

Rapport de stage



LE MAREC Trystan
BTS SIO (Service aux Organisations Informatiques)
Option SLAM
2024-2025

06 janvier 2025 - 14 février 2025

Tuteur de l'entreprise : Jean-Marc Moreau
Tuteur pédagogique : Béatrice Caramigeas

Entreprise d'accueil : EXAEGIS
Etablissement : Paul-Louis Courier Tours

Remerciements

Je souhaite exprimer ma profonde gratitude à mon maître de stage, Jean-Marc Moreau, pour son accompagnement et son expertise tout au long de cette expérience. Grâce à ses conseils avisés, j'ai pu élargir mes connaissances et explorer de nouvelles perspectives dans mon domaine.

Je remercie également l'ensemble des membres de l'entreprise pour leur disponibilité et leur patience. Leur bienveillance et leurs précieux conseils m'ont permis de progresser et de mieux appréhender les différentes facettes de mon travail.

Enfin, je suis reconnaissant d'avoir eu l'opportunité d'effectuer ce stage au sein de cette équipe. Cette expérience m'a non seulement permis de renforcer mes compétences, mais aussi d'évoluer dans un environnement professionnel enrichissant. Je remercie mes collaborateurs pour leur confiance et leur soutien, qui ont contribué de manière significative à mon apprentissage et à mon développement professionnel.

Sommaire

1. Introduction.....	4
1.1. Présentation globale de l'entreprise.....	5
1.2. Situation géographique.....	6
1.3. Organigramme hiérarchique et fonctionnel.....	7
1.4. Environnement matériel et logiciel de l'entreprise.....	8
2. Missions.....	10
2.1. Mission n°1 : Correction de code de scraping sur le projet exaetrack_scraping_webext.....	10
2.1.1. Présentation du projet.....	10
2.1.2. Ressources.....	11
2.1.3. Gestion de projet.....	12
2.1.4. Collectivités corrigées.....	14
2.1.5. Documentation réalisé :.....	27
2.2. Mission n°2 : Analyse et récupération de données via l'API Sewan.....	28
2.2.1. Présentation de la mission.....	28
2.2.2. Pourquoi utiliser l'API ?.....	28
2.2.3. Problème rencontré.....	28
2.2.4. Présentation du script.....	29
3. Mesure de réussite du projet.....	33
4. Conclusion.....	33

1. Introduction

Au cours de l'année 2024-2025, la réalisation d'un stage en milieu professionnel constitue un prérequis essentiel à l'obtention du diplôme du BTS SIO. Ce stage, d'une durée de six semaines (environ un mois et demi), vise à renforcer nos compétences en matière de développement, de services et de systèmes, tout en enrichissant notre expérience sur le terrain. En outre, il représente un atout majeur pour notre curriculum vitae, en nous offrant une immersion concrète dans le monde du travail.

Mon stage se déroule chez **exaegis**, une entreprise que je présenterai plus en détail dans la section suivante.

Ce rapport de stage a pour but de mettre en évidence plusieurs dimensions de l'environnement professionnel. Dans un premier temps, une présentation de l'entreprise sera proposée, puis nous aborderons l'organisation de son environnement de développement. Enfin, nous approfondirons les différentes missions réalisées tout au long du stage. Le rapport s'achèvera par une conclusion synthétique, suivie d'une annexe contenant les fiches de procédure qui m'ont permis de mener à bien l'ensemble de mes missions.

1.1. Présentation globale de l'entreprise

Fondée en 2012, exaegis est une société française spécialisée dans l'évaluation et la gestion des risques des entreprises du secteur numérique. Elle analyse notamment la solidité financière, la gouvernance et la pérennité des fournisseurs de solutions IT et des éditeurs de logiciels, afin de leur attribuer des notations reconnues par les acteurs du marché.

Pour mener à bien ses missions, exaegis s'appuie sur plusieurs pôles d'activités :

- **exaegis Markess (Mon service)** : cette branche se consacre aux études de marché et à l'analyse des tendances du secteur numérique. Elle publie régulièrement des rapports et offre des services de veille stratégique, aidant ainsi les entreprises à anticiper l'évolution de leur environnement.
- **Corporate Development** : ce pôle intervient en conseil et en accompagnement stratégique, en particulier sur les problématiques de gouvernance, de financement et d'organisation. Son objectif est de renforcer la compétitivité et la stabilité des acteurs numériques en leur proposant des solutions adaptées à leurs besoins.
- **exaegis** (activité principale) : il s'agit du noyau historique de l'entreprise, dédié à la notation et à l'évaluation de la fiabilité des acteurs du digital. Ses équipes assurent la continuité d'activité et la gestion des risques des entreprises clientes, en leur délivrant des labels et des notations reconnus sur le marché.

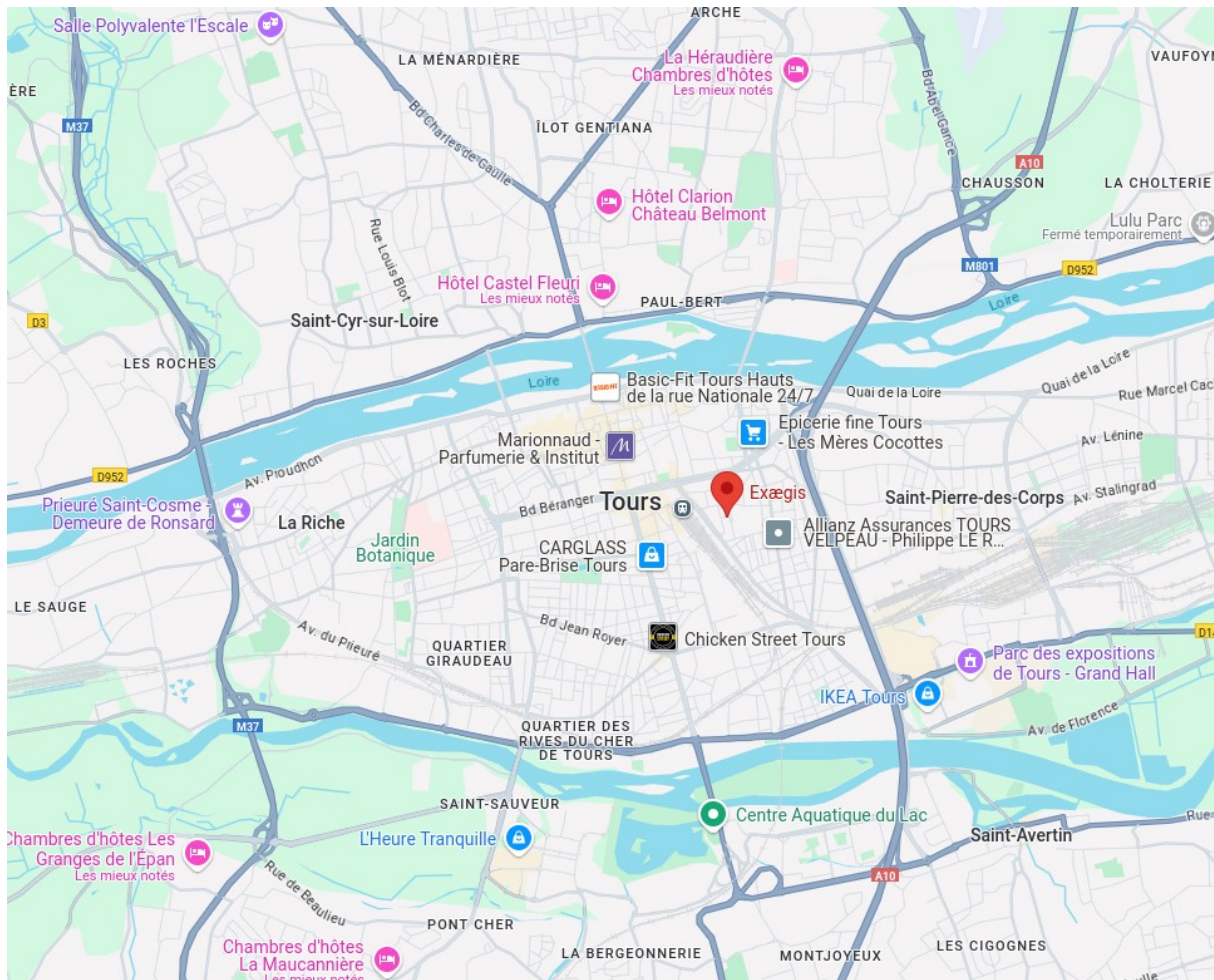
exaegis.
markess

exaegis.

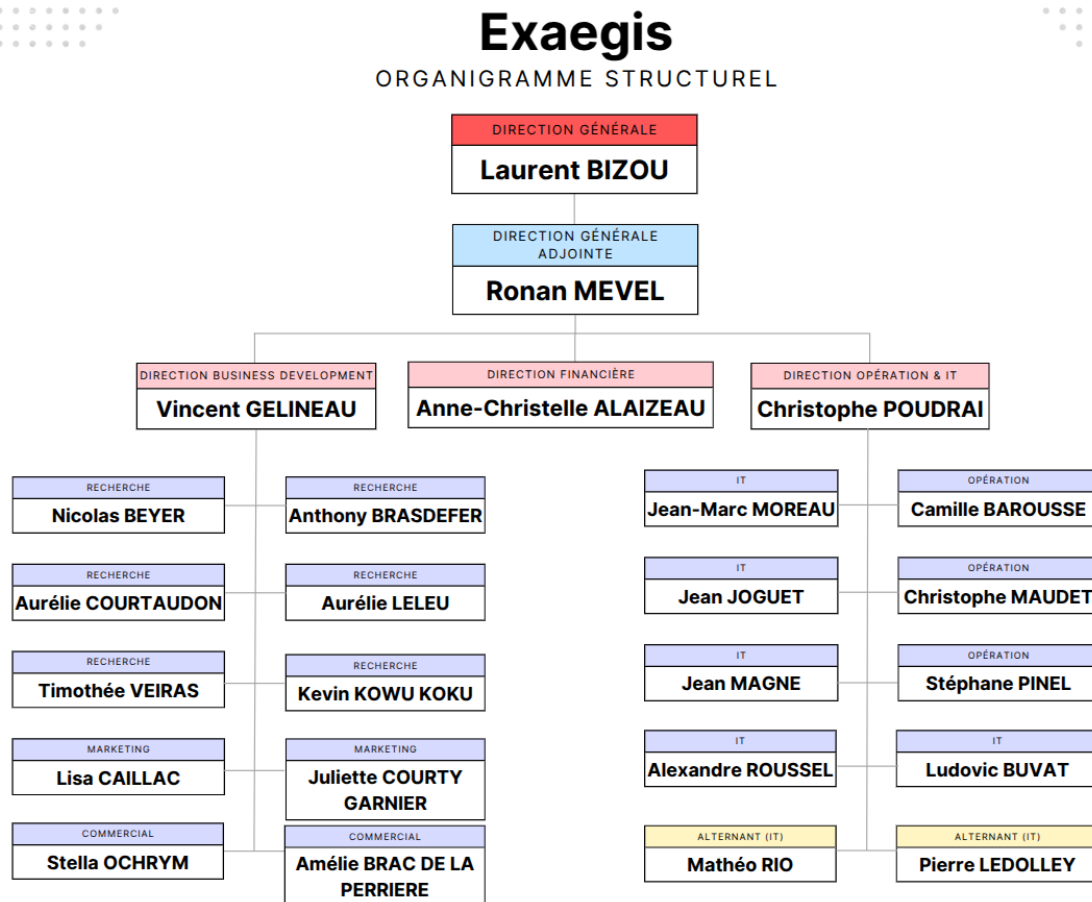
exaegis.
corporate development

1.2. Situation géographique

Adresse : 4 Rue Albert Dennerly, 37000 Tours



1.3. Organigramme hiérarchique et fonctionnel



1.4. Environnement matériel et logiciel de l'entreprise

Dans un premier temps voici mon environnement de travail sur le quel je réalise mes missions (un ordinateur fixe, clavier, souris) :



De plus le système d'exploitation que j'utilise est Ubuntu basé sur Linux.
Les logiciels généraux que je peux utiliser sont :

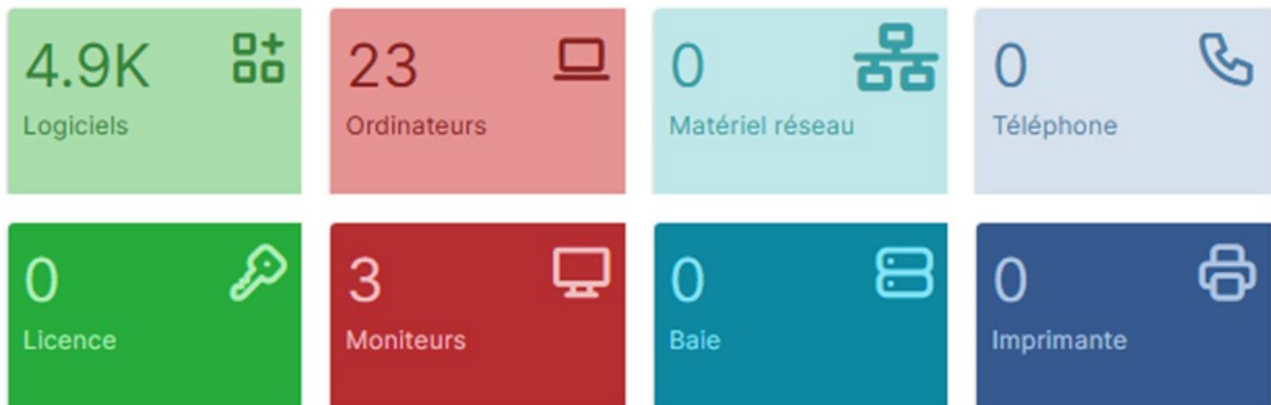
- Messagerie Outlook de l'entreprise avec un mail professionnel
- Teams pour communiquer avec les membres de l'entreprise
- Lucca permettant de gérer les absences, tout ce qui est RH (comme les congés, notes de frais, et gestion des talents), etc.

A propos de l'entreprise

Chaque membre de l'entreprise possède son pc portable fourni par l'entreprise.

Les développeur sont sous Linux et le reste sous Windows

L'entreprise possède un NAS, un AD ainsi qu'une multitude de logiciels tels que IntelliJ, Visual Studio Code, Jira, GitLab, Git etc.



Ci-dessus, un extrait du GLPI de l'entreprise(seulement les logiciels et les serveurs sont présents ici)

2. Missions

2.1. Mission n°1 : Correction de code de scraping sur le projet exaetrack_scraping_webext

2.1.1. Présentation du projet

Ce projet met en œuvre les technologies suivantes :

- Langage principal : Java
- Framework de développement : Spring-boot
- Gestionnaire de dépendances : Maven

Scraping : « Le scraping définit de façon générale une technique permettant d'extraire du contenu(des informations) d'un ou de plusieurs sites web de manière totalement automatique. Ce sont des scripts, des programmes informatiques qui sont chargés d'extraire ces informations » → <https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203521-scraping-definition-traduction/>

Objectif du projet : Récupérer des documents administratifs (délibérations, vote du budget, actes administratifs, ...) renseignés sur les sites des différentes collectivités françaises. Après analyse par une IA, les documents récupérés vont permettre aux entreprises abonnées au service de l'entreprise d'anticiper la mise sur marché public de projets liés au secteur du numérique, leur offrant un avantage concurrentiel et un temps pour préparer au mieux leur réponse. Le projet sert aussi à d'autres collectes sur le web tel que des offres d'emploi ou annonces diverses.

Mon objectif personnel dans cette mission est de consulter les tickets JIRA présentant certains problèmes rencontrés au cours de différents scraping et de les résoudre. (Un ticket par collectivité) De plus, ce projet ainsi que chaque ticket n'ont pas de « deadline » c'est à dire qu'on ne peut pas savoir combien de temps on va passer sur chaque problème.

The screenshot displays the JIRA interface for the EXAETRACK project. The main area shows a list of tickets under the 'Tickets en fait' section, with a progress bar indicating 91% completion. The tickets are organized by priority and status. The right sidebar shows the 'Champs épinglés' section with details for the selected ticket, including the assignee (Non assigné), reporter (Ludovic Buvat), and development status (Créer une branche, Créer un commit).

Ticket ID	Description	Priority	Status
XTRCK-34	scraping : utiliser le transport client pour webext	3	À FAIRE
XTRCK-8	Boamp : rapprochement erroné sur une scraping : utiliser le transport client pour webext	3	À FAIRE
XTRCK-35	OCR Tesseract	5	À FAIRE
XTRCK-105	scraping département Oise	2	TO REVIEW
XTRCK-218	Scraping métropole Strasbourg	3	TO REVIEW
XTRCK-227	Scraping ville Le Mans	3	À FAIRE
XTRCK-241	Scraping départ. Haute Saône	3	À FAIRE
XTRCK-256	Scraping départ. Seine Maritime	3	À FAIRE
XTRCK-257	Scraping départ. Somme	3	À FAIRE
XTRCK-285	Scraping région Bourgogne	3	À FAIRE
XTRCK-303	Scraping département Martinique	3	À FAIRE
XTRCK-312	Scraping département Lot	3	À FAIRE
XTRCK-314	Scraping département Saône et Loire	3	À FAIRE
XTRCK-318	Scraping département Val d'Oise	3	À FAIRE

2.1.2. Ressources

Pour ce projet plusieurs ressources sont mises à notre disposition :

- **ElasticSearch** (ou ES) est un moteur de recherche permettant de stocker des documents ou des données. C'est ici que seront déposés les documents administratifs récupérés, affectés d'un ID.
- **Kibana** est une plateforme permettant de consulter le contenu d'ES, les données affichées sont brutes et il n'est pas possible d'appliquer de filtres de recherche.
- **Poc-Ia**, tout comme Kibana, est une API permettant d'effectuer une recherche de documents parmi ceux présents sur ElasticSearch. La différence concrète est la possibilité d'appliquer des filtres de recherche comme la date, le nom, le type de document, un mot-clef, etc.
- **SonarQube** est un outil d'analyse de code statique qui permet de mettre en exergue les failles potentielles d'un code nouveau (bugs, vulnérabilités, sécurité, ...)
- **GitLab** est une plateforme DevOps open source qui réunit la gestion du code source, l'intégration continue (CI), la livraison continue (CD) et de nombreuses fonctionnalités de collaboration pour les équipes de développement. En unifiant toutes les étapes du cycle de vie logiciel, GitLab facilite la collaboration, l'automatisation et la gestion de projets, permettant ainsi aux organisations de gagner en efficacité et en rapidité dans le déploiement de leurs applications. Le GitLab de l'entreprise est donc relié directement à Jira ce qui facilite la création de branche / inspection de modifications etc.
- **IntelliJ** (ou IntelliJ IDEA) est un IDE (Integrated Development Environment) développé par JetBrains. Conçu principalement pour le développement Java, il prend également en charge de nombreux autres langages grâce à des plugins.
- **JSoup** est une bibliothèque Java qui facilite l'analyse, la manipulation et l'extraction d'informations au sein de pages HTML. Elle permet d'effectuer des opérations de parsing, de navigation dans le DOM et de nettoyage du code HTML, ce qui en fait un outil pratique pour le web scraping ou le traitement de données issues de sites web.
- **Selenium** est un ensemble d'outils d'automatisation et de test pour les navigateurs web. Il permet de simuler les actions d'un utilisateur (clics, saisie de texte, navigation, etc.) dans différents navigateurs, facilitant ainsi la mise en place de tests fonctionnels et la validation de la compatibilité entre diverses plateformes.

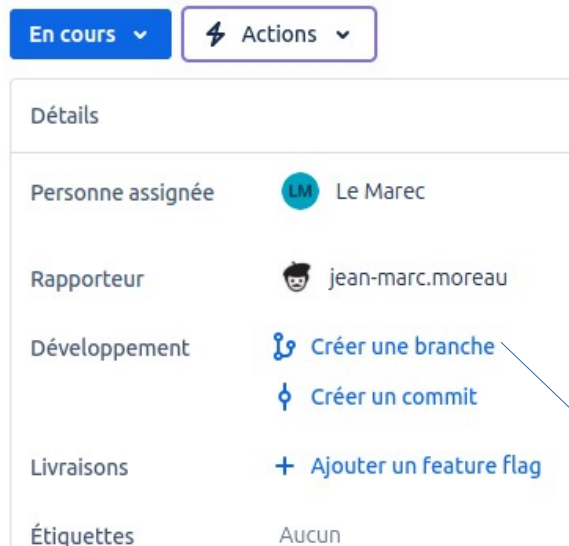
JSoup	Selenium
Syntaxe facile à appréhender, efficience dans la sélection des balises	Fonctionne bien même en présence de scripts JavaScripts
Ne fonctionne pas si le code HTML est construit par des scripts JavaScripts	Syntaxe moins évidente, devient compliqué à manipuler en cas de mauvaise structure du code HTML

2.1.3. Gestion de projet

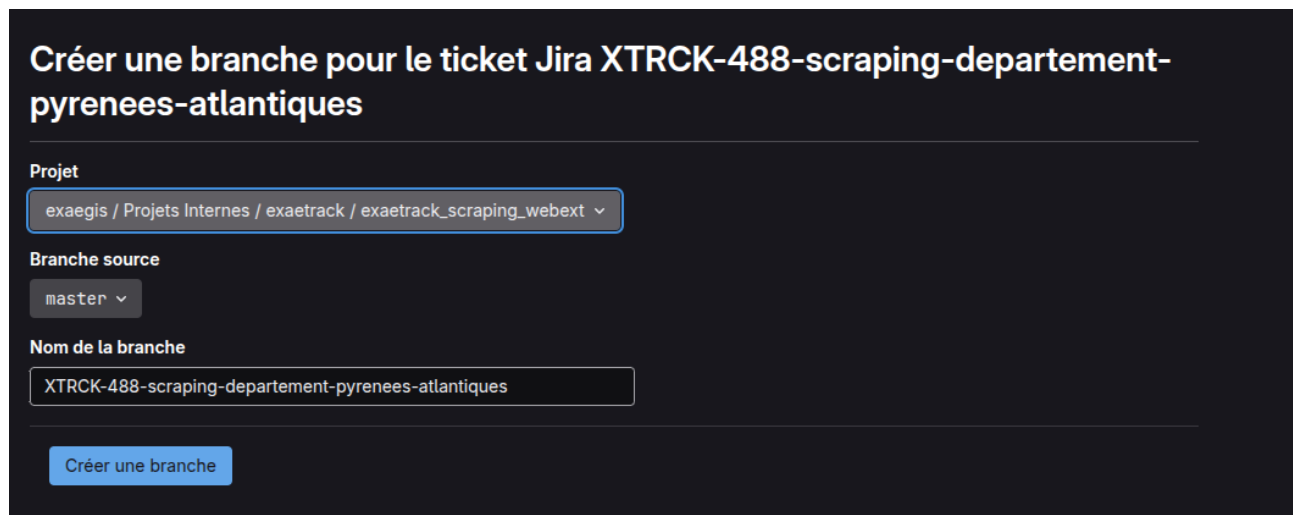
Comme vu ci-dessus le projet s'organise en différents **tickets** (un par collectivité).

Voici donc le **fonctionnement** de ces derniers :

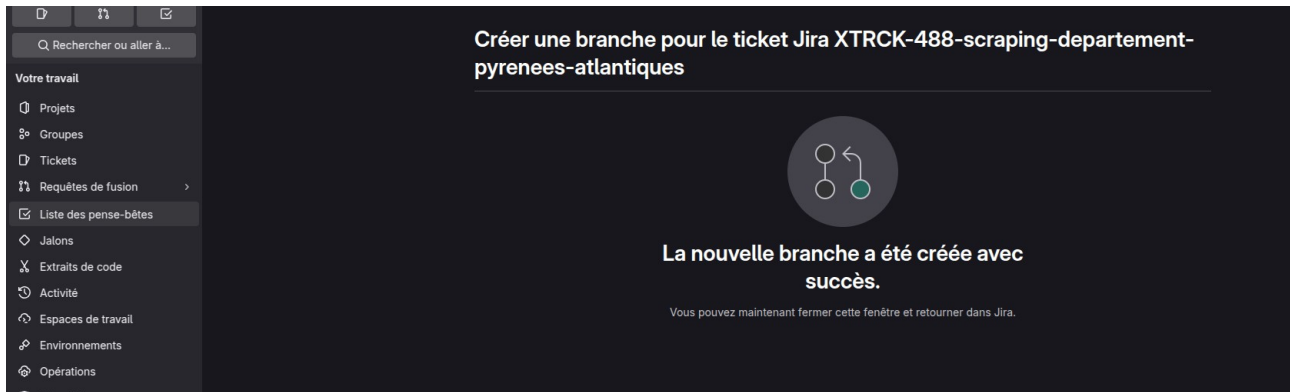
Dans un premier temps, on choisit le ticket à traiter puis on se **l'assigne** et on passe son **état** à « *En cours* »



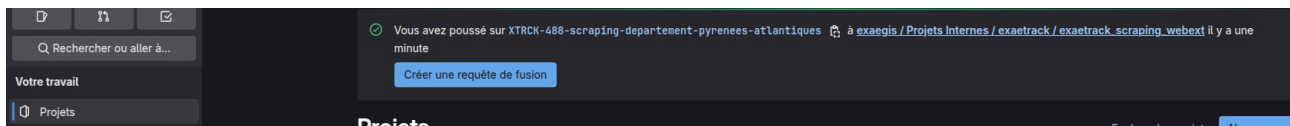
Ensuite, on crée une branche en cliquant sur « **Créer une branche** » ce qui nous amène sur GitLab :



On sélectionne le projet « **exaetrack_scraping_webext** » puis on clique sur « **Créer une branche** »



Une fois la branche créée on clique sur **Projets** dans la section à gauche



Et on clique sur « **Créer une requête de fusion** » (ce qui va permettre au **rapporteur** (personne faisant la revue de code et les tests) d'accepter cette requête et de **merger** la branche de la collectivité sur la branche **master** si tous les tests sont fonctionnels.)

Ensuite une fois qu'on pense avoir terminé les corrections, on **commit** puis **push** les modifications sur notre branche puis on se rend sur **Jira** pour passer l'état du ticket à « *To review* » et on l'assigne à notre rapporteur qui va s'occuper comme indiqué ci-dessus de tester la branche.

Ensuite, il y a deux cas :

- 1 – Si tous les tests passent, le rapporteur passe l'état du ticket à « *Terminé* » et me l'assigne pour comprendre que c'est moi qui ai traité le ticket.
- 2 – Si il y a une erreur, il me réassigne le ticket en passant l'état du ticket à « *Reviewing* » avec un commentaire prouvant l'erreur et je dois réexaminer le problème et essayer de corriger le code.

2.1.4. Collectivités corrigées

L'entreprise a mis en place deux proxys distincts pour effectuer du scraping sur les sites des collectivités : SQUID et TOR. Les collectivités sont réparties en plusieurs listes, déterminant le proxy utilisé dans le cadre du projet :

1. **Liste par défaut** : Les collectivités sont assignées au proxy **TOR**.
2. **Liste 1** : Les collectivités figurant sur cette liste sont accédées **sans proxy**.
3. **Liste 2** : Les collectivités de cette liste utilisent le proxy **SQUID**.

Cependant, un problème majeur est apparu : de plus en plus de collectivités bloquent l'accès via le proxy **TOR**, entraînant des dysfonctionnements et des échecs fréquents dans les opérations de scraping.

Accueil

Service et version de la demande

service (jobsall/jobsapec/jobsi)	Veillez indiquer un service.
version (1.0)	Veillez indiquer la version du service désiré.

Paramètres de la demande

returnType (inline,async,topic)	**Veillez indiquer le retour souhaité (inline,async,topic).
url (https://www.google.com)	*Veillez indiquer l'url du site a scraper.
offerCount (true/false)	Veillez indiquer si webext doit récupérer le nombre d'offres d'emploi des entreprises.
countyNumbers	Veillez indiquer les numéros de départements (séparés par des virgules) (tous si vide).
cnyNames	Veillez indiquer les noms d'entreprise (séparés par des virgules)
cnySiren	Le siren d'une entreprise
Collectivité	Le Nom de la collectivité (EGov)
Début	La date de début de collecte (EGov) (jj/mm/aaaa)
Fin	La date de fin de collecte (EGov) (jj/mm/aaaa)
headOfficeOnly (true/false)	Indique si la recherche entrep/étab se limite au siège.
mapSirenCny (siren;cny;siren;	Liste siren1;cny1;siren2;cny2;... (trademarks)
Lancer le scraping	

Voici ci-dessus la page de lancement du scraping où l'on doit renseigner :

- Le service (egov)
- La version (1.0)
- Le returnType (async)
- Le nom de la collectivité (ex : STRASBOURG)
- La date de fin et de début

Métropole de Strasbourg (1h~):

✓	XTRACK-218	Scraping métropole Strasbourg	=	LM	TERMINÉ(E) ✓
---	------------	-------------------------------	---	----	--------------

Problème rencontré :

Problème de connexion au proxy ainsi que non détection des éléments du DOM (mauvais nom de balises HTML) .

Résolution du problème :

Pour résoudre le problème du proxy, j'ai juste ajouté Strasbourg dans la liste du proxy Squid (liste 2).

Pour ce qui concerne la non détection des éléments du DOM, beaucoup de site sont rénovés et par conséquent ils décident de changer leurs balises HTML ce qui causes des problèmes dans notre code. J'ai donc simplement changé les noms de divisions dans le code en allant inspecter le site de la collectivité de Strasbourg.

Région de Bourgogne-Franche-Comté (30m~):

✓	XTRACK-295	Scraping région Bourgogne	=	LM	TERMINÉ(E) ✓
---	------------	---------------------------	---	----	--------------

Problème rencontré :

La collectivité avait changé d'URL

Résolution du problème :

Récupération de la nouvelle URL et modification dans le code.

Département du Val D'Oise (1h~) :

✓	XTRACK-318	Scraping département Val d'Oise	=	LM	TERMINÉ(E) ✓
---	------------	---------------------------------	---	----	--------------

Problème rencontré :

Problème d'accès à une des URL

Problème d'accès à différents éléments du DOM

La date des délibérations récupérée étant nulle, les documents n'étaient pas enregistrés dans ES(ElasticSearch).


Résolution du problème :

Modification de l'URL

Changement des noms de div dans le DOM du site de la collectivité

On voit que dans le premier code ci dessous on récupère les éléments « li.item » de l'élément « ul.liste » ce qui ne fonctionnait plus donc j'ai du adapter en récupérant d'abord une div qui a pour id « par1572 » pour ensuite récupérer tout les éléments « li » de classe « c-ressources__item » qui sont présent dans cette div.

```
boolean aucunDoc = true;  
Elements elementsLi = page.selectFirst("ul.liste").select("li.item");
```



```
boolean aucunDoc = true;  
Element div = page.getElementById("par1572");  
Elements elementsLi = div.selectFirst("ul.c-ressources__list").select("li.c-ressources__item");
```

Formatage de la date de délibération pour ne plus qu'elle soit nulle.

```
String[] lienSplit = lienDelib.split("/");  
Calendar calendar = Calendar.getInstance();  
calendar.set(Integer.parseInt(lienSplit[4]), Integer.parseInt(lienSplit[5])-1, 1);  
Date dateDelib = calendar.getTime();
```

Le code ci-dessus split chaque élément « / » d'une date qui est sous format dd/MM/yyyy

Puis on instancie une variable de type Calendar puis on lui attribut le jour, le mois et l'année et on récupère la date formé dans une variable dateDelib de type Date.

Département Bouche-Du-Rhône (4h~):

<input checked="" type="checkbox"/>	XTRCK-368	Scraping département Bouches du Rhône				TO REVIEW
-------------------------------------	-----------	---------------------------------------	--	--	--	-----------

Problème rencontré :

On constate que les docs ne sont pas trouvés (on a un code 404 sur le download du doc). Mais si on le télécharge à la main via un navigateur, alors après l'appli le trouve ...

L'explication semble être la suivante :

- A la base, on a un scraping webdelib standard ; mais il y a de l'enrobage autour :
- on a d'abord un proxy de cache (genre squid ou équivalent)
- puis un filtre F5 de répartition réseau
- puis le serveur webdelib.

Quand on demande un doc, le F5 contrôle les cookies ; il a au préalable envoyé sur la page web un script de chiffage des cookies. Si on est en JSoup, et qu'on n'a pas explicitement chiffré les cookies, ça ne colle pas et on se prend un 404. Si on est dans un navigateur, le js de chiffage est exécuté et on récupère le document.

Si on a téléchargé le fichier à la main depuis un navigateur, alors le fichier est dans le cache, on ne passe pas par le F5 et le contrôle des cookies, et on récupère le document avec l'application en JSoup.

=> on a donc 2 solutions dans ce cas :

- reproduire en java le chiffage de cookies fait en js (pas très fin, il y a plusieurs codes de chiffrements, ça n'est pas toujours le même script envoyé sur la page)
- refaire le scraping en ignorant le code mutualisé "webdelib", et en refaisant tout à partir de zéro en selenium (en espérant que le F5 ne détecte pas le selenium)

Ci-dessus, une capture d'écran du ticket Jira de la collectivité.

Résolution du problème :

Choix de la deuxième solution qui est de refaire tout le code à partir de zéro en utilisant Selenium.

```
private void scrapeDelib(String cookies, Set<String> listeHashDoc, Map<String, String> headers, Map<String, Long> totaux) { 1 usage  trystan *
    driverGlob = DriverSettings.loadDriver(EGovConst.URL_DELIB_BOUCHE_DU_RHONE, this.serviceEgov);

    driverGlob.switchTo().frame( nameOrId: "iframeWebdelib");

    attente( millisecondes: 2000);

    WebElement inputStartDate = driverGlob.findElement(By.xpath( xpathExpression: "//input[@name='dtdebut']"));
    inputStartDate.sendKeys(formatDate(startDate));

    WebElement inputEndDate = driverGlob.findElement(By.xpath( xpathExpression: "//input[@name='dtfin']"));
    inputEndDate.sendKeys(formatDate(endDate));

    WebElement searchButton = driverGlob.findElement(By.xpath( xpathExpression: "//input[@value='Rechercher']"));
    searchButton.click();
    try {
        WebElement tbody = driverGlob.findElement(By.xpath( xpathExpression: "//tbody"));
        List<WebElement> deliberations = tbody.findElements(By.tagName("tr"));
        int i = 0;
        for (WebElement deliberation : deliberations) {
            if (i == 50) {
                i = 0;
                WebElement tfoot = driverGlob.findElement(By.xpath( xpathExpression: "//tfoot"));
                List<WebElement> nextPageButton = tfoot.findElements(By.tagName("a"));
                for (WebElement nextPageButtonElement : nextPageButton) {
                    WebElement img = nextPageButtonElement.findElement(By.tagName("img"));
                    if (img.getAttribute( name: "title").contains("suivante")) {
                        // Faire défiler la page jusqu'au bouton "Page suivante"
                        JavascriptExecutor js = (JavascriptExecutor) driverGlob;
```

```

        // Attendre un peu après le défilement pour s'assurer que l'élément est visible
        attente( millisecondes: 500);

        // Cliquer sur le bouton "suivante"
        nextPageButtonElement.click();
        break;
    }
}

}

List<WebElement> td = deliberation.findElements(By.tagName("td"));
List<WebElement> liens = td.get(1).findElements(By.tagName("a"));
String refDelib = td.get(2).getText();
String titreDelib = td.get(3).getText();
String date = td.get(4).getText();
Date dateDelib = ServiceDates.parseDate(date, new SimpleDateFormat( pattern: "dd/MM/yyyy"));
String idMain = null;
for (WebElement lien : liens) {
    String url = lien.getAttribute( name: "href");
    String urlDelib = EGovConst.URL_BASE_DOWNLOAD_BOUCHE_DU_RHONE + extractBetweenQuotes(url);
    if (idMain == null) {
        idMain = saveInfo(EGovConst.RECHERCHE_TYPE_DELIB, this.localAuthorityName, this.esIndexDocuments, this.esIndexDocumentsType, this.esNodeIp, this.esNodePort,
    } else {
        saveInfo(EGovConst.RECHERCHE_TYPE_DELIB, this.localAuthorityName, this.esIndexDocuments, this.esIndexDocumentsType, this.esNodeIp, this.esNodePort, idMain,
    }
}
i++;
}
} catch (Exception ex) {
    log.log(Level.INFO, EGovConst.AUCUN_0_DANS_LA_PLAGE, EGovConst.RECHERCHE_TYPE_DELIB);
}
}

```

Dans les captures d'écrans ci-dessus la ligne la plus importante est l'appel a la méthode `saveInfo` qui va permettre d'aller enregistrer directement dans ElasticSearch le document voulu en renseignant en paramètre notamment le lien du PDF, la date, le titre et la référence de la délibération. Le reste étant des paramètres par défaut.

Ville de Nîmes(4h~)

	XTRACK-456	Scraping ville Nîmes		Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ✓
--	------------	----------------------	--	----------	------------------	--------	--------------

Problème rencontré :

Non-récupération des budgets à cause du changement de leur site.

Résolution du problème :

Refonte du code de récupération des budgets pour le nouveau site. Ajout d'une méthode dans **ServiceDates** pour récupérer une liste d'années entre deux dates pour les collectivités ne disposant que de certaines années à vérifier.

Département du Lot(4h~)

	XTRACK-342	Scraping département Lot		Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ✓
--	------------	--------------------------	--	----------	------------------	--------	--------------

Problème rencontré :

Non-récupération des délibérations, Jsoup ne permettait plus de récupérer les données suite au changement du site qui a intégré du JavaScript.

Résolution du problème :

Reconstruction du code en utilisant Selenium à la place de Jsoup.

Département de la Haute-Saône(4h~)

	XTRACK-241	Scraping départ. Haute Saône		Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ✓
--	------------	------------------------------	--	----------	------------------	--------	--------------

Problème rencontré :

Aucune récupération de documents en raison de changements d'éléments sur le site nécessitant une refonte du code.

Résolution du problème :

Reconstruction complète du code en fonction des nouvelles structures du site.

Département de l'Oise(1h~)

	XTRACK-105	scraping département Oise	Le Marec	jean-marc.moreau	Low	TERMINÉ(E) ▼
--	------------	---------------------------	----------	------------------	-----	--------------

Problème rencontré :

- Problème de proxy (ajout nécessaire à Squid).
- Problème de gestion des dates causant des erreurs lors de la récupération.

Résolution du problème :

- Ajout d'Oise au proxy Squid.
- Refonte de la partie du code traitant les dates, notamment leur extraction dans les titres qui provoquaient des erreurs.

Département Saône-et-Loire(4h~)

	XTRACK-314	Scraping département Saône et Loire	Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ▼
--	------------	-------------------------------------	----------	------------------	--------	--------------

Problème rencontré :

- Erreur 403 liée au proxy.
- Changement de site rendant le code obsolète.

Résolution du problème :

- Passage par le proxy Squid.
- Refonte complète du code pour s'adapter au nouveau site.

Métropole de Rouen(1h~)

	XTRACK-478	Scraping métropole Rouen	Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ▼
--	------------	--------------------------	----------	------------------	--------	--------------

Problème rencontré :

Lors du clic sur "Rechercher les délibérations", un autre élément masquait le bouton.

Résolution du problème :

Ajout d'un clic sur le bouton d'acceptation des cookies avant d'exécuter l'action principale + des tests en cas de non présence du bouton de cookies dans certains cas.

Métropole de Villeurbanne(30m~)

	XTRACK-457	Scraping ville Villeurbanne	Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ▼
--	------------	-----------------------------	----------	------------------	--------	--------------

Problème rencontré :

Aucune récupération de documents car le lien récupéré était construit avec HTTP, hors, une redirection de port était effectué pour télécharger le document.

Résolution du problème :

- Passage des URL en HTTPS.
- Renommage du nom de colonne recherché (modification de Date en Date de la séance).

Département des Landes(15m~)

	XTRACK-487	Scraping département Landes	Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ▼
--	------------	-----------------------------	----------	------------------	--------	--------------

Problème rencontré :

Erreur 403 lors de l'accès au site.

Résolution du problème :

Passage sans proxy pour contourner les restrictions.

Département Pas-de-Calais(4h~)

	XTRACK-453	Scraping département Pas-de-Calais	Le Marec	jean-marc.moreau	Medium	TERMINÉ(E) ▼
--	------------	------------------------------------	----------	------------------	--------	--------------

Problème rencontré :

WebDelib n'était plus alimenté, rendant les données inutiles à récupérer car trop ancienne.

Résolution du problème :

Refonte complète du code en se basant sur le site de la collectivité avec les documents plus récents.

Département Val-de-Marne(30m~)

<input checked="" type="checkbox"/>	XTRACK-490	Scraping département Val de Marne	 Le Marec	 jean-marc.moreau	Medium	TERMINÉ(E) ✓
-------------------------------------	------------	-----------------------------------	--------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------	--------	--------------


Problème rencontré :

- Problème de proxy (erreur 403).
- Problème de récupération de la date cible des délibérations.

Résolution du problème :

- Passage par le proxy Squid.
- Correction du problème de récupération de la date.
- Modification de la méthode de parcours des délibérations.

Ville Le Mans(4h~)

<input checked="" type="checkbox"/>	XTRACK-227	Scraping ville Le Mans	 Le Marec	 jean-marc.moreau	Medium	TERMINÉ(E) ✓
-------------------------------------	------------	------------------------	--------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------	--------	--------------

Problème rencontré :

Aucun document récolté depuis 2022 (collectivité ayant effectué une refonte de leur site).

Résolution du problème :

Refonte complète du code pour s'adapter à la nouvelle structure.

Département du Doubs

XTRACK-484	Scraping département du Doubs	 Tristan Le Marec	 jean-marc.moreau	Medium	TERMINÉ(E) ✓
------------	-------------------------------	------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------	--------	--------------




Problème rencontré :

Le code était basé sur des URLS où l'on plaçait les années dedans, hors aucune vérification n'étaient faites si il n'y avait aucun document pour une nouvelle année (2025 ayant commencé par exemple) ce qui causé un crash.

Résolution du problème :

Ajout d'un test pour vérifier quelles dates sont bien présentes dans les documents.

Ville d'Angers (~30m)

XTRACK-454	Scraping ville Angers		 Trystan Le Marec	 jean-marc.moreau	Medium	TERMINÉ(E) 
------------	-----------------------	-----------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------	--------	------------------------------------------------------------------------------------------------




Problème rencontré :

Erreur 403 (Proxy).

Résolution du problème :

Passage par le proxy SQUID.

Département du Vaucluse(~4h)

XTRACK-319	Scraping département Vaucluse		 Trystan Le Marec	 jean-marc.moreau	Medium	TERMINÉ(E) 
------------	-------------------------------	--	----------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------	--------	------------------------------------------------------------------------------------------------

Problème rencontré :

Plus aucun documents récupérés.

Résolution du problème :

Refonte du code complet en récupérant les données sous format JSON.

Collectivité Européenne Département d'Alsace(~4h)

XTRACK-500	Scraping département Alsace		 Trystan Le Marec	 Trystan Le Marec	Medium	TERMINÉ(E) 
------------	-----------------------------	--	------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------	--------	--------------------------------------------------------------------------------------------------




Problème rencontré :

Documents de la collectivité pas encore récupérés

Résolution du problème :

Création du code à partir de 0.

Département Charente-Maritime(~4h)

XTRACK-503	Scraping département Charente-Maritime		 Trystan Le Marec	 Trystan Le Marec	Medium	TERMINÉ(E) 
------------	----------------------------------------	--	------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------	--------	--------------------------------------------------------------------------------------------------


Problème rencontré :

Documents de la collectivité pas encore récupérés

Résolution du problème :

Création du code à partir de 0.

Département Île-de-France(~30m)

XTRCK-477	Scraping région Ile de France		 Trystan Le Marec	 jean-marc.moreau	 Medium	TERMINÉ(E) ✓
-----------	-------------------------------	-----------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------	--------------

Problème rencontré :

Problème de proxy

Résolution du problème :

Ajout au proxy **Squid** pour contourner les restrictions.

Département Ile-et-Vilaine(~30m)

XTRCK-485	Scraping département Ile et Vilaine	 Trystan Le Marec	 jean-marc.moreau	 Medium	TERMINÉ(E) ✓
-----------	-------------------------------------	----------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------	--------------





Problème rencontré :

Problème de proxy

Résolution du problème :

Ajout au proxy **Squid** pour contourner les restrictions.

Département de l'Hérault, Département Mayenne, Ville de Rennes, Métropole de Rennes, Département Yonne (~5h)

XTRACK-484	Scraping département Hérault	 Trystan Le Marec	 jean-marc.moreau	 Medium	TERMINÉ(E) 
------------	------------------------------	----------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------

Problème rencontré :

Pour ces 5 collectivités, une méthode générique pour l'API **DematDOC** a été créée pour éviter d'écrire pleins de fois le même code. Or, l'API de **DematDOC** a changé le format de Json qu'on reçoit ce qui rendait le code de la méthode générique obsolète ainsi que les URLs d'appels de la méthode POST.

Résolution du problème :

Refonte d'une partie de la méthode générique **DematDOC** et mise à jour des URLs d'API pour assurer la récupération des données.

```
Document json = EGovDematDOC.getFirstJsonDematDOC(this.sessionData, postParams, urlFirstRequest);

EGovDematDOC.ContratParams params = new EGovDematDOC.ContratParams(EGovConst.URL_PARTDOWNLOAD_YONNE, EGovConst.URL_PRINCIPALE_YONNE,
    EGovConst.URL_NEXT_REQUEST_YONNE);

EGovDematDOC.ContratParamsInfos paramsInfoCollectivity = new EGovDematDOC.ContratParamsInfos(this.esDataParameters, this.sessionData,
    listeHashDoc, resultHashDoc, scrapingResult);

EGovDematDOC.getDelibsDematDOC( numberRequest: 1, paramsInfoCollectivity, json,
    keyDate: "CI_DATE_DEBUT_AFFICHAGE_PUBLIC", params, typeDoc, eGovGeneric: this);

if (i == 0) {

    log.log(Level.INFO, EGovConst.DEBUT_DE_RECHERCHE_0, EGovConst.RECHERCHE_TYPE_ARCHIVE);

    postParams.add("Délibération Commission permanente!");
    typeDoc = "archive";
    urlFirstRequest = String.format(EGovConst.URL_FIRST_REQUEST_YONNE, typeDoc);

}
```

Voici ci-dessous le point d'entrée de la méthode générique **DematDOC** appelée par l'une des collectivités par exemple Yonne ci-dessus où l'on va récupérer en paramètre un objet Document Jsoup récupéré par la méthode `getFirstJsonDematDOC()` qui va traiter la requête HTTP et qui contient le body json que l'on va analyser par la suite.

```

public static ScrapingResults getDelibsDematDoc(int numberRequest, ContratParamsInfos infosCollectivity, Document json, String keyDate, 6 usages
    ContratParams urls, String typeDoc, EGovGeneric eGovGeneric) throws IOException {

    Logger log = eGovGeneric.getLog();

    Element body = json.body();
    String jsonText = body.text();
    JSONObject jsonObject = new JSONObject(jsonText);

    JSONArray idNextDocs = jsonObject.getJSONArray( key: "nextDocsIds");

    // Récupération du tableau "documents"
    JSONArray documentsArray = jsonObject.getJSONArray( key: "documents");

    // Boucle à travers les documents
    for (int i = 0; i < documentsArray.length(); i++) {
        JSONObject docJSON = documentsArray.getJSONObject(i);

        // Récupération des valeurs essentielles
        JSONObject contentTitle = docJSON.getJSONObject( key: "values").getJSONObject( key: "OBJET");
        String titleDoc = contentTitle.getString( key: "displayValue");

        String href = docJSON.getString( key: "path");
        String linkDoc = urls.urlPartDownload() + href;
        String idDoc = String.valueOf(docJSON.getInt( key: "id"));
        // Vérification et récupération de la date du document
        if (docJSON.getJSONObject( key: "values").has(keyDate)) {
            JSONObject contentKeyDate = docJSON.getJSONObject( key: "values").getJSONObject(keyDate);
            String stringDate = contentKeyDate.getString( key: "displayValue");
            // Vérification si la date est dans l'intervalle demandé
            if (eGovGeneric.isBetweenDate(docDate, infosCollectivity.esDataParameters().getStartDate(), infosCollectivity.esDataParameters().getE
                // Formatage du nom du fichier
                String fileName = titleDoc + ".pdf";
                fileName = fileName.replace( target: " ", replacement: "_").replace( target: "°", replacement: "-");

                // Sauvegarde des informations du document dans ES
                eGovGeneric.saveInfo(
                    EGovConst.RECHERCHE_TYPE_DELIB,
                    infosCollectivity.esDataParameters().getLocalAuthorityName(),
                    infosCollectivity.esDataParameters().getEsIndexDocuments(),
                    infosCollectivity.esDataParameters().getEsIndexDocumentsType(),
                    infosCollectivity.esDataParameters().getEsNodeIp(),
                    infosCollectivity.esDataParameters().getEsNodePort(),
                    idMain: null,
                    infosCollectivity.listeHashDoc(),
                    infosCollectivity.sessionData().getCookiesInString(),
                    infosCollectivity.sessionData().getHeaders(),
                    docDate,
                    urls.urlPrincipale(),
                    linkDoc,
                    idDoc,
                    titleDoc,
                    zipped: false,
                    fileName,
                    infosCollectivity.resultHashDoc());
            } else {
                log.log(Level.INFO, msg: "Ce document n'est pas présent dans la plage de date demandée");
            }
        } else {
            log.log(Level.WARNING, msg: "Clé {0} on trouvée dans les valeurs du document ID: {1} ", new Object[]{keyDate, idDoc});
        }
    }
}

```

Ci-dessous, la méthode getNextJsonDematDOC() va s'occuper de récupérer le Json des documents suivant en exécutant une méthode POST prenant en paramètre un JSONArray des IDS des documents pour ensuite appeler une nouvelle fois la méthode getDelibsDematDOC comme indiquez ci-dessus afin de les enregistrer dans ElasticSearch.

```
application.properties  EGovDematDOC.java x
29  public class EGovDematDOC { 35 usages  Noah Guillard +1
154  private static Document getNextJsonDematDOC(JSONArray idNextDocs, SessionData sessionData, String constUrlRequest)
155
156      OkHttpClient client = new OkHttpClient();
157
158      // Convertir le tableau d'IDS en JSON brut
159      JSONArray jsonArray = new JSONArray(idNextDocs);
160      String jsonBody = jsonArray.toString();
161
162      // Construire le corps JSON de la requête
163      RequestBody body = RequestBody.create(jsonBody, MediaType.get("application/json"));
164
165      // Construire la requête avec les headers
166      Request.Builder requestBuilder = new Request.Builder()
167          .url(constUrlRequest)
168          .post(body)
169          .addHeader(name: "Content-Type", value: "application/json")
170          .addHeader(name: "Accept", value: "application/json");
171
172      // Ajouter les headers de la session
173      for (Map.Entry<String, String> entry : sessionData.getHeaders().entrySet()) {
174          requestBuilder.addHeader(entry.getKey(), entry.getValue());
175      }
176
177      Request request = requestBuilder.build();
178
179      // Exécuter la requête
180      try (Response response = client.newCall(request).execute()) {
181          if (!response.isSuccessful()) {
182              throw new RuntimeException("Réponse HTTP non valide : " + response.code());
183          }
184
185          // Transformer la réponse en Document Jsoup
186          String responseBody = response.body().string();
187          return Jsoup.parse(responseBody);
188      }
189  }
```

2.1.5. Documentation réalisé :

Voici une [documentation](#) que j'ai réalisé pour le scraping : [Lien](#)

2.2. Mission n°2 : Analyse et récupération de données via l'API Sewan

2.2.1. Présentation de la mission

La mission consiste à développer un script en Python permettant d'analyser et d'exploiter l'API de Sewan. Cette API contient les données d'un prestataire spécialisé dans la gestion d'équipements de téléphonie. Mon objectif est d'automatiser la récupération et le traitement des informations des clients de ce prestataire, notamment :

- **Les adresses des clients**
- **Les équipements et services installés**

Ces données seront ensuite consolidées et répertoriées dans un fichier CSV afin de simplifier leur gestion et leur exploitation.

2.2.2. Pourquoi utiliser l'API ?

Actuellement, la saisie des informations dans le fichier CSV est effectuée manuellement par chaque client. Cependant, ce processus est jugé fastidieux et source d'erreurs. De nombreux clients s'en plaignent en raison :

- Du **temps nécessaire** pour remplir les données à la main
- Des **risques d'erreurs humaines** dans la saisie
- De la **perte de temps** engendrée par les corrections et les vérifications

L'utilisation de l'API permet d'automatiser cette tâche en garantissant une collecte plus rapide, fiable et efficace des informations. Cette solution apporte une **meilleure structuration des données**, tout en **réduisant considérablement les interventions manuelles**.

2.2.3. Problème rencontré

Cette mission avait été laissée de côté pendant plus d'un an, rendant la documentation que le prestataire nous avait envoyée assez obsolète, notamment au niveau des identifiants et des méthodes de connexion à l'API.

J'ai donc dû contacter ce prestataire afin qu'il me débloque l'accès, ce qui a été fait très rapidement. J'ai ainsi pu commencer à analyser l'API et récupérer des informations.

2.2.4. Présentation du script

Tout d'abord, j'ai décomposé le code en 4 fichiers :

- La classe « Client »
- Un fichier qui comporte les méthodes d'appels aux WebServices de l'API
- Un fichier de config où sont renseignés les informations de connexion à l'API
- Un fichier main qui va exécuter les méthodes et ajouter les lignes au CSV.

Extrait de la classe Client

```
class Client:
    def __init__(self, siren):
        self.siren = siren
        self.name = None
        self.city = None
        self.zipcode = None
        self.address = None
        self.services = []

    #region Setters...

    #region Getters...

    #region Méthodes
    def printInfo(self):
        print(f"SIREN : {self.siren}")
        print(f"Nom : {self.name}")
        print(f"Adresse : {self.address}")
        print(f"Ville : {self.city}")
        print(f"Code Postal : {self.zipcode}")
        print("Services :")
        for service in self.services:
            print(f"    * {service}")

    def newLine(self):
        return [self.siren, self.name, self.city, self.zipcode, self.address, ";".join(self.services)]
    #endregion
```

Extrait du fichier getInfosJsonSophia

```
import configparser
import sys
import requests

config = configparser.ConfigParser()
config.read("config.ini")

USER_AGENT = config["general"]["USER_AGENT"]

API_TOKEN = config["general"]["API_TOKEN"]

LANG = config["general"]["LANG"]

SOPHIA_JSON_GATEWAY = config["general"]["SOPHIA_JSON_GATEWAY"]

HEADERS = {
    "User-Agent": USER_AGENT,
    "Authorization": f"bearer {API_TOKEN}",
    "Accept-Language": LANG,
}

> def getOwnerID(siren): ...

> def getServices(owner_id): ...

> def getinfoClient(owner_id): ...
```


Voici la fonction `getServices` qui va permettre de récupérer tout les équipements/services d'un client en plaçant en paramètres son **owner_id**

```
def getServices(owner_id):
    """
    Description :
    -----
    Cette fonction récupère le Json contenant tout les Services d'un client à partir
    de son owner_id

    Paramètres :
    -----
    owner_id : str
    | owner_id du client ciblé.

    Retourne :
    -----
    json
    | Json contenant les services d'un client.
    """
    url = (
        f"{SOPHIA_JSON_GATEWAY}"
        "?service=sophia.service.PhoneManagement"
        "&method=get_all"
        f"&owner_id={owner_id}"
    )

    response_search = requests.get(url, headers=HEADERS, timeout=10)

    if response_search.status_code != 200:
        sys.stderr.write(f" => HTTP error {response_search.status_code} on search.\n")
        sys.stderr.write(f" => Body {response_search.content}\n")
        sys.exit(2)
    rjson = response_search.json()

    code = int(rjson["code"])
    if code < 200 or code >= 300:
        # Search failed
        sys.stderr.write(
            f" => Search failed with code {rjson['code']} : {rjson['msg']}\n"
            "(see Webservice response for more details)\n"
        )
        sys.exit(2)

    return rjson["result_object1"]
```

Extrait du fichier main

L'utilisateur du script doit renseigner le SIREN du client recherché ainsi que le nom du csv dans le quel les informations vont être ajoutés.

```
biSewanFinal.py > find_info
from Client import Client
from getInfosJsonSophia import *
import csv

def find_info():
    siren = input("Entrez le SIREN de l'entreprise : ")
    nom_csv = ""
    # On initialise un client à partir de son SIREN
    client = Client(siren)

    # On récupère son owner_id
    owner_id = getOwnerID(siren)

    # Grâce à l'owner_id on récupère tout ses services
    results_array = getServices(owner_id)

    # On les parcourt puis on lui ajoute
    for service in results_array:
        unService = f"{service['phm_name']} ( {service['serial']} | {service['mac_address']})"
        client.addServices(unService)

    # On récupère un dictionnaire contenant toutes les infos du client
    results_array = getinfoClient(owner_id)

    # On set toutes les infos du client |
    client.setName(results_array["name"])
    client.setZipcode(results_array["default_address"]["zipcode"])
    client.setAddress(results_array["default_address"]["line1"])
    client.setCity(results_array["default_address"]["city"])

    client.printInfo()

    try:
        # On récupère le fichier csv puis on ajoute une ligne avec les informations du client
        with open(f"{nom_csv}", "a", newline="", encoding="utf-8") as fichier:
            writer = csv.writer(fichier)
            writer.writerow(client.newLine())
    except FileNotFoundError as e:
        print(f"Impossible de trouver le fichier")
    except Exception as ex:
        print(f"Une erreur est survenue : {ex}")
```


3. Mesure de réussite du projet

Concernant l'évaluation de mes missions, mon stage a été divisé en deux parties principales :

- **Première mission : résolution de tickets**

L'objectif principal était de résoudre le plus de tickets possible. J'ai avancé efficacement dans cette tâche en traitant un grand nombre de demandes et en améliorant ma capacité à analyser et corriger des problèmes techniques rapidement. Cependant, aucun objectif chiffré précis n'avait été défini, ce qui signifie que l'évaluation de ma performance reposait principalement sur ma progression et ma capacité à gérer différentes problématiques.

- **Seconde mission : développement spécifique**

Pour cette mission, j'avais des attentes clairement définies et des tâches précises à accomplir. J'ai réussi à livrer les résultats attendus en respectant les consignes et les exigences techniques, ce qui m'a permis de mesurer concrètement ma réussite sur cet aspect du projet.

4. Conclusion

Au cours de mon stage d'un mois en tant que développeur, j'ai eu l'opportunité d'acquérir et de renforcer des compétences essentielles dans le domaine du développement logiciel. Cette expérience enrichissante m'a permis d'approfondir mes connaissances aussi bien pratiques que théoriques.

Durant mon stage, j'ai été impliqué dans diverses tâches liées à la conception, au développement et à l'optimisation d'applications. J'ai appris à appliquer les bonnes pratiques de programmation, à structurer mon code de manière efficace et à collaborer avec l'équipe technique pour assurer la qualité des livrables. En parallèle, j'ai développé mes compétences en gestion du temps, en recherche et en autonomie, car la résolution de bugs et l'implémentation de nouvelles fonctionnalités nécessitent des investigations précises et une analyse approfondie.

J'ai également appris à documenter mon travail de manière rigoureuse, un aspect essentiel pour assurer la maintenance et l'évolution des projets. Ce stage m'a offert une vision concrète des défis et des responsabilités d'un développeur, notamment l'importance d'écrire un code propre, optimisé et sécurisé pour garantir la stabilité et la performance des applications.

Je suis reconnaissant envers l'équipe qui m'a accueilli et guidé tout au long de cette expérience. En conclusion, mon stage en tant que développeur chez Exaegis a été une expérience particulièrement enrichissante. J'ai acquis des compétences précieuses et renforcé ma compréhension du développement logiciel.