Marketing Content Generator Report

Course Name: Generative AI

Institution Name: Medicaps University – Datagami Skill Based Course

Student Name(s) & Enrolment Number(s):

Sr no	Student Name	Enrolment Number
1	Aditi Gaikwad	EN22CS301040
2	Akhilesh Tiwari	EN22CS301085
3	Alfez Khan	EN22CS301102
4	Akshat Sakalye	EN22CS301093
5	Aditya Patidar	EN22CS301060
6	Aanchal Chaturvedi	EN22CS301014

Group Name: Group 06D1

Project Number: GAI-06

Industry Mentor Name: Aashruti Shah (Program Director, Datagami)

University Mentor Name: Prof. Ajaj Khan

Academic Year: 2025-2026

1 - Problem Statement & Objectives

1.1 - Problem Statement

In the fast-paced digital marketing landscape, professionals spend a disproportionate amount of time brainstorming, drafting, and refining content across various platforms. Maintaining a consistent brand tone while adapting to the specific nuances of different mediums (e.g., LinkedIn vs. Instagram vs. Email) is a manual, labor-intensive process. There is a critical need for a specialized, locally-hosted Generative AI application that allows users to rapidly generate, iterate, and regenerate high-quality marketing copy tailored to specific topics, target platforms, and desired emotional tones.

1.2 - Project Objectives

- **Rapid Content Iteration:** To build a web application that significantly reduces the time from ideation to first draft for marketing professionals.
- **Platform-Specific Tailoring:** To ensure generated content is structurally and contextually optimized for the user's selected platform (e.g., character limits, hashtag usage).
- **User Personalization:** To implement an authentication system linked to a Vector Database, creating a personalized environment that can learn or store user-specific contextual preferences over time.
- **Low-Latency Generation:** To integrate the gemini-flash model via API to ensure near-instantaneous text generation and regeneration capabilities.

1.3 - Scope of the Project

The current scope of the project encompasses a full-stack web application deployed on a local server environment. The scope includes:

1. **Authentication Module:** A landing page requiring user Sign-Up and Login to establish secure, personalized session environments.

2. **Input Interface:** A clean dashboard featuring a form with three primary data points: Topic (the subject matter), Platform (e.g., Twitter, Facebook, Blog), and Tone (e.g., Professional, Humorous, Urgent).

3. **AI Integration & Output:** Backend integration with the gemini-flash LLM API to process the inputs through prompt engineering, displaying the resulting content directly beneath the form.

4. **Regeneration Capability:** A dynamic state management system that allows users to click the generate button multiple times to yield completely new variations of the content without refreshing the page.

2 - Proposed Solution

We propose a locally hosted, decoupled web architecture utilizing a React frontend and a Python FastAPI backend.

When a user accesses the local server via their browser, they are greeted with an authentication gateway. Upon logging in, the FastAPI backend establishes a session, which is linked to a Vector Database designed to store user-specific embeddings (such as past successful tones or frequently used industry jargon) to create a personalized environment.

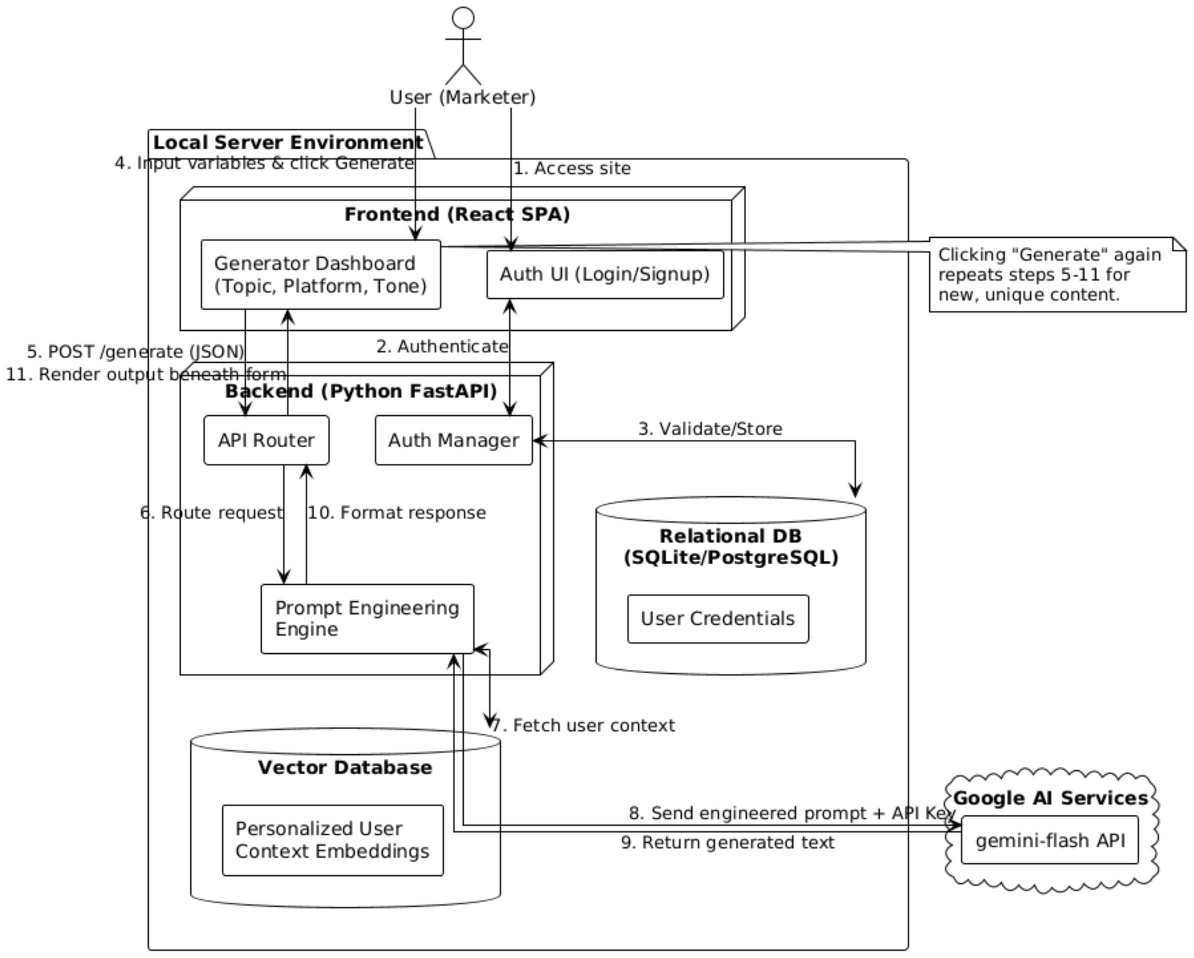
In the main workspace, the user inputs their topic, platform, and tone. The React frontend captures this state and sends an asynchronous payload to the FastAPI API. The backend constructs a highly engineered prompt—combining the user's explicit inputs with contextual data retrieved from the Vector DB—and dispatches it to the gemini-flash API. The LLM processes the request and returns the generated marketing copy, which the FastAPI backend sanitizes and passes back to the React UI for immediate rendering. The system is designed to handle repeated requests efficiently, generating fresh outputs on every button click.

2.1 - Key features

- **Secure User Authentication:** Dedicated Sign-up and Login workflows to protect user data and maintain session integrity.

- **Personalized Vector Environment:** Integration of a Vector DB to maintain a personalized context window for each authenticated user, improving the relevance of generated outputs over time.
- **Tri-Variable Input System:** Streamlined user experience focusing on the three most critical marketing variables: Topic, Platform, and Tone.
- **Lightning-Fast Generation:** Utilization of the gemini-flash model, ensuring the application remains highly responsive during the generation phase.
- **On-Demand Regeneration:** Built-in capability to request new, unique variations of the content instantly by re-triggering the generation sequence.

2.2 - Overall Architecture / Workflow



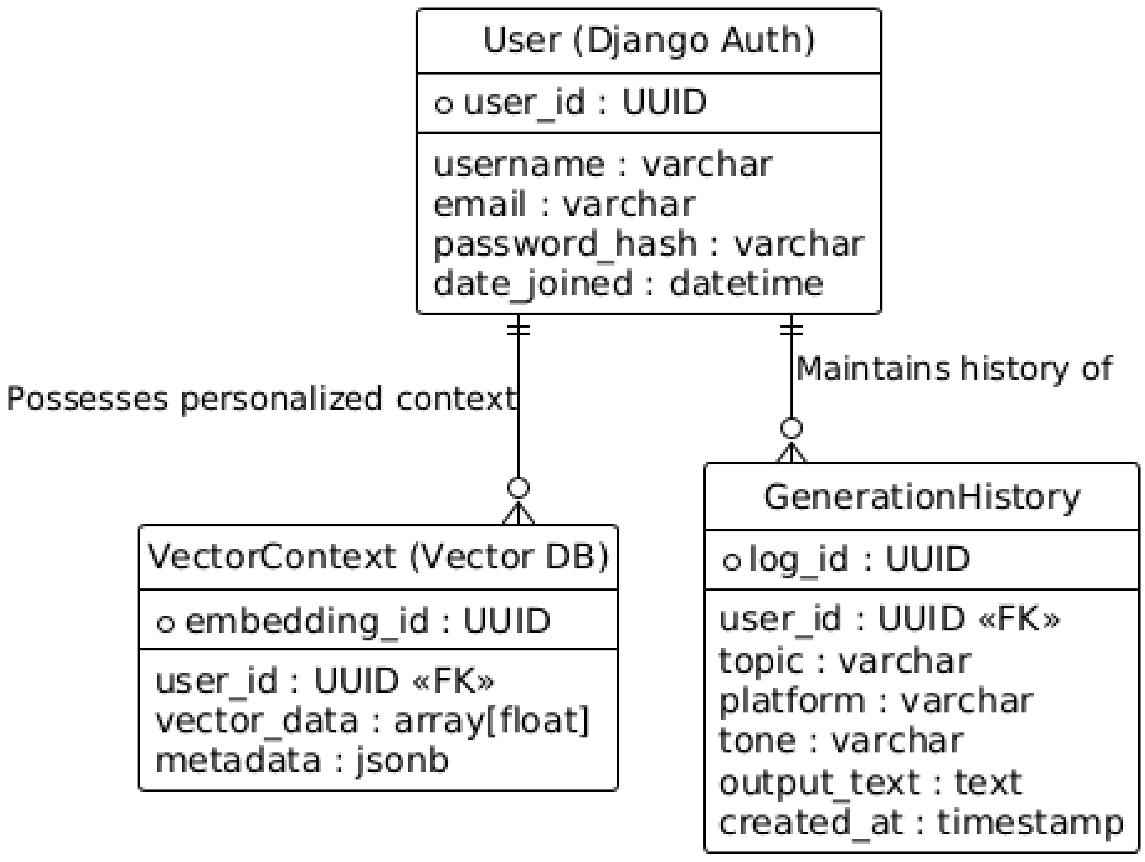
- **Client Tier:** The user's web browser running the React application. It handles routing between the login screen and the main generator dashboard.

- **Application Tier (Local Server):** The Python FastAPI backend. It exposes RESTful API endpoints for authentication and content generation. It acts as the orchestrator, taking the basic UI inputs and wrapping them in advanced prompt engineering logic.
- **Data & AI Tier:** Comprises the local relational database (for user credentials), the Vector DB (for personalized context embeddings), and the external gemini-flash API (for the actual generative heavy lifting).

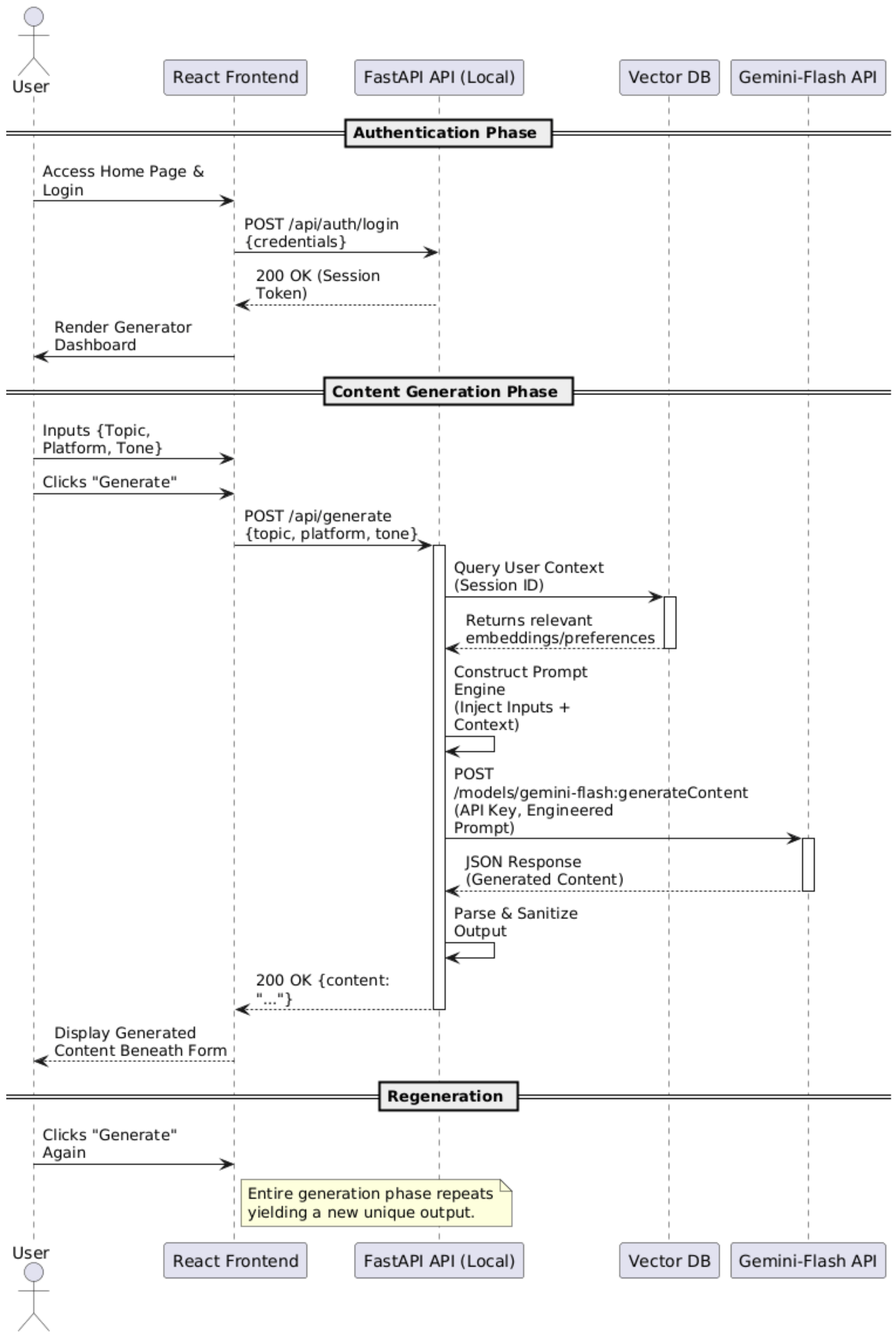
2.3 - Low-Level Design (LLD)/Workflow

The Low-Level Design delves into the specific interactions between the system's components, detailing the exact sequence of operations from user input to final content delivery, as well as the database structure that supports user personalization.

2.3.1 - Entity-Relationship (ER) Diagram: To support the personalized environment, the local server must maintain a relationship between the authenticated user, their generation history, and their personalized embeddings stored in the Vector DB.



2.3.2 - Sequence Diagram: This diagram illustrates the step-by-step execution flow when a user interacts with the application, highlighting the seamless integration with the gemini-flash API and the local Vector DB.



User
React Frontend
FastAPI API (Local)
Vector DB
Gemini-Flash API
Authentication Phase
Access Home Page & Login
POST /api/auth/login {credentials}
200 OK (Session Token)
Render Generator Dashboard
Content Generation Phase
Inputs {Topic, Platform, Tone}
Clicks "Generate"
POST /api/generate {topic, platform, tone}
Query User Context (Session ID)
Returns relevant embeddings/preferences
Construct Prompt Engine (Inject Inputs + Context)
POST /models/gemini-flash:generateContent (API Key, Engineered Prompt)
JSON Response (Generated Content)
Parse & Sanitize Output
200 OK {content: "..."}
Display Generated Content Beneath Form
Regeneration
Clicks "Generate" Again
Entire generation phase repeats yielding a new unique output.

2.4 - Tools & Technologies Used:

The Marketing Content Generator is built upon a modern, decoupled technology stack running on a local development server:

- **Frontend Interface:** React.js, HTML5, CSS3 (used for building the dynamic, single-page application and capturing the topic, platform, and tone inputs).
- **Backend Server:** Python, FastAPI, FastAPI REST Framework (handles user authentication, API routing, and prompt construction).
- **Large Language Model:** Google gemini-flash model accessed via API key implementation (chosen for its exceptionally low latency and high-quality generative capabilities).
- **Vector Database:** ChromaDB or Pinecone (utilized locally to store and retrieve contextual embeddings for user personalization).
- **Local Database:** SQLite or PostgreSQL (for managing standard relational data like user credentials and session states).

3 - Results & Output

3.1 - Screenshots / outputs:

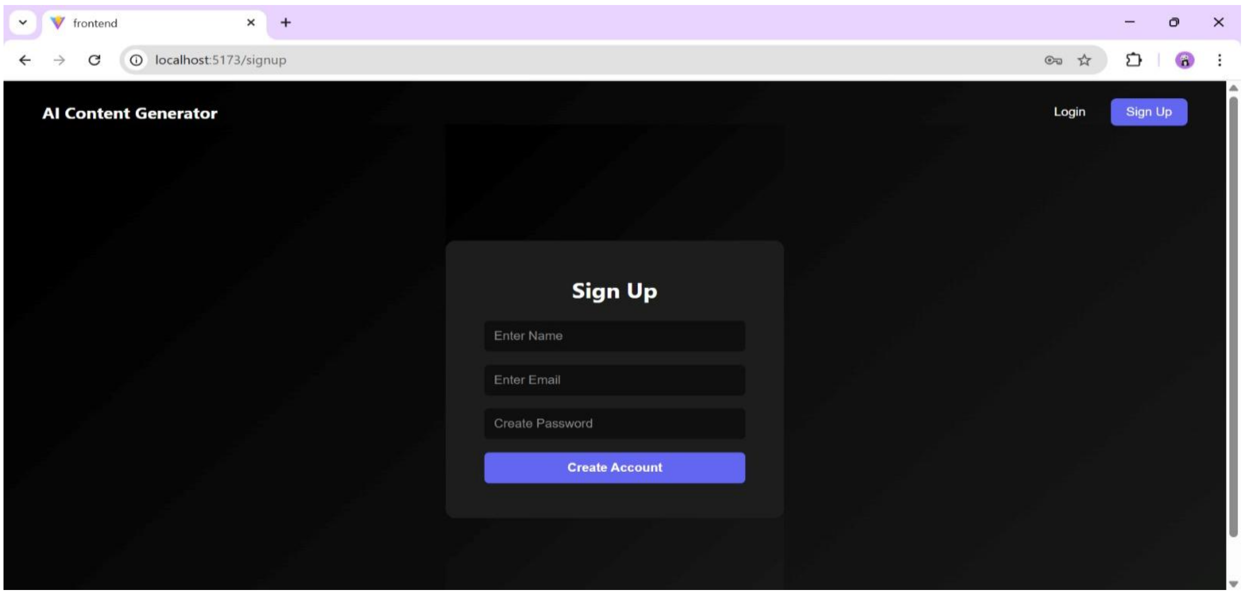


Fig. Screenshot of Sign Up Page

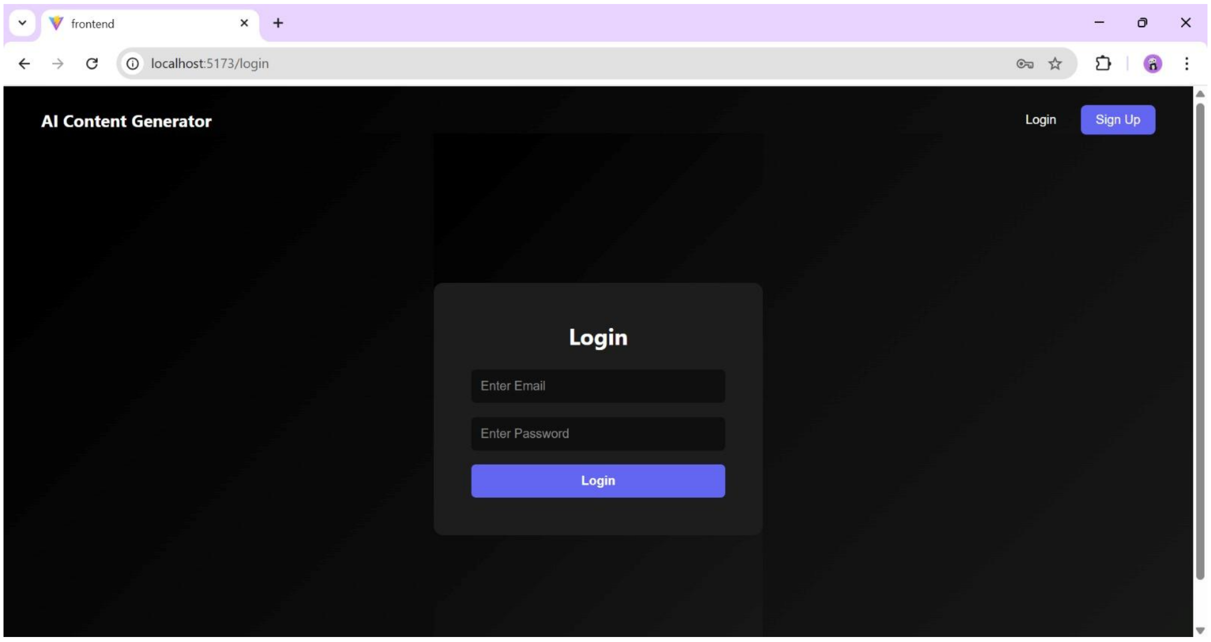


Fig. Screenshot of Login Page

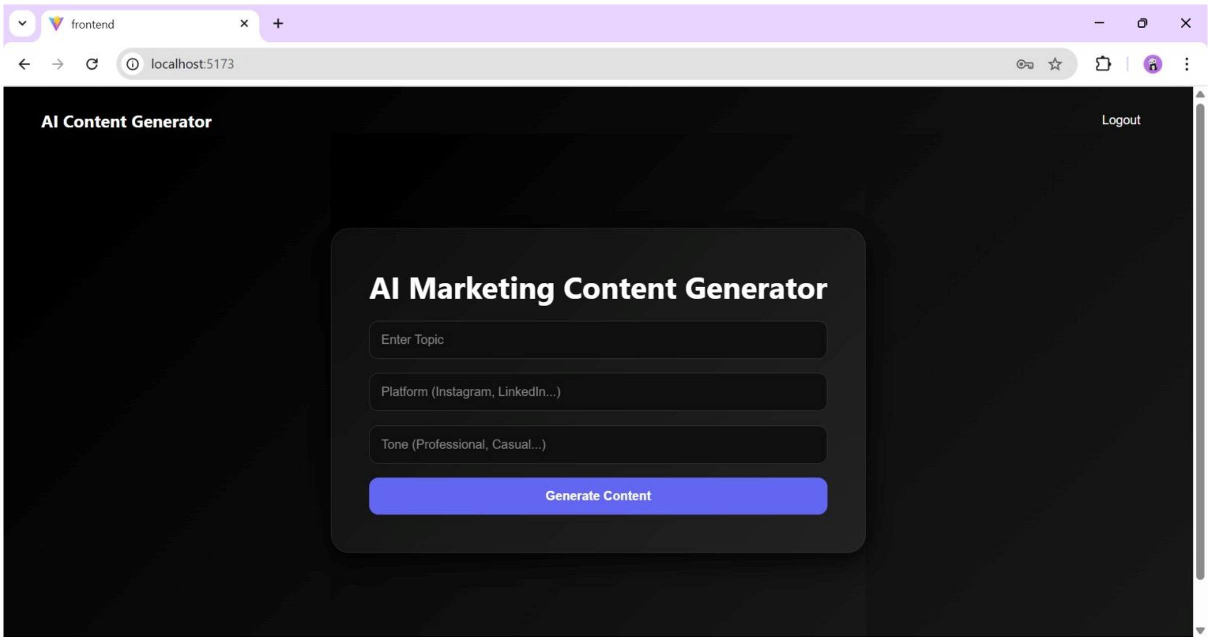
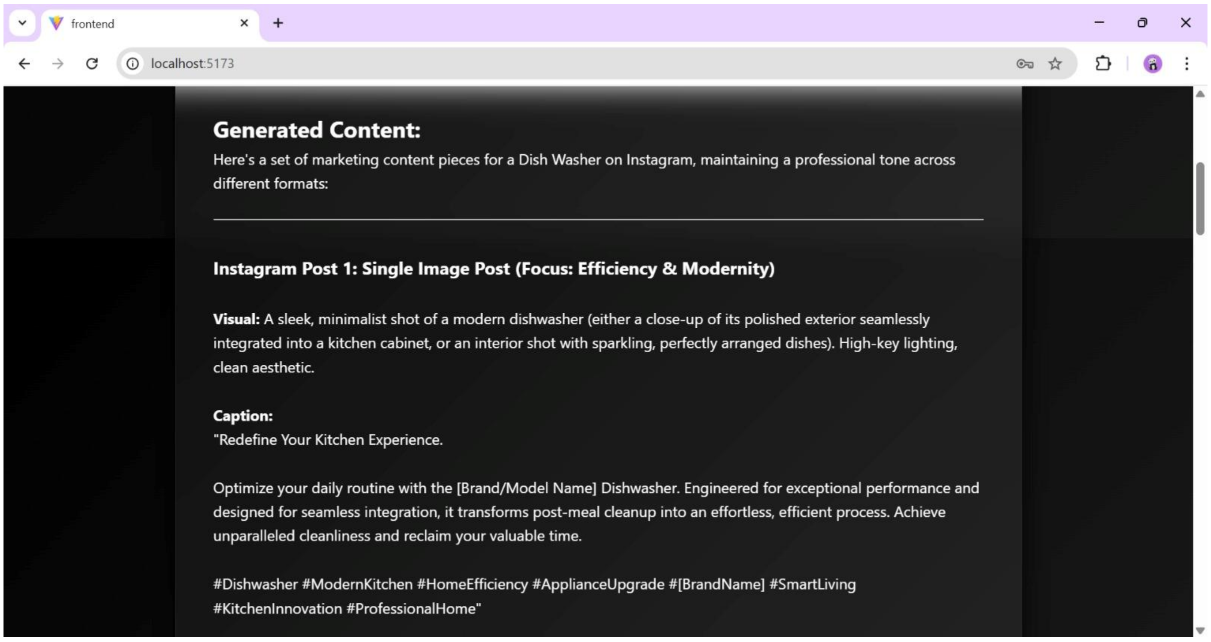

frontend
localhost:5173
AI Content Generator
Logout
AI Marketing Content Generator
Enter Topic
Platform (Instagram, LinkedIn...)
Tone (Professional, Casual...)
Generate Content

Fig. Screenshot of Home Page



Generated Content:
Here's a set of marketing content pieces for a Dish Washer on Instagram, maintaining a professional tone across different formats:

Instagram Post 1: Single Image Post (Focus: Efficiency & Modernity)

Visual: A sleek, minimalist shot of a modern dishwasher (either a close-up of its polished exterior seamlessly integrated into a kitchen cabinet, or an interior shot with sparkling, perfectly arranged dishes). High-key lighting, clean aesthetic.

Caption:
"Redefine Your Kitchen Experience.

Optimize your daily routine with the [Brand/Model Name] Dishwasher. Engineered for exceptional performance and designed for seamless integration, it transforms post-meal cleanup into an effortless, efficient process. Achieve unparalleled cleanliness and reclaim your valuable time.

#Dishwasher #ModernKitchen #HomeEfficiency #ApplianceUpgrade #[BrandName] #SmartLiving #KitchenInnovation #ProfessionalHome"

Fig. Screenshot of Generated Output

3.2 - Reports / dashboards / models:

The application successfully models user requests by wrapping the three core inputs into a sophisticated backend prompt before sending it to the gemini-flash model. The UI dashboard acts as a seamless conduit, providing a clean, distraction-free environment for the user to iterate on their content.

3.3 - Key outcomes:

The integration of the gemini-flash model allows the application to achieve near-instantaneous content generation. The "Regenerate" functionality successfully provides distinct, fresh variations of the copy on every click without page reloads. The authentication gateway successfully isolates user sessions, laying the groundwork for the Vector DB to curate a highly personalized marketing assistant.

4 - Conclusion

The Marketing Content Generator successfully bridges the gap between advanced Generative AI and everyday marketing workflows. By developing a full-stack local web application using React and FastAPI, we created a robust platform that transforms simple inputs (Topic, Platform, Tone) into professional-grade copy. Integrating the gemini-flash API alongside a Vector Database demonstrated how speed and personalization can be combined to solve real-world operational bottlenecks. The key learning from this project was mastering the orchestration of disparate technologies—managing user state in React, handling API communication and prompt engineering in FastAPI, and structuring data for vectorization—to create a unified, user-centric AI tool.

5 - Future Scope & Enhancements

While the current local deployment meets all core objectives, future enhancements could significantly expand the application's utility:

- **Cloud Deployment:** Migrating the local server architecture to a cloud provider (like AWS or Google Cloud) to allow multi-user access across different geographic locations.
- **Automated Publishing Integrations:** Adding OAuth integrations to allow users to push the generated content directly to platforms like LinkedIn, Twitter, or WordPress straight from the dashboard.
- **Campaign Analytics:** Building a feedback loop where users can rate the performance of the generated copy, using those metrics to fine-tune the backend prompt engineering over time.