



# Rapport de stage Réalisé par Etienne POURCHON Du 8 janvier 2024 au 16 février 2024

Maître de stage : Nicolas MACHADO

Enseignant référent : Bastien DUFOUR

# **AUTOMATISATION DE TESTS**

2ème 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: 2023-2024

# Table des matières

Remerciements:	2
1. Contexte	3
1.1 Présentation du Rectorat	3
1.2 Présentation de la DSI	3
2. Mes missions/déroulement	3
2.1 Rappel de la Mission	4
2.2 Poursuite du procédé de développement	4
2.3 Intelligence artificielle et automatisation des tests	6
2.3.1 La recherche	7
2.3.2 Les essais	8
3 Conclusion	12
Annexe 1 – Organigramme de la DSI	13
Annexe 2 – Organigramme de l'ENSN	14
Annexe 3 – Présentation	15
Annexe 4 - Fil d'échanges avec Chat GPT	20

# Remerciements:

- <u>Xavier Berchebru, chef de division :</u> de m'avoir fait confiance pour ce poste et m'avoir accordé du temps pour me présenter à la division.
- <u>Nicolas Machado, automaticien</u>: pour son approche pédagogique qui a rendu chaque aspect du stage agréable et intéressant à faire.
- <u>Le pôle BEE / pôle cœur de métier et orientation :</u> pour son aide durant tout le long de ma mission sur le projet BAN.
- <u>Toute l'équipe de la DSI :</u> pour son accueil et son soutient 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 séparée 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 automatisation assiste les testeurs manuels en développant des suites de tests automatisés.

# 2. Mes missions/déroulement

Ayant déjà effectué mon stage de 1<sup>ère</sup> année dans la DSI et plus précisément au pôle automatisation de l'équipe de qualification, j'ai donc, dans un premier temps, poursuivi la mission qui m'avait été confiée.

Puis dans un second temps, il m'a été proposé de faire des recherches sur l'IA appliquée à l'automatisation des tests.

# 2.1 Rappel de 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 Elémentaires 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...).

# 2.2 Poursuite du procédé de développement

A la fin de mon stage de 1<sup>ère</sup> année, une personne de l'équipe a pris en charge la suite du projet d'automatisation des tests de BAN.

Nous étions donc deux à travailler sur le même projet mais sur des pages différentes.

Déjà familiarisé avec l'application et les outils, j'ai pu être opérationnel dès le début du stage.

Lors de mon premier stage, grâce à GitLab, j'avais créé plusieurs branches successives afin de coder sans se soucier de possibles erreurs rendant le code irrécupérable.

Cela s'est avéré utile car être deux sur le même projet rendait chaque ajout au code potentiellement problématique, la création de branche permettait de faire des merges avec des risques moindres.

Le principal problème rencontré durant cette période a été le manque de précision voire l'absence de documentation fonctionnelle relative au code source de l'application, d'autant que les méthodes de codification ont évolué depuis sa création.

Les requêtes SQL, utilisées pour interroger le système de gestion de base des données, n'étaient pas répertoriées. Afin d'automatiser les tests, nous devions retrouver ces requêtes.

La solution a été d'élaborer plusieurs requêtes afin de trouver la bonne.

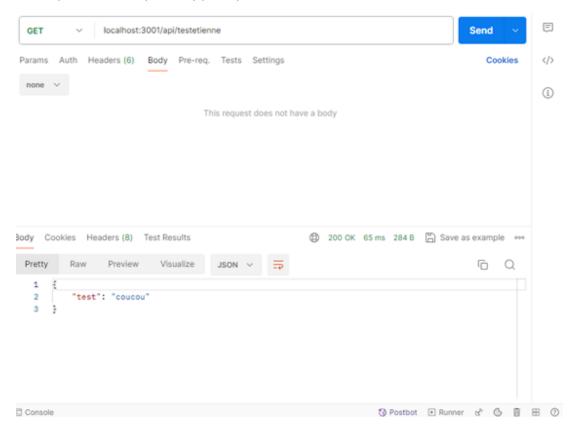
Cette opération a été extrêmement chronophage.

En parallèle de cette mission, mon maître de stage m'a présenté le code du Dashboard de l'automatisation du service qualification.

Etienne POURCHON

Cela m'a permis de découvrir Postman (), Prisma (ORM) et Typescript qui ont servi à la rédaction des fonctions pour le back-end du Dashboard.

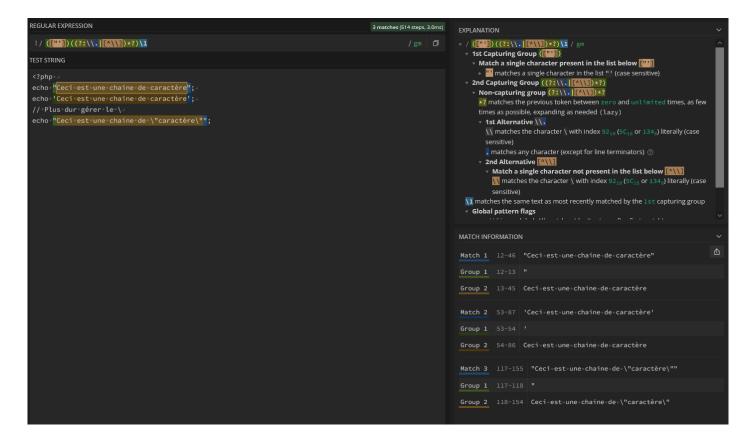
Exemple d'un endpoint appelé par Postman :



```
app.get('/api/testetienne', async(req, res) -> {
    //Requête prisma pour obtenir app dont id > 7
                      const applications = await prismaClient.appli.findMany({
                           where: {
                                       gt: 7
                     applications.forEach(a => console.log(a.id + " " + a.nom));
                      const test: String = "fdgdf";
                      return res.status(200).json({
              OUTPUT DEBUG CONSOLE TERMINAL PORTS
                                                                                                             PROBLEMS
[nodemon] starting "node --loader ts-node/esm src/index.ts"
(node:9620) ExperimentalWarning: `--experimental-loader` may be removed in the future; instead use `registe --import 'data:text/javascript,import { register } from "node:module"; import { pathToFileURL } from "node: (Use `node --trace-warnings ... `to show where the warning was created)
Serveur API démarré sur l'environnement development, port 3001
8 NOMENCLATURES
9 LSU
10 BAN
11 SIECLE ADMIN
```

J'ai pu aussi apprendre à mieux me servir des expression régulières (Regex) à l'aide de site tel que Regex101, ce qui m'a permis d'extraire des parties de texte (requête SQL) afin de modifier leur coloration syntaxique dans l'IHM du Dashboard.

Exemple de code rédigé sur Regex101 :



# 2.3 Intelligence artificielle et automatisation des tests

Afin de diversifier mes missions, Xavier Berchebru, chef de la Division, m'a proposé de faire des recherches sur l'Intelligence Artificielle et sa possible utilisation dans l'automatisation des tests puis d'en faire une présentation simple sous la forme d'un PowerPoint.

# 2.3.1 La recherche

J'ai effectué mes recherches sur internet et retenu 2 articles :

- « Comment l'intelligence artificielle peut révolutionner les techniques de test de logiciels » de Moez Krichen paru en 2023 sur le site HAL open science
- « Comment améliorer les tests continus à l'heure de l'IA générative » de Isaac Sacolick publié le 1<sup>er</sup> septembre 2023 sur le site Le Monde de l'informatique

Ces deux articles présentent les avantages de l'utilisation de l'IA pour élaborer des tests.

Tout d'abord, l'IA dotée d'une fonction d'apprentissage automatique, peut générer automatiquement des cas de test, ce qui est utile notamment dans les cas de test de régression car les scripts s'adaptent automatiquement aux changements de l'application. Cela permet d'améliorer la fiabilité des tests.

Une autre fonction intéressante de certains outils d'IA est la possibilité d'utiliser le langage naturel (NLP) pour demander l'écriture d'un script sans avoir recours au codage.

La vérification visuelle par l'IA permet d'effectuer des comparaisons entre les différentes interfaces utilisateur à l'aide de captures d'écran afin d'indiquer au testeur où se situent les « différences visuelles », ce qui réduit le nombre de cas de test à réaliser.

La fonction auto-réparatrice de l'IA réduit la maintenance manuelle des tests en modifiant le script pour l'adapter aux nouveautés apportées dans le logiciel.

L'apport de l'IA dans les outils de test permet d'alléger la charge de travail du testeur notamment sur les tâches répétitives et chronophages. D'après les auteurs de ces articles, l'automatisation des tests traditionnelle ne sera plus suffisante pour suivre le développement de plus en plus rapide des logiciels, d'autant plus si ces derniers ont été développés à partir d'outil d'IA générative.

A l'heure actuelle, les solutions de test basées sur l'IA sont utiles au testeur mais encore basiques. Avec l'émergence de nouvelles IA, leur intégration dans les outils d'automatisation de test sera indispensable pour accélérer le développement de logiciels et en améliorer la qualité.

# 2.3.2 Les essais

Suite à mes recherches sur l'IA et ses applications possibles dans l'automatisation, j'ai essayé d'utiliser une Al générative (ici ChatGPT) afin de produire un scénario de test à insérer dans Squash.

J'ai d'abord essayé en décrivant le visuel de l'application et ses données pour créer un premier prompt, en précisant plusieurs détails au fur et à mesure pour obtenir un exemple de scénario correspondant à l'application BAN.

J'ai ensuite changé d'approche en utilisant la documentation (Confluence) et les spécifications de l'application pour la rédaction d'un prompt, ce qui m'a permis d'obtenir un visuel sous forme de tableau afin de permettre à ChatGPT de rédiger un nouvel exemple de scénario.

Ces différents essais m'ont permis de voir la compatibilité de ChatGPT avec le travail d'automatisation de test.

Même si les réponses étaient approximatives, chaque prompt a permis d'avoir une idée de comment construire un scénario.

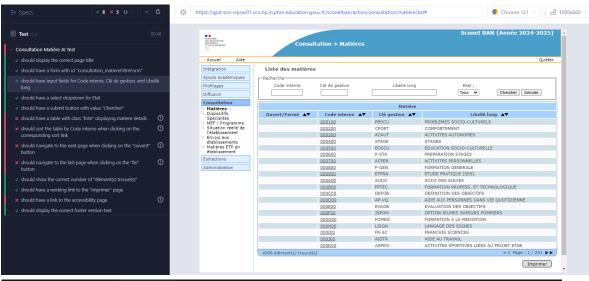
Un exemple du fil d'échanges avec Chat GPT se trouve en annexe.

J'ai donc commencé à chercher des outils utilisant l'IA spécialisés dans la génération de tests automatisés afin de faire d'autres essais, j'ai retenu TestCraft car c'est le seul qui ne demandait pas la création d'un compte.

Pour tester les capacités de TestCraft, j'ai installé Cypress à l'aide de npm, un gestionnaire de paquet.

J'ai pu apprendre à me servir de Cypress et voir son utilité notamment dans sa capacité à enregistrer chaque étape du test avec des captures d'écran, montrant où le click a été effectué ou quel élément a été sélectionné, ce qui a été pratique dans la compréhension des résultats nuls de certains tests générés par TestCraft.

# Affichage via Cypress:



```
Objective to the state of the s
```

J'ai pu créer une nouvelle commande pour que chaque test se connecte au site et puisse être fait depuis une session vierge :

```
describe('Consultation Matière AI Test', () => {
    beforeEach(() => {
        cy.login('username', 'password')
        cy.visit(urlPageVoulu)
    })
```

```
Cypress.Commands.add('login', (username, password) => {
    cy.visit(urlBase)

it('has the correct initial state', () => {
        cy.get('input[name="user"]').should('be.visible').should('be.empty');
        cy.get('input[name="password"]').should('be.visible').should('be.empty');
        cy.get('input[name="button"]').should('be.visible').should('have.value', 'Valider');
    });

cy.get('input[name=user]').type(username)

// {enter} causes the form to submit
    cy.get('input[name=password]').type() ${password}{enter}', { log: false }}
}
```

Plusieurs essais ont été réalisés sur des pages différentes, les résultats diffèrent à chaque fois bien que les pages aient la même structure, les premiers tests sont basiques et peu intéressants, et les derniers tests sur une autre page montrent que Testcraft arrive à créer des tests intéressants mais incomplets.

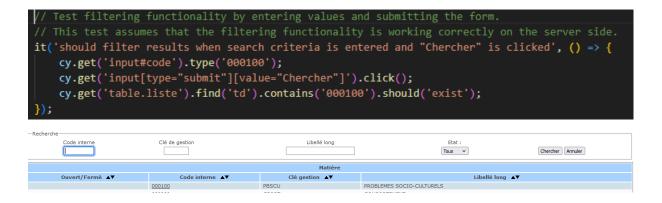
Dans les faits, Testcraft est un outil intéressant mais pas suffisant pour être utile au projet d'automatisation des tests de régression de l'application BAN, soit les tests sont basiques et donc peu intéressants mais fonctionnent, soit ils sont un peu plus complexes mais il faut une intervention manuelle pour que ses tests fonctionnent et offrent des résultats fiables.

Cet outil a un principe intéressant pour tester l'application sans accès à la base de données qui est de récupérer les données de la première ligne du tableau et les mettre dans les champs de recherche pour voir si les données obtenues sont les bonnes.

Quant à quel test sera réalisé, c'est l'outil qui décide.

Exemple de différences dans les tests réalisés avec Testcraft :

-Test qui fonctionne et qui vérifie correctement les résultats (obtenu après 4 prompts sur un même élément):



- Même test sur une page différente mais dont la vérification des résultats est invalide (obtenu après un prompt) :



Ces essais permettent quand même de voir le potentiel de l'IA dans le domaine de l'automatisation des tests.

Un outil plus perfectionné aurait peut-être permis de gagner du temps dans la recherche de requêtes SQL pour le projet d'automatisation de test de l'application BAN.

# 2.5 Outils utilisés

IntelliJ, environnement de développement intégré utilisé pour la création des tests automatisés



DBeaver, logiciel open-source utilisé pour consulter ou modifier le contenu de BDDs



Git/GitLab, gestionnaire de version et dépôt distant hébergé sur un serveur de l'EN.



SQUASH TM pour la consultation et la rédaction des scénarios de tests, étape par étape



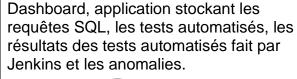
Xwiki servant de documentation pour les applications et leurs requêtes SQL

Jenkins pour la planification de l'exécution des tests sur une machine virtuelle



**\***W|K|

Selenium, outil puissant pour contrôler les navigateurs web grâce à des programmes et effectuer l'automatisation du navigateur.







Cypress, outil d'automatisation des tests frontaux pour les tests de régression des applications web

Postman, outil qui permet d'effectuer des requêtes API





log.webnet.fr

Etienne POURCHON

# 3 Conclusion

Cette 2<sup>ème</sup> période de stage au sein du Rectorat et des équipes nationales m'a fait entrevoir une partie plus laborieuse du travail de testeur et de l'importance de la documentation fonctionnelle.

Mes débuts de recherche sur l'intelligence artificielle m'ont fait prendre conscience de l'intérêt de la veille technologique surtout dans le domaine de l'informatique où les avancées sont rapides. La nécessité d'adapter ses outils et de s'adapter aux nouveaux outils est une condition impérative pour rester productif et compétitif.

Cela m'a aussi apporté une vision plus large quant aux possibilités de l'intelligence artificielle et peut-être une piste à explorer pour mes futures études.

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

RESSOURCES HUMAINES & FINANCES

FINANCES

Marie-Danièle Cadez Carine Diogon

Frédéric Roche

Macha Thomas-Charr.

Christelle Gagnol adjointe Christiane Gonod

Arif Dursun

Florent Espinoza



# Direction des Systèmes d'Information - Janvier 2024

Ludovic Dodin

**IDENTITES ET ACCES** 

**OLIVIER EXBRAYAT** 

Aurélien Carrette

Assistante Catherine Plazenet

Formation Nathalie Calchera

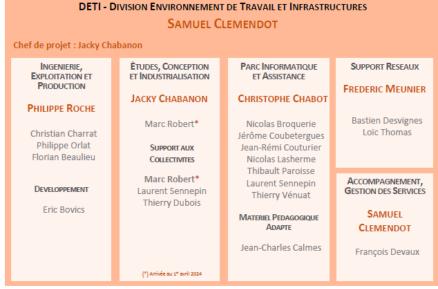
DIRECTEUR DES SYSTEMES
D'INFORMATION
REGIS JACQUELIN

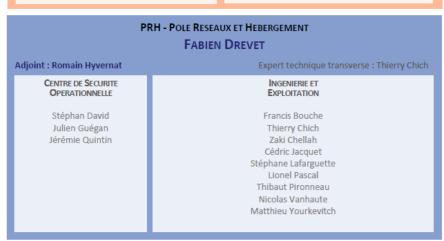
RSSI Thierry Chich

Chargé de mission nationale Gilles Chaideyrou

Périmètre Académique

#### **DDAM - DIVISION DES DONNEES ET DES APPLICATIONS METIERS FABRICE JOUAN** Adjointe : Frédérique Vérissimo Responsable plateformes d'assistance : Eliane Mancilha APPLICATIONS METIERS VALORISATION INGENIERIE DES DONNEES LOGICIELLE FREDERIQUE VERISSIMO LISIANE VERRIER **CELINE SAHY** SCOLARITE - VIE SCOLAIRE & EXAMENS - CONCOURS APPLICATIONS LOCALES Pierre-Philippe Aubert 1ER DEGRE 2<sup>ND</sup> DEGRE Mickaël Bardeau Fabienne Pertuet Carine Delaire Céline Chapat Simon Bucourt Lionel Diellab Ludovic Verger **EXAMENS - CONCOURS** PUBLIK & SITES WEB Carine Delaire Pierre-Philippe Aubert







# Annexe 2 – Organigramme de l'ENSN

#### DIRECTION DES SYSTEMES D'INFORMATION - EQUIPE NATIONALE SERVICES NUMERIQUES ACADÉMIE DE CLERMONT-FERRAND Organigramme fonctionnel - Octobre 2023 SAND-EO-**POLE FINANCIER** POLE TECHNIQUE RNVP SYGNE Coordination Fonctionnelle Bourses : **XAVIER BERCHEBRU** S. DEGOUL ARCHITECTE DIAMAN / Calculateur Chef de division Product Owner déléguées Cheffe de projet Sophie DEGOUL V. AGUILERA M-E. PINSON Adjointe au chef de division DEVELOPPEMENT N. CALCHERA Ingénieurs en développement Cheffe de projet Chef de projet Chef de projet Chefs de projet A. VAN PETTEN Scrum Master D. AUVEILER S. DEGOUL X. BERCHEBRU X. BERCHEBRU Carole OBENICHE C. CEPEDA O. BARBECOT Responsable pôle qualification N. CALCHERA Ingénieure analyste H. CHOSSONNERIE Architecte S. DEGOUL B. ROSSI technique et TS BOURSE Ingénieur en Ingénieurs en Ingénieur en coordination ieurs en développeme développement développement Ingénieurs en dév. N. CARRARO Cheffe de projet R. CADILLON D. AUVEILER E. VIDAL développement D. AUVEILER E. GICQUEL S. DEGOUL R. CADILLON C. CEPEDA P.A. JACQUET Ingénieurs en développement C. OLEWSKI A. VAN PETTEN R. CADILLON E. PECHAUBES E. VIDAL T. NICOLAS MAINTENANCE Responsable Chefs de projet F. CLEMENCE V. AGUILERA, N. CALCHERA Domaine SIECLE Ingénieur en développement Ingénieur infra POLE CŒUR DE METIER POLE SIECLE + E. VIDAL, P.A. JACQUET F. AMIAUD Référente Référent POLE SYGNE - SYSCA L. TAUTOU E. SAVY POLE Ingénieurs en qualification AUTOMATISATION POLE FINANCIER QUALIFICATION M. ROMEU B. DESOLME Chef de projet Ingénieur en Ingénieurs en Ingénieurs en T. VIDAL O. EL MOUMEN qualification qualification qualification D. ZAMMIT N. MACHADO V. MACHARD T. OULION Responsable C. LECOMTE A définir E. ASSELIN Ingénieure en Ingénieurs en J. PASSELAIGUE C. OBENICHE T. OULION Ingénieurs en qualification qualification qualification ORIENTATION / C. PRAZAK B. ABBAS G. DUBOE-LAURENCE L. MARTINEZ DECROCHAGE C. PICARD Etienne POURCHON M. GARAVILLON-POLE LIVRETS stagiaire du 8 janvier Référente I. ZERMANI **TOURNEYRE** au 16 février 2024 C.OBENICHE Référente ngénieurs en qualification M. CHAPUT L. DE LA HERMOSA Ingénieur en qualification DIFFUSION K. KHARROUBI T. OULION C. LECOMTE Responsable Ingénieurs en diffusion C. LECOMTE F. AMIAUD M. NEE Internes Apprentis

D. AUVEILER

**EXPLOITATION** 

Prestataires



**Etienne POURCHON** 

# **AUTOMATISATION DES TESTS**

L'automatisation est principalement utilisée pour les tests de régression. Ces tests permettent de vérifier que le logiciel ne perde pas sa fonctionnalité après chaque modification apportée

# **Avantages**

- Améliorer la qualité d'un produit en détectant rapidement les bugs
- Gain de temps et de coûts sur le long terme en supprimant les tests manuels répétitifs
- Fiabilité
- ▶ Livraison continue

# Inconvénients

- Chronophage sur la mise en place
- Coût initial élevé
- > Limitation de couverture
- Rigidité
- Rétention des connaissances limité
- Maintenance des tests

# PRÉSENTATION DE L'IA

- L'IA est un programme qui cherche à reproduire l'intelligence humaine par le biais d'algorithmes afin d'automatiser des tâches ou de résoudre des problèmes complexes.
- Une des formes principales de l'IA est l'apprentissage automatique (machine learning) qui permet à l'ordinateur d'apprendre à partir de données.
- Une forme avancée de l'apprentissage automatique est l'apprentissage profond (deep learning) qui utilisent des réseaux neuronaux artificiels et permet à l'ordinateur d'apprendre par l'observation.
- L'IA générative qui repose sur l'apprentissage profond alimente 3 technologies :
  - la vision par ordinateur
  - la reconnaissance vocale
  - le traitement du langage naturel (TLN)

Un de outils les plus abouti d'IA générative est Chat GPT.

# IA APPLIQUÉE A L'AUTOMATISATION DES TESTS

# Avantages:

- Meilleure couverture des tests, peut choisir les tests à exécuter ou générer automatiquement des nouveaux cas de tests
- Détection précoce des anomalies
- Tests prédictifs, peut prédire les défauts avant qu'ils ne se produisent
- Auto-réparatrice, peut corriger ou réécrire des scripts d'automatisation de tests, ce qui réduit la maintenance manuelle des tests
- Efficacité et précision accrue

# Inconvénients:

- Besoin de nombreuses données pour être efficace et fiable
- Mise en œuvre complexe
- Besoin de personnel expérimenté

# CONCLUSION

- L'IA est en plein développement et son utilité dans l'automatisation des tests présente un réel intérêt voire une nécessité dans le futur afin de s'adapter au développement de plus en plus rapide des logiciels.
- Avec le développement de l'IA, le métier de testeur sera amené à évoluer.

# Annexe 4 – Fil d'échanges avec Chat GPT

#### Essai n'1

#### You:

Dans l'optique de faire des tests automatisés formels, que devrais-je faire pour tester une application affichant un tableau de données, contenant différents filtres et ordres de tri

#### ChatGPT:

Pour tester une application affichant un tableau de données avec différents filtres et ordres de tri de manière automatisée et formelle, vous pouvez suivre plusieurs étapes :

# 1. Analyse des exigences :

- Comprendre les spécifications fonctionnelles de l'application, y compris les exigences pour les filtres et les ordres de tri.
- Identifier les différentes combinaisons possibles de filtres et d'ordres de tri.

#### 2. Sélection d'un outil de test automatisé :

- Choisissez un outil de test automatisé qui prend en charge le type d'application que vous testez (web, mobile, etc.).
- Assurez-vous que l'outil peut interagir avec les éléments du tableau de données, appliquer des filtres et vérifier les résultats.

# 3. Développement des scripts de test :

- Créez des scripts de test qui automatisent les actions utilisateur pour appliquer les filtres et les ordres de tri.
- Assurez-vous que les scripts de test couvrent toutes les combinaisons importantes de filtres et d'ordres de tri.

# 4. Mise en place des données de test :

- Préparez un ensemble de données de test qui permettent de valider les résultats du tableau dans différentes situations.
- Assurez-vous d'avoir des données qui couvrent toutes les conditions possibles de filtrage et de tri.

## 5. Exécution des tests :

- Exécutez les scripts de test sur l'application.
- Vérifiez que les résultats du tableau correspondent aux attentes définies pour chaque combinaison de filtres et d'ordres de tri.

# 6. Gestion des rapports de test :

- Générez des rapports de test détaillés qui indiquent les résultats de chaque scénario de test.
- Identifiez les éventuels problèmes ou écarts par rapport aux attentes.

# 7. Automatisation continue et maintenance :

- Intégrez les tests automatisés dans votre pipeline de développement pour les exécuter régulièrement.
- Mettez à jour les scripts de test lorsque de nouvelles fonctionnalités sont ajoutées ou que des modifications sont apportées à l'application.

Etienne POURCHON

En suivant ces étapes, vous pouvez mettre en place un processus de test automatisé formel pour valider efficacement l'affichage du tableau de données avec différents filtres et ordres de tri dans votre application.

#### You:

L'application peut ordonner suivant 10 éléments les résultats (Code, Type, Sigle, Patronyme, Commune, Contrat, Date de diffusion, Profilage MEF, Profilage Langues, Profilage ETP) et rechercher à l'aide de 8 éléments (Code, Type, Profilage, Commune, Code MEF, libelle MEF, MEFSTAT, Matière), comment faire un test automatisé avec ses informations et rédiger le tout sur l'application SQUASH

#### ChatGPT:

Pour automatiser les tests de votre application utilisant Squash, vous pouvez suivre les étapes suivantes :

# Étape 1 : Identifier les scénarios de test

# 1. Filtrage:

- Filtrer les résultats par chacun des 8 éléments de recherche.
- Vérifier que les résultats affichés correspondent aux critères de recherche.

# 2. **Tri**:

- Trier les résultats par chacun des 10 éléments d'ordonnancement.
- · Vérifier que les résultats sont triés dans le bon ordre.

# Étape 2 : Préparation de l'environnement de test

1. Assurez-vous que l'environnement de test est configuré avec les données nécessaires pour exécuter les tests.

# Étape 3 : Rédaction des scripts de test dans Squash

- 1. Créez un nouveau cas de test pour chaque scénario identifié.
- 2. Pour chaque cas de test :
  - Définissez les étapes de test, par exemple :
    - Sélectionner un critère de recherche.
    - Appliquer le filtre ou trier les résultats.
    - Vérifier que les résultats correspondent aux attentes.

# Étape 4 : Exécution des tests

- 1. Exécutez les cas de test dans Squash.
- 2. Notez les résultats de chaque cas de test, enregistrant les réussites et les échecs.

# Étape 5 : Analyse des résultats

1. Analysez les résultats des tests pour identifier les problèmes éventuels.

2. Pour les cas de test qui échouent, identifiez les raisons sous-jacentes et documentezles.

# Étape 6 : Génération de rapports

- 1. Générez des rapports détaillés sur l'exécution des tests.
- 2. Incluez les résultats des tests, les problèmes identifiés et les étapes de reproduction des erreurs.

# Étape 7 : Répétez le processus

- 1. Effectuez des itérations de test supplémentaires pour valider les corrections et les nouvelles fonctionnalités.
- 2. Mettez à jour les cas de test en fonction des changements apportés à l'application.

En suivant ces étapes, vous pourrez automatiser les tests de votre application avec Squash en utilisant les informations sur les filtres et les ordres de tri disponibles.

#### You:

Un exemple de scenario sur Squash est il possible

## ChatGPT:

Bien sûr, voici un exemple de scénario de test pour Squash basé sur le filtrage des résultats de l'application :

Scénario de test : Filtrage par code postal

# Objectif:

Vérifier que l'application filtre correctement les résultats en fonction du code postal saisi.

# Préreguis:

- L'application est ouverte et fonctionnelle.
- Des données de test sont disponibles dans l'application.

# Étapes de test :

# 1. Saisie du code postal :

- · Ouvrir l'application.
- Accéder à l'écran de recherche.
- Localiser le champ de recherche par code postal.
- Saisir un code postal valide dans le champ de recherche.

# 2. Application du filtre :

- Cliquer sur le bouton de recherche ou appuyer sur la touche "Entrée".
- Attendre que les résultats se chargent.

# 3. Vérification des résultats :

 Vérifier que seuls les résultats correspondant au code postal saisi sont affichés.  Examiner chaque résultat pour s'assurer que le code postal correspond à celui saisi.

# 4. Vérification des résultats non pertinents :

- Vérifier qu'aucun résultat ne contient un code postal différent de celui saisi.
- S'assurer qu'aucun résultat n'est affiché s'il n'y a pas de correspondance avec le code postal saisi.

# Résultat attendu:

- Les résultats affichés correspondent uniquement au code postal saisi.
- Aucun résultat ne contient de code postal différent.
- Aucun résultat n'est affiché s'il n'y a pas de correspondance avec le code postal saisi.

# Conditions de sortie :

• Les résultats du filtrage par code postal sont conformes aux attentes.

Ce scénario de test peut être implémenté dans Squash en créant un cas de test avec ces étapes détaillées. En exécutant ce cas de test dans Squash, vous pourrez vérifier si l'application fonctionne correctement selon les critères définis.

#### You

Un exemple sans utiliser de filtre

#### ChatGPT:

D'accord, voici un exemple de scénario de test pour Squash qui ne nécessite pas l'utilisation de filtres, mais plutôt se concentre sur le tri des résultats :

Scénario de test : Tri par ordre alphabétique des noms de commune

# Objectif:

Vérifier que l'application trie correctement les résultats par ordre alphabétique des noms de commune.

# Prérequis:

- L'application est ouverte et fonctionnelle.
- Des données de test sont disponibles dans l'application.

# Étapes de test :

# 1. Accès aux résultats :

- Ouvrir l'application.
- Accéder à l'écran des résultats affichant les communes.

#### 2. Vérification de l'ordre initial :

 Vérifier l'ordre initial des résultats des communes, avant tout tri. Prendre note du premier et du dernier résultat dans cet ordre.

# 3. Sélection du tri par nom de commune :

- Rechercher l'option de tri par nom de commune.
- Sélectionner cette option.

# 4. Application du tri:

 Appliquer le tri en cliquant sur le bouton de tri ou en sélectionnant l'option de tri.

# 5. Vérification de l'ordre trié :

- Attendre que les résultats se rechargent après l'application du tri.
- Vérifier que les résultats sont maintenant triés par ordre alphabétique des noms de commune.

# 6. Vérification de la cohérence de tri :

- Vérifier que le premier résultat est le nom de commune qui vient en premier dans l'ordre alphabétique.
- Vérifier que le dernier résultat est le nom de commune qui vient en dernier dans l'ordre alphabétique.

# Résultat attendu:

- Les résultats sont triés par ordre alphabétique des noms de commune, du premier au dernier.
- Le premier résultat est le nom de commune qui vient en premier dans l'ordre alphabétique.
- Le dernier résultat est le nom de commune qui vient en dernier dans l'ordre alphabétique.

# Conditions de sortie :

• Les résultats du tri par nom de commune sont conformes aux attentes.

Ce scénario de test peut être créé dans Squash en tant que cas de test, avec les étapes détaillées ci-dessus. Son exécution permettra de vérifier si l'application trie correctement les résultats selon les critères spécifiés.

# Essai n°2:

# You:

Un écran affiche la liste de toutes les matières nationales et académiques qui sont fermées ou ouvertes, listant le code interne, la clé gestion et le libelle de chaque matière dans un tableau. L'utilisateur peut modifier l'affichage par défaut en choisissant un tri croissant ou décroissant sur la colonne Ouvert/fermé, code interne, clé de gestion, libellé long. L'affichage peut être demandé à partir d'une occurrence précise en utilisant le bouton <<Chercher>>. Le filtre est possible sur le code interne, clé de gestion, et libellé long ou sur l'état ouvert/fermé/Tous.

# ChatGPT:

Voici un exemple de spécification pour l'interface utilisateur basée sur vos besoins :

# Interface Utilisateur - Liste des Matières

# Options de Tri

- Tri par état (Ouvert/Fermé)
- Tri par Code Interne
- Tri par Clé de Gestion
- Tri par Libellé Long

# **Actions Disponibles**

- Chercher: Permet de trouver une occurrence précise en utilisant des filtres.
- **Filtrer**: Filtrer les résultats par Code Interne, Clé de Gestion, Libellé Long ou par état (Ouvert/Fermé/Tous).

# Tableau des Matières

Code Interne	Clé de Gestion	Libellé Long	État
001	XYZ123	Mathématiques	Ouvert
002	ABC456	Français	Fermé
003	DEF789	Physique	Ouvert

# **Actions**

 Modifier Affichage: Permet de changer le tri par défaut (croissant/décroissant) sur chaque colonne.

Cette spécification fournit une vue claire des fonctionnalités disponibles pour l'utilisateur, y compris les options de tri, les actions possibles et les données affichées dans le tableau des matières.

## You:

Avec ses info, On souhaite créer un test automatisé de l'application en java et Rédiger les étapes/résultats attendu sur Squash, comment procéder

### ChatGPT:

Pour créer un test automatisé de l'application en Java et rédiger les étapes/résultats attendus sur Squash, vous pouvez suivre ces étapes :

# Étapes pour Créer un Test Automatisé en Java :

# 1. Choix des Outils :

 Utilisez un framework de test comme JUnit ou TestNG pour écrire et exécuter vos tests. • Choisissez une bibliothèque de test d'interface utilisateur comme Selenium ou Appium pour tester l'interface utilisateur de votre application.

# 2. Configuration de l'Environnement :

 Configurez votre projet Java avec les dépendances nécessaires pour les outils de test choisis.

## 3. Écriture des Tests :

- Écrivez des cas de test pour chaque fonctionnalité de votre application en utilisant les fonctionnalités du framework de test choisi.
- Pour les tests d'interface utilisateur, écrivez des scripts qui simulent les interactions de l'utilisateur avec votre application.

## 4. Exécution des Tests :

• Exécutez vos tests automatisés pour vérifier le bon fonctionnement de votre application.

# Rédaction des Étapes/Résultats Attendus sur Squash :

#### 1. Création d'un Nouveau Cas de Test :

 Créez un nouveau cas de test sur Squash pour chaque fonctionnalité que vous testez.

# 2. Description du Cas de Test :

 Décrivez l'objectif du test, les prérequis éventuels et les données d'entrée nécessaires.

# 3. Étapes du Test :

• Énumérez les étapes spécifiques que le test doit effectuer pour vérifier la fonctionnalité. Par exemple : ouvrir l'application, cliquer sur un bouton, saisir des données, etc.

#### 4. Résultats Attendus :

• Spécifiez les résultats attendus pour chaque étape du test. Cela peut inclure des vérifications d'interface utilisateur, des validations de données, etc.

#### 5. État de Validation :

 Indiquez si chaque étape du test a réussi ou échoué, ainsi que les raisons des échecs éventuels.

### 6. Conclusion:

 Résumez les résultats du test et indiquez si la fonctionnalité testée est conforme aux attentes.

# Exemple de Rédaction sur Squash :

Nom du Cas de Test : Vérification du Tri par Code Interne

**Description**: Vérifie que les matières sont triées correctement en fonction du code interne dans l'application.

# Étapes :

- 1. Sélectionner l'option de tri par code interne.
- 2. Vérifier que les matières sont affichées dans l'ordre croissant du code interne.

# Résultats Attendus :

- Toutes les matières sont affichées dans l'ordre croissant du code interne.
- Le premier élément dans la liste a le code interne le plus bas.
- Le dernier élément dans la liste a le code interne le plus élevé.

# État de Validation :

- Étape 1 : Réussi
- Étape 2 : Réussi

**Conclusion :** Le tri par code interne fonctionne comme prévu, les matières sont correctement triées dans l'ordre croissant du code interne.

En suivant ces étapes, vous pouvez créer des tests automatisés en Java et documenter les étapes et résultats attendus sur Squash de manière claire et précise.