



ESTEBAN LE CALVEZ

Rapport de Stage

Alternance Qualité & Intégration

Postulant au titre de Concepteur Développeur Informatique

Réalisé par
LE CALVEZ Estéban

Sommaire

Remerciements.....	4
Introduction	5
Présentation de l'entreprise.....	6
Présentation de l'entreprise.....	6
Organigramme.....	6
Les différents produits de Kurmi Software	9
1 - Kurmi Unified Provisionning.....	9
2 - Kurmi unified selfcare	10
3 - Kurmi Migration Tool.....	11
Technologies utilisées.....	12
Eclipse IDE	12
Win SCP	12
Visual Studio Code	13
SourceTree	13
Microsoft Teams.....	14
BugZilla.....	14
Trello.....	15
Squash.....	15
Selenium	17
Projet.....	18
Analyse de l'existant.....	18
Objectif du projet	18
Fonctionnalités de Selenium	19
Utilisation de Selenium chez Kurmi.....	20
Pattern adopté	20

Les tests avec Selenium.....	22
Les tests sans Sélénium.....	24
Les différents tests exécutés	25
Formulaire React.....	25
Tests	27
Les tests sur les super administrateurs.....	34
Création	35
Connexion de l'utilisateur créé.....	38
Modification.....	39
Suppression.....	47
Les tests sur les divisions	51
Création	52
Conclusion	57
Bilan	57
Problèmes rencontrés.....	58
Remerciements	59
Annexes.....	60
Consultation des divisions	60
TENANT	61
KURMISERVER	61
TENANTTEMPLATE	62
Formulaire React.....	63
Kurmi Unified Provisionning.....	64
Kurmi Unified Selfcare.....	65
Kurmi Migration Tool	67

Remerciements

Je tiens à remercier toute l'équipe Rennes de **Kurmi Software** de m'avoir si bien accueilli dans leurs locaux, de m'avoir apporté leur aide quand j'en avais besoin et d'avoir toujours apporté une bonne ambiance et une envie d'aller travailler.

Je remercie **Gilles Nouais** de m'avoir recruté et aidé dans mon projet.

Je remercie mes **collègues testeurs** pour m'avoir apporté des réponses et des solutions à mes problèmes.

Je remercie l'équipe développeurs, particulièrement **Sébastien Deprez** pour m'avoir appris à utiliser Sélénium et m'avoir aidé quand j'avais des questions.

Je remercie aussi l'ensemble de MyDigitalSchool de m'avoir permis d'apprendre différentes technologies pour développer tout au long de l'année.

Introduction

J'ai rejoint **Kurmi Software**, société de communications unifiées basée à Cesson Sévigné, le vendredi 05 janvier 2018 en tant que Testeur logiciel/Intégrateur de tests automatisés.

Je m'occupais principalement de développer des tests automatisés qui étaient déployés à chaque nouvelle version afin de tester les composants d'interface de la suite de produits **Kurmi**.

Au cours des missions effectuées, j'ai dû apprendre à communiquer, poser des questions quand je n'y arrivais pas.

Apprendre à travailler efficacement pour effectuer le travail demandé dans les temps. Et surtout apprendre un métier pour pouvoir comprendre l'application. J'ai aussi dû effectuer des recherches sur internet et plus particulièrement sur **StackOverflow** qui est un (sinon le plus complet) forum de développement.

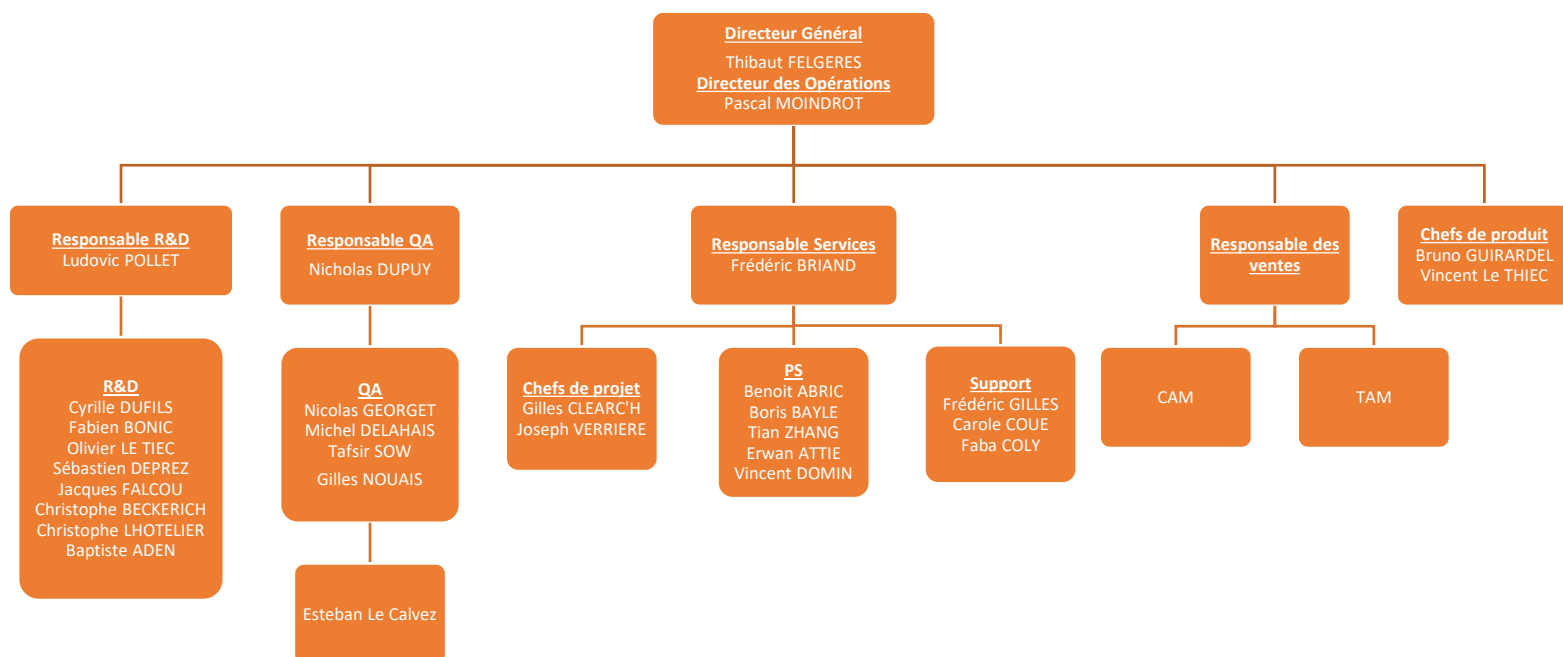
Présentation de l'entreprise

Présentation de l'entreprise

Kurmi-Software implanté depuis 2011 en tant qu'éditeur de logiciel spécialisé dans les communications unifiées est une filiale de Niji. L'entreprise est indépendante et autonome vis-à-vis de Niji.

L'entreprise compte à ce jour environ 35 salariés répartis sur 2 pôles et sites distincts. La première équipe est celle des commerciaux basés à Paris et ils ont pour but de trouver de nouveaux clients pour Kurmi. Le second pôle situé à Cesson-Sévigné est dédié à la Recherche et Développement de la suite logiciel Kurmi.

Organigramme



Ce pôle est constitué de plusieurs équipes.

L'équipe R&D (Recherche & Développement) développant les nouvelles fonctionnalités demandées par les clients ou suggérées par une des équipes et trouvé utile par les autres équipes.

L'équipe QA (Quality Assurance) dont je fais partie, va se charger de tester les nouvelles fonctionnalités en écrivant au préalable des cahiers de test et l'exécuter afin de voir s'il contient des bugs, auquel cas il ouvrira BugZilla pour créer un bug en indiquant toutes les informations nécessaires à l'équipe développement.

L'équipe PS (Professional Services) quant à elle, s'occupe de l'intégration de Kurmi chez le client par l'installation mais également la configuration. Chaque client a des demandes différentes et c'est à l'équipe PS d'en répondre en faisant une configuration adaptée.

L'équipe Support va s'occuper des problèmes techniques rencontré chez le client et répondre à leurs questions. Ils fonctionnent avec des tickets au préalable ouvert par le client, relatant le/les problèmes leur faisant face (dysfonctionnement, bloquant, etc.). A la suite de cela, l'équipe Support va dans le cas d'un bug le remonter à l'équipe R&D, pour dans la suite apporter une correction. Dans les autres cas, ils sont dans la capacité d'aider le client afin de résoudre le problème, car ils sont avant tout des experts en téléphonie et connaissent très bien le produit Kurmi et les configurations du client.

De ce qui est du management, il y a les 2 chefs de projets en contact avec les entreprises clientes pour connaître les besoins et les attentes vis-à-vis de la suite logiciel Kurmi. Ainsi la tâche est attribuée à une personne du Professional Service ou du Support selon la demande. Chaque fin de semaine un mail est envoyé à tout le monde pour informer l'avancer des différents projets chez les entreprises clientes, les projets finis et ceux qui sont en attente. Différentes réunions sont organisées au sein des équipes pour parler des problèmes rencontrés et de clarifier certains points sur qui a fait quoi et comment. Frédéric Briand et Ludovic Pollet chapotent respectivement les équipes Support/PS et R&D.

Et il y a aussi toutes les 2 semaines (quand tout le monde est présent si possible) une réunion permettant à chacun d'expliquer ce sur quoi il/elle a avancé et d'ouvrir des suggestions pour la suite.

Les différents produits de Kurmi Software

*Cf Annexes pour un aperçu des différents produits Kurmi.

1 -Kurmi Unified Provisionning

Un provisioning plus rapide, un déprovisioning en un clic

Kurmi vous permet d'automatiser vos process de moves, adds, changes & deletes (MACD) sur vos services de Communications Unifiées, de Collaboration et de Centre de Contact. Créez des profils utilisateurs par site et/ou fonction, accédez à votre plan de numérotation en temps réel, connectez Kurmi à votre SIRH, votre solution d'ITSM ou vos Active Directories. En automatisant l'administration de votre infrastructure, vous gagnez du temps, de l'argent et réduisez les risques d'erreurs humaines.

Kurmi permet de réduire de plus de 80% le temps de provisioning d'un nouvel utilisateur. Le déprovisioning est encore plus rapide, il suffit d'un clic !

Simplifiez et déléguez

Permettez à vos experts en Communications Unifiées se concentrer sur des nouveaux projets et ne laissez plus les tickets de support s'accumuler. La simplicité d'utilisation de Kurmi vous permet de déléguer l'administration quotidienne à des profils non-experts comme des agents de helpdesk, des administrateurs de site ou du personnel RH.

Grâce à des fonctionnalités avancées de délégation, de gestion des droits et de suivi des logs, Kurmi vous fournit tous les outils pour mettre en place et contrôler les équipes qui gèrent au quotidien vos infrastructures de communication.

Multi-constructeurs et hybride

Vous souhaitez unifier l'administration d'un call manager Cisco sur site, d'une solution de webconférence Webex, d'une messagerie instantanée Microsoft et d'un centre de contact Cisco UCCX ? Kurmi offre un point de contrôle central pour administrer tous vos services de communication, dans un environnement hybride (« on-prem » et/ou cloud) et multi-constructeurs.

Vous pouvez simplement intégrer de nouveaux services (messagerie instantanée, softphones, collaboration...) ou migrer votre téléphonie sans avoir à changer ou démultiplier vos outils d'administration. Vous n'êtes plus verrouillé par un constructeur et vous pouvez donc choisir les services qui conviendront le mieux à vos utilisateurs finaux.

En simple : Kurmi Unified Provisionning permet de déployer un ensemble de clients avec différentes configurations selon les droits accordés.

2 - Kurmi unified selfcare

Un portail selfcare pour vos utilisateurs finaux

Déléguer certaines opérations à vos utilisateurs : activation et self-provisioning de services, gestion des renvois d'appels, reset de password et de code PIN, accès à distance au téléphone, etc... Vous pouvez personnaliser le portail avec votre charte graphique et définir finement le niveau d'autonomie de vos utilisateurs en fonction de vos politiques IT.

Si vous disposez déjà d'un portail self-service pour vos utilisateurs (développé en interne ou Service Now par exemple), toutes nos fonctionnalités sont disponibles via notre API Selfcare pour les intégrer sur vos interfaces existantes.

Une meilleure expérience utilisateur, moins de tickets pour votre support

Vos employés veulent désormais une expérience de support comparable à celle qu'ils connaissent avec leurs services grand public : attendre plusieurs jours pour activer une nouvelle application ou réinitialiser un mot de passe n'est plus acceptable.

Avec Kurmi Unified Selfcare, vous leur offrez une expérience utilisateur à laquelle ils sont habitués avec leurs applications personnelles, tout en réduisant la charge de votre équipe support sur des opérations d'administration à faible valeur ajoutée.

Une meilleure gestion des ressources

Avec Kurmi Unified Selfcare, vous offrez à vos utilisateurs le droit d'activer un service particulier (softphone, IM...) sur le Service Store en fonction de leur profil, mais vous leur laissez décider s'ils en ont besoin. Ainsi vous ne consommez plus des licences et des ressources sur des déploiement en masse de services, qui ne sont ensuite pas forcément utilisés.

En simple : Kurmi Unified Selfcare est l'interface que possèdera l'utilisateur final.

3 - Kurmi Migration Tool

Kurmi gère la migration de vos données

Enrichi par 10 ans d'activité sur des environnements hétérogènes, Kurmi Unified Provisioning peut se connecter sur vos serveurs Cisco, Avaya, Alcatel ou Microsoft pour découvrir votre parc et réaliser automatiquement l'inventaire de tous les numéros, postes téléphoniques et groupements déployés. Une fois l'import des ressources réalisé, il ne vous restera plus qu'à sélectionner le profil cible pour que votre utilisateur soit migré sur le nouvel environnement.

Une migration réalisée à l'aide de Kurmi Unified Provisioning, vous permet d'auditer vos ressources existantes, pour mieux uniformiser et transformer vos données lors du provisioning du nouvel environnement.

Migrez au rythme que vous souhaitez

L'ADN multi-constructeur de Kurmi Unified Provisioning vous permet à l'aide d'une même interface Web, d'administrer l'environnement que vous déployez. Vous pouvez également administrer l'environnement que vous quittez, vous laissant ainsi maître de votre planning.

Vos administrateurs peuvent migrer site par site, utilisateur par utilisateur et réagir à tout imprévu en adaptant les règles de migration tout au long de votre projet.

Reportez la configuration de vos collaborateurs

Épargnez-vous tout risque d'erreur grâce à la solution de migration Kurmi Unified Provisioning.

La puissante couche de customisation offerte par Kurmi Unified Provisioning vous permet de faire correspondre vos usages sur l'environnement source à son équivalence sur l'environnement cible. Vous serez ainsi en mesure de reporter les spécificités de chaque utilisateur sur le nouvel environnement, vous offrant assurément une migration plus douce.

En simple : Kurmi Migration Tool permet de migrer des configurations et/ou données d'un environnement à un autre.

Technologies utilisées



Eclipse IDE

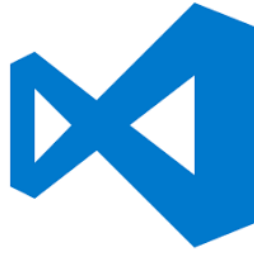
Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.



Win SCP

WinSCP est un client SFTP graphique pour Windows. Il utilise SSH et est open source. Le protocole SCP est également supporté. Le but de ce programme est de permettre la copie sécurisée de fichiers entre un ordinateur local et un ordinateur distant.

Dans mon cas : j'ai surtout utilisé Win SCP pour modifier les fichiers de clés de traduction et obtenir tous les logs quand je créais un bug.



Visual Studio Code

Visual Studio Code est un éditeur de code cross-platform open source gratuit.

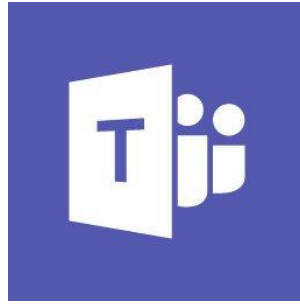
Il supporte de base une dizaine de langages mais possède des plugins pouvant en supporter plus.

Je l'ai utilisé pour développer des tests en javascript (ce sont ces tests qui étaient sur le serveur et exécutés à chaque sortie de version).



SourceTree

SourceTree, logiciel développé par le groupe Atlassian propose une interface graphique pour git et Mercurial desktop. Il permet de faire toutes les commandes de git en ayant une interface graphique.



Microsoft Teams

Microsoft teams est une plateforme qui contient différents espaces. Un espace de chat sur lequel nous pouvons discuter avec nos collègues, il donne la possibilité de télécharger des fichiers, des notes et des réunions.

Nous avons plusieurs « channels » pour différentes équipes, dans différents buts.



BugZilla

Bugzilla est une plateforme de déclaration de bugs sur une application ainsi que de communication autour de ce bug.

Nous déclarons une anomalie sur l'application, et discutons au sujet de cette dernière. Nous donnons les logs, l'environnement sur lequel ce bug a été trouvé, des images ou gif indiquant où se trouve le bug et une description de ce dernier.



Trello

Trello est un projet web basé sur la gestion d'applications.

Il répertorie différents tableaux sur lequel des cartes sont disposés afin de savoir qui travaille sur quoi.

Par exemple pour une équipe de développement il y aura plusieurs tableaux :

- A développer
- En cours de développement
- Développement fini
- (quand on trouve un bug) A corriger
- Correction terminée



Squash

Squash est une plateforme web de test permettant de déclarer :

- des exigences : on exige que tel partie de l'appli soit testée à tel endroit
- des cahiers de tests : un ensemble de pas de test à réaliser pour se dire qu'une exigence est réalisée.
- Des pas de tests : une ou plusieurs actions utilisateurs à faire pour parvenir à un résultat rapidement : Remplir les champs login et mot de passe et cliquer sur le bouton de connexion > Résultat : vous êtes connectés/vous avez un message d'erreur



Selenium

Selenium est un framework de test informatique développé en Java. Il permet d'automatiser des tests sur une application web en lançant un navigateur et en exécutant des tests qu'on lui ordonne d'exécuter.

On lui demande si le résultat qu'on attends est égal au résultat obtenu et on envoi tout ça dans les logs.

Projet

Analyse de l'existant

La mise en place de l'automatisation des tests au niveau graphique de l'application à l'aide de Selenium et Java, a été mis en place par Sébastien Deprez et un ancien stagiaire quelques temps avant que j'arrive.

Ils avaient mis en place un driver Selenium pour une version de Mozilla afin d'exécuter les tests automatiques sur l'interface de Kurmi.

Objectif du projet

L'objectif du projet est la reprise et l'amélioration de l'existant.

En effet le code qu'ils avaient commencé avant que j'arrive était historique, les bonnes manières avaient été réécrite et il fallait continuer de développer des tests automatisés.

Les objectifs étaient de :

- Augmenter la couverture de code
- Pouvoir s'assurer rapidement de la non régression de fonctionnalités.
- Être plus serein lors d'une sortie de version sur des éléments qu'ils n'auraient pas eu le temps de tester à la main

C'est pourquoi ils m'ont engagé pour développer des tests avec le framework Selenium afin d'augmenter la couverture de code.

Fonctionnalités de Selenium

Selenium est un framework que nous avons mis en place et utilisé avec Java.

Il propose des fonctionnalités qu'un utilisateur pourrait pratiquer sur l'interface graphique d'une application afin de s'assurer du bon fonctionnement de ces dernières.

Dans mon utilisation de Selenium, il y avait aussi des méthodes créées par mes collègues auparavant. Voici une liste non exhaustive:

- Des méthodes cherchant des éléments de la page (dans le DOM) par différentes manières :
 - css selector/id/ name /class (que j'ai le plus utilisé)
 - XPath
 - Element (div, a , p etc.)
 - Attribute
 - Child (quand plusieurs enfants dans un élément parent)
 - Index (dans un tableau)
 - Content (contenu dans l'élément)
- Des méthodes permettant de cliquer sur des éléments de la page :
 - Click() //Clique simplement sur l'élément
- Des méthodes permettant de s'assurer que des éléments sont présents sur la page (qu'ils soient cachés de l'utilisateur ou non)
 - IsPresent() //Si présent sur la page mais pas forcément visible
 - isDisplayed() // Si présent et visible par l'utilisateur à l'heure où la fonction est appelée
- Des méthodes permettant de remplir des champs (input)
 - sendKeys() // Permettant d'envoyer une String quelque part (en sélectionnant l'élément)
// Auparavant

Utilisation de Selenium chez Kurmi

Pattern adopté

Le pattern que nous avons adopté est le pattern de Page Object.

Le pattern Page Object est un pattern compréhensible par tous qui présente une application en différentes pages, qui comprennent chacune différents composants, qui comprennent chacun différents éléments web

Les principaux éléments du pattern Page Object sont donc :

- Les PageObject
- Les ComponentObject
- Les WebElements .

Exemple concret :

The screenshot shows the Kurmi web application. The header includes the Kurmi logo, a 'Substitution' dropdown, a search bar, and a user profile for 'Admin ADMIN'. The sidebar on the left lists navigation options: Home, Tenants Management, Tenants (highlighted), Tenant Templates, Quick Tenant, Scenarios, Organization, Configuration, Design, Report, Supervision, and Bulk Management. The main content area is titled 'TENANTS' and features a table with columns for Identifier, Description, and a status field. The table lists various tenants, including 'bugmail20965' with a status of 'provisoire'.

Identifier	Description	Status
BBB		
bugmail20965	provisoire	
CCC		
ELCTenan		
GNS		
MAJ		
NGETenant		
teantmigrate		
tenantForCustomScenarioTitle		
testDefault		
testScenario	Tenant Michel	
tso_bestone		

Ici nous avons une page recensant les Tenants (Clients) d'une plateforme interne à Kurmi.

 Composants

 Eléments

Les composants sont donc un ensemble d'éléments.

C'est alors à nous de définir qui sera composant et qui sera élément.

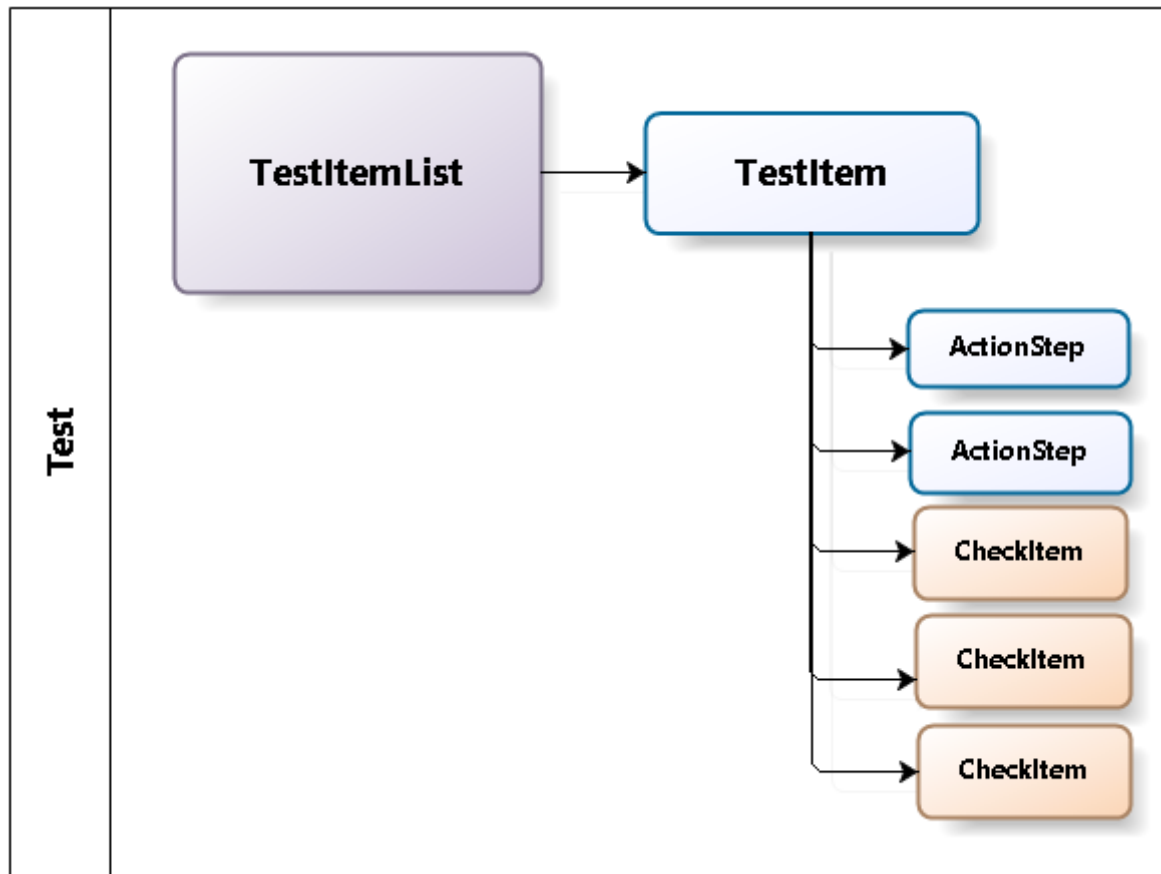
Dans cette page, nous avons défini le menu de gauche comme un Composant et chaque menu/sous menu comme un élément.

Nous avons défini la liste des tenants comme un Composant et en-tête, chaque checkbox, chaque champ du tableau comme des éléments.

La barre de bouton est un composant, chaque bouton est un élément.

Les tests avec Selenium

Chez Kurmi, pour tester l'application, nous avons décidé d'adopter le pattern de test suivant :



Nous déclarons une liste de tests que l'on nomme TestItemList.

Cette liste contiendra plusieurs TestItem.

Ces test items contiendront :

- Une TestResultList qui est une liste de résultats
- actionStep : plusieurs actions exécutées par Selenium
- CheckItem : qui viendront vérifier si les action tests executés on fait ce qu'on attendait d'elles.

Un exemple avec du code que j'ai développé :

```
//On déclare testList
List<AbstractTestItem> testList = new ArrayList<>();

//On y ajoute les TestItem en passant les données demandées dans le constructeur de ce test.
// Ici nous utilisons un enum pour lister les menus (MenuMapping) et nous cliquons sur le bouton qui mènera
// à la liste des divisions
testList.add(new GoToMenuAndCheckTitleTestItem(
    "Initialisation du test", MenuMapping.LIST_DIVISIONS,
    EXPECTED_TITLE_DIVISIONS_LIST));

testList.add(new DivisionsPage(
    "KUP_GUI_Administrator_02_02-01 - Divisions Page")
);
```

On va ensuite dans le testItem DivisionsPage :

```
//On cree la liste de resultat TestResultList
TestResultList resultList = new TestResultList(getId());

//On y ajoute ensuite des tests
//Ici nous vérifions que le bouton Create est affiché.
resultList.add(check(listDivisionPage.isActionCreateDisplayed(),
    MESSAGE_CREATE, listDivisionPage, true));

//Ici nous verifions que le bouton Delete est affiché.
resultList.add(check(listDivisionPage.isActionDeleteDisplayed(),
    MESSAGE_DELETE, listDivisionPage, true));

[...]
//On retourne cette liste pour obtenir en logs des chaines de chaque test
//et voir quels tests ont été exécutés avec succès (résultat attendu bon)
// et quels tests non.
return resultList.generateGlobalResult(listDivisionPage);
```

Les tests sans Sélénium

Ma mission était de développer des tests en Sélénium sur des parties de l'application que les autres testeurs n'avaient pas le temps de tester. Cependant j'ai aussi pu tester l'application à la main à plusieurs reprises.

Pour cela je suivais des cahiers de tests écrits par mes collègues testeurs afin de savoir quelle démarche suivre pour pouvoir tester une ou plusieurs fonctionnalités de l'application.

Par exemple, il y a récemment eu une feature développée par un collègue développeur qui permettait au créateur d'un scénario d'afficher un titre ou un autre sur les étapes du scénario selon les droits de l'utilisateur de consulter les scénarios ou non.

Pour cette Feature Request en particulier, il y a eu beaucoup d'aller-retour entre moi et le développeur en charge de cette dernière. En effet des titres devaient s'afficher dans un certain ordre, si l'étape était en erreur ou non et que l'utilisateur avait ou non les droits d'accéder à cette étape.

Certains titres n'étaient pas affichés au bon moment ou même pas du tout affichés, ou affichés quand ils ne devaient pas.

Les différents tests exécutés

Pour chaque test exécuté, je devais réfléchir à toutes les possibilités possibles.

Par exemple lors d'une création, je devais savoir quels champs n'étaient pas censés être corrects en les entrant (login trop long, email incorrect etc.).

Je vais donc lister ici l'ensemble des tests que j'ai développé par ensemble de page.

Formulaire React

Pour donner suite à une mise à jour de l'interface perpétrée par les développeurs de l'entreprise, nous avons décidé de mettre peu à peu en place React pour rendre l'application beaucoup plus fluide et intuitive, rendant l'expérience utilisateur meilleure.

Pour cela, les développeurs ont commencé par faire en sorte que sur les pages possédant des formulaires, il n'y ait plus de rechargement quand avait lieu une/plusieurs requête(s) en base.

En effet, l'application n'était pas du tout asynchrone, et chaque requête en base entraînait un rechargement complet de toute la page. Grâce à React, ces rechargements ont de moins en moins lieu. Maintenant toutes les requêtes en base se font de manière asynchrone et sans rechargement complet de page. Pour remplacer ce rechargement, il y a maintenant une barre de chargement en haut de la page, et parfois une barre de chargement ronde à côté du champ en question.

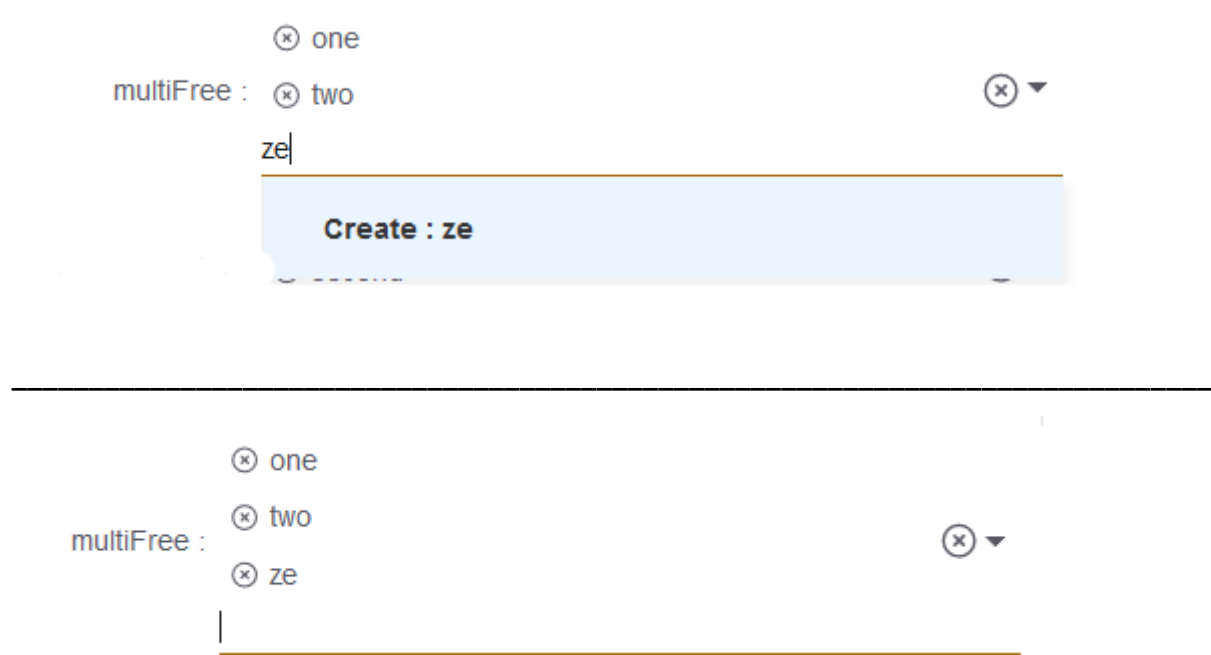
Ce changement dans la structure de l'application pouvant entraîner de nombreux bugs, un développeur a créé une page avec à peu près tous les champs possible modifiés par l'implémentation de react dans Kurmi. Cette page est une quick feature que l'on peut intégrer ou non très facilement dans l'application. Cette page étant uniquement destinée aux tests, elle n'a aucun besoin d'être en production chez les clients.

Le test de ce formulaire est le dernier test que j'ai fait à ce jour, et est encore en cours.

Pour ce test, j'ai dû créer de nouveaux composants dans l'application Kurmi, pour pouvoir les tester plus facilement par la suite, sachant que ces composants ont toujours la même classe CSS, il est très facile de les repérer et les réutiliser dans toute l'application.

Prenons l'exemple de deux composants qui sont très similaires mais cependant ont leur différence.

Un composant qui expose un inputText où l'on peut entrer une chaîne de caractère pour qu'il s'ajoute à cette liste :



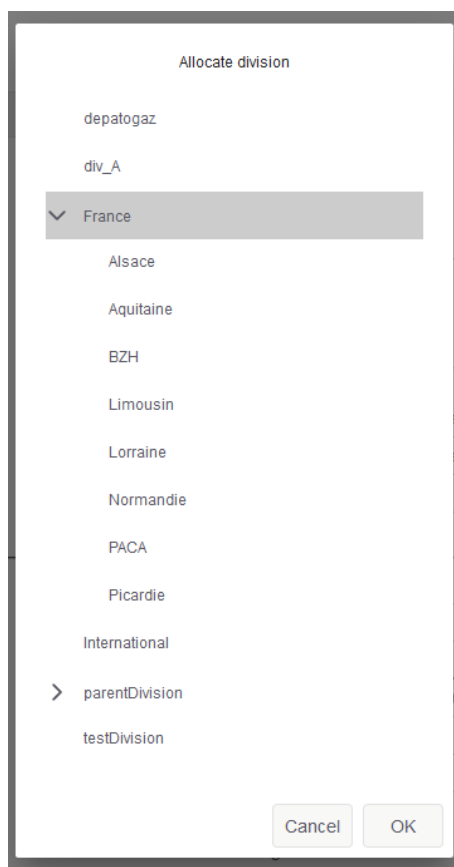
Il y a deux composants permettant cette interaction, un des deux à une fonctionnalité en plus, permettant à l'utilisateur d'ordonner cette liste, en déplaçant ces composants, vers le haut ou le bas.

Tests

La première étape est de se rendre sur la page de Quick Feature pour commencer le test. Pour cela, comme pour les autres tests, on utilise un MenuMapping avec l'identifiant CSS de ce menu pour qu'il aille cliquer dessus et vérifier le titre de la page pour être sûr qu'on soit sur la bonne page.

On appelle ensuite un seul testItem (pour le moment) qui ira tester tous les champs de la page. Cf : Annexe Formulaire React.

Cependant, avec cette mise à jour de React, les composants ont changé de sélecteurs CSS, c'est pourquoi je suis resté bloqué sur le champ allocate division, qui fait apparaître un Modal permettant de choisir parmi les divisions existantes.



J'ai remarqué que sur cette page, il y avait deux composants globaux : General Information et Scheduling, j'ai donc créé le premier composant (sachant que l'autre composant est utilisé sur de nombreuses autres pages, et est le même).

Dans le premier composant, il y a un nombre de getters privés, uniquement utilisés dans cette classe. Et des méthodes renvoyant des types connus, booléens, chaînes de caractères, nombres.

Par exemple, `numberOfSwitchTest()` renvoie un nombre qui est le nombre de champs en dessous de la checkbox `switchTest`. Le nombre de champs est à 2 quand la checkbox n'est pas cochée, et est à 4 quand la checkbox est cochée.

<code>switchTest :</code>	<input checked="" type="checkbox"/>
<code>switchTestTrue :</code>	<code>switchTestTrue</code>
<code>switchTestTrue2 :</code>	<code>switchTestTrue2</code>
<code>switchTestTrue3 :</code>	<code>switchTestTrue3</code>
<code>switchTestTrue4 :</code>	<code>switchTestTrue4</code>

Dans le cas ci-dessus, la méthode renverra 4.

En regardant chaque élément de la page, je commence par faire le getter privé qui renverra le ou les éléments désirés en fonction du sélecteur css désigné.

```
private WebElement getMultiFree() {  
    return nav.driver.findElement(By.cssSelector(".fieldId_input_multiFree .field-value-table"));  
}
```

Nous décidons de laisser les sélecteurs privés car ils renvoient un `WebElement` qui doit rester caché du reste de l'exécution. En effet le `WebElement` est un type propre à Sélénium, c'est pourquoi nous utilisons ce `WebElement` pour renvoyer des types connus de Java (`String`, `boolean`, `int`) afin de garder une certaine flexibilité dans le cas où nous changerions de framework de tests automatisés.

On regarde ensuite toutes les fonctionnalités d'un champ afin de créer les méthodes qui pourront être utilisés dans le reste des tests.

Prenons pour exemple le champ `multiFree`, qui permet d'ajouter des objets dans la barre et de les ajouter après (là où se trouvent `one` et `two`) puis de les supprimer individuellement (avec les croix à côté de chaque champ) ou bien tous d'un coup (avec la croix à droite).

⊗ first
multiFreeOrder : ⊗ second



On crée donc des méthodes permettant d'utiliser ce champ.

-boolean removeAllFromMultiFreeOrder() //Ira cliquer sur la croix à droite pour tout supprimer et comparera le nombre d'éléments avant et après suppression, en attendant un de moins après suppression.

-boolean removeMultiFreeOrder(String chaine) //Ira cliquer sur la croix à gauche du champs passé en paramètre de la fonction et regardera le nombre d'éléments présents dans la liste, en s'attendant à en avoir 0

-boolean addMultiFreeOrder(String chaine) //Ira écrire dans la barre de texte et appuyer sur entrer pour valider la saisie du champs et comparera le nombre d'élément avant et après en s'attendant à en avoir un de plus après l'ajout.

-int nombreOfOptionsIntoMultiFreeOrder() //Renvoie le nombre de champs dans cette liste.

Chaque champ aura ces méthodes permettant de tester si le comportement est bon ou non.

```
/** Champ multiFreeOrder */  
resultList.add(check(generalInfo.isMultiFreeOrderDisplayed(), "[ Component multiFreeOrder is display ] ", qfReact, true));  
resultList.add(check(generalInfo.addMultiFreeOrder("patate"), "[ add multiFree ] ", qfReact, true));  
resultList.add(check(!generalInfo.addMultiFreeOrder("patate"), "[ No add multiFree because of already existing ] ", qfReact, true));
```

On commence donc par tester le champ multiFreeOrder, en vérifiant qu'il est bien affiché.

Ensuite on ajoute un nouveau champ à l'intérieur en utilisant la méthode addMultiFreeOrder décrite au-dessus.

Après avoir ajouté le champs « patate » on essaie de le rajouter une seconde fois, ce qui ne va pas marcher, on ne peut avoir qu'une seule fois le même champ. On vérifiera donc que la méthode addMultiFreeOrder renvoi cette fois un booléen faux.

On test ensuite le champ multiFree

```
/** champ multiFree */  
resultList.add(check(generalInfo.isMultiFreeDisplayed(), "[ Component multiFree is display ] ", qfReact, true));  
  
resultList.add(check(generalInfo.addMultiFree("patate"), "[ add multiFree ] ", qfReact, true));  
resultList.add(check(!generalInfo.addMultiFree("patate"), "[ No add multiFree because of already existing ] ", qfReact, true));  
resultList.add(check(generalInfo.addMultiFree("oui"), "[ add multiFree ] ", qfReact, true));  
resultList.add(check(generalInfo.addMultiFree("non"), "[ add multiFree ] ", qfReact, true));  
  
resultList.add(check(generalInfo.removeMultiFree("patate"), "[ remove multiFree ] ", qfReact, true));
```

Pour ce champ, nous allons tester qu'il est bien affiché, de la même façon que l'ancien champ, testé d'ajouté une fois « patate » avec succès puis une seconde fois « patate » sans succès.

Nous ajouterons cette fois-ci deux champs de plus, oui et non.

Nous essaierons ensuite de supprimer le champ « patate » en utilisant la méthode `removeMultiFree(String chaine)` décrite au-dessus.

On test ensuite la checkbox et le nombre de champs générés automatiquement en la cochant/décochant.

```
/** SwitthTest et champs générés automatiquement. */
resultList.add(check(!generalInfo.isSwitthTestChecked(), "[ Component Switth Checkbox is unchecked ] ", qfReact, true));
resultList.add(check(generalInfo.numberOfSwitthTest() == 2, "Switth checkbox unchecked . number expected of input :2. result : " + generalInfo.numberOfSwitthTest(), qfReact, true));

/** Checking checkbox */
generalInfo.checkSwitthTest();

resultList.add(check(generalInfo.isSwitthTestChecked(), " [ Component SwitthTest is checked ]", qfReact, true));
resultList.add(check(generalInfo.numberOfSwitthTest() == 4, "Switth checkbox checked. number expected of input :4. result : " + generalInfo.numberOfSwitthTest(), qfReact, true));
```

De base, la checkbox est décochée, nous vérifions donc qu'il y a 2 éléments affichés en dessous.

Nous cochons ensuite cette checkbox et vérifions que le nombre d'éléments affichés en dessous de celle-ci est maintenant passé à 4.

Le prochain champ à tester est le champ « A », celui-ci est un `inputText` dans lequel nous allons passer une chaîne de caractère qui sera reportée dans le titre du champs en dessous.

A : Titre ici|



value de A: Titre ici

```
/** A */
resultList.add(check(generalInfo.sendKeysToA("Patate."), "Is value of A title contain the String: " + generalInfo.getValueOfA().getText() + " ", qfReact, true));
```

La méthode `sendKeysToA(String chaine)` va envoyer la chaîne de caractère passé en paramètre dans le champs « A » puis va aussi vérifier que la valeur du champ « value de A :XXX » contient bien la chaîne de caractère passée dans le champs A, puis renverra un booléen.

Le prochain champ « B » qui quand il est vide ne crée pas de second champ. Et qui quand on le remplit crée un second champ « depends B » qui va contenir la même chaîne de caractère que « B ».

B* : Titre ici

dependsB : Titre ici

```
/** B */
resultList.add(check(!generalInfo.isDependsBDisplayed(), "Check that dependsB isn't displayed", qfReact, true));

generalInfo.sendKeysToB("Kurmi");
resultList.add(check(generalInfo.isDependsBCorrectlyDisplayed(),
    "Check that dependsB is displayed and contains same text as B. \n Is dependsB displayed ? : "
    + generalInfo.isDependsBDisplayed() +
    "\n B text : " + generalInfo.valueOfB() +
    "\n dependsB text : "+ generalInfo.valueOfDependsB() , qfReact, true));
```

Nous allons donc regarder si ce champ est affiché sur la page, puis lui passer la chaîne Kurmi, grâce à la méthode isDependsBCorrectlyDisplayed() qui permettra de vérifier si le champ « depends B » est affiché et vérifier que la chaîne de caractère passée dans « B » est la même que celle qui apparaîtra dans « depends B ».

Le prochain champ à tester est celui du ChargeMePossibleValNb, un champ dans lequel nous allons passer un nombre, afin qu'il génère ce nombre d'éléments en dessous.

chargeMePossibleValNb : 5

C0 :	Select...	▼
C1 :	Select...	▼
C2 :	Select...	▼
C3 :	Select...	▼
C4 :	Select...	▼

Cette liste d'éléments générés ira chercher un json contenant des noms de personnes.

C0 : Select...

C1 : Noreen Mccarthy

C2 : Gladys Guthrie

C3 : Renee Lowe

C4 : Kelly Odonnell

C4 : Hunter Carpenter

sion : May Bray

Dans les champs CX nous pouvons choisir plusieurs noms, s'il y a 2 noms ou plus de sélectionner dans un C, il va créer un champ D contenant une liste de champs égale aux noms de personnes entrés dans le champ C.

⊗ Gladys Guthrie

C0 : ⊗ May Bray

D0 : Gladys Guthrie

Gladys Guthrie

May Bray

Dans cette liste, nous ne pourrions sélectionner qu'un seul élément.

```
/** ChargeMePossibleValNb */
generalInfo.insertNumberOfChargeMePossibleValNb("3");
resultlist.add(check(generalInfo.numberOfD() == 0, "Number of generated D expected : 0, actual : "+generalInfo.numberOfD(), qfReact, true));
resultlist.add(check(generalInfo.numberOfC() == 3, "Number of generated C expected : 3, actual : "+generalInfo.numberOfC(), qfReact, true));
resultlist.add(check(generalInfo.selectOptionsC(1, "Noreen"), "Adding Noreen to C1 ", qfReact, true));
resultlist.add(check(generalInfo.selectOptionsC(1, "Gladys"), "Adding Gladys to C1 ", qfReact, true));
resultlist.add(check(generalInfo.numberOfD() == 1, "Number of generated D expected : 1, actual : "+generalInfo.numberOfD(), qfReact, true));
```

J'ai donc testé de la manière suivante cet ensemble de champs.

J'ai commencé par insérer « 3 » dans le champ chargeMePossibleValNb afin de générer 3 champs.

J'ai ensuite vérifié qu'il n'y avait aucun sous champ « D » d'affiché, et qu'il y avait 3 champs « C » d'affiché, car nous avions entré 3.

J'ai ensuite sélectionné le champ C1 et lui ai passé deux paramètres pour qu'il aille cliquer dessus « Noreen » et « Gladys » .

Enfin, j'ai vérifié qu'un champ D était apparu sachant que nous avions entré 2 noms dans une liste « C »

Pour le champ « allocate Division » le test ne fonctionne pas encore (le sélecteur CSS est pour le moment incorrect) CF divisions.

Les tests sur les super administrateurs.

Sur cet ensemble de page, il y avait :

- Une liste avec tous les super administrateurs
- Une page de création
- Une page d'information
- Une page de modification
- 2 pages de suppressions (l'une depuis la liste en cochant une ou plusieurs checkbox, l'autre depuis la page d'information)

Sur chaque page je devais vérifier que tous les composants étaient présents.

J'ai donc commencé par la page de liste. Le menu de gauche et la bannière étant présents sur toutes les pages, je n'avais pas besoin de le vérifier (c'était vérifié à chaque test. Il s'agissait d'une PageObject parente contenant ces deux éléments.)

Chaque test commence de l'écran de connexion de l'application, et fait appel à un autre test qui va nous connecter en tant qu'administrateur.

Création

La première chose que j'avais à faire était de créer un jeu de test. Pour cela j'ai appelé deux fois une ActionStep de création de super administrateur en lui passant les paramètres que je souhaitais lui passer.

```
List<AbstractTestItem> testList = new ArrayList<>();
List<String> logins = new ArrayList<String>();
logins.add(params.loginSA);
logins.add(params.loginSA+"2");
logins.add("Patrick");
new CreateSuperAdministratorActionStep(params.lastName, params.firstName,
    params.email, params.language,
    params.prefferedSpreadSheetFormat, params.secureMode,
    params.sendEmailUser, params.keepSettings, logins.get(1),
    params.passwordSA, params.passwordConfirmationSA)
    .execute(nav, new LoggedPageObject(nav)).cast(ListSuperAdministratorPageObject.class);
new CreateSuperAdministratorActionStep(params.lastName, params.firstName,
    params.email, params.language,
    params.prefferedSpreadSheetFormat, params.secureMode,
    params.sendEmailUser, params.keepSettings, logins.get(2),
    params.passwordSA, params.passwordConfirmationSA)
    .execute(nav, new ListSuperAdministratorPageObject(nav));
```

Cette ActionStep :

- Clique sur le menu des super administrateurs
- Clique sur le bouton de création
- Remplit les différents champs du formulaire avec les paramètres passés
- Clique sur le bouton de création du super administrateur.
- Vérifie qu'on arrive bien sur la liste des super administrateurs (sinon, c'est que quelque chose s'est mal passé).

Après la création de ces 2 Super administrateurs, je pouvais commencer mes tests. J'ai donc commencé par vérifier que le nombre de super administrateurs présents sur cette page étaient corrects.

```
testList.add(new ListPage(
    "KUP_GUI_Administrators_02_01-01 - Super-Administrator Page",3));
```

Ici, j'attends 3 super administrateurs. (les deux créés plus celui de base)

Le test va ensuite cliquer sur le bouton de création et vérifier que le titre de la page sur laquelle on arrive est bon, et que tous les éléments sont présents.

```
testList.add(new ListPageCreateButton(
    "KUP_GUI_Administrator_02_01-02 - Super-Administrator - Create action"));
testList.add(new CreationPage(
    "KUP_GUI_Administrator_02_01-03 - Super-Administrator Creation Page",
    EXPECTED_TITLE_CREATION_SUPER_ADMIN));
```

```
resultList.add(check(
    creationPage.isCreateButtonDisplayed(),
    "[ Button Create displayed ]",
    creationPage,true));
```

Résultat en console quand tout est OK.

```
***** Sub Tests details for TestResultlist Creation page *****
DONE_OK : Creation page : [ Super Administrator Creation Page title must be 'SUPER-ADMINISTRATOR CREATION']
DONE_OK : Creation page : [ Button Create displayed ]
DONE_OK : Creation page : [ Button Cancel displayed ]
DONE_OK : Creation page : [ LastName field displayed ]
DONE_OK : Creation page : [ FirstName field displayed ]
DONE_OK : Creation page : [ Email field displayed ]
DONE_OK : Creation page : [ Language field displayed ]
DONE_OK : Creation page : [ PreferredSpreadSheetFormat field displayed ]
DONE_OK : Creation page : [ KeepSettings field displayed ]
DONE_OK : Creation page : [ SecureMode field displayed ]
DONE_OK : Creation page : [ SendUserEmail field displayed ]
DONE_OK : Creation page : [ Login field displayed ]
DONE_OK : Creation page : [ Password field displayed ]
DONE_OK : Creation page : [ PasswordConfirmation field displayed ]
*****
```

J'ai ensuite testé les messages d'erreurs dans le cas où l'utilisateur n'entrait pas des choses corrects dans les champs.

```
// Checks about creation
testList.add(new CreationFailed(
    "KUP_GUI_Administrator_02_01-04 - Super-Administrator Creation Page - Create : no login",
    params.lastName, params.firstName, params.email, params.language,
    params.preferredSpreadSheetFormat, params.secureMode,
    params.sendEmailUser, params.keepSettings, "",
    params.keepPassword, "", "",
    EXPECTED_MANDATORY_LOGIN));
```

On passe en paramètres les valeurs des champs qui vont être remplis dans le formulaire ainsi que le message d'erreur attendu. (EXPECTED_MANDATORY_LOGIN)

Les résultats en console sont les suivants.

```
DONE_OK :          CreationFailed :          [ At least one error message must be
: {The parameter 'Login of the administrator' is mandatory} result are : {The
parameter 'Login of the administrator' is mandatory} ]

DONE_OK :          CreationFailed :          [ At least one error message must be
: {The parameter 'Password of the administrator' is mandatory} result are : {The
parameter 'Password of the administrator' is mandatory} ]

DONE_OK :          CreationFailed :          [ At least one error message must be
: {The parameter 'Lastname of the administrator' cannot exceed 64 characters}
result are : {The parameter 'Lastname of the administrator' cannot exceed 64
characters} ]

DONE_OK :          CreationFailed :          [ At least one error message must be
: {The parameter 'Firstname of the administrator' cannot exceed 64 characters}
result are : {The parameter 'Firstname of the administrator' cannot exceed 64
characters} ]

DONE_OK :          CreationFailed :          [ At least one error message must be
: {The parameter 'E-mail of the administrator' is not in the correct format}
result are : {The parameter 'E-mail of the administrator' is not in the correct
format} ]

DONE_NOT_OK :      CreationFailed :          [ At least one error message
must be : {BDD_UNIQUE_CONSTRAINT_VIOLATION_NATES_ADMIN_USERID} result are : {The
value "admin" already exists in the database.} ]
```

La console renvoi :

-**DONE_OK** quand le message attendu est le bon.

-**DONE_NOT_OK** et arrête l'exécution du test quand le message attendu n'est pas le bon.
(c'est souvent le cas quand en local on a un message, mais que sur le serveur où seront exécutés les tests en a un différent)

Après avoir testé qu'on avait les bons messages d'erreur quand une création ne réussis pas ; on test que la création marche bien quand on rentre les bons paramètres.

```
testList.add(new CreationSuccess(
    "KUP_GUI_Administrator_02_01-04 - Super-Administrator Creation Page - Create : Success",
    params.lastName, params.firstName,
    params.email, params.language,
    params.prefferedSpreadSheetFormat, params.secureMode,
    params.sendEmailUser, params.keepSettings, logins.get(0),
    params.keepPassword, params.passwordSA, params.passwordConfirmationSA,
    EXPECTED_CREATIONSUCCESS));
```

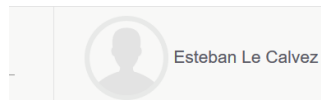
Ce test est sensiblement le même mais on arrivera sur la page avec la liste des super administrateurs et un message de succès nous disant que l'utilisateur est créé.

Connexion de l'utilisateur créé

Après avoir créé avec succès le super administrateur, on s'assure que ce n'est pas qu'un message vert nous disant qu'il est créé et on test de le connecter.

```
testList.add(new ConnectSuperAdmin(  
    "KUP_GUI_Administrator_02_01-04 - Super-Administrator Creation Page - Create : connect with the new admin",  
    params.loginSA,params.passwordSA,params.firstName, params.lastName));  
//Reconnexion de admin admin  
testList.add(new ConnectSuperAdmin(  
    "KUP_GUI_Administrator_02_01-04 - Super-Administrator Creation Page - Create : connect with the new admin",  
    "admin","admin","admin", "admin"));
```

Ce test commence sur la page de login et s'attend à arriver sur une page loguée de l'application. Il vérifiera dans la bannière si le nom et prénom du super administrateur sont corrects.



On reconnecte ensuite le super administrateur par défaut pour continuer les tests.

Modification

Une fois s'être assuré que la création des super administrateurs fonctionne bien, nous passons à la modification.

La première étape est de se rendre sur le super administrateur que nous voulons modifier afin de le modifier.

```
//Checks about modification
testList.add(new ListPageToConsultPage(
    "KUP_GUI_Administrator_02_01-11 - Super-Administrator - Link to - Consultation page",
    params.loginSA,
    EXPECTED_TITLE_CONSULT_SUPER_ADMIN));
```

On passe donc en paramètre le login du Super administrateur afin que le test clique dessus et vérifie l'intitulé de la page.

On clique ensuite sur le bouton de modification.

```
testList.add(new ConsultPageModifyButton(
    "KUP_GUI_Administrator_02_01-13 - Super-Administrator Information Page - Modify action",
    EXPECTED_TITLE_MODIFY_SUPER_ADMIN));
testList.add(new ModificationPage(
    "KUP_GUI_Administrator_02_01-14 - Super-Administrator Modification Page",
    EXPECTED_TITLE_MODIFY_SUPER_ADMIN));
testList.add(new ModificationPageCancelButton(
    "KUP_GUI_Administrator_02_01-16 - Super-Administrator Modification Page - Cancel",
    EXPECTED_TITLE_CONSULT_SUPER_ADMIN));
testList.add(new ConsultPageModifyButton(
    "KUP_GUI_Administrator_02_01-11 - Super-Administrator - Link to - Modification page",
    EXPECTED_TITLE_MODIFY_SUPER_ADMIN));
```

Ici on vérifie qu'on arrive bien sur une page de modification avec tous les champs présents, à la manière de la création.

Une fois sur la page de modification, on vérifie le bon fonctionnement du bouton Cancel en cliquant dessus et en vérifiant qu'on arrive sur la bonne page, puis on retourne sur la page de modification pour commencer les tests sur les modifications.

Sur ce test :

```
testList.add(new ModificationPage(
    "Modification page",
    EXPECTED_TITLE_MODIFY_SUPER_ADMIN));
testList.add(new ModificationPageCreateButton(
```

Nous vérifions la présence de tous les champs sur la page :

```
***** Sub Tests details for TestResultList Modification page *****
DONE_OK :      Modification page :      [ Super Administrator
Modification Page title must be 'SUPER-ADMINISTRATOR MODIFICATION']

DONE_OK :      Modification page :      [ Button Create displayed ]

DONE_OK :      Modification page :      [ Button Cancel displayed ]

DONE_OK :      Modification page :      [ LastName field displayed ]

DONE_OK :      Modification page :      [ FirstName field displayed ]

DONE_OK :      Modification page :      [ Email field displayed ]

DONE_OK :      Modification page :      [ Language field displayed ]

DONE_OK :      Modification page :      [ PrefferedSpreadSheetFormat
field displayed ]

DONE_OK :      Modification page :      [ KeepSettings field displayed
]

DONE_OK :      Modification page :      [ SecureMode field displayed ]

DONE_OK :      Modification page :      [ SendUserEmail field
displayed ]

DONE_OK :      Modification page :      [ Login field displayed ]

DONE_OK :      Modification page :      [ keepPassword checkbox
displayed ]
*****
```


A la manière de la création, on va ensuite vérifier qu'en entrant des choses incohérentes dans les champs du formulaire, on obtient des messages d'erreur.

The screenshot shows a web application interface for modifying a super-administrator. At the top, a red error banner states: "The parameter 'Lastname of the administrator' cannot exceed 64 characters". The page title is "SUPER-ADMINISTRATOR MODIFICATION". On the left is a sidebar menu with options: Home, Organization, Super-administrators (selected), Divisions, Providers, Providers Profiles, and Configuration. The main form area is titled "General information" and contains three input fields: "Lastname" (filled with 64 'a' characters), "Firstname" (filled with "superAdmin"), and "E-mail" (filled with "superadmin@superadmin.fr"). There are "Apply" and "Cancel" buttons at the top right of the form area.

Les résultats en console sont les suivants :

```
DONE_OK :      ModificationFailed :      [ At least one error message
must be : {The parameter 'Lastname of the administrator' cannot exceed 64
characters} result are : {The parameter 'Lastname of the administrator' cannot
exceed 64 characters} ]

DONE_OK :      ModificationFailed :      [ At least one error message
must be : {The parameter 'Firstname of the administrator' cannot exceed 64
characters} result are : {The parameter 'Firstname of the administrator' cannot
exceed 64 characters} ]

DONE_OK :      ModificationFailed :      [ At least one error message
must be : {The parameter 'E-mail of the administrator' is not in the correct
format} result are : {The parameter 'E-mail of the administrator' is not in the
correct format} ]

DONE_OK :      ModificationFailed :      [ At least one error message
must be : {The parameter 'E-mail of the administrator' cannot exceed 128
characters} result are : {The parameter 'E-mail of the administrator' cannot
exceed 128 characters} ]

DONE_OK :      ModificationFailed :      [ At least one error message
must be : {The parameter 'Password of the administrator' cannot exceed 30
characters} result are : {The parameter 'Password of the administrator' cannot
exceed 30 characters} ]

DONE_OK :      ModificationFailedNotMatchingPasswords :      Expected
alert message {Passwords do not match !}. Actual alert message : {Passwords do not
match !}

DONE_OK :      ModificationSuccess :      [ At least one error message
is expected : {The administrator has been updated successfully.} Result : {The
administrator has been updated successfully.} ]
```

Après avoir testé les messages d'erreur, on vérifie une bonne modification du super administrateur :

The screenshot displays the KURMI web interface. At the top, a green notification bar states "The administrator has been updated successfully." with a close button. Below this, the page title is "SUPER-ADMINISTRATOR INFORMATION". The left sidebar contains a menu with items: Home, Organization, Super-administrators (highlighted), Divisions, Providers, Providers Profiles, Configuration, Design, Report, Supervision, and Bulk Management. The main content area is divided into two sections: "General information" and "Identification". The "General information" section lists: Lastname: testModified (with a blue arrow pointing to it), Firstname: superAdminnn, E-mail: superadmin@superadmin.fr, Language *: English, Preferred spreadsheet format *: Excel (.xlsx), Secure mode *: Yes, Send emails to affected users when changes are performed *: No, and Keep my settings *: Yes. The "Identification" section lists: Login *: testSuperAdmin and Password *: *****. A "Back" button is located in the top right corner of the main content area.

✓ The administrator has been updated successfully. X

Substitution Search ... adminadmin

Home > SUPER-ADMINISTRATOR INFORMATION ✓ Back

Organization v

Super-administrators

Divisions

Providers

Providers Profiles

Configuration >

Design >

Report >

Supervision >

Bulk Management >

General information

LastName : testModified

Firstname : superAdminnn

E-mail : superadmin@superadmin.fr

Language * : English

Preferred spreadsheet format ? : Excel (.xlsx)

Secure mode ? : Yes

Send emails to affected users when changes are performed : No

Keep my settings ? : Yes

Identification

Login * : testSuperAdmin

Password * : *****

On vérifie aussi que les champs ont correctement été modifiés.

```
testList.add(new ConsultSuperAdminPage(  
    "Consult page",  
    EXPECTED_TITLE_SUPER_ADMIN_LIST,  
    EXPECTED_TITLE_CONSULT_SUPER_ADMIN,  
    "testModified", "superAdminnn",  
    params.email,  
    params.language,  
    params.prefferedSpreadSheetFormat,  
    params.secureMode,  
    params.sendEmailUser,  
    params.keepSettings,  
    params.loginSA,  
    params.passwordSA+"modified"));
```

Les résultats en console :

```
***** Sub Tests details for TestResultList Consult page  
  
DONE_OK :          Consult page :          [ Page title : Expected {SUPER-  
ADMINISTRATOR INFORMATION} : Result {SUPER-ADMINISTRATOR INFORMATION} ]  
  
DONE_OK :          Consult page :          [ Action modify must be present in  
the action menu bar ]  
  
DONE_OK :          Consult page :          [ Action Delete must be present in  
the action menu bar ]  
  
DONE_OK :          Consult page :          [ The First name field is incorrect :  
Expected {superAdminnn} : Result {superAdminnn} ]  
  
DONE_OK :          Consult page :          [ The Last name field is incorrect :  
Expected {testModified} : Result {testModified} ]  
  
DONE_OK :          Consult page :          [ The E-mail field is incorrect :  
Expected {superadmin@superadmin.fr} : Result {superadmin@superadmin.fr} ]  
  
DONE_OK :          Consult page :          [ The Language field is incorrect :  
Expected {English} : Result {English} ]  
  
DONE_OK :          Consult page :          [ The Preffered spread sheet format  
field is incorrect : Expected {Excel (.xlsx)} : Result {Excel (.xlsx)} ]
```

```

DONE_OK :          Consult page :          [ The Keep Settings field is
incorrect : Expected {Yes} : Result {Yes} ]

DONE_OK :          Consult page :          [ The Send user email field is
incorrect : Expected {No} : Result {No} ]

DONE_OK :          Consult page :          [ The Secure mode field is incorrect
: Expected {Yes} : Result {Yes} ]

DONE_OK :          Consult page :          [ The Login field is incorrect :
Expected {testSuperAdmin} : Result {testSuperAdmin} ]

DONE_OK :          Consult page :          [ The password field is incorrect :
Expected {*****} : Result {*****} ]

```

Ensuite, on test le lien de la page de consultation à la page de suppression, ainsi que le bouton cancel de la page de suppression.

```

testList.add(new ConsultPageDeleteButton(
    "KUP_GUI_Administrator_02_01-17 - Super-Administrator Information Page - Delete action",
    EXPECTED_TITLE_SUPER_ADMIN_CONSULT_DELETE));
testList.add(new DeleteFromConsultPageCancelButton(
    "KUP_GUI_Administrator_02_01-18 - Super-Administrator Information Page - Delete - Confirm removal page",
    EXPECTED_TITLE_CONSULT_SUPER_ADMIN));
testList.add(new ConsultSuperAdminPageReturnButton(
    "KUP_GUI_Administrator_02_01-19 - Super-Administrator Information Page - Back",
    EXPECTED_TITLE_SUPER_ADMIN_LIST));

```

Resultats en console.

```

***** Sub Tests details for TestResultList KUP_GUI_Administrator_02_01-17 -
Super-Administrator Information Page - Delete action *****

DONE_OK :          KUP_GUI_Administrator_02_01-17 - Super-Administrator
Information Page - Delete action :          [ A cancel button must be displayed ]

DONE_OK :          KUP_GUI_Administrator_02_01-17 - Super-Administrator
Information Page - Delete action :          [ A delete button must be displayed ]

DONE_OK :          KUP_GUI_Administrator_02_01-17 - Super-Administrator
Information Page - Delete action :          [ Page title must be : Expected
{SUPER-ADMINISTRATOR INFORMATION} : Result {SUPER-ADMINISTRATOR INFORMATION} ]
*****

DONE_OK :          KUP_GUI_Administrator_02_01-18 - Super-Administrator
Information Page - Delete - Confirm removal page :          [ Expected page title :
SUPER-ADMINISTRATOR INFORMATIONActual title is :SUPER-ADMINISTRATOR INFORMATION

```

```

DONE_OK :          KUP_GUI_Administrator_02_01-19 - Super-Administrator
Information Page - Back :          [ Expected page title : SUPER-
ADMINISTRATORSActual title is :SUPER-ADMINISTRATORS
*****

```

Après avoir testé la page de suppression en lien avec la page de consultation, nous revenons sur la liste pour tester le filtre de cette liste. Étant donné que ce composant de liste est présent sur toutes les pages listant des composants, nous ne testerons le filtre que sur celle-ci.

```

//Test le tri par login en ordre décroissant.
testList.add(new ListPageColumnSort("KUP_GUI_Administrator_02_01-01 - Super-Administrator Page - Sorting",
    params.loginSA, "admin", params.lastName,"admin"));
testList.add(new ListPageFilters("KUP_GUI_Administrator_02_01-10 - Super-Administrator Page - Filter",
    "0:FuserId",FilterConditions.CONTAINS.toString(),params.loginSA));
testList.add(new ListPageFilters("KUP_GUI_Administrator_02_01-10 - Super-Administrator Page - Filter",
    "0:FuserId",FilterConditions.EQUALS.toString(),"admin"));
testList.add(new ListPageToConsultPage(
    "KUP_GUI_Administrator_02_01-11 - Super-Administrator Page - Link to super admin page",
    "admin", EXPECTED_TITLE_CONSULT_SUPER_ADMIN));
testList.add(new VerifyChangeDoneBySuperAdministrator(
    "KUP_GUI_Administrator_02_01-12 - Super-Administrator Information Page",
    "Deletion"));
testList.add(new ConsultSuperAdminPageReturnButton(
    "KUP_GUI_Administrator_02_01-19 - Super-Administrator Information Page - Back",
    EXPECTED_TITLE_SUPER_ADMIN_LIST));
testList.add(new ListPageCreateStatisticsButton(
    "KUP_GUI_Administrator_02_01-08 - Super-Administrator - Statistic action",
    EXPECTED_TITLE_STATS_SUPER_ADMIN));
testList.add(new CreateStatsPage(
    "KUP_GUI_Administrator_02_01-09 - Super-Administrator - Statistic creation Page",
    EXPECTED_TITLE_STATS_SUPER_ADMIN, "4"));
testList.add(new CreateStatsPageCancelButton(
    "KUP_GUI_Administrator_02_01-09 - Super-Administrator - Statistic creation Page "));

```

Filter

Firstname of the administrator

Contains

superAdmin

×

Add a filter

	Login	Firstname	Lastname
<input type="checkbox"/>	Patrick	superAdmin	superAdmin
<input type="checkbox"/>	testSuperAdmin	superAdminnn	testModified
<input type="checkbox"/>	testSuperAdmin2	superAdmin	superAdmin

Les résultats en console :

```

***** Tests details *****
DONE_OK :          :          current title: 'SUPER-ADMINISTRATORS', expected
title: 'SUPER-ADMINISTRATORS'.
DONE_OK :          KUP_GUI_Administrator_02_01-01 - Super-Administrator Page -
Sorting

```

Ici, on test de trier la liste par login et par lastName:

```

***** Sub Tests details for TestResultList KUP_GUI_Administrator_02_01-01 -
Super-Administrator Page - Sorting *****

```

```

DONE_OK :      KUP_GUI_Administrator_02_01-01 - Super-Administrator Page -
Sorting :      [ sorting ascendant for Login column ]

DONE_OK :      KUP_GUI_Administrator_02_01-01 - Super-Administrator Page -
Sorting :      [ sorting descendant for Login column ]

DONE_OK :      KUP_GUI_Administrator_02_01-01 - Super-Administrator Page -
Sorting :      [ sorting ascendant for Lastname column ]

DONE_OK :      KUP_GUI_Administrator_02_01-01 - Super-Administrator Page -
Sorting :      [ sorting descendant for Lastname column ]
*****
DONE_OK :      KUP_GUI_Administrator_02_01-10 - Super-Administrator Page -
Filter :

```

Ici, on test de trier la liste par paramètres contenant une chaîne de caractère : le firstname contenant la chaîne « superAdmin »

```

***** Sub Tests details for TestResultList KUP_GUI_Administrator_02_01-10 -
Super-Administrator Page - Filter *****
      DONE_OK :      KUP_GUI_Administrator_02_01-10 - Super-
Administrator Page - Filter :      theConditionContains filter performed well

DONE_OK :      KUP_GUI_Administrator_02_01-10 - Super-Administrator Page -
Filter :

***** Sub Tests details for TestResultList KUP_GUI_Administrator_02_01-10 -
Super-Administrator Page - Filter *****

DONE_OK :      KUP_GUI_Administrator_02_01-10 - Super-Administrator Page -
Filter :      theConditionEquals filter performed well
*****

      DONE_OK :      KUP_GUI_Administrator_02_01-11 - Super-
Administrator Page - Link to super admin page :      [ Expected page title :
SUPER-ADMINISTRATOR INFORMATIONActual title is :SUPER-ADMINISTRATOR INFORMATION
      DONE_OK :      KUP_GUI_Administrator_02_01-12 - Super-
Administrator Information Page :      [ Change dones by administrator list:
Deletion ]

```

On test ensuite de créer des statistiques. Cf Annexes

```

      DONE_OK :      KUP_GUI_Administrator_02_01-19 - Super-
Administrator Information Page - Back :      [ Expected page title : SUPER-
ADMINISTRATORSActual title is :SUPER-ADMINISTRATORS
      DONE_OK :      KUP_GUI_Administrator_02_01-08 - Super-
Administrator - Statistic action :      [ After click on create page title
must be : Expected {STATISTIC CREATION} : Result {STATISTIC CREATION} ]
      DONE_OK :      KUP_GUI_Administrator_02_01-09 - Super-
Administrator - Statistic creation Page :      [ All fields/components are
displayed ]
      DONE_OK :      KUP_GUI_Administrator_02_01-09 - Super-
Administrator - Statistic creation Page :      [ Good return page ]
*****

```

Suppression

Pour la suppression, nous avons 5 cas à tester :

- La non suppression d'un administrateur si on ne coche pas de case.
- La non suppression d'un administrateur si on ne valide pas la suppression.
- La suppression d'un super administrateur depuis la liste des super administrateurs (une seule checkbox à cocher).
- La suppression d'un super administrateur depuis sa page de consultation.
- La suppression de plusieurs super administrateurs depuis la liste des super administrateurs (plusieurs checkbox à cocher).

Pour la première option, nous passons simplement une chaîne de caractère avec le login du super administrateur pour que le test aille cocher la checkbox et cliquer sur le bouton de suppression.

Dans la deuxième option, nous passons simplement une chaîne de caractère avec le login du super administrateur pour que le test aille cliquer sur le super administrateur afin d'arriver sur la page de consultation de ce dernier et cliquer sur le bouton de suppression puis valider sa suppression.

Dans la troisième option, au lieu de passer une chaîne de caractère, on passe un tableau de chaîne pour qu'il aille cocher les différentes cases puis cliquer sur le bouton de suppression et valider ou non la suppression (nous aurons le cas d'essayer de supprimer le compte connecté à l'application, ce qui est impossible.)

```
testList.add(new DeleteFromListFailed("KUP_GUI_Administrator_02_01-06 - Super-Administrator - Delete Failed(no Super admin selected)",
    EXPECTED_DELETION_FAILED));
testList.add(new ListPageToDeletePage(
    "KUP_GUI_Administrator_02_01-07 - Super-Administrator - Delete - Confirm removal Page",
    params.loginSA, EXPECTED_TITLE_SUPER_ADMIN_LIST_DELETE));
testList.add(new ConfirmRemovalPage("KUP_GUI_Administrator_02_01-18 - Super-Administrator Information Page - Delete - Confirm removal page",
    EXPECTED_TITLE_SUPER_ADMIN_LIST_DELETE));
testList.add(new DeletePageCancelButton(
    "KUP_GUI_Administrator_02_01-07 - Super-Administrator - Delete - Cancel Button",
    EXPECTED_TITLE_SUPER_ADMIN_LIST));
testList.add(new DeleteFromList(
    "KUP_GUI_Administrator_02_01-06 - Super-Administrator - Delete action - One super admin",
    EXPECTED_TITLE_SUPER_ADMIN_LIST_DELETE,
    logins,
    EXPECTED_DELETION_SUCCESS));
testList.add(new CreationSuccess(
    "KUP_GUI_Administrator_02_01-04 - Super-Administrator Creation Page - Create - Success",
    params.lastName, params.firstName,
    params.email, params.language,
    params.preferredSpreadSheetFormat, params.secureMode,
    params.sendEmailUser, params.keepSettings, params.loginSA,
    params.keepPassword, params.passwordSA, params.passwordConfirmationSA,
    EXPECTED_CREATION_SUCCESS));
testList.add(new DeleteFromConsultPage(
    "KUP_GUI_Administrator_02_01-18 - Super-Administrator Information Page - Delete - Confirm removal page",
    EXPECTED_TITLE_SUPER_ADMIN_CONSULT_DELETE, params.loginSA));
```

Les résultats en console :

Non suppression :

```
DONE_OK :          Delete Failed(no Super admin selected) :

***** Sub Tests details for TestResultList Delete Failed(no Super admin
selected) *****

DONE_OK :          Delete Failed(no Super admin selected) :          [ there is
an alert with text : Please select at least one object.

DONE_OK :          Delete Failed(no Super admin selected) :          [
TotalNumber is the same]
*****
```

Page de suppression :

```
DONE_OK :          Delete - Confirm removal Page :

***** Sub Tests details for TestResultList Delete - Confirm removal Page
*****

DONE_OK :          Delete - Confirm removal Page :          [ A cancel button
must be displayed ]

DONE_OK :          Delete - Confirm removal Page :          [ A delete button
must be displayed ]

DONE_OK :          Delete - Confirm removal Page :          [ Page title must
be : Expected {SUPER-ADMINISTRATORS} : Result {SUPER-ADMINISTRATORS} ]
*****
```


Suppression réussie :

```
DONE_OK :          Delete - Confirm removal Page :

***** Sub Tests details for TestResultList Delete - Confirm removal Page
*****

DONE_OK :          Delete - Confirm removal Page :          [ A cancel button
must be displayed ]

DONE_OK :          Delete - Confirm removal Page :          [ A delete button
must be displayed ]

DONE_OK :          Delete - Confirm removal Page :          [ Page title must
be : Expected {SUPER-ADMINISTRATORS} : Result {SUPER-ADMINISTRATORS} ]
*****
```

Suppression depuis la page d'information :

```
DONE_OK :          Information Page - Delete - Confirm removal page :

***** Sub Tests details for TestResultList Information Page - Delete -
Confirm removal page *****

DONE_OK :          Information Page - Delete - Confirm removal page
: [ A cancel button must be displayed ]
DONE_OK :          Information Page - Delete - Confirm removal page
: [ A delete button must be displayed ]
DONE_OK :          Information Page - Delete - Confirm removal page
: [ Page title must be : Expected {SUPER-ADMINISTRATORS} : Result
{SUPER-ADMINISTRATORS} ]
*****
```

Suppression d'un admin depuis la page de liste :

```
DONE_OK :          Delete action - One super admin :

***** Sub Tests details for TestResultList Delete action - One super admin
*****

DONE_OK :          Delete action - One super admin :          [
TotalNumber -1 ] & [ Super administrator :testSuperAdmin not in the list anymore ]
DONE_OK :          Delete action - One super admin :          [
TotalNumber -1 ] & [ Super administrator :testSuperAdmin2 not in the list anymore
]
DONE_OK :          Delete action - One super admin :          [
TotalNumber -1 ] & [ Super administrator :Patrick not in the list anymore ]
DONE_OK :          Delete action - One super admin :          [ At
least one error message is expected : {Administrator(s) has(have) been deleted
successfully.} Result : {Administrator(s) has(have) been deleted successfully.} ]
*****
```

Suppression de plusieurs super administrateurs :

```
DONE_OK :          Delete - Confirm removal page :          [ Deleted super  
admin from consult page ]
```

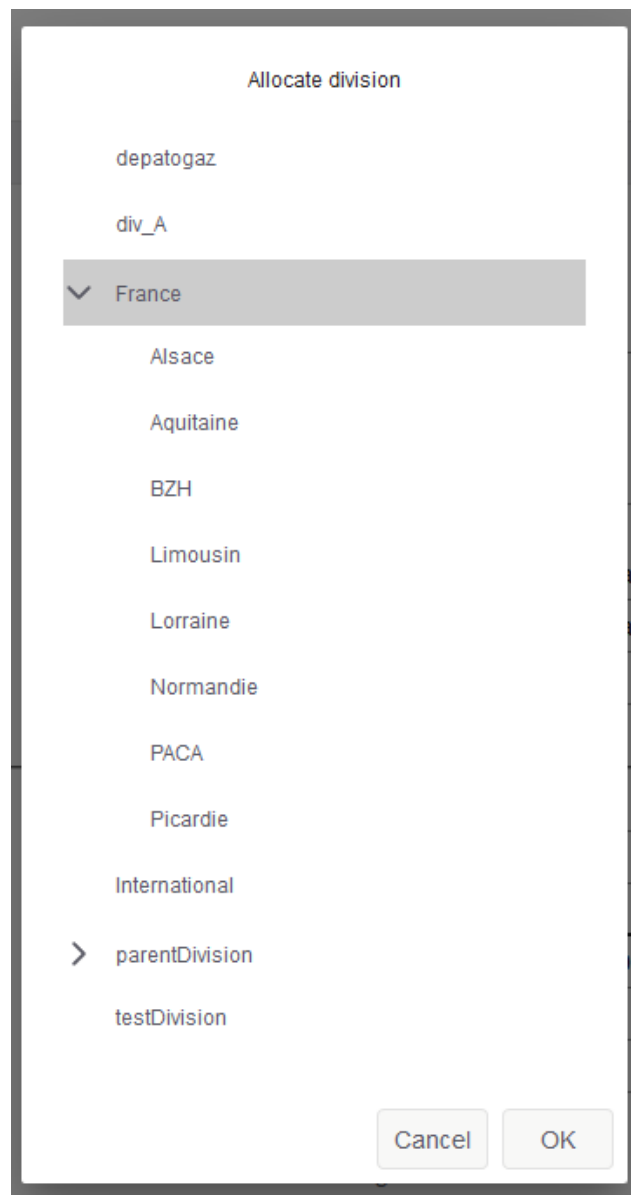
Les tests sur les divisions

(je n'ai pas eu le temps de finir tous les tests sur cet ensemble de page, on m'a rapidement demandé de passer sur autre chose car très long à tester avec Selenium)

Les divisions sont gérées par les super administrateurs et permettent de compartimenter les différents tenants et ce qui leur est relié.)

Par exemple une division peut être reliée à un tenant qui sera lui-même relié à un tenant Template (qui lui donnera ses droits)

Une division peut avoir des enfants et des parents. Par exemple : La France aura comme enfants l'Alsace, l'Aquitaine, la Bretagne, le Limousin, la Normandie, la Picardie etc. Et chacun de ses enfants auront le même parent. Ce qui permettra de les hiérarchiser.



Création

A la manière des super administrateurs, nous allons tester de créer des divisions ou non, si elles existent déjà ou sont incorrectes.

Pour cela, nous commençons par nous rendre sur la page des divisions et vérifions que l'ensemble des champs soit présent.

```
testList.add(new GoToMenuAndCheckTitleTestItem("Initialisation du test",  
    MenuMapping.LIST_DIVISIONS ,EXPECTED_TITLE_DIVISIONS_LIST));  
testList.add(new DivisionsPage("KUP_GUI_Administrator_02_02-01 - Divisions Page"));
```

On test ensuite que le bouton de création depuis la page de liste des divisions marche correctement.

```
testList.add(new DivisionsPageCreateButton(  
    "KUP_GUI_Administrator_02_02-02 - Divisions Page - Create Action",  
    EXPECTED_TITLE_DIVISION_CREATE));
```

Nous vérifions ensuite que tous les champs sont présents sur cette page.

```
testList.add(new CreationPageDivision(  
    "KUP_GUI_Administrator_02_02-03 - Add division Page",  
    EXPECTED_TITLE_DIVISION_CREATE));
```

```
resultList.add(check(creationPage.isCreateButtonDisplayed(),  
    " [ A Create button have to be displayed ] ",  
    creationPage, true));  
resultList.add(check(creationPage.isCancelButtonDisplayed(),  
    " [ A Cancel button have to be displayed ] ",  
    creationPage, true));
```

Une fois sur la page de création, on test que le bouton retour marche correctement.

```
testList.add(new CreationPageCancelButtonDivision(  
    "KUP_GUI_Administrator_02_02-05 - Add division Page - Cancel",  
    EXPECTED_TITLE_DIVISIONS_LIST_ANOTHER_PAGE));
```

Une fois cela fait, on peut commencer à tester la création des divisions.

Nous allons tester qu'elle ne fonctionne pas quand :

- Nous ne mettons pas de nom à la division,
- Le nom excède les 30 caractères,
- Cette division existe déjà,
- La description excède les 30 caractères.

```
//Mandatory name
testList.add(new CreationFailedDivision(
    "KUP_GUI_Administrator_02_02-04 - Add division Page - Create - Fail ",
    "",
    params.description, params.parentDivision, "The parameter 'Name of the division' is mandatory"));
// name exceed 30 characters
testList.add(new CreationFailedDivision(
    "KUP_GUI_Administrator_02_02-04 - Add division Page - Create - Fail ",
    "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa",
    params.description, params.parentDivision, "The parameter 'Name of the division' cannot exceed 30 characters"));
// already exist Division
testList.add(new CreationFailedDivision(
    "KUP_GUI_Administrator_02_02-04 - Add division Page - Create - Fail ",
    logins.get(0), params.description, null, "Division name already exists."));
// description exceed 30 characters
testList.add(new CreationFailedDivision(
    "KUP_GUI_Administrator_02_02-04 - Add division Page - Create - Fail ",
    params.name, "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa", null,
    "The parameter 'Description of the department' cannot exceed 30 characters"));
testList.add(new CreationPageCancelButtonDivision(
    "KUP_GUI_Administrator_02_02-05 - Add division Page - Cancel",
    EXPECTED_TITLE_DIVISIONS_LIST_ANOTHER_PAGE));
```

On vérifie, comme pour les tests de super administrateurs, la présence du bon message d'erreur et la non création d'une division incorrecte (en vérifiant que nous restons sur la même page. Sachant qu'une création réussie renverrait sur la page de liste des divisions.)

Nous testons ensuite que le lien vers la page de statistiques s'effectue bien.

```
//Checks about statistics
testList.add(new DivisionsPageStatisticsButton("", EXPECTED_TITLE_STATISTICS_CREATION, "Division"));
testList.add(new StatisticsPageCancelButtonDivision("", EXPECTED_TITLE_DIVISIONS_LIST_ANOTHER_PAGE));
```

Résultats en console :

```
***** Sub Tests details for TestResultList *****
DONE_OK :           : [ Component type must be : Expected {Division} :
Result {Division} ]
DONE_OK :           : [ After click on create page title must be :
Expected {STATISTIC CREATION} : Result {STATISTIC CREATION} ]
*****
```

Nous testons ensuite la page de consultation d'une division.

```
// Checks about modification
testList.add(new DivisionsPageToConsultPage("",params.name,EXPECTED_TITLE_CONSULT_DIVISION));
testList.add(new ConsultPageDivisions("", params.name, params.description,
    "parentDivision","This division has no child.));

testList.add(new ConsultPageDivisionsToTenantResourceConsultPage("",EXPECTED_TITLE_RESOURCE_CONSULT));
testList.add(new TenantResourceConsult(""));
testList.add(new ResourceConsultCancelButton("",EXPECTED_TITLE_CONSULT_DIVISION));
testList.add(new ConsultPageDivisionsToTenantTemplateResourceConsultPage("",EXPECTED_TITLE_RESOURCE_CONSULT));
testList.add(new TenantTemplateResourceConsult(""));
```

Cela va aller sur chaque page liée à la division. Et décocher toutes les cases, les cocher à nouveau, pour vérifier la présence ou non de nouveaux champs. Cf annexe.

Résultats en console :

```
***** Sub Tests details for TestResultList *****

DONE_OK : : [ is the Component maintenance mode displayed ]
DONE_OK : : [ opening the Component maintenance mode ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : : is Activate maintenance mode displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : : is Activate maintenance mode SELECT displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : : is Activate maintenance mode YES displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : : is Activate maintenance mode NO displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is Display the following message(s) displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is Display the following message(s) SELECT displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is This service is temporarily under maintenance. displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is This service is temporarily under maintenance. SELECT displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is This service is temporarily under maintenance. YES displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is This service is temporarily under maintenance. NO displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is The reopening is scheduled at .... displayed ]
DONE_OK : : [ List of restrictions applied to this equipment : Maintenance
Mode : is The reopening is scheduled at .... SELECT displayed ]
```

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is The reopening is scheduled at YES displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is The reopening is scheduled at NO displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Enter reopening time displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Enter reopening time SELECT displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Display a custom message displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Display a custom message SELECT displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Display a custom message YES displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Display a custom message NO displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Custom message displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Custom message SELECT displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : Send keys bonjour to Custom message]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : add bonjour to Custom message]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : delete bonjour from Custom message]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Activate Sunrise maintenance mode displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Display a custom message SELECT displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Display a custom message YES displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Display a custom message NO displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Activate maintenance mode displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Activate maintenance mode SELECT displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Activate maintenance mode YES displayed]

DONE_OK : : [List of restrictions applied to this equipment : Maintenance
Mode : is Activate maintenance mode NO displayed]

```

DONE_OK : : [ is the Component Template application displayed ]
DONE_OK : : [ opening the Component Template application ]
DONE_OK : : [ List of restrictions applied to this equipment : Template
application : is reapply the template displayed ]
DONE_OK : : [ is the Component Database application displayed ]
DONE_OK : : [ opening the Component Database application ]
DONE_OK : : [ List of restrictions applied to this equipment : Database
application : is reapply the template displayed ]

```

*Ces logs sont à multiplier par 3, beaucoup de composants à tester sur cet ensemble de pages.

Après avoir vérifier toutes les pages en rapport avec la consultation des divisions, on test La suppression d'une division.

```

testList.add(new DeleteFromListFailedDivision(
    "KUP_GUI_Administrator_02_02-06 - Divisions Page - Delete Action - Failed ",
    "Please select at least one object."));
testList.add(new ListDivisionLinkToDeletePage(
    "KUP_GUI_Administrator_02_02-06 - Divisions Page - Delete Action - check link",
    login.get(0),EXPECTED_TITLE_DIVISION_DELETE_FROM_LIST));
testList.add(new DeletePageCancelButtonDivision(
    "KUP_GUI_Administrator_02_02-06 - Divisions Page - Delete Action - Cancel button",
    EXPECTED_TITLE_DIVISIONS_LIST_ANOTHER_PAGE));
testList.add(new DeleteFromListFailedParent(
    "KUP_GUI_Administrator_02_02-06 - Divisions Page - Delete Action - Failed Parent",
    EXPECTED_TITLE_DIVISION_DELETE_FROM_LIST, login,
    "Unable to delete a department which has child departments."));
testList.add(new DeleteFromListDivision(
    "KUP_GUI_Administrator_02_02-06 - Divisions Page - Delete Action - Multiple",
    EXPECTED_TITLE_DIVISION_DELETE_FROM_LIST, logins,
    "The division(s) has(have) been deleted successfully."));
testList.add(new DeleteFromListDivision(
    "KUP_GUI_Administrator_02_02-06 - Divisions Page - Delete Action - One",
    EXPECTED_TITLE_DIVISION_DELETE_FROM_LIST, login,
    "The division(s) has(have) been deleted successfully."));

```

Comme pour les super administrateurs, on test de

- Ne supprimer aucune division en ne cochant pas la case dans la liste
- Ne supprimer aucune division en cochant la case et en cliquant sur supprimer mais en ne validant pas la suppression.
- Ne pas pouvoir supprimer une division parente qui a des enfants
- Supprimer une division
- Supprimer plusieurs divisions.

Conclusion

Bilan

Etant arrivé dans l'entreprise Kurmi Software en janvier, j'ai tout de même eu le temps de me familiariser avec les pratiques de cette entreprise ainsi que les salariés de cette dernière.

J'ai passé un certain nombre de temps sur les tests automatisés avec Selenium.

J'ai aussi eu du mal à commencer car il y avait beaucoup de pattern à respecter, et sachant que c'était aussi nouveau pour l'entreprise, il y a eu des changements pendant que je développais ces tests.

J'ai donc dû m'adapter et revoir mes tests au fur et à mesure. Cela m'a appris à être flexible et à développer de sorte que s'il y a un changement dans l'architecture des tests ou les patterns adoptés, je puisse facilement et plus ou moins rapidement modifier mes tests pour qu'ils correspondent aux nouvelles exigences.

J'ai donc acquis de l'expérience en développement, mais aussi dans mes méthodes de travail et ma hiérarchisation des différentes tâches à réaliser.

Après avoir testé de nombreux champs de l'application et m'être familiarisé avec le framework Selenium, je suis plus rapide, je développe plus proprement et suis plus exigeant avec moi-même sur ce que je développe.

J'ai aussi appris à écrire des cahiers de test, en partant d'une Feature request développée par un ou plusieurs développeurs sur la demande des clients.

J'ai aussi appris à valider et exécuter un cahier de test écrit par mes collègues après la sortie de feature request.

Problèmes rencontrés

Comme dit précédemment l'un des problèmes que j'ai pu rencontré est d'avoir dû m'adapter aux nouvelles exigences et patterns de tests Sélénium à adopter qui ont changé plusieurs fois entre ma formation d'une semaine au début et le mois de juin.

Une des parties compliquées était aussi de comprendre comment fonctionnait l'application et les différentes pages, étant donné que c'est une grosse application dans un domaine que je ne connaissais pas du tout en arrivant chez Kurmi. J'ai dû passer beaucoup de temps à regarder le code développé par mes collègues afin de savoir comment réutiliser l'existant et le mettre à jour au besoin.

L'un des points sur lesquels j'ai eu du mal parfois étaient les sélecteurs CSS, trouver le moyen pour de multiples éléments de ne sélectionner que ceux qui m'intéressait, et que ce soit aussi dynamique.

Par exemple pour un tableau d'éléments, trouver le bon sélecteur permettant de ne sélectionner que le contenu de CE tableau en particulier, en sachant que chaque tableau avait la même classe et que le sélecteur CSS de ce tableau utilisé sur plusieurs pages pouvait changer selon la page.

Ne faisant pas uniquement des tests automatisés avec Sélénium mais faisant aussi des tests à la main en suivant des cahiers de tests, pour tester des nouvelles fonctionnalités et reporter des bugs avec l'outil Bugzilla, il était parfois difficile de se replonger dans ce que j'avais fait il y a de ça plus d'un mois en Sélénium.

Remerciements

Je tiens encore une fois à remercier Kurmi Software de m'avoir accueilli.

Ainsi que MyDigitalSchool et l'ensemble des intervenants de cette école de m'avoir permis d'apprendre de nombreuses choses et de m'avoir conforter dans mon but de devenir développeur.

Annexes

Consultation des divisions

General information
<div>Name : parentDivision</div> <div>Description : divisionDescription</div> <div>Division's parent : This division is a root division.</div> <div>Division's children : childDivision</div>

Additional information
<div>a :</div> <div>biglimit :</div>

Division management
<div>List of available resource types for this division (Click for restrictions) :<div>✓ TENANT</div><div>✓ KURMISERVER</div><div>✓ TENANTTEMPLATE</div><div></div></div> <div>Creation profile list available for this division : <div>test</div><div></div></div>

Components
<div>Providers :</div> <div>Tenants :</div> <div>Tenant templates :</div>

Et les pages liées : TENANT / KURMISERVER / TENANTTEMPLATE

TENANT

General information	
Division name : parentDivision	
Equipment name : TENANT	

List of restrictions applied to this equipment	
Maintenance mode	
Activate maintenance mode ⓘ :	<input checked="" type="checkbox"/> Keep inherited value
Display the following message(s) :	<input checked="" type="checkbox"/> Keep inherited value
" This service is temporarily under maintenance. " :	<input checked="" type="checkbox"/> Keep inherited value
" The reopening is scheduled at " :	<input checked="" type="checkbox"/> Keep inherited value
Enter reopening time :	<input checked="" type="checkbox"/> Keep inherited value
Display a custom message :	<input checked="" type="checkbox"/> Keep inherited value
Custom message :	<input checked="" type="checkbox"/> Keep inherited value
Activate Sunrise maintenance mode ⓘ :	<input checked="" type="checkbox"/> Keep inherited value

Template application	
Reapply the template : <input checked="" type="checkbox"/> Keep inherited value	

Database information	
Database server :	<input checked="" type="checkbox"/> Keep inherited value
Database server default configuration :	<input checked="" type="checkbox"/> Keep inherited value
Login :	<input checked="" type="checkbox"/> Keep inherited value
Auto-generated password :	<input checked="" type="checkbox"/> Keep inherited value
Password :	<input checked="" type="checkbox"/> Keep inherited value
Pool maximum number of connections :	<input checked="" type="checkbox"/> Keep inherited value
Pool maximum number of idle connections :	<input checked="" type="checkbox"/> Keep inherited value

Tenant custom parameters	
accessListMapping :	<input checked="" type="checkbox"/> Keep inherited value
adresseip :	<input checked="" type="checkbox"/> Keep inherited value
fezfe :	<input checked="" type="checkbox"/> Keep inherited value
siret :	<input checked="" type="checkbox"/> Keep inherited value
tenantHostname :	<input checked="" type="checkbox"/> Keep inherited value

KURMISERVER

General information	
Division name : parentDivision	
Equipment name : KURMISERVER	

List of restrictions applied to this equipment	
General information	
Prepare server for shutdown ⓘ :	<input checked="" type="checkbox"/> Keep inherited value
Disable Sunrise ⓘ :	<input checked="" type="checkbox"/> Keep inherited value

TENANTTEMPLATE

General information

Division name : parentDivision

Equipment name : TENANTTEMPLATE

List of restrictions applied to this equipment

Database information

Database server : ☒ Keep inherited value
Database server default configuration : ☒ Keep inherited value
Login : ☒ Keep inherited value
Auto-generated password : ☒ Keep inherited value
Password : ☒ Keep inherited value
Pool maximum number of connections : ☒ Keep inherited value
Pool maximum number of idle connections : ☒ Keep inherited value

Tenant custom parameters

accessListMapping : ☒ Keep inherited value
adresseip : ☒ Keep inherited value
fezfe : ☒ Keep inherited value
siret : ☒ Keep inherited value
tenantHostname : ☒ Keep inherited value

Other informations

Synchronized tables : ☒ Keep inherited value
☒ Use regular expressions

Cases décochées :

Database server : ☒ Keep inherited value
☐ Keep inherited value

Database server default configuration : ☒ No

☐ Keep inherited value

Login : Select...

☐ Keep inherited value

Auto-generated password : Select...

Password : ☐ No

☐ Yes

Pool maximum number of connections : ☒ Keep inherited value

Pool maximum number of idle connections : ☒ Keep inherited value

Formulaire React

NORELOAD_REACT

Apply

Mass extract file...

Add to table

Reset

General information

⊗ one

multiFree : ⊗ two

⊗ ▼

⊗ first

multiFreeOrder : ⊗ second

⊗ ▼

swichTest : ☐

swichTestFalse : swichTestFalse

swichTestFalse2 : swichTestFalse2

A :



value de A:

B * :

chargeMePossibleValNb : 0

Allocate division :



askDate : 1/1/14 12:00:00 am



compoType : TENANT

askCompoSimple :

⊗ ▼

Advanced

chargeMe : ☒

chargeMeNb : 0

Scheduling

Processing ⓘ : Immediate



Tracking number ⓘ :

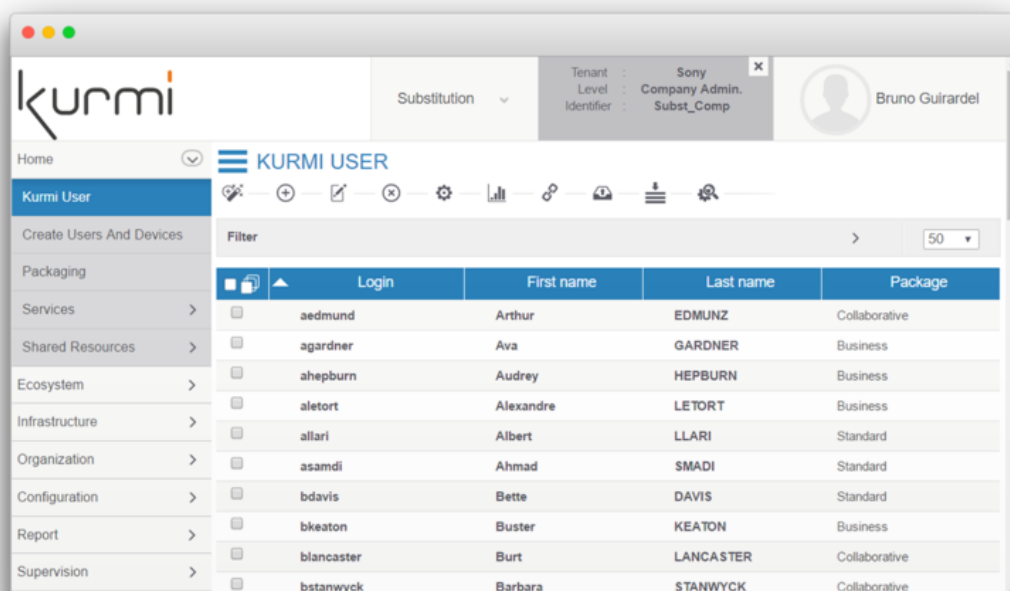


Comment ⓘ :



Kurmi Unified Provisionning

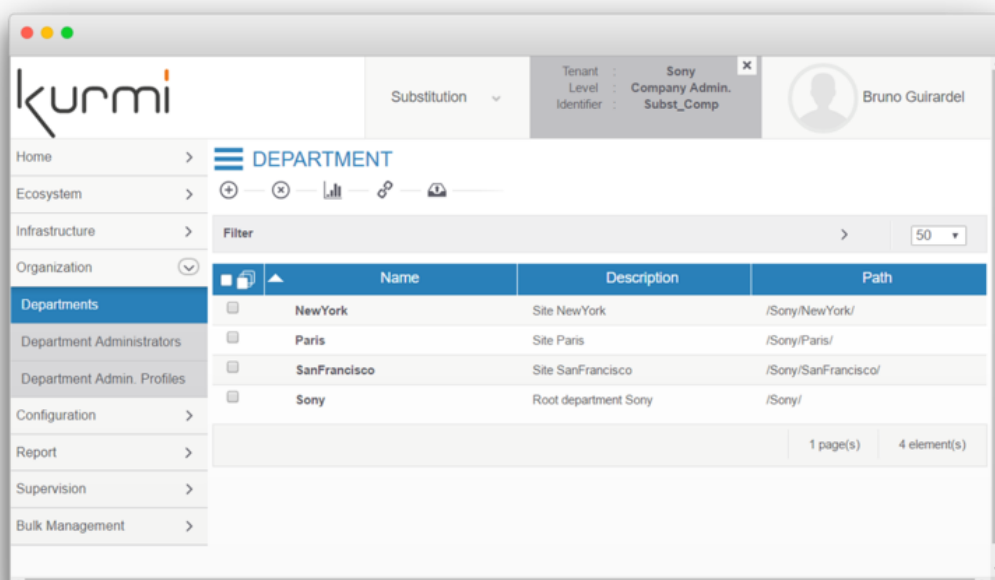
Interface Basique



The screenshot shows the Kurmi User interface. The top header includes the Kurmi logo, a 'Substitution' dropdown, tenant information (Sony, Company Admin, Subst_Comp), and a user profile for Bruno Guirardel. The left sidebar contains navigation links: Home, Kurmi User (selected), Create Users And Devices, Packaging, Services, Shared Resources, Ecosystem, Infrastructure, Organization, Configuration, Report, and Supervision. The main content area is titled 'KURMI USER' and features a table of users with columns for Login, First name, Last name, and Package. A filter bar is located above the table.

Filter	Login	First name	Last name	Package
<input type="checkbox"/>	aedmund	Arthur	EDMUNZ	Collaborative
<input type="checkbox"/>	agardner	Ava	GARDNER	Business
<input type="checkbox"/>	ahepburn	Audrey	HEPBURN	Business
<input type="checkbox"/>	aletort	Alexandre	LETORT	Business
<input type="checkbox"/>	allari	Albert	LLARI	Standard
<input type="checkbox"/>	asamdi	Ahmad	SMADI	Standard
<input type="checkbox"/>	bdavis	Bette	DAVIS	Standard
<input type="checkbox"/>	bkeaton	Buster	KEATON	Business
<input type="checkbox"/>	blancaster	Burt	LANCASTER	Collaborative
<input type="checkbox"/>	bstanwyck	Barbara	STANWYCK	Collaborative

Departements

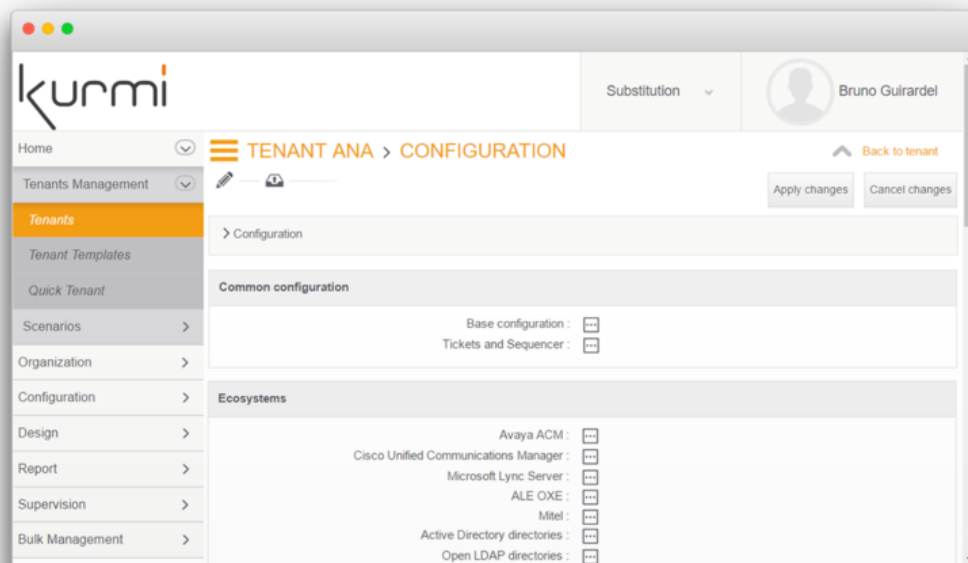


The screenshot shows the Kurmi DEPARTMENT interface. The top header is identical to the previous screenshot. The left sidebar shows navigation links: Home, Ecosystem, Infrastructure, Organization, Departments (selected), Department Administrators, Department Admin. Profiles, Configuration, Report, Supervision, and Bulk Management. The main content area is titled 'DEPARTMENT' and features a table of departments with columns for Name, Description, and Path. A filter bar is located above the table.

Name	Description	Path
NewYork	Site NewYork	/Sony/NewYork/
Paris	Site Paris	/Sony/Paris/
SanFrancisco	Site SanFrancisco	/Sony/SanFrancisco/
Sony	Root department Sony	/Sony/

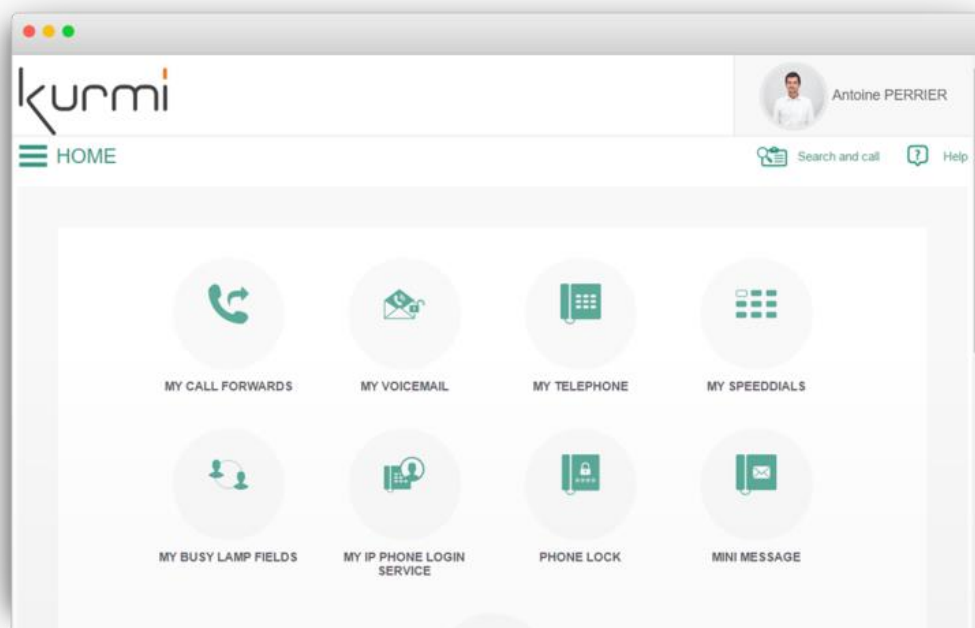
1 page(s) 4 element(s)

Configuration d'un tenant

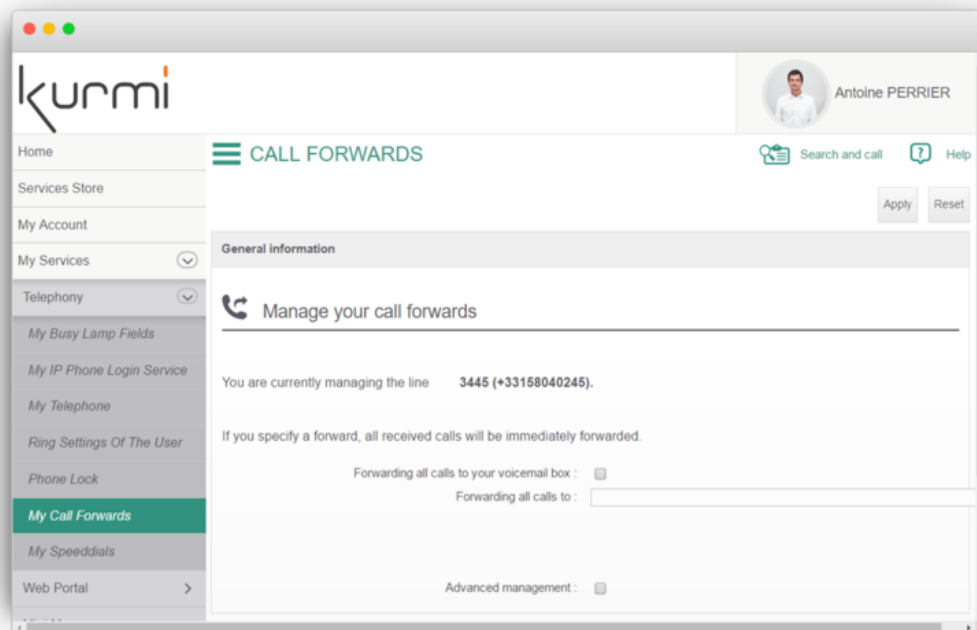


Kurmi Unified Selfcare

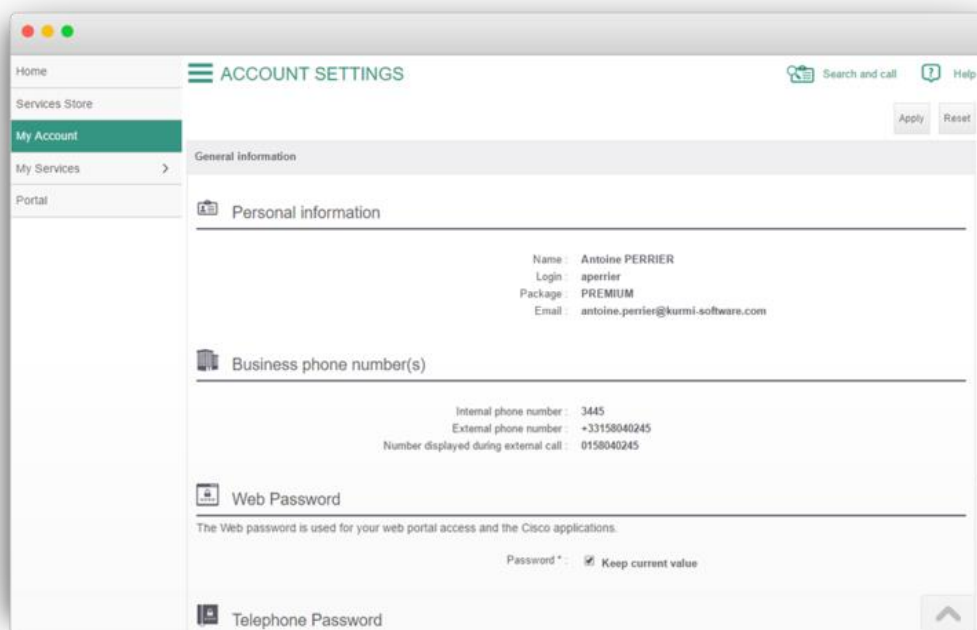
Interface Basique



Call Forward

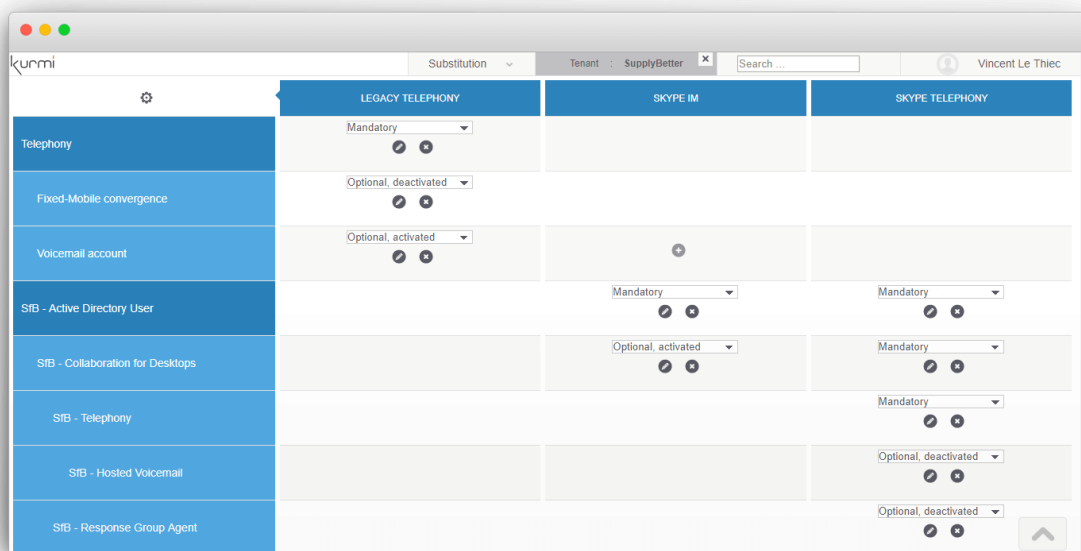


Account Settings



Kurmi Migration Tool

Interface de base



Choisir la durée et la date de migration

The screenshot shows the 'Mass Modify' tab. The table below lists 16 users with their component labels, login details, departments, and migration status.

	Component label	Login	Department	Language	Login	First name	Last name	Package	
1	Brigitte L OERPER	brigitte.loerper@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
2	David Newton	david.newton@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
3	Hakan ERKIN	hakan.erkim@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
4	Isabelle Martin	isabelle.martin@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
5	Jane Rambert	jane.rambert@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
6	Joel Kerman	joel.kerman@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
7	Kim WOLTERING	kim.woltering@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
8	Kristin NAIDZINAVIC...	kristin.naidzinavicius@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
9	Lone SCHULZE	lone.schulze@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
10	Loris Cleveau	loris.cleveau@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
11	Lucas Pierce	lucas.pierce@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
12	Luisa RODE	luisa.rodé@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
13	Mark Schmidt	mark.schmidt@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
14	Susann MÜLLER	susann.muller@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
15	Thomas BEHNEKE	thomas.behneke@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep
16	Udo SMITS	udo.smits@supplybetter.com	/Head/EMEA/Germany	Keep current value	Keep current value	Keep current value	Keep current value	Skype IM	Keep

Reporter la configuration des futurs collaborateurs après migration

