

Project Report on

**DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE**

**Submitted by**

**Pravin Sawant (230944223036)**

**Heena Shinganjude (230944223014)**

**Yogesh Dhanawade (230944223011)**

**Rutvija Patil (230344223034)**

Under the guidance of

**Mr. Sandeep Walvekar**

**In partial fulfillment of the award of Post Graduate Diploma in**

**IT Infrastructure, Systems and Security**

**(PG-DITISS)**



**Sunbeam Institute of Information Technology,**

**Pune (Maharashtra)**

**PG-DITISS -2023**

# DECLARATION

We declare that this written submission represents our ideas in our own words and where other's ideas or words have been included; we have adequately cited and referenced the sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented, fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will cause disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed**.**

Place: Pune

Date:

**Pravin Sawant Heena Shinganjude Yogesh Dhanawade Rutvija Patil**

**(230944223036) (230944223014) (230944223011) (230344223034)**

# CERTIFICATE

This is to certify that the project report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”**, submitted by **Pravin Sawant** is the bonafide work completed under our supervision and guidance in partial fulfillment for the award of Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

**Mr. Sandeep Walvekar Mr. Vishal Salunkhe**

Guide Course Coordinator

**Mr. Nitin Kudale**

CEO

Sunbeam Institute of Information Technology

Pune (M.S.) – 411057

# CERTIFICATE

This is to certify that the project report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”**, submitted by **Heena Shinganjude** is the bonafide work completed under our supervision and guidance in partial fulfillment for the award of Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

**Mr. Sandeep Walvekar Mr. Vishal Salunkhe**

Guide Course Coordinator

**Mr. Nitin Kudale**

CEO

Sunbeam Institute of Information Technology

Pune (M.S.) – 411057

# CERTIFICATE

This is to certify that the project report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”**, submitted by **Yogesh Dhanawade** is the bonafide work completed under our supervision and guidance in partial fulfillment for the award of Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

**Mr. Sandeep Walvekar Mr. Vishal Salunkhe**

Guide Course Coordinator

**Mr. Nitin Kudale**

CEO

Sunbeam Institute of Information Technology

Pune (M.S.) – 411057

# CERTIFICATE

This is to certify that the project report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”**, submitted by **Rutvija Patil** is the bonafide work completed under our supervision and guidance in partial fulfillment for the award of Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

**Mr. Sandeep Walvekar Mr. Vishal Salunkhe**

Guide Course Coordinator

**Mr. Nitin Kudale**

CEO

Sunbeam Institute of Information Technology

Pune (M.S.) – 411057

# APPROVAL CERTIFICATE

This Project II report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”** by **Pravin Sawant (230944223036)** is approved for Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

Examiner: 

**(Signature)**



**(Name)**

# APPROVAL CERTIFICATE

This Project II report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”** by **Heena Shinganjude (230944223014)** is approved for Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

Examiner: 

**(Signature)**



**(Name)**

# APPROVAL CERTIFICATE

This Project II report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”** by **Yogesh Dhanawade (230944223011)** is approved for Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

Examiner: 

**(Signature)**



**(Name)**

# APPROVAL CERTIFICATE

This Project II report entitled **“DEPLOYING A WEBSITE ON AWS S3 USING TERRAFORM AND JENKINS CI/CD PIPELINE”** by **Rutvija Patil (230944223034)** is approved for Post Graduate Diploma in IT Infrastructure, Systems and Security (PG-DITISS) of Sunbeam Institute of Information Technology, Pune (M.S.).

Place: Pune

Date:

Examiner: 

**(Signature)**



**(Name)**

# CONTENTS

| **TITLE** | **PAGE NO** |
| --- | --- |
| **Declaration** |  |
| **Certificate** |  |
| **Approval Certificate** |  |
| **Abstract** | i |
| **1. INTRODUCTION** | 1 |
| 1.1 Applications | 2 |
| 1.2 Organization and Project Plan | 4 |
| **2. LITERATURE SURVEY** | 5 |
| Paper 1 | 5 |
| Paper 2 | 5 |
| Paper 3 | 6 |
| Paper 4 | 6 |
| Paper 5 | 7 |
| **3. SYSTEM DEVELOPMENT AND DESIGN** | 8 |
| 3.1 Proposed System | 8 |
| 3.2 Flow Chart | 10 |
| 3.3 Technology used | 10 |
| 3.3.1 AWS S3 | 10 |
| 3.3.2 Git | 11 |
| 3.3.3 Jenkins | 12 |
| 3.3.4 Sonarqube | 13 |
| 3.3.5 Trivy | 14 |
| 3.3.6 DynamoDB | 16 |
| 3.3.7 Nagios | 17 |
| 3.3.8 CloudFront | 18 |
| **4. PROJECT OUTPUT** | 20 |
| **5. CONCLUSION** | 24 |
| 5.1 Conclusion | 24 |
| 5.2 Future Scope | 24 |
| **REFERENCES** | 25 |

# 

# ABSTRACT

In the realm of modern web development, efficient deployment practices are indispensable for ensuring seamless delivery of content while maintaining robust security measures. This project presents an innovative approach to deploying a website on Amazon Web Services (AWS) S3 using Terraform for infrastructure provisioning and Jenkins for Continuous Integration/Continuous Deployment (CI/CD) pipeline automation. The deployment process is fortified with AWS Cloudfront for content delivery and security enhancement, alongside the implementation of Nagios Intrusion Detection System (IDS) for proactive monitoring and threat detection.

The deployment workflow begins with the setup of infrastructure using Terraform, allowing for the definition of AWS resources such as S3 buckets. Jenkins CI/CD pipeline orchestrates the automation of build, test, and deployment stages, ensuring rapid and consistent delivery of website updates. Leveraging AWS Cloudfront, the website's content is distributed across a global network of edge locations, providing improved latency and DDoS protection through SSL termination and Web Application Firewall (WAF) capabilities.

Furthermore, security is bolstered with the incorporation of Nagios IDS, which continuously monitors the website for suspicious activities, unauthorized access attempts, and potential security breaches. Nagios alerts administrators in real-time, enabling swift response to security incidents and proactive mitigation of threats, thereby safeguarding the integrity and availability of the deployed website.

This project encapsulates a comprehensive approach to deploying a website on AWS S3, encompassing infrastructure as code, CI/CD automation, content delivery optimization, and robust security measures. By integrating Terraform, Jenkins, AWS Cloudfront, and Nagios IDS, organizations can streamline their deployment processes, fortify their web applications against evolving threats, and uphold high standards of performance and reliability in today's dynamic digital landscape.

# INTRODUCTION

In today's digital landscape, the deployment of websites has become a critical aspect of establishing a strong online presence for businesses. With the increasing reliance on cloud computing platforms like Amazon Web Services (AWS), organizations are constantly seeking efficient and reliable methods to deploy their websites. However, traditional manual deployment processes often prove to be time-consuming, error-prone, and lack scalability.

To address these challenges, this project aims to automate the deployment process of a website on AWS S3 using Terraform and Jenkins CI/CD pipelines. Additionally, it integrates supplementary tools such as CloudFront, Nagios, SonarQube, Trivy, and DynamoDB to enhance security, ensure code quality, enable proactive monitoring, and provide scalable data storage.

The significance of automation in deploying websites cannot be overstated. By automating the deployment process with Terraform and Jenkins CI/CD pipelines, organizations can achieve greater efficiency and consistency in their deployment workflows. Terraform enables the infrastructure provisioning process to be codified, ensuring consistency and reproducibility across environments. Jenkins CI/CD pipelines automate the build, test, and deployment processes, enabling teams to deliver software changes more frequently and reliably.

Moreover, the integration of supplementary tools like CloudFront, Nagios, SonarQube, Trivy, and DynamoDB further enhances the deployment process. CloudFront acts as a content delivery network (CDN), improving website performance and security by caching content at edge locations worldwide. Nagios provides proactive monitoring of AWS resources, enabling teams to detect and respond to issues promptly. SonarQube ensures code quality by analyzing code for potential issues and vulnerabilities. Trivy scans container images for security vulnerabilities, ensuring a secure deployment environment. DynamoDB offers scalable data storage, enabling efficient data access and storage for dynamic website components.

By leveraging automation and these supplementary tools, organizations can streamline the deployment process, enhance security, ensure code quality, enable proactive monitoring, and provide scalable data storage. Ultimately, this project aims to empower organizations to deliver high-quality websites efficiently and securely, thus enhancing the overall user experience.

## Applications

* E-commerce Platforms: E-commerce businesses can use this project to automate the deployment of their websites, ensuring quick updates to product listings, prices, and promotions. Continuous integration and deployment pipelines ensure that changes are swiftly propagated to the live site, enhancing the customer experience and increasing sales.
* Media and Entertainment: Streaming platforms and media companies can leverage AWS CloudFront to deliver high-quality video content globally. With Terraform and Jenkins, they can automate the deployment of new features and updates, while Nagios monitors the streaming infrastructure for performance issues, ensuring uninterrupted viewing experiences for users.
* Software as a Service (SaaS) Providers: SaaS companies can use this project to deploy and manage their applications on AWS S3. By automating the deployment process with Terraform and Jenkins, they can rapidly iterate on their software, introduce new features, and scale their infrastructure to meet growing demand. Nagios ensures the availability and reliability of its services, minimizing downtime for users.
* Corporate Websites and Portals: Enterprises can use this project to host their corporate websites, intranets, and customer portals on AWS S3. By automating deployment and monitoring processes, they can ensure that their online presence remains secure, responsive, and available to employees, customers, and partners around the clock.
* Educational Platforms: Online learning platforms can benefit from this project by deploying course materials, videos, and interactive content on AWS S3. Terraform and Jenkins enable automated deployment of new courses and updates, while Nagios monitors the platform's performance to ensure a seamless learning experience for students and educators.
* News and Media Outlets: News websites and media outlets can use this project to deliver up-to-date content to their audience worldwide. By leveraging AWS CloudFront for content delivery and Jenkins for continuous deployment, they can quickly publish breaking news stories and multimedia content while Nagios monitors the website's uptime and performance.
* Healthcare Applications: Healthcare organizations can use this project to host patient portals, telemedicine platforms, and health information websites on AWS S3. Automated deployment ensures timely updates to medical records, appointment scheduling systems, and educational resources, while Nagios monitors the availability and performance of critical healthcare services.

## 1.2 Project Plan

**Table: Activities Details**

| **Sr.**  **No.** | **ACTIVITY** | **WEEK** | | | |
| --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** |
| **1** | **Project group formation** |  |  |  |  |
| **2** | **Project work to be started in respective labs** |  |  |  |  |
| **3** | **First review with PPT presentation** |  |  |  |  |
| **4** | **Design Use-Case view as per project** |  |  |  |  |
| **5** | **Design Block diagram as per project** |  |  |  |  |
| **6** | **Second review with PPT presentation** |  |  |  |  |
| **7** | **Selection** |  |  |  |  |
| **8** | **Final review with PPT presentation** |  |  |  |  |
| **9** | **Implementation coding as per project** |  |  |  |  |
| **10** | **Testing, Troubleshooting with different techniques** |  |  |  |  |
| **11** | **Created Soft copy of project and then final hard copy** |  |  |  |  |

# 2. LITERATURE SURVEY

## Paper 1: - A Qualitative Study of DevOps Usage in Practice

**Author:** Floris Erich, C. Amrit & M. Daneva

**Description:** Organizations are introducing agile and lean software development techniques in operations to increase the pace of their software development process and to improve the quality of their software. They use the term DevOps, a portmanteau of development and operations, as an umbrella term to describe their efforts. In this paper we describe the ways in which organizations implement DevOps and the outcomes they experience. We first summarize the results of a Systematic Literature Review that we performed to discover what researchers have written about DevOps. We then describe the results of an exploratory interview-based study involving six organizations of various sizes that are active in various industries. As part of our findings, we observed that all organizations were positive about their experiences and only minor problems were encountered while adopting DevOps.

## Paper 2: - Devops, A New Approach To Cloud Development & Testing

**Author:** Dhaya Sindhu Battina

**Description:** The main purpose of this paper is to explore DevOps and its applications in Cloud development and testing. There's no denying it: DevOps and cloud go hand in hand. This trend will only continue since the bulk of cloud development projects now use DevOps. The advantages of utilizing DevOps with cloud applications are increasingly becoming evident. Competing well in the market necessitates a company's ability to supply services and applications at a rapid rate. To be effective, management procedures and tools need a model that is both swift and dependable. Because of this, we must automate the DevOps processes utilizing cloud and non cloud DevOps automation technologies while designing cloud-native apps. The purpose of this article is to discuss how to migrate DevOps to the cloud and improve software development and operational agility. Likewise, this project will examine ways to expand such DevOps processes and automation to public and/or private clouds. If one is interested in learning more about how the emerging field of DevOps is changing the IT industry, read this paper. Understanding how DevOps and the Cloud work together to aid organizations in transforming themselves is the ultimate objective.

## Paper 3: - Review paper on Network's server monitoring and analysis using Nagios

## Author: Renita Johnson, N. Edna Elizabeth

**Description:** A Survey is a comprehensive research paper that delves into the realm of network server monitoring through the lens of Nagios, an open-source monitoring tool. The paper offers a detailed examination of Nagios' architecture, its components, and its capabilities in monitoring network servers. It explores various monitoring strategies and techniques employed by Nagios, including host checks, service checks, and performance metrics monitoring. Moreover, the paper presents real-world case studies and examples illustrating Nagios' effectiveness in detecting and resolving server-related issues across diverse network environments. Furthermore, it discusses emerging trends and future directions in network server monitoring, addressing challenges and potential areas for improvement in Nagios' features and functionality. Overall, this survey paper serves as a valuable resource for understanding the role of Nagios in network server monitoring and analysis, providing insights for organizations seeking to optimize their IT infrastructure management practices.

## Paper 4: - DevOps: Introducing Infrastructure-as-Code

**Author**: Matej Arta, Tadej Borovˇsak, Elisabetta Di Nitto, Michele Guerriero

**Description:**DevOps entails a series of software engineering tactics aimed at shortening the actionable operation of software design changes. One of these tactics is to harness infrastructure-as-code, that is, writing a blueprint containing deploymentspeciﬁcations ready for cloud orchestration. This abstractbrieﬂy discusses all necessary elements and abstractions in writing and maintaining that blueprint, revolving around a key standard for its expression, namely, the OASIS “Topology andOrchestration Speciﬁcation for Cloud Applications” (TOSCA)industrial standard adopted by as many as 60+ big industrial players worldwide

## 

## 

## 

## 

## Paper 5: - DevOps: Introducing Infrastructure-as-Code

**Author:**Md Rejaul Karim Chowdhury and Abdullah Gani

**Description:**Cloud computing is currently emerging as an ever-changing, growing paradigm that models ``everything-as-a-service.” Virtualise Physical resources, infrastructure, and applications are supplied by service provisioning in the cloud. e evolution in the adoption of cloud computing is driven by clear and distinct promising features for both cloud users and cloud providers. However, the increasing number of cloud providers and the variety of service offerings have made it difficult for the customers to choose the best services. By employing successful service provisioning, the essential services required by customers, such as agility and availability,pricing, security and trust, and user metrics can be guaranteed by service provisioning. Hence, continuous service provisioning that satisfies the user requirements is a mandatory feature for the cloud user and vitally important in cloud computing service offerings.therefore, we aim to review the state-of-the-art service provisioning objectives, essential services, topologies, user requirements,necessary metrics, and pricing mechanisms.

# SYSTEM DEVELOPMENT AND DESIGN

## 3.1 Proposed System

**1.Requirement Analysis:**

* Define project objectives, scope, and requirements, including the features and components of the website to be deployed on AWS S3.
* Identify the AWS resources needed for hosting the website, such as S3 buckets, CloudFront distributions, and DynamoDB tables.

**2.Environment Setup:**

* Set up version control using Git for managing the website's codebase.
* Configure development, testing, and production environments for deployment.
* Provision AWS credentials with appropriate permissions for Terraform, Jenkins, and other tools.

**3.Infrastructure Provisioning with Terraform:**

* Develop Terraform scripts to define the infrastructure components required for hosting the website on AWS S3.
* Define AWS resources such as S3 buckets, CloudFront distributions, Route 53 DNS records, and DynamoDB tables using Terraform configuration files.

**4.Continuous Integration and Deployment with Jenkins:**

* Configure Jenkins CI/CD pipelines to automate the build, test, and deployment processes.
* Integrate Jenkins with Git to trigger pipeline executions upon code commits or merges.
* Implement stages in the pipeline for compiling code, running tests, and deploying artifacts to AWS S3.

**5.Security Integration with CloudFront:**

* Configure CloudFront distributions with security features such as DDoS protection, HTTPS encryption, and access control.
* Ensure proper configuration of CloudFront cache behaviors and origin settings for optimal performance and security.

**6.Code Quality Analysis with SonarQube:**

* Set up SonarQube for static code analysis to identify and address code quality issues in the website's codebase.
* Configure SonarQube analysis tasks in the Jenkins pipeline to analyze code quality metrics during the build process.

**7.Container Security Scanning with Trivy:**

* Integrate Trivy into the Jenkins pipeline to scan container images for vulnerabilities before deployment.
* Define Trivy scanning tasks to analyze container images for known security vulnerabilities and dependencies.

**8.Proactive Monitoring with Nagios:**

* Deploy Nagios for proactive monitoring of AWS resources, including S3 buckets, CloudFront distributions, and DynamoDB tables.
* Configure Nagios checks and alerts to monitor critical metrics such as website availability, latency, and resource utilization.

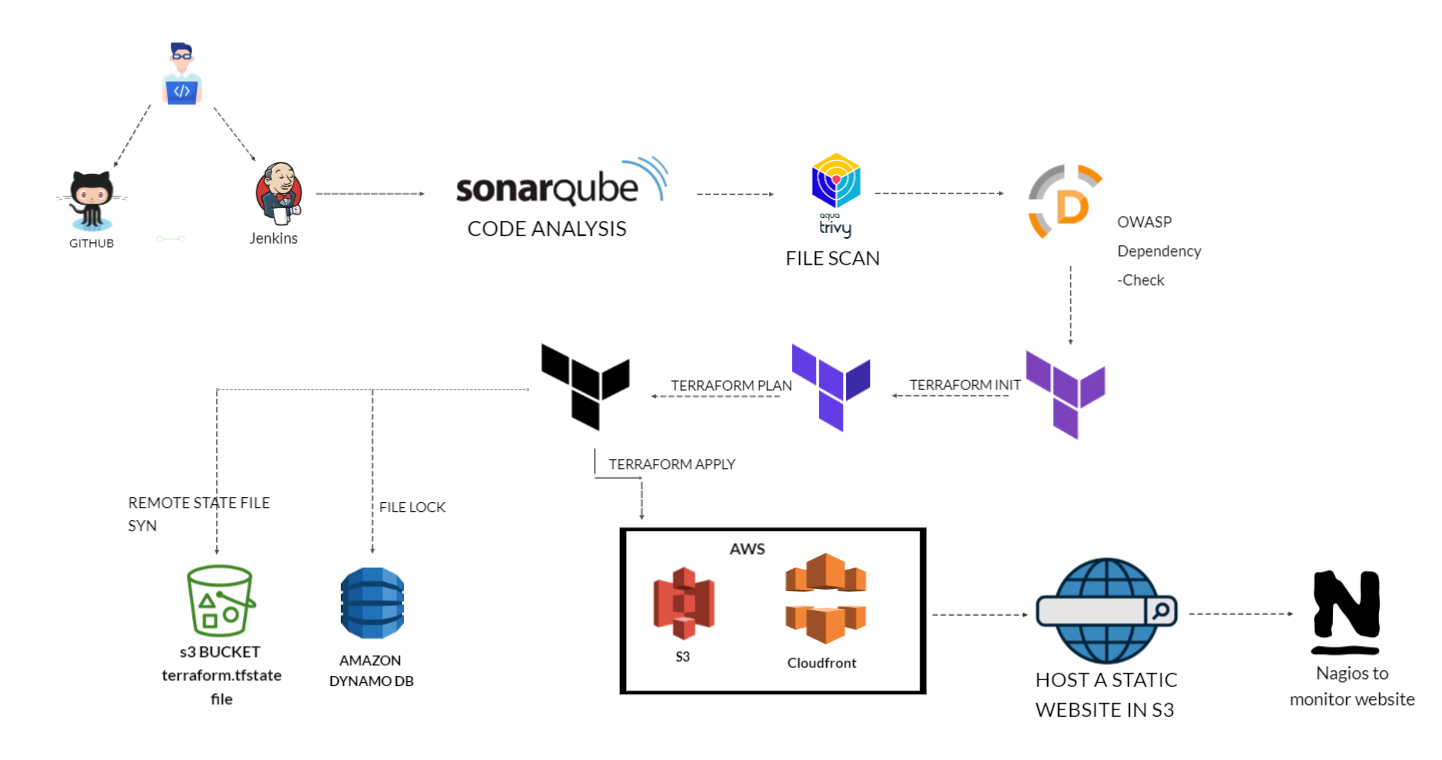
**9.Testing and Validation:**

* Conduct thorough testing of the deployment pipeline to ensure functionality, reliability, and performance.
* Validate website deployment on AWS S3 and verify proper functioning of security configurations, code quality analysis, and monitoring systems.

**10.Documentation and Knowledge Sharing:**

* Document the project setup, configurations, and workflows for future reference and knowledge sharing.
* Provide training and guidance to team members on using and maintaining the deployed infrastructure and CI/CD pipeline.

**3.2 Flow chart**



**Figure: Flowchart**

## 3.3 Technology used

### 3.3.1 Amazon S3

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

**Key features**

1. Scalability: S3 is highly scalable, allowing users to store and retrieve virtually unlimited amounts of data without worrying about capacity constraints. It seamlessly scales to accommodate growing storage requirements, making it suitable for a wide range of use cases, from small-scale applications to enterprise-level solutions.
2. Durability and Reliability: S3 is designed for 99.999999999% (11 nines) durability, ensuring that data stored in S3 remains highly available and resilient to hardware failures, data corruption, and other potential disruptions. It replicates data across multiple Availability Zones within a region, providing redundancy and fault tolerance.
3. Security and Access Controls: S3 offers robust security features to safeguard stored data, including encryption at rest and in transit, access control policies, and integration with AWS Identity and Access Management (IAM). Users can define fine-grained access controls to regulate who can access, modify, and delete objects stored in S3 buckets.

### 3.3.2 Git

Git is a distributed version control system (VCS) designed to manage source code history and facilitate collaborative software development.

**Key features of Git:**

1. Distributed Architecture: Unlike centralized version control systems, Git is distributed. Each developer has a complete copy of the repository, including its entire history. This allows for offline work, faster operations, and improved resilience.
2. Branching and Merging: Git makes it easy to create branches, which are separate lines of development. Developers can work on features, bug fixes, or experiments in their own branches without affecting the main codebase. Merging branches back together is relatively simple and allows for collaborative development.
3. Commit History: Git maintains a detailed history of changes to the codebase. Each change is represented by a commit, which includes information about who made the change, when it was made, and what was changed. This commit history provides a clear view of the evolution of the project.
4. Fast and Efficient: Git is designed for speed and efficiency. Most operations are local, as the repository resides on the developer's machine. This results in rapid commits, branching, and merging.
5. Collaboration: Git enables effective collaboration among developers. Multiple developers can work on different branches simultaneously, and changes can be shared by pushing them to a remote repository. Pull requests or merge requests facilitate the process of reviewing and integrating changes from different contributors.

### 3.3.3 Jenkins

Jenkins is an open-source automation server that facilitates the continuous integration and continuous delivery (CI/CD) of software projects. It helps automate various tasks related to building, testing, and deploying applications, making the development and release process more efficient and reliable.

**Key features of Jenkins:**

1. Continuous Integration: Jenkins automates the process of integrating code changes from multiple contributors into a shared repository. It triggers builds whenever code is committed, allowing developers to identify and fix integration issues early.
2. Automated Builds: Jenkins can automatically build projects from source code repositories. It supports various build tools, languages, and platforms, making it versatile for different types of projects.
3. Extensibility: Jenkins can be extended through a wide range of plugins that provide additional functionalities. Plugins are available for source code management, build tools, testing frameworks, and deployment options.
4. Pipeline as Code: Jenkins uses a domain-specific language called Groovy to define build pipelines as code. This enables you to define complex workflows that include build, test, and deployment stages in a version-controlled script.
5. Continuous Delivery: Jenkins supports continuous delivery by automating the deployment process after successful builds. It can deploy applications to different environments, such as development, staging, and production.
6. Distributed Builds: Jenkins can distribute builds across multiple machines, allowing for parallel builds and improved build performance. This is particularly useful for large and resource-intensive projects.

**3.3.4 Sonarqube**

SonarQube is a static analysis code tool. It basically goes through developers' code and identifies errors at the early stage. It is an open-source static testing analysis software. It is used by developers to manage source code quality and consistency.

**Key features of Jenkins:**

1. Static Code Analysis: SonarQube conducts static code analysis to identify various issues such as bugs, code smells, security vulnerabilities, and other potential pitfalls in the codebase. It analyzes source code without executing it, allowing developers to detect issues early in the development process.
2. Multi-Language Support: SonarQube supports a wide range of programming languages, including popular ones such as Java, JavaScript, C#, Python, Ruby, and many others. This enables teams working with diverse technology stacks to leverage SonarQube for code quality analysis across their projects.
3. Customizable Quality Profiles: SonarQube allows users to define custom quality profiles tailored to their specific requirements and coding standards. Quality profiles enable teams to enforce coding best practices, coding conventions, and security standards consistently across projects.
4. Integration with CI/CD Pipelines: SonarQube seamlessly integrates with Continuous Integration/Continuous Deployment (CI/CD) pipelines, enabling automated code analysis as part of the build and deployment process. Integration with popular CI/CD tools such as Jenkins, Azure DevOps, and GitLab CI/CD streamlines the workflow and ensures that code quality checks are performed continuously.
5. Issue Tracking and Remediation: SonarQube provides detailed reports and dashboards highlighting code issues, along with actionable insights for remediation. Developers can prioritize and track issues, assign them to team members, and monitor the progress of code quality improvement efforts over time.
6. Security Vulnerability Detection: SonarQube includes built-in security rules and plugins to detect common security vulnerabilities and weaknesses in the codebase, such as injection flaws, authentication issues, encryption weaknesses, and more. This helps teams identify and mitigate security risks early in the development process.
7. Technical Debt Management: SonarQube measures technical debt, which represents the effort required to address existing code issues and improve code quality. It provides insights into the cumulative impact of unresolved issues on code maintainability and facilitates informed decision-making regarding prioritization and resource allocation for technical debt reduction.
8. Integration with IDEs: SonarQube integrates with popular Integrated Development Environments (IDEs) such as IntelliJ IDEA, Visual Studio Code, and Eclipse, allowing developers to view code quality metrics and analysis results directly within their development environment. This enables developers to address issues promptly while writing code, promoting a proactive approach to code quality.

**3.3.5 Trivy**

Trivy is an open-source vulnerability scanner for container images, primarily designed to detect security vulnerabilities within containerized environments. It focuses on analyzing container images and their dependencies to identify known vulnerabilities and potential security risks.

**Key features of Jenkins:**

1. Comprehensive Vulnerability Database: Trivy leverages a comprehensive vulnerability database that includes information from multiple sources, such as CVE (Common Vulnerabilities and Exposures), NVD (National Vulnerability Database), and vendor advisories. This database is regularly updated to ensure that Trivy can detect the latest known vulnerabilities affecting container images and their dependencies.
2. Container Image Scanning: Trivy scans container images for known vulnerabilities by analyzing their package dependencies, libraries, and software components. It examines both the operating system packages and application dependencies within the container image to identify vulnerabilities across the entire stack.
3. Fast and Lightweight: Trivy is designed to be fast and lightweight, making it suitable for integration into CI/CD pipelines and automated workflows. It employs efficient scanning techniques to quickly analyze container images without significantly impacting build times or deployment processes.
4. Easy Integration: Trivy can be easily integrated into existing container workflows and toolchains. It provides command-line interface (CLI) and API interfaces, allowing developers and DevOps teams to incorporate vulnerability scanning into their build, test, and deployment pipelines seamlessly.
5. Multiple Image Formats Support: Trivy supports various container image formats, including Docker images, OCI (Open Container Initiative) images, and container images stored in registries such as Docker Hub, Amazon ECR (Elastic Container Registry), and Google Container Registry. This flexibility enables Trivy to scan container images across different environments and platforms.
6. Flexible Configuration Options: Trivy offers flexible configuration options to customize the scanning process according to specific requirements and preferences. Users can configure scan policies, ignore certain vulnerabilities or CVEs, and define severity thresholds for reporting vulnerabilities based on their organization's risk tolerance and compliance requirements.
7. Detailed Scan Reports: Trivy generates detailed scan reports that provide comprehensive insights into detected vulnerabilities, including their severity levels, CVE identifiers, affected packages, and remediation guidance. These reports help users prioritize and address security issues effectively, enabling them to mitigate risks and strengthen the security posture of their containerized applications.
8. Integration with Container Orchestration Platforms: Trivy can be integrated with container orchestration platforms such as Kubernetes, Docker Swarm, and Amazon ECS (Elastic Container Service) to automate vulnerability scanning of containerized workloads and enforce security policies within container environments.

### 3.3.6 DynamoDB

### Amazon DynamoDB is a fully managed NoSQL database service provided by Amazon Web Services (AWS). It offers fast and predictable performance with seamless scalability, making it a popular choice for applications that require low-latency data access and flexible scaling capabilities.

### Key features of DynamoDB include:

### Fully Managed: DynamoDB is a fully managed service, which means that AWS handles administrative tasks such as hardware provisioning, setup, configuration, replication, backups, and scaling. This allows developers to focus on building applications without worrying about infrastructure management.

### Highly Scalable: DynamoDB automatically scales to accommodate fluctuating workloads without requiring manual intervention. It can handle millions of requests per second and can scale both read and write capacity independently based on demand. This scalability ensures consistent performance even as the workload grows.

### Low Latency: DynamoDB provides single-digit millisecond latency for read and write operations, making it suitable for applications that require fast data access. It achieves low latency by replicating data across multiple Availability Zones within a region and using SSD storage for high-speed access.

### Flexible Data Model: DynamoDB supports both key-value and document data models, allowing developers to choose the most appropriate schema for their application requirements. It can store structured, semi-structured, and unstructured data, making it versatile for a wide range of use cases.

### Consistent Performance: DynamoDB offers predictable and consistent performance, with performance metrics such as read and write capacity units (RCUs and WCUs) that can be provisioned to meet specific throughput requirements. It also provides adaptive capacity, which automatically adjusts throughput capacity based on workload patterns.

### Built-in Security: DynamoDB offers built-in security features such as encryption at rest and in transit, fine-grained access control with AWS Identity and Access Management (IAM) policies, and integration with AWS Key Management Service (KMS) for managing encryption keys.

### Global Tables: DynamoDB Global Tables enable multi-region, active-active replication, allowing applications to provide low-latency access to data from any geographic region. This feature enhances availability and disaster recovery capabilities by replicating data across multiple AWS regions.

### DAX Integration: DynamoDB Accelerator (DAX) is an in-memory caching service that can be integrated with DynamoDB to provide even lower latency and higher throughput for read-heavy workloads. DAX caches frequently accessed data, reducing the need to access DynamoDB directly for repetitive queries.

### 3.3.7 Nagios Nagios is an open-source monitoring system that provides comprehensive monitoring and alerting capabilities for IT infrastructure components. It helps organizations monitor the health and performance of their networks, servers, applications, and services, enabling proactive identification and resolution of issues before they impact business operations.

### Key features of Nagios include:

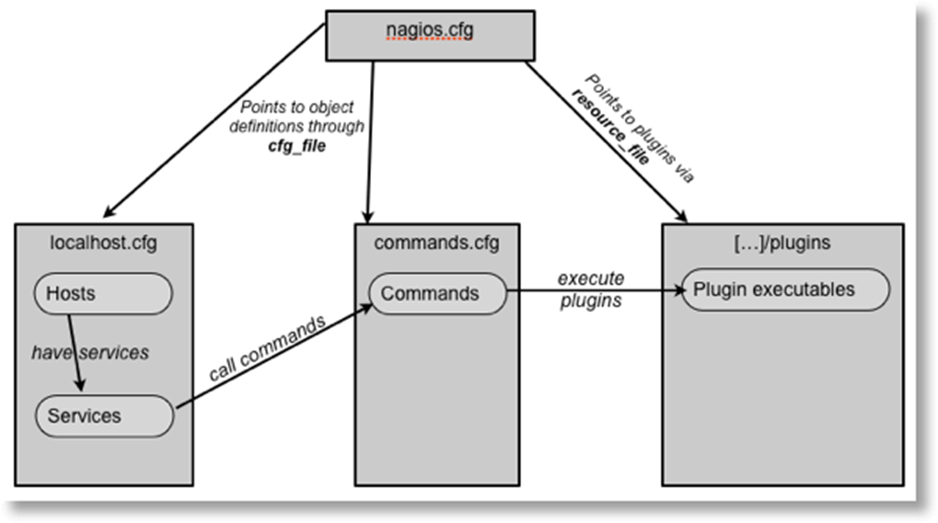
### Monitoring Hosts and Services: Nagios can monitor various types of hosts (servers, devices) and services (applications, network services) by regularly checking their availability and responsiveness.

### Alerting: Nagios generates alerts when it detects that a monitored host or service has a problem. Alerts can be sent via email, SMS, or other notification methods to ensure timely response and issue resolution.

### Threshold Monitoring: Nagios enables you to define thresholds for various metrics (CPU usage, memory usage, response time) and generate alerts when those thresholds are exceeded.

### Flexible Notification: Nagios supports flexible notification configurations, allowing you to define who should be notified based on the time of day, the severity of the issue, and other criteria.

### Plugins: Nagios uses plugins to perform monitoring checks. There are a wide variety of pre-built plugins available, and you can also create custom plugins to monitor specific aspects of your environment.



**3.3.8 Cloudfront**

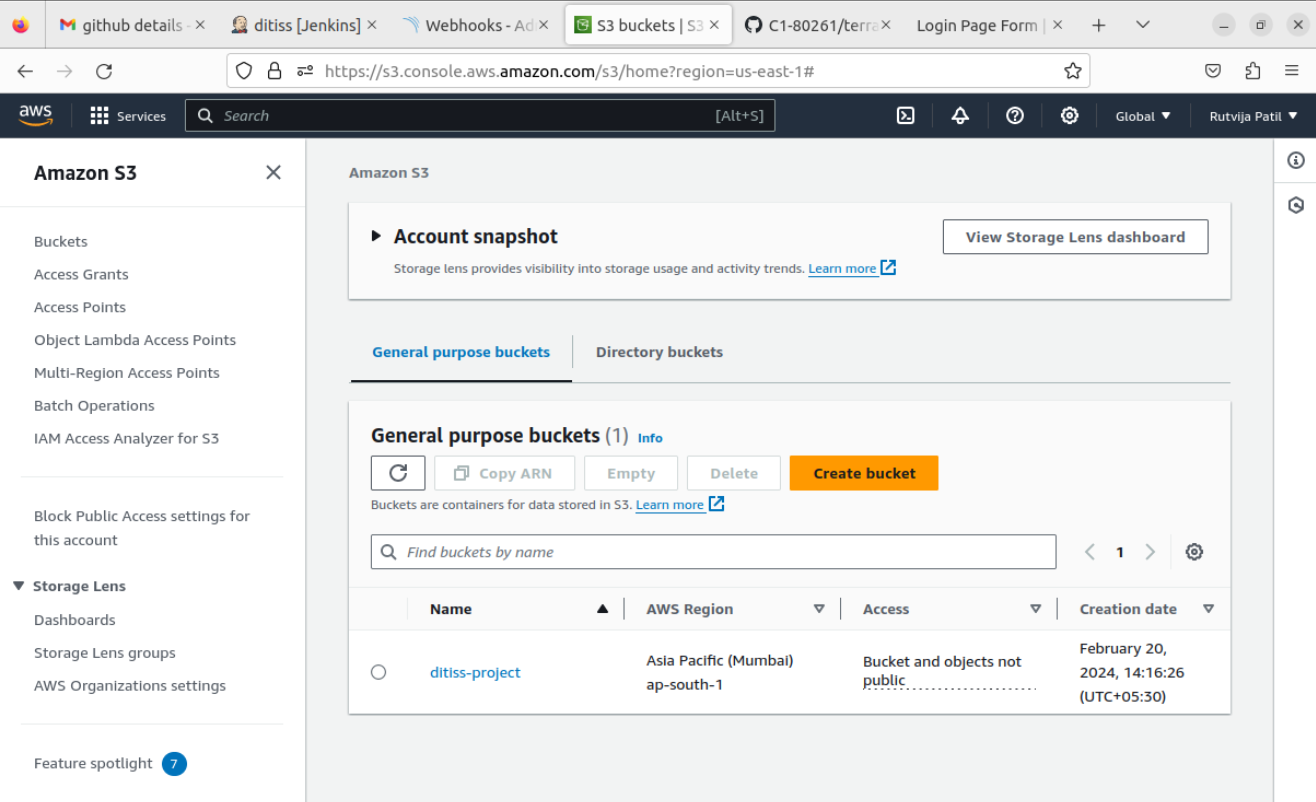
Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency and high transfer speeds. It's integrated with AWS – both physically in terms of its locations around the world and programmatically with AWS services. **When used for websites, CloudFront offers several key features that enhance performance, security, and manageability**

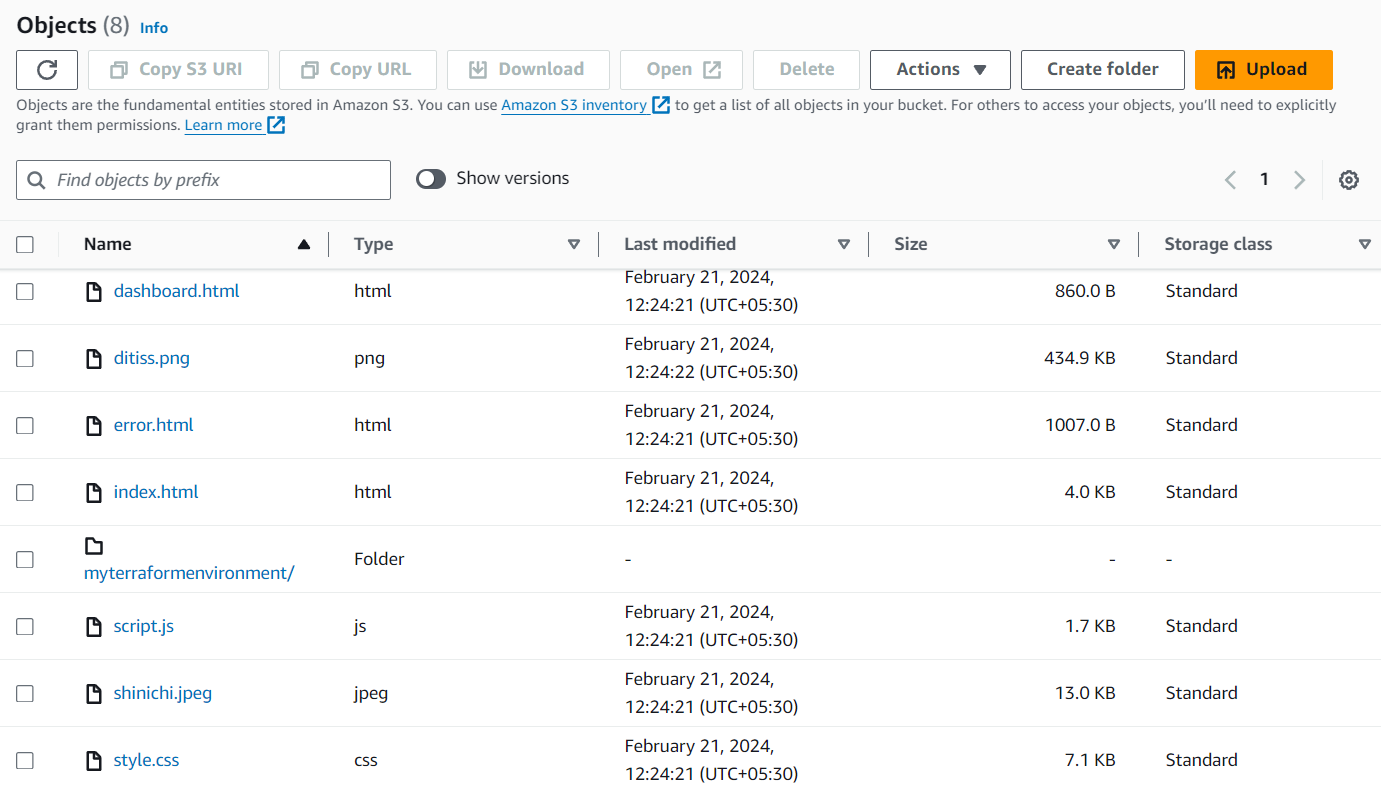
### Key features of include:

1. Global Distribution: CloudFront has a worldwide network of edge locations, enabling content to be delivered from the nearest location to the user, reducing latency.
2. Caching: Content is cached at edge locations, meaning that it's stored closer to end-users for faster access upon subsequent requests.
3. DDoS Protection: Integrates with AWS Shield, providing automatic protection against Distributed Denial of Service (DDoS) attacks.
4. TLS Encryption: Supports HTTPS to encrypt data in transit between CloudFront and end-users, with the option to manage SSL/TLS certificates through AWS Certificate Manager.
5. Geo-Restriction (Geoblocking): Allows you to control who can access your content based on geographic location.
6. Real-time and Historical Reporting: Offers insights through Amazon CloudWatch metrics, access logs, and real-time statistics to monitor and analyze viewer data and activity.
7. HTTP/2 and HTTP/3 Support: Utilizes the latest in web protocols to manage multiple requests over a single connection, improving loading times.
8. Origin Shield: A centralized caching layer that helps protect your origin server from excessive traffic that could potentially impact its performance and availability.
9. AWS WAF Integration: Works with AWS Web Application Firewall to filter malicious traffic and enforce specific content restrictions.

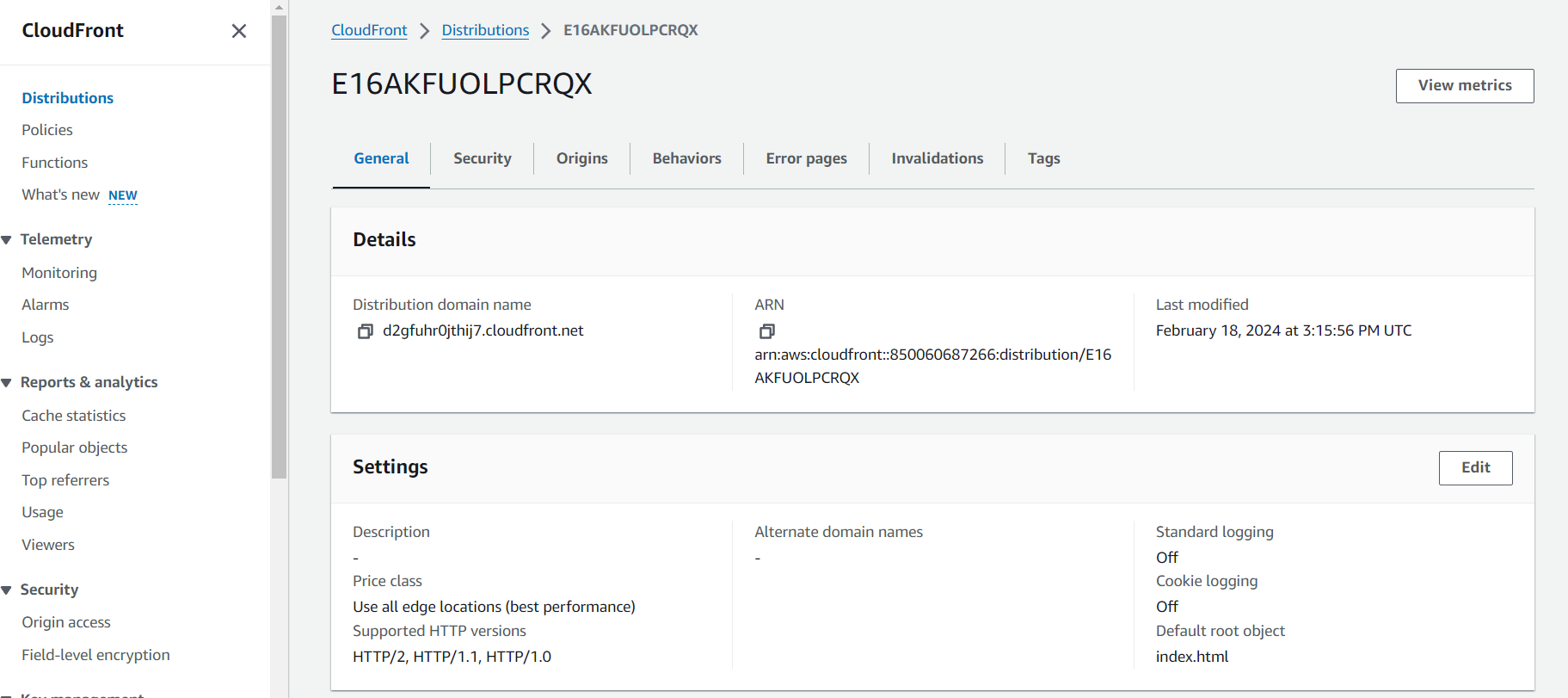
# 4. Project Output

## AWS S3

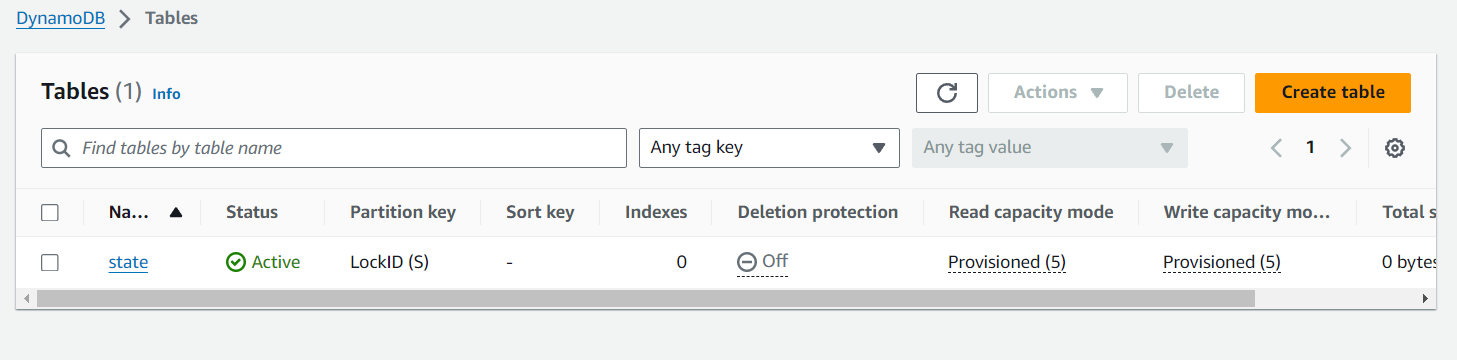




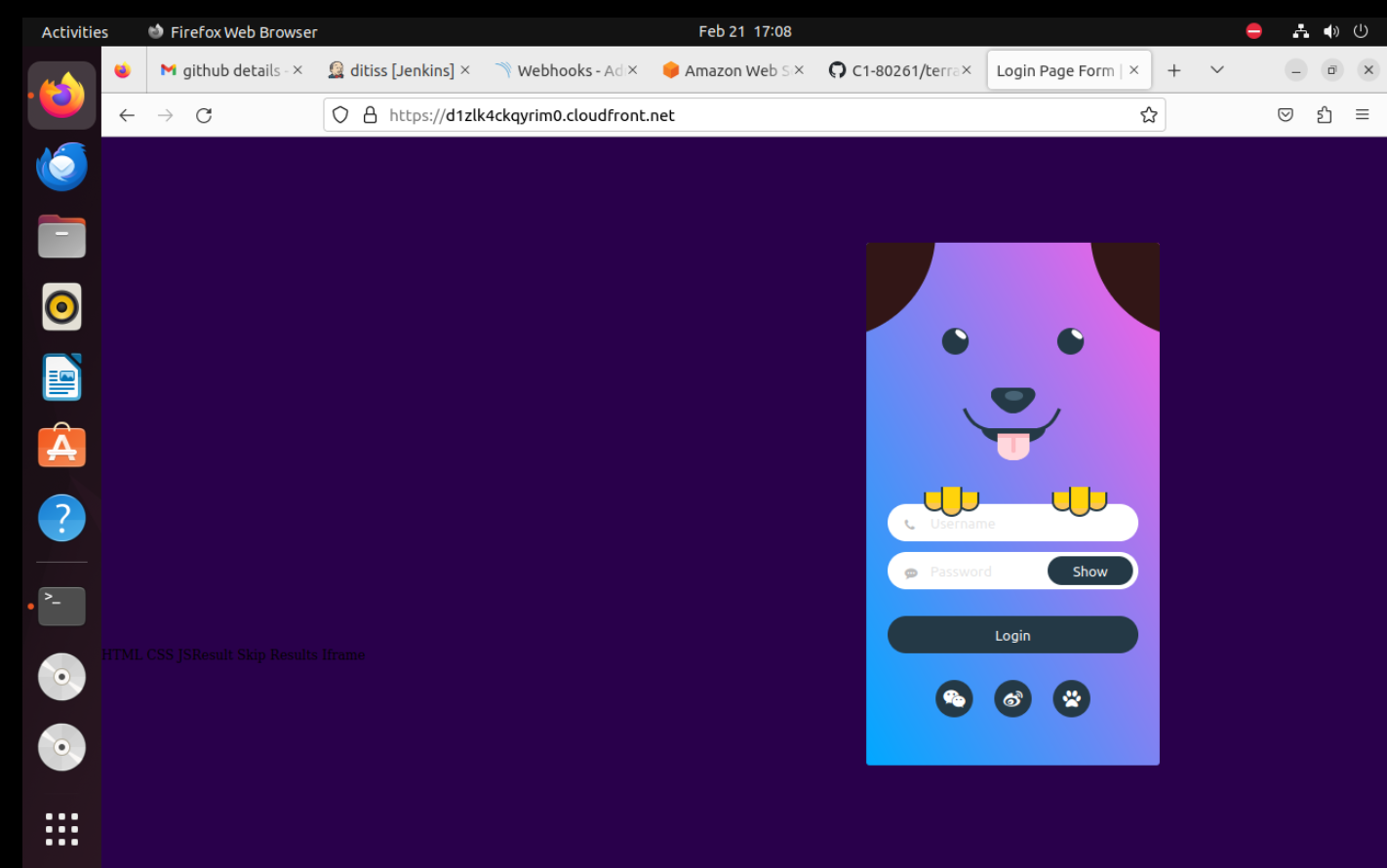
## Cloudfront



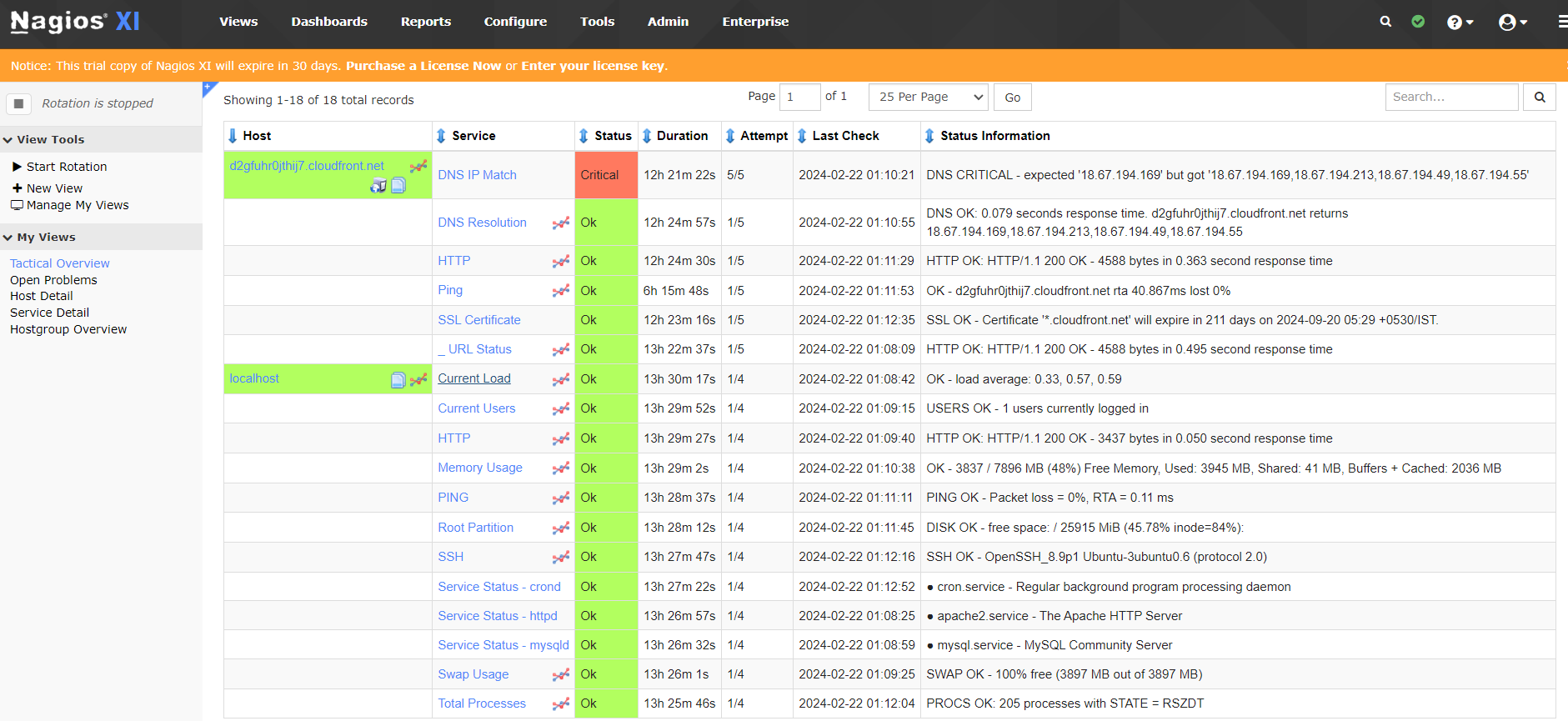
**4.4 DynamoDb**

****

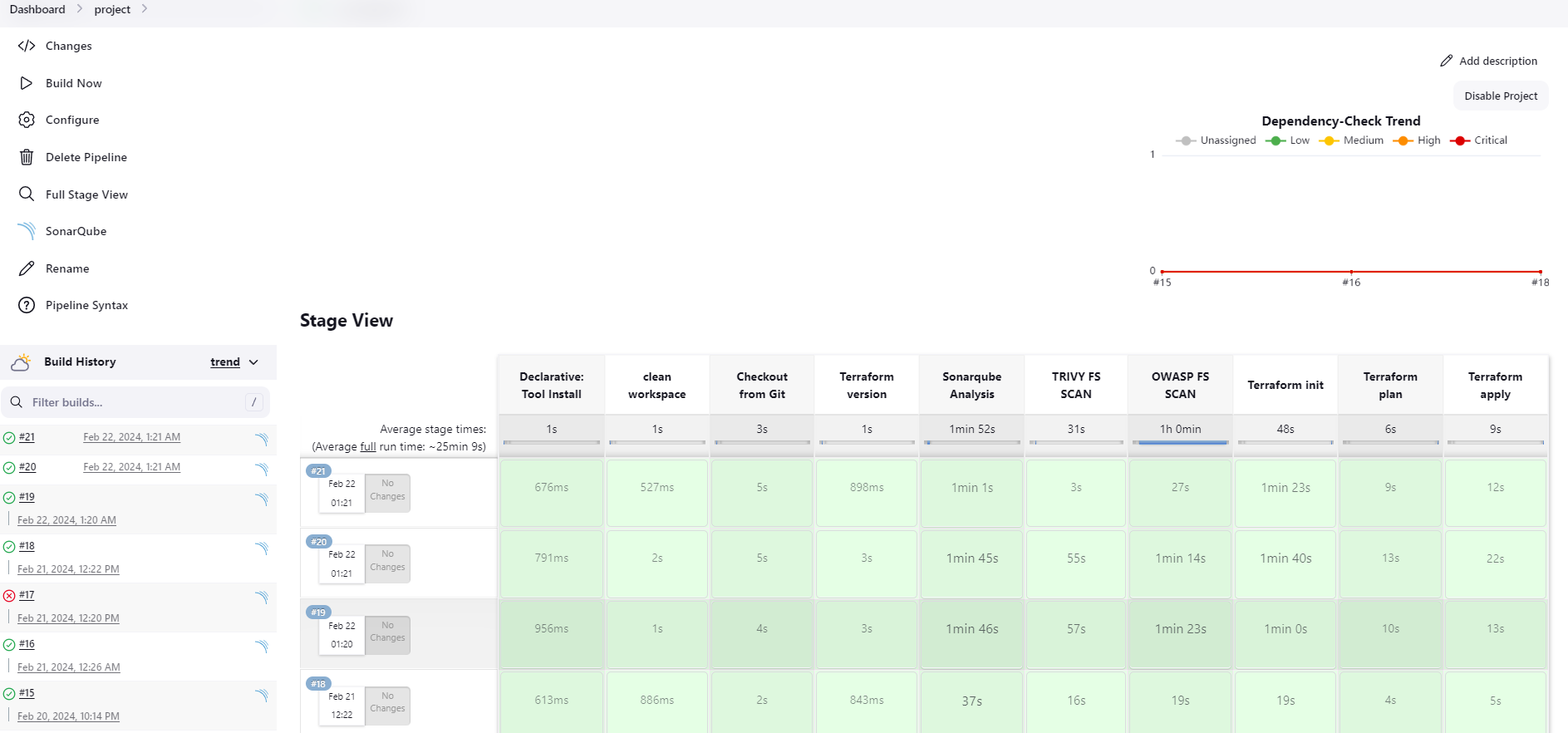
## 4.4 Web Application



**4.5 Nagios**



**4.6 Jenkins**

****

# 5. CONCLUSION

## 5.1 Conclusion

Hence, we have successfully implemented an automated pipeline for deploying websites on AWS S3 using Terraform and Jenkins CI/CD pipelines, with additional tools like CloudFront, Nagios, SonarQube, Trivy, and DynamoDB. The streamlined process improved efficiency, enhanced security, ensured code quality, enabled proactive monitoring, and provided scalable data storage. Overall, these outcomes contribute to more reliable and secure website deployments, ultimately enhancing the user experience.

## 5.2 Future Scope

In the future, this project can be expanded to incorporate advanced security measures like AWS WAF and Security Hub, ensuring robust protection against threats. Containerization with Docker could streamline deployment and management processes, while serverless architecture using AWS Lambda could enhance scalability. Refining Terraform scripts to automate provisioning of additional services and integrating advanced monitoring solutions for deeper insights into system performance are also potential areas of growth. Multi-environment support, automated testing integration, and cost optimization strategies can further enhance efficiency and reliability. Additionally, implementing high availability and disaster recovery solutions and fostering a culture of continuous improvement through feedback loops will ensure the project evolves to meet the evolving needs of modern web hosting environments on AWS.

# REFERENCES

**Paper 1:** - A Qualitative Study of DevOps Usage in Practice

Author: Floris Erich, C. Amrit & M. Daneva

**Paper 2:** - Devops, A New Approach To Cloud Development & Testing

Author: Dhaya Sindhu Battina

**Paper 3:** - Review paper on Network's server monitoring and analysis using Nagios

Author: Renita Johnson, N. Edna Elizabeth

## Paper 4: - DevOps: Introducing Infrastructure-as-Code

**Author**: Matej Arta, Tadej Borovˇsak, Elisabetta Di Nitto, Michele Guerriero

## Paper 5: - DevOps: Introducing Infrastructure-as-Code

**Author:**Md Rejaul Karim Chowdhury and Abdullah Gani