RAPPORT DE STAGE

Réalisation de la version 3 d'une application d'aide à la gestion des pêcheries d'holothuries

Morgan DUBOIS

Avril - Août 2018, 5 mois

Tuteurs en entreprise : Mme. Sylvie Fiat / M. Marc Léopold

Tuteur académique : M. Gérôme Canals

Formation : Licence Pro CISIIE - Iut Nancy Charlemagne - 2 ter boulevard Charlemagne, Nancy 54000

Entreprise d'accueil : IRD - 101 Promenade Roger Laroque, Noumea 98848, Nouvelle-Calédonie

IUT Nancy Charlemagne
Université de Lorraine
2 ter boulevard Charlemagne
BP 55227
54052 Nancy Cedex

Département informatique

Réalisation de la version 3 d'une application d'aide à la gestion des pêcheries d'holothuries

Rapport de stage de licence pro CISIIE

Entreprise : Institut de Recherche pour le Développement

Dubois Morgan

Tutrice : Fiat Sylvie

Tuteur: Léopold Marc

Enseignant référent : Canals Gérôme

Année universitaire : 2017 - 2018

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué au bon déroulement de mon stage et qui m'ont aidé lors de la rédaction de ce rapport.

Tout d'abord, j'adresse mes remerciements à tous les enseignants de la licence pro CISIIE qui m'ont apporté de nombreuses compétences qui m'ont été très utiles tout au long de mon stage et qui le seront également dans ma vie professionnelle. Je tiens plus particulièrement à remercier M. Canals qui a transmis cette offre de stage et m'a donc permis de réaliser ce stage.

Je tiens à remercier vivement ma tutrice de stage, Mme. Fiat, ingénieur en systèmes d'informations, pour son accueil, sa confiance et le partage de son expertise au quotidien. Je la remercie également pour son exigence qui m'a poussé à toujours donner le meilleur de moi et son aide précieuse quand j'en avais besoin. Je remercie également M. Léopold pour ses retours sur mon travail.

Je remercie également toute l'équipe de l'unité de recherche Entropie de l'IRD pour leur accueil et l'aide qu'ils ont pu me fournir lors de mon stage et en dehors de celui-ci.

Enfin, je tiens à remercier toutes les personnes qui m'ont conseillé et relu lors de la rédaction de ce rapport de stage : Mme. Fiat, M. Canals, ma famille.

Table des matières

Introduction

Dans un monde où 52 % du trafic web s'effectue depuis des Smartphones et où les utilisateurs exigent une rapidité ainsi qu'une expérience utilisateur irréprochable, les Progressives Web Apps (PWA) sont devenus une évidence. En effet, ces dernières ont pour mots d'ordres d'être **Progressive** (elles doivent fonctionner sur n'importe quel périphérique), **Engageante** (elles proposent une expérience utilisateur immersive) et **Rapide** (une fois l'application chargée, la navigation doit se faire de manière rapide et fluide même dans les cas de faibles connexions). Elles répondent donc à toutes les problématiques actuelles.

Dans le cadre de ma licence pro CISIIE à l'IUT Nancy Charlemagne, j'ai souhaité réaliser mon stage dans une entreprise qui me permettrait d'utiliser toutes les technologies relatives aux PWA. Ce stage correspondait donc exactement ce que je recherchais à savoir l'utilisation des technologies Angular*, ElectronJS*, Docker*, RxJS* ou encore PouchDB*. Voulant exercer par la suite le métier de développeur full stack, ma formation ainsi que ce stage m'ont permis d'acquérir une bonne partie des connaissances nécessaires.

Le sujet de mon stage porte sur le développement de la version 3.0 d'une application d'aide à la gestion des pêcheries d'holothuries (et autres invertébrés marins) en collaboration avec la *Seychelles Fishing Authority** ainsi que l'association des pêcheurs professionnels des Seychelles.

Dans un premier temps, je présenterai l'environnement professionnel particulier de l'IRD, puis j'expliquerais les différentes tâches que j'ai eu à effectuer lors de ces 5 mois de stage avant de conclure sur ce que j'aurais retiré de cette expérience.

**Glossaire disponible à la fin du rapport*

I. Présentation de l'entreprise : l'IRD

A. Une science engagée pour un futur durable

Les avancées scientifiques sont nécessaires pour faire progresser le développement durable et humain. Cette conviction, l'IRD, institut français de recherche pour le développement la porte partout où il est présent, partout où il agit avec ses partenaires. L'IRD, est un acteur français majeur de l'agenda international pour le développement. Son modèle est original : le partenariat scientifique équitable avec les pays en développement, principalement ceux des régions intertropicales et de l'espace méditerranéen. Pour l'IRD, seul ce modèle permet de concevoir des solutions adaptées aux défis auxquels les hommes et la planète font face : pandémies, dérèglements climatiques, crises humanitaires et politiques...

B. Les marqueurs d'une action au service du développement

Voici les différents marqueurs de l'action de l'IRD au service du développement :

- Un partenariat scientifique équitable et des co-publications avec les partenaires des pays en développement
- Des solutions adaptées aux défis globaux fondées sur l'évidence scientifique
- Des politiques publiques éclairées par les avancées de la science
- Des citoyens acteurs du changement
- Des innovations responsables
- Des expertises et savoir-faire spécifiques

C. Un acteur majeur de la recherche pour le développement

Organisme pluridisciplinaire reconnu internationalement, travaillant principalement en partenariat avec les pays méditerranéens et intertropicaux, l'Institut de Recherche pour le Développement est un établissement public français placé sous la double tutelle des

ministères de l'Enseignement supérieur et de la Recherche et des Affaires étrangères et du Développement international. Il porte, par son réseau et sa présence dans une cinquantaine de pays, une démarche originale de recherche, d'expertise, de formation et de partage des savoirs au bénéfice des territoires et pays qui font de la science et de l'innovation un des premiers leviers de leur développement.

- Une communauté riche de plus de 7000 personnes dont 2019 agents IRD (805 chercheurs et 1214 ingénieurs et techniciens IRD)
- 34,5 % des agents en poste à l'étranger
- 65 unités de recherche
- près de 1300 références d'articles publiés en 2016 par les chercheurs de l'IRD dans le Web of Science
- 52 % de co-publications avec un partenaire du Sud
- 230 M€ de budget annuel

II. Présentation du projet BDMer

BDMer est une application qui existe déjà et qui est actuellement utilisée pour la gestion des stocks d'invertébrés marins en Nouvelle-Calédonie et au Vanuatu, mais qui n'a pas été prévue pour une "industrialisation" informatique et dont les technologies commençaient à être obsolètes. L'entrée des Seychelles dans le projet a entraîné de grosses modifications dans les besoins de l'application et la décision a été prise de re-développer complètement l'application pour sa nouvelle version.

A. Outil de gestion de données

BDMer est un outil de gestion et d'analyse de données sur les ressources d'invertébrés coralliens (holothuries, trocas, bénitiers...), créé selon les besoins des utilisateurs. Il est financé grâce à un partenariat entre l'IRD (Institut de recherche pour le Développement), la

province Nord de Nouvelle-Calédonie, la Seychelles Fishing Authority et le Gouvernement du Vanuatu (Département des pêches) depuis 2011. Les utilisateurs de l'application sont les gestionnaires des départements des pêches de chaque pays participant au projet.

B. Traitements statistique

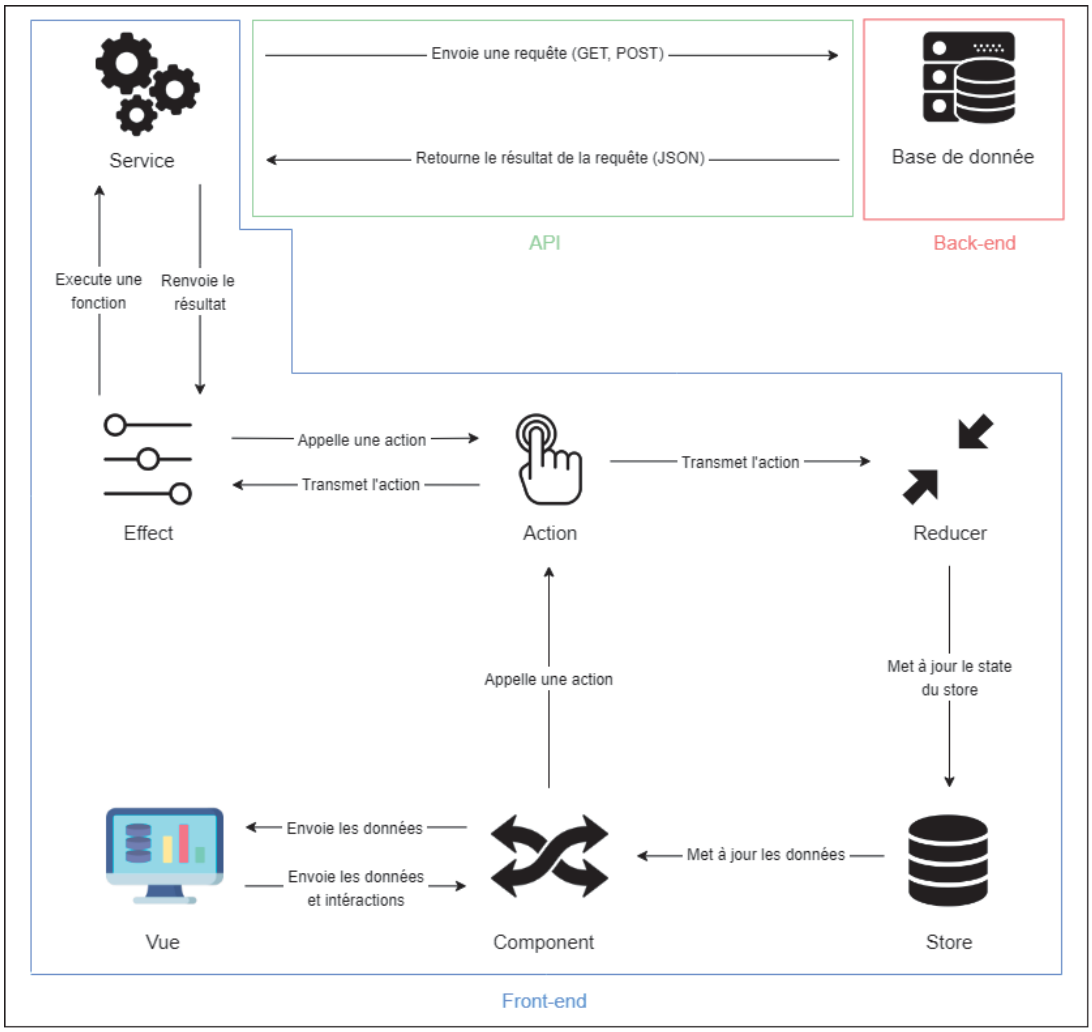
BDMer permet à la fois d'alimenter une base de données à partir d'observations d'invertébrés sur le terrain et de cartes d'habitats marins, puis de réaliser des traitements statistiques. L'utilisateur peut estimer l'état des stocks d'invertébrés (biomasse, abondance, densité, etc. par espèce) et les incertitudes associées grâce à un module d'analyse des données. Ces données vont permettre de faire des prévisions de stock ainsi que de fixer les espèces autorisées à la pêche ainsi que les quotas associés.

C. Création de rapport

BDMER est également doté d'un module de présentation automatique des résultats d'analyse (tableaux, graphiques...), permettant une lecture rapide et fonctionnelle dans des fiches de synthèse par espèce. Ces fiches peuvent être exportées en PDF pour constituer des rapports d'évaluation standardisés.

III. Corps du mémoire

A. Structure de l'application BDMer



Workflow de l'application BDMer

Vue : Affichage de l'utilisateur

Component : Présente les données et délègue l'accès aux données aux services via les actions

Store : Container de "state" (il peut donc y avoir plusieurs "state")

State : Un objet javascript immuable

Action : Un objet qui représente une intention de changement de state

Reducer : Accepte une accumulation de donnée et une valeur, transforme une collection de valeurs en une seule valeur

Effect : Contrôle le flux de donnée

Service : Permet de traiter les données

Base de données : Permet de stocker les données

B. Base de données

1. Structure de la base de données

Pour ce projet, nous travaillons avec Couchdb* qui est une base de données NoSQL* orientée document. Nous utilisons également Pouchdb qui est une base de données interne aux navigateurs utilisant IndexedDB* et inspirée par Couchdb. Cette architecture permet notamment de stocker les données dans le navigateur afin de pouvoir utiliser l'application même lorsque la connexion internet est faible voire perdue ce qui est un des dogmes des Progressives Web Apps.

Les données sont organisées en 4 bases de données :

- _users : les utilisateurs de l'application, les utilisateurs sont divisés en deux catégories : les administrateurs qui gèrent complètement l'application et ses données et les gestionnaires des pays qui ont accès aux données concernant leur pays uniquement ;
- countries : les pays participants au projet et pour lesquels des données sont récoltées et analysées ;
- platforms : les plateformes sont les entités à partir desquelles on effectue la récolte des données, il peut s'agir d'un bateau dans le cas des Seychelles ou d'un lieu géographique étudié dans le cas de la Nouvelle-Calédonie ou du Vanuatu.
    - zones : ces plateformes sont découpées en zones logiques basées soit sur des habitats similaires soit sur une bathymétrie cohérente ce qui permet de comparer les données recueillies ;
    - stations : les données sont recueillies ponctuellement par des plongeurs sur des stations (point géographique) ;
    - relevés : les relevées représentent une session de récolte ;
        - comptages : les comptages sont les données récoltées sur une station ;
- species: les différentes espèces d'invertébrés marins

2. Schéma de la base de données

Étant dans une configuration NoSQL, il est compliqué de schématiser la structure de la base de données. Cependant, selon moi, pour bien comprendre comment fonctionne le projet il faut avoir une vision globale de la base de données. Pour faire ceci, j'ai décidé de faire un schéma "relationnel" même si cela est techniquement faux, cela apporte une bonne vision.

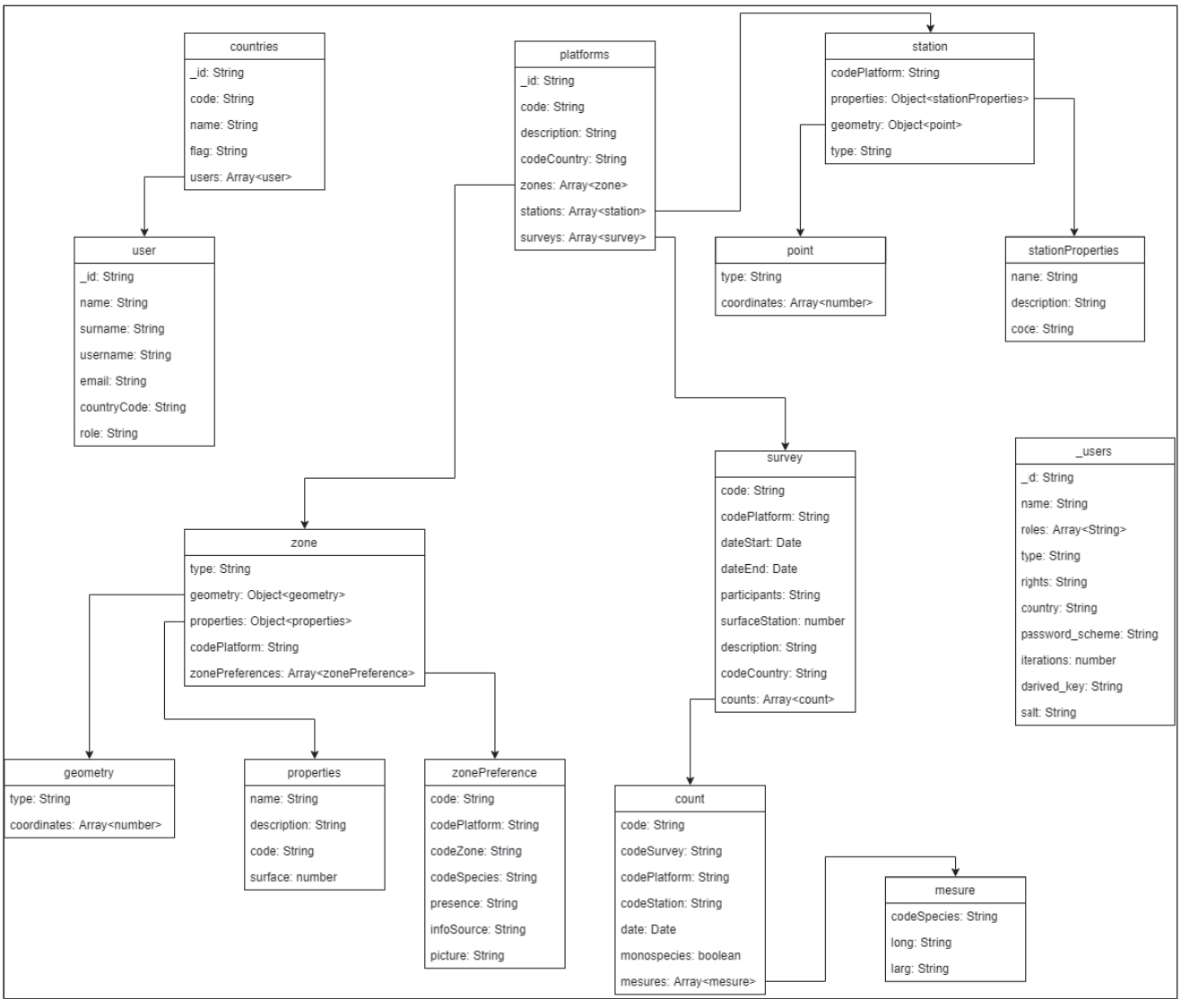


Schéma de la base de données

C. Le développement

Lors du développement, je suis intervenu sur les cinq grosses fonctionnalités suivantes :

- le développement sur les données de Pays
- les cartes interactives
- l'import et de traitement de fichier
- les formulaires de données
- une application d'importation de données, développé en parallèle

1. Pays

Les pays présents dans la base de données sont les pays possédant des plateformes et participant au projet BDMer. Pour cette partie, j'ai été amené à effectuer différentes tâches suivantes :

En tant qu'utilisateur, je veux pouvoir voir le drapeau du pays sur lequel je travaille.

La première était l'enregistrement d'une image du drapeau du pays dans la base de données. La base de données acceptant des fichiers au format base64, il m'a fallu convertir les fichiers d'entrée fournis en SVG* vers base64, ce que j'ai résolu en utilisant des promesses, ajax et btoa*.

En tant qu'administrateur, je veux voir les pays gérés dans l'application sur une carte.

La seconde était d'ajouter les coordonnées du pays lors de l'ajout d'un nouveau pays dans l'application et donc dans la base de données. Les coordonnées seront utiles pour les cartes interactives dont je parlerai après. Pour récupérer ces coordonnées, j'ai utilisé l'API Google Maps et son service de Geocoding. Il m'a suffit donc de faire une requête "GET" avec le nom du pays pour avoir une réponse au format JSON et extraire les coordonnées :

Ex :

http://maps.googleapis.com/maps/api/geocode/json?address=Seychelles&sensor=false&apiKey=YOUR_KEY

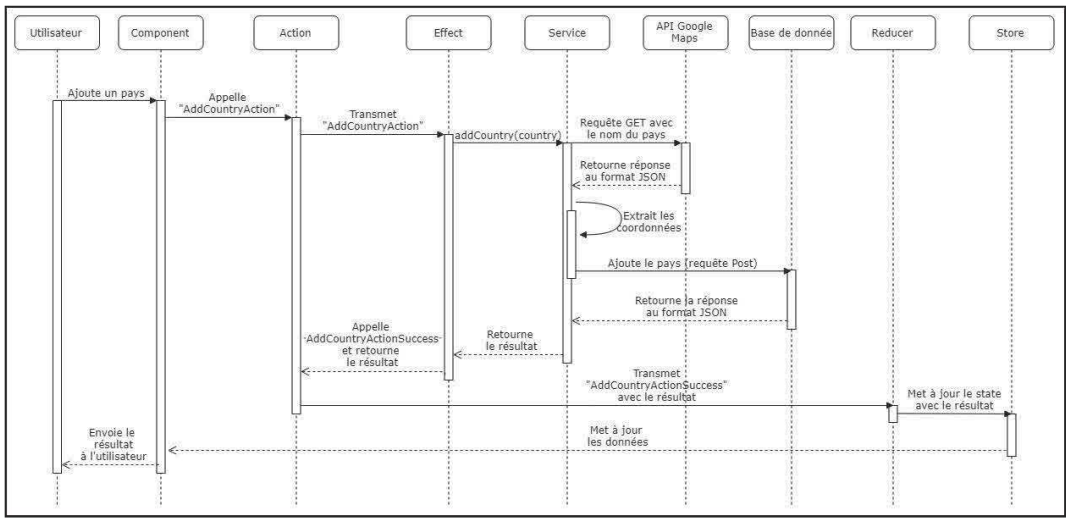


Diagramme de séquence d'ajout d'un pays

"En tant qu'utilisateur je veux que lorsque je supprime un pays, toutes les plateformes associées soient supprimés."

Pour réaliser cela, lors de la demande de suppression d'un pays, l'application récupère toutes les plateformes du pays et une fenêtre apparaît expliquant à l'utilisateur quelles plateformes vont être supprimées et lui demandant de confirmer son action.

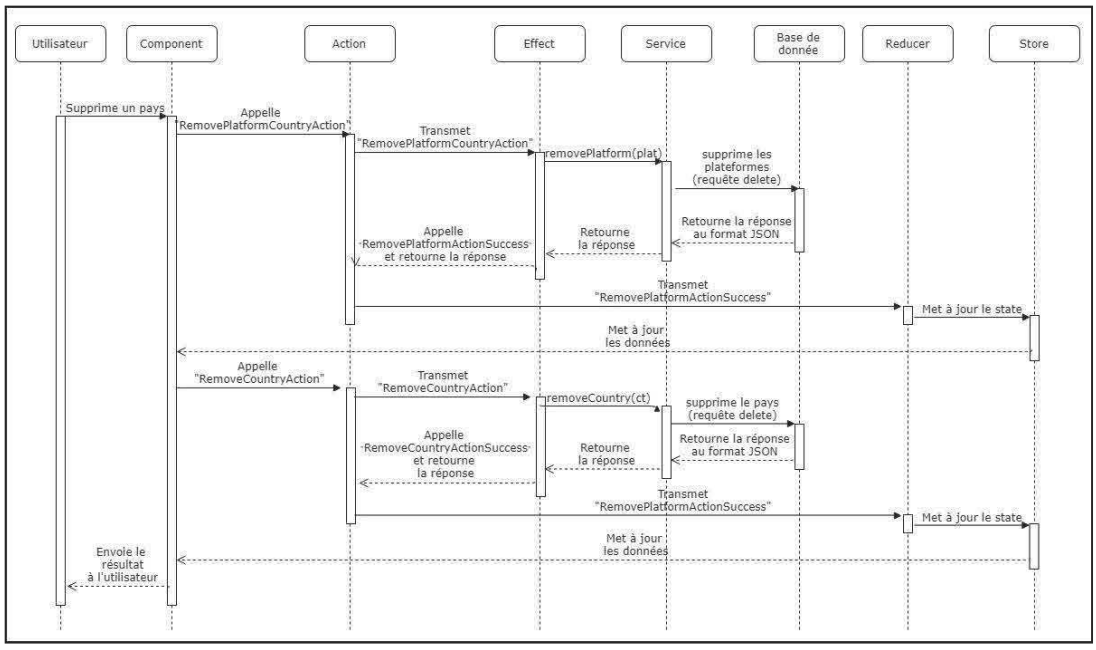
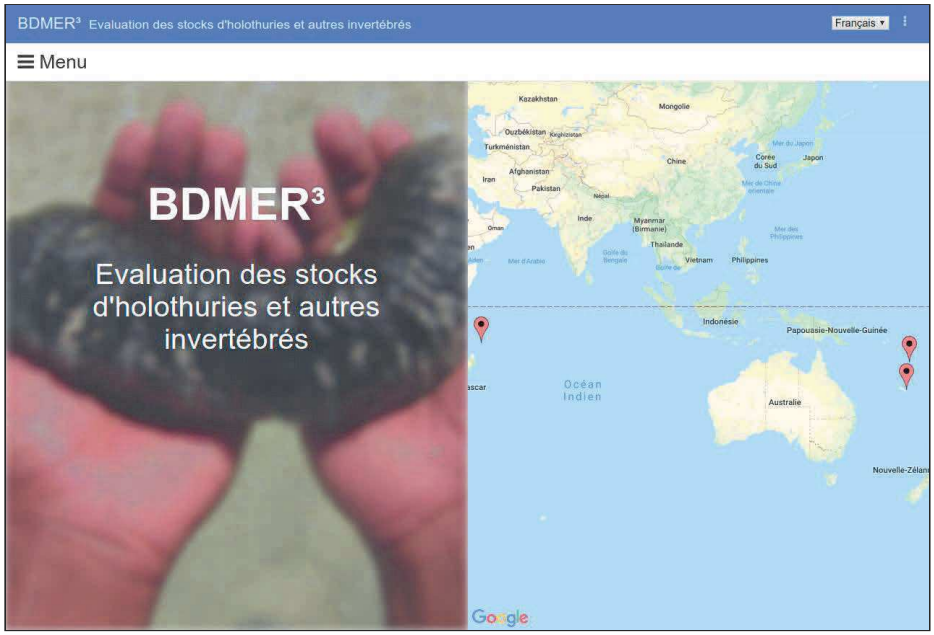


Diagramme de séquence de suppression d'un pays

2. Cartes

Les cartes représentent une partie importante de mon stage, en effet ces dernières sont utilisées sous différentes formes et à de multiples endroits de l'application. L'application a besoin de cartes dynamiques mais également de cartes statiques afin de les pouvoir les sauvegarder dans la base de donnée et les rendre disponibles à l'utilisateur même si la connexion à internet est faible ou inexistante. Pour faire cela j'ai besoin d'un service de cartographie, j'ai donc, dans un premier temps, commencé par utiliser l'API Google Maps, ces derniers étant leaders du marché et proposent un très bon service. Ayant déjà travaillé plusieurs fois avec, j'avais de bonnes affinités sur ce dernier. Voici les différents travaux que j'ai réalisé dans cette partie :

"En tant qu'utilisateur je veux que sur page d'accueil soit affichées sur une carte dynamique toutes les données relatives au pays pour lequel je veux faire une évaluation de stock."
Cette carte affiche les pays, les zones et les stations présents dans la base de données. Il y'a un affichage adapté selon l'utilisateur, par exemple un utilisateur des Seychelles ne verra que les informations relatives à ce pays. Les administrateurs ont quant à eux, un affichage complet de toutes les informations, tous pays confondus.



Carte de la page d'accueil sous Google Map (version administrateur)

“En tant qu’utilisateur, je veux que lorsque j’ajoute ou j’importe une zone, celle-ci soit sauvegardée sur une carte statique afin de pouvoir la consulter même si je ne suis pas connecté à internet.”

Les cartes statiques sont générées lors de l’importation ou de l'ajout de zones ou de stations. Une prévisualisation de la carte statique est également générée avant d’envoyer les données afin que l’utilisateur vérifie les données qu’il a ajouté.

Carte statique d’ajout d’une zone

Cependant, lors du mois de Mai, Mme Fiat et moi même, avons appris que Google souhaitait limiter le nombre de requêtes gratuites par utilisateur. Cela posait donc des problèmes pour une bonne utilisation de l’application, nous avons donc décidé de changer de service cartographique. Nous avons donc opté pour Mapbox* qui a été intégré dans l’application par ma tutrice Sylvie Fiat.

Durant cette même période nous avons eu une réunion avec Marc Léopold pour échanger sur le développement de l’application en adéquation avec le besoin métier. Il nous a demandé de nouvelles fonctionnalités par rapport aux cartes. D’une part, le fait que sur les cartes, doivent apparaître les informations de la zone/station sélectionnée ainsi que tous les autres éléments (zones, stations) présents autour avec une distinction pour les objets sélectionnés. D’autre part, le fait d’utiliser un fond de carte satellite, qui est plus précis par rapport à un type Open Street Map pour le marin. Enfin, lorsque l’on arrive sur une page, la carte doit s’ajuster

automatiquement sur les pays, zones ou stations en fonction des objets sélectionnés. Nous avons ajouté ces nouvelles fonctionnalités sur toutes les cartes.

Bien que Mapbox propose autant de service voir plus que Google Map, j'ai eu des problèmes pour les cartes statiques. En effet, les zones étant des polygones et donc composées d'une succession de point, et les requêtes statiques de Mapbox n'acceptant qu'un certains nombres de points, j'ai été bloqué par cette restriction pour les zones les plus grandes. Nous avons prit la décision de laisser de côté les cartes statiques pour cette version de l'application et donc d'utiliser uniquement des cartes dynamiques.
Parmi les cartes dynamiques on distingue deux types, les premières sont des cartes qui affichent les données présentes dans la base et les secondes qui reprendront cela, et ajoutent de manière dynamique les données que l'utilisateur est en train de rentrer. Voici donc la liste des cartes du premier type ainsi que leurs fonctionnalités :

- Carte globale sur la page d'accueil

"En tant qu'utilisateur, je veux que sur ma page d'accueil soit affichées sur une carte dynamique toutes les données relatives au pays pour lequel je veux faire de la gestion de stock ».
« En tant qu'utilisateur, Je veux pouvoir choisir les couches de données qui sont affichées sur la page d'accueil ainsi que le type du fond de carte »
Cette carte est identique à celle qui a été faite avec Google Maps, avec les améliorations demandées par Marc Léopold.

Carte de la page d'accueil

- Carte d'une plateforme

"En tant qu'utilisateur, je veux que lorsque je sélectionne une plateforme la carte s'ajuste automatiquement aux zones de cette dernière et affiche les zones et les stations qui lui sont rattachées"

Carte d'une plateforme avec des zones et des stations

- Carte d'une zone

"En tant qu'utilisateur, je veux que lorsque je sélectionne une zone, la carte s'ajuste à cette dernière et la démarque des autres. Je veux également que toutes les stations et zones rattachées à la même plateforme soient affichées."

Carte d'une zone

- Carte d'une station

"En tant qu'utilisateur je veux que lorsque je sélectionne une station, la carte s'ajuste à la zone dans laquelle la station se trouve et la démarque des autres. Si la station ne se trouve dans aucune zone, je veux que la carte s'ajuste à la station. Je veux que la station soit démarquée des autres. Je veux également que toutes les stations et zones rattachées à la même plateforme soient affichées."

Carte d'une station

- Carte d'un relevé

"En tant qu'utilisateur, je veux que lorsque j'affiche un relevé, la carte s'ajuste aux stations liées aux comptages de ce dernier. Je veux que ces stations soient démarquées des autres. Je veux également que toutes les stations et zones rattachées à la même plateforme soient affichées."

Carte d'un relevé

- Carte d'un comptage

"En tant qu'utilisateur, je veux que lorsque j'affiche un comptage, la carte s'ajuste à la zone dans laquelle la station liée au comptage se trouve. Si la station ne se trouve dans aucune zone, je veux que la carte s'ajuste à la station. Je veux que la station soit démarquée des autres. Je veux également que toutes les stations et zones rattachées à la même plateforme soient affichées."

Carte d'un comptage

Je vais maintenant vous présenter le second type de carte :

Carte d'ajout de stations (import et formulaire)

"En tant qu'utilisateur, je veux que lorsque j'arrive sur l'ajout de station, la carte s'ajuste sur les stations de la plateforme. Je veux que lorsque j'ajoute des données, les nouvelles stations soient affichées en vert sur la carte si elles sont dans une zone et en orange dans le cas contraire. Je veux que la carte s'ajuste en fonction des données que j'ai rentré. Je veux également que toutes les stations et zones rattachées à la même plateforme soient affichées."

Carte d'ajout de station (prévisualisation import)

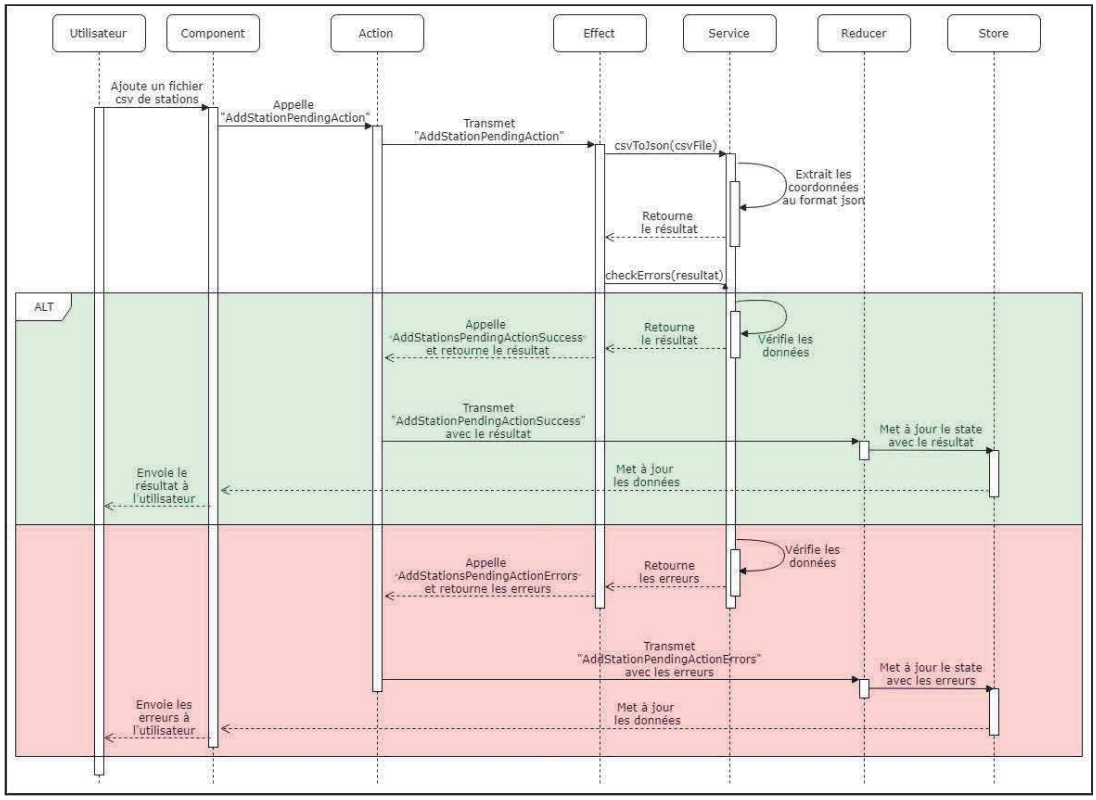


Diagramme de séquence de prévisualisation de stations

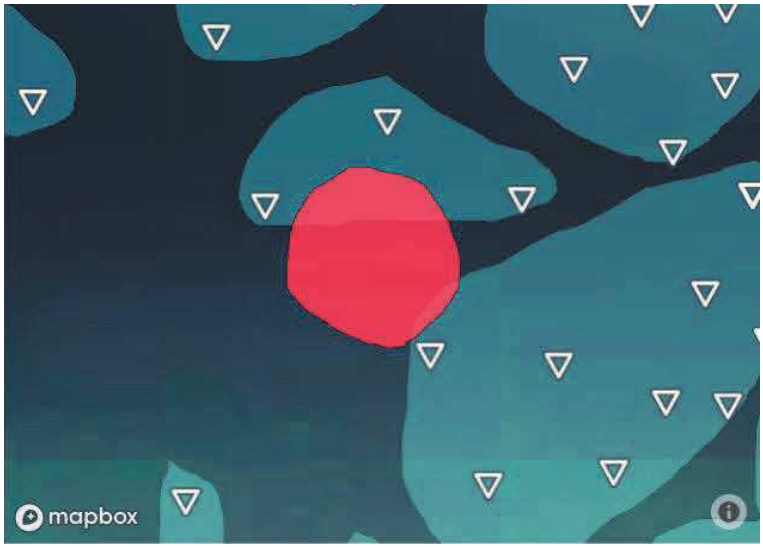
- Carte d'import de comptages

"En tant qu'utilisateur, je veux que lorsque j'importe des comptages, la carte s'ajuste aux stations liées aux nouveaux comptages et que ces derniers soient démarqués. Je veux également que toutes les stations et zones rattachées à la même plateforme soient affichées."

Carte d'import de comptage (prévisualisation)

- Carte d'ajout de zone (formulaire)

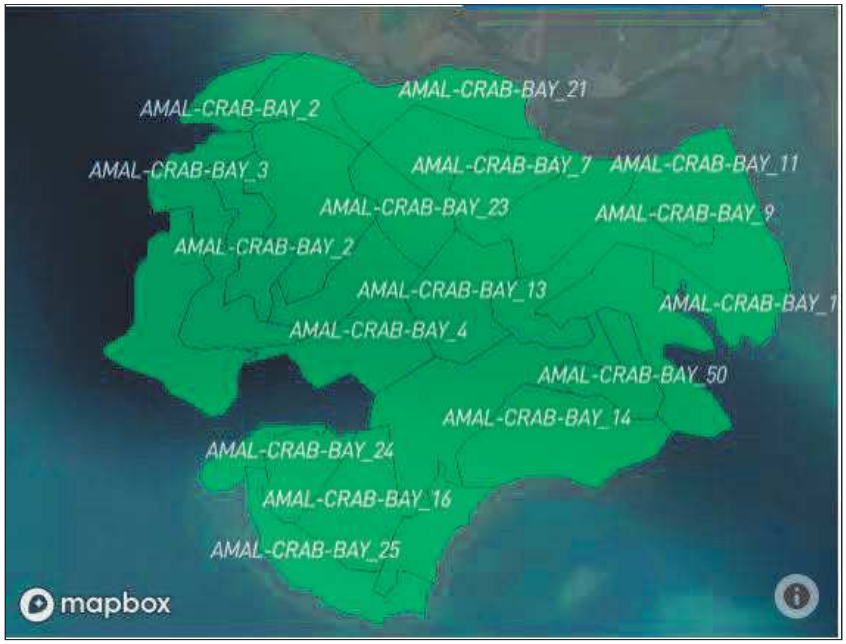
"En tant qu'utilisateur, je veux que lorsque j'ajoute une zone, la carte s'ajuste à cette dernière. Je veux également que la zone soit affiché en rouge si elle intersecte une zone déjà existante de la plateforme et en vert dans le cas contraire. Je veux également que toutes les stations et zones rattachées à la même plateforme soient affichées."



Carte d'ajout d'une zone non valide (prévisualisation)

- Carte d’ajout de zone (import)

“En tant qu’utilisateur, je veux que lorsque j’ajoute un fichier contenant des zones, la carte s’ajuste aux nouvelles zones et affichent seulement ces dernières.”



Carte d’import de zones

Pour conclure cette partie, nous avons eu des problèmes ne dépendants pas forcément de nous, mais nous avons réussi à trouver des solutions. Ces cartes sont la plus grosse partie en terme de temps de mon stage car il y’a d’abord eu le changement de service géographique (SIG*), les nouvelles fonctionnalités et enfin tous les petits ajustements à faire. Cela m’a permis de me familiariser avec un autre service de géographique que Google Maps. Une des évolutions possibles est l’utilisation de fonds de carte que Marc Léopold fournira, qui sont beaucoup plus précis.

3. Import de fichiers

L'une des principales missions de l'application est de bancariser des données, pour cela l'utilisateur a deux possibilités, importer des fichiers ou saisir les données à la main. Dans cette partie je vais présenter la première possibilité. Je vais donc parler de l'import de zones, stations, relevés et comptages. Ces fonctionnalités existants déjà, j'ai donc été amené à y apporter des modifications. D'autre part, j'ai ajouté un outil d'import plus général que j'expliquerai.

"En tant qu'utilisateur, je veux que lorsque j'importe des zones une vérification sur la validité de mes données soit faite et que l'on m'affiche les erreurs si il y en a."

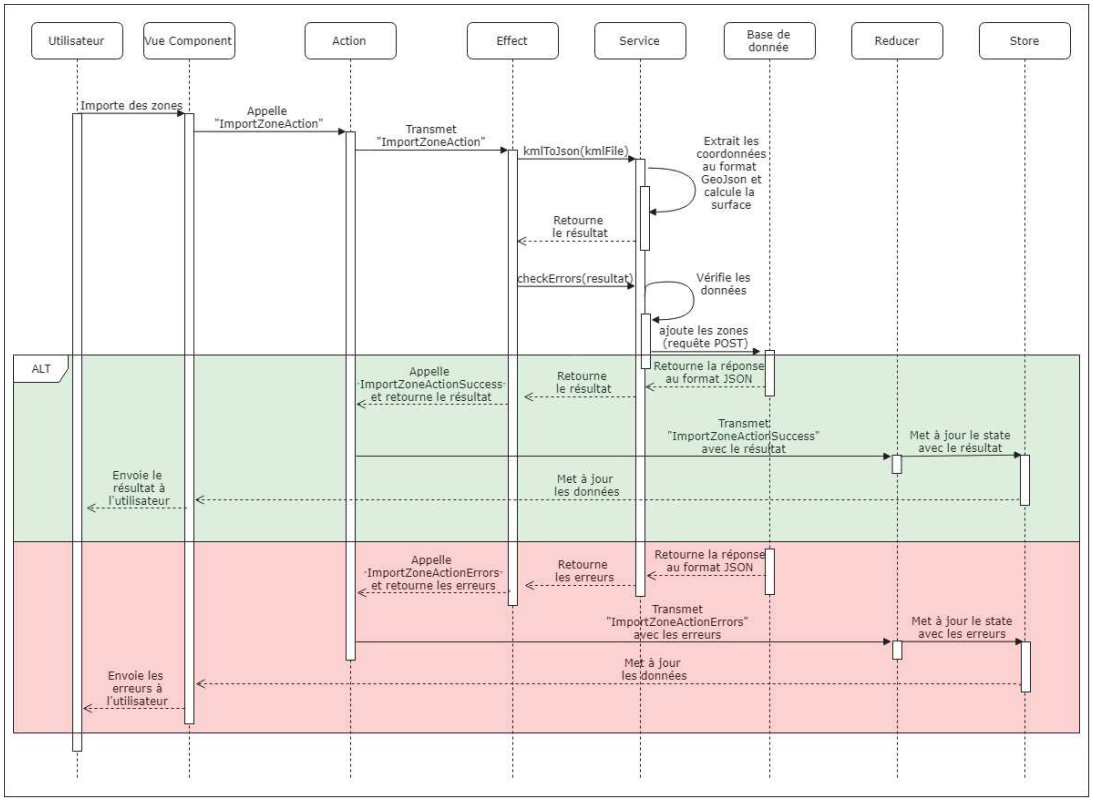
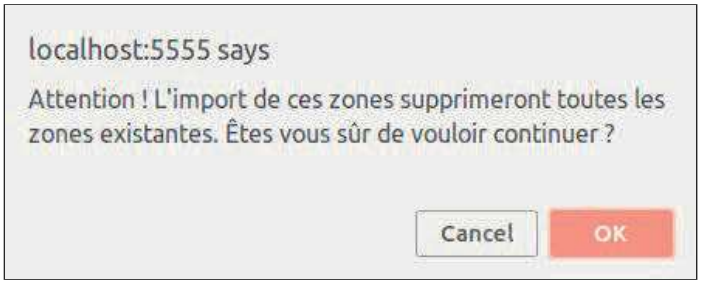


Diagramme de séquence d'import de zone

L’import de zone est un outil qui permet aux utilisateurs d’importer une collection de zones en une seule fois par le biais d'un fichier CSV*. La première tâche a été de changer le type de fichier à importer afin de pouvoir intégrer la composante géographique qui permet d'afficher les zones sur les cartes interactives. On est donc passé d’un fichier au format CSV à un fichier au format KML*. Le format est plus approprié car il permet de préparer les zones sur un outil SIG (par exemple Google Earth) puis d’exporter son fichier au format KML. Lorsque que l’utilisateur ajoute son fichier, une vérification de la validité du fichier est effectuée puis une transformation des données en un objet GeoJSON* est faite et les zones en cours de création sont affichées sur une carte interactive. Si tout est valide, l’utilisateur peut envoyer ses données. L’import de zones sur une plateforme est conçu pour être effectué en une seule fois, ainsi si l’utilisateur cherche a importer des zones dans une plateforme possédant déjà des zones ces dernières seront écrasées par les nouvelles. Lors de l’import, une fenêtre informera l’utilisateur que l’ajout de ses données entraînera la suppression de toutes les zones existantes le cas échéant et demandera une confirmation. S’il choisit de continuer, la surface de chaque zone sera calculée afin que les utilisateurs puissent utiliser cette donnée, utile au calcul de biomasses notamment, puis les données seront insérées dans la base.



Message de confirmation d’import de zone

“En tant qu’utilisateur, je veux que lorsque j’importe des stations une vérification sur la validité de mes données soit faite et que l’on m’affiche les erreurs si il y en a.”

Pour l’import de station, l’utilisateur ajoute un fichier CSV. Une vérification sur le fait que le fichier contient les champs attendus est effectuée. Si le fichier est valide, ses données sont transformées en un objet GeoJSON et affiché sur une carte. Sinon l’utilisateur est informé des

erreurs ligne par ligne ainsi que la cause de l'erreur. Si une station est en dehors d'une zone, un warning est affiché. L'utilisateur peut envoyer ses données dans la base si il n'y a pas d'erreur.

Message d'avertissement

Exemple d'erreurs

"En tant qu'utilisateur, je veux que lorsque j'importe des relevés une vérification sur la validité de mes données soit faite et que l'on m'affiche les erreurs si il y en a."

Pour l'import de relevé, l'utilisateur ajoute un fichier CSV. Une vérification sur le fait que le fichier contient les champs attendus et que les champs station et zone correspondent à des données existantes en base, est effectuée. Si il y a des erreurs, elles sont détaillées ligne par ligne à l'utilisateur avec la cause de l'erreur..
"En tant qu'utilisateur, je veux que lorsque j'importe des comptages une vérification sur la validité de mes données soit faite et que l'on m'affiche les erreurs si il y en a."

L'import de comptage est celui où le plus de vérifications est nécessaire. En effet, une vérification de la validité des champs du fichier CSV est effectuée, ainsi qu'une vérification de l'existence des données suivantes dans la base de données : la plateforme, les stations, les relevés et les espèces. Ces vérifications sont nécessaires car un comptage est relié à toutes ces données. Si tout est validé, l'utilisateur pourra envoyer son fichier, sinon il sera informé des erreurs de chaque ligne du CSV.

"En tant qu'utilisateur, je veux pouvoir importer des stations, des relevés et des comptages en une seule fois sous forme d'étapes. Je veux pouvoir baser mes données sur celles que j'ai ajouté à l'étape précédente. Je veux que lorsque j'ajoute un fichier une vérification sur la validité de mes données soit faite et que l'on m'affiche les erreurs si il y en a."

L'outil d'import général permet d'importer dans une même action des stations, des relevés et des comptages par le biais d'étapes d'import. Chacun de ces imports héritent de ce qui à été chargé dans les étapes précédentes. La principale fonctionnalité de cet outil est que lorsque l'on importe un fichier et qu'il est valide, un ajout des données au store de Ngrx est effectué pour que les étapes suivantes puissent baser leurs vérifications de données sur les précédentes. Ce fut assez complexe à réaliser car il faut gérer tout les cas de possibles tel que le fait que si l'utilisateur importe un fichier avec des erreurs et veut passer à une étape suivante, il ne faut pas enregistrer dans le store le fichier erroné, pour remédier à cela, l'utilisateur est informé que s'il souhaite changer d'étape, le fichier comportant des erreurs sera supprimé. D'autre part, les comptages étant basés sur les stations et relevés, si

l'utilisateur ajoute un fichier de comptage valide basé sur des fichiers stations et relevés importés dans les étapes précédentes puis qu'il décide de changer le fichier de relevés ou stations, la validité des comptages ne pouvant plus être assurée, il est informé que cette action supprimera également son fichier de comptage.

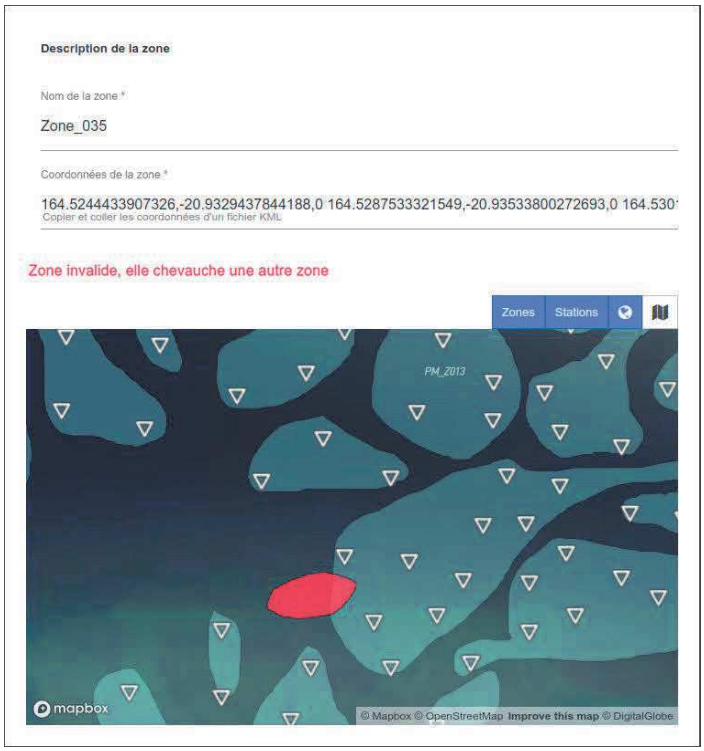
Outil d'import global

4. Formulaires de données

Les formulaires de données sont une alternative aux imports. Il y a six formulaires différents : pour une créer zone, une station, une espèce, un relevé, un comptage et enfin une plateforme. Je parlerai ici des trois premiers qui sont ceux sur lesquels j'ai travaillé.

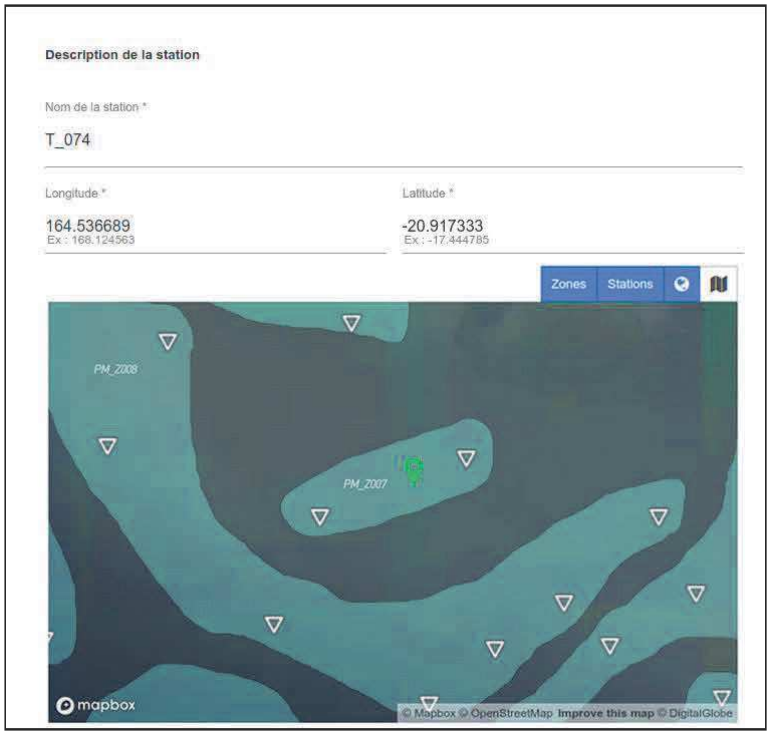
"En tant qu'utilisateur, je veux pouvoir ajouter une zone en rentrant le nom et les coordonnées aux formats:
Latitude,Longitude,Altitude Latitude,Longitude,Altitude ... Latitude,Longitude,Altitude
Je veux également que l'on vérifié si les coordonnées forment bien un polygone."

Ce format permet de prendre les coordonnées présentes dans un fichier KML et de les coller dans le champ prévu.



Formulaire d'ajout d'une zone (prévisualisation)

"En tant qu'utilisateur, je veux pouvoir ajouter une station en entrant son nom, sa latitude et sa longitude. Je veux que la validité de mes coordonnées soit vérifiée et que l'on m'affiche si la station se situe dans une zone ou non."



Formulaire d'ajout d'une station (prévisualisation)

"En tant qu'utilisateur, je veux pouvoir ajouter une photo d'une espèce et la prévisualiser."
Le formulaire d'ajout d'espèce étant déjà presque complet, il ne manquait que l'ajout d'image. J'ai donc rajouté un champ dans le formulaire pour envoyer une image. Une fois l'image ajoutée, l'utilisateur a une prévisualisation de son image. Quand l'utilisateur a fini de remplir le formulaire et l'envoie, l'image va être convertie en base64 à l'aide des promesses, d'ajax et btoa.

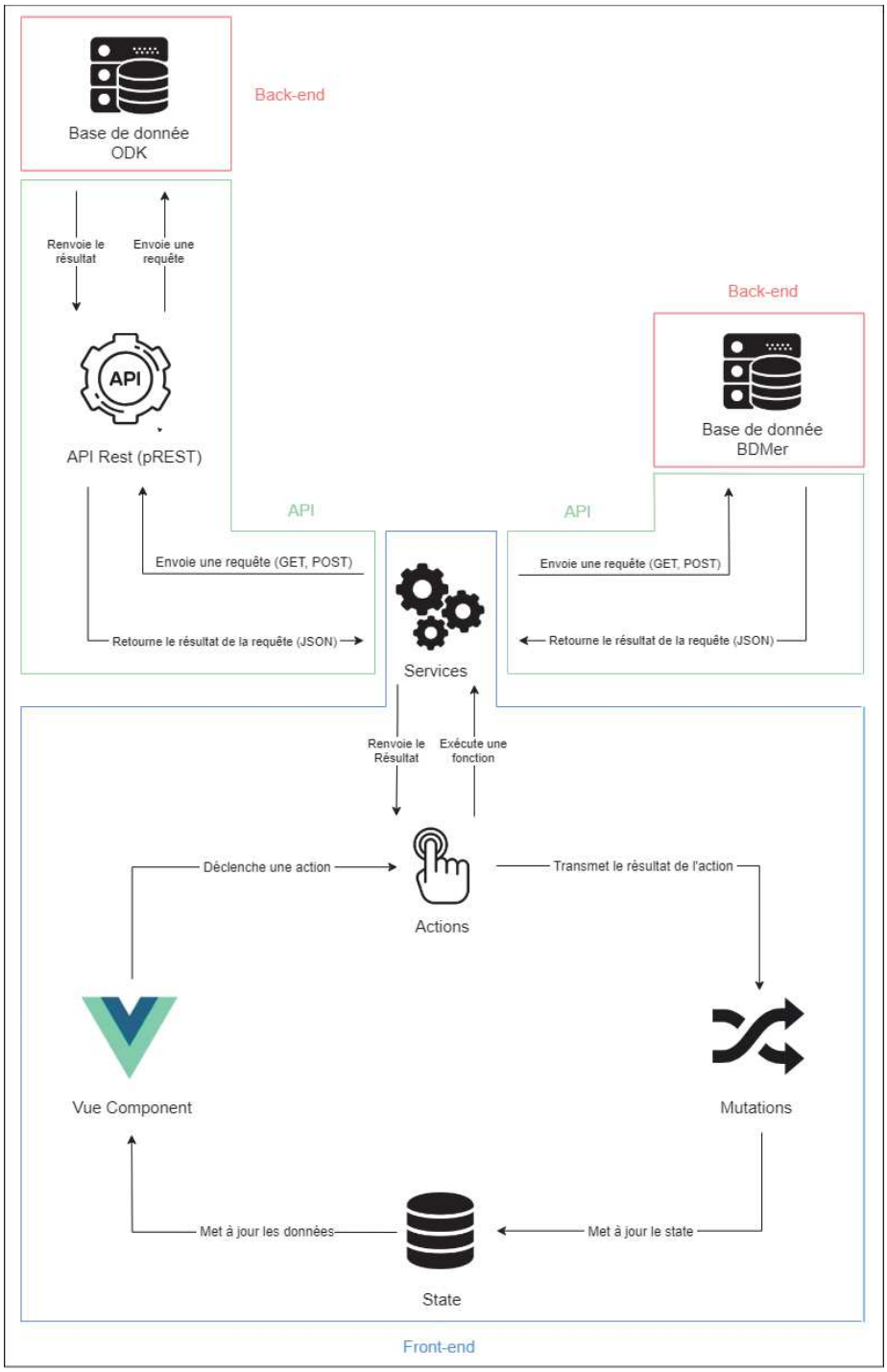
5. Application d'import de données BDMer Tools

L'application d'import de données est application annexe dont le besoin avait été exprimé par Marc Léopold en réunion afin de pouvoir convertir les données issues d'une première application, que nous appellerons Seacusey*, utilisée par les pécheurs des Seychelles vers l'application BDMer. J'ai effectué ce travail pendant que ma tutrice effectuait la migration de l'application principale BDMer d'angular 4 à angular 6. Ma mission est de concevoir une application bureau et web qui permet d'importer les données depuis une base de données Postgres* utilisée par l'application Seacusey vers la base Couchdb de BDMer.
Pour répondre à cette problématique, j'ai fait une étude et proposé trois solutions techniques répondant au problème posé que j'ai du justifier :

- Première proposition : développement avec ElectronJs, Angular et NgRx. Pros : garder une cohérence avec BDMer qui utilise ces technologies. Cons : le temps de mise en place et la complexité sont pénalisants pour une application légère, censée être rapidement mise en place.
- La seconde proposition : faire une application NodeJS. Pros : Rapide à développer grâce à un boilerplate. Cons : il est très compliqué de convertir une application NodeJS en application bureau et on perd énormément en expérience utilisateur, rapidité et fluidité.
- La troisième solution : faire une application ElectronJS et vueJs* utilisant le boilerplate electron-vue. Pros : VueJs étant plus léger et rapide que Angular, et ayant déjà travaillé plusieurs fois sur ce framework. Cons : ne plus avoir de cohérence technologique avec BDMer.

Nous avons retenu la troisième solution car elle présentait le plus d'avantages et que, ayant déjà travaillé dans ces technologies, je pouvais proposer un travail rapide et efficace pour mettre en place petite application.

a. Structure de l'application



Workflow de l'application BDMer Tools

Vue component : Composant vue permettant l'affichage pour l'utilisateur et l'interaction avec les actions

Actions : Objet contenant des fonctions exécutants les fonctions des services et transmettant le résultat aux mutations via un "commit"

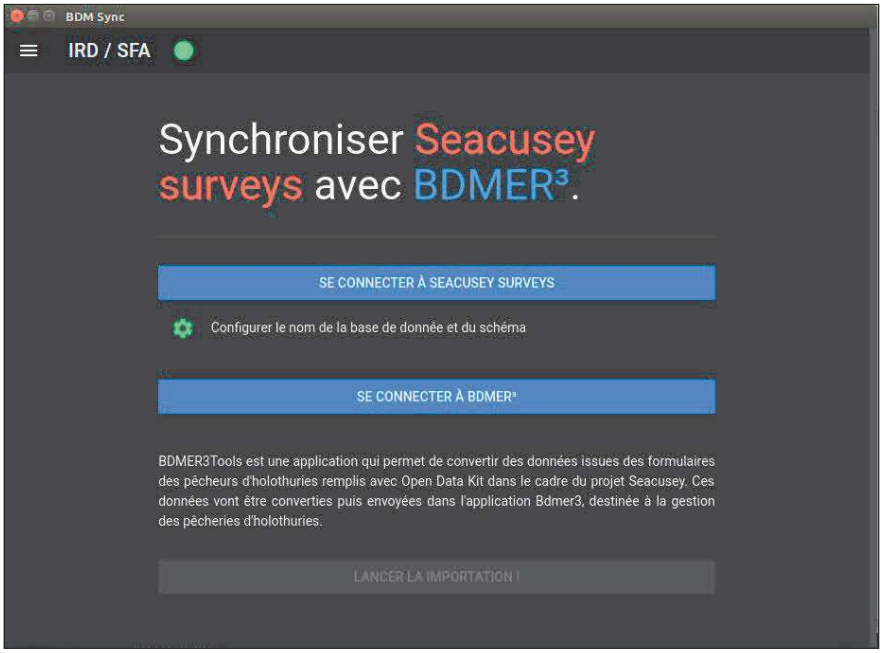
Mutations : Fonction qui prend en paramètre la state ainsi que la nouvelle valeur du state

State : Un objet javascript

Services : Des fonctions qui permettent de traiter les données

API Rest (pREST) : API Rest permettant d'interagir avec la base de donnée d'ODK

b. Fonctionnement de l'application



Page d'accueil de l'application

Le fonctionnement de l'application est simple. D'un côté il l'application Seacusey basée sur ODK Aggregate* et de l'autre l'application BDMer. Le but est que les données saisies par les pêcheurs via les formulaires ODK soient injectées dans BDMer. Dans un premier temps, l'utilisateur va paramétrer les connexions à Seacusey, puis à BDMer. Pour se connecter il doit fournir l'adresse du serveur, son nom de compte ainsi que le mot de passe. Il y a une

vérification de l'url (si elle est valide) ainsi qu'une vérification de connexion pour l'url fournie.

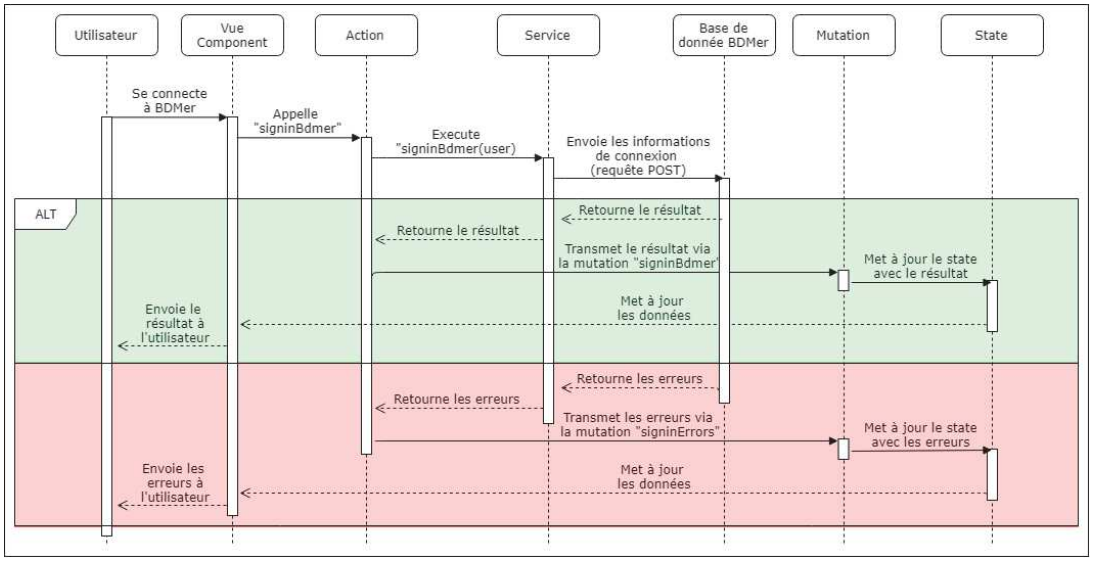


Diagramme de séquence de la connexion à BDMer

Connexion refusée à BDMer

Une fois connecté aux deux bases de données, l'utilisateur peut cliquer sur le bouton « Lancer l'importation » pour importer les données et l'application se charge du reste.

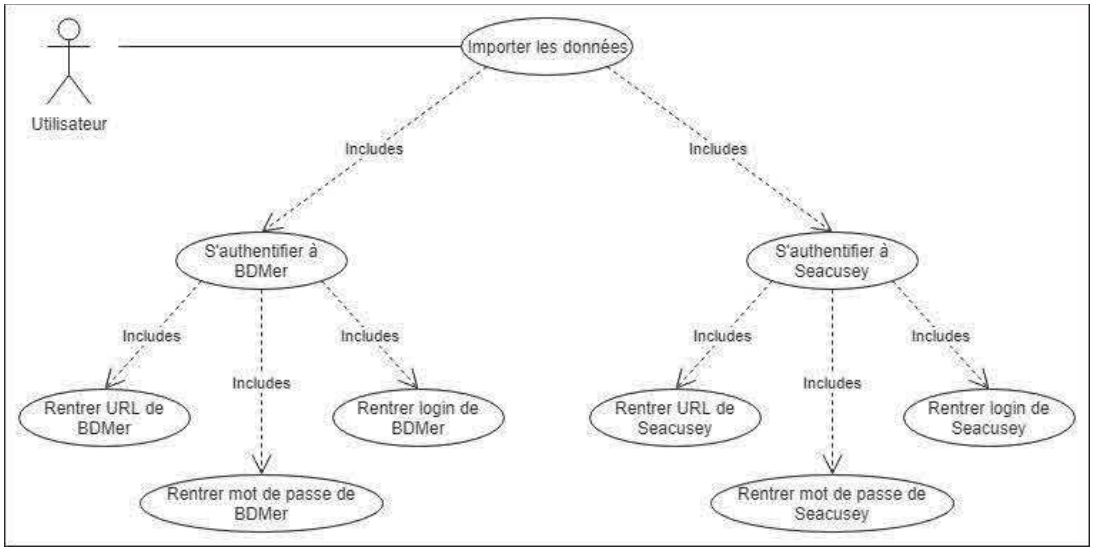
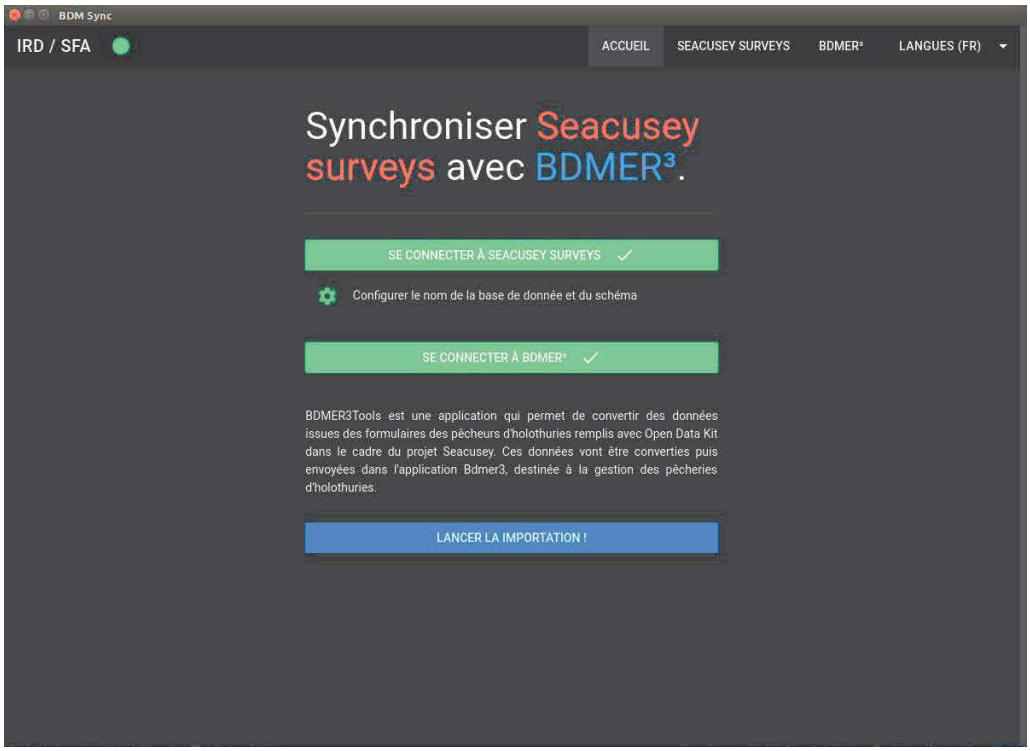


Diagramme de cas d'utilisation d'import de donnée de Seacusey vers BDMer



Une fois connecté l'utilisateur peut lancer l'importation

Pour se connecter à la base postgres, l'utilisateur n'utilise pas directement à un accès à la base de données Seacusey mais à une API Rest qui a été positionnée devant à la base Postgres. En effet, nous n'avons pas pu établir une connexion directe à la base de donnée car aucune solution existante (package npm*) n'était utilisable dans notre contexte. L'application Seacusey ainsi que sa base de donnée étant déployées dans des containers docker, il a été facile de rajouter un container déployant pREST* qui est un utilitaire qui crée une API Rest* mettant à disposition les données contenues dans la base de donnée Postgres. Cela permet donc d'interagir facilement avec la base de données via des requêtes Rest (voir *Diagramme de séquence d'importation de donnée de Seacusey vers BDMer)*.
De l'autre côté, avec la base de donnée CouchDB, les packages npm "pouchdb" et "pouchdb-authentification" sont utilisés et permettent d'interagir directement avec cette dernière. On va donc dans un premier temps avoir la connexion, puis la récupération des données des espèces et des plateformes. Ces données sont récupérées pour permettre une vérification entre ces dernières et celles fournies par les pêcheurs (vérification que les espèces et plateformes rentrées par les pêcheurs existent dans BDMer). L'application utilise les cookies afin de sauvegarder les données des connexions pour éviter à l'utilisateur de rentrer ces données de connexion à chaque fois. La dernière étape est donc l'import des données dans la base.

Lors de l'import vers BDMer, les données de Seacusey sont d'abord reformatées et il y a ensuite une gestion des erreurs pour informer l'utilisateur de l'état de son importation. Comme on peut le voir ci dessous il y'a trois "niveaux". Le premier (en vert) détaille toutes les données qui ont étés insérés dans la base de donnée de l'application BDMer. Le second (en jaune) va afficher toutes les données qui n'ont pas pu être importées car elles sont déjà présentes dans l'application BDMer. Le dernier (en rouge) montre les données qui n'ont pas pu être importées et pourquoi, par exemple si il y a une donnée manquante (longitude, latitude, plateforme etc..) ou si la plateforme ou encore l'espèce n'existe pas dans BDMer. Cette séparation permet à l'utilisateur de savoir exactement comme son importation s'est déroulée.

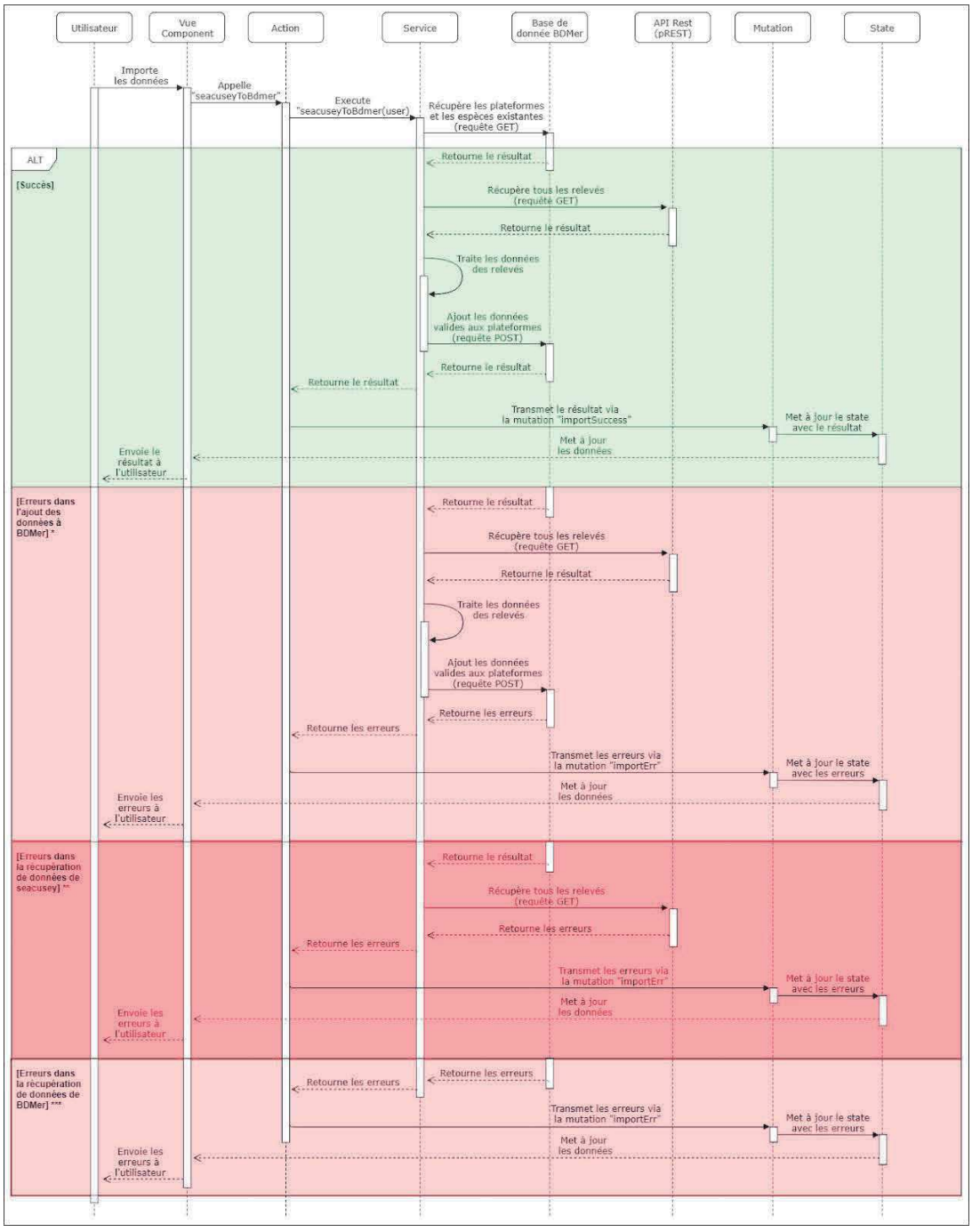
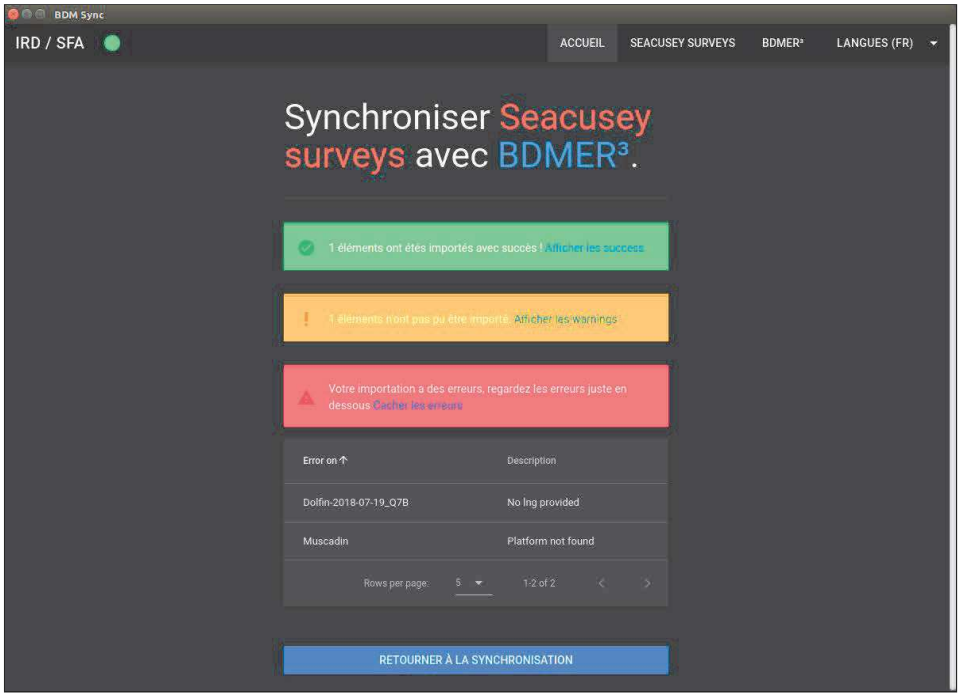


Diagramme de séquence d'importation de donnée de Seacusey vers BDMer

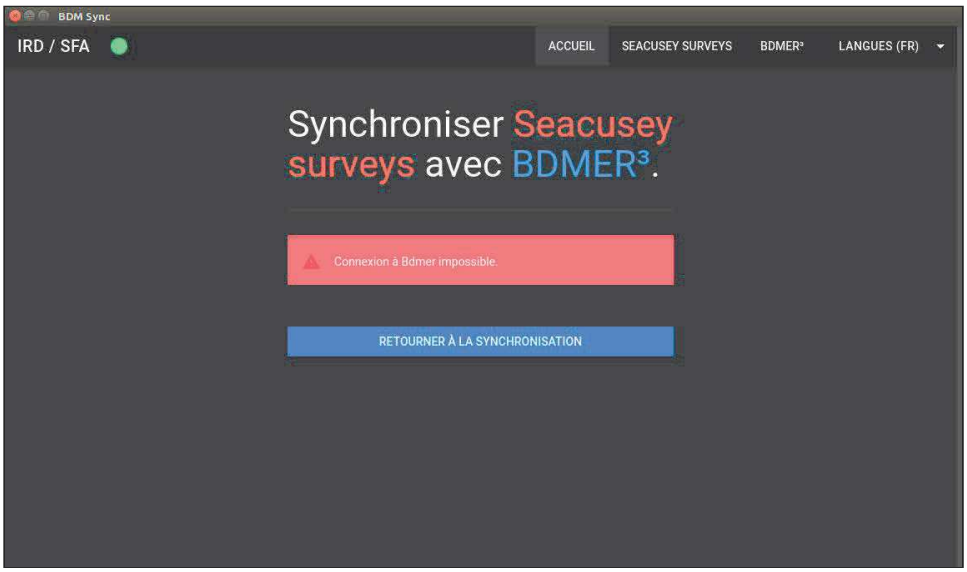
* Erreurs dans l'ajout des données à BDMer : ces erreurs se produisent lorsque le processus veut ajouter les données à BDMer et que BDMer n'est pas accessible

** Erreurs dans la récupération des données de Seacusey : ces erreurs se produisent lorsque le processus veut récupérer les relevés de Seacusey mais que ce dernier n'est pas accessible ou que les informations de connexion à Seacusey de l'utilisateur ne sont plus valides.

*** Erreurs dans la récupération des données de BDMer: ces erreurs se produisent lorsque le processus veut récupérer les plateformes et espèces existantes de BDMer mais que la base de donnée n'est pas accessible ou que les informations de connexion à BDMer de l'utilisateur ne sont plus valides.

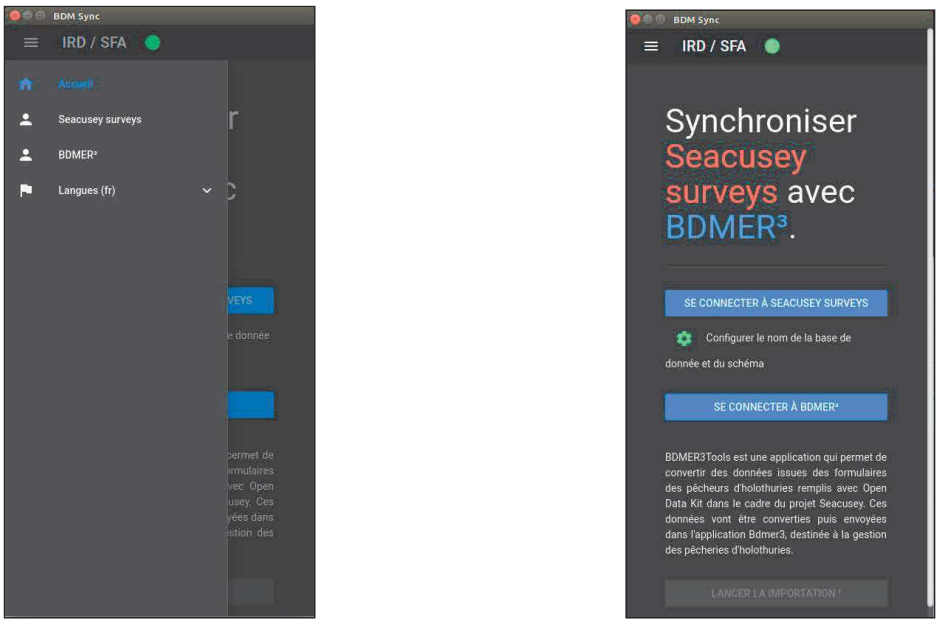


Résultat d'une importation de donnée



Résultat lorsque l'on lance une importation de donnée et que la connexion à BDMer échoue

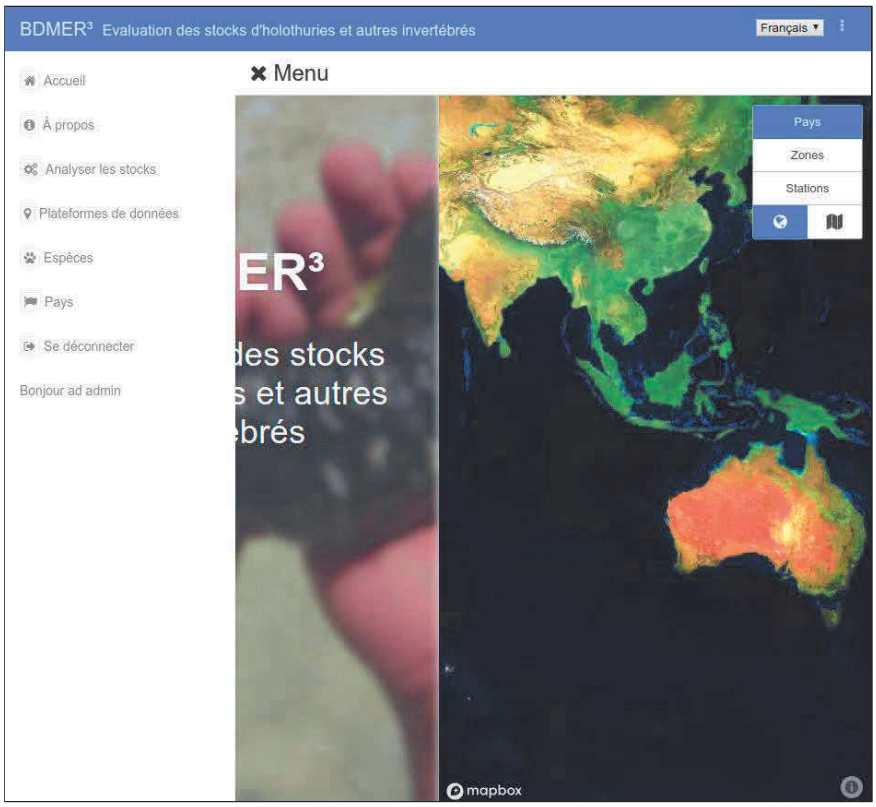
L'application dispose également d'un système pour savoir si l'utilisateur est en ligne ou non matérialisé par un rond vert dans la barre de navigation. Le rond se met à clignoter en orange lorsque l'on perd la connexion internet. Elle est totalement responsive pour permettre une expérience utilisateur agréable et dans le cas où l'application serait utilisée sur mobile.



Aperçu de l'application au format mobile

6. Autres tâches

Tout au long de mon stage, j'ai été amené à effectuer d'autres tâches qui ne rentraient pas forcément dans les parties précédentes. Par exemple, j'ai fait l'internationalisation des messages d'erreurs en anglais notamment. Autre exemple, le menu de l'application n'était pas responsive, j'ai donc utilisé les composants d'Angular Material qui est utilisé par l'application pour rendre cela possible.



Page d'accueil avec menu version mobile / tablette

IV. Conclusion

Tout au long du stage, nous avons travaillé de manière agile. Nous avons utilisé l'outil de gestion de projet "Trello" et voici donc un récapitulatif des cartes du tableau de bord :

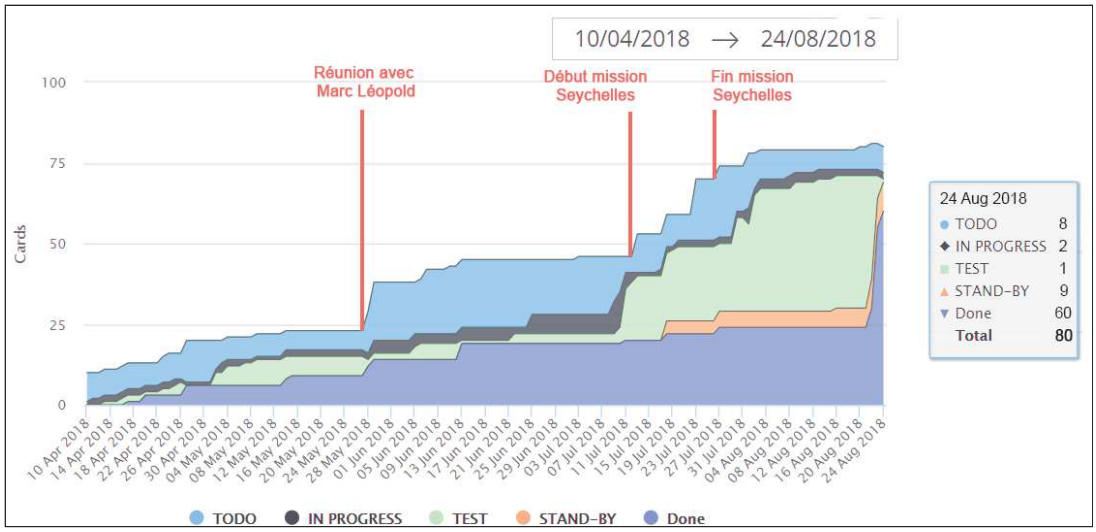


Diagramme de flux cumulatif des cartes du tableau de bord Trello
(70% des cartes me concernent dont 0 en 'TODO')

Ce stage a été très enrichissant pour moi et m'a permis, je pense, de bien me préparer pour le monde du travail. Il m'a également permis de confirmer ma voie, qui est celle des technologies du web. J'ai énormément appris auprès de ma tutrice, Sylvie Fiat, notamment sur le fait qu'il n'y a pas de problème que des solutions et que "ce n'est pas possible" n'est pas une réponse. Je pense que c'est une chose qui me servira toute ma vie et qui est ce que je retiendrai le plus de mon stage.

En dehors du travail, ce stage était l'opportunité pour moi de vivre une véritable expérience en sortant de ma zone de confort en étant loin de chez moi dans un lieu où je ne connaissais rien ni personne. Tout cela m'a aidé à renforcer mon mental. J'ai également découvert une façon de travailler totalement différente de ce à quoi j'ai été habitué. Passant de service informatique avec des dizaines de personnes à un service où il n'y que ma tutrice et moi même change beaucoup de choses et notamment la façon de travailler, qui est beaucoup plus autonome.

Ma mission qui portait donc sur l'application BDMer a été très gratifiante car j'ai le sentiment de m'être rendu vraiment utile car des personnes vont utiliser ce que j'ai développé. De plus j'ai pu forger mes connaissances et en apprendre d'autres.

Pour conclure ce rapport je tiens à remercier une dernière fois toutes les personnes qui m'ont aidés lors des mes études supérieurs et lors de ce stage. Maintenant un nouveau monde s'offre à moi, celui du travail, que j'ai hâte de découvrir pleinement.

Glossaire

Seacusey : Projet de recherche sur l'appui à la cogestion de la pêcherie d'holothuries aux Seychelles (http://umr-entropie.ird.nc/index.php/portfolio/projet-seacusey)

Angular : Angular est une plateforme de développement JavaScript qui permet de créer des applications web dynamiques et immersives (https://angular.io/)

VueJS : Vue.js est un framework JavaScript open-source pour créer des interfaces utilisateur. (https://vuejs.org/)

ElectronJS : Electron est un framework permettant de développer des applications multi-plateformes de bureau avec des technologies web. (https://electronjs.org/)

Docker : Docker est un logiciel libre qui automatise le déploiement d'applications dans des conteneurs logiciels. (https://www.docker.com/)

RxJS : RxJS est une librairie pour la programmation réactive à l'aide d'Observable pour produire du code asynchrone ou basé sur des "callback". (https://rxjs-dev.firebaseapp.com/)

CouchDB : Apache CouchDB est un système de gestion de base de données orienté documents, écrit en langage Erlang et distribué sous licence Apache. (http://couchdb.apache.org/)

PouchDB : PouchDB est une base de donnée inspirée par Apache CouchDB qui est une une base de donnée interne aux navigateurs utilisant indexdb (https://pouchdb.com/)

Mapbox : Mapbox développe un ensemble de technologies et d'outils cartographiques, dont la bibliothèque Mapbox.js basée sur Leaflet, l'éditeur Mapbox Studio successeur de TileMill, et le langage de feuille de style CartoCSS. Ces projets reposent en très grande partie sur le logiciel libre et sur les données d'OpenStreetMap. (https://www.mapbox.com/)

Postgres : Postgres est un système de gestion de base de données relationnelle et objet. (https://www.postgresql.org/)

IndexedDB : IndexedDB est une API de stockage côté client qui permet de gérer des quantités importantes de données structurées, incluant des fichiers/blobs. Cette API utilise des indexes afin de permettre des recherches performantes sur ces données.
(https://developer.mozilla.org/fr/docs/Web/API/API_IndexedDB)

KML : KML (Keyhole Markup Language) que l'on peut traduire par « langage à base de balises géolocales », est un langage fondé sur le formalisme XML et destiné à la gestion de l'affichage de données géospatiales dans les logiciels de SIG.
(https://fr.wikipedia.org/wiki/Keyhole_Markup_Language)

SIG : Un système d'information géographique (SIG) est un système d'information conçu pour recueillir, stocker, traiter, analyser, gérer et présenter tous les types de données spatiales et géographiques. (https://fr.wikipedia.org/wiki/Système_d%27information_géographique)

base64 : En informatique, base64 est un codage de l'information utilisant 64 caractères, choisis pour être disponibles sur la majorité des systèmes.
(https://fr.wikipedia.org/wiki/Base64)

CSV : Comma-separated values, connu sous le sigle CSV, est un format informatique ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules. (https://fr.wikipedia.org/wiki/Comma-separated_values)

SVG : Le Scalable Vector Graphics (en français « graphique vectoriel adaptable »), ou SVG, est un format de données conçu pour décrire des ensembles de graphiques vectoriels et basé sur XML. (https://fr.wikipedia.org/wiki/Scalable_Vector_Graphics)

GeoJSON : GeoJSON (de l'anglais Geographic JSON, signifiant littéralement JSON géographique) est un format ouvert d'encodage d'ensemble de données géospatiales simples utilisant la norme JSON (JavaScript Object Notation).
(https://fr.wikipedia.org/wiki/GeoJSON)

NoSQL : En informatique et en bases de données, NoSQL désigne une famille de systèmes de gestion de base de données (SGBD) qui s'écarte du paradigme classique des bases relationnelles. (https://fr.wikipedia.org/wiki/NoSQL)

Seychelles Fishing Authority : La Seychelles Fishing Authority est une organisation para-étatique créée en 1984. Elle a pour mission de développer la pêcherie seychelloise (artisanale et semi-industrielle) dans un cadre de pêche raisonnable et responsable.

btoa : La méthode WindowOrWorkerGlobalScope.btoa() crée une chaîne ASCII codée en base 64 à partir d'un objet String dans lequel chaque caractère de la chaîne est traité comme un octet de données binaires.
(https://developer.mozilla.org/fr/docs/Web/API/WindowBase64/btoa)

API Rest : REST (Representational State Transfer) est un style d'architecture définissant un ensemble de contraintes et de propriétés basées sur le protocole HTTP. Les services web conformes au style d'architecture REST, aussi appelés services web RESTful, établissent une interopérabilité entre les ordinateurs sur Internet.
(https://fr.wikipedia.org/wiki/Representational_state_transfer)

ODK : Open Data Kit (ODK) un générateur d'applications Open Source pour créer des masques de saisie et applications personnalisées sur smartphones Android et accessible hors connexion. (https://opendatakit.org/)

pREST : pREST est un utilitaire permettant de créer une api Rest permettant d'interagir avec une base de donnée postgres (https://postgres.rest/)

package npm : npm est le gestionnaire de paquets officiel de Node.js. Depuis la version 0.6.3

de Node.js, npm fait partie de l'environnement et est donc automatiquement installé par défaut. npm fonctionne avec un terminal et gère les dépendances pour une application. Il permet également d'installer des applications Node.js disponibles sur le dépôt npm. (https://www.npmjs.com/)

FICHE RAPPORT DESTINEE A LA BIBLIOTHEQUE

RAPPORT CONFIDENTIEL ET NE DEVANT PAS FIGURER A LA BIBLIOTHEQUE :

☐ Oui ☒ non

NOM ET PRENOM DE L'ETUDIANT : Dubois Morgan

DUT : INFORMATIQUE

□ S4 □ S4 bis □ Année Spéciale/AETP

LICENCE PROFESSIONNELLE

☐ ASRALL ☒CISIIE

_____

TITRE DU RAPPORT : Réalisation de la version 3 d'une application d'aide à la gestion des pêcheries d'holothuries

Nom de l'Entreprise : IRD

Adresse : 101 Promenade Roger Laroque, Noumea 98848, Nouvelle-Calédonie

Type d'activité (domaines couverts par l'entreprise) : Institut de Recherche sur les relations entre l'homme et son environnement en Afrique, Méditerranée, Amérique latine, Asie et dans l'Outre-Mer tropical français.

Nom du parrain (enseignant IUT) : Gérôme Canals

Mots-clés (sujets traités) : Progressive Web App, Angular, ngrx, Gestion des pêcheries d'holothurie, vuejs

---

Résumé

Développement de la version 3.0 d'une application d'aide à la gestion des pêcheries d'holothuries, en collaboration avec les Seychelles Fishing Authority et l'association des pêcheurs professionnels. Le développement est basé sur les nouvelles technologies telles que docker, angular4-6, electron, nativescript, pouchdb, ngrx.