Windows Hyper-V Deployment Guide - Nonprofit Fund Accounting System v8.6

Document Version: 1.0

Last Updated: July 9, 2025

Applicable System Versions: Nonprofit Fund Accounting System (v8.6)

Table of Contents

1. Prerequisites and System Requirements
2. Enabling and Configuring Hyper-V
3. Creating an Ubuntu 22.04 LTS Virtual Machine
4. Installing Required Software
5. Migrating the Application Code
6. Setting Up the Database
7. Configuring the Application
8. Setting Up Networking
9. Production Considerations
10. Troubleshooting Common Issues

1. Prerequisites and System Requirements

1.1 Windows System Requirements

To run the Nonprofit Fund Accounting System v8.6 in a Hyper-V virtual machine, your Windows system should meet the following requirements:

Component	Minimum Requirement	Recommended
Windows Version	Windows 10 Pro, Enterprise, or Education (64-bit) Windows Server 2016 or newer	Windows 11 Pro or Windows Server 2022
Processor	64-bit processor with SLAT (Second Level Address Translation)	Intel Core i5/i7 or AMD Ryzen 5/7 (8th gen or newer)
Memory	8 GB RAM (4 GB for host, 4 GB for VM)	16 GB RAM or more (4 GB for host, 8+ GB for VM)
Storage	100 GB free space (SSD recommended)	250 GB SSD or more
Network	Ethernet adapter with internet access	Gigabit Ethernet

> **Note:** Hyper-V is not available in Windows 10/11 Home editions. You need Pro, Enterprise, or Education editions.

1.2 Virtualization Requirements

1 Ensure Virtualization is Enabled in BIOS/UEFI

Virtualization must be enabled in your system's BIOS/UEFI settings. This is typically labeled as:

- Intel Virtualization Technology (Intel VT-x)
- AMD Virtualization (AMD-V)
- Virtualization Extensions

To check if virtualization is enabled:

1. Open Task Manager (Ctrl+Shift+Esc)
2. Go to the "Performance" tab
3. Select "CPU"
4. Look for "Virtualization: Enabled" at the bottom right

Task Manager showing virtualization enabled

1.3 Required Software

You'll need the following software:

- Windows 10/11 Pro, Enterprise, or Education (64-bit) with Hyper-V feature
- Ubuntu 22.04 LTS ISO image (Download link)
- SSH client (Windows includes OpenSSH by default, or you can use PuTTY)
- SCP client for file transfer (WinSCP recommended)

1.4 Nonprofit Fund Accounting System Requirements

The v8.6 of the Nonprofit Fund Accounting System has the following specific requirements:

- Node.js 18.x or newer
- PostgreSQL 14.x or newer
- NPM 8.x or newer
- Git (for code management)

Important: Version 8.6 includes the Inter-Entity Transfer feature which requires specific database schema updates and configuration. This guide includes the necessary steps to ensure this feature works correctly.

2. Enabling and Configuring Hyper-V

2.1 Enable Hyper-V Feature

1 Enable Hyper-V Using PowerShell (Administrator)

The fastest way to enable Hyper-V is through PowerShell. Run PowerShell as Administrator and execute:

```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Hyper-V -All
```

Your system will need to restart after this command completes.

2 Alternative: Enable Hyper-V Using Windows Features

1. Press **Windows + R**, type `control panel`, and press Enter
2. Go to **Programs > Programs and Features**
3. Click **Turn Windows features on or off**
4. Check the box next to **Hyper-V** and click OK
5. Restart your computer when prompted

Windows Features dialog with Hyper-V selected

2.2 Create a Virtual Switch

1 Open Hyper-V Manager

Press **Windows + R**, type `virtmgmt.msc`, and press Enter.

2 Create a Virtual Switch

1. In Hyper-V Manager, select your computer name in the left panel

2. Click **Virtual Switch Manager** in the right Actions panel
3. Select **New virtual network switch** in the left panel
4. Choose **External** and click **Create Virtual Switch**
5. Name the switch (e.g., "External Network")
6. Ensure **External network** is selected and choose your physical network adapter
7. Click **OK**

Hyper-V Virtual Switch Manager

Warning: Creating an external virtual switch may temporarily disconnect your network connection as the network adapter is reconfigured.

3. Creating an Ubuntu 22.04 LTS Virtual Machine

3.1 Download Ubuntu 22.04 LTS

1 Download Ubuntu 22.04 LTS ISO

Download the Ubuntu 22.04 LTS Desktop ISO from the official Ubuntu website.

3.2 Create a New Virtual Machine

1 Open Hyper-V Manager

Press **Windows + R**, type `virtmgmt.msc`, and press Enter.

---

2 Create a New Virtual Machine

1. In Hyper-V Manager, click **New > Virtual Machine** in the Actions panel
2. In the New Virtual Machine Wizard:

    - **Name:** Enter "Ubuntu-NFA-v86" (or your preferred name)
    - **Generation:** Choose **Generation 2** (for better performance)
    - **Memory:** Assign at least 4096 MB (4 GB), preferably 8192 MB (8 GB)
    - Check **Use Dynamic Memory**
    - **Networking:** Select the external virtual switch you created earlier
    - **Hard Disk:** Create a virtual hard disk with at least 60 GB
    - **Installation Options:** Select **Install an operating system from a bootable image file** and browse to your Ubuntu ISO

3. Click **Finish** to create the VM

3.3 Configure VM Settings for Ubuntu

1 Adjust VM Settings for Ubuntu Compatibility

1. In Hyper-V Manager, right-click the new VM and select **Settings**
2. Under **Security**, disable Secure Boot or select **Microsoft UEFI Certificate Authority**
3. Under **Processor**, assign at least 2 virtual processors (4 recommended)
4. Click **OK** to save the settings

VM Settings dialog with security options

3.4 Install Ubuntu 22.04 LTS

1 Start the VM and Install Ubuntu

1. In Hyper-V Manager, right-click the VM and select **Start**
2. Click **Connect** to open the VM console
3. Follow the Ubuntu installation prompts:
    - Select your language and click **Install Ubuntu**
    - Choose your keyboard layout
    - For "Updates and other software," select **Normal installation** and check **Download updates while installing Ubuntu**

    - For "Installation type," choose **Erase disk and install Ubuntu** (this only affects the virtual disk)
    - Select your time zone
    - Create a user account with a strong password (remember these credentials!)
    - Wait for the installation to complete and restart when prompted

2 Install Hyper-V Integration Services

After Ubuntu is installed and you've logged in:

```
sudo apt update
sudo apt install -y linux-image-virtual linux-tools-virtual linux-cloud-tools-virtual
```

3 Update Ubuntu

Ensure your Ubuntu system is fully updated:

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

4. Installing Required Software

4.1 Install Basic Development Tools

1 Install Essential Packages

```
sudo apt update
sudo apt install -y build-essential curl git unzip
```

4.2 Install Node.js and NPM

1 Install Node.js 18.x

The Nonprofit Fund Accounting System v8.6 requires Node.js 18.x or newer:

```
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt install -y nodejs
```

2 Verify Node.js and NPM Installation

```
node --version
npm --version
```

You should see output confirming Node.js v18.x and NPM v8.x or newer.

4.3 Install and Configure PostgreSQL

1 Install PostgreSQL 14

```
sudo apt install -y postgresql-14 postgresql-contrib-14
```

2 Verify PostgreSQL Installation

```
sudo systemctl status postgresql
```

You should see that PostgreSQL is active and running.

3 Configure PostgreSQL for Remote Access

Edit the PostgreSQL configuration file to allow connections from your Windows host:

```
sudo nano /etc/postgresql/14/main/postgresql.conf
```

Find the line with `listen_addresses` and change it to:

```
listen_addresses = '*'
```

Now edit the client authentication configuration:

```
sudo nano /etc/postgresql/14/main/pg_hba.conf
```

Add the following line at the end of the file (replace with your actual subnet if needed):

```
host all all 0.0.0.0/0 md5
```

Restart PostgreSQL to apply the changes:

```
sudo systemctl restart postgresql
```

Warning: The above configuration allows connections from any IP address. In a production environment, you should restrict this to specific IP addresses or subnets.

4 Create PostgreSQL User and Database

```
sudo -u postgres psql
```

In the PostgreSQL prompt, run the following commands:

```
CREATE USER nfaadmin WITH PASSWORD 'your_secure_password';
CREATE DATABASE fund_accounting_db OWNER nfaadmin;
ALTER USER nfaadmin WITH SUPERUSER;
\q
```

Replace 'your_secure_password' with a strong password.

4.4 Install Additional Dependencies

1 Install Additional Required Packages

```
sudo apt install -y nginx prince
```

Prince is required for PDF generation in the application.

5. Migrating the Application Code

5.1 Prepare the Application Directory

1 Create Application Directory

```
sudo mkdir -p /opt/nonprofit-fund-accounting
sudo chown $USER:$USER /opt/nonprofit-fund-accounting
```

5.2 Transfer Application Files

1 Option 1: Clone from Git Repository

If your code is in a Git repository:

```
cd /opt/nonprofit-fund-accounting
git clone https://github.com/your-organization/nonprofit-fund-accounting.git
.
git checkout v8.6
```

Replace the repository URL with your actual repository.

---

2 Option 2: Transfer Files from Existing System

If transferring from an existing system, use SCP or SFTP. From your Windows machine with WinSCP:

1. Install WinSCP if you haven't already
2. Connect to your Ubuntu VM using its IP address and your credentials
3. Navigate to the source directory on your local machine
4. Navigate to `/opt/nonprofit-fund-accounting` on the VM
5. Copy all files from the source to the VM

Alternatively, you can use the built-in SCP command from PowerShell:

```
scp -r C:\path\to\nonprofit-fund-accounting\* username@vm-ip-address:/opt/
nonprofit-fund-accounting/
```

5.3 Install Application Dependencies

1 Install Node.js Dependencies

```
cd /opt/nonprofit-fund-accounting
npm install
```

---

6. Setting Up the Database

6.1 Migrate Database Schema

1 Option 1: Run Database Initialization Script

If your application includes a database initialization script:

```
cd /opt/nonprofit-fund-accounting
node database-init.js
```

Replace `database-init.js` with your actual initialization script.

2 Option 2: Import Database from Existing System

If you have a database dump from your existing system:

First, on your existing system, create a database dump:

```
pg_dump -U postgres fund_accounting_db > fund_accounting_backup.sql
```

Transfer the dump file to your Ubuntu VM using SCP or SFTP.

Then, on the Ubuntu VM, import the database:

```
psql -U nfaadmin -d fund_accounting_db < /path/to/fund_accounting_backup.sql
```

6.2 Verify Database Schema for Inter-Entity Transfers

1 Check for Required Tables and Columns

The Inter-Entity Transfer feature in v8.6 requires specific database tables and columns. Connect to the database and verify:

```
psql -U nfaadmin -d fund_accounting_db
```

In the PostgreSQL prompt, run these queries to check for required tables and columns:

```
\dt
-- Should include tables: entities, accounts, journal_entries,
journal_entry_lines

\d journal_entries
-- Should include columns: is_inter_entity, matching_transaction_id,
target_entity_id

\d accounts
-- Should include proper account types for Due To/Due From accounts

\q
```

2 Apply Any Missing Schema Updates

If any required tables or columns are missing, you may need to apply schema updates. If your application includes migration scripts, run them:

```
cd /opt/nonprofit-fund-accounting
node migrations/update-to-v8.6.js
```

Replace with your actual migration script.

6.3 Set Up Test Data (Optional)

1 Load Test Data

If you want to set up test data, run your data seeding script:

```
cd /opt/nonprofit-fund-accounting
node add-test-data.js
```

---

7. Configuring the Application

7.1 Create Environment Configuration

1 Set Up Environment Variables

Create or update the .env file with the correct configuration for your VM environment:

```
cd /opt/nonprofit-fund-accounting
nano .env
```

Add or update the following variables:

```
# Database Configuration
DB_HOST=localhost
DB_PORT=5432
DB_NAME=fund_accounting_db
DB_USER=nfaadmin
DB_PASSWORD=your_secure_password

# Application Configuration
NODE_ENV=production
PORT=3000
HOST=0.0.0.0

# Feature Flags
ENABLE_INTER_ENTITY_TRANSFERS=true
```

Replace `your_secure_password` with the actual password you set for the PostgreSQL user.

7.2 Configure Application for Production

1 Update Production Settings

If your application has specific production configuration files, update them as needed:

```
cd /opt/nonprofit-fund-accounting
nano config/production.js
```

Ensure the configuration matches your VM environment.

7.3 Set Up Inter-Entity Transfer Configuration

1 Configure Inter-Entity Transfer Settings

Version 8.6 includes the Inter-Entity Transfer feature which requires specific configuration:

```
cd /opt/nonprofit-fund-accounting
nano config/inter-entity-config.js
```

Ensure the configuration includes:

- Due To/Due From account mapping
- Entity relationship definitions
- Approval workflow settings (if applicable)

7.4 Test the Application

1 Start the Application Manually

```
cd /opt/nonprofit-fund-accounting
node server.js
```

The application should start and be accessible at http://vm-ip-address:3000 (replace with your

VM's actual IP address).

---

2 Verify Application Functionality

Test key functionality to ensure everything is working correctly:

- User login
- Chart of accounts access
- Fund management
- Entity management
- Journal entry creation
- Inter-Entity Transfer functionality
- Report generation

---

8. Setting Up Networking

8.1 Configure Firewall

1 Set Up UFW Firewall

Configure the Ubuntu firewall to allow necessary connections:

```
sudo apt install -y ufw
sudo ufw allow ssh
sudo ufw allow 3000/tcp
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw enable
```

Confirm the firewall is active:

```
sudo ufw status
```

8.2 Set Up Nginx as a Reverse Proxy (Recommended)

1 Configure Nginx

Using Nginx as a reverse proxy provides better security and performance:

```
sudo nano /etc/nginx/sites-available/nonprofit-fund-accounting
```

Add the following configuration:

```
server {
  listen 80;
  server_name your-server-name;

  location / {
    proxy_pass http://localhost:3000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
  }
}
```

Replace `your-server-name` with your VM's hostname or IP address.

2 Enable the Nginx Configuration

```
sudo ln -s /etc/nginx/sites-available/nonprofit-fund-accounting /etc/nginx/
sites-enabled/
sudo nginx -t
sudo systemctl restart nginx
```

8.3 Configure Static IP Address (Recommended)

1 Set a Static IP for the VM

Configure a static IP address to ensure the VM always has the same IP:

```
sudo nano /etc/netplan/01-netcfg.yaml
```

Add a configuration like this (adjust based on your network):

```
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: no
      addresses: [192.168.1.100/24]
      gateway4: 192.168.1.1
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
```

Apply the configuration:

```
sudo netplan apply
```

8.4 Set Up Host Name Resolution (Optional)

1 Add Host Entry on Windows

For easier access, add an entry to your Windows hosts file:

1. Open Notepad as Administrator
2. Open the file `C:\Windows\System32\drivers\etc\hosts`
3. Add a line like: `192.168.1.100 nfa-system` (replace with your VM's IP)
4. Save the file

Now you can access the system using `http://nfa-system/` in your browser.

9. Production Considerations

9.1 Set Up Automatic Startup

1 Create a Systemd Service

Create a systemd service to automatically start the application on boot:

```
sudo nano /etc/systemd/system/nonprofit-fund-accounting.service
```

Add the following configuration:

```
[Unit]
Description=Nonprofit Fund Accounting System v8.6
After=network.target postgresql.service

[Service]
Type=simple
User=ubuntu
WorkingDirectory=/opt/nonprofit-fund-accounting
ExecStart=/usr/bin/node server.js
Restart=on-failure
Environment=NODE_ENV=production

[Install]
WantedBy=multi-user.target
```

Replace `ubuntu` with your actual username.

2 Enable and Start the Service

```
sudo systemctl enable nonprofit-fund-accounting
sudo systemctl start nonprofit-fund-accounting
sudo systemctl status nonprofit-fund-accounting
```

9.2 Set Up Regular Backups

1 Create a Backup Script

```
sudo nano /opt/backup-nonprofit-system.sh
```

Add the following script:

```
#!/bin/bash
TIMESTAMP=$(date +"%Y%m%d_%H%M%S")
BACKUP_DIR="/opt/backups"
DB_BACKUP="$BACKUP_DIR/db_backup_$TIMESTAMP.sql"
APP_BACKUP="$BACKUP_DIR/app_backup_$TIMESTAMP.tar.gz"

# Create backup directory if it doesn't exist
mkdir -p $BACKUP_DIR

# Backup the database
sudo -u postgres pg_dump fund_accounting_db > $DB_BACKUP

# Backup the application files
tar -czf $APP_BACKUP -C /opt nonprofit-fund-accounting

# Keep only the 10 most recent backups
ls -t $BACKUP_DIR/db_backup_* | tail -n +11 | xargs -r rm
ls -t $BACKUP_DIR/app_backup_* | tail -n +11 | xargs -r rm

echo "Backup completed: $(date)"
```

Make the script executable:

```
sudo chmod +x /opt/backup-nonprofit-system.sh
```

2 Schedule Regular Backups with Cron

```
sudo crontab -e
```

Add the following line to run backups daily at 2 AM:

```
0 2 * * * /opt/backup-nonprofit-system.sh >> /var/log/nonprofit-backup.log
2>&1
```

9.3 Security Considerations

1 Secure the Application

- **Keep the system updated:**

  ```
  sudo apt update
  sudo apt upgrade
  ```

- **Set up automatic security updates:**

  ```
  sudo apt install -y unattended-upgrades
  sudo dpkg-reconfigure -plow unattended-upgrades
  ```

- **Restrict SSH access:**

  ```
  sudo nano /etc/ssh/sshd_config
  ```

  Make these changes:

```
PermitRootLogin no
PasswordAuthentication no
AllowUsers your_username
```

Restart SSH:

```
sudo systemctl restart ssh
```

- **Set up fail2ban to prevent brute force attacks:**

```
sudo apt install -y fail2ban
sudo systemctl enable fail2ban
sudo systemctl start fail2ban
```

9.4 Monitoring and Logging

1 Set Up Application Logging

Ensure your application logs are properly configured:

```
sudo mkdir -p /var/log/nonprofit-fund-accounting
sudo chown ubuntu:ubuntu /var/log/nonprofit-fund-accounting
```

Update your application's logging configuration to use this directory.

---

2 Set Up Log Rotation

```
sudo nano /etc/logrotate.d/nonprofit-fund-accounting
```

Add the following configuration:

```
/var/log/nonprofit-fund-accounting/*.log {
  daily
  missingok
  rotate 14
  compress
  delaycompress
  notifempty
  create 0640 ubuntu ubuntu
}
```

---

10. Troubleshooting Common Issues

10.1 Application Startup Issues

1 Check Application Logs

```
cd /opt/nonprofit-fund-accounting
cat app.log
# Or if using systemd
sudo journalctl -u nonprofit-fund-accounting
```

---

2 Common Startup Issues and Solutions

Issue	Possible Cause	Solution
EADDRINUSE error	Port 3000 is already in use	`sudo lsof -i :3000` `sudo kill -9 [PID]`

Database connection error	PostgreSQL not running or incorrect credentials	`sudo systemctl status postgresql`
		`sudo systemctl restart postgresql`
		`# Check credentials in .env file`

Missing dependencies	NPM packages not installed	`cd /opt/nonprofit-fund-accounting`
		`npm install`

Permission issues	Incorrect file permissions	`sudo chown -R ubuntu:ubuntu /opt/nonprofit-fund-accounting`
		`chmod -R 755 /opt/nonprofit-fund-accounting`

10.2 Database Issues

1 Common Database Issues and Solutions

Issue	Possible Cause	Solution

PostgreSQL not accepting connections	Configuration issues or service not running	`sudo systemctl restart postgresql`
		`# Check pg_hba.conf and postgresql.conf`

Missing tables for Inter-Entity Transfers	Schema not updated for v8.6	`# Run the v8.6 migration script`
		`cd /opt/nonprofit-fund-accounting`
		`node migrations/update-to-v8.6.js`

Permission denied errors	PostgreSQL user lacks necessary permissions	`sudo -u postgres psql`
		`ALTER USER nfaadmin WITH SUPERUSER;`
		`\q`

10.3 Networking Issues

1 Common Networking Issues and Solutions

Issue	Possible Cause	Solution

Cannot access application from Windows host	Firewall blocking connections	`sudo ufw status`
		`sudo ufw allow 3000/tcp`
		`# Check Windows Firewall settings`

Application only accessible on localhost	Application not binding to all interfaces	`# Ensure HOST=0.0.0.0 in .env file`
		`# Check server.js for binding address`

Nginx proxy not working	Misconfiguration or service not running	`sudo nginx -t`
		`sudo systemctl restart nginx`
		`# Check error logs: sudo tail -f /var/log/nginx/error.log`

10.4 Inter-Entity Transfer Issues

1 Common Inter-Entity Transfer Issues and Solutions

Issue	Possible Cause	Solution

Inter-Entity Transfer feature not available	Feature flag not enabled	`# Ensure ENABLE_INTER_ENTITY_TRANSFERS=true in .env file`

Transfers fail with database errors	Missing database schema updates	`# Check for required columns:`
		`psql -U nfaadmin -d fund_accounting_db -c "\d journal_entries"`
		`# Run migration script if needed`

Due To/Due From accounts not working	Incorrect account configuration	`# Check account setup:`
		`psql -U nfaadmin -d fund_accounting_db -c "SELECT * FROM accounts WHERE name LIKE 'Due To%' OR name LIKE 'Due From%'"`
		`# Update configuration in inter-entity-config.js`