

Image Generation using Artificial Intelligence

Progress Report

Submitted in the partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION SECURITY

Submitted by:

Prakash Singh

20BCS3707

Under the Supervision of:

Er. Shiwani Sharma



CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413,

PUNJAB

MAY 2023

DECLARATION

I, **'PRAKASH SINGH'** student of **'Bachelor of Engineering in Computer Science with specialization in Information Security, session: 2020-2024**, Department of Computer Science and Engineering, Apex Institute of Technology, Chandigarh University, Punjab, hereby declare that the work presented in this Project Work entitled **"Image Generation using Artificial Intelligence"** is the outcome of our own bona fide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Date: 15/5/2023

**Place: Chandigarh University, Mohali,
Punjab**

PRAKASH SINGH

Candidate UID: 20BCS3707

ABSTRACT

Artificial intelligence (AI) is a rapidly advancing field that aims to develop machines and computer programs that can perform tasks that would normally require human intelligence. AI is based on the idea of creating intelligent machines that can simulate human cognitive abilities, such as reasoning, learning, problem-solving, perception, and language understanding. AI is being applied to a wide range of fields, including robotics, natural language processing, computer vision, and expert systems, and has already had a significant impact on society. In the context of Image generation using Artificial Intelligence has gained significant attention in recent years, as it has the potential to revolutionize the way we create and manipulate visual content. This paper presents an overview of the state-of-the-art techniques for image generation using AI, including generative adversarial networks (GANs), variational autoencoders (VAEs), and autoregressive models. These techniques have been used to generate realistic images of faces, landscapes, and even entire cities, and have a wide range of applications in fields such as art, design, and entertainment.

Keywords: Artificial Intelligence, GAN, Variational Autoencoders, natural language processing.

ACKNOWLEDGEMENT

I would like to thank my supervisor, Ms Shiwani Sharma for his guidance and advice through each stage of making this project. I would also like to thank Ms Neeru Bala for giving me this opportunity to work on a minor project in which I can show our true potential, creativity, and hard work. I would also like to thank my family and friends, who have been a constant support and have always motivated me to work hard and bring out the best in me. This project's success and end necessitated a great deal of direction and assistance from many people, and we are extremely fortunate to have received it all as part of the project's completion. I owe everything we've accomplished to their oversight and help, and we'd like to express our gratitude.

TABLE OF CONTENT

Sr.number	Title	Page number
0	1. Title page 2. Declaration of Student 3. Abstract 4. Acknowledgement 5. Table of content	1-5
1	1. Introduction 1.1. Problem Definition 1.2. Project Overview 1.2.1.Existing System 1.2.2.Proposed System 1.3. Hardware Specification 1.4. Software Specification	6-13
2	Literature Survey	14-18
3	Problem Formulation	19
4	Methodology	20-21
5	Objectives	22
6	Result	23
7	Discussion	24
8	Conclusion	25
9	Future work	26
10	References	27
11	Code and output	28-34

1 INTRODUCTION

1.1 Problem Definition

The problem this project aimed to solve was the difficulty in generating unique and creative images without advanced design skills or software. While there are several applications and tools for image generation, most of them require technical knowledge or expertise, making them inaccessible to non-experts. Additionally, these tools may not generate images that align with the user's creative vision or meet their specific requirements. Therefore, the problem was to create an application that allows users to generate unique and creative images without any technical knowledge or expertise.

Moreover, Art and Design industry has always been an expensive and time-consuming field which require lot of resources and skilled creative people to create interactive interfaces, logo, NFTs, posters and product design. Small time businesses and individuals can't afford the kind of prices and time required for this.

1.2 Project Overview

The project aimed to create a web application that uses DALL-E API and MERN stack for image generation. DALL-E API generates images from textual descriptions, and MERN stack is a combination of MongoDB, Express.js, React.js, and Node.js that allows for the development of full-stack web applications. The application allows users to generate unique and creative images by simply entering a textual description, without requiring any technical knowledge or expertise.

The project was developed using the agile methodology, with iterative and incremental development. The MERN stack was used for the development of the application, with MongoDB used as the database, Express.js used as the backend framework, React.js used as the frontend framework, and Node.js used as the runtime environment. The DALL-E API was integrated into the application using the Axios library, which allows for making HTTP requests to the API.

The user interface of the application was developed using React.js and Tailwind CSS with generated images displayed on the user interface. Users can save the images to their account or download them to their local device.

The project aimed to provide a user-friendly, responsive, and scalable web application for image generation, accessible to non-experts. The project aimed to demonstrate the capabilities of DALL-E API and the effectiveness of MERN stack for the development of full-stack web applications.

1.2.1. EXISTING SYSTEM

- Digital painting: Digital painting involves using software such as Adobe Photoshop or Corel Painter to create images using digital brushes and other painting tools.
- Photo manipulation: Photo manipulation involves using software such as Adobe Photoshop or GIMP to edit and manipulate existing images, often by combining multiple images or altering the colours and shapes of elements within an image.
- 3D modelling: 3D modelling involves creating three-dimensional objects and environments using software such as Blender or Maya. These objects and environments can then be rendered as 2D images or animations.

1.2.2. PROPOSED SYSTEM

The proposed system is a web application for image generation using DALL-E API and MERN stack. The system will allow users to generate unique and creative images by simply entering a textual description, without requiring any technical knowledge or expertise. The proposed system will be developed using the agile methodology, with iterative and incremental development.

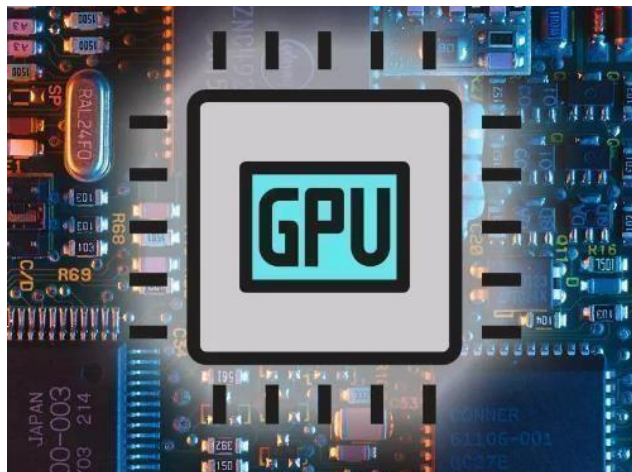
The system will use DALL-E API for image generation, which generates images from textual descriptions. The system will integrate the DALL-E API using the Axios library, which allows for making HTTP requests to the API. The system will use MERN stack for the development of the application, with MongoDB used as the database, Express.js used as the backend framework, React.js used as the frontend framework, and Node.js used as the runtime environment.

The proposed system will be scalable, allowing for the addition of new features and functionalities as required. The system will be designed with a responsive layout, allowing users to access the application from different devices and screen sizes. The proposed system will be user-friendly, with an intuitive and easy-to-use interface that requires no technical knowledge or expertise

1.3 Hardware Specification

PC/Laptop: - A PC is a personal computer that can be used for multiple purposes depending on its size, capabilities, and price. They are to be operated directly by the end user.

Personal computers are single-user systems and are portable. This makes it feasible for individual use.



1.4 Software Specification

Visual Studio Code

Visual Studio Code is a code editor redefined and optimized for building and debugging modern web and cloud applications.

MongoDB:

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database, MongoDB is written in C++. This tutorial will give you great understanding on MongoDB concepts needed to create and deploy a highly scalable and performance-oriented database.

program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server-Side Public License (SSPL).



Fig : <https://techcrunch.com/2019/06/18/mongodb-gets-a-data-lake-new-security-features-and-more/>

Before proceeding to learn “mongoDB” , you should have a basic understanding of database, text editor and execution of programs, etc. Because we are going to develop high performance database, so it will be good if you have an understanding on the basic concepts of Database (RDBMS).

Express:

In layman terms, Express.js is a web application framework that is built on top of Node.js. It provides a minimal interface with all the tools required to build a web application. Express.js adds flexibility to an application with a huge range of modules available on npm that you can directly plug into Express as per requirement. It helps in easy management of the flow of data between server and routes in the server-side applications. It was developed by TJ Holowaychuk and was released in the market on 22nd of May, 2010. Formerly it was managed by IBM but currently, it is placed under the stewardship of the Node.js Foundation incubator.



Fig <https://www.edureka.co/blog/expressjs-tutorial/>

Express is majorly responsible for handling the backend part in the MEAN stack. Mean Stack is the open-source JavaScript software stack that is heavily used for building dynamic websites and web applications in the market. Here, MEAN stands for MongoDB, Express.js, AngularJS, and Node.js.

That being said, let's now move further with this Express.js Tutorial and find out the various features of this framework.

ReactJS:

ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front-end library responsible only for the view layer of the application.

It was created by Jordan Walke, who was a software engineer at Facebook. It was initially developed and maintained by Facebook and was later used in its products like WhatsApp & Instagram. Facebook developed ReactJS in 2011 in its newsfeed section, but it was released to the public in the month of May 2013.



Fig : [https://en.wikipedia.org/wiki/React_\(JavaScript_library\)](https://en.wikipedia.org/wiki/React_(JavaScript_library))

Today, most of the websites are built using MVC (model view controller) architecture. In MVC architecture, react is the 'V' which stands for view, whereas the architecture is provided by the Redux or Flux.

A ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code. The components are the heart of all React applications. These Components can be nested with other components to allow complex applications to be built of simple building blocks. ReactJS uses virtual DOM based mechanism to fill data in HTML DOM. The virtual DOM works fast as it only changes individual DOM elements instead of reloading complete DOM every time.

1. NodeJS:

Node.js tutorial provides basic and advanced concepts of Node.js. Our Node.js tutorial is designed for beginners and professionals both.

Node.js is a cross-platform environment and library for running JavaScript applications which is used to create networking and server-side applications.

Our Node.js tutorial includes all topics of Node.js such as Node.js



Fig : NodeJS

installation on windows and linux, REPL, package manager, callbacks, event loop, os, path, query string, cryptography, debugger, URL, DNS, Net, UDP, process, child processes, buffers, streams, file systems, global objects, web modules etc. There are also given Node.js interview questions to help you better understand the Node.js technology

3 Literature Survey

Image generation using Artificial Intelligence (AI) has seen significant developments in recent years, and there are several methods that researchers use to generate images.

"Generative Adversarial Networks" by Ian J. Goodfellow et

[1] GANs consist of two neural networks: a generator and a discriminator. The generator produces images, and the discriminator evaluates the images to determine whether they are real or fake. Over time, the generator learns to produce increasingly realistic images that can fool the discriminator. Since the introduction of GANs, researchers have proposed many variations, including conditional GANs, progressive GANs, and CycleGANs. Conditional GANs allow for the generation of images with specific characteristics, while progressive GANs generate high-resolution images by gradually increasing the size of the images. CycleGANs can be used to generate images in a different style from the input images, which has applications in transferring styles between images.

Variational Autoencoders (VAEs), which was introduced by Kingma and Welling [2] in 2013. VAEs are a type of unsupervised learning model that learns the underlying distribution of data. VAEs comprise of an encoder and a decoder. The encoder maps the input image into a lower-dimensional space, and the decoder maps the lower-dimensional representation back to the image space. Researchers have proposed variations of VAEs, including Adversarial Autoencoders and Vector Quantized VAEs. Adversarial Autoencoders combine the principles of GANs and VAEs to generate more realistic images. Vector Quantized VAEs use a discrete latent space, which allows for greater control over the generation of images.

Tero Karras et al [3] proposes A Style-Based Generator Architecture for Generative Adversarial Networks which uses a novel generator architecture that allows for increased control over the image synthesis process, enabling better resolution, placement, and texture. The authors also introduce a new regularization technique called path length regularization (PLR), which encourages smooth and continuous changes in the image space. Evaluation on several benchmark datasets showed that StyleGAN outperforms other state-of-the-art generative models in terms of image quality, diversity, and realism. The paper presents interesting applications of StyleGAN such as image editing, interactive image synthesis, and image interpolation.

Phillip Isola et al [4] proposes a conditional adversarial network (cGAN) for image-to-image translation tasks such as style transfer, colorization, and image synthesis. The cGAN learns a mapping between input and output images by using a loss function that measures the difference between the generated and ground-truth images. The paper demonstrates the effectiveness of cGANs on various tasks such as converting sketches to photographs, synthesizing street scenes from aerial photos, and colorizing grayscale images. The approach presented in the paper has since become a popular method for image-to-image translation tasks.

Justin Johnson et al [5] examines a new approach to image generation that uses a perceptual similarity metric to guide the generator towards generating more realistic and visually pleasing images. The authors use a pre-trained deep neural network to measure the similarity between generated and real images and incorporate this measure into the generator's loss function. The proposed method improves image quality and reduces artifacts in generated images. The paper demonstrates the effectiveness of the proposed approach on various datasets, including faces, natural images, and handwritten digits. The proposed approach has since inspired several related works in the field of image generation.

Ting-Chun Wang et al [6] developed a novel architecture for conditional GANs, called Progressive Attention GAN, that enables high-resolution image synthesis and semantic manipulation. The proposed approach uses a progressive growing scheme to synthesize high-resolution images and introduces an attention mechanism that allows the generator to selectively attend to different image regions. The paper also demonstrates the effectiveness of the proposed approach for semantic manipulation tasks, such as object removal and addition. Evaluation on several datasets shows that the proposed approach outperforms existing state-of-the-art methods in terms of image quality, diversity, and manipulation capability.

Alec Radford et al propose a new deep learning framework called DCGAN for unsupervised learning of hierarchical representations from image data. The authors introduce a new architecture for generative models that combines convolutional neural networks (CNNs) and GANs. The proposed DCGAN framework is capable of generating high-quality images with a variety of object shapes and background patterns. The paper also demonstrates that the learned representations can be used for image classification and image retrieval tasks, indicating the effectiveness of the proposed approach. DCGAN has since become a popular approach for unsupervised representation learning.

Paper	Objective	Methodology	Contribution	Date	Data Source
Goodfellow et al.	Introduce Generative Adversarial Networks (GANs)	GAN framework	Introduce the GAN framework for unsupervised learning and demonstrate its effectiveness in generating realistic images.	2014	Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS)
Kingma and Welling	Introduce the Variational Autoencoder (VAE)	VAE framework	Introduce the VAE framework for unsupervised learning and demonstrate its effectiveness in image generation and compression.	2013	Proceedings of the 2nd International Conference on Learning Representations (ICLR)
Karras et al.	Introduce StyleGAN	StyleGAN framework	Introduce the StyleGAN framework for high-quality image synthesis with fine-grained control over the generated images' style.	2019	Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Isola et al	Introduce Conditional GANs	Conditional GAN framework	Introduce the Conditional GAN framework for image-to-image translation, which can be used for tasks such as style transfer and semantic segmentation.	2017	Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
Johnson et al.	Introduce Perceptual Losses	Perceptual Loss framework	Introduce the Perceptual Loss framework for style transfer and super-resolution, which uses a pre-trained deep neural network to compute a perceptual distance metric between images.	2016	Proceedings of the European Conference on Computer Vision (ECCV)
Wang et al.	Introduce High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs	High-Resolution Conditional GAN framework	Introduce a variant of the Conditional GAN framework for high-resolution image synthesis and semantic manipulation.	2018	ACM Transactions on Graphics
Radford et al	Introduce Deep Convolutional GANs (DCGANs)	DCGAN framework	Introduce the DCGAN framework for unsupervised learning and demonstrate its effectiveness in generating high-quality images.	2016	4th International Conference on Learning Representations (ICLR)

4. PROBLEM FORMULATION

The problem formulation for the project of creating a web application for image generation using DALL-E API and MERN stack can be summarized as follows. The traditional methods of generating high-quality images require significant time, effort, and expertise, making it challenging for non-experts to create visually appealing images. To address this issue, the project aims to develop a web application that can automatically generate high-quality images based on user inputs. The goal is to create a user-friendly web application that leverages the DALL-E API and MERN stack to generate images efficiently. The objectives include studying the DALL-E API and MERN stack, integrating them, developing a user-friendly interface, testing the application, and evaluating user experience. However, some assumptions have been made, including the seamless integration of the DALL-E API and MERN stack, appropriate user inputs, accessibility of the web application, high-quality images, and an intuitive user interface.

5. METHODOLOGY

1. Research and study:

- 1.1. Research and study the DALLE API and its functionality for image generation.
- 1.2. Research and study the MERN stack and its components for building a web application.

2. Design and planning:

- 2.1. Identify the requirements of the web application.
- 2.2. Design the architecture of the web application.
- 2.3. Plan the development process, including setting milestones and timelines.

3. Development:

- 3.1. Develop the backend of the web application using Node.js, Express, and MongoDB.
- 3.2. Integrate the DALLE API with the backend to generate images.
- 3.3. Develop the frontend of the web application using React and Redux.
- 3.4. Develop a user-friendly interface for users to input their requirements for image generation.
- 3.5.

4. Testing and debugging:

- 4.1. Test the functionality of the web application.
- 4.2. Debug any issues or errors.

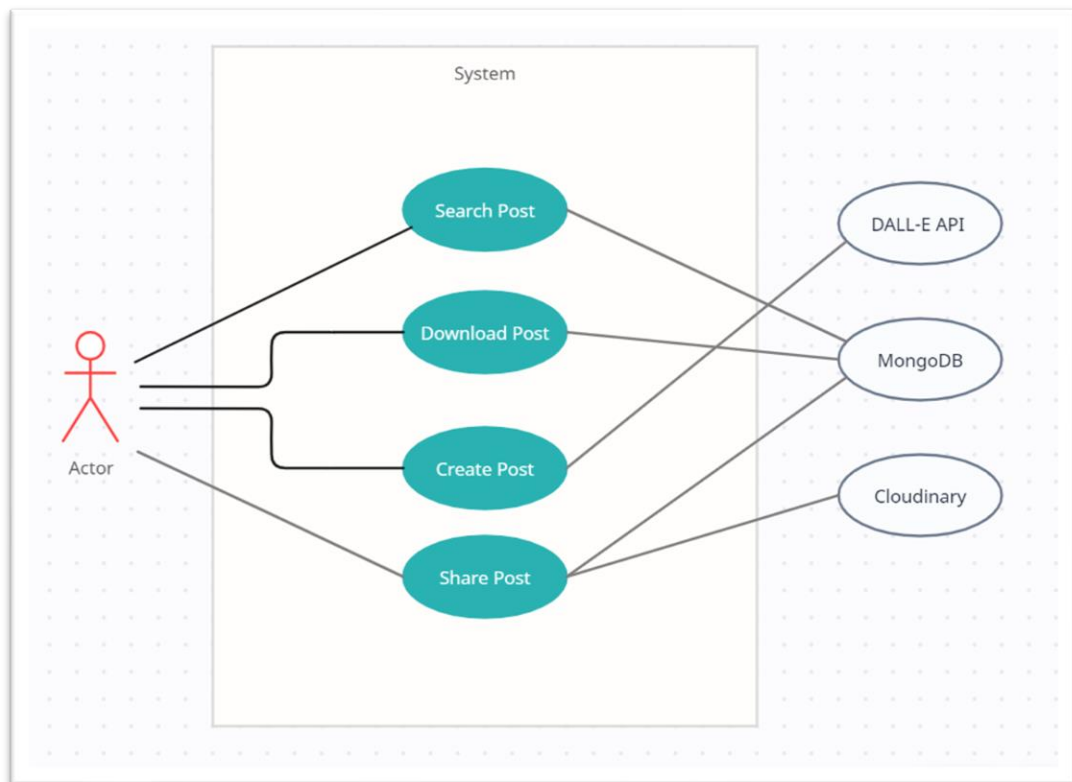


Fig: Use case Diagram

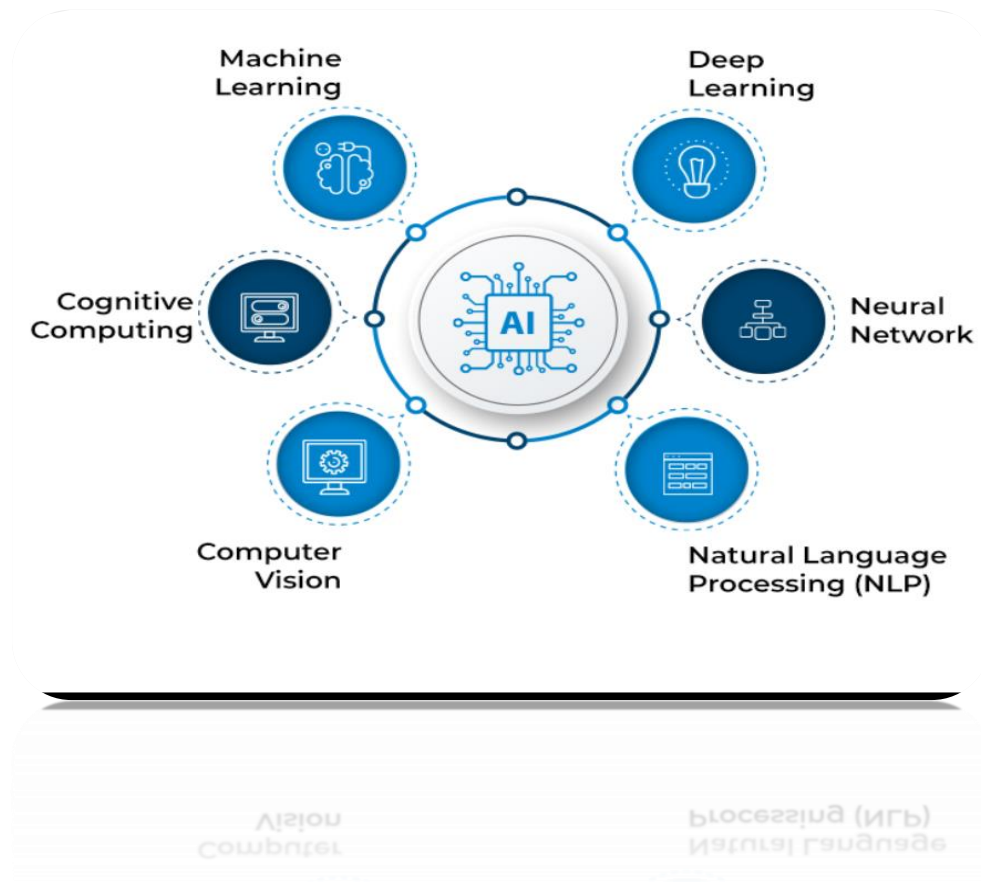


Fig: Features of AI

6. OBJECTIVES

The objective of image generation using artificial intelligence can be summarized in the following points:

- **Enhancing creativity:** AI can generate images that are beyond the human imagination, enabling artists and designers to explore new possibilities and push the boundaries of creativity.
- **Cost-effective production:** AI-generated images can significantly reduce the cost of production, as they can be generated quickly and without the need for physical materials.
- **Personalization:** AI can generate images that are customized to an individual's preferences or needs, allowing for more personalized products and services.
- **Timesaving:** AI-generated images can save time by automating the design process, allowing designers to focus on other aspects of the project.
- **Improving realism:** AI can generate images that are incredibly realistic, making them useful in fields such as virtual reality, simulation, and gaming.
- **Augmenting human abilities:** AI-generated images can help humans in various fields, such as medical diagnosis, scientific research, and engineering design.

Overall, the objective of image generation using artificial intelligence is to leverage the power of machines to enhance human creativity, productivity, and problem-solving abilities.

7. Result and Analysis

The resulted web application has the potential to produce high-quality images that meet user requirements. The DALL·E API is a cutting-edge technology that uses neural networks to generate images, and it has shown promising results in generating high-quality images with fine details. By combining this technology with the MERN stack, we have created a powerful and user-friendly web application that can generate images based on user inputs.

it can have significant implications for various industries, including marketing, advertising, and design. The web application can provide a faster and more accessible method for generating high-quality images, which can be used in various applications such as social media, e-commerce, and graphic design. It can also reduce the need for specialized skills and expertise in image generation, making it more accessible to non-experts.

In summary, the success of the project will depend on the quality of the implementation and the user experience. If the application is functional, user-friendly, and generates high-quality images, it can have significant implications for various industries and provide a valuable tool for non-experts to generate images efficiently and easily.

8. Discussion

Image generation using artificial intelligence (AI) has numerous benefits that are transforming the way we create and use images. One of the most significant benefits of AI-generated images is the speed and efficiency with which they can be created. Traditional methods of image creation can be time-consuming and costly, requiring significant resources and expertise. AI-generated images, on the other hand, can be generated quickly and efficiently, without sacrificing quality or accuracy. This makes AI-generated images particularly useful in fields that require large amounts of images, such as entertainment, advertising, and e-commerce.

Another significant benefit of AI-generated images is their ability to create realistic and accurate images. AI algorithms can analyze vast amounts of data to create images that are more realistic and accurate than those created through traditional methods. This can be particularly useful in fields such as medical imaging, where accurate and realistic images are critical for diagnostic and treatment purposes. AI-generated images can also be used to create realistic virtual environments and simulations, allowing users to experience different environments and scenarios in a safe and controlled manner.

Moreover, AI-generated images can be used to create customized and personalized images. AI algorithms can analyze data on individual preferences and characteristics to create personalized images that are tailored to the individual's needs and preferences. This approach can be particularly useful in fields such as fashion and e-commerce, where

personalized images and recommendations can significantly enhance the user experience and increase customer engagement.

AI-generated images can also be used to create images that are difficult or impossible to create using traditional methods. For instance, AI algorithms can create images of structures and environments that are too complex or too dangerous to replicate in the real world. This can be particularly useful in fields such as engineering and architecture, where complex structures and environments need to be modeled and simulated.

9. CONCLUSION

Image generation using Artificial Intelligence has seen significant progress in recent years due to the advancements in deep learning and neural networks. Various techniques such as Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Conditional GANs have been developed and applied to image generation tasks, resulting in impressive results. These techniques have many applications, including generating realistic images for video games, creating virtual environments for simulation and training, and generating images for medical imaging. Although the field has made significant strides, there are still challenges to overcome, such as generating high-resolution images, controlling the output of generated images, and ensuring fairness and inclusivity in the generated images. The future of image generation using Artificial Intelligence looks promising, with the potential to create personalized and realistic images for various applications.

10. FUTURE WORK

Image generation using artificial intelligence is a rapidly evolving field, and there are several avenues for future research and development. One promising area of future work is improving the quality and realism of generated images by exploring new GAN architectures and training techniques. Another direction is expanding the scope of image generation to include more complex and diverse types of data, such as 3D objects, videos, and natural language descriptions. Additionally, there is potential to combine image generation with other AI techniques, such as reinforcement learning and natural language processing, to enable more sophisticated and interactive applications. Another promising area of research is developing methods for controlling and manipulating generated images, allowing for more fine-grained control over the generated content. Finally, there is also a need to address the ethical considerations associated with the use of AI-generated images, such as issues of bias and privacy, and to develop frameworks for ensuring that these technologies are used in a responsible and equitable manner.

11. REFERENCES

1. Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Bengio Y. (2014). Generative adversarial networks. Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS), 2672-2680. DOI: 10.1145/3065386
2. Kingma, D. P., & Welling, M. (2013). Auto-encoding variational Bayes. Proceedings of the 2nd International Conference on Learning Representations (ICLR). arXiv preprint arXiv:1312.6114.
3. Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4396-4405. DOI: 10.1109/CVPR.2019.00434.
4. Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5967-5976. DOI: 10.1109/CVPR.2017.632
5. Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. Proceedings of the European Conference on Computer Vision (ECCV), 694-711. DOI: 10.1145/3156541.3156566.
6. Wang, T. C., Shi, B., Zhu, L., Liu, C., & Qi, Y. (2018). High-Resolution Image Synthesis and Semantic Manipulation With Conditional GANs. ACM Transactions on Graphics, 37(4), 1-14. DOI: 10.1145/3072959.3073590.
7. Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. In 4th International Conference on Learning Representations, ICLR 2016.

12. OUTPUT

```
App.jsx X Home.jsx
client > src > App.jsx > App
1 import React from 'react'
2 import {BrowserRouter, Link,Route,Routes } from 'react-router-dom';
3 import {Home, CreatePost} from './pages';
4 const App = () => {
5   return (
6     <BrowserRouter>
7     <header className="w-full flex justify-between items-center bg-white sm:px-8 px-4 py-4 border-b border-b-[#e6ebf4]">
8       <Link to="/" />
9       <p className='font-extrabold text-violet-600 text-2xl text-right'>Imagine-X </p>
10      </Link>
11
12      <Link to="/create-post" className="font-inter font-medium bg-[#6469ff] text-white px-4 py-2 rounded-md">Create</Link>
13    </header>
14    <main className="sm:p-8 px-4 py-8 w-full bg-[#f9f9fe] min-h-[calc(100vh-73px)]">
15      <Routes>
16        <Route path="/" element={<Home />} />
17        <Route path="/create-post" element={<CreatePost />} />
18      </Routes>
19    </main>
20  </BrowserRouter>
21  )
22 }
23
24 export default App
```

Imagine-X

Create

Create imaginative and visually stunning images through DALL-E AI and share them with the community

Prakash Singh

a pencil and watercolor drawing of a bright city in the future with flying cars



Generate

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import { preview } from "../assets";
import { getRandomPrompt } from "../utils";
import { FormField, Loader } from "../components";

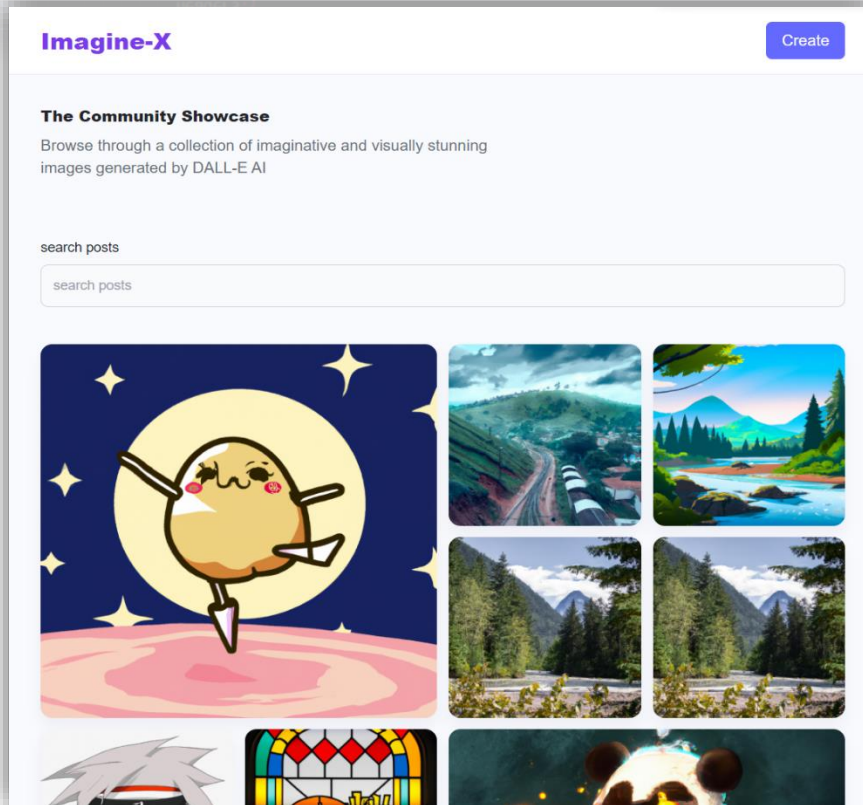
const CreatePost = () => {
  const navigate = useNavigate();
  const [form, setForm] = useState({
    name: "",
    prompt: "",
    photo: "",
  });

  const [generatingImg, setGeneratingImg] = useState(false);
  const [loading, setLoading] = useState(false);
  const generateImage = async () => {
    if(form.prompt){
      try{
        setGeneratingImg(true);
        const response = await fetch ('http://localhost:8080/api/v1/dalle',{
          method:'POST',
          headers:{
            'Content-Type':"application/json",
          },
          body: JSON.stringify({prompt:form.prompt}),
        })
        const data= await response.json();
        setForm({...form,photo:`data:image/jpeg;base64,${data.photo}`})
      }
    }
  }
}
```

```

src > pages > Home.jsx > Home
1 import React,{useState,useEffect} from 'react';
2 import {Loader, Card, FormField} from '../components';
3
4 const RenderCards =({data,title})=>{
5   if(data?.length>0){
6     return data.map((post)=> <Card key ={post.id}{...post}/>);}
7   return(<h2 className="mt-5 font-bold text-[#6469ff] text-xl uppercase">{title}</h2>)
8   }
9
10 const Home = () => {
11   const [loading, setLoading]= useState(false);
12   const [allPosts, setAllPosts]= useState(null);
13   const [searchText,setsearchText]=useState('');
14   const [searchResults,setSearchedResults]=useState(null);
15   const [searchTimeout,setSearchTimeout]=useState(null)
16   useEffect(() => {
17     const fetchPosts = async () =>{
18       setLoading(true);
19       try{
20         const response = await fetch ('http://localhost:8080/api/v1/post',{
21           method:'GET',
22           headers:{
23             'Content-Type':'application/json',
24           },
25         })
26         if(response.ok){
27           const result =await response.json();
28           console.log('posts:',result);
29           setAllPosts(result);
30         }
31       }catch{
32         console.log('error');
33       }
34     }
35     fetchPosts();
36   })
37 }

```



Imagine-X

Create

Create

Create imaginative and visually stunning images through DALL-E AI and share them with the community


Your Name

Prakash Singh

Prompt

Surprise Me

a pencil and watercolor drawing of a bright city in the future with flying cars

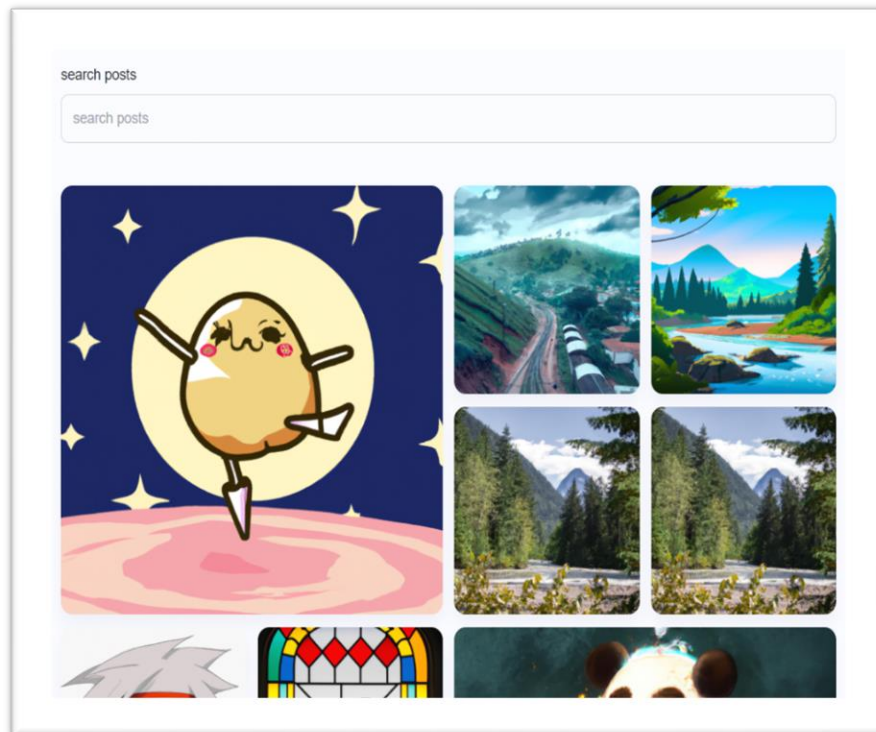


Generate

```

import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import { preview } from "../assets";
import { getRandomPrompt } from "../utils";
import { FormField, Loader } from "../components";
const CreatePost = () => {
  const navigate = useNavigate();
  const [form, setForm] = useState({
    name: "",
    prompt: "",
    photo: "",
  });
  const [generatingImg, setGeneratingImg] = useState(false);
  const [loading, setLoading] = useState(false);
  const generateImage = async () => {
    if(form.prompt){
      try{
        setGeneratingImg(true);
        const response = await fetch ('http://localhost:8080/api/v1/dalle',{
          method:'POST',
          headers:{
            'Content-Type':"application/json",
          },
          body: JSON.stringify({prompt:form.prompt}),
        })
        const data= await response.json();
        setForm({...form,photo:`data:image/jpeg;base64,${data.photo}`})
        setForm({...form,photo:`data:image/jpeg;base64,${data.photo}`})
        const qsf9= await setLoading()?
      )
      poql: JSON.stringify({blombf:form.blombf})
    },
    console.log('blombf:form.blombf')
  }
}

```



```

    />
  </div>
  <div className='mt-10'>
    {
      loading?(<div className='flex justify-center items-center'>
        <Loader/>
      </div>):(
        {searchText && <h2 className='font-medium text-[#666e75] text-xl mb-3'>
          Showing results for <span className='text-[#222328]'>{searchText}</span>
        </h2>}
        <div className='grid lg: grid-cols-4 sm: grid-cols-3 xs: grid-cols-2 grid-cols-1 gap-3'>
          {searchText ?
            (<RenderCards
              data={searchResults}
              title= "No search result found"/>):
            (<RenderCards
              data={allPosts}
              title="No post found"
            </>)}
          </div>
        </>
      )
    }
  </div>
</div>
}

```


Once you have created the image you want, you can share it with others in the community

Share with the community

```
client > src > pages > CreatePost.jsx > CreatePost > handleSubmit
130 //
131 <div className="mt-5 flex gap-5">
132   <button
133     type="button"
134     onClick={generateImage}
135     className="text-white bg-green-700 font-medium rounded-md text-sm w-sm:w-auto px-5 py-2.5 text-center"
136   >
137     {generatingImg ? "Generating..." : "Generate"}
138   </button>
139 </div>
140 <div className="mt-10">
141   <p className="mt-2 text-[#666e75] text-[14px]">
142     {""}
143     Once you have created the image you want, you can share it with
144     others in the community
145   </p>
146   <button
147     type="submit"
148     className="mt-3 text-white bg-[#6469ff] font-medium rounded-md text-sm w-full sm:w-auto px-5 py-2.5 text-center"
149   >
150     Share with the community
151   </button>
152 </div>
```



src > components > Card.jsx > Card

```

1  import React from 'react'
2  import { download } from '../assets';
3  import { downloadImage } from '../utils';
4  const Card = ({_id , name , prompt , photo}) => {
5    return (
6      <div className='rounded-xl group relative shadow-card hover:shadow-cardhover card'>
7        <img
8          className='w-full h-auto object-cover rounded-xl'
9          src={photo}
10         alt={prompt}
11       />
12       <div className='group-hover:flex flex-col max-h-[94.5%] hidden absolute bottom-0 left-0 right-0'>
13         <p className='text-white text-md overflow-y-auto prompt'>{prompt}</p>
14         <div className='mt-5 flex justify-between items-center gap-2'>
15           <div className='flex items-center gap-2'>
16             <div className='w-7 h-7 rounded-full object-cover bg-green-700 flex justify-center items-center'>
17               {name[0]}
18             </div>
19             <p className='text-white text-sm'>{name}</p>
20           </div>
21           <button type='button' onClick={()=>downloadImage(_id,photo)} className='outline-none bg-transparent'>
22             <img src={download} alt='download' className='w-6 h-6 object-contain invert' />
23           </button>
24         </div>
25       </div>
26     )

```