



Dossier de programmation

Composants serveur applicatif

Description générale

Système d'exploitation et organisation interne générale

Liste des logiciels utilisés

Drupal

API REST

Modules additionnels installés :

Types de contenus

Rôles

Termes de taxonomie

MariaDB

Geoserver

Style cartographique

Autres composants cartographiques

Script de génération des cartes hors lignes

Synchronisation des données cartographiques externes

Autres composants

Image (resizer et cache)

Proxys Web

Composants application mobile

React native

Modules et versions

Écrans et structure

Affichage des données cartographiques

Compilation

iOS

Android

Composants serveur applicatif

Description générale

Système d'exploitation et organisation interne générale

Le serveur applicatif est construit sur chacun des environnements (developpement, preprod, prod), de la manière suivante, avec l'offre EcoCompose.

Les containers :

VM	Container	Usage
web	web-web-1	Front Web (Apache), service proxy proxy tuilage Shom Raster marine
web	web-drupal-1	Instance Drupal
web	web-offline_maps_builder-1	Constructeur de cartes hors-lignes
web	web-static-1	Données pseudo-statiques (style cartographique, images OFB et DGAMPA avec gestion du cache)
geoserver	naveco_geoserver	Instance Geoserver
geoserver	naveco_geosync_crons	Scripts de synchronisation de données géolocalisées (crons)
EcoSQL		Serveur de base de données PostgreSQL/PostGIS (utilisé par geoserver)
Base MariaDB managée		Serveur MariaDB associé à l'instance Drupal

Liste des logiciels utilisés

Outre les logiciels et bibliothèques venant avec Debian, des logiciels ont été déployés spécifiquement pour le projet Nav&Co. Les logiciels externes sont détaillés en premier, suivi des composants développés pour le compte du projet.

Nom	Container	Licence
MariaDB (MySQL)	-	GPL v2
Drupal	web-drupal-1	GPL v2
GeoServer	naveco_geoserver	GPL v2
PostgreSQL	-	PostgreSQL License (compatible GPL)
PostGIS	-	GPL v2

Apache 2	web-web-1, web-drupal-1, web-static-1	Apache-2 (compatible GPL-3)
----------	---------------------------------------	-----------------------------

Drupal



Version 10.6.10.

Accès sur <https://naveco.ping-info-nautique.fr/users>.

Documentation de Drupal : <https://www.drupal.org/documentation>

API REST

Le back office de gestion de contenu permet de synchroniser les données relatives aux utilisateurs (bateaux, parcours, informations personnelles) et de gérer l'authentification (identification, inscription) lors de l'utilisation de l'application.

Liste des services :

- Authentification : permet à l'utilisateur de s'identifier dans l'application, renvoie l'identifiant du compte et un token d'accès requis par d'autres services ;
- Re-initialisation de mot de passe : permet de déclencher l'envoi du mail contenant les instructions de re-initialisation de mot de passe à l'utilisateur de l'application ;
- Versions de l'application : permet de récupérer les informations sur les versions minimales requises en fonction des plateformes ainsi qu'un lien vers la mise à jour sur les stores ;
- Inscription utilisateur : permet à l'utilisateur de créer un compte dans l'application ;
- Édition du profil utilisateur : permet à l'utilisateur de mettre à jour ses informations personnelles (image, pseudo et courriel) ;
- Enregistrement de l'avatar de l'utilisateur : permet à l'application d'enregistrer une image sur le serveur pour qu'elle soit ensuite liée à un compte utilisateur (image d'avatar) ;

- Création de bateau : permet à un utilisateur identifié de créer un bateau ;
- Modification de bateau : permet à un utilisateur identifié de modifier un bateau présent dans sa liste (nom, type de bateau) ;
- Suppression de bateau : permet à un utilisateur identifié de supprimer un bateau présent dans sa liste ;
- Création de parcours : permet à un utilisateur identifié de créer un parcours ;
- Modification de parcours : permet à un utilisateur identifié de modifier un parcours présent dans sa liste (nom, bateau) ;
- Suppression de parcours : permet à un utilisateur identifié de supprimer un parcours présent dans sa liste ;
- Liste de parcours : permet de récupérer la liste des parcours d'utilisateur ;
- Session token (X-CSRF-Token) : permet la récupération d'une clé requise pour le service d'upload d'avatar.

Modules additionnels installés :

Modules bookBeo

Module	Description
bookBeo utils	Bookbeo utils for Nav Co project
Marine Map	Integrate the map from the Shom.
Nav&Co - Custom Boat serializer	Custom Boat serializer for Nav&Co project
Nav&Co - Custom Trip serializer	Custom Trip serializer for Nav&Co project
REST API App Version	NavCo project REST Resource for appVersion

Modules contribués (juin 2025)

Module	Description	Version
Admin Toolbar	Provides an improved drop-down menu interface to the site Toolbar.	3.6.3
Consumers	Declare all the consumers of your API	8.x-1.24
Entity	Provides expanded entity APIs	8.x-1.6
Feeds	Regroupe les flux RSS/Atom/RDF, importe des fichiers CSV	8.x-3.2

Feeds extensible parser	A generic Feeds parser used to create extensible parsers	8.x-1.0
File entity	Extends Drupal file entities to be fieldable and viewable.	8.x-2.4
Geocoder Field	Provides Geocoder integration with contrib Geofield module.	8.x-3.31
Geofield	Stores geographic and location data (points, lines, and polygons).	8.x-1.67
Image Base64 Formatter	Adds a simple formatter for image fields that prints the base64 of the image.	2.0.5
Leaflet	Provides a leaflet map widget for geofields.	10.4.8
Login with email or username	Allow users to log in with either their username OR email address using the same input box on the login form.	2.1.0
Password Policy	Définit les contraintes et expirations de mots de passe.	4.0.3
Password Policy Extras	Various additions and enhancements to the Password Policy module.	4.0.0
Redirect	Permet aux utilisateurs de rediriger depuis d'anciennes URLs vers de nouvelles URLs.	8.x-1.13
rest password	Rest Service to Request Forgotten password	8.1.20
REST UI	Provides a user interface to manage REST resources.	8.x-1.22
RESTful Web Services	Expose les entités et d'autres ressources comme API web RESTful	10.6.9 (cœur)
Serialization	Provides a service for (de)serializing data to/from formats such as JSON and XML.	10.6.9 (cœur)
Simple OAuth	The OAuth 2.0 Authorization Framework	5.2.5
Token	Provides a user interface for the Token API and some missing core tokens.	8.x-1.17
Views GeoJSON	Provides a Views feed style for GeoJSON output.	8.x-1.4
X-frame-options Configuration	Add x-frame-options header	8.x-1.5

Types de contenus

Page (page de contenu rédactionnel)

- Description (texte riche)

Parcours (parcours utilisateurs)

- Date
- Distance
- Geofield (trace GPS)
- Région
- Identifiant de bateau
- Nom du bateau
- Type de bateau

Bateau (bateaux utilisateurs)

- Nom du bateau
- Type de bateau (conservé pour assurer la rétro-compatibilité démonstrateur)
- Taille (conservé pour assurer la rétro-compatibilité démonstrateur)

Configuration (lié au module Rest AppVersion)

Rôles

Utilisateur anonyme : Peut accéder à un formulaire d'identification, visualiser les pages de contenu rédactionnel ;

Utilisateur app : Utilisateur identifié, ne dispose pas de droits particulier sur les contenus ;

Administrateur : Utilisateur disposant de tous les droits disponibles sur Drupal ;

Utilisateur authentifié : Utilisateur issu de l'application, peut créer, voir et modifier sur contenus de type "Parcours" ou "Bateau", peut modifier ses informations utilisateur ;

Termes de taxonomie

Région

- Atlantique-Manche-Mer du Nord / atlantiquenord
- Bretagne / bretagne (pour le démonstrateur seulement)

- Guadeloupe / guadeloupe
- Méditerranée / mediterranee
- Martinique / martinique
- Saint-Martin et Saint-Barthélemy / stmartin_stbarthelemy

Type de bateau

- Bateau à moteur habitable
- Bateau à moteur sans cabine
- Canoé kayak
- Véhicules nautiques à moteur : jet ski, scooter des mers
- Voile légère
- Voilier habitable
- Voilier sans cabine

MariaDB

Cf. https://portail-support.din.developpement-durable.gouv.fr/projects/1836/wiki/Infoscs_cloud_eco4prod

Geoserver



Version 2.25.4.

Accès sur <https://geoserver.naveco.ping-info-nautique.fr/>

Documentation de Geoserver (installation et usage) :

<https://docs.geoserver.org/latest/en/user/>

Geoserver est configuré selon l'architecture technique, avec ses espaces de travail (de développement et de production), ses entrepôts, ses couches.

Geoserver a été configuré avec la création d'une couche complémentaire, d'agrégation, appelée « pois ». Elle contient l'ensemble des couches

Drawing order		Type	Couche	Style par défaut	Style	Retirer
1	↓	Layer	dev:nw	☐	generic	
2	↑ ↓	Layer	dev:amp	☐	polygon	
3	↑ ↓	Layer	dev:aerial	☐	point	
4	↑ ↓	Layer	dev:seabed	☐	point	
5	↑ ↓	Layer	dev:submarine	☐	point	
6	↑ ↓	Layer	dev:regulations	☐	polygon	
7	↑ ↓	Layer	dev:bcncar	☐	generic	
8	↑ ↓	Layer	dev:bcnisd	☐	generic	
9	↑ ↓	Layer	dev:bcnlat	☐	generic	
10	↑ ↓	Layer	dev:bcnspp	☐	generic	
11	↑ ↓	Layer	dev:boyar	☐	generic	
12	↑ ↓	Layer	dev:boyinb	☐	generic	
13	↑ ↓	Layer	dev:boylat	☐	generic	
14	↑ ↓	Layer	dev:boysaw	☐	generic	
15	↑ ↓	Layer	dev:boyspp	☐	generic	
16	↑ ↓	Layer	dev:daymar	☐	generic	
17	↑ ↓	Layer	dev:forstc	☐	generic	
18	↑ ↓	Layer	dev:goodpractices	☐	point	
19	↑ ↓	Layer	dev:Indmrk	☐	generic	
20	↑ ↓	Layer	dev:morfac	☐	generic	
21	↑ ↓	Layer	dev:newobj	☐	generic	
22	↑ ↓	Layer	dev:ofsplf	☐	generic	
23	↑ ↓	Layer	dev:pilpnt	☐	generic	
24	↑ ↓	Layer	dev:pylons	☐	generic	
25	↑ ↓	Layer	dev:siltnk	☐	generic	
26	↑	Layer	dev:simplified_seagrass	☐	generic	

Cette couche agrégée détermine est indépendante de l'ordre d'affichage des couches dans l'application mobile. Cet ordre est déterminé dans le fichier de thème décrit plus loin.

L'application mobile utilise deux flux WMTS issus de Geoserver :

- Un flux général, `navcogeo`, vectoriel, de format `application/vnd.mapbox-vector-tile` pour l'ensemble des points d'intérêt, ponctuels, linéaires ou polygonaux.
- Un flux complémentaire, `seagrass`, raster, de format `image/png` pour la diffusion de la couche d'herbiers simplifiée.

Cache de données

Les données de ces deux flux sont mises en cache. Le cache de navcogeo est supprimé à chaque nouvelle synchronisation de données (chaque nuit). Le cache de seagrass, aux données plus pérennes et à la génération plus longues (ce sont des images), est lui permanent – il peut être supprimé et re-généré manuellement lorsqu'il y a besoin d'intégrer de nouvelles données.

Les données utiles dans l'application mobile sont affichées jusqu'à un niveau de zoom 15 inclus.

Style cartographique



CT web-static-1, répertoire `/var/www/html/static/map`

Le style est un fichier (et non un logiciel) déposé sur le serveur – dans le container de données statiques – et utilisé/mis en cache par l'application mobile à chaque utilisation. Il décrit les éléments de style de chaque couche de données : couleurs, pictogrammes, etc.

Il suit les spécifications de Mapbox styles, également utilisé par son *fork* opensource MapLibre : <https://docs.mapbox.com/mapbox-gl-js/style-spec/>

C'est un fichier JSON qui décrit l'ensemble des sources de données, les styles des couches et d'autres attributs (zoom initial, etc.). Il fait appel également, dans le cadre de Nav&Co, à un fichier complémentaire appelé « tilejson » décrivant la source de données générale associée à la couche `navcogeo` et dont les spécifications sont disponibles à l'adresse suivante :

<https://github.com/mapbox/tilejson-spec>. Les polices d'écriture des objets cartographique et les images des pictogrammes sont aussi référencés dans le fichier de style.

L'application mobile part de ce style là, et peut aussi enrichir le style par du code en propre (et le fait).

Ci-après une version réduite du fichier de style avec l'ensemble des sources de données et attributs ainsi que de quelques couches (fond de carte Raster marine, Raster herbiers, polygones de la couche de réglementation (sans les pictogrammes) et les points d'intérêt environnementaux de type Faune sous-marine (submarine)).

```
{
  "version": 8,
  "name": "NavCo",
  "center": [
    -2.741802,
    48.564436
  ],
  "zoom": 10,
  "glyphs": "https://naveco.ping-info-nautique.fr/static/map/fonts/{fontstack}/{range}.pbf",
  "sprite": "https://naveco.ping-info-nautique.fr/static/map/sprite",
  "sources": {
    "pois": {
      "type": "vector",
      "url": "https://naveco.ping-info-nautique.fr/static/map/pois-tilejson.json"
    },
    "navw": {
      "type": "vector",
      "url": "https://naveco.ping-info-nautique.fr/static/map/navw-tilejson.json"
    },
    "raster": {
      "type": "raster",
      "tiles": [
        "https://naveco.ping-info-nautique.fr/tiles/tiles/{z}/{x}/{y}"
      ],
      "tileSize": 256
    },
    "seagrass": {
```

```

        "type": "raster",
        "url": "https://naveco.ping-info-nautique.fr/static/m
ap/seagrass-tilejson.json"
    },
    "layers": [
        {
            "id": "raster",
            "source": "raster",
            "type": "raster",
            "minzoom": 0,
            "maxzoom": 16
        },
        {
            "id": "seagrass",
            "source": "seagrass",
            "type": "raster",
            "minzoom": 0,
            "maxzoom": 16,
            "paint": {
                "raster-opacity": 0.75
            },
            "layout": {
                "visibility": "none"
            }
        },
        {
            "id": "regulations-layer-fill",
            "source": "pois",
            "source-layer": "regulations",
            "type": "fill",
            "paint": {
                "fill-antialias": true,
                "fill-color": "#BA0000",
                "fill-opacity": 0.3,
                "fill-outline-color": "#BA0000"
            },
            "layout": {
                "visibility": "none"
            }
        }
    ]
}

```

```

    }
  }, {
    "id": "pois-env-submarine",
    "source": "pois",
    "source-layer": "submarine",
    "type": "symbol",
    "minzoom": 0,
    "maxzoom": 24,
    "layout": {
      "visibility": "none",
      "icon-image": "pin_fish",
      "icon-size": [
        "interpolate",
        ["linear"],
        ["zoom"],
        6,
        0.1,
        9.5,
        0.5
      ],
      "icon-anchor": "bottom",
      "icon-allow-overlap": true
    },
    "paint": {
      "icon-opacity": 1.0
    }
  }
]
}

```

Autres composants cartographiques

Script de génération des cartes hors lignes



CT web-offline_maps_builder-1, répertoire `/app`

Les cartes hors lignes sont des fichiers de type base de données SQLite, au format compris et utilisé par MapLibre pour la transmission de paquets de données.

Un script, situé dans le fichier `sideload_maps.py`, a été réalisé pour pouvoir générer ces cartes. Chaque carte a un périmètre défini (bbox) et est référencée dans un fichier GeoJSON donnant leur géométrie et les propriétés suivantes :

- `carte_id` (numéro de carte)
- `title` (nom de la carte hors ligne, lisible par les utilisateurs)
- `region` (valeurs possibles : `atlantiquenord`, `guadeloupe`, `martinique`, `stmartin_stbarthelemy` ou `mediterranee`)

Ce fichier GeoJSON est un export Shom, transmis, auquel a été ajouté l'attribut région. Chaque géométrie est un rectangle, assimilable à une bbox.

Le script prend en entrée ce fichier ainsi que le fichier de style cartographique. Pour chaque carte hors-ligne à construire, il génère un fichier SQLite (avec l'extension `.db`) grâce au programme `mbgl-offline` de Mapbox (composant opensource de Mapbox) installé dans le répertoire `mapbox/build/bin/`.

Ce fichier SQLite est chiffré à l'aide de la librairie `SQLCipher` en utilisant une clé secrète, générée à partir des méta-données de la carte (coordonnées, numéro de carte, propriétés...).

À la fin de l'exécution, des fichiers SQCipher sont produits ainsi qu'un fichier JSON d'index de l'ensemble des cartes hors lignes disponibles, que l'application mobile charge lorsque l'utilisateur va sur l'écran de téléchargement de cartes hors ligne.

Synchronisation des données cartographiques externes



CT geoserver-geosync-crons, répertoire `/app`

Des scripts ont été créés pour pouvoir synchroniser un ensemble de données issues de sources externes :

- Balisage (WFS Shom INSPIRE)
- Aires marines protégées (GeoJSON OFB en fichier statique)

- Règlements (GeoJSON OFB en flux, puis géorèglements PING par WFS)
- Périmètres simplifiés des herbiers (flux WFS OFB)
- Points d'intérêt environnementaux (GeoJSON OFB en flux)
- Avis (urgents) à la navigation (API PING, XML)

Chacun de ces six scripts Python, pour autant de sources, télécharge les données depuis les flux correspondants (sauf pour les aires marines protégées, qui sont sous forme d'un fichier statique), puis les intègre dans la base de données PostGIS dédiée, en retraitant les données si besoin.

À noter également :

- la présence d'un script complémentaire, `tools/dead-links-urls.js`, de contrôle des liens morts, lui, en Javascript. Il contrôle les fichiers ou URL soumises en GeoJSON. C'est à dire l'ensemble des flux traités par les scripts de synchronisation détaillés.
- la présence d'un script de suppression de cache, `geoserver/empty-cache.py`, qui est exécuté la fin de l'import de l'ensemble des données.
- Le script de synchronisation du balisage, importe l'ensemble des types de balisage, à l'exclusion des bouées d'amarrage (MORFAC et BOYINB), types de données non utilisés dans le cadre de Nav&Co.

Les scripts de synchronisation de données, puis le script de suppression de cache sont exécutables manuellement, soit automatiquement chaque nuit, dans un crontab de l'utilisateur root (édition avec la commande `crontab -e`). Les fréquences y sont réglables à travers cet outil. Les avis à la navigation peuvent être rafraichis plus fréquemment.

Autres composants

Image (resizer et cache)



CT web-static-1, répertoire `/var/www/html/static/images`

/envpois Lors de l'interrogation, par l'application mobile, d'images de points d'intérêt balisage ou environnementaux, une image est chargée. Chaque image d'un point d'intérêt environnemental est située sur un serveur OFB et d'une taille trop élevée par rapport aux besoins d'affichage en bulle d'information/imagette (de l'ordre, parfois, de plusieurs méga-octets). Le serveur Nav&Co fait donc office de proxy pour atteindre ces images, et, dans le cadre des images POI environnementaux, générer des imagettes (vignettes) de quelques dizaines de kilo-octets, plus facilement consommables par l'application mobile. Ces imagettes sont mises en cache pour utilisations futures. Le service retaillant les images utilise CDNThumbnailer (logiciel opensource, PHP) qui opère la conversion.

/beaconage Pour les imagettes du balisage, un script php a été créé pour télécharger les images du balisage, les conserver dans un dossier de le cache et de les retransmettre dans le flux Web à l'app. Pas de redimensionnement.

Proxys Web

Tuiles Raster marine Shom



CT web-web-1

Les tuiles Raster marine sont pris en charge directement depuis le front Web Apache, avec proxy pour atteindre la vraie source des images, situées sur un serveur Shom. La couche « assemblage de cartes » est utilisée. Des ajustements sont faits dans la configuration de manière à afficher certaines carte pour des zones non couvertes sur certains niveaux de zoom (cas du goulet de Brest et du golfe du Morbihan, qui ne sont pas couverts en échelle 1/50 000 mais en échelle 1/20 000 seulement).

Accès aux pages de marégrammes



CT web-static-1 répertoire `/var/www/html/static/tides`

Faux proxy pour amener l'utilisateur sur une page marée d'un port : une page Web php avec iframe vers la page de marées située sur

https://services.data.shom.fr/hdm/vignette/petite/ID_PORT.

Bannissement des attaques contre le Raster marine, avec Nginx, Fail2ban et Sendmail

1. Nginx.



CT #100, fichier </etc/nginx/sites-available/naveco.ping-info-nautique.fr>

Configuration du module de limitation des requêtes dans le temps :

```
limit_req_zone $binary_remote_addr zone=rasterlimit:10m rate=30r/s;
```

...

```
limit_req zone=rasterlimit burst=120;  
limit_req_status 429;
```

Limitation à 30 requêtes par seconde avec un burst pour chaque IP de 120.

2. Fail2ban



CT #100, fichier </etc/fail2ban/filter.d/nginx-limit-req.conf>

Configuration des zones suivies (rasterlimit seulement)

[Definition]

```
ngx_limit_req_zones = rasterlimit
```

```
failregex = ^\s*\[[a-z]+\] \d+#\d+: \*\d+ limiting requests, excess: [\d\.]+ by zone "(?:(ngx_limit_req_zones)s)", c
```

```
lient: <HOST>,  
  
ignoreregex =  
  
datepattern = {^LN-BEG}
```

Et du jail (`/etc/fail2ban/jail.local`)

- Durée de bannissement : 86400 secondes (1 jour) pour une fenêtre de 600 secondes.

```
[nginx-limit-req]  
enabled = true  
filter = nginx-limit-req  
action = iptables-multiport[name=ReqLimit, port="http,http  
s", protocol=tcp]  
logpath = %(nginx_error_log)s  
findtime = 600  
bantime = 86400  
maxretry = 10
```

Composants application mobile

React native

Modules et versions

Des modules npm contenant du code natif ainsi que du code application sont utilisés pour la majorité des fonctions développées. Ils peuvent permettre de réaliser des fonctions complexes (gestion des menus, de la carte ...), contenir des composant graphiques (menus déroulants, barre de recherche) ou contenir des fonctions utilitaires (gestion des traductions, des formats de date, des structures de données etc..)

Les modules utilisés sont :

Nom	Version	Licence	Fonction
@babel/core	7.29.0	MIT	Compilation/transpilation JavaScript via Babel

Nom	Version	Licence	Fonction
@babel/eslint-parser	7.28.6	MIT	Parser Babel pour ESLint
@babel/plugin-proposal-logical-assignment-operators	7.20.7	MIT	Support Babel des opérateurs d'assignation logique
@babel/preset-env	7.29.2	MIT	Preset Babel pour compatibilité JavaScript
@babel/runtime	7.29.2	MIT	Helpers Babel embarqués à l'exécution
@mapbox/geo-viewport	0.4.1	BSD-2-Clause	Calcul du viewport/zoom à partir d'une emprise géographique
@react-native-async-storage/async-storage	1.24.0	MIT	Stockage local persistant clé/valeur
@react-native-community/cli-platform-android	15.0.1	MIT	Outils CLI React Native pour Android
@react-native-community/cli-platform-ios	15.0.1	MIT	Outils CLI React Native pour iOS
@react-native-community/cli	15.0.1	MIT	Interface en ligne de commande React Native
@react-native-community/eslint-config	3.2.0	MIT	Configuration ESLint historique pour React Native
@react-native-community/masked-view	0.1.11	MIT	Composant de masque visuel
@react-native-community/netinfo	11.5.2	MIT	Détection de l'état réseau et connectivité
@react-native-community/toolbar-android	0.2.1	MIT	Composant Toolbar natif Android
@react-native-mapbox-gl/maps	8.5.0	MIT	Affichage de cartes Mapbox/MapLibre

Nom	Version	Licence	Fonction
@react-navigation/native	6.1.18	MIT	Socle de navigation de l'application
@react-navigation/bottom-tabs	6.6.1	MIT	Navigation par onglets inférieurs
@react-navigation/drawer	6.7.2	MIT	Navigation par menu latéral
@react-navigation/stack	6.4.1	MIT	Navigation par pile d'écrans
@rneui/base	4.0.0-rc.7	MIT	Composants graphiques de base
@rneui/themed	4.0.0-rc.8	MIT	Thématisation des composants graphiques
@turf/buffer	5.1.5	MIT	Calcul de zones tampons géographiques
@turf/circle	6.5.0	MIT	Génération de cercles géographiques
@turf/union	6.5.0	MIT	Union de géométries GeoJSON
i18n-js	3.9.2	MIT	Gestion des traductions
react-native-localize	2.2.6	MIT	Localisation : langue, région, fuseau horaire
react-native-location	2.5.0	MIT	Géolocalisation et suivi de position
react-native-permissions	3.6.1	MIT	Gestion des permissions natives
react-native-webview	11.26.1	MIT	Affichage de pages ou composants web
react-native-size-matters	0.4.2	MIT	Adaptation responsive des tailles
react-native-sqlcipher-16kb	1.0.5	MIT	Accès/déchiffrement de bases SQLite chiffrées
matomo-tracker-react-native	0.3.3	MIT	Envoi de données Analytics vers Matomo
redux	4.2.1	MIT	Gestion centralisée de l'état applicatif

Nom	Version	Licence	Fonction
redux-persist	6.0.0	MIT	Persistance du store Redux
react-redux	7.2.9	MIT	Liaison React avec Redux
redux-thunk	2.4.2	MIT	Actions Redux asynchrones
redux-logger	3.0.6	MIT	Journalisation des actions Redux
react-native	0.77.3	MIT	Framework mobile natif basé sur React
react	18.3.1	MIT	Bibliothèque UI de base
react-native-fs	2.20.0	MIT	Accès au système de fichiers natif
react-native-image-picker	4.10.3	MIT	Sélection ou prise de photo/vidéo
react-native-device-info	14.1.1	MIT	Informations sur l'appareil et l'application
react-native-config	1.6.1	MIT	Gestion des variables d'environnement
react-native-background-actions	4.1.0	MIT	Exécution de tâches en arrière-plan
react-native-gesture-handler	2.25.0	MIT	Gestion avancée des gestes tactiles
react-native-reanimated	3.18.0	MIT	Animations natives performantes
react-native-screens	4.18.0	MIT	Optimisation native des écrans
react-native-safe-area-context	5.7.0	MIT	Gestion des zones sûres écran
react-native-svg	15.11.1	MIT	Rendu SVG dans React Native
react-native-vector-icons	7.1.0	MIT	Bibliothèque d'icônes vectorielles
maplibre-gl	5.23.0	BSD-3-Clause	Moteur cartographique MapLibre GL
moment	2.30.1	MIT	Gestion et formatage des dates

Nom	Version	Licence	Fonction
lodash	4.18.1	MIT	Fonctions utilitaires JavaScript
uuid	8.3.2	MIT	Génération d'identifiants UUID
check-password-strength	3.0.0	MIT	Évaluation de la robustesse d'un mot de passe
compare-versions	4.1.4	MIT	Comparaison de versions
remove-accents	0.4.4	MIT	Suppression des accents dans les chaînes
html-entities	1.4.0	MIT	Encodage/décodage d'entités HTML
buffer	6.0.3	MIT	Polyfill Buffer Node.js
react-native-app-intro-slider	4.0.4	MIT	Écrans d'introduction/onboarding
react-native-autolink	4.2.0	MIT	Détection automatique de liens
react-native-dropdown-picker	5.4.6	MIT	Liste déroulante
react-native-hyperlink	0.0.22	MIT	Liens cliquables dans du texte
react-native-image-header-scroll-view	0.10.3	MIT	ScrollView avec en-tête image
react-native-keep-awake	4.0.0	MIT	Empêche la mise en veille
react-native-keyboard-aware-scroll-view	0.9.5	MIT	Adaptation au clavier
react-native-linear-gradient	2.8.3	MIT	Dégradés linéaires
react-native-orientation-locker	1.7.0	MIT	Gestion de l'orientation écran
react-native-popup-menu	0.18.0	MIT	Menus contextuels/popups
react-native-progress	4.1.2	MIT	Indicateurs de progression

Nom	Version	Licence	Fonction
react-native-reanimated-carousel	3.0.6	MIT	Carrousels animés
react-native-simple-toast	1.1.4	MIT	Messages toast natifs
react-native-status-bar-height	2.6.0	MIT	Hauteur de la barre de statut
react-native-svg-transformer	1.5.3	MIT	Import/transformation de SVG
react-native-unordered-list	1.0.4	MIT	Listes à puces
react-native-walkthrough-tooltip	1.6.0	MIT	Bulles d'aide/tutoriels
rn-fetch-blob	0.12.0	MIT	Téléchargement et manipulation de fichiers/blobs
rn-sliding-up-panel	2.4.6	MIT	Panneau coulissant depuis le bas
prop-types	15.8.1	MIT	Validation des props React
react-icomoon	2.6.1	MIT	Affichage d'icônes IcoMoon
deprecated-react-native-prop-types	5.0.0	MIT	Compatibilité avec anciens PropTypes
@react-native/babel-preset	0.77.3	MIT	Preset Babel officiel React Native
@react-native/metro-config	0.77.3	MIT	Configuration Metro officielle
@react-native/eslint-config	0.77.3	MIT	Configuration ESLint officielle
@react-native/typescript-config	0.77.3	MIT	Configuration TypeScript officielle
typescript	5.0.4	Apache-2.0	Langage TypeScript et compilateur
jest	30.3.0	MIT	Framework de tests
babel-jest	30.3.0	MIT	Transformation Babel pour Jest

Nom	Version	Licence	Fonction
ts-jest	29.4.9	MIT	Support TypeScript dans Jest
react-test-renderer	18.3.1	MIT	Rendu React pour tests
@testing-library/react-native	9.2.0	MIT	Tests de composants React Native
@testing-library/jest-dom	5.17.0	MIT	Matchers Jest DOM
@testing-library/user-event	14.6.1	MIT	Simulation d'interactions utilisateur
eslint	8.57.1	MIT	Analyse statique du code
prettier	2.8.8	MIT	Formatage automatique du code
@typescript-eslint/eslint-plugin	5.62.0	MIT	Règles ESLint TypeScript
eslint-plugin-react	7.37.5	MIT	Règles ESLint React
eslint-plugin-react-hooks	4.6.2	MIT	Règles ESLint hooks React
eslint-plugin-react-native	4.1.0	MIT	Règles ESLint React Native
eslint-plugin-react-native-a11y	3.5.1	MIT	Règles d'accessibilité React Native
eslint-plugin-jsx-a11y	6.10.2	MIT	Règles d'accessibilité JSX
@types/react	18.3.28	MIT	Types TypeScript pour React
@types/react-native	0.70.19	MIT	Types TypeScript pour React Native
@types/react-dom	18.3.7	MIT	Types TypeScript pour React DOM
@types/node	18.19.130	MIT	Types TypeScript pour Node.js
@types/jest	29.5.14	MIT	Types TypeScript pour Jest
patch-package	8.0.1	MIT	Application de patches locaux sur dépendances
jetifier	2.0.0	MIT	Compatibilité AndroidX

Nom	Version	Licence	Fonction
typedoc	0.23.28	Apache-2.0	Génération de documentation TypeScript
npm	9.9.4	Artistic-2.0	Gestionnaire de paquets Node.js
chalk	4.1.2	MIT	Coloration des sorties console
add	2.0.6	MIT	Utilitaire npm d'ajout de dépendances
react-native-eject	0.1.2	MIT	Éjection/conversion d'un projet React Native
react-native-typescript-transformer	1.2.13	MIT	Ancien transformateur TypeScript React Native
metro-react-native-babel-preset	0.71.3	MIT	Ancien preset Babel Metro
remote-redux-devtools	0.5.16	MIT	Outils de debug Redux distants
redux-persist-transform-filter	0.0.20	MIT	Filtrage des données persistées Redux
lodash.memoize	4.1.2	MIT	Mémoïsation de fonctions
react-native-get-random-values	1.11.0	MIT	Génération de valeurs aléatoires compatibles crypto
react-native-safe-area-view	1.1.1	MIT	Ancien composant SafeAreaView

Aucun module issu de la communauté satisfaisant la condition de licence libre n'ayant été assez correct par rapport au cas d'usage Nav&Co, un module « custom » a aussi été développé pour remplir la fonction de gestion de l'enregistrement de la trace sur Android. Il se situe dans `android/app/src/main/java/com/navco/location`. Le module a été librement inspiré de <https://github.com/andersryanc/ReactNative-LocationSample>. Sur iOS, cette gestion de la géolocalisation est gérée directement via le module `react-native-location`.

En plus de ceux-là, de nombreux modules mineurs sont utilisés pour remplir des fonctions spécifiques, afficher des composants d'interface particulier ou sont des dépendances d'autres modules.

La liste complète des modules utilisé et la version correspondante peut être trouvé dans le "package.json" de l'application.

Écrans et structure

Le module react-native-navigation permet d'organiser les différents composants en les séparant par écrans.

L'écran principal, contenant la carte et l'accès aux différents menus correspond à un seul composant écran. Celui-ci est le même que l'on soit en mode navigation ou découverte.

Cela permet de conserver la position et le niveau de zoom actuel lorsque l'utilisateur change de mode.

Dans les écrans marées et les pages informatives (réglementation pêche, météo...), toutes les pages sont rassemblés dans un même écran. Celui-ci affiche le contenu sous la forme d'une vue web. De cette façon, seule l'adresse url de la page à afficher doit être conservée. Celles-ci sont stockées dans des fichiers spécifiques de sorte à être facilement modifiable si ces pages viennent à évoluer.

Affichage des données cartographiques

L'affichage des données cartographiques est effectué à l'aide du module MapLibre. Le module react-native de MapLibre permet de modifier le fichier de style (décrit dans le dossier d'architecture serveur) de façon dynamique. Dans l'application, les styles cartographiques sont modifiés pour :

- Cacher / Afficher les catégories sélectionnées
- Mettre en valeur les POIs / zones sélectionnées (grossissement de l'icône, épaississement des contours ...)
- Traiter les modification de pictogrammes (balises et bouées latérales en Guadeloupe)
- Afficher le rayon d'effet des POIs environnementaux
- Afficher la position et la trace courante de l'utilisateur

Compilation

iOS

La compilation iOS peut être effectuée directement en lançant le dossier iOS du projet avec Xcode ou avec l'une des commandes suivantes `npm run ios` ou `npx react-native run-ios`

Android

De la même façon, la compilation Android peut être effectuée à partir d'Android Studio ou avec les commandes `npm run android` ou `npx react-native run-android`