Training guide
Odoo 19

Installing Odoo

Step by step on Linux, Windows and Docker: the source code, PostgreSQL and its user, the configuration file, IP, port and DNS, and what to do when it fails.

1 **Prepare** the machine, Python and the libraries

2 **PostgreSQL** and the user Odoo connects with

3 **Odoo** from source, from the installer, or as a container

4 **Configure** odoo.conf, IP address, port, DNS, HTTPS

5 **Run, check, secure** and read the errors

Every step comes with the reason for it and what you should see. Written for someone installing Odoo for the first time.

Powered by **Amir Akbari**

October 2026

The short version

Installing Odoo means putting five pieces on a machine and telling them how to find each other. Almost every installation problem is one piece that cannot find another.

	Piece	What it is	How Odoo finds it
1	**Python** and its libraries	The language Odoo is written in, plus about sixty libraries it uses.	You start Odoo with the Python that has the libraries.
2	**PostgreSQL**	The database server.	Host, port, user and password in the configuration.
3	**The Odoo source code**	A folder with the server and the standard modules.	The path you run, and the `addons_path`.
4	**wkhtmltopdf**	A small program that turns reports into PDF.	It must be on the system path.
5	**The configuration file**	One text file, `odoo.conf`, that holds every setting.	The `-c` option when Odoo starts.

Which way should I install?

Your situation	Method	Section
A server for a company, on Linux	From source, as a service	5, then 8 to 13
A developer's laptop, Linux or macOS	From source	5, steps 1 to 10
A first look, on Windows, in ten minutes	The Windows installer	6, route A
Development on Windows	From source on Windows, or WSL	6, routes B and C
You already use Docker, or want several versions side by side	Docker Compose	7

> **Three rules that prevent most trouble**
> Do every step in the order given and check its result before the next one. Read the last twenty lines of the log whenever something fails: the answer is nearly always written there. And never run Odoo as `root` or connect to the database as `postgres`.

What is in this guide

Understand

1 What you are installing
2 Four identities, four passwords
3 Choosing a method
4 Before you start

Install

5 Linux, from source
6 Windows
7 Docker

Connect and configure

8 Odoo and PostgreSQL
9 The configuration file
10 IP, port, DNS, HTTPS
11 First start and first users

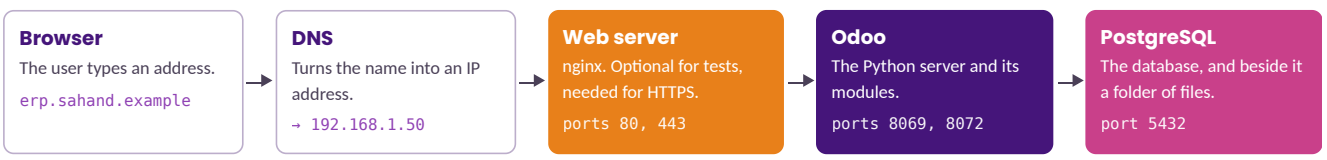
Operate and repair

12 Day-two operations
13 Securing a server
14 Errors and fixes
15 Cheat sheet and checks

Scope: Odoo 19 Community, with notes for Enterprise. Linux commands are for Ubuntu 24.04 LTS and work with small changes on Debian 12. Names used in the examples: system user `odoo`, folder `/opt/odoo`, database `sahand`, server address 192.168.1.50, domain erp.sahand.example. Replace them with your own. Operating systems are marked Linux Windows Docker.

1 What you are installing

Odoo is not one program you double-click. It is a Python program that needs a database server beside it, and usually a web server in front of it.



Each arrow is a connection that must be configured, and each box listens on a **port**: a numbered door on the machine. When a request cannot get through, you find the broken arrow by testing them from left to right.

The pieces in detail

Piece	Version for Odoo 19	Why Odoo needs it	If it is missing
Python	3.10 or later	Odoo is Python code. The libraries in `requirements.txt` must match the version.	Odoo does not start: `ModuleNotFoundError`.
PostgreSQL	13 or later	The only database Odoo supports. All business data lives there.	Odoo starts and every page fails with a connection error.
Odoo source	Branch 19.0	The server (`odoo-bin`) and the standard modules.	Nothing to run.
wkhtmltopdf	0.12.6	Converts quotations, invoices and other reports to PDF.	Odoo works; printing fails or loses headers and footers.
Node.js and rtlcss	Any recent	Builds the right-to-left stylesheet for Persian, Arabic, Hebrew.	Odoo works in English; RTL languages look broken.
Filestore folder		Where attachments and generated assets are kept on disk.	Pages load without styling; attachments fail.
nginx	Any recent	Encryption (HTTPS), the standard web ports, routing of live notifications.	Odoo works on port 8069 without encryption. Fine for tests only.

The ports

Port	Used by	Open it to
8069	Odoo, normal requests	The web server on the same machine. To the network only while testing.
8072	Odoo, live notifications (WebSocket)	The web server only. Used when Odoo runs with several workers.
5432	PostgreSQL	Odoo only. Never the internet.
80, 443	nginx: HTTP and HTTPS	The users.
22	SSH, for administering a Linux server	Administrators.

2 Four identities, four passwords

"Create the user" means four different things during an installation. Mixing them up is the most common source of confusion, so sort them out before typing anything.

System user
`odoo`
An account of the operating system. The Odoo process runs under it and owns the source and data folders.
Created with `useradd`

Database role
`odoo`
An account inside PostgreSQL. Odoo logs in to the database server with it and owns the databases it creates.
Created with `createuser`

Master password
`admin_passwd`
Not a user. One password in `odoo.conf` that protects the page where databases are created, copied and deleted.
Set in the configuration file

Odoo user
`admin`
A login of the application, stored *inside* one Odoo database. The first one is the administrator.
Created in Settings › Users

Identity	Lives in	Who uses it	If it is wrong
System user	The operating system	The service manager, to start Odoo	"Permission denied" on files and folders.
Database role	PostgreSQL	The Odoo server, on every request	"role does not exist", "authentication failed".
Master password	`odoo.conf`	Whoever opens the database manager page	"Access Denied" when creating or restoring a database.
Odoo user	The Odoo database	People logging in through the browser	"Wrong login/password" on the login page.

Why the first two usually share a name. On Linux, PostgreSQL can identify a local connection by the system account that makes it. If the system user and the database role are both called `odoo`, Odoo connects without any password stored anywhere. Section 8 explains this in full.

3 Choosing a method

There are three ways to put Odoo on a machine. They install the same software; they differ in how much you see and control.

	Packaged installer	Source code	Docker
What you do	Run an installer (`.exe`, `.deb`, `.rpm`).	Download the code with git and install each dependency yourself.	Describe the containers in one file and start them.
Time to first screen	Minutes	About an hour the first time	Minutes, once Docker is installed
You learn	Little	How every piece connects	Docker; less about Odoo
Updating Odoo	Install the newer package	`git pull`	Pull a newer image
Custom modules	Possible, awkward	Natural	A mounted folder
Several versions on one machine	No	Yes, with one folder and one port each	Yes, easily
Good for	A quick look; a simple single server	Development, and servers you want to control	Development, tests, and teams that already run containers
Weak point	Hides what it does; harder to adapt	More steps, more to get wrong	One more layer to understand when something fails

The recommendation of this guide
Install from source on Linux at least once, even if you will use Docker afterwards. It is the only method that shows you where everything is, and every error message in section 14 will then make sense.

Windows or Linux?
Odoo runs on both. Odoo's own documentation discourages Windows for production: the multi-process mode that serves many users exists on Linux only. Use Windows for a look and for development, Linux for anything a company depends on.

4 Before you start

Ten minutes of preparation. Most failed installations were missing one of these before the first command was typed.

Have ready	Details
A machine	For a test: 2 processor cores, 4 GB of memory, 20 GB of free disk. For a small company: 4 cores, 8 GB, and fast disks. Odoo's rule of thumb is one worker for about six simultaneous users, and (cores × 2) + 1 workers.
The operating system	Ubuntu 24.04 LTS for the Linux sections. Windows 10 or 11, 64-bit, for the Windows section.
Administrator rights	`sudo` on Linux; an administrator account on Windows.
Internet access from the machine	To GitHub, to the Python package index, to the operating system's package servers, and for Docker to the image registry. If any of these is filtered on your network, arrange a mirror or a proxy first.
The address plan	Which IP address the server will have, whether it is fixed, and which name users will type. Ask the network administrator before, not after.
Passwords	Decide the master password and the database password now and store them in a password manager.
For Enterprise	Access to the Enterprise source code, and the subscription code.

The layout this guide builds on Linux

```
/opt/odoo/                home of the system user "odoo"
├── odoo/                 the Community source code (git)
├── enterprise/           the Enterprise modules, if licensed
├── custom/               your own and third-party modules
└── venv/                 the Python environment with the libraries
/etc/odoo/odoo.conf       the configuration file
/var/lib/odoo/            the data folder: filestore and sessions
/var/log/odoo/odoo.log    the log file
```

Code, configuration, data and logs are kept apart on purpose. Code can be replaced at any time, configuration is small and precious, data must be backed up, and logs may be deleted.

---

5 Linux, from source Linux

Twelve steps on a fresh Ubuntu 24.04 server. Each one has the command, the reason, and what you should see before going on.

1 Update the system and install the tools

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y git curl build-essential python3 python3-venv python3-dev python3-pip \
    libxml2-dev libxslt1-dev libldap2-dev libsasl2-dev libpq-dev libjpeg-dev zlib1g-dev \
    libffi-dev libssl-dev nodejs npm
```

Why Several of Odoo's Python libraries are compiled on your machine when they are installed. The `-dev` packages are the files the compiler needs: `libpq-dev` for the PostgreSQL driver, `libldap2-dev` and `libsasl2-dev` for directory logins, `libxml2-dev` and `libxslt1-dev` for XML. Node.js is there for step 8.

Check `python3 --version` prints 3.10 or higher. Ubuntu 24.04 gives 3.12.

2 Install PostgreSQL

```
sudo apt install -y postgresql postgresql-client
sudo systemctl enable --now postgresql
```

Why The database server must exist before Odoo starts. `enable --now` starts it and makes it start again after every reboot.

Check `sudo -u postgres psql -c "SELECT version();"` prints a PostgreSQL version of 13 or higher.

3 Create the system user

```
sudo useradd --system --create-home --home-dir /opt/odoo --shell /bin/bash odoo
```

Why Odoo should run under its own account with no administrator rights. If someone breaks into Odoo, they get the rights of this account and nothing more. The account's home, `/opt/odoo`, will hold the code.

Check `id odoo` shows the account.

4 Create the database role

```
sudo -u postgres createuser --createdb --no-createrole --no-superuser odoo
```

Why Odoo needs an account inside PostgreSQL. It gets one right only, `--createdb`, because Odoo creates a new PostgreSQL database for each Odoo database. It must not be a superuser: Odoo refuses to start with one. The role has the same name as the system user, so PostgreSQL will accept local connections from that user without a password.

Check `sudo -u postgres psql -c "\du"` lists the role `odoo` with the attribute "Create DB".

5 Download the Odoo source

```
sudo -u odoo git clone --depth 1 --branch 19.0 https://github.com/odoo/odoo.git /opt/odoo/odoo
sudo -u odoo mkdir /opt/odoo/custom
```

Why `--branch 19.0` takes version 19, not the development version. `--depth 1` downloads only the latest state instead of twenty years of history, which saves several gigabytes. `sudo -u odoo` makes the files belong to the Odoo account from the start. The `custom` folder is for modules that are not part of Odoo.

Check `ls /opt/odoo/odoo` shows `odoo-bin`, `addons`, `requirements.txt`.

6 Create a Python environment and install the libraries

```
sudo -u odoo python3 -m venv /opt/odoo/venv
sudo -u odoo /opt/odoo/venv/bin/pip install --upgrade pip wheel
sudo -u odoo /opt/odoo/venv/bin/pip install -r /opt/odoo/odoo/requirements.txt
```

Why A virtual environment is a private copy of Python for Odoo. The sixty libraries go there, not into the system's Python, so they cannot conflict with other software, and recent Ubuntu and Debian versions refuse system-wide `pip install` anyway. From now on Odoo is always started with `/opt/odoo/venv/bin/python3`.

Check The last line reads "Successfully installed ...". An error here names the missing `-dev` package; see section 14.

7 Install wkhtmltopdf

```
cd /tmp
wget https://github.com/wkhtmltopdf/packaging/releases/download/0.12.6.1-3/wkhtmltox_0.12.6.1-3.jammy_amd64.deb
sudo apt install -y ./wkhtmltox_0.12.6.1-3.jammy_amd64.deb
```

Why Odoo prints PDF through this program. The version in Ubuntu's own repository lacks the patched engine, and reports come out without headers and footers. Odoo asks for 0.12.6 from the project's own packages; choose the file that matches your distribution.

Check `wkhtmltopdf --version` prints "0.12.6.1 (with patched qt)".

8 Install right-to-left support

```
sudo npm install -g rtlcss
```

Why When a user switches to Persian or Arabic, Odoo mirrors the whole layout with this tool. Without it the language loads and the screen is broken. Skip this step only if no right-to-left language will ever be used.

9 Create the folders and the configuration file

```
sudo mkdir -p /etc/odoo /var/lib/odoo /var/log/odoo
sudo chown odoo:odoo /var/lib/odoo /var/log/odoo
sudo nano /etc/odoo/odoo.conf
```

Type this minimal configuration. Section 9 explains every line and adds the production settings.

/etc/odoo/odoo.conf

```
[options]
; database: local PostgreSQL, same-name role, no password needed
db_host = False
db_port = False
db_user = odoo
db_password = False

; where things are
addons_path = /opt/odoo/odoo/addons,/opt/odoo/custom
data_dir = /var/lib/odoo

; web
http_port = 8069

; protects the database manager page: choose your own long random value
admin_passwd = change-this-master-password
```

```
sudo chown root:odoo /etc/odoo/odoo.conf
sudo chmod 640 /etc/odoo/odoo.conf
```

Why Putting the settings in a file means the same settings apply whether Odoo is started by hand or by the system. The file holds passwords, so only root may change it and only the Odoo account may read it. In this file a comment must be on a line of its own: text after a value becomes part of the value.

10 Start Odoo by hand, once

```
sudo -u odoo /opt/odoo/venv/bin/python3 /opt/odoo/odoo/odoo-bin -c /etc/odoo/odoo.conf
```

Why Started by hand, Odoo writes its log on the screen, so the first errors are in front of you. A service hides them in a file.

Check A line "HTTP service (werkzeug) running on 0.0.0.0:8069" appears. From another computer, open `http://192.168.1.50:8069` : the page to create a database is shown. Stop Odoo with Ctrl+C.

11 Make Odoo a service

First add one line to `/etc/odoo/odoo.conf`, so that the service writes its log to a file: `logfile = /var/log/odoo/odoo.log`. Then create the service definition:

`/etc/systemd/system/odoo.service`

```
[Unit]
Description=Odoo 19
After=network.target postgresql.service

[Service]
Type=simple
User=odoo
Group=odoo
ExecStart=/opt/odoo/venv/bin/python3 /opt/odoo/odoo/odoo-bin -c /etc/odoo/odoo.conf
Restart=on-failure
KillMode=mixed

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

```
sudo systemctl daemon-reload
sudo systemctl enable --now odoo
sudo systemctl status odoo
```

Why A service starts with the machine, restarts if it crashes, and runs without anyone logged in. `User=odoo` is what keeps it away from root. `After=postgresql.service` makes it wait for the database at boot.

Check The status shows "active (running)". Follow the log with `sudo tail -f /var/log/odoo/odoo.log`.

12 Open the firewall

```
sudo ufw allow OpenSSH
sudo ufw allow 8069/tcp
sudo ufw enable
```

Why A firewall that is off lets everything in; one that is on and not configured lets nothing in. Port 8069 is opened here for the test. Once nginx is in front (section 10), close 8069 again and open 80 and 443 instead.

Adding Enterprise

Enterprise is a second folder of modules, not a different server. Put the Enterprise source in `/opt/odoo/enterprise`, owned by `odoo`, and list it *first* in the configuration: `addons_path = /opt/odoo/enterprise,/opt/odoo/odoo/addons,/opt/odoo/custom`. Restart Odoo. After the database is created, enter the subscription code in the banner on the home screen.

The quicker alternative. Odoo publishes a package for Ubuntu 24.04: adding its repository and running `sudo apt install odoo` creates the user, the database role, the service and `/etc/odoo/odoo.conf` in one go. It is a fair choice for a simple single server. What you gain here by doing it step by step is knowing what each of those things is.

6 Windows Windows

Three routes on Windows: the installer for a quick look, the source for development, and Linux inside Windows for the best of both.

Route A: the installer

1. **Download** the Windows installer for Odoo 19 from Odoo's download page.
2. **Run it.** Windows may show a security prompt: choose More info, then Run anyway. Accept the administrator prompt.
3. **Choose the components.** Keep "Odoo Server" and "PostgreSQL Database" both ticked unless a PostgreSQL server is already installed.
4. **Set the PostgreSQL connection** when asked: host `localhost`, port `5432`, and a user name and password of your choice. The installer creates that database role for you.
5. **Finish.** Odoo is installed as a Windows service that starts with the computer. Open `http://localhost:8069`.

Where is…	On a Windows installer set-up
The configuration file	`odoo.conf` in the `server` folder of the installation, under `C:\Program Files\Odoo 19.0…`
The log	`odoo.log` in the same `server` folder.
Start, stop, restart	The Services panel of Windows (`services.msc`): the Odoo server service.
PDF printing	Install wkhtmltopdf 0.12.6 separately and restart the service.

> **Why use it** It is the fastest way to see Odoo. It is not meant for production, and it hides the steps that this guide wants you to learn.

Route B: from source

1 Install Python

Download Python 3.10 or later, 64-bit, from python.org. On the first screen of the installer tick **Add Python to PATH**, and keep **pip** selected.

> **Why** "PATH" is the list of folders where Windows looks for a program when you type its name. Without that tick, the command `python` is not found.

Check In a new terminal, `python --version` and `pip --version` both answer.

2 Install PostgreSQL and create the database role

Install PostgreSQL 13 or later from postgresql.org. The installer asks for the password of the superuser `postgres`: note it. Then open **pgAdmin**, which is installed with it:

1. Double-click the server and enter the `postgres` password.
2. Menu Object › Create › Login/Group Role.
3. General tab: name `odoo`.
4. Definition tab: a password.
5. Privileges tab: set **Can login?** and **Create database?** to Yes. Save.

Add the PostgreSQL `bin` folder to PATH, by default `C:\Program Files\PostgreSQL\<version>\bin`.

> **Why** The role is the account Odoo logs in to the database with. On Windows the connection always goes over the network interface with a password, so unlike Linux the password is required. The `bin` folder on PATH lets Odoo find the backup tools.

3 Install Git and the C++ build tools

Install Git for Windows. Then install **Build Tools for Visual Studio** and, in its Workloads tab, select **C++ build tools**.

> **Why** Some Python libraries have no ready-made Windows version and are compiled during installation. Without the build tools, `pip` stops with "Microsoft Visual C++ 14.0 or greater is required".

4 Download Odoo and install the libraries

```
cd C:\
mkdir odoo && cd odoo
git clone --depth 1 --branch 19.0 https://github.com/odoo/odoo.git odoo
python -m venv venv
venv\Scripts\activate
pip install setuptools wheel
pip install -r odoo\requirements.txt
```

Why The same reasons as on Linux: version 19 only, a private Python environment, and the libraries listed by Odoo. After `activate` the prompt starts with `(venv)` ; in every new terminal, activate again before running Odoo.

5 Install wkhtmltopdf and rtlcss

Install wkhtmltopdf 0.12.6 for Windows and add its `bin` folder to PATH. For right-to-left languages, install Node.js, then:

```
npm install -g rtlcss
```

and add `C:\Users\<you>\AppData\Roaming\npm` to PATH.

6 Write the configuration and run

`C:\odoo\odoo.conf`

```
[options]
db_host = localhost
db_port = 5432
db_user = odoo
db_password = the-password-from-step-2
addons_path = C:\odoo\odoo\addons,C:\odoo\custom
data_dir = C:\odoo\data
http_port = 8069
admin_passwd = change-this-master-password
```

```
mkdir C:\odoo\custom C:\odoo\data
venv\Scripts\activate
python odoo\odoo-bin -c C:\odoo\odoo.conf
```

Check The terminal shows "HTTP service (werkzeug) running". Open `http://localhost:8069` . Stop with Ctrl+C.

Route C: Linux inside Windows

Windows 10 and 11 can run a real Ubuntu beside Windows, called WSL. In an administrator terminal, `wsl --install` installs it; after a restart you have an Ubuntu terminal in which section 5 works unchanged, and the browser on Windows reaches it at `http://localhost:8069` . For someone who develops on Windows and deploys on Linux, this removes a whole class of differences.

---

7 Docker Docker

Docker packs a program with everything it needs into an image and runs it in a box called a container. For Odoo that means two containers: one for Odoo, one for PostgreSQL.

Word	Meaning, for this guide
Image	A ready-made, read-only package: "Odoo 19 with Python and all libraries". Downloaded once.
Container	A running copy of an image. It can be deleted and recreated at any time, which is why nothing valuable may be stored inside it.
Volume	Storage that lives outside the container and survives it. The database files and the filestore go here.
Compose file	One text file that describes all the containers, their volumes and how they connect.

1 Install Docker

On Linux, install Docker Engine with the Compose plugin from Docker's repository. On Windows, install Docker Desktop, which runs on WSL.

Check `docker --version` and `docker compose version` both answer.

2 Create a project folder

```
mkdir -p ~/odoo19/config ~/odoo19/addons
cd ~/odoo19
```

Why `config` will hold `odoo.conf` and `addons` your custom modules. Both stay on your disk, where you can edit them, and are shown to the container.

3 Write the two files

`~/odoo19/compose.yaml`

```
services:
  db:
    image: postgres:16
    environment:
      POSTGRES_DB: postgres
      POSTGRES_USER: odoo
      POSTGRES_PASSWORD: change-this-db-password
    volumes:
      - odoo-db-data:/var/lib/postgresql/data
    restart: unless-stopped

  web:
    image: odoo:19.0
    depends_on:
      - db
    ports:
      - "8069:8069"
    environment:
      HOST: db
      USER: odoo
      PASSWORD: change-this-db-password
    volumes:
      - odoo-web-data:/var/lib/odoo
      - ./config:/etc/odoo
      - ./addons:/mnt/extra-addons
    restart: unless-stopped

volumes:
  odoo-db-data:
  odoo-web-data:
```

`~/odoo19/config/odoo.conf`

```
[options]
addons_path = /mnt/extra-addons
data_dir = /var/lib/odoo
admin_passwd = change-this-master-password
```

Why each part The `db` service creates the PostgreSQL role `odoo` with the password you give. The `web` service receives the same three values as `HOST`, `USER` and `PASSWORD`: that is the whole connection between Odoo and its database. `HOST: db` works because Compose puts both containers on a private network where each is reachable by its service name. `8069:8069` publishes the container's port on the machine. The two named volumes keep the database and the filestore when containers are recreated.

4 Start, look, stop

```
docker compose up -d            # download the images if needed, start both containers
docker compose ps               # both should be "running"
docker compose logs -f web      # follow Odoo's log; Ctrl+C to leave
docker compose stop             # stop, keeping everything
docker compose down             # remove the containers; volumes and data stay
```

Check Open `http://localhost:8069`, or the machine's IP address from another computer.

Everyday Docker commands for Odoo

Command	Does
`docker compose restart web`	Restarts Odoo, for example after changing `odoo.conf` or a module's Python.
`docker compose run --rm web odoo -d sahand -u my_module --stop-after-init`	Updates a module in database `sahand`, then exits.
`docker compose exec web bash`	Opens a shell inside the Odoo container.
`docker compose exec db psql -U odoo -d postgres`	Opens a PostgreSQL prompt.
`docker compose exec -T db pg_dump -U odoo -Fc sahand > sahand.dump`	Dumps the database to a file on your disk.
`docker compose pull && docker compose up -d`	Fetches the latest build of the same version and recreates the containers.

Four things that surprise people with Docker

Data outside volumes is lost. Anything written inside a container disappears when it is recreated. **"localhost" means the container itself.** From inside the Odoo container the database is at `db`, not at localhost. **Pin the version.** `odoo:19.0` stays on version 19; `odoo:latest` will one day become version 20 and refuse your database. **Do not use** `down -v` unless you mean to delete the database: `-v` removes the volumes.

8 Odoo and PostgreSQL: how they connect

Odoo keeps nothing in itself. Every time it starts and on every request it opens a connection to PostgreSQL, and four settings decide where and as whom.

Setting	Meaning	Typical values
`db_host`	Where the database server is.	`False`: this machine, through a local socket. `localhost`: this machine, over the network interface. An IP address: another machine.
`db_port`	The port PostgreSQL listens on.	`False` or `5432`
`db_user`	The PostgreSQL role to log in as.	`odoo`. Never `postgres`.
`db_password`	That role's password.	`False` for a local socket; the password otherwise.

Two ways to reach a local database Linux

Through the local socket: no password

`db_host = False`. Odoo and PostgreSQL talk through a file on the same machine. PostgreSQL asks the operating system which account is connecting, and accepts it if a role of the *same name* exists. This is called **peer authentication**.

It is why steps 3 and 4 of section 5 used the same name, and why no password is written anywhere. It is the simplest and safest set-up for one machine.

Through the network interface: with a password

`db_host = localhost`. The connection goes through TCP port 5432 even though both are on one machine, and PostgreSQL asks for the role's password.

This is the only way on Windows and in Docker, and the way to reach a database on another machine. The role needs a password first:

```
sudo -u postgres psql -c "ALTER ROLE odoo WITH PASSWORD 'change-this-db-password';"
```

Test the connection without Odoo

When Odoo cannot connect, take Odoo out of the picture. If these commands fail, the problem is in PostgreSQL or the network, and no Odoo setting will fix it.

```
# local socket, as the system user odoo
sudo -u odoo psql -d postgres -c "SELECT current_user;"

# over TCP, with the password; works from another machine too
psql -h 192.168.1.60 -p 5432 -U odoo -d postgres -c "SELECT current_user;"
```

The database on another machine

Larger installations put PostgreSQL on its own server. Suppose Odoo is on 192.168.1.50 and PostgreSQL on 192.168.1.60. Three things must change on the database server, and one on the Odoo server.

1 Make PostgreSQL listen on the network

`/etc/postgresql/16/main/postgresql.conf`

```
listen_addresses = 'localhost,192.168.1.60'
```

Why A fresh PostgreSQL listens only on the machine itself. This line tells it to accept connections arriving on its network address too.

2 Allow the Odoo server to log in

`/etc/postgresql/16/main/pg_hba.conf`

```
# TYPE  DATABASE  USER  ADDRESS            METHOD
host    all       odoo  192.168.1.50/32    scram-sha-256
```

Why `pg_hba.conf` is PostgreSQL's guest list: which role, from which address, proving its identity how. This line admits the role `odoo`, only from the Odoo server, with a password. Without a matching line PostgreSQL answers "no pg_hba.conf entry for host". On PostgreSQL 13, write `md5` as the method.

3 Restart PostgreSQL and open its firewall to the Odoo server only

```
sudo systemctl restart postgresql
sudo ufw allow from 192.168.1.50 to any port 5432 proto tcp
```

4 Point Odoo at it

`/etc/odoo/odoo.conf, on the Odoo server`

```
db_host = 192.168.1.60
db_port = 5432
db_user = odoo
db_password = change-this-db-password
```

Check Run the TCP test above from the Odoo server before restarting Odoo.

What Odoo does with the connection

Fact	Consequence
One Odoo database is one PostgreSQL database.	Creating a database in Odoo runs `CREATE DATABASE`, which is why the role needs the "Create DB" right.
Odoo refuses the role `postgres`.	It stops with "Using the database user 'postgres' is a security risk". A superuser could read and destroy every database on the server.
Odoo lists the databases its role owns.	A database restored by another role does not appear. Restore as `odoo`, or change the owner.
Each worker keeps several connections open.	With many workers, raise `max_connections` in PostgreSQL, or lower `db_maxconn` in Odoo.
Attachments are not in the database.	They are files under `data_dir`. A database without its filestore is half a backup.

9 The configuration file

Everything that can be given on the command line can be written in `odoo.conf`. A server should have all of it there, so that how it runs is written down in one place.

A complete file for a small production server Linux

`/etc/odoo/odoo.conf`

```
[options]
; ---------- database ----------
db_host = False
db_port = False
db_user = odoo
db_password = False

; ---------- paths ----------
addons_path = /opt/odoo/odoo/addons,/opt/odoo/custom
data_dir = /var/lib/odoo
logfile = /var/log/odoo/odoo.log
log_level = info

; ---------- web ----------
; listen only on this machine: nginx is the one that talks to the outside
http_interface = 127.0.0.1
http_port = 8069
gevent_port = 8072
proxy_mode = True

; ---------- security ----------
admin_passwd = change-this-master-password
list_db = False
dbfilter = ^sahand$

; ---------- performance: 2 cores, 4 GB ----------
workers = 5
max_cron_threads = 1
limit_memory_soft = 1073741824
limit_memory_hard = 1610612736
limit_time_cpu = 600
limit_time_real = 1200
```

Every line explained

Option	If you leave it out	What it does, and why you set it
`addons_path`	Only Odoo's own `addons`	Folders, separated by commas, where modules are looked for. Each folder must directly contain module folders. A module outside these paths does not exist for Odoo.
`data_dir`	A hidden folder in the user's home	Filestore (attachments, generated styles and scripts) and sessions. Set it explicitly so that you know what to back up.
`logfile`	The log goes to the screen	A file for the log. Needed for a service.
`log_level`	`info`	`debug` for more detail while hunting a problem, `warn` for less.
`http_interface`	`0.0.0.0` : every address of the machine	The address Odoo listens on. `127.0.0.1` makes it reachable only from the machine itself, which is right when nginx is in front.
`http_port`	8069	The port for normal requests. Change it to run a second Odoo on the same machine.
`gevent_port`	8072	The port for live notifications, used when `workers` is above zero.
`proxy_mode`	Off	Tells Odoo to believe what the web server in front says about the visitor's real address and about HTTPS. Switch it on only when there is such a web server.
`admin_passwd`	`admin`	The master password of the database manager. The default is known to everyone: change it before the server is reachable.
`list_db`	On	Off hides the list of databases and the database manager from the web.
`dbfilter`	All databases	A pattern for the database names this server may show. `^sahand$` means exactly "sahand". `^%d$` means "the database named like the first part of the web address".

Option	If you leave it out	What it does, and why you set it
`workers`	0: one process, threads	The number of processes serving requests. Zero is for development. Production uses (cores × 2) + 1. Linux only.
`max_cron_threads`	2	Processes for scheduled jobs.
`limit_memory_soft` , `_hard`	About 2 GB and 2.5 GB	In bytes. A worker above the soft limit is restarted after its request; above the hard limit it is stopped at once. They keep one runaway request from taking the server down.
`limit_time_cpu` , `_real`	60 and 120 seconds	How long one request may run. Raise them if large imports or reports are cut off.

Rules of the file
The first line must be `[options]` . One setting per line, `name = value` . Comments start with `;` or `#` and stand on a line of their own. Option names use underscores where the command line uses hyphens: `--http-port` becomes `http_port` . After any change, restart Odoo.

File or command line?
A value on the command line wins over the file. That is handy for one-off jobs:
`odoo-bin -c /etc/odoo/odoo.conf -d sahand -u sale --stop`
uses the file for everything and adds the update. Adding `-s` writes the current options back into the file.

10 IP address, port, DNS and HTTPS

Odoo is running. Now other people have to reach it. That is four questions: at which address, on which port, under which name, and protected how.

IP addresses: which one do users type?

Address	What it is	Reachable from
`127.0.0.1` , `localhost`	The machine talking to itself.	That machine only. Useless to anyone else.
A private address, such as `192.168.1.50`	The machine's address inside the office network.	Computers on the same network, or connected by VPN.
A public address	An address on the internet, given by the provider or the data centre.	Anywhere, if the firewall allows.
`0.0.0.0`	Not an address to visit. In a setting it means "listen on every address this machine has".	

```
hostname -I                 # Linux: the machine's addresses
ip -4 addr show             # Linux: the same, in detail
ipconfig                    # Windows
```

A server needs an address that does not change. Ask the network administrator for a fixed address or a reservation; if the address changes after a restart of the router, every bookmark and every DNS record breaks.

Ports: the door, and who may use it

An address finds the machine; a port finds the program on it. Odoo's default is 8069, set by `http_port` . Three things decide whether a browser gets through:

	Check	How
1	Is Odoo listening on that port?	`sudo ss -ltnp | grep 8069` shows the Python process.
2	On which address?	The same line shows `0.0.0.0:8069` (all addresses) or `127.0.0.1:8069` (this machine only). That is `http_interface` .
3	Does the firewall let it in?	Linux: `sudo ufw status` . Windows: an inbound rule for the port. In a cloud: the provider's security group as well.

```
REM Windows: allow port 8069 through the firewall (administrator terminal)
netsh advfirewall firewall add rule name="Odoo 8069" dir=in action=allow protocol=TCP localport=8069
```

Why not simply run Odoo on port 80, so that users need no port number? Ports below 1024 are reserved for programs started by root, and Odoo should not run as root. The clean answer is a web server on 80 and 443 that passes requests to Odoo on 8069.

Three ways to reach the same server

Stage	Users type	What is needed
Testing, on the server itself	`http://localhost:8069`	Nothing more.
Inside the office	`http://192.168.1.50:8069`	`http_interface` not limited to 127.0.0.1; port 8069 open in the firewall.
For real use	`https://erp.sahand.example`	A DNS name, nginx on ports 80 and 443, a certificate, `proxy_mode` .

DNS: a name instead of a number

DNS is the phone book that turns `erp.sahand.example` into `192.168.1.50` . Odoo does not manage it; you create one record where the domain's names are managed.

Situation	What to create, and where
Reachable from the internet	An **A record** at the domain's registrar or DNS provider: name `erp` , value the server's public address. Allow some minutes to hours for it to spread.
Office network only	The same A record on the company's internal DNS server or router, pointing at the private address.
Just my own computer, for a test	One line in the hosts file: `192.168.1.50 erp.sahand.example` . The file is `/etc/hosts` on Linux and `C:\Windows\System32\drivers\etc\hosts` on Windows.

```
nslookup erp.sahand.example        # which address does the name give? Works on Linux and Windows
ping erp.sahand.example            # does that address answer?
```

nginx in front of Odoo Linux

A web server in front of Odoo does four jobs: it speaks HTTPS, it listens on the standard ports so nobody types 8069, it sends live notifications to the right Odoo port, and it hides Odoo from direct access.

```
sudo apt install -y nginx
sudo nano /etc/nginx/sites-available/odoo
```

/etc/nginx/sites-available/odoo

```
upstream odoo     { server 127.0.0.1:8069; }
upstream odoochat { server 127.0.0.1:8072; }
map $http_upgrade $connection_upgrade { default upgrade; '' close; }

server {
    listen 80;
    server_name erp.sahand.example;

    proxy_read_timeout 720s;
    proxy_connect_timeout 720s;
    proxy_send_timeout 720s;
    client_max_body_size 200m;

    # live notifications go to the gevent port
    location /websocket {
        proxy_pass http://odoochat;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection $connection_upgrade;
        proxy_set_header X-Forwarded-Host $http_host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Real-IP $remote_addr;
    }

    # everything else goes to Odoo
    location / {
        proxy_pass http://odoo;
        proxy_set_header X-Forwarded-Host $http_host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_redirect off;
    }

    gzip on;
    gzip_types text/css text/plain text/xml application/xml application/json application/javascript;
}
```

```
sudo ln -s /etc/nginx/sites-available/odoo /etc/nginx/sites-enabled/odoo
sudo nginx -t                       # checks the file for mistakes
sudo systemctl reload nginx
sudo ufw allow 'Nginx Full'         # ports 80 and 443
sudo ufw delete allow 8069/tcp      # Odoo is no longer reached directly
```

Then set `proxy_mode = True`, `http_interface = 127.0.0.1` and `workers` above zero in `odoo.conf`, as in section 9, and restart Odoo.

Adding HTTPS

```
sudo apt install -y certbot python3-certbot-nginx
sudo certbot --nginx -d erp.sahand.example
```

Why Without HTTPS, passwords and business data cross the network readable by anyone on the way. Certbot obtains a free certificate, rewrites the nginx file to listen on 443 and to redirect port 80 to it, and renews the certificate by itself. It needs the name to be reachable from the internet; for a server that is internal only, use a certificate from the company's own authority and add the `listen 443 ssl` lines by hand.

Why `proxy_mode` matters

Behind nginx, Odoo sees every request arriving from 127.0.0.1 over plain HTTP. With `proxy_mode = True` it reads the `X-Forwarded` headers that nginx adds and learns the visitor's real address and that the visitor used HTTPS. Without it, links in e-mails start with `http://`, logins may loop, and the log shows the same address for everybody. Without a proxy in front it must stay off, or visitors could fake those headers.

The address Odoo puts in e-mails. Odoo stores its own web address in a system parameter, `web.base.url`, and uses it in every link it sends. It is set when an administrator logs in, from the address in their browser. Log in once through the final name, `https://erp.sahand.example`, and check the value under Settings › Technical › System Parameters.

11 First start: the database and the first users

The server is installed, and it is still empty. An Odoo installation becomes usable when its first database exists and someone other than the administrator can log in.

Creating the first database

In the browser

On a server with no database, the first visit shows a form:

Master Password	The `admin_passwd` of `odoo.conf`.
Database Name	`sahand`. Lower case, no spaces.
Email	Becomes the administrator's login.
Password	The administrator's password.
Language, Country	The country decides the proposed accounting set-up.
Demo data	Tick only for a database you will throw away.

From the command line

Needed when `list_db = False` hides that form, and useful in scripts:

```
sudo systemctl stop odoo
sudo -u odoo /opt/odoo/venv/bin/python3 \
  /opt/odoo/odoo/odoo-bin -c /etc/odoo/odoo.conf \
  -d sahand -i base --stop-after-init
sudo systemctl start odoo
```

The database is created with the login `admin` and the password `admin`. Change that password in the first minute.

Why the name matters The database name is what `dbfilter` matches, what backup files are called, and the name of the filestore folder. Choose it once; renaming later is possible and tedious.

The first ten minutes in the new database

	Do	Where	Why now
1	Change the administrator's password	User menu, top right › Preferences › Account Security	`admin` / `admin` is the first thing anyone tries.
2	Fill in the company	Settings › Companies › Update Info	Name, address and currency appear on every document.
3	Install the apps	Apps	Only those the project needs.
4	Add the Persian language, if used	Settings › Languages › Add Languages	This is where a missing `rtlcss` shows up.
5	Print any report	For example a quotation	This is where a missing or wrong wkhtmltopdf shows up.
6	Set the outgoing mail server	Settings › Emails	Invitations and password resets need it.
7	Register Enterprise, if used	The banner on the home screen	An unregistered Enterprise database expires.

Creating the people who will log in

1. **Open** Settings › Users › Manage Users and press New.
2. **Enter** the name and the e-mail address. The e-mail address is the login.
3. **Choose the role** in each app on the Access Rights tab, and save.
4. **Give them a password.** If outgoing mail works, Odoo sends an invitation with a link. If it does not yet, open the user, choose the gear menu › Change Password, and pass the password on yourself.

These are Odoo users, the fourth identity of section 2. They exist only in this database: a second database on the same server has its own users.

12 Day-two operations

Installing is done once. Starting, stopping, updating, backing up and reading the log are done for years.

The commands Linux

Command	Does
`sudo systemctl status odoo`	Is it running? Since when? The last log lines.
`sudo systemctl restart odoo`	Stop and start. Needed after changing `odoo.conf` or any Python file.
`sudo systemctl stop odoo` / `start odoo`	Stop; start.
`sudo tail -f /var/log/odoo/odoo.log`	Follow the log live. Ctrl+C to leave.
`sudo grep -n "ERROR|CRITICAL" /var/log/odoo/odoo.log | tail`	The most recent errors.
`sudo ss -ltnp | grep -E "8069|8072|5432"`	Which of the ports are being listened on, and by what.

Reading a log line

```
2026-10-07 09:14:02,481 4242 ERROR sahand odoo.sql_db: Connection to the database failed
└── date and time ─────┘ └pid┘ └level┘ └db──┘ └── which part of Odoo ┘ └── the message
```

The level is the first thing to read: INFO is normal life, WARNING deserves a look, ERROR and CRITICAL are problems. A Python error is followed by a **traceback**, a list of files and line numbers; the useful part is nearly always its *last* line.

Routine jobs

A Add a custom module

```
sudo cp -r my_module /opt/odoo/custom/
sudo chown -R odoo:odoo /opt/odoo/custom/my_module
sudo systemctl restart odoo
```

Then, in developer mode, Apps › Update Apps List, search for the module and activate it.

> **Why** Odoo reads the list of available modules only when asked. The restart loads the Python code; "Update Apps List" makes the new folder appear.

B Update a module after its files changed

```
sudo systemctl stop odoo
sudo -u odoo /opt/odoo/venv/bin/python3 /opt/odoo/odoo/odoo-bin -c /etc/odoo/odoo.conf \
    -d sahand -u my_module --stop-after-init
sudo systemctl start odoo
```

> **Why** Views, menus and data are stored in the database. Changing the XML file changes nothing until `-u` reloads it.

C Bring Odoo itself up to date, within version 19

```
sudo systemctl stop odoo
sudo -u odoo git -C /opt/odoo/odoo pull
sudo -u odoo /opt/odoo/venv/bin/pip install -r /opt/odoo/odoo/requirements.txt
sudo -u odoo /opt/odoo/venv/bin/python3 /opt/odoo/odoo/odoo-bin -c /etc/odoo/odoo.conf \
    -d sahand -u all --stop-after-init
sudo systemctl start odoo
```

> **Why** Odoo publishes fixes for version 19 continuously. Back up first, try it on a copy, and do it at a quiet hour: `-u all` can take many minutes.

D Back up: the database *and* the filestore

/opt/odoo/backup.sh

```
#!/bin/bash
DB=sahand
STAMP=$(date +%F)
DEST=/var/backups/odoo

pg_dump -Fc "$DB" > "$DEST/${DB}_${STAMP}.dump"
tar -czf "$DEST/${DB}_filestore_${STAMP}.tar.gz" -C /var/lib/odoo/filestore "$DB"
find "$DEST" -type f -mtime +14 -delete
```

```
sudo mkdir -p /var/backups/odoo && sudo chown odoo:odoo /var/backups/odoo
sudo chmod +x /opt/odoo/backup.sh
sudo crontab -u odoo -e
# add this line: every night at 02:00
0 2 * * * /opt/odoo/backup.sh
```

Why The dump holds the records; the filestore holds every attachment and image. Restoring one without the other gives a database with broken pages. A backup that stays on the same disk protects against mistakes, not against the disk: copy the files to another machine as well.

E Restore, and prove the backup works

```
sudo -u odoo createdb sahand_test
sudo -u odoo pg_restore --no-owner -d sahand_test /var/backups/odoo/sahand_2026-10-07.dump
sudo -u odoo mkdir -p /var/lib/odoo/filestore/sahand_test
sudo -u odoo tar -xzf /var/backups/odoo/sahand_filestore_2026-10-07.tar.gz \
    -C /var/lib/odoo/filestore/sahand_test --strip-components=1
```

Why A backup that was never restored is a hope, not a backup. Restore into a test name once a month. Remember that `dbfilter` must allow the new name, and that a copy of production still knows the real mail server: disable outgoing mail in the copy before anyone uses it.

13 Securing a server

A test machine in the office can be relaxed. A server that holds a company's accounts and can be reached from outside cannot. This is the minimum.

	Do this	How	Because
1	Set a long, random master password.	`admin_passwd`	The default lets anyone download or delete every database.
2	Hide the database manager.	`list_db = False`	Nobody outside needs to see the list of databases.
3	Restrict the databases served.	`dbfilter = ^sahand$`	A test database on the same server must not be reachable through the public name.
4	Change the administrator's login and password.	In the database	`admin` / `admin` is tried by every scanner.
5	Never load demo data.	Leave the box unticked	Demo data brings demo users with known passwords.
6	Run Odoo as its own unprivileged user.	`User=odoo` in the service	A fault in Odoo must not become control of the machine.
7	Use a database role that is not a superuser.	`createuser --no-superuser`	The same reasoning, for the database server.
8	Put HTTPS in front.	nginx and a certificate	Passwords and data must not travel in clear text.
9	Close every port that is not needed.	Firewall: 22, 80, 443 only	8069, 8072 and 5432 are for the machine itself.
10	Keep PostgreSQL off the internet.	`listen_addresses` , `pg_hba.conf`	The database should accept Odoo and nobody else.
11	Use several workers with limits.	`workers` , `limit_*`	One heavy request cannot block or exhaust the server.
12	Protect SSH.	Keys, not passwords	The server is only as safe as its administrator's login.
13	Apply updates.	The system's and Odoo's	Security fixes are published all year.
14	Back up daily to another machine, and restore regularly.	Section 12	It is the answer to disk failure, mistakes and ransomware alike.

	Do this	How	Because
15	Keep test and production apart.	Different machines or at least different databases and names	A test must never send a real e-mail or touch real data.

14 Errors and fixes

Nearly every installation hits two or three of these. First the method for finding where the problem is, then the messages, grouped by the moment they appear.

Find the broken arrow

Test the chain of section 1 from the inside outwards. The first test that fails is where the problem is; everything further out fails only as a consequence.

	Question	Test, on the server	If it fails, look at
1	Is PostgreSQL running?	`sudo systemctl status postgresql`	The PostgreSQL log
2	Can the role connect?	`sudo -u odoo psql -d postgres -c "select 1"`	Section 8
3	Is Odoo running?	`sudo systemctl status odoo`	The end of `odoo.log`
4	Is it listening?	`sudo ss -ltnp | grep 8069`	`http_port`, `http_interface`
5	Does it answer locally?	`curl -I http://127.0.0.1:8069/web/login`	The log, at the time of the request
6	Does it answer from another computer?	`curl -I http://192.168.1.50:8069/web/login`	The firewall; `http_interface`
7	Does the name lead to the server?	`nslookup erp.sahand.example`	The DNS record
8	Does nginx pass it on?	`sudo nginx -t`, then the nginx error log	The nginx site file

While installing the libraries

Message	Cause	Fix
`error: externally-managed-environment`	pip was run against the system's Python, which recent Ubuntu and Debian forbid.	Use the virtual environment: `/opt/odoo/venv/bin/pip`.
`fatal error: Python.h: No such file or directory`	The Python headers are missing.	`sudo apt install python3-dev`
`Error: pg_config executable not found` `fatal error: libpq-fe.h`	The PostgreSQL development files are missing.	`sudo apt install libpq-dev`
`fatal error: lber.h / sasl.h: No such file`	The LDAP and SASL development files are missing.	`sudo apt install libldap2-dev libsasl2-dev`
`error: Microsoft Visual C++ 14.0 or greater is required`	Windows has no compiler.	Install Build Tools for Visual Studio with the C++ workload.
`'python' is not recognized as an internal or external command`	Python is not on PATH.	Reinstall with "Add Python to PATH", or add the folder by hand; open a new terminal.
`Read timed out / Connection reset` during git clone, pip install, npm install or docker pull	The download site is slow, filtered or unreachable from your network.	Retry; use `--depth 1` for git; configure the mirror or proxy your organisation provides.

When Odoo starts

Message	Cause	Fix
`ModuleNotFoundError: No module named 'psycopg2'` or another library	Odoo was started with a Python that does not have the libraries.	Start it with the environment's Python: `/opt/odoo/venv/bin/python3`. On Windows, activate the environment first.
`FATAL: role "odoo" does not exist`	The database role was never created, or has another name.	Section 5, step 4. Check `db_user`.
`FATAL: Peer authentication failed for user "odoo"`	A local-socket connection made by a system user with a different name, often root or your own account.	Start Odoo as the system user `odoo`, or connect with `db_host = localhost` and a password.

Message	Cause	Fix
`FATAL: password authentication failed for user "odoo"`	Wrong `db_password`, or the role has no password.	`ALTER ROLE odoo WITH PASSWORD '…'`, and the same value in `odoo.conf`.
`connection to server at "localhost", port 5432 failed: Connection refused`	PostgreSQL is not running, or listens on another address or port.	Start PostgreSQL. Check `db_host`, `db_port`, and `listen_addresses`.
`FATAL: no pg_hba.conf entry for host "192.168.1.50", user "odoo"`	PostgreSQL has no rule admitting this address.	Section 8: add the line to `pg_hba.conf` and restart PostgreSQL.
`Using the database user 'postgres' is a security risk, aborting.`	`db_user` is the superuser.	Create and use the role `odoo`.
`permission denied to create database`	The role lacks the right to create databases.	`ALTER ROLE odoo CREATEDB;`
`OSError: [Errno 98] Address already in use`	Another program, usually another Odoo, already uses the port.	Find it with `sudo ss -ltnp | grep 8069`. Stop it, or choose another `http_port`.
`PermissionError: [Errno 13] Permission denied: '/var/lib/odoo/…'` or the log file	The folder belongs to another account.	`sudo chown -R odoo:odoo /var/lib/odoo /var/log/odoo`
`invalid literal for int() … in odoo.conf` or a setting that seems ignored	A comment was written on the same line as a value, or the file does not start with `[options]`.	Put comments on their own lines. Check the first line.
`the path … is not a valid addons directory`	A folder in `addons_path` does not exist or does not directly contain modules.	Correct the path. Point at the folder that *contains* the module folders.

In the browser

Symptom	Cause	Fix
The page does not open from another computer; it works on the server.	The firewall blocks the port, or Odoo listens on 127.0.0.1 only.	Open the port. Check `http_interface`. In a cloud, check the provider's firewall too.
The name does not open; the IP address does.	The DNS record is missing, wrong, or not yet spread.	`nslookup` the name. Correct the A record; wait; or use the hosts file for a test.
`502 Bad Gateway` from nginx	nginx is running and Odoo is not, or it listens elsewhere.	Start Odoo. Compare the ports in the nginx file with `odoo.conf`.
`Internal Server Error` right after choosing a database	The database exists in PostgreSQL but was never initialised by Odoo, or belongs to another version.	Read the log. Initialise with `-d name -i base`, or restore a proper backup.
The page opens without colours or layout, as plain text.	The generated styles are missing: the filestore was not restored with the database, or `data_dir` is not writable.	Restore the filestore; fix the folder's owner. In developer mode, the bug menu › Regenerate Assets.
Persian is selected and the layout is broken or not mirrored.	`rtlcss` is missing or not on PATH.	`sudo npm install -g rtlcss`, restart Odoo, regenerate the assets.
`Unable to find Wkhtmltopdf on this system. The PDF can not be created.`	wkhtmltopdf is not installed, or not on PATH of the Odoo service.	Section 5, step 7. Restart Odoo.
PDFs print without header, footer or logo.	The wkhtmltopdf build lacks the patched engine, or Odoo cannot reach its own address to fetch the images.	Install 0.12.6 "with patched qt". Check `web.base.url`.
`Access Denied` in the database manager	The master password typed is not `admin_passwd`.	Read the value in `odoo.conf`; restart after changing it.
The list of databases shows too many, too few, or a login page for the wrong one.	`dbfilter` does not match, or the database belongs to another role.	Correct the pattern. Check the owner with `psql -l`.
"Connection lost" keeps appearing; chat and notifications do not arrive.	With workers, live notifications use port 8072, and nginx does not send `/websocket` there.	The `location /websocket` block of section 10.
My module is not in the Apps list.	Its folder is not in `addons_path`, it has no `__manifest__.py`, Odoo cannot read it, or the list was not refreshed.	Check path and owner; restart; Update Apps List; remove the "Apps" filter in the search.
Links in e-mails start with http://localhost:8069.	`web.base.url` was set while browsing locally; `proxy_mode` is off.	Section 10: log in through the final address; switch on `proxy_mode`.

Under load, and in Docker

Message	Cause	Fix
`Worker (pid) exceeded memory limit / CPU time limit exceeded`	One request needed more than the limits allow: a large import, a heavy report.	Raise `limit_memory_*` or `limit_time_*`; or split the job.
`FATAL: sorry, too many clients already`	More connections than PostgreSQL's `max_connections`.	Raise it in `postgresql.conf`, or lower `db_maxconn` or `workers`.
`could not translate host name "db" to address`	The Odoo container cannot find the database container.	Use the service name of the database in `HOST`, and start both with the same compose file.
`Database connection failure` when the containers start	Odoo started before PostgreSQL was ready, or the passwords differ.	Wait and check `docker compose logs db`. Make `PASSWORD` equal to `POSTGRES_PASSWORD`.
`Permission denied` on /var/lib/odoo or /mnt/extra-addons in the container	The mounted folder is not readable by the user inside the container.	Use a named volume for data; make the addons folder readable by everyone.
The password in the compose file was changed and Odoo no longer connects.	PostgreSQL sets its password only when its volume is first created.	Change the role's password inside PostgreSQL with `ALTER ROLE`, or recreate the volume if it holds nothing.
All data gone after "docker compose down".	No volumes were defined, or `-v` was used.	Define named volumes as in section 7. Restore from backup.

The exact wording of a message varies a little between versions. Search the log for its distinctive words.

15 Cheat sheet and checks

Where things are in each kind of installation, the test that says an installation is finished, and six questions.

Where is it?

	Linux from source, this guide	Windows installer	Docker
Configuration	`/etc/odoo/odoo.conf`	`server\odoo.conf` in the install folder	`./config/odoo.conf`, seen as `/etc/odoo`
Source code	`/opt/odoo/odoo`	`server\odoo`	Inside the image
Custom modules	`/opt/odoo/custom`	A folder you add to `addons_path`	`./addons`, seen as `/mnt/extra-addons`
Data folder	`/var/lib/odoo`	Under the user's AppData, unless set	Volume, seen as `/var/lib/odoo`
Log	`/var/log/odoo/odoo.log`	`server\odoo.log`	`docker compose logs web`
Start and stop	`systemctl … odoo`	Windows Services	`docker compose up -d` / `stop`
Database server	Local socket	`localhost:5432`	Service `db`

The acceptance test

An installation is finished when all ten succeed, not when the login page appears.

	Test	Proves
1	Restart the machine; the login page comes back by itself.	PostgreSQL and Odoo are services that start at boot.
2	Log in from another computer, by name.	Address, port, firewall and DNS.
3	The address bar shows a closed padlock.	HTTPS and the certificate.
4	Install an app.	The database role can create tables; the addons path is right.
5	Attach a file to a record, log out, log in, open it.	The data folder is writable.
6	Print a quotation; it has the logo, header and footer.	wkhtmltopdf, and Odoo can reach its own address.
7	Switch a user to Persian; the screen is mirrored.	rtlcss.
8	Send a chat message between two users; it arrives without reloading.	The WebSocket route through nginx.
9	The database manager page is not reachable from outside.	`list_db`, and the master password is not the default.

	Test	Proves
10	Run the backup, restore it under another name, log in there.	The backup is real.

Check yourself

	Question	Answer
1	Name the four "users" of an installation.	The system user, the database role, the master password, and the Odoo users inside a database.
2	On Linux, Odoo connects to a local PostgreSQL with no password in the file. How?	Through the local socket with peer authentication: the system user and the role have the same name.
3	The page works at localhost:8069 on the server and not from your laptop. Two likely causes?	The firewall blocks 8069, or `http_interface` is 127.0.0.1.
4	Why is nginx put in front instead of running Odoo on port 443?	For HTTPS, the standard ports without root rights, and routing of WebSocket traffic.
5	What does `proxy_mode = True` change?	Odoo trusts the forwarded headers, so it knows the visitor's real address and that HTTPS was used.
6	A restored database opens without any styling. What was forgotten?	The filestore.

> **The whole guide in two lines.** Install the pieces in order, give each its own account, write every setting in one file, and put a web server in front. When something fails, test the chain from the database outwards and read the last lines of the log.