JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Cloud-based Messenger Service for customized emails using AWS and Spring Boot Framework

By Radha K C, Soumya A, R V College of Engineering

ABSTRACT

Cloud computing is a technology for delivering ubiquitous, appropriate, on-demand networking access to shared computing resources and services that can be quickly equipped and supplied with minimal maintenance or service provider engagement. Amazon, with its Amazon Web Services (AWS) business, is at the forefront when it comes to providing cloud computing services around the world..

The previous email solutions are entirely adequate when having to send a couple of dozen emails manually but don't work when need to send out thousands of emails over the same time and take more time to send manually and unverified email addresses cause security issues. Serverless microservice architecture is used in the proposed work to implement a cloud-based messenger service using Springboot Framework and Amazon Web Services. AWS Lambda, Elastic Compute Cloud, Simple Queue Service (SQS), Simple Notification Service, Simple Storage Service (S3), Simple Email Service (SES), and cloudwatch are some of the Amazon Web Services used in the implementation to send bulk emails with less time using email templates and configure services to monitor the email. The proposed work is implemented as followed: The vehicle or applications will send a message scheme over SQS for an email to be sent. The templates inside the SES bucket will be transferred through Lambda to the S3 payload bucket. The specific template will be formatted with the message. Then the SQS event will trigger Lambda. The lambda will start fetching messages from SQS and start formatting inside the Lambda, once the format is done it will start publishing to SES. Further, SES will transfer the email to the cloud as well as CloudWatch and SNS to keep track of email sends, deliveries, bounces, and complaints.

Keywords: AWS, Springboot Framework, Serverless microservice, Lambda.

1. INTRODUCTION

In recent years, cloud computing has become a popular and well-established method for administering and offering valuable services over the internet. Serverless computing is a development of cloud-based programming methodologies and a demonstration of the widespread approval of cloud ideas [1]. Massive internet corporations like Amazon, Netflix, and LinkedIn create large, multistage cloud apps that can be built, validated, extended, deployed, operated, and upgraded independently in this era of quick technology breakthroughs. Due to increased server traffic and the requirement to increase server capacity, infrastructure expenditures, in terms of achieving agility and flexibility, are significant barriers for businesses using this strategy [2].

One of the most well-known cloud platform suppliers in the market is Amazon Web Services (AWS). It provides over 200 cloud service providers to build and manage applications. The user need not be concerned about the security, servers, or databases because they are simple to use. With analytics to track each email's impact, AWS's adaptable mail authentication options and IP deployment help to increase deliverability and safeguard sender reputation [3].

The scenario is such that if a message must be sent to many users with the specific formatted template, it will take more time to send manually by previous cloud-based solutions. To effectively use a cloud-

based email service for bulk email communication, email templates are created to store specific content formatted templates, so the emails are sent in less time. AWS services help to promote products and services as customers can receive emails automatically from a cloud day-to-day basis, so they have a better insight into their product performance with customized content and email templates.

Another scenario is that the email service must be triggered automatically based on sudden events rash driving, accidents, fault occurrence, etc found in the vehicle. This should notify the customer immediately of a suitable solution for the problem and send immediate, trigger-based communications from applications to customers, such as fault reports or password resets.

Also, to prevent unauthorized access to send emails from any domain or email address, the AWS service provides email authentication options and flexible deployment choices, including pooled, dedicated, and customer-owned IPs, which aid in influencing sending reputation. Monitor and alert the sender if emails bounce, cause an issue, or are successfully delivered to the recipient's email system [4].

2. LITERATURE SURVEY

Ambika Gupta et.al has presented an architecture using the Elastic Compute Cloud (EC2), AWS Simple Storage Service S3, virtual private cloud (VPC), and Identity and Access Management (IAM) to allow verified clients access to confidential information within the enterprise while also assisting with data recovery [5]. This study is about using a public key, putting data in AWS' Simple Storage Service, and using a private key to retrieve it successfully. Additionally, utilizing multi-factor authentication (MFA), a virtual network was created for users to access the storage. The data is stored in a Virtual Private Cloud to ensure better protection for storage Simple Notification Service is implemented to send emails to the administrator's account regarding the user's log-in details.

Brijesh Choudhary et.al investigated the idea of serverless cloud services and services such as AWS Lambda, as well as other emerging AWS services [6]. In this paper, Serverless Microservice architecture was implemented to create a serverless chat application that enables scaling without adding new servers. Their study provides an analysis to understand the value and significance of 'Function as a Service' and serverless computing for hosting dynamic apps with a lot of user interaction.

The benefits and drawbacks of cloud computing, cloud storage systems, and infrastructure built using web services like Amazon Web Services are discussed in this paper [7]. It is well recognized that AWS is the most successful at offering cloud computing services to individuals, companies, and organizations. It is significantly more efficient and economical than its rivals. Because it enables them to accomplish projects with high computing demands at a reasonable cost, IaaS is a well-liked service among researchers. As a result, all start-ups and growing companies favor AWS as their platform of choice.

Dr. Manish Saraswat et.al described and analyzed the capabilities of Azure, AWS, and Google's cloud computing to assist enterprises and customers are assisted in selecting the most appropriate features which will satisfy their long-term requirements [8]. Their study provides a thorough review of some of the tools offered by AWS, Azure, and Google's cloud computing such as performance, storage space management, and computing.

M. Villamizar et al. present a cost analysis of three distinct techniques for developing and the same flexible scenarios are used when delivering a web application. The three types of architectures used are monolithic, cloud customer-operated microservice, and cloud provider-operated microservice [9]. According to test results, microservices can assist cut infrastructure costs in contrast to conventional monolithic patterns. Additionally, leveraging these services designed expressly for installing and scaling microservices can reduce infrastructure expenses by up to 70%. Then authors discuss the difficulties encountered during the implementation and deployment of microservice applications.

Garrett McGrath et.al designed a new serverless computing architecture that is performance-oriented, written in .NET, deployed in Microsoft Azure, and uses Windows containers as functional execution

environments [10]. They've also provided metrics for evaluating serverless platform execution performance and conducted testing on their prototype, in addition to Azure Functions, AWS Lambda, Google Cloud Functions, and IBM's Apache OpenWhisk deployment. At most concurrent levels, the prototype outperforms competing platforms in terms of throughput.

Niko Mäkitalo et al. have designed a Serverless messenger chatbot application. The chatbot employs serverless microservices built on Amazon Web Services (AWS), which includes DynamoDB, Lambda, API Gateway, CloudWatch, and SNS. This research examines the architecture, which consists of several components calling one another with low latencies [11]. One of the concerns was that each module contributed n milliseconds towards the query time, and the chatbot would become too slow to react to users.

T. Ayodele et al. gave an examination of many functionalities, primarily supplied by cloud storage providers, that could be misused by attackers if the provided email address is not verified, as well as a generic approach to espionage, malware distribution, and incrimination operations [12]. Because of unverified email addresses, security for cloud storage services is a major concern.

T. Hahn et.al examined the limits, vulnerabilities, privacy, and varying regulation of cloud-based emails, as well as how to minimize them in providing users and organizations with efficient and secure email service in the cloud [13]. Intelligent Cloud-Based Email Encryption and Decryption System is a new framework proposed to improve email message security in the cloud. The concept is to encrypt the content of email messages before they are sent from users' mailboxes. This machine-learning system enhances security against re-construction, email eavesdropping, relaying, phishing emails, spoofing, previous messages, and snooping while also providing a high level of confidentiality.

C. Kotas et.al have examined the performance of many HPC benchmarks on the Azure cloud and AWS platforms, with an emphasis on the compute-oriented H16r and c4.8xlarge instance types [14]. These benchmarks measure processing speed, bandwidth, network bandwidth, and memory, among other elements of computer system performance. However, the most cost-effective cloud platform for a given use case is determined by the application's processing and communication habits. According to this study, the AWS c4.8xlarge instance was relatively cheaper for raw computation at the time the tests were done, but Azure's H16r offered cheaper bandwidth.

3. PROPOSED METHODOLOGY

A. System Architecture

The architecture diagram comprises a CICD pipeline that deploys the code and lambda modules for uploading actions and components into the cloud.

The system architecture of the cloud-based messenger service is shown in Figure 1. The main components of the architecture are Publisher, SQS, Lambda, and SES. The publisher can be any application or product. The publisher will send the request to SQS Queue. The SQS Queue withholds the messages in the First in-First Out (FIFO) order. Then SQS will trigger the Lambda for a request. The lambda will fetch the messages needed from SQS, then gets the required attachment from S3 Bucket or payload and picks the required template from the SES template Bucket for the requested message [15]. The lambda will build the whole email and once the formatting is done, the email is sent to SES. Further, SES will authenticate email addresses and transfer the email to the cloud as well as CloudWatch and SNS to keep track of email sends, deliveries, bounces, and complaints [16].

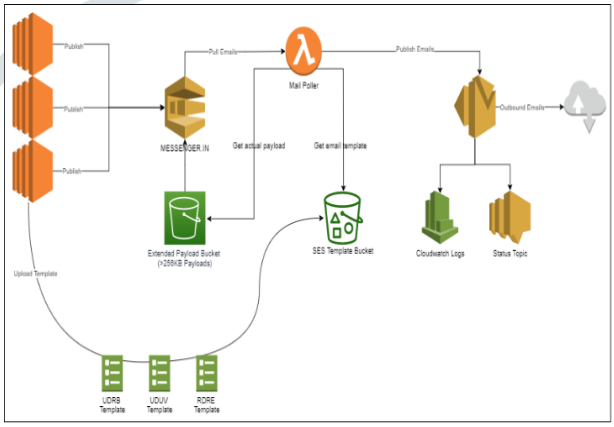

Figure 1 System Architecture

B. MODULE DESCRIPTION

1. Simple Queue Service Module

Amazon SQS is a scalable queuing system that enables web service applications to queue messages generated by one component of the program for consumption by another component. With SQS, any

amount of message transmission, storing, and receiving between software components is possible without message loss. SQS separates application components so that they can run independently, making message management between components easier. Messages in the queue can be stored by any component of a distributed application [17].

Purpose: To store the payload received from the application in FIFO order then trigger AWS Lambda for each request.

Input: The payload (JSON) from the application is sent as a request.

Output: Trigger the Lambda for the request.

Functionality: Validate the payload and trigger the Lambda

Flowchart:

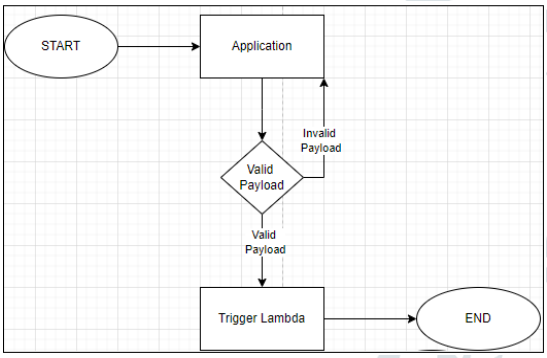

Figure 2 Flowchart for SQS Module

2. Lambda Module

AWS Lambda is serverless, event-driven compute solution that enables the execution of code for virtually any kind of backend service or application without the requirement for server management or provisioning [18]. The Lambda function contains the code to generate a formatted email while picking a template from the SES Template bucket and an attachment from S3 Bucket. Publish the fully formatted mail to SES.

Purpose: To build formatted email by picking a specific template and attachment

Input: The payload (JSON) from the application is sent as a request.

Output: Publish formatted email to SES.

Functionality: The Lambda must read the payload and then the pickup-specific template and attachment. The formatted email must be built and published to SES

Flowchart:

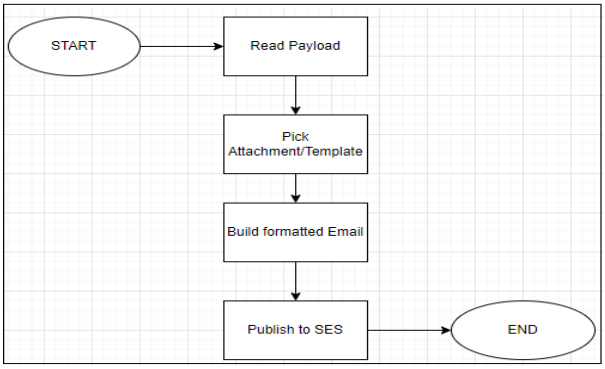

Figure 3 Flowchart for Lambda Module

3. Simple Email Service Module

AWS Simple Email Service (SES) will need some configuration before the emails can be sent for Production workloads [19]. The configurations are required to track the email sends, deliveries, bounces, etc. there will be 2 different configuration sets with the success written to the cloud watch [20] and failures (Bounces, Complaints) published to SNS Topic, for further analysis.

Purpose: To send emails to users from the cloud.

Input: The formatted email published from Lambda.

Output: Trigger the emails to respective users.

Functionality: Validate the sender's and receiver's email addresses and send emails to users. SES triggers the cloudwatch and SNS to monitor the email being sent.

Flowchart:

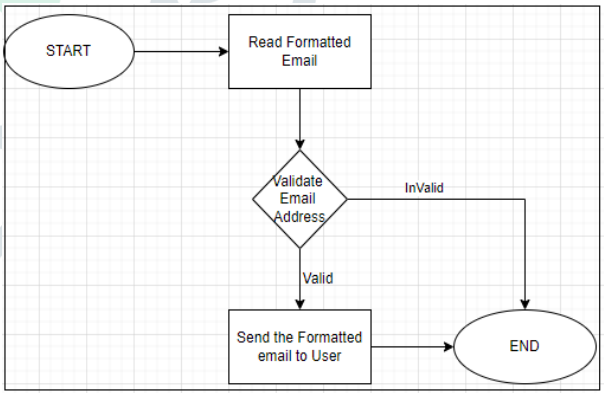

Figure 4 Flowchart for SES Module

C. EXPERIMENTAL RESULTS

1. Templated Email without Attachment

When the payload received does not have any attachments, the email body is formed using the JSON and the email is generated and sent to the recipients. Figure 5 describes the results of Templated Email without attachment.

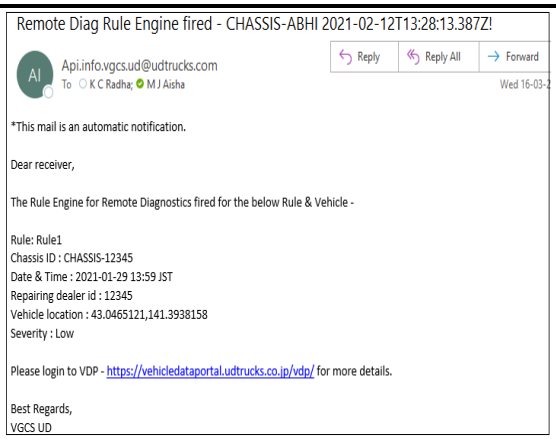

Figure 5 Templated Email without attachment

2. Templated Email with Attachment having a small payload

When the payload is less than 256kb in size, the whole payload is received as part of the JSON. The details required for constructing the email will be present in the JSON received. Figure 6 describes the results of Templated Email with an attachment having a small payload.

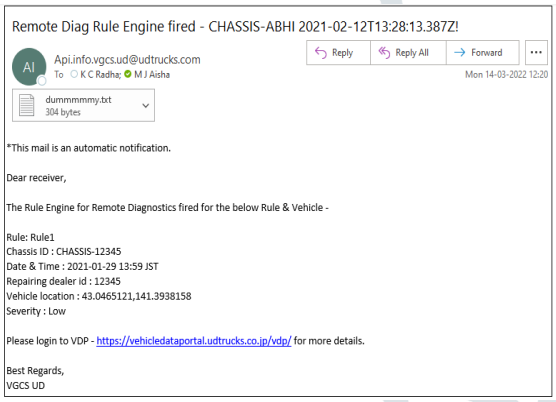

Figure 6 Templated Email with an attachment having a small payload.

3. Templated Email with Attachment having a large payload

When the payload is too large to be sent via SQS, that is, if the size is greater than 256KB, then the "isPayloadLarge" attribute is set to true. The JSON received will contain the details of the S3 bucket where the JSON will be placed. First, the JSON must be fetched from the S3 bucket, and the email has to be generated based on the payload. Figure 7 describes the results of Templated Email with an attachment having a large payload.

Figure 7 Templated Email with an attachment having a large payload

D. CONCLUSION

Modern technology advancements like cloud computing have the potential to significantly impact the entire world. It offers users and companies a number of advantages. One benefit it provides to enterprises is the ability to focus on core competencies while spending less on administration and software upgrades, which cuts operational expenses.

The serverless microservice architecture used to build the cloud-based email service was implemented using the Springboot Framework and Amazon Web Services (AWS) Managed Service. This project implemented a cloud-based messenger service using which the bulk emails will be sent within less time using contexed email templates and configured services to monitor the email sends deliveries, failures, and complaints. Also facilitates sending immediate, trigger-based communications from applications to customers, such as regarding alert messages or password resets. While delivering analytics to track the effects of each email, AWS's adaptable mail authentication options and IP deployment with messenger service managed to boost deliverability and safeguard sender reputation.. This service can be integrated with any applications or products using different languages like Bahasa, Thai, Japanese, and English.

E. REFERENCES

[1] Godson Michael D'silva, Sanket Thakare , Sharddha More, and Jeril Kuriakose," Real-World Smart Chatbot for Customer Care using a Software as a Service (SaaS) Architecture", International conference on I-SMAC (IoT in Social, Mobile, Analytics, and Cloud),2017 IEEE.

[2] Jacob Koch ,Wei Hao"An Empirical Study in Edge Computing Using AWS", IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC),2021.

[3] SaiAkash Neela, Yashwanth Neyyala, VamsiNadh Pendem, Kanishk Peryala, "Cloud Computing Based Learning Web Application Through Amazon Web Services", 7th International Conference on Advanced Computing and Communication Systems (ICACCS), 2021.

[4] M. Sewak and S. Singh," Winning in the Era of Serverless Computing and Function as a Service,", 3rd International Conference for Convergence in Technology (I2CT), Pune, 2018, pp. 1-5.

[5] Ambika Gupta, Anjani Mehta, Lakshya Daver, Priya Banga," Implementation of Storage in Virtual Private Cloud using Simple Storage Service on AWS", 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), 2020.

[6] Brijesh Choudhary, Aditya Gutte, Ankit Dani, Shilpa Sonawani, "Case Study: Use of AWS Lambda for Building a Serverless Chat Application", Proceeding of International Conference on Computational Science and Applications,2020, pp.237-244

[7] A. Alalawi, A. Mohsin and A. Jassim, "A survey for AWS cloud development tools and services," 3rd Smart Cities Symposium (SCS 2020), 2020, pp. 17-23.

[8] Dr. Manish Saraswat1 and Dr. R.C. Tripathi, "Cloud Computing: Comparison and Analysis of Cloud Service Providers—AWS, Microsoft and Google", 9th International Conference on System Modeling & Advancement in Research Trends,2020.

[9] M. Villamizar et al., "Infrastructure Cost Comparison of Running Web Applications in the Cloud Using AWS Lambda and Monolithic and Microservice Architectures,", 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid) 2016.

[10] Garrett McGrath, Paul R.Brenner," Serverless Computing: Design, Implementation and performance", IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW),2017.

[11] Niko Mäkitalo, Tommi Mikkonen,"Case Study: Building a Serverless Messenger Chatbot", Current Trends in Web Engineering,2018, pp.75-86.

[12] T. Ayodele and D. Adeegbe, "Cloud-based emails boundaries and vulnerabilities," 2013 Science and Information Conference, 2013, pp. 912-914.

[13] T. Hahn, T. Kunz, M. Schneider and S. Vowé, "Vulnerabilities through Usability Pitfalls in Cloud Services: Security Problems due to Unverified Email Addresses," 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, 2012, pp. 850-856.

[14] T. Hahn, T. Kunz, M. Schneider and S. Vowé, "Vulnerabilities through Usability Pitfalls in Cloud Services: Security Problems due to Unverified Email Addresses," 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, 2012, pp. 850-856.

[15] Amazon Simple Storage Service. https://aws.amazon.com/s3/. [Online accessed 17-July-2022]

[16] Amazon Simple Notification Service, https://aws.amazon.com/sns/. [Online accessed 17-July-2022]

[17] Amazon Simple Queue Service, https://aws.amazon.com/sqs/ [Online accessed 16-July-2022]

[18] AWS Lambda, https://aws.amazon.com/lambda/, [Online accessed 30-July-2022].

[19] Amazon Simple Email Service. https://aws.amazon.com/ses/ [Online accessed 17-July-2022]

[20] Amazon cloudwatch, https://aws.amazon.com/cloudwatch/, [Online accessed 15-July-2022]

[21] Sewak M, Singh S, "Winning in the era of serverless computing and function as a service", 3rd international conference for convergence in technology (I2CT), Pune, 2018, pp 1–5.

[22] Lynn T, Rosati P, Lejeune A, Emeakaroha V,"A preliminary review of enterprise serverless cloud computing (Function-as-a-service) platforms", IEEE international conference on cloud computing technology Hong Kong, 2017, pp 162–169.

[23] Swedha K, Dubey T, " Analysis of web authentication methods using Amazon web services", 9th international conference on computing, communication and networking technologies (ICCCNT), Bangalore, 2018, pp 1–6.

[24] Lloyd W, Ramesh S, Chinthalapati S, Ly L, Pallickara S," Serverless computing: an investigation of factors influencing microservice performance", IEEE international conference on cloud engineering, Orlando, FL, 2018, pp 159–169

[25] Al-Ali Z et al, "Making serverless computing more serverless", IEEE 11th international conference on cloud computing, San Francisco, CA, 2018, pp 456–459.