Gabriel Gostiaux
Hamza Khitous
IOGS 3A

Wind-Watch

Providing wind-optimized sailing courses, on the go

https://github.com/GabrielGst/winged-watch

Summary

- Project
- Bill of specifications
- Design & integration
- Algorithm
- Installation & usage
- Testing
- Perspectives

Project

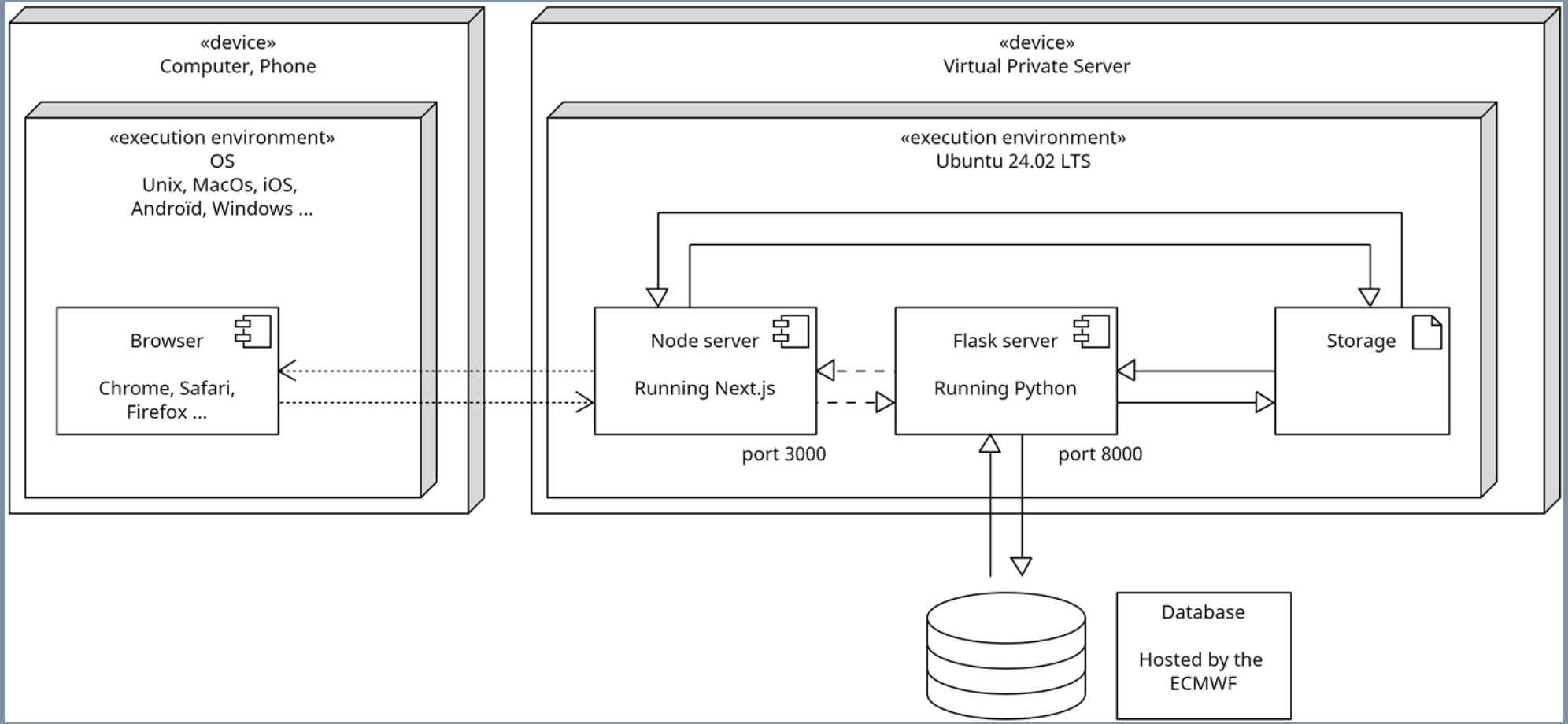
Aims to provide real-time meteorological forecasts and sailing course optimization.

- Real-time data monitoring
- Advanced analytics and reporting
- Scalable and flexible architecture
- User-friendly interface

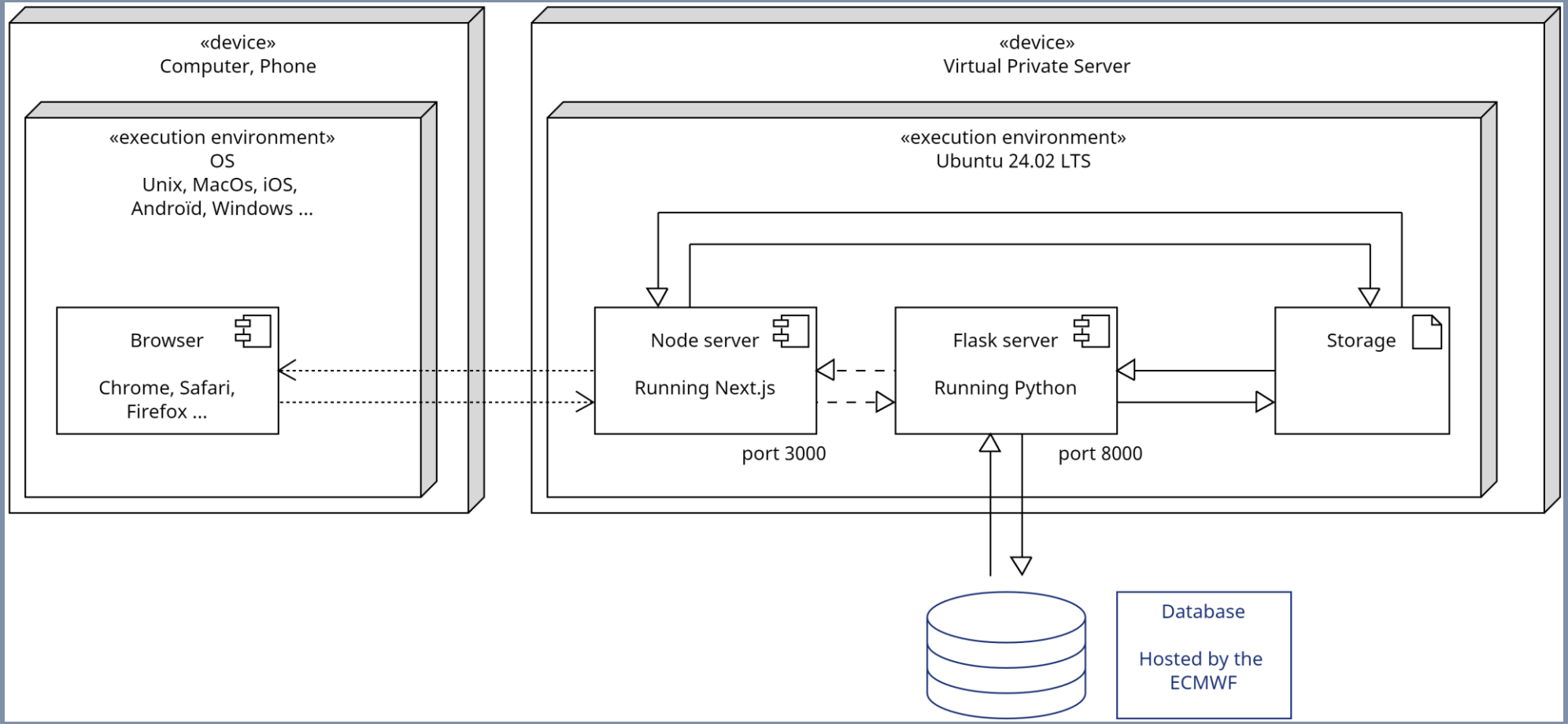
Specifications

1. the app should use (and show) the bathymetric map at different scales,	Possible using raster maps
2. the app should use (and show) the wind forecast,	100%
3. the app should use (and show) the ocean currents forecast,	Abandonned
4. the app should use (and show) the tidal heights forecast,	Possible with ECMWF
5. the app should use (and show) basics marine beacons,	Possible in some areas
6. the app should computes (and show) the optimal sailing course,	100%
7. the app should provide a list of timed directions to follow the computed course.	Not implemented yet

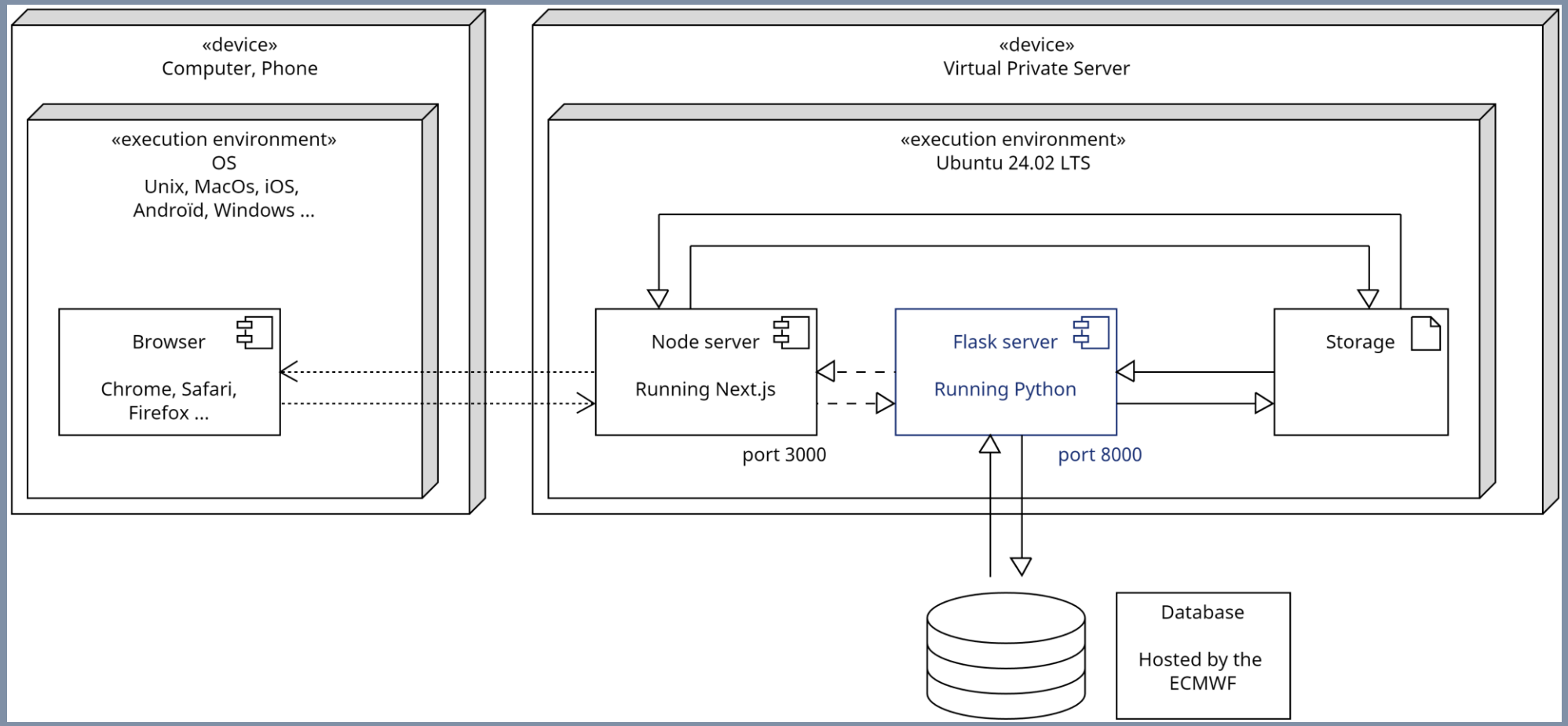
Network structure



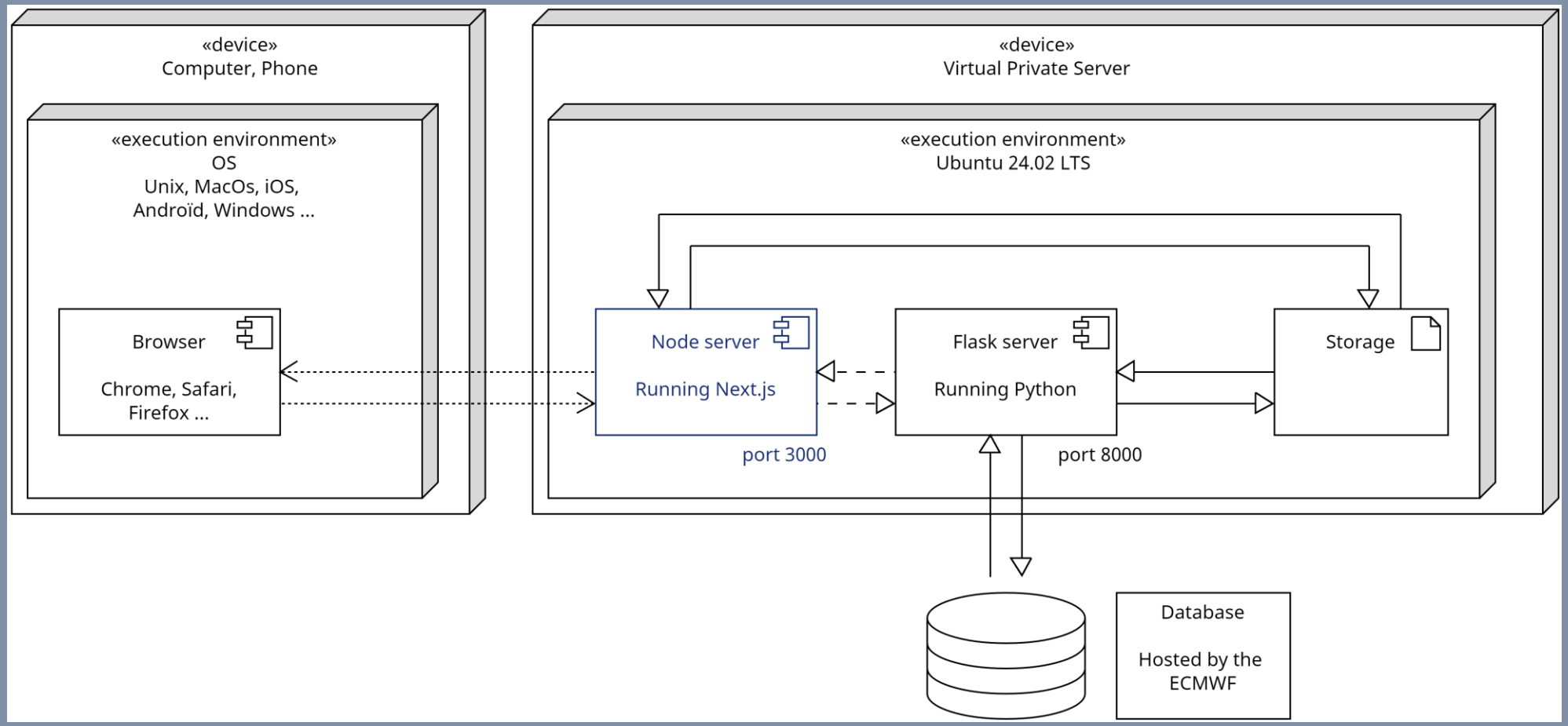
Database


«device»
Computer, Phone
«execution environment»
OS
Unix, MacOs, iOS,
Androïd, Windows ...
Browser
Chrome, Safari,
Firefox ...
«device»
Virtual Private Server
«execution environment»
Ubuntu 24.02 LTS
Node server
Running Next.js
port 3000
Flask server
Running Python
port 8000
Storage
Database
Hosted by the
ECMWF

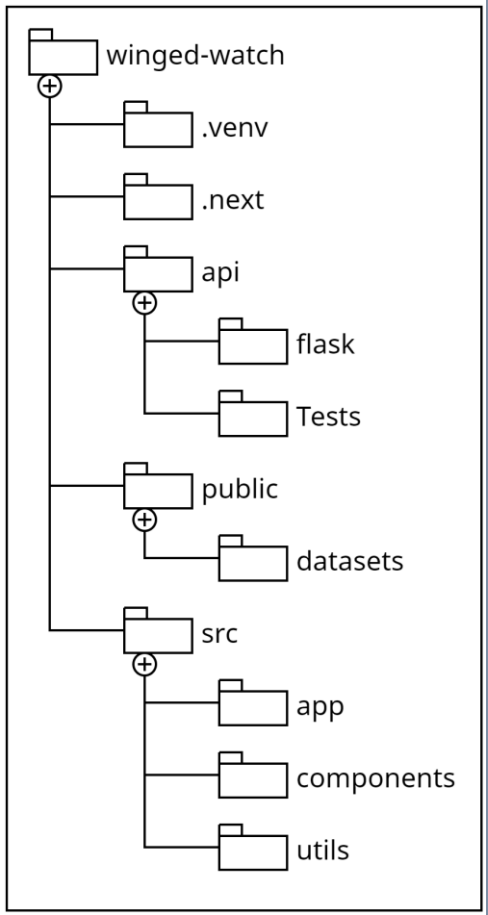
Flask (python) server



Node (Nextjs) server

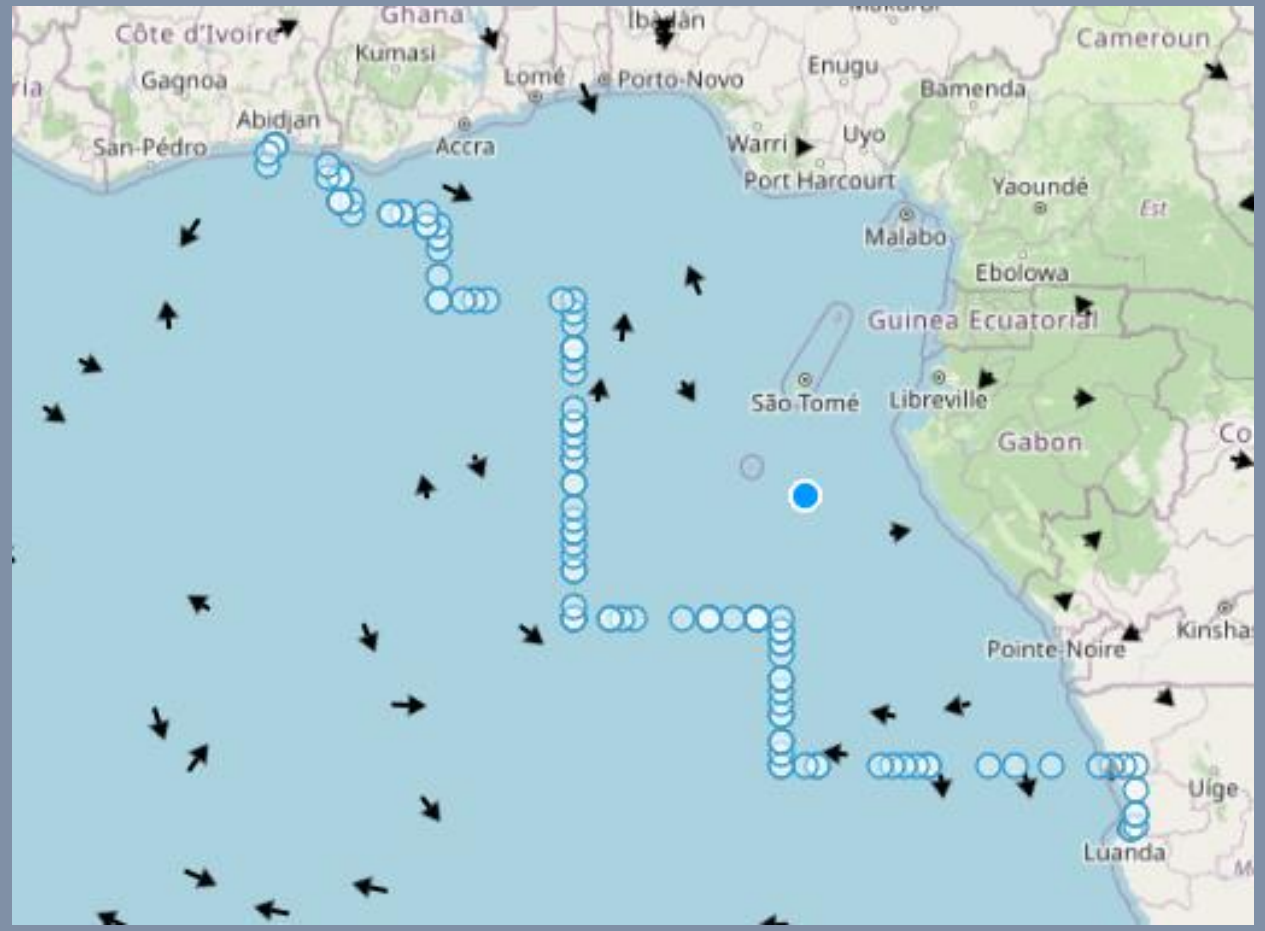


File structure




```
.
├── README.md                   # README file
├── .github                     # GitHub folder
├── .venv                       # Python virtual environment
├── .next                       # next cache (created on build)
├── .vscode                     # VSCode configuration
├── api                         # Python flask server & dependencies
│   ├── requirements.txt        # Python requeriments
│   ├── index.py                # Python main
│   └── Tests                   # Unit python test functions
├── public                      # Public assets folder
├── src
│   ├── app                     # Next JS App (App Router)
│   ├── components              # React components
│   └── utils                   # Utilities folder
├── tailwind.config.js          # Tailwind CSS configuration
└── tsconfig.json               # TypeScript configuration
```

Computing optimized courses



Requirements

Requirements

Python

- Python 3.12+
- Required Libraries: see `requirements.txt`

Node.js

- Node.js 20+
- `npm` & `pnpm`
- Required Packages: see `package.json`

Installing and using the app

Installation

This is a Next.js project bootstrapped with `create-next-app`.

Run the following command on your local environment:

```
git clone --depth=1 https://github.com/GabrielGst/winged-watch.git my-project-name
cd my-project-name
npm install
```

First, run the Flask intall:

```
npm run flask-install
```

Second, run the development server:

```
npm run dev
# or
yarn dev
# or
pnpm dev
# or
bun dev
```

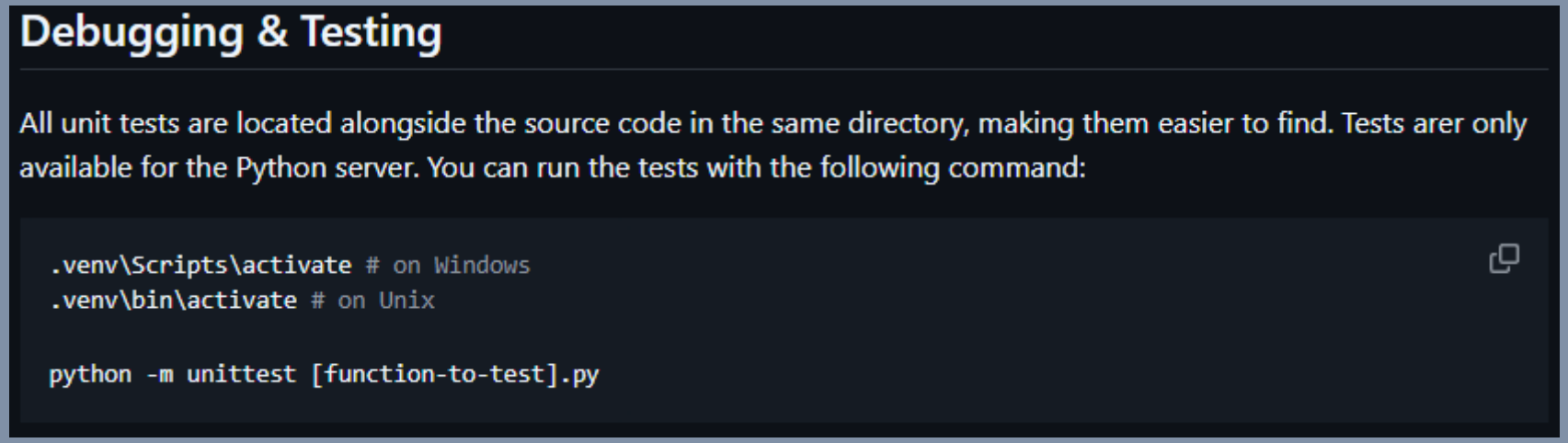
Open http://localhost:3000 with your browser to see the result.

Logs

```
[2025-02-13 10:22:29.790119] Processing to daily retrieve.
[2025-02-13 10:22:29.790119] Retrieving and processing forecast data. Export: False
[2025-02-13 10:22:29.790119] Proceeding with forecast retrieval.
[2025-02-13 10:22:31.367544] Response [2]. The weather dataset file time 2025-02-11 01:00:00 and current time 2025-02-13 10:22:31.367544 are on different days. Proceeding with
[2025-02-13 10:22:32.398865] Processing to daily retrieve.
[2025-02-13 10:22:32.398865] Retrieving and processing forecast data. Export: False
[2025-02-13 10:22:32.398865] Proceeding with forecast retrieval.
[2025-02-13 10:22:33.705214] Response [2]. The weather dataset file time 2025-02-11 01:00:00 and current time 2025-02-13 10:22:33.705214 are on different days. Proceeding with
[2025-02-13 10:23:40.706649] Latest forecast available : 2025-02-13 00:00:00
[2025-02-13 10:23:41.685089] Forecast downloaded : <ecmwf.opendata.client.Result object at 0x0000018EA338B740>
[2025-02-13 10:23:41.685089] Proceeding with grib to json export.
[2025-02-13 10:23:47.229066] Wind data exported to ./public/wind.json
[2025-02-13 10:25:01.956598] Received data: {'message': [[-1.8489491682641699, 4.242973708068902], [12.507545770557073, -10.335455416336131]], 'process': True}
[2025-02-13 10:25:02.034818] Starting optimization. Start: (-1.8489491682641699, 4.242973708068902), End: (12.507545770557073, -10.335455416336131)

[2025-02-13 10:29:27.811280] Processing to daily retrieve.
[2025-02-13 10:29:27.811745] Retrieving and processing forecast data. Export: False
[2025-02-13 10:29:27.812298] Proceeding with forecast retrieval.
[2025-02-13 10:29:29.188006] Response [0]. The weather dataset file time 2025-02-13 01:00:00 and current time 2025-02-13 10:29:29.188006 are on the same day. No download neede
[2025-02-13 10:29:29.188006] Proceeding with grib to json export.
[2025-02-13 10:29:29.215140] Wind data not exported.
[2025-02-13 10:59:07.907678] Processing to daily retrieve.
[2025-02-13 10:59:07.907678] Retrieving and processing forecast data. Export: False
[2025-02-13 10:59:07.915840] Proceeding with forecast retrieval.
[2025-02-13 10:59:09.727489] Response [0]. The weather dataset file time 2025-02-13 01:00:00 and current time 2025-02-13 10:59:09.727489 are on the same day. No download neede
[2025-02-13 10:59:09.727489] Proceeding with grib to json export.
[2025-02-13 10:59:09.753532] Wind data not exported.
```

Debugging



Debugging & Testing

All unit tests are located alongside the source code in the same directory, making them easier to find. Tests arer only available for the Python server. You can run the tests with the following command:

```
.venv\Scripts\activate # on Windows
.venv\bin\activate # on Unix

python -m unittest [function-to-test].py
```

- Basic functions testes with unittest
- Wind forecast : official forecast at reference harbors, and reference tools.

Unit tests

- modUtils :
  - test_log_writes_to_file : output log prompts existence and accuracy
  - test_log_debug_prints_message : prints to stout existence and accuracy
- modOptim :
  - test_create_grid : degArray and degSpeed build existence and accuracy.
  - test_closest_index : accuracy of index retrieve in a list.
  - test_main : export call existence.
- modRetrieve :
  - test_retrieve_forecast : if statement accuracy, input file existence.
  - test_process_forecast : if statement accuracy, output file existence.

Future Implementation

- Bathymetric maps integration
- Tidal heightsd and current forecast integration
- Boats specification integration
- Loggin methods
- Shipyard implementation : register boats to be used
- Trips list : register trips onto the app for future analytics or sharing with friends