Admin & Operations

Downloadable reference generated from the NetOS Markdown documentation.

Xiber NetOS — Admin & Operations Manual

> Development setup, services, configuration, migrations, roles, and troubleshooting.

Tech Stack

COMPONENT	TECHNOLOGY	NOTES
Database	PostgreSQL 16	PostGIS (geospatial), pgvector (embeddings)
Backend	FastAPI + SQLAlchemy 2.0	Async via asyncpg, Pydantic v2 validation
Migrations	Alembic	Baseline + infrastructure + agreements + audit/feedback versions
Frontend	Next.js 14 + React 18	TypeScript 5.5, TailwindCSS, App Router
Maps	MapLibre GL	Vector tile rendering
Queue	Celery + Redis	Wired but workers not yet active
MCP	Python FastMCP	Skeleton for XOS agent tools
Dev Deployment	Docker Compose	Hot-reload for both API and web

Prerequisites

- Docker Engine 24+ with Docker Compose v2
- 4 GB RAM minimum
- Ports 3000, 5432, 6379, 8000 available
- (Optional) Python 3.12+ and Node.js 20+ for local development outside Docker

Repository Layout

```
circuitos/
├── apps/
│   ├── api/                FastAPI backend
│   │   ├── app/
```

```
│   │   │   ├── main.py          Entry point
│   │   │   ├── api/v1/          Route handlers
│   │   │   ├── core/            Auth, config
│   │   │   ├── db/              Session management
│   │   │   ├── models/          SQLAlchemy entities + enums
│   │   │   ├── schemas/         Pydantic request/response models
│   │   │   ├── services/        Business logic (import, extraction)
│   │   │   └── workers/         Celery task definitions
│   │   ├── alembic/             Migration versions
│   │   ├── scripts/             Seed data, utilities
│   │   ├── pyproject.toml       Python dependencies
│   │   └── Dockerfile
│   ├── web/                 Next.js frontend
│   │   ├── app/                 App Router pages
│   │   ├── components/          React components
│   │   ├── lib/                 API client, utilities
│   │   ├── package.json
│   │   └── Dockerfile
│   └── mcp/                 MCP server skeleton
│       ├── app/server.py
│       └── pyproject.toml
├── packages/
│   └── shared/              TypeScript type contracts
├── infra/
│   └── docker/
│       ├── docker-compose.yml          Local dev stack
│       ├── docker-compose.public.yml   Production profile (SSO + tunnel)
│       ├── nginx/                      Public reverse proxy config
│       └── postgres/                   Custom PostgreSQL Dockerfile
├── docs/                    Documentation (you are here)
├── examples/                Sample CSV data
└── .env.example             Environment variable template
```

Local Development

Start Everything

```
cd infra/docker
docker compose up --build
```

Run Migrations

```
docker compose exec api alembic upgrade head
```

Seed Sample Data

```
docker compose exec -T -e PYTHONPATH=/app api python scripts/seed_sample_data.py
```

Creates: 4 carriers (Lumen, Zayo, Cogent, Hurricane Electric), 10 endpoints with coordinates, 8 circuits with contracts, and lifecycle events. The script is idempotent — safe to run multiple times.

Restart Services

```
docker compose restart api web
```

Rebuild After Dependency Changes

```
docker compose up --build api web
```

View Logs

```
docker compose logs -f api        # API logs
docker compose logs -f web        # Frontend logs
docker compose logs -f postgres   # Database logs
```

Services

SERVICE	URL	DOCKER NAME
Web UI	`http://localhost:3000`	`web`
API	`http://localhost:8000`	`api`
API Docs (Swagger)	`http://localhost:8000/docs`	`api`
API Health	`http://localhost:8000/healthz`	`api`
PostgreSQL	`localhost:5432`	`postgres`
Redis	`localhost:6379`	`redis`

Server Deployment

The current demo deployment runs on the Xiber NetOS server.

SERVICE	URL
Public UI	`https://circuitos.xiberian.net`
Direct UI	`http://172.98.130.238:3010`
Direct API Docs	`http://172.98.130.238:8010/docs`

Server path:

```
/home/stephen/netos
```

Server compose file:

```
infra/docker/docker-compose.server.yml
```

The server uses legacy Compose v1:

```
docker-compose -p netos -f docker-compose.server.yml ps
docker-compose -p netos -f docker-compose.server.yml logs -f api
docker-compose -p netos -f docker-compose.server.yml logs -f web
```

After code changes, rebuild and restart API/web:

```
cd ~/netos/infra/docker
docker-compose -p netos -f docker-compose.server.yml build api web
docker rm -f netos_api_1 netos_web_1 || true
docker-compose -p netos -f docker-compose.server.yml up -d api web
```

The server web service runs `next start` from a built image. Do not mount `apps/web` or `.next` over the production container; only `docs/` is mounted read-only at `/app/content/docs`.

For fast frontend iteration, run a separate local preview service instead of putting the public web container into development mode:

```
cd ~/netos
scripts/start-web-preview.sh
```

The preview service runs `next dev` with the web source bind-mounted. It is available locally at `http://127.0.0.1:3011/preview` and through the protected public proxy at `https://netos.xiberian.net/preview`. When a set of changes is ready for the public app, promote it with:

```
cd ~/netos
scripts/deploy-prod-web.sh
```

Run migrations on the server:

```
cd ~/netos/infra/docker
docker-compose -p netos -f docker-compose.server.yml run --rm api alembic upgrade head
```

Configuration

RF System Standards

RF system standards are maintained from the Admin page. They define the selectable defaults used when creating RF Links.

FIELD	PURPOSE
Name	System label used in generated RF Link names and equipment defaults
Description	Short operating note or intended use case
Frequency Used	Human-readable spectrum band, such as `CBRS / 5 GHz` or `11 GHz licensed`
Capacity Mbps	Default capacity applied to new RF Links
Cost	Default CAPEX applied to new RF Links
Type	PTP or PtMP
Role	Standalone, PtMP Base/Sector, or PtMP Subscriber
Compatible Bases	For PtMP subscriber radios, the base/sector systems that can serve that subscriber
Active	Controls whether the system appears as a selectable option for new RF Links

Use **PtMP Base/Sector** for equipment installed on an infrastructure asset as a PtMP node, such as Tarana BN, Siklu Terragraph DN, or Cambium ePMP AP. Use **PtMP Subscriber** for customer-side radios, such as Tarana RN/RNv, Siklu T280/T265, or Cambium Force 4600c. When a user creates a PtMP RF Link from a selected PtMP source node, NetOS shows only subscriber radios whose compatible base list includes that source node's base system.

RF Links no longer require a manually entered link name. NetOS generates the display name from the selected A facility, Z facility or customer endpoint, and RF system/equipment. Links between two infrastructure assets are classified as backhaul. Links from an infrastructure asset to a customer endpoint are classified as customer endlinks.

Environment Variables

VARIABLE	DEFAULT	DESCRIPTION
`DATABASE_URL`	`postgresql+asyncpg://circuitos:circuitos@postgres:5432/circuitos`	Async database connection string
`REDIS_URL`	`redis://redis:6379/0`	Redis connection for Celery broker
`ENTRA_TENANT_ID`	`common`	Microsoft Entra tenant (dev placeholder)
`ENTRA_CLIENT_ID`	`dev-client-id`	Entra app client ID (dev placeholder)
`JWT_AUDIENCE`	`api://circuitos`	Expected JWT audience claim
`API_BASE_URL`	`http://api:8000`	Server-side API URL (Docker internal)
`NEXT_PUBLIC_API_BASE_URL`	*(empty)*	Browser-side API URL (falls back to current host:8000)
`AI_PROVIDER`	`deterministic`	Bootstrap provider until Admin > AI settings are saved. Supported app providers are `openai`, `minimax`, and `deterministic`
`OPENAI_API_KEY`	*(empty)*	Bootstrap API key used by the backend assistant service; never expose this to the browser
`OPENAI_MODEL`	`gpt-5.2`	

VARIABLE	DEFAULT	DESCRIPTION
		Bootstrap model used for assistant extraction
`OPENAI_BASE_URL`	`https://api.openai.com/v1`	Bootstrap OpenAI-compatible base URL
`SMTP_HOST`	*(empty)*	Bootstrap SMTP host used until Admin notification settings are saved. If unset and no Admin setting exists, scheduled renewal reports are skipped
`SMTP_PORT`	`587`	SMTP port
`SMTP_USERNAME`	*(empty)*	Optional SMTP username
`SMTP_PASSWORD`	*(empty)*	Optional SMTP password
`SMTP_FROM`	`netos@xiber.com`	Bootstrap from address for system email
`NETOS_PUBLIC_BASE_URL`	`https://netos.xiberian.net`	Bootstrap public URL used to generate report links back into NetOS
`RENEWAL_REPORT_ENABLED`	`true`	Bootstrap toggle for the weekly renewal email scheduler
`RENEWAL_REPORT_WEEKDAY`	`0`	Bootstrap UTC weekday for the scheduled report, where Monday is `0`
`RENEWAL_REPORT_HOUR`	`8`	Bootstrap UTC hour after which the scheduled report can send
`RENEWAL_REPORT_LOOKAHEAD_DAYS`	`180`	Bootstrap renewal decision lookahead window included in the report

Copy `.env.example` to `.env` and adjust as needed. For production SSO, see Public Hostname & SSO.

AI Assistant Provider

The NetOS Assistant uses `/api/v1/ai/extract`. With no provider configured it uses the deterministic parser. Admin users should normally configure the live provider from **Admin > AI**, where API keys are encrypted server-side and only the last four characters are shown after save.

Supported AI providers:

- **OpenAI**: default base URL `https://api.openai.com/v1`, using the Responses API with strict JSON output.
- **MiniMax**: default base URL `https://api.minimax.io/v1`, default model `MiniMax-M3`, using the OpenAI-compatible Chat Completions API for text/image tasks.
- **Fallback parser**: deterministic extraction with no external model.

Use **Test** in Admin > AI after entering a key. The test calls the provider's OpenAI-compatible `/models` endpoint and verifies that the selected model is advertised before users rely on the provider.

The assistant never writes inventory directly. It returns draft payloads that users apply to forms and save through the normal NetOS APIs. Durable assistant memory is stored in `ai_memory_entries` and is limited to explicit user or organization preferences/rules, such as lines beginning with "remember..." or "always...".

The Admin > AI page also includes a **Customer Research Connector** for Sonar/MCP-style enrichment. Configure the MCP URL, tool name, optional authorization token, and enable the connector. NetOS calls this connector only from the backend, then converts the research result into a normal customer import preview row. Users must still review, edit, select, and commit the staged row through the standard import workflow.

The connector supports either an MCP SSE URL such as `http://mcp.xiberian.net:8081/sse?agent_id=netos-customer-research` or a direct HTTP endpoint that accepts MCP JSON-RPC `tools/call` requests. For SSE, NetOS opens the stream, reads the server-provided message endpoint, initializes the MCP session, and then calls the configured tool. The configured tool should accept a `query` argument and return either structured JSON with `draft`, `citations`, `warnings`, and `summary`, or text containing that JSON. Browser clients never receive the connector credential. If the MCP server expects `Authorization: <token>` rather than `Authorization: Bearer <token>`, paste only the raw token in the API key field; NetOS sends the value exactly as entered.

Use the **Test** button after entering connector values. The test initializes the MCP session and runs `tools/list`; it does not perform customer research or create import rows. A passing test means NetOS can reach the MCP server and the configured tool name appears in the advertised tool list.

Weekly Renewal Email Report

Admin users with `settings_view` / `settings_edit` can manage delivery from **Admin > Notifications**. This page controls:

- SMTP host, port, username, password, and sender
- public NetOS base URL used in email links
- weekly report enablement, day, UTC hour, and lookahead window
- test email delivery before saving or sending reports

Saved Admin notification settings override environment variables. Environment variables remain useful as bootstrap defaults and for deployments that have not yet saved a database-backed notification setting. SMTP passwords are encrypted server-side and are never returned to the browser; the UI only shows whether a password exists and its last four characters.

The Timeline page includes a **Weekly Renewal Email Report** section. It previews and manually sends an HTML report for circuits and infrastructure agreements whose renewal decision deadline is overdue or within the configured lookahead window. Each report includes:

- urgency state (`watch`, `active`, `critical`, `overdue`)
- agreement/vendor/type/status details
- MRC, term end, renewal notice, and renewal term
- links back to the circuit or infrastructure detail
- links to available agreement documents
- an HTML timeline-style snapshot for the report items

The API process also runs a lightweight scheduler. When SMTP is configured and the renewal report is enabled, it checks every 30 minutes and sends once on the configured UTC weekday/hour to all active registered users. Sends are recorded in `user_activity_logs` with `event_type=report`, so a container restart does not send the same report date twice.

Docker Network

- Server-side web calls (Next.js SSR) use `API_BASE_URL=http://api:8000` (Docker internal DNS)
- Browser-side calls use the UI host origin on port 8000

Authentication & RBAC

Current State (Development)

A **header-based shim** accepts the `x-user-role` header on all API requests:

```
curl -H "x-user-role: exec" http://localhost:8000/api/v1/circuits
```

No real identity verification occurs in direct/dev mode. All requests are trusted. When no SSO identity header is present, audit logs fall back to `dev@xiber.com` instead of `anonymous`.

Accepted identity headers include:

- `x-user-email`
- `x-auth-request-email`
- `x-forwarded-email`
- `x-forwarded-user`
- `cf-access-authenticated-user-email`

Production Target

Microsoft Entra OIDC JWT validation. Roles derived from Entra group membership. See Roadmap for implementation details.

Roles

ROLE	DESCRIPTION	WRITE ACCESS
`exec`	Executive — full access	Yes
`finance`	Finance team — financial fields, contracts	Yes
`network_eng`	Network engineering — technical fields, endpoints	Yes
`operations`	Operations — operational fields, lifecycle events	Yes
`pm`	Project management — project-related fields	Yes
`sales`	Sales — sales-related fields	Yes
`read_only`	View only	No
`agent`	MCP/XOS agent — scoped by tool definitions	Limited

Database

Custom PostgreSQL Image

Built from `pgvector/pgvector:pg16` with PostGIS installed.

Dockerfile: `infra/docker/postgres/Dockerfile`

Core Tables

TABLE	PURPOSE
`users`	User identity and role
`carriers`	Carrier/provider companies
`service_type_catalog`	Editable circuit service type catalog
`rf_system_catalog`	Editable RF equipment/system presets
`customers`	Commercial and MFC customer sites used by circuits, RF Links, and attribution
`endpoints`	Physical locations with PostGIS geometry
`circuits`	Wholesale circuit records
`rf_links`	Wireless backhaul and customer endlink records
`rf_link_documents`	FCC licenses and supporting RF Link documents
`infrastructure_assets`	Towers, rooftops, data centers, POPs, carrier hotels, offices, and aggregation sites
`infrastructure_documents`	Site agreements, floorplans, invoices, and supporting infrastructure documents
`subtended_links`	Explicit infrastructure/customer/circuit/RF attribution hierarchy
`electrical_services`	Electrical utility service accounts attached to infrastructure or customer sites
`electrical_service_documents`	Bills, invoices, agreements, meter photos, and supporting electrical service documents
`contracts`	Contract terms, renewal windows, ETF
`customer_attributions`	Revenue attribution (future Sonar link)
`lifecycle_events`	Timeline events (order, install, outage, etc.)
`invoices`	Carrier invoices
`invoice_line_items`	Invoice detail lines
`renewal_decisions`	Renewal state machine records
`outage_events`	Outage tracking
`sla_credits`	SLA credit recovery records
`import_jobs`	CSV/XLSX import metadata
`import_staging_rows`	Staged import rows before commit
`user_activity_logs`	Request/write audit activity

TABLE	PURPOSE
`feedback_requests`	Bug and feature requests
`feedback_comments`	Progress comments on bug/feature requests

Materialized View

- `circuit_financial_mv` — pre-computed financial rollups for dashboard performance

---

Documentation PDFs

Markdown documentation is the source of truth. Downloadable PDFs are generated from the Markdown files and published from `apps/web/public/docs`.

Run the exporter after material documentation changes:

```
python3 scripts/build_doc_pdfs.py
```

Generated files include:

PDF	PUBLIC URL
Complete bundle	`/docs/netos-documentation.pdf`
User Manual	`/docs/user-manual.pdf`
Admin & Operations	`/docs/admin-operations.pdf`
Data & Import Guide	`/docs/data-import-guide.pdf`
API & Integrations	`/docs/api-integrations.pdf`
Public Hostname & SSO	`/docs/public-hostname-sso.pdf`
Infrastructure Attribution	`/docs/infrastructure-attribution.pdf`
Roadmap & Known Gaps	`/docs/roadmap-known-gaps.pdf`

The exporter embeds public documentation images such as the infrastructure attribution and waterfall diagrams.

Extensions

- **PostGIS** — geometry columns for endpoint coordinates, GeoJSON queries
- **pgvector** — vector embeddings (installed, not yet used)

Migrations

Migration Files

Located in `apps/api/alembic/versions/`:

MIGRATION	DESCRIPTION
`0001_foundation_schema.py`	Core tables, enums, constraints
`0002_infrastructure_assets.py`	Infrastructure asset tables
`0003_agreement_extractions.py`	Agreement extraction tables
`0004_audit_feedback.py`	User activity logs and feedback requests
`0005_interface_requested_fields.py`	BAN, agreement, escalator, MSA/ETF fields
`0006_feedback_comments.py`	Progress comments for feedback requests

Commands

Apply all migrations:

```
# Via Docker
docker compose exec api alembic upgrade head

# Or standalone
cd apps/api
alembic upgrade head
```

Check current version:

```
docker compose exec api alembic current
```

Create a new migration:

```
docker compose exec api alembic revision --autogenerate -m "description"
```

Review auto-generated migrations before applying — they may need manual adjustments for PostGIS columns or custom types.

Seed Data

Script: `apps/api/scripts/seed_sample_data.py`

What it creates:

ENTITY	COUNT	EXAMPLES
Carriers	4	Lumen, Zayo, Cogent, Hurricane Electric
Endpoints	10	With coordinates for map rendering
Circuits	8	Various service types and statuses
Contracts	8	With term dates and renewal windows
Lifecycle Events	~15	Orders, installs, bandwidth changes

The script uses upsert logic on natural keys — safe to run repeatedly without creating duplicates.

Audit Trail

Audit data is available in the Admin view at `/admin`.

What Is Logged

EVENT TYPE	EXAMPLES
Request activity	API path, method, user, role, IP address, response status
Record writes	Create/update circuit, infrastructure, provider
Bulk actions	Per-record entries for bulk circuit and infrastructure updates/deletes
Feedback	Bug/feature request creation, status/priority changes, progress comments

Timestamps are stored in UTC in PostgreSQL and displayed in the UI as Eastern time with the timezone label.

Soft Deletes

Circuit and infrastructure bulk deletes set `deleted_at`. They do not physically remove rows. Normal list/map/dashboard views filter out soft-deleted rows.

Bug & Feature Queue

The Admin view includes a feedback form and queue.

FIELD	VALUES
Priority	`low`, `normal`, `high`, `urgent`
Status	`new`, `triaged`, `planned`, `in_progress`, `done`, `declined`

Admins can add progress comments to each request. Public comments attempt to email the requester when SMTP is configured. SMTP failures are non-blocking, so feedback updates still save if email is unavailable.

SMTP configuration is normally managed from **Admin > Notifications**. The variables below are bootstrap fallbacks before Admin settings are saved:

VARIABLE	DESCRIPTION
`SMTP_HOST`	SMTP server hostname
`SMTP_PORT`	SMTP port, default `587`
`SMTP_USERNAME`	Optional SMTP username
`SMTP_PASSWORD`	Optional SMTP password
`SMTP_FROM`	From address, default `netos@xiber.com`
`ADMIN_EMAIL`	Admin notification recipient

Verification Commands

Check API Health

```
curl -fsS http://localhost:8000/healthz
# → {"status": "ok"}
```

Check Map Data

```
curl -fsS http://localhost:8000/api/v1/circuits/map | python3 -m json.tool | head -20
```

Check Dashboard

```
curl -fsS http://localhost:8000/api/v1/dashboard/summary | python3 -m json.tool | head -20
```

Compile Check (Python)

```
python3 -m compileall apps/api/app apps/api/scripts apps/mcp/app
```

Build Check (Next.js)

```
cd apps/web && npm run build
```

Troubleshooting

Map Doesn't Load

CHECK	SOLUTION
API reachable?	`curl http://localhost:8000/api/v1/circuits/map`
CORS?	Ensure `http://localhost:3000` is in allowed origins
Data exists?	Run seed script, verify circuits have endpoint coordinates
Browser cache?	Hard refresh (Cmd+Shift+R / Ctrl+Shift+R)
Containers running?	`docker compose ps` — restart `api` and `web` if needed

Frontend 500 on Server-Rendered Pages

- Verify `API_BASE_URL=http://api:8000` is set in Docker Compose
- Verify the `api` container is running: `docker compose logs api`
- Check for Python exceptions in API logs

No Sample Circuits Showing

- Run the seed script (see above)
- Verify: `curl http://localhost:8000/api/v1/circuits` returns data
- Check migrations: `docker compose exec api alembic current`

PostGIS Extension Errors

- Rebuild the custom PostgreSQL image: `docker compose build postgres`
- Verify `infra/docker/postgres/Dockerfile` is referenced in `docker-compose.yml`
- Check: `docker compose exec postgres psql -U circuitos -c "SELECT PostGIS_Version();"`

Container Won't Start

```
docker compose logs <service>     # Check for errors
docker compose down -v            # Nuclear option: remove volumes and rebuild
docker compose up --build
```

Warning: `down -v` destroys all data. Re-run migrations and seed script after.