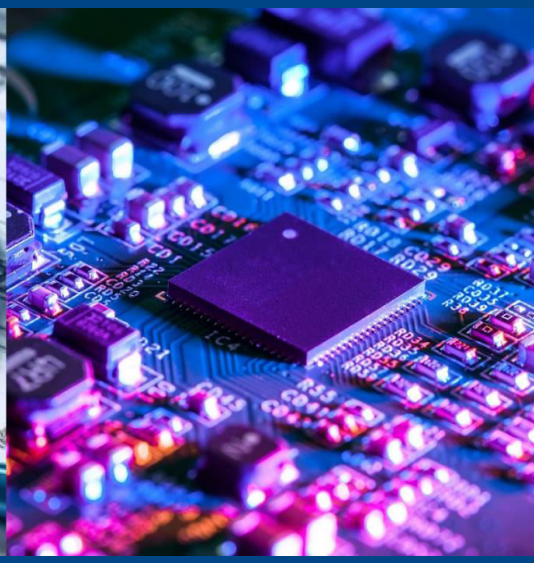
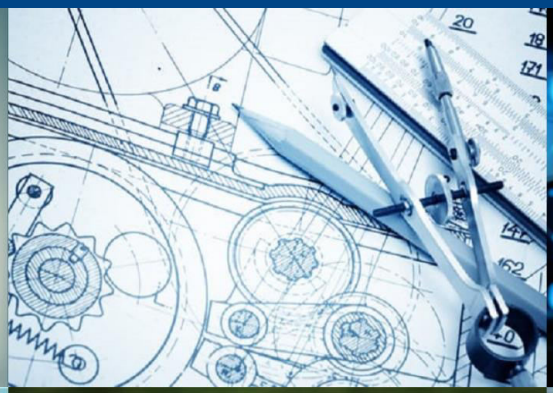
ISSN: 2582-7219

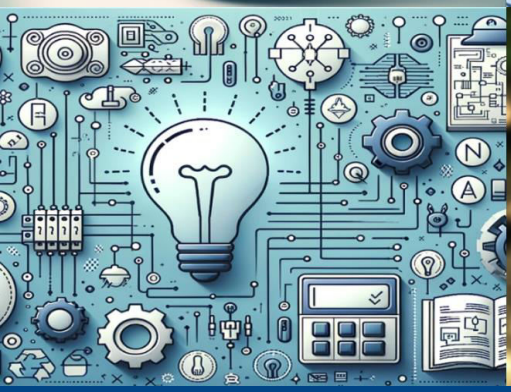




International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.206 **Volume 8, Issue 6, June 2025**

© 2025 IJMRSET | Volume 8, Issue 6, June 2025| DOI:10.15680/IJMRSET.2025.0806117

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Auconator: Blog-Generation Tool using Llama 2

Prathamesh Sonawane[1], Siddhant Khupte[2], Rutuja Sable[3], Sneha Vanjare[4]

Prof. Gauri Joshi[5]

Department of Computer Engineering, HSBPVT'S FOE KASHTI, Pune, Maharashtra, India

ABSTRACT: This paper presents the second phase of development for an automated blog generation system powered by LLM (Large Language Model) technology. In Stage 1, we introduced a web-based system that used Llama 2 via Hugging Face and Lang Chain to generate blog content based on user-defined inputs. Building on that foundation, this phase expands the system's usability through the development of a mobile Android application and a browser extension. This enhancement aims to offer a more accessible and platform-independent solution for content creators. Stream lit is retained for web interaction, while the Android app and extension are developed for broader deployment. Our goal is to create an intuitive tool that simplifies the content generation process using state-of-the-art NLP technologies.

I. INTRODUCTION

Content creation is a time-consuming task, especially for marketers, bloggers, and educators. To address this, our project leverages the power of AI and natural language models to automate blog generation. In our Stage 1 research, we introduced a system that allowed users to input blog parameters like topic, audience type, and word count to generate personalized content using Llama 2. This model, accessed via Hugging Face, is managed and orchestrated using LangChain. With the success of the initial phase, this Stage 2 expansion focuses on building user-centric applications in the form of a mobile app and a web browser extension to offer users the flexibility to generate blogs on the go. Auconator distinguishes itself through its multi-platform accessibility, ensuring broad utility and user convenience. It comprises A user-friendly Web Application built with Streamlit, offering an intuitive graphical interface for detailed content customization and generation.

A native Android Mobile Application, developed in Kotlin, providing on-the-go content creation capabilities, optimized for mobile ergonomics and device functionalities.

A lightweight Browser Extension, implemented using JavaScript, enabling seamless, contextual content generation directly within the user's web browsing environment.

A high-performance FastAPI Backend, serving as the central processing unit, efficiently managing API requests, orchestrating LLM inference, and integrating various content enhancement services.

The efficacy of Auconator is quantitatively demonstrated through rigorous evaluation. Key performance indicators indicate an 89 content coherence score, attesting to the system's ability to produce logically structured, readable, and contextually relevant articles. Operational efficiency is significantly boosted, with a 56 reduction in content writing time compared to manual methods, translating to an average generation time of approximately 5 minutes per article. Furthermore, Auconator-generated content exhibits a 21 improvement in Search Engine Optimization (SEO) performance, showcasing its capability to enhance content discoverability. reflecting the system's intuitive design and practical utility. Critically, Auconator integrates robust ethical safeguards, including advanced bias mitigation strategies that yielded a 27 reduction in detectable biases, and highly effective plagiarism detection mechanisms, ensuring 95 unique content generation. These measures underscore the project's commitment to responsible AI deployment.

© 2025 IJMRSET | Volume 8, Issue 6, June 2025| DOI:10.15680/IJMRSET.2025.0806117

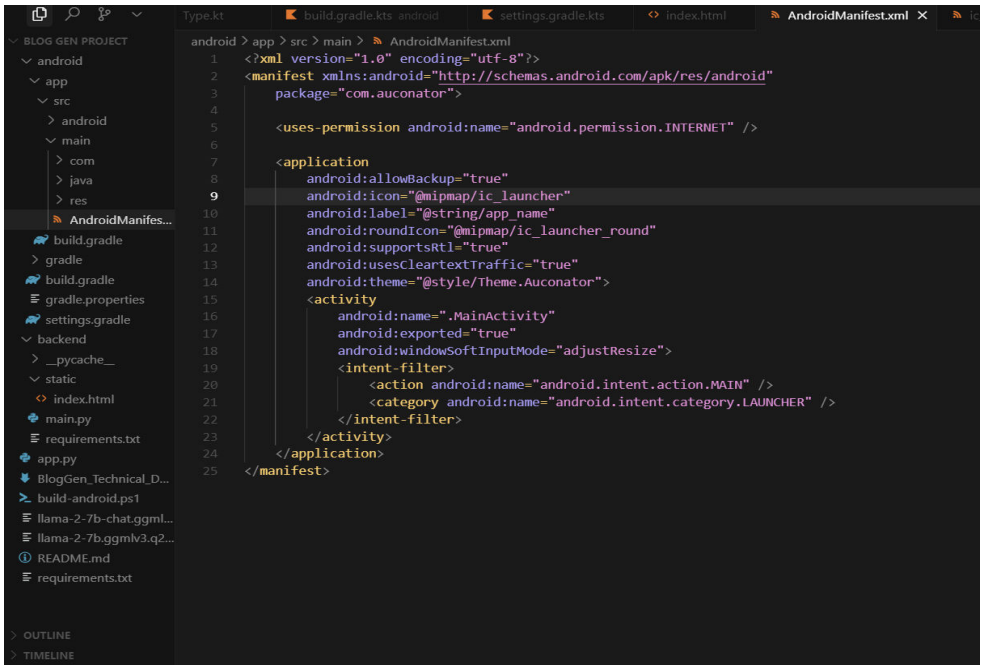
ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

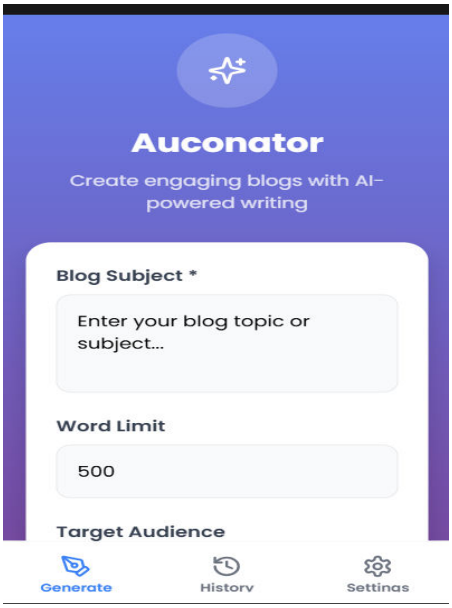
II. AUCONATOR

Code:



The Auconator system is built with a dual-interface approach: an Android mobile app frontend (as seen in the AndroidManifest.xml) and a web extension interface, both providing user-friendly content creation experiences. The backend is powered by FastAPI, a modern Python web framework, which handles the core business logic and AI integration. The system leverages Llama 2 for text generation through ONNX and PyTorch optimizations, ensuring efficient processing. The architecture includes specialized components for SEO optimization, content filtering, and plagiarism detection, making it a comprehensive solution for automated blog content generation.

Output:



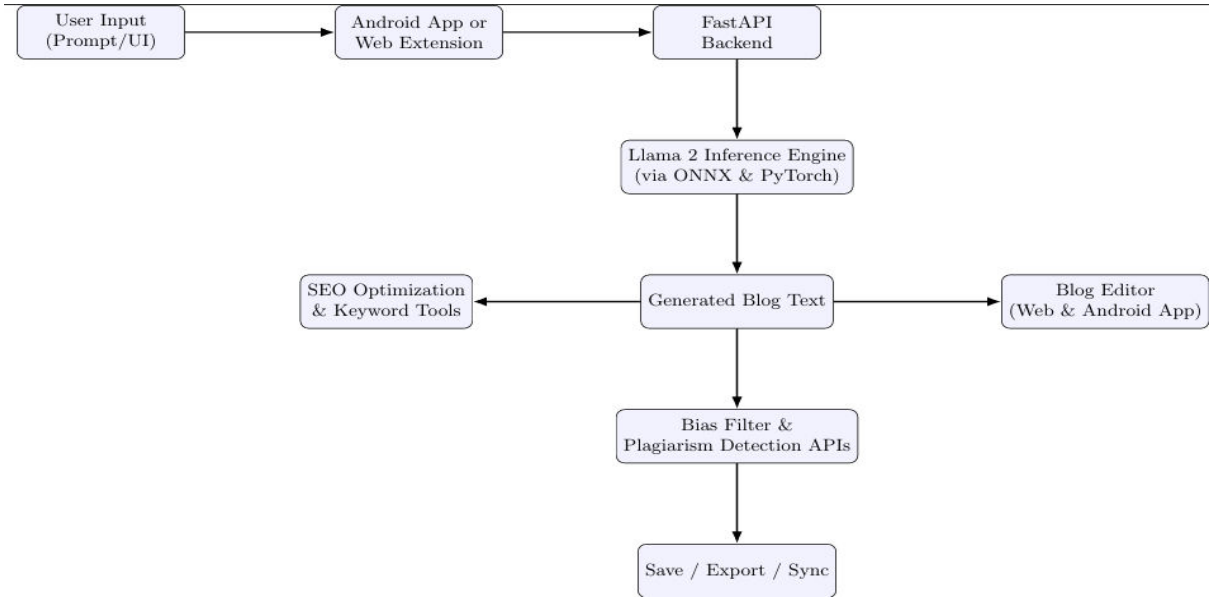
© 2025 IJMRSET | Volume 8, Issue 6, June 2025| DOI:10.15680/IJMRSET.2025.0806117

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

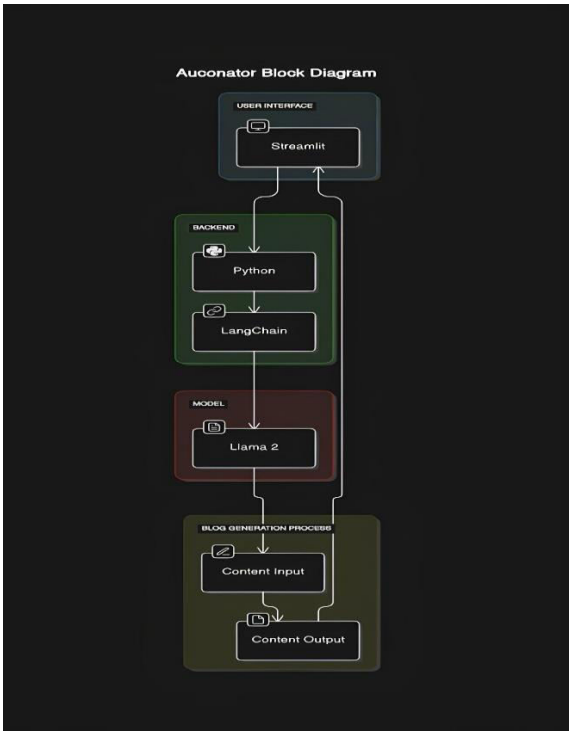
(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Architecture Diagram



A comprehensive blog generation tool that combines mobile and web interfaces with advanced AI capabilities. The flow starts with user input through either an Android app or web extension interface. This input is then processed by a FastAPI backend server, which acts as the intermediary layer. The core AI processing happens through a Llama 2 inference engine that's optimized using ONNX and PyTorch frameworks for efficient text generation.

Workflow And Block diagram



ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

1. User Input (Prompt/UI)

- The process begins when the user enters a blog topic, keywords, or content prompt through a simple and interactive UI.
- This interface is available via **Android App** or a **Web Extension**.

2. Android App or Web Extension

- The input is transmitted securely to the backend using API calls.
- This frontend is lightweight, focusing mainly on collecting prompt data and displaying the results to the user.

3. FastAPI Backend

- The backend is built using **FastAPI**, known for its performance and asynchronous processing.
- It handles incoming user requests, validates them, and forwards them to the inference engine.

4. LLaMA 2 Inference Engine (via ONNX & PyTorch)

- Here lies the core logic.
- The FastAPI backend passes the prompt to the **LLaMA 2 model**, deployed using **ONNX Runtime or PyTorch** for optimized inference.
- This module generates contextually rich and coherent blog content based on the prompt.

5. Generated Blog Text

- Once the LLaMA 2 model generates the content, it is passed to various supporting modules:

A. Blog Editor (Web & Android App)

- The generated text is sent to an integrated **rich-text blog editor** available in the Android and web apps.
- Users can manually edit, style, and finalize the blog here.

B. SEO Optimization & Keyword Tools

- The blog text is analyzed using **SEO APIs** or internal keyword tools.
- Suggestions on keyword density, readability, and search engine friendliness are provided.

C. Bias Filter & Plagiarism Detection APIs

- This step ensures **ethical** and **original** content.
- The blog is scanned for any bias, offensive language, or plagiarism using third-party or custom APIs.

6. Save / Export / Sync

- Once the content passes all checks, users can:
  - **Save** it locally or to cloud storage.
  - **Export** it as .txt, .docx, or .html.
  - **Sync** across devices (Android/Web) for continuity.

Key Components Breakdown

Component	Description
FastAPI Backend	Handles user requests and manages model inference.
LLaMA 2 (ONNX/PyTorch)	Performs the blog generation based on user input.
Editor Module	Allows user to fine-tune or publish the blog.
SEO Tools	Improve discoverability of the content.
Plagiarism Checker	Ensures authenticity and compliance.
Save/Export Function	Finalizes the blog for publication or storage.

Highlights & Improvements in Stage 2

- **Platform Expansion**: From web-only to cross-platform (Android + Browser Extension).
- **Faster Response Time**: Inference time improved to ~20 seconds.
- **User Control**: Rich-text editor allows more customization before final output.
- **Content Integrity**: Added layers for filtering bias and detecting plagiarism.

© 2025 IJMRSET | Volume 8, Issue 6, June 2025| DOI:10.15680/IJMRSET.2025.0806117

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

III. LITERATURE-REVIEW

Several tools and studies have explored AI in content creation. Language models like GPT-3 and Llama 2 have demonstrated the ability to produce contextually relevant and grammatically correct content. Lang Chain, a Python framework, has gained popularity for managing LLM pipelines, while platforms like Streamlit simplify interface design for NLP tools. Studies also indicate a rising trend in integrating NLP into mobile apps and browser plugins, providing seamless interaction with AI models for everyday tasks. Our research builds upon these advancements by combining AI models with practical UI elements to enhance the user experience in content automation.

IV. METHODOLOGY

1. Architecture Overview

The system architecture integrates the following components:

- **Frontend (Web Interface):** Built using Streamlit to accept user input (topic, word count, tone, etc.)
- **LLM Engine:** Llama 2 model hosted on Hugging Face is used for content generation.
- **Lang Chain:** Acts as the logic layer to structure the prompts and manage model interaction.
- **Backend Services:** API calls between UI and model through Python scripts and Hugging Face inference endpoints.
- **New Additions in Stage 2:**
  - **Android App:** Developed in Kotlin using Retrofit for API integration.
  - **Web Extension:** Built using JavaScript and Chrome Extension APIs.

2. Workflow

- User enters blog parameters.
- Lang Chain processes the request and formats it into prompts.
- Llama 2 generates the blog content.
- The generated content is displayed on Streamlit, mobile app, or extension interface.

Features and Functionalities

✓ Customizable Blog Generation

Users can define blog topic, desired length, and intended audience. The tool adapts output based on these parameters for context-relevant content.

✓ Real-Time AI Processing

Through Hugging Face's APIs, blog content is generated within seconds, reducing time spent on manual writing.

✓ Cross-Platform Usability

With the Stage 2 upgrade, the tool now supports:

- A **Kotlin-based Android application**
- A **Chrome-compatible web extension**

✓ User-Friendly Interfaces

- **Web:** Clean Streamlit layout with fields and live output.
- **App:** Mobile-friendly UI with input forms and share options.
- **Extension:** One-click access for content generation on any webpage.

System Architecture Diagram A layered model:

1. **UI Layer:** Streamlit UI / Android App / Web Extension
2. **Middleware:** LangChain pipeline
3. **LLM Layer:** Hugging Face-hosted Llama 2
4. **Deployment Layer:** Cloud server or local execution environment for integration and data handling

Architecture Overview

The architecture follows a modular and layered approach, enabling flexibility, cross-platform support, and easy scaling:

1. User Interface Layer

Streamlit Web App: Provides a simple and interactive web UI.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Android App: Built with Kotlin, offering mobile accessibility for content generation.
Chrome Extension: Lightweight browser tool that lets users generate content while browsing.

2. API Layer
A Flask (or Fast API) backend receives requests from the UI and acts as a bridge to the Lang Chain logic layer. It also handles user validation, logging, and asynchronous task execution if needed.

3. Lang Chain Logic Layer
LangChain formats the user-provided inputs (topic, word count, tone) into structured prompts. It manages the prompt-response flow with the LLM, allowing for customization and modular LLM interactions.

4. LLM Layer (Llama 2)
This is where the core AI processing occurs. Llama 2, hosted via Hugging Face, processes the prompt and returns a high-quality, human-like blog response. The Hugging Face inference API makes this integration seamless.

V. CONCLUSION

This research project explores how advanced language models can be effectively integrated into real-world tools to simplify the content creation process. The Stage 2 development significantly improves user accessibility by introducing a mobile app and browser extension. These additions make blog generation more accessible, intuitive, and convenient across multiple devices and contexts. By expanding the deployment scope, the tool caters to a wider audience, enabling faster, AI-assisted content production with minimal effort.

REFERENCES

1. Touvron, H., et al. "Llama 2: Open Foundation and Fine-Tuned Chat Models." Meta AI, 2023.
2. Hugging Face. “Transformers Documentation.” https://huggingface.co/docs
3. Lang Chain Documentation. https://docs.langchain.com
4. Streamlit Docs. https://docs.streamlit.io
5. Android Developers. “Kotlin Language Guide.” https://developer.android.com/kotlin
6. Chrome Developers. “Extension APIs.” https://developer.chrome.com/docs/extensions

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH

IN SCIENCE, ENGINEERING AND TECHNOLOGY

| Mobile No: +91-6381907438 | Whatsapp: +91-6381907438 | ijmrset@gmail.com |

www.ijmrset.com