Part 2 Solution

Step 1: Stop and Remove Any Running Containers

Before setting up the new application, ensure no conflicting containers are running from previous setups. Use the following commands to stop and remove containers:

bash
Copy code

```
# List all running containers
docker ps

# Stop all running containers
docker stop $(docker ps -q)

# Remove all containers
docker rm $(docker ps -a -q)
```

Step 2: Create a `docker-compose.yaml` File

The `docker-compose.yaml` file will define your application's services, networks, and volumes.

Here is an example of the `docker-compose.yaml` file:

yaml
Copy code

```
version: "3.8"
```

```
services:
  csvserver:
    image: <your_image_name> # Replace with the actual
image name used in Part I
    ports:
      - "8080:8080" # Map container port 8080 to host
port 8080
    env_file:
      - csvserver.env # Pass environment variables from
an external file
    volumes:
      - ./data:/data # Mount a local directory to the
container

volumes:
  data:
```

Step 3: Create the `csvserver.env` File

This file will hold the environment variables required for the application. For example, if Part I required an environment variable `INPUT_FILE`, create a file named `csvserver.env`:

bash
Copy code

```
# csvserver.env
INPUT_FILE=/data/inputfile.csv
```

Place this file in the same directory as the `docker-compose.yaml`.

Step 4: Prepare the Data Directory and Input File

The application may require a specific file (e.g., `inputfile.csv`) to function. Create a `data` directory and place your input file there:

bash
Copy code

```
mkdir data
echo "1,2,3" > data/inputfile.csv
```

Step 5: Start the Application Using `docker-compose`

Run the following command to start the application:

bash
Copy code

```
docker-compose up
```

This command will:

1. *Read the `docker-compose.yaml` file.*
2. *Create and start the `csvserver` service.*
3. *Pass the environment variables from `csvserver.env`.*
4. *Map the host's `data` directory to the container's `/data` directory.*

Step 6: Verify the Application

Once the application is up, check the logs to ensure it started successfully:

bash
Copy code
```
docker-compose logs
```

You can also verify that the service is running by accessing it via the browser or using `curl`:

bash
Copy code
```
curl http://localhost:8080
```