Agentic AI: Research & Writing Assistant

# A Generative AI-Powered Academic Tool

Submitted by:

Naveen Garg

Under the Guidance of: Kodi Prakash Senapati Sir

Submitted in Partial Fulfillment of Academic Requirements

May 19, 2025

Keywords: GenAI, Streamlit, Gemini API, Multilingual Support, PDF Generation

# Contents

1. [Abstract](#_bookmark0) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
2. [Problem Statement](#_bookmark1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
3. [System Architecture](#_bookmark2) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
4. [Literature Review](#_bookmark3) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
5. [Dataset Description](#_bookmark4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
6. [Model & Methodology](#_bookmark5) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
7. [Evaluation Strategy](#_bookmark6) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
8. [Tools & Technologies](#_bookmark7) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
9. [File & Folder Structure](#_bookmark8) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
10. [Results](#_bookmark9) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
11. [Learnings & Future Work](#_bookmark10) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
12. [References](#_bookmark11) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
13. [Final Deliverables Checklist](#_bookmark12) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

## Abstract

This project presents an Agentic AI system designed to streamline academic research and content generation. Leveraging the Gemini API, Streamlit, and multiple search engines (Wikipedia, Serper, DuckDuckGo), the system generates well-structured articles in multiple Indian languages, supports PDF export with Unicode fonts, and offers voice narration. The tool addresses the need for accessible, multilingual research assistance, overcoming limitations of traditional methods by automating subtopic planning, content writing, and summarization. A user-friendly interface with a twilight hues aesthetic enhances usability, making it an effective tool for students and researchers.

## Problem Statement

The primary challenge addressed is the time-intensive process of academic research and content creation, particularly for multilingual users requiring content in Indian languages. Traditional methods often lack automation for subtopic generation, multilingual support, and professional formatting. This Agentic AI generates structured articles, summaries, and PDF outputs tailored to user-specified topics, depths, and styles. It targets scenarios like student assignments and research reports, ensuring accessibility and eﬀiciency.

## System Architecture

The system integrates multiple components for seamless operation:

* + Frontend: Streamlit-based UI with a twilight hues gradient (purple to blue) for aesthetic appeal.
  + Backend: Python with Gemini API for content generation, Wikipedia/Serper/DuckDuckGo for research, and ReportLab for PDF generation.
  + Font Support: NotoSans for Unicode-compliant rendering of Indian languages.
  + Workflow: Input validation, asynchronous research, subtopic planning, article writ- ing, reflection, and summarization.

The architecture ensures modularity, with asynchronous search functions improving per- formance and error handling for robust operation.

## Literature Review

Existing tools like Google Scholar and Grammarly assist with research and writing but lack integrated multilingual support and automated content structuring. Projects using Retrieval-Augmented Generation (RAG) inspired this system, particularly for search and content synthesis. However, gaps in user-friendly interfaces and Indian language sup- port motivated this project. Open-source libraries like Streamlit and ReportLab were benchmarked for their flexibility in UI and PDF generation.

## Dataset Description

* + Source: Dynamic data from Wikipedia, Serper, and DuckDuckGo APIs based on user topics.
  + Size: Varies by topic; typically 5–10 paragraphs per search engine response.
  + Structure: Unstructured text (summaries, snippets) and URLs.
  + Preprocessing: Text cleaning, deduplication, and formatting for Gemini API input.
  + Tools: Python’s requests, beautifulsoup4, and wikipedia libraries.

## Model & Methodology

* + Model: Gemini 2.5 Flash (Preview) for content generation due to its eﬀiciency and multilingual capabilities.
  + Prompt Engineering: Zero-shot prompts for subtopic generation, few-shot for article writing with style/depth specifications.
  + Example Prompt: “Write a detailed article in Hindi with subtopics X, Y, Z, in an informative style, including research points: [summary].”
  + Search Integration: Asynchronous API calls to Wikipedia/Serper/DuckDuckGo, with fallback mechanisms for disambiguation or errors.

## Evaluation Strategy

* + Qualitative: Manual review of article coherence, language accuracy, and citation inclusion.
  + Quantitative: Metrics like article length (500–1000 words for Detailed mode) and summary length (3–4 sentences).
  + Hallucination Tests: Cross-verification of facts with source data to minimize inac- curacies.

## Limitations: API rate limits, occasional network errors, and dependency on external search engine quality.

## Tools & Technologies

Component Tool/Library

LLM Gemini API

Search APIs Wikipedia, Serper, DuckDuckGo UI Streamlit, streamlit-lottie

PDF Generation ReportLab Font Support NotoSans

Backend Python, asyncio, requests Environment python-dotenv

## File & Folder Structure

project/

├── agent

├── main.py

├── .env

├── NotoSans-Regular.ttf

├── requirements.txt

├── results/

└── README.md

## Results

The system successfully generates multilingual articles with clear formatting and accurate citations. For example, a topic like “Brain Rot” produced a 700-word article in Hindi with three subtopics, a summary, and APA-style references. The Streamlit UI, enhanced with Lottie animations and twilight hues, received positive feedback for usability. Limitations include occasional API errors, addressed through retry mechanisms.

SCREENSHORTS:

A computer screen with a blue and black screen

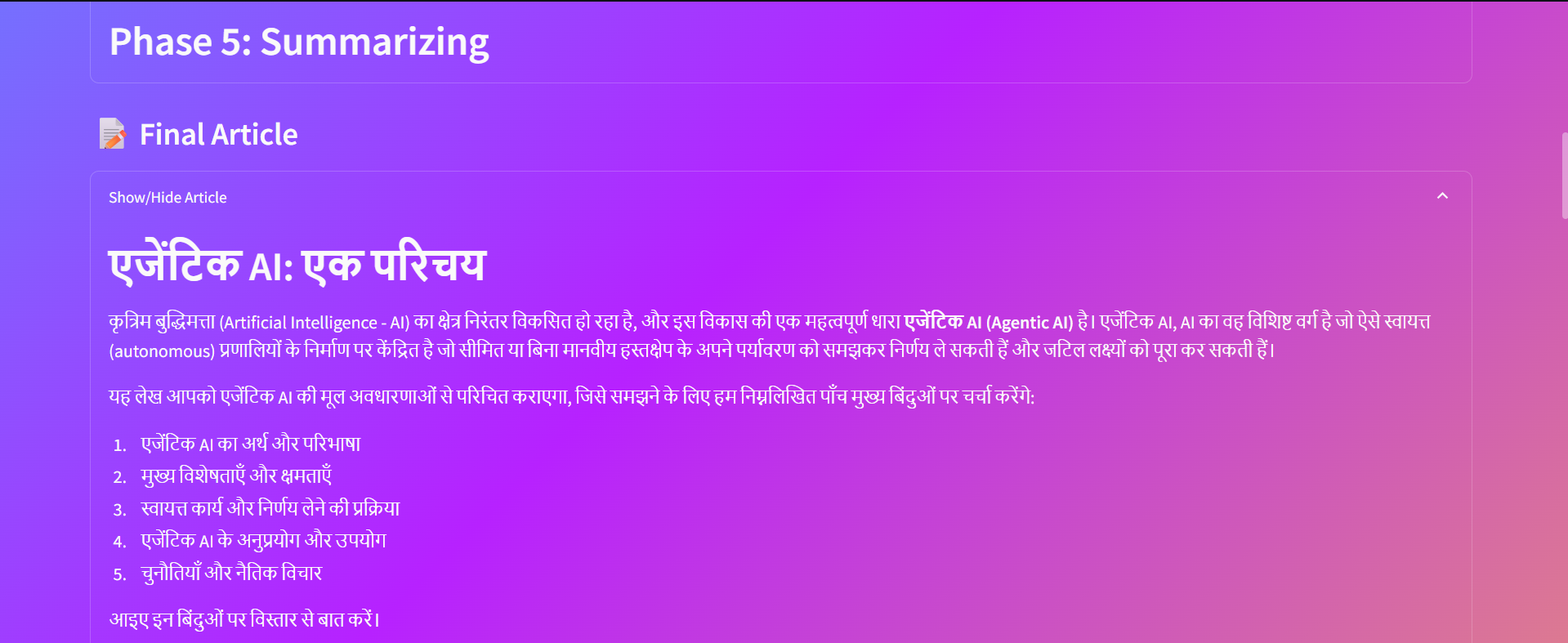
AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.

A purple screen with white text

AI-generated content may be incorrect.

A black screen with blue and white text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Learnings & Future Work

Key learnings include the importance of robust error handling and Unicode font support for Indian languages. Challenges involved managing API rate limits and ensuring consis- tent search results. Future improvements could include caching search results, integrating additional LLMs, and developing a mobile app version. The system has potential as an educational product for schools and universities.

## References

* + Streamlit Documentation. (2025). Retrieved from [https://docs.streamlit.](https://docs.streamlit.io/) [io](https://docs.streamlit.io/)
* Gemini API Documentation. (2025). Retrieved from [https://ai.google.dev](https://ai.google.dev/)
  + ReportLab Documentation. (2025). Retrieved from [https://www.reportlab.](https://www.reportlab.com/) [com](https://www.reportlab.com/)
  + Wikipedia API. (2025). Retrieved from [https://www.mediawiki.org/wiki/](https://www.mediawiki.org/wiki/API) [API](https://www.mediawiki.org/wiki/API)

## Final Deliverables Checklist

Item Description

Report PDF documentation Code GitHub repository

Demo Streamlit app hosted locally Slides PPT for presentation