International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Serverless Computing: The Future of Cloud Applications

Tisha[1], Sunvish Charak[2], Utkarsh Vashisht[3]

[1,2,3]CGC-College OF Engineer

Abstract:
server less computing is the future requirement for developers to build and deploy applications in the cloud by abstracting server management and offering on-demand portability. Server less platforms like aws ,lambda ,azure functions and google cloud functions enable developers and organizations to focus more on code and business logic rather than managing the platform or the server. This paper explores the core of serverless computing and its advantages in real-world use cases and how it has the potential of shaping the future of cloud-based applications.

1. Introduction

Over the last decade, cloud computing has transformed the way applications are built, deployed, and scaled. Among the latest advancements in this space is Serverless Computing—which restructures the traditional client-server model by reducing the server management altogether. As a final-year computer science student, I've observed firsthand how serverless technologies are making their way into academic projects, startups, and enterprise solutions and also the small businesses alike.

While it's called 'serverless,' this computing model still relies on servers—just not ones the developer has to manage. Instead, it refers to a model where the infrastructure is fully managed by the cloud provider, enabling developers to focus only on improving the logic building of respective application. This chapter explores the fundamentals of serverless computing, its architecture, real-world use cases, advantages, limitations, and its potential to shape the next generation of software development.

2. What is Serverless Computing?

Serverless computing is a modern cloud-based paradigm where the responsibility of managing servers is entirely shifted to the cloud provider .It allows developers to build and deploy applications without needing to manage the servers that run them. The code runs in lightweight, stateless environments that are triggered by specific events—like HTTP requests, file storage changes, or scheduled time intervals.

This model is widely known through Function-as-a-Service (FaaS), which enables developers to build and deploy modular, single-purpose functions. These functions are executed only when invoked, scale automatically, and typically have a short execution time. Notable serverless platforms include AWS Lambda, Google Cloud Functions, Microsoft Azure Functions, and IBM Cloud Functions.

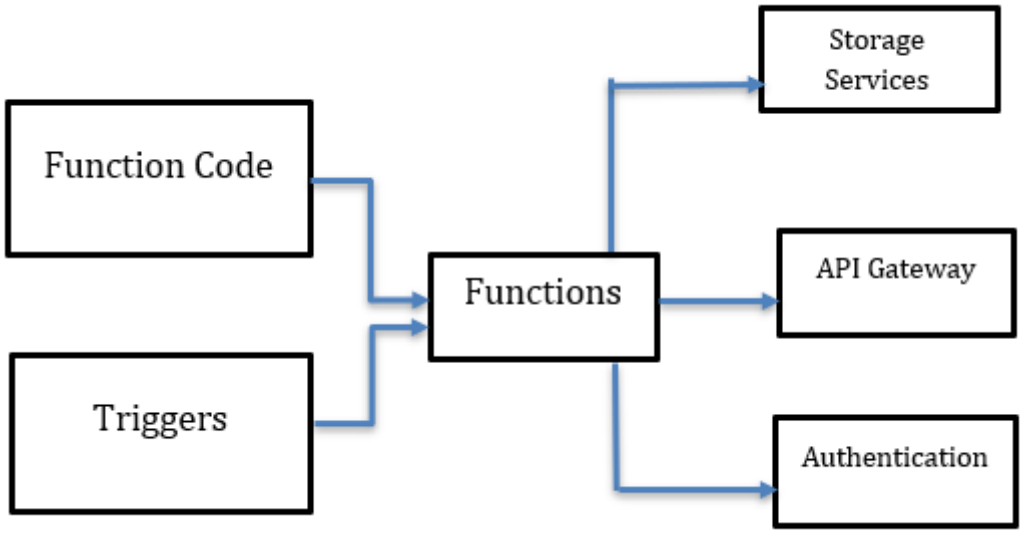
3. Architecture of Serverless Applications

A serverless setup is generally composed of multiple interconnected components, each playing a specific role in the application workflow:

- **Function Code**: The logic that performs a specific task.

- **Triggers**: Events that invoke the function (e.g., HTTP requests, database updates).
- **API Gateway** serves as the bridge between users and backend logic, taking incoming HTTP requests and directing them to the correct serverless function for processing.
- **Storage Services**: Like AWS S3 or Firebase Storage, used for persisting data.
- **Authentication**: Serverless apps often integrate with identity providers like Firebase Auth or Amazon Cognito.
- **Databases**: NoSQL (e.g., DynamoDB, Firestore) or SQL databases (e.g., Amazon RDS). Each function is deployed independently and can scale horizontally. When an event occurs, the cloud provider initializes a container, runs the function, and shuts it down automatically.

Figure 1: Serverless Application Architecture



4.Types Of Cloud

Imagine if instead of buying your own computer, you could just borrow one from the internet whenever you needed it—and it could grow or shrink depending on how much power you need. That's exactly what cloud computing is! Thanks to cloud providers like Amazon, Google, and Microsoft, you don't have to own massive hardware. You just rent the computing power you need, and they take care of everything else.

You can start small and scale up as your needs grow, without worrying about buying and maintaining expensive equipment.

4.1 Cloud Deployment Models

There are four main ways cloud computing can be set up, depending on who uses it and how it's managed. These are called deployment models:

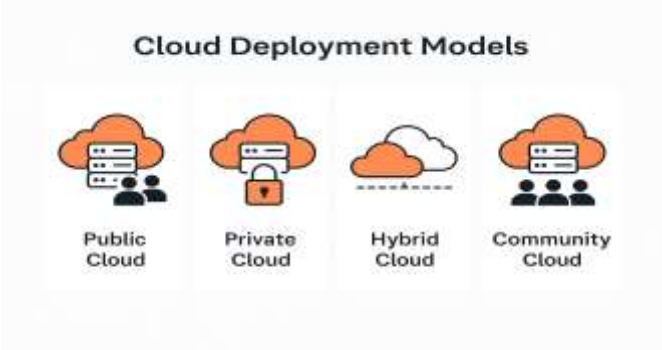
Public Cloud: Think of a public cloud like a gym membership. You don't own the building or the equipment, but you pay to use it whenever you need. Companies like DigitalOcean, Linode, and Alibaba Cloud rent out computing power, storage, and tools over the internet. Anyone can sign up and use them. It's convenient, but you won't have full control over the environment.

Private Cloud: A private cloud is more like having your own personal gym equipment and a personal trainer at home. Only you and people you trust can use it. This model is usually adopted by large

enterprises or institutions that need stricter control over their data—such as banks or hospitals. It can be hosted in-house or managed by a dedicated vendor but is never shared with others.

Hybrid Cloud: Hybrid clouds are like working out both at your private home gym and occasionally at a public fitness center. You use the private setup for sensitive data and core processes, and the public one for everyday tasks. This mix offers more flexibility and helps businesses optimize cost and performance.

Community Cloud: A community cloud is like a shared coworking space that's only for people from the same industry. For instance, multiple universities might pool their resources into a shared cloud platform for research, data analysis, and educational tools. It allows them to collaborate while maintaining common standards and security requirements.



5. Benefits of Serverless Computing

5.1 No Server Management

The most significant advantage is that developers do not need to manage or provision servers. The cloud provider handles all maintenance, patching, and scaling of the infrastructure.

5.2 Automatic Scaling

One of the biggest advantages of serverless computing is that it handles scaling for you. As traffic increases, the system automatically spins up more instances of your function to keep things running smoothly—no need to set up load balancers or manually adjust server capacity during high-traffic times.

5.3 Cost Efficiency

With serverless platforms , you have been charged money only for the actual time your code runs from down to milliseconds to hours. This usage-based pricing means you don't have to pay for idle servers or unused resources, making it a cost-effective option, especially for applications with fluctuating workloads.

5.4 Faster Time-to-Market

With reduced setup and operational overhead, teams can build and deploy features rapidly, making serverless ideal for startups and MVPs (Minimum Viable Products).

6. Real-World Use Cases

6.1 Web and Mobile Backends

Developers use serverless functions to create backend services for web and mobile applications. These include user authentication, payment processing, and data retrieval APIs.

6.2 Data Processing

Serverless functions are ideal for transforming, filtering, and analyzing large volumes of data, such as processing images uploaded to cloud storage or parsing data logs.

6.3 IoT (Internet of Things)

Serverless computing is a great fit for IoT systems, where countless connected devices constantly trans-

mit small bits of data. Serverless functions can be triggered instantly to process, analyze, or store this data as it arrives, enabling real-time responsiveness without the need for always-on servers.

6.4 Scheduled Tasks

Developers often use serverless for cron jobs, such as cleaning databases, sending emails, or generating reports at regular intervals.

7. Limitations and Challenges

7.1 Cold Starts

When a function is not used for a period, its container is shut down. The next invocation requires a "cold start," which introduces latency. Though providers are minimizing this, it's still a concern for real-time applications.

7.2 Vendor Lock-in

Switching from one provider to another can be difficult and a costly approach due to the ownership of serverless ecosystems. This tight coupling can limit flexibility .

7.3 Limited Execution Time

Most providers set a maximum execution time for functions (e.g., AWS Lambda has a 15-minute limit), which makes serverless unsuitable for long-running processes.

7.4 Monitoring and Debugging Complexity

Since serverless apps are distributed and stateless, tracking bugs and performance issues requires specialized tools and logging strategies.

8. Tools and Frameworks

To streamline the process of developing and launching serverless applications, several open-source tools and frameworks have been introduced:

Serverless Framework: Widely recognized for its ease of use, this tool helps deploy serverless applications across various cloud platforms.

AWS SAM (Serverless Application Model): Designed specifically for AWS, this framework simplifies the development and deployment of serverless solutions within the AWS ecosystem.

Google Cloud Functions Emulator: Useful for running and testing Google Cloud Functions locally before pushing them to the cloud.

Netlify Functions and Vercel Functions: These tools are especially popular among front-end developers for integrating serverless functions with JAMstack-based websites.

9. Future Trends and Research Areas

9.1 Multi-cloud and Hybrid Serverless

As organizations aim to reduce lock-in, there is a growing interest in building serverless apps that span multiple cloud environments.

9.2 Edge Computing Integration

Combining serverless with edge computing will allow faster data processing at the source, enabling applications like autonomous vehicles and smart cities.

9.3 Serverless for AI/ML

There is active research in deploying machine learning models using serverless infrastructures. This could drastically reduce the cost of inference for AI workloads.

10. Conclusion

Serverless computing is more than just a trend—it's a paradigm that changes how we think about building software. By offloading infrastructure management to cloud providers, developers gain the freedom to focus on writing clean, functional code that solves real-world problems.

As I prepare to step into the software industry, it's clear to me that understanding serverless architectures is not just beneficial—it's essential. It aligns with the ongoing shift toward microservices, agile development, and continuous delivery, making it a valuable tool in the modern developer's arsenal.

References

1. Amazon Web Services. "What is AWS Lambda?" https://aws.amazon.com/lambda
2. Google Cloud. "Cloud Functions Documentation." https://cloud.google.com/functions
3. Microsoft Azure. "Azure Functions Overview." https://learn.microsoft.com/en-us/azure/azure-functions
4. Roberts, Mike. "Serverless Architectures." MartinFowler.com, 2016. https://martinfowler.com/articles/serverless.html
5. Baldini, Ioana et al. "Serverless Computing: Current Trends and Open Problems." Research Advances in Cloud Computing, 2017.
6. The Serverless Framework. https://www.serverless.com
7. Adzic, Gojko and Chatley, Robert. "Serverless computing: economic and architectural impact." Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ACM, 2017.