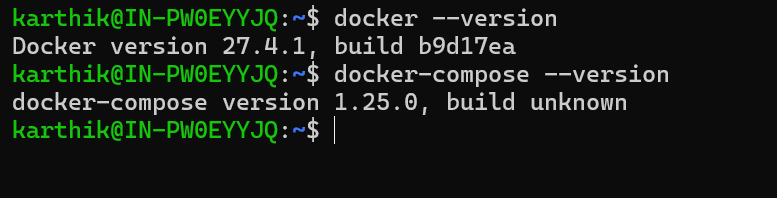
**Docker Basic Assignments**

**Install Docker and Docker Compose on your machine.**



**Verify the installation by running a test container.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screen shot of a computer

AI-generated content may be incorrect.**

**Create a simple web application (e.g., a Python Flask, DotNet, Java or Node application).**

**Cd C:\Users\karthikn\karthik-python-projects\bench-code**

**pip install flask**

**python app.py**

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

**Write a Dockerfile to containerize the application.**

Vi app.py

# app.py

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route("/")

def home():

return "Welcome to My Simple Flask App!"

@app.route("/hello/<name>")

def hello(name):

return f"Hello, {name}!"

if \_\_name\_\_ == "\_\_main\_\_":

# Important: bind to 0.0.0.0 so it works in Docker

app.run(host="0.0.0.0", port=5000, debug=True)

vi requirements.txt

flask==3.0.3

**Build the Docker image and run a container from it.**

docker build -t flask-app .

docker run -p 5000:5000 flask-app

A close up of a screen

AI-generated content may be incorrect.

**Use Docker commands to list, start, stop, and remove containers.**

A screenshot of a computer program

AI-generated content may be incorrect.

**Inspect running containers and view logs.**

A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.

**Write a docker-compose.yml file to define a multi-container application.**

Cd C:\Users\karthikn\karthik-python-projects\bench-code

Vi docker-compose.yml

version: "3.3"

services:

web:

build: .

container\_name: flask-app

ports:

- "5000:5000"

depends\_on:

- redis

environment:

- REDIS\_HOST=redis

redis:

image: "redis:7-alpine"

container\_name: redis-server

ports:

- "6379:6379"

**Use Docker Compose to bring up the application and ensure all services are running correctly.**

docker-compose up –build

Open <http://localhost:5000> → Flask app running.

Redis is accessible internally at redis:6379.

**Extend the multi-container application to include a database service and configure the web application to interact with the database.**

Vi docker-compose.yml

version: "3.3"

services:

web:

build: .

container\_name: flask-app

ports:

- "5000:5000"

depends\_on:

- redis

- db

environment:

- REDIS\_HOST=redis

- DATABASE\_URL=postgresql://flask\_user:flask\_pass@db:5432/flask\_db

redis:

image: "redis:7-alpine"

container\_name: redis-server

ports:

- "6379:6379"

db:

image: postgres:15-alpine

container\_name: postgres-db

restart: always

environment:

POSTGRES\_USER: flask\_user

POSTGRES\_PASSWORD: flask\_pass

POSTGRES\_DB: flask\_db

ports:

- "5432:5432"

volumes:

- pgdata:/var/lib/postgresql/data

volumes:

pgdata:

vi requirements.txt

flask==3.0.3

redis==5.0.1

psycopg2-binary==2.9.9

SQLAlchemy==2.0.31

Dockerfile

# Use a lightweight Python base image

FROM python:3.10-slim

# Set working directory inside the container

WORKDIR /app

# Copy requirements first (for caching layers)

COPY requirements.txt .

# Install dependencies

RUN pip install --no-cache-dir -r requirements.txt

# Copy the rest of the application code

COPY . .

# Expose Flask's default port

EXPOSE 5000

# Run the application

CMD ["python", "app.py"]

app.py

from flask import Flask, request, jsonify

import redis, os

from sqlalchemy import create\_engine, text

app = Flask(\_\_name\_\_)

# Redis config

redis\_host = os.getenv("REDIS\_HOST", "redis")

r = redis.Redis(host=redis\_host, port=6379, decode\_responses=True)

# Database config

db\_url = os.getenv("DATABASE\_URL")

engine = create\_engine(db\_url, echo=True, future=True)

@app.route("/")

def home():

return "Welcome to My Extended Flask App with Redis + Postgres!"

@app.route("/visits")

def visits():

count = r.incr("counter")

return f"Number of visits: {count}"

@app.route("/users", methods=["POST"])

def add\_user():

data = request.get\_json()

name = data.get("name")

if not name:

return jsonify({"error": "Name is required"}), 400

with engine.begin() as conn:

conn.execute(text("CREATE TABLE IF NOT EXISTS users (id SERIAL PRIMARY KEY, name TEXT)"))

conn.execute(text("INSERT INTO users (name) VALUES (:name)"), {"name": name})

return jsonify({"message": f"User {name} added!"})

@app.route("/users", methods=["GET"])

def list\_users():

with engine.connect() as conn:

result = conn.execute(text("SELECT id, name FROM users"))

users = [{"id": row.id, "name": row.name} for row in result]

return jsonify(users)

if \_\_name\_\_ == "\_\_main\_\_":

app.run(host="0.0.0.0", port=5000, debug=True)

Build

docker-compose up –build

A screenshot of a computer

AI-generated content may be incorrect.

**Test**

<http://localhost:5000/>

<http://localhost:5000/visits>

Add new user in database

curl -X POST http://localhost:5000/users \

-H "Content-Type: application/json" \

-d '{"name": "Alice"}'

A computer screen with white text

AI-generated content may be incorrect.

List down all users in Database

<http://localhost:5000/users>

A screenshot of a computer

AI-generated content may be incorrect.