**Docker Notes by Varad Dhumale**

**Containers and Images**

* **Docker Containers**: Provide isolated environments to run applications with specific configurations.
* **Docker Images**: Essentially the operating system and application dependencies for the container.

**Basic Commands**

* **Create and run a container**: docker run -it ubuntu (sets up an Ubuntu container).
* **List running containers**: docker container ls.
* **List all containers**: docker container ls -a.
* **Start a container by name**: docker start flamboyant\_babbage.
* **Stop a container by name**: docker stop flamboyant\_babbage.
* **Execute a command in a container**: docker exec flamboyant\_babbage ls.
* **Attach a terminal to a container**: docker exec -it flamboyant\_babbage bash.

**Port Mapping**

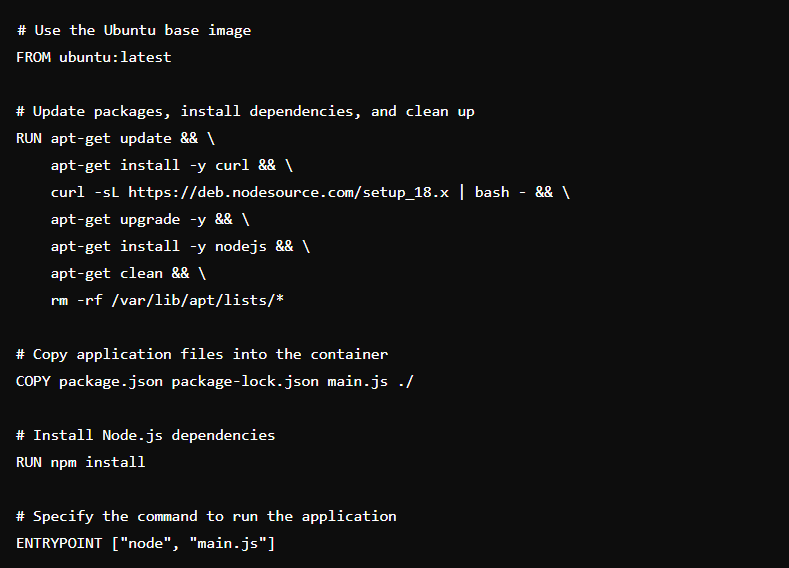
* Containers run services on specific ports which are isolated from the host machine.
* **Expose a container's port to the host**: docker run -it -p 3000:1025 mailhog/mailhog.
  + This maps port 1025 in the container to port 3000 on the host machine.

**Environment Variables**

* Used to pass configuration information to containers.
* **Set environment variables**: docker run -it -p 3000:1025 -e key=value -e key=value mailhog/mailhog.
* **Example**: docker run -it -e PORT=4000 -p 4000:4000 myfirstdocker.

**Dockerizing a Node.js Application**

1. **Create a Node.js file**.
2. **Create a Dockerfile**:



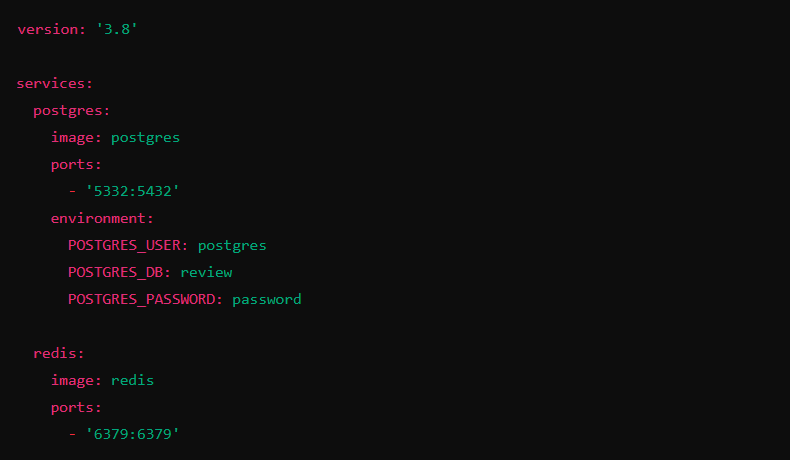
1. **Build the Docker image**:
   * docker build -t myfirstDocker .
   * Note: Docker caches layers to optimize subsequent builds.

**Publishing to Docker Hub**

1. **Create a repository on Docker Hub**.
2. **Build the Docker image with the same name as the repository**.
3. **Push the image to Docker Hub**:
   * docker push varad177/myfirstdocker.

**Docker Compose**

* **Purpose**: Manage multiple containers for a single application.
* **Create a docker-compose.yml file**:

****

* **Commands**:
* **Start services**: docker compose up.
* **Stop services**: docker compose down.
* **Detached mode (run in the background)**: docker compose up -d.

-----------------------------------------------------------------------------------------------------------------

# Docker Commands and Descriptions

### Basic Commands

* **Create and run a container:** docker run -it ubuntu  
  Sets up and starts an Ubuntu container with interactive terminal.
* **List running containers:** docker container ls  
  Displays all currently running containers.
* **List all containers:** docker container ls -a  
  Shows all containers, including those that are stopped.
* **Start a container by name:** docker start flamboyant\_babbage  
  Starts a stopped container named flamboyant\_babbage.
* **Stop a container by name:** docker stop flamboyant\_babbage  
  Stops the running container named flamboyant\_babbage.
* **Execute a command in a container:** docker exec flamboyant\_babbage ls  
  Runs the ls command inside the container flamboyant\_babbage.
* **Attach a terminal to a container:** docker exec -it flamboyant\_babbage bash  
  Opens an interactive terminal session inside the container flamboyant\_babbage.

### Port Mapping

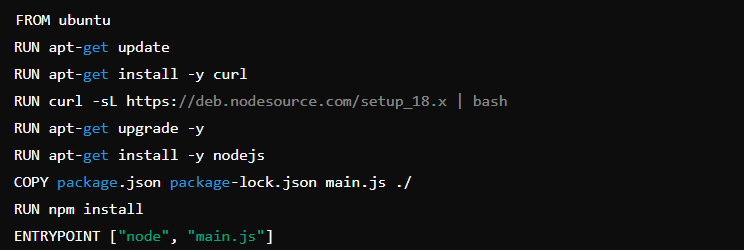
* **Expose a container's port to the host:** docker run -it -p 3000:1025 mailhog/mailhog  
  Maps port 1025 in the container to port 3000 on the host machine.

### Environment Variables

* **Set environment variables:** docker run -it -e key=value -e key=value mailhog/mailhog  
  Example: docker run -it -e PORT=4000 -p 4000:4000 myfirstdocker

### Dockerizing a Node.js Application

1. **Create a Dockerfile:**



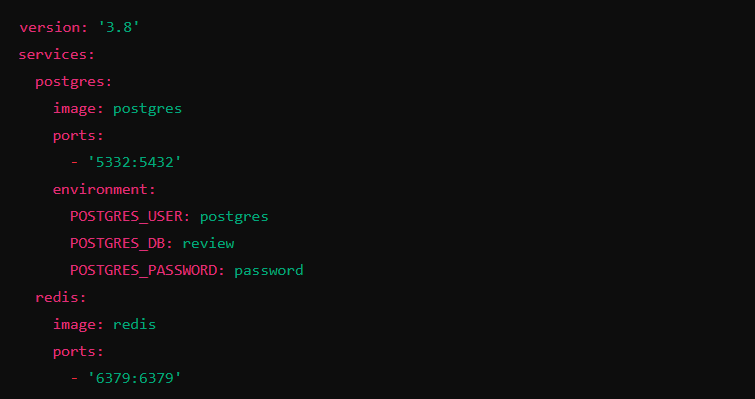
1. **Build the Docker image:** docker build -t myfirstDocker .

**Publishing to Docker Hub**

1. **Create a repository on Docker Hub.**
2. **Build the Docker image with the same name as the repository.**
3. **Push the image to Docker Hub:** docker push varad177/myfirstdocker

**Docker Compose**

* **Create a docker-compose.yml file:**



 Start **services:** docker compose up

 Stop **services:** docker compose down

 Detached **mode (run in the background):** docker compose up -d