Asian Journal of Business and Technology
Vol. 3, No. 1, (2020)
ISSN 2651-6713 (Print)
ISSN 2651-6721 (Online)

A Literature Review on the challenges of using Serverless Computing in Web Services

Jayson E. Tamayo
Pagasinan State University
jayson.tamayo2@gmail.com

Abstract – *This paper presents a review of literature on the challenges of using Serverless Computing in Web Services. The research paper reflects on the understanding of the different issues and challenges in the earlier years. The literature review demonstrates the solutions to the challenges of using Serverless Computing in Web Services, examines current research on the topic, and points out gaps in the existing literature. The study's findings revealed that using Serverless Computing in Web Services encountered challenges in the past years. The researcher presented the challenges in using Serverless Computing in Web Services, namely cold start, security, programming and debugging, and other subcategories. Still, there are some purpose solutions to resolve most of the challenges of Serverless Computing. Those solutions are data processing and SaaS integration, serverless application use cases, web application architecture, and mobile back-end, and, lastly, scheduled tasks, automated backups, and log analysis. However, the challenges of using Serverless Computing in Web Services brought about potential opportunities for serverless computing in web services. These opportunities provide insight for further studies and keep future researchers using Serverless Computing in Web Services.*

Keywords – Serverless Computing, Web Services, function-as-a-service (FaaS), Web applications, and Cold Starts

1. INTRODUCTION

In these long stretches of current innovation advancement, many ways can be utilized to make web applications. Choosing one over the other ought to be exclusively founded on specialized contentions and their abilities conveyed by every other option. Thus, Web Services have gained enormous popularity in how devices communicate with each other. Furthermore, current advancements on the Web are described by the more extensive utilization of organization-empowered gadgets, like sensors, cell phones, and wearables that fill in as information suppliers or actuators, regarding customer Web applications[3]. Briefly, web services, according to Mironela, are self-describing and modular business applications exposing business logic as a service over the Internet[1]. They are delivered through programmable interfaces, while their functionality can be consumed and invoked through their IP address, as Wagh et al. [2] stated.

The Web has been continuously extending its reach to empower more advanced shapes of interaction between browser clients and servers: "single form-based interactions, retail e-commerce applications, and more complex business-to-business interactions"[4]. Traditionally, before the advent of serverless computing, websites were hosted on single servers. The controlling entity would build and maintain servers, increasing cost and programmer workload. There would be a user-facing client on these servers, a back-end to handle business logic, and a database to house information. This kind of architecture had several issues, primarily lack scalability and cost. According to Daniel et al., a company would build or rent a server that could handle a predetermined amount of traffic. If the traffic to the website exceeded the limit, the website would slow down or even crash[6].

Furthermore, if the website traffic were too low, the owning company would be wasting money on server fees[5]. This all changed with the appearance of serverless computing.

The research conducted by Won Kim [15] suggests that a traditional cloud architecture consists of a 3-tier client-server architecture. This solution allows for high availability by utilizing data allocation across multiple availability zones, where information can exist in more than one location. However, this basic approach to cloud computing fails to solve extensive server costs, vastly overbearing infrastructures, and considerable server maintenance. Other research [16] suggests that cloud computing is vulnerable to security breaches and access from unauthenticated users. It holds for cloud computing in the traditional sense, where database integrity and access are incredibly susceptible to harmful tampering when hosted within a server-based cloud architecture. This susceptibility is a result of entities being interconnected through the third party's standards of initialization. The original promise of cloud computing was the ability to have elastic resources that scaled up and down as needed, and users only paid for what they used. It's a promise that serverless computing delivers upon.

Serverless is an emerging computing worldview within the cloud environment. The Serverless model is also known as function-as-a-service (FaaS), where clients are free to create, run, and oversee the functionalities of their applications without the complexity of building and keeping up the foundation ordinarily when creating and propelling an app [7]. Serverless computing requires breaking the workload into short-running and stateless capacities to attain ease-of-maintenance and auto-scaling[8]. Its pay-as-you-go show as it were charges when capacities are running, which is additionally invited by cloud clients. With those serverless stages, designers for web administrations as it were got to transfer their capacities for adaptable and disseminated execution. There is moreover open-sourced usage accessible for serverless computing like OpenLambda and OpenWhisk. Besides, numerous cloud merchants have discharged their serverless platforms, such as Amazon Lambda, Google Cloud Functions, IBM Cloud Functions, and Microsoft Azure Functions.

Notwithstanding the excellencies, serverless computing also has constraints and issues. "First, the short-running nature of serverless functions limits the scope of application[8]". "Second, breaking applications into functions introduces performance overhead[9][10]". "Third, in terms of security properties, traditional execution abstractions like VM and container are not optimal for holding these small functions from different customers"[8][9][10]. Earlier work has talked about those issues and given different arrangements, but numerous stay uncertain and open. Furthermore, stated by Hellerstein, one of the most concerns in these administrations that have genuine downsides on the in general execution of serverless applications is the taken a toll of protecting the state[11]. It is also affirmed by Adzic that "shared memory space on a single machine is not always available in serverless applications due to its auto-scaling nature that does not guarantee the execution of related functions on a single machine" [12].

This paper aims to carry out a literature review of the challenges of using Serverless Computing in Web Services. This review will focus on answering the following questions:

1. What are the different challenges of using serverless computing in web services?
2. What are the solutions presented to address those challenges? and,
3. What are the potential opportunities of serverless computing in web services?

This paper will recognize the challenges of using Serverless Computing in Web Services by reviewing the existing literature on the topic.

2. METHODOLOGY

2.1 Literature Search

In this literature review, the results of functional studies were published in electronic databases such as Google Scholar, Research

Gate, and EBSCOhost. A variety and combination of keywords were used in the review, including *Challenges of Serverless Computing, Serverless Computing, Web Services application, Web Services, function-as-a-service (FaaS) issues, and solutions Serverless Computing*. These search terms were used for all fields, including title, abstract, keywords, and full text.

In perusing for research articles on the Web, the following procedure, as created by Manu Bhatia [13] and expressed in the Humans of Data site, was applied:

In addition, after conducting preliminary searches in electronic databases, a set of criteria is established to select appropriate studies for inclusion in the study. The paper selection process included 1) literature study evidence regarding challenges of using Serverless Computing in Web Services, 2) published as articles or journals, 3) have an abstract, and 4) the research method is demonstrated clearly and 5) the published article must at the year publish range from 2017 to 2021. A citation was omitted if 1) it was in academic settings such as book review, 2) it was a book or chapter, or 3) no theoretical data was reported.

The authors, the year of publication, the study's goals, the settings, the procedure used (research design, data collection, and analysis techniques), and the key findings were all methodically collected. However, some articles published earlier than 2017 were utilized to establish research development throughout the years and establish a need for further research. The research looked for the most common responses to questions, identifying data or patterns to answer research questions and finding areas that can be explored further. Furthermore, the references section of each research article has been looked upon for any relevant researches that could help in a more comprehensive review. This method addresses all the questions and identifies solutions by this process of literature search.

2.2 Identifying the different challenges of using serverless computing in web services.

Based on journal papers, global surveys, current industry happenings, and showcase trends, the researcher gathered data from various writing sources. After collecting the literature, the researcher sorts it to see if it's essential to finalize the subject's representative literature. The importance of the literature is determined by its meaning, authority, effectiveness, and dependability.

There are two parts in organizing the process of determining the different challenges of using serverless computing in web services. The articles are analyzed in the first part using the existence and prohibition criteria. The journals without practical evidence related to the challenges in using serverless computing in web services are eliminated in the review. The remaining journals with potential are obtained and will be examined further in the following section. The reviewed journals' challenges are listed in the second part, and the overview of each issue is summarized.

2.3 The solutions presented challenges of using serverless computing in web services

Further investigation of the listed challenges of using serverless computing in web services was made. The researcher further analyzed the solutions to every issue and challenge in using serverless computing in web services. Several solutions have been identified for the challenges; however, these solutions did not resolve all the problems using serverless computing in web services. In this phase, all of the identified challenges and their corresponding solutions are separated into two parts; 1) challenges that already have a solution, and 2) remained challenges that await solution. The additional searches through references are also conducted in this phase, where the researcher further investigated the references of the initially found papers and the references made to those papers. The supplementary investigation is used to validate if using serverless computing in web services still has no solution to the identified remaining challenges in recent years.

2.4 Identifying the potential opportunities of serverless computing in web services

Qualified peer-reviewed research papers related to the potential opportunities of serverless computing in web services were left for review. To identify the potential opportunities for serverless computing in web services, a context of potential opportunities was derived from the challenges encountered in using serverless computing in web services. To explain further the possible opportunities, recommendations from the studies are also taken into consideration. A modified Webster and Watson matrix[14] will also be utilized to collect and analyze the different opportunities systematically. In this part, the articles were examined to determine other challenges affecting the use of serverless computing in web services concerning the issues that remained.

3. RESULTS AND DISCUSSION

3.1 Literature search

The initial process of browsing for articles in various research databases, presented in Table 1.0, is the average number of results that appeared in the different electronic databases of research, exclusive of the article citations, using the various search keywords presented in Section 2.1. Introduced in Column 2 is the absolute number of results signifying the popularity of the research subject throughout January 2017 up to July 2021, which contains all results, including non-scientific writings such as magazines, trade publications, and patents.

Electronic Database	Total Number of Result	Peer-Reviewed
Google Scholar	6,040 (approx.)	23
Research Gate	1,892 (approx.)	12
EBSCOhost	237	2

Table 1.0 Search result from electronic databases

A total of 75 relevant types of research and journal articles have been collected as potential references. However, to ensure the paper's quality, these articles have been filtered according to their publishing date in which articles were published from the year 2017 to 2021. Based on the reading of abstracts, 25 journals are excluded from the review, and the remaining 55 journals with potential are acquired. The remaining full texts journal articles are evaluated to check if it has theoretical evidence. Finally, 18 articles are excluded, and 37 qualified peer-reviewed research papers on the challenges of using Serverless Computing in Web Services were left for review. Table 2.0 presents the tally of the references used in this review and their electronic databases of origin.

Electronic Database	References
Google Scholar	[17],[18],[26],[28], [36],[34],[24],[31], [32],[35],[37],[38],[39],[25], [40],[25],[47],[53],[54], [55],[50], [51],[48]
Research Gate	[27],[23],[22],[29], [30],[41],[42],[43], [44],[45],[46],[49]
EBSCOhost	[55],[52]

Table 2.0 Tally of Utilized Research Articles and Journals

3.2 Challenges of using serverless computing in web services.

This section presents challenges toward serverless computing. Instead of referring to various papers individually, the researcher presents the papers investigating the challenges of using serverless computing on table 3.0.

Challenges	Sub Categories of the Challenges	Paper	Total
Cold start	Resource limits	[18],[24],[25], [26],[27],[28]	6
Programming & debugging	Vendor lock-in	[17],[23],[22], [29],[30],[31]	6

Security	Scalability and Long-running	[17],[18],[26],[28],[36],[34],[24],[31]	9

Table 3.0 Challenges of using serverless computing in web services

As the subject of serverless computing is moderately modern, there are a number of challenges that got to be tended to. Some of these challenges are not unique to the serverless computing model and can arise in any computing environment, such as security and privacy [18 - 21]. Some types of challenges are specifically unique to serverless services, such as Cold start, programming, and debugging[18][19][22][23].

3.2.1 Cold start

Serverless computing can scale to zero while there is no request for functions and services. Scaling to zero leads to a problem called cold start. A cold start occurs when serverless functions remain idle for some time, and the next time these functions are invoked, a longer start time is required. Methods and techniques to reduce the cold start problem are crucial, and many papers have been studied this problem, as shown in Table 3.0.

3.2.1.1 Resource limits

In serverless computing, resources are required to ensure that the platform can deal with load increases. It includes CPU usage, memory, execution time, and bandwidth [24, 25, 26, 27, 28].

3.2.2 Programming & debugging

As the topic of serverless computing is relatively new, its development tools are not rich enough. It poses a significant challenge for software developers. Also, the lack of proper modeling paradigms leads to non-unified development approaches, which will reduce the code quality and complicate collaborations of developers in the long term.

According to Hassan et al., Serverless computing is currently lacking of debugging tools[17]. Further, monitoring tools are required since developers need to monitor the application and observe how functions are working. More advanced integrated development environments (IDEs) are required, so developers can perform refactoring functions, such as merging or splitting functions and reverting functions to the previous version. Moreover, logs from serverless function invocations need to be sent to the developer and provide detailed stack traces. When an error occurs, a good method is required to report details on the occurrence to the developer. The equivalent of a stack trace for serverless computing is currently not available.

3.2.2.1 Vendor lock-in

The FaaS paradigm separates the code from the data, which leads the functions to depend strongly on the could provider's ecosystem for storing, obtaining, and transferring data [29]. This issue makes the customers dependent on the serverless provider for products and services, and the customers cannot easily use different vendors in the future without substantial cost. Thus, customers have to wait on the serverless provider for additional services[30, 31, 25].

3.2.3 Security

Security is an indispensable concern in any computation service, be it serverless or not. Various security challenges are common between serverless and other cloud services. Those challenges are not the concern of this paper. Instead, it focuses on the security issues that specifically threaten the normal operation of serverless services. Security is the most challenging issue in serverless cloud computing[17]. One of the security issues is isolation because many users run on a shared platform. Therefore, strong isolation is required. Another security issue is trust when it comes to process-sensitive data. The serverless applications work with many system components, which must function correctly to maintain security properties.

3.2.3.1 Scalability

Serverless computing must ensure function scalability and elasticity. For example, when many requests are issued to a serverless application, these requests should all be served, and the used serverless cloud provider should provide the required resources to process all these requests and should scale up with the number of requests [26, 28, 36].

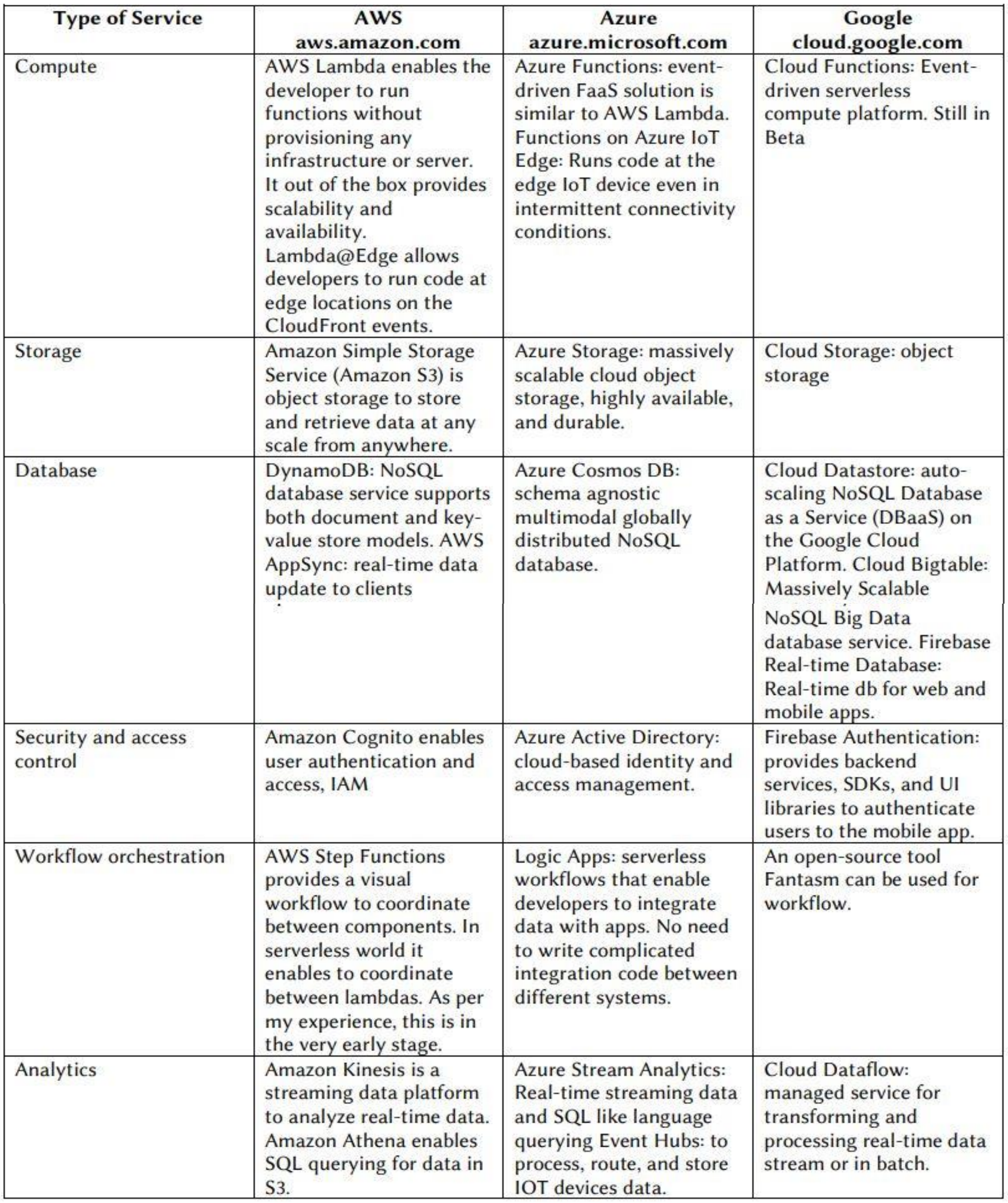
3.2.3.2 Long-running

Serverless computing runs function in a limited and short execution time, while some tasks might require a long execution time. It does not support long execution running since these functions are stateless, which means that if the function is paused, it cannot be resumed again[31, 24, 35, 28].

3.3 Solutions presented on the challenges of using serverless computing

Serverless technology provides an abstraction to servers, infrastructures, and operating systems. In recent years almost, all the major cloud provides extended their serverless cloud offerings. Following Table 4.0 is listing down all the serverless solutions from top providers.

Asian Journal of Business and Technology
Vol. 3, No. 1, (2020)
ISSN 2651-6713 (Print)
ISSN 2651-6721 (Online)



Type of Service	AWS aws.amazon.com	Azure azure.microsoft.com	Google cloud.google.com
Compute	AWS Lambda enables the developer to run functions without provisioning any infrastructure or server. It out of the box provides scalability and availability. Lambda@Edge allows developers to run code at edge locations on the CloudFront events.	Azure Functions: event-driven FaaS solution is similar to AWS Lambda. Functions on Azure IoT Edge: Runs code at the edge IoT device even in intermittent connectivity conditions.	Cloud Functions: Event-driven serverless compute platform. Still in Beta
Storage	Amazon Simple Storage Service (Amazon S3) is object storage to store and retrieve data at any scale from anywhere.	Azure Storage: massively scalable cloud object storage, highly available, and durable.	Cloud Storage: object storage
Database	DynamoDB: NoSQL database service supports both document and key-value store models. AWS AppSync: real-time data update to clients	Azure Cosmos DB: schema agnostic multimodal globally distributed NoSQL database.	Cloud Datastore: auto-scaling NoSQL Database as a Service (DBaaS) on the Google Cloud Platform. Cloud Bigtable: Massively Scalable NoSQL Big Data database service. Firebase Real-time Database: Real-time db for web and mobile apps.
Security and access control	Amazon Cognito enables user authentication and access, IAM	Azure Active Directory: cloud-based identity and access management.	Firebase Authentication: provides backend services, SDKs, and UI libraries to authenticate users to the mobile app.
Workflow orchestration	AWS Step Functions provides a visual workflow to coordinate between components. In serverless world it enables to coordinate between lambdas. As per my experience, this is in the very early stage.	Logic Apps: serverless workflows that enable developers to integrate data with apps. No need to write complicated integration code between different systems.	An open-source tool Fantasm can be used for workflow.
Analytics	Amazon Kinesis is a streaming data platform to analyze real-time data. Amazon Athena enables SQL querying for data in S3.	Azure Stream Analytics: Real-time streaming data and SQL like language querying Event Hubs: to process, route, and store IOT devices data.	Cloud Dataflow: managed service for transforming and processing real-time data stream or in batch.

Table 4.0 Comparing Leading Serverless solutions. AWS [53], Azure [54], Google Cloud [55]

Amazon Web Services (AWS) [53] is the market leader in the cloud space with the most advanced set of serverless products. These fully managed services enable developers to build and run serverless applications. Recently AWS launched Serverless Application Repository, which provides a starting point for serverless projects. This repository has several publicly available serverless applications which can be searched, deployed, and published in few clicks. Developers can search multiple serverless applications, deploy and modify them as per their requirements and even integrate them.

Microsoft Azure [54] holds the second position in the cloud market share. They provide a good set of toolsets in serverless space. Azure can be the best choice if a respective organization already uses a significant number of Microsoft software.

Google [55] is a third major player in the cloud market share. It also has a wide variety of serverless offerings. Over the years, Google Cloud Platform (GCP) has launched several serverless products covering application development and analytics.

Comparing the efforts in the industry from features of the solutions and the application. Industrial solutions cover the three main challenges mention in section 3.2; nonetheless, the solutions show different characteristics. Thus, providers prefer methods that have been proven in long-term practice and usually combine several methods to achieve multilayer protection[17]. In addition, by Hassan et al. research-led solutions often adopt a more holistic approach, from developer to platform[17]. According to Bila et al., without control over other parties, providers can promote their solutions only through guidelines or best practices[48].

3.3.1 Data processing and SaaS integration

Functions support triggers based on activity in a Software as a service (SaaS)-based application. As stated by Eivy, it demonstrates an example to save a file in OneDrive, which invokes an Azure function[47]. The function can modify excel and create analytical charts using Microsoft Graph API, which helps ease resource limits[54].

The real-time data processing system can be build using AWS Lambda[53], Amazon Kinesis, S3, and DynamoDB. According to Mohamed, cold start can be fixed by suggesting sample serverless architecture for Amazon Kinesis, S3, and DynamoDB[50].

3.3.2 Serverless Application Use cases, Web Application Architecture, and Mobile Backend

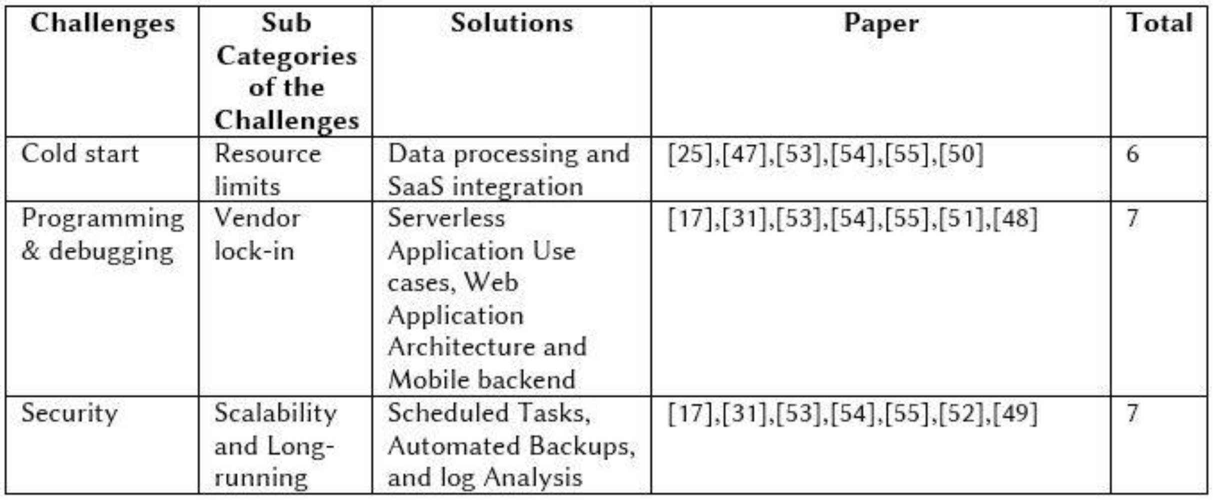
Azure Functions are FaaS offering from Azure and can be exposed using WebHook URL to work as microservices[54]. According to Lynn et al., this is good architecture to perform CRUD operations of a single web service[48]. Azure Functions support HTTP triggers and output bindings. These HTTP triggers can be customized to respond to WebHooks and work like APIs. These APIs work as a back-end to web services apps.

As an example stated by APEX Software development Web applications and Web services, we can use serverless News application that is one of the features of AWS[51]. This can host web services in S3, and Lambda can be used to perform data processing units. It will get code from DynamoDB and expose it through the API gateway[53].

3.3.3 Scheduled Tasks, Automated Backups, and log Analysis

Lambda is great for scheduling tasks, log, and build schedules. According to Patrizio, it shows the log collection process and logs analytics process using kinesis and Elasticsearch. As a complete serverless web application, a developer can envision something like, as stated by Bila in this application website code is hosted at S3. When a browser makes a call for a webpage, the Web page code checks for authentication token if a new login call redirects to Amazon Cognito for authentication [49][53]. Once the web page gets token, it makes a call to Lambda using API Gateway. Lambda gets data from DynamoDB and returns it to the browser through API Gateway[53].

the function call behavior and then dynamically select the optimal ones. This technique could reduce the cold start by 90%. According to Xu et al. in the article "Adaptive function launching acceleration in serverless computing platforms," they proposed a strategy to predict functions



Challenges	Sub Categories of the Challenges	Solutions	Paper	Total
Cold start	Resource limits	Data processing and SaaS integration	[25],[47],[53],[54],[55],[50]	6
Programming & debugging	Vendor lock-in	Serverless Application Use cases, Web Application Architecture and Mobile backend	[17],[31],[53],[54],[55],[51],[48]	7
Security	Scalability and Long-running	Scheduled Tasks, Automated Backups, and log Analysis	[17],[31],[53],[54],[55],[52],[49]	7

Table 5.0 Solutions presented on the challenges of using serverless computing.

3.4 Potential opportunities of serverless computing in web services

As the advancement of serverless processing is generally new, there are a few potential opportunities available to be centered around as follows:

3.4.1 Function startup

One of the major potential opportunities is overcoming the cold start problem without affecting the primary feature of serverless, which is scaling to zero [32][35]. This could be performed via enhancing scheduling policies and developing more accurate function performance measurements containers. The researchers are still working to avoid cold start by reducing high delayed function startups via optimizing compute resources [31].

According to Horovitz et al., the authors in [25] proposed FaaStest, an autonomous approach based on machine learning to capture invoking time and warming the function using a fine-grained regression method [40]. However, overcoming the issue of function startups is still

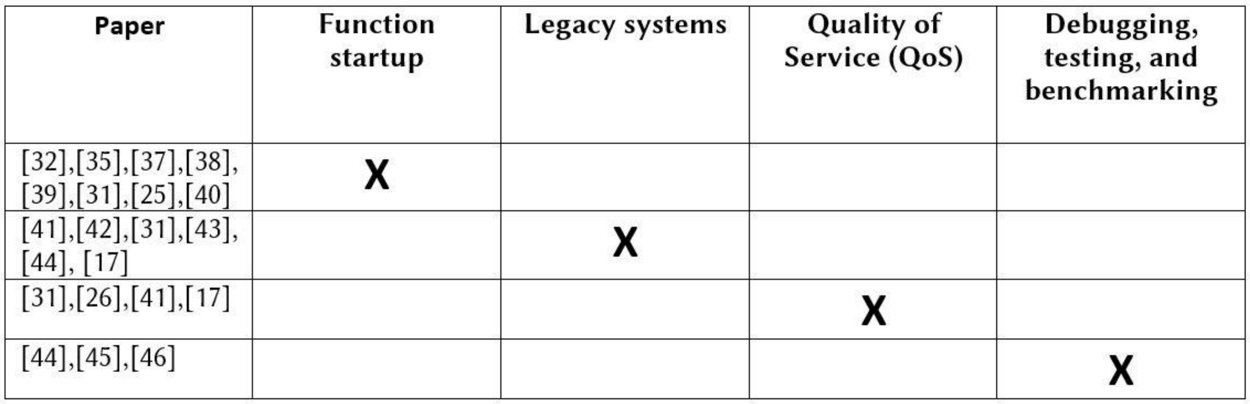
considered a potential opportunity direction to investigate.

3.4.2 Legacy systems

Since the serverless emergence, researchers are working on the open question of how to decompose legacy systems into FaaS without degrading performance; this is according to McGrath and Brenner in the article" Serverless computing: Design, implementation, and performance." [41]. Several works have been done on migrating to FaaS [42][31][43]. The currently available automated tools for migrating legacy code into FaaS are not fully practical due to the remaining manual work that needs to be done. According to the article "A mixed-method empirical study of function-as-a-service software development in industrial practice" [44].

Therefore, finding optimal automatic migration solutions for existing legacy systems is a possible opportunity [31]. Moreover, research on tools for checking whether a legacy system will fit the serverless paradigm is a crucial line[17]. According to Hassan et al. developing and enhancing automatic and semi-automatic analysis strategies based on artificial intelligence could be another future research field [17].

Table 6.0 Potential opportunities of serverless computing in web services.



Paper	Function startup	Legacy systems	Quality of Service (QoS)	Debugging, testing, and benchmarking
[32],[35],[37],[38], [39],[31],[25],[40]	X			
[41],[42],[31],[43], [44], [17]		X		
[31],[26],[41],[17]			X	
[44],[45],[46]				X

3.4.3 Quality of Service (QoS)

According to Yussupov et al., keeping a guaranteed QoS level in the software level agreement (SLA) that describes the lower service level offered by the service providers is a major obstacle for cloud providers to offer optimal performance metrics[31]. However, also added by Yussupov, serverless frameworks should consider the objectives of both providers and users[31]; customers and developers have none or little QoS support over the functions, according to Rajan[26]. In addition, the affirmed by McGrath and Brenner auto-scaling feature lacks QoS guarantees. This lack of QoS affects the performance of serverless applications. Increasing response time leads to decreasing the QoS level[41].

Furthermore, Hassan et al. stated that it also raises the cost of the service [17]. Therefore, achieving an ideal resource allocation management is complicated and challenging as several objectives should be fulfilled together [17]. Hence, providing more efficient QoS management of functions by the auto-scaling is essential without degrading the fault-tolerance features and increasing the cost.

3.4.4 Debugging, testing, and benchmarking

The available tools for testing, debugging, and deployment are immature; this prevents some developers from entering the FaaS is a core problem, particularly the testing tools this is according to Leitner et al. [44]. Moreover, most FaaS environments lack powerful local emulation platforms for testing. Therefore, developers mostly depend on the server side, which is expensive. Developers need to be ensured of adequate testing tools before diving into the serverless world. A challenging aspect in benchmarking is the lack of information due to the heterogeneity of the cloud provider data center: hardware, software, and configurations [45]. Additionally, benchmarking FaaS platforms should take advantage of analyzing the cloud services, which lack limited accessible measurements and hidden modification of services over time [46]. Thus, it is essential to have transparent, fair, and standardized benchmarking tools available for developers; this is for web services development.

4. CONCLUSION

The contributions of the work presented in this research paper about Literature Review on the challenges of using Serverless Computing in Web Services are threefold answers the research questions (1) What are the different challenges of using serverless computing in web services? (2) What are the solutions presented to address those challenges? And; (3) What are the potential opportunities of serverless computing in web services. Serverless computing challenges in web services are Cold start, Programming & debugging, and security issues. In contrast, the solution to these challenges based on the peer-reviewed articles is Data processing and SaaS integration, Serverless Application Use cases, Web Application Architecture, and Mobile Backend, Scheduled Tasks, Automated Backups, and log analysis. The potential opportunities of serverless computing in web services stated by different articles are Function startup, Legacy systems, Debugging, testing, and benchmarking. Given the fast evolution and growing interest in the field, this research paper focused on gathering the most notable trends and outcomes of using serverless computing, as described by recent researchers and industries.

This research paper could significantly reduce ambiguity and the entry barrier for novice developers to adapt to the serverless environment in developing web services. Furthermore, the findings presented in this research paper could be of great value for future researchers interested in further investigating serverless computing. Finally, it is worth mentioning that the interest that both commercial and academic efforts fueled into studying, developing, and implementing serverless tools in forthcoming years could help maximize the potential that serverless computing could bring to the IT community.

5. REFERENCES

[1] Mironela, P. (2018). The Importance of Web Services using the RPC and REST Architecture, IEEE, International Conference on Computer Technology and Development, 2018.

[2] Wagh, K. (2018). A Comparative study of SOAP Vs REST Web Services Provisioning Techniques for Mobile Host, Journal of Information Engineering and Applications, ISSN 2225- 0506 (Online), 2(5), 2018.

[3] D. Guinard, V. Trifa, F. Mattern and E. Wilde, E. From the Internet of Things to the Web of Things: Resource-oriented Architecture and Best Practices. In Architecting the Internet of Things. Springer Berlin Heidelberg, 2017.
[4] Web Services: Why and How Francisco Curbera, William A. Nagy and Sanjiva Weerawarana IBM T.J. Watson Research Center August 9, 2016

[5] AWS White Pages (2017). Optimizing Enterprise Economics with Serverless Architectures. Retrieved from https://d0.awsstatic.com/whitepapers/optimizing -enterprise economics-serverless-architectures.pdf

[6] Lisingo: A Text-To-Speech Web Service-Based Application Using Serverless Cloud Computing Andrew Daniel, Bransyn Luther and Imran Ghani 2018 International Conference on Computational Science and Computational Intelligence (CSCI)

[7] Serverless architecture. https://martinfowler.com/articles/serverless. html#unpacking-faas, 2018.

[8] Mingyu Wu, Zeyu Mi, and Yubin Xia', A Survey on Serverless Computing and its Implications for JointCloud Computing'2020 IEEE International Conference on Joint Cloud Computing (JCC)

[9] H. Shafiei, Member, IEEE, A. Khonsari, and P. Mousavi "Serverless Computing: A Survey of Opportunities, Challenges, and Applications," 2018

[10] Phani Kishore Gadepalli, Gregor Peach, Ludmila Cherkasova, Rob Aitken, Gabriel Parmer, "Challenges and Opportunities for Efficient Serverless Computing at the Edge" Arm Research, San Jose, CA 95134, USA The George Washington University, Washington, DC, USA 2019

[11] J. M. Hellerstein, J. Faleiro, J. E. Gonzalez, J. Schleier-Smith,V. Sreekanti, A. Tumanov, and C. Wu, "Serverless computing: One step forward, two steps back," arXiv preprint arXiv:1812.03651, 2018.

[12] G. Adzic and R. Chatley, "Serverless computing: economic and architectural impact," in Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. ACM, 2017, pp. 884–889

[13] Manu Bhata. (2018). Your Guide to Qualitative and Quantitative Data Analysis Methods. Retrieved from https://humansofdata.atlan.com/2018/09/qualitativequantitativedataanalysismethods/#Data_Preparation_and_Basic_Data_Analysis

[14] Jane Webster and Richard T. Watson. 2002. Analyzing the past to prepare for the future: writing a literature review.

[15] Kim, W. (2016). Cloud Architecture: A Preliminary Look. Proceedings of the 9th International Conference on Advances in Mobile Computing and Multimedia, pp 2-6, Vietnam.

[16] Diaby, T., Rad, B.B., & Rana, M.E. (2017). Cloud Computing Adoption: A Short Review of Issues and Challenges. Proceedings of the 2017 International Conference on Ecommerce, E-Business and E-Government, pp: 51-55, NY, USA.

[17] Hassan et al. "Survey on serverless computing", Journal of Cloud Computing: Advances, Systems and Applications (2021) 10:39 https://doi.org/10.1186/s13677-021-00253-7

[19] G. C. Fox, V. Ishakian, V. Muthusamy, and A. Slominski, "Status of serverless computing and function-as-a-service (faas) in industry and research," arXiv preprint arXiv:1708.08028, 2017.

[18] E. Jonas, J. Schleier-Smith, V. Sreekanti, C.-C. Tsai, A. Khandelwal, Q. Pu, V. Shankar, J. Carreira, K. Krauth, N. Yadwadkar et al., “Cloud programming simplified: a berkeley view on serverless computing,” arXiv preprint arXiv:1902.03383, 2019.

[20] I. Baldini, P. Castro, K. Chang, P. Cheng, S. Fink, V. Ishakian, N. Mitchell, V. Muthusamy, R. Rabbah, A. Slominski et al., "Serverless computing: Current trends and open problems," in Research Advances in Cloud Computing. Springer, 2017, pp. 1–20.

[21] G. McGrath and P. R. Brenner, "Serverless computing: Design, implementation, and performance," in 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW).IEEE, 2017, pp. 405–410.

[22] W. Lloyd, S. Ramesh, S. Chinthalapati, L. Ly, and S. Pallickara, "Serverless computing: An investigation of factors influencing microservice performance," in 2018 IEEE International Conferenceon Cloud Engineering (IC2E). IEEE, 2018, pp. 159–169.

[23] M. Shahrad, J. Balkind, and D. Wentzlaff, "Architectural implications of function-as-a-service computing," in Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitec ture. ACM, 2019, pp. 1063–1075.

[24] Wang L, Li M, Zhang Y, Ristenpart T, Swift M (2018) Peeking behind the curtains of serverless platforms. In: Proceedings of the 2018 USENIX Conference on Usenix Annual Technical Conference (USENIX ATC' 18). USENIX Association, USA. pp 133–145

[25] Horovitz S, Amos R, Baruch O, Cohen T, Oyar T, Deri A (2019) Faastest -machine learning based cost and performance faas optimization. In:Coppola M, Carlini E, D'Agostino D, Altmann J, Bañares JÁ (eds). Economics of Grids, Clouds, Systems, and Services. Springer, Cham. pp 171–186. https://doi.org/10.1007/978-3-030-13342-9_15

[26] Rajan RAP (2018) Serverless architecture - a revolution in cloud computing. In: 2018 Tenth International Conference on Advanced Computing (ICoAC). pp 88–93. https://doi.org/10.1109/ICoAC44903.2018.8939081

[27] Palade A, Kazmi A, Clarke S (2019) An evaluation of open source serverless computing frameworks support at the edge. In: 2019 IEEE World Congress on Services (SERVICES), vol. 2642-939X. pp 206–211. https://doi.org/10.1109/SERVICES.2019.00057

[28] Lynn T, Rosati P, Lejeune A, Emeakaroha V (2017) A preliminary review of enterprise serverless cloud computing (function-as-a-service) platforms. In: 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom). pp 162–169. https://doi.org/10.1109/ CloudCom.2017.15

[29] Elsakhawy M, Bauer M (2020) Faas2f: A framework for defining execution-sla in serverless computing. In: 2020 IEEE Cloud Summit. pp 58–65. https://doi.org/10.1109/IEEECloudSummit48914.2020.00015

[30] Jambunathan B, Yoganathan K (2018) Architecture decision on using microservices or serverless functions with containers. In: 2018 International Conference on Current Trends Towards Converging Technologies

[31] Yussupov V, Breitenbücher U, Leymann F, Müller C (2019) Facing the unplanned migration of serverless applications: A study on portability problems, solutions, and dead ends. In: Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing (UCC'19). Association for Computing Machinery, New York. pp 273–283. https://doi.org/10.1145/3344341.3368813

[32] Carver B, Zhang J, Wang A, Cheng Y (2019) In search of a fast and efficient serverless dag engine. In: 2019 IEEE/ACM Fourth International Parallel Data Systems Workshop (PDSW). pp 1–10. https://doi.org/10.1109/PDSW49588.2019.00005

[33] Rajan RAP (2018) Serverless architecture - a revolution in cloud computing. In: 2018 Tenth International Conference on Advanced Computing (ICoAC). pp 88–93. https://doi.org/10.1109/ICoAC44903.2018.8939081

[34] Enes J, Expósito RR, 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

[35] Aditya P, Akkus IE, Beck A, Chen R, Hilt V, Rimac I, Satzke K, Stein M (2019) Will serverless computing revolutionize nfv? Proc IEEE 107(4):667–678. https://doi.org/10.1109/JPROC.2019.2898101

[36] Keshavarzian A, Sharifian S, Seyedin S (2019) Modified deep residual network architecture deployed on serverless framework of iot platform based on human activity recognition application. Futur Gener Comput Syst 101:14–28. https://doi.org/10.1016/j

[37] van Eyk E, Grohmann J, Eismann S, Bauer A, Versluis L, Toader L, Schmitt N, Herbst N, Abad CL, Iosup A (2019) The spec-rg reference architecture for faas: From microservices and containers to serverless platforms. IEEE Internet Comput 23(6):7–18. https://doi.org/10.1109/MIC.2019.2952061

[38] Tankov V, Golubev Y, Bryksin T (2019) Kotless: A serverless framework for kotlin. In: 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE). pp 1110–1113. https://doi.org/10.1109/ASE.2019.00114

[39] Akkus IE, Chen R, Rimac I, Stein M, Satzke K, Beck A, Aditya P, Hilt V (2018) Sand: Towards high-performance serverless computing. In: Proceedings of the 2018 USENIX Conference on Usenix Annual Technical Conference (USENIX ATC '18). USENIX Association, USA. pp 923–935

[40] Xu Z, Zhang H, Geng X, Wu Q, Ma H (2019) Adaptive function launching acceleration in serverless computing platforms. In: 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS). pp 9–16. https://doi.org/10.1109/ICPADS47876.2019.00011

[41] McGrath G, Brenner PR (2017) Serverless computing: Design, implementation, and performance. In: 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW). pp 405–410. https://doi.org/10.1109/ICDCSW.2017.36

[42] Lloyd W, Vu M, Zhang B, David O, Leavesley G (2018) Improving application migration to serverless computing platforms: Latency mitigation with keep-alive workloads. In: 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion). pp 195–200. https://doi.org/10.1109/UCCCompanion.2018.00056

[43] Soltani B, Ghenai A, Zeghib N (2018) A migration-based approach to execute long-duration multi-cloud serverless functions. In: Maamri R, Belala F (eds). Proceedings of the 3rd International Conference on Advanced Aspects of Software Engineering, ICAASE 2018, Constantine, Algeria, December 1-2, 2018 (CEUR Workshop Proceedings), vol. 2326.pp 42–50

[44] Leitner P, Wittern E, Spillner J, Hummer W (2019) A mixed-method empirical study of function-as-a-service software development in industrial practice. J Syst Softw 149:340–359. https://doi.org/10.1016/j.jss.2018.12.013

[45] Martins H, Araujo F, da Cunha PR (2020) Benchmarking serverless computing platforms. J Grid Comput 18(4):691–709. https://doi.org/10.1007/s10723-020-09523-1

46. Kuhlenkamp J, Werner S, Borges MC, El Tal K, Tai S (2019) An evaluation of faas platforms as a foundation for serverless big data processing. In: Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing (UCC'19). Association for Computing Machinery, New York. pp 1–9. https://doi.org/10.1145/3344341.3368796
8796

[47] A. Eivy, "Be Wary of the Economics of "Serverless" Cloud Computing," IEEE Cloud Computing, vol. 4, (2), pp. 6-12, 2017.

[48] T. Lynn et al., "A preliminary review of enterprise serverless cloud computing (function-as-a-service) platforms," in 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 2017.

[49] N. Bila et al., "Leveraging the serverless architecture for securing Linux containers," in 2017 IEEE 37th International Conference on

Distributed Computing Systems Workshops (ICDCSW), 2017.

[50] A. Mohamed, "A history of cloud computing," Computer Weekly.com, 2018. [Online]. Available: https://www.computerweekly.com/feature/A-history-of cloud computing.

[51] Open Source Collective 501c6 (Non-Profit) - APEX Software, "APEX | Serverless Infrastructure.," Github, 2018. [Online]. Available: https://github.com/apex/apex.

[52] A. Patrizio, "One in Five Serverless Apps has a Critical Security Vulnerability," NetworkWorld, Apr 12, 2018. [Online].Available: https://www.networkworld.com/article/3268415/security/one-in five-serverless-apps-has-a-critical-security-vulnerability.html.

[53] Amazon Web Services, "Serverless Computing and Applications," AWS, 2018. [Online]. Available: https://aws.amazon.com/serverless.

[54] Microsoft, "Serverless Computing," Azure, 2018. [Online].Available: https://azure.microsoft.com/enus/overview/serverless computing.

[55] Google, "Serverless," GCP, 2018. [Online]. Available: https://cloud.google.com/serverless.