NAUTILUS

Maritime Compliance Platform

System Specification, Feature Set & Instructions Manual

Field	Value
Document Reference	NAUT-SPEC-001
Version	2.1.0
Classification	Internal — Restricted
Applicable Standards	IMO / EU Maritime Law
Date	03 May 2026
Stack	FastAPI + async SQLAlchemy 2.0 + PostgreSQL (asyncpg)
Backend Port	9000
Frontend Port	8084

Table of Contents

1. Executive Summary

NAUTILUS is a full-stack maritime regulatory compliance monitoring platform (v2.1.0) serving shipowners, ship management companies, classification societies, and flag state administrations. It continuously monitors vessel compliance against applicable IMO and EU maritime regulations, detects regulatory changes from official sources, automatically updates its rule catalogue, and dispatches targeted notifications to responsible personnel.

The system operates in two complementary modes:

- Vessel Compliance Mode — continuously evaluates real-time sensor and document telemetry from registered vessels against a comprehensive IMO/EU regulation catalogue, raising alarms and tracking lifecycle from detection through acknowledgement and resolution.
- Regulatory Intelligence Mode — autonomously scrapes official regulatory sources (IMO MEPC, IMO MSC, EUR-Lex, Paris MoU), detects changes to thresholds, effective dates, and applicability, and auto-applies confirmed changes to the live rule catalogue while dispatching daily digest notifications.

NAUTILUS is built on a FastAPI/PostgreSQL backend with a React SPA frontend, providing operators with a single pane of glass for fleet-wide regulatory risk.

2. System Architecture

2.1 Technology Stack

Component	Technology
Backend Framework	Python 3.12 — FastAPI (async) with Uvicorn ASGI server
Database	PostgreSQL 16 with SQLAlchemy 2.0 async ORM (asyncpg driver)
Scheduler	APScheduler 3.x (AsyncIOScheduler) — in-process background tasks
Auth	JWT Bearer tokens, bcrypt hashing, 5 role levels
Frontend	React SPA (Vite dev server) — Bearer auth wired
Backend Port	9000
Frontend Port	8084
Containerisation	Docker + Docker Compose with PostgreSQL service
Testing	pytest + pytest-asyncio — unit and integration tests

2.2 Architecture Overview

Layer	Responsibility
API Layer	FastAPI routers handle HTTP requests, validate input via Pydantic schemas, and delegate to services
Service Layer	Rule Engine, Notification Service, Scraper, Diff Engine, Auto-Apply, Digest Service
Task Layer	APScheduler tasks: daily fleet sweep (06:00 UTC), Regulatory Update Cycle, Daily Digest (07:00 UTC)

Layer	Responsibility
Model Layer	SQLAlchemy ORM models: Vessel, Voyage, FuelConsumption, EmissionsRecord, EUAPosition, Alert, Alarm, User, RegulatoryDocument, RegulatoryUpdate, RegulatoryDigest, Regulation, Contact
Database Layer	PostgreSQL 5432 with async connection pool (asyncpg driver)

2.3 Quick Start

2.3.1 Services

Both services run via systemd user services:

```
# Backend (port 9000)
systemctl --user status nautilus-backend.service
systemctl --user restart nautilus-backend.service

# Frontend (port 8084)
systemctl --user status nautilus-frontend.service
systemctl --user restart nautilus-frontend.service
```

2.3.2 Development

```
cd ~/hermes-research/new_apps/nautilus

# Backend
.venv/bin/python -m uvicorn app.main:app --host 0.0.0.0 --port 9000 --reload

# Frontend (separate terminal)
npm run dev -- --host 0.0.0.0 --port 8084
```

2.3.3 Default Admin Credentials

Field	Value
Email	admin@ingeniat.eu
Password	admin123
Role	superadmin

2.4 Environment Configuration

```
DATABASE_URL=postgresql+asyncpg://nautilus:<password>@localhost/nautilus
SECRET_KEY=<your-jwt-secret-min-32-chars>
ACCESS_TOKEN_EXPIRE_MINUTES=30
CORS_ORIGINS=http://localhost:3000,http://localhost:8084
```

2.5 Project Structure

Path	Description
app/main.py	FastAPI app + APScheduler
app/core/config.py	Settings (DATABASE_URL, JWT, CORS)
app/core/security.py	JWT + bcrypt + RBAC
app/db/session.py	Async engine (PostgreSQL or SQLite)
app/models/vessel.py	Vessel, Alarm
app/models/voyage.py	Voyage, VoyageLeg, FuelConsumption, EmissionsRecord, EUAPosition, Alert
app/models/user.py	User (5 roles)
app/models/regulatory_update.py	Sources, documents, updates, digests
app/models/regulation.py	Regulation catalogue
app/models/notification.py	Contacts, NotifyChannel
app/schemas/	Pydantic request/response schemas
app/api/v1/endpoints/	All API route handlers
app/calculations/emissions.py	Emissions calculator (454 LOC)
app/calculations/constants.py	Emission factors, CII parameters
app/services/alert_service.py	Async compliance alert generator
app/services/auto_apply.py	Regulatory update auto-apply
app/services/digest_service.py	Digest email/SMS builder
src/	React frontend
src/App.jsx	Auth gate + routes
src/services/api.js	API client (Bearer auth)
src/store/useStore.js	Zustand store
migrate_to_pg.py	SQLite to PostgreSQL migration script
SPEC.md	Full API specification
.env	DATABASE_URL, SECRET_KEY, etc.

3. Feature Set

Feature	Status
JWT Auth (5 roles)	✅
Vessel CRUD + Telemetry	✅
Voyage Management + Auto ETS Classification	✅
Fuel Consumption Tracking (BDN-grade)	✅

Feature	Status
Emissions Records (MRV/ETS/FuelEU/CII)	✅
Calculation Engine (ETS, FuelEU, CII, Voyage)	✅
Fleet Summary + ETS Exposure	✅
Alert Service (CII/ETS/FuelEU thresholds)	✅
Regulatory Updates + Digests	✅
Alarms Engine	✅
Dashboard Fleet Status	✅
APScheduler Fleet Sweep (daily 06:00 UTC)	✅
PostgreSQL Backend	✅
Frontend Auth + Dashboard	✅

4. Authentication & Authorization

4.1 Roles

Role	Permissions
superadmin	Full access
fleet_manager	Vessels, voyages, fuel, alarms, alerts
compliance_officer	Calculations, emissions, alerts, voyages
technical_superintendent	Vessels, fuel, telemetry
viewer	Read-only all endpoints

4.2 Auth Endpoints

Method	Endpoint	Description
POST	/api/v1/auth/token	Login (form-data: username, password). Returns JWT.
POST	/api/v1/auth/register	Register new user
GET	/api/v1/auth/me	Current user profile

4.3 Usage Example

```
TOKEN=$(curl -s -X POST http://localhost:9000/api/v1/auth/token \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "username=admin@ingeniat.eu&password=admin123" \
  | python3 -c "import sys,json; print(json.load(sys.stdin)['access_token'])")

# All subsequent requests:
```

```
curl http://localhost:9000/api/v1/vessels/ \
  -H "Authorization: Bearer $TOKEN"
```

5. API Reference

Base URL: http://localhost:9000. Auth: JWT Bearer token required on all requests except /auth/token. Interactive docs: http://localhost:9000/docs

5.1 Vessels

Method	Endpoint	Description
GET	/api/v1/vessels/	List active vessels
POST	/api/v1/vessels/	Create vessel
GET	/api/v1/vessels/{id}	Get vessel
PATCH	/api/v1/vessels/{id}	Update vessel
DELETE	/api/v1/vessels/{id}	Soft delete
POST	/api/v1/vessels/{id}/telemetry	Ingest compliance metrics

Vessel Model Fields

id (UUID), name, imo_number (unique), mmsi, callsign, vessel_type, gross_tonnage, dwt, year_built, flag_state, cii_rating, cii_attained, cii_required, eu_ets_allowance_balance, eu_ets_projected_annual_co2, fueleu_ghgi_actual, fueleu_ghgi_limit, fueleu_compliant, last_port, next_port, latitude, longitude, is_active, created_at, updated_at

5.2 Voyages

Method	Endpoint	Description
GET	/api/v1/voyages/	List voyages (filter: vessel_id, status, year)
POST	/api/v1/voyages/	Create voyage (auto-classifies EU ETS from LOCODEs)
GET	/api/v1/voyages/{id}	Get voyage
PATCH	/api/v1/voyages/{id}	Update voyage
DELETE	/api/v1/voyages/{id}	Cancel voyage
POST	/api/v1/voyages/{id}/complete	Complete voyage + background recalculation
POST	/api/v1/voyages/{id}/fuel	Add fuel consumption record
GET	/api/v1/voyages/{id}/fuel	List fuel records
PATCH	/api/v1/voyages/{id}/fuel/{fuel_id}	Update fuel record
DELETE	/api/v1/voyages/{id}/fuel/{fuel_id}	Delete fuel record

Method	Endpoint	Description
POST	/api/v1/voyages/{id}/legs	Create voyage leg
GET	/api/v1/voyages/{id}/legs	List voyage legs

> **NOTE:** EU ETS classification is auto-calculated from LOCODE country codes: intra-EU = 100%, extra-EU departure = 50%, extra-EU arrival = 50%, non-EU = 0%.

5.3 Emissions Records

Method	Endpoint	Description
GET	/api/v1/emissions/	List records (filter: vessel_id, year)
POST	/api/v1/emissions/	Create record
GET	/api/v1/emissions/{id}	Get record
PATCH	/api/v1/emissions/{id}	Update record
DELETE	/api/v1/emissions/{id}	Delete record

5.4 Fleet

Method	Endpoint	Description
GET	/api/v1/fleet/summary	Aggregated fleet compliance dashboard
GET	/api/v1/fleet/ets/exposure	Per-vessel EUA positions

5.5 Alerts

Method	Endpoint	Description
GET	/api/v1/alerts/	List alerts (filter: level, regulation, vessel_id, active_only)
GET	/api/v1/alerts/summary	Count by severity + unacknowledged
PATCH	/api/v1/alerts/{id}/acknowledge	Acknowledge alert
DELETE	/api/v1/alerts/{id}	Dismiss alert

Alert Thresholds

Metric	Critical	Warning
EUA shortfall	≥ 5,000 EUAs	≥ 1,000 EUAs
CII rating	E	D
FuelEU deficit	Penalty ≥ €500k	Deficit ≥ 1,000t VLSFO-eq

5.6 Alarms

Method	Endpoint	Description
GET	/api/v1/alarms/active	List active alarms
GET	/api/v1/alarms/	List all alarms
GET	/api/v1/alarms/{id}	Get alarm
POST	/api/v1/alarms/{id}/acknowledge	Acknowledge alarm
POST	/api/v1/alarms/{id}/resolve	Resolve alarm
POST	/api/v1/alarms/{id}/suppress	Suppress alarm

5.7 Dashboard

Method	Endpoint	Description
GET	/api/v1/dashboard/fleet-status	Fleet status (vessels, CII breakdown, alerts, pending updates)
GET	/api/v1/dashboard/rule-engine/checkers	Rule checker results

5.8 Calculations

All calculation endpoints are stateless — they run the emissions engine without touching the database.

Method	Endpoint	Description
POST	/api/v1/calculate/ets	EU ETS calculation
POST	/api/v1/calculate/fueleu	FuelEU Maritime calculation
POST	/api/v1/calculate/cii	CII AER + rating calculation
POST	/api/v1/calculate/voyage	Full voyage calculation (ETS + FuelEU + CII)
GET	/api/v1/calculate/factors	Emission factors reference

5.9 Regulatory Updates

Method	Endpoint	Description
GET	/api/v1/regulatory-updates/documents	List regulatory documents
GET	/api/v1/regulatory-updates/documents/{doc_id}	Get document
GET	/api/v1/regulatory-updates/updates	List all updates
GET	/api/v1/regulatory-updates/updates/pending	List pending updates awaiting approval
POST	/api/v1/regulatory-updates/updates/{id}/approve	Approve update
POST	/api/v1/regulatory-updates/updates/{id}/reject	Reject update
GET	/api/v1/regulatory-updates/summary	Summary counts by status

Method	Endpoint	Description
GET	/api/v1/regulatory-updates/digests	List digest records
POST	/api/v1/regulatory-updates/digests/send-now	Trigger digest send
GET	/api/v1/regulatory-updates/fetch-logs	List fetch audit logs
POST	/api/v1/regulatory-updates/trigger-fetch	Trigger source fetch

5.10 Regulations

Method	Endpoint	Description
GET	/api/v1/regulations/	List all regulations
GET	/api/v1/regulations/{id}	Get regulation

5.11 Contacts

Method	Endpoint	Description
GET	/api/v1/contacts/	List contacts
POST	/api/v1/contacts/	Create contact
DELETE	/api/v1/contacts/{id}	Delete contact
GET	/api/v1/contacts/notifications/log	Notification audit log

6. Enums Reference

Enum	Values
AlertLevel	critical | warning | advisory
AlertRegulation	EU_ETS | FUELEU | CII | MRV | IMO_DCS | GENERAL
VoyageStatus	planned | in_progress | completed | cancelled
FuelType	HFO | VLSFO | LSMGO | MGO | LNG | LPG | METHANOL | ETHANOL | B30 | B100 | AMMONIA | HYDROGEN
VoyageClassification	intra_eu | extra_eu_departure | extra_eu_arrival | non_eu
UpdateStatus	fetched | parsed | diffed | applied | failed | superseded
ApprovalState	pending | approved | rejected | auto
ChangeType	new_regulation | threshold_change | effective_date | applicability | revocation | amendment
DigestStatus	pending | sent | failed
NotifyChannel	EMAIL | SMS | WHATSAPP | TELEGRAM | DISCORD

7. Database

7.1 Connection

Parameter	Value
Host	localhost:5432
Database	nautilus
User	nautilus
Driver	asyncpg
Connection string	postgresql+asyncpg://nautilus:<password>@localhost/nautilus

7.2 PostgreSQL Migration (Completed 2026-05-03)

Full SQLite to PostgreSQL migration has been executed. All data preserved. 15 tables migrated:

users, vessels, voyages, fuel_consumptions, emissions_records, alerts, eua_positions, voyage_legs, regulatory_documents, regulatory_digests, source_fetch_logs, regulations, contacts, regulatory_updates, alarms

Schema Adaptations

- UUID primary keys stored as TEXT (not native PG UUID type)
- All Enum columns use native_enum=False — stored as VARCHAR in PostgreSQL
- Foreign key constraints re-added after migration
- 12 PostgreSQL ENUM types created but not used by app (kept for reference)

> **NOTE:** Ensure native_enum=False is set on all Enum columns. See troubleshooting section for enum error resolution.

Re-running the Migration

```
cd ~/hermes-research/new_apps/nautilus
.venv/bin/python migrate_to_pg.py
```

8. Scheduled Jobs

Job	Schedule	Description
fleet_sweep	Daily 06:00 UTC	Evaluates all vessels for compliance alerts (CII/ETS/FuelEU thresholds). Deduplicates active alerts.
regulatory_update_cycle	Every 6 hours	Runs all four source scrapers, diff engine, and auto-apply.
daily_digest	Daily 07:00 UTC	Dispatches HTML email + SMS digest of 24h changes to all active contacts.

9. Regulations Covered

NAUTILUS ships with a pre-seeded catalogue of 16 IMO/EU regulatory frameworks covering the full spectrum of maritime compliance requirements. Each regulation includes a dedicated checker in the Rule Engine, threshold definitions, and applicability filters by vessel type.

9.1 IMO MARPOL Regulations

Code	Reference	Description
MARPOL_SOX_GLOBAL	Annex VI Reg. 14 / MEPC.328(76)	Sulphur oxide emissions: 0.50% m/m global, 0.10% in ECAs. Scrubber equivalence accepted.
MARPOL_NOX_TIER3	Annex VI Reg. 13	NOx Tier III: ≤3.4 g/kWh for new engines in NOx ECAs (Baltic, North Sea, North American).
MARPOL_SEWAGE	Annex IV Reg. 11-12	Sewage treatment plant compliance and discharge restrictions in Special Areas (cruise ships).
MARPOL_GARBAGE	Annex V Reg. 3-6	Garbage Management Plan; total prohibition on plastic discharge at sea.
BWM_D2	BWM Convention / D-2 Standard	Ballast Water Treatment System active; ≤10 viable organisms/m^3 ≥50µm.
AFS_CONVENTION	AFS Convention 2001	TBT-free anti-fouling coatings on hull; IAFS Certificate required.

9.2 IMO Energy Efficiency & GHG Regulations

Code	Reference	Description
CII_RATING	MEPC.339(76) / MEPC.354(78)	Carbon Intensity Indicator: annual A-E rating. D/E requires corrective action plan within 1 year.
EEXI	MEPC.333(76)	Energy Efficiency Existing Ship Index: one-time attained EEXI must not exceed required EEXI.

9.3 EU Maritime Regulations

Code	Reference	Description
EU_MRV	EU 2015/757 (amended)	Annual CO_2 monitoring, reporting and verification for ships >5,000 GT calling EU ports.
EU_ETS	EU 2003/87/EC (Maritime from 2024)	EU Emissions Trading System: phase-in 40% EUA (2024), 70% (2025), 100% (2026+).
FUELEU_MARITIME	EU 2023/1805	GHG intensity of marine fuels: ≤91.16 gCO_2eq/MJ from 2025; reduces to 2030 and 2050 targets.

9.4 Safety, Labour & Port State Control

Code	Reference	Description
SOLAS	SOLAS Ch. II-2 / III / FSS Code	Fire safety, life-saving appliances, structural fire protection.
ISM_CODE	SOLAS Ch. IX / Res. A.741(18)	Safety Management System: DOC, SMC, annual internal audits, near-miss reporting.
MLC_2006	ILO MLC 2006 / EU Dir. 2009/13	Seafarer rights: min. 10h rest/24h, 77h/week; medical certification; MLC Certificate.
PARIS_MOU	Paris MoU 1982 / NIR	Port State Control risk profiling; targeting factor from deficiency history.
IGC_CODE	MSC.370(93)	Gas carrier design and safety: ESD, gas detection, cargo containment integrity (LNG-specific).
MODU_CODE	IMO MODU Code 2009	Mobile offshore unit structural and safety certification (Offshore-specific).

10. Alarm Lifecycle

State	Description
ACTIVE	Alarm raised by the rule engine. Notification dispatched. Dashboard displays with indicator.
ACKNOWLEDGED	Operator has acknowledged the alarm via API or dashboard. Escalation timer is paused. No further notifications.
ESCALATED	Alarm remains unacknowledged after the escalation threshold (default: 30 minutes). Escalation count incremented.
SUPPRESSED	Operator suppresses re-notification for N minutes (5-1440). Alarm remains visible but silent.
RESOLVED	Rule engine detects compliance restored, or operator manually resolves. Alarm archived.

11. Troubleshooting

11.1 Backend Won't Start After Code Changes

```
fuser -k 9000/tcp
systemctl --user restart nautilus-backend.service
```

11.2 Backend Failing on Enum Errors

Ensure native_enum=False is set on all Enum columns. All enum columns must be stored as VARCHAR in PostgreSQL.

11.3 Frontend Not Connecting to Backend

Check CORS_ORIGINS includes http://localhost:8084. Check backend is running on port 9000.

11.4 Port Already in Use

```
fuser -k 9000/tcp 8084/tcp
systemctl --user restart nautilus-backend.service nautilus-frontend.service
```

11.5 Alarm Engine Not Raising Alarms

- Verify telemetry has been ingested via POST /vessels/{id}/telemetry
- Check the in-memory cache is populated — restart resets it
- Check logs for 'no_telemetry' entries at DEBUG level
- Verify the vessel is_active=true and the regulation checker applies_to() returns true for the vessel type

11.6 Database Connection Errors

- Verify DATABASE_URL is correct and PostgreSQL is running
- Check pool size settings — reduce DATABASE_POOL_SIZE if hitting connection limits
- Run: python -m app.db.seed to reinitialise tables if schema is out of sync

12. Extending the System

12.1 Adding a New Regulation Checker

The Rule Engine uses a self-registering plugin pattern. To add a new regulation:

- Open app/services/rule_engine.py
- Add any new telemetry fields to the VesselData dataclass if required
- Create a new checker class inheriting from BaseChecker with @register_checker decorator
- Add a seed entry to app/db/seed.py for the new regulation code
- Add the regulation code to EXTERNAL_ID_MAP in app/services/diff_engine.py if it maps to an external document ID
- Restart the server — the checker auto-registers and is active on the next alarm cycle

> **TIP:** No changes are required to the API, scheduler, or notification system. The checker is automatically discovered and run against all applicable vessels.

12.2 Adding a New Regulatory Source Scraper

- Create a new scraper class in app/services/scrapers.py
- Add a KNOWN_DOCUMENTS catalogue as a class attribute for fallback
- Implement async fetch(self) -> List[ScrapedDocument]

- Add a new value to the RegulatorySource enum in app/models/regulatory_update.py
- Register in SOURCE_SCRAPERS dict at the bottom of scrapers.py
- Add entries to EXTERNAL_ID_MAP and MONITORED_SOURCES as appropriate

13. Regulatory Compliance Notes

IMPORTANT: NAUTILUS does not constitute legal advice and does not replace the official regulatory compliance obligations of the shipowner, operator, or management company. All alarm outputs and regulatory update notifications must be reviewed and acted upon by qualified Designated Persons Ashore (DPA) and technical superintendents.

- Threshold values in the regulation catalogue are maintained based on publicly available IMO resolutions and EU Official Journal publications. Operators should verify all auto-applied threshold changes against the original source document before relying on them for compliance decisions.
- CII ratings are calculated from telemetry data. The actual IMO DCS-submitted CII rating may differ due to voyage corrections, fuel quality adjustments, or flag state specific guidance. NAUTILUS CII calculations are indicative only.
- EU ETS surrender percentages must be verified against the verified MRV emissions report submitted to the administering authority. NAUTILUS tracks the percentage field as provided by the operator.
- FuelEU Maritime GHG intensity calculations include Well-to-Wake (WtW) emission factors per EU 2023/1805 Annex II. Operators using non-standard fuels should verify the applicable emission factors.
- The alarm escalation feature (30-minute default) is a monitoring aid only. It does not fulfil the escalation procedural requirements of the ISM Code, which require documented procedures, designated persons, and flag state notifications.

14. Glossary

Term	Definition
AER	Annual Efficiency Ratio — metric used in CII calculation (gCO_2 / capacity × distance)
APScheduler	Advanced Python Scheduler — the library used for background task scheduling in NAUTILUS
BWTS	Ballast Water Treatment System — equipment that treats ballast water to meet the D-2 biological standard
CII	Carbon Intensity Indicator — annual operational efficiency rating A (best) to E (worst) per MEPC.339(76)
DCS	Data Collection System — IMO mandatory fuel oil consumption reporting system
DPA	Designated Person Ashore — shore-based person responsible for ISM Code compliance
ECA	Emission Control Area — designated sea area with stricter SOx (0.10%) and/or NOx limits
EEXI	Energy Efficiency Existing Ship Index — one-time design efficiency requirement for

Term	Definition
	existing ships per MEPC.333(76)
ESD	Emergency Shutdown — system that safely stops cargo operations on LNG carriers
EUA	EU Allowance — unit of CO_2 used in the EU Emissions Trading System (1 EUA = 1 tonne CO_2)
GHG	Greenhouse Gas — includes CO_2, CH_4 (methane), N_2O for WtW accounting under FuelEU Maritime
ISM Code	International Safety Management Code — mandatory safety management framework under SOLAS Chapter IX
MEPC	Marine Environment Protection Committee — IMO committee responsible for environmental regulations
MLC 2006	Maritime Labour Convention 2006 — ILO convention setting minimum standards for seafarer working conditions
MSC	Maritime Safety Committee — IMO committee responsible for safety regulations
MRV	Monitoring, Reporting and Verification — EU CO_2 reporting framework under EU 2015/757
NIR	New Inspection Regime — Paris MoU risk-based PSC inspection targeting system
OWS	Oily Water Separator — equipment that separates oil from bilge water; must maintain <15ppm oil content
PSC	Port State Control — inspection regime by which port states verify foreign vessels comply with international conventions
SEEMP	Ship Energy Efficiency Management Plan — mandatory operational plan under MARPOL Annex VI
SOx	Sulphur Oxides — combustion products of sulphur in marine fuel; limited by MARPOL Annex VI
STP	Sewage Treatment Plant — equipment that treats sewage to MARPOL Annex IV discharge standards
WtW	Well-to-Wake — full lifecycle GHG emission factor used in FuelEU Maritime calculations

15. Document History

Version	Date	Author	Changes
1.0.0	28/02/2026	NAUTILUS Project Team	Initial release — full system specification covering compliance monitor, regulatory intelligence module, API reference, and operator manual
2.1.0	03/05/2026	NAUTILUS Project Team	Updated to v2.1.0: added React frontend, PostgreSQL migration, JWT auth with 5 roles, Voyage Management with auto ETS classification, Fuel Consumption (BDN-grade), Emissions Records (MRV/ETS/FuelEU/CII), Calculation Engine, Fleet Summary, Alert Service, APScheduler fleet sweep, full API reference update