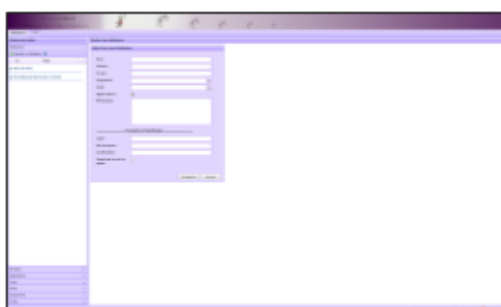




Rapport de Stage:

Refonte d'une application web de gestion centralisée des utilisateurs



Rapport de :

Laumond Gabin
Dut Informatique
Année 2017-2018

Maître de stage:

Camille Monchicourt
Parc national des Ecrins

Tuteur Pédagogique:

Fanny Binet
IUT Charlemagne



UNIVERSITÉ
DE LORRAINE



nancy

Charlemagne
Info-Com

IUT Nancy Charlemagne

Université de Lorraine

2 ter boulevard Charlemagne

BP 55227

54052 Nancy Cedex

Département informatique

Refonte d'une application web de gestion centralisée des utilisateurs

Rapport de stage de DUT informatique

Entreprise : Parc national des Ecrins

Laumond Gabin

Tuteur : Fanny Binet

Année universitaire 2017–2018

Remerciements

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

Tout d'abord, j'adresse mes remerciements à mes parents qui m'ont beaucoup aidé dans ma recherche de stage et m'ont permis de postuler dans cette entreprise. Ils m'ont permis de cibler mes candidatures, et de trouver ce stage qui était en totale adéquation avec mes attentes.

Je tiens à remercier vivement mon maître de stage, Monsieur Camille Monchicourt, responsable du pôle Système d'Informations (SI) au sein du Parc national des Ecrins, pour son accueil, le temps passé ensemble et la confiance qu'il m'a accordé dès mon arrivée dans l'entreprise.

Je remercie également Théo Lechémi, pour le soutien et les connaissances qu'il m'a apporté au quotidien.

Enfin, je tiens à remercier toute l'équipe du pôle SI, du service Scientifique ainsi que tous les collègues du Parc national des Ecrins pour leur accueil chaleureux et leur disponibilité en cas de problème.

Table des matières

Rapport de Stage:	1
Refonte d'une application web de gestion centralisée des utilisateurs	1
Refonte d'une application web de gestion centralisée des utilisateurs	3
Remerciements	4
Table des matières	5
Présentation de l'entreprise et du service	8
• Le Parc national des Ecrins	8
• Mon rôle durant ce stage	9
• Les missions du SI	9
• Contexte applicatif du stage	11
• Généricité	13
• Objectif du stage	14
Présentation du projet de stage	15
• UsersHub version 1	15
• Objectif UsersHub version 2	17
Déroulement du stage et évolution de l'application	23
• Mon diagramme de Gantt	23
• Mes deux premières semaines de stage, découverte des technologies	23
• Affichage des autres tables du schéma «Utilisateur»	25
• Ajout d'un élément	26
• Séparation utilisateurs/groupes	27
• Modification d'un élément	28
• Suppression	29
• Récapitulatif	29
• Ajouts de membre	30
• Étiquetage d'un rôle, d'une application ou d'un organisme	31
• Type de tag	32
• CRUVED	32
• Droits application	34
Conclusion	37

Introduction

Mon stage se déroule au sein du Parc national des Ecrins, établissement public national dépendant du ministère de l'Environnement. Un parc national est un territoire généralement vaste dont la richesse biologique, la qualité paysagère, l'intérêt culturel et le caractère historiquement préservé justifient une protection et une gestion qui garantissent la pérennité de ce patrimoine considéré comme exceptionnel. En France, ils ont été créés par la loi du 22 juillet 1960. Le Parc national des Ecrins a été créé le 27 mars 1973, il est devenu le cinquième parc national français après les parcs de la Vanoise (1960), de Port-Cros (1963), des Pyrénées (1967) et des Cévennes (1970). Aujourd'hui il existe 10 parcs nationaux en France.

Au départ, l'objectif des parcs nationaux était de récolter des données scientifiques afin de mieux comprendre la biodiversité et la géologie présentes. Depuis 2006 et l'instauration d'une «Charte des Parcs Nationaux», le territoire du parc national est divisé en deux parties. D'une part, le coeur du parc national avec une réglementation stricte et sans habitat humain et d'autre part, une aire d'adhésion comprenant les communes acceptant la charte du parc national.

En intégrant des communes au parc national, celui-ci entre dans une nouvelle dynamique. Une dynamique d'ouverture et de diffusion des données est mise en place. Celles-ci deviennent donc accessibles au grand public.

Dans le cadre de cette dynamique d'ouverture, et de l'arrivée des nouvelles technologies, le Parc national des Ecrins (PNE) a réorganisé ses services en regroupant l'équipe informatique et l'équipe géomatique afin de former un pôle Système d'Informations (SI). Actuellement, le SI permet de faire le lien entre la collecte de données qui étaient sur papier puis saisies manuellement sur tableurs et la collecte numérique et un transfert automatique dans des bases de données spatiales centralisées.

Ces données, faunes et flores, sont récupérées grâce à la première version de GeoNature. GeoNature est une application qui permet aux agents de terrain de recenser leurs observations.

Ainsi une fois les données stockées dans des bases de données, le SI a développé des applications web qui mettent en lumière la biodiversité et la singularité du territoire auprès du grand public.

Aujourd'hui trois applications existent :

- L'application «Bouquetin» qui permet de suivre en temps réel le parcours de plusieurs bouquetins équipés de capteur GPS (<http://bouquetins.ecrins-parcnational.fr/>)
- L'application «Rando Ecrins», permet au grand public de rechercher un itinéraire de randonnée selon le lieu, la distance, la difficulté et les points d'intérêts choisis. Application qui existe aussi sur mobile. (<http://rando.ecrins-parcnational.fr/fr/>)
- L'application «Biodiv'Ecrins» (GeoNature-Atlas), celle-ci est un atlas de la faune et de la flore observées par les agents du parc national depuis 1973. Ainsi vous avez accès à des fiches espèces, des photos, et les lieux où les espèces ont été observées. (<http://biodiversite.ecrins-parcnational.fr/>)

Le SI publie toutes ses applications en Open-Source pour privilégier le travail collaboratif et le partage des outils et des connaissances.

Dans un premier temps, je présenterai l'entreprise, le service et l'équipe avec lesquels j'ai travaillé ainsi que mon rôle durant ce stage. Dans un second temps, je vous présenterai le projet de mon stage et son contexte. Enfin, je m'attacherai à développer chronologiquement les modalités de réalisation de l'application.

Présentation de l'entreprise et du service

- Le Parc national des Ecrins

Le Parc national des Ecrins (PNE) est un établissement public national créé en 1973 qui a pour vocation la préservation de la biodiversité et la diffusion des connaissances. Il est situé dans les Alpes du Sud. Le territoire s'étend sur 53 communes sur les départements des Hautes-Alpes (région Provence-Alpes-Côte d'Azur) et de l'Isère (région Rhône-Alpes-Auvergne).

L'équipe permanente du PNE est constituée de 92 employés, répartis entre le siège à Gap (lieu de mon stage) et sept secteurs qui couvrent l'ensemble du territoire du parc national (figure 1).

Au sein du siège il existe quatre services : le service scientifique, le service aménagement, le service communication et le service de secrétariat général.



Figure 1 : Carte de la localisation du PNE et de ses secteurs

- Mon rôle durant ce stage

Mon stage se déroule au sein du siège du parc national, établi dans le château du domaine de Charance à Gap. En tant que stagiaire, j'ai intégré le service scientifique du PNE. Celui-ci est divisé en deux pôles:

- le pôle connaissance qui travaille à la mise en place de protocoles de suivis scientifiques (faune, flore et mesures physiques),
- le pôle Système d'Informations, dans lequel je me trouve, qui s'occupe de la géomatique et de l'informatique. Il est composé de deux chargés de mission base de données et développement web , Gil Deluermoz et Théo Lechémi, d'un chargé de mission administration réseau, téléphonie et informatique, Vincent Pietri, et enfin d'un géomaticien qui est également chef du pôle, Camille Monchicourt, mon tuteur.

Durant ce stage ma mission a été la refonte d'une application web de gestion centralisée des utilisateurs. J'ai été amené à travailler en grande partie avec Théo Lechémi que l'on peut considérer comme mon tuteur technique.

- Les missions du SI

Le pôle SI occupe une position transversale puisqu'il est amené à travailler avec tous les services du parc national. Il assiste aussi bien le service scientifique dans la mise en place de protocoles de suivi faune-flore, que le service aménagement dans le suivi du patrimoine bâti et de l'agriculture, ou encore le service communication dans la mise en place d'outils de mise en valeur des sentiers de randonnées et l'animation du site web. De part ses missions de protection de la faune et de la flore, le parc national est amené à collecter des quantités importantes de données spatialisées. Le rôle du SI au sein du parc national est donc d'organiser et de faciliter la collecte de ces données, de les gérer mais également de créer des outils pour les analyser. Une grande composante du métier du SI tient donc dans l'administration de bases de données. Le PNE a été novateur dans la mise en place de la collecte et du stockage des données sur informatique et possède aujourd'hui une architecture de base de données et des outils structurés. Le schéma ci-dessous (figure 2) résume la modernisation de la stratégie générale du SI, du recueil de la donnée jusqu'à son traitement et sa consultation. En fonction des protocoles et des besoins des agents, plusieurs chaînes de travail ont été mises en place. La collecte sur le terrain, anciennement effectuée sur papier (en

blanc sur le schéma) est aujourd'hui saisie sur des outils nomades (applications mobiles sur tablette). Pour les protocoles importants, l'ensemble de ces données sont ensuite centralisées dans des bases de données PostgreSQL (avec extension PostGIS pour les besoins spatiaux). A partir de ces bases de données, des applications de consultation métier ou grand public sont développées soit par des prestataires extérieurs, soit directement par le SI.

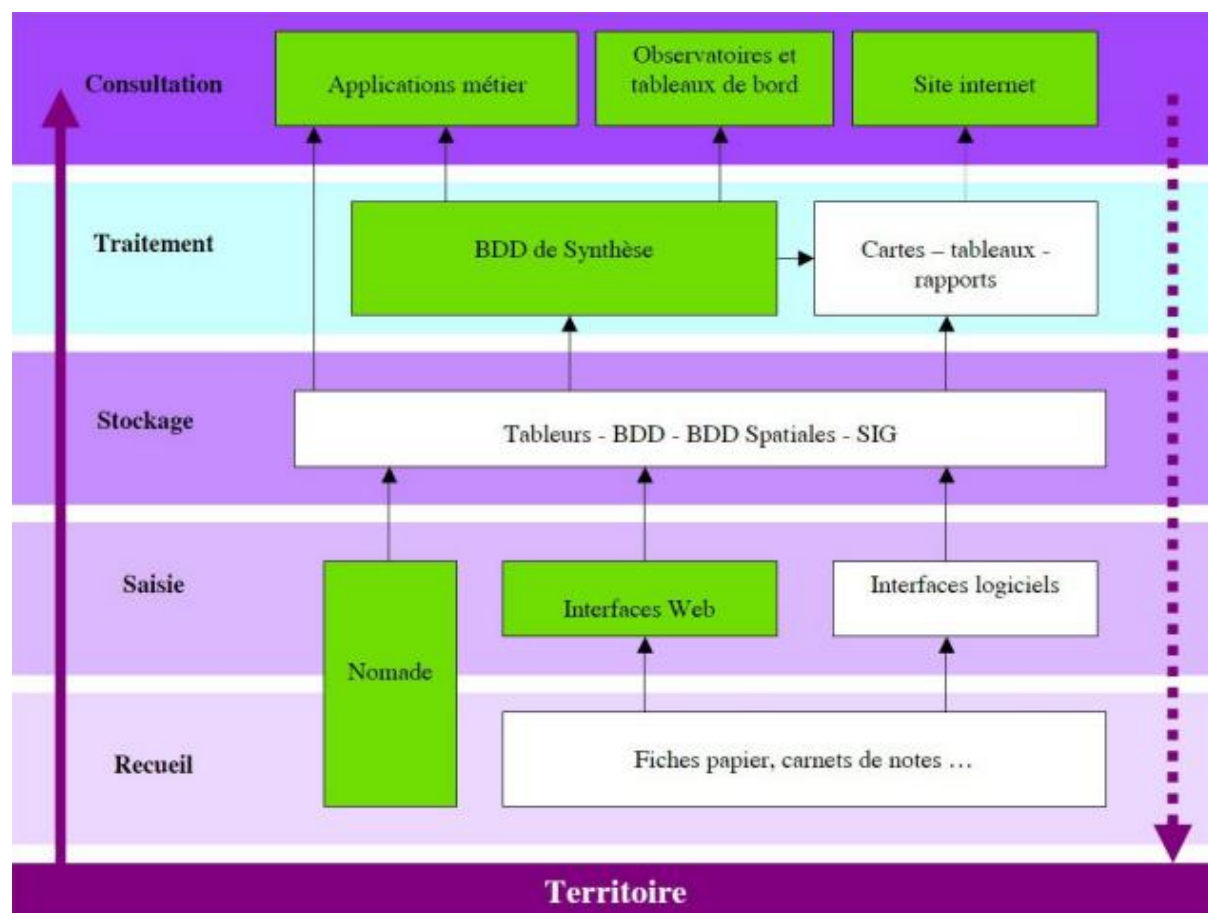


Figure 2: La chaîne de travail du SI: du recueil à la consultation des données (document interne). On distingue en vert la chaîne de travail actuelle et en blanc l'ancienne.

Contexte et Objectifs

- Contexte applicatif du stage

Dès le dépôt de ma candidature au Parc national des Ecrins, en consultant le site web, j'ai vu que le service informatique travaillait sur une refonte de l'application GeoNature.

Mais qu'est ce que GeoNature ?

GeoNature est un outil open source développé par les parcs nationaux français. C'est une application de saisie et de synthèse des observations faune et flore. Elle permet de regrouper l'ensemble des données provenant des protocoles Faune et Flore, de saisir les protocoles de contact occasionnel Faune et Flore Station et de consulter l'ensemble de ces données dans une application de synthèse. Celle-ci regroupe toutes les données des différents protocoles FAUNE et FLORE en les limitant au niveau QUI QUOI QUAND OU.

Le parc national réalise un inventaire de la faune et de la flore depuis la création de celui-ci. En plus de protocoles particuliers de suivi de certaines espèces, les agents relèvent, à chaque sortie de terrain les espèces qu'ils observent. Le parc national dispose ainsi de plus de 550 000 données d'observations pour plus de 5000 espèces différentes.

GeoNature se base sur deux autres applications : TaxHub et UsersHub.

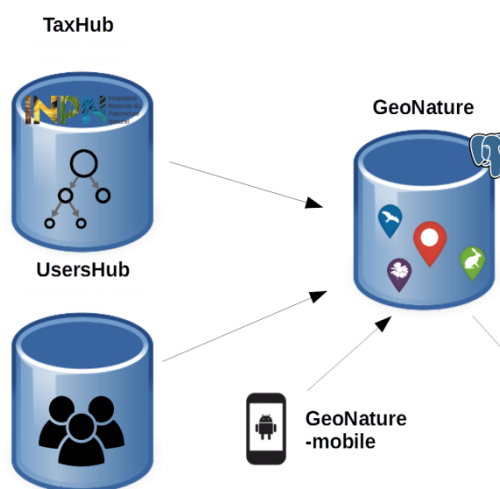
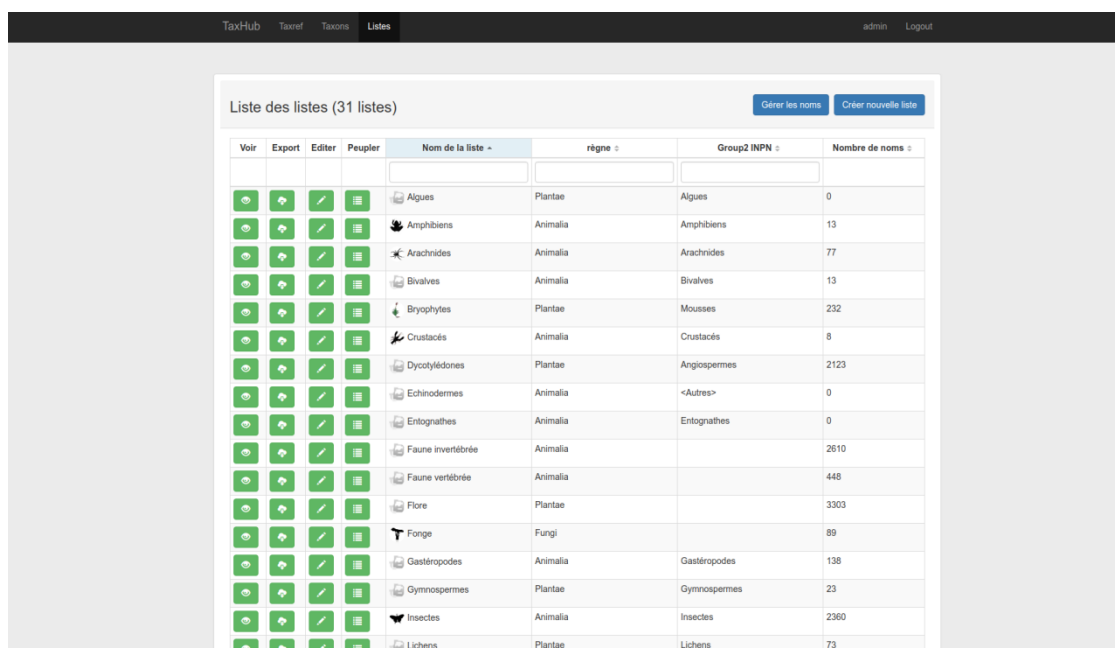


Figure 3 : Architecture simplifiée de l'outil GeoNature.

TaxHub est une application développée par les parcs nationaux français. Cette application permet de gérer les informations relatives aux espèces et d'administrer la base de données des taxons. Elle s'appuie sur une base de données nationale de référence nommée TAXREF (<https://inpn.mnhn.fr/programme/referentiel-taxonomique-taxref>) administrée par le Muséum National d'Histoire Naturel. Elle centralise toutes les espèces existantes sur le territoire français. Dans la base, chaque espèce est identifiée par un identifiant unique : le « cd_nom », ce qui facilite grandement l'échange des données naturalistes entre les structures. Dans l'application, les chargés de mission faune et flore peuvent ainsi ajouter une espèce nouvellement découverte dans le parc national, lui associer une description ou encore renseigner des caractéristiques spécifiques. Une fonctionnalité en lien avec l'atlas permet d'associer à chaque espèce des médias (photo, vidéos, son) ou encore des articles.



Voir	Export	Editer	Peupler	Nom de la liste	règne	Group2 INPN	Nombre de noms
				Algues	Plantae	Algues	0
				Amphibiens	Animalia	Amphibiens	13
				Arachnides	Animalia	Arachnides	77
				Bivalves	Animalia	Bivalves	13
				Bryophytes	Plantae	Mousses	232
				Crustacés	Animalia	Crustacés	8
				Dicotylédones	Plantae	Angiospermes	2123
				Echinodermes	Animalia	<Autres>	0
				Entognathes	Animalia	Entognathes	0
				Faune invertébrée	Animalia		2610
				Faune vertébrée	Animalia		448
				Flore	Plantae		3303
				Fonge	Fungi		89
				Gastéropodes	Animalia	Gastéropodes	138
				Gymnospermes	Plantae	Gymnospermes	23
				Insectes	Animalia	Insectes	2360
				Lichens	Plantae	Lichens	73

Figure 4: interface de Taxhub listant des espèces animales

UsersHub est une application web, développée par le PNE, permettant de regrouper l'ensemble des utilisateurs d'applications web afin de gérer de manière différenciée et centralisée les droits d'accès à ces applications ainsi que le contenu des listes déroulantes d'observateurs. Elle permet de gérer de manière centralisée des utilisateurs et de les placer dans des groupes ; de créer différents niveaux de droits et de les affecter aux utilisateurs et/ou

aux groupes d'utilisateurs pour chacune de nos applications. Elle permet également de gérer des organismes, des unités et des listes déroulantes regroupant des utilisateurs ou des groupes d'utilisateurs.

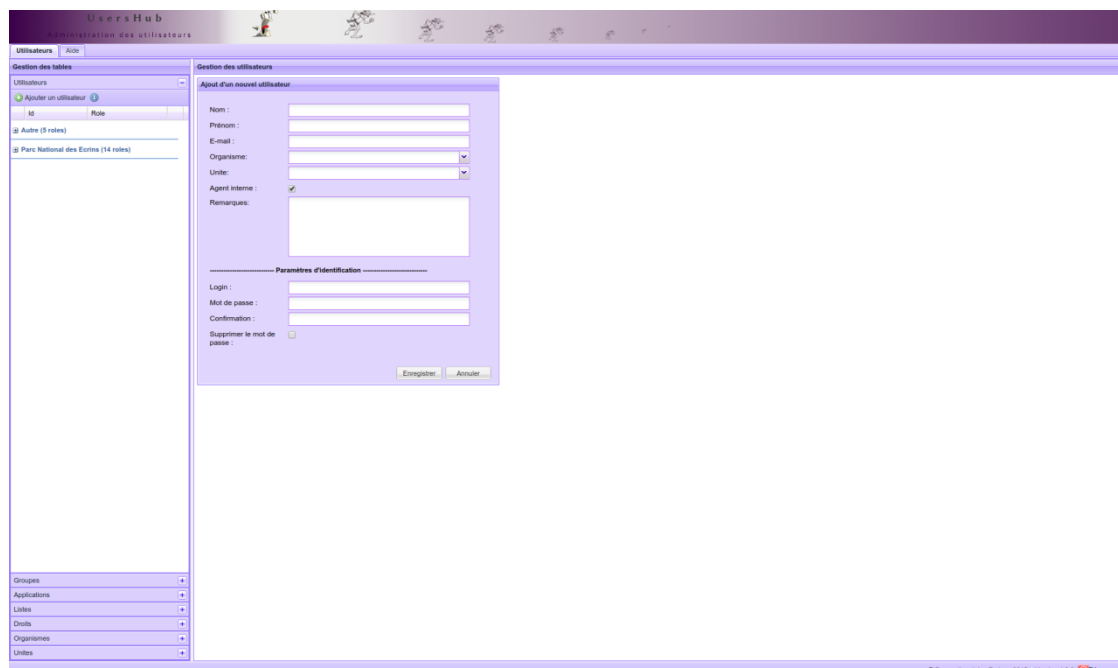


Figure 5 : interface actuelle de UsersHub

- **Généricité**

Le parc national, s'est engagé depuis le début de ses développements dans une démarche open-source. À l'image de GeoNature ou TaxHub, les diverses applications développées sont amenées à être déployées dans d'autres parcs naturels ou dans d'autres structures dont les besoins pourraient être similaires. Chaque application est donc développée dans un souci de généricité afin qu'elles puissent être facilement partagées. Cette démarche est actuellement commune à de nombreux Parcs nationaux et permet le partage et la mutualisation des coûts de développement. Un effort important est donc fait sur la généricité et la documentation du code. La totalité des codes-sources sont publiées sous licence libre (GPL V3) sur Github (<https://github.com/PnEcrins/> et <https://github.com/PnX-SI>).

- Objectif du stage

GeoNature est actuellement en refonte complète et dans sa nouvelle version, l'application utilisera un nouveau modèle de base de données. Comme le montre la figure 3 GeoNature repose sur deux autres applications, UsersHub et TaxHub. Suite à la refonte de la base de données de GeoNature, une partie du modèle de données de UsersHub devient obsolète. De plus, elle utilise une technologie vieillissante.

Mon stage consiste donc à réaliser une refonte de cette application UsersHub, s'adaptant à la nouvelle base de données. La version actuelle est en php, mais les autres applications du PNE sont en python. Le parc national souhaite que le langage python soit utilisé pour UsersHub.

Présentation du projet de stage

- UsersHub version 1

Pour rappel UsersHub est une application web qui permet à un administrateur d'ajouter des utilisateurs, groupes, organismes, ou unités et de leur donner un niveau de droits pour une application donnée. Il peut aussi associer les utilisateurs et les groupes à des unités ou des organismes.

Principe général : UsersHub permet de gérer et de synchroniser le contenu d'un ou plusieurs schéma «Utilisateurs» d'une ou plusieurs bases PostgreSQL. A condition que le modèle mais aussi que toutes les données de ces bases soient identiques, UsersHub permet de maintenir le contenu du schéma «Utilisateurs» de ces bases strictement identiques.

Dans un Système d'Informations, les applications web 'métier' nécessitent généralement une identification par login/pass. Les applications disposent donc d'un dispositif de gestion des utilisateurs et de leur droits. L'utilisateur n'a pas forcément les mêmes droits d'une application à l'autre et l'administrateur doit maintenir une liste d'utilisateurs dans chacune des applications. Ces applications ont le plus souvent chacune une base de données dédiée. A condition d'organiser la gestion de ces utilisateurs de manière identique dans toutes les bases des applications web, UsersHub permet de centraliser cette gestion et de réaliser les modification dans toutes les bases en une seule opération.

UsersHub dispose de sa propre base de données mais utilise un fichier de settings permettant de déclarer les paramètres de connexion aux différentes bases. Lorsqu'une modification est réalisée dans la base principale, elle est reproduite dans toutes les bases déclarées dans ce fichier de settings. Si un utilisateur arrivent dans une structure, si un mot de passe doit être changé, il est nécessaire de le faire qu'une seule fois.

Une fois enregistré, un utilisateur peut être placé dans un groupe et ses droits d'accès à telle ou telle application web sont hérités des droits du groupe. Mais on peut aussi affecter des droits spécifiques à un utilisateur pour telle application ou telle autre. Si certains des utilisateurs ou groupe d'utilisateurs doivent figurer dans une liste déroulante de l'application

(par exemple une liste d'observateurs ou de représentants), UsersHub crée ces listes et en gère le contenu. Il ne reste alors plus qu'à utiliser cette liste dans l'application choisi.

La base de données utilisé, nécessite le schéma «Utilisateurs» suivant pour fonctionner. Ce schéma est inclus dans la base de donnée de GeoNature (voir annexe 1). Ainsi toutes bases de données respectant le schéma «Utilisateurs» peuvent utiliser l'application UsersHub v1.

Schéma Utilisateurs V1

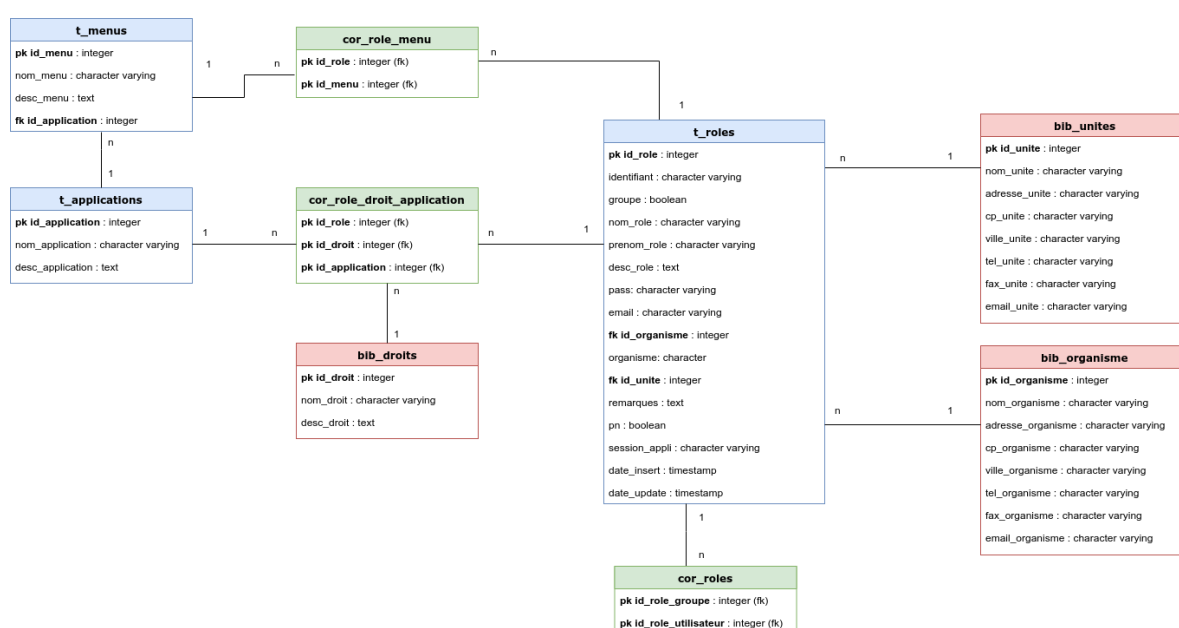


Figure 6 : Schéma «Utilisateurs» de la base de données.

Le nommage des tables facilite une compréhension de la base. Les tables avec pour préfixe «cor» définissent une table de correspondance entre deux autres tables. Le préfixe «t» définit les tables qui stockent les données dynamiques, par exemple pour certaines entreprises des tables comme «t_roles» pourront contenir des milliers de données. Enfin le préfixe «bib» définit des tables bibliothèques de valeurs, ce ne sont pas des tables qui grossiront énormément dans le temps, elles contiennent majoritairement des données statiques, utilisées pour les listes déroulantes.

Dans ce schéma, on retrouve une table «t_roles» qui stocke les utilisateurs ou les groupes (on différencie un groupe d'un utilisateur si la valeur de la colonne «groupe» est vrai), une table «t_applications» pour définir une application et une table «t_menus» pour la table des listes déroulantes d'utilisateurs des applications. Suite à ces tables, on génère des tables de correspondance entre elles.

«cor_role_menu» fait le lien entre un utilisateur et les listes déroulantes, «cor_role_droit_application» établit la correspondance entre un rôle, un droit et une application et enfin une dernière table de correspondance «cor_roles» qui gère la relation utilisateur-groupe.

«bib_droits» définit les niveaux de droits, cette table contient 6 droits, du niveau 1 au niveau 6 : utilisateur, rédacteur, référent, modérateur, validateur, et administrateur. «bib_organismes» contient la liste d'organismes et «bib_unité» définit les unités possibles (une unité peut être par exemple un service d'une entreprise, ou encore le conseil d'administration).

- Objectif UsersHub version 2

Le nouveau UsersHub doit s'adapter à un nouveau schéma «Utilisateurs». Le schéma est le suivant :

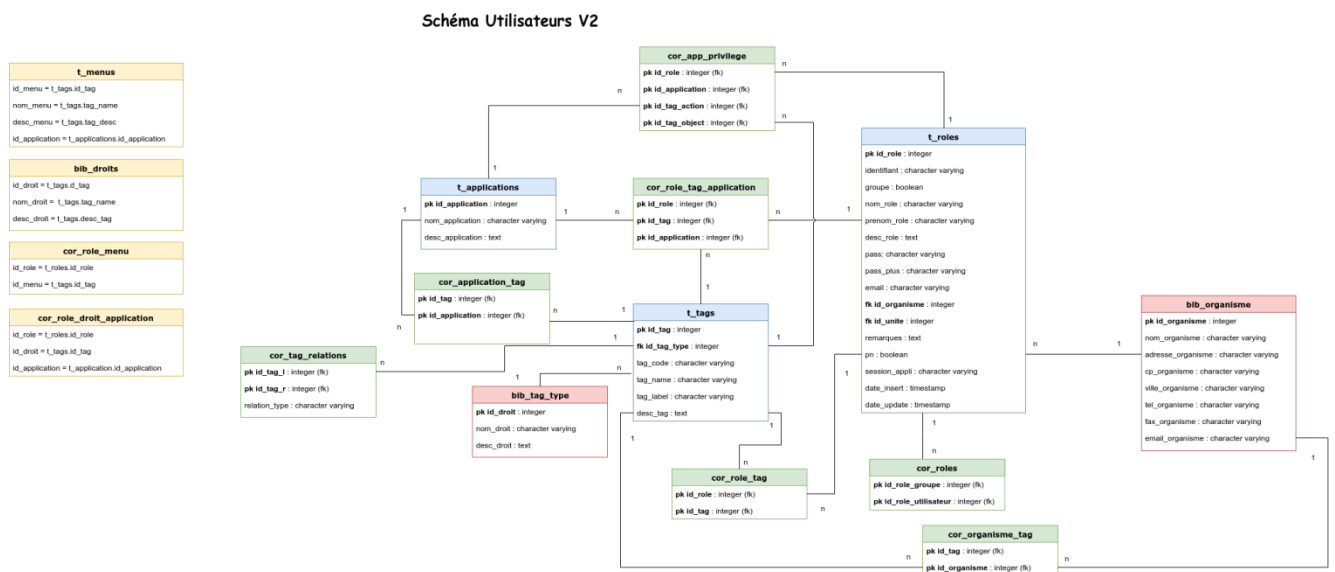


Figure 7: Schéma «Utilisateurs» UsersHub version 2

Comme on peut le voir, les droits les menus ont disparu et les tags sont apparus. Pour permettre à UsersHub v2 d'être compatible avec les anciennes applications et l'ancien schéma «Utilisateurs» Les vues se basent sur la nouvelle table «t_tags». Un tag est un mot clé que l'on peut associer à un objet de la base.

L'inconvénient d'utiliser des vues pour remplacer des tables importantes est la perte d'intégrité. Les vues ne contiennent pas de clé étrangère. Il est donc aussi impossible d'y stocker des valeurs, la vue se met automatiquement à jour par rapport aux tables sur lesquelles elle s'appuie. C'est pourquoi dans un futur proche il faudra surement ajouter des triggers aux vues afin qu'on ne perde aucune valeur en cas de changements de données.

Comment fonctionne donc les tags ?

Le mécanisme des tags est complexe mais permet une grande souplesse et généricité dans la répartition des droits aux rôles. Ainsi les tags permettent d'étiqueter des rôles, mais aussi des applications et des organismes.

Tout d'abord pour comprendre un tag, on doit connaître son type de tag d'où la table «bib_types_tag». Les types de tag peuvent être un objet, une action un privilège, une liste ou encore une portée. Ce sont les types de tags de base mais un administrateur peut en rajouter selon ses besoins.

Le type liste remplace par exemple la table «t_menus» de la version 1, le type privilege remplace la table «bib_droits».

Mais le système le plus intéressant de cette nouveauté est le CRUVED, mis en place pour gérer les droits des utilisateurs de manière plus avancée dans GeoNaure V2.

id_tag_type	tag_code	tag_name
2 (2 = type action)	C	Create
2	R	Read
2	U	Update
2	V	Validate
2	E	Export
2	D	Delete

Figure 8: tableau représentant les tags du CRUVED dans la table «t_tags»

Dans un schéma traditionnel on retrouve souvent le CRUD, les éléments “export” et «validate» n’existent pas. Mais dans le cas de l’application GeoNature, un rôle peut exporter ou valider des données, c’est pourquoi nous avons ces deux éléments en plus.

Les tags du CRUVED sont des types de tag «action». Mais ils sont liés à trois autres tags de type «Scope» (Scope = Portée) :

id_tag_type	tag_name
5 (5 = type Scope)	my data
5	my organism data
5	all data

Figure 8: tableau illustrant les tags de type Scope dans la table «t_tags»

Ainsi si on cumule un tag de type action, un tag de type Scope, une application et un rôle on établit un CRUVED pour un utilisateur, à une application. Cette relation est enregistrée dans la table «cor_app_privilege».

Pour les applications non-compatible au CRUVED, la table de correspondance «cor_role_tag_application» permet de lier un rôle, une application et un tag qui est un niveau de droits.

L’objectif est donc de réaliser une nouvelle application web de UsersHub pour implémenter cette évolution de la base de données.

Pour UsersHub version 2 l’objectif est aussi d’harmoniser les différentes applications web du SI et ainsi utiliser les mêmes technologies pour celles-ci.

De plus Usershub est une application, qui sera utilisée par l’administrateur de la structure la plupart du temps, donc l’application ne nécessite pas un moteur graphique puissant en front-end.

De ce fait, la programmation de cette application web sera réalisée en python pour plusieurs raisons :

- la souplesse, la puissance et la simplicité de ce langage
- c'est le langage préférentiel de la communauté open-source Système d'Informations Géographique (SIG), GeoNature, Taxhub et UsersHub sont des applications open-source.
- les autres applications du PNE sont désormais réalisées en python (GeoNature v2, UsersHub, TaxHub, GeoNature-Atlas)

Voici donc les technologies que j'ai utilisées pour réaliser mon projet:



Figure 9 : Illustrations de technologies utilisées

Pour la base de données, le PNE utilise PostgreSQL, c'est un moteur de base de données open-source, solide et robuste. Cela respecte tous les critères de GeoNature.

Pour requêter facilement la base de données depuis le langage python il faut utiliser une ORM. Elle permet de transformer une table en un objet facilement manipulable via ses attributs. Pour cela je vais devoir utiliser l'ORM sqlalchemy.

Pour créer une application web, certaines de nos méthodes seront des routes. Chaque route définit une ou plusieurs adresses URL.

Donc, en ce qui concerne le routing je vais utiliser python avec le micro framework Flask. Il est facile d'apprendre à l'utiliser, c'est un framework léger, qui de plus consomme moins de ressources et impose souvent moins de contraintes et le moins de lignes de code possible. Ce micro-framework est déjà utilisé sur TaxHub, GeoNature v2 et GeoNature-Atlas.

Jinja2 est un langage de template moderne et convivial pour Python, modelé sur les templates de Django. Il est rapide, largement utilisé et sécurisé avec l'environnement d'exécution de modèle en sandbox facultatif.

Et enfin, pour colorer et animer légèrement notre application (et oui on ne va pas avoir une application en html pure non plus) on va utiliser Bootstrap (framework css et javascript) pour ajouter du css facilement à notre code et un peu de jQuery (framework javascript) pour ajouter quelques animations.

En ce qui concerne mon environnement de travail lors de ce stage, je travaille sous Ubuntu, j'utilise Visual Studio Code. De plus, je programme sur une branche du dépôt git du PNE de UsersHub sur GitHub.

Pour toutes les applications créées par le PNE et par les parcs nationaux français, GitHub est utilisé en tant que gestionnaire de version, partage et suivi de code.

GitHub est un gestionnaire de code en ligne très utilisé dans le monde de l'open-source. Tout d'abord la gestion de branches est optimale. On peut travailler sur plusieurs projets en parallèle sans conflits. Cela permet aussi aux personnes extérieures au service de tester, collaborer et de travailler en équipe. GitHub est très rapide lors de la mise à jour des données.

Dans mon cas, je travaille sur une branche dédiée à la version 2 de UsersHub. Je développe en local puis je mets à jour mon travail sur le dépôt dédié en continu. Ainsi depuis leur poste de travail mes collègues assistent à l'avancée de mon travail et me corrige en cas de besoin. De plus on peut créer des issues, une issue est un ticket. Les tickets permettent aux personnes participantes et extérieures au projet de proposer des idées, de fixer des problèmes rencontrés.

Pour résumer il y a deux objectifs principaux:

- Refondre l'application pour utiliser le nouveau schéma «Utilisateurs»
- Utiliser Python pour moderniser et harmoniser avec les autres applications du PNE

The screenshot shows the GitHub interface for the repository **PnEcrins / UsersHub**. At the top, there are navigation tabs for **Code**, **Issues** (8), **Pull requests** (1), **Projects** (0), **Wiki**, and **Insights**. Below the repository name, it says "Application web de gestion centralisée des utilisateurs". A summary bar shows **212 commits**, **4 branches**, **9 releases**, and **6 contributors**. Under "Your recently pushed branches", the **uhv2** branch is listed as pushed "less than a minute ago". Below this, there are buttons for "Branch: uhv2", "New pull request", "Create new file", "Upload files", "Find file", and "Clone or download". A status message indicates "This branch is 85 commits ahead, 36 commits behind master." Below this, a table lists the files in the repository:

File	Description	Time
app	ajout des test droit/application	a minute ago
config	V2 - commit initial	a month ago
data	adaptation des tables de la V1 en vue pour la V2	6 days ago
docs	V2 - commit initial	a month ago
.gitignore	transfert visuel users	27 days ago
LICENSE	Ajout license GPLv3	a month ago
README.rst	Correction URL logos	2 years ago
VERSION	V2 - commit initial	a month ago
__init__.py	V2 - commit initial	a month ago
config.py.sample	ajout de param dans le fichier config	14 days ago
install_app.sh	mise à jour de l'installation	7 months ago
install_db.sh	mise à jour de l'installation	7 months ago
requirements.txt	test avec la vue VUsersactionForallGnModules et modif nom fichier js	21 days ago
server.py	changement de la langue de Datatables	5 days ago
README.rst		

Figure 10 : Mon dépôt Git (<https://github.com/PnEcrins/UsersHub/tree/uhv2>)

Déroulement du stage et évolution de l'application

- Mon diagramme de Gantt

	Avril				Mai				Juin	
	Semaine	Semaine	Semaine	Semaine	Semaine	Semaine	Semaine	Semaine	Semaine	Semaine
Découverte des technologies										
Affichage des tables										
Ajout d'un éléments										
Modification d'un éléments										
Suppression d'un éléments										
Etiquetage										
Ajouts de membres										
CRUVED/Droit										
Test/mise en service										
Rapport de stage										

- Mes deux premières semaines de stage, découverte des technologies

Mes deux premières semaines de stages au sein du PNE m'ont permis de découvrir les technologies que j'utiliserais pour mon application web future. A ce moment du stage je ne connais pas encore mon sujet de stage, je sais juste que j'utiliserai le langage python et le micro-framework Flask. Pour cela mon collègue Théo, me partage des tutoriels qu'il a lui même réalisé lors de son arrivée au PNE. Ce n'est pas la première fois que j'utilise python, j'ai déjà réalisé un jeu vidéo en terminal avec l'option Informatique et Sciences du Numérique (ISN). Je m'adapte donc assez facilement au langage même si je suis perturbé par le simple faites qu'il ne faut pas terminer une ligne de code avec un point virgule (réflexe de programmation java).

Les tutoriels sont issues du site OpenClassrooms (site de cours en ligne sur la programmation informatique). J'ai donc appris à l'aide de ces tutoriels à utiliser le micro-framework Flask, l'ORM sqlalchemy. J'ai aussi utilisé sur mes petits projets de test Bootstrap afin de voir le rendu que l'on peut obtenir. Sur mes tests, j'utilise une base donnée de type GeoNature V2, cela me permet donc de me familiariser facilement avec la base et plus particulièrement avec les tables du schéma «Utilisateurs». L'ORM, sqlalchemy me facilite

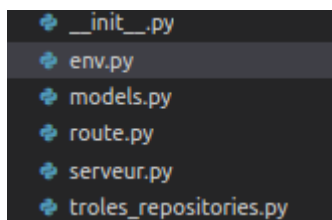
aussi la réalisation de correspondance entre plusieurs tables. A l'issue de ces deux semaines j'ai donc réussi à afficher sur une page web dans un tableau les éléments de la table «t_roles».

localhost:5000/users/list

Liste des Utilisateurs								
#	Id	Identifiant	Nom	Prenom	Description	Email	ID organisme	Remarques
1	20001	None	grp_socle 2	None	Bureau d'étude socle 2	None	None	Groupe à droit étendu
2	20002	None	grp_en_poste	None	Tous les agents en poste au PN	None	None	groupe test
3	20003	None	grp_socle 1	None	Bureau d'étude socle 1	None	None	Groupe à droit limité
4	2	agent	Agent	test	None	None	-1	utilisateur test à modifier ou supprimer
5	3	partenaire	Partenaire	test	None	None	-1	utilisateur test à modifier ou supprimer
6	4	pierre.paul	Paul	Pierre	None	None	-1	utilisateur test à modifier ou supprimer
7	5	validateur	validateur	test	None	None	-1	utilisateur test à modifier ou supprimer
8	1000138	None	test	None	testt	None	None	None
9	1000141	None	sdf	None	<wxcv	None	None	None
10	1000142	OL	Jean-michel	Aulas	rtuyi	test@gmail.com	-1	ertyui
11	1	admin	Administrateur	test	None	None	-1	utilisateur test à modifier

Figure 10: premier affichage de la table t_roles

Il me faut vous expliquer comment à l'aide du langage python et de Flask, on peut obtenir une page internet (bien sûr dans mon exemple je suis en local sur le port 5000).



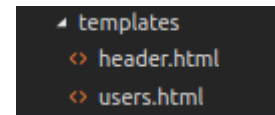
Tout d'abord au niveau de l'architecture du projet, il y a un fichier «__init__.py» qui permet aux autres fichiers de se retrouver entre eux.

Ensuite on a un fichier «env.py» c'est dans ce fichier que l'on appelle l'ORM sqlalchemy. Ainsi si on souhaite utiliser SQLAlchemy, on l'utilise à travers le fichier «env.py» et on ne doit pas l'instancier de nouveau à chaque appel. Le fichier «models.py» est le fichier qui permet de créer les tables du schéma «Utilisateurs» sous forme d'objets python. Le fichier «route.py» est de son côté le cerveau des pages générées. La plupart des méthodes dans ce fichier sont des routes, c'est à dire des méthodes qui possèdent une URL et retournent un template Jinja dans notre cas.

Le «serveur.py» est le fichier qui fait fonctionner le programme, c'est lui que l'on exécute pour lancer l'application web. Dans ce fichier on retrouve la connexion avec la base de données.

Et enfin dans le fichier «troles_repositories.py» j'ai écrit les méthodes que je souhaitais sur des objets «t_roles» par exemples.

Mais pour que cela fonctionne il faut aussi quelques fichiers html dans un dossier templates. Le fichier «header», fait le lien entre le html et le css. Le fichier «users.html» donne un rendu identique à celui de la figure 10.



- Affichage des autres tables du schéma «Utilisateur»

L'objectif maintenant était d'afficher les autres tables contenant des données. Les tables de correspondance n'ont aucun intérêt à être affichées directement, elles auront le pouvoir de créer des relations, on les garde pour plus tard. La mission n'est pas compliquée en soit mais il fallait dupliquer le code html et certaine partie du code python. A l'aide de JinJa2 on peut passer des paramètres au fichier html. J'ai créé un fichier générique afin qu'il s'adapte aux paramètres. J'ai donc implémenté les tables «bib_organismes», «t_tags», et «t_applications».

#	ID	Nom	Adresse	Code_Postal	Ville	Telephone	Fax	Email
1	1	PNF	None	None	Montpellier	None	None	None
2	2	Parc National des Ecrins	Domaine de Charance	05000	GAP	04 92 40 20 10		
3	-1	Autre						
4	0	ALL	Représente tous les organismes	None	None	None	None	None
5	100012	Parc International du Ballon des Voges	Voges	88000	Gérardmer	0339 81 00 00		edfghk_dfgh@gmail.com
6	100013	test validators	fghjklgh					

#	ID	Nom	Description	ID Parent
1	1	application utilisateurs	application permettant d'administrer la présente base de données.	None
2	2	taxhub	application permettant d'administrer la liste des taxons.	None
3	14	application geonature	Application permettant la consultation et la gestion des relevés faune et flore.	None
4	15	contact (Geonature2)	Module contact faune-flore-forge de GeoNature	14
5	16	occtax	None	14
6	60	ghf	xc	16
7	57	Jean michroou	aulas	None

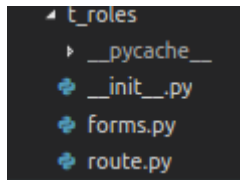
#	ID	ID_type	CODE	Nom	Label	Description
1	1	3	1	utilisateur	utilisateur	Ne peut que consulter
2	2	3	2	redacteur	redacteur	Il possède des droit d'écriture pour créer des enregistrements
3	3	3	3	réfèrent	réfèrent	utilisateur ayant des droits complémentaires au rédacteur (par exemple exporter des données ou autre)
4	4	3	4	modérateur	modérateur	Peu utilisé
5	5	3	5	validateur	validateur	Il valide bien sur
6	6	3	6	administrateur	administrateur	Il a tous les droits
7	11	2	C	create	Create	can create/add new data
8	12	2	R	read	Read	can read data
9	13	2	U	update	Update	can update data
10	14	2	V	validate	Validate	can validate data
11	15	2	E	export	Export	can export data
12	16	2	D	delete	Delete	can delete data
13	100	4	None	observateurs flore	Observateurs flore	liste des observateurs pour les protocoles flore
14	101	4	None	observateurs faune	Observateurs faune	liste des observateurs pour les protocoles faune
15	102	4	None	observateurs aigle	Observateurs aigle	liste des observateurs pour le protocole suivi de la reproduction de l'aigle royal

Figure 11: affichage des tables bib_organismes, t_tags, et t_applications

- Ajout d'un élément

Maintenant que l'on affiche les listes des objets de la base de données, il nous faut pouvoir ajouter un élément. Pour cela j'ai utilisé une librairie de Flask , WTForms. Cette librairie me permet de créer facilement des formulaires.

Ainsi je peux récupérer facilement les données transmises par l'utilisateur au formulaire et ainsi insérer l'élément à la table correspondante.



L'ajout du formulaire influe sur la structure du projet. J'ai créé un dossier pour chaque table de la base de donnée, et à l'intérieur de celui-ci on retrouve un fichier «__init__.py», un fichier «route.py» et enfin un fichier «forms.py» qui contient la classe formulaire appelé ici

«utilisateurs».

Pour afficher un formulaire, il faut un fichier html par table afin d'avoir leur formulaire correspondant. On a donc dans notre dossier templates un fichier header, un fichier qui génère toutes les tables et un fichier de formulaires pour chaque table.

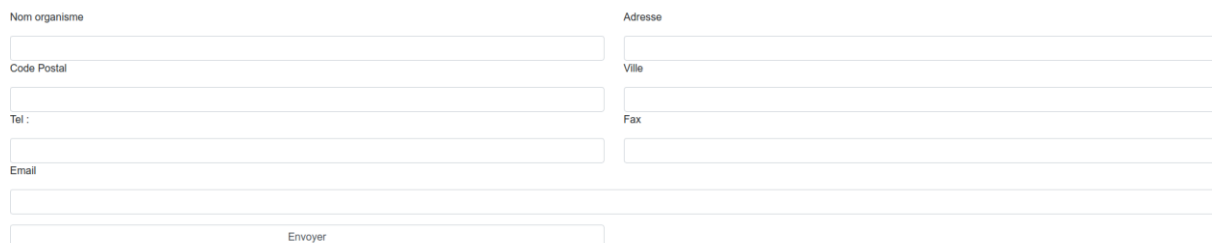


Figure 12: Formulaire pour ajouter un organisme

Il est important de prendre en compte que lors d'un ajout, l'id (donc la clé primaire) s'incrémente automatiquement, il n'y aura donc pas de duplicata dans la base. Afin de ne pas enregistrer des objets vides, l'utilisateur est obligé de remplir la case du nom sinon un message d'erreur s'ouvre sous forme d'alertbox.

Comme on peut le voir sur la figure 12, le formulaire pour ajouter un organisme est simple, il suffit de remplir les cases. Mais par exemple lorsque l'on ajoute un rôle, il faut automatiquement assigner celui-ci à un organisme. Pour cela l'utilisateur, choisit dans un «selectfield» qui propose tous les organismes référencés dans la base de données.

Choix Organisme

Autre

Figure 13: «selectfield» de sélection d'un organisme lors d'un ajout de rôles

Il existe aussi des champs «mot de passe», ces champs existent dans l'ajout d'un rôle, car le rôle nécessiterait un mot de passe pour se connecter aux applications. A de l'inscription du nouvel objet rôle dans la base de données, je crypte le mot de passe à l'aide de la librairie python Bcrypt. Ainsi je ne stocke pas de mot de passe en clair dans la base de données.

Et enfin j'ai utilisé un champ «MultipleSelectfield» pour permettre à l'utilisateur de définir directement dans quels groupes le rôle appartient lors de l'ajout.

Pour faire le lien entre l'ajout d'un élément et la liste de ceux-ci j'ai ajouté un bouton qui le permet au dessus du tableau de ceux-ci.

Listes des Organismes

Ajouter un organisme

#	ID	Nom	Adresse	Code_Postal	Ville	Telephone	Fax	Email
1	1	PNF	None	None	Montpellier	None	None	None

Figure 14 : Ajout du bouton permettant le lien entre la page d'ajout et la liste de l'élément

Une fois le formulaire «Enregistrer», le site redirige vers la liste de l'éléments, on peut donc constater que l'ajout s'est réalisé correctement.

- Séparation utilisateurs/groupes

Comme évoqué précédemment la base de données est faite de la façon suivante : dans la table «t_roles», une colonne définit si le rôle est un utilisateur ou un groupe. Si la valeur est vrai le rôle est un groupe et inversement. Ainsi, j'ai décidé de séparer l'affichage des groupes et des utilisateurs. J'ai maintenant deux pages, une pour la liste d'utilisateurs et une pour la liste de groupes.

- Modification d'un élément

Bien sûr, si on peut ajouter un élément, il faut pouvoir aussi le modifier. En m'inspirant de l'application Taxhub, j'ai décidé d'ajouter un bouton modification à chaque élément lorsque l'on est sur les pages qui listent ceux-ci.

Voici un exemple:

Listes des Organismes

Ajouter un organisme

#	ID	Nom	Adresse	Code_Postal	Ville	Telephone	Fax	Email	Modification
1	1	PNF	None	None	Montpellier	None	None	None	Modification
2	2	Parc National des Ecrins	Domaine de Charance	05000	GAP	04 92 40 20 10			Modification
3	-1	Autre							Modification
4	0	ALL	Représente tous les organismes	None	None	None	None	None	Modification
5	1000012	Parc International du Ballon des Vosges	Vosges	88000	Gérardmer	sdfghj	sdfghjk	dfghj@gmail.com	Modification
6	1000013	test validateurs	fghjkd f dgh						Modification

Figure 15 : liste des organismes avec l'ajout du bouton modification pour chaque élément

Ensuite quand on clique sur le bouton modification, le site nous redirige vers le formulaire (voir figure 12) mais à contrario de l'ajout, celui est pré-rempli. Le remplissage s'effectue à l'aide de la récupération de l'id de l'élément sélectionné. Ainsi on obtient ce formulaire si on clique sur le bouton modification de la ligne 2 de la figure 15.

Nom organisme	Adresse
Parc National des Ecrins	Domaine de Charance
Code Postal	Ville
05000	GAP
Tel :	Fax
04 92 40 20 10	
Email	
Envoyer	

Figure 16: formulaire pré-rempli de l'organisme PNE

- Suppression

Qui dit ajout et modification, dit suppression. C'est pourquoi il faut aussi songer à pouvoir supprimer un élément de la base de données. Pour cela, on reprend le même principe que pour la modification, on ajoute un bouton suppression pour chaque élément. Ainsi on ne peut pas se tromper sur quel élément on souhaite supprimer. De plus afin d'assurer la suppression d'un bon élément un pop-up d'alerte apparaît pour confirmer celle-ci.

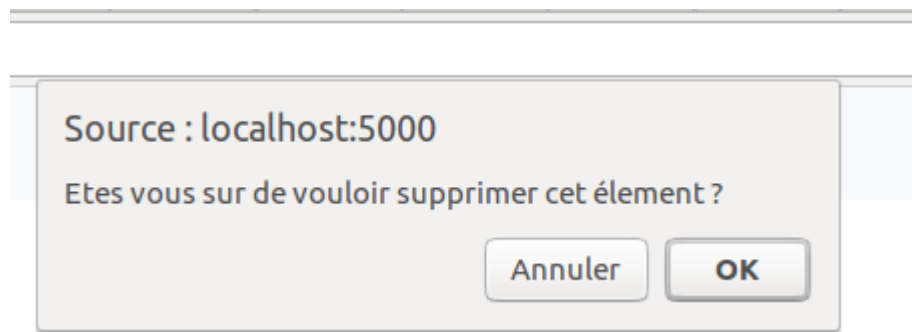


Figure 17: aperçu de l'alert-box de suppression

- Récapitulatif

Actuellement mon application me permet :

- Afficher les tables :
 - «bib_organismes»
 - «t_roles» (utilisateurs et groupes séparés)
 - «t_applications»
 - «t_tags»
- Ajouter un élément dans ces tables
- Modifier un élément dans ces tables
- Supprimer un élément dans ces tables

Afin de lier ces pages, j'ai utilisé une navbar, comme sur l'application TaxHub.

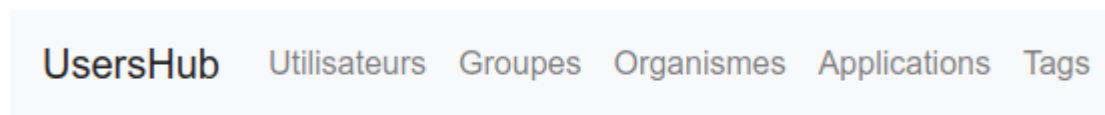


Figure 18: navbar liant, les utilisateurs, les groupes, les organismes, les applications et les tags

- Ajouts de membre

Maintenant l'application doit nous permettre d'ajouter des membres à des groupes. Pour cela on doit créer une nouvelle interface. Pour accéder à cette interface on ajoute de nouveau une colonne à la liste de groupes, de la même façon que le bouton modification ou suppression.

UsersHub Utilisateurs Groupes Organismes Applications Tags CRUVED

Liste des Groupes

Ajouter un groupe

#	ID groupe	nom	description	Membres	Modification	Suppression
1	20001	grp_socle 2	Bureau d'étude socle 2	Membres	Modification	Supprimer
2	20002	grp_en_poste	Tous les agents en poste au PN	Membres	Modification	Supprimer
3	20003	grp_socle 1	Bureau d'étude socle 1	Membres	Modification	Supprimer
4	1000138	test	testt	Membres	Modification	Supprimer
5	1000141	sdf	<wxcv	Membres	Modification	Supprimer

Figure 19 : ajout de la colonne membres à l'interface groupe

La nouvelle interface est simple à comprendre. Elle est séparée en deux parties, d'un côté les rôles (groupes et utilisateurs) n'appartenant pas au groupe sélectionné, de l'autre les rôles appartenant déjà au groupe. Un groupe peut appartenir à un autre groupe. On reconnaît un groupe à la couleur de la ligne. C'est ici qu'intervient pour la première fois du javascript dans l'application, il permet d'ajouter ou supprimer des éléments à un groupe. Pour cela il suffit de cocher les lignes souhaitées et ensuite de déplacer d'un côté ou de l'autre à l'aide des flèches. La transaction s'enregistre dans la base de données qu'une fois le bouton «Enregistrer» activé.

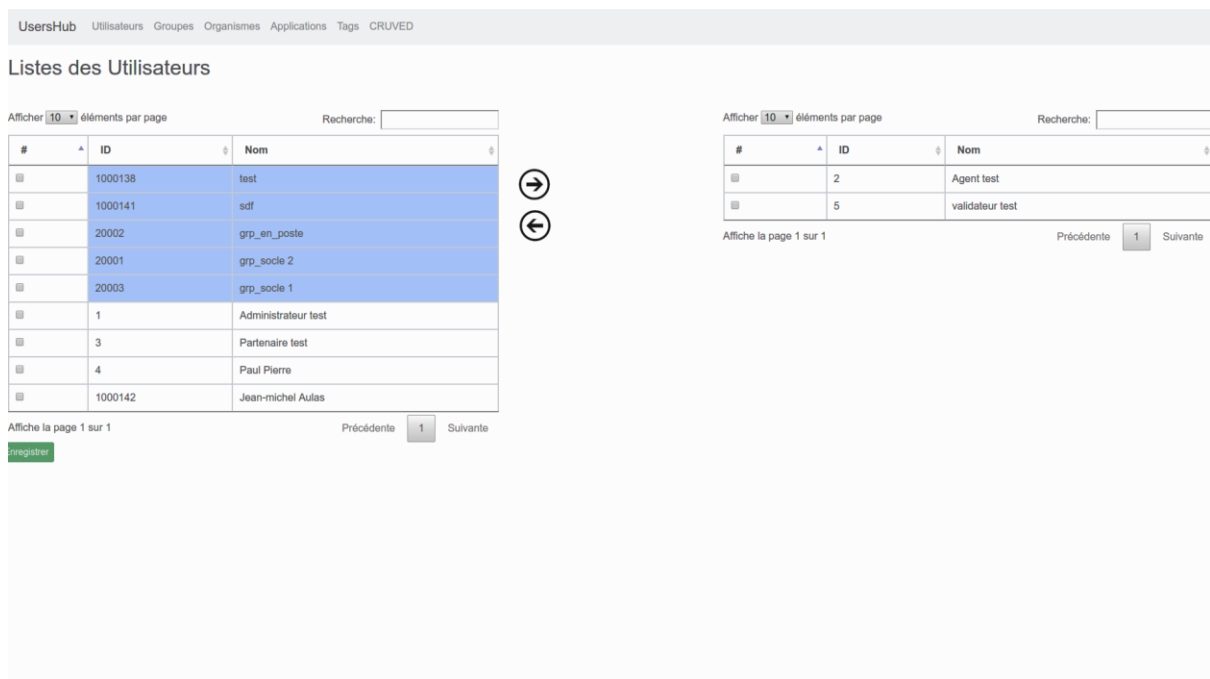


Figure 20 : interface de transfert des rôles vis à vis d'un groupe

Sur cette interface, j'ai implémenté une librairie javascript appelée DataTable, celle-ci me permet de réaliser un tri de mes tableaux par colonnes, de rechercher un élément, de choisir combien d'éléments je souhaite afficher, le nombre de page de ce tableau.

Cette interface est utilisable par plusieurs tables, on a donc réfléchi lors de son écriture à réaliser un fichier générique pour réutiliser la même structure.

- Étiquetage d'un rôle, d'une application ou d'un organisme

Cette interface fonctionne de la même façon que la précédente.

Mais tout d'abord à quel type de tag pouvons-nous donner cette possibilité d'étiquetage ?

Cela est possible si le type de tag est une liste. Ainsi, par exemple, nous pouvons définir que l'utilisateur «jean-michel» est étiqueté par le tag observateurs-faune. Les tags nous permettent donc de définir un rôle, une application ou un organisme de manière très précise.

On choisit quel objet on souhaite taguer depuis la liste de tags, et ensuite on obtient une interface, identique à l'interface d'ajout des rôles à des groupes.

UsersHub Utilisateurs Groupes Organismes Applications Tags CRUVED

Liste des Tags

Ajouter un tag Accéder aux types de tags

#	ID	ID_type	CODE	Nom	Label	Description	Utilisateurs	Organismes	Application	Modification	Suppression
1	1	3	1	utilisateur	utilisateur	Ne peut que consulter				Modification	Supprimer
2	2	3	2	rédacteur	rédacteur	Il possède des droit d'écriture pour créer des enregistrements				Modification	Supprimer
3	3	3	3	réfèrent	réfèrent	utilisateur ayant des droits complémentaires au rédacteur (par exemple exporter des données ou autre)				Modification	Supprimer
4	4	3	4	modérateur	modérateur	Peu utilisé				Modification	Supprimer
5	5	3	5	validateur	validateur	Il valide bien sur				Modification	Supprimer
6	6	3	6	administrateur	administrateur	Il a tous les droits				Modification	Supprimer
7	11	2	C	create	Create	can create/add new data				Modification	Supprimer
8	12	2	R	read	Read	can read data				Modification	Supprimer
9	13	2	U	update	Update	can update data				Modification	Supprimer
10	14	2	V	validate	Validate	can validate data				Modification	Supprimer
11	15	2	E	export	Export	can export data				Modification	Supprimer
12	16	2	D	delete	Delete	can delete data				Modification	Supprimer
13	100	4	None	observateurs flore	Observateurs flore	liste des observateurs pour les protocoles flore	Utilisateurs	Organismes	Application	Modification	Supprimer
14	101	4	None	observateurs faune	Observateurs faune	liste des observateurs pour les protocoles faune	Utilisateurs	Organismes	Application	Modification	Supprimer
15	102	4	None	observateurs aigle	Observateurs aigle	liste des observateurs pour le protocole suivi de la reproduction de l'aigle royal	Utilisateurs	Organismes	Application	Modification	Supprimer

Figure 21 : Dans la liste de tag on retrouve les tags qui permettent d'étiqueter, des rôles, des organismes, des applications.

- Type de tag

Si l'utilisateur de l'application souhaite accéder à la liste de type de tag ou ajouter un type de tag à la base de données, il peut accéder à une interface de liste de type de tag. Pour se faire il doit cliquer sur le bouton «Accéder au types de tags».

De ce fait, le site le redirige vers la liste de type de tags. Sur cette interface, il peut modifier, supprimer ou ajouter un nouveaux type de tag.

- CRUVED

Dans les objectifs du stage, je vous ai présenté le CRUVED. Imaginer graphiquement une interface facile et programmable n'était pas le plus simple au vu de la complexité du CRUVED. De plus, l'objectif du site est de ne pas réaliser des interfaces complètes en javascript. L'idée retenue est la suivante : l'interface est séparé en deux parties d'un côté un tableau contenant les rôles (groupes et utilisateurs, les lignes des groupes sont colorés). De l'autre côté, un tableau affichant leur CRUVED pour leur(s) application(s).

Pour afficher le CRUVED d'un rôle, il suffit de cliquer sur le bouton «Voir CRUVED» et ainsi le tableau de droite sera mis automatiquement à jour.

Un rôle peut ne pas avoir de CRUVED établi mais si il appartient à un groupe, il possède le CRUVED du groupe.

UsersHub Utilisateurs Groupes Organismes Applications Tags CRUVED							
Liste d'Utilisateurs et de Groupes							
ID	Nom	Cruved					
20003	grp_socle 1	Voir CRUVED					
1000138	test	Voir CRUVED					
1000141	sdf	Voir CRUVED					
20002	grp_en_poste	Voir CRUVED					
20001	grp_socle 2	Voir CRUVED					
1000142	Jean-michel Aulas	Voir CRUVED					
1	Administrateur test	Voir CRUVED					
2	Agent test	Voir CRUVED					
3	Partenaire test	Voir CRUVED					
4	Paul Pierre	Voir CRUVED					
5	validateur test	Voir CRUVED					

grp_en_poste							
Application	Create	Read	Update	Validate	Export	Delete	
application geonature	3	2	1	0	2	1	Editer
Ajouter un Cruved pour une application							

Figure 21 : Interface illustrant le CRUVED du rôle grp_en_poste sur l'application GeoNature

Pour accéder à cette interface, j'ai rajouté un lien dans la navbar de la figure 18.

Si l'utilisateur souhaite ajouter un CRUVED à un rôle pour une application, il clique sur le bouton «Ajouter un CRUVED pour une application» et il est redirigé vers un formulaire de CRUVED.

Celui-ci contient six «selectfield» qui représente les six actions et qui permettent de choisir la portée pour chacune des actions. A l'ajout les champs affichent une portée aucune, qui est une portée par défaut si aucun CRUVED n'est établie.

Mais si on accède à l'interface depuis la modification du CRUVED, alors les champs seront pré-remplis.

UsersHub Utilisateurs Groupes Organismes Applications Tags CRUVED	
Vom	Application
grp_socle 1	application geonature
Portee Create	Portee Read
all data	my organism data
Portee Update	Portee Validate
all data	Aucun
Portee Export	Portee Delete
my organism data	all data
Enregistrer	

Figure 23: Formulaire du CRUVED de grp_socle_1 sur l'application GeoNature.

Une fois le formulaire «Enregistrer», le site redirige vers la liste de CRUVED du rôle sur lequel il s'est effectué.

Pour rappel, le CRUVED n'est définissable que si l'application est GeoNature ou fille de GeoNature. Sinon il faut utiliser l'ancienne méthode d'ajout de droit à un rôle pour une application.

- Droits application

C'est pourquoi, il faut réfléchir à une autre interface qui nous permet d'ajouter de simple droit à un rôle pour une application sans passer par le CRUVED.

Pour accéder à cette interface, j'ai ajouté une colonne à la liste d'application, colonne que j'ai appelé «Utilisateurs».

On y retrouve un bouton à chaque ligne pour y accéder. Si l'application utilise le CRUVED, alors le bouton n'existe pas.

Listes des Applications

Ajouter une application

#	ID	Nom	Description	ID Parent	Utilisateurs	Modification	Suppression
1	1	application utilisateurs	application permettant d'administrer la présente base de données.	None	Utilisateurs	Modification	Supprimer
2	2	taxhub	application permettant d'administrer la liste des taxons.	None	Utilisateurs	Modification	Supprimer
3	14	application geonature	Application permettant la consultation et la gestion des relevés faune et flore.	None		Modification	Supprimer
4	15	contact (GeoNature2)	Module contact faune-flore-fonge de GeoNature	14		Modification	Supprimer
5	16	occtax	None	14		Modification	Supprimer
6	60	ghjkl	xc	16	Utilisateurs	Modification	Supprimer
7	57	jean michoouu	aulas	None	Utilisateurs	Modification	Supprimer

Figure 24: Liste des applications, avec la colonne utilisateurs

On va donc se baser sur l'interface de transfert du type rôle/groupe (figure 20) afin d'ajouter ou supprimer un rôle à une application.

Mais, il faut pouvoir choisir un niveau de droit pour ce rôle. J'ai décidé d'ajouter une colonne dans le tableau des rôles ayant des droits sur une application.

Cette colonne s'appelle «droit», elle ajoute à chaque ligne un «selectfield». Ce selectfield permet de donner un niveau de droit au rôle concerné. Les niveaux de droits sont

piochés dans la base de données dans la table «t_tags» dont le type de tag correspond à privilège.

Cette interface offre la possibilité d'ajouter ou supprimer un rôle et de lui assigner un niveau de droits.

Listes des Utilisateurs

Afficher 10 éléments par page

Recherche:

#	ID	Nom
<input type="checkbox"/>	20002	grp_en_poste
<input type="checkbox"/>	20003	grp_socle 1
<input type="checkbox"/>	1000138	test
<input type="checkbox"/>	1000141	sdf
<input type="checkbox"/>	20001	grp_socle 2
<input type="checkbox"/>	1000142	Jean-michel Aulas
<input type="checkbox"/>	2	Agent test
<input type="checkbox"/>	3	Partenaire test
<input type="checkbox"/>	4	Paul Pierre
<input type="checkbox"/>	5	validateur test

Affiche la page 1 sur 1

Précédente

1

Suivante

Enregistrer

Afficher 10 éléments par page

Recherche:

#	ID	Nom	Droit
<input type="checkbox"/>	1	Administrateur test	<input type="text" value="s"/>

Affiche la page 1 sur 1

Précédente

1

Suivante

Figure 25 : Interface de transfert et d'ajout de droit pour un rôle sur une application

Les ajouts et suppressions s'enregistrent dans la base de données que lorsque l'on clique sur le bouton «Enregistrer».

Difficultés rencontrées

Afin de bien comprendre le sujet du stage, il faut bien connaître son contexte, comment les autres applications fonctionnent entre elles par exemple. Il faut aussi bien comprendre le système de CRUVED et de tag, et ceci n'est pas simple au premier abord. Une fois le système bien compris on peut facilement dérouler la suite.

Plusieurs fois durant le stage mes collègues dont Théo et Camille étaient absents en même temps pour raisons diverses. J'ai donc dû me débrouiller par moi-même, chercher les informations qu'il me fallait. J'ai gagné en autonomie mais même si le projet avançait plus lentement, j'ai réussi à trouver des solutions par moi-même.

Technologiquement, la librairie WTForms m'a causé plusieurs fois quelques problèmes. Il faut savoir que, un formulaire pour un élément est une classe en python. L'envoi du formulaire, se fait sous la forme d'un dictionnaire, lorsque l'on appuie sur le bouton «Enregistrer». Si je souhaitais avoir plusieurs boutons de validation était impossible. J'ai donc simplifié mon affichage graphique pour ne plus avoir cette erreur. De plus certaines valeurs non connues étaient retournées dans le dictionnaire. Cela m'a empêché plusieurs fois d'ajouter un élément à la Base de données. Je supprime dorénavant les valeurs cachées avant l'insertion de l'élément dans la base de données.

L'objectif du sujet était de refondre UsersHub afin de fonctionner avec le nouveau schéma «Utilisateurs», mais au fur et à mesure de l'avancement du projet, j'ai relevé certaines erreurs dans la base de données. J'ai donc corrigé certaines vues afin de permettre la compatibilité de UsersHub v2 avec les applications qui n'utilisent pas le CRUVED et qui ont été développées sur la base de UsersHub v1.

Conclusion

L'application répond aujourd'hui en grande partie aux attentes fixées en début de stage, il me reste deux semaines après la soutenance pour la peaufiner.

Au niveau du contenu, l'application respecte la demande initial formulée par mon maître de stage : un administrateur peut ajouter un rôle, un organisme, une application, un tag, un type de tag, les modifier et les supprimer. Il peut assigner des droits, CRUVED ou non, ajouter des rôles à des groupes, étiquetés des rôles, des applications, des organismes.

D'un point de vue personnel, ce stage m'a permis de développer de nombreuses compétences dont des nouvelles en développement web et me familiariser un peu plus avec le langage python. J'ai énormément appris, autant sur la partie architecture et développement web, que sur la partie base de données que j'ai pu découvrir pour implémenter cette interface web. J'ai découvert l'utilisation de python pour réaliser une application web, et vite compris son énorme potentiel de création.

J'ai également pu faire quelque sortie de terrain et au regard ma mission et du sujet de stage, je comprend mieux l'intérêt des applications développées au sein du PNE.

J'ai eu la chance de m'investir dans un parc national Alpin, de rencontrer des collègues enrichissants tels que Camille, Gil, Vincent, Théo et encore beaucoup d'autres collègues. Ce fût pour moi une expérience très instructives de m'insérer au sein d'une équipe. J'ai ainsi pu appréhender les réalités du monde du travail.

Cette expérience va être un atout pour la poursuite des mes études et pour mon cursus professionnel.

Annexes

Schéma Utilisateurs V1

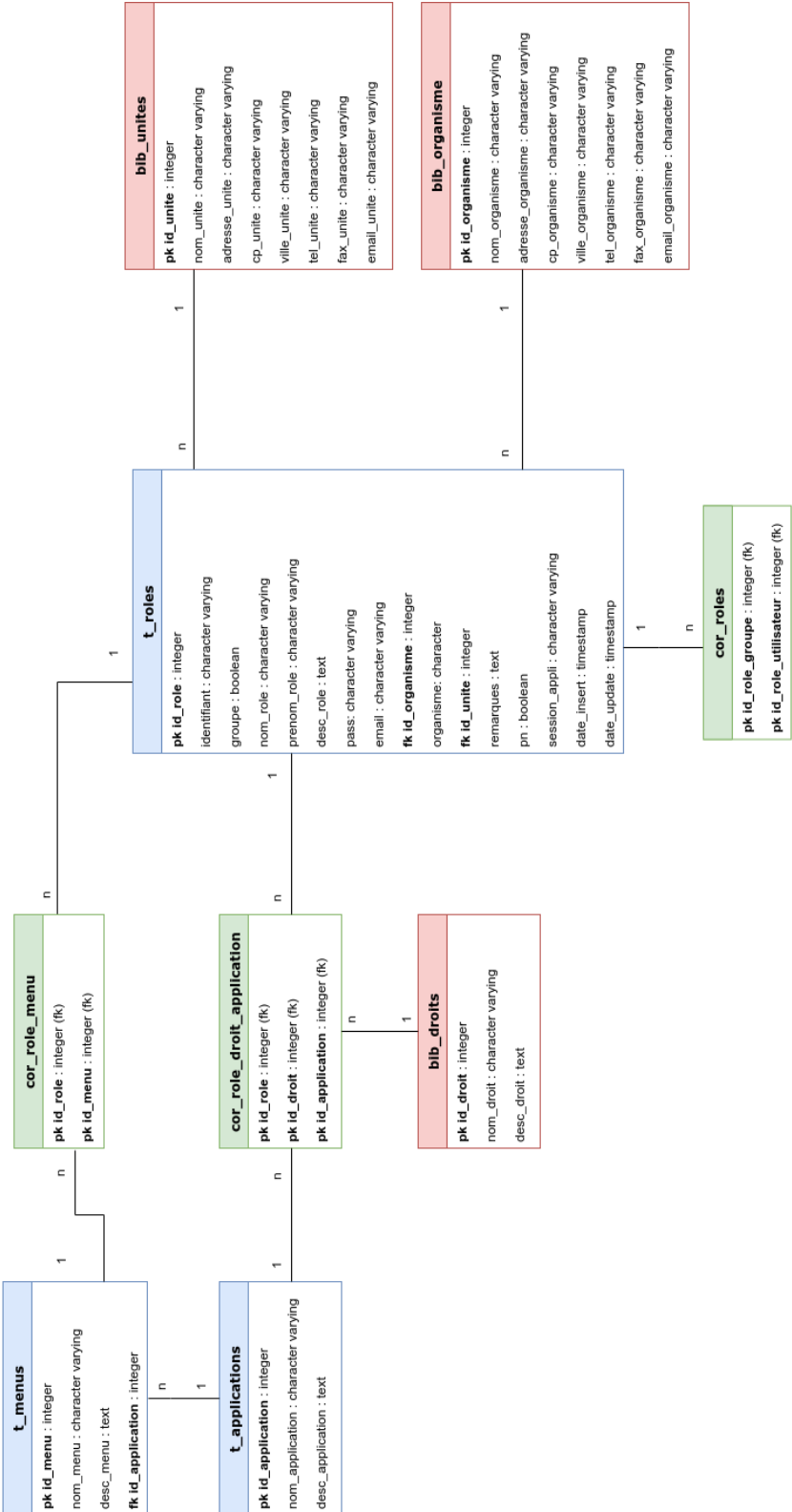
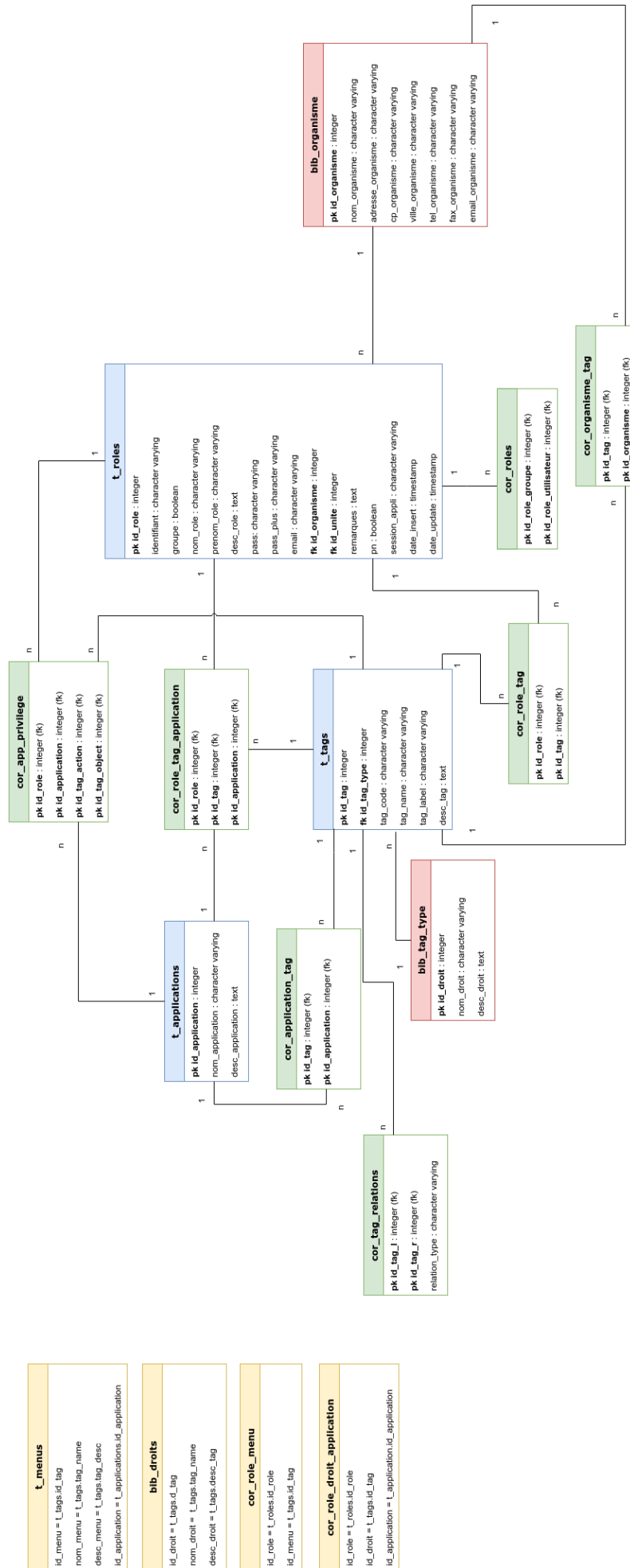


Schéma Utilisateurs V2



FICHE RAPPORT DESTINEE A LA BIBLIOTHEQUE

RAPPORT CONFIDENTIEL ET NE DEVANT PAS FIGURER A LA BIBLIOTHEQUE :

non

NOM ET PRENOM DE L'ETUDIANT :

Laumond Gabin

DUT : INFORMATIQUE

S4

TITRE DU RAPPORT :

Refonte d'une application web de gestion centralisée des utilisateurs

Nom de l'Entreprise :

Parc national des Ecrins

Adresse :

Domaine de Charance, 05000 Gap

Type d'activité (domaines couverts par l'entreprise) :

Protection de la biodiversité et de l'environnement

Nom du parrain (enseignant IUT) :

Fanny Binet

Mots-clés (sujets traités) :

Python, application web, gestionnaire d'utilisateurs, parc national, GeoNature, open-source

Résumé

Refonte d'une application web de gestion centralisé des utilisateurs, utilisé par les applications des parcs nationaux français notamment pour l'application GeoNature. Stage réalisé au sein du pôle Système d'informations du parc national des Ecrins à Gap.