

The current version is #ident
"@(#) \$Format:LocalFoodAI_lanfr144:Operator_Installation_Guide.md:Francois
Lange:lanfr144@school.lu:2026/06/16 09:25:09:Francois
Lange:lanfr144@school.lu:2026/06/16
09:25:09:1e316d1030d64b01b8cbc5cdb3a3a272a963caff:: \$"

Local Food AI - Detailed Operator Installation Guide

This document is a step-by-step installation, mapping, configuration, and verification manual for deploying the **Local Food AI** system in an enterprise environment. It covers hybrid hypervisor infrastructure (WSL2, Hyper-V, and VirtualBox), cross-node networking, SNMPv3 monitoring, alert channels, and acceptance testing.

1. Pre-Deployment Operator Survey (Pre-requisites Gathering)

Before running installation scripts, the operator **must** collect the following physical/virtual infrastructure parameters and store them in the deployment matrix:

| REQUIRED PARAMETER | OPERATOR INPUT / DESCRIPTION |
|----------------------------|---|
| Deployment Workstation IP | e.g., 192.168.1.50 |
| Hyper-V Host VM IP | e.g., 192.168.130.170 |
| VirtualBox Host VM IP | e.g., 192.168.130.161 |
| SSH Key Location (Private) | e.g., ~/.ssh/id_rsa |
| SMTP Relay Password | e.g., ***** (For Zabbix/App password reset email) |
| Teams/Discord Webhook URL | e.g., https://discord.com/api/webhooks/... |

2. Platform Mapping: Which Container Goes Where?

To maximize CPU/GPU efficiency and secure database read/writes, services are distributed across three distinct environments:

| COMPONENT CONTAINER | DEPLOYMENT ENVIRONMENT | WHY |
|----------------------------------|--------------------------|--|
| streamlit-app (app.py) | Local WSL2 (Windows) | Low-latency rendering and direct client access |
| mysql (Database Node) | Hyper-V VM (Server A) | Persistent enterprise-grade disk storage |
| ollama (NLP Qwen2.5:7b Engine) | VirtualBox VM (Server B) | Dedicated CPU/GPU virtualization allocation |
| zabbix-server & web (Monitoring) | Hyper-V VM (Server A) | Centralized SNMPv3 alert processing and logs |

3. Platform Provisioning Commands

3.1: WSL2 Provisioning (Local Client Workstation)

Enable WSL2 and install Ubuntu 24.04:

```
# Run in Administrator PowerShell
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
wsl --install -d Ubuntu-24.04
```

3.2: Hyper-V VM Provisioning (Server A - Database & Zabbix)

Deploy a dedicated Ubuntu VM on Hyper-V using PowerShell:

```
# Run in Administrator PowerShell on Server A
New-VM -Name "FoodAI-Database-Node" -MemoryStartupBytes 8GB -
Generation 2 -NewVHDXPath "C:\VMs\FoodAI_DB.vhdx" -VHDXSizeBytes 80GB -
SwitchName "External Switch"
Set-VMFirmware -VMName "FoodAI-Database-Node" -EnableSecureBoot Off
Start-VM -Name "FoodAI-Database-Node"
```

3.3: VirtualBox VM Provisioning (Server B - Ollama AI Engine)

Deploy a dedicated VM on VirtualBox using Command Line:

```
# Run in Command Prompt on Server B
vboxmanage createvm --name "FoodAI-AI-Node" --ostype "Ubuntu_64" --
register
vboxmanage modifyvm "FoodAI-AI-Node" --memory 8192 --cpus 4 --vram
128 --nic1 bridged --bridgeadapter1 "Intel Ethernet Connection"
vboxmanage createhd --filename "C:\VMs\FoodAI_AI.vdi" --size 60000
vboxmanage storagectl "FoodAI-AI-Node" --name "SATA Controller" --add
sata --controller IntelAHCI
vboxmanage storageattach "FoodAI-AI-Node" --storagectl "SATA
Controller" --port 0 --device 0 --type hdd --medium "C:
\VMs\FoodAI_AI.vdi"
vboxmanage startvm "FoodAI-AI-Node" --type headless
```

4. Secure Authentication & SSH Exchange

Exchange SSH public keys to allow automated, passwordless container management across nodes:

```
# 1. Generate SSH Keys on WSL Client
```

```
ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa_foodai -N ""

# 2. Push Key to Database VM (Server A)
ssh-copy-id -i ~/.ssh/id_rsa_foodai.pub operator@192.168.130.170

# 3. Push Key to AI VM (Server B)
ssh-copy-id -i ~/.ssh/id_rsa_foodai.pub operator@192.168.130.161
```

5. Multi-Node Docker Network & Configuration

To allow WSL, Hyper-V, and VirtualBox nodes to communicate, update the `.env` variables and `docker-compose.yml` to use bridged network endpoints.

Step 5.1: Configure WSL Client `.env`

Update `.env` in the Streamlit workspace:

```
DB_HOST=192.168.130.170
DB_USER=food_reader
DB_PASS=reader_pass
APP_AUTH_USER=food_app_auth
APP_AUTH_PASS=auth_pass
OLLAMA_HOST=http://192.168.130.161:11434
SEARXNG_HOST=http://localhost:8080
ZBX_SERVER_HOST=192.168.130.170
```

Step 5.2: Configure Ollama (VirtualBox Server B) Listening Port

Ensure the Ollama daemon inside VirtualBox binds to `0.0.0.0` (all interfaces):

```
# SSH into Server B (192.168.130.161)
sudo systemctl edit ollama.service

# Add the environment variables:
[Service]
Environment="OLLAMA_HOST=0.0.0.0"

# Reload and restart service
sudo systemctl daemon-reload
sudo systemctl restart ollama
```

6. Zabbix Reconfiguration for Multi-Node SNMPv3 Telemetry

To monitor all distributed deployment environments securely:

Step 6.1: Deploy SNMPv3 Daemons

Install and configure SNMPv3 daemons on WSL, Hyper-V Database VM, and VirtualBox AI VM:

```
sudo apt update && sudo apt install -y snmpd
```

Edit `/etc/snmp/snmpd.conf`:

```
# Listen on all interfaces
agentAddress udp:161

# Create secure SNMPv3 User
createUser securityUser SHA "securityAuthPassword" AES
"securityPrivPassword"
rouser securityUser authpriv
```

Restart daemon:

```
sudo systemctl restart snmpd
```

Step 6.2: Configure Zabbix Server Dashboard (Web UI)

- Open Zabbix in your browser at `http://192.168.130.170:8081`.
 - Navigate to **Configuration > Hosts > Create Host**.
 - Create three distinct hosts:
 - **WSL-Workstation** (IP: `192.168.1.50`)
 - **Database-Node** (IP: `192.168.130.170`)
 - **AI-Node** (IP: `192.168.130.161`)
 - Add the **SNMP Interface** pointing to Port 161 for each host.
 - In the **Security Tab**, select SNMPv3, enter Username `securityUser`, select Auth Protocol `SHA / securityAuthPassword`, and Privacy Protocol `AES / securityPrivPassword`.
 - Attach the pre-installed **Local Food AI Telemetry** Template.
-

7. Verifying Alert Channels

7.1: Microsoft Teams / Discord Alert Webhook

To verify Zabbix is communicating with Discord / Teams:

- Trigger a test CPU threshold spike inside WSL:

```
yes > /dev/null & sleep 10 ; killall yes
```

- Verify Zabbix triggers the alert and transmits the notification.
- Check your designated channel for the incoming payload:
 - Expected Output: `[PROBLEM] High CPU Utilization Detected on WSL-Workstation`.

7.2: Password Reset Email (SMTP Gateway)

- In the Streamlit UI Sidebar, select **Reset Password**.
- Trigger a reset link for user `ClinicianA`.
- Check the inbox or SMTP system log (`tail -f /var/log/mail.log` on Server A) to verify outbound delivery.

8. Operator Post-Installation Checklist

Run these test cases to verify the installation:

| TEST CASE ID | ACTIONS TO PERFORM | EXPECTED RESULTS | STATUS |
|--------------|-------------------------------|---|--------|
| TC-OP-01 | Search 'Cheese' on Search Tab | 10+ records returned in <0.04s. Listeria warning flags on unpasteurized. | [] |
| TC-OP-02 | Enter '1.5 cups' in Plate Tab | Parsed and converted to metric grams based on density index. | [] |
| TC-OP-03 | Ask Chat: 'Can I eat sushi?' | llama3.2:3b retrieves database context and flags raw fish as forbidden for pregnancy. | [] |
| TC-OP-04 | Trigger manual db backup | Timestamped compressed .sql.gz created inside backups/ folder. | [] |
| TC-OP-05 | Terminate Ollama Container | Zabbix PROBLEM active alert generated on dashboard in < 30 seconds. | [] |