International Journal on Computational Modelling Applications
Vol. 02, Iss. 04, S. No. 05, pp. 41--52, Dec 2025
ISSN (Online): 3048-8516

Received: 04 Jan 2026, Accepted: 15 Jan 2026, Published: 16 Jan 2026
Digital Object Identifier: https://doi.org/10.63503/j.ijcma.2025.216

Research Article

Serverless and NoSQL Databases in the Cloud: A Systematic Review of Security and Management Techniques

Ravi Kiran Kanneganti[1], Karthik Linkalapalli[2*] , Mallikarjun Gouru[3], Geetha Krishna Sangam[4], Nachiket Patil[5], Krishna Chaganti[6]

[1] IT Department, Capgemini America Inc, Dalla, USA

[2] Mainframe Architect, Tata Consultancy Services, Atlanta, USA

[3] Enterprise Content Management, Rocket, Dallas, USA

[4] IT Analyst, Tata Consultancy Services, Irving, USA

[5] Senior Software Engineer, Box Inc, USA

[6] Associate Director, S&P Global, New Jersey, USA

kanneganti.ravi@gmail.com[1], karthik.linkalapalli05@gmail.com[2], mallik.gouru@gmail.com[3], sgkrishna1707@gmail.com[4], hi.nachiket.patil@gmail.com[5], K.chaganti@spglobal.com[6]

*Corresponding author: Karthik Linkalapalli, karthik.linkalapalli05@gmail.com

ABSTRACT

Cloud computing has shifted the paradigm of design of digital systems towards on-demand access to a scalable resource and encouraging fast innovation. Databases provide efficient data storage, retrieval, and management and are thus fundamental to these systems. Applications are becoming increasingly complex and data diverse, and traditional relational databases and monolithic architectures struggle to meet these demands. Serverless computing and NoSQL databases have emerged as important pillars of cloud-native systems in response and offer unmatched scalability, flexibility, and cost-effectiveness. Serverless computing eliminates infrastructure management, enabling automatic code deployment and scaling, while NoSQL databases offer flexible, schema-less storage for large, unstructured data. Despite their benefits, both technologies pose security risks—serverless environments face issues like insecure APIs and limited runtime visibility, while NoSQL systems are prone to weak access controls and injection attacks. This review outlines the most critical vulnerabilities related to the two technologies and shows the most suitable mitigation measures, such as strong authentication, encryption, least privilege access, and continuous monitoring. It also covers the best practices of operations like performance optimization, autoscaling, data backup, and disaster recovery to build resilience and reliability of the systems. The study offers an in-depth overview of the effective serverless and NoSQL infrastructure protection and management by combining the knowledge of the latest research and real-world models. The results should help developers and organizations to create effective, resilient, and secure cloud-based systems that need to be developed according to the changing technology landscapes.

Keywords: *Serverless Computing, NoSQL Databases, Cloud Security, Access Control, Cloud-native Applications, Data Management, Event-driven Architecture, API Security, Encryption, Cloud Infrastructure*

1. Introduction

The modern information technology infrastructure is based on cloud computing [1], which provides elastic storage, on-demand access to computer resources, and rapid application deployment. The database is essential to the effective storage, management, and retrieval of data and is hence the backbone

of many cloud-based systems. Scheduling information and ensuring consistency in transactions have long been the domain of traditional relational databases. However, with the rise of big data and real-time applications, more flexible and scalable data solutions have become necessary.

Two transformative technologies addressing these evolving needs are serverless computing and NoSQL databases, which accommodate the dynamic and decentralized quality of contemporary digital applications. Serverless computing removes the administration of servers in informatics by letting developers concentrate on the execution of code with provisioning, scaling, and administration being managed by cloud providers. Conversely, NoSQL databases [2] provide the schema-less data storage options which are aimed at flexibility and performance, particularly in situations when huge amounts of semi-structured or unstructured data need to be handled in a distributed environment [3].

These technologies bring a new category of security and management issues as they become a part of cloud-native applications. Serverless architectures [4] are known to be efficient and cost-effective but usually do not run in a stateful and event-driven environment, which hinders observability into system behavior and enforcing security policies. Existing monitoring solutions might not fit the fleeting nature of serverless functions, and an incorrect setup can provide an entry point that is subject to exploitation. On the same note, NoSQL databases also focus on speed and scale, and they usually do not include all security features out of the box. The most typical vulnerabilities are a lack of authentication, the presence of sensitive information that can be exposed through insecure API, and the vulnerability to injection attacks, which can cause more chances of data breaches in the cloud-hosted setting.

In order to fully explore the advantages of these technologies and reduce the risks to a minimum, it is vital to enforce stringent security and management measures that are applicable to their specific attributes. That involves the implementation of least-privilege access, API protection, data encryption both in transit and at rest, runtime behavior monitoring, and automation of compliance and recovery processes. Further, with the ongoing development of serverless computing and NoSQL databases, the inner workings of both systems in terms of operational and security models have to be learned to create robust cloud systems. Upon weighing the advantages and constraints and considering the upcoming trends in the field of securing and managing these technologies, organizations be capable of creating awareness strategies to secure their assets and keep their agility and performance in the cloud.

Structure of the Paper

The remaining document is organised as follows: Serverless computing's design, advantages, and security problems are covered in Section II. Section III explores the types of NoSQL databases and their associated vulnerabilities. Section IV outlines best practices for securing and managing both serverless and NoSQL systems. Section V reviews recent studies addressing key challenges in performance, security, and architecture. Section VI concludes the paper, highlighting the need for integrated security and proposing future directions involving AI-driven monitoring and blockchain-based access control.

2. Fundamentals Of Serverless Computing

Serverless computing is an execution model that works in the cloud. The cloud provider handles the infrastructure on the fly, so developers only have to write code. In contrast to conventional computing, serverless platforms dynamically manage server provisioning, maintenance, and scaling in response to demand. Advantages of this paradigm include streamlined scalability, lower operational overhead, and cost effectiveness via pay-as-you-go pricing. Microservices, APIs, and event-driven apps work particularly well with it. Deploying new features more quickly and easily is made possible by abstracting server management. Table 1 [5] below displays a comparative analysis of traditional vs. serverless computing, highlighting the key differences in application, configuration, cost, and maintenance.

Table 1. Comparison Between Serverless and Traditional Computing

Factors	Traditional cloud computing	Serverless computing
Development phase	Difficult	Easy
Automatic scalability	Unavailable	Available
Stateful applications	Easy	Difficult
Security	Complex and less secure	Easy and more secure
Function's life cycle	Long	Short
Troubleshooting and debugging	Easy	Difficult
Server and hardware configuration and maintenance	Required	Unavailable
Failure tolerance	Less reliable	More reliable
Cost (Variable workload)	Expensive	Affordable
Cost (Stable workload)	Affordable	Expensive
Applicable user	Administrator and developer	Developer

Advantages of Serverless Computing

1. Cost-Effective

Services cost according to consumption, as serverless apps are decoupled from server infrastructure. To illustrate the point, programs launch anytime a user initiates a request to an application service. While their apps are idle, cloud providers only charge for the space that is actually used [6].

2. Scalability

Allocating resources was a reasonable problem that serverless solved. As a result, developers won't need to worry about the app's scalability; it handles rising user application demands automatically. Serverless providers launch additional servers to manage an influx of requests to an application function [7].

3. Server-Side Management

Developers in serverless computing don't have to worry about the server-side or its administration. The management and maintenance of the necessary hardware and software to launch applications is handled by serverless cloud providers. They also take care of all the administrative tasks so that the developers may concentrate on the various resources, like the CPU, memory, and storage.

4. Easy To Deploy

Deploying serverless applications is a breeze. For instance, developers merely have to upload a few functionalities and produce a new product in order to launch an application. Issues with infrastructure, including server provisioning and scalability, as well as deployment management, handled by the serverless.

5. Decrease latency

A server is not necessary for the execution of a serverless application; rather, the code can reside on any server, regardless of location. Therefore, cloud service providers can host the app on servers close to the user's physical location. Since the original server is no longer accessible via the Internet, latency is reduced.

Security concerns of Serverless Computing

Serverless computing promises scalability and less infrastructure to manage at the cost of new security challenges presented by an event-driven and distributed computation and architecture. The traditional security controls might not be completely applicable, and misconfigurations can result in severe vulnerabilities. In such environment, it is essential to make sure that there is proper authentication, secure APIs and data protection[8]. The significant issues related with security are as follows:

- **Insufficient Authentication & Authorization:** Weak or misconfigured access controls can lead to unauthorized data access.
- **Serverless Vulnerability Exploitation:** Vulnerabilities in admin interfaces or configurations may be exploited by attackers.
- **Data Exposure:** Misconfigured storage systems can result in unintentional public access to sensitive information.
- **Misconfigured Serverless APIs:** Poor API security practices may allow unauthorized access or data leaks.
- Weak Database Security: Lack of monitoring, auditing, and access controls increases breach risks [9].
- **Supply Chain Attacks:** Compromised third-party services or software can serve as entry points for attackers.
- **Inadequate Network Segmentation:** Poor segmentation allows lateral movement across systems after initial compromise.
- **Cloud Storage Misconfigurations:** Incorrect permissions and access settings can expose confidential data.
- **Weak Serverless Environment Security:** Missing safeguards in serverless platforms can lead to exploitation [10].
- **API-Driven Breaches:** Exposed or insecure APIs in serverless environments are frequent attack targets [11].

3. Nosql Databases: Concepts and Classifications

NoSQL databases have become one of the main solutions to work with big scale and heterogeneous data of the contemporary cloud computing systems. Compared to conventional relational databases, NoSQL systems are meant to achieve better flexibility, scalability, and performance through the process of loosening the rigid schema constraints and embracing distributed data storage [12]. They excel especially at working with unstructured or semi-structured data and are thus well-suited to work with big data, real-time processing, and fast-changing datasets. NoSQL databases in cloud-native and serverless emit architectures scale dynamically, are highly available, and can be accessed with low latency, which are important characteristics in building responsiveness and fault tolerance. They are

designed to implement the ideas of eventual consistency and partition tolerance that are quite suitable in decentralized, high-throughput applications. Consequently, the role of NoSQL databases as the basis of scalable and resilient data management in the cloud computing era is established.

Types of NoSQL Databases

Large amounts of structured, semi-structured, and unstructured data can be easily handled by NoSQL databases due to their scalability and flexibility. They are not based on a predetermined schema like in the case of the traditional relational databases, which makes them well suited to modern cloud applications as well as real-time analytics and big data workloads. NoSQL systems have been classified according to their data models and the way they store as well as retrieve data. The following are the main types(Figure 1):

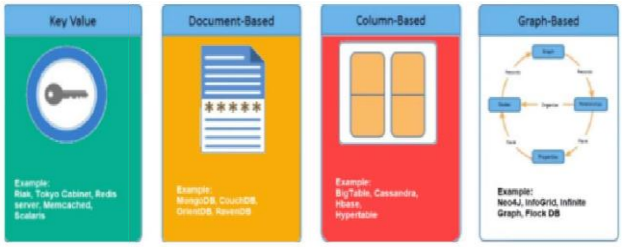

Figure 1. Types of NoSQL Databases

1. Document-Oriented Databases

A document-oriented database's data structure is similar to that of JSON (JavaScript Object Notation) objects [13]. Each document has two fields and two values. In most cases, the values can be anything from strings of text to numbers, Booleans, arrays, or even physical objects. When dealing with collections of semi-structured or even unstructured data, document databases offer a flexible data paradigm that is effective. Another fantastic feature that simplifies the expression of data with complex relationships or hierarchies is the ability to construct nested structures.

2. Key-Value Pairs store

Each object in a key-value store [14] contains both a key and a value, making it the simplest sort of database. Table 2 shows that each key has its own unique value. They offer excellent read/write speeds and are perfect for caching and session management because of their memory retention behaviour.

Table 2. Example Of How Key Pairs Are Stored in a Database

Key	Value
Book Title	Business Intelligence and Analytics: Systems and Decision Support
Aurthor(set)	Ramesh Sharda
	Dursun Delen
	Efraim Turban
Publication Date	2015
Edition	10th
Publisher	Pearson

3. Column Family Stores

Using column families as rows, these databases store and display the information. There is a fixed number of columns in a row that correspond to its key. A collection of linked data that can be viewed separately. A table's row container is like a column family in relational database management systems [15].

4. Graph Database stores

Graph databases are designed to handle data sets with a high degree of connectivity by utilising three fundamental components: nodes, edges, and attributes. The data about entities is stored by nodes, and the relationships between them are defined by edges, which have attributes, a direction, and a type. In graph databases, relationships can be either one-way or two-way, and they specify the path that a traversal can take from one node to another. Nodes and relationships both have properties that can be used to offer further context [16].

Security Challenges in Nosql Databases

Flexibility and scalability provided by NoSQL databases also come with special security considerations not seen with the more traditional relational databases: the absence of a pre-defined schema and a less developed security model in many cases. Most NoSQL systems are focused on performance and ease of use which makes them misconfigured and vulnerable when not managed well. typical security concerns that need to be considered during the implementation of NoSQL databases [17] in cloud are as follows:

- **No authentication:** Some NoSQL databases allow access without verifying user identity.
- **No encryption**: Data is often stored or transmitted without encryption, risking exposure.
- **Insecure defaults**: Default settings may leave ports open or access unrestricted.
- **NoSQL injection**: Improper input handling can let attackers alter database queries.
- **No RBAC**: Lack of role-based access control can give users more access than needed.
- **Weak logging**: Minimal logging makes it hard to detect or investigate attacks.
- **DoS vulnerability:** Large or complex queries can overwhelm and crash the database.
- **Insecure APIs:** Publicly exposed REST interfaces can be exploited without safeguards.
- **Unsafe replication/backups:** Replicas and backups may be unencrypted or poorly secured.
- **No centralized security:** Managing security across multiple NoSQL systems can be inconsistent.

4. Security And Management Techniques in Serverless and Nosql Systems

The security and management approaches to serverless and NoSQL systems should be different as they are calculated in a decentralized and scaled way. In serverless, security is concentrated in the protection of the endpoints of functions, identity and access control, and safe data transfer, frequently by encryption and API gateways. Isolation and monitoring at runtime are important to identify and act on threats. NoSQL databases [18], which often do not have the strict schema or embedded security capabilities of older relational databases, need strong access control, input validation, and encryption at rest and in transit. Management practices should also consider dynamic scaling, event-driven execution and the distributed nature of data; in this regard, automated monitoring, logging and compliance-enforcing tools are mandatory to provide visibility and control over system behavior.

Security Techniques in Serverless and NoSQL Systems

Reduced risk is the primary goal of serverless and NoSQL security measures such input validation, secure secret management, and least privilege access. Utilizing virtual private clouds (VPCs) and firewalls, block network access and implement identity control measures like role-based access control (RBAC). At rest and while in transit, the data must be encrypted. Early detection of threats is assisted by logging, monitoring and vulnerability scanning. The API gateways prevent abuse, and a consistent backup makes it ready to recover disasters. The most significant of the methods are the following:

a) **Least Privilege Access:** Grant the least possible permissions to serverless functions and NoSQL users to minimize the effect of a compromise [19].
b) **Input Validation & Sanitization:** Sanitize and always validate user inputs to avoid injection attacks (e.g. NoSQL injection) [20].
c) **Secure Secrets Management:** Credentials and API keys, not environment variables, should be stored by sensitive services like AWS Secrets Manager or Azure Key Vault.
d) **Network Access Control:** Limit access to databases and functions through VPCs, firewalls and security groups to avoid unauthorized access.
e) **Strong Authentication & RBAC:** Implement authentication of every component and restrict access with Role-Based Access Control (RBAC) [21].
f) **Encryption (At Rest & In Transit):** Keep data in transit and data at rest secure by enabling TLS/SSL and encryption, respectively.
g) **Logging and Monitoring:** Enable logging (e.g., CloudWatch, MongoDB audit logs) and monitor for suspicious activities or anomalies.
h) **Dependency & Package Scanning:** Constantly vulnerability scan all serverless functions dependencies with the help of tools such as Snyk or Dependabot.
i) **API Gateway Protection:** Protect against abuse and DDoS attacks by using API gateways, authentication and rate limiting and throttling.
j) **Backup and Disaster Recovery:** Automated Backup NoSQL databases and test the restore processes on a regular basis [22].

Management and Operational Practices

Performance, reliability, and automation are the key areas of effective management of serverless and NoSQL systems. The most important procedures are memory tuning, reducing cold starts, query optimization, and choosing an economical price. Autoscaling makes the utilization of resources efficient, and the data resiliency is promoted by backups and replication. Monitoring tools give visibility into systems in real time and CI/CD pipelines automate secure and fast deployments. The major practices include:

1. *Performance and Cost Optimization:* Memory allocation, lessening cold starts and adjusting NoSQL queries contribute to enhanced performance and cost decrease. Proper selection of the pricing models (e.g., provisioned vs. on-demand) can also make a contribution to cost efficiency.
2. *Autoscaling and Resource Provisioning:* Serverless platforms [23], and NoSQL databases allow autoscaling to prevent consistent workloads. When configured well, resources are efficiently utilized without over or under provisioning.
3. *Backup and Disaster Recovery:* The cloud offers built-in backups, point-in-time recovery, and tools for cross-region replication. The resilience of both data and businesses is ensured by these traits.

4. *Monitoring and Observability:* Observability tools, such as AWS CloudWatch or Google Operations Suite, allow seeing the state of performance and system health in real-time and help identify and fix problems fast.
5. *CI/CD and Automation:* Infrastructure-as-code and CI/CD pipelines help with accelerating the deployment and automate updates, adding security checks to ensure safer, quicker software deliver.

5. Literature Review

This section reviews key contributions from recent literature, examining how researchers have addressed the critical issues of data protection, system management, and performance optimization in serverless and NoSQL systems.

A and Nathiya (2025) examines the security concerns related to serverless architecture and explores the advancements made in the advanced model architecture to enhance security. The paper discusses the challenges posed by serverless architecture, including data protection, access control, insecure configurations, and vulnerability to attacks. The key areas of focus in enhancing the serverless architecture model are secure deployment, zero-trust security model, runtime security, and identity and access management. The paper recommends implementing secure deployment practices, enforcing a zero-trust security model, ensuring runtime security, and managing identity and access to improve security in serverless architecture. This study surveys the current literature on serverless architecture and security issues in great detail [24].

Ferreira et al. (2025) analyse the impact of various data consistency settings on the efficiency of three well-known NoSQL (Not simply SQL) databases—Cassandra, MongoDB, and Redis—in a cloud context spanning multiple regions. To run workload simulations, analyse performance metrics, and compare outcomes, use the Yahoo! Cloud Serving Benchmark (YCSB) tool. According to the research, settings with excellent data consistency lead to a noticeable decrease in performance. A good example of this is Cassandra, which can reduce the amount of reading and writing operations executed per second by as much as 95% under certain workloads. Similarly, Redis execution times for writing/reading operations might be nearly 20 times slower when high data consistency is enforced [25].
Bhatt, Sharma and Bhadula (2024) present a literature review on serverless computing that covers topics such as the framework and its potential vulnerabilities, threat models and assaults on serverless computing, and methods employed thus far to ensure the safety of serverless computing for both users and cloud service providers. Following a brief overview of the project's goals, the following section presents the serverless computing architecture [26].

Meher, Mishra and Sahoo (2024) tries to filter out the problems in traditional databases and makes a comparison with NoSQL database. Big data is a term applied to a huge unit of heterogeneous data, including a typical form that are rapidly increasing in size using existing tools and approaches to process, analyze, or display. Big data is also the term that summarizes the processes, tools, and techniques used to learn from this data using tools like NoSQL. Now a day as compare to structured data, semi-structured and unstructured data are released in more amounts. In order to manage all types of data with a good efficiency, speed and storage and focus on the new and advanced featured databases like NoSQL. have explored four major types of NoSQL databases based on which industries are focusing on application development mostly [27].

Krishan, Gupta and Bhathal, (2024) aim to examine the complexities associated with maintaining data consistency in widely used NoSQL databases such as Redis, CouchDB, MongoDB, Neo4J, and others. The main aim of this work is to explore the intricate challenges related to maintaining data consistency in widely used NoSQL databases. Prominent examples in this investigation encompass Redis, CouchDB, MongoDB, and Neo4J, among other notable options. The study aims to analyze and understand the unique characteristics and capabilities of these databases, examining how they address and overcome issues associated with ensuring consistency in the always-changing data environment. This study provides significant insights into the ongoing evolution of database architectures and their important role in defining the present digital environment by explaining the strengths and complexities of each NoSQL database [28].

Dey, Reddy and G (2023) analyse serverless computing by looking at its foundational ideas, major obstacles, and potential advancements in the future. The article gives a general outline of serverless architecture, focusing on its scalability, event-driven design, and departure from infrastructure administration in favour of code-centric development. Function as a Service (FaaS) and Backend as a Service (BaaS) are two architectural paradigms that are examined in the study, and their advantages and use cases are compared. The article delves into recent developments in development tools, hybrid cloud integration, serverless orchestration, and current problems. Serverless computing, the Internet of Things (IoT), and machine learning are also taken into account. Academics and professionals in the field find this work useful for learning about serverless computing, filling in knowledge gaps, and fostering innovation in cloud computing [29].

Table 3 presents a summary of key literature related to serverless computing and NoSQL databases in cloud environments. It highlights each study's focus area, objectives, core technologies, and major findings related to architecture, security, and performance.

Table 3. Summary Of Key Studies On Serverless and Nosql in Cloud Computing

Author(s) & Year	**Focus Area**	**Key Objective**	**Key Technologies/Concepts**	**Key Findings/Contributions**
A and Nathiya (2025)	Serverless Security	Examine serverless architecture vulnerabilities and propose enhancements	Zero-Trust Model, IAM, Runtime Security	Recommends secure deployment, zero-trust practices, and improved identity and access controls
Ferreira et al. (2025)	NoSQL Consistency & Performance	Evaluate the impact of consistency configurations on performance in multi-region cloud deployments	Cassandra, MongoDB, Redis, YCSB	Strong consistency significantly degrades performance; benchmarking supports trade-off analysis
Bhatt, Sharma, and Bhadula (2024)	Serverless Threat Models	Review threat models, attacks, and security techniques in serverless computing	Threat Models, Serverless Frameworks	Identifies common attacks and summarizes current security practices for both users and providers

Meher, Mishra, and Sahoo (2024)	NoSQL vs Traditional DBs	Compare traditional databases with NoSQL for big data management	Key-Value, Document, Column, Graph NoSQL Types	Highlights NoSQL's flexibility, scalability, and suitability for semi/unstructured data
Meher, Mishra, and Sahoo (2024)	NoSQL Data Consistency	Explore consistency challenges across major NoSQL platforms	Redis, CouchDB, MongoDB, Neo4J	Provides insight into database capabilities and the evolving complexity of data consistency
Dey, Reddy, and G (2023)	Serverless Architecture & Innovation	Discuss serverless paradigms and innovations in orchestration, hybrid cloud, edge computing, and ML integration	FaaS, BaaS, IoT, Edge Computing, ML	Emphasizes emerging trends and research gaps in serverless and cloud technology

6. Conclusion And Future Work

Databases are still absolutely essential for storing and managing data across all of today's applications, but cloud computing has revolutionized modern IT by providing adaptable, on-demand resources. As data volume and application complexity grow, more adaptive solutions are needed. The evolution of cloud-native development has been changed by serverless computing and NoSQL databases, which provide scalable, agile and cost-effective solutions. By incorporating them into present application architectures, developers can innovate faster by removing the complexities of the infrastructure. Nonetheless, accompanying this move is the creation of formidable security and management risks, such as restricted observability over serverless applications, unsecured API settings, data exposure in NoSQL databases, and the absence of centralized access controls. This review has described the most common vulnerabilities of these technologies and has highlighted the need to implement integrated security measures, including least privilege access, encryption, authentication protocols, and monitoring, in order to enhance security. Also essential to resilience and system integrity are automated scaling, effective backup practices, and performance tuning operational practices.

The future direction of research ought to be the standardization of security frameworks within serverless and NoSQL ecosystems, specifically. That involves the incorporation of AI-based monitoring systems that can identify abnormalities in real-time, the development of zero-trust architectures that are streamlined toward ephemeral functions, and the development of consolidated dashboards to manage multi-database setups. Additionally, the investigation of the possibilities of access control using blockchain in decentralized cloud infrastructures may provide additional avenues of improving transparency and trust.

Funding source

None.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] Kaur, "A review paper on cloud computing and privacy problems," *International Research Journal of Modern Engineering, Technology and Science*, vol. 3, no. 11, 2021, doi: 10.17577/IJERTCONV5IS23002.

[2] V. S. Thokala, "A comparative study of data integrity and redundancy in distributed databases for web applications," *International Journal of Research and Analytical Reviews*, vol. 8, no. 4, pp. 383–390, 2021.

[3] Chaudhari and S. Chitraju, "Achieving high-speed data consistency in financial microservices platforms using NoSQL technologies," *International Journal of Advanced Research in Science, Communication and Technology*, vol. 4, no. 2, pp. 750–759, Jun. 2024, doi: 10.48175/IJARSCT-18890.

[4] S. Shah and M. Shah, "Deep reinforcement learning for scalable task scheduling in serverless computing," *International Research Journal of Modern Engineering, Technology and Science*, vol. 3, no. 12, Jan. 2021, doi: 10.56726/IRJMETS17782.

[5] H. B. Hassan, S. A. Barakat, and Q. I. Sarhan, "Survey on serverless computing," *Journal of Cloud Computing*, 2021, doi: 10.1186/s13677-021-00253-7.

[6] V. K. Thatikonda, "Serverless computing: Advantages, limitations and use cases," *European Journal of Theoretical and Applied Sciences*, 2023, doi: 10.59324/ejtas.2023.1(5).25.

[7] Jain, S. James, and A. Chambers, "Serverless computing: A comprehensive survey," 2025.

[8] S. K. Mondal, R. Pan, H. M. D. Kabir, T. Tian, and H.-N. Dai, "Kubernetes in IT administration and serverless computing: An empirical study and research challenges," *The Journal of Supercomputing*, vol. 78, no. 2, pp. 2937–2987, Feb. 2022, doi: 10.1007/s11227-021-03982-3.

[9] V. S. Thokala, "Improving data security and privacy in web applications: A study of serverless architecture," *International Research Journal*, vol. 11, no. 12, pp. 74–82, 2024.

[10] S. Ahmadi, "Challenges and solutions in network security for serverless computing," *International Journal of Current Science Research and Review*, vol. 7, no. 1, 2024, doi: 10.47191/ijcsrr/v7-i1-23.

[11] J. Kanamugire, F. Alhajri, and J. Swen, "Serverless security and privacy," 2024, doi: 10.13140/RG.2.2.34691.52006.

[12] S. Bloria, "Evolution and impact of NoSQL databases: A comprehensive review," *International Research Journal of Modern Engineering, Technology and Science*, vol. 6, no. 6, 2024.

[13] O. Mishra, P. Lodhi, and S. Mehta, "Document-oriented NoSQL databases: An empirical study," in *Communications in Computer and Information Science*. Singapore: Springer, 2018, pp. 126–136, doi: 10.1007/978-981-10-8527-7_12.

[14] N. D. Karande, "A survey paper on NoSQL databases: Key-value data stores and document stores," *International Journal of Research in Advent Technology*, vol. 16, no. 2, 2018.

[15] Sethi, S. Mishra, and P. K. Patnaik, "A study of NoSQL databases," *International Journal of Engineering Research and Technology*, vol. 67, no. 6, pp. 14–21, 2014.

[16] V. S. Thokala, "Efficient data modeling and storage solutions with SQL and NoSQL databases in web applications," *International Journal of Advanced Research in Science, Communication and Technology*, vol. 2, no. 1, pp. 470–482, Apr. 2022, doi: 10.48175/IJARSCT-3861B.

[17] S. Ramzan, I. S. Bajwa, R. Kazmi, and Amna, "Challenges in NoSQL-based distributed data storage: A systematic literature review," *Electronics*, vol. 8, no. 5, Art. no. 488, Apr. 2019, doi: 10.3390/electronics8050488.

[18] S. S. S. Neeli, "The significance of NoSQL databases: Strategic business approaches and management techniques," *Journal of Advanced Development Research*, vol. 10, no. 1, p. 11, 2019.

[19] Marin, D. Perino, and R. Di Pietro, "Serverless computing: A security perspective," *Journal of Cloud Computing*, 2022, doi: 10.1186/s13677-022-00347-w.

[20] J. Al Jarri, D. Alharthi, and M. Alquraishi, "Security implications of serverless computing in the cloud," Zenodo, vol. 12, no. 3, pp. 1–4, 2024, doi: 10.5281/zenodo.12634327.

[21] H. Khan, "Analysis of role-based access control methods in NoSQL databases," 2019.

[22] Abadi, A. Haib, R. Melamed, A. Nassar, A. Shribman, and H. Yasin, "Holistic disaster recovery approach for big data NoSQL workloads," in *Proc. IEEE Int. Conf. Big Data*, 2016, doi: 10.1109/BigData.2016.7840833.

[23] S. S. S. Neeli, "Serverless databases: A cost-effective and scalable solution," *International Journal of Innovative Research in Engineering, Multidisciplinary Physical Sciences*, vol. 7, no. 6, p. 7, 2019.

[24] S. A. and R. Nathiya, "Enhancing advanced model architecture in serverless architecture to improve security," in *Proc. 8th Int. Conf. Trends in Electronics and Informatics (ICOEI)*, IEEE, Apr. 2025, pp. 653–659, doi: 10.1109/ICOEI65986.2025.11013761.

[25] S. Ferreira, J. Mendonça, B. Nogueira, W. Tiengo, and E. Andrade, "Benchmarking consistency levels of cloud-distributed NoSQL databases using YCSB," *IEEE Access*, vol. 13, pp. 63428–63438, 2025, doi: 10.1109/ACCESS.2025.3558923.

[26] Bhatt, S. Sharma, and S. Bhadula, "Security issues in serverless cloud computing architectures," in *Proc. IEEE Int. Conf. Computing, Power and Communication Technologies (IC2PCT)*, Feb. 2024, pp. 39–43, doi: 10.1109/IC2PCT60090.2024.10486369.

[27] K. Meher, D. Mishra, and A. K. Sahoo, "Exploring flexibility of NoSQL databases in big data environments," in *Proc. IEEE Int. Conf. Applied Electromagnetics, Signal Processing, and Communication (AESPC)*, Nov. 2024, pp. 1–6, doi: 10.1109/AESPC63931.2024.10872339.

[28] K. Krishan, G. Gupta, and G. S. Bhathal, "Striking the balance: Comprehensive insights into data consistency in NoSQL realms," in *Proc. IEEE Int. Conf. Computing for Sustainable Global Development (INDIACom)*, Feb. 2024, pp. 715–720, doi: 10.23919/INDIACom61295.2024.10498626.

[29] N. S. Dey, S. P. K. Reddy, and L. G, "Serverless computing: Architectural paradigms, challenges, and future directions in cloud technology," in *Proc. 7th Int. Conf. I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, IEEE, Oct. 2023, pp. 406–414, doi: 10.1109/I-SMAC58438.2023.10290253.