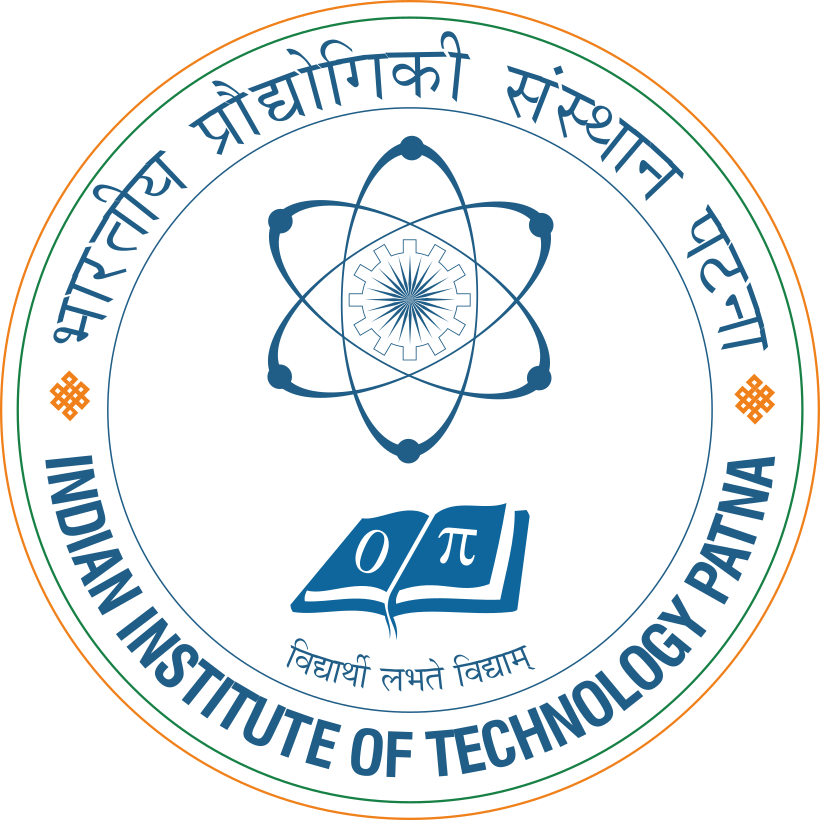
****

**Project Report for M. TECH 3rd Sem**

****

**Project Name:** **AI Multi-Agent Blog Generator & Publisher**

## **Student Information**

**Student Name:** Sushil Kumar  
**Roll Number:** 24A03RES180  
**Program:** M.Tech (AI & Data Science Engineering)  
**Institute:** Indian Institute of Technology (IIT) Patna

**Academic Year:** 2024–2025

## **Supervisor Information**

**Supervisor:** Rahul Kumar  
**Designation:** Lead Machine Learning Architect  
**Organization:** Thales India Pvt. Ltd.  
**Email:** rahul-l.kumar@thalesgroup.com  
**Mobile:** 7415489635

**1. Certificate**

This is to certify that the project titled **“AI Multi-Agent Blog Generator & Publisher”**submitted by **Sushil Kumar (Roll No. 24A03RES180)** is a bona fide record of original work completed under my supervision. This work has not been submitted elsewhere for any degree.

**Supervisor:**  
(Signature)  
**Rahul Kumar**  
Lead Machine Learning Architect, Thales India Pvt. Ltd.

**Head of Department:**  
(Signature)

**Date:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# ****2. Acknowledgement****

I would like to express my sincere gratitude to my supervisor **Mr. Rahul Kumar**, Lead Machine Learning Architect at Thales India Pvt. Ltd., for his constant guidance, encouragement, and technical insights throughout the duration of this project.

I also thank the faculty members of the **AI & Data Science Engineering Department, IIT Patna**, for providing a strong academic foundation and continuous support. I am grateful to my family and friends for their motivation and encouragement.

Finally, I acknowledge the contributions of the open-source community and developers whose tools, libraries, and frameworks made this project possible.

# ****3. Abstract****

This project presents an automated blog generation and publishing system built using a **multi-agent architecture** powered by **Generative Artificial Intelligence (AI)** and **Large Language Models (LLMs)**. The system automates the entire blog creation lifecycle, including topic discovery, research, content generation, image integration, and multi-platform publishing.

Each phase is handled by a specialized agent coordinated through a central orchestrator. The system integrates multiple LLM providers with an intelligent fallback mechanism to ensure reliability and performance. A **Streamlit-based interactive dashboard** enables users to control generation parameters and preview content before publishing.

The solution significantly reduces manual effort and time required for high-quality, SEO-optimized content creation and demonstrates a scalable approach to AI-driven content automation.

# ****4. Introduction****

Digital content creation plays a vital role in marketing, education, and information dissemination. Traditional blog writing requires extensive research, writing expertise, formatting, and SEO optimization, making it a time-consuming and skill-intensive process.

With advancements in **Generative AI and LLMs**, automated content generation has become increasingly feasible. This project aims to leverage these technologies by developing a **multi-agent system** that automates end-to-end blog generation and publishing while maintaining content quality and consistency.

# ****5. Objectives****

### **Primary Objectives**

* Design and implement an AI-powered blog generator using a multi-agent architecture
* Integrate multiple LLM providers with fallback support
* Generate well-structured, readable, and SEO-optimized blogs
* Develop an interactive Streamlit-based user interface
* Support publishing to Blogger, WordPress, and Medium
* Automate image sourcing and integration

### **Secondary Objectives**

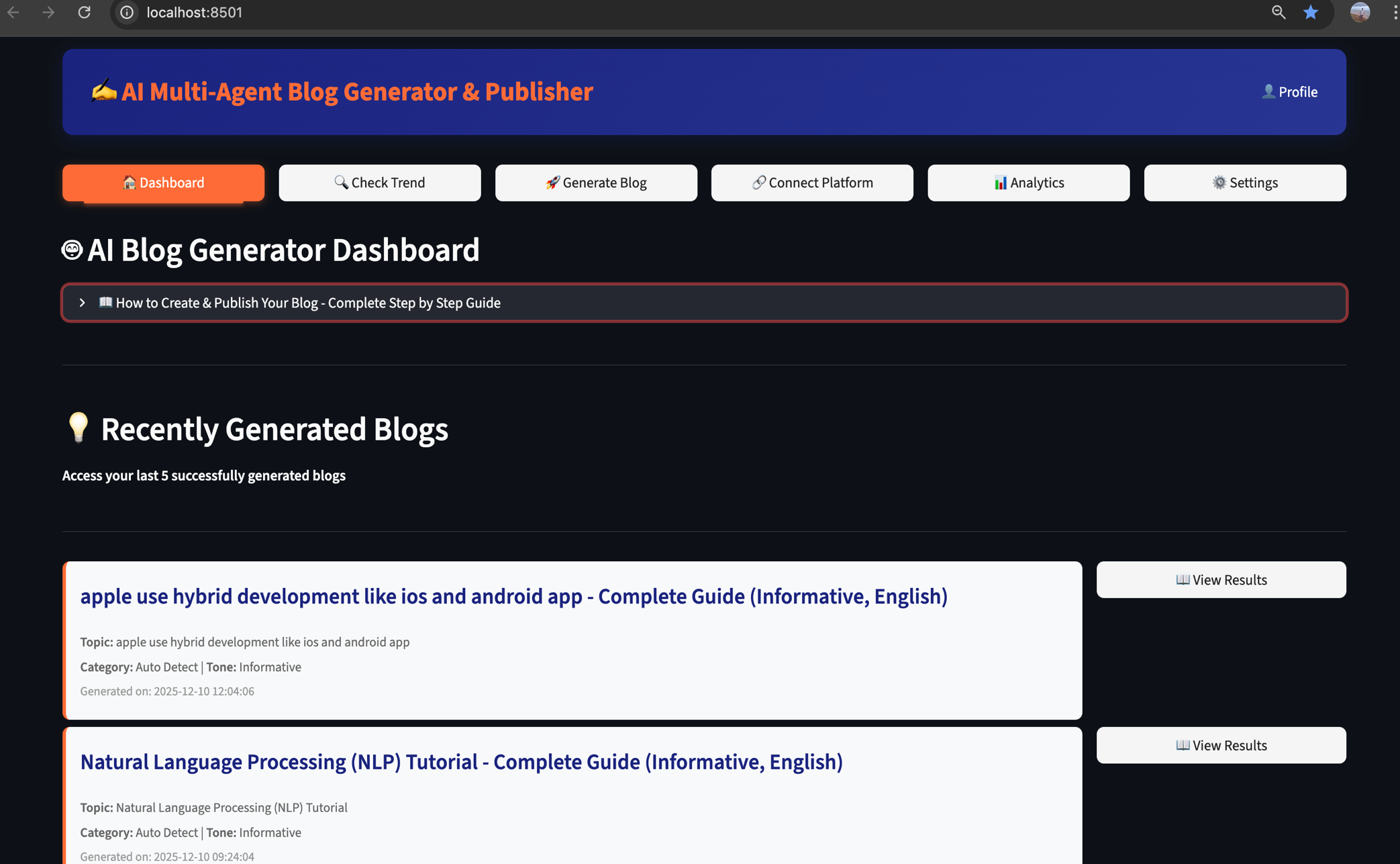
* Reduce content creation time
* Maintain consistency and quality across outputs
* Support multiple writing tones and languages
* Provide logging and analytics

# ****6. System Architecture****

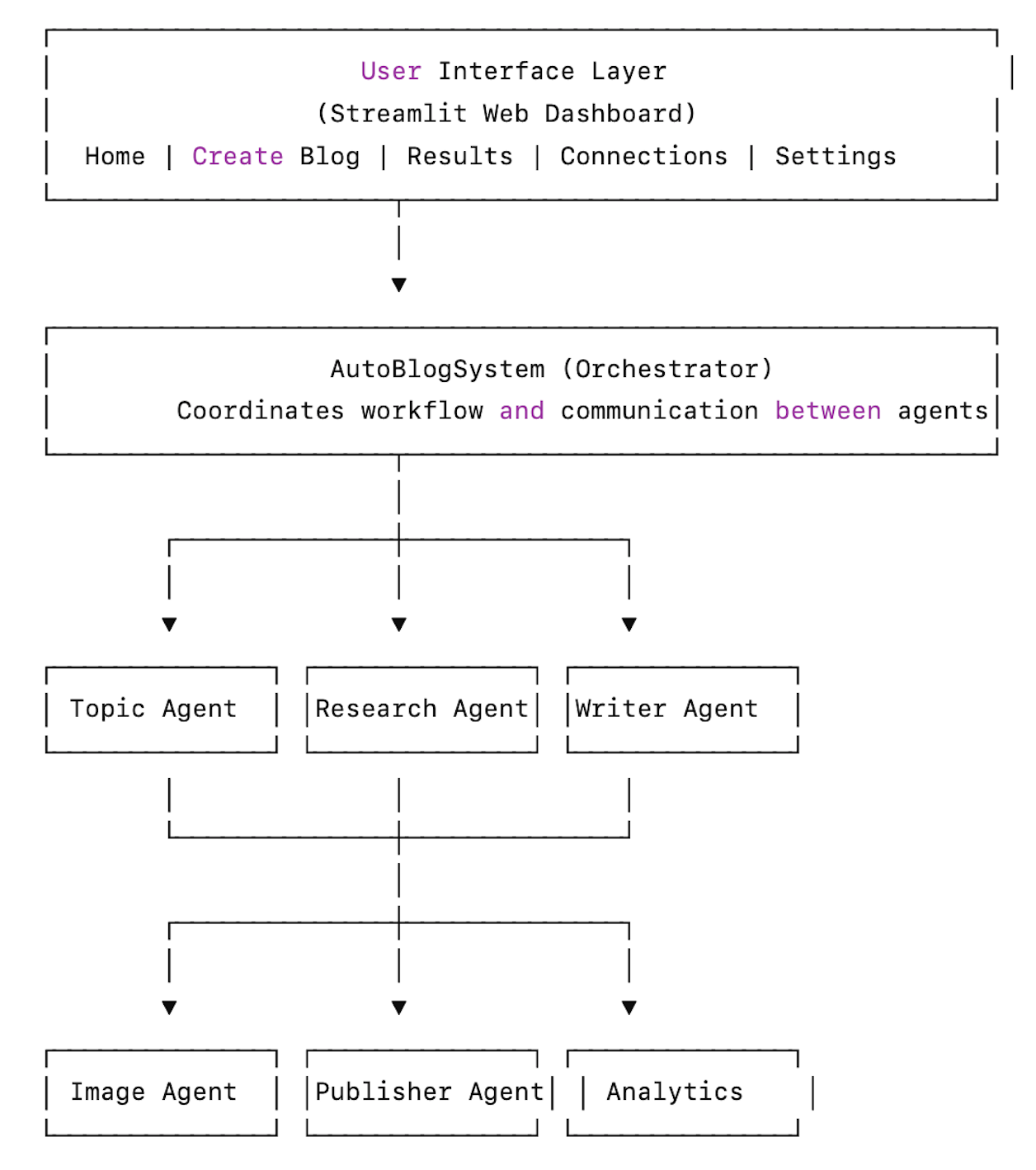
The system follows a **modular multi-agent architecture** where each agent performs a dedicated task. A central **orchestrator** coordinates the workflow and manages inter-agent communication.

### **Key Components**

* User Interface Layer (Streamlit)
* Orchestrator (Workflow Manager)
* Topic Agent
* Research Agent
* Content Writer Agent
* Image Agent
* Publisher Agent
* External APIs and LLM Providers



**System Architecture Diagram**



# ****7. User Interface Layer**** Architecture

### **1. User Interface Technology and Features Overview:**

**Technology:** Streamlit  
**Features:**

* Multi-page navigation (Dashboard, Check Trends, Generate Blog, Connections, Settings, Analytics)
* Real-time progress tracking
* Interactive content editing
* Preview functionality
* Publishing interface

### **2. AutoBlogSystem (Orchestrator):**

**Role:** Coordinates all agents

**Responsibilities:**

* Manages workflow execution
* Handles agent communication
* Error handling and recovery
* Result aggregation

### **3. Topic Agent:**

**Functionality:**

* Discovers trending topics using Wikipedia API
* Fetches related topics from NewsAPI
* Analyzes Google Trends data
* Suggests relevant blog topics

### **4. Research Agent:**

**Functionality:**

* Gathers information from multiple sources
* Uses LLM to analyze and synthesize research
* Creates structured research summaries
* Identifies key points and facts

### **5: Content Writer Agent**

**Functionality:**

* Generates blog outlines and structure
* Creates introduction, body sections, and conclusion
* Converts markdown to HTML
* Implements SEO optimization
* Supports multiple languages and tones

### **6.** **Image Agent** **(Under Development):**

**Functionality:**

* Sources images from Pexels and Pixabay APIs
* Generates thumbnail images (1200×628 px)
* Adds content images (800×400 px)
* Creates image placeholders with text overlays
* Optimizes images for web

### **7. Publisher Agent (Under Development):**

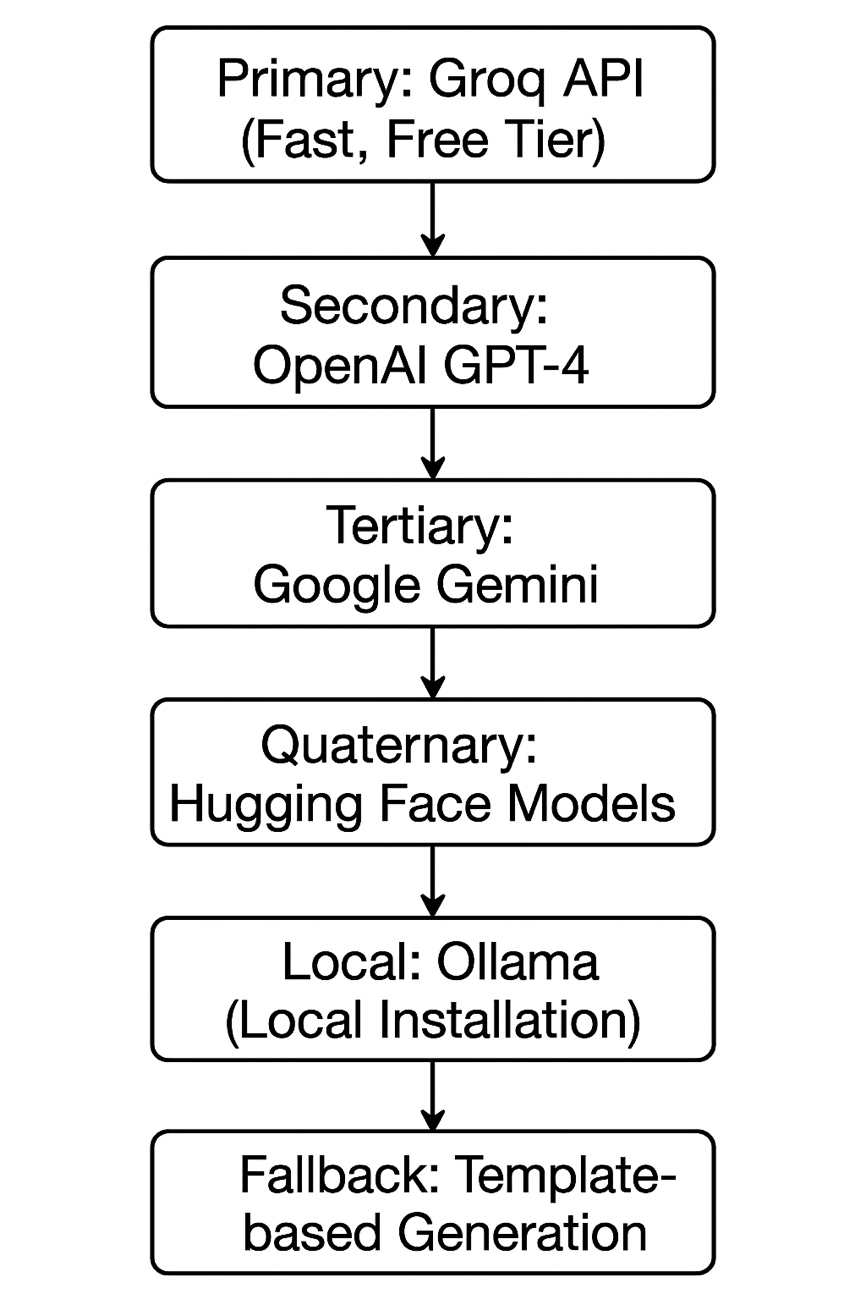
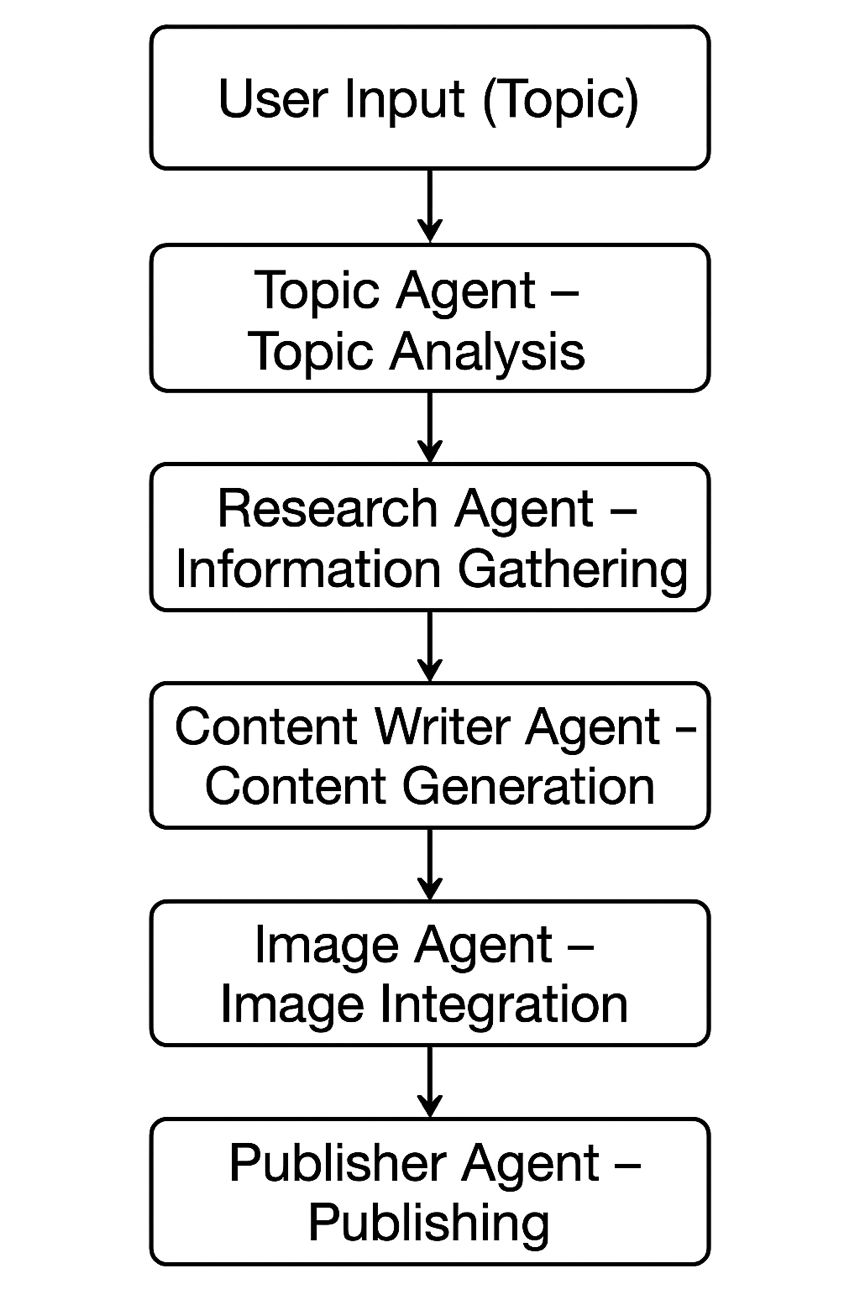
**Functionality:**

* Manages OAuth authentication
* Publishes to Blogger, WordPress, and Medium
* Handles token refresh
* Saves drafts and published posts
* Logs publishing history

# 

# ****8. User Data Flow & LLM Integration Architecture****

The system supports multiple LLM providers with automatic fallback:



# ****9**.** Methodology and System Output

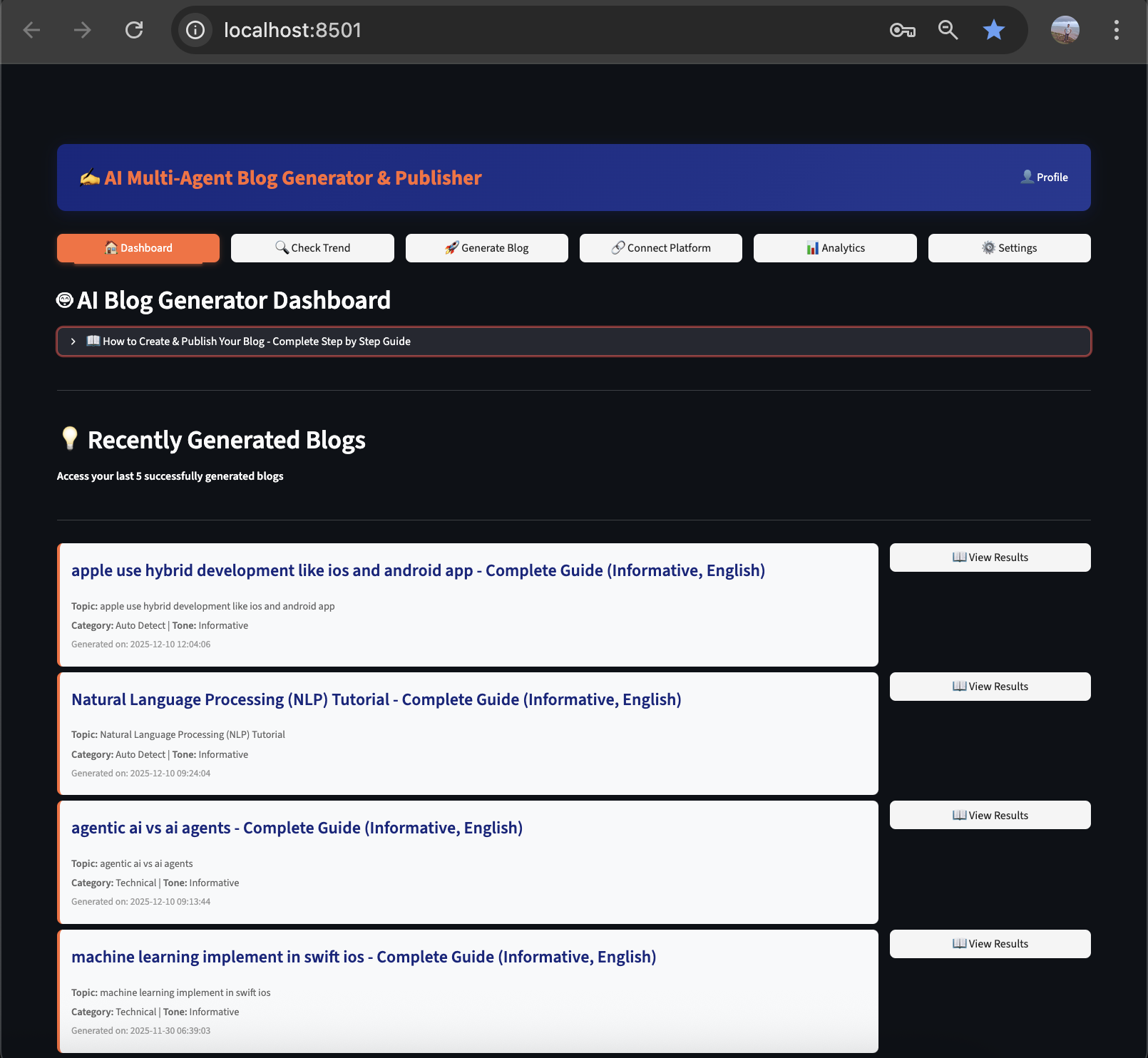
#### This section describes the working methodology of the **AI Multi-Agent Blog Generator & Publisher** along with screenshots demonstrating the actual system execution at each major stage.

#### **Step 1: User Input Interface:**

The system begins with user interaction through a **Streamlit-based web dashboard** (Dashboard Home Page screenshot).

The user provides:

* Blog topic or keyword
* Optional parameters such as topic recency, region, blog category, writing tone, and AI model selection
* Toggles for SEO optimization, image integration, and automatic tag generation



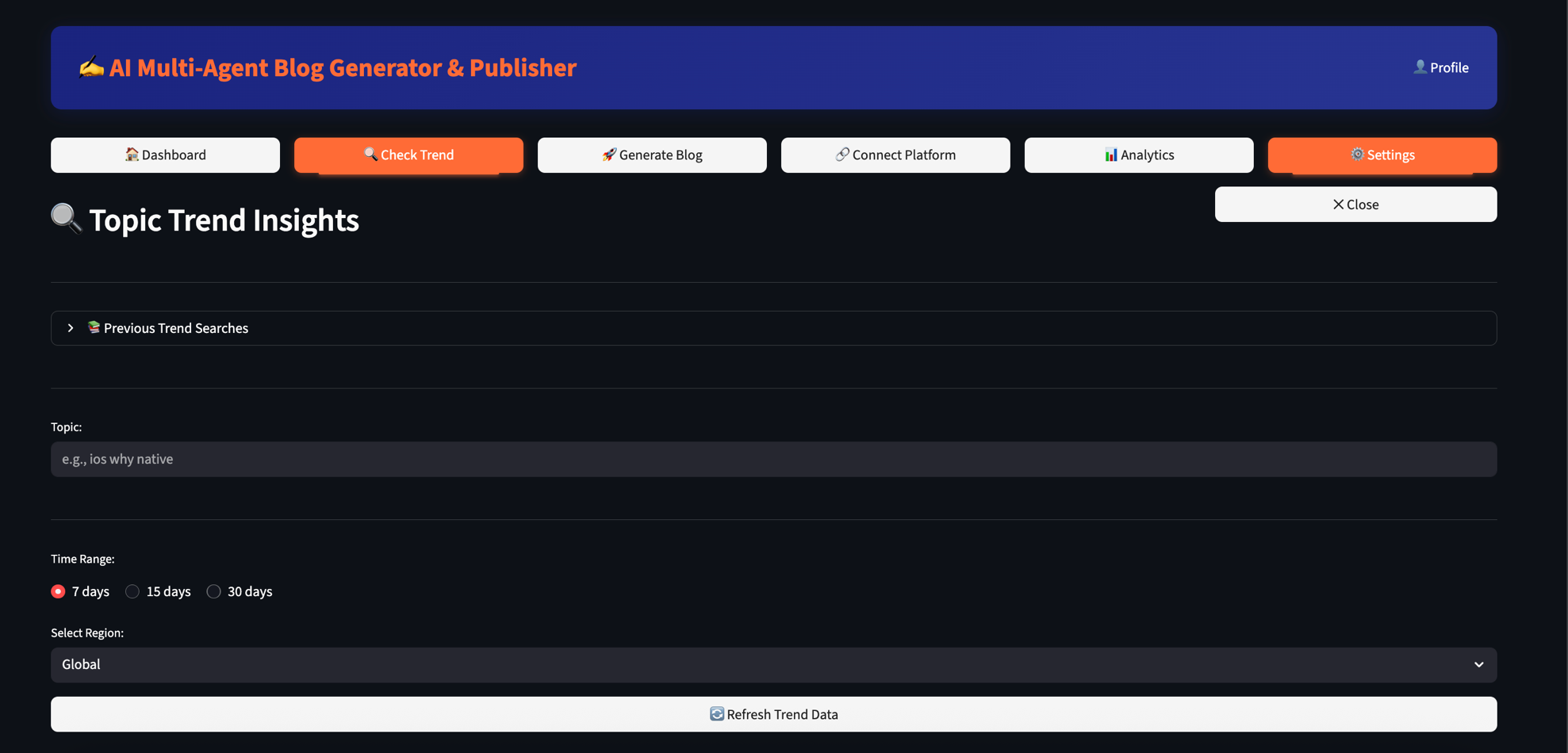


## **Step 2: Topic Analysis and Trend Insights**

After receiving the input, the **Topic Agent** analyzes the topic by:

* Fetching contextual data from Wikipedia, News sources, and Google Trends
* Identifying trending and related topics
* Generating relevant tags
* Estimating reading time

This process is visualized in the **Topic Trend Insights** screen, where users can select time range and region before proceeding.

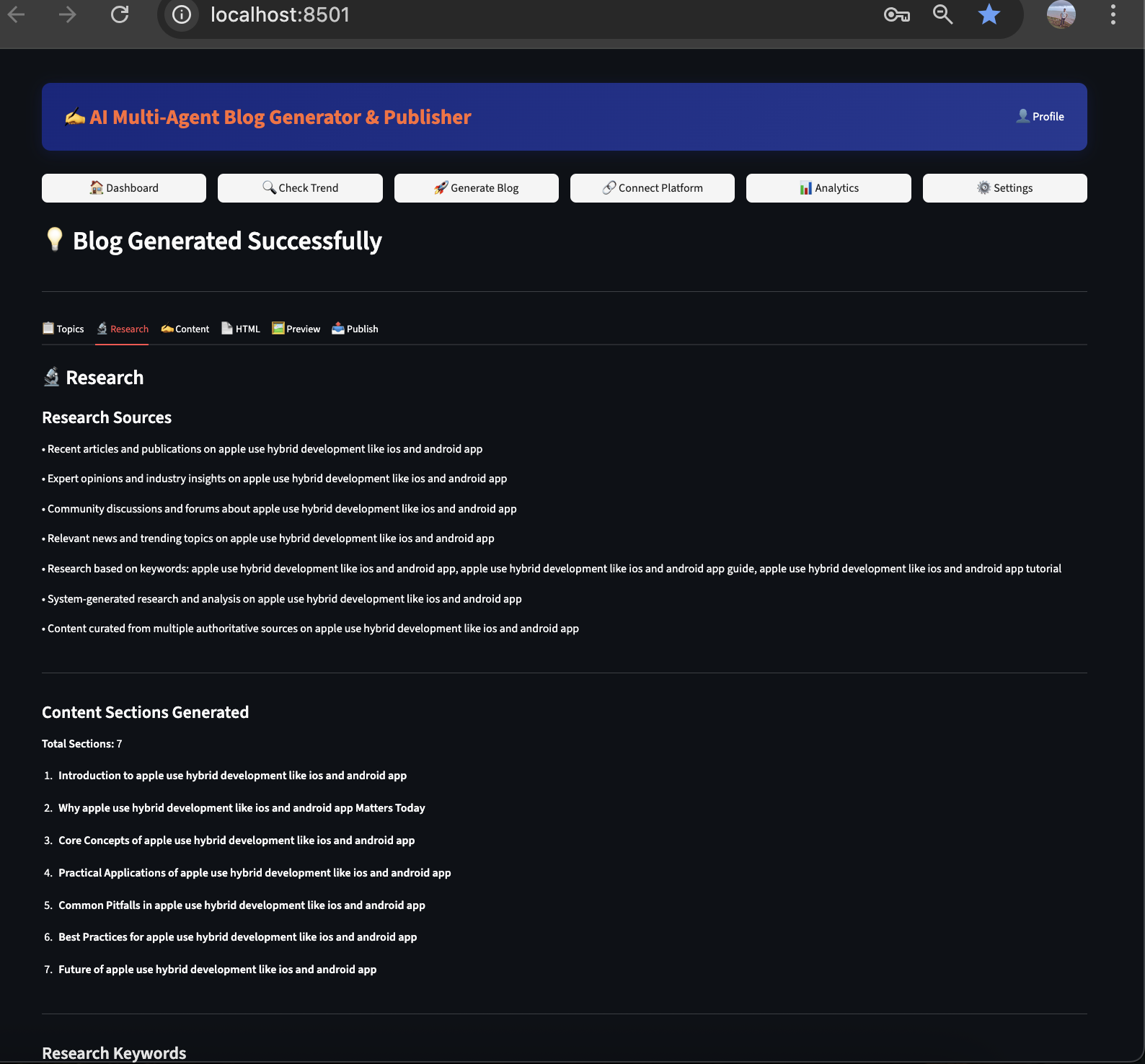


## **Step 3: Research Phase**

The **Research Agent** gathers and organizes information by:

* Using Large Language Models (LLMs) to analyze the topic
* Extracting key facts, concepts, and statistics
* Creating a structured research summary
* Identifying logical sections for content generation

This step ensures the generated blog is informative, accurate, and well-structured.

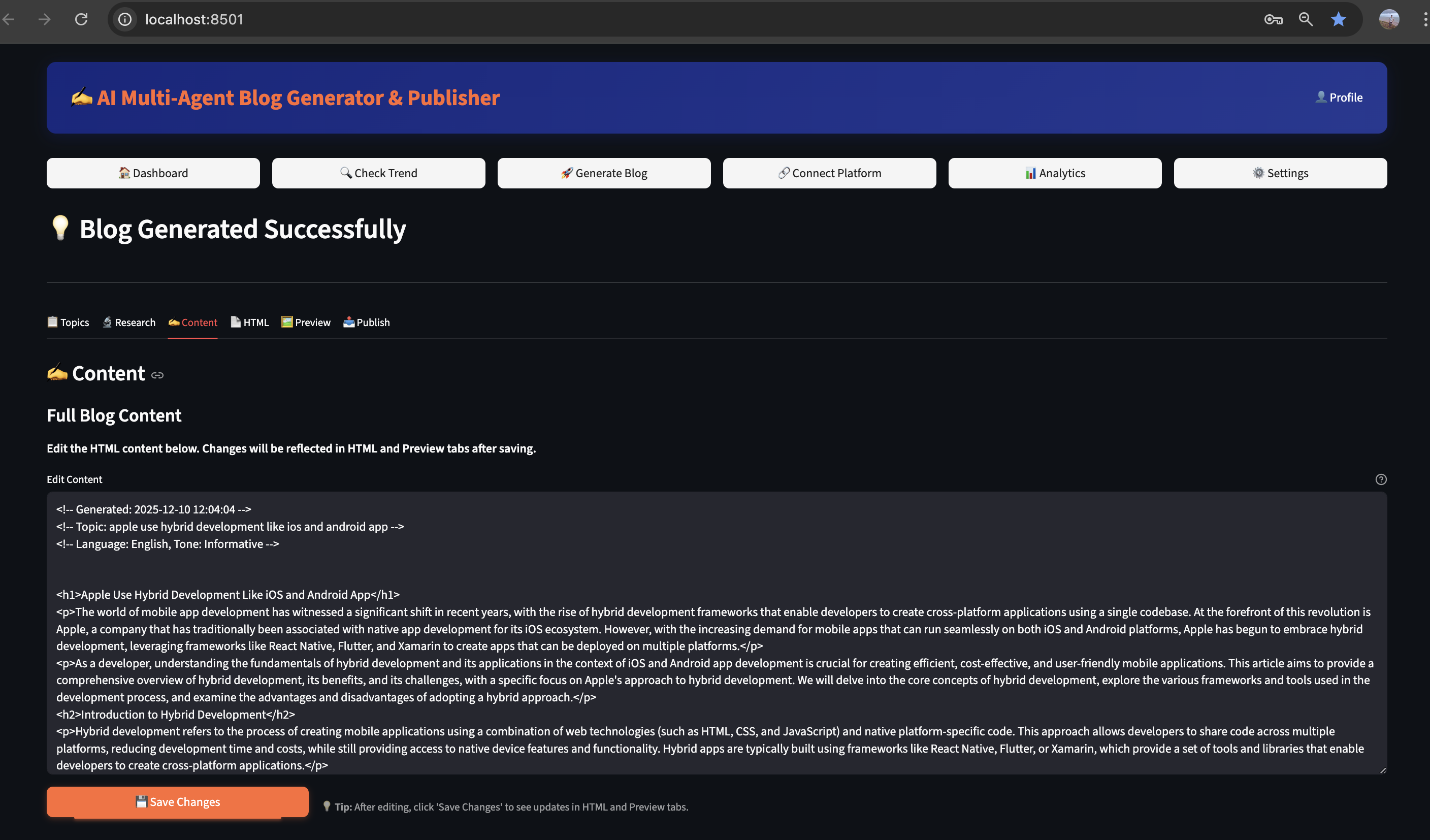


## **Step 4: Content Generation**

The **Content Writer Agent** generates the complete blog post by:

* Creating a modular outline
* Writing introduction, body sections, and conclusion
* Applying SEO optimization
* Supporting technical content with formatted code blocks
* Converting the content into HTML format

The generated content is displayed in the **Content View and HTML Preview** screens.



### **Step 5: Image Integration** **(Under Development)**

The **Image Agent** enhances the blog visually by:

* Fetching relevant images from Pexels and Pixabay APIs
* Generating blog thumbnails and inline images
* Optimizing images for web performance

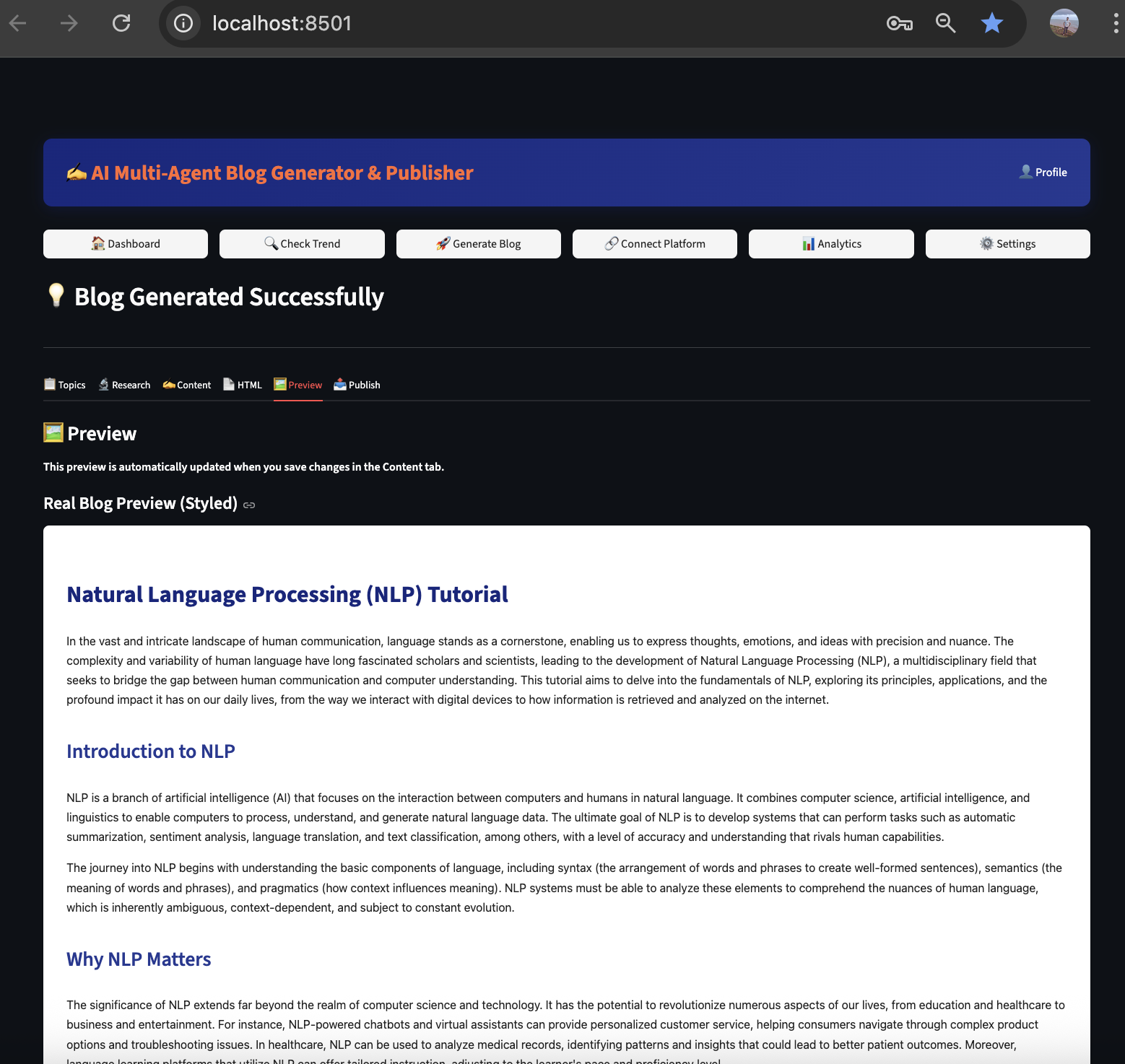
Images are automatically embedded into the generated blog content.

## **Step 6: Preview and Editing**

Before publishing, users can review the generated output through:

* Content preview
* HTML preview
* Inline editing options

This allows users to validate and refine the blog before final publishing.

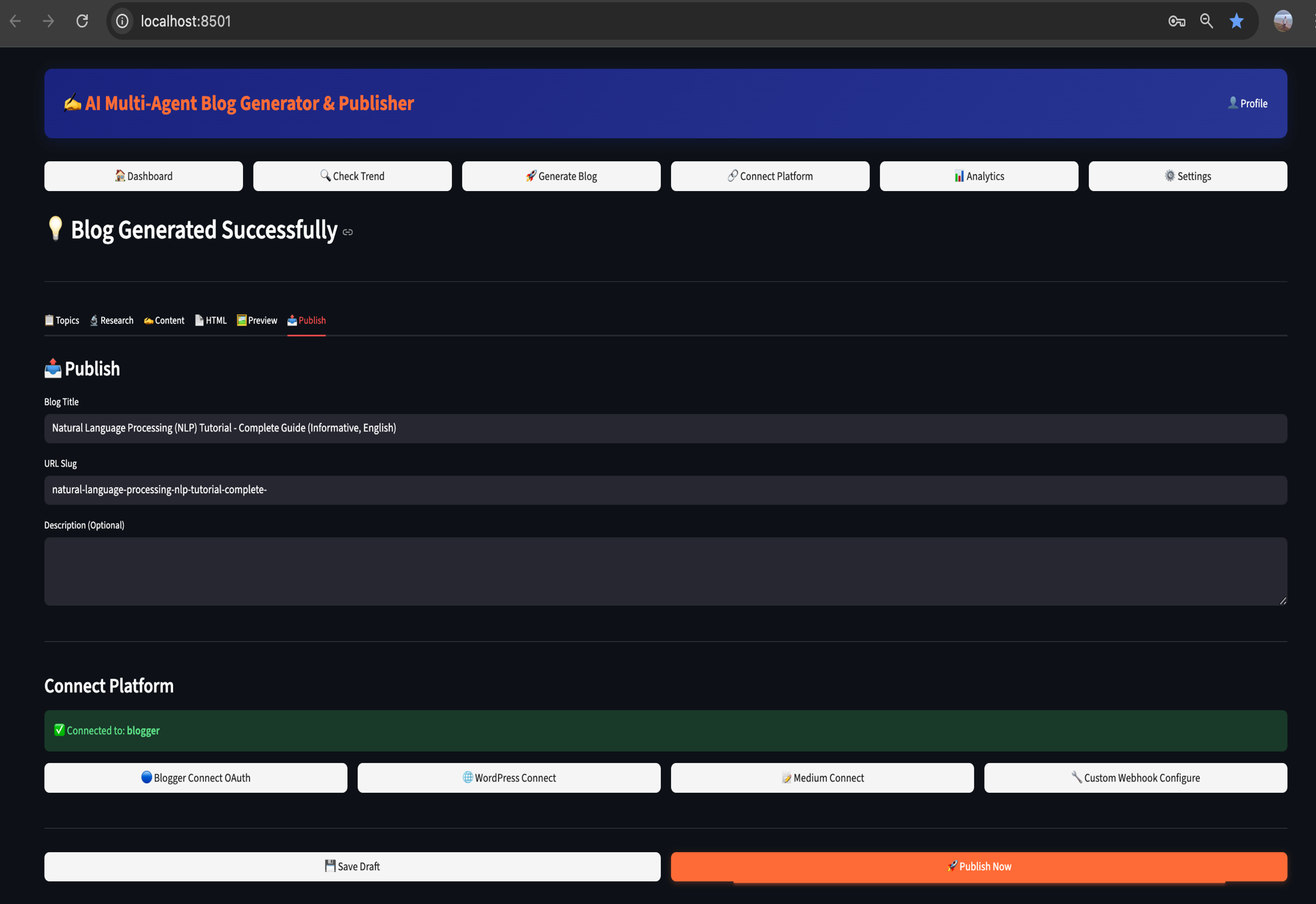


## **Step 7: Publishing**

The **Publisher Agent** manages the publishing process by:

* Allowing users to select platforms such as Blogger, WordPress, or Medium
* Handling OAuth-based authentication
* Supporting draft or live publishing
* Logging published blog URLs

After successful publishing, the post URL is returned to the user.

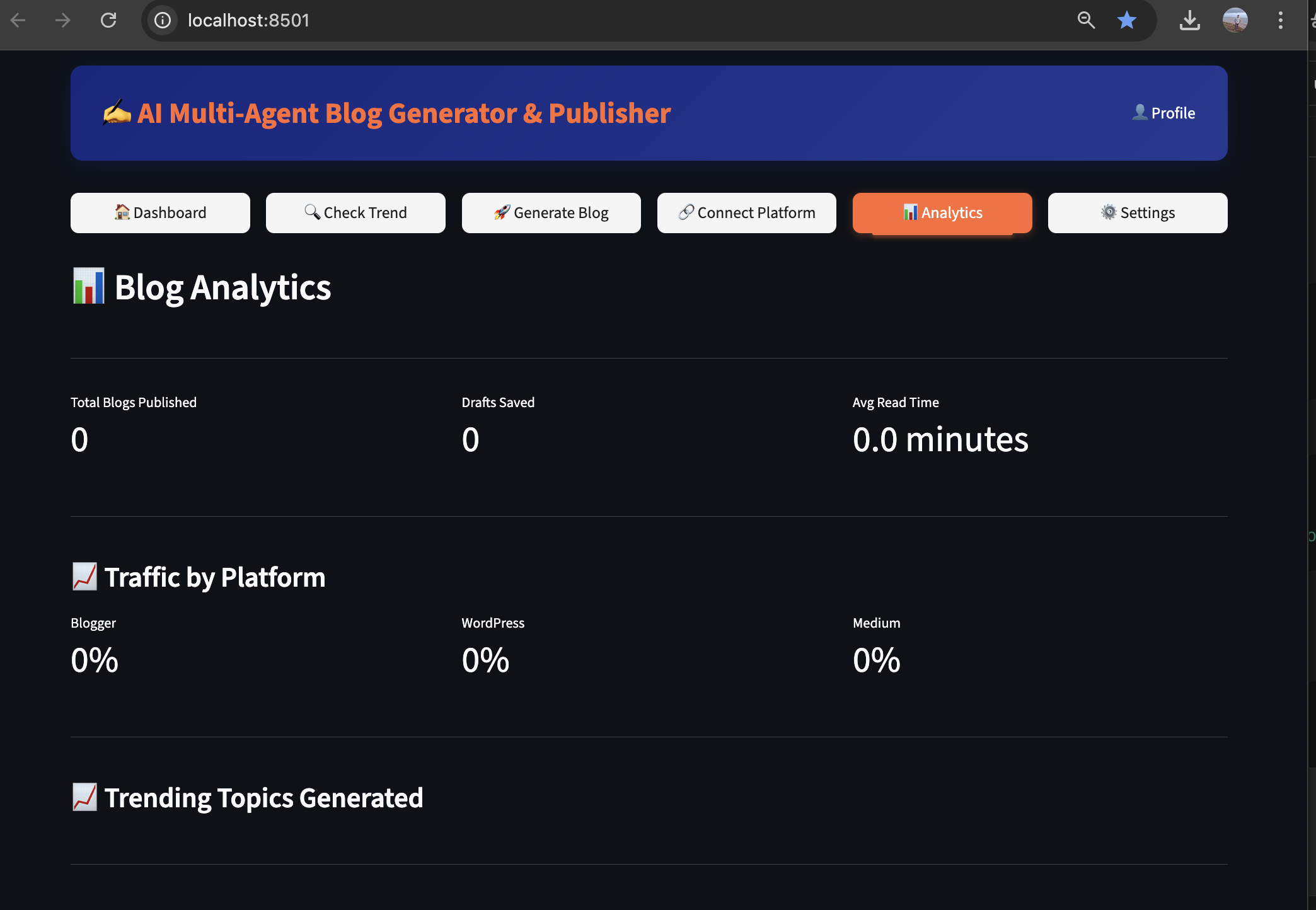


## **Step 8: Analytics and History Tracking (Under Development)**

The system records and displays:

* Generated blog history
* Published posts log
* Platform usage statistics
* Trending topics and basic performance metrics

These details are shown in the **Analytics Dashboard**.



# ****10. Technologies Used****

# ****Programming Language:**** Python 3.8+

# ****Frameworks and Libraries:****

* **Streamlit** – Web interface and dashboard
* **Requests** – API communication
* **Flask** – OAuth handling
* **Pytrends** – Google Trends integration

## **LLM Service Providers:**

* **Groq API** – High-speed inference using Llama/Mixtral models
* **OpenAI API** – GPT-4 family
* **Google Gemini API** – Multimodal generation
* **Hugging Face Inference API** – Open-source model access
* **Ollama** – Local LLM execution

## **External APIs:**

* **Wikipedia API** – Topic extraction
* **NewsAPI** – News-based topic enhancement
* **Pexels / Pixabay APIs** – Image sourcing
* **Blogger, WordPress, Medium APIs** – Blog publishing

**11. Implementation Project Structure**



# ****12. Advantages****

* Significant time savings in content creation
* Consistent and high-quality outputs
* Automatic SEO enhancement
* User-friendly interface
* Multi-platform publishing
* Multiple LLM options and fallback strategy
* Scalable and extensible architecture

# ****13. Limitations****

* Occasional factual inaccuracies
* Internet dependency for API operations
* API rate limits and token restrictions
* OAuth setup complexity
* Local LLMs require strong hardware

# ****14. Applications****

* Content marketing and SEO
* Educational tutorials and guides
* Business documentation
* Personal and professional blogging
* Technical writing

# ****15. Conclusion****

# The project successfully demonstrates the automation of blog generation using a multi-agent AI system. It reduces manual effort, improves consistency, and enables scalable content creation across platforms. The integration of LLMs, image automation, and publishing pipelines showcases the real-world applicability of AI in digital content creation.

# ****16.**** ****References****

**Documentation and Online Resources**

1. Streamlit Documentation. (n.d.). Streamlit. Retrieved from  
   <https://docs.streamlit.io>
2. OpenAI API Documentation. (n.d.). OpenAI Developer Platform. Retrieved from  
   <https://platform.openai.com/docs>
3. Groq API Documentation. (n.d.). Groq Developer Portal. Retrieved from  
   <https://console.groq.com/docs>
4. Google Gemini Documentation. (n.d.). Google AI Developer Resources. Retrieved from  
   <https://ai.google.dev/docs>
5. Hugging Face Documentation. (n.d.). Hugging Face Docs. Retrieved from  
   <https://huggingface.co/docs>
6. OAuth 2.0 Specification. (2012). RFC 6749. Retrieved from  
   <https://oauth.net/2/>
7. Google Blogger API Documentation. (n.d.). Google Developers. Retrieved from  
   <https://developers.google.com/blogger/docs/3.0>
8. WordPress REST API Reference. (n.d.). WordPress Developer Resources. Retrieved from  
   <https://developer.wordpress.org/rest-api>
9. Medium API Documentation. (n.d.). Medium Developer Docs. Retrieved from  
   <https://github.com/Medium/medium-api-docs>
10. Requests Library Documentation. (n.d.). Requests: HTTP for Humans. Retrieved from  
    <https://requests.readthedocs.io>
11. Pytrends Documentation. (n.d.). Pytrends: Google Trends API. Retrieved from  
    <https://github.com/GeneralMills/pytrends>
12. Flask Documentation. (n.d.). Flask Framework. Retrieved from  
    <https://flask.palletsprojects.com>

**AI Models and Frameworks**

1. Meta AI. (2024). LLaMA Models. Retrieved from  
   <https://ai.meta.com/llama>
2. Mistral AI. (2024). Mixtral Model Family. Retrieved from  
   <https://mistral.ai/news/mixtral-of-experts>
3. Ollama. (2024). Local LLM Runtime. Retrieved from  
   <https://ollama.ai>

**Image APIs**

1. Pexels API Documentation. (n.d.). Pexels Developer Resources. Retrieved from  
   <https://www.pexels.com/api>
2. Pixabay API Documentation. (n.d.). Pixabay API Docs. Retrieved from  
   <https://pixabay.com/api/docs>

**Date of Submission:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Signature:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_