

Run n8n Locally with Docker

Run n8n Locally with Docker

Requirements

- **Docker Desktop**
 - Windows / macOS / Linux
 - Basic command-line usage
-

Step 1: Install Docker

Download and install Docker Desktop

👉 <https://www.docker.com/products/docker-desktop/>

After installation, confirm:

```
docker --version
```

Step 2: Create a Folder for n8n

```
mkdir n8n-local  
cd n8n-local
```

Step 3: Create docker-compose.yml

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

```
version: "3.8"  
  
services:  
  n8n:  
    image: n8nio/n8n  
    container_name: n8n  
    ports:  
      - "5678:5678"  
    environment:
```

```
- N8N_BASIC_AUTH_ACTIVE=true
- N8N_BASIC_AUTH_USER=admin
- N8N_BASIC_AUTH_PASSWORD=admin123
- N8N_HOST=localhost
- N8N_PORT=5678
- N8N_PROTOCOL=http
- WEBHOOK_URL=http://localhost:5678/
- GENERIC_TIMEZONE=Asia/Kuala_Lumpur
volumes:
- n8n_data:/home/node/.n8n
restart: always

volumes:
n8n_data:
```

Step 4: Start n8n

```
docker compose up -d
```

Check status:

```
docker ps
```

Step 5: Open n8n

Open browser:

```
http://localhost:5678
```

Login:

- **Username:** admin
- **Password:** admin123

 n8n is now running locally

Run n8n Locally with Node

Run n8n Locally with Node.js (No Docker)

 More issues possible (Node version, dependencies)

Requirements

- Node.js **v18 or v20**
- npm

Check:

```
node -v  
npm -v
```

Step 1: Install n8n

```
npm install -g n8n
```

Step 2: Start n8n

```
n8n
```

Step 3: Open n8n

```
http://localhost:5678
```

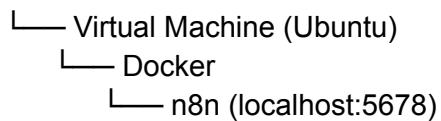
Ubuntu VM + Docker Setup

Ubuntu VM + Docker Setup



Architecture (What You're Building)

Your PC



Step 1: Prepare Ubuntu VM

✓ Recommended Ubuntu Version

- **Ubuntu 22.04 LTS** (recommended)
- Minimum:
 - 2 GB RAM (4 GB better)
 - 2 CPU
 - 30 GB disk

✓ Update system

```
sudo apt update && sudo apt upgrade -y
```



Step 2: Install Docker on Ubuntu

1 Install dependencies

```
sudo apt install -y ca-certificates curl gnupg lsb-release
```

2 Add Docker GPG key

```
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg
--dearmor -o /etc/apt/keyrings/docker.gpg
```

3 Add Docker repository

```
echo \
"deb [arch=$(dpkg --print-architecture)
signed-by=/etc/apt/keyrings/docker.gpg] \
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
```

4 Install Docker Engine

```
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io
docker-compose-plugin
```

5 Enable Docker without sudo (IMPORTANT)

```
sudo usermod -aG docker $USER
newgrp docker
```

Verify:

```
docker run hello-world
```



Step 3: Create n8n Project Folder

```
mkdir ~/n8n
```

```
cd ~/n8n
```



Step 4: Create docker-compose.yml

```
nano docker-compose.yml
```

Paste this production-safe local setup ⤵

```
version: "3.8"

services:
  n8n:
    image: n8nio/n8n:latest
    container_name: n8n
    ports:
      - "5678:5678"
    environment:
      - N8N_BASIC_AUTH_ACTIVE=true
      - N8N_BASIC_AUTH_USER=admin
      - N8N_BASIC_AUTH_PASSWORD=strongpassword
      - N8N_HOST=localhost
      - N8N_PORT=5678
      - N8N_PROTOCOL=http
      - WEBHOOK_URL=http://localhost:5678/
      - GENERIC_TIMEZONE=Asia/Kuala_Lumpur
    volumes:
      - n8n_data:/home/node/.n8n
    restart: unless-stopped

volumes:
  n8n_data:
```

Save:

CTRL + O → Enter

CTRL + X

► Step 5: Start n8n

```
docker compose up -d
```

Check:

```
docker ps
```

Step 6: Access n8n (IMPORTANT for VM)

Option A: Access from inside VM

Open browser in Ubuntu:

<http://localhost:5678>

Option B: Access from Host Machine (Recommended)

✓ VM Network Setting

Set your VM to:

- **NAT** (default) → use **port forwarding**
- OR **Bridged Adapter** (easier)

✓ If Bridged:

Find VM IP:

```
ip a
```

Example:

```
inet http://192.168.1.120
```

Open from host:

```
http://192.168.1.120:5678
```

Step 7: Login

Use credentials you set:

- Username: `admin`
- Password: `strongpassword`

 n8n is running on Docker inside Ubuntu VM

Pro Tips (Very Important)

◆ Check logs

```
docker logs -f n8n
```

◆ Stop / Start

```
docker compose stop  
docker compose start
```

◆ Restart safely

```
docker compose down  
docker compose up -d
```

(Data is safe due to volume)