Name:-Rahul Shinde

Date:______

DSO

Experiment No:-4

Aim:- To deploy and test Java/Web/Python application on Jenkins Server.

Date of Perform

Date of submission

Sub Incharge

IoT-56

Name:-Rahul Shinde

Date:

DSO

Experiment No:- 4

Aim:- To deploy and test Java / Web / Python application on Jenkins Server.

Objective:- To learn how to set up a continuous Integration (CI) and continuous Delivery (CD) pipeline using Jenkins for Python application.

Pre-requisites:-

- A Working of Jenkins Installation (locally or on a server)
- A Python application code hosted on a version control system (eg. GitHub)
- Basic knowledge of python, Git and Jenkins.

Theory:-

Jenkins is an open source automation platform that helps to automate the parts of s/w development related to building, testing and the deploying, facititation continoous Integration and Continuous Delivery.

-Continuous Integration

It is a practice of automating the integration of code changes from multiple of contributors into a single softume project.

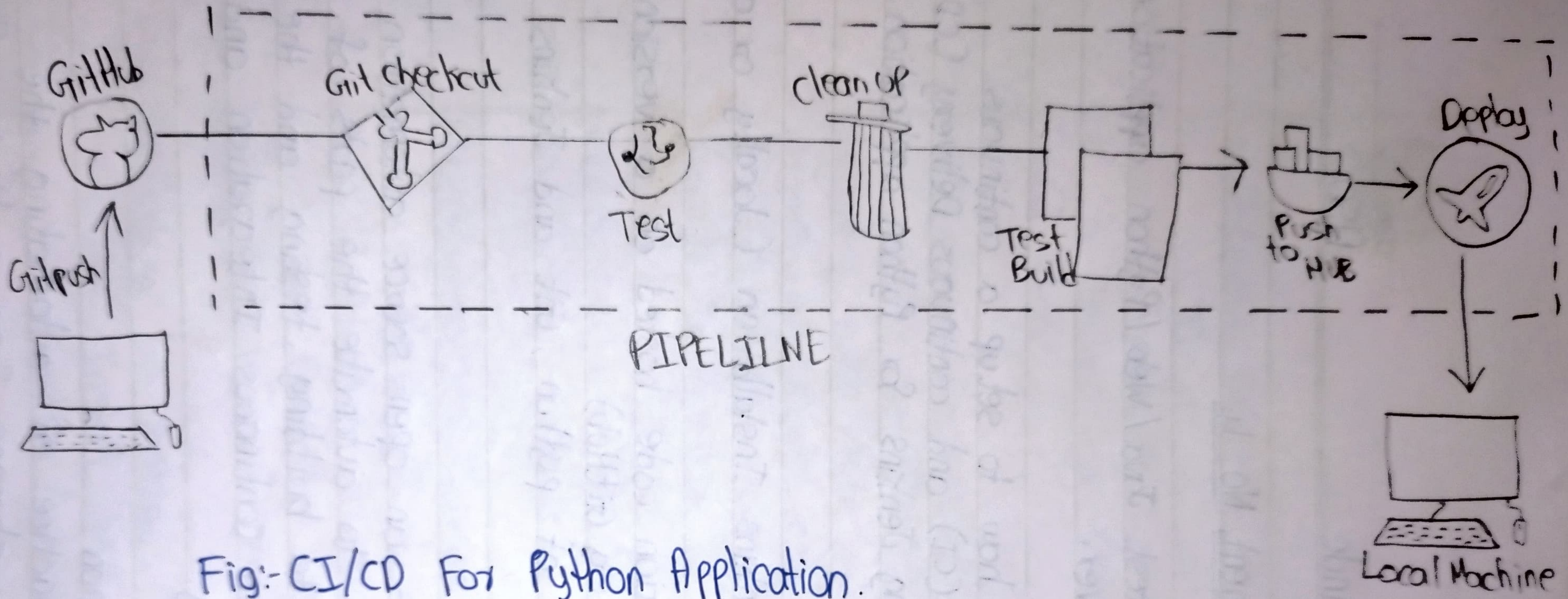

Fig:- CI/CD For Python Application.

- Continuous Delivery

It is an extension of continuous Integration which automatically deploys all the code changes to a testing or testing environment after the build stage.

- Continuous Deployment

It is goes one step further than continuous delivery with the practice, every change that passes all stages of your production of pipeline is released to your customers and there's no Human Intervention.

- Experiment setps to deploy python application on Jenkins Server.

① Understanding Jenkins and CI/CD:-

Read about Jenkins and the concepts of Continuous Integration (CI), continuous Delivery (CD) and Continous Deployment (CD) from the provided information.

② Setting up Jenkins:-

Install Jenkins and configure it to run on your system / server.

③ Configuring Jenkins for Python CI/CD:-

→ Install the necessary plugins, such as the Docker pipeline plugin.

Date:

→ Configure credentials for GitHub and Docker Hub in the Jenkins.

④ Preparing the python application:-
Create a simple flask application in Python.

⑤ Creating the Jenkins pipeline:-
Write a Jenkins file that defines the pipeline for building, testing, and deplaying flask application.

⑥ Building the pipeline:-
- Create a Jenkins Job and paste the Jenkins file contents into the pipeline script editor.
- Trigger a build to initiate the pipeline.

⑦ Validation and Testing:-
- Verify that pipeline builds the application successfully.
- Confirm that the Docker image is pushed to Docker-Hub.
- Ensure that the flask application is running on the specified port.

Conclusion:- Hence, we successfully Deploy & Test Python Application on the Jenkins Server.