



# SyncTide

## Developer Manual

Technical Reference & Architecture Guide

Version 1.6.0 — April 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 3.14)	Uvicorn ASGI server
Frontend UI	Streamlit 1.55	Pages in pages/ directory
Database	PostgreSQL 17	psycopg2 adapter via SQLAlchemy
Compilation	Cython	Source .py compiled to .pyd (Windows DLL)
Process Manager	Custom supervisor	Windows service with system tray
Installer	Inno Setup	Pascal scripting for post-install
Reports	openpyxl + LibreOffice	Excel generation + PDF conversion
Messaging	requests + smtplib	HTTP APIs + SMTP, zero extra deps

### 1.2 Project Structure

Path	Description
main.py	FastAPI app — all API endpoints, alarm detection, messaging escalation
app_core.py	Streamlit shared logic — navigation, auth gate, map, asset tree
api_client.py	HTTP client for Streamlit pages to call the backend API
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



<b>ingest_csvs.py</b>	CSV file ingestion worker with duplicate detection
<b>license_manager.py</b>	Ed25519 license validation, tier enforcement
<b>updater.py</b>	Update package verification and installation
<b>platform_supervisor.py</b>	Windows service manager for all SyncTide processes
<b>init_db.py</b>	Database initialisation and migration runner
<b>Home.py</b>	Streamlit entry point (home page)
<b>pages/*.py</b>	Streamlit page files (hot-reloaded, no compilation needed)
<b>migrations/*.sql</b>	Database migration scripts (001-012)
<b>setup_build.py</b>	Cython build configuration

### 1.3 Request Flow

1. User interacts with Streamlit UI (browser)
2. Streamlit page calls `api_client.py` functions
3. `api_client.py` makes HTTP requests to `localhost:8000` (FastAPI)
4. FastAPI endpoint executes SQL via SQLAlchemy engine
5. Response flows back: FastAPI -> `api_client` -> Streamlit -> browser



## 2. Development Environment Setup

### 2.1 Prerequisites

- Python 3.14+ (Windows)
- PostgreSQL 17
- 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

Start three processes:

```
# Backend API
.venv/Scripts/python.exe -m uvicorn main:app --host 0.0.0.0 --port 8000

# Frontend UI
.venv/Scripts/python.exe -m streamlit run Home.py --server.port 8501 --server.headless true

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

*Note: Use 'python -m streamlit' not 'streamlit.exe' when launching from bash.*



## 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 files are compiled to .pyd (C extension DLLs):

- `main.py`, `app_core.py`, `api_client.py`, `i18n.py`, `security.py`, `settings.py`
- `messaging_engine.py`, `report_template_engine.py`, `report_helpers.py`
- `license_manager.py`, `updater.py`, `platform_supervisor.py`, `ingest_csvs.py`
- `init_db.py`, `init_admin.py`, `integrity.py`, `paths.py`, `version.py`, `platform_config.py`

### 3.3 Hot-Reloaded (No Compilation)

- `Home.py` and all `pages/*.py` files — Streamlit hot-reloads them on change

### 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)

### 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_alarm_rules	Per-device alarm thresholds per tag



<b>alarm_events</b>	Detected alarm instances (active until cleared)
<b>alarm_categories</b>	Hierarchical device grouping tree
<b>users</b>	User accounts with roles and permissions
<b>user_sessions</b>	Active login sessions (hashed tokens)
<b>messaging_gateways</b>	Configured messaging channels
<b>messaging_contacts</b>	Notification recipients
<b>messaging_escalation_lists</b>	Ordered contact groups with parent chains
<b>messaging_alarm_rules</b>	Links device alarms to escalation lists
<b>messaging_history</b>	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	<code>/auth/login</code>	Login with {username, password} — returns access_token
POST	<code>/auth/logout</code>	Revoke current session token
GET	<code>/auth/me</code>	Get current user profile

All authenticated endpoints require: Authorization: Bearer <token>

### 5.2 Device & Data Endpoints

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

### 5.3 Map & Alarm Endpoints

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

## 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/{token}	Acknowledge via 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_urlsafes(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
<code>SYNCTIDE_DB_HOST</code>	localhost	Database host
<code>SYNCTIDE_DB_PORT</code>	5432	Database port
<code>SYNCTIDE_DB_NAME</code>	synctide_db	Database name
<code>SYNCTIDE_DB_USER</code>	synctide	Database user
<code>SYNCTIDE_DB_PASSWORD</code>	(required)	Database password
<code>SYNCTIDE_API_PORT</code>	8000	Backend API port
<code>SYNCTIDE_UI_PORT</code>	8501	Frontend UI port
<code>SYNCTIDE_AUTH_SESSION_HOURS</code>	12	Session duration (1-12)
<code>SYNCTIDE_ENABLE_API_DOCS</code>	0	Enable Swagger/ReDoc
<code>SYNCTIDE_MAPBOX_API_KEY</code>	(empty)	Optional satellite map tiles
<code>SYNCTIDE_OPEN_BROWSER</code>	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
<b>ImportError after code change</b>	Stale .pyd not recompiled	<code>touch &lt;file&gt;.py &amp;&amp; python setup_build.py build_ext --inplace</code>
<b>Port already in use</b>	Previous process not killed	<code>taskkill /F /IM python.exe /T</code>
<b>Database connection error</b>	PostgreSQL not running	<code>net start postgresql-x64-17</code>
<b>Streamlit won't start from bash</b>	streamlit.exe binding issue	Use: <code>python -m streamlit run Home.py</code>
<b>IntegrityError on INSERT</b>	NOT NULL constraint	Check migration applied; ALTER column if needed
<b>License invalid after MAC change</b>	Network adapter changed	Re-generate license with new MAC addresses

### 8.2 Log Locations

Log	Path	Contents
<b>Backend</b>	<code>logs/backend.log</code>	API errors, request logs
<b>Messaging</b>	<code>logs/messaging.log</code>	Send attempts, delivery status
<b>Ingestion</b>	<code>logs/ingestion.log</code>	File processing, duplicate detection
<b>Supervisor</b>	<code>logs/supervisor.log</code>	Service health, restart events