An-Najah National University
Computer Engineering Department

Mayar Abdulkareem
Nora Al-Sadder

Docker Compose File Description:

docker-compose.yml

docker-compose.yml 〉 No Selection

```yaml
version: '3'
services:
  catalog:
    image: bazar
    command: sh -c "node app.js && sleep infinity"
    volumes:
      - ./catalog:/home
    ports:
      - "4000:3000"
    networks:
      - net
    container_name: catalog

  order:
    image: bazar
    command: sh -c "node app.js && sleep infinity"
    volumes:
      - ./order:/home
    ports:
      - "4001:3000"
    networks:
      - net
    container_name: order

  front:
    image: bazar
    command: sh -c "node app.js && sleep infinity"
    volumes:
      - ./front:/home
    ports:
      - "4002:3000"
    networks:
      - net
    container_name: front

networks:
  net:
```

The provided Docker Compose file is used to define and manage a multi-container Docker application. The application consists of three services: catalog, order, and front. Each service is based on the same Docker image named "bazar" and runs a Node.js application.

Services:

Catalog Service:

- **Image:** bazar
- **Command:** Executes the Node.js application using the command "sh -c 'node app.js && sleep infinity.'"
- **Volumes:** Mounts the local directory "./catalog" to the "/home" directory within the container.
- **Ports:** Maps port 4000 on the host to port 3000 on the container.
- **Network:** Connects to the "net" network.
- **Container Name:** catalog

Order Service:

- **Image:** bazar
- **Command:** Executes the Node.js application using the command "sh -c 'node app.js && sleep infinity.'"
- **Volumes:** Mounts the local directory "./order" to the "/home" directory within the container.
- **Ports:** Maps port 4001 on the host to port 3000 on the container.
- **Network:** Connects to the "net" network.
- **Container Name:** order

Front Service:

- **Image:** bazar
- **Command:** Executes the Node.js application using the command "sh -c 'node app.js && sleep infinity.'"
- **Volumes:** Mounts the local directory "./front" to the "/home" directory within the container.
- **Ports:** Maps port 4002 on the host to port 3000 on the container.
- **Network:** Connects to the "net" network.
- **Container Name:** front

Network:

- A network named "net" is defined, but its configuration is incomplete in the provided snippet.

Summary:

This Docker Compose file sets up three containers, each running a Node.js application based on the "bazar" image. The containers are isolated within the "net" network, and

their respective codebases are mounted from local directories. The defined ports allow external access to the services.

To run this project:

First, clone the repository from GitHub SSH: git@github.com:norasadder/Bazar.git
HTTP: https://github.com/norasadder/Bazar.git

After that, create the image using docker build -t bazar .

Then, use docker compose up -d to run the docker containers or docker compose down to stop the containers.

Note: You can find the URLs explained in the output document.

Improvement:

Whenever we want to update a file, we need to read the whole file, and then rewrite it. Using a database as an extension, we can only change the part of data we want without reading the whole file. In this way, the performance can be enhanced.

Part 2:

The updated Docker Compose file introduces significant enhancements to the deployment configuration, focusing on high availability and load distribution. By setting the `replicas` to 2 for both the `catalog` and `order` services, it ensures that there are always two instances of each service running, thereby providing redundancy and enabling load balancing. The `endpoint_mode: dnsrr` (DNS Round Robin) implies that DNS will alternate network traffic among the available service instances, distributing the load evenly. This is crucial for maintaining performance during high traffic periods. Additionally, the shared volumes for `catalog-data` and `order-data` across the replicas mean that stateful data is consistent and can be accessed by either instance, allowing for a seamless and coherent user experience even as requests are handled by different service instances.

Note: in part 2, the port for the front is changed to 4004 and you can't access the API of catalog and order from your browser only the front. Use this as your base URL

http://localhost:4004

docker-compose.yml

```yaml
version: '3.8'

services:

  catalog:
    image: bazar
    command: sh -c "node app.js && sleep infinity"
    volumes:
      - ./catalog:/home
      - catalog-data:/home/catalog
    networks:
      - net
    deploy:
      replicas: 2
      endpoint_mode: dnsrr
      placement:
        constraints:
          - node.labels.type == worker

  order:
    image: bazar
    command: sh -c "node app.js && sleep infinity"
    volumes:
      - ./order:/home
      - order-data:/home/order
    networks:
      - net
    deploy:
      replicas: 2
      endpoint_mode: dnsrr
      placement:
        constraints:
          - node.labels.type == worker

  front:
    image: bazar
    command: sh -c "node app.js && sleep infinity"
    volumes:
      - ./front:/home
    networks:
      - net
    deploy:
      replicas: 1
      endpoint_mode: dnsrr
      placement:
        constraints:
          - node.labels.type == worker
    ports:
      - "4004:3000"

networks:
  net:

volumes:
  catalog-data:
  order-data:
```