Isabela State University Linker:
Journal of Engineering, Computing, and Technology
Volume 1, Issue 2
ISSN (Print): 3082-3676 ISSN (Online): 3082-3684
DOI: https://doi.org/10.65141/ject.v1i2.n8

Benefits and Challenges of Adopting Serverless Architectures and Function-as-a-Service (FaaS) for Developing and Deploying Web Services

Jessa A. Revilla-Dave[1], Edward Panganiban[2]
Atticus Advisory Solutions, Pasig City, Philippines [1]
College of Computing Studies, Information, and Communication Technology, Isabela State University-Main Campus, Echague, Isabela, 3309, Philippines [2]
jessarevilla.16@gmail.com

RESEARCH ARTICLE INFORMATION

Received: June 22, 2024
Reviewed: November 8, 2024
Accepted: December 28, 2024
Published: December 31, 2024

ABSTRACT

Web services are everywhere in today's technology acting as the backbone for many applications across diverse industries. It affects how applications are planned, presented, created, and implemented. Serverless architecture promotes flexibility in software development, allowing developers to design and develop components independently and scale them horizontally on demand. This approach has given rise to Function-as-a-Service (FaaS) in the world of cloud computing. FaaS and serverless architecture both support agility, can quickly cycle in the process of software or application development, and seamlessly integrate services and functions into existing workflows, thus having minimal effort or without overhauling the entire technology stack from top to bottom. It empowers organizations to innovate their technology stack to meet today's demands and/or business needs. On the other hand, implementing innovative systems with cutting-edge technology has its difficulties such as but not limited to complexity in administration, volatility in performance, and vendor lock-in. Highlighting the key trade-offs between system performance and agility to emphasize balancing the organization's decisions in adapting a serverless architecture, this paper review aimed to examine the benefits and challenges of serverless architectures and FaaS paradigms in the context of web service development and deployment. Strategies for mitigating vendor lock-in and

approaches to optimizing performance without sacrificing scalability are critical for organizations striving to enhance their systems. These strategies provide organizations with valuable insights and practical guidance, serving as a roadmap for navigating serverless computing while addressing common pitfalls to maximize its potential.

Keywords: *Cloud computing, function-as-a-service, serverless architecture, web services*

Introduction

Cloud computing opens a new era of web service development and deployment; one of its characteristics is the rise of serverless architecture and FaaS (Leitner et al., 2019). Serverless computing increased its impact on our modern society with its groundbreaking technology, thus its adoption is rising in both academia and industry circles, showcasing a very meaningful transformational capability (Eismann et al., 2022). Operating costs, inefficient resource use, and complex scalability requirements are the significant challenges of traditional server-based systems. These limitations make them unsuitable for meeting rapidly changing industry needs. If these inefficiencies are not fixed, infrastructure costs may increase, organizational efficiency may be reduced, and reaction times to market demands may be prolonged. These effects may lead to a loss of competitive advantage and a drop in customer satisfaction.

Moreover, serverless architecture presents a framework in cloud computing that allows the developers to focus on writing code related to business processes without the burden of managing infrastructure. Resources can be allocated dynamically and can be scaled based on demand resulting in improving efficiency and cost effectiveness (Zhang et al., 2019). It enables the developers to create an entire application without provisioning and managing servers, thus the term serverless is derived (Chaudhary et al., 2020).

On the other hand, FaaS enhances this approach by allowing the developers to deploy individual functions also known as microservices on a pay-as-you-go basis, thus reducing operational cost and making the development increase its flexibility. Every function is treated as a single microservice and can be executed in response to an event or trigger, without the need for the developer to manage the underlying server infrastructure (Morabito et al., 2019).

For example, in an e-commerce application, users make an action to “add to cart” and “check out” products they want to purchase. In a serverless architecture, the entire application can be developed and deployed without the need to provision servers. With the use of FaaS, each user can make an action on a pay-per-use basis. This allows the service to scale up when needed without provisioning additional costs and resources for idle time.

In any technological landscape, challenges are inevitable no matter how sophisticated and advanced the technology is, therefore, serverless architecture and FaaS are not exempted from this. The most common challenges include vendor lock-in, security implications, and getting the proper tools to effectively monitor and debug components in a distributed environment (Graves & Nielsen, 2020). Having a serverless environment means having complex and careful considerations to attain optimized performance and to make managing dependencies flow smoothly.

Discussing thoroughly the benefits and challenges associated with adopting serverless architecture and FaaS for the development and deployment of web services is the main goal of this paper. This would help the organizations and developers to make a thorough decision before adopting this kind of cutting-edge technology. Knowing its benefits and challenges will help the organization have an insight into what lies ahead that allows it to have strategic planning on implementing serverless architecture and FaaS.

Methods

This research was conducted using a systematic literature review of the reports, cloud computing official website, and existing studies on serverless architecture and FaaS focusing on their benefits and challenges. Comprehensive research using academic databases like IEEE, Google Scholar, AWS official website, Microsoft Azure articles, and official documents that were published between 2019 and 2024 had been used for this review paper.

Ethical Considerations

The paper discusses the challenges and benefits of serverless architecture and FaaS, thereby adhering to ethical principles in maintaining credibility and integrity. Thus, no humans were involved in the study. This paper ensures proper citation of all the resources to avoid plagiarism in upholding academic honesty. Rights for intellectual property were properly respected, and findings were presented correctly and objectively without any bias or exaggeration. Environmental and potential impacts of serverless computing were properly considered as a guide that is responsible when adopting the technology. Lastly, contributors were appropriately acknowledged to honor guidance and contributions to this paper.

Results and Discussion

Overview of Serverless Architecture and FaaS

Serverless computing is a cutting-edge technology that is currently revolutionizing the way applications are managed, developed, and provisioned. With this technology, software can be developed, managed, and deployed separately and it is called functions, which comes with a minimal cost for the organization (Kritikos & Skrzypek, 2019). Applications that run in a serverless architecture are event-driven, which means that it is a loosely coupled service called microservices that communicate through events and let the independent team focus on a single business process without the need to understand the entire application. On the other hand, FaaS plays a role in offering a standard CPU and memory power to serverless applications to rapidly scale up and/or scale down the instances of the application without any additional configuration or cost that grants a performance edge to a process (Raza et al., 2021).

Understanding Serverless Architecture

In a serverless computing model, the cloud provider will dynamically manage the underlying infrastructure in terms of allocation and provisioning of servers that will allow the developers to solely focus on writing and deploying source code. The serverless computing model provides a way for developers to build and deploy applications by abstracting away the complexity of infrastructure management (Castro et al., 2019).

The main principles and characteristics of serverless architecture include event-driven models, scalability, pay-as-you-go pricing model, and stateless nature. Event-

driven models utilize predetermined events such as timer ticks, message queues, or HTTP requests to activate functions autonomously to handle tasks efficiently to execute distributed computations (Morenets & Shabinskiy, 2020). Scalability, on the other hand, is used to handle increases or decreases in the number of resources such as computing power, storage, or network bandwidth to accommodate requests and adapt to frequent changes in demands without sacrificing its performance and reliability to ensure consistent performance and responsiveness as demand fluctuates.

In addition, the pay-as-you-go pricing model is a billing approach where the customers are charged based on the actual usage of a specific service rather than a fixed rate. There is no need to have an upfront cost (Kumar, 2019). Lastly, stateless nature refers to the absence of a stored state or session between interactions. Each function can perform independently and does not rely on other functions or previous interactions.

Key Components of Serverless Architecture and FaaS

They key components of serverless architecture include functions or the units of code that perform specific tasks, events and triggers or actions or events that invoke functions, external services and APIs used by serverless applications, and data stores used to persist application state.

On the other hand, FaaS can be presented as small snippets of source code commonly written in but not limited to JavaScript, Python, C#, VB.Net, Ruby, etc. as functions that can be deployed in an API gateway that can be consumed to a well-defined user interface. FaaS can offer polyglot development that allows developers to develop applications that use multiple programming languages (Hao & Glassman, 2020). These functions represent the business logic that can be triggered via events, HTTP requests, timers, and/or when data is added to a storage service that allows an event-driven architecture (Scheuner & Leitner, 2020).

Moreover, there are some of the popular FaaS platforms across different cloud providers. This includes Amazon Web Service: AWS Lambda which supports multiple programming languages like but not limited to C#, Java, Python, and many more and offers tight integration with other AWS services. It also considers Microsoft: Azure Functions which offers seamless integration with Microsoft Azure tools and provides excellent support for Microsoft enterprise applications. Lastly, it includes Google: Google Cloud Functions which focused on simplicity and scalability, with strong support for event-driven programming using Google services.

Additionally, the most common use cases for serverless architecture and FaaS are web applications, IoT (Internet of Things), real-time data processing, and batch processing.

Below is a simple web application for a serverless architecture using the AWS platform.

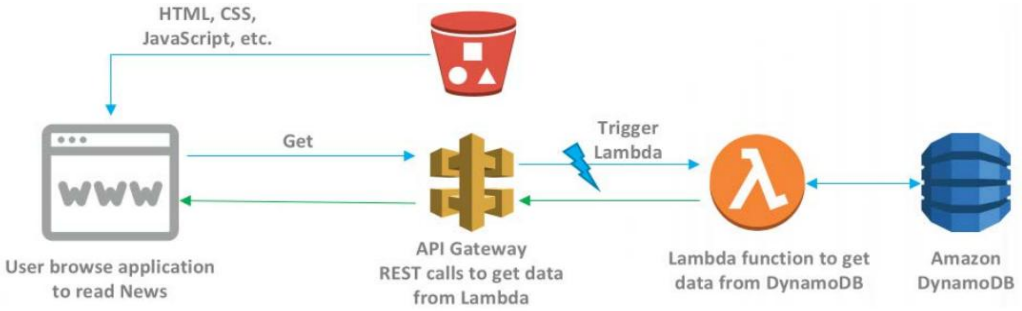

Figure 1. *AWS Serverless Architecture for Web Application (Kumar, 2019)*

Benefits of Serverless Architecture and FaaS

Polyglot programming is defined as the practice of using multiple programming languages within a single development context (Uesbeck, 2019). It allows multiple developers with diverse skills in different programming language backgrounds to collaborate in the development of a single application, which makes collaboration easier by leveraging each developer's expertise in their respective languages.

In terms of scalability, leveraging principles of an event-driven architecture allows to automatically scale resources up or down based on incoming demand (Enes et al., 2020). Event-driven scaling means once an event has been triggered, the serverless platform automatically provisions resources to execute the service or function. Resource allocation will only be invoked if an event has been triggered to minimize idle resource consumption. An example of it is using AWS Lambda, if the function is triggered in an event click that executes an AWS Lambda function, AWS will automatically scale the compute resource with that specific AWS Lambda function to respond to the incoming workload. Additionally, AWS Lambda functions can be triggered in parallel which allows multiple instances of the same function to handle traffic. AWS Lambda pricing is based on how many times and the duration of execution performed.

Serverless architecture and FaaS are cost-efficient since cloud providers enable the organization to minimize overall costs in relation to application development and deployment with the utilization of cloud computing platforms effectively (Maurya et al., 2021). This means that in going in the path of serverless architecture and FaaS, the organization has the option to go with a pay-as-you-go pricing model where billing is based on the resources consumed without upfront cost, or have various flexible pricing models including on-demand, reserved instances, and spot instances that allow the organization to choose pricing models that fit with what is needed with various discounts which leads to cost savings over time.

Furthermore, the use of serverless architecture and FaaS would pave the way for reduced operational overhead. Investing in serverless architecture and FaaS is pivotal for organizations that are aiming to streamline optimization and operational costs. This will include various services for reducing administrative tasks through already available managed services. From having a compliance certification to built-in security features, managed services ensure security within the cloud that does not include extensive manual efforts (Tatineni, 2021). It also eliminates server management since there is no need to have a physical server that involves a lot of effort but is not limited to procurement, installation, configuration, maintenance, and monitoring.

Likewise, serverless architecture and FaaS lead to faster time-to-market service. Going cloud is not just to reduce cost but will also significantly reduce time-to-market

delivering applications to users (Backes et al., 2019). Having a proper setup in configuration, monitoring and logging will lessen the investigation time spent in each function deployed. Additionally, cloud providers provide insight and fault tolerance capabilities that are able to contribute to having a faster time-to-market service.

In terms of availability and fault tolerance, with today's serverless architecture, having a high availability is already embedded in one application, having an application deployed redundantly in multiple availability zones within a single region is a genius way of enabling a service to ensure that applications will be accessible and functional once a failure kicks in or any other reasons for disruption (Hao & Glassman, 2020). This also assures that cloud providers will offer a built-in fault tolerance deeply connected to the service, which gives an application that is resilient to failures and can handle errors during operation. Separating business logic from a microservice means any function will work independently and will not affect other functions related to any other service; failures will be isolated which minimizes the risk of spreading failures.

Challenges of Serverless Architecture and FaaS

Cold start latency refers to having a delay in executing a function during the initialization of containers if during an invocation there is no existing container (Ebrahimi, 2024). It imposes a significant impact in terms of system performance since it will increase the execution time to invoke a function, thereby it will reduce efficiency. It is one of the major challenges in serverless computing since it has the potential effect of contradicting the benefits offered such as affordability and scalability.

On the other hand, vendor lock-in refers to a situation where an organization becomes dependent on a single or specific cloud service provider, which results to struggling to switch to an alternative cloud service provider due to constraints agreed upon with the existing cloud service provider (Kumar, 2022). Lock-ins may be a pricing lock-in where an organization did not adhere to practices to follow usage-based pricing resulting in paying higher than what is needed. Another lock-in may be a data lock-in where there is a difficulty in retrieving and duplicating data, hence it is held hostage when attempting to switch cloud service providers. There is also a so-called flexibility lock-in where choosing between internal and external databases faces limitations and it is not part of the service offered by the cloud service provider. Lastly, renewal lock-in occurs when it discourages organizations or customers from switching providers by delaying renewal discussions to have a price increase after the initial term.

Debugging and monitoring are also considered as challenges. Today's cutting-edge technologies, with the abstraction of operational tasks combined with various service-managed activities, result in a fine-grained serverless architecture. This architecture also establishes a tight connection with the managed services running in the backend. Though it was appealing, in some instances, it also creates complex layers of services and numerous data logs since there are events and actions happening on the backend (Manner, 2019). If there was no strategy for performing analysis on logs of data provided by cloud service providers, it would be a tedious activity to perform debugging and monitoring of functions.

Below is the matrix of how logging works in a serverless architecture and FaaS that can be done in the absence of robust data log strategies that can lead to several challenges (Kurbegovic, 2023).

Table 1. Impacts of Inadequate Data Log Analytics Strategies in Serverless Architectures Matrix

Challenges	Potential Cost	Impact
Fragmented Log Data	High in manual effort	Delays in the debugging process may lead to longer application downtime
Lack of Centralized Monitoring	Performance bottleneck	Degraded user experience
Insufficient Log Quality	Inaccurate fault analysis	Identifying the root cause can impose higher maintenance costs
No Automated Fault Detection	Increase time in troubleshooting	Higher operational costs
Unstructured Log Analytics	Difficult in managing data	Poor visibility in application health

In addition, in terms of serverless computing, security and compliance are common concerns due to the distributed nature of the cloud nature. Since multiple tenants are accessing a single server, this raises a concern in terms of access, data breach, and cyber threats. Cloud service providers encompass various aspects that include but are not limited to data encryption, access controls, network security, and compliance with regulatory industry standards and best practices (Brandis et al., 2019). In terms of compliance, it is a shared responsibility between cloud service providers and organizations with a wide range of regulatory and specific requirements to maintain trust with the parties involved.

Also, having a serverless architecture and FaaS approach can produce an issue related to scalability and performance having a complex nature of microservices. Managing the development and deployment of multiple services can be a bit challenging as the system grows. Increases in services included can result to complexity in managing dependencies and configuration that can potentially lead to scalability bottlenecks (Blinowski et al., 2022). Serverless architectures communicate services via HTTP-based APIs or messaging protocols, thus sometimes introducing latency compared to applications developed using a traditional monolithic approach. Each interaction between calling a service involves a network call, encrypting and decrypting of data, and potentially copies data between user and kernel space impacting overall performance. Thus, both need an efficient architectural design consideration based on industry best practices to eliminate such issues.

On the other side, resource management refers to a challenge with the shared resources provided to an organization within the cloud computing environment, this includes sometimes an over or under-provisioning of a service which can then lead to performance issues (Shukur, 2024).

Conclusion and Future Works

In conclusion, adopting serverless architectures and FaaS in terms of developing and deploying web services has both benefits and challenges that need to be considered thoroughly. Serverless architecture allows developers to solely concentrate on building business logic in creating microservices or so-called functions that can be written using various programming languages that promote collaboration amongst developers. The benefits aligned with serverless architecture and FaaS are apparent in terms of

scalability, cost efficiency, reducing operational overhead, accelerating time-to-market, and robust availability, enabling the organization to streamline the development process to further enhance user experience. However, alongside are the challenges like cold start latency, vendor lock-in, complexities in debugging and monitoring, security and compliance issues, and scalability/performance. Careful planning that is aligned with industry standards and best practices should be established to mitigate such challenges for the organization to fully experience the full potential of cloud computing.

Moreover, the absence of standardized guidelines in implementing FaaS can increase operational inefficiencies like debugging costs, extended downtimes, and unoptimized resource utilization. The lack of clear frameworks makes the organization face difficulties in implementing a streamlined deployment process and consistent performance matrix. At the end of the day, organizations must weigh and assess the benefits and challenges of adopting cloud computing as provided in this paper for valuable insights and guidance before embarking on the journey to serverless architecture and FaaS.

Considering the information provided in this paper, various practical implications and recommendations can be made. Adopting a serverless architecture should start with having a clearer picture and understanding of the existing infrastructure. Migrating to serverless can be approached in several phases accordingly to have a lesser operational disruption. Furthermore, across all cloud computing platforms, security should be the priority. Leveraging services for encryption and user management can mitigate risks in serverless environments. Also, having a pay-as-you-go pricing is highly advantageous in reducing upfront costs. However, organizations still need to strategically monitor usage to avoid unexpected costs with unpredicted workloads. Additionally, organizations may invest in training a team to understand serverless architecture, microservices, and cloud security best practices.

Future researchers could also focus on developing a framework for serverless deployment to address the challenges in resource management, performance optimization, and debugging. Research on improving cold latency in FaaS will be highly useful in improving performance in a serverless environment.

References

[1] Almansouri, M., Verkerk, R., Fogliano, V., & Luning, P. (2021). Exploration of heritage food concept. *Trends in Food Science & Technology, 111*, 790–797. https://doi.org/10.1016/J.TIFS.2021.01.013

[2] Alonso, E., Cockx, L., & Swinnen, J. (2018). Culture and food security. *Global Food Security*. https://doi.org/10.1016/J.GFS.2018.02.002

[3] Backes, J., Bayless, S., Cook, B., Dodge, C., Gacek, A., Hu, A. J., Kahsai, T., Kocik, B., Kotelnikov, E., Kukovec, J., McLaughlin, S., Reed, J., Rungta, N., Sizemore, J., Stalzer, M., Srinivasan, P., Subotić, P., Varming, C., & Whaley, B. (2019). Reachability analysis for AWS-based networks. In I. Dillig & S. Tasiran (Eds.), *Computer Aided Verification: 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019. Proceedings, Part II* (Lecture Notes in

Computer Science, Vol. 11562, pp. 231–241). Springer. https://doi.org/10.1007/978-3-030-25543-5_14

[4] Blinowski, G., Ojdowska, A., & Przybylek, A. (2022). Monolithic vs. microservice architecture: A performance and scalability evaluation. *IEEE Access, 10*, 20359–20361. https://doi.org/10.1109/ACCESS.2022.3152803

[5] Brandis, K., Dzombeta, S., Colomo-Palacios, R., & Stantchev, V. (2019). Governance, risk, and compliance in cloud scenarios. *Applied Sciences (Switzerland), 9*(2). https://doi.org/10.3390/app9020320

[6] Castro, P., Ishakian, V., Muthusamy, V., & Slominski, A. (2019). The rise of serverless computing. *Communications of the ACM, 62*(12), 44–54. https://doi.org/10.1145/3368454

[7] Caudwell, J., Choe, J., Dickinson, J., Lavrushkina, N., & Littlejohns, R. (2020). 'Multicultural lunches': Sharing food in post-Brexit south coast of England. *Annals of Leisure Research, 23*(4), 544–561. https://doi.org/10.1080/11745398.2019.1568892

[8] Chaudhary, S., et al. (2020). Exploring serverless computing: Characteristics and implementation. *IEEE Cloud Computing, 7*(6), 10–18.

[9] Dovetail Editorial Team. (2023, February 7). What is phenomenology in qualitative research? *Dovetail*. https://dovetail.com/research/phenomenology-qualitative-research/

[10] Ebrahimi, A., Ghobaei-Arani, M., & Saboohi, H. (2024). Cold start latency mitigation mechanisms in serverless computing: Taxonomy, review, and future directions. *Journal of Systems Architecture*. https://doi.org/10.1016/j.sysarc.2024.103115

[11] Edwards, F., & Davies, A. (2018). Connective consumptions: Mapping Melbourne's food sharing ecosystem. *Urban Policy and Research, 36*(4), 476–495. https://doi.org/10.1080/08111146.2018.1476231

[12] Eismann, S., et al. (2022). The state of serverless applications: Collection, characterization, and community consensus. *IEEE Transactions on Software Engineering, 48*(10), 4152–4166. https://doi.org/10.1109/TSE.2021.3113940

[13] Enes, J., Exposito, R. R., & Touriño, J. (2020). Real-time resource scaling platform for big data workloads on serverless environments. *Future Generation Computer Systems, 105*, 361–379. https://doi.org/10.1016/j.future.2019.11.037

[14] Graves, T., & Nielsen, K. (2020). Challenges and best practices in serverless architecture. In *Proceedings of the 2020 International Conference on Cloud Engineering*, 230–239.

[15] Hao, R. L., & Glassman, E. L. (2020). Approaching polyglot programming: What can we learn from bilingualism studies? In *OpenAccess Series in Informatics* (Schloss Dagstuhl–Leibniz-Zentrum für Informatik GmbH). https://doi.org/10.4230/OASIcs.PLATEAU.2019.1

[16] Holder, M. (2019). The contribution of food consumption to well-being. *Annals of Nutrition and Metabolism, 74*, 44–52. https://doi.org/10.1159/000499147

[17] Kandil, S. (2022). The role of food culture in developing the nutritional awareness and healthy behaviour of students. *International Journal of Humanities and Language Research*. https://doi.org/10.21608/ijhlr.2023.215929.1012

[18] Kapelari, S., Alexopoulos, G., Moussouri, T., Sagmeister, K., & Stampfer, F. (2020). Food heritage makes a difference: The importance of cultural knowledge for improving education for sustainable food choices. *Sustainability, 12*, 1509. https://doi.org/10.3390/su12041509

[19] Kritikos, K., & Skrzypek, P. (2019). A review of serverless frameworks. *2019 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion),* 161–168

[20] Kumar, M. (2019). Serverless architectures review: Future trends and the solutions to open problems. *American Journal of Software Engineering, 6*(1), 1–10. https://doi.org/10.12691/ajse-6-1-1

[21] Kumar, P. (2022). Vendor lock-in situation and threats in cloud computing. *International Journal of Innovative Science Research and Technology, 7*(9).

[22] Liaqat, A., Axtell, B., & Munteanu, C. (2022). "With a hint she will remember": Collaborative storytelling and culture sharing between immigrant grandparents and grandchildren via magic thing designs. *Proceedings of the ACM on Human-Computer Interaction, 6*, 1–37. https://doi.org/10.1145/3555158

[23] Liu, R., & Grunert, K. (2020). Satisfaction with food-related life and beliefs about food health, safety, freshness, and taste among the elderly in China: A segmentation analysis. *Food Quality and Preference, 79*, 103775. https://doi.org/10.1016/J.FOODQUAL.2019.103775

[24] Manner, S., Kolb, S., & Wirtz, G. (2019). Troubleshooting serverless functions: A combined monitoring and debugging approach. *SICS Software-Intensive Cyber-Physical Systems, 34*(2). https://doi.org/10.1007/s00450-019-00398-6

[25] Morabito, F., et al. (2019). Function-as-a-service: An in-depth overview. *ACM Computing Surveys, 52*(5), 1–34.

[26] Morenets, I., & Shabinskiy, A. (2020). Serverless event-driven applications development tools and techniques. *Scientific Notes of NaUKMA. Computer Science, 3*, 36–41. https://doi.org/10.18523/2617-3808.2020.3.36-41

Acknowledgment

The authors would like to express gratitude to Isabela State University and the reviewers of this paper. Their valuable insights and constructive feedback have significantly contributed to enhancing the clarity and quality of this study.

Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.