ISSN 2581 – 3781

Available Online at www.ajcse.info
Asian Journal of Computer Science Engineering 2025;10(2):1-9

REVIEW ARTICLE

A Review of API Management Systems and Their Role in Seamless Integration Between Software Applications

Mahi Ratan Reddy Deva*
Independent Researcher

Received on: 13-04-2025; Revised on: 25-05-2025; Accepted on: 17-06-2025

Abstract—**Application Programming Interfaces (APIs) have become essential in modern software ecosystems, enabling seamless integration, interoperability, and scalability across applications and services. However, as API adoption increases, challenges related to security, performance, and governance arise, necessitating effective API management systems. This study explores the taxonomy of APIs, their evolving role in software integration, and the significance of API management solutions in addressing these challenges. Key components of API management, including API gateways, monitoring, security enforcement, and lifecycle governance, are examined. Furthermore, the study highlights emerging trends such as AI-driven API orchestration, decentralized API architectures, and automated security protocols. By analyzing API management's impact on software ecosystems, this research provides insights into best practices for optimizing API-driven digital transformation.**

Keywords—Application Programming Interfaces (APIs), API Management, API Gateways, API Security, Authentication.

I. Introduction

A key component of contemporary software development in the digital age, Application Programming Interfaces (APIs) facilitate the smooth integration of apps, services, and systems. APIs define standardized functions and protocols that facilitate interoperability, scalability, and secure data exchange across distributed environments. Their role as digital connectors allows businesses to enhance functionality, improve efficiency, and foster innovation through third-party service integrations[1].

With the increasing adoption of cloud computing, IoT, and AI, APIs serve as critical enablers of real-time data exchange and interoperability. API-driven development has become the foundation of digital ecosystems, allowing organizations to expand market reach and create new revenue streams by opening their platforms to third-party developers. In sectors such as finance, e-commerce, and enterprise applications, APIs streamline digital transactions and enhance customer experiences, demonstrating their transformative business impact[2].

However, API management has become a critical aspect of software ecosystems, ensuring the secure, efficient, and scalable use of APIs. Organizations leverage API management systems to govern access, monitor performance, enforce security policies, and enable seamless connectivity between applications. RESTful APIs, known for their simplicity and scalability, remain dominant, while Graph QL is gaining popularity in data-rich environments[3].

Thus, API management systems serve as the foundation for seamless software integration by optimizing API lifecycles, enhancing performance, and securing digital transactions. To keep up with the ever-changing digital ecosystem, organizations will need API management solutions that include intelligent orchestration, automated security standards, and decentralized API designs[4].

Motivation of the Study

APIs are crucial for seamless integration in modern digital ecosystems, but without proper management, it pose security, scalability, and performance challenges. API management systems ensure secure access, efficient traffic control, and lifecycle governance, enabling businesses to optimize interoperability and innovation. With industries increasingly adopting API first strategies, understanding API management's role in enhancing security, scalability, and digital transformation is essential.

Structure of the paper

This paper is organized in the following way: Section II categorizes APIs and analyzes their characteristics. Section III explores API management systems, including architecture, key components, and evolution. Section IV discusses API management's role in software integration, covering security, authentication, and monitoring. Section V examines challenges such as security, scalability, and legacy integration. Section VI wraps up with important results and suggestions for further study.

II. API Taxonomy and Management for Seamless Software Integration

A key component in enabling interaction between various services and products is an API. A substantial portion of a company's income comes from APIs. For instance, modern APIs conform to standards that are readily available and understood, which is one of the many qualities that have made them very important and helpful. Such application programming interfaces are well-structured and tailored to a certain audience via the use of their own Software Development Life Cycle (SDLC)[5].

The most popular API is undoubtedly the web-based kind. The service facilitates communication between web apps that must communicate with one another via the Internet[6]. The vast majority of APIs are these. Web APIs have proliferated online due to the prevalence of the Web as a platform. Technology and the need for enhanced services have been driving forces in the growth of modern networks. As a result, additional application criteria have become necessary.

Figure 1 shows the API taxonomy. Numerous Web formats and standards have emerged as a result of the meteoric rise in API use on the Web[7]. The advent of JavaScript Object Notation (JSON) has forced an increase in the reliance on Hypertext Transfer Protocol (HTTP) from the previously dominant format, Simple Object Access Protocol (SOAP) among business APIs. As a result, RPC (Remote Procedure Call) APIs that use JSON have become more popular. The RESTful API is now the industry standard for web applications[8]. API implementations in Java are one example. Another name for class APIs is library-based API. These specialized APIs provide certain data and the actions associated with it. Such APIs are available from the Java programming community.

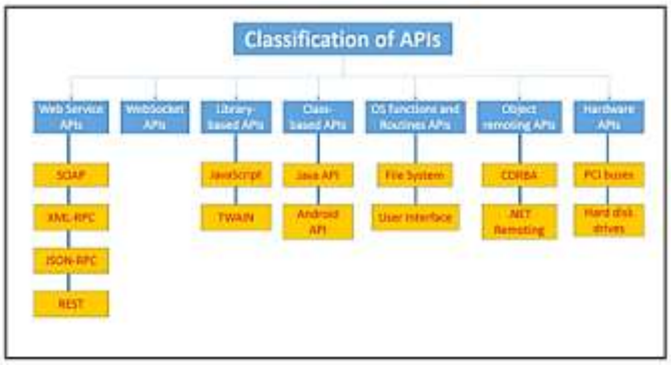

Fig. 1. Taxonomy/Classification of APIs.

A. *Taxonomy of Web APIs*

A web service makes its services accessible via a World Wide Web URI or URL[9]. The presentation of the data to other apps is the most important aspect since it facilitates comprehension. Data exchanged by a web service occurs over HTTP and HTTPS. The program makes an HTTP request, including the necessary arguments and the URL as a path, and the web service returns an HTTP response. Just a few data formats are widely used, including XML and JSON. "Data serialization" describes the steps used to transform data into a format that may be easily transferred.

- The SOAP is both a protocol and a framework for messages. It facilitates application-to-Web communication. If a SOAP client can accept the requests, then SOAP may be used with any protocol. The use of XML and WSDL is crucial to its operation.
- Additionally, XML-RPC RPC APIs are simpler to set up in comparison to SOAP. When making calls via HTTP, the XML-RPC is a necessary component since it forms the foundation for SOAP.
- JSON-RPC Original intent was to facilitate client-browser communication using a message system similar to AJAZ. In subsequent revisions, JSON was improved to provide a more suitable format for deserialization by making it easier to record data types and states.
- Built-in protocols are a boon to RESTful APIs. The use of HTTP is maximized. Adaptability to various kinds of requests, return data formats, and hypermedia structures is made possible by REST since data is not reliant on its methods or resources.

Table 1 shows a concise summary of the advantages and disadvantages of REST, SOAP, and RPC APIs. Unlike SOAP, which is limited to the XML format, REST may return data in any format, including JSON. REST users do not have to be familiar with the names of the procedures and their arguments in the same sequence as RPC users. On the other hand, REST lacks the capacity to save state, including sessions.

Table 1: Summarizes the strengths and weaknesses of SOAP, RPC and REST APIs.

Description	SOAP	RPC	REST
Library support	Requires a SOAP library on client	TightlyCoupled	No library support needed, typically used over HTTP
Type of data support	Not strongly supported by all languages	Can return any format	Returns data without exposing methods
Method exposure	Exposes operations/method calls	Requires user to know procedure names	Supports any content-type
Format	Larger packets of data, XML format is required	Specific parameters and order	NotNeeded
Resource connections	WSDL - Web Service Description	Requires a separate URI/resource for every action/method	Single resource for multiple actions
Connection Type	All calls sent through POST	Typically utilizes just GET/POST	Typically uses explicit HTTP ActionVerbs
Documentation	Not Necessary	Require extensive documentation	Documentation can be supplemented with hypermedia
Session Type	Stateless or Stateful	Stateless	Stateless
Ease of use	Most difficult for developers to use	Easy for developers to get started	More difficult for developers to use

III. Understanding API Management Systems

Different APIs stand for different types of application programming interfaces, which might include different ideas. APIs, or interfaces among modules, were its foundation. APIs have been described as " the interface to an externally distributed, reusable software item that is utilized by several customers outside of the creating organization" in previous endeavors. There is terminology to describe certain kinds of APIs, while the word "API" may be used generically to describe any interaction between software components[10].

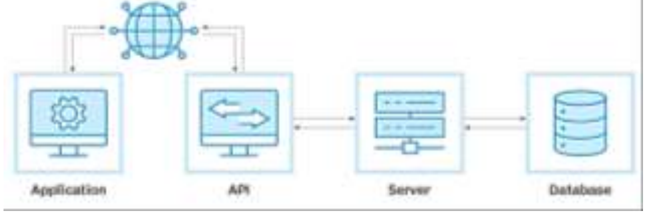

Fig. 2. API Workflow

In Figure 2 collection of instructions, functions, protocols, and objects make up an API. It carries out routine tasks to communicate with external systems. API is adaptable, user-friendly, and effective[11]. Web, mobile, and SaaS programs rely on APIs to connect modules, software, and developers [12].

The API management system provides visibility and monitoring capabilities[13] to handle privacy and security concerns, and it also provides audit trails that record the use of its APIs[14].

API management is a critical component of modern digital ecosystems, enabling organizations to efficiently control, monitor, and secure their APIs. Figure 3 illustrates key aspects of API management[15], highlighting essential functionalities such as partner onboarding, security, Mobile Backend as a Service (MBaaS), governance meetings, monitoring, management, developer support, and mediation.

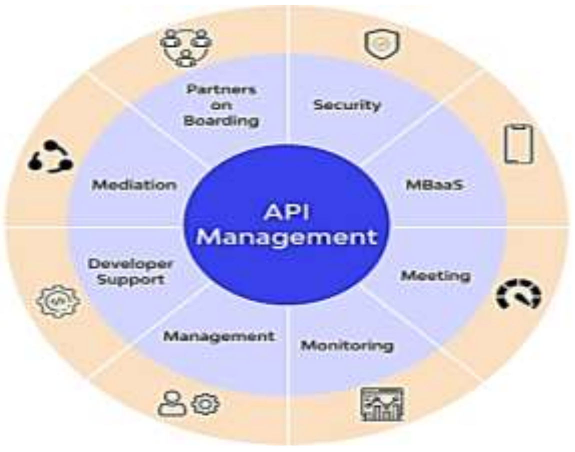

Fig. 3. API Management

A. *Evolution of API management system*

In the rapidly evolving landscape of enterprise technology, APIs have emerged as the cornerstone of modern digital architectures[10]. This technical exploration delves into the transformative role of APIs in contemporary integration strategies and examines how API management platforms are reshaping enterprise connectivity.

B. *Components of API management system*

A collection of procedures, methods, and tools for managing APIs in a safe and extensible service architecture is known as API management. The following are the primary parts of an API management system:

- **API Portal:** The API enhancement lifecycle, API control, profile, and distribution may all be managed with this configuration-time administration tool.
- **API Gateway:** This is a tool for managing APIs during runtime, which includes features like security, multi-tenure, and steering.
- **API Service Manager:** This tool is designed to manage the many stages of an API's lifespan, including dynamic formation, relocation, setup, arrangement, and API updates[16].
- **API Monitor:** This is a component of the API runtime administration segments that measure the API runtime practices, such as execution and utilization. The API billing or chargeback system is designed for open APIs that are organized by utilities.

C. *Purpose of API management system*

- API management is the process by which businesses create, publish, and release their APIs to the development community.
- API Management features, including lifecycle management, authentication and access to APIs, monitoring, throttling, and consumption analysis[17].
- An API gateway generally facilitates their implementation via an integrated platform, which in turn provides security and documentation[18].

IV. The Role of API Management in Software Application Integration

Security, scalability, effective utilization, governance, and control over the lifespan of an API are all responsibilities of API administration. A collection of procedures, methods, and tools for managing APIs in a safe and extensible service architecture is known as API management. The fundamental parts of the API management system are the API Gateway, API Portal, API Service Manager, API Monitor, and API Billing. By offering a systematic strategy for developing, implementing, protecting, and monitoring APIs, API management is essential to guaranteeing smooth interaction across software applications[19].

Key Aspects of API Management in Integration:

A. *API Gateways*

The foundation of every API management system is an API gateway, which allows digital applications and back-end services to communicate in a safe, flexible, and dependable manner. As RESTful APIs, it facilitates the exposure, security, and management of back-end data and services.

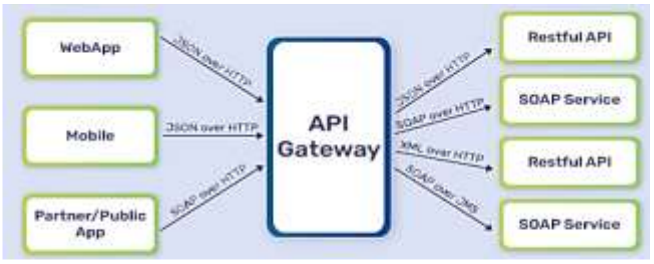

Fig. 4. API Gateways

API Gateway architecture is shown in Figure 4, which serves as an intermediary between client applications and backend services, streamlining communication and enhancing security[20]. On the left side, various client applications, including web apps[21], mobile applications, and partner/public apps, interact with the API Gateway using different communication protocols, such as JSON over HTTP and SOAP over HTTP.

B. *Security and Access Control Mechanisms in API Management*

APIs allow users to have access to sensitive information and resources. The security of APIs is crucial in preventing unauthorized and unauthenticated access to assets.

1) *Authentication Mechanisms*

Authentication mechanisms are crucial for verifying the identity of users and applications accessing an API. Authentication is the procedure for checking a client's identification in a certain way [22].

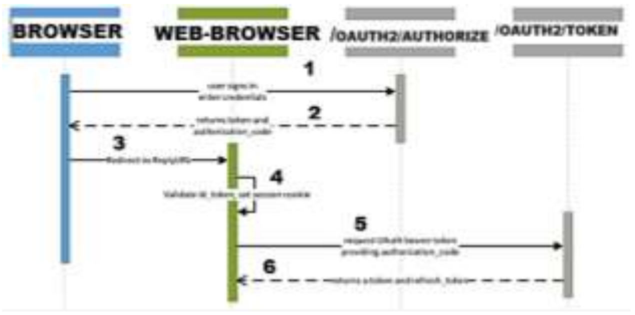

Fig. 5. Authentication Flow using OpenID Connect

As shown in Figure 5, the OpenID Connect authentication flow begins with the user signing in with their credentials. Upon successful authentication, an authorization code and token are returned, and the user is redirected to the ReplyURL, where the id_token is validated, and a session cookie is set[23].

2) *Authorization and Role-Based Access Control*

An app's degree of access and the resources and methods it may call via the API are both controlled by authorization. Authorization may be implemented by the organization via the generation of access tokens or through the use of other protocols and techniques[24].

C. Monitoring and Analytics:

Comprehensive monitoring and analytics capabilities have become essential components of successful API management strategies. Organizations implementing advanced API monitoring solutions have reported significant improvements in their ability to detect and resolve issues proactively[25].

Performance optimization capabilities have become a cornerstone of modern API management platforms, with organizations reporting significant improvements in API reliability and response times[25][26].

V. Challenges in API Management and Integration

API management plays a crucial role in ensuring seamless communication between software applications[27]. However, several challenges arise in managing and integrating APIs effectively. Applications with integrity and security issues cannot be adopted [28]. These challenges must be addressed to ensure an integrity, confidentiality and availability of data within an organization.

- **Data Privacy Concerns:** Data privacy regulations must be complied. The well-known regulations include HIPAA or GDPR. These regulations impose a strong impact on the success or failure of any organization. Compliance with regulation is compulsory for any organization[29].
- **Authentication and Authorization:** RBAC is very important for API security[30]. It ensures that the user has appropriate permission to do anything in the system. The definition and management of roles efficiently is a big challenge[24]. Organizations must establish a check and balance system for role defining and implementation to tackle such challenges.
- **Scalability Challenges:** With the development of API demands, organizations are facing scaling challenges. To face such challenges, organizations should develop and optimize relevant systems to process requests on time[31].
- **Caching Mechanisms:** Caching mechanisms are also a positive system to tackle scalability challenges. Caching means providing responses to requests of the same types through temporary storage. It helps organizations tackle scalability issues and optimize response time.
- **Performance Issues in API Management:** It is also a big concern during the Enterprise Integration of API and API management. APIs are implemented to tackle slow systems and enhance the workability of any organization. If the API doesn't process requests on time and does not use effective strategies to optimize resources, it may develop challenges for the organization.
- **Technical Challenges:** Issues such as security vulnerabilities, performance bottlenecks, scalability concerns, and compatibility problems (including versioning and standardization) that arise when integrating disparate applications via APIs.
- **Process Challenges:** Difficulties related to API lifecycle management, such as designing, testing, deploying, and monitoring APIs in dynamic and often distributed environments[32].
- **Organizational Challenges:** The need for aligning business strategies with technical implementations, ensuring cross-team coordination, and managing changes in an increasingly API-driven digital ecosystem.

VI. Literature Review

This section reviews API management systems, their role in software integration, best practices, challenges, and advancements. Table 2 summarizes key studies.

Dos Santos and Casas (2024) The primary objective of this work was to examine the definitions, measurement methods, and evaluation criteria used to assess API Management software quality, with a particular emphasis on methodological perspectives, API Management capabilities and quality attributes, analyzed and categorized this work from a methodological standpoint in several dimensions, such as research types, outcome types and validation methods. Key quality characteristics included performance efficiency, reliability, functional suitability, and security, encompassing various API management capabilities according to current industry trends[31].

Li et al. (2024) provide a method to identify API abuses by integrating all available API use limitations from various sources, such as client code, API documentation, and library code. Their process starts with translating client code into API using Graphs (AUGs). Then, they look for trends in API use and utilize heuristic filtering criteria to determine API usage constraints. Developers must adhere to certain rules and restrictions while utilizing APIs; otherwise, APIs might be misused[33].

Duan (2023) offers a framework for data-driven smart buildings that is service-oriented. An important obstacle to building scalable, modular, and reusable apps is the absence of an API-rich system architecture. It is quite similar to the headless, cloud-based, API-first, and microservices characteristics of the MACH architecture. As a proof-of-concept, this architecture is put into practice with three smart building applications[34].

Li et al. (2024) provide a method to identify API abuses by integrating all available API use limitations from various sources, such as client code, API documentation, and library code. The next step is to create a number of API preliminary constraint graphs by combining the API use restrictions that were collected from various sources. They develop various solutions for API constraints based on these first API constraint graphs[33].

Jonnada and Joy 2019 provide a structure for validating functionality and conformance. They can improve the design of their interfaces by being able to measure the complexity of APIs; simpler APIs are easier to adopt and integrate. APIs are the latest standard for software program communication. The operation of next-generation technology relies heavily on these interfaces. Their importance in technology and business has been growing substantially due to the extensive use of these interfaces[35].

Kumar et al. (2023) these resources are the foundation upon which the APIs rest. The REST architecture is stateless since no data is stored by either the client or the server. The resources returned by RESTful services are represented in some way. In an attempt to modernize and simplify their systems, some SOAP-oriented websites transitioned to REST implementation. The APIs were discovered to be using basic HTTP verbs devoid of headers. HTTP is the protocol of choice for developing REST web APIs[36].

Table 2: Summary of reviewed studies on API Management Systems and Integration.

References	Study On	Approach	Key Findings	Challenges	Limitations
Dos Santos and Casas, (2024)	API Management software quality	Examined definitions, measurement methods, and evaluation criteria	Identified key API Management capabilities and quality characteristics such as performance efficiency, reliability, functional suitability, and security	Complexity in assessing API quality across different industry standards	Limited scope on real-world API deployments
Li et al., (2024)	API misuse detection	Integrated API usage constraints from client code, API documentation, and library code	Developed API Usage Graphs (AUGs) and heuristic filtering rules to identify API constraints	Handling diverse API constraints from multiple sources	Requires extensive validation across various API environments
Duan, (2023)	Service-oriented system architecture for smart buildings	Implemented a MACH-based API architecture	Demonstrated modularity, scalability, and reusability in smart building applications	Ensuring seamless API integration across heterogeneous environments	Proof-of-concept may not cover all real-world scalability concerns
Li et al. (2024)	API misuse detection (Extended Analysis)	Converted client code into API Usage Graphs (AUGs) and extracted API usage patterns	Created alternate constraint graphs for APIs using methodologies for designing constraints	Combining API usage constraints from multiple sources accurately	Heuristic filtering rules may not cover all API misuse cases
Jonnada and Joy, 2019	API conformity and functionality validation	Proposed a framework for measuring API complexity and usability	Emphasized the importance of quantifying API complexity for better design	Lack of universal standards for API validation	Framework effectiveness may vary based on API design patterns
Kumar et al. (2023)	RESTful APIs and their evolution	Analyzed RESTful services and their transition from SOAP	Found that REST APIs simplify operations by using HTTP verbs without headers	Ensuring security and consistency in RESTful API implementation	Does not cover advanced RESTful API security concerns.

VII. Conclusion and Future Work

API taxonomy and management are essential for seamless software integration, ensuring security, access control, and efficient data exchange. The comparison of API architectures, including SOAP, RPC, and REST, highlights their strengths and application-specific suitability. Effective API management, incorporating gateways, monitoring, and authentication, enhances security and performance. As digital ecosystems evolve, robust API strategies remain critical for interoperability and operational efficiency. Additionally, adopting best practices in API lifecycle management helps organizations enhance scalability, maintainability, and innovation. A well-structured API ecosystem fosters collaboration and accelerates digital transformation across industries. Future work can explore advanced security measures against evolving threats and the integration of AI/ML for real-time monitoring and optimization. Standardized frameworks for API governance and compliance need further development. Additionally, the impact of API management on microservices and serverless computing scalability warrants deeper investigation.

References

[1] L. C. -, "Review of Application Software Used in Educational Research," *Int. J. Multidiscip. Res.*, vol. 5, no. 3, pp. 1–8, 2023, doi: 10.36948/ijfmr.2023.v05i03.2816.

[2] O. O. Efuntade and A. O. Efuntade, "Application Programming Interface (API) And Management of Web-Based Accounting Information System (AIS): Security of Transaction Processing System, General Ledger and Financial Reporting System," *J. Account. Financ. Manag.*, 2023, doi: 10.56201/jafm.v9.no6.2023.pg1.18.

[3] J. Ofoeda, R. Boateng, and J. Effah, "API integration and organizational agility outcomes in digital music platforms: A qualitative case study," *Heliyon*, vol. 10, no. 11, p. e31756, 2024, doi: 10.1016/j.heliyon.2024.e31756.

[4] M. Mudassir and M. Mushtaq, "The role of APIs in modern software development," *World J. Adv. Eng. Technol. Sci.*, vol. 13, no. 01, pp. 1045–1047, 2024.

[5] S. R. Thota, S. Arora, and S. Gupta, "Al-Driven Automated Software Documentation Generation for Enhanced Development Productivity," in *2024 International Conference on Data Science and Network Security (ICDSNS)*, 2024, pp. 1–7. doi: 10.1109/ICDSNS62112.2024.10691221.

[6] S. M. Sohan, C. Anslow, and F. Maurer, "A Case Study of Web API Evolution," in *Proceedings - 2015 IEEE World Congress on Services, SERVICES 2015*, 2015. doi: 10.1109/SERVICES.2015.43.

[7] M. Maleshkova, C. Pedrinaci, and J. Domingue, "Investigating Web APIs on the World Wide Web," in *Proceedings - 8th IEEE European Conference on Web Services, ECOWS 2010*, 2010. doi: 10.1109/ECOWS.2010.9.

[8] C. Severance, "Roy T. Fielding: Understanding the REST Style," *Computer*. 2015. doi: 10.1109/MC.2015.170.

[9] V. S. Thokala, "Integrating Machine Learning into Web

Applications for Personalized Content Delivery using Python," *Int. J. Curr. Eng. Technol.*, vol. 11, no. 06, 2021, doi: https://doi.org/10.14741/ijcet/v.11.6.9.

[10] M. Lamothe, Y. G. Guéhéneuc, and W. Shang, "A Systematic Review of API Evolution Literature," *ACM Comput. Surv.*, vol. 54, no. 8, 2022, doi: 10.1145/3470133.

[11] H. Luzern, *Communications in Computer and Information Science*, no. March. 2017. doi: 10.1007/978-981-16-9229-1.

[12] Vasudhar Sai Thokala, "Efficient Data Modeling and Storage Solutions with SQL and NoSQL Databases in Web Applications," *Int. J. Adv. Res. Sci. Commun. Technol.*, vol. 2, no. 1, pp. 470–482, Apr. 2022, doi: 10.48175/IJARSCT-3861B.

[13] M. R. S. Vishwakarma, Pawan Kumar, "A Study on Energy Management Systems (EMS) in Smart Grids Industry," *IJRAR*, vol. 10, no. 2, pp. 558–563, 2023.

[14] M. Mathijssen, M. Overeem, and S. Jansen, "Identification of Practices and Capabilities in API Management: A Systematic Literature Review," no. June, 2020, doi: 10.48550/arXiv.2006.10481.

[15] A. V. V Sudhakar, A. Tyagi, P. S. Patil, K. S. Kumar, M. Sathe, and B. T. Geetha, "Cognitive Computing in Intelligent Traffic Management Systems," in *2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)*, 2024, pp. 1–6. doi: 10.1109/ACCAI61061.2024.10602427.

[16] S. Moiz Ali and T. Rahim Soomro, "Comparative Study of API Management Solutions," *6th Int. Conf. Innov. Sci. Technol.*, pp. 41–54, 2019, doi: 10.33422/6th-istconf.2019.07.411.

[17] M. Mathijssen, M. Overeem, and S. Jansen, "Identification of Practices and Capabilities in API Management: A Systematic Literature Review," 2020.

[18] A. and P. Khare, "Cloud Security Challenges : Implementing Best Practices for Secure SaaS Application Development," *Int. J. Curr. Eng. Technol.*, vol. 11, no. 6, pp. 669–676, 2021, doi: https://doi.org/10.14741/ijcet/v.11.6.11.

[19] A. Goyal, "Optimising Cloud-Based CI/CD Pipelines: Techniques for Rapid Software Deployment," *Tech. Int. J. Eng. Res.*, vol. 11, no. 11, pp. 896–904, 2024.

[20] N. Abid, "Improving Accuracy and Efficiency of Online Payment Fraud Detection and Prevention with Machine Learning Models," *Int. J. Innov. Sci. Res. Technol.*, vol. 9, no. 12, pp. 711–723, 2024.

[21] V. S. Thokala, "A Comparative Study of Data Integrity and Redundancy in Distributed Databases for Web Applications," *IJRAR*, vol. 8, no. 4, pp. 383–389, 2021.

[22] M. Mathijssen, M. Overeem, and S. Jansen, "Source Data for the Focus Area Maturity Model for API Management," no. August, 2020, doi: 10.48550/arXiv.2007.10611.

[23] J. Vijaya Chandra, N. Challa, and S. K. Pasupuletti, "Authentication and authorization mechanism for cloud security," *Int. J. Eng. Adv. Technol.*, 2019, doi: 10.35940/ijeat.F8473.088619.

[24] A. Coates, M. Hammoudeh, and K. G. Holmes, "Internet of things for buildings monitoring: Experiences and challenges," *ACM Int. Conf. Proceeding Ser.*, vol. Part F130522, 2017, doi: 10.1145/3102304.3102342.

[25] R. L. Pinheiro, D. Landa-Silva, R. Qu, E. Yanaga, and A. A. Constantino, "Towards an efficient API for optimisation problems data," *ICEIS 2016 - Proc. 18th Int. Conf. Enterp. Inf. Syst.*, vol. 2, no. December, pp. 89–98, 2016, doi: 10.5220/0005915800890098.

[26] N. Xie, "Strategic approaches to API design and management," *Appl. Comput. Eng.*, vol. 64, no. 1, pp. 230–236, 2024, doi: 10.54254/2755-2721/64/20241395.

[27] A. Goyal, "Optimising Software Lifecycle Management through Predictive Maintenance : Insights and Best Practices," *Int. J. Sci. Res. Arch.*, vol. 07, no. 02, pp. 693–702, 2022.

[28] M. Gopalsamy, "An Optimal Artificial Intelligence (AI) technique for cybersecurity threat detection in IoT Networks," *Int. J. Sci. Res. Arch.*, vol. 07, no. 02, pp. 661–671, 2022.

[29] F. Hussain, R. Hussain, B. Noye, and S. Sharieh, "Enterprise API Security and GDPR Compliance: Design and Implementation Perspective," *IT Professional*. 2020. doi: 10.1109/MITP.2020.2973852.

[30] S. Murri, "Data Security Environments Challenges and Solutions in Big Data," *Int. J. Curr. Eng. Technol.*, vol. 12, no. 6, pp. 565–574, 2022.

[31] E. dos Santos and S. Casas, "Unveiling Quality in API Management: A Systematic Mapping Study," in *2024 L Latin American Computer Conference (CLEI)*, 2024, pp. 1–10. doi: 10.1109/CLEI64178.2024.10700447.

[32] S. Andreo and J. Bosch, "API management challenges in ecosystems," *Lect. Notes Bus. Inf. Process.*, vol. 370 LNBIP, no. October 2019, pp. 86–93, 2019, doi: 10.1007/978-3-030-33742-1_8.

[33] C. Li, J. Zhang, Y. Tang, Z. Li, and T. Sun, "Boosting API Misuse Detection via Integrating API Constraints from Multiple Sources," in *2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR)*, 2024, pp. 14–26.

[34] Z. Duan, "Application development exploration and practice based on LangChain+ChatGLM+Rasa," in *2023 2nd International Conference on Cloud Computing, Big Data Application and Software Engineering, CBASE 2023*, 2023. doi: 10.1109/CBASE60015.2023.10439133.

[35] S. Jonnada and J. K. Joy, "Measure your API complexity and reliability," in *Proceedings - 2019 IEEE/ACIS 17th International Conference on Software Engineering Research, Management and Application, SERA 2019*, 2019. doi: 10.1109/SERA.2019.8886790.

[36] K. Kumar, A. K. Jain, R. G. Tiwari, N. Jain, V. Gautam, and N. K. Trivedi, "Analysis of API Architecture: A Detailed Report," in *Proceedings - 2023 12th IEEE International Conference on Communication Systems and Network Technologies, CSNT 2023*, 2023. doi: 10.1109/CSNT57126.2023.10134658.