1. Containers – Running Applications

Q1. Run an Nginx Web Server

- Pull the nginx image.
- Run it in a container.
- Map container port 80 to host port 8080.
- Open the browser and check if Nginx default page is visible.

Q2. Run a Python App inside a Container

- Use the official python:3.10-slim image.
- Run a container that executes a one-line Python program printing "Hello Docker".
- Observe what happens after execution.

Q3. Run a MySQL Database

- Start a mysql:8 container with:
    - Root password set via environment variable.
    - A new database created at startup.
- Access the container shell.
- Connect to MySQL inside and verify the database exists.

2. Custom Images – Build Your Own

Q4. Build a Flask Application Image

- Write a small Flask app that returns "Hello from Flask in Docker".
- Create a requirements.txt file and Dockerfile.
- Build your own image.
- Run the container and check the response in a browser.

Q5. Create a Custom Nginx Image

- Create a simple index.html file with your name.
- Use nginx:alpine as base image.
- Copy your HTML file into the container.
- Build and run the custom image.

- Open in browser to confirm your page loads.

3. Volumes – Persisting Data

Q6. WordPress with Persistent MySQL

- Run a mysql:8 container with a **named volume** for /var/lib/mysql.
- Run a wordpress container connected to the database.
- Stop and remove the containers.
- Restart them and confirm WordPress still remembers your data.

Q7. Flask with Bind Mount

- Create or modify your Flask app locally.
- Run the app container with a **bind mount** mapping your project folder.
- Edit code on the host machine.
- Check if changes reflect without rebuilding the image.

4. Networks – Multi-Container Applications

Q8. Flask + Redis Counter App

- Write a Flask app that connects to Redis to store a counter.
- Create a user-defined Docker network.
- Run a Redis container in the network.
- Run your Flask container in the same network.
- Refresh the app page multiple times and confirm the counter increments.

Q9. MySQL + WordPress Blogging App

- Create a custom Docker network.
- Run a MySQL container with environment variables for root password and database.
- Run a WordPress container connected to the same network.
- Complete WordPress setup in the browser.

Docker Compose

Q10. Convert Flask + Redis App into Docker Compose

- Write a docker-compose.yml file to define Flask and Redis services.
- Bring up the stack using docker-compose up.
- Verify application works as expected.

Q11. Node.js + MongoDB Full Stack Setup

- Write a simple Node.js app that saves data to MongoDB.
- Create a custom image for the Node.js app.
- Store application logs in a **volume**.
- Connect Node.js and MongoDB containers via a **custom network**.
- Test end-to-end data flow.