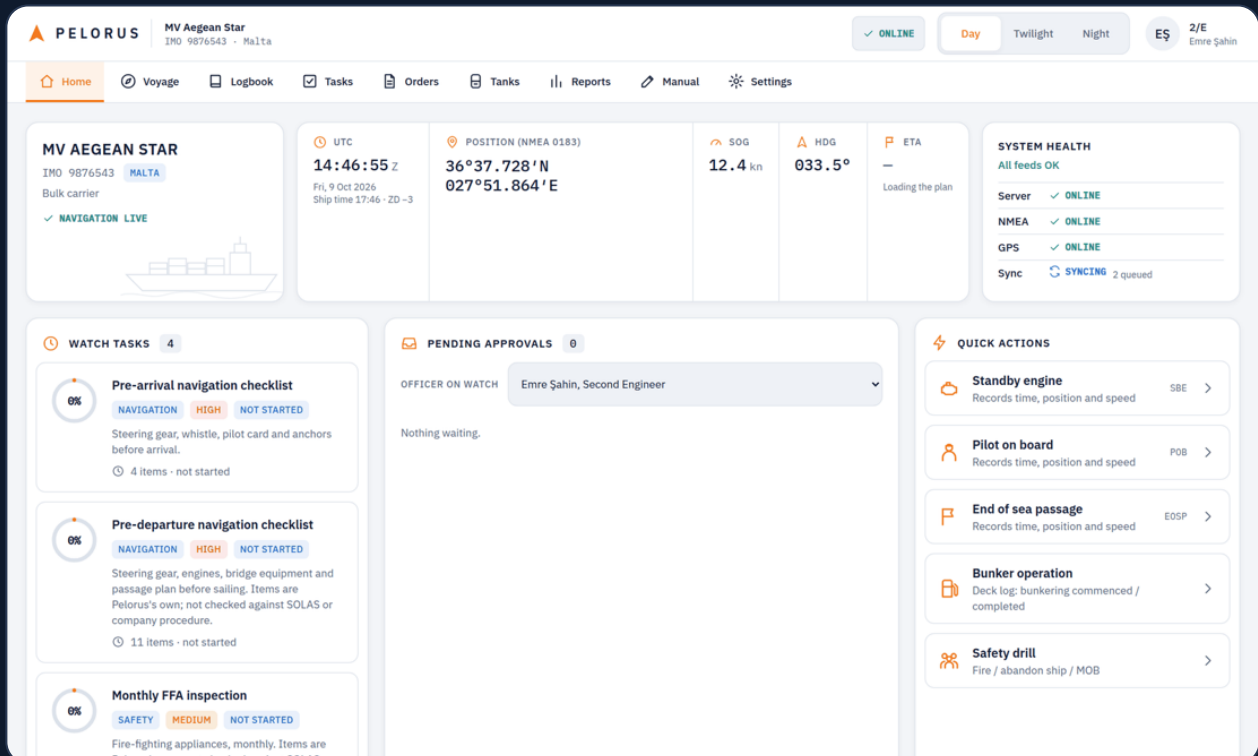


Ship Server Installation Guide

Offline deck and engine logbook for the bridge and the engine room.
Installation, first setup, NMEA connection, licence, office sync and
maintenance.



Contents

1. About this guide.....	3
Who it is for	3
About the screenshots	3
2. System architecture	4
3. Requirements	5
Hardware	5
Software	5
Have this ready	5
4. Installation steps	6
Install Docker and start it at boot.....	6
Copy the Pelorus files	6
Prepare the .env file	6
Start the services	7
Check the automatic start	7
5. Distributing the certificate.....	8
6. First start and the Master's account.....	9
Open the screen: first start	9
Sign in	9
Enrol the officers and engineers	10
Save the vessel particulars	12
7. NMEA connection	13
Give the container access to the NMEA source	13
Set the feed on screen	13
8. Licence	15
Take the hardware id.....	15
Install the licence.....	15
9. Office sync (optional).....	16
Ship-side settings	16
Running the office on the same machine for testing	16
10. Software update	18
How it works.....	18
11. Power cuts, backups and maintenance	19
After a power cut	19
Backups.....	19
Common commands.....	19
12. Troubleshooting.....	20
13. Appendix: sample .env	21

1 About this guide

This guide takes you step by step through installing the Pelorus ship server on a ship's computer and bringing it into service for the first time. Daily use (making entries, approving, signing) is in a separate document, the **Operator Guide**.

Who it is for

Who	What they do in this guide
Ship's IT officer / ETO	Prepares the computer, installs Docker, fills in the <code>.env</code> file, starts the services and distributes the certificate to the bridge PCs.
Master	Creates the first account (the Master's), enrolls the officers and engineers, saves the vessel particulars and the NMEA feed with their PIN, and installs updates.
Company (office)	Provides the licence file; optionally sets up office sync and the fleet dashboard.

About the screenshots

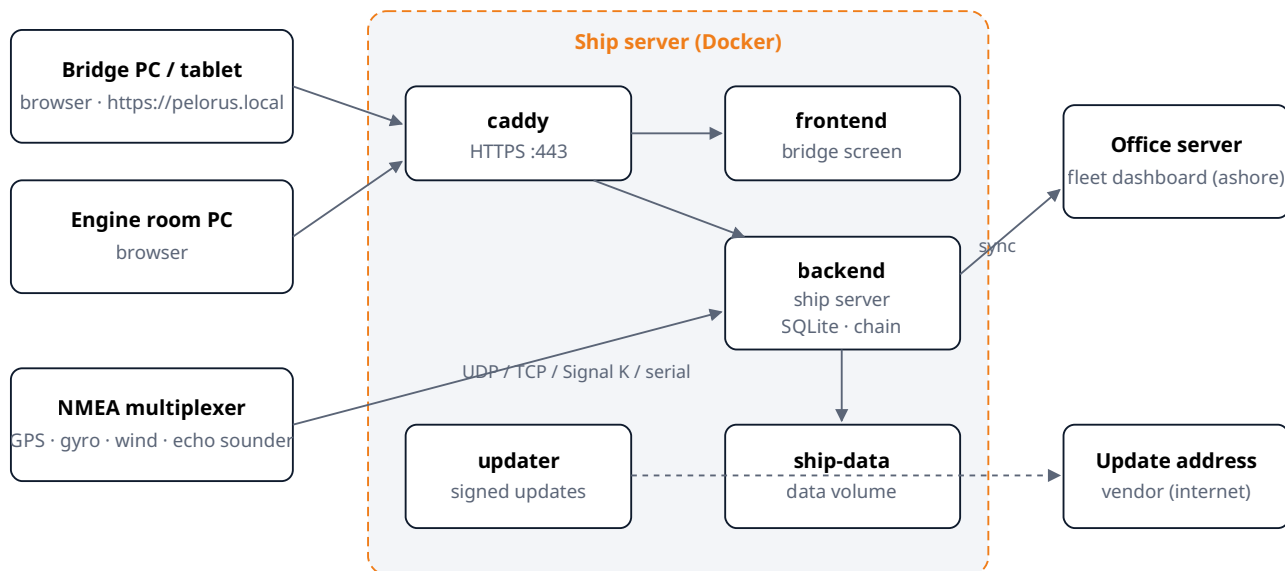
The screenshots come from a test installation running a sample ship (**MV Aegean Star**, IMO 9876543). Names, positions and values are examples. The test installation runs without a licence (development mode), so some PDFs show a *"LICENCE BYPASS (dev)"* stamp. A real ship installation does not.

The basic principle

A sensor feed, a voice note or a typed value only opens a **draft**. The draft is not an official record until the responsible officer has read it and signed it with their PIN. Records are never deleted or overwritten: every change is a new event, chained to the one before with SHA-256. The design follows the electronic record principles of IMO resolution MEPC.312(74). Until class approval is complete, the paper book remains the official record where required.

2 System architecture

Pelorus runs on one computer on board, as Docker containers. Bridge PCs and tablets open it in a web browser over the ship's LAN with **HTTPS**. No internet connection is needed; when a link is available, records go to the office by themselves.



Service	What it does
caddy	The only door to the outside: port 443 . Issues the HTTPS certificate from its own certificate authority (Pelorus Ship CA). Sends <code>/v1</code> requests to the server and everything else to the screen.
backend	The ship server: entries, signatures, rule engine, NMEA listener, PDF reports, office sync, sign-in. The database is one SQLite file in the <code>ship-data</code> volume.
frontend	The bridge screen (PWA). Read-only container.
updater	Software updates. The only container with the Docker socket and the only one that can reach the internet. Installs signed releases only.

`backend` and `frontend` sit on an internal network with no route out. Reaching the NMEA device therefore needs a deliberate opening (see *NMEA connection*).

3 Requirements

Hardware

- A computer on the ship's network that stays on (x86-64). Suggested: 4 GB RAM, 20 GB free disk, wired network.
- A fixed IP address (e.g. `192.168.1.10`) or a name the ship's network resolves (e.g. `pelorus.local`).
- BIOS/UEFI set to **Restore on AC power loss = Power On**, so it starts again after a power cut (see *Power cuts*).
- Optional: a network connection to the NMEA 0183 multiplexer, or a USB-serial adapter.

Software

- Linux (e.g. Ubuntu Server 22.04/24.04 LTS).
- Docker Engine and the Docker Compose plugin (the `docker compose` command).
- The Pelorus distribution folder (`deploy/` with the `core/` and `bridge/` folders next to it).
- **Internet for the first build:** packages are downloaded when the container images are built the first time (`--build`). Later starts need no internet.

Have this ready

Item	Example
The ship's IMO number	9876543 (the identity of the record chain; it cannot be changed later)
Server name and IP address	<code>pelorus.local</code> , <code>192.168.1.10</code>
Ship name, flag, type, call sign, port of registry, MMSI	MV Aegean Star, Malta, Bulk carrier, 9HA1234, Valletta, 256123000
Ship's time (UTC offset)	+3 hours
NMEA source	UDP 10110 broadcast, a TCP address, a Signal K server or <code>/dev/ttyUSB0</code>
The Master's name and a 4–6 digit PIN of their own choosing	—
Licence file (<code>license.key</code>)	From the company / licensor, after you send the hardware id

4 Installation steps

Install Docker and start it at boot

Install Docker Engine and the Compose plugin the way your distribution recommends, then make Docker start with the computer:

```
sudo systemctl enable --now docker
docker compose version          # is the Compose plugin there?
```

Copy the Pelorus files

Copy the Pelorus folder to the server (e.g. `/opt/pelorus`) and change to the deployment folder:

```
cd /opt/pelorus/deploy
ls
# Caddyfile  docker-compose.yml  docker-compose.office.yml  .env.example  check-autostart.sh  ...
```

Prepare the `.env` file

```
cp .env.example .env
nano .env
```

Fields to fill in:

Variable	Meaning
PELORUS_HOSTS	The names and IPs the bridge PCs type in the browser, comma-separated with a space after each comma : <code>pelorus.local, 192.168.1.10</code>
PELORUS_DEFAULT_SNI	If the server is opened by IP address, that IP (needed for the certificate): <code>192.168.1.10</code>
COMPOSE_ENV_FILES	Leave as is: <code>.env</code> , <code>.update.env</code>
PELORUS_VESSEL_IMO	The ship's IMO number. Never changed later.
PELORUS_STATION_TOKEN	A random secret of at least 16 characters (for scripts and the engine-deck bridge).
PELORUS_UPDATER_TOKEN	A second random secret of at least 16 characters (between the server and the updater).
PELORUS_VESSEL_NAME, PELORUS_FLAG, PELORUS_VESSEL_TYPE	Starting values. The Master can change them later on screen.
PELORUS_UTC_OFFSET_MIN	Ship's time, minutes east of UTC (e.g. <code>180</code> for UTC+3).
PELORUS_REQUIRE_LOGIN, PELORUS_SESSION_IDLE_MIN	Sign-in with a PIN on the bridge screen (on by default) and the minutes a session may stay unused before it ends (default 720 = 12 hours).

PELORUS_NMEA	Starting value of the NMEA feed: <code>udp:10110</code> , <code>tcp:HOST:PORT</code> , <code>signalk:HOST:PORT</code> , <code>serial:/dev/ttyUSB0:4800</code> . A setting saved on screen replaces it.
PELORUS_VERIFY_URL	The office verification address the QR code on PDF pages opens. May stay empty until the office runs.
PELORUS_UPDATE_URL	The update address the vendor gives you.

To make a random secret:

```
openssl rand -hex 24
```

Never on a ship

The lines `PELORUS_ENV=development`, `PELORUS_BYPASS_LICENSE=true` and `PELORUS_UPDATE_ALLOW_UNSIGNED=true` are for test installations only. On a ship they stay commented out (`#`).

Start the services

```
touch .update.env          # version pins written by the updater (once)
docker compose up -d --build # the first time needs internet
```

After a few minutes, check that all services run:

```
docker compose ps
NAME                SERVICE    STATUS
pelorus-backend-1   backend    Up (healthy)
pelorus-frontend-1 frontend    Up (healthy)
pelorus-updater-1    updater     Up
pelorus-caddy-1      caddy      Up
```

Caddy waits until `backend` and `frontend` report *healthy*. If a service keeps restarting, read its log: `docker compose logs backend`.

Check the automatic start

```
./check-autostart.sh
OK      Docker starts at boot
OK      backend restarts by itself
OK      frontend restarts by itself
OK      updater restarts by itself
OK      caddy restarts by itself
CHECK   BIOS/UEFI: set 'Restore on AC power loss' (or 'After power failure') to Power On ...
```

Every line starting with `FIX` is fixed with the command printed next to it. The BIOS setting cannot be changed by software; it is made in the computer's setup menu.

5 Distributing the certificate

Caddy creates a certificate authority of the ship's own (**Pelorus Ship CA**). For bridge PCs and tablets not to show a “not secure” warning, this root certificate is installed once on every client. It stays the same when the containers are rebuilt (it is kept in the `caddy-data` volume).

Take the certificate from the server

```
docker compose cp caddy:/data/caddy/pki/authorities/local/root.crt ./pelorus-root-ca.crt
```

Windows clients (as administrator)

```
certutil -addstore -f Root pelorus-root-ca.crt
```

Linux clients (Ubuntu/Debian)

```
sudo cp pelorus-root-ca.crt /usr/local/share/ca-certificates/pelorus-root-ca.crt
sudo update-ca-certificates
```

macOS clients

```
sudo security add-trusted-cert -d -r trustRoot -k /Library/Keychains/System.keychain pelorus-root-ca.crt
```

On tablets, copy the certificate file to the device and install it under *Settings* › *Security* › *Install certificate* (Android) or as a profile (iOS/iPadOS: after installing, enable it under *General* › *About* › *Certificate Trust Settings*).

Firefox may use its own certificate store. If the warning stays, import the same file in Firefox under *Settings* › *Privacy & Security* › *View Certificates* › *Authorities* › *Import*.

6 First start and the Master's account

Open the screen: first start

On a bridge PC, open the browser at `https://pelorus.local` (or `https://192.168.1.10`). With no account yet, the screen opens on the **First start: the master's account** form.

FIRST START: THE MASTER'S ACCOUNT

No crew member is enrolled on this ship server yet. The master's account comes first; the master then adds the officers and engineers under Settings.

NAME (PRINTED WITH THE SIGNATURE)

Capt. Ahmet Demir

USER ID (LETTERS, DIGITS, DOT, DASH; FIXED)

master

PIN (4-6 DIGITS)

REPEAT PIN

....

....

Create the master's account

Figure 1 · First start: creating the Master's account.

- 1 Name:** the name printed with the signature (e.g. *Capt. Ahmet Demir*).
- 2 User id:** letters, digits, dot, dash. It identifies the person on every signature and never changes. Default `master`.
- 3 PIN** and its repeat: 4-6 digits. Repeated or straight runs such as `1111` or `1234` are refused.
- 4 Create the master's account:** the account is made and the Master is signed in.

On a new installation the **first account must be the Master's**. This form only appears while no account exists. From then on, the Master creates every account.

Sign in

Every later visit starts on **Sign in**. Each officer and engineer picks their own name and enters **their own PIN**. The sign-in PIN is the same PIN that signs entries, with the same lockout: after 5 wrong tries 5 minutes, then 15 minutes, then 1 hour.

SIGN IN

Choose your name and enter your own PIN. The same PIN signs your entries.

CREW MEMBER

☐ Emre Şahin, Second Engineer

☐ Deniz Aksoy, Third Engineer

☒ Can Öztürk, Third Officer

☐ Ali Arslan, Second Officer

☐ Hasan Kaya, Chief Engineer

☐ Mehmet Yılmaz, Chief Officer

☐ Capt. Ahmet Demir, Master

PIN (4–6 DIGITS)

1

2

3

4

5

6

7

8

9

Clear

0

Sign in

Figure 2 · Sign-in screen: crew list on the left, PIN and keypad on the right.

A session ends after 12 hours unused, at **Account > Sign out**, when the browser is closed, when the person is signed off, or when the server restarts (e.g. after an update). The idle time is set with `PELORUS_SESSION_IDLE_MIN`. Sign-in can be switched off with `PELORUS_REQUIRE_LOGIN=false`; this is not recommended on a ship.

Scripts and office integrations keep reaching the `/v1` routes with `Authorization: Bearer <PELORUS_STATION_TOKEN>`.

Enrol the officers and engineers

Settings > Crew on board > Add crew member: name, rank and the person's first PIN. Below, the Master picks themselves and confirms with their own PIN.

Figure 3 · New crew member: name, rank, first PIN; the Master's confirmation.

Figure 4 · Crew on board: **New PIN** resets a PIN, **Sign off** deactivates someone who left the ship.

Figure 3 · New crew member: name, rank, first PIN; the Master's confirmation.

Figure 4 · Crew on board: **New PIN** resets a PIN, **Sign off** deactivates someone who left the ship.

The rank decides which books a person may sign:

Role	Ranks	Books they may sign
Master	Master	Deck, engine, ORB Part I, Annex VI, ORB Part II, Cargo, Garbage, BWRB, Compass, Radar, GMDSS; page countersignature; orders
OOW (Officer of the watch)	Chief Officer, 2nd Officer, 3rd Officer	Deck log, ORB Part II, Cargo Record Book, Garbage, BWRB, Compass, Radar, GMDSS
Chief Engineer	Chief Engineer	Engine log (and engine page countersignature), ORB Part I, Annex VI, Garbage, BWRB
Duty Engineer	2nd, 3rd, 4th Engineer	Engine log, ORB Part I, Annex VI, Garbage, BWRB

Cadet, rating and ETO accounts can be enrolled but carry no signing right. The office can also create accounts other than the Master's remotely (see *Office sync*).

Save the vessel particulars

Settings › Vessel › Edit vessel details: ship name, flag, type, call sign, port of registry, MMSI and ship's time, saved with the Master's PIN. Saved values replace the starting values from `.env`. The IMO number is the identity of the record chain and cannot be changed.

 **VESSEL**

Name	MV Aegean Star
IMO number	9876543
Flag	Malta
Type	Bulk carrier
Call sign	9HA1234
Port of registry	Valletta
MMSI	256123000
Ship's time	ZD -3

Edit vessel details

Figure 5 · Vessel particulars.

 **VESSEL**

Name

MV Aegean Star

IMO number

9876543

Flag

Malta

Call sign

9HA1234

MMSI

256123000

Master

Capt. Ahmet Demir (Master)

Fixed: the record chain is tied to it.

Type

Bulk carrier

Port of registry

Valletta

Ship's time, hours from UTC

3

PIN

Save

Cancel

Figure 6 · Edit form: saved with the Master and their PIN.

7 NMEA connection

Pelorus reads navigation data (position, course, speed, heading, wind, depth, GNSS time) from the ship's NMEA feed. Sentences: RMC, GGA, ZDA, VTG, HDT, THS, MWD, MWV, DPT, DBT. NMEA 2000 data comes through a gateway that converts to 0183 sentences, or through a Signal K server.

Sensor data only fills drafts. An old value is never shown as current: position is used for 5 seconds, other values for 10 seconds. If the feed goes silent, drafts get no values and the reason goes on record (no_feed, no_fix, invalid_checksum).

Give the container access to the NMEA source

backend sits on an internal network by default. Depending on the source, add to the backend service in docker-compose.yml:

Source	docker-compose.yml change
TCP / Signal K (WebSocket default port 3000)	networks: [internal, lan]
UDP broadcast	networks: [internal, lan] and ports: ["10110:10110/udp"] (the port chosen in Settings)
Serial port (USB)	The device mapping below

```
backend:
  devices: ["/dev/ttyUSB0:/dev/ttyUSB0"]
  group_add: ["dialout"] # or the device's group id: ls -ln /dev/ttyUSB0
```

Then `docker compose up -d backend`. For a USB adapter whose number changes after a re-plug, use its stable name under `/dev/serial/by-id/...`.

Set the feed on screen

Settings > NMEA feed > Edit NMEA feed: source type (UDP, TCP, Signal K WebSocket, Signal K TCP, serial port), address/port and on/off. **Test Connection** tries the source for a few seconds before saving. **Save and restart feed** saves it with the Master's PIN; the listener restarts at once.

Figure 7: NMEA setting (Master only).

The interface shows the 'NMEA FEED' settings. A green 'CONNECTED' status is displayed in the top right. A checkbox 'Read navigation data from a sensor feed' is checked. The 'Source' dropdown is set to 'NMEA 0183 over UDP (listen)' and the 'Port' is '20110'. The 'Master' dropdown is set to 'Capt. Ahmet Demir (Master)'. Below these fields, a note says 'Port the multiplexer broadcasts to (usually 10110)'. At the bottom are buttons for 'Save and restart feed', 'Test Connection', and 'Cancel'.

Figure 8: Connected feed: state, sentence counts and the latest raw sentences.

The interface shows the 'NMEA FEED' status. A green 'CONNECTED' status is displayed in the top right. The 'Source' is 'NMEA 0183 over UDP (listen)', 'Address' is 'listening on UDP port 20110', 'State' is 'Receiving', 'Last data received' is '0 s ago', 'Sentences' is '9642 received, 9642 used, 0 rejected (checksum or bad value)', and 'Drafts get' is 'captured'. Below this is a section for 'Sensor or GPS wrong? Cut the feed off at once: drafts then take no NMEA value and everything is entered by hand.' with an 'Officer' dropdown set to 'Can Öztürk (Third Officer)' and a button 'Disable Sensor Data / Force Manual Entry'. Below this is a section 'Set by the installation (PELORUS_NMEA). Saving here replaces it.' with a note 'Sensors only fill drafts. Nothing from the feed is a record until an officer signs it.' and a 'Latest sentences' section showing raw NMEA sentences. At the bottom is an 'Edit NMEA feed' button.

Figure 7 · NMEA setting (Master only).

Figure 8 · Connected feed: state, sentence counts and the latest raw sentences.

The card should show **CONNECTED**, the *Sentences* count should rise and *Drafts get* should read **captured**. If the source cannot be reached, the error shows on the *Source* line.

Check on board

```
docker compose exec backend python -m pelorus_core.nmea --udp 10110
# prints every 5 seconds the snapshot a draft would get
```

8 Licence

The ship server needs a signed `license.key` for the vessel's IMO number and this computer. Without a valid licence, or after its validity and grace period, it runs **read-only**:

- No new entries or signatures are accepted (403 `license_read_only`).
- History stays readable and PDFs stay downloadable; they carry an "UNLICENSED COPY" watermark.
- No data is deleted. Office sync is not affected.

Take the hardware id

```
docker compose run --rm --no-deps backend python -m pelorus_core.license_validator hwid
```

Send the 64-character hex string to the licenser.

Install the licence

```
docker compose cp license.key backend:/data/license.key
```

The file is picked up without a restart. The licence state shows on the *Licence* line in **Settings** and in the banner at the top of the screen. In the last 30 days it says *expiring*; after expiry a grace period (14 days by default) begins.

Replacing the mainboard or network card, or reinstalling the operating system (new `machine-id`), needs a new licence. Setting the clock back does not revive an expired licence.

9 Office sync (optional)

The ship server pushes its record chain to the office server every minute and pulls the office's corrections. Without a link the ship works normally; when the link returns, the remaining records go. The office can never overwrite a signed ship record: an office correction is kept as a separate revision.

Ship-side settings

Variable	Meaning
<code>PELORUS_OFFICE_URL</code>	The office server address
<code>PELORUS_OFFICE_SYNC_TOKEN</code>	This ship's secret at the office (16+ characters)
<code>PELORUS_OFFICE_SYNC_INTERVAL_S</code>	Sync interval, seconds (default 60)
<code>PELORUS_VERIFY_URL</code>	The office verification page the QR code on PDFs opens

The state shows on the **Sync** line of the home screen: **Online**, **Offline** (no link, records queued) or **Halted** (ship and office chains disagree).

If “Halted” appears

The ship and office record chains do not match (e.g. a wrong backup was restored). The system does not resolve this by itself. Do not reset anything; contact the company.

Running the office on the same machine for testing

To see what the office sees, the office server and fleet dashboard can run on the same machine. This is **not** used on a ship; the office runs ashore.

- 1 In `.env` uncomment: `PELORUS_OFFICE_URL=http://office-server:8090`, `PELORUS_OFFICE_SYNC_TOKEN`, `PELORUS_OFFICE_TOKEN` (16+ characters) and `PELORUS_OFFICE_HOSTS` (the same names as `PELORUS_HOSTS`, each ending in `:8443`).
- 2 `docker compose -f docker-compose.yml -f docker-compose.office.yml up -d --build`
- 3 Open `https://pelorus.local:8443` and type the office token (`PELORUS_OFFICE_TOKEN`). The ship's records appear after the first sync round (about a minute).

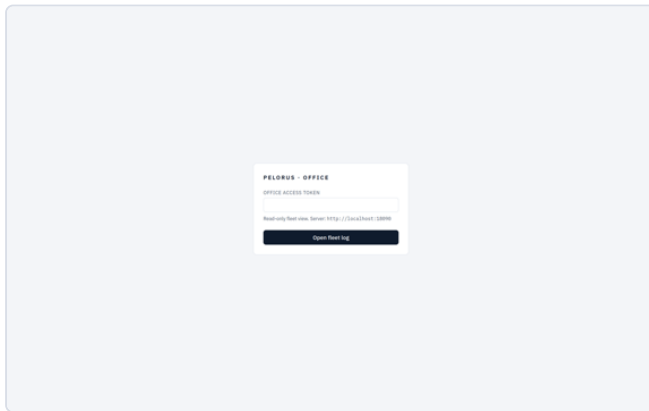


Figure 9 · Office dashboard sign-in: the office token is typed once per browser tab.

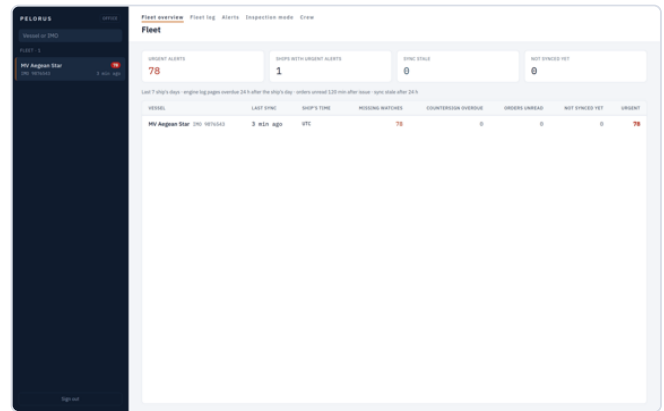


Figure 10 · Fleet overview: last sync, missing watches, overdue countersignatures and unread orders.

Mail to the ship when the office creates a crew account (optional)

For a notice mail to the ship when the office creates an account (the PIN is never in the mail):

PELORUS_VESSEL_EMAIL , PELORUS_SMTP_HOST , PELORUS_SMTP_PORT , PELORUS_SMTP_USER ,
PELORUS_SMTP_PASSWORD , PELORUS_SMTP_FROM , PELORUS_SMTP_TLS (starttls , ssl or none). Recreate the
office server after changing them.

10 Software update

The Master updates the ship from the bridge, no terminal needed: **Settings › Software update › Check for updates**, then **Install** and the Master's PIN.

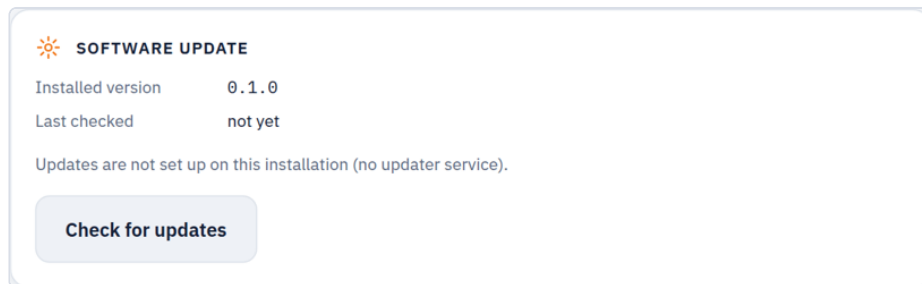


Figure 11 · Software update card. (The test installation has no updater service, so it says “not set up”.)

How it works

- 1 The updater fetches the release manifest from `PELORUS_UPDATE_URL` and checks the vendor's signature itself. An unsigned or older release is refused.
- 2 Before installing, the ship server copies its database to `backups/` (the last five are kept).
- 3 The two images named in the manifest (pinned by digest) are pulled, `backend` and `frontend` are swapped and `/health` is awaited.
- 4 If the new version does not come up healthy, the previous one is put back. The data volume is never touched.

The installed version is kept in `.update.env`, so a later `docker compose up -d` stays on it. Updates work on an unlicensed or lapsed copy too. `updater` and `caddy` are not replaced by an update; rebuild them when needed with `docker compose up -d --build updater`. An update restarts the server, so everyone signs in again.

11 Power cuts, backups and maintenance

After a power cut

The ship server comes back by itself when power returns, with nobody logging in. Three things make that true (`./check-autostart.sh` tells you which is missing):

1. BIOS/UEFI: **Restore on AC power loss** (or *After power failure*) = **Power On**.
2. Docker starts at boot: `sudo systemctl enable docker`.
3. The containers carry `restart: always` and start with Docker.

The database is written with `synchronous=FULL` in WAL mode. A cut loses at most the entry being written at that instant; earlier entries are never lost and the chain stays verifiable.

Backups

The ship's data lives in the Docker volume `pelorus_ship-data`. For a regular backup, stop the server briefly, archive the volume and copy the archive to a separate disk:

```
docker compose stop backend
docker run --rm -v pelorus_ship-data:/data -v "$PWD":/b alpine \
  tar czf /b/pelorus-data-$(date +%F).tgz -C /data .
docker compose start backend
```

Restoring a backup may put office sync into **Halted** (the ship chain falls behind the office's copy). Plan a restore together with the company.

Common commands

Command	What it does
<code>docker compose ps</code>	Service status
<code>docker compose logs -f backend</code>	Ship server log (live)
<code>docker compose restart backend</code>	Restarts the server (data kept; everyone signs in again)
<code>docker compose up -d</code>	Starts missing or stopped services
<code>./check-autostart.sh</code>	Automatic start check
<code>curl --cacert pelorus-root-ca.crt https://pelorus.local/health</code>	Is the server up?

12 Troubleshooting

Symptom	Likely cause	Fix
Browser says “your connection is not private”	Root certificate not installed on the client	See <i>Distributing the certificate</i> .
Certificate error when opened by IP	<code>PELORUS_DEFAULT_SNI</code> empty	Put the IP in <code>PELORUS_HOSTS</code> and <code>PELORUS_DEFAULT_SNI</code> in <code>.env</code> , then <code>docker compose up -d</code> .
NO FEED on screen, NMEA/GPS OFFLINE	No feed, or the container cannot reach the source	Check the multiplexer and cable; check the <code>lan</code> network / UDP port / serial device mapping; Test Connection .
Open error under <i>Source</i> on the NMEA card	Serial device not passed to the container, or no permission	The <code>devices</code> and <code>group_add</code> lines (see <i>NMEA connection</i>).
No entries or signatures, <code>license_read_only</code>	No licence, wrong machine or expired	See <i>Licence</i> . Reading and PDFs stay open.
Screen went back to the sign-in page	Session ended (12 h unused, server restarted, or the person was signed off)	Pick your name and sign in again with your PIN.
Sync Offline	No link to the office	Nothing to do. Records are queued and go when the link returns.
Sync Halted	Ship and office chains disagree	Do not reset. Contact the company.
A service keeps restarting	Missing or wrong <code>.env</code> value (e.g. a token shorter than 16 characters)	The error line in <code>docker compose logs <service></code> .

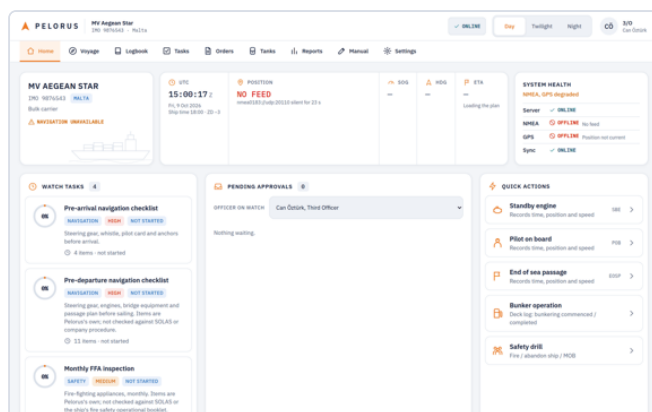


Figure 12 · Home screen with the feed lost: position, SOG and HDG are blanked and the reason shown. No old value is displayed.

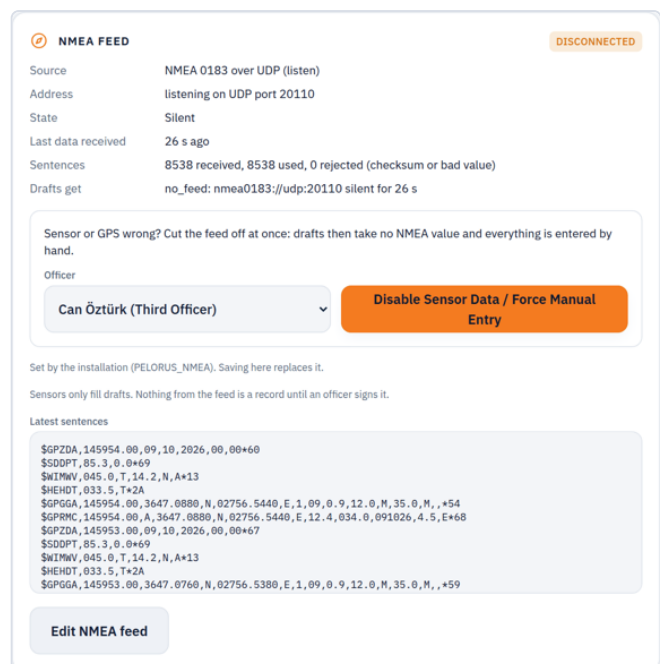


Figure 13 · NMEA card: **DISCONNECTED**, “Silent”; drafts get `no_feed`.

13 Appendix: sample .env

```
PELORUS_HOSTS=pelorus.local, 192.168.1.10
PELORUS_DEFAULT_SNI=192.168.1.10
COMPOSE_ENV_FILES=.env, .update.env

PELORUS_VESSEL_IMO=9876543
PELORUS_STATION_TOKEN=<openssl rand -hex 24>
PELORUS_VESSEL_NAME=MV Aegean Star
PELORUS_FLAG=Malta
PELORUS_VESSEL_TYPE=Bulk carrier
PELORUS_CALL_SIGN=9HA1234
PELORUS_PORT_OF_REGISTRY=Valletta
PELORUS_MMSI=256123000
PELORUS_UTC_OFFSET_MIN=180
PELORUS_NMEA=udp:10110
PELORUS_VERIFY_URL=https://office.example.com

# Sign-in on the bridge screen (default on)
# PELORUS_REQUIRE_LOGIN=true
# PELORUS_SESSION_IDLE_MIN=720

# Office sync (optional)
# PELORUS_OFFICE_URL=https://office.example.com
# PELORUS_OFFICE_SYNC_TOKEN=<16+ characters>

# Software update
PELORUS_UPDATE_URL=https://updates.example.com/pelorus/stable.json
PELORUS_UPDATER_TOKEN=<openssl rand -hex 24>
```

© 2026 Pelorus. All rights reserved.