SyncTide

Developer Manual

Technical Reference & Architecture Guide

Version 1.4.1 — June 2026

Table of Contents

(Use Insert > Table of Contents in Word/LibreOffice to auto-generate)

1. Architecture Overview

1.1 Technology Stack

Component	Technology	Notes
Backend API	FastAPI (Python 3.13.13)	Uvicorn ASGI server — pinned to 3.13 until SCADA libs (asyncua, c104) ship cp314 wheels
Frontend UI	React + Vite (SPA)	Source in web/, built to web/dist, served by FastAPI via StaticFiles at /ui
Database	PostgreSQL 17 + TimescaleDB	psycopg2 adapter via SQLAlchemy; hypertable on measurements
Compilation	Cython	Backend source .py compiled to .pyd (Windows DLL) for distribution
Process Manager	NSSM (11 Windows services)	One service per worker; reverse-proxied by Caddy
Reverse proxy	Caddy	Serves the app + UI on the LAN (port 80)
Installer	Inno Setup	Pascal scripting for post-install
Reports	openpyxl + LibreOffice	Excel generation + PDF conversion
Messaging	requests + smtplib	HTTP APIs + SMTP, zero extra deps
Protocol drivers	pymodbus + asyncua + c104 + paho-mqtt	One dedicated worker process per protocol (Modbus TCP, OPC UA, IEC 60870-5-104, MQTT/Sparkplug B)

1.2 Project Structure

Path	Description
main.py	FastAPI app — all API endpoints, alarm detection, messaging escalation, serves web/dist at /ui
web/	React + Vite single-page app source (src/api.js holds the REST/bearer-auth adapter)

web/dist/	Built SPA bundle served by FastAPI via StaticFiles at /ui
i18n.py	Internationalisation — all UI strings in EN and PT-PT
settings.py	Configuration from environment variables / .env
security.py	Password hashing (PBKDF2), session management, role checks
messaging_engine.py	Abstract channel layer — Teltonika, Twilio, Telegram, SMTP, HTTP
report_template_engine.py	Excel template processing and chart generation
report_helpers.py	Data aggregation helpers for report generation
ingest_csvs.py	CSV file ingestion worker with duplicate detection
protocol_drivers/	Driver interface + Modbus TCP, OPC UA, IEC 60870-5-104, MQTT/Sparkplug B implementations
modbus_watchdog.py	Modbus TCP poller (SyncTideModbus service)
opcua_watchdog.py	OPC UA poller (SyncTideOPCUA service)
iec104_watchdog.py	IEC 60870-5-104 poller (SyncTideIEC104 service)
mqtt_watchdog.py	MQTT/Sparkplug B subscriber (SyncTideMQTT service)
alarm_watchdog.py	Real-time alarm evaluator (SyncTideAlarms service)
license_manager.py	Ed25519 license validation, tier enforcement
updater.py	Update package verification and installation
init_db.py	Database initialisation and migration runner
migrations/*.sql	Database migration scripts (numbered SQL)
setup_build.py	Cython build configuration

1.3 Request Flow

1. User interacts with the React single-page app in the browser
2. The SPA calls the FastAPI REST API with a bearer token (via web/src/api.js)
3. Requests reach FastAPI (through Caddy on the LAN, or directly on port 8000)
4. The FastAPI endpoint executes SQL via the SQLAlchemy engine against PostgreSQL/TimescaleDB
5. The response flows back: PostgreSQL -> FastAPI REST API -> React SPA -> browser

2. Development Environment Setup

2.1 Prerequisites

- Python 3.13.13 (Windows) — pinned; the installer ships an embedded copy
- PostgreSQL 17 + TimescaleDB
- Node.js + npm (for building the React/Vite frontend in web/)
- LibreOffice (for PDF report generation)
- Visual Studio Build Tools (for Cython compilation)

2.2 Installation

6. Clone the repository
7. Create virtual environment: python -m venv .venv
8. Activate: .venv\Scripts\activate
9. Install dependencies: pip install -r requirements.txt
10. Install build deps: pip install -r requirements-build.txt
11. Create .env file from .env.example
12. Create database: psql -f create_database.sql
13. Run migrations: handled automatically by init_db.py on first startup

2.3 Running Locally

Build the frontend once (or run the Vite dev server), then start the backend and workers:

```
# Frontend build (output goes to web/dist, served by the backend at /ui)
cd web && npm install && npm run build

# Backend API (also serves web/dist at /ui)
.venv/Scripts/python.exe -m uvicorn main:app --host 0.0.0.0 --port 8000

# Ingestion worker
.venv/Scripts/python.exe ingest_csvs.py
```

Note: Open the UI at http://localhost:8000/ui. For live frontend editing, run 'npm run dev' in web/ instead of a one-off build.

3. Cython Compilation

3.1 Build Workflow

14. Kill all Python processes: taskkill /F /IM python.exe /T
15. Compile: python setup_build.py build_ext --inplace
16. Restart all services

3.2 What Gets Compiled

The following backend files are compiled to .pyd (C extension DLLs):

- main.py, i18n.py, security.py, settings.py
- messaging_engine.py, report_template_engine.py, report_helpers.py
- license_manager.py, updater.py, ingest_csvs.py
- the protocol watchdogs and protocol_drivers/ modules
- init_db.py, init_admin.py, integrity.py, paths.py, version.py, platform_config.py

3.3 Frontend (No Cython)

- The React/Vite app in web/ is not Cython-compiled — it is built with 'npm run build' to web/dist and served by FastAPI at /ui. Rebuild the bundle after frontend changes; no backend restart is needed for static assets.

3.4 Common Gotcha

Cython only recompiles files that changed since the last build. If a .py file was fixed in a prior session but the .pyd wasn't rebuilt, use 'touch <file>.py' before running setup_build.py to force recompilation.

4. Database Schema

4.1 Migration System

Migrations are numbered SQL files in the migrations/ directory. The init_db.py module tracks applied migrations in the schema_migrations table and auto-applies pending ones on startup.

Migration	Description
001	Core tables: devices, measurements, source_files, tag_metadata
002	System tags, device tag mappings, device alarm rules
003	Users, sessions, RBAC
004	Reports, templates, scheduled jobs
005	Messaging: gateways, contacts, escalation lists, alarm rules, history
006	Alarm categories (hierarchical tree), alarm events
007	must_change_password on users
008	webhook_secret on messaging gateways
009	Messaging retry (retry_count, next_retry_at, last_error)
010	Multi-channel + ack (recipient_address, ack_token, ack_timeout_seconds)
011	require_ack flag on messaging alarm rules
012	Communication Lost alarm (nullable system_tag_id, comm_loss_alarm_enabled)
013-018	Tag types, digital/totalizer alarms, virtual tags, system-tag folders
019	Nullable alarm_category_id on devices
020	Multi-protocol ingestion: protocol_type, connection_config, device_tag_addresses
021+	Later migrations: MQTT/Sparkplug ingestion, TimescaleDB continuous aggregate, TOTP MFA + audit log, alarm one-open guards, report-job template-optional, and ongoing fixes (current schema version 67)

4.2 Key Tables

Table	Purpose
devices	Registered equipment with metadata and coordinates
measurements	Time-series measurement data (device_id, timestamp, tag, value)
system_tags	Standardised tag library (name + unit)
device_tag_mappings	Maps raw CSV tags to system tags per device
device_tag_addresses	Per-tag Modbus register / OPC UA NodeId map with data type, scale, offset
device_alarm_rules	Per-device alarm thresholds per tag
alarm_events	Detected alarm instances (active until cleared)
alarm_categories	Hierarchical device grouping tree
users	User accounts with roles and permissions
user_sessions	Active login sessions (hashed tokens)
messaging_gateways	Configured messaging channels
messaging_contacts	Notification recipients
messaging_escalation_lists	Ordered contact groups with parent chains
messaging_alarm_rules	Links device alarms to escalation lists
messaging_history	Sent message log with delivery status

5. API Reference

All endpoints are served at http://localhost:8000. Enable interactive docs by setting SYNCTIDE_ENABLE_DOCS=1 in .env then access /docs (Swagger UI) or /redoc.

5.1 Authentication

Method	Endpoint	Description
POST	/auth/login	Login with {username, password} — returns access_token
POST	/auth/logout	Revoke current session token
GET	/auth/me	Get current user profile

All authenticated endpoints require: Authorization: Bearer <token>

5.2 Device & Data Endpoints

Method	Endpoint	Description
GET	/devices	List all devices (filtered by user scope)
GET	/devices/{id}/tags	Get tags for a device
GET	/summary	System-wide metrics (totals)
GET	/measurements/query	Query raw measurements with filters
GET	/measurements/trend	Aggregated trend data for charting

5.3 Map & Alarm Endpoints

Method	Endpoint	Description
GET	/map/devices-status	Device status with alarm info for map
GET	/map/asset-tree	Hierarchical device tree with alarm counts
GET	/map/alarm-history/{device_id}	Alarm events for a device
POST	/map/alarm-acknowledge/{alarm_id}	Acknowledge a single alarm
POST	/map/alarm-acknowledge-all	Acknowledge all alarms globally

5.4 Admin Endpoints

Method	Endpoint	Description
GET/POST	/admin/users	List or create users
PUT/DELETE	/admin/users/{id}	Update or delete user
GET/PUT	/admin/equipment-metadata[/{id}]	Device metadata CRUD
GET/PUT	/admin/device-monitoring-config/{id}	Alarm rules + comm timeout
GET/POST/PUT/DELETE	/admin/system-tags[/{id}]	System tag CRUD
GET/POST/PUT/DELETE	/admin/alarm-categories[/{id}]	Category tree CRUD
GET/PUT	/admin/platform-config	Runtime configuration
POST	/admin/devices	Register a new device with its protocol + connection config
PUT	/admin/devices/{id}/comm-config	Update protocol, connection_config, poll_interval_seconds, polling_enabled
GET/PUT	/admin/device-tag-addresses/{id}	Per-tag register/NodeId map for Modbus TCP / OPC UA
POST	/admin/devices/{id}/test-connection	Probe the device without persisting anything

5.5 Messaging Endpoints

Method	Endpoint	Description
GET/POST/PUT/DELETE	/messaging/gateways[/{id}]	Gateway CRUD
GET/POST/PUT/DELETE	/messaging/contacts[/{id}]	Contact CRUD
GET/POST/PUT/DELETE	/messaging/escalation-lists[/{id}]	Escalation list CRUD
GET/POST/PUT/DELETE	/messaging/alarm-rules[/{id}]	Alarm rule CRUD
GET	/messaging/history	Message history with filters
POST	/messaging/test-send	Send a test message
POST/GET	/messaging/acknowledge/	Acknowledge via token

	{token}	
POST	/messaging/webhook/incoming	Delivery status webhook

5.6 Report Endpoints

Method	Endpoint	Description
POST	/reports/generate	Generate simple report
POST	/reports/generate-from-template	Generate template report
GET	/reports/{id}/download	Download generated report
GET/POST	/report-jobs[/{id}]	Scheduled job CRUD
POST	/report-jobs/{id}/run	Run scheduled job now
POST	/report-jobs/{id}/toggle	Enable/disable job

5.7 Rate Limiting

Endpoint	Limit	Window
/auth/login	10 requests per IP	60 seconds (burst)
/auth/login	30 requests per IP	900 seconds (sustained)
/auth/login	8 requests per username	300 seconds
/users/me/password	6 requests per user	300 seconds
/messaging/webhook/incoming	60 requests per IP	60 seconds

Rate-limited responses return HTTP 429 with a Retry-After header.

6. Security Architecture

6.1 Password Security

- PBKDF2-HMAC-SHA256 with 260,000 iterations
- 16-byte random salt per password
- Default password detection on startup — forces change on first login
- Password reuse and common-password rejection

6.2 Session Management

- Tokens generated via secrets.token_urlsafe(32)
- Stored as SHA-256 hashes (raw token never stored)
- Configurable session duration (1-12 hours)
- Automatic cleanup of expired sessions

6.3 Licence Signing

- Ed25519 signature scheme (separate keys for licences and updates)
- Public key embedded in application binary
- MAC address binding prevents licence transfer

6.4 API Security

- Interactive docs disabled by default in production
- Rate limiting on authentication endpoints
- Webhook authentication via HMAC secret comparison
- User-scoped data access (equipment whitelist enforcement)
- Installer sets ACL restrictions on sensitive files (.env, licence, manifest)

7. Deployment & Installation

7.1 Installer Build

17. Compile all Python modules: python setup_build.py build_ext --inplace
18. Generate integrity manifest: python tools/generate_manifest.py
19. Build installer: Inno Setup compiles installer/synctide.iss
20. Sign the installer (optional, recommended for production)

7.2 Client Installation

21. Run the SyncTide installer as Administrator
22. PostgreSQL is installed automatically (unattended)
23. The database is created and migrations are applied
24. Windows services are registered and started
25. Sensitive files are locked via ACL (icacls)

7.3 Environment Variables

Variable	Default	Description
SYNCTIDE_DB_HOST	localhost	Database host
SYNCTIDE_DB_PORT	5432	Database port
SYNCTIDE_DB_NAME	synctide_db	Database name
SYNCTIDE_DB_USER	synctide	Database user
SYNCTIDE_DB_PASSWORD	(required)	Database password
SYNCTIDE_API_PORT	8000	Backend API port (also serves the web UI at /ui)
SYNCTIDE_AUTH_SESSION_HOURS	12	Session duration (1-12)
SYNCTIDE_ENABLE_API_DOCS	0	Enable Swagger/ReDoc
SYNCTIDE_MAPBOX_API_KEY	(empty)	Optional satellite map tiles
SYNCTIDE_OPEN_BROWSER	1	Auto-open browser on start

7.4 Client Update Workflow

26. Stop services: Stop-Service SyncTideBackend -Force
27. Copy new .pyd files to the installation directory
28. Update integrity_manifest.json with new SHA-256 hashes
29. Apply any new database migrations
30. Start services: Start-Service SyncTideBackend

Note: For automated updates, use the built-in update package system (tools/build_update_package.py).

8. Troubleshooting

8.1 Common Development Issues

Issue	Cause	Fix
ImportError after code change	Stale .pyd not recompiled	touch <file>.py && python setup_build.py build_ext --inplace
Port already in use	Previous process not killed	taskkill /F /IM python.exe /T
Database connection error	PostgreSQL not running	net start postgresql-x64-17
UI shows 'Not Found' on login	web/dist built with VITE_API_BASE=/api but Caddy is down	Run behind Caddy, or rebuild web/dist with an empty VITE_API_BASE for same-origin
IntegrityError on INSERT	NOT NULL constraint	Check migration applied; ALTER column if needed
License invalid after MAC change	Network adapter changed	Re-generate license with new MAC addresses

8.2 Log Locations

Log	Path	Contents
Backend	logs/backend.log	API errors, request logs
Messaging	logs/messaging.log	Send attempts, delivery status
Ingestion	logs/ingestion.log	File processing, duplicate detection
Health Monitor	logs/health_monitor.log	Service health, restart events