

ESTEBAN LE CALVEZ

Rapport de Stage

Alternance Qualité & Intégration

Postulant au titre de Concepteur Développeur Informatique

Réalisé par

LE CALVEZ Estéban

Sommaire

[Remerciements 6](#_Toc517538798)

[Introduction 7](#_Toc517538799)

[Présentation de l’entreprise 8](#_Toc517538800)

[Présentation de l’entreprise 8](#_Toc517538801)

[Organigramme 8](#_Toc517538802)

[Les différents produits de Kurmi Software 11](#_Toc517538803)

[1 - Kurmi Unified Provisionning 11](#_Toc517538804)

[2 - Kurmi unified selfcare 12](#_Toc517538808)

[3 - Kurmi Migration Tool 13](#_Toc517538812)

[Technologies utilisées 14](#_Toc517538816)

[Eclipse IDE 14](#_Toc517538817)

[Win SCP 14](#_Toc517538818)

[Visual Studio Code 15](#_Toc517538819)

[SourceTree 15](#_Toc517538820)

[Microsoft Teams 16](#_Toc517538821)

[BugZilla 16](#_Toc517538822)

[Trello 17](#_Toc517538823)

[Squash 17](#_Toc517538824)

[Selenium 18](#_Toc517538825)

[Projet 19](#_Toc517538826)

[Analyse de l’existant 19](#_Toc517538827)

[Objectif du projet 19](#_Toc517538828)

[Fonctionnalités de Selenium 20](#_Toc517538829)

[Utilisation de Selenium chez Kurmi 21](#_Toc517538830)

[Pattern adopté 21](#_Toc517538831)

[Les tests avec Selenium 23](#_Toc517538832)

[Les tests sans Sélénium 25](#_Toc517538834)

[Les différents tests exécutés 26](#_Toc517538835)

[Formulaire React 26](#_Toc517538836)

[Tests 28](#_Toc517538837)

[Les tests sur les super administrateurs. 34](#_Toc517538838)

[Création 35](#_Toc517538839)

[Connexion de l’utilisateur créé 38](#_Toc517538840)

[Modification 39](#_Toc517538841)

[Suppression 47](#_Toc517538842)

[Les tests sur les divisions 51](#_Toc517538843)

[Création 52](#_Toc517538844)

[Conclusion 57](#_Toc517538845)

[Bilan 57](#_Toc517538846)

[Problèmes rencontrés 58](#_Toc517538847)

[Remerciements 59](#_Toc517538848)

[Annexes 60](#_Toc517538849)

[Consultation des divisions 60](#_Toc517538850)

[TENANT 61](#_Toc517538851)

[KURMISERVER 61](#_Toc517538852)

[TENANTTEMPLATE 62](#_Toc517538853)

[Formulaire React 63](#_Toc517538854)

[Kurmi Unified Provisionning 64](#_Toc517538855)

[Kurmi Unified Selfcare 65](#_Toc517538856)

[Kurmi Migration Tool 67](#_Toc517538857)

# 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 Selenium et m’avoir aidé quand j’avais des questions.

# 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 étais déployés à chaque nouvelle version afin de tester les composants d’interface de la suite de produits Kurmi.

Au cours des missions effectués, 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 suggérés par une des équipe 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 connaitre 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 équipe 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](https://www.kurmi-software.com/fr/solutions/outil-de-provisioning-cisco-call-manager/) 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.

### 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.

### 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.

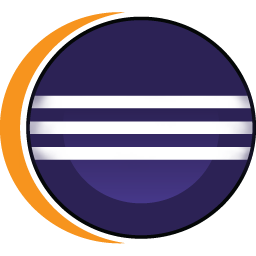
Vos administrateurs peuvent migrer site par site, utilisateur par utilisateur et réagir à tout imprévu en adaptant les règles de migration tout au long de votre projet.

## **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.

# Technologies utilisées



#### Eclipse IDE

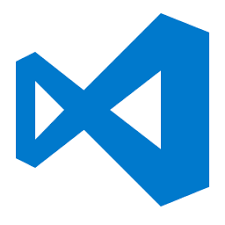
Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.



#### 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 ??? 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.

- Etre 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 element 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 ou 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

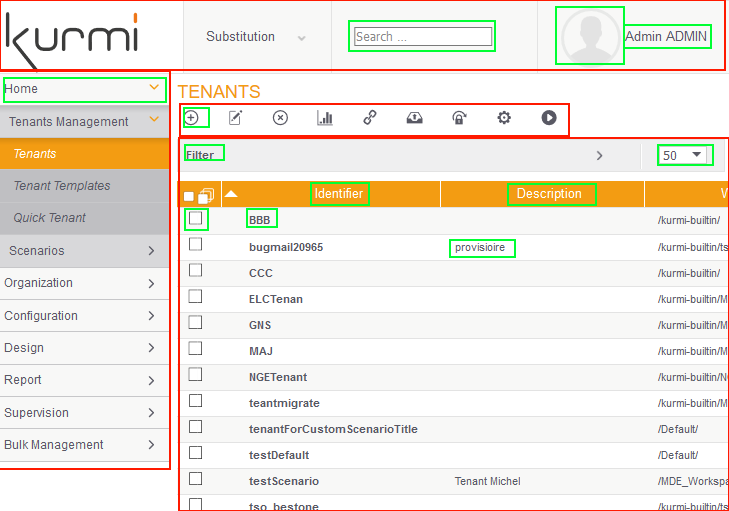
Le pattern que nous avons adopté est le Page Object.

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 :



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

Eeee 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 sous 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.isActionCreateDiplayed(),

MESSAGE\_CREATE, listDivisionPage, **true**));

//Ici nous verifions que le bouton Delete est affiché.

resultList.add(check(listDivisionPage.isActionDeleteDiplayed(),

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 autre 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 développé par un collègue développeur qui permettait au créateur d’un scénario d’afficher un titre ou un autre sur les étapes du scénario selon si l’utilisateur à les droits 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 devait s’afficher dans un certain ordre, si l’étape était en fonction de si l’étape était en erreur ou non et que l’utilisateur avait ou non les droits d’accéder à cette étape.

Certain 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 développés par ensemble de page.

#### Formulaire React

Pour donner suite à une mise à jour de l’interface 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 entrainait 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.

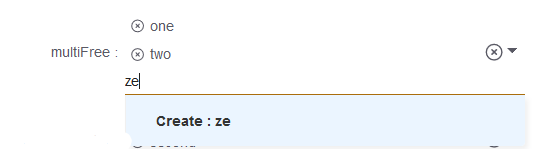
Ce changement dans la structure de l’application pouvant entrainer de nombreux bugs, un développeur à créer une page avec à peu près tous les champs possibles modifié par l’implémentation de react dans Kurmi. Cette page est une quick feature que l’on peut intégrer ou non très facilement dans l’application.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

Le test de se 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 chaine 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.

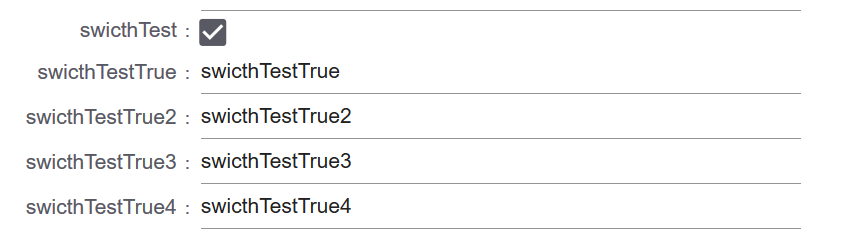
On appel ensuite un seul testItem (pour le moment) qui ira tester tous les champs de la page. Cf : Annexe Formulaire React.

Cependant, avec cette mise à jour de React, les composants ont changé de sélecteurs CSS, c’est pourquoi je suis resté bloqué sur le champs allocate division, qui fait apparaitre 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, chaines de caractères, nombres.

Par exemple, numberOfSwicthTest() renvoit un nombre qui est le nombre de champs en dessous de la checkbox swicthTest. Le nombre de champs et à 2 quand la checkbox n’est pas cochée, et est à 4 quand la checkbox est cochée.



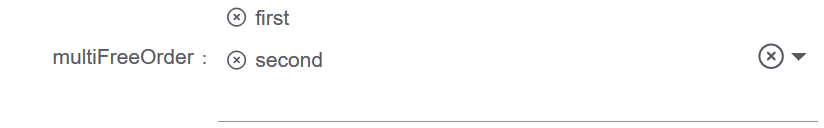
Dans le cas ci-dessus, la méthode renverra 4.

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. 

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).



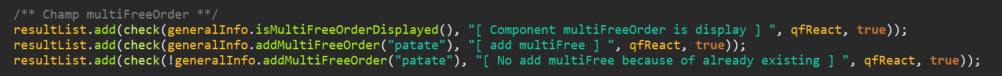
On crée donc des méthodes permettant d’utiliser ce champ.

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

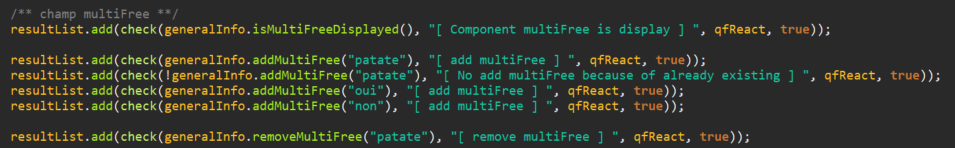
-int numbreOfOptionsIntoMultiFreeOrder() //Renvoit le nombre de champs dans cette liste.

Chaque champ aura ces méthodes permettant de tester si le comportement est bon ou non. On commence donc par tester le champ multiFreeOrder, en verifiant 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

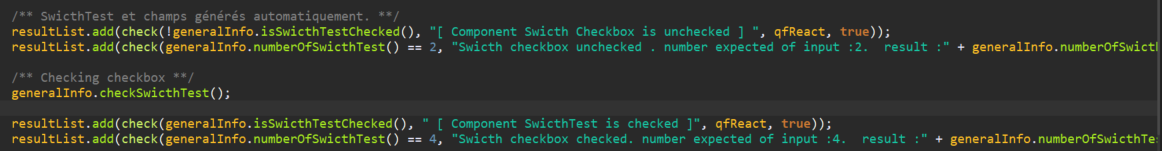


Pour ce champ, nous allons tester qu’il est bien affiché, de la même façon que l’ancien champ, testé d’ajouté une fois « patate » avec succès puis une seconde fois « patate » sans succès.

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.

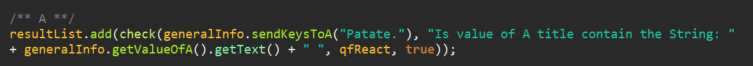


De base, la checkbox est décochée, nous vérifions donc qu’il y a 2 éléments affichés en dessous.

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 chaine de caractère qui sera reportée dans le titre du champs en dessous.

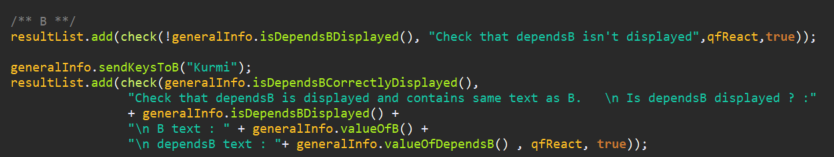




La méthode sendKeysToA(String chaine) va envoyer la chaine de caractère passé en paramètre dans le champs « A » puis va aussi vérifier que la valeur du champ   
« value de A :XXX » contient bien la chaine de caractère passée dans le champs A, puis renverra un booléen.

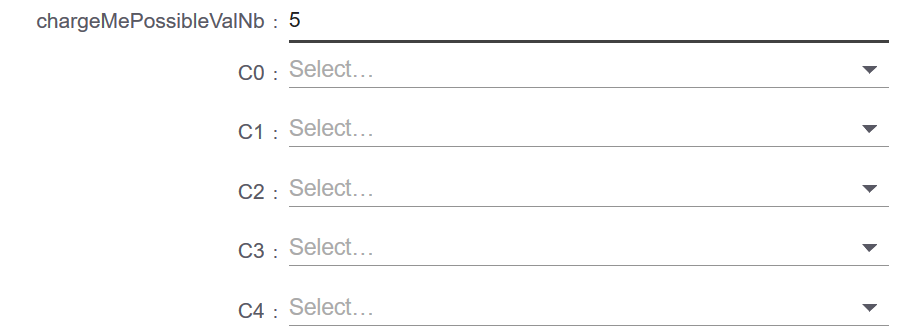
Le prochain champ « B » qui quand il est vide ne crée pas de second champ. Et qui quand on le remplit crée un second champ « depends B » qui va contenir la même chaine de caractère que « B ».



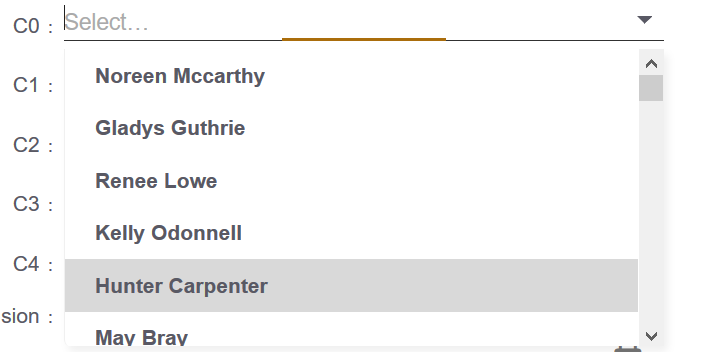


Nous allons donc regardé si ce champ est affiché sur la page, puis lui passer la chaine Kurmi, grâce à la méthode isDependsBCorrectlyDisplayed() qui permettra de vérifier si le champs « depends B » est affiché et vérifier que la chaine de caractère passée dans « B » est la même que celle qui apparaitra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

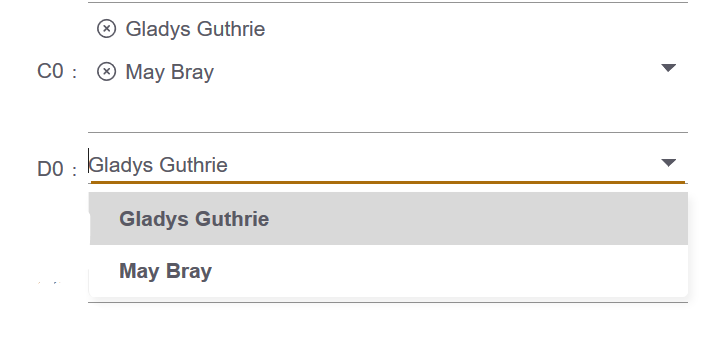
Le prochain champ à tester est celui du ChargeMePossibleValNb, un champ dans lequel nous allons passer un nombre, afin qu’il génère se nombre d’éléments en dessous.



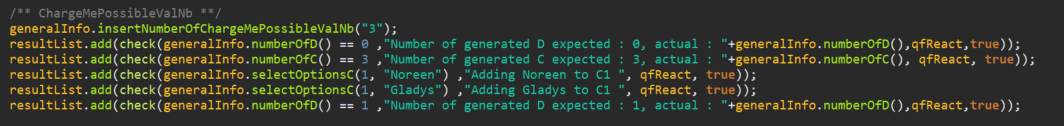
Cette liste d’éléments générés ira chercher un json contenant des noms de personnes.



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.



Dans cette liste, nous ne pourrons sélectionner qu’un seul élément.



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

J’ai commencé par inséré « 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 chekcbox, l’autre depuis la page d’information)

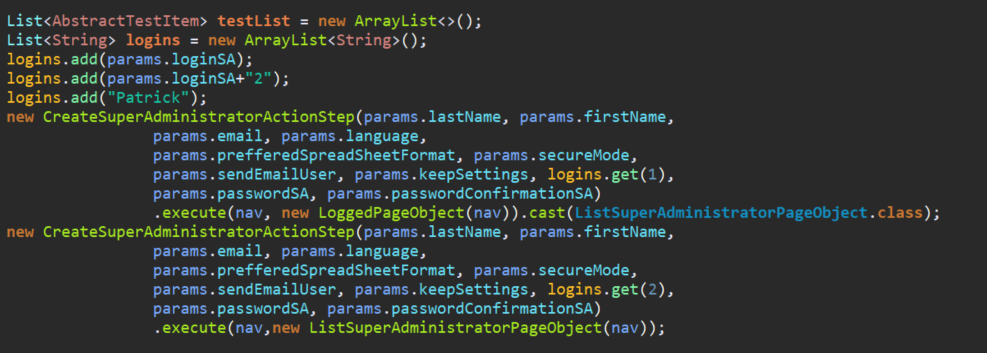
Sur chaque page je devais vérifier que tous les composants étaient présents.

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.



Cette ActionStep :

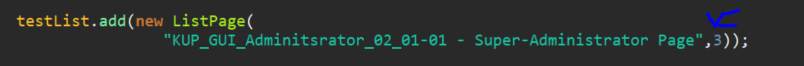
-Clique sur le menu des super administrateurs

-Clique sur le bouton de création

-Remplit les différents champs du formulaire avec les paramètres passés

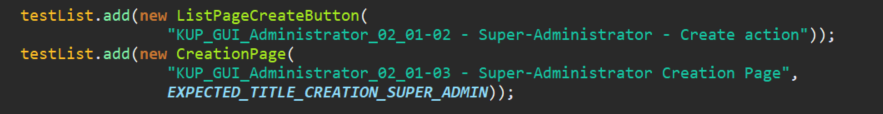
-Clique sur le bouton de création du super administrateur.

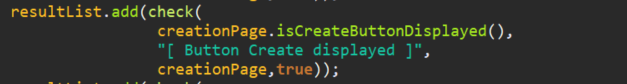
-Vérifie qu’on arrive bien sur la liste des super administrateurs (sinon, c’est que quelque chose s’est mal passé).

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.

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.

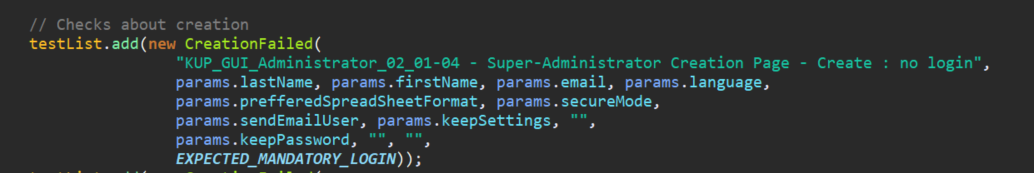




Résultat en console quand tout est OK.



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



On passe en paramètres les valeurs des champs qui vont être remplit 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} ]

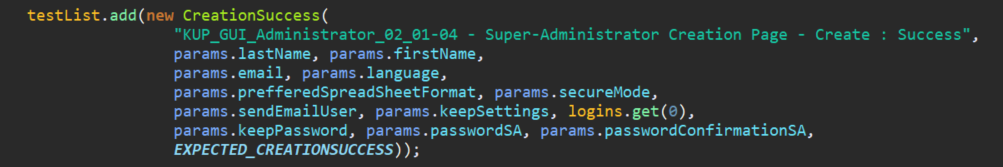
DONE\_NOT\_OK : CreationFailed : [ At least one error message must be : {BDD\_UNIQUE\_CONSTRAINT\_VIOLATION\_NATES\_ADMIN\_USERID} result are : {The value "admin" already exists in the database.} ]

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)

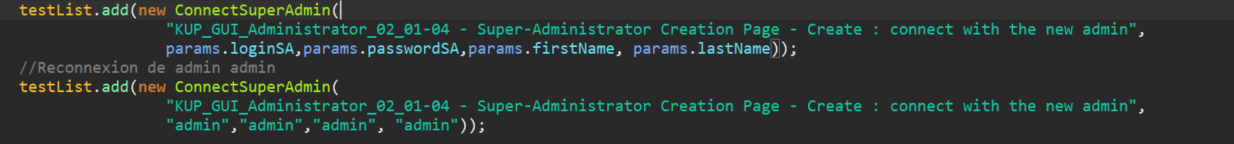
Après avoir testé qu’on avait les bons messages d’erreur quand une création ne réussis pas ; on test que la création marche bien quand on rentre les bons paramètres.



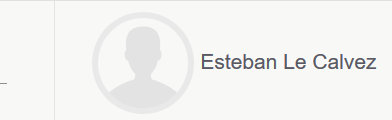
Ce test est sensiblement le même mais on arrivera sur la page avec la liste des super administrateurs et un message de succès nous disant que l’utilisateur est créé.

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.



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.

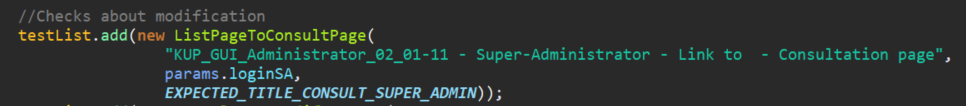


On reconnecte ensuite le super administrateur par défaut pour continuer les tests.

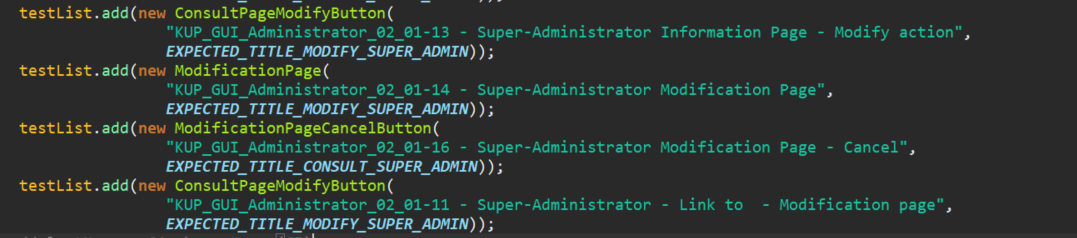
Modification

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.



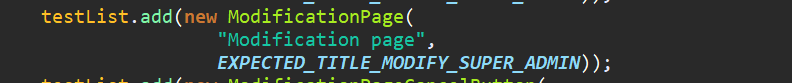
On passe donc en paramètre le login du Super administrateur afin que le test clique dessus et vérifie l’intitulé de la page.

On clique ensuite sur le bouton de modification. 

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 :



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 ]

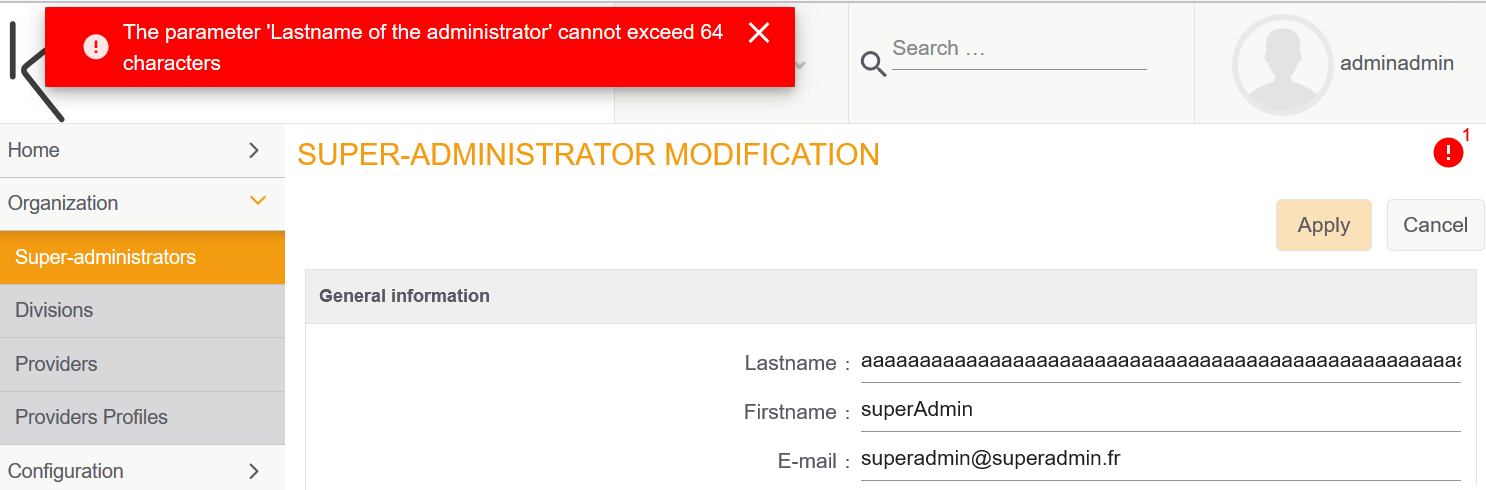
DONE\_OK : Modification page : [ SendUserEmail field displayed ]

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.



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} ]

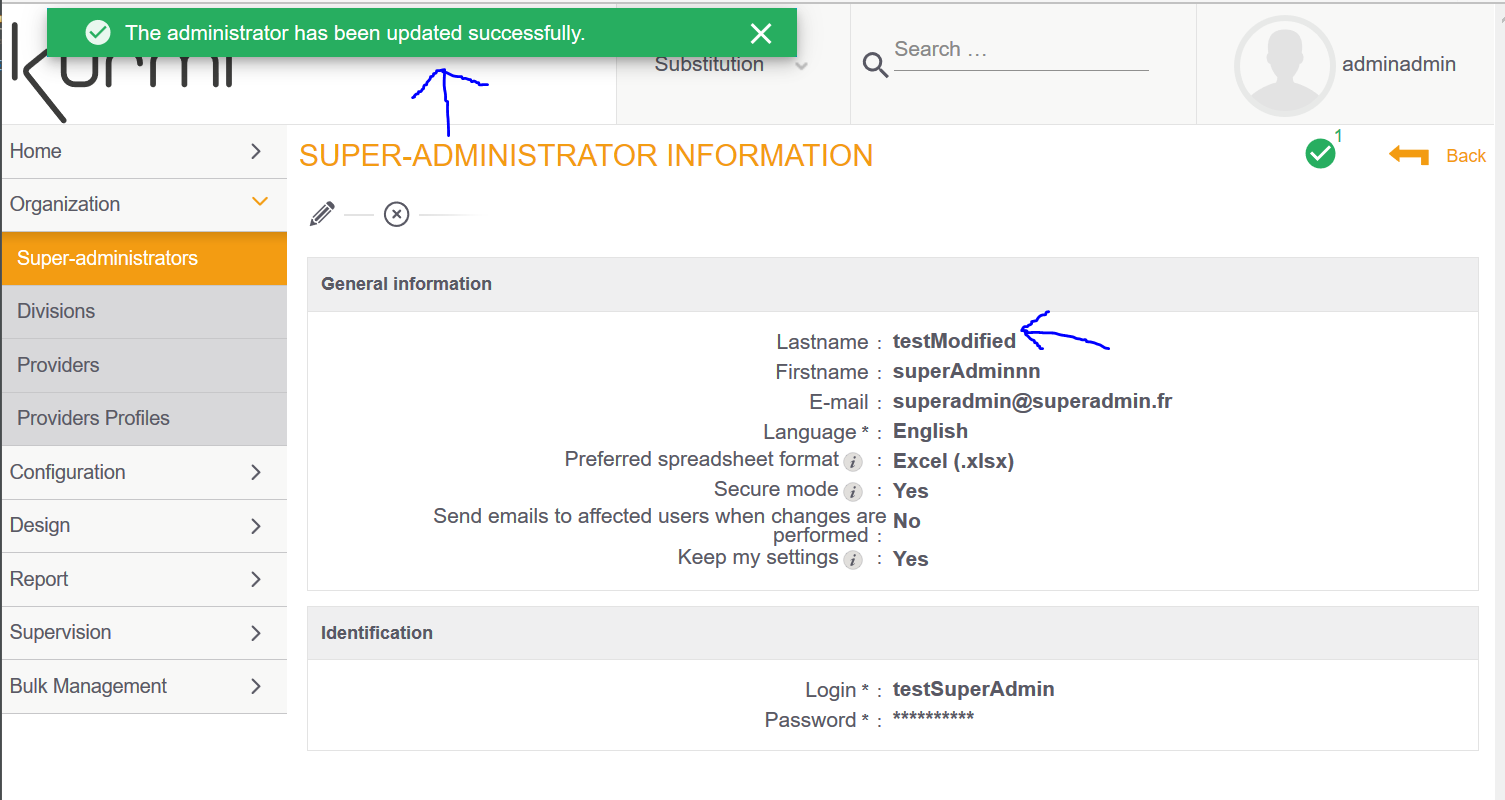
DONE\_OK : ModificationFailed : [ At least one error message must be : {The parameter 'Password of the administrator' cannot exceed 30 characters} result are : {The parameter 'Password of the administrator' cannot exceed 30 characters} ]

DONE\_OK : ModificationFailedNotMatchingPasswords : Expected alert message {Passwords do not match !}. Actual alert message : {Passwords do not match !}

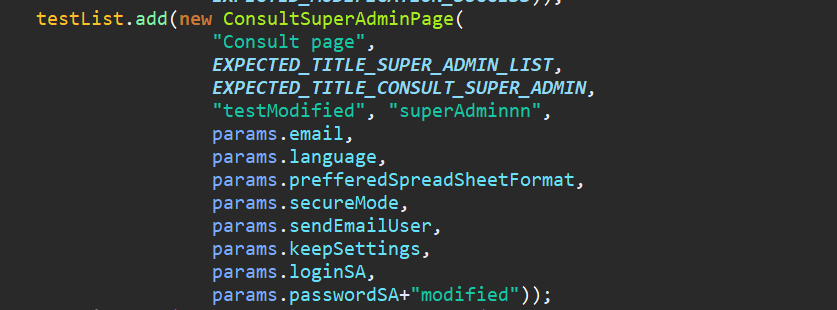
DONE\_OK : ModificationSuccess : [ At least one error message is expected : {The administrator has been updated successfully.} Result : {The administrator has been updated successfully.} ]

DONE\_OK : KUP\_GUI\_Administrator\_02\_01-12 - Super-Administrator Information Page

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



On vérifie aussi que les champs on correctement été modifiés.



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)} ]

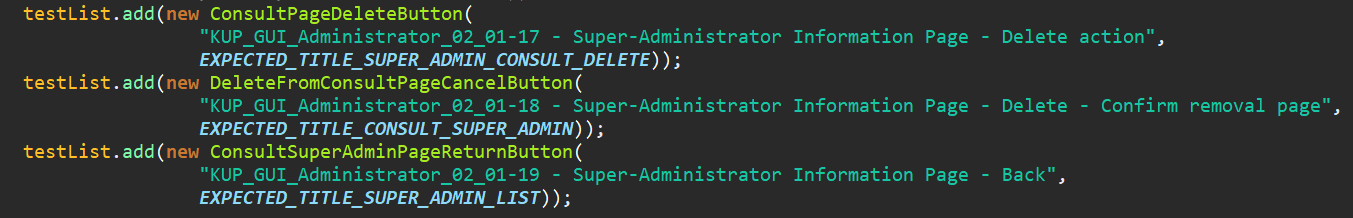
DONE\_OK : Consult page : [ The Keep Settings field is incorrect : Expected {Yes} : Result {Yes} ]

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. 

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} ]

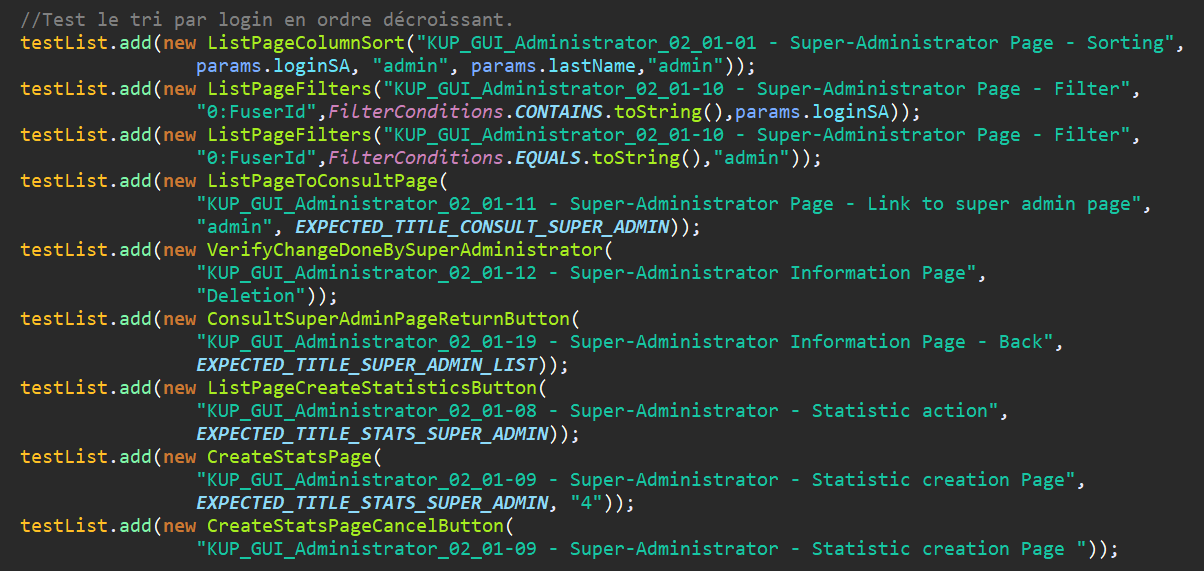
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

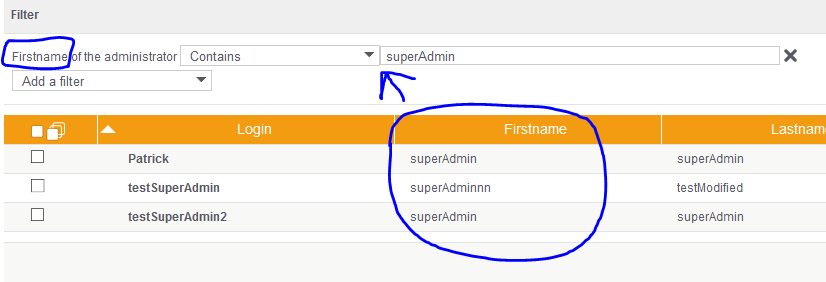
DONE\_OK : KUP\_GUI\_Administrator\_02\_01-18 - Super-Administrator Information Page - Delete - Confirm removal page : [ Expected page title : SUPER-ADMINISTRATOR INFORMATIONActual title is :SUPER-ADMINISTRATOR INFORMATION

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. Etant donné que se composant de liste est présent sur toutes les pages listant des composants, nous ne testerons le filtre que sur celle-ci.





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 chaine de caractère : le firstname contenant la chaine « 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).

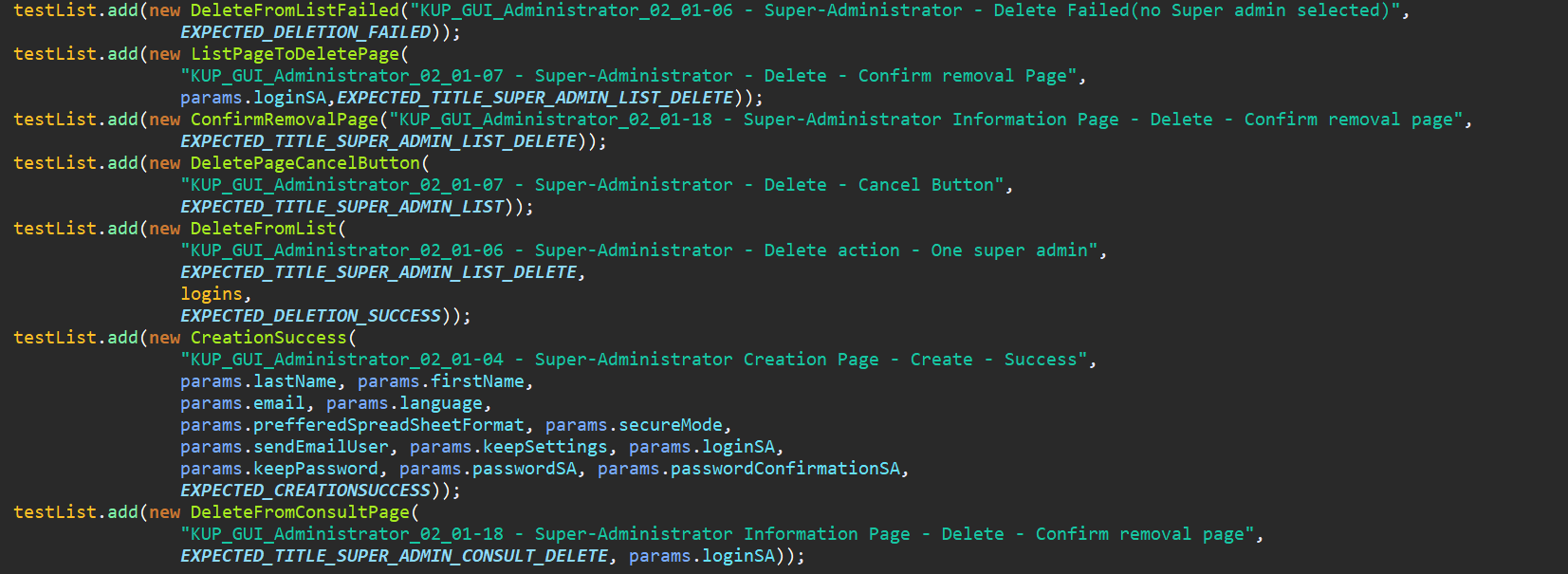
-La suppression d’un super administrateur depuis sa page de consultation.

-La suppression de plusieurs super administrateurs depuis la liste des super administrateurs (plusieurs checkbox à cocher).

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

Dans la deuxième option, nous passons simplement une chaine 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 se dernier et cliquer sur le bouton de suppression puis valider sa suppression.

Dans la troisième option, au lieu de passer une chaine de caractère, on passe un tableau de chaine 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.)



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 ]

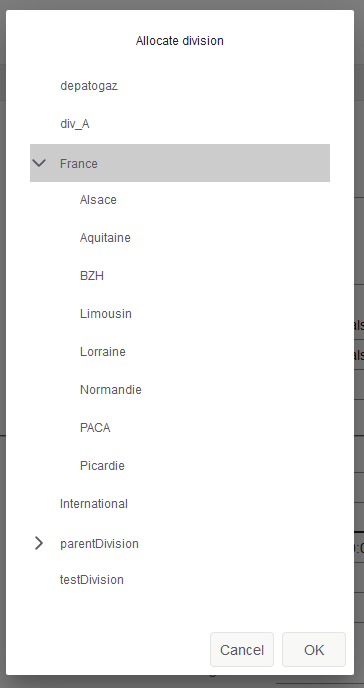
#### Les tests sur les divisions

(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 tenant et ce qui leur est relié.)

Par exemple une division pour été relié 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.

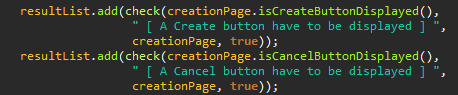


On test ensuite que le bouton de création depuis la page de liste des divisions marche correctement.



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





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



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

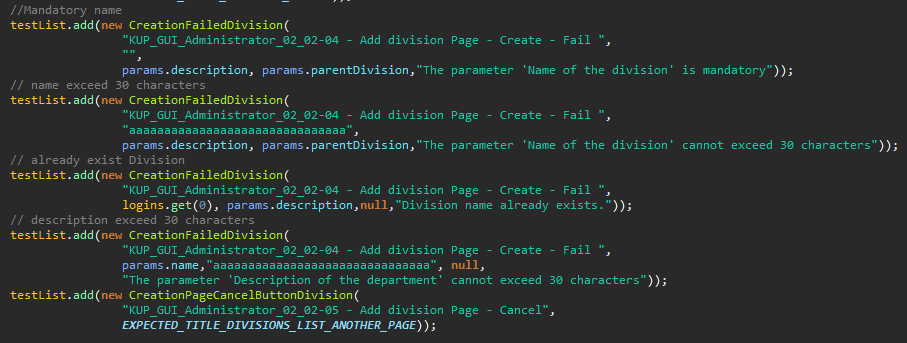
Nous allons tester qu’elle ne fonctionne pas quand :

-Nous ne mettons pas de nom à la division,

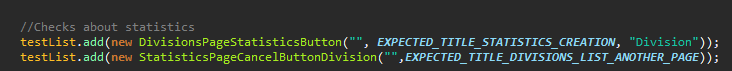
-Le nom excède les 30 caractères,

-Cette division existe déjà,

-La description excède les 30 caractères.



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. 

Résultats en console :

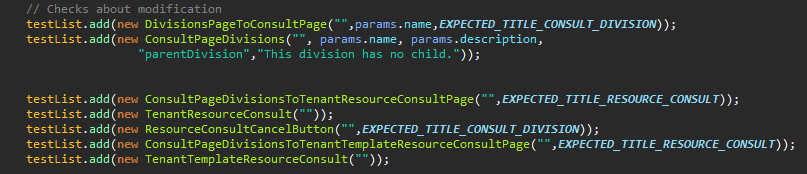
\*\*\*\*\*\*\*\*\*\*\*\* Sub Tests details for TestResultList \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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.



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 ]

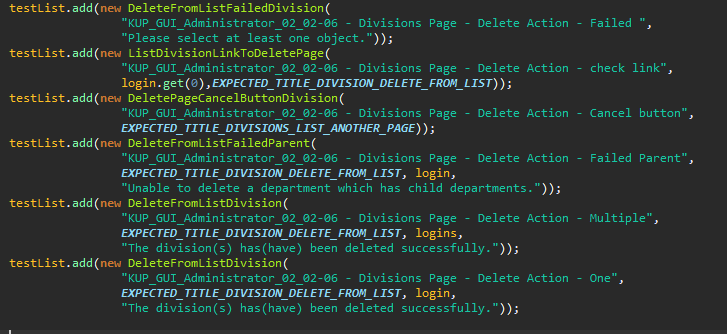
DONE\_OK : : [ is the Component Database application displayed ]

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.



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 exigent 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 écris 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 comprends 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 fais il y a de ça plus d’un mois en Sélénium.

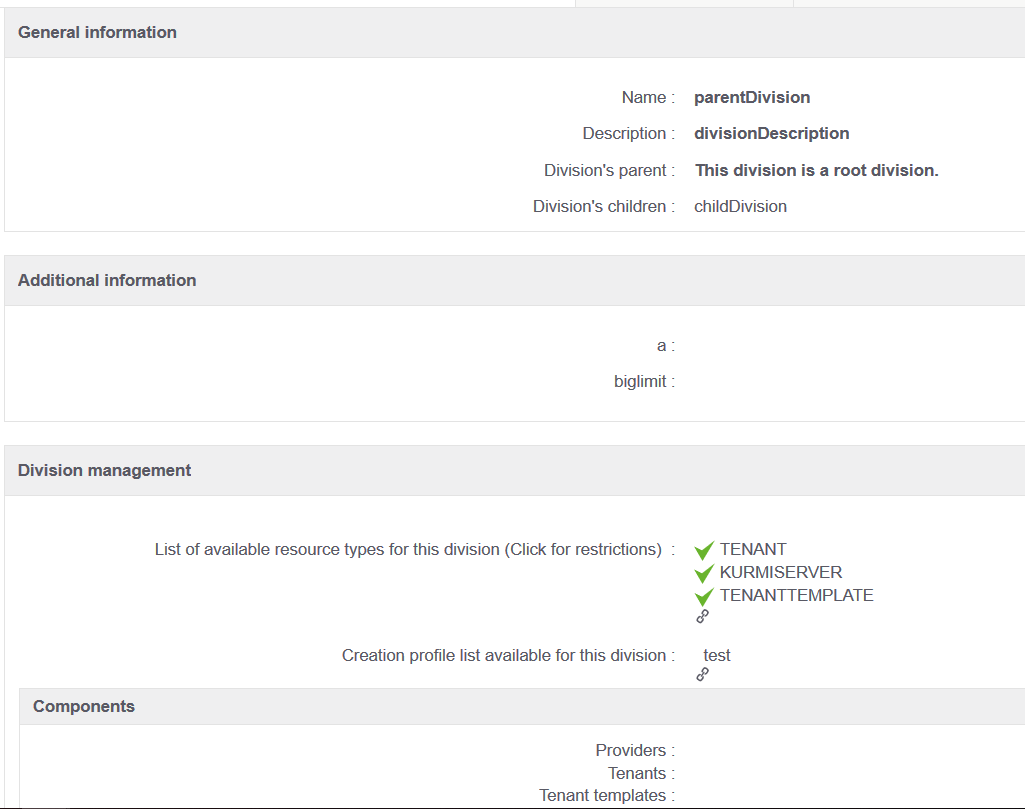
#### Remerciements

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.

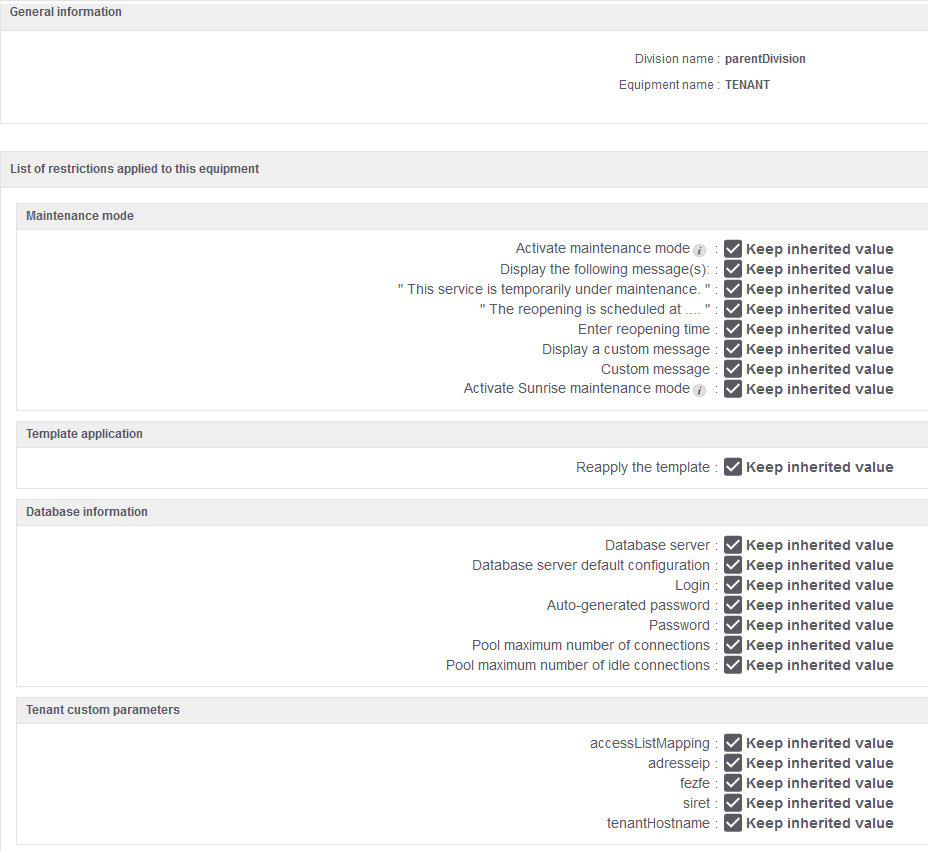
## Annexes

Consultation des divisions

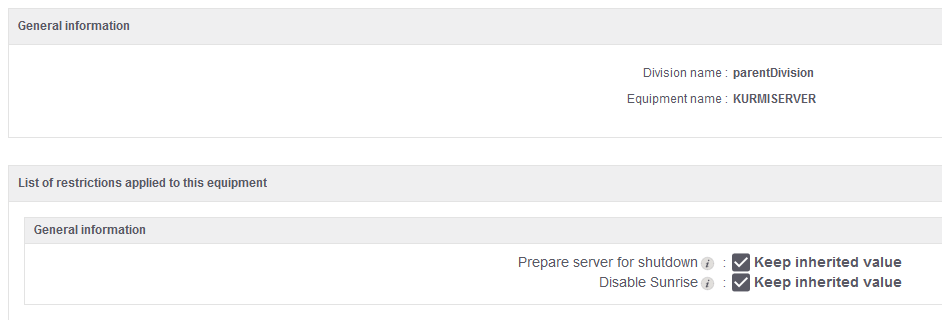


Et les pages liées : TENANT / KURMISERVER / TENANTTEMPLATE

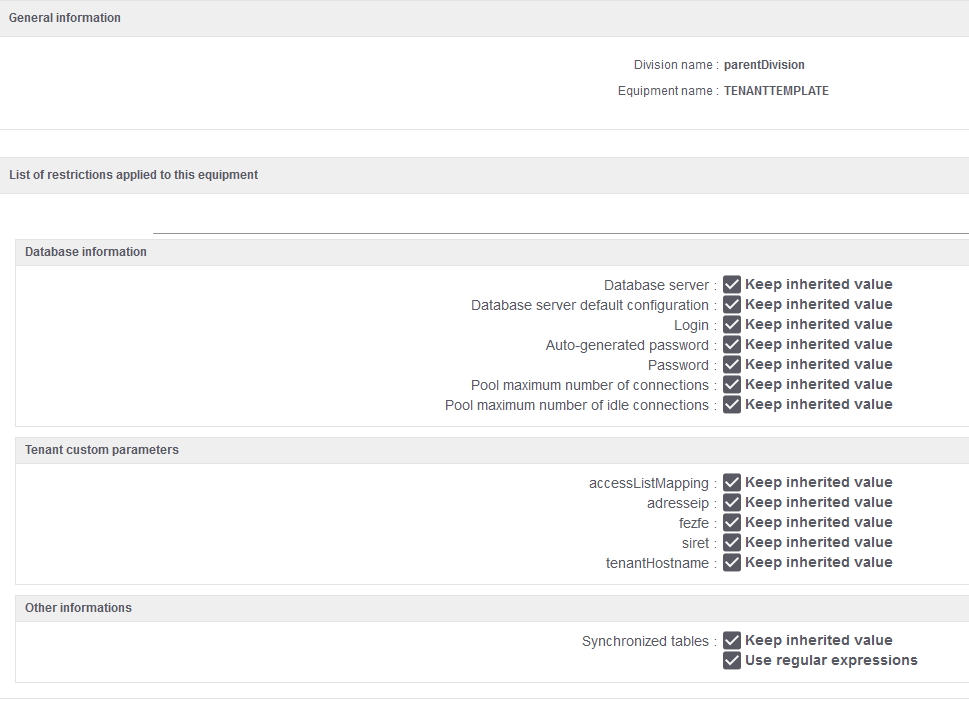
TENANT



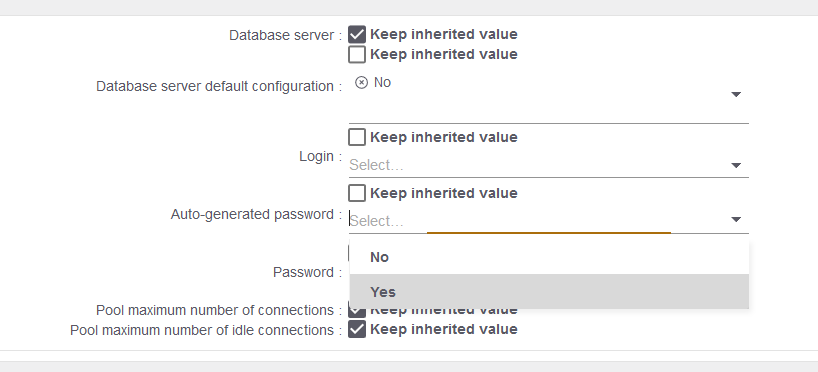
KURMISERVER



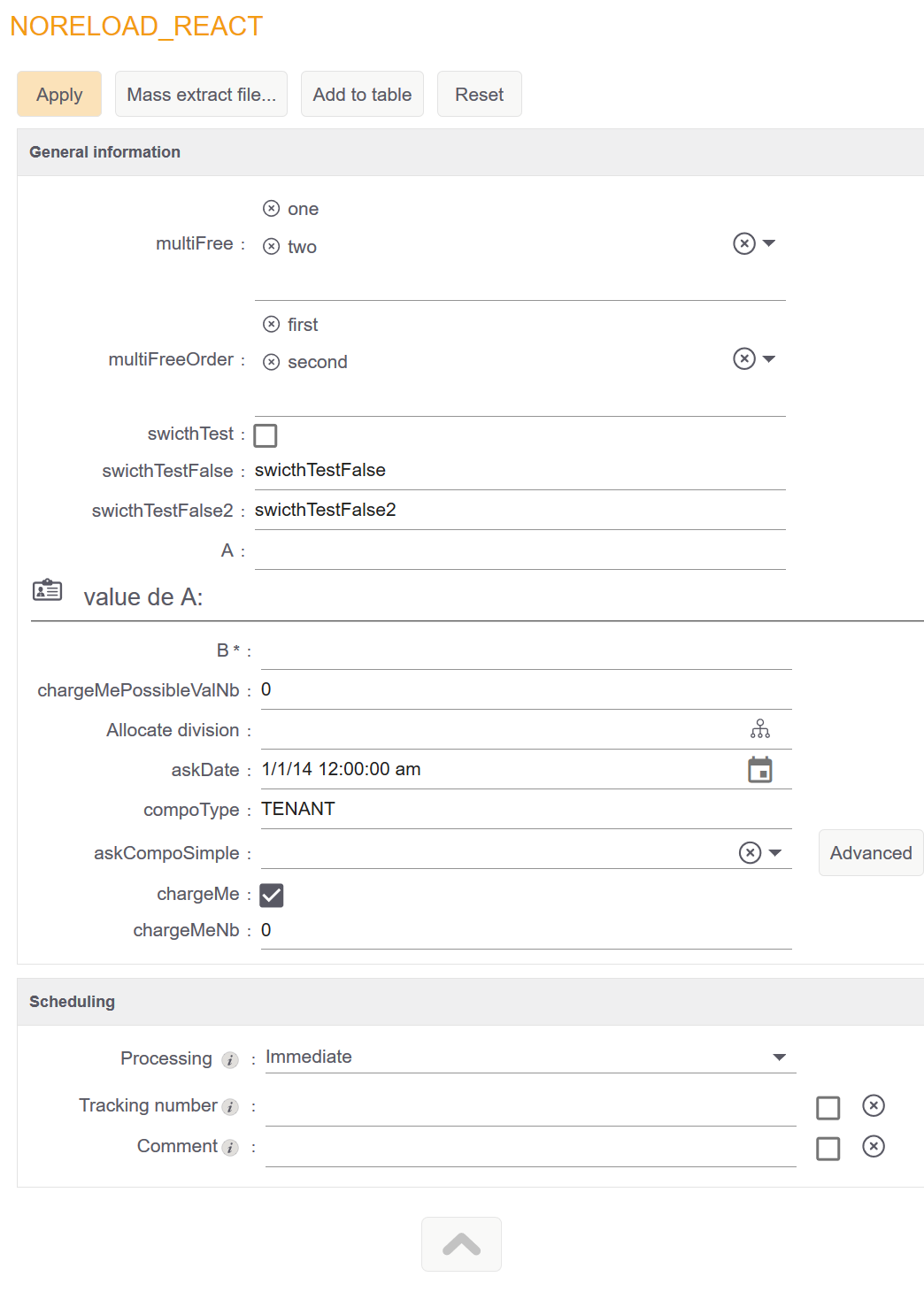
TENANTTEMPLATE



Cases décochées :

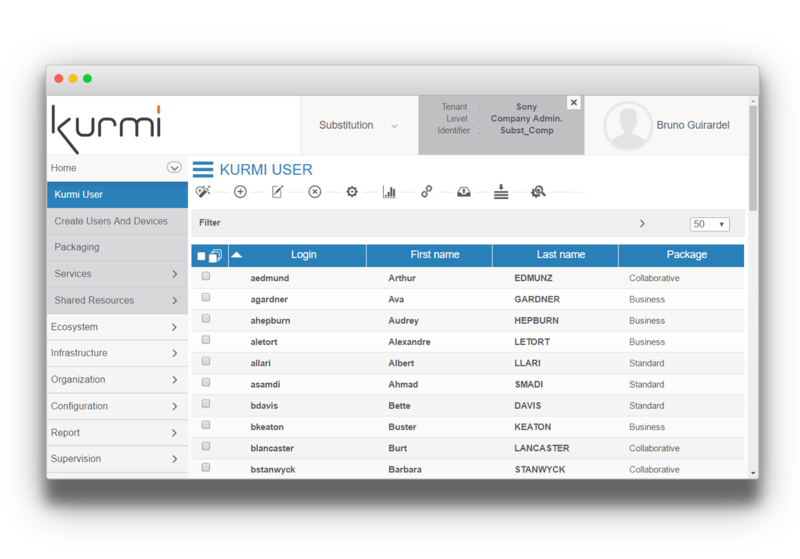


Formulaire React

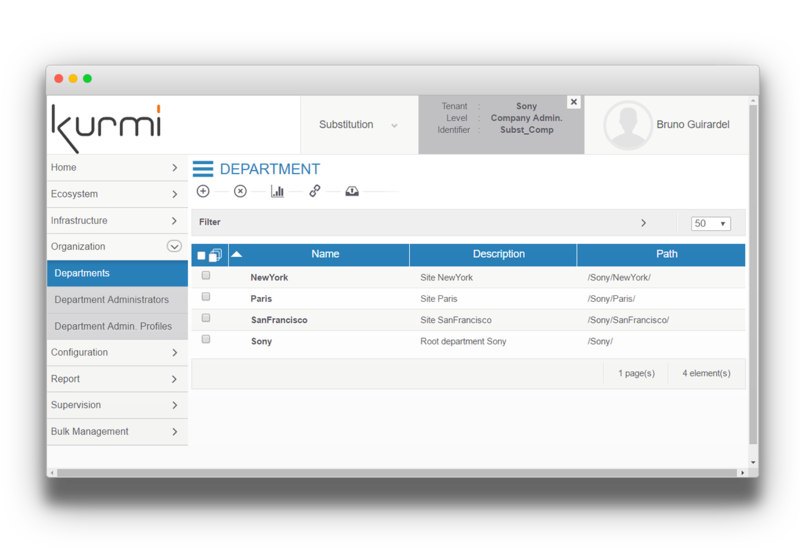


Kurmi Unified Provisionning

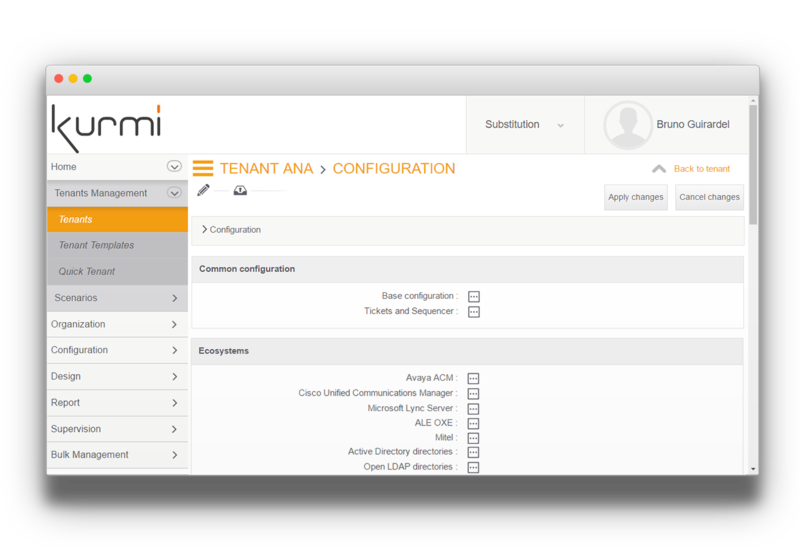
Interface Basique



Departements

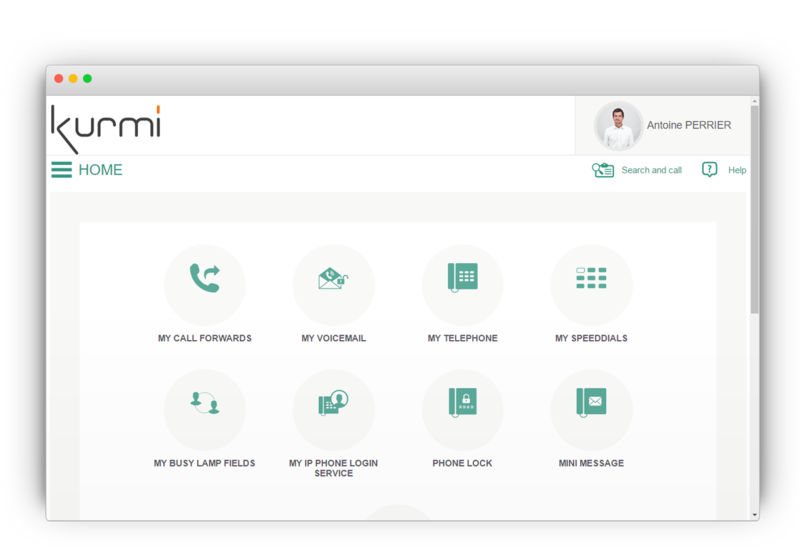


Configuration d’un tenant

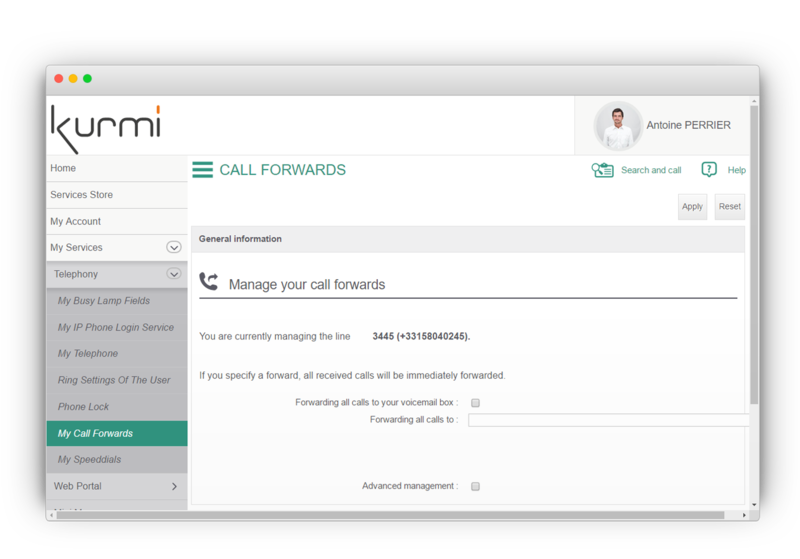


Kurmi Unified Selfcare

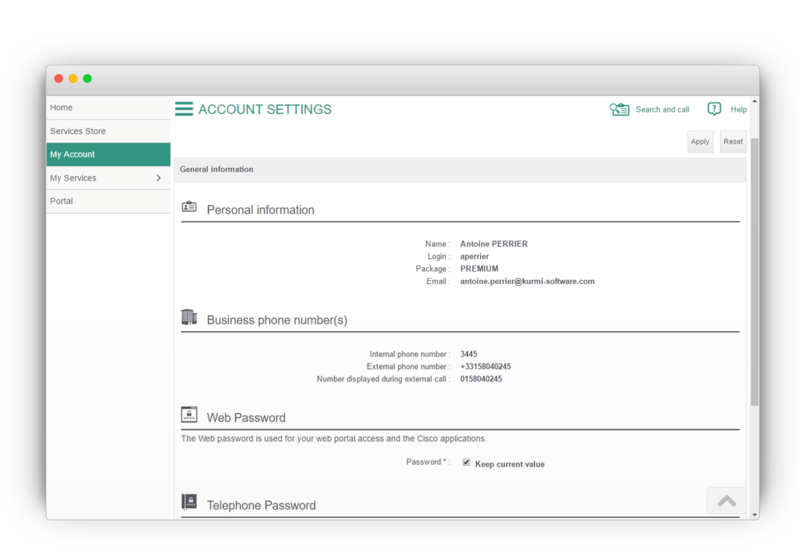
Interface Basique



Call Forward

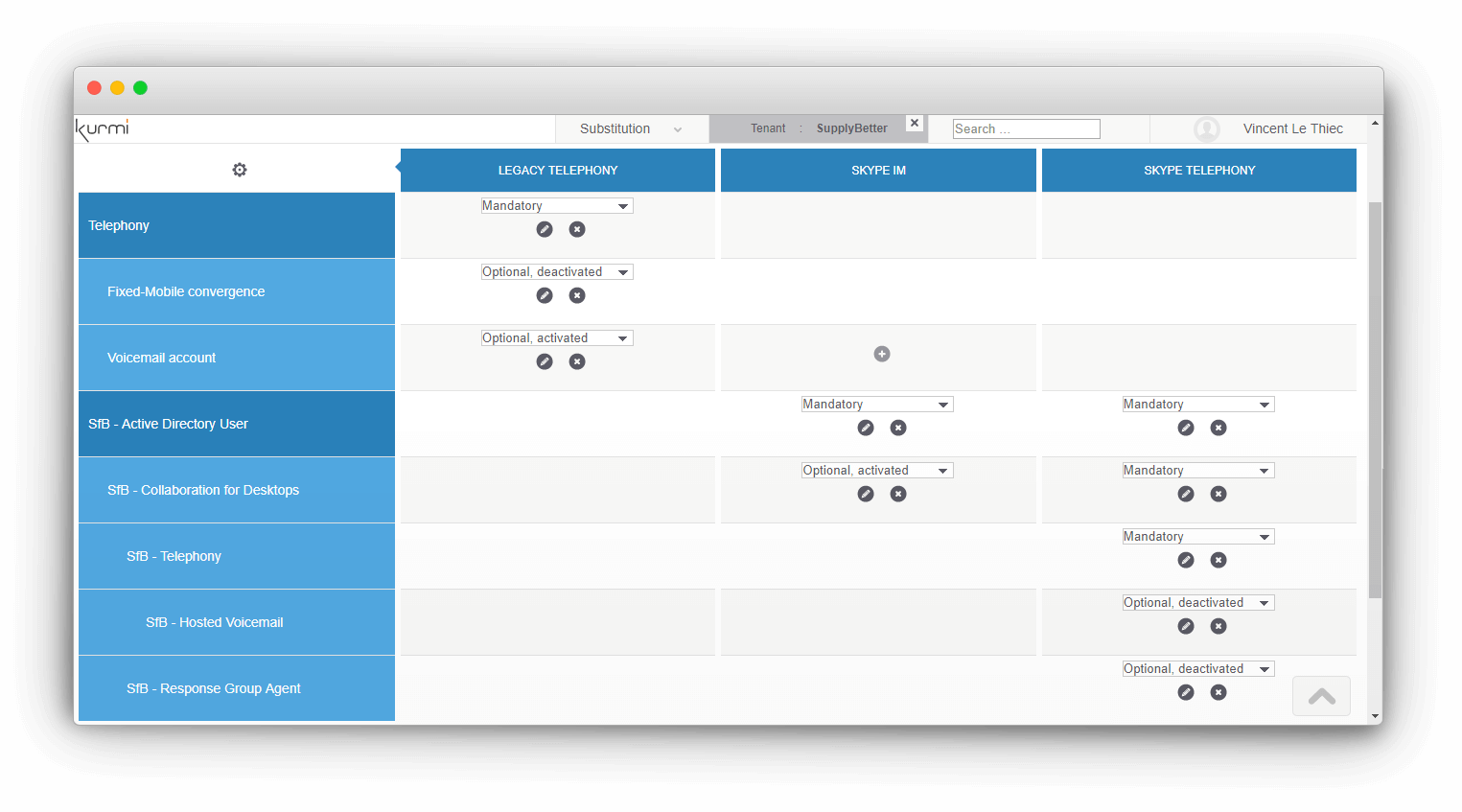


Account Settings

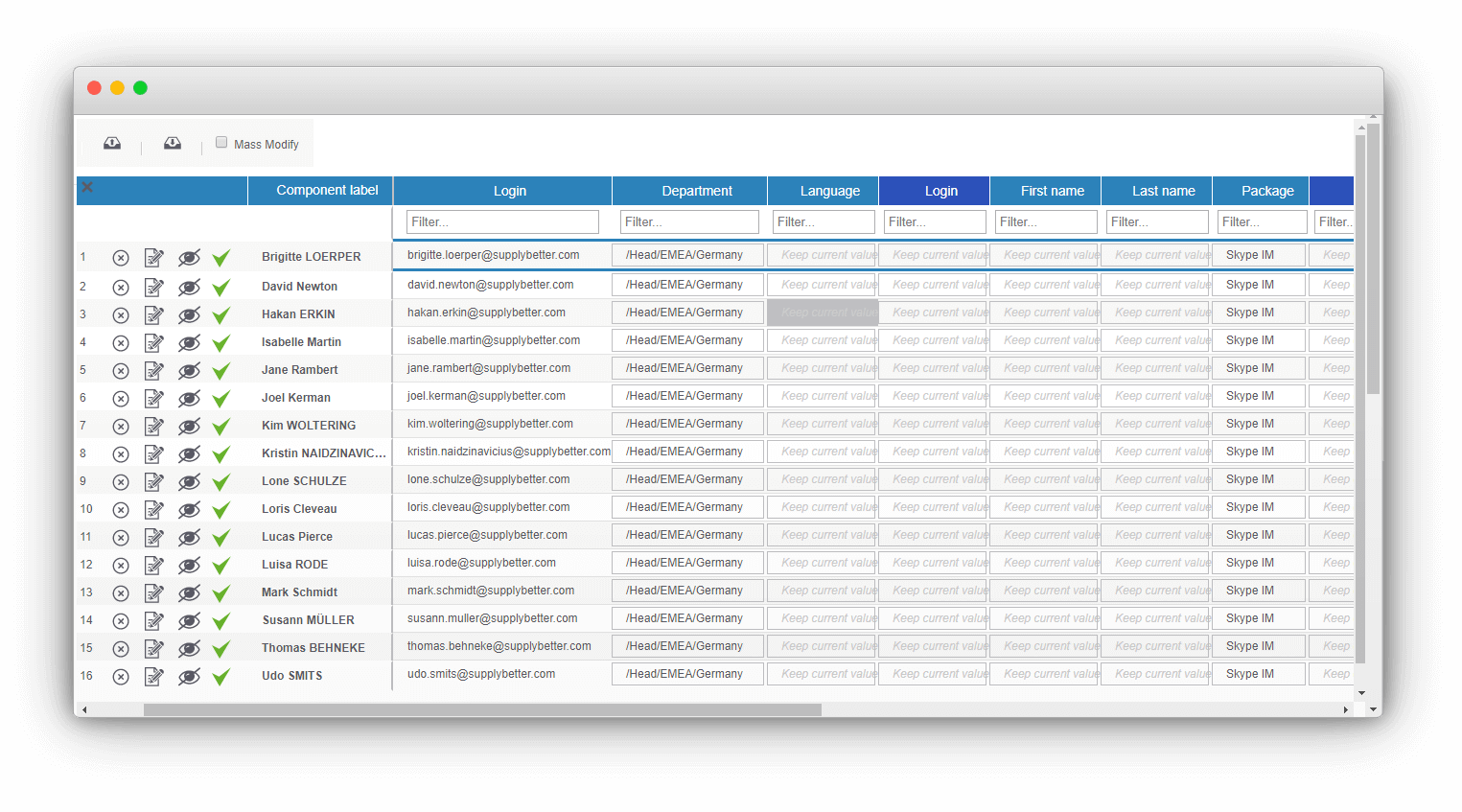


Kurmi Migration Tool

Interface de base



Choisir la durée et la date de migration



Reporter la configuration des futurs collaborateurs après migration

