

# PREPARATION PHASE

## Deployment of App in IBM Cloud

### Containerize The App

Date	27 August 2022
Team ID	PNT2022TMID26477
Project Name	Personal Expense Tracker Application

## 1. DOCKER IMAGE CREATION:

### STEP 1: To Create a Kubernetes Cluster

The screenshot shows the IBM Cloud mycluster dashboard for a cluster named 'mycluster'. The cluster is in a 'Normal' state and is scheduled to expire in 30 days. The dashboard provides an overview of the cluster's status, including node status (1 of 1 Normal), add-on status (0 of 0 Normal), master status (Normal), and ingress status (Healthy). It also displays details such as the cluster ID (cdqv317f0otoo@svhtdg), version (1.24.8\_1544), infrastructure (Classic), and zones (Milan 01). A warning banner indicates that the cluster will be deleted in 30 days and advises backing up data. A button labeled 'Enable' is visible for image security enforcement.

### STEP 2: Containerize the Flask Application

The screenshot shows a Visual Studio Code editor with a Dockerfile open. The Dockerfile is for a Python 2.7 application and includes instructions for setting the maintainer, updating the apt-get, creating the /app directory, setting the working directory, copying the application files, installing requirements, exposing port 5000, and running the application. The terminal output shows the Docker build process, including the installation of the Docker extension and the successful build of the image.

```
FROM python:2.7
LABEL maintainer="Buvaneswari M"
RUN apt-get update
RUN mkdir /app
WORKDIR /app
COPY . /app
RUN pip install -r requirements.txt
EXPOSE 5000
ENTRYPOINT [ "python" ]
CMD [ "app.py" ]
```

Terminal Output:

```
2022-11-17 15:14:08.021 [info] update#setState idle
2022-11-17 15:14:14.568 [info] Starting extension host with pid 5324 (fork() took 76 ms).
2022-11-17 15:14:38.032 [info] update#setState checking for updates
2022-11-17 15:14:38.207 [info] update#setState idle
```

(The Process will continue in **Upload Image to IBM Container Registry** and **Deploy in Kubernetes Cluster**)

## 2. CREATING DOCKER IMAGE FOR FLASK APP

**STEP 1:** Make a Project folder

**STEP 2:** Insert the following code into the Dockerfile created earlier

**STEP 3:** Copy the following into “requirements.txt” file

**STEP 4:** Test the flask app

**STEP 5:** Close the server by pressing CTRL + C

**STEP 6:** Build the Docker image

**STEP 7:** Run the docker image

**STEP 8:** Test Again

### CODE:

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello():
    return "welcome to the flask tutorials"

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5001, debug=True)
```

```
FROM python:alpine3.7
COPY . /app
WORKDIR /app
RUN pip install -r requirements.txt
EXPOSE 5001
ENTRYPOINT [ "python" ]
CMD [ "demo.py" ]
```

### OUTPUT:

