Non-Profit Fund Accounting System v9.x

Installation Guide – Ubuntu 24.04 LTS Guest in VirtualBox (Windows / macOS Host)

Document version 9.x – July 2025

---

1 Overview

This guide walks through a **clean installation** of the Non-Profit Fund Accounting System v9.x inside an Ubuntu 24.04 LTS virtual machine (VM) running under Oracle VM VirtualBox.
The procedure matches the **working macOS reference environment** (Node.js 18, PostgreSQL 16, Express 5) and leverages the project's cross-platform automation scripts.

---

2 Prerequisites & Host Requirements

Component	Minimum	Recommended
Host OS	Windows 10/11 or macOS 13+	—
CPU	4 physical cores with VT-x / AMD-V	6+ cores
RAM	8 GB	16 GB (allocate ≥ 6 GB to guest)
Disk	40 GB free	80 GB on SSD / NVMe
Software	Oracle VirtualBox ≥ 7.0, Ubuntu 24.04 Desktop ISO	—

> Enable hardware virtualization in BIOS/UEFI before installing VirtualBox.

---

3 Create the VirtualBox VM

1. **Download software**
   • VirtualBox: https://www.virtualbox.org/wiki/Downloads
   • Ubuntu 24.04 ISO: https://ubuntu.com/download/desktop

2. **New VM**
   • Name `Ubuntu24-FundAcct-v9x`
   • Type *Linux* → Version *Ubuntu (64-bit)*
   • Memory **6144 MB** • Processors **4 vCPU**
   • Disk **VDI**, dynamically allocated, **60 GB**

3. **Tweaks**
   Display → Graphics Controller **VBoxSVGA**, enable **3-D Acceleration**
   Network Adapter 1 → **Bridged** (preferred) or **NAT**
   Storage → attach the Ubuntu ISO.

4. **Install Ubuntu**
   Normal installation, enable third-party codecs (optional).

Create user **fundadmin** (will have sudo).
After first boot:

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

4 Install Runtime Dependencies inside the VM

```
# Essential tools
sudo apt install -y git build-essential curl

# Node.js 18 LTS (via NodeSource)
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt install -y nodejs   # node 18.x, npm 10.x+

# PostgreSQL 16 (official PGDG repository)
echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" | \
  sudo tee /etc/apt/sources.list.d/pgdg.list
curl -fsSL https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo apt update
sudo apt install -y postgresql-16

# Verify toolchain
node -v   # v18.x
npm  -v   # 10.x+
psql -V   # 16.x
```

5 Application Installation

5.1 Clone the Repository

```
sudo mkdir -p /opt && cd /opt
sudo git clone https://github.com/tpfbill/nonprofit-fund-accounting-v9.x.git
sudo chown -R $USER:$USER nonprofit-fund-accounting-v9.x
cd nonprofit-fund-accounting-v9.x
```

5.2 Create `.env`

```
cat > .env <<'EOF'
# Database
PGHOST=localhost
PGPORT=5432
PGDATABASE=fund_accounting_db
PGUSER=npfadmin
PGPASSWORD=npfa123

# Server
PORT=3000
EOF
chmod 600 .env
```

5.3 Install Node Dependencies

```
npm ci
```

Dependencies (excerpt from *package.json*):

- express **5.1.0**
- pg 8.16
- http-server 14.1
- concurrently 8.2

6 Database Setup

Option A (one-line, interactive)

Run the hardened Ubuntu helper script (idempotent):

```
scripts/setup-ubuntu-database.sh
```

> **Note** All schema and data-load SQL/JS files live in the repository's top-level `database/` directory (e.g. `database/db-init.sql`).
> The helper script already points there, but if you customise it ensure you **do not** prefix the path with `scripts/`.

The script:

1. Ensures PostgreSQL service is running.
2. Creates role **npfadmin / npfa123**.
3. Creates database **fund_accounting_db** owned by `npfadmin`.
4. Executes `database/db-init.sql` (core schema) and `database/nacha-vendor-payments-schema.sql` (ACH vendor payments).
5. Loads demo data via `database/load-principle-foundation-data.js`.
6. Writes/updates the `.env` file and verifies connectivity.

Option B (manual)

```
# 1. Create role & DB (cross-platform SQL)
sudo -u postgres psql -f database/setup-database-cross-platform.sql

# 2. Core schema
sudo -u postgres psql -d fund_accounting_db -f database/db-init.sql

# 3. NACHA vendor-payment extension
sudo -u postgres psql -d fund_accounting_db -f database/nacha-vendor-payments-schema.sql

# 4. (Optional) Demo data
node database/load-principle-foundation-data.js
```

7 Run the Application

Open **two terminals** or use `npm run dev`.

```
# Terminal 1 – REST API (port 3000)
node server.js
```

```
# Terminal 2 – Static front-end (port 8080)
npx http-server . -p 8080 --no-cache
```

Helper scripts:

```
npm run client   # only front-end
npm run dev      # backend + frontend concurrently
```

Browse to **http://localhost:8080/index.html**.

8 Testing & Verification Checklist

Test	Steps	Expected
API health	`curl http://localhost:3000/api/health`	`{"status":"ok"}`
Dashboard	Open `/index.html`	Summary cards and charts render
Vendor Directory	*Vendor Payments* ➜ *Vendors*	List shows, **Add Vendor** modal works
Payment Batch	*Vendor Payments* ➜ *Batches* ➜ *New*	Entity & Fund dropdowns populate
NACHA File	Create batch ➜ Approve ➜ **Generate NACHA**	`.ACH` file appears & downloads
DB inspection	`psql -d fund_accounting_db -c '\dt'`	16 tables incl. `payment_batches`, `nacha_files`

9 Troubleshooting

Symptom	Resolution
“DB offline” badge	`sudo systemctl restart postgresql` and verify `.env`
Port 3000 in use	`sudo lsof -i:3000` → `kill <PID>`
Empty dropdowns (batch modal)	Re-run step 6.3 (NACHA schema) & restart API
CSS not updating	Hard-refresh (Ctrl-F5) or `--no-cache` flag
Script permission denied	`chmod +x scripts/setup-ubuntu-database.sh`

10 Performance Tips

1. Allocate extra vCPU/RAM to the VM.
2. Enable **Nested Paging** & **KVM Paravirtualization** in VirtualBox.
3. Store the VDI on an SSD/NVMe host drive.
4. Tune PostgreSQL (`shared_buffers = 512MB`, `work_mem = 16MB`).
5. Use **Bridged** networking for faster host↔guest transfers.

11 Security Considerations

- Change default passwords (`npfa123`) before production.
- Keep `.env` file **chmod 600** and outside version control.
- Enable UFW:

  ```
  sudo ufw allow 8080/tcp
  sudo ufw allow 3000/tcp
  sudo ufw enable
  ```

- Regularly apply `apt upgrade` and PostgreSQL minor updates.

- Snapshot the VM after successful installation.

12 Appendix A – Useful Commands

```
# Stop both services
pkill -f http-server
pkill -f node

# Backup the database
sudo -u postgres pg_dump -Fc fund_accounting_db > fundacct_$(date +%F).dump

# Restore
sudo -u postgres pg_restore -d fund_accounting_db -c fundacct_2025-07-22.dump
```

Enjoy your fully-functional Non-Profit Fund Accounting System v9.x on Ubuntu 24.04!

For details on API endpoints and data model, see the in-app **Documentation** tab or `README.md`.