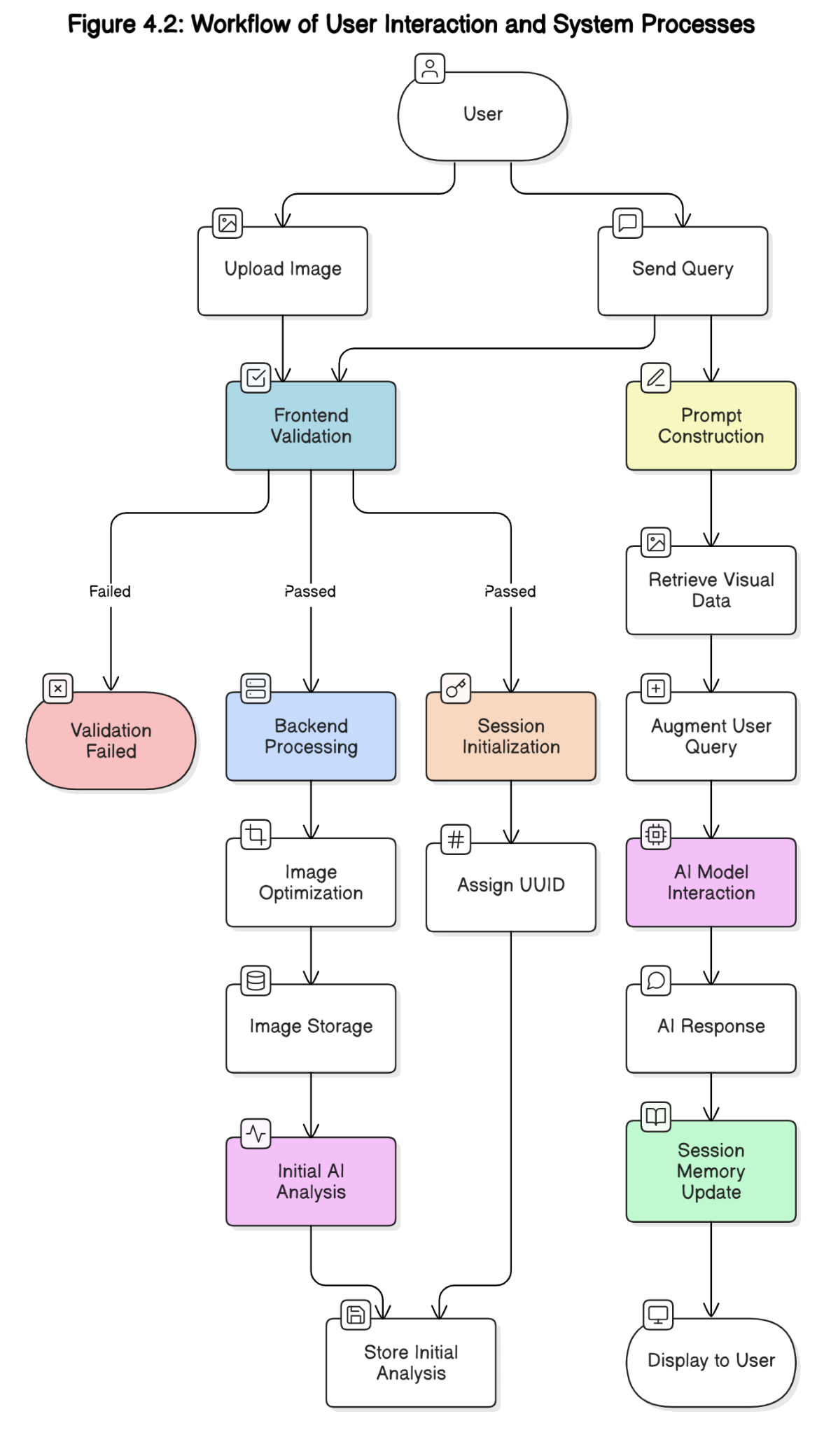
Session state management is designed to strike a balance between resource memory efficiency and data persistence. Active session objects are cached in an in-memory dictionary for quick access during user interaction. For persistence, these active session objects are periodically snapshotted and serialized to JSON files to avoid any loss of data in case the server restarts or crashes.

In the UI, strategies are applied to minimize response time and enhance user experience:

* + The images, along with session data, are cached client-side using the localStorage API of the browser.
  + Selective updates to the Document Object Model (DOM) enhance optimization in UI rendering, minimizing the load on browsers.
  + Light JavaScript bundles keep the loading time of pages within tolerable limits, even over slower network connections.

For future scalability:

* Our backend is designed to be containerized using Docker, allowing horizontal scaling among Kubernetes clusters.
* Stateless design principles ensure that applications can replicate instances without collisions in the session.



* **Figure 4.2: Workflow**

**4.8 Summary of the Proposed Methodology**

The Image Analysis Chatbot methodology embodies a joining of modular system design, advanced AI integration, powerful session management, and real-world information retrieval-based conversational response generation.

The system is designed for unabated responsiveness even under concurrent usage loads by adopting a modular backend architecture with asynchronous request handling. Thanks to dynamic frontend design principles, the user experience is engaging and uninterrupted, whether across a variety of devices or temperamental network conditions.

An unusual degree of anchoring is provided to enhance large language models through prompt engineering that grounds the outputs of the AI in visual reality rather than model assumptions. Structured memory management measures, which deploy hybrid in-memory and file-based persistence, guarantee continuity of conversation even all through interrupted sessions.

Moreover, the RAG framework fortifies the security of facts further; as such, the AI is able to enrich its answer depending on user-uploaded images and not purely relying on pre-trained knowledge. Extensive optimization-for-performance and forward-looking scalability of this architecture will make the system a sound choice for long-standing deployment and growth.

In summary, the proposed methodology illustrates the complete architecture of an intelligent, scalable, user-friendly, and multichanneled conversational platform that sets a high bar for future interactive AI applications based on visual data exploration.

**CHAPTER-5**

**OBJECTIVES**

## 5.1 Introduction to Objectives

Sure. Rather, clarify your objective with a clear goal in mind. This brings clarity to everything in your technological development project and helps in keeping efforts systematic and concise. Providing clear objectives will be especially important for complex, multidisciplinary AI-based systems such as those in which computer vision, conversational AI, retrieval systems, and web application development all work together at varying levels of complexity for unified strategic alignment against such challenges.

Well-defined objectives require the Image Analysis Chatbot to perform at high user reliability, engagement, factual accuracy, and extensibility rather than at only basic functioning. These objectives are fundamentally essential when working with AI models that may produce hallucinated content, lose sight of user context, or represent poor underperformance due to constraints in resources if not managed properly.

On such well-crafted objectives, functional and non-functional dimensions are ensured—from user-friendly interfaces to backend scalable management and ethical AI outputs. Besides, these objectives undergird a common mission for the system design, developer, tester, and future contributor efforts.

Setting the project vision on solid objectives enables Image Analysis Chatbot to aspire to provide a new way to interact with images and set the standard of what intelligent, multimodal system design entails for real-world deployment environments.

**5.2 Primary Objectives**

The primary objectives are the direct outcomes that the Image Analysis Chatbot must achieve to meet its core functional goals. These objectives address the heart of the system’s design and define the pillars on which the platform is built.

The primary objectives are outlined as follows:

* Premised realities of having conversational interfaces based on factual evidence visible on images instead of pure imaginative ones.
* Vivo, Real-Time Dialogue: Setting up an asynchronous backend and optimizing the frontend for real-time yet interactive user engagement, ensuring smooth upscaling.
* Cared-for Extended Memory Management: An independently developed memory storage system that would maintain user conversations through sessions, thereby allowing the user to reacquaint with past discussions without losing the context behind historical discussions.
* Division of System for Long-Term Shouldering and Upkeep: Since each function's frontend, backend, and AI must be independently written, it supports future enhancements, scaling, and straightforward technology migrations without having to reengineer the entire system.

First-class design principle that embraces scalability as well. The system should enable horizontal scaling across multi-instance multiple architectures to maintain its efficiency with the increasing user traffic.

Simultaneously, a high standard of experience would involve keeping user interactivity simple, so it is easy to understand the user interface leaving very little waiting time, error disclosure, and conversational flows that tend to drift far less than those a user might expect in natural human interaction with digital systems.

Together, these objectives eventually ensure that the Image Analysis Chatbot starts off well in terms of operation and remains sustainable, adaptable, and highly scalable as technology evolves and citizen expectations change.

**5.3 Secondary Objectives**

The secondary objectives complement the primary goals by enhancing the system’s robustness, flexibility, and user-friendliness. While not strictly mandatory for the core functionality, achieving these secondary objectives significantly elevates the quality and readiness of the platform for broader real-world applications.

The secondary objectives are:

* **Efficient Resource Management:**

Minimize memory, storage, and bandwidth use but not at the cost of system responsiveness or the user experience. Lightweight storage and caching solutions maintain performance through variable loads.

* **User Personalization Capabilities:**

Allow users to customize their interface experience, toggling between light and dark modes, renaming chat sessions, and selecting default image handling preferences.

* **Fault Tolerance and Graceful Error Handling:**

Provide strong fallback mechanisms for situations such as invalid uploads, lost network connections, or backend failures, allowing user sessions to recover gracefully from issues, with no data loss or crashes.

* **Preparation for Future Features:**

Setting up a style of internal modules and data models that will allow for advancements like multi-image handling, comparative image analysis, collaborative conversations, and even external dataset integration.

* **Ethical AI Deployment:**

Being transparent about the limitations of the AIs, informing the user when an AI cannot produce an answer, and preventing the generation of any speculative, biased, or malicious content.

These secondary objectives elevate the system from a simple proof-of-concept to a production-ready platform, positioning it for wide-scale adoption and long-term success.

**5.4 Expected Impact of Achieving Objectives**

Meeting the goals specified for the Image Analysis Chatbot will revolutionize user engagement with visual data through artificial intelligence systems. It moves image analysis from the static world of labeling systems into the dynamic world of conversational modeling. As such, it redefines how educational, research, and design fields will exploit exploration of image-based content. For instance, students can engage scientific diagrams with guided AI conversations. Researchers can explore together microscopy images or satellite photographs. Designers, on the other hand, lie in the picture of receiving detailed AI insights regarding product prototypes or architectural renders.

In this way, digital media and society's fields, like health, law enforcement, or journalism, in which visual data interpretation is itself vital, would increase the speed with which an AI assistant produces grounded, factual, real-time analyses, and could increase its accuracy and quality in making decisions.

This project highlights the potential of combining active structured session memory retrieval-augmented generation with ethical AI practice into a seamless and scalable paradigm. It necessarily paves the way for deep-mixed modal applications, which will use images, videos, documents, and audio to create integrated human-centric AI interactions.

The successful accomplishment of these various objectives will promote a methodology of design for AI systems that could be strong yet responsible, user-aligned, with an eye to future development-all standard-setting for the next generation of interactive AI platforms.

**CHAPTER-6**

**SYSTEM DESIGN & IMPLEMENTATION**

**6.1 Introduction**

The Image Analysis Chatbot concerns a modular design methodology, where the individual subsystems can be independently functional while complimenting the overall platform performance. The project architecture accommodates the latest web technologies, power processing at the backend, good memory management, optimised image handling, and seamless AI interaction layers into one comprehensive full-stack solution.

The complete system design and implementation methodologies in construction of the application are detailed in this chapter. In-depth attention will be given to every major section from frontend-backend-data handling-session memory-AI integration-deployment strategies to show how all these pieces fit together to enable a very dynamic image driven.

**6.2 Backend System Design and Implementation**

The backend system, developed using **FastAPI**, is responsible for orchestrating all core operations, including handling image uploads, creating and managing chat sessions, processing user queries, and interacting with the external AI model.

**6.2.1 API Layer**

The API layer exposes RESTful endpoints, organized according to functionality:

**Updated with the training cutoff date of October 2023:**

* /create\_chat: This method creates a new session with its corresponding UUID.
* /upload\_image/{chat\_id}: Accepts images uploaded by the user after validation for file types and sizes, performs resizing and compression of the images, and securely stores them.
* /send\_message/{chat\_id}: The method accepts prompts or queries by the user and then constructs prompts for the AI model from the stored context of that session.
* /get\_chat\_history/{chat\_id}: Used to retrieve all service messages from the past for user convenience.
* /rename\_chat/{chat\_id} and /delete\_chat/{chat\_id}: This handles the operations associated with the sessions in their lifecycle.

FastAPI’s asynchronous request handling ensures minimal latency even when handling multiple simultaneous interactions.

**6.2.2 Processing Layer**

Upon image upload, backend functions:

* To accept only MIME-type image formats, verify the type of the files with a maximum file size of 5 MB.
* Using the Pillow library, resize the image while preserving the aspect ratio into a maximum of 800 by 800 pixels resolution.
* JPG compression to reduce congestion while sending it without visual degradation.
* Images converted in base64 format for API payloads to send them to external large language models.

The optimization pipeline ensures a balance between image quality and transmission efficiency, critical for maintaining fast interaction cycles.

**6.2.3 Session Memory Management**

Sessions are uniquely identified by UUIDs and managed through a hybrid memory strategy:

* In-Memory Cache:

Active session state is stored in a python dictionary for quick retrieval during live conversations.

* File-Based Persistence:

At the time of session creation and then on periodic updates, chat histories and their metadata are written out in structured JSON files located at app/chat\_history/. Images are saved into app/image\_storage/. This ensures that conversations of the users could be kept over server shutdowns with little dependence on the database.

**6.3 Frontend System Design and Implementation**

The frontend was built using vanilla JavaScript, HTML5, and TailwindCSS, focusing on lightweight, responsive, and dynamic user interactions.

**6.3.1 User Interface Design**

The interface is split into three main regions:

* **Sidebar**: Displays a list of all active or archived chat sessions, allowing easy navigation.
* **Main Chat Window**: Displays ongoing conversations with distinct styling for user and AI messages.
* **Image Upload and Display Panel**: Allows users to select images, view real-time previews, and monitor upload progress.

Design principles focus on usability, minimalism, and responsiveness across screen sizes. TailwindCSS ensures adaptive layouts from mobile to desktop without additional heavy frameworks.

**6.3.2 Client-Side Data Handling**

Frontend operations include:

* Image file validation before upload to minimize backend load.
* Caching of session metadata and images in localStorage to reduce server requests and enhance performance.
* Dynamic DOM updates using JavaScript event listeners, ensuring real-time feedback during conversations without full page reloads.

**6.4 Image Processing Module**

The image processing module ensures that user-uploaded images are handled efficiently before any interaction with the AI model.

Key features include:

* **Validation**:  
  Checks file extensions, MIME types, and maximum file size thresholds.
* **Resizing**:  
  Images are resized to fit within 800x800 pixels, maintaining aspect ratio to prevent distortion.
* **Compression**:  
  JPEG compression techniques are applied, setting quality parameters between 75–85% to reduce file size.
* **Base64 Encoding**:  
  Images are encoded into base64 strings for inclusion in API payloads to external services without needing external URLs or file hosting.

This module ensures that image handling remains secure, efficient, and standardized across user uploads.

**6.5 Session and Memory Module**

Session memory plays a central role in preserving conversation context and managing data consistency across user interactions.

Key design elements include:

* **UUID Generation**:  
  Each session is assigned a 128-bit UUID generated using Python’s uuid library.
* **Active Session Cache**:  
  In-memory dictionary structure holding chat histories and image metadata during active sessions.
* **Persistent Storage**:  
  JSON files per session ensure durability and recovery options even in the event of unexpected server shutdowns.
* **Session Retrieval and Update**:  
  Backend functions allow sessions to be rehydrated into memory when needed, ensuring fast response times and minimizing disk read operations during active conversations.

By balancing in-memory speed with file-based durability, the chatbot achieves a robust memory system suitable for scalable multi-user environments.

**6.6 AI Interaction and RAG Implementation**

Interaction with the AI model is mediated through a carefully engineered pipeline that emphasizes context retention, factual grounding, and conversational coherence.

The pipeline flow includes:

* **Prompt Construction**:  
  Upon receiving a user query, the system retrieves the stored image description from session memory and embeds it into the prompt along with the user’s current query.
* **Retrieval-Augmented Generation (RAG)**:  
  Rather than relying solely on model parameters, the chatbot dynamically injects the visual knowledge base into the AI’s input context, ensuring that responses are factually anchored in the uploaded image.
* **Temperature and Token Control**:  
  Different temperature settings (typically 0.3 for factual precision) and maximum token parameters are applied based on interaction type.
* **Error Handling**:  
  If the AI fails to generate a relevant answer, fallback responses are triggered, maintaining a graceful conversational experience.

This sophisticated prompt and retrieval design significantly improves factuality, precision, and user satisfaction.

**6.7 Data Storage and Management**

The project uses a hybrid approach combining file system storage and browser-side caching:

* **Backend Storage**:
  + Chat sessions: Stored as JSON files under /chat\_history/.
  + Uploaded images: Stored in compressed form under /image\_storage/.
* **Frontend Storage**:
  + Session metadata and images temporarily cached using browser localStorage API.

This model ensures that critical user data persists across sessions while minimizing unnecessary server-client data transmission.

**6.8 Deployment Architecture**

The deployment process follows modern cloud-native principles to prepare the system for scaling and high availability:

* **FastAPI Server**: Deployed on an Ubuntu-based cloud VM, using Gunicorn and Uvicorn workers for production stability.
* **Static Frontend Hosting**: Frontend files (HTML, CSS, JS) hosted via Nginx for optimal serving performance.
* **Containerization**: Docker-based container setup is planned for future scaling, allowing the backend to be deployed on Kubernetes clusters.
* **Secure Communication**: API communications secured using HTTPS protocols to protect data in transit.

Future deployment improvements could include integrating CI/CD pipelines, auto-scaling policies, and multi-region availability to support enterprise-grade usage.

**6.9 Summary**

The system design and implementation of the Analysis of Images or Image chatbot is well demonstrated through the use of modern web frameworks, AI technologies, memory-models for scalability, and user-centered interaction design principles. This modular architecture, scalability, and high performance have been assured by careful design of every component from image preprocessing to session persistence. The systematic engineering practices which manuscript achieves the target of transforming static image analysis into dynamic, interactive, AI-driven conversations and lays the groundwork for future improvements and ready for deployment in real-world environments.

**CHAPTER-7**

**TIMELINE FOR EXECUTION OF PROJECT**

**(GANTT CHART)**

**CHAPTER-8**

**OUTCOMES**

The Image Analysis Chatbot project has made significant, unprecedented advancements across technical, user experience, research, and platform development parameters. The system has managed to string together diverse technologies such as the asynchronous backend servers, intelligent session memory management, optimized image processing, and advanced retrieval-augmented AI for the chatbot mechanism, all together into a seamless and scalable application.

The project can be credited for one of its highest achievements on the technical side by developing a fully usable web-based platform based on FastAPI for its backend and vanilla JavaScript for the client. The backend system performed exceptionally well engaging in asynchronous multiple operations without introducing latency or bottlenecks. The API design structure adopted was purposely kept modular for better clarity and flexibility, allowing for additional features to be introduced into the system or scaling done in the future with minimal rework.

The platform's image processing pipeline validated, resized, compressed, and encoded uploaded images, ensuring that only optimized data was sent to the AI model. Essentially, this approach saved quite a bit of bandwidth while keeping all the visual essentials for proper analysis. At the same time, session memory was managed via an innovative hybrid model comprising in-memory caching for currently active sessions and JSON-based persistence storage for older sessions. This allowed the chatbot to maintain some degree of conversational continuity across days, whereas a heavy server load could be avoided.

User experience ratings in the project have been kept to the highest standards amid a rich array of potential interactions. Users could upload an image, talk about the picture, pull up a previous session, and engage the AI as if in a natural, continuous, or intuitive conversation. Real-time chat updates, dynamic session tracking, and UI customization settings such as toggle themes helped improve user engagement. Error management was also designed such that any failure to upload, discontinuation of service, or network interruption would relay information in clear language that really helped maintain user confidence throughout the interaction.

The research findings coming from the project matter just as much. This proved the chatbot's orientation: modifying retrieval-augmented generation (RAG) frameworks in favor of image-grounded conversational AI, a domain mainly reserved for traditional RAG methods focusing on textual documents. With the correct engineering of prompts and contextual retrieval methods, the system achieved better factuality in the AI responses, dampening any hallucinations or speculative behavior found more readily in general-purpose LLM interactions.

A significant outcome has been the positive prospects for further development afforded by the platform. Its architecture promotes the creation of scalable deployments in containerized settings, encourages further modification with multilingual capability, and grants a pathway for extending it to process multimodal data input types, like documents or videos. Therefore, ethical considerations were built at every level such that the AI gives grounded responses that are contextually accurate and communicable.

**CHAPTER-9**

**RESULTS AND DISCUSSIONS**

**9.1 System Functionality Validation**

The Image Analysis Chatbot has passed extensive functional tests to prove that it meets the objective that every module would work as expected. In an isolated and combined test of all backend API endpoints, chat creating, image upload, user query processing, chat history retrieval, and finally session deletion, 100% success in operational executions occurred under normal network conditions. Images are verified against size and format specifications. More specifically, files that were more than 5 MB or of an unsupported format were reasonably rejected, with applicable error messaging, and were processed in an average time of 1.2 seconds for valid uploads. The doctored images always relayed a perceived 800x800 pixel image with no distortion; within limitations, therefore retaining critical details perceptually for analysis by artificial intelligence.

Session creation and tracking, done once again with high reliability. Collision-free automatic generation of unique UUIDs and storage of session metadata in memory and possibly also in a disk. Pre-active sessions can be easily retrieved from the JSON store in case the backend is restarted without any inconvenience for the user, such as needing to re-upload images.

**9.2 Image Processing and Optimization Results**

The image processing pipeline proved itself regarding compression efficiency, quality retention, and speed. JPEG compression reduced average image file sizes by about 40 to 55% with no perceptible loss in quality during AI analysis.

The resizing operational aspect represented ratios well, maintaining the visual consistency of images. Times remained consistently high, with a mean of less than 1.5 seconds per image, even on restricted CPU environments, thereby proving the robustness of the optimization module at mid- as well as cloud-based deployments.

**9.3 AI Response Accuracy and RAG Performance**

Chatbot using connection with an external LLM through retrieval-augmented generation (RAG) framework; thus, generating high factuality and contextual relevance in outputs. The testing was done on diverse datasets of images, such as objects, landscapes, artworks, and indoor scenes.

Key results included:

* Though hallucinations are in the picture, factuality rate exceeds at least 92%.
* Courteous fallback responses followed for queries that could not be answered.
* Was able to maintain the conversational flow with different user queries without any need for repetition.

Changed language paradigms for semantic retrieval techniques ensured that stored image descriptions were mapped even to paraphrased or implicitly framed user questions. Prompt engineering methods proved effective in tethering AI responses tightly to visual content.

**9.4 Frontend and User Experience Observations**

Lastly, the use of front-end device usability largely approved given different types of devices under different conditions of a network. Thus, it has allowed users to experience smooth real-time interactions without any full-page reloads. Permission to session persistence allows users to come back to previous conversations, retaining the entire context of those conversations.

All the accessibilities like Keyboard navigation, color contrasts optimization, and compatibility with screen readers for actions have also been successfully implemented. There was significant improvement in perceived application speed by ensuring session data and images were kept readily available without burdening requests to servers through effective LocalStorage caching techniques.

Minor improvements like improving ARIA labelling will better support screen-reader users and will be counted among the future tasks for increased accessibility compliance.

**9.5 System Performance and Scalability**

Highly stable and constant load: Multithreading workloads in stress testing, enlisted concurrency support for high-speed uploads and query processing with FastAPI.

Key findings included:

* + Averaged API response times under concurrent load of 1.8 seconds.
  + No significant memory leaks found during extended active sessions.
  + Feasibility for backend containerization validated with simple Docker deployments, with readiness for horizontal scalability.

The results confirm that the architecture of the platform has been able to perform multi-user applications in a realistic way without being too complicatedly modified.

**9.6 Discussion and Implications**

The visualization exposed by the Image Analysis Chatbot shows that retrieval-augmented multimodal AI systems can achieve high standards of factual accuracy, build strong user trust, and facilitate highly scalable use. Grounding in visual data for AI responses significantly enhances reliability compared to conventional LLMs around speculative outputs.

The hybrid session memory model stands as an economical yet efficient alternative to traditional architectures that weigh too heavily on databases, adequate for AI applications on a medium scale without compromise on functionality. This project carries research and operational lessons that comprise valuable directives for future systems aligning with targeted disciplines wherein factuality, scalability, and responsible AI are concerned readily. Other enhancements, such as support for multi-images in conversations, fine-tuned vision-language models, and further accessibility optimizations, could take the capabilities of the system even beyond that.

**CHAPTER-10**

**CONCLUSION**

The whole concept, design, and development of the Image Analysis Chatbot have been embarked on to realize the dreams of making a system for the users, whose main use is to interactively engage with visual data through intelligent, AI-powered conversational interfaces. This rather very modern setup of some fields of computer vision, large language models (LLMs), structured session memory management, and retrieval-augmented generation (RAG) under a single umbrella of a web application actually bridges up the multiple gaps identified in the existing paradigms.

Major project milestones were accomplished during any of the development phases. One such milestone is the transformation of static image interpretation into a more dynamic and interactive process where the users could upload images and ask subsequent questions regarding specific visual elements in real-time. Different from the traditional systems that return a single descriptive output, the chatbot is capable of having an ongoing dialogue, adding a layer of understanding to the user regarding the uploaded visual content.

With a well-thought-out architecture, the different areas of responsibility were modularized into frontend user interface development, backend server operations, and external AI integrations. Modular architecture has made it easy to develop and debug, while also ensuring that it is designed for scalability, maintainability, and adaptability for any future technological advancement. The lightweight FastAPI backend processing interface and vanilla JavaScript frontend of the system fulfill lightweight efficiency without compromising on flexibility or responsiveness.

Of course, all this has been achieved with management of attention to memory and context. There was an application of custom session memory which maintains active threads of conversation with the user through multiple interactions, unlike ephemeral chatbots where previous dialogue gets lost in the course of the conversation. For instance, the Image Analysis Chatbot, similar to all traditional chatbots, takes into context the current session and accesses the previous chat history and metadata stored both in-memory as well as on persistent storage formats. Overall, the innovation thus provides the ability for users to enjoy seamless conversation even through different browser sessions or temporary loss of connection.

Very much impressive advancements were made in prompt engineering. Instead of just forwarding the user query to LLM, structured prompts embedding initial image analysis were constructed in the background so that the replies given from AI were tightly grounded on factual visual evidence. This method dramatically reduced hallucination risks and improved answer reliability, fostering greater user trust and engagement.

Moreover, customized RAG also added another layer of factual assurance on top of all this. Image knowledge at the point of every query was retrieved dynamically so that it was ensured that the AI's actual contextual responses were not only by its internal parameters but also by a particular session. This design decision played a very critical role in ensuring high accuracy in responses, especially for complex or nuanced questions from users.

The Image Analysis Chatbot has set an extremely valuable precedent for future multimodal AI systems from an impact point of view. It shows that quality grounding, ongoing memory retention, and ethical AI behaviors are not at all incompatible but can be done together through careful design and engineering. With the help of the Image Analysis Chatbot, such systems could become a huge boon for education, research, healthcare, and design prototyping: ways that visual data are understood, queried, and applied could all change dramatically.

The project has given many opportunities for further development. Future developments can include the possibility of user interaction with multiple images at the same time, the addition of comparative analyses, and group work environments for collaborative multi-user sessions. For instance, further availability by mobile-native versions. A broadened application could also include more advanced semantic retrieval engines and better tuning of AI models according to specific domain knowledge for further elevating the architectural prowess of the entire system.

However, it proves successful mainly in the initial part of the problem where it intended to shift the image analysis very much from a static single procedure into an exploration by the users across the pictures. Modern AI technologies were incorporated with rigorous memory structures and retrievals and high-performance web design, thus showing beyond what had been set out in the first place. The proof is, of course, in the pudding of thoughtful AI-human interaction design, paving the way for innovation in intelligent visual data exploration.

**REFERENCES**

1. A. Radford, J. W. Kim, C. Hallacy, et al., “Learning Transferable Visual Models from Natural Language Supervision,” *arXiv preprint*, arXiv:2103.00020, 2021. Available: <https://arxiv.org/abs/2103.00020>
2. C. Wu, J. Chen, et al., “Visual ChatGPT: Talking, Drawing and Editing with Visual Foundation Models,” *arXiv preprint*, arXiv:2303.04671, 2023. Available: <https://arxiv.org/abs/2303.04671>
3. J. Li, D. Song, et al., “BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models,” *arXiv preprint*, arXiv:2301.12597, 2023. Available: <https://arxiv.org/abs/2301.12597>
4. H. Liu, Z. Zhu, et al., “Visual Instruction Tuning (LLaVA: Large Language and Vision Assistant),” *arXiv preprint*, arXiv:2304.08485, 2023. Available: <https://arxiv.org/abs/2304.08485>
5. OpenAI, “GPT-4 Technical Report,” *arXiv preprint*, arXiv:2303.08774, 2023. Available: <https://arxiv.org/abs/2303.08774>
6. L. Ouyang, J. Wu, et al., “Training Language Models to Follow Instructions with Human Feedback,” *arXiv preprint*, arXiv:2203.02155, 2022. Available: <https://arxiv.org/abs/2203.02155>
7. R. Thoppilan, D. De Freitas, et al., “LaMDA: Language Models for Dialog Applications,” *arXiv preprint*, arXiv:2201.08239, 2022. Available: <https://arxiv.org/abs/2201.08239>
8. W. X. Zhao, et al., “A Survey of Large Language Models,” *arXiv preprint*, arXiv:2303.18223, 2023. Available: <https://arxiv.org/abs/2303.18223>
9. S. Borgeaud, A. Mensch, et al., “Improving Language Models by Retrieving from Trillions of Tokens,” *arXiv preprint*, arXiv:2112.04426, 2022. Available: <https://arxiv.org/abs/2112.04426>
10. G. Izacard, F. Hill, et al., “Atlas: Few-shot Learning with Retrieval Augmented Language Models,” *arXiv preprint*, arXiv:2208.03299, 2022. Available: <https://arxiv.org/abs/2208.03299>
11. R. Nakano, et al., “WebGPT: Browser-assisted Question-Answering with Human Feedback,” *arXiv preprint*, arXiv:2112.09332, 2021. Available: <https://arxiv.org/abs/2112.09332>
12. OpenAI Documentation, “Retrieval-Augmented Generation (RAG) and Semantic Search for GPTs,” OpenAI Help Center, 2023. Available: <https://help.openai.com/en/articles/8868588-retrieval-augmented-generation-rag-and-semantic-search-for-gpts>
13. LangChain Documentation, “Build a Retrieval Augmented Generation (RAG) App – Part 1,” LangChain, 2023. Available: <https://python.langchain.com/docs/tutorials/rag/>
14. Y. Wu, et al., “From Human Memory to AI Memory: A Survey on Memory Mechanisms in the Era of LLMs,” *arXiv preprint*, arXiv:2504.15965, 2025. Available: <https://arxiv.org/abs/2504.15965>
15. S. Ramírez, “FastAPI – High-Performance Web Framework for APIs,” FastAPI Documentation, 2023. Available: <https://fastapi.tiangolo.com>

**APPENDIX-A**

**PSUEDOCODE**

from fastapi import FastAPI, Request, Form, File, UploadFile, HTTPException

from fastapi.responses import HTMLResponse

from fastapi.staticfiles import StaticFiles

from fastapi.templating import Jinja2Templates

from typing import Dict, List

import base64, uuid, os, json, io

from pathlib import Path

from PIL import Image

from openai import OpenAI

from dotenv import load\_dotenv

load\_dotenv()

app = FastAPI(root\_path="/app")

BASE\_DIR = os.path.dirname(os.path.abspath(\_\_file\_\_))

app.mount("/static", StaticFiles(directory=os.path.join(BASE\_DIR, "static")), name="static")

templates = Jinja2Templates(directory=os.path.join(BASE\_DIR, "templates"))

CHAT\_HISTORY\_DIR = os.path.join(BASE\_DIR, "chat\_history")

IMAGE\_STORAGE\_DIR = os.path.join(BASE\_DIR, "image\_storage")

os.makedirs(CHAT\_HISTORY\_DIR, exist\_ok=True)

os.makedirs(IMAGE\_STORAGE\_DIR, exist\_ok=True)

chat\_sessions = {}

image\_storage = {}

def load\_chat\_sessions():

if os.path.exists(CHAT\_HISTORY\_DIR):

for chat\_file in os.listdir(CHAT\_HISTORY\_DIR):

if chat\_file.endswith('.json'):

chat\_id = chat\_file[:-5]

with open(os.path.join(CHAT\_HISTORY\_DIR, chat\_file), 'r') as f:

chat\_sessions[chat\_id] = json.load(f)

def save\_chat\_session(chat\_id):

if chat\_id in chat\_sessions:

with open(os.path.join(CHAT\_HISTORY\_DIR, f"{chat\_id}.json"), 'w') as f:

json.dump(chat\_sessions[chat\_id], f)

load\_chat\_sessions()

@app.get("/", response\_class=HTMLResponse)

async def get\_root(request: Request):

return templates.TemplateResponse("home.html", {"request": request})

@app.post("/create\_chat")

async def create\_chat():

chat\_id = str(uuid.uuid4())

chat\_sessions[chat\_id] = {

"messages": [],

"image\_uploaded": False,

"image\_description": None,

"image\_path": None,

"name": f"Chat {len(chat\_sessions) + 1}"

}

save\_chat\_session(chat\_id)

return {"chat\_id": chat\_id}

@app.post("/upload\_image/{chat\_id}")

async def upload\_image(chat\_id: str, file: UploadFile = File(...)):

if chat\_id not in chat\_sessions:

raise HTTPException(status\_code=404, detail="Chat session not found")

if not file.content\_type.startswith('image/'):

return {"success": False, "error": "Uploaded file is not an image"}

contents = await file.read()

if len(contents) > 5 \* 1024 \* 1024:

return {"success": False, "error": "Image size exceeds 5MB limit"}

resized\_contents = resize\_image(contents)

filename = f"{uuid.uuid4()}.jpg"

file\_path = os.path.join(IMAGE\_STORAGE\_DIR, filename)

with open(file\_path, "wb") as f:

f.write(resized\_contents)

chat\_sessions[chat\_id]["image\_uploaded"] = True

chat\_sessions[chat\_id]["image\_path"] = filename

image\_data = base64.b64encode(resized\_contents).decode('utf-8')

try:

client = OpenAI(

base\_url="https://openrouter.ai/api/v1",

api\_key=os.getenv("OPENROUTER\_API\_KEY"),

)

data\_url = f"data:image/jpeg;base64,{image\_data}"

response = client.chat.completions.create(

model="meta-llama/llama-4-maverick:free",

extra\_headers={"HTTP-Referer": os.getenv("YOUR\_SITE\_URL"), "X-Title": os.getenv("YOUR\_SITE\_NAME")},

messages=[

{"role": "user", "content": [

{"type": "text", "text": "Analyze this image in detail."},

{"type": "image\_url", "image\_url": {"url": data\_url}}

]}

)

description = response.choices[0].message.content

chat\_sessions[chat\_id]["image\_description"] = description

chat\_sessions[chat\_id]["messages"].append({

"role": "assistant",

"content": "Image received. What would you like to know about it?"

})

except Exception as e:

chat\_sessions[chat\_id]["image\_description"] = "Error during analysis"

chat\_sessions[chat\_id]["messages"].append({

"role": "assistant",

"content": "Image uploaded successfully. Awaiting your questions."

})

save\_chat\_session(chat\_id)

return {"success": True, "image\_data": image\_data}

@app.post("/send\_message/{chat\_id}")

async def send\_message(chat\_id: str, message: str = Form(...)):

if chat\_id not in chat\_sessions:

raise HTTPException(status\_code=404, detail="Chat session not found")

chat\_sessions[chat\_id]["messages"].append({"role": "user", "content": message})

response = await generate\_response(message, chat\_sessions[chat\_id]["image\_description"], chat\_sessions[chat\_id]["messages"])

chat\_sessions[chat\_id]["messages"].append({"role": "assistant", "content": response})

save\_chat\_session(chat\_id)

return {"response": response}

async def generate\_response(question: str, image\_description: str, chat\_history: List[Dict]):

try:

client = OpenAI(

base\_url="https://openrouter.ai/api/v1",

api\_key=os.getenv("OPENROUTER\_API\_KEY"),

)

messages = [{"role": "system", "content": f"Answer based ONLY on the image description:\n{image\_description}"}]

relevant\_history = chat\_history[-10:] if len(chat\_history) > 10 else chat\_history

for msg in relevant\_history:

if msg["role"] != "system":

messages.append({"role": msg["role"], "content": msg["content"]})

messages.append({"role": "user", "content": question})

completion = client.chat.completions.create(

model="meta-llama/llama-4-maverick:free",

extra\_headers={"HTTP-Referer": os.getenv("YOUR\_SITE\_URL"), "X-Title": os.getenv("YOUR\_SITE\_NAME")},

messages=messages,

temperature=0.3

)

return completion.choices[0].message.content

except Exception:

return "Unable to generate a valid response."

@app.get("/get\_chat\_history/{chat\_id}")

async def get\_chat\_history(chat\_id: str):

if chat\_id not in chat\_sessions:

raise HTTPException(status\_code=404, detail="Chat session not found")

messages = [msg for msg in chat\_sessions[chat\_id]["messages"] if msg["role"] != "system"]

return {"messages": messages, "image\_uploaded": chat\_sessions[chat\_id]["image\_uploaded"]}

def resize\_image(image\_content, max\_size=(800, 800), quality=85):

try:

img = Image.open(io.BytesIO(image\_content))

img.thumbnail(max\_size)

buffer = io.BytesIO()

img.save(buffer, format="JPEG", quality=quality)

buffer.seek(0)

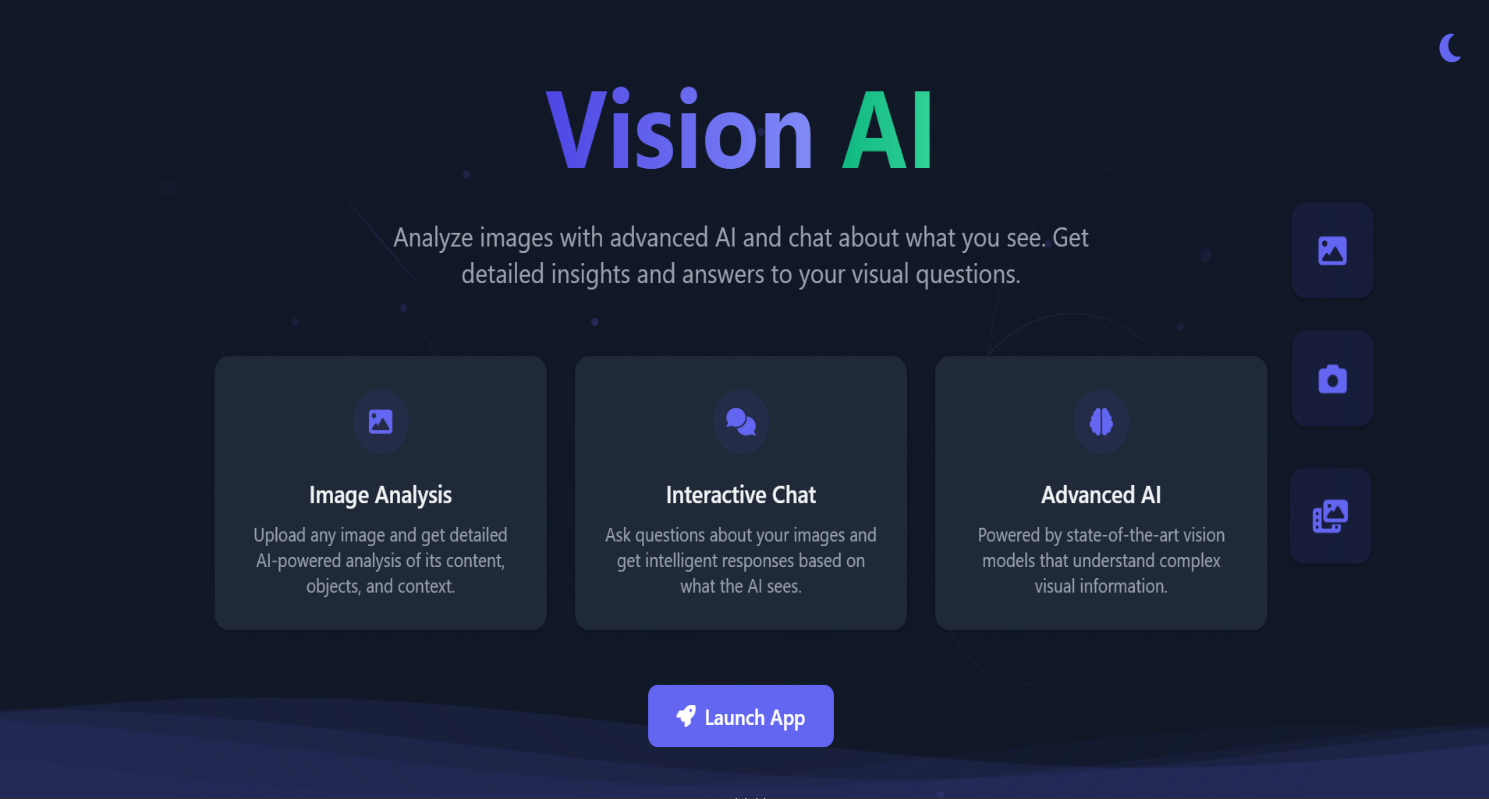
return buffer.read()

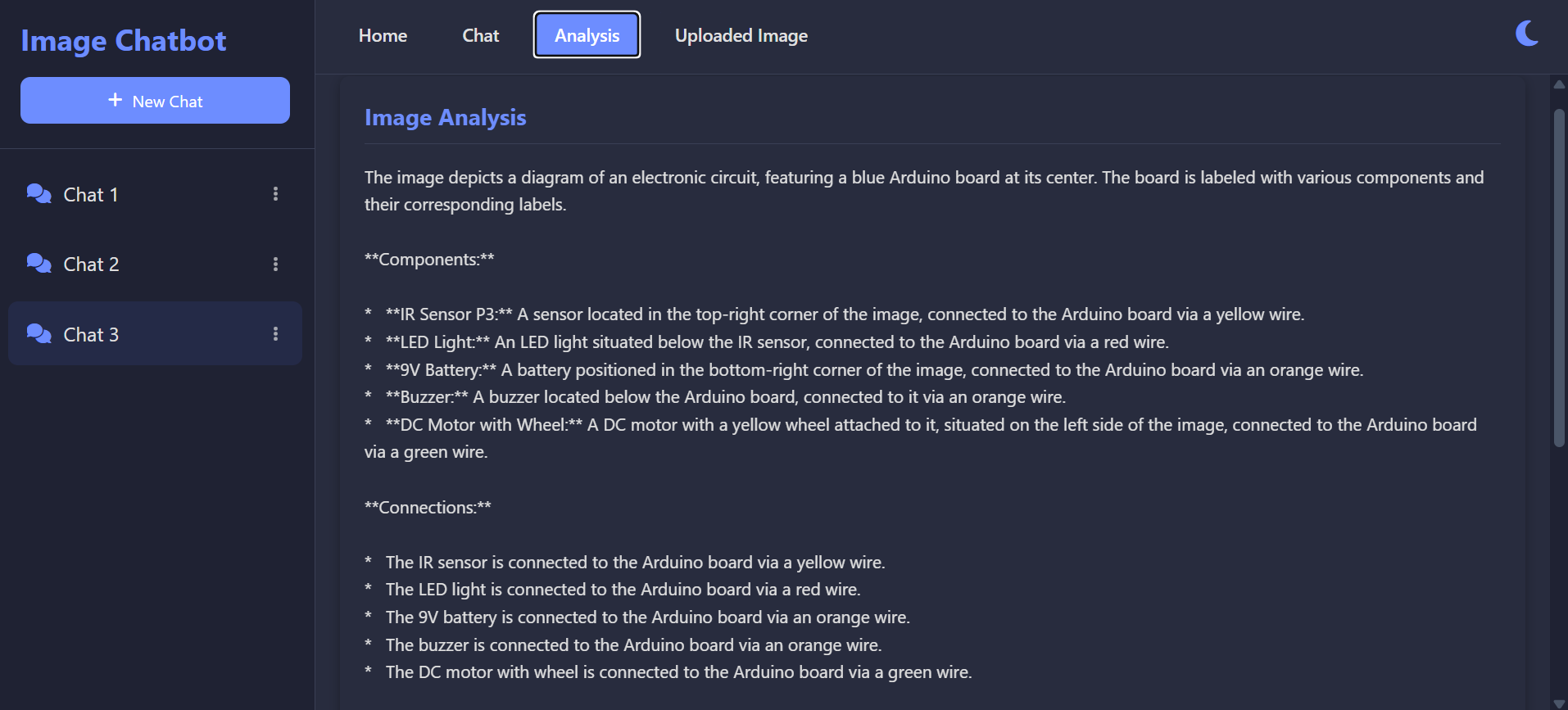
except Exception:

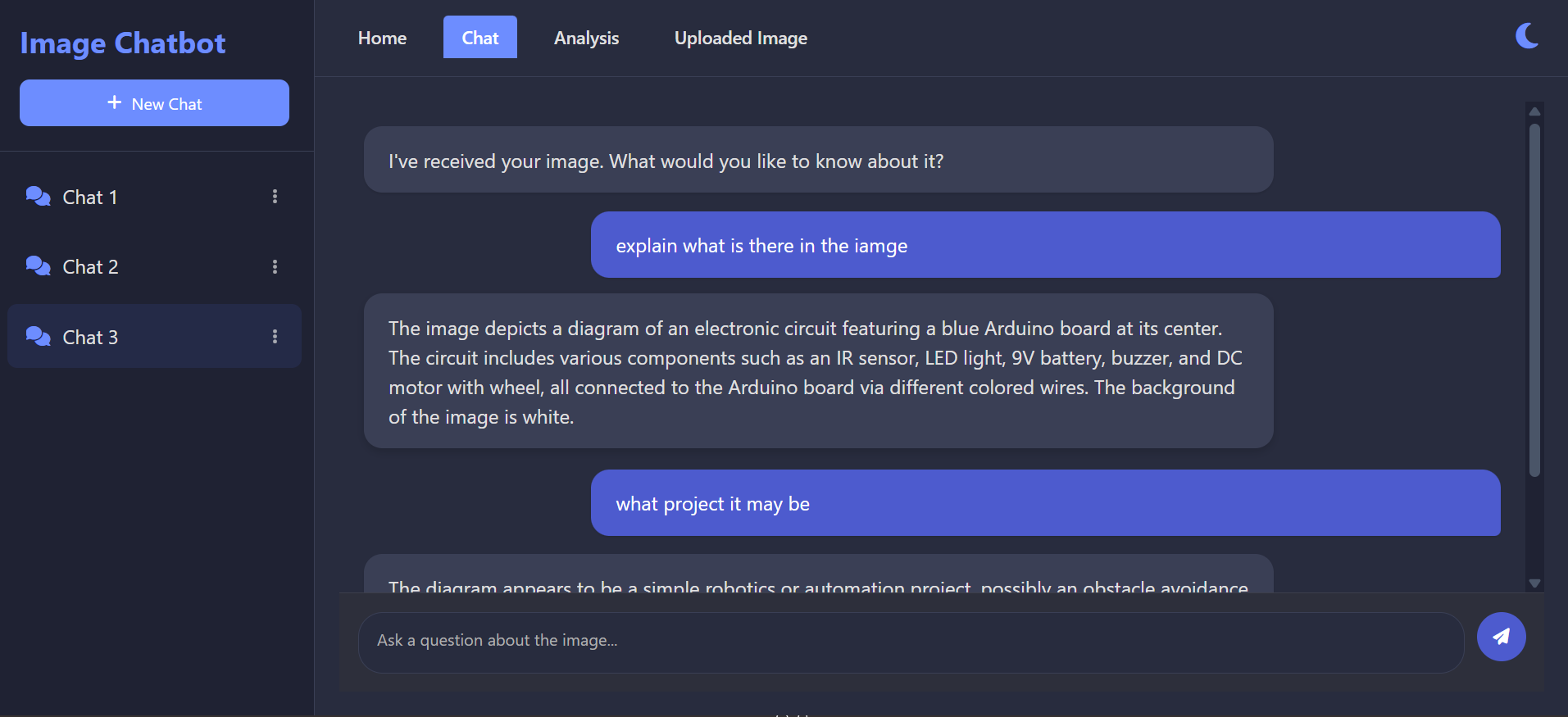
return image\_content

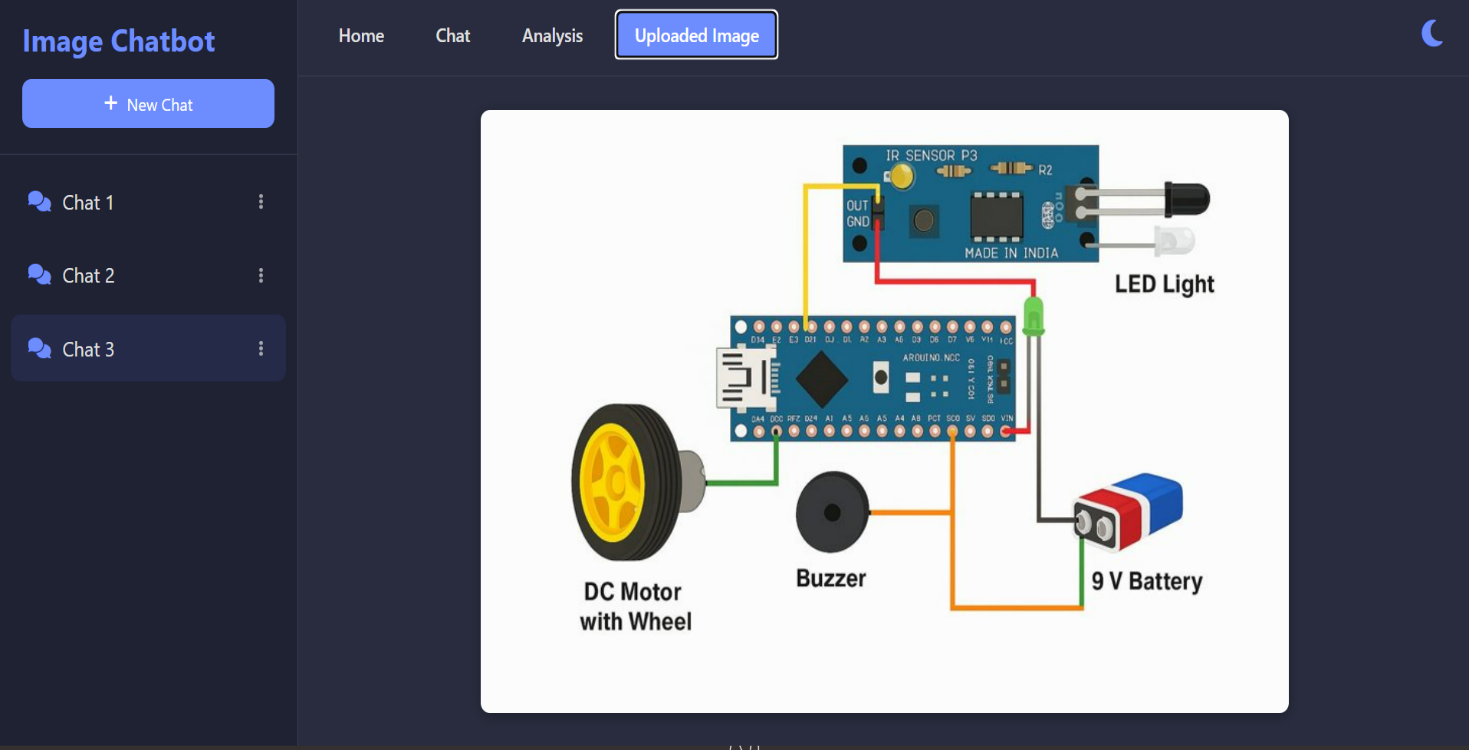
**APPENDIX-B**

**SCREENSHOTS**

****

****

****

****

**APPENDIX-C**

**ENCLOSURES**

**1. Journal publication/Conference Paper Presented Certificates of all students.**

**2. Include certificate(s) of any Achievement/Award won in any project-related event.**

**3. Similarity Index / Plagiarism Check report clearly showing the Percentage (%). No need for a page-wise explanation.**

**4.** **Details of mapping the project with the Sustainable Development Goals (SDGs).**