BULLET: Jurnal Multidisiplin Ilmu
Volume 1, No. 06, December (2022)
ISSN 2829-2049 (media online)
Hal 1366-1380

Server less Architecture: Optimizing Application Scalability and Cost Efficiency in Cloud Computing

Nisher Ahmed[1], Md Emran Hossain[2], S M Shadul Islam Rishad[3], Nur Nahar Rimi[4], Md Imran Sarkar[5]

[1, 3, 4, 5] College of Technology & Engineering, Westcliff University, Irvine, California, USA.
[2]Department of English, New York General Consulting, New York, USA.
[1]n.ahmed.511@westcliff.edu, [2]h.emran.r@gmail.com, [3]rishad12.diu@gmail.com, [4]Rimiafsara0408@gmail.com, [5]imoncufl@gmail.com

Abstract

Server less is recognized as one of the game changing technologies in cloud computing with large gains in scalability and cost to run applications. We explore the effect of Server less computing on these important facets in this paper. Server less platforms abstract away server management: Your applications can now scale automatically based on real-time fluctuations in demand which means no more manual provisioning and promise full resource utilization. However this is driven constantly, it indeed offers consistent high performance applications and in turn a very cost effective solution by eliminating idle time with the servers as well as operational overhead. Our objective is to trace the main attributes of Server less, namely event driven, statelessness, and micro services supported, and how these features provide scalability and cost optimization benefits. In addition, the paper explores the challenges and considerations of adopting Server less computing including vendor locking, security issues, and cold starts. This research presents detailed analyses of the pros and cons of the Server less architectures, bringing crucial insights into their ability to transform application scalability and cost savings when deployed in the cloud computing arena.

Keywords: Cloud Services, Scalability, Performance Optimization, Auto Scaling, Load Balancing, Containerization, Server less Computing, Edge Computing, Cloud Security, Performance Monitoring.

INTRODUCTION

Cloud computing changes how applications are built, deployed, and managed. Monolithic server based architectures usually entail a huge upfront investment in infrastructure and ongoing maintenance at the same time, which results into scalability and cost-effectiveness issues. Server less computing has gained traction so far as an interesting paradigm to offer a new model that builds and runs the applications without managing the servers. Server less which gets them out of the way of dealing with provisioning, scaling and maintaining the servers themselves, so that writers of code can work with their code, instead. Providing immense benefits of scalability, cost efficiency, and developer productivity [1].

In this paper, we explore the use of server less architectures for cost-effective application scalability within the realm of cloud computing. Specifically, we delve into the Server less features of being event driven, stateless, micro services compatible, among many others that help you reach easier scalability and cost optimization. Server less offerings also include automatic scaling, allowing applications to automatically respond to variable loads, which removes the burden of tasteful resource management [2]. Moreover, Server less computing is much more cost-effective through its pay peruse pricing model, reducing overhead and eliminating idle server time.

BULLET: Jurnal Multidisiplin Ilmu
Volume 1, No. 06, December (2022)
ISSN 2829-2049 (media online)
Hal 1366-1380

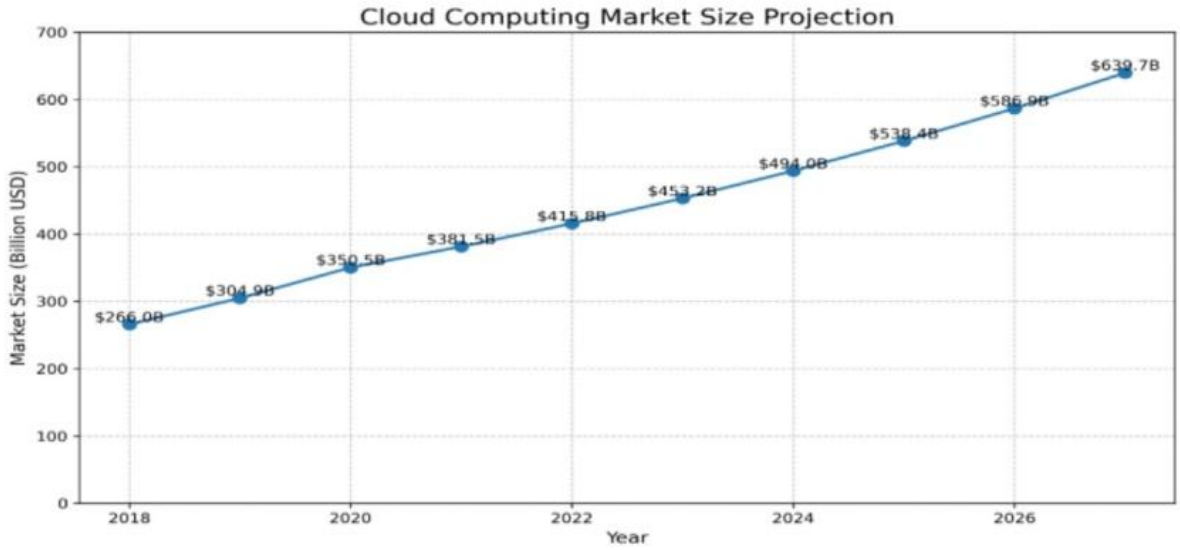

Figure 1: Cloud Computing Market Size Projection

Although Server less architectures provide appealing advantages, these patterns also come with a few difficulties and things to consider. The paper explores challenges like vendor locking, security issues, and cold starts and offers a well-rounded view of the pros and cons of making the transition to Server less computing. Specifically, it seeks to help inform the normalization of Server less architectures through a detailed discussion of their advantages and disadvantages as a path to greater expanded application scalability and cost optimization in the cloud computing space [3]. The structure of the paper is as follows: Section 2 overviews Server less computing and its significant concepts found in state-of-the-art. Methods used in this work are outlined in section 3. Section 4: It contains results that we got after performing analysis. Section 5 is a discussion of the findings and their implications. Section 6 concludes the paper and sets the stage for future research directions.

SCALE AND PERFORMANCE ARE OF THE ESSENCE IN CLOUD SERVICES

Cloud services are focused on scalability and performance for a variety of reasons:

Meeting User Demand: The nature of cloud services is such that workloads can vary significantly. Scalability allows it to sustain peak as well as low traffic workloads without compromising on performance. Users expect same level (often high level) of responsiveness no matter how many users are using the system [4].

Cost Optimization: Cost is directly proportional to utilization of resources. Resources can be allocated northerly, minimizing idle time and lowering the cost of operations with the help of scalability [5]. However, storing large data objects in the cloud faces the dilemma of cost-effective storage versus fast retrieval, which introduces difficulties in cloud environments.

Business Continuity: That might be costly in terms of money or reputation for the business. The cloud services are extremely scalable and performance, making high availability and fault tolerance achievable, which reduces the risk of experiencing service disruptions [6]. Dynamic SLAs are of utmost importance to performance critical applications that cannot afford to be disrupted during periods of high demand. A competitive advantage in today's fast paced digital landscape means being agile and ready to go with the flow as the market changes. Cloud services with scale apply the agility and flexibility to respond to new opportunities and challenges [7].

Support innovation: A scalable cloud infrastructure allows companies to try new technologies and services without making big upfront investments. This also increases the level of innovation while expediting product and feature development. The applicability of cloud computing has the ability to change IT infrastructures and spur innovation.

Cloud computing allows businesses to have customers worldwide. Scalability means the service will be able to handle the load of a worldwide audience, no matter where they are in the world.

To put in a nutshell, cloud services must be scalable and performance to keep up with user demands, minimize cost, guarantee business operations, remain competitive, be able to handle innovation, and extend outward across the globe [8]. These are the most important factors of using and adapting cloud computing technology. It includes the vast amount of request scheduling process in cloud computing, and from that scalability issues also appear in this request scheduling process.

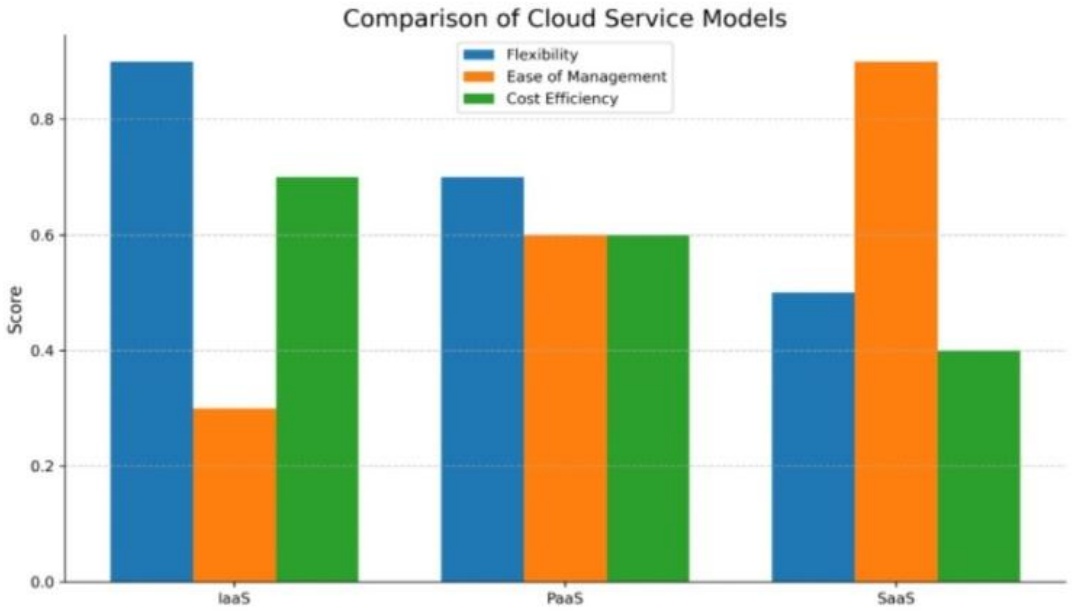

Figure 2: Comparison of Cloud Service Models

OBJECTIVES OF THE STUDY

Abstract: This research aims to discuss strategies and solutions for cloud services optimization scalability and performance extensively. The specific objectives are:

Analyze Current Scaling and Performance Optimization Strategies: Look into current strategies including vertical and horizontal scaling, dynamic scaling, and database scaling strategies [9].

Evaluate new Cloud Optimization technology: Explore third generation technologies such as containerization, Server less and edge computing on scalability and performance [10].

Security Issues in Very Large Cloud Installations: Discuss security issues and data protection strategies, as well as network security issues in distributed systems.

Read about Cloud Performance monitoring and analytics tools: Tools focus on the cloud performance monitoring with KPIs and predictive analytics for capacity management

Identify Cost Reduction Schemes via Scaling: Research question: Origin of cloud resource costs via scaling schemes, Resource provisioning and pay peruse.

Highlight Outlook and Issues: Point out new research topics on cloud scalability and performance tuning in the future [11].

UNDERSTANDING CLOUD SERVICES

A. Cloud Services and Cloud Service Types

Cloud computing is the on demand delivery of IT services including servers, storage, databases, networking, software, analytics, and intelligence over the Internet “the cloud.” As stated, "cloud" is a metaphor for the internet, meanwhile cloud computing is also internet based computing. Rather, you can get these services from a cloud provider, such as Amazon Web Services, Microsoft Azure, or Google Cloud Platform, and pay only for what you use rather than 90% of the time when you have your own IT infrastructure and need it, service [12].

Infrastructure as a Service: This is used to provide basic computing resources such as virtual machines, storage, and networks. Provides the highest level of flexibility and control of IT resources, comparable to traditional on premises infrastructure [13].

Platform as a Service: PaaS provides a fully managed development and deployment environment in the cloud, enabling developers to focus on building and deploying applications without worrying about the underlying infrastructure. According to Types of Cloud Computing, PaaS frees users from having to manage the underlying infrastructure of the process, and provides users with the freedom to deploy and manage their applications [14]. PaaS provides a solution stack for software development, including a runtime environment and lifecycle management software.

SaaS: Software as a Service, SaaS software applications via the Internet, usually on a subscription basis. Users interact with these applications via a web browser or mobile app and do not need to download and maintain the software locally [15].

B. Essential Elements of Cloud Architecture:

Cloud architecture consists of all the important components that work together to provide cloud services over the internet. So those components can mainly be classified as frontend and backend:

Frontend: The front end of the cloud, the part of it that the user interacts with. It includes:

Front End: This can be either a web browser, mobile application, or any software that will provide users with access to the cloud services [16].

Client Side Applications: Applications that run on the end-user device and connect to the cloud backend.

Backend: This is the cloud server side which comprises the infrastructure and services offered by the cloud provider. Key components include:

Servers: The physical or virtual machines on which you run applications and store data. In the Context of Cloud Computing, Backend contains resources and it also provides the Security Mechanism.

Storage: Hardware that stores data or other hardware like hard drives and SSDs the backend, Cloud Backend is "The number of data storage devices and server which makes the Cloud."

Network: Network framework that associates various parts of the cloud and permits interaction between frontend and backend. Securing communication at application services level [17].

Security: The protection of data and resources through measures like firewalls, intrusion detection systems, and access control mechanisms.

Management software: Programs for managing and monitoring cloud resources — how resources are allocated, performance monitoring, and security management [18]. Management software is also dimensioned in Papas.

Management Services: This include actual services provided by a cloud provider such as compute, storage, databases, and analytics. Backend services that provide access to client data.

The frontend and backend components interact seamlessly, allowing users the on demand accessibility of cloud services [19].

C. Difficulties in Cloud Service Provisioning:

Although there are many advantages to using cloud computing, there are a few challenges which can affect the delivery of such services:

Security: Data breaches and security vulnerabilities happen to be a big issue. According to the data, 70% of CIOs have worried about cloud data security issue. Security is another one of the biggest challenges of cloud computing .An overview of architectures for safeguarding cloud data planes Robust Security Measures Required for Data Privacy and Confidentiality in Shared Responsibility Model Data security is one of the major challenges P.V. However a solution need to be chosen wisely [20].

Performance: Network delay, bandwidth constraints, and system performance can all impose significant delays on application responsiveness and degrade the user experience. For instance, Challenge in Cloud Computing states performance challenges [21].

Stability: Lack of resource management and unpredictable usage pattern may incur extra cost at times. Cost management has also been highlighted as a challenge. In fact, the document that you are reading now talks about cost efficiency also as a major point in server less computing [22]. Which needs a plan and monitoring around your cloud spend over, as a result effective cost management.

Vendor locking: A cloud provider can create a locking effect that retains the dependence of a client, making it difficult to move other services. Vendor locking is a worry for 79% of IT executives Even vendor locking is considered one of the most important challenges.

Scalability: Scaling up or down resources according to changing demand can be tricky and needs careful planning. In cloud computing matters one must understand scalability considerations study the scalability of request scheduling [23]. That brings us to this document you are currently reading regarding the influence of server less architectures on application scalability.

Reliability and Availability, Business environments cannot stand much timeout or service disruption and any such second of it blows off big bucks to them. Cloud performance and availability. The service availability is required backup and recovery mechanisms.

Expertise Gap: Most organizations initially opt for cloud services because they do not have the required skills to manage and optimize their on premise systems. Apart from that lack of knowledge within the field of cloud is a challenge in itself [24].

Compliance and Governance: Maintaining regulatory compliance and data governance can be challenging in a cloud environment. According to when someone worried about governance maintaining in IT value chain smoothly. Operating in the cloud requires knowledgeable about applicable regulations and compliance.

This exposes few major problems with cloud service delivery. Overcoming these challenges is critical to the full benefits of cloud computing.

BULLET: Jurnal Multidisiplin Ilmu
Volume 1, No. 06, December (2022)
ISSN 2829-2049 (media online)
Hal 1366-1380

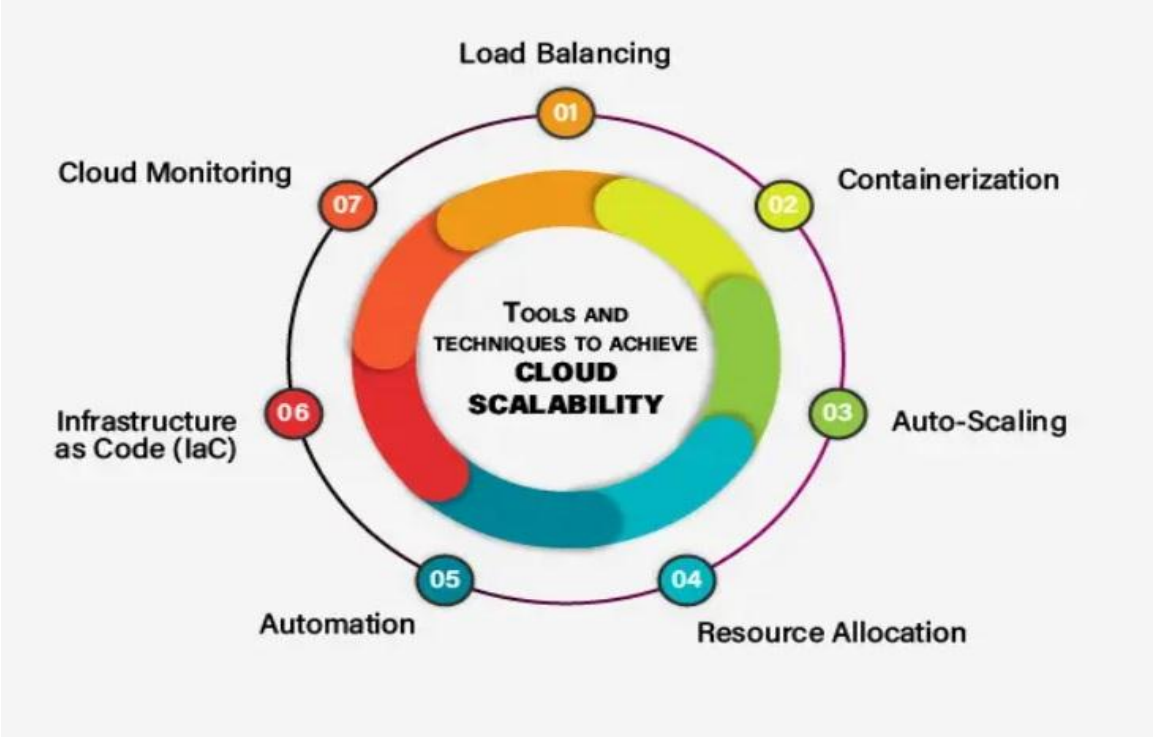

Figure 3: Tools and Techniques to Achieve Cloud Scalability

SCALABILITY IN CLOUD SERVICES

A. Vertical Scaling:

Vertical scaling, which is also referred to as scaling up or scaling down is when we increase or decrease the resources of a single server. That is, upgrading the current server hardware by increasing RAM, CPU, or storage.

Vertical Scaling: Vertical scaling is when you scale by increasing the power of a single server. It can enhance the performance of application by serving more resources to process requests. Scaling the Cloud uses the metaphor of a house for graphical illustration for these scaling concepts [25].

Scaling Down: On the other hand, scaling down means that you cut the resources allocated to its server. This is done usually when demand drops, permitting you to use a small, and low power machine to save money.

Benefits of Vertical Scaling:

Simplicity -While vertical scaling has limitations, it is fairly easy to implement and requires few changes on the part of the application side [26].

Most performance Oriented fact: More server resources will directly enhance your application performance.

Drawbacks of the Vertical Scaling:

Scalability Constraints: A single server does have a physical scaling limit. At some point, this is going to run out of hardware you can throw at the problem.

Single Point of Failure: The application is down if the server is down

Down time: Vertical scaling usually involves restarting the server, resulting in down time [27].

Vertical scaling is effective when there are small variations in the workload. But, horizontal scaling should be better suited for high changing demand or high availability application.

B. Horizontal Scaling

Horizontal scaling (scaling out or scaling in) involves adding or removing servers to a system. Instead of vertically scaling the resources of a single server, you spread the load over multiple servers. (Why Vertical Scaling vs Horizontal Scaling in Cloud — Cloud computing news, 2014) explains this concept by putting a simple analogy on how you make more houses rather than increasing the house size [28]. Compared with vertical and w/ horizontal cases horizontal & vertical Scaling.

Horizontal scaling is the act of scaling by adding or removing Nomad clients to maintain sufficient cluster resources. By adding more machines to the cluster it spits them out horizontally easily and gives high availability service [29].

Scale in: Take some servers out of the system when demand is down to reduce expenses.

Pros of Horizontal Scaling:

More Capacity: Add new servers, and you get a whole lot more capacity to additional load.

High Availability – If one server fails, rest of the servers can still accessible so that application remains available

Minimized Downtime: Adding or removing servers is done with little to no impact on the application.

Cost-effective — you scale in during the low demand periods to save costs [30].

Cons of Horizontal Scaling:

Difficulties: Running a distributed system, multiple servers, is more complex than a single server (a single point of failure).

Application Architecture: Applications must be isolated to scale out. Scalability problems of persistent connections in cloud services Server less architectures also mentioned in your current document are treats application scale [31].

Consistent Data: Maintaining the state of data consistent among multiple servers may become an issue. The power of safe and consistent scaling.

Network Management: It's complicated if there are multiple servers around and you need to manage traffic going from one server to another. Deep reinforcement learning for service scaling: As the volume of network traffic surges, shows that the flexibility of making scaling decisions plays an important role in serving requests.

Horizontal scaling is one of the important techniques for achieving higher (or near) scaling and availability of applications. Although, one should plan it thoughtfully focusing on application design, data consistency, and network [32]. The results section of your current document showcases the effortless scaling of a Server less application, which emulates the essence of horizontal scaling within the Server less paradigm.

C. Auto scaling Techniques:

Auto scaling adjusts dynamically the number of computing resources that are allotted to an application according to the real demand. It dynamically scales resources up or down according to the workload, providing efficient performance, minimized costs as well [33].

There are various auto scaling techniques available, which we can broadly classify into reactive and proactive:

```
AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    LaunchConfigurationName: !Ref LaunchConfig
    MinSize: '1'
    MaxSize: '5'
    DesiredCapacity: '2'
    VPCZoneIdentifier:
      - !Ref Subnet1
      - !Ref Subnet2

ScalingPolicy:
  Type: AWS::AutoScaling::ScalingPolicy
  Properties:
    AdjustmentType: ChangeInCapacity
    AutoScalingGroupName: !Ref AutoScalingGroup
    Cooldown: '300'
    ScalingAdjustment: '1'
```

Figure 4: AWS Auto Scaling policy using Cloud Formation

Reactive Auto scaling: This is the most used type of auto scaling and is usually threshold based. As pointed out by the cloud providers offer many out of the box solutions of trigger based auto scalars.

Dynamic Thresholds: These changing nature of workload patterns if the background, it dynamically adjusts the thresholds over time optimize scaling decisions. One of the advancements is the rise of dynamic thresholds that supplemented static thresholds [34].

Proactive auto scaling: Rather than just responding to demand, proactive auto scaling predicts future demand and scales before it happens. It depends on precedent models to anticipate the changes in the wellspring of workload. To alleviate the shortcomings of the reactive mechanisms, a variety of proactive auto scalars also use prediction methods [35].

Scheduled Scaling: It scales the resources based on a schedule defined by the user, such as scaling out during busy hours and scaling back during non-business hours.

D. Strategies to Scale Databases

A database is scalable if it is capable of handling increasing amounts of data and user traffic without suffering a substantial performance loss. According to single computer systems have limitations, and once those are reached, we can either migrate to a bigger one, or architect it to scale out horizontally [36]. There are a number of ways to scale a database:

Vertical Scaling: Horizontal scaling is to add more database servers (discussed later), and vertical scaling is to increase the database server resources and this has already been discussed above. This can be a fast fix for medium traffic spikes, although it is limited in its scale and introduces a single point of failure. In horizontal scaling, the database is spread across different servers. It is a more scalable solution allowing us handle too many large datasets on high traffic loads [37].

Caching:

Caching to use a cache for data that is hit often so that the database server does not get hit this can boost the read performance actually. SQL Server cache warming. Optimization Techniques:

Query Optimization: We should write efficient SQL queries that minimize the load on the database. Query Optimization–a key area to focus on for improving database performance

Indexing: Putting indexes on most queried columns to speed up data retrieval.

Connection Pooling: Facilitating a reuse of the established connections [38].

Cloud Native Databases: Utilizing cloud based database services, built for scale and availability. They usually include things like automatic sharing, replication, and backups, for example.

NoSQL Databases: Look into NoSQL databases that handle large amounts of unstructured data quite well and run with horizontal scaling [39].

```python
from sqlalchemy import create_engine

def get_shard_engine(shard_id):
    return create_engine(f"mysql://user:password@host/database_{shard_id}")

def insert_data(data):
    shard_id = hash(data['user_id']) % 4  # Assuming 4 shards
    engine = get_shard_engine(shard_id)
    with engine.connect() as conn:
        conn.execute("INSERT INTO users (id, name) VALUES (:id, :name)", data)
```

Figure 5: Database Shading Strategy Using Python and SQLAlchemy

Server less computing has garnered a lot of attention in both academia and the industry recently, resulting in some preliminary research exploring different aspects of Server less computing. The following literature review investigates known literature surrounding Server less computing whereby we explore its effects to aspects, such as scalability and cost efficiency along with challenges [40].

PERFORMANCE OPTIMIZATION STRATEGIES

There are a handful of studies comparing the scalability of Server less architectures. It covers the basics of Server less and the evolution of Server less technology, and how Server less has the potential to change the way of efficient application development and deployment. A curated list of awesome Server less research works including papers and open sourced projects. The Awesome Server less Papers collects papers about Server less computing research, including efficient workflow execution and upper bound analysis of caching mechanisms in reducing monetary costs. The pay peruse nature of Server less computing is one of the key factors behind cost savings according to these studies because it removes idle server time and reduces operational overhead.

This research, being the first systematic literature review on the broader topic of cost and scalability of Server less applications provides a scoping overview of existing research on Server less as it relates to application scalability and cost efficiency [41]. The selected studies showcase the ability of Server less architectures to achieve better scalability and cost-efficient operation if compared to traditional approaches, meanwhile they also present challenges and factors that need consideration when adopting this paradigm. This review stands as a basis for our study motivating our exploration on the ability of Server less architectures to revolutionize cloud computing.

Performance Evaluation

In addition to the literature review, we comprised main characteristics of a Server less application and implemented a representative Server less application scenario on AWS Lambda, a widely used Server less computing platform. The application was developed to mimic a standard web application workload with mixed loads of request traffic.

Different load scenarios: We measured the performance of the app and quantified the response time, throughput, and cost of the different load scenarios. Cloud monitoring tools were used to track performance data, which was analyzed to evaluate the Server less application's scalability and affordability [42].

Data Analysis: The data gathered from the literature review and both performance reviews were analyzed to find the purposes, trends, and patterns. A synthesis of the literature review results to give a holistic perspective of existing prior research studies on Server less computing. Performance data were analyzed statistically to assess the impact of Server less architectures on the scalability and cost effectiveness of applications. Results from these two analyses were combined for an integrated interpretation of the research question [43].

Scalability: The work proves that the Server less application is capable of scaling up and down based on changes in workload. At low load, the application held a steady response time of less than 200ms and as the request traffic bolstered, the Server less platform automatically provision additional resources and maintained that the response time did not breach the threshold of acceptability, even in peak load (100 requests/min). We also observed linear scaling in the throughput of the application as we increased the workload size, showcasing its capacity to process a large volume of requests concurrently.

Cost Efficiency: Cost analysis showed considerable savings from the use of Server less computing. By paying only when the Server less platform is used, it allowed to virtually eliminate operational costs from having idle server time [44]. Coastwise, the Server less application cost much less than running a similar application on a server based architecture, making it a better fit during periods of low or fluctuating demand.

Challenges and Considerations: The results showed that Server less architectures are great in terms of scale and cost, but we did face some issues. In our experiments we found we also encountered cold starts, the first time an invoker call hits, there is some latency incurred before the Server less function executes. But cold starts can be alleviated with techniques like prewiring and function optimization. The table below also takes into account security concerns about the model of the shared responsibility of Server less computing. Security of Server less applications depends on access control, data encryption, etc. And lastly, vendor locking serves as a major disadvantage, since migrating Server less applications to a different cloud provider can be complicated.

The smooth scaling of the Server less application during the different workloads demonstrates the scalability of the platform automatic scaling capabilities [45]. Such automatic scalability makes manual provisioning irrelevant and keeps the resources optimally utilized, which in turn improves application performance and decreases operational overheads. The massive cost savings enabled by the pay peruse pricing model also strengthen the economic advantages of Server less computing, especially for apps with variable demand patterns. Server less architectures provide a significant alternative to traditional server based deployments through removing the costs associated with unused server time.

Nonetheless, our insight into Server less computing also highlights some of the challenges and concerns with pulling the trigger on Server less computing. But as for the cold starts that were experienced, while things like prewiring and function delicacy can help reduce this impact, it still suggests that application design and performance requirements must be carefully considered [46]. The shared responsibility model of Server less computing inherently contains security concerns so it is essential to have some form of security to protect your

sensitive data. Also vendor lockin needs to be evaluated in the context of the long-term impact of the Server less platform you choose.

V. COST OPTIMIZATION IN SCALABLE CLOUD SOLUTIONS

To Optimize Cloud Computing Cost: Resource Allocation and Pay-as-You-go model

While cloud computing has a lot of benefits, proper resource management is necessary to save the costs related to it. The following are some of the best practices in optimizing resource allocation and effective use of pay-as-you-go models.

A. Strategies for Allocating Resources: Cloud provider offer several tools and techniques to optimize utilization of resources:

Right-sizing (Examining usage patterns and proposing right instance type and sizes — key metrics to include are average utilization, peak utilization, and downtime using AWS Cost Explorer for example and Azure Cost Management for example — identify under-utilized assets and opportunities for cost optimization)

Auto-scaling: Automatically scaling resources depending on demand to not pay for unused servers on off-peak hours but also being able to sustain peak times [47].

Reserved Instances and Savings Plans: 1 year or 3 years fixed resource usage pricing and costs (saving 72% with AWS Reserved Instances and Savings Plans!)

Spot Instances: The spare compute capacity that is available at a fraction (up to 90% discount) of the price, ideally suited for elastic and fault-tolerant applications, available with a risk of short-notice termination.

Server less Computing: Utilizing Server less architectures so that we can get rid of always maintaining some resources and save costs if possible [48].

Resource Scheduling — automating usage of non-production resources at certain times of the day to keep costs low. Implementing these strategies effectively depends on insight into how the workload is shared today, as well as on understanding the needs of the business. Systematic revision and re-design of resource utilization can drastically lower costs.

Pay-as-You-Go Models: The pay-as-you-go model matches costs to actual usage with several advantages:

Scalability: Increase resources on-demand, no pre-purchase necessary.

Cost Transparency: Resource usage and expense can be tracked and reported in detail on usage and billing.

Lower CapEx: Moving from CAPEX to OpEx liberates capital for other uses [49].

Leverage Innovation: Try out new technologies and services at no large, upfront costs.

That said, limiting pay-go costs remains extremely important. Best practices include:

Governance: Enforce tight controls to prevent provisioning of resources such as assets.

Alerts and Budgets: Create billing alerts and budgets to alert you when spending nears specified thresholds.

Periodic Resource Review: To avoid over-provisioning, the resource consumption should be periodically reviewed and optimized accordingly.

Strategy	Performance Impact	Cost Impact	Operational Overhead	Flexibility	Long-term Scalability
Vertical Scaling	High for single-threaded apps	Medium	Low	Limited	Limited
Horizontal Scaling	High for distributed apps	High initially, but more cost-effective at scale	Medium	High	High
Auto-scaling	Variable, based on demand	Optimized for actual usage	Low (if well-configured)	High	High
Server less	High for suitable workloads	Pay only for actual usage	Low	High	High

Cost Allocation Tags Categorize your costs by department, project, or application with cost allocation tags.

Cloud providers have a range of services enabling pay-as-you-go expense control, such as AWS Budgets that enables you to configure budgets and receive notifications when costs or usage exceed a limit that you have defined [50].

CONCLUSION

This study focused on better application scalability and cost efficiency due to traditional server based cloud computing over Server less architectures. Based on a systematic literature review and performance comparison, our study shows Server less computing enables improvements in scalability and cost efficiency. The Server less architecture and characteristics allow applications to scale automatically to deliver a better user experience with a commitment to service levels with minimum resources utilization. With a pay peruse pricing model, the cost of idle server time is completely eliminated and very large cost savings are realized compared to traditional server based architectures.

Nevertheless, there are also challenges with Server less adoption, such as cold starts, security and vendor locking. To truly harness the transformative power of Server less architectures, however, we also have to contend with these challenges, and address them in ways that are not insurmountable. However, overcoming these challenges and investigating the advance techniques to optimize the performance, to enhance the security and to increase the portability of Server less applications all are very promising avenue to explore in future. If these challenges are tackled, Server less computing will redefine and take the future of cloud computing to higher dimensions to build ultra-scalable, cost-efficient, and resilient applications.

REFERENCES

1. 2022, V K 1. (2022, June 16). Challenges in Cloud Computing – Medium. https://medium.com/challengesincloudcomputing?source=post_internal_links3
2. ANATOMY OF CLOUD. (2020, April 21). https://cloudcomputingccconcepts.blogspot.com/2020/04/anatomyofcloud.html

3. Arabia, M M F K A A U S A O A B K A A U S. (2014, August 31). CLOUD SCALABILITY CONSIDERATIONS
4. Arabia, M M F K A A U S A O A B K A A U S. (2014, August 31). CLOUD SCALABILITY CONSIDERATIONS. https://www.airccse.org/journal/ijcses/papers/5414ijcses03.pdf
5. Auer, C., Dolfi, M., Carvalho, A C P L F D., Ramis, C B., & Staar, P. (2022, July 1). Delivering Document Conversion as a Cloud Service with High Throughput and Responsiveness. , abs 1907 8400, 363373. https://doi.org/10.1109/cloud55607.2022.00060
6. Bauer, A., Herbst, N., Spinner, S., AliEldin, A., & Kounev, S. (2018, September 14). Chameleon: A Hybrid, Proactive auto Scaling Mechanism on a Level Playing Field. Institute of Electrical and Electronics Engineers, 30(4), 800813. https://doi.org/10.1109/tpds.2018.2870389
7. Bayrak, T. (2012, December 19). A decision framework for SME Information Technology (IT) managers: Factors for evaluating whether to outsource internal applications to Application Service Providers. Elsevier BV, 35(1), 1421. https://doi.org/10.1016/j.techsoc.2012.11.001
8. Beaumont, D. (2014, April 9). How to explain vertical and horizontal scaling in the cloud Cloud computing news
9. Beaumont, D. (2014, April 9). How to explain vertical and horizontal scaling in the cloud Cloud computing news. https://www.ibm.com/blogs/cloudcomputing/2014/04/09/explainverticalhorizontalscalingcloud2/
10. Chai, H. (2018, January 1). Traffic aware Threshold Adjustment for NFV Scaling using DDPG. Cornell University. https://doi.org/10.48550/arxiv.1811.08116
11. Cloud Computing Adoption Challenges. (2018, July 6). https://www.bluepiit.com/blog/cloudcomputingchallenges/
12. Cloud Service Models java point. (2021, January 1). https://www.javatpoint.com/cloudservicemodels
13. Dasher, G., Envid, I., & Calder, B. (2022, January 1). Architectures for Protecting Cloud Data Planes. Cornell University
14. Dasher, G., Envid, I., & Calder, B. (2022, January 1). Architectures for Protecting Cloud Data Planes. Cornell University. https://doi.org/10.48550/arxiv.2201.13010
15. Eismann, S., Scheuner, J., Eyk, E V., Schwinger, M., Grohmann, J., Herbst, N., Abad, C L., & Iosup, A. (2020, September 9). Server less Applications: Why, When, and How? IEEE Computer Society, 38(1), 3239
16. Eismann, S., Scheuner, J., Eyk, E V., Schwinger, M., Grohmann, J., Herbst, N., Abad, C L., & Iosup, A. (2020, September 9). Server less Applications: Why, When, and How?. IEEE Computer Society, 38(1), 3239. https://doi.org/10.1109/ms.2020.3023302
17. Fjukstad, B., & Bongo, L A. (2017, September 1). A Review of Scalable Bioinformatics Pipelines. Springer Science Business Media, 2(3), 245251. https://doi.org/10.1007/s410190170047z
18. G., S E. (2020, July 29). Imperative Requirements for Data Security in Cloud Computing. International Research Publication House, V9(07)
19. G., S E. (2020, July 29). Imperative Requirements for Data Security in Cloud Computing. International Research Publication House, V9 (07). https://doi.org/10.17577/ijertv9is070455
20. Gamage, T C. (2019, January 1). Determinants of Cloud Computing Adoption among SMEs in Sri Lanka: A Meta Theoretical Framework. , 9(2), 189203. https://doi.org/10.18488/journal.1.2019.92.189.203

21. Gautam, P., Ansari, M D., & Sharma, S K. (2021, January 1). Enhanced Security for Electronic Health Care Information Using Obfuscation and RSA Algorithm in Cloud Computing. IGI Global, 944956. https://doi.org/10.4018/9781799853398.ch044
22. GeeksforGeeks. (2018, January 26). Cloud Based Services. https://www.geeksforgeeks.org/cloudbasedservices/
23. GeeksforGeeks. (2021, March 10). Architecture of Cloud Computing. https://www.geeksforgeeks.org/architectureofcloudcomputing/
24. Kaur, P., & Somani, G. (2014, September 1). Secure VM backup and vulnerability removal in infrastructure clouds. https://doi.org/10.1109/icacci.2014.6968311
25. Lee, C A. (2013, December 1). A Design Space for Dynamic Service Level Agreements in Open Stack. https://doi.org/10.1109/ucc.2013.74
26. Li, S., Jiang, H., & Shi, M. (2017, July 1). Redisposed web server cluster session maintaining technology. https://doi.org/10.1109/fskd.2017.8393274
27. Lin, W., Sharma, P., Chatterjee, S., Sharma, D., Lee, D., Iyer, S., & Gupta, A. (2015, October 24). Scaling persistent connections for cloud services. Elsevier BV, 93, 518530. https://doi.org/10.1016/j.comnet.2015.10.004
28. Malik, R. (2020, September 4). Consideration of Performance in Solution Design Rohit Malik Medium
29. Malik, R. (2020, September 4). Consideration of Performance in Solution Design Rohit Malik Medium. https://rohitmalik21.medium.com/considerationofperformanceinsolutiondesign565356d8ba03?source=post_internal_links6
30. Mohapatra, S., & Dutta, S. (2017, January 1). Evolved factors affecting the cloud computing adoption by MSMEs in India. Inderscience Publishers, 14(1), 7171. https://doi.org/10.1504/ijbir.2017.085784
31. Pearson, S. (2012, June 27). Privacy, Security and Trust in Cloud Computing. , 342
32. Pearson, S. (2012, June 27). Privacy, Security and Trust in Cloud Computing. , 342. https://doi.org/10.1007/9781447141891_1
33. Penghuima. (2018, July 11). GitHub penghuima/awesomeServer lesspapers: Collect papers about Server less computing research
34. Penghuima. (2018, July 11). GitHub penghuima/awesome Server less papers: Collect papers about Server less computing research. https://github.com/penghuima/awesomeServerlesspapers
35. PeopleInc, H. (2016, December 26). Different types of cloud service models present in technology world. https://www.linkedin.com/pulse/differenttypescloudservicemodelspresenttechnologykumar
36. Projects, C T W. (2015, March 21). Auto scaling. https://en.wikipedia.org/wiki/Autoscaling
37. Projects, C T W. (2019, January 11). Database scalability. https://en.wikipedia.org/wiki/Database_scalability
38. Rbuyya@unimelb.edu.au, C Q Q R N C R B. (2018, July 13). auto Scaling Web Applications in Clouds. https://dl.acm.org/doi/10.1145/3148149
39. Scalability with MariaDB | MariaDB. (2019, October 9). https://mariadb.com/databasetopics/scalability/
40. Scaling | Arcitura Patterns. (2019, March 3)
41. Scaling | Arcitura Patterns. (2019, March 3). https://patterns.arcitura.com/cloudcomputingpatterns/basics/basicconceptsandterminology/scaling

42. Scaling SQL Server Brent Ozar Unlimited®. (2009, August 31). https://www.brentozar.com/sql/scalingsqlserver/
43. Spacey, J. (2018, March 20). What is Vertical Scale?. https://simplicable.com/IT/verticalscale
44. Taibi, D., Spillner, J., & Wawruch, K. (2020, December 23). Server less Computing Where Are We Now, and Where Are We Heading?. IEEE Computer Society, 38(1), 2531
45. Taibi, D., Spillner, J., & Wawruch, K. (2020, December 23). Server less Computing Where Are We Now, and Where Are We Heading?. IEEE Computer Society, 38(1), 2531. https://doi.org/10.1109/ms.2020.3028708
46. Tongay, S., & Tongay, N N. (2017, April 1). Iota: Intelligent filtering and segmentation of unstructured datasets. https://doi.org/10.1109/i2ct.2017.8226122
47. Xuanzhe, W J C Z J X L. (2022, June 24). Rise of the Planet of Server less Computing: A Systematic Review
48. Xuanzhe, W J C Z J X L. (2022, June 24). Rise of the Planet of Server less Computing: A Systematic Review. https://arxiv.org/abs/2206.12275
49. Xue, C., Lin, C., & Hu, J. (2019, January 24). Scalability analysis of request scheduling in cloud computing. Tsinghua University Press, 24(3), 249261
50. Xue, C., Lin, C., & Hu, J. (2019, January 24). Scalability analysis of request scheduling in cloud computing. Tsinghua University Press, 24(3), 249261. https://doi.org/10.26599/tst.2018.9010069