

Software Dev 2025

Technology Stack

FrontEnd

React, Leaflet.js, WebSocket API

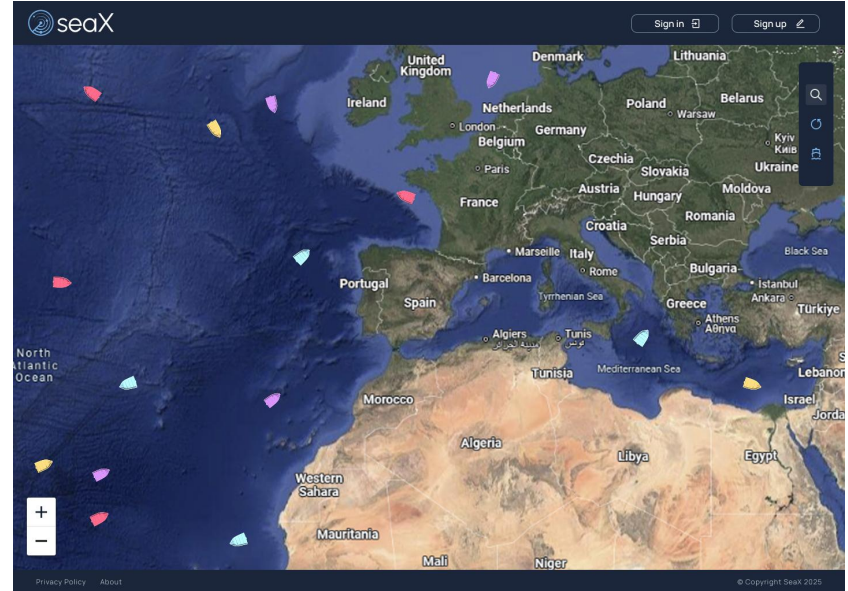
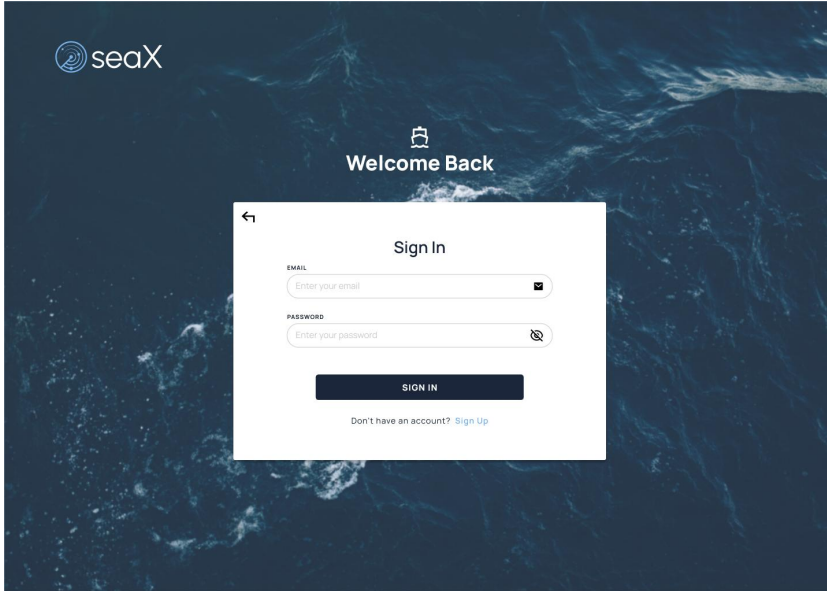
BackEnd

Java-Spring Boot, Apache Kafka, PostgreSQL, PostGIS, Redis


DevOps


GitHub actions, Docker

Login page, Guest Home page






























The screenshot displays the seaX web application. The main map area shows a geographical view of Europe, North Africa, and the Middle East. A semi-transparent blue rectangle labeled 'ZONE OF INTEREST' is positioned in the Atlantic Ocean. Several colored markers (yellow, pink, red, blue, cyan) are scattered across the map, primarily in the Atlantic and around the Mediterranean coast. A sidebar on the right side of the map contains a vertical stack of navigation icons, including a magnifying glass, a heart, a list, a home icon, a location pin, and a triangle. At the top right, there is a text input field labeled 'Name Surname'. At the bottom left, there is a zoom control with '+' and '-' buttons. At the bottom center, there is a 'Privacy Policy' link and the text 'About'. At the bottom right, there is a copyright notice: '© Copyright seaX 2025'.


seaX

Name Surname


Vessels

Vessel Name 	Country 	MMSI 	Type 	Destination 	
NEW ENERGY 	 HK	477439400	 Oil Products Tanker	ARZEW ALGERIA	<div>Past Track</div> <div> My Fleet</div>
NEW ENERGY	 HK	477439400	 Oil Products Tanker	ARZEW ALGERIA	<div>Past Track</div> <div> My Fleet</div>
NEW ENERGY	 HK	477439400	 Oil Products Tanker	ARZEW ALGERIA	<div>Past Track</div> <div> My Fleet</div>
NEW ENERGY	 HK	477439400	 Oil Products Tanker	ARZEW ALGERIA	<div>Past Track</div> <div> My Fleet</div>
NEW ENERGY	 HK	477439400	 Oil Products Tanker	ARZEW ALGERIA	<div>Past Track</div> <div> My Fleet</div>
NEW ENERGY	 HK	477439400	 Oil Products Tanker	ARZEW ALGERIA	<div>Past Track</div> <div> My Fleet</div>

< Previous

1 2

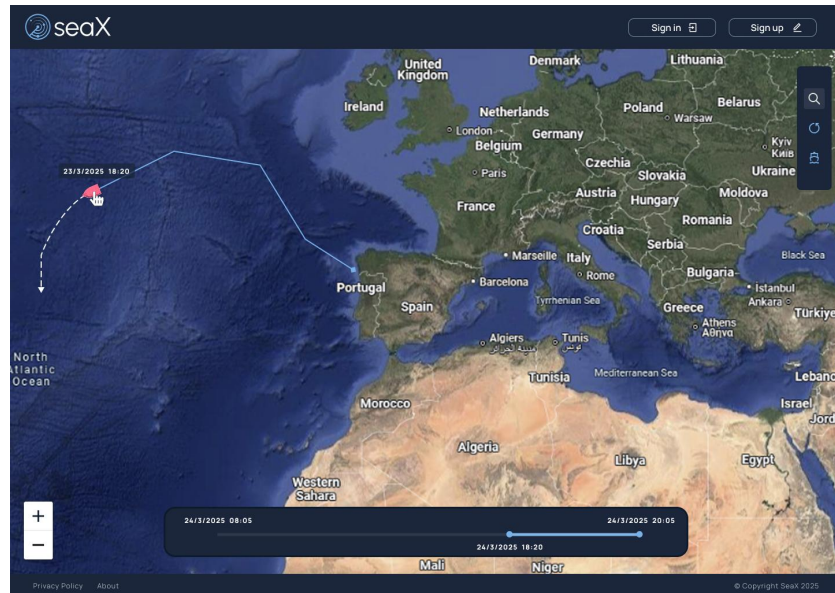
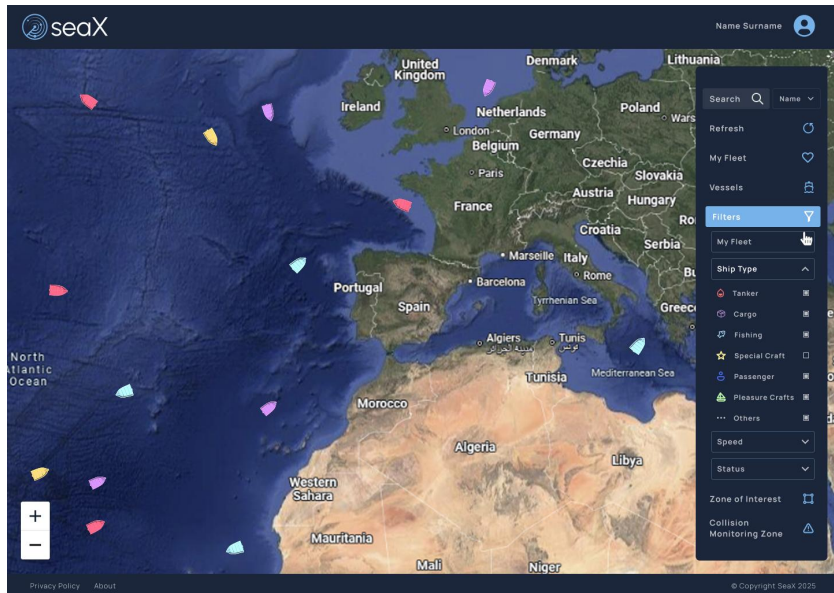
Next >

Privacy Policy

About

© Copyright SeaX 2025

Filters, History



Front End - Implementation

User Management

- Forms for registration, login, password reset
- Profile view for editing / deleting
- UI elements hidden/shown based on user role fetched from backend.

Front End - Implementation

Real-Time Ship Tracking

- Map component displaying ship icons.
- WebSocket connection to receive real-time position updates.
- Popup to show detailed ship info when a ship is clicked.
- Filtering controls (dropdowns, sliders) for speed, type, status.
- "My Fleet" view (fetches saved ships for the user).

Historical Data & Search

- Search bar (by Name, MMSI).
- Results list/table, sortable.
- Option to view historical track (last 12h) for a selected ship on the map.
- Clicking a search result highlights/centers the ship on the map.

Front End - Implementation

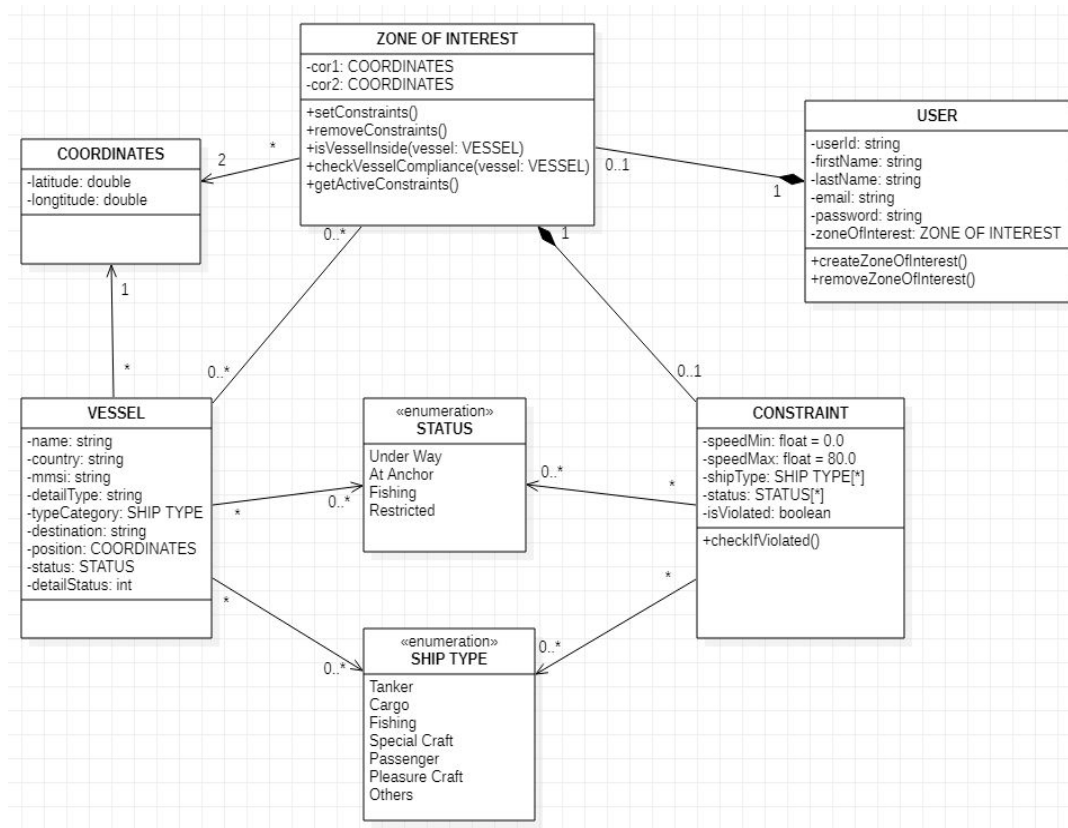
Zones of Interest & Alerts

- Map interface to draw polygonal zones (google.maps.drawing.DrawingManager).
- Form to define rules for the zone (e.g., Max Speed: 10 knots, Allowed Types: Cargo, Tanker, Status: Anchored).

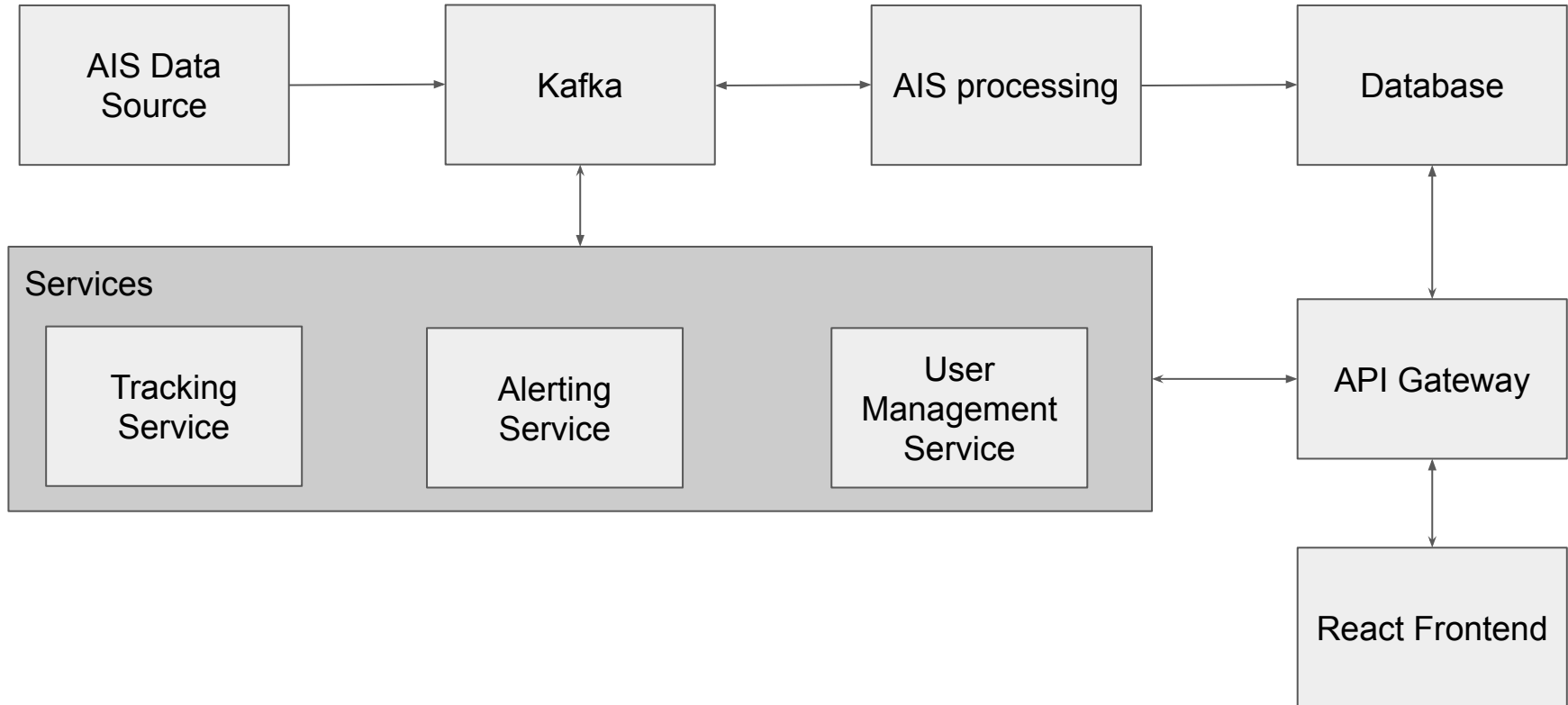
Collision Monitor Zones

- Similar interface to Zones of Interest for drawing zones.

Class Diagram - Zones of Interest



Backend High-Level Architecture Overview



AIS Data Processing and Storage

Ingestion

AIS Feed, Kafka messages, ais.raw

Backend Processing

@KafkaListener, data validation, java object mapping

Database Storage

- PostgreSQL, TimescaleDB
- Tables: ship_positions, ship_static_data, ship_voyage_data

Backend - Ship Tracking

REST API endpoints

`/api/ships/current` , `/api/ships/details/{mmsi}`, `/api/ships/fleet`

Websocket

- `/ws/tracking`
- Push real-time ship position updates
- Consume (ship.position.updated topic) from `@KafkaListener`
- Broadcast general updates or client subscriptions

Backend - Historical Data & Search

REST API endpoints

`/api/ships/search?query=...` , `/api/ships/history/{mmsi}?hours=...`

Database

- `ship_static_data`: name, mmsi
- `ship_positions`: compound index: (mmsi, timestamp)

TimescaleDB

Timestamp in the `ship_positions` hypertable

Backend - Zones of Interest & Alerts

REST API endpoints

/api/zones/interest , POST-GET-DELETE

Database

- zones_of_interest (id, user_id, name, geom GEOMETRY())
- PostGIS spatial index (geom column)

Rule Engine Logic

- Consume kafka ship.position.updated
- On rule violations generate alerts

Backend - Collision Monitor Zones

REST API endpoints

/api/zones/collision , POST-GET-DELETE

Database

- Collision_monitor_zones (id, user_id, name, geom GEOMETRY())
- PostGIS spatial index (geom column)

Collision Detection

- Consume kafka ship.position.updated
- Collision detection logic
- Publish message to kafka , alert.collisions

Modifications over previous plans

Database

PostgreSQL with TimescaleDB

API Gateway

Centralize concerns like routing, authentication and frontend interaction

Thank you for your time!

