Rapport de stage

David CHOCHO

IUT De Nantes

15 Avril 2024 — 21 Juin 2024

Maîtresse de stage : AVRILLON Gaëlle

Référente pédagogique : TAMZALIT Dalila

Lieu du stage : NAOMIS, 4 rue Rene Viviani 44200 NANTES

Remerciements

Je tiens tout d'abord à remercier l'IUT de Nantes sans qui rien de tout cela n'aurait été possible et Mme. TAMZALIT Dalila pour son aide et ses conseils sur l'écriture de ce rapport de stage.

Ensuite, je voudrais remercier chaleureusement M. DALICHAMPT Christophe, Directeur Général de NAOMIS, qui m'a donné la chance de pouvoir intégrer sa boite en tant que stagiaire.

Je tiens également à remercier Mme. AVRILLON Gaëlle, ma maîtresse de stage, cheffe du pôle informatique chez NAOMIS, et qui m'a accueilli comme stagiaire dans son service.

Merci aussi à M. CERISIER Fabien pour son aide qui fut souvent précieuse à plusieurs reprises durant ce stage.

Et enfin, un grand merci à tous ceux que je n'ai pas cité précedemment pour votre bonne humeur qui contribue tellement à faire de NAOMIS une société où il fait bon vivre.

*PS : Tous les mots stylisées comme **ceci** sont définis dans le glossaire (et clickables si votre lecteur PDF le permet)*

Résumé Technique

Français

Dans le cadre de mon stage de 12 semaines chez NAOMIS[6], j'ai eu 4 missions différentes de développement de scripts **Python**[11] :

L'écriture d'un script permettant la récupération des données dans des CV Word internes à l'entreprise pour lequel j'ai du utiliser des **RegEx**, de la théorie algorithmique et de la conversion de fichiers ;

La réalisation d'un programme permettant de simuler des courses de bateaux en temps réels et en 'replay' en suivant un parcours défini transmit au format **WKT** et permettant l'envoi régulier des positions des bateaux à une **base de données** et une **API** ;

La création d'une solution permettant de synchroniser une base de données **PostgreSQL** avec l'équivalent de cette base sous la forme d'un ensemble de **geopackages** représentant chacun une table de cette base ;

Et enfin l'accompagnement d'un autre stagiaire de la boite dans sa propre mission d'écriture de script python consistant à extraire des données de fichiers excel afin de les traiter et de les intégrer de manière formalisées dans une base de données PostgreSQL.

Anglais

As part of my 12-week internship at NAOMIS, I had 4 different Python script development assignments :

Writing a script to retrieve data from the company's internal Word CVs, for which I had to use RegEx, algorithmic theory and file conversion ;

The creation of a program to simulate boat races in real time and in replay following a defined path transmitted in WKT format, and sending boat positions regularly to a database and an API ;

The creation of a solution to synchronize a PostgreSQL database with the equivalent of this database in the form of a set of geopackages, each representing a table in this database ;

And last, the support for another of the company's trainees in his own mission to write a python script extracting data from excel files in order to process and integrate them in a formalized way into a PostgreSQL database.

Table des matières

Première partie

Contexte

Lorsque j'ai commencé à rechercher un stage mes critères étaient tout d'abord :

- la taille de l'équipe (priorité aux petites entreprises)
- le domaine des missions (priorité à **Kotlin**/**Java**, puis à Python, puis à **PHP** avec l'administration réseaux ou en Base de données, mais à éviter)
- le lieu (à proximité de mon logement)

Par l'intermédiaire d'un proche, j'ai découvert NAOMIS qui correspondait parfaitement à tous mes critères.

Suite à mon entretien de sélection, j'ai été informé que mes missions seraient l'écriture de divers scripts Python qui serviraient dans le cadre de projets réels, ce qui me satisfaisait totalement. Et même en dehors des missions, le cadre était parfait, l'environnement de travail excellent et les employés sympathiques.

La perspective de travailler sur des projets concrets et de voir mon travail réellement utilisé m'enchantait grandement. Cependant, malgré ma grande confiance en mes capacités de Développement, je m'interrogeais sur la qualité des prestations que je pouvais fournir à cause de l'absence de référentiel sur lequel me baser, et sur les compétences attendues par le pôle développement.

Deuxième partie

L'entreprise

1 Présentation de l'entreprise

Le groupe KERAN

KERAN[5] est un groupe spécialisé dans l'aménagement durable et éco responsable des territoires. Il travaille souvent avec les municipalités (conception du pont Anne de Bretagne à Nantes notamment). Anciennement Groupe SCE, KERAN n'a cessé de grandir depuis sa création en 2003 jusqu'à atteindre plus de 600 collaborateurs répartis en 6 filiales :

- **SCE**[8], la société d'origine, spécialisée dans l'aménagement des territoires
- **Groupe Huit**[4], spécialisé dans le développement urbain des pays du sud
- **Créocéan**[3], spécialisé dans l'aménagement des océans et des territoires littoraux notamment
- **S3D**[7] qui est un bureau centré sur la valorisation des déchets et la recherche de carburants alternatifs
- **YS EMD**[9] qui est lié à l'énergie marine (projets hydroliens et houlomoteurs)
- Et enfin **NAOMIS** qui est la filiale numérique de KERAN et qui s'occupe de développer tous les outils internes au groupe en plus de travailler sur des projets avec le secteur public et/ou privé.

C'est dans cette dernière filiale que j'effectue mon stage.

La filiale NAOMIS

NAOMIS possède 3 domaines d'activités :

- **La transformation numérique**, c'est-à-dire trouver des solutions numériques pour l'aménagement durable des territoires, améliorer les services fournis par les organisations et les territoires et le développement de nouveaux outils et démarches pour eux
- **La (géo) data**, qui est le développement de méthodes et outils afin que les bonnes données soient facilement accessibles et délivrées aux bonnes personnes et dans les bons outils
- **Les services (géo) numériques**, qui représentent la conception de solutions centrées sur l'utilisateur en fournissant des services de maintenance des solutions sur la durée

Les valeurs de l'entreprise

5 valeurs sont communes à toutes les filiales du groupe KERAN :

- **Proximité** avec l'utilisateur final car sa satisfaction est l'objectif ultime de tout projet
- **Audace** dans l'usage des technologies pour toujours pouvoir s'adapter aux dernières évolutions du domaine et donc pouvoir proposer les meilleurs outils au meilleur moment
- **Confiance** des clients due à la maîtrise des technologies numériques associée à une expertise métier en aménagement des territoires
- **Talents** des collaborateurs qui voient leurs expertises et connaissances exploitées et développées
- **Engagement** sur la performance globale des projets ainsi que sur un numérique responsable

2 Informations pratiques & Quelques chiffres clés

NAOMIS à été fondée par le groupe KERAN en 2010. Elle possède à ce jour ≈25 employés et a son siège social au 4 rue Rene Viviani 44200 NANTES. En 2022, elle a fait un chiffre d'affaires de 2.1 millions d'euros.

On peut citer quelques projets significatifs de NAOMIS :

- Accompagnement de la Métropole Rouen Normandie (76) dans la conception d'une solution cartographique en appui de son Projet Alimentaire Territorial et dans la consultation pour sélectionner un prestataire.
- Cartographie des données et des systèmes et applicatifs liés dans un projet d'urbanisation et de mise en qualité en préparation du Maas de Nantes Métropole (44)
- Intégration des données du cadastre napoléonien de la Région Pays de la Loire afin de bénéficier de données SIG d'analyse historique
- Développement d'une solution numérique pour référencer les points d'eau en Polynésie pour le BRGM.

3 Présentation des équipes

Les équipes de NAOMIS sont découpés en deux gros 'pôles' en plus d'autres postes :

- Le pôle **Data**, dirigé par FAYAUD Guillaume, chargé de la gestion des bases et transferts de données au sein des différents projets

- Le pôle **Dev**, dirigé par AVRILLON Gaëlle, chargé du développement des applications et la réalisation des différents projets.
- La **Direction**, constitué de DALICHAMPT Christophe (directeur général), BLANCHET Sophie (directrice générale adjointe), MICHAUD Sarah (Assistante) et FEVRE Valentine (chargée de communication)
- Il existe également quelques postes auxiliaires qui ne font partis d'aucun pôle comme le poste de CHALMEL Vincent qui s'occupe de l'intelligence artificielle au sein de NAOMIS

voir annexe 3.1: Organigramme de NAOMIS, pôle Direction et postes auxiliaires

voir annexe 3.2: Organigramme de NAOMIS, pôle Développement

voir annexe 3.3: Organigramme de NAOMIS, pôle Data

4 L'environnement de travail

NAOMIS est située en plein coeur de l'île de Nantes à 2 minutes de l'arrêt de busway "Ile de Nantes". Elle occupe également un grand bâtiment qu'elle partage avec tous les autres membres du groupe KERAN, lui conférant donc certains avantages structurels comme une cafétéria d'entreprise, un local vélo et un parking entre autre.

Mais si on ne parle que des locaux de NAOMIS, on peut les décrire comme un lieu convivial et agréable avec une machine à café à disposition, quelques plantes et où est mis à disposition une caisse de fruits toutes les 2 semaines pour les équipes.

Pour ce qui est de mon poste de travail, j'ai été placé à un bureau possédant deux écrans avec un ordinateur portable à ma disposition.

Sur cet ordinateur tournant sur Windows 11, je n'ai installé que très peu de choses :

- **VsCode** 1.90.0, que j'ai utilisé comme IDE pour l'ensemble de mes projets chez NAOMIS
- Python 3.11.9, que j'ai aussi utilisé pour la totalité de mes projets chez NAOMIS
- **Docker** 4.29.0, qui m'a servi notamment pour faire tourner des bases de données localement sans avoir à installer leurs moteurs directement sur mon pc
- **Git**, pour interagir avec les repertoires **GitHub** de NAOMIS

Troisième partie

Présentation des missions

1 Préambule

Au total, j'ai eu 4 missions chez NAOMIS. Elles sont toutes sur des sujets totalement différents et requièrent des compétences et technologies totalement différentes. Le seul point commun de ces 4 missions à été l'utilisation intégrale de python pour l'écriture des scripts. Je vais donc présenter ces 4 missions dans l'ordre chronologique de début de leur mise en oeuvre permettant ainsi d'avoir un aperçu du déroulé de mon stage.

2 Gestion de CV

Contexte

Au sein du groupe KERAN, une banque de CV est constituée et régulièrement mise à jour dans chaque filiale. Ces CV sont utilisées dans le cadre des projets pour présenter les intervenants au client. Pour ce faire, chaque société a fourni un modèle de CV à respecter pour normer la structure de ceux-ci. Chaque employé peut avoir 1 ou plusieurs CV selon les circonstances.

Ceci dit, SCE une des sociétés du groupe KERAN, se développe rapidement avec actuellement plus de 700 employés. De ce fait, le nombre de CV internes grandit alors que le modèle de CV n'ai pas toujours bien respecté. La gestion des CV devient donc de plus en plus problématique pour l'entreprise.

Pour pallier ce problème, NAOMIS a proposé le développement d'une application web qui permettrait de générer automatiquement des CV au bon format, à partir de données que l'utilisateur aurait juste à fournir. Cette application est censée permettre de mieux gérer ses propres CV pour les employés, et d'avoir une vue d'ensemble pour les managers.

Néanmoins, demander à chacun des 700 employés de refaire leurs CV sur l'application une fois qu'elle sera finalisée est une solution qui n'a pas été retenue. Le plan était donc plutôt d'introduire les données des CV déjà faits dans une base de données pour les rendre directement disponibles au lancement de l'application. Telle était ma mission.

Objectif

L'objectif de cette mission est donc d'écrire un script python permettant de récupérer les informations dans une liste de fichiers Word avec un modèle normalisé en tenant compte des variations dans le modèle. Il devra aussi permettre de détecter et de signaler les données qui n'ont pas été correctement récupérées (au moyen de logs) afin d'alimenter directement une base de données PostgreSQL.

3 Simulation de courses de bateaux

Contexte

M.DALICHAMPT, président de NAOMIS, est également président du club de voile de Saint-Gilles-Croix-de-Vie et membre du conseil d'administration de Team Vendée Formation.

Pour la sardinha cup 2022[13] (une course de reliant la France au Portugal) NAOMIS avait développé une application permettant de suivre la course de bateaux en temps réel. De même qu'une application mobile qui proposait un itinéraire terrestre qui suivait en partie celui des bateaux et qui permettait de compter les pas.

Pour le Défi Des Villes[2] à Saint-Gilles-Croix-de-Vie initialement prévu le 6 & 7 juin, l'application mobile avait la possibilité d'être relancée après 3 ans d'inactivité. Mais pour cet événement, il n'y avait pas de course de bateaux de prévue, malgré que l'application mobile en était dépendante.

Objectif

Cette mission consiste donc à développer un script Python permettant de simuler des courses de bateaux ou plus précisément le comportement des balises présentes sur ces-dits bateaux durant ces courses. Le script doit être paramétrable et être totalement formaté pour l'application web de suivi de courses de bateaux.

4 Aide à stagiaire en Python

Contexte

Tom est un étudiant en master de géosciences océan et est actuellement en stage de fin d'études chez NAOMIS. Sa mission est la récupération et la formalisation de données d'études à partir de fichiers excels afin de créer des outils de valorisation qui les exploiterais au mieux.

Dans le cadre de sa mission, il doit écrire un script python permettant de récupérer les données d'un fichier excel formaté d'une certaine façon et de les inclure dans le modèle de données prévu à cet effet.

Cependant, il n'a pas du tout une formation de développeur et donc l'utilisation de Python est très complexe pour lui.

Objectif

Ma mission est donc de l'assister (à une fréquence d'1 à 3 heures par jour) dans l'écriture de son script afin qu'il puisse avancer sans être trop contraint par la barrière du language. Pour ce faire, je dois, en plus de l'aider directement avec mes connaissances, le sensibiliser sur les bases et les bonnes pratiques de la programmation et de Python.

5 Synchronisation de QFieldCloud avec une base de données QGIS

Contexte

Pour la DIRSO[10] (Direction Interdépartementale des Routes du Sud-Ouest) qui est responsable des autoroutes liées à Toulouse et leurs équipements (barrières, caméras, panneaux à messages variables,...), NAOMIS a créé un projet sur **QGIS** permettant de visualiser sur une carte tous ces éléments à partir d'une base de données interne.

Les agents de terrains, eux, possédaient une application nommée **QField** qui fonctionne en offline et qui leur permettaient de modifier et ajouter des données puis de les synchroniser avec un cloud lié à l'application appelé **QFieldCloud**. Puis, la base de données était synchronisé avec QFieldCloud pour que QGIS et QField ai leurs données à jour.

Malheureusement, suite à de nouvelles restrictions du gouvernement, Il est désormais interdit (pour des raisons de sécurité) de permettre des connexions externes à des bases de données internes comme avec QFieldCloud. Et donc il n'est désormais plus possible pour le client de synchroniser ces deux parties.

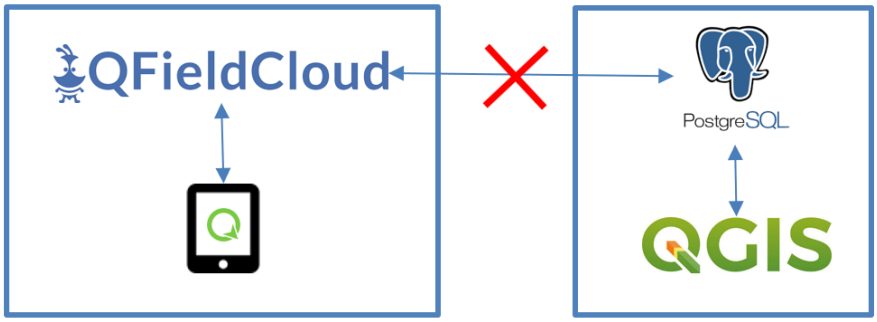


Figure 1: Image explicative de l'environnement du projet après réforme

Objectif

L'objectif de cette mission est donc d'écrire un script python permettant de récupérer les données d'un projet QFieldCloud et de les synchroniser avec la base de données correspondante. Ce script devra permettre de choisir entre mettre à jour la bd avec les gepackages ou mettre à jour les geopackages avec la bd. Peut importe le sens, il devra ajouter les nouvelles données dans les tables cibles et mettre à jour les données déjà existantes en se basant sur la date de modification (on garde la donnée qui à été modifiée en dernier).

Quatrième partie

Développement des missions

⚠ Warning: Missions développées

Seul la mission "Gestion de CV" sera développé dans ce rapport car elle à été ma plus grosse mission (probablement 50% de mon stage) et donc occupera l'entièreté de cette section sans difficulté.

1 Gestion de CV

J'affiche ci-dessous une image du modèle de CV que doivent respecter tous les collaborateurs afin que vous ayez une image de sa structure et de ses différentes sections.

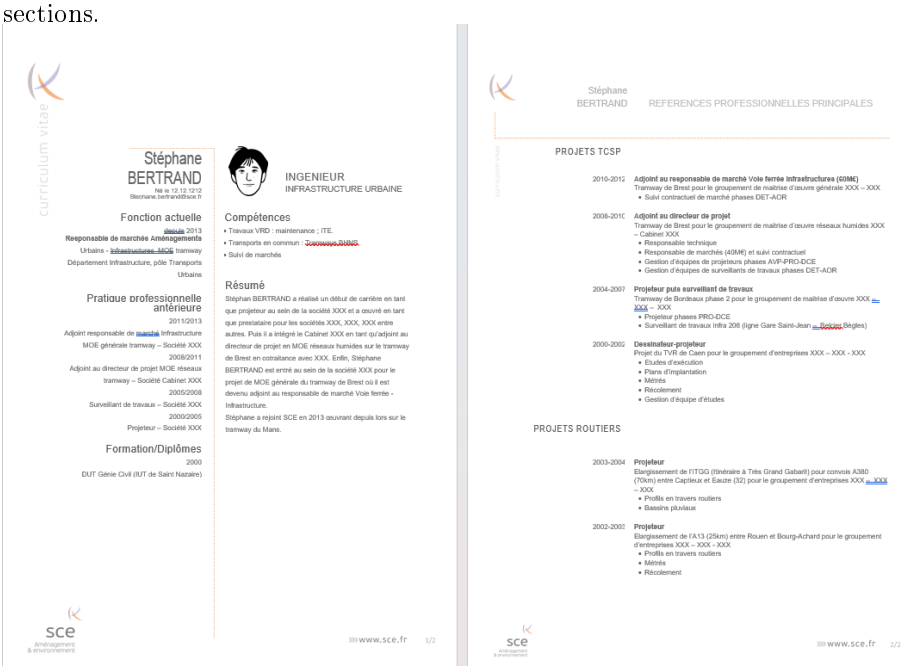


Figure 2: Modèle officiel des CV de SCE

Sous sommaire

1.1 Les tâches et leurs réalisations

Recherche du format de données le plus adapté à cette mission

La première étape a été d'analyser plusieurs formats de fichier dans lesquels convertir les cv word. Le but étant de trouver celui qui sera le plus facilement exploitable.

J'en ai essayé plusieurs comme :

- Le **HTML dans un fichier unique** et le **XML** : L'avantage était qu'ils englobaient chaque information d'une balise avec une classe unique qui pouvait servir à différencier les données. Malheureusement les fichiers étaient illisibles et très peu débuggables et analysables.

Figure 3: Aperçu d'un fichier Word converti en HTML

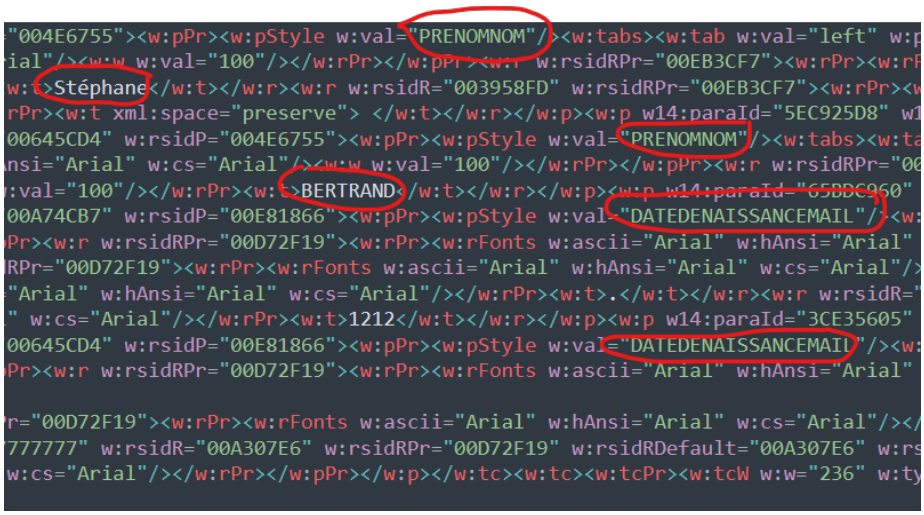


Figure 4: Aperçu d'un fichier Word converti en XML

- Le **Texte brut** : L'avantage de ce format était qu'il était extrêmement simple et que l'agencement du fichier restait le même si le CV n'était pas trop loin du modèle. Ce format perdait néanmoins les styles des textes qui auraient pu être utiles pour identifier certains champs. Toutefois, c'est ce format que j'ai choisi d'exploiter.

Figure 5: Aperçu d'un fichier Word converti en Texte brut

- Le **Fichier Word d'origine** : Il est possible d'exploiter le format Word directement, ce qui nous donne accès à énormément d'informations en plus, même si j'ai trouvé qu'il était assez compliqué de récupérer toutes les données en se basant sur ce format. Je l'ai exploité en complément du format Texte brut.

Recherche de discriminants et de contraintes algorithmiques dans le format choisi

Maintenant que j'ai choisi le format de nos fichiers, il faut analyser la structure des documents afin de repérer des moyens de detecter et reconnaître les différentes données ainsi que les variations classiques et probables entre ces fichiers.

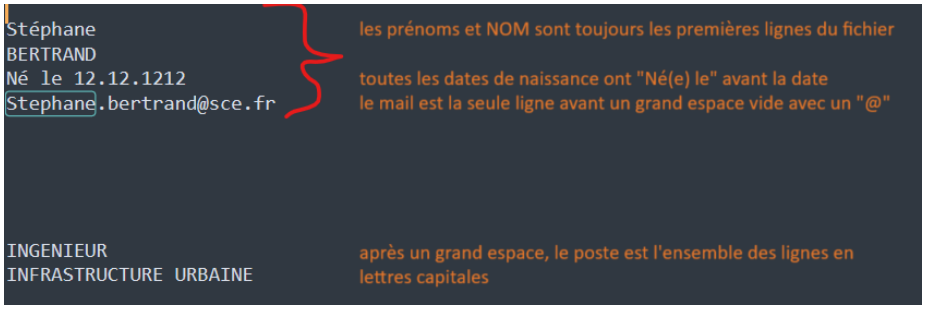


Figure 6: Exemple de discriminants trouvés

Malheureusement, les CVs n'étant pas tous exactement aux mêmes normes j'ai dù prendre en compte un certain nombre de variables pour essayer de faire fonctionner mon script sur le plus de variations possibles.

voir annexe 3.4: Exemple 1 de variations dans les CVs

voir annexe 3.5: Exemple 2 de variations dans les CVs

Extraction des données des CVs

⚠ **Warning:**

Cette partie de mon code est peut-être la seule dont je ne suis pas très fier de tout mon stage. Malgré quelques améliorations, je n'arrive pas du tout à en être totalement satisfait.
Je trouve notamment que cette partie respecte peu les "bonnes pratiques" de développement et que l'algorithme d'extraction de données est très brut/naïf. Pourtant, d'ordinaire j'essaie toujours de coder "proprement" d'où mon insatisfaction.

Maintenant que la phase d'analyse est terminée et que nous possédons les fichiers convertis, il est temps de passer à la phase d'extraction avec les nouvelles problématiques qu'elle apporte (performance, efficacité, choix des bibliothèques,...).

Pour récupérer les données dans les CVs Word convertis en fichiers texte, on parcourt le fichier de haut en bas en essayant de reconnaître les différents facteurs **discriminants** qui englobent chaque donnée pour les récupérer. À cette étape, on ne cherche pas encore à séparer les différentes composantes d'un élément mais plutôt à les récupérer dans leurs entièretés en essayant de bien détecter leurs débuts et leurs fins.

Dans l'algorithme que j'utilise pour séparer les données, je fais souvent appel à la structure de la donnée pour l'identifier. Par exemple, si je suis dans la section Formations j'estime que le début de la formation commence par une date et que la fin d'une formation se trouve juste au dessus du début de la prochaine, donc juste au dessus d'une date.

Et c'est là que cela se complique car une 'date' peut être autant '27/01/2012' que '27 janvier 2012' que '12.01.27'. Et donc, j'ai besoin de détecter toutes les sous-chaines de caractères qui satisfassent l'un de ces modèles pour les récupérer ensuite.

Heureusement, le traitement de texte existe depuis extrêmement longtemps et une méthode de détection de paternes existe bel et bien, ce sont les ***RegEx***.

Info: Les expressions régulières

Les *RegEx* (**Reg**ular **Ex**pression) ou *Expressions Régulières* en français sont des expressions logiques permettant de représenter un ensemble de conditions que la chaîne de caractères à analyser devra respecter pour être considérée valide.
De nos jours, ils sont utilisés dans beaucoup de domaines nécessitant une validation de chaîne de caractères. Par exemple, l'analyse d'une adresse mail ou d'un mot de passe bien formé ou même la fonction Rechercher présente dans tous les navigateurs et logiciels de traitement de textes utilisent des RegEx.

Une bonne partie de mon temps à été consacrée à l'apprentissage et l'utilisation de ces RegEx dans mon algorithme. Heureusement, M.Mottu nous avait déjà sensibilisé à cette notion d'expressions régulières ce qui fait que je ne partait pas de zéro. J'ai essayé de capturer le plus de cas différents possibles et je suis arrivé à un nombre de formats pris en compte assez satisfaisant.

voir annexe 3.6: Tous les formats de dates que j'ai réussi à prendre en compte

Malgré tout, je n'arrivais toujours pas à récupérer correctement la section 'projets' des CVs (faute à une trop grande variance entre les CVs et peu de discriminants communs). Sur les conseils de Fabien CERISIER, j'ai essayé d'exploiter la structure du CV Word directement à mon avantage. Le fait est que dans le fichier Word, la section projet est présentée sous forme de tableau avec

dans la colonne de gauche les dates et dans la colonne de droite le descriptif du projet. Il y avait évidemment de la variance d'un CV à l'autre. Malgré tout, en exploitant cette particularité j'ai drastiquement augmenté la quantité de CVs pour lesquels les projets étaient en grande partie bien récupérés.

voir annexe 3.7: Exemple de tableau de la partie Projets des CVs

Enfin, pour s'assurer de la bonne récupération des données, il a fallu nettoyer avant, pendant et après la récupération de celles-ci. Que ce soit en supprimant tous les caractères spéciaux, les sauts de pages ou les différents espaces au début et à la fin des chaines de caractères, le travail de nettoyage à fait partie intégrante de la récupération des données.

Conversion des CVs Word dans le format choisi

Pour mes tests pendant la phase d'extraction de données, j'ai utilisé des CVs que j'avais manuellement converti avec Word directement. Mais pour ce projet, la donnée d'entrée devait rester une liste de CVs au format Word et donc toute conversion devait se faire automatiquement et de manière interne au script.
Pour effectuer ces conversions, j'ai testé beaucoup de librairies python qui ne m'ont malheureusement pas convaincu comme Aspose (payant) ou docx-to-txt. Le problème étant que malgré le fait qu'elles fonctionnaient, les données n'étaient pas formatées comme les fichiers convertis directement dans Word. Il y avait notamment beaucoup de sauts de lignes et de tabulations rajoutés un peu partout.

J'ai donc pensé à élargir mon champ d'action et à ne pas me limiter aux bibliothèques python, ce qui m'a permis de trouver un utilitaire qui correspondait parfaitement à mon besoin.
L'utilitaire DocTo[12] a la particularité d'utiliser directement le moteur de Word lui même pour les conversions. Ce qui garantit que le formatage soit exactement le même que précédemment. La seule contrainte était donc d'avoir Word installé sur sa machine. Dans mon cas ce n'était pas une contrainte restrictive car toutes les machines du groupe KERAN en sont équipés.

Info: Double conversion

L'utilitaire DocTo permet de convertir les fichiers Word **récents** (.docx) en fichier texte (.txt). Néanmoins, tout fichier Word datant d'avant 2003 (date de mise à jour de Word sur le format des fichiers) possède une autre extension (.doc). Heureusement, DocTo permet également de convertir les fichiers Word anciens en fichier Word récent. Il est malgré tout important de noter que cela veux dire que pour tout fichier word ancien, une double conversion est nécessaire .doc->.docx et .docx->.txt faisant donc perdre en efficacité et en performance mon script.

Séparation des composantes des éléments

A ce niveau du script nous possédons donc les différents éléments récupérés dans les CVs séparés entre eux. Par contre, il nous reste toujours à trouver et séparer les différents composants de chaque élément.

Pour ce faire, pour chaque type de données on essaye d'analyser quels sont les composants standards de l'élément.

Après cette phase d'analyse, on essaye, pour chaque composant, de le séparer de l'objet de base.

Et en définitive, j'obtiens un format de données plutôt satisfaisant.

voir annexe 3.8: Elements séparés entre eux

voir annexe 3.9: Composants dans un élement

Info: Résultats attendus

Il y a des moments où je n'ai pas réussi à séparer tous les composants d'un élément ou alors le formatage était trop variable et pas assez identifiable pour pouvoir baser mon script dessus. Dans ces cas-là j'ai demandé aux personnes en charge du projet ce que je devais faire. Elles m'ont précisé que mon script n'avais pas vocation à être parfait et que bien récupérer et séparer les données que je prend en charge était plus important que de prendre en charge le plus de données possible.

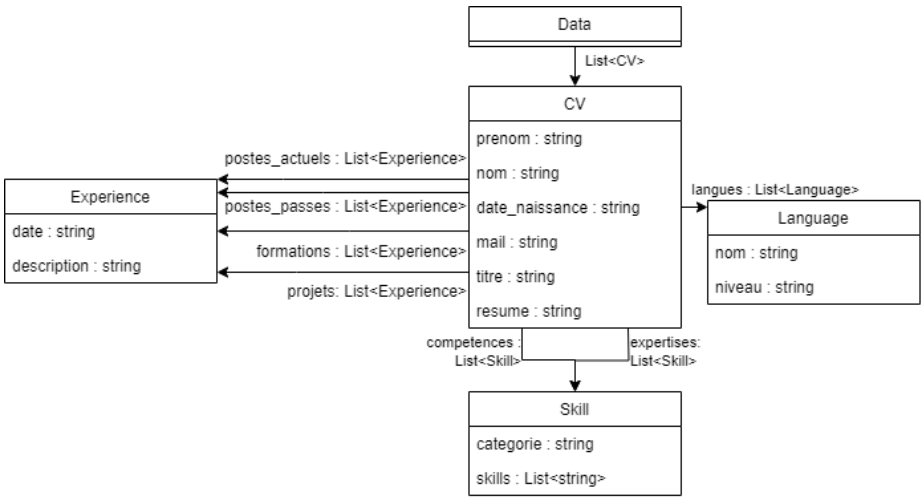


Figure 7: modèle finale des données de mon programme

Ici, il y a 4 **classes** représentant mon modèle avec **CV** comme classe centrale et les autres classes permettant de rassembler les différentes en groupe de

sections selon leurs champs.

Par exemple, la classe **Experience** représente les sections qui possèdent des composants pour lesquels j'arrive à extraire une date et une description. C'est notamment le cas de la section "formations" et la section "projets". Donc, par exemple, la liste des projets d'un CV sera construite comme un ensemble d'objets de type "Experience".

Formatage des données pour correspondre au modèle de données de la base de données

Pour le formatage des données, j'ai utilisé des **DataClasses** qui représentaient chacune l'une des tables de la base de données que je devais alimenter avec mon script.

voir annexe 3.10: Partie du modèle de données de la base de données sur laquelle j'agis

voir annexe 3.11: Modèle de données de lé abse de données complet

- La table **CV** correspond à toutes les **métadonnées** d'un CV
- La table **Experience** correspond à toutes les métadonnées d'une expérience (le type d'expérience, la langue dans laquelle est écrite l'expérience, l'utilisateur à qui elle appartient, ...)
- La table **info_experience** correspond à toutes les données d'une expérience, c-à-d les données que j'ai extraites des CVs avec mon script. D'ailleurs, les tables **Experience** et **info_experience** ont été séparées pour des besoins de traduction : toutes les données dans **info_experience** sont traductibles et toutes celles dans **Experience** non.
- La table **affichage_reservoir** fait le lien entre un CV et l'ensemble des expériences qui le compose. La notion d'ordre correspond à l'ordre d'affichage des expériences dans leur rubrique correspondante dans l'application.

Pour la table **info_experience**, une autre difficulté est que selon le type d'expérience à laquelle elle est affiliée, ce ne sont pas les mêmes champs qui doivent être alimentés. Pour s'y retrouver, une table de correspondance a été établie par une collègue.

Rubrique	**date**	**intitulé**	**description**	**organisme/client**	**niveau langue**
résumé			x		
fonction actuelle	x	x	x		
postes_actuel	x	x	x		
postes_passes	x	x	x	x (si existant)	
formations	x	x	x	x (organisme)	
projets	x	x	x	x	
expertise		x (=categorie)	x (=skills)		
compétence		x (=categorie)	x (=skills)		
langue		x (=nom langue)			x

Figure 7 : Table de correspondance des champs selon le type d'expérience

⚠ Warning: Exception

Je n'ai pas réussi à séparer de manière efficace et constante les composants intitulé,description et organisme/client. Alors, l'élément représentant ces trois composants est envoyé directement dans le champ description.
Le problème de ces composants est qu'ils sont très variables et aucune norme ne ressort quant à leurs spécifications. Si on prend l'exemple du client, certains peuvent le spécifier en l'écrivant en gras, en italique ou en les soulignant (styles que je perds lors de la conversion en Texte Brut). D'autres, peuvent le spécifier en le plaçant juste après la date ou à la fin de l'élément et dans ce cas, je n'ai toujours aucun moyen de connaitre le choix effectué pour chaque CV et je n'ai aussi aucun moyen de vérifier ma donnée après récupération.

Ensuite, au moyen de **DAOs**, j'ai récupéré les éléments formatés avec mon modèle de données afin de les convertir en éléments formatés avec le modèle de données de la base de données.

J'ai notamment utilisé les nom, prénom et mail récupérés dans les CVs pour récupérer l'identifiant de l'utilisateur dans la base de données du groupe KERAN ou encore utiliser les intitulés des rubriques pour retrouver leurs id et les renseigner aux bons endroits.

Une fois ces étapes effectuées et tous les champs de toutes mes simili tables correctement renseignés, mes objets python étaient complets et prêts à être envoyés en base de données.

Envoi des données récupérées dans la base de données

C'est aussi avec mes DAOs que je mets à jour la base de données. Pour ce faire, j'utilise les objets construits à l'étape précédente et j'écris mes requêtes en utilisant comme données les informations contenues dans mes classes.

Info: Sécurité des requêtes SQL

J'utilise des requêtes préparées pour toutes les interactions de mon code avec la base de données. Celles-ci permettent de parer les injections SQL indésirables de potentiels utilisateurs mal intentionnés et donc de rendre le code plus sûr.

voir annexe 3.12: Implémentation d'une requête préparée

1.2 Autres réalisations & spécificités

Pendant la réalisation de ce projet, il y a certains moments où je me retrouvait bloqué par certaines difficultés ou alors j'avais besoin de faire des pauses. Dans ces moments là, je m'affairais donc à réaliser un certain nombre de réalisations non nécessaires au fonctionnement du programme mais qui s'avéraient agréables.

Documentation du projet

Pour tout projet visant à être utilisé par d'autres personnes ou même par nous même plus tard, la documentation est indispensable. C'est pour cela que toutes mes fonctions, classes et modules ont été documentés avec soin grâce aux docstrings de Python.

De plus, j'ai pu générer une documentation HTML grâce à Sphinx[1], ce qui veut dire que j'ai une documentation lisible et disponible en dehors même de mon programme.

voir annexe 3.13: Aperçu de la page d'accueil du site généré de la documentation

Ligne de commande

Mon script a besoin d'un certain nombre d'informations pour fonctionner (la liste des fichiers à traiter, la base de données à alimenter,...). Pour pouvoir rendre mon script paramétrable, j'ai donc décidé de faire en sorte que les paramètres nécessaires (ou optionnels) de mon script puissent être renseignés au moyen d'une ligne de commande dans un terminal.

voir annexe 3.14: Description de la ligne de commande avec tous les paramètres disponibles

Cycle de vie du programme

Au début, le programme agissait naïvement en effectuant chaque différent traitement pour l'ensemble des fichiers avant de passer à l'étape suivante. Cette approche posait quelques problèmes de robustesse notamment car cela voulait dire que pour arriver à récupérer les informations à la fin du programme, l'entièreté des fichiers devait être traitée et si une erreur était provoquée l'ensemble des traitements se révélaient donc inutiles.

Pour pallier ce problème, l'idée à été de changer le **cycle de vie** du programme de telle sorte que pour chacun des fichiers à traiter, le programme effectue l'ensemble des traitements sur lui avant de passer au traitement du prochain fichier. De ce fait, même si le programme venait à planter toute la 'progression' ne serait pas perdue.

voir annexe 3.15: Ancien cycle de vie

voir annexe 3.16: Nouveau cycle de vie

Logging

Le **logging** est quelque chose d'essentiel pour pouvoir debugger un programme avec de multiples traitements. Cela permet de donner à l'utilisateur une vue d'ensemble du fonctionnement du programme et de lui rendre visible les anomalies ainsi que les potentielles raisons et solutions pour le programme.

voir annexe 3.17: Aperçu d'un fichier de log

Création d'un éxecutable Windows

Dans une optique d'indépendance par rapport au système, j'ai cherché un moyen de créer un exécutable (.exe) à partir de mon programme pour faire en sorte que quelque soit le système (tant que c'est Windows), le programme puisse s'exécuter. C'est un peu la continuité de la création d'une commande pour le terminal pour mon programme car c'est grâce à cela que l'utilisateur pourra paramétrer le programme même sans avoir accès au code source.

1.3 Conclusion

Résultats

Pour tester mon programme final j'ai eu une soixantaine de CVs. En effectuant des tests de performances sur ceux-ci j'obtiens comme résultats :

- un temps d'exécution d'environ 0.1 à 0.3 seconde par CV sans les conversions, environ 3 secondes avec une seule conversion et environ 8 secondes avec la double conversion
- l'espace de stockage utilisé en plus équivaut à 1 .docx pour chaque cv en .doc et 1 .txt en plus pour tout les cv (.docx et .doc).
- d'après mon analyse des résultats, j'estime la bonne récupération des données à environ 75%
- toujours selon moi, j'estime récupérer 30% des potentielles erreurs avec mon logging (pas eu le temps de le développer à l'échelle de l'entiereté du projet)

Au final, la section que je n'ai pas réussi à récupérer est une section optionnelle présente uniquement dans les CVs de NAOMIS (section Formation continue). Et les seules données que je n'ai pas réussi à séparer sont les organismes, les intitulés et les descriptifs des expériences.

Le programme est donc livré avec son code source et sa version exécutable tous deux paramétrables ainsi qu'une documentation HTML et toutes les fonctionnalités précédemment évoquées.

Savoir et compétences acquises

Ce que j'ai le plus appris dans ce projet c'est l'utilisation des RegEx qui a été un point culminant de celui-ci. J'ai aussi énormément appris en terme d'algorithmique. Les compétences acquises notamment dans les cours de M.ATTIOGBE et M.SIMONNEAU m'ont été très utiles.

J'ai également beaucoup appris sur les librairies classiques python (os, time/datetime, dataclasses, et j'en passe).

Et enfin, j'ai appris à utiliser PostgreSQL (jusqu'alors je n'avais utilisé qu'**Oracle**, **MariaDB** et **MongoDB**).

Synthèse

Le traitement de données est quelque chose d'extrêmement complexe dont j'ai toujours eu peur. Cependant, c'est un domaine très intéressant qui est de plus extrêmement présent dans toutes nos utilisations du numérique.

Concernant mon programme, il a une bonne marge d'amélioration notamment sur la partie récupération de données qui peut toujours être meilleure ainsi que la partie détection des potentielles erreurs.

Mais malgré tout, je pense que mon programme est suffisamment efficace pour remplir sont rôle de première initialisation. Par ailleurs, il permettra, a minima, de faciliter la tâche de nettoyage aux personnes qui passeront après.

Cinquième partie

Conclusion

Ce stage de deuxième année de BUT Informatique a été une expérience extrêmement enrichissante tant sur le plan professionnel que personnel. Il m'a permis d'appliquer les connaissances théoriques acquises durant mes cours à des situations concrètes et de me familiariser avec le milieu professionnel.

Au cours de ce stage, j'ai eu l'opportunité de travailler sur divers projets qui m'ont aidé à renforcer mes compétences techniques et à acquérir de nouvelles aptitudes. En particulier, j'ai pu approfondir ma compréhension des bases de données relationnelles et non relationnelles, ainsi que l'intérêt et l'utilité de Docker et Docker-compose. Ces expériences m'ont permis de mieux saisir l'importance d'une bonne architecture logicielle et des pratiques de codage rigoureuses.

J'ai également développé des compétences en gestion de projet et en communication, essentielles pour travailler efficacement en équipe. Les échanges réguliers avec mes collègues et ma tutrice de stage ont été des éléments clés de cette expérience, me permettant d'améliorer ma capacité à résoudre des problèmes complexes et à proposer des solutions innovantes.

En conclusion, ce stage m'a non seulement permis de valider mes acquis, mais aussi de découvrir de nouveaux aspects du domaine de l'informatique que je souhaite explorer davantage. Je suis désormais plus confiant dans mes compétences et motivé à poursuivre ma formation.

Je tiens à remercier toute l'équipe de l'entreprise pour leur accueil chaleureux, leur soutien et les précieux conseils qu'ils m'ont prodigués tout au long de cette période.

Cette expérience m'a conforté dans mon choix de carrière et m'a ouvert les yeux sur les multiples facettes de la vie en entreprise. Je suis impatient de continuer à apprendre et à évoluer dans le domaine de l'informatique et à découvrir toutes les possibilités qui s'offrent à moi.

Sixième partie

Annexes

1 Sources

[1] *Documentation de Sphinx, le générateur de documentation Python*. 2024. URL : https://www.sphinx-doc.org/en/master/.

[2] Team Vendée FORMATION. *article informatif sur le défi des villes 2024*. 2024. URL : https://teamvendeeformation.fr/defi-des-villes/.

[3] *lien du site officiel de Créocéan*. 2024. URL : https://creocean.fr/.

[4] *lien du site officiel de Groupe Huit*. 2024. URL : https://groupehuit.com/.

[5] *lien du site officiel de KERAN*. 2024. URL : https://www.groupe-keran.com.

[6] *lien du site officiel de NAOMIS*. 2024. URL : https://www.naomis.fr/fr.

[7] *lien du site officiel de S3D*. 2024. URL : https://sol3d.fr/.

[8] *lien du site officiel de SCE*. 2024. URL : https://sce.fr/.

[9] *lien du site officiel de YS EMD*. 2024. URL : https://ysemd.com/.

[10] *Site officiel de la DIRSO*. 2024. URL : https://www.dir.sud-ouest.developpement-durable.gouv.fr/.

[11] *site officiel de Python*. 2024. URL : url.

[12] TOBYA. *repertoire github de l'utilitaire DocTo*. 2024. URL : https://github.com/tobya/DocTo.

[13] Vendée TOURISME. *article sur la sardinha cup 2022*. 2022. URL : https://www.vendee-tourisme.com/la-sardinha-cup.

2 Glossaire

A | **B** | **C** | **D** | **G** | **H** | **J** | **K** | **L** | **M** | **P** | **Q** | **R** | **S** | **V** | **W** | **X**

A

API Un ensemble de fonctions et de procédures permettant à des applications de communiquer entre elles. Derrière, YouTube, Google Maps ou encore Netflix se cachent des API permettant de transférer et traiter les données et les informations des serveurs aux sites web, pour ces exemples. . 2

B

base de données Une base de données est un ensemble organisé de données structurées et interconnectées, généralement stockées électroniquement dans un système informatique. Les bases de données permettent de stocker, gérer et récupérer efficacement des informations, souvent utilisées dans des applications informatiques, des systèmes d'information et des sites web pour gérer de grandes quantités de données de manière structurée et cohérente. . 2

C

classe (en programmation orientée objet) En programmation orientée objet (POO), une classe est un modèle ou un plan de construction pour créer des objets. Elle définit les propriétés (attributs) et les comportements (méthodes) communs à tous les objets qui en sont instanciés. Les classes permettent de structurer le code en regroupant des données et des fonctionnalités associées ensemble. . 22

Cycle de vie du programme Ce que j'appelle le "cycle de vie du programme", c'est l'ensemble des traitements qu'il effectue pendant son exécution (conversion des fichiers, extraction des données, formatage, envoi en base de données,...). . 25

D

DAO En programmation et en architecture logicielle, DAO est un motif de conception qui permet d'encapsuler l'accès aux données dans une couche d'abstraction. Cela permet de séparer la logique métier de la logique d'accès aux données, en fournissant une interface simple et unifiée pour interagir avec différentes sources de données, comme des bases de données ou des services web. . 24

data class En programmation, particulièrement en Kotlin et d'autres langages comme Java, une data class est une classe dont le seul but est de contenir des données. Elle est généralement utilisée pour représenter des structures de données simples en regroupant des propriétés (champs) et en fournissant des méthodes utilitaires telles que les méthodes d'accès (getters) et de modification (setters). . 22, *see also* classe (en programmation orientée objet)

discriminant (en traitement de texte) En traitement de texte, le discriminant fait référence à un critère utilisé pour distinguer ou choisir entre différentes options, comme dans la mise en page automatique ou la gestion des styles. . 19

Docker Plateforme logicielle open-source permettant de créer, déployer et gérer des applications en utilisant des conteneurs. Cela permet entre autres de ne pas avoir à installer les différentes dépendances d'une application sur son propre pc. . 9

G

Git Système de gestion de version décentralisé, conçu pour gérer tout projet, de petit à très large, avec rapidité et efficacité. 9

GitHub Plateforme de développement collaboratif de logiciels reposant sur Git, permettant l'hébergement et le partage de code source. 9, *see also* Git

.gpkg Le format de fichier .gpkg (GeoPackage) est un format de données géospatiales basé sur SQLite. Il permet de stocker des données géographiques ainsi que leurs métadonnées dans une seule base de données SQLite portable. . 2, *see also* SQLite

H

HTML le langage standard utilisé pour créer des pages web. 17

J

Java Un langage de programmation polyvalent, utilisé dans de nombreux domaines tels que le développement d'applications web, mobiles et d'entreprise. 5, *see also* Kotlin

K

Kotlin Un langage de programmation objet moderne et concis, souvent utilisé pour le développement d'applications Android et basé en très grande partie sur Java. 5, *see also* Java

L

Logging Le logging (ou journalisation) est le processus d'enregistrement des événements significatifs qui se produisent lors de l'exécution d'un programme informatique. Il permet de capturer des informations telles que les erreurs, les avertissements, les transactions, et autres actions importantes, dans le but de diagnostiquer les problèmes, de surveiller les performances et de faciliter l'audit des systèmes logiciels. . 25

M

MariaDB MariaDB est un système de gestion de base de données relationnelle open-source, dérivé de MySQL. Il est principalement développé et maintenu par la communauté open-source avec le support de la société MariaDB Corporation et est utilisé pour des applications où la fiabilité, la sécurité et la performance sont essentielles. . 26, *see also* MySQL

MongoDB MongoDB est un système de gestion de base de données NoSQL open-source, orienté document. Contrairement aux bases de données relationnelles traditionnelles, MongoDB stocke les données sous forme de documents JSON flexibles et évolutifs, permettant une modélisation des données plus souple et une mise à l'échelle horizontale facile. Il est largement utilisé dans les applications web modernes et les systèmes nécessitant une manipulation efficace des données non structurées ou semi-structurées. . 26

MySQL MySQL est un système de gestion de base de données relationnelle open-source développé par Oracle Corporation. Il est largement utilisé dans les applications web et les environnements d'entreprise pour stocker et gérer des données structurées. MySQL prend en charge le langage SQL (Structured Query Language) pour interagir avec la base de données, offrant des fonctionnalités telles que la gestion des transactions, la gestion des utilisateurs, la réplication de données et la haute disponibilité. . 26

métadonnée En informatique et en gestion de l'information, une métadonnée est une donnée qui décrit d'autres données. Elle fournit des informations contextuelles et descriptives sur les données, telles que leur origine, leur structure, leur format, leur contenu et leurs relations avec d'autres données. Les métadonnées facilitent la gestion, la découverte, l'organisation et l'utilisation des données. . 22

P

PHP Un langage de script côté serveur principalement utilisé pour le développement web. 5

PostgreSQL un système de gestion de base de données relationnelle et objet open source reconnu pour sa fiabilité, sa richesse fonctionnelle et ses performances. Il est connu pour sa conformité aux standards et son extensibilité. . 2, *see also* SQLite &

Python un language de programmation très utilisé dans l'écriture de script et le développement d'intelligences artificielles. C'est un language interprété qui favorise la programmation fonctionnelle et orientée objet . 2

Q

QField Une application mobile open-source basée sur QGIS, utilisée pour la collecte de données sur le terrain. 13, *see also* QGIS

QFieldCloud Plateforme de synchronisation et de gestion des données collectées avec QField, permettant l'accès et la gestion des données sur le terrain à distance. 13, *see also* QField

QGIS un logiciel SIG libre et open-source pour la cartographie et l'analyse spatiale. 13, *see also* SIG

R

RegEx Une expression régulière, souvent abrégée en regex, est une séquence de caractères qui forme un motif de recherche. C'est avec cela que fonctionne la fonction Rechercher de nos navigateurs internet par exemple. . 2

S

SQLite une bibliothèque en langage C qui implémente un moteur de base de données SQL léger, rapide, autonome, fiable et complet. C'est le moteur de base de données le plus utilisé au monde. . *see also* PostgreSQL & .gpkg

V

VSCode un éditeur de code source développé par Microsoft, disponible sur Windows, macOS et Linux. il a l'avantage de pouvoir facilement installer et créer des extensions permettant d'enrichir très facilement ses possibilités. . 9, *see also* IDE

W

WKB l'équivalent de WKT mais en binaire, très utilisé pour des transferts entre base de données. . *see also* WKT

WKT un format de représentation de données géométriques permettant notamment de représenter des points, des lignes et des polygones sur une carte. . 2, *see also* WKB

X

XML un langage de balisage conçu pour le stockage et le transport de données. 17

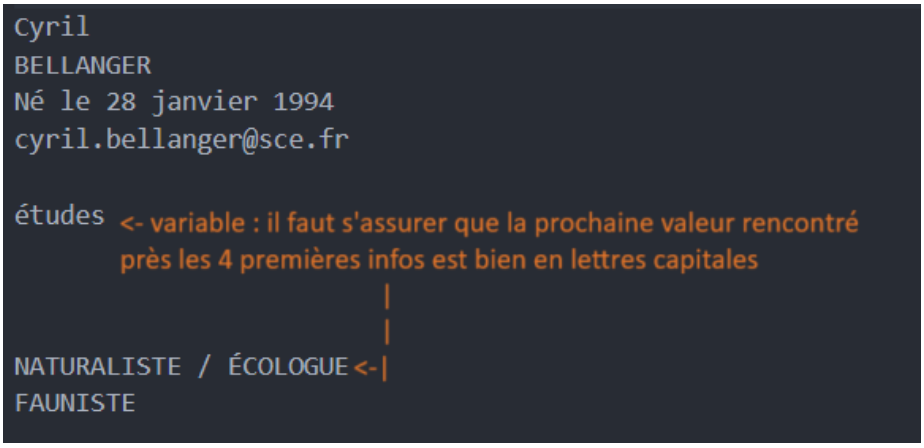
3 Assets

3.1 Asset 1: Organigramme hiérarchique de NAOMIS, Direction et postes auxiliaires

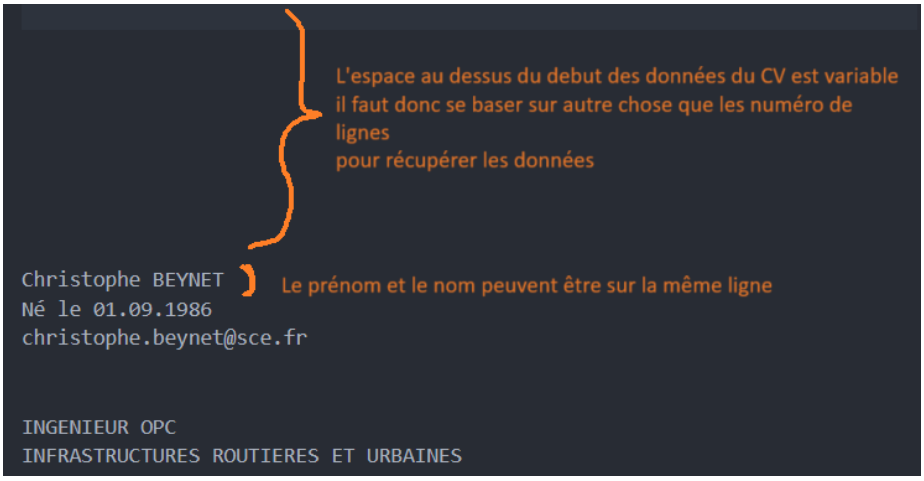
3.2 Asset 2: Organigramme hiérarchique de NAOMIS, pôle Développement

3.3 Asset 3: Organigramme hiérarchique de NAOMIS, pôle Data

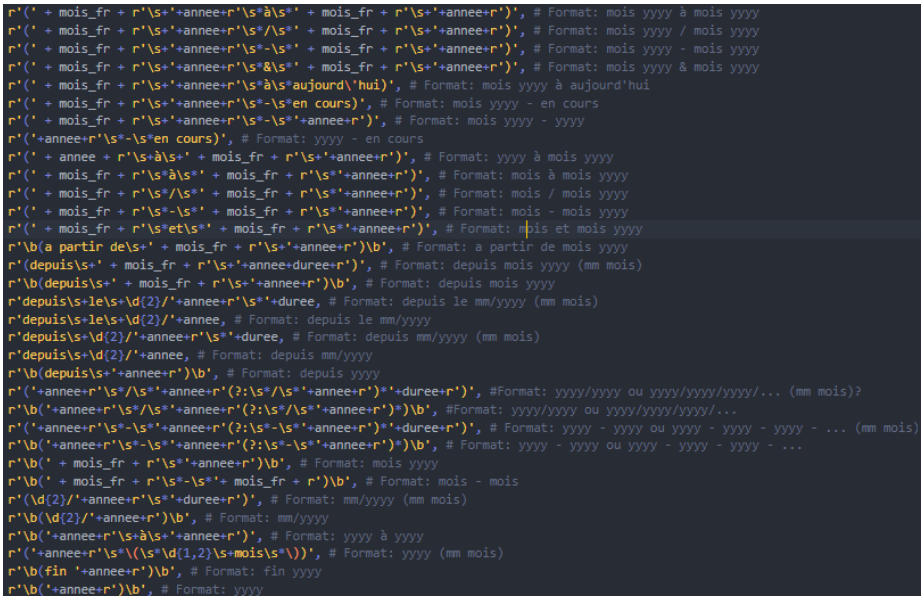
3.4 Asset 4: Exemple de CV avec champ entre le mail et le poste



3.5 Asset 5: Exemple de CV avec espace au début du fichier



3.6 Asset 6: L'ensemble des formats de dates récupérés par mon programme



```
r'(' + mois_fr + r'\s+'+annee+r'\s*à\s*' + mois_fr + r'\s+'+annee+r')', # Format: mois yyyy à mois yyyy
r'(' + mois_fr + r'\s+'+annee+r'\s*/\s*' + mois_fr + r'\s+'+annee+r')', # Format: mois yyyy / mois yyyy
r'(' + mois_fr + r'\s+'+annee+r'\s*-\s*' + mois_fr + r'\s+'+annee+r')', # Format: mois yyyy - mois yyyy
r'(' + mois_fr + r'\s+'+annee+r'\s*&\s*' + mois_fr + r'\s+'+annee+r')', # Format: mois yyyy & mois yyyy
r'(' + mois_fr + r'\s+'+annee+r'\s*à\s*aujourd\'hui)', # Format: mois yyyy à aujourd'hui
r'(' + mois_fr + r'\s+'+annee+r'\s*-\s*en cours)', # Format: mois yyyy - en cours
r'(' + mois_fr + r'\s+'+annee+r'\s*-\s*'+annee+r')', # Format: mois yyyy - yyyy
r'('+annee+r'\s*-\s*en cours)', # Format: yyyy - en cours
r'(' + annee + r'\s+à\s+' + mois_fr + r'\s+'+annee+r')', # Format: yyyy à mois yyyy
r'(' + mois_fr + r'\s*à\s*' + mois_fr + r'\s*'+annee+r')', # Format: mois à mois yyyy
r'(' + mois_fr + r'\s*/\s*' + mois_fr + r'\s*'+annee+r')', # Format: mois / mois yyyy
r'(' + mois_fr + r'\s*-\s*' + mois_fr + r'\s*'+annee+r')', # Format: mois - mois yyyy
r'(' + mois_fr + r'\s*et\s*' + mois_fr + r'\s*'+annee+r')', # Format: mois et mois yyyy
r'\b(a partir de\s+' + mois_fr + r'\s+'+annee+r')\b', # Format: a partir de mois yyyy
r'(depuis\s+' + mois_fr + r'\s+'+annee+duree+r')', # Format: depuis mois yyyy (mm mois)
r'\b(depuis\s+' + mois_fr + r'\s+'+annee+r')\b', # Format: depuis mois yyyy
r'depuis\s+le\s+\d{2}/'+annee+r'\s*'+duree, # Format: depuis le mm/yyyy (mm mois)
r'depuis\s+le\s+\d{2}/'+annee, # Format: depuis le mm/yyyy
r'depuis\s+\d{2}/'+annee+r'\s*'+duree, # Format: depuis mm/yyyy (mm mois)
r'depuis\s+\d{2}/'+annee, # Format: depuis mm/yyyy
r'\b(depuis\s+'+annee+r')\b', # Format: depuis yyyy
r'('+annee+r'\s*/\s*'+annee+r'(?:\s*/\s*'+annee+r')*'+duree+r')', #Format: yyyy/yyyy ou yyyy/yyyy/yyyy/... (mm mois)?
r'\b('+annee+r'\s*/\s*'+annee+r'(?:\s*/\s*'+annee+r')*)\b', #Format: yyyy/yyyy ou yyyy/yyyy/yyyy/...
r'('+annee+r'\s*-\s*'+annee+r'(?:\s*-\s*'+annee+r')*'+duree+r')', # Format: yyyy - yyyy ou yyyy - yyyy - yyyy - ... (mm mois)
r'\b('+annee+r'\s*-\s*'+annee+r'(?:\s*-\s*'+annee+r')*)\b', # Format: yyyy - yyyy ou yyyy - yyyy - yyyy - ...
r'\b(' + mois_fr + r'\s*'+annee+r')\b', # Format: mois yyyy
r'\b(' + mois_fr + r'\s*-\s*'+ mois_fr + r')\b', # Format: mois - mois
r'(\d{2}/'+annee+r'\s*'+duree+r')', # Format: mm/yyyy (mm mois)
r'\b(\d{2}/'+annee+r')\b', # Format: mm/yyyy
r'\b('+annee+r'\s+à\s+'+annee+r')', # Format: yyyy à yyyy
r'('+annee+r'\s*\(\s*\d{1,2}\s+mois\s*\))', # Format: yyyy (mm mois)
r'\b(fin '+annee+r')\b', # Format: fin yyyy
r'\b('+annee+r')\b', # Format: yyyy
```

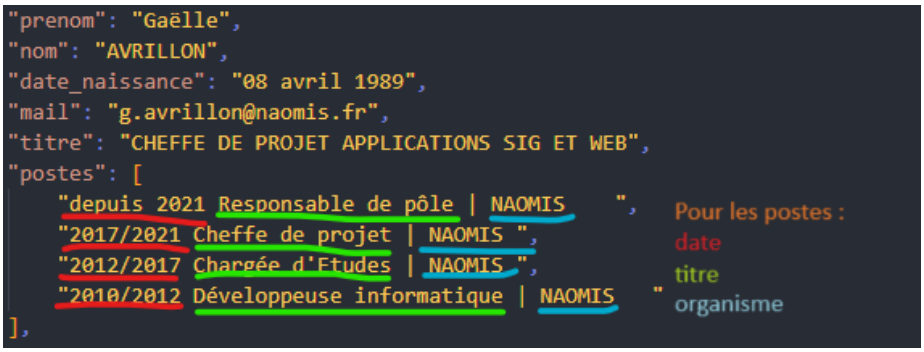
3.7 Asset 7: Exemple de CV avec le tableau des projets mis en évidence

PROJETS TCSP		
2010-2012		**Adjoint au responsable de marché Voie ferrée Infrastructures (60M€)** Tramway de Brest pour le groupement de maitrise d'œuvre générale XXX – XXX
		• Suivi contractuel de marché phases DET-AOR
2008-2010		**Adjoint au directeur de projet** Tramway de Brest pour le groupement de maitrise d'œuvre réseaux humides XXX – Cabinet XXX
		• Responsable technique
		• Responsable de marchés (40M€) et suivi contractuel
		• Gestion d'équipes de projeteurs phases AVP-PRO-DCE
		• Gestion d'équipes de surveillants de travaux phases DET-AOR
2004-2007		**Projeteur puis surveillant de travaux** Tramway de Bordeaux phase 2 pour le groupement de maitrise d'œuvre XXX – XXX – XXX
		• Projeteur phases PRO-DCE
		• Surveillant de travaux Infra 206 (ligne Gare Saint-Jean – Belcier Bègles)
2000-2002		**Dessinateur-projeteur** Projet du TVR de Caen pour le groupement d'entreprises XXX – XXX - XXX
		• Etudes d'exécution
		• Plans d'implantation
		• Métrés
		• Récolement
		• Gestion d'équipe d'études

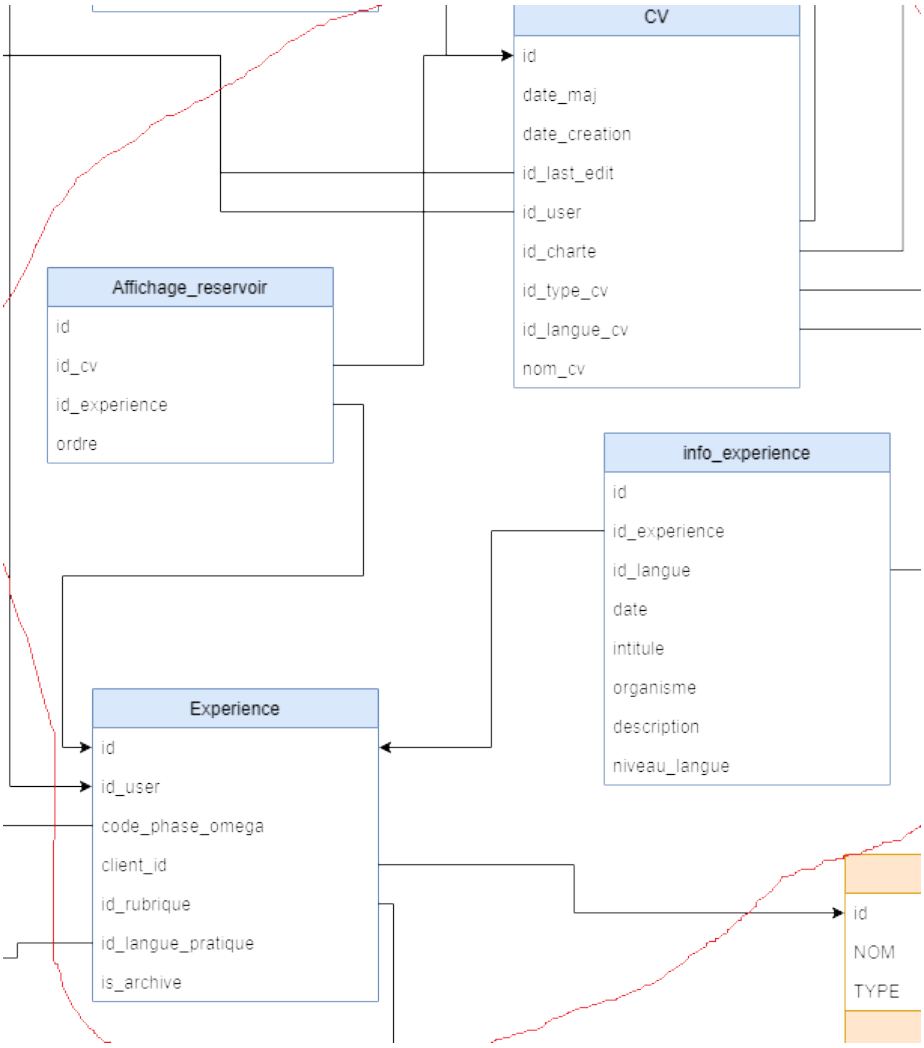
3.8 Asset 8: Exemple d'élements récupérés et séparés entre eux


```
"prenom": "Gaëlle",
"nom": "AVRILLON",
"date_naissance": "08 avril 1989",
"mail": "g.avrillon@naomis.fr",
"titre": "CHEFFE DE PROJET APPLICATIONS SIG ET WEB",
"postes": [
    "depuis 2021 Responsable de pôle | NAOMIS    ",
    "2017/2021 Cheffe de projet | NAOMIS ",
    "2012/2017 Chargée d'Etudes | NAOMIS ",
    "2010/2012 Développeuse informatique | NAOMIS   "
],
"formations": [
    "2012 - 2010 Expert en Informatique et Système d'Information
    certifié au Niveau I (Bac +5) - diplôme réalisé en alternance dans
    la société NAOMIS | EPSI Nantes (44) ",
    "2009 BTS Informatique de gestion, option développeur
    d'applications | EPSI Nantes (44) ",
    "2007 Baccalauréat scientifique mention Bien | Lycée Sacré-Cœur,
    Angers (49)    "
],
"expertises": [],
"competences": [
    "Langages :, HTML 5, CSS, JavaScript, Angular, PrimeNG, PHP,
    Symfony, Java Android, Java Spring Boot, Ionic",
    "Bases de données et ETL : Postgresql, Postgis, MySQL, SQL Server,
    Oracle, Talend Open Studio, ElasticSearch",
    "Outils cartographiques : OpenLayers, Leaflet, WebApp Builder (ESRI)
    ",
    "Systèmes d'exploitation : Windows, Linux, Android",
    "Autres outils : GitHub, JIRA, Confluence, CircleCI",
    "Langues : Anglais"
],
```

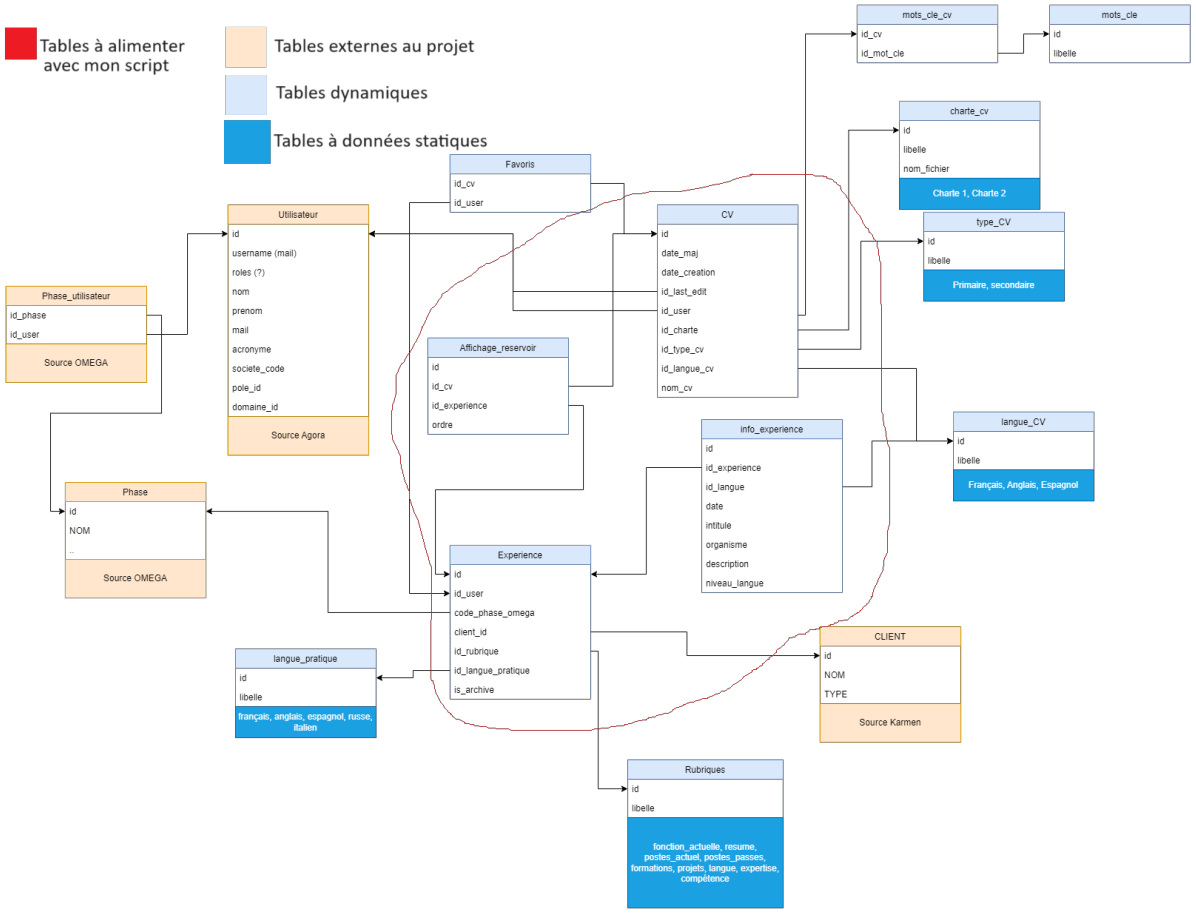
3.9 Asset 9: Exemple de composants à l'intérieur d'un élément



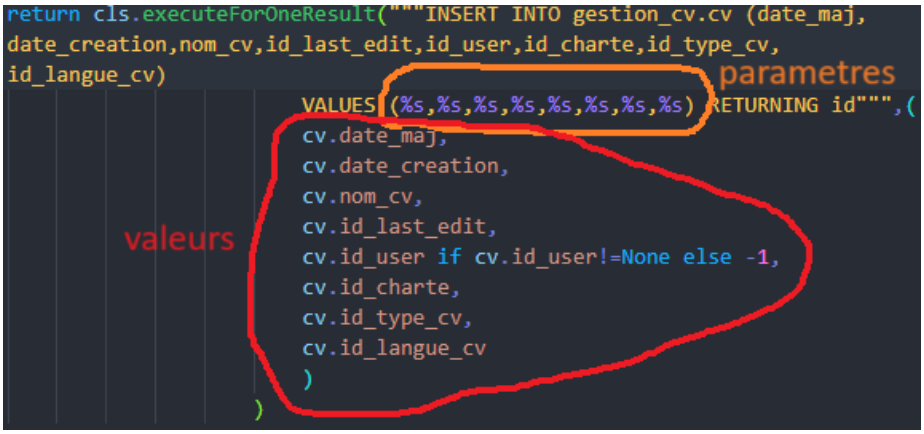
3.10 Asset 10: modèle de données partiel de la base de données



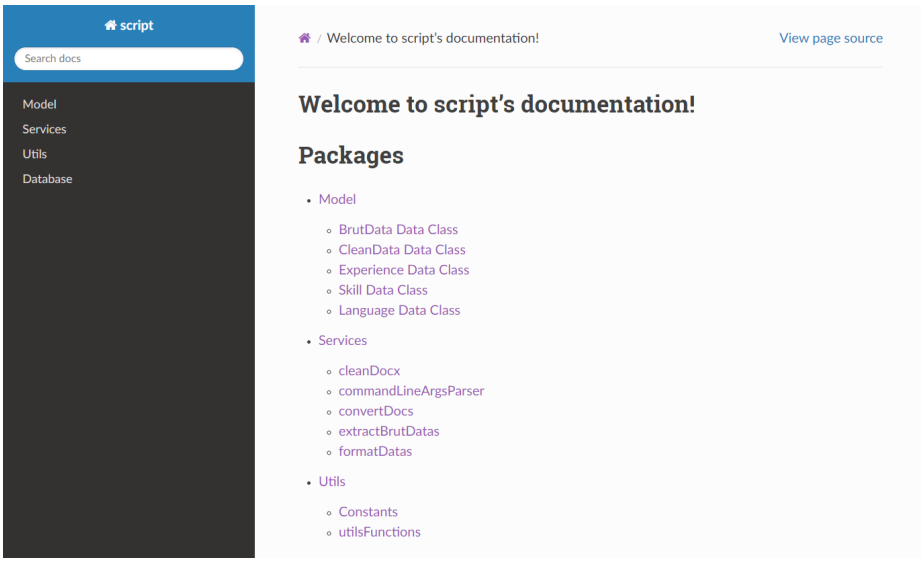
3.11 Asset 11: modèle de données complet de la base de données



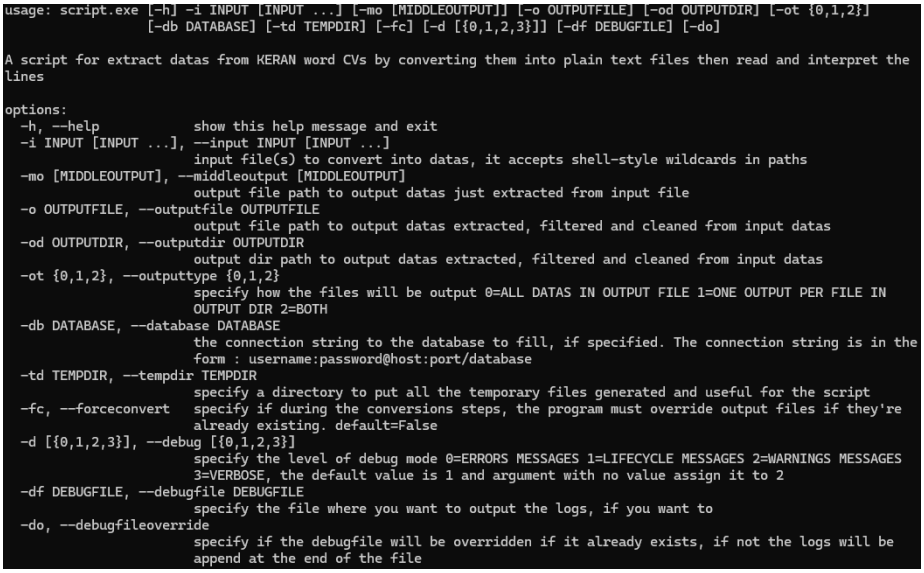
3.12 Asset 12: Exemple de requête préparée



3.13 Asset 13: Page d'accueil du site de la documentation de mon programme

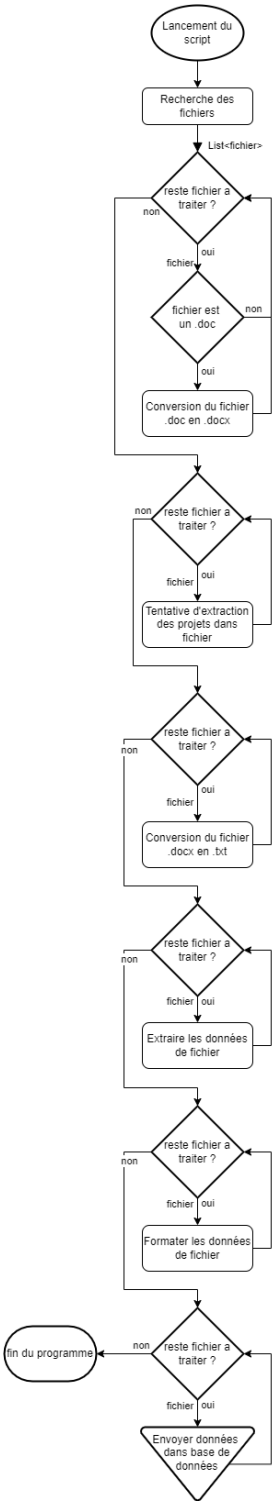


3.14 Asset 14: Page d'aide de la ligne de commande regroupant tout les paramètres possibles

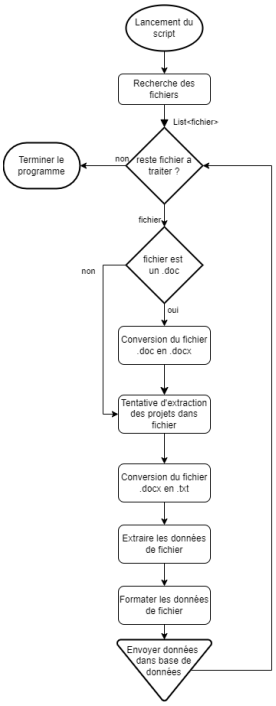


```
usage: script.exe [-h] -i INPUT [INPUT ...] [-mo [MIDDLEOUTPUT]] [-o OUTPUTFILE] [-od OUTPUTDIR] [-ot {0,1,2}]
                  [-db DATABASE] [-td TEMPDIR] [-fc] [-d [{0,1,2,3}]] [-df DEBUGFILE] [-do]

A script for extract datas from KERAN word CVs by converting them into plain text files then read and interpret the
lines

options:
  -h, --help            show this help message and exit
  -i INPUT [INPUT ...], --input INPUT [INPUT ...]
                        input file(s) to convert into datas, it accepts shell-style wildcards in paths
  -mo [MIDDLEOUTPUT], --middleoutput [MIDDLEOUTPUT]
                        output file path to output datas just extracted from input file
  -o OUTPUTFILE, --outputfile OUTPUTFILE
                        output file path to output datas extracted, filtered and cleaned from input datas
  -od OUTPUTDIR, --outputdir OUTPUTDIR
                        output dir path to output datas extracted, filtered and cleaned from input datas
  -ot {0,1,2}, --outputtype {0,1,2}
                        specify how the files will be output 0=ALL DATAS IN OUTPUT FILE 1=ONE OUTPUT PER FILE IN
                        OUTPUT DIR 2=BOTH
  -db DATABASE, --database DATABASE
                        the connection string to the database to fill, if specified. The connection string is in the
                        form : username:password@host:port/database
  -td TEMPDIR, --tempdir TEMPDIR
                        specify a directory to put all the temporary files generated and useful for the script
  -fc, --forceconvert   specify if during the conversions steps, the program must override output files if they're
                        already existing. default=False
  -d [{0,1,2,3}], --debug [{0,1,2,3}]
                        specify the level of debug mode 0=ERRORS MESSAGES 1=LIFECYCLE MESSAGES 2=WARNINGS MESSAGES
                        3=VERBOSE, the default value is 1 and argument with no value assign it to 2
  -df DEBUGFILE, --debugfile DEBUGFILE
                        specify the file where you want to output the logs, if you want to
  -do, --debugfileoverride
                        specify if the debugfile will be overridden if it already exists, if not the logs will be
                        append at the end of the file
```

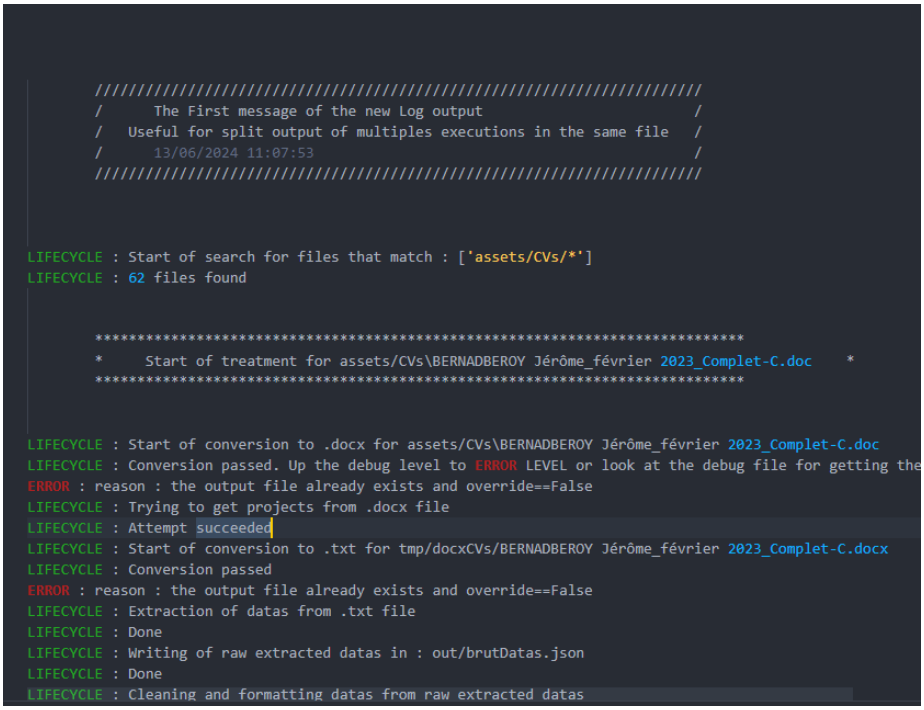
3.15 Asset 15: Diagramme de flux représentant l'ancien cycle de vie de la mission Gestion de CV



3.16 Asset 16: Diagramme de flux représentant le nouveau cycle de vie de la mission Gestion de CV



3.17 Asset 17: Exemple de fichier de log produit par mon programme



```
////////////////////////////////////////////////////////////////////////
/      The First message of the new Log output                       /
/   Useful for split output of multiples executions in the same file  /
/      13/06/2024 11:07:53                                           /
////////////////////////////////////////////////////////////////////////


LIFECYCLE : Start of search for files that match : ['assets/CVs/*']
LIFECYCLE : 62 files found


        **************************************************************************
        *     Start of treatment for assets/CVs\BERNADBEROY Jérôme_février 2023_Complet-C.doc    *
        **************************************************************************


LIFECYCLE : Start of conversion to .docx for assets/CVs\BERNADBEROY Jérôme_février 2023_Complet-C.doc
LIFECYCLE : Conversion passed. Up the debug level to ERROR LEVEL or look at the debug file for getting the
ERROR : reason : the output file already exists and override==False
LIFECYCLE : Trying to get projects from .docx file
LIFECYCLE : Attempt succeeded
LIFECYCLE : Start of conversion to .txt for tmp/docxCVs/BERNADBEROY Jérôme_février 2023_Complet-C.docx
LIFECYCLE : Conversion passed
ERROR : reason : the output file already exists and override==False
LIFECYCLE : Extraction of datas from .txt file
LIFECYCLE : Done
LIFECYCLE : Writing of raw extracted datas in : out/brutDatas.json
LIFECYCLE : Done
LIFECYCLE : Cleaning and formatting datas from raw extracted datas
```