Stagiaire : Basile André

Tuteur entreprise : Rémi Bouilly

Tuteur académique : Jean-François Favennec

b1andre@enib.fr

Basile André

Remerciements

Remerciements

Je remercie en particulier mon tuteur en entreprise, M. Rémi Bouilly, pour son accompagnement, sa disponibilité et la confiance accordée tout au long du stage. Je remercie également mon tuteur académique, M. Jean-François Favennec, pour ses conseils et son suivi pédagogique.

Enfin, je remercie l'ensemble des agents de la DEAL Réunion, et plus particulièrement l'équipe de l'Unité Biodiversité, pour leur accueil et leur collaboration.

Résumé (FR)

Au cours de ma formation d'ingénieur généraliste à l'ENIB (École Nationale d'Ingénieurs de Brest), j'ai eu l'opportunité d'explorer des domaines variés, dont l'informatique, le traitement des données et la communication réseau. Ces enseignements, combinés au contexte environnemental actuel, m'ont donné l'envie d'explorer comment les outils numériques peuvent répondre à des enjeux concrets, en particulier ceux liés à l'environnement et au milieu marin, un domaine qui m'inspire depuis longtemps. C'est cette motivation qui m'a conduit à choisir la DEAL (Direction de l'Environnement, de l'Aménagement et du Logement), où le développement d'outils informatiques joue un rôle clé dans la valorisation et l'accessibilité des données, offrant une occasion idéale d'allier mes compétences techniques à un sujet qui a du sens.

J'ai effectué mon stage au sein de la DEAL Réunion, plus précisément au SEB (Service Eau et Biodiversité), à l'UBIO (Unité Biodiversité). L'île de La Réunion présente des enjeux importants de suivi et de préservation de la biodiversité, en raison de la singularité de ses écosystèmes et des pressions qu'ils subissent (notamment les EEE (Espèces Exotiques Envahissantes) et le développement du territoire). Ce contexte renforce la nécessité d'un suivi rigoureux des données naturalistes.

La DEAL utilise Géonature (Application de gestion et de centralisation des données naturalistes) pour centraliser et gérer ces données. Le sujet principal de mon stage consistait à développer un module externe permettant d'y importer automatiquement les données marines issues de Quadrige (système d'information de l'Ifremer dédié aux données environnementales marines). Ces données, essentielles pour le suivi environnemental littoral, représentent un enjeu majeur pour les missions de la DEAL.

N'ayant pas suivi le module CAI (Conception d'Applications Interactives), ce stage m'a permis d'acquérir des compétences complémentaires en développement logiciel et applicatif, à travers la conception d'un module d'import, la manipulation de données et l'interfaçage de systèmes hétérogènes, afin d'obtenir un outil simple d'utilisation, robuste et maintenable.

Il m'a également offert une immersion dans le fonctionnement d'une administration publique de taille moyenne, où l'alternance entre travail autonome et collaboration avec les agents en charge des données naturalistes et de l'administration de Géonature s'est révélée particulièrement enrichissante. Cette expérience en environnement professionnel réel a renforcé ma compréhension des enjeux liés à la gestion des données environnementales et a affiné ma capacité à questionner la pertinence, l'utilité et le sens des projets auxquels je contribue, afin de m'assurer qu'ils soient en accord avec mes valeurs et porteurs d'un impact positif.

Abstract (EN)

As part of my general engineering studies at ENIB, I had the opportunity to explore a wide range of fields, including computer science,

data processing and network communications. These courses, combined with the current environmental context, strengthened my interest in understanding how digital tools can address real-world challenges — particularly those related to the environment and the marine domain, which has long inspired me. This motivation led me to choose DEAL, where the development of digital tools plays a key role in enhancing the value and accessibility of environmental data, offering an ideal opportunity to align my technical skills with a topic that holds meaning for me.

I completed my internship at DEAL Réunion, more specifically within the SEB and the UBIO. Réunion Island faces major biodiversity challenges due to its high proportion of endemic species and the presence of particularly sensitive natural habitats. These ecosystems are under strong pressure, notably from EEE and ongoing land-use development, which reinforces the need for rigorous naturalist data. DEAL uses Géonature to centralise and manage these datasets. The main objective of my internship was to develop an external module for this application, enabling the automated import of marine and coastal environmental data produced by Ifremer (Institut Français de Recherche pour l'Exploitation de la Mer). These data are essential for biodiversity monitoring and are accessed through Quadrige,Ifremer's dedicated information system.

Since I had not taken the CAI module, this internship allowed me to acquire complementary skills in software and application development through the design of an import module, data manipulation and the interfacing of heterogeneous systems, with the aim of producing a simple, robust and maintainable tool.

It also offered me insight into the functioning of a medium-sized public administration, where the alternation between autonomous work and collaboration with staff in charge of naturalist data and the administration of Géonature proved particularly enriching. This experience in a real professional environment strengthened my understanding of the challenges associated with environmental data management and refined my ability to reflect on the relevance, usefulness and purpose of the projects I contribute to, ensuring that they align with my values and have a positive impact.

Table des matières

Glossaire

API	Interface de Programmation de l'Application
api2GN	module Géonature permettant la création et l'utilisation de parseurs
Borbonica	Plateforme régionale de diffusion des données naturalistes du SINP
CAI	Conception d'Applications Interactives
DEAL	Direction de l'Environnement, de l'Aménagement et du Logement
EEE	Espèces Exotiques Envahissantes
ENIB	École Nationale d'Ingénieurs de Brest
ERC	Éviter – Réduire – Compenser
Géonature	Application de gestion et de centralisation des données naturalistes
Ifremer	Institut Français de Recherche pour l'Exploitation de la Mer
OFB	Office Français de la Biodiversité
Pl@ntNet	Application mobile de reconnaissance des plantes par photo
PNRun	Parc National de La Réunion
Quadrige	système d'information de l'Ifremer dédié aux données environnementales marines
RNE	Responsabilité Numérique des Entreprises
RSE	Responsabilité Sociétale des Entreprises
SEB	Service Eau et Biodiversité
SEOR	Société d'Études Ornithologiques de La Réunion
SINP-974	Système d'Information de l'iNventaire du Patrimoine naturel de la Réunion
SREPEN	Société réunionnaise pour l'étude et la protection de la nature
UBIO	Unité Biodiversité

1. Présentation de l'entreprise

1.1. DEAL Réunion

La Direction de l'Environnement, de l'Aménagement et du Logement (DEAL) est le service déconcentré de l'État. Elle met en œuvre, à l'échelle régionale, les politiques publiques relevant du Ministère de la Transition Écologique et de la Cohésion des Territoires, ainsi que du Ministère de la Transition Énergétique. La Réunion est un territoire insulaire soumis à de fortes pressions environnementales et à des enjeux d'aménagement complexes. Dans ce contexte, la DEAL occupe une place centrale. Elle se situe au croisement des questions d'environnement, de biodiversité, d'eau, d'urbanisme et de développement territorial.

La DEAL Réunion assure l'application des réglementations environnementales. Elle instruit les projets d'aménagement et gère les risques naturels. Elle suit la ressource en eau et met en œuvre les politiques de protection des milieux naturels. Elle travaille en étroite collaboration avec les collectivités et plusieurs établissements publics, tels que l'OFB (Office Français de la Biodiversité), le PNRun (Parc National de La Réunion) ou l'Ifremer. Elle coopère aussi avec de nombreuses associations locales, comme la SEOR (Société d'Études Ornithologiques de La Réunion) ou la SREPEN (Société réunionnaise pour l'étude et la protection de la nature), qui contribuent activement au suivi et à la préservation de la biodiversité insulaire.

Au sein de cette structure, le Service Eau et Biodiversité (SEB) porte les missions liées à la préservation des milieux aquatiques et terrestres. Il développe la connaissance des espèces, veille à leur protection et régule les activités susceptibles d'impacter la biodiversité. Le SEB se trouve ainsi au cœur des enjeux écologiques de l'île.

Mon stage s'est déroulé au sein de l'Unité Biodiversité (UBIO). Cette unité est responsable du suivi des espèces et des habitats naturels. Elle gère et valorise les données naturalistes et instruit les dossiers réglementaires liés à la biodiversité. Elle anime également le Système d'Information sur la Nature et les Paysages (SINP-974 (Système d'Information de l'iNventaire du Patrimoine naturel de la Réunion)) régional et assure la gestion de la plateforme Borbonica. L'unité intervient aussi sur des thématiques transversales telles que les espèces exotiques envahissantes (EEE), la séquence ERC (Éviter – Réduire – Compenser) ou la diffusion des connaissances naturalistes. L'organisation interne de l'UBIO est présentée en annexe (Fig. 7.1).

L'organigramme interne montre une équipe composée de profils scientifiques, techniques et administratifs. Ces personnels travaillent de manière complémentaire pour répondre aux enjeux liés à la biodiversité du territoire. Mon stage s'inscrit dans cette dynamique, au sein du pôle

dédié aux données naturalistes. Il contribue à la structuration et à la modernisation des outils numériques utilisés par la DEAL.

Cette présentation de la DEAL et de son organisation permet de situer le contexte global de mon stage. Le chapitre suivant propose un **diagnostic RSE (Responsabilité Sociétale des Entreprises)** de la structure. Il vise à évaluer ses pratiques au regard des enjeux sociaux, environnementaux et organisationnels.

1.2. Diagnostic RSE

En raison des responsabilités qui lui sont confiées, la DEAL Réunion doit intégrer des considérations sociales, environnementales et organisationnelles dans l'ensemble de ses pratiques. Le diagnostic qui suit s'appuie sur la norme ISO 26000 et sur les principes de la RNE (Responsabilité Numérique des Entreprises), afin d'évaluer la manière dont la structure prend en compte ces enjeux.

1.2.1. Risques et impacts des activités

Impacts environnementaux
La DEAL n'engendre pas d'impacts industriels directs. Cependant, plusieurs de ses activités présentent des risques environnementaux indirects :

- un usage intensif d'outils numériques (bases de données naturalistes, SIG, plateformes SINP-974, Géonature, Borbonica (Plateforme régionale de diffusion des données naturalistes du SINP)), impliquant une consommation énergétique importante et une dépendance aux infrastructures serveurs ;
- le stockage et le traitement de volumes croissants de données, contribuant à l'empreinte carbone numérique et à la production de déchets électroniques liés au renouvellement du matériel ;
- des déplacements professionnels réguliers pour l'instruction de projets, le suivi de chantiers ou les missions de terrain, occasionnant des émissions de gaz à effet de serre.

Ces activités consomment de l'énergie, mobilisent des ressources non renouvelables et contribuent à l'empreinte environnementale globale du service.

Impacts sociétaux
Les missions de la DEAL comportent également plusieurs risques sociétaux :

- une influence directe sur les décisions d'aménagement (urbanisme, infrastructures, gestion des risques), pouvant affecter l'organisation du territoire et la qualité de vie des habitants.

- une responsabilité dans l'instruction des permis CITES et le suivi des espèces menacées, dont une mauvaise évaluation pourrait fragiliser le patrimoine naturel.
- la coordination des actions de lutte contre les espèces exotiques envahissantes (EEE), où un retard ou un défaut d'intervention peut entraîner une dégradation durable des écosystèmes.
- une forte dépendance aux partenaires (collectivités, OFB, Ifremer, PNRun, associations), rendant certaines actions vulnérables à des manques de coordination ou de moyens.

Ces risques peuvent affecter la cohérence des politiques publiques, la préservation des milieux naturels et l'équité territoriale.

Impacts sociaux

Le fonctionnement interne de la DEAL comporte également des risques sociaux. Ses équipes travaillent dans un environnement pluridisciplinaire soumis à des contraintes organisationnelles et réglementaires :

- une charge de travail variable selon les périodes, pouvant entraîner des tensions dans la répartition des tâches et la disponibilité des agents.
- des missions de terrain présentant des risques pour la santé et la sécurité (déplacements fréquents, conditions difficiles, chantiers, zones accidentées).
- un besoin constant de mise à jour des compétences (réglementations, outils numériques, SIG), créant un risque d'obsolescence professionnelle pour les agents non formés.
- un renouvellement régulier des stagiaires ou contractuels, pouvant fragiliser la continuité des projets ou l'organisation des équipes.

Ces éléments peuvent affecter la qualité de vie au travail, la sécurité en mission et la stabilité organisationnelle.

1.2.2. Stratégie globale en matière de RSE

Bien qu'elle agisse dans un cadre réglementaire strict, la DEAL cherche à renforcer la cohérence et la qualité de ses pratiques internes. Elle s'inscrit dans une dynamique de *pré-conformité active*, allant au-delà des obligations minimales en matière d'organisation, de gestion de l'information et de modernisation numérique. Cette démarche vise à améliorer la fiabilité des données environnementales, à harmoniser les outils utilisés par les services et à garantir une application rigoureuse des réglementations.

Sur le plan sociétal, la DEAL renforce ses méthodes de travail pour limiter les risques identifiés. Elle veille à la transparence dans l'instruction des projets d'aménagement, à sécuriser

les procédures liées aux permis CITES et à structurer davantage la coordination des actions de lutte contre les (EEE).

Cette dynamique repose également sur une coopération accrue avec les collectivités territoriales, les établissements publics et les associations naturalistes, afin de réduire les vulnérabilités liées à la gouvernance et à la circulation de l'information. Ces partenariats contribuent à une meilleure gestion des risques sociaux, environnementaux et sociétaux auxquels le territoire est confronté.

1.2.3. Conclusion personnelle

Ce diagnostic montre que la DEAL Réunion intègre progressivement les enjeux de la RSE et de la RNE au cœur de ses pratiques, malgré les contraintes propres à une administration publique. Cette approche illustre la volonté de la structure de moderniser son fonctionnement et de réduire ses vulnérabilités, tout en répondant aux attentes réglementaires et territoriales. Cette réflexion renforce l'idée que le numérique et l'organisation interne jouent un rôle essentiel dans la transition écologique et l'amélioration des politiques publiques.

2. Organisation du stage

2.1. Contexte du stage

Le SINP-974 est un dispositif collectif de mise en partage des données d'observations d'espèces sauvages sur l'île. Il a été mis en service à La Réunion en 2018, au travers de la plateforme Borbonica gérée conjointement par PNRun et la DEAL.

Borbonica s'appuie sur un portail web, qui permet d'accéder aux différentes interfaces utilisateurs du SINP :

- Borbonica obs : consultation cartographique des données du SINP. Il s'agit d'une interface web basée sur le plugin Lizmap et reposant sur une base de données PostgreSQL. Cette plateforme offre un accès aux données selon différents niveaux et profils (grand public, experts, validateurs, etc.) ;
- Borbonica atlas : consultation des fiches espèces disponibles dans le SINP, sous forme de synthèse des principales informations (observations, documents, photos, etc.). Il s'agit d'un outil issu de la solution open source GéoNature ;
- Borbonica stats : module statistique qui permet de consulter des tableaux de bord chiffrés sur le contenu de Borbonica (données disponibles, principaux usages, etc.). Ce module s'appuie sur un projet Lizmap, exploitant des données de la base PostgreSQL.

Afin de pérenniser le SINP 974, un projet a été lancé en 2023, avec pour principaux objectifs la modernisation et la simplification du fonctionnement du système. Plusieurs nouvelles fonctionnalités vont être ajoutées, et certaines briques logicielles seront remplacées par de nouvelles solutions plus modernes et plus simples à utiliser et à maintenir. C'est le cas notamment de la solution Géonature qui est en cours de déploiement pour remplacer le périmètre de Borbonica obs.

2.2. Objectifs du stage

À la suite de la modernisation engagée pour le SINP-974 et du déploiement progressif de GeoNature au sein de la DEAL, le stage avait pour finalité d'apporter un appui opérationnel au renforcement de la chaîne d'acquisition et de gestion des données naturalistes. L'objectif principal était de faciliter l'intégration de données externes dans GeoNature et, plus largement, d'enrichir les informations accessibles via la plateforme régionale Borbonica.

Dans ce cadre, plusieurs objectifs spécifiques ont été définis :

- développer un module externe GeoNature interfacé avec le système d'information Quadrige de l'Ifremer, afin d'automatiser la récupération, le filtrage et la préparation des données issues du milieu marin ;
- poser les fondations techniques nécessaires à une future intégration des données produites par l'application Pl@ntNet, en réalisant un client Python et un premier outil de structuration des résultats ;
- documenter les procédures d'appel aux API et proposer une organisation des développements garantissant leur maintenabilité et leur évolution dans le temps.

En complément du module Quadrige, une seconde orientation du stage concernait la préparation d'un futur connecteur GeoNature dédié aux données issues de Pl@ntNet. Pl@ntNet produit chaque jour un volume important d'observations naturalistes géolocalisées, particulièrement pertinent pour le suivi des EEE végétales et des dynamiques de végétation. Afin d'explorer la faisabilité de cette intégration, un travail préliminaire a consisté à :

- étudier la documentation de l'API Pl@ntNet v3,
- concevoir un client Python robuste, capable d'interroger l'API selon plusieurs critères (taxon, polygone GeoJSON, plage temporelle),
- structurer les données extraites au format *Darwin Core* afin d'assurer une compatibilité immédiate avec GeoNature et le SINP-974.

Ce travail prépare le développement d'un parser complet destiné au module `api2gn` de GeoNature.

Ces objectifs s'inscrivent pleinement dans la stratégie portée conjointement par la DEAL Réunion et le PNRun, visant à fiabiliser et à harmoniser les données du SINP-974 tout en renforçant la représentation de thématiques encore peu renseignées, comme le milieu marin. Ils ont guidé l'ensemble des actions menées durant le stage et constituent le cadre des travaux détaillés dans les sections suivantes.

2.3. Méthodologie de travail

La méthodologie adoptée au cours du stage s'est construite de manière progressive et pragmatique. Elle a été définie en tenant compte des contraintes techniques du projet, du cadre institutionnel de la DEAL et de l'organisation particulière de l'équipe. Le projet étant mené uniquement à deux personnes — mon maître de stage, Rémi Bouilly, et moi-même — il n'était pas pertinent de mettre en place une méthodologie formelle lourde. Le cadre de travail retenu s'inspire néanmoins des principes des méthodes agiles, tout en restant volontairement souple.

Le développement du module Quadrige et des outils associés a ainsi été conduit selon une démarche itérative et incrémentale. Le travail était organisé en cycles courts, assimilables à des sprints d'une durée moyenne d'environ une semaine. Chaque sprint correspondait à un ensemble cohérent de fonctionnalités, définies en amont, puis développées, testées et ajustées avant d'engager le cycle suivant. Cette organisation permettait d'obtenir régulièrement des résultats fonctionnels, même partiels, tout en identifiant rapidement les difficultés techniques ou fonctionnelles.

Dans ce contexte de travail en effectif réduit, il n'était ni nécessaire ni utile de mettre en place des rituels agiles classiques tels que les daily meetings ou les revues de sprint formelles. En revanche, des points réguliers étaient organisés une à deux fois par semaine avec mon maître de stage. Ces échanges permettaient de faire un état d'avancement, de valider les choix techniques effectués et d'ajuster les priorités à court terme en fonction des besoins métier. Ils ont joué un rôle central dans le bon déroulement du projet, en assurant une compréhension partagée des objectifs tout en me laissant une large autonomie dans la mise en œuvre.

Le cadrage initial du stage s'est appuyé sur deux cahiers des charges distincts, fournis par mon maître de stage. Le premier concernait le développement du module externe GeoNature interfacé avec Quadrige, tandis que le second portait sur la conception d'un parser dédié aux données issues de Pl@ntNet et à leur structuration au format Darwin Core. Ces documents définissaient le périmètre fonctionnel attendu, les principales contraintes techniques ainsi que les livrables envisagés. Ils ont servi de fil conducteur tout au long du stage, tout en restant suffisamment flexibles pour intégrer des ajustements liés aux contraintes rencontrées ou à l'évolution du projet.

Afin de suivre l'avancement global du projet et de visualiser les grandes phases du stage, un diagramme de Gantt a été utilisé. Il a permis de planifier les principales étapes, depuis la prise en main de l'environnement GeoNature jusqu'aux phases de tests, de documentation et de préparation des livrables. Cet outil offrait une vision synthétique du projet et facilitait l'anticipation des périodes plus sensibles, notamment lors des phases de migration des serveurs ou d'attente d'accès aux environnements distants.

À un niveau plus opérationnel, le travail quotidien était organisé sous forme de tâches unitaires, réparties selon un statut simple : à faire, en cours ou terminées. Cette organisation, inspirée des tableaux de type Kanban, permettait de visualiser rapidement l'état d'avancement, de prioriser les actions et de conserver une trace claire du travail réalisé. Elle s'est révélée particulièrement adaptée à un projet individuel encadré, en apportant une structure légère sans alourdir inutilement le processus.

La gestion du code source reposait intégralement sur l'utilisation de Git. L'ensemble des développements a été versionné dès les premières phases du projet, aussi bien pour le module Quadrige que pour les outils liés à Pl@ntNet. Les évolutions étaient enregistrées de manière régulière, avec des messages de commit explicites. Cette pratique a permis de sécuriser les

développements, de tester différentes approches sans risque de perte de code et de faciliter le transfert vers les serveurs de la DEAL une fois l'accès ouvert. Dans un contexte où l'accès aux environnements distants était limité et parfois instable, Git a également servi de principal moyen de synchronisation entre l'environnement local et l'environnement distant.

Le développement a d'abord été mené intégralement en local, en l'absence d'un accès immédiat aux serveurs du SINP-974. Cette situation a nécessité la mise en place d'un environnement de travail stable, capable de reproduire au mieux les conditions de production. La séparation stricte entre le backend Python/Flask et le frontend Angular, conforme à l'architecture modulaire de GeoNature, a facilité cette organisation et permis de développer chaque composante de manière indépendante.

Les contraintes liées au proxy ministériel et aux versions spécifiques des outils techniques ont également nécessité une attention particulière à la configuration de l'environnement. L'utilisation de pyenv a permis de stabiliser la version de Python utilisée, condition indispensable pour garantir la compatibilité avec GeoNature et ses dépendances.

Enfin, l'ensemble du travail s'est appuyé sur une démarche de validation progressive. Les appels à l'API Quadrige ont d'abord été testés à l'aide de scripts Python indépendants, avant leur intégration dans le module GeoNature. De la même manière, les premières interfaces Angular ont été testées en local afin de vérifier leur cohérence fonctionnelle et leur ergonomie avant toute intégration plus large.

Cette méthodologie, mêlant principes agiles, rigueur technique et adaptation constante au contexte, a permis d'assurer la robustesse des développements malgré un environnement parfois contraint. Elle a également facilité l'anticipation des étapes futures d'intégration et de reprise du projet par la DEAL ou ses partenaires.

3. Introduction à GeoNature

3.1. Principe général de GeoNature

GeoNature est une application web dédiée à la gestion, la centralisation et la valorisation des données naturalistes. Développé initialement en 2010 puis entièrement refondu en 2017, le projet est aujourd'hui maintenu par le PNRun. L'application repose sur une architecture moderne combinant un backend Python/Flask et un frontend Angular, ce qui lui permet d'assurer à la fois la saisie, la consultation, la validation et la restitution des données.

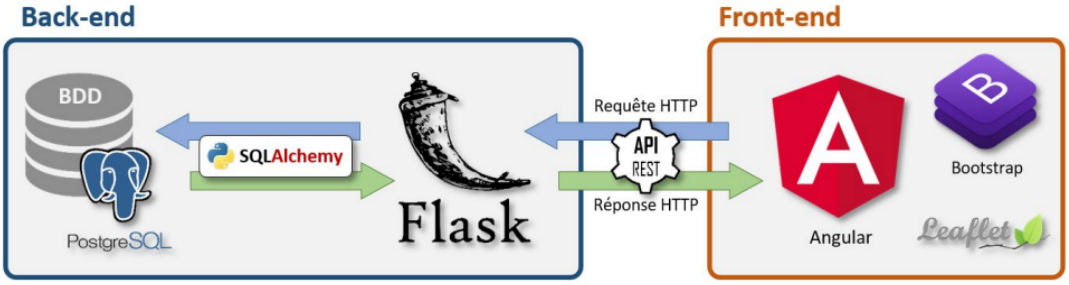

Figure 3.1: Architecture générale du backend et du frontend de GeoNature

Le fonctionnement de GeoNature est modulaire : un noyau applicatif fournit les briques communes (API, référentiels, schéma de synthèse), et différents modules viennent étendre les fonctionnalités selon les besoins (Occtax, Occhab, Validation, Import, Export, etc.). La documentation officielle détaille l'ensemble de cette architecture, (documentation officielle) ainsi que les sources du projet, disponibles sur GitHub.

3.2. Architecture technique

GeoNature combine une partie serveur (backend) et une partie cliente (frontend) qui dialoguent via une API REST.

Le backend est développé en Python à l'aide du framework Flask. Il assure l'ensemble des traitements métiers, l'accès aux données et la gestion des opérations spatiales via PostgreSQL et PostGIS. La gestion des utilisateurs et de leurs permissions repose sur UsersHub, qui fournit l'authentification et le contrôle des accès. L'API exposée par le backend constitue le point central de communication avec l'interface web.

Le frontend, développé en Angular, constitue l'interface visible par l'utilisateur. Il interroge exclusivement l'API du backend pour afficher les formulaires, les cartes, les graphiques et les données attributaires. Des bibliothèques comme Leaflet ou Bootstrap renforcent les fonctionna-

lités cartographiques et l'ergonomie générale. Cette séparation nette entre backend et frontend garantit la stabilité et la modularité du système, tout en facilitant les évolutions futures.

3.3. Organisation interne et modules

L'application est structurée autour de modules fonctionnels s'appuyant tous sur les mêmes référentiels : taxonomie (TaxHub), nomenclatures, utilisateurs (UsersHub) et schéma de synthèse. Les modules principaux — Occtax pour les observations, Occhab pour les habitats, ou encore la Validation — s'intègrent directement au cœur applicatif.

Chaque module possède son propre schéma de base de données, son API et ses composants Angular. Cette organisation modulaire permet de faire évoluer GeoNature, d'ajouter de nouveaux protocoles d'acquisition ou de développer des extensions externes. La documentation décrit précisément les bonnes pratiques et l'architecture à respecter pour développer un module GeoNature externe.

4. Developpement des modules d'imports de données

4.1. Contexte du projet

Le SINP-974, déployé à La Réunion à travers la plateforme Borbonica, constitue le dispositif régional de référence pour la centralisation, la gestion et la diffusion des données naturalistes. Depuis sa mise en service, il a permis de structurer et de valoriser un volume important d'observations, issues de sources multiples et couvrant un large éventail de taxons.

La dynamique de peuplement de la base reflète toutefois les modalités historiques et opérationnelles de production des données naturalistes sur le territoire. Les observations actuellement intégrées concernent majoritairement les milieux terrestres et dulçaquicoles, qui bénéficient de réseaux d'acteurs structurés, de protocoles d'acquisition largement diffusés et d'outils de saisie directement compatibles avec le SINP.

À l'inverse, les données issues du milieu marin demeurent plus faiblement représentées. Les indicateurs de contenu disponibles en 2024 montrent que les observations relatives aux taxons marins ne constituent qu'une part marginale des données référencées. Cette situation ne traduit pas une absence de connaissances ou de suivis en milieu marin, mais résulte principalement de la complexité des chaînes d'acquisition, de structuration et de diffusion propres aux données marines, historiquement gérées au sein de systèmes d'information spécialisés.

Dans ce contexte, l'enjeu principal n'était pas de refondre les outils existants, mais de renforcer progressivement la représentation des données marines au sein du SINP-974, en s'appuyant sur des sources de données déjà structurées, pérennes et reconnues institutionnellement. Parmi celles-ci, le système d'information Quadrige, maintenu par l'Ifremer, occupe une place centrale. Quadrige regroupe un volume important de données environnementales et biologiques collectées en milieu marin, issues de programmes de surveillance, d'observation et de recherche conduits sur le long terme.

L'ouverture récente d'une API (Interface de Programmation de l'Application) GraphQL par l'Ifremer constitue à ce titre une évolution majeure. Elle permet une interrogation directe, fine et structurée des données de Quadrige, en ciblant précisément les programmes, les zones géographiques, les périodes temporelles et les types d'observations d'intérêt. Cette évolution ouvre la voie à une exploitation plus systématique de ces données par des acteurs institutionnels tels que la DEAL, et à leur mobilisation progressive dans le cadre du SINP-974.

C'est dans ce cadre que s'inscrit le projet développé durant le stage. L'objectif n'était pas de réaliser une intégration directe et entièrement automatisée des données Quadrige dans la base

GeoNature, mais de concevoir un module externe permettant d'explorer, d'extraire et de préparer ces données de manière structurée, traçable et reproductible. Le module développé interroge l'API Quadrige afin d'identifier les programmes pertinents pour le territoire réunionnais, puis permet d'en extraire les observations associées selon des critères définis par l'utilisateur.

Les résultats de ces extractions ne prennent pas la forme de fichiers immédiatement importables dans GeoNature. Pour chaque programme sélectionné, le module génère une archive compressée contenant plusieurs éléments complémentaires : un fichier CSV brut correspondant aux données extraites, un fichier JSON décrivant précisément les filtres appliqués lors de l'appel à l'API, ainsi qu'un fichier README documentant le déroulement de l'export et les éventuelles anomalies rencontrées. Le module conserve également un accès aux trois derniers exports réalisés afin d'en faciliter la consultation et la réutilisation.

En complément de ces archives, le module expose également des liens directs vers les fichiers CSV générés lors de la phase d'extraction des programmes. Deux niveaux de fichiers sont distingués : – un CSV brut issu directement de la réponse de l'API Quadrige, contenant l'ensemble des instances des programmes dont au moins une occurrence est localisée sur le territoire ciblé ; – un CSV filtré géographiquement, obtenu par un traitement a posteriori à l'aide de la bibliothèque pandas, permettant d'exclure les instances dont le code de monitoring location ne correspond pas au territoire d'intérêt (par exemple, codes ne commençant pas par le préfixe « 126- » pour La Réunion).

Ce filtrage complémentaire est nécessaire car, lors de l'extraction des programmes, la requête adressée à l'API Quadrige sélectionne l'ensemble des programmes possédant au moins une instance sur le territoire demandé, mais retourne également les autres instances associées à ces programmes, y compris celles situées hors du périmètre géographique ciblé.

Cette approche intermédiaire répond à plusieurs objectifs. Elle garantit d'une part la traçabilité complète des extractions réalisées depuis Quadrige, en conservant une description explicite des paramètres et traitements appliqués. Elle offre d'autre part une souplesse d'usage, en laissant aux administrateurs et gestionnaires de données la possibilité de contrôler, d'analyser et, le cas échéant, d'adapter les fichiers produits avant toute intégration dans GeoNature ou Borbonica. Le module s'inscrit ainsi comme une brique préparatoire, destinée à sécuriser et à faciliter l'intégration future des données marines dans le SINP-974.

Ce développement s'inscrit pleinement dans la dynamique d'amélioration continue portée par la DEAL Réunion et ses partenaires. Il vise à renforcer progressivement la place des données issues du milieu marin au sein du système régional, tout en respectant les contraintes techniques, méthodologiques et organisationnelles propres aux outils existants.

4.2. Périmètre fonctionnel

Le module d'import devait couvrir l'ensemble de la chaîne d'acquisition : de la découverte des programmes Quadrige jusqu'à la production d'un fichier structuré pour GeoNature. La première

étape consistait à interroger l'API en mode authentifié afin d'obtenir la liste des programmes disponibles pour un utilisateur donné. Un filtrage automatique sur un périmètre géographique — principalement La Réunion, mais extensible à d'autres territoires comme les Îles Éparses — permettait d'isoler les programmes pertinents. Une interface dédiée intégrée à GeoNature offrait ensuite la possibilité de rechercher des programmes, d'affiner l'affichage par mots-clés et de sélectionner ceux à importer.

Une fois les programmes choisis, l'utilisateur pouvait définir les filtres à appliquer aux données (la période d'intérêt, les champs souhaités, ou encore la reprise des stations déjà extraites). Le module interrogeait alors `https://quadrige-core.ifremer.fr/api/extraction/doc?doc=standard&lang=fr&name=result&type=standard`

pour récupérer les observations correspondantes Tutoriels vidéo. Seuls les champs utiles au modèle SINP-974 étaient extraits : identifiants des programmes et stations, localisation géographique, taxon observé, date, ainsi que les métadonnées essentielles (auteur, organisme, méthode d'acquisition, etc.). Une transformation était appliquée pour obtenir une structure compatible avec les mécanismes d'import de GeoNature.

Enfin, le module produisait un fichier CSV intermédiaire, destiné à être importé via l'infrastructure existante de GeoNature. Chaque opération d'import était consignée dans un historique affiché dans un second onglet, permettant de suivre les actions réalisées, leur date, leur statut et les éventuelles erreurs rencontrées. L'ensemble du module était réservé aux administrateurs, conformément aux pratiques habituelles de contrôle des imports dans GeoNature.

4.3. Conception technique du module d'import Quadrige

Le module Quadrige développé durant le stage a été conçu comme un module externe à GeoNature, respectant l'architecture recommandée par le projet tout en prenant en compte les contraintes spécifiques de l'API GraphQL mise à disposition par l'Ifremer. L'objectif principal était de proposer une chaîne d'extraction robuste, traçable et exploitable, sans perturber le cœur applicatif de GeoNature ni les processus existants de gestion des données naturalistes.

4.3.1. Architecture générale et découplage frontend/backend

Le module repose sur une architecture client–serveur classique, intégrée à GeoNature via un *blueprint* Flask côté backend et un module Angular dédié côté frontend. Le backend est responsable de l'ensemble des interactions avec l'API Quadrige, incluant l'authentification, la construction des requêtes GraphQL, le suivi des extractions, la gestion des fichiers produits et l'exposition de routes REST sécurisées.

Le frontend se limite volontairement au pilotage des opérations : sélection des paramètres, lancement des extractions et visualisation des résultats. Il n'accède jamais directement à l'API Quadrige ni aux paramètres sensibles, ce qui permet de renforcer la sécurité globale du dispositif.

Ce découplage permet :

- de centraliser les accès à l'API Quadrige et les jetons d'authentification ;
- de limiter la surface d'exposition des données sensibles ;
- de faciliter la maintenance et les évolutions indépendantes du backend et du frontend.

4.3.2. Gestion de la configuration et des paramètres sensibles

Le module s'appuie sur un fichier de configuration dédié, chargé côté backend via le mécanisme standard de GeoNature (`geonature/config`). Ce fichier centralise l'ensemble des paramètres techniques nécessaires au fonctionnement du module, notamment :

- l'URL de l'API GraphQL Quadrige ;
- le jeton d'authentification requis pour les appels à l'API ;
- des paramètres métiers optionnels, tels que les localisations suggérées ou les champs extractibles.

Une route backend permet d'exposer cette configuration au frontend de manière contrôlée, afin de rendre l'interface dynamique et adaptable. Certaines données métiers restent actuellement définies côté frontend, mais l'architecture mise en place permettrait de les centraliser entièrement côté backend dans une version ultérieure, sans remise en cause du fonctionnement global.

4.3.3. Extraction des programmes Quadrige

La première étape du processus consiste à identifier les programmes Quadrige pertinents pour un territoire donné. Cette extraction repose sur une requête GraphQL spécifique fournie par l'API Quadrige, filtrée à partir d'un préfixe de *monitoring location* correspondant au périmètre géographique ciblé.

L'API Quadrige retournant l'ensemble des instances associées à un programme dès lors qu'au moins une station correspond au critère de recherche, un traitement complémentaire est appliqué côté backend. Les fichiers CSV bruts générés par l'API sont filtrés a posteriori à l'aide de la bibliothèque `pandas`, afin de ne conserver que les programmes effectivement localisés sur le territoire demandé.

Cette étape permet de produire :

- un CSV brut, conservé à des fins de traçabilité et de vérification ;
- un CSV filtré, utilisé pour l'affichage et la sélection des programmes dans le frontend.

Les métadonnées associées à chaque extraction (localisation, horodatage, filtre utilisé) sont sauvegardées afin de permettre la reprise des extractions et la consultation des résultats antérieurs.

4.3.4. Extraction des données et gestion asynchrone des traitements

L'extraction des données repose sur un mécanisme asynchrone propre à l'API Quadrige. Pour chaque programme sélectionné, une requête GraphQL est soumise afin de lancer un job d'extraction côté serveur Ifremer. Le module implémente un mécanisme de *polling* permettant de suivre l'évolution de l'état de chaque extraction (PENDING, RUNNING, SUCCESS, WARNING, ERROR).

Les extractions sont gérées de manière indépendante, ce qui permet :

- de paralléliser les traitements sur plusieurs programmes ;
- de gérer finement les échecs ou avertissements retournés par l'API ;
- d'éviter un blocage global en cas d'échec partiel.

À l'issue des traitements, les fichiers ZIP générés par l'API Quadrige sont téléchargés et stockés temporairement sur le serveur GeoNature. Chaque fichier est renommé selon une convention explicite intégrant le programme, la localisation et l'horodatage, garantissant ainsi sa traçabilité.

4.3.5. Gestion des états, des erreurs et de la traçabilité

Un soin particulier a été apporté à la gestion des états et des erreurs tout au long du processus d'extraction. Les situations suivantes sont explicitement prises en compte :

- erreurs de communication avec l'API GraphQL ;
- délais excessifs lors des extractions longues ;
- incohérences ou réponses partielles retournées par l'API ;
- extractions aboutissant à des avertissements ou à l'absence de données.

Chaque extraction génère un retour structuré indiquant son statut, les avertissements éventuels et les liens vers les fichiers produits. Ces informations sont transmises au frontend afin de permettre à l'utilisateur d'identifier rapidement les extractions exploitables et celles nécessitant une vérification.

4.3.6. Interface utilisateur et pilotage des extractions

Afin de rendre le fonctionnement du module plus concret, les figures suivantes présentent les principales interfaces utilisateur du module Quadrige, depuis l'exploration des programmes jusqu'à l'accès aux résultats d'extraction.

Le frontend Angular du module a été conçu pour accompagner l'utilisateur à travers les différentes étapes du processus : définition des filtres, extraction des programmes, sélection des

données et lancement des extractions. Des contrôles de cohérence sont intégrés afin de prévenir les erreurs de paramétrage, notamment sur les champs obligatoires ou les périodes temporelles.

Compte tenu du volume potentiellement élevé de programmes retournés par l'extraction — pouvant dépasser plusieurs dizaines de résultats selon le périmètre géographique — l'interface intègre des mécanismes de recherche par mots-clés afin de faciliter leur exploration et leur sélection. Il est ainsi possible de filtrer dynamiquement la liste des programmes extraits en saisissant un terme dans la barre de recherche, permettant de cibler rapidement un programme par son nom ou ses métadonnées associées.

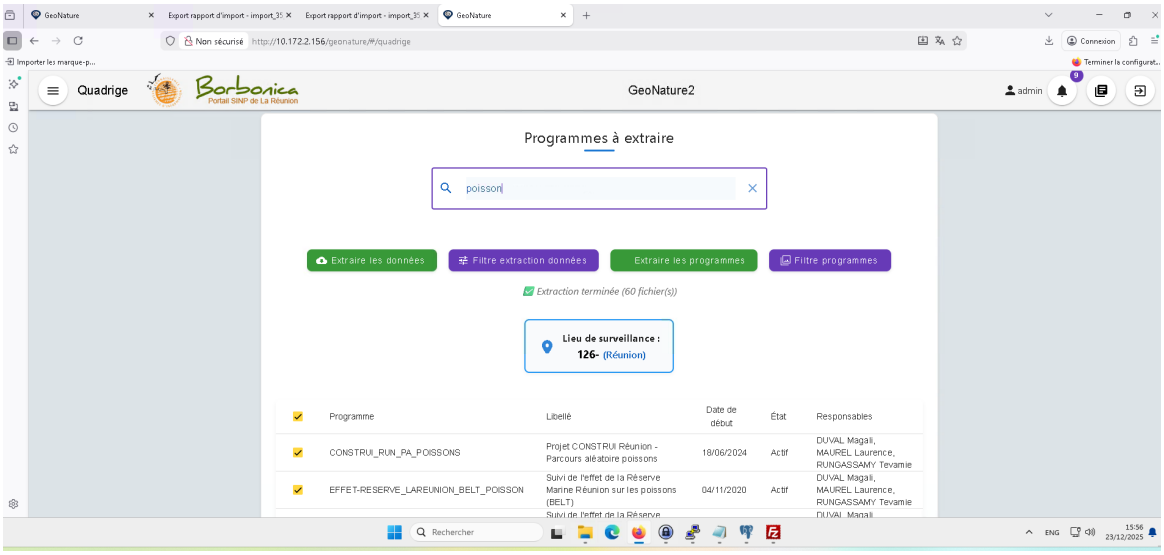

Figure 4.1: Liste des programmes Quadrige extraits pour un périmètre géographique donné, tri par mots clés avec la barre de recherche

Cette logique de recherche est également appliquée lors des phases de paramétrage. Dans les interfaces de filtrage, l'utilisateur peut rechercher les champs disponibles à l'extraction en saisissant directement du texte dans le champ dédié, ce qui facilite la sélection lorsque la liste des champs est étendue. De la même manière, les localisations suggérées lors de la définition des filtres peuvent être recherchées par saisie textuelle, améliorant l'ergonomie et limitant les erreurs de sélection.

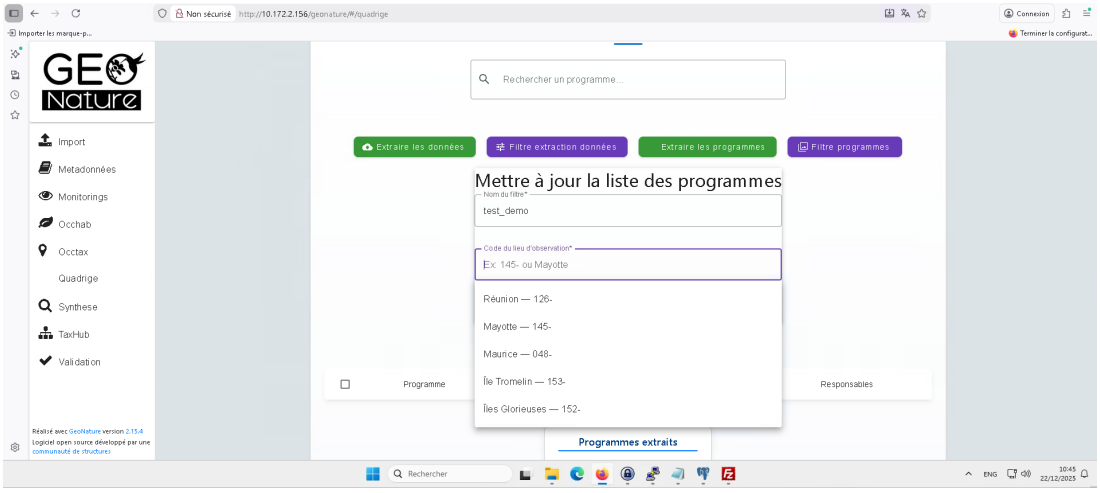

Figure 4.2: Interface de définition des filtres d'extraction des programmes Quadrige

Les résultats des extractions sont présentés sous forme de liens téléchargeables, accompagnés d'indicateurs visuels signalant les éventuels avertissements. Cette approche permet à l'administrateur de conserver une maîtrise complète sur les données produites avant toute intégration dans GeoNature ou dans des plateformes partenaires telles que Borbonica.

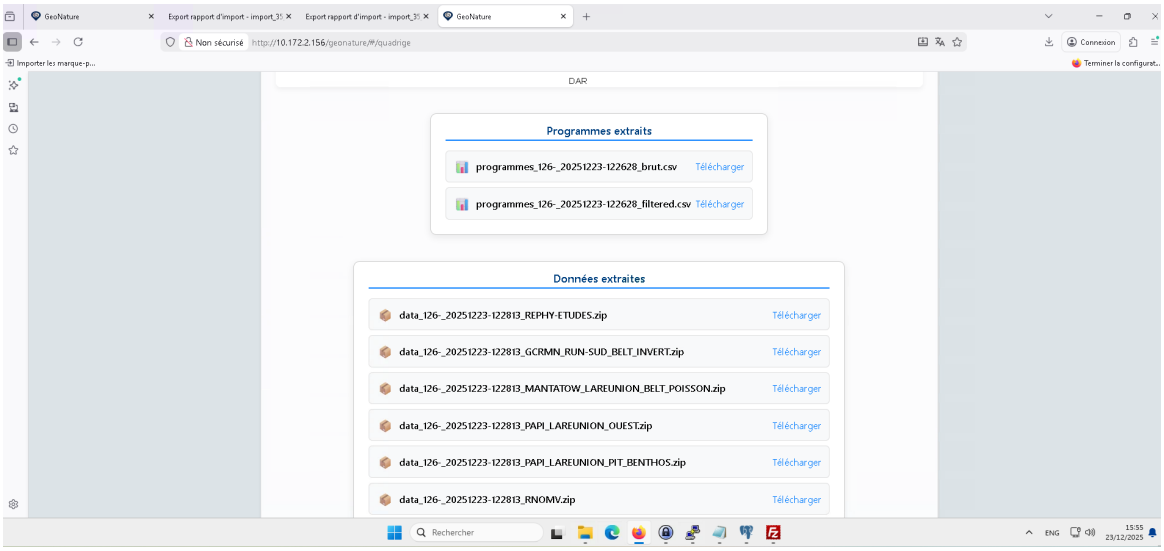

Figure 4.3: Résultats des extractions Quadrige et accès aux fichiers générés

Une interface spécifique permet également de définir les filtres d'extraction des données associées aux programmes sélectionnés (périodes temporelles, localisations, champs à extraire). À des fins de lisibilité, cette interface est présentée en annexe (voir Figure 7.2).

4.3.7. Synthèse

L'architecture retenue pour le module Quadrige répond aux contraintes spécifiques de l'API Ifremer tout en respectant les principes de modularité, de sécurité et de traçabilité propres à

GeoNature. Elle constitue une base technique solide pour des évolutions futures, notamment vers une automatisation partielle des imports, tout en conservant un contrôle humain sur les données produites.

4.4. Contraintes, dépendances et livrables du module Quadrige

La réussite du projet dépendait principalement de l'accessibilité de l'API Quadrige, de la stabilité de ses services, et de la disponibilité d'une documentation actualisée. L'intégration dans GeoNature nécessitait également de respecter la structure modulaire du noyau applicatif et les contraintes du modèle SINP-974.

Les livrables attendus comprenaient le code source du module, un fichier de configuration, une documentation à destination des administrateurs (installation, configuration, maintenance) ainsi qu'un guide d'utilisation orienté métier.

Plusieurs évolutions ont été envisagées : automatisation des imports périodiques, gestion des imports incrémentaux, intégration à la gestion des droits de GeoNature et, à plus long terme, publication du module dans le catalogue officiel des extensions GeoNature.

4.5. Extension du module api2GN pour l'intégration des données Pl@ntNet

4.5.1. Rôle du module api2GN dans l'écosystème GeoNature

Le module `api2GN` constitue une brique transversale au sein de l'écosystème GeoNature. Contrairement aux modules d'import classiques, fondés sur des fichiers intermédiaires produits manuellement, `api2GN` a été conçu pour permettre l'intégration directe et automatisée de données issues de sources externes, exposées via des services web (API REST, flux JSON, services WFS).

Son principe repose sur une séparation claire entre :

- le **cœur du moteur d'import**, chargé de l'orchestration des appels API, de la gestion des erreurs, de l'historisation et de l'insertion en base ;
- des **parseurs spécialisés**, responsables de la transformation des données sources vers le modèle de la table *Synthèse* de GeoNature.

Chaque parseur implémente ainsi une logique métier propre à une source de données donnée, tout en s'appuyant sur un socle commun fourni par `api2GN` : gestion des géométries, résolution des nomenclatures, validation du mapping, historisation des imports et intégration dans les mécanismes de sécurité de GeoNature.

Dans ce cadre, le module `api2GN` apparaît comme un outil particulièrement adapté à l'intégration de données issues de plateformes participatives ou ouvertes, telles que Pl@ntNet, dont les volumes, la fréquence de mise à jour et les modalités d'accès se distinguent fortement des bases institutionnelles comme Quadrige.

4.5.2. Objectifs et positionnement du parser Pl@ntNet

Le développement du parser Pl@ntNet s'inscrit dans une logique complémentaire à celle du module Quadrige. Alors que l'import Quadrige vise des données institutionnelles, fortement structurées et produites dans un cadre scientifique contrôlé, les données Pl@ntNet relèvent d'un contexte participatif, caractérisé par :

- une production massive et continue d'observations ;
- une grande hétérogénéité des contributeurs ;
- une qualité variable des identifications taxonomiques ;
- une forte dépendance à des mécanismes de validation automatisés.

L'objectif du parser Pl@ntNet n'était donc pas de proposer une intégration exhaustive et aveugle de l'ensemble des observations disponibles, mais de mettre en place une chaîne d'import :

- **automatisée**, afin de pouvoir être exécutée régulièrement ;
- **configurable**, pour adapter le périmètre d'import aux besoins territoriaux ;
- **sélective**, en appliquant des filtres taxonomiques, spatiaux et temporels ;
- **rigoureuse**, en imposant un contrôle strict de la compatibilité avec le référentiel TAXREF.

Le parser Pl@ntNet a ainsi été conçu comme un cas d'étude illustrant une stratégie d'intégration différente de celle retenue pour Quadrige : une automatisation poussée, compensée par des mécanismes renforcés de validation et de traçabilité.

4.5.3. Architecture technique et principes de conception

Sur le plan technique, le parser Pl@ntNet repose sur l'architecture standard des parseurs `api2GN`. Il hérite directement de la classe `JSONParser`, elle-même dérivée de la classe générique `Parser`. Cette hiérarchie permet de mutualiser les fonctionnalités communes à l'ensemble des sources JSON, tout en laissant au parser la responsabilité de la logique métier spécifique à Pl@ntNet.

Le flux général d'exécution peut être résumé ainsi :

1. initialisation du parser et chargement de la configuration ;
2. construction des requêtes vers l'API Pl@ntNet ;
3. récupération paginée des occurrences ;

4. transformation et normalisation des données ;

5. résolution taxonomique ;

6. insertion des observations valides dans la table *Synthèse*.

Le parser a été conçu selon un principe fondamental : **aucune logique métier n'est codée en dur**. Les paramètres d'appel à l'API, les filtres appliqués, le mapping des champs et les règles de validation sont intégralement pilotés par la configuration, ce qui permet d'adapter le comportement du parser sans modifier le code Python.

Cette approche favorise la maintenabilité du module et facilite son appropriation par d'autres structures souhaitant exploiter les données Pl@ntNet sur des territoires ou des périmètres différents.

4.5.4. Configuration dynamique et pilotage par fichier TOML

L'un des apports majeurs du travail réalisé réside dans la mise en place d'un système de configuration entièrement externalisé, basé sur un fichier `TOML` chargé via le mécanisme standard de configuration de GeoNature.

Ce fichier permet de définir de manière centralisée :

- les paramètres d'accès à l'API Pl@ntNet (URL, clé API) ;
- les filtres taxonomiques (liste d'espèces ciblées ou absence de filtrage) ;
- les bornes temporelles d'extraction ;
- l'emprise géographique, définie par un polygone GeoJSON ;
- les règles de mapping entre les champs Pl@ntNet et la table *Synthèse*.

Le parser implémente un mécanisme de *fallback* permettant de fonctionner même en l'absence de configuration complète. Dans ce cas, des valeurs par défaut sont appliquées, accompagnées de messages d'avertissement explicites dans les logs. Cette stratégie garantit la robustesse du module en environnement de test ou de développement, tout en incitant à une configuration explicite en production.

Ce choix de conception permet également d'envisager, à terme, une interface d'administration graphique dédiée à la gestion de ces paramètres, sans remise en cause de l'architecture existante.

4.5.5. Résolution taxonomique et contrôle de la qualité des données

La résolution taxonomique constitue l'enjeu central de l'intégration des données Pl@ntNet dans GeoNature. Afin de garantir la cohérence avec le référentiel national TAXREF et les exigences du SINP, le parser met en œuvre une chaîne de validation stricte.

Pour chaque observation, le nom scientifique fourni par l'API Pl@ntNet est d'abord normalisé afin de supprimer les mentions infra-spécifiques ou les annotations non standard. Le parser tente ensuite de résoudre le `cd_nom` selon un ordre hiérarchique précis :

1. recherche dans le référentiel TAXREF local de GeoNature ;
2. en cas d'échec, interrogation du service TAXREF-LD du MNHN ;
3. vérification de l'existence du `cd_nom` obtenu dans la base locale.

Un cache mémoire est utilisé afin d'éviter les résolutions répétées pour un même taxon, ce qui améliore significativement les performances lors des imports volumineux.

Le parser fonctionne par défaut en **mode strict**. Dans ce mode, toute observation pour laquelle aucun `cd_nom` valide ne peut être résolu est rejetée. Ces rejets sont explicitement journalisés, permettant :

- d'identifier les taxons problématiques ;
- d'évaluer la qualité globale des données importées ;
- d'envisager, le cas échéant, une mise à jour du référentiel TAXREF local.

Ce mécanisme constitue un compromis entre automatisation et exigence de qualité, indispensable dans le contexte des données participatives.

4.5.6. Traçabilité, sécurité et automatisation des imports

Le parser Pl@ntNet s'intègre pleinement aux mécanismes de traçabilité fournis par `api2GN`. Chaque exécution est historisée dans une table dédiée, permettant de conserver :

- la date du dernier import ;
- le nombre d'observations importées lors de la dernière exécution ;
- le volume total d'observations intégrées ;
- une fréquence d'exécution planifiée, le cas échéant.

Un mode *dry-run* est systématiquement disponible. Il permet de simuler un import sans insertion en base, ce qui constitue un outil essentiel pour :

- tester une nouvelle configuration ;
- évaluer l'impact d'un changement de périmètre ;
- analyser la qualité des données avant intégration effective.

Enfin, le module est compatible avec le système de tâches planifiées de GeoNature, permettant d'envisager des imports périodiques automatisés, tout en conservant un contrôle fin sur leur déclenchement et leur suivi.

4.5.7. Apports du travail réalisé

Le développement du parser Pl@ntNet a permis de démontrer la capacité de GeoNature à intégrer des données participatives de manière automatisée, tout en respectant les exigences du SINP en matière de qualité, de traçabilité et de cohérence taxonomique.

Au-delà du cas spécifique de Pl@ntNet, ce travail met en évidence :

- la pertinence de l'architecture `api2GN` pour l'intégration de sources hétérogènes ;
- l'intérêt d'une configuration externalisée pour favoriser la réutilisation et l'adaptabilité ;
- la complémentarité des stratégies d'import entre données institutionnelles et participatives.

Le parser Pl@ntNet constitue ainsi un prototype fonctionnel et robuste, susceptible d'être étendu à d'autres sources de données ouvertes, et de contribuer à l'enrichissement progressif de la base GeoNature, notamment pour les taxons terrestres peu couverts par les réseaux institutionnels traditionnels.

5. Travaux réalisés et compétences acquises

5.1. Travaux réalisés

Les travaux réalisés au cours du stage se sont concentrés principalement sur le développement d'un module externe pour GeoNature, capable d'interagir avec le système d'information Quadrige. La première étape a consisté à concevoir un client Python permettant de tester l'envoi de requêtes GraphQL auprès de l'API Quadrige, de récupérer les données et de gérer l'authentification via un jeton.

Une fois les fonctions d'interrogation terminées, je suis passé au développement d'un prototype local du frontend Angular. Il se compose de quatre composants principaux. Le premier, `program-list`, affiche la liste des programmes extraits en fonction d'une localisation (par exemple « 126- » comme préfixe pour La Réunion). C'est depuis ce composant principal que s'effectuent les échanges backend/frontend.

Le composant `program-filter` permet de définir un filtre incluant une localisation et un nom de filtre. Les critères sont sauvegardés dans le fichier `memory/last_filter.json`. Ce filtre sert à extraire les programmes selon des règles bien définies. La requête GraphQL utilisée est `executeProgramExtraction`, qui comporte de nombreux paramètres optionnels, mais dont seuls ceux relatifs à la localisation nous intéressent. L'objectif est d'extraire tout programme comportant au moins une instance dans la zone spécifiée.

Un bouton « Extraire les programmes » déclenche la fonction d'extraction, qui envoie la requête GraphQL à Quadrige et récupère les programmes correspondant au filtre. Un message d'état informe l'utilisateur de l'avancement (« extraction en cours », « programmes extraits ») et, en cas d'erreur, un message approprié s'affiche.

Lorsque l'extraction est terminée, un fichier CSV est produit et stocké dans `memory/csv_brut_suffixe`, associé au filtre utilisé. Il contient toutes les instances des programmes ayant au moins une occurrence dans la zone filtrée.

Le fichier CSV généré à l'issue de cette première extraction contient l'ensemble des instances associées aux programmes sélectionnés, y compris celles situées en dehors du périmètre géographique d'intérêt. Cette situation résulte du fonctionnement même de l'API Quadrige, qui raisonne à l'échelle des programmes et non des stations individuelles.

Afin de garantir la cohérence géographique des données exploitées, un filtrage a posteriori a été mis en œuvre à l'aide de la bibliothèque pandas. Ce traitement consiste à ne conserver que les instances dont le code de monitoring location correspond au territoire ciblé, identifié par un

préfixe spécifique (par exemple « 126- » pour La Réunion).

Ce filtrage permet de produire un second fichier CSV, strictement limité aux stations pertinentes, utilisé à la fois pour l'affichage dans l'interface utilisateur et comme base pour les extractions de données ultérieures.

Par la suite, je n'ai besoin que du code du programme pour extraire les données recherchées.

Une fois les programmes extraits, ils sont affichés dans `program-list` avec leur nom, leurs dates de début et de fin et le(s) responsable(s). L'utilisateur peut sélectionner les programmes via une barre de recherche, des cases cochables et une option « tout cocher ».

Un composant `data-filter` permet ensuite de filtrer les données des programmes sélectionnés selon plusieurs critères, notamment les dates ou la localisation (automatiquement reprise depuis le dernier filtre créé).

—

En parallèle, j'ai conçu l'architecture du backend, fondée sur un *blueprint* Flask dédié, afin d'assurer l'authentification auprès de l'API Quadrige, l'interrogation des programmes disponibles et la structuration des données nécessaires à un futur import dans GeoNature. Une attention particulière a été portée à la construction des requêtes GraphQL, afin de cibler les informations utiles tout en limitant le volume des échanges.

Le développement du frontend a permis de mettre en place une première interface d'exploration des programmes Quadrige directement dans GeoNature. Bien qu'encore partielle, cette interface constitue une base solide pour les futures fonctionnalités de sélection, de filtrage et d'extraction. Ces travaux ont donné lieu à plusieurs phases de tests, notamment via PyTest pour le backend et Cypress pour l'interface utilisateur, afin de valider les mécanismes centraux avant leur intégration complète.

Travaux parallèles sur l'intégration de Pl@ntNet En parallèle du développement principal consacré à Quadrige, j'ai également initié un travail structurant autour de l'intégration des données Pl@ntNet dans GeoNature. L'objectif était double : d'une part valider la possibilité technique d'interroger l'API Pl@ntNet depuis un service Python, et d'autre part poser les bases d'un parseur Darwin Core utilisable par un module GeoNature, selon une logique similaire à celle mise en place pour Quadrige.

Pour cela, j'ai développé un client Python générique capable de :

- envoyer des requêtes POST vers l'endpoint `dwc/occurrence/search` de l'API Pl@ntNet ;
- gérer dynamiquement les paramètres taxonomiques, spatiaux (bbox ou polygone GeoJSON) et temporels ;
- enregistrer les réponses brutes au format JSON ;
- transformer ces réponses en un CSV conforme au standard Darwin Core (DwC), en s'appuyant sur des fichiers de configuration modulaires.

Une attention particulière a été portée à la normalisation des données :

- homogénéisation des dates au format ISO-8601 ;
- nettoyage et standardisation des coordonnées ainsi que du datum géodésique ;
- génération d'identifiants stables (`occurrenceID`) ;
- mappage des vocabulaires Pl@ntNet vers ceux utilisés dans GBIF (`basisOfRecord`) ;
- sérialisation propre des champs complexes (listes, objets JSON).

Ce travail a conduit à la mise en place d'un pipeline complet :

$$\text{API} \rightarrow \text{JSON brut} \rightarrow \text{CSV Darwin Core normalisé}$$

entièrement automatisé et adaptable à d'autres territoires ou taxons via un simple fichier de configuration.

Ce client constitue aujourd'hui la base technique du parser Pl@ntNet destiné au module `API2gn`, dont l'intégration a nécessité plusieurs ajustements dans les migrations SQL et dans la structure interne du module.

Enfin, une documentation détaillée a été rédigée tout au long du projet. Elle décrit les procédures d'appel aux API, les choix techniques effectués ainsi que les étapes d'installation et de maintenance du module. Ce travail de cAPItalisation est essentiel pour permettre la reprise du projet par la DEAL ou ses partenaires.

5.2. Difficultés rencontrées et solutions apportées

Le déroulement du stage a été marqué par un ensemble de contraintes techniques qui ont largement influencé l'organisation du travail. L'absence d'une installation locale stable de GeoNature a constitué l'un des principaux obstacles : les incompatibilités entre les versions de Debian, les erreurs liées au serveur applicatif et les difficultés rencontrées avec Docker ont rendu nécessaire une approche alternative, centrée sur une instance de travail personnalisée. Cette situation a été accentuée par les limitations du réseau de la DEAL, où la présence d'un proxy strict perturbait l'utilisation de nombreux outils de développement. La mise en place d'un mécanisme dynamique de gestion du proxy a progressivement permis de stabiliser l'environnement.

La migration des serveurs du SINP-974 vers un nouvel hébergeur a également retardé l'accès à un environnement de test réaliste. Pendant plusieurs semaines, le développement a dû être poursuivi sans possibilité de déployer ou de tester directement sur l'environnement cible. Une fois l'accès ouvert via une machine virtuelle bastion, les premières manipulations ont été freinées par les restrictions imposées par l'outil d'accès à distance. L'usage de Git comme canal de transfert a alors été déterminant pour poursuivre les tests et ajuster le module.

Ces difficultés ont nécessité une grande capacité d'adaptation et ont conduit à affiner l'organisation du projet : clarification des environnements, adoption d'outils de gestion de version adaptés, multiplication des tests exploratoires et documentation progressive pour sécuriser les phases suivantes d'intégration.

Difficultés spécifiques au travail sur Pl@ntNet Le développement du client Pl@ntNet a également présenté plusieurs défis techniques. L'API renvoie une structure JSON très hétérogène selon les taxons, les médias associés ou les métadonnées disponibles, ce qui a nécessité la création d'un parseur flexible, capable d'absorber des variations importantes de schéma sans échouer.

Par ailleurs, le standard Darwin Core impose un modèle strict qui ne correspond pas directement aux champs fournis par Pl@ntNet. Il a donc fallu définir un mapping robuste et configurable pour garantir la compatibilité avec GeoNature et Borbonica.

Enfin, l'intégration du parser dans le module `api2gn` a nécessité de modifier certaines migrations SQL existantes, qui empêchaient l'installation correcte du module. Ces ajustements feront l'objet d'un retour formalisé aux développeurs officiels de GeoNature afin d'améliorer la compatibilité du module avec des sources de données externes diversifiées.

5.3. Compétences acquises

Ce stage m'a permis de développer une maîtrise approfondie de l'écosystème technique de GeoNature, tant sur le plan backend (Python, Flask, SQLAlchemy, Marshmallow) que frontend (Angular, TypeScript). J'ai acquis une expérience concrète dans l'utilisation d'API REST et GraphQL, dans la mise en place de tests unitaires et fonctionnels, ainsi que dans la gestion rigoureuse d'environnements soumis à des contraintes fortes.

Ce travail m'a également permis de me familiariser avec les pratiques du développement open source, notamment via l'utilisation de forks GitHub, la gestion des issues et la rédaction de documentation technique destinée à des utilisateurs externes.

Au-delà des aspects techniques, cette expérience m'a amené à travailler en autonomie tout en collaborant étroitement avec des acteurs variés : agents en charge des données, administrateurs réseau, prestataires et agents du Parc national. J'ai pu renforcer mes compétences en organisation de projet, en documentation technique et en adaptation à des environnements complexes, autant de dimensions essentielles dans un contexte d'ingénierie logicielle appliquée aux enjeux environnementaux.

6. Perspectives

Les travaux réalisés au cours de ce stage ont permis de poser des bases techniques solides pour l'intégration de données externes au sein de GeoNature, en particulier pour les données issues du milieu marin via Quadrige, ainsi que pour les données participatives produites par Pl@ntNet. Plusieurs perspectives d'évolution ont été identifiées, tant sur le plan fonctionnel que technique. Elles visent à renforcer l'ergonomie des outils développés, à améliorer leur intégration dans l'écosystème GeoNature et à faciliter leur appropriation par les administrateurs et gestionnaires de données.

6.1. Perspectives d'évolution du module d'import Quadrige

Le module d'import Quadrige développé durant le stage constitue une première brique fonctionnelle permettant d'explorer, d'extraire et de préparer des données marines issues du système d'information Quadrige. Plusieurs axes d'amélioration peuvent être envisagés afin d'en accroître la maturité et de faciliter son intégration opérationnelle à long terme.

6.1.1. Amélioration de l'ergonomie et de l'intégration visuelle

Une première évolution concernerait l'intégration visuelle du module au sein de l'interface GeoNature. L'ajout d'un pictogramme dédié dans la barre de navigation ou dans la liste des modules permettrait d'identifier plus clairement le module Quadrige parmi les autres extensions disponibles.

Ce pictogramme contribuerait à améliorer l'expérience utilisateur en rendant le module plus visible et plus intuitif pour les administrateurs amenés à l'utiliser ponctuellement. Il faciliterait également son appropriation par de nouveaux utilisateurs, notamment dans un contexte de reprise du projet ou de déploiement sur d'autres territoires.

6.1.2. Ajout d'une étape de mise en forme et d'intégration des données

À l'heure actuelle, le module produit des archives de données (fichiers CSV bruts, CSV filtrés, fichiers JSON de paramètres) à l'issue de la phase d'extraction depuis l'API Quadrige. Ces archives constituent une étape intermédiaire volontaire, permettant un contrôle manuel des résultats avant toute intégration dans GeoNature.

Une évolution naturelle du module consisterait à ajouter une étape supplémentaire, positionnée après la génération des fichiers ZIP, dédiée à la mise en forme finale et à l'intégration des données dans la base GeoNature. Cette étape pourrait inclure la transformation des données vers le schéma de synthèse, ainsi que leur préparation pour un import direct dans la base de données.

Cette évolution permettrait de proposer une chaîne complète allant de l'extraction à l'intégration, tout en conservant la possibilité de valider manuellement les données en amont. Elle resterait compatible avec les exigences de traçabilité et de contrôle associées au SINP-974.

6.1.3. Clarification et extension de la gestion de la configuration

Le module repose déjà sur un fichier de configuration centralisé, chargé côté backend dans l'environnement GeoNature. Ce fichier permet notamment de définir les paramètres techniques essentiels tels que l'URL de l'API Quadrige et le jeton d'authentification, qui sont ensuite mis à disposition du backend.

Le backend expose également le contenu de ce fichier de configuration via une route dédiée, permettant théoriquement au frontend d'accéder dynamiquement à ces informations. Toutefois, par manque de temps, certaines données métier — telles que les localisations suggérées lors du filtrage des programmes ou la liste des champs disponibles pour l'extraction — restent définies directement dans le frontend.

Une évolution souhaitable consisterait à étendre le rôle du fichier de configuration afin d'y inclure ces éléments métier, puis à adapter le frontend pour les charger dynamiquement depuis le backend. Cette approche permettrait de réduire les valeurs codées en dur dans l'interface, d'améliorer la cohérence entre backend et frontend et de faciliter l'adaptation du module à d'autres territoires ou contextes d'usage.

6.1.4. Vers une automatisation partielle et maîtrisée des imports

À plus long terme, le module pourrait évoluer vers une automatisation partielle ou contrôlée des imports, par exemple via des imports périodiques planifiés, des imports incrémentaux basés sur les dates de mise à jour des données, ou des déclenchements manuels avec validation préalable.

Toutefois, certaines spécificités des données Quadrige imposent de conserver un contrôle humain sur les résultats des extractions. Certaines informations peuvent être anonymisées (par exemple les données relatives aux observateurs) ou soumises à des moratoires de diffusion. Il est donc actuellement nécessaire de vérifier les résultats des extractions avant tout traitement visant à leur intégration dans GeoNature.

Ces contraintes devront être pleinement prises en compte dans toute évolution vers une automatisation plus poussée, afin de rester cohérent avec les exigences du SINP-974 et les pratiques de contrôle des données naturalistes.

6.1.5. Renforcement de la stratégie de tests et de validation

Les modules externes GeoNature sont conçus pour s'inscrire dans une logique de maintenance à long terme. À ce titre, la documentation officielle recommande la mise en place de tests automatisés, notamment des tests unitaires et fonctionnels côté backend (via PyTest) ainsi que des tests de bout en bout (*end-to-end*) côté frontend (via Cypress).

Dans le cadre de ce stage, des tests ont été développés et validés sur la version locale du

module Quadrige, afin de vérifier le bon fonctionnement des principaux composants : appels à l'API Quadrige, gestion des erreurs, génération des fichiers d'export et cohérence des routes exposées par le backend. De la même manière, des scénarios de tests frontend ont été amorcés pour valider les parcours utilisateurs essentiels.

Toutefois, en raison des contraintes de temps et du calendrier de déploiement sur les serveurs du SINP-974, ces tests n'ont pas pu être intégrés dans la version du module effectivement déployée sur l'environnement serveur.

Une perspective importante consisterait donc à intégrer ces suites de tests dans la version déployée du module, afin de renforcer sa robustesse, de sécuriser les évolutions futures et de faciliter sa reprise par d'autres développeurs. La généralisation des tests automatisés permettrait également de s'inscrire plus étroitement dans les bonnes pratiques recommandées pour le développement et la maintenance des modules externes GeoNature.

6.2. Perspectives autour du parser Pl@ntNet et du module api2gn

Les travaux menés sur l'intégration des données Pl@ntNet ont permis de valider la faisabilité technique d'un pipeline complet, depuis l'interrogation de l'API jusqu'à la production de fichiers conformes au standard Darwin Core. Toutefois, ces données étant issues d'une application participative, leur exploitation soulève des enjeux spécifiques de validation et de qualité scientifique.

6.2.1. Ajout d'une interface d'administration dédiée

Une perspective majeure consisterait à proposer une interface frontend permettant aux administrateurs de paramétrer le parser Pl@ntNet sans recourir à la modification de fichiers de configuration. Cette interface pourrait permettre de définir les taxons ciblés, les périmètres géographiques, les plages temporelles ainsi que les paramètres de filtrage et de normalisation.

Une telle évolution améliorerait significativement l'ergonomie du dispositif et faciliterait son utilisation par des profils non techniques. Elle permettrait également d'harmoniser l'expérience utilisateur avec celle du module Quadrige, en proposant une logique d'interaction cohérente au sein de GeoNature.

6.2.2. Contraintes liées au module api2gn

Dans l'état actuel, cette évolution n'est toutefois pas immédiatement réalisable. Le module `api2gn` est un module officiel de GeoNature, dont l'architecture n'a pas été conçue à l'origine pour intégrer des interfaces frontend spécifiques à chaque source de données.

Toute extension de ce type nécessiterait soit une évolution structurelle du module `api2gn` lui-même, soit la création d'un module externe complémentaire venant piloter le parser Pl@ntNet. Cette réflexion dépasse le cadre du stage, mais les travaux réalisés constituent une base technique solide pour initier ce type d'évolution à l'avenir.

6.2.3. Renforcement des mécanismes de validation et d'incrémentalité

Les données Pl@ntNet étant saisies par des utilisateurs non experts, la validation des taxons constitue un enjeu central. Le parser développé intègre déjà un mécanisme de validation basé sur la comparaison des *scientific names* avec les référentiels taxonomiques utilisés par GeoNature, ainsi que sur des interrogations complémentaires de services externes lorsque cela est nécessaire. Un historique des taxons validés et rejetés est également conservé.

Une perspective d'amélioration consisterait à renforcer ces mécanismes, notamment en affinant la gestion des requêtes incrémentales. Actuellement, les extractions sont réalisées par blocs de 1 000 occurrences. Il pourrait être pertinent de permettre une reprise plus fine des extractions interrompues, afin d'éviter de relancer l'ensemble des requêtes en cas d'arrêt partiel du traitement.

6.3. Ouvertures transversales

Les développements réalisés durant ce stage ont permis de mettre en évidence l'intérêt d'adapter les stratégies d'intégration des données en fonction de la nature et de l'origine des sources mobilisées. Les travaux menés sur Quadrige et Pl@ntNet illustrent deux approches complémentaires, répondant à des contraintes distinctes.

Pour les données issues de Quadrige, produites dans un cadre institutionnel et scientifique, la priorité a été donnée à la traçabilité et au contrôle des extractions. La séparation entre les phases d'extraction, de préparation et d'intégration permet de conserver un contrôle humain sur des données pouvant être soumises à des règles de diffusion spécifiques (anonymisation, moratoires), tout en préparant leur intégration progressive dans GeoNature.

À l'inverse, les données issues de Pl@ntNet, bien que participatives, peuvent faire l'objet d'un traitement largement automatisé. Le pipeline mis en place intègre des mécanismes de validation taxonomique reposant sur les référentiels utilisés par GeoNature, permettant d'identifier, de corriger ou de rejeter automatiquement les observations non conformes. Les données validées peuvent ainsi être intégrées directement dans la base GeoNature, sans intervention manuelle, tout en conservant un historique des décisions prises par le système.

Cette distinction entre intégration contrôlée et intégration automatisée souligne l'importance d'une architecture d'import flexible, capable de s'adapter aux caractéristiques des sources de données. Elle ouvre la voie à une gestion différenciée des flux, conciliant exigences de qualité scientifique, efficacité opérationnelle et montée en charge des systèmes d'information naturalistes.

7. Conclusion

Bibliographie

Annexes

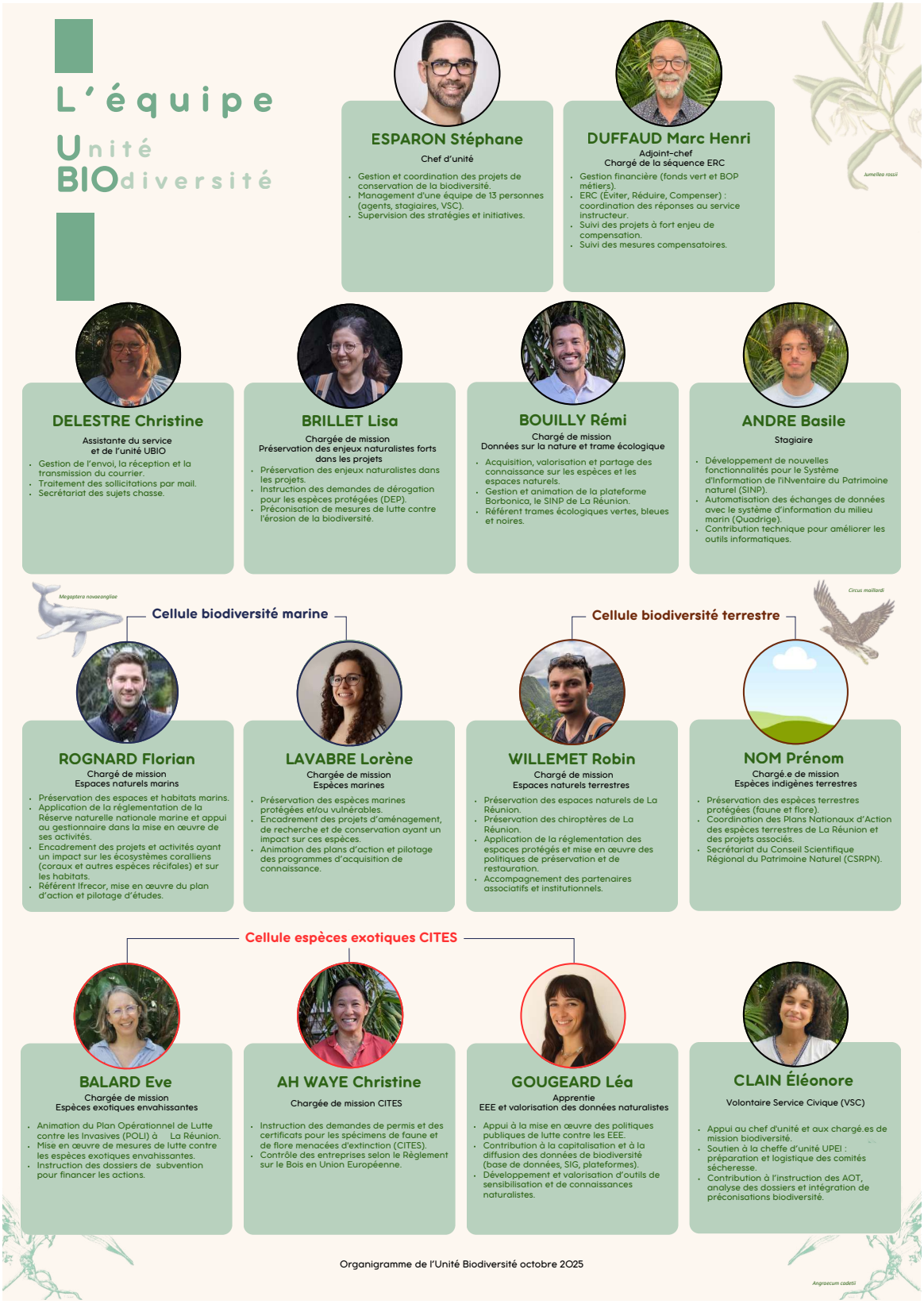

Figure 7.1: Organigramme de l'Unité Biodiversité (UBIO) – SEB / DEAL Réunion

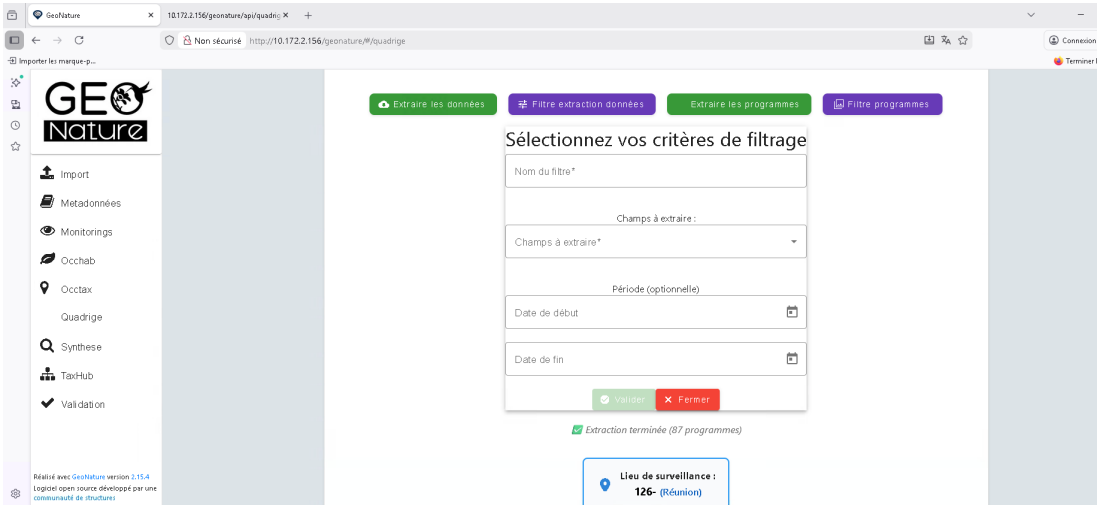

Figure 7.2: Interface de filtrage des données extraites pour les programmes Quadrige

Documentation du module Pl@ntNet

La documentation associée au module Pl@ntNet a été produite sous forme de fichiers Markdown afin de faciliter sa maintenance et sa réutilisation.

Elle comprend :

- une documentation technique décrivant l'architecture du client Python et du parser api2gn
- une documentation utilisateur détaillant les paramètres configurables et les cas d'usage
- un fichier `README.md` présentant le projet, son installation et ses dépendances

L'ensemble de cette documentation est disponible dans le dépôt GitHub, dans le dossier documentation : documentation

Webographie

- **Quadrige**
  - Documentation officielle de l'API Quadrige
  - Tutoriels vidéo Ifremer
  - Accès à l'application Quadrige
- **GeoNature**
  - Site officiel GeoNature
  - Dépôt GitHub du projet
  - Documentation générale
  - Guide de développement des modules externes
- **DEAL Réunion**
  - Site officiel de la DEAL Réunion
  - Présentation institutionnelle de la DEAL Réunion
- **Pl@ntnet**
  - Documentation de l'API Pl@ntNet
  - Présentation du format Darwin Core sur le site de GBIF
  - Module api2gn (officiel GeoNature)
- **Liens vers les dépôts GitHub des modules développés**
  - Fork personnel du module api2gn (ajout du parser Pl@ntNet et adaptations des migrations)
  - **Module Quadrige**
    - Version locale avec tests fonctionnels
    - Version finale en cours d'intégration dans GeoNature
  - ***Module Pl@ntNet***
    - Client Python, normalisation Darwin Core et documentation