**School of Computer Science**

**Experiment 2 of “Cloud Computing”**

**Docker Desktop Installation and Development**

1. **Introduction**

Docker Desktop is Docker Desktop for Windows, designed to run on Windows 10. It is a native Windows application that provides an easy-to-use development environment for building, delivering, and running dockerized applications. Docker Desktop for Windows uses Windows native Hyper-V virtualization and networking and is the fastest and most reliable way to develop Docker applications on Windows. Docker Desktop for Windows supports running Linux and Windows Docker containers.

1. **Install**

**2.1 Installation address**

Docker Desktop installation address:

https://desktop.docker.com/win/stable/amd64/Docker%20Desktop%20Installer.exe

**2.2 Installation success interface**

After the installation is successful, you can see Docker on the Windows desktop Desktop icon, as shown in Figure 2.2.1 .



Figure 2.2.1

1. **Use and inspection**

**3.1 Interface Usage**

Click on Docker Desktop Desktop icon, you can see the following interface, as shown in Figure 3.1.1 .



Figure 3.1.1

**3.2 Command line usage**

Open the Windows command line and enter the command: docker

You can see the following interface, proving that Docker Desktop has been installed successfully, as shown in Figure 3.1.2 .



Figure 3.2.1

1. **Build an image using Dockerfile**

Step 1: Create a dockerfile folder in the /root directory



Step 2: Create a Dockerfile named Dockerfile1



Step 3: Use the vi editor to edit dockerfile1 and enter the following content



Step 4: Build the image, name it httpd:v11



Step 5: View the created image



Step 6: Run a container with the image httpd:v11



Step 7: Verify container contents



Step 8: To facilitate subsequent experiments, delete the container in this section.

docker kill

docker rm

1. **Docker Compose Practice for Front-end and Back-end Projects**

Develop a simple web application based on microservice architecture. This application consists of three main components:

* Web server: provides a simple front-end interface for displaying data obtained from the API
* API service: A backend service that returns data in JSON format.
* Database : A service for storing data. This assignment will use the MySQL database.

The goal is to configure and run these three services using the docker-compose.yml file. Each service should run in its own container and be able to communicate with each other.

**Detailed requirements**

1. Web Server:
* Use any front-end technology stack, for example, you can choose Nginx as a static file server.
* It should be accessible through a browser and mapped to the local port 8080.
1. API Services:
* Build using Node.js or SprinqBoot
* Rely on database services to obtain data
* The service uses port 3000 internally, but there is no need to map this port on the host.