Archives available at journals.mriindia.com

International Journal of Electrical, Electronics and Computer Systems

ISSN: 2347-2820
Volume 14 Issue 01, 2025

AI Image Generator Using Open Source for Text to Image

Prof.Samina Anjum Lateef[1], Mohd Alkama Ansari[2], Monir Ansari[3], Tanmay Manwatkar[4]

[1]Assistsnt Professor Department of computer science and Engineering Anjuman College Of Engineering and Technology, Nagpur, India
[2,3,4] Department of computer science and Engineering Anjuman College Of Engineering and Technology, Nagpur, India

Peer Review Information

Submission: 07 Feb 2025
Revision: 16 Mar 2025
Acceptance: 18 April 2025

Keywords

AI Image Generators
Generative Adversarial Networks
Test To Image

Abstract

AI image generators have revolutionized digital creativity by facilitating the automatic production of visual content. These tools employ sophisticated machine learning methods, like Generative Adversarial Networks (GANs) and diffusion models, to create images from text descriptions or existing data. Their uses extend across multiple sectors, including art, advertising, and entertainment, enhancing design processes and encouraging innovation. Despite their advantages, issues related to ethical considerations, biases in training data, and copyright concerns persist. As technology advances, AI-generated images are anticipated to become more accurate, accessible, and versatile, thereby influencing the future of digital content creation.

INTRODUCTION

AI image generators are advanced tools that leverage artificial intelligence to create visuals from text descriptions, sketches, or existing images. These systems rely on deep learning techniques, including Generative Adversarial Networks (GANs) and diffusion models, to produce high-quality, realistic images.
The emergence of AI-driven image generation has transformed industries such as digital art, design, marketing, and entertainment. With just a simple text prompt, users can create a wide range of visuals—from lifelike portraits to imaginative landscapes—without requiring professional design expertise.

LITERATURE REVIEW

The field of AI image generation has evolved significantly, driven by advancements in deep learning techniques such as Generative Adversarial Networks (GANs) and diffusion models. Marked a breakthrough in generating realistic images by training a generator and a discriminator network in competition. More recent models, such as DALL·E, Stable Diffusion, and Mid Journey, utilize diffusion processes to enhance image quality and provide greater control over the generated outputs. Research has shown that AI-generated images have wide-ranging applications in digital art, advertising, gaming, and entertainment, where they streamline creative workflows and open new possibilities for artistic expression.
Despite their advantages, AI image generators present several challenges that researchers continue to explore. Ethical concerns, such as biases in training data, misinformation, and the unauthorized use of copyrighted material, have raised debates about responsible AI use. Studies emphasize the need for improved model transparency and user control to mitigate these risks. Additionally, legal discussions surrounding

ownership and authenticity of AI-generated content remain unresolved. Ongoing research aims to refine these models by addressing biases, improving accuracy, and incorporating ethical guidelines to ensure AI-generated images are used responsibly across industries.

PROBLEM STATEMENT

The rapid advancement of artificial intelligence (AI) in image generation has brought significant improvements in visual content creation. However, despite these advancements, several challenges remain in generating high-quality, realistic images while ensuring ethical considerations and computational efficiency. The proposed work aims to address these issues by developing an optimized AI image generation model that overcomes existing limitations.

- **High Computational Costs:** Most AI image generation models require extensive computing power, making them inaccessible to users with limited resources.
- **Image Artifacts and Distortions:** Some models struggle with producing sharp and realistic images, resulting in visual artifacts.
- **Bias in Generated Images:** Existing models are often trained on biased datasets, leading to ethical concerns and representation issues.
- **Lack of Fine-Grained Control:** Users have limited ability to customize generated images in a precise manner.

WORK FLOW

The development process starts with building the frontend using React.js and Vite for efficient rendering, along with Tailwind CSS for styling. Three key pages are designed: a login page for user authentication, a home page with a "Generate" button, and a prompt entry page where users can input text to generate images. Real-time loading effects are implemented to enhance user experience, ensuring smooth interactions during image generation. State management is applied to handle the UI flow effectively.

For the backend, Node.js and Express.js are used to set up a server, with Nodemon enabling automatic restarts during development. MongoDB is integrated using Mongoose to store user data and past image prompts securely. Environment variables are managed through dotenv to protect sensitive API keys. The Clipdrop API is incorporated to process user inputs and generate images dynamically. RESTful API endpoints are created to facilitate data transfer between the frontend and backend.

API testing is conducted using Postman to validate request and response cycles, ensuring correct data retrieval and storage in MongoDB. Debugging is carried out to address any API or authentication issues. Once verified, the frontend and backend are connected using Axios or Fetch API. This integration allows the frontend to send user prompts to the backend and retrieve generated images, which are displayed with real-time updates. A loading animation enhances the user experience while images are being processed. The final phase includes end-to-end testing to confirm seamless functionality and optimize performance, ensuring a fully operational AI image generation system.

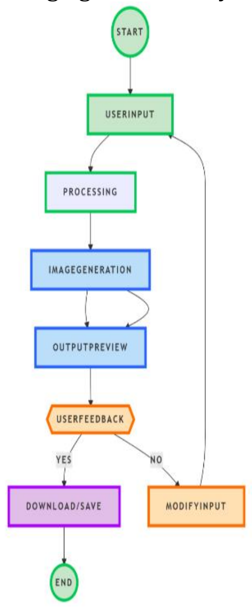


Fig. 1: Flow chart

METHODOLOGY

The development of the AI image generator follows a structured approach divided into four key stages: frontend development, backend development, API testing, and frontend-backend integration. Each step ensures smooth functionality, efficient data processing, and an optimized user experience.

Step 1: Frontend Development

1. Project Setup – The frontend is developed using React.js with Vite for faster performance. Tailwind CSS is added to create a responsive and visually appealing interface.
2. User Interface Design – Three main pages are created:
   - Login Page: Provides a simple user authentication interface for secure access.
   - Home Page: Features a "Generate" button, allowing users to navigate to the image generation page.
   - Prompt Entry Page: Enables users to input text prompts for image creation, with a "Generate" button and a loading animation to enhance user experience.
3. State Management & Real-Time Updates – React's built-in state hooks or an external state management library like Redux is used to track

user input, loading status, and image retrieval. A loading effect ensures a smooth transition while waiting for the generated image to appear.

Step 2: Backend Development

1. Server Configuration – The backend is developed using Node.js and Express.js. Nodemon is used for automatic server restarts during development to improve efficiency.
2. Database Setup – MongoDB is chosen for data storage, with Mongoose handling database interactions. It stores user credentials, prompt history, and generated images.
3. Managing Environment Variables – The dotenv package is utilized to keep sensitive information, such as API keys and database credentials, secure.
4. Integration of AI Image API – The Clipdrop API is connected to process user prompts and return AI-generated images. The backend sends the prompt request to the API and retrieves the generated image for frontend display.
5. API Development – Various endpoints are created for different functions:
   - Authentication API – Manages user login requests.
   - Prompt Submission API – Sends user-entered text to the AI model for image generation.
   - Image Retrieval API – Retrieves and sends the generated image back to the frontend.
   - Data Storage API – Saves user-generated content for future reference.

Step 3: API Testing

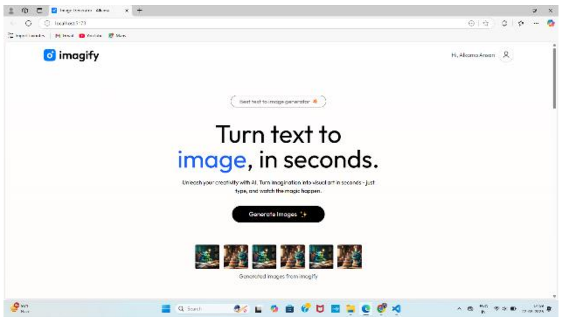
1. Postman Testing – API endpoints are tested using Postman to ensure smooth data flow and correct responses from the server.
2. Database Verification – MongoDB is checked to confirm proper storage and retrieval of user data, prompts, and images. Any database-related issues are identified and resolved.
3. Error Handling & Debugging – API responses are reviewed for errors such as incorrect requests, authentication failures, and processing delays. Proper error handling is implemented to ensure reliability.

Step 4: Frontend and Backend Integration

1. Connecting Frontend to Backend – Axios or Fetch API is used to establish communication between the frontend and backend, ensuring smooth data transfer.
2. Real-Time Image Processing – A loading effect is added to the frontend while the AI processes the prompt. Once the image is generated, it is displayed dynamically.
3. System Testing – A complete system test is conducted to verify that the frontend and backend function correctly together. This includes user interactions, API requests, and data retrieval.

RESULT

The homepage features a sleek and simple user interface. The logo, Imagify, is positioned at the top left, while a user profile icon sits in the top right. Bold main heading states, "Turn text into images in seconds," clearly conveying the tool's function. Directly beneath the heading, a "Generate Images" button enables users to start creating images instantly. At the bottom, a gallery showcases previously generated images as examples. The design is modern and visually appealing, utilizing Tailwind CSS for a clean, minimalist aesthetic.


imagify

Hi, Alkama Arsan

Best text to image generator

Turn text to image, in seconds.

Unleash your creativity with AI. Turn imagination into visual art in seconds - just type, and watch the magic happen.

Generate Images

Generated images from imagify

Fig. 3: Frontend

After entering a prompt and clicking the **"Generate"** button, users are taken to the results page.
The AI-generated image, created from the given text, is displayed in the center.
Users can choose from two options:

- **"Generate Another"** – Create a different image.
- **"Download"** – Save the image to their device.

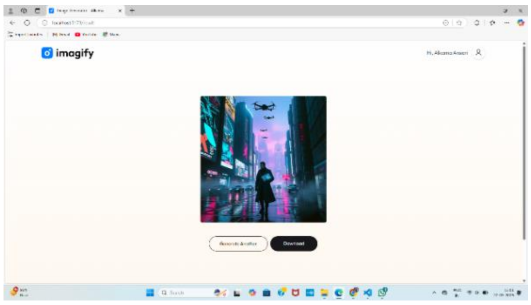


Fig. 3: Results

SYSTEM REQUIRMENT

Hardware Requirements

Processor: Intel Core i3 (8th Gen or higher) / AMD Ryzen 3 or better
RAM: 4GB (8GB recommended for smoother performance)
Storage: 10GB free space (HDD works, but SSD is better)

Graphics Card: Integrated GPU (No need for a dedicated GPU)
Internet connection: Stable broadband for API access.
Software Requirements
Frontend Development:
Operating System: Windows 7/10/11, macOS, or Linux
Frameworks & Libraries:

- React.js (Frontend)
- Vite (For fast development)
- Tailwind CSS (For styling, lightweight alternative to Bootstrap)

Browser: Google Chrome or any modern browser
Package Manager: npm
Backend Development:
Runtime Environment: Node.js (v14 or later)
Frameworks & Libraries:

- Express.js (For server-side logic)
- Mongoose (For database interaction)
- Dotenv (For managing environment variables)

Database: MongoDB Atlas (cloud-based, no need for local setup)
API Service: Clipdrop API (handles AI image generation in the cloud)
Testing Tool: Postman (For testing API endpoints)

FUTURE SCOPE

Enhanced Image Quality – Future AI will produce clearer, more detailed, and highly realistic images.
Instant Generation – AI models will create images almost instantly, eliminating long wait times.
Greater Customization – Users will have the ability to modify various aspects of generated images, such as colors, backgrounds, and styles.

CONCLUSION

The development of an AI image generator involves integrating frontend, backend, and AI-powered APIs to create a seamless user experience. By using React.js with Vite and Tailwind CSS, the frontend provides an intuitive interface for users to input text prompts and generate images. The backend, built with Node.js, Express.js, and MongoDB, ensures efficient data management and API communication. The integration of the Clipdrop API enables real-time image generation, while Postman testing ensures the reliability of the API endpoints.

References

Zolnamar Dorjsembe, Hsing-Kuo Pao, Sodtavilan Odonchimed, and Furen Xiao. Conditional diffusion models for semantic 3d brain mri synthesis. IEEE Journal of Biomedical and Health Informatics, 28(7):4084–4093, 2024.

David Foster. Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play. O'Reilly Media, 2nd edition, 2023.

Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Greater creative control for ai image generation, Jul 2022.

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. 2023. Multidiffusion: Fusing Diffusion Paths for Controlled Image Generation. (2023).