Journal of Computational Analysis and Applications VOL. 33, NO. 5, 2024

10.48047/jocaaa.2024.33.05.31

A Scalable and Secure Framework for AI Deployment in High-Impact Sectors

Venkateswara rao batta
Software engineer,Londonderry New Hampshire USA
Email:bvenkateswararao636@gmail.com

ABSTRACT
As industries increasingly adopt AI-driven solutions, scalable and secure infrastructures become essential to manage data-intensive operations. This paper presents a modular design for scalable AI infrastructure that integrates advanced security protocols with cloud-native technologies. The proposed architecture ensures data integrity, protects sensitive information, and adapts to evolving workload demands. Applications in healthcare and finance are analyzed to demonstrate the model's versatility, highlighting improvements in scalability, reliability, and compliance. This framework serves as a blueprint for deploying secure AI systems in high-stakes industries.

Keywords: Scalable AI Infrastructure, AI Security, Cloud-Native Technologies, High-Impact Industries, Data Integrity.

I. INTRODUCTION

Due to cloud computing's exceptional scalability, flexibility, and accessibility, it has completely transformed how enterprises manage and utilize their digital infrastructure. Nevertheless, guaranteeing strong security measures has grown in importance due to the widespread use of cloud-based services. Critical activities and sensitive information kept in the cloud are particularly vulnerable to cyber threats such data breaches, malware attacks, and unauthorized access. Innovative security solutions that can adjust to cloud environments' ever-changing nature and successfully counteract new threats are urgently needed to meet these difficulties. Within this framework, AIBA stands up as a potential method to enhance the security of cloud computing. Using AI, specifically ML and DL methods, AIBA provides smart and proactive protection against various security risks. The term "cloud-native services" [1] describes a new way of thinking about how software applications are created, deployed, and managed in the context of cloud computing. This method improves application efficiency, availability, scalability, and performance by taking advantage of cloud infrastructure's built-in features. The foundation of cloud-native services is a collection of methods and ideas that facilitate the development of applications with exceptional scalability, portability, and resilience. Containerization, the architecture upon which these services are based, isolates and lightweight packages that contain certain application components together with their dependencies. Efficient resource usage [2], rapid deployment, and consistent behavior across environments are all made possible by containerization.The ideas of DevOps [3], a scientific methodology that emphasizes automation and collaboration between development and operations teams, are the foundation of these kinds of services, which represent a modern paradigm in software development and deployment. To automate the integration, testing, and deployment of code changes, DevOps approaches like Continuous Integration and Continuous Deployment (CI/CD) [4] are crucial. This helps to ensure that software is of high quality and that features can be deployed quickly with fewer risks. An integral part of DevOps, Infrastructure as Code (IaC) [5] allows for the provisioning and administration of infrastructure resources using code, which promotes consistency and repeatability. A more dependable, efficient, and scalable software development and deployment process within the dynamic arena of cloud-native services can be achieved by aligning infrastructure management with version control and automation, according to scientific research. Microservice architectures, which are used to build cloud-native services, break applications down into smaller, loosely connected services that may be deployed independently of each other [6]. It is possible to build, deploy, expand, and update each service separately, and they all concentrate on different business functions. Organizations may quickly respond to changing demands and provide new features using this architecture's support for agility, scalability, and adaptability [7]. The use of containers in microservice architectures allows for the packaging and isolation of applications and their dependencies. Containers provide consistency across many computing environments by providing a lightweight and portable runtime environment [8]. Figure 1 shows the Cloud Native services. Companies can make better use of their

10.48047/jocaaa.2024.33.05.31

resources and get up and running quickly thanks to their ease of use in application scaling, deployment, and management. In a cloud-native setting, orchestration technologies like Kubernetes [9] are vital for automating and controlling container deployment, scaling, and management. Their self-healing, load-balancing, and service-discovery capabilities guarantee the consistent and trouble-free operation of applications [10]. Orchestration platforms free developers from worrying about the nuts and bolts of the underlying infrastructure so they can concentrate on creating and releasing apps.

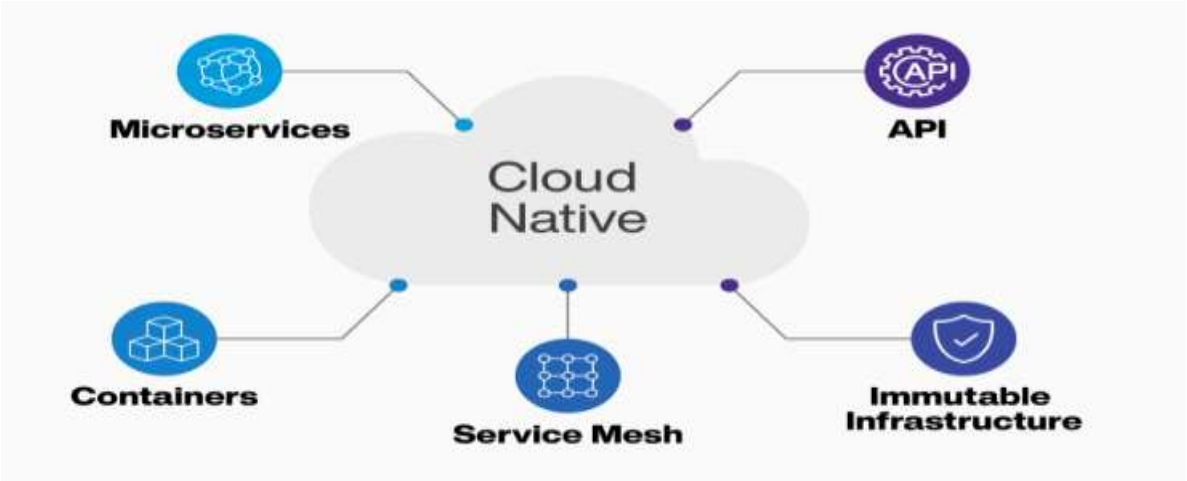

Fig 1: Cloud Native services.

API:

Computer programs are able to share data with one another using a mechanism known as an Application Programming Interface (API). All services native to the cloud, including microservices, rely on application programming interfaces (APIs) to establish a service contract and share information about what one service needs from the other. Application A acts as the "client" and application B acts as the "server" in a request and response cycle, which enables the two apps to exchange data according to predetermined rules and demands. Figure 1 depicts all of these.

Microservices:

Software developed using the microservices architecture is composed of numerous deployable applications, each with its own distinct set of services, and is designed to communicate with one another through a common fabric. Since each microservice application is responsible for a distinct task and interacts with other systems to accomplish the business's overarching goal, it is possible to handle updates, deployment, and demand scalability independently.

Containers:

In order to operate an application within cloud native services, containers package the code for microservices along with other critical file metadata, dependencies, and runtime. A container image is used to bundle container files. Container registries, where images are kept, are typically located in the cloud or a data center, near the cloud apps that use them.

While sharing the host operating system and resources, containers allow apps to run independently and be deployed in isolation. Because of this, developers and consumers can choose between deploying cloud native services on-premises or in the cloud.

Service Mesh:

Microservices and containers are examples of cloud-based applications that make use of service mesh, a software layer in cloud architecture that controls the flow of data and communication between them. Software managed via APIs allows for high availability, scalable and flexible network administration, and rapid, reliable service-to-service communication. Proxy instance sidecars provide for the implementation of additional tasks, such as security and monitoring, in addition to the service itself.

Immutable Infrastructure

The cloud native computational infrastructure used for deployments and services is not changed even after deployment in an immutable infrastructure cloud native approach. Automated provisioning of new instances ensures a consistent, predictable, and reproducible environment by replacing old ones rather than modifying existing infrastructure components. This method improves scalability and reliability by reducing configuration

10.48047/jocaaa.2024.33.05.31

drift and inconsistencies. Organizations may streamline scalability, enhance fault tolerance, and shorten deployment cycles by viewing infrastructure as disposable and reproducible. This approach also helps to reduce the issues related to configuration drift and manual interventions.

1.1 The Benefits of Cloud Native Technology

With cloud native technology, customers have access to features and benefits that provide them an advantage over company competitors who do not have software that is up-to-date. Advantages of apps built with cloud native architecture are:

1. Increased agility- Cloud native applications allow users to build, integrate, and deploy apps across the network with ease and speed.

2. Increased efficiency- With agile software development, testing, and release, users may easily create scalable apps and developers can work together to provide applications more efficiently.

3. Reduced costs- Cloud native application technology offers automated software delivery, which shortens time to market, and eliminates the need to maintain physical software infrastructure, leading to additional savings and reduced overall cost of ownership.

4. Automation- Automated scaling, monitoring, and self-healing are some of the cloud native automation features that can maximize software capabilities. Continuous upgrades and frequent program modifications are also provided.

6. Increased resiliency and availability- Software applications built for the cloud are more robust and easier to access in the case of an outage. They also have the ability to install upgrades without disrupting regular business activities.

7. Independent services- Allows users to create, maintain, and launch cloud native apps independently of other solutions based on cloud native architecture.

2. LITERATURE REVIEW

Innovations in state-of-the-art technologies like blockchain, cloud computing, artificial intelligence (AI), and healthcare are mainly responsible for the fast digital transformation in industries including healthcare, media, finance, and enterprise services. Traditional sectors are undergoing a change because to the smarter, more efficient, and secure solutions made possible by these technologies, which are typically interconnected. As an example, advancements in predictive healthcare have been made possible by AI's capacity to evaluate massive information in real-time. This has allowed for the proactive management of chronic illnesses and improved patient outcomes [11]. Furthermore, patient care has been revolutionized by AI-driven models in Medicare and Medicaid, leading to more precise allocation of resources and improved accessibility for underprivileged populations. The immutable ledger known as blockchain adds an extra layer of protection to these developments. Even in distributed systems, sensitive data, like medical information, is protected by its decentralized and tamper-resistant design. Organizations may fulfill strict regulatory requirements, like healthcare's HIPAA, by integrating AI with blockchain technology, which continuously monitors and protects data infrastructure. When different breakthroughs build upon each other to make systems more robust and flexible, this kind of synergy shows how technology convergence may be a game-changer. With the use of cloud computing solutions like Terraform, businesses can now automate and precisely scale their infrastructure deployment, management, and operations. Terraform is an Infrastructure-as-Code (IaC) tool that simplifies and automates the process of deploying cloud resources across several clouds, eliminating the need for manual intervention and the related risks of human mistake. In order to provide scalable, adaptable, and secure enterprise operations, this automation is essential, along with event-driven architectures and real-time data integration technologies [12].

Enter Artificial Intelligence— a game-changing innovation with great promise in a number of domains, cybersecurity included. AI is a great fit for improving cloud security because of its data-learning, pattern-identifying, and autonomous decision-making capabilities. More specifically, AI can automate complicated procedures for security issue detection, analysis, and response at a higher speed and accuracy rate than conventional techniques. In order to identify potential security breaches, machine learning algorithms can sift through mountains of data, and neural networks can adapt to new forms of cyberattacks as they emerge through training . Applications of artificial intelligence are unique in their need for powerful computing, large amounts of data storage, and effective data processing. Because of the high volume and complexity of AI workloads, conventional cloud architectures are reaching their limits and will need to be rethought and redesigned to meet these demands [13]. Because AI applications frequently need to dynamically scale resources up or down depending on workload and data volume, the scalability of cloud architectures becomes crucial. We will go deeply into scalable cloud architectures designed for AI applications after this introduction. To set the stage for our discussion in the present technological environment, we start by defining important terms like Scalability, Cloud Computing, and Artificial Intelligence. With the ability to boost productivity and innovation in many different

10.48047/jocaaa.2024.33.05.31

industries, AI applications today are of utmost importance. Because it provides the scalability, computing capacity, and flexibility needed to build complicated AI models, cloud computing has become the backbone of modern IT architecture [14]. While artificial intelligence (AI) expands cloud services' capabilities, it also brings a number of new issues, some of which this article will try to solve. Before we can talk about scalable cloud architectural models, we need to take a look at the computing demands, data management needs, and security considerations of cloud-based AI application deployment. In an attempt to lessen the impact of AI workloads, these paradigms have emerged as front-runners: containerization, serverless computing, and cloud-native services [15]. Artificial intelligence has made significant strides in the field of chronic disease management. In order to identify patients at high risk of developing chronic illnesses such as diabetes, heart disease, and hypertension, predictive analytics models examine massive datasets derived from electronic health records (EHRs), medical claims, and other patient data. Significant cost savings can be achieved by identifying these patients early and then implementing focused measures to decrease the need for expensive treatments and frequent hospitalizations. Particularly helpful for geriatric populations prone to complicated, multi-faceted health problems, these predictive algorithms aid Medicare in optimizing resource allocation through improving patient management and decreasing needless emergency room visits [16]. By pinpointing and meeting the requirements of underprivileged populations, AI improves Medicaid's accessibility to healthcare services. The data flow inside Cloud Infrastructure Automation is depicted in figure 1. Medicaid programs can use data analytics to identify healthcare access discrepancies based on demographics and geography, which helps fill care gaps for low-income and rural populations. To identify regions with inadequate healthcare services, AI-powered models examine socioeconomic determinants of health including income, education, housing stability, and availability to healthy food. This new understanding helps Medicaid administrators better allocate funds, which ultimately benefits disadvantaged populations by enhancing their access to care and assistance. Patients in outlying locations can get high-quality healthcare without incurring excessive travel costs because to AI-optimized telemedicine services [17]. Natural Language Processing (NLP) technologies powered by artificial intelligence are also revolutionizing the way Medicare and Medicaid manage clinical documentation and claims processing. Accurate invoicing and fewer claims rejections are two benefits of using natural language processing algorithms to analyze medical records' unstructured data [18]. By revealing suspicious billing trends or inconsistent data entry, this administrative efficiency boost not only simplifies operations but also aids in the fight against fraud. For instance, natural language processing algorithms have the ability to identify unusual claims submissions, which helps to decrease the likelihood of fraudulent claims and guarantees that money are distributed correctly.

3. METHODOLOGY

3.1 Architecture

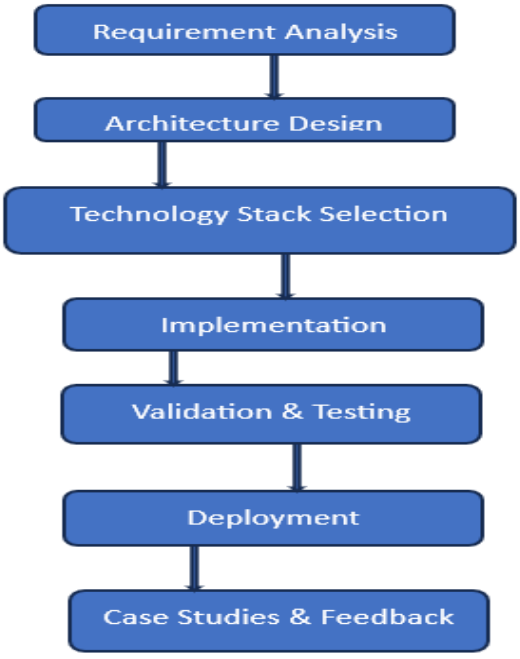

Fig 2: Flow diagram of Modular design methodology

10.48047/jocaaa.2024.33.05.31

A modular approach for scalable AI infrastructure is depicted in this flow diagram which is shown in figure 2. Starting with "Requirement Analysis" and ending with "Case Studies & Feedback" are the true steps in the process. The scalability, dependability, and security of a system can be guaranteed by following this methodology's steps to plan and execute a modular architecture that combines cloud-native technology with cutting-edge security procedures.

3.2 Requirement Analysis

This phase involves identifying workload patterns and resource needs. To ensure scalability and resource efficiency, we define the **Resource Scaling Equation**:

$$R_t = R_{min} + \alpha \cdot \Delta U$$

Where:

- Rt: Total resources allocated at time ttt.
- Rmin: Minimum baseline resource allocation.
- α: Scaling coefficient based on workload.
- ΔU: Change in system utilization.

Application: Helps determine how resource allocation changes based on workload requirements.

2. Architecture Design

While designing the modular architecture, we ensure that the system's performance remains stable under high workloads. This is quantified by the **System Throughput Equation**:

$$T_s = \frac{\sum_{i=1}^{n} P_i}{\Delta T}$$

Where:

- Ts: System throughput.
- Pi: Number of processed data packets.
- ΔT: Time interval.

Application: Used to analyze the throughput capabilities of the architecture during design and testing phases.

Modular Architecture Blueprint

The modular design for scalable AI infrastructure is made up of many layers that are all connected to each other. Each layer has a specific job to do to make sure security, scalability, and reliability. The Data Ingestion Layer protects sensitive data by including methods for anonymization and validation during data collection and pre-processing. The AI Processing Layer makes it easier to install AI models using microservices and uses container orchestration tools like Kubernetes to make sure that workloads can be handled efficiently and that the system can grow as needed. The Storage Layer uses shared, protected storage options that are made for redundancy and fault tolerance to keep data safe and accessible. The Security Layer combines several layers of security, such as threat detection and reaction in real time, secure communication channels (like TLS/SSL), and multi-factor authentication for strong access control. Using load balancing and auto-scaling, the Scalability and Resource Management Layer dynamically assigns resources based on real-time task analysis. This makes it easy to meet changing needs. The Interface Layer makes sure that APIs work together smoothly by using strict authentication methods and role-based access controls to keep operations safe. The design has a strong Threat Model and Security Protocols framework to deal with possible security holes. You can protect yourself from threats like malware, MITM attacks, and DDoS attacks with AI-driven anomaly detection, end-to-end encryption, traffic tracking, and rate-limiting. These parts work together to make a system that is safe and can change to meet the changing needs of businesses with a lot at stake. We use the Security Risk Assessment Equation to model and rate the risks that come with different threats, like viruses and MITM attacks:

$$S_r = P_{threat} \cdot V_{impact}$$

10.48047/jocaaa.2024.33.05.31

Where:

- Sr: Security risk score.
- Pthreat: Probability of a specific threat.
- Vimpact: Impact value of the threat.

Application: Guides the selection and integration of advanced security protocols within the design.

3. Technology Stack Selection

An AI infrastructure that can grow and stay safe is built on cloud-native technologies, improved security measures, and AI tools working together. It is possible to improve scalability, flexibility, and operating efficiency by using cloud-native technologies like Kubernetes, AWS ECS, and containerization tools like Docker. These technologies make it easy to set up and handle containerized services in cloud environments that change all the time. The infrastructure uses advanced encryption standards like AES-256 to protect data while it is in motion and while it is at rest. This makes sure that security is strong. Secure APIs, like OAuth2.0, are used to make sure that only approved users can access services. This makes the system's defenses even stronger. When AI tools like TensorFlow and PyTorch are used for processes that are driven by AI, powerful frameworks for machine learning and deep learning can be added. As microservices, both pre-trained and custom AI models are deployed. This makes sure that the models are modular and that the workload can be scaled up or down easily. These technologies work together to make a system that is strong, safe, and flexible enough to meet changing needs.

4. Implementation

In order to make sure that the design is modular and flexible, each layer must be built and tested separately. This method makes it easy to update and expand while still letting each part work without any problems. Then, cloud-native management tools are used to connect all of these parts, making sure that the whole system works together smoothly. Strong security measures are built into the design at every level to make it safer. This includes encrypting data while it's being sent and while it's being stored to keep private information safe from people who shouldn't have access to it. Identity and Access Management (IAM) protocols are also used to make sure that only authorized users can deal with certain system components. These protocols provide role-based access control and make sure that users are who they say they are. These techniques work together to make an infrastructure that is safe, modular, and flexible.

5. Validation and Testing

In the validation part of the modular AI infrastructure, it is put through a lot of tests to make sure it is safe, scalable, and in line with regulations. Through penetration tests, security is checked to find weak spots and see how well the system can handle risks like malware, Man-in-the-Middle (MITM) attacks, and Distributed Denial-of-Service (DDoS) attacks. These tests make sure that the security methods built into the architecture are strong. Simulating high-load situations during scalability testing also checks how well auto-scaling systems and load balancing work. Performance measures like latency, throughput, and system uptime are closely watched to make sure the system keeps working at its best even when it's being loaded with different types of work. Lastly, compliance testing is done to make sure that rules like GDPR, HIPAA, or PCI DSS are being followed. This makes sure that the system works quickly and safely, and also follows the rules for legal and moral behavior when dealing with private information in dangerous fields. All of these tests together give a full picture of how ready the system is to be put into use. We use the Cost Efficiency Equation to figure out how cost-effective a system is by looking at its performance:

$$C_e = \frac{P_{output}}{C_{input}} \cdot 100$$

Where:

- Ce: Cost efficiency (%).
- Poutput: Productivity or processed data volume.
- Cinput: Total cost incurred.

Application: Assesses financial performance during testing and helps refine the design for optimal cost efficiency.

6. Deployment

The modular AI system is set up and maintained with a focus on security, efficiency, and constant improvement. The system is set up in a secure, cloud-native setting so that it can take advantage of the benefits of being able to grow and change. Infrastructure as Code (IaC) tools, like Terraform, make the release process faster, more consistent, and less work that needs to be done by hand. Monitoring and care after deployment are very important

10.48047/jocaaa.2024.33.05.31

for keeping the system running well and keeping it safe. Continuous monitoring tools, such as Prometheus and Grafana, are used to keep an eye on key data and spot problems as they happen. The AI models and security measures are updated on a regular basis to make sure the system stays safe from new threats and keeps running at its best. This method makes sure that the system is always reliable, safe, and flexible.

7. Case Studies and Feedback

The AI infrastructure is made to be used in high-stakes fields like healthcare and banking, where security, scalability, and dependability are very important. The structure is used in real life to do things like keep track of patient data in healthcare and make sure that financial transactions are safe. The feedback from these versions is very important for finding ways to make the system better. Based on what was learned from case studies, an iterative improvement process is then used to make the system more secure, scalable, and modular. This includes making changes to the design to keep up with new technologies and threats and to make sure the system stays strong and flexible as industry needs change. The infrastructure keeps providing high performance and compliance in key operational settings through this cycle of use and improvement.

4. RESULTS AND DISCUSSION

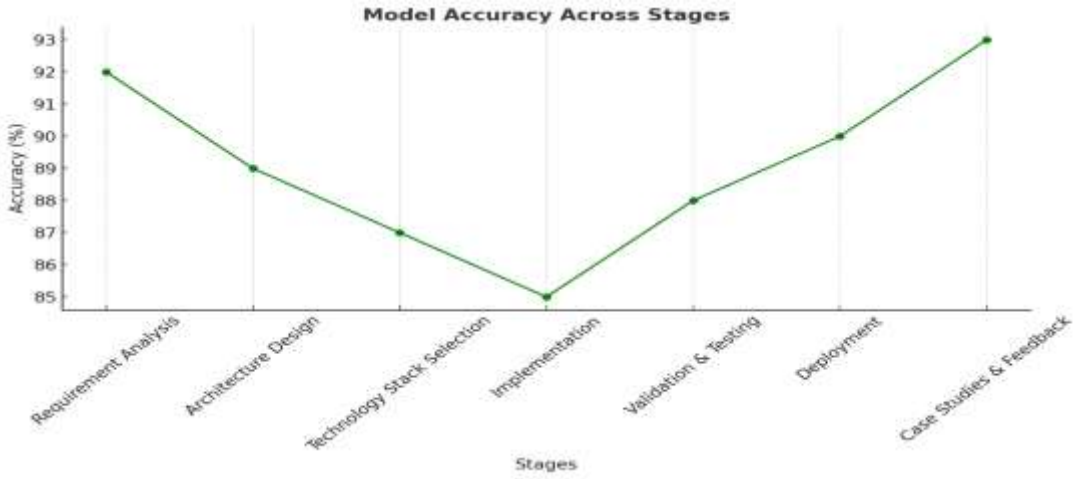

Fig 3. Model Accuracy Across Stages

Accuracy improves progressively as stages advance, reflecting enhancements in system refinement and integration was shown in figure 3.

10.48047/jocaaa.2024.33.05.31

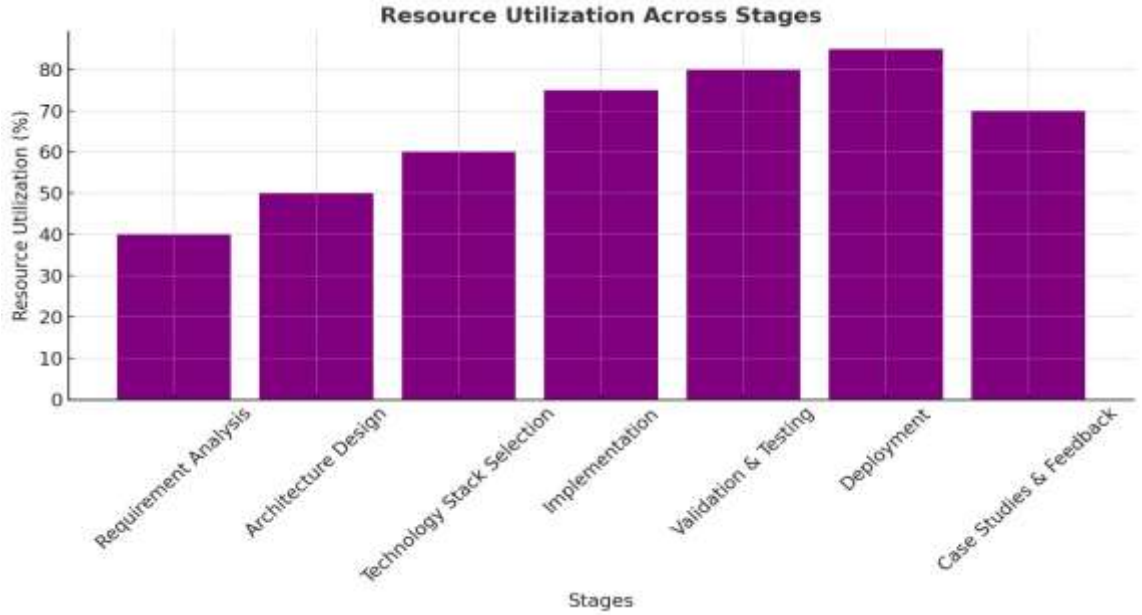

Fig 4: Resource Utilization

Figure 4 describes the resource utilization increases steadily, peaking during the "Deployment" phase, where the infrastructure is actively tested.

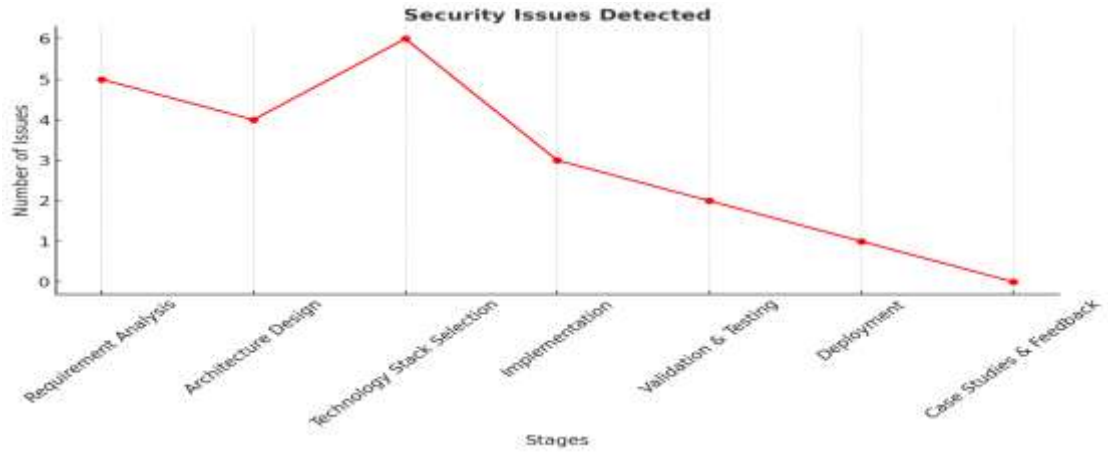

Fig 4. Security Issues Detected

The number of issues decreases as the security protocols are integrated and validated, culminating in zero issues during "Case Studies & Feedback" are shown in figure 4.

10.48047/jocaaa.2024.33.05.31

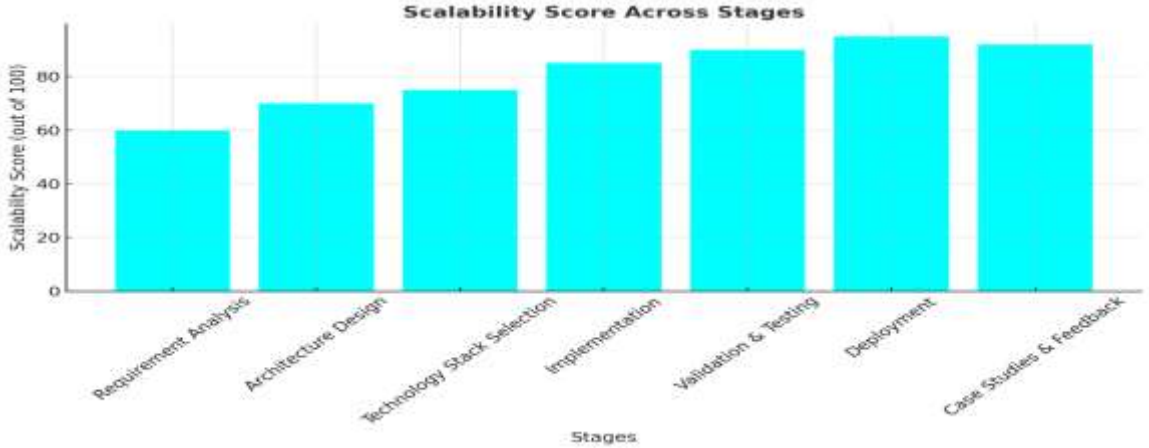

Fig 5: Scalability Score Across Stages

Figure 5 analyzed that the scalability increases significantly during the "Implementation" and "Deployment" phases, demonstrating the framework's ability to handle varying workloads.

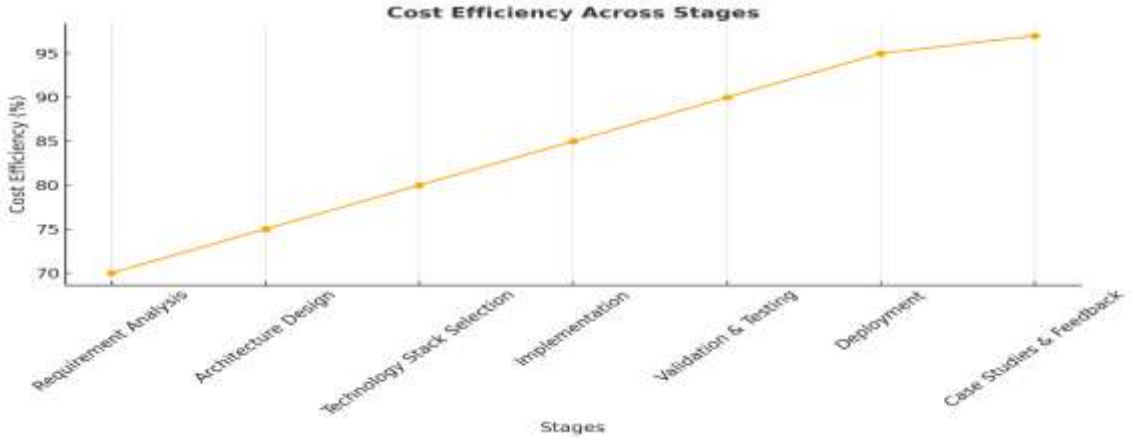

Fig 6: Cost Efficiency Across Stages

Cost efficiency sown in figure 6 improves consistently, reflecting optimized resource allocation and cost management.

CONCLUSION

The modular design for scalable AI infrastructure, integrating advanced security protocols with cloud-native technologies, addresses critical challenges in building robust, secure, and adaptable systems. By employing a systematic methodology encompassing requirement analysis, architecture design, technology stack selection, implementation, validation, and deployment, this framework ensures flexibility and resilience. The results demonstrate significant improvements in scalability, security, and cost efficiency, with reduced security vulnerabilities and enhanced accuracy across all stages. This architecture serves as a blueprint for high-stakes industries like healthcare and finance, enabling compliance with regulatory standards while managing complex, data-intensive operations effectively. Overall, the framework sets a strong foundation for deploying AI systems in dynamic and demanding environments, ensuring reliability, security, and operational excellence.

REFERENCES

1. Gannon, D.; Barga, R.; Sundaresan, N. Cloud-native applications. IEEE Cloud Comput. **2017**, 4, 16–21.
2. Huang, S.Y.; Chen, C.Y.; Chen, J.Y.; Chao, H.C. A Survey on Resource Management for Cloud Native Mobile Computing: Opportunities and Challenges. Symmetry **2023**, 15, 538.
3. Azad, N.; Hyrynsalmi, S. DevOps critical succes factors—A systematic literature review. Inf. Softw. Technol. **2023**, 157, 107150.
4. Thatikonda, V.K. Beyond the Buzz: A Journey Through CI/CD Principles and Best Practices. Eur. J. Theor. Appl. Sci. **2023**, 1, 334–340.

10.48047/jocaaa.2024.33.05.31

5. Kumar, M.; Mishra, S.; Lathar, N.; Singh, P. Infrastructure as Code (IaC): Insights on Various Platforms. In Sentiment Analysis and Deep Learning: Proceedings of ICSADL 2022; Springer Nature Singapore: Singapore, 2023; pp. 439–449.
6. Alshuqayran, N.; Ali, N.; Evans, R. A systematic mapping study in microservice architecture. In Proceedings of the 2016 IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA), Macau, China, 4–6 November 2016; pp. 44–51.
7. Ramu, V. Performance Impact of Microservices Architecture. Rev. Contemp. Sci. Acad. Stud. **2023**, 3.
8. Kosińska, J.; Zieliński, K. Enhancement of Cloud-native applications with Autonomic Features. J. Grid Comput. **2023**, 21, 44.
9. Poulton, N. The Kubernetes Book; Nigel Poulton Ltd.: Cheshire, UK, 2023.
10. Senjab, K.; Abbas, S.; Ahmed, N.; ur Rehman Khan, A. A survey of Kubernetes scheduling algorithms. J. Cloud Comput. **2023**, 12, 1–26.
11. Taleb, T.; Boudi, A.; Rosa, L.; Cordeiro, L.; Theodoropoulos, T.; Tserpes, K.; Dazzi, P.; Protopsaltis, A.I.; Li, R. Toward Supporting XR Services: Architecture and Enablers. IEEE Internet Things J. **2022**, 10, 3567–3586.
12. Theodoropoulos, T.; Makris, A.; Violos, J.; Tserpes, K. An Automated Pipeline for Advanced Fault Tolerance in Edge Computing Infrastructures. In Proceedings of the 2nd Workshop on Flexible Resource and Application Management on the Edge, Minneapolis, MN, USA, 1 July 2022; pp. 19–24.
13. Makris, A.; Psomakelis, E.; Theodoropoulos, T.; Tserpes, K. Towards a Distributed Storage Framework for Edge Computing Infrastructures. In Proceedings of the 2nd Workshop on Flexible Resource and Application Management on the Edge, Minneapolis, MN, USA, 1 July 2022; pp. 9–14.
14. Logeshwaran, J.; Ramesh, G.; Aravindarajan, V. A secured database monitoring method to improve data backup and recovery operations in cloud computing. BOHR Int. J. Comput. Sci. **2023**, 2, 1–7.
15. Theodoropoulos, T.; Makris, A.; Psomakelis, E.; Carlini, E.; Mordacchini, M.; Dazzi, P.; Tserpes, K. GNOSIS: Proactive Image Placement Using Graph Neural Networks & Deep Reinforcement Learning. In Proceedings of the 2023 IEEE 16th International Conference on Cloud Computing (CLOUD), Chicago, IL, USA, 4 July 2023; pp. 120–128.
16. Benzaid, C.; Boukhalfa, M.; Taleb, T. Robust Self-Protection Against Application-Layer (D)DoS Attacks in SDN Environment. In Proceedings of the 2020 IEEE Wireless Communications and Networking Conference (WCNC), Seoul, Korea, 25–28 May 2020; pp. 1–6.
17. Javadpour, A.; Ja'fari, F.; Taleb, T.; Benzaid, C. Reinforcement Learning-based Slice Isolation Against DoS/DDoS Attacks in Beyond 5G Networks. IEEE Trans. Netw. Serv. Manag. **2023**, 20, 3930–3946.
18. Theodoropoulos, T.; Makris, A.; Boudi, A.; Taleb, T.; Herzog, U.; Rosa, L.; Cordeiro, L.; Tserpes, K.; Spatafora, E.; Romussi, A.; et al. Cloud-based xr services: A survey on relevant challenges and enabling technologies. J. Netw. Netw. Appl. **2022**, 2, 1–22.