AI-03022 Docker Inventree in Ubuntu 25

Owner	Ⓡ Rigel Arcayan
Tags	Project
Created time	@September 22, 2025 11:36 AM

I’m new to docker and I have not installed docker in my Ubuntu server. Create a comprehensive guide to install docker and deploy Inventree using docker from start to finish

Here is a comprehensive guide to installing Docker and deploying Inventree using Docker from start to finish. This guide is tailored for an Ubuntu server and assumes you have `sudo` access.

Part 1: Install Docker on Ubuntu 🐳

Docker is an open-source platform that simplifies the process of building, sharing, and running applications in containers. Installing it on Ubuntu is straightforward.

1. **Update Your System**: Before installing any new software, it's a good practice to update your system's package list and upgrade existing packages.Bash

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

2. **Install Necessary Packages**: Install a few packages that allow `apt` to use a repository over HTTPS.Bash

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

3. **Add Docker's GPG Key**: Docker's GPG key is used to ensure the authenticity of the packages you're about to install.Bash

   ```
   sudo install -m 0755 -d /etc/apt/keyrings
   curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
   ```

4. **Add Docker Repository**: Add the Docker repository to your `apt` sources. This tells your system where to download the Docker packages.Bash

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

5. **Install Docker Engine and Docker Compose**: Now, update your `apt` package index again and install the Docker Engine, CLI, and Docker Compose plugin.Bash

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

6. **Verify the Installation**: To confirm Docker is installed correctly, run the `hello-world` container.Bash

   ```
   sudo docker run hello-world
   ```

   If successful, this command downloads a test image and prints a message confirming the installation.

Part 2: Deploy Inventree with Docker Compose ⚙️

InvenTree is an open-source inventory management system. The easiest way to deploy it is using **Docker Compose**, which manages multi-container Docker applications.

2.1: Initial Setup

1. **Create a Directory**: Create a dedicated directory for your Inventree deployment. This keeps all configuration files organized.Bash

   ```
   mkdir inventree
   cd inventree
   ```

2. **Download Configuration Files**: The InvenTree team provides a set of pre-configured files to simplify deployment. You'll need the `docker-compose.yml`, `.env`, and `Caddyfile`. You can download them directly from the official InvenTree GitHub repository.Bash

   ```
   curl -o docker-compose.yml https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/docker-compose.yml
   curl -o .env https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/.env.example
   curl -o Caddyfile https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/Caddyfile
   ```

   Alternatively, you can manually copy and paste the contents into new files.

3. **Configure the Environment**: The `.env` file contains important variables that customize your InvenTree instance. Use a text editor like `nano` to edit it.Bash `nano .env`

   In the `.env` file, you should at a minimum:

   - Set the `INVENTREE_SECRET_KEY` to a strong, random string. You can generate one using an online tool or a command like `openssl rand -hex 32`.
   - Set the `INVENTREE_SITE_URL` to your server's public IP address or domain name, e.g., `http://192.168.1.100` or `https://inventree.yourdomain.com`.
   - Change the `INVENTREE_DB_PASSWORD` to a secure password.

   Save the file and exit the editor.

2.2: Deploy and Access

1. **Start the Containers**: With the configuration files in place, you can now start the InvenTree stack. The `d` flag runs the containers in the background.Bash `sudo docker compose up -d`

   This command will download the necessary Docker images for the InvenTree server, a PostgreSQL database, a Caddy proxy, and a background worker, then start them all. This can take a few minutes.

2. **Check Container Status**: You can verify that all containers are running correctly with the following command:Bash `sudo docker compose ps`

   The `State` column for all services should show `Up`.

3. **Run Initial Setup**: The first time you run InvenTree, you need to set up the database.Bash `sudo docker compose run --rm inventree-server invoke update`

   This command runs the `invoke update` script which applies database migrations and performs other necessary setup tasks.

4. **Create an Admin User**: Finally, create the initial administrator account to log in to InvenTree.Bash `sudo docker compose run --rm inventree-server invoke superuser`

Follow the prompts to enter a username, email address, and password for your admin account.

2.3: Access InvenTree

Once all the steps are complete, you can access your InvenTree instance by navigating to the `INVENTREE_SITE_URL` you specified in the `.env` file in your web browser. You should be greeted by the login page, where you can use the admin credentials you just created.

If you encounter any issues, you can view the logs for a specific service to troubleshoot, for example:

Bash

```
sudo docker compose logs inventree-server
```

Sudo commands during deployment

Bash

```
rarcayan@ubuntuserver-7:~$ cat ~/.bash_history
sudo apt update
sudo apt upgrade
sudo nano /etc/netplan/01-netcfg.yaml
sudo netplan apply
ip a
sudo reboot now
sudo apt install webmin --install-recommends
sudo apt install apt-transport-https curl gnupg software-properties-common -y
curl -fsSL https://download.webmin.com/jcameron-key.asc | sudo gpg --dearmor -o /usr/share/keyrings/webmin.gpg
sudo nano /etc/apt/sources.list
sudo apt install webmin -y
wget https://raw.githubusercontent.com/webmin/webmin/master/setup-repos.sh
sudo bash setup-repos.sh
sudo ufw allow 10000/tcp
sudo shutdown now
sudo apt install ca-certificates curl gnupg lsb-release -y
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
sudo docker run hello-world
sudo mkdir inventree
cd inventree
```

```
ls
cd ..
curl -o docker-compose.yml https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/docker-compose.yml
curl -o .env https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/.env.example
curl -o Caddyfile https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/Caddyfile
sudo curl -o docker-compose.yml
https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/docker-compose.yml
sudo curl -o .env https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/.env.example
sudo curl -o Caddyfile https://raw.githubusercontent.com/inventree/InvenTree/master/docker/production/Caddyfile
openssl rand -base64 32
cd ~
mkdir -p ./inventree-data
docker compose run --rm inventree-server invoke update
sudo docker compose run --rm inventree-server invoke update
sudo rm -r /home/rarcayan/inventree-data
docker compose run inventree-server invoke superuser
sudo docker compose run inventree-server invoke superuser
sudo docker compose run --rm inventree-server invoke migrate
sudo mkdir -p /home/rarcayan/inventree-db-data
exit
ssh rarcayan@192.168.1.7
docker compose up -d
sudo docker compose up -d
sudo docker compose down
sudo nano /home/rarcayan/inventree-data/config.yaml
sudo docker compose logs inventree-server
sudo docker compose restart inventree-server
sudo pip3 show django-dbbackup
sudo pip3 install django-dbbackup
psql -U your_db_user -h localhost -d your_db_name
sudo psql -U your_db_user -h localhost -d your_db_name
sudo apt install -y build-essential zlib1g-dev libncurses5-dev libgdbm-dev libnss3-dev libssl-dev libreadline-dev libffi-dev curl libsqlite3-dev wget
cd /usr/src
sudo wget https://www.python.org/ftp/python/3.12.0/Python-3.12.0.tgz
sudo tar xzf Python-3.12.0.tgz
cd Python-3.12.0
sudo ./configure --enable-optimizations
sudo make -j$(nproc)
sudo make altinstall
python3.12 --version
sudo update-alternatives --install /usr/bin/python3 python3 /usr/local/bin/python3.12 1
sudo docker compose up
sudo docker compose down
sudo docker compose down -v
sudo docker exec -it inventree-db bash
sudo docker compose ps
sudo ufw allow 80
sudo docker compose logs -f inventree-server
docker volume ls
sudo docker volume ls
```

```
docker inspect inventree-db
sudo docker inspect inventree-db
sudo rm -rf ./inventree-data
sudo docker compose logs inventree-db
sudo docker exec -it inventree-server python3 manage.py migrate
sudo docker exec -it inventree-server python3 /home/inventree/InvenTree/manage.py migrate
sudo docker exec -it inventree-server python3 /home/inventree/InvenTree/manage.py createsuperuser
sudo docker exec -it inventree-server bash
sudo docker ps
docker logs inventree-server --tail=50
docker logs inventree-frontend --tail=50
docker logs inventree-worker --tail=50
sudo docker logs inventree-server --tail=50
sudo docker logs inventree-frontend --tail=50
sudo docker logs inventree-worker --tail=50
sudo docker ps | grep inventree
curl http://localhost:8000
sudo ufw allow 8000
sudo docker exec -it inventree-proxy ping inventree-server
sudo docker exec -it inventree-server invoke update
sudo docker exec -it inventree-server invoke collectstatic
sudo shutdown now
```

I will categorize the provided commands from your bash history and provide a short description for each category.

System and Package Management

These commands are used for managing the Ubuntu system and its software packages.

- `sudo apt update`: Updates the list of available packages from the repositories.
- `sudo apt upgrade`: Upgrades all installed packages to their latest versions.
- `sudo reboot now`: Restarts the system immediately.
- `sudo shutdown now`: Shuts down the system immediately.
- `sudo apt install <package_name>`: Installs new software packages.
- `sudo apt install -y`: The `y` flag automatically answers yes to prompts.
- `sudo apt install --install-recommends`: Installs recommended packages along with the main one.
- `sudo apt-transport-https curl gnupg software-properties-common -y`: Installs essential tools for managing software repositories.

- `sudo install -m 0755 -d /etc/apt/keyrings`: Creates a directory with specific permissions.
- `sudo chmod a+r /etc/apt/keyrings/docker.gpg`: Changes the file permissions to allow read access for all users.
- `sudo bash setup-repos.sh`: Executes a shell script to set up repositories.
- `sudo ufw allow <port>`: Configures the Uncomplicated Firewall to allow traffic on a specific port.

File and Directory Operations

These commands are used for navigating and manipulating files and directories.

- `cat ~/.bash_history`: Displays the contents of the `.bash_history` file.
- `sudo nano <file_path>`: Opens a file for editing using the nano text editor. `sudo` is used for administrative privileges.
- `ip a`: Shows network interface addresses.
- `ls`: Lists files and directories in the current directory.
- `cd <directory_name>`: Changes the current working directory.
- `cd ..`: Moves up one level in the directory tree.
- `cd ~`: Changes to the user's home directory.
- `sudo mkdir <directory_name>`: Creates a new directory with administrative privileges.
- `sudo rm -r <directory_name>`: Deletes a directory and its contents recursively.

Docker and Container Management

These commands are for managing Docker containers, images, and services.

- `sudo docker run <image_name>`: Runs a command in a new container.
- `sudo docker compose up -d`: Starts the services defined in a `docker-compose.yml` file in detached mode.
- `sudo docker compose down`: Stops and removes containers, networks, and volumes for a project.

- `sudo docker compose up -d --remove-orphans`: Starts services and removes any containers not defined in the compose file.
- `sudo docker compose down -v`: Stops services and removes containers and associated volumes.
- `sudo docker compose ps`: Lists the containers for a project.
- `sudo docker compose logs`: Displays log output from containers.
- `sudo docker compose restart`: Restarts a container.
- `sudo docker exec -it <container> <command>`: Executes a command inside a running container.
- `docker volume ls`: Lists Docker volumes.
- `docker inspect <object>`: Returns detailed information about a Docker object.
- `docker ps`: Lists running containers.

Network and Data Transfer

These commands are used for network-related tasks and downloading files.

- `curl <URL>`: Transfers data from a URL.
- `wget <URL>`: Downloads a file from a URL.
- `ssh <user>@<host>`: Connects to a remote server using SSH.
- `ping <host>`: Tests connectivity to a host.

Application-Specific and Development

These commands are specific to the InvenTree application, Python, and other development tasks.

- `sudo netplan apply`: Applies the network configuration defined in Netplan.
- `curl -o <filename> <URL>`: Downloads content from a URL and saves it to a specified file.
- `openssl rand -base64 32`: Generates a random 32-byte string encoded in base64.

- `sudo pip3 show <package_name>`: Shows information about an installed Python package.
- `sudo pip3 install <package_name>`: Installs a Python package.
- `psql`: A command-line client for PostgreSQL.
- `sudo ./configure --enable-optimizations`: Configures the source code for building, often enabling optimizations.
- `sudo make`: Compiles the source code.
- `sudo make altinstall`: Installs the compiled program, often in a way that doesn't overwrite the default system version.
- `python3.12 --version`: Checks the version of a specific Python installation.
- `sudo update-alternatives --install ...`: Manages symbolic links for a command, allowing different versions to be used.