ISSN 2277-2685
IJESR/June. 2025/ Vol-15/Issue-3s/163-169
Pathlavath Swathi *et. al.*, / International Journal of Engineering & Science Research

NexGen AI : GENIUS

A.Hima Bindu, Pathlavath Swathi, Racha Veda, Karanam Vani.
[1]Assistant Professor **Department of CSE, Bhoj Reddy Engineering College for Women, India.**
[2,3,4]**B. Tech Students, Department of CSE, Bhoj Reddy Engineering College for Women, India.**

ABSTRACT

In the era of rapid digital transformation, businesses across industries are increasingly adopting AI-driven Software as a Service (SaaS) platforms to enhance productivity, automate tasks, and scale operations. These platforms integrate artificial intelligence (AI) with cloud-based service models to deliver intelligent, scalable, and accessible solutions on demand. A modern AI SaaS platform utilizes advanced technologies such as natural language processing (NLP), computer vision (CV), and deep learning (DL) to offer services like automated customer support, predictive analytics, personalized recommendations, and intelligent process automation. These capabilities reduce operational costs, increase efficiency, and enable real-time decision- making. This paper outlines the development and implementation of an AI-powered SaaS platform, leveraging a full-stack technology stack including Next.js, React, Tailwind CSS, Prisma ORM, and Stripe for seamless subscription management. Furthermore, the integration of AI APIs enables the delivery of intelligent services such as text summarization, image generation, and sentiment analysis to end users via a user-friendly web interface.

Keywords : *Deep Learning(DL); Artificial Intelligence(AI);Computer Vision(CV).*

1. INTRODUCTION

In today's fast-evolving technological landscape, artificial intelligence (AI) is no longer a luxury but a necessity for businesses and individuals striving for efficiency, accuracy, and automation. Traditional AI adoption, however, has long been constrained by the need for complex infrastructure, specialized talent, and high costs. To bridge this gap, **AI Software as a Service (SaaS)** platforms have emerged as a transformative solution, enabling access to intelligent tools directly through web-based applications.

AI SaaS platforms combine the power of AI with the flexibility of cloud services to offer intelligent features — such as natural language processing, speech recognition, computer vision, and predictive analytics — as subscription-based modules. These platforms democratize access to AI, making it possible for startups, enterprises, students, and freelancers to use advanced technologies without having to build or train their own models. With just an internet connection, users can log in to the platform and perform tasks like:

- Generating and summarizing large amounts of text in seconds,
- Creating AI-generated images from text prompts,
- Converting audio files to written transcripts,
- Writing, debugging, or improving code,
- And more — all within a single interface.

This shift in how AI is consumed mirrors the way productivity software moved from desktop applications to the cloud. AI SaaS allows businesses to avoid the hassles of model maintenance, server deployment, and security updates, as all of this is handled by the platform provider.

The motivation behind developing this AI SaaS platform stems from the need to provide a **centralized, accessible, and intelligent**

environment where users can benefit from a wide variety of AI services without any programming background. It also supports scalability for commercial use by integrating **Stripe** for secure payment processing and subscription management, and ensures real-time, low-latency performance through the use of **Next.js 13**, **React**, and **Prisma ORM**.

Especially in post-pandemic times, when remote work, virtual education, and online businesses have become standard, AI SaaS platforms present an ideal solution for enhancing productivity and decision-making without geographic or technical barriers.

This project not only addresses the lack of unified AI tools available to the public but also contributes toward the growing trend of **AI for Everyone** — bringing the intelligence of machines into the hands of every user with a simple, intuitive interfaces.

2-REQUIREMENT ANALYSIS

Funtional Requirements

Functional requirements define the key features and services provided by the AI SaaS platform. These requirements describe how the system behaves in response to specific inputs and outlines expected user interactions.

- **User Registration**: Users must be able to register using their email, username, and password.
- **User Login**: Registered users can log in securely using authentication credentials.
- **AI Tool Selection**: Users can select from various AI tools such as text summarization, image generation, audio transcription, or code generation.
- **Input Handling**: Users provide input in the form of text, audio, or image prompts depending on the selected tool.
- **Model Execution**: The platform loads pre-trained AI models (e.g., GPT, DALL·E, Whisper) and executes them based on user input.
- **Output Display**: The processed output is displayed to the user in real-time.
- **Subscription & Payment**: Integration with Stripe allows users to choose and subscribe to different plans.
- **Usage Tracking**: The system tracks API usage and limits access based on the user's subscription tier.

Non-Functional Requirements

Non-functional requirements ensure the usability, efficiency, and maintainability of the system. These are crucial for enhancing the overall user experience and ensuring the platform operates reliably at scale.

- **Performance**: Quick response times for model execution, with real-time feedback for all AI services.
- **Usability**: Clean, intuitive UI designed using Tailwind CSS to support users with minimal technical background.
- **Reliability**: System should function without crashes or unexpected behaviors, even under heavy traffic.
- **Scalability**: Designed to handle thousands of concurrent users with scalable infrastructure.

Security: Implements user authentication, authorization, and secure payment processing via Stripe.

- **Portability**: The platform is browser-based and compatible with all major operating systems.
- **Maintainability**: Code is modular and well-documented for easy updates and debugging.
- **Compatibility**: Integrates well with modern browsers and supports mobile responsiveness.

3. DESIGN

The AI SaaS Content Generation Platform employs a **multi-layered cloud-native architecture** that

balances user experience, system scalability, and resource optimization by distributing functionality across frontend, backend, database, and external AI service layers.

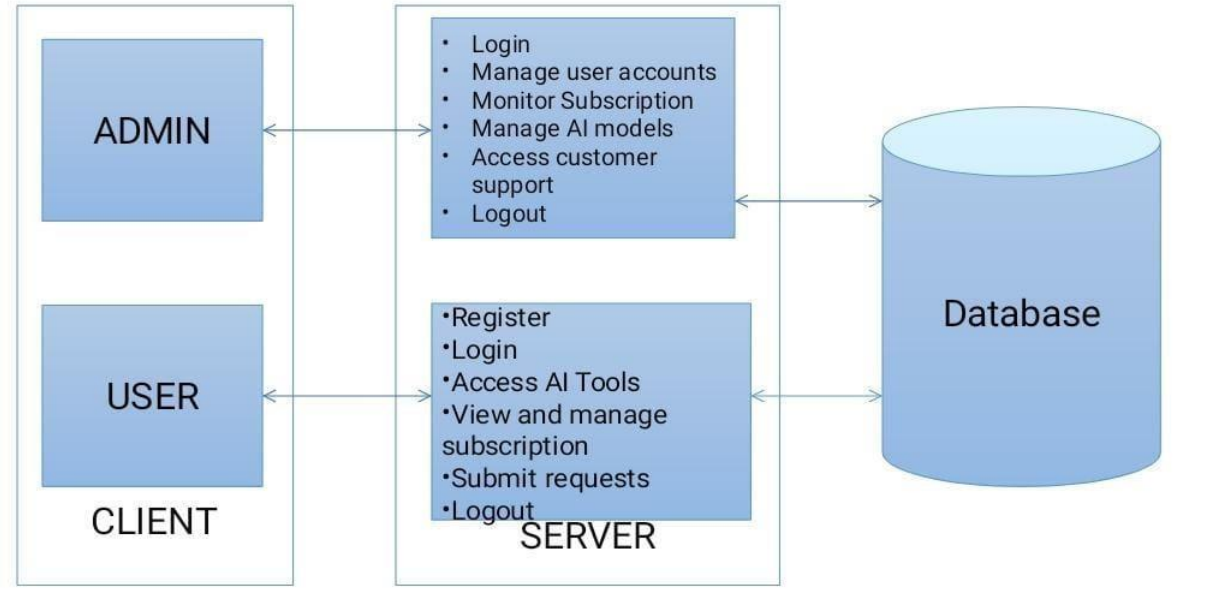
Architecture is of two types. They are

(1) Software Architecture

(2) Technical Architecture

Software Architecture:

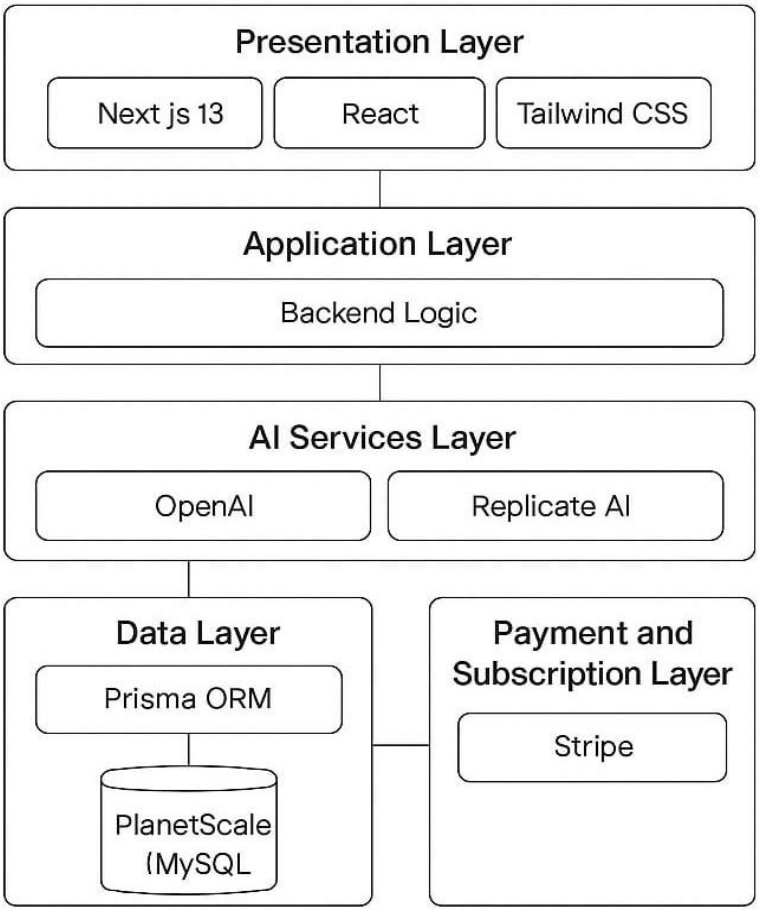
The AI SaaS Content Generation Platform uses Next.js 13 and React for a responsive frontend, styled with Tailwind CSS. User authentication is managed by Clerk, supporting multiple login methods and securing key routes. The backend, built with Next.js API routes, handles core logic and connects to AI services like OpenAI and Replicate AI for content generation. It also enforces API rate limits for different user tiers. Data is stored securely in a MySQL-compatible database accessed via Prisma ORM, managing users, sessions



Software Architecture

Technical Architecture:

The hardware architecture supporting the AI SaaS Content Generation Platform is based primarily on cloud infrastructure. Users connect to the platform through web browsers on various client devices including desktops, laptops, tablets, and smartphones, benefiting from a responsive and efficient interface. The application's frontend and backend components are hosted on cloud servers, which provide the computational resources needed to serve web pages and process API requests reliably. These cloud servers ensure scalability and high availability of the platform. The database is also hosted on cloud-based managed services, such as PlanetScale, which offers a scalable and secure environment for storing user data and generated content. Intensive AI computations, including content generation tasks, are offloaded to third-party cloud AI services like OpenAI and Replicate AI. This architecture allows the platform to handle demanding AI workloads without stressing its own hardware resources, enabling smooth performance and the ability to scale based on user demand.



Technical Architecture

4. IMPLEMENTATION

This system is developed using a modern web development stack focused on building scalable and efficient AI-powered SaaS platforms..

AI SaaS Platform Technology Stack

The AI SaaS platform is built using **Next.js 13**, a React-based framework for server-side rendering and static site generation, which provides high performance and SEO benefits. The frontend is styled using **Tailwind CSS**, a utility-first CSS framework that allows rapid UI development with customizable components. For backend services, **Prisma ORM** is used to interact with databases efficiently, providing type safety and easy migrations. **Stripe** is integrated to handle secure payment processing and subscription management. Authentication is managed via **Clerk**, which simplifies user sign-up, login, and session management. State management across the platform is handled with **Zustand**, enabling lightweight and scalable state control.

For AI functionalities, the platform connects with APIs or integrates machine learning models to deliver intelligent features, such as content generation or data analytics.

The platform supports deployment on scalable cloud providers and can be easily extended with additional services using microservice architecture or serverless functions

5-TESTING

Unit Testing

During This first round of testing, the program is submitted to assessments that focus on specific units or components of the software to determine whether each one is fully functional

In this phase, a unit can refer to a function, individual program or even a procedure, and White box testing method is usually used to get the job done. One of the biggest benefits of this testing phase is that it can be run every time a piece of code is changed, allowing issues to be resolved as quickly as possible. It quite common for software developers to perform unit tests before delivering software to testers for formal testing.

Integration Testing

Integration testing allows individuals the opportunity to combine all of the units within a program and test them as a group. This testing level is designed to find interface defects between the modules/functions. This is particularly beneficial because it determines how efficiently the units are running together. Keep in mind that no matter how efficiently each unit is running, if they properly integrated, it will affect the functionality of the software program. In order to run these types of tests, individuals can make use of various testing methods, but the specific method that will be used to get the job done will depend greatly on the way in which the units are defined.

System Testing

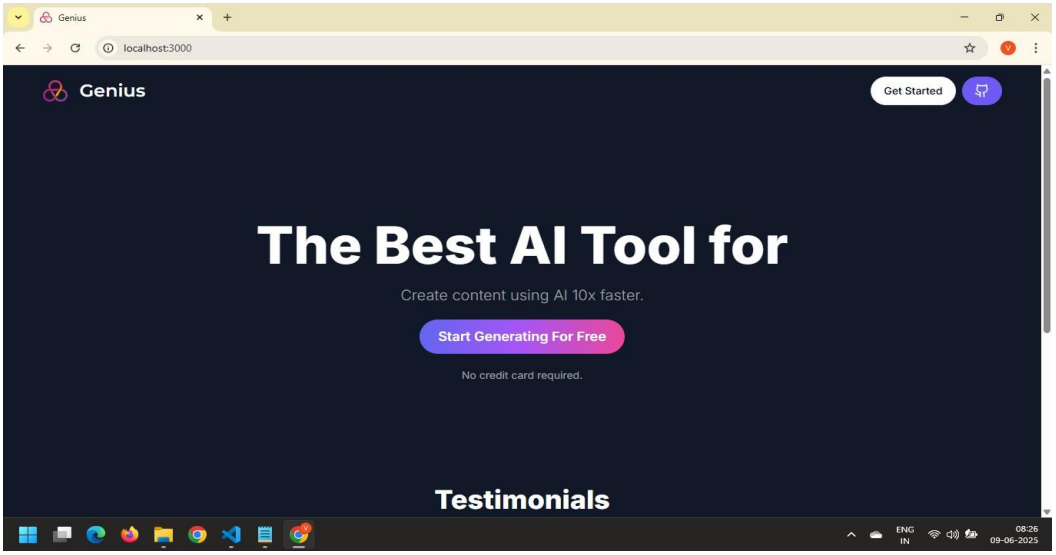
System testing is the first level in which the complete application is tested as a whole. The goal at this level is to evaluate whether the system has complied with all of the outlined requirements and to see that it meets Quality Standards. System testing is undertaken by independent testers who haven't played a role in developing the program. This testing is performed in an environment that closely mirrors production. System Testing is very important because it verifies that the application meets the technical, functional, and business requirements that were set by the customer.

Acceptance Testing

The final level, Acceptance testing is conducted to determine whether the system is ready for release. During the Software development life cycle, requirements changes can sometimes be misinterpreted in a fashion that does not meet the intended needs of the users. During this final phase, the user will test the system to find out whether the application meets their business needs. Once this process has been completed and the software has passed, the program will then be delivered to production.

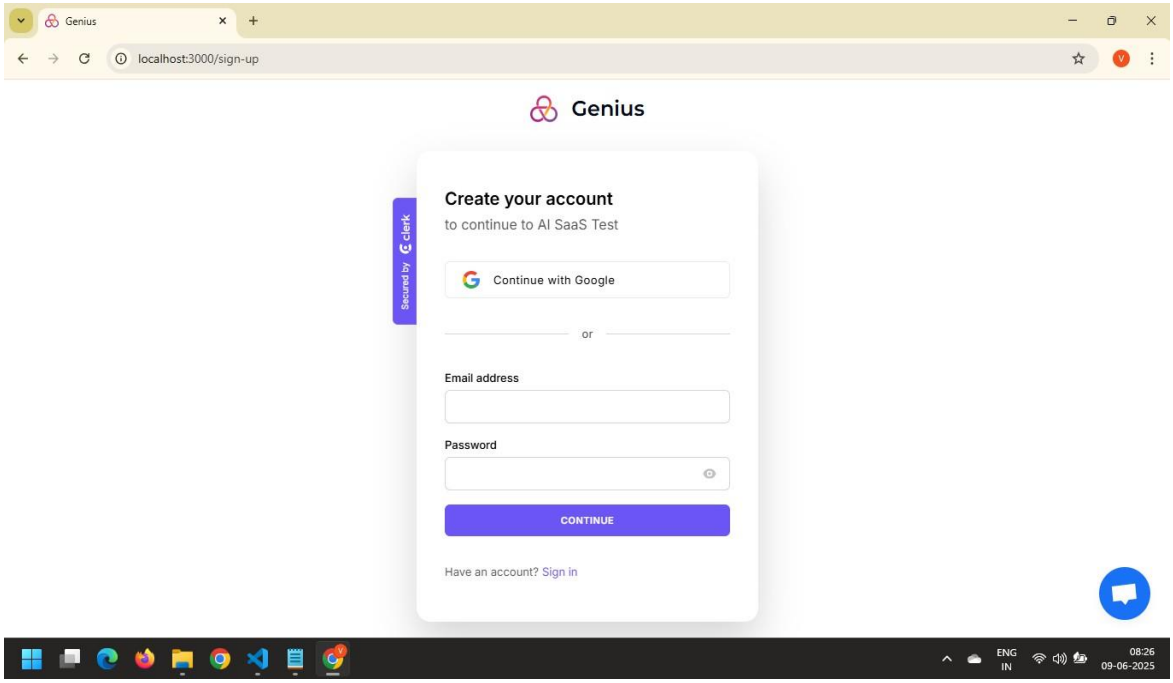
6-SCREENSHOTS

Home Page

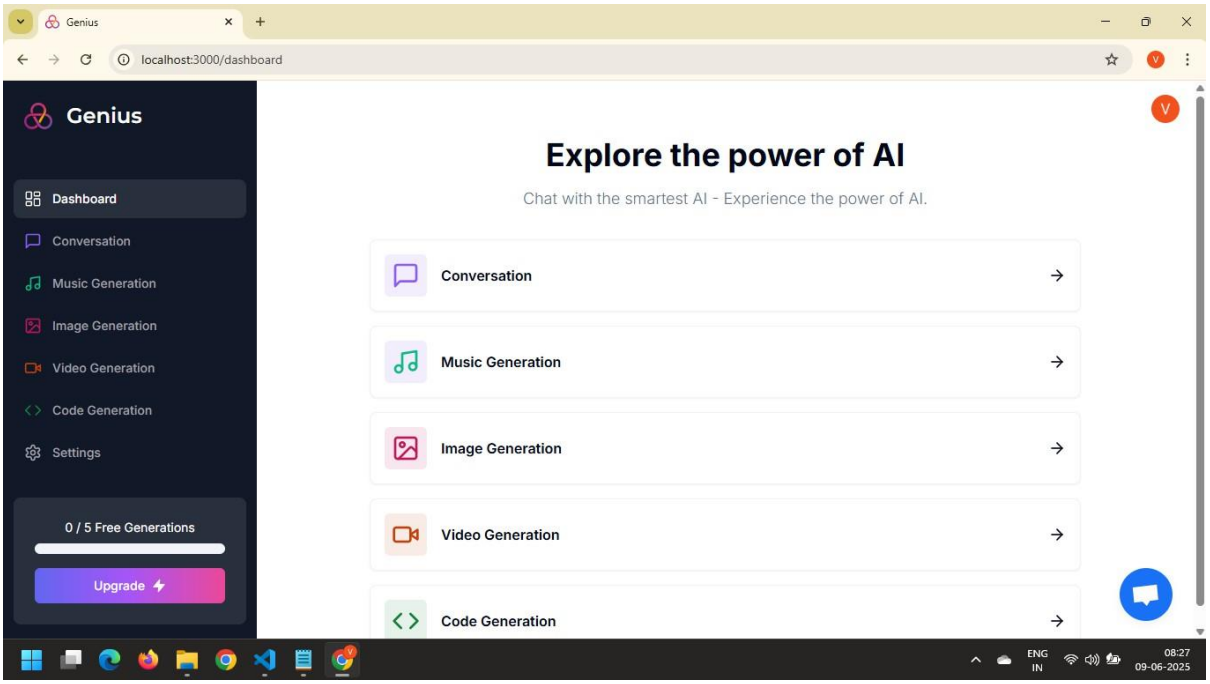


Screenshot 6.1 Home Page

Admin and user login

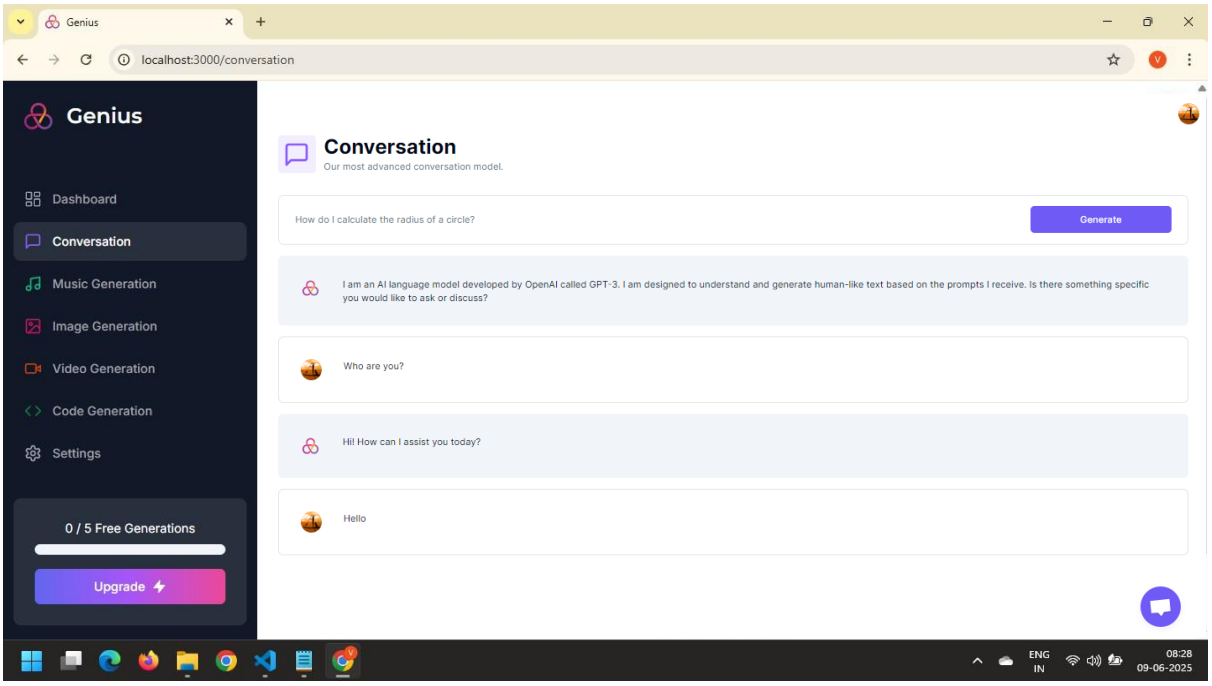


Screenshot 6.2 Admin and User Login

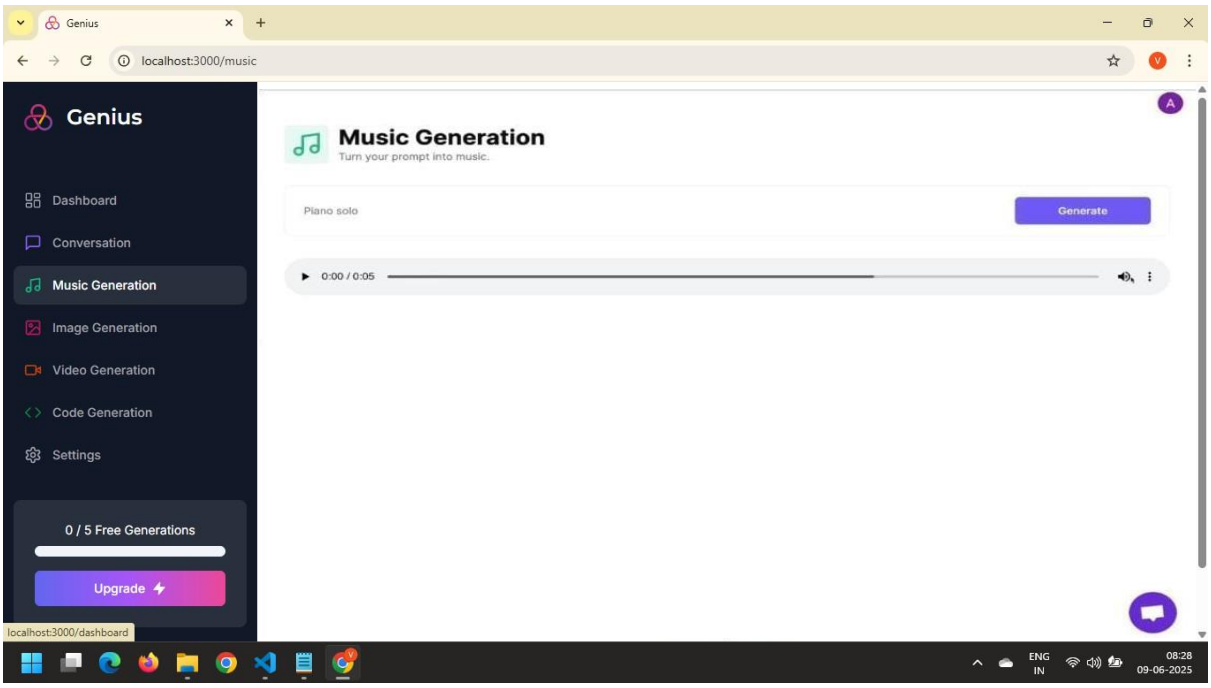


Screenshot 6.3 Admin Dashboard

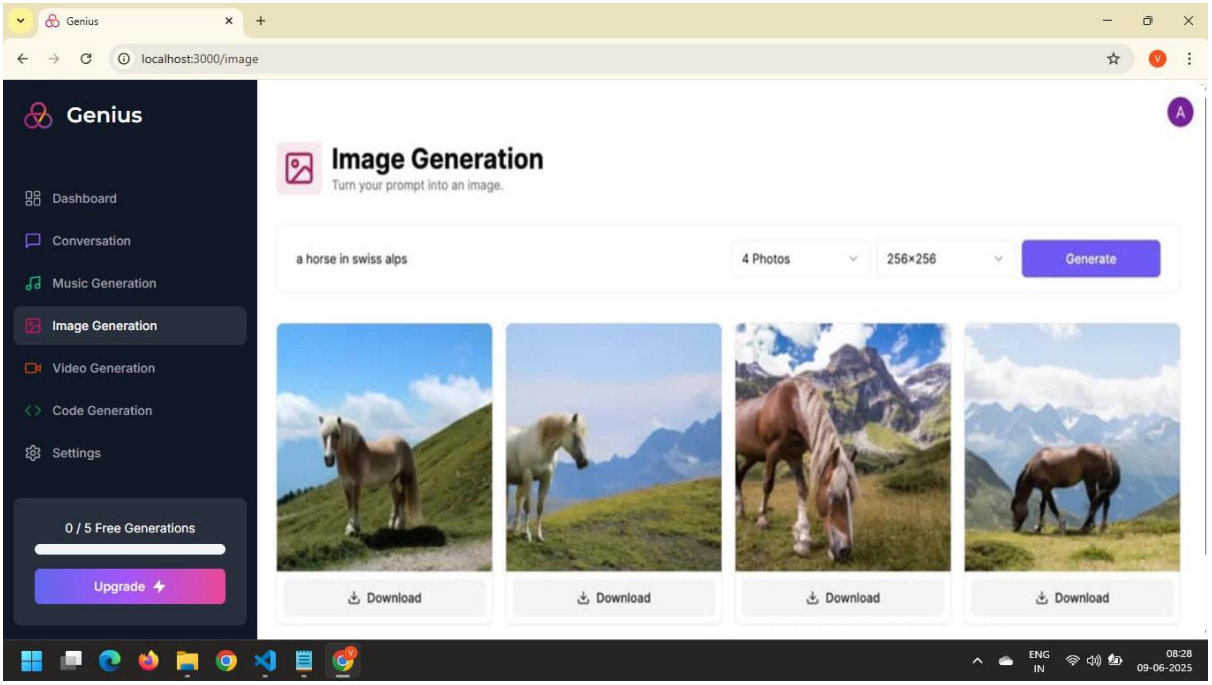
Conversation



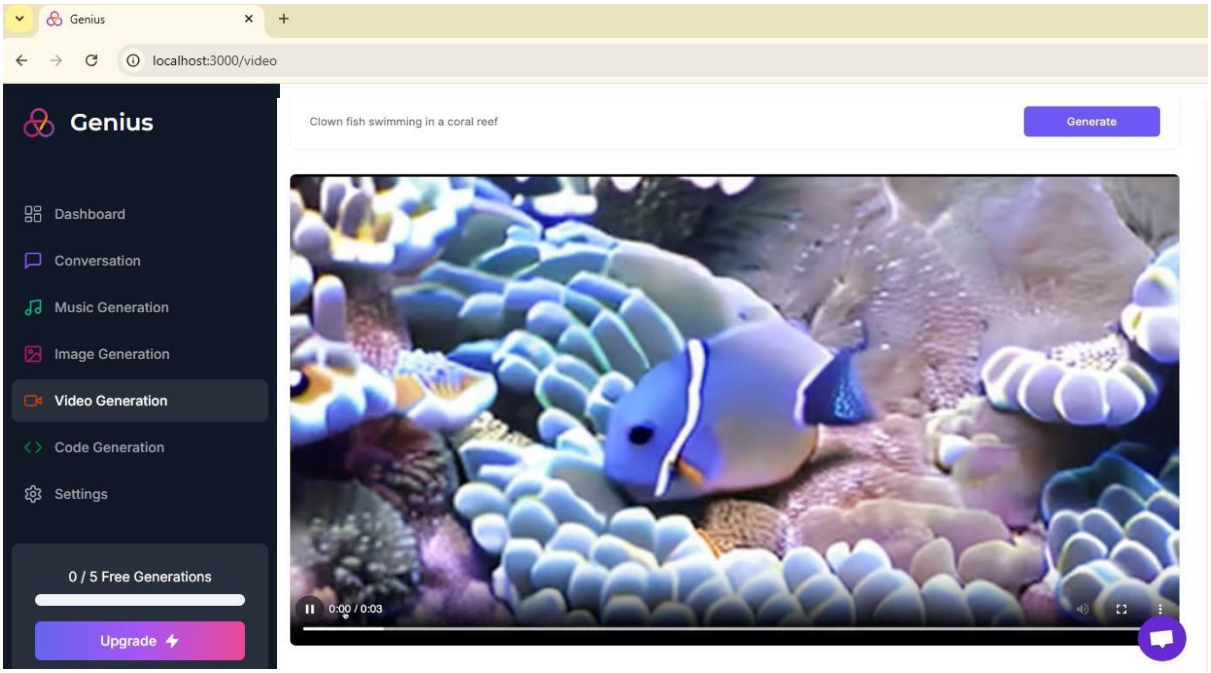
Screenshot 6.4 Conversation



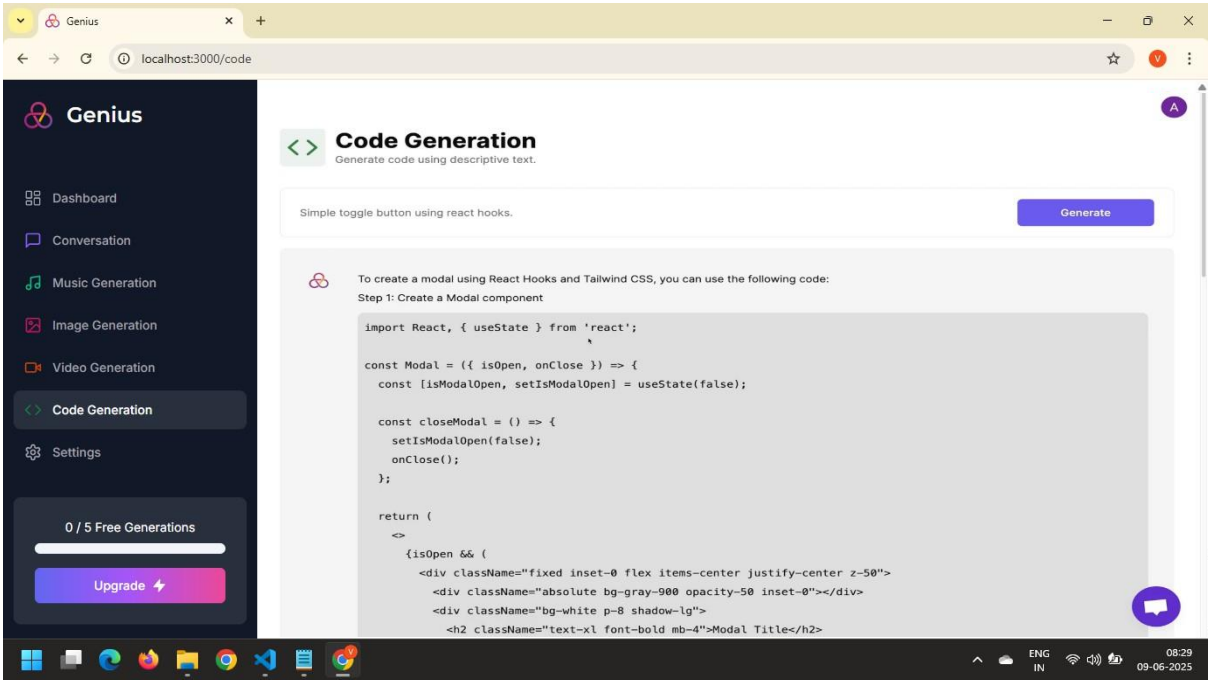
Screenshot 6.5 Music



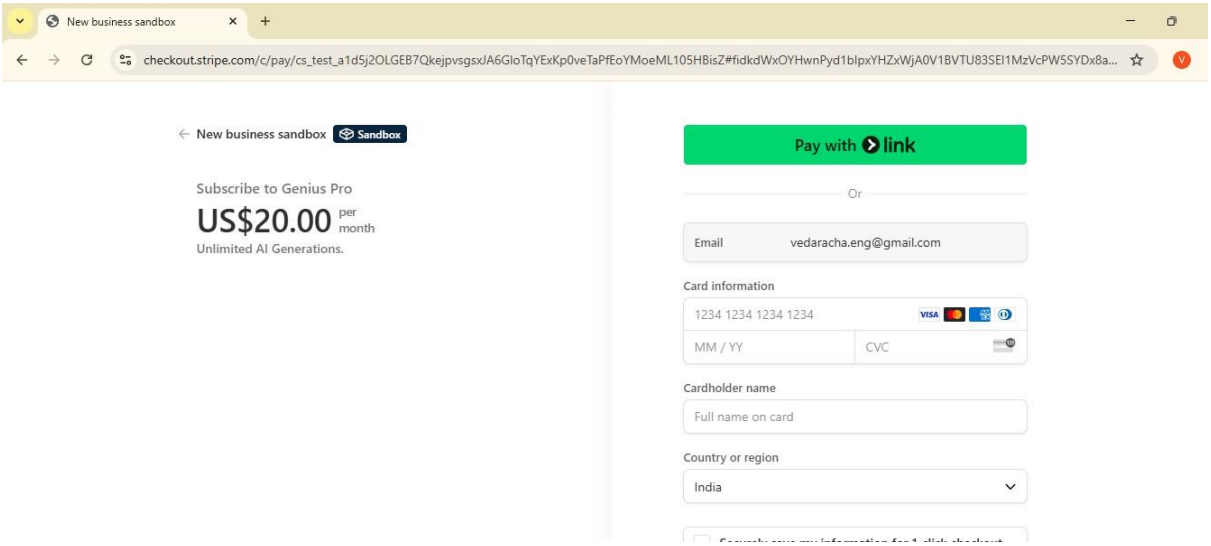
Screenshot 6.6 image



Screenshot 6.7 Video



Screenshot 6.8 code



Screenshot 6.9 Subscription

7-CONCLUSION

This web-based SaaS platform offers a unified suite of AI-powered generators—covering chat, images, music, video, and code—all built with Next.js 13 App Router, React, Tailwind CSS, Prisma/Drizzle ORM, Clerk auth, and Stripe billing. It integrates OpenAI and Replicate for varied AI generation tools and is structured to be fully responsive and scalable. It successfully delivers a polished user experience with secure sign-in/signup, free-tier usage limits, and subscription management, making it a robust starting point for anyone looking to deploy AI content services.

REFERENCES

1. athrala. (2025). AI-SaaS-Content-Generation-Platform [Computer software]. GitHub. Retrieved June 2025

2. Kumar, P. K., Sekhar, S., & Singh, R. (2025). AI Content Generator SaaS Product Using Next.js and LLM. International Journal for Scientific Research & Technology (IJSRT). This study outlines a cloud-based AI content generation platform built on Next.js with GPT-3/4 and discusses architecture, authentication, subscription, and deployment using Vercel, Razor pays, Drizzle ORM, and PostgreSQL.

3. Li, J., Tang, T., He, G., Jiang, J., Hu, X., Xie, P., Chen, Z., Yu, Z., Zhao, W. X., & Wen, J.-R. (2021). TextBox: A Unified, Modularized, and Extensible Framework for Text Generation. arXiv. Highlights modular architectures for text-generation systems and reusable pipelines.

4. Xu, L., & Wang, Y. (2019). XCloud: Design and Implementation of AI Cloud Platform with RESTful API Service. arXiv. Discusses architecture for scalable AI SaaS platforms exposing AI models via RESTful APIs.

5. Hu, Z., Shi, H., Tan, B., Wang, W., Yang, Z., Zhao, T., He, J., Qin, L., Wang, D., Ma, X., Liu, Z., Liang, X., Zhu, W., Sachan, D. S., & Xing, E. P. (2018). Texar: A Modularized, Versatile, and Extensible Toolkit for Text Generation. arXiv. A toolkit that influences design of modular AI-content engines.