



Rapport de stage

Réalisé par Etienne POURCHON

Du 30 Mai 2023 au 30 Juin 2023

Maître de stage : Nicolas MACHADO

Enseignant référent : Bastien DUFOUR

# AUTOMATISATION DE TESTS

1ère année BTS SIO SLAM

Organisme d'accueil : Rectorat de l'académie de Clermont-Ferrand – site Gergovia –  
43 Bd François Mitterrand – 63000 Clermont-Ferrand

Année scolaire : 2022-2023

## Table des matières

Remerciements :	2
<b>1. Contexte</b>	3
1.1 Présentation du Rectorat.....	3
1.2 Présentation de la DSI.....	3
<b>2. Mes missions/déroulement</b>	3
2.1 Début des travaux, la découverte du travail .....	3
2.2 Immersion et mise à niveau.....	4
2.3 La Mission .....	4
2.4 Description du procédé de développement .....	5
2.5 Outils utilisés.....	10
<b>3 Conclusion</b>	11
Annexe 1 – Organigramme de la DSI.....	12
Annexe 2 – Organigramme de l'ENSN.....	13
Annexe 3 – Exemples de pages.....	14

### Remerciements :

- Xavier Berchebru, chef de division : de m'avoir fait confiance pour ce poste et m'avoir accordé du temps pour me présenter à la division.
- Nicolas Machado, automaticien : pour son approche pédagogique qui a rendu chaque aspect du stage agréable et intéressant à faire.
- Le pôle BEE / pôle cœur de métier et orientation : pour son aide durant tout le long de ma mission sur le projet BAN.
- Toute l'équipe de la DSI : pour son accueil et son soutien durant tout mon stage.

## 1. Contexte

### *1.1 Présentation du Rectorat*

J'ai effectué mon stage au sein de la Direction des Systèmes d'Information (DSI) du Rectorat de l'académie de Clermont-Ferrand.

Le rectorat est la direction des services de l'Education Nationale chargée de mettre en œuvre, dans l'académie, la politique éducative définie au niveau national.

L'académie de Clermont-Ferrand correspond au territoire géographique de l'Auvergne et regroupe les 4 départements de l'Allier, du Cantal, de la Haute-Loire et du Puy-de-Dôme.

### *1.2 Présentation de la DSI*

La DSI est séparé en 4 divisions, 2 académiques, la Division des Données et des Applications Métiers (DDAM) et la Division Environnement de Travail et Infrastructures (DETI), et 2 nationales, le Pôle Réseaux et Hébergement (PRH) et l'Equipe Nationale Services Numériques (ENSN). C'est au sein de cette dernière que j'ai effectué mon stage.

L'ENSN reçoit les directives du ministère afin de créer des applications et des solutions informatiques qui seront mises en pratique sur le territoire académique mais aussi national. Elles sont au total de 14 équipes nationales réparties sur tout le territoire.

Le service de l'équipe nationale services numériques est composé d'une équipe de développement et d'une équipe de qualification et diffusion (documentation, communication et assistance). L'équipe de qualification a pour mission d'exécuter des tests de régression afin de garantir la qualité de diverses applications de l'Éducation Nationale. Le pôle automatisé assiste les testeurs manuels en développant des suites de tests automatisés.

## 2. Mes missions/déroulement

### *2.1 Début des travaux, la découverte du travail*

J'ai donc effectué mon stage dans la DSI et plus précisément au pôle automatisé de l'équipe de qualification.

Au cours des premiers jours, j'ai pu découvrir divers aspects du monde du travail tel que l'organisation du Rectorat, les différentes équipes et comment ces dernières fonctionnent ensemble.

J'ai assisté à la réunion mensuelle de tous les membres de l'équipe nationale (~40 personnes). Cette réunion abordait plusieurs points :

- Présentation des nouveaux arrivants (moi compris), présentation de leur parcours et de leur mission au sein de l'équipe. Certains d'entre eux sont des prestataires.
- Point sur l'avancée des projets et définition des objectifs
- Direction politique et impact sur les équipes

## 2.2 Immersion et mise à niveau

Afin de me familiariser avec les différents outils utilisés en automatisation, j'ai réalisé divers exercices. Mon premier projet a consisté à automatiser l'ajout d'un objet dans le panier sur Amazon via un navigateur. J'ai aussi pu me familiariser avec les interactions sur différents éléments (checkbox, radio, formulaire...) sur le site de test demoqa.

J'ai découvert les sélecteurs CSS et XPath, qui permettent de sélectionner et interagir avec les éléments voulus d'un DOM (Document Object Model). Les sélecteurs CSS définissent les éléments sur lesquels s'applique un ensemble de règles CSS.

L'alternative au CSS est le Xpath qui est un langage de requête pour localiser une portion d'un document XML à l'aide d'un chemin de localisation. J'ai appris à utiliser le Xpath à l'aide du site Xpath Diner.

1 usage	no usages
<code>@FindBy(css = "#item-1")</code>	<code>@FindBy(xpath = "(/tbody)[2]")</code>
<code>private WebElement boutonCheckBox;</code>	<code>private WebElement tableauNumeroDeux;</code>

Le contrôle des différents navigateurs (Chrome, Edge, Firefox) est effectué par Selenium, un framework disponible pour plusieurs langages, dont Java.

## 2.3 La Mission

Ma mission durant le stage était de mettre en place le projet d'automatisation (Java/Maven) des tests de régression de l'application Base Académique de Nomenclature (BAN) en appliquant les connaissances acquises durant mon année scolaire (Java, SQL, Maven...). Cette application regroupe les établissements, les matières et les Modules Élémentaires de Formation (MEF) d'une académie et a pour but de décrire l'offre de formation de chaque établissement et rendre cette nomenclature utilisable par les différentes applications du SI. Les principaux utilisateurs de BAN sont les services statistiques académiques (SSA).

Pour développer le test en Java, j'ai utilisé l'EDI IntelliJ. Le framework central utilisé est appelé « AutoLab ». Cet outil a été développé spécifiquement par le pôle automatisation afin de faciliter le développement des tests. Il se base principalement

sur les frameworks TestNG et Selenium. AutoLab se présente sous forme de dépendance Maven et facilite de nombreux types de tests (navigateur, API, fichiers...).

## 2.4 Description du procédé de développement

L'un des premiers problèmes rencontrés était de n'avoir aucun accès au code source des applications. J'ai donc recherché dans la documentation comment fonctionne l'application BAN.

En premier lieu, j'ai dû me familiariser avec l'application que je devais tester. Explorer les pages afin de décider de la meilleure approche possible pour créer un code qui teste chacune des fonctions mais qui soit aussi simple à comprendre pour les futures maintenances.

Tout d'abord j'ai créé un dépôt sur le gitLab qui est hébergé sur les serveurs de l'Éducation Nationale. Grâce à Git, j'ai créé plusieurs branches successives afin de coder sans se soucier de possibles erreurs rendant le code irrécupérable.

Le projet est organisé en 4 à 5 parties :

- PageObject :

Les PageObject sont des classes dans lesquelles sont définis les différents éléments du DOM de type WebElement grâce à leurs sélecteurs respectifs. Ce sont ces éléments qui vont permettre de manipuler, via le code, l'interface graphique et permettre de tester l'application.

Exemple : Dans cette classe, un WebElement déclaré va contenir le champ d'un formulaire qui peut être trouvé grâce à son sélecteur. Il sera ensuite possible de remplir ce champ avec une méthode rédigée dans cette même classe avec les paramètres renseignés.

```
1 usage
@FindBy(css = "#age")
private WebElement champAge;

1 usage  E. Pourchon
public void remplirChampAge(String age) {
    driverC.remplir(champAge, age);
}
```

Registration Form

First Name

First Name

Last Name

Last Name

Email

name@example.com

Age

Age

Salary

Salary

Department

Department

Submit

- PageFragment :

Les PageFragment sont des PageObject situationnelles. Elles définissent une partie de la page actuelle comme une page indépendante, utile lorsqu'un élément se répète comme la ligne d'un tableau.

- Fonctionnalités :

Les Fonctionnalités font appels aux actions exposées par les PageObject au sein de méthodes, et de manière séquentielle.

C'est aussi ici que les requêtes SQL se trouvent. Elles sont utilisées pour interroger le système de gestion de base de données (DB2).

C'est aussi dans les Fonctionnalités que j'ai découvert la possibilité de surcharger une méthode, facilitant le développement de certaines fonctions sans à avoir besoin de répéter des lignes de code rédigées dans une précédente méthode.

- Scénario :

Les Scénarios font généralement appel à une suite de fonctionnalités. Ils sont exécutés afin de passer en test les différentes fonctions souhaitées.

- Suites de test :

Les suites de test sont des ensembles de scénarios. C'est avec les suites qu'une application peut être testée dans son intégralité automatiquement.

Un exemple de chacun de ses fichiers se trouve en annexe.

Ces différentes parties, liées ensembles, permettent de passer un test ou une suite de tests sur une application spécifique à l'aide des différentes assertions présentes qui permettent de vérifier une condition considérée comme vraie.

Une fois le code rédigé et testé, j'utilise le dashboard pour stocker des requêtes SQL que j'appelle via leur clé, je précise comment sera ordonnée la requête via des variables ajoutées.

Le dashboard permet aux utilisateurs d'Autolab de consulter les résultats des suites de tests, mais aussi de stocker les requêtes SQL qui sont utilisées par les tests automatisés.

La requête sur le dashboard :

Clé: consultationMatiere

```
SELECT r.*
FROM (
    SELECT CASE
        WHEN bm.DATE_FERMETURE < %DATE_ANNEE%
        OR bm.DATE_OUVERTURE >= %DATE_ANNEE%
        THEN 'Fermé'
        END AS ETAT
    ,bm.*
    FROM BAN_MATIERE bm
    ) AS r
ORDER BY %ORDRE%
```

COPIERMODIFIERCOMMENTAIRE

L'appel de la requête sur le fichier Java

```
8 usages  E. Pourchon
public void testerTableauAvecTriSansFiltre(TypeTri orderBy) {
    String ordre = selectionTypeTri(orderBy);
    ResultSetC consultationMatiere =
        SqlStore.getResultSetC( cle: "consultationMatiere", ...vars: "%ORDRE%", ordre, "%DATE_ANNEE%", annee);
    verificationTableau(consultationMatiere);
}
```

La requête une fois effectuée avec les variables changées :

```
-----
          Exécution de la requête SQL
-----
SELECT r.*
FROM (
    SELECT CASE
    WHEN bm.DATE_FERMETURE < '2024-01-01'
    OR bm.DATE_OUVERTURE >= '2024-01-01'
    THEN 'Fermé'
    END AS ETAT
    ,bm.*
    FROM BAN_MATIERE bm
    ) AS r
ORDER BY CODE ASC
```

Ensuite je documente le test sur SQUASH en décrivant chaque étape et détaillant les paramètres utilisés sur la gauche, et le résultat attendu sur la droite. (Cette étape est normalement réalisée avant l'automatisation des tests).

**SN 0501 Matières**

Référence du cas de test

● En cours de rédaction

✎ Faible

Pas d'exécution

Prérequis et pas de test

	ACTION	RÉSULTAT ATTENDU
3	Cliquer sur le lien Base Académique de Nomenclature (BAN)	Accès à la page d'accueil de BAN
4	Sélectionner de l'année correspondante à la base de donnée utilisé	(Cliquer pour éditer...)
5	Cliquer sur l'option du menu "Consultation" puis cliquer sur l'option "Matières"	Affichage du tableau sans tri ni filtre de la base de donnée
6	Parcours de chaque case présentes sur chaque pages du tableau et comparaison à la base de donnée SCOMET23 qp4 à l'aide de la requête : <code>consultationMatières</code>	Les données affichées sur le tableau sont identiques à la base de donnée
7	Retour à la page 1 du tableau et test de chaque bouton de tri + comparaison à la base de donnée avec la même requête	Les données affichées sur le tableau sont identiques à la base de donnée

Enfin, si une erreur est repérée durant le test automatique, je l'indique et le documente via Sésam, une plateforme répertoriant les rapports d'anomalie des applications, et le dashboard d'automatisation afin que ces anomalies soient corrigées suivant leur ordre de priorité :



BUILDS JENKINS

SQL STORE

ANOMALIES

ELIGIBILITE

BAN

Statut

Toutes

Id

Version

+ AJOUTER

1 anomalies(s)

Rows per page: 25 1-1 of 1

	Id	Titre	Version	Statut	Lien	Date	Modifier
▼	105	Exception dans l'organisation Ouvert / Fermé en BDD 2022 / 2023	23.3.0.0.3	En cours		20/06/2023	

Identifiant	Projet	Module	Visibilité	Date de soumission	Dernière mise à jour
0440111	Sconet BAN-Q	60- Consultation/ 3- MEF Programme	public	20-06-23 16:08	20-06-23 16:10
Rapporteur	SESAM				
Assigné à	SCOBAN-DEV				
Priorité	normal	Sévérité	mineur	Reproductibilité	toujours
Etat	transmis pour traitement	Résolution	ouvert	Version de l'OS	
Plate-forme		OS			
Version du produit	23.3.0.0.1	Build			
Version ciblée		Résolu dans la version			
Résumé	0440111: Trie par Ouvert / Fermé ne fonctionne pas sur un cas particulier				
Description	[Bug trouvé en automatisation de BAN] Voir copie d'écran 1. Aller sur Consultation > MEF / Programme 2. Mettre une clé de gestion à 1 dans la case de droite 3. Cliquez sur l'une des deux flèches Ouvert / Fermé 4. Naviguer sur les tableaux => En page 3, Un "Fermé" apparaît au milieu des "Ouvert"				
Classification complémentaire	non renseigné				
Type de signalement	Anomalie				
Site	QUALIF Clermont				
Origine	Qualification				
Date de correction souhaitée					
Date de correction prévue					
Correction dans la version de qualification	non renseignée				
Livraison dans la version de production					
Date de livraison en production					
TMA	Non				
Garantie	Non				
Charge Développement (j.h)					
Charge Qualification (j.h)					

## GESTIONNAIRE D'ANOMALIES SESAM

C'est ainsi que le procédé s'est déroulé pour le projet qui m'a été fourni.

Tout ce projet a été fait en relation avec les équipes présentes au sein de la DSI à qui j'ai pu demander de l'aide. Ce travail m'a permis de voir qu'un projet ne peut se réaliser sans communication et collaboration entre différents acteurs.

## 2.5 Outils utilisés

<p>IntelliJ, environnement de développement intégré utilisé pour la création des tests automatisés</p> 	<p>DBeaver, logiciel open-source utilisé pour consulter ou modifier le contenu de BDDs</p> 
<p>Git/GitLab, gestionnaire de version et dépôt distant hébergé sur un serveur de l'EN.</p> 	<p>SQUASH TM pour la consultation et la rédaction des scénarios de tests, étape par étape</p> 
<p>Xwiki servant de documentation pour les applications et leurs requêtes SQL</p> 	<p>Jenkins pour la planification de l'exécution des tests sur une machine virtuelle</p> 
<p>Selenium, outil puissant pour contrôler les navigateurs web grâce à des programmes et effectuer l'automatisation du navigateur.</p> 	<p>Dashboard, application stockant les requêtes SQL, les tests automatisés, les résultats des tests automatisés fait par Jenkins et les anomalies.</p> 

### 3 Conclusion

Ce stage au sein du Rectorat et des équipes nationales m'a permis de trouver une ambiance de travail professionnelle mais dans un cadre bien différent de ce que j'ai pu connaître.

En effet, durant ce stage, j'ai travaillé en autonomie mais aussi avec d'autres équipes du service pour finaliser certaines parties du projet qui m'a été confié.

L'utilisation de Java, langage qui me semblait abstrait, m'a permis de visualiser ses diverses applications, de prendre confiance en mes capacités et d'affiner mes compétences.

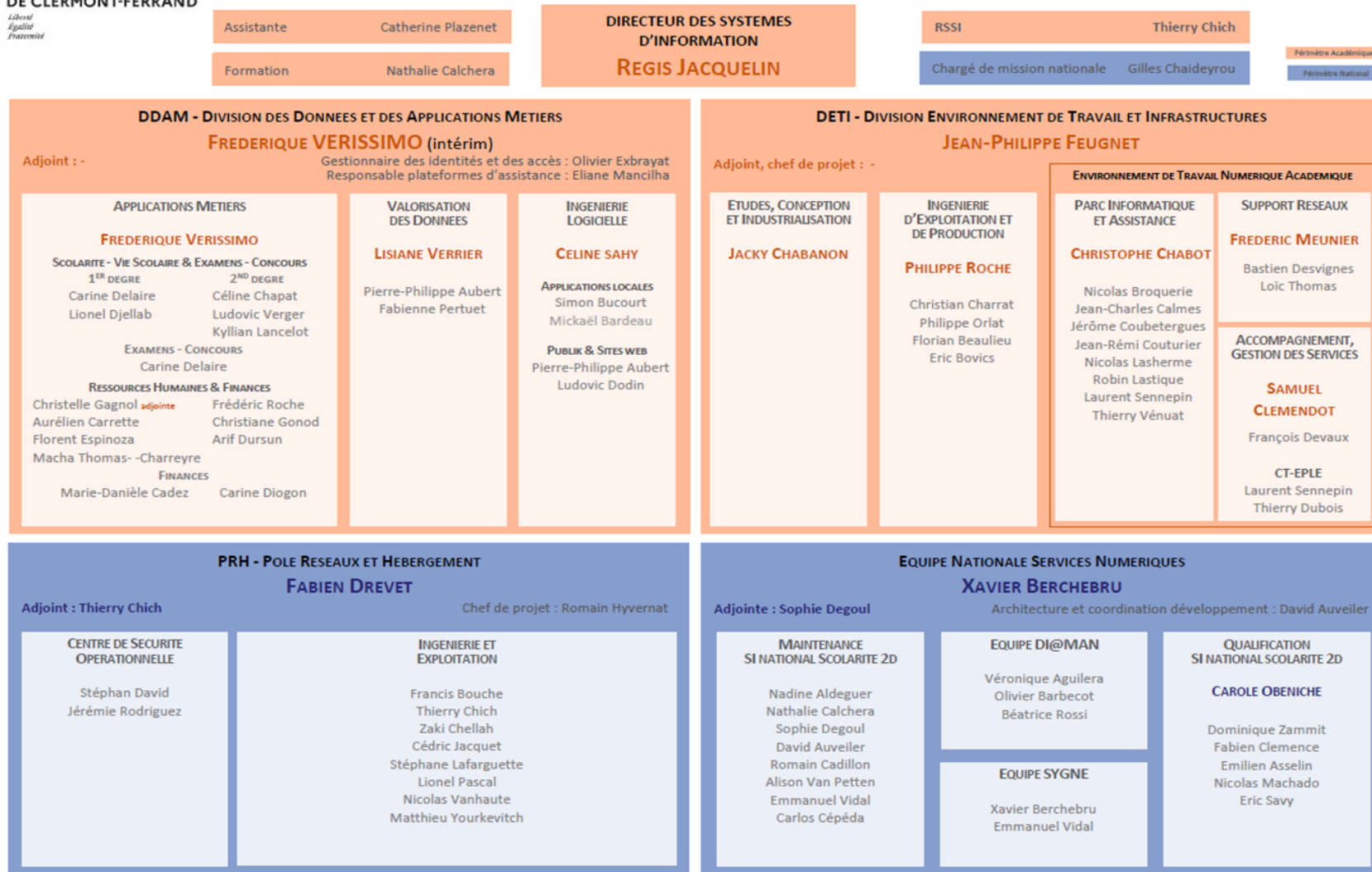
La création de tests automatisés a été un plaisir pour moi et me sera utile pour la suite de mes études et de mes projets et me conforte dans l'envie de continuer sur cette voie.

Je tiens à remercier encore une fois les personnels de la DSI du Rectorat pour m'avoir accordé leur confiance en m'intégrant dans leur équipe durant toute la durée du stage, me permettant de prendre confiance en moi et sur le choix de mes études.

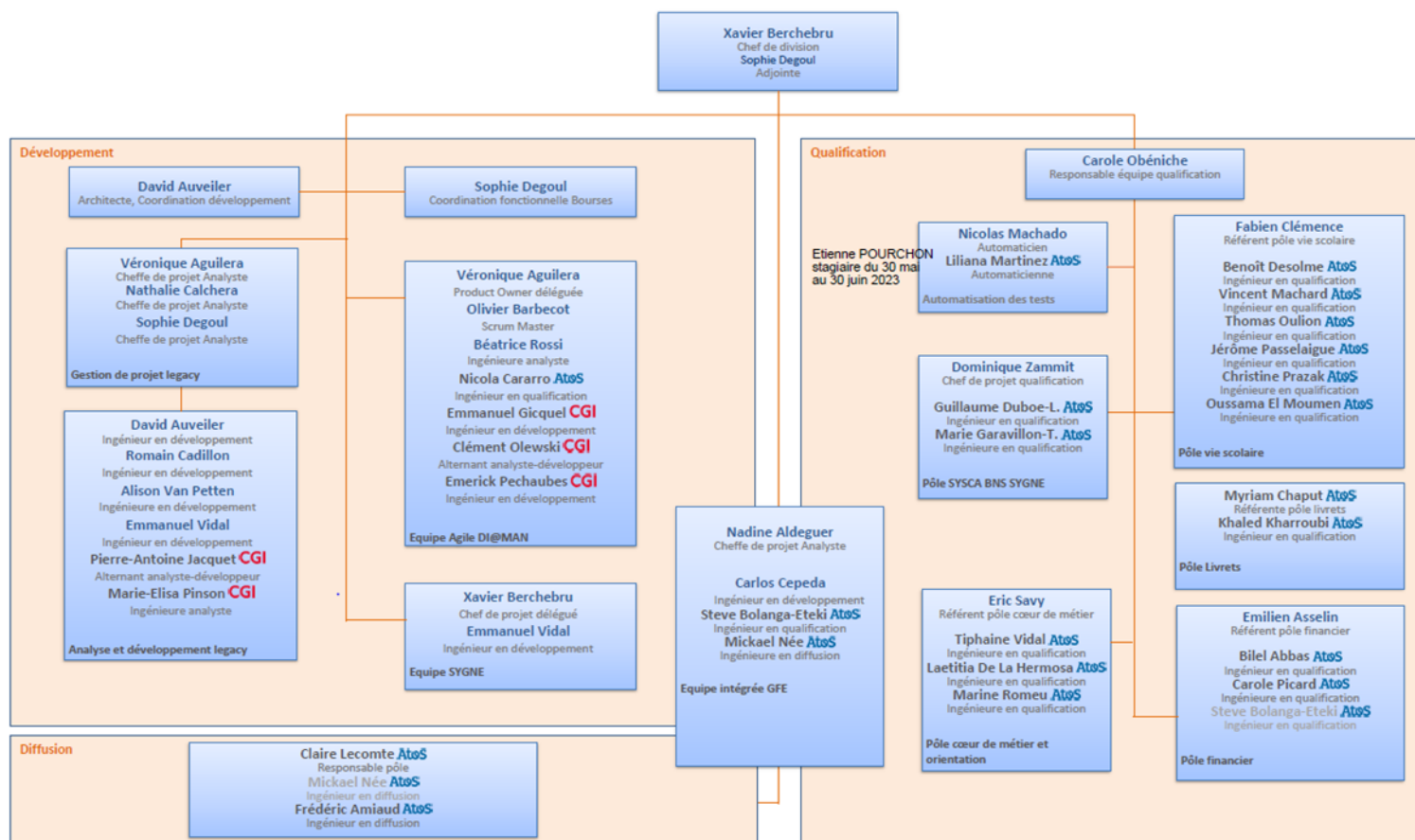
## Annexe 1 – Organigramme de la DSI



### Direction des Systèmes d'Information – Mai 2023



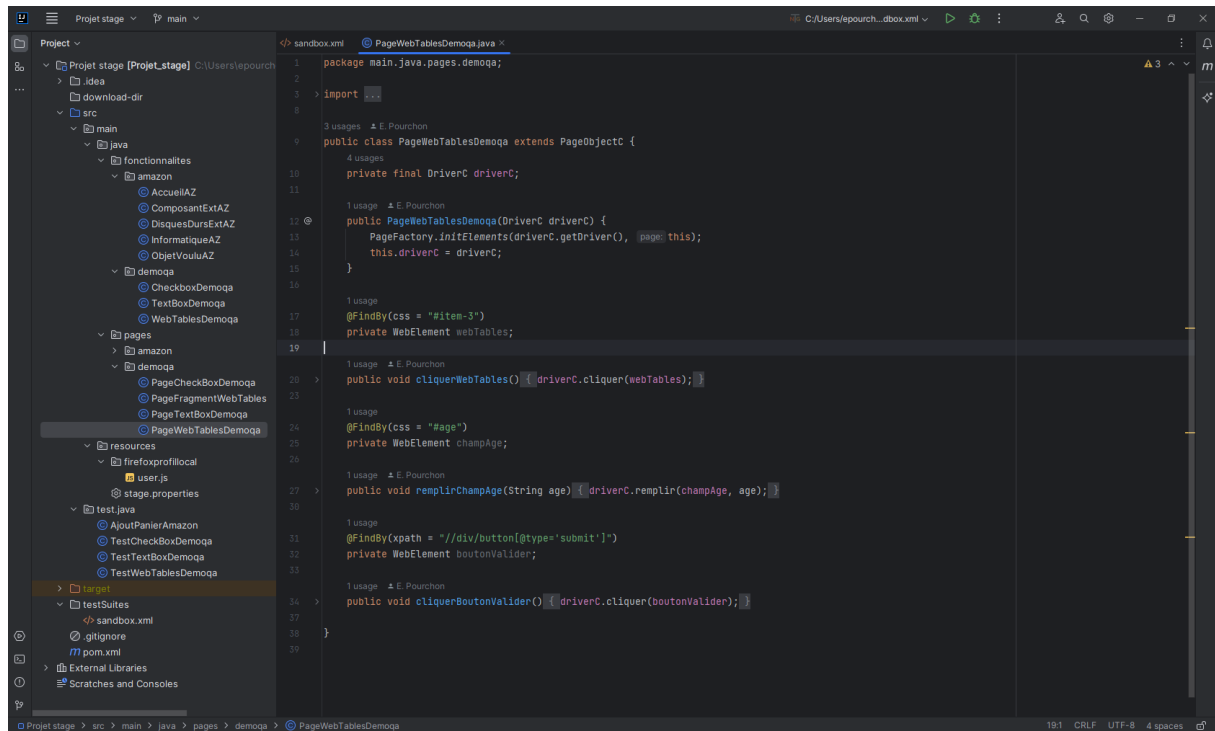
## Annexe 2 – Organigramme de l'ENSN



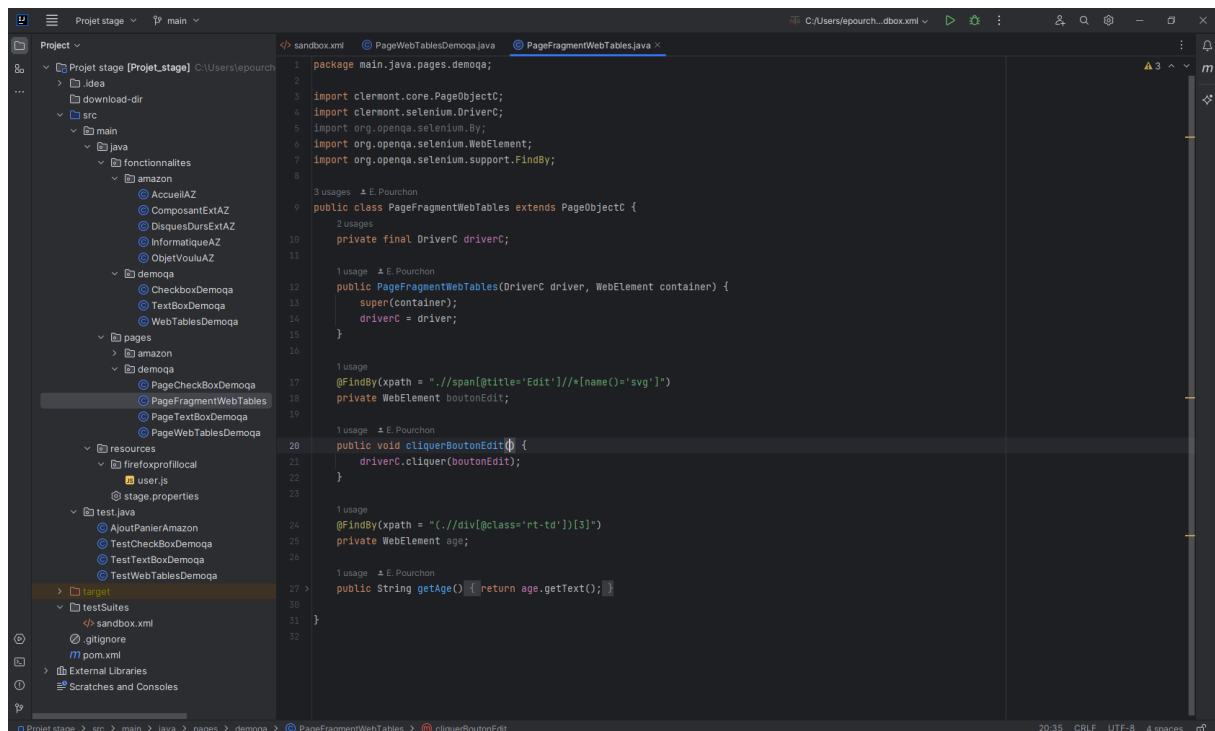
Grisé : Activité principale au sein de l'équipe intégrée GFE

## Annexe 3 – Exemples de pages

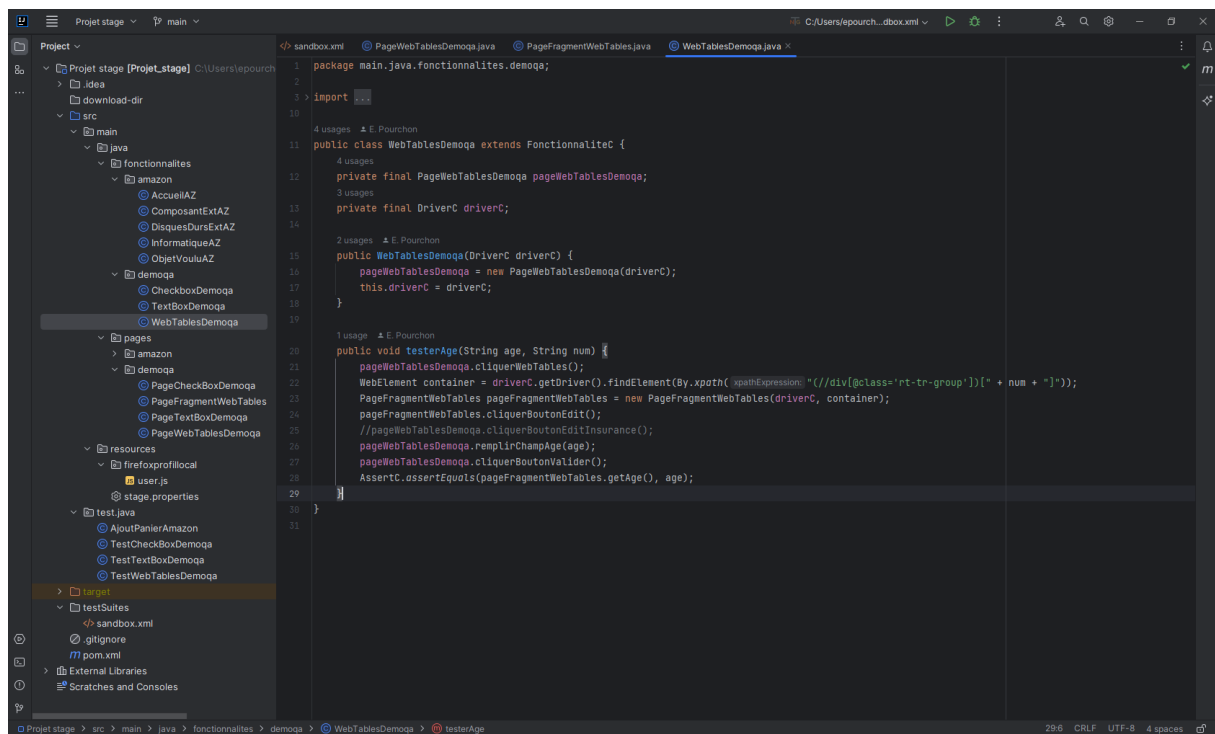
### Exemple PageObject :



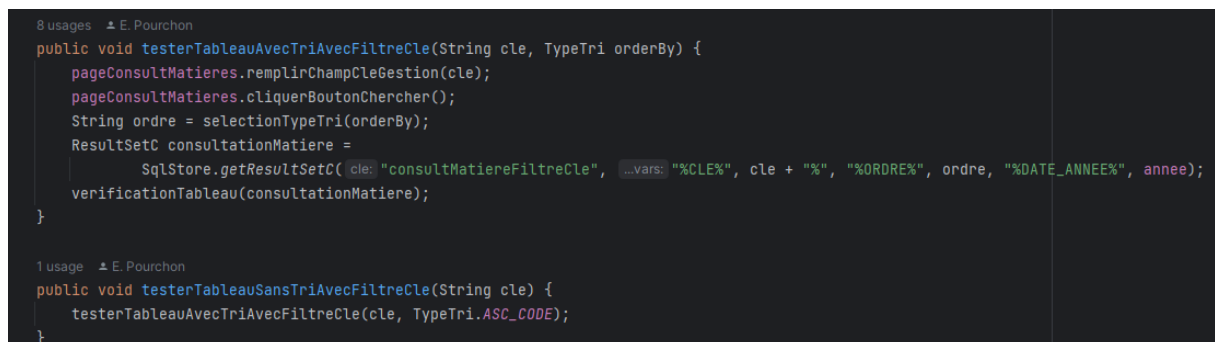
### Exemple PageFragment :



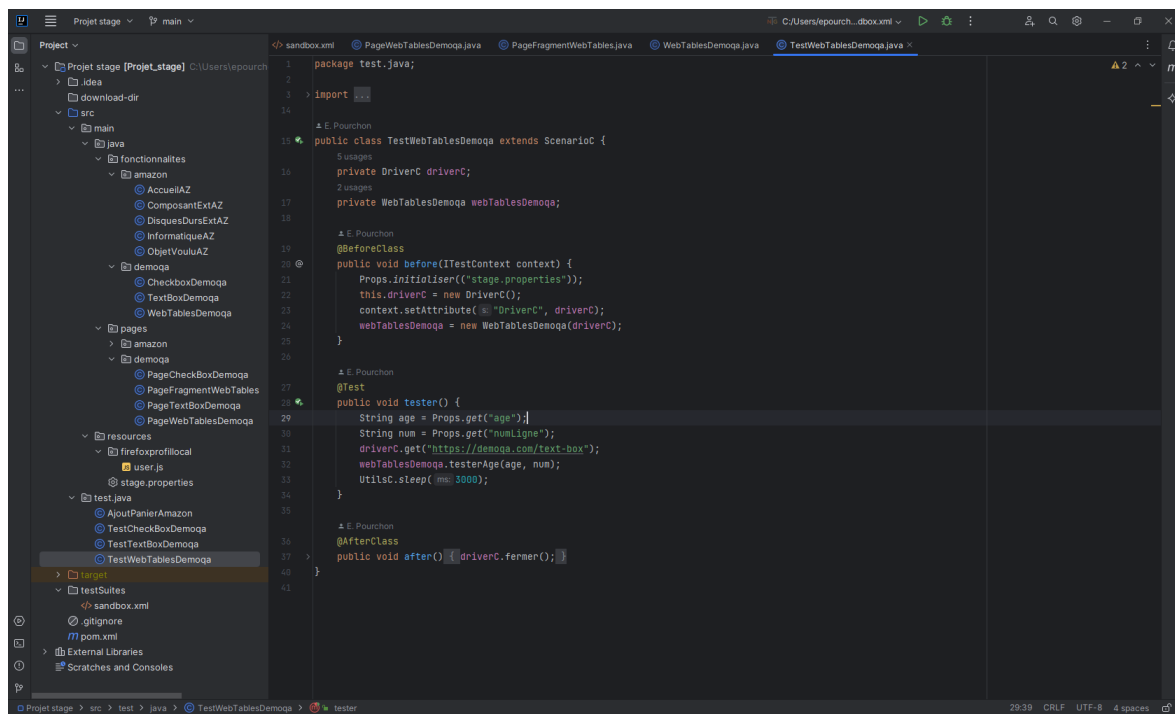
## Exemple Page Fonctionnalité :



## Exemple de surcharge de méthode :

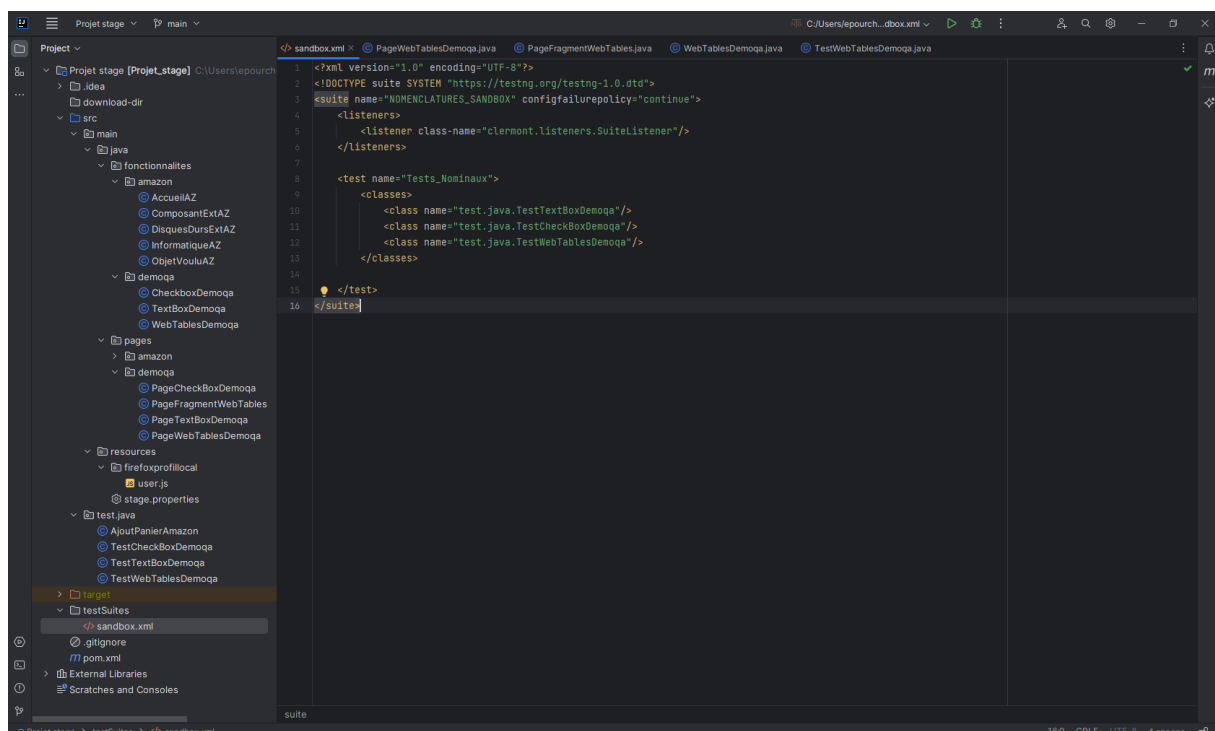


## Exemple Page Scénario :



```
1 package test.java;
2
3 > import org.openqa.selenium.*;
4
5 // E. Pourchon
6 public class TestWebTablesDemoqa extends ScenarioC {
7
8     // E. Pourchon
9     private DriverC driverC;
10
11     // E. Pourchon
12     private WebTablesDemoqa webTablesDemoqa;
13
14     // E. Pourchon
15     @BeforeClass
16     public void before(ITestContext context) {
17         Props.initialiser("stage.properties");
18         this.driverC = new DriverC();
19         context.setAttribute("@DriverC", driverC);
20         webTablesDemoqa = new WebTablesDemoqa(driverC);
21     }
22
23     // E. Pourchon
24     @Test
25     public void tester() {
26         String age = Props.get("age");
27         String num = Props.get("numLigne");
28         driverC.get("https://demoqa.com/text-box");
29         webTablesDemoqa.testerAge(age, num);
30         UtilsC.sleep(3000);
31     }
32
33     // E. Pourchon
34     @AfterClass
35     public void after() { driverC.fermer(); }
36 }
37
38
39
40
41
```

## Exemple Page Suite de test :



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
3 <suite name="NOMENCLATURES_SANDBOX" configfailurepolicy="continue">
4
5     <listeners>
6         <listener class-name="clermont.listeners.SuiteListener"/>
7     </listeners>
8
9     <test name="Tests_Nomineux">
10         <classes>
11             <class name="test.java.TestTextBoxDemoqa"/>
12             <class name="test.java.TestCheckBoxDemoqa"/>
13             <class name="test.java.TestWebTablesDemoqa"/>
14         </classes>
15     </test>
16 </suite>
```