A
Project Report
On
" AUTOMATION FOR INSTALLATION OF SPLUNK APP ON CLUSTERED ARCHITECTURE USING JENKINS AND ANSIBLE "

Prepared by

Dirgh Jani (17IT035)
Prachi Patel (17IT075)

Under the guidance of

Prof. Sandip Patel

A Report Submitted to
Charotar University of Science and Technology
for Partial Fulfillment of the Requirements for the
Degree of Bachelor of Technology
in Information Technology
(8th Semester Software Project Major-IT447)

Submitted at

CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

SMT. KUNDANBEN DINSHA PATEL DEPARTMENT OF INFORMATION TECHNOLOGY
Chandubhai S. Patel Institute of Technology
At: Changa, Dist: Anand – 388421
April 2021

CANDIDATE'S DECLARATION

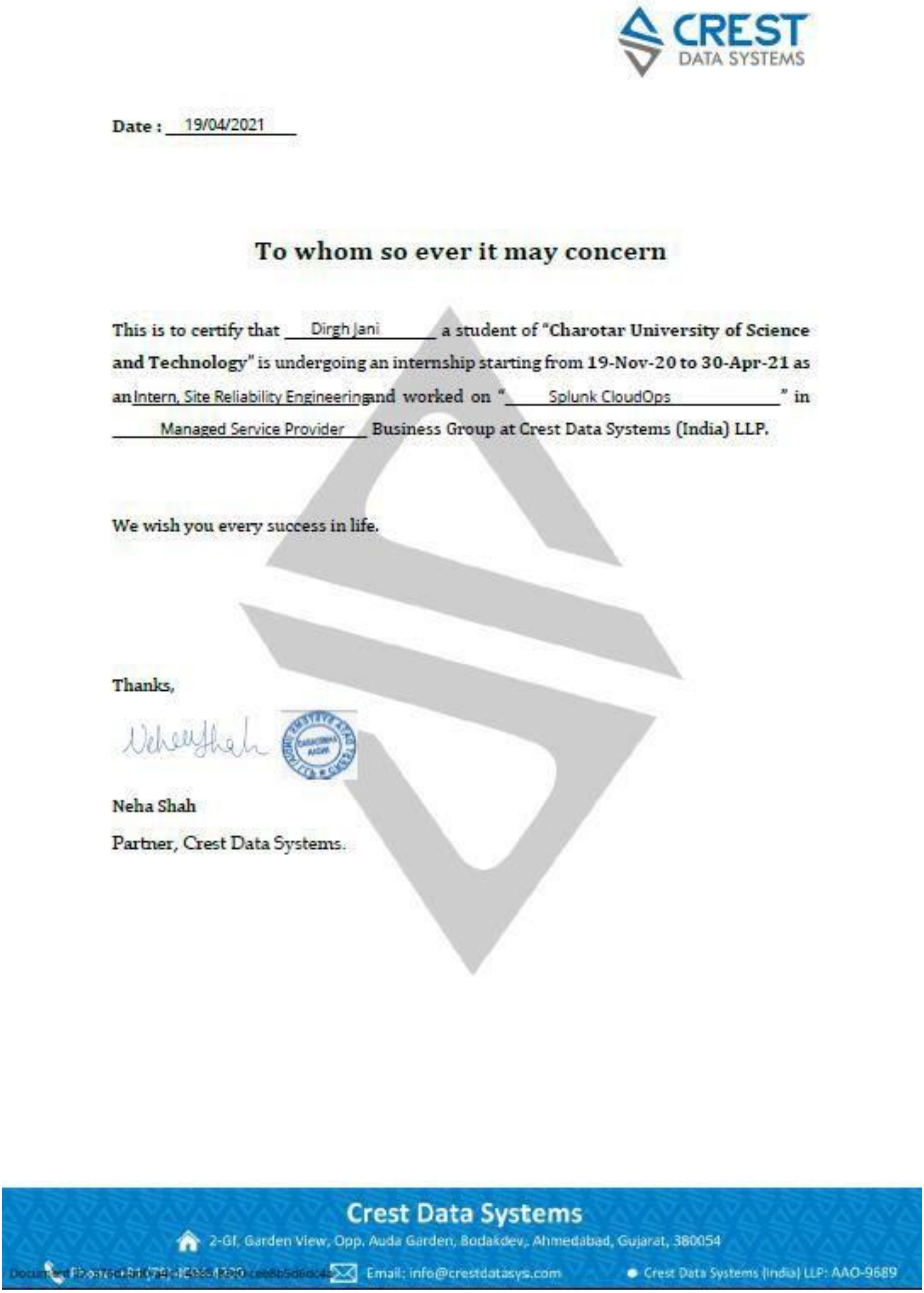
We hereby declare that the project entitled "**AUTOMATION FOR INSTALLATION OF SPLUNK APP ON CLUSTERED ARCHITECTURE USING JENKINS AND ANSIBLE"** is our own work conducted under the guidance of **Prof. Sandip Patel** and **Mr. Dhruval Sharma & Ms. Shreya Palejkar.**

We further declare that to the best of our knowledge, the project for B. Tech does not contain any part of the work, which has been submitted for the award of any degree either in this University or in other University without proper citation.

Dirgh Jani
(17IT035)

Prachi Patel
(17IT075)

Prof. Sandip Patel
Assistant Professor,
Smt. Kundanben Dinsha Patel Department of Information Technology,
Faculty of Technology & Engineering,
Changa – 388425.



CREST
DATA SYSTEMS

Date : 19/04/2021

To whom so ever it may concern

This is to certify that Dirgh Jani a student of **"Charotar University of Science and Technology"** is undergoing an internship starting from **19-Nov-20 to 30-Apr-21** as an Intern, Site Reliability Engineering and worked on " Splunk CloudOps " in Managed Service Provider Business Group at Crest Data Systems (India) LLP.

We wish you every success in life.

Thanks,

Nehashah

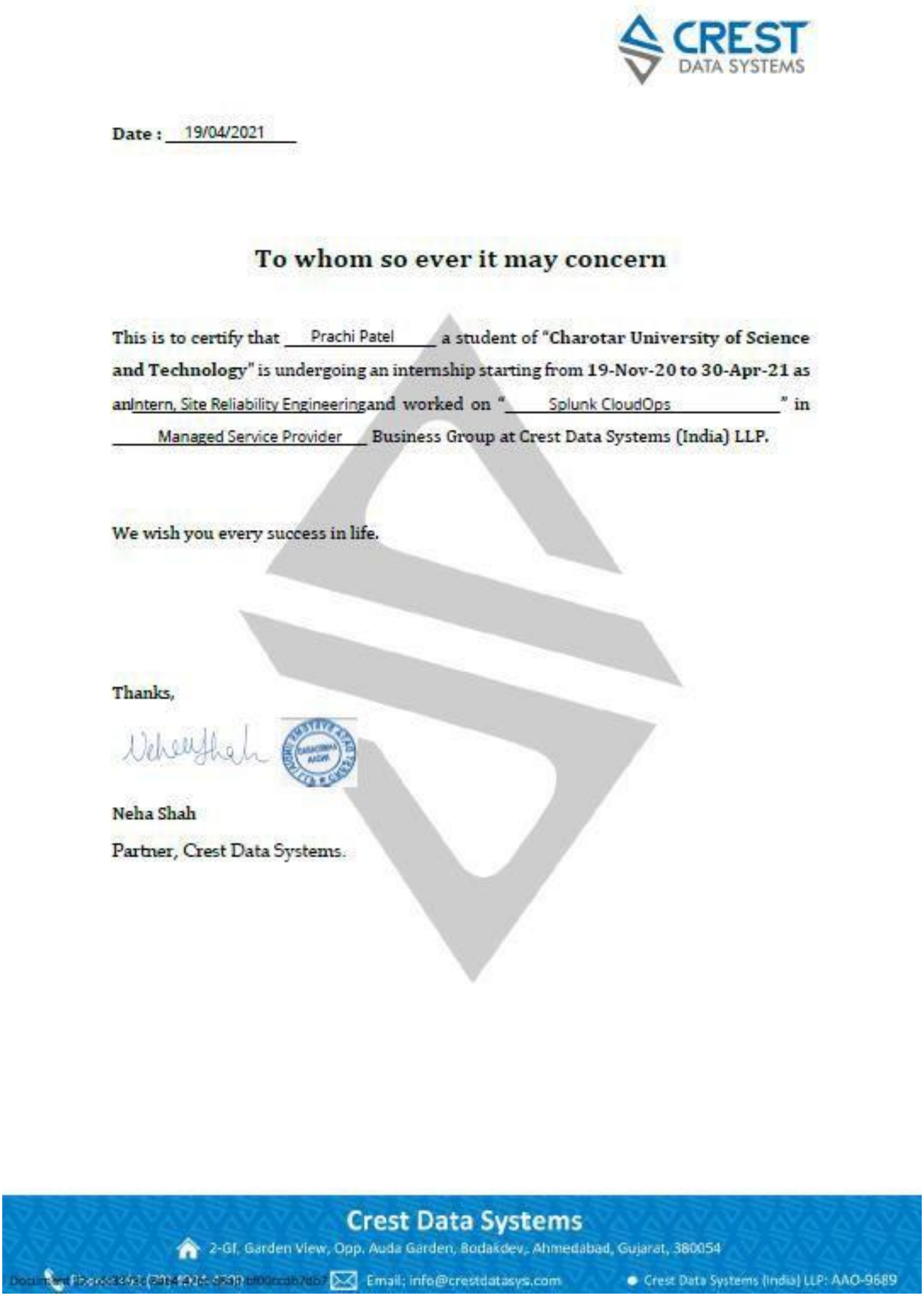
Neha Shah

Partner, Crest Data Systems.

Crest Data Systems

2-Gf, Garden View, Opp. Auda Garden, Bodakdev,. Ahmedabad, Gujarat, 380054

Phone: +91 (79) [illegible] Email: info@crestdatasys.com ● Crest Data Systems (India) LLP: AAO-9689



CREST DATA SYSTEMS

Date : 19/04/2021

To whom so ever it may concern

This is to certify that Prachi Patel a student of "Charotar University of Science and Technology" is undergoing an internship starting from 19-Nov-20 to 30-Apr-21 as anIntern, Site Reliability Engineeringand worked on " Splunk CloudOps " in Managed Service Provider Business Group at Crest Data Systems (India) LLP.

We wish you every success in life.

Thanks,

NehaShah

Neha Shah

Partner, Crest Data Systems.

अमृतं तु विद्या

CHARUSAT

CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CERTIFICATE

This is to certify that the report entitled "**AUTOMATION FOR INSTALLATION OF SPLUNK APP ON CLUSTERED ARCHITECTURE USING JENKINS AND ANSIBLE**" is a bonafied work carried out by **JANI DIRGH(17IT035) and PRACHI PATEL(17IT075)** under the guidance and supervision of **Prof. Sandip Patel** & **Mr. Dhruval Sharma and Ms. Shreya Palejkar** for the subject **Software Project Major (IT447)** of 8th Semester of Bachelor of Technology in **Information Technology** at Faculty of Technology & Engineering – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred to the examiner.

Under supervision of,

Sandip Patel

Prof. Sandip Patel
Assistant Professor
Smt. Kundanben Dinsha Patel
Department of Information Technology
CSPIT, Changa, Gujarat.

Dhruval Sharma

Mr. Dhruval Sharma
Junior Software Engineer
Crest Data Systems.

Shreya Palejkar

Ms. Shreya Palejkar
Site Reliability Engineer
Crest Data Systems.

Dr. Parth Shah
Head & Associate Professor
Smt. Kundanben Dinsha Patel
Department of Information CSPIT,
Changa, Gujarat.

Chandubhai S Patel Institute of Technology

At: Changa, Ta. Petlad, Dist. Anand, PIN: 388 421. Gujarat

ABSTRACT

Over the last decade, we know that there is an exponential growth in machine data. The main reason can be the growing number of machines in the IT infrastructure and also due to the increased use of IoT devices. The data generated by machine contains a lot of valuable information that can drive efficiency, productivity and visibility for any business. The purpose why Splunk was founded is: *To Make Sense of Machine Generated Log Data.*

Splunk is a software platform that serves the engine for searching, monitoring, visualizing, analysing and acting on large volumed streams of real-time machine generated data. It is versatile technology because of its wide application and suitability.

For setting up Splunk, the basic requirement is some sort of scalable computing platform. Due to elasticity and better service packages, AWS fits perfect. Amazon Web Services(AWS) provides computing resources for cloud with useful services like EC2, EBS, VPC. The application can be easily managed and more efficient in terms of computing by using all the mentioned services.

With the growing enterprise, the number of clients also increases. This change requires to scale up and don infrastructure as per the need which is a complex task. Here comes Terraform in picture to handle huge numbers of machines in an easy and secure way. From just one click, we can do spin up on any number of machines. Also with the help of Terraform, we can install Splunk on each machine and add necessary steps for forming the cluster. The app can be installed on the infrastructure with the help of ansible playbook.

ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many hands. We would like to extend my sincere thanks to all of them.

We are highly indebted to Prof. Sandip Patel for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project.

We are grateful to my external guide Mr. Dhruval Sharma and Ms. Shreya Palejkar in Crest Data System for giving us the support and encouragement that was necessary for the completion of this project.

We would like to express our gratitude to H.O.D. Dr. Parth Shah and we are also grateful to all our faculty members of Chandubhai S. Patel Institute of Technology for their kind cooperation and encouragement which help us in completion of this project and preparing the report.

Last but not the least, We would also like to thank our colleagues, who have co – operated during the preparation of our report and without them this project has not been possible. Their ideas helped us a lot to improve my project report.

“We may not achieve everything we dream, but we cannot achieve anything unless we dream.” –

Dirgh Jani(17IT035)

Prachi Patel(17IT075)

Table of Contents

List of Figures

1. INTRODUCTION

1.1 PURPOSE

The purpose of Software Requirements Specification (SRS) is to provide a detailed description of**: Automation for installation of splunk app on Clustered Architecture using Jenkins and Ansible.**

SRS will give the complete understanding of purpose and its functionality. This document helps developers to understand software correctly as well as it can be used as a software validation document for users.

1.2 SCOPE

Manual configuration of Splunk Cluster on AWS is a very tedious job, like we need to create instances, manage SG's and other network related parts. We need to install and set up Splunk on each instance. Also deploying applications includes multiple steps like configuring its environment. Forwarding application and its machine generation data into the Splunk Cluster environment and creating a story out of that data becomes extremely gruesome.

This process is very time consuming as well as erroneous because as there is a human intervention, there is a little scope of error for sure. Another reason is for the big cluster having hundreds of instances, this manual work can't be possible because it may happen that the few machines are configured differently due to human error.

This Jenkins, Ansible and Terraform automation system will take care of this lengthy process by reducing the scope of error thereby atomising this deployment.

1.3 DEFINITIONS, ACRONYMS and ABBREVIATIONS

Instances: AWS EC2 instance means a VM (Virtual machine) provided by AWS. Here we are using Linux VM

AWS: Amazon Web Services – cloud computing service

c0m1: Cluster Master (Part of Splunk architecture which manages the indexer clustering)

sh: Search-head (Part of Splunk architecture where we can access and search data)

idx: Indexer (Part of Splunk architecture where all data gets indexed)

fwd: Forwarder (Part of Splunk architecture which is setup at data generation machine to forward data to Splunk system.

SRE: Site Reliability Engineer. It is the user of the system whose work is to manage the cloud or on prime infrastructure

1.4 OVERVIEW

The following sections of this document will focus on describing the system in terms of product, functionalities and dependencies, external interface requirements, functional requirements, performance requirements and another requirement.

2. PROJECT MANAGEMENT

2.1 PROJECT PLANNING

2.1.1 PROJECT DEVELOPMENT APPROACH AND JUSTIFICATION

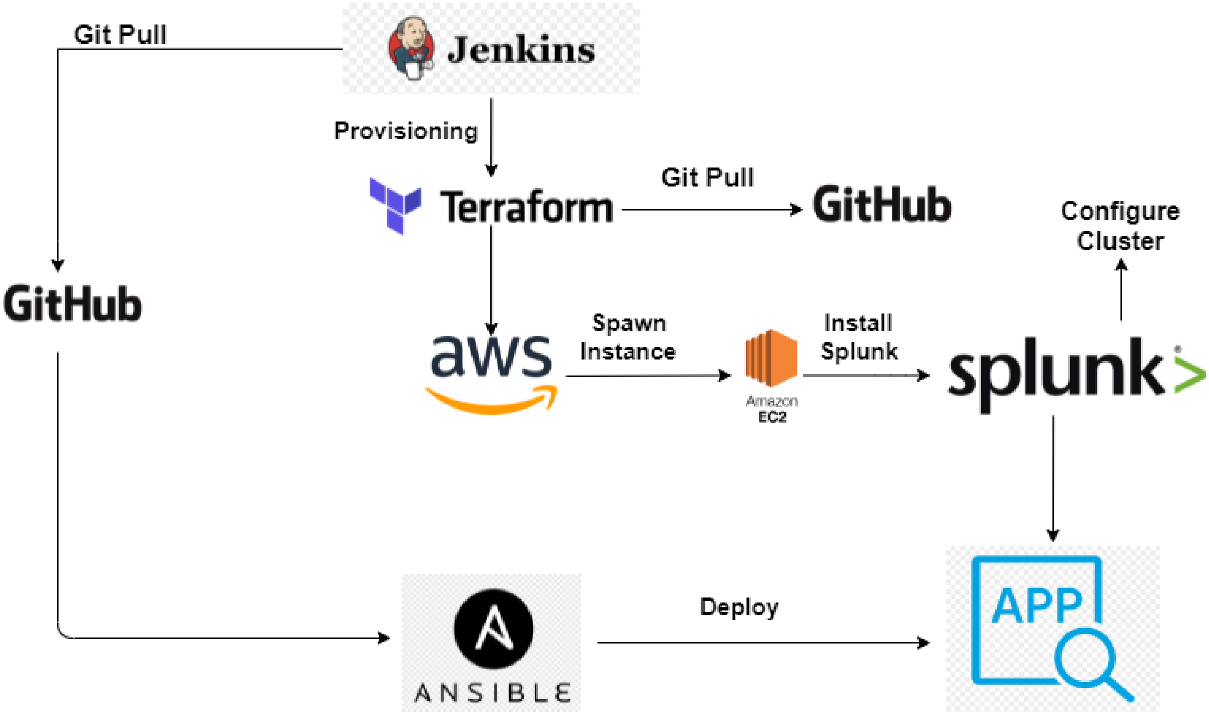


Figure 1: Project Flow

PROJECT DEVELOPMENT APPROACH:

Here, we adopted the Agile methodology to develop the project. We deployed the various modules of this project and tested them along with real time feedback from the guide at the company.

JUSTIFICATION:

Splunk is considered to be one of the best tool for collecting variety of data from multiple types of domains and indexes them in a distributed/clustered architecture providing real-time monitoring of applications which is the exact requirement for our project.

When it comes to automation with Continuous Integration-Continuous Deployment (CI-CD) pipeline, Jenkins is flexible to serve that purpose of the project.

The project requires a virtual, scalable, secure and fault tolerant platform to deploy applications, Jenkins and Splunk cluster environment on, AWS provides services to address all of the above requirements and much more.

The project needs a lot of automated provisioning and configuration, Terraform is suited for provisioning and Ansible is suited for configuration and app install.

The project uses Git so that the work done by all group members is aligned, shared and version controlled.

2.1.2 PROJECT EFFORT AND TIME, COST ESTIMATION

Estimation technique used: COCOMO model

For evaluating the cost of this project the things which needed to be considered is:-

1.Effort-(how many people)
2. Time- (duration)
3. Person involved
4. Cost

KLOC = 1.10

Effort = ab* (KLOC)^b^{b}

= 3.0*(1.10) ^1.12

= 3.0*1.1126 PM

= 3.33 PM

Tdev = cb* (Effort) ^d^{b}

= 2.5*(3.33) ^0.35

= 3.71 Months

Cost = Avg. salary * Tdev

= 5000 * 3.71

= 20000 Rs.

2.2PROJECT WORK SCHEDULING

2.2.1 PERT CHART REPRESENTATION

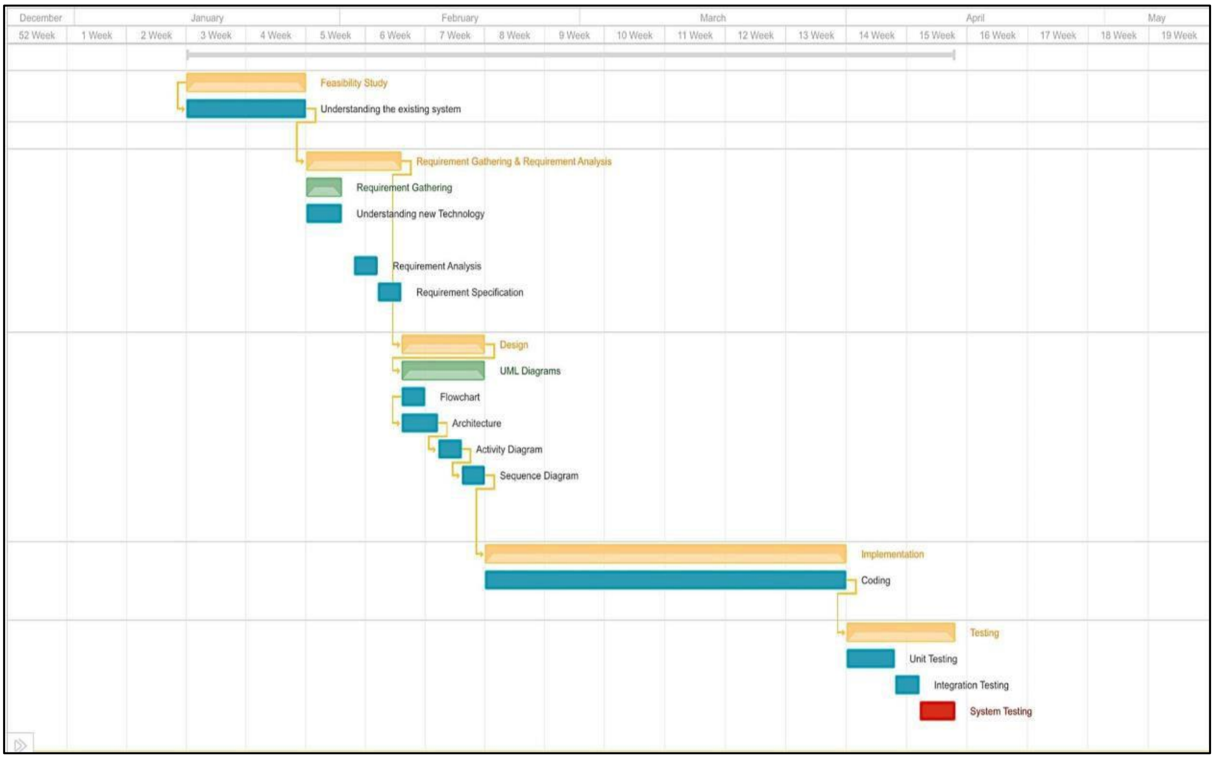


Figure 2: Pert Chart

3. SYSTEM REQUIREMENTS STUDY

3.1 USER CHARACTERISTICS

This Application focuses on the problems faced by SRE to deploy and manage the Splunk architecture.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

SOFTWARE REQUIREMENTS:

- Platform: Linux
- Provisioning Tool: Terraform
- Configuration Management Tool: Ansible
- Integration Tool: Jenkins
- Technology: Splunk, AWS EC2

HARDWARE REQUIREMENTS(DEPLOYMENT):

- 1x t2.micro (search Head) 8 GB RAM,1 vCPU
- 1x t2.micro (Cluster master) 8 GB RAM, 1 vCPU
- 3x t2.micro (Indexers) 8 GB RAM, 1 vCPU

3.3 ASSUMPTIONS AND CONSTRAINTS

Cloud SRE must have a basic knowledge for Jenkins UI and how to use it. He/she must know how the output of Ansible and Terraform looks like in case of successful or failed execution. He/she must have basic knowledge of how Splunk's clustered environment works. He/she must have basic knowledge of AWS as well.

It is assumed that the machine has valid AWS credentials (Access key and Secret Key). Internet Connectivity.

4. SYSTEM ANALYSIS

4.1 SYSTEM INTRODUCTION

To create and understand the whole system we require to get basic knowledge of the below mentioned three main technology

1. Splunk: Product
2. AWS: Cloud platform to deploy the product
3. Terraform: Infrastructure provisioning tool
4. Ansible: Configuration and management tool
5. Jenkins: Automation Pipeline tool

SPLUNK:

Splunk is a tool which aim to squeeze the important information from unorganized machine data. Splunk provides easy and faster service to search and indexing the unorganized machine data. Splunk has three main components

1. Search-Head:

   It is the console where all searching and visualization of data is being performed.

2. Indexer:

   It indexes the data so Splunk search it in a faster way. Basically, it the instance where all data get stored.

3. Forwarder

   It setup on data generation machine. It will send data to indexers. The data sent by the forwarder is a raw data which is not organized or structured.

Splunk provides highly scalable architecture to fulfils the giant need of searching and indexing the terabytes of data. It provides clustering on search-head and indexers.

Splunk Enterprise monitors and also analyses machine generated data from any source to serve Operational Intelligence to optimize your security, IT and business performance. Using the machine learning, intuitive analysis features, packaged applications and open APIs, Splunk Enterprise becomes a flexible tool that can be used to scales from focused use cases to an enterprise-wide analytics backbone.

AWS:

AWS is abbreviation for Amazon Web Services. It is a cloud computing platform which provides services as per requirement elastic resources to fulfil our computing needs whenever required. Along with the computing resources, it also provides other services to manage the security and other IT stuff.

Some useful services provided by AWS:

1. EC2:
   A VM with customized hardware and software requirement.
2. VPC:
   Virtual Private Cloud which is useful to manage network for cloud.
3. S3:
   Storage service to statically manage data.
4. SG:
   Security group which manages restriction of the inbound and outbound traffic

TERRAFORM:

Terraform id s plstform created by HashiCorp. It is an open-source infrastructure as a code software. It permits us to set up their Infrastructure that can be either on-premises or on any other cloud like AWS, Google, etc.

Terraform basically is a very high-level configuration language with so many modules and plugins provided. It can also stores the status of your infrastructure, i.e. if you want to add/remove/modify anything in your current created infrastructure, no need to write extra snippet code for modification. You just need to make changes to existing code and terraform with compare the desired state and the current state of your infrastructure and it will only apply the additional required changes and leave the rest of the infrastructure as it is.

ANSIBLE:

Ansible is an automation platform. It is an open source platform. It is very powerful and very, very simple to setup. Ansible is tool that can be used for application deployment, configuration management and also for task automation. It can also do orchestration, where requirement is to run tasks in sequence and set-up multiple events in a c–hain that must happen on several seperate servers or devices.

One of the many advantage of Ansible is that it doesn't need any client/host to be installed on a remote machine. With Ansible, Every machines that are present on remote server are accessed by SSH. Ansible is also capable in managing services like AWS. So, here we are using Ansible to manage configuration along with Terraform and configure Splunk apps.

JENKINS:

Jenkins is an open source Continuous Integration (CI) automation platform that is capable of orchestrating a chain of multiple actions in difffrent stages that helps to achieve the Continuous Integration.

Using Jenkins we can run multiple tasks in just one go. Pipeline is dvided into stages. Jenkins also provides the facility to put condition between the stages in the pipeline for conditional behaviour outfput.

4.2 PRODUCT FUNCTION

The user specifically SRE (Site Reliability Engineer) can create Splunk architecture on AWS using a single script. SRE can provide options like how many indexers should be there, how many search-heads should be there and all these stuffs.

Even after deployment SRE can modify the system like adding or removing the indexers and search-heads from the cluster. SRE can reset the configuration.

4.3 USER CHARACTERISTICS

This Application focuses on the problems faced by SRE to deploy and manage the Splunk architecture manually.

4.4 REQUIREMENTS OF NEW SYSTEM

4.4.1 FUNCTIONAL REQUIREMENTS

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

R1 → Running Jenkins Pipeline

Description: This is the part that triggers the other tasks

Input: Pipeline code, Secrets, IP address

Output: Automated deployment of infrastructure and configuration of application and Splunk cluster

R2 → Deploying Infrastructure in Cloud

Description: Here the entire infrastructure for Application and Splunk is deployed in Cloud.

Input: Terraform files, Secretes and Installation files.

Output: Up and running infrastructure in Cloud.

R3 → Deploying Application

Description: Here the entire Application is deployed in the respective machine with its dependencies.

Input: Ansible play, Secrets and Installation files.

Output: Up and running application in Cloud.

R4 → Deploying Splunk Cluster

Description: Here the entire Splunk Cluster is configured.

Input: Ansible play, Secretes and Installation files.

Output: Up and running Splunk Cluster environment in cloud

R5 → Adding indexers to the indexer cluster

Description: It is used to modify the indexer cluster.

Input: Number indexer needs to be added.

Output: Indexers added in the Splunk Cluster environment.

4.4.2 NON-FUNCTIONAL REQUIREMENTS

- Reliability – The ability of the architecture to consistently perform as per its defined specifications
- Maintainability - the probability of performing a successful downtime or repair action can be done within a given period of time
- Portability - the ability for transfer of applications and data from one cloud computing environment to another environment with minimal disruption
- Security - the set of processes, procedures and standards that are designed to serve security assurance within a cloud platform environment
- Availability – highly available, the ratio of time the architecture is functional to the total time it is required or expected to function
- Real time monitoring of data – All the monitored real-time is data indexed

4.5 USE CASE DIAGRAM

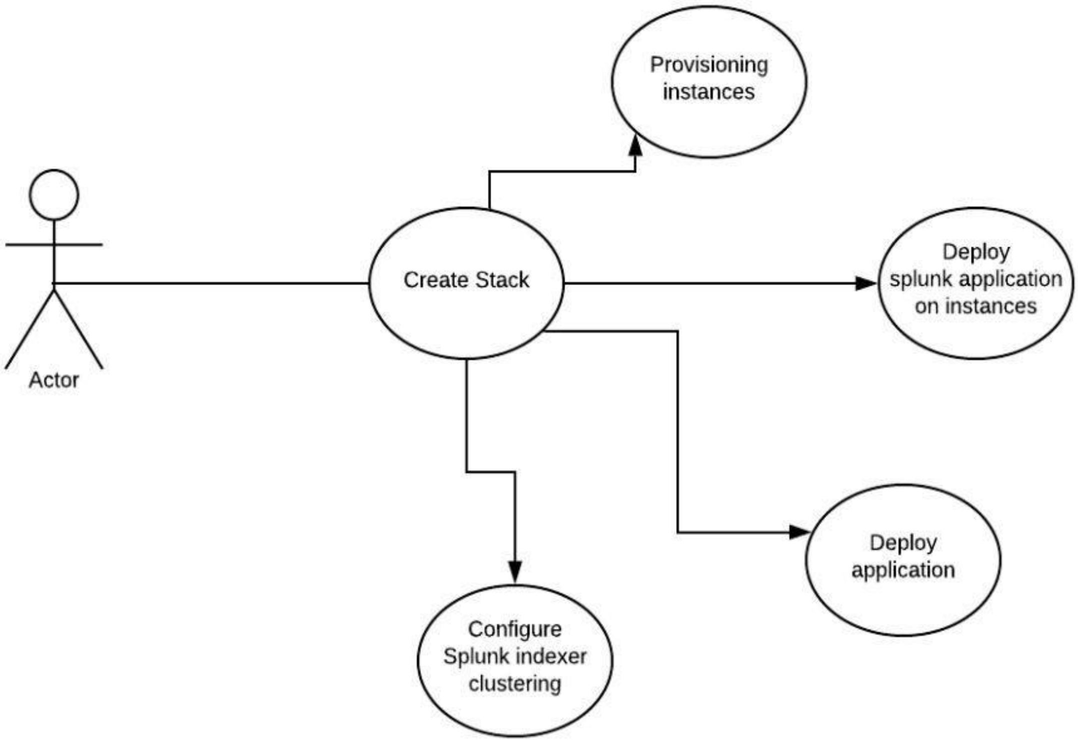

Provisioning instances
Actor
Create Stack
Deploy splunk application on instances
Deploy application
Configure Splunk indexer clustering

Figure 3: Use case diagram

4.6 SEQUENCE DIAGRAM

4.6.1 SEQUENCE DIAGRAM FOR CREATING NEW STACK

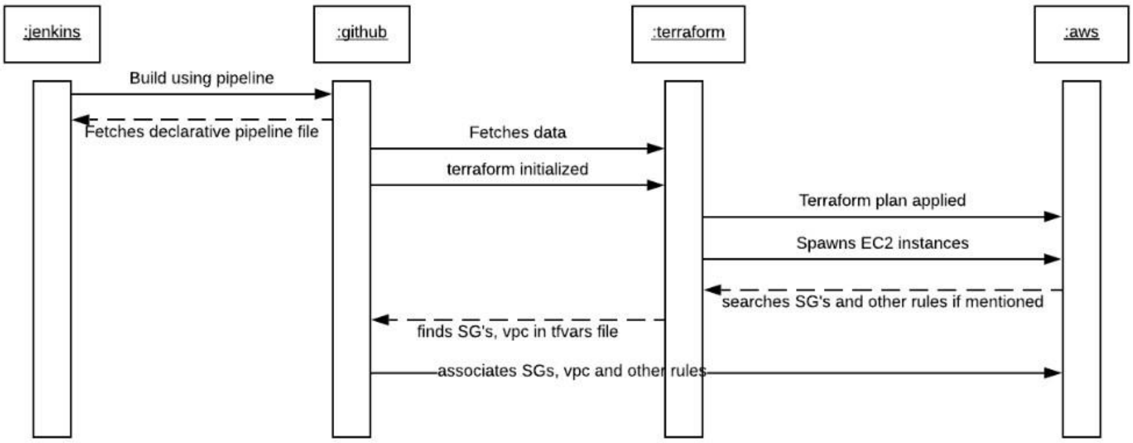

:jenkins
:github
:terraform
:aws
Build using pipeline
Fetches declarative pipeline file
Fetches data
terraform initialized
Terraform plan applied
Spawns EC2 instances
searches SG's and other rules if mentioned
finds SG's, vpc in tfvars file
associates SGs, vpc and other rules

Figure 4 : Sequence Diagram for creating the stack

Brief Description:

The above sequence diagram shows how the stack is being creating for the first time

Here is the flow sequentially:

1. Initially SRE builds stack using Jenkins pipeline.
2. Jenkins pipeline fetches terraform files from GitLab.
3. Terraform is initialized and plan is made to spawn instances on AWS
4. Terraform files will manage AWS modules to create required instances and other instance specific configurations like Security Groups.
5. Instances will be created on AWS and Security Groups, VPC and subnets will be assigned based on specified configurations in terraform.

4.6.2 SEQUENCE DIAGRAM FOR SPLUNK INSTALLATION

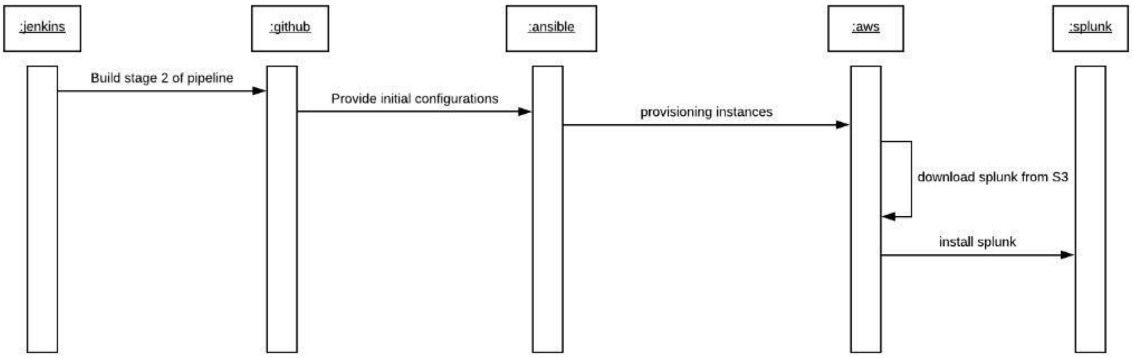


Figure 5: Sequence Diagram for Splunk Installation

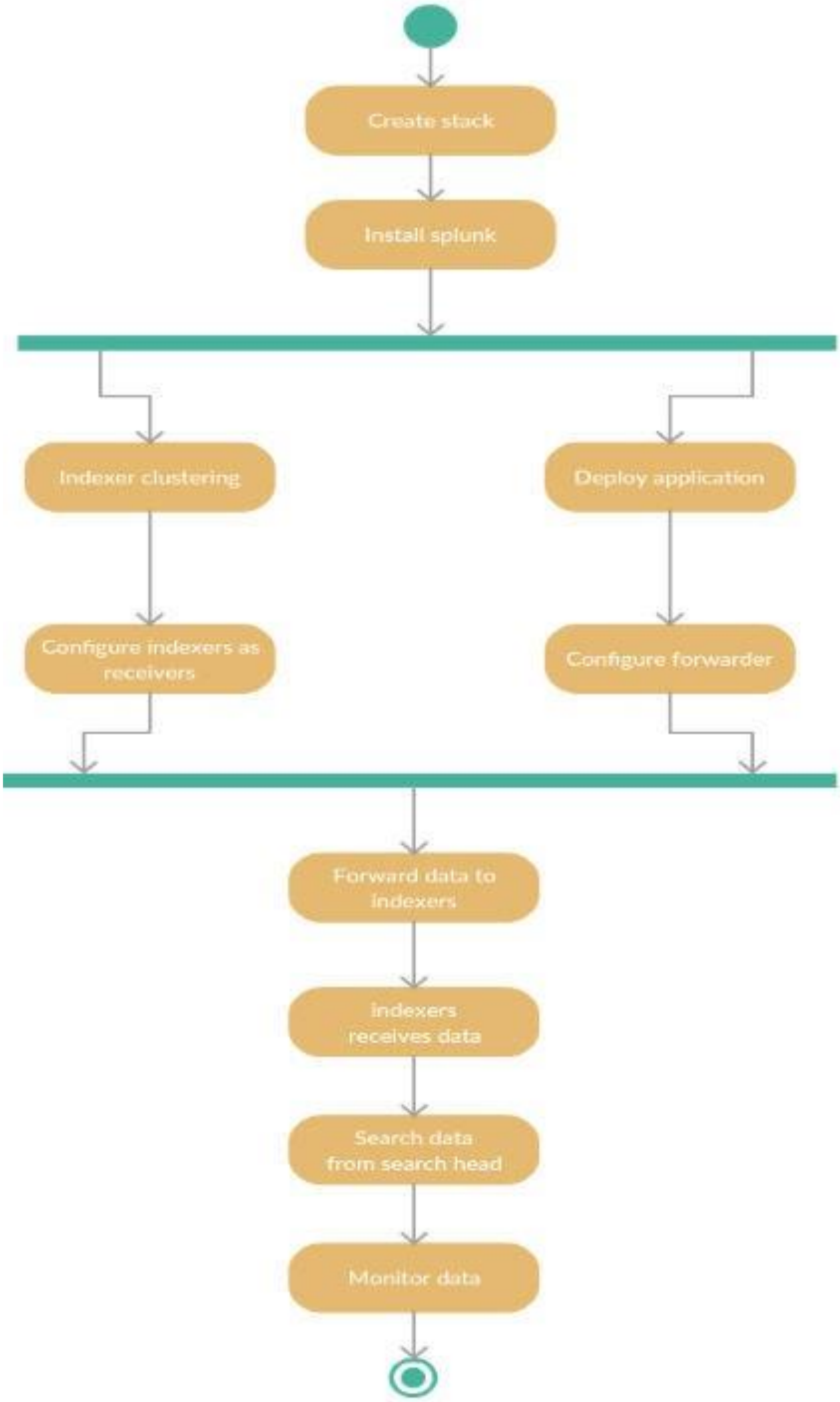
Brief Description:

The above sequence diagram shows how Splunk is installed on instances.

Here is the flow sequentially:

1. Jenkins built 2nd stage of the pipeline.
2. It fetches initial configuration of Splunk deployment from GitLab.
3. Terraform main.tf manages the module.
4. Splunk installation file will be downloaded from website and Splunk will be installed on instances.

4.7 ACTIVITY DIAGRAM



5. SYSTEM DESIGN

5.1 PRODUCTPERSPECTIVE

Deploying Application via Jenkins/Ansible/Terraform and monitoring it by Splunk-cluster is a totally independent system.

5.1.1 SYSTEM INTERFACE

This system will not interact with any types of external interfaces except the CLI.

5.1.2 INTERFACE

There is the UI tool which helps to reduce the manual efforts. There is GUI, but all the tasks are performed in backend using CLI

5.1.3 HARDWARE INTERFACE

The system will run on any basic PC which is connected through the Internet and does not require any external hardware interfaces.

5.1.4 SOFTWARE INTERFACE

Ansible and Terraform Host:

This is the machine on which Ansible and Terraform are installed and configured and all of its tasks will be executed here by Jenkins.

Jenkins Host:

This is the machine on which Jenkins is installed and configured.

5.1.5 COMMUNICATION INTERFACE

Jenkins will communicate with the instances over the internet using SSH protocol.

5.1.6 MEMORY CONSTRAINTS

There is no specific memory constraint, but the system can be best run on machine having primary memory greater than 250 MB.

6. IMPLEMENTATION

6.1 SPLUNK

Nowadays we use servers, devices, apps logs, system logs , network traffic, cloud data. We see data everywhere and use modern technology. Machine data is everywhere. Intersect with our lives and businesses that change for the betterment of our life . Splunk offers the leading platform for Operational Intelligence and mange machine data. It enables with detail analysis of machine data and also convert in human readable form which can help make your company more productive, profitable, competitive and secure.

Splunk Enterprise monitors and analyses machine data which is generated from system logs and devices from any source to deliver to Splunk Platform for optimization of raw data. With these analysis features, machine learning, packaged Apps and Add-ons, Splunk Enterprise is a flexible platform that focus on all the use cases which are:

- Collects and indexes log and machine raw data from any source
- Platform to search, visualize and analyze data from users of all types
- Apps and Add-ons to provide solutions for security, business analysis and more
- Can be used across on premise, cloud and hybrid environments

Machine-generated data is one of the fastest growing and complex areas of big data.

This machine data is center of all the digital information created by application, servers, and network devices, sensors and more. In nutshell it produces a large amount of data which is complex to understand as it does not contain any visualization and also in unformatted format. It's collects record of all user transactions, customer, machine, security threats, fraudulent activity and behaviour and more. Splunk turns machine data into valuable insights of all the business. It's what we call Operational Intelligence.

Operational Intelligence is collection of all the business analysis system designed to help in decision making of all the raw data in real time. It gather various data from sources and then analyzes and digests these feeds as the data arrives. It gives you a real-time understanding of across your IT systems and technology and business so you can make proper decisions of company. It is enabled by the Splunk platform, and some of Splunk's products like premium app solutions, apps and add-ons.

Machine data is one of the most unused and unhelpful sort of data assets of any organization. But some of the most important information that you can get in IT and business is hidden in this data: where things went wrong, how to optimize the customer experience, the fingerprints of fraud. All of these information can be found in the machine data generated by the normal operations of your organization.

Machine data is valuable because it contains a definitive record of all the activity and behaviour of customers, users, transactions, applications, servers, networks and mobile devices. It includes configurations, data from APIs, message queues, change events, the output of diagnostic commands, call detail records and sensor data from industrial systems, and more.

The challenge with exploiting machine data is that it comes in a dizzying array of unpredictable formats, and traditional monitoring and analysis tools were not designed for the variety, velocity, volume or variability of this data. This is where Splunk comes in.

The Splunk platform uses machine data—the digital offloading created by the systems, technologies and infrastructure powering modern businesses—to address big data, IT operations, security and analytics use cases. The information captured from machine data can support any number of use cases of any organization and can also be enrich with data in understandable from other sources. The enterprise machine data fabric shares and provides access to machine data across the organization to facilitate these information. This is what we call Operational Intelligence.

6.1.1 ARCHITECTURE

There are 3 main components (instances) in Splunk,

- Search-head
- Indexer
- Forwarder

All this component can reside in a single machine which is called standalone as well as each on a different machine which is called distributed architecture. You can get the best results if all the three are on different machines. All these are instances of Splunk means all the machine will have the same Splunk installed only the configuration for each of them will be different as per the components which we want to configure.

Search-head:

Search heads is the component used for interacting with Splunk. It provides GUI to users to perform various operation like you can search query which data is stored in indexer to gain access to specific data. It will use Splunk processing language. After the query gets executed the data can be visualized in graphical formats like charts, graphs and reports.

Indexer:

The data can be injected into Splunk in three ways: By monitoring the files and directories, API calls, Forwarding the files. When data enters indexer process machine data and transform into events and store it in indexes which then can be searched by the Search Head. It enhances the data in separating data stream into individual and searchable event. This indexing process is called as event processing.

Forwarder:

The forwarder is agent to deploy on IT system, which collects logs and data and send to indexer. It can be installed on multiple system and collects data simultaneously from different machine in real time.

There are two types of forwarder,

Universal Forwarder: forwards raw data without any priority treatment. It is faster and require less resources on host machine but huge amount of data is send to indexer **Heavy Forwarder**: performs parsing and indexeing before forwarding to indexer on host machine.

Clustering:

Clustering refers to a group of similar components in which captain handles the all the rest other component of that type. There are 2 types search head clusters, indexer clusters.

Replication Factor:

The replication factor determines the number of copies of each search artifact or result that cluster maintains. One Copy of data will act as a master copy and another will act as a duplicate copy. Search head will first search for data in the master copy. The default values are 3.

Search Factor:

The search factor determines the number of searchable copies of data the indexer cluster maintains i.e it maintains copies of each bucket. It must be less than or equal then the replication factor and default is 2.

Search-head captain:

It manages the search load coming from user. It distributes search job based on load and ensure same amount of set of objects are distributed among different search head member in cluster.

Cluster master:

It will hold the data of the indexer's. Which data is stored in which indexer is known to the index master. So, when search head fires queries for data then it will be first directed to the index master from which it will find which indexer has the data the search head is looking for. Manages configuration of all the peer nodes/indexer.

6.1.2 SPLUNK INDEXES

When first hearing about Splunk some think “database”. But that is a misunderstanding. In a database we require to create tables and fields before storing data Splunk has totally different concept in which we accepts almost any machine data spontaneously after installation. In other words, Splunk does not have a fixed schema.

Instead, it performs field extraction at search time. Splunk, logs formats automatically as they are recognised, else data is specified in configuration files or right in the search expression.

This approach allows for great flexibility. Splunk indexes any kind of machine data that can be represented as any visualization.

During the indexing phase, when Splunk processes incoming data and prepares it for storage, the indexer makes one significant modification: it splits characters into individual events. The events typically correspond to lines within the log file being processed. Each event gets a time-stamp, typically parsed directly from the input line, and some other default properties like source machine. Event keywords are then added to an index file to hurry up subsequent searches and therefore the event text is stored in very a compressed file located in the file system.

Infinite retention without losing granularity. Some tracking products only allow you to keep large amount of data for many months, weeks or even days data. Others reduce the granularity of older events, compressing many data points into one due of capacity constraints. It same is not true for Splunk. It can literally index hundreds of terabytes and gigabyte of data per day and keep practically unlimited amounts of data storage.

Types of Data Splunk Can Read:

One of the common characteristics of machine data is that it nearly always contains some indication of when the data was created or when an event described by the data occurred. Given this feature, Splunk’s indexes are optimized to retrieve events in time-series order. If the raw data does not have an explicit timestamp, Splunk assigns the time at event wa s indexed by Splunk to the events, such as the time the file was last modified or the timestamp of previous events.

The other requirement is that the machine data should be textual and not binary data. Image and sound files cannot be used to process the data in Splunk as they are examples of binary data files. Splunk can call your scripts to perform that conversion before indexing the data. Ultimately though, Splunk data must have a textual representation of raw data to be indexed and searched for further use.

Meaning of Index:

Splunk Enterprise provides a repository of Splunk data which is called indexes. An index is a collection of databases, which are sub-directories of all the Splunk logs located at *$SPLUNK_HOME/var/lib/splunk.*

Indexes consist of two types of files: raw data files and index files. Splunk Enterprise can index any type of time-stamp data. When Splunk Enterprise indexes data, it breaks and converts it into events, based on the time-stamps of data.

Event processing and the data pipeline

The data enters the indexer and proceeds through a pipeline where event processing occurs. Finally, the processed data is written to disk. The main pipeline consists of several shorter pipelines which are connected together. A single instance of this end-to-end data pipeline is termed a pipeline set.

Event processing occurs in two main stages, parsing and indexing. All data that arriving into Splunk Enterprise enters through the parsing pipeline as large chunks of data.

During parsing, Splunk Enterprise splits these chunks of data into events that it passes to the indexing pipeline, where final processing occurs.

Number of steps while parsing:

1. Extracting a collection of default fields for each and every event, including host, source and source type.
2. Configuration character set encoding.
3. Identifying line termination using line breaking rules. These events form a queue if they there are many events.
4. Identifying time-stamps or creating them if they do not exist. At the same time that it processes time-stamps, Splunk identifies event boundaries.
5. Splunk can be set up to mask sensitive event data (such as credit card or social security numbers) at this stage and can also configure custom metadata for any events will occur in future.

While in the indexing pipeline, Splunk performs additional steps:

1. Dividing all events into segments that can be searched. You can determine the level of segmentation, which affects indexing and search speed, search capacity and disk compression efficiency.
2. Building the index data structures
3. Writing the raw data and index files to disk, where post-indexing compression occurs.

The breakdown between the parsing and indexing pipelines is of primary relevance when deploying forwarders. Heavy forwarders will run raw data through the parsing pipeline and then forward the parsed data on to indexers for final indexing. While instead, universal forwarders forward the raw data to the indexer, which then processes it through both pipelines. Note, however, that both types of forwarders do a type of parsing on certain structured data.

When Splunk indexes raw data, it tags each event with number of fields and it will become part of the index event data. The fields that are added automatically are known as default fields. It serves number of purposes:

Internal fields:

- *_raw*
- *_time*

- *_indextime*
- *_cd*

These fields which is used by Splunk for internal processes and contain information regarding software usage.

Basic default fields:

- host
- index
- linecount
- punct
- source
- sourcetype
- splunk_server
- timestamp

These fields provide information about an event, like what and where its source, what data it contains, what index it's located in, how many lines it contains, and when it occurred

Default date-time fields:

date_hour, date_mday, date_minute, date_month, date_second, date_wday, date_year, date_zone

These fields provide additional searchable regularity to event timestamps.

Three basic default fields are as follows:

1) **Host**

   A default field which contains the host name or IP address of the network device which are generated are converted into an event and each event has a host field. The indexer generates this field during indexing. Use the host field to searches and narrow the search results of the data to events source from a particular device.

2) **Source**

A default field that identifies the source of an event, that is, what is the event source. When data are indexed from files and directories, the event which is generated contains the full path name of the file or directory from where it is searched and when network-based source, it consists of the protocol and port. Indexer generates the source field at during indexing of data.

3) **Source-type**

A default field that identifies the data structure of an event. It determines how data is formatted during the indexing process. Splunk already has a large set of predefined source types, and it automatically assigns a source type to your data and we can also create a custom source type.

Default set of indexes:

Splunk contains some preconfigured indexes, which are:

main:

This is the default Splunk Enterprise index. All processed and indexed data is stored here or else it is specified.

_internal:

Stores Splunk Enterprise internal logs and processing metrics.

_audit:

Contains events related to the file system change monitor, auditing, and all user search history.

The Splunk admin creates new indexes, edit properties, remove and relocate of currently defined indexes. It can also manage indexes through Splunk Web UI and via backend which are through CLI, and configuration files of index which known as indexes.conf.

Indexer Workflow to store Indexes:

As the indexer indexes your machine data which are collected in indexer, it creates a number of files. These files contain two types of data:

- The raw data in compressed form (raw data)

- Indexes that point to the raw data, plus some metadata files (index files, also known as tsidx files)

Together, these files constitute the Splunk Enterprise index. The files are located on sets of directories arranged by age. Some directories/folder contains newly indexed data, others contain previously indexed data. The number of such directories can grow quite large, depending on how much data you're indexing.

Data Aging:

In Splunk each of the index is stored as indexed data in a bucket. To summarize so far:

- An "index" contains compressed raw data and associated index files.
- The data is stored is limited by the time range and age.
- An indexed data is stored in directory called as bucket.
- A bucket moves through several stages as it ages:
  - Hot
  - Warm
  - Cold
  - Frozen
  - Thawed
- As buckets age, they "roll" from one stage to the next. When indexing is done and all data is indexed it is written to hot bucket. Hot buckets are searchable and active.
- When certain conditions occur and meet (for example, the hot bucket reaches a certain size or Splunk service gets restarted), the hot bucket data starts writing data to warm bucket and a new hot bucket is created in its place. Warm buckets are searchable but are not actively written to.
- When next conditions are reached (for example, the index reaches some maximum number of warm buckets), the indexer begins to write in the warm buckets to cold, based on certain age.
- The Splunk setting always selects the oldest warm bucket to write data next to cold
- Buckets continue to write data to cold as the condition get matched in the policy.

- After a certain age period of time, cold buckets writes data to frozen bucket, in this bucket it is decided that whether data should be archived or deleted.
- All the attributes and properties are set in indexes.conf file in which all the bucket rotation policy and and age policy is specified the next level of the buckets
- If the frozen data has been archived, it can later be thawed. Thawed data is valid for searching data.
- The collection of buckets in a particular stage is referred to as a database or *"db"*: the *"hot db"*, the *"warm db"*, the *"cold db"*, etc.

Index Directories:

Each index occupies its own directory under *$SPLUNK_HOME/var/lib/splunk***.** The index stanza which we want to create is defined as same name of directory. Under the index directory are a series of sub-directories the buckets cycle is stated by stage (hot/warm, cold, or thawed) to store indexed data. The buckets themselves are sub-directories within those directories. The age of the data are based on bucket directory name and date.

6.2 TERRAFORM SCRIPTS

Terraform is a open-source software as infrastructure as code for building, configuring and managing infrastructure securely and efficiently. Terraform can manage resources like cloud, on premises and hybrid infrastructure. Terraform builds a graph or flow diagram of whole infrastructure of all resources and parallelizes the creation and modification of any non-dependent resources. Because of this, Terraform builds infrastructure as efficiently and operators get information of dependencies of infrastructure. It is also store template which can be used for future.

Important components of Terraform:

- Terraform init
- Terraform plan
- Terraform apply
- Terraform destroy

6.3 ANSIBLE PLAYBOOKS

Ansible playbooks is the YAML file which helps to interact when we configure a file with Ansible. The playbook is structured using play which are basically defines the steps to create infrastructure or install apps on architecture. According to the logic and instructions given on the playbook Ansible will first fetch the instances IP and then perform specific task.

6.4 SYSTEM STUDY

- AWS provides on dynamics instances creation which is helpful while deploying the Splunk architecture and create the cluster
- Ansible provide the orchestration of the tasks which will be performed on each Linux machine
- Splunk is the tool which is deployed on AWS instances

6.5 DEPLOYMENT OF INFRASTRUCTURE

The deployment is divided into three parts

- Deploying the infrastructure
- Configuring Splunk cluster

Deploying the infrastructure:

This is the pipelined stage of **JENKINS** which does below tasks:

- Verify with Github environment with credentails
- Fetching scripts from Github
- SSH into machine with Terraform environment
- Running those scripts
- Spinning up instances in AWS
- Attaching Security Groups to those instances.

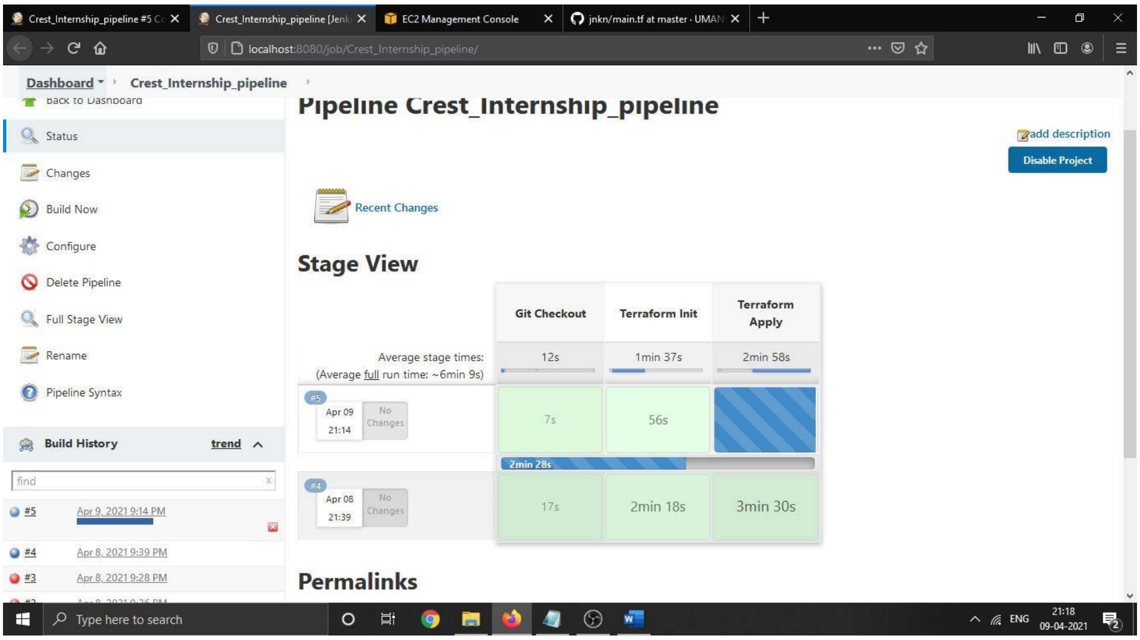
Configuring Splunk cluster:

This is the pipeline stage which does below tasks:

- SSH into machine with Ansible environment
- Fetching playbooks from Gitlab
- Running those terraform
- Creating Splunk cluster environment

- Deploying application with its dependencies

6.6 SCREENSHOTS



Pipeline Crest_Internship_pipeline

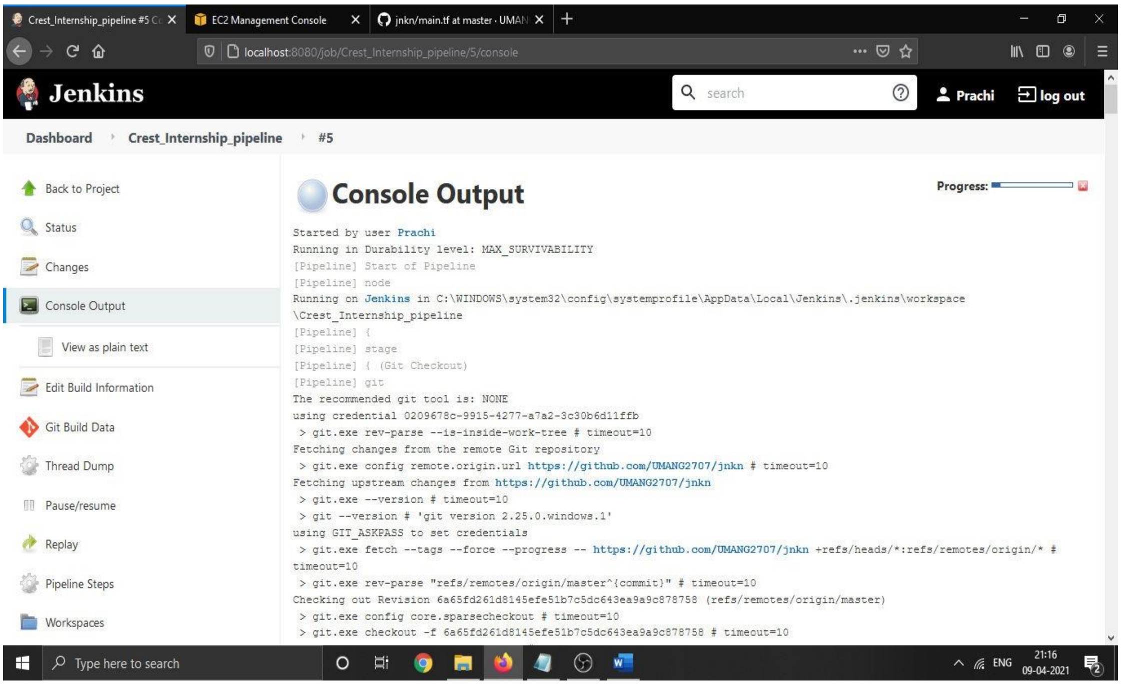
Stage View

	Git Checkout	Terraform Init	Terraform Apply
Average stage times: (Average full run time: ~6min 9s)	12s	1min 37s	2min 58s
#5 Apr 09 21:14 No Changes	7s	56s	2min 28s
#4 Apr 08 21:39 No Changes	17s	2min 18s	3min 30s

Build History:

- #5 Apr 9, 2021 9:14 PM
- #4 Apr 8, 2021 9:39 PM
- #3 Apr 8, 2021 9:28 PM

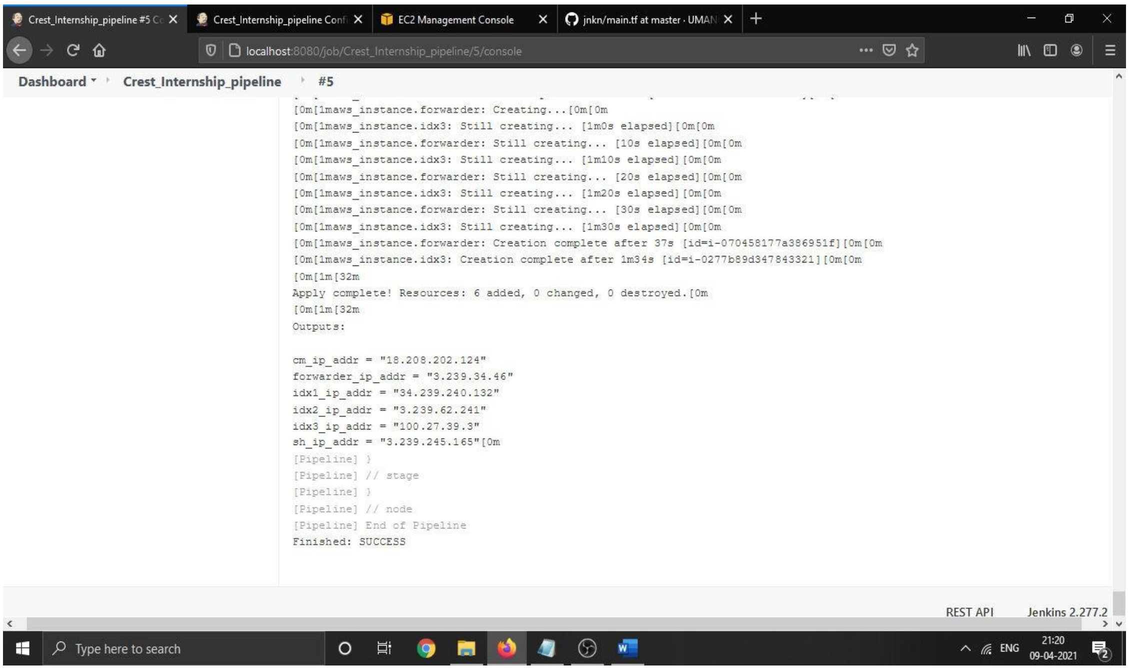
Figure 7: Jenkins Pipeline



Console Output

```
Started by user Prachi
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace
\Crest_Internship_pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Git Checkout)
[Pipeline] git
The recommended git tool is: NONE
using credential 0209678c-9915-4277-a7a2-3c30b6d11ffb
 > git.exe rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/UMANG2707/jnkn # timeout=10
Fetching upstream changes from https://github.com/UMANG2707/jnkn
 > git.exe --version # timeout=10
 > git --version # 'git version 2.25.0.windows.1'
using GIT_ASKPASS to set credentials
 > git.exe fetch --tags --force --progress -- https://github.com/UMANG2707/jnkn +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision 6a65fd261d8145efe51b7c5dc643ea9a9c878758 (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f 6a65fd261d8145efe51b7c5dc643ea9a9c878758 # timeout=10
```

Figure 8: Spawning Instances

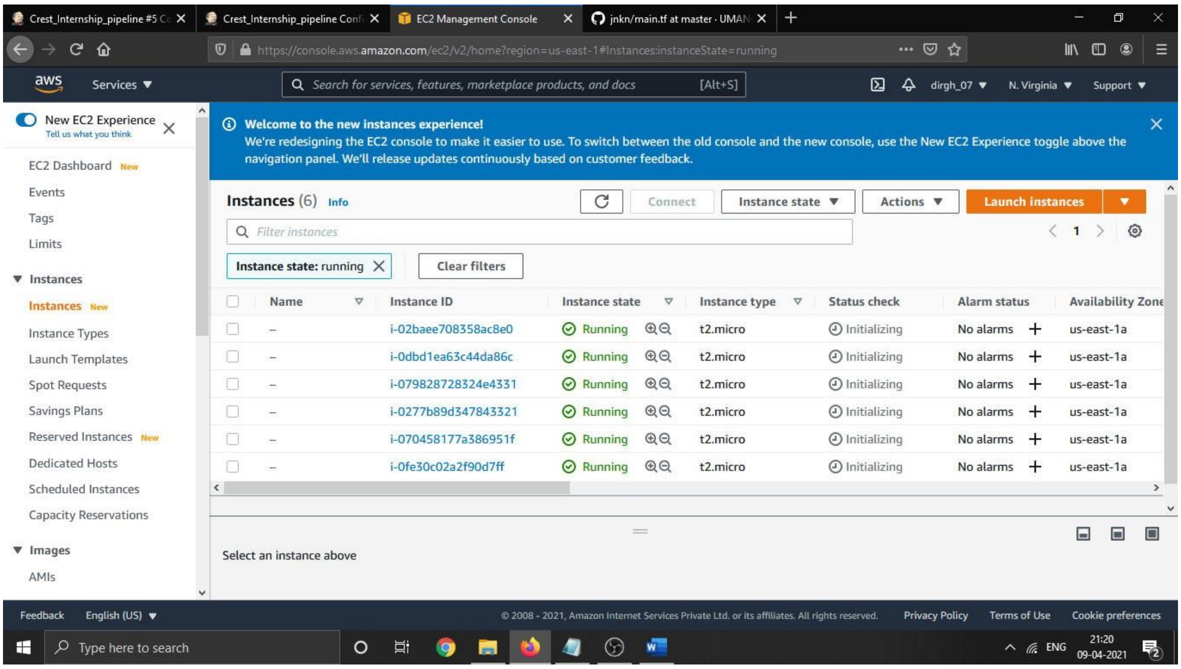


```
[0m[1maws_instance.forwarder: Creating...[0m[0m
[0m[1maws_instance.idx3: Still creating... [1m0s elapsed][0m[0m
[0m[1maws_instance.forwarder: Still creating... [10s elapsed][0m[0m
[0m[1maws_instance.idx3: Still creating... [1m10s elapsed][0m[0m
[0m[1maws_instance.forwarder: Still creating... [20s elapsed][0m[0m
[0m[1maws_instance.idx3: Still creating... [1m20s elapsed][0m[0m
[0m[1maws_instance.forwarder: Still creating... [30s elapsed][0m[0m
[0m[1maws_instance.idx3: Still creating... [1m30s elapsed][0m[0m
[0m[1maws_instance.forwarder: Creation complete after 37s [id=i-070458177a386951f][0m[0m
[0m[1maws_instance.idx3: Creation complete after 1m34s [id=i-0277b89d347843321][0m[0m
[0m[1m[32m
Apply complete! Resources: 6 added, 0 changed, 0 destroyed.[0m
[0m[1m[32m
Outputs:

cm_ip_addr = "18.208.202.124"
forwarder_ip_addr = "3.239.34.46"
idx1_ip_addr = "34.239.240.132"
idx2_ip_addr = "3.239.62.241"
idx3_ip_addr = "100.27.39.3"
sh_ip_addr = "3.239.245.165"[0m
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Crest_Internship_pipeline #5 C
Crest_Internship_pipeline Conf
EC2 Management Console
jnkn/main.tf at master · UMAN
localhost:8080/job/Crest_Internship_pipeline/5/console
Dashboard
Crest_Internship_pipeline
#5
REST API
Jenkins 2.277.2
Type here to search
ENG
21:20
09-04-2021

Figure 9: Installing Splunk



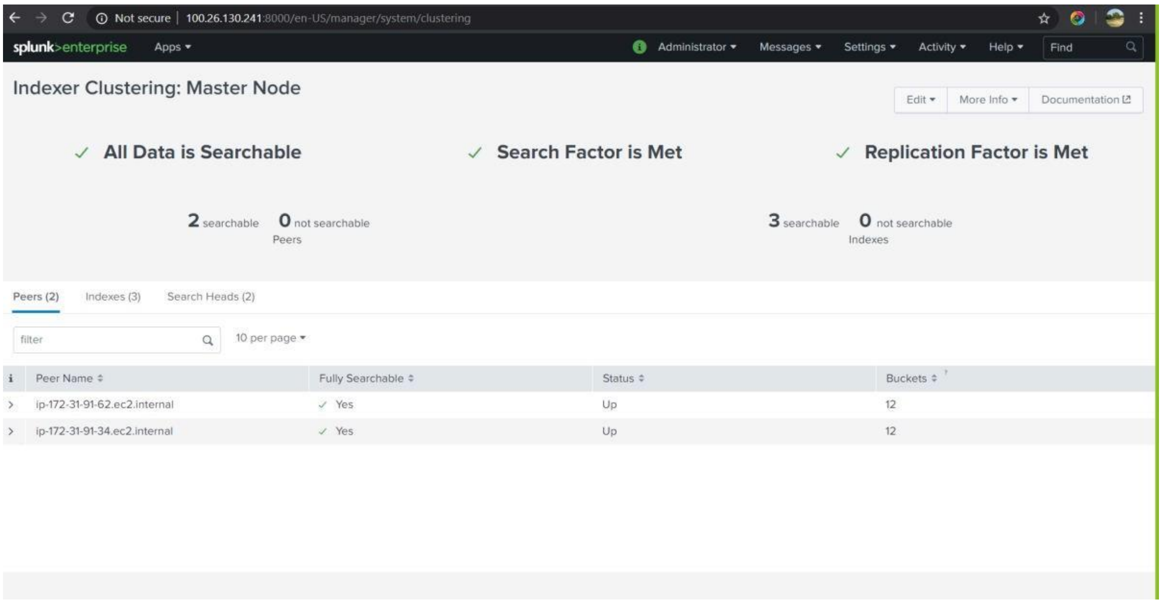
Welcome to the new instances experience!
We're redesigning the EC2 console to make it easier to use. To switch between the old console and the new console, use the New EC2 Experience toggle above the navigation panel. We'll release updates continuously based on customer feedback.

Instances (6) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
–	i-02baee708358ac8e0	Running	t2.micro	Initializing	No alarms	us-east-1a
–	i-0dbd1ea63c44da86c	Running	t2.micro	Initializing	No alarms	us-east-1a
–	i-079828728324e4331	Running	t2.micro	Initializing	No alarms	us-east-1a
–	i-0277b89d347843321	Running	t2.micro	Initializing	No alarms	us-east-1a
–	i-070458177a386951f	Running	t2.micro	Initializing	No alarms	us-east-1a
–	i-0fe30c02a2f90d7ff	Running	t2.micro	Initializing	No alarms	us-east-1a

Select an instance above

Figure 10: Instances on AWS



Indexer Clustering: Master Node

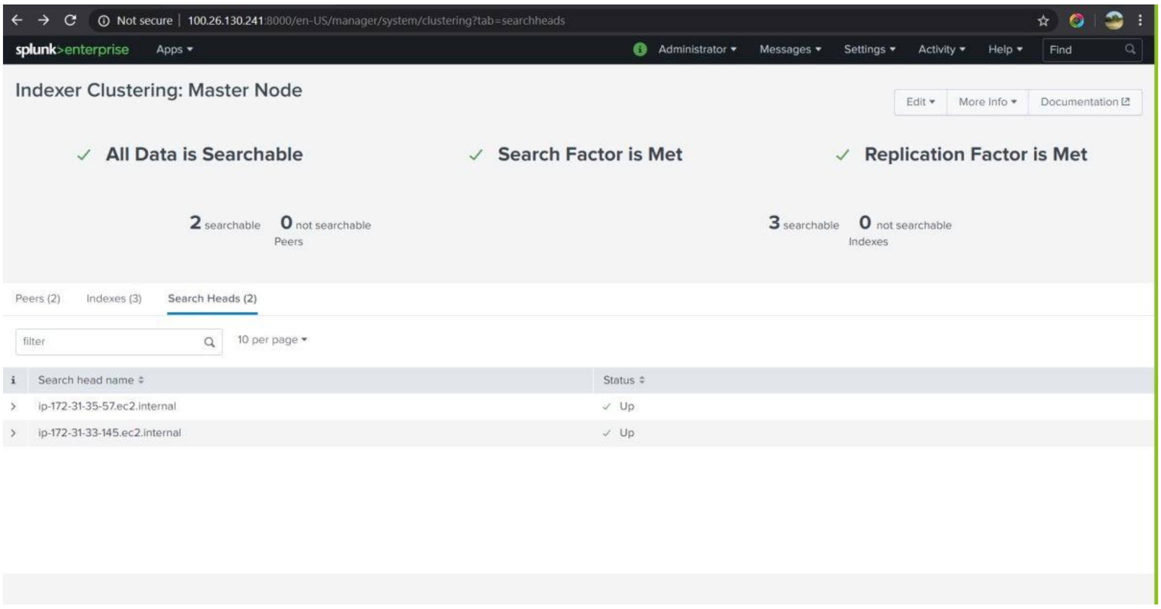
✓ All Data is Searchable ✓ Search Factor is Met ✓ Replication Factor is Met

2 searchable 0 not searchable
Peers

3 searchable 0 not searchable
Indexes

i	Peer Name	Fully Searchable	Status	Buckets
>	ip-172-31-91-62.ec2.internal	✓ Yes	Up	12
>	ip-172-31-91-34.ec2.internal	✓ Yes	Up	12

Figure 11: Cluster Master(Indexer Clustering)



Indexer Clustering: Master Node

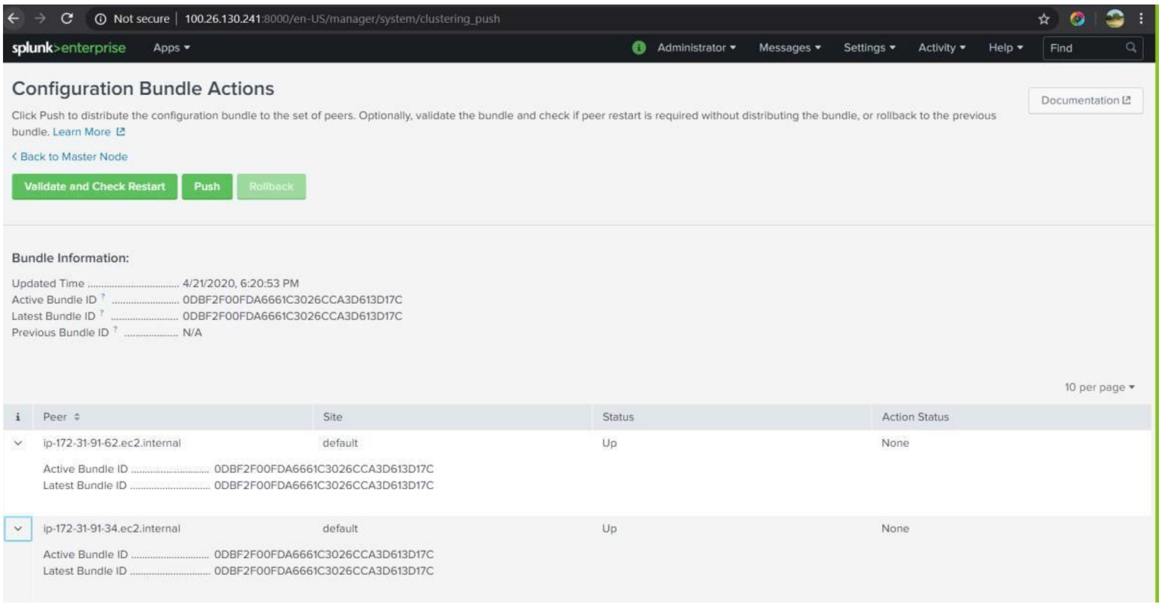
✓ All Data is Searchable ✓ Search Factor is Met ✓ Replication Factor is Met

2 searchable 0 not searchable
Peers

3 searchable 0 not searchable
Indexes

i	Search head name	Status
>	ip-172-31-35-57.ec2.internal	✓ Up
>	ip-172-31-33-145.ec2.internal	✓ Up

Figure 12: Cluster Master (search heads)



Configuration Bundle Actions

Click Push to distribute the configuration bundle to the set of peers. Optionally, validate the bundle and check if peer restart is required without distributing the bundle, or rollback to the previous bundle. Learn More

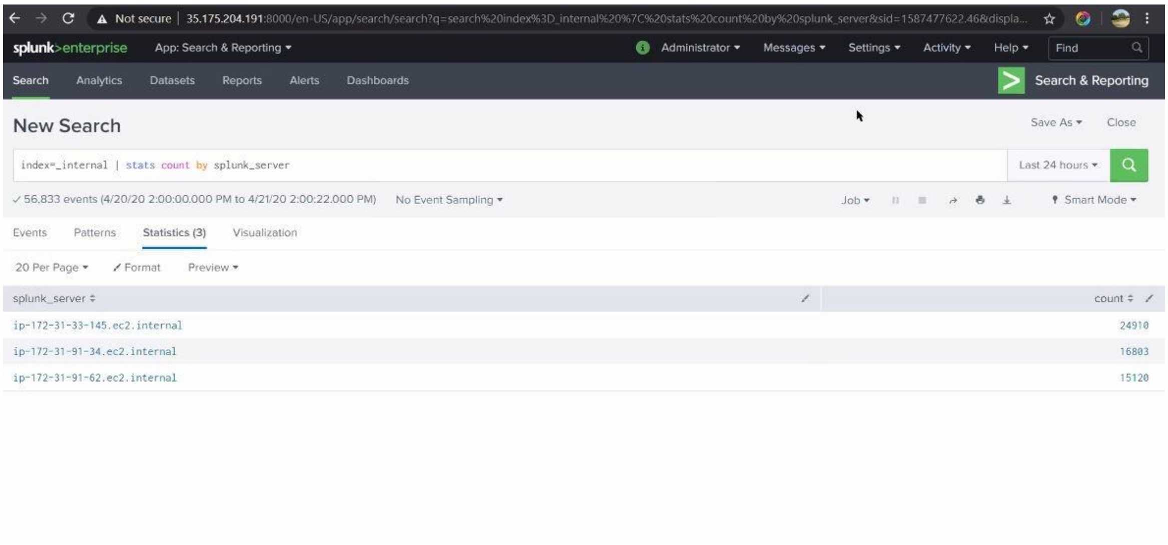
‹ Back to Master Node

Bundle Information:

Updated Time .... 4/21/2020, 6:20:53 PM
Active Bundle ID 0DBF2F00FDA6661C3026CCA3D613D17C
Latest Bundle ID 0DBF2F00FDA6661C3026CCA3D613D17C
Previous Bundle ID .... N/A

i	Peer	Site	Status	Action Status
⌄	ip-172-31-91-62.ec2.internal Active Bundle ID .... 0DBF2F00FDA6661C3026CCA3D613D17C Latest Bundle ID 0DBF2F00FDA6661C3026CCA3D613D17C	default	Up	None
⌄	ip-172-31-91-34.ec2.internal Active Bundle ID .... 0DBF2F00FDA6661C3026CCA3D613D17C Latest Bundle ID 0DBF2F00FDA6661C3026CCA3D613D17C	default	Up	None

Figure 13: Bundle Configurations



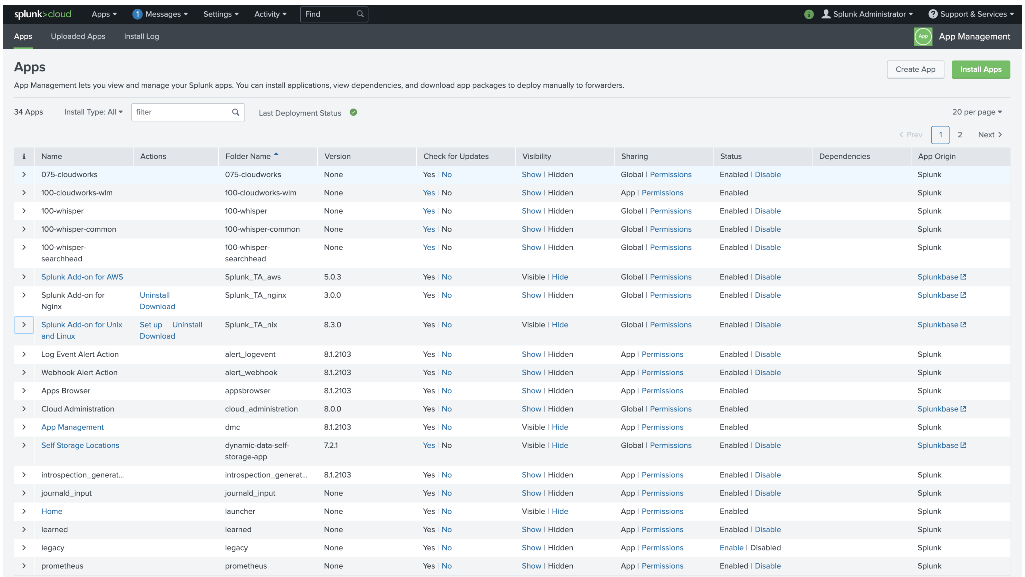
New Search

```
index=_internal | stats count by splunk_server
```

✓ 56,833 events (4/20/20 2:00:00.000 PM to 4/21/20 2:00:22.000 PM) No Event Sampling

splunk_server	count
ip-172-31-33-145.ec2.internal	24910
ip-172-31-91-34.ec2.internal	16803
ip-172-31-91-62.ec2.internal	15120

Figure 14: All Indexers are searchable



Apps

App Management lets you view and manage your Splunk apps. You can install applications, view dependencies, and download app packages to deploy manually to forwarders.

34 Apps

Last Deployment Status ✓

i	Name	Actions	Folder Name	Version	Check for Updates	Visibility	Sharing	Status	Dependencies	App Origin
›	075-cloudworks		075-cloudworks	None	Yes | No	Show | Hidden	Global | Permissions	Enabled | Disable		Splunk
›	100-cloudworks-wlm		100-cloudworks-wlm	None	Yes | No	Show | Hidden	App | Permissions	Enabled		Splunk
›	100-whisper		100-whisper	None	Yes | No	Show | Hidden	Global | Permissions	Enabled | Disable		Splunk
›	100-whisper-common		100-whisper-common	None	Yes | No	Show | Hidden	Global | Permissions	Enabled | Disable		Splunk
›	100-whisper-searchhead		100-whisper-searchhead	None	Yes | No	Show | Hidden	Global | Permissions	Enabled | Disable		Splunk
›	Splunk Add-on for AWS		Splunk_TA_aws	5.0.3	Yes | No	Visible | Hide	Global | Permissions	Enabled | Disable		Splunkbase
›	Splunk Add-on for Nginx	Uninstall Download	Splunk_TA_nginx	3.0.0	Yes | No	Show | Hidden	Global | Permissions	Enabled | Disable		Splunkbase
›	Splunk Add-on for Unix and Linux	Set up Uninstall Download	Splunk_TA_nix	8.3.0	Yes | No	Visible | Hide	Global | Permissions	Enabled | Disable		Splunkbase
›	Log Event Alert Action		alert_logevent	8.1.2103	Yes | No	Show | Hidden	App | Permissions	Enabled | Disable		Splunk
›	Webhook Alert Action		alert_webhook	8.1.2103	Yes | No	Show | Hidden	App | Permissions	Enabled | Disable		Splunk
›	Apps Browser		appsbrowser	8.1.2103	Yes | No	Show | Hidden	App | Permissions	Enabled		Splunk
›	Cloud Administration		cloud_administration	8.0.0	Yes | No	Show | Hidden	Global | Permissions	Enabled		Splunkbase
›	App Management		dmc	8.1.2103	Yes | No	Visible | Hide	App | Permissions	Enabled		Splunk
›	Self Storage Locations		dynamic-data-self-storage-app	7.2.1	Yes | No	Visible | Hide	Global | Permissions	Enabled | Disable		Splunkbase
›	introspection_generat...		introspection_generat...	8.1.2103	Yes | No	Show | Hidden	App | Permissions	Enabled | Disable		Splunk
›	journald_input		journald_input	None	Yes | No	Show | Hidden	App | Permissions	Enabled | Disable		Splunk
›	Home		launcher	None	Yes | No	Visible | Hide	App | Permissions	Enabled		Splunk
›	learned		learned	None	Yes | No	Show | Hidden	App | Permissions	Enabled | Disable		Splunk
›	legacy		legacy	None	Yes | No	Show | Hidden	App | Permissions	Enable | Disabled		Splunk
›	prometheus		prometheus	None	Yes | No	Show | Hidden	App | Permissions	Enabled | Disable		Splunk

7. TESTING

7.1 TESTING PLAN

The testing technique that is going to be used in the project is black box testing, that is only the outputs are checked when we give a expected input. The testing sub-process includes the following activities in a phase dependent manner:

a) Create Test Plans.

b) Create test Specifications.

c) Review Test Plans and Test Specifications

d) Conduct tests according to the Test Specifications

e) Fix defects, if any present in project.

f) When defects are fixed continue from activity.

7.2 TEST TESTING STRATEGY

The development process repeats this testing sub-process a number of times for the following steps:

- Unit Testing
- Integration Testing

Unit Testing tests a unit of code (module or program) after coding of that unit is completed. Integration Testing verify whether the various programs that make up a system, interface with each other as desired, adapt and if the interfaces between the programs are correct. Testing the system ensures that the system meets its stated design specifications. The acceptance Testing is testing by the users to ascertain whether the system developed is a correct implementation of the SRS.

Tests are performed hierarchically to ensure that each component is correct and the assembly/combination of components is correct. Simply testing a entire system would eventually most likely to generate component errors that would be very expensive to trace and fix. We ran both Unit Testing and System Testing to detect and fix errors.

7.3 TEST CASES

In all these cases, we have checked that the system is working with all types of options. In this system the testing includes, starting the script and the expected task should properly be done.

Test Case ID	Test Steps	Expected Result	Actual Result	Pass/Fail
T1	Jenkins run	Provision instances and Splunk install and clustering without failure	As Expected	Pass
T2	AWS instances Security Groups	Security groups should be created as defined under specified AWS account	As Expected	Pass
T3	Deploying App	App deployment on instances and all packages should be downloaded	As Expected	Pass
T4	Replicate Data to all indexers	SF/RF should Met and No Fix-up Tasks	As Expected	Pass
T5	App Data searchable	Deployed App should be searchable from UI and monitored using search head	As Expected	Pass

8. LIMITATIONS AND FUTURE ENHANCEMENT

8.1 LIMITATIONS

- Cluster Master has Public IP.
- No Static IP address so no such track of IPs.
- Security vulnerabilities may occur as IP is random.
- SF/RF take more time to met

8.2 FUTURE ENHANCEMENTS

- It can be enhanced by including other automation tools like Puppet and Synet which can help with Splunk.
- Bastion hosts can be created.
- Monitoring system software like Nagios can be configured to check the health status of infrastructure.
- Auto-scaling groups can be configured for cluster architecture so that we can balance the data which is ingested in architecture.
- GUI interface can be created for performing each task.

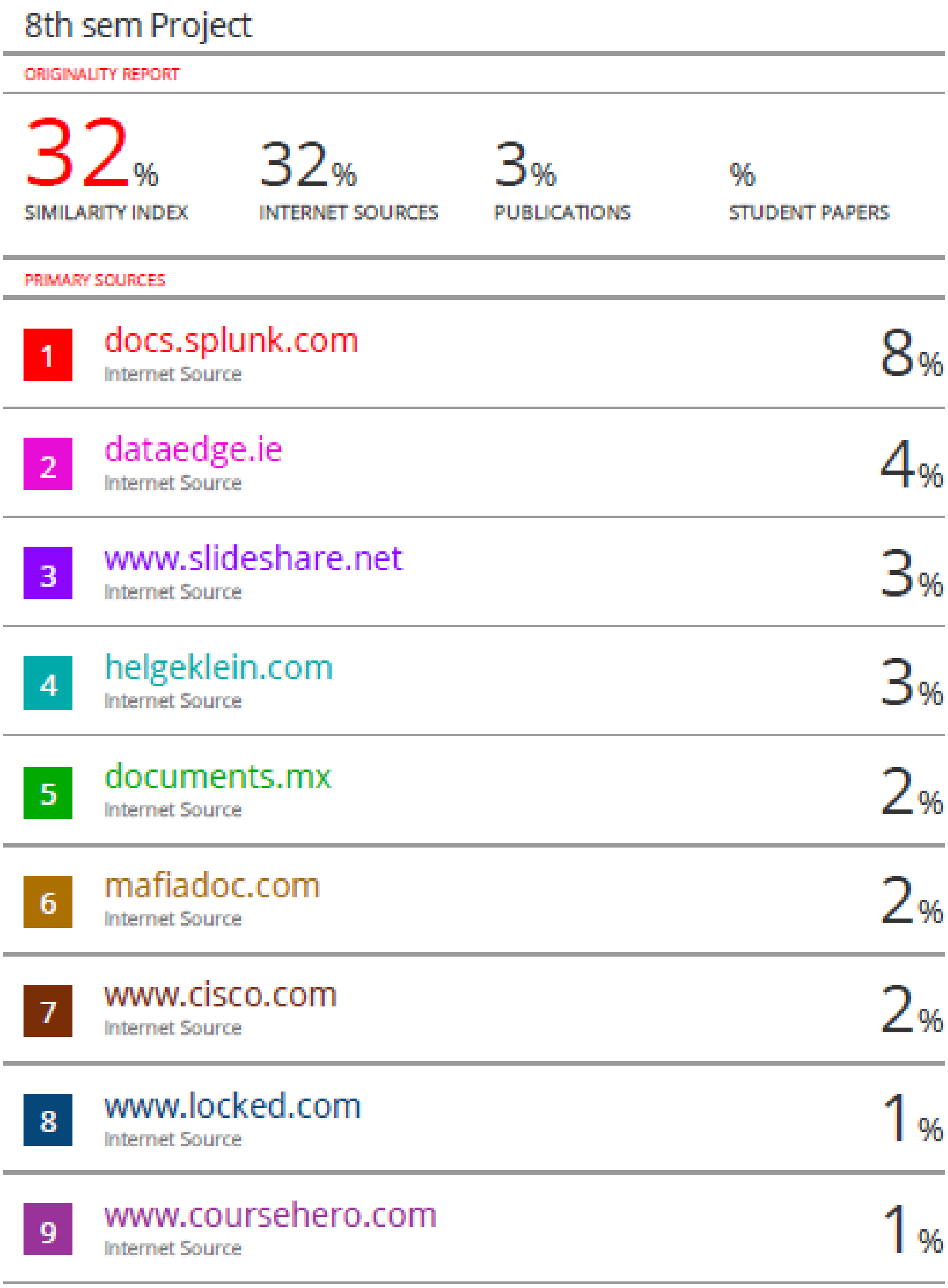
9. CONCLUSION

The proposed system will work efficiently for deploying the Splunk indexer clustered architecture on AWS using Jenkins/Ansible/Terraform. It mainly focuses on reducing the manual efforts and make automation for managing the whole Splunk architecture with creating automated architecture and installing Splunk App on it which becomes use for investing the logs and more.

Using an automation script is a better approach rather than manually configuring all things because manual tasks may tend to make some mistakes and have to use the same methodology when there is same requirement in future. But if we have script in hand we just need to pull a trigger and whole setup is up and running.

BIBLIOGRAPHY:

- **Splunk Documentation:** https://docs.splunk.com/Documentation
- **Splunk troubleshooting:** https://answers.splunk.com/
- **Ansible Documentation:** http://docs.ansible.com/
- **AWS Documentation:** https://aws.amazon.com/documentation/
- **Terraform Documentation:** https://www.hashicorp.com/
- **Jenkins Documentation:** https://jenkins.io/doc/



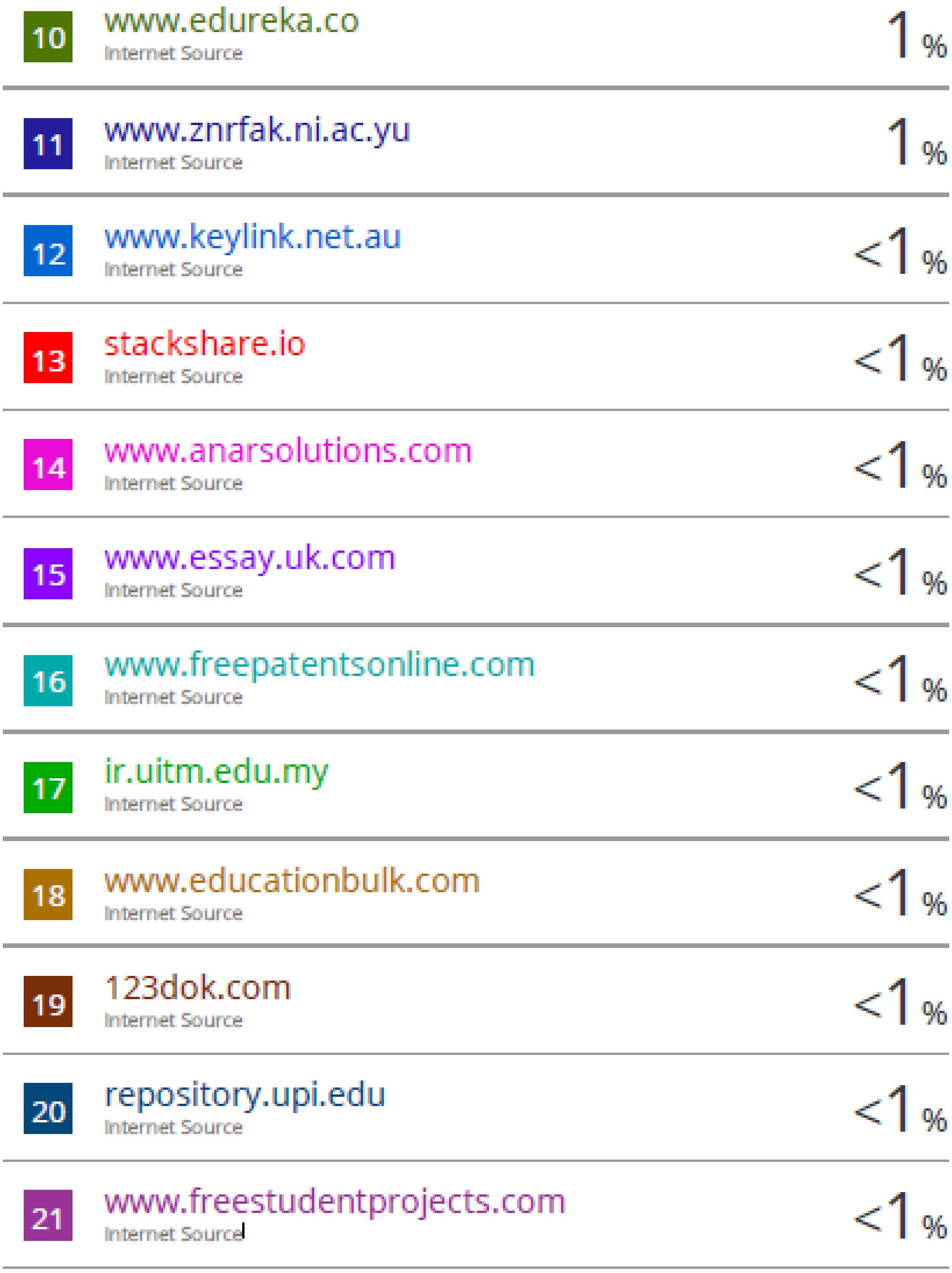
8th sem Project

ORIGINALITY REPORT

32%	32%	3%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	docs.splunk.com Internet Source	8%
2	dataedge.ie Internet Source	4%
3	www.slideshare.net Internet Source	3%
4	helgeklein.com Internet Source	3%
5	documents.mx Internet Source	2%
6	mafiadoc.com Internet Source	2%
7	www.cisco.com Internet Source	2%
8	www.locked.com Internet Source	1%
9	www.coursehero.com Internet Source	1%



10	www.edureka.co Internet Source	1 %
11	www.znrfak.ni.ac.yu Internet Source	1 %
12	www.keylink.net.au Internet Source	<1 %
13	stackshare.io Internet Source	<1 %
14	www.anarsolutions.com Internet Source	<1 %
15	www.essay.uk.com Internet Source	<1 %
16	www.freepatentsonline.com Internet Source	<1 %
17	ir.uitm.edu.my Internet Source	<1 %
18	www.educationbulk.com Internet Source	<1 %
19	123dok.com Internet Source	<1 %
20	repository.upi.edu Internet Source	<1 %
21	www.freestudentprojects.com Internet Source	<1 %

Signature Certificate

Document Ref.: RHJHE-QLK4R-MV7YZ-2ZDNR

Document signed by:

Dhruval Sharma

Verified E-mail:
dhruval.sharma@crestdatasys.com

IP: 103.143.30.2 Date: 27 Apr 2021 06:49:15 UTC

Dhruval Sharma

Parth Shah

Verified E-mail:
parthshah.ce@charusat.ac.in

IP: 117.239.83.193 Date: 27 Apr 2021 06:56:05 UTC

Shreya Palejkar

Verified E-mail:
shreya.palejkar@crestdatasys.com

IP: 103.143.30.2 Date: 27 Apr 2021 08:55:22 UTC

Shreya Palejkar

Sandip Patel

Verified E-mail:
sandippatel.it@charusat.ac.in

IP: 103.39.130.8 Date: 29 Apr 2021 15:53:36 UTC

Sandip Patel

Document completed by all parties on:
29 Apr 2021 15:53:36 UTC