



**INSTITUTE FOR ADVANCED
COMPUTING AND
SOFTWARE DEVELOPMENT
AKURDI, PUNE**

DOCUMENTATION ON

**“SenitelWeb : Multi layered security
approach CI/CD Pipeline with
Web application deployment”**

PG-DITISS March-2023

SUBMITTED BY:

GROUP NO: 16

OJAS JAWALE (233426)

OMKAR GADRE (233427)

**MR. KARTIK AWARI
PROJECT GUIDE**

**MR. ROHIT PURANIK
CENTRE CO-ORDINATOR**

ABSTRACT

As web applications play an increasingly pivotal role in modern business operations, the need for swift, reliable, and secure deployment has become paramount. Continuous Integration and Continuous Deployment (CICD) pipelines have emerged as a critical solution to achieve seamless application delivery. However, the surge in cyber threats demands a fortified security approach to protect sensitive data and ensure the integrity of deployed applications. This paper presents a comprehensive abstract for a "Web Application Deployment with CICD Pipeline Using a Multilayered Security Approach," outlining a systematic and robust strategy to harmonize deployment speed with security resilience.

The proposed approach revolves around the integration of a multilayered security framework within the CICD pipeline, catering to every phase of application development, testing, and deployment. Beginning with access controls and user authentication, the foundational layer establishes a strong barrier against unauthorized access to the pipeline and its components.

Moving up the layers, advanced code analysis tools conduct both static and dynamic examinations of the application's source code. This proactive assessment detects vulnerabilities and weak points, empowering developers to rectify issues before they escalate. A subsequent layer implements runtime security mechanisms, such as web application firewalls and intrusion detection systems, which safeguard against attacks during the deployment process..

The proposed "SenitelWeb : Multi layered security approach CICD Pipeline Web application deployment" seamlessly merges swiftness with security. By orchestrating an array of security layers, from access control to runtime protection, supply chain validation, automated security testing, and behavioral analysis, the approach ensures that web applications are fortified against a diverse range of threats. As organizations navigate the digital frontier, this approach offers a vital shield, ensuring that web application deployment remains a secure and resilient endeavor.

TABLE OF CONTENTS

Topics	Page No.
ABSTRACT	
LIST OF ABBREVIATIONS	
LIST OF FIGURES	
LIST OF TABLES	
1. INTRODUCTION	1
1.1 Problem Statement	1
2. LITERATURE SURVEY	2
3. METHODOLOGY	
3.1 System Architecture	3
4. REQUIREMENT SPECIFICATION	
4.1 Hardware Requirement	
4.2 Software Requirement	4
5. WORKING	5
6. IMPLEMENTATION	8
7. APPLICATIONS	23
8. ADVANTAGES & DISADVANTAGES	24
9. CONCLUSION	25
10. REFERENCES	26

LIST OF ABBREVIATIONS

Sr. No.	Abbreviation	Full-Form
1.	AWS	Amazon Web Services
2.	EC2	Elastic Compute Cloud
3.	SSH	Secure Shell
4.	SCP	Secure Copy
5.	CI	Continuous Integration
6.	CD	Continuous Deployment
7.	GIT	Global Information Tracker
8.	HTTPS	Secure Hyper Text Transfer Protocol
9.	SNS	Simple Notification Service
10.	NLB	Network Load Balancing

LIST OF FIGURES

Figure No.	Figure Name	Page No.
Figure 1.	Architecture Diagram	4
Figure 2.	Working Diagram	6

LIST OF TABLES

Table No.	Table Name	Page No.
Table 1.	Author vs Publisher	3

1. INTRODUCTION

In today's fast-paced digital landscape, the security of web applications and the efficiency of their deployment pipelines have become paramount concerns for businesses and organizations. The project "**SenitelWeb**" addresses these concerns by implementing a comprehensive multi-layered security approach for Continuous Integration/Continuous Deployment (CI/CD) pipelines and the subsequent deployment of web applications. By integrating security measures at various stages of the CI/CD pipeline and the application deployment process, SenitelWeb aims to provide a robust defense against potential cyber threats and vulnerabilities.

The increasing adoption of Continuous Integration and Continuous Deployment (CI/CD) pipelines has transformed software development by enabling rapid and automated code integration, testing, and deployment. However, as these pipelines operate within the TCP/IP networking model, they inherit the security challenges associated with the interconnected nature of modern applications. This paper proposes a comprehensive approach to implementing multilayered security within the TCP/IP model for CI/CD pipelines.

1.1. PROBLEM STATEMENT

In the modern digital landscape, the rapid development and deployment of web applications have become essential for businesses to remain competitive. However, this speed and agility often come at the cost of security vulnerabilities that can be exploited by malicious actors. The project "**SenitelWeb**" addresses the critical issue of inadequate security measures in Continuous Integration/Continuous Deployment (CI/CD) pipelines and the subsequent deployment of web applications.

The existing approach to CI/CD pipelines and web application deployment lacks a comprehensive security framework, resulting in several challenges:

1. **Security as an Afterthought:** In many development processes, security considerations are often introduced after the development is well underway. This reactive approach can lead to vulnerabilities being identified late in the cycle, increasing the risk of successful cyber attacks.
2. **Vulnerability Propagation:** Traditional CI/CD pipelines may unknowingly propagate security vulnerabilities from development to deployment stages due to limited security checks during the pipeline process.

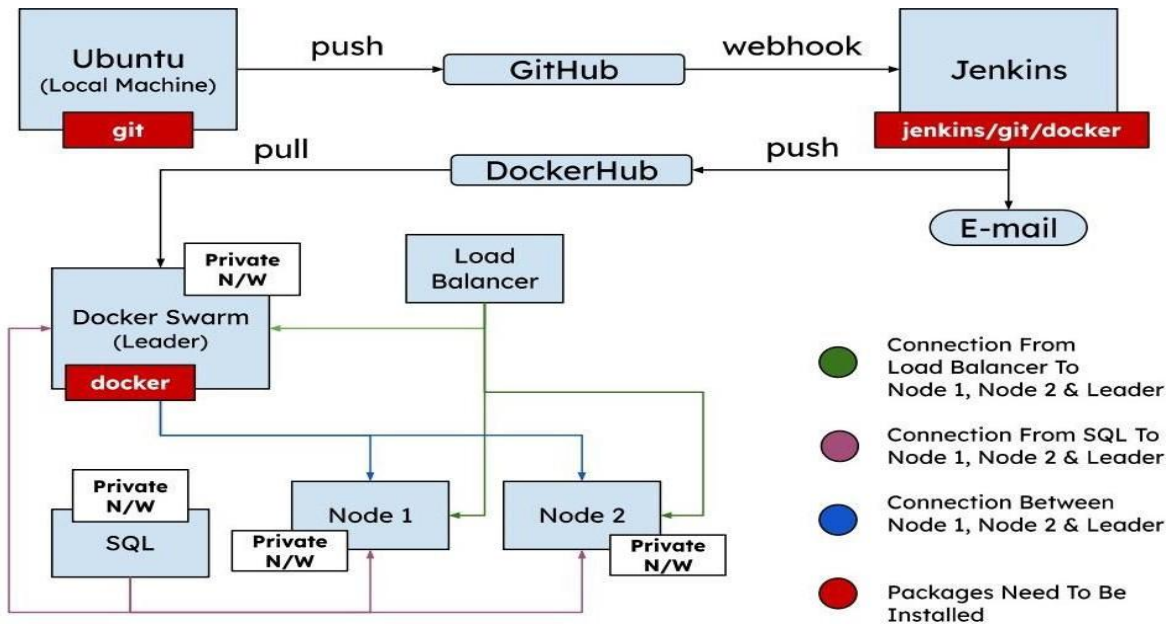
2. LITERATURE SURVEY

The deployment of web applications through Continuous Integration and Continuous Deployment (CICD) pipelines has become integral to modern software development practices. However, the increasing frequency and complexity of cyber threats demand a comprehensive security approach to ensure the integrity and confidentiality of these applications. The concept of a multilayered security approach has gained prominence as a strategy to address these challenges effectively. This literature review examines key research and resources that explore the implementation of a multilayered security approach for web application deployment within CICD pipelines

Title	Author	Published by
Securing Continuous Delivery Pipelines in DevOps	Ericsson, et al.	ACM SIGSAC Conference
Security in DevOps and Continuous Deployment	Melanie Rieback	arXiv preprint arXiv:1802.08785
Application Security in the DevOps World	NCC Group	NCC Group
Cybersecurity for DevOps	Jon Hawes, et al.	Apress
Securing Microservices in Continuous Deployment	Prakriti Trivedi, et al.	IEEE International Conference
Continuous Integration, Delivery and Deployment	Rafal Leszko	Packt Publishing

3. METHODOLOGY

3.1 SYSTEM ARCHITECTURE



This approach involves implementing multiple layers of security measures within a Continuous Integration/Continuous Deployment (CI/CD) pipeline. This pipeline automates the process of building, testing, and deploying web applications while incorporating various security checks at each stage. By doing so, it aims to identify and mitigate vulnerabilities and threats, ensuring that the deployed web application is resilient to potential attacks and breaches.

4. REQUIREMENT SPECIFICATION

4.1 HARDWARE REQUIREMENTS

Based on the hardware requirement need to configure every instance of infrastructure with requirement like;

1. Instance type: t2.micro
2. RAM: minimum 4GB
3. CPU: 2.5 GHZ

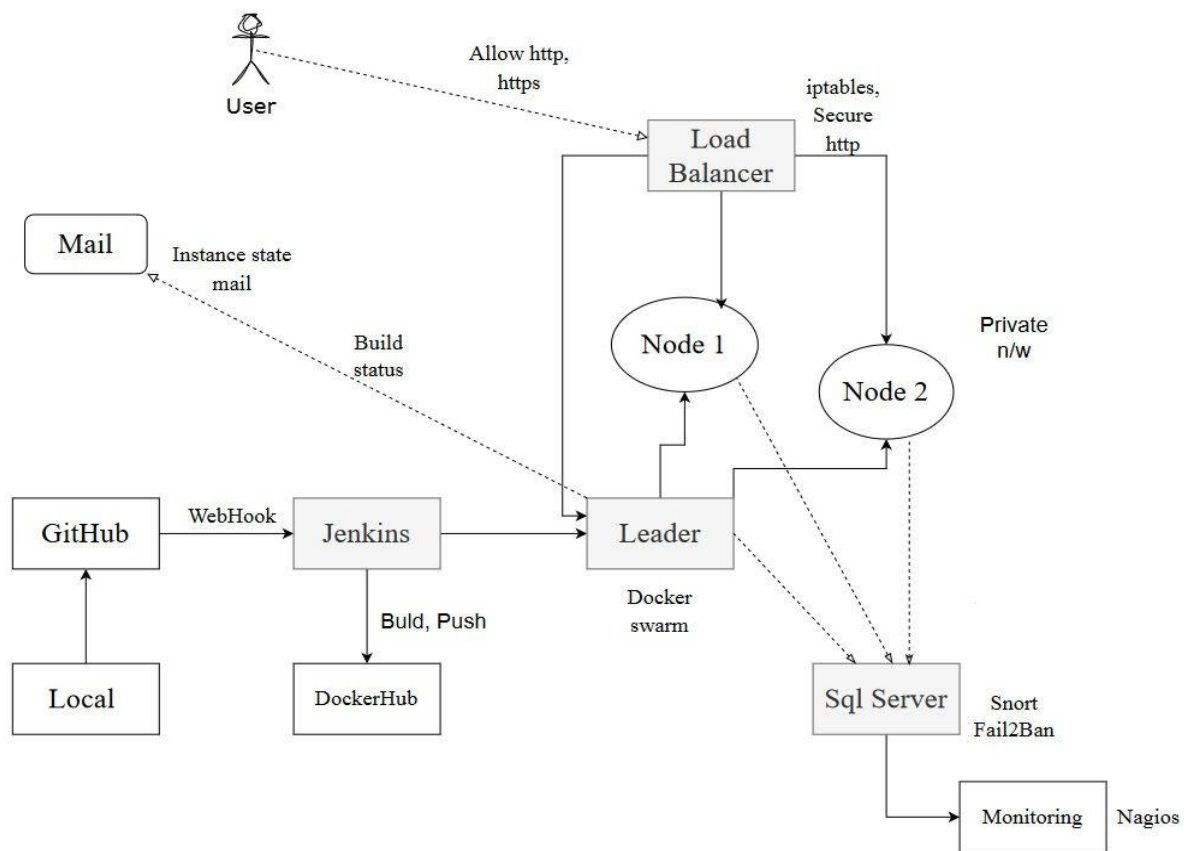
4.2 SOFTWARE REQUIREMENTS

1. Operating system: Debian and Ubuntu
2. Firewall: IPtables
3. Platform: Amazon Web Service
4. CI Platform : Jenkins
5. Database: MariaDB

In our project, we use **Debian and Ubuntu** Version. And other tools used are;

1. Git
2. Snort
3. Nagios

5. WORKING



The increasing adoption of Continuous Integration and Continuous Deployment (CI/CD) pipelines has transformed software development by enabling rapid and automated code integration, testing, and deployment.

AWS:

Amazon Web Services (AWS) is a comprehensive and widely used cloud computing platform provided by Amazon. It offers a vast array of cloud services that enable businesses and individuals to access and utilize computing resources over the internet on a pay-as-you-go basis. AWS provides a scalable and flexible environment to build, deploy, and manage applications and services without the need to invest in and manage physical hardware.

Ubuntu:

Ubuntu is a popular open-source operating system based on the Debian distribution of Linux. It's known for its ease of use, user-friendly interface, and strong community support. Ubuntu is designed to provide a versatile platform for both desktop and server environments.

GIT:

Git is a widely used distributed version control system (VCS) that enables developers to collaborate on software development projects efficiently and track changes to source code over time. It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel, and it has since become an essential tool for software development teams.

Jenkins:

Jenkins is an open-source automation server that facilitates continuous integration (CI) and continuous delivery (CD) processes in software development. It was created to automate the building, testing, and deployment of software projects, helping teams streamline development workflows and ensure code quality.

Docker :

Docker is an open-source platform that enables developers to create, deploy, and manage applications and their dependencies in lightweight, isolated containers. Containers package software and its dependencies into a consistent environment that can run reliably across different environments, from development to production.

Docker Hub :

Docker Hub is a cloud-based registry service provided by Docker that serves as a central repository for Docker container images. It offers a platform for developers, teams, and organizations to store, share, and distribute container images, making it easier to collaborate on software development and streamline deployment workflows.

Docker Swarm:

Its a native clustering and orchestration solution for Docker, designed to manage a group of Docker containers and services as a single unit. It simplifies the deployment, scaling, and management of containerized applications by providing features for load balancing, high availability, and service discovery.

Squid :

Squid is an open-source caching and forwarding HTTP proxy server that is widely used to improve the performance, efficiency, and security of web browsing. It serves as an intermediary between clients (users' web browsers) and web servers, intercepting and caching web content to reduce bandwidth usage and accelerate page load times.

HTTPS:

Docker Hub is a cloud-based registry service provided by Docker that serves as a central repository for Docker container images. It offers a platform for developers, teams, and organizations to store, share, and distribute container images, making it easier to collaborate on software development and streamline deployment workflows.

Snort :

Snort is an open-source, signature-based intrusion detection and prevention system (IDS/IPS) that monitors network traffic for suspicious activities and attempts to detect and prevent intrusions or attacks. It was developed by Martin Roesch in the late 1990s and has since become one of the most widely used intrusion detection tools.

Nagios :

In the complex and dynamic world of IT infrastructure and systems management, ensuring the availability, performance, and health of various components is paramount. Nagios, a widely recognized open-source monitoring and alerting solution, plays a crucial role in this domain. It empowers organizations to proactively monitor their infrastructure, detect anomalies, and respond swiftly to potential issues before they impact critical services.

Iptables :

Iptables is a command-line utility for managing packet filtering and network address translation (NAT) rules in a Linux-based operating system. It is a core component of the Linux kernel's netfilter framework, which enables administrators to control network traffic, implement security measures, and configure network-related functions.

VPC :

Amazon Virtual Private Cloud (Amazon VPC) is a cloud computing service provided by Amazon Web Services (AWS) that allows users to create isolated, private network environments within the AWS cloud. VPC enables users to define their own virtual network topology, configure network settings, and launch resources like Amazon EC2 instances in a secure and controlled manner.

6. IMPLEMENTATION

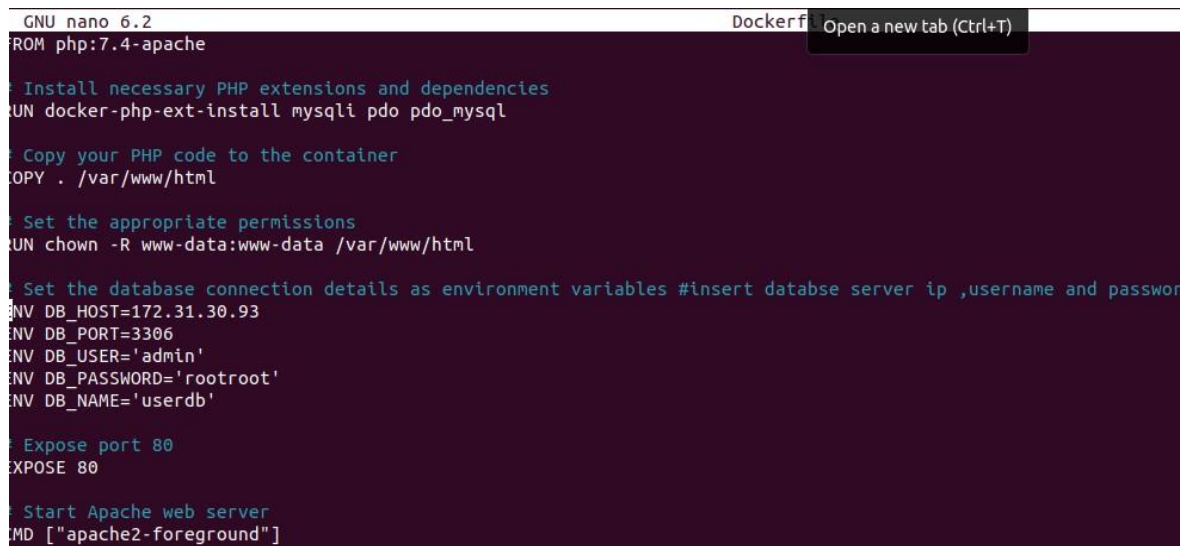
Basic requirements-

Deploy local machine (ubuntu) + Launch all instance ensure all are in same subnet in our case it is us -east 1a .

- 1st instance : Jenkins
Default vpc / Subnet us -east 1a /Enable public ip
- 2nd instance : Docker swarm leader
Default vpc /Subnet us -east 1a /Disbale public ip
- 3rd instance : Docker swarm node 1
Default vpc /Subnet us-east 1a/Disable public ip
- 4th instance : Docker swarm node 2
Default vpc/Subnet us-east 1a/Disable public ip
- 5th instance : Sql server
Default vpc/ Subnet us -east 1/Disable public ip
- 6th instance : load balancer
Default vpc /Subnet us-east 1/Enable public ip

Implementation-

-> Dockerfile



```
GNU nano 6.2 Dockerfile
FROM php:7.4-apache

# Install necessary PHP extensions and dependencies
RUN docker-php-ext-install mysqli pdo pdo_mysql

# Copy your PHP code to the container
COPY . /var/www/html

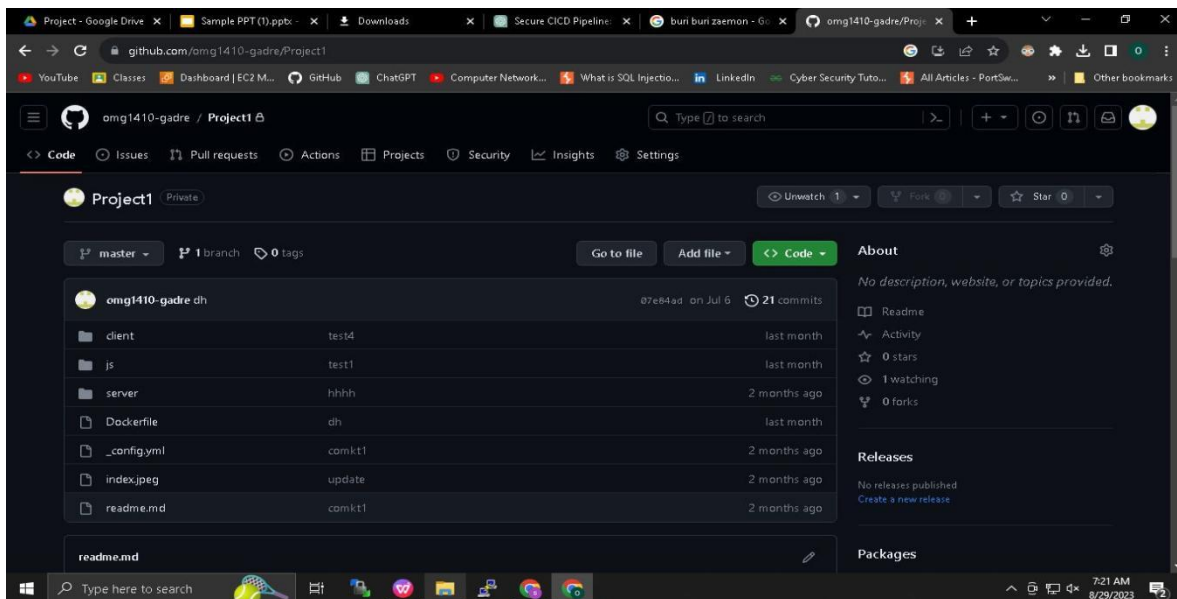
# Set the appropriate permissions
RUN chown -R www-data:www-data /var/www/html

# Set the database connection details as environment variables #insert databse server ip ,username and password
ENV DB_HOST=172.31.30.93
ENV DB_PORT=3306
ENV DB_USER='admin'
ENV DB_PASSWORD='rootroot'
ENV DB_NAME='userdb'

# Expose port 80
EXPOSE 80

# Start Apache web server
CMD ["apache2-foreground"]
```

Push all source code to GitHub private repository-



Generate GitHub token to create credentials in Jenkins-

Add global credentials on Jenkins-

- Add Webhook trigger
- Go to Jenkins -> credentials -> add credentials

new credentials

Kind
Username with password

Scope
Global (Jenkins, nodes, items, all child items, etc)

Username
omg1410-gadre

☐ Treat username as secret

Password

ID
demo

Add Docker Hub credentials-

Secret values are masked on a best-effort basis to prevent *accidental* disclosure. Multiline secrets, such as the contents of a SSH private key file, are not masked. See the inline usage guidelines.

Bindings

Username and password (separated)

Username Variable

USR

Password Variable

PASS

Credentials

omg1410/***** (for docker hub login)

Add

Add

Generate Pipeline Script

```
withCredentials([usernamePassword(credentialsId: 'docker_hub', passwordVariable: 'PASS', usernameVariable: 'USR')]) {
// some block
}
```

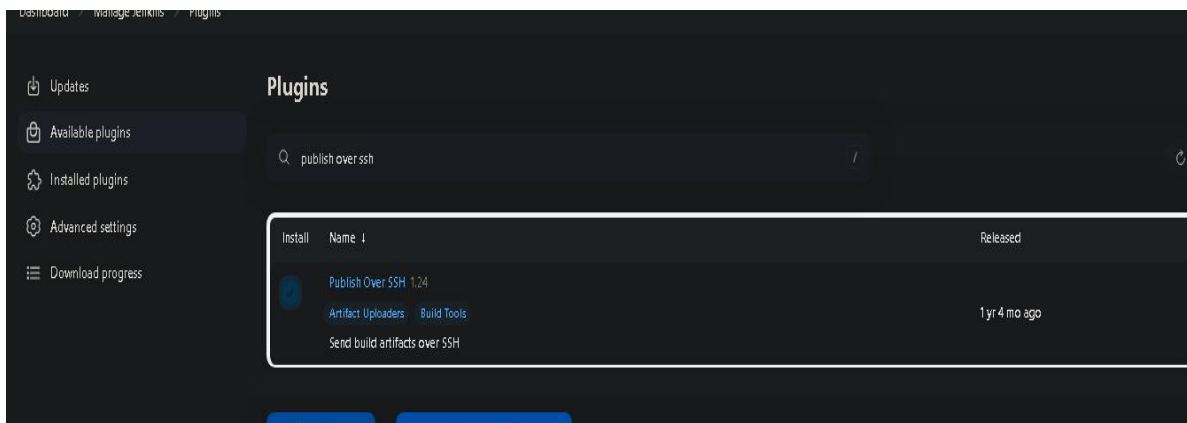
Script-

```
pipeline {
    agent any
    stages {
        stage('git checkout ') {
            steps {
                git credentialsId: 'Jenkins_Token', url: 'https://github.com/omg1410-gadre/Project1.git'
                echo 'git login'
            }
        }
        stage('docker build image ') {
            steps {
                sh 'docker build -t project1 .'
                echo 'Build image completed'
            }
        }
        stage('docker login') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker_hub', passwordVariable: 'PASS', usernameVariable: 'USR')]) {
                    echo 'docker login complete '
                }
            }
        }
    }
}
```

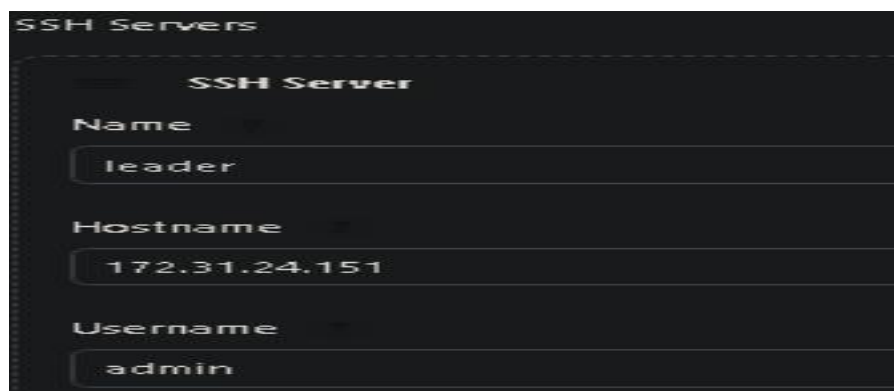
```
stage('docker push image') {
    steps {
        sh 'docker push omg1410/project1:latest'
        echo 'docker push complete '
    }
}
stage('jenkins login to leader') {
    steps {
        sh 'ssh -i /home/admin/leader.pem admin@172.31.24.151'
        echo 'jenkins login to leader complete '
    }
}
```

```
stage('Service creation') {
    stage('Service creation') {
        steps {
            sshPublisher(publishers: [sshPublisherDesc(configName: 'leader'
                , transfers: [sshTransfer(cleanRemote: false, excludes: '',
                    execCommand: '''sudo docker login -u omg1410 -p Omkar1410@
                    sudo docker service rm sir
                    sudo docker rmi omg1410/project1
                    sudo docker pull omg1410/project1:latest
                    sudo docker service create --name sir -p 1314:80 --replicas 3
                    omg1410/project1:latest''', execTimeout: 120000, flatten:
                    false, makeEmptyDirs: false, noDefaultExcludes: false,
                    patternSeparator: '[, ]+', remoteDirectory: '',
                    remoteDirectorySDF: false, removePrefix: '', sourceFiles:
                    ''')], usePromotionTimestamp: false, useWorkspaceInPromotion
                    : false, verbose: false)])
            echo 'service created'
        }
    }
}
```

Publish over ssh-



Manage jenkins -> system -> publish over ssh section-



Build Successful-

	Git login	docker build image	docker login	docker push image	jenkins login to leader	Service creation	Email alert
Average stage times: Average full run time: ~39s)	807ms	18s	200ms	3s	582ms	12s	7s
No Changes	827ms	17s	131ms	3s	576ms	8s	1s

Creation of load balancer-

- sudo apt-get install squid
- sudo nano /etc/squid/squid.conf

```

acl localnet src 192.168.0.0/16      # RFC 1918 local private network (LAN)
acl localnet src fc00::/7          # RFC 4193 local private network range
acl localnet src fe80::/10         # RFC 4291 link-local (directly plugged) machines
acl project_users dstdomain 54.234.86.153
acl SSL_ports port 443
acl Safe_ports port 80             # http

```

```

# Adapt localnet in the ACL section to IIS
# from where browsing should be allowed
#http_access allow localnet
http_access allow localhost
http_access allow project_users

```

```

# Squid normally listens to port 3128
http_port 3128
http_port 80 vhost
http_port 1314 vhost

```

```

admin@ip-172-31-17-170: /etc/squid
GNU nano 5.4 squid.conf
# proxy-only      objects fetched from the peer will not be stored locally.
#
#Default:
# none
cache_peer 172.31.24.151 parent 1314 0 no-query originserver round-robin weight=1 name=one
cache_peer 172.31.20.193 parent 1314 0 no-query originserver round-robin weight=1 name=two
cache_peer 172.31.21.89 parent 1314 0 no-query originserver round-robin weight=1 name=three
# TAG: cache_peer_access
# Restricts usage of cache_peer proxies.

```

Setting up SQL server-

- sudo apt-get update
 - sudo apt-get install mariadb-server
 - cd /etc/mysql/mariadb.conf.d
 - sudo nano 50-server.conf
- Bind-address = ip of database server I.e own ip

```

admin@ip-172-31-30-93: /etc/mysql/mariadb.conf.d
GNU nano 5.4 50-server.cnf
# * Basic Settings
#
user                        = mysql
pid-file                    = /run/mysqld/mysqld.pid
basedir                    = /usr
datadir                    = /var/lib/mysql
tmpdir                     = /tmp
lc-messages-dir            = /usr/share/mysql
lc-messages                 = en_US
skip-external-locking
# Broken reverse DNS slows down connections considerably and name resolu
# safe to skip if there are no "host by domain name" access grants
#skip-name-resolve
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address                = 172.31.30.93

```

Grant access to nodes and Load balancer-

- > CREATE DATABASE userdb;
- > CREATE USER 'admin'@'pvt ip of leader' IDENTIFIED BY 'password';
- > CREATE USER 'admin'@'pvt ip of node 1' IDENTIFIED BY 'password';
- > CREATE USER 'admin'@'pvt ip of node 2' IDENTIFIED BY 'password';
- > SHOW GRANTS;
- > GRANT ALL ON *.* TO 'admin'@'pvt ip of leader';
- > GRANT ALL ON *.* TO 'admin'@'pvt ip node 1';
- > GRANT ALL ON *.* TO 'admin'@'pvt ip of node 2';

```

Aborted
admin@ip-172-31-30-93:/etc/mysql/mariadb.conf.d$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 33
Server version: 10.5.19-MariaDB-0+deb11u2 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SHOW GRANTS
-> ;
+-----+
| Grants for root@localhost |
+-----+
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' IDENTIFIED VIA mysql_native_password USING 'invalid' OR unix_socket WITH GRANT OPTION |
| GRANT PROXY ON ''@'%' TO 'root'@'localhost' WITH GRANT OPTION |
+-----+

```

Create table-

```

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";
CREATE TABLE `user_table` (
  `id` int(11) NOT NULL,
  `username` varchar(30) NOT NULL,
  `email` varchar(30) NOT NULL,
  `password` varchar(30) NOT NULL,
  `image` longblob NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `user_table`
  ADD PRIMARY KEY (`id`);
ALTER TABLE `user_table`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;
COMMIT;

```

>SHOW tables;

>DESC user_table;

```

Database changed
MariaDB [userdb]> SHOW TABLES
-> ;
+-----+
| Tables_in_userdb |
+-----+
| user_table        |
+-----+
1 row in set (0.000 sec)

MariaDB [userdb]> DESC user_table
-> ;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id     | int(11) | NO | PRI | NULL | auto_increment |
| username | varchar(30) | NO | | NULL | |
| email  | varchar(30) | NO | | NULL | |
| password | varchar(30) | NO | | NULL | |
| image  | longblob | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.001 sec)

```

Database connection-

```

GNU nano 6.2 db.php Open a new tab (Ctrl+T)
?php
connection oriented
class db{
    public function connect(){
        $connectionResource=mysqli_connect("172.31.30.93","admin","rootroot");
        mysqli_select_db($connectionResource,"userdb");
        if(!$connectionResource)
        {
            die("connection failed".mysqli_connect_error());
        }
        else
        echo "Connection Successful";
        return $connectionResource;
    }
    function __constructor(){

```

Testing connection-

```

admin@ip-172-31-24-151:~$ mysql -u admin -p -h 172.31.30.93
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.5.19-MariaDB-0+deb11u2 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>

```

Email alert configuration-

Enable 2 step verification on your google account -> manage google account -> security -> password -> custom -> generate app based password

Jenkins : install plugin : email alert

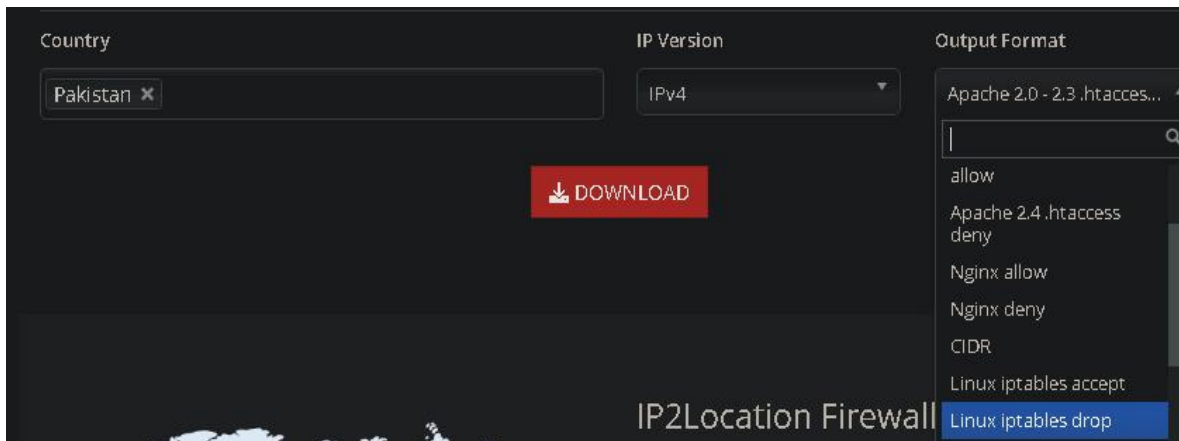
Manage jenkins -> system -> extended email notification

Extended E-mail Notification	
SMTP server	smtp.gmail.com
SMTP Port	465

IPtables firewall configuration-

Go to website : https://lite.ip2location.com/china-ip-address-ranges?lang=en_US

Select Ip2location firwall list -> Download -> extract the file



China IP Address Ranges

China has a total of 351,083,264 IP address assigned. Below are all IP address ranges in China.

Data Source:

- You can download the complete free data in CSV format from [IP2Location LITE DB1](#).
- You can also download the list in firewall format from [IP2Location Firewall List](#).

Simple Notification Service :

SNS Console -> Create Topic -> Type (standard) -> Give name to topic -> Create topic
On the subscription tab -> create subscription -> protocol (email) -> endpoint (mail id) -> create subscription .

Event bridge Console -> create rule -> assign name -> keep default event bus and rule type setting -> next -> in event pattern.

Event pattern [Info](#)

Event source
AWS service or EventBridge partner as source

AWS services ▼

AWS service
The name of the AWS service as the event source

EC2 ▼

Event type
The type of events as the source of the matching pattern

EC2 Instance State-change Notification ▼

☒ Any state
☐ Specific state(s)
▼

☒ Any instance
☐ Specific instance Id(s)

Event pattern
Event pattern, or filter to match the events

```
1 {  
2   "source": ["aws.ec2"],  
3   "detail-type": ["EC2 Instance State-change Notification"]  
4 }
```

[Copy](#) [Test pattern](#) [Edit pattern](#)

Select Target -> choose SNS topic -> name which we assigned earlier

Target 1

Target types
Select an EventBridge event bus, EventBridge API destination (SaaS partner), or another AWS service as a target.

☐ EventBridge event bus
☐ EventBridge API destination
☒ AWS service

Select a target [Info](#)
Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule)

SNS topic ▼

Topic
Mytopic ▼ [Refresh](#)

▼ Additional settings

Configure target input

Info

You can customize the text from an event before EventBridge passes the event to the target of a rule.

Input transformer

▼

Configure input transformer

Rules

A rule watches for specific types of events. When a matching event occurs, the event is routed to the targets associated with the rule. A rule can be associated with one or more targets.

Select event bus

Event bus

Select or enter event bus name

default

▼

Rules (1)

↺

Delete

Enable

Edit

CloudFormation Template ▼

Create rule

Find rules

Any status ▼

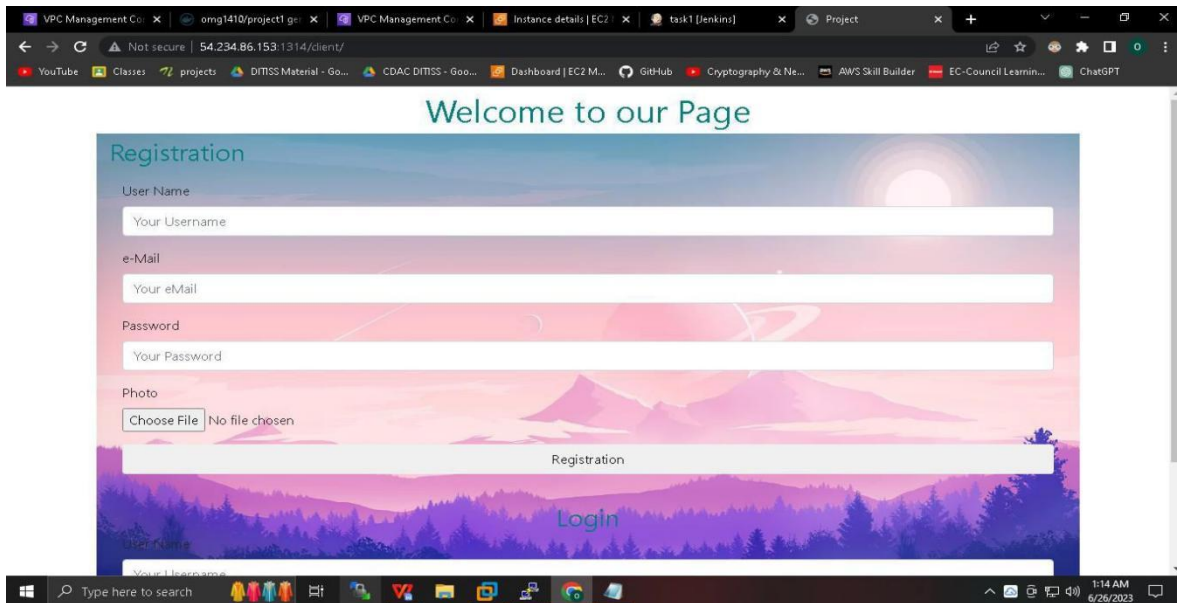
< 1 ... > ⚙

☐	Name	▲	Status	▼	Type	▼	Description	▼
☐	email_alert		🟢 Enabled		Standard		for instances	

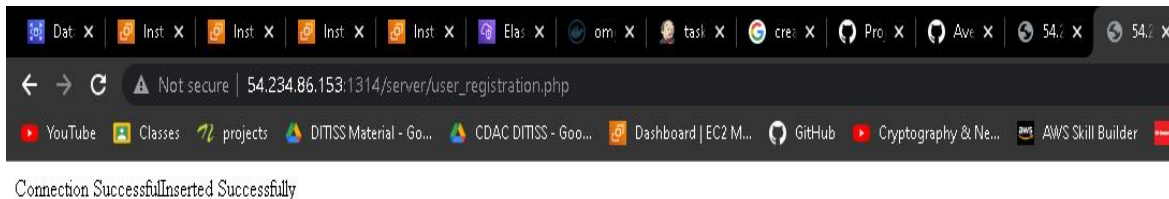
19 | Page

Output-

Browser -> IP_of_Load balancer:1314/client



Testing registration and Login-



Database Check-

>SELECT * FROM user_table;

```
MariaDB [userdb]> SELECT * FROM user_table;
+----+-----+-----+-----+-----+
| id | username | email | password | image |
+----+-----+-----+-----+-----+
| 2 | fg sdfg | dfsadfas | asdfsfsa | NULL |
| 3 | Ojas | ojasjawale1010@gmail.com | Ojas1234 | NULL |
| 4 | komal | komal@baramati.com | komal123 | NULL |
| 5 | omkar | omkar@gmail.com | omkar123 | NULL |
| 6 | kaustubh | kaustubh@gmail.com | kaustubh123 | NULL |
| 7 | nitin | nitin@gmail.com | nitin123 | NULL |
| 8 | OMG | omkargadre1410@gmail.com | Omkargadre | NULL |
+----+-----+-----+-----+-----+
```


Testing firewall-

```
admin@ip-172-31-17-170:~$ nslookup www.tsinghua.edu.cn
Name:      www.tsinghua.edu.cn
Address: 2402:f000:1:404:166:111:4:100

admin@ip-172-31-17-170:~$ ping 166.111.4.100
PING 166.111.4.100 (166.111.4.100) 56(84) bytes of data.
^C
--- 166.111.4.100 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1004ms

admin@ip-172-31-17-170:~$ nslookup english.pku.edu.cn
Server:      172.31.0.2
Address:     172.31.0.2#53

Non-authoritative answer:
english.pku.edu.cn canonical name = webs6.lb.pku.edu.cn.
Name:      webs6.lb.pku.edu.cn
Address: 162.105.120.171
Name:      webs6.lb.pku.edu.cn
Address: 2001:da8:201:1122::a269:78ab

admin@ip-172-31-17-170:~$ ping 162.105.120.171
PING 162.105.120.171 (162.105.120.171) 56(84) bytes of data.
^C
--- 162.105.120.171 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1020ms

admin@ip-172-31-17-170:~$ nslookup www.fmprc.gov.cn
Server:      172.31.0.2
Address:     172.31.0.2#53
```

```
admin@ip-172-31-17-170:~$ nslookup mofa.gov.pk
Server:      172.31.0.2
Address:     172.31.0.2#53

Non-authoritative answer:
Name:      mofa.gov.pk
Address: 203.101.184.123

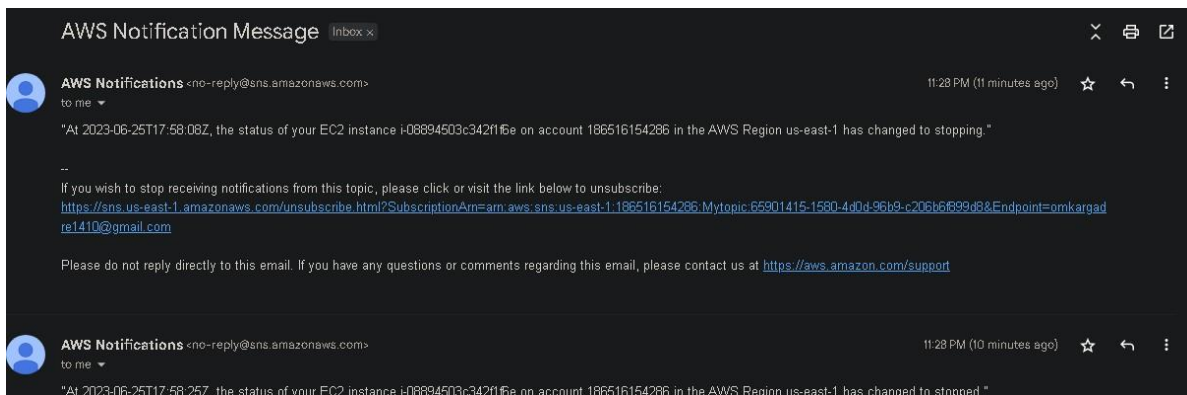
admin@ip-172-31-17-170:~$ ping 203.101.184.123
PING 203.101.184.123 (203.101.184.123) 56(84) bytes of data.
■
```

```
admin@ip-172-31-17-170:~$ nslookup mofa.gov.pk
Server:          172.31.0.2
Address:         172.31.0.2#53

Non-authoritative answer:
Name:   mofa.gov.pk
Address: 203.101.184.123

admin@ip-172-31-17-170:~$ ping 203.101.184.123
PING 203.101.184.123 (203.101.184.123) 56(84) bytes of data.
```

Testing SNS-



7. APPLICATIONS

Layered-to-layered security in a CI/CD pipeline involves implementing multiple security measures at different levels of the development and deployment process to ensure comprehensive protection against potential threats. Here are some applications of layered-to-layered security in a CI/CD pipeline:

1. Source Code Security:

- Code Review: Conduct thorough code reviews to identify vulnerabilities and ensure code quality before it enters the pipeline.
- Static Code Analysis: Integrate static code analysis tools to automatically scan code for security issues and coding standards violations.
- Secrets Management: Utilize secure storage and management of sensitive information like API keys, passwords, and tokens.

2. Build and Packaging Security:

- Container Scanning: Perform container image scanning to identify vulnerabilities and insecure configurations before deploying containers.
- Dependency Analysis: Analyze dependencies for known vulnerabilities and ensure that only approved and up-to-date libraries are used.

3. Pipeline Access Control:

- Role-Based Access Control (RBAC): Implement RBAC to ensure that only authorized personnel can modify the CI/CD pipeline and configurations.
- Two-Factor Authentication (2FA): Require 2FA for access to the CI/CD tooling and infrastructure.

4. Infrastructure Security:

- Immutable Infrastructure: Use immutable infrastructure principles to reduce the attack surface and ensure consistency across deployments.
- Infrastructure as Code (IaC): Apply security controls through IaC tools like Terraform or CloudFormation to define and manage infrastructure securely.

5. Deployment Security:

- Automated Security Testing: Integrate automated security testing tools to identify vulnerabilities in the deployed application.
- Application Whitelisting: Implement application whitelisting to control the executables and scripts that are allowed to run in the production environment.

8. ADVANTAGES & DISADVANTAGES

Advantages of Multilayered Security in a CI/CD Pipeline:

1. **Comprehensive Protection:** Multilayered security provides defense against a wide range of threats, reducing the risk of vulnerabilities slipping through the cracks.
2. **Early Detection:** Security measures at different stages of the pipeline catch vulnerabilities and issues early in the development process, saving time and resources.
3. **Minimized Attack Surface:** Multiple layers reduce the attack surface by addressing vulnerabilities across various aspects of the pipeline.
4. **Adaptability:** If one layer is compromised, other layers can provide backup protection, making it harder for attackers to exploit weaknesses.
5. **Regulatory Compliance:** Multilayered security aids in meeting regulatory requirements by implementing various security controls and best practices.

Disadvantages of Multilayered Security in a CI/CD Pipeline:

1. **Complexity:** Managing multiple security layers can lead to increased complexity, potentially affecting ease of use and maintenance.
2. **Resource Intensive:** Implementing and maintaining multiple security measures requires additional resources, including time, personnel, and tools.
3. **False Positives:** Different security layers may generate false positives, leading to time spent investigating non-issues.
4. **Configuration Challenges:** Coordinating and configuring multiple security solutions can be challenging, leading to misconfigurations or conflicts.
5. **Integration Difficulties:** Ensuring smooth integration between various security tools and layers may require additional effort.

9. CONCLUSION

In conclusion, the adoption of a multilayered security approach in a CI/CD pipeline is a strategic imperative for organizations aiming to deliver secure and reliable software products efficiently. This approach acknowledges the evolving threat landscape and the need to fortify every stage of the development and deployment process. By implementing security measures across multiple layers, from source code to runtime environments, organizations can significantly enhance their ability to detect, prevent, and respond to security threats effectively. The advantages of multilayered security are evident. It offers a holistic defense strategy that minimizes the risk of vulnerabilities going undetected, providing early threat identification and reducing the attack surface. This approach supports regulatory compliance, strengthens incident response capabilities, and fosters a culture of security awareness throughout the development lifecycle. However, it's essential to approach multilayered security with careful consideration. Balancing security measures with operational efficiency is key to ensuring that the CI/CD pipeline remains agile and productive. Proper configuration, seamless integration of security tools, and ongoing training for development teams are critical components of successful implementation.

10. REFERENCES

- 1) Snort Manual : <http://manual-snort-org.s3-website-us-east-1.amazonaws.com/>
- 2) Git tutorials Geeks for geeks : <https://www.geeksforgeeks.org/git-tutorial/>
- 3) Jenkins _Documentation: <https://www.jenkins.io/doc/> , <https://www.jenkins.io/user-handbook.pdf>
- 4) Docker Documentation : <https://docs.docker.com/>
- 5) Linux Tutorial Geek for geek : <https://www.geeksforgeeks.org/linux-tutorial/>
- 6) SQL Manual : <https://dev.mysql.com/doc/refman/8.0/en/>
- 7) AWS tutorials : <https://aws.amazon.com/getting-started/hands-on/>
- 8) AWS Documentation: <https://docs.aws.amazon.com/>
- 9) Iptables Manual : <https://linux.die.net/man/8/iptables>
- 10) HTML Tutorial W3 School : <https://www.w3schools.com/>