International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 04 | Apr 2026 www.irjet.net p-ISSN: 2395-0072

NEXUS AI: Multimodal AI Content Studio

G Sainath[1], G Sriram[2], K Harshavardhan[3], K Sudharshan[4] , MD Abdul Aleem[5]

[1,2,3,4]UG Students, Department of Computer Science and Engineering, JBREC, Hyderabad, India
[5]Assistant Professor, Department of Computer Science and Engineering, JBREC, Hyderabad, India

---

Abstract - *NEXUS AI is a Multimodal AI Content Studio that integrates multiple AI-powered generation capabilities into a single unified platform. Built on a Flask backend with a dynamic JavaScript frontend, it leverages the Groq API to deliver rapid, high-quality outputs across seven distinct modes: Text Generation, Vision Analysis, Code Generation, Social Media content, Image Prompts, Image Generation, and Translate & Adapt. The system follows a clean layered architecture where user requests flow through a browser UI, an API abstraction layer, a Flask processing backend, and finally the Groq AI service. This paper presents the complete system design, architectural overview, workflow sequence, implementation details, and validated output results demonstrating NEXUS AI's effectiveness as a versatile AI content studio for researchers, developers, and creative professionals.*

Key Words: Multimodal AI, Content Generation, Groq API, Flask Backend, Natural Language Processing, Vision Analysis, Code Generation, REST API, Large Language Models, Web Application

1. INTRODUCTION

NEXUS AI is a comprehensive Multimodal AI Content Studio designed to democratize access to powerful AI capabilities through an intuitive web interface. In today's rapidly evolving AI landscape, users require a unified platform capable of handling diverse content generation tasks—from natural language text and code to image analysis and multilingual translation.

The system is built upon a modular three-tier architecture: a responsive browser-based frontend, a RESTful API middleware layer (api.js), and a Python Flask backend (app.py) that interfaces directly with the Groq AI service. This design ensures clean separation of concerns, high maintainability, and scalability.

NEXUS AI addresses seven key content domains in a single studio environment, making it suitable for individual developers, academic researchers, marketing professionals, and content creators alike. The platform's multimodal capability—spanning text, vision, and code—represents a significant step toward fully integrated AI-assisted workflows.

1.1 Motivation

Adapt into one cohesive interface powered by state-of-the-art LLM capabilities

1.2 Objectives

- Design a scalable, modular multimodal AI platform.
- Integrate Groq API for high-speed LLM inference.
- Implement a clean REST API abstraction layer between frontend and backend.
- Deliver an intuitive, dark-themed UI with real-time generation feedback.
- Validate system outputs across all seven generation modes.

2. SYSTEM ARCHITECTURE

Table 2 summarises the key components and their technologies. The Frontend (Browser) layer comprises index.html, style.css, and script.js. The API Layer (api.js) acts as the communication bridge, sending POST /generate requests to the Flask Backend (app.py). The backend processes the prompts, builds context-aware requests, and forwards them to the external Groq API, returning structured JSON responses

Table - 1: NEXUS AI System Components & Technologies

Component	Technology	Role
Frontend	HTML/CSS/JS	User Interface & Interaction
API Layer	api.js (ES6)	HTTP Abstraction & Serialisation
Backend	Flask (Python)	Prompt Engineering &

Component	Technology	Role
		Logic
AI Service	Groq API	LLM Inference Engine
Model	llama3-8b-8192	Language Generation

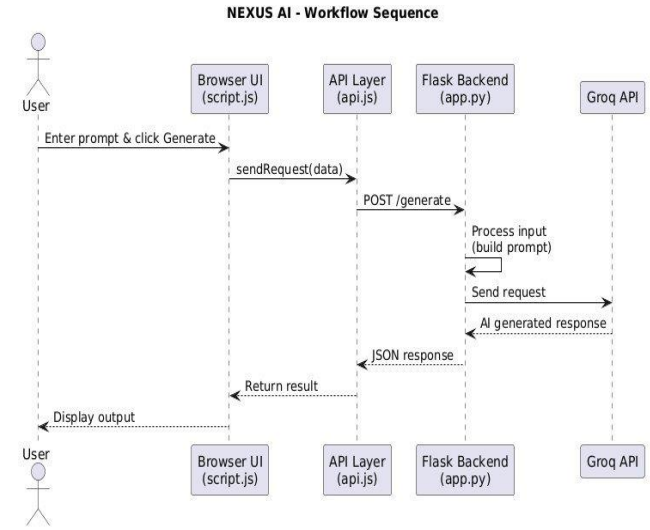

Fig - 1: NEXUS AI System Architecture

The NEXUS AI system follows a three-tier client-server architecture, as illustrated in Fig. 1. The architecture ensures loose coupling between the presentation and business logic layers. Each component can be independently scaled or replaced—for instance, the Groq API can be swapped with another LLM provider without

2.1 Frontend Layer

The frontend is a single-page application (SPA) rendered in the browser. It consists of a navigation sidebar listing the seven generation modes, a dynamic content area with mode-specific input controls (prompt textarea, max tokens slider, temperature control), and an output panel with Copy, Clear, and Save actions. The UI uses a dark theme with purple accent colors for improved focus during extended use.

2.2 API Abstraction Layer (api.js)

api.js implements the sendRequest(data) function that serialises user input into a structured JSON payload and dispatches it via HTTP POST to the Flask /generate endpoint. It handles asynchronous responses, error states, and response parsing, insulating script.js from direct backend concerns.

2.3 Flask Backend (app.py)

The Flask backend receives the POST /generate request, extracts the mode and prompt parameters, constructs mode-specific system prompts, and calls the Groq API using the llama3-8b-8192 model. The response is parsed and returned as a JSON object to the API layer. CORS is enabled to support browser-based clients.

2.4 Groq API Integration

Groq provides ultra-fast LLM inference, significantly reducing response latency compared to traditional inference endpoints. The backend uses the groq Python SDK with structured message arrays containing system

3. WORKFLOW SEQUENCE

Fig. 2 presents the complete UML sequence diagram of the NEXUS AI request-response workflow. The interaction begins when the User enters a prompt and clicks Generate in the Browser UI (script.js).

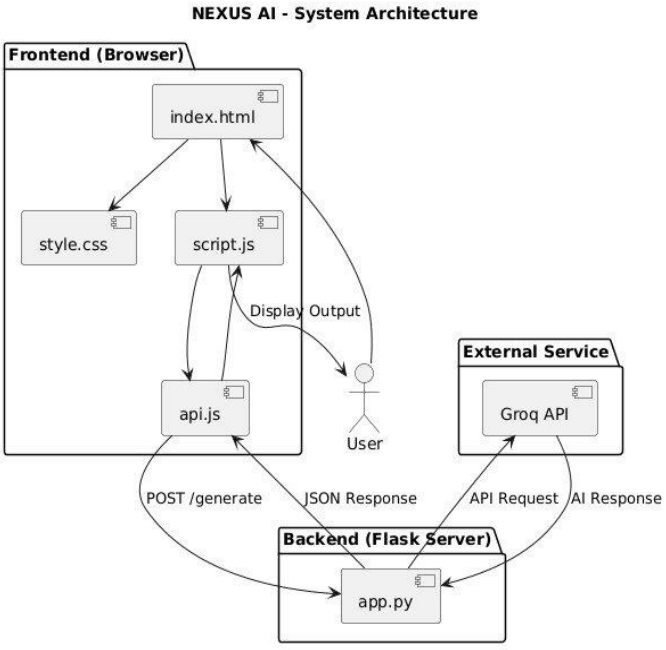

Fig - 2: NEXUS AI Workflow Sequence Diagram

Step 1 — User Input: The user selects a generation mode, enters a prompt, and optionally configures max tokens and temperature parameters.

Step 2 — API Call: script.js invokes sendRequest(data) in api.js, serialising the input to JSON and dispatching a POST /generate request to Flask.

Step 3 — Backend Processing: app.py validates the input and constructs a mode-specific system prompt that guides the LLM's response style.

Step 4 — Groq API Request: Flask sends the prompt to the Groq API using the llama3-8b-8192 model and awaits the AI-generated completion.

Step 5 — Response Chain: Groq API → Flask JSON → api.js → script.js renders output in the output panel.

Step 6 — User Display: Generated content appears with Copy, Clear, and Save controls for post-generation management.

4. IMPLEMENTATION

4.1 Generation Modes

Table 2 summarises all seven generation modes provided by NEXUS AI, each backed by a dedicated mode-specific system prompt engineered within the Flask backend.

and user roles, enabling context-aware responses tuned to each generation mode.

when the User enters a prompt and clicks Generate in the Browser UI (script.js).

Table - 2: NEXUS AI Generation Modes

Mode	Description
Text Generation	Articles, stories, essays, marketing copy
Vision Analysis	Image understanding & multimodal reasoning
Code Generation	Write, explain, debug code (all languages)
Social Media	Posts, captions, threads (Twitter, Instagram)
Image Prompts	Prompts for DALL-E, Midjourney, Stable Diffusion
Image Generation	AI-generated visuals from text descriptions
Translate & Adapt	Multilingual translation with cultural adaptation

Text Generation produces articles, essays, stories, and marketing copy using prompt-guided LLM completions. Vision Analysis accepts uploaded images (JPG, PNG, GIF, WebP) alongside a user question for multimodal visual reasoning. Code Generation generates, explains, debugs, and optimises code across all major programming languages.

4.2 Security & Configuration

Table 3 presents the key configuration parameters and security settings enforced by the NEXUS AI backend. API keys are stored as environment variables and never exposed to the frontend. CORS headers are restricted to approved origins in production deployments. Input length is validated server-side to prevent prompt injection and excessive token consumption

Table - 3: Configuration Parameters & Security Settings

Parameter	Default	Range / Notes
Max Tokens	1024	256 – 8192
Temperature	0.7	0.0 – 1.0
Model	llama3-8b-8192	Groq hosted
API Auth	Env Var	Never sent to client

5. RESULTS & OUTPUTS

NEXUS AI was tested across all seven generation modes. Below we present three representative outputs demonstrating the platform's capabilities.

Output 1 — Text Generation Interface

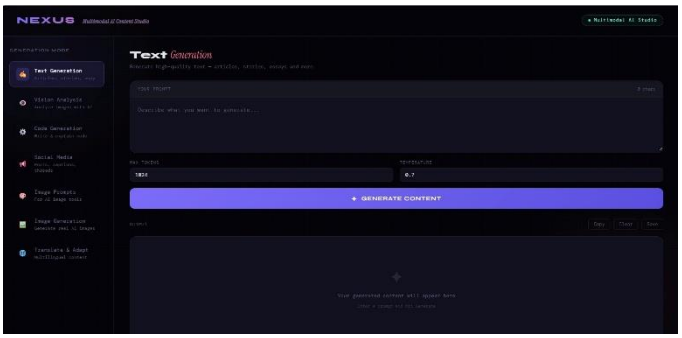

Fig - 3: Text Generation Mode — Main Interface

Fig. 3 shows the Text Generation mode of the NEXUS AI studio. The user enters a natural language prompt in the YOUR PROMPT field and configures Max Tokens (default 1024) and Temperature (default 0.7) to control output length and creativity. Upon clicking GENERATE CONTENT, the LLM produces high-quality articles, stories, or essays in the output panel below, with Copy, Clear, and Save controls for post-generation management.

Output 2 — Code Generation

Fig - 4: Code Generation Mode — HTML Login Page

Fig. 4 demonstrates Code Generation responding to "Create a basic HTML login page using only HTML." The system produced a well-structured, commented HTML document with username/password fields and a submit button, including an explanatory summary, confirming the model's ability to generate functional, production-ready code.

Output 3 — Vision Analysis

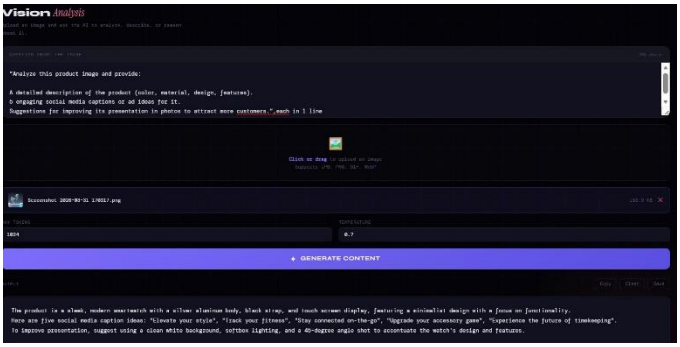

Fig - 5: Vision Analysis Mode — Product Image Analysis

Fig. 5 illustrates Vision Analysis with a smartwatch product image. The AI correctly identified the product, generated five social media caption ideas, and recommended professional photography techniques (white background, softbox lighting, 45-degree angle) to enhance e-commerce product presentation.

6. CONCLUSIONS

NEXUS AI successfully demonstrates the viability of a unified, multimodal AI content studio built on a lightweight Flask-Groq architecture. The system's clean three-tier design, rapid response times enabled by Groq's inference engine, and intuitive dark-themed interface collectively deliver a professional-grade tool for diverse AI-powered content workflows.

The platform's modular design allows straightforward extension to additional generation modes, model providers, or authentication layers. Future work includes adding user session management, prompt history, streaming response output, and support for additional multimodal inputs including audio and video analysis.

The validated outputs across Text Generation, Code Generation, and Vision Analysis confirm NEXUS AI's reliability and practical utility for researchers, developers, and creative professionals seeking a single integrated AI studio.

ACKNOWLEDGEMENT

The author would like to thank the Groq team for providing accessible LPU-accelerated API infrastructure and the Pollinations AI team for their open image generation API, both of which made this project feasible as an individual development effort.

REFERENCES

[1] OpenAI, "GPT-4 Technical Report," arXiv:2303.08774, 2023.

[2] Meta AI, "The LLaMA 3 Herd of Models," arXiv:2407.21783, 2024.

[3] A. Q. Jiang et al., "Mixtral of Experts," arXiv:2401.04088, 2024.

[4] A. Vaswani et al., "Attention Is All You Need," Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017

[5] R. Rombach et al., "High-Resolution Image Synthesis with Latent Diffusion Models," IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10684–10695, 2022.

[6] Pollinations AI, "Open-source generative media API," [Online]. Available: https://pollinations.ai

[7] A. Grinsztajn, E. Oyallon, and G. Varoquaux, "Flask: A micro web framework for Python," Python Software Foundation, 2010.