**PROJECT REPORT**

(Project Term January- May 2023)

## (CICD pipeline using Jenkins)

## (Website deployment using Docker and Jenkins)

Submitted by:

**Shiv Shankar Singh Registration Number :12018635**

**Course Code: INT332**

**Course Title: Devops virtualization and configuration management**

**Roll No: RKO212A28**

Submitted To

**(Priyanka Gupta)**

**School of Computer Science and Engineering**



**TABLE OF CONTENTS**

Inner first page……………………………………………………….…….………….(i)

Table of content……………………...…………………………….……….…………(ii)

Introduction......……………………………………………………..……….………..(iii)

Project Description…………..……………………………………….…….…………(iv)

Technology Used……………………...…………………………………….…….......(v)

Project, Code, and Output……………………...……………………………..………(vi)

Discord Bot (1)………………………………………………………………………..(vii)

Discord Bot (2)………………………………………………………………………..(viii)

Bibliography…………………………………………………………………………..(ix)

Introduction

**What is Devops: -**

DevOps is a set of practices that combines software development (Dev) and IT operations (Ops) to optimize and automate the processes of building, testing, deploying, and monitoring software applications.

In traditional software development approaches, developers and operations teams work in separate silos, resulting in a lack of collaboration and communication, which can lead to inefficiencies, errors, and delays. DevOps aims to break down these barriers and foster a culture of collaboration, automation, and continuous improvement.

The key principles of DevOps include continuous integration and continuous delivery (CI/CD), infrastructure as code, automated testing, and monitoring. By adopting these principles, organizations can accelerate the delivery of software, improve the quality and reliability of their applications, and respond quickly to changes in customer needs and market demands.

**What is Jenkins: -**

Jenkins is an open-source automation tool used for continuous integration (CI) and continuous delivery (CD) of software applications. It allows developers to automate the building, testing, and deployment of their code, enabling them to catch and fix bugs more quickly and release software updates more frequently.

Jenkins is a Java-based application that runs on a web container such as Apache Tomcat. It can be configured through a web-based interface or by writing scripts using its own scripting language, called "Jenkinsfile."

Jenkins supports a wide range of plugins that can be used to extend its functionality and integrate with various software development tools and platforms. It can be integrated with version control systems such as Git, SVN, and Mercurial, as well as with testing and deployment tools such as Docker, Kubernetes, and AWS.

Jenkins is widely used in software development organizations of all sizes and is known for its flexibility, scalability, and ease of use.

**What is CI/CD Pipeline: -**

A CI/CD (Continuous Integration/Continuous Delivery) pipeline is a set of practices and tools that help automate the process of building, testing, and deploying software applications.

The pipeline typically starts with the continuous integration stage, where developers push their code changes to a shared repository, and the code is built and tested automatically to detect and fix errors early on. This stage ensures that the application is always in a working state and that all code changes are integrated smoothly.

The pipeline then moves on to the continuous delivery stage, where the code is deployed to a staging or production environment automatically, using tools such as Jenkins or GitLab. This stage ensures that the code is always ready for release and that it can be deployed to production at any time with minimal effort and risk.

Finally, the pipeline may include a continuous deployment stage, where the code is automatically deployed to production after passing all necessary tests and reviews.

The CI/CD pipeline is a critical component of modern software development practices, as it helps teams to deliver software more quickly, reliably, and with higher quality. It allows developers to focus on writing code while the pipeline takes care of the repetitive and time-consuming tasks of building, testing, and deploying the application.

**Benefits of CICD pipeline: -**

There are several benefits to using a CI/CD (Continuous Integration/Continuous Delivery) pipeline in software development.

* Faster feedback: A CI/CD pipeline can provide immediate feedback on the quality of the code changes, allowing developers to catch and fix issues early in the development process. This helps to reduce the time and effort required for bug fixing and testing later.
* Improved quality: By automating the testing and deployment of code changes, a CI/CD pipeline can help ensure that the application is always in a working state and that new features and fixes are delivered reliably and consistently.
* Faster time to market: With a CI/CD pipeline, developers can deliver new features and updates to the application more quickly and frequently, enabling the organization to respond to changing market needs and customer demands faster.
* Increased collaboration: A CI/CD pipeline encourages collaboration between developers, testers, and other stakeholders, as it provides a common platform for sharing code, feedback, and ideas.
* Reduced risk: By automating the testing and deployment process, a CI/CD pipeline can help reduce the risk of human error and ensure that the application is deployed safely and securely.

Overall, a CI/CD pipeline can help organizations to deliver software more quickly, reliably, and with higher quality, while enabling teams to collaborate more effectively and streamline their development workflows.

**What is Cloud computing: -**

Cloud computing refers to the delivery of computing services over the internet, such as servers, storage, databases, networking, software, and analytics. Rather than storing and processing data locally on a personal computer or server, cloud computing enables users to access computing resources on-demand and pay for what they use.

Cloud computing providers, such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud, own and manage the infrastructure required to run applications and services, including data centres, servers, and networking equipment. Users can access these resources through a web-based interface or an API, allowing them to quickly provision and scale up or down their computing resources as needed.

Cloud computing offers several benefits, such as cost savings, scalability, flexibility, reliability, and security. It allows organizations to reduce their capital expenses and pay for only the resources they need, enabling them to focus on their core business instead of managing IT infrastructure. Additionally, cloud computing enables teams to collaborate more easily, access applications and data from anywhere, and leverage advanced technologies such as machine learning and big data analytics.

**What is Git and GitHub: -**

Git is a distributed version control system used to manage and track changes in software code. It allows developers to collaborate on a project by keeping track of changes made to the code and enabling team members to work on different branches of the code simultaneously. Git allows developers to easily revert to previous versions of the code, merge changes made by multiple developers, and manage conflicts that may arise.

GitHub is a web-based hosting service for Git repositories. It provides a platform for developers to share their code with others, collaborate on projects, and contribute to open-source projects. GitHub offers a range of features, such as pull requests, issues, and code reviews, that facilitate collaboration and help teams manage their code effectively. GitHub also offers additional features, such as continuous integration and deployment, project management tools, and analytics, that help teams automate and streamline their software development workflows.

Together, Git and GitHub have become essential tools for software developers, enabling them to work more efficiently, collaborate more effectively, and produce high-quality software code.

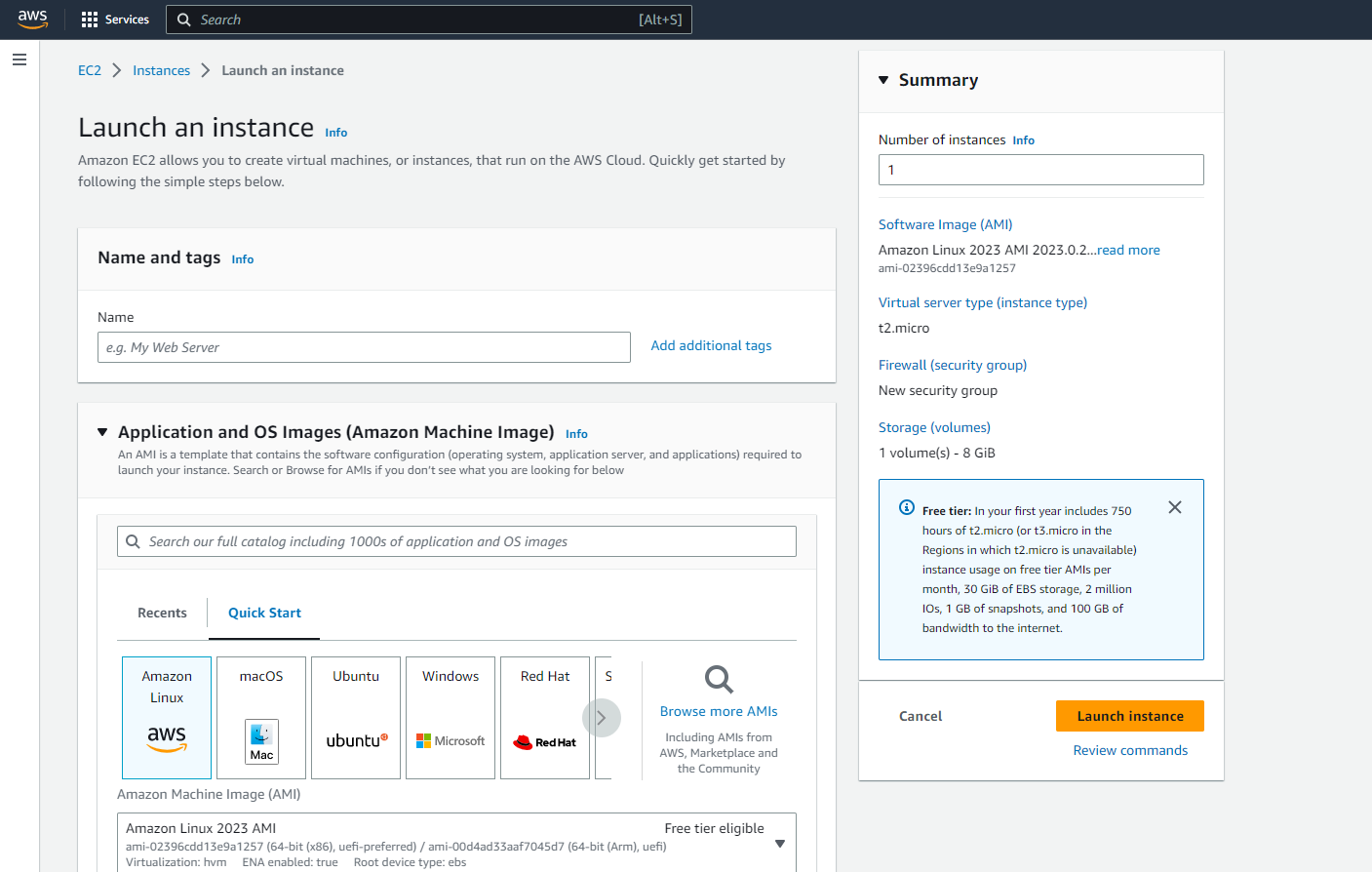
Project Description

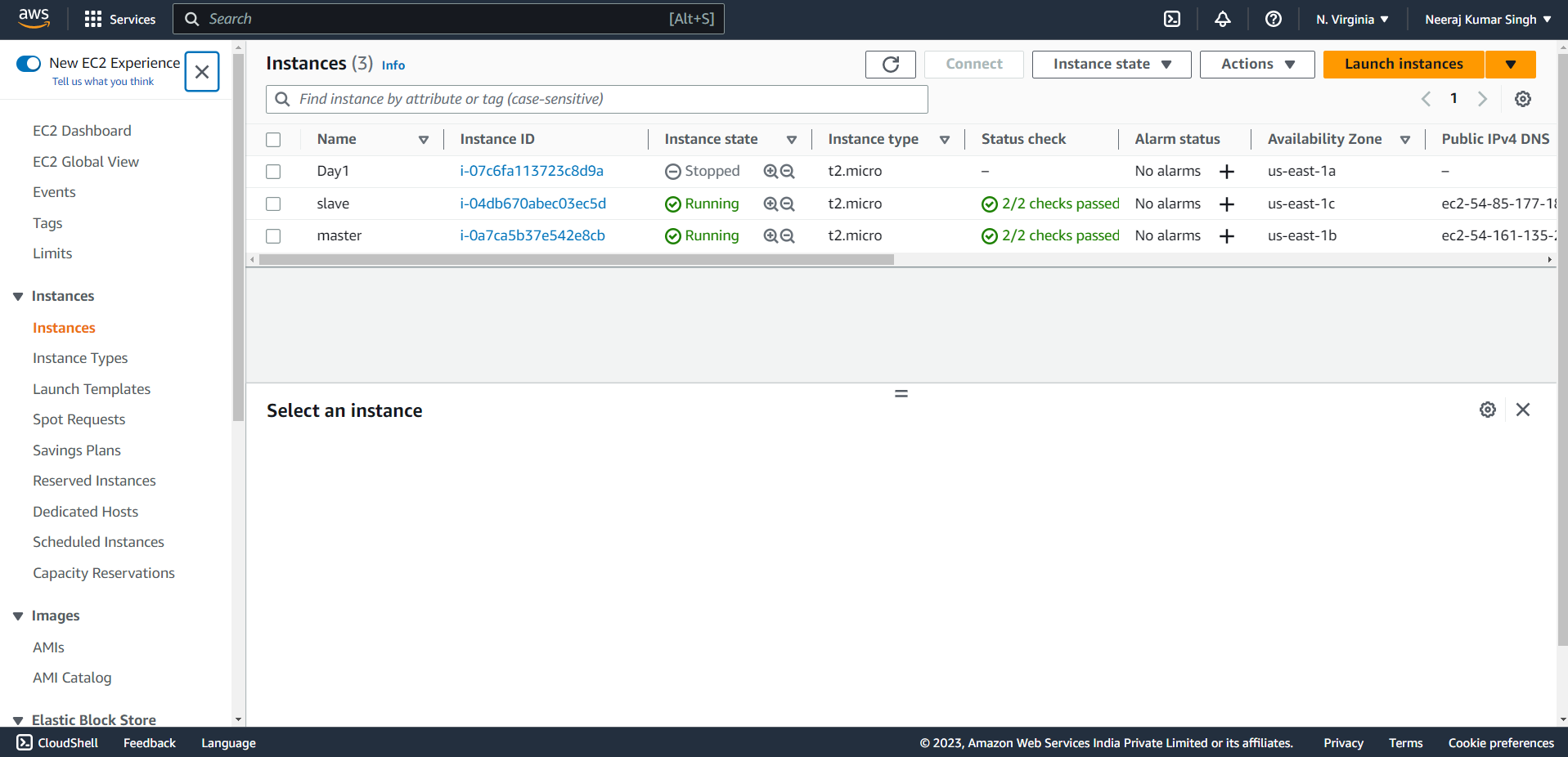
Technology used:

* **Jenkins**: Jenkins is an open-source automation server that helps teams automate their software development workflows. It enables faster building, testing, and deployment of code changes, and is known for its flexibility and ease of use.
* **DOCKER:** Docker is a containerization platform that enables developers to package applications and dependencies into portable containers that can be deployed on any platform.
* **GitHub:** GitHub is a web-based hosting service for Git repositories. It provides a platform for developers to share their code with others, collaborate on projects, and contribute to open-source projects.
* **AWS instances:** An AWS instance is a virtual server provided by Amazon Web Services (AWS) that can be used to run applications and services. It allows users to access computing resources such as CPU, memory, storage, and networking on-demand, without the need to purchase or maintain physical hardware.

Project description and steps:

* **Creating AWS instances:** to set up an environment.

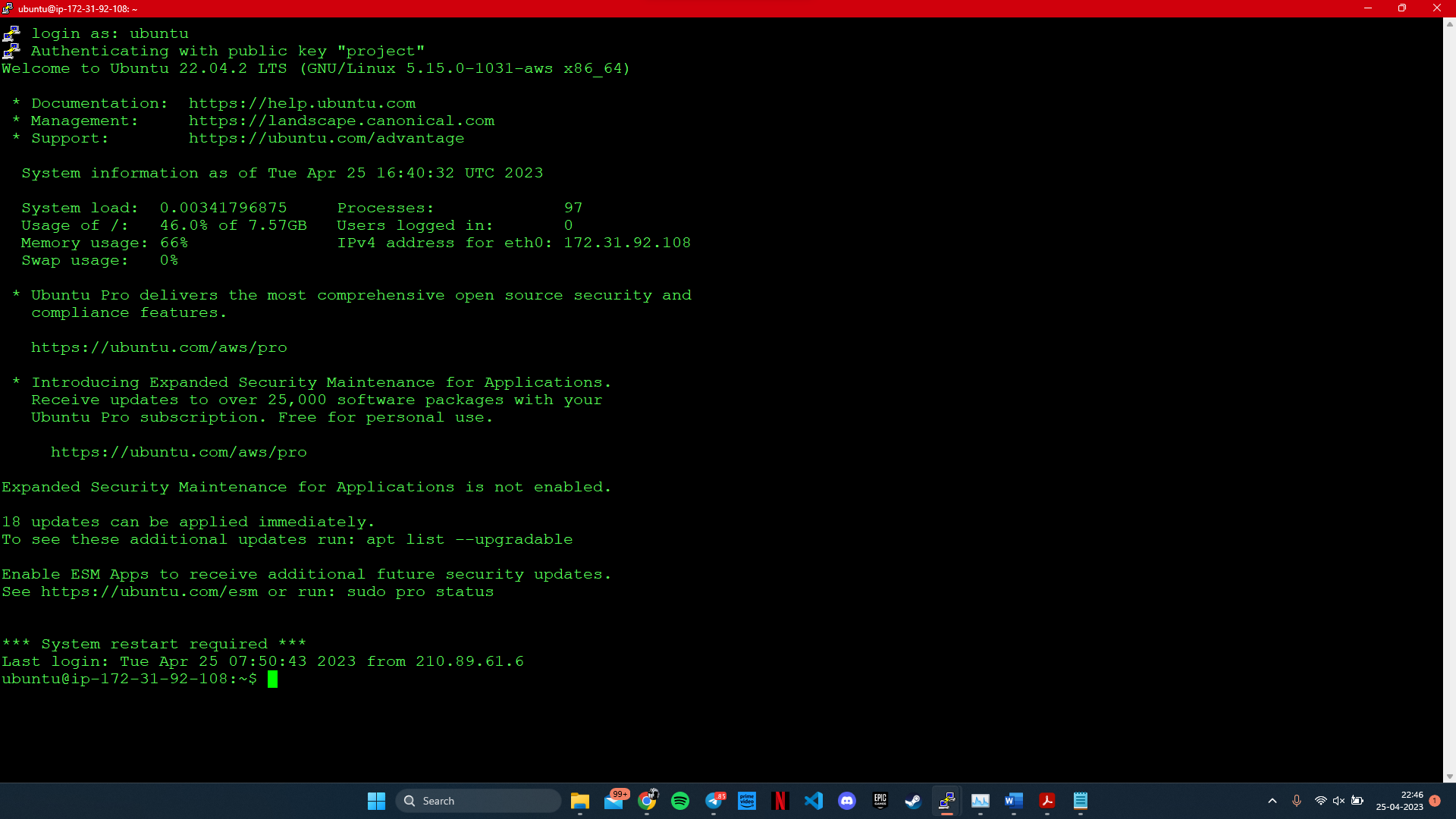




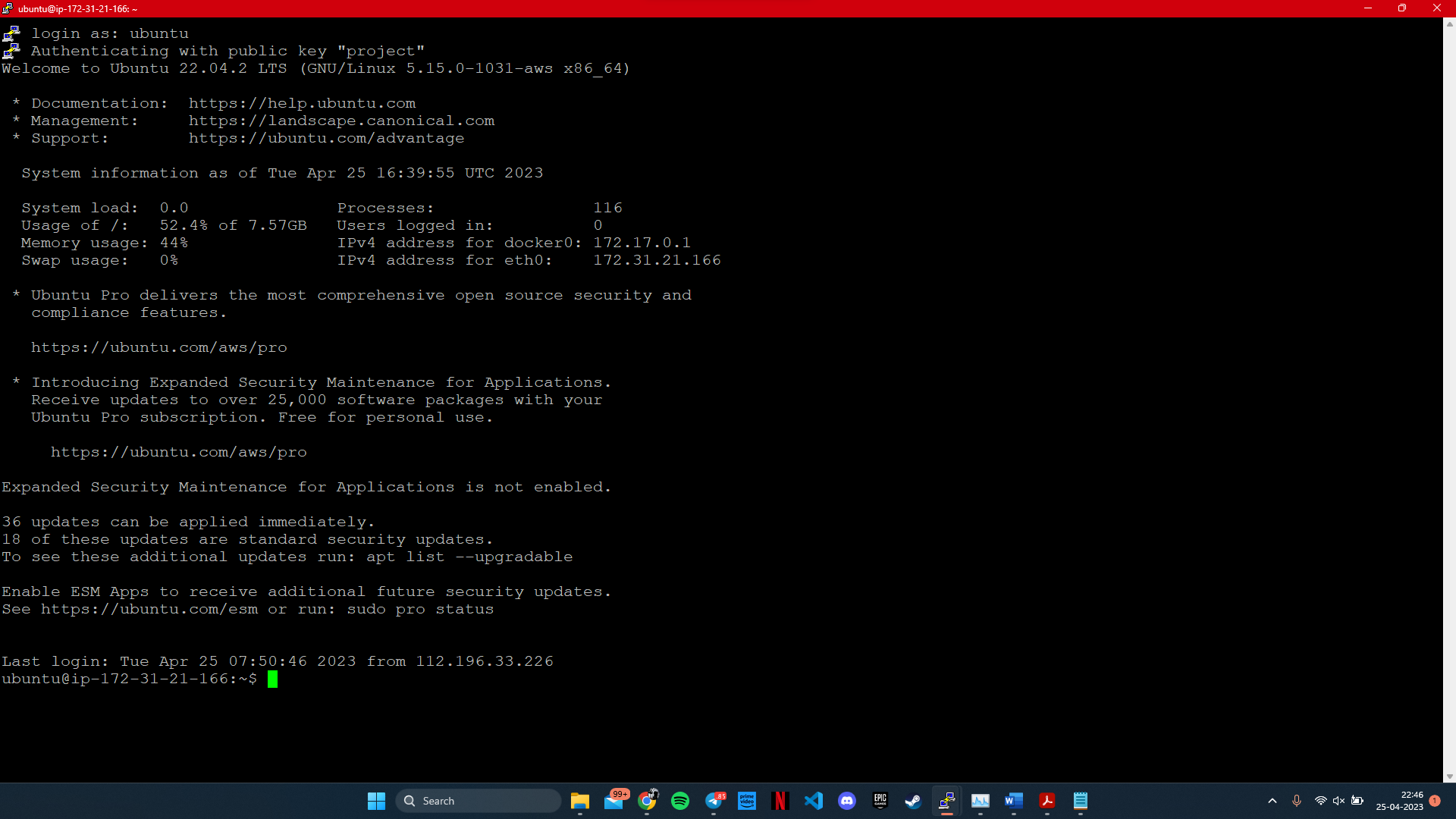
* **Connecting with PuTTy terminal:**

Open putty and provide public IP address of master and slave instance and connect the .ppk key to start terminal.

Slave terminal: -



Master terminal: -



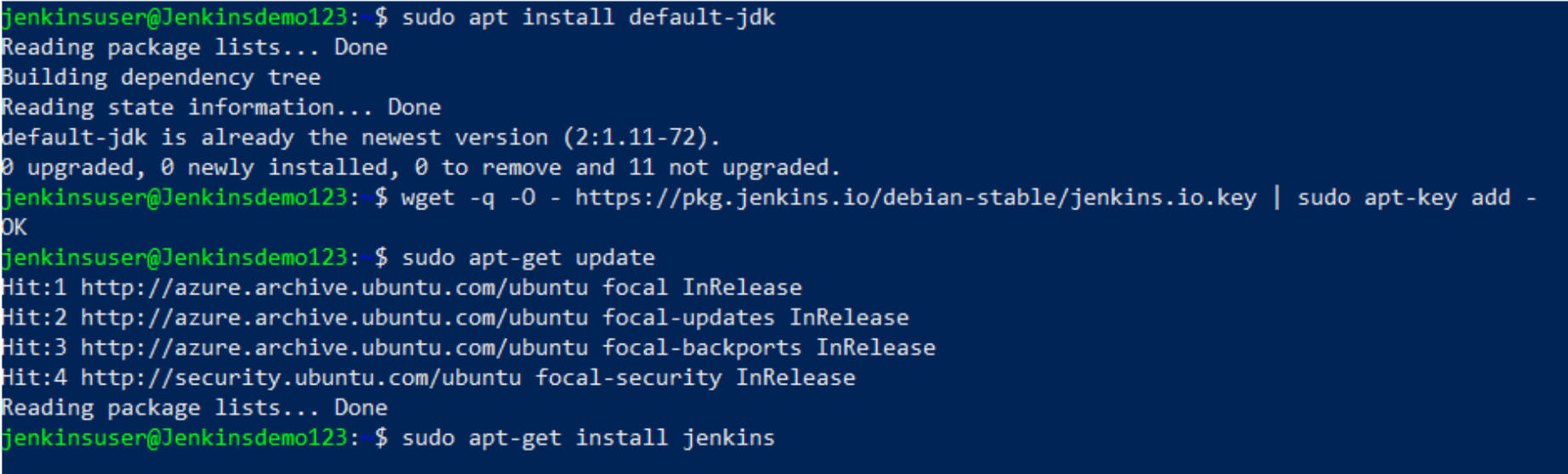
* **Download and installation or Jenkins:**

we need to write some commands.

1. curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \/usr/share/keyrings/jenkins-keyring.asc > /dev/null
2. echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \https://pkg.jenkins.io/debian-stable binary/ | sudo tee \/etc/apt/sources.list.d/jenkins.list > /dev/null
3. sudo apt update
4. sudo apt install default-jdk
5. wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.

io.key | sudo apt-key add -

1. sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
2. sudo apt-get update
3. sudo apt-get install jenkins.





* **Starting Jenkins:**

Copy master’s instance public IP address followed by “:8080”.

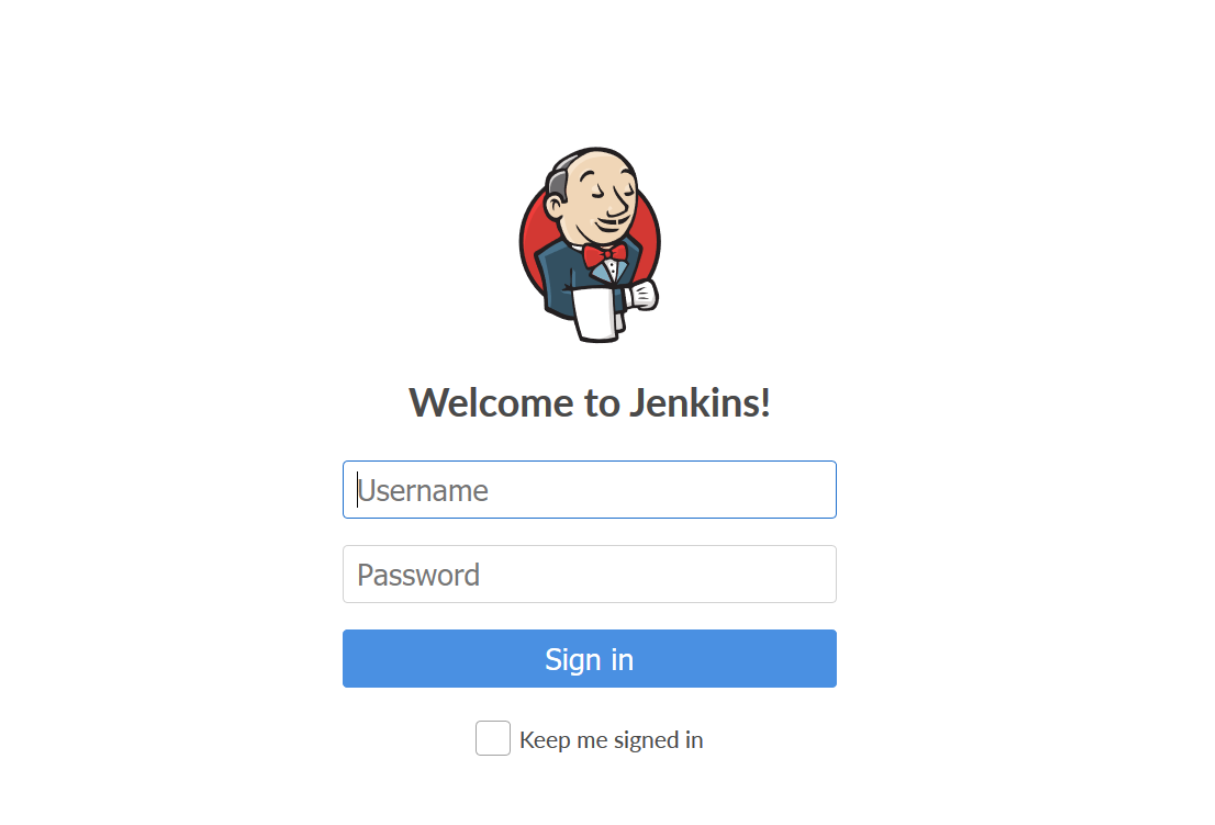
<http://54.161.135.252:8080/>

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated



Jenkins Dashboard:

A screenshot of a computer

Description automatically generated

* **Committing source code on GitHub:**

A screenshot of a computer

Description automatically generated

Text

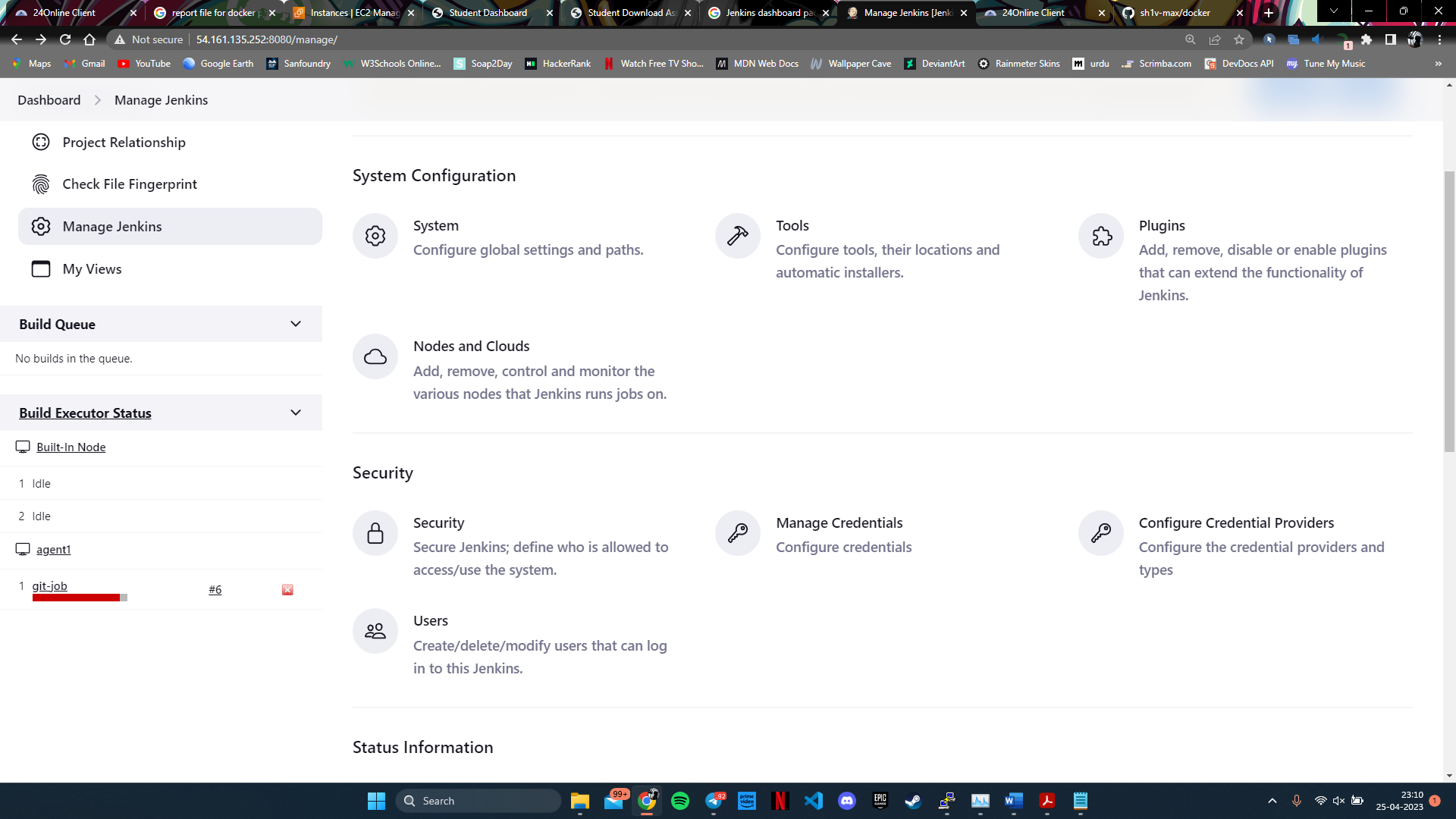
Description automatically generated

* **Importing files in Instances:**

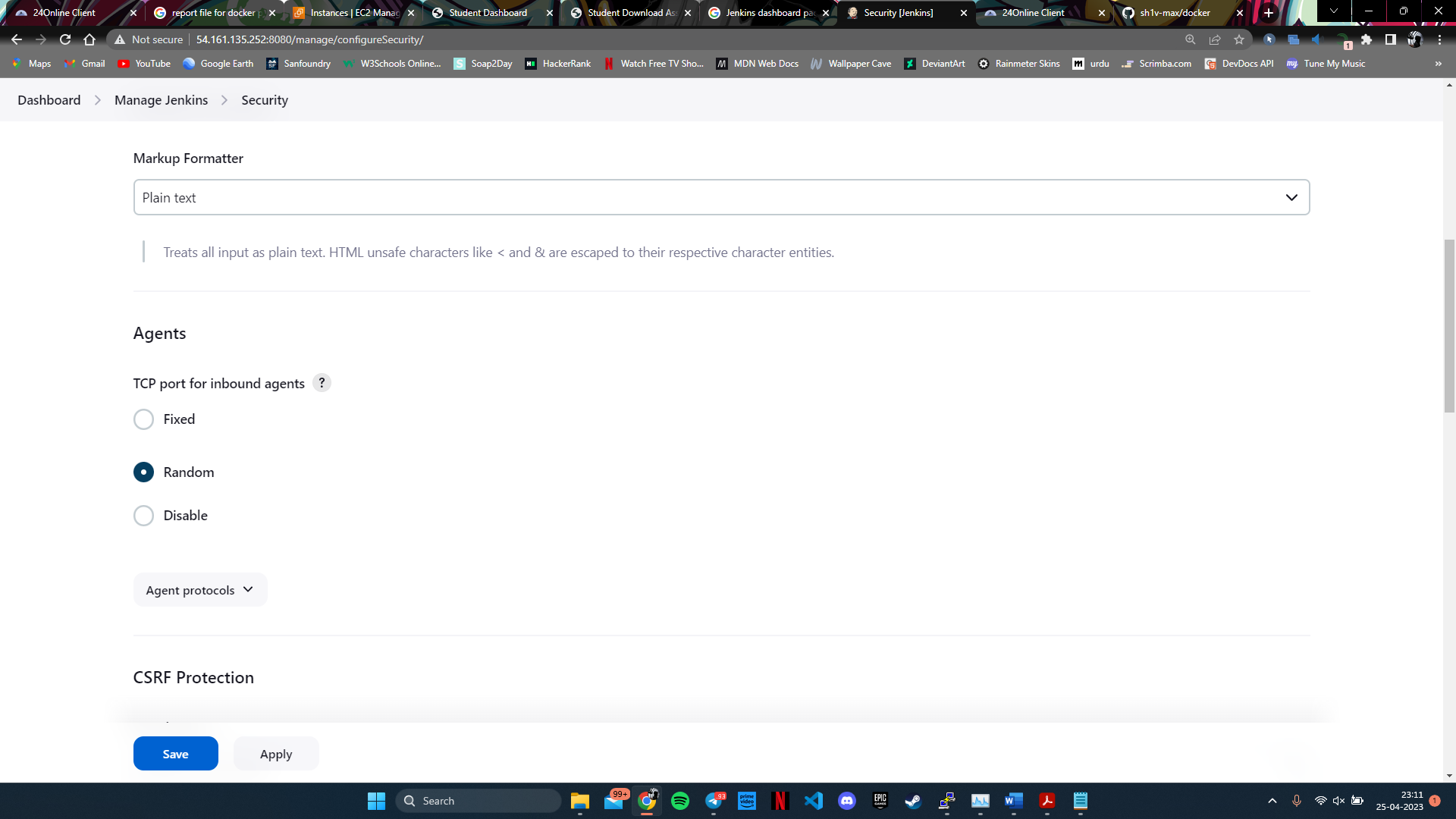


* **Creating Item/Job and agent in Jenkins:**

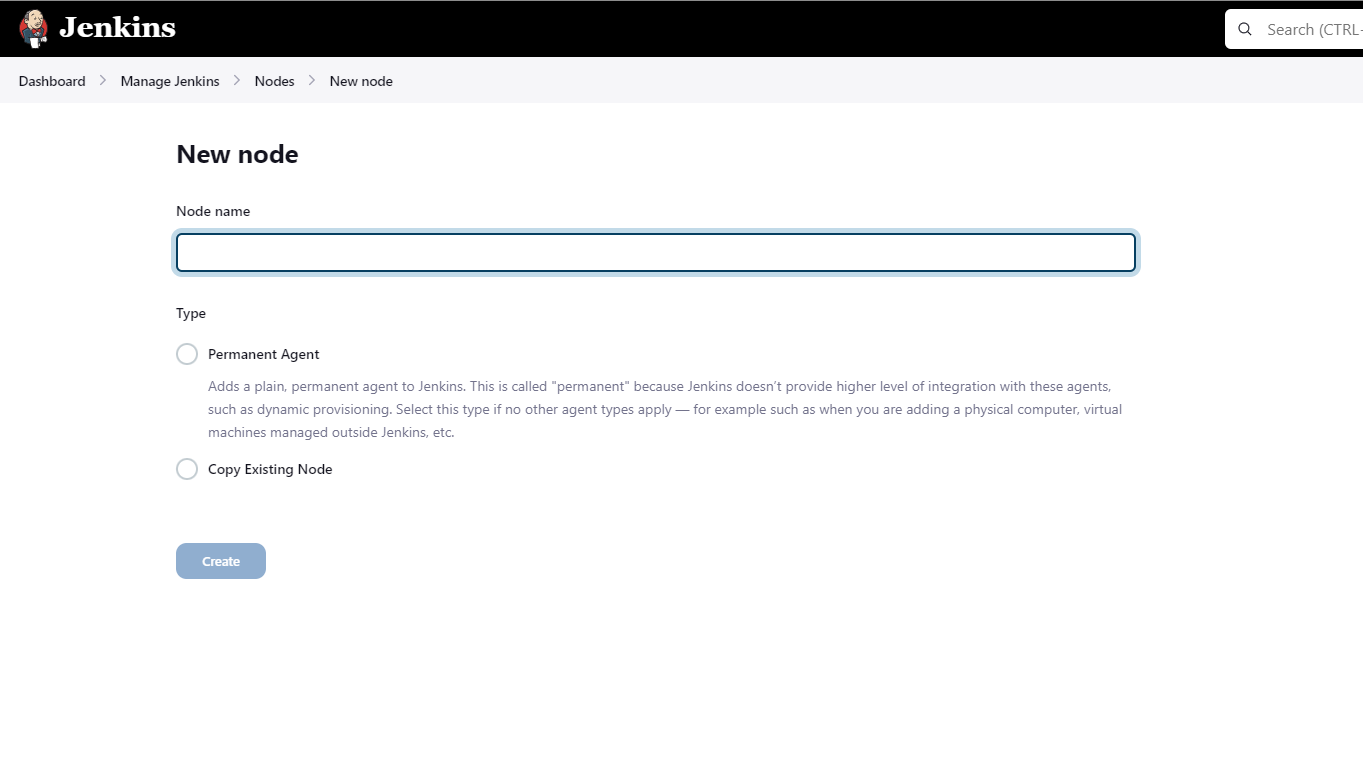
Creating agent: -



Changing agent (TPC port for inbound agent) to “random” under security.



Creating Node and Cloud.

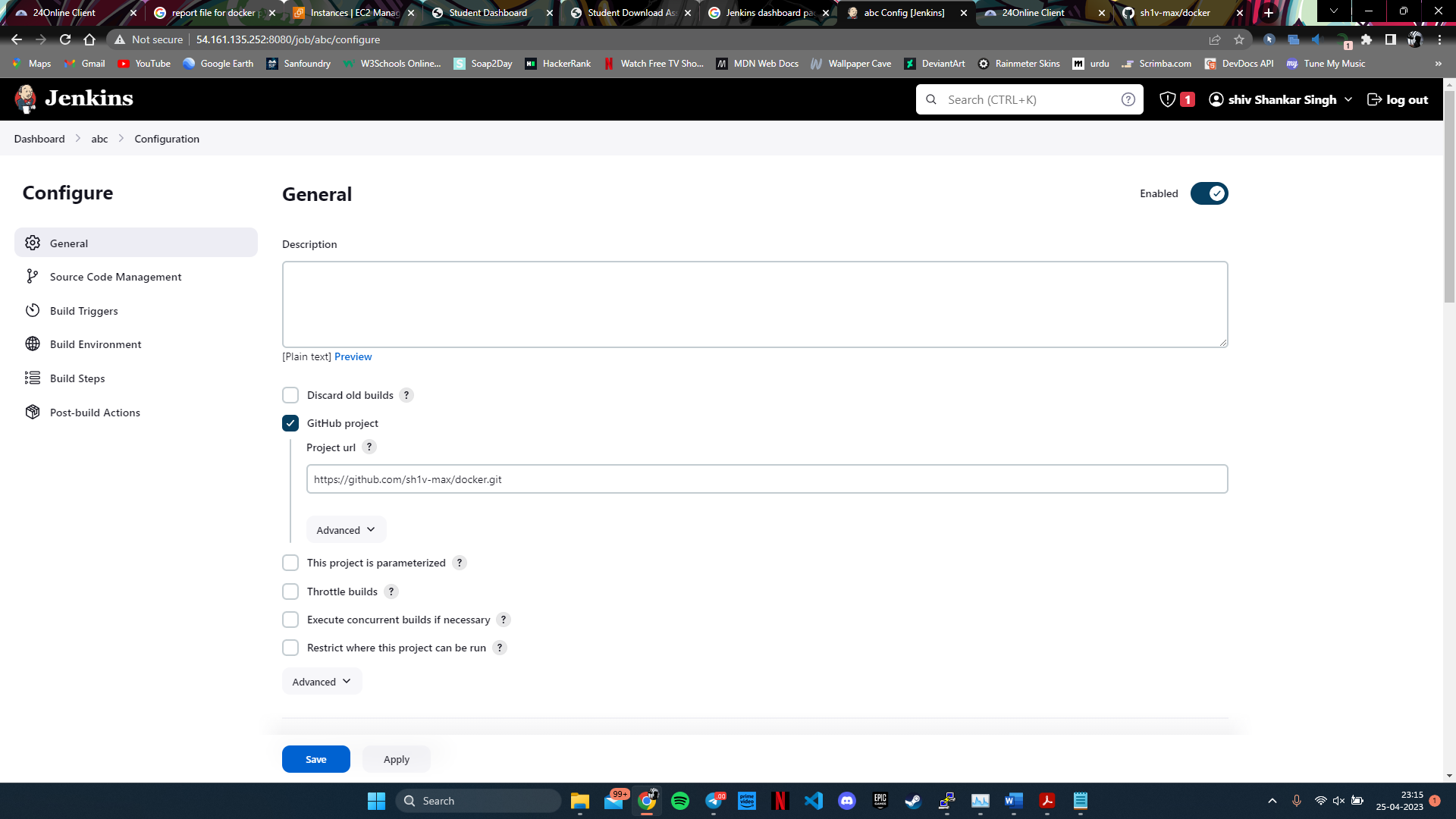


Creating New item.

Must click on freestyle project:

A screenshot of a computer

Description automatically generated

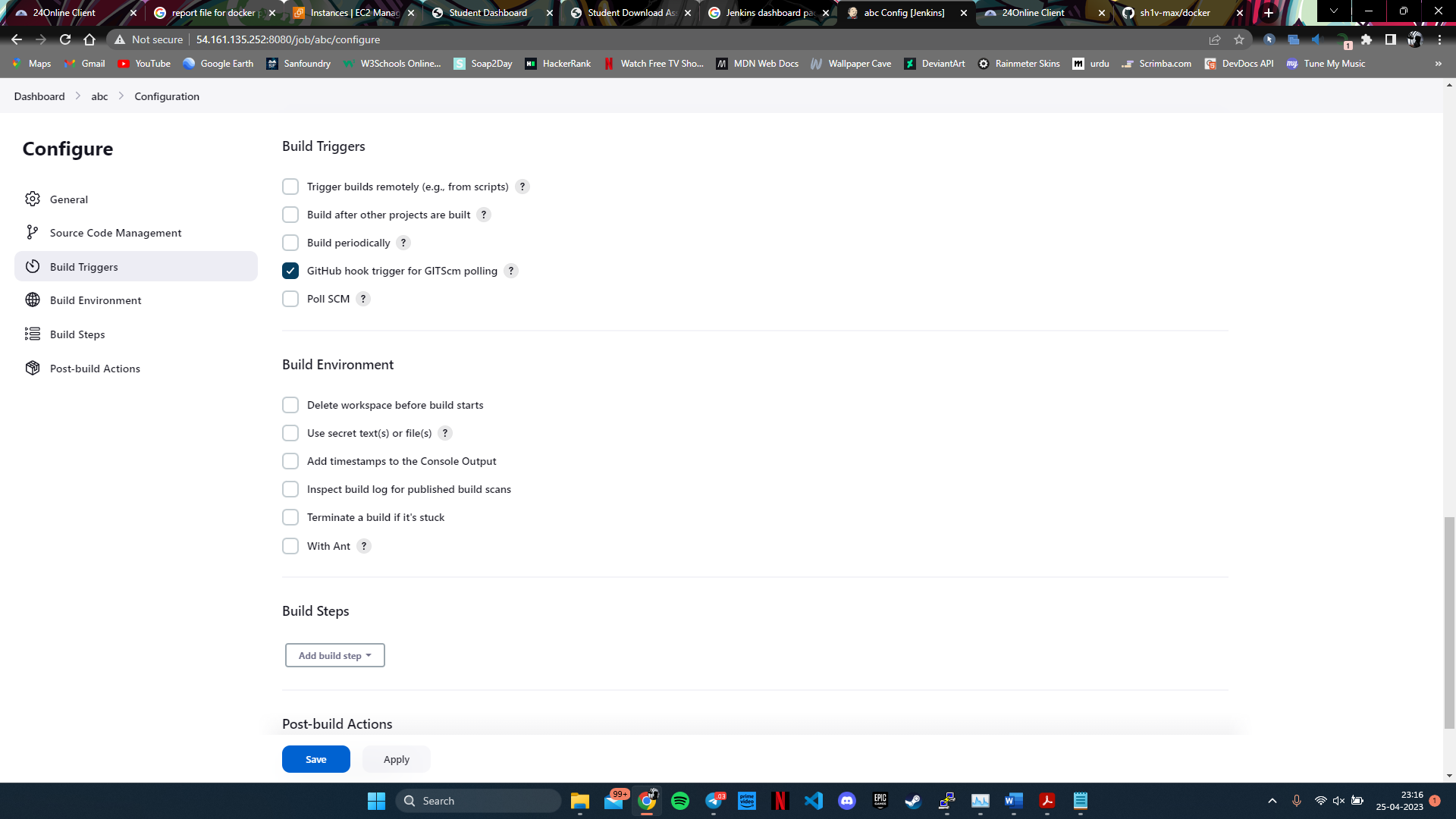


Providing GithHub URL and specifying branch to master branch:

A screenshot of a computer

Description automatically generated

Build trigger must be “GitHub hook trigger for GITScm polling”



Creating Execute shell in build step for project deployment:

Command in execute shell: -

sudo docker build . -t forder

sudo docker run -i -p 82:80 Forder

it specifies the name of folder in instance where my git is committed, in this case it is in “forder” folder.

A screenshot of a computer

Description automatically generated

Now save changes.

Graphical user interface, application

Description automatically generated

**GitHub repository link: -**

[**https://github.com/sh1v-max/docker**](https://github.com/sh1v-max/docker)

Bibliography

1. <https://aws.amazon.com/>
2. <https://www.jenkins.io/doc/>
3. <https://stackoverflow.com/>
4. <https://github.com/>
5. <https://hub.docker.com/>
6. <https://www.youtube.com/>
7. <https://docs.docker.com/engine/reference/builder/>