1. Introduction:

This document provides a step-by-step guide to setting up a CI/CD pipeline using Git, Ansible, Jenkins, and separate virtual machines for the master node (Ansible and Jenkins) and the web/DB server. The pipeline involves pulling code from a GitHub repository, triggering Ansible deployment through Jenkins, and deploying the application to the web server.

1. Prerequisites:

* Three virtual machines: Master node (Ansible and Jenkins), Web Server, and DB Server.
* Network connectivity between the machines.
* Administrator access to configure the virtual machines.

1. Setting Up the Master Node (Ansible and Jenkins):
   1. Master Node Configuration:

* Install the required operating system on the master node (e.g., Centos server).
* Assign a static IP address to the master node for easy access.
  1. Install Jenkins:

Now we will install Jenkins on the master VM.  
1. Jenkins is a java program so firstly we need to install java on the master VM.  
# yum install java-11-openjdk.x86\_64 -y

2. To install the latest stable version of Jenkins , we have to add the official Jenkins  
repository to the system. Execute the below commands to add the key and repo.  
# yum install wget -y  
# sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo  
# sudo rpm --import <https://pkg.jenkins.io/redhat/jenkins.io-2023.key>

3. Now install, start and enable jenkins by below commands  
# yum install jenkins -y  
# systemctl start jenkins  
# systemctl enable Jenkins

* 1. Install Ansible:

Now we will install Ansible on the master VM.  
1. Firstly we need to install the epel-release package so that we can install ansible from that  
package. For this we need to run the below command:  
# sudo yum install epel-release  
2. Now we will install ansible by below command:  
# yum install ansible

* 1. Ansible hosts:

We will add ansible clients which is our Web Server VM, Database Server VM ip address in  
ansible configuration file.  
# sudo vim /etc/ansible/hosts  
[webservers]  
192.168.20.15  
[dbservers]  
192.168.20.14

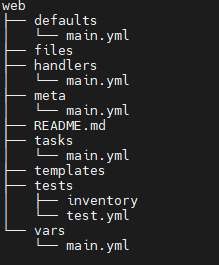
* 1. SSH key copy master to client:

We need to copy the ssh key of the master server in ansible client machines. So that the Ansible server to its nodes communication will be password less.  
# ssh-keygen  
# ssh-copy-id root@192.168.20.225 [Web Server IP]  
# ssh-copy-id root@192.168.20.144 [Database Server IP]

1. Setting Up the Web Server and DB Server:
   1. Web Server Configuration:

Now we will create an ansible-playbook to install, start mandatory services in node.

Firstly we need to create files (webserver and db server) and a directory (roles) in the same directory. Under the “roles” directory we created another directory which is called “web” and “db”. Under this “web” and “db” directory we created four directory (vars, tasks, templates, files). We created “main.yml” files in the vars and tasks directory. In the files of “db” directory we created two files “db-load-script.sql” and “my.cnf”.



* Now in the “webserver.yml” file we have added the below lines:

---

- name: Web Service

hosts: web

roles:

- web

* **I have mentioned the below lines in “roles/web/tasks/main.yml”.**

- name: Installation Services

yum:

name:

- libselinux-python

- libsemanage-python

- httpd

- git

- php

- php-mysql

state: installed

tags: install

- name: Start firewalld

service: name=firewalld state=started enabled=yes

tags: start firewalld

- name: Insert firewalld rule for httpd

firewalld: port={{ httpd\_port }}/tcp permanent=true state=enabled immediate=yes

tags: enable httpd port

- name: insert firewalld rule for mysql

firewalld: port={{ mysql\_port }}/tcp permanent=true state=enabled immediate=yes

tags: enable mysql port

- name: Set index.php as the default page

replace:

path: /etc/httpd/conf/httpd.conf

regexp: 'DirectoryIndex index.html'

replace: '#DirectoryIndex index.html \nDirectoryIndex index.php'

tags: rename html file

- name: http service state

service: name=httpd state=started enabled=yes

tags: httpd start

- name: Copy the code from repository

git: repo={{ repository }} dest=/var/www/html/ force=yes

tags: clone

- name: replace ip in index.php file

command: sed -i 's/172.20.1.101/192.168.20.144/g' /var/www/html/index.php

tags: replace IP

* **I added the below lines in “roles/web/vars/main.yml” file**  
  ---  
  httpd\_port: 80  
  mysql\_port: 3306  
  repository: <https://github.com/mdnahidulislam1/Crispy_Kitchen.git>

* 1. DB Server Configuration:

• Install the required operating system on the DB server (e.g., Centos 7).

• Assign a static IP address to the DB server for easy access.

• Install and configure the desired database server software (e.g., MySQL).

* I have mentioned the below lines in “roles/db/tasks/main.yml”.

# tasks file for db

- name: Installation Services

yum:

name:

- libselinux-python

- libsemanage-python

- mariadb-server

- MySQL-python

- php-mysql

state: installed

tags: install

- name: Start firewalld

service: name=firewalld state=started enabled=yes

tags: start firewalld

- name: Insert firewalld rule for mysql

firewalld: port={{ mysql\_port }}/tcp permanent=true state=enabled immediate=yes

tags: enable mysql port

- name: Restart firewalld

service: name=firewalld state=reloaded enabled=yes

tags: restarted firewalld

- name: Copy Mysql configuration file

copy: src=files/my.cnf dest=/etc/my.cnf

tags: mysql conf copy

- name: Start MariaDB Service

service: name=mariadb state=started enabled=yes

tags: start mariadb

- name: Create Application Database

mysql\_db: name={{ dbname }} state=present

tags: create database

- name: Create Application DB User

mysql\_user: name={{ dbuser }} password={{ dbpassword }} priv=\*.\*:ALL host='192.168.20.144' state=present

tags: create user

- name: Move db-load-script to db host

copy:

src: files/db-load-script.sql

dest: /tmp/db-load-script.sql

tags: copy sql

- name: Load Inventory Data

shell: mysql -f < /tmp/db-load-script.sql

tags: run sql

* **I have mentioned the below lines in “roles/db/vars/main.yml”**.

---

mysql\_port: 3306

dbname: ecomdb

dbuser: ecomuser

dbpassword: ecompassword

* **I have mentioned the below lines in “roles/db/files/db-load-script.sql”**

GRANT ALL PRIVILEGES ON \*.\* TO 'ecomuser'@'192.168.20.225' IDENTIFIED BY

'ecompassword' WITH GRANT OPTION;

FLUSH PRIVILEGES;

USE ecomdb;

CREATE TABLE products (id mediumint(8) unsigned NOT NULL auto\_increment,Meal\_time\_name varchar(255) default NULL,Name varchar(255) default NULL,Price varchar(255) default NULL, ImageUrl varchar(255) default NULL,PRIMARY KEY (id)) AUTO\_INCREMENT=1;

INSERT INTO products (Meal\_time\_name,Name,Price,ImageUrl) VALUES ("Morning","Morning Fresh","12","c-1.png"),("Breakfast","Tooplate Soup","20","c-2.png"),("Night","Premium Steak","30","c-3.png"),("Morning","Seafood Set","25","c-5.png"),("Dinner","Burger Set","90","c-6.png"),("Breakfast","Burger Set","20","c-7.png");

* **I have mentioned the below lines in “roles/db/files/my.cnf”**

[mysqld]

datadir=/var/lib/mysql

socket=/var/lib/mysql/mysql.sock

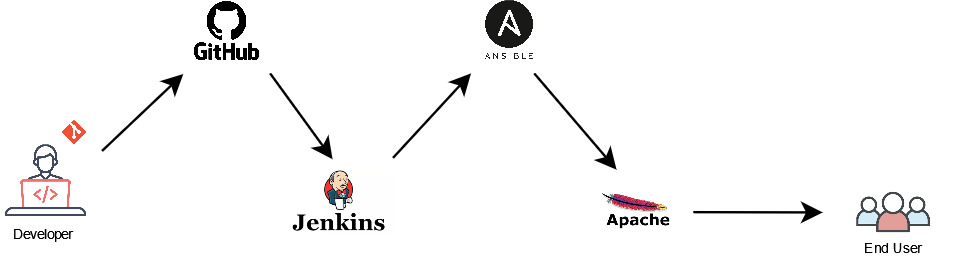
symbolic-links=0

[mysqld\_safe]

log-error=/var/log/mariadb/mariadb.log

pid-file=/var/run/mariadb/mariadb.pid

!includedir /etc/my.cnf.d

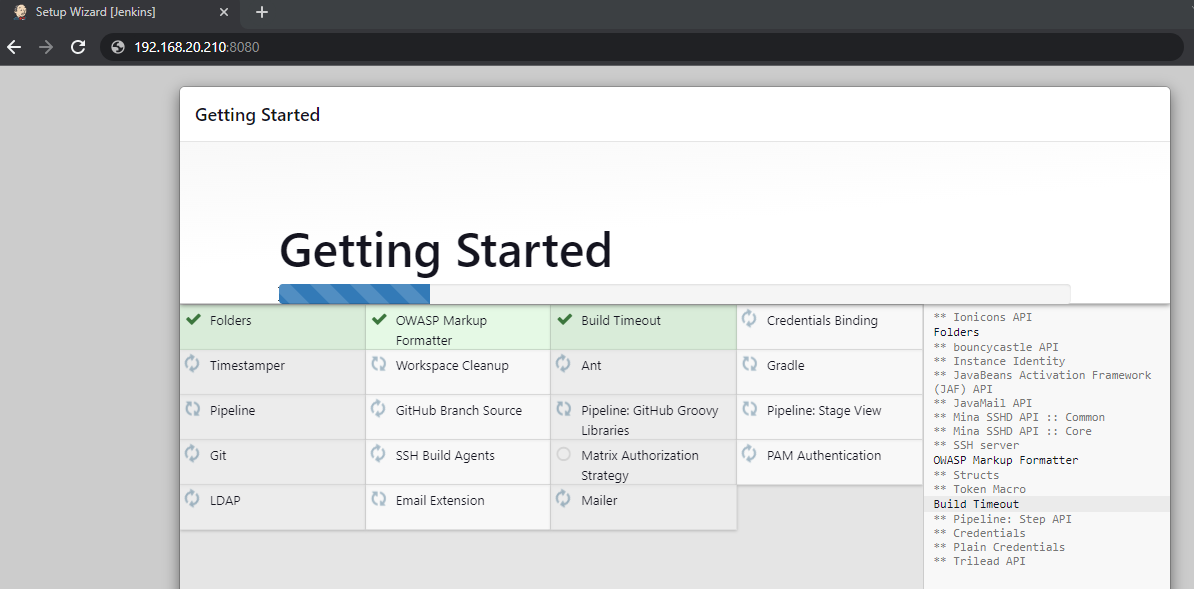


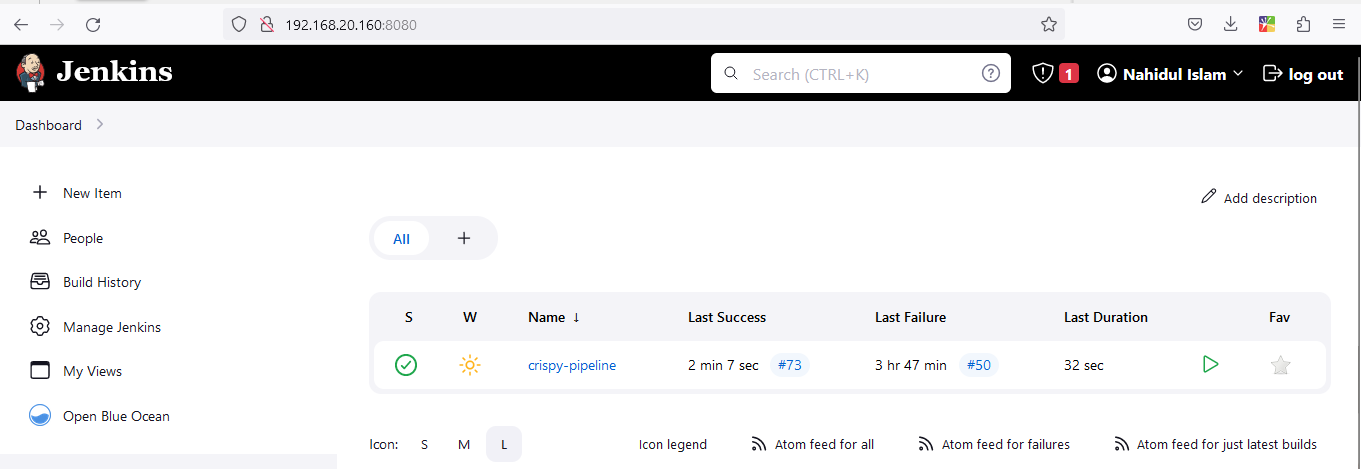
1. **Pipeline Workflow**:
   1. Configure Jenkins Pipeline Job:

• Create a Jenkins pipeline job to define the CI/CD workflow.

• Specify the pipeline script using the Pipeline syntax.

• Set up stages and steps to execute the necessary tasks.

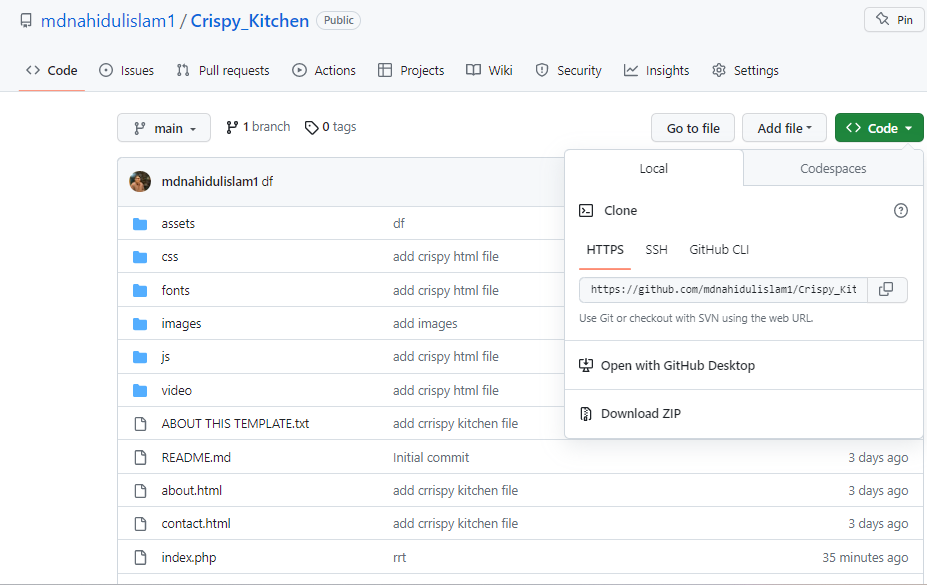




* 1. Integration with Git:

• Configure the Jenkins pipeline job to fetch the latest code changes from the GitHub repository.

• Utilize **Poll SCM** to automate the pipeline execution whenever there are new commits.

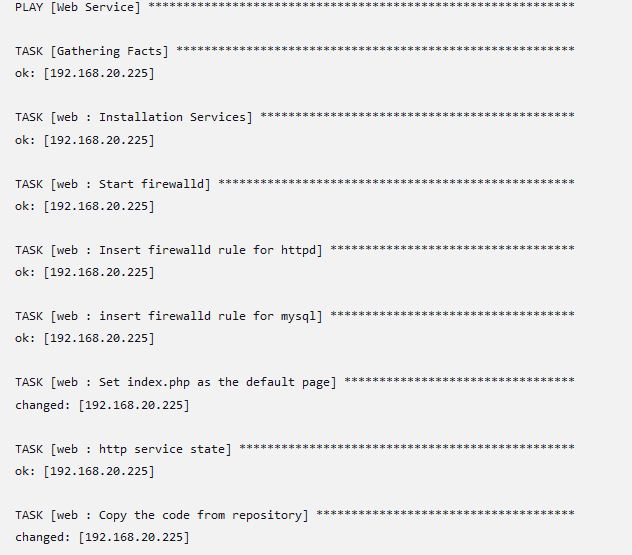


* 1. Deployment using Ansible:

• Utilize the Ansible plugin in Jenkins to trigger Ansible playbooks for deployment.

• Write Ansible playbooks to configure the web server and deploy the application code from the GitHub repository.

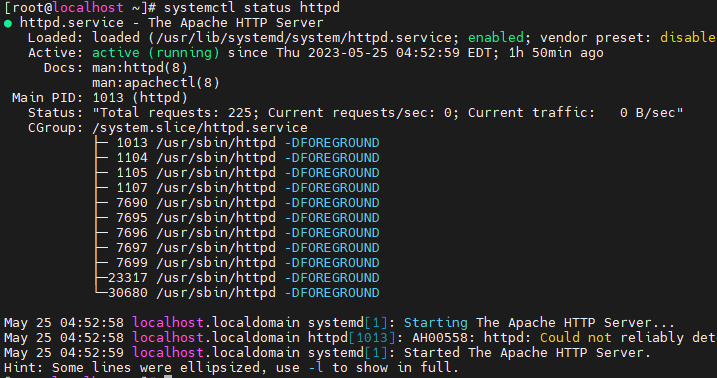
• Ensure the necessary Ansible playbooks are accessible from the Jenkins pipeline.



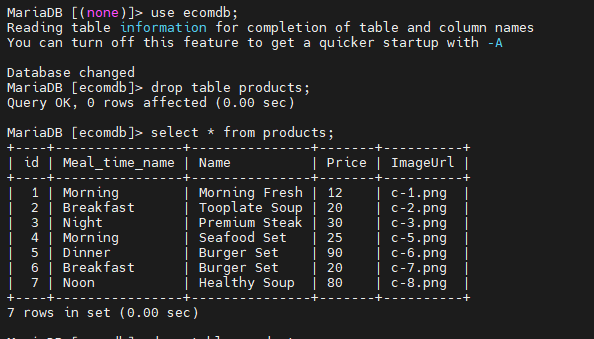
* 1. Deployment to the Web Server:

• Configure the pipeline to execute Ansible playbooks that deploy the application code to the web server.

• Ensure the Ansible inventory file specifies the target web server.



* 1. Show DB server Database:
* Ansible install mysql in db server and load the sql file.

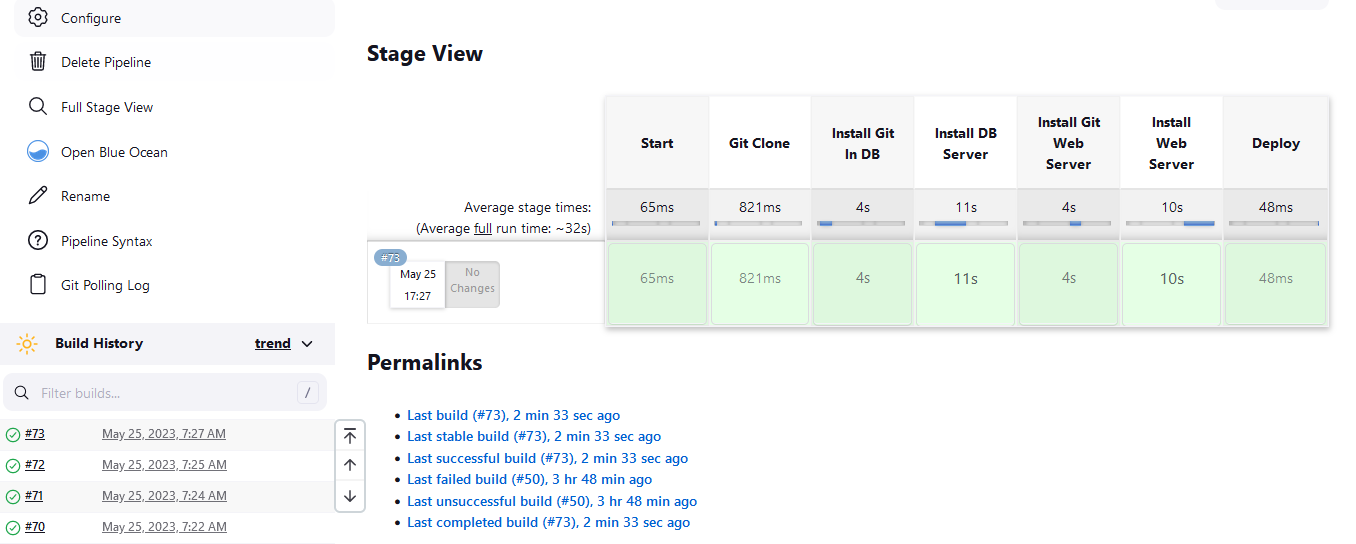


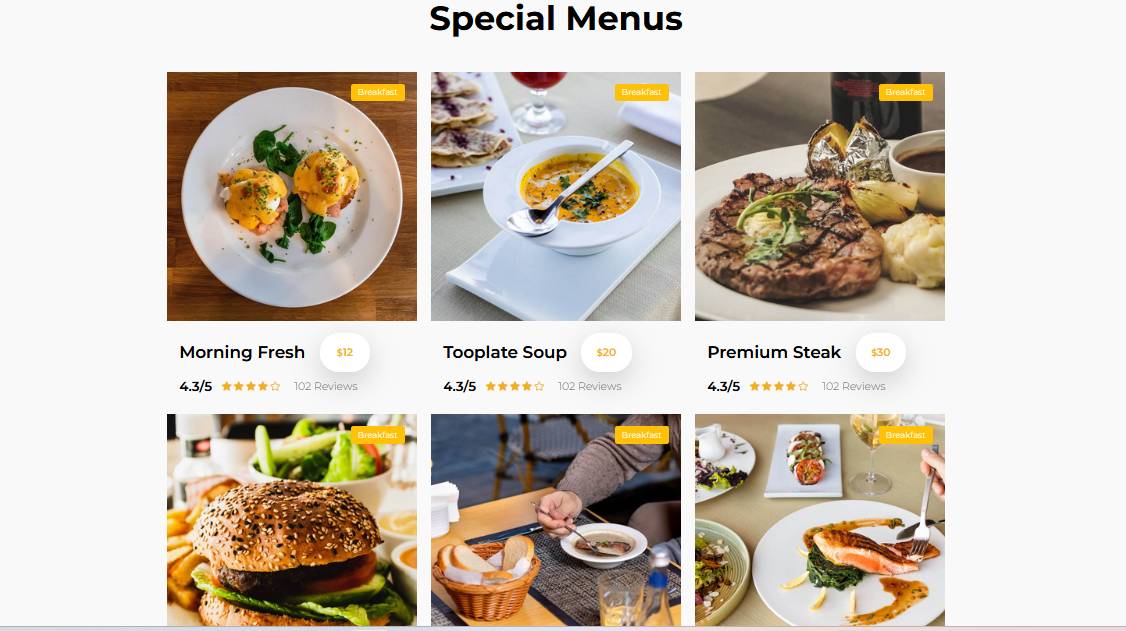
5.6 Pipeline Execution and Monitoring:

• Start the Jenkins pipeline job to trigger the CI/CD pipeline.

• Monitor the pipeline execution in the Jenkins web interface, ensuring all stages complete successfully.

• Check the web server to verify the application has been deployed successfully.





1. Conclusion: This document has provided a comprehensive guide to setting up a CI/CD pipeline using Ansible, Jenkins, and separate virtual machines for the master node, web server, and DB server. Following the steps outlined will allow me to automate the deployment of your application from GitHub to the web server through the Ansible and Jenkins integration.