Serverless, a cloud-native approach to building APIs

**C768, Task 1**

**Robert J. Robinson**

**01168728**

**Western Governors University**

Contents

[Abstract 3](#_Toc85895513)

[Introduction 4](#_Toc85895514)

[Conclusion 6](#_Toc85895515)

[References 7](#_Toc85895516)

# Abstract

In the current world, technology has improved with modern cloud-first solutions. Most companies have embraced the technology to get to market faster, scale without complications, and offload expensive critical infrastructure. The paper has demonstrated the products provided by Amazon Web Services, such as AWS Lambda. It has explained the serverless computing architecture and the serverless application model with its benefits in empowering modern organizations with cloud-first approaches. A discussion in creating a new API is well articulated with less dependency on infrastructure to attain on-demand scalability that serves our organization. When building our public API, the focus is to make it quickly and scale it on demand. Using a combination of API Gateway and AWS Lambda creates small parts of the API while focusing on quality. The paper shows how the API Gateway allows a Lambda to process and respond to the HTTP request. It explains the Serverless Application Model, SAM, and the accompanying command-line interface tool that provides a simple syntax for event triggers and provisioned resources.

# Introduction

Cloud-first solutions have grown to be what modern companies view as a primary resource to get to market faster, scale with less effort, and offload expensive procurement of critical infrastructure. The most established company in the provision of these cloud services is Amazon Web Services. One of the products they offer is AWS Lambda that permits infinitely scalable compute processes and gives any company the ability to offload processing to these services at a fraction of the cost with minimal training. Lambda is described as an event-driven serverless computing platform that links to several other services by AWS—supported by the Serverless Application Model (SAM), an open-source framework that allows a simple syntax and command-line tool to express functions, APIs, and event source mappings. When used in conjunction with other AWS services, Lambda provides the compute and logic layer of an application. This layer, a micro-service event-driven application, is dispersed in the entire AWS footprint. An example application of the technology is the ability to build a REST API and not bothered with a particular programming language to use, a lengthy deployment time, and limiting over or under capacity planning. With Lambda, you can run code for virtually any type of application or backend service. Lambda runs and scales your code with high availability (Amazon Web Services, 2017).

The AWS Lambda is a serverless computing platform and is sometimes referred to as a Function as a Service (FaaS). Developers who embrace this platform focus more on the product and business logic than responsibilities such as the operating system, access control, operating system patching, scaling, right-sizing, provisioning, and availability in a traditional approach. Going serverless allows developers to write and execute code without configuration management of the underlying servers. AWS handles the entire compute lifecycle and infrastructure in the background, and the developer handles the logic located in the function. While developers may lose some flexibility in various respects, applications/functions are only billed when the code is executed. If the user uses 1 ½ seconds of computer time, that is all of what they had billed. Lambda's best activity or practice is one function that only handles one type of request or an event trigger. The lambda function then accepts a JSON payload and will either respond to the requestor or pass along the payload to another service, and then the function completes. Most operations run in milliseconds, but there is the ability to extend process time to a maximum of five minutes. Another fantastic feature of this service is that this function is spread across many locations allowing for concurrent requests to be handled without worrying about a traditional server overload. Lambda can automatically run code in response to multiple events, such as HTTP requests through Amazon API Gateway, modifications to objects in Amazon S3 buckets, table updates in Amazon DynamoDB, and state transitions in AWS Step Functions (Amazon Web Services, 2021)

As we develop our public API, careful consideration should be made on what technology stack to use. Following the Serverless Application Model, AWS has outlined a set of best practices that allows for a quick and efficient deployment across many deployment stages and an iterative approach. Using the SAM CLI, a simple syntax allows the developer to wire up an entirely distributed application across various services, with minimal configurations.

This CLI can cut deployment and provisioning time to a fraction of a traditional approach and considerably save capital expenses. This CLI tool creates and will also tear down provisioned resources if they are no longer used in the application.

# Conclusion

Building a new API has been discussed with less dependency on infrastructure and attaining on-demand scalability that saves the company money. Using an AWS Lambda model is advantageous and beneficial to any organization. The creation and utility of the serverless applications on AWS relieve developers of some responsibilities and constraints introduced by managing servers. It also relieves the company of some expenditure burdens by minimizing the workforce, perhaps a dedicated system administrator, that maximizes profit. The utility of Lambda as the serverless logic layer allows the organization to expand its growth and focus on its development efforts that differentiate the company from competitors. Alongside Lambda, the AWS offers additional serverless capabilities that can assist in building robust, event-driven, performant, secure, reliable, and cost-effective applications. Understanding the capabilities and suggestions described in the white papers can help ensure success when building serverless applications for an organization. For example, a standard Rest API, traditionally a massive monolith, is slow to change, difficult to scale, and expensive. Selecting this technology stack allows the organization's focus to be able to build quickly without worrying about how it could impact stakeholders or customers. Using API Gateway together with Lambda, small sections of API are created at a time with a strong focus on quality. The SAM CLI tools bring all of the services together and provide the team a simple interface, automate the provisioning and de-provisioning of resources, and speed up the mean time to deployment. These actions have a direct value of saving the company financial recourses that could be put elsewhere.

# References

Amazon Web Services. (2017, November). *Serverless Architectures with AWS Lambda*. https://aws.amazon.com/. Retrieved October 23, 2021, from <https://d1.awsstatic.com/whitepapers/serverless-architectures-with-aws-lambda.pdf>

Amazon Web Services. (2021, February 12). *Security Overview of AWS Lambda*. https://docs.aws.amazon.com. Retrieved October 23, 2021, from https://docs.aws.amazon.com/whitepapers/latest/security-overview-aws-lambda/security-overview-aws-lambda.pdf#document-revisions