GEBO.AI SETUP GUIDE · STEP 5 OF 7

Connect the company folder and publish it

Let Gebo.ai read a company folder, create the data source on it and publish it: Gebo.ai reads and indexes the documents.

TIME	5 minutes, plus the indexing time of your documents.
YOU NEED	The addresses of the company shares and an account that can read them. With Docker, also the time zone of your office.

1 On the server: give Gebo.ai the company shares

How Gebo.ai reaches the company shares depends on how it is installed: with Docker Compose (step 1), as a service on a Linux virtual machine, or on a Windows server. Follow the option that matches your installation.

Option 1: Gebo.ai in Docker Compose

Gebo.ai reads the company documents from `/opt/gebo.ai/shares` inside its container. Mount each company share, of a Windows file server or a NAS, in its own subfolder there, as a Docker volume of type `cifs` or `nfs`. Docker mounts it like an fstab line, with the options you write, and Gebo.ai sees it read only.

Write the lines in a file named `docker-compose.override.yml` in the gebo-ai folder of step 1, next to `docker-compose.yml`. Docker Compose reads both files, so `docker-compose.yml` can be downloaded again on upgrades without losing your shares. The first volume line keeps the volume of step 1 on `/opt/gebo.ai/shares` but without `:ro`: Docker creates the subfolders in it, which a read-only volume does not allow.

```
services:
  gebo.ai:
    environment:
      TZ: "Europe/Rome"
    volumes:
      - "gebo-shares:/opt/gebo.ai/shares"
      - "finance:/opt/gebo.ai/shares/finance:ro"
      - "hr:/opt/gebo.ai/shares/hr:ro"
```

```
volumes:
  finance:
    driver: local
    driver_opts:
      type: cifs
      device: "//files/Finance"
      o: "addr=files,user=${SMB_USER},pass=${SMB_PASSWORD},iocharset=utf8,ro"
  hr:
    driver: local
    driver_opts:
      type: nfs
      device: ":/export/hr"
      o: "addr=nas,nfsvers=4,ro"
```

Write the account that reads the Windows shares in a file named `.env` in the same folder, readable only by you (`chmod 600 .env`). Docker Compose fills `${SMB_USER}` and `${SMB_PASSWORD}` from it:

```
SMB_USER=gebo-reader
SMB_PASSWORD=...
```

Restart Gebo.ai with the shares (the databases keep running) and list what it sees:

```
docker compose up -d
docker compose exec gebo.ai ls /opt/gebo.ai/shares
```

Changing a share later

Docker keeps the options a volume was first created with. After changing them, run `docker compose down`, remove the volume, for example `docker volume rm gebo-monolith_finance`, then `docker compose up -d`. This removes only the mount: the files stay on the file server.

Set the time zone

The `TZ` line sets the time zone of the scheduled sync of step 6. Without it the container runs on UTC, and an evening sync at 20:00 would start at 21:00 or 22:00 Italian time.

Option 2: Gebo.ai as a service on a Linux virtual machine

The Linux appliance of gebo.ai/downloads and the Ubuntu installation of the technical configuration manual (section *Linux Ubuntu and Debian style installation*) both install Gebo.ai from the `.deb` package, which runs it as the `gebo-ai` service under the `gebo-ai` user.

Mount each Windows share on the virtual machine through `/etc/fstab`, read only and owned by that user, in one parent folder, for example `/mnt/company`:

```
sudo apt install cifs-utils language-pack-en
sudo locale-gen en_US.UTF-8
sudo update-locale LANG=en_US.UTF-8
sudo mkdir -p /mnt/company/finance
# /etc/gebo-share, chmod 600: username=..., password=..., domain=...
sudo nano /etc/fstab
```

Add one line per share to `/etc/fstab`, then mount it and check that the service user can read it:

```
//files/Finance /mnt/company/finance cifs ro,credentials=/etc/gebo-share,uid=gebo-
ai,gid=gebo-ai,iocharset=utf8,_netdev 0 0
```

```
sudo mount -a
sudo -u gebo-ai ls /mnt/company/finance
```

With NFS the line is `files:/export/finance /mnt/company/finance nfs ro,_netdev 0 0`; the files must then be readable by the `gebo-ai` user on the NFS server. Finally add the parent folder `/mnt/company` in Gebo.ai, as described in option 3.

Option 3: Gebo.ai on a Windows server

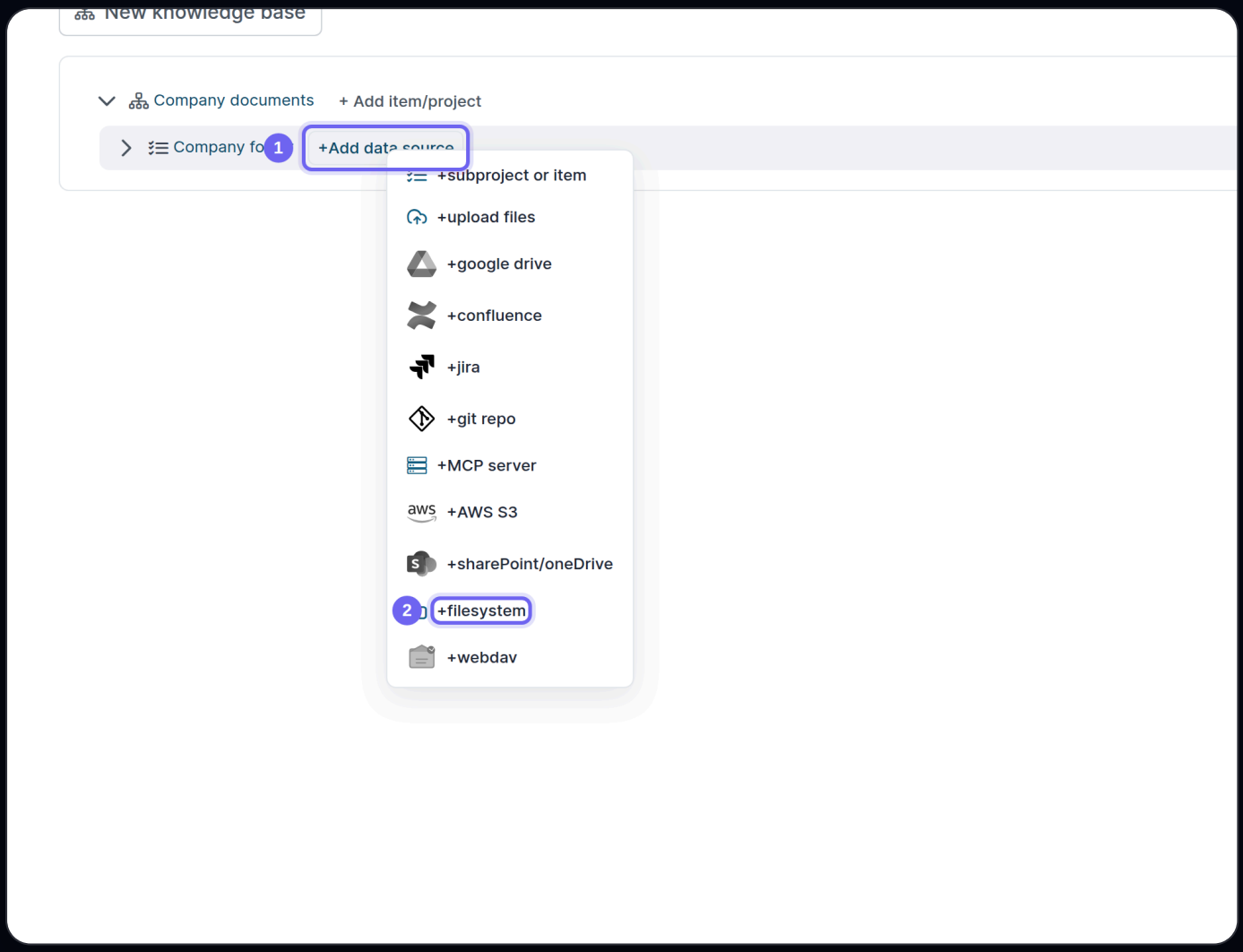
The Windows installation is the `.msi` package of gebo.ai/downloads, described in the manual's section *Installing Gebo.ai on a windows server*. On Windows, and for the Linux service of option 2, you choose the shared folders in the Gebo.ai setup. In **Setup → Enterprise integrations**, click **Shared filesystems paths setup**, then **Add folder**. In **Add a root folder of a shared filesystem** choose, among the server's disks and folders, the one holding the company documents, enter a **Description** and save. Repeat for each disk or folder.

A Windows share must be visible to the Gebo.ai service as a disk or folder of the server. The installer creates the **Gebo.ai Service** under the Local System account, which does not see the drive letters mapped by a logged-in user: make the share available to the service, and readable by the account it runs as.

> **Set the character set**
> On Linux, run the server in a UTF-8 locale and keep `iocharset=utf8` in the options of a Windows share, in the Docker volume or in `/etc/fstab`. They keep the file and folder names with accented letters and other non-English characters as they are; the manual also documents `codepage=cp850`.

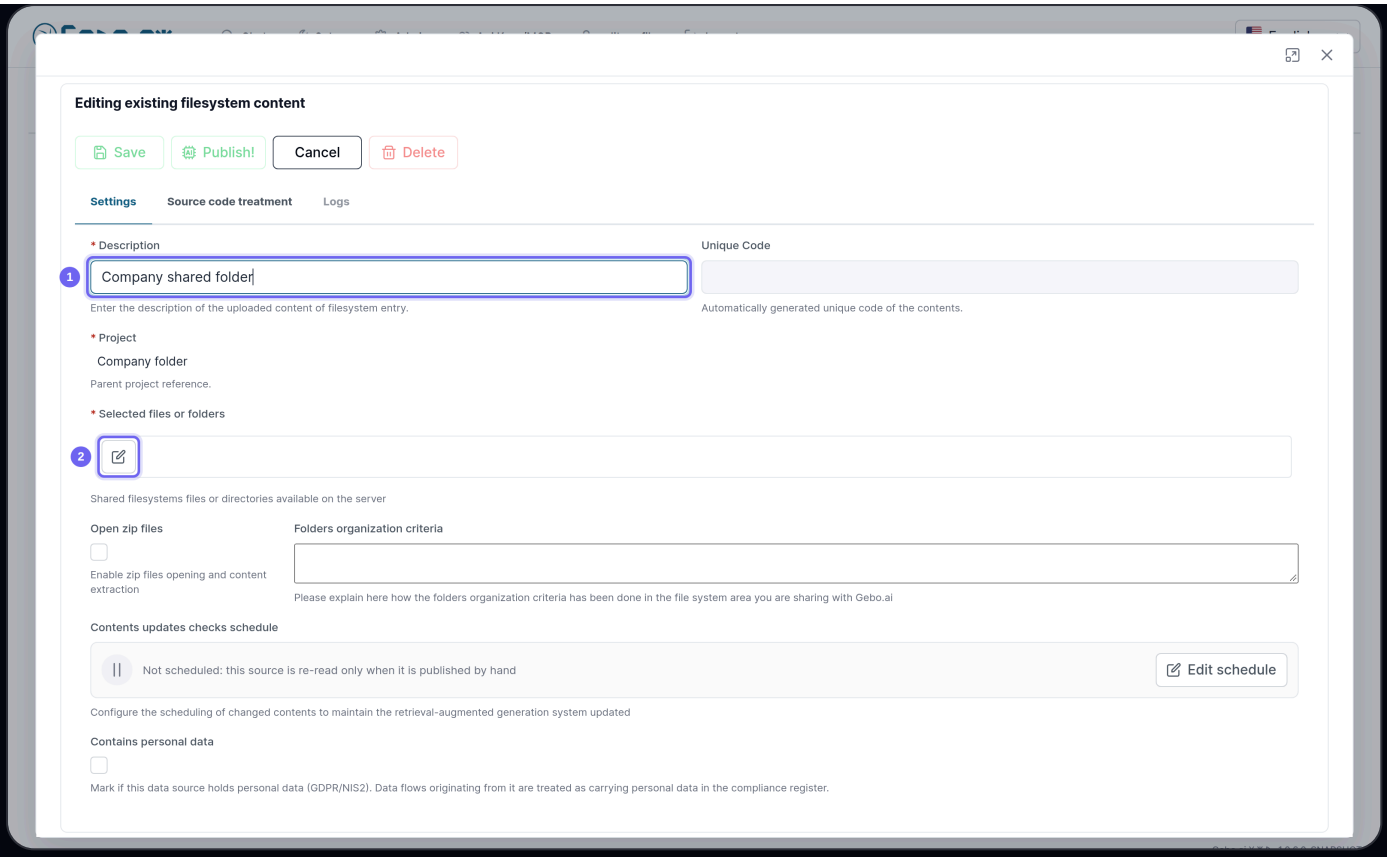
2 In Gebo.ai: create the data source

In **Admin** → **Knowledge bases & projects**, click **+Add data source** next to the project of step 4 and choose **+filesystem**.



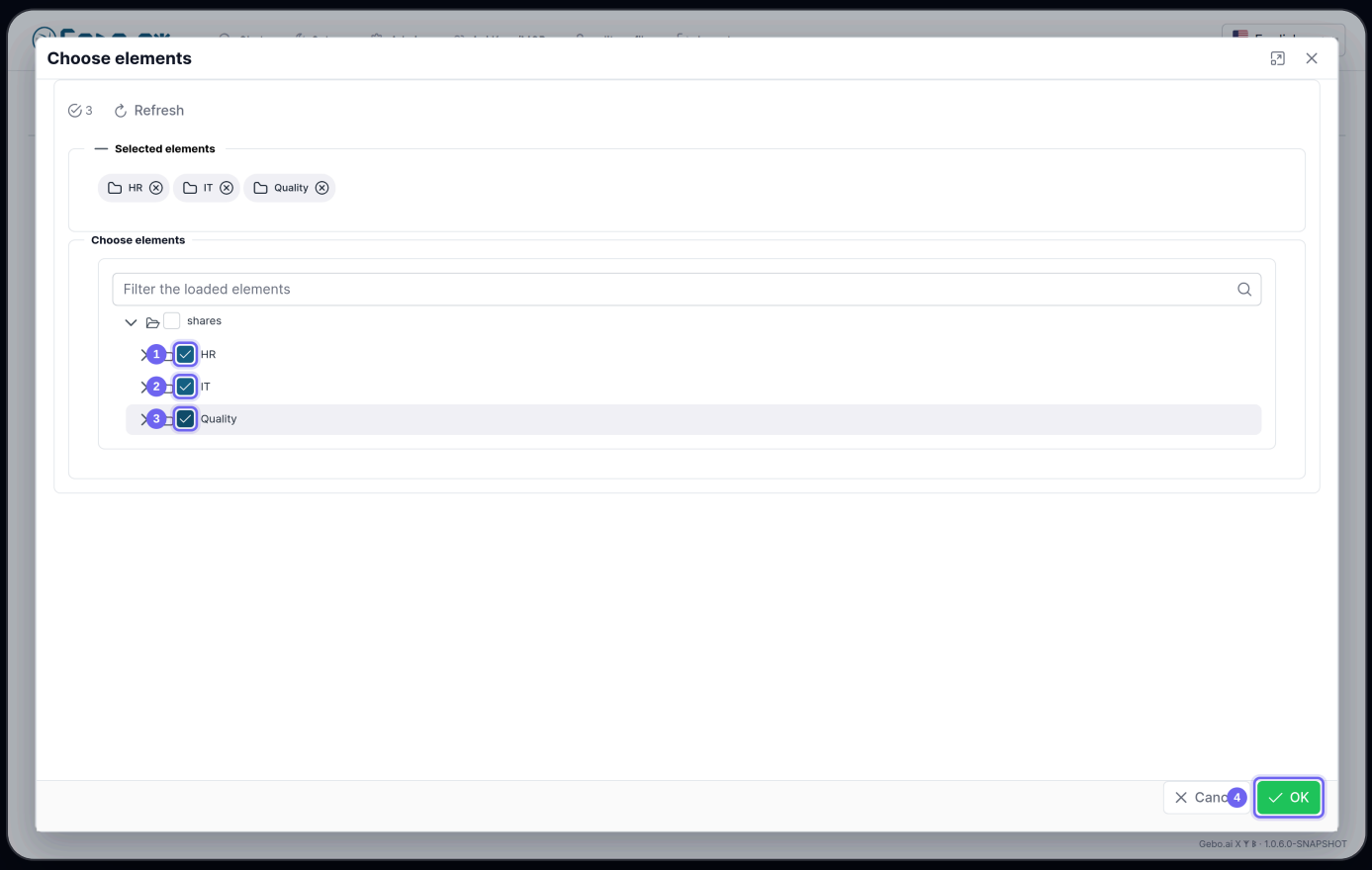
The data source types.

Enter a **Description**, for example *Company shared folder* (1), then click the pencil (2) to choose the folders.



The new filesystem data source.

Expand the shared folder (**shares** with Docker, or the folder added in the setup), tick the subfolders to publish and click **OK**.



Choosing the folders: the subfolders, not the shares root.

Tick the subfolders

Ticking the subfolders keeps the knowledge base safe when a share is momentarily unreachable: the run stops with an error and the documents already read stay in Gebo.ai.

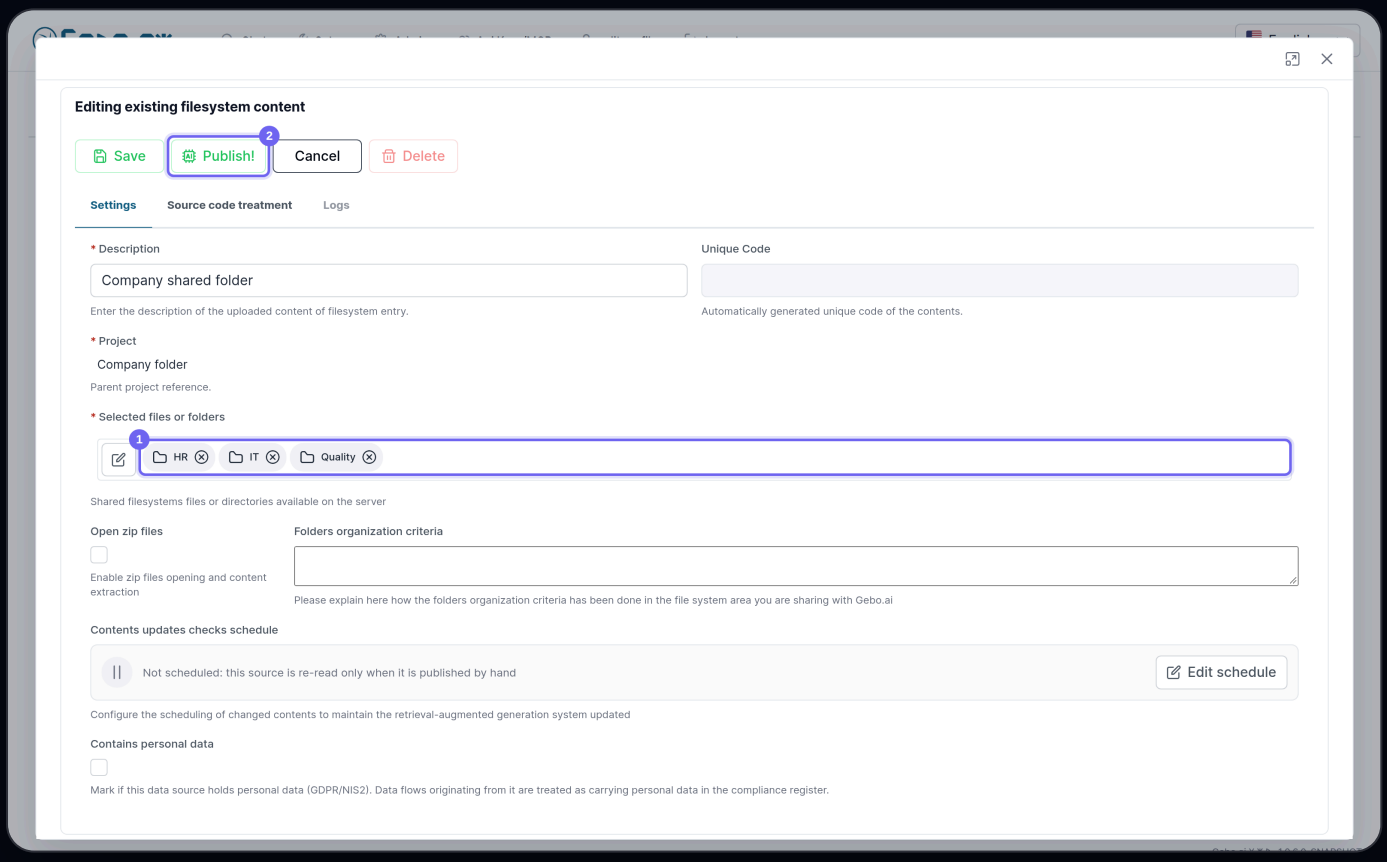
Choose the folders first

A data source keeps the folders chosen when it is saved: to read other folders, add another data source.

---

3 Publish

The chosen folders appear in the editor (1). Click **Publish!** (2).



The data source, ready to publish.

The **Publishing log** opens. Gebo.ai lists the documents, reads them, indexes them for semantic search and for full-text search. When the status reads **Processed completely**, your users can chat with the documents. You can close the window: publishing goes on in the background.

The publishing log: 5 documents received and processed at every stage, none in error.

Changes are picked up by themselves

Gebo.ai also watches the chosen folders and indexes new and changed files within about a minute. The scheduled sync of step 6 is the nightly safety net that checks the whole folder.

← Create a knowledge base and a project

Next: Schedule the evening sync →