

## RAPPORT DE PROJET DE FIN D'ÉTUDE

Présenté en vue de l'obtention de la  
LICENCE EN GÉNIE LOGICIEL ET SYSTÈMES D'INFORMATIUS

Parcours : Génie logiciel et Systèmes d'informations

---

# Conception et développement d'un framework d'automatisation de test

---

*Par*  
RANA SOLTANI

NADA EL MOKHTAR

Réalisé au sein d'Allence Tunisie



Soutenu publiquement le 28 mai 2022 devant le jury composé de :

Président : M. Mohamed Wahb OURTANI

Rapporteur : Mme. Asma NAJJAR

Examinateur : Mme. Mohja BEN AMARA

Encadrant professionnel : M. Walid MENSIA

Encadrant académique : Mme. Sabrine NAIMI

Année Universitaire : 2021 - 2022

## RAPPORT DE PROJET DE FIN D'ÉTUDE

Présenté en vue de l'obtention de la  
LICENCE EN GÉNIE LOGICIEL ET SYSTÈMES D'INFORMATIUS

Parcours : Génie logiciel et Systèmes d'informations

---

# Conception et développement d'un framework d'automatisation de test

---

*Par*  
RANA SOLTANI

NADA EL MOKHTAR

Réalisé au sein d'Allence Tunisie



Autorisation de dépôt du rapport de Projet de Fin d'Etudes :

---

Encadrant professionnel :  
M. Walid MENSIA

Encadrant académique :  
Mme. Sabrine NAIMI

Le :

Le :

Signature :

Signature :

# Dédicaces

Du profond de nos cœurs, nous dédions ce modeste travail :

- A nos chers parents pour leur patience, amour, soutien et leurs encouragements.
- A nos professeurs d'avoir enrichi nos connaissances et de nous avoir guidé durant les trois années.
- A notre cher collègue **Mohamed Taher Hlaoui** qui nous a accompagné, aidé, encouragé tout au long de la réalisation de notre projet.
- A une personne qui nous a beaucoup aidé à comprendre comment gérer les problématiques de notre travail : **Monsieur Amine Krimi**.

# Remerciement

On remercie dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce projet.

On remercie notre Maître de stage, Monsieur **Walid Mensia**, pour son accueil. Grâce aussi à sa confiance, nous avons pu accomplir totalement dans notre mission.

Ce travail ne serait pas aussi riche sans l'aide et l'encadrement de Madame **Sabrine Naimi**, nous remercions pour la qualité de son encadrement exceptionnel, pour sa patience, sa rigueur et sa disponibilité durant notre préparation de ce projet.

Nous tenons à remercier vivement les membres de jury qui ont accepté d'évaluer notre travail .Merci également pour leur contribution par leurs présieuses remarques et conseils à améliorer notre projet.

Nous remercions également toute l'équipe d'Allence tunisie pour leur accueil, leur esprit d'équipe et en particulier Madame **Hajer Ayari** pour son aide pratique, son soutien moral et ses encouragements.

Enfin, nous tenons à remercier toutes les personnes qui ont contribué à l'élaboration de ce travail : nos familles et nos amis.

# Table des matières

|  |           |
|--|-----------|
| <b>Introduction générale</b>   | <b>1</b>  |
| <b>1 Contexte du projet</b>  | <b>3</b>  |
| 1.2 Introduction . . . . .   | 3         |
| 1.3 Présentation générale de l'organisme d'accueil . . . . .           | 3         |
| 1.4 Etude de l'existant . . . . .                                      | 4         |
| 1.4.1 Solution actuelle adoptée par l'entreprise . . . . .             | 4         |
| 1.4.2 Critiques de l'existant . . . . .                                | 5         |
| 1.4.3 Solution proposée . . . . .                                      | 5         |
| 1.5 Méthodologie de travail . . . . .                                  | 6         |
| 1.6 Conclusion . . . . .   | 6         |
| <b>2 Analyse des besoins</b>   | <b>7</b>  |
| 2.1 Introduction . . . . .   | 7         |
| 2.2 Identification des besoins . . . . .                               | 7         |
| 2.2.1 Besoins fonctionnels . . . . .                                   | 7         |
| 2.2.2 Besoins non fonctionnels . . . . .                               | 7         |
| 2.3 Technologies et architecture utilisées . . . . .                   | 8         |
| 2.3.1 Technologies utilisées pour l'automatisation des tests . . . . . | 8         |
| 2.3.2 Architecture utilisée pour l'automatisation des tests . . . . .  | 10        |
| 2.4 Identification et structuration des cas d'utilisation . . . . .    | 11        |
| 2.4.1 Identification des acteurs . . . . .                             | 11        |
| 2.4.2 Diagramme de cas d'utilisation globale . . . . .                 | 12        |
| 2.4.3 Backlog de Produit . . . . .                                     | 12        |
| 2.4.4 Diagramme de classe globale . . . . .                            | 13        |
| 2.5 Conclusion . . . . .   | 14        |
| <b>3 Release 1</b>   | <b>15</b> |
| 3.1 Introduction . . . . .   | 15        |
| 3.2 Identification de backlog de release1 . . . . .                    | 15        |
| 3.2.1 Préparation de l'environnement . . . . .                         | 15        |
| 3.2.2 Diagrammes de cas d'utilisation détaillés de release1 . . . . .  | 17        |
| 3.2.3 Description détaillée du Release1 . . . . .                      | 19        |
| 3.2.4 Diagramme de séquence . . . . .                                  | 20        |
| 3.3 Réalisation . . . . .  | 20        |
| 3.3.1 Test manuel de l'authentification . . . . .                      | 20        |
| 3.3.2 Test automatique de l'authentification . . . . .                 | 22        |
| 3.4 Conclusion . . . . .   | 23        |

|   |           |
|---|-----------|
| <b>4 Release 2</b>  | <b>24</b> |
| 4.1 Introduction . . . . .  | 24        |
| 4.2 Identification de backlog de Release 2 . . . . .                          | 24        |
| 4.2.1 Diagrammes de cas d'utilisation détaillés de Release 2 . . . . .        | 25        |
| 4.2.2 Description détaillée du Release 2 . . . . .                            | 28        |
| 4.2.3 Diagrammes de séquence du Release 2 . . . . .                           | 28        |
| 4.3 Réalisation du Release 2 . . . . .  | 31        |
| 4.3.1 Interfaces à tester . . . . .   | 31        |
| 4.3.2 Interface Xray . . . . .  | 35        |
| 4.3.3 Interfaces du fichier "automation.log" . . . . .                        | 37        |
| 4.3.4 Interface Jenkins . . . . .   | 39        |
| 4.4 Conclusion . . . . .  | 41        |
| <b>5 Release 3</b>  | <b>42</b> |
| 5.1 Introduction . . . . .  | 42        |
| 5.2 Identification de backlog de release3 . . . . .                           | 42        |
| 5.2.1 Diagrammes de cas d'utilisation détaillés de Relese 3 . . . . .         | 43        |
| 5.2.2 Description détaillée du Release 3 . . . . .                            | 45        |
| 5.2.3 Diagrammes de séquence du Release 3 . . . . .                           | 46        |
| 5.3 Réalisation . . . . .   | 51        |
| 5.3.1 Interfaces à tester . . . . .   | 51        |
| 5.3.2 Interfaces Xray des tests manuels . . . . .                             | 52        |
| 5.3.3 Interfaces du fichier "automation.log" des tests automatiques . . . . . | 54        |
| 5.3.4 Interfaces Jenkins des tests automatiques . . . . .                     | 56        |
| 5.4 Conclusion . . . . .  | 57        |
| <b>Conclusion Générale</b>  | <b>58</b> |
| <b>Bibliographie</b>  | <b>60</b> |

# Table des figures

|      |  |    |
|------|--|----|
| 1.1  | Société "Allence Tunisie" . . . . .  | 4  |
| 1.2  | Cahier de Test de «Chosa» . . . . .  | 5  |
| 2.3  | Logo de Seleniuù Webdriver . . . . .   | 8  |
| 2.4  | Logo de Python . . . . .   | 9  |
| 2.5  | Logo de Jenkins . . . . .  | 9  |
| 2.6  | Logo de Xray . . . . .   | 9  |
| 2.7  | Logo de Gitlab . . . . .   | 10 |
| 2.8  | Architecture CI/CD . . . . .   | 10 |
| 2.9  | Diagramme de cas d'utilisation globale . . . . .   | 12 |
| 2.10 | Diagramme de classe global . . . . .   | 13 |
| 3.11 | Structure de familiarisation de Python avec Selenium . . . . .   | 16 |
| 3.12 | Structure de Framework . . . . .   | 17 |
| 3.13 | Diagramme de cas d'utilisation «Test manuel de l'authentification» . . . . .                                 | 18 |
| 3.14 | Diagramme de cas d'utilisation «Test automatique de l'authentification» . . . . .                            | 18 |
| 3.15 | Diagramme de séquence détaillé «Authentification» . . . . .  | 20 |
| 3.16 | Interface Authentification . . . . .   | 21 |
| 3.17 | Interface Authentification via Facebook . . . . .  | 21 |
| 3.18 | Réultat sur Jira dashboard de test manuel authentification . . . . .   | 22 |
| 3.19 | Description de fichier "automation.log" . . . . .  | 22 |
| 3.20 | Interface de rapport html de test authentification . . . . .   | 23 |
| 4.21 | Diagramme de cas d'utilisation «tests manuels et automatiques de l'espace parent» . . . . .                  | 26 |
| 4.22 | Diagramme de cas d'utilisation «Implémenter et exécuter les tests manuels de l'espace parent» . . . . .      | 26 |
| 4.23 | Diagramme de cas d'utilisation «Implémenter et exécuter les tests automatiques de l'espace parent» . . . . . | 27 |
| 4.24 | Diagramme de séquence détaillé «tester l'ajout d'un enfant dans l'espace parent» . . . . .                   | 29 |
| 4.25 | Diagramme de séquence détaillé «tester l'ajout d'un jardin dans l'espace parent» . . . . .                   | 29 |
| 4.26 | Diagramme de séquence détaillé «tester l'ajout d'un statut dans l'espace parent» . . . . .                   | 30 |
| 4.27 | Diagramme de séquence détaillé «tester l'ajout d'un membre de la famille dans l'espace parent» . . . . .     | 30 |
| 4.28 | Diagramme de séquence détaillé «tester l'ajout d'un message dans l'espace parent» . . . . .                  | 31 |
| 4.29 | Interface «Ajout enfant» testée . . . . .  | 32 |
| 4.30 | Suite de l'interface «Ajout enfant» testée . . . . .   | 32 |
| 4.31 | Interface «Ajout jardin» testée . . . . .  | 33 |

|   |    |
|---|----|
| 4.32 Interface «Ajout statut» testée . . . . .  | 33 |
| 4.33 Suite de l'interface «Ajout statut» testée . . . . .   | 34 |
| 4.34 Interface «Ajout membre» testée . . . . .  | 34 |
| 4.35 Suite de l'interface «Ajout membre» testée . . . . .   | 35 |
| 4.36 Interface Xray de l'ajout d'un enfant . . . . .  | 35 |
| 4.37 Interface Xray de l'ajout d'un jardin . . . . .  | 36 |
| 4.38 Interface Xray de l'ajout d'un statut . . . . .  | 36 |
| 4.39 Interface Xray de l'ajout d'un membre . . . . .  | 37 |
| 4.40 Description du «test ajout enfant» dans fichier "automation.log" . . . . .                                   | 37 |
| 4.41 Description du «test ajout jardin» dans fichier automation.log . . . . .                                     | 38 |
| 4.42 Description du «test ajout statut» dans fichier "automation.log" . . . . .                                   | 38 |
| 4.43 Description du «test ajout membre» dans fichier "automation.log" . . . . .                                   | 39 |
| 4.44 Interface de configuration Jenkins avec Gitlab . . . . .   | 39 |
| 4.45 Interface build de Jenkins . . . . .   | 40 |
| 4.46 Résultat de l'exécution de test «Ajout enfant» sur console Jenkins . . . . .                                 | 40 |
| 4.47 Résultat de l'exécution de test «Ajout jardin» sur console Jenkins . . . . .                                 | 41 |
| 4.48 Résultat de l'exécution de test «Ajout membre» sur console Jenkins . . . . .                                 | 41 |
| 5.49 Diagramme de cas d'utilisation «tests manuels et automatiques de l'espace jardin» . . . . .                  | 44 |
| 5.50 Diagramme de cas d'utilisation «Implémenter et exécuter les tests manuels de l'espace jardin» . . . . .      | 44 |
| 5.51 Diagramme de cas d'utilisation «Implémenter et exécuter les tests automatiques de l'espace jardin» . . . . . | 45 |
| 5.52 Diagramme de séquence détaillé «tester l'ajout et suppression d'un enfant dans l'espace jardin» . . . . .    | 47 |
| 5.53 Diagramme de séquence détaillé «tester l'ajout d'une activité dans l'espace jardin» . . . . .                | 48 |
| 5.54 Diagramme de séquence détaillé «tester l'ajout d'un enseignant dans l'espace jardin» . . . . .               | 49 |
| 5.55 Diagramme de séquence détaillé «tester l'ajout d'une classe dans l'espace jardin» . . . . .                  | 50 |
| 5.56 Interface «Ajout activité» testée . . . . .  | 51 |
| 5.57 Interface «Ajout enseignant» testée . . . . .  | 51 |
| 5.58 Interface «Ajout classe» testée . . . . .  | 52 |
| 5.59 Interface Xray de l'ajout d'un enfant . . . . .  | 52 |
| 5.60 Interface Xray de l'ajout d'une activité . . . . .   | 53 |
| 5.61 Interface Xray de l'ajout d'un enseignant . . . . .  | 53 |
| 5.62 Interface Xray de l'ajout d'une classe . . . . .   | 54 |
| 5.63 Description du «test ajout enfant» dans fichier "automation.log" . . . . .                                   | 54 |
| 5.64 Description du «test ajout activité» dans fichier "automation.log" . . . . .                                 | 55 |
| 5.65 Description du «test ajout enseignant» dans fichier "automation.log" . . . . .                               | 55 |
| 5.66 Description du «test ajout classe» dans fichier "automation.log" . . . . .                                   | 55 |
| 5.67 Interface de console Jenkins après l'exécution des tests du Release3 . . . . .                               | 56 |
| 5.68 Résultat de test :Exécution de test automatique sur Jenkins . . . . .  | 56 |
| 5.69 Interface de console Jenkins après l'exécution des tests du Release3 . . . . .                               | 57 |
| 5.70 Interface de rapport html de tous les tests . . . . .  | 57 |

# Liste des tableaux

|  |    |
|--|----|
| 2.1 Identification des acteurs . . . . .   | 11 |
| 2.2 Backlog de produit . . . . .   | 13 |
| 3.3 Identification de Backlog de Release 1 . . . . .                               | 15 |
| 3.4 Description du cas d'utilisation «Test manuel de l'authentification» . . . . . | 19 |
| 3.5 Description du cas d'utilisation «Test automatique de l'authentification» . .  | 19 |
| 4.6 Identification de Backlog de release 2 . . . . .                               | 25 |
| 4.7 Description du cas d'utilisation «Tests manuels de l'espace parent» . . . . .  | 28 |
| 4.8 Description du cas d'utilisation «Test automatique de l'espace parent» . .     | 28 |
| 5.9 Identification de Backlog de release 3 . . . . .                               | 43 |
| 5.10 Description du cas d'utilisation «Tests manuels de l'espace jardin» . . . . . | 46 |
| 5.11 Description du cas d'utilisation «Test automatique de l'espace jardin» . . .  | 46 |

# Introduction générale

Dans le monde du développement logiciel, nous parlons beaucoup de développeurs, d'architectes, de chefs de projets, etc. Aujourd'hui, un nouveau profil est mis en avant : celui de testeur valideur qui est devenu un métier à part entière. La preuve en est que dans des sociétés comme Microsoft, où les logiciels sont déployés par dizaines de millions d'exemplaires, certaines équipes comportent jusqu'à deux testeurs par développeur ! Leur objectif principal est d'identifier un nombre maximal de problématiques du logiciel. Ils permettent ainsi, dès lors que les problèmes identifiés seront corrigés, d'en augmenter la qualité du produit sur une période déterminée. Ils donnent aussi des indications de fiabilité. Tout cela, permet de vérifier l'adéquation du produit aux besoins exprimés par le client et donc de satisfaire la demande du client.

La question qui s'oppose à ce niveau est : Comment tester notre produit ?

Le test ressemble à une expérience scientifique. Il examine une hypothèse exprimée en fonction de trois éléments : les données en entrée, l'objet à tester et les observations attendues. Cet examen est effectué sous conditions contrôlées pour pouvoir tirer des conclusions et, dans l'idéal, être reproduit.

En outre, pour déployer un meilleur logiciel et trouver les bogues qui affectent le développement d'applications, nous devons tout d'abord faire la distinction entre les deux principaux types de tests : les tests manuels et les tests automatisés.

- Les tests manuels sont effectués en personne, en cliquant dans l'application ou en interagissant avec le logiciel et les API avec les outils appropriés. Cette méthode est très coûteuse, car il faut quelqu'un pour configurer un environnement et exécuter les tests, et elle peut être sujette à l'erreur humaine, car le testeur peut faire des fautes de frappe ou omettre des étapes dans le script de test.

- Les tests automatisés, d'autre part, sont effectués par une machine qui exécute un script de test programmé à l'avance. Ces tests peuvent énormément varier en termes de complexité, de la vérification d'une seule méthode dans une classe jusqu'à s'assurer que l'exécution d'une séquence d'actions complexes dans l'interface utilisateur mène aux mêmes résultats. Cette méthode est beaucoup plus robuste et fiable que les tests manuels, mais la qualité de vos tests automatisés dépend de la qualité de vos scripts de test.

Dans le cadre de notre projet, Allence Tunisie nous propose de tester le fonctionnement d'une application mobile en cours de développement afin de détecter des erreurs et accroître les chances d'obtenir un logiciel qui corresponde aux objectifs (cahier des charges).

Le présent rapport présente notre travail ainsi que les différentes étapes que nous allons suivre pour réaliser ce projet. Il s'appuie sur quatre chapitres :

**-chapitre 1** intitulé « Cadre de Projet » qui présente l'organisme d'accueil, expose une

étude de l'existant et la problématique et souligne notre méthodologie adoptée.

**-chapitre 2** intitulé « identification des besoins » qui détaille la spécification des besoins de notre projet, les acteurs ainsi que le diagramme de cas d'utilisation et le backlog produit.

**-chapitre 3** intitulé « release1 » qui décrit le premier release.

**-chapitre 4** intitulé « release2 » qui décrit le deuxième release.

**-chapitre 5** intitulé « release3 » qui décrit le troisième release.

# **chapitre 1**

## **Cadre de projet**

### **1.2 Introduction**

Dans le premier chapitre, nous commençons par mettre l'accent sur le contexte général de notre projet. Ensuite, nous présentons l'organisme d'accueil au sein duquel s'est déroulé notre projet de fin d'étude. Après, nous faisons une étude de l'existant qui nous donne un aperçu sur la problématique. L'analyse de cette dernière nous permet de définir les objectifs à atteindre pour réaliser notre solution. Et enfin, nous proposons la méthodologie adoptée dans notre projet.

### **1.3 Présentation générale de l'organisme d'accueil**

Allence Tunisie est une société franco-tunisienne de développement de logiciels. Elle s'occupe des services de développement de solutions.

Au moins 10 ans d'expérience, elle a travaillé avec des sociétés de développement de logiciels, des startups, des agences numériques et des entreprises afin de les aider à simplifier leur expérience dans l'externalisation informatique et réduire les coûts et le temps de mise sur le marché. [a].

Grâce à la patience du jeune personnel talentueux et à la confiance des fondateurs d'Allence Tunisie nombreux problème et obstacles ont été surmontés et relevés. Cette aventure attire de plus en plus des nouveaux partenaires. En effet, des ingénieurs et des architectes intelligents et des personnels Sélectionnés viennent rejoindre Allence Tunisie afin de répondre aux attentes de ses Clients et d'assurer la qualité d'exécution.

Allence Tunisie croit en l'esprit d'échange et de partage des informations. Elle n'hésite pas d'intégrer des nouveaux partenaires et d'échanger différents points de vue. C'est sa raison de l'innovation et de l'intégration de nouvelles technologies ce qui lui permet de résoudre les problèmes et trouver des solutions adaptées aux problèmes à résoudre.

La figure 1.1 présente la société Allence Tunisie ainsi que ces objectifs.



FIGURE 1.1 – Société "Allence Tunisie"

## 1.4 Etude de l'existant

Dans cette section, nous décrivons l'approche actuelle de la société Allence Tunisie, ses avis et ses solutions proposées afin de pouvoir comprendre les lacunes du système existant et clarifier nos objectifs d'amélioration.

### 1.4.1 Solution actuelle adoptée par l'entreprise

Allence Tunisie offre comme service le développement de logiciels en passant généralement par le test manuel dont le scénario est déroulé par un être humain. C'est un processus de « recherche manuelle des défauts d'un logiciel ». Grâce à cette méthode, les testeurs se mettent à la place des utilisateurs finaux pour vérifier que tout fonctionne correctement avant la mise en production du logiciel.

Il est important de distinguer deux types de scénarios possible à réaliser :

- Soit le scénario est écrit, c'est alors le cas d'un test guidé. Les étapes et les vérifications ont donc préalablement été écrites et sont juste suivies à la lettre.
- Soit le scénario n'est pas écrit, c'est alors le cas d'un test exploratoire. Le testeur fait alors preuve de sérendipité en partant à la découverte du produit, comme peut le faire l'utilisateur final. Il effectue des actions réfléchies sur l'instant et analyse les résultats en les comparant à ceux attendus en tant qu'utilisateur réel.

Actuellement, Allence Tunisie travaille avec des tests manuels, dont le scénario est écrit. La figure 1.2 présente un exemple de cahier de test qui a guidé le testeur pour contrôler le bon fonctionnement d'une application «Chosa» sur des points bien précis.

| Via Facebook  | OK/KO |  |
|---|-------|--|
| <b>1.Connexion PARENT</b><br>1.1 Ajout enfant<br>1.1 Ajout photo de profil<br>1.2 Ajout avatar<br>1.3 Ajout statut<br>1.4 Ajout image<br>1.5 Ajout PDF<br>1.6 Ajout mp3<br>1.7 Ajout photos multiples<br>1.8 Ajout vidéo<br>1.9 Suppression enfant<br>1.10 Affichage code de l'enfant<br>1.11 ajout membre famille              |       |  |
| <b>2.Ajout jardin</b><br>2.1 Ajout enfant dans jardin<br>2.2 Ajout statut<br>2.3 Ajout image<br>2.4 Ajout PDF<br>2.5 Ajout vidéo<br>2.5 Ajout mp3<br>2.6 Ajout photo multiple   |       |  |
| <b>3.AJOUT ENSEIGNANT</b>   |       |  |
| <b>4.AJOUT CLASSE</b><br>4.1 AJOUT ENFANT DANS CLASSE<br>4.2 SUPPRIMER INFANT DE LA CLASSE<br>4.3 AJOUT ACTIVITE PHOTO A UNE CLASSE<br>4.4 AJOUTE PDF A UNE CLASSE<br>4.5 AJOUTER IMAGE A UNE CLASSE<br>4.6 AJOUTER MP3 A UNE CLASSE<br>4.7 AJOUT PHOTO MULTIPLE A UNE CLASSE<br>4.8 SUPPRIMER CLASSE<br>4.9 supprimer activité |       |  |
| <b>5.CODE PARENT</b>  |       |  |
| <b>6.NOTIFICATIONS</b><br>7.MESSAGE   |       |  |

FIGURE 1.2 – Cahier de Test de «Chosa»

### 1.4.2 Critiques de l'existant

Le test manuel nécessite peu de budget, ne demande aucune connaissance en outils de test. Il est effectué par un humain, assis devant une machine et exécutant avec prudence ses étapes du cycle de vie, ce qui engendre un manque de fiabilité et de précision. Aussi, il peut prendre beaucoup de temps car il ne peut pas être répliqué facilement. Encore, cette tâche, est indispensable pour assurer une certaine qualité de produit. En effet, un défaut dans une application peut avoir des conséquences plus ou moins graves pour les clients et l'entreprise. La correction peut s'accompagner d'un coût supplémentaire qui n'a pas été prévu initialement. Dans ce cas, l'automatisation présente une solution pour résoudre tous ces problèmes.

### 1.4.3 Solution proposée

Les méthodes de développement sont de plus en plus en évolution ce qui nécessite une réflexion vis-à-vis les processus de test. Dans un environnement toujours plus concurrentiel, l'enjeu des tests est fondamental. Mais les exécuter manuellement régulièrement, toutes les semaines voir tous les jours, devient impossible sans faire exploser les coûts et sans faire perdre l'envie des testeurs. L'automatisation des tests est la solution pour améliorer la qualité des livraisons des test, et encore plus, pour :

- Le gain de temps dû à une réduction de la charge de travail des équipes.
- L'augmentation de la productivité.
- La fiabilité et la qualité des tests effectués.
- L'accroissement du niveau de complétude des tests réalisés.

- La démobilisation des équipes sur les tâches répétitives au profit de tâches à plus grande valeur ajoutée (Analyse des résultats, définition des cas de tests, planification, etc.).
- La collaboration entre les différents membres des équipes.

C'est dans ce contexte que s'inscrit notre projet de fin d'études qui vise à construire un ensemble de tests automatisés pour l'application Chosa qui donne l'occasion à chaque parent, là où il se trouve de suivre la journée de son enfant dans l'un des jardins. En même temps, il peut communiquer avec les animatrices grâce à la messagerie instantané.

Notre solution doit :

- faire l'exécution des tests manuels et automatiques
- obtenir les résultats de ces tests (réussite ou échec)
- avoir des rapports décrivant en détails les résultats de ces tests plus facile.

Nous allons donc travailler à automatiser les tests nécessaires en les exécutant par un outil externe, contrôlant cette exécution, rapportant les résultats des tests automatisés et en les associant à des tests nouveaux ou existants dans un outil de gestion des tests tel que Xray dans notre Cas.

## 1.5 Méthodologie de travail

La méthodologie c'est un système de pratiques techniques et procédures utilisées par ceux qui travaillent dans une discipline. Les pratiques Lean, Kanban, Scrum et Six Sigma sont des exemples de méthodologies de gestion de projet que nous pouvons utiliser pour planifier nos tâches et atteindre nos objectifs. Mais différentes méthodologies vont être bénéfiques à différents projets, et les modèles de gestion de projet ne sont pas tous efficaces pour différentes tâches. En effet, notre étude nous conduise vers la méthodologie Scrum pour aboutir à un bon résultat et garantir le déroulement de différentes phases de notre projet.

Scrum est une méthode agile de gestion de projets qui est adaptée aux projets informatiques. En l'utilisant nous pouvons aborder des problèmes complexes et adaptatifs, tout en livrant de manière efficace et créative des produits de la plus grande valeur possible . Avec Scrum, le projet est découpé en parties, chaque partie est appelée "Sprint"[b].

Choisir Scrum comme une méthodologie de pilotage pour notre projet s'est basé sur les atouts de ce dernier. Il se résume comme suit :

- Plus de souplesse et de réactivité.
- La grande capacité d'adaptation au changement grâce à des itérations courtes.

Le plus important c'est que Scrum rassemble les deux cotés théorique et pratique et se rapproche beaucoup de la réalité .

## 1.6 Conclusion

Dans ce chapitre, nous avons présenté notre organisme d'accueil. Puis, nous avons élaboré une étude de l'existant pour dégager les faiblesses de la solution actuelle et exposer les objectifs de notre solution proposée. Enfin, nous avons défini la méthodologie utilisée dans notre projet.

# chapitre 2

## Spécification des besoins

### 2.1 Introduction

L'analyse et la spécification des besoins est une étape fondamentale. Elle permet de définir le cadre général de notre projet et de déterminer ses orientations globales[c]. Dans ce chapitre, nous allons identifier les besoins fonctionnels et non fonctionnels. Ensuite, nous allons présenter nos choix techniques pour l'automatisation des tests et l'architecture adoptée.

Après, nous allons présenter les principaux acteurs et les exigences correspondantes à leurs activités. Et pour décrire l'aspect fonctionnel de notre projet, nous allons exposer le diagramme de cas d'utilisations général et le diagramme de classe global. Finalement, nous détaillons le Backlog Produit de notre travail.

### 2.2 Identification des besoins

Dans cette section, nous allons présenter les besoins fonctionnels et non fonctionnels de notre projet.

#### 2.2.1 Besoins fonctionnels

Les besoins fonctionnels sont des exigences qui spécifient ce que le système doit être capable de faire, quelles que soient les contraintes physiques. Ce sont des besoins du point de vue de l'utilisateur[d].

Pour une première partie, nous présentons l'ensemble des besoins fonctionnels auxquels doit répondre notre système :

- S'authentifier,
- Etablir un cahier de test,
- Implémenter et exécuter les tests manuels,
- Implémenter et exécuter les tests automatiques,
- Déetecter les bugs.

#### 2.2.2 Besoins non fonctionnels

Les besoins non fonctionnels sont les besoins qui caractérise une applicaion.Ils présentent les règles à respecter pour assurer la bonne qualité et le bon fonctionnement de

notre futur système[e].

Dans cette seconde partie, nous définissons les besoins non fonctionnels :

- L'ergonomie : notre système doit offrir une interface conviviale, facile à utiliser et cohérente ce qui permet aux utilisateurs de se familiariser rapidement avec le contenu.
- La disponibilité : notre système doit être accessible indépendamment des flux d'informations entrés.
- La sécurité : les tâches demandées nécessitent une authentification pour y accéder au système.
- La performance : le temps de réponse doit être assez raisonnable.

## 2.3 Technologies et architecture utilisées

Dans cette section, nous allons présenter nos choix techniques pour l'automatisation des tests et nous allons exposer l'architecture adoptée.

### 2.3.1 Technologies utilisées pour l'automatisation des tests

Dans cette partie, nous présentons nos choix techniques des outils pour l'automatisation des tests. Ils sont composés de deux catégories :

#### - Les outils de tests :

Ils permettent d'exécuter et de rejouer des tests fonctionnels. Ils disposent de fonctionnalités de reconnaissance d'objets d'interface utilisateur permettant de détecter et de mettre à jour automatiquement ces objets. Le principal intérêt est de produire des résultats cohérents et de réduire l'effort de maintenabilité des scripts de test[f].

#### Selenium Webdriver

Dans notre projet, nous avons choisi l'outil de test Selenium Webdriver, dont le logo est présenté dans la figure 2.3. C'est un framework web qui vous permet d'exécuter des tests multi-navigateurs. Cet outil est utilisé pour automatiser les tests d'applications Web pour vérifier qu'il fonctionnent correctement[g].



FIGURE 2.3 – Logo de Selenium Webdriver

Selenium Webdriver permet de choisir un langage de programmation de votre choix pour

créer des scripts de test : JavaScript, Python, Ruby..

### Python

Notre étude, nous a conduit à utiliser Python (logo figure 2.4) car il est le langage de programmation open source le plus utilisée par les informaticiens. Elle permet d'analyser les données et gérer l'infrastructure d'un projet[h].

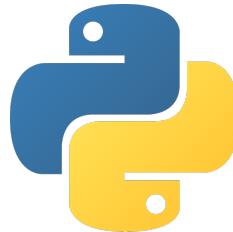


FIGURE 2.4 – Logo de Python

### Jenkins

Jenkins (logo figure 2.5) est un outil d'intégration développé à l'aide du langage de programmation Java. Il permet de tester et de rapporter les changements effectués sur une large base de code en temps réel[i].



FIGURE 2.5 – Logo de Jenkins

### - Les outils de gestion de test :

Ils permettent d'évaluer les fonctionnalités d'une application web et optimiser la qualité du produit pour que l'expérience utilisateur soit parfaite[j].

#### Xray

Xray dont le logo est illustré dans la figure 2.6 est une application permettant d'intégrer à Jira une dimension de tests. Il permet ainsi de gérer ses tests directement depuis Jira[k].



FIGURE 2.6 – Logo de Xray

#### Gitlab

Gitlab dont le logo est illustré dans la figure 2.7 est un système d'intégration continue destiné aux équipes qui permet d'automatiser les builds, les tests, les livraisons et les

déploiements des applications[1].



FIGURE 2.7 – Logo de Gitlab

### 2.3.2 Architecture utilisée pour l'automatisation des tests

CI/CD [m](Continuous Integration/ Continuous Delivery) est composé des pratiques d'intégration continue et de déploiement continu. c'est une méthode de travail qui permet d'améliorer la répartition des applications en intégrant l'automatisation au niveau des étapes de développement.

- Le « CI » de CI/CD désigne l'« intégration continue », qui consiste, pour les développeurs, à apporter continuellement des modifications au code de leur application, à les tester, puis à les fusionner dans un référentiel partagé.
- Le « CD » de CI/CD désigne la « déploiement continu », qui favorise la transmission automatique des changements établis au niveau du référentiel vers l'environnement de production. Ce processus technique permet de soulager les équipes d'exploitation surchargées par les tâches manuelles qui ralentissent la distribution des applications.

La figure 2.8 décrit l'architecture CI/CD[n].

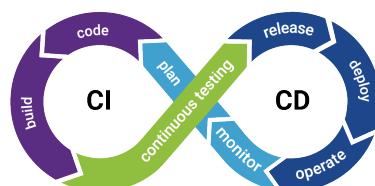


FIGURE 2.8 – Architecture CI/CD

#### Avantages de cette architecture

L'architecture CI/CD offre plusieurs avantages lors de l'introduction de l'automatisation au niveau des étapes de développement des applications ce qui renforce notre choix.

En effet, elle permet d'accélérer la production d'un code et la résolution des bugs. Ce qui contribue à une livraison, à une mise en production et à une commercialisation plus rapides des applications.

La méthode CI/CD permet aussi d'intégrer plus facilement de nouveaux développeurs à un projet existant. De plus, elle favorise une amélioration continue et un environnement stable au niveau de la production vu que le code est toujours testé avant déploiement.

## 2.4 Identification et structuration des cas d'utilisation

Nous présentons, dans cette section, les acteurs et les différents cas d'utilisations qui offrent des fonctionnalités spécifiques pour chaque utilisateur.

### 2.4.1 Identification des acteurs

Un acteur est une personne, un matériel ou un logiciel qui interagit avec le système dans le but de réaliser une ou plusieurs fonctions concernant les cas d'utilisation. Les acteurs en interactions avec notre plateforme sont spécifiés dans le tableau 2.1.

TABLE 2.1 – Identification des acteurs

| Acteur         | Rôle  |
|----------------|---|
| Testeur        | C'est le concepteur et l'exécuteur des cas de tests.<br>Il génère les plans, fait la configuration entre les scripts de tests et le code source de produit, exécute les tests développés et consulte les rapports.<br>Il envoie au développeur les rapports de test contenant les anomalies et les défauts du produit (bugs) pour pouvoir les résoudre. |
| Administrateur | C'est le responsable de gestion des comptes utilisateurs et des projets. Il gère les utilisateurs en créant, modifiant ou supprimant un utilisateur. En plus, il est le responsable de la gestion des projets en affectant à chaque utilisateur un projet.  |
| Développeur    | C'est le responsable de développement de produit à tester. Il consulte les scénarios et les rapports de tests pour avoir une idée sur les anomalies et les défauts du produit (bugs) pour les résoudre.   |

## 2.4.2 Diagramme de cas d'utilisation globale

Les cas d'utilisation permettent de décrire sous forme d'actions et de réactions le système du point de vue utilisateur.

Les diagrammes de cas d'utilisation modélisent à QUOI sert le système, en organisant les interactions possibles avec les acteurs. Pour cela, nous décrivons les principales fonctionnalités de notre projet en spécifiant les actions que chaque utilisateur peut effectuer[o].

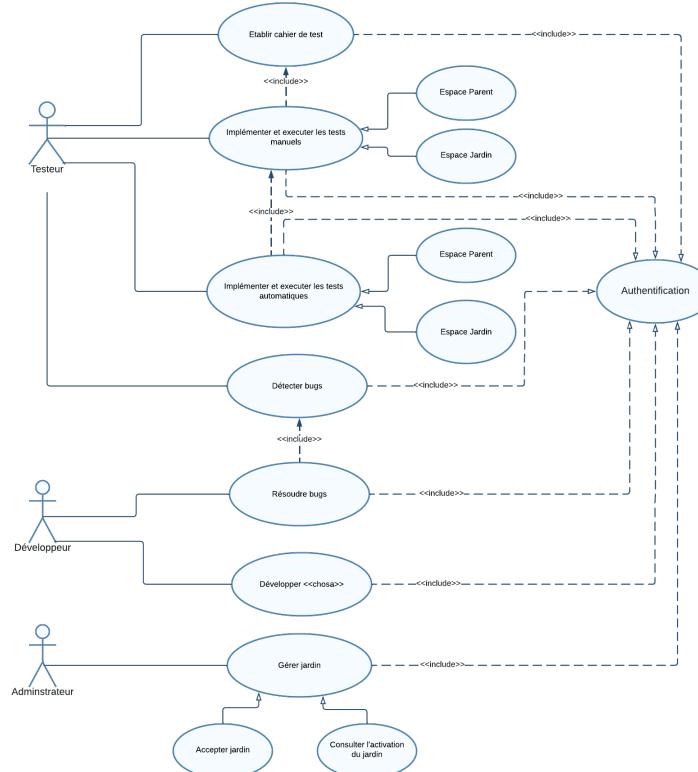


FIGURE 2.9 – Diagramme de cas d'utilisation globale

La figure 2.9 représente le diagramme de cas d'utilisation général qui illustre qu'après une authentification réalisée, le testeur peut établir un cahier de test qui va le guider pour implémenter les tests automatiques après qu'il exécute les tests manuels. Et finalement, il peut détecter les bugs qui vont être résolus par le développeur.

L'administrateur peut gérer les jardins d'enfant spécifique à l'application «chosa» : soit en acceptant un nouveau jardin d'enfant, soit en consultant l'activation d'un jardin d'enfant déjà existant.

## 2.4.3 Backlog de Produit

Le backlog produit est constitué d'une liste de tâches à réalisées. Elles sont classées par priorité ce qui permet de définir l'ordre de la réalisation[p].

Le tableau 2.2 centralise la liste des exigences attendues.

TABLE 2.2 – Backlog de produit

| Back log de Produit   | Priorité | Estimation | Numéro de Release |
|---|----------|------------|-------------------|
| Authentification  | 1        | Faible     | Release1          |
| Implémenter et exécuter les tests manuels et automatique de l'espace parent | 2        | moyen      | Release2          |
| Implémenter et exécuter les tests manuels et automatique de l'espace Jardin | 2        | moyen      | Release3          |

#### 2.4.4 Diagramme de classe globale

Un diagramme de classes fournit une vue globale d'un système en présentant ses classes, interfaces et collaborations, et les relations entre elles[q].

Dans la figure2.10, nous présentons le diagramme de classes global et les différentes classes et les méthodes que nous avons utilisées.

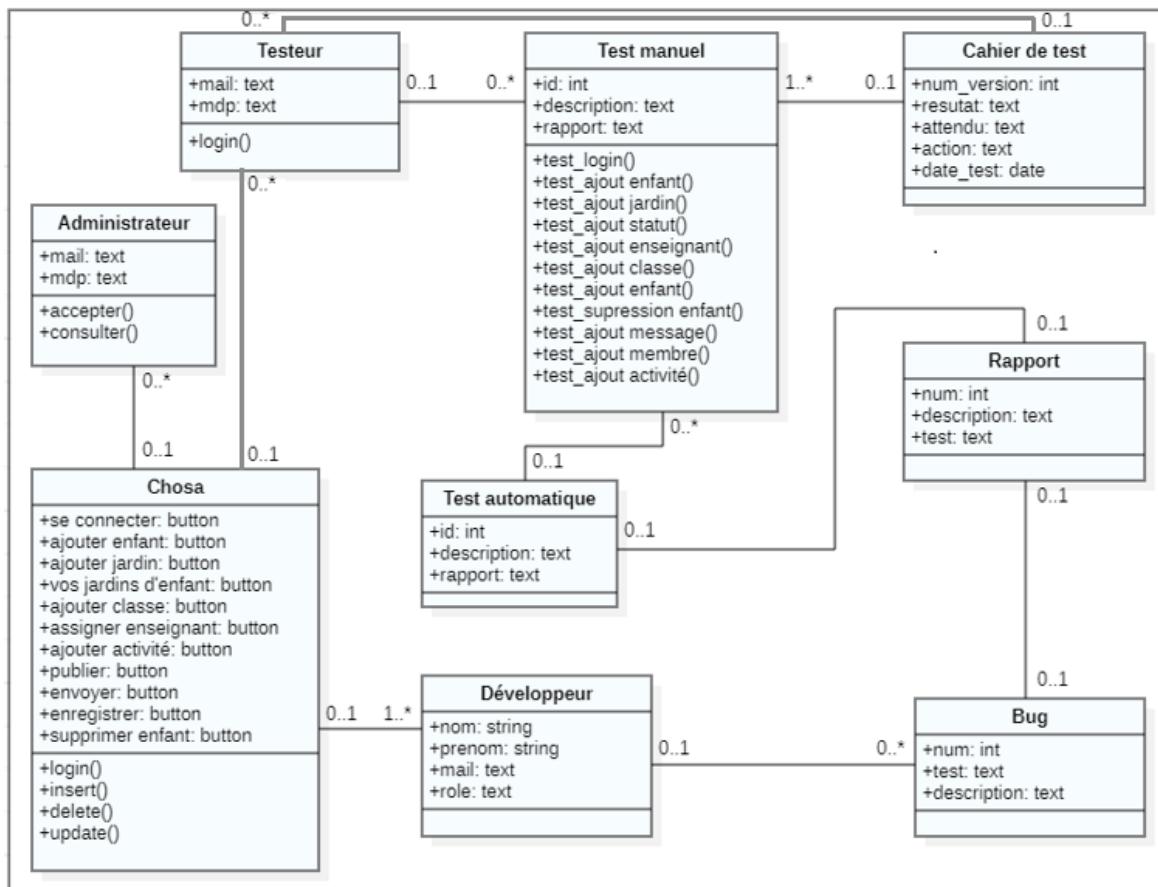


FIGURE 2.10 – Diagramme de classe global

## 2.5 Conclusion

Dans ce chapitre, nous avons présenté les besoins fonctionnels et non fonctionnels. Puis, nous avons décrit les technologies et l'architecture utilisées. La dernière partie est consacrée pour l'identification et structuration des cas d'utilisation. Dans le chapitre suivant, nous allons élaborer le premier release tout en exposant la conception et la réalisation.

# chapitre 3

## Release 1 : Authentification

### 3.1 Introduction

Dans ce chapitre, nous allons présenter notre premier release «Authentification». Ce chapitre couvre les diagrammes de cas d'utilisation détaillés, les diagrammes de séquence et les interfaces de développement de ce release.

### 3.2 Identification de backlog de release1

Le tableau 5.9 décrit les différents éléments spécifiques au release 1.

TABLE 3.3 – Identification de Backlog de Release 1

| Numéro tache | Backlog de Produit                                | Priorité | Estimation |
|--------------|---|----------|------------|
| 1            | Mise en place de l'environnement                  | 1        | Faible     |
| 1.1          | Familiarisation de Python avec Selenium Webdriver | 1        | Faible     |
| 1.2          | Structure de framework                            | 1        | Faible     |
| 2            | Test authentification                             | 1        | Fort       |
| 2.1          | Test manuel de l'authentification                 | 1        | Fort       |
| 2.2          | Test automatique de l'authentification            | 1        | Fort       |

#### 3.2.1 Préparation de l'environnement

L'environnement de travail désigne certaines conditions que nous devons respecter afin de travailler en mode sans échec sans affecter les anciens fichiers et l'architecture actuelle [r].

Pour réaliser notre environnement, nous avons suivi les étapes suivantes :

### Etape 1 : Installation et familiarisation de Python avec Selenium Webdriver

Dans cette étape, nous avons écrit nos tests fonctionnels à l'aide du Sélénum Web Driver. Ensuite, nous avons envoyé une demande au serveur Sélénum. Après, nous avons choisi un navigateur où nos scénarios de test vont être exécutés tels que Google Chrome, Internet Explorer ou Mozilla Firefox. Dans notre cas, nous avons choisi Google chrome. Mais, l'exécution de nos scénarios sur le navigateur ne peut pas être réalisé qu'avec une liaison entre Python et Selenium. Cette dernière, fournit une API pratique qui nous permet d'accéder aux Sélénum Web Driver.

D'où une deuxième étape doit être réalisée, c'est la configuration de Sélénum à l'aide de Python. Tout d'abord, nous avons téléchargé et installé Python sur Windows. Ensuite, nous avons installé les bibliothèques Sélénum en Python à l'aide de la commande :

« `pip install -U selenium` ».

Puis, nous avons téléchargé et installé Pycharm [s]et nous avons créé notre nouveau projet. la figure 3.11 expose la liaison de Python avec Selenium[t]

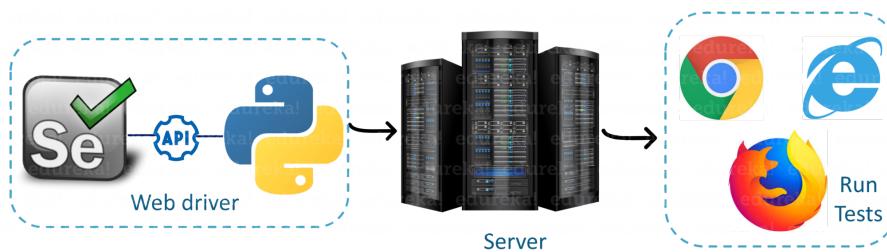


FIGURE 3.11 – Structure de familiarisation de Python avec Selenium

### Etape 2 : Structure de Framework

Maintenant, notre environnement est mis en place ce qui nous permet de présenter la structure de notre Framework comme illustrée dans la figure 3.12.

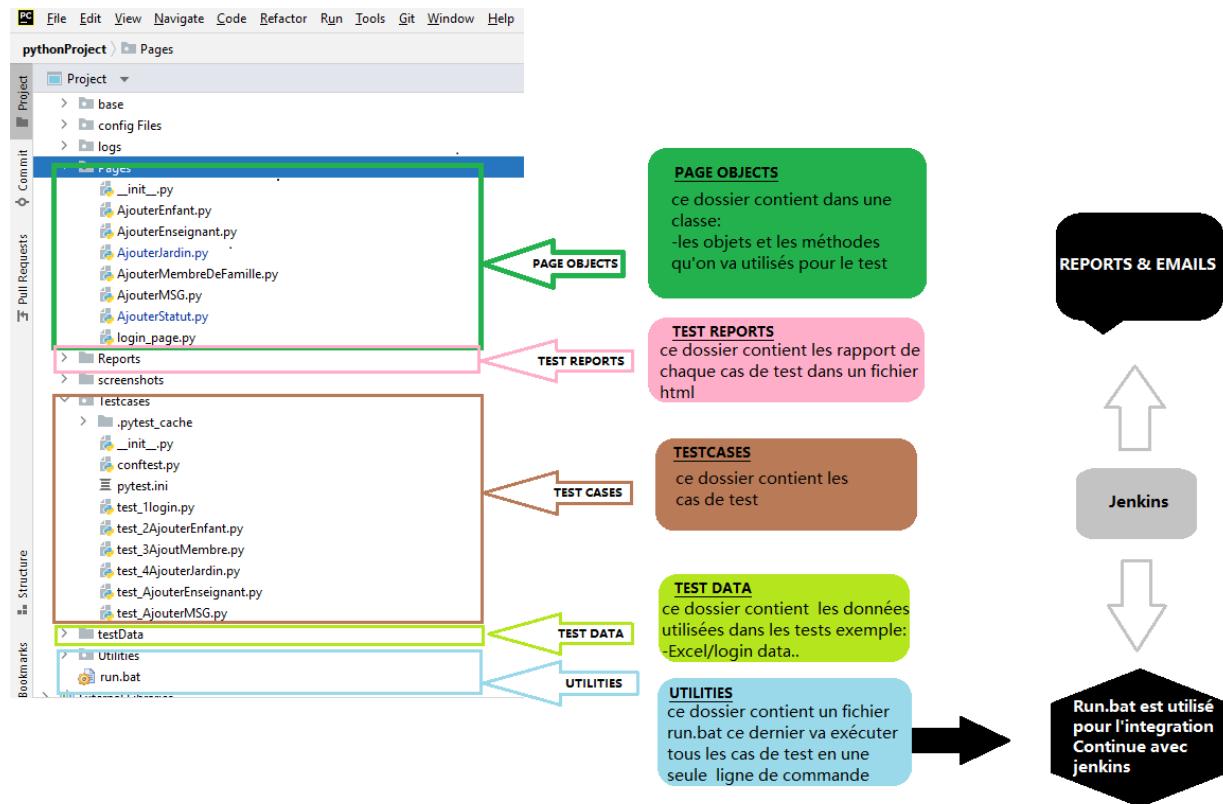


FIGURE 3.12 – Structure de Framework

### 3.2.2 Diagrammes de cas d'utilisation détaillés de release1

Cette partie présente les diagrammes de cas d'utilisation de notre premier release Authentification.

L'authentification est une condition préalable nécessaire à tous les autres processus décrits dans les cas d'utilisation. Elle permet au système de vérifier l'identité d'une entité (personne, ordinateur...). Le but de cette procédure étant d'autoriser la personne à accéder à certaines ressources sécurisées[u].

La figure 3.13 présente le diagramme de cas d'utilisation spécifique au «Test manuel de l'authentification».

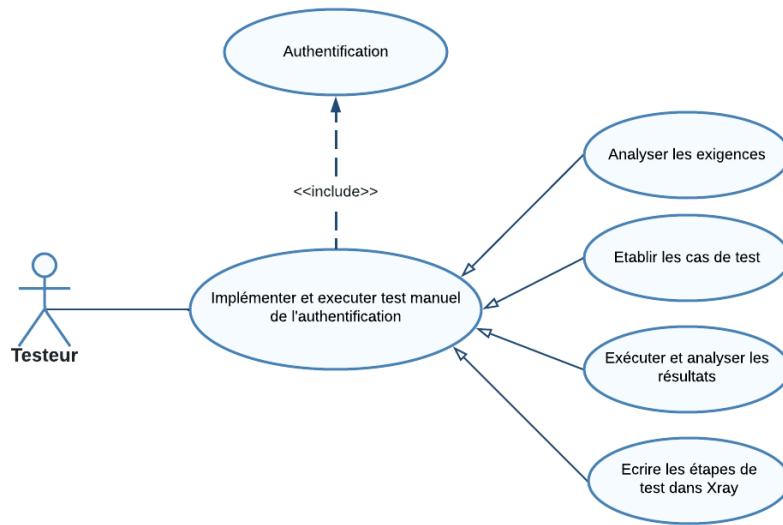


FIGURE 3.13 – Diagramme de cas d'utilisation «Test manuel de l'authentification»

La figure 3.14 présente le diagramme de cas d'utilisation spécifique au «Test automatique de l'authentification».

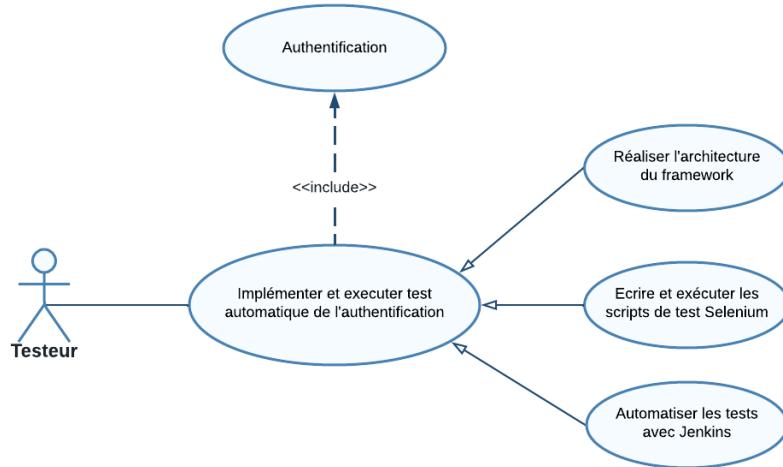


FIGURE 3.14 – Diagramme de cas d'utilisation «Test automatique de l'authentification»

### 3.2.3 Description détaillée du Release1

le tableau4.7 décrit le scénario suivi par le testeur pour réaliser «Test manuel de l'authentification»

TABLE 3.4 – Description du cas d'utilisation «Test manuel de l'authentification»

|                                   |  |
|-----------------------------------|--|
| Cas d'utilisation                 | Test manuel de l'authentification  |
| Acteur                            | Testeur  |
| Pré-Condition                     | Connexion à Chosa et analyse de l'interface «l'authentification»   |
| Post condition                    | Test manuel de l'authentification crée et exécuté  |
| Description du scénario principal | <p>l'interface de l'authentification s'affiche et le testeur va tester manuellement :</p> <ul style="list-style-type: none"> <li>-le fonctionnement du bouton «Se connecter avec Facebook»</li> <li>-les zones de text de l'adresse et mot de passe</li> <li>-fonctionnement du bouton «Log in».</li> </ul> <p>Et finalement, apres avoir configuré l'application Xray il va exécuter le test authentification a partir de description définie sur le dashboard jira</p> |
| Exception                         | Echec de connexion à Chosa   |

le tableau4.8 décrit le scénario suivi par le testeur pour réaliser «Test automatique de l'authentification»

TABLE 3.5 – Description du cas d'utilisation «Test automatique de l'authentification»

|                                   |  |
|-----------------------------------|--|
| Cas d'utilisation                 | Test automatique de l'authentification   |
| Acteur                            | Testeur  |
| Pré-Condition                     | Test manuel de l'authentification établi   |
| Post condition                    | Test automatique de l'authentification crée et exécuté   |
| Description du scénario principal | <p>Pour automatiser test de l'authentification,le testeur :</p> <ul style="list-style-type: none"> <li>-réalise structure du framework</li> <li>-réaliser script de test Selenium et exécute les tests</li> <li>-Déploie les tests avec Jenkins</li> </ul> |

### 3.2.4 Diagramme de séquence

Un diagramme de séquence est un diagramme qui serve à illustrer le déroulement des cas d'utilisations en fonction du temps. En effet, Il permet de représenter la succession chronologique des opérations réalisées par un acteur et qui font passer d'un objet à un autre pour représenter les scénarios[v].

Le diagramme ci-dessous 3.15 décrit le diagramme de séquence pour la fonctionnalité «Test manuel de l'authentification»

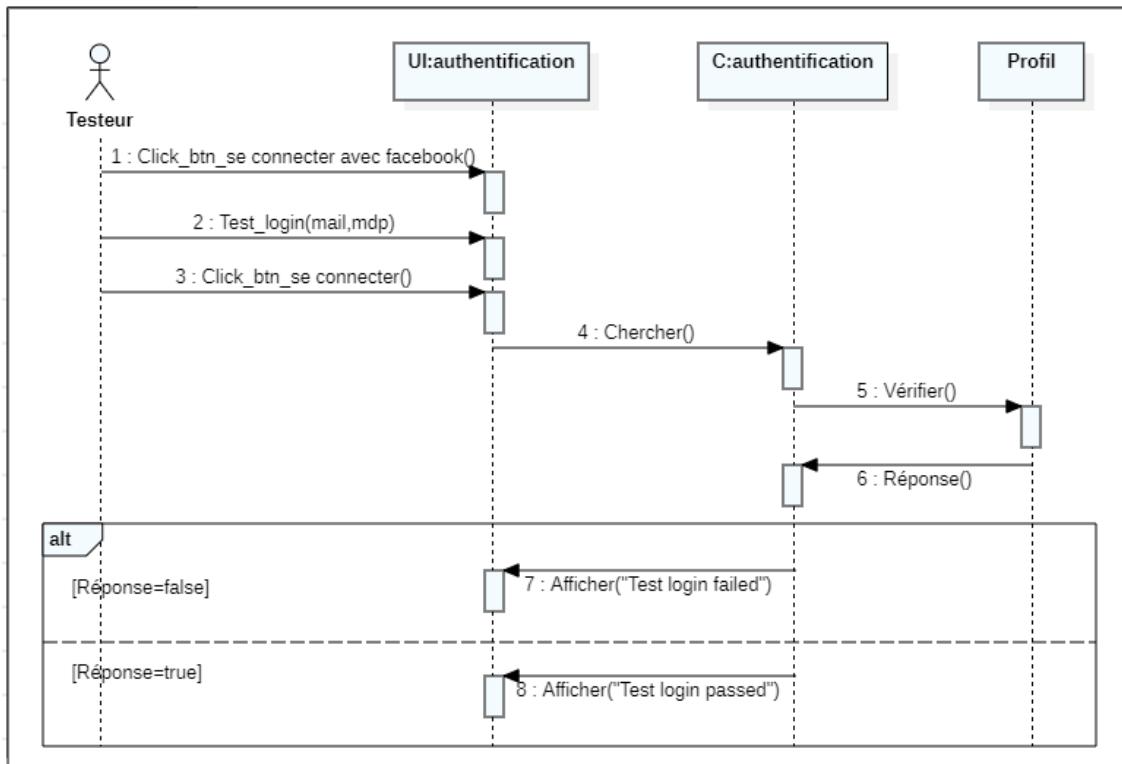


FIGURE 3.15 – Diagramme de séquence détaillé «Authentification»

## 3.3 Réalisation

Dans cette partie, nous présentons les différents interfaces relatives au Release1.

### 3.3.1 Test manuel de l'authentification

Cette partie présente les interfaces à tester spécifiques au «test manuel de l'authentification» .

La figure 3.16 illustre la page d'authentification à l'application "Chosa". En cliquant sur l'authentification via Facebook, l'interface de la figure 3.17 s'affiche.

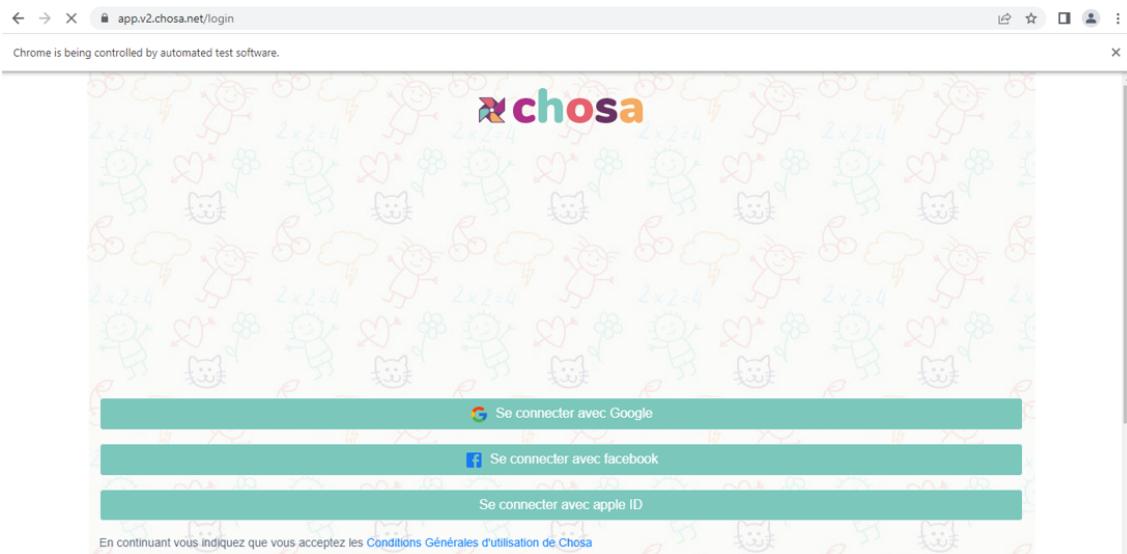


FIGURE 3.16 – Interface Authentification

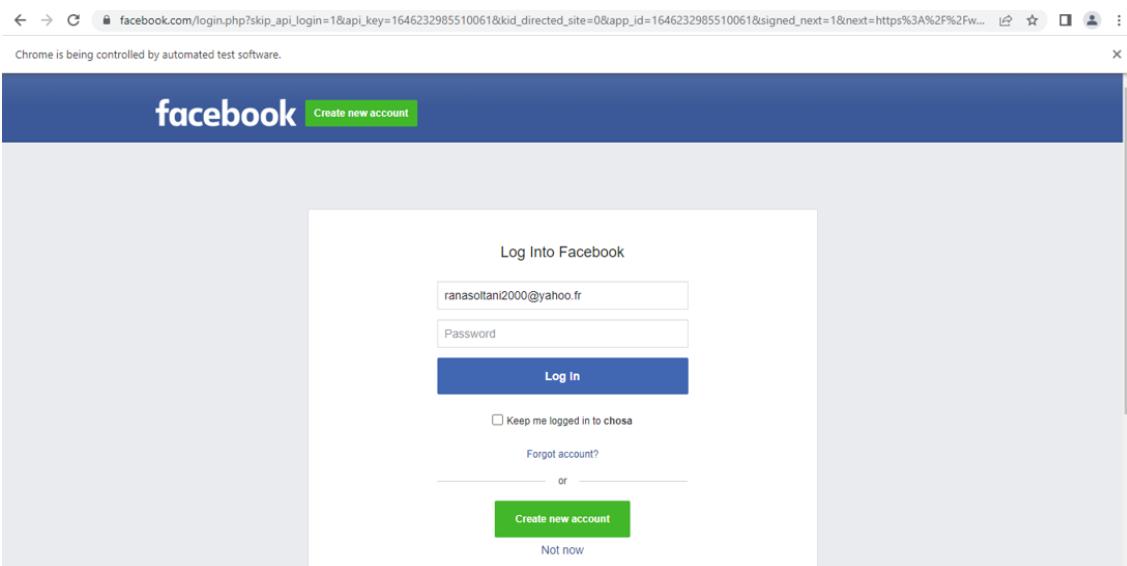


FIGURE 3.17 – Interface Authentification via Facebook

Après avoir écrire les étapes de test sur Xray, Le résultat de test manuel de l'authentification sera affiché sur le Dashboard comme le montre la figure 3.18

The screenshot shows the Jira dashboard for a manual test named 'TC001: Test\_Login'. The 'Test details' section contains the following information:

- Test Issue Link:** [Login test via Facebook](#)
- Preconditions:** CHOS-3: chosa login page should be up
- Steps:**
  - Action: go to, Data: <https://app.chosa.net/login>, Expected Result: <https://app.chosa.net/garden/account>, Step Status: PASSED
  - Action: click on, Data: btn\_id\_connecter\_avec\_facebook, Expected Result: <https://app.chosa.net/garden/account>, Step Status: PASSED

FIGURE 3.18 – Résultat sur Jira dashboard de test manuel authentification

### 3.3.2 Test automatique de l'authentification

Cette partie présente les interfaces spécifiques au «Test automatique de l'authentification».

Ce fichier nous montre les étapes de test automatique et son résultat . Dans notre cas 3.19le test est passé avec succès a la page d'accueil Chosa.

```

2022-05-11 01:09:50,283 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-11 01:09:56,284 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-11 01:09:57,294 - WDM - INFO - Driver [C:\Users\nada\.wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found
2022-05-11 01:09:59,666 - root - INFO - ***** TEST ADD GARDEN *****
2022-05-11 01:10:27,166 - root - INFO - ***** START ADDING GARDEN *****
2022-05-10 10:47:42,216 - WDM - INFO -
2022-05-10 10:47:42,232 - WDM - INFO - ===== WebDriver manager =====
2022-05-10 10:47:46,887 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-10 10:47:46,903 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-10 10:47:48,200 - WDM - INFO - Trying to download new driver from https://chromedriver.storage.googleapis.com/101.0.4951.41
2022-05-10 10:48:17,250 - WDM - INFO - Driver has been saved in cache [C:\Users\nada\.wdm\drivers\chromedriver\win32\101.0.4951.41]
2022-05-10 10:48:29,166 - root - INFO - ***** LOGIN TEST *****
2022-05-10 10:48:29,166 - root - INFO - ***** VERIFYING LOGIN TEST *****
2022-05-10 10:49:15,641 - root - INFO - ***** LOGIN TEST PASSED *****

test_login.py::Test_001_login::test_login
C:/Users/nada/PycharmProjects/pythonProject/Pages\login_page.py:28: DeprecationWarning: find_element_by_* commands are deprecated. Please use find_element() instead
    self.driver.find_element_by_xpath(self.button_nextp_id).click()

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
----- generated html file: file:///C:/Users/nada/PycharmProjects/pythonProject/Reports/report.html -----
=====
1 passed, 5 warnings in 97.18s (0:01:37)
=====
```

FIGURE 3.19 – Description de fichier "automation.log"

Après avoir établir les tests automatiques de l'authentification, un rapport de test sera généré automatiquement comme l'indique la figure 3.20.

Report generated on 15-May-2022 at 10:49:18 by [cytest-html](#) v3.1.1

**Environment**

|              |   |
|--------------|---|
| JAVA_HOME    | C:\Program Files\Java\jdk-11.0.15   |
| PROJECT_NAME | CHOSA   |
| Packages     | {'pluggy': '1.0.0', 'py': '1.11.0', 'pytest': '7.1.1'}                    |
| Platform     | Windows-10-10.0.19041-SP0   |
| Plugin       | {'forked': '1.4.0', 'html': '3.1.1', 'metadata': '2.0.1', 'viz': '2.5.0'} |
| Python       | 3.7.0   |
| TESTERS      | NADA & RANA   |

**Summary**  
1 tests ran in 07.10 seconds.  
(Un)check the boxes to filter the results.  
 1 passes  0 skipped  0 failed  0 errors  0 expected failures  0 unexpected passes

**Results**  
[Show all details](#) / [Hide all details](#)

| Result                | Test                                      | Duration | Links |
|-----------------------|---|----------|-------|
| Passed (hide details) | test_login.py::Test_001_Login::test_login | 06.00    |       |

```
-----Captured stdout setup-----
warning: WebDriver manager missing
Current chrome version is 101.0.4951
Get LATEST chromedriver version for 101.0.4951 google-chrome
Trying to download new driver from https://chromedriver.storage.googleapis.com/101.0.4951.41/chromedriver_win32.zip
Driver has been saved in cache (C:\Users\hadal\wdm\drivers\chromedriver\win32\101.0.4951.41)
-----Captured log setup-----
INFO Wdmlogger.py:27 warning WebDriver manager missing
INFO Wdmlogger.py:27 Current google-chrome version is 101.0.4951
INFO Wdmlogger.py:27 Get LATEST chromedriver version for 101.0.4951 google-chrome
INFO Wdmlogger.py:27 Trying to download new driver from https://chromedriver.storage.googleapis.com/101.0.4951.41/chromedriver_win32.zip
INFO Wdmlogger.py:27 Driver has been saved in cache (C:\Users\hadal\wdm\drivers\chromedriver\win32\101.0.4951.41)
-----
INFO root@test_llogin:py:15 ***** LOGIN TEST *****
INFO root@test_llogin:py:16 ***** VERIFYING LOGIN TEST *****
```

FIGURE 3.20 – Interface de rapport html de test authentification

## 3.4 Conclusion

Dans ce chapitre, nous avons détaillé la conception et la réalisation du premier Release : Authentification. Dans le chapitre d'après, nous allons établir le même démarche pour la mise en place du deuxième Release : Les tests manuels et automatiques de l'espace parent.

# **chapitre 4**

## **Release 2 : Les tests manuels et automatiques de l'espace parent**

### **4.1 Introduction**

Dans ce chapitre, nous allons présenter notre deuxième release «Les tests manuels et automatiques de l'espace parent». L'étude de ce sprint couvre les diagrammes de cas d'utilisation détaillés, les diagrammes de séquence et les interfaces de réalisation de ce sprint.

### **4.2 Identification de backlog de Release 2**

Le tableau 5.9 décrit les différents éléments spécifiques au Release 2.

TABLE 4.6 – Identification de Backlog de release 2

| Numéro tâche | Backlog de Produit   | Priorité | Estimation |
|--------------|--|----------|------------|
| 1            | Implémenter et exécuter les tests manuels de l'espace parent               | 1        | Faible     |
| 1.1          | Test manuel de l'ajout d'un enfant dans l'espace parent                    | 1        | Faible     |
| 1.2          | Test manuel de l'ajout d'un jardin dans l'espace parent                    | 1        | Faible     |
| 1.3          | Test manuel de l'ajout d'un statut dans l'espace parent                    | 1        | Faible     |
| 1.4          | Test manuel de l'ajout d'un membre de la famille dans l'espace parent      | 1        | Faible     |
| 1.5          | Test manuel de l'ajout d'un message dans l'espace parent                   | 1        | Faible     |
| 2            | Implémenter et exécuter les tests automatiques de l'espace parent          | 2        | Fort       |
| 2.1          | Test automatique de l'ajout d'un enfant dans l'espace parent               | 2        | Fort       |
| 2.2          | Test automatique de l'ajout d'un jardin dans l'espace parent               | 2        | Fort       |
| 2.3          | Test automatique de l'ajout d'un statut dans l'espace parent               | 2        | Fort       |
| 2.4          | Test automatique de l'ajout d'un membre de la famille dans l'espace parent | 2        | Fort       |
| 2.5          | Test automatique de l'ajout d'un message dans l'espace parent              | 2        | Fort       |

#### 4.2.1 Diagrammes de cas d'utilisation détaillés de Release 2

Cette partie présente les diagrammes de cas d'utilisation de notre premier release «Les tests manuels et automatiques de l'espace parent».

En premier lieu, nous allons présenter le diagramme de cas d'utilisation globale de ce release «Les tests manuels et automatiques de l'espace parent». En second lieu, nous allons décrire le diagramme de cas d'utilisation détaillé concernant «Les tests manuels de l'espace parent». En dernier lieu, nous allons présenter le diagramme de cas d'utilisation détaillé concernant «Les tests automatiques de l'espace parent». La figure 4.21 schématisé le diagramme de cas d'utilisation qui décrit d'une manière générale notre sprint1.

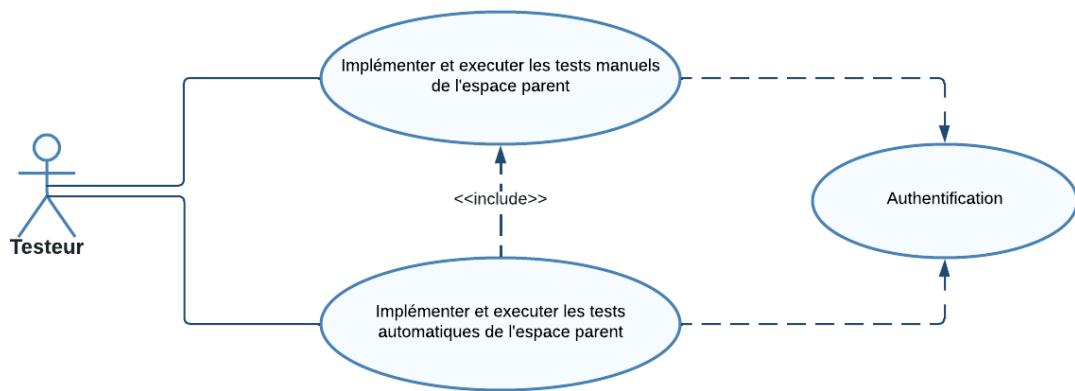


FIGURE 4.21 – Diagramme de cas d'utilisation «tests manuels et automatiques de l'espace parent»

Pour expliquer plus en détail, nous présentons dans la figure 4.22 le diagramme de cas d'utilisation des «tests manuels réalisés au niveau de l'espace parent» .

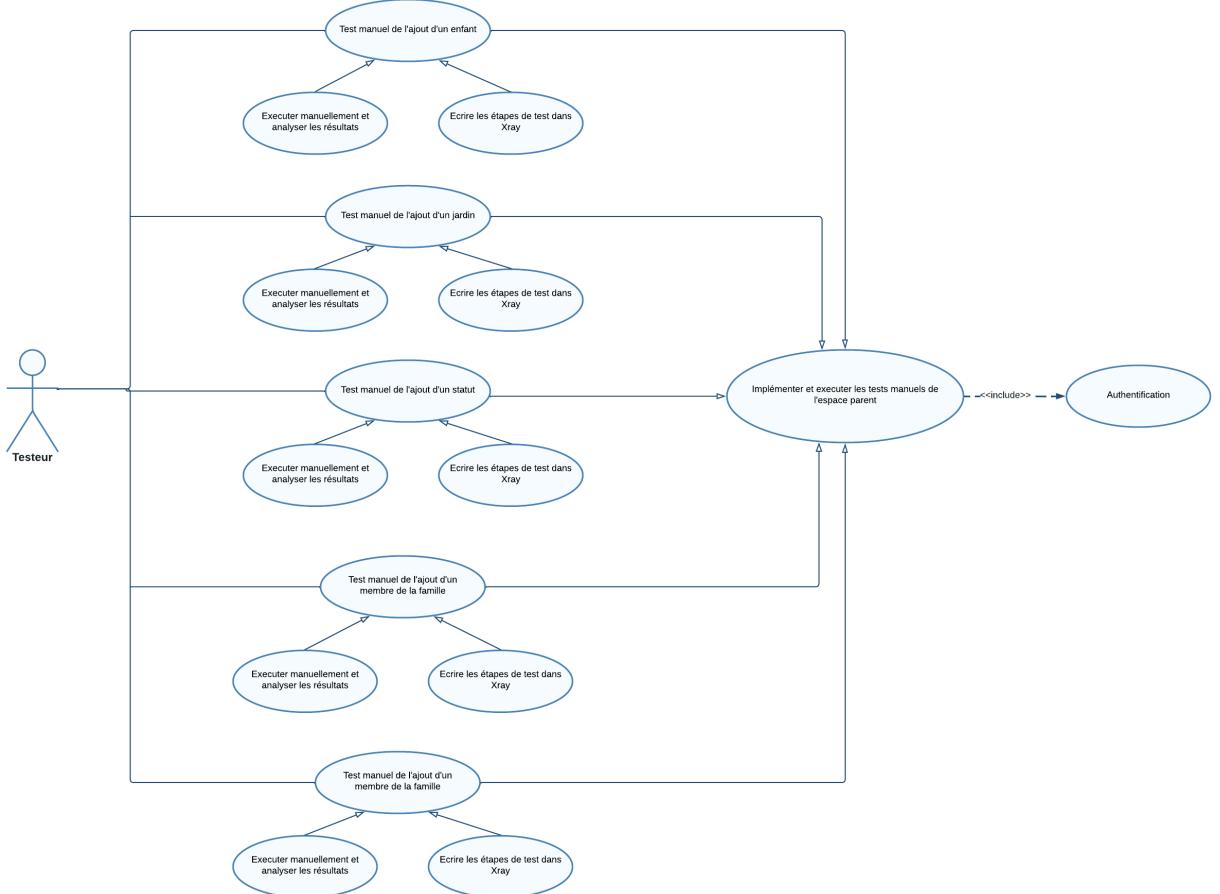


FIGURE 4.22 – Diagramme de cas d'utilisation «Implémenter et exécuter les tests manuels de l'espace parent»

Après la figure 4.23 illustre le diagramme de cas d'utilisation des «tests automatique réalisés au niveau de l'espace parent» par le testeur.

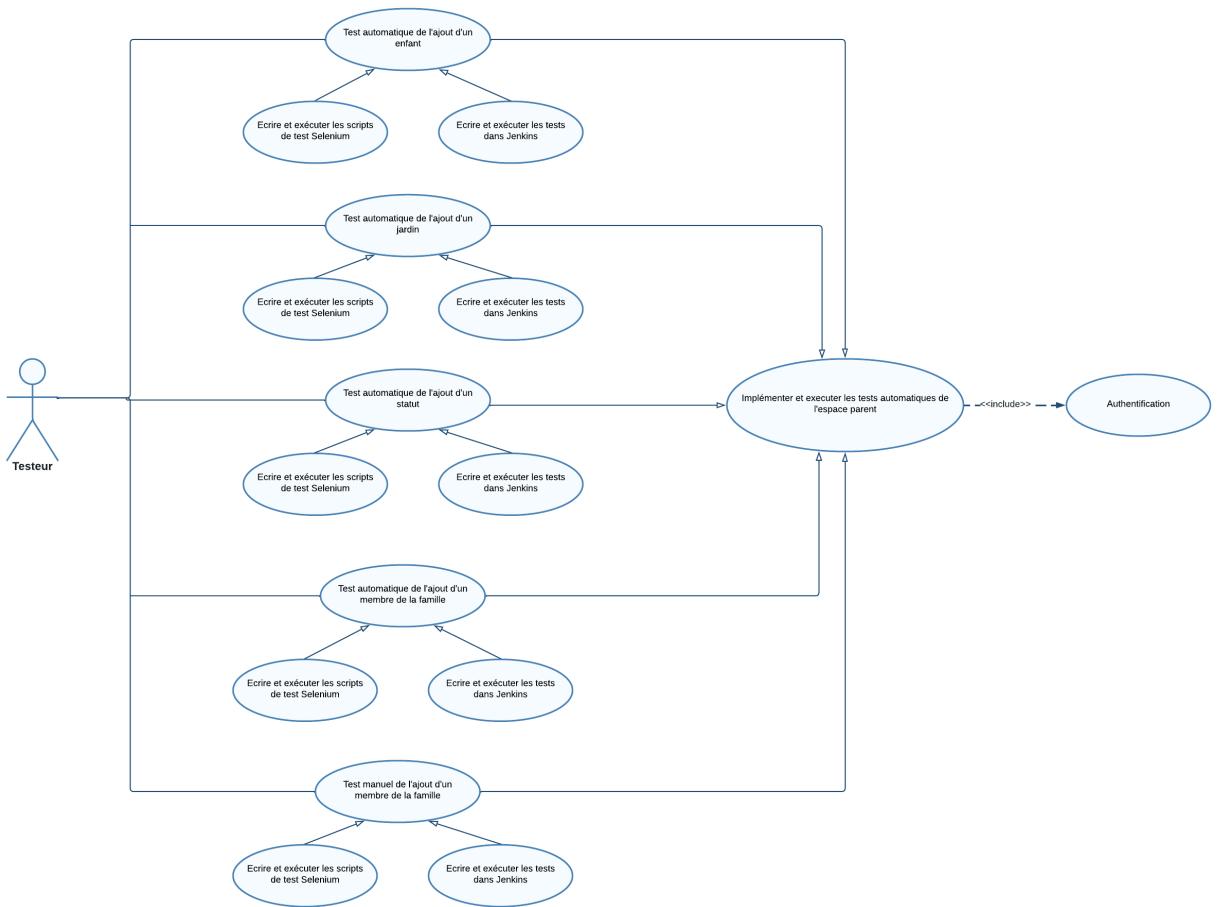


FIGURE 4.23 – Diagramme de cas d'utilisation «Implémenter et exécuter les tests automatiques de l'espace parent»

### 4.2.2 Description détaillée du Release 2

Le tableau 4.7 décrit le scénario suivi par le testeur pour réaliser «Tests manuels de l'espace parent»

TABLE 4.7 – Description du cas d'utilisation «Tests manuels de l'espace parent»

|                                   |   |
|-----------------------------------|---|
| Cas d'utilisation                 | Tests manuel de l'espace parent   |
| Acteur                            | Testeur   |
| Pré-Condition                     | <ul style="list-style-type: none"> <li>- Connexion à Chosa</li> <li>- Connexion à Xray</li> </ul>   |
| Post condition                    | Tests manuels de l'espace parent créés et exécutés  |
| Description du scénario principal | <p>Pour exécuter manuellement les tests de l'espace parent, le testeur suit les étapes suivantes :</p> <ul style="list-style-type: none"> <li>- exécution manuelle des tests</li> <li>- Analyse des résultats générés</li> <li>- Implémentation de la description des tests sur Xray</li> </ul> |

Le tableau 4.8 décrit le scénario suivi par le testeur pour réaliser «Test automatique de l'espace parent»

TABLE 4.8 – Description du cas d'utilisation «Test automatique de l'espace parent»

|                                   |  |
|-----------------------------------|--|
| Cas d'utilisation                 | Test automatique de l'espace parent  |
| Acteur                            | Testeur  |
| Pré-Condition                     | Tests manuels de l'espace parent établis   |
| Post condition                    | Tests automatiques de l'espace parent créés et exécutés  |
| Description du scénario principal | <p>Pour automatiser les tests de l'espace parent, le testeur suit les étapes suivantes :</p> <ul style="list-style-type: none"> <li>- configuration de Jenkins</li> <li>- implémentation et exécution des tests de l'espace parent sur le job Jenkins déjà créé</li> <li>- vérification sur la console Jenkins si les tests sont exécutés avec succès ou non.</li> </ul> |

### 4.2.3 Diagrammes de séquence du Release 2

Dans cette partie, nous allons présenter les diagrammes de séquence détaillés des fonctionnalités illustrées dans Release 2 relatives aux tests manuels .

Le diagramme 4.24 décrit le diagramme de séquence de la fonctionnalité «tester l'ajout d'un enfant dans l'espace parent».

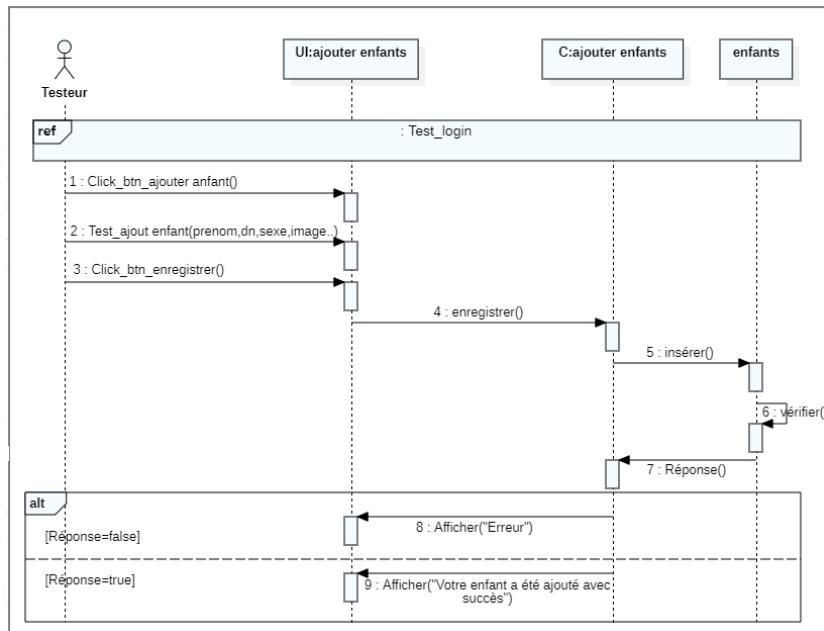


FIGURE 4.24 – Diagramme de séquence détaillé «tester l'ajout d'un enfant dans l'espace parent»

Le diagramme 4.25 décrit le diagramme de séquence de la fonctionnalité «tester l'ajout d'un jardin dans l'espace parent».

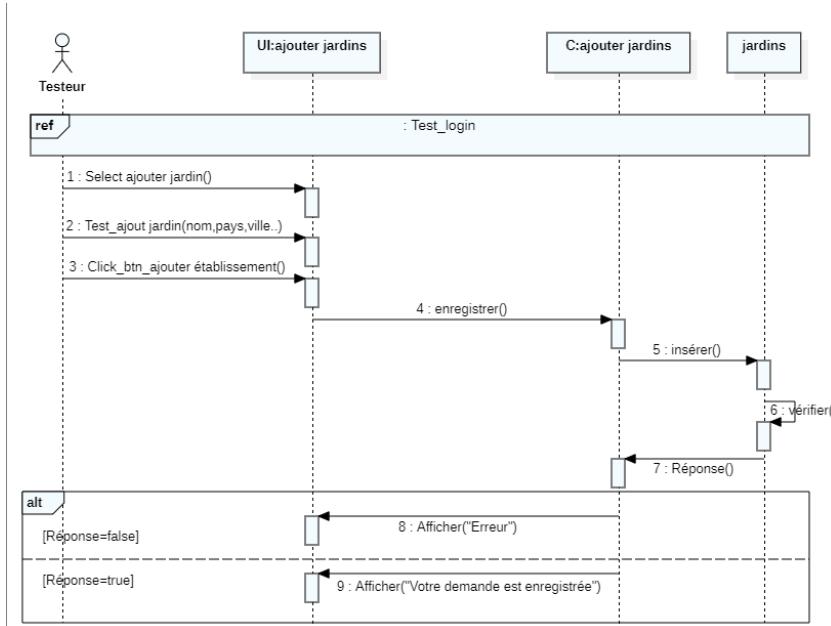


FIGURE 4.25 – Diagramme de séquence détaillé «tester l'ajout d'un jardin dans l'espace parent»

Le diagramme 4.26 définit le diagramme de séquence de la fonctionnalité «tester l'ajout d'un statut à un enfant dans l'espace parent»

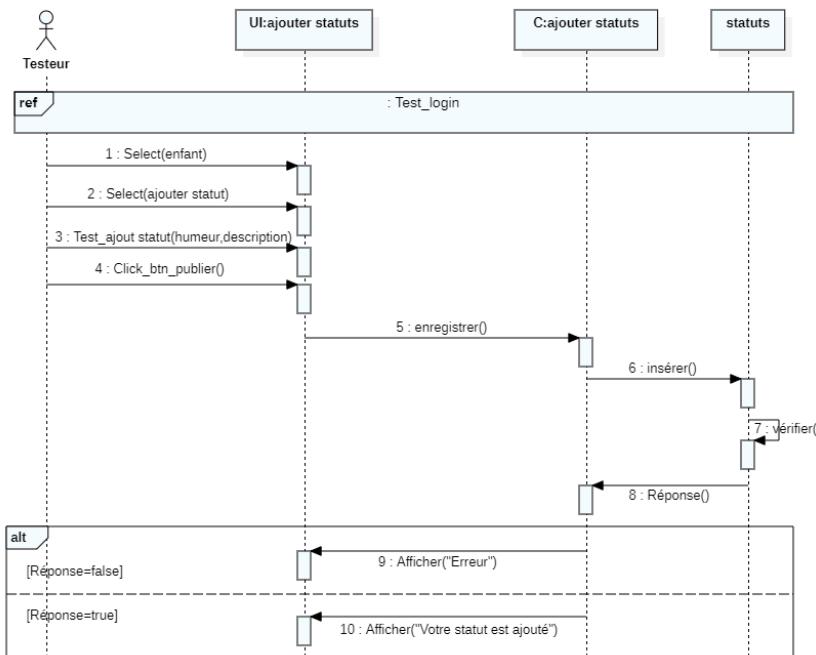


FIGURE 4.26 – Diagramme de séquence détaillé «tester l'ajout d'un statut dans l'espace parent»

Le diagramme 4.27 illustre le diagramme de séquence de la fonctionnalité «tester l'ajout d'un membre de la famille dans l'espace parent».

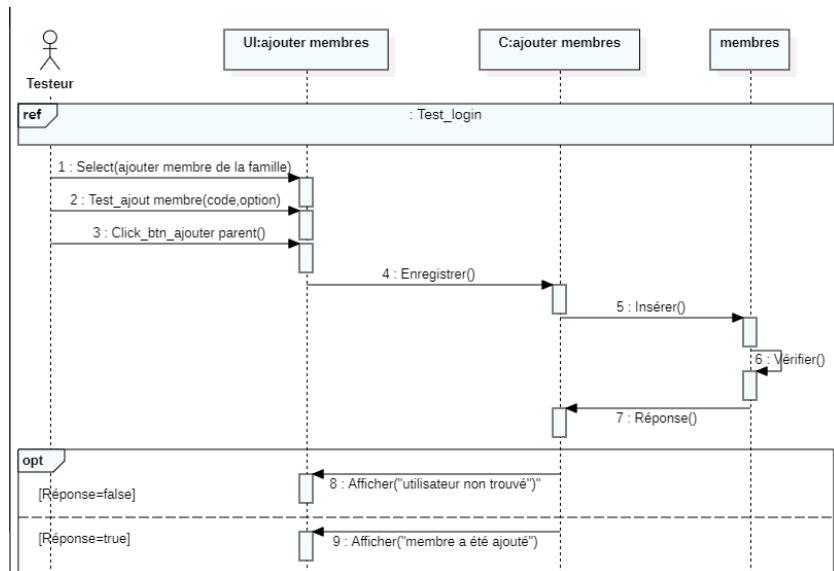


FIGURE 4.27 – Diagramme de séquence détaillé «tester l'ajout d'un membre de la famille dans l'espace parent»

Le diagramme 4.28 décrit le diagramme de séquence de la fonctionnalité «tester l'ajout d'un message dans l'espace parent»

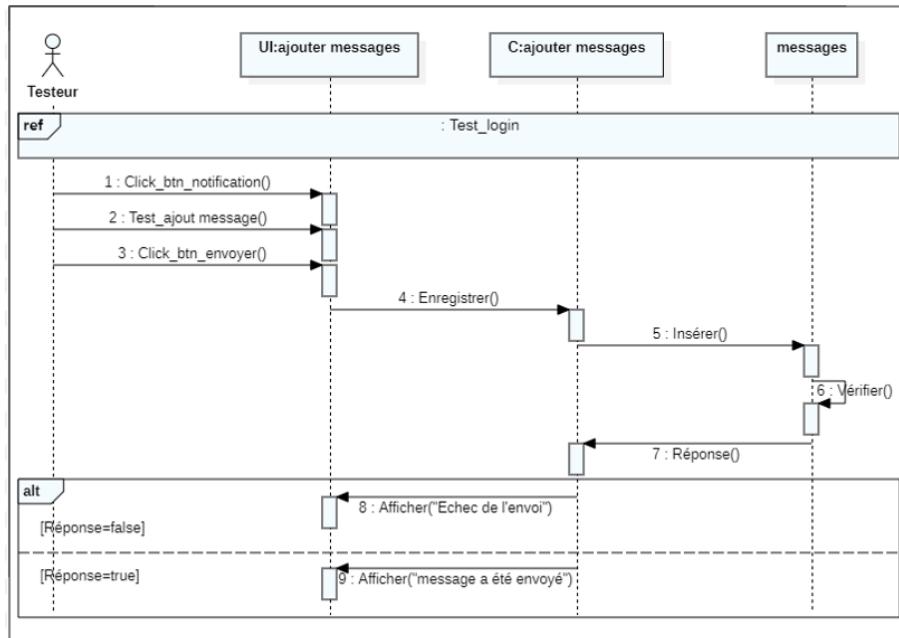


FIGURE 4.28 – Diagramme de séquence détaillé «tester l'ajout d'un message dans l'espace parent»

## 4.3 Réalisation du Release 2

Dans cette partie, nous présentons les différents interfaces réalisées relatives au Release 2.

### 4.3.1 Interfaces à tester

Cette partie présente les interfaces à tester spécifiques au Release 2.

Les figures 4.29 et 4.30 présentent l'interface de l'ajout d'un enfant dans l'espace parent testé.

The form contains the following fields:

- Nom:** nada
- Date de naissance:** 05/02/2021
- Sexe de votre enfant\*:**  FILLE  GARÇON
- Description supplémentaire:** A text area asking if the child has specific activities, dietary preferences, chronic diseases, or allergies. It also asks if the child is allergic.
- l'image de votre enfant:** A file input field with a 'Browse' button.
- enregistrer:** A large purple 'enregistrer' (register) button at the bottom.

FIGURE 4.29 – Interface «Ajout enfant» testée

FIGURE 4.30 – Suite de l'interface «Ajout enfant» testée

La figure 4.31 présente l'interface testée de l'ajout d'un jardin.

Remplissez le formulaire suivant

Nom \*

Pays \*

Ville \*

Adresse \*

Ajouter établissement

FIGURE 4.31 – Interface «Ajout jardin» testée

Les deux figures 4.32 et 4.33 décrivent le déroulement du test de l'ajout d'un statut. Commençant par l'interface de l'espace enfant 4.32, en cliquant automatiquement sur le bouton «Ajouter un statut», l'interface 4.33 s'affiche.

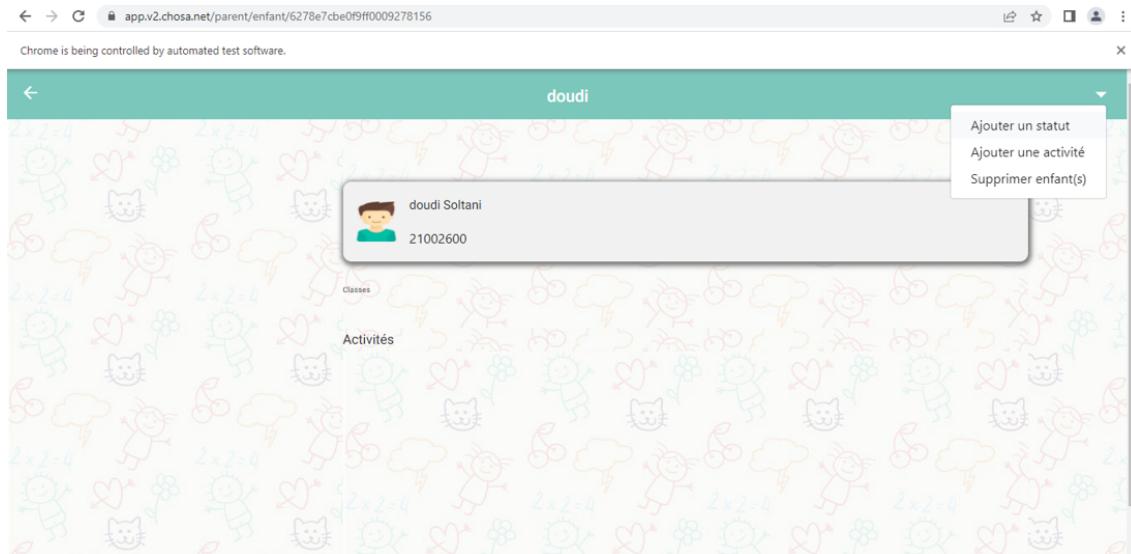


FIGURE 4.32 – Interface «Ajout statut» testée

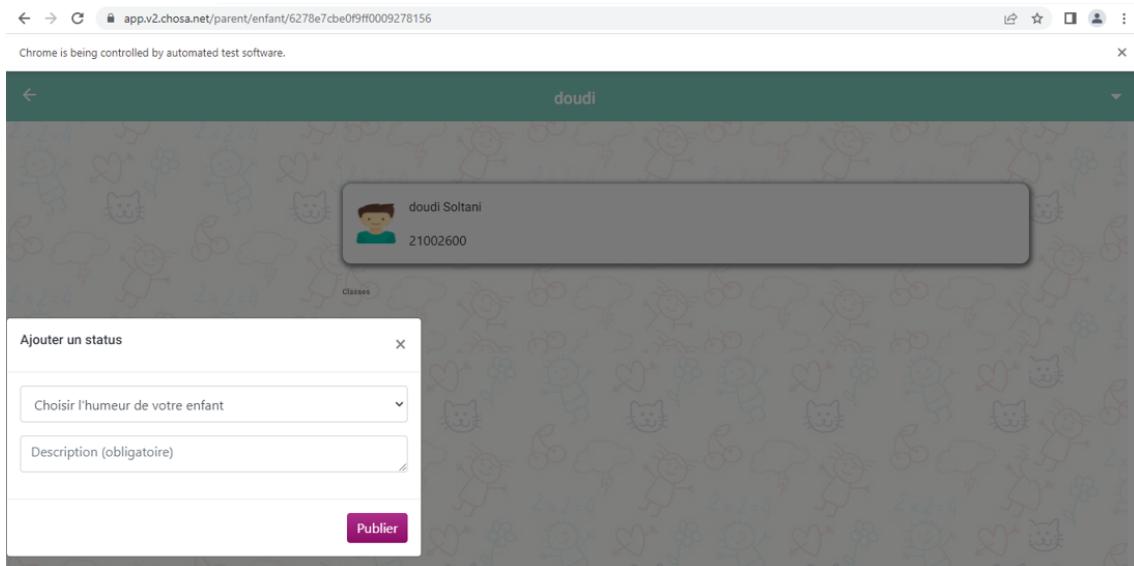


FIGURE 4.33 – Suite de l'interface «Ajout statut» testée

Les figures 4.34 et 4.35 présentent les interfaces qui expliquent le déroulement de test de l'ajout d'un membre de la famille dans l'espace parent.

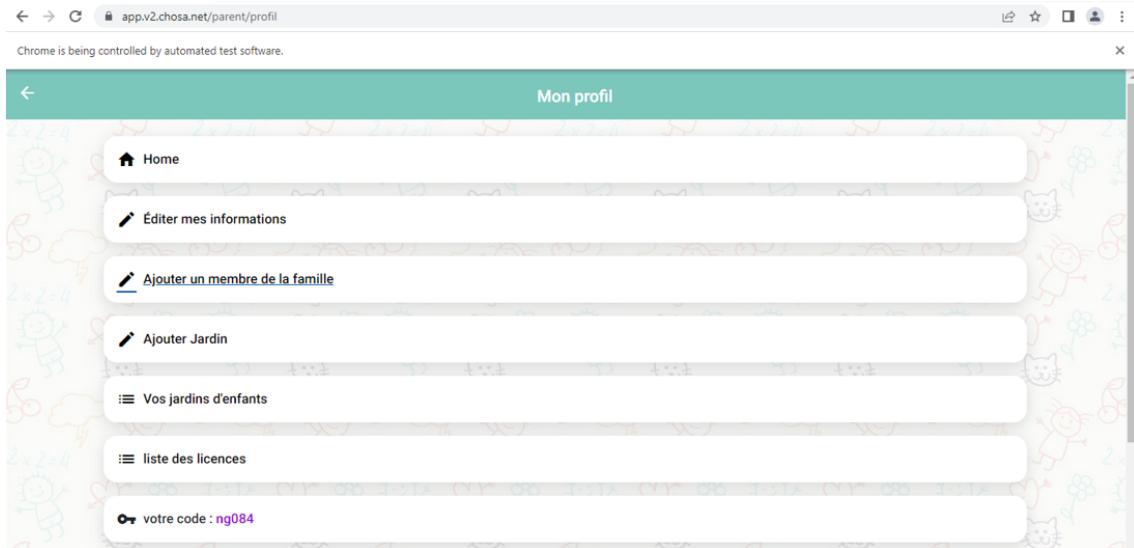


FIGURE 4.34 – Interface «Ajout membre» testée

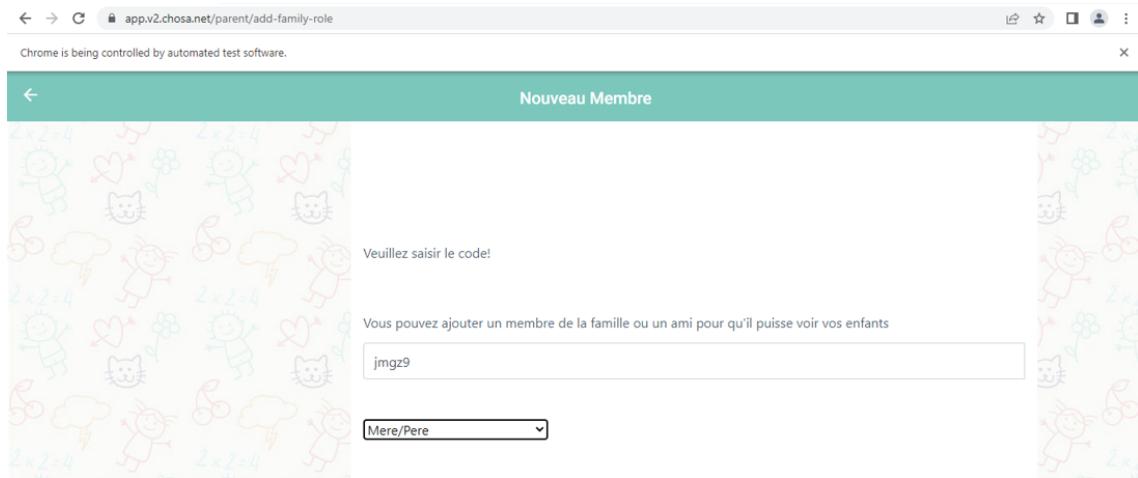


FIGURE 4.35 – Suite de l'interface «Ajout membre» testée

### 4.3.2 Interface Xray

Cette partie illustre les interfaces Xray qui présentent la description de nos tests réalisés dans l'espace parent.

La figure 4.36 affiche l'interface Xray qui illustre la description de test de l'ajout d'un enfant dans l'espace parent.

The screenshot shows a Jira Xray interface for a project named "chosatesting". The left sidebar shows navigation options like "Issues" (which is selected), "Components", "Code", and "Releases". The main content area displays a test case titled "Test Ajouter Enfant". The test description includes the following steps:

- 1-authentication
- 2-click\_btn\_ajout enfant
- 3-remplir formulaire
- 4-click\_btn\_enregister

The "Overall Execution Status" shows 1 PASSED out of 1 TOTAL TESTS. A table below provides more details about the test run:

| Rank | Key    | Summary               | Test Type | Dataset | #Defects | Status |
|------|--------|-----------------------|-----------|---------|----------|--------|
| 1    | CHOS-8 | TE002: ajouter enfant | Manual    |         | 0        | PASSED |

On the right side, there is a "Details" panel showing information such as Assignee (rana.soltani), Reporter (Nada el mokhtar), Labels (None), Sprint (CHOS Sprint 2), Priority (Medium), Test Plans (Open Test Plans), and Test Environments (Open Test Environments). The status bar at the bottom indicates the test was created 11 minutes ago and updated 2 minutes ago.

FIGURE 4.36 – Interface Xray de l'ajout d'un enfant

La figure 4.37 décrit l'interface Xray qui illustre la description de test de l'ajout d'un jardin dans l'espace parent.

The screenshot shows the Jira Xray interface for the project 'chos testing'. The left sidebar shows navigation options like Planning, Issues, Components, and Project settings. The main area displays a 'Test Ajouter Jardins' page with a 'Tests' section containing two entries: 'CHOS-2' and 'CHOS-12'. The status for both is 'PASSED'. On the right, there's a 'Details' panel showing fields like Assignee (Unassigned), Reporter (Nada el mokhtar), Labels (None), Priority (Medium), Test Plans (Open Test Plans), and Test Environments (Open Test Environments). A 'To Do' panel is also visible.

| Rank | Key     | Summary                    | Test Type | Dataset | #Defects | Status | Actions |
|------|---------|----------------------------|-----------|---------|----------|--------|---------|
| 1    | CHOS-2  | TC001: Test_Login          | Manual    | 0       | 0        | PASSED | ...     |
| 2    | CHOS-12 | TE001: test ajouter jardin | Manual    | 0       | 0        | FAILED | ...     |

FIGURE 4.37 – Interface Xray de l'ajout d'un jardin

La figure 4.38 détaille l'interface Xray qui illustre la description de test de l'ajout d'un statut dans l'espace parent.

The screenshot shows the Jira Xray interface for the project 'chos testing'. The left sidebar shows navigation options like Planning, Issues, Components, and Project settings. The main area displays a 'Test Ajouter Statut' page with a 'Tests' section containing two entries: 'CHOS-2' and 'CHOS-10'. Both entries have a status of 'PASSED'. On the right, there's a 'Details' panel showing fields like Assignee (Unassigned), Reporter (Nada el mokhtar), Labels (None), Priority (Medium), Test Plans (Open Test Plans), and Test Environments (Open Test Environments). A 'To Do' panel is also visible.

| Rank | Key     | Summary                    | Test Type | Dataset | #Defects | Status | Actions |
|------|---------|----------------------------|-----------|---------|----------|--------|---------|
| 1    | CHOS-2  | TC001: Test_Login          | Manual    | 0       | 0        | PASSED | ...     |
| 2    | CHOS-10 | TE001: test ajouter statut | Manual    | 0       | 0        | PASSED | ...     |

FIGURE 4.38 – Interface Xray de l'ajout d'un statut

La figure 4.39 affiche l'interface Xray qui illustre la description de test de l'ajout d'un membre de la famille.

The screenshot shows a Jira Software interface for a project named 'chos testing'. The main view displays a test case titled 'Ajouter Membre de famille' under the 'CHOS-22' issue. The test has been executed, resulting in 1 Passed and 1 Failed outcome. The failed test is identified as 'TE001: test ajouter membre de la famille'. The interface includes a sidebar with project navigation, a 'Tests' section with an attached file, and a 'Details' panel on the right showing assignee, reporter, labels, priority, and test plans.

Test Ajouter Membre de famille

Description  
1-authentication  
2-select menu → ajouter membre de la famille  
3-emplir zone de texte (code + option)  
4-click sur ajouter classe  
5-enseignant ajouté

Tests

Add Tests

Overall Execution Status

1 PASSED 1 FAILED

TOTAL TESTS: 2

| Rank | Key     | Summary                                  | Test Type | Dataset | #Defects | Status | Actions |
|------|---------|--|-----------|---------|----------|--------|---------|
| 1    | CHOS-22 | TC001: Test_1>Login                      | Manual    | grid    | 0        | PASSED | grid    |
| 2    | CHOS-22 | TE001: test ajouter membre de la famille | Manual    | grid    | 0        | FAILED | grid    |

To Do

Pinned fields

Click on the next to a field label to start pinning.

Details

Assignee: Unassigned

Reporter: Nada el mokhtar

Labels: None

Priority: Medium

Test Plans: Open Test Plans

Test Environments: Open Test Environments

More fields

Created 2 minutes ago  
Updated 2 minutes ago

Configure

FIGURE 4.39 – Interface Xray de l'ajout d'un membre

### 4.3.3 Interfaces du fichier "automation.log"

Cette partie illustre les interfaces du fichier "automation.log" qui va nous montrer les étapes des tests automatiques au niveau de l'espace parent et leurs résultats.

La figure 4.40 affiche le fichier "automation.log" qui montre le résultat du test d'«ajout enfant dans l'espace parent» : le test est passé avec succès.

| 1  | screenshots               | 5550  |
|----|---------------------------|---|
| 2  | Testcases                 | 5551 2022-05-17 10:59:58,558 - WDM - ===== WebDriver manager =====  |
| 3  | _pytest_cache             | 5552 2022-05-17 10:59:59,164 - WDM - INFO - Current google-chrome version is 101.0.4951   |
| 4  | __init__.py               | 5553 2022-05-17 10:59:59,164 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome                                  |
| 5  | confest.py                | 5554 2022-05-17 11:00:00,409 - WDM - INFO - Driver [C:\Users\nada\wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in |
| 6  | pytest.ini                | 5555 2022-05-17 11:00:01,809 - root - INFO - ***** TEST ADD KID PARENT *****  |
| 7  | test_llogin.py            | 5556 2022-05-17 11:00:35,340 - root - INFO - ***** LOGIN SUCCESSFUL *****   |
| 8  | test_AjouterEnfant.py     | 5557 2022-05-17 11:00:35,340 - root - INFO - ***** START ADDING KID TEST *****  |
| 9  | test_AjouterMembre.py     | 5558 2022-05-17 11:00:46,514 - root - INFO - ***** PROVIDING KID INFO *****   |
| 10 | test_AjouterJardin.py     | 5559 2022-05-17 11:01:01,128 - root - INFO - ***** saving Kid info *****  |
| 11 | test_AjouterEnseignant.py | 5560 2022-05-17 11:01:01,128 - root - INFO - ***** ADD KID TEST PASSED*****   |
| 12 | test_AjouterMSG.py        |   |

FIGURE 4.40 – Description du «test ajout enfant» dans fichier "automation.log"

La figure 4.41 affiche le fichier "automation.log" qui montre le résultat du «test de l'ajout d'un jardin dans l'espace parent» : le test est passé avec succès.

```

File Edit View Navigate Code Refactor Run Tools Git Window Help pythonProject - automation.log
pythonProject logs automation.log
Project Pages
Pages
  __init__.py
  AjouteEnfant.py
  AjouteEnseignant.py
  AjouteJardin.py
  AjouteMembreFamille.py
  AjouteMSG.py
  AjouteStatut.py
  login_page.py
Reports
screenshots
Testcases
  .pytest_cache
    __init__.py
    confest.py
    pytest.ini
    test_login.py
    test_2AjouterEnfant.py
    test_3AjouterMembre.py
    test_4AjouterJardin.py
    test_4AjouterEnseignant.py
    test_AjouteMSG.py
  testData
    LoginData.xlsx
Utilities
run.bat
External Libraries
Scratches and Consoles
Terminal Local + v
automation.log x test_4AjouterJardin.py x
2022-05-17 11:15:42,232 - WDM - INFO -
2022-05-17 11:15:42,235 - WDM - INFO - ===== WebDriver manager =====
2022-05-17 11:15:42,653 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-17 11:15:42,653 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-17 11:15:43,372 - WDM - INFO - Driver [C:\Users\nada\wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in
2022-05-17 11:15:45,280 - root - INFO - ***** TEST ADD GARDEN *****
2022-05-17 11:16:11,278 - root - INFO - ***** START ADDING GARDEN *****
2022-05-17 11:50:01,372 - WDM - INFO -
2022-05-17 11:50:01,372 - WDM - INFO - ===== WebDriver manager =====
2022-05-17 11:50:02,729 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-17 11:50:02,729 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-17 11:50:03,691 - WDM - INFO - Driver [C:\Users\nada\wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in
2022-05-17 11:50:06,309 - root - INFO - ***** TEST ADD GARDEN *****
2022-05-17 11:50:34,365 - root - INFO - ***** START ADDING GARDEN *****
2022-05-17 11:50:34,366 - root - INFO - ***** TEST ADDING GARDEN *****
2022-05-17 11:50:34,366 - root - INFO - ***** PROVIDING GARDEN INFORMATION *****
2022-05-17 11:51:05,882 - root - INFO - ***** ADDING GARDEN TEST PASSED *****

```

FIGURE 4.41 – Description du «test ajout jardin» dans fichier automation.log

La figure 4.42 décrit le fichier "automation.log" qui montre le résultat du test d'«ajout statut dans l'espace parent» : le test est passé avec succès.

```

File Edit View Navigate Code Refactor Run Tools Git Window Help pythonProject - automation.log
pythonProject logs automation.log
Project Pages
Pages
  __init__.py
  AjouteMembreFamille.py
  AjouteMSG.py
  AjouteStatut.py
  login_page.py
Reports
screenshots
Testcases
  .pytest_cache
    __init__.py
    confest.py
    pytest.ini
    test_login.py
    test_2AjouterEnfant.py
    test_3AjouterMembre.py
    test_4AjouterJardin.py
    test_AjouteMSG.py
  testData
    LoginData.xlsx
Utilities
run.bat
External Libraries
Scratches and Consoles
Terminal Local + v
automation.log x test_4AjouterJardin.py x
2022-05-11 00:51:14,881 - WDM - INFO - ===== WebDriver manager =====
2022-05-11 00:51:16,526 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-11 00:51:16,527 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-11 00:51:17,484 - WDM - INFO - Driver [C:\Users\nada\wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in
2022-05-11 00:51:21,192 - root - INFO - ***** TEST ADD KID PARENT *****
2022-05-11 00:51:57,432 - root - INFO - ***** LOGIN SUCCESSFUL *****
2022-05-11 00:51:57,432 - root - INFO - ***** START ADDING KID TEST *****
2022-05-11 00:52:10,747 - root - INFO - ***** PROVIDING KID INFO *****
2022-05-11 00:52:25,555 - root - INFO - ***** saving kid info *****
2022-05-11 00:52:25,555 - root - INFO - ***** ADD KID TEST PASSED*****
2022-05-11 00:52:26,556 - root - INFO - ***** START ADDING STATUT *****
2022-05-11 00:52:45,212 - root - INFO - ***** ADD status passed*****
2022-05-11 00:54:10,924 - WDM - INFO -

```

C:\Users\nada\PycharmProjects\pythonProject\Pages\login\_page.py:21: DeprecationWarning: find\_element\_by\_\* commands are deprecated. Please use find\_element() instead  
self.driver.find\_element\_by\_id(self.textbox\_username\_id).send\_keys(username)

test\_4AjouterJardin.py::Test\_addGarden::test\_ajouteJardin  
C:\Users\nada\PycharmProjects\pythonProject\Pages\login\_page.py:26: DeprecationWarning: find\_element\_by\_\* commands are deprecated. Please use find\_element() instead  
self.driver.find\_element\_by\_id(self.textbox\_password\_id).send\_keys(password)

FIGURE 4.42 – Description du «test ajout statut» dans fichier "automation.log"

La figure 4.43 affiche le fichier "automation.log" qui montre le résultat du test d'«ajout membre dans l'espace parent» : le test n'est pas passé.

```

AjouterMembreDeFamille 3301
AjouterMSG.py 3302 2022-05-09 11:49:01,755 - WDM - INFO - ===== WebDriver manager =====
AjouteStatut.py 3303 2022-05-09 11:49:02,547 - WDM - INFO - Current google-chrome version is 101.0.4951
login_page.py 3304 2022-05-09 11:49:02,547 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
Reports 3305 2022-05-09 11:49:04,514 - WDM - INFO - Driver [C:\Users\nada].wde\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found
assets 3306 2022-05-09 11:49:06,393 - root - INFO - ****TEST ADD KID PARENT ****
screenshots 3307 2022-05-09 11:50:02,882 - root - INFO - ***** LOGIN SUCCESSFUL *****
Testcases 3308 2022-05-09 11:50:02,882 - root - INFO - ***** START ADDING MEMBER OF FAMILY*****
3309 2022-05-09 11:50:32,607 - root - ERROR - ***** ADD MEMBER TEST FAILED*****
```

-- Docs: <https://docs.pytest.org/en/stable/how-to/capture-warnings.html>

===== short test summary info =====

FAILED Testcases/test\_AjoutMembre.py::test\_addKid::test\_ajouterEnfant - selenium.common.exceptions.InvalidSessionIdException: Message: invalid session id

===== 1 failed, 10 warnings in 92.39s (0:01:32) =====

FIGURE 4.43 – Description du «test ajout membre» dans fichier "automation.log"

#### 4.3.4 Interface Jenkins

Dans cette partie, nous allons finir par présenter les interfaces Jenkins des tests automatiques du Release2.

Les figures 4.44 et 4.45 montrent la configuration de Jenkins avec notre projet déployé sur Gitlab.

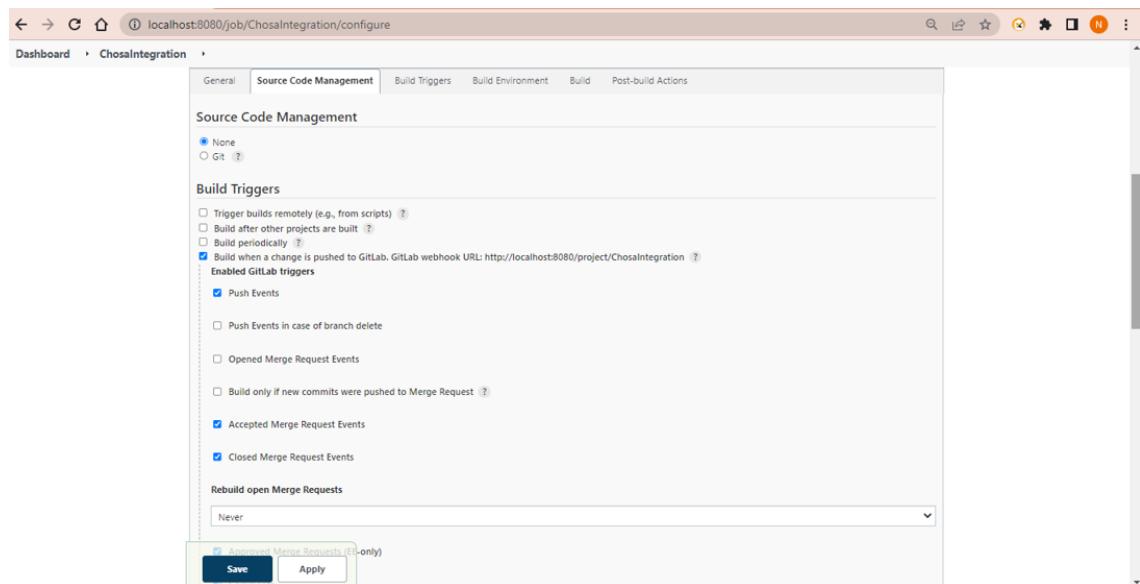


FIGURE 4.44 – Interface de configuration Jenkins avec Gitlab

La figure 4.45 présent le Script qui permet d'exécuter les tests de Release2 sur Jenkins.

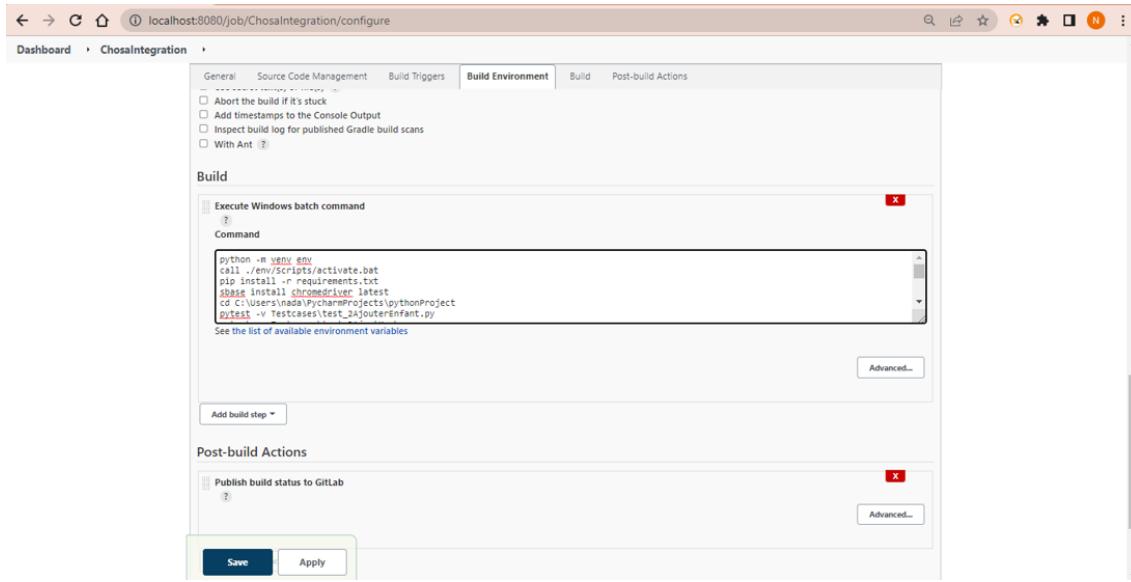


FIGURE 4.45 – Interface build de Jenkins

Les interfaces 4.46, 4.47 et 4.48 présentent le résultat de l'intégration avec Jenkins des tests automatiques.

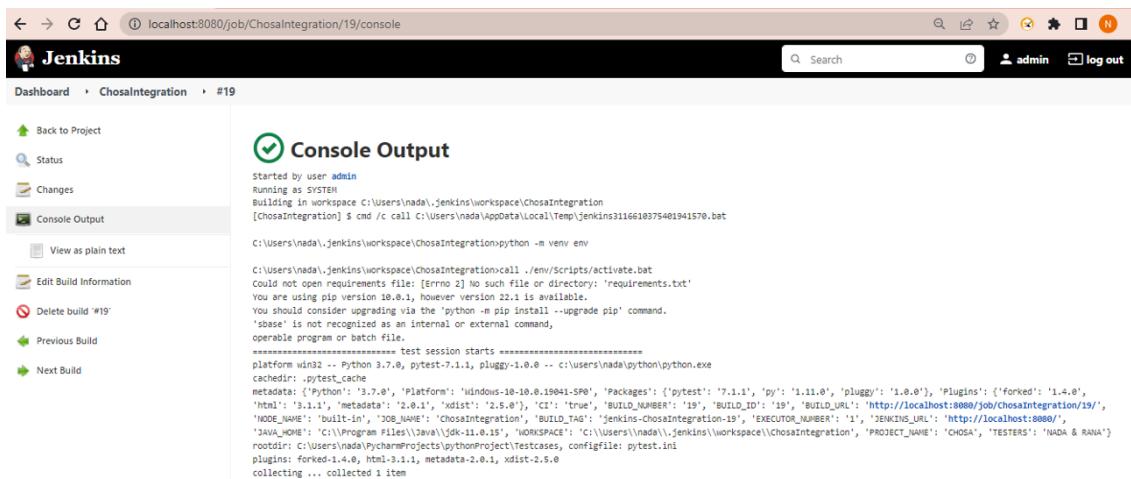


FIGURE 4.46 – Résultat de l'exécution de test «Ajout enfant» sur console Jenkins

FIGURE 4.47 – Résultat de l'exécution de test «Ajout jardin» sur console Jenkins

```
Current google-chrome version is 101.0.4951
GET LATEST chromedriver version for 101.0.4951 google-chrome
Driver [C:\Users\nadal\Downloads\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in cache
..... Captured log setup .....
INFO WdM\logger.py:27

INFO WdM\logger.py:27 ===== WebDriver manager =====
INFO WdM\logger.py:27 Current google-chrome version is 101.0.4951
INFO WdM\logger.py:27 Get LATEST chromedriver version for 101.0.4951 google-chrome
INFO WdM\logger.py:27 Driver [C:\Users\nadal\Downloads\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in cache
..... Captured log call .....
INFO root: test_2AjouterEnfant.py:17 ===== TEST ADD KID PARENT =====
INFO root: test_2AjouterEnfant.py:29 ===== LOGIN SUCCESSFUL =====
INFO root: test_2AjouterEnfant.py:38 ===== START ADDING KID TEST =====
Test.addmember.test_ajouternembre

self = <Testcases.test_3Ajouternembre.Test_addmember object at 0x000001D03AF7AD0>
setup = < selenium.webdriver.chrome.webdriver.WebDriver (session=6a39c5a4288b53b52a6e659803d8ef8f)>

    @pytest.mark.sanity
def test_ajouternembre(self,setup):
    self.logger.info("===== TEST ADD KID PARENT =====")
    self.driver = setup
    self.driver.get(self.baseURL)
    self.driver.maximize_window()

    #create login object
    self.lp=LoginPage(self.driver)
    self.lp.clickOnLogin()
    # we still need to login
    self.lp.setUsername(self.username)
    self.lp.setPassword(self.password)
    self.logger.info("===== LOGIN SUCCESSFUL =====")
    self.logger.info("===== START ADDING MEMBER OF FAMILY=====")
    self.member = AjouterMember(self.driver)
    self.member.clickOnAddmember()
    self.member.clickOnAddmember()

Testcases/test_3Ajouternembre.py:31:
```

FIGURE 4.48 – Résultat de l'exécution de test «Ajout membre» sur console Jenkins

#### 4.4 Conclusion

Dans ce chapitre, nous avons détaillé la conception et la réalisation du Release 2 : Les tests manuels et automatiques de l'espace parent. Dans le chapitre suivant, nous allons établir la même démarche pour la mise en place du Release 3 : Les tests manuels et automatiques de l'espace jardin.

# **chapitre 5**

## **Release 3 : Les tests manuels et automatiques de l'espace jardin**

### **5.1 Introduction**

Dans ce chapitre, nous allons présenter notre Release «Les tests manuels et automatiques de l'espace jardin». l'étude de ce release couvre les diagrammes de cas d'utilisation détaillés, les diagrammes de séquence et les interfaces de réalisation de ce Release.

### **5.2 Identification de backlog de release3**

Le tableau 5.9 décrit les différents éléments spécifiques au Release 3.

TABLE 5.9 – Identification de Backlog de release 3

| Numéro tache | Backlog de Produit  | Priorité | Estimation |
|--------------|---|----------|------------|
| 1            | Implémenter et exécuter les tests manuels de l'espace jardin                | 1        | Faible     |
| 1.1          | Test manuel de l'ajout et suppression d'un enfant dans l'espace jardin      | 1        | Faible     |
| 1.2          | Test manuel de l'ajout d'une activité dans l'espace jardin                  | 1        | Faible     |
| 1.3          | Test manuel de l'ajout d'un enseignant dans l'espace jardin                 | 1        | Faible     |
| 1.4          | Test manuel de l'ajout d'une classe dans l'espace jardin                    | 1        | Faible     |
| 2            | Implémenter et exécuter les tests automatiques de l'espace jardin           | 2        | Fort       |
| 2.1          | Test automatique de l'ajout et suppression d'un enfant dans l'espace jardin | 2        | Fort       |
| 2.2          | Test automatique de l'ajout d'une activité dans l'espace jardin             | 2        | Fort       |
| 2.3          | Test automatique de l'ajout d'un enseignant dans l'espace jardin            | 2        | Fort       |
| 2.4          | Test automatique de l'ajout d'une classe dans l'espace jardin               | 2        | Fort       |

### 5.2.1 Diagrammes de cas d'utilisation détaillés de Relese 3

Cette partie expose les diagrammes de cas d'utilisation de notre troisième release «Les tests manuels et automatiques de l'espace jardin».

premièrement,nous allons introduire le diagramme de cas d'utlisation globale de ce release «Les tests manuels et automatiques de l'espace jardin.».deuxièmement,nous allons présenter le diagramme de cas d'utlisation détaillé concernant «Les tests manuels de l'espace jardin» et finalement nous allons présenter le diagramme de cas d'utlisation détaillé concernant «Les tests automatiques de l'espace jardin» La figure 5.49 expose le diagramme de cas d'utilisation global qui décrit d'une manière générale notre derinière Release.

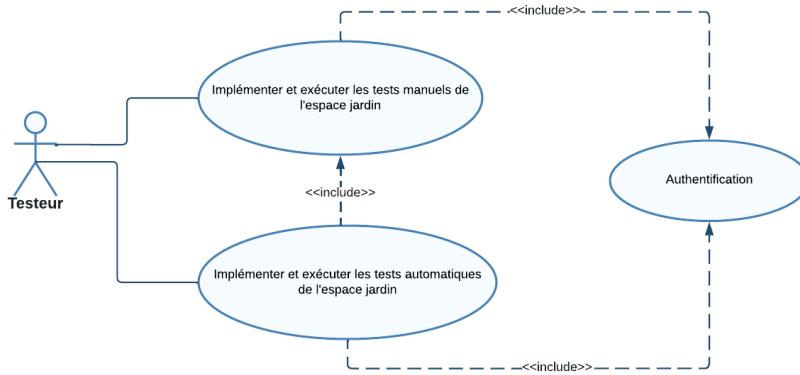


FIGURE 5.49 – Diagramme de cas d'utilisation «tests manuels et automatiques de l'espace jardin»

Pour expliquer plus en détail notre troisième Release , nous allons présenter dans la figure 5.50 le diagramme de cas d'utilisation de «tests manuels réalisés au niveau de l'espace jardin» .



FIGURE 5.50 – Diagramme de cas d'utilisation «Implémenter et exécuter les tests manuels de l'espace jardin»

Après, nous allons présenter dans la figure 5.51 le diagramme de cas d'utilisation des tests automatiques réalisés par le testeur au niveau de l'espace jardin.

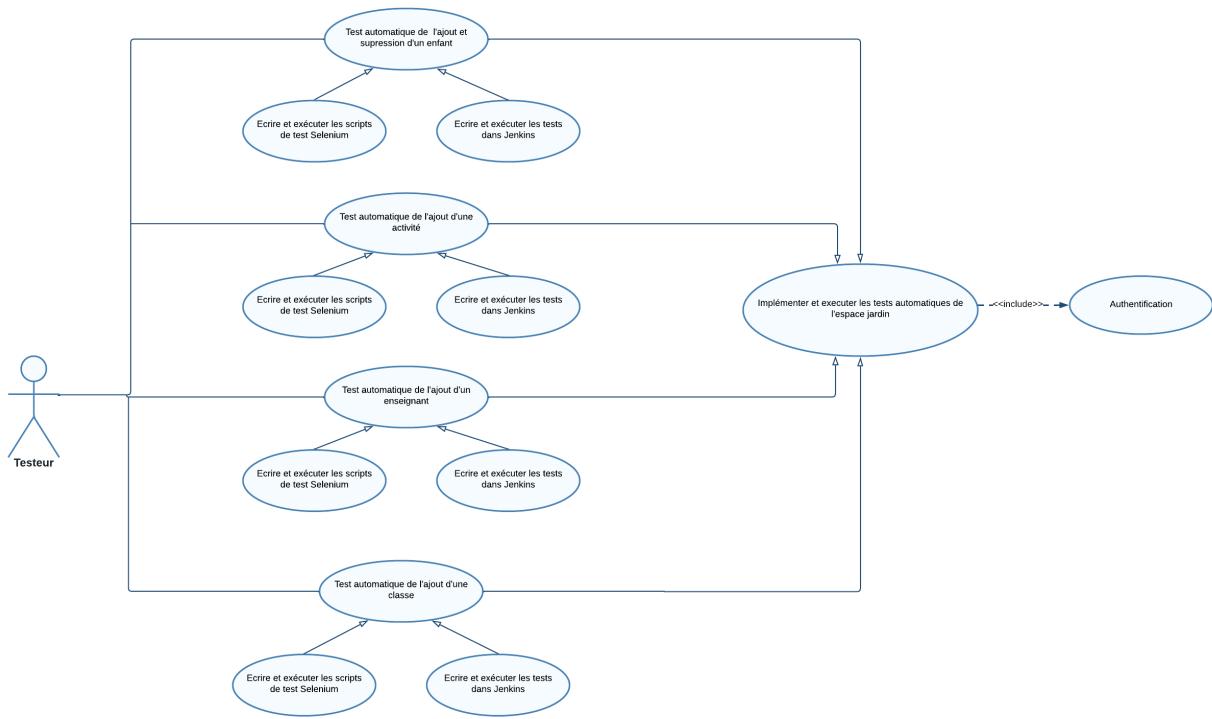


FIGURE 5.51 – Diagramme de cas d'utilisation «Implémenter et exécuter les tests automatiques de l'espace jardin»

## 5.2.2 Description détaillée du Release 3

le tableau 5.10 décrit le scénario suivi par le testeur pour réaliser «Tests manuels de l'espace jardin»

TABLE 5.10 – Description du cas d'utilisation «Tests manuels de l'espace jardin»

|                                   |   |
|-----------------------------------|---|
| Cas d'utilisation                 | Tests manuels de l'espace jardin  |
| Acteur                            | Testeur   |
| Pré-Condition                     | <ul style="list-style-type: none"> <li>- Connexion à Chosa</li> <li>- Connexion à Xray</li> </ul>   |
| Post condition                    | Tests manuels de l'espace jardin créés et exécutés  |
| Description du scénario principal | <p>Pour exécuter manuellement les tests de l'espace jardin, le testeur suit les étapes suivantes :</p> <ul style="list-style-type: none"> <li>- Exécution manuelle les tests réalisés au niveau de l'espace jardin</li> <li>- Analyse des résultats trouvés</li> <li>- Implémentation de la description des tests sur Xray</li> </ul> |

Le tableau 5.11 décrit le scénario suivi par le testeur pour réaliser «Test automatique de l'espace jardin».

TABLE 5.11 – Description du cas d'utilisation «Test automatique de l'espace jardin»

|                                   |  |
|-----------------------------------|--|
| Cas d'utilisation                 | Test automatique de l'espace jardin  |
| Acteur                            | Testeur  |
| Pré-Condition                     | Tests manuels de l'espace jardin établis   |
| Post condition                    | Tests automatiques de l'espace jardin créés et exécutés  |
| Description du scénario principal | <p>Pour automatiser les tests de l'espace jardin, le testeur suit les étapes suivantes :</p> <ul style="list-style-type: none"> <li>- configuration de Jenkins</li> <li>- implémentation et exécution des tests de l'espace jardin sur le job Jenkins déjà créé</li> <li>- vérification sur la console Jenkins si les tests sont exécutés avec succès ou non.</li> </ul> |

### 5.2.3 Diagrammes de séquence du Release 3

Dans cette partie, nous allons présenter les diagrammes de séquence détaillés des fonctionnalités du Release3.

Le diagramme 5.52 décrit le diagramme de séquence de la fonctionnalité «tester l'ajout et suppression d'un enfant dans l'espace jardin».

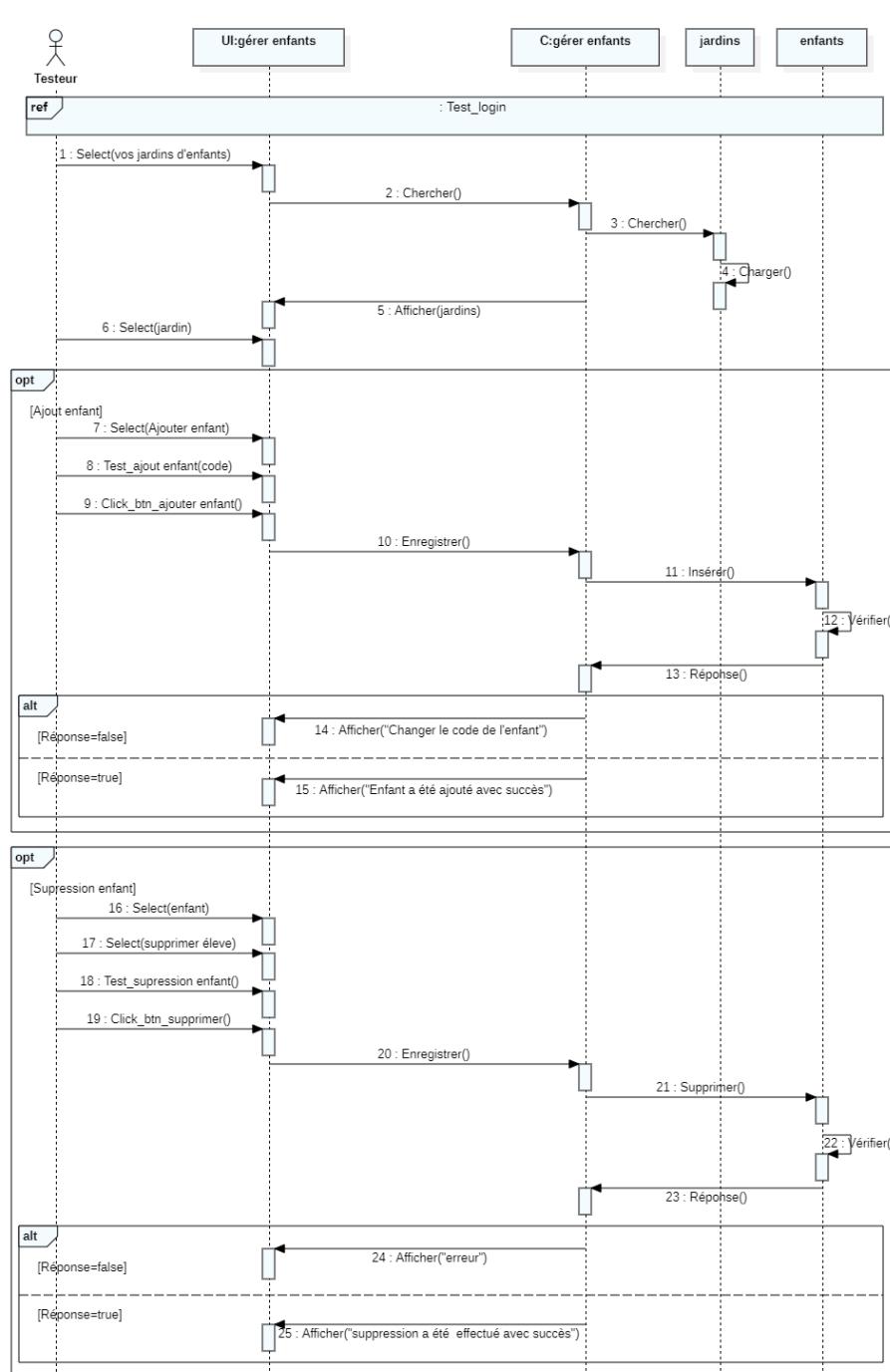


FIGURE 5.52 – Diagramme de séquence détaillé «tester l'ajout et suppression d'un enfant dans l'espace jardin»

Le diagramme 5.53 illustre le diagramme de séquence de la fonctionnalité «tester l'ajout d'une activité dans l'espace jardin».

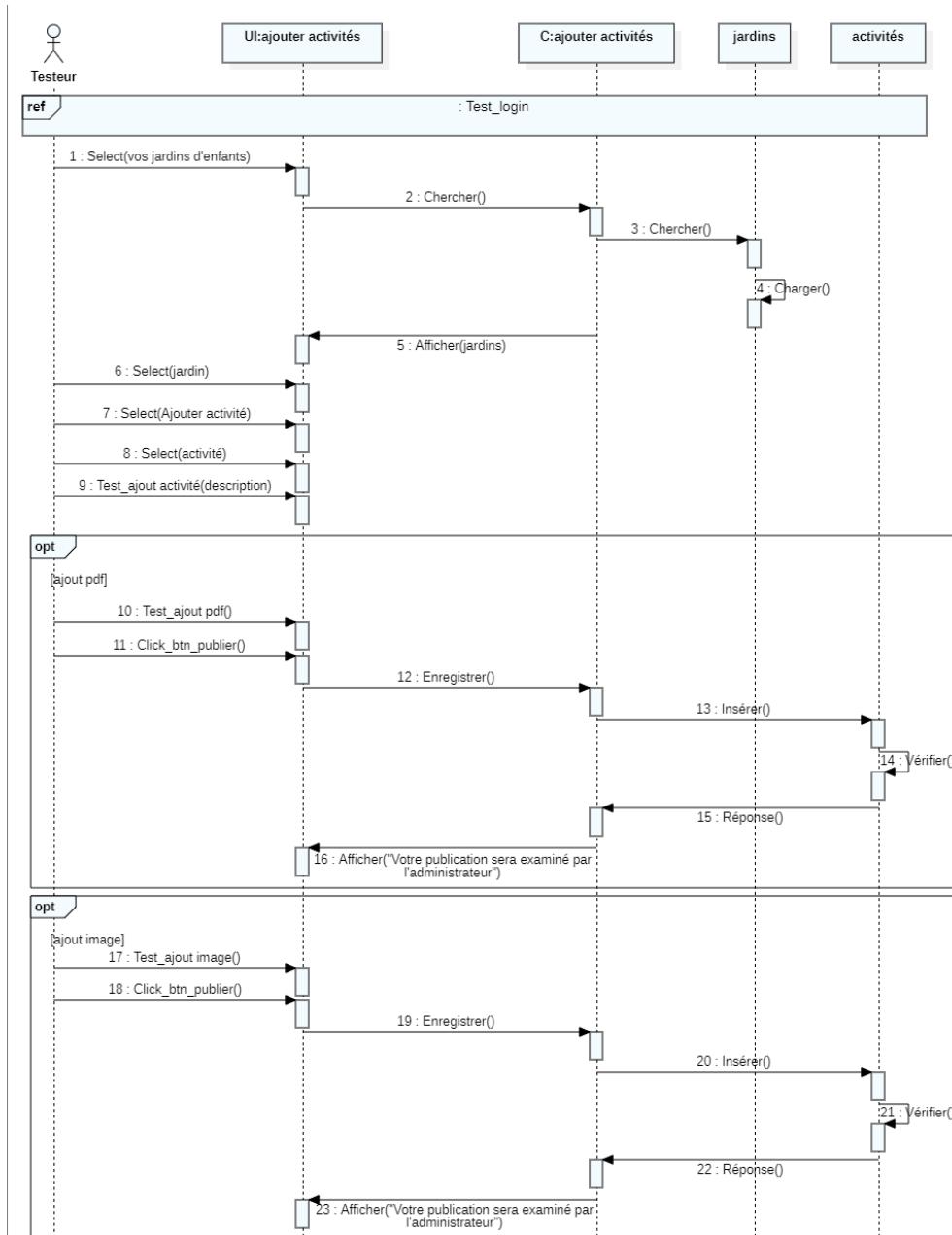


FIGURE 5.53 – Diagramme de séquence détaillé «tester l'ajout d'une activité dans l'espace jardin»

Le diagramme 5.54 schématise le diagramme de séquence de la fonctionnalité «tester l'ajout d'un enseignant dans l'espace jardin»

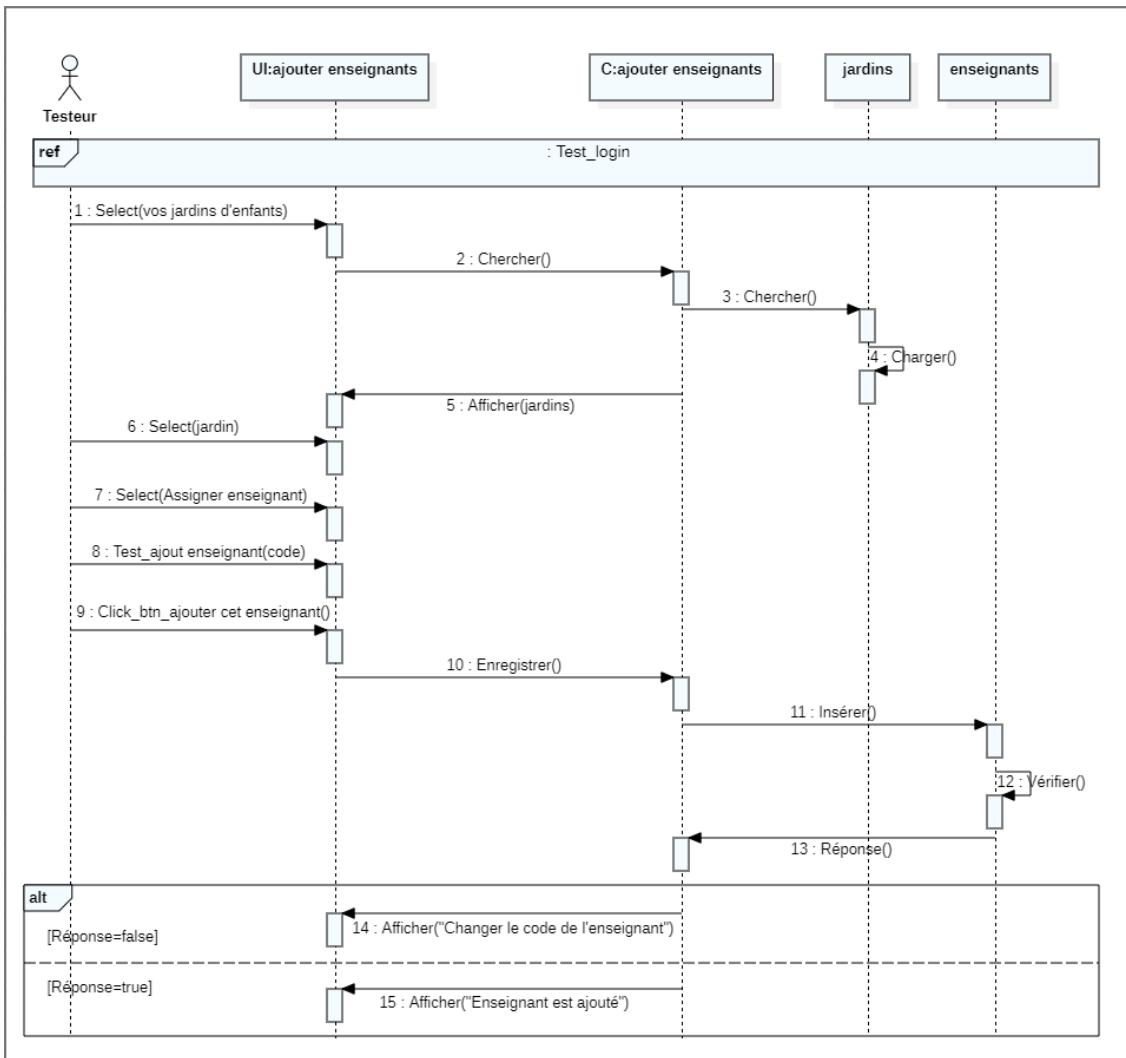


FIGURE 5.54 – Diagramme de séquence détaillé «tester l'ajout d'un enseignant dans l'espace jardin»

Le diagramme 5.55 schématise le diagramme de séquence de la fonctionnalité «tester l'ajout d'une classe dans l'espace jardin»

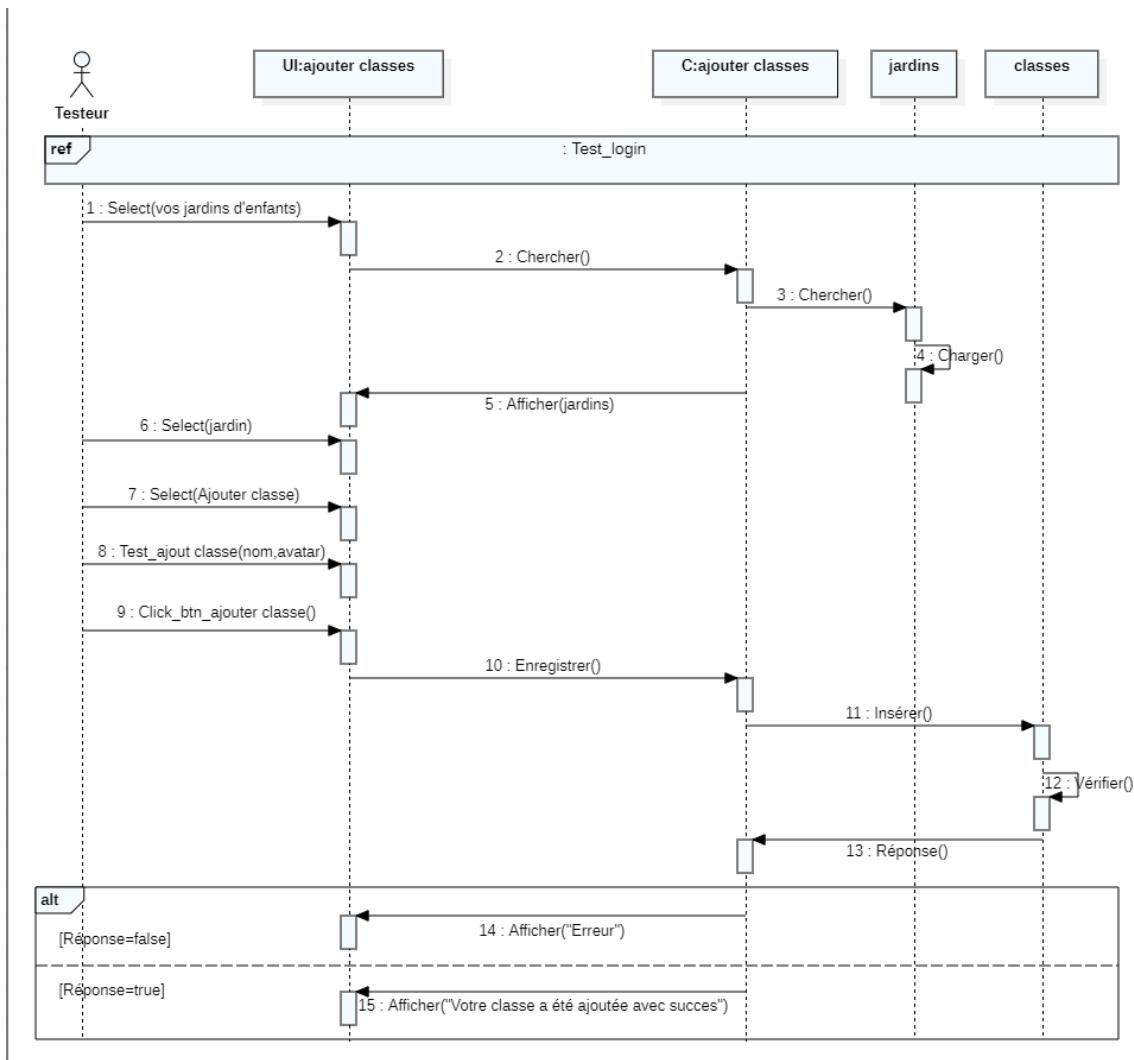


FIGURE 5.55 – Diagramme de séquence détaillé «tester l'ajout d'une classe dans l'espace jardin»

## 5.3 Réalisation

Dans cette partie, nous présentons les différents interfaces réalisées relatives au Release 3.

### 5.3.1 Interfaces à tester

Cette partie présente les interfaces à tester spécifiques au Release 3. La figure 5.56 illustre l'interface testée de l'ajout d'une activité.

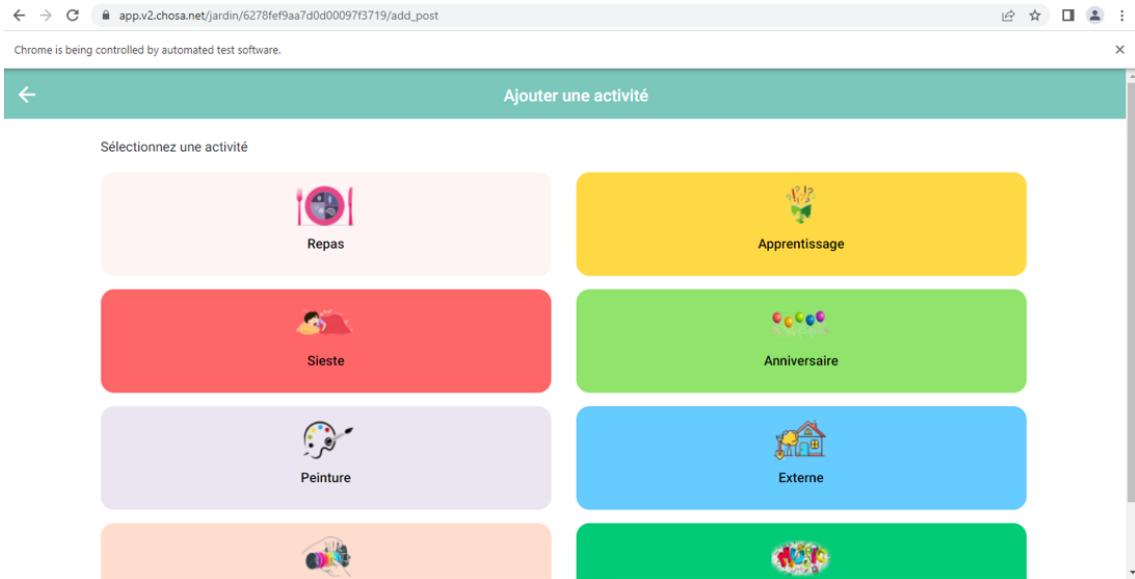


FIGURE 5.56 – Interface «Ajout activité» testée

La figure 5.57 montre l'interface testée de l'ajout d'un enseignant dans l'espace jardin.

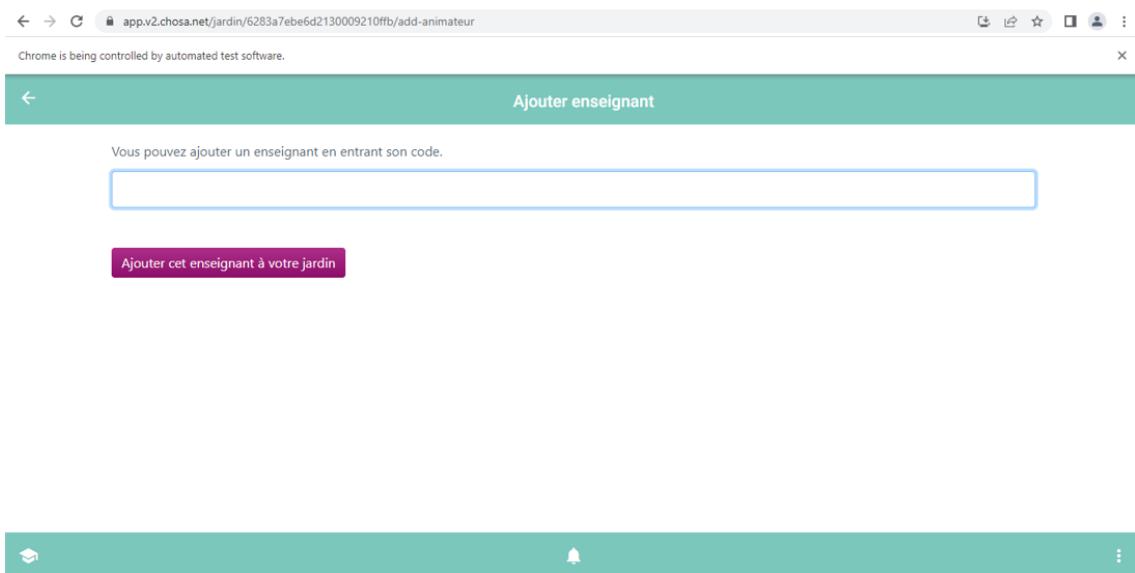


FIGURE 5.57 – Interface «Ajout enseignant» testée

La figure 5.58 décrit l'interface testée de l'ajout d'une classe dans l'espace jardin.

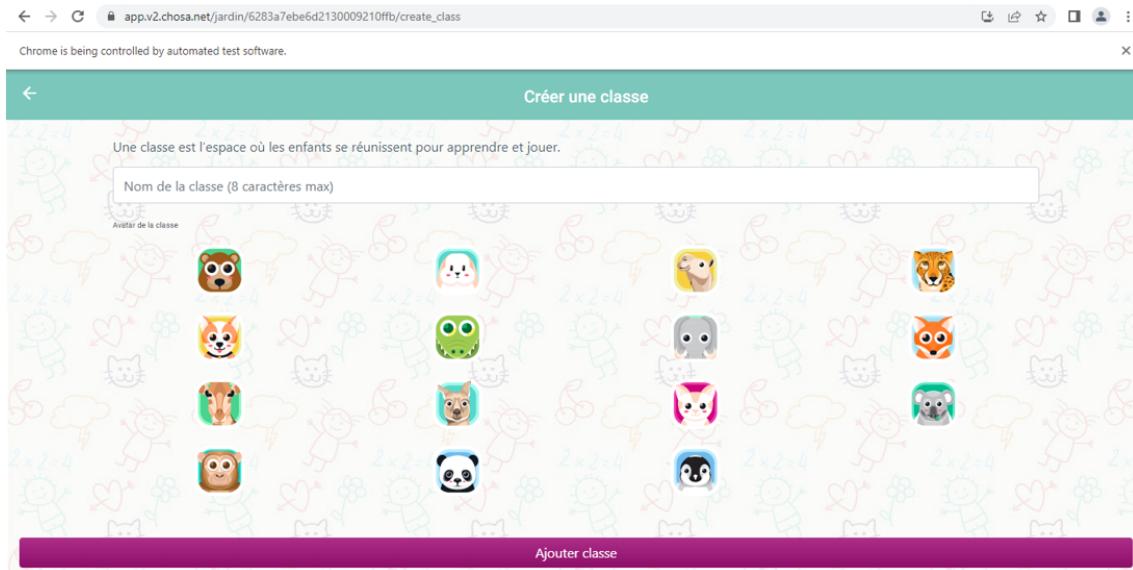


FIGURE 5.58 – Interface «Ajout classe» testée

### 5.3.2 Interfaces Xray des tests manuels

Cette partie illustre les interfaces Xray qui présentent la description de nos tests manuels réalisés dans l'espace jardin.

La figure 5.59 illustre l'interface Xray qui illustre la description de test de l'ajout et la suppression d'un enfant dans l'espace parent.

| Rank | Key     | Summary               | Test Type | Dataset   | #Defects | Status | Actions       |
|------|---------|-----------------------|-----------|-----------|----------|--------|---------------|
| 1    | CHOS-2  | TC001: Test_1login    | Manual    | grid icon | 0        | PASSED | grid icon ... |
| 2    | CHOS-8  | TB002: ajouter enfant | Manual    | grid icon | 0        | PASSED | grid icon ... |
| 3    | CHOS-14 | test supprimer enfant | Manual    | grid icon | 0        | PASSED | grid icon ... |

FIGURE 5.59 – Interface Xray de l'ajout d'un enfant

La figure 5.60 affiche l'interface Xray qui illustre la description de test de l'ajout d'une activité dans l'espace jardin.

The screenshot shows the Xray interface for a project named 'chosatesting'. The 'Test Ajouter Activité' page is displayed. The 'Description' section contains steps 1 through 5. The 'Tests' section shows an overall execution status with 2 passed and 1 failed test. The 'Overall Execution Status' bar is green. The 'Test Details' table lists three tests:

| Rank | Key     | Summary                             | Test Type | Dataset   | #Defects | Status | Actions       |
|------|---------|-------------------------------------|-----------|-----------|----------|--------|---------------|
| 1    | CHOS-2  | TC001: Test_Login                   | Manual    | grid icon | 0        | PASSED | grid icon ... |
| 2    | CHOS-18 | TE001: test ajouter Activité(image) | Manual    | grid icon | 0        | PASSED | grid icon ... |
| 3    | CHOS-19 | TE001: test ajouter Activité(pdf)   | Manual    | grid icon | 0        | FAILED | grid icon ... |

The 'Activity' tab is selected in the navigation bar. A comment box is present for adding comments.

FIGURE 5.60 – Interface Xray de l'ajout d'une activité

La figure 5.61 schématise l'interface Xray qui illustre la description de test de l'ajout d'un enseignant dans l'espace jardin.

The screenshot shows the Xray interface for a project named 'chosatesting'. The 'Test Ajouter Enseignant' page is displayed. The 'Description' section contains steps 1 through 6. The 'Tests' section shows an overall execution status with 1 passed and 1 executing test. The 'Overall Execution Status' bar is green. The 'Test Details' table lists two tests:

| Rank | Key     | Summary                        | Test Type | Dataset   | #Defects | Status    | Actions       |
|------|---------|--------------------------------|-----------|-----------|----------|-----------|---------------|
| 1    | CHOS-2  | TC001: Test_Login              | Manual    | grid icon | 0        | PASSED    | grid icon ... |
| 2    | CHOS-16 | TE001: test ajouter enseignant | Manual    | grid icon | 0        | EXECUTING | grid icon ... |

The 'Issues' tab is selected in the navigation bar. A comment box is present for adding comments.

FIGURE 5.61 – Interface Xray de l'ajout d'un enseignant

La figure 5.62 affiche l'interface Xray qui illustre la description de test de l'ajout d'une classe.

FIGURE 5.62 – Interface Xray de l'ajout d'une classe

### 5.3.3 Interfaces du fichier "automation.log" des tests automatiques

Cette partie illustre les interfaces du fichier "automation.log" qui va nous montrer les étapes des tests automatiques au niveau de l'espace jardin et leurs résultats.

La figure 5.63 affiche le fichier "automation.log" qui montre résultat du «test ajout et suppression enfant dans l'espace jardin» : le test est passé avec succès

```

2022-05-09 18:08:13,470 - WDM - INFO - ***** WebDriver manager *****
2022-05-09 18:08:13,482 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-09 18:08:13,482 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-09 18:08:14,797 - WDM - INFO - Driver [C:\Users\nada\wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in
2022-05-09 18:08:14,797 - WDM - INFO - ***** TEST ADD TEACHER *****
2022-05-09 18:08:35,021 - root - INFO - ***** LOGIN AS GARDEN PASSED *****
2022-05-09 18:08:35,021 - root - INFO - ***** ADD GARDEN *****
2022-05-09 18:09:06,520 - root - INFO - ***** ADDING GARDEN PASSED *****
2022-05-09 18:09:06,520 - root - INFO - ***** START TEST ADDING MEMBER *****
2022-05-09 18:09:06,520 - root - INFO - ***** TEST ADDING MEMBER PASSED *****
2022-05-09 18:09:08,520 - root - INFO - ***** TEST ADDING CLASS *****
2022-05-09 18:09:18,099 - root - INFO - ***** TEST ADDING CLASS PASSED *****
2022-05-09 18:09:18,100 - root - INFO - ***** TEST ADDING KID TO CLASS *****
2022-05-09 18:10:05,121 - WDM - INFO -

```

FIGURE 5.63 – Description du «test ajout enfant» dans fichier "automation.log"

La figure 5.64 affiche le fichier "automation.log" qui montre résultat du «test de ajout activité dans l'espace jardin» : le test est passé avec succès.

```

2022-05-17 14:22:23,616 - WDM - INFO - ===== WebDriver manager =====
2022-05-17 14:22:24,412 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-17 14:22:24,413 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-17 14:22:25,392 - WDM - INFO - Driver [C:\Users\nada.wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in
2022-05-17 14:22:27,726 - root - INFO - ***** START ADD GARDEN *****
2022-05-17 14:23:10,161 - root - INFO - ***** TEST ADDING GARDEN *****
2022-05-17 14:23:10,161 - root - INFO - ***** PROVIDING GARDEN INFORMATION *****

```

FIGURE 5.64 – Description du «test ajout activité» dans fichier "automation.log"

La figure 5.65 illustre le fichier "automation.log" qui montre résultat du test de «ajout enseignant dans l'espace jardin» : le test est passé avec succès.

```

2022-05-17 12:09:03,677 - WDM - INFO -
2022-05-17 12:09:04,729 - WDM - INFO - ===== WebDriver manager =====
2022-05-17 12:09:04,730 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-17 12:09:04,730 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-17 12:09:06,058 - WDM - INFO - Driver [C:\Users\nada.wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in
2022-05-17 12:09:08,420 - root - INFO - ***** TEST ADD TEACHER *****
2022-05-17 12:09:34,683 - root - INFO - ***** PROVIDING TEACHER *****
2022-05-17 12:09:41.017 - root - INFO - ***** TEST ADD TEACHER PASSED *****

```

FIGURE 5.65 – Description du «test ajout enseignant» dans fichier "automation.log"

La figure 5.66 détaille le fichier "automation.log" qui montre résultat du test de «ajout classe dans l'espace jardin» : le test est passé avec succès.

```

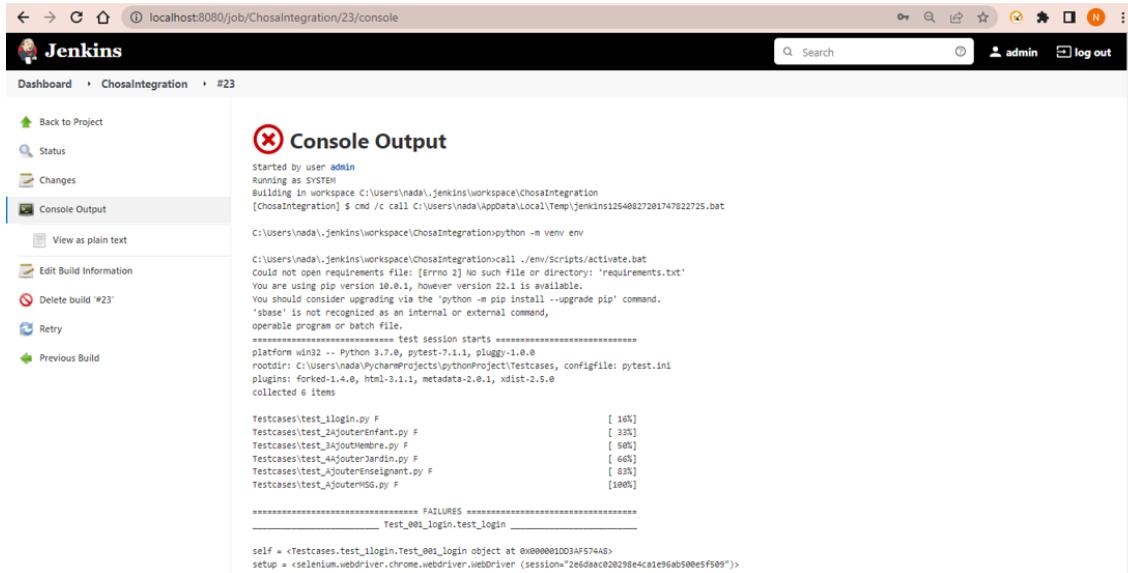
2022-05-09 17:32:21,643 - WDM - INFO - ===== WebDriver manager =====
2022-05-09 17:32:22,014 - WDM - INFO - Current google-chrome version is 101.0.4951
2022-05-09 17:32:22,014 - WDM - INFO - Get LATEST chromedriver version for 101.0.4951 google-chrome
2022-05-09 17:32:22,770 - WDM - INFO - Driver [C:\Users\nada.wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in
2022-05-09 17:32:24,301 - root - INFO - ***** TEST ADD TEACHER *****
2022-05-09 17:32:43,279 - root - INFO - ***** LOGIN AS GARDEN PASSED *****
2022-05-09 17:32:43,279 - root - INFO - ***** ADD GARDEN *****
2022-05-09 17:33:16,649 - root - INFO - ***** ADDING GARDEN PASSED *****
2022-05-09 17:33:16,650 - root - INFO - ***** START TEST ADDING MEMBER *****
2022-05-09 17:33:16,650 - root - INFO - ***** TEST ADDING MEMBER PASSED *****
2022-05-09 17:33:16,650 - root - INFO - ***** TEST ADDING CLASS *****
2022-05-09 17:33:27.113 - root - INFO - ***** TEST ADDING CLASS PASSED *****

```

FIGURE 5.66 – Description du «test ajout classe» dans fichier "automation.log"

### 5.3.4 Interfaces Jenkins des tests automatiques

Les interfaces 5.67, 5.68, 5.69 et 5.70 présentent les résultats de l'intégration avec Jenkins des tests automatiques.



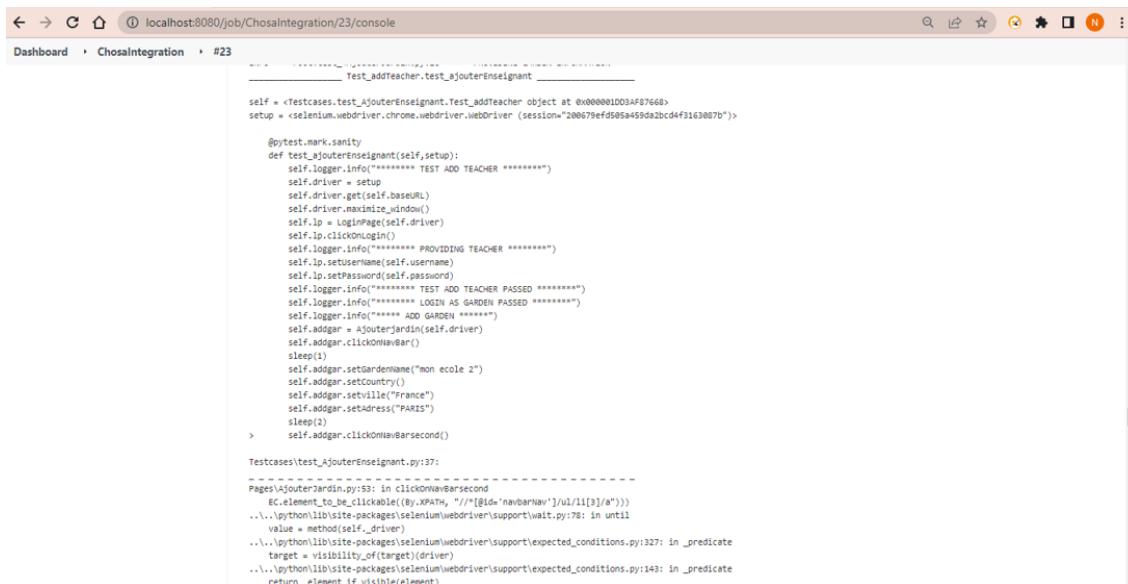
The screenshot shows the Jenkins interface for a job named 'ChosalIntegration' with build #23. The left sidebar has links for 'Back to Project', 'Status', 'Changes', and 'Console Output'. The main area is titled 'Console Output' with a red 'X' icon. It shows the command 'python -m venv env' being run, followed by several error messages related to Python requirements and environment setup. Below this, it lists test cases: 'Testcases\test\_ajouterEnfant.py F', 'Testcases\test\_ajouterMembre.py F', 'Testcases\test\_ajouterJardin.py F', 'Testcases\test\_ajouterEnseignant.py F', and 'Testcases\test\_ajouterMSG.py F'. A failure message '\*\*\*\*\* FAILURES \*\*\*\*\*' is shown, followed by a stack trace for 'Test\_001\_login.test\_login'. The log ends with a note about collecting 6 items.

```
C:\Users\nada\jenkins\workspace\ChosalIntegration>call .\env\scripts\activate.bat
Could not open requirements file: [Errno 2] No such file or directory: 'requirements.txt'
You are using pip version 18.0.1, however version 22.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
'base' is not recognized as an internal or external command,
operable program or batch file.
=====
test session starts =====
platform win32 -- Python 3.7.6, pytest-7.1.1, pluggy-1.0.0
rootdir: C:\Users\nada\PycharmProjects\pythonProject\tests, configfile: pytest.ini
plugins: forked-1.4.0, html-3.1.1, metadata-2.0.1, xdist-2.5.0
collected 6 items

Testcases\test_ajouterEnfant.py F [ 16%]
Testcases\test_ajouterMembre.py F [ 33%]
Testcases\test_ajouterJardin.py F [ 50%]
Testcases\test_ajouterEnseignant.py F [ 66%]
Testcases\test_ajouterMSG.py F [100%]

=====
***** FAILURES *****
----- Test_001_login.test_login -----
self = <Testcases.test_ajouterEnseignant.Test_001_login object at 0x000001D3A5F744B>
setup = <selenium.webdriver.chrome.webdriver.WebDriver (session="2eddaac020298e4c1e96ab500eef909")>
```

FIGURE 5.67 – Interface de console Jenkins après l'exécution des tests du Release3



This screenshot shows the Jenkins interface for build #23 of the 'ChosalIntegration' job. The left sidebar includes 'Dashboard', 'ChosalIntegration', and '#23'. The main content area displays a large amount of test log output. It starts with a success message for 'Test\_001\_login.test\_login'. Following this, there is a series of test cases under 'Testcases\test\_AjouterEnseignant.py': 'test\_ajouterEnseignant', 'test\_ajouterTeacher', 'test\_ajouterGarden', and 'test\_ajouterJardin'. Each of these tests includes its own block of code and associated log entries. For example, 'test\_ajouterEnseignant' shows steps like setting up a driver, logging in, providing teacher information, and adding a garden. The log concludes with a note about collecting 37 items.

```
----- Test_001_login.test_login -----
self = <Testcases.test_ajouterEnseignant.Test_001_login object at 0x000001D3A5F744B>
setup = <selenium.webdriver.chrome.webdriver.WebDriver (session="2eddaac020298e4c1e96ab500eef909")>

@ pytest.mark.sanity
def test_ajouterEnseignant(self, setup):
    self.logger.info("***** TEST ADD TEACHER *****")
    self.driver = setup
    self.driver.get(self.baseURL)
    self.driver.maximize_window()
    self.ip = LoginPage(self.driver)
    self.ip.clickOnLogin()
    self.logger.info("***** PROVIDING TEACHER *****")
    self.ip.setUserName(self.username)
    self.ip.setPassword(self.password)
    self.ip.clickOnAddTeacher()
    self.logger.info("***** ADD TEACHER PASSED *****")
    self.logger.info("***** LOGIN AS GARDEN PASSED *****")
    self.logger.info("***** ADD GARDEN *****")
    self.addgar = AjouterJardin(self.driver)
    self.addgar.clickOnNavBar()
    sleep(1)
    self.addgar.setGardenName("mon ecole 2")
    self.addgar.setCountry(),
    self.addgar.setVille("France")
    self.addgar.setAddress("PARIS")
    sleep(2)
    self.addgar.clickOnNavBarSecond()

Testcases\test_AjouterEnseignant.py:37:
-----
```

FIGURE 5.68 – Résultat de test :Exécution de test automatique sur Jenkins

```

----- Captured stderr setup -----

===== WebDriver manager =====
Current google-chrome version is 101.0.4951
Get LATEST chromedriver version for 101.0.4951 google-chrome
Driver [C:\Users\nada\wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in cache
----- Captured log setup -----
INFO  |com.webdriver.WebDriverManager|INFO|Captured log setup
INFO  |com.webdriver.WebDriverManager|INFO|Current google-chrome version is 101.0.4951
INFO  |com.webdriver.WebDriverManager|INFO|Get LATEST chromedriver version for 101.0.4951 google-chrome
INFO  |com.webdriver.WebDriverManager|INFO|Driver [C:\Users\nada\wdm\drivers\chromedriver\win32\101.0.4951.41\chromedriver.exe] found in cache
----- Captured log call -----
INFO  |root|test_ajouterMSG.py|INFO|TEST ADD MESSAGE *****
INFO  |root|test_ajouterMSG.py|INFO|PROVIDING MESSAGE INFORMATION *****
INFO  |root|test_ajouterMSG.py|INFO|START TEST ADD MESSAGE *****
ERROR |root|test_ajouterMSG.py|INFO|ADD MESSAGE TEST FAILED *****
----- warnings summary -----
test_login.py::Test_001_login::test_login
test_2AjouterEnfant.py::test_addKid::test_ajouterEnfant
test_3AjouterMembre.py::test_addMember::test_ajoutermembre
test_4AjouterJardin.py::test_addGarden::test_ajouteJardin
test_AjouterEnseignant.py::test_addTeacher::test_ajouterEnseignant

----- Captured log call -----
INFO  |root|test_ajouterMSG.py|INFO|TEST ADD MESSAGE *****
INFO  |root|test_ajouterMSG.py|INFO|PROVIDING MESSAGE INFORMATION *****
INFO  |root|test_ajouterMSG.py|INFO|START TEST ADD MESSAGE *****
ERROR |root|test_ajouterMSG.py|INFO|ADD MESSAGE TEST FAILED *****
----- warnings summary -----
test_login.py::Test_001_login::test_login
test_2AjouterEnfant.py::test_addKid::test_ajouterEnfant
test_3AjouterMembre.py::test_addMember::test_ajoutermembre
test_4AjouterJardin.py::test_addGarden::test_ajouteJardin
test_AjouterEnseignant.py::test_addTeacher::test_ajouterEnseignant

```

FIGURE 5.69 – Interface de console Jenkins après l'exécution des tests du Release3

Après avoir établi l'intégration avec Jenkins de tous les tests, un rapport html sera généré automatiquement comme l'indique la figure 5.70 .

| Environment  |   |
|--------------|---|
| JAVA_HOME    | C:\Program Files\Java\jdk-11.0.15   |
| PROJECT_NAME | CHOSA   |
| Packages     | ("pluggy": "1.0.0", "py": "1.11.0", "pytest": "7.1.2")                      |
| Platform     | Windows-10-10.0.19041-SP0   |
| Plugins      | ("forked": "1.4.0", "html": "3.1.1", "metadata": "2.0.1", "xdist": "2.5.0") |
| Python       | 3.7.0   |
| TESTERS      | NADA & RANA   |

| Summary   |  |
|---|--|
| 6 tests ran in 431.33 seconds.  |  |
| (Un)check the boxes to filter the results.  |  |
| <input checked="" type="checkbox"/> 1 passed, <input type="checkbox"/> 0 skipped, <input checked="" type="checkbox"/> 5 failed, <input type="checkbox"/> 0 errors, <input type="checkbox"/> 0 expected failures, <input type="checkbox"/> 0 unexpected passes |  |

| Results   |  |          |       |
|---|--|----------|-------|
| <a href="#">Show all details</a> / <a href="#">Hide all details</a> |  |          |       |
| Result  | Test   | Duration | Links |
| Passed (show details)   | test_login.py::Test_001_login::test_login                          | 85.58    |       |
| Failed (show details)   | test_2AjouterEnfant.py::Test_addKid::test_ajouterEnfant            | 76.80    |       |
| Failed (show details)   | test_3AjouterMembre.py::test_addMember::test_ajoutermembre         | 65.66    |       |
| Failed (show details)   | test_4AjouterJardin.py::test_addGarden::test_ajouteJardin          | 100.22   |       |
| Failed (show details)   | test_AjouterEnseignant.py::test_addTeacher::test_ajouterEnseignant | 77.92    |       |
| Failed (show details)   | test_AjouterMSG.py::test_addMember::test_ajoutermembre             | 23.78    |       |

FIGURE 5.70 – Interface de rapport html de tous les tests

## 5.4 Conclusion

Ce chapitre présente les tests manuels et automatiques de l'espace jardin effectués. Nous avons montré les interfaces des tests manuels avec Xray. Les interfaces des résultats des tests automatiques sont affichés sur la console Jenkins.

# Conclusion Générale

Au terme de ce rapport, nous pouvons conclure que ce stage de fin d'étude nous a donné une occasion opportune qui permet de confronter l'acquis théorique à l'environnement pratique.

En effet, le stage nous a permis de prendre certaines responsabilités, ce qui mène à consolider de plus en plus nos connaissances pratiques et théoriques.

C'est là où se trouve la valeur de ce projet qui combine les exigences de la vie professionnelle aux cotés bénéfiques de l'enseignement pratique que nous avons eu tout au long de notre parcours universitaire. Ce travail de conception et de développement d'un framework d'automatisation de test nous a permis de perfectionner nos connaissances acquises en ISTQB. Du point de vue technique, ce projet nous a permis de s'adapter avec l'automatisation des tests, aussi il nous a permis de maîtriser la méthodologie SCRUM.

Et comme tout projet, dans la phase de l'élaboration, nous avons rencontré quelques difficultés au niveau de l'intégration des tests avec Jenkins et nous avons pu les surpasser afin de présenter un telle travail.

finalement, notre prochaine objectif est d'améliorer notre framework afin qu'il nous permet d'exécuter les tests fonctionnels sur n'importe quelle l'application.

# Bibliographie

- [a] <https://www.allence-tunisie.com/services/>. Société Allence Tunisie.
- [b] <https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale>. Définition de SCRUM.
- [c] [https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique/](https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique) : :text=Python Définition de specification des besoins.
- [d] <https://www.advaloris.ch/nos-services/gestion-de-projet/definir-besoins-fonctionnels-gestion-de-projet> : :text=Le Définition des besoins fonctionnels.
- [e] [https://www.memoireonline.com/02/09/1973/m\\_conception \\_ \\_ et \\_ \\_ developpement \\_ dune \\_ application \\_ de \\_ la \\_ gestion \\_ dune \\_ bibliothque5.html](https://www.memoireonline.com/02/09/1973/m_conception _ _ et _ _ developpement _ dune _ application _ de _ la _ gestion _ dune _ bibliothque5.html). *Définition des besoins non fonctionnels*.
- [f] <https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale/1443820-jenkins-logiciel-star-de-l-integration-continue-open-source-gratuit/> : :text=D Définition des outils de test.
- [g] [https://fr.wikipedia.org/wiki/Selenium\\_\(informatique\)](https://fr.wikipedia.org/wiki/Selenium_(informatique)). *Définition de Selenium*.
- [h] <https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique/python-definition>. Définition de Python.
- [i] <https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale/1443820-jenkins-logiciel-star-de-l-integration-continue-open-source-gratuit/>. Définition de Jenkins.
- [j] <https://www.appvizer.fr/services-informatiques/outils-de-gestion-de-tests-logiciels>. Définition des outils de gestion de tests.
- [k] <https://www.cadremploi.fr/editorial/conseils/conseils-carriere/detail/article/jose-pas-demander-mais-quest-ce-que-jira.html>. Définition de Xray.
- [l] <https://www.syloë.com/glossaire/gitlab-ci/> : :text=GitLab Définition de Gitlab.
- [m] <https://www.redhat.com/fr/topics/devops/what-is-ci-cd>. Définition de l'architecture CI/CD.
- [n] <https://www.synopsys.com/glossary/what-is-cicd.html>. Architecture CI/CD.
- [o] <https://www.ibm.com/docs/fr/rational-soft-arch/9.5?topic=diagrams-use-case>. Définition diagramme de cas d'utilisation.
- [p] <https://hubvisory.com/blog/qu-est-qu-un-backlog-comment-le-construire-et-le-gerer/>. Définition de Backlog produit.
- [q] <https://www.thesaurus.gouv.qc.ca/tag/terme.do?id=MDL416> : :text=D Définition de diagramme de classe.

- [r] <https://www.appvizer.fr/magazine/ressources-humaines/bien-etre-employes/environnement-travail>. Définition de l'environnement de travail.
- [s] <https://www.jetbrains.com/fr-fr/pycharm/>. Définition de l'environnement de travail.
- [t] <https://www.edureka.co/blog/selenium-using-python/>. Architecture de liaison python avec selenium.
- [u] <https://www.syloë.com/glossaire/authentification>. Définition de l'authentification.
- [v] <https://docwiki.embarcadero.com/RADStudio/Sydney/fr/Définition de diagramme de séquence>.