

The current version is #ident
 "@(#)\$Format:LocalFoodAI_lanfr144:generate_docs.py:Francois
 Lange:lanfr144@school.lu:2026/06/16 21:48:22:Francois
 Lange:lanfr144@school.lu:2026/06/16
 21:48:22:2a8ed056889f3b796f9266feda591b12b72f3b96:HEAD -> main, origin/main:\$"

Local Food AI - Detailed Installation and Deployment Guide

This guide describes how to provision the host hypervisor, install Docker on Ubuntu, clone the repository, check out the correct branch, and launch the application.

1. WSL2 Ubuntu Instance Setup

To create a dedicated WSL2 environment for the application, execute the following command in an Administrator PowerShell window:

```
wsl --install -d Ubuntu-22.04 --name Dopro1
```

During initialization, configure the default Unix user and password as prompted:

```
Create a default Unix user account: lanfr144
New password:
Retype new password:
passwd: password updated successfully
```

[!WARNING] WSL Filesystem Mounts: By default, launching WSL may place you in a Windows filesystem mount (e.g. /mnt/d/...). To prevent performance degradation and permission bugs, navigate to your WSL home directory immediately:

```
cd ~
```

2. Docker & Docker Compose Installation inside WSL Ubuntu

To install Docker directly inside your WSL Ubuntu instance (without Docker Desktop):

Step 2.1: Clean Existing Docker Versions

```
sudo apt remove -y docker.io docker-compose docker-compose-v2 docker-doc podman-
docker containerd runc
```

Step 2.2: Add Docker's Official GPG Key & Repository and Install Docker

```
sudo apt update
sudo apt install -y ca-certificates curl
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/
keyrings/docker.asc
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/
docker.asc] https://download.docker.com/linux/ubuntu jammy stable" | sudo tee /
etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-
plugin docker-compose-plugin
```

Step 2.3: Start and Enable Docker Daemon

```
sudo systemctl start docker
sudo systemctl enable docker
```

Step 2.4: Add User to the Docker Group

Ensure you can execute Docker commands without **sudo**:

```
grep "^docker:" /etc/group || sudo addgroup docker
sudo usermod -aG docker $USER
```

Step 2.5: Reboot the WSL Instance

To reboot the WSL instance, you must shutdown and restart WSL. You can choose one of the following methods:

Option A: Restart from Windows Host (Recommended & Safest)

- Close your Ubuntu terminal.
- Open Windows PowerShell or Command Prompt.
- Run the shutdown command:

```
wsl --shutdown
```

- Re-open your Ubuntu terminal.

Option B: Restart from inside WSL Terminal If you prefer to trigger the reboot directly from the WSL terminal:

```
bash
cd /mnt/c/ && cmd.exe /c start "rebooting WSL" cmd /c "timeout 5 && wsl -d
Ubuntu-22.04" && wsl.exe --terminate Ubuntu-22.04
```

Upon reconnecting, verify Docker is running by starting the hello-world container:

```
bash
docker run hello-world
```

3. Network Configuration & Performance Tuning

Step 3.1: Switch to Legacy IPTables

Ubuntu 22.04 uses `nftables` by default. Switch to legacy iptables to ensure Docker network NAT rules match correctly:

```
bash
sudo update-alternatives --config iptables
# Select option 1 (iptables-legacy)
```

Step 3.2: Configure DNS Settings

To ensure reliable package downloads and LLM registry calls:

```
bash
echo '1,$ s/^/#/'
$ a
nameserver 1.1.1.1
.
w
q' | sudo ed /etc/resolv.conf

echo '$ a
# Added these 2 lines:
[network]
generateResolvConf = false
.
w
q' | sudo ed /etc/wsl.conf
```

4. Repository Clones & Branch Governance

There are two repositories configured for this project:

- Primary Git Repository: https://git.btshub.lu/lanfr/LocalFoodAI_lanfr144.git
- Alternative Git Repository (Worldwide Access - Clone): https://github.com/lanfr144/LocalFoodAI_lanfr144.git

Clone the primary repository inside your home directory:

```
bash
cd ~
git clone https://git.btshub.lu/lanfr/LocalFoodAI_lanfr144.git
cd LocalFoodAI_lanfr144
```

Step 4.1: List Available Branches

Inspect both local and remote branches on the server:

```
bash
git branch -a
```

(Shows available branches like *remotes/origin/main* or *remotes/origin/dev*)

Step 4.2: Track and Check Out the Right Branch

Select the main production branch and extract it:

```
bash
git checkout main
```

(If the repository uses a master branch, replace 'main' with 'master')

Step 4.3: Set Default Branch (Optional)

To set the default tracking branch for your local copy:

```
bash
git remote set-head origin main
```

5. Launching the App

Ensure the runbooks and sync scripts have executable permissions:

```
bash
chmod +x data_sync.sh backup_db.sh manage_services.sh scripts/manage_models.sh
```

Follow the standard runbook to initialize credentials and launch services:

```
bash
# 1. Create a local [.env file](../.env) based on the [Environment Variables
Configuration Runbook](Env_Configuration.pdf) guidelines
# 2. Run the service manager to spin up containers
./manage_services.sh start
```