# Retraction Notice

The Editor-in-Chief and the publisher have retracted this article, which was submitted as part of a guest-edited special section. An investigation uncovered evidence of compromised peer review and determined the paper is unrelated to the special section. The Editor and publisher no longer have confidence in the results and conclusions of the article.

SK disagrees with the retraction. KG, VS, and NS either did not respond or could not be reached.

# Cloud and deep learning-based image analyzer

**Sunil Kumar,* Kartik Gautam, Vatsal Singhal, and Nitin Sharma**
Amity University, Noida, Uttar Pradesh, India

**Abstract.** The purpose of the proposed cloud and deep learning-based image analyzer technique is to discuss various challenges with existing conventional methods of managing business images and why we should adopt a better approach. We propose a cloud-based solution, which is more secure, durable, robust, and much faster compared with conventional methods of business image management. This approach tries to rectify the issue of storing, searching, categorizing, and managing images by combining the benefits of cloud along with computer vision and image processing. We thoroughly discuss the idea along with detailed analysis of requirement gathering, feasibility study, and architecture designing. Furthermore, we discuss the implementation code, detailed comparison of different image processing algorithms, empirical analysis of this approach compared with other conventional approaches in terms of cost, accuracy of image processing algorithms, data entry time, searching time, etc. Finally, we emphasize the best approach and image processing algorithm chosen based on the practically proven results. © *2022 SPIE and IS&T* [DOI: 10.1117/1.JEI.32.2.021602]

**Keywords:** cloud; image processing; computer vision; business card; web application; Amazon Web Services.

Paper 220431SS received Apr. 29, 2022; accepted for publication Aug. 10, 2022; published online Sep. 19, 2022.



## 1 Introduction

The main objective of implementing the cloud and deep learning-based image analyzer technique is to remove the hassle of collecting and managing business images. The use case of cloud and deep learning-based image analyzer technique is mostly in organizations that deal with a lot of business images. In these organizations, as the number of images increases, it becomes more difficult to track, store, and search for physical images. Even if the organization uses a computer to store card data, it will create its own problems that are not limited to lengthy data entry processes, the problem of teaching computer skills, huge data storage, security issues, data loss in case of malfunctions, access to data in remote places, etc. To fix all these problems, we have found a solution that is to shift all the processes to the cloud and automate the process of data extraction from images using an application of computer vision and image processing. In this, a user accesses a website application hosted on cloud servers, captures and uploads a business image using the device, that captured image gets processed in the cloud, and gets stored in cloud storage services along with data extracted from it in the database. After that the user would be able to access all the data that he has stored by running a search query on the website. Shifting the conventional process to cloud-based offers several benefits, which include reduced fixed costs, scalability, flexibility, security, mobility, disaster recovery, etc. Computer vision and image processing implemented in the system also help reduce the time taken to store data as it does the job automatically.

There are several benefits of using this technique in place of the conventional method:

1. First, it is more secure, as it resides in the cloud. The security of hardware is completely managed by the cloud provider; on top of that it is also secure from any kind of malfunction or data loss issue.
2. It is more accessible because it resides in the cloud. It can be accessed from around the globe, with minimal issue.

*Address all correspondence to Sunil Kumar, skumar58@amity.edu

1017-9909/2022/$28.00 © 2022 SPIE and IS&T

3. It is easy to manage because it is on the cloud and is using services provided by another provider; hence, most services are managed by a service provider.
4. It is easy to collaborate, because it is based on the cloud; hence, it can be mutually used by multiple users at the same time to work on the same task.
5. It does not include any kind of fixed costs such as setting up servers, maintaining them, or running costs.
6. The computer vision and image processing used in the backend for processing images completely removes the dependency on entering data manually.

## 2 Problem Statement

File management has always presented challenges. One such problem is also related to management of business images. Many businesses still use business images for communication. This paper improves the conventional method of image management. Currently if you want to get someone's contact details you ask for their business card and keep it with you. Either you maintain a folder to store it physically or you scan and store business images with you on your local devices. This approach of storing and organizing images is tedious. Now suppose you are working with an organization, and you receive hundreds of business images per day. Imagine the hassle you would face in maintaining all those images, searching for a single card out of thousands or even millions. Here is an example of use case from a first person perspective. I am an HR professional at an organization. My work is to get in contact and meet at least hundred new people a day. Each time I meet someone I ask for their business card. To get in contact with them in the future without wasting work time, I upload all the images I get on this cloud-based business card manager website/application. Now whenever I need to search for someone in the future—say I need to contact someone who works as an auto mechanic—I just type it in the search bar of the website application and all the people who work as a mechanic appear in front of me, then I could sort them based on either the company they work for or some other criteria. This process is so much faster compared with conventional methods of saving images physically and also much safer because all data is safe in cloud servers.

## 3 Literature Review

To develop this paper, we referred to a few research papers, whitepapers, and independent news articles, written by scholars. These papers were based on cloud computing, web development, machine learning, and text recognition. Author Ray Smith in his research paper talked about tesseract optical character recognition (OCR) engine's accuracy in a comprehensive overview and concluded that tesseract engine is now lagging behind in accuracy. It does perform better in choosing unusual features and lags because of polygonal approximation for classifier rather than raw outlines. He also gave ideas on how we can improve its accuracy.[1] In another research paper on "Historical review of OCR research and development," authors talked about the historical point of view of research OCR and development of OCR. The research paper also guided us on which OCR technology we should use and how.[2] In the research paper "Optical Character Recognition by Open-Source OCR Tool Tesseract: A Case Study," authors discussed the introduction of OCR method, history of open-source OCR tool tesseract, architecture of it, and experiment result of OCR performed by tesseract on different kinds of images. Also provided with comparative analysis of commercial OCR tool Transym. And concluded that tesseract provides better accuracy in getting text from images. This research was conducted on vehicle number plates, and they specifically mentioned that tesseract might be better in this case but might perform differently in other cases.[3] Next, we read about backend technologies that will be used in the processing module. The chapter we read was titled "Developing a Web Application on NodeJS and MongoDB using ES6 and Beyond," written by Aashis Rimal. The chapter talked briefly about JavaScript, its versions, NodeJs, Express framework, MongoDB, Mongoose, its packages, security, etc. It discussed most topics with hands-on work. Finally, we concluded by developing a full stack paper.[4]

After deciding upon which technology to use for converting image to text and technology to use for APIs, we had to decide which technologies to use for the user interaction module. For

this, we read research papers related to front-end technologies and frameworks. The first paper we came across was "Overview of the Angular Framework: Pros and Cons" written by A. K. Oglukyan, This research paper talked about angular framework, a very popular web framework. It explained angular framework, how it works, its updates, when to use it, and pros and cons of incorporating it.[5] Next, we came across a few more web frameworks such as React and Vue.js, we were unsure about which framework to use and how one is better than the other. To get answers to these questions, we read a book, "Supporting Web Development Decisions by Comparing Three Major JavaScript Frameworks: Angular, React and Vue.js." This book had content that focused on comparing aforementioned web frameworks. The comparison was broadly based on "features and technical aspects," "support and accessibility," and "community statistics." It concluded that every framework is good in its own sense and has its own positive points. Angular framework had its own plus points and perfectly fits our use case, such as better types of support, better structuring, better for large-scale papers, and better OOPs support.[6] In a chapter named "JavaScript Frameworks: Angular vs React vs Vue" by Elar Saks, the authors compare the three most famous JavaScript frameworks in popularity, to know the overall performance so that the reader can make a wise decision in figuring out which framework to analyze or to use for the paper.[7] This chapter explains that for purchasing an activity, react being the maximum famous one, is probably the right framework to study. The Vue is the best framework to learn. It offers opportunities to extend the functionalities of the web application with customized modules and visual components. Though the Angular framework seems a little old compared to newly developed frameworks like Vue, it is not leaving any stones unturned in keeping up with the recent advancements in user experience and development needs. Angular is an obvious choice for building enterprise-based applications because of its extensive built-in functionalities and community support.[8–11] More research may be needed to compare frameworks in building huge, more complex multipage applications.[12] In another research paper that we read named "Angular JS" by Sneha Ambulkar, the author provides consumer-centric views for cyberinfrastructure resources, simplifying use and permitting a richer user enjoy.[13–15] After researching and finalizing the user interaction module technologies to use, we had to do research on cloud technologies to use. We read research papers, whitepapers, and documentations provided by cloud service providers. Below, we have included all the literature we came across. In the research paper named "A Review Paper on Cloud Computing" by Priyanshu Srivastava and Rizwan Khan, the author discussed cloud computing technology in IT industries.[16] This research paper describes the creation, evolution, sorts, and additives of cloud computing and strategies of cloud computing and some of its benefits. This paper gives a brief assessment of cloud computing using reviews of more than 30 articles on cloud computing.[17] The outcome of this evaluation shows the face of the IT industries before and after cloud computing.[18–21] The research paper "Cloud Computing Using Amazon Web Services (AWS)" by Suyog Bankar provides a comprehensive assessment of the motivational factors of adopting AWS as cloud computing and reviews the several cloud deployment and service models of AWS. It also explores the benefits of AWS over traditional approaches and its scope in the future.[22] Arabolu Chandra Sekhar and Dr. R. Praveen Sam discussed the basic structure of EC2 Instance and the Architecture of AWS. The primary goal of this paper is on the AWS services list; all services are defined well. The basic emphasis is that Amazon gives a whole set of IT tools for organizations to create devoted virtual clouds.[23] In another research paper named "Benefits of AWS in Modern Cloud" by Sourav Mukherjee, the author gives an overview of the benefits of AWS in the modern cloud. It is popular in all the businesses tasked with enhancing the satisfaction of the provider decreasing charges as the organization will pay for the provider only in what they consume based on the incoming and outgoing traffic. This paper mentioned approximate benefits of every AWS cloud computing carrier and the safety functions.[24] In research paper named "Amazon Web Services" by Avinash Bandaru, the author discussed cloud computing services provided by Amazon, which allows clients to keep data on the platform.[25] Most SMEs are visible to opt for AWS over other carrier companies as AWS is efficient and much less expensive.[26,27] As a result, new and up-coming businesses are more likely to use AWS as their carrier provider for cloud computing. In this paper, the pros and cons of cloud computing, cloud garage structures, and infrastructure using net services, which include Amazon Web Services, are defined in a designated manner.[28]

## 4 System Architecture

We have divided the paper into three modules: user interaction module, processing module, and cloud module. Each module can interact with other modules to do its individual assigned task. Below is the complete detailed information of each module. Each part of the application is visible to the user or the user can interact with what comes under a module. The step by step process of the model is given in Fig. 1.

Upon successful login, the user will be redirected to the user dashboard. The user dashboard will consist of the summaries of users, such as of the past images uploaded, organization details, etc. From here, the user can navigate to three different pages:

1. Past uploaded images

   This page will show history of all the uploaded images. User can perform the basic operations such as view uploaded images, delete existing images, edit existing card information, or can even download the images.

2. Scan a new image

   This will be used to upload new card. Upon the successful capturing of the card, data will be sent to a processing module that will scan and process all the data from the image and it will store data in the database. Depending on data received, it can result in either an error or successful storage of the card. A success page will prompt the user to the dashboard again and an error page will prompt the user directly to the dashboard or ask to recapture the card.

3. Logout screen

   It will sign out the user from the dashboard and redirect the user to the login page. The user cannot extract/see any further information until he or she again signs in.

**Processing module:** So all requests that are raised from the interaction model get processed inside the processing module, the processing module consists of all the business logic. It sends data directly back to the user interaction model or to the cloud model, depending upon the nature and type of request. This module consists of server-side code, databases, APIs, etc.

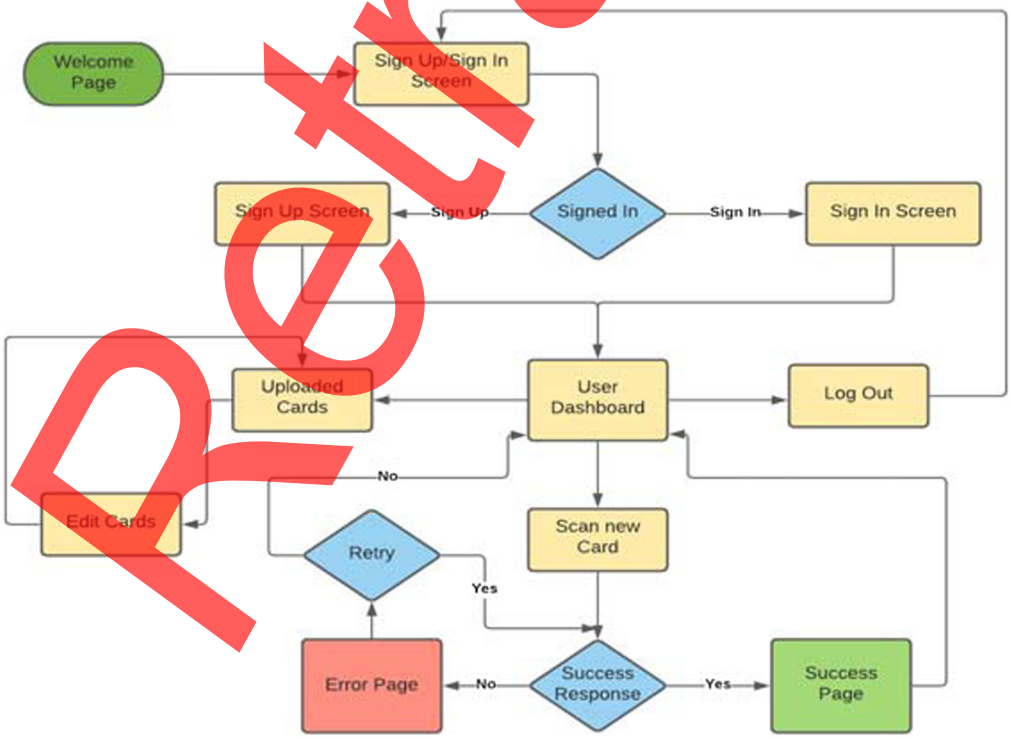


**Fig. 1** Step-by-step process of the model.

### 4.1 *API Design*

In a processing module, most of the tasks were to interact with and make a bridge between cloud services and user interaction module. Hence, it was important to create fail-proof API that can handle heavy and concurrent requests raised from user interaction module without any issue. The Fig. 2 shows the API's total lifecycle in processing module and how it will handle different scenarios.

The API will be hosted on AWS and will use API gateway to provide endpoint for usage. This endpoint will be used to communicate with services in backend such as database, image processors, etc. Each request will be first checked whether it is initiated by an authorized user or not. If a user will not be authorized, then the request will send an error response back. On the other hand if a request will be authorized, then it will be taken further; first, it will be checked if the content in request is valid and sufficient to process the request. Depending on whether the request is valid, API will send an error response or send the request further. Afterward, the request will be checked whether it is available in the server cache. If an available response will be sent from cache, then it will process the request and will send a response accordingly.

### 4.2 *Database Design*

Database designing was another very important step to be considered for successful paper completion. Hence, we took utmost care while designing the database tables and designed them to comply with future needs. In total, two tables were to be created, one for saving details of user and another to save data about images uploaded by the user. The database model is presented in Fig. 3.

The user table was to be used to store data about the user. In the user table, user id will be primary key in table, full name will be a derived attribute, which will be derived from first name

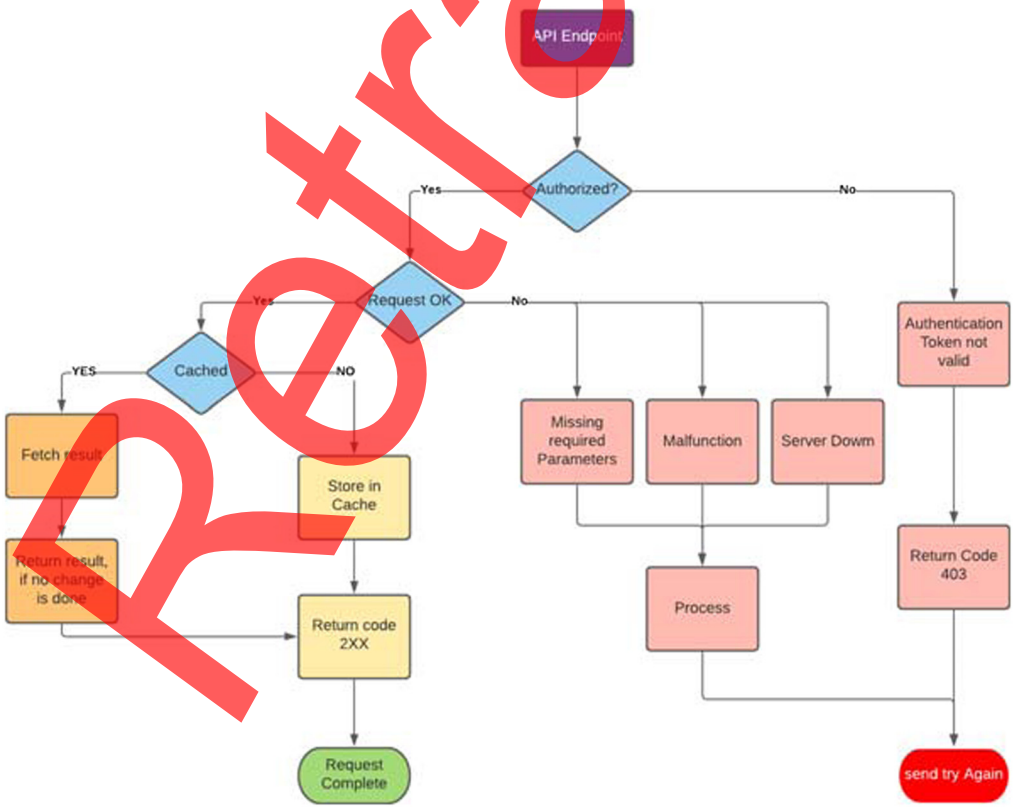


**Fig. 2** API design model.

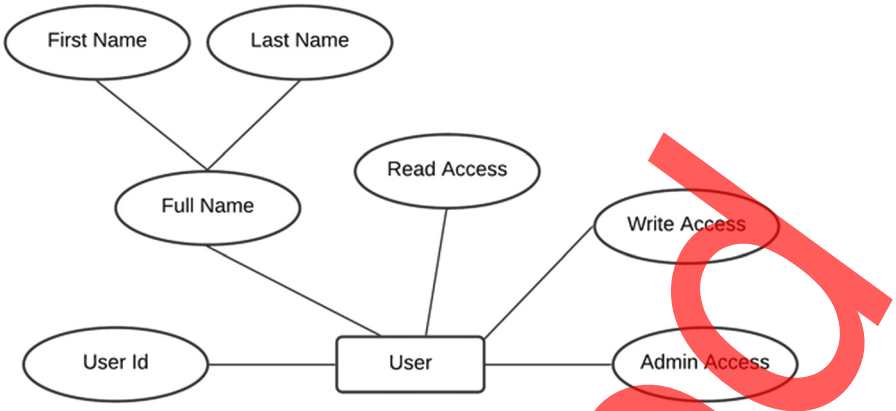


**Fig. 3** Database model.

and last name, then there will be access flags in table, such as "read access," "write access," and "admin accesses." These access flags will be used to store the various kinds of permissions available with the user. The detailed database description is presented in Fig. 4.

Moving on to the image table, it will consist of all information related to images that are uploaded by users. It will have "Card id" as primary key. "User id" will be the foreign key that will be used to determine which card is uploaded by which user. Some information about the card will be available in "Card Title," "Card Person name," "Card Business title," and "Card address." Then there will be "timestamp" to store date and time when the card is uploaded. Lastly 'Card Row Metadata,' which will be storing all information about the card, will be extracted from the card. It will be used in search queries and to restore information about the card. One of the major and most important parts of this paper is its cloud architecture. It is different from other such applications because of its cloud dependence, which provides it a tremendous number of benefits. Hence, it is the module that was to cared for. The first diagram that was developed was of cloud interaction module. Here, the user, its functions, and cloud services will interact to complete different functions. The user will be checked for whether he/she is logged in, then the user will be able to sign up from where data will be sent to Cognito in AWS. If the user will be signed in, then he/she will be able to scan the card and search the card. For a scanning card, data will be extracted using APIs in the processing module, which then will be further sent to Cloud Database service of AWS. For searching of images, again the cloud DB will be requested for results. The user interaction diagram is presented in Fig. 5.

Another important step was to design the cloud architecture and how services will be working and communicating with each other in the cloud. The application was to be served on the cloud,

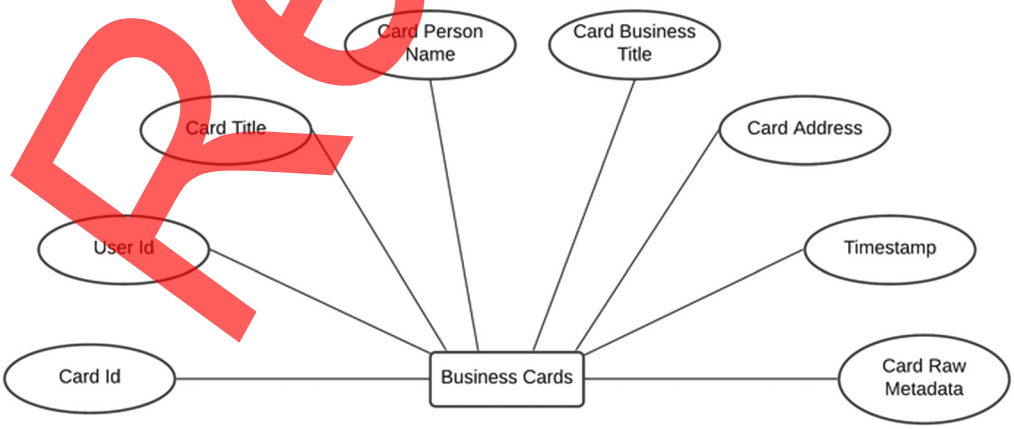


**Fig. 4** Database of images table.

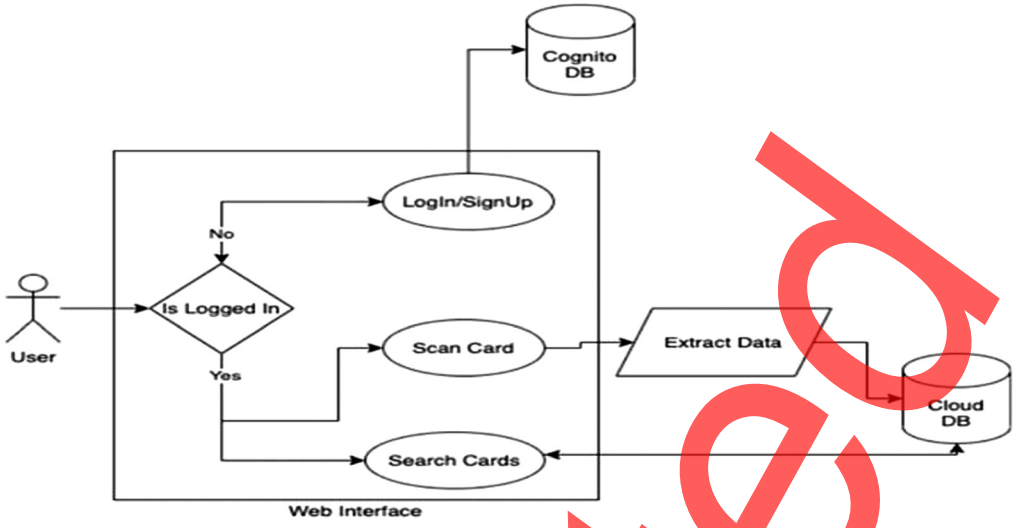


**Fig. 5** User interaction diagram for cloud and user-interaction module.

ranging from front-end, APIs, data storage, asset distribution, etc. Below is the diagram that shows how everything is to be connected.

So, the website will be hosted along with the processing module in Elastic Compute Cloud (EC2) provided by Amazon. This Amazon EC2 will consist of the front-end code and used to process the data of images. The static assets will be hosted on S3 and these static assets will be distributed using AWS Cloud-Front, which then could be used to provide to the user upon request using Amazon Route 53. The front end can also communicate with Amazon Cognito, which is a service that provides authentication, authorization, and user management for web application from Amazon. Apart from this the processing module will be hosted inside Amazon EC2; hence, all the requests that will be raised by the user will get processed inside the EC2 server. After the processing, some APIs will run using Amazon Lambda, which then will store the data in a database hosted on Amazon RDS and will also communicate with the user back and forth with meaningful responses, depending upon the nature of the request. The cloud architecture diagram is presented in Fig. 6.

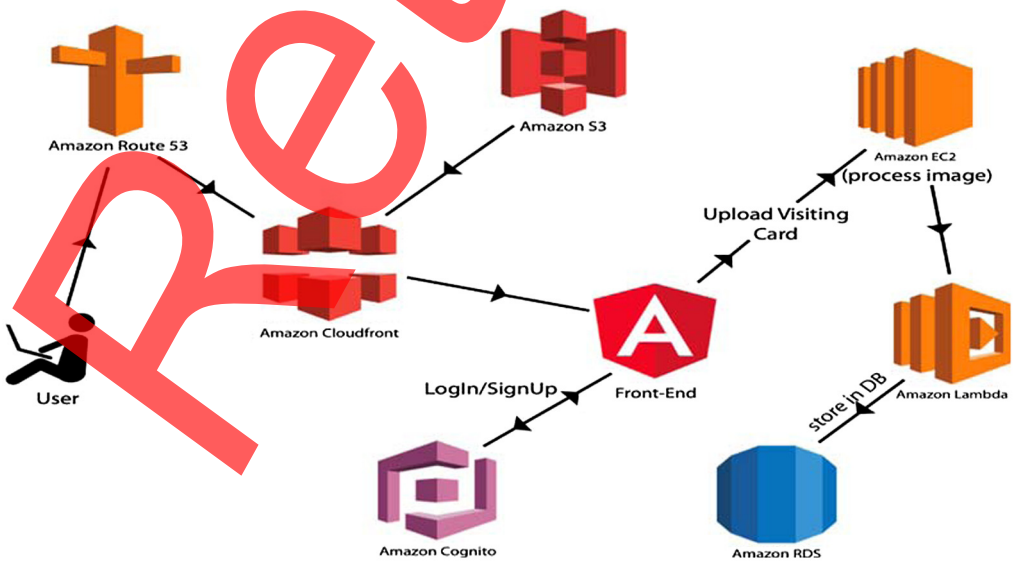


**Fig. 6** Cloud architecture diagram.

## 5 Implementation of Proposed Model and Result

We created the API, which was able to extract text from the card's image and was then able to bifurcate the extracted text into different categories depending upon their nature. This API was developed in Python Flask, and our front-end frameworks were able to communicate with it to carry out the required task. Below is the pseudocode of the API named "/upload Image," which is carrying out this task. What happens in the below code is that whenever a user uploads a card in the frontend, our server receives a request, which triggers the desired function of the python flask. This function is supposed to determine text available in the card image and send it back to the client. It receives an image in request, which is first stored in a local storage server. After the image is stored in local storage, a function is called on the server to process the image and fetch text from it. This function first instantiates python tesseract, followed by importing of Json of cities, which is stored in local storage. After this a conditional statement is called, which pre-processes the received image using three types. First, one passes the image as original without changing anything. Next, one changes the image into grayscale then passes. Last, one changes the image into a high-contrast black-and-white image. After preprocessing of the image, py-tesseract library is used to fetch text with its data in that image. This data extracted consists of various information such as column, line number, height, width of text, along with their respective words. This extracted information is used to determine largest text in card, to get company name/title, followed by extraction of address of card issuer by comparing name of city available in card and then phone, email, and website. This extracted data from three kinds of preprocessed images are further sent back to client after parsing it in a json format. The json consists of emails, phone in form of array and address, and company name is in string format (Algorithm 1).

The aforementioned code is explained using a flowchart, in an image given below. The request comes from the client and, after going through the flow mentioned below, the response is sent back to the client. The flow chart of Python API used to extract image data is presented in Fig. 7.

**Algorithm 1** Cloud and Deep Learning Based Image Analysis algorithm

```
Call '/uploadImage':
    extract image data from request
    store image in server locally
    call extractText() function with stored file name as parameter:
        extract cities from 'cities.json' file
        get image_path from file name
        text = Extract text only from image using pytesseract OCR
        txt = Extract text with data from image using pytesseract OCR
        obj = Run a loop to create an object with text and their respective word heights using txt
        initialize empty array for addr, mobile, email, website
        run loop in obj:
            determine line with largest height and store it as company name
            determine phone number from words
        run loop in text:
            extract and push mobile, email and website in arrays
        return an object with mobile, email, website and address and raw text extracted from card
    run extractText() on greyscale image and pre-processed image
return response of extracted data
```

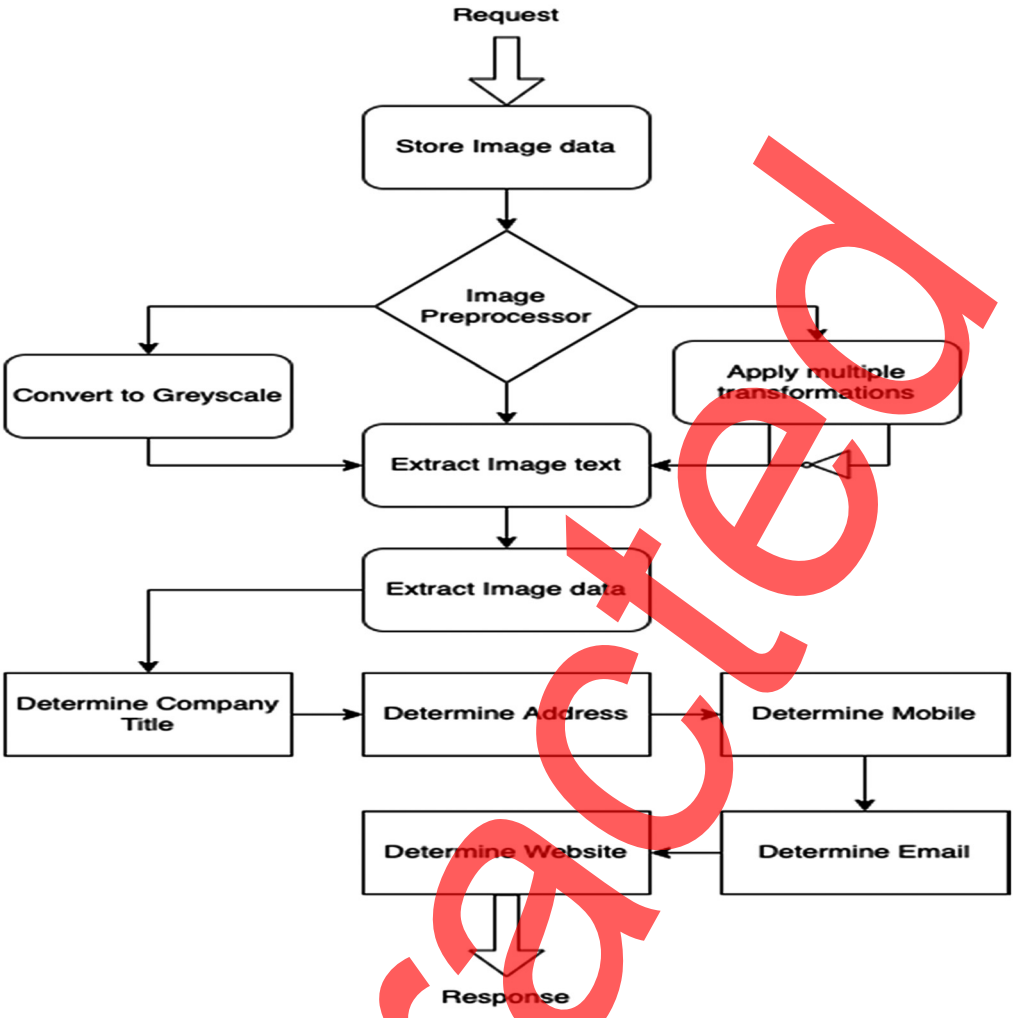


**Fig. 7** Flow chart of Python API used to extract image data.

For extracting text from an image, we passed the uploaded image from a pipeline of tesseract OCR. Accuracy of the text recognition is calculated against five categories determined by the pipeline from the business images. These five categories include email, phone, business title, address, and website embedded in the business card. To improve the accuracy of text recognition from the card, the images of images are processed beforehand. Images are processed using four different algorithms and then their accuracy is determined for each algorithm, to choose the best algorithm for extracting text from the card's image. Below, we have explained each image processing algorithm working in detail:

Algorithm one: No transformation is applied to the original uploaded image. It is passed through the tesseract pipeline as it is.

Algorithm two: The original uploaded image is passed through tesseract pipeline after transforming it into a grayscale image, which changes the way text recognition works on the image.

Algorithm three: The upload image is passed through multiple pipelines of image transformations, which include converting image into grayscale, smoothening it a bit, passing it through a high-pass filter, and contrast increasing of image. These transformations help in increasing chances of detection of a few hidden texts from the image.

Algorithm four: This algorithm is similar to algorithm three except it adds one more step in processing of the image, that is, it takes a complement of the image processed in algorithm three.

For testing each algorithm, we took a set of thousand images where each set contains 100 images and uploaded their images in the system to check each algorithm's accuracy.

**Table 1** Accuracies for different algorithms.

| Card sets | Algorithm one (%) | Algorithm two (%) | Algorithm three (%) | Algorithm four (%) |
|---|---|---|---|---|
| Set-1 | 100 | 80 | 60 | 100 |
| Set-2 | 100 | 80 | 60 | 100 |
| Set-3 | 80 | 100 | 100 | 100 |
| Set-4 | 100 | 100 | 80 | 100 |
| Set-5 | 60 | 100 | 20 | 80 |
| Set-6 | 100 | 80 | 80 | 100 |
| Set-7 | 80 | 100 | 80 | 80 |
| Set-8 | 80 | 80 | 80 | 100 |
| Set-9 | 100 | 100 | 100 | 100 |
| Set-10 | 80 | 80 | 80 | 80 |
| **Total %age** | **88** | **90** | **74** | **94** |

Note: bold characters represent average percentage of accuracy.



Four algorithms performed differently on each set of images. The worst performer was algorithm three with accuracy being 74% and best performer was algorithm four with accuracy of 94%. Table 1 is showing algorithms and their corresponding accuracies showing each algorithm and their corresponding accuracies.

The following bar graph shows each algorithm and its accuracy in a graphical manner. It is clear from the graph that algorithm four is the go-to algorithm for recognizing text from business images. The comparison of different algorithms is presented in Fig. 8.

After creating the algorithm for capturing and sorting data from business images, we have determined how well it performs in an actual scenario. Here, we have made a comparison between three different approaches used to store and search business images in present times. These three methods are: traditional approach where a folder is used to store business images that one receives and is generally followed in MSME's. Second approach is the noncloud approach where an Excel file or some other local storage based software is used to store various details


**Fig. 8** Comparison between different algorithms (%age).

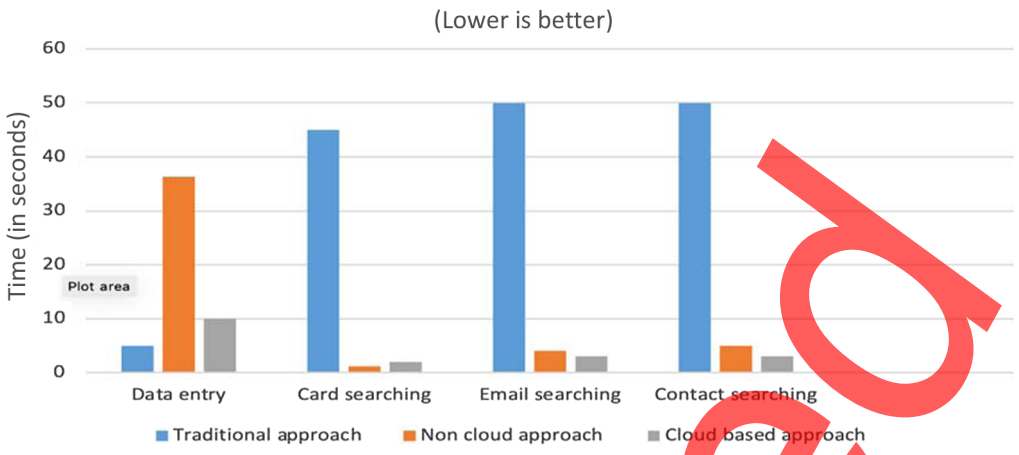


**Fig. 9** Comparison of time taken for different tasks in different approaches.

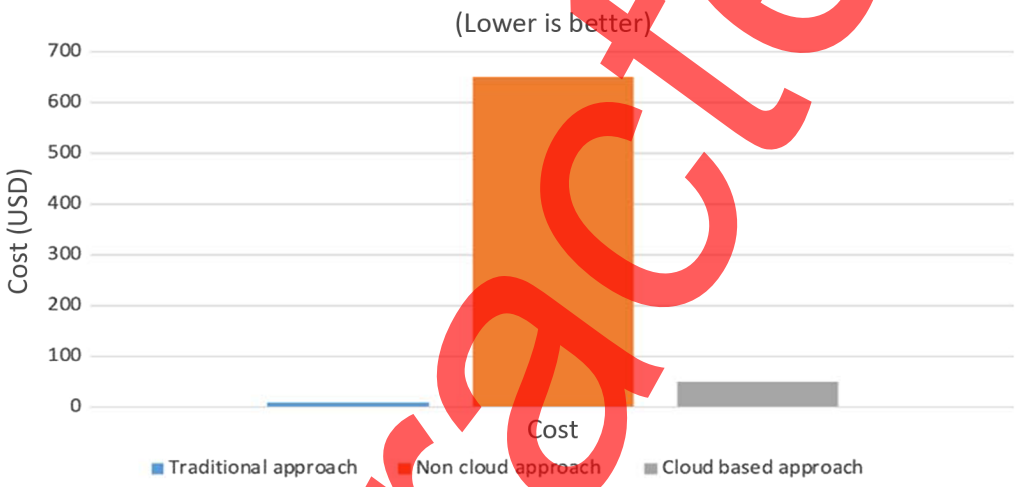


**Fig. 10** Comparison of cost in different approaches.

about a particular person or organization. The Third approach that we have developed is the cloud-based business card analyzer in which the issue of storing and searching the images is solved using a centralized cloud platform. This platform can be used to upload a card image using a camera. Then the image will be processed to extract its contents and store on the cloud. This stored data could then be searched using a frontend with which the user will interact. For this comparison, we have taken in account data from at least 100 business images and then have calculated average time taken for each card. The comparison of time taken for different tasks in different approaches is presented in Fig. 9.

Previously in this paper we have included two figures with comparisons, one containing time taken for different tasks to carry out in three different approaches, another comparing cost to set up three approaches. As we can observe in the Fig. 10, the cloud-based approach is performing optimally in most cases. In some cases, the traditional approach is able to surpass the cloud-based approach such as data entry and cost, traditional approach comes at its own risks and caveats. The data is presented in Table 2. Cloud-based approach is able to outperform in terms of cost, security, robustness, availability, etc.

We tested our model on a few images to check its accuracy in the real world. For this, we took about 800 images in total gathered from different sources and uploaded them on our website. Upon successful upload this gave us the extracted text from the card sorted into its respected category. Then, we calculated the accuracy of the model by comparing the actual text and the extracted text. The data available in Table 3. For each incorrectly extracted text, we flagged it as incorrect and then calculated the accuracy using the equation given below:

**Table 2** Comparison B/W different approaches.

| Parameters | Traditional approach | Noncloud approach | Cloud-based approach |
|---|---|---|---|
| Data entry time (s) | 5 | 36.30 | 10 |
| Card searching time (s) | 45 | 1.25 | 2 |
| Email searching time (s) | 50 | 4 | 3 |
| Mobile number searching time (s) | 50 | 5 | 3 s |
| Cost (USD) | 10 | 650 | 50 |

**Table 3** Accuracy of cloud-based approach (total images taken = 1000).

| Parameters | Number of correctly detected images | Accuracy (%) |
|---|---|---|
| Company name | 888 | 88 |
| Email | 886 | 86 |
| Mobile number | 992 | 92 |
| Address | 990 | 90 |
| Website | 994 | 94 |

$$\text{Accuracy}(\%) = \frac{\text{Number of Cards Correctly Identified}}{\text{Number of Total Cards}} * 100.$$

The cost of our system was about 50 USD per month to the user. This cost can be further reduced by sourcing this system on a subscription basis to the general public. The average estimate of the cost was calculated using an AWS simple cost calculator. In it we included costs of AWS services that we were using. Table 4 that shows the services, their monthly charge in USD, and their configuration.

We implemented an application based on the proposed model. The application can be accessed on url in Ref. 29. From landing page user can navigate to "Log In" page, which provides user access to his/her own profile in website. The authentication is completely managed through AWS Cognito.

Figure 11 shows the dashboard of the application. On this homepage, we can see that there are various subcomponents showing different details. On the top, we can see a search bar, through which we can search any particular image by just entering particular keywords.

## 6 Conclusion

The main goal of the paper was to compare different image processing algorithms and to develop an application, which can be used to store, search, categories, and manage images by combining the benefits of cloud along with computer vision and image processing. Further, we have compared the proposed model with the conventional methods used in the present time. We have compared four different image processing algorithms and have practically proven that algorithm four is the go-to algorithm for recognizing the text from business images. In this algorithm, the upload image is passed through multiple pipelines of image transformations, which include converting the image into grayscale, smoothening it a bit, passing it through a high-pass filter and contrast increasing of the image. These transformations help in increasing chances of detection of a few hidden texts from the image. After that we have complemented the resultant image. After the comparison with already existing approaches, we can draw a conclusion that our

**Table 4** AWS cost estimate.

| Service | Monthly | Currency | Configuration summary |
|---|---|---|---|
| Amazon API Gateway | 0.04 | USD | HTTP API requests units (exact number), average size of each request (34 KB), REST API request units (exact number), Cache memory size (GB) (none), WebSocket message units (exact number), average message size (32 KB), requests (0 per month), requests (10,000 per month), messages (0 per second), average connection duration (0 s) |
| AWS Lambda | 0 | USD | Architecture (x86), architecture (x86), number of requests (10,000 per month) |
| Amazon EC2 | 8.6 | USD | Operating system (Linux), quantity (1), pricing strategy (EC2 Instance Savings Plans 1 Year No Upfront), storage amount (30 GB), instance type (t3.micro) |
| Amazon RDS for MySQL | 39.01 | USD | Storage for each RDS instance [general purpose SSD (gp2)], storage amount (20 GB), quantity (1), instance type (db.t3.micro), utilization (on-demand only) (100 %utilized/month), deployment option (multi-AZ), pricing strategy (OnDemand) |
| S3 Standard | 0.57 | USD | S3 Standard storage (20 GB per month) |
| Data transfer | 0 | USD | — |
| Amazon CloudFront | 2.6 | USD | Data transfer out to internet (20 GB per month), data transfer out to origin (20 GB per month), and number of requests (HTTPS) (20 per month) |
| **Total** | **50.82** | **USD** | — |

Note: bold character represents total monthly cost.

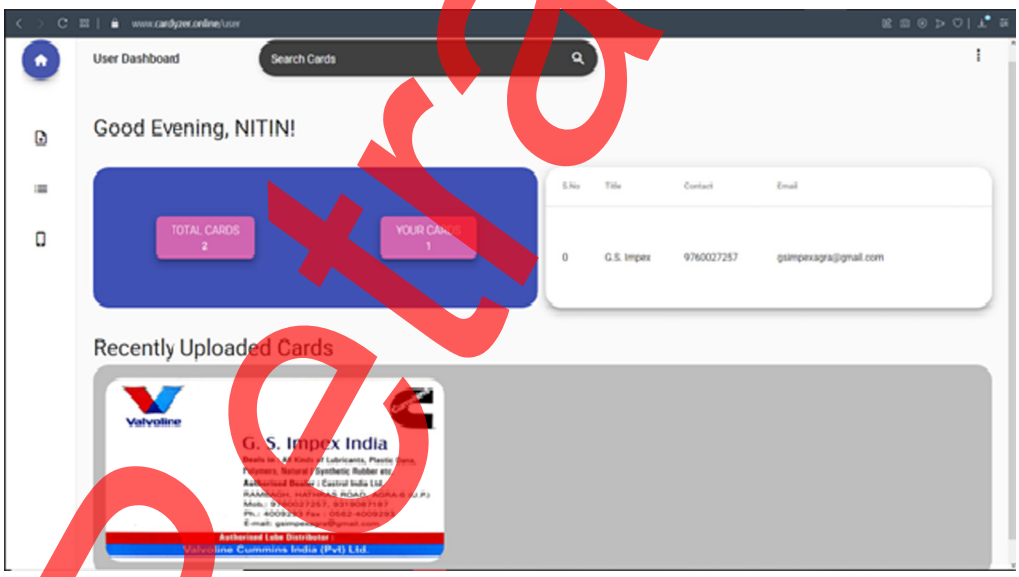


**Fig. 11** User dashboard component of the implemented application based on proposed model.

proposed approach best suits someone who deals with a lot of images and have little time to store, manage, and retrieve this information.

## 7 Future Scope

This paper does have a lot of potential when it comes to improvements and future applications. We brainstormed a bit to come up with a few pointers by which we can improve this paper. Below is the list of things that we can do to improve this paper:

1. **Faster detection of business images:** This is one of the ways we can improve this technology by adding high-speed cameras and motion sensors to do faster detection of business images. This will enhance the existing functionality and will allow users to scan images on the go.
2. **Application in other areas apart from business images:** We can use this technology to extract and work in other fields of daily life. This will increase the scope of this card analyzer.
3. **Scanning of documents:** One of the common applications to add to this card analyzer would be to make it work for documents, to extract information and text from scanned documents. This will make this application multipurpose.
4. **Automated specialized area scanning:** One of the improvements can be to add a feature where the user will put the card in a specialized area ,which will trigger the CCTV to capture it and send the image to a remote server for processing.
5. **Mobile application for scanning:** In this paper, our main focus was to develop a web application, but we can make this application more accessible to the user by wrapping all the APIs and functions in a mobile-based application. This will be much easier to use and access compared to a website.

Apart from the aforementioned applications, we can further improve the scanning algorithms using some advanced techniques such as manipulating and adjusting the image transformation algorithms to achieve higher accuracy in text extraction. We can also add some kind of deep learning layer to the model, which can learn from images and further enhance the result of algorithms we have already used.



## References


1. R. Smith, "An overview of the teserract OCR engine," in *Ninth Int. Conf. Doc. Anal. and Recognit. (ICDAR 2007)* (2007).
2. S. Mori, C. Y. Suen, and K. Yamamoto, "Historical review of OCR research and development," *Proc. IEEE* **80**(7), 1029–1058 (1992).
3. P. Chirag, P. Atul, and P. Dharmendra, "Optical character recognition by open source OCR tool tesseract: a case study," *Int. J. Comput. Appl. (0975–8887)* **55**(10), 50–56 (2012).
4. A. Rimal, "Developing a web application on NodeJS and MongoDB using ES6 and beyond," *Metropol. Univ. Appl. Sci.*, 1–36 (2019).
5. S. Delcev and D. Draskovic, "Modern javascript frameworks: a survey study," in *Zooming Innovation in Consumer Technologies Conference (ZINC)*, pp. 106–109 (2018).
6. E. Wohlgethan, *Supporting Web Development Decisions by Comparing Three Major Javascript Frame-Works: Angular, React and Vue.js*, Hamburg University of Applied Sciences (2018).
7. E. Saks, *JavaScript Frameworks: Angular vs React vs Vue*, pp. 1–43, Haaga-Helia University of Applied Sciences (2019).
8. A. Sneha, "Angular JS," *Int. J. Sci. Eng. Res.* **7**(2), 73–76 (2016).
9. P. Srivastava and R. Khan, "A review paper on cloud computing," *Int. J. Adv. Res. Comput. Sci. Software Eng.* **8**(6), 17–20 (2018).
10. B. Suyog, "Cloud computing using AmazonWeb Services," *Int. J. Trend Sci. Res. Dev.* **2**(4), 2156–2157 (2018).
11. S. Arabolu Chandra and R. Praveen Sam, "A walkthrough of AWS (Amazon Web Services)," *Int. Res. J. Eng. Technol.* **2**(3), 178–180 (2015).
12. S. Mukherjee, "Benefits of AWS in modern cloud," Social Science Research Network, Publication: 331586578 (2019).
13. A. Bandaru, "Amazon Web Services," Research Methods and Professional Issues, Publication: 347442916 (2020).
14. S. Kumar et al., "Energy efficient resource migration based load balance mechanism for high traffic applications IoT," *Wireless Person. Commun.*, 1–14 (2021).

15. S. Kumar et al., "Energy efficient multichannel MAC protocol for high traffic applications in heterogeneous wireless sensor networks," *Recent Adv. Electr. Electron. Eng. (Formerly Recent Patents on Electr. Electron. Eng.)* **10**(3), 223–232 (2017).
16. S. Kumar et al., "Resource efficient clustering and next hop knowledge based routing in multiple heterogeneous wireless sensor networks," *Int. J. Grid High Perform. Comput.* **9**(2), 1–20 (2017).
17. A. Punhani, N. Faujdar, and S. Kumar, "Design and evaluation of cubic torus network-on-chip architecture," *Int. J. Innov. Technol. Explor. Eng.* **8**(6), 1672–1676 (2019).
18. G. Dubey, S. Kumar, and P. Navaney, "Extended opinion lexicon and ML based sentiment analysis of tweets: a novel approach towards accurate classifier," *Int. J. Comput. Vision Rob.* **10**(6), 505–521 (2020).
19. S. Kumar et al., *Evolution of Software-Defined Networking Foundations for IoT and 5G Mobile Networks*, p. 350, IGI Publisher (2020).
20. S. Kumar et al., "Energy aware distributed protocol for heterogeneous wireless sensor network," *Int. J. Control Autom.* **8**(10), 421–430 (2015).
21. S. Reghu and S. Kumar, "Development of robust infrastructure in networking to survive a disaster," in *4th Int. Conf. Inf. Syst. Comput. Networks, ISCON 2019*, pp. 250–255 (2019).
22. S. Kumar et al., "An NS3 implementation of physical layer based on 802.11 for utility maximization of WSN," in *Proc.- 2015 Int. Conf. Comput. Intell. Commun. Networks, CICN 2015*, pp. 79–84 (2016).
23. S. Kumar et al., "A utility maximization approach to MAC layer channel access and forwarding," in *Prog. Electromagn. Res. Symp.*, pp. 2363–2367 (2015).
24. K. Gautam, N. Sharma, and P. Kumar, "Empirical analysis of current cryptocurrencies in different aspects," in *8th Int. Conf. Reliab. Infocom Technol. and Optim. (Trends and Future Directions) (ICRITO)*, IEEE, pp. 344–348 (2020).
25. S. Kumar et al., "EMEEDP: enhanced multi-hop energy efficient distributed protocol for heterogeneous wireless sensor network," in *Proc. - 2015 5th Int. Conf. Commun. Syst. and Network Technol., CSNT 2015*, pp. 194–200 (2015).
26. S. Kumar, P. Ranjan, and R. Ramaswami, "Energy optimization in distributed localized wireless sensor networks," in *Proc. Int. Conf. Issues and Challenges Intell. Comput. Tech. (ICICT)* (2014).
27. K. Gautam et al., "COVID 19 Visitor Management System," in *Int. Conf. Comput. Intell. and Knowl. Econ. (ICCIKE)*, Amity University, Dubai, pp. 560–564 (2021).
28. S. Sudhakaran et al., "Blockchain-based transparent and secure decentralized algorithm," in *Int. Conf. Intell. Comput. and Smart Commun. Algorithms for Intelligent Systems*, Springer, Singapore (2019).
29. S. Kumar et al., "Cloud and deep learning based image analysis," http://www.cardlyzer.online (2022).

**Sunil Kumar** is an associate professor of computer science and engineering at Amity University, Noida, India. His research interests include computer networks, image processing, SDN, and big data. He is industry CCNA, CCNP, AWS certified. He is a member of the IET, CSTA, IAER, IAENG.

**Kartik Gautam** is a research scholar in the Department of Computer Science and Engineering at Amity School of Engineering and Technology, Amity University, Noida, India.

**Vatsal Singhal** is a research scholar in the Department of Computer Science and Engineering at Amity School of Engineering and Technology, Amity University, Noida, India.

**Nitin Sharma** is a research scholar in the Department of Computer Science and Engineering at Amity School of Engineering and Technology, Amity University, Noida, India.