CLOUD WEBHOSTING &
PREVENT FROM SERVER CRASH

USING AWS MANAGEMENT CONSOLE

MINI PROJECT REPORT

Submitted by

GOKUL NATH H (312319104035)
DEEPAN BALAJI R(312319104028)

in partial fulfillment for the requirement of award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

St. JOSEPH'S INSTITUTE OF TECHNOLOGY

CHENNAI - 119

ANNA UNIVERSITY: CHENNAI 600 025

JUNE - 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this mini project report "**CLOUD WEBHOSTING using AWS management console"** is the bonafide work of "**GOKUL NATH H (312319104035)DEEPAN BALAJI R(312319104028)"** who carried out the project under my supervision

SIGNATURE

Dr.R.Pugalendhi M.E., Ph.D.,
Head of the Department-
Lab Affairs
Computer Science and Engineering,
St.Joseph's College of Engineering.
OldMamallapuram road,
Chennai - 600119.

SIGNATURE

Dr.J. Ramya M.E., (Ph.D.),
SUPERVISOR Assistant Professor,
Computer Science and Engineering,
St. Joseph's College of Engineering.
Old Mamallapuram road,
Chennai - 600 119.

ACKNOWLEDGEMENT

At the outset, we would like to express our sincere gratitude to the **ALMIGHTY** and our beloved **Chairman**, **Dr. B. Babu Manoharan M.A., M.B.A., Ph.D St. Joseph's Group of Institutions** for his constant guidance and support to the student community and the Society.

We would like to express our hearty thanks to our respected **Managing Director**, **Mrs. S. Jessie Priya M.Com. St. Joseph's Group of Institutions** for her kind encouragement and blessings. We wish to express our sincere thanks to our **Executive Director Mr. B. Shashi Sekar, M.Sc. St. Joseph's Group of Institutions** for providing ample facilities in the institution.

We would like to express sincere gratitude to our beloved and highly respected **Principal Dr. Vaddi Seshagiri Rao M.E., M.B.A., Ph.D., F.I.E**. for his inspirational ideas during the course of the project. We would like to express sincere gratitude and our utmost respect to our beloved **Dr. B. Parvathavarthini M.E., M.B.A., Ph.D., Dean (Research)** for her inspirational ideas during the course of the project.

I also express my sincere thanks and most heartfelt sense of gratitude to **Dr**. **A.Chandrasekar, M.E., Ph.D., Head of the Department of Computer Science** and **Engineering** for his dedication, commendable support and encouragement for the completion of project work with perfection

We would like to acknowledge our gratitude to our supervisor **Dr. J.Ramya M.E., Ph.D** for her excellent guidance and connoisseur's suggestion throughout the study carried out successfully.

Finally, we thank the **Faculty Members** and our **Family**, who helped and encouraged us constantly to complete the project successfully.

CERTIFICATE OF EVALUATION

College Name : St. JOSEPH'S COLLEGE OFENGINEERING

Branch : COMPUTER SCIENCE ANDENGINEERING

Semester VI

Sl.No	Name of the Students	Title of the Project	Name of the Supervisor with designation
1	**GOKUL NATH H (312319104035)**	CLOUD WEBHOSTING Using AWS management console	**Dr.J. Ramya M.E., (Ph.D.) Assistant Professor,**
2	**DEEPAN BALAJI R (312319104028)**		

The reports of the project work submitted by the above students in partial fulfillment for the award of Bachelor of Engineering Degree in **Computer Science and Engineering** of Anna University were evaluated and confirmed to be reports ofthe work done by above students.

Submitted to Project and Viva Examination held on_____.

(INTERNALEXAMINER) **(EXTERNAL EXAMINER)**

ABSTRACT

Traditional on-premises web architectures require complex solutions and accurate reserved capacity forecast in order to ensure reliability. Dense peak traffic periods and wild swings in traffic patterns result in low utilization rates of expensive hardware. This yields high operating costs to maintain idle hardware, and an inefficient use of capital for underused hardware. Amazon Web Services (AWS) provides a reliable, scalable, secure, and highly performing infrastructure for the most demanding web applications. This infrastructure matches IT costs with customer traffic patterns in near-real time. This whitepaper is meant for IT Managers and System Architects who want to understand how to run traditional web architectures in the cloud to achieve elasticity, scalability, and reliability. Highly available and scalable web hosting can be a complex and expensive proposition. Traditional scalable web architectures have not only needed to implement complex solutions to ensure high levels of reliability, but they have also required an accurate forecast of traffic to provide a high level of customer service. Dense peak traffic periods and wild swings in traffic patterns result in low utilization rates of expensive hardware, yielding high operating costs to maintain idle hardware, as well as an inefficient use of capital for underutilized hardware. Amazon Web Services provides a reliable, scalable, secure, and highly performing infrastructure for the most demanding web applications, an infrastructure that matches IT costs with customer traffic patterns in real time. This section helps you evaluate an AWS cloud solution. It compares deploying your web application in the cloud to an on-premises deployment, presents an AWS cloud architecture for hosting your application, and discusses the key components of this solution. An even more dire consequence of the slow provisioning associated with a traditional hosting model is the inability to respond in time to unexpected traffic spikes. There are many stories about web applications going down because of an unexpected spike in traffic after the site is mentioned in the popular media.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Amazon Web Services offers cloud web hosting solutions that provide businesses, non-profits, and governmental organizations with low-cost ways to deliver their websites and web applications. Whether you're looking for a marketing, rich-media, or ecommerce website, AWS offers a wide-range of website hosting options, and we'll help you select the one that is right for you.

OVERVIEW

1.1 An overview of traditional web hosting

Scalable web hosting is a well-known problem space. The following image depicts a traditional web hosting architecture that implements a common three-tier web application model. In this model, the architecture is separated into presentation, application, and persistence layers. Scalability is provided by adding hosts at these layers. The architecture also has built-in performance, failover, and availability features. The traditional web hosting architecture is easily ported to the AWS Cloud with only a few modifications.

However, that infrastructure was a mess, leading the team to conduct a complete overhaul and instead create a large set of uniform APIs to drive development. The result? Services the internal DevOps team and clients alike could access and build on without the need to start from scratch with each project, and scale without the need to build and house a new cluster of servers.

If Amazon's scaling hurdles are reminiscent of your own business's need to grow outside the parameters of your current infrastructure, cloud computing services will be a critical component of your growth.

When Amazon realized its infrastructure solutions put it in an ideal position to host database, compute, and storage services, Amazon AWS was born. Now, the internet giant offers dozens of AWS core services and infrastructure primed to help your business grow.

Of course, AWS is not alone in its provision of cloud compute, cloud storage, cloud database, and cloud infrastructure services–competitors like Microsoft's Azure, Google Cloud Platform, Oracle Cloud, and others emerged soon after. In this model, the architecture is separated into presentation, application, and persistence layers.

1.2 PROBLEM STATEMENT

In AWS web hosting the problem faced is hosting the Apache server in the correct path i.e., in the /var/www/html/ with the service start the load balancer and auto scaling group must be configured with the correct launch template, if the launch template is not correctly configured the web site can't be hosted it will throw an error or the site can't be reached.
Then while creating a launch template the model instance must be created with the Apache server on. in the load balancer the instance state must be in service if it is in out of service the website can't be reached

1.3 EXISTING SYSTEM

Provide building facility teams a holistic view of building status and performance, alerting them to problems sooner and helping them solve problems faster. Provide a detailed record of the efficiency and usage of the building over time.

Use historical building data to help optimize building operations and predict maintenance needs. Offer enriched tenant engagement through services like building control and personalized experiences. Allow building owners to gather granular usage data from multiple buildings so they can react to changing usage patterns in a single platform.

With the introduction of cloud technology and by extension the rapid emergence of Internet of Things (IoT), the barrier to entry for creating smart building solutions has never been lower. These solutions offer commercial real estate customers potential cost savings and the ability to enhance their tenants' experience. You can differentiate your business from competitors by offering new amenities and add new sources of revenue by understanding more about your buildings' operations.

There are several building management systems to consider in commercial buildings, such as air conditioning, fire, elevator, security, and grey/white water. Each system continues to add more features and become more automated, meaning that control mechanisms use all kinds of standards and protocols. This has led to fragmented building systems and inefficiency.

Amazon CloudFront can be used to deliver your website, including dynamic, static and streaming content using a global network of edge locations. Requests for your content are automatically routed to the nearest edge location, so content is delivered with the best possible performance. The barrier to entry for creating smart building solutions has never been lower. These solutions offer commercial real estate customers potential.

1.4 PROPOSED SYSTEM

AWS Cloud console management provides various and different kinds of operations for the execution maintenance of web hosting and managing system. This console gives the elaborate and easy way of creating and hosting the websites with secure method This web hosting uses different consoles like Route 53, Cloud watch, EC2 instance, Load balancer, Auto scaling, S3, RDS.

The website created in the cloud will be highly available and can run without any server down issue The server health is checked and will be maintained in good health. The logs can be stored for every user hitting the website and any malfunction or issue is created it will sort out with the help of logs. We can remotely maintain the server with the prior notification.

Proposed Multi Region Access Point Policy Class Search: Entire Site Articles & Tutorials Documentation - This Product Documentation - This Guide Release Notes Sample Code & Libraries The proposed access control policy for the Multi-Region Access Point. When you update the policy, the update is first listed as the proposed policy. After the update is finished and all Regions have been updated, the proposed policy is listed as the established policy. If both policies have the same version number, the proposed policy is the established policy. Inheritance Hierarchy System.

Proposed Segment Change Class Search: Entire Site Articles & Tutorials Documentation - This Product Documentation - This Guide Release Notes Sample Code & Libraries Describes a proposed segment change. In some cases, the segment change must first be evaluated and accepted. Inheritance Hierarchy System. When FlexMath builds a match, all the matchmaking tickets involved in the proposed match are placed into status REQUIRES_ACCEPTANCE. This Product Documentation - This Guide Release Notes Sample Code & Libraries The proposed access control policy for the Multi-Region Access Point. When you update the policy, the update is first listed as the proposed policy. This web hosting uses different consoles like Route 53, Cloud watch, EC2 instance, Load balancer, Auto scaling, S3.

CHAPTER 2

LITERATURE SURVEY

Mehdi Bahrami et al. [1] The tutorial will begin with an explanation of the concepts behind cloud computing systems, cloud software architecture, the need for mobile cloud computing as an aspect of the app industry to deal with new mobile app design, network apps, app designing tools, and the motivation for migrating apps to cloud computing systems. The tutorial will review facts, goals and common architectures of mobile cloud computing systems, as well as introduce general mobile cloud services for app developers and marketers. This tutorial will highlight some of the major challenges and costs, and the role of mobile cloud computing architecture in the field of app design, as well as how the app-design industry has an opportunity to migrate to cloud computing systems with low investment.

Syed R Rizvi et al. [2] Open Data Cube (ODC) initiative, with support from the Committee on Earth Observation Satellites (CEOS) System Engineering Office (SEO) has developed a state-of-the-art suite of software tools and products to facilitate the analysis of Earth Observation data. This paper presents a short summary and cost analysis of our experience using Amazon Web Services (AWS) to host one such software product, the CEOS Data Cube (CDC) web-based User Interface (UI). In order to provide adaptability, flexibility, scalability, and robustness, we leverage widely-adopted and well-supported technologies such as the Django web framework and the AWS Cloud platform. The UI has empowered users by providing features that assist with streamlining data preparation, data processing, data visualization. This paper presents a short summary and cost analysis of our experience using Amazon Web Services. Mobile cloud computing architecture in the field of app design, as well as how the app-design industry has an opportunity to migrate to cloud

He-Sheng Wu et al. [3] Load balancing is the core of virtual resource management and scheduling in cloud computing. For network applications, the cost of user would be greatly saved if load balancer could dynamically adjust cluster resources in accordance with the current applied load. The current load balancing products of cloud, such as Amazon's ELB, can be used to manage virtual machines in the cloud. The main drawbacks are still only supporting template-based deployment of new virtual machines, not supporting the trend prediction, failing to gain resources dynamically, and not sufficiently providing the elastic management of resources. Since the virtual machine for load balancing management in cloud computing can be dynamically applied and released, an algorithm of prediction-based elastic load balancing resource management (TeraScaler ELB) is presented to overcome the drawbacks. Experiments have shown that the required number of virtual machines change in compliance with the change of network load, thus TeraScaler ELB is able to dynamically adjust.

Nnamdi Ekwe-Ekwe et al. [4] Cloud computing is a ubiquitous part of the computing landscape. For many companies today, moving their computing infrastructure to the cloud reduces time to deployment and saves money. Spot Instances, a subset of Amazon's cloud computing infrastructure (EC2), expands upon this. They allow a user to bid on spare compute capacity in EC2 at heavily discounted prices. If other bids for the spare capacity exceeds the user's own, the user's instance is terminated. In this paper, we conduct one of the first detailed analyses of how location affects the overall cost of deployment of a Spot Instance. We analyze pricing data across all available AWS regions for 60 days for a variety of Spot Instances. We relate the pricing data we find to the overall AWS region and examine any patterns we see across the week. Windows type show that weekday dimension in time hierarchy is as important as hour dimension in describing spot price dynamics. We analyze pricing data across all available AWS regions for 120 days for a variety of Spot Instances.

Khandelwal Veena et al. [5] Time dimension in Amazon EC2 spot instance pricing is decomposed into weekday and hour to study Amazon EC2 spot price dynamics. Statistical tests performed on various compute optimized instances of Linux/UNIX and Windows type show that weekday dimension in time hierarchy is as important as hour dimension in describing spot price dynamics. Coefficient of correlation in spatial trend analysis suggests that there exists a medium to strong correlation between different compute optimized instance types within any given region. Amazon's ELB, can be used to manage virtual machines in the cloud. The main drawbacks are still only supporting template-based deployment of new virtual machines, not supporting the trend prediction, failing to gain resources dynamically. We proposed supersaturation method and developed educational cloud based on supersaturation. In cloud computing, the supersaturation method is a method of assigning a lot of logic resources than the physical resources.

Minoru Uehara [6] Recently, cloud computing becomes popular. Cloud is used for education. We proposed supersaturation method and developed educational cloud based on supersaturation. In cloud computing, the supersaturation method is a method of assigning a lot of logic resources than the physical resources. The cost is more important for supersaturated cloud than the performance. Recently, Linux containers such as LXC have been developed. LXC is suited for supersaturation. In this paper, we build a supersaturated cloud on Amazon EC2 (Elastic Compute Cloud) and evaluate its performance. In order to analyze the drought, the Scaled Drought Condition Index (SDCI) was calculated utilizing concurrent precipitation, temperature and vegetation factors. Unlike conventional research that utilizes only satellite data, the supersaturation method is a method of assigning a lot of logic resources than the physical resources. The cost is more important for supersaturated cloud than the performance. Recently, Linux containers such as LXC have been developed. LXC is suited for supersaturation.

Ravi Kishore Kodali et al [7] Internet of Things (IoT) is a technology which is rapidly growing, the future of IoT is limitless as the data streams have quadrupled over the years. Future markets are going to shift from traditional data processing techniques to Big Data Analytics and Cloud computing as businesses worldwide are shifting to cloud-based work approaches which was largely boosted by the Covid-19 pandemic. Fishes are well known to be highly sensitive to the environment, hence they require proper care and attention from their owners. Many a times they forget to feed the fish, change the water, check the pH levels etc. these problems look simple but when it comes to tracking hundreds of fishes it can be difficult, these issues can be easily resolved by using IoT. An IoT system that can be highly beneficial for small to large scale aquariums is necessary e.g., Dubai Aquarium (where hundreds of fishes are constantly monitored).

Fabio Palumbo [8]. With the growing adoption of cloud infrastructures to deliver a variety of IT services, monitoring cloud network performance has become crucial. However, cloud providers only disclose qualitative info about network performance, at most. This hinders efficient cloud adoption, resulting in no performance guarantees, uncertainties about the behavior of hosted services, and sub-optimal deployment choices. In this work, we focus on cloud-to-user latency, i.e., the latency of network paths interconnecting datacenters to worldwide-spread cloud users accessing their services. In detail, we performed a 14-day measurement campaign from 25 vantage points deployed via Planet lab infrastructure (emulating spatially- spread users) and considering services running in distinct locations on the infrastructures of the two most popular public-cloud providers, namely Amazon Web Services and Microsoft Azure. . In this paper, we build a supersaturated cloud on Amazon EC2 (Elastic Compute Cloud)**,** network performance, at most. This hinders efficient cloud adoption, resulting in no performance, follows DevOps best practices and relies on Jenkins for the Continuous Integration stage. Two more convolutional layers and one fully connected layer, connected to a SoftMax

Artur Cepuc et al. [9] Nowadays, cloud computing has become the go to solution for most enterprises. This has led to the introduction of DevOps techniques in which developers work closely with network engineers in order to ensure fast and reliable deployment of their applications. This paper presents an entire automated pipeline, starting with detecting changes in the Java-based web application source code, creating new resources in the Kubernetes cluster to host this new version and finally deploying the containerized application in AWS. The solution follows DevOps best practices and relies on Jenkins for the Continuous Integration stage. Two more convolutional layers and one fully connected layer, connected to a SoftMax output layer, complete the network. Dropout has been applied tothe fully connected layer and all layers contain units of ReLu.

Beaulah A. Navamani et al. [10] Cloud computing offers the potential for productivity, cost savings and innovation advantages to organizations. To utilize these benefits, to facilitate wide-scale cloud adoption and to embrace cloud computing, we must address many clouds security challenges. Attackers can compromise the vulnerable hosts and can either take over their resources or use them as stepping stones for future attacks. Open ports can be the most prominent gateway for many threats and potential attacks. Security communities, reports and surveys often state that port scans can be considered as precursors to an attack. In this paper, we performed an analysis pinpointing the risks of open ports in a cloud environment. To embrace cloud computing, we must address many clouds security challenges. Attackers can compromise the vulnerable hosts and can either take over their resources we must address many clouds security challenges. Depending upon the test result we evaluate the essential parameters like precision, recall, accuracy and false positive rates. These parameters evaluate the efficiency of our model. Once we design our model, we test our model using two features in AWS Machine learning. One, using real time prediction where we give real time input data and test our model. Two, we do batch prediction, where we have a set of customer data and we upload our data to evaluate our prediction.

Yongmin Kim Soo et al. [11] This study analyzed the ongoing drought situation in South Korea using MODIS satellite data and AWS ground observation data. In order to analyze the drought, the Scaled Drought Condition Index (SDCI) was calculated utilizing concurrent precipitation, temperature and vegetation factors. Unlike conventional research that utilizes only satellite data, we utilized AWS ground-based observation data that has advantages in terms of data accuracy and collection stability. Precipitation and temperature inputs were sourced from AWS ground observation data while vegetation coverage was obtained from MODIS NDVI. Hosted cloud-based video surveillance service. Even though this technology is complex and widely used, some security experts have pointed out that some of its vulnerabilities can be exploited

Nicholas Okita et al [12] Cloud computing platforms offer a wide variety of computational resources with different performance specifications for different prices. In this work, we experiment how Spot instances and Availability Zones on the Amazon Web Services (AWS) could be utilized to reduce the processing cost. Not only that, but we propose an instance management algorithm in AWS to minimize the execution cost of programs implemented using the programming model Scalable Partially Idempotent Task System (SPITS). Our results show that the proposed method is able to identify and dynamically adjust the virtual machine types that offer the best cost per performance ratio. The main contributions of our paper are: (1) we provided a holistic view of the security model of cloud-based video surveillance summarizing the underlying threats, vulnerabilities and mitigation techniques

Geetha Vinayagam et al. [13] In the last few years, Cloud computing technology has benefited many organizations that have embraced it as a basis for revamping the IT infrastructure. Cloud computing utilizes Internet capabilities in order to use other computing resources. Amazon Web Services (AWS) is one of the most widely used cloud providers that leverages the endless computing capabilities that the cloud technology has to offer. AWS is continuously evolving to offer a variety of services,

including but not limited to, infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service. Among the other important services offered by AWS is Video Surveillance as a Service (VSaaS) that is a hosted cloud-based video surveillance service. Even though this technology is complex and widely used, some security experts have pointed out that some of its vulnerabilities can be exploited in launching attacks aimed at cloud technologies. In this paper, we present a holistic security analysis of cloud-based video surveillance systems by examining the vulnerabilities, threats, and attacks that these technologies are susceptible to. We illustrate our findings by implementing several of these attacks on a test bed representing an AWS-based video surveillance system.

A. Allawi et al. [14] Cloud computing is currently considered one of the fastest-growing businesses in the field of IT. One of cloud computing main benefits is that it can be accessed and managed anytime and from anywhere using the web. In order to allow for such a feature, the cloud service providers offer several development tools and services. This paper shows some of the services offered by AWS, one of the biggest cloud service providers if not the biggest. Services including cloud build, cloud deployment, cloud9, code pipeline and code commit. The paper as well presents a comparison between those features and some other features that are offered by the other competitors. Moreover, it presents the pros and cons, pricing and users' feedback.

Ranjith Ramesh et al. [15] The aim of the project is to develop a Machine Learning model to perform predictive analytics on the banking dataset. The banking data set consists of details about customers like and whether the customer will buy a product provided by the bank or not. The data set is obtained from University of California Irvine Machine Learning Repository. This data set is used to create a binary classification model using Amazon Web Service (AWS) Machine Learning platform. 70 % of the data is used to train the binary classification model and 30 % of the dataset is used to test the model.

CHAPTER 3 SYSTEM DESIGN

In this chapter, the various UML diagrams for Cloud Webhosting using AWS is represented and the various functionalities are explained.

3.1 UNIFIED MODELLING LANGUAGE

Unified Modeling language (UML) is a standardized modeling language enabling developers to specify, visualize, construct and document artifacts of a software system. Thus, UML makes these artifacts scalable, secure and robust in execution. It uses graphic notation to create visual models of software systems. UML is designed to enable users to develop an expressive, ready touse visual modeling language. In addition, it supports high-level development concepts such as frameworks, patterns and collaborations. Some of the UML diagrams are discussed.

3.1.1 USE CASE DIAGRAM OF CLOUD WEBHOSTING USING AWS

Use case diagrams are considered for high level requirement analysis of asystem. So when the requirements of a system are analysed the functionalities are captured in use cases. So, it can be said that uses cases are nothing but the system functionalities written in an organized manner. Now the second things which are relevant to the use cases are the actors. Actors can be defined as something that interacts with the system. The actors can be human user, some internal applications or may be some external applications. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

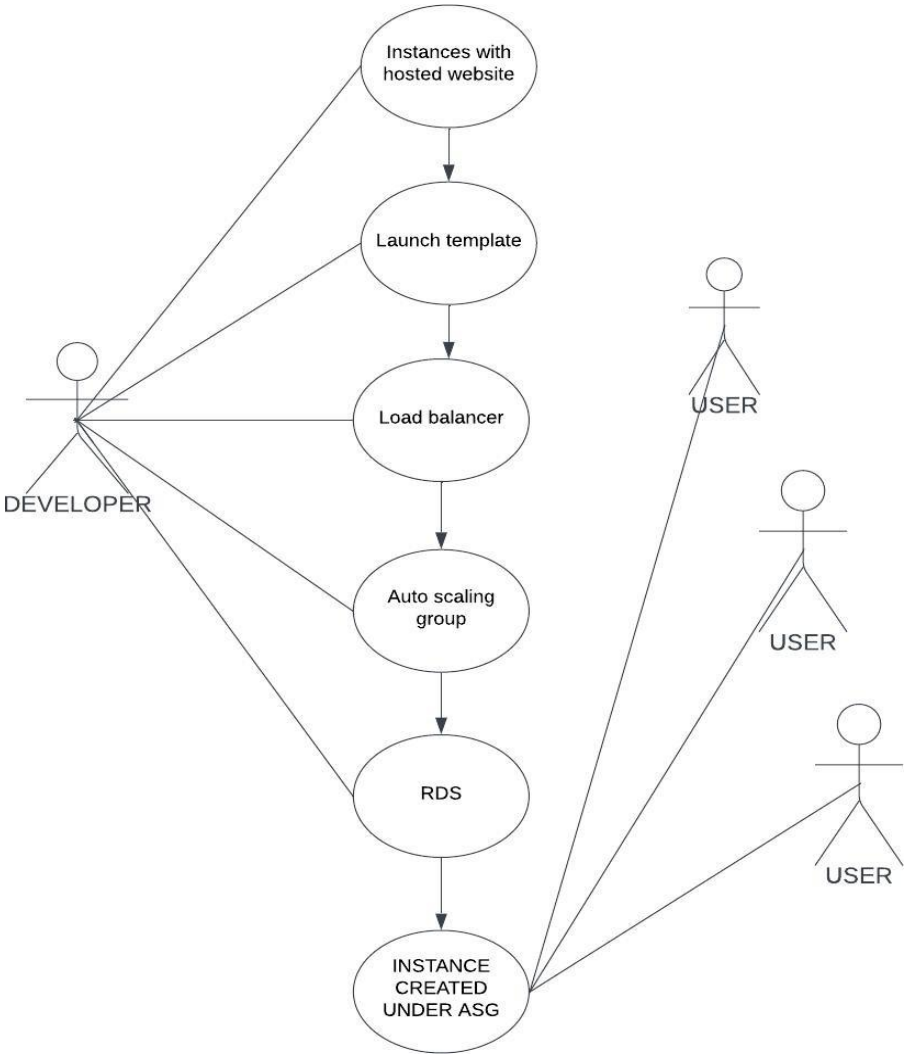


Figure 3.1 Use case diagram of Cloud

Web Hosting AWS

The Functionalities are to be represented as a use case in the representation. Each and every use case is a function in which the user or the server can have the access on it. The names of the use cases are given insuch a way that the functionalities are preformed, because the main purpose of the functionalities is to identify the requirements. To add some extra notes that should be clarified to the user, the notes kind of structure is added to the usecase diagram. The use case diagram as shown in Figure 3.1 provides details based on the Emotion Recognition System.

3.1.2 SEQUENCE DIAGRAM OF BCX MODEL FOR CLOUD WEBHOSTING USING AWS

Sequence diagrams model the flow of logic within thesystem in a visual manner, enabling to both document and validate the logic,and are commonly used for both analysis and design purposes.

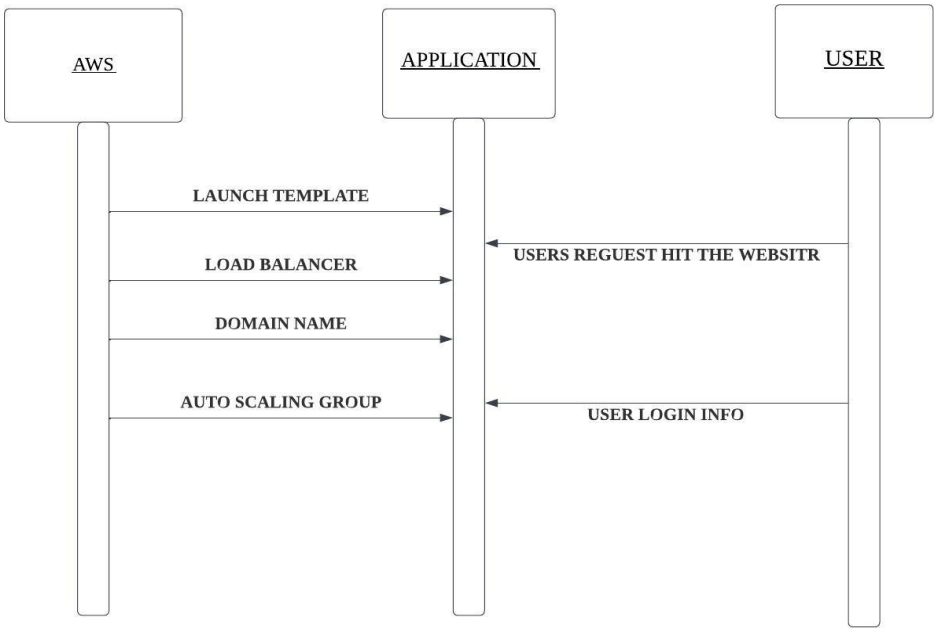


Figure 3.2 Sequence diagram ofCloud

Web Hosting AWS

The various actions that take place in the application in the correct sequence are shown in Figure 3.2 Sequence diagrams are the most popular UML for dynamic modeling.

3.1.3 ACTIVITY DIAGRAM OF CLOUD WEBHOSTING USING AWS

Activity is a particular operation of the system. Activity diagram is suitable for modeling the activity flow of the system.

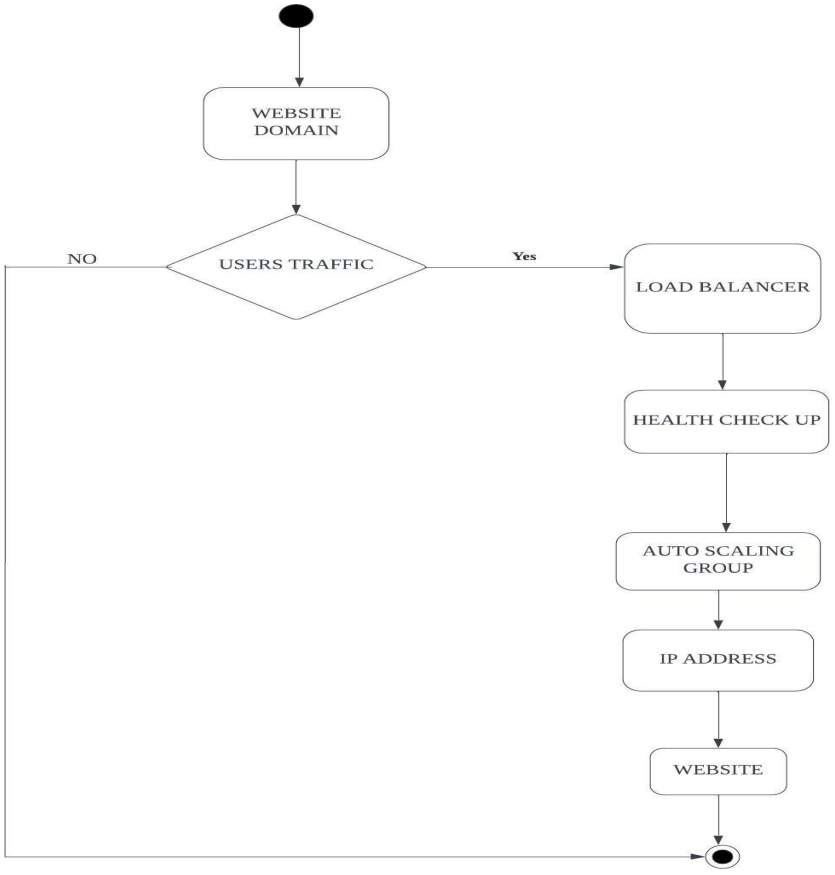


Figure 3.3 Activity diagram of Cloud Web Hosting AWS

Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part.

An application can have multiple systems. Activity diagram also captures these systems and describes the flow from one system to another. This specific usage is not available in other diagrams. These systems can be database, external queues, or any other system.

It does not show any message flow from one activity to another. Activity diagram is sometime considered as the flow chart. Although the diagrams look like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single. The Figure 3.3 shows the activity diagram of the developed application.

3.1.3 COLLABORATION DIAGRAM OF CLOUD WEBHOSTING USING AWS

The next interaction diagram is collaboration diagram. It shows the object organization. Here in collaboration diagram the method call sequence is indicated by some numbering technique. The number indicates how the methods are called one after another. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization. The various objects involved and their collaboration is shown in Figure 3.4.

Now to choose between these two diagrams the main emphasis is given on the type of requirement.

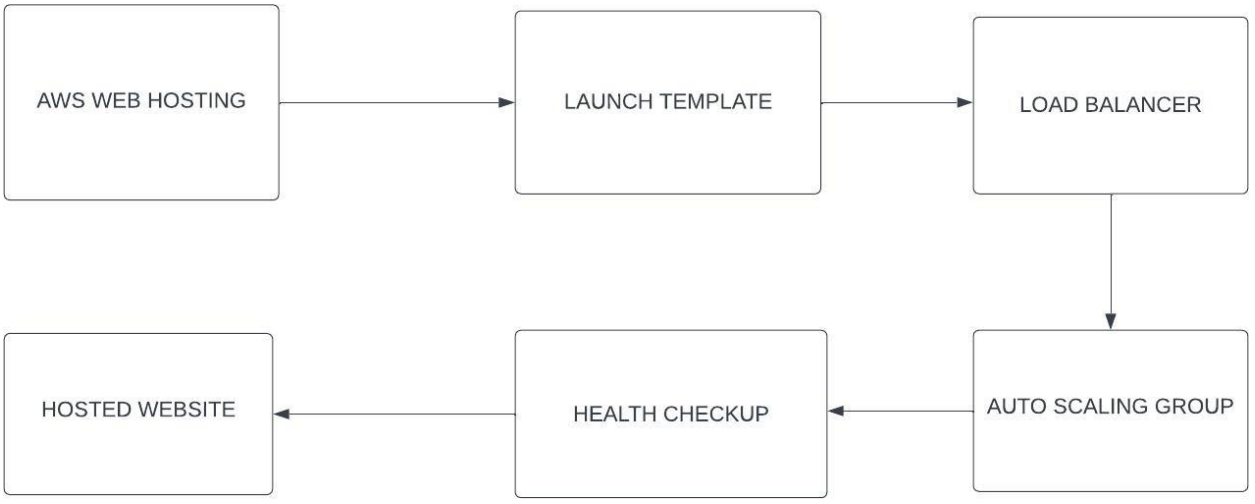


Figure 3.4 Collaboration diagram of Cloud WebHostingAWS

3.1.6 COMPONENT DIAGRAM OF CLOUD WEBHOSTING USING AWS

A component diagram displays the structural relationship of components of a software system. These are mostly used when working with complex systems that have many components such as sensor nodes, cluster head and base station. It does not describe the functionality of the system but it describes the components used to make those functionalities.

Components communicate with each other using interfaces. The interfaces are linked using connectors. The Figure 3.5 shows a component diagram.

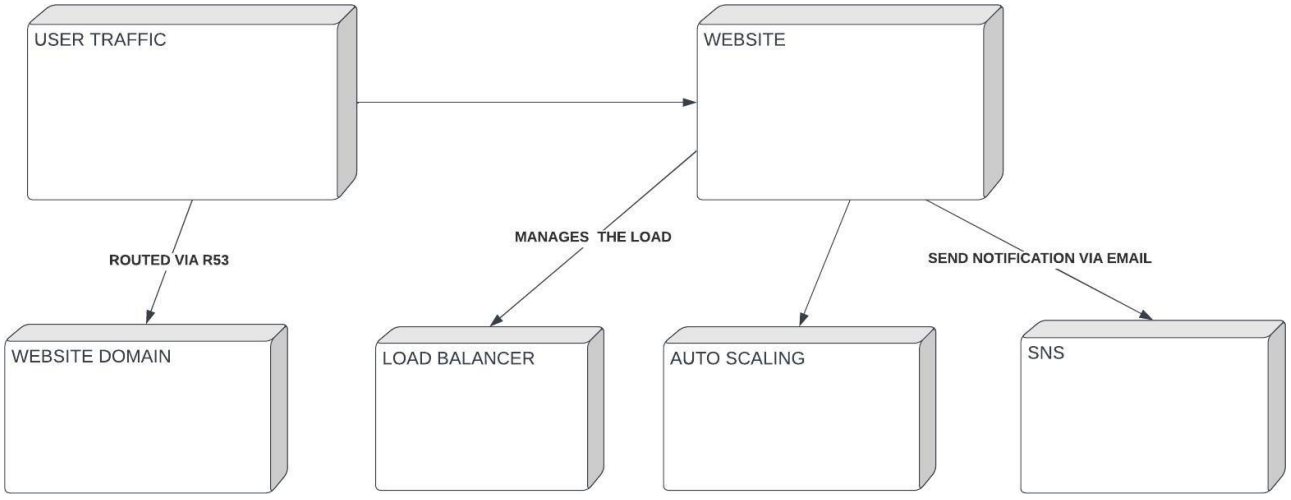


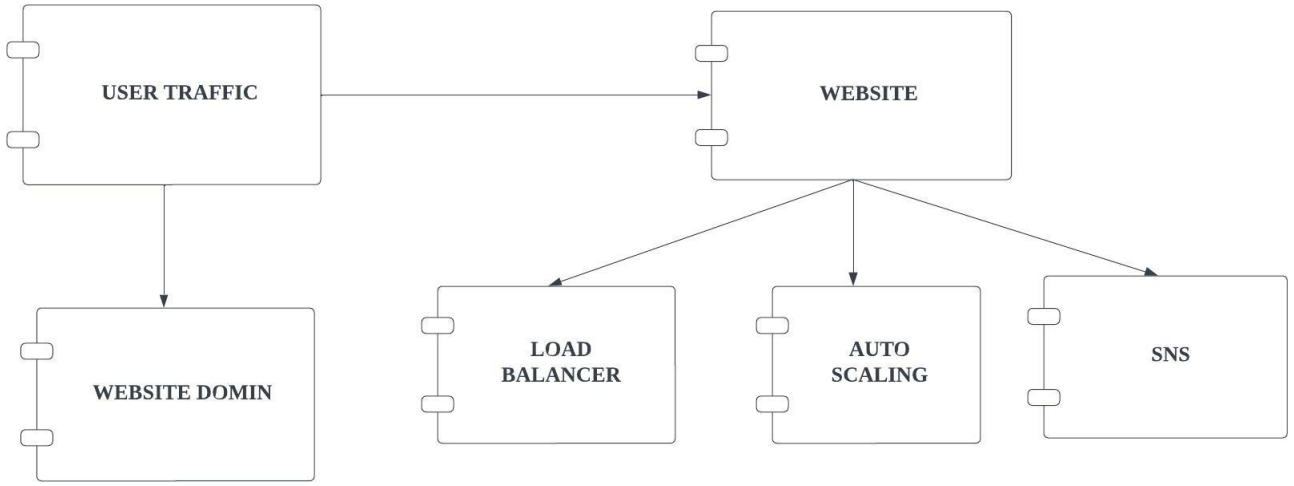
Figure 3.6 Component diagram of Cloud Web Hosting AWS

3.1.6 DEPLOYMENT DIAGRAM OF BCX MODEL FOR CLOUD WEBHOSTING USING AWS

A deployment diagrams shows the hardware of your system and the software in that hardware. Deployment diagrams are useful when your software solution is deployed across multiple machines such as sensor nodes, cluster head and base station with each having a unique configuration.

Deployment Diagram in the figure 3.6 shows how the modules gets deployed in the system.

Figure 3.7 Deployment diagram of Cloud Web HostingAWS



Deployment Diagram in the figure 3.6 shows how the modules getsdeployed in the system.

CHAPTER 4

SYSTEM ARCHITECTURE

In this chapter, the System Architecture for Cloud Web Hosting usingAWS is represented and the modules are explained.

4.1 ARCHITECTURE DESCRIPTION

In system architecture the detailed description about the system modules and the working of each module is discussed as shown in figure 4.1.

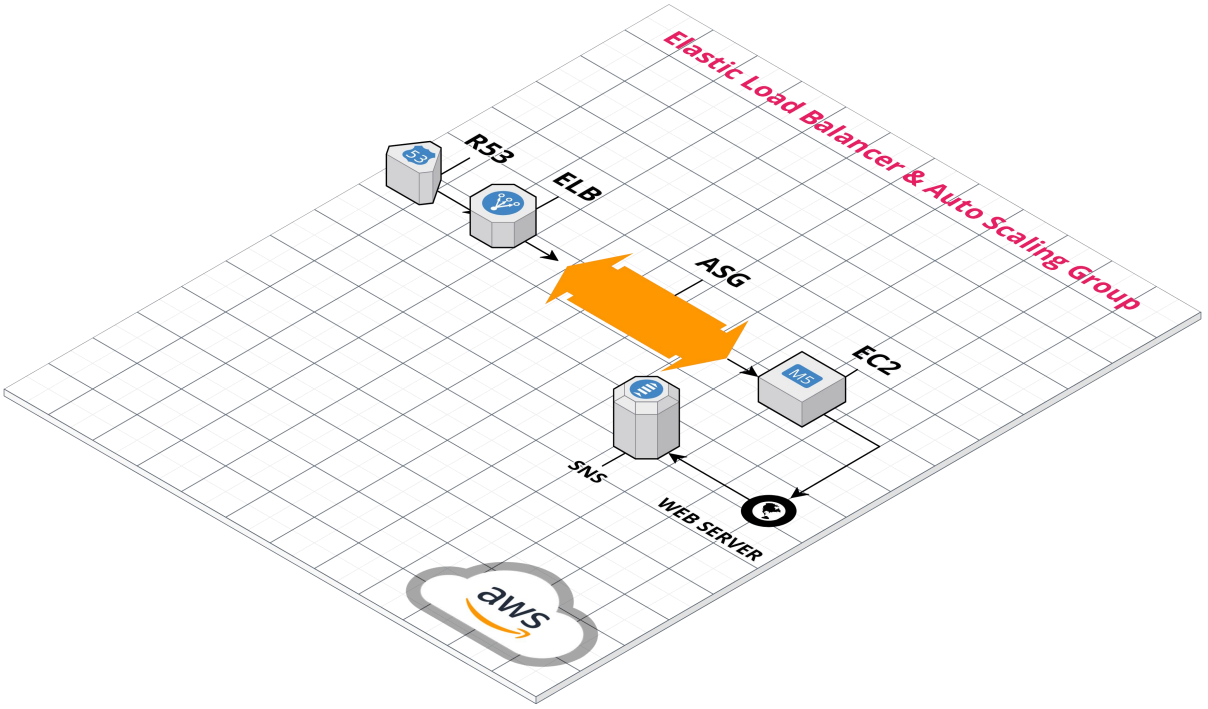


Figure 4.1 System Architecture of Cloud Web Hosting AWS

Amazon Web Services (AWS) can be used to change the way you design multi-tier architectures and implement popular patterns such as microservices, mobile backends, and single-page applications. Architects and developers can use Amazon API Gateway, AWS Lambda, and other services to reduce the development and operations cycles required to create and manage multi-tiered applications.

The multi-tier application (three-tier, *n*-tier, and so forth.) has been a cornerstone architecture pattern for decades, and remains a popular pattern for user-facing applications. Although the language used to describe a multi-tier architecture varies, a multi-tier application generally consists of the following components: A convolution tool that separates and identifies the various features of the image for analysis in a process called as Feature Extraction.

4.2 Architectural Description

WEBAPPLICATION HOSTING:

Highly available and scalable web hosting can be complex and expensive. Dense peak periods and wild swings in traffic patterns result in low utilization of expensive hardware. Amazon Web Services provides the reliable, scalable, secure, and high-performance infrastructure required for web applications while enabling an elastic, scale-out and scale-down infrastructure to match IT costs in real time as customer traffic fluctuates.

4.2.1 Amazon Elastic Compute Cloud (EC2)

Amazon Elastic Compute Cloud (EC2) is a part of Amazon.com's cloud-computing platform, Amazon Web Services (AWS), that allows users to rent virtual computers on which to run their own computer applications. EC2 encourages scalable deployment of applications by providing a web service through which a user can boot an Amazon Machine Image (AMI) to configure a virtual machine, which Amazon calls an "instance", Convolutional Layer

4.2.2 Amazon Route 53 (Route 53)

Amazon Route 53 (Route 53) is a scalable and highly available Domain Name System (DNS) service. Released on December 5, 2010, it is part of Amazon.com's cloud computing platform, Amazon Web Services (AWS). The name is a possible reference to U.S. Routes, and "53" is a reference to the TCP/UDP port 53, where DNS server requests are addressed. In addition to being able to route users to various AWS services, including EC2 instances, Route 53 also enables AWS customers to route users to non-AWS infrastructure and to monitor the health of their application and its endpoints.

4.2.3 Elastic Load Balancing (ELB)

Elastic Load Balancing (ELB) is a load-balancing service for Amazon Web Services (AWS)deployments ELB automatically incoming application traffic and scales resources to meet traffic demands' helps an IT team adjust capacity according to incoming application and network traffic. Users enable ELB within a single availability zone or across multiple availability zones to maintain consistent application performance is a load-balancing service for Amazon Web Services AWS deployments ELB automatically

4.2.4 Launch Templates

Launch Templates is a new capability that enables a new way to templatize your launch requests. Launch Templates streamline and simplify the launch process for Auto Scaling, Spot Fleet, Spot, and On-Demand instances. Launch Templates reduce the number of steps required to create an instance by capturing all launch parameters within one resource. This makes the process easy to reproduce. Also, through support for Auto Scaling, Spot Fleet, Spot and On-Demand instances, Launch Templates make it easier to implement standards and best practices.

4.2.5 Auto Scaling group (ASG)

An Auto Scaling group contains a collection of Amazon EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An Auto Scaling group also enables you to use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies. Both maintaining the number of instances in an Auto Scaling group and automatic scaling are the core functionality of the Amazon EC2 Auto Scaling service Amazon EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An Auto Scaling group. An Auto Scaling group also enables you to use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies.

4.2.6 Amazon Simple Notification Service (Amazon SNS)

Amazon Simple Notification Service (Amazon SNS) is a fully managed messaging service for both application-to-application (A2A) and application-to-person (A2P) communication. The A2A pub/sub functionality provides topics for high-throughput, push-based, many-to-many messaging between distributed systems, microservices, and event-driven serverless applications. Using Amazon SNS topics, your publisher systems can fanout messages to a large number of subscriber systems, including Amazon SQS queues, AWS Lambda functions, HTTPS endpoints, and Amazon Kinesis Data Firehose, for parallel processing. The A2P functionality enables you to send messages to users at scale via SMS, mobile push, and email.

CHAPTER 5

SYSTEM IMPLEMENTATION

In this chapter, the System Implementation for the Cloud webhosting using AWS

5.1 SYSTEM DESCRIPTION

Amazon Web Services, Inc. (AWS) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. These cloud computing web services provide distributed computing processing capacity and software tools via AWS server farms. One of these services is Amazon Elastic Compute Cloud (EC2), which allows users to have at their disposal a virtual cluster of computers, available all the time, through the Internet. AWS's virtual computers emulate most of the attributes of a real computer, including hardware central processing units (CPUs) and graphics processing units (GPUs) for processing; local/RAM memory; hard-disk/SSD storage;

Amazon markets AWS to subscribers as a way of obtaining large-scale computing capacity more quickly and cheaply than building an actual physical server farm.[10] All services are billed based on usage

a choice of operating systems; networking; and pre-loaded application software such as web servers, databases, and customer relationship management (CRM).

AWS services are delivered to customers via a network of AWS server farms located throughout the world. Fees are based on a combination of usage (known as a "Pay-as-you-go" model), hardware, operating system, software, or networking features chosen by the subscriber required availability, redundancy, security, and service options. Subscribers can pay for a single virtual AWS computer, a dedicated physical computer, or clusters of either. Amazon provides select portions of security for subscribers (e.g. physical security of the data centers) while other aspects of security are the responsibility of the subscriber (e.g. account management, vulnerability scanning, patching). AWS operates from many global geographical regions including 6 in North America.

Amazon markets AWS to subscribers as a way of obtaining large-scale computing capacity more quickly and cheaply than building an actual physical server farm.[10] All services are billed based on usage, but each service measures usage in varying ways. As of 2021 Q4, AWS has 33% market share for cloud infrastructure while the next two competitors Microsoft Azure and Google Cloud have 21%, and 10% respectively, according to Synergy Group.

5.2 DATASET FOR CLOUD WEBHOSTING USING AWS

Weighing in at a whopping 500 GB (388 GB of data and 112 GB of free space to allow for some in-place decompression), the Wikipedia XML data is our newest Public Data Set.

This data set contains all of the Wikimedia wikis in the form of wikitext source and metadata embedded in XML. We'll be updating this data set every month and we'll keep the sets for the previous three months around.

As you can see from this screen shot of my PuTTY window, there are some pretty beefy files in this data set:

```
[root@ip-10-245-114-17 wikipedia]# ls -lSh | head -10
total 388G
-rw-r--r-- 1 root root   65G Aug 15 06:46 dewiki-20090728-pages-meta-history.xml.bz2
-rw-r--r-- 1 root root   29G Aug 15 09:10 frwiki-20090810-pages-meta-history.xml.bz2
-rw-r--r-- 1 root root   16G Aug 15 07:55 eswiki-20090812-pages-meta-history.xml.bz2
-rw-r--r-- 1 root root   14G Aug 15 05:40 jawiki-20090810-pages-meta-history.xml.bz2
-rw-r--r-- 1 root root   14G Aug 15 05:20 itwiki-20090808-pages-meta-history.xml.bz2
-rw-r--r-- 1 root root   13G Aug 15 06:54 ruwiki-20090807-pages-meta-history.xml.bz2
-rw-r--r-- 1 root root   11G Aug 15 07:16 enwiki-20090810-stub-meta-history.xml.gz
-rw-r--r-- 1 root root  9.6G Aug 15 05:06 enwiki-20090810-pages-meta-current.xml.bz2
-rw-r--r-- 1 root root  8.5G Aug 15 03:59 nlwiki-20090812-pages-meta-history.xml.bz2
[root@ip-10-245-114-17 wikipedia]#
```

As an example of what can be done with this data, take a look at Cloudera's blog post on Grouping Related Trends with Hadoop and Hive. This article shows how to create a trend tracking site using a Cloudera Hadoop cluster running on EC2, using Apache Hive queries to process the data.

In the Encyclopedic category, we have added access to the DBpedia Knowledge Base, the Freebase Data Dump, and the Wikipedia Extraction, or WEX.The DBpedia Knowledge Base currently describes more than 2.6 million things including 213,000 people, 328,000 places, 57,000 music albums, 36,000 films, and 20,000 companies. There are 274 million RDF triples in the 67 GB data set. The 66 GB Freebase Data Dump is an open database of the world's information, covering millions of topics in

5.3 PREPROCESSING

Amazon SageMaker enables developers and data scientists to build, train, tune, and deploy machine learning (ML) models at scale. You can deploy trained ML models for real-time or batch predictions on unseen data, a process known as inference. However, in most cases, the raw input data must be preprocessed and can't be used directly for making predictions. This is because most ML models expect the data in a predefined format, so the raw data needs to be first cleaned and formatted in order for the ML model to process the data.

Amazon SageMaker built-in Scikit-learn library for preprocessing input data and then use the Amazon SageMaker built-in Linear Learner algorithm for predictions. We'll deploy both the library and the algorithm on the same endpoint using the Amazon SageMaker Inference Pipelines feature so you can pass raw input data directly to

Amazon SageMaker. We'll also show how you can make your ML workflow modular and reuse the preprocessing code between training and inference to reduce development overhead and errors.

To accomplish this, we'll first do some simple preprocessing with the Amazon SageMaker built in Scikit-learn library. We will use the Simple Imputer , StandardAero, and OneHotEncoder transformers on the raw abalone data; these are commonly-used data transformers included in Scikit-learns preprocessing library that process the data into a format required by ML models. Then, we'll use our processed data to train the Amazon SageMaker Linear Learner algorithm to predict the age of abalones. Finally, we'll create a pipeline that combines the data processing and model prediction steps using Amazon SageMaker Inference Pipelines.

5.4 TESTING

Device Farm is an application testing service that lets you test and interact with applications on many real devices at once. You can also reproduce issues on a real device in real time. You can use Device Farm to test any type of application—including native iOS and Android, JavaScript (React Native, Ionic, Cordova), Xamarin, and other applications.

With Device Farm, you can view videos, screenshots, logs, and performance data to pinpoint and fix issues. This helps you improve quality before shipping your app. Device Farm lets you test your application on a shared fleet of over 2,500 real devices, or on your own private device fleet in the cloud. The Serverless Architecture is an integration of separate, distributed services, which must be tested both independently, and together. The Serverless Architecture is dependent on internet/cloud services, which are hard to emulate locally.

Device Farm recently added support for Node.js. This means you can run Appium, Detox, and any JavaScript testing framework. The AWS API and Command Line Interface tools support test automation. These tools allow developers to automatically create and configure test environments, connect databases, run scripts and even use continuous integration methods to automatically run a test suite on each successful build.

CHAPTER 6

CODING AND SCREENSHOTS

The following code has been implemented in using LINUX with EC2 root user
The code used is bash and Linux commands

6.1 SAMPLE CODING FOR EMOTION DETECTION SYSTEM

```
 login as: ec2-user
Authenticating with public key "imported-openssh-key"

       __|___|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
2 package(s) needed for security, out of 6 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-11-196 ~]$ sudo su -
[root@ip-172-31-11-196 ~]# yum install httpd
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
| 3.7 kB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.53-1.amzn2 will be installed
--> Processing Dependency: httpd-tools = 2.4.53-1.amzn2 for package: httpd-2.4.53-
1.amzn2.x86_64
--> Processing Dependency: httpd-filesystem = 2.4.53-1.amzn2 for package: httpd-
2.4.53-1.amzn2.x86_64
--> Processing Dependency: system-logos-httpd for package: httpd-2.4.53-
1.amzn2.x86_64
--> Processing Dependency: mod_http2 for package: httpd-2.4.53-1.amzn2.x86_64
--> Processing Dependency: httpd-filesystem for package: httpd-2.4.53-1.amzn2.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.4.53-1.amzn2.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.4.53-
1.amzn2.x86_64
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.4.53-
1.amzn2.x86_64
--> Running transaction check
---> Package apr.x86_64 0:1.7.0-9.amzn2 will be installed
---> Package apr-util.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
--> Processing Dependency: apr-util-bdb(x86-64) = 1.6.1-5.amzn2.0.2 for package: apr-
```

```
util-1.6.1-5.amzn2.0.2.x86_64
---> Package generic-logos-httpd.noarch 0:18.0.0-4.amzn2 will be installed
---> Package httpd-filesystem.noarch 0:2.4.53-1.amzn2 will be installed
---> Package httpd-tools.x86_64 0:2.4.53-1.amzn2 will be installed
---> Package mailcap.noarch 0:2.1.41-2.amzn2 will be installed
---> Package mod_http2.x86_64 0:1.15.19-1.amzn2.0.1 will be installed
--> Running transaction check
---> Package apr-util-bdb.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
================================================================================
================================================================================
 Package                    Arch                 Version
Repository                 Size
================================================================================
================================================================================
================================================================================
Installing:
 httpd                      x86_64               2.4.53-1.amzn2
amzn2-core                 1.3 M
Installing for dependencies:
 apr                        x86_64               1.7.0-9.amzn2
amzn2-core                 122 k
 apr-util                   x86_64               1.6.1-5.amzn2.0.2
amzn2-core                 99 k
 apr-util-bdb               x86_64               1.6.1-5.amzn2.0.2
amzn2-core                 19 k
 generic-logos-httpd        noarch               18.0.0-4.amzn2
amzn2-core                 19 k
 httpd-filesystem           noarch               2.4.53-1.amzn2
amzn2-core                 24 k
 httpd-tools                x86_64               2.4.53-1.amzn2
amzn2-core                 88 k
 mailcap                    noarch               2.1.41-2.amzn2
amzn2-core                 31 k
 mod_http2                  x86_64               1.15.19-1.amzn2.0.1
amzn2-core                 149 k

Transaction Summary
================================================================================
================================================================================
================================================================================
```

```
Install  1 Package (+8 Dependent packages)

Total download size: 1.9 M
Installed size: 5.2 M
Is this ok [y/d/N]: yes
Downloading packages:
(1/9): apr-1.7.0-9.amzn2.x86_64.rpm
| 122 kB  00:00:00
(2/9): apr-util-1.6.1-5.amzn2.0.2.x86_64.rpm
|  99 kB  00:00:00
(3/9): apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64.rpm
|  19 kB  00:00:00
(4/9): generic-logos-httpd-18.0.0-4.amzn2.noarch.rpm
|  19 kB  00:00:00
(5/9): httpd-filesystem-2.4.53-1.amzn2.noarch.rpm
|  24 kB  00:00:00
(6/9): httpd-2.4.53-1.amzn2.x86_64.rpm
| 1.3 MB  00:00:00
(7/9): httpd-tools-2.4.53-1.amzn2.x86_64.rpm
|  88 kB  00:00:00
(8/9): mailcap-2.1.41-2.amzn2.noarch.rpm
|  31 kB  00:00:00
(9/9): mod_http2-1.15.19-1.amzn2.0.1.x86_64.rpm
| 149 kB  00:00:00
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Total
9.6 MB/s | 1.9 MB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : apr-1.7.0-9.amzn2.x86_64
1/9
  Installing : apr-util-bdb-1.6.1-5.amzn2.0.2.x86_64
2/9
  Installing : apr-util-1.6.1-5.amzn2.0.2.x86_64
3/9
  Installing : httpd-tools-2.4.53-1.amzn2.x86_64
4/9
  Installing : generic-logos-httpd-18.0.0-4.amzn2.noarch
5/9
  Installing : mailcap-2.1.41-2.amzn2.noarch
6/9
  Installing : httpd-filesystem-2.4.53-1.amzn2.noarch
```

```
7/9
  Installing : mod_http2-1.15.19-1.amzn2.0.1.x86_64
8/9
  Installing : httpd-2.4.53-1.amzn2.x86_64
9/9
  Verifying : apr-util-1.6.1-5.amzn2.0.2.x86_64
1/9
  Verifying : apr-util-bdb-1.6.1-
5.amzn2.0.2.x86_64
2/9
  Verifying : mod_http2-1.15.19-1.amzn2.0.1.x86_64
3/9
  Verifying : httpd-filesystem-2.4.53-1.amzn2.noarch
4/9
  Verifying : httpd-tools-2.4.53-1.amzn2.x86_64
5/9
  Verifying : mailcap-2.1.41-2.amzn2.noarch
6/9
  Verifying : generic-logos-httpd-18.0.0-4.amzn2.noarch
7/9
  Verifying : httpd-2.4.53-1.amzn2.x86_64
8/9
  Verifying : apr-1.7.0-9.amzn2.x86_64
9/9

Installed:
  httpd.x86_64 0:2.4.53-1.amzn2

Dependency Installed:
  apr.x86_64 0:1.7.0-9.amzn2                  apr-util.x86_64 0:1.6.1-5.amzn2.0.2       apr-
util-bdb.x86_64 0:1.6.1-5.amzn2.0.2       generic-logos-httpd.noarch 0:18.0.0-4.amzn2
httpd-filesystem.noarch 0:2.4.53-1.amzn2       httpd-tools.x86_64 0:2.4.53-1.amzn2
mailcap.noarch 0:2.1.41-2.amzn2             mod_http2.x86_64 0:1.15.19-1.amzn2.0.1

Complete!
[root@ip-172-31-11-196 ~]# systemctl start httpd
[root@ip-172-31-11-196 ~]# systemctl enable httpd
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to
/usr/lib/systemd/system/httpd.service.
[root@ip-172-31-11-196 ~]# cd /var/www/html/
[root@ip-172-31-11-196 html]# chmod -Rf 777
[root@ip-172-31-11-196 html]# /var/www/html
-bash: /var/www/html: Is a directory
[root@ip-172-31-6-199 ~]# ls
[root@ip-172-31-6-199 ~]# ls /var/www/html/
```

90's games escape room gamers bout index.html index.txt JEC marvel paper Poster quiz

6.2 SAMPLE SCREENSHOTS FOR CLOUD WEBHOSTING USING AWS.

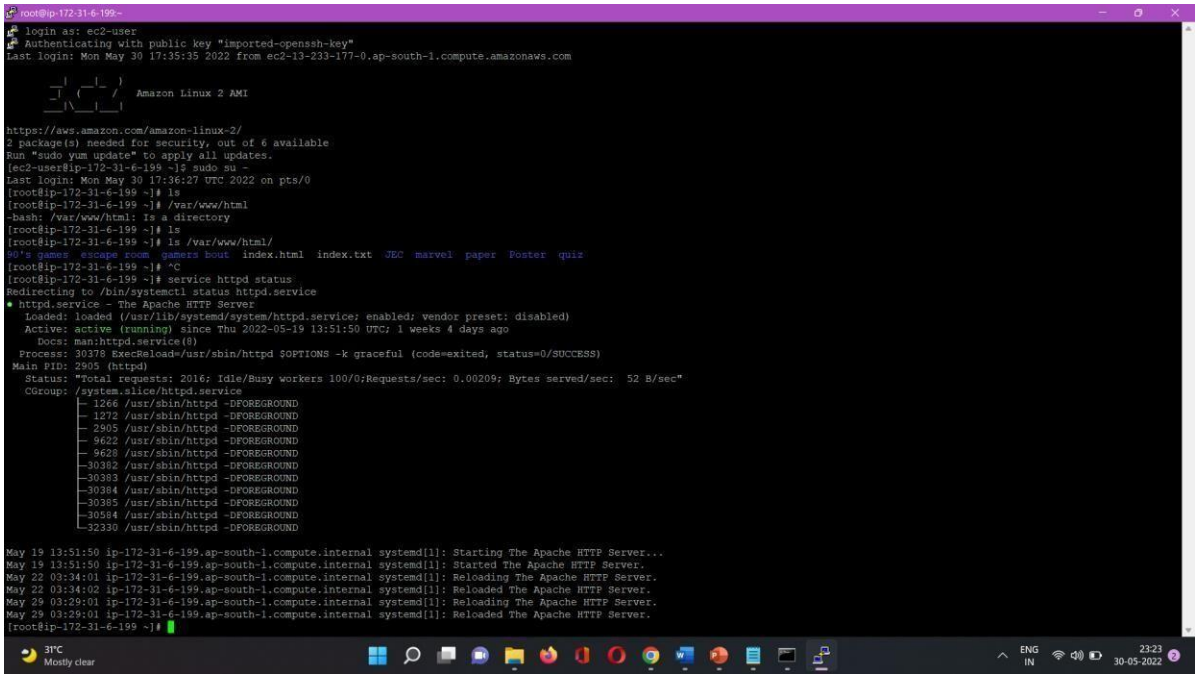
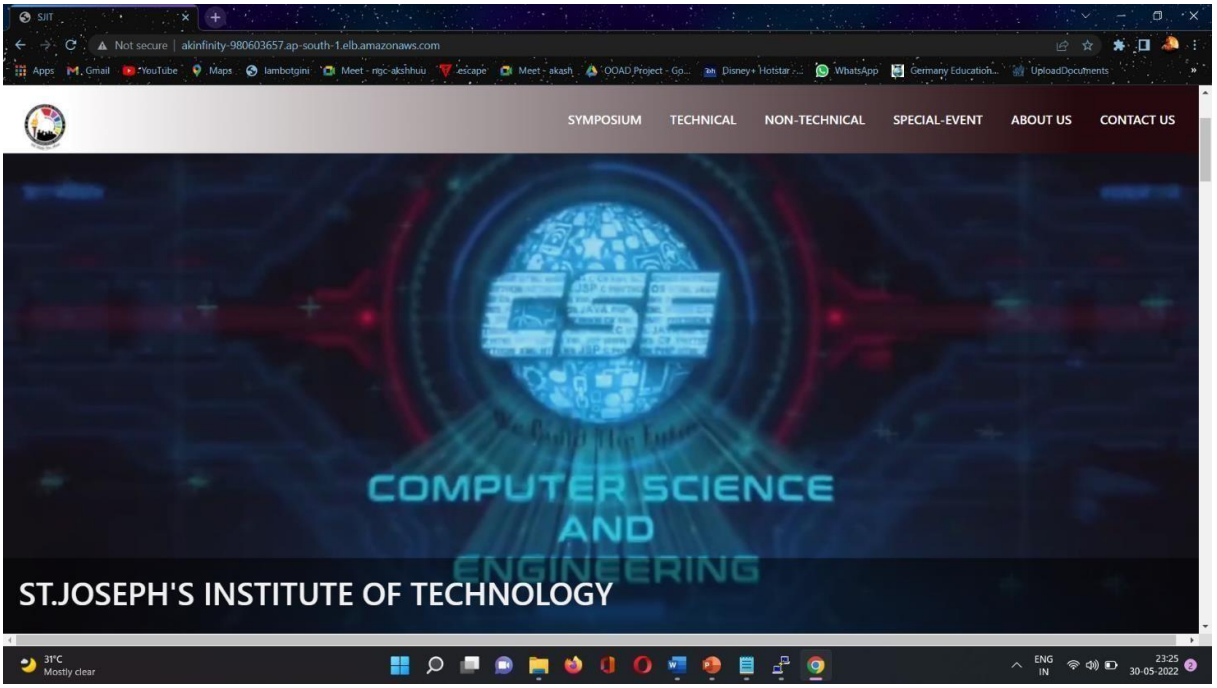
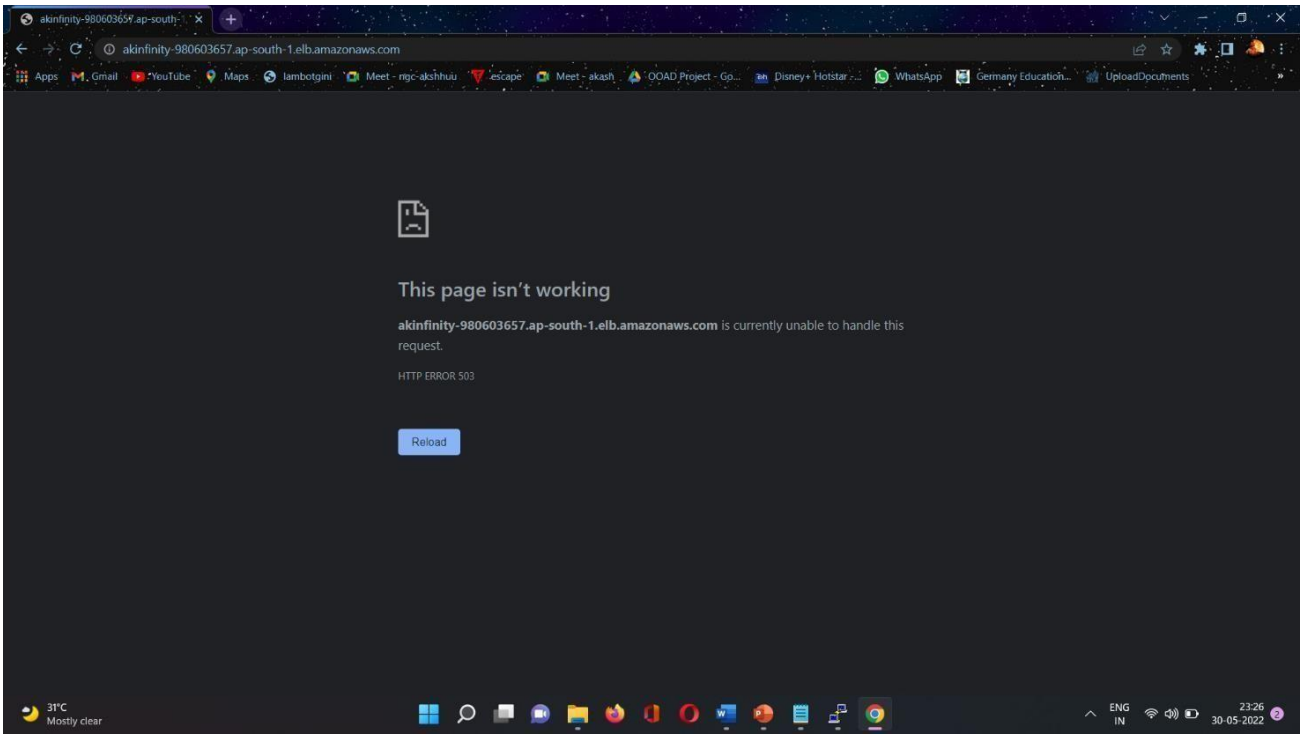


Figure 6.1 Sample coding screenshot

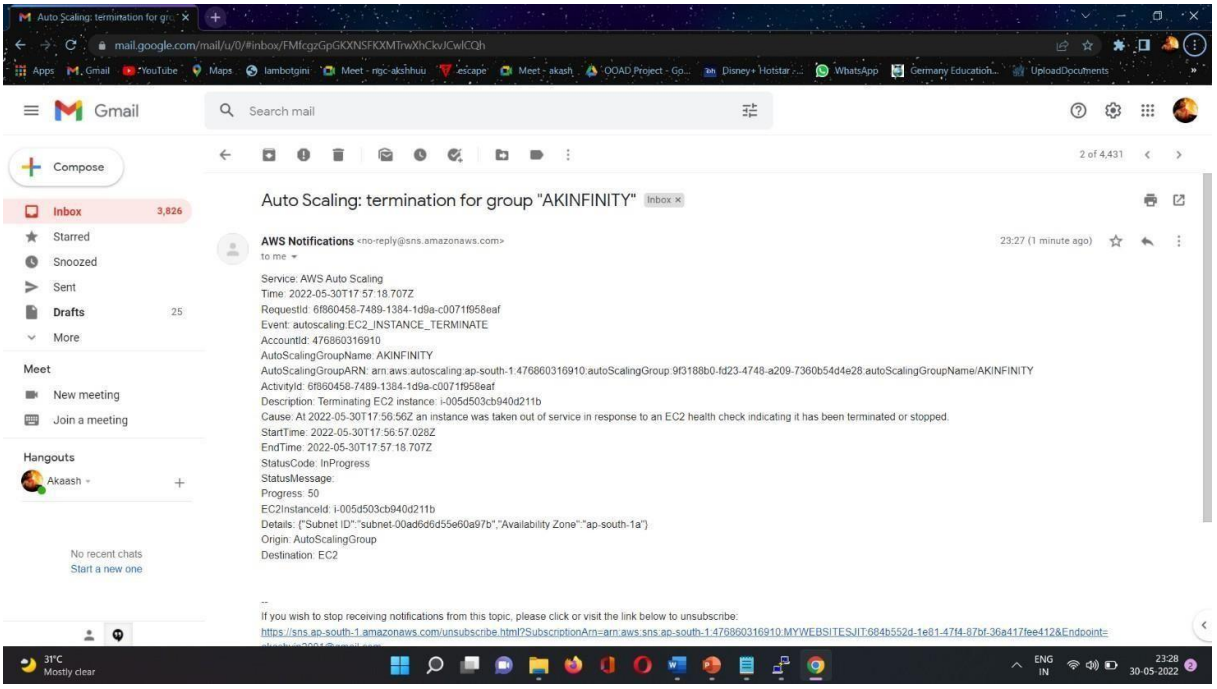
SAMPLE OUTPUT



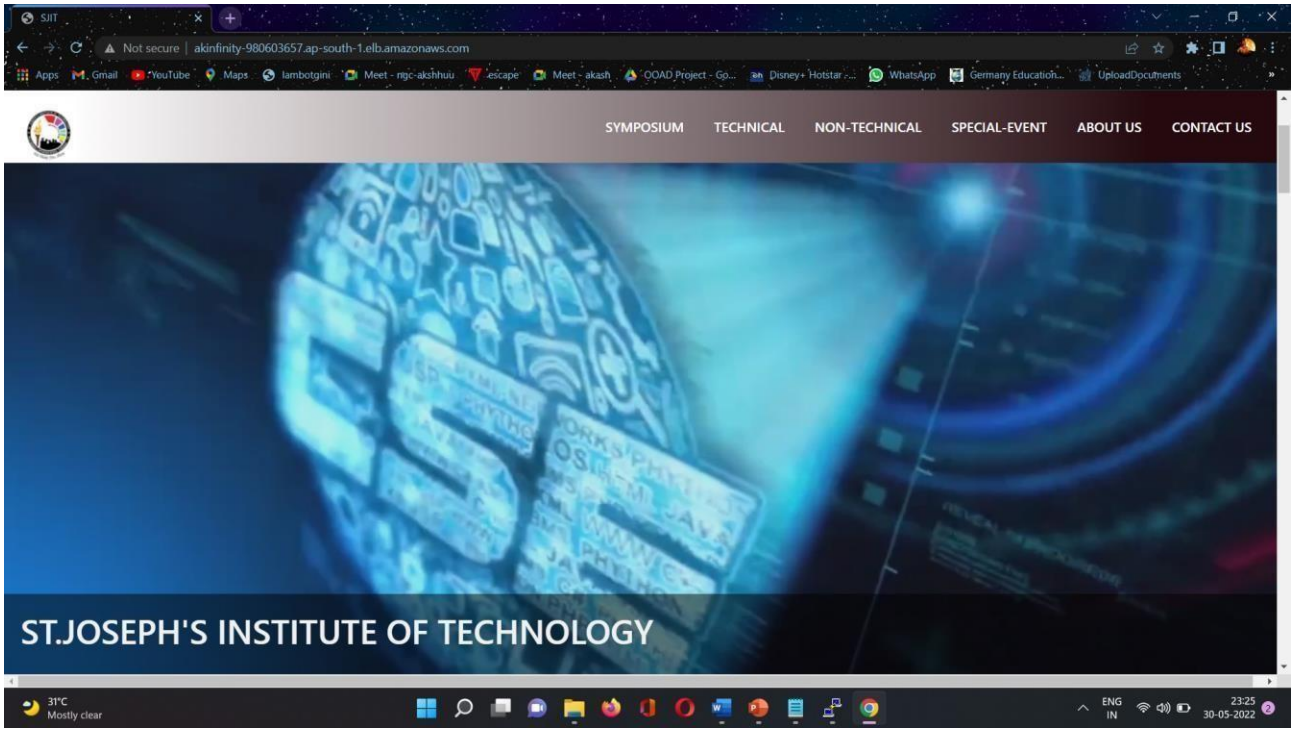
6.2 Website is hosted in the specified domain.



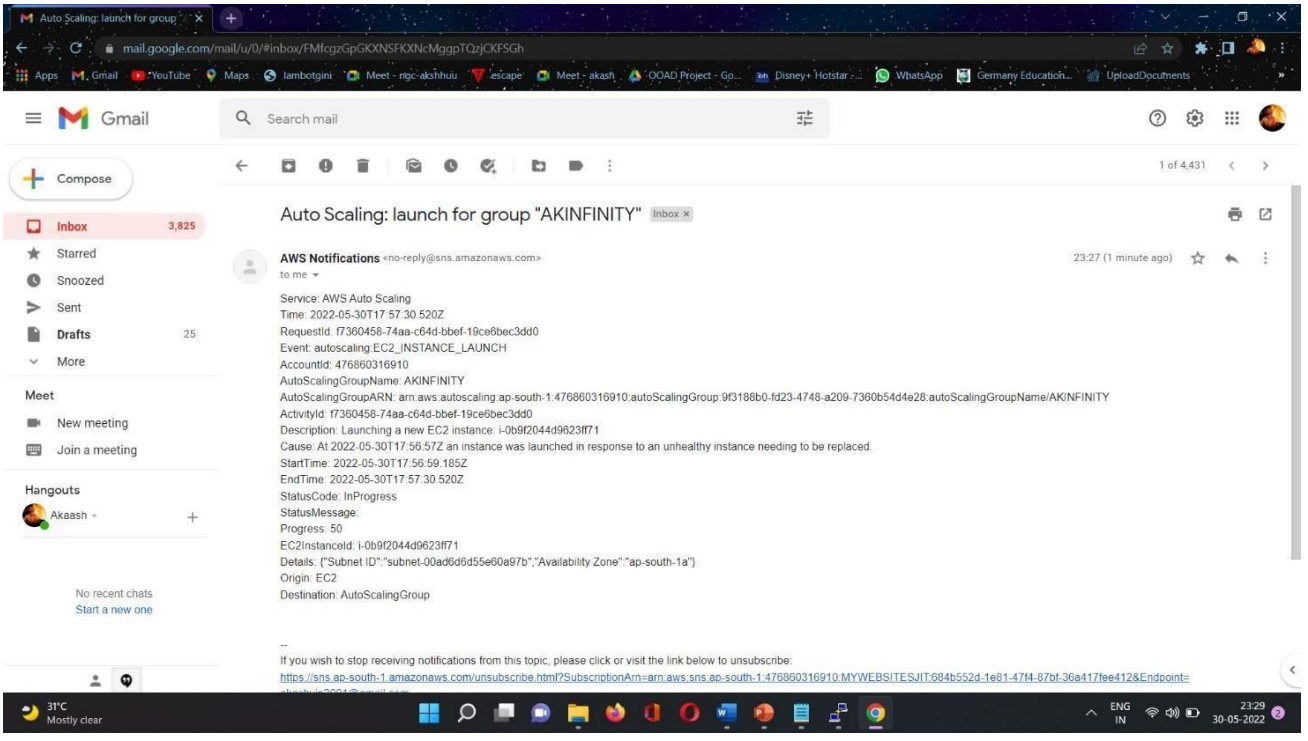
6.3 Server is crashed because for high traffic, Now the health checkup will trigger the auto scaling group to create another machine.



6.3 The SNS pops the indication that instance under ASG is terminated.



6.4. Auto scaling group triggered the instance a new machine with the hosted website.



6.5 SNS notification of newly launched instance under the autoscaling group

CHAPTER 7

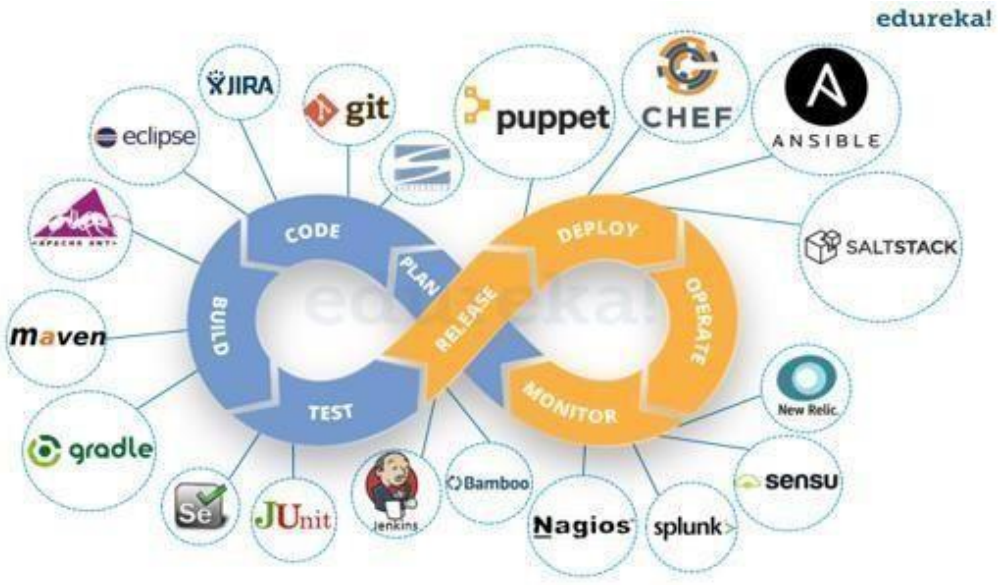
CONCLUSION AND FUTURE WORK

7.0 CONCLUSION

There are numerous architectural and conceptual considerations when you are contemplating migrating your web application to the AWS Cloud. The benefits of having a cost-effective, highly scalable, and fault-tolerant infrastructure that grows with your business far outstrips the efforts of migrating to the AWS Cloud.

7.1 FUTURE WORK

The main enhancement can be done in this structure is AUTOMATION, the creation of server and hosting which is done in manually cane automated by triggering it.
The automation can be done by different tools called git, Jenkins, docor, terraform by using this we can automate the full process. This automation is called DEVOPS. DevOps defines an agile relationship between operations and Development. It is a process that is practiced by the development team and operational engineers together from beginning to the final stage of the product.



REFERENCES

1. Mehdi Bahrami, "Round Robin Algorithm for the Prediction of AWS". Proceeding of International Conference on Systems Compu-tation Automation and Networking, 2019.

2. Syed R Rizvi, Brian Killough, Andrew Cherry, Sanjay Gowda "Classification of Load balancer in auto scaling group waveform analysis: logistic regression modelling," aws Research International, vol. 2020, Article ID 3764653, 6 pages, 2020.

3. Y. K. Qawqzeh, A. S. Bajahzar, M. Jemmali, M. M. Otoom, and A. Thaljaoui, "Classification of launch template in the create image: logistic regression modeling," cloud Research International, vol. 2020, Article ID 3764653, 6 pages, 2020.

4. S. Gupta, H. K. Verma, and D. Bhardwaj, "Classification of ASG using launch configuration and support vector machine as a technique," computer science Management and Systems Engineering, Springer, Singapore, 2021.

5. D. K. Choubey, M. Kumar, V. Shukla, S. Tripathi, and V. K. Dhandhania, "Comparative analysis of classification methods with algorithmic web application hosting on cloud computing,vol. 16, no. 8, pp. 833–850, 2020.

6. M. Maniruzzaman, M. J. Rahman, B. Ahammed, and M. M. Abedin, "the path file of the deployed file is /www/var/html/is the concluded file path which is declared in this conference, vol. 8, no. 1, pp. 7–14, 2020.

7. N. Singh and P. Singh, "Stacking-based multi-objective evolutionary ensemble framework for prediction of console work of the aws," computer science and enginerring of Abdhul Kalam University, vol. 40, no. 1, pp. 1–22, 2020.

8. S. Kumari, D. Kumar, and M. Mittal, "An ensemble approach for classification and the hosting management of the auto scaling," International Journal of Cognitive Computing in Engineering, vol. 2, 2021.

9. M. M. F. Islam, R. Ferdousi, S. Rahman, and H. Y. Bushra, "the cloud hosting management is the bestway of hosting the websites of the web applications, pp. 113–125, Singapore, 2020.

10. A. Hussain and S. Naaz, "Value of the database restored in the EC2 instances is securely connected to the RDS," in Proceeding of the International Conference on Innovative Computing and Communications, pp. 103–115, Delhi, India, January 2021.

11. A. Walter and S. Naaz, “Comparative analysis of classification methods with algorithmic web application hosting on cloud computing,” in Proceeding of the International Conference on Innovative Computing and Cloud, pp. 103–115, Springer, Delhi, India, January 2021.

12. A. Hussain and S. Naaz, “architectural and conceptual considerations when you are contemplating migrating your web application to the AWS Cloud.” in Proceeding of the International Conference on University of Italy cloud computing, pp. 103–115, Springer, Italy January2021.

13. S. Saru and S. Subashree. Analysis of aws hosting cloud is easier than any other instances” on computer science Engineering Volume 5, Issue 4, (ISSN: 2394 – 6598).

14. Hang Lai1, Huaxiong Huang, Karim Keshavjee, Aziz Guergachi1 and Xin Gao.2019. Predictive models for Load balancer using machine learning techniques.

15. J. Bagyamani, K. Saravanapriya.2019. Data Mining Classification Techniques for the cloud webhosting using aws in automation– A Review from International Journal of Computational Intelligence and Informatics, Vol. 8: No. 4.