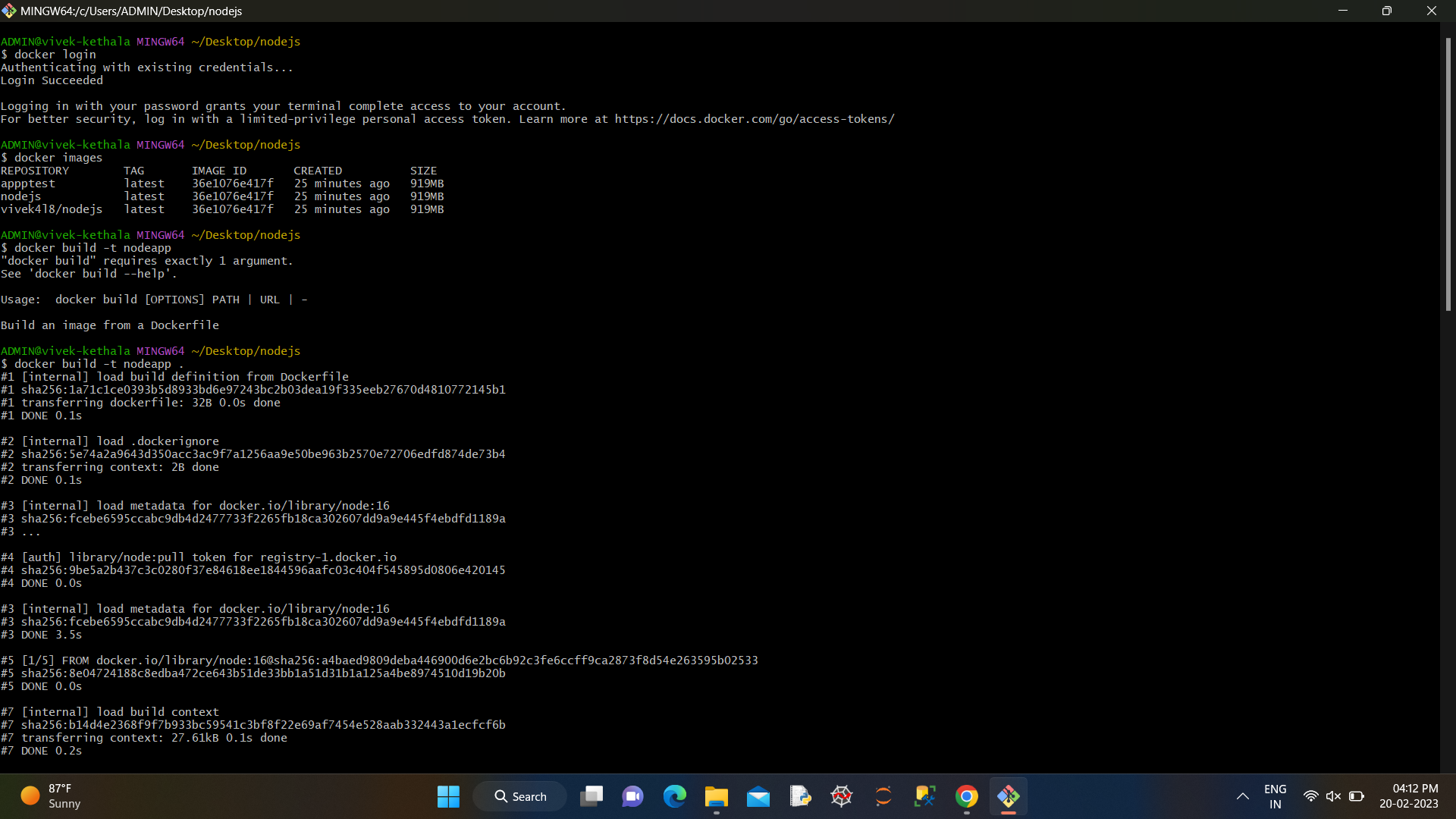
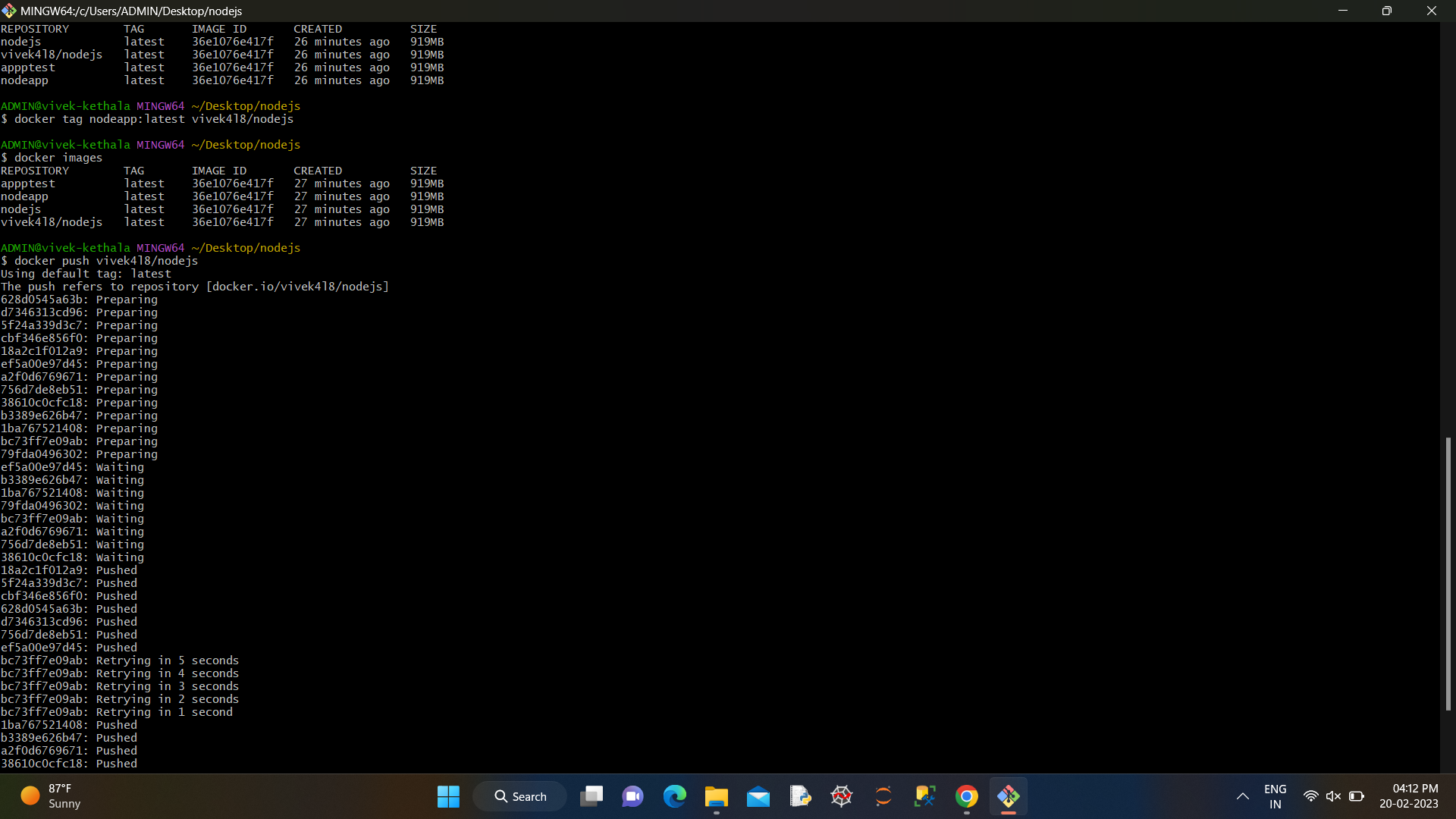
Q1.Create an docker container and image out of dockerfile for following application:

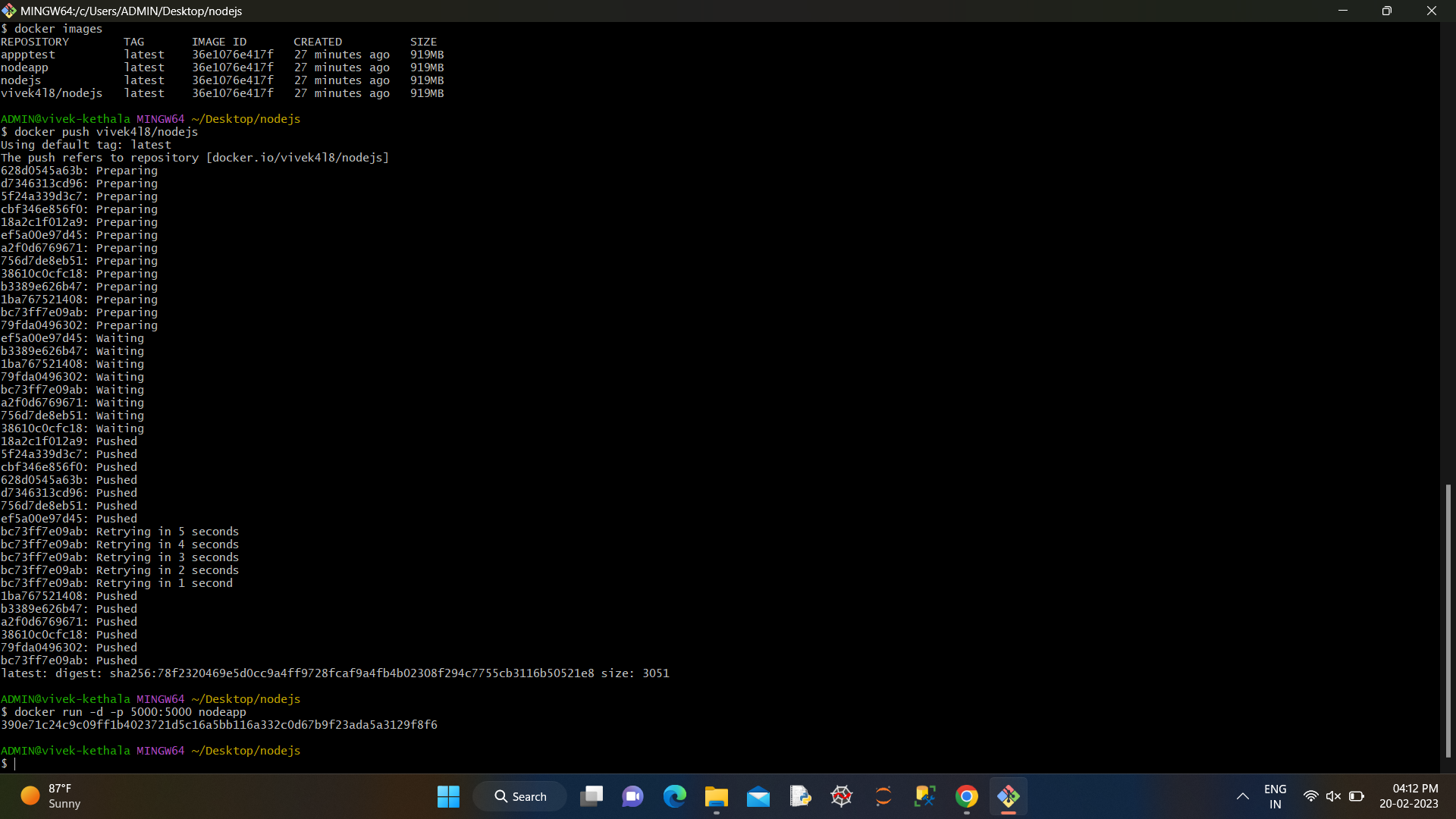
* Python Flask
* Nodejs

 Kindly, provide the details of steps written in dockerfile and why the steps are required with explanation.

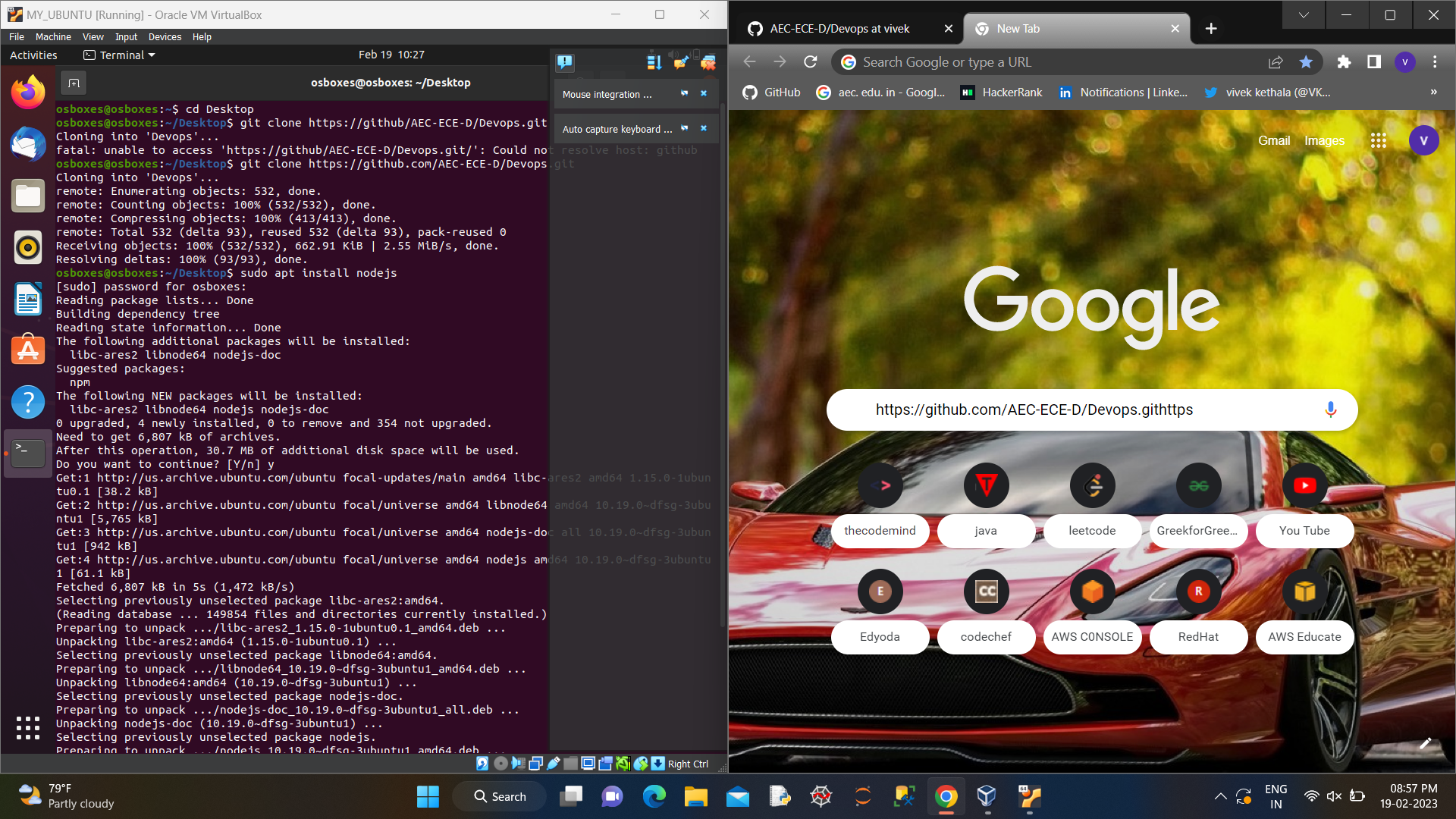
All the steps from creating images to container and running the container should be provided in a screenshot , where your username can be seen in the terminal.

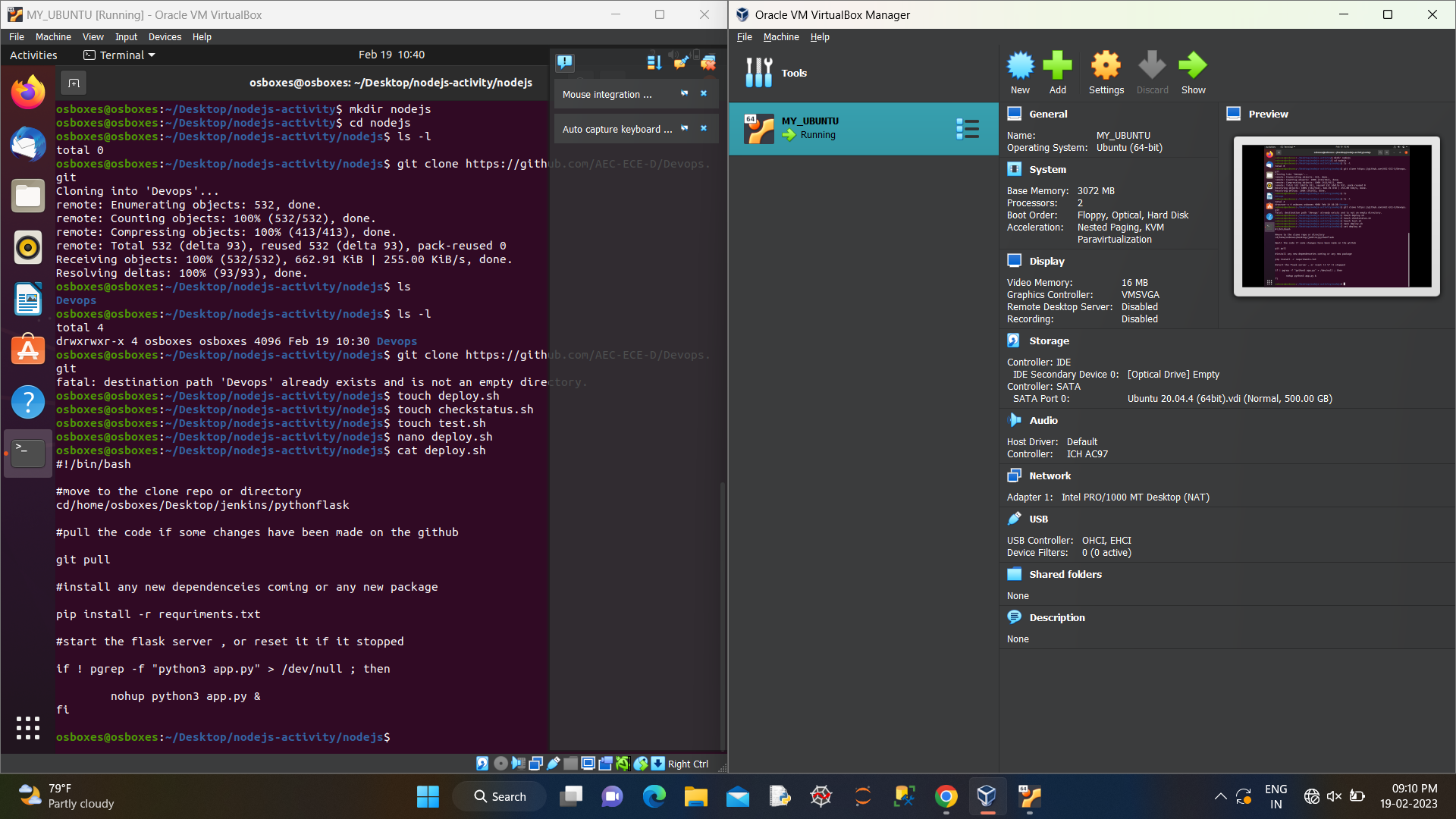


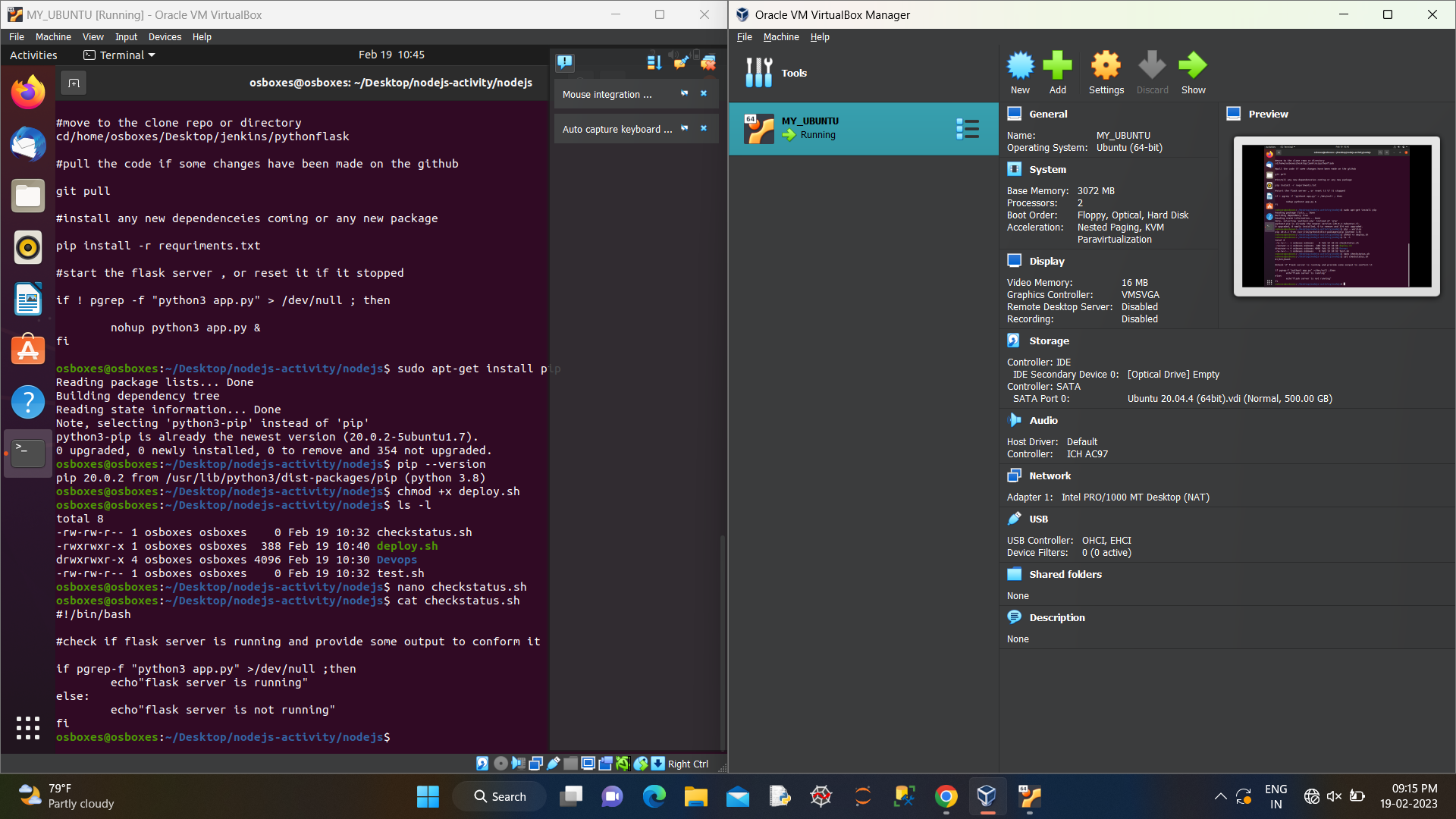


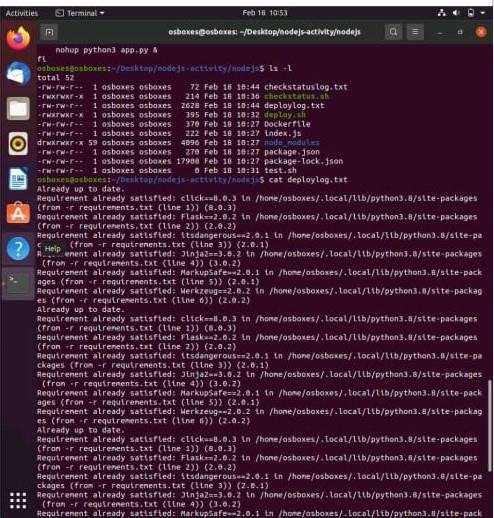


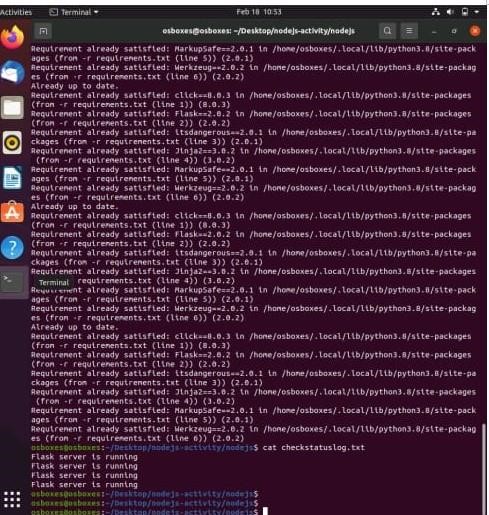
Q2.Create a CI-CD pipeline for a Nodejs Application in jenkins and all the steps involved in it should be given in   a screenshot  and your jenkins username must be visible in the screenshot.











Q3. Create documentation of how you are going to create a CI-CD pipeline for python applications.

In the documentation you have mentioned each step which should be taken to configure the CI-CD pipeline for a python application including the plugins you are using  and the global tool configuration.

ANS:

Creating a Continuous Integration/Continuous Delivery (CI/CD) pipeline for Python applications is an essential part of modern software development. The pipeline ensures that changes made to code are tested, integrated, and delivered to production as quickly and efficiently as possible. In this documentation, we will cover the steps to create a CI/CD pipeline for a Python application.

TOOLS USED:

1)Git-version control system

2)GitHub-hosting platform

3)Jenkins-CI/CD automation server

4)Docker-Containerization technology

Steps:

1. **Setup Git Repository** - Create a Git repository on GitHub and clone it to your local machine.
2. **Setup Jenkins** - Download and install Jenkins on a server or a local machine. Once installed, access Jenkins in a web browser and configure the necessary plugins required for Python development, such as **Git Plugin**, **Docker Plugin**, **Pipeline Plugin**, and **Python Plugin**.
3. **Setup Jenkinsfile** - Create a **Jenkinsfile** in the root of the Git repository. This file contains the instructions for the CI/CD pipeline. Below is an example of a **Jenkinsfile**:
4. **Configure Global Tool** - Configure the Python environment on the Jenkins server. Navigate to the Jenkins Global Tool Configuration page and add a new installation of Python.
5. **Create Dockerfile** - Create a **Dockerfile** in the root of the Git repository. This file contains the instructions for building a Docker image of the Python application. Below is an example of a **Dockerfile**:
6. **Create Unit Tests** - Create unit tests for the Python application in the **tests** directory. These tests should cover all the functionality of the application and be runnable in a Docker container.
7. **Commit Changes** - Commit the changes made to the Git repository.
8. **Setup Webhook** - Configure a webhook in the Git repository to trigger the Jenkins pipeline when changes are pushed to the repository.
9. **Run Pipeline** - Push the changes made to the Git repository to trigger the Jenkins pipeline. Jenkins will automatically build the Docker image, run the unit tests, and deploy the application to a Docker registry if all tests pass.

That’s it! We have a fully functional CI/CD pipeline for python applications