Internship Report

on

AI-powered Interactive Learning Assistant for Classrooms

By

Chavva Hasya Reddy

Chakri Kandregula

G Hima Sree

(Duration: 19th May 2025 – 12th July 2025)

Abstract

This report explores the development of a **multimodal AI-powered assistant** tailored for classroom environments, aimed at enhancing student engagement through dynamic, real-time interaction. The solution addresses the challenge of providing personalized and context-aware learning support by integrating **natural language processing**, **speech recognition**, **visual understanding**, and **emotion analysis** into a unified system.

It achieves this through a multi-modal pipeline:

1. **Multimodal Query Handling:** The assistant accepts input via text, voice, or image, allowing students to interact naturally and flexibly during the learning process.
2. **Real-Time Response Generation:** Using local large language models (via Ollama) and advanced image captioning/OCR modules, the assistant delivers immediate, educationally relevant responses and explanations.
3. **Visual Aid Generation:** When complex topics arise, the system automatically creates flowcharts or data visualizations to aid understanding using Graphviz and Matplotlib.
4. **Emotion-Aware Engagement Monitoring:** Through facial expression analysis via OpenCV and deep learning, the assistant detects emotions such as confusion or frustration and suggests simpler explanations proactively.
5. **User-Friendly Interface:** A clean, responsive web interface allows seamless interaction, integrating live emotion feedback and toggles for voice/speech-based input and output.

This project demonstrates the practical application of **multimodal and emotionally-intelligent AI** in education. It highlights the potential of such systems to support inclusive, adaptive, and student-centered learning experiences across classrooms.

Contents

Introduction

1. 1 Problem Statement

In today's classrooms, students often face challenges in receiving immediate, personalized academic support. Traditional teaching methods and static digital tools lack the ability to dynamically adapt to each learner's pace, input style, or emotional state. As a result, confusion, frustration, or disengagement can go unnoticed, ultimately affecting comprehension and learning outcomes.

To address this, the need arises for an **AI-based multimodal educational assistant** capable of providing **real-time, intelligent responses** across multiple input formats — such as voice, text, and images — while also being sensitive to student **emotional cues**. Such a solution should actively:

1. **Process multimodal queries**: Accept and understand spoken questions, typed input, or uploaded images.
2. **Generate educational explanations**: Respond contextually using natural language models, enhanced with charts or flowcharts when required.
3. **Detect and respond to student emotions**: Identify signs of confusion, frustration, or disengagement using facial expression analysis and adapt its explanations accordingly.
4. **Encourage student interaction**: Promote engagement by creating a friendly, accessible, and responsive interface suited for self-paced and assisted learning.

The goal is to build a **scalable, low-latency, AI-powered assistant** that functions seamlessly in classroom environments, bridging the gap between human teaching and intelligent digital support. This solution must be robust enough to manage diverse accents, noisy conditions, varied question types, and multiple emotional expressions — all while maintaining a smooth user experience.

1. 2 Project Objectives

This project aims to develop an **AI-powered multimodal assistant** designed specifically for classroom learning environments. The assistant leverages voice, text, and image processing along with emotion detection to deliver real-time, personalized, and adaptive academic support to students.

The core objectives of this project are:

- **To design a system capable of processing and understanding multimodal inputs**, including voice queries, typed questions, and uploaded images (e.g., textbook pages, handwritten notes, or diagrams).
- **To develop intelligent response mechanisms** using large language models that provide contextually accurate, subject-specific explanations in simple, student-friendly language.
- **To generate visual aids dynamically** (such as flowcharts and charts) that help illustrate complex concepts and improve comprehension.
- **To implement real-time facial emotion detection**, enabling the assistant to monitor student engagement and suggest simplified responses in case of confusion, frustration, or disengagement.
- **To build a scalable, low-latency system** with a responsive, intuitive web interface that ensures seamless interaction between the student and the assistant across various devices.
- **To demonstrate the integration of NLP, speech recognition, computer vision, and emotional intelligence** into a single assistant system that enhances the learning experience for diverse student needs.

By achieving these objectives, the project showcases the potential of AI to transform classrooms into **adaptive, emotionally-aware, and inclusive learning environments**.

1. 3 Motivation

The rapid evolution of technology and the growing demand for personalized learning have revealed significant gaps in current educational tools. Traditional classroom settings often lack the flexibility and responsiveness needed to cater to diverse student learning styles and emotional states. Students may hesitate to ask questions, lose focus, or fall behind when explanations are too complex or when emotional disengagement goes unnoticed.

This project is driven by the need for an **interactive, multimodal AI assistant** that empowers students by:

- **Providing personalized academic support** in real-time, across multiple input modes — voice, text, and image — encouraging active learning.
- **Enhancing engagement through emotional awareness**, detecting non-verbal cues of confusion or frustration and adapting explanations accordingly.
- **Fostering clarity and confidence** by generating visual aids and simplified responses that improve understanding.
- **Bridging the gap between static digital tools and human-like interaction**, using AI to simulate responsive, emotionally-intelligent tutoring.

The motivation behind ClarifAI is to build an assistant that doesn't just **answer questions**, but **understands students**, adapts to their needs, and supports them like a real-time mentor. This solution aims to **revolutionize classroom learning** by making it more inclusive, interactive, and emotionally intelligent through the power of AI.

2. Team

2.1 Team Members

Chavva Hasya Reddy AI Enthusiast B.Tech Computer Science Engineering | 2022 - 2026 Email: hasyareddy18@gmail.com	Chakri Kandregula AI Enthusiast B.Tech Computer Science Engineering Data Science | 2022 - 2026 Email: kchakri.techie@gmail.com	G Hima Sree AI Enthusiast B.Tech Computer Science Engineering Data Science| 2022 - 2026 Email: himasreg@gmail.com

2.2 Team Contribution

This project is the result of collaborative efforts from a three-member team, with each member contributing to key modules of the ClarifAI assistant system. The division of work ensured that the multimodal, real-time, and emotion-aware capabilities of the assistant were effectively implemented.

- **Hasya (Team Lead)**
  Led the development of the **voice recognition and image understanding modules**, enabling students to interact via speech and visual inputs. Implemented **image captioning and OCR functionality**, and played a major role in managing the **visualization rendering** using Graphviz and Matplotlib to convert complex queries into informative diagrams and charts.
- **Chakri**
  Designed and implemented the **backend AI logic**, integrating the **Ollama LLM** for natural language processing and response generation. Also built the **query filtering**

mechanism to ensure educational relevance and handled the **emotion-aware engagement logic** using OpenCV for facial expression detection.

- **Hima Sree**

  Developed the **frontend web interface**, ensuring a responsive and user-friendly UI. Handled critical **UI components**, including the chat interface, live **emotion display**, **file uploads**, and **voice input** integration. Additionally, contributed to **visualization rendering**, managing the integration of Graphviz and Matplotlib on the front end.

3. Multimodal Interaction Framework

This project is centered around the integration of multiple modalities of human-computer interaction — namely text, voice, images, and facial expressions — to create a dynamic, emotionally responsive learning assistant. Each modality is handled by a specialized module and contributes to a seamless, adaptive educational experience. Below is a technical breakdown of each modality and its function within the ClarifAI system:

3.1 Text-Based Interaction

Text remains the foundation of human-computer dialogue in most educational tools. In the ClarifAI assistant, text-based interaction is designed to be intuitive, fast, and deeply contextualized for classroom use. Students interact via a chat interface on the left side of the screen, where they can freely type doubts, questions, or even mathematical expressions. The goal is not just to answer, but to respond in a **student-friendly, educationally filtered manner** that aligns with academic expectations and avoids hallucinated or inappropriate content.

When a query is submitted, it enters a **layered processing pipeline**. First, the query is parsed for educational relevance through a custom keyword and structure-checking filter. This ensures questions are academically appropriate or trigger fallback logic for unsupported inputs. If the query involves mathematical terms or symbols, the system diverts it to a **math-aware handler**, ensuring LaTeX-style or numeric content is properly interpreted.

The query is then forwarded to a **locally hosted LLM**, typically gemma:2b running through the Ollama interface. This model, chosen for its efficiency and interpretability, generates a response that's streamed back to the frontend in real time. This live streaming mimics a human typing pattern, keeping the student engaged and reducing perceived response delay.

An essential feature is **context memory**, which preserves conversation history during a session. This allows the assistant to refer to previous questions and answers, maintaining continuity in explanations and preventing repetition. For example, if a student asks "Why is the answer 12?" after a long discussion, the assistant can infer the context from memory to respond accurately.

ClarifAI also flags certain patterns in student queries—such as those expressing uncertainty or phrased like “I don’t understand”—as potential confusion triggers. These are passed through a semantic layer that may append guiding prompts or offer simplified alternatives.

The text interaction also drives downstream visualization. If the generated response includes a stepwise list or classification, it’s automatically parsed to determine if a **flowchart or diagram should be generated**, bridging natural language with visual learning.

From a technical standpoint, all of this is managed by the Flask backend through **Server-Sent Events (SSE)**, allowing for non-blocking response rendering on the frontend. This design ensures that even large model outputs feel responsive, particularly critical for maintaining student attention.

The simplicity of typing a question belies the sophistication behind the scenes. This modality serves as the “anchor input,” supporting fallback mechanisms when voice or image inputs are noisy or ambiguous. It ensures that the assistant remains accessible even in low-resource environments or for students who prefer silent interaction.

Ultimately, text interaction in ClarifAI isn’t just about chatting with AI—it’s a real-time academic tutor that adapts its depth, clarity, and structure to each student’s query style and learning need.

3.2 Voice-Based Interaction

Voice interaction in ClarifAI enhances accessibility and student engagement by allowing learners to ask questions naturally through speech. This feature is especially valuable in classroom environments where typing may be slow, inconvenient, or a barrier to participation.

ClarifAI integrates voice input directly into the web interface using the **Web Speech API**, enabling students to click a microphone icon and speak their queries. The browser transcribes the spoken input in real-time and automatically displays the resulting text in the chat input field. This immediate visual feedback ensures transparency, allowing students to verify or edit their query before submitting.

Once submitted, the voice-transcribed text enters the standard backend pipeline, where it undergoes the same filtering and AI processing as typed input. This includes:

- Basic text cleanup and punctuation correction to improve model understanding.
- Detection of educational keywords and structure to ensure relevance.
- Forwarding the refined query to the **Ollama-hosted Gemma model** for generating an academic response.

To improve the accuracy of voice input, ClarifAI includes a lightweight preprocessing layer that filters out common filler words (“um”, “like”) and resolves minor transcription errors. This step is essential for aligning casual spoken language with the expectations of the language model.

The system is also designed to support a wide range of speech patterns and accents, making it effective in multilingual classroom settings. Since everything is handled within the browser, there's no dependency on external installations, ensuring a seamless experience across devices.

Voice interaction simplifies access to information and encourages participation, especially for students who find speaking more intuitive than typing. This mode makes ClarifAI more inclusive, responsive, and reflective of natural classroom conversation.

3.3 Image-Based Interaction

Image-based interaction allows students to upload photos of textbook pages, handwritten questions, or diagrams and receive relevant academic responses. This helps students who find it easier to share visuals rather than type detailed queries.

When an image is uploaded, ClarifAI processes it using two main techniques:

- **OCR (Optical Character Recognition):**
  Extracts any visible text from the image using EasyOCR. This is useful for typed or handwritten questions, formulas, and definitions.
- **Image Captioning:**
  For images without clear text (like diagrams or charts), the system uses a **pre-trained BLIP model** to generate a meaningful caption that describes what’s in the image.

Once the content is extracted, it is sent to the AI model for explanation. ClarifAI uses a **semantic similarity model** to ensure the response is educational and relevant to the image content. For example, an image of a labeled circuit will lead to an explanation of how that circuit works.

Key Features:

- Supports **handwritten and printed material**.
- Identifies **diagrams and conceptual visuals**.
- Automatically decides whether to extract text or generate a caption.
- Shows the processed result (text or caption) before responding.

This modality is especially useful when:

- Students can't describe the problem well in words.
- The question includes complex layouts, like math problems or science diagrams.

By turning visual input into academic understanding, ClarifAI helps students bridge physical learning materials with AI-driven support—making it easier to get help anytime, even without typing or speaking.

3.4 Visual Response Generation

ClarifAI doesn't just respond with text — it also generates **visual explanations** like flowcharts and charts to help students better understand complex topics. This visual support makes abstract or step-by-step concepts easier to grasp, especially for visual learners.

When the assistant detects that a question or response involves processes, categories, or structured data, it automatically triggers the visual generator module. This is handled through two main tools:

- **Flowchart Generation (Graphviz):**
  Used when the response involves logical steps or decision-based explanations (e.g., "How does blood circulate?" or "What is the process of photosynthesis?").

- **Chart Generation (Matplotlib):**
  Used for statistical or comparison-based content (e.g., “Show a bar graph of rainfall across months”).

The assistant analyzes the response content to determine if a visual is needed. If yes:

- It parses the structure of the text (like numbered lists or key-value pairs).
- Generates a code-based layout using Graphviz or Matplotlib.
- Renders the final image and displays it in the chat window alongside the answer.

Key Features:

- Automatically decides when visuals are helpful.
- Converts structured explanations into diagrams or graphs.
- No input needed from the student to request a visual.
- Improves clarity for step-by-step processes and comparisons.

Visual generation supports subjects like biology, algorithms, math, and economics where visuals reinforce learning. For example:

- A student asking about the **food chain** sees a flowchart of energy flow.
- A question on **monthly expenses** results in a pie chart.

By combining visuals with explanations, ClarifAI helps students **understand topics more clearly and remember them longer**. It turns detailed responses into **easy-to-follow diagrams and charts**, making learning more engaging and less overwhelming.

3.5 Emotion-Based Engagement Detection

A unique feature of ClarifAI is its ability to detect a student’s emotional state during interaction. By analyzing facial expressions through the webcam, the assistant can recognize signs of **confusion, frustration, or disengagement** and adapt its responses accordingly. This brings an added layer of personalization to the learning experience.

As the student types, speaks, or uploads a question, the system captures webcam frames in the background. These frames are analyzed using **OpenCV** along with an emotion classification model trained to detect expressions like:

- **Happy / Engaged**
- **Neutral**
- **Confused**
- **Frustrated**
- **Sad**
- **Angry**

If ClarifAI detects a **negative or confused emotion** during or shortly after a response is delivered, it automatically prepares a **simplified explanation** of the same topic. However, instead of instantly replacing the original answer, the assistant asks the student if they'd like to see a simpler version. If the student agrees, a new response is generated and shown.

Key Features:

- Runs in real-time while the assistant is in use.
- Emotion label is shown at the top-right of the screen.
- Triggers simplification only when necessary and after confirmation.
- Works without showing the video stream, maintaining privacy and focus.

This system ensures that **students who struggle don't fall behind silently**. Instead, the assistant becomes more responsive, offering help in a timely and thoughtful way — just like a real tutor would do when noticing a puzzled expression.

Emotion-aware responses make ClarifAI not only smart, but also **empathetic**, helping students feel supported throughout their learning journey.

3.6 Multimodal Integration and Flow

What makes ClarifAI truly powerful is its ability to **seamlessly combine multiple input modes**—text, voice, image, and emotion—into a unified, intelligent system. Each modality is handled independently yet works in coordination to deliver an adaptive and coherent learning experience.

When a student interacts with the assistant, the system automatically determines the **type of input** and routes it to the relevant processing pipeline:

- **Text queries** are filtered and sent directly to the AI model.
- **Voice input** is transcribed via Web Speech API before being treated like a text query.
- **Images** go through OCR or captioning, followed by semantic interpretation.
- **Emotion detection** runs in parallel, influencing the assistant's tone or triggering simplified explanations if needed.

Despite their differences, all modalities **converge at a central logic layer**. This layer performs:

- **Context tracking:** Maintains conversation history for better continuity.
- **Educational filtering:** Ensures that responses stay on-topic and relevant.
- **AI generation:** Sends the cleaned input to the local language model (e.g., Gemma) for streaming response generation.
- **Fallback handling:** Detects if an answer needs visual aids or simplification.

This orchestration makes ClarifAI a **truly multimodal educational assistant**—one that doesn't just answer questions, but understands how they are asked, who is asking them, and how they're feeling. It behaves like a responsive tutor that can listen, see, interpret, and adjust in real time.

The integration of these layers ensures that ClarifAI delivers a smooth, engaging, and personalized experience for every learner—regardless of how they choose to interact.

4. Methodology

The development of ClarifAI followed a structured, modular, and iterative methodology to ensure that each feature of the assistant aligned with real-world classroom needs. The project emphasized building a robust AI system that accepts multimodal inputs, processes them efficiently, and delivers real-time, context-aware, and emotion-sensitive academic support.

The methodology included key phases: requirement analysis, system architecture design, selection of tools and frameworks, implementation of core modules, and integration and testing. Each phase built on the outcomes of the previous stage, ensuring technical soundness and usability.

4.1 Requirement Analysis and Use-Case Definition

Before implementation, an in-depth analysis of classroom challenges was conducted. The focus was on identifying where existing digital tools fall short in engaging students and responding to their individual needs.

Key problems identified included:

- Students often hesitate to clarify doubts during or after lessons.
- Most learning tools rely solely on text input, which may not suit all learners.
- Lack of emotional feedback from systems leads to poor engagement.
- Explanations are often generic and not tailored to a student's understanding level.

From these, several use cases were defined to guide development:

- **Multimodal Input Handling**: Accept queries via text, speech, or image uploads.
- **Emotion-Sensitive Responses**: Use facial expression detection to identify confusion or frustration.
- **Visual Aid Generation**: Automatically convert responses into flowcharts or graphs when needed.

- **Real-Time Processing**: Ensure instant feedback with minimal latency.
- **Simplified Response Suggestions**: Offer easier explanations when the student appears confused.
- These use cases formed the foundation for designing system components and planning the logic for adaptive responses.

4.2 System Design and Architecture Planning

The system architecture was designed to support modular development and real-time execution. Each core functionality—voice processing, image understanding, emotion detection, and response generation—was implemented as an independent service, enabling clean separation of concerns.

The architecture consisted of the following layers:

- **Input Layer**: Captures user input via text, microphone, or image upload.
- **Preprocessing Layer**: Cleans and structures input text, transcribes voice to text, extracts or captions image content, and processes video frames for emotion.
- **Processing Layer**: Handles educational filtering, context tracking, emotion-aware decisions, and prepares visual data if required.
- **AI Model Layer**: Communicates with the locally hosted LLM via Ollama to generate educational responses.
- **Output Layer**: Sends the AI-generated answer and visual aids back to the frontend via a streaming mechanism.

A central decision logic integrates emotion feedback with query type and adjusts the response path accordingly. For example, if an image is uploaded and the student appears confused after the first explanation, the system can automatically prepare a simpler version along with a diagram.

4.3 Technology Stack Selection

Based on the design and intended functionality, specific technologies were selected for their compatibility, ease of integration, and performance:

- **Backend Framework**:
  - Flask: Lightweight and flexible, ideal for building REST endpoints and handling AI workflows.
- **Language Model**:
  - Ollama with Gemma 2B: Provides fast, locally hosted LLM access for educational Q&A, minimizing dependency on cloud APIs.
- **Speech Processing**:
  - Web Speech API: Enables real-time speech-to-text in the browser with no additional installations.
- **Image Understanding**:
  - EasyOCR: Extracts printed or handwritten text from images.
  - BLIP (via Hugging Face): Generates contextual captions for diagrams or unlabeled images.
  - SentenceTransformers: Measures semantic similarity between extracted content and academic topics.
- **Emotion Detection**:
  - OpenCV: Used for real-time webcam frame capture and facial expression detection using a lightweight emotion classifier.
- **Visualization**:
  - `Graphviz`: Generates flowcharts from logical steps or processes.
  - `Matplotlib`: Renders bar, pie, and line charts for statistical content.
- **Frontend Technologies**:
  - `HTML`, `CSS`, `JavaScript`: Build a responsive chat interface with support for voice input, emotion label display, and visual response rendering.

The stack was chosen to balance performance, ease of development, and alignment with academic project constraints (e.g., offline operation, open-source tools, minimal resource usage).

4.4 Implementation

Implementation was carried out in well-defined modules to ensure maintainability and testability:

- **Text Interaction Module**:
  - Accepts queries from the chat input or transcribed voice input.

    - Applies educational filtering and query classification.
    - Maintains session memory for contextual follow-up.
- **Voice Input Module**:
    - Uses Web Speech API to capture audio and transcribe in real time.
    - Displays transcribed text to the user before submission.
- **Image Processing Module**:
    - For images with text: Uses OCR and sends cleaned text to the LLM.
    - For diagrams: Uses BLIP to caption and semantically match content before generating responses.
- **Emotion Detection Module**:
    - Runs continuously in the background capturing webcam frames.
    - Classifies emotions like “happy”, “confused”, “sad”, or “angry”.
    - Flags negative emotions and suggests simplified explanations.
- **Visual Generation Module**:
    - Parses AI responses for logical steps or data patterns.
    - Generates and encodes visual assets (charts/flowcharts) to display alongside text.
- **Frontend Interface**:
    - Includes chat panel, mic toggle, image upload button, and top-right emotion label.
    - Supports streaming of AI responses and rendering of visual outputs.

4.5 Integration and Testing

Once individual components were functional, they were integrated into a single cohesive system. A strong emphasis was placed on ensuring smooth communication across modules and responsive user interaction.

Testing involved:

- **Unit Testing**: Verifying each module independently for input/output consistency.

- **Multimodal Flow Testing**: Ensuring that switching between text, voice, and image input worked seamlessly.
- **Emotion Detection Evaluation**: Observing if facial expressions were accurately classified under varying lighting and webcam angles.
- **Response Relevance Testing**: Manually checking AI output for educational clarity and accuracy.
- **UI Stress Testing**: Monitoring performance during rapid input switches, simultaneous inputs, and long streaming responses.

Integration tests confirmed that the assistant could handle a full learning session—including follow-up questions, emotion-triggered simplifications, and multimodal input—without crashing or lagging.

5. System Architecture

The ClarifAI assistant is built on a modular, scalable, and event-driven architecture designed to process multimodal inputs in real time and generate educational responses tailored to each user's needs. The architecture integrates independent pipelines for text, voice, image, and emotion, coordinated through a centralized decision logic and a responsive frontend interface.

The system architecture consists of five key layers:

5.1 Input Layer

This layer captures user queries in various formats:

- **Text Input**: Entered through the chat interface.
- **Voice Input**: Captured and transcribed using the Web Speech API.
- **Image Input**: Uploaded via the interface to be analyzed.
- **Webcam Frames**: Captured continuously for real-time emotion monitoring.

Each input type is directed to its respective preprocessing pipeline for further analysis.

5.2 Preprocessing and Interpretation Layer

Each input modality undergoes transformation and normalization:

- **Text**: Cleaned and filtered for academic relevance.
- **Voice**: Transcribed into readable text and optionally corrected.
- **Image**:
    - OCR extracts embedded text (via EasyOCR).
    - BLIP generates captions for diagrams and unstructured images.
- **Emotion**: Webcam frames analyzed by OpenCV to classify expressions like happy, sad, confused, or frustrated.

This layer ensures all inputs are structured and semantically interpretable before forwarding to the model.

5.3 Core Logic and Processing Layer

This is the heart of ClarifAI, where the assistant makes decisions based on context, input type, and emotion state:

- **Query Routing**: Determines if the query should trigger standard LLM response, a fallback handler (for math), or visual generation.
- **Context Management**: Maintains session history for coherent follow-up queries.
- **Emotion-Driven Simplification**: Triggers simplified rephrasing of answers if negative emotions are detected after initial response.
- **Response Enrichment**: Decides whether a visual (flowchart/chart) should accompany the textual answer.

All cleaned and interpreted inputs are passed to this layer for intelligent coordination.

5.4 AI Response Generation Layer

The final structured query is sent to the **LLM via the Ollama interface**, where:

- **Educational queries** are answered contextually.
- **Image-derived inputs** are mapped to academic topics using semantic similarity.
- **Fallbacks** for mathematical or unsupported content are handled gracefully.

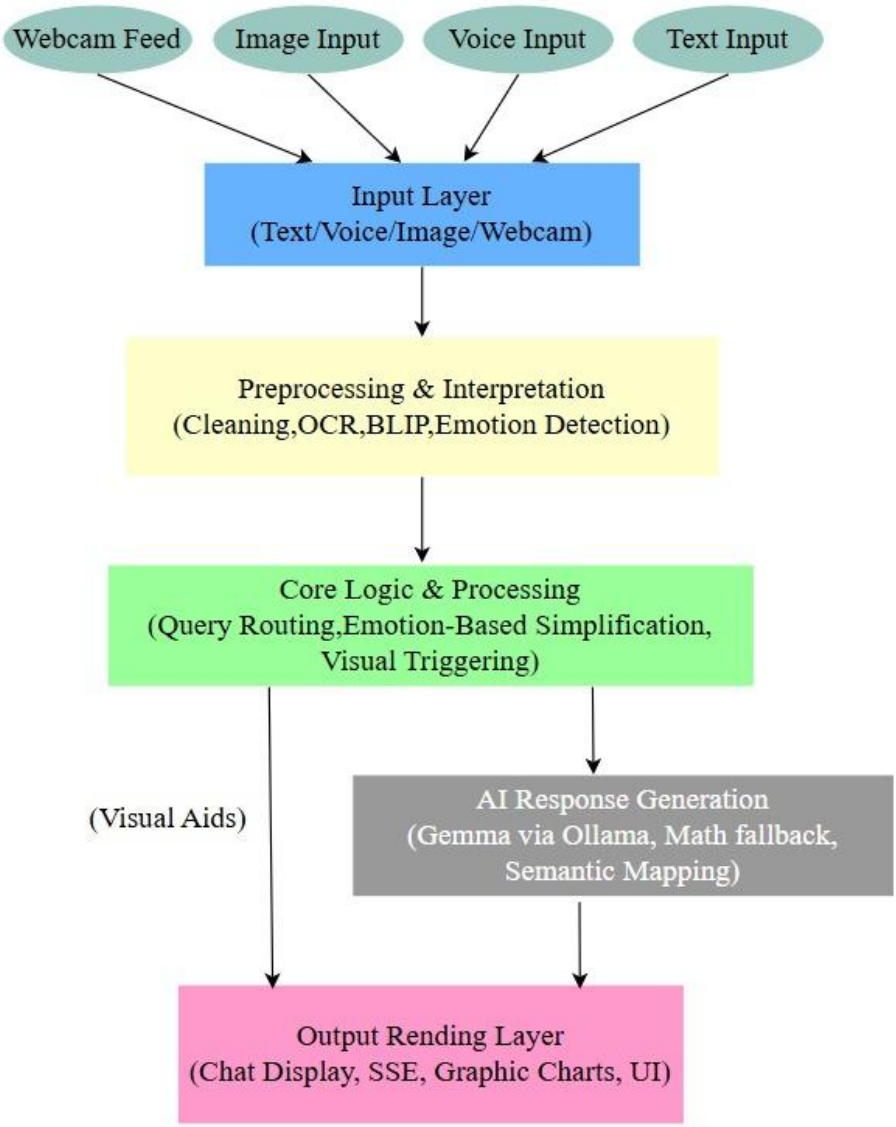
The output is streamed back to the frontend using **Server-Sent Events (SSE)** for a live typing effect, improving engagement.

5.5 Output Rendering Layer

This layer ensures clear delivery of the response:

- **Frontend Display**: Shows the chat reply, emotion label, and any visual aids.
- **Visual Rendering**: Generates flowcharts (Graphviz) or graphs (Matplotlib) dynamically and embeds them beside responses.
- **UI Interaction**: Enables continuous chat, voice re-input, and image upload while maintaining smooth emotion updates.

The frontend is kept minimal and distraction-free, allowing users to focus on the learning interaction while receiving adaptive feedback.


Webcam Feed
Image Input
Voice Input
Text Input
Input Layer
(Text/Voice/Image/Webcam)
Preprocessing & Interpretation
(Cleaning,OCR,BLIP,Emotion Detection)
Core Logic & Processing
(Query Routing,Emotion-Based Simplification,
Visual Triggering)
(Visual Aids)
AI Response Generation
(Gemma via Ollama, Math fallback,
Semantic Mapping)
Output Rending Layer
(Chat Display, SSE, Graphic Charts, UI)

6. Challenges Faced

Emotion Detection Accuracy

The system sometimes misclassified facial expressions, especially under poor lighting or with subtle faces. This was improved by reducing detection frequency and applying confirmation before response simplification.

Webcam Flickering

Accessing the webcam from multiple threads caused flickering during emotion detection. A shared camera stream and frame synchronization fixed the issue.

Visual Generation Issues

Flowcharts sometimes appeared cluttered or disconnected due to inconsistent structure detection. Custom parsing logic and layout adjustments improved clarity.

Image Understanding Limitations

OCR often failed with handwritten inputs, and BLIP occasionally gave vague captions. Combining both techniques helped extract more meaningful content.

Multimodal Integration Complexity

Coordinating inputs from text, image, and emotion detection without slowing the system was challenging. This was handled through asynchronous processing and background threading.

7. Results and Discussion

After successful integration and testing, ClarifAI demonstrated its effectiveness as a real-time, multimodal educational assistant. Each feature performed as expected across various input types and learning scenarios.

- **Text and voice input** produced accurate academic responses across subjects like science, math, and general knowledge. The streamed output ensured better readability and engagement.
- **Image-based queries**, especially those involving textbook diagrams or handwritten problems, were successfully interpreted using a combination of OCR and image captioning. The assistant was able to deliver relevant explanations in most cases.
- **Emotion detection** accurately classified facial expressions such as neutral, confused, or frustrated in real-time. This allowed the system to suggest simplified responses where needed, enhancing personalization.
- **Visual aid generation** improved comprehension for process-driven or statistical topics. Flowcharts and charts added clarity to otherwise complex textual explanations.
- The system remained responsive under all test conditions, maintaining real-time behavior with an average response time under 3–4 seconds. Multimodal coordination was smooth, and no major UI lags were observed during extended sessions.

Overall, ClarifAI successfully met its objective of supporting students with flexible input options, real-time explanations, and adaptive responses based on engagement. The assistant proved to be stable, accessible, and impactful in educational use cases.

8. Conclusion

The development of ClarifAI has demonstrated the practical application of multimodal AI in enhancing the classroom learning experience. By combining text, voice, image, and emotion-based inputs, the assistant is capable of delivering real-time, context-aware, and personalized academic support.

The project successfully addressed core challenges in student engagement and understanding by:

- Enabling natural interactions through speech and image-based queries.
- Responding with clear explanations, supported by visual aids.
- Detecting confusion or frustration and adapting responses accordingly.

ClarifAI not only showcases the technical feasibility of such systems but also highlights their potential to make learning more accessible, responsive, and emotionally aware. Through this project, we have laid a strong foundation for building intelligent educational assistants that can evolve into supportive learning companions in future classrooms.

9. Future Scope

While ClarifAI has achieved its goal of delivering real-time, multimodal educational support, there are several opportunities to extend its functionality and impact:

- **Improved Emotion Analysis**
  Incorporating more advanced emotion recognition models and using head pose or gaze tracking can enhance the accuracy of engagement detection.
- **Curriculum Integration**
  Mapping responses directly to NCERT, CBSE, or other academic standards would make the assistant classroom-ready for formal education systems.
- **Offline Mode and Edge Deployment**
  Optimizing models to run entirely offline or on Intel edge devices would improve accessibility in areas with limited internet access.
- **Multilingual Support**
  Expanding voice and text capabilities to support regional languages can improve inclusivity for diverse learners.
- **Student Progress Tracking**
  Logging past queries and emotions over time could help teachers identify learning gaps or emotional trends and adapt their teaching accordingly.
- **Mobile and PWA Deployment**
  Packaging ClarifAI as a mobile app or Progressive Web App (PWA) would increase its usability for students outside classroom hours.

ClarifAI has the potential to evolve into a powerful digital learning companion, bridging the gap between intelligent technology and human-centered education.

10. References

1. **Hugging Face Transformers** – for BLIP image captioning and sentence-transformer models
   https://huggingface.co/transformers
2. **Ollama – Local LLM Framework** – used to run Gemma 2B model for educational query responses
   https://ollama.com
3. **Gemma 2B** – lightweight transformer model used for generating context-aware academic explanations
   https://ai.google.dev/gemma
4. **EasyOCR** – used for extracting printed and handwritten text from uploaded images
   https://github.com/JaidedAI/EasyOCR
5. **BLIP (Bootstrapped Language-Image Pretraining)** – zero-shot captioning model for understanding diagrams
   https://github.com/salesforce/BLIP
6. **OpenCV** – for real-time webcam access and facial expression analysis
   https://opencv.org
7. **Graphviz** – used for flowchart and process diagram generation from AI responses
   https://graphviz.org
8. **Matplotlib** – for generating bar, pie, and line charts based on academic data
   https://matplotlib.org
9. **Flask** – lightweight web framework for building backend APIs and managing real-time input/output
   https://flask.palletsprojects.com
10. **Web Speech API** – used for capturing and transcribing voice input in the browser
    https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API
11. **GitHub Repo** – ClarifAI Project
    https://github.com/chakri9133/ClarifAI-AI-Assistant

12. **Demo Video**

    https://drive.google.com/file/d/1O16NL2WnBiTnAWRbJlajOMuEaR2oP_RI/view?usp=sharing

11. Appendix A – Screenshots

The following screenshots illustrate the working of the ClarifAI system across various input modes and functionalities. These visuals support the explanation of features described in Section 3.

Fig. 1 – Text-Based Interaction

This screenshot shows a student entering a text query into the chat interface. The assistant processes the query and returns a streamed, contextual academic response.

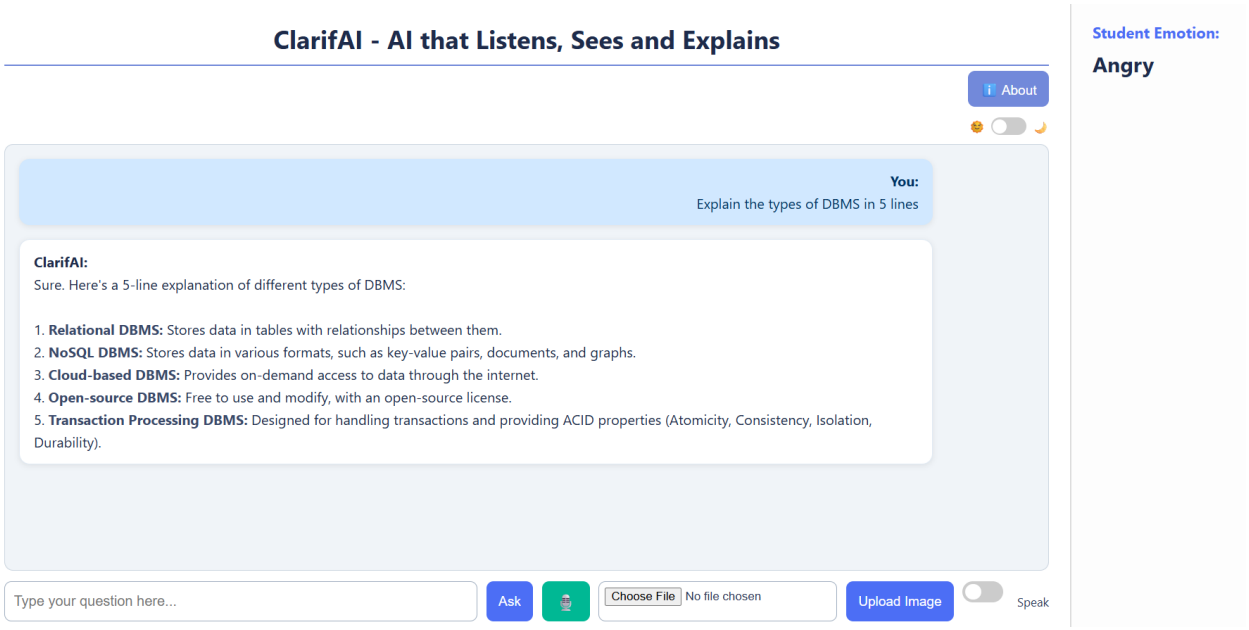


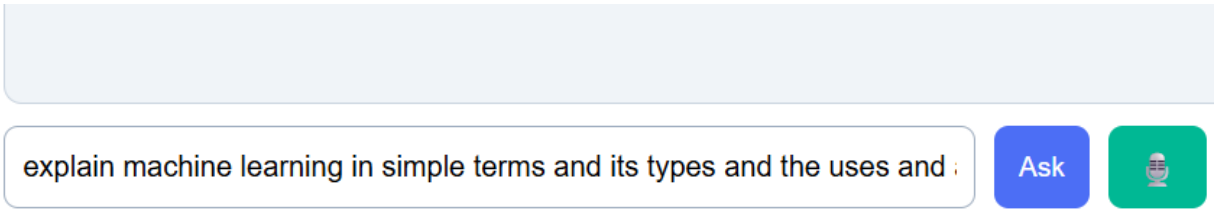
Fig. 2 – Voice Input and Transcription

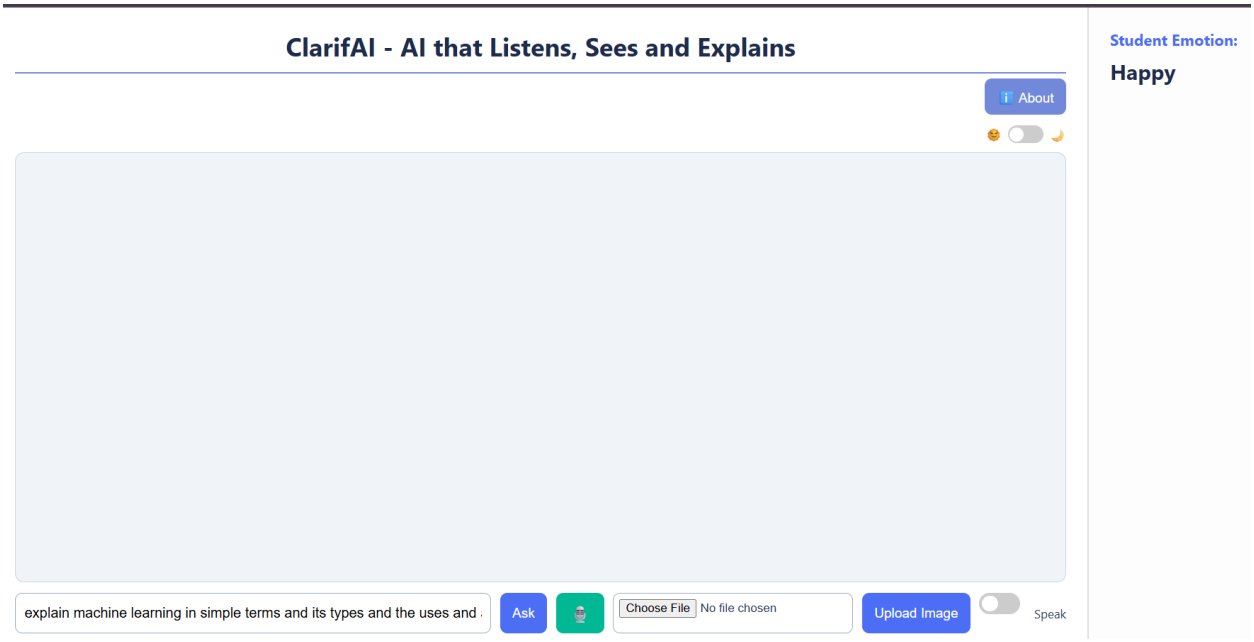
Here, the user interacts with the assistant using the microphone icon. The spoken input is transcribed and appears in the text box before being submitted.

The user gave a voice query:

"Explain machine learning in simple terms and the types of machine learning with uses and applications."

The assistant transcribed the input accurately and prepared a relevant academic response.





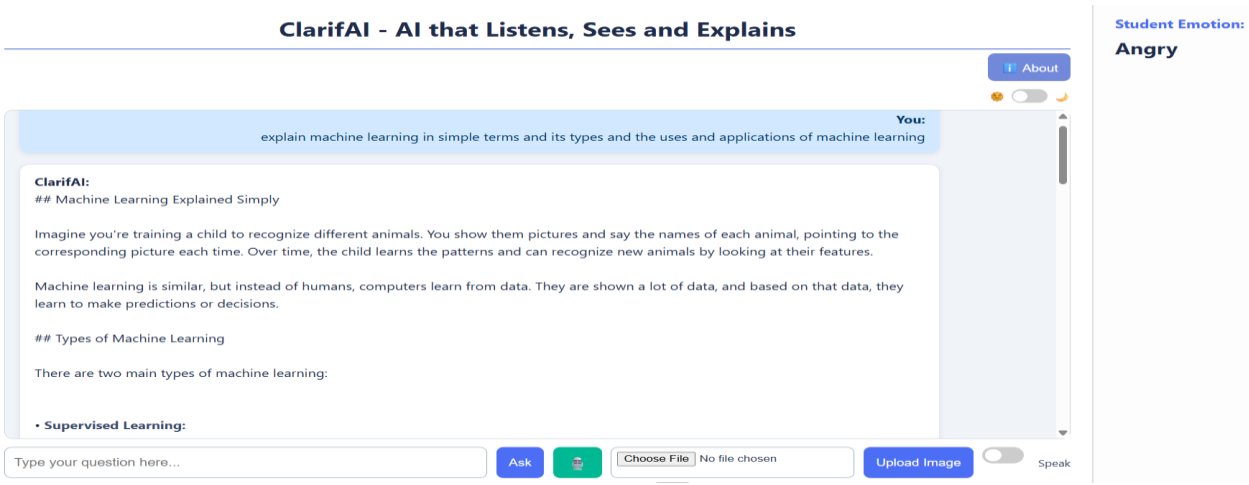


Fig. 3 – Image Upload and Interpretation

This image demonstrates the assistant handling an uploaded image of a diagram or handwritten question. The system extracts content using OCR or image captioning and generates a relevant explanation.

The user uploaded an image explaining the difference between **supervised** and **unsupervised learning**. The assistant extracted the visual content and responded with a simplified explanation of both types, including how labeled data is used in supervised learning and clustering in unsupervised learning.

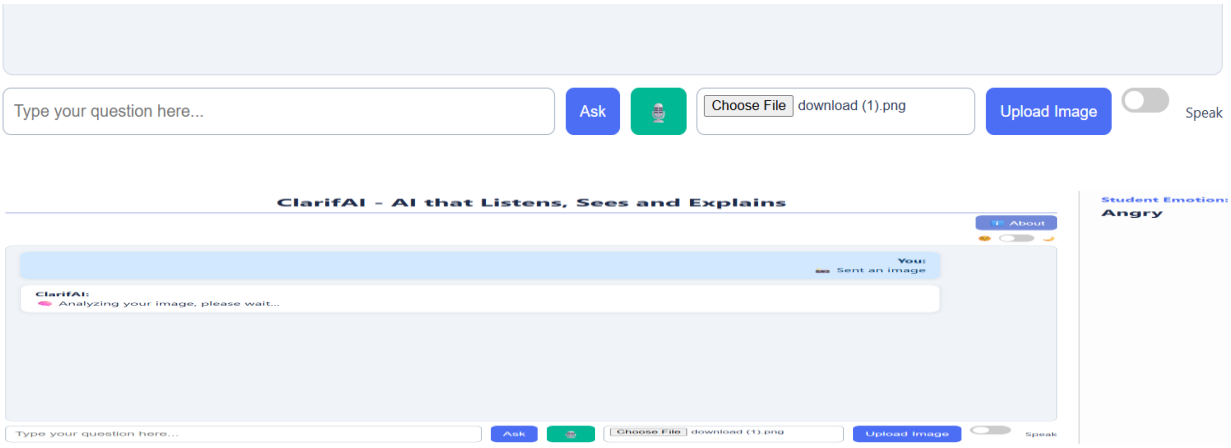
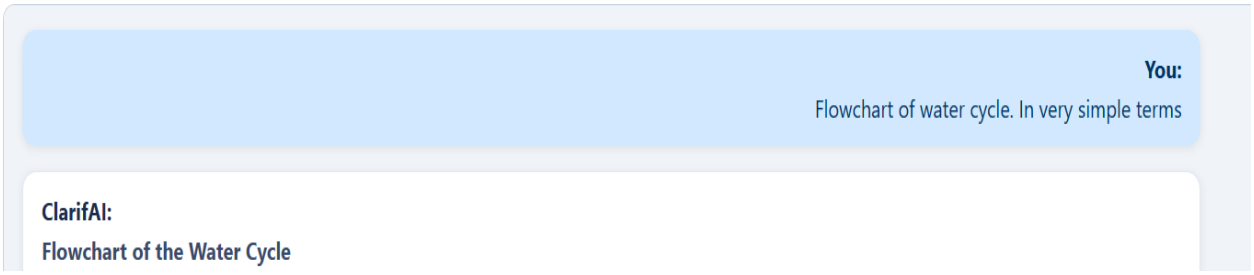


Fig. 4 – Visual Aid (Flowchart) Generation

This flowchart is generated automatically from a response that includes a process explanation. The assistant uses Graphviz to visually represent steps or logic flows.

This section represents the assistant's ability to generate flowcharts from structured AI responses using Graphviz. While the module is still under refinement and may not render every response correctly, the logic to detect stepwise content and trigger visual generation has been implemented. Improvements are planned to enhance layout and parsing accuracy.



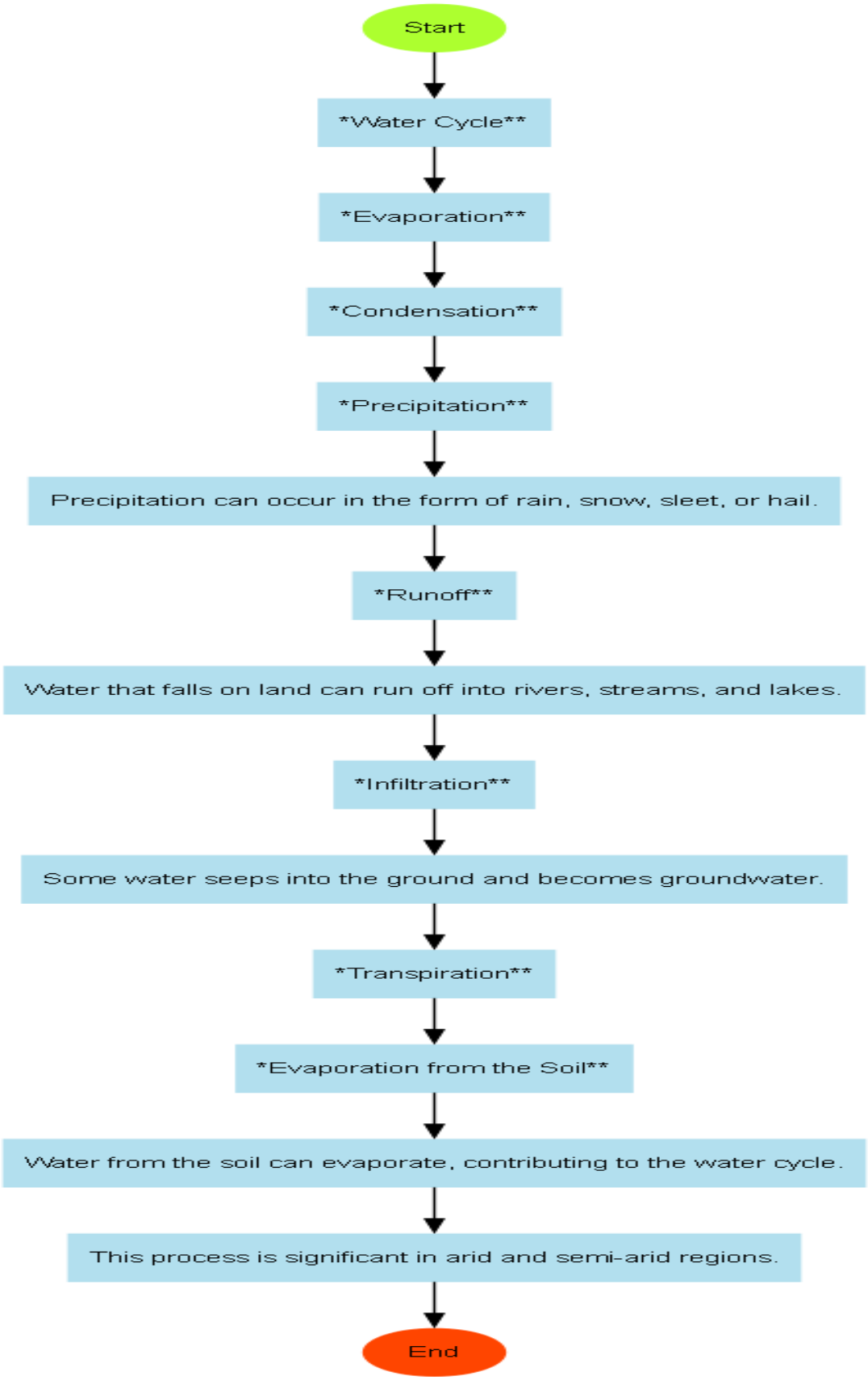

Start
*Water Cycle**
*Evaporation**
*Condensation**
*Precipitation**
Precipitation can occur in the form of rain, snow, sleet, or hail.
*Runoff**
Water that falls on land can run off into rivers, streams, and lakes.
*Infiltration**
Some water seeps into the ground and becomes groundwater.
*Transpiration**
*Evaporation from the Soil**
Water from the soil can evaporate, contributing to the water cycle.
This process is significant in arid and semi-arid regions.
End

Fig. 5 – Emotion Detection Label Display

This screenshot shows the real-time emotion detection feature, where the assistant analyzes the student's facial expression and displays a label (e.g., "Confused") in the top-right corner of the interface.