

# Image Manipulation Web Application: A Next JS Implementation

Assistant Professor Ms. Priyanka Kapila, Mr. Mayank Kumar Grade,  
Mr. Shubham, Mr. Himanshu Shahoo

CSE Department  
HMR Institute of Technology and Management, GGSIPU, Delhi

**Abstract-** The enhancement in web technologies has contributed to the evolution of web applications that are very dynamic and engaging. This research work focuses on the creation of an online image editing application that is based on cloud infrastructure and modern web layouts/development tools such as Next.js, TailwindCSS, and Cloudinary's APIs, among other resources, to deliver advanced image editing features. The application incorporates Clerk to allow users to create login accounts and easily register, while data is managed using MongoDB to facilitate the security of users and edited pictures across several devices. Necessary and basic features such as object removal, editing backgrounds, recoloring pictures, restoring, and changing the size of images are handled within the cloud and therefore benefit the functionality of the application and users as well. In addition, a contact form utilizing EmailJS has been integrated to enable communication with users. This research work highlights the legitimacy of cloud-based solutions as well as their expanded geographic reach in catering to an advanced user experience within image editing applications, thus supporting the growth of cloud computing and web technology.

**Index Terms-** Web Application, Image Manipulation, Cloud-Based Services, UI/UX Design

## I. INTRODUCTION

The increasing popularity of web applications has changed the perception of clients towards computerized tools for carrying out different assignments such as photo editing. The use of applications that run in the cloud is attractive to the end user as well as corporations because of their availability, scalability, and effectiveness. It is in this light that the attention of many has been drawn towards the creation of web-based image processing applications. This research work focuses on the design and development of a web-based image editing application that is built on the cloud framework using Next.js, TailwindCSS, and Cloudinary APIs. The application allows for user login management, enables real-time editing from different locations, and uses MongoDB as a medium for storing user information.

### 1. Background and Motivation

Earlier tools for image processing were restricted to desktop applications, but thanks to technological developments, they have been integrated into web-based applications. Historically, images could only be edited with heavy computing software that most people could not afford, thus limiting photo editing to the rich. Nevertheless, with the rise of cloud services, image editing undertakings have veered towards the web; hence, users can edit photos online without requiring much local hardware. Next.js, the go-to framework for building

React applications, has demonstrated its viability in creating interactive web applications due to its ability for server-side rendering (SSR) and static site generation (SSG), which enhance performance and search engine optimization (SEO) as well as user experience. When developers wish to progress in their work, the framework provides TailwindCSS, which helps in styling by enabling developers to design user interfaces in a short period of time. All of this, with Cloudinary, an image uploading and hosting platform, allows for performing image manipulation actions using REST APIs conveniently. The need for developing this project arose with the increasing popularity of web applications that allow simple and appealing image editing in real time. This application integrates Clerk for user restrictions and MongoDB as backend databases, thus allowing access to the application across devices and maintaining data integrity. User engagement is also improved as needed through the use of EmailJS, which provides functional contact forms embedded on the homepage.

### 2. Problem Statement

Despite the availability of numerous online image editing tools, only a handful have all the requisite features and are user-friendly enough to provide fast and high-quality results. Most of the existing platforms either offer their services at a premium level or have limited functionality, making them difficult for a layperson to use. Thus, finding an equilibrium between user interface and performance has been a challenge

for developers of these websites. Common problems include long waiting periods, the inability to allow multiple edits at the same time, or poor data management systems that cause frustration and hinder access to one's information from other devices. This undertaking aims at addressing such issues by conceptualizing and developing an online system based on the latest technologies to enhance effectiveness and offer rich functionalities. The fundamental objective of the entire project is to build an intuitive and advanced image editing application that allows cross-device usage without degrading the functionalities of the application.

### 3. Objectives

The main objectives of the project are specified as follows: Create a Next.js-based application for an image manipulation solution that is hosted in the cloud. Incorporate TailwindCSS so that users can enjoy beautiful graphics that are responsive to their different screens. Use Cloudinary APIs to enable other image manipulation features like, but not limited to, background removal, object recoloring, image restoration, and image cropping. Provide a Clerk-based authentication system to manage the login and account registration processes securely. Install a MongoDB database as the server-side storage for user data and project files, allowing users to access their work from separate devices. Introduce EmailJS in order to allow easy implementation of the contact form, enabling users to communicate.

### 4. Overview of Modern Technologies Used

Next.js has become very popular due to its support for server-side rendering, which has great advantages in performance and SEO. These considerations are paramount when developing responsive applications that are resource-hungry when it comes to rendering images. For the reasons above, Next.js does not require any third-party libraries for routing or server-side capabilities, which makes the development process easier and loading times faster. TailwindCSS makes the designer's work easier by bringing a utility-first way of designing that allows developers to add styles directly within an HTML tag. This cuts down the time spent designing with custom CSS, as well as ensuring that the look of the application is uniform across all its many pages. Cloudinary provides an advanced set of APIs that are built for convenient global image and video management. Cloudinary's platform allows the application to perform heavy processing tasks in the cloud instead of on the user's device. This includes features like smart formatting, responsiveness of images, and adaptive cropping. The NoSQL database MongoDB was selected due to its agility and its ability to work hand in hand with JavaScript-based environments like Node.js. Its schemaless nature enables it to efficiently store user data in the form of images, without being limited to a certain structure. User management functionality was achieved through the implementation of Clerk. Clerk is suitable for the modern environment as its use is geared towards social login

and supports advanced multi-factor authentication features. Analyzing the anticipated functionality of the project, EmailJS provides a sufficient solution to this challenge since it permits sending emails from the client side without the need for a server to handle the emailing process.

### 5. Contributions of this Work

This research work adds to the body of knowledge in a number of ways and presents a working implementation that combines different modern web technologies. The key contributions include: Creation of a sophisticated online image manipulation application that, unlike all existing solutions, operates in a web browser without the need for local or cloud computing power. Clerk was integrated so that secure authentication of users is enforced, thus enhancing user data security. Adoption of MongoDB to ensure efficient data storage and access, allowing data to be accessed from any device. Incorporation of EmailJS for effective and efficient communication with users, thus allowing interaction with the developers to enhance the user experience.

## II. LITERATURE REVIEW

The rapid evolution of the web has brought forth new technologies, frameworks, libraries, and tools that have been developed for the purpose of optimizing performance, security, as well as the usability of web applications. This review integrates important research on the design and realization of a cloud-based server for image processing web applications, focusing on SEO, API testing, database systems, and security systems. The obtained results offer guidelines for web development, specifically for systems that need heavy load operation in the back-end, with optimal interaction on the front-end and high performance over a range of devices.

One of the important aspects of web application development is enhancing the visibility and engagement of users on websites. Sharma et al. [1], note that website ranking improvement strategies, known as SEO strategies, are very helpful. The research dealt with such concepts as keyword research, indexing, the two ways of optimization—on-page and off-page and finally concluded that such approaches help a great deal in improving engagement levels. In the same depth of understanding, Kowalczyk et al. [2], researched the problem of SEO in single-page applications (SPAs) and the challenges posed by client-side rendering to such an endeavor. They present hybrid rendering techniques as a solution to this problem and, by doing so, aid SEO performance strategies in modern applications such as those built with the Next.js framework.

In the architecture of contemporary web applications, another crucial factor is the selection of database technology. Hodijah et al. [3], in their research work, research the features of the document-oriented NoSQL database MongoDB. Their study

has shown that the embedded data model of this NoSQL database works particularly well for applications with performance-based information retrieval for large amounts of data. By furthering this comparison, Matallah et al. [4], analyze the benefits of using both MySQL and MongoDB databases, indicating that mobility and efficiency perfectly characterize the operation of MongoDB as it processes dynamic data. Therefore, it is very appropriate for applications, like web ones, that need fast storage and retrieval of such data, like images.

The building of web applications also depends on Asynchronous Programming Interfaces in a significant way. Lamothe et al. [5], conducted research on the evolution of APIs, noting the importance of changing while keeping the existing works. API integration of third-party ones, especially Cloudinary, a cloud-based API for image processing and hosting, as in this project, is very much related to this observation. Ehsan et al. [6], provide more information on the issues found in utilizing RESTful services, particularly the API integration, which may call for advanced testing approaches, such as automated unit testing. In the case of this study, it is necessary to address the concerns relating to the Cloudinary API that is used at the back end of the web application.

Security over the web plays a very important role in the process of web development, more so when user data and databases are in consideration. The work is attributed to Khairuddin et al. [7], entitled “Intelligent Face Detection and Recognition for Cloud-based Security Systems,” provides some information about cloud-based security systems equipped with intelligent face detection and recognition capabilities. The findings of this research demonstrate the need for cloud services such as Cloudinary and Dropbox for the purposes of storing and retrieving data for the web application, which requires a secure way of keeping images and accessing them from different devices.

The user experience is one of the prime elements for present-day web applications. Naiki et al. [8], present a new graphical front-end apparatus that is additional in React.js, a framework that allows building dynamic web pages using state-transition diagrams. As a result of this process, developers have to do less work themselves and can design dynamic components of web pages more beautifully.

This concept sits well with Next.js, which enables one to engage in web development by providing, for example, server-side rendering and static site generation, among other capabilities. Also, Keshari et al. [9] suggest the adoption of the component-based architecture of React.js together with the virtual DOM for better performance and code reusability. These additional benefits of Next.js include the server-side rendering that enhances SEO, as explained by Patel [10], who

explains Next.js with its features Incremental Static Regeneration (ISR), image optimization, and small JavaScript bundle size in the context of web performance enhancement.

### Conclusion

The literature examines modern web Application Development as a complex process requiring effective search engine optimization approaches, complex databases, advanced application programming interfaces, and cloud technology. The research justifies the choice of backing storage by employing MongoDB, images processing with Cloudinary and frontend development with NextJS to utilize the architecture in the best possible way in relation to performance, scalability and security. These aspects play a pivotal role in the process of designing a web based image manipulation application, which encompasses the knowledge on how to achieve a high level structured, operable and secure final product together with the relevant tools.

## III. METHODOLOGY

The creation of the cloud-based image manipulation web application has been in accordance with the current technologies and frameworks, capable of and building an efficient, expandable and attractive platform. This part describes the development tools, programming languages and implementation techniques applied for the purposes of this project.

### 1. Technology Stack

#### Framework: Next.js

Because of its server-rendered (SSR) architecture and static site generation capabilities, which enhances performance with no effort required on search engine optimization, the team deployed the Next.js framework which incorporates the react framework for building the applications. This way, pages were easier to load, and the general experience was far better than the previous one.

#### Styling Framework: TailwindCSS

A utility-first CSS framework, TailwindCSS was employed for building a nice user interface that is mobile-friendly and easy on the eye. This option allowed for quick development and ensured uniform designs across the entire application thus enhancing coordination and consistency.

#### Cloud Image Processing: Cloudinary APIs

The application included advanced capabilities of Cloudinary for manipulation such as removing objects, changing backgrounds, resizing and restoring images among others. Thanks to the extensive library of APIs offered by Cloudinary, it was easy to create an application with backend handling images in a more sophisticated way.

### Database Management: MongoDB

MongoDB was chosen to be the database where users and edited pictures would be stored. This made it ideal for use because it was easy to implement its design for locating information within machines for different users, thus making access across devices for users possible.

### User Registration: Clerk

The coordinator's decision was to include clerks in the system so that user registration and login management could be undertaken. This service was useful since it offered complete solutions for user verification, like social login and two-way login features, thus maximizing security with minimal coding.

### Email Service : EmailJS

With the implementation of EmailJS, it was made possible for the users of the web application to send messages through the application, thus enabling the functionality of the contact UIs. This ensured that all communication from the users was handled in an automated and secure manner.

## 2. Development Process

**Initial Planning and Wireframing:** The project began with feature-rich precedes, which assisted in creating user flows & wireframes of the application. Mock-ups in Figma were done with the aim of making the design of the interface more user-friendly.

**Front-End Development:** The front-end was developed with Next.js and styled with the TailwindCSS library. This approach adopted the use of a modular system to achieve reusability and maintainability of the components. Moreover, the TailwindCSS utilities also enhanced the speed of implementation of the responsive design, which supports layouts of various proportions.

**API Integration:** The Next.js platform integrated RESTful Cloudinary APIs. Every operational feature (e.g., background removal as well as imaging) was encased in a functional module for easy enhancement/modification in the future.

**Database and Backend:** Consequently, the database integration related to the achieving network services for the project was also accomplished through Next.js API routes. The main database was MongoDB, which held information about system users and their uploaded pictures, as well as other user data for easy access and use across a range of devices.

**Authentication Setup:** In order to facilitate the user's common custom login page and user registration functionalities, Clerk system was integrated into the modules. This scenario was important not only for user management but also for increasing the application's overall security level.

**Email Service Craig:** The EmailJS service was integrated into the home page SDK – a contact box after which user messages were transmitted to a contact email, improving the support capability of the service.

## 3. Perspectives on Truly Improving and Resolving the Issues

**Rate Limiting:** Cloudinary imposed limits on how frequently a user can call the API within a time span. Improving this, by tirelessly working on techniques to reduce unnecessary API calls and recover from failures, was one of the challenges author faced.

**Adaptive Interfaces:** The application posed the challenge of creating an interface in which the layout would look the same and remain functional on any device, which required the extensive application of the responsive utilities of TailwindCSS.

**Control Access:** For the purpose of user data control, it was possible to implement Clerk's authentication service, which helped control token usage through encryption.

## 4. Tools

As far as development environments are concerned, Author would like to point out Visual Studio Code for the writing of the code, GitHub for the version control and Vercel for the deployment of the code.

**Languages:** TypeScript is utilized for the application while HTML and CSS are for the markup and design purposes respectively.

**Methods:** API operations in this application have been performed using axios. Server-side assistance has been done with the help of Node.js and Mongoose has been used in coupling with the Mongo database to provide a database system.

In the development of the application, emphasis has been placed on these objectives: productivity, security, and ease of use. Every tool and every technique has been used with the goal of developing serious commercial-grade solutions.

**Adaptive Interfaces:** The application posed the challenge of creating an interface in which the layout would look the same and remain functional on any device, which required the extensive application of the responsive utilities of TailwindCSS.

**Control Access:** For the purpose of user data control, it was possible to implement Clerk's authentication service, which helped control token usage through encryption.

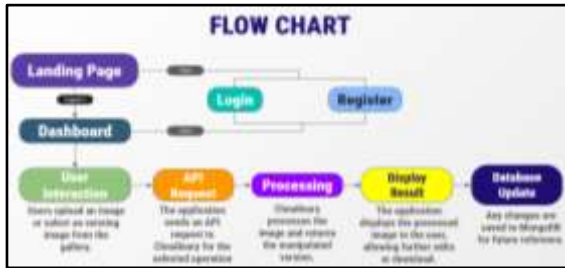


Fig.1 Flow Chart of Work Flow

## IV. RESULTS

The Results section provides the findings pertaining to the development process along with some screenshots and evaluation of the web application’s features and performance. These images and text emphasize the architectural integration of Next JS, Tailwind CSS and Cloudinary API towards achieving a fully functional cloud based image editing service.

### 1. User Interface Design and Responsiveness

The UI was designed and constructed using TailwindCSS which guaranteed an adaptable design on different screen sizes such as that of a desktop or a tablet or a phone. Below is an example screenshot of the main dashboard:

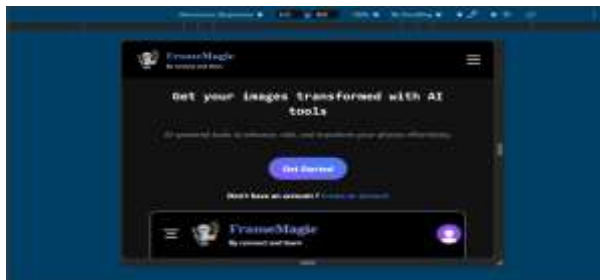


Fig.2 Site Responsiveness

### 2. Steps for User Accessing the Portal

With the integration of Clerk, the user login and registration experiences have been enhanced. Below are screenshots to illustrate: The registration page incorporates social login options. The rigorous login screen after email verification.



Fig.3 User Registration using Clerk

### 3. Imaging Related Features

The following aspects of image manipulation were handled using Cloudinary APIs:

**Remove Objects and Adjust Background:** An image adjustment feature that lets the user select certain areas of the image and remove or change them without a trace.

**Color Correction and Image Enhancement:** These tools allowed users to change the color depth, contrast, and even restore old images for clarity enhancement.

**Scaling Images:** Scaling images with fixed aspect ratios or custom sizes.

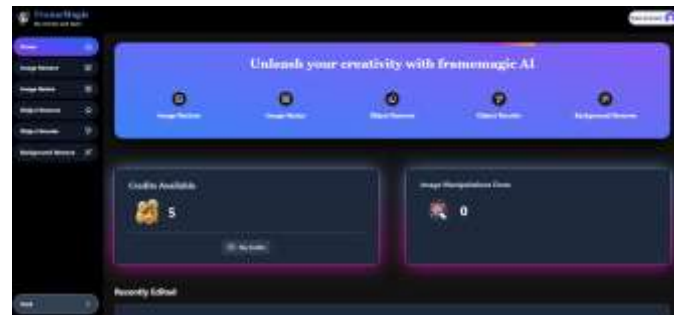


Fig.4 Image Manipulation Features

### 4. Data Management and Accessibility Using

MongoDB as a backend database, user-edited images and information were adequately saved within the application, giving access on any device. This made it easier for users as they could access their earlier edits regardless of the device.



Fig.5 MongoDB DataBase

### 5. Performance Metrics

Load time analysis: Next.js augmented other metrics like time-to-first-byte, which were significantly faster compared to the previous versions and thus made the application efficient and responsive. Page load times improved significantly.

API response times: Image manipulations were performed under various load conditions and the outcome was satisfactory.



Fig.6 Web Analytics

## 6. User Feedback and Testing

The beta testing results pointed out the intuitive design of the app, as well as its effective operation on different devices. Users', in particular, were fond of the latter's logging in modes and the editing tools offered by Cloudinary were very fast.

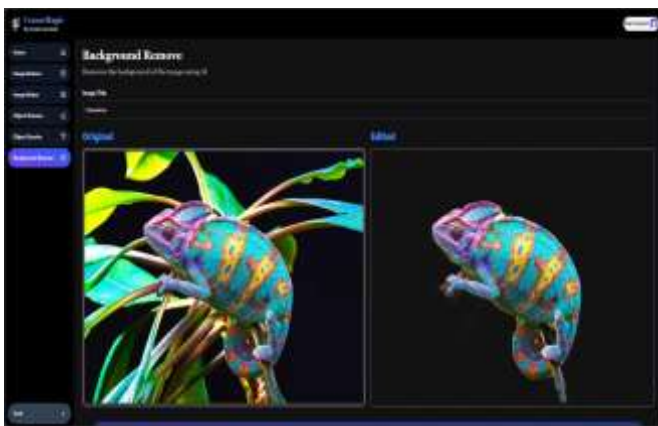


Fig.7 User Registration

## V. CONCLUSION

The use of Next.js, TailwindCSS, and Cloudinary APIs in the creation of a website for manipulation of images in the cloud has positively proven how web applications of high scalability, high responsiveness, and high usability can be built. This project tackled the issues of contemporary web development approaches, which stress the focus on the user, the need for access across devices, and the use of powerful cloud services.

The utilization of Next.js turned out to be very efficient, enhancing the performance of the application in comparison to previous works outlined in the Literature Review because of the server-side rendering technique. The application of TailwindCSS further made the UI design process easier by making it possible to design the UI responsively, yet with the freedom to customize as desired. The availability of the API from Cloudinary created an all-encompassing backend for editing images, eliminating the drawbacks that could be posed by local stand-alone libraries or basic third-party libraries.

The system of authentication provided by Clerks services offered safe as well as straightforward access, which fortified

the users' confidence in the application. Recruitment solutions made the storage of data seamless, as users were able to carry the data wherever they go, owing to the fact that it is cloud-based, meaning it offers the benefits of storage over the geographic dispersal of the places accessing the service.

The successful deployment and operation of this web application indicates that it is possible to employ new web technologies with such efficiency in order to satisfy the expectations of users in terms of powerful and rich image editing features. Many difficulties have been met and resolved in the process of developing the application, including issues related to the necessity to comply with the rate limits set for the use of the provided API and also implementing schemes aimed at improving the response rate of the said application. These problems were, as always, solved by way of some strategic coding techniques and also by the versatile nature of the technology stack selected.

## Future Scope

Over the years, software solutions have dramatically evolved, and so have their use. The latest version of the web application for manipulation of images in the cloud presented is way more encouraging in terms of user engagement and editing skills. But there is enormous room for improvement and feature enhancement in the future. This part describes the views of the applications that optimize/improve the utility of the application and the experience of the users of the application.

## AI Based Edit Tools

- **Advanced Image Restoration:** The addition of advanced AI models in the application for complex image restoration processes, such as portrait image upscaling or even colorizing previously black-and-white images, will be a game changer for the application.
- **Clever Background Changes:** Using smart background machine learning algorithms to change backgrounds without them feeling forced or fake to the user.

## User Personalization

- **Possibility for Customizing Presets and Filters:** Improvement of the application's symmetric feature by adding the possibility for a user to use their own art, which motivates and enhances photo editing, and keep it.
- **Analytics Within User Dashboards:** Development of a feature that allows a user to view a specific aspect or several indicators and their statistics in the context of a user's editing activity and the history attached to it.

## 3. Enhanced Authentication Options

- **Biometric Devices in Login Applications:** An implementation of accessing the application with a

fingerprint or face scan, making it easier and less complicated to use the application.

- **Integration of additional services using OAuth 2.0:** In addition, it is sought to widen the reach by providing social logins using Twitter, LinkedIn, Github, Google and Facebook.

#### 4. Collaboration and Social Sharing

- **Real-Time Collaboration:** In this scenario, anyone can work on a particular file, as editing rights are provided to more than one user.
- **Integrated Social Sharing:** Elements like sharing to Instagram, Facebook, Pinterest, among others, will increase the interaction of users and the sharing of created content.

#### 5. Enhancements in Scalability and Effective Performance

- **Serverless Functions for Scalability:** Scaling serverless computing further for effective management of heavy image processing operations, especially during peak hours.
- **Caching Strategies:** Adopting advanced caching strategies for better load times and less pressure on the servers.

#### 6. Editing Improvement

- **3D Object Handling:** Looking into allowing users to work with 3D objects for complex image editing projects.
- **Vector Graphic Support:** Allowing users to edit vector graphics in order to broaden the scope of the application to graphic designers.

#### 7. Accessibility Improvements

- **Enhanced Accessibility:** This entails the application of assistive capabilities such as screen reading services, high contrast, operating in a keyboard-only mode, among others.
- **Content Localization:** Improving the application in such a way that more languages can be supported, therefore ensuring that all non-English-speaking users across the world will be able to utilize the application.

#### 8. Monetization and Subscription Models

- **Freemium Model:** Enabling extra features of advanced tools while retaining a free leg of basic functionalities, thus ensuring an avenue of making money.
- **Subscription Tiers:** Various subscription plans must be created to suit the needs of a recreational user and a serious design user.

The adoption of these future improvements, however, would enable the application to take the leading position in web

image editing software for the benefit of its users and in the current technological climate.

## REFERENCES

1. S. Sharma, S. P. Panda, and S. Verma, "Role and analysis of various SEO strategies to improve website ranking," in Proc. 2022 Int. Conf. Mach. Learn., Big Data, Cloud Parallel Comput. (COM-IT-CON), 2022, pp. 1-5, May 26-27, 2022, <https://doi.org/10.1109/COM-IT-CON54601.2022.9850597>.
2. K. Kowalczyk and T. Szandala, "Enhancing SEO in single-page web applications in contrast with multi-page applications," IEEE Access, vol. 12, pp. 11597-11614, Jan. 18, 2024, <https://doi.org/10.1109/ACCESS.2024.3355740>.
3. A. Hodijah and U. T. Setijohatmo, "Experimental evaluation of relationships model between documents in MongoDB," Adv. Eng. Res., vol. 2020, Art. no. 087, Dec. 22, 2020, <https://doi.org/10.2991/aer.k.201221.087>.
4. H. Matallah, A. B. Belkaid, G. Belalem, and K. Bouamrane, "Comparative study between the MySQL relational database and the MongoDB NoSQL database," Int. J. Softw. Sci. Comput. Intell., vol. 13, no. 3, pp. 38-63, Jun. 2021, <http://doi.org/10.4018/IJSSCI.2021070104>.
5. M. Lamothe, Y.-G. Guéhéneuc, and W. Shang, "A systematic review of API evolution literature," ACM Comput. Surv., vol. 54, no. 8, pp. 1-36, Oct. 2021, <http://doi.org/10.1145/3470133>.
6. A. Ehsan, M. A. M. E. Abuhaliqa, C. Catal, and D. Mishra, "RESTful API testing methodologies: Rationale, challenges, and solution directions," Appl. Sci., vol. 12, no. 9, Art. no. 4369, Apr. 2022, <http://doi.org/10.3390/app12094369>.
7. M. H. Khairuddin, S. Shahbudin, and M. M. Kassim, "A smart building security system with intelligent face detection and recognition," IOP Conf. Ser. Mater. Sci. Eng., vol. 1176, no. 1, Art. no. 012030, Aug. 2021, <http://doi.org/10.1088/1757-899X/1176/1/012030>.
8. S. Naiki, M. Kohana, M. Niibori, S. Okamoto, Y. Ohtaki, and M. Kamada, "A graphical front-end interface for React.js considering state-transition diagrams," Int. J. Grid Utility Comput., vol. 13, no. 4, pp. 482-494, Oct. 2022. <https://doi.org/10.1504/IJGUC.2022.126169>.
9. P. Keshari, P. Maurya, P. Kumar, and A. Katiyar, "Web development using ReactJS," in Proc. 5th Int. Conf. Adv. Computing, Commun. Control Netw. (ICAC3N), 2023, pp. 1-5, Dec. 15-16, 2023, <https://doi.org/10.1109/ICAC3N60023.2023.10541743>.
10. V. Patel, "Analyzing the impact of Next.js on site performance and SEO," Int. J. Comput. Appl. Technol. Res., vol. 12, no. 10, pp. 24-27, Oct. 2023, <http://doi.org/10.7753/IJCATR1210.1004>.