**CAT - 2**

**CLOUD COMPUTING LAB**

**TEAM MEMBERS:**

V L HARISUDHAN (1934010)

K P NITHISH KUMAAR (1934026)

S BARATH (1934004)

**AIM:** To dockerize a web application and deploy it in the Kubernetes cluster.

**APPLICATION:** Breast cancer prediction using machine learning.

**PROBLEM STATEMENT:** Prediction of breast cancer using certain features like radius, texture, area, smoothness, concavity, compactness, symmetry, and fractal dimension.

**TOOLS OR PLUG-INS USED**

* Jupyter Notebook
* Visual Studio Code
* Docker desktop
* AZURE
* Docker Playground
* Github

**USES OF PLUG-INS**

Jupyter notebook: Used for training and building the machine learning model for breast cancer prediction.

Github: Had the entire web application with all requirements.

Visual Studio Code: Used for creating Dockerfile.

Docker desktop: Used to dockerize the web application i.e, creating images, running containers with creating required packages and dependencies.

Docker playground: Uploaded the dockerized image, in order to fetch it from AZURE for deployment.

AZURE: Used for creating a Kubernetes cluster node for deploying the dockerized web application which has been pushed in docker playground and retrieved using a YAML file.

**PROPOSED SOLUTION**

1. Built a machine learning model for the given problem statement.
2. Developed a web application.
3. Pushed into Git repository.
4. Created a Dockerfile and built it.
5. Pushed the Docker image into the docker hub.
6. From the docker hub, the image is retrieved to deploy it in Kubernetes in AZURE.

**DOCKER**

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

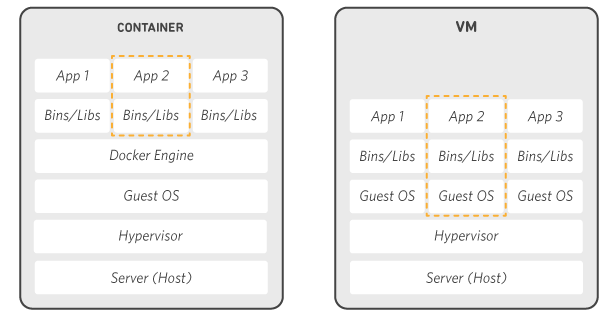
## **HOW DOCKER WORKS**

## 

## Docker works by providing a standard way to run your code. Docker is an operating system for containers. Similar to how a virtual machine virtualizes (removes the need to directly manage) server hardware, containers virtualize the operating system of a server. Docker is installed on each server and provides simple commands you can use to build, start, or stop containers.

## 

## AWS services such as AWS Fargate, Amazon ECS, Amazon EKS, and AWS Batch make it easy to run and manage Docker containers at scale.



**DOCKER IMAGES:**

A Docker image is a file used to execute code in a Docker container. Docker images act as a set of instructions to build a Docker container, like a template. Docker images also act as the starting point when using Docker. An image is comparable to a snapshot in virtual machine (VM) environments.

**CONTAINER:**

A Docker container is a virtualized runtime environment used in application development. It is used to create, run and deploy applications that are isolated from the underlying hardware. A Docker container can use one machine, share its kernel and virtualize the OS to run more isolated processes. As a result, Docker containers are lightweight.

**OUR DOCKERFILE:**

FROM ubuntu

RUN apt-get update

RUN apt-get install -y python3 python3-pip

COPY requirements.txt requirements.txt

RUN pip3 install -r requirements.txt

RUN mkdir app

WORKDIR /opt/app

COPY . /opt/app

ENTRYPOINT FLASK\_APP=/opt/app/app.py flask run --host 0.0.0.0

**DOCKER COMMANDS:**

Docker build - Build an image from a Dockerfile

*sudo docker build -t tiger1 .*

Docker run - to run a command in a new container

*sudo docker run -t tiger1*

Docker log - Fetch log for our image (tiger1)

*sudo docker logs tiger1*

Docker images - to display all images

*sudo docker images*

Docker ps & Docker ps -a - Check running and available image.

*sudo docker ps*

*sudo docker ps -a*

Docker image - To display all available images

*sudo docker images*

**DOCKER HUB:**

1. Login into docker hub.
2. Push the dockerized image into the Docker hub using the following command,

*sudo docket push -it sarika088/tiger1*

1. Pull the dockerized image using the following command,

*sudo docker pull sarika/tiger1*

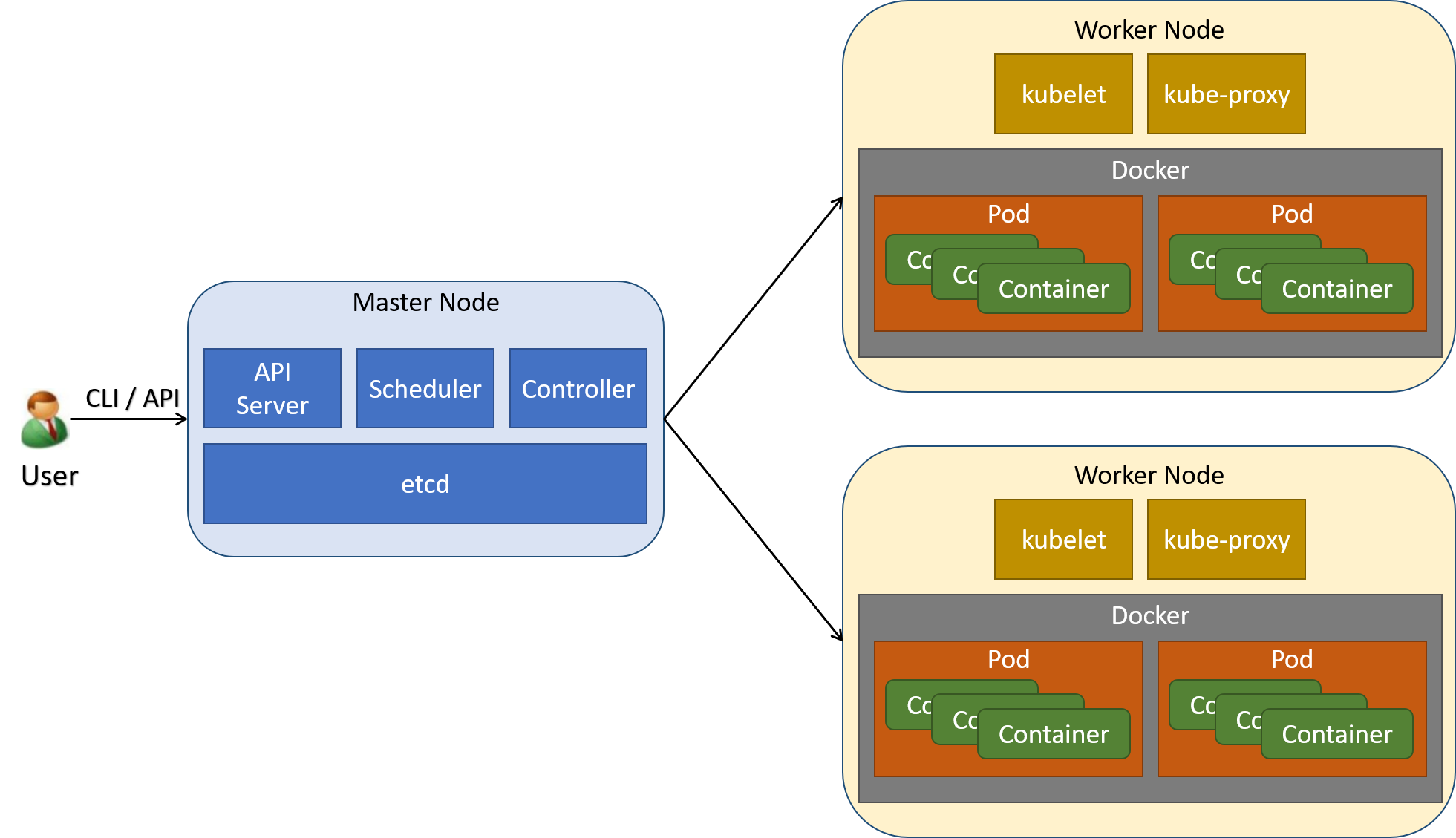
1. Run the pulled image using the following command,

*Sudo docker run -p 4000:5000 sarika088/tiger1*

**KUBERNETES:**

Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available.

**KUBERNETES ARCHITECTURE:**



**KUBERNETES DEPLOYMENT:**

Created a new kubernetes cluster node and deployed it in AZURE. Opened the cloud shell and created a YAML file using the command,

*code service.yaml*

YAML FILE

apiVersion: apps/v1

kind: Deployment

metadata:

name: tiger1-app

spec:

replicas: 2

selector:

matchLabels:

app: tiger1-app

template:

metadata:

labels:

app: tiger1-app

spec:

nodeSelector:

"kubernetes.io/os": linux

containers:

- name: tiger1-app

image: sarika088/tiger1

env:

- name: ALLOW\_EMPTY\_PASSWORD

value: "yes"

resources:

requests:

cpu: 100m

memory: 128Mi

limits:

cpu: 250m

memory: 256Mi

ports:

- containerPort: 5000

name: tiger1

---

apiVersion: v1

kind: Service

metadata:

name: tiger1-app

spec:

type: LoadBalancer

ports:

- port: 5000

targetPort: 5000

nodePort: 40000

selector:

app: tiger1-app

**DEPLOYMENT OF KUBERNETES IN AZURE:**

1. To run the config file for deployment

*kubectl apply -f service.yaml*

1. To get the service

*kubectl get service*

