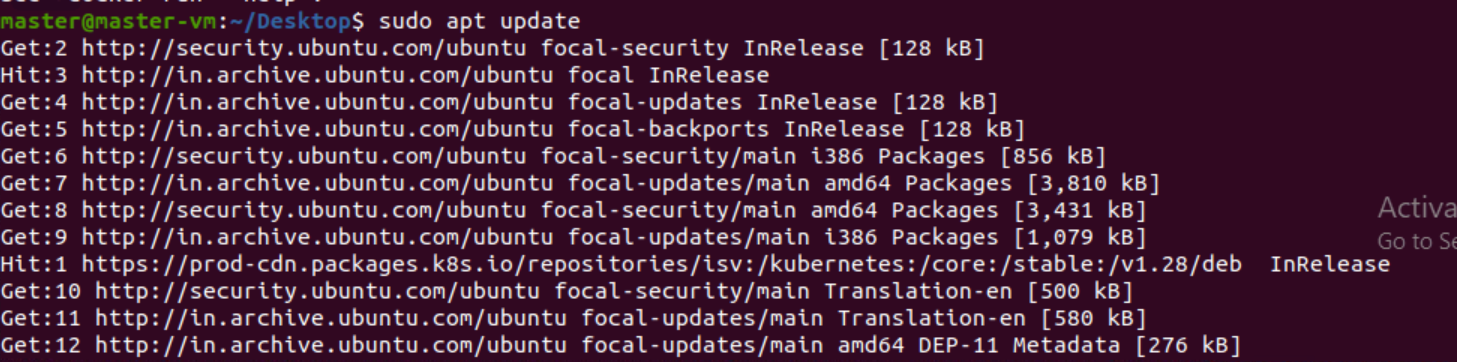
**Kubernetes and CI/CD Setup for Flask App**

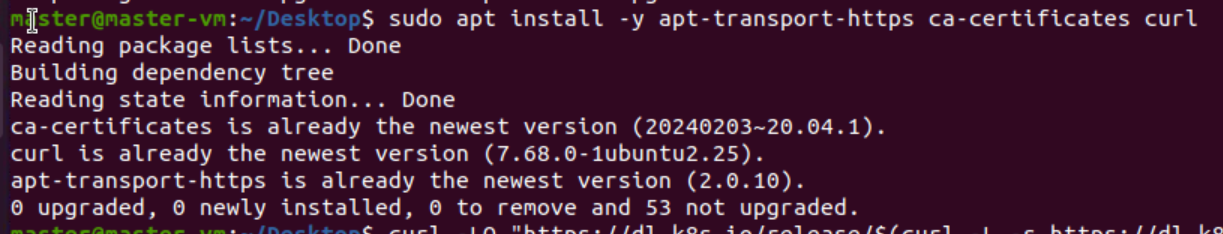
This guide covers the steps to install Kubernetes on Ubuntu, deploy a Flask app, and set up a Jenkins pipeline for continuous integration and deployment (CI/CD). Additionally, it includes details about the deployment process using Minikube and Kubernetes.

**1. Install Kubernetes on Ubuntu**

**a. Install Dependencies**

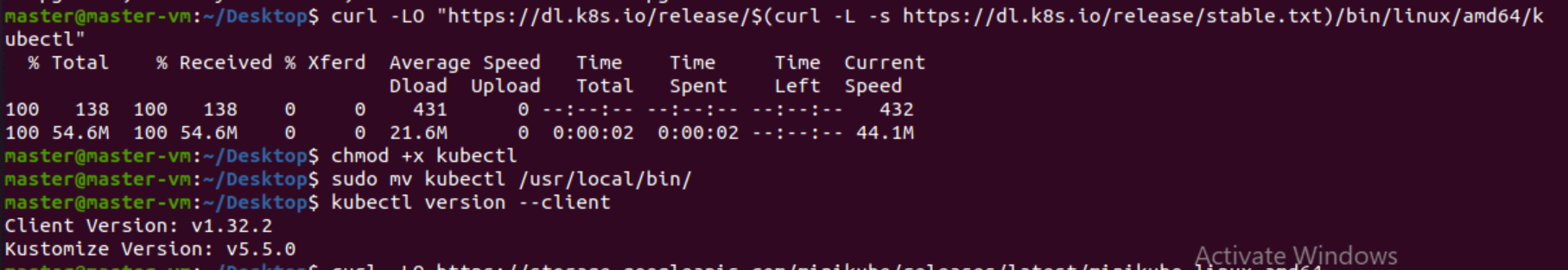
To begin with, update your system and install the required dependencies.

****

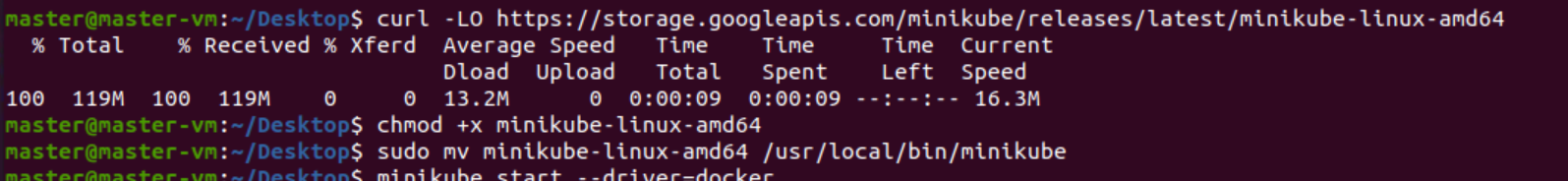


**b. Install kubectl**

Next, install the kubectl command-line tool to interact with Kubernetes.

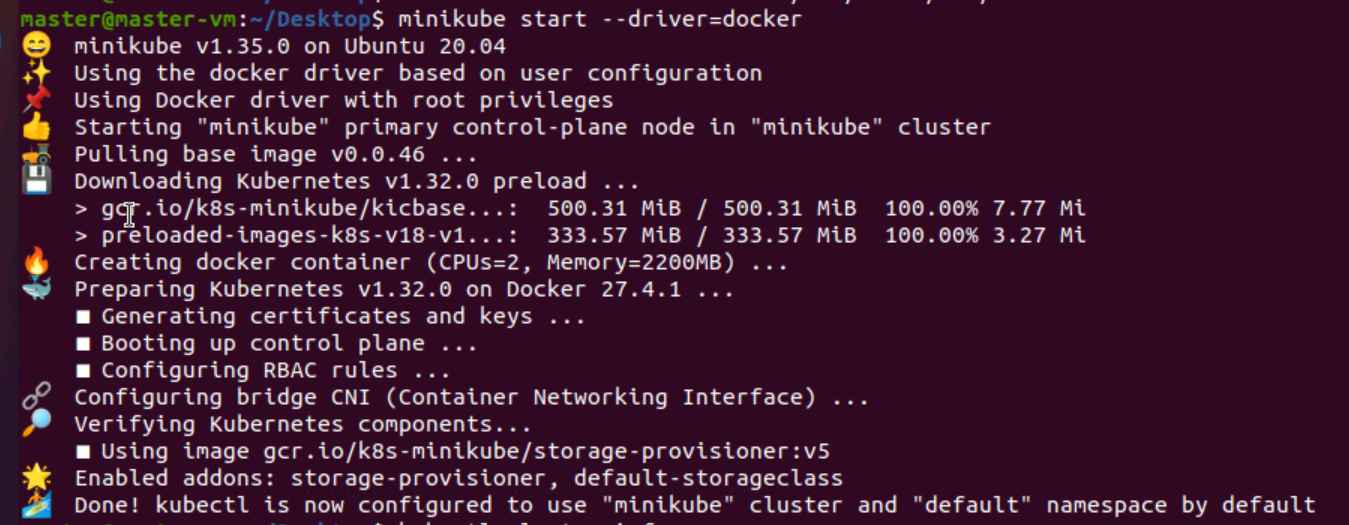
**c. Install Minikube**

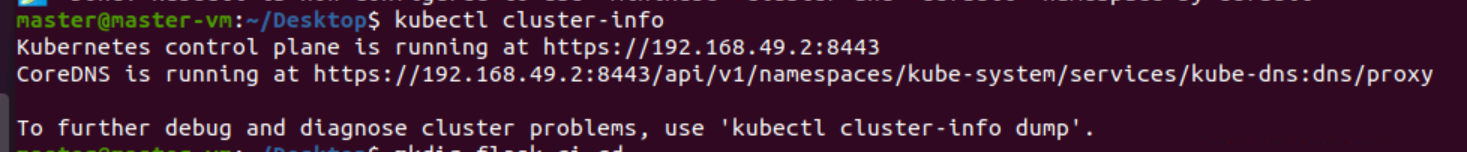
Now, install Minikube to create a local Kubernetes cluster.



**d. Start Minikube**

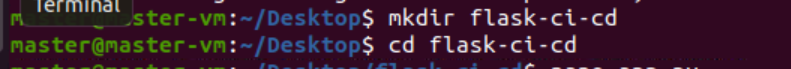
Start the Minikube cluster using Docker as the driver.

**e. Check Status**

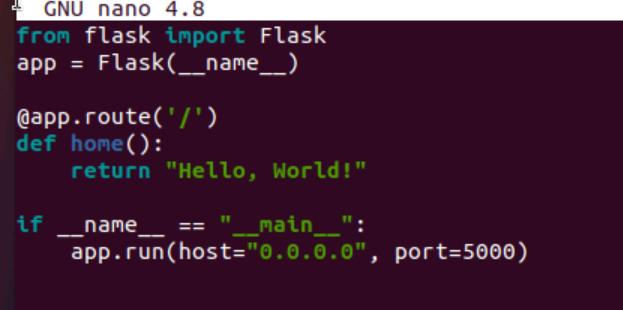


**2. Create Project**

**a. Create Folder**

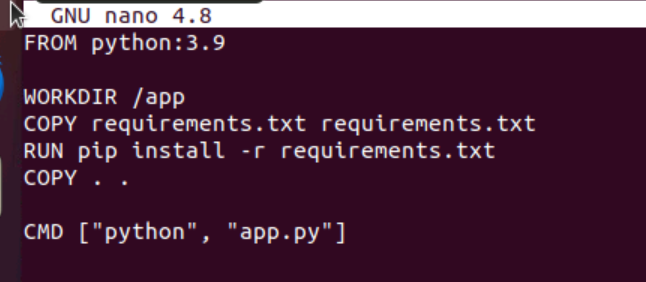
**b. Create Flask App**

Create a simple Flask application (app.py).

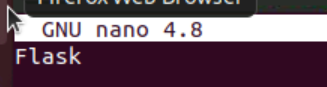


**c. Create Docker File**

Create a Dockerfile for the Flask app.

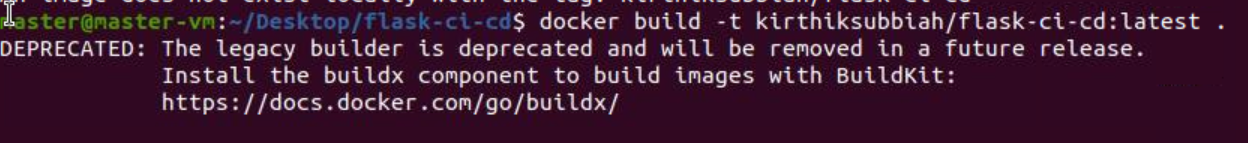
**d. Create requirements.txt**

Create the requirements.txt file to list the required dependencies.

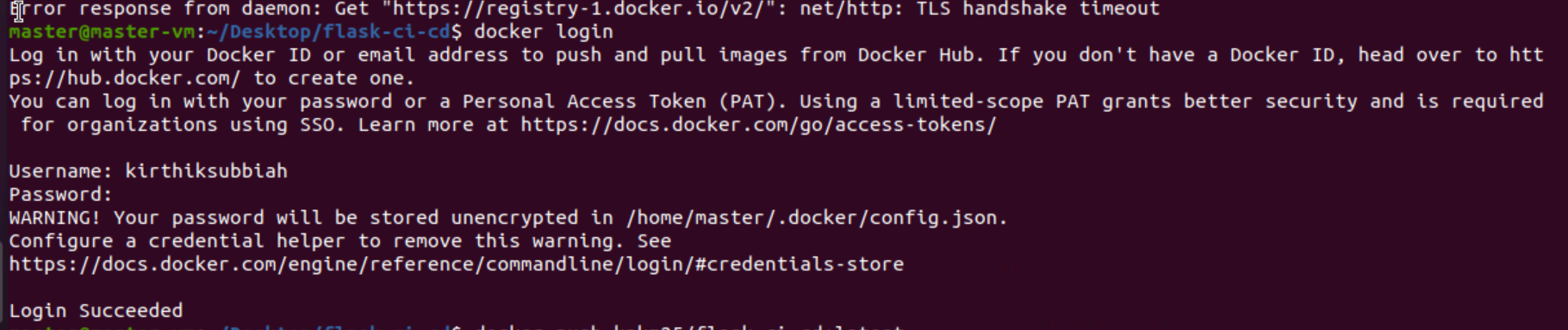


**3. Build and Push Docker Image**

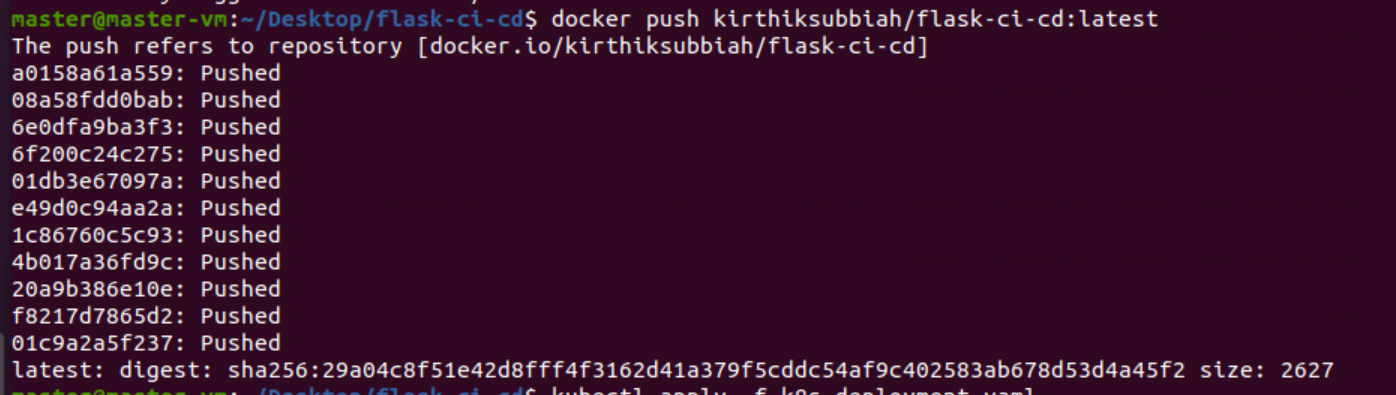
**a. Build Docker Image**

****

**b. Login to Docker Hub**

****

**c. Push Docker Image**



**4. Connect Kubernetes to Docker**

**a. Create Docker Hub Secret**

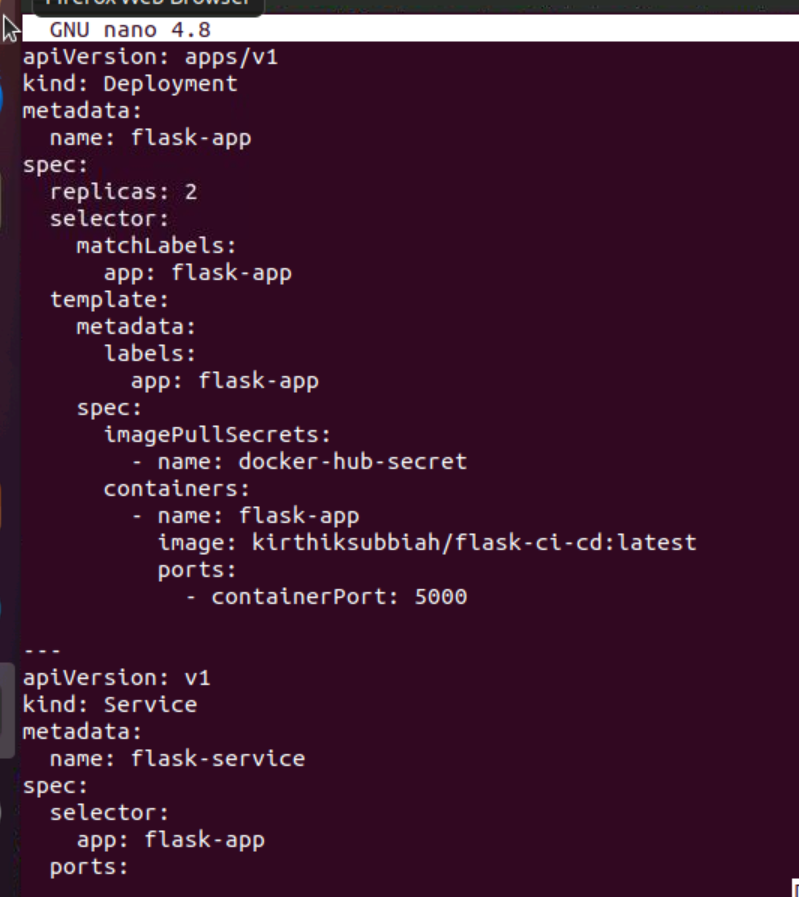
Create a Kubernetes secret to authenticate Docker with your Docker Hub credentials.

****

**5. Kubernetes Deployment**

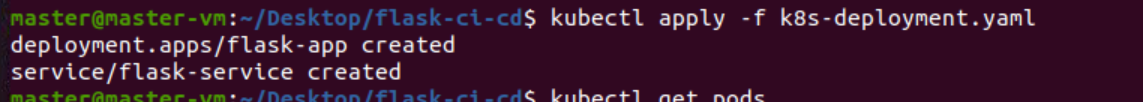
**a. Create Deployment YAML**

Create the k8s-deployment.yaml file to define your Kubernetes deployment and service.

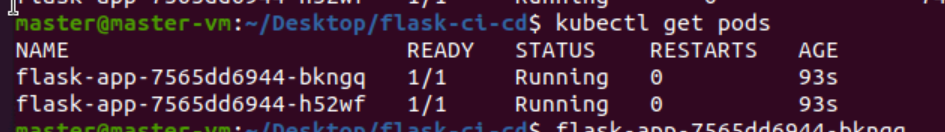


**b. Apply the Deployment**

Apply the deployment configuration to Kubernetes.



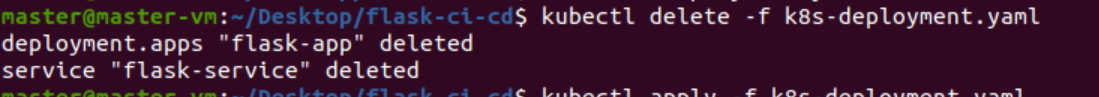
**c. Check Pods Status**



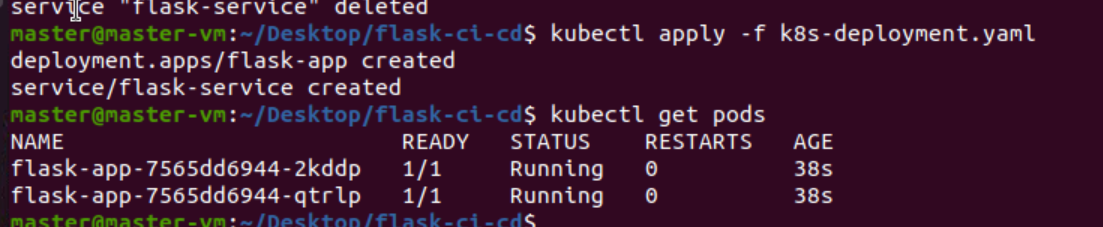
**6. How to Stop and Restart Deployment**

**a. Delete Deployment**

To stop the deployment, delete the Kubernetes resources.

**b. Restart Deployment**

To restart the deployment, apply the deployment YAML again.



**7. Jenkins Pipeline**

**a. Create Jenkinsfile**

Create a Jenkins pipeline to automate the build, push, and deployment process.

groovy

Copy

pipeline {

agent any

stages {

stage('Clone Code') {

steps {

git 'https://github.com/kpkm25/flask-ci-cd.git'

}

}

stage('Build Docker Image') {

steps {

sh 'docker build -t kpkm25/flask-ci-cd:latest .'

}

}

stage('Push to Docker Hub') {

steps {

withDockerRegistry([credentialsId: 'docker-creds', url: '']) {

sh 'docker push kpkm25/flask-ci-cd:latest'

}

}

}

stage('Deploy to Kubernetes') {

steps {

sh 'kubectl apply -f k8s-deployment.yaml'

}

}

}

}

*Add screenshot here*

**Conclusion**

This guide has provided step-by-step instructions to set up Kubernetes on Ubuntu, build a Flask app, deploy it on Kubernetes, and create a Jenkins pipeline for continuous integration and deployment. Use this as a foundation for your Kubernetes and CI/CD projects.