HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Travail de Bachelor

Automatisation du stock cible des succursales de Digitec Galaxus AG

Non confidentiel

Département : COMEM+

Filière : Ingénierie des médias

Nicolas Meuwly

8 août 2024

Travail proposé par :

Steve Cosandey

Digitec Galaxus AG

Rue du Grand-Pré 2B, 1007 Lausanne

Supervisé par :

Laurent Bolli

Préambule

Ce travail de Bachelor (ci-après TB) est réalisé en fin de cursus d'études, en vue de l'obtention du titre de Bachelor of Science HES-SO en Ingénierie.

En tant que travail académique, son contenu, sans préjuger de sa valeur, n'engage ni la responsabilité de l'auteur, ni celles du jury du travail de Bachelor et de l'Ecole.

Toute utilisation, même partielle, de ce TB doit être faite dans le respect du droit d'auteur.

HEIG-VD

Le Chef du Département

Yverdon-les-Bains, le 08.08.2024

Remerciements

Je souhaiterais remercier tout particulièrement Steve Cosandey, mandant du projet sur le stock cible, pour m'avoir fait confiance pour ce travail et m'avoir accompagné et aidé durant ces 15 semaines.

Merci à Laurent Bolli, Professeur HES et doyen COMEM, pour son temps, son expertise et ses conseils précieux.

Merci aux participants des sessions des tests utilisateurs pour leur temps et leurs retours constructifs.

Merci aux différents collaborateurs de Digitec Galaxus AG contactés avant et pendant le projet pour leur temps et leur aide.

Enfin, merci également aux professeurs et chargés de cours de la HEIG-VD de m'avoir apporté le bagage nécessaire à la réalisation de ce travail de Bachelor durant ces 3 années d'études.

Ce projet n'aurait pas pu voir le jour sans toutes ces personnes. Pour leur temps, leur aide et leur expertise apportés tout au long de ce travail, je leur adresse ma plus sincère et entière gratitude.

Authentification

Le soussigné, Nicolas Meuwly, atteste par la présente avoir réalisé seul ce travail et n'avoir utilisé aucune autre source que celles expressément mentionnées.

N. Meuwly

Nicolas Meuwly

Yverdon-les-Bains, le 08.08.2024

Département : COMEM

Filière : Ingénierie des médias

Étudiant : Nicolas Meuwly

Enseignant responsable : Laurent Bolli

Travail de Bachelor 2023-2024

Résumé publiable

Digitec Galaxus AG, entreprise suisse de commerce électronique fondée en 2001, opère via deux plateformes, digitec.ch et galaxus.ch, offrant un large éventail de produits. Elle possède également 10 succursales en Suisse permettant aux clients de retirer leurs commandes, obtenir des conseils, retourner des produits et recevoir de l'aide. Les succursales disposent d'un stock permanent appelé « stock cible ». Actuellement, la définition de ce stock se fait manuellement, et son enregistrement dans l'ERP de l'entreprise peut être manuel ou automatique mais nécessite des prérequis techniques, rendant le processus inaccessible à tous les employés.

Ce projet vise à automatiser l'ensemble du processus lié au stock cible, de la sélection des produits en fonction de divers critères à leur enregistrement dans l'ERP. L'objectif est de réduire les tâches répétitives et les erreurs humaines, tout en offrant une interface intuitive pour tous les utilisateurs. Le développement de la solution a débuté par une analyse des besoins et des processus actuels. Ensuite, des recherches sur les technologies et outils disponibles ont été menées pour identifier les meilleures options. Enfin, des tests utilisateurs ont été réalisés pour affiner l'interface et garantir une expérience utilisateur optimale.

La solution proposée résulte en une maquette interactive représentant la future solution. Cette maquette permet de gérer le stock cible, de sa définition à son enregistrement dans l'ERP, en proposant une méthode automatique et manuelle de génération de listes de produits à partir de différents critères tout en laissant suffisamment de liberté à l'utilisateur pour gérer ce stock à sa guise. En complément de la maquette, une version préliminaire du développement a été réalisée, incluant la génération de produits. Des recommandations ont été élaborées pour améliorer les résultats futurs et un guide utilisateur a été rédigé pour accompagner l'utilisateur dans son utilisation de la solution.

Étudiant :	Date et lieu :	Signature :
Nicolas Meuwly	07.08.2024, Yverdon-les-Bains	N. Meuwly
Enseignant·e responsable :	Date et lieu :	Signature :
Laurent Bolli	07.08.2024 Yverdon-les-Bains	
Digitec Galaxus AG :	Date et lieu :	Signature :
Steve Cosandey	le 02.08.24 Lausanne	

Table des matières

Préambule I
Remerciements III
Authentification V
Résumé publiable VII
1. Introduction 1
1.1 Contexte et besoins 1
1.2 Objectifs 1
1.3 Méthodologie 1
2. Phase 1 - Conception de la solution 2
2.1 Choix des outils 2
2.1.1 Framework web 2
2.1.2 Dépendances 3
2.2 Architecture de base 4
2.2.1 Initialisation du projet 4
2.2.2 Préparation et fonctionnement de Next.js 6
2.2.3 Interface de test 9
2.3 Récupération des données 9
2.3.1 Requêtes API 10
2.3.2 Liste de produits 10
2.3.3 Liste et quantité de catégories 13
2.3.4 Problèmes rencontrés 19
2.4 Étude du design system 19
2.4.1 Lignes directrices 19
2.4.2 Prise en main du design sur Figma 21
2.5 Élaboration de la maquette 22
2.5.1 Préambule 22

2.5.2 Première ébauche 23
2.5.3 Version intermédiaire 27
2.6 Tests utilisateurs et adaptations 53
2.6.1 Objectifs et déroulement 53
2.6.2 Synthèse des résultats 55
2.6.3 Adaptations à la suite des tests 56
2.7 Répercutions sur les autres phases 77
3. Phase 2 - Développement des fonctionnalités 77
3.1 Fonctionnalité de génération automatique des produits 77
3.1.1 Utilisation du web scraping 77
3.1.2 Accès à la base de données 88
3.1.3 État du développement 90
3.2 Fonctionnalité d'enregistrement auto. des produits dans l'ERP 90
3.2.1 Recherches sur le sujet 90
3.3 Fonctionnalités annexes et tests unitaires 90
3.4 Point de situation sur la faisabilité du projet 91
4. Phase 3 - Intégration des fonctionnalités 91
4.1 Développement de l'interface 91
4.1.1 Initialisation du projet Next.js 91
4.1.2 Navigation dans l'application 92
4.1.3 Composants 93
4.1.4 Fonctionnalités 100
4.2 Structure des données 101
4.2.1 Préparation en amont 101
4.2.2 Intégration dans le développement 103
5. Phase 4 - Tests et optimisations 109
6. Phase 5 - Finalisation et recommandations 110
7. Présentation de la solution 111
8. Gestion du projet 111
8.1 Planification initiale 111
8.2 Déroulement du travail 112
8.2.1 Problèmes rencontrés 112
8.2.2 Planification effective du projet 112
9. Conclusion 113
Bibliographie 114
Annexes 121

Table des figures

Figure 1 - Processus d'initialisation complet du projet Next.js 5
Figure 2 - Structure des fichiers du projet Next.js 6
Figure 3 - Page de démarrage par défaut d'un projet Next.js créé avec l'assistant d'initialisation 7
Figure 4 - Interface de tests permettant de récupérer une liste de produits et une quantité de catégories 9
Figure 5 - Structure des fichiers pour accéder aux points de terminaison de l'API 10
Figure 6 - Liste des téléphones portables sur digitec.ch illustrant le conteneur des produits 11
Figure 7 - Interface de tests avec les résultats de la requête de récupération des produits 13
Figure 8 - Emplacement des catégories principales sur galaxus.ch 16
Figure 9 - Interface de tests avec les résultats de la requête de récupération des catégories 19
Figure 10 - Page d'accueil du site dédié au design system de Digitec Galaxus AG 20
Figure 11 - Principes de conception UX de digitec.ch et galaxus.ch 20
Figure 13 - Description sur l'utilisation des boutons sur digitec.ch et galaxus.ch 21
Figure 12 - Présentations des icônes utilisées sur digitec.ch et galaxus.ch 21
Figure 14 - Recherche de composants depuis Figma 22
Figure 15 - Composant de chronologie utilisé pour indiquer les étapes d'un processus du design system 23
Figure 16 - Entête de la solution du stock cible (version initiale) 23
Figure 17 - Entête de galaxus.ch en thème clair 23
Figure 18 - Entête du site dédié au design system de Digitec Galaxus AG 24
Figure 19 - Pied de page de la solution du stock cible (version initiale) 24
Figure 20 - Pied de page de galaxus.ch en thème clair 24
Figure 21 - Version initiale de la maquette avec la 1re étape de génération du stock cible 25
Figure 22 - Version initiale de la maquette avec la 2e étape de génération du stock cible 26
Figure 23 - Version initiale de la maquette avec la 4e étape de génération du stock cible 27
Figure 24 - Page de connexion 28
Figure 25 - Page du tableau de bord de la maquette 29
Figure 26 - Page de sélection du mode de création du nouveau stock cible 30
Figure 27 - Illustration Digitec représentant les avis/retours des clients 31
Figure 28 - Icônes existantes et nouvellement créées pour l'illustration animée 31
Figure 29 - Ordre d'apparition des images dans l'illustration animée 32
Figure 30 - Liste de produits vide lors de la sélection manuelle 32
Figure 31 - Chargement de la liste de produits lors de la sélection automatique 33
Figure 32 - Début de la liste de produits lors de la sélection automatique 34
Figure 33 - Fin de la liste de produits lors de la sélection automatique 35
Figure 34 - Vue des produits avec la mise en page définie sur "liste" 36
Figure 35 - Fenêtre contextuelle des réglages du stock cible 37
Figure 37 - Vue de la liste de produits lorsque plusieurs d'entre eux sont sélectionnés 38

Figure 36 - Vue de la liste de produits avec la sélection multiple activée....38
Figure 38 - Message temporaire informant la suppression de plusieurs produits 39
Figure 39 - Fenêtre contextuelle des réglages d'une sélection de produits 40
Figure 40 - Fenêtre contextuelle permettant l'ajout de produit à la liste....41
Figure 41 - Message de confirmation d'ajout d'un nouveau produit 42
Figure 42 - Affichage des options individuelles lors du survol d'un produit 42
Figure 43 - Message temporaire informant la suppression d'un produit 43
Figure 44 - Fenêtre contextuelle des réglages d'un produit 43
Figure 45 - Fenêtre contextuelle permettant l'accès à la page d'un produit sur Galaxus ou l'ERP 44
Figure 46 - Page résumant les détails du stock cible avant son enregistrement dans l'ERP....45
Figure 47 - Page de chargement lors de l'enregistrement des produits dans l'ERP 45
Figure 49 - Ordre d'apparition des images dans l'illustration animée 46
Figure 48 - Illustration Digitec représentant la rubrique communauté 46
Figure 50 - Message de succès lors de la réussite de l'enregistrement des produits dans l'ERP 47
Figure 51 - Message d'erreur à la suite d'un problème survenu lors de l'enregistrement des produits dans l'ERP .47
Figure 52 - Page listant les produits du stock cible actuel 48
Figure 53 - Réglage de la succursale et de la langue de la solution 48
Figure 54 - Page des paramètres généraux de l'application 49
Figure 55 - Liste des critères de sélection disponible pour le mode automatique 49
Figure 56 - Liste des catégories de sélection disponible pour le mode automatique....50
Figure 57 - Menu d'utilisateur....50
Figure 59 - Fenêtre contextuelle permettant l'ajout d'un utilisateur 51
Figure 58 - Page de gestion de l'accès à l'application 51
Figure 60 - Nouvelles icônes créées pour la solution....52
Figure 61 - Vue d'ensemble des pages et composants sur Figma avec les relations....52
Figure 62 - Composant de la tuile de produit sur Figma 53
Figure 63- Liste déroulante de sélection de la succursale....57
Figure 64 - Liste déroulante de sélection de la langue....57
Figure 65 - Menu d'utilisateur....57
Figure 66 - Page d'historique des modifications du stock cible 58
Figure 67 - Page de gestion des accès de l'application (version pour administrateur)....59
Figure 68 - Fenêtre contextuelle permettant l'ajout d'un nouvel utilisateur 59
Figure 69 - Page du tableau de bord 60
Figure 70 - Fenêtre contextuelle contenant les détails des produits dans le classement des meilleures ventes 61
Figure 71 - Message informatif sur la signification des différents statuts d'étape de processus 62
Figure 72 - Infobulle indiquant la 1re étape du processus de création du stock cible....62
Figure 73 - Infobulle indiquant la 2e étape du processus de création du stock cible 62
Figure 74 - Infobulle indiquant la 3e étape du processus de création du stock cible 62
Figure 76 - Critères de sélection pour les produits générés par le système 63
Figure 75 - Fenêtre contextuelle de paramétrage du mode automatique pour la création du stock cible....63
Figure 77 - Page présentant les produits générés par le mode automatique....64
Figure 78 - Infobulle indiquant d'établir les règles du stock cible avant de continuer le processus 64
Figure 79 - Fin de la liste de produits avec le système de pagination....65
Figure 80 - Liste déroulante permettant le tri des produits....65
Figure 81 - Fenêtre contextuelle permettant la définition des paramètres et règles du nouveau stock cible....66
Figure 82 - Fenêtre contextuelle permettant l'ajout d'un produit....67
Figure 83 - Illustration du fait de pouvoir ajouter plusieurs produits à la suite....67
Figure 84 - Sélection de la quantité lors de l'ajout d'un produit....68
Figure 85 - Illustration du fait de pouvoir ajouter plusieurs produits à la suite....68
Figure 86 - Message de confirmation d'ajout de produits....68
Figure 90 - Champ de formulaire pour la saisie de la quantité d'un produit dans le panier sur digitec.ch 69
Figure 89 - Liste déroulante de sélection de la quantité d'un produit dans le panier sur digitec.ch....69

Figure 87 - Liste déroulante de sélection de la quantité d'un produit 69
Figure 88 - Champ de formulaire pour la saisie de la quantité d'un produit 69
Figure 91 - Fenêtre contextuelle permettant la saisie d'une quantité pour une sélection de produits 70
Figure 92 - Page du résumé des informations avant l'enregistrement des produits dans l'ERP 71
Figure 93 - Page de chargement lors de l'enregistrement des produits dans l'ERP 71
Figure 94 - Page listant les différents stocks actifs et non actifs (version administrateur) 72
Figure 95 - Fenêtre contextuelle permettant la duplication d'une liste (version administrateur) 73
Figure 96 - Page lisant les différents stocks actifs et non actifs 73
Figure 97 - Fenêtre contextuelle permettant la duplication d'une liste 74
Figure 98 - Page d'une liste actuelle de produits du stock cible 74
Figure 99 - Fenêtre contextuelle permettant la redéfinition des paramètres et règles d'une liste du stock actuel .75
Figure 100 - Mise en page d'un produit lorsque celui-ci est "EOL" 75
Figure 101 - Mise en page lors du survol d'un produit "EOL" 76
Figure 102 - Liste déroulante permettant de trier la liste de produits et d'afficher les produits "EOL" en premier .76
Figure 103 - Page d'un produit sur galaxus.ch illustrant les informations de disponibilité de ce dernier 81
Figure 104 - Page d'une catégorie illustrant le total de produits, les filtres et l'indicateur de disponibilité 85
Figure 105 - Page d'une catégorie sur galaxus.ch illustrant l'emplacement du nombre de produits affichés 85
Figure 106 - Interface de tests avec les résultats de la requête de récupération des produits avec 1 catégorie 86
Figure 107 - Interface de tests avec les résultats de la requête de récupération des produits avec 2 catégories 86
Figure 108 - Interface de tests avec les résultats de la requête de récupération des produits avec 3 catégories 87
Figure 109 - Interface de tests avec les résultats de la requête de récupération des produits avec 4 catégories 87
Figure 110 - Interface de tests avec les résultats de la requête de récupération des produits avec 5 catégories 87
Figure 111 - Capture d'écran de l'arborescence des fichiers pour les pages du site 92
Figure 112 - Capture d'écran de l'arborescence des fichiers pour les composants du site 93
Figure 115 - Représentation du composant de bouton en tuile 94
Figure 114 - Représentation du composant de bouton secondaire 94
Figure 113 - Représentation du composant de bouton principal 94
Figure 116 - Représentation du composant de structure du contenu sur les pages 95
Figure 117 - Représentation des composants de bouton d'icône principal et secondaire avec texte 96
Figure 118 - Représentation du composant de bouton d'icône secondaire sans texte 96
Figure 119 - Représentation du composant de bouton d'icône de compte 96
Figure 121 - Représentation du composant de liste déroulante (ouverte) 97
Figure 120 - Représentation du composant de liste déroulante (fermée) 97
Figure 122 - Représentation du composant de menu de sélection de la succursale et de la langue 98
Figure 123 - Représentation du composant de menu d'utilisateur 98
Figure 124 - Représentation du composant de l'indicateur de processus 99
Figure 125 - Représentation du composant de tuile de produit (vue en cases) 99
Figure 126 - Représentation du composant de tuile de produit (vue en liste) 100
Figure 127 - Représentation du composant de la barre d'outils 100
Figure 128 - Aperçu des données stockées dans le stockage local du navigateur 101
Figure 129 - Diagramme de la base de données de la solution du stock cible 103
Figure 130 - Capture d'écran de l'emplacement du fichier de la base de données 107
Figure 131 - Capture d'écran de l'organisation de l'espace de travail sur Figma 110
Figure 132 - Planification initiale du projet au format Gantt Chart 112
Figure 133 - Planification effective du projet au format Gantt Chart 113

Liste des tableaux

Tableau 1 - Liste des scénarios et des tâches du test utilisateurs....54
Tableau 2 - Liste des essais de récupération de produits à partir d'un nombre défini de catégories....86
Tableau 3 – Exemple d'une table de base de données listant des produits....88
Tableau 4 – Exemple d'une table de base de données listant le niveau de stock d'un produit dans un entrepôt....89
Tableau 5 – Structure du résultat de la requête SQL d'exemple....89
Tableau 6 - Liste des pages et de leur URL....93

Liste des codes sources

Listing 1 - Balise HTML pour un titre de niveau 1, le plus important dans la hiérarchie des titres 3
Listing 2 - Exemple de comment styliser un titre de niveau 1 en CSS....3
Listing 3 - Exemple d'application de styles avec Tailwind CSS directement dans une balise HTML 3
Listing 4 - Commande pour initialiser un nouveau projet avec Next.js 4
Listing 5 - Questions posées lors de l'initialisation d'un projet Next.js....4
Listing 6 - Commande utilisée pour démarrer un serveur de développement local de l'application Next.js....6
Listing 7 - Exemple de fonction Home retournant un élément HTML qui représente la page d'accueil....7
Listing 8 - Code pour le fichier « layout.tsx », définissant la structure de base de chaque page du site....8
Listing 9 - Code CSS global incluant les styles de base, les composants et les utilitaires de Tailwind CSS 8
Listing 10 - Code de la route API permettant de récupérer une liste de produits 11
Listing 11 - Code de la fonction permettant de récupérer une liste de produit à partir d'une page web....12
Listing 12 - Code de la fonction permettant de récupérer les données de produits via la route API dédiée....13
Listing 13 - Code de la route API permettant de récupérer la liste des catégories présentes sur galaxus.ch 15
Listing 14 - Code de la fonction permettant de récupérer une liste de catégories à partir d'une page web 17
Listing 15 - Code de la fonction transmettant une page web à scrapeCategories() via une URL 17
Listing 16 - Code de la fonction récupérant et regroupant les sous-catégories dans un tableau....18
Listing 17 - Code de la fonction permettant de récupérer les données de catégories via la route API dédiée....18
Listing 18 - Code de la fonction permettant de récupérer les détails des produits d'une catégorie sur galaxus.ch .78
Listing 19 - Code de la fonction récupérant les détails des produits d'une liste de catégories sur galaxus.ch 79
Listing 20 - Code de la fonction permettant de vérifier si un produit peut être inclus dans le stock cible 80
Listing 21 - Code définissant le type "Product"....81
Listing 22 - Code de la fonction parcourant tous les produits et vérifiant leur inclusion dans le stock cible 82
Listing 23 - Code de la fonction exécutant la récupération et la vérification des produits 83
Listing 24 - Code de la route API permettant de récupérer une liste de produits 83
Listing 25 - Code de la fonction permettant de récupérer les données de produits via la route API dédiée....84
Listing 26 - Code illustrant le découpage d'un tableau 84
Listing 27 - Exemple de requête SQL récupérant une liste de produits en stock dans un entrepôt particulier 89
Listing 28 - Code d'une icône SVG dans le composant IconsSVG....97
Listing 29 - Code du tableau d'icônes utilisé pour l'indicateur de processus 99
Listing 30 - Exemple de code permettant de récupérer tous les produits d'une table avec Prisma 104
Listing 31 - Exemple de code d'une requête SQL permettant de récupérer tous les produits d'une table....104
Listing 32 - Code du schéma Prisma décrivant la structure de la base de données 107
Listing 33 - Commandes pour utiliser Prisma et configurer le fichier de base de données....107
Listing 34 - Code du "seeder" permettant de peupler la base de données 108
Listing 35 - Commande permettant d'exécuter le "seeder" 108

1. Introduction

1.1 Contexte et besoins

Digitec Galaxus AG, fondée en 2001, est une entreprise suisse de commerce électronique renommée, opérant via digitec.ch et galaxus.ch. Ces plateformes offrent une vaste gamme de produits, allant des articles électroniques aux biens de consommation courante. En complément de ses plateformes en ligne, Digitec Galaxus AG possède dix succursales physiques en Suisse, où les clients peuvent retirer leurs commandes, obtenir des conseils, retourner des produits et bénéficier de divers services d'assistance.

Chaque succursale dispose d'un stock permanent appelé « stock cible ». Ce stock cible est un assortiment de produits essentiels et de nouveautés qui sont régulièrement renouvelés pour garantir la disponibilité des produits clés. Ce stock est important, car il augmente l'attrait des succursales. Les clients peuvent non seulement venir récupérer leurs commandes, mais aussi acheter directement sur place et retirer leurs achats le jour même.

Actuellement, la gestion du stock cible repose sur des processus manuels. Les employés doivent sélectionner manuellement les produits à inclure dans le stock cible en se basant sur divers critères comme les ventes passées, les tendances du marché et les prévisions de demande. Une fois les produits sélectionnés, ils doivent être enregistrés dans l'ERP[1] de l'entreprise. Ce processus est non seulement chronophage et répétitif, mais il nécessite également des compétences techniques pour automatiser certaines parties du processus, rendant la tâche complexe pour les employés non spécialisés. De plus, la nature manuelle de la gestion actuelle introduit un risque accru d'erreurs humaines, ce qui peut affecter l'efficacité et la précision de la gestion des stocks. Il y a donc une réelle plus-value ici à créer en trouvant une solution qui puisse automatiser ce processus pour améliorer l'efficacité, réduire les erreurs humaines et libérer les employés de tâches répétitives.

1.2 Objectifs

L'objectif principal de ce travail de Bachelor est de concevoir une solution automatisée pour la gestion du stock cible. Cette solution vise à simplifier et optimiser l'ensemble du processus, de la sélection des produits à leur enregistrement dans l'ERP. Les objectifs spécifiques incluent :

1. Réduire les tâches répétitives et minimiser les erreurs humaines.
2. Offrir une interface utilisateur intuitive et accessible à tous les employés, indépendamment de leurs compétences techniques.
3. Automatiser la sélection des produits en fonction de critères prédéfinis tels que les ventes, les nouveautés et les produits saisonniers.
4. Assurer une mise à jour dynamique et précise des stocks dans l'ERP.

1.3 Méthodologie

L'approche adoptée pour ce travail de Bachelor s'inspire des méthodes agiles et des enseignements acquis en gestion de projet. Le projet sera structuré en plusieurs phases, chacune ayant des objectifs et des livrables spécifiques.

1. **Analyse des besoins** : Cette première étape consiste à comprendre les exigences spécifiques des utilisateurs finaux et les contraintes techniques. Une analyse approfondie des besoins et des processus existants sera réalisée pour identifier les défis et les opportunités. Cette étape a déjà été réalisée en amont durant la pré-étude.

[1] ERP : Système de gestion intégré utilisé pour gérer et automatiser les processus d'affaires.

2. **Recherche technologique** : Une recherche approfondie sur les technologies disponibles sera menée. Cette recherche inclura l'exploration d'outils de web scraping tels que Puppeteer et Cheerio, ainsi que des frameworks de développement web tels que Next.js.
3. **Conception de la maquette** : Une première version de l'interface sera créée avec Figma. Cette maquette interactive permettra de simuler la gestion du stock cible depuis sa définition jusqu'à son enregistrement dans l'ERP. Elle servira également de base pour les tests utilisateurs.
4. **Tests utilisateurs** : Des tests utilisateurs seront effectués pour affiner l'interface et garantir une expérience utilisateur optimale. Les retours des utilisateurs seront utilisés pour ajuster et améliorer la maquette.
5. **Développement préliminaire** : Une première phase de développement sera réalisée avec Next.js, permettant notamment de récupérer dynamiquement une liste de produits. Ce développement préliminaire permettra de tester la faisabilité technique et de valider les choix technologiques.
6. **Documentation et recommandations** : Une série de recommandations sera rédigée pour guider les développements futurs. Un guide utilisateur sera également créé pour accompagner les utilisateurs dans l'utilisation de la solution.

Chaque section de ce rapport détaillera ces étapes, fournissant une vue d'ensemble complète du processus de développement et des solutions proposées pour atteindre les objectifs du projet.

2. Phase 1 - Conception de la solution

2.1 Choix des outils

2.1.1 Framework web

À la suite de la phase de pré-étude, il a été décidé de concevoir la solution sous forme d'application web pour plusieurs raisons. La première est liée aux connaissances techniques disponibles : il était important d'utiliser des outils familiers afin de proposer au minimum un prototype du concept général de l'application. Les technologies web étant un des piliers de la formation en Ingénierie des Médias, le choix de ces technologies s'est imposé naturellement. La seconde raison est une question d'accessibilité. Il est essentiel que la solution soit facilement accessible pour les utilisateurs sans nécessiter d'installation ou autres prérequis techniques. Un site web peut être consulté depuis une multitude d'appareils, ne nécessitant qu'un navigateur et une connexion internet.

La base d'un site web repose sur une combinaison de trois technologies : HTML, CSS et JavaScript. Chacune de ces technologies a un rôle précis. HTML est un langage de balisage qui structure le site web en définissant les éléments qui le composent, tel que les titres, les paragraphes, les images ou encore les liens. CSS, quant à lui, décrit la présentation visuelle des pages web, en définissant des aspects tels que la police du texte, les couleurs ou encore les effets de survol. Enfin, JavaScript apporte l'interactivité au site, permettant par exemple d'exécuter du code lors du clic sur un bouton ou de récupérer des données via des requêtes envoyées à d'autres sites.

De nos jours, il est rare de voir un site web utiliser uniquement ces trois technologies de base. Il est de plus en plus courant de recourir à des frameworks web pour accélérer et simplifier le processus de développement. Un framework est une sorte de boîte à outils qui intègre plusieurs technologies et propose des fonctions préconstruites que les développeurs peuvent utiliser directement. Cela peut inclure des systèmes d'authentification ou une gestion optimisée de la navigation entre les pages du site.

Digitec Galaxus AG utilise un framework web pour le développement de ses plateformes de vente en ligne. Une rapide analyse sur le site wappalyzer.com en entrant l'URL galaxus.ch permet de découvrir les technologies utilisées. Comme indiqué dans le rapport de la pré-étude, l'entreprise utilise Next.js comme framework web pour ses sites e-commerce. Next.js est un framework basé sur React, une librairie JavaScript qui permet la création et la réutilisation de composants. Un des principaux avantages de ce framework est sa capacité à gérer à la fois le back-end et le front-end, s'occupant des opérations serveur (traitement de données) et de la partie visuelle (interface

utilisateur). Cela est particulièrement utile pour ce projet, car il permet de traiter les données, générer la liste de produits et les enregistrer dans l'ERP de l'entreprise, tout en offrant une interface utilisateur pour effectuer les actions nécessaires à la gestion du stock cible. De plus, ce framework a été utilisé à plusieurs reprises à titre personnel, ce qui constitue un avantage pour éviter de perdre du temps à apprendre un nouveau framework. Pour toutes ces raisons, le choix s'est porté sur Next.js comme base pour la solution.

2.1.2 Dépendances

Bien que Next.js soit très complet, plusieurs dépendances supplémentaires seront nécessaires pour mener à bien ce projet. N'ayant pas accès aux bases de données à ce stade, l'utilisation du web scraping a été privilégiée pour récupérer et générer une liste de produits. Cette méthode permet d'accéder à un site web et d'en extraire les informations souhaitées. Dans notre cas, il s'agira d'ouvrir une page contenant des produits sur galaxus.ch et de récupérer un certain nombre de ces derniers.

Après avoir étudié la faisabilité technique lors de la pré-étude, deux librairies JavaScript ont été retenues pour le web scraping : Puppeteer et Cheerio. Puppeteer permet d'automatiser des actions dans le navigateur, tandis que Cheerio manipule les éléments HTML. Avec Puppeteer, nous pourrons ouvrir des pages sur le site et utiliser Cheerio pour extraire les produits de ces pages. Puppeteer sera également utile pour automatiser l'enregistrement des produits dans l'ERP, une tâche qui nécessite normalement d'ouvrir chaque page de produit et de remplir des champs de formulaire pour intégrer les produits dans le stock cible. Des extensions pour Puppeteer existent, mais elles seront abordées dans la section sur la récupération des données plus loin dans ce chapitre.

Une autre dépendance à inclure est Tailwind CSS. Ce framework CSS, comme d'autres frameworks, offre une multitude d'outils pour accélérer et optimiser l'expérience de développement. Lors de la création du projet Next.js, l'assistant d'initialisation propose d'inclure Tailwind CSS. Cet outil simplifie la mise en forme des éléments sur une page web. Bien que le CSS puisse être rédigé dans une balise spécifique dans un fichier HTML, il est plus courant de voir ce code dans un fichier séparé. Ce qui représenterait plusieurs lignes dans un fichier CSS peut se réduire à une seule ligne avec Tailwind CSS. En effet, au lieu de définir nos styles dans un fichier séparé, il est possible de les inclure directement dans une balise HTML via l'attribut "class". Par exemple, voici comment ajouter du style à un titre de niveau 1 en CSS.

```
<h1>Titre 1</h1> <!-- Cette balise correspond au titre de niveau 1 en HTML -->
```

Listing 1 *- Balise HTML pour un titre de niveau 1, le plus important dans la hiérarchie des titres*

```
/* Le code ci-dessous permet de styliser les titres de niveau 1 */
h1 {
    color: red; /* Couleur du texte */
    font-size: 20px; /* Taille de la police */
    font-weight: bold; /* Niveau de graisse de la police */
}
```

Listing 2 *- Exemple de comment styliser un titre de niveau 1 en CSS*

Bien que ce code ne soit pas conséquent en soi, l'utilisation de Tailwind CSS permettra d'accélérer le processus et de faciliter la maintenance du code. Plus besoin de faire des allers-retours entre les fichiers CSS et HTML, tout est à portée de main. Voici la reproduction de l'exemple précédent, mais en utilisant Tailwind CSS cette fois-ci :

```
<!-- Les éléments de style sont séparés par un espace dans l'attribut "class" -->
<h1 class="text-red-600 text-xl font-bold">Titre 1</h1>
```

Listing 3 *- Exemple d'application de styles avec Tailwind CSS directement dans une balise HTML*

Comme Tailwind CSS est proposé par défaut lors de l'initialisation d'un projet Next.js et pour les raisons citées précédemment, il a été décidé d'inclure Tailwind CSS dans le développement de la solution pour ce Travail de Bachelor.

2.2 Architecture de base

2.2.1 Initialisation du projet

Avant de pouvoir commencer le développement de la solution, il faut initialiser le projet dans un éditeur de code. Celui utilisé dans le cadre de ce Travail de Bachelor se nomme Visual Studio Code. Développé par Microsoft, cet éditeur permet non seulement de rédiger des lignes de code, mais aussi de gérer les fichiers du projet et d'installer diverses extensions pour rendre l'expérience de développement plus simple et personnalisée. De nombreux IDE (Environnement de Développement Intégré) offrent des fonctionnalités similaires à celles de Microsoft, mais le choix de cet IDE repose avant tout sur une préférence personnelle et une question d'habitude.

Pour initialiser un projet avec Next.js, il suffit de suivre la documentation officielle. Celle-ci indique d'exécuter la commande ci-dessous pour démarrer l'initialisation. Cette commande peut être exécutée dans le terminal de l'éditeur de code ou directement sur l'ordinateur. Dans les deux cas, il est judicieux d'ouvrir un dossier avec l'IDE ou de se placer dans un dossier via l'invite de commande de l'ordinateur avant de lancer la commande. Cela permettra de savoir où le projet sera créé.

```
> npx create-next-app@latest
```

Listing 4 *- Commande pour initialiser un nouveau projet avec Next.js*

Cette commande va faire apparaître une série de questions dans le terminal, permettant de paramétrer le projet. En fonction des questions, il suffit soit de taper les réponses, comme pour le nom du projet, soit de se déplacer avec les flèches du clavier, puis d'appuyer sur la touche "Enter" pour valider le choix.

```
> What is your project named? my-app
> Would you like to use TypeScript? No / Yes
> Would you like to use ESLint? No / Yes
> Would you like to use Tailwind CSS? No / Yes
> Would you like to use `src/` directory? No / Yes
> Would you like to use App Router? (recommended) No / Yes
> Would you like to customize the default import alias (@/*)? No / Yes
> What import alias would you like configured? @/*
```

Listing 5 *- Questions posées lors de l'initialisation d'un projet Next.js*

La capture d'écran ci-dessous présente le rendu réel de cette série de questions. Pour ce projet, toutes les questions ont été répondues avec leurs réponses par défaut, à l'exception de celle concernant l'utilisation du répertoire "src/". Dans la configuration actuelle, les dossiers contenant les pages du site se trouvent dans un dossier "app". Par défaut, ce dossier se trouve à la racine du projet, mais ici, le dossier "app" se trouve dans un dossier "src" qui, lui, se trouve à la racine. Cela permet de bien séparer les fichiers liés à l'application de ceux de configuration.

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    POSTMAN CONSOLE

PS C:\Users\nicom\Documents\Projets\TB\dg-stockflow> npx create-next-app@latest
Need to install the following packages:
create-next-app@14.2.3
Ok to proceed? (y) y
√ What is your project named? ... my-app
√ Would you like to use TypeScript? ... No / Yes
√ Would you like to use ESLint? ... No / Yes
√ Would you like to use Tailwind CSS? ... No / Yes
√ Would you like to use `src/` directory? ... No / Yes
√ Would you like to use App Router? (recommended) ... No / Yes
√ Would you like to customize the default import alias (@/*)? ... No / Yes
Creating a new Next.js app in C:\Users\nicom\Documents\Projets\TB\dg-stockflow\my-app.

Using npm.

Initializing project with template: app-tw


Installing dependencies:
- react
- react-dom
- next

Installing devDependencies:
- typescript
- @types/node
- @types/react
- @types/react-dom
- postcss
- tailwindcss


added 137 packages, and audited 138 packages in 14s

31 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
Success! Created my-app at C:\Users\nicom\Documents\Projets\TB\dg-stockflow\my-app
```

Figure 1 *- Processus d'initialisation complet du projet Next.js*

Une fois l'initialisation terminée avec le message de succès, on peut constater que le dossier accueillant le projet s'est enrichi d'un certain nombre de dossiers et de fichiers, comme le montre la capture d'écran ci-dessous. Dans les grandes lignes, on trouve :

- Le dossier "src", qui contient les fichiers liés à l'application, comme expliqué précédemment.
- Un dossier "public", destiné à stocker des fichiers accessibles publiquement sur l'application, tel que des images.
- Un dossier "node_modules", qui contient tous les librairies et modules nécessaires au fonctionnement de Next.js ainsi que ceux qui seront installés ultérieurement, comme Puppeteer.
- Les fichiers de configuration de l'application.

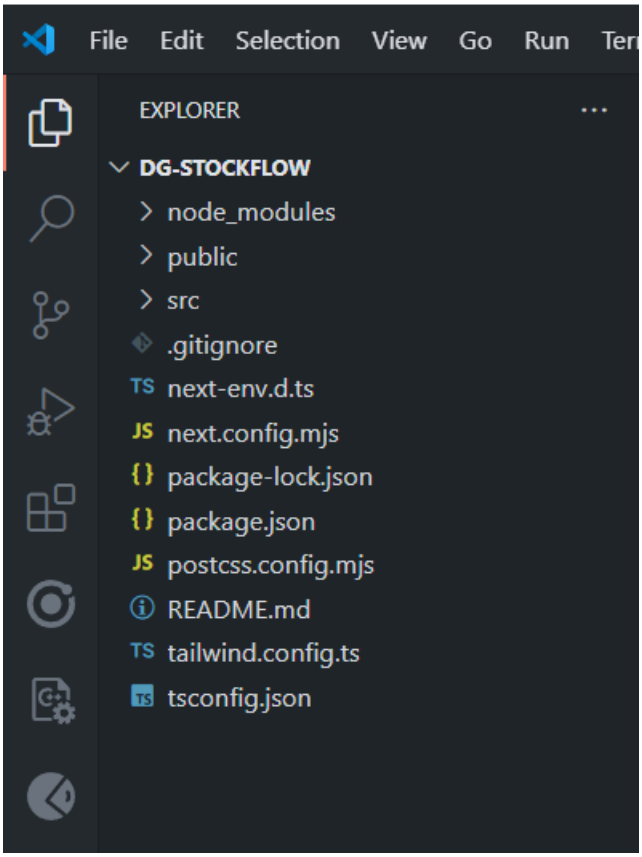


Figure 2 *- Structure des fichiers du projet Next.js*

À noter que le nom du projet et du dossier du projet est susceptible d'évoluer et n'est donc pas définitif. Les captures d'écran provenant d'une phase préliminaire du projet, le nom était donc provisoire.

2.2.2 Préparation et fonctionnement de Next.js

Le projet est à présent créé avec tous les fichiers initiaux nécessaires au bon fonctionnement de Next.js. Il est possible de démarrer un serveur de développement localement sur l'ordinateur en exécutant la commande ci-dessous dans le terminal de l'éditeur de code.

```
> npm run dev
```

Listing 6 *- Commande utilisée pour démarrer un serveur de développement local de l'application Next.js*

Le site sera ensuite accessible depuis un navigateur à l'adresse http://localhost:3000/. Cette adresse fait référence à l'adresse IP locale de l'ordinateur et utilise le port 3000. L'adresse IP locale, souvent représentée par "localhost"

ou "127.0.0.1", désigne l'ordinateur que l'on utilise actuellement. Cela signifie que le site est hébergé sur notre propre machine et n'est accessible que depuis celle-ci. Le port 3000 est un canal spécifique par lequel les données sont envoyées et reçues. En informatique, un port est un point d'entrée ou de sortie pour les informations, permettant de gérer plusieurs connexions sur une seule adresse IP.

En accédant à l'adresse du site, on constate que le site a déjà une mise en page et du contenu. Il s'agit en fait de la page de démarrage de Next.js, comprenant des liens vers la documentation et des modèles de site préconstruits. Bien entendu, il faudra retirer tout le code existant avant de commencer pour démarrer sur une base vierge.

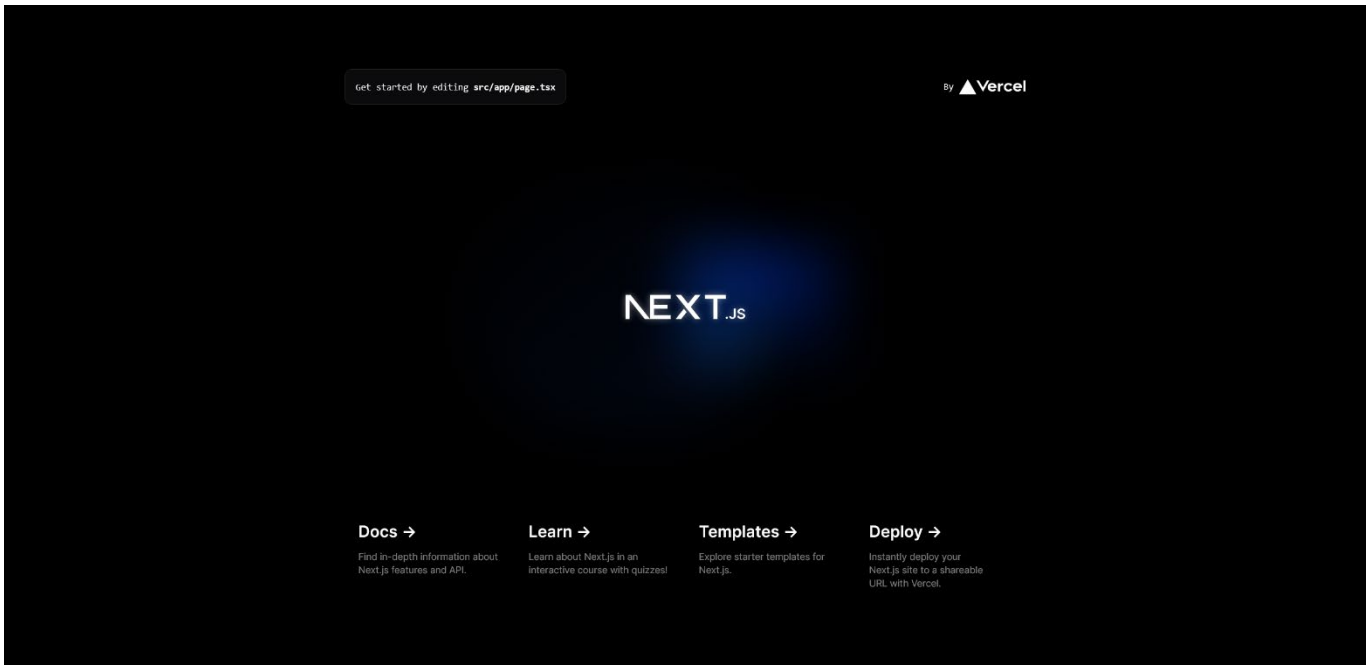


Figure 3 *- Page de démarrage par défaut d'un projet Next.js créé avec l'assistant d'initialisation*

Il faut modifier trois fichiers pour partir sur une base vide. Tous ces fichiers sont localisés dans le dossier "app". Le premier fichier est "page.tsx", qui correspond à la page d'accueil du site. Cela permet d'exposer le fonctionnement de Next.js concernant la navigation et le rendu des pages.

La navigation fonctionne avec les dossiers présents dans le répertoire "app". Par exemple, un dossier "about" contenant un fichier "page.tsx" sera accessible à l'adresse du site suivie de "/about". Le chemin d'accès aux pages du site correspond aux chemins des dossiers dans le répertoire "app". Chaque fichier "page.tsx" renvoie ainsi à une page à l'adresse du dossier. Le dossier "app" représente la racine du site pour la navigation.

Actuellement, il n'y a qu'une seule page, et elle peut être modifiée pour ne retenir que la base nécessaire à son fonctionnement. On remarque ici qu'il ne s'agit pas d'un fichier HTML directement, mais d'une fonction nommée "Home" qui retourne du code HTML, comme dans l'exemple ci-dessous.

```
export default function Home() {
    // Il est possible d'exécuter du code ici avant de retourner quelque chose
    return (
        <main>
            <h1>Je suis la page d'accueil</h1>
        </main>
    );
}
```

Listing 7 *- Exemple de fonction Home retournant un élément HTML qui représente la page d'accueil*

Sans entrer trop dans les détails, Next.js utilise des composants React écrits en TypeScript pour créer des pages web. Dans l'exemple précédent, la fonction "Home" est un composant React utilisé comme page d'accueil. TypeScript, dérivé de JavaScript, permet d'écrire du code fortement typé, améliorant ainsi la robustesse et la maintenance du projet. Le code TypeScript peut être exécuté avant de retourner du contenu, permettant ainsi de préparer des données ou d'effectuer des actions nécessaires avant le rendu. La fonction retourne un élément JSX avec une balise `<main>` contenant un titre `<h1>Je suis la page d'accueil</h1>`. Ce composant sera affiché lorsque l'on accède à la page d'accueil du site.

Le deuxième fichier à modifier se nomme "layout.tsx" et se trouve également à la racine du dossier "app". Ce fichier permet de définir une structure commune à toutes les pages présentes dans "app". En pratique, il est possible de réutiliser des éléments comme un entête ou un pied de page sans avoir à les copier-coller dans chaque page. Comme il s'agit également d'un fichier TSX, il exporte une fonction qui retourne du code HTML. Ce dernier représente la base d'une page web, notamment avec la balise `<html>`. Le "children" entre accolades en tant que contenu de la balise `<body>` indique l'emplacement des éléments "enfants" du "layout". Cela signifie que le code HTML de chaque page se retrouvera ici. Ce fichier définit également le titre du site visible dans l'onglet du navigateur. L'extrait de code ci-dessous illustre un exemple de ce que contient le fichier "layout.tsx".

```
import type { Metadata } from "next";
import "./globals.css";

export const metadata: Metadata = {
    title: "Titre du site",
    description: "Description de ce que fait le site",
};

export default function RootLayout({
    children,
}: Readonly<{
    children: React.ReactNode;
}>) {
    return (
        <html lang="en">
            <body>{children}</body>
        </html>
    );
}
```

Listing 8 *- Code pour le fichier « layout.tsx », définissant la structure de base de chaque page du site*

Le dernier fichier à modifier est "globals.css", situé à la racine du dossier "app" comme les fichiers précédents. Ce fichier permet de définir les styles globaux de l'application en complément ou en remplacement de Tailwind CSS. Il doit être importé dans un fichier pour être utilisé avec le code de ce dernier, notamment dans "layout.tsx". Ainsi, tout le code du "layout" ainsi que ses enfants bénéficient de ce style global. Une fois dépouillé, il ne devrait rester que les quelques lignes utiles au fonctionnement de Tailwind CSS, comme illustré dans l'extrait de code suivant.

```
@tailwind base;
@tailwind components;
@tailwind utilities;

/* Mise en forme et styles globaux */
```

Listing 9 *- Code CSS global incluant les styles de base, les composants et les utilitaires de Tailwind CSS*

L'application est à présent prête à être alimentée de tout le code nécessaire au bon fonctionnement des différentes fonctionnalités de la solution de ce projet.

2.2.3 Interface de test

Avant de commencer pleinement à coder toute la solution, il est nécessaire de faire quelques tests afin d'évaluer la faisabilité des fonctions principales, notamment la sélection des produits.

Pour tester convenablement et visualiser plus facilement les résultats, une interface préliminaire a été développée. Cette interface permet d'afficher les résultats de deux tests liés au web scraping. Le premier doit afficher une liste de produits et le deuxième le nombre de catégories de galaxus.ch ainsi que le temps nécessaire pour récupérer ces informations.

L'interface se compose d'un entête avec les logos de l'entreprise, le nom du projet, le nombre de catégories récupérées à partir du site et le temps nécessaire pour effectuer cette opération, ainsi que des boutons permettant d'enclencher les différentes requêtes. Le tout se compose d'une unique page accessible à la racine de l'adresse du site, http://localhost:3000/.

Cette page préliminaire permet de centraliser les tests et de s'assurer que les fonctionnalités principales, comme le web scraping, fonctionnent correctement avant de les intégrer pleinement dans le projet.

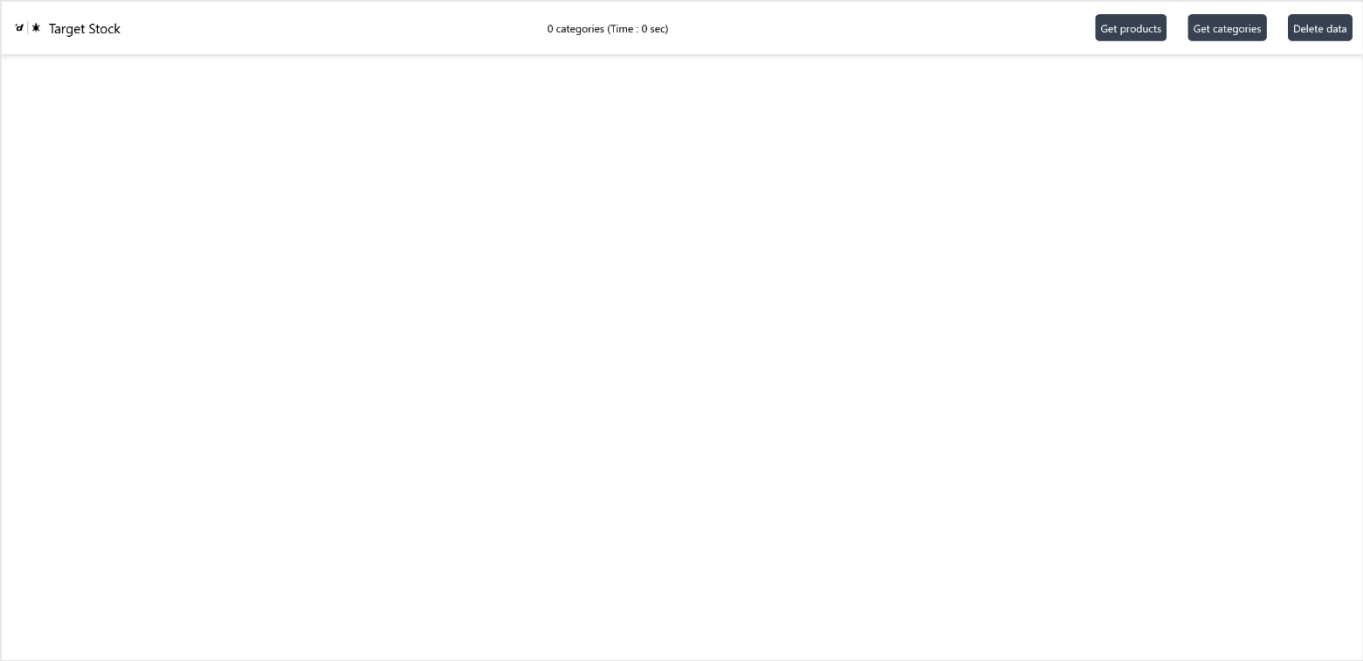


Figure 4 *- Interface de tests permettant de récupérer une liste de produits et une quantité de catégories*

Cette partie du projet est disponible dans un repository GitHub sous la branche "test-phase" à l'adresse suivante : https://github.com/nicomeuwly/dg-targetstock/tree/test-phase. Cela est particulièrement utile pour collaborer sur un projet, mais le repository est surtout utile dans notre cas pour avoir une sauvegarde du code. Il contient également une marche à suivre pour pouvoir tester le code localement sur sa propre machine. De plus, ce code est également annexé au présent rapport via la clé USB fournie.

En utilisant GitHub, nous pouvons facilement suivre les modifications du code, collaborer avec d'autres développeurs, et assurer une gestion efficace des versions. La branche "test-phase" permet de travailler spécifiquement sur les tests sans affecter la branche principale du projet.

2.3 Récupération des données

Afin de vérifier la viabilité du projet, des versions préliminaires des futures fonctionnalités ont été élaborées. Ces fonctionnalités sont liées à la sélection des produits à destination du stock cible.

Le premier test a consisté à récupérer une liste de produits à partir d'une URL. Le deuxième test permet de récupérer la liste des catégories, lesquelles fourniront les URL nécessaires pour extraire les produits du site.

2.3.1 Requêtes API

Dans Next.js, le système d'API utilise le "App Router" pour faciliter la création de points de terminaison API. Cela correspond à ce qui a été dit précédemment sur la navigation dans l'application, qui passe par les dossiers et fichiers se trouvant dans le répertoire "app". Il est possible de créer des fichiers JavaScript ou TypeScript dans le dossier "app/api" pour définir des routes API. Chaque fichier correspond à une route et exporte des fonctions asynchrones pour gérer les requêtes HTTP telles que GET, POST, PUT, PATCH ou DELETE.

Chacun de ces types de requêtes a une fonction spécifique :

- **GET** : Récupère des données d'un serveur. C'est comme demander une copie d'un document.
- **POST** : Envoie des données au serveur pour créer quelque chose de nouveau. C'est comme remplir et soumettre un formulaire.
- **PUT** : Met à jour des données existantes sur le serveur. C'est comme remplacer une ancienne version d'un document par une nouvelle.
- **PATCH** : Met à jour partiellement des données existantes sur le serveur. C'est comme corriger une partie spécifique d'un document sans le remplacer entièrement.
- **DELETE** : Supprime des données sur le serveur. C'est comme déchirer un document dont on n'a plus besoin.

Ainsi, chaque chemin dans le dossier "api" contiendra un fichier exportant une ou plusieurs fonctions correspondant aux types de requêtes HTTP. Pour les besoins des tests, il y aura deux fichiers, chacun nommé "route.ts", l'un s'occupant des catégories et l'autre des produits.

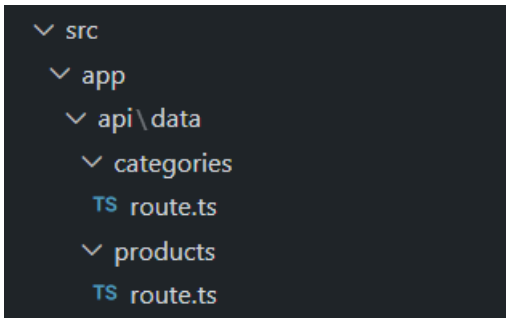


Figure 5 *- Structure des fichiers pour accéder aux points de terminaison de l'API*

Pour accéder à ces fonctions, il faudra utiliser l'adresse de base du site suivie de "/api/data/categories" pour le premier et "/api/data/products" pour le second, en indiquant le type de requête HTTP que l'on souhaite utiliser.

2.3.2 Liste de produits

Pour récupérer une liste de produits, une requête GET est utilisée, car il faut simplement accéder à une ressource sans effectuer de modifications. La fonction est structurée en deux temps à l'aide de l'instruction `try…catch`, qui tente d'exécuter une opération et, en cas d'échec, exécute un code spécifique pour traiter l'erreur.

La fonction permet d'ouvrir un navigateur "virtuel", qui ne sera pas visible à l'utilisateur, et d'accéder à une page à partir d'une URL fournie au préalable. La fonction `openProductsPage()` s'occupe d'ouvrir cette page à l'aide du navigateur et de transmettre cette page à une autre fonction qui va en extraire les informations souhaitées. Si tout se passe bien, les résultats sont sauvegardés dans un fichier JSON, et la requête renvoie une réponse avec un statut de réussite et les données recueillies. Ensuite, le navigateur virtuel peut être fermé. En cas d'erreur, la requête retourne une réponse indiquant qu'une erreur s'est produite. Le fichier JSON sert de stockage temporaire pour les premiers tests.

```
export async function GET() {
    try {
        const browser = await puppeteer.launch({
            args: ["--no-sandbox", "--disable-setuid-sandbox"],
        });
        const list = await openProductsPage(
            browser,
            `https://www.digitec.ch/fr/s1/producttype/telephone-portable-24?fil
            ter=t_off%3DInStock&so=15`
        );
        await browser.close();
        const data = JSON.stringify(list);
        fs.writeFileSync("./data/products.json", data);
        return new Response(JSON.stringify(list), {
            status: 200,
            headers: { "Content-Type": "application/json" },
        });
    } catch (error: any) {
        return new Response(JSON.stringify({ error: error.message }), {
            status: 500,
            headers: { "Content-Type": "application/json" },
        });
    }
}
```

Listing 10 *- Code de la route API permettant de récupérer une liste de produits*

La fonction permettant d'extraire les informations des produits se nomme `scrapeProducts()` et va fouiller dans la page fournie pour trouver un élément spécifique : le conteneur qui englobe tous les produits présents sur la page et ses enfants.

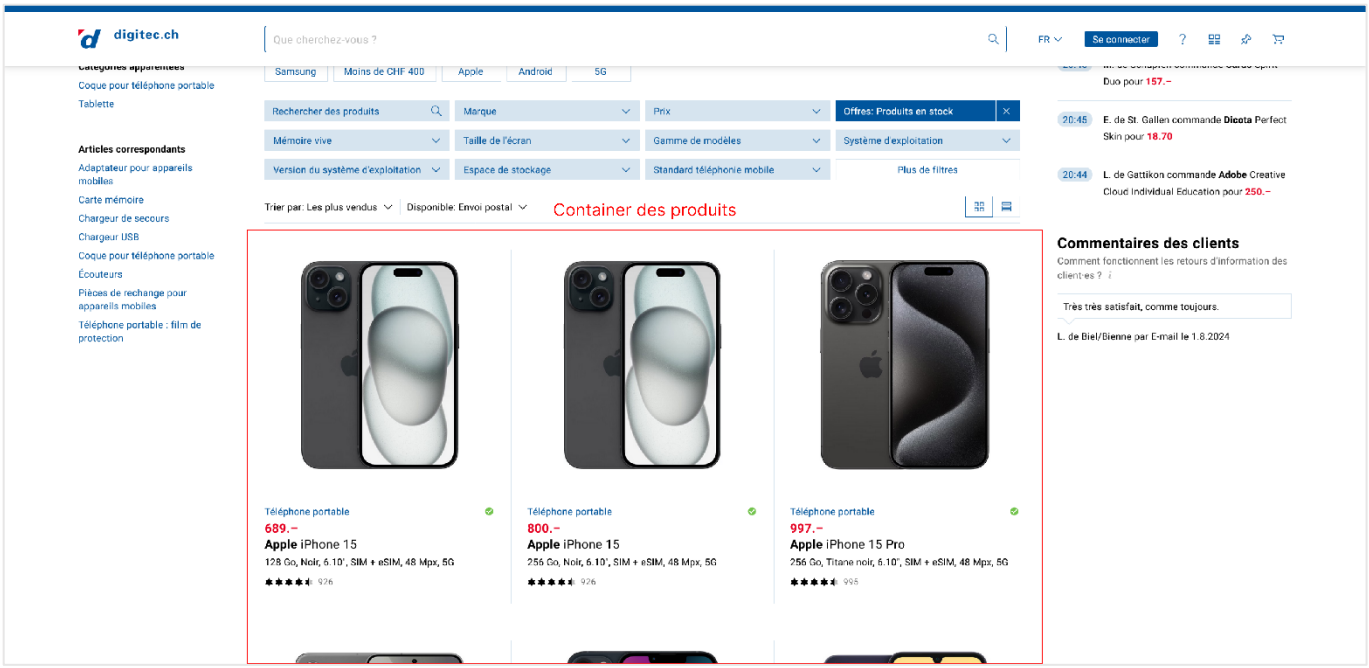


Figure 6 *- Liste des téléphones portables sur digitec.ch illustrant le conteneur des produits*

Les enfants en question sont les produits, qui, dans le code HTML de la page, sont définis dans une balise `<article>`. On utilise alors un sélecteur qui indique l'élément HTML parent avec certaines classes et les éléments enfants.

Une fois que tout cela est défini, la librairie Cheerio est utilisée pour charger le contenu de la page et tenter d'accéder aux éléments déterminés par le sélecteur. S'il y a un problème pour accéder au contenu désiré, la fonction réessaye après un délai de 2 secondes, jusqu'à un maximum de 3 tentatives avant d'abandonner. Si le sélecteur est trouvé, cela crée un tableau regroupant tous les éléments `<article>` représentant les produits. Il ne reste plus qu'à parcourir toutes les entrées du tableau et, pour chacune d'entre elles, extraire différentes propriétés afin de créer un objet contenant ces propriétés. Chaque objet est ensuite inséré dans un tableau qui est retourné par la fonction `scrapeProducts()`.

```
const scrapeProducts = async (page : any) => {
    const selector = "div.sc-5d2f6f43-1.jwLEDS article";
    let list: {
        product: string,
        details: string,
        price: string,
        url: string | undefined,
        imageURL: string | undefined,
    }[] = [];
    const $ = cheerio.load(await page.content());
    if (!(await page.$(selector))) {
        productRequestCounter++;
        console.log("No article found, retrying... " + productRequestCounter);
        if (productRequestCounter < 3) {
            await delay(2000);
            await scrapeProducts(page);
        } else {
            console.log("No article found, aborting...");
            productRequestCounter = 0;
        }
        return;
    }
    const articleTags = $(selector);
    articleTags.each((i, el) => {
        const imageURL = $(el).find("img").attr("src");
        const product = $(el).find("p.sc-2e9036-0.cNsIaf").text();
        const details = $(el).find("p.sc-b3dc936d-9.BunLw").text();
        const priceString = $(el).find("span.sc-3ffcdfc9-1.cHHHJV").text();
        const url = $(el).find("a.sc-d9c28d7f-0.btOvGx").attr("href");
        const price = priceString.replace("CHF", "");
        list.push({ product, details, price, url, imageURL });
    });
    return list;
};
```

Listing 11 *- Code de la fonction permettant de récupérer une liste de produit à partir d'une page web*

Pour accéder aux résultats de ces fonctions, il faudra envoyer une requête à l'adresse du fichier dans le dossier "api" depuis la page où nous souhaitons avoir les données. La fonction `getProducts()` sur la page d'accueil essaie d'envoyer une requête à l'adresse "/api/data/products" et attend le résultat. Si le processus est réussi, les résultats seront stockés dans un tableau et, en cas d'erreur, celle-ci sera affichée dans la console du navigateur.

```
const getProducts = async () => {
    setData([]);
    setLoading(true);
    try {
        const res = await fetch("/api/data/products");
        const data = await res.json();
        setData(data);
    } catch (error) {
        console.error("Failed to fetch data:", error);
    } finally {
        setLoading(false);
    }
};
```

Listing 12 *- Code de la fonction permettant de récupérer les données de produits via la route API dédiée*

La fonction précédente sera rattachée à un bouton et, au clic de ce dernier, les données seront récupérées. Une fois cela fait, il suffira de les afficher sur la page pour visualiser le résultat. Pour plus de lisibilité, chaque produit bénéficiera d'une mise en page spécifique.

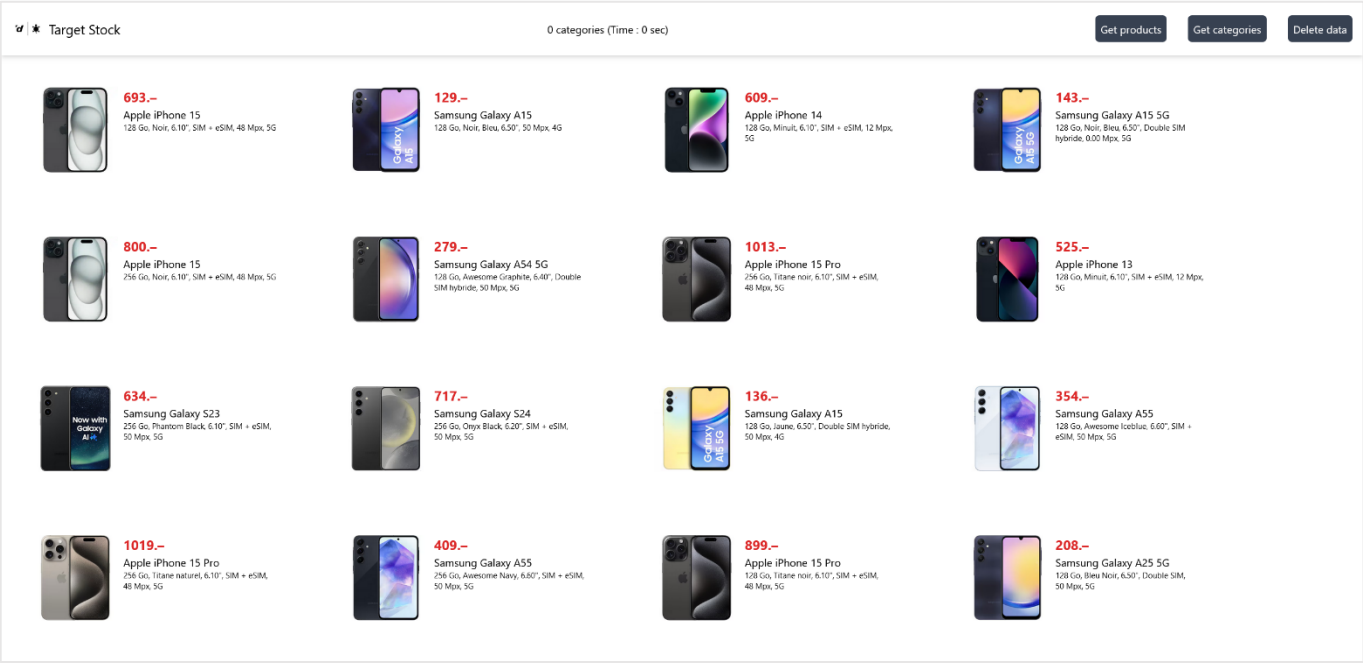


Figure 7 *- Interface de tests avec les résultats de la requête de récupération des produits*

2.3.3 Liste et quantité de catégories

Pour obtenir une liste des différentes catégories présentes sur galaxus.ch, nous allons également utiliser une requête GET, car il s'agit d'un simple accès à une ressource. Le principe sera globalement le même que pour récupérer les produits. Un navigateur virtuel va se lancer, une URL lui sera fournie pour qu'il puisse ouvrir une page et l'extraction de données pourra commencer à partir de cette dernière.

La subtilité ici est qu'il y a plusieurs niveaux de catégories. Lorsqu'on ouvre une catégorie, une liste de sous-catégories apparaît, et ainsi de suite. Il faut donc répéter l'opération plusieurs fois pour obtenir toutes les catégories sur quatre niveaux dans notre cas. La première fois, la fonction `openCategoriesPage()` sera utilisée pour ouvrir la page d'accueil de galaxus.ch et extraire la liste des catégories principales. Cette fonction va ouvrir une page à l'aide du navigateur virtuel sur l'adresse fournie et, avec l'aide d'une autre fonction, elle va retourner la première liste de

catégories. À partir de cette dernière, il faudra exécuter la fonction `openCategoryLinks()`, qui va ouvrir chaque lien présent dans la première liste et en extraire les sous-catégories. L'opération est répétée deux fois de plus, chaque fois avec la dernière liste récupérée.

Une fois la dernière liste en notre possession, il faudra la traiter afin d'enlever les doublons avant de l'envoyer comme réponse à la requête GET. En effet, il peut arriver que toutes les catégories n'aient pas le même nombre de sous-catégories et que, au lieu d'accéder aux liens inférieurs, on reste sur les précédents. Il faut donc éliminer tous les liens en double récupérés tout au long du processus. Si tout se passe bien, le résultat de la requête pourra être stocké dans un fichier JSON et la requête retournera une réponse indiquant son succès et incluant les données récoltées. En cas de problème, la requête retournera une réponse indiquant une erreur. Le fichier JSON sert également de stockage temporaire pour les premiers tests ici.

```
export async function GET() {
    try {
        console.log("Scraping categories...");
        const browser = await puppeteer.launch({
            args: ["--no-sandbox", "--disable-setuid-sandbox"],
        });
        const firstList = await openCategoriesPage(
            browser,
            "https://www.galaxus.ch/"
        );
        if (!firstList) {
            throw new Error("Failed to retrieve first list");
        }
        const secondList = await openCategoryLinks(browser, firstList);
        if (!secondList) {
            throw new Error("Failed to retrieve second list");
        }
        const thirdList = await openCategoryLinks(browser, secondList);
        if (!thirdList) {
            throw new Error("Failed to retrieve third list");
        }
        const fourthList = await openCategoryLinks(browser, thirdList);
        if (!fourthList) {
            throw new Error("Failed to retrieve fourth list");
        }
        const uniqueList = fourthList.filter((obj, index) => {
            return index === fourthList.findIndex((o) => obj.url === o.url);
        });
        await browser.close();
        const data = JSON.stringify(uniqueList);
        fs.writeFileSync("./data/categories.json", data);
        console.log("Scraping completed");
        return new Response(JSON.stringify(uniqueList), {
            status: 200,
            headers: { "Content-Type": "application/json" },
        });
    } catch (error: any) {
        return new Response(JSON.stringify({ error: error.message }), {
            status: 500,
            headers: { "Content-Type": "application/json" },
        });
    }
}
```

Listing 13 *- Code de la route API permettant de récupérer la liste des catégories présentes sur galaxus.ch*

Comme pour les produits, il y a également une fonction permettant d'extraire les catégories, appelée `scrapeCategories()`. Cette fonction va fouiller dans la page fournie pour trouver un élément spécifique : le conteneur qui englobe les différentes catégories.

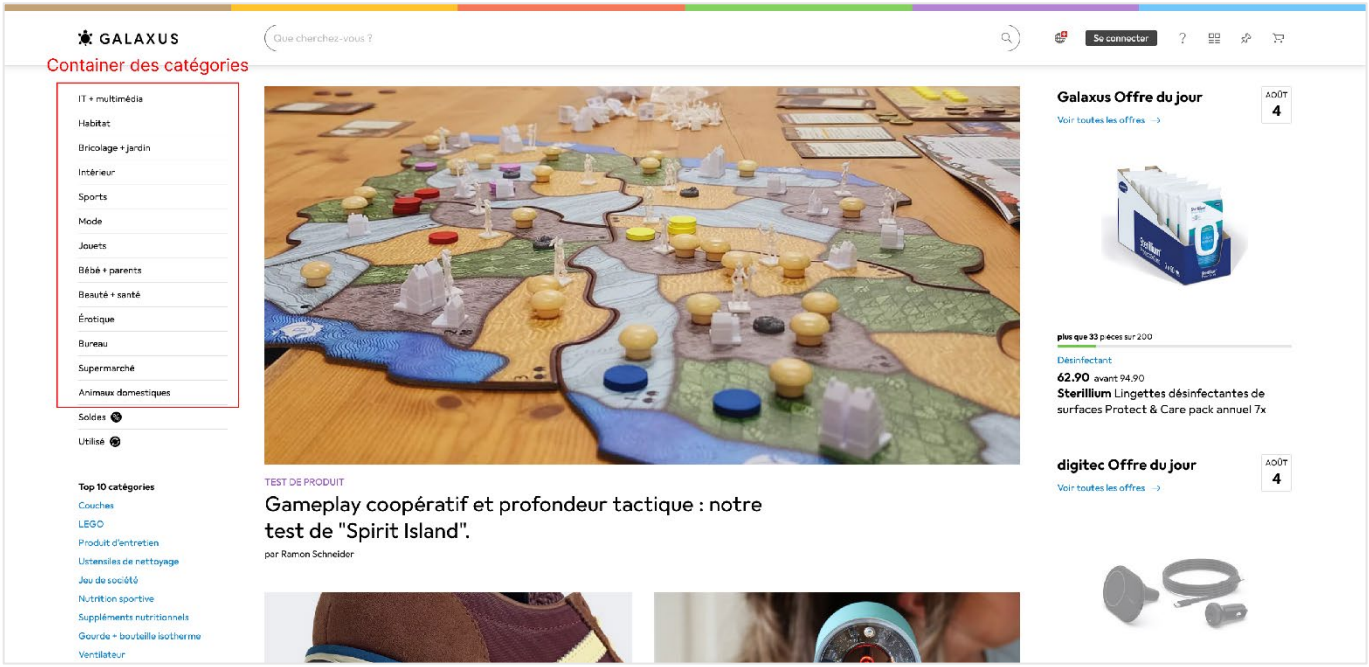


Figure 8 *- Emplacement des catégories principales sur galaxus.ch*

Les catégories principales sont contenues dans un élément `<ul>`, utilisé en HTML pour déclarer des listes, notamment à puces ou à chiffres. Chaque catégorie est représentée par un élément `<li>` et contient le nom de cette dernière ainsi que le lien permettant d'y accéder.

La fonction `scrapeCategories()` commence par définir un tableau d'éléments vide en vue de récupérer la liste des catégories. Ensuite, à l'aide de la librairie Cheerio, elle charge le contenu de la page envoyé comme paramètre de la fonction. La fonction essaie ensuite de trouver l'élément correspondant au sélecteur indiqué en paramètre, qui correspond aux éléments de liste des catégories. Si aucun résultat n'est trouvé, la fonction se relance jusqu'à 3 essais avant de retourner un message d'erreur et de s'arrêter en cas d'échec. Si le sélecteur est trouvé, la fonction récupère tous ces éléments de liste et, pour chacun d'entre eux, extrait le texte et le lien pour les insérer dans la liste vide initialisée en début de fonction. Enfin, elle retourne la liste avec les catégories trouvées.

```
const scrapeCategories = async (page : any, selector: string) => {
    let list: { category: string, url: string | undefined }[] = [];
    const $ = cheerio.load(await page.content());
    if (!(await page.$(selector))) {
        categoryRequestCounter++;
        console.log("No category found, retrying... " + categoryRequestCounter);
        if (categoryRequestCounter < 3) {
            await delay(2000);
            await scrapeCategories(page, selector);
        } else {
            console.log("No category found, aborting...");
            categoryRequestCounter = 0;
        }
        return;
    }
    const categoriesTag = $(selector);
    categoriesTag.each((i, el) => {
        const category = $(el).text();
        const url = $(el).attr("href");
        list.push({ category, url });
    });
    return list;
};
```

Listing 14 *- Code de la fonction permettant de récupérer une liste de catégories à partir d'une page web*

Pour récupérer toutes les catégories, il faut ouvrir chaque lien pour de nouveau sélectionner une liste. Le premier niveau de catégories, celui sur la page d'accueil, va être récupéré à l'aide de la fonction `openCategoriesPage()`, qui va ouvrir une page à l'aide du navigateur virtuel, accéder à un lien et lancer la fonction `scrapeCategories()` avec cette même page et un sélecteur CSS.

Un compteur de requêtes permet de savoir s'il s'agit du premier niveau de catégories ou des suivants. En effet, le premier niveau n'a pas le même sélecteur que les sous-catégories. En procédant de cette manière, nous avons une seule fonction qui récupère à la fois le premier niveau et les suivants.

```
const openCategoriesPage = async (browser: any, url: string) => {
    const page = await browser.newPage();
    await page.goto(url, { waitUntil: "domcontentloaded" });

    let selector = "div.sc-e8cbc69d-0.dKMQVj a";
    if (requestCounter > 0) selector = "ul.sc-1656bbdd-0.gQqszz a";
    const list = await scrapeCategories(page, selector);
    if (requestCounter < 1) list?.splice(-2, 2);
    requestCounter++;
    await page.close();
    return list;
};
```

Listing 15 *- Code de la fonction transmettant une page web à scrapeCategories() via une URL*

Une fois que nous avons la première liste de catégories principales, nous allons pouvoir passer aux sous-niveaux. La fonction `openCategoryLinks()` est utilisée du premier au dernier sous-niveau de catégories. Elle prend en paramètre la dernière liste de catégories générée, celle provenant de la fonction `openCategoriesPage()` la première

fois, et la parcourt pour ouvrir chaque lien avec la fonction précédente, ce qui résulte à chaque fois en une liste temporaire.

Cette liste temporaire est ensuite parcourue pour enregistrer dans une liste commune tous ses éléments. Ainsi, si la liste fournie en paramètre contient 10 liens et que chacun de ces liens permet d'en récupérer 10 de plus, nous aurons finalement une liste de 100 éléments, retournée par `openCategoryLinks()`, qui pourra à son tour être utilisée avec cette même fonction. Cette opération est répétée jusqu'à atteindre le niveau de catégorie souhaitée, soit le 4e dans notre cas.

```
const openCategoryLinks = async (browser: any, list: any) => {
    const results: { category: string, url : string | undefined }[] = [];
    for (const item of list) {
        const tempList = await openCategoriesPage(
            browser,
            "https://www.galaxus.ch" + item.url
        );
        tempList?.forEach((element) => {
            results.push(element);
        });
    }
    return results;
};
```

Listing 16 *- Code de la fonction récupérant et regroupant les sous-catégories dans un tableau*

Pour accéder aux résultats de ces fonctions, il faudra envoyer une requête à l'adresse du fichier dans le dossier "api" depuis la page où nous souhaitons obtenir les données. La fonction `getCategories()` sur la page d'accueil envoie une requête à l'adresse "/api/data/categories" et attend le résultat. Si le processus est réussi, les résultats seront stockés dans un tableau. En cas d'erreur, celle-ci sera affichée dans la console du navigateur.

Afin de déterminer le temps nécessaire à la réalisation de cette tâche, un chronomètre est déclenché au lancement de la requête et s'arrête une fois cette dernière réussie.

```
const getCategories = async () => {
    setCategories([]);
    setTime(0);
    setRunning(true);
    try {
        const res = await fetch("/api/data/categories");
        const data = await res.json();
        setCategories(data);
        setRunning(false);
    } catch (error) {
        console.error("Failed to fetch data:", error);
    }
};
```

Listing 17 *- Code de la fonction permettant de récupérer les données de catégories via la route API dédiée*

Les résultats s'affichent dans l'entête de la page d'accueil de notre interface de test. On peut ainsi voir le nombre de catégories récupérées et le temps qu'il a fallu pour le faire. Par exemple, 3157 catégories ont été enregistrées en 18 minutes et 17 secondes (1097 secondes). À noter que ce temps peut varier selon plusieurs facteurs, tels que l'ordinateur sur lequel tourne le site web et la connexion internet. Les résultats ont varié entre 2530 et 1097 secondes après des tests effectués sur deux ordinateurs différents. Il y a également eu quelques optimisations entre

le premier et le dernier essai, ce qui a permis de réduire drastiquement le temps nécessaire à la réussite de la requête.

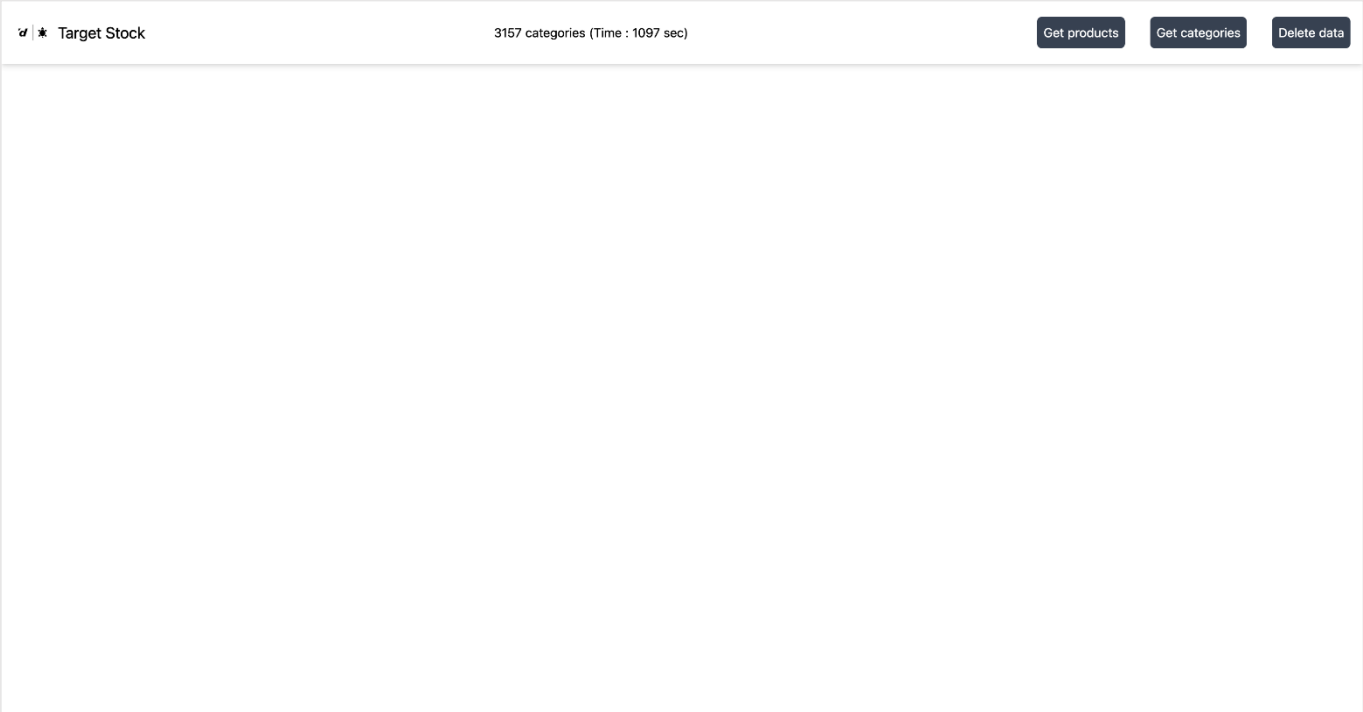


Figure 9 *- Interface de tests avec les résultats de la requête de récupération des catégories*

2.3.4 Problèmes rencontrés

La récupération des produits n'est pas un problème à ce stade du développement, mais elle risque de le devenir. En effet, accéder à une page pour simplement extraire son contenu était aisé, mais toutes les contraintes n'ont pas été respectées lors de cette première phase. Il est possible de filtrer les résultats avec l'URL pour obtenir les produits en stock à l'entrepôt et d'ajouter un tri pour afficher les meilleures ventes en premier, par exemple. Cependant, il reste un paramètre que l'URL ne prend pas en charge : vérifier si un produit peut être retiré en succursale.

À l'heure actuelle, il n'y a ni filtres ni tris pour afficher uniquement ces produits. Il faut survoler chaque pastille relative à la disponibilité du produit pour afficher une liste des options disponibles pour obtenir le produit. C'est à ce moment que l'on peut voir s'il est possible d'obtenir le produit en succursale. C'est un point qu'il faudra tenter de résoudre lors de la prochaine phase.

2.4 Étude du design system

2.4.1 Lignes directrices

Avant de commencer l'élaboration de l'interface de la solution, il a été nécessaire de s'imprégner du design system de l'entreprise. Heureusement, ce dernier est accessible publiquement via galaxus.ch/designsystem, ce qui permet d'en apprendre davantage sur les différents éléments qui composent les plateformes de vente en ligne.

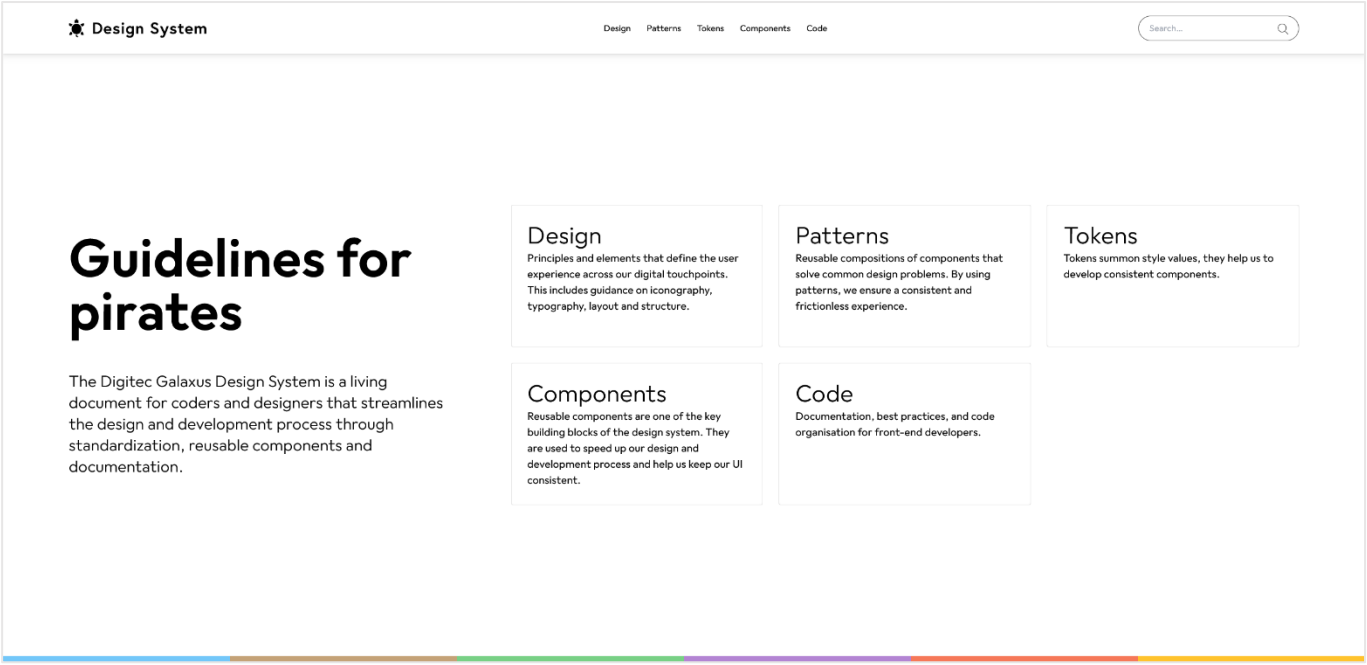


Figure 10 *- Page d'accueil du site dédié au design system de Digitec Galaxus AG*

Ce site a notamment permis d'en savoir plus sur la mise en page, incluant les espacements, les styles de texte, les couleurs, et bien plus encore. En complément des différentes directives, il propose également une liste de composants tels que des boutons, des cases à cocher, des listes déroulantes, etc. Une consultation approfondie de ce site a permis de s'immerger dans le design utilisé par l'entreprise afin de pouvoir l'appliquer à la solution du stock cible.

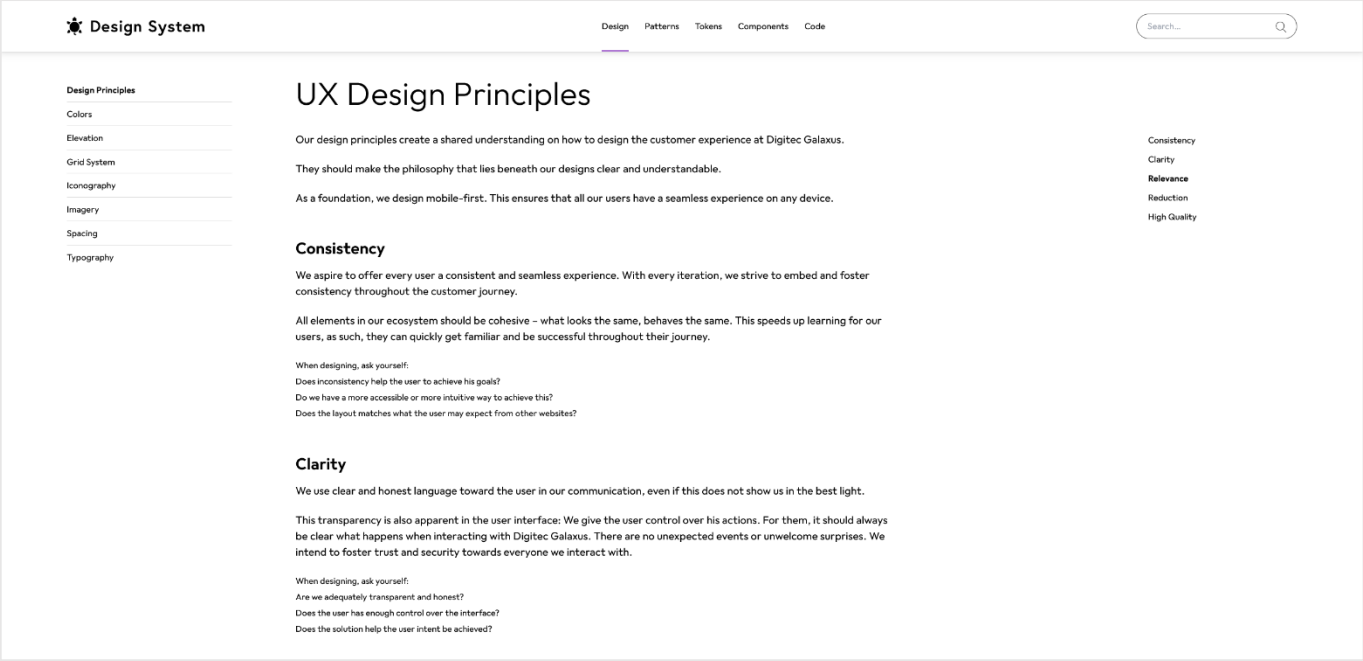


Figure 11 *- Principes de conception UX de digitec.ch et galaxus.ch*

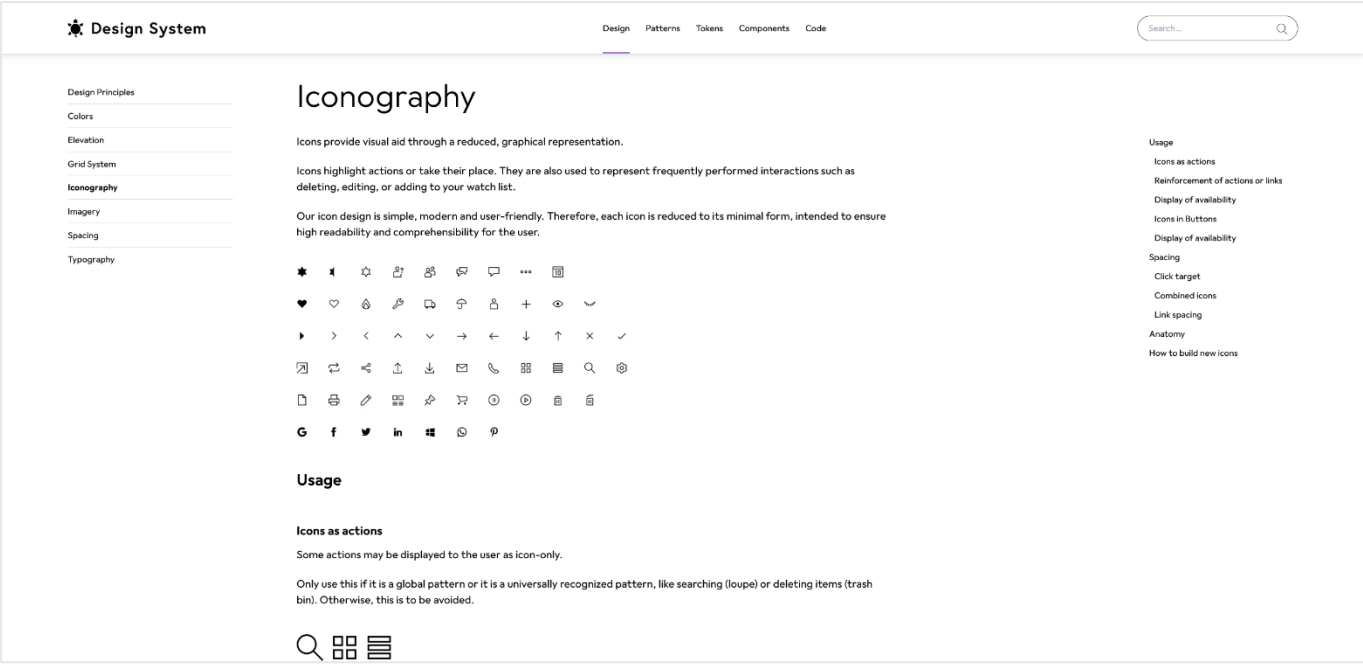


Figure 13 *- Présentations des icônes utilisées sur digitec.ch et galaxus.ch*

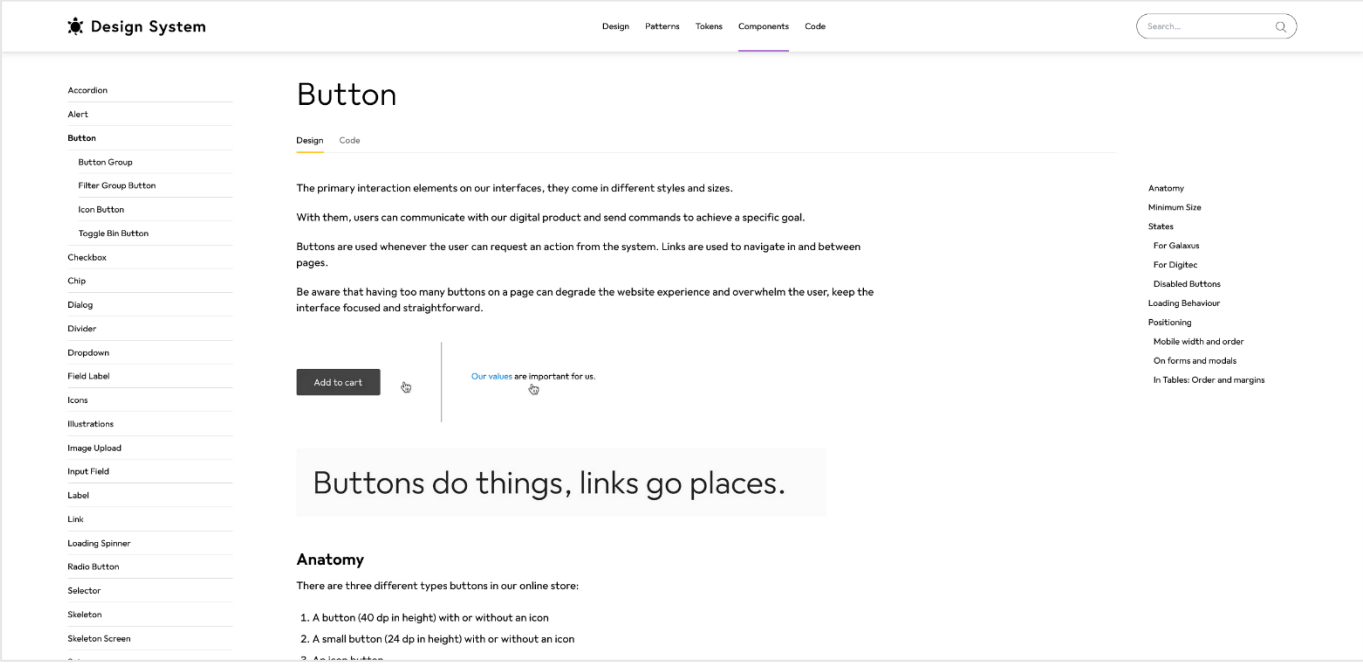


Figure 12 *- Description sur l'utilisation des boutons sur digitec.ch et galaxus.ch*

Bien qu'il y ait beaucoup de matière présente sur ce site, il est important de noter que ce design system est prévu pour un site e-commerce. De ce fait, des ajustements seront nécessaires pour créer un système de gestion, tel que la création de nouvelles icônes, des adaptations de la mise en page et la recherche d'exemples pour des éléments de design inexistants.

2.4.2 Prise en main du design sur Figma

En complément du design system accessible en ligne, un accès aux projets de l'entreprise sur Figma a pu être obtenu. Pour faire simple, Figma permet de créer des interfaces et de les animer à l'aide d'interactions. Bien qu'il s'agisse essentiellement de dessin vectoriel, Figma met un accent particulier sur le design d'interface. L'entreprise

a intégré son design system sur Figma, ce qui permet de gagner du temps lors de l'élaboration de l'interface de la solution en copiant directement des composants depuis les projets des équipes en charge de l'UI et de l'UX.

En utilisant le compte Microsoft de l'entreprise sur Figma pour créer l'interface, il est possible d'accéder directement aux composants depuis le menu "Assets" lors de la conception. Il suffit de rechercher l'élément souhaité et, s'il existe, il sera proposé, avec une indication du fichier dans lequel il se trouve. Par exemple, un bouton peut être disponible dans plusieurs fichiers, à la fois en tant qu'icône et en tant que simple bouton, comme illustré ci-dessous.

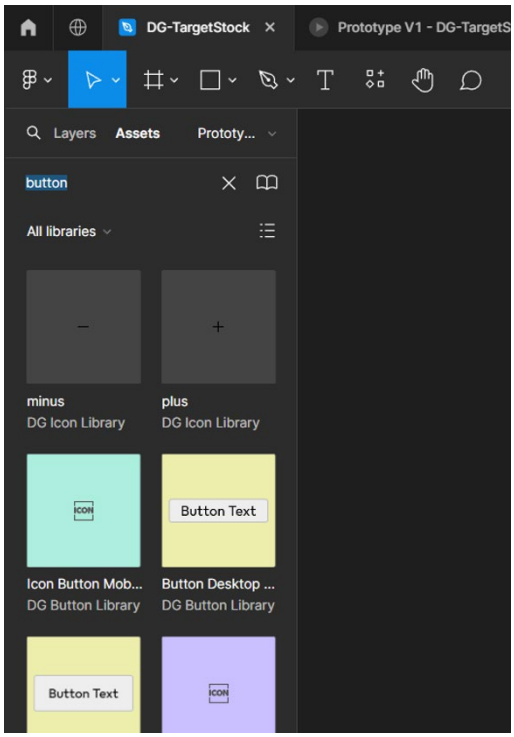


Figure 14 *- Recherche de composants depuis Figma*

À partir de ces différents fichiers, il a été possible d'explorer les divers types de composants et leurs variantes, tels que les changements d'état lors du survol d'un bouton. Des composants ont ainsi pu être récupérés et mis de côté pour l'élaboration de la maquette de la solution du stock cible. Certains composants, comme les boutons, peuvent être repris tels quels, mais d'autres, comme l'entête ou le pied de page, nécessiteront des adaptations pour correspondre au type de système à concevoir pour ce projet.

2.5 Élaboration de la maquette

2.5.1 Préambule

Afin de gagner du temps et étant donné que le design ne part pas de zéro mais bien d'une base existante en termes de mise en page, de composants, etc., une maquette haute-fidélité a été directement conçue. Cela permet également de visualiser immédiatement les ajustements nécessaires et d'obtenir des retours plus précis de la part des parties prenantes, facilitant ainsi la validation et l'itération du design.

La maquette est en anglais afin de simplifier la compréhension pour un large public, notamment pour les personnes ne parlant pas forcément français. De plus, l'anglais est largement utilisé et compris au sein de l'entreprise, ce qui facilite la communication et les tests avec des collaborateurs de différentes régions linguistiques. En fonction de l'avancement du développement, la version définitive de la solution devrait permettre à l'utilisateur de choisir la langue.

2.5.2 Première ébauche

En parcourant les différents éléments du design system de l'entreprise, un composant en particulier a attiré mon attention : la "Timeline". Cet élément de chronologie permet, dans le cadre du site, de décrire les étapes d'un processus, comme une expédition ou un cas de garantie, par exemple. Il se trouve que la fonctionnalité principale de la solution, à savoir la génération d'une liste de produits et son enregistrement dans l'ERP, est un processus nécessitant plusieurs étapes distinctes. Une première ébauche de la solution a donc été conçue autour de ce concept de "Timeline".

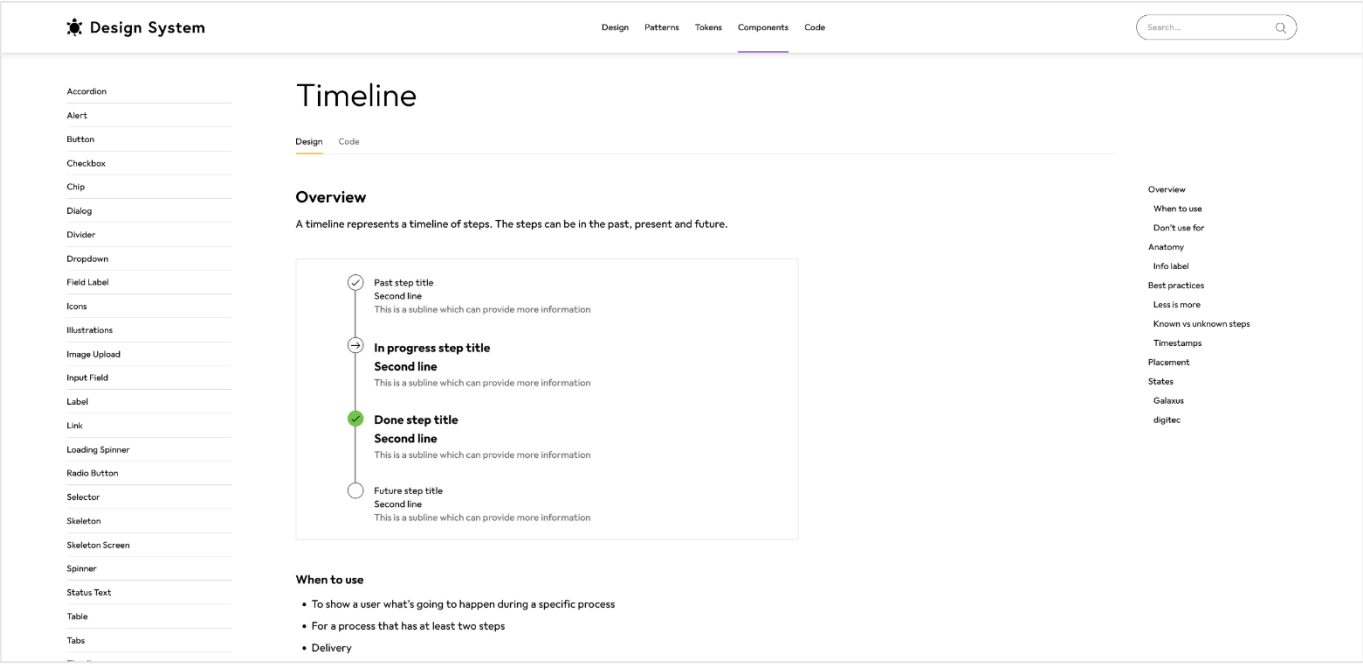


Figure 15 *- Composant de chronologie utilisé pour indiquer les étapes d'un processus du design system*

Les images qui suivent montrent ce à quoi ressemblait initialement le processus de génération du stock cible. Certains éléments de l'interface ont été conservés par la suite, comme l'entête et le pied de page. Ces derniers sont inspirés de ce que l'on retrouve sur galaxus.ch mais également sur le site du design system, avec notamment, pour l'entête, un logo à gauche, une navigation au centre et des icônes menant à d'autres fonctions ou pages du site à droite.

Parmi ces dernières, il y a :

- La fonction pour modifier la langue et la succursale, représentée par une icône de globe.
- Les réglages du mode automatique avec l'icône d'engrenage.
- Le bouton relatif au compte utilisateur, comme le bouton de déconnexion.

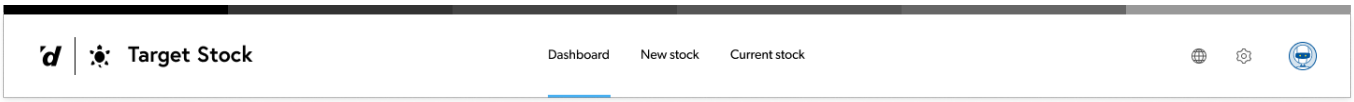


Figure 16 *- Entête de la solution du stock cible (version initiale)*

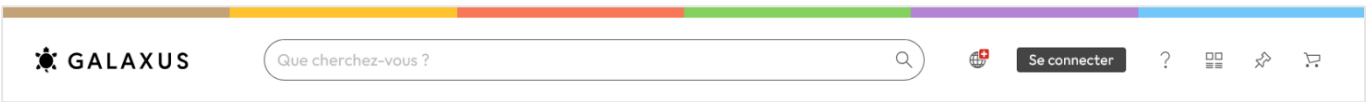


Figure 17 *- Entête de galaxus.ch en thème clair*

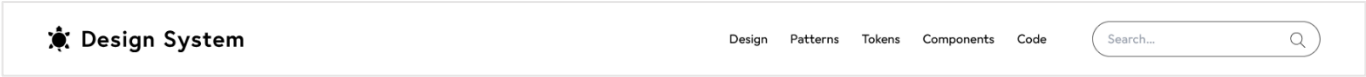


Figure 18 *- Entête du site dédié au design system de Digitec Galaxus AG*

Le pied de page est une version minimaliste de celui que l'on retrouve sur galaxus.ch, incluant notamment l'option pour modifier le thème de l'application (clair, sombre et automatique) et une indication sur la succursale sélectionnée via le réglage correspondant. Le nom de la succursale fait également office de raccourci pour accéder au stock actuel.

Figure 19 *- Pied de page de la solution du stock cible (version initiale)*

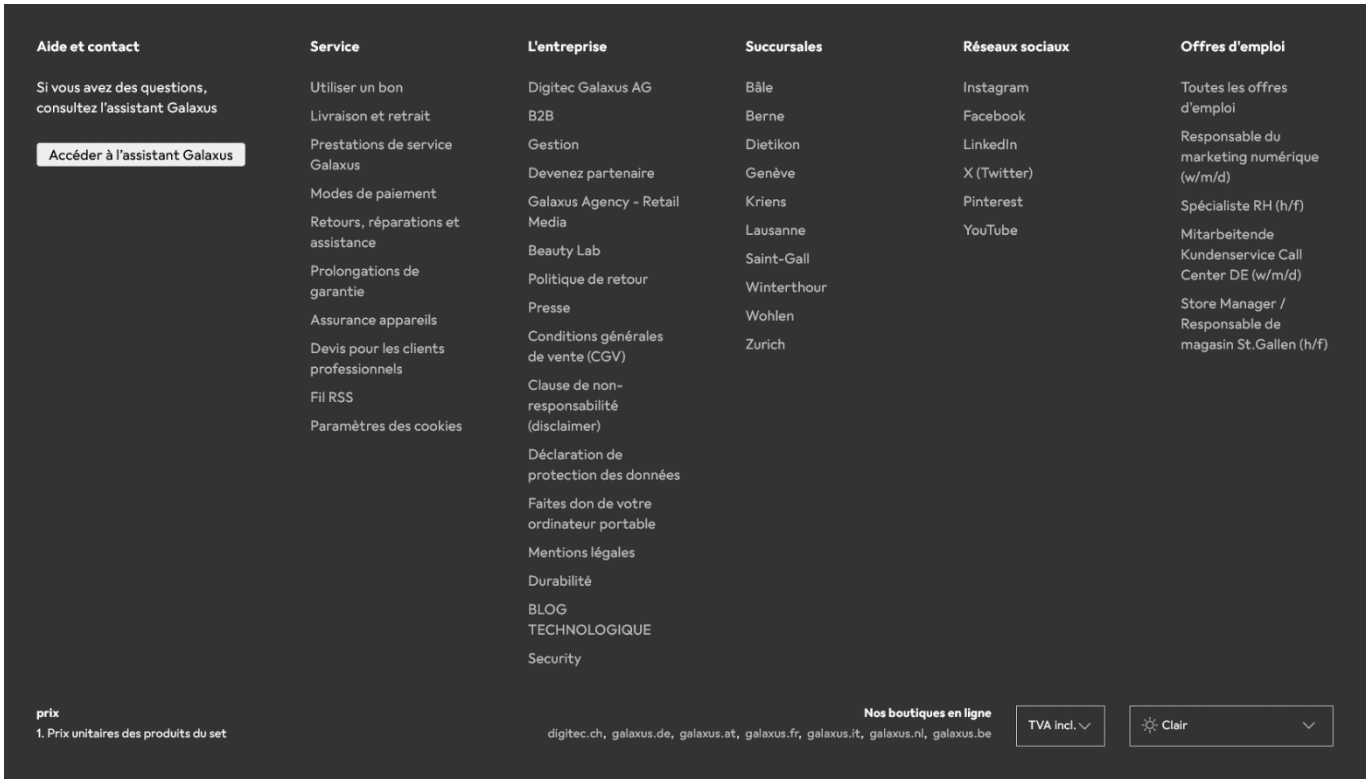


Figure 20 *- Pied de page de galaxus.ch en thème clair*

On commence ici par déterminer si l'on souhaite générer une liste automatiquement ou manuellement. La chronologie de l'étape actuelle et celles à venir est également affichée. La mise en page est assez simple avec des pages contenant un titre, du texte, le processus, des boutons, le tout centré horizontalement sur la page. Cela reprend en partie ce que l'on trouve sur galaxus.ch, bien que dans ce cas, il n'y ait pas de contenu sur les côtés gauche et droit.

Chaque étape du processus comprend un titre et une description. Cette dernière est davantage détaillée lorsque l'on est à l'étape correspondante, et un cercle sur la gauche sert d'indicateur visuel de l'état de la tâche.

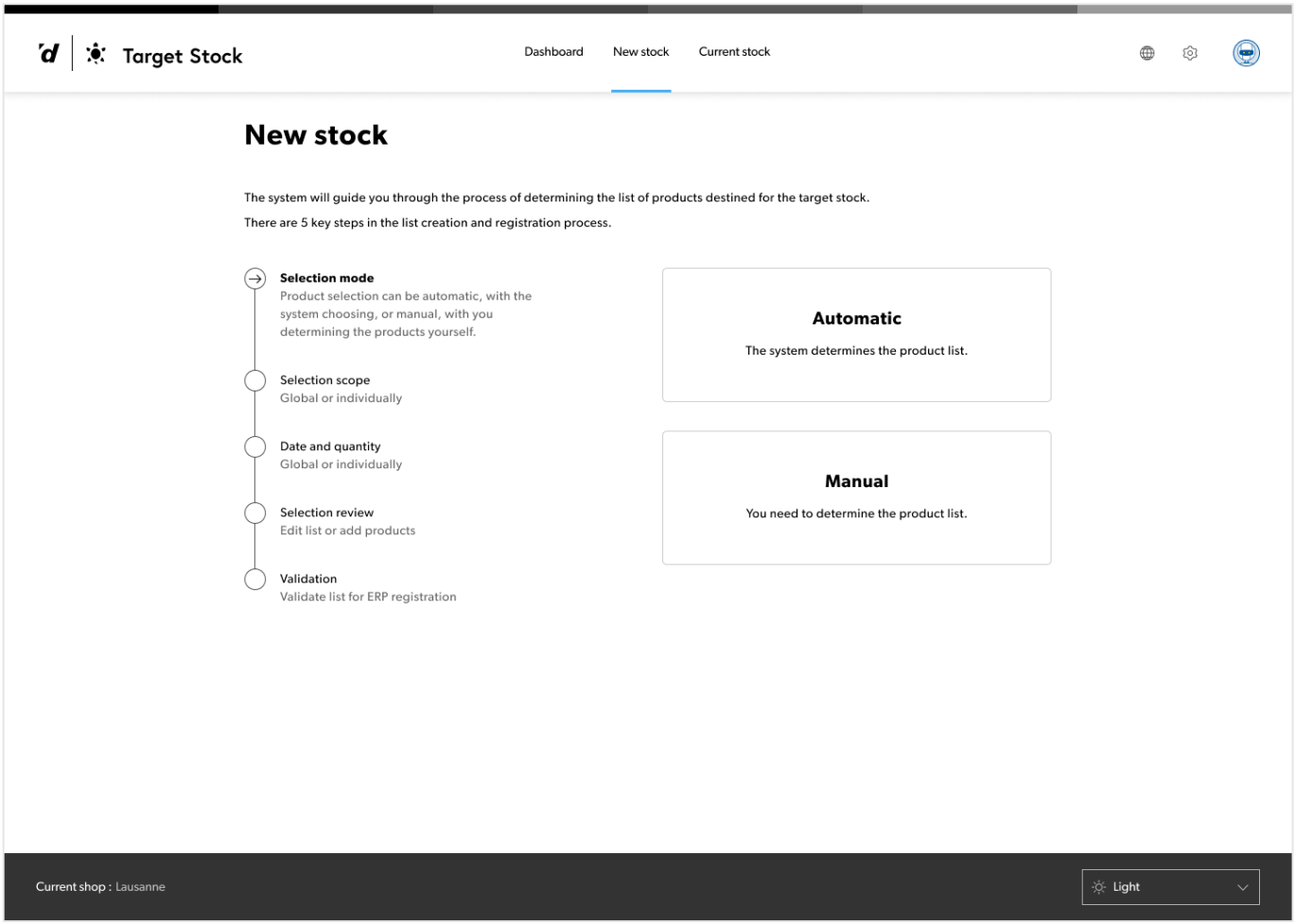


Figure 21 *- Version initiale de la maquette avec la 1re étape de génération du stock cible*

Une fois la méthode de sélection choisie, il faut déterminer si le stock cible est prévu pour toutes les succursales ou seulement pour une seule. Il est toujours possible de revenir en arrière si nécessaire.

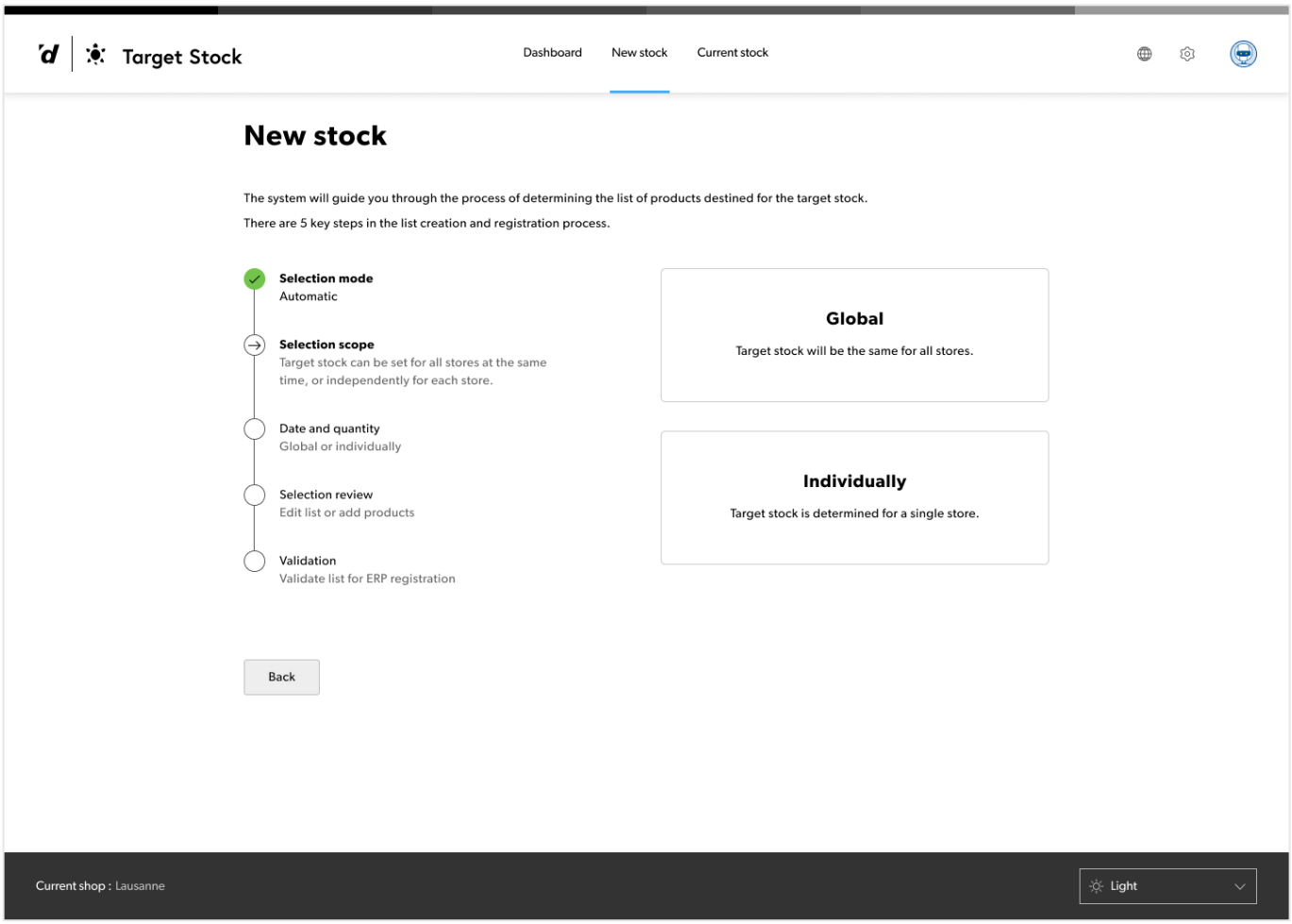


Figure 22 *- Version initiale de la maquette avec la 2e étape de génération du stock cible*

À un certain point, nous arrivons à la liste de produits générée par le système. Un problème de mise en page se pose alors : l'espace prévu n'est pas suffisant en raison de la place occupée par le schéma du processus. La liste de produits étant au cœur de la solution, il est nécessaire de trouver une autre manière de la mettre en avant et de rendre sa consultation simple et agréable. C'est pourquoi le design de cette version s'est arrêté à ce stade, et une redéfinition de la mise en page du contenu principal sera effectuée dans la version intermédiaire de la maquette.

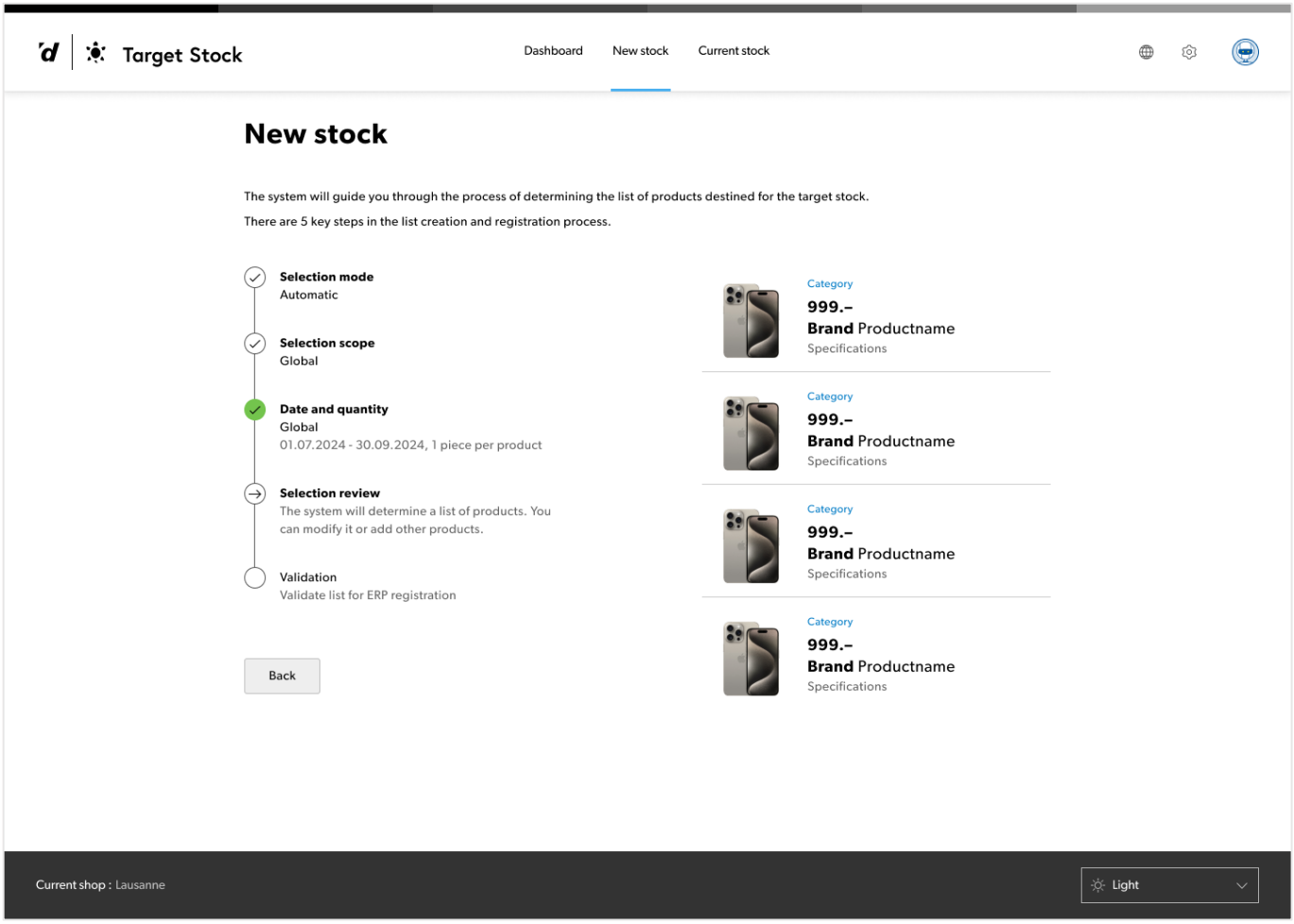


Figure 23 *- Version initiale de la maquette avec la 4e étape de génération du stock cible*

2.5.3 Version intermédiaire

À la suite de la première ébauche de la maquette, une nouvelle version plus ergonomique a été imaginée et créée avant d'être évaluée lors des tests utilisateurs. Cette nouvelle itération a été pensée pour correspondre au maximum au site e-commerce de l'entreprise afin de faciliter la prise en main des utilisateurs. Bien entendu, il y a eu des adaptations, car une solution permettant la gestion du stock cible est relativement éloignée d'un site de vente en ligne. Cependant, on y retrouve de nombreuses similitudes, notamment la façon d'afficher les produits ou encore l'emplacement de certaines options.

La première étape pour accéder à la solution est une page de connexion, qui permet de limiter l'accès aux employés de l'entreprise en obligeant l'utilisateur à utiliser son compte Microsoft. Il y aura également une vérification de l'adresse e-mail, qui doit être une adresse de l'entreprise. Enfin, l'utilisateur doit également être autorisé à accéder à l'application via un système supplémentaire de gestion des accès, qui sera détaillé plus loin dans ce rapport.

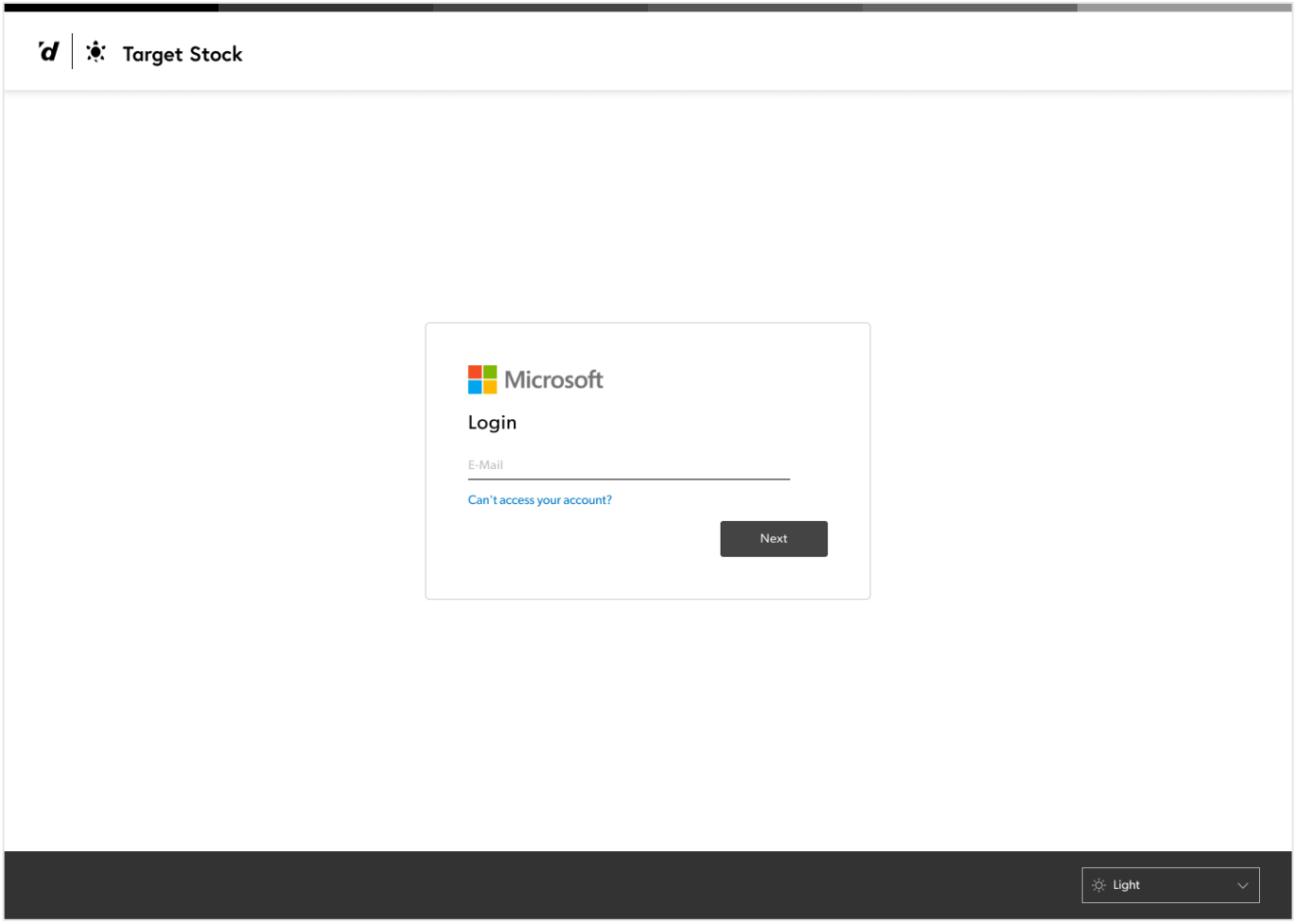


Figure 24 *- Page de connexion*

La première page de contenu du site est un tableau de bord, en raison de son intérêt relevé lors du sondage réalisé pendant la pré-étude. Ce tableau de bord met en lumière des informations et statistiques en lien avec le stock cible de la succursale sélectionnée. On peut notamment y retrouver le produit du stock le plus vendu, le temps restant pour le stock actuel ou encore la quantité d'articles restants. N'ayant pas d'autres idées au moment de réaliser cette page, des rectangles gris ont été ajoutés pour représenter de potentielles autres informations. Il sera possible de définir ces dernières après les tests utilisateurs, car cela fait partie des questions posées lors de ces derniers. Ici aussi, le minimalisme est de mise avec peu de texte et des symboles sensés illustrer les données.

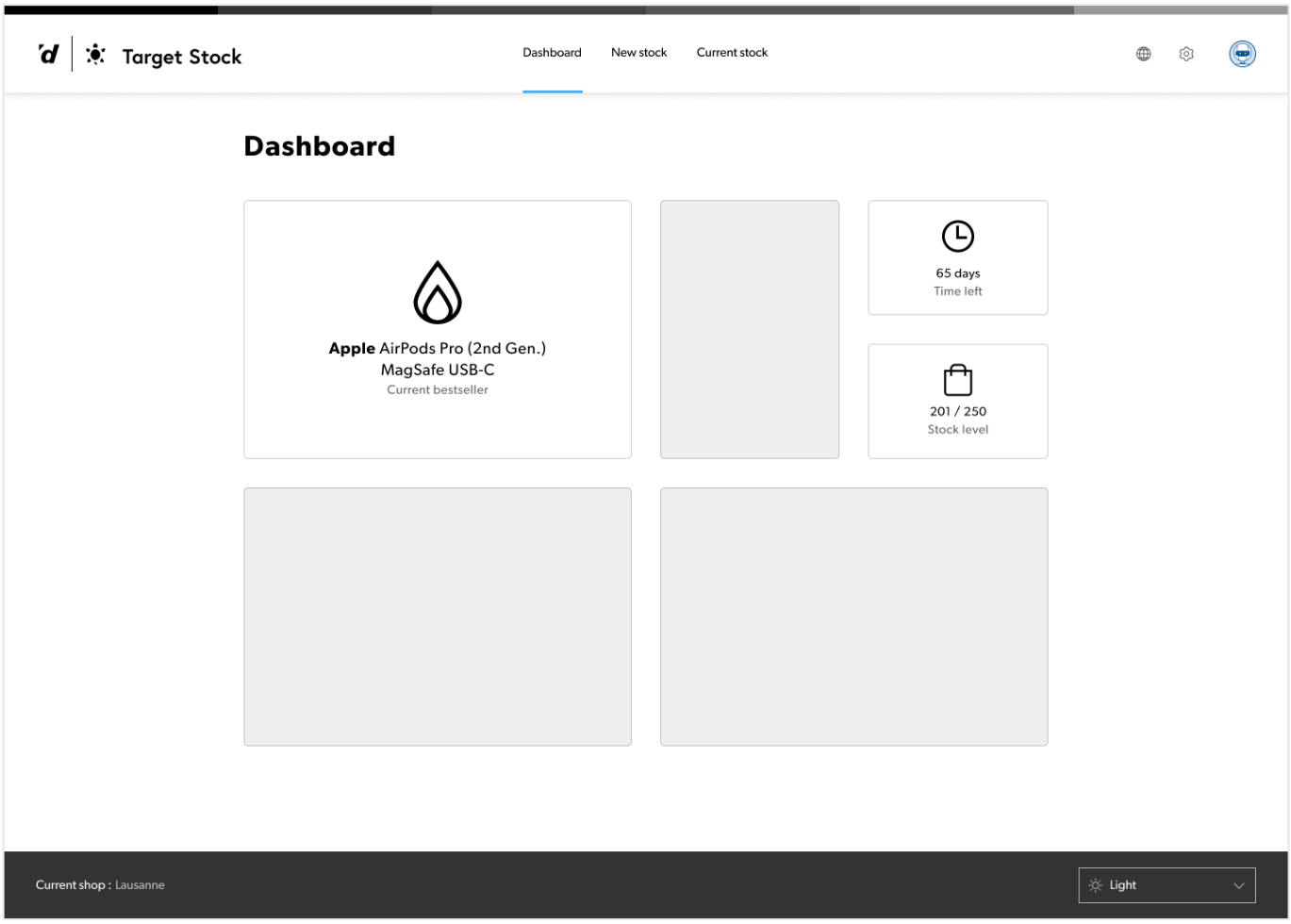


Figure 25 *- Page du tableau de bord de la maquette*

La page de création d'un nouveau stock a été simplifiée depuis la première itération, et on peut voir que le processus est passé de 5 à 3 étapes. Parmi ces dernières, il y a tout d'abord la sélection du mode de création du nouveau stock cible, représentée avec l'image ci-dessous. Le reste des pages de la solution va garder cette structure avec un titre principal, un texte de description de la page, un titre secondaire et enfin le contenu principal de la page.

Il y a quelques variations, notamment sur le haut de la page, comme ici avec le statut du processus, simplifié par rapport à la première itération. On pourra aussi retrouver un bouton dans cette zone en fonction du cas d'usage.

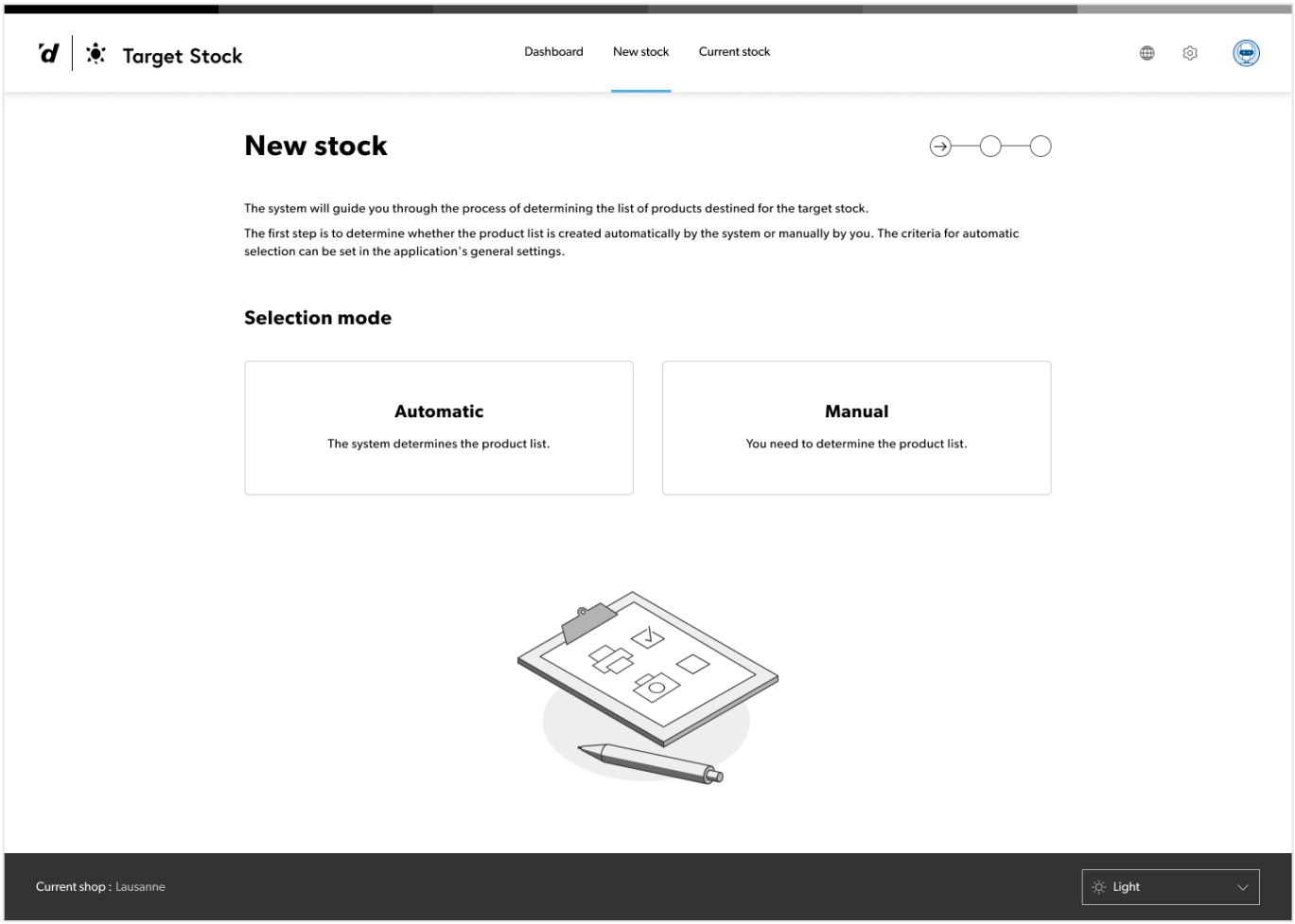


Figure 26 *- Page de sélection du mode de création du nouveau stock cible*

Sur cette première page du processus, il y a également le premier élément graphique créé spécialement pour ce projet. Il s'agit de l'illustration en bas de page, animée avec des icônes de produits qui changent toutes les x secondes. Une illustration présente dans le design system a servi de base pour cet élément visuel animé. Il s'agissait d'une image illustrant les évaluations et avis des clients, et les emojis ont été remplacés par des produits pour montrer que l'on crée une liste de produits. Parmi ces derniers, deux étaient déjà présents dans la liste d'icônes de l'entreprise, tandis que quatre ont été créés en s'inspirant des existants. La couleur a également été adaptée pour correspondre au thème général.

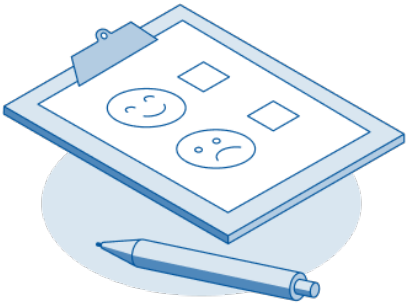


Figure 27 *- Illustration Digitec représentant les avis/retours des clients*

Il a fallu imaginer des icônes suffisamment simples pour correspondre au style existant des autres icônes, mais également reconnaissables afin d'éviter de devoir deviner leur signification. Ainsi, pour la partie Digitec, on trouve une icône d'ordinateur portable, une enceinte et une souris, en plus de l'appareil photo et de l'imprimante. Pour représenter l'assortiment de Galaxus, une raquette de ping-pong a été ajoutée.

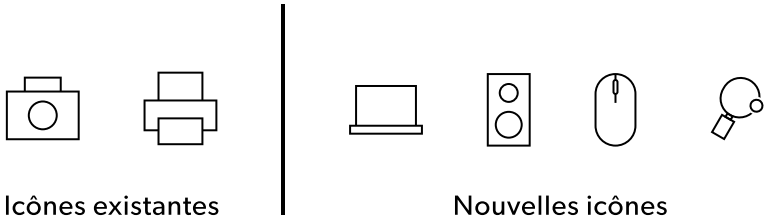


Figure 28 *- Icônes existantes et nouvellement créées pour l'illustration animée*

Avec deux icônes présentes par étape de l'animation, cette dernière se découpe donc en trois parties. Les images défilent toutes les deux secondes et recommencent indéfiniment en formant une boucle.

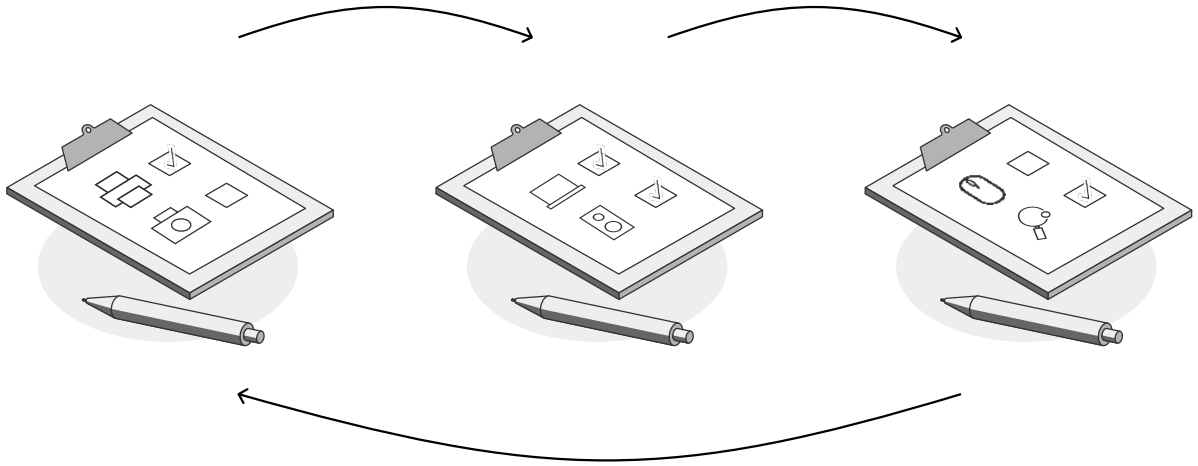


Figure 29 *- Ordre d'apparition des images dans l'illustration animée*

En choisissant le mode manuel pour la sélection des produits, nous arrivons sur une page contenant une liste vide et nous invitant à ajouter des produits pour afficher la liste. Une barre d'outils située juste au-dessus de la future liste permet de revenir en arrière, d'ajouter un produit, d'activer le mode « sélection multiple » et d'éditer les règles de plusieurs produits. Sur cette barre, on trouve également le nombre de produits contenus dans la liste et des boutons pour permuter l'affichage entre la vue en cases et la vue en liste, et vice-versa.

Enfin, dans la partie supérieure droite, se trouve le statut du processus, indiquant que nous sommes à l'étape intermédiaire et que l'étape précédente a été réussie, ainsi que deux boutons. Le premier, en forme d'engrenage, permet de définir les règles du stock cible telles que la ou les succursales visées, la période de disponibilité et la quantité à stocker par produit. Le deuxième bouton permet de continuer le processus en passant à l'étape suivante. Lorsque la liste est vide, ce bouton est désactivé, car il est impossible de continuer le processus sans produits.

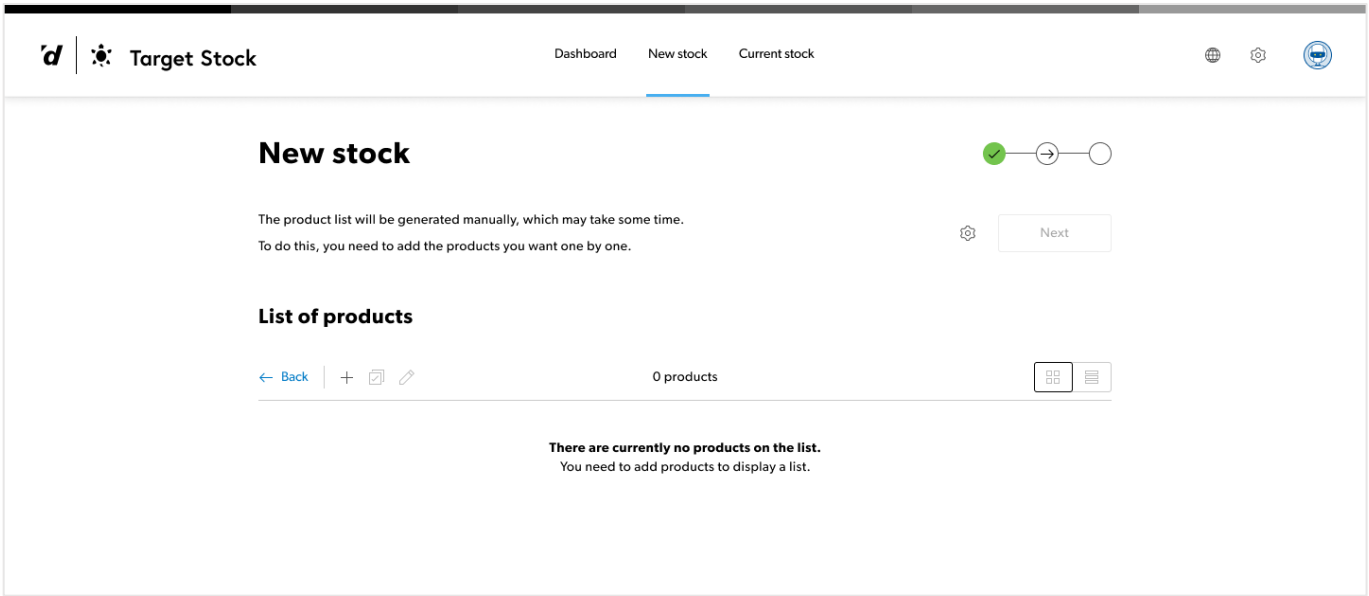


Figure 30 *- Liste de produits vide lors de la sélection manuelle*

Si l'on choisit de générer la liste de produits de manière automatique, la vue est similaire à la précédente, mais il y a un temps de chargement avant l'affichage des produits. Ce temps de chargement est illustré par un message au centre de la barre d'outils ainsi que par le squelette des produits, qui passe par plusieurs nuances de gris pour indiquer que les produits vont apparaître sous peu.

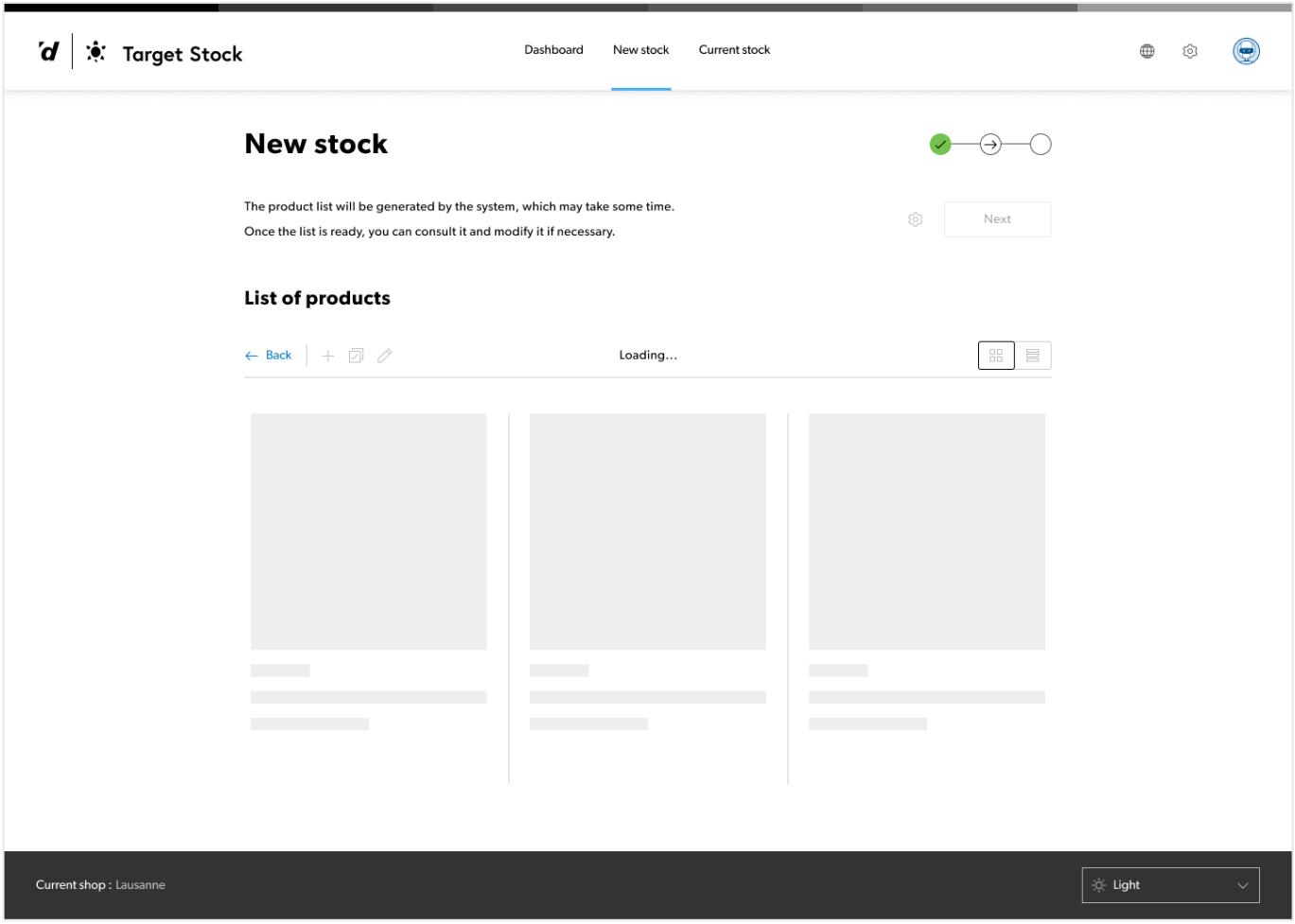


Figure 31 *- Chargement de la liste de produits lors de la sélection automatique*

Une fois le chargement terminé, les produits apparaissent sous forme de grille par défaut. Chaque produit est présenté avec une image, une catégorie, son prix, son nom avec la marque et quelques caractéristiques. Le nombre de produits générés est affiché au milieu de la barre d'outils. Il est désormais possible d'utiliser les différents outils mis à disposition, comme ajouter un produit ou en modifier plusieurs via la sélection multiple.

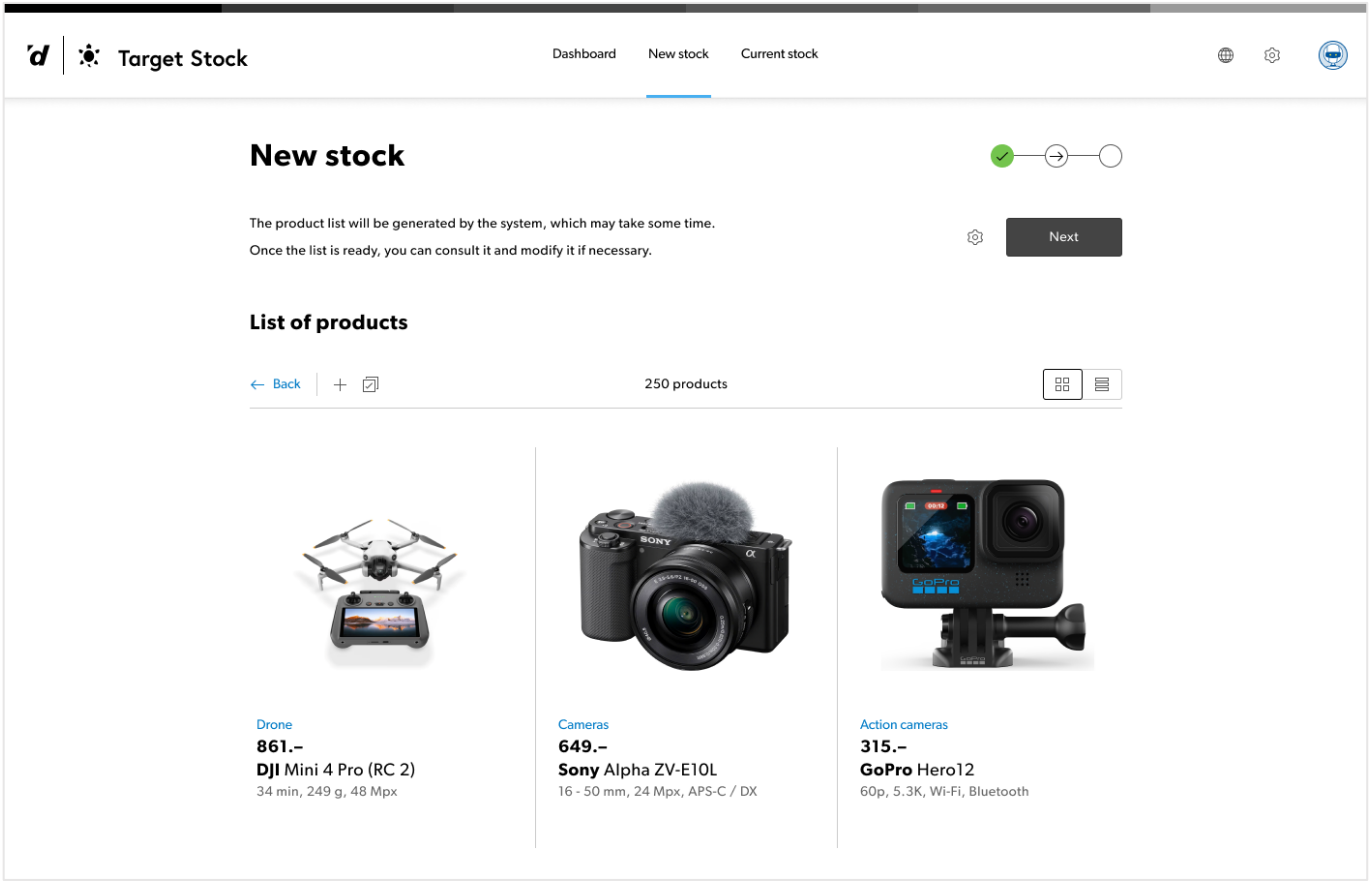


Figure 32 *- Début de la liste de produits lors de la sélection automatique*

La liste continue jusqu'en bas de la page, où une barre de progression indique la proportion de produits affichés par rapport au total. Il est ainsi possible de cliquer sur un bouton pour afficher davantage de produits, à la manière de ce qu'on retrouve sur Digitec et Galaxus. Enfin, un lien permet de remonter en haut de la liste afin d'éviter de devoir faire défiler la page manuellement.

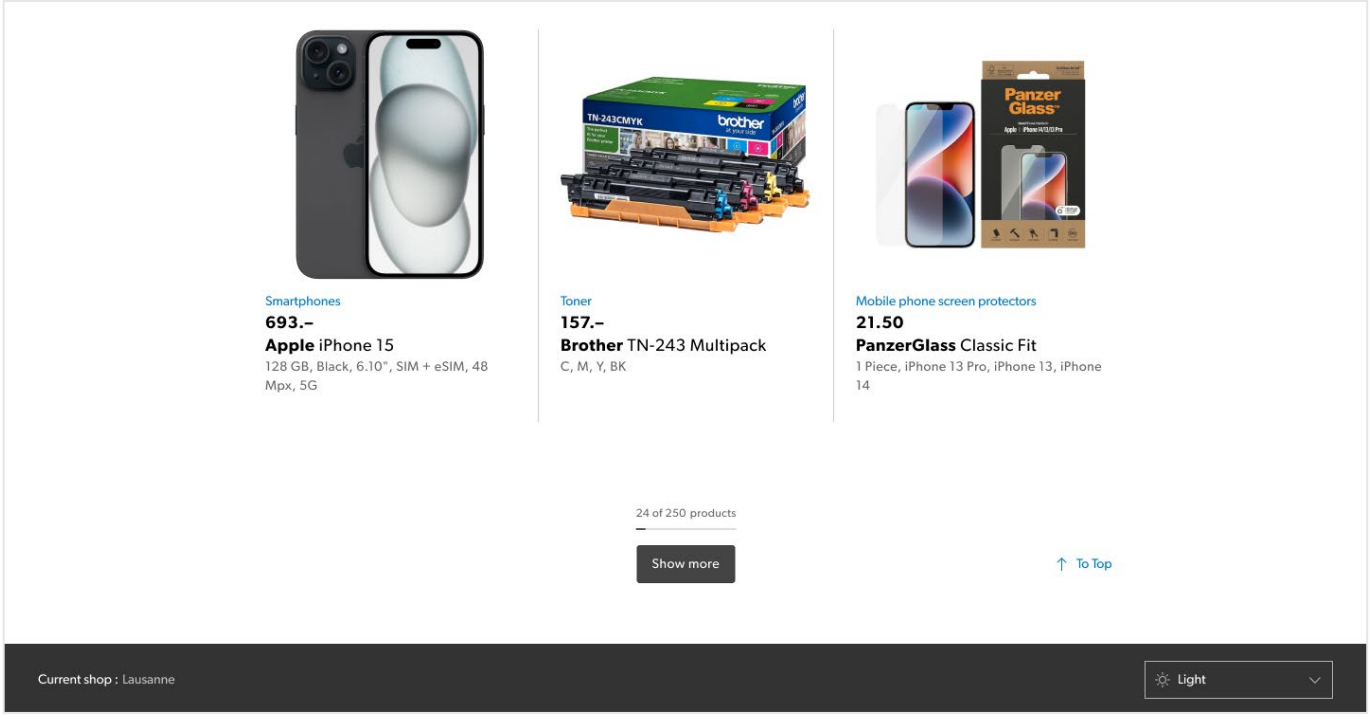


Figure 33 *- Fin de la liste de produits lors de la sélection automatique*

Comme dit précédemment, la mise en page de la liste de produits peut être modifiée selon l'envie de l'utilisateur pour passer de la vue en cases à la vue en liste. Ces mises en page sont directement reprises des plateformes de vente de l'entreprise.

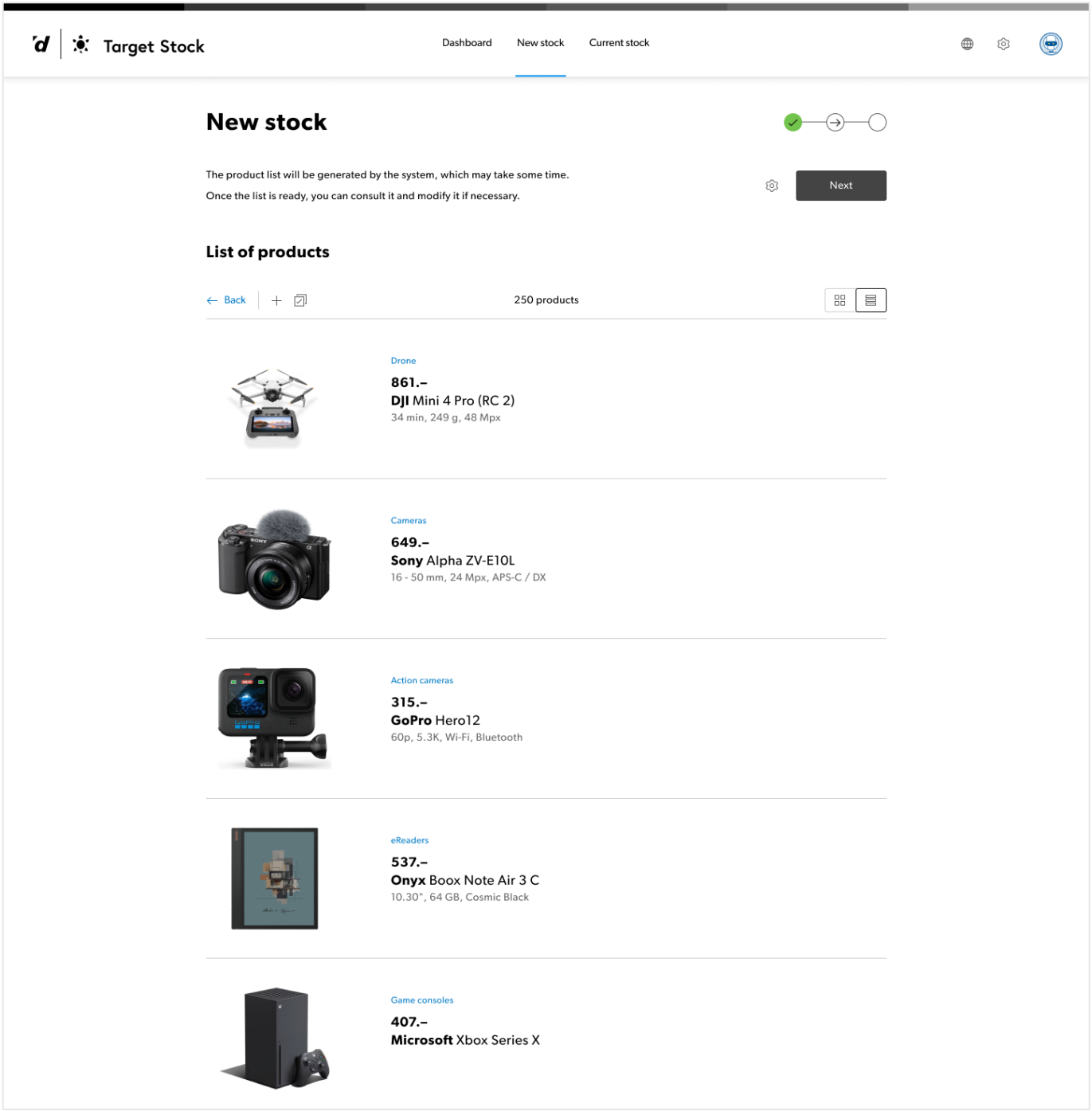


Figure 34 *- Vue des produits avec la mise en page définie sur "liste"*

Les outils présents dans la barre d'outils sont facultatifs à ce stade du processus, mais il y a une étape importante qu'il ne faut pas oublier de poursuivre : paramétrer les règles du stock cible. Il suffit de cliquer sur le bouton avec l'icône d'engrenage pour faire apparaître ce menu. Ce dernier permet de choisir dans quelle succursale seront disponibles les produits, avec la possibilité de les rendre disponibles dans toutes les succursales, de déterminer la période de disponibilité et enfin d'attribuer une quantité à stocker pour chaque produit.

Cette étape est essentielle, car, sans ces informations, il ne serait pas possible de procéder à l'enregistrement dans l'ERP. Comme indiqué dans le texte descriptif de ce menu, il s'agit de paramètres globaux qui seront appliqués à l'ensemble des produits, mais il est également possible de les définir individuellement par produit ou pour une sélection de produits.

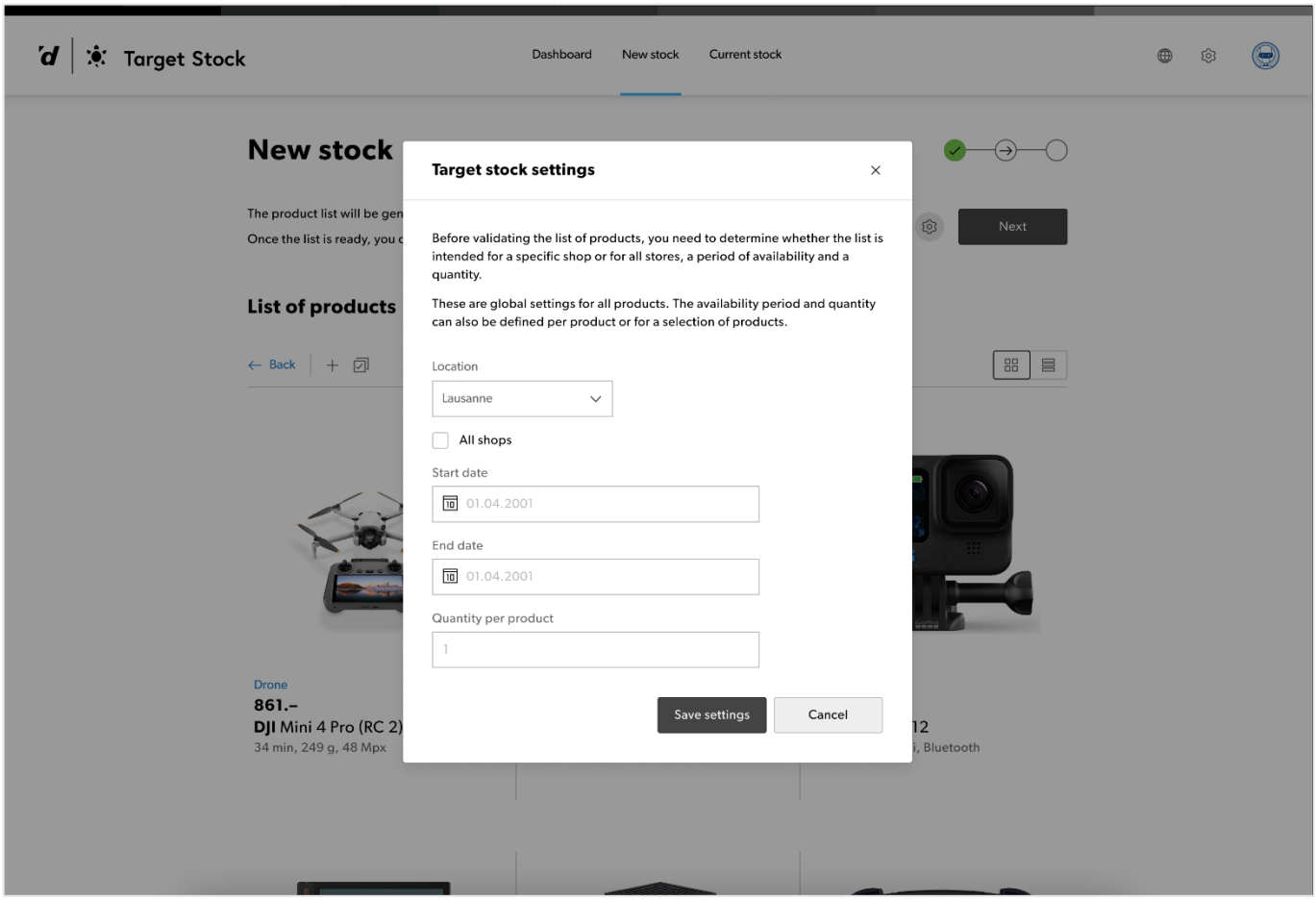


Figure 35 *- Fenêtre contextuelle des réglages du stock cible*

C’est justement un des outils disponibles dans la barre d’outils. La sélection multiple permet, comme son nom l’indique, de sélectionner plusieurs produits afin de les supprimer ou de modifier leurs paramètres. Il suffit de cliquer sur un produit pour le sélectionner.

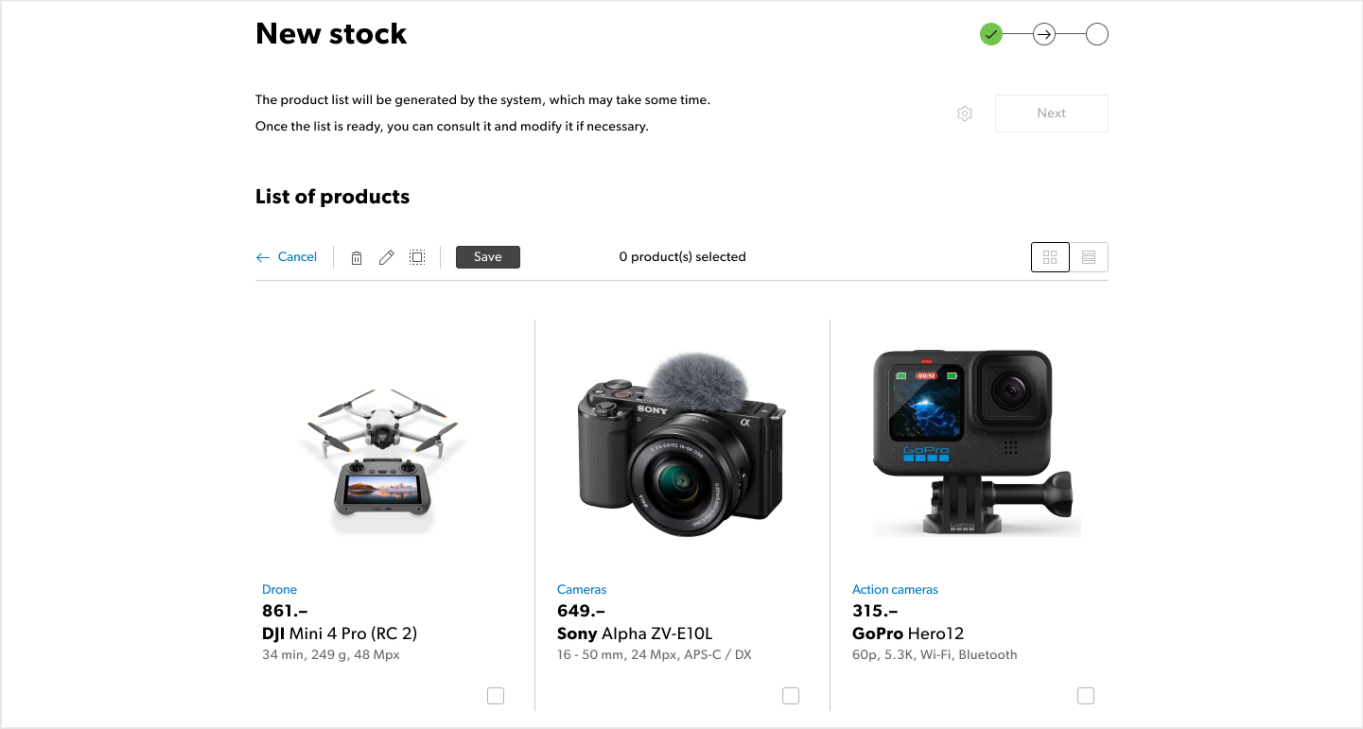


Figure 37 *- Vue de la liste de produits avec la sélection multiple activée*

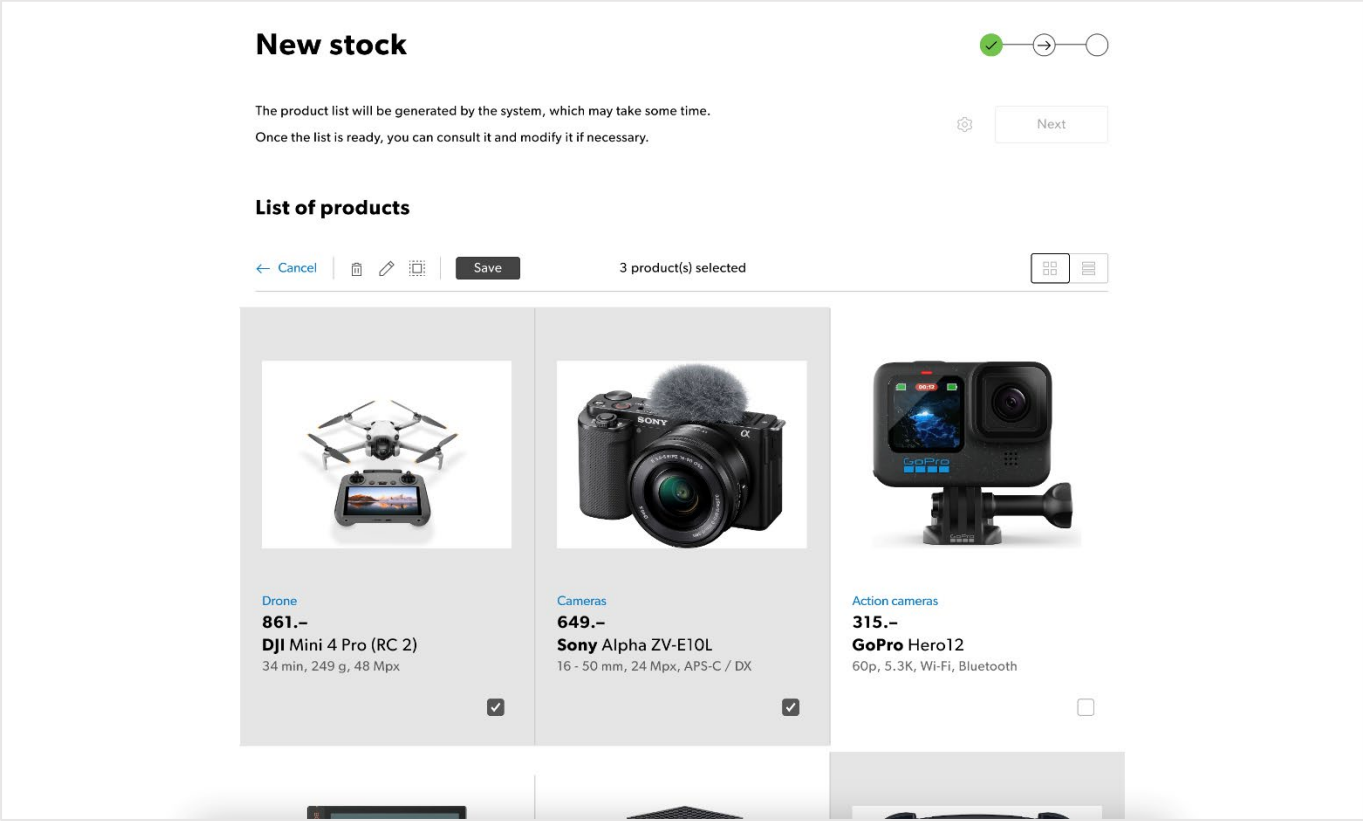


Figure 36 *- Vue de la liste de produits lorsque plusieurs d’entre eux sont sélectionnés*

Une fois la sélection effectuée, il est possible de supprimer ces produits de la liste en utilisant le bouton en forme de corbeille présent dans la barre d'outils. Lors du clic sur ce bouton, un petit message apparaît en bas à droite pour confirmer la suppression.

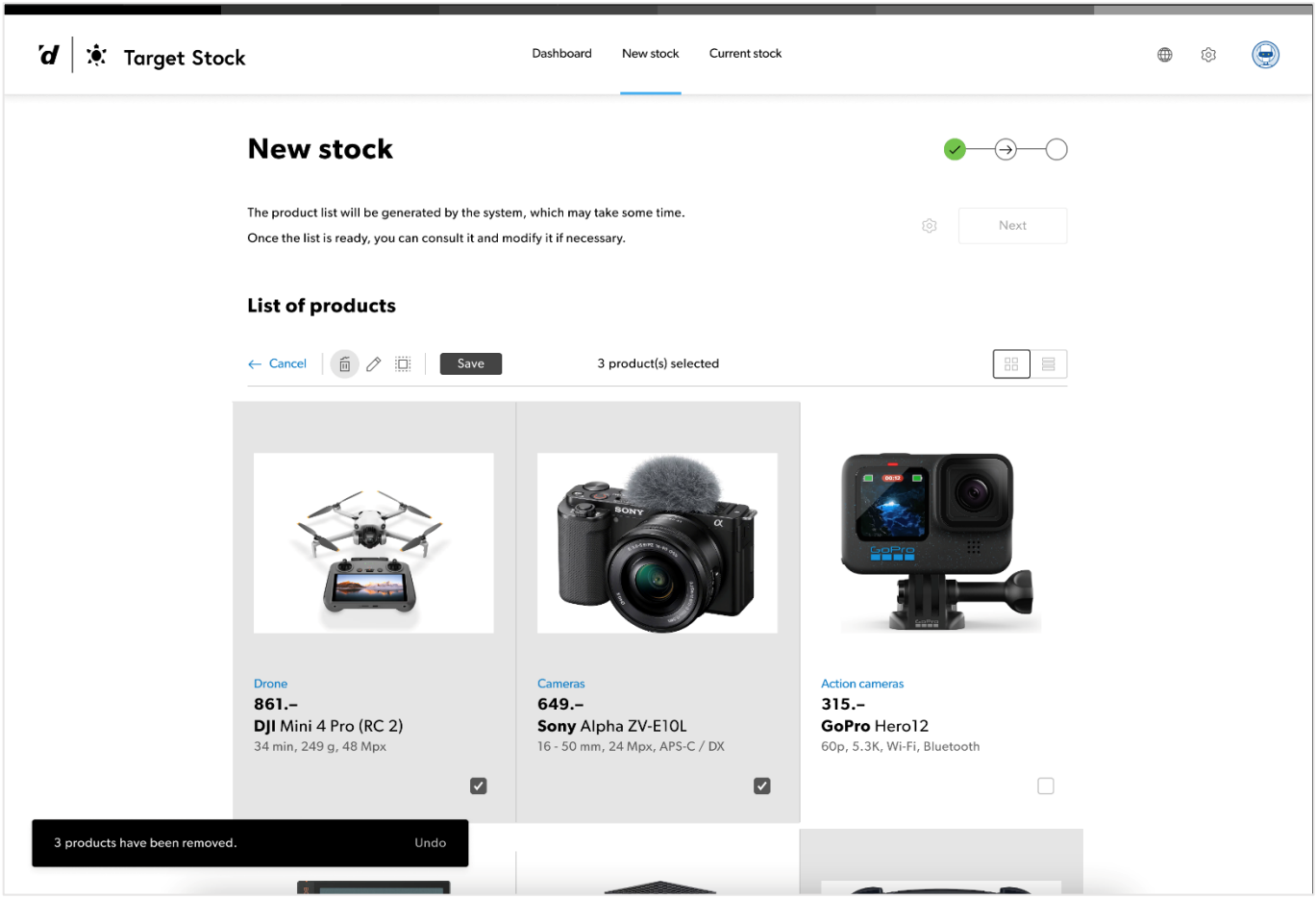


Figure 38 *- Message temporaire informant la suppression de plusieurs produits*

Hormis la suppression de plusieurs produits, il est également possible, comme énoncé précédemment, de modifier les règles du stock cible pour une sélection de produits. Cette fonction est représentée par une icône de crayon dans la barre d'outils et ouvre une fenêtre lorsque l'on clique dessus. On peut ainsi définir une période de validité ainsi qu'une quantité différente pour les produits sélectionnés.

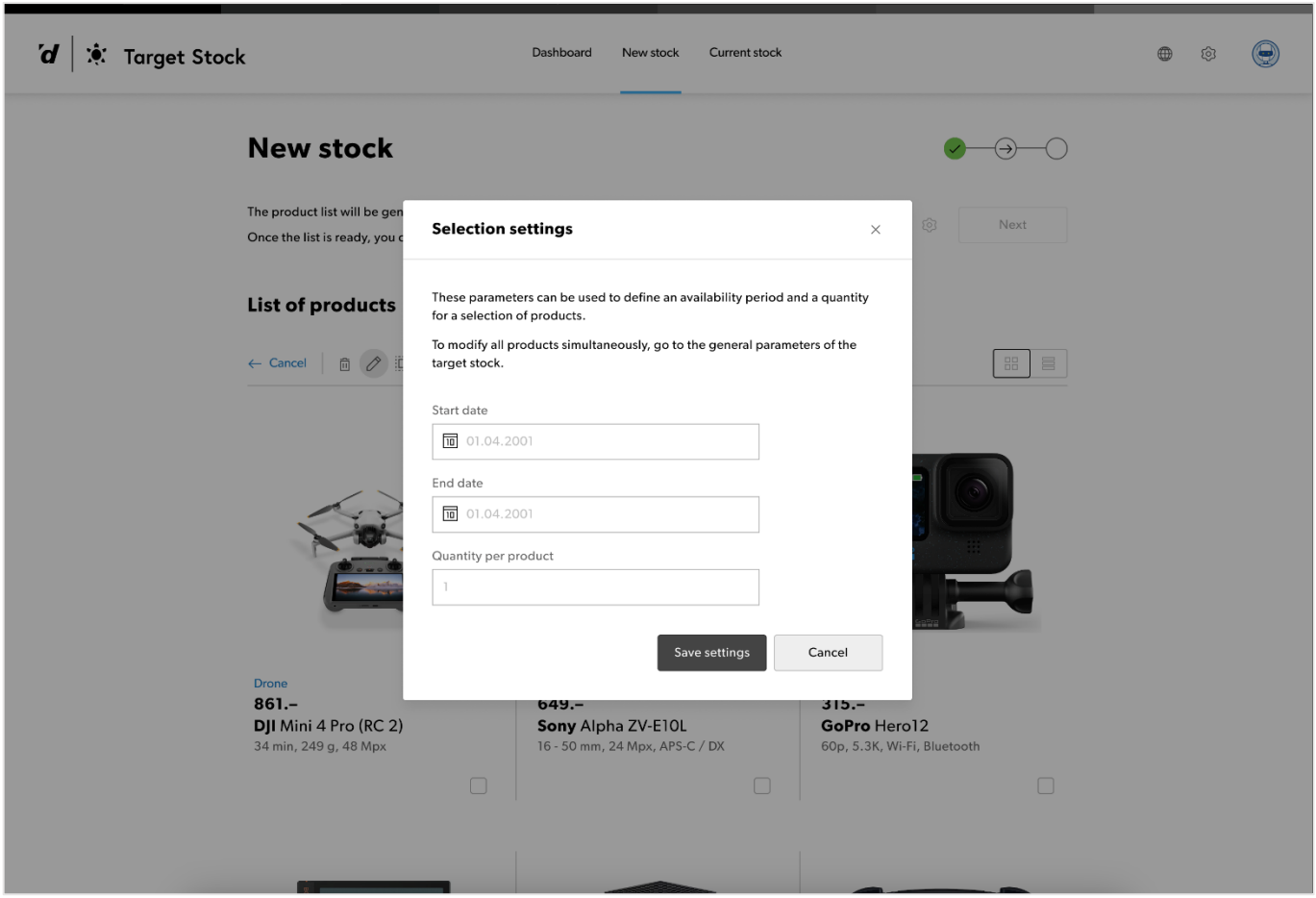


Figure 39 *- Fenêtre contextuelle des réglages d'une sélection de produits*

Une dernière fonction est possible en mode sélection multiple : la possibilité de tout sélectionner. Cela pourrait paraître inutile étant donné qu'il y a les réglages globaux qui permettent d'appliquer les règles à tous les produits. Cependant, cette fonction a du sens si l'on souhaite appliquer des règles sur une grande quantité de produits sans affecter l'entièreté de ces derniers. Il est alors possible de tout sélectionner et de décocher les éléments que l'on ne veut pas affecter. Cette fonction est représentée par une icône spécialement créée pour l'occasion, un carré entouré d'une seconde bordure en traits discontinus.

Quittons la sélection multiple pour revenir à la vue initiale de la liste de produits. Il reste quelques fonctions utiles au paramétrage et à l'affinage de la liste. Il est possible d'ajouter des produits si l'on trouve qu'il en manque ou que l'on ne retrouve pas les produits souhaités. En cliquant sur le bouton "+", une fenêtre s'ouvre en nous demandant d'entrer le lien du produit sur Galaxus pour l'ajouter à la liste.

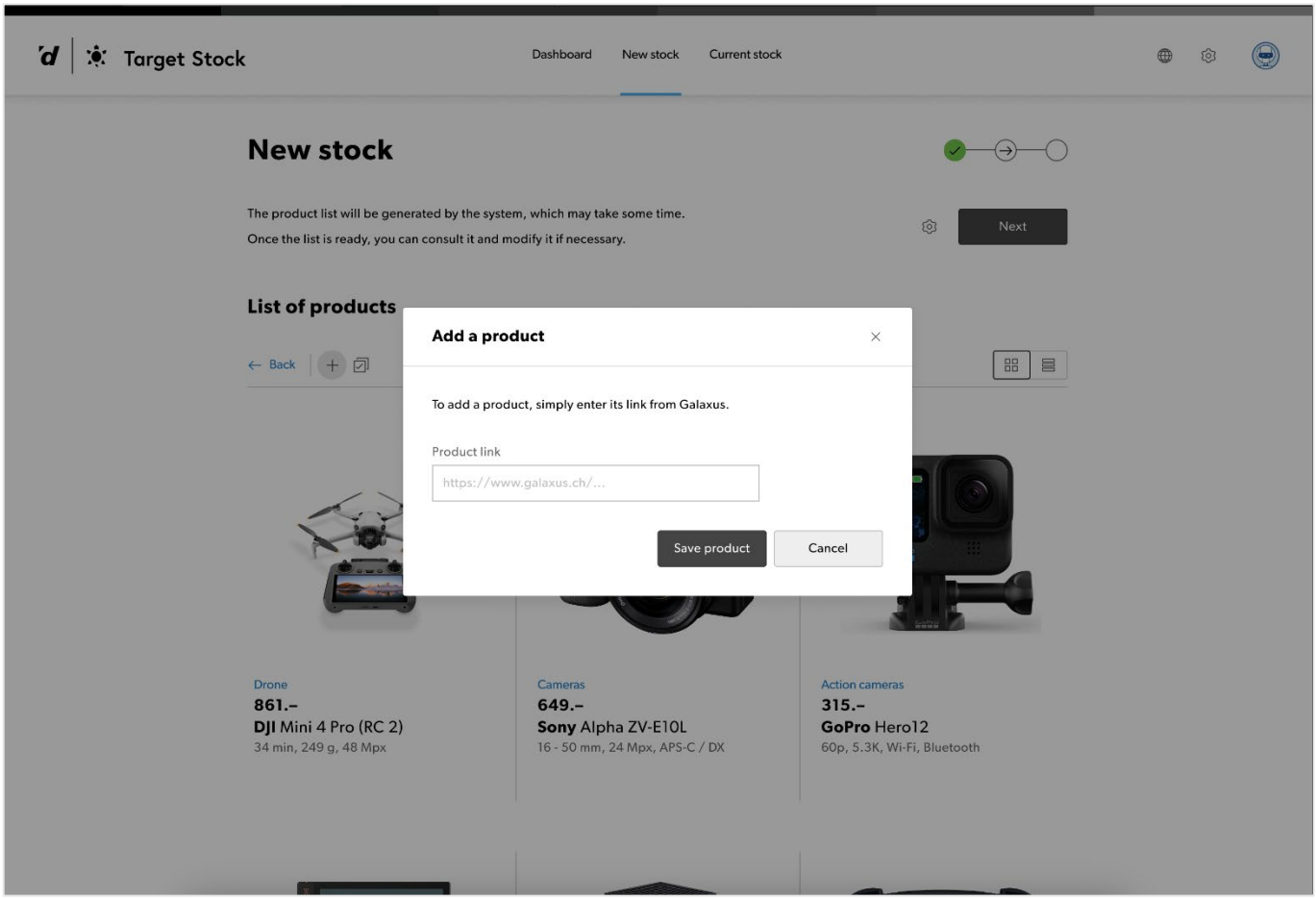


Figure 40 *- Fenêtre contextuelle permettant l'ajout de produit à la liste*

Une fois le produit ajouté, un message de succès apparaît avec la possibilité d'en ajouter un autre. Ce processus se répète jusqu'à ce que l'on ferme cette fenêtre.

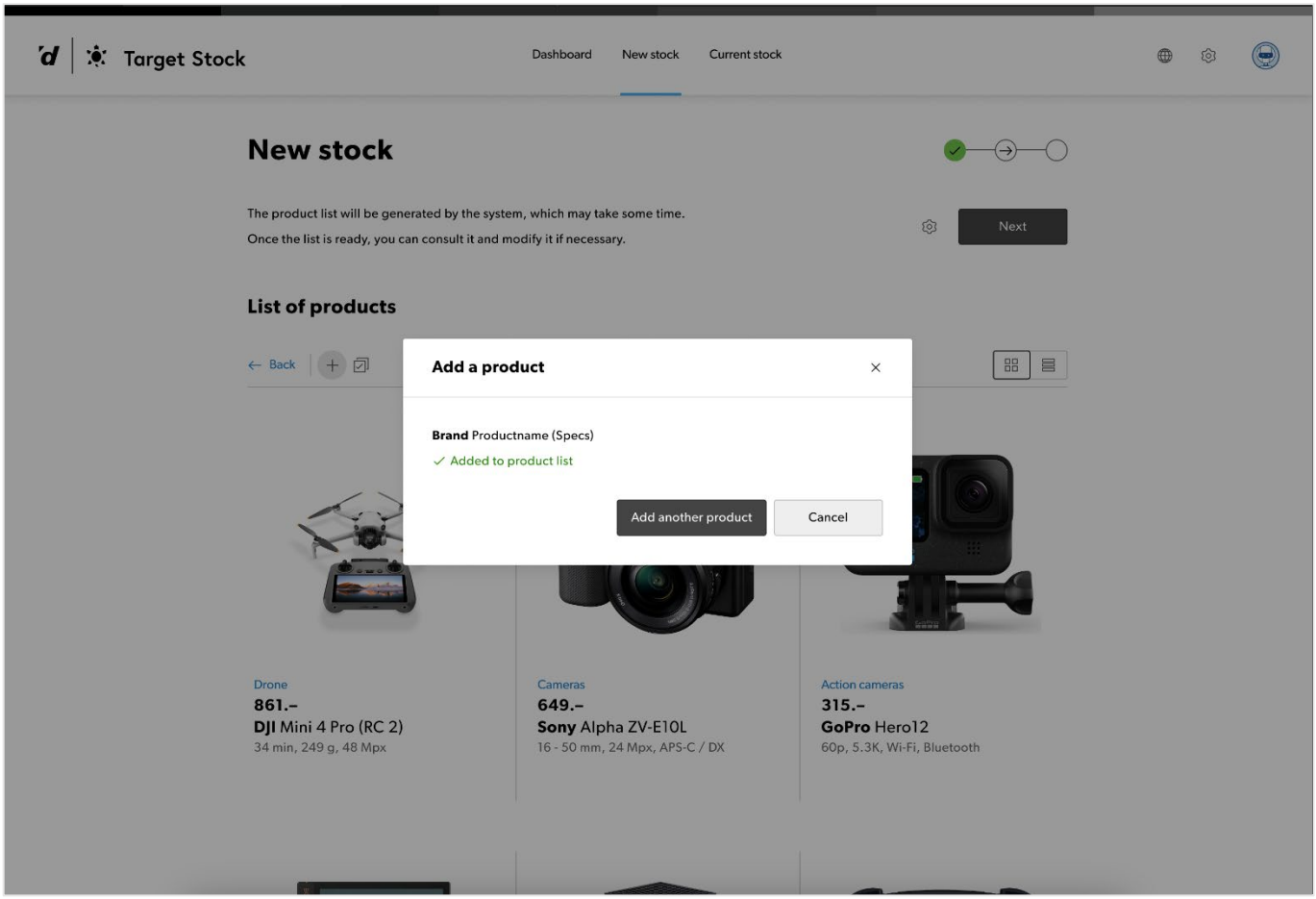


Figure 41 *- Message de confirmation d'ajout d'un nouveau produit*

Enfin, il est possible d'accéder aux options individuelles des produits en les survolant. Cela permet de supprimer le produit, de modifier ses règles et d'accéder à la page du produit sur Galaxus ou l'ERP. Cette fonction de survol reprend son design des plateformes de vente, qui affichent des options comme l'ajout au panier lors du survol.

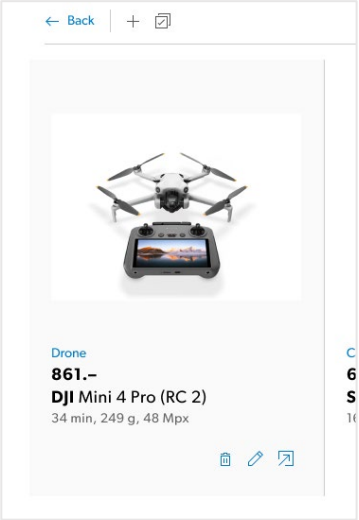


Figure 42 *- Affichage des options individuelles lors du survol d'un produit*

Pour supprimer un produit, il suffit de cliquer sur l'icône de corbeille. Un petit message en bas à droite de l'écran confirmera sa suppression avec la possibilité d'annuler cette action, comme avec l'option similaire dans la sélection multiple.

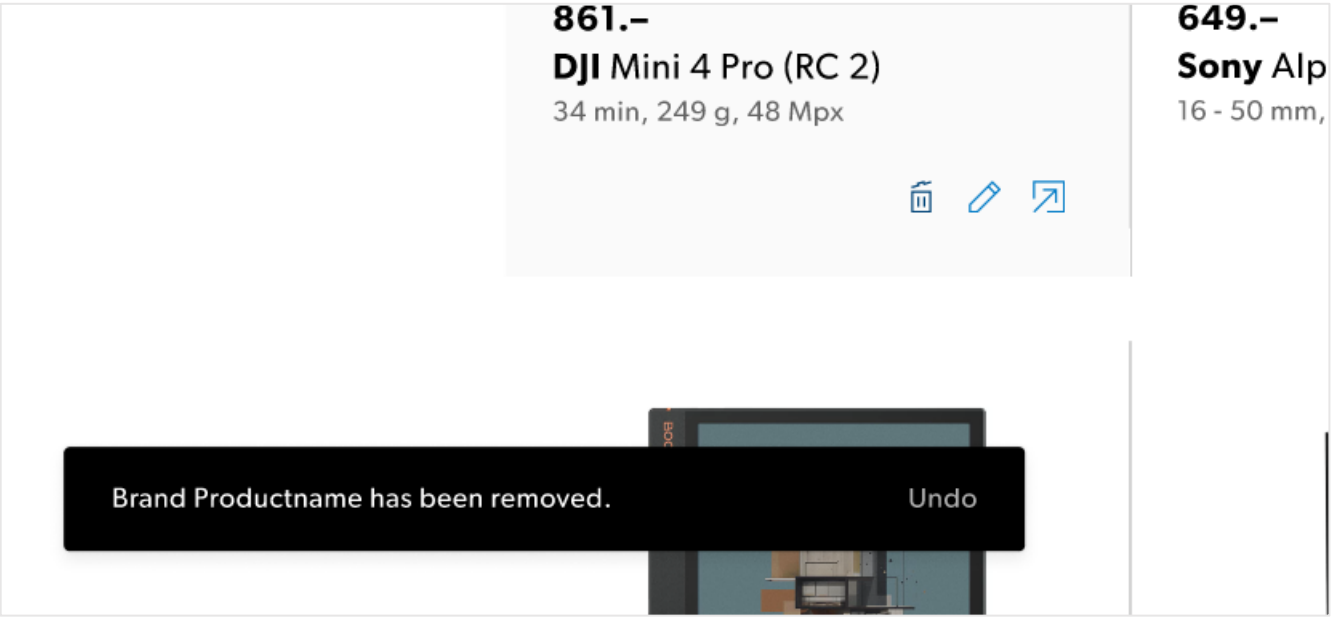


Figure 43 *- Message temporaire informant la suppression d'un produit*

La modification des règles se fait en cliquant sur l'icône de crayon, ce qui ouvre une fenêtre où vous pouvez ajuster les paramètres.

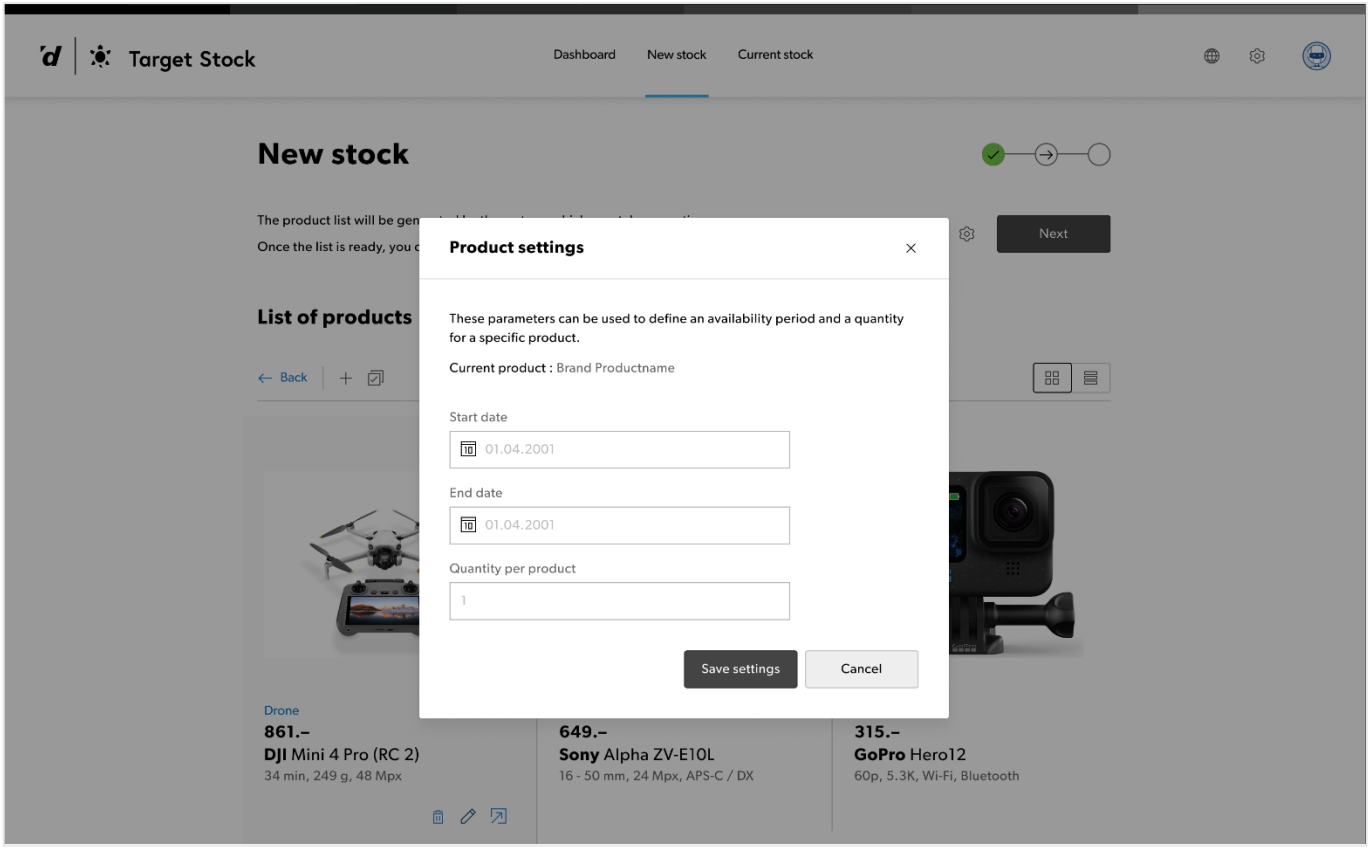


Figure 44 *- Fenêtre contextuelle des réglages d'un produit*

Pour accéder à la page du produit, que ce soit sur Galaxus ou l'ERP, il suffit de cliquer sur l'icône de carré avec une flèche à l'intérieur. Une fenêtre s'ouvrira alors, vous demandant sur quelle plateforme vous souhaitez ouvrir la page du produit.

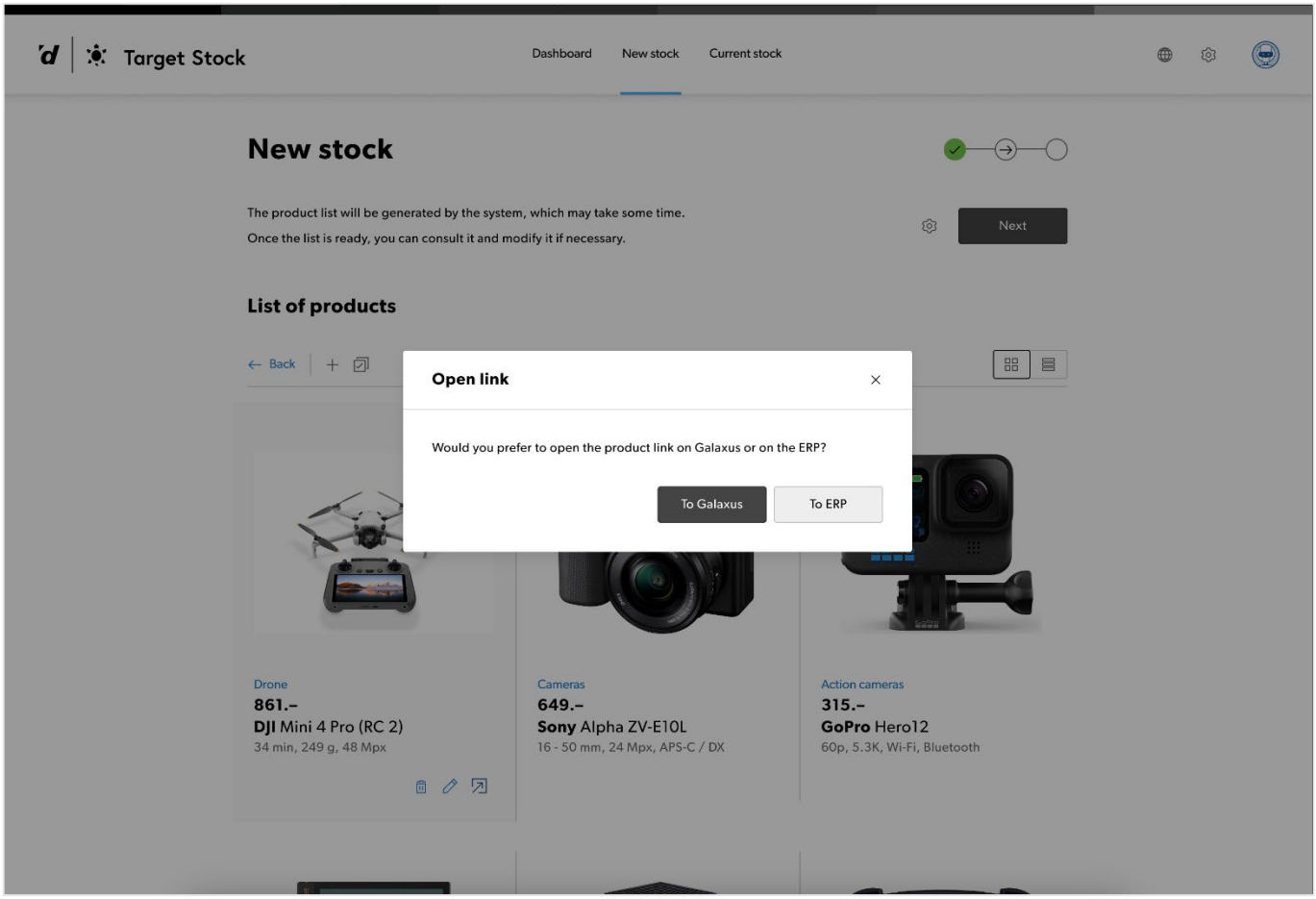


Figure 45 *- Fenêtre contextuelle permettant l'accès à la page d'un produit sur Galaxus ou l'ERP*

Une fois que nous sommes satisfaits de la liste de produits et que tous les réglages ont été effectués, nous pouvons continuer le processus en appuyant sur le bouton “Next”. L’utilisateur arrivera alors sur une page de résumé lui montrant quelques informations clés avant de poursuivre. Cela permet de procéder à une dernière vérification afin de s’assurer qu’il n’y ait aucune erreur. Si tout est correct, nous pouvons continuer. Dans le cas contraire, il est possible de retourner en arrière si besoin.

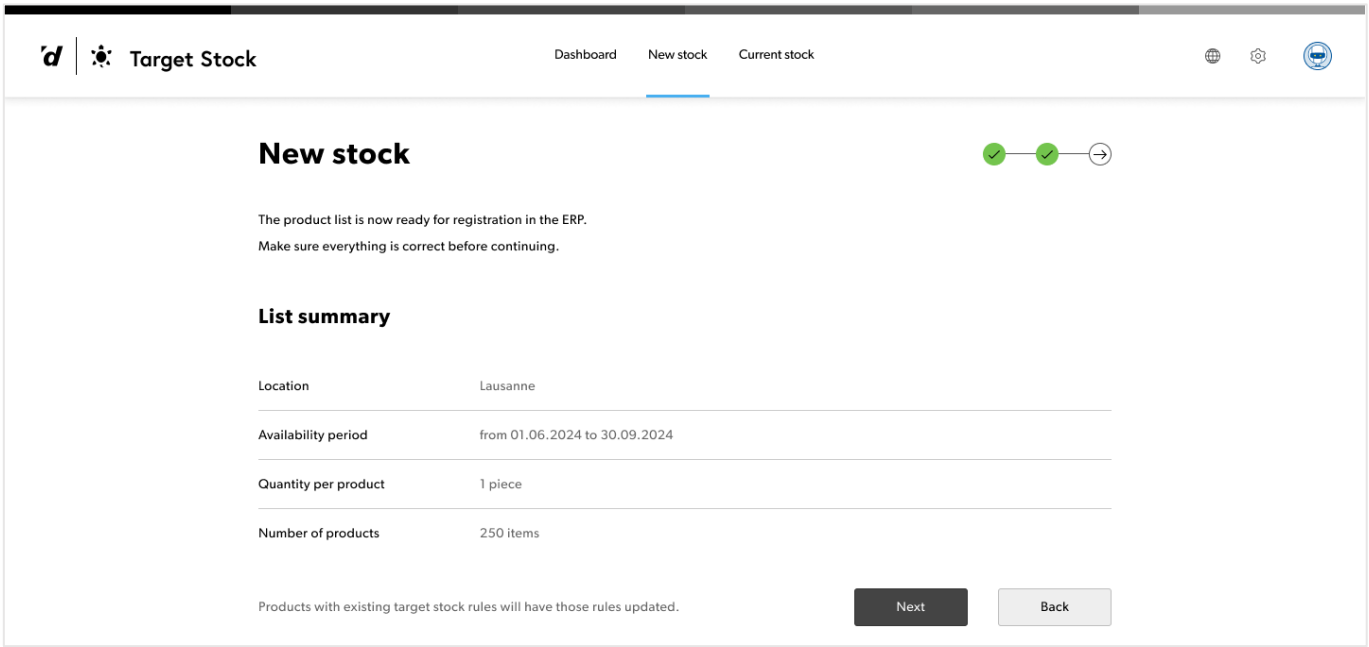


Figure 46 *- Page résumant les détails du stock cible avant son enregistrement dans l’ERP*

En continuant le processus, nous arrivons à la dernière étape : l’enregistrement des produits dans l’ERP. Une barre de chargement indique l’état d’avancement de cette tâche, qui peut s’avérer assez longue. En effet, chaque produit doit être ouvert dans l’ERP pour ajouter les différentes règles du stock cible. Plus il y a de produits, plus le processus sera long. Pour les besoins de la maquette, le temps de chargement simulé est très court afin de ne pas bloquer la prise en main.

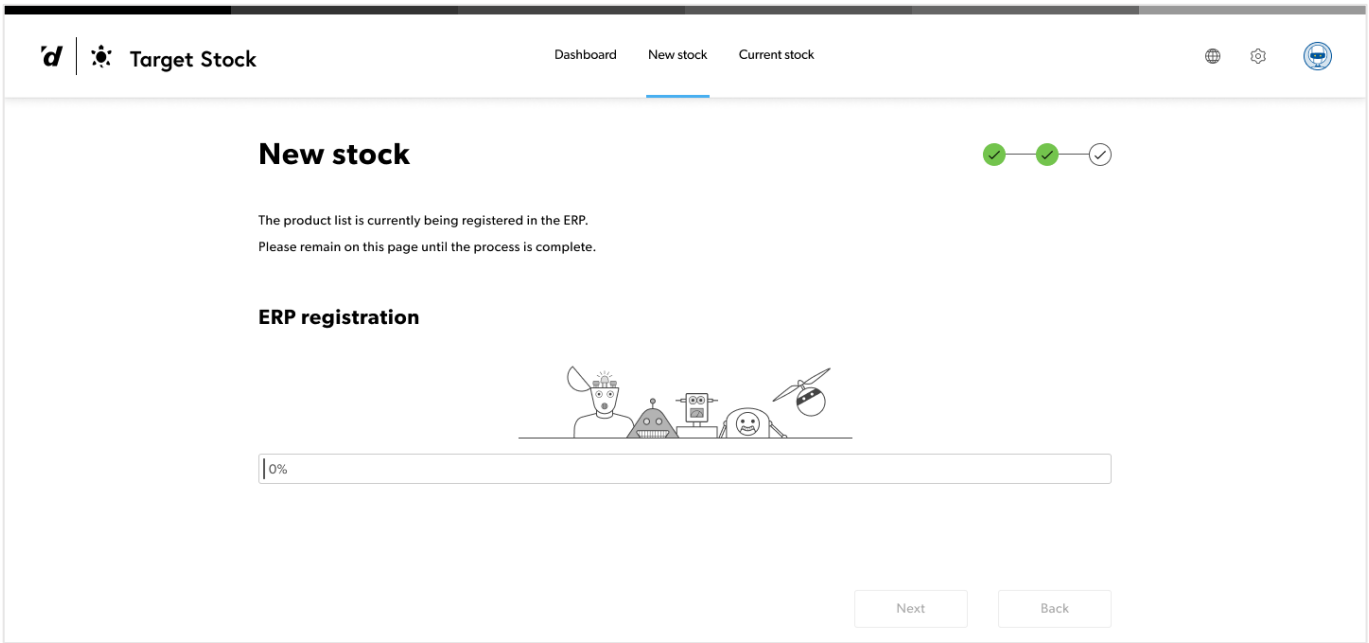


Figure 47 *- Page de chargement lors de l’enregistrement des produits dans l’ERP*

Un deuxième élément graphique animé est présent lors du processus d'enregistrement. Comme pour l'autre illustration animée, il n'a pas réellement de but fonctionnel. Il s'agit surtout de rendre l'interface plus vivante, surtout lors d'une étape comme celle-ci qui peut s'avérer très longue.

Cette animation reprend une illustration présente sur Digitec, avec des couleurs adaptées pour correspondre au thème sobre de la solution. L'animation se traduit par de petits personnages qui se déplacent verticalement, passant d'une image à l'autre tout en formant une boucle.

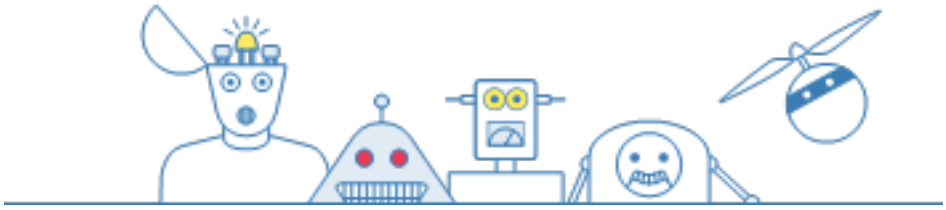


Figure 49 *- Illustration Digitec représentant la rubrique communauté*

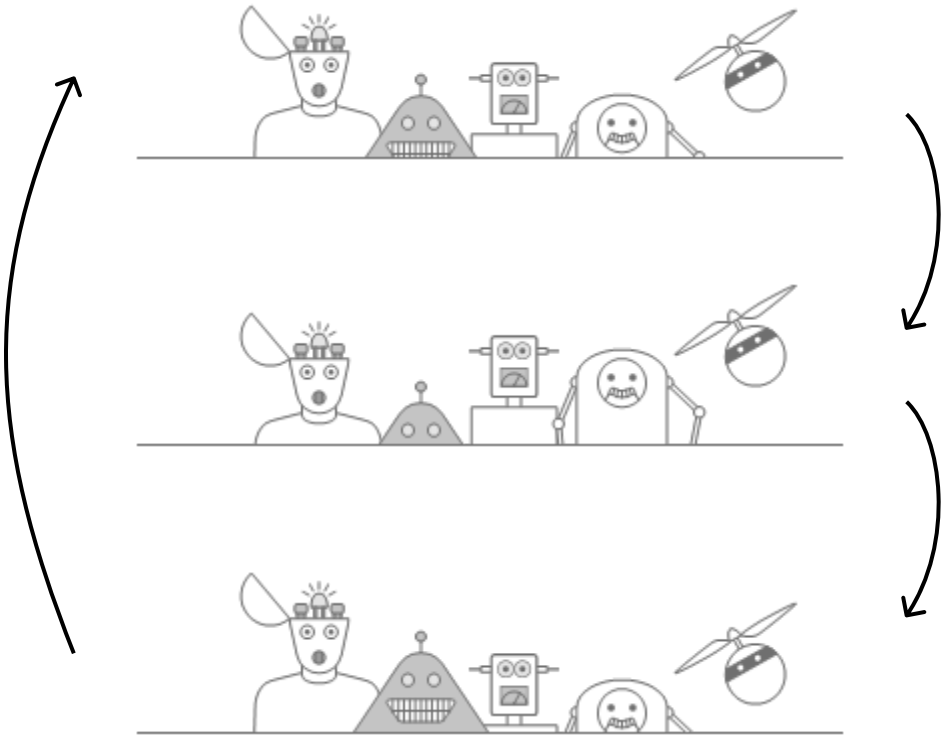


Figure 48 *- Ordre d'apparition des images dans l'illustration animée*

Une fois le processus terminé et en cas de réussite de l'opération, un message de succès sera affiché, indiquant à l'utilisateur que tous les produits ont été enregistrés dans l'ERP et qu'il peut à présent quitter cette page. Un raccourci vers le stock actuel est présent dans ce même message pour rediriger l'utilisateur vers cette rubrique.

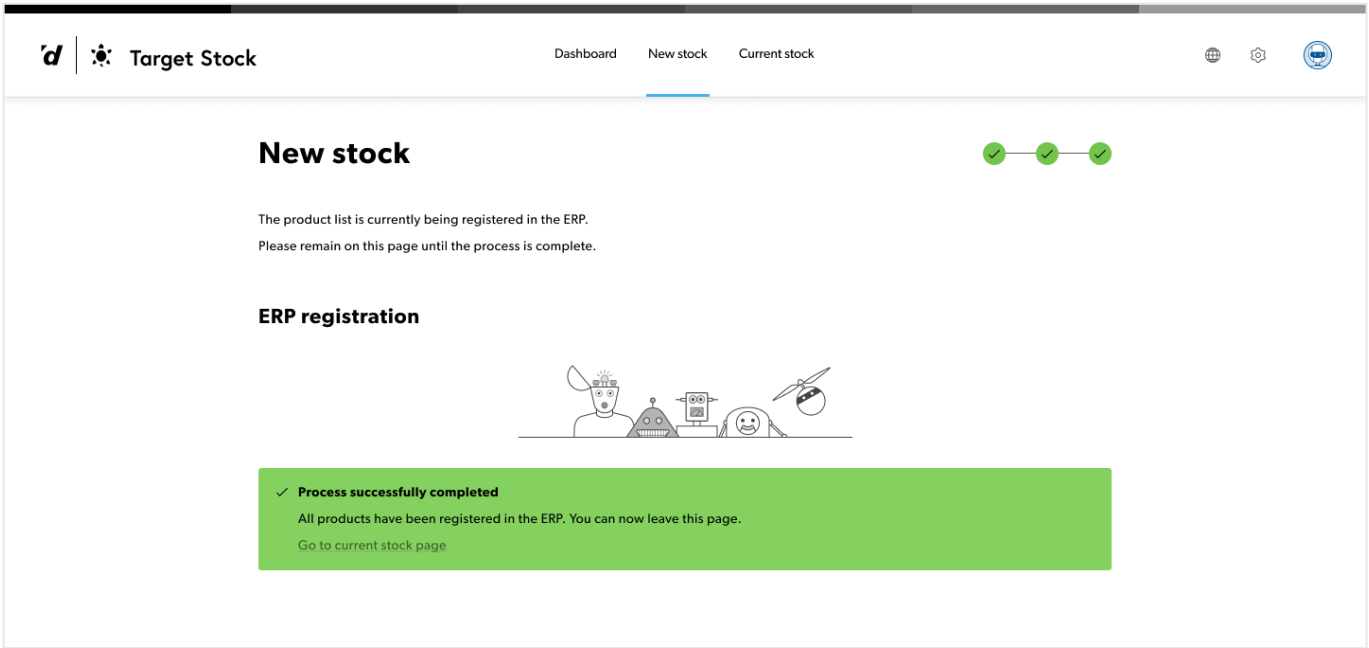


Figure 50 *- Message de succès lors de la réussite de l'enregistrement des produits dans l'ERP*

En cas de problèmes durant le processus d'enregistrement, un message d'erreur sera affiché, indiquant à l'utilisateur que quelque chose ne s'est pas passé comme prévu et qu'il peut relancer le processus directement.

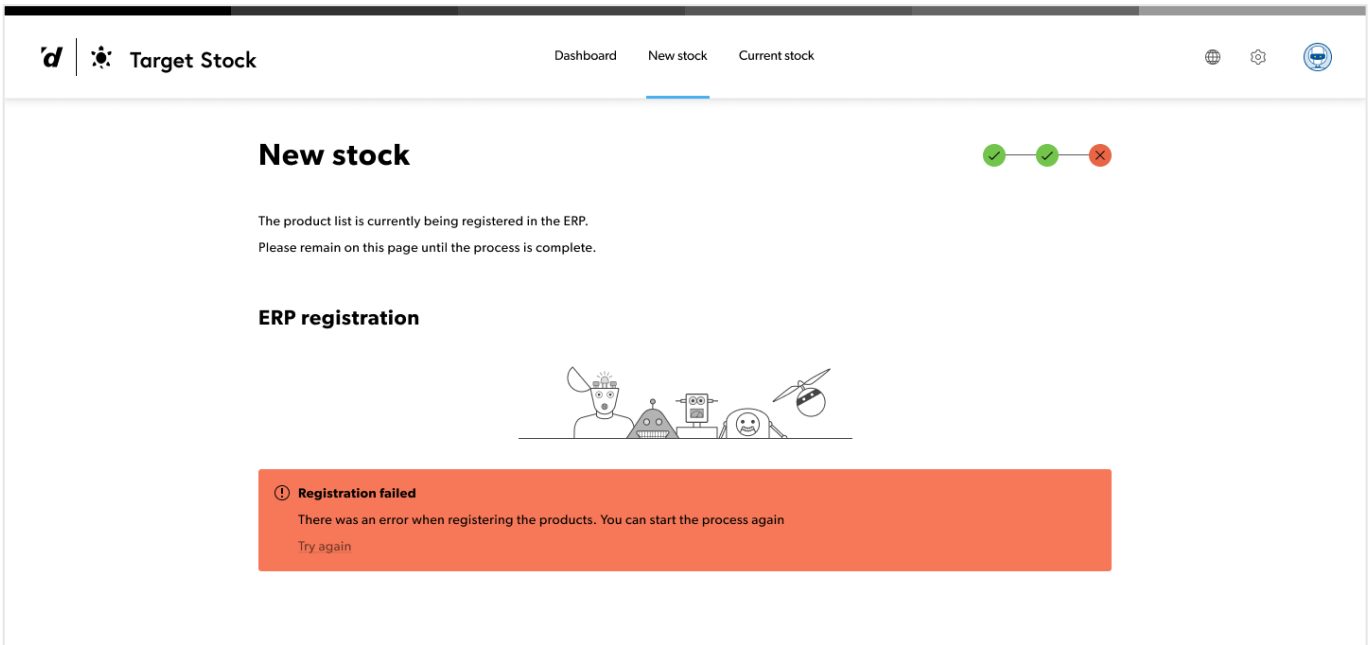


Figure 51 *- Message d'erreur à la suite d'un problème survenu lors de l'enregistrement des produits dans l'ERP*

Une fois la nouvelle liste établie, il est possible de la retrouver dans l'onglet dédié au stock actuel. Ce dernier reprend la mise en page vue précédemment, avec une liste de produits affichable sous forme de cases ou de liste. Il est également possible ici de modifier la liste après coup afin de laisser le plus de flexibilité possible aux utilisateurs, au cas où ils penseraient à un produit à ajouter ou à supprimer après avoir défini la nouvelle liste. Globalement, les mêmes outils sont disponibles : l'ajout de produits, la sélection multiple, les réglages du stock et les options disponibles au survol des produits.

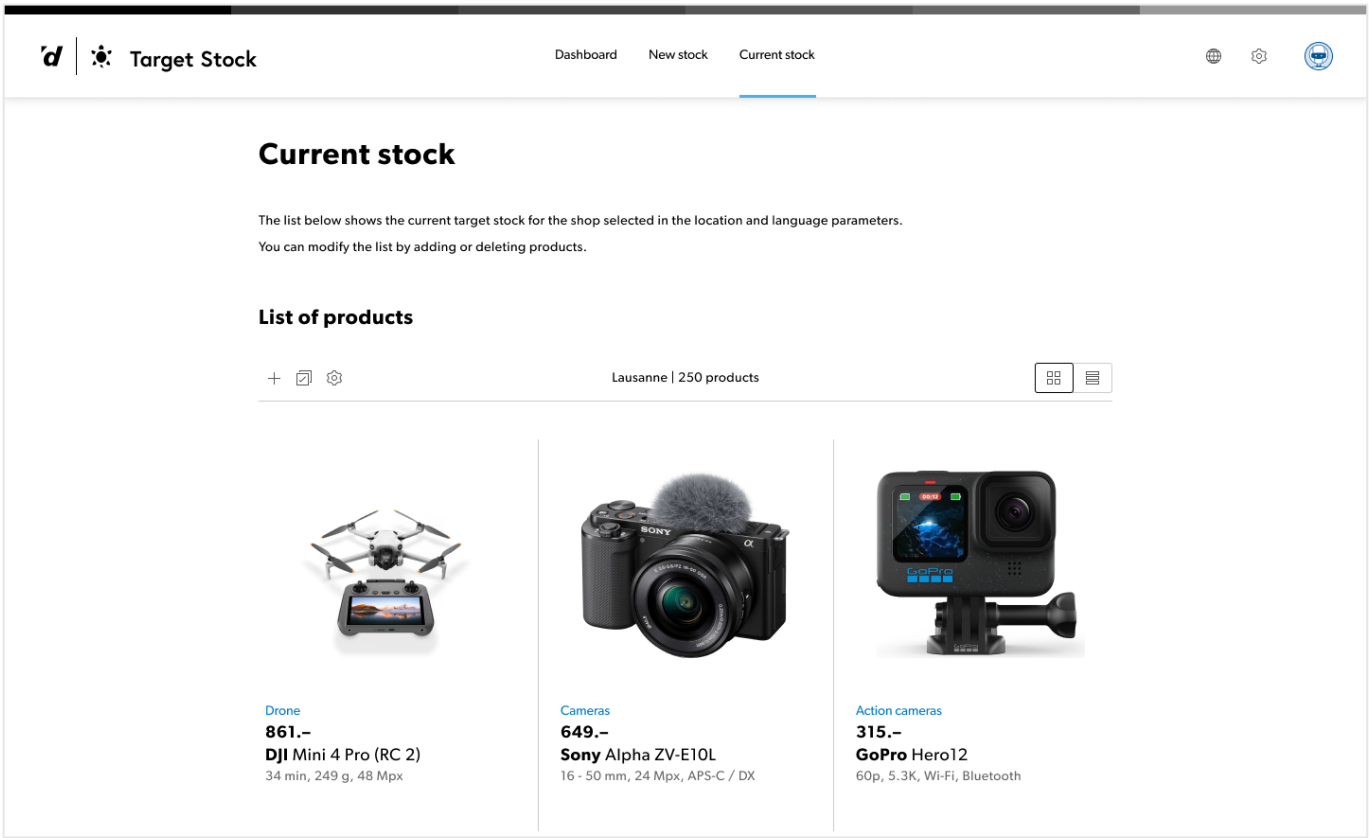


Figure 52 *- Page listant les produits du stock cible actuel*

Terminons par les quelques fonctions présentes dans l'entête. En plus du logo et de la navigation, il y a l'option permettant de changer la succursale et la langue sous la forme d'une icône de globe. Ce dernier ouvre un petit menu avec deux listes déroulantes permettant de choisir les deux réglages mentionnés précédemment. Cela reprend le design de la sélection du pays et de la langue sur Galaxus.

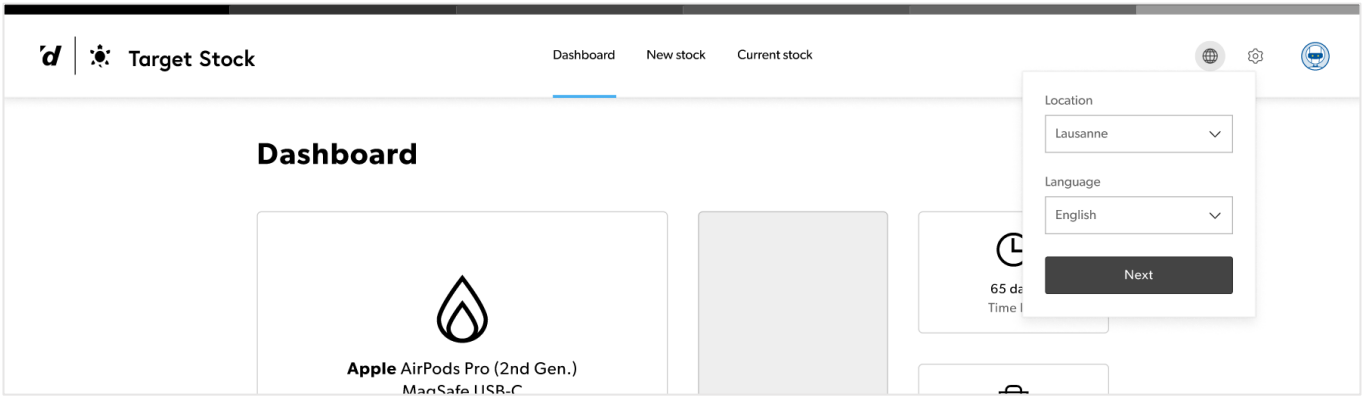


Figure 53 *- Réglage de la succursale et de la langue de la solution*

Nous avons ensuite les paramètres généraux avec l'icône d'engrenage, qui ouvre une nouvelle page lorsque l'on clique dessus. Ces réglages permettent de définir comment le mode automatique de la sélection de produits fonctionne. Ces paramètres seront ensuite enregistrés et conservés pour les prochaines utilisations.

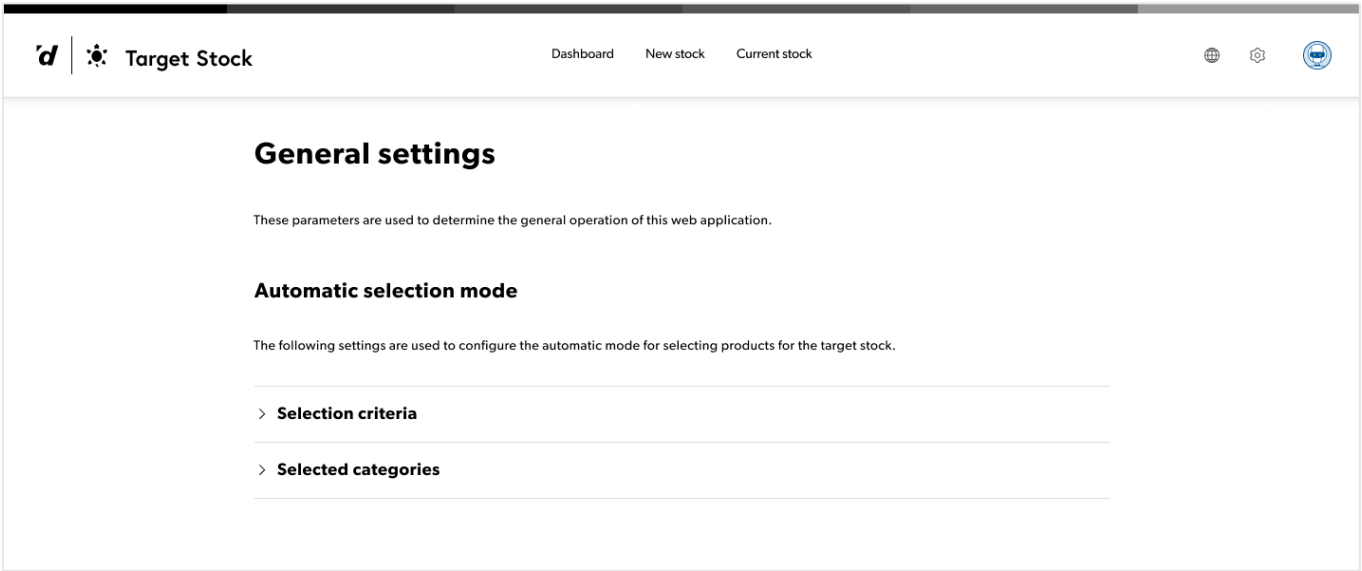


Figure 54 *- Page des paramètres généraux de l'application*

La première façon de paramétrer le mode automatique est via des critères de sélection. On va déterminer ici quels seront les produits qui vont figurer dans notre liste. Ils peuvent être issus des meilleures ventes, des nouveautés, ou bien être relatifs à la saison, comme des ventilateurs en été. Il est possible de combiner plusieurs critères pour obtenir une plus grande variété de produits.

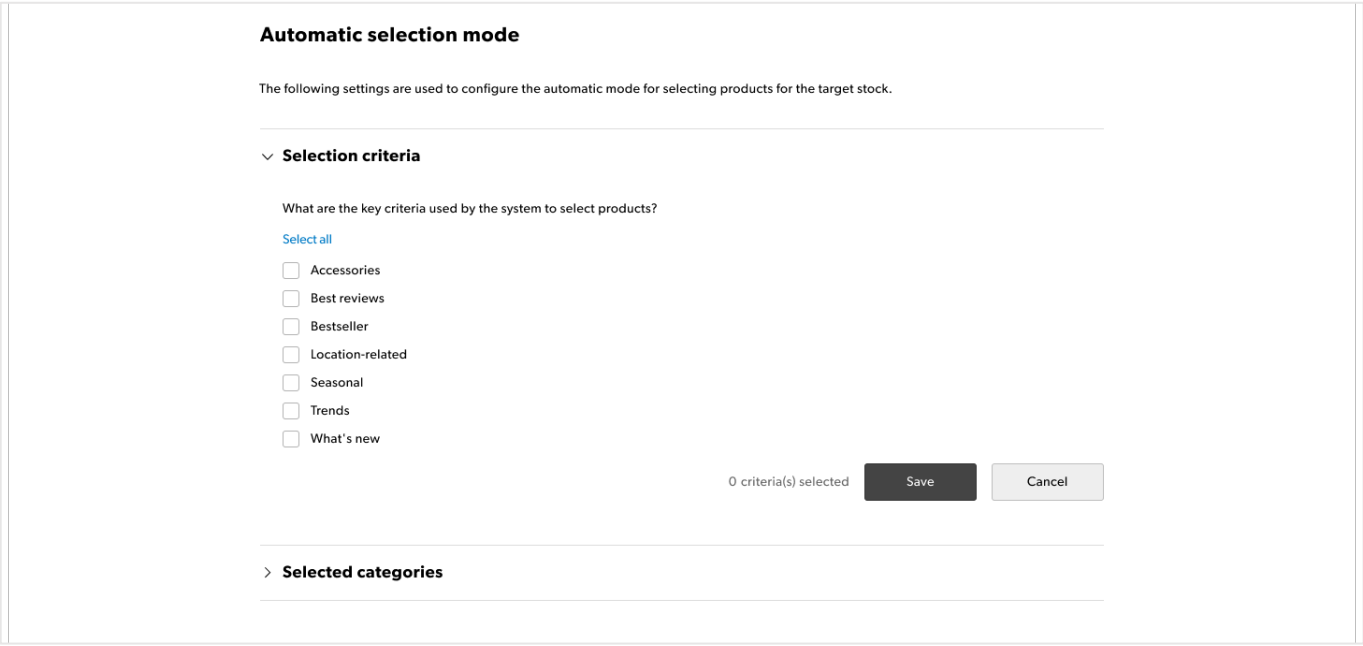


Figure 55 *- Liste des critères de sélection disponible pour le mode automatique*

Il est également possible de filtrer les résultats du mode automatique en indiquant quelles catégories principales nous souhaitons voir. En se concentrant sur les catégories principales, cela évite d'avoir une liste interminable en incluant les milliers de sous-catégories finales.

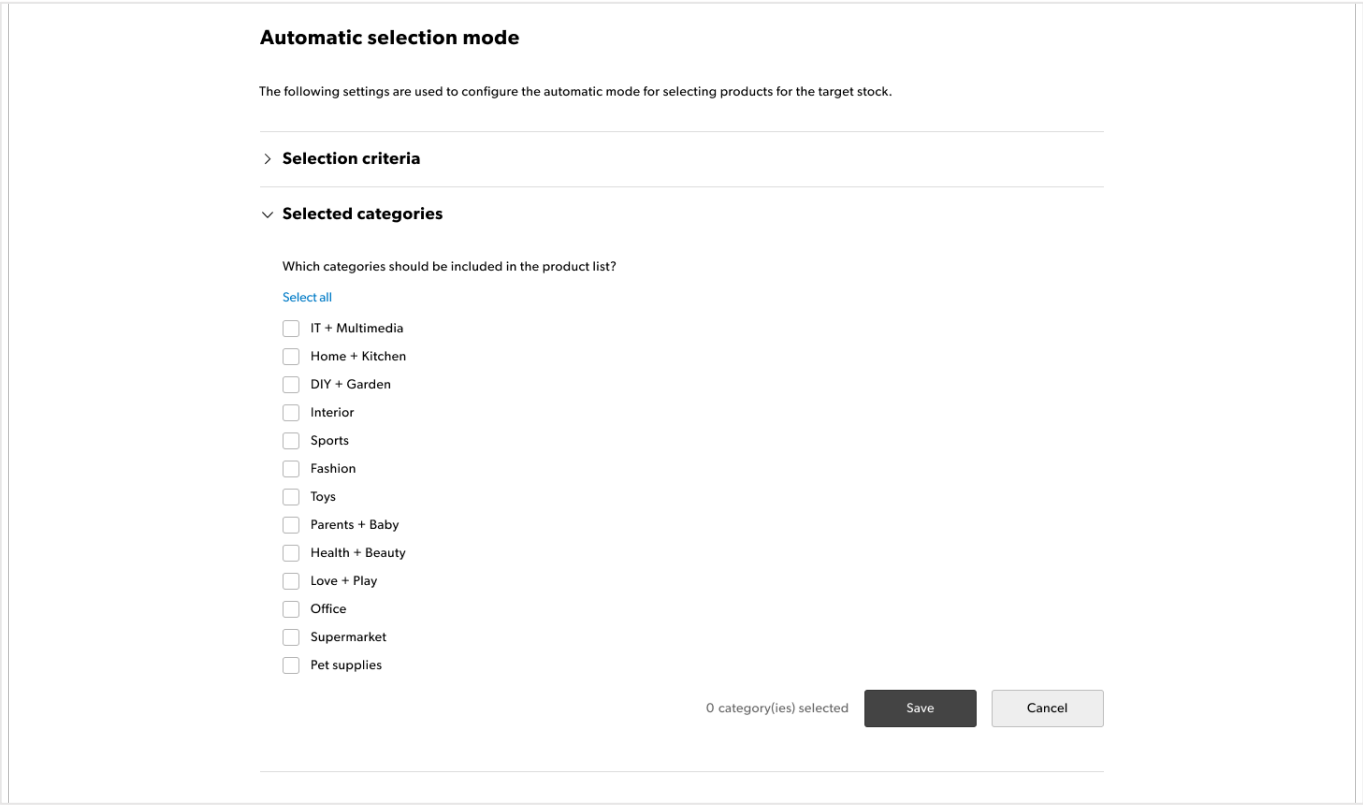


Figure 56 *- Liste des catégories de sélection disponible pour le mode automatique*

Toujours dans l'entête, il y a l'icône représentant l'utilisateur qui permet de révéler les différentes options relatives à ce dernier, comme la gestion des accès à l'application et la possibilité de se déconnecter.

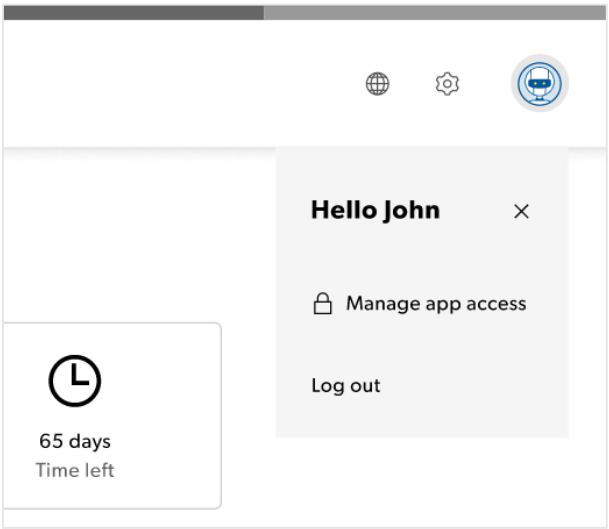


Figure 57 *- Menu d'utilisateur*

Bien que l'accès à la solution soit déjà restreint via le système de connexion utilisant le compte Microsoft de l'utilisateur, il est possible de définir précisément qui aura accès à l'application. Un utilisateur de base n'aura pas accès à cette vue, seul un "superuser" peut y accéder. Ce super utilisateur peut attribuer les mêmes accès que lui à d'autres utilisateurs si nécessaire. Cela peut être utile, par exemple, s'il est en vacances et a besoin d'un remplaçant. En plus d'ajouter de nouveaux utilisateurs, le super utilisateur peut également modifier les informations d'un utilisateur et même le supprimer.

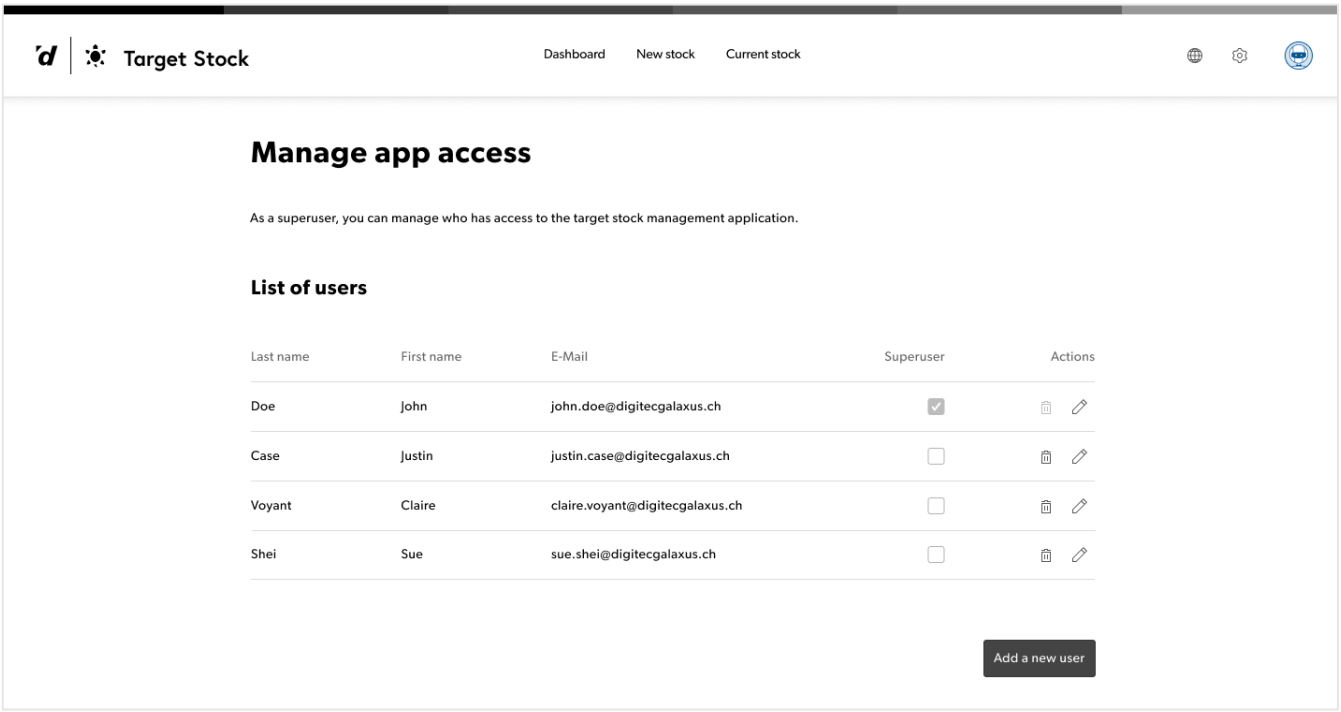


Figure 59 *- Page de gestion de l'accès à l'application*

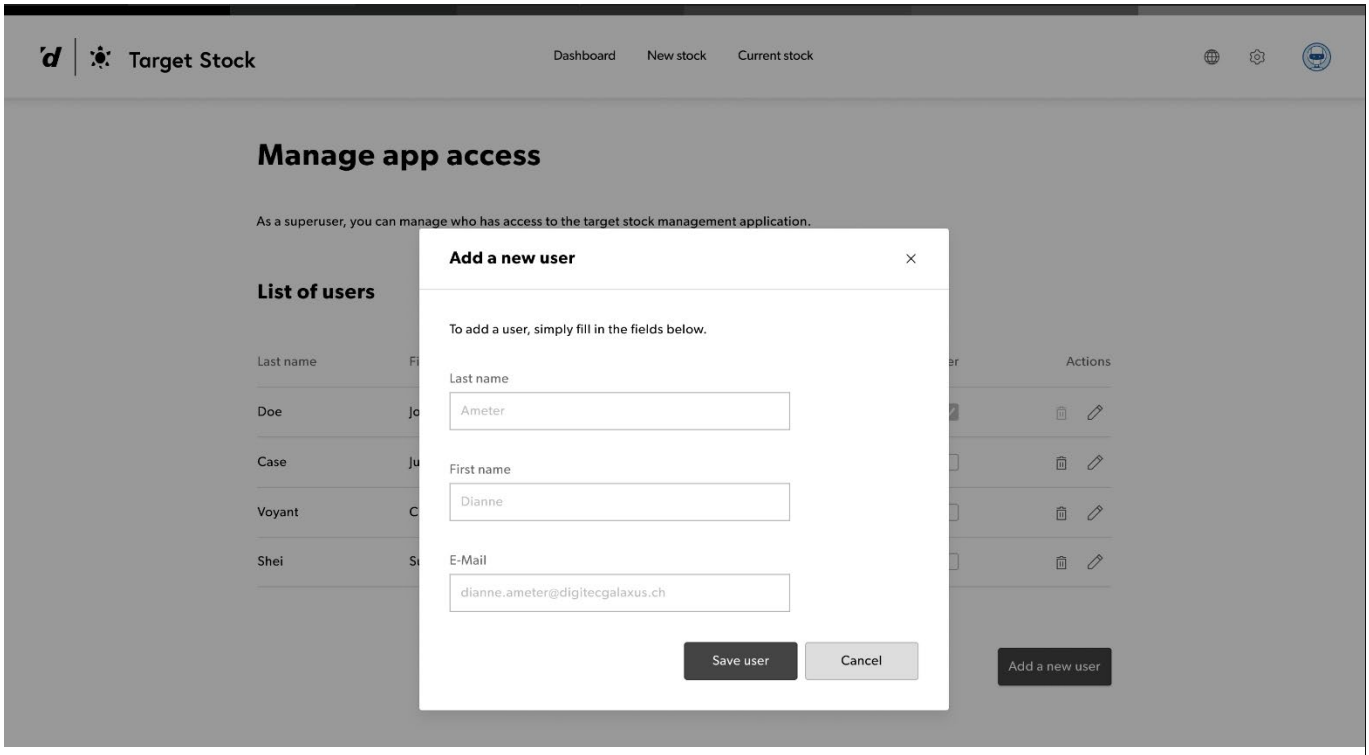


Figure 58 *- Fenêtre contextuelle permettant l'ajout d'un utilisateur*

En complément de tous les éléments présentés précédemment, il y a de nombreuses aides visuelles qui aident à comprendre comment fonctionne la solution. On retrouve ainsi de nombreux textes sur les pages et fenêtres contextuelles pour expliquer ce qui se passe à un moment spécifique. En plus des zones de texte, il y a de nombreuses infobulles servant à indiquer à l'utilisateur ce que permet de faire telle ou telle fonction. On les retrouve notamment au survol des boutons n'utilisant qu'une simple icône.

Bien que la majorité des éléments graphiques aient pu être repris du design system de l'entreprise, il a été nécessaire de créer de nouveaux éléments pour répondre aux besoins de cet outil et d'en adapter d'autres. Il y a notamment deux nouvelles icônes utiles dans les outils présents dans les vues avec une liste de produits. Elles ont été créées en combinant des inspirations trouvées sur internet et des icônes existants dans le design system, dont la liste se trouve sur le site dédié à ce sujet (Liste d'icônes du Design System).

Figure 60 *- Nouvelles icônes créées pour la solution*

Enfin, cette maquette ne se limite pas aux éléments graphiques, elle est également interactive. La quasi-totalité des fonctions visuelles est fonctionnelle jusqu'à un certain niveau. Il est possible de cliquer et de défiler pour tester les fonctionnalités. Cela est particulièrement utile lors des tests utilisateurs, permettant aux personnes d'interagir avec la maquette et d'avoir un premier aperçu de l'application. Cela se traduit par une grande quantité de liens entre les différents composants et pages, que ce soit au clic ou au survol d'un élément.

Sur Figma, le logiciel de design utilisé dans ce projet, plusieurs composants ont été créés ou repris du design system pour être incorporés dans les différentes pages de la solution. Ces composants ont généralement plusieurs variantes pour pouvoir changer d'état lors d'une action.

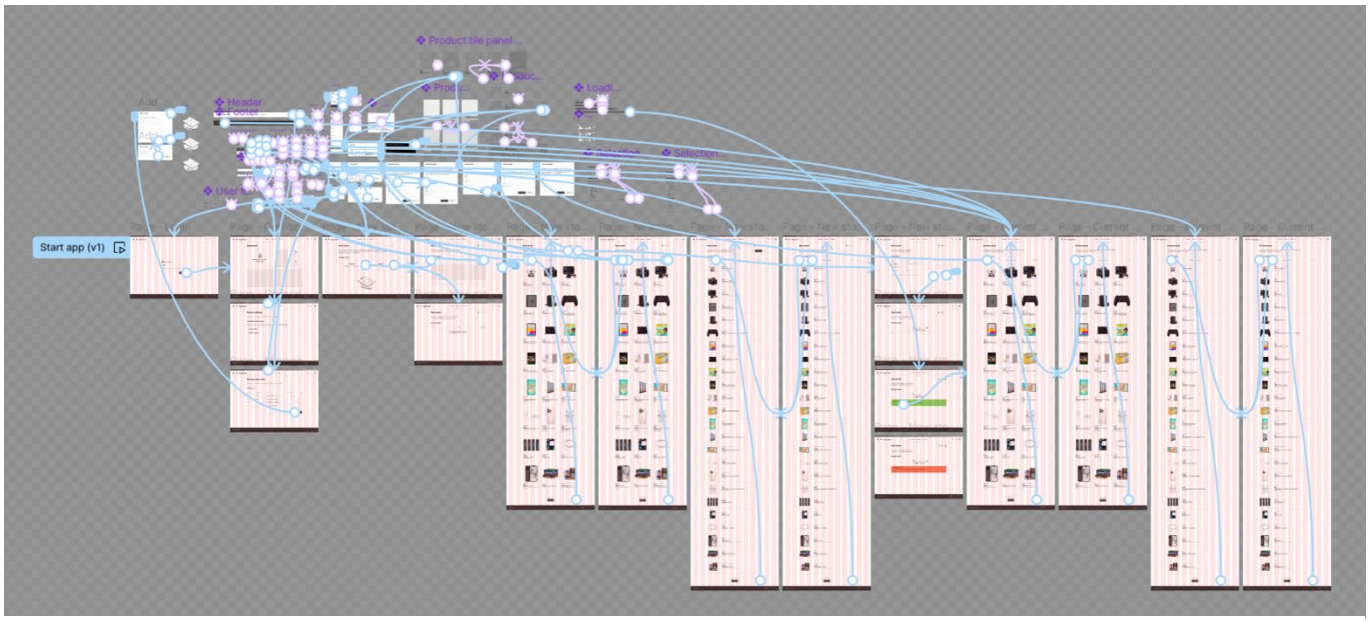


Figure 61 *- Vue d'ensemble des pages et composants sur Figma avec les relations*

Par exemple, pour le composant des produits, il y a la variante par défaut qui a des bordures et n'a ni fond ni outils apparents. La deuxième variante survient lors du survol de l'élément. Le fond devient gris clair, les bordures s'allongent et les outils apparaissent. La troisième variante est utilisée lors du mode de sélection multiple et est identique à la variante par défaut mais avec une case à cocher en plus. L'avant-dernière variante est liée à la précédente puisqu'elle intervient au survol de cette dernière, avec un fond gris clair et des bordures plus longues. Enfin, la dernière variante correspond à l'état "actif" de la tuile du produit lorsque l'on clique dessus pour le sélectionner. Le fond devient plus foncé et la case est cochée.

Ce principe de création est repris sur de nombreux composants à travers ce projet, permettant de gérer les différents états et interactions de manière cohérente et intuitive.

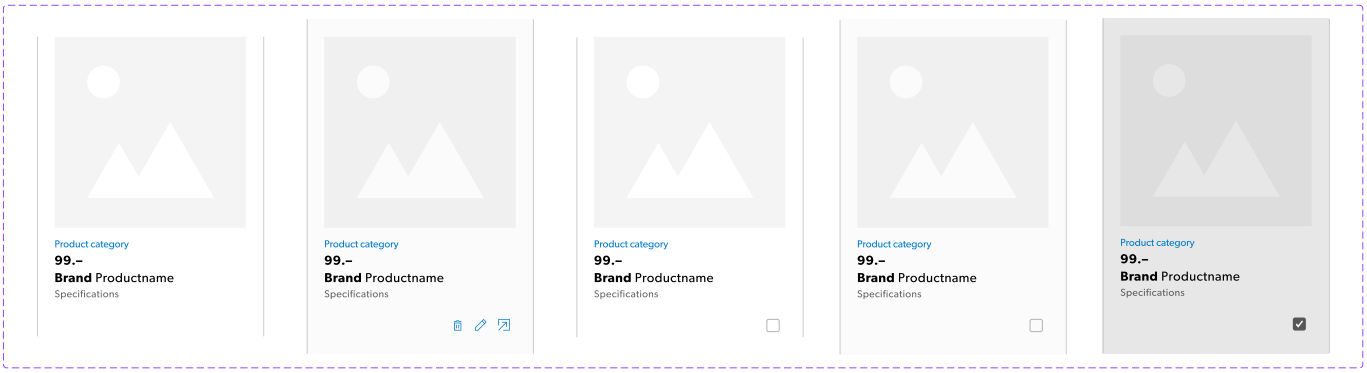


Figure 62 *- Composant de la tuile de produit sur Figma*

Il est possible de tester cette version intermédiaire de cette maquette interactive via le lien suivant : Prototype Figma (version intermédiaire). Il s'agit de la même version que celle utilisée lors des tests utilisateurs, qui seront abordés dans le prochain sujet de ce rapport.

2.6 Tests utilisateurs et adaptations

2.6.1 Objectifs et déroulement

Les tests utilisateurs avaient pour objectif d'évaluer l'intuitivité de l'interface utilisateur et la pertinence des fonctionnalités proposées par la solution. Chaque session, d'une durée d'environ 1h30, a permis aux participants de réaliser une série de tâches basées sur des scénarios précis, couvrant la quasi-totalité de l'application. Pendant ces sessions, les utilisateurs ont exprimé leurs impressions à haute voix, permettant ainsi de recueillir des retours détaillés sur leur expérience. Les tests se sont déroulés soit sur place, soit à distance via Teams, en fonction de la localisation des testeurs. À chaque étape, une combinaison d'observations et de questions a permis de comprendre les points forts et les points d'amélioration de la solution.

Il y a eu quatre participants au total lors de cette session de tests. Les deux premières séances se sont déroulées sur place à la succursale de Lausanne avec le gérant du magasin et son adjoint. Les deux séances suivantes ont été réalisées à distance via Teams en raison de la distance me séparant des deux derniers participants. La première séance à distance était avec le gérant de la succursale de Winterthur et la seconde avec le gérant de Dietikon. Ne pouvant réaliser les tests en français ou en allemand par manque de connaissances des deux parties dans la langue adverse, l'anglais a été privilégié en raison de sa bonne adoption par la majorité du personnel.

Le test s'est découpé en neuf parties, dont huit concernaient les différents scénarios et les tâches qui leur étaient rattachées. La dernière partie était consacrée à un feedback général sur la solution et l'expérience vécue.

N°	Scénario	Situation	Tâche
1	Connexion et accès au tableau de bord	Vous souhaitez définir le stock cible de la saison à venir et vous vous rendez donc sur l'outil dédié. Vous vous retrouvez premièrement sur la page de connexion.	Connectez-vous et accédez au tableau de bord.
2	Choix de la succursale et de la langue	Avant de commencer à gérer le stock cible, vous souhaitez définir quelques paramètres de base comme la langue et votre succursale notamment.	Trouvez l'option permettant de modifier la succursale et la langue.
3	Sélection des produits pour le stock cible	Vous souhaitez à présent générer une liste de produits à destination du stock cible et souhaitez que le système définisse automatiquement la liste.	Suivez le processus pour générer une liste de produits automatiquement.
4	Modification des règles pour le stock cible	Maintenant que vous avez votre liste de produits, vous souhaitez définir les règles qui vont paramétrer la disponibilité du stock cible.	Modifiez les règles (succursale, dates et quantité) pour tous les produits puis pour un produit.
5	Modification de la liste de produit	La liste de produits vous convient partiellement, vous souhaitez la modifier en ajoutant un produit et en supprimant d'autres.	Ajoutez un produit puis supprimez-en plusieurs.
6	Validation et enregistrement des produits	Vous êtes satisfait de la liste de produits et souhaitez finaliser le processus de création et d'enregistrement du stock cible.	Continuez le processus et validez la liste de produits pour l'enregistrement dans l'ERP.
7	Consultation et modification du stock actuel	Le processus de sélection et d'enregistrement est à présent terminé mais vous venez de penser à un produit qui serait intéressant pour le stock cible. Vous souhaitez donc rajouter ce produit au stock actuel.	Accédez à la liste de produits du stock actuel et ajoutez un nouveau produit.
8	Gestion des accès à l'application	En tant que super utilisateur, vous pouvez gérer les accès à l'application et vous souhaitez ajouter un collègue pour qu'il puisse également gérer le stock cible. (L'accès est premièrement limité par l'utilisation de la connexion avec Microsoft puis en vérifiant qu'il s'agisse bien d'une adresse e-mail de l'entreprise).	Accédez aux paramètres d'accès à l'application et ajoutez un utilisateur.

Tableau 1 *- Liste des scénarios et des tâches du test utilisateurs*

En raison du temps nécessaire pour créer la première version de l'interface et de divers problèmes survenus durant cette première phase, les sessions de test se sont étalées sur plusieurs semaines et ont chevauché la phase de développement. Bien que décalées, ces sessions n'ont pas empêché l'avancement du développement, car certains aspects de la solution pouvaient être traités simultanément et adaptés facilement si nécessaire.

2.6.2 Synthèse des résultats

Les résultats complets et détaillés de chaque session de test se trouvent en annexe du présent rapport[2]. Bien que les tests aient duré en moyenne 1h30, les résultats peuvent donner l'impression qu'ils ont duré moins longtemps. Cela est dû au fait que la retranscription des tests a déjà été synthétisée, et aussi parce que les discussions ont parfois dérivé vers d'autres sujets. Par exemple, lors du dernier test avec le gérant de Dietikon, étant le référent actuel pour la gestion du stock cible de manière globale, l'entretien a permis de mieux comprendre ses attentes vis-à-vis de la solution et d'éclairer ma compréhension du fonctionnement du stock cible en plus de tester l'interface.

Principales observations et recommandations

1. **Connexion et accès au tableau de bord**
   - **Points positifs** : Les utilisateurs ont trouvé le tableau de bord facilement accessible.
   - **Suggestions** : Ajouter des informations explicatives sur les statistiques, inclure des classements de ventes et des comparaisons avec des concurrents.
2. **Choix de la succursale et de la langue**
   - **Points positifs** : La sélection de la langue et de la succursale a été jugée intuitive.
   - **Suggestions** : Déplacer ces options dans les paramètres globaux ou les regrouper avec d'autres paramètres similaires.
3. **Sélection des produits pour le stock cible**
   - **Points positifs** : Le processus de génération automatique de la liste de produits a été compris et jugé conforme aux attentes.
   - **Suggestions** : Fournir plus d'informations sur les critères de sélection des produits, utiliser une pagination plutôt qu'un bouton "Afficher plus".
4. **Modification des règles pour le stock cible**
   - **Points positifs** : Les options de réglages globaux et individuels ont été trouvées logiques.
   - **Suggestions** : Ajouter des avertissements pour certaines actions, comme la sélection de toutes les succursales et réduire les options non nécessaires, comme la modification des dates individuelles.
5. **Modification de la liste de produits**
   - **Points positifs** : L'ajout et la suppression de produits ont été jugés intuitifs.
   - **Suggestions** : Permettre l'ajout de plusieurs produits à la fois, offrir des guides explicatifs pour les nouvelles fonctionnalités et améliorer la gestion des listes par succursale.
6. **Validation et enregistrement des produits**
   - **Points positifs** : Le processus de validation et d'enregistrement a été jugé clair.
   - **Suggestions** : Remplacer la quantité par produit par le total de produits à stocker, séparer les critères saisonniers en deux (été et hiver).

[2] Vers les annexes

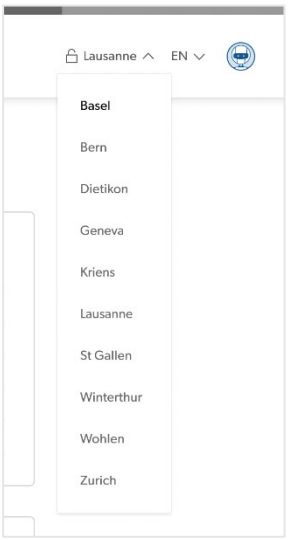
7. **Consultation et modification du stock actuel**
    - **Points positifs** : Accès rapide et facile au stock actuel.
    - **Suggestions** : Ajouter des fonctionnalités pour identifier les produits EOL (End Of Life) et proposer des produits similaires lors des suppressions.
8. **Gestion des accès à l'application**
    - **Points positifs** : La gestion des accès a été jugée pertinente et facilement trouvable.
    - **Suggestions** : Réserver l'accès par succursale avec un administrateur pour chaque magasin et ajouter un historique des modifications.
9. **Feedback général**
    - **Appréciations** : Design épuré, intuitivité, facilité d'utilisation, animations agréables.
    - **Difficultés** : Trouver certaines options, comprendre certaines icônes.
    - **Suggestions d'amélioration** : Déplacer certaines options pour plus de clarté, fournir des informations supplémentaires sur les processus en cours.
    - **Évaluation de l'utilisabilité** : Note moyenne de 8.5 sur 10.

2.6.3 Adaptations à la suite des tests

Plusieurs points d'amélioration ont été avancés durant les tests utilisateurs. Une seconde et dernière itération de la maquette a donc été réalisée pour correspondre au mieux avec les attentes des futurs utilisateurs. Bien entendu, il y a potentiellement d'autres améliorations qui pourront être faites à l'avenir avec davantage de tests et de retours, mais cela demandera plus de temps et de ressources.

Dans les premières modifications apportées à la maquette, la sélection de la localisation et de la langue a évolué. Ces derniers ont été séparés pour plus de clarté et pour que les utilisateurs puissent les distinguer plus rapidement. Ainsi, le choix de la succursale se présente désormais sous forme de liste déroulante. Le choix de la succursale est une fonctionnalité réservée aux administrateurs qui gèrent l'entièreté de la solution. Cela est représenté visuellement par un cadenas ouvert si l'utilisateur est un administrateur et fermé s'il s'agit d'un utilisateur de base. Cette modification fait suite au dernier entretien utilisateur avec la personne en charge de cette thématique dans l'entreprise, qui a indiqué que les personnes dans les succursales ne peuvent gérer que leur propre stock cible. Une infobulle est également présente pour informer l'utilisateur de cette restriction lorsqu'il survole cette option. Cela implique également que l'information sur la succursale actuelle a été retirée du pied de page.

Le choix de la langue reprend le même design sous forme de liste déroulante, mais il n'est pas limité à un type d'utilisateur, tout le monde y a accès. Ces modifications visent à rendre l'interface plus intuitive et à aligner les fonctionnalités avec les besoins et les rôles spécifiques des utilisateurs.



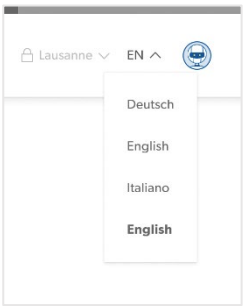


Figure 64 *- Liste déroulante de sélection de la langue*

Figure 63*- Liste déroulante de sélection de la succursale*

Le menu de l'utilisateur a été enrichi d'une nouvelle fonction qui permet d'accéder à un historique des modifications, comme cela a été suggéré lors des tests.

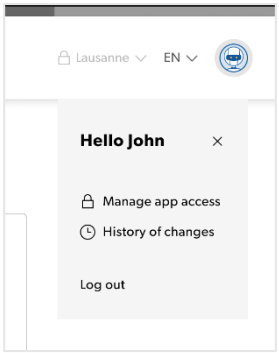


Figure 65 *- Menu d'utilisateur*

L'historique permet d'afficher les modifications effectuées aux différentes listes afin de garder un suivi sur ce qui a été fait. On peut notamment voir quand la modification a été faite, par qui, par rapport à quelle liste, pour quelle succursale et la nature de la modification. La liste peut s'étendre sur plusieurs pages au besoin.

Les administrateurs ont, en plus des informations citées précédemment, la possibilité de trier les modifications pour voir celles d'une succursale en particulier ou de toutes les succursales à la fois. L'utilisateur de base, lui, ne verra que les modifications relatives à sa succursale. En cliquant sur les éléments de cette liste, il est possible d'avoir plus de détails sur la modification. Malheureusement, cette partie n'a pas été créée dans la maquette par manque de temps et en raison de la priorité donnée à d'autres éléments.

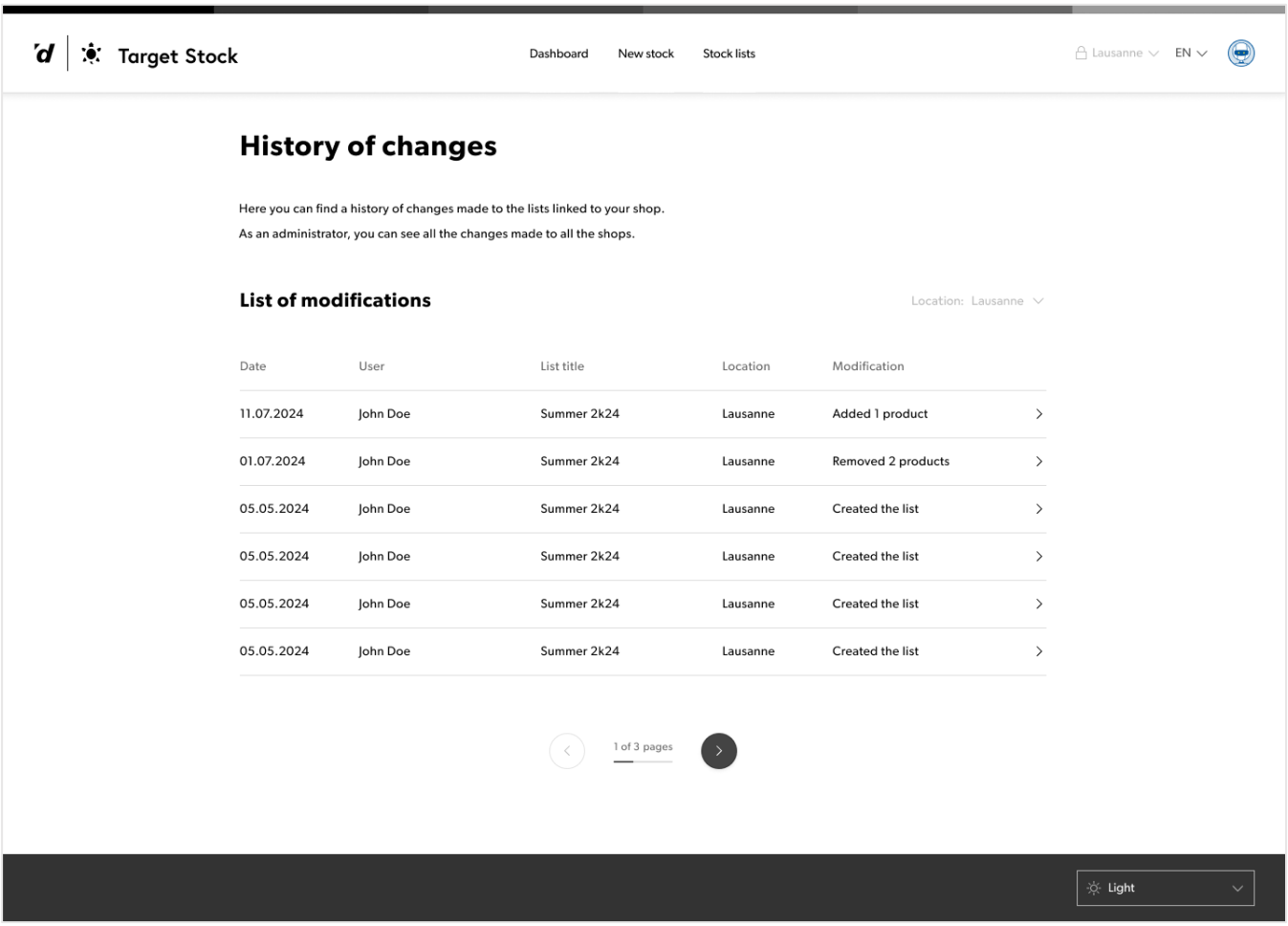


Figure 66 *- Page d'historique des modifications du stock cible*

La gestion des accès a également évolué légèrement avec l'ajout d'une pagination afin d'éviter un défilement infini de la page et le déplacement du bouton d'ajout d'utilisateur en conséquence. On peut désormais voir quelle succursale gère quel utilisateur.

L'image ci-dessous montre la version administrateur de la page avec toutes les options de gestion. Un utilisateur normal verra uniquement la liste avec l'indication de qui est un administrateur.

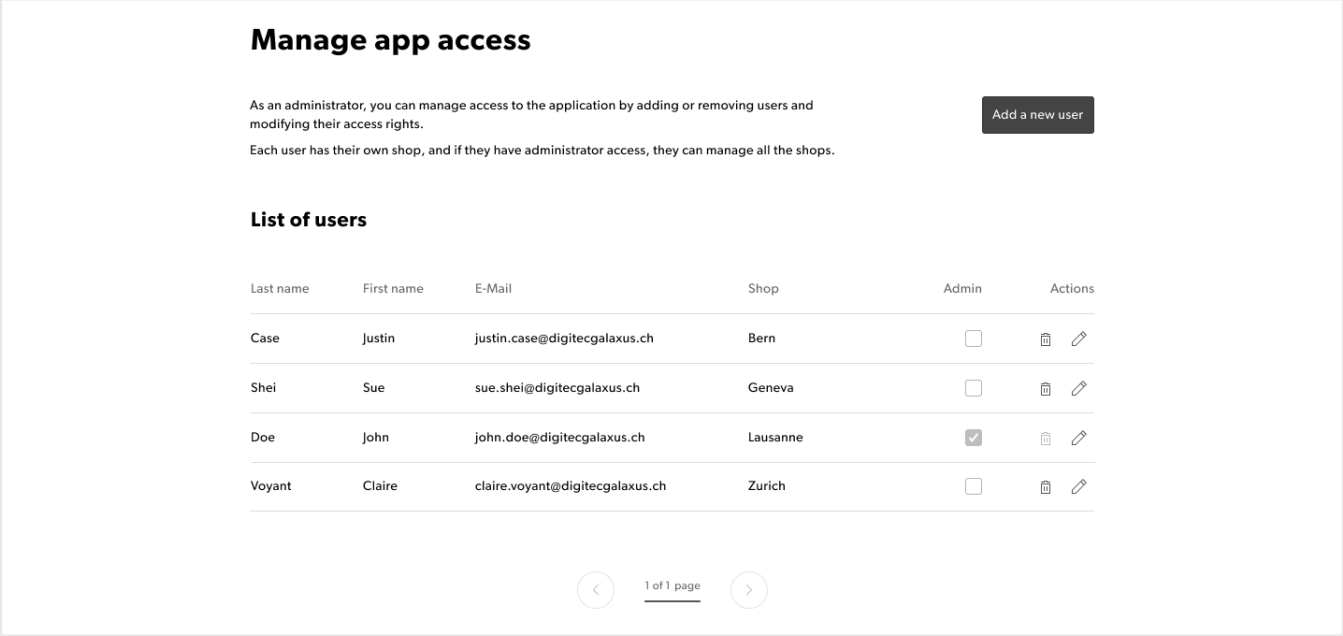


Figure 67 *- Page de gestion des accès de l'application (version pour administrateur)*

Un champ a été ajouté lors de la saisie d'un nouvel utilisateur afin de déterminer quelle succursale ce dernier sera en charge. Pour pouvoir gérer toutes les succursales, il suffit d'attribuer le rôle d'administrateur à l'utilisateur depuis la liste.

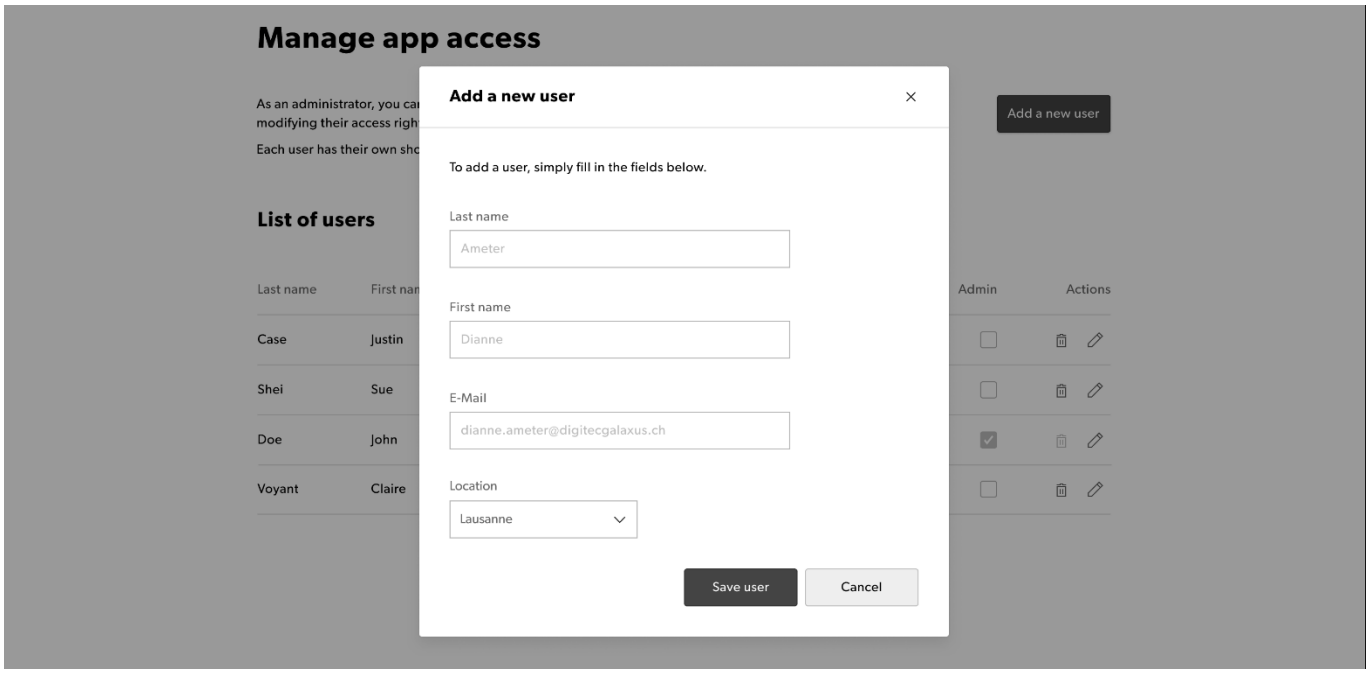


Figure 68 *- Fenêtre contextuelle permettant l'ajout d'un nouvel utilisateur*

Passons à présent au tableau de bord, qui a bien évolué depuis la version intermédiaire de la maquette. Les cases ont été réorganisées et leur contenu adapté pour davantage de clarté.

La première case permet de mieux comprendre ce qui est affiché sur ce tableau de bord, en précisant qu'il s'agit d'informations relatives à la succursale de l'utilisateur. En dessous de cette case informative, on retrouve les données sur le temps restant du stock, mais avec plus de détails cette fois-ci. Comme le stock cible n'est pas constitué d'une seule liste à la fois, mais comprend généralement une liste commune et d'autres plus petites listes pour les produits saisonniers par exemple, on peut faire défiler les listes ici pour voir combien de temps il reste pour chacune d'entre elles. Il s'agit uniquement des listes actives.

Ensuite, nous retrouvons la précédente statistique sur le niveau de stock, mais cette fois-ci séparée en deux parties. Nous pouvons ainsi voir, sur la totalité des listes, combien de produits différents nous avons et combien il nous en reste en stock. Le même principe est appliqué à côté avec le niveau du stock, qui répertorie la quantité de pièces dans l'espace de stockage.

Enfin, sur la partie droite, il y a les différents classements de produits en lieu et place du simple produit unique qui se vend le mieux. On y trouve d'abord les cinq produits qui se vendent le mieux, puis les cinq produits qui se vendent le moins. Cela permet d'ajuster au mieux le stock cible en augmentant la quantité des produits qui se vendent le plus ou en retirant les produits qui ne se vendent pas. Les informations contenues dans ce tableau de bord, notamment les classements, ne reflètent pas la réalité des ventes et sont surtout là à titre d'exemple.

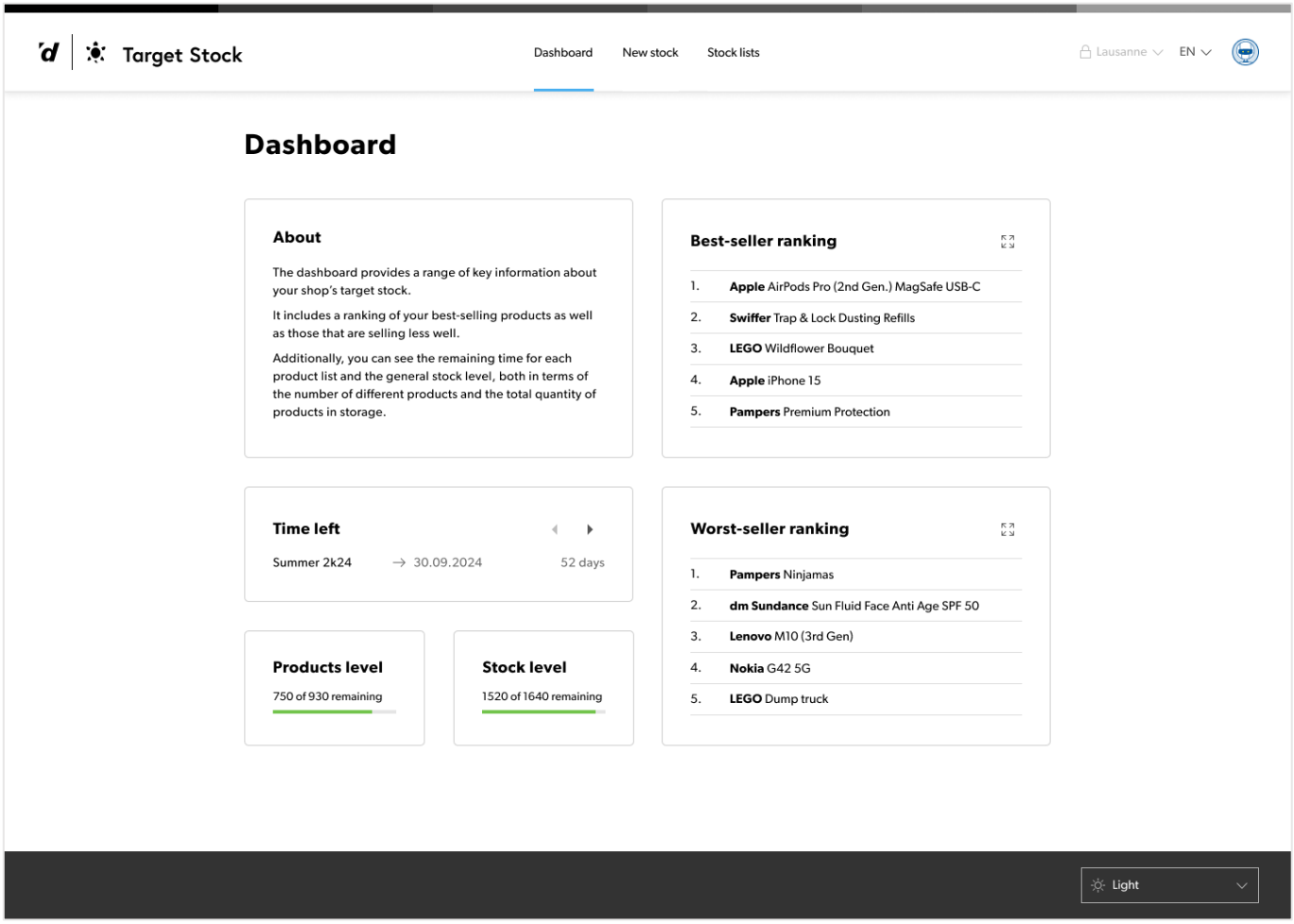


Figure 69 *- Page du tableau de bord*

Les classements peuvent être agrandis avec le bouton situé dans le coin supérieur droit de leur case afin d'afficher davantage d'informations. Le classement descend alors jusqu'à la 15e place, et les spécifications des produits sont également affichées. En complément, un bouton permettant d'ouvrir la page du produit a été ajouté pour chacun d'entre eux, fonctionnant de la même manière que celui positionné dans les cases des produits lorsqu'on affiche une liste de produits dans la création d'un nouveau stock et dans le stock actuel.

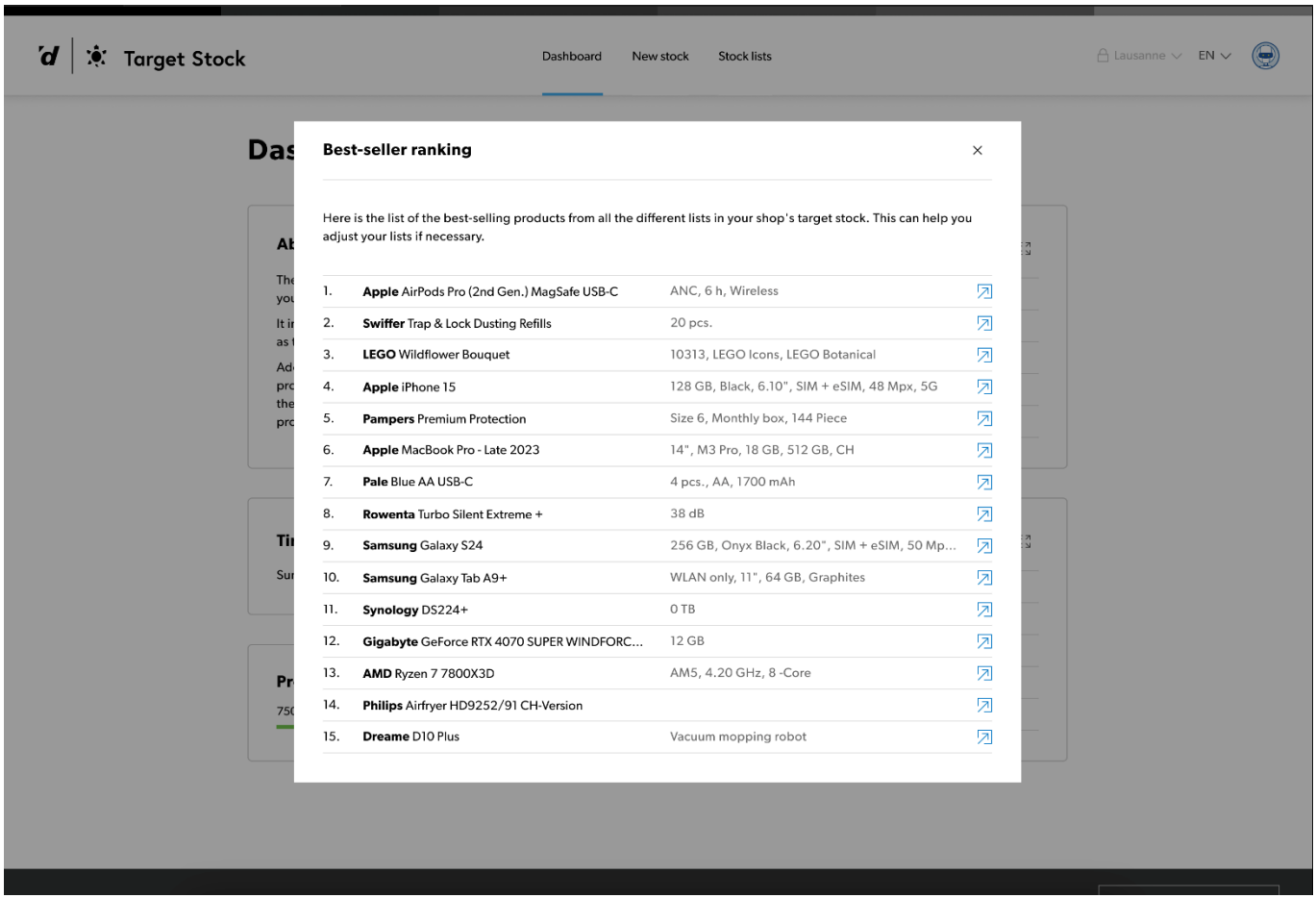


Figure 70 *- Fenêtre contextuelle contenant les détails des produits dans le classement des meilleures ventes*

La page de création du nouveau stock n'a pas trop évolué. Les améliorations ont principalement porté sur la clarté de l'information et les aides pour l'utilisateur. Par exemple, une description des symboles des étapes du processus de création du stock cible apparaît lorsque l'on survole le petit "i" à gauche du processus. L'utilisateur peut ainsi accéder à une légende des différents statuts et mieux comprendre où il en est dans le processus.

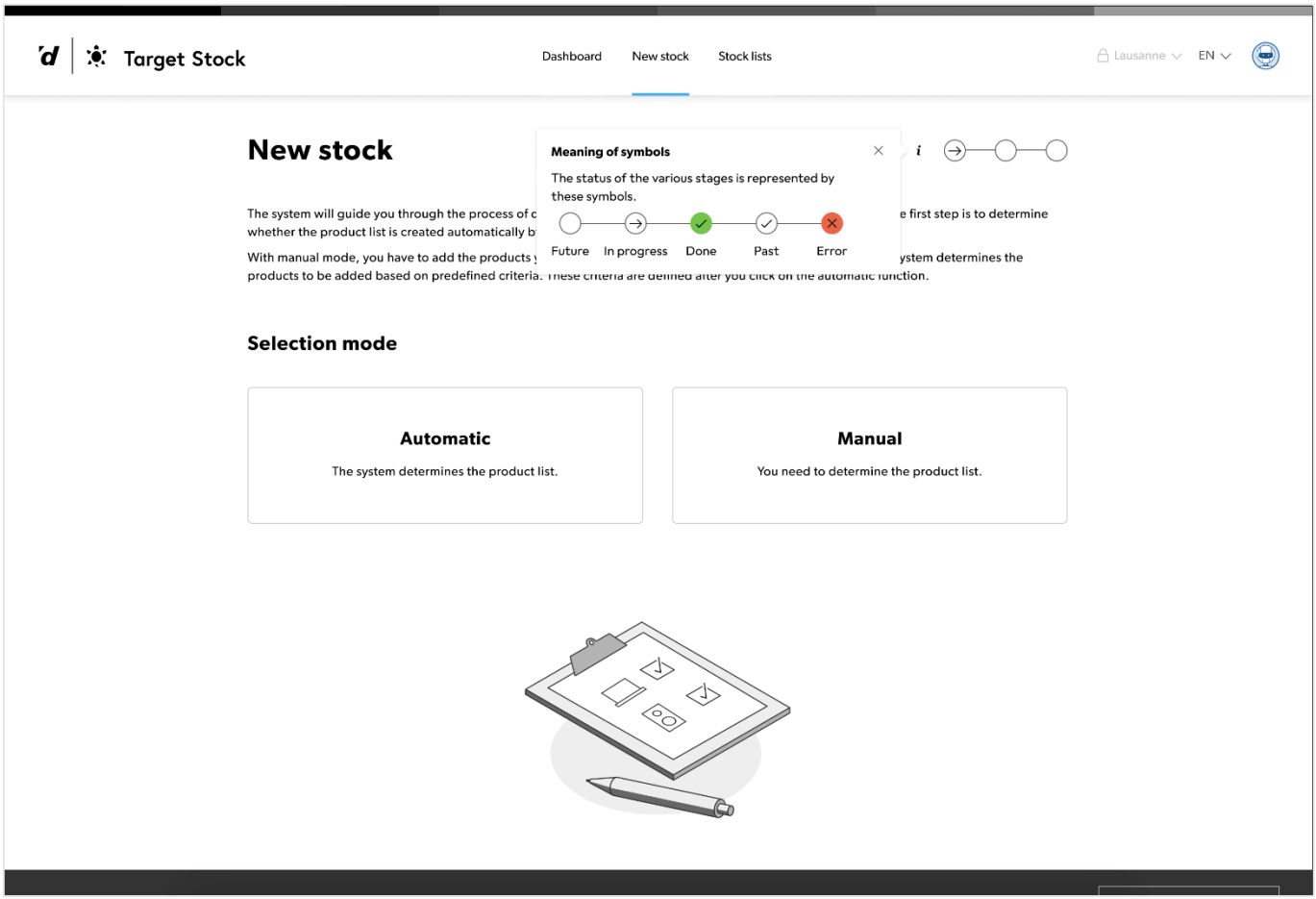


Figure 71 *- Message informatif sur la signification des différents statuts d'étape de processus*

Des infobulles sont présentes au survol des trois étapes du processus afin de décrire la nature de chaque étape. Cela aide l'utilisateur à comprendre les actions requises et les objectifs de chaque étape, rendant le processus plus transparent et facile à suivre.

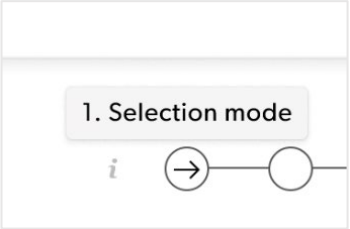


Figure 72 *- Infobulle indiquant la 1re étape du processus de création du stock cible*

Figure 73 *- Infobulle indiquant la 2e étape du processus de création du stock cible*

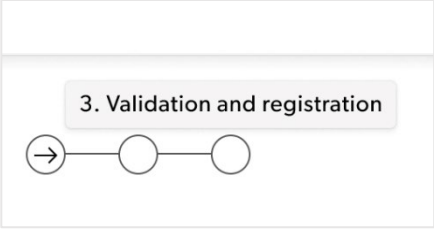


Figure 74 *- Infobulle indiquant la 3e étape du processus de création du stock cible*

Le gros changement à ce stade est l'ajout de réglages pour le mode automatique directement après avoir cliqué sur ce dernier. Auparavant, ces réglages étaient situés dans l'entête et n'étaient pas vraiment mis en avant. Lors des tests, il a été relevé que l'on ne comprenait pas forcément comment la liste de produits était générée. Désormais, une fenêtre contextuelle s'ouvre avant de générer les produits pour permettre de paramétrer la génération. On doit ainsi communiquer le nombre de produits différents souhaités et à partir de quels critères et catégories le système doit se baser.

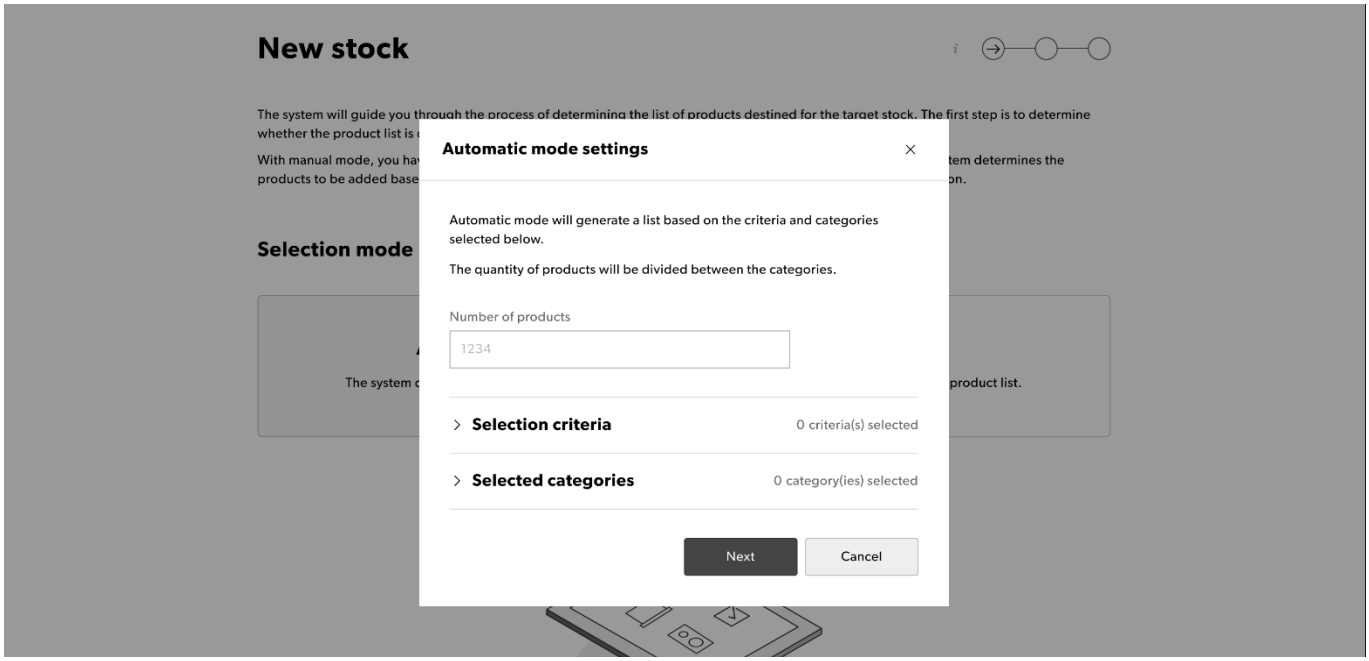


Figure 76 *- Fenêtre contextuelle de paramétrage du mode automatique pour la création du stock cible*

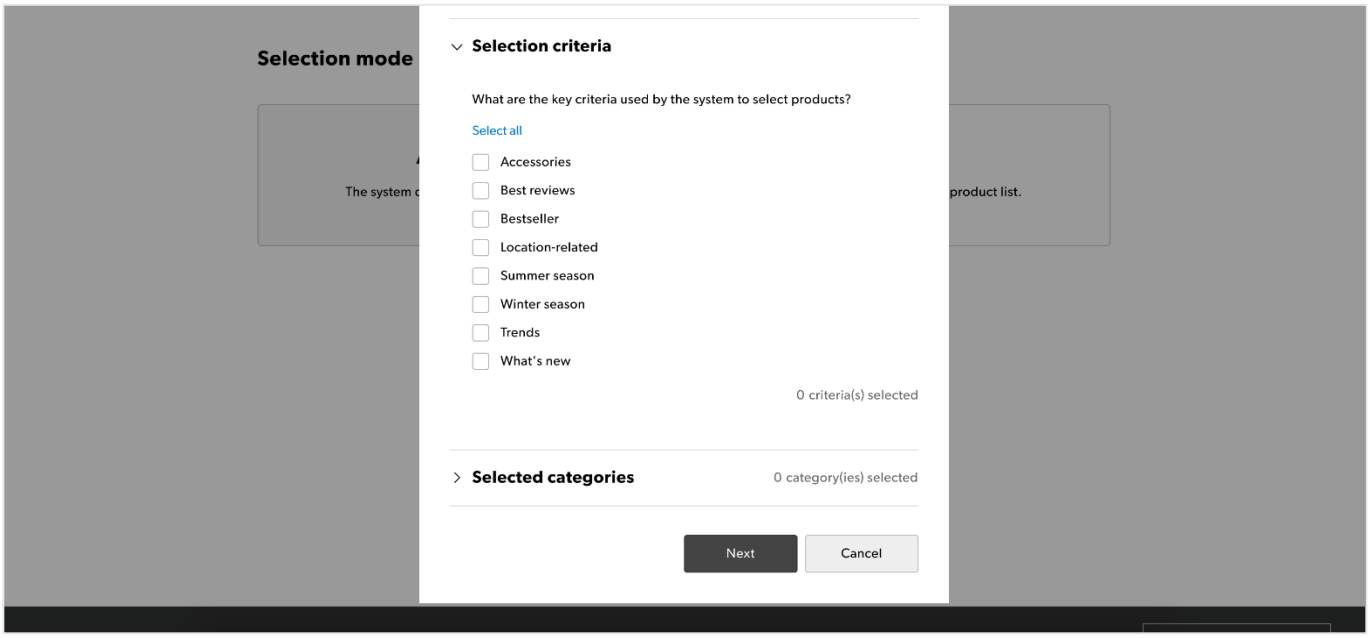


Figure 75 *- Critères de sélection pour les produits générés par le système*

Une fois ces paramètres définis et après un temps de chargement, nous arrivons sur la page contenant les produits générés par le système. Cette page a subi quelques modifications, comme la description qui a été retravaillée, ou encore le bouton “Suivant” qui est désactivé par défaut tant que les règles du stock cible n’ont pas été définies.

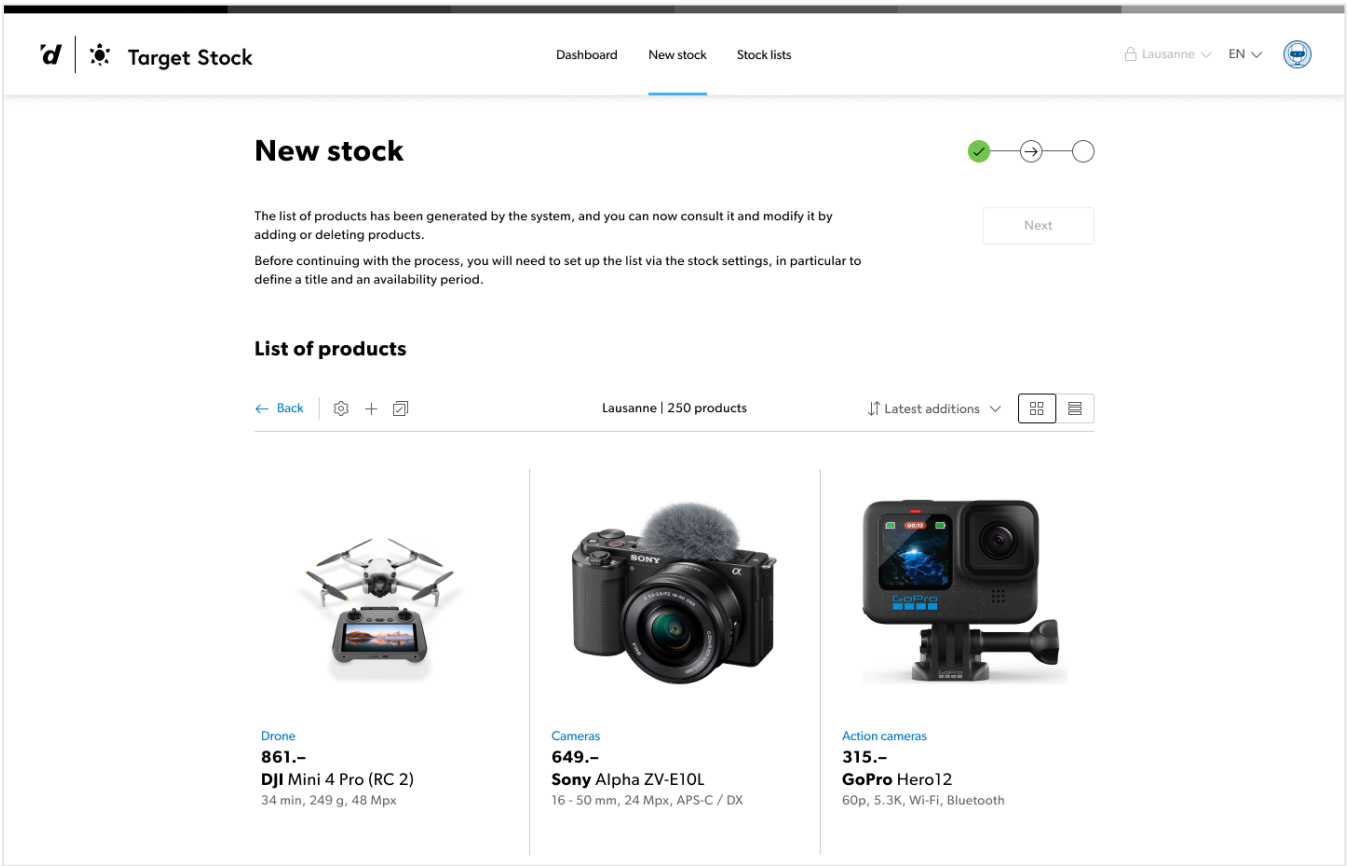


Figure 77 *- Page présentant les produits générés par le mode automatique*

Une infobulle apparaissant au survol du bouton “Next” informe l’utilisateur qu’il doit encore réaliser une tâche avant de pouvoir continuer. Cela aide à clarifier les étapes nécessaires et à éviter toute confusion quant à la progression du processus.

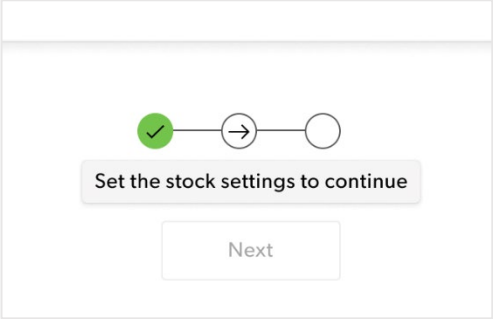


Figure 78 *- Infobulle indiquant d’établir les règles du stock cible avant de continuer le processus*

La fin de la liste a évolué pour ajouter une pagination en lieu et place d'un bouton "Afficher plus", comme conseillé lors des tests. Cela permet d'éviter un défilement infini de la page et rend l'expérience plus agréable.

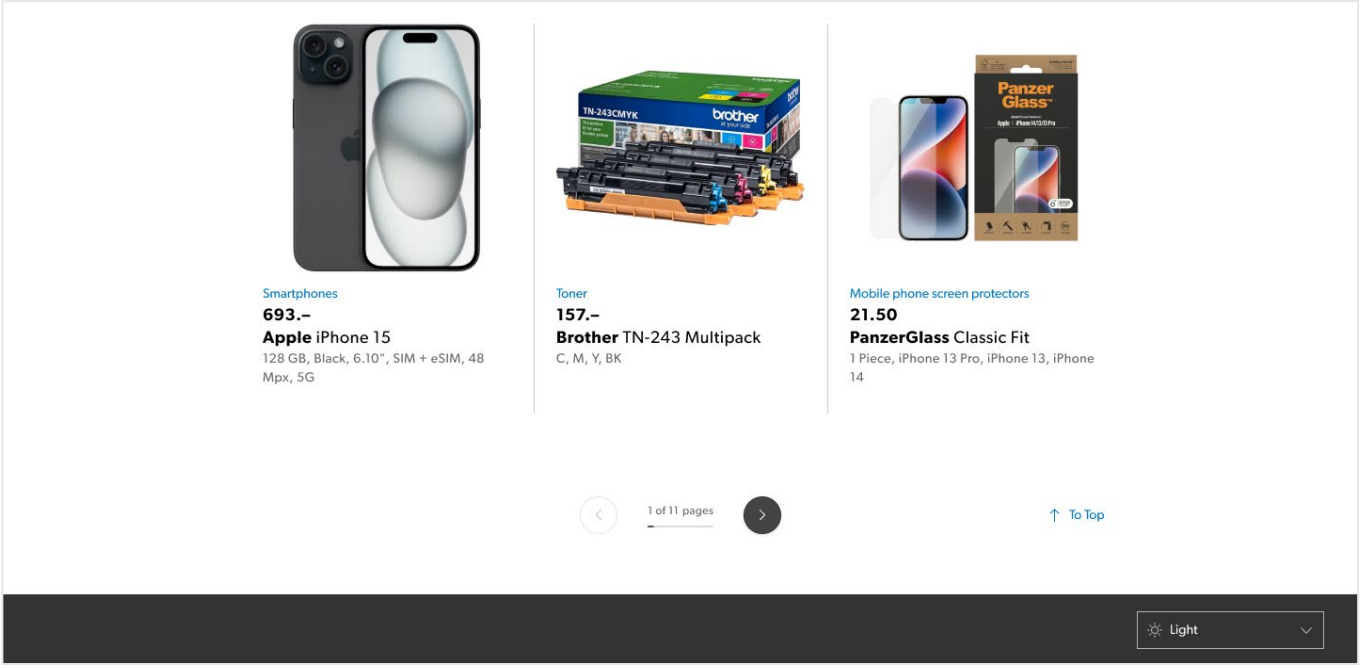


Figure 79 *- Fin de la liste de produits avec le système de pagination*

Un élément a été ajouté en ce qui concerne la mise en page de la liste : le système de tri. Plusieurs utilisateurs avaient demandé lors des tests comment se comportait la liste lorsque l'on ajoutait un produit et dans quel ordre les produits étaient affichés. Pour répondre à cette préoccupation, une liste déroulante a été ajoutée pour définir l'ordre d'affichage des éléments.

Par défaut, cette liste est réglée sur "Derniers ajouts", ce qui fait que les produits ajoutés manuellement apparaîtront en haut de la liste. Bien que les produits générés par le système apparaissent tous en même temps, ils ont techniquement été enregistrés dans la base de données à quelques millisecondes d'intervalle, ce qui permet de les trier de cette manière.

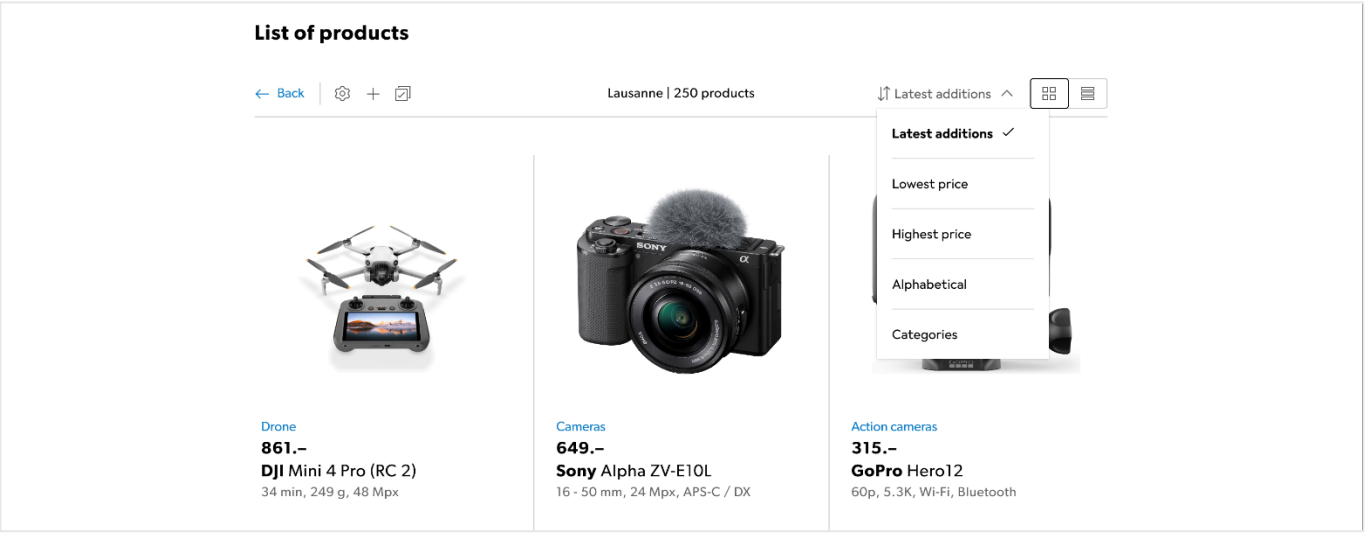


Figure 80 *- Liste déroulante permettant le tri des produits*

Les réglages du stock cible ont évolué pour mieux refléter la réalité de la situation. Notamment, il n'y a pas qu'un seul grand stock cible mais plusieurs listes qui ont des périodes de disponibilité et des objectifs différents. Cela permet ainsi de définir une liste contenant le stock cible général, disponible toute l'année, et un stock saisonnier qui ne sera disponible que pendant une période plus courte.

On doit ainsi donner un titre à chaque liste pour bien la différencier des autres. La succursale dans laquelle elle sera disponible est réglée par défaut sur celle attribuée à l'utilisateur, et seul un administrateur peut sélectionner une succursale différente ou toutes les succursales à la fois. Il ne restera alors plus qu'à définir une période de disponibilité et une quantité par produit avant de pouvoir passer à la suite.

Le texte présent sur cette fenêtre a également été étoffé pour apporter plus d'explications et de clarté sur cette étape.

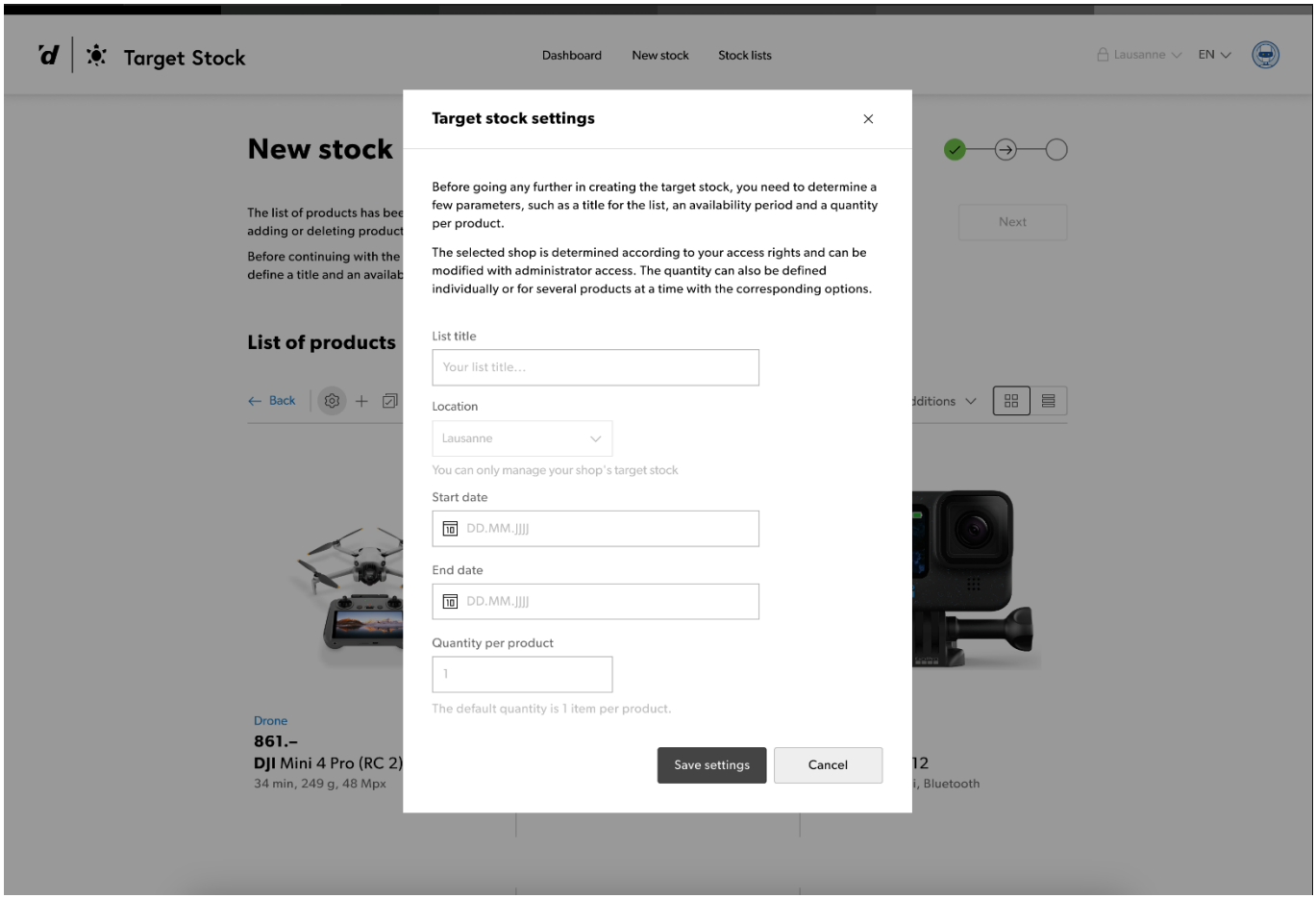


Figure 81 *- Fenêtre contextuelle permettant la définition des paramètres et règles du nouveau stock cible*

La manière d'ajouter des produits a également été améliorée en y ajoutant plus de fonctionnalités. Tout d'abord, une fenêtre s'ouvre lors du clic sur le bouton d'ajout, indiquant à l'utilisateur de saisir un lien de produit sur Galaxus pour ajouter ce dernier. Cette fenêtre précise également qu'il est possible d'ajouter plusieurs produits en une fois et de saisir une quantité directement ici si l'utilisateur souhaite avoir autre chose que la valeur par défaut.

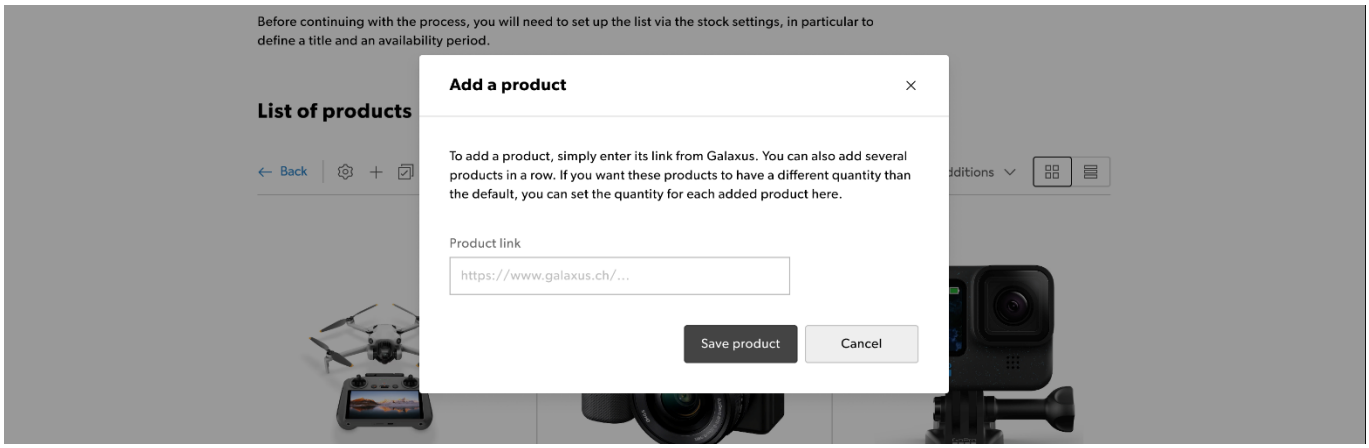


Figure 82 *- Fenêtre contextuelle permettant l'ajout d'un produit*

À partir du moment où un lien est ajouté, un deuxième champ apparaît, permettant à l'utilisateur d'ajouter un autre produit à la suite. Le champ de saisie de quantité apparaît également à ce moment-là. Cela permet à l'utilisateur d'ajouter plusieurs produits successivement tout en spécifiant les quantités désirées pour chacun.

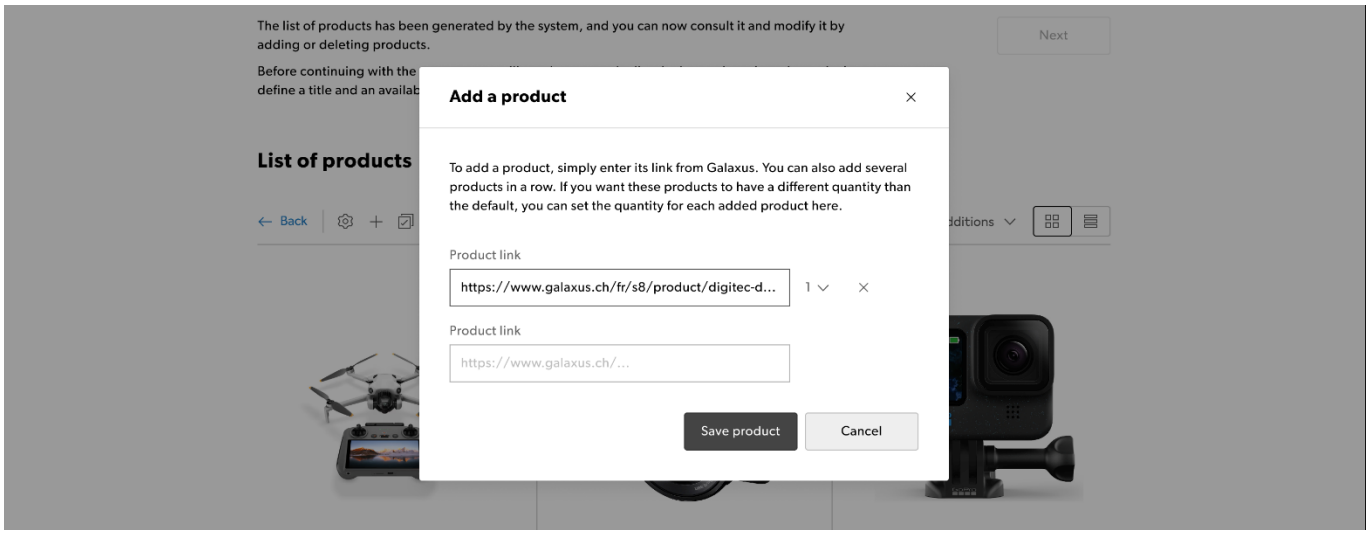


Figure 83 *- Illustration du fait de pouvoir ajouter plusieurs produits à la suite*

Il suffit de cliquer sur le chiffre situé à côté du champ de texte pour faire apparaître la liste déroulante et saisir une quantité. Cela permet à l'utilisateur de spécifier facilement la quantité désirée pour chaque produit ajouté.

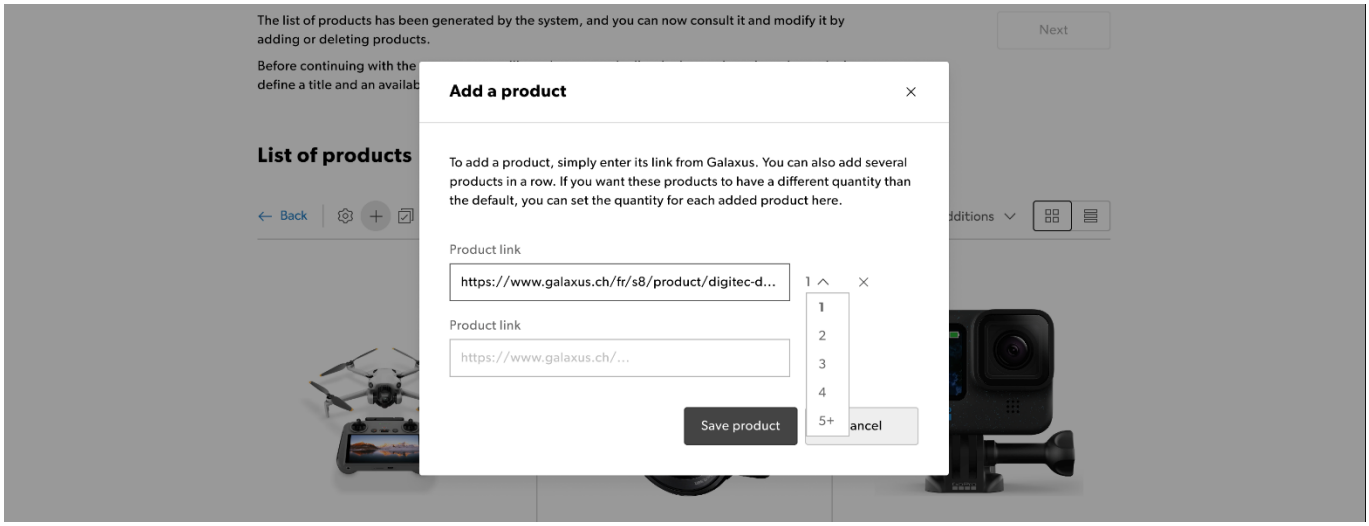


Figure 84 *- Sélection de la quantité lors de l'ajout d'un produit*

Le schéma peut se répéter plusieurs fois jusqu'à ce que l'utilisateur ait ajouté tous les produits souhaités. Il est également possible de supprimer des produits via l'icône de croix tout à droite de chaque élément. Cela offre une flexibilité maximale dans la gestion des ajouts de produits.

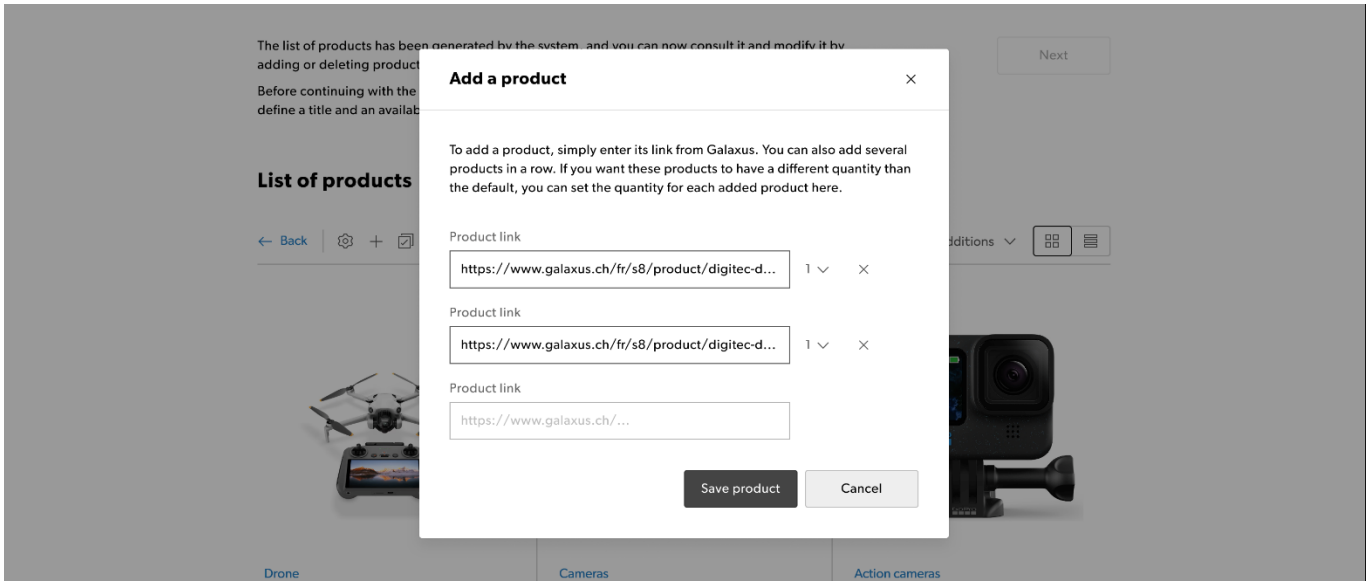


Figure 85 *- Illustration du fait de pouvoir ajouter plusieurs produits à la suite*

Un petit message apparaîtra en bas à gauche de l'écran pour confirmer l'ajout des produits à la liste. Comme pour les autres messages informatifs présents sur l'application, il est possible d'annuler l'action via le bouton "Undo", permettant ainsi de corriger rapidement une éventuelle erreur.

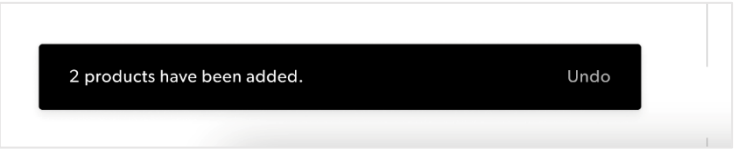


Figure 86 *- Message de confirmation d'ajout de produits*

La façon de modifier les règles du stock cible de manière individuelle a également évolué pour correspondre aux attentes des utilisateurs. À la suite de l'entretien avec la personne en charge du stock cible, il a été indiqué que les dates de disponibilité ne sont pas différentes entre les produits d'une même liste. En enlevant les dates, il ne reste que la quantité que l'on peut modifier par produit.

Afin d'éviter d'avoir une fenêtre qui s'ouvre uniquement pour cela, la définition de la quantité a été directement ajoutée sur le produit avec les autres outils lors du survol. Comme lors de l'ajout de produit, il s'agit d'une liste déroulante permettant de sélectionner une quantité prédéfinie. Si l'on souhaite mettre 5 et plus comme quantité, la liste déroulante se transforme en champ de formulaire pour saisir la quantité désirée. Cela reprend le fonctionnement de la quantité par produit dans le panier sur les deux plateformes de vente.

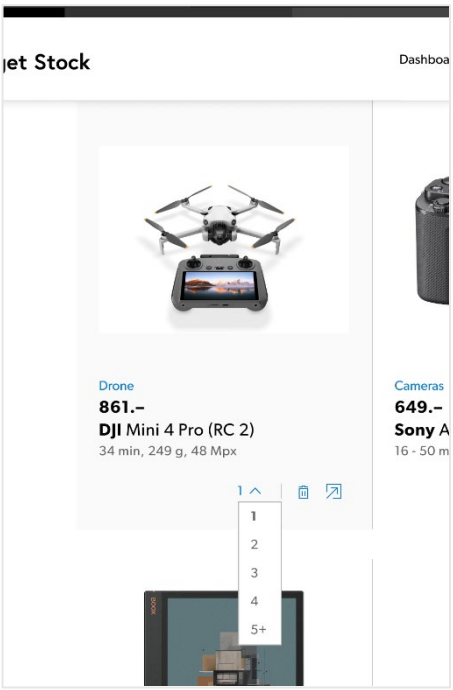


Figure 89 *- Liste déroulante de sélection de la quantité d'un produit*

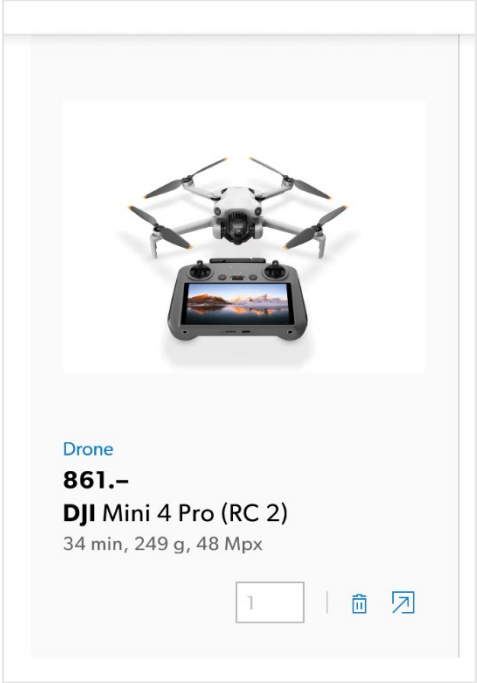


Figure 90 *- Champ de formulaire pour la saisie de la quantité d'un produit*

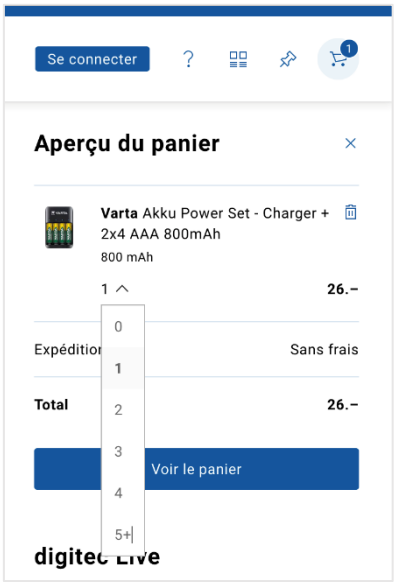


Figure 88 *- Liste déroulante de sélection de la quantité d'un produit dans le panier sur digitec.ch*

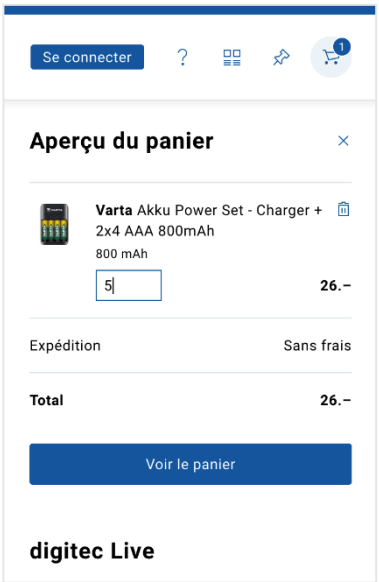


Figure 87 *- Champ de formulaire pour la saisie de la quantité d'un produit dans le panier sur digitec.ch*

Le mode de sélection multiple n'a pas beaucoup changé, à l'exception de la fonction permettant de modifier les règles de la sélection de produits. Comme mentionné précédemment, il n'est pas nécessaire de modifier les dates pour certains produits, car ces derniers ont tous la même période au sein de la liste. Ainsi, il est désormais uniquement possible de modifier la quantité pour la sélection de produits.

Un clic sur l'icône de crayon ouvre une fenêtre permettant de définir cela. Une alternative avec simplement une liste déroulante, à la manière de ce qui a été précédemment décrit et placé dans la barre d'outils, a été envisagée mais n'était pas vraiment optimale dans ce cas de figure.

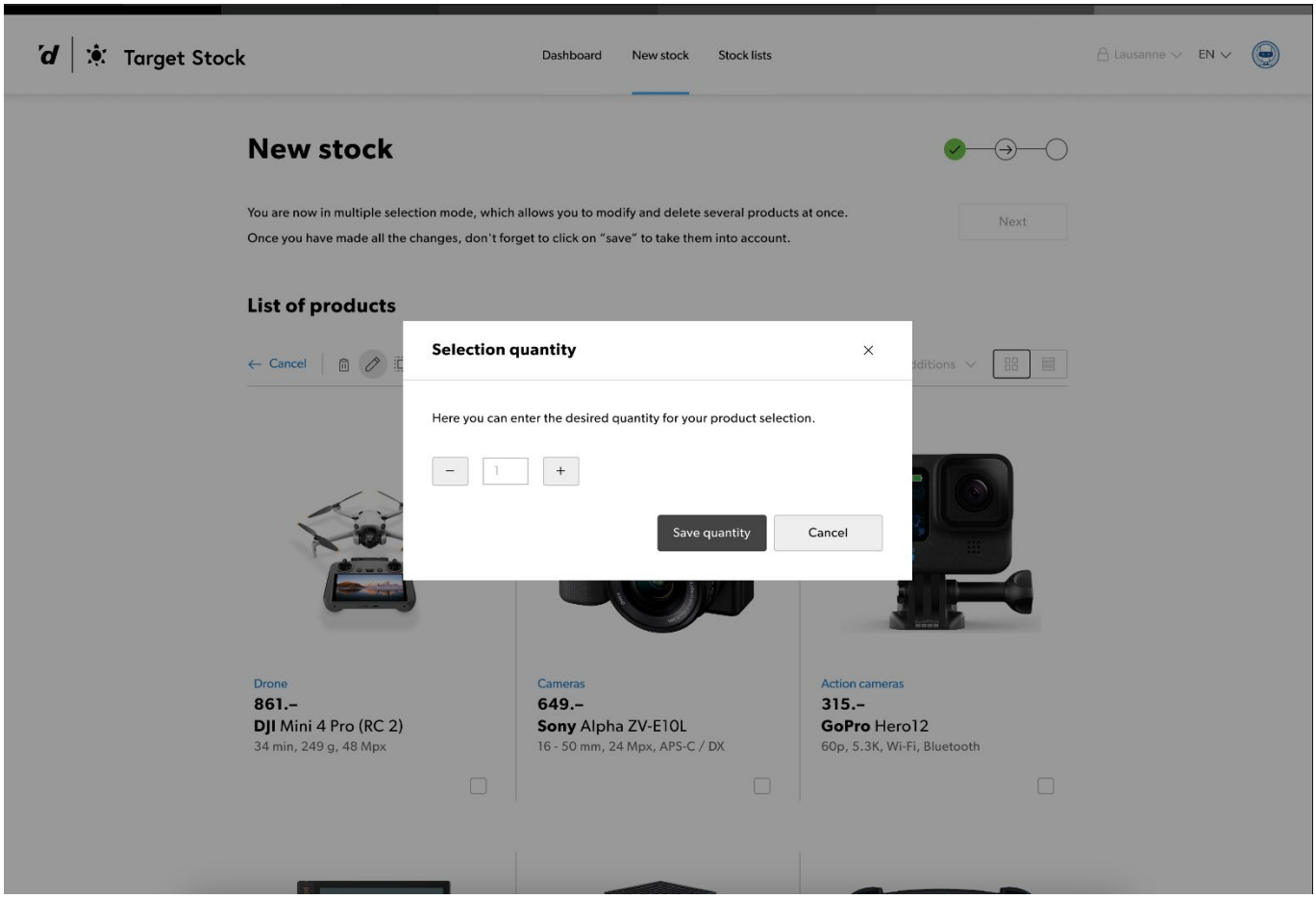


Figure 91 *- Fenêtre contextuelle permettant la saisie d'une quantité pour une sélection de produits*

Continuons le processus avec la validation des différents détails et l'enregistrement dans l'ERP. La vue avec le résumé a légèrement évolué en incluant maintenant le titre de la liste et la quantité totale de produits à stocker, en plus des éléments déjà présents précédemment.

On remarque également que le statut du processus a changé pour mieux respecter ce qui est décrit dans le design system de l'entreprise. Avant, il y avait uniquement des ronds verts avec une coche pour représenter une étape accomplie et maintenant, on distingue l'étape qui vient d'être terminée de celle du passé.

Enfin, la disposition des boutons de navigation a changé pour correspondre au reste de l'application. Le bouton pour revenir en arrière est désormais sous forme de lien, tandis que celui pour continuer est sous forme de bouton standard. Cela permet de mieux les distinguer et d'éviter les erreurs de précipitation expérimentées lors des tests utilisateurs.

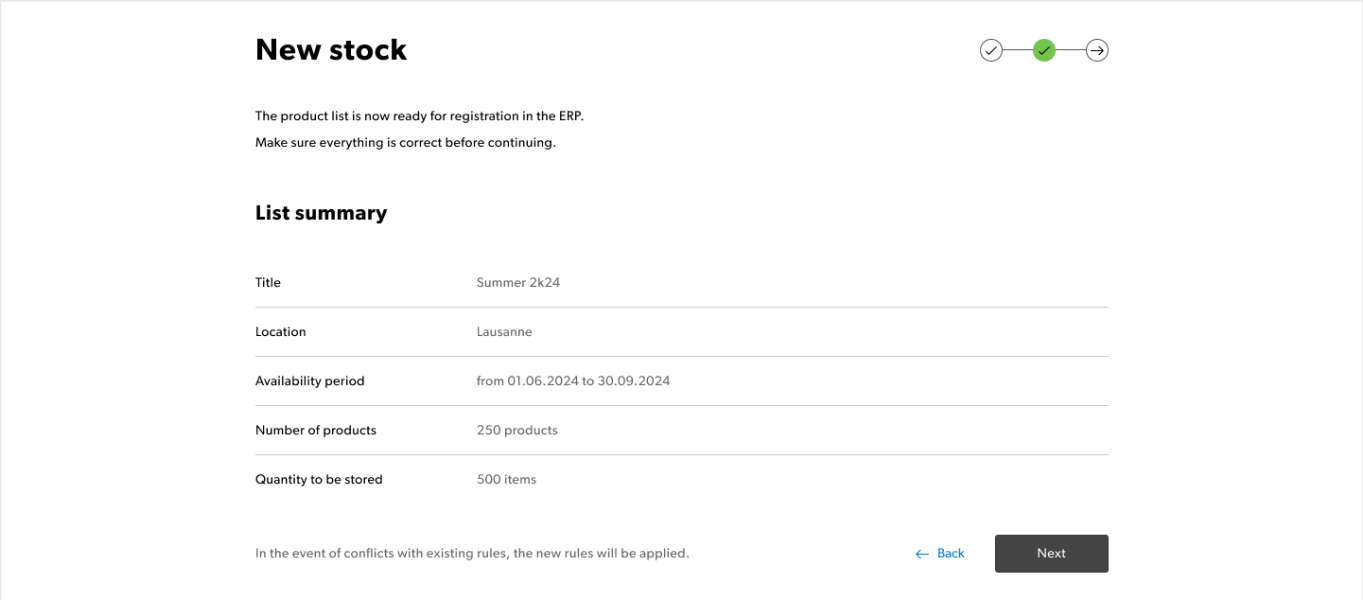


Figure 92 *- Page du résumé des informations avant l'enregistrement des produits dans l'ERP*

Dernière étape de création du stock cible, l'enregistrement des produits dans l'ERP. Un changement minime a été effectué ici : le pourcentage de progression a été déplacé en dessous de la barre de chargement pour plus de visibilité. Un statut en termes de nombre de produits traités a également été ajouté afin de donner plus de détails sur l'état d'avancement de la tâche.

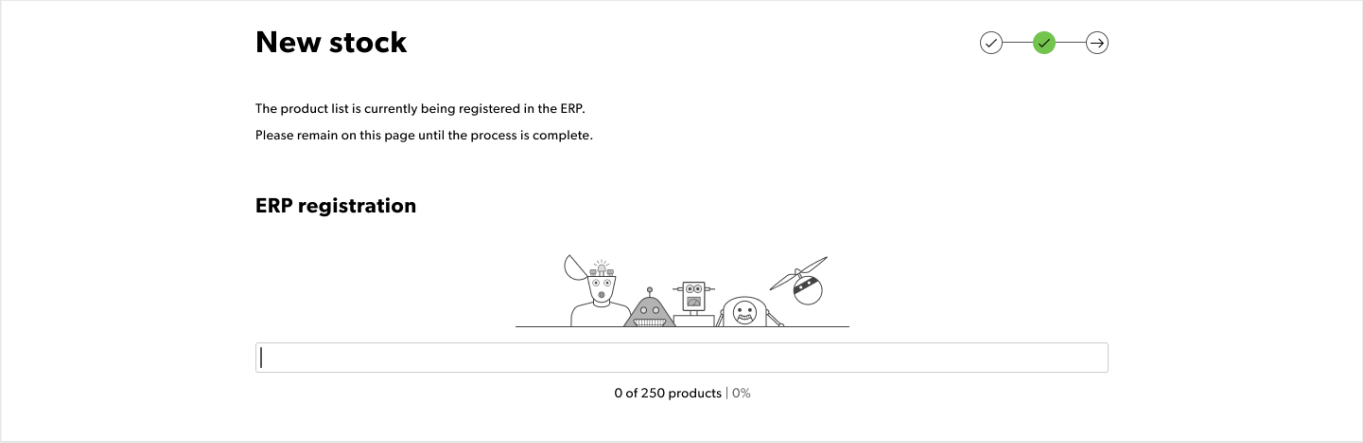


Figure 93 *- Page de chargement lors de l'enregistrement des produits dans l'ERP*

Passons directement à l'affichage du stock actuel, qui a considérablement changé. Il ne montre plus une simple liste de produits, mais une liste des différents stocks créés. Cela reflète la façon réelle de gérer le stock cible avec un stock général et des stocks saisonniers, comme expliqué à plusieurs reprises.

Chaque liste de stock est affichée avec son titre, sa succursale (ou toutes les succursales dans le cas du stock général), sa période de disponibilité, le nombre de produits différents, son statut et des actions d'édition de la liste. Le statut indique l'état de la liste afin de permettre de rapidement se rendre compte si celle-ci est à l'état de brouillon, disponible, expirée ou même supprimée.

Le panel d'actions permet de supprimer une liste (ce qui entraînera le statut "supprimé") ou de dupliquer une liste. Un administrateur a accès aux actions sur toutes les listes et peut également trier la liste pour n'afficher que les stocks d'une seule succursale ou de toutes les succursales.

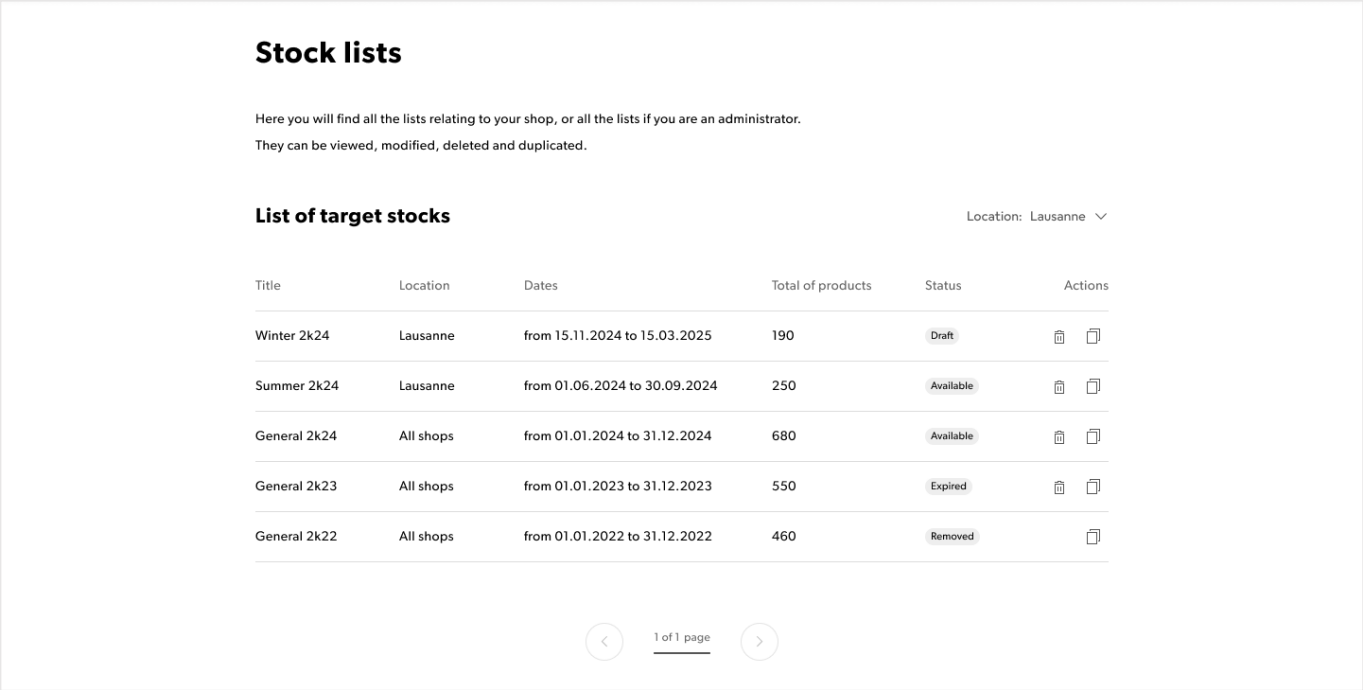


Figure 94 *- Page listant les différents stocks actifs et non actifs (version administrateur)*

Le fait de dupliquer une liste peut s'avérer utile si l'on remarque qu'une succursale a créé une liste intéressante pour la saison d'été. Au lieu de refaire une liste à partir de zéro, il est possible de la dupliquer et de l'attribuer à une autre succursale avec une période de disponibilité différente, ou non, par exemple. Le titre peut également être modifié au besoin.

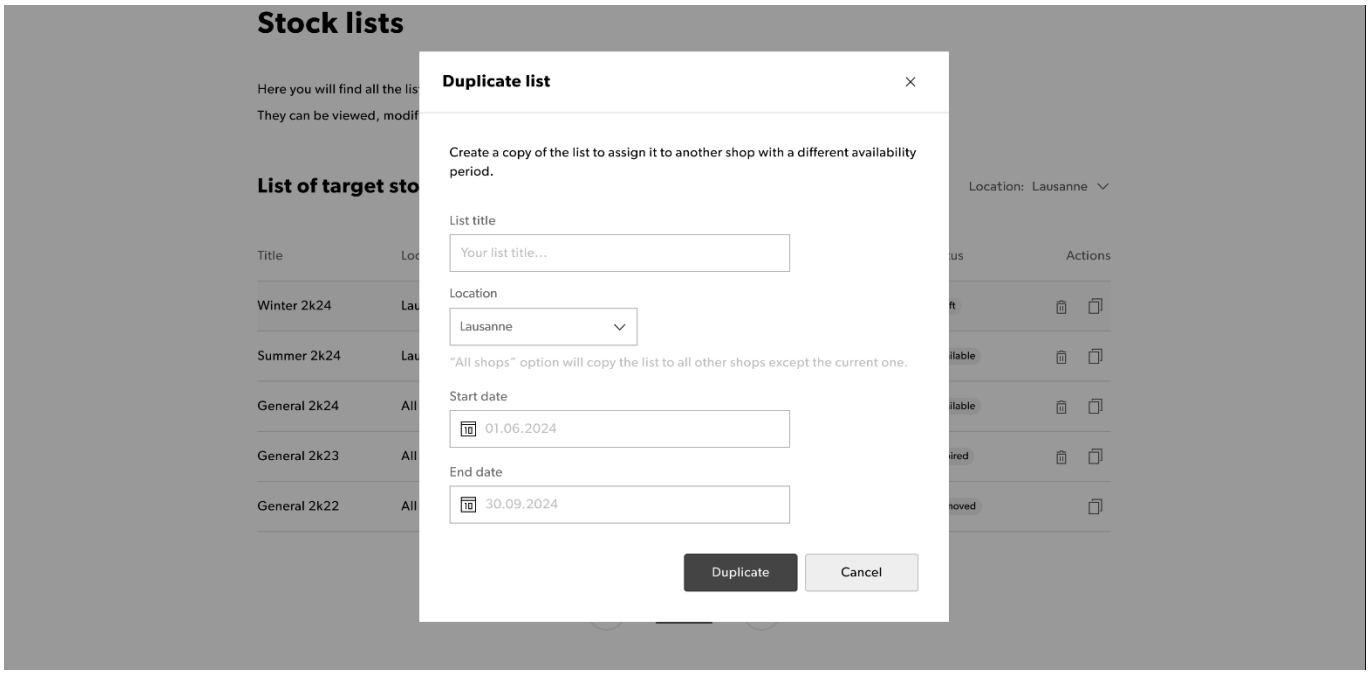


Figure 95 *- Fenêtre contextuelle permettant la duplication d'une liste (version administrateur)*

Du côté de l'utilisateur standard, la liste des différents stocks est également visible, mais elle est uniquement relative à sa succursale. De plus, il peut uniquement effectuer des actions sur les listes spécifiques à sa succursale. Comme avec les autres affichages sous forme de liste de la solution, une pagination permet de naviguer à travers les différents éléments.

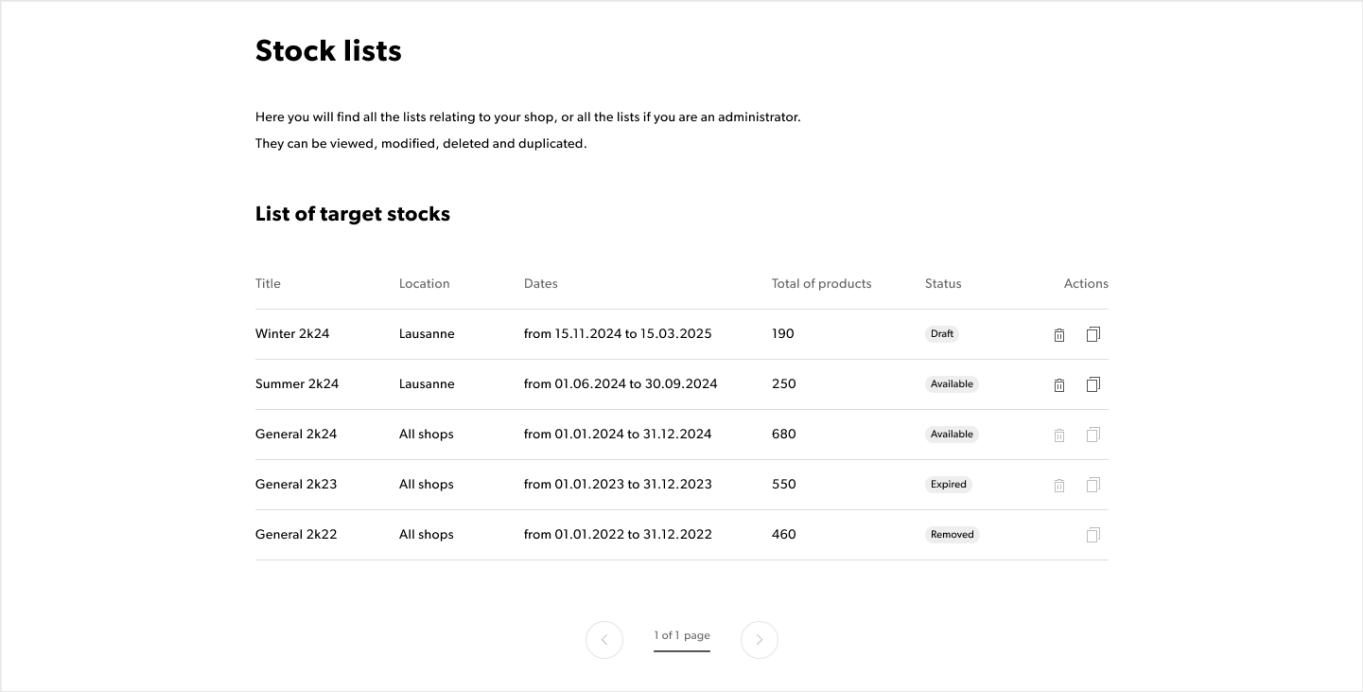


Figure 96 *- Page lisant les différents stocks actifs et non actifs*

La fonction de duplication de liste est également plus limitée pour l'utilisateur standard, ne permettant notamment pas de changer de succursale. Bien que limitée, cette fonction reste utile. Par exemple, si un utilisateur a créé l'année passée une liste pour le stock d'hiver et est toujours satisfait de son contenu, il peut simplement dupliquer l'ancienne liste, lui attribuer une nouvelle période de disponibilité, et le tour est joué.

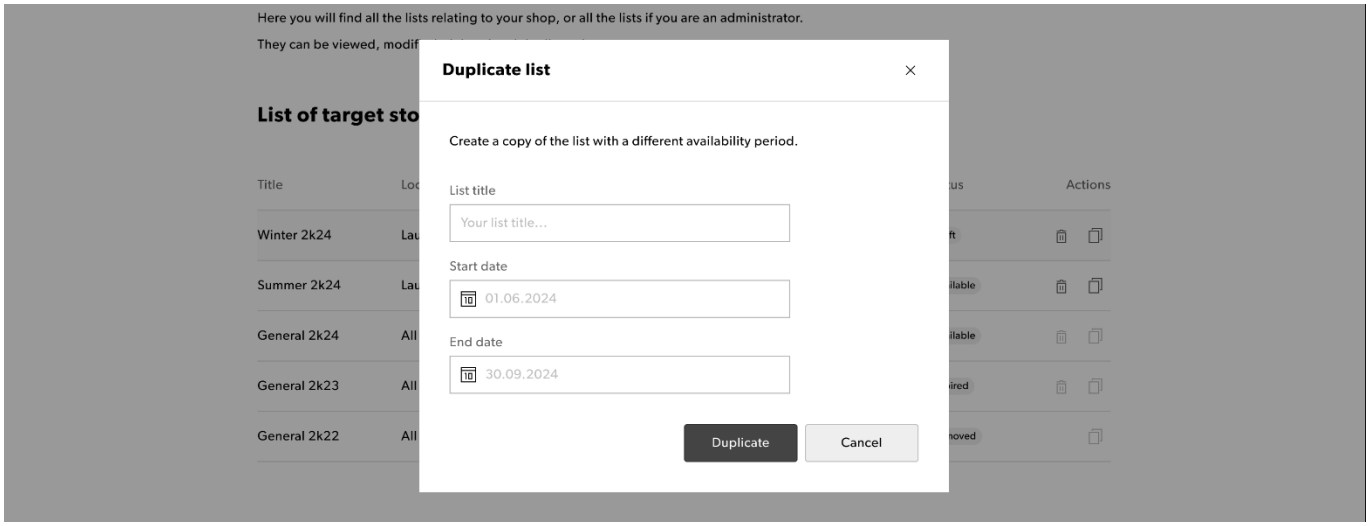


Figure 97 *- Fenêtre contextuelle permettant la duplication d'une liste*

En cliquant sur une liste, nous arrivons sur une vue sensiblement identique à celle de la précédente version de la maquette, et qui se rapproche encore davantage de la vue lors de la création du stock. Quelques différences subsistent, comme le titre de la liste affichée en grand et la description de la vue qui a été révisée.

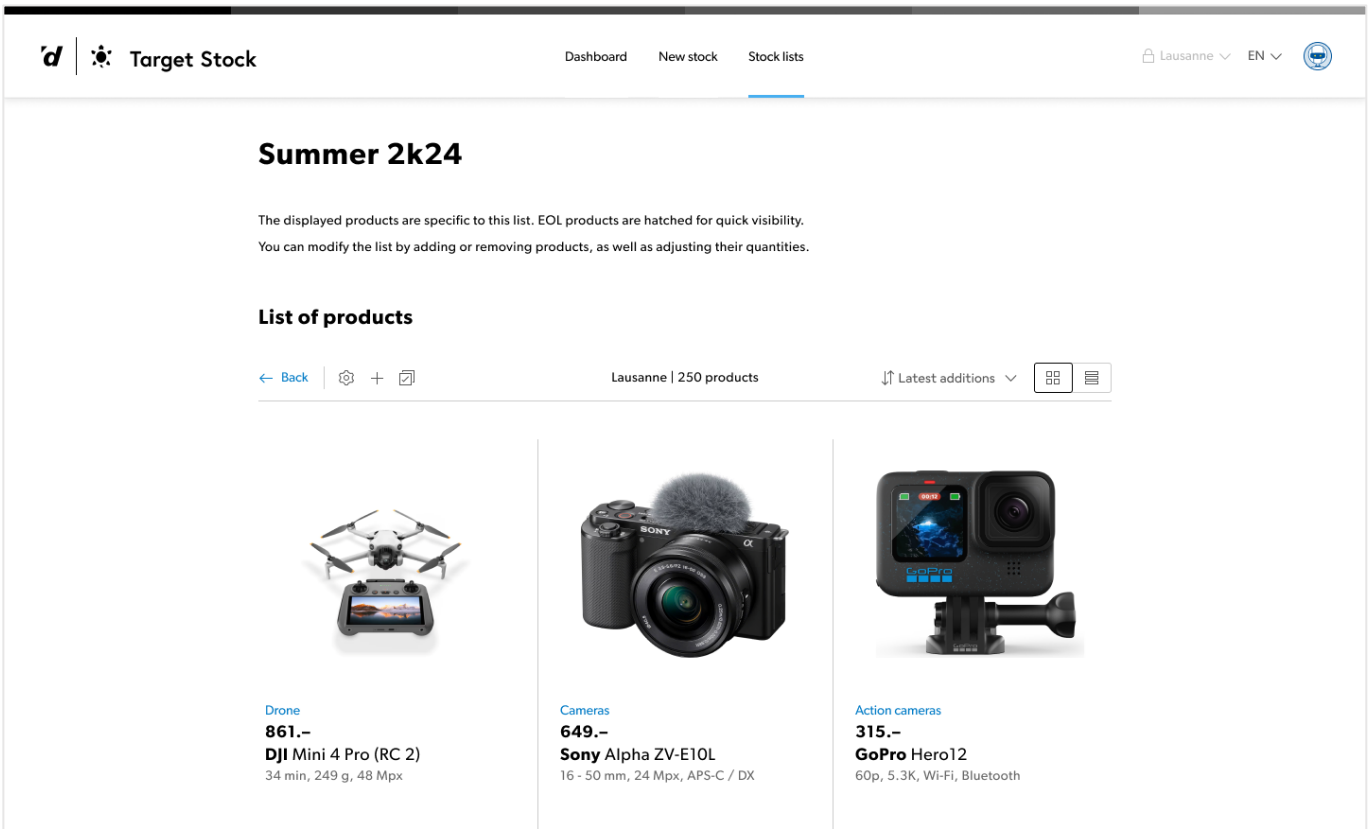


Figure 98 *- Page d'une liste actuelle de produits du stock cible*

Il est toujours possible de modifier les règles du stock cible à ce stade. La possibilité de changer le titre a été ajoutée à l'éventail d'options déjà présentes auparavant.

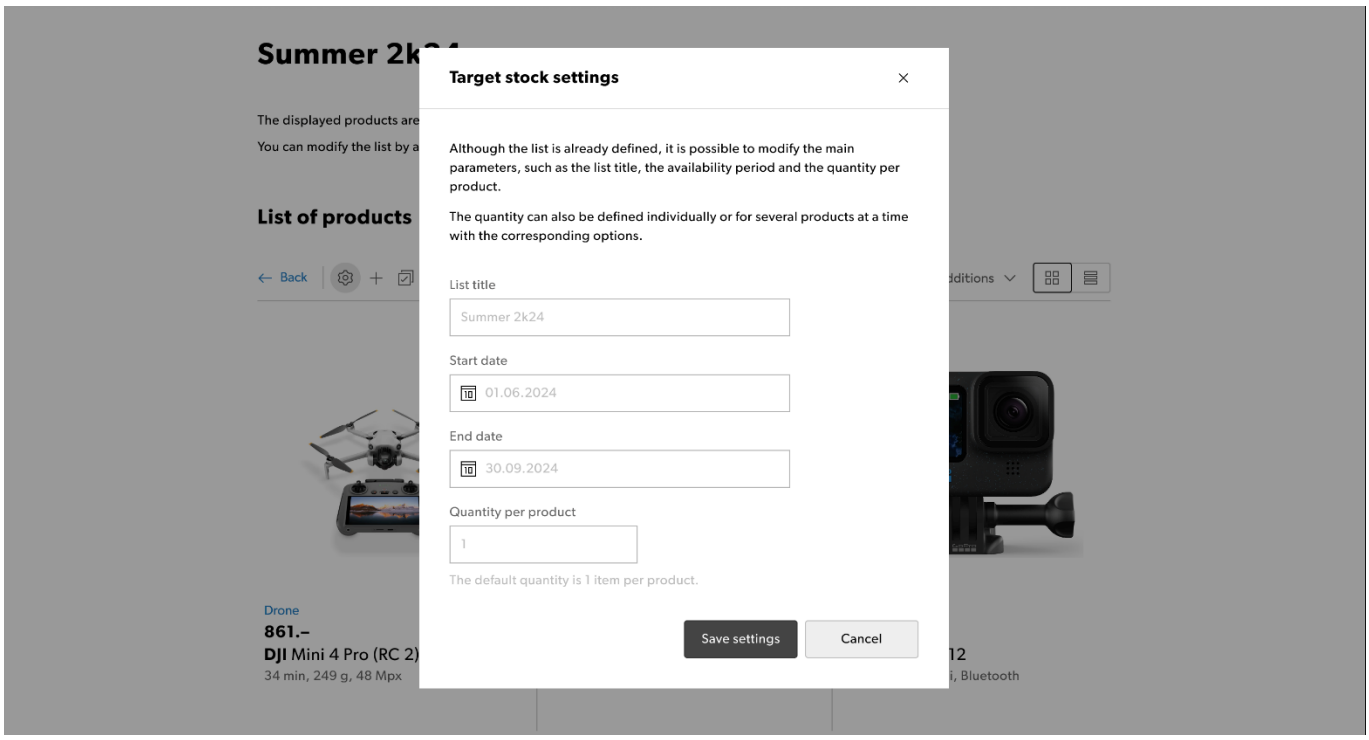


Figure 99 *- Fenêtre contextuelle permettant la redéfinition des paramètres et règles d'une liste du stock actuel*

Une personne a fait remarquer lors des tests utilisateurs qu'il serait intéressant de pouvoir identifier les produits "EOL" dans les listes. Les produits "EOL" ou "End Of Life" sont des produits qui ne sont plus vendus et que nous ne pouvons plus obtenir, ils n'ont donc plus leur place dans le stock cible.

Pour reconnaître ces produits, ils sont marqués d'un fond gris hachuré. Cela permet de les distinguer visuellement de manière subtile mais claire. La description de ces produits est faite dans le texte situé juste après le titre de la page, pour informer l'utilisateur dès son arrivée sur la liste de l'existence potentielle de produits "EOL".

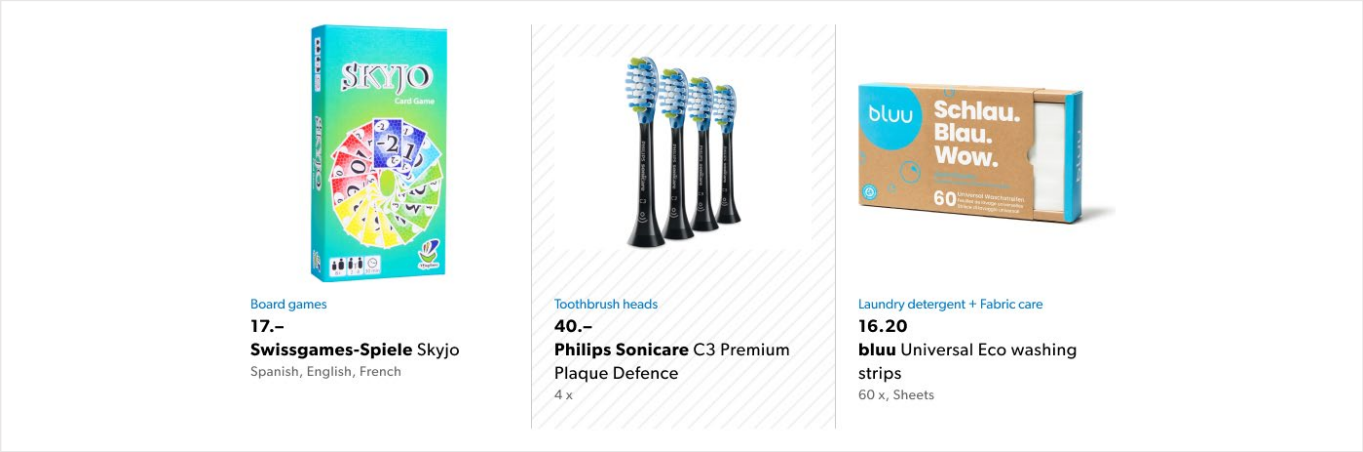


Figure 100 *- Mise en page d'un produit lorsque celui-ci est "EOL"*

Au survol des produits "EOL", un badge apparaît avec le texte "EOL" et il ne reste plus que l'option de suppression disponible pour ces produits. Cela permet de signaler clairement leur statut et de simplifier leur gestion en permettant uniquement leur suppression.

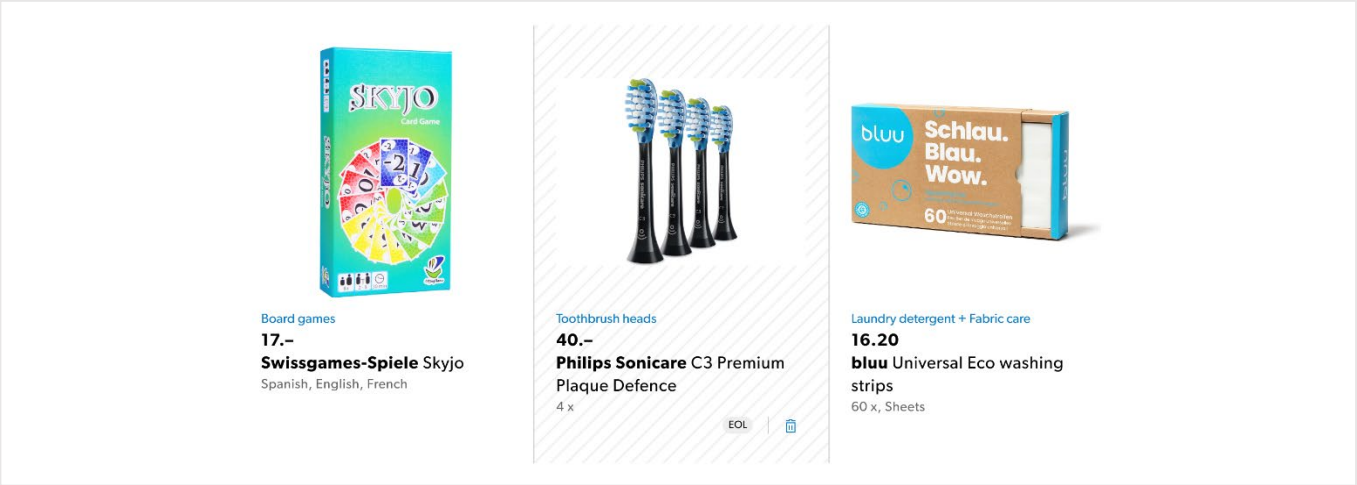


Figure 101 *- Mise en page lors du survol d'un produit "EOL"*

Il est possible de voir les produits "EOL" en premier dans la liste à l'aide de la fonction de tri. Cela permet de les identifier rapidement et de prendre les mesures nécessaires, comme leur suppression, sans avoir à parcourir toute la liste.

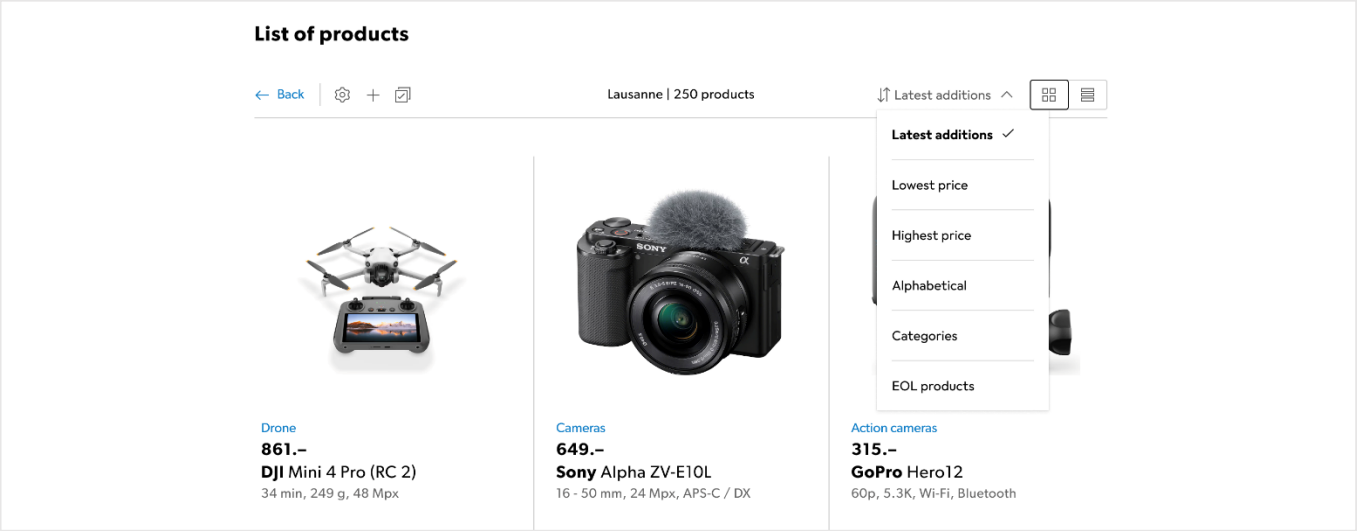


Figure 102 *- Liste déroulante permettant de trier la liste de produits et d'afficher les produits "EOL" en premier*

Il y a eu encore quelques améliorations par-ci par-là, comme des descriptions plus complètes et davantage d'info-bulles là où c'était nécessaire afin de mieux guider et aider l'utilisateur. Toutes ces améliorations ont été permises grâce aux tests utilisateurs et aux réflexions personnelles à la suite des très nombreuses heures passées sur la maquette.

Il est possible de tester cette version définitive de la maquette via ce lien Figma : Prototype Figma (version définitive). De plus, le fichier Figma de la maquette est également annexé au présent rapport via la clé USB fournie.

Petite note lors du test du prototype, il est possible de cliquer sur le cadenas à côté de la liste déroulante des succursales dans l'entête pour pouvoir ensuite se rendre sur les vues d'administrateur des pages.

2.7 Répercutions sur les autres phases

Cette phase de conception est à présent achevée et la suite du travail peut commencer en théorie. Cependant, cette phase a pris beaucoup plus de temps que prévu, ce qui l'a fait déborder sur les prochaines phases. Cela est principalement dû à une sous-estimation du temps nécessaire pour réaliser toutes ces premières tâches et aux quelques problèmes rencontrés tout au long du travail.

Cela a grandement impacté la suite du développement, car moins de temps restait à disposition. C'est pourquoi les prochaines parties de ce rapport sont bien moins conséquentes que cette première phase.

3. Phase 2 - Développement des fonctionnalités

3.1 Fonctionnalité de génération automatique des produits

Lors de la première phase, la génération de produits est restée à un stade préliminaire. En effet, il était possible de récupérer une liste de produits à partir d'une URL menant aux produits d'une catégorie sur les plateformes de vente, ainsi qu'une liste de catégories, mais cela s'arrêtait là.

Cette partie du projet est disponible dans un repository GitHub à l'adresse : https://github.com/nicomeuwly/dg-targetstock/tree/main. Il contient également une marche à suivre pour tester le code localement sur sa propre machine. De plus, ce code est également annexé au présent rapport via la clé USB fournie.

3.1.1 Utilisation du web scraping

N'ayant pas accès aux bases de données de l'entreprise lors du début de cette phase, le web scraping a été privilégié pour poursuivre le développement. Il faut donc maintenant récupérer des produits à partir des milliers de liens obtenus avec la fonction dédiée aux catégories présentée en phase 1. Mais ce n'est pas tout, il manque quelques vérifications avant de simplement récupérer les produits. En effet, il nous manque une méthode permettant de vérifier si un produit peut être retiré en succursale et une autre pour voir le niveau de stock du produit. Si un produit ne peut pas être retiré ou s'il est en stock mais en trop faible quantité, il ne pourra pas faire partie du stock cible.

Les fonctions se chargeant des produits ont donc été complètement revues pour tenter d'atteindre l'objectif de cette phase. Nous allons commencer par la fonction qui permet de récupérer les détails des produits. Pour que la fonction `getProductsDetails()` fonctionne, elle a besoin de deux paramètres : une page provenant de Puppeteer et une URL d'une catégorie de produit.

Sans entrer trop dans les détails, la fonction va commencer par vérifier si l'URL contient le caractère "?" (synonyme de paramètre d'une URL) pour savoir s'il faut ajouter des paramètres avec "&" parce que des paramètres sont déjà présents, ou s'il faut inclure le "?" pour indiquer qu'il n'y aura que nos paramètres à nous. Les paramètres que nous souhaitons ajouter sont un filtre afin d'avoir uniquement les produits en stock à l'entrepôt, "filter=t_off%3DInStock", et un tri pour afficher les meilleures ventes en premier, "so=15". Il est possible de voir quel nombre est attribué à quel tri en se rendant sur Digitec ou Galaxus et en choisissant un tri. Ce dernier sera alors automatiquement représenté dans l'URL de la page.

Le reste de la fonction est globalement similaire à celle de la phase de test. Une page est ouverte à l'URL fournie, suivie de nos paramètres, puis les produits vont tenter d'être trouvés. S'il y en a, la fonction va extraire toutes les informations souhaitées pour, à la fin, les insérer dans un objet qui finira dans un tableau.

```
const getProductsDetails = async (page : any, url: string) => {
    const filters = url.includes("?")
        ? "&filter=t_off%3DInStock&so=15"
        : "?filter=t_off%3DInStock&so=15";
    await page.goto(`https://www.galaxus.ch${url + filters}`, {
        waitUntil: "domcontentloaded",
    });

    const articles = await page.$$("div.sc-5d2f6f43-1.jwLEDS > article");
    if (articles.length === 0 || !articles) return [];

    const products: any[] = [];

    for (const article of articles) {
        const product = await article.$eval(
            "span.sc-6923aaa7-0.fkBRnq",
            (el: any) => el.textContent
        );
        const brand = await article.$eval("strong", (el: any) => el.textContent);
        const priceSpan = await article.$eval(
            "span.sc-812f8453-1.fMoCQC",
            (el: any) => el.textContent
        );
        const price = parseFloat(priceSpan.replace("CHF", "").replace(",", "."));
        const detailsTemp = await article.$("p. sc-ce74e31a-9.gcbXXJ");
        const details = detailsTemp
            ? await detailsTemp.evaluate((el: any) => el.textContent)
            : "";
        const imageURL = await article.$eval("img", (el: any) =>
            el.getAttribute("src")
        );
        const productUrl = await article.$eval(
            "a.sc-b0fdbb10-0.iNiATr",
            (el: any) => el.getAttribute("href")
        );
        const category = await article.$eval(
            "a.sc-ccd25b80-0.beNCEW.sc-430e9524-6.gSfbKC",
            (el: any) => el.textContent
        );
        const categoryUrl = await article.$eval(
            "a.sc-ccd25b80-0.beNCEW.sc-430e9524-6.gSfbKC",
            (el: any) => el.getAttribute("href")
        );
        const id = productUrl?.match(/[^-]*$/)?.pop()?.slice(0, 8);
        products.push({id, product, brand, details, price, imageURL,
        url: productUrl, category, categoryUrl, scrapedCategory: url });
    }

    return products;
};
```

Listing 18 *- Code de la fonction permettant de récupérer les détails des produits d'une catégorie sur galaxus.ch*

La prochaine fonction a justement besoin de la précédente pour fonctionner. La fonction `getProductsList()` a également deux paramètres : une page provenant de Puppeteer et un tableau de catégories. La fonction commence par vérifier que le tableau de catégories n'est pas vide, puis initie un tableau vide.

Ensuite, pour chacun des éléments contenus dans le tableau de catégories, une liste temporaire est créée, contenant le résultat de la fonction `getProductsDetails()`, à qui on aura transmis la page contenue en paramètres et l'URL de chaque catégorie. Une fois la liste temporaire prête, son contenu est inséré dans le tableau vide initié précédemment. La boucle exécute ces actions du premier au dernier élément du tableau de catégories. Finalement, le tableau de produits est retourné comme résultat de la fonction.

```
const getProductsList = async (page : any, categories: any[]) => {
    if (categories.length === 0) return [];
    const products: any[] = [];
    for (const category of categories) {
        const tempList = await getProductsDetails(page, category.url);
        products.push(...tempList);
    }
    return products;
};
```

Listing 19 *- Code de la fonction récupérant les détails des produits d'une liste de catégories sur galaxus.ch*

La prochaine fonction va permettre de vérifier si un produit est, d'une part, retirable en succursale et, d'autre part, s'il y en a au moins 10 pièces en stock. La fonction `openProductPage()` va, comme son nom l'indique, ouvrir une page de produit. Elle prend comme paramètres une page de Puppeteer, un objet de type "Product" et un tableau vide que l'on va ensuite remplir de produits.

La fonction commence donc par essayer d'accéder à l'URL du produit contenu dans l'objet "product". Une fois sur cette page, elle va chercher un élément particulier, le message informant de la disponibilité du produit. Cet élément est cliqué pour révéler les détails de disponibilité, qui contiennent notamment la liste des succursales si le produit peut être retiré dans ces dernières.

La fonction va ensuite tenter de trouver l'élément et, si elle y arrive, elle va pouvoir en extraire les informations. Les succursales sont écrites en utilisant la balise `<strong>`, synonyme de texte en gras, et on va chercher à récupérer ces éléments via le nom de la balise. On va également récupérer le texte relatif à l'expédition, car c'est là que la quantité en stock du produit est indiquée.

Enfin, on va vérifier via une condition si le texte précédent contient le nombre 10 et si nous avons plus de 0 élément `<strong>`. Si la condition s'avère exacte, le produit sera ajouté au tableau indiqué en paramètre. Une série de gestion des erreurs suit afin d'avoir des retours sur la nature des erreurs et de relancer le processus si nécessaire.

```
const openProductPage = async (page : any, product: Product, stocks: Product[])
=> {
    try {
        await page.goto("https://www.galaxus.ch" + product.url, { waitUntil:
        "domcontentloaded" });

        await page.$eval("button.sc-58bde996-0.bQTVcy", (el: HTMLElement) =>
        el.click())

        await page.waitForSelector("div.sc-d0d34be1-2.dMCIvG", { visible: true,
        timeout: 5000 });
        const availability = await page.$("div.sc-d0d34be1-2.dMCIvG");

        if (availability) {
            const locations = await availability.$$("strong");
            const stockLevel = await availability.$eval("div.sc-dbd9c505-
            1.fwiPKQ", (el: any) => el.textContent);

            if (stockLevel.includes("10 items") && locations.length > 0) {
                stocks.push(product);
            }
        } else {
            console.error(`Availability div not found for product ${
            product.id}`);
        }

    } catch (error: any) {
        console.error(`Error processing product ${product.id}:`, error);

        if ((error.message.includes('net::ERR_HTTP2_PROTOCOL_ERROR') ||
        error.message.includes('Node is detached from document')) && retryCounter
        < 3) {
            retryCounter++;
            console.log(`Retrying product https://www.galaxus.ch${
            product.url}...`);
            await new Promise(resolve => setTimeout(resolve, 1000));
            await openProductPage(page, product, stocks);
        }

    }
}
```

Listing 20 *- Code de la fonction permettant de vérifier si un produit peut être inclus dans le stock cible*

Le type de l'objet "product" demandé en paramètre de la fonction `openProductPage()` a été défini au préalable, comme le montre l'extrait de code ci-dessous. Cela permet de spécifier précisément les informations que doit contenir un objet de type "Product".

```
interface Product {
    id: number;
    product: string;
    brand : string;
    details: string;
    price: number;
    imageURL: string;
    url: string;
    category: string;
    categoryUrl: string;
    scrapedCategory: string;
}
```

Listing 21 *- Code définissant le type "Product"*

La fonction précédente va automatiser le processus de vérification de la disponibilité des produits comme illustré ci-dessous. On voit notamment le message en vert au-dessus du bouton d'ajout au panier. Lorsque l'on survole ou clique sur ce dernier, une petite fenêtre apparaît avec les détails de la disponibilité.

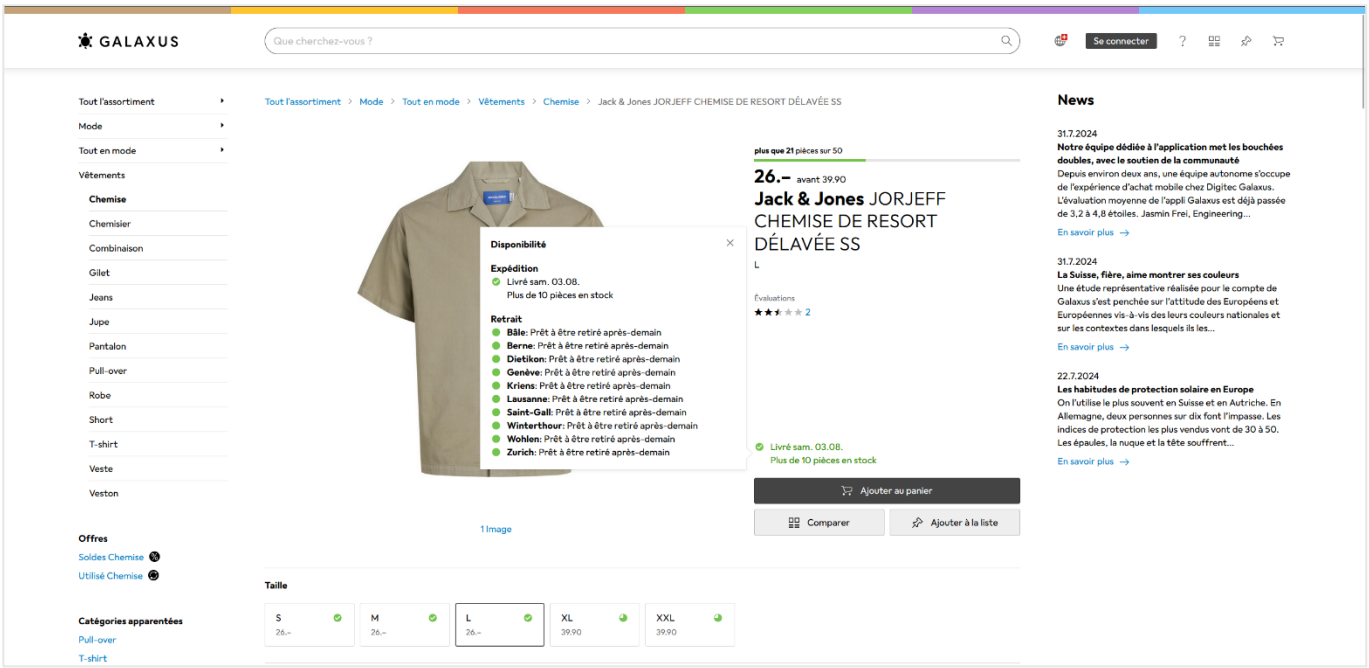


Figure 103 *- Page d'un produit sur galaxus.ch illustrant les informations de disponibilité de ce dernier*

Maintenant que nous avons un moyen de vérifier les différents critères afin de déterminer si un produit peut aller dans le stock cible, nous devons appliquer cette vérification à tous les produits. C'est là que la fonction `getProductsAvailability()` entre en jeu. Cette dernière a besoin d'une page et d'un tableau de produits.

La fonction commence par définir un tableau vide, qui sera la destination des produits validés pour le stock cible. Ensuite, pour chaque élément du tableau `products` fourni en paramètres, la fonction `openProductPage()` décrite précédemment va s'exécuter avec la page fournie en paramètres, un élément du tableau `products`, et enfin le tableau vide `stocks`.

Un temps d'attente d'une seconde est ajouté avant de passer au produit suivant afin de limiter les erreurs. Un compteur d'essais, également présent dans la fonction précédente, est remis à zéro en cas de réussite. Ce compteur est utilisé dans `openProductPage()` pour réessayer jusqu'à trois fois maximum avant d'abandonner la tentative de récupération des détails de disponibilité.

Une fois la boucle terminée, la fonction retourne le tableau `stocks` comme résultat.

```
const getProductsAvailability = async (page: any, products: any[]) => {
    const stocks : any[] = [];
    console.log("Total products : " + products.length);

    for (const product of products) {
        await openProductPage(page, product, stocks);

        await new Promise((resolve) => setTimeout(resolve, 1000));
        retryCounter = 0;
    }

    return stocks;
};
```

Listing 22 *- Code de la fonction parcourant tous les produits et vérifiant leur inclusion dans le stock cible*

La dernière fonction majeure, `getProducts()`, va être la base sur laquelle une partie des autres fonctions vont se reposer. En effet, elle va fournir des éléments indispensables pour que le reste fonctionne. Elle va initialiser un navigateur virtuel, utilisé pour réaliser les différentes actions sur le web, et créer une nouvelle page. Cette dernière sera transmise aux fonctions `getProductsList()` et `getProductsAvailability()`, qui vont également la transmettre là où cela est nécessaire.

La fonction `getProducts()` a besoin de recevoir un tableau contenant des catégories pour fonctionner. Lors des premiers tests et pour éviter que le processus dure plusieurs heures en raison des milliers de catégories présentes sur le site, ce tableau est réduit pour n'utiliser que les premiers éléments, les trois premiers en l'occurrence dans l'extrait de code ci-dessous.

La fonction va commencer par lancer la fonction `getProductsList()` avec les catégories et la page, puis son résultat va être transmis à `getProductsAvailability()` avec la page également. C'est le résultat de cette dernière qui sera finalement retourné par `getProducts()`, pour autant que le tableau de résultats "stocks" ne soit pas vide. Le cas échéant, tout le processus est relancé.

Ainsi, `getProducts()` joue un rôle central en orchestrant l'exécution des autres fonctions, garantissant que les produits sont correctement récupérés et vérifiés avant d'être inclus dans le stock cible.

```
const getProducts = async (categories: any[]) => {
    try {
        const browser = await puppeteer.launch({
            args: ["--no-sandbox", "--disable-setuid-sandbox"],
        });
        const page = await browser.newPage();
        const tempCategories = categories.slice(0, 3);

        const products = await getProductsList(page, tempCategories);
        const stocks = await getProductsAvailability(page, products);

        await browser.close();
        if (stocks.length === 0) {
            await getProducts(categories);
        } else {
            return stocks;
        }
    } catch (error: any) {
        console.error(error.message);
    }
}
```

Listing 23 *- Code de la fonction exécutant la récupération et la vérification des produits*

Pour récupérer nos résultats, une requête API sera préparée en allant tout d'abord chercher les catégories stockées localement dans un fichier JSON. Ces catégories seront ensuite transmises à la fonction `getProducts()` pour obtenir la liste des produits destinés au stock cible. Ces derniers seront enregistrés localement dans un fichier JSON, comme lors des phases de test, afin de visualiser rapidement la structure des résultats sans devoir les afficher dans la console après avoir relancé la requête.

Si tout se passe bien, la méthode GET retournera une réponse contenant les produits et un statut de succès. En cas de problème, une réponse avec un message d'erreur et un code d'erreur sera transmise.

```
export async function GET() {
    try {
        const categories = JSON.parse(fs.readFileSync("./data/categories.json",
        "utf8"));
        const products = await getProducts(categories);
        const data = JSON.stringify(products);
        fs.writeFileSync("./data/products.json", data);
        return new Response(data, {
            status: 200,
            headers: { "Content-Type": "application/json" },
        });
    } catch (error: any) {
        return new Response(JSON.stringify({ error: error.message }), {
            status: 500,
            headers: { "Content-Type": "application/json" },
        });
    }
}
```

Listing 24 *- Code de la route API permettant de récupérer une liste de produits*

Finalement, pour récupérer et afficher les résultats sur une page, une petite fonction se chargera d'appeler la requête API pour recevoir les produits. Durant cette phase, l'interface de test créée au début du projet a encore été utilisée pour se concentrer en premier lieu sur le fonctionnement de cette fonctionnalité avant de s'occuper de son apparence. Un compteur sera déclenché au moment de l'envoi de la requête pour afficher le temps nécessaire à cette dernière pour retourner une réponse.

```
const getProducts = async () => {
    setData([]);
    setLoading(true);
    setTime(0);
    setRunning(true);
    try {
        const res = await fetch("/api/data/products");
        const data = await res.json();
        setData(data);
    } catch (error) {
        console.error("Failed to fetch data:", error);
    } finally {
        setLoading(false);
        setRunning(false);
    }
};
```

Listing 25 *- Code de la fonction permettant de récupérer les données de produits via la route API dédiée*

Plusieurs essais ont été effectués avec ce processus en commençant avec une seule catégorie. Pour déterminer combien de catégories nous souhaitons traiter, il suffit de le spécifier dans la première fonction `getProducts()` lorsqu'elle traite le tableau de catégories. La méthode "slice" appliquée au tableau de catégories permet de définir à partir de quel index du tableau nous souhaitons créer notre nouvelle liste et à quel index cette dernière s'arrête.

Dans l'extrait de code ci-dessous, nous prenons les éléments depuis l'index 0 jusqu'à l'index 3 non compris. Cela fait donc trois éléments extraits du tableau initial (index 0, 1, 2).

```
const tempCategories = categories.slice(0, 3);
```

Listing 26 *- Code illustrant le découpage d'un tableau*

Pour vérifier que les résultats soient corrects, il a fallu accéder à chaque catégorie testée et compter manuellement les produits. Premièrement, il est nécessaire de savoir combien de produits le code va traiter, et cela peut être rapidement déterminé. Lorsqu'on applique un filtre sur Digitec ou Galaxus, on peut voir la quantité de produits qui correspondent à ce filtre. Par exemple, si le filtre indique 952 produits, cela signifie que 952 produits correspondent aux critères spécifiés.

Ces produits sont affichés par groupes de 48 avant qu'il soit nécessaire de cliquer sur le bouton "Afficher plus" pour voir la suite des produits. Dans notre exemple, nous savons que le code va traiter 48 produits, car il s'arrête à ce premier groupe.

Pour savoir combien de produits seront finalement récupérés puis affichés dans l'interface de test, il faut survoler un par un la pastille indicatrice de disponibilité de chaque produit pour connaître les détails. Cela permettra de vérifier s'il y a au moins 10 pièces en stock et si le produit peut être retiré en succursale.

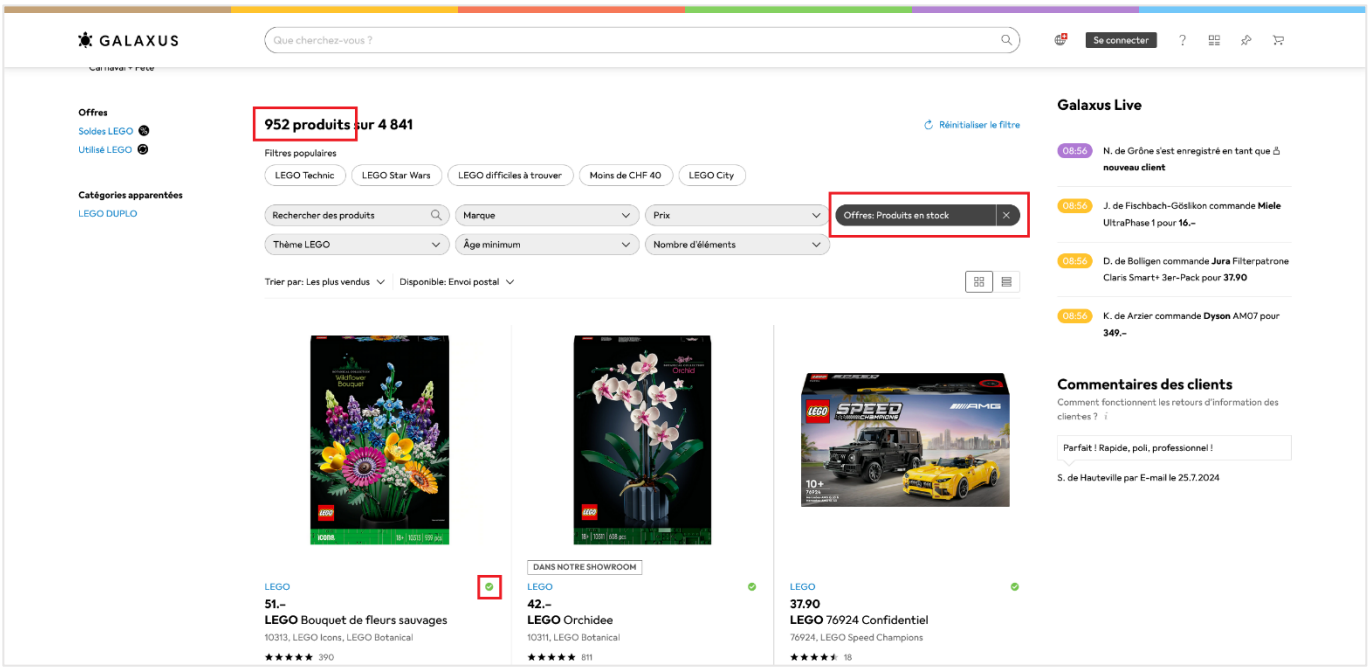


Figure 104 *- Page d'une catégorie illustrant le total de produits, les filtres et l'indicateur de disponibilité*

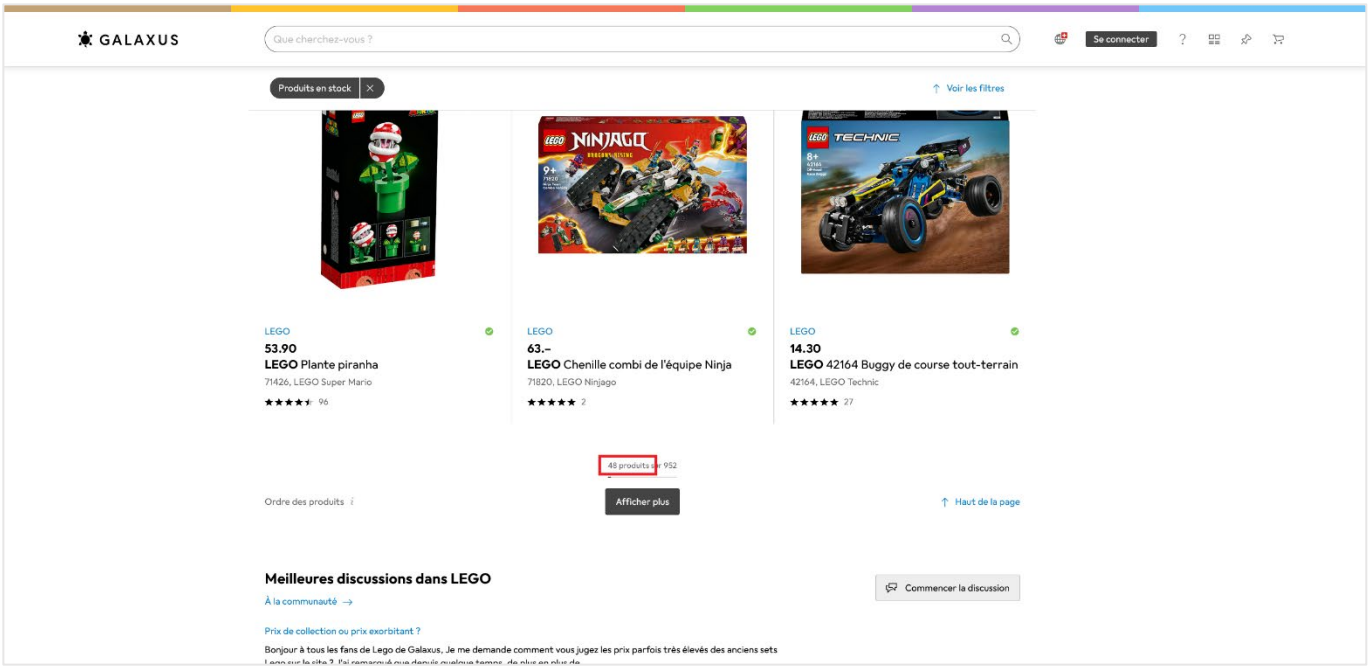


Figure 105 *- Page d'une catégorie sur galaxus.ch illustrant l'emplacement du nombre de produits affichés*

Les cinq premières catégories contenues dans le fichier JSON des catégories ont été utilisées lors de ces tests. À chaque itération, nous avons testé avec une catégorie supplémentaire, en commençant par une seule et en augmentant progressivement jusqu'à inclure les cinq premières catégories.

Ordre d'essai	Catégorie	URL
1	Câble audio	https://www.galaxus.ch/fr/s1/producttype/cable-audio-296?tagIds=591-1388
2	Connecteur audio	https://www.galaxus.ch/fr/s1/producttype/connecteur-audio-3137?tagIds=591-1388
3	Accessoires Audio Player	https://www.galaxus.ch/fr/s1/tag/accessoires-audio-player-548?tagIds=591-1388
4	Coffre + housse audio	https://www.galaxus.ch/fr/s1/producttype/coffre-housse-audio-3019?tagIds=591-1388
5	Accessoires hifi auto	https://www.galaxus.ch/fr/s1/tag/accessoires-hifi-auto-542?tagIds=591-1388

Tableau 2 *- Liste des essais de récupération de produits à partir d'un nombre défini de catégories*

Lors de l'essai avec une seule catégorie et en comptant manuellement, le site propose 48 produits, dont 37 peuvent aller dans le stock cible. L'objectif du système est donc de récupérer ces 37 produits. Après 102 secondes, le système a réussi à récupérer tous les produits corrects.

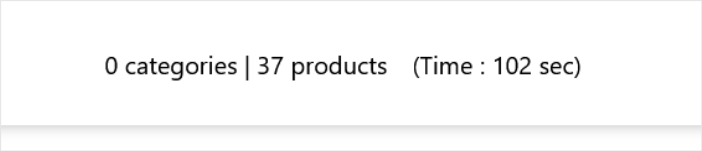


Figure 106 *- Interface de tests avec les résultats de la requête de récupération des produits avec 1 catégorie*

Lors de l'essai avec 2 catégories et en comptant manuellement, le site propose 70 produits et 42 d'entre eux peuvent aller dans le stock cible. L'objectif du système est donc de récupérer ces 42 produits. Après 160 secondes, le système a bien récupéré tous les produits.

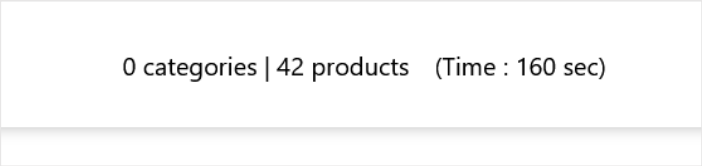


Figure 107 *- Interface de tests avec les résultats de la requête de récupération des produits avec 2 catégories*

Lors de l'essai avec 3 catégories et en comptant manuellement, le site propose 118 produits et 54 d'entre eux peuvent aller dans le stock cible. L'objectif du système est donc de récupérer ces 54 produits. Après 267 secondes, le système a récupéré 53 produits. Une erreur dans la console indique que le système n'a pas trouvé un sélecteur CSS sur une page et en consultant cette dernière, il s'agit de celle du produit manquant. Il est difficile de déterminer pourquoi ce produit en particulier n'a pas fonctionné, car il est identique aux autres dans sa structure en termes de code HTML.

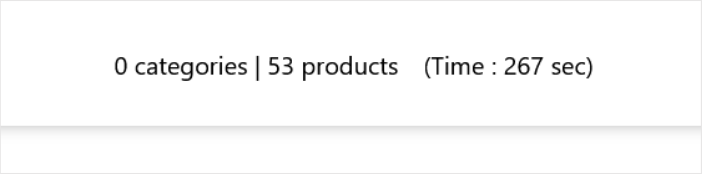


Figure 108 *- Interface de tests avec les résultats de la requête de récupération des produits avec 3 catégories*

Lors de l'essai avec 4 catégories et en comptant manuellement, le site propose 125 produits et 54 d'entre eux peuvent aller dans le stock cible. L'objectif du système est donc de récupérer ces 54 produits. Après 300 secondes, le système a bien récupéré tous les produits. Le nombre de produits à obtenir est identique à l'essai précédent, car la 4e catégorie ne possède aucun produit pouvant aller dans le stock cible. L'erreur apparue à l'essai précédent ne s'est pas répétée ici.

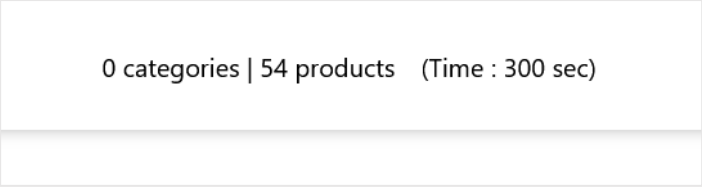


Figure 109 *- Interface de tests avec les résultats de la requête de récupération des produits avec 4 catégories*

Lors de l'essai avec 5 catégories et en comptant manuellement, le site propose 173 produits et 93 d'entre eux peuvent aller dans le stock cible. L'objectif du système est donc de récupérer ces 93 produits. Après 287 secondes, le système n'a récupéré que 79 produits. Ce qui est étonnant, c'est que la console affiche qu'elle a trouvé un total de 125 produits, identique à l'essai précédent. Le système a trouvé plus de produits qu'avant mais moins que le nombre espéré. Il n'y avait même pas tant d'erreurs que cela dans la console, peut-être 2-3 mais pas 14. La 5e catégorie a été testée individuellement, le premier essai n'a rien donné et le second a permis de récupérer ses produits. Il est difficile de comprendre pourquoi cela s'est passé ainsi.

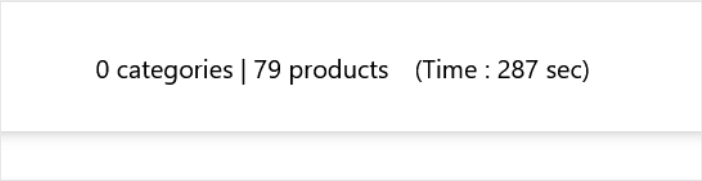


Figure 110 *- Interface de tests avec les résultats de la requête de récupération des produits avec 5 catégories*

Les tests se sont arrêtés ici, car, plus on augmente le nombre de catégories, plus le temps de traitement s'allonge. De plus, le temps nécessaire pour vérifier manuellement toutes les catégories sur le site serait démesurément long. En moyenne, il a fallu 2,274 secondes par produit pour le trouver et vérifier s'il peut aller dans le stock cible.

Le fichier JSON contenant les catégories possède 3157 éléments de catégorie. Bien que toutes les catégories n'aient pas nécessairement 48 produits à traiter, nous pouvons aisément estimer combien de temps le processus prendrait dans le pire des cas pour analyser ces milliers de catégories. Avec 48 produits par catégorie, cela revient à 151'536 produits. S'il faut 2,274 secondes pour traiter chacun d'entre eux, nous arrivons à un total de 3 jours, 23 heures, 43 minutes et 13 secondes. Quasiment 4 jours de traitement ne sont pas acceptables et ne permettent donc pas d'optimiser le temps nécessaire à cette tâche.

Il serait possible d'ajouter une étape supplémentaire dans le code pour ne récupérer que les 5 premiers éléments qui peuvent aller dans le stock cible, par exemple. Malheureusement, il n'est pas possible de savoir à l'avance combien d'éléments il faudra analyser avant d'atteindre ce total de 5. Cette étape a été testée en partie mais n'a pas été une réussite.

Il y a bien sûr toujours la possibilité de spécifier quelles catégories nous souhaitons pour le stock cible, ce qui pourrait réduire considérablement le temps de traitement. Cependant, en raison des problèmes rencontrés et du temps restant après la phase 1 qui a largement débordé sur le reste, cette fonctionnalité n'a pas été développée davantage.

À cela s'ajoute le fait que cette méthode ne permet pas de vérifier correctement si un produit peut être retiré en succursale. En effet, nous regardons s'il y a au moins 10 pièces en stock, car le message n'affiche pas de nombre au-delà. Le problème est qu'il faudrait en réalité au moins 20 produits en stock pour les inclure dans le stock cible. Comme il y a 10 succursales et que la quantité par défaut par produit est de 1, cela laisserait au moins 10 pièces en stock à l'entrepôt pour les personnes souhaitant se faire livrer ces produits chez eux.

Le seul moyen avec le web scraping pour récupérer cette information serait de se rendre dans l'ERP pour chaque produit afin de voir la quantité réelle, ce qui rallongerait encore le processus. De plus, cela impliquerait de trouver une solution pour automatiser la connexion au système, ajoutant une couche de complexité supplémentaire à l'automatisation.

3.1.2 Accès à la base de données

Un accès à la base de données de l'entreprise a finalement été octroyé à la fin de la semaine 7 de cette phase de réalisation, à la suite d'une visioconférence avec une Product Owner du secteur Business Intelligence. Étant donné le moment où cette nouvelle est arrivée, il ne restait plus beaucoup de temps pour se documenter et trouver une façon d'utiliser cet accès. La base de données est hébergée dans le cloud et il est possible d'y accéder directement depuis une plateforme web. Pour des raisons de confidentialité, le nom du service utilisé ne sera pas communiqué.

Lors de la visioconférence, le fonctionnement de la base de données a été évoqué et des pistes ont été données pour en extraire les données. Depuis la plateforme web, il est possible de créer des requêtes vers les différentes tables de la base de données et d'exporter les résultats dans un fichier pour les télécharger. Cela peut être utile, mais implique également de devoir automatiser un accès à cette plateforme pour exécuter la requête et télécharger le résultat. Le web scraping pourrait être utilisé, mais cela ne va pas être simple, comme vu précédemment. Dans ce cas de figure, il faudrait en plus pouvoir passer le système de connexion à doubles facteurs.

Une seconde façon de procéder est d'utiliser une librairie qui fera le pont entre la solution du stock cible et la plateforme de la base de données. Cela est notamment possible avec Python, car c'est ce qu'utilise l'entreprise. Cependant, pour utiliser cela avec JavaScript, cela risque d'être plus compliqué, car ce n'est pas le langage proposé de base par la plateforme pour la relier à un projet.

En attendant de trouver une solution pour relier la base de données au projet, des tests de requêtes ont été effectués depuis la plateforme afin de vérifier si on peut obtenir les résultats souhaités. Il faut savoir que chaque requête exécutée coûte de l'argent à l'entreprise. La plateforme facture 5,00 $ par téraoctet de données traitées. La base de données fonctionne avec plusieurs tables qui contiennent chacune des informations spécifiques et plus on lie de tables entre elles, plus la quantité de données sera importante.

Voici des exemples factices de tables que l'on peut retrouver dans une base de données. Il y a une table qui contient une liste de produits, chacun ayant un identifiant unique (ID) sous forme de nombre, un nom, une marque et une catégorie sous forme de texte, et enfin un prix qui est un nombre. La deuxième table répertorie le niveau de stock dans les différents entrepôts. Chaque élément de cette table a un identifiant unique et un identifiant de produit sous forme de nombre, un entrepôt sous forme de texte et enfin une quantité qui est un nombre.

ID	Nom	Marque	Categorie	Prix
Nombre	Texte	Texte	Texte	Nombre

***Tableau 3** – Exemple d'une table de base de données listant des produits*

ID	Produit_ID	Entrepot	Quantite
Nombre	Nombre	Texte	Nombre

***Tableau 4** – Exemple d'une table de base de données listant le niveau de stock d'un produit dans un entrepôt*

Ces tables peuvent être liées entre elles lors d'une requête, car elles ont un élément en commun, l'identifiant d'un produit. En effet, ce dernier est présent dans les deux tables et peut servir de lien entre elles. Imaginons que nous avons un entrepôt à Lausanne et que nous souhaitons voir tous les produits qui y sont présents en quantité supérieure à 5. La requête se construirait de la manière suivante :

- Le "SELECT" permet de spécifier quels champs nous aimerions avoir dans les résultats.
- Le "FROM" indique la première table que l'on souhaite utiliser.
- Le "INNER JOIN" indique vers quelle autre table on souhaite avoir un lien.
- Le "ON" détermine l'élément commun entre les deux tables.
- Le "WHERE" permet de donner des consignes pour filtrer les résultats.

Le langage SQL (Structured Query Language) est généralement utilisé lors de l'exploitation de bases de données relationnelles.

```
SELECT
    Produits.Nom,
    Produits.Marque,
    Produits.Categorie,
    Produits.Prix,
    Stocks.Quantite
FROM
    Produits
INNER JOIN
    Stocks
ON
    Produits.ID = Stocks.Produit_ID
WHERE
    Stocks.Entrepot = "Lausanne"
    AND Stock.Quantite > 5
```

***Listing 27** - Exemple de requête SQL récupérant une liste de produits en stock dans un entrepôt particulier*

Le résultat de la requête donnera un tableau avec les champs demandés et les éléments trouvés correspondant à nos critères, s'il y en a.

Nom	Marque	Categorie	Prix	Quantite
…	…	…	…	…

***Tableau 5** – Structure du résultat de la requête SQL d'exemple*

Pour obtenir des produits qui peuvent être retirés en succursales, qui ont un minimum de 20 pièces en stock et qui font partie des meilleures ventes, il a fallu lier trois tables entre elles.

Bien que des résultats aient pu être obtenus depuis la base de données, il n'y a toujours pas de moyen de les intégrer dans la solution à l'heure actuelle. Comme expliqué à plusieurs reprises, certaines tâches de ce projet ont pris beaucoup plus de temps que prévu et les problèmes rencontrés tout au long n'ont pas arrangé les choses, ce qui fait que ce sujet n'a pas pu être expérimenté davantage. Ce dernier méritera un approfondissement futur si le projet est poursuivi au-delà de ce travail de Bachelor.

3.1.3 État du développement

Comme vu dans les deux précédents points, la fonctionnalité de génération des produits est donc en suspens. Plusieurs pistes ont été explorées, mais elles méritent plus de temps pour être vraiment abouties et il reste encore de nombreux problèmes à résoudre. Du temps et des ressources supplémentaires pourraient permettre d'aller au bout de cette thématique, mais cela ne se fera malheureusement pas dans le cadre de ce travail de bachelor.

3.2 Fonctionnalité d'enregistrement auto. des produits dans l'ERP

Malheureusement, la fonctionnalité d'enregistrement automatique des produits dans l'ERP n'a pas dépassé le stade de la réflexion. Toutes les raisons exprimées jusqu'à présent dans ce rapport à propos de la gestion du temps et des problèmes rencontrés en sont la cause.

3.2.1 Recherches sur le sujet

Le défi ici était de réussir à accéder à l'ERP de manière déportée. Il faut que l'utilisateur n'ait qu'un bouton à cliquer pour que les produits s'enregistrent automatiquement dans l'ERP. Mais avant d'arriver à ce point, il faut déjà accéder à la plateforme et réussir à s'y connecter. Un compte Microsoft de l'entreprise est requis et la double authentification est par défaut active.

Dans le script développé en Python par un ancien employé, cette tâche a été rendue possible, mais il y a une subtilité. Ce script ouvre également un navigateur visible par l'utilisateur, qui peut alors remplir ses informations de connexion et procéder à la vérification à doubles facteurs. De plus, le script est exécuté depuis l'ordinateur de l'utilisateur.

Dans notre cas de figure, le navigateur est ouvert mais non visible par l'utilisateur. Comme la solution était vouée à être hébergée en tant que site web pour garantir un accès facile, si un navigateur s'ouvre, cela se fera sur le serveur où le site est hébergé. Par conséquent, l'utilisateur ne le verra pas. Il faudrait donc analyser la page de connexion de l'ERP pour demander à l'utilisateur, depuis l'interface de la solution, de fournir les informations nécessaires. Ces dernières seraient alors utilisées lors de l'automatisation des tâches dans le navigateur avec Puppeteer. En effet, avec Puppeteer, il est possible de remplir des formulaires, de cliquer sur des boutons, etc.

Il restera cependant le problème de la double authentification, qui peut se faire de plusieurs manières. Il faudrait donc décortiquer chaque méthode pour la reconnaître et informer l'utilisateur de ce dont la solution a besoin pour valider la connexion.

Malheureusement, ce sujet n'a pas été approfondi davantage et n'a donc pas pu bénéficier de tests pour vérifier la faisabilité réelle de cette fonctionnalité. En théorie, tout cela est possible, de la connexion à l'enregistrement des produits, car cela a déjà été fait. Il faudra toutefois trouver une façon d'adapter cette solution à notre cas de figure. Cela pourra potentiellement être fait si le travail est poursuivi après la fin du travail de Bachelor.

3.3 Fonctionnalités annexes et tests unitaires

Les autres fonctionnalités prévues, comme le tableau de bord, n'ont pas pu être développées. Elles restent toutefois visibles et testables dans une certaine mesure grâce à la maquette sur Figma[3]. Il en va de même pour les tests unitaires. Étant donné que la fonctionnalité principale n'a pas vraiment abouti, il n'était pas pertinent d'inclure des fonctions dédiées à ce sujet.

Le développement est actuellement séparé en deux parties : d'une part, l'interface de test avec la récupération des produits et des catégories, et d'autre part, l'interface et les pages de la solution. On pourrait ainsi considérer l'interface de tests comme une forme de "tests unitaires", car elle permet de tester les fonctionnalités qui auraient dû être intégrées à l'interface finale.

[3] Vers le prototype sur Figma

En conclusion, bien que certaines fonctionnalités n'aient pas pu être entièrement développées, les bases et les prototypes établis offrent une bonne fondation pour de futurs travaux et améliorations du projet.

3.4 Point de situation sur la faisabilité du projet

Un point de situation a été fait avec le mandant quant à la faisabilité du projet. Au milieu du projet, on pouvait être assez confiant, car il restait suffisamment de temps pour développer au moins la fonctionnalité principale jusqu'au bout. Mais avec les tests utilisateurs et l'amélioration de la maquette qui ont nécessité un temps plus long qu'estimé à l'origine, le temps dédié au développement s'est vu grandement réduit. Sans oublier les problèmes rencontrés jusqu'à présent et le fait que l'accès aux bases de données est arrivé tardivement dans le projet.

Il a été convenu qu'il fallait se concentrer sur ce qui pouvait être le plus abouti. Dans notre cas, il s'agit de la maquette qui donne une bonne idée de ce que ce genre de solution peut ressembler. Bien qu'elle soit avant tout un élément visuel, il est possible de tester les diverses fonctionnalités à travers cette dernière, car un travail considérable a été fait pour ajouter des interactions entre les éléments graphiques afin de rendre cette maquette interactive.

Comme envisagé dans la gestion des risques, en cas de manque de temps ou de difficulté, il fallait se concentrer sur la fonctionnalité principale. C'est ce qui a été fait dans un premier temps, mais le temps avançant rapidement, la priorité a été donnée à la maquette afin d'avoir un rendu le plus fidèle et complet à l'idée de base.

4. Phase 3 - Intégration des fonctionnalités

En parallèle du développement des fonctionnalités principales, la mise en chantier de l'interface, de la structure des pages et de la structure des données a débuté pour pouvoir, à terme, accueillir ces fonctionnalités. L'interface ayant été développée après la première version de la maquette et en parallèle des tests utilisateurs, elle ne bénéficie pas des améliorations apportées à la maquette après les tests.

4.1 Développement de l'interface

Cette partie du projet est disponible dans un repository GitHub à l'adresse : https://github.com/nicomeuwly/dg-targetstock-frontend. Il contient également une marche à suivre pour tester le code localement sur sa propre machine. De plus, ce code est également annexé au présent rapport via la clé USB fournie.

4.1.1 Initialisation du projet Next.js

Afin de bien séparer la partie "back end" comprenant les fonctionnalités liées au web scraping et la partie "front end" dédiée à l'interface et ses interactions, un deuxième projet Next.js a été initialisé. Une autre raison pour laquelle ces parties ont été séparées est qu'elles ont été développées en partie simultanément. Cela permet de préparer individuellement les parties jusqu'à ce qu'elles soient suffisamment abouties pour être regroupées.

Nous n'allons pas passer en revue toute la configuration de l'application, car elle est similaire à la partie de test décrite au début de ce rapport. Nous pouvons cependant passer rapidement en revue quelques librairies installées dans le cadre de cette partie "front end".

1. **TailwindCSS** : Cette librairie CSS permet de gagner en efficacité lors de la mise en forme des éléments HTML en écrivant sur une seule ligne tous nos styles.
2. **Prisma** : Cet ORM (Object-Relational Mapping) sert d'interface entre l'application et la base de données et simplifie grandement les requêtes faites à cette dernière. Un des avantages est qu'il n'est pas nécessaire de connaître parfaitement son SGBD pour pouvoir l'utiliser, car Prisma traduit notre code en requête adressable à la base de données.
3. **Sqlite3** : Il s'agit du système de gestion de base de données (SGBD) utilisé durant cette partie de développement. Cela permet notamment de l'utiliser avec Prisma.

4. **React-tooltip** : Cette librairie permet de créer rapidement et simplement des infobulles personnalisables au survol d'éléments. Il aurait été tout à fait envisageable de créer soi-même ce composant, mais ce dernier est particulièrement long et un peu complexe à mettre en place de A à Z. C'est pourquoi, dans l'optique d'optimiser le développement, cette fonctionnalité a été déléguée à une librairie spécialisée.

4.1.2 Navigation dans l'application

Comme expliqué au début du rapport lors de la description du fonctionnement de Next.js, la navigation dans ce framework fonctionne avec le système de fichiers. Ainsi, tous les dossiers présents dans le répertoire "app" qui possèdent un fichier "page.tsx" en leur sein représente une page du site. La capture d'écran ci-dessous représente l'arborescence des fichiers au sein du dossier "src".

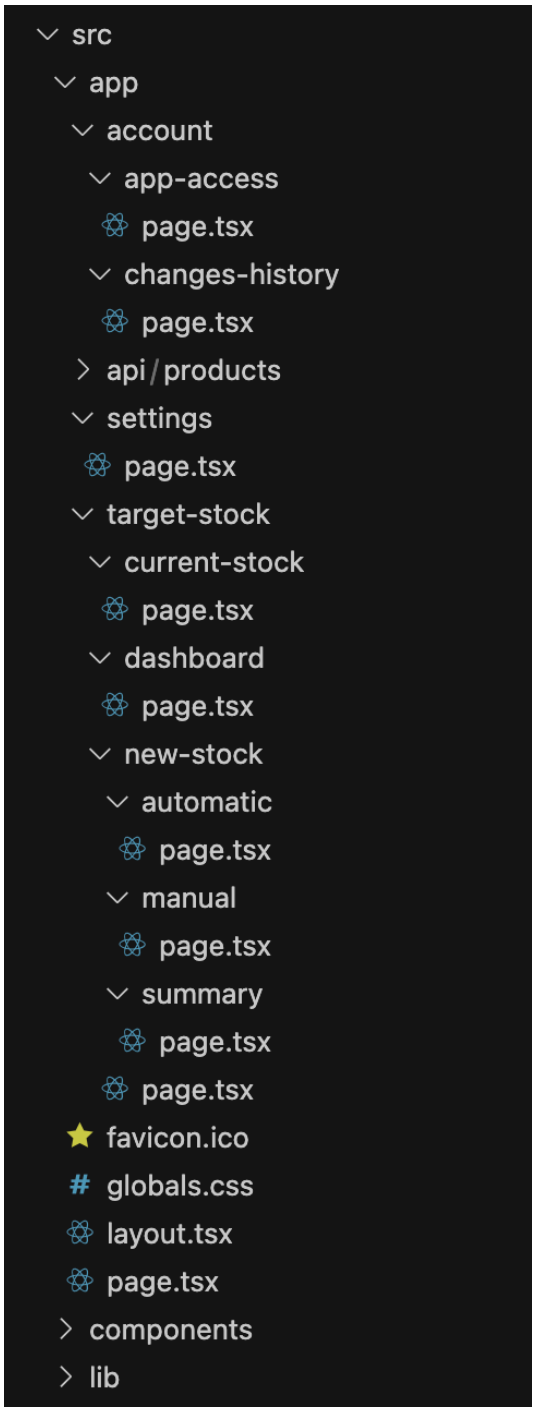


Figure 111 *- Capture d'écran de l'arborescence des fichiers pour les pages du site*

Dans l'état actuel, l'application liste les pages ci-dessous avec leur chemin d'accès respectif. L'URL représente ce qu'il y a après l'adresse de base du site. Lors du développement, il s'agit de "localhost:3000" mais en production, l'adresse pourrait être par exemple "targetstock.digitec.ch" ou "dg-targetstock.ch".

Page	URL
Accueil	/
Tableau de bord	/target-stock/dashboard
Nouveau stock (choix du mode)	/target-stock/new-stock
Mode automatique du nouveau stock	/target-stock/new-stock/automatic
Mode manuel du nouveau stock	/target-stock/new-stock/manual
Résumé des informations avant enregistrement du nouveau stock	/target-stock/new-stock/summary
Stock actuel	/target-stock/current-stock
Réglages de l'application	/settings
Gestion des accès	/account/app-access
Historique des modifications	/account/changes-history

***Tableau 6** - Liste des pages et de leur URL*

Il y a donc 10 pages actuellement sur le site. Le dossier "api" n'a pas été évoqué, car ce dernier ne contient pas de fichier "page.tsx", car il a un autre rôle. C'est à travers ce dossier que se trouveront les différentes routes en charge des opérations avec la base de données.

Il manque encore la page d'enregistrement des produits dans le stock cible. Cela est dû au fait que l'attention a été redirigée vers l'amélioration de la maquette et la tentative d'aboutir à la fonction de génération de produits.

Il y aurait encore d'autres pages à prévoir lorsque l'on compare cette version développée à la version améliorée et définitive de la maquette. Malheureusement, la suite du développement n'a pas pu se faire pour les mêmes raisons évoquées précédemment dans ce rapport.

4.1.3 Composants

L'application se découpe en une multitude de composants servant tous des buts différents. Cette façon de procéder permet de bien séparer le code dans différents fichiers et surtout de pouvoir réutiliser ce code là où cela est utile. Les composants ont été pensés pour être suffisamment flexibles pour maximiser leur utilité.

Les composants se trouvent tous dans un dossier dédié au sein du répertoire "src" nommé "components".

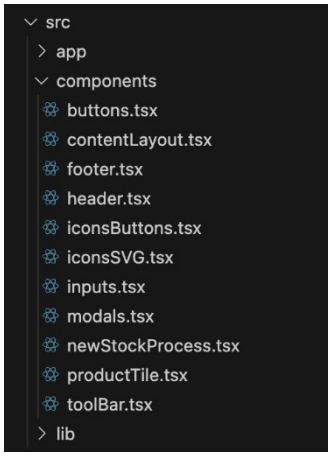


***Figure 112** - Capture d'écran de l'arborescence des fichiers pour les composants du site*

Sans rentrer trop dans les détails techniques, voici en quoi consiste chaque composant développé pour la solution du stock cible. Chaque composant utilise TailwindCSS pour son style et ses différents changements d'état comme l'évènement de survol, par exemple.

Buttons : Ce composant regroupe les différents boutons de l'application, principalement des boutons de grande taille contenant du texte. Il comprend trois types de boutons : le bouton principal, le bouton secondaire et le bouton en tuile.

Le bouton principal et le bouton secondaire doivent recevoir un "label" (le texte affiché sur le bouton), une fonction à exécuter au clic pour définir l'action à réaliser, et un état indiquant si le bouton est désactivé ou non. En revanche, le bouton en tuile, bien qu'il nécessite également un "label", a des prérequis différents. En plus du "label", il doit recevoir une courte description et un lien, car ces boutons ne déclenchent pas une fonction mais servent de lien vers une autre partie de l'application.

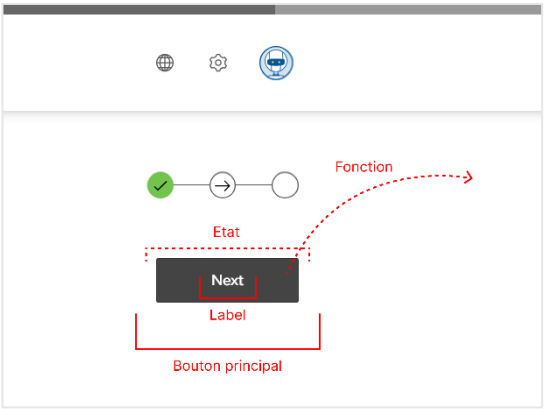


Figure 115 *- Représentation du composant de bouton principal*

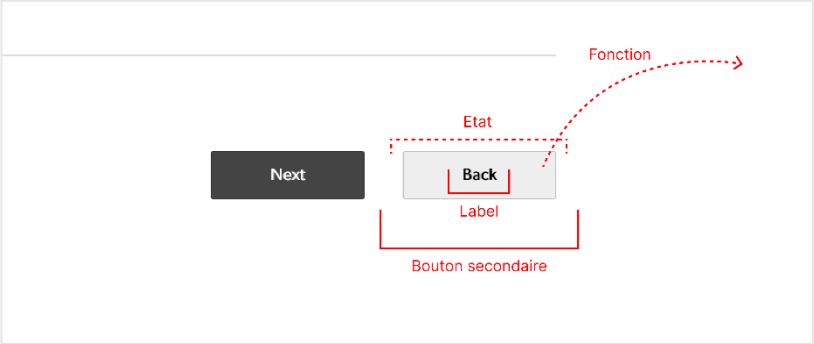


Figure 114 *- Représentation du composant de bouton secondaire*

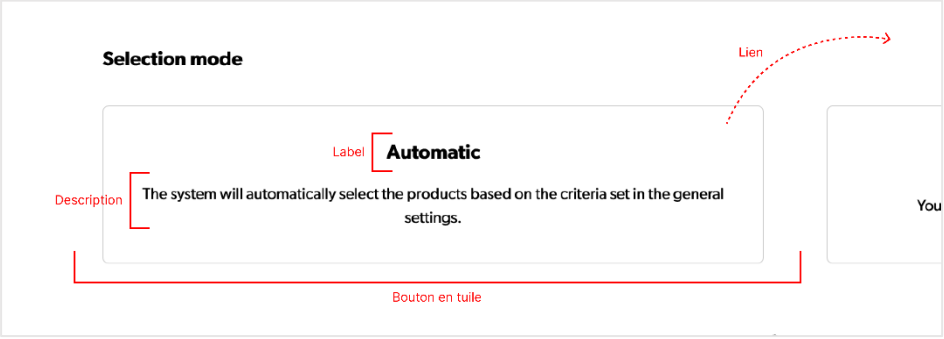


Figure 113 *- Représentation du composant de bouton en tuile*

ContentLayout : Ce composant permet de donner une structure cohérente à toutes les pages, ce qui permet de se concentrer uniquement sur la mise en page du contenu proprement dit. La disposition d'une page comporte des paramètres obligatoires et facultatifs. Parmi les paramètres obligatoires, il y a le titre principal et le contenu de la page. Les paramètres facultatifs sont inclus pour couvrir un maximum de cas de figure différents. En facultatif, il y a donc un indicateur de processus, une description, un bouton (qui est un composant réutilisable vu précédemment), et un titre secondaire.

Cela permet de s'adapter aussi bien à une page complète, comme celle où une liste de produits est affichée lors de la création d'un nouveau stock, qu'à une page plus simple comme celle du tableau de bord avec juste un titre et des informations clés. Cette flexibilité facilite la maintenance et l'évolution de l'application en permettant de réutiliser et d'adapter des composants standards à différents contextes.

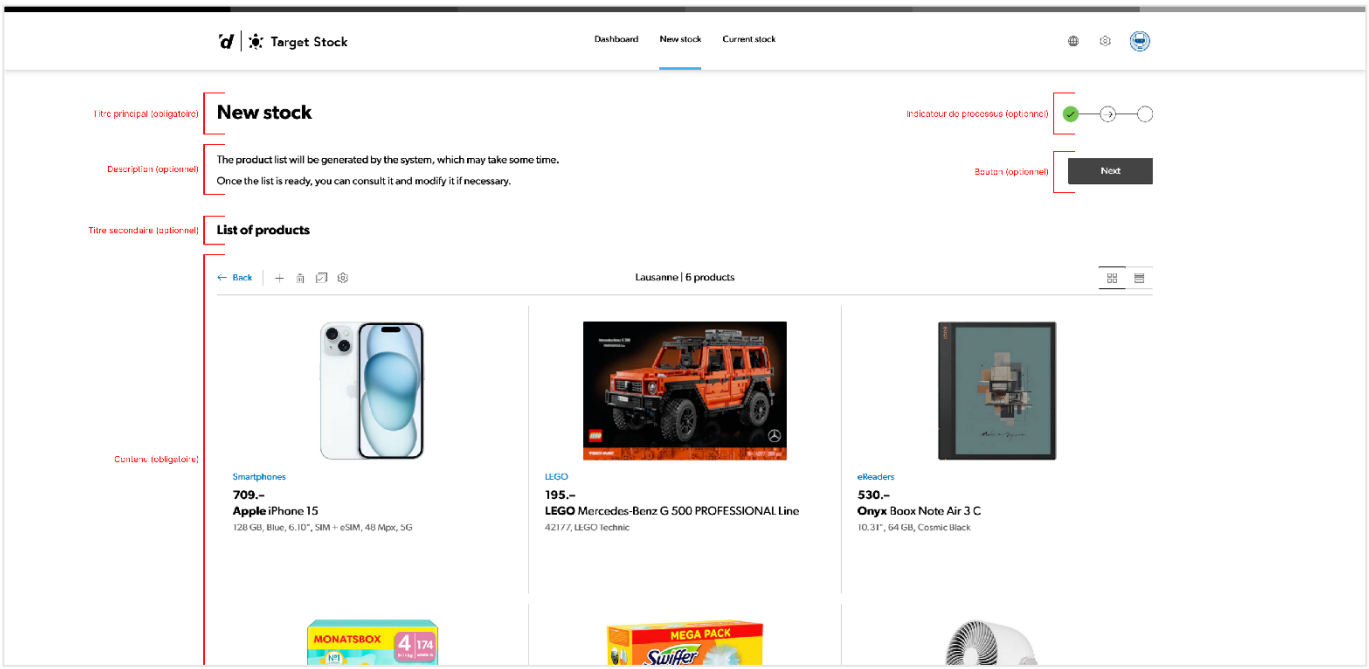


Figure 116 *- Représentation du composant de structure du contenu sur les pages*

Footer : Il s'agit du pied de page du site. Il est présent sur toutes les pages grâce au "layout" de l'application. Un "layout" est un style général qui englobe toute l'application et permet d'avoir des éléments constants sur toutes les pages sans devoir les insérer à chaque fois dans chacune d'entre elles. Le pied de page n'a pas de paramètres, car il n'est pas variable.

Dans la version actuelle, il comprend le nom de la succursale actuelle et la fonction permettant de changer le thème de l'application (clair ou sombre). Cela assure une cohérence visuelle et fonctionnelle à travers toutes les pages de l'application, offrant aux utilisateurs une expérience uniforme et intuitive.

Header : L'entête du site, tout comme le pied de page, ne prend pas de paramètres, car il reste invariable étant donné qu'il est inclus dans le "layout". Cependant, il contient plus de code, car il regroupe davantage de fonctions que le pied de page.

Tout d'abord, on trouve le titre, qui sert également de lien vers la page d'accueil. Ensuite, le système de navigation permet d'accéder aux trois grandes parties de l'application. En outre, l'entête inclut les options de langues et de localisation, les paramètres généraux, ainsi que des liens vers les pages de gestion des accès à l'application et de l'historique des modifications, accessibles via l'icône de l'utilisateur.

Au sein de l'entête se trouvent également d'autres composants, comme les boutons d'icônes, qui ajoutent à la fonctionnalité et à l'interactivité de la barre de navigation. Cette disposition facilite une navigation fluide et intuitive pour les utilisateurs, leur permettant d'accéder rapidement aux différentes sections et réglages de l'application.

IconsButtons : Ce fichier regroupe les différents boutons en forme d'icônes de la solution, qui se comptent au nombre de trois.

Le premier type est le bouton d'icône principale, de couleur gris foncé, qui accepte plusieurs paramètres obligatoires et facultatifs. Parmi les paramètres obligatoires, on trouve le nom de l'icône souhaitée, avec une liste disponible dans le fichier "iconsSVG.tsx", ainsi que l'état du bouton, indiquant s'il est désactivé ou non. Les paramètres facultatifs incluent la possibilité de passer une fonction exécutable lors du clic sur le bouton et un message d'info-bulle qui apparaît au survol du bouton.

Le deuxième type, le bouton d'icône bleu, prend les mêmes paramètres que le bouton principal, mais avec l'ajout d'un texte facultatif à placer après l'icône. Ce type de bouton est notamment utilisé pour les boutons de retour en arrière.

Enfin, le bouton d'icône du compte utilisateur, un peu à part, est situé tout à droite dans l'entête. Ce bouton permet d'ouvrir le menu utilisateur, qui contient les liens menant aux pages de gestion des accès et de l'historique des modifications. Il accepte les mêmes paramètres que le bouton principal, mais ne les utilise pas tous en pratique.

Figure 117 *- Représentation des composants de bouton d'icône principal et secondaire avec texte*

Figure 118 *- Représentation du composant de bouton d'icône secondaire sans texte*

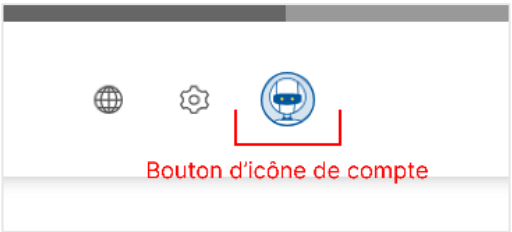


Figure 119 *- Représentation du composant de bouton d'icône de compte*

IconsSVG : Comme évoqué au point précédent, ce composant contient toutes les icônes utilisables par les boutons d'icônes. Chaque composant contenu dans le fichier représente un élément SVG spécifique, exporté de Figma en tant que fichier SVG. En ouvrant ces fichiers avec un éditeur de code, on peut récupérer le code HTML de l'icône, lequel est ensuite intégré dans chaque composant d'IconsSVG. La majorité de ces icônes demandent un paramètre, qui est la couleur à utiliser pour l'icône.

```
export function AddIcon({ colorVar }: IconProps) {
    return (
        <svg
            width="16"
            height="16"
            viewBox="0 0 16 16"
            fill="none"
            xmlns="http://www.w3.org/2000/svg"
        >
            <path
                fillRule="evenodd"
                clipRule="evenodd"
                d="M8 2V7.999H2V8.999H8V15H9V8.999H14.999V7.999H9V2H8Z"
                fill={`var(--${colorVar})`}
            />
        </svg>
    );
}
```

Listing 28 *- Code d'une icône SVG dans le composant IconsSVG*

Inputs : Ce fichier regroupe tous les composants de saisie par l'utilisateur. Pour l'instant, il ne contient que les listes déroulantes, mais à l'avenir, il inclura également des champs de formulaire. La liste déroulante demande trois paramètres essentiels. Premièrement, un tableau d'options représentant les éléments sélectionnables dans la liste. Deuxièmement, un texte affiché au-dessus de la liste, indiquant ce que permet de sélectionner la liste. Enfin, une fonction est nécessaire pour récupérer l'élément sélectionné par l'utilisateur afin de le transmettre à la page ou au composant où la liste est utilisée. Toute la logique permettant d'ouvrir et de fermer la liste est également incluse.

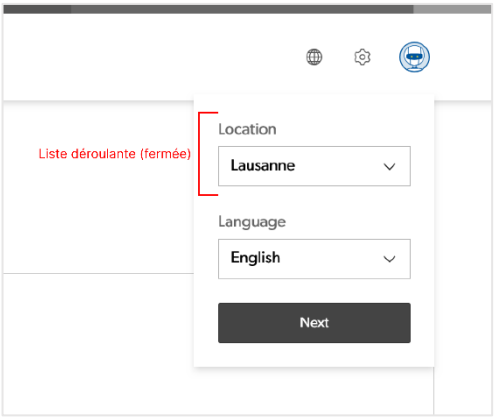


Figure 121 *- Représentation du composant de liste déroulante (fermée)*

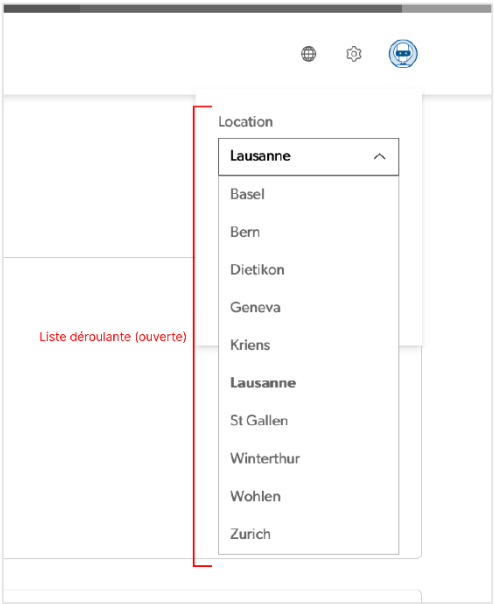


Figure 120 *- Représentation du composant de liste déroulante (ouverte)*

Modals : Les “modals” sont des éléments qui s’ouvrent par-dessus d’autres éléments, à la manière des fenêtres sur un ordinateur. Actuellement, ce fichier ne contient que deux composants, mais il est prévu d’inclure toutes les fenêtres contextuelles, comme celles des réglages du stock cible ou de l’ajout de produits.

L’un des deux composants est le menu de sélection de la succursale et de la langue. Ce menu intègre deux listes déroulantes provenant du fichier “inputs.tsx” ainsi qu’un bouton principal issu du fichier des boutons classiques. Ce menu nécessite un seul paramètre, une fonction permettant de gérer l’état du menu pour déterminer s’il est ouvert ou non.

Le deuxième composant est le menu utilisateur, qui contient différents liens et le bouton de déconnexion. Bien qu’il n’utilise pas d’autres composants, il demande le même paramètre que le premier menu, à savoir la gestion de son état d’ouverture.

Ces modals sont conçues pour offrir une interface utilisateur intuitive et interactive, facilitant la navigation et l’interaction avec les différentes fonctionnalités de l’application.

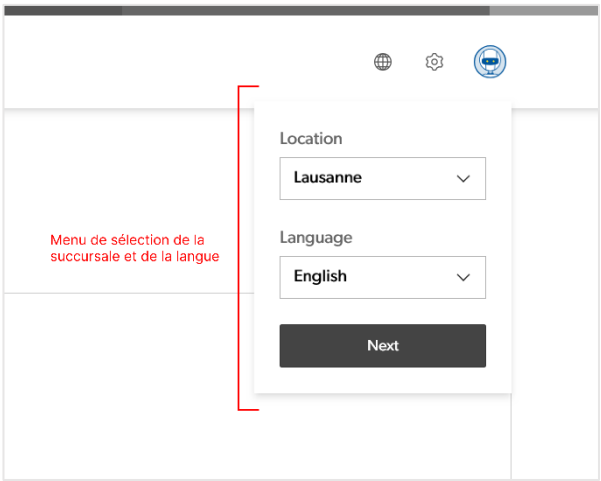


Figure 122 *- Représentation du composant de menu de sélection de la succursale et de la langue*

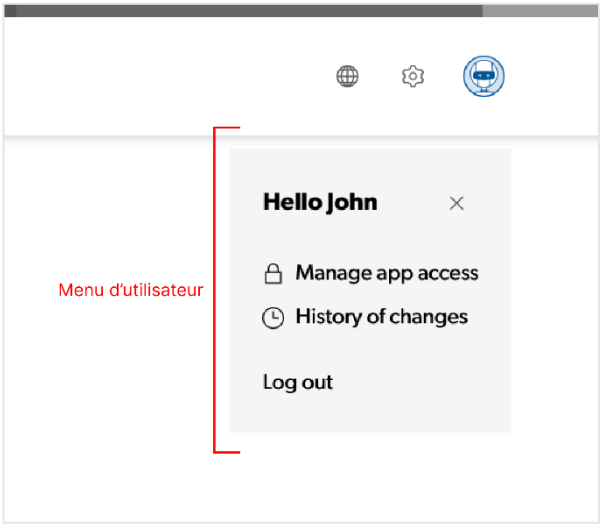


Figure 123 *- Représentation du composant de menu d’utilisateur*

NewStockProcess : Ce petit composant est utilisé sur toutes les pages en lien avec la définition d’un nouveau stock cible. Il s’agit de l’indicateur de processus situé en haut à droite de chacune de ces pages. Il permet de situer l’utilisateur dans le processus en lui indiquant à quelle étape il se trouve et d’afficher le statut des étapes passées, présentes et futures.

Le composant est conçu pour fonctionner avec trois étapes et nécessite un tableau de nombres en paramètres pour déterminer le statut de chaque étape. Le tableau doit contenir trois chiffres, chaque chiffre représentant le statut d'une étape spécifique (par exemple, 1 pour la première étape, 2 pour la deuxième, etc.). Chaque chiffre correspond à l'index d'une icône contenu dans un tableau au sein du composant. Les icônes utilisées sont celles référencées dans le fichier "iconsSVG.tsx".

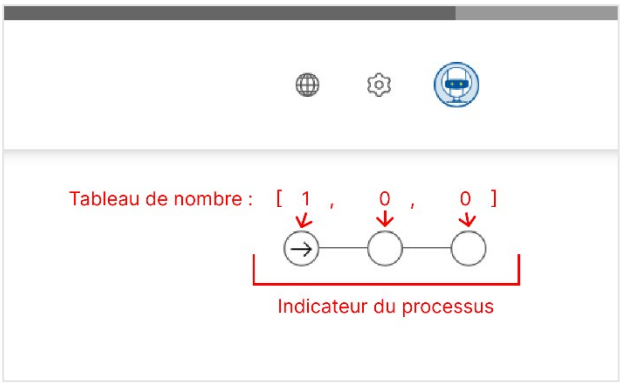


Figure 124 *- Représentation du composant de l'indicateur de processus*

```
const steps = [
    <FutureStepIcon />,
    <CurrentStepIcon />,
    <DoneStepIcon />,
    <PastStepIcon />,
    <ErrorStepIcon />,
    <WarningStepIcon />,
];
```

Listing 29 *- Code du tableau d'icônes utilisé pour l'indicateur de processus*

ProductTile : Il s'agit d'un des composants les plus essentiels à la solution, régissant la mise en page d'un produit. C'est lui qui met en forme les différents éléments qui composent la tuile de produit, tel que le nom, la marque, le prix et l'image. Il implémente également les différentes fonctions présentes au survol du produit qui, pour l'instant, sont juste des boutons d'icône sans fonction rattachée.

Ce composant a besoin en premier lieu d'un objet "product" contenant toutes les informations d'un produit. Il est polyvalent, s'adaptant à la mise en page de la liste de produits, que ce soit la vue sous forme de case ou la vue en liste. Pour cela, il a besoin de recevoir le nom du type de vue utilisé. Enfin, il doit savoir si le produit est le premier ou le dernier de sa rangée pour ajouter ou non une bordure à gauche et à droite de la tuile de produit, ce qui est utilisé pour le type de vue en cases.

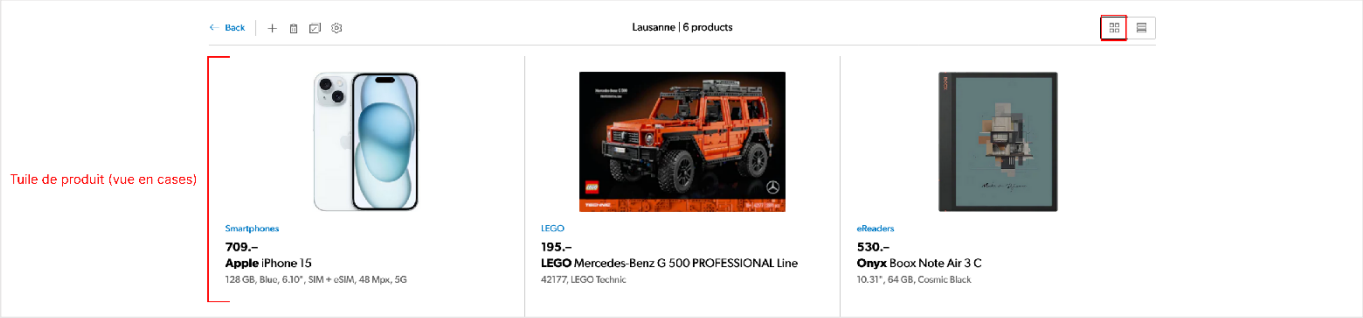


Figure 125 *- Représentation du composant de tuile de produit (vue en cases)*

Figure 126 *- Représentation du composant de tuile de produit (vue en liste)*

Toolbar : La barre d'outils est présente sur toutes les pages présentant une liste de produits. Elle permet de réaliser diverses actions vis-à-vis de cette liste, comme ajouter un produit ou changer la disposition des produits. Pour fonctionner, elle a besoin de plusieurs paramètres. Tout d'abord, il faut indiquer si quelque chose est en train de charger afin d'afficher le message correspondant au milieu de la barre d'outils, ce qui est utilisé lors de la génération automatique de produits. Ensuite, un nombre représentant le total de produits présents sur la page est requis, ce nombre étant également affiché au milieu de la barre.

Une fonction est nécessaire pour transmettre le type de vue utilisé à la page, afin qu'elle puisse à son tour le transmettre aux tuiles de produits. Enfin, deux paramètres sont optionnels : un statut indiquant si les boutons doivent être désactivés ou non, et une instruction sur l'utilisation ou non d'un bouton permettant le retour en arrière. Le composant utilise différents boutons d'icônes issus du composant du même nom. Pour l'instant, seuls les éléments visuels ont été implémentés, et aucune fonction réelle ne fonctionne, à l'exception du changement du type de vue.

Figure 127 *- Représentation du composant de la barre d'outils*

4.1.4 Fonctionnalités

Le développement étant encore à ses balbutiements, la majorité des fonctionnalités visibles sur le site ne sont pas encore actives. L'aspect visuel est prêt, notamment avec les différents changements d'état lors du survol, mais les fonctions ne sont pas encore rattachées aux différents boutons. Bien que le site ait surtout été préparé visuellement, quelques fonctionnalités ont été implémentées. Voici la liste des fonctionnalités actuellement disponibles :

Navigation : Il est possible de naviguer dans le site via les liens au centre de l'entête pour accéder aux pages principales du site : le tableau de bord, la création d'un nouveau stock et la liste du stock actuel. Cliquer sur le logo ramène à la page d'accueil, actuellement vide. L'icône d'engrenage à droite de l'entête mène aux paramètres généraux de la solution, et l'icône de compte ouvre un menu avec des liens vers la gestion des accès et l'historique des modifications, ainsi que la fonction de déconnexion. Le pied de page possède également un lien fonctionnel à gauche permettant d'accéder directement au stock actuel de la succursale choisie par l'utilisateur. En plus de la navigation dans l'entête et le pied de page, il est possible de parcourir le mode automatique de création du stock jusqu'à l'étape du résumé des informations, avec la possibilité de revenir en arrière.

Changement de vue : Comme évoqué précédemment, il est possible de changer la façon dont une liste de produits s'affiche. Pour l'instant, cette fonction est disponible uniquement sur la page des produits lors de la génération automatique, car il s'agit de la seule page à afficher des produits dans l'état actuel du développement. Il est ainsi possible de passer du mode d'affichage en cases à celui en liste.

Mode automatique : Afin de pouvoir développer le design des produits et leur disposition, il faut pouvoir les afficher pour voir le rendu de ces derniers sur une page. Pour ce faire, une route API a été créée pour, lors de son appel, recevoir une liste de produits à afficher. Les produits sont directement écrits en dur dans la route, ce qui signifie

qu'ils sont prédéfinis et ne sont pas récupérés dynamiquement, comme cela devrait idéalement être le cas à l'avenir. Il aurait été possible de définir la liste de produits directement dans la page où elle est utilisée, mais cela aurait rendu le code moins modulaire et plus difficile à maintenir. En utilisant une route API, nous préparons le terrain pour une future amélioration où les produits pourront être récupérés dynamiquement à partir d'une base de données ou d'une autre source, permettant ainsi une mise à jour plus flexible et évolutive des produits affichés.

Sauvegarde des préférences : Comme la gestion des utilisateurs n'a pas encore été développée, les préférences de ces derniers sont stockées dans le stockage local du navigateur. Les données sauvegardées incluent la succursale, la langue et le thème préféré. Ainsi, depuis le menu de sélection de la succursale et de la langue, on peut choisir la succursale et la langue de notre choix, et lors du clic sur le bouton "Next", ces informations sont mises à jour dans le stockage local. C'est le même principe pour la sélection du thème depuis le pied de page. En inspectant la page et en se rendant dans l'onglet dédié au stockage, on peut constater que les données sont enregistrées dans le stockage local et liées à l'adresse du site.

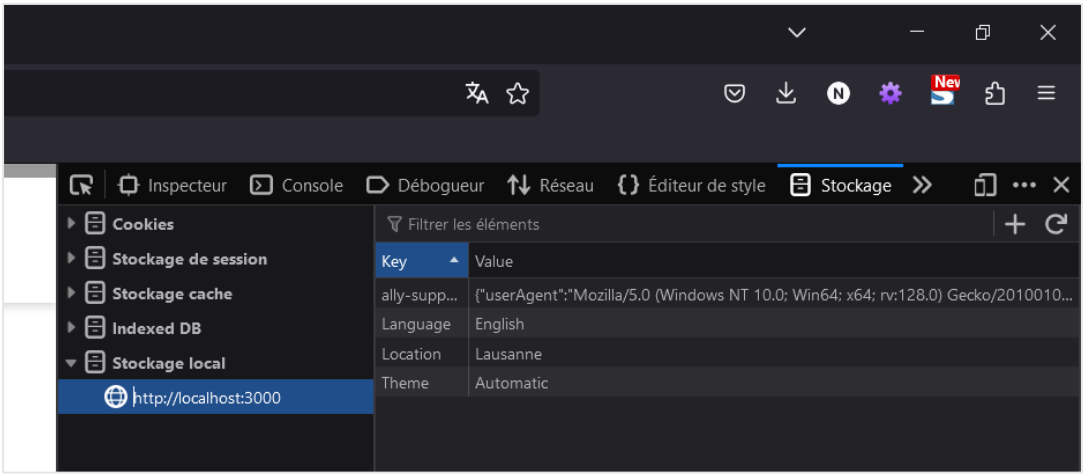


Figure 128 *- Aperçu des données stockées dans le stockage local du navigateur*

4.2 Structure des données

4.2.1 Préparation en amont

Afin d'accueillir les données générées par les principales fonctionnalités, il va falloir concevoir une structure pour ces données. Il est également nécessaire de déterminer ce que l'on va faire de ces données, comment elles vont être manipulées et quels liens il faut établir entre elles.

À partir de la première version de la maquette et des premiers retours utilisateurs, il a été possible de définir le fonctionnement de l'application et la manière dont les données seront structurées. Ainsi, il a été déterminé qu'il y a cinq types de données que l'on souhaite stocker.

1. **Les succursales** : Elles servent de point de référence pour diverses données importantes. Chaque succursale répertorie les listes du stock cible qui y sont disponibles, les utilisateurs qui y travaillent, ainsi que les critères et catégories utilisés pour le mode automatique de génération de produits. Cela permet de ne pas devoir redéfinir ces paramètres à chaque nouvelle génération de stock. Les succursales sont identifiables par leur nom, facilitant ainsi la gestion et l'organisation des informations spécifiques à chaque emplacement.

2. **Les listes du stock cible** : Ces listes contiennent tous les produits, qu'ils aient été ajoutés par le système ou par l'utilisateur. Chaque liste est rattachée à une succursale spécifique et dispose d'une période de disponibilité ainsi que d'un statut, indiquant par exemple si la liste est à l'état de brouillon. Pour une identification facile et une gestion efficace, chaque liste possède également un nom distinctif.

3. **Les produits** : Ils constituent l'essence même du stock cible, répondant aux besoins et désirs des consommateurs, qu'il s'agisse de produits de première nécessité ou de biens saisonniers. Chaque produit est décrit

par plusieurs attributs : une image, une catégorie, une marque, un nom, un prix et un lien vers sa page détaillée. Un statut spécifique est également associé à chaque produit, permettant notamment de repérer les produits “EOL” (End Of Life), qui ne sont plus vendus et doivent être retirés du stock cible.

4. **Les critères de sélection** : Bien qu’il ne soit pas strictement nécessaire de sauvegarder ces critères, ils sont enregistrés dans la base de données pour mémoriser les préférences du mode automatique d’une succursale. Chaque critère possède un nom et un numéro d’identification unique. Ce numéro est essentiel, car il est utilisé dans les paramètres d’URL sur Digitec et Galaxus, facilitant ainsi l’application précise des préférences de sélection de produits.

5. **Les catégories du site** : Comme observé lors de la phase de test, récupérer les catégories du site peut être un processus assez long. Étant donné que ces catégories ne changent pas constamment, il n’est pas nécessaire de les générer à chaque fois. Elles seront donc stockées dans la base de données. Cela permettra également de sauvegarder les préférences des succursales dans le cadre des paramètres du mode automatique. Chaque catégorie comprend un nom, une catégorie principale à laquelle elle est rattachée, une URL et un statut, facilitant ainsi l’organisation et l’accès rapide aux informations pertinentes.

Il y a donc cinq tables principales correspondant aux cinq types de données décrits précédemment, mais il faudra également trois tables de liaison. Ces tables de liaison sont nécessaires lorsqu’une relation entre deux tables est de type "Plusieurs à plusieurs". Par exemple, cela s’applique dans le cas des listes de stock cible et des produits, car un produit peut être dans plusieurs listes et une liste peut avoir plusieurs produits.

Parmi ces tables de liaison, il y a :

- Une table qui relie un produit à une liste de stock cible.
- Une table qui relie un critère à une succursale.
- Une table qui relie une catégorie à une succursale.

En plus de ces relations "Plusieurs à plusieurs", il existe également des relations "Un à plusieurs" entre certaines tables. Par exemple, une succursale peut avoir plusieurs listes de stock cible, mais une liste n’est attribuée qu’à une seule succursale. Dans le cas du stock général attribué à toutes les succursales, cela créera une liste par succursale avec le même nom, les mêmes paramètres et les mêmes produits. Seul l’identifiant de la liste dans la table sera unique.

Ces structures de relations permettent une gestion efficace et flexible des données, facilitant ainsi les opérations de recherche, d’ajout, de modification et de suppression des enregistrements dans la base de données.

Pour représenter visuellement les différentes tables que la base de données va contenir ainsi que les relations entre ces dernières, l'outil en ligne de création de diagrammes de base de données, dbdiagram.io, a été utilisé. Cet outil permet de créer des diagrammes ER (Entité-Relation) de manière simple et intuitive, facilitant ainsi la compréhension des relations et des structures de la base de données.

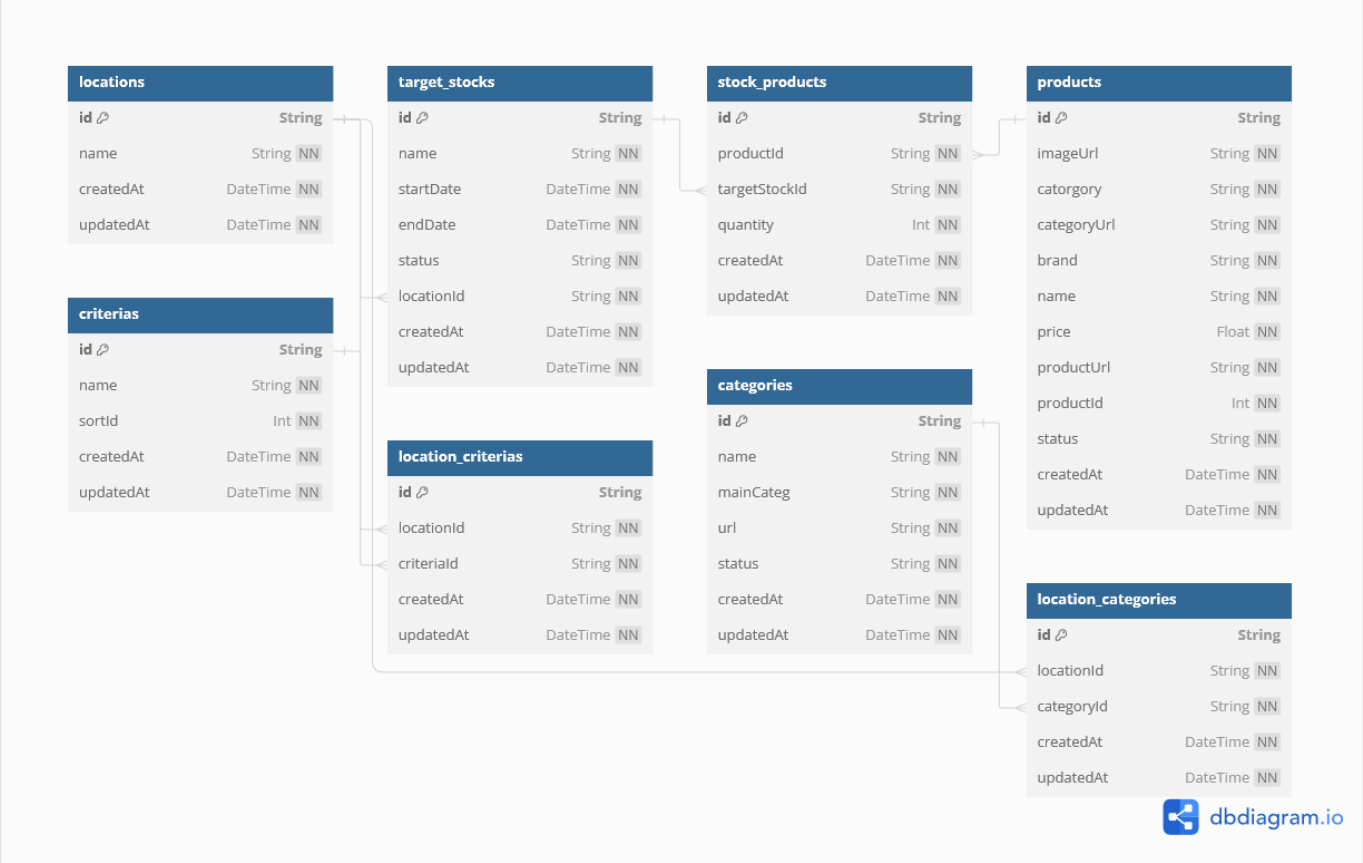


Figure 129 *- Diagramme de la base de données de la solution du stock cible*

L'aspect de la gestion des utilisateurs n'a pas été oublié, mais il a été volontairement mis de côté pour le moment. Étant donné que la question de la connexion et de l'intégration avec l'ERP n'a pas encore été résolue, seules les données liées à la création des listes de stock cible ont été envisagées pour l'instant.

Cela dit, l'architecture de la base de données a été pensée de manière flexible afin de permettre l'inclusion des utilisateurs dans le futur. Si le projet se poursuit, il sera possible d'ajouter des tables et des relations pour gérer les utilisateurs sans avoir à remanier en profondeur la structure existante.

4.2.2 Intégration dans le développement

Il est possible d'utiliser de nombreux systèmes de gestion de bases de données (SGBD) dans notre cas de figure. La seule contrainte est qu'ils permettent d'utiliser des bases de données relationnelles, car c'est ainsi que nos données sont structurées.

Dans un premier temps, SQLite sera utilisé pour contenir et gérer les données de la solution lors du développement. Il s'agit d'une solution légère, et la base de données est directement disponible sous forme de fichier dans les dossiers du projet. Cela permet notamment d'éviter de configurer un serveur local dédié pour MySQL ou PostgreSQL, d'autres SGBD plus performants. Il sera tout à fait possible de passer à l'une de ces solutions une fois le projet plus avancé.

La raison pour laquelle il sera aisé de changer de SGBD est que nous n'allons pas directement communiquer avec ces derniers. Nous allons passer par un ORM (Object-Relational Mapping) qui va s'en charger pour nous. Un ORM

sert d'interface entre la base de données et notre code, rendant l'interaction entre les deux plus simple à comprendre et à prendre en main. L'ORM en question se nomme "Prisma". Il s'agit d'un ORM souvent utilisé avec Next.js et, ayant déjà été utilisé à de nombreuses reprises dans des projets personnels et scolaires, le choix était alors évident. Cela évite de perdre du temps sur une technologie peu connue. La façon de communiquer ne change quasiment pas entre les différents SGBD, c'est pour cette raison qu'il sera aisé de changer en cours de route.

Par exemple, pour récupérer une liste de produits avec Prisma, il suffira du code suivant.

```
const products = await prisma.products.findMany();
```

Listing 30 *- Exemple de code permettant de récupérer tous les produits d'une table avec Prisma*

Prisma va ensuite traduire cela en requête SQL, ce qui donnera le code suivant.

```
SELECT * FROM products
```

Listing 31 *- Exemple de code d'une requête SQL permettant de récupérer tous les produits d'une table*

Cet exemple n'était certes pas très poussé en termes de SQL et reste compréhensible à ce niveau en SQL, mais cela deviendra plus intéressant lors de requêtes plus complexes.

Il faut ainsi commencer par installer Prisma dans le projet. La documentation est d'ailleurs bien faite et permet de rapidement configurer la librairie. Il faudra également installer SQLite, puis le reste pourra suivre. Sans entrer trop dans les détails, Prisma a besoin de deux fichiers : un pour configurer le client qui permettra de communiquer avec la base de données, et un autre pour le schéma décrivant la structure de la base de données. Le code à inclure dans ces deux fichiers est décrit dans la documentation officielle. Nous allons nous concentrer surtout sur le fichier du schéma.

La documentation explique comment décrire une table et ses relations. Ainsi, il a été possible de décrire la structure de nos tables sur lesquelles le diagramme présenté plus haut est basé. Une table est représentée par un "model" et il faut indiquer à l'intérieur les différents champs souhaités ainsi que le type de données qu'ils vont accueillir. Les relations sont gérées avec l'annotation "@relation" en indiquant quels sont les éléments en commun qui lient les deux tables. C'est dans ce fichier également qu'est indiqué le type de base de données utilisée, SQLite dans notre cas. Il ne faut pas oublier de mentionner l'adresse du fichier SQLite dans le fichier ".env" situé à la racine du projet. Ce fichier permet de référencer toutes les informations sensibles à un endroit et ainsi de cacher ces informations dans le code, car on y fait référence simplement via le fichier ".env".

```
// This is your Prisma schema file,
// learn more about it in the docs: https://pris.ly/d/prisma-schema

generator client {
  provider = "prisma-client-js"
}

generator dbml {
  provider = "prisma-dbml-generator"
}

datasource db {
  provider = "sqlite"
  url      = env("DATABASE_URL")
}

model Location {
  id                 String             @id @default(cuid())
  name               String
  targetStocks       TargetStock[]
  locationCriterias  LocationCriteria[]
  locationCategories LocationCategory[]
  createdAt          DateTime           @default(now())
  updatedAt          DateTime           @updatedAt

  @@map("locations")
}

model TargetStock {
  id            String         @id @default(cuid())
  name          String
  startDate     DateTime
  endDate       DateTime
  status        String
  Location      Location       @relation(fields: [locationId], references: [id])
  locationId    String
  stockProducts StockProduct[]
  createdAt     DateTime       @default(now())
  updatedAt     DateTime       @updatedAt

  @@map("target_stocks")
}

model StockProduct {
  id            String      @id @default(cuid())
  product       Product     @relation(fields: [productId], references: [id])
  productId     String
  targetStock   TargetStock @relation(fields: [targetStockId], references: [id])
  targetStockId String
  quantity      Int
  createdAt     DateTime    @default(now())
  updatedAt     DateTime    @updatedAt
```

```
  @@map("stock_products")
}

model Product {
  id            String         @id @default(cuid())
  imageUrl      String
  catorgory     String
  categoryUrl   String
  brand         String
  name          String
  price         Float
  productUrl    String
  productId     Int
  status        String
  stockProducts StockProduct[]
  createdAt     DateTime       @default(now())
  updatedAt     DateTime       @updatedAt

  @@map("products")
}

model Criteria {
  id                String             @id @default(cuid())
  name              String
  sortId            Int
  locationCriterias LocationCriteria[]
  createdAt         DateTime           @default(now())
  updatedAt         DateTime           @updatedAt

  @@map("criterias")
}

model LocationCriteria {
  id         String   @id @default(cuid())
  location   Location @relation(fields: [locationId], references: [id])
  locationId String
  criteria   Criteria @relation(fields: [criteriaId], references: [id])
  criteriaId String
  createdAt  DateTime @default(now())
  updatedAt  DateTime @updatedAt

  @@map("location_criterias")
}

model Category {
  id                 String             @id @default(cuid())
  name               String
  mainCateg          String
  url                String
  status             String
  locationCategories LocationCategory[]
  createdAt          DateTime           @default(now())
  updatedAt          DateTime           @updatedAt
```

```
  @@map("categories")
}

model LocationCategory {
  id         String   @id @default(cuid())
  location   Location @relation(fields: [locationId], references: [id])
  locationId String
  category   Category @relation(fields: [categoryId], references: [id])
  categoryId String
  createdAt  DateTime @default(now())
  updatedAt  DateTime @updatedAt

  @@map("location_categories")
}
```

Listing 32 *- Code du schéma Prisma décrivant la structure de la base de données*

Dans les faits, ce schéma a été créé avant le diagramme en raison de préférences personnelles. Il a été plus simple pour moi de d'abord décrire en texte la base de données puis, grâce à une librairie, prisma-dbml-generator en l'occurrence, générer un fichier au format "dbml". Le contenu de ce dernier a pu ensuite être inséré dans un projet vierge sur dbdiagram.io, qui va interpréter ce fichier pour créer un diagramme de base de données.

Il faut encore créer le fichier de la base de données. Ce fichier peut être placé dans le même dossier que le schéma Prisma pour être facilement retrouvé. Le nom de ce fichier n'est pas particulièrement important, mais il sera nommé "dev.db" dans notre cas pour indiquer qu'il s'agit de la base de données de développement. Une extension de l'éditeur de code peut permettre de consulter le contenu du fichier directement depuis ce dernier, sans avoir à passer par un logiciel tiers. Cela peut être utile pour rapidement voir le contenu de la base de données.

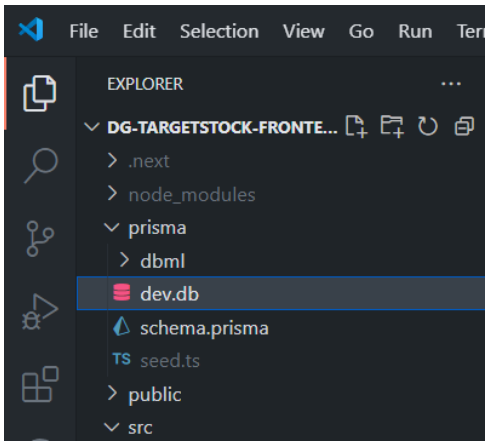


Figure 130 *- Capture d'écran de l'emplacement du fichier de la base de données*

Une fois le schéma et le client Prisma configurés ainsi que le fichier de base de données créé, il suffira d'exécuter deux commandes dans la console pour générer le client et appliquer les instructions du schéma au fichier SQLite.

```
> prisma generate
> npx prisma db push
```

Listing 33 *- Commandes pour utiliser Prisma et configurer le fichier de base de données*

Cela n'est pas obligatoire, mais il est possible de déjà remplir la base de données avec quelques données de base comme les succursales ou encore les critères de sélection. Pour ce faire, un fichier "seeder" est créé. Un "seeder"

peut être utilisé pour remplir les tables de données factices afin de pouvoir rapidement tester des fonctions lors du développement ou, dans notre cas, pour insérer des données qui n'auront pas de fonction dédiée au sein de la solution pour remplir ce rôle. La structure d'un "seeder" peut être trouvée dans la documentation de Prisma et se présente comme suit dans notre cas.

```
import { PrismaClient } from '@prisma/client'
const prisma = new PrismaClient()

async function main() {
    await prisma.location.createMany({
        data: [
            { name: "Basel" },
            { name: "Bern" },
            { name: "Dietikon" },
            { name: "Geneva" },
            { name: "Kriens" },
            { name: "Lausanne" },
            { name: "St Gallen" },
            { name: "Winterthur" },
            { name: "Wohlen" },
            { name: "Zurich" },
        ],
    })
    await prisma.criteria.createMany({
        data: [
            { name: "Top-selling", sortId: 15 },
            { name: "New on Galaxus", sortId: 6 },
            { name: "Rating", sortId: 7 },
        ],
    })
}
main()
    .then(async () => {
        await prisma.$disconnect()
    })
    .catch(async (e) => {
        console.error(e);
        await prisma.$disconnect();
        process.exit(1)
    })
```

Listing 34 *- Code du "seeder" permettant de peupler la base de données*

Le "seeder" peut être exécuté via une commande dans la console, commande qu'il faudra au préalable définir dans le fichier "package.json". Ici aussi, tout peut se trouver dans la documentation. Une fois cela paramétré, il ne nous reste plus qu'à lancer la commande suivante pour remplir notre base de données des données souhaitées.

```
> npx prisma db seed
```

Listing 35 *- Commande permettant d'exécuter le "seeder"*

Tout est prêt à présent pour pouvoir utiliser la base de données dans ce projet. Malheureusement, comme pour d'autres parties de ce travail, il n'a pas été possible d'aller plus loin dans le développement. Le tout reste cependant prêt pour une continuité future.

Cette partie du projet est également disponible dans un repository GitHub à l'adresse suivante : https://github.com/nicomeuwly/dg-targetstock-frontend. Le repository contient également des instructions pour tester le code localement sur votre propre machine. En outre, ce code est également annexé au présent rapport via la clé USB fournie.

5. Phase 4 - Tests et optimisations

Le développement n'ayant pas atteint un niveau suffisant d'avancement, les tests utilisateurs sur le prototype n'ont donc pas eu lieu. De même, la version améliorée de la maquette a été finalisée trop tard pour organiser une nouvelle session de tests. Quelques optimisations ont toutefois pu être réalisées en fin de projet pour corriger le code, notamment pour la fonctionnalité de génération de produits. À plusieurs reprises, les sélecteurs CSS ont changé sur Galaxus, ce qui a causé des erreurs lors des essais de la fonction à différents moments. Il a donc fallu à chaque fois récupérer le sélecteur actuel sur le site pour que cela fonctionne à nouveau.

Malgré le manque de temps pour finaliser cette phase, il est important de garder cette étape en tête pour de futurs projets. Les tests utilisateurs et l'optimisation du code sont essentiels pour garantir la qualité et la performance de la solution finale. Ils permettent de détecter les erreurs potentielles, d'améliorer l'expérience utilisateur et d'assurer que le produit répond aux attentes des utilisateurs. En intégrant systématiquement ces étapes dans le processus de développement, on peut augmenter significativement les chances de succès des projets futurs.

Les tests utilisateurs fournissent des insights précieux sur l'utilisation réelle du produit, permettant de découvrir des problèmes qui ne sont pas forcément visibles en phase de développement. Par exemple, des bugs, des problèmes de performance ou des éléments d'interface utilisateur qui ne sont pas intuitifs peuvent être identifiés et corrigés avant le déploiement. De plus, ces tests peuvent révéler des besoins ou des attentes non anticipés des utilisateurs, permettant d'ajuster la solution pour mieux répondre à ces exigences.

L'optimisation du code, quant à elle, garantit que l'application fonctionne efficacement, même avec un grand nombre d'utilisateurs ou de données. Cela inclut la réduction des temps de chargement, la minimisation des ressources utilisées et l'amélioration de la stabilité globale de l'application. Une application bien optimisée offre non seulement une meilleure expérience utilisateur mais réduit aussi les coûts d'exploitation, que ce soit en termes de ressources serveur ou de maintenance.

En conclusion, même si ces étapes n'ont pas été complètement réalisées dans ce projet, elles restent cruciales pour tout développement futur. En les intégrant dès le début du processus et en leur accordant une attention constante, on s'assure que le produit final sera de haute qualité, performant et répondra parfaitement aux attentes des utilisateurs.

6. Phase 5 - Finalisation et recommandations

Ce travail s'achève avec la rédaction du présent rapport et des documents annexes. Les différents fichiers ne pouvant être transmis via le PDF du rapport ont été préparés pour être inclus dans les clés USB fournies avec les exemplaires du rapport. Cela inclut l'ajout de documentation dans le code, la rédaction des fichiers "README.md" servant de présentation et d'instructions pour l'utilisation du code, ainsi que la préparation du fichier Figma contenant les différentes versions de la maquette et les composants visuels utilisés. L'espace de travail a également été réorganisé pour être plus structuré.

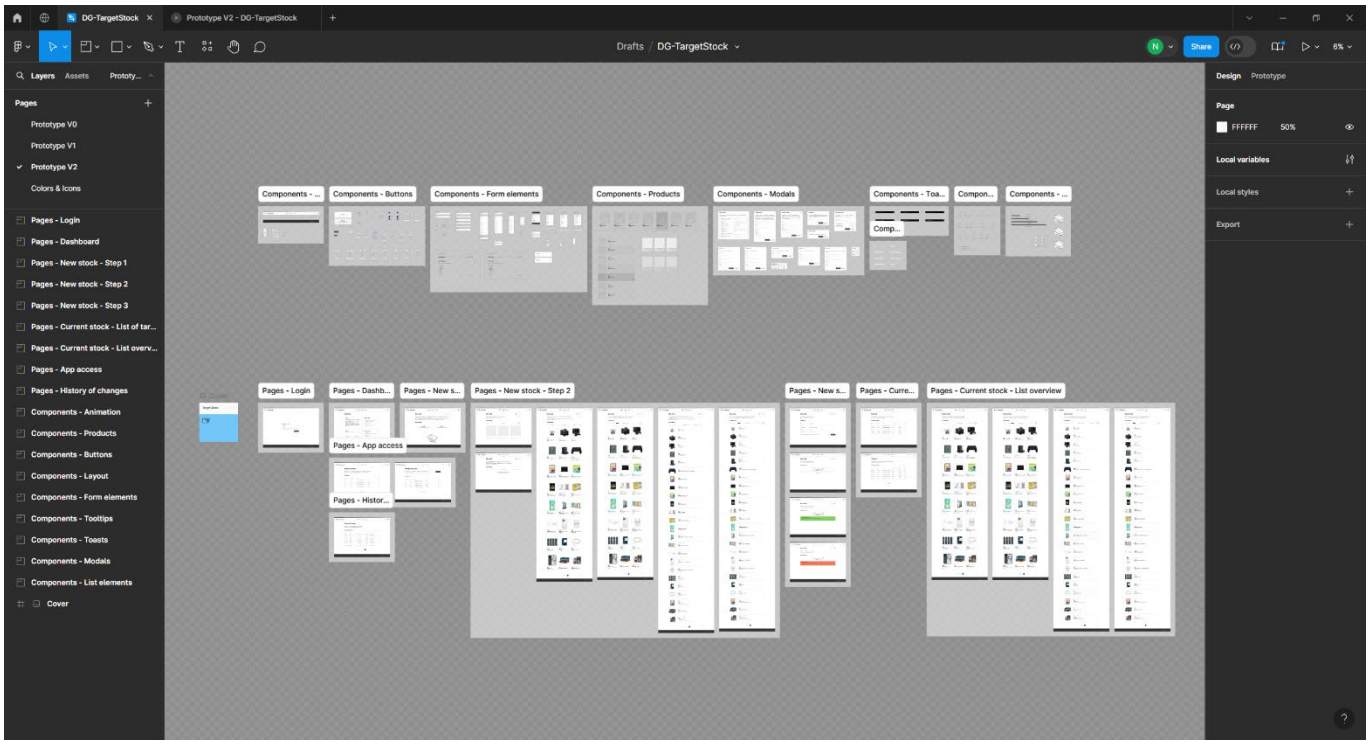


Figure 131 *- Capture d'écran de l'organisation de l'espace de travail sur Figma*

Afin de proposer des pistes de recherche pour aller plus loin dans le projet, des recommandations ont été rédigées et sont incluses dans un document à part entière dans les annexes[4]. Ces recommandations regroupent différentes visions de la suite du projet. Elles sont au nombre de deux, explorant des pistes avec peu de ressources et d'autres avec davantage de ressources, tant humaines qu'en termes d'accès. La première approche se concentre sur l'optimisation des processus actuels en utilisant les ressources existantes et en mettant en œuvre des solutions à court terme pour améliorer l'efficacité. La deuxième approche envisage une expansion plus ambitieuse, incluant l'intégration de nouvelles technologies et l'augmentation des capacités de l'équipe, permettant ainsi une transformation plus radicale et durable du système de gestion du stock cible. Ces recommandations fournissent un cadre pour les développements futurs et aident à orienter les efforts pour maximiser l'impact du projet.

Enfin, un guide d'utilisation a également été rédigé afin de fournir une documentation aux utilisateurs de la solution et se trouve, comme pour les recommandations, dans un document en annexe. Ce guide se base sur la version améliorée de la maquette et pourra être adapté en cas de développement futur de la solution. Il détaille les étapes pour utiliser efficacement la solution, y compris la génération de listes de produits, la gestion du stock cible, et l'enregistrement dans l'ERP. En fournissant des instructions claires et précises, ce guide vise à garantir une transition en douceur pour les utilisateurs et à minimiser les erreurs lors de l'utilisation de la solution. Il servira également de référence précieuse pour toute formation nécessaire et pour l'assistance technique.

[4] Vers les annexes

7. Présentation de la solution

La solution a été présentée au mandant via la maquette améliorée. Les différents points d'amélioration ont pu être démontrés et l'état actuel et définitif du projet a également pu être expliqué. Le mandant est satisfait du résultat final, même si le développement technique n'a pas pu aller très loin.

Comme évoqué plus tôt dans le rapport, il est possible de tester la maquette définitive via le lien Figma suivant : Prototype Figma (version définitive). Cette maquette représente l'état le plus complet de la solution. Il est possible de découvrir toutes les fonctionnalités majeures allant de la génération de produits à la gestion des accès de l'application. Il y a cependant des limitations quant à l'utilisation de ce genre de prototype. Par exemple, il n'est pas possible d'écrire dans un champ de formulaire. De plus, certaines interactions ont été ajoutées pour qu'un composant puisse revenir à son état de base. La façon dont cela se passe est, dans certains cas, spécifique à la maquette et sera différente sur une version développée de la solution.

En plus de la maquette, dont le fichier complet est annexé au rapport via les clés USB fournies, plusieurs dossiers contiennent tout le code créé pour ce projet. Dans l'ordre, il y a l'interface de test avec la première version de la fonctionnalité de génération de produits, une version améliorée de cette dernière en gardant l'interface de test, et enfin la partie "front-end" de l'application avec son interface et la base de données. Parmi les livrables, il y a également les deux documents cités au point précédent, à savoir les recommandations et le guide utilisateur.

8. Gestion du projet

8.1 Planification initiale

Lors de la pré-étude de ce projet, une planification a été établie avec un certain nombre de tâches et une répartition de ces dernières sur les 11 semaines de travail. Cette répartition a été imaginée avec la perception du projet à un instant T, soit avant que cette phase de réalisation ne commence.

Il était prévu de passer 2 semaines sur la conception de la solution avec la première version de la maquette, la mise en place de l'architecture de la solution, l'organisation de tests utilisateurs sur la maquette et enfin des adaptations à la suite des tests.

Il devait y avoir ensuite 4 semaines pour le développement concret des principales fonctionnalités. En parallèle, une première ébauche de l'affiche et du résumé devait être rédigée en vue de la séance du bilan intermédiaire du travail.

Par la suite, 1 semaine était prévue pour mettre en commun les fonctionnalités principales et effectuer les ajustements nécessaires. Une préparation en vue du déploiement de la solution était également prévue durant cette même semaine.

Une 2e session de tests utilisateur était prévue avant la dernière phase du projet. Cette fois-ci, c'est la version développée de la solution qui devait faire l'objet du test. À la suite des tests, diverses corrections et optimisations étaient également au menu dans l'optique d'améliorer la solution. Cette avant-dernière phase était prévue pour se dérouler sur 2 semaines.

Enfin, la dernière phase de ce travail avait pour but, en 1 semaine, de déployer la solution, de la présenter au mandant et de rédiger les recommandations et le guide utilisateur.

En parallèle de toutes ces phases, il était prévu de rédiger au fur et à mesure le présent rapport. Le travail se concluait alors par l'impression et le rendu de ce dernier avec l'échéance au 9 août 2024 à midi.

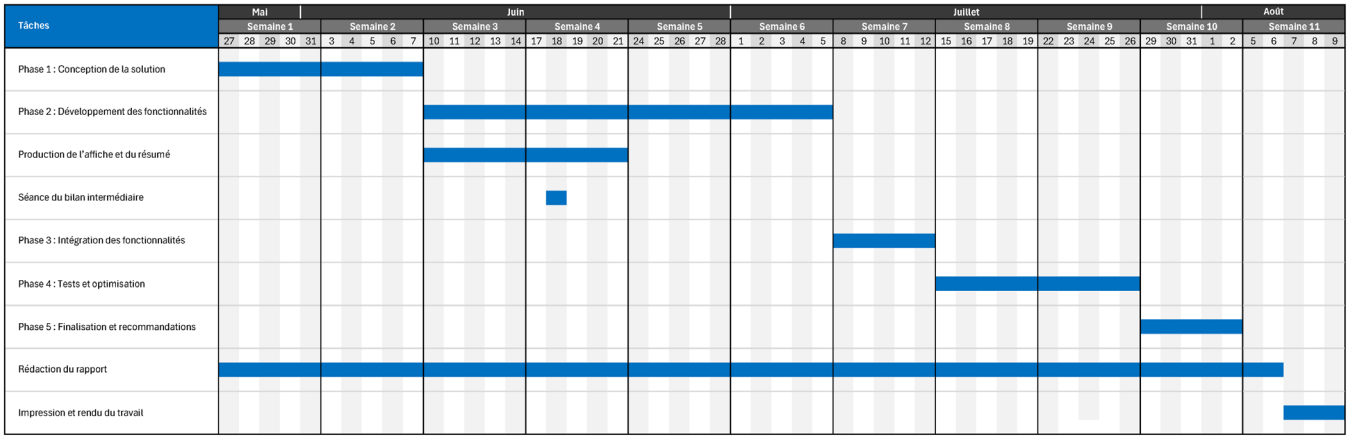


Figure 132 *- Planification initiale du projet au format Gantt Chart*

8.2 Déroulement du travail

8.2.1 Problèmes rencontrés

Comme nous avons pu le voir à travers les points précédents de ce rapport, la première phase de conception a posé un problème et a impacté toutes les autres étapes. En effet, 2 semaines n'étaient vraiment pas suffisantes pour réaliser autant de tâches. La phase de conception a finalement été la partie la plus conséquente du travail et s'est étalée sur de nombreuses semaines supplémentaires, se déroulant même en parallèle d'autres phases pouvant être entreprises simultanément.

La gestion du temps a clairement été l'élément déterminant tout au long de ce travail. Il aurait fallu prévoir moins de tâches au total ou plus de temps pour cette première phase qui pose les bases pour la suite du travail. De plus, de nombreux problèmes ont également ralenti le développement de la solution. Le web scraping a montré ses limites par moments et a fait perdre des heures à trouver et tenter de corriger les erreurs ainsi qu'à optimiser le processus.

L'accès à la base de données est arrivé assez tard, lors de la semaine 7, et même si les résultats potentiels en utilisant cette dernière étaient positifs, il fallait encore trouver un moyen de pouvoir communiquer avec depuis la solution. Afin de minimiser les risques et ne pas perdre trop de temps là-dessus, le web scraping a continué à être utilisé en parallèle des recherches sur comment intégrer l'accès à la base de données. Malheureusement, comme le décrivent différents points de ce rapport sur le sujet, il n'a pas été possible de trouver une solution pour utiliser la base de données de l'entreprise et la méthode avec le web scraping a posé beaucoup de problèmes, ce qui résulte en une fonctionnalité de génération de produits à moitié fonctionnelle.

8.2.2 Planification effective du projet

Dans les faits, le projet a commencé par la première phase sur la conception de la solution comme planifiée initialement. Cependant, c'est à partir de cette dernière que le reste de la planification a été chamboulé.

La première phase a finalement duré 9 semaines, soit 4.5 fois plus que prévu à l'origine. En réalité, la seconde phase a quand même démarré en parallèle et a duré 6 semaines, soit 1.5 fois plus que prévu.

Une première ébauche du résumé et de l'affiche a pu être réalisée et présentée lors de la séance du bilan intermédiaire, puis amélioré à la fin du travail pour être inclus dans le rapport.

La phase 3 d'intégration des fonctionnalités a pu avoir lieu en quelque sorte, puisqu'elle a permis de commencer à développer l'interface de la solution en attendant l'aboutissement des fonctionnalités. Cependant, l'idée du déploiement a totalement été retirée de l'équation par manque de temps et d'avancement plus poussé dans le projet.

La fin du travail s'est déroulée comme prévu avec les tâches de finalisation, telles que la rédaction des recommandations et du guide utilisateur. Il y a également eu tout un travail de préparation des livrables qui ne sont pas inclus directement dans le rapport, comme l'entièreté du code source des différentes parties ou le fichier Figma.

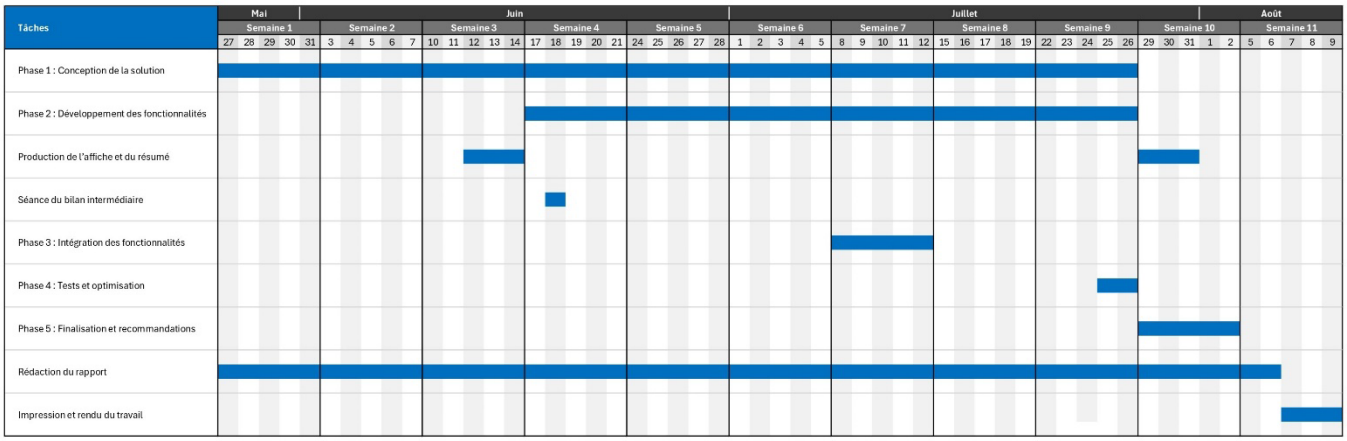


Figure 133 *- Planification effective du projet au format Gantt Chart*

9. Conclusion

En conclusion, ce projet de Bachelor a permis d'établir des bases solides pour l'automatisation de la gestion des stocks cibles chez Digitec Galaxus AG. Bien que l'ensemble des fonctionnalités prévues n'ait pas pu être développé dans le temps imparti, des avancées significatives ont été réalisées.

Les objectifs initiaux consistaient à concevoir une solution automatisée pour simplifier et optimiser le processus de gestion du stock cible, de la sélection des produits à leur enregistrement dans l'ERP. L'interface utilisateur devait être intuitive et accessible, permettant de réduire les tâches répétitives et les erreurs humaines. Malgré les défis rencontrés, notamment des délais serrés et des ressources limitées, le projet a abouti à la création d'une maquette interactive et au développement préliminaire de certaines fonctionnalités.

Ce travail a également mis en lumière l'importance de la planification minutieuse et de l'adaptabilité face aux imprévus. Les tests utilisateurs ont été déterminants dans l'amélioration de l'interface, des fonctionnalités proposées et de l'expérience utilisateur en général. L'utilisation de technologies telles que Puppeteer et Cheerio a permis de faire un pas en avant sur la faisabilité du projet, mais a également posé des problèmes à certains moments. Plus d'expérience avec ces technologies ou plus de temps pour tester en profondeur les possibilités qu'elles offrent permettraient d'améliorer encore la solution et de se rapprocher d'une version pleinement fonctionnelle. En combinant cela avec une meilleure compréhension de la base de données de l'entreprise et sur la façon de l'utiliser dans le projet, il serait possible d'arriver à une solution non seulement fonctionnelle, mais également efficace.

En somme, ce projet a non seulement permis de poser des bases techniques solides pour l'avenir, mais il a également offert des enseignements précieux pour la gestion de projets similaires. Les recommandations et le guide utilisateur serviront de référence pour les développements futurs, aidant à orienter les actions et à éviter les problèmes précédemment rencontrés.

Bibliographie

AHSAN, Mahmud, 2023. Answer to « Go back to the previous page ». Stack Overflow [en ligne]. 6 octobre 2023. Disponible à l'adresse : https://stackoverflow.com/a/77241320 [consulté le 4 juillet 2024].

ASAOLU, Elijah, 2024. Building an app with Next.js and Electron. LogRocket Blog [en ligne]. 8 mars 2024. Disponible à l'adresse : https://blog.logrocket.com/building-app-next-js-electron/ [consulté le 31 mai 2024].

BERSTEND, 2023. puppeteer-extra/packages/puppeteer-extra at master · berstend/puppeteer-extra. GitHub [en ligne]. 1 mars 2023. Disponible à l'adresse : https://github.com/berstend/puppeteer-extra/tree/master/packages/puppeteer-extra [consulté le 31 mai 2024].

BLUE SPHERE, 2023. How to Scrape (Almost) Any Website with Next.js and Puppeteer [en ligne]. 31 août 2023. Disponible à l'adresse : https://www.youtube.com/watch?v=M2Qnp9TcBiQ [consulté le 30 mai 2024].

CODINGDEFT, 2024. How to use Local Storage in Next.js [en ligne]. 27 janvier 2024. Disponible à l'adresse : https://www.youtube.com/watch?v=lB-O5q6Pdp8 [consulté le 2 juillet 2024].

COLLEGEWAP, 2023. How to use Local Storage in Next.js. DEV Community [en ligne]. 9 avril 2023. Disponible à l'adresse : https://dev.to/collegewap/how-to-use-local-storage-in-nextjs-2l2j [consulté le 2 juillet 2024].

DIGITEC GALAXUS AG, 2024a. Design System - Galaxus. Design System [en ligne]. 2024. Disponible à l'adresse : https://www.galaxus.ch/designsystem [consulté le 3 juin 2024].

DIGITEC GALAXUS AG, 2024b. Digitec Galaxus AG. GitHub [en ligne]. 2024. Disponible à l'adresse : https://github.com/DigitecGalaxus [consulté le 28 mai 2024].

DORO, Isi, 2022. Answer to « Go back to the previous page ». Stack Overflow [en ligne]. 8 novembre 2022. Disponible à l'adresse : https://stackoverflow.com/a/74368247 [consulté le 4 juillet 2024].

ERKEC, Esat, 2019. SQL multiple joins for beginners with examples. SQL Shack - articles about database auditing, server performance, data recovery, and more [en ligne]. 16 octobre 2019. Disponible à l'adresse : https://www.sqlshack.com/sql-multiple-joins-for-beginners-with-examples/ [consulté le 15 juillet 2024].

FIGMA BITES, 2024. Learn how to use Number Variables in Figma Prototypes | Figma Bites, [en ligne]. 15 février 2024. Disponible à l'adresse : https://www.youtube.com/watch?v=ZEW3jx9i10w [consulté le 10 juin 2024].

GGORLEN, 2022. Answer to « Why does headless need to be false for Puppeteer to work? » Stack Overflow [en ligne]. 1 février 2022. Disponible à l'adresse : https://stackoverflow.com/a/70936552 [consulté le 31 mai 2024].

GOOGLE, 2024. Headless mode | Puppeteer. Puppeteer [en ligne]. 2024. Disponible à l'adresse : https://pptr.dev/guides/headless-modes [consulté le 31 mai 2024].

HÄGI, Coya Vallejo, 2022. « Beaucoup de cœur à l'ouvrage » : C'est pourquoi nos concepteurs et développeurs UX ont créé un système de conception. digitec [en ligne]. 23 décembre 2022. Disponible à l'adresse : https://www.digitec.ch/fr/page/beaucoup-de-coeur-a-louvrage-cest-pourquoi-nos-concepteurs-et-developpeurs-ux-ont-cree-un-systeme-de-25977 [consulté le 13 juin 2024].

HOLISTICS SOFTWARE, 2024. dbdiagram.io - Database Relationship Diagrams Design Tool. dbdiagram.io [en ligne]. 2024. Disponible à l'adresse : https://dbdiagram.io/home [consulté le 1 août 2024].

ISAIAH, Ayooluwa, 2019. Build a web scraper with Node | Pusher tutorials. Pusher [en ligne]. 21 février 2019. Disponible à l'adresse : https://pusher.com/tutorials/web-scraper-node/#scrape-a-dynamic-website-using-puppeteer [consulté le 28 mai 2024].

JOSH TRIED CODING, 2022. Let's Build a WebScraper in 15 Minutes with NextJS! [en ligne]. 22 novembre 2022. Disponible à l'adresse : https://www.youtube.com/watch?v=bGShHOOoC-U [consulté le 28 mai 2024].

LIENDO, Michael, 2020. Create a public API by web scraping in NextJS. DEV Community [en ligne]. 16 octobre 2020. Disponible à l'adresse : https://dev.to/focusotter/create-a-public-api-by-web-scraping-in-nextjs-2f5n [consulté le 28 mai 2024].

MANY CONTRIBUTORS, 2023. @types/cheerio. npm [en ligne]. 20 novembre 2023. Disponible à l'adresse : https://www.npmjs.com/package/@types/cheerio [consulté le 28 mai 2024].

MANY CONTRIBUTORS, 2024. Documentation - Object Types. TypeScript [en ligne]. 5 août 2024. Disponible à l'adresse : https://www.typescriptlang.org/docs/handbook/2/objects.html [consulté le 31 juillet 2024].

MAWA, Joseph, 2023. Reading and writing JSON files in Node.js: A complete tutorial. LogRocket Blog [en ligne]. 28 septembre 2023. Disponible à l'adresse : https://blog.logrocket.com/reading-writing-json-files-node-js-complete-tutorial/ [consulté le 12 juin 2024].

MDN CONTRIBUTORS, 2023a. Storage quotas and eviction criteria - Web APIs | MDN. MDN Web Docs [en ligne]. 25 octobre 2023. Disponible à l'adresse : https://developer.mozilla.org/en-US/docs/Web/API/Storage_API/Storage_quotas_and_eviction_criteria [consulté le 5 juillet 2024].

MDN CONTRIBUTORS, 2023b. try...catch - JavaScript | MDN. MDN Web Docs [en ligne]. 4 août 2023. Disponible à l'adresse : https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements/try...catch [consulté le 19 juin 2024].

MICROSOFT, [sans date]. Visual Studio Code - Code Editing. Redefined. Visual Studio Code [en ligne]. [sans date]. Disponible à l'adresse : https://code.visualstudio.com/ [consulté le 13 juin 2024].

MUHSEN, musaa, 2022. Answer to « Go back to the previous page ». Stack Overflow [en ligne]. 22 juin 2022. Disponible à l'adresse : https://stackoverflow.com/a/72721050 [consulté le 4 juillet 2024].

OPENJS FOUNDATION AND ELECTRON CONTRIBUTORS, 2023. Build cross-platform desktop apps with JavaScript, HTML, and CSS | Electron. Electron [en ligne]. 2023. Disponible à l'adresse : https://electronjs.org/ [consulté le 31 mai 2024].

OSTEZER AND MARK DRAKE, 2022. SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems | DigitalOcean. DigitalOcean [en ligne]. 9 mars 2022. Disponible à l'adresse : https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems [consulté le 1 août 2024].

OZHANLI, 2023. Passing Data from Child to Parent Components in React. Medium [en ligne]. 11 avril 2023. Disponible à l'adresse : https://medium.com/@ozhanli/passing-data-from-child-to-parent-components-in-react-e347ea60b1bb [consulté le 4 juillet 2024].

POLACEK, John, 2021 a. Deployments · johnpolacek/nextjs-scraper-playground. GitHub [en ligne]. 23 mai 2021. Disponible à l'adresse : https://github.com/johnpolacek/nextjs-scraper-playground [consulté le 28 mai 2024].

POLACEK, John, 2021b. Next.js Scraper Playground. Next.js Scraper Playground [en ligne]. 23 mai 2021. Disponible à l'adresse : https://nextjs-scraper-playground.vercel.app/ [consulté le 28 mai 2024].

PRÉVOST, Clément, 2018. Answer to « Go back to the previous page ». Stack Overflow [en ligne]. 9 septembre 2018. Disponible à l'adresse : https://stackoverflow.com/a/52243879 [consulté le 4 juillet 2024].

PRICE, Todd, 2020. Answer to « Why does headless need to be false for Puppeteer to work? » Stack Overflow [en ligne]. 9 septembre 2020. Disponible à l'adresse : https://stackoverflow.com/a/63820507 [consulté le 31 mai 2024].

PRISMA DATA, INC., 2024a. Many-to-many relations | Prisma Documentation. Prisma [en ligne]. 2024. Disponible à l'adresse : https://www.prisma.io/docs/orm/prisma-schema/data-model/relations/many-to-many-relations [consulté le 5 juillet 2024].

PRISMA DATA, INC., 2024b. Prisma Documentation. Prisma [en ligne]. 2024. Disponible à l'adresse : https://www.prisma.io/docs/ [consulté le 1 août 2024].

PRISMA DATA, INC., 2024c. Seeding | Prisma Documentation. Prisma [en ligne]. 2024. Disponible à l'adresse : https://www.prisma.io/docs/orm/prisma-migrate/workflows/seeding [consulté le 5 juillet 2024].

PRISMA DATA, INC., 2024d. Seeding | Prisma Documentation. Prisma [en ligne]. 2024. Disponible à l'adresse : https://www.prisma.io/docs/orm/prisma-migrate/workflows/seeding [consulté le 1 août 2024].

PRISMA DATA, INC., 2024e. SQLite database connector | Prisma Documentation. Prisma [en ligne]. 2024. Disponible à l'adresse : https://www.prisma.io/docs/orm/overview/databases/sqlite [consulté le 5 juillet 2024].

REACT TOOLTIP, 2024. Getting Started | React Tooltip. React Tooltip [en ligne]. 2024. Disponible à l'adresse : https://react-tooltip.com/docs/getting-started [consulté le 3 juillet 2024].

REKIER, Skrevet av Geoffrey, 2023. Running Puppeteer on Vercel or other Serverless. AgilXP [en ligne]. 2023. Disponible à l'adresse : https://agilxp.no/blogg/running-puppeteer-on-vercel [consulté le 31 mai 2024].

RSTORM861, 2021. [Bug] Using puppeteer-extra-plugin-stealth doesn't compile when using React and Electron · Issue #597 · berstend/puppeteer-extra. GitHub [en ligne]. 6 décembre 2021. Disponible à l'adresse : https://github.com/berstend/puppeteer-extra/issues/597 [consulté le 31 mai 2024].

SAKI, Qrow, 2020. Why does headless need to be false for Puppeteer to work? Stack Overflow [en ligne]. 9 septembre 2020. Disponible à l'adresse : https://stackoverflow.com/q/63818869 [consulté le 31 mai 2024].

TAILWIND LABS INC, 2024. Installation - Tailwind CSS. [en ligne]. 2024. Disponible à l'adresse : https://tailwindcss.com/docs/installation [consulté le 2 juillet 2024].

THE CHEERIO CONTRIBUTORS, 2024a. Loading Documents | cheerio. Cheerio [en ligne]. 2024. Disponible à l'adresse : https://cheerio.js.org/docs/basics/loading [consulté le 28 mai 2024].

THE CHEERIO CONTRIBUTORS, 2024b. Selecting Elements | cheerio. Cheerio [en ligne]. 2024. Disponible à l'adresse : https://cheerio.js.org/docs/basics/selecting [consulté le 28 mai 2024].

THE CHEERIO CONTRIBUTORS, 2024c. Welcome to Cheerio! | cheerio. Cheerio [en ligne]. 2024. Disponible à l'adresse : https://cheerio.js.org/docs/intro [consulté le 28 mai 2024].

THE CURIOUS PROGRAMMER, 2023. Super Powers with Puppeteer. The Curious Programmer [en ligne]. 14 juillet 2023. Disponible à l'adresse : https://jasonroell.com/2023/07/14/super-powers-with-puppeteer/ [consulté le 27 juin 2024].

THEHUMANJPG, [sans date]]. Goodbye John Doe. Goodbye John Doe [en ligne]. [sans date]]. Disponible à l'adresse : http://goodbyejohndoe.com [consulté le 11 juin 2024].

TUTORIALS, Coding Beauty, 2023. How to quickly listen for a route/page change in Next.js. Medium [en ligne]. 3 octobre 2023. Disponible à l'adresse : https://medium.com/@codingbeautytutorials/nextjs-detect-route-change-bb8e6c316d1 [consulté le 2 juillet 2024].

VENTURINO, Eric, 2024. (7) Qu'est-ce qu'un ORM (Object-Relational Mapping) ? | LinkedIn. Linkedin [en ligne]. 18 mars 2024. Disponible à l'adresse : https://www.linkedin.com/pulse/quest-ce-quun-orm-object-relational-mapping-eric-venturino-0ofgf/ [consulté le 1 août 2024].

VERCEL, 2024a. Getting Started: Installation | Next.js. NEXT.js [en ligne]. 2024. Disponible à l'adresse : https://nextjs.org/docs/getting-started/installation [consulté le 28 mai 2024].

VERCEL, 2024b. next.js/examples/with-electron-typescript at canary · vercel/next.js. GitHub [en ligne]. 12 janvier 2024. Disponible à l'adresse : https://github.com/vercel/next.js/tree/canary/examples/with-electron-typescript [consulté le 31 mai 2024].

VICTORCESAE, 2023. Answer to « Go back to the previous page ». Stack Overflow [en ligne]. 27 décembre 2023. Disponible à l'adresse : https://stackoverflow.com/a/77723420 [consulté le 4 juillet 2024].

VORA, Shubham, 2023. Extract a number from a string using JavaScript. Tutorials Point [en ligne]. 14 septembre 2023. Disponible à l'adresse : https://www.tutorialspoint.com/extract-a-number-from-a-string-using-javascript [consulté le 15 juillet 2024].

WIERUCH, Robin, 2024. Next.js with Prisma and SQLite. Robin Wieruch - Freelance Web Developer [en ligne]. 21 mars 2024. Disponible à l'adresse : https://www.robinwieruch.de/next-prisma-sqlite/ [consulté le 5 juillet 2024].

WIKIPÉDIA, 2023. Méthode de la pensée à voix haute. Wikipédia [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=M%C3%A9thode_de_la_pens%C3%A9e_%C3%A0_voix_haute&oldid=204791612 [consulté le 12 juin 2024]. Page Version ID : 204791612

ДМИТРИЙ, 2024. Go back to the previous page. Stack Overflow [en ligne]. 11 avril 2024. Disponible à l'adresse : https://stackoverflow.com/q/51843119 [consulté le 4 juillet 2024].

Glossaire

API	Ensemble de définitions et de protocoles permettant à des logiciels de communiquer entre eux.
Back-end	Partie d'une application web responsable du traitement des données et de la logique métier.
Cheerio	Bibliothèque JavaScript permettant de manipuler le HTML du côté serveur.
Code source	Ensemble d'instructions écrites dans un langage de programmation spécifique, définissant le comportement d'un logiciel.
ERP	Système de gestion intégré utilisé pour gérer et automatiser les processus d'affaires.
Figma	Outil de conception d'interface collaborative permettant de créer des maquettes interactives.
Framework web	Ensemble d'outils et de bibliothèques facilitant le développement d'applications web.
Front-end	Partie visible d'une application web avec laquelle les utilisateurs interagissent.
Interface utilisateur (UI)	Conception visuelle permettant aux utilisateurs d'interagir avec une application ou un site web.
JSON	Format léger de données utilisé pour les échanges entre un serveur et une application web.
Maquette interactive	Prototype de l'application permettant de simuler et tester les fonctionnalités prévues.
Méthode agile	Approche de gestion de projet favorisant la flexibilité, la collaboration et les cycles de développement itératifs.

Next.js	Framework web basé sur React pour construire des applications web modernes.
Puppeteer	Bibliothèque Node.js permettant de contrôler un navigateur de manière programmatique pour automatiser des tâches web.
Stock cible	Assortiment de produits essentiels et nouveautés renouvelés régulièrement dans les succursales.
SQL	Langage de programmation utilisé pour gérer et manipuler des bases de données relationnelles.
Web scraping	Technique utilisée pour extraire des données de sites web.

Table des abréviations et acronymes

API	Application Programming Interface
CSS	Cascading Style Sheets
ERP	Enterprise Resource Planning
EOL	End Of Life
HTML	HyperText Markup Language
JS	JavaScript
SQL	Structured Query Language
TB	Travail de Bachelor
UX	User Experience
UI	User Interface

Annexes

Annexe 1 Clé USB

Annexe 2 Tests utilisateurs

Annexe 3 Liste de recommandations

Annexe 4 Guide d'utilisation

Annexe 5 Rapport de la pré-étude

Annexe 6 Reproduction de l'affiche

Annexe 7 Copie du formulaire de Clause de confidentialité

Annexe 8 Copie du procès-verbal du bilan intermédiaire

Annexe 1

Clé USB

La clé USB fournie avec ce rapport contient les fichiers suivants :

- Code source de l'interface de test avec les fonctions de récupération de produits et de catégories.
- Code source de l'interface de test avec la fonction de récupération de produits améliorée et celle de récupération des catégories.
- Code source de l'interface correspondant à la version intermédiaire de la maquette et incluant la base de données.
- Fichier Figma regroupant toutes les versions de la maquette.

Annexe 2

Tests utilisateurs

Tests utilisateurs

Objectifs et procédure

Le but de ces entretiens est d'évaluer l'intuitivité de l'interface utilisateur de la solution et la pertinence des fonctionnalités proposées. D'une durée d'environ 1h30, les entretiens permettront de passer en revue la quasi-totalité de l'application en réalisant un certain nombre de tâches en lien avec un scénario précis.

Pour chaque scénario, une combinaison d'observations et de questions sur la tâche réalisée permettront d'établir un retour d'expérience de l'utilisateur.

Au cours de ces entretiens, les utilisateurs exprimeront leurs impressions à haute voix afin de mieux comprendre ce qu'ils pensent et voient. Ils pourront également partager leur avis sur les points positifs et les points d'amélioration de la solution. Les tests se feront sur place ou à distance via Teams, selon la localisation du testeur.

Personne 1

Département	Retail
Fonction	Store Manager
Emplacement	Lausanne
Date	14.06.2024
Lieu du test	Sur place
Langue du test	Français

Scénario 1 : Connexion et accès au tableau de bord

Situation	Vous souhaitez définir le stock cible de la saison à venir et vous vous rendez donc sur l'outil dédié. Vous vous retrouvez premièrement sur la page de connexion.
Tâche	Connectez-vous et accédez au tableau de bord.

Les informations du tableau de bord sont-elles claires et compréhensibles ?

> Il manque un peu de clarté sur la signification des statistiques.

Quelles informations souhaitez-vous voir sur ce tableau de bord ?

> Un classement des ventes et pourquoi pas la possibilité de comparer avec des concurrents pour avoir des idées de produits à ajouter.

Observations et recommandations

L’utilisateur est arrivé rapidement sur le tableau de bord et son regard est attiré par la statistique sur la meilleure vente. Il regarde ensuite les 2 autres données mais ne comprend pas directement ce que les statistiques signifient et à quoi elles sont relatives. Il suggère d’ajouter ces informations dans les légendes pour comprendre plus rapidement leurs significations.

Scénario 2 : Choix de la succursale et de la langue

Situation	Avant de commencer à gérer le stock cible, vous souhaitez définir quelques paramètres de base comme la langue et votre succursale notamment.
Tâche	Trouvez l'option permettant de modifier la succursale et la langue.

L'option est-elle facilement reconnaissable ?

Moyennement, il ferait sens d'avoir ce réglages avec les autres paramètres.

Le fonctionnement de l'option est-il intuitif ?

La sélection en soit est intuitive mais l’emplacement pose problème, le participant suggère de déplacer l’option soit en 2 listes déroulantes séparées ailleurs pour plus de facilité ou alors dans les paramètres généraux.

Observations et recommandations

Le participant s’est d’abord rendu dans les paramètres globaux mais n’a pas trouvé l’option puis il a survolé les icônes de l’entête et il a découvert l’option. Il n'était pas évident de reconnaitre l'icône lié à ce paramètre. Pour lui, il ferait sens de déplacer ce réglage, par exemple le regrouper avec le reste des paramètres.

Scénario 3 : Sélection des produits pour le stock cible

Situation	Vous souhaitez à présent générer une liste de produits à destination du stock cible et souhaitez que le système définisse automatiquement la liste.
Tâche	Suivez le processus pour générer une liste de produits automatiquement.

Le processus de génération est-il compréhensible ?

En partie, il faudrait avoir plus d’informations sur comment la liste est définie.

La disposition des produits est-elle agréable ?

Oui, conforme à ce que l’on retrouve sur le site (galaxus.ch et digitec.ch).

Observations et recommandations

Le participant arrive sur l'onglet dédié est comprend rapidement qu'il a 2 options, une rapide avec l'automatique et une plus lente avec le manuel. Il clique ensuite sur le bouton correspondant pour générer une liste automatiquement et arrive, après un temps de chargement, sur la vue avec la liste générée. Il a cependant un manque de compréhension sur comment la liste est générée. Il suggère d'ajouter cette information sur la page où l'on clique sur la méthode de sélection.

Scénario 4 : Modification des règles pour le stock cible

Situation	Maintenant que vous avez votre liste de produits, vous souhaitez définir les règles qui vont paramétrer la disponibilité du stock cible.
Tâche	Modifiez les règles (succursale, dates et quantité) pour tous les produits puis pour un produit.

Avez-vous trouvé facilement où modifier les règles ? (Global et individuel)

Pas vraiment, il y a eu une confusion entre l'icône de réglage sur la page des produits et celui sur l'entête. Sinon l'option individuelle est facilement reconnaissable.

L'éventail d'options pour les règles est-il complet ?

Oui, il y a toutes les options nécessaires.

Observations et recommandations

Le participant a parcouru la page mais n'a pas vu tout de suite où se trouvait l'option des règles du stock cible mais a fini par la voir en survolant plusieurs éléments. Il serait pertinent de le déplacer avec les autres options globales. Il y a également un manque d'informations sur la façon dont cette phase fonctionne et un manque de compréhension sur l'ordre des actions à réaliser avant de continuer.

Scénario 5 : Modification de la liste de produit

Situation	La liste de produits vous convient partiellement, vous souhaitez la modifier en ajoutant un produit et en supprimant d'autres.
Tâche	Ajoutez un produit puis supprimez en plusieurs.

La méthode d'ajout d'un produit est-elle intuitive ?

Oui, elle est vite reconnaissable avec l'icône "+" et le fait de simplement ajouter un lien galaxus vers le produit simplifie le processus.

L'option vous permettant de sélectionner plusieurs produits et de les supprimer est-elle facile d'utilisation ?

Oui, c'est quelque-chose de connu.

Observations et recommandations

Le participant a trouvé rapidement comment ajouter un produit et en supprimer plusieurs. Il n’a eu aucun problèmes durant cette tâche.

Scénario 6 : Validation et enregistrement des produits

Situation	Vous êtes satisfait de la liste de produits et souhaitez finaliser le processus de création et d'enregistrement du stock cible.
Tâche	Continuez le processus et validez la liste de produits pour l'enregistrement dans l'ERP.

Le résumé des détails du stock cible vous semble-t-il complet ?

Oui, il est complet. On retrouve toutes les informations nécessaires avant de continuer.

Le processus d'enregistrement des produits est-il clair ?

Oui, le processus est clair, on comprend facilement à quelle étape nous sommes.

Observations et recommandations

Le participant a passé en revue les informations présentes sur le résumé puis a continué le processus. Il a cliqué instinctivement sur le bouton à droite pour continuer alors qu’il s’agit de celui pour revenir en arrière. Il n’y a pas eu d’autres problèmes autrement. Il est envisageable de revoir le positionnement des boutons pour éviter ce genre d’erreurs. Il a apprécié le fait que le chargement pour l’enregistrement dans l’ERP bénéficie d’une animation particulière.

Scénario 7 : Consultation et modification du stock actuel

Situation	Le processus de sélection et d'enregistrement est à présent terminé mais vous venez de penser à un produit qui serait intéressant pour le stock cible. Vous souhaitez donc rajouter ce produit au stock actuel.
Tâche	Accédez à la liste de produits du stock actuel et ajoutez un nouveau produit.

La liste du stock actuel est-elle rapidement et facilement accessible ?

Oui, on le remarque vite. On peut passer au choix par la navigation dans l’entête ou via le raccourci présent dans le message de succès à la fin du processus précédent.

Observations et recommandations

Le participant a rapidement trouvé l’onglet et a pu ajouter un produit. Le fait d’avoir expérimenter quelque chose de similaire avant a sûrement aider.

Scénario 8 : Gestion des accès à l'application

Situation	En tant que super utilisateur, vous pouvez gérer les accès à l'application et vous souhaitez ajouter un collègue pour qu'il puisse également gérer le stock cible. (L'accès est premièrement limité par

	l'utilisation de la connexion avec Microsoft puis en vérifiant qu'il s'agisse bien d'une adresse e-mail de l'entreprise).
Tâche	Accédez aux paramètres d'accès à l'application et ajouter un utilisateur.

Avez-vous trouvé facilement où modifier les accès à l'application ?

> Pas vraiment, il ferait sens de l'avoir avec les autres réglages globaux.

Trouvez pertinent de limiter l'accès en plus des limitations déjà en place ?

> Oui et non, cela a du sens mais l'accès est déjà limité par les accès que les personnes ont dans l'ERP mais par principe de sécurité cette option a du sens.

Observations et recommandations

> L'utilisateur a hésité entre les paramètres généraux et ceux du compte. Mais il a finit par trouver et à arriver au but. Ce n'était pas clair où allait être l'option. Il a par la suite pu rapidement trouver l'option permettant d'ajouter un utilisateur.

Feedback général

Qu'avez-vous le plus apprécié dans ce prototype ?

> L'intégration d'une option relative au thème (sombre et clair), le design system respecté, les animations sympathiques et le design épuré.

Quelles sont les principales difficultés que vous avez rencontrées ?

> Trouver une option était parfois difficile, comprendre ce que fait le processus par moment.

Avez-vous des suggestions d'améliorations pour ce prototype ?

> Un peu trop d'options dans certains cas, l'emplacement de certaines options pourraient être revus, manque d'informations par moment sur ce qu'il faut faire ou ce qu'il se passe.

Sur une échelle de 1 à 10, comment évalueriez-vous l'utilisabilité de cette interface ?

> 7 sur 10

Personne 2

Département	Retail
Fonction	Deputy Store Manager
Emplacement	Lausanne
Date	15.06.2024

Lieu du test	Sur place
Langue du test	Français

Scénario 1 : Connexion et accès au tableau de bord

Situation	Vous souhaitez définir le stock cible de la saison à venir et vous vous rendez donc sur l'outil dédié. Vous vous retrouvez premièrement sur la page de connexion.
Tâche	Connectez-vous et accédez au tableau de bord.

Les informations du tableau de bord sont-elles claires et compréhensibles ?

> Oui, on comprend ce que les données signifient mais elles pourraient être triées par pertinence.

Quelles informations souhaitez-vous voir sur ce tableau de bord ?

> Une liste des meilleures ventes actuelle et celle de l'année précédente (top 10, top 50, top 100), classement séparé entre digitec et galaxus.

Observations et recommandations

> L'utilisateur est arrivé rapidement sur le tableau de bord et il a visualisé l'ensemble des informations présentées. L'information sur le magasin actuelle est peut être en trop d'après le participant. On pourrait réarranger les cases du tableau de bord.

Scénario 2 : Choix de la succursale et de la langue

Situation	Avant de commencer à gérer le stock cible, vous souhaitez définir quelques paramètres de base comme la langue et votre succursale notamment.
Tâche	Trouvez l'option permettant de modifier la succursale et la langue.

L'option est-elle facilement reconnaissable ?

> Oui très, cela correspond à l'option présente sur galaxus.

Le fonctionnement de l'option est-il intuitif ?

> Oui, encore une fois c'est du connu, c'est simple et efficace.

Observations et recommandations

> Le participant a tout de suite trouvé l'option car elle est similaire à celle présente sur galaxus.

Scénario 3 : Sélection des produits pour le stock cible

Situation	Vous souhaitez à présent générer une liste de produits à destination du stock cible et souhaitez que le système définisse automatiquement la liste.
Tâche	Suivez le processus pour générer une liste de produits automatiquement.

Le processus de génération est-il compréhensible ?

> Oui, il pourrait manqué l'information sur quels critères se base la liste mais on devine en voyant la liste de produits.

La disposition des produits est-elle agréable ?

> Oui, cela reprend la mise en page du site , on identifie facilement les produits

Observations et recommandations

> Le participant a rapidement trouvé l'onglet dédié au nouveau stock et a cliquer sur le bon bouton pour générer la liste. Il a ensuite parcouru la liste de produits.

Scénario 4 : Modification des règles pour le stock cible

Situation	Maintenant que vous avez votre liste de produits, vous souhaitez définir les règles qui vont paramétrer la disponibilité du stock cible.
Tâche	Modifiez les règles (succursale, dates et quantité) pour tous les produits puis pour un produit.

Avez-vous trouvé facilement où modifier les règles ? (Global et individuel)

> Oui, on devine facilement que l'engrenage correspond au paramètres globaux et le crayon sur les produits pour les paramètres individuels.

L'éventail d'options pour les règles est-il complet ?

> Oui mais il y en a un peu trop, les dates dans les paramètres individuels seraient en trop, en général les produits ont tous la même période de disponibilités, l'option pour avoir tous les magasins n'est pas nécessaire, chaque magasin gère son propre stock.

Observations et recommandations

> Le participant a parcouru la page et a trouvé facilement les 2 options. Il suggère de réduire les options dans ces réglages.

Scénario 5 : Modification de la liste de produit

Situation	La liste de produits vous convient partiellement, vous souhaitez la modifier en ajoutant un produit et en supprimant d'autres.
Tâche	Ajoutez un produit puis supprimez en plusieurs.

La méthode d'ajout d'un produit est-elle intuitive ?

> Oui, on comprend ce que l'icône fait et on peut rapidement entrer un lien pour ajouter un produit, il serait bien de pouvoir ajouter un lien de l'ERP également.

L'option vous permettant de sélectionner plusieurs produits et de les supprimer est-elle facile d'utilisation ?

> L’icône est parlant et indique qu’on peut sélectionner plusieurs produit et la sélection se fait rapidement.

Observations et recommandations

> Le participant a rapidement compris comment ajouter un produit et faire une sélection pour en modifier d’autres.

Scénario 6 : Validation et enregistrement des produits

Situation	Vous êtes satisfait de la liste de produits et souhaitez finaliser le processus de création et d'enregistrement du stock cible.
Tâche	Continuez le processus et validez la liste de produits pour l'enregistrement dans l'ERP.

Le résumé des détails du stock cible vous semble-t-il complet ?

> Oui mais la quantité par produits est utile seulement quand tous les produits ont la même quantité, à la place d’avoir cela, il pourrait y avoir la quantité de produits totale et le nombre de références.

Le processus d'enregistrement des produits est-il clair ?

> Oui, on comprend ce qu’il se passe aisément.

Observations et recommandations

> Le participant est arrivé rapidement à la fin du processus et comprend ce qu’il se passe à chaque étape. Il a juste relevé des points d’amélioration au niveau du résumé.

Scénario 7 : Consultation et modification du stock actuel

Situation	Le processus de sélection et d'enregistrement est à présent terminé mais vous venez de penser à un produit qui serait intéressant pour le stock cible. Vous souhaitez donc rajouter ce produit au stock actuel.
Tâche	Accédez à la liste de produits du stock actuel et ajoutez un nouveau produit.

La liste du stock actuel est-elle rapidement et facilement accessible ?

> Oui, autant dans les onglets de navigation que à partir du lien présent sur le message de succès à la fin du processus de création du stock cible

Observations et recommandations

Le participant est arrivé rapidement sur la bonne page, il est passé par le lien présent dans le message de succès à l'étape précédente qu'il trouve par ailleurs utile et il a pu ajouter un produit aisément car le processus est identique à ce qui a été vu précédemment.

Scénario 8 : Gestion des accès à l'application

Situation	En tant que super utilisateur, vous pouvez gérer les accès à l'application et vous souhaitez ajouter un collègue pour qu'il puisse également gérer le stock cible. (L'accès est premièrement limité par l'utilisation de la connexion avec Microsoft puis en vérifiant qu'il s'agisse bien d'une adresse e-mail de l'entreprise).
Tâche	Accédez aux paramètres d'accès à l'application et ajouter un utilisateur.

Avez-vous trouvé facilement où modifier les accès à l'application ?

Oui

Trouvez pertinent de limiter l'accès en plus des limitations déjà en place ?

Oui mais il faudrait avoir une vue par succursale où il y a un administrateur par magasin qui peut gérer les accès.

Observations et recommandations

Le participant a trouvé rapidement l'option et a pu ajouter un utilisateur. Il a cliqué instinctivement sur l'onglet compte et est arrivé sur la bonne page. Il suggère que l'accès soit régi par succursale dans le sens où seulement les personnes des succursales concernés peuvent éditer le stock cible avec chaque fois un admin qui gère les accès et que le remplaçant du chef du retail puisse lui tout modifier.

Feedback général

Qu'avez-vous le plus apprécié dans ce prototype ?

Design, intuitivité, facilité d'utilisation

Quelles sont les principales difficultés que vous avez rencontrées ?

La compréhension de l'icône permettant de tout sélectionner

Avez-vous des suggestions d'améliorations pour ce prototype ?

Délocaliser l'option permettant de gérer les options de la sélection automatique sur la page du nouveau stock pour définir ce paramètre avant de générer la liste.

Sur une échelle de 1 à 10, comment évalueriez-vous l'utilisabilité de cette interface ?

9 sur 10

Personne 3

Département	Retail
Fonction	Store Manager
Emplacement	Winterthur
Date	01.07.2024
Lieu du test	A distance via Teams
Langue du test	Anglais

Scénario 1 : Connexion et accès au tableau de bord

Situation	Vous souhaitez définir le stock cible de la saison à venir et vous vous rendez donc sur l'outil dédié. Vous vous retrouvez premièrement sur la page de connexion.
Tâche	Connectez-vous et accédez au tableau de bord.

Les informations du tableau de bord sont-elles claires et compréhensibles ?

Oui, mais il manque quelques détails pour mieux comprendre à quoi les informations se rapportent.

Quelles informations souhaitez-vous voir sur ce tableau de bord ?

Un classement de ce que l'on vend le plus et la possibilité de voir les produits qui se vendent le moins dans la succursale pour pouvoir les retirer si nécessaire.

Observations et recommandations

L'utilisateur est arrivé rapidement sur le tableau de bord et il a visualisé l'ensemble des informations présentées. Il soulevé le fait que l'ensemble pourrait être plus détaillé pour mieux comprendre ce que les données représentes en plus des autres suggestions qu'il a faites.

Scénario 2 : Choix de la succursale et de la langue

Situation	Avant de commencer à gérer le stock cible, vous souhaitez définir quelques paramètres de base comme la langue et votre succursale notamment.
Tâche	Trouvez l'option permettant de modifier la succursale et la langue.

L'option est-elle facilement reconnaissable ?

Oui, il s'agit du même emplacement que sur galaxus.

Le fonctionnement de l'option est-il intuitif ?

Oui, il est utile de l'avoir à portée de main rapidement.

Observations et recommandations

> Le participant a instinctivement cliquer sur le bon icône car son design et son utilisation est similaire à ce que l'on retrouve sur galaxus. Il apprécie le fait que l'option est rapidement accessible car lors d'une réunion sur Teams, s'il doit partager son écran, il peut rapidement changer la langue pour que tout le monde puisse comprendre.

Scénario 3 : Sélection des produits pour le stock cible

Situation	Vous souhaitez à présent générer une liste de produits à destination du stock cible et souhaitez que le système définisse automatiquement la liste.
Tâche	Suivez le processus pour générer une liste de produits automatiquement.

Le processus de génération est-il compréhensible ?

> Oui, le système semble générer une liste de produits que l'on vend le plus.

La disposition des produits est-elle agréable ?

> Oui, mais une pagination serait préférable à un bouton "Afficher plus" car elle éviterait d'avoir à faire défiler une liste presque infinie, rendant la navigation plus agréable.

Observations et recommandations

> Le participant a accéder à la section via la navigation de l'entête et a cliqué directement sur le bon bouton permettant la génération automatique. Il n'a pas eu de difficultés à réaliser cette tâche, il a juste relever le fait qu'il serait plus agréable d'avoir une pagination pour la liste plutôt qu'une liste qui se rallonge au clic d'un bouton.

Scénario 4 : Modification des règles pour le stock cible

Situation	Maintenant que vous avez votre liste de produits, vous souhaitez définir les règles qui vont paramétrer la disponibilité du stock cible.
Tâche	Modifiez les règles (succursale, dates et quantité) pour tous les produits puis pour un produit.

Avez-vous trouvé facilement où modifier les règles ? (Global et individuel)

> Oui, leur emplacement était logique.

L'éventail d'options pour les règles est-il complet ?

> Oui, il y a les options qu'il faut.

Observations et recommandations

Il a facilement trouvé l’emplacement des deux options, ce qui lui a semblé logique. Il en va de même pour les réglages disponibles. Cependant, il a fait une remarque concernant la sélection de toutes les succursales dans les paramètres globaux. Il serait utile d'ajouter un avertissement indiquant que cela pourrait supprimer les listes existantes si d’autres personnes ont déjà défini leur propre stock cible.

Scénario 5 : Modification de la liste de produit

Situation	La liste de produits vous convient partiellement, vous souhaitez la modifier en ajoutant un produit et en supprimant d'autres.
Tâche	Ajoutez un produit puis supprimez en plusieurs.

La méthode d'ajout d'un produit est-elle intuitive ?

Oui mais il pourrait être utile d'ajouter plusieurs produits en même temps.

L'option vous permettant de sélectionner plusieurs produits et de les supprimer est-elle facile d'utilisation ?

Oui, elle est facile à repérer et à utiliser.

Observations et recommandations

Le participant a trouvé rapidement l’option pour ajouter un produit et l’a trouvée facile d’utilisation. Cependant, il suggère qu'il serait intéressant de pouvoir ajouter plusieurs liens en même temps afin de réduire le nombre de clics et de gagner du temps. Par la suite, il a accédé aisément à la vue permettant de sélectionner plusieurs produits à la fois pour les modifier ou les supprimer. Il est satisfait de voir qu’il peut également modifier les règles pour plusieurs produits en plus des règles globales et individuelles. La question sur la pertinence du bouton permettant de tout sélectionner a été posée, et il trouve cette option utile. En effet, s'il souhaite ne garder que cinq produits par exemple, il peut tout sélectionner puis désélectionner uniquement ceux qu'il veut conserver avant de supprimer le reste.

Scénario 6 : Validation et enregistrement des produits

Situation	Vous êtes satisfait de la liste de produits et souhaitez finaliser le processus de création et d'enregistrement du stock cible.
Tâche	Continuez le processus et validez la liste de produits pour l'enregistrement dans l'ERP.

Le résumé des détails du stock cible vous semble-t-il complet ?

Oui mais il serait préférable d’avoir la quantité totale de produits à stocker plutôt que la quantité par produit sachant que des produits peuvent avoir des quantités différentes.

Le processus d'enregistrement des produits est-il clair ?

Oui, on comprend facilement avec la barre de chargement et le message de fin.

Observations et recommandations

Le participant a cliqué instinctivement sur le bon bouton pour continuer le processus d'enregistrement et a facilement compris les informations présentes dans le résumé. Cependant, il a remarqué que la quantité par produit devrait plutôt être la quantité totale, sachant que les produits peuvent avoir des quantités différentes. Une fois le résumé validé, il a poursuivi et a terminé le processus. Il a particulièrement apprécié l'animation lors du chargement des produits dans l'ERP.

Scénario 7 : Consultation et modification du stock actuel

Situation	Le processus de sélection et d'enregistrement est à présent terminé mais vous venez de penser à un produit qui serait intéressant pour le stock cible. Vous souhaitez donc rajouter ce produit au stock actuel.
Tâche	Accédez à la liste de produits du stock actuel et ajoutez un nouveau produit.

La liste du stock actuel est-elle rapidement et facilement accessible ?

Oui, il est facile d'accès via la navigation dans l'entête.

Observations et recommandations

Il a rapidement trouvé la page en y accédant depuis les liens dans l'entête et, comme le processus pour ajouter un produit est le même que lors de la génération de la liste, il a pu en ajouter un facilement. Il souligne qu'il serait utile de pouvoir ajouter la quantité du produit au moment de l'ajouter dans la liste afin d'accélérer le processus. En testant les autres options sur cette page, il remarque que la possibilité de modifier les dates à ce stade n'est pas forcément utile. Il a également questionné la manière dont le produit s'affiche une fois ajouté, notamment son emplacement dans la liste, c'est-à-dire s'il apparaît au début ou à la fin de la liste. Ce point n'avait pas été pris en compte lors de la création de l'interface, mais sera considéré lors de son amélioration. Il note également que la modification des dates par produit n'est pas nécessaire, que ce soit lors de l'ajout ou de la génération de la liste, car tous les produits sont prévus pour la même période.

Scénario 8 : Gestion des accès à l'application

Situation	En tant que super utilisateur, vous pouvez gérer les accès à l'application et vous souhaitez ajouter un collègue pour qu'il puisse également gérer le stock cible. (L'accès est premièrement limité par l'utilisation de la connexion avec Microsoft puis en vérifiant qu'il s'agisse bien d'une adresse e-mail de l'entreprise).
Tâche	Accédez aux paramètres d'accès à l'application et ajouter un utilisateur.

Avez-vous trouvé facilement où modifier les accès à l'application ?

Oui

Trouvez pertinent de limiter l'accès en plus des limitations déjà en place ?

Oui

Observations et recommandations

Le participant a rapidement trouvé l'option en cliquant sur l'icône du compte et a ensuite accédé aux paramètres de gestion des accès. Il pense qu'il est suffisant de voir uniquement les personnes de sa succursale pour facilement ajouter une personne remplaçante qui pourra gérer le stock en son absence, plutôt que d'avoir la liste complète des utilisateurs. Il suggère également d'ajouter un historique des modifications pour savoir qui a ajouté, modifié ou supprimé un élément d'une liste, afin d'éviter de faire le travail en double.

Feedback général

Qu'avez-vous le plus apprécié dans ce prototype ?

Les options de sélection, le processus d'ajout des produits et le fait d'avoir les images des produits.

Quelles sont les principales difficultés que vous avez rencontrées ?

Pas grand chose, cela ressemble à galaxus alors on retrouve vite nos marques.

Avez-vous des suggestions d'améliorations pour ce prototype ?

Il serait intéressant d'ajouter un petit pup-up sur le bouton automatique pour dire de modifier les réglages de sélection avant de continuer. Il peut être utile de voir le stock des autres succursales avec une option de comparaison entre 2 succursales pour voir quels produits sont différents et se donner des idées dans la vue du stock actuel.

Sur une échelle de 1 à 10, comment évalueriez-vous l'utilisabilité de cette interface ?

9 sur 10

Personne 4

Département	Retail
Fonction	Store Manager
Emplacement	Dietikon
Date	08.07.2024
Lieu du test	A distance via Teams
Langue du test	Anglais

Scénario 1 : Connexion et accès au tableau de bord

Situation	Vous souhaitez définir le stock cible de la saison à venir et vous vous rendez donc sur l'outil dédié. Vous vous retrouvez premièrement sur la page de connexion.

Tâche	Connectez-vous et accédez au tableau de bord.

Les informations du tableau de bord sont-elles claires et compréhensibles ?

> Oui, elles sont simples et faciles de compréhension.

Quelles informations souhaitez-vous voir sur ce tableau de bord ?

> Il propose d'ajouter la possibilité de choisir entre "nouveauté" et "meilleure vente" pour l'affichage principal de cette donnée, avec un classement des meilleures ventes en cliquant sur la case. Un graphique montrant l'évolution des ventes pour un produit serait également un plus.

Observations et recommandations

> Le participant est arrivé rapidement sur le tableau de bord et a pu le parcourir facilement. Il propose plusieurs améliorations fonctionnelles, comme la possibilité d'ajouter du temps automatiquement lorsque le compteur arrive proche de zéro, avec des rappels pour le temps restant. Il suggère également la possibilité de voir les différents stocks, notamment entre les stocks saisonniers et annuels.

Scénario 2 : Choix de la succursale et de la langue

Situation	Avant de commencer à gérer le stock cible, vous souhaitez définir quelques paramètres de base comme la langue et votre succursale notamment.
Tâche	Trouvez l'option permettant de modifier la succursale et la langue.

L'option est-elle facilement reconnaissable ?

> Oui mais l'option fait davantage référence à la langue, peut-être faudrait-il avoir un icône différent pour la succursale.

Le fonctionnement de l'option est-il intuitif ?

> Oui

Observations et recommandations

> Il a parcouru la page pour trouver l'option et a d'abord cliqué sur la localisation dans le pied de page, ce qui l'a ramené vers le stock actuel. Par la suite, il a survolé d'autres éléments avant de trouver l'option. Il recommande de prendre exemple sur la sélection de la langue sur digitec.ch pour l'appliquer à la solution du stock cible et de séparer la localisation.

Scénario 3 : Sélection des produits pour le stock cible

Situation	Vous souhaitez à présent générer une liste de produits à destination du stock cible et souhaitez que le système définisse automatiquement la liste.
Tâche	Suivez le processus pour générer une liste de produits automatiquement.

Le processus de génération est-il compréhensible ?

> Oui très facile

La disposition des produits est-elle agréable ?

> Oui mais il faudrait peut-être avoir plus de produits par ligne, 4 par exemple, et avoir le choix du nombre de produits par ligne.

Observations et recommandations

> Le participant a immédiatement compris où aller pour commencer une liste automatiquement et a cliqué sur le bon bouton pour arriver sur la page affichant les produits. Il a simplement suggéré d'afficher plus de produits par ligne pour réduire la longueur de la liste, tout en laissant à l'utilisateur le choix de la disposition qu'il préfère.

Scénario 4 : Modification des règles pour le stock cible

Situation	Maintenant que vous avez votre liste de produits, vous souhaitez définir les règles qui vont paramétrer la disponibilité du stock cible.
Tâche	Modifiez les règles (succursale, dates et quantité) pour tous les produits puis pour un produit.

Avez-vous trouvé facilement où modifier les règles ? (Global et individuel)

> Oui dans les 2 cas.

L'éventail d'options pour les règles est-il complet ?

> Oui, pour les paramètres globaux, mais la quantité par produit pourrait être simplement indicative, avec une quantité par défaut de 1. Pour des quantités plus importantes, il serait nécessaire de passer par les produits individuellement.

Observations et recommandations

> Il a rapidement trouvé l'option pour les règles globales du stock cible et a pu les modifier aisément. Il a fait une remarque concernant la définition de la quantité. Par défaut, cette dernière est de 1 pour tous les produits, mais une note pourrait indiquer que pour définir une quantité plus élevée, il faut passer par les réglages individuels ou groupés. Cela éviterait également les erreurs en redéfinissant une quantité pour tous les produits. Ensuite, il a trouvé l'option pour gérer individuellement les règles. Selon lui, la période de disponibilité n'est pas utile ici, car tous les produits ont la même. Il suggère que la quantité pourrait être directement ajustée sur le produit lors du survol, avec un simple "+" et "-" pour définir cette dernière.

Scénario 5 : Modification de la liste de produit

Situation	La liste de produits vous convient partiellement, vous souhaitez la modifier en ajoutant un produit et

	en supprimant d'autres.
Tâche	Ajoutez un produit puis supprimez en plusieurs.

La méthode d'ajout d'un produit est-elle intuitive ?

> Oui, il est très aisé de choisir un produit sur galaxus et de simplement copier son lien pour l'ajouter ici.

L'option vous permettant de sélectionner plusieurs produits et de les supprimer est-elle facile d'utilisation ?

> Oui, peut-être un petit guide qui explique ce qu'on peut faire lorsqu'on arrive sur la liste de produits pour comprendre directement les options à disposition.

Observations et recommandations

> Le participant a trouvé aisément les deux options permettant de réaliser cette tâche, bien qu'il n'ait pas tout de suite compris l'icône permettant la sélection multiple. Il a ensuite suggéré plusieurs améliorations. Premièrement, il serait utile de pouvoir créer une liste globale pour toutes les succursales, avec la possibilité pour chacune de définir leur liste saisonnière, ce qui impliquerait d'avoir plusieurs listes actives par succursale. Ces listes pourraient avoir un nom et être fusionnées si nécessaire. Deuxièmement, il faudrait penser à l'affichage des produits ajoutés ultérieurement, en spécifiant s'ils apparaissent au début ou à la fin de la liste.

Scénario 6 : Validation et enregistrement des produits

Situation	Vous êtes satisfait de la liste de produits et souhaitez finaliser le processus de création et d'enregistrement du stock cible.
Tâche	Continuez le processus et validez la liste de produits pour l'enregistrement dans l'ERP.

Le résumé des détails du stock cible vous semble-t-il complet ?

> Oui mais il faudrait avoir le nombre de produits total à stocker plutôt que la quantité par produit.

Le processus d'enregistrement des produits est-il clair ?

> Oui

Observations et recommandations

Il est arrivé rapidement à la prochaine étape du processus et a passé en revue le résumé des informations du stock cible. Il suggère de remplacer la quantité par produit par le total de produits à stocker. Bien que cela ne soit pas directement lié à cette tâche, il s'est interrogé sur le fonctionnement du mode automatique. Après qu'on lui ait montré l'option dédiée, il propose de la déplacer pour qu'elle apparaisse au moment de choisir entre automatique et manuel. En examinant les critères de sélection, il suggère également de séparer le critère saisonnier en deux, pour distinguer l'été et l'hiver. Enfin, il a pu terminer le processus sans problème avec l'enregistrement des produits dans l'ERP.

Scénario 7 : Consultation et modification du stock actuel

Situation	Le processus de sélection et d'enregistrement est à présent terminé mais vous venez de penser à un produit qui serait intéressant pour le stock cible. Vous souhaitez donc rajouter ce produit au stock actuel.
Tâche	Accédez à la liste de produits du stock actuel et ajoutez un nouveau produit.

La liste du stock actuel est-elle rapidement et facilement accessible ?

Oui, cela se fait rapidement via la navigation dans l'entête.

Observations et recommandations

Le participant a rapidement trouvé le stock actuel via les liens de navigation dans l'entête et a pu ajouter un produit sans difficulté. Il a ensuite fait plusieurs observations qui sont également pertinentes pour d'autres parties de la solution. Lors de la sélection des produits, il suggère qu'il soit nécessaire d'avoir au minimum 20 pièces en entrepôt pour les acheminer dans les succursales sans pénaliser les commandes via expédition. Il trouverait intéressant de pouvoir identifier rapidement les produits EOL (End Of Life) dans les listes du stock actuel pour pouvoir les retirer facilement. Une autre fonctionnalité utile serait d'avoir une suggestion de produit similaire lors de la suppression d'un produit, afin de le remplacer facilement. Enfin, il a questionné le fonctionnement des modifications apportées au stock actuel, à savoir si elles sont effectuées en temps réel ou de manière groupée dans l'ERP. Ce dernier point devra être défini par la suite.

Scénario 8 : Gestion des accès à l'application

Situation	En tant que super utilisateur, vous pouvez gérer les accès à l'application et vous souhaitez ajouter un collègue pour qu'il puisse également gérer le stock cible. (L'accès est premièrement limité par l'utilisation de la connexion avec Microsoft puis en vérifiant qu'il s'agisse bien d'une adresse e-mail de l'entreprise).
Tâche	Accédez aux paramètres d'accès à l'application et ajouter un utilisateur.

Avez-vous trouvé facilement où modifier les accès à l'application ?

Oui

Trouvez pertinent de limiter l'accès en plus des limitations déjà en place ?

Cela correspond à la manière dont le stock cible est géré actuellement : une personne s'occupe de la liste générale et les autres peuvent ensuite gérer leur stock saisonnier individuellement.

Observations et recommandations

Il a rapidement trouvé l'option car il s'agissait du dernier icône sur lequel il n'avait pas cliqué. Par la suite, il a pu apporter quelques éclaircissements sur la façon dont l'accès pourrait être géré. Il propose d'avoir des administrateurs qui peuvent modifier et créer le stock pour l'ensemble des succursales, et des utilisateurs limités à la gestion du stock de leur propre succursale. Cela est déjà le cas ici mais il faudrait en plus rajouter une option chez les utilisateurs pour les restreindre à leur succursale.

Feedback général

Qu'avez-vous le plus apprécié dans ce prototype ?

Le design et le minimalisme, il s'agit d'une très bonne base pour ce genre d'outil.

Quelles sont les principales difficultés que vous avez rencontrées ?

Quelques incompréhensions à certains moment sur ce qu'il faut ou ce que l'on peut faire mais c'est assez minime.

Avez-vous des suggestions d'améliorations pour ce prototype ?

Tout a été dit précédemment.

Sur une échelle de 1 à 10, comment évalueriez-vous l'utilisabilité de cette interface ?

9 sur 10

Annexe 3

Liste de recommandations

Recommandations

Afin d'optimiser et de poursuivre le développement de la solution pour la gestion du stock cible chez Digitec Galaxus AG, deux pistes principales sont proposées, chacune adaptée à des ressources et des capacités différentes.

Approche avec des ressources limitées

1. **Continuer le projet dans son état actuel :**

   - **Optimisation progressive** : Poursuivre l'amélioration des fonctionnalités existantes en résolvant les problèmes identifiés et en affinant les processus actuels. L'objectif est de parvenir à une solution fonctionnelle malgré les contraintes de ressources.
   - **Automatisation partielle** : Introduire des scripts et des outils simples pour automatiser certaines parties du processus de sélection des produits, tout en gardant une surveillance humaine pour les ajustements nécessaires.
   - **Utilisation des données existantes** : Exploiter les données de ventes actuelles et les retours des utilisateurs pour affiner la sélection des produits de manière plus précise.
   - **Formation du personnel** : Former les employés à l'utilisation des nouveaux outils et processus pour maximiser leur efficacité et réduire les erreurs humaines.

2. **Amélioration continue de l'interface utilisateur :**

   - **Feedback utilisateur** : Recueillir régulièrement des retours d'expérience des utilisateurs finaux pour identifier les points d'amélioration de l'interface.
   - **Mises à jour itératives** : Apporter des améliorations continues à l'interface en se basant sur les feedbacks, tout en maintenant une documentation claire et accessible.

Approche avec des ressources étendues

1. **Développement de fonctionnalités avancées et innovantes :**

   - **Intégration de nouvelles technologies** : Intégrer des algorithmes de machine learning pour analyser les données de ventes et de tendances de manière plus sophistiquée, permettant une sélection automatisée plus précise et proactive des produits.
   - **Infrastructure cloud** : Migrer les processus vers des solutions cloud pour améliorer la scalabilité et la flexibilité du système, facilitant l'accès et la gestion des données en temps réel.
   - **Intégration directe à l'ERP** : Connecter la solution directement à l'ERP de l'entreprise pour améliorer l'efficacité et simplifier l'accès aux données.

2. **Expansion des capacités de l'équipe :**

   - **Mobilisation des ressources internes** : Impliquer plusieurs secteurs de l'entreprise dans le projet, notamment les développeurs et les designers UX/UI existants.
   - **Collaboration interdisciplinaire** : Promouvoir une collaboration étroite entre les départements (ventes, IT, logistique) pour s'assurer que tous les aspects du processus de gestion des stocks sont pris en compte et optimisés.

3. **Tests et validation :**

   - **Tests utilisateurs approfondis** : Mener des sessions de tests utilisateurs plus approfondies avec un panel plus large pour valider les nouvelles fonctionnalités et les améliorations apportées.
   - **Itérations rapides** : Adopter une approche itérative avec des cycles de développement courts pour intégrer rapidement les retours et ajuster la solution en conséquence.

Conclusion

Ces recommandations visent à fournir un cadre structuré pour les développements futurs, en tenant compte des différentes contraintes de ressources et des objectifs de l'entreprise. En adoptant ces approches, Digitec Galaxus AG pourra non seulement améliorer l'efficacité de la gestion des stocks cibles, mais également se doter d'un système robuste, efficace et innovant pour répondre aux défis futurs du commerce électronique.

Annexe 4

Guide d'utilisation

Guide utilisateur

Introduction

Ce guide utilisateur est destiné à fournir des instructions détaillées sur l'utilisation de la solution automatisée pour la gestion des stocks cibles développée pour Digitec Galaxus AG. La solution permet de sélectionner automatiquement les produits selon divers critères et de les enregistrer dans l'ERP de l'entreprise, simplifiant ainsi le processus et réduisant les erreurs humaines.

On distingue deux types d'utilisateurs pour cette solution : l'utilisateur "standard" qui pourra gérer le stock cible de sa succursale uniquement, et l'administrateur qui a accès à toutes les succursales et à la gestion des accès de la solution.

Le développement d'un prototype n'ayant pas abouti, ce guide se base sur la version définitive de la maquette interactive sur Figma. À noter que certaines fonctionnalités ne sont pas visibles sur certains éléments si des éléments identiques ont déjà ces mêmes fonctionnalités.

Accès à l'application

Pour accéder au prototype de l'application sur Figma, il vous suffit d'ouvrir le lien suivant qui vous amènera sur le site de Figma et chargera le prototype. La première page est une page de connexion où, dans le cas du prototype, il faut simplement appuyer sur le bouton « Next » pour continuer et accéder aux différentes pages.

Lien du prototype

Navigation dans l'interface

L'interface utilisateur est conçue pour être intuitive et facile à naviguer. Voici les principales sections de l'application :

- **Tableau de bord (Onglet « Dashboard ») :** Affiche un aperçu général des informations clés sur le stock cible, y compris les produits les plus vendus, le temps restant, et plus encore.
- **Nouvelle liste de stock cible (Onglet « New stock ») :** Permet de créer une nouvelle liste pour le stock cible de manière automatique ou manuelle.
- **Gestion des listes actuelles (Onglet « Stock lists ») :** Affiche la liste des différentes listes du stock cible et permet de les ajuster au besoin.
- **Paramètres (Icône d'utilisateur) :** Permet de gérer les accès à l'application et d'accéder à un historique des modifications apportées aux différentes listes.

Préférences personnelles

Avant de commencer à utiliser pleinement la solution, quelques réglages facilitant l'expérience peuvent être définis :

- **Succursale :** En tant qu'administrateur, il est possible de choisir la succursale utilisée pour afficher le tableau de bord via la liste déroulante présente à droite dans l'entête. L'utilisateur « standard » verra simplement sa succursale.

- **Langue :** Via l'autre liste déroulante présente dans l'entête, il est possible de choisir la langue à utiliser pour les différents textes de la solution.
- **Thème :** Depuis le bas de page, il est possible de définir le thème à utiliser pour la solution. Trois options sont proposées : clair, sombre ou automatique. Cette option est disponible dans la liste déroulante tout à droite du pied de page.

Page du tableau de bord

Pour accéder à cette page, cliquez sur le lien « Dashboard » dans la navigation au milieu de l'entête.

Depuis cette page, il est possible de consulter les informations suivantes :

- **Temps restant** : À l'aide des boutons en forme de triangle, il est possible de faire défiler les différentes listes actives relatives à la succursale de l'utilisateur. Sur chacune de ces listes, on peut voir son titre, sa date de fin et le temps restant en nombre de jours.
- **Niveau de produits** : Une jauge indique le nombre de produits différents qu'il reste dans l'espace de stockage sur le total de ces produits.
- **Niveau du stock** : Une jauge indique le nombre de pièces qu'il reste dans l'espace de stockage sur le total de ces pièces.
- **Classement des meilleures ventes** : Ce classement liste les produits parmi ceux présents dans l'intégralité du stock cible de la succursale de l'utilisateur qui se vendent le mieux. Le classement montre d'abord les cinq premiers éléments avec pour chacun leur place dans le classement, leur marque et leur nom. En cliquant sur le bouton avec quatre flèches, il est possible d'avoir plus de détails via une fenêtre qui s'ouvre. Cette dernière affichera le classement étendu jusqu'à la 15e place. En plus des détails déjà présents avant, il y a les caractéristiques des produits visibles et également un bouton permettant d'ouvrir la page du produit sur Galaxus ou l'ERP.
- **Classement des moins bonnes ventes** : Ce classement reprend exactement le même principe que le précédent en termes d'informations et de fonctionnalités. La seule différence est qu'il s'agit d'un classement de produits se vendant le moins bien.

Page de création d'une nouvelle liste

Pour se rendre sur la page permettant de définir une nouvelle liste pour le stock cible, cliquez sur le lien « New stock » dans la navigation au milieu de l'entête.

Cette page commence par introduire via deux courts paragraphes en quoi elle consiste. Elle accompagne l'utilisateur dans la création d'une nouvelle liste pour le stock cible.

Tout au long du processus, il est possible de visualiser l'état d'avancement via l'indicateur dédié en haut à droite de la page. Une icône d'information *i* permet d'en apprendre plus sur les différents statuts d'étapes du processus.

Le choix à faire sur cette page est de savoir si l'on souhaite que le système génère une liste de produits à partir de différents critères avec le mode automatique ou si l'on préfère définir soi-même la liste de produits avec le mode manuel.

1. Mode automatique

Avec ce mode, le système va commencer par demander à l'utilisateur quelques informations sur la façon dont il va générer la liste de produits. Cela commence par la saisie du nombre de produits différents souhaités, puis par les critères de sélection comme le nombre de ventes, l'aspect saisonnier ou encore les nouveautés, pour finir par les catégories à inclure dans la génération.

Une fois cela défini, le processus de génération va commencer et après un certain temps, une liste de produits va apparaître. Il est possible de parcourir cette liste en la faisant défiler, puis à l'aide du système de pagination, il est possible d'accéder à la suite de la liste.

À ce stade du processus, il est possible de réaliser différentes actions sur la page :

- **Ajouter un produit :** Via le bouton d'ajout + situé dans la barre d'outils au-dessus de la liste de produits, il est possible d'ajouter un produit à la liste en indiquant son lien Galaxus. À chaque lien ajouté, un deuxième champ apparaît pour ajouter plusieurs produits à la suite. Un message de confirmation apparaîtra après avoir sauvegardé les produits.
- **Supprimer un produit :** Les options individuelles apparaissent au survol d'un produit. Parmi ces dernières, il y a la suppression du produit. Un clic sur le bouton de corbeille et le produit est supprimé. Un message de confirmation apparaîtra après la suppression avec la possibilité d'annuler l'action avec le bouton « Undo ».
- **Sélectionner plusieurs produits :** Via le bouton de sélection multiple situé dans la barre d'outils, l'affichage va basculer en mode sélection. Il sera ainsi possible de cliquer sur un produit pour le sélectionner.
- **Sélectionner tous les produits :** Une fois en mode sélection, un clic sur le bouton permet de sélectionner tous les produits de la liste.
- **Supprimer une sélection de produits :** Toujours en mode sélection, il est possible de supprimer les produits sélectionnés via le bouton de corbeille.
- **Trier la liste :** Depuis la vue normale de la liste, il est possible de modifier l'ordre d'affichage des produits via la liste déroulante à droite dans la barre d'outils. Plusieurs choix s'offrent alors comme un tri par prix ou alphabétique, par exemple.
- **Changer la mise en page :** Tout à droite de la barre d'outils se trouve l'option permettant de passer d'un affichage en cases à un affichage en liste pour les produits, et vice-versa.
- **Ouvrir la page du produit :** Un clic sur le bouton affiché au survol de chaque produit permet d'ouvrir une petite fenêtre demandant sur quelle plateforme ouvrir la page du produit en question, Galaxus ou l'ERP.

Hormis ces actions qui sont toutes facultatives, il y a un paramètre à définir avant de pouvoir continuer dans le processus. Il s'agit des règles du stock cible et ces dernières contiennent entre autres la période durant laquelle les produits seront disponibles en

succursale mais également la quantité par produit qu'il faudra stocker. Pour définir ce paramètre, il faut cliquer sur l'icône d'engrenage situé dans la barre d'outils et une fenêtre va s'ouvrir en proposant de remplir les champs suivants :

- **Titre :** Permet de différencier cette liste des autres plus facilement.
- **Succursale :** Permet de définir dans quelle succursale la liste va se retrouver. Cette option est uniquement disponible en tant qu'administrateur.
- **Date de début :** Indique à partir de quand les produits seront disponibles dans le stock cible.
- **Date de fin :** Indique jusqu'à quand les produits seront disponibles dans le stock cible.
- **Quantité par produit :** Définit la quantité à stocker pour chaque produit. Il est recommandé de laisser cette quantité sur la valeur par défaut, à savoir 1.

Ces paramètres sont appliqués à l'ensemble des produits de la liste. Il n'y a que la quantité qui peut être définie individuellement ou pour une sélection de produits par la suite. Pour ce faire, il faut :

- **Pour la quantité individuelle :** Survoler un produit et une liste déroulante 1 apparaîtra aux côtés des autres outils. Si la quantité souhaitée est supérieure à 5, un champ apparaîtra pour permettre une saisie manuelle de la quantité.
- **Pour la quantité d'une sélection :** Cliquer sur le bouton de crayon lors du mode de sélection multiple et définir la quantité désirée dans la fenêtre qui vient de s'ouvrir. La saisie peut se faire via le champ ou à l'aide des boutons « + » et « - ».

Une fois ces paramètres définis, il sera possible de continuer le processus de création d'une liste pour le stock cible à l'aide du bouton « Next » situé juste en dessous de l'indicateur d'avancement du processus.

2. Mode manuel

Le mode manuel reprend exactement les mêmes fonctionnalités que le mode automatique. La seule différence ici est que le système ne va pas générer de liste étant donné que c'est à l'utilisateur de la créer de A à Z.

Une fois tous les produits ajoutés et les règles du stock cible définies, il sera possible de continuer dans le processus en cliquant sur le bouton « Next ».

La suite du processus de création d'une liste pour le stock cible est très simple et commune aux deux modes. Après avoir cliqué sur le bouton « Next », un résumé des détails de la liste apparaîtra. Il suffit de les passer en revue puis de continuer le processus si tout est correct. En cas d'erreur, il est toujours possible de retourner en arrière à ce stade.

Un chargement va alors démarrer pour indiquer que les produits vont être enregistrés dans l'ERP. Cela peut prendre plus ou moins de temps selon la quantité de produits à traiter. Une fois le processus d'enregistrement terminé, un message de succès apparaîtra indiquant à l'utilisateur que tous les produits ont bien été enregistrés dans l'ERP. En cas de problème, un message

d'erreur viendra indiquer à l'utilisateur qu'il y a eu une erreur lors du processus et qu'il a la possibilité de relancer l'enregistrement.

Page des listes actuelles

Pour se rendre sur la page listant les différentes listes du stock cible, il suffit de cliquer sur le lien « Stock lists » dans la navigation au milieu de l'entête.

Dans cette page, toutes les listes relatives à la succursale de l'utilisateur sont affichées. Pour chacune de ces listes, il y a le titre, la succursale, la période de disponibilité, la quantité de produits différents et un statut. Ce dernier indique à l'utilisateur si la liste est à l'état de brouillon, si elle est active, expirée, voire supprimée. Cela permet de garder une trace de tout ce qui a été créé. Les listes peuvent être parcourues à l'aide du système de pagination.

De plus, il est possible d'effectuer différentes actions sur chaque liste :

- **Supprimer :** Il est possible de supprimer la liste et cette dernière verra alors son statut changer pour passer à « Supprimé ». Cela implique que la liste n'est plus active et donc plus disponible. Toutefois, elle reste quand même affichée dans la liste.
- **Dupliquer :** Il est possible de dupliquer une liste pour lui attribuer une nouvelle période de disponibilité et un nouveau titre si besoin. Cela créera une copie de la liste pour la succursale de l'utilisateur avec de nouvelles dates et un nouveau titre.

En tant qu'administrateur, il est possible, en plus des options précitées, de voir les listes de toutes les succursales mais également de supprimer et dupliquer ces dernières peu importe la succursale. En outre, la fonction de duplication gagne la possibilité de choisir une succursale pour la copie de la liste.

Enfin, il est possible d'accéder à chaque liste en cliquant dessus. L'affichage et les options disponibles sont les mêmes que lors de l'affichage des produits lors de la création d'une liste.

Paramètres

Quelques pages sont disponibles à travers l'icône d'utilisateur dans l'entête. En cliquant sur ce dernier, un petit menu s'ouvre et permet d'accéder à deux pages en plus de permettre la déconnexion de la solution :

- **Gestion des accès :** En tant qu'administrateur, il est possible d'ajouter ou de supprimer des personnes pour l'accès à la solution. Lors de l'ajout, il est demandé de saisir un nom, un prénom, une adresse e-mail et la succursale à laquelle le futur utilisateur sera rattaché. Il est également possible d'attribuer des droits d'administrateur à d'autres utilisateurs. Ces derniers ne verront que la liste des autres utilisateurs en se rendant sur cette page et n'ont donc pas accès à l'ajout ni à la suppression d'utilisateurs.
- **Historique des modifications :** Afin d'avoir un suivi de ce qui se fait sur la solution, une page affichant un historique est disponible. En fonction des accès de l'utilisateur, ce dernier pourra voir uniquement les modifications relatives aux listes de sa succursale ou de toutes ces dernières s'il est administrateur. La liste des modifications se compose pour chacun de ces éléments d'une date, d'un utilisateur, du titre de la liste, de la succursale en question et enfin de la nature de la modification. Les modifications peuvent être parcourues à l'aide du système de pagination.

Conclusion

Ce guide utilisateur vous offre une vue d'ensemble des principales fonctionnalités de la solution automatisée pour la gestion des stocks cibles. En suivant ces instructions, vous pourrez naviguer efficacement dans l'application, optimiser la gestion des stocks et réduire les erreurs.

Annexe 5

Rapport de la pré-étude

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Pré-étude

Automatisation du stock cible des succursales de Digitec Galaxus AG

Étudiant·e :	Nicolas Meuwly
Travail proposé par :	Steve Cosandey
	Digitec Galaxus AG
	Rue du Grand-Pré 2B
	1007 Lausanne
Enseignant responsable :	Laurent Bolli
Année académique :	2023-2024
Yverdon-les-Bains, le	23.05.2024

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Authentification

J'atteste par la présente avoir réalisé ce travail et n'avoir utilisé aucune autre source que celles expressément mentionnées.

Nicolas Meuwly

Signature

N. Meuwly

Yverdon-les-Bains, le 23.05.2024

Table des matières

1. Présentation de l'entreprise et du secteur d'activité 6
2. Présentation de la problématique et des besoins 6
3. Recherche documentaire et recherche utilisateurs/clients 7
3.1. Analyse de l'existant 8
3.1.1. Sélection des articles 8
3.1.2. Enregistrement des produits dans le système 8
3.2. Besoins des utilisateurs 9
3.2.1. Validation du sondage 10
3.2.2. Résultats de la question 1 11
3.2.3. Résultats de la question 2 11
3.2.4. Résultats de la question 3 12
3.2.5. Résultats de la question 4 12
3.2.6. Résultats de la question 5 13
3.2.7. Résultats de la question 6 14
3.2.8. Résultats de la question 7 15
3.2.9. Résultats de la question 8 16
3.2.10. Résultats de la question 9 16
3.2.11. Conclusion du sondage 16
3.3. Faisabilité technique 17
3.3.1. Machine Learning 17
3.3.2. Web Scraping 17
3.3.3. Utilisation des outils de Google 20
3.3.4. Utilisation des catégories phares 22
3.3.5. Utilisation de l'IA 23
3.4. Accès aux bases de données et à l'ERP 24
3.5. Choix de l'environnement de développement 24
3.5.1. Framework des portails de vente 24
3.5.2. Environnement alternatif 27

3.6. Conclusion 29
4. Analyse et description des objectifs et des besoins 29
4.1. Besoins 29
4.1.1. Besoins fonctionnels 30
4.1.2. Besoins des utilisateurs 30
4.2. Objectifs 30
4.2.1. Automatisation complète du processus 30
4.2.2. Simplification et accessibilité de la solution 30
4.2.3. Optimisation du temps et de l'efficacité 30
5. Organisation et description des tâches et des livrables 31
5.1. Découpage en phases 31
5.1.1. Phase 1 : Conception de la solution 31
5.1.2. Phase 2 : Développement des fonctionnalités 31
5.1.3. Phase 3 : Intégration des fonctionnalités 31
5.1.4. Phase 4 : Tests et optimisation 31
5.1.5. Phase 5 : Finalisation et recommandations 32
5.2. Livrables 32
6. Planning 33
7. Gestion des risques 34
7.1. Risques et mesures 34
7.2. Matrice des risques 35
8. La conclusion 35
Glossaire 37
Table des abréviations et acronymes 41
Bibliographie 42
Annexes 48

Table des figures

Figure 1 : Résultats et graphique de la question 1 du sondage réalisé sur Microsoft Forms 11
Figure 2 : Résultats et graphique de la question 2 du sondage réalisé sur Microsoft Forms 11
Figure 3 : Résultats et graphique de la question 3 du sondage réalisé sur Microsoft Forms 12
Figure 4 : Résultats et graphique de la question 4 du sondage réalisé sur Microsoft Forms 12
Figure 5 : Classement des réponses de la question 4 du sondage réalisé sur Microsoft Forms 13
Figure 6 : Résultats de la question 5 du sondage réalisé sur Microsoft Forms 13
Figure 7 : Détails des résultats de la question 5 du sondage réalisé sur Microsoft Forms 13
Figure 8 : Résultats et graphique de la question 6 du sondage réalisé sur Microsoft Forms 14
Figure 9 : Résultats et graphique de la question 7 du sondage réalisé sur Microsoft Forms 15
Figure 10 : Résultats de la question 8 du sondage réalisé sur Microsoft Forms 16
Figure 11 : Résultats de la question 9 du sondage réalisé sur Microsoft Forms 16
Figure 12 - Exemple de tableau en HTML (W3Schools) 18
Figure 13 : Liste des produits de la catégorie « Téléphone portable » 19
Figure 14 : Construction de l'URL pour l'accès aux listes de produits sur galaxus.ch 19
Figure 15 : Documentation sur l'utilisation de l'API Google Trends 21
Figure 16 : Liste des prix de SerpApi.com 21
Figure 17 : Résultats Google Trends sur le thème « Informatique et électronique » avec Google Shopping 22
Figure 18 : Construction de l'URL pour l'accès aux données de Google Trends 22
Figure 19 : Page d'accueil de galaxus.ch avec les catégories phares 23
Figure 20 : Analyse technologique de galaxus.ch réalisée à l'aide de l'outil de wappalyzer.com (1/2) 25
Figure 21 : Analyse technologique de galaxus.ch réalisée à l'aide de l'outil de wappalyzer.com (2/2) 26
Figure 22 : Analyse technologique de l'ERP réalisée à l'aide de l'outil de wappalyzer.com 28
Figure 23 : Planification du projet au format Gantt Chart 33
Figure 24 : Tableau des risques et mesures 34
Figure 25 : Matrice des risques 35

1. Présentation de l'entreprise et du secteur d'activité

Digitec Galaxus AG est une entreprise suisse de commerce électronique fondée en 2001. Elle gère deux principales plateformes de vente en ligne : Digitec et Galaxus. Digitec se concentre sur les produits électroniques, informatiques et de télécommunications, proposant des articles tels que des ordinateurs, des smartphones, des composants informatiques et des équipements audiovisuels. Galaxus, en plus d'inclure l'assortiment de Digitec, offre une variété de produits couvrant plusieurs catégories, notamment la maison, le jardin, la mode, les loisirs, le sport et la beauté. Galaxus sert également de plateforme globale regroupant toutes les données du client. Ce dernier peut ainsi retrouver tout ce qui touche à son compte comme ses commandes ou ses factures provenant des 2 plateformes de vente.

En 2018, l'entreprise a commencé son expansion en Europe en lançant Galaxus en Allemagne et elle est maintenant présente dans six pays[1] en dehors de la Suisse.

L'entreprise se positionne comme un acteur clé du commerce électronique en Suisse, ayant su étendre son catalogue et ses services pour répondre aux besoins variés des consommateurs. Digitec Galaxus AG se distingue par son approche de distribution en ligne, permettant une large accessibilité à ses produits grâce à une interface utilisateur optimisée et une logistique efficace. Les deux plateformes proposent également des services de conseil et de support client, contribuant à améliorer l'expérience d'achat des utilisateurs.

En termes de structure, Digitec Galaxus AG appartient majoritairement au groupe Migros, ce qui lui permet de bénéficier d'un soutien solide et de synergies au sein du groupe. L'entreprise a su développer une stratégie multicanale en intégrant des magasins physiques en Suisse, offrant ainsi aux clients la possibilité de retirer leurs commandes en ligne en succursale, et de bénéficier d'un service après-vente accessible.

Digitec Galaxus AG est également engagée dans l'innovation technologique et logistique, investissant continuellement dans l'amélioration de ses infrastructures et processus pour optimiser les délais de livraison et la satisfaction client. Cette approche lui permet de maintenir une position compétitive sur le marché suisse du commerce électronique, caractérisé par une forte demande pour des solutions d'achat en ligne fiables et efficaces.

2. Présentation de la problématique et des besoins

Chaque succursale de Digitec Galaxus AG dispose d'un espace de stockage destiné aux commandes des clients ainsi qu'à des produits disponibles à l'achat. Parmi ces derniers, on trouve des produits issus de commandes annulées et des produits faisant partie du « stock cible ». Ce stock comprend des articles saisonniers, tels que des ventilateurs en été ou des couvertures chauffantes en hiver, ainsi que des articles régulièrement commandés et des nouveautés. Actuellement, il est possible de trouver des iPhone 15, des

[1] Galaxus est présent en Allemagne, Autriche, France, Italie, Pays-Bas et Belgique en dehors de la Suisse.

coques de protection, des LEGO, des couches pour bébé, de la lessive, et bien plus encore. Le stock cible permet de proposer un assortiment de produits disponibles en permanence, renouvelé automatiquement dès qu'un article est réservé. Ce stock est important car il augmente l'attrait des succursales. Les clients peuvent non seulement venir récupérer leurs commandes, mais aussi acheter directement sur place et retirer leurs achats le jour même.

Pour déterminer les articles à inclure dans le stock cible, une ou plusieurs personnes de l'entreprise établissent une liste des produits à inclure. Comme mentionné précédemment, elles examinent les articles fréquemment commandés, les nouveautés du moment et les articles saisonniers pertinents. Une liste est ensuite établie avec tous les articles à inclure dans le stock cible pour la saison à venir.

Une fois la liste établie, il est nécessaire d'accéder à la page produit de chaque article dans l'ERP[2] pour les définir en tant que stock cible pour chaque succursale. Pour chaque produit, une quantité, une date de début et une date de fin doivent être spécifiées. Jusqu'à présent, ce processus est en partie manuel, nécessitant l'accès à chaque page produit et l'exécution d'une suite d'actions pour chaque article de la liste. Bien qu'un script[3] ait été développé pour automatiser ce processus, il nécessite des prérequis techniques spécifiques et ne peut donc pas être utilisé par tout le monde. En effet, le script requiert l'installation de Python[4], de plusieurs dépendances et la configuration de divers éléments sur l'ordinateur, ce qui le rend difficilement utilisable tel quel.

Comme illustré, le processus de détermination et d'enregistrement du stock cible est long, répétitif et ne peut pas être réalisé par n'importe qui en raison de certaines contraintes techniques. Bien que cette tâche ne doive être réalisée que deux fois par an (été et hiver), elle n'est pas optimale. Mon travail consistera donc à trouver et développer une solution pour automatiser entièrement le processus. La solution devra être capable de générer une liste d'articles en fonction de plusieurs critères et d'enregistrer chaque article de la liste dans l'ERP de l'entreprise. Le produit final devra également être accessible et agréable à utiliser, afin que quiconque puisse l'utiliser sans prérequis techniques.

3. Recherche documentaire et recherche utilisateurs/clients

Dans un premier temps, il est utile d'effectuer une recherche détaillée sur le processus actuel pour la création et gestion du stock cible. Ce processus se séparant en 2 points, la sélection des produits et leur enregistrement dans le système, il est tout naturel que les 2 premiers points de cette section leur soient dédiés. Par la suite, je vais traiter de recherches portant sur la future solution apportée par mon Travail de

[2] Un ERP (Enterprise Resource Planning) est un système de gestion des ressources d'une entreprise.
[3] Un script est un fichier contenant une série d'instructions exécutées par un programme.
[4] Python est un langage de programmation interprété, multiparadigme et multiplateformes. (Python (langage) – Wikipédia, 2024)

Bachelor. Étant employé par l'entreprise à temps partiel depuis plus de 3 ans à la succursale de Lausanne, le système, les processus internes ainsi que la problématique en elle-même me sont déjà familiers.

3.1. Analyse de l'existant

3.1.1. Sélection des articles

Un fichier Excel est utilisé pour regrouper tous les articles à inclure dans le stock pour la saison à venir. Les produits sont sélectionnés manuellement par une personne et incluent des produits issus des meilleures ventes mais également des produits essentiels au stock cible. Ces produits essentiels peuvent varier en fonction des saisons et pourraient être des ventilateurs en été ou des couvertures en hiver. Un humain doit donc intervenir et déterminer une liste en fonction de données et de son expérience dans le métier. La liste doit également tenir compte des nouveautés et des tendances du moment.

Cette tâche prend un temps considérable pour rassembler tous ces articles et obtenir une liste de plus de 200 articles. Elle doit en principe être faite 2 fois par année, pour l'été et l'hiver. Cependant, elle peut évoluer au cours d'une période pour intégrer des nouveautés ou des tendances du moment. Cela demande donc une certaine surveillance du marché et d'une expérience dans le métier suffisante pour ajouter les bons produits.

3.1.2. Enregistrement des produits dans le système

Un script en Python avait été développé par un ancien employé pour automatiser l'enregistrement des produits dans le système pour qu'ils figurent dans le stock cible. J'ai pu avoir accès à ce script et n'ayant pas vraiment de connaissances en Python au préalable, je suis allé me renseigner sur internet sur le fonctionnement de ce langage. Heureusement pour moi, les fichiers fournis avec le script sont bien documentés donc je peux comprendre assez aisément ce qu'il permet de faire. Ce système n'est pas directement intégré à l'ERP, il s'exécute en dehors et a besoin de quelques prérequis pour fonctionner comme notamment un cookie de session[5] et un accès à la section de l'ERP permettant l'enregistrement du stock cible. Il m'a fallu un certain temps pour faire fonctionner le script en partie à la suite des nombreux problèmes que j'ai rencontrés comme des dépendances à mettre à jour ou des pilotes de navigateur introuvables. Il me manquait un pilote pour Google Chrome et il était nécessaire de mettre à jour Selenium, un outil d'automatisation des actions dans le navigateur compatible avec plusieurs langages de programmation, dont Python. Lors de mes tests, l'absence d'accès à la gestion du stock cible m'a également empêché d'aller beaucoup plus loin.

[5] Un cookie de session est un petit fichier stocké sur le navigateur qui permet de garder en mémoire la session de l'utilisateur. Cela permet de vérifier si un utilisateur est connecté ou non entre autres.

Sans rentrer dans trop de détails techniques, un premier script va commencer par ouvrir une fenêtre d'un navigateur sur l'adresse de l'ERP demandant ainsi à l'utilisateur de se connecter avec son compte Microsoft pour accéder à l'ERP. Une fois connecté, le script pourra récupérer le cookie de session de l'utilisateur. Ce cookie permet de savoir si un utilisateur est connecté ou non et va être utile pour autoriser le script principal à exécuter des actions sur l'ERP en se faisant passer pour l'utilisateur. Ce dernier a 120 secondes pour se connecter et une fois ce délai passé, le script va récupérer le cookie et le stocker dans un dossier. Si la connexion n'a pas pu être établie, le script indiquera un message d'erreur dans l'invite de commande[6] depuis lequel le script a été lancé.

Le script principal lui s'occupera d'automatiser l'enregistrement des produits dans le stock cible. Il commencera par vérifier l'existence d'un cookie dans le dossier créé précédemment et la validité du dit cookie en tentant d'accéder à l'ERP. S'il atterrit sur la page de bienvenue des utilisateurs connectés, le cookie est valide, sinon il affichera un message d'erreur indiquant que le cookie n'est pas valide et qu'il faut relancer le premier script pour récupérer le cookie. Pour ajouter chaque article dans le stock cible, le script va analyser chaque produit provenant d'une liste pour, dans un premier temps, vérifier que les articles soient bien en stock dans les principaux entrepôts. Une fois cela fait, le script va aller supprimer toutes les règles existantes concernant le stock cible afin de, par la suite, en créer de nouvelles. Ces règles comprennent la quantité du produit à stocker dans le stock cible, la date de début et la date de fin pour définir la plage de temps que le produit restera dans ce stock et la succursale dans laquelle il faut ajouter le produit. La liste Excel contient notamment ces éléments en plus de l'identifiant du produit. Pour terminer, le script pourra ajouter chaque produit de la liste à chaque succursale pour le stock cible.

Il s'agissait ici de la version « automatisé » de l'ajout des produits dans l'ERP. Il est bien également possible de le faire à la main en se connectant directement au système en ligne et d'ajouter manuellement un article au stock cible.

3.2. Besoins des utilisateurs

Dans le but de mieux cerner la problématique et les besoins des utilisateurs finaux, j'ai préparé un sondage à destination des chefs d'équipe de chaque succursale. En effet, ce sont eux qui s'occupent de cette thématique pour leur succursale respective. Pour presque chacune d'entre elles, il y a 3 personnes occupant des rôles différents (Supervisor, Deputy Store Manager et Store Manager). Cela représente un échantillon de 31 personnes réparties dans les 10 succursales en Suisse. Ces réponses me permettront de mettre en lumière un certain nombre de points essentiels à la poursuite de mon travail.

[6] Une invite de commande est une interface textuelle qui permet aux utilisateurs d'exécuter des commandes système en tapant des instructions directement dans un terminal ou une console.

Le sondage commence par des questions sur le fonctionnement global de la future solution comme son aspect ou la manière dont elle doit atteindre son but. Par la suite, je vais demander aux sondés les critères importants selon eux pour définir quels produits seront disponibles dans le stock cible. Il sera également question de déterminer la pertinence d'inclure le client dans le processus de sélection des produits. Lors de l'avant-dernière section du sondage, je vais pouvoir étudier les fonctionnalités attendues par les utilisateurs. Pour ce faire, j'ai imaginé un certain nombre de fonctionnalités après plusieurs jours à étudier le sujet. Les sondés vont ainsi pouvoir sélectionner celles qu'ils estiment importantes et m'en proposer d'autres s'ils le souhaitent. Pour conclure le sondage, je propose aux personnes qui le souhaitent de me laisser leur adresse e-mail afin de les contacter en cas de tests utilisateurs d'un prototype ou d'une maquette de la solution. Un exemplaire vierge du questionnaire se trouve en annexe[7].

3.2.1. Validation du sondage

Avant de pouvoir envoyer le sondage aux personnes concernées, il devait être validé par les supérieurs de l'entreprise, en particulier par le responsable de la branche Retail[8]. Ce dernier étant absent lors de la deuxième semaine de cette pré-étude, la validation n'a pu être obtenue qu'à la fin de la troisième semaine. J'ai donc dû fixer un délai relativement court pour répondre aux questions, afin d'avoir le temps de les analyser avant de rendre le présent rapport. Le mail a été envoyé le jeudi 16 mai avec une date limite pour répondre fixée au mardi 21 mai au soir, ce qui laissait peu de temps, d'autant plus qu'il y a eu un week-end prolongé entre-temps.

En date du 22 mai, un total de six réponses a été reçu. Ce nombre peut sembler faible, mais il est compréhensible compte tenu du contexte. Le sondage reste cependant ouvert afin de recueillir davantage de réponses et d'obtenir plus d'éléments pour l'élaboration de la solution. Pour l'heure, ces premières réponses fourniront déjà des informations précieuses sur les besoins des utilisateurs. Le condensé des réponses du sondage se trouve également en annexe[9].

[7] Annexe 1 : Sondage vierge
[8] Le Retail est la branche de l'entreprise qui s'occupe de la présence physique de cette dernière (les succursales).
[9] Annexe 2 : Résultats du sondage

3.2.2. Résultats de la question 1

Figure 1 : Résultats et graphique de la question 1 du sondage réalisé sur Microsoft Forms

En répondant à la première question, qui portait sur la manière de gérer le stock cible, la majorité des participants ont exprimé une préférence pour une solution dynamique. Il faudra donc permettre à l'utilisateur de choisir, dès son arrivée sur la solution, s'il souhaite définir le stock de manière globale ou par succursale. Une fois le stock cible en place, il doit également être possible de choisir où ajouter des produits supplémentaires. Cette flexibilité devrait convenir à tous les utilisateurs, offrant ainsi une solution polyvalente.

3.2.3. Résultats de la question 2

Figure 2 : Résultats et graphique de la question 2 du sondage réalisé sur Microsoft Forms

Cette question pourrait paraître inutile de prime abord, mais il ne faut pas oublier que la solution actuelle pour enregistrer les produits dans l'ERP passe par l'exécution de scripts sur un invite de commande. Bien que cette façon de faire impose quelques prérequis pour fonctionner, elle aurait pu être préférable pour certaines personnes d'un point de vue de simplicité. En une seule commande, tout pourrait se faire. Dans notre cas, il n'y a apparemment pas débat et tous préfèrent passer par une interface graphique pour

effectuer des actions plutôt qu'en tapant des commandes dans un terminal[10]. Dans un premier temps, il me faudra alors designer cette interface avant de la coder pour la solution finale. Cela passera par des maquettes et des tests utilisateurs afin de proposer l'expérience la plus optimale et agréable possible.

3.2.4. Résultats de la question 3

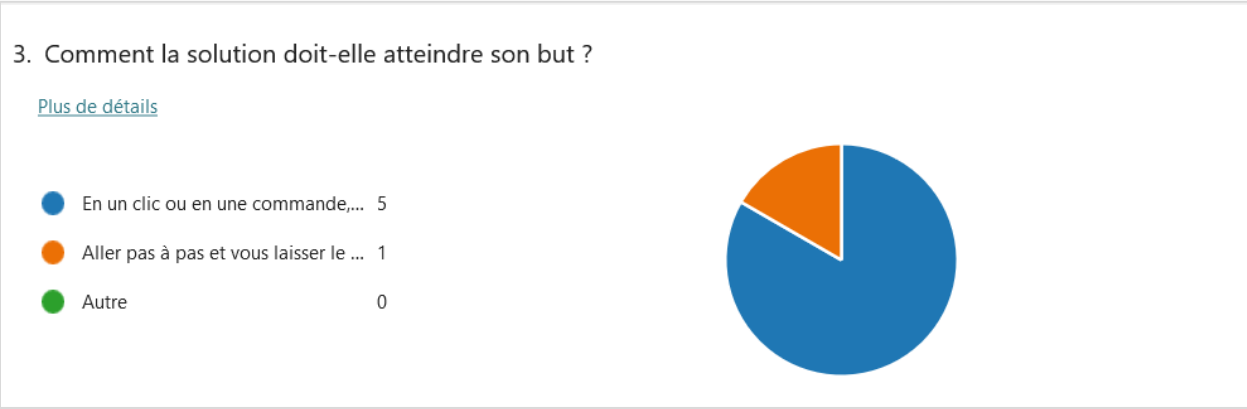


Figure 3 : Résultats et graphique de la question 3 du sondage réalisé sur Microsoft Forms

À travers cette question, on peut remarquer ici le besoin d'une solution rapide et simple. En effet, la majorité souhaite pouvoir qu'en un clic, tout se fait et il ne reste plus qu'à valider à la fin. Nous avons donc le fonctionnement global de la solution. Les questions suivantes permettront de définir les détails de ce fonctionnement.

3.2.5. Résultats de la question 4

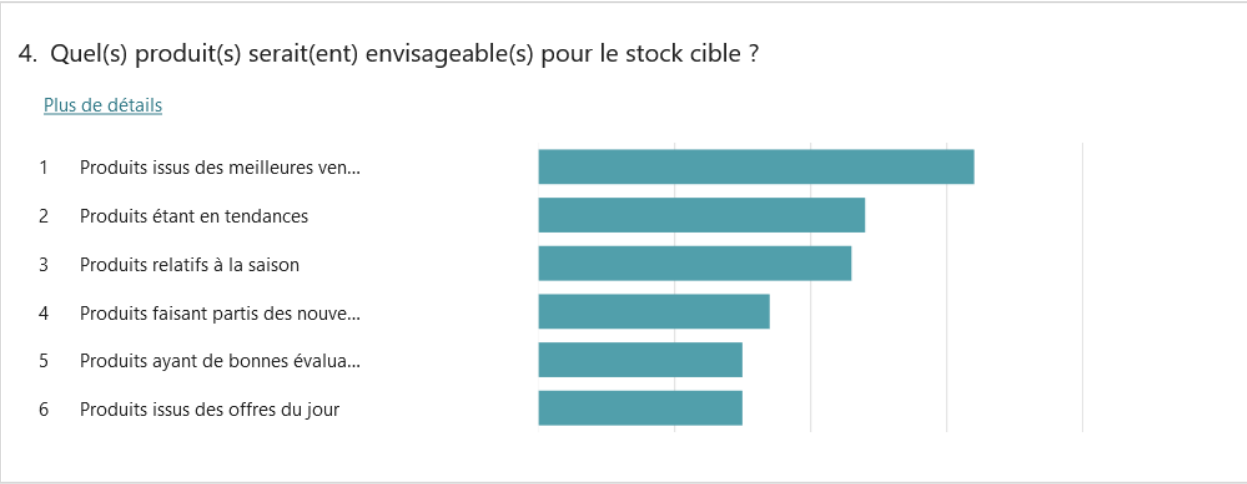


Figure 4 : Résultats et graphique de la question 4 du sondage réalisé sur Microsoft Forms

[10] Un terminal est une interface utilisateur textuelle qui permet d'interagir avec le système d'exploitation en saisissant et en exécutant des commandes.

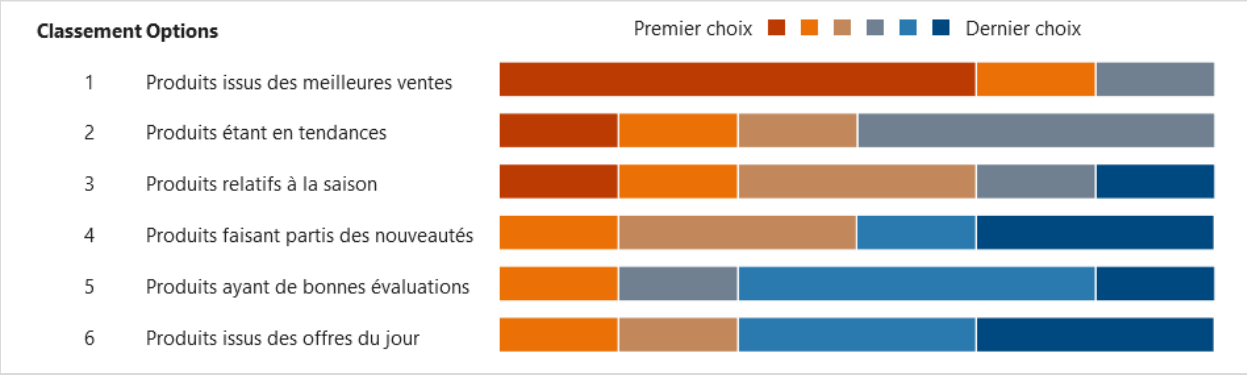


Figure 5 : Classement des réponses de la question 4 du sondage réalisé sur Microsoft Forms

Les réponses reflètent la façon de faire actuelle concernant les critères pour déterminer quels produits iront dans le stock cible. Il s'agira alors de mettre en avant les produits issus des meilleures ventes en priorité. S'en suivront les produits tendance, de saison et les nouveautés. Les 4 meilleurs critères seraient alors sélectionnés. Des réponses supplémentaires pourraient être utiles pour avoir un classement plus clair, car hormis le premier critère, le reste est assez partagé entre les différentes places du classement.

3.2.6. Résultats de la question 5

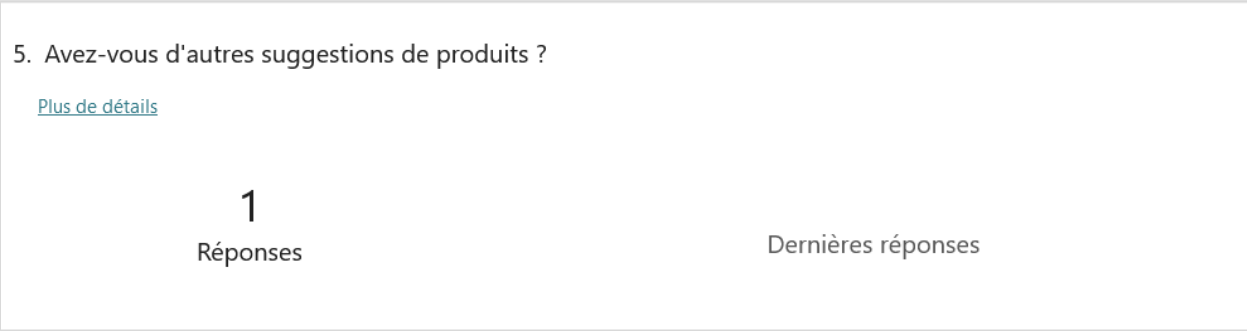


Figure 6 : Résultats de la question 5 du sondage réalisé sur Microsoft Forms

ID ↑	Nom	Réponses	Langue
1	anonymous	Zubehör zu den meistverkauften Produkten	Deutsch

Figure 7 : Détails des résultats de la question 5 du sondage réalisé sur Microsoft Forms

Cette question était facultative et liée à la précédente. Il y avait un certain nombre de critères à la question précédente, mais il en existe sûrement d'autres qui pourraient être pertinents. La seule réponse propose ici des accessoires en lien avec les produits les plus vendus. Cela pourrait être des coques de protection pour un téléphone par exemple. Après réflexion, il aurait été pertinent d'inclure ce critère précédemment pour découvrir où il se serait placé dans le classement, car ce genre de produit est déjà disponible dans le stock cible actuel. Des réponses supplémentaires pourraient potentiellement appuyer ce point.

3.2.7. Résultats de la question 6

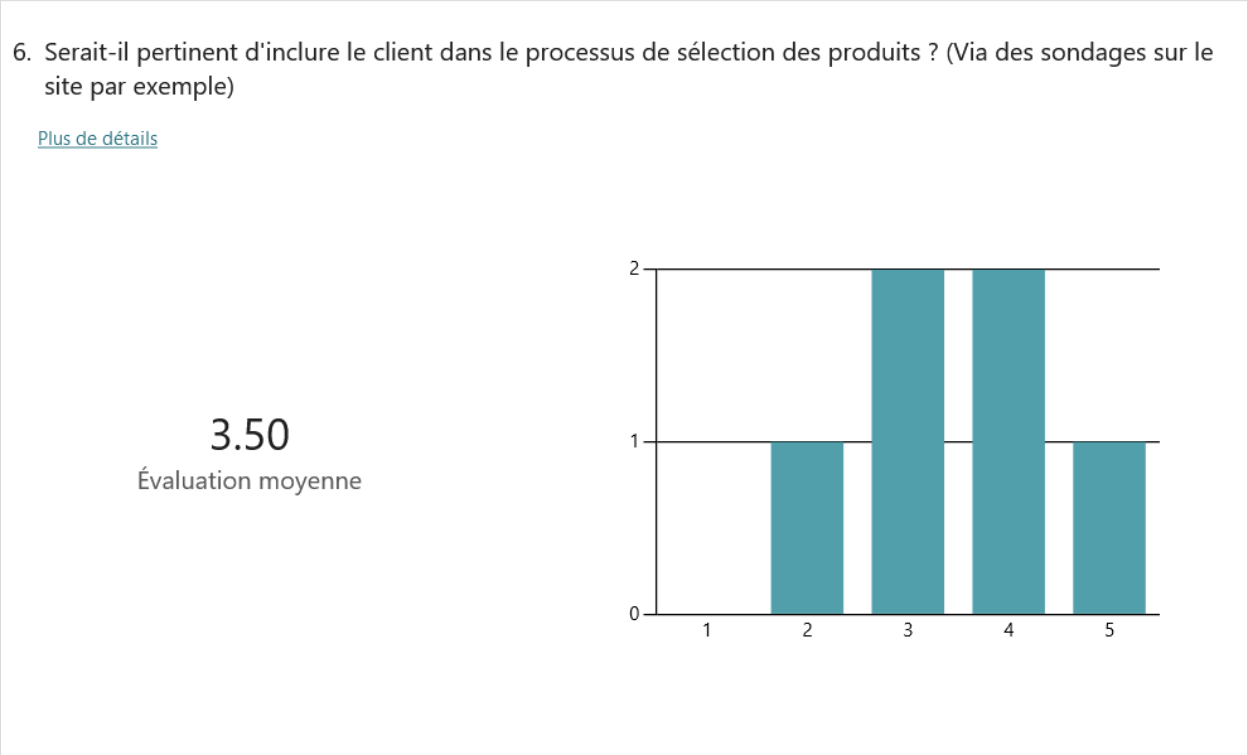


Figure 8 : Résultats et graphique de la question 6 du sondage réalisé sur Microsoft Forms

Le client n'est à l'heure actuelle pas directement incorporé dans le processus de sélection des produits. Il a plutôt un impact indirect, car ce que les clients achètent le plus se retrouve dans le stock cible. Il fallait répondre ici sur une échelle allant de 1 (pas pertinent) à 5 (très pertinent) et il semblerait que l'avis général soit mitigé avec un léger penchant en direction du « très pertinent ». Il y a donc un intérêt, mais ce dernier ne semble pas être une priorité. Intégrer cette fonction pourrait s'avérer techniquement difficile avec les accès actuels, mais un concept sous forme de maquette serait envisageable. Cela permettrait d'avoir un aperçu de ce que cela pourrait donner sans forcément investir trop de temps étant donné l'intérêt moyen pour cette fonctionnalité.

3.2.8. Résultats de la question 7

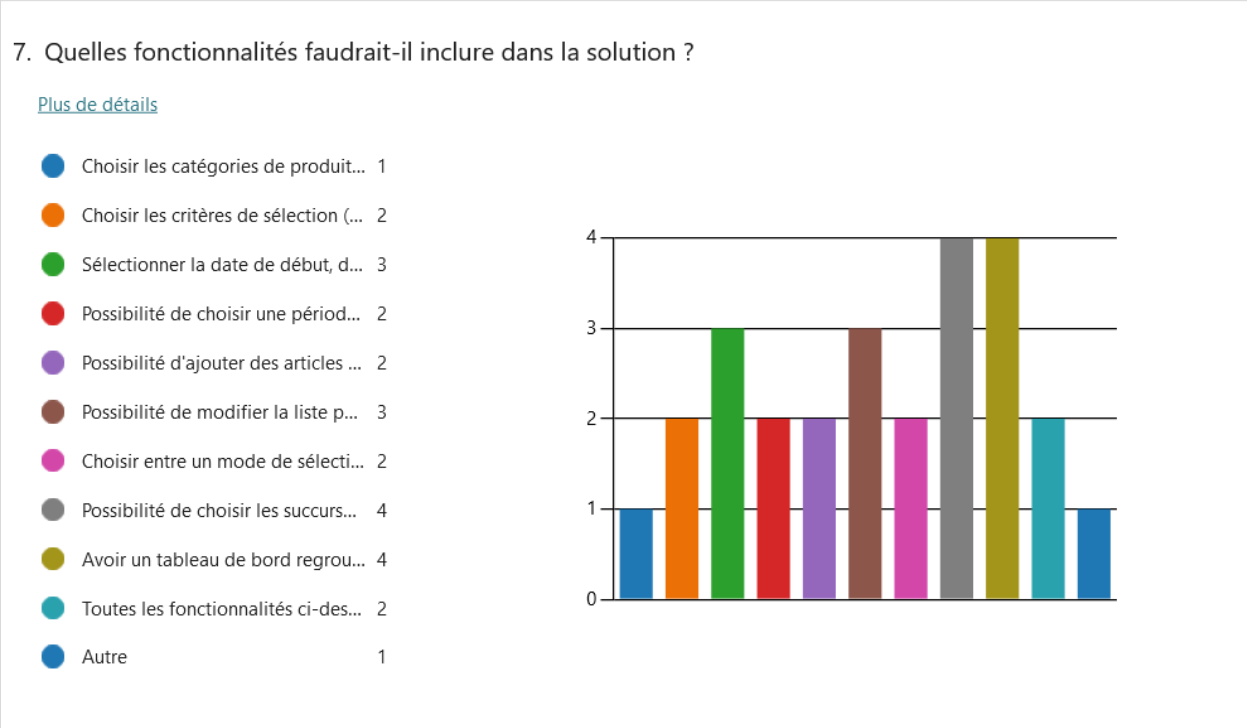


Figure 9 : Résultats et graphique de la question 7 du sondage réalisé sur Microsoft Forms

On en vient ici aux choix des fonctionnalités à inclure à la solution de ce projet. Les 2 fonctionnalités phares sont la possibilité de sélectionner la succursale afin d'avoir une gestion indépendante, cela reflète l'avis général lors de la première question, et l'ajout d'un tableau de bord regroupant plusieurs informations comme l'état du stock cible et le temps restant. Il faudra par la suite déterminer quelles informations précisément seront affichées sur ce tableau de bord. Viennent ensuite la fonctionnalité de sélection des dates et quantités des produits à destination du stock cible et la possibilité de modifier la liste de produits pendant sa création et après sa finalisation. Cette dernière fonction contraste bien avec la majorité des réponses à la question 3. Il faudra donc idéalement une solution qui est en mesure de tout faire tout en laissant un minimum de contrôle à l'utilisateur. Une seule personne a coché la case « Autre » et propose d'afficher le nombre d'articles vendu (par référence) lors de la sélection d'articles. Dans un premier temps, je vais me concentrer sur les 4 fonctionnalités préférées et, si le temps me le permet, j'envisagerai l'ajout d'autres fonctions afin d'offrir une solution plus polyvalente et personnalisée.

3.2.9. Résultats de la question 8

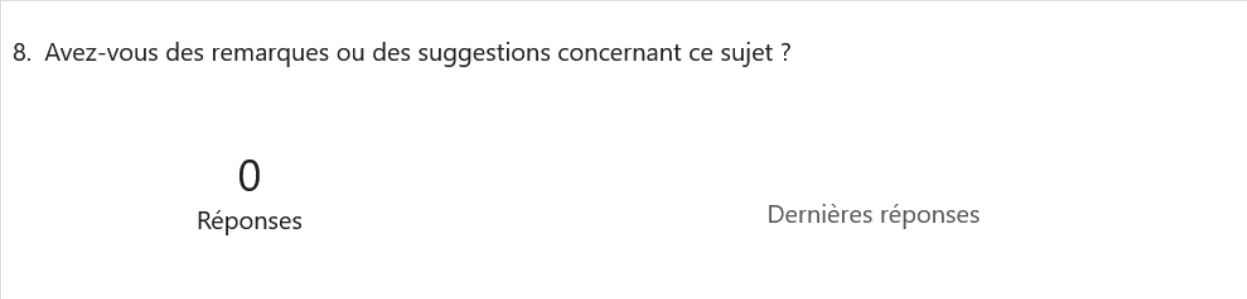


Figure 10 : Résultats de la question 8 du sondage réalisé sur Microsoft Forms

Cette question était facultative et liée en partie à la précédente. Elle permettait aux personnes sondées de proposer, d'une part d'autres fonctionnalités en plus de celles présentées précédemment et de l'autre, des suggestions ou remarques en lien avec le sujet. Il n'y a pas vraiment de matière à analyser ici étant donné l'absence de réponse.

3.2.10. Résultats de la question 9

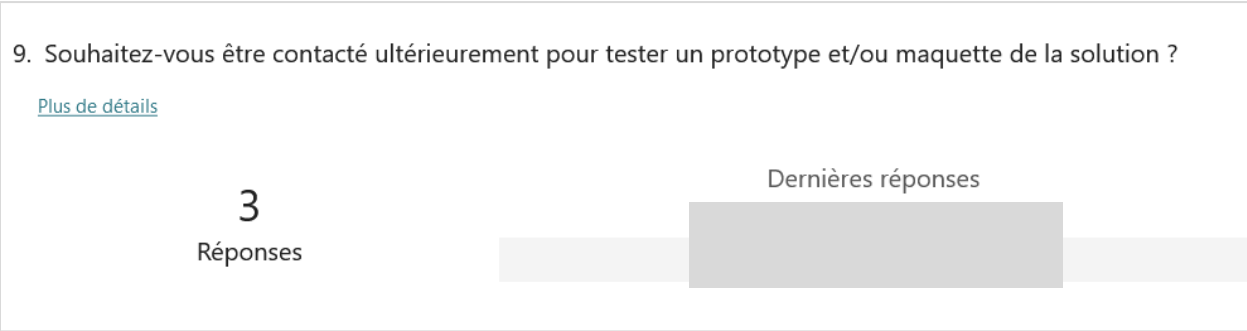


Figure 11 : Résultats de la question 9 du sondage réalisé sur Microsoft Forms

La question finale du sondage était, comme la précédente, facultative et permettait de récolter les adresses e-mail des personnes intéressées à participer à des sessions de test sur une maquette ou un prototype de la solution. Comme le sondage était anonyme, il s'agit ici de la seule manière de lier des réponses à une personne en particulier. Cela risque d'être peu pour réaliser des tests, mais il n'est pas impossible ni même extrêmement compliqué de trouver davantage de personnes pour ces tests.

3.2.11. Conclusion du sondage

Malgré le faible nombre de réponses en raison du délai réduit, le sondage a permis de mettre en lumière les besoins et attentes des utilisateurs. Il sera nécessaire de relancer les sondés pour obtenir davantage de réponses et augmenter la précision des résultats. Cette expérience m'a permis de mieux comprendre les contraintes auxquelles on peut être confronté en entreprise. Bien que le sondage ait été prêt à la fin de la première semaine, il aurait été judicieux d'y penser dès le début afin d'obtenir une validation plus rapidement. Cela souligne l'importance de la gestion du temps et de la coordination avec les parties prenantes dans de tels projets.

3.3. Faisabilité technique

3.3.1. Machine Learning

Avant de commencer à développer une solution à la problématique, il va falloir rechercher comment réaliser la chose techniquement parlant. D'après mes premières recherches sur le sujet, il serait pertinent d'utiliser du Machine Learning dans le but d'automatiser la sélection des produits. Le Machine Learning, ou apprentissage automatique, est une branche de l'intelligence artificielle qui donne aux machines la capacité d'apprendre de leurs expériences sans être explicitement programmées. Il utilise des algorithmes et des modèles statistiques pour analyser et interpréter des ensembles de données, identifier des modèles et faire des prédictions ou prendre des décisions avec un minimum d'intervention humaine. Dans mon cas, il serait utile pour analyser de grands jeux de données comme des données de ventes ou de recherches faites par l'utilisateur dans le but de déterminer les besoins des clients et les produits à destination du stock cible. Cependant, je ne vais pas rentrer trop dans les détails ici, car je vais y consacrer une partie dans le rapport du Travail de Bachelor. En effet, ne possédant ni de compétences en Machine Learning ni de connaissances très avancées en Python, je ne me sens pas à même de développer une solution utilisant ces concepts. Il me faudra un temps considérable pour me former dans ces domaines et ne disposant que de 11 semaines pour réaliser mon travail, je ne peux pas me permettre de prendre trop de temps pour me former sur une technologie. C'est pourquoi je réserve une partie du rapport final pour exposer des recherches sur la « solution idéale ». Une solution qui, avec les bonnes ressources et compétences, devrait permettre d'obtenir de meilleurs résultats en matière d'automatisation de la liste des produits.

3.3.2. Web Scraping

Une autre façon de faire serait d'utiliser du web scraping pour récupérer une partie des données utiles à la sélection des produits. Cette technique consiste à, par des lignes de code, accéder à des pages web et à en extraire les informations souhaitées. Cela passe par la sélection de balises[11] HTML[12] et de leur contenu. Il s'agit d'un procédé vu en cours lors de la 2e année du cursus en Ingénierie des médias. Imaginons que sur une page web, il y a un tableau comprenant plusieurs lignes et entêtes. Si nous souhaitons récupérer le contenu de chaque cellule du tableau, nous pouvons, à l'aide du web scraping, accéder à la page web en question, indiquer la balise HTML correspondant à une cellule du tableau et ensuite accéder à son contenu. Dans l'exemple de tableau sur l'image ci-dessous, nous pourrions commencer par sélectionner le tableau par la balise « table » puis sélectionner tous les éléments « td » depuis le tableau.

[11] Une balise est un élément de code en HTML utilisé pour définir et structurer des parties spécifiques du contenu sur une page web.
[12] Le HTML (HyperText Markup Language) est un langage de balisage utilisé pour créer et structurer le contenu des pages web.

Example

A simple HTML table:

```
<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

Try it Yourself »

Figure 12 - Exemple de tableau en HTML (W3Schools)

Le web scraping peut être utilisé de plusieurs manières que ce soit via un langage de programmation à l'aide de librairies spécialisées ou même depuis des plugins sur les navigateurs. Étant donné que ce projet tend à une solution proposant plusieurs fonctionnalités personnalisées, les plugins risquent de ne pas suffire pour couvrir tous les cas d'usage de la solution. Je vais alors me concentrer sur les outils utiles au développement du produit final. En termes de langage de programmation, qui sert de base pour créer le produit, je vais m'orienter vers quelque chose de connu et d'éprouvé durant mes 3 années en Ingénierie des médias, JavaScript[13].

Il serait tout à fait possible d'imaginer un autre langage pour développer la solution, mais pour les mêmes raisons que pour le Machine Learning et Python, je préfère consacrer du temps à créer la solution et également me former sur des dépendances du langage principal plutôt que d'apprendre quelque chose de complètement nouveau et qui prendra un certain temps à assimiler et appliquer. Il est vrai que ce travail pourrait être l'occasion de s'essayer à de nouveaux langages, mais le temps à disposition risque de ne pas être suffisant pour couvrir le nécessaire à la solution finale. JavaScript étant un pilier des technologies web et un langage étudié de nombreuses façons lors de la formation, ce fut un choix évident pour moi.

De plus, il est possible de faire un nombre significatif de choses avec grâce notamment aux nombreuses librairies disponibles en ligne. Cependant, JavaScript seul risque de ne pas être suffisant, car il faudra aussi une interface pour l'utilisateur. Il faudra donc y ajouter du HTML pour la structure et du CSS[14] pour la mise en forme et le style. Nous aurons ainsi les 3 éléments à la base d'une solution web, c'est-à-dire la structure (HTML), la mise en forme (CSS) et la logique (JavaScript).

[13] JavaScript est un langage de programmation script qui permet de créer des pages web interactives et dynamiques.
[14] Le CSS (Cascading Style Sheets) est un langage de feuille de style utilisé pour décrire la présentation et la mise en forme des pages web écrites en HTML.

Le web scraping permettrait dans les faits d'accéder à une page contenant une liste de produits et de récupérer cette même liste. Comme les tris et filtres font partie intégrante de l'URL[15] des pages sur galaxus.ch[16] et digitec.ch[17], il sera possible de charger une liste directement avec certains critères. Nous pourrions ainsi, comme sur la capture d'écran ci-dessous, accéder à la liste des téléphones portables, les trier par meilleures ventes et ajouter un filtre afin d'obtenir les produits en stock. Ce dernier étant important pour que les produits soient disponibles dans le stock cible de chaque succursale. Si un produit n'est pas en stock à l'entrepôt, il ne peut ainsi pas être ajouté dans le stock cible.

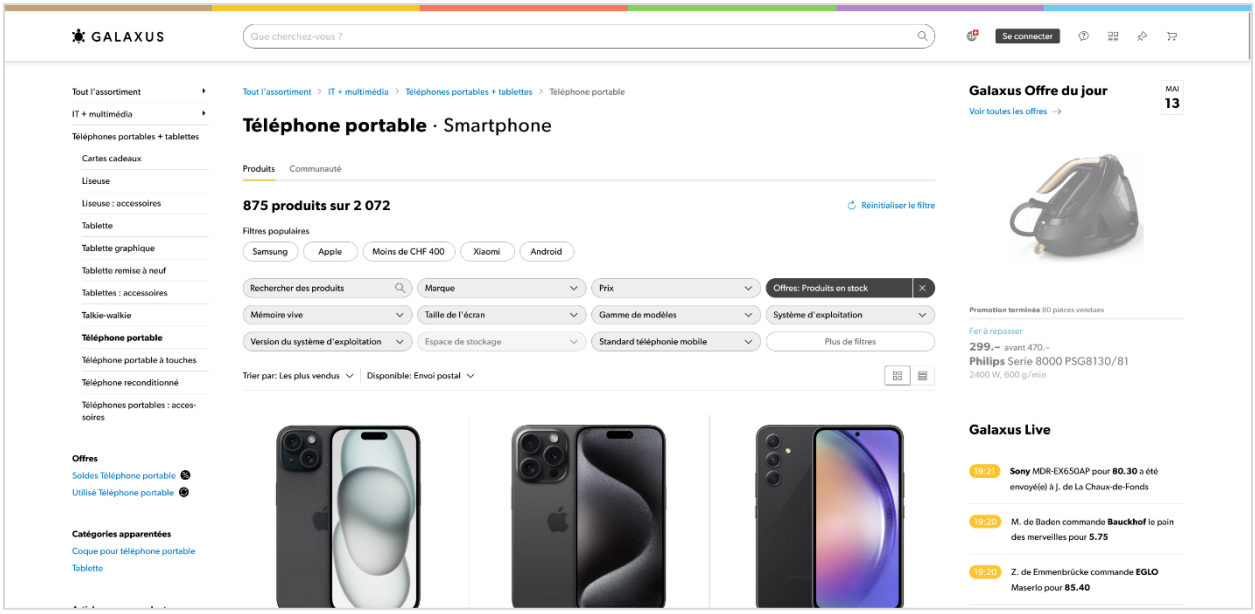


Figure 13 : Liste des produits de la catégorie « Téléphone portable »

https://www.galaxus.ch/fr/s1/producttype/telephone-portable-24?filter=t_off%3DInStock&so=15

Catégorie de produits — Filtres — Tris

Figure 14 : Construction de l'URL pour l'accès aux listes de produits sur galaxus.ch

[15] Une URL (Uniform Resource Locator) est l'adresse web qui spécifie l'emplacement d'une ressource sur Internet.
[16] https://galaxus.ch/
[17] https://digitec.ch/

Le web scraping est une chose, mais ne va pas suffire pour l'entièreté de la solution. En effet, il va permettre de récupérer une partie des données nécessaires à la sélection des produits pour le stock cible, mais il nous manquera encore quelques éléments pour perfectionner cela. Il nous faut quelque chose qui puisse non seulement accéder à des ressources web, mais également interagir avec ces dernières comme cliquer sur un bouton, remplir un formulaire, etc. Cela nous sera particulièrement utile pour enregistrer les produits dans l'ERP. Cette technique est également utilisée dans le script en Python, utilisé jusqu'à présent par l'entreprise. J'ai choisi cette solution, car je suis assez limité en termes d'accès sur la plateforme. En effet, je ne pourrai pas modifier le code pour y insérer directement des fonctionnalités sur l'ERP. Il faudra donc trouver quelque chose qui puisse le faire depuis l'extérieur.

Après plusieurs recherches sur le web, j'ai découvert 2 librairies pour JavaScript qui pourraient m'être utiles dans mon projet. La première se nomme « Puppeteer » et permet entre autres de faire du web scraping, mais également d'automatiser des actions sur des sites web. La deuxième se nomme « Cheerio » et permet de manipuler des éléments HTML. Je pense qu'il sera utile de combiner ces 2 outils pour à la fois récupérer des données, les traiter et enregistrer le tout dans l'ERP. De ce que j'ai compris, « Puppeteer » est un peu plus gourmand en ressources comparé à « Cheerio ». Il sera alors pertinent de réserver le premier pour l'automatisation de l'enregistrement des produits dans l'ERP, car cette tâche se fait en général à la fin une fois que la liste est définie. Afin de m'assurer de la viabilité de ces solutions, je vais réserver du temps pour tester ces librairies au début de la phase de réalisation du TB[18].

Ces méthodes me permettront donc de récupérer des informations autant sur la partie publique du site, galaxus.ch[19], et la partie privée, l'ERP. Il restera cependant un aspect que ces solutions ne couvrent pas, les tendances et les saisons. En effet, je pourrai aisément récupérer des produits en fonction de critères et filtres présents sur le site, mais ces 2 derniers critères n'en font pas partie.

3.3.3. Utilisation des outils de Google

En consultant une des ressources qui me parlait de web scraping, il se trouve que le site proposant l'article sur le web scraping avec JavaScript, serpapi.com[20] en l'occurrence, propose une grande variété d'API permettant de récupérer des données des différents outils de Google tels que Google Search, Google Maps, Google Trends, etc. C'est justement ce dernier qui pourrait me permettre de savoir ce qui est tendance et de saison sur une période donnée. L'entreprise propose un plan gratuit permettant un total de 100 recherches par mois via leurs API, cela devrait être suffisant pour les besoins du projet.

[18] TB : Travail de Bachelor
[19] https://galaxus.ch/
[20] https://serpapi.com

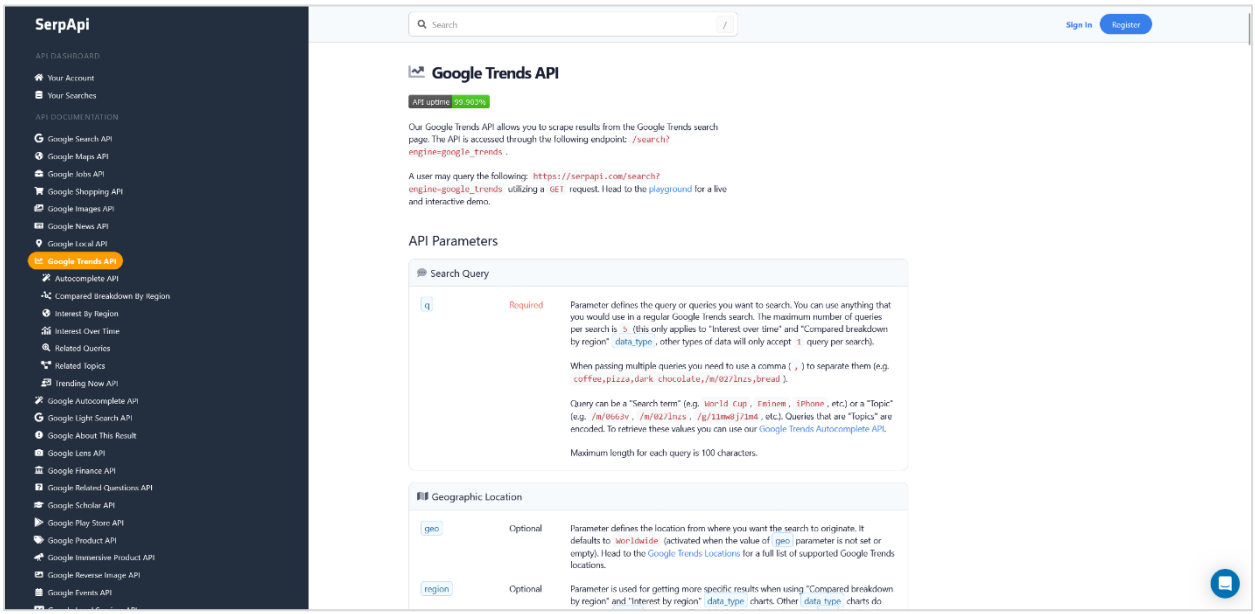


Figure 15 : Documentation sur l'utilisation de l'API Google Trends

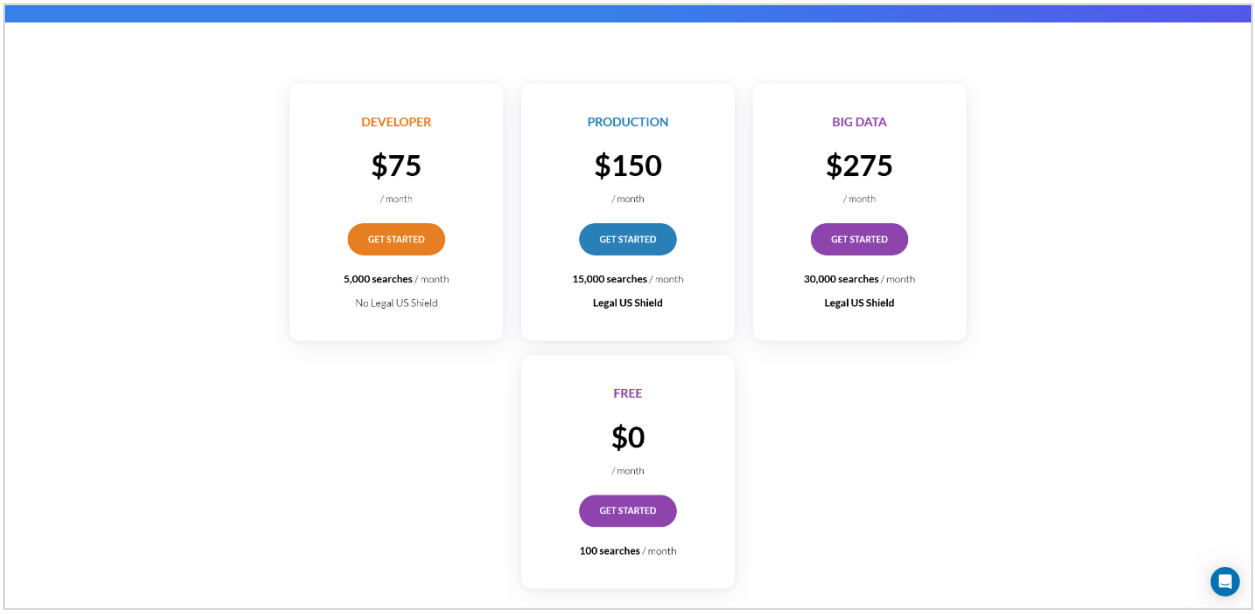


Figure 16 : Liste des prix de SerpApi.com

Sans forcément passer par une API, il serait également envisageable de récupérer les données directement d'une page de Google Trends via le web scraping également, car les paramètres de recherches sont inclus dans l'URL et donc on peut moduler cette dernière pour récupérer les données désirées.

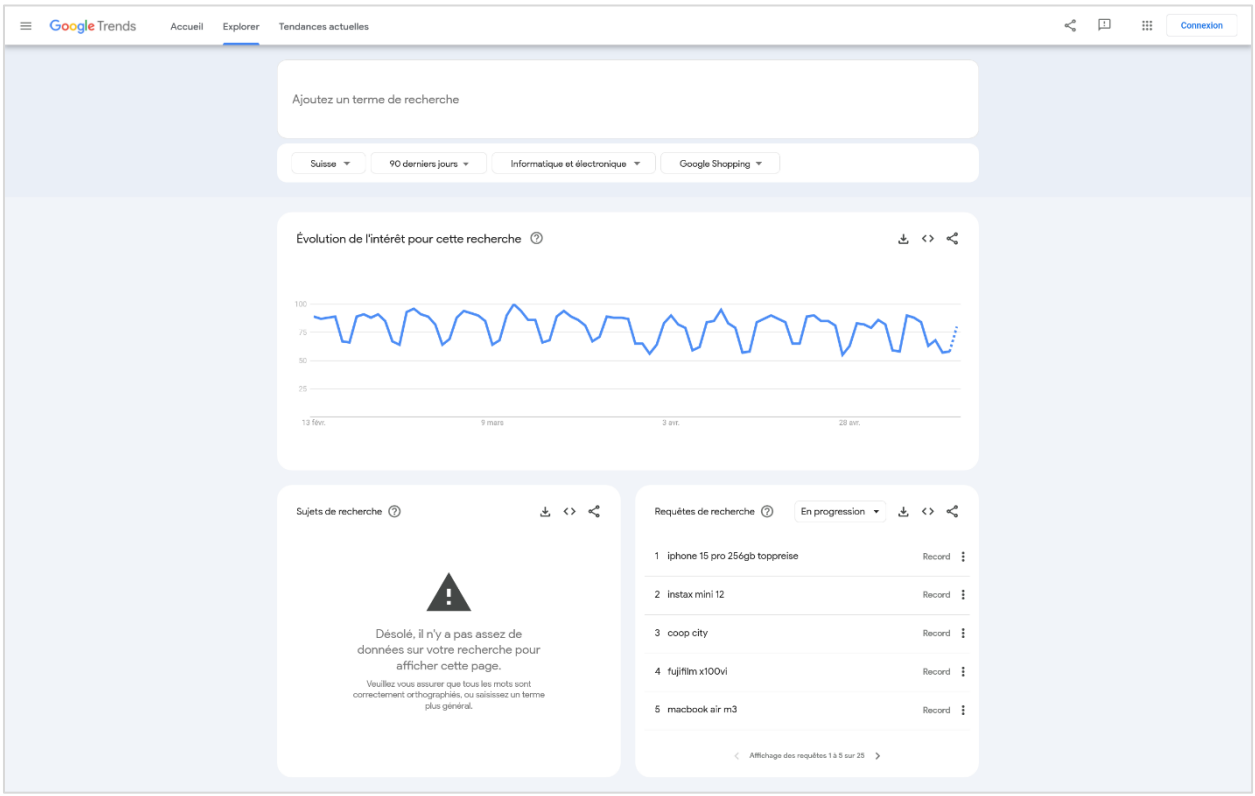


Figure 17 : Résultats Google Trends sur le thème « Informatique et électronique » avec Google Shopping

https://trends.google.fr/trends/explore?cat=5&date=today%203-m&geo=CH&gprop=froogle&hl=fr

Catégorie de recherche

Zone géographique

Période de recherche

Plateforme de recherche

Figure 18 : Construction de l'URL pour l'accès aux données de Google Trends

Il me faudra également tester ces 2 méthodes afin de voir si elles sont vraiment optimales pour mon projet. Ces dernières me permettent de récupérer des mots-clés recherchés par les utilisateurs des différentes plateformes de Google. Cependant, ces mots-clés ne correspondent pas nécessairement à des produits disponibles sur Digitec et Galaxus et de ce fait, il faudra une solution pour les vérifier ou transformer pour en tirer quelque chose d'utilisable par la suite.

3.3.4. Utilisation des catégories phares

Afin d'obtenir des résultats plus probants pour les produits tendance ou saisonniers, je pourrais utiliser le web scraping pour récupérer les produits provenant des catégories phares sur Digitec et Galaxus. En effet, il y a une section dédiée aux 10 catégories les plus consultées sur la page d'accueil des 2 plateformes.

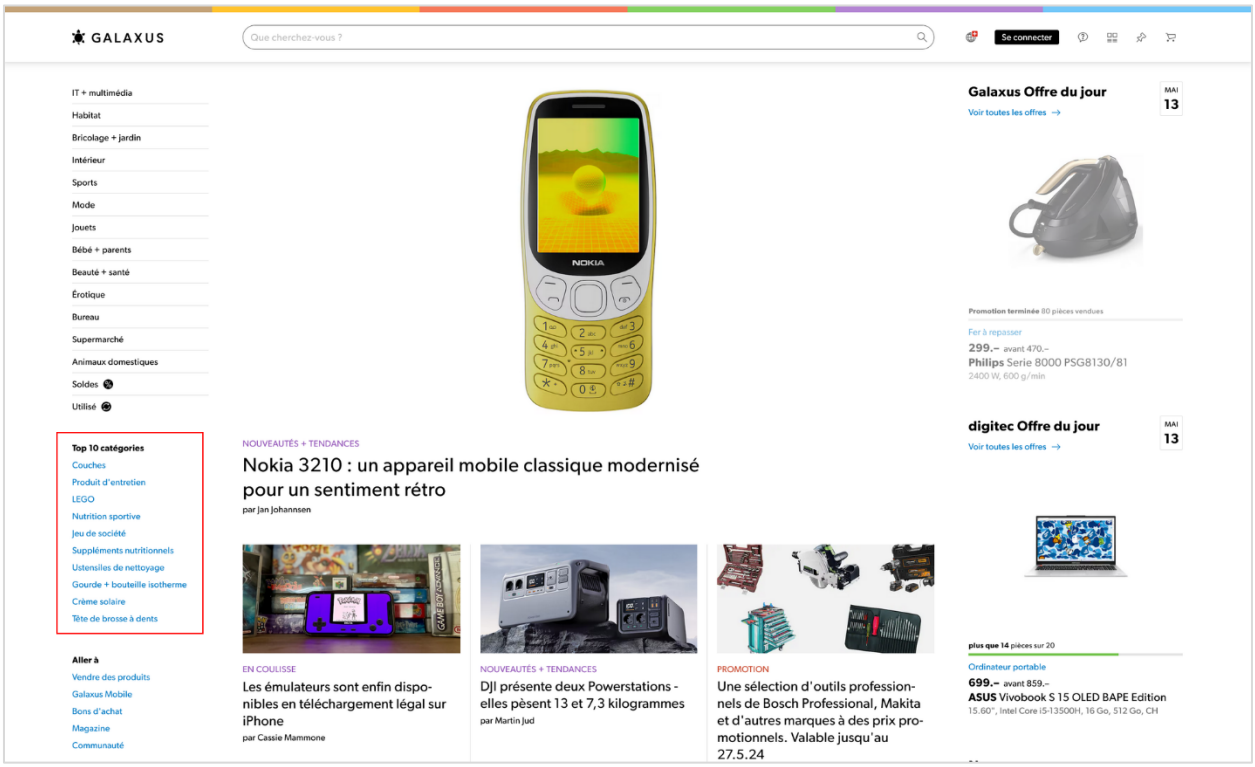


Figure 19 : Page d'accueil de galaxus.ch avec les catégories phares

Ces catégories reflètent ce que les clients cherchent et achètent le plus sur les 2 plateformes et ce classement peut évoluer au cours de l'année. Cela permettra à la fois d'avoir l'aspect tendance et saisonnier. Il pourrait être utile de mélanger cette méthode avec les précédentes pour obtenir de meilleurs résultats et ainsi proposer une liste de produits la plus pertinente possible.

3.3.5. Utilisation de l'IA

Un autre point est également apparu au fur et à mesure de mes recherches, l'intelligence artificielle (IA). En effet, cette dernière pourrait être d'une grande utilité lors de l'analyse de données. Il serait possible de transmettre une liste de produits à l'IA en lui demandant une série d'actions comme faire une nouvelle sélection ou suggérer des produits complémentaires à ceux présents sur la première liste. Des IA comme ChatGPT d'OpenAI, Gemini de Google ou encore Mistral AI de l'entreprise du même nom pourraient accélérer le processus de sélection de produits et améliorer la pertinence de la liste. Il est possible d'utiliser ce genre d'IA en passant par les API proposées par les entreprises. Les API servent d'interface entre l'utilisateur et la machine. L'utilisateur envoie une requête à l'API et cette dernière va retourner une réponse. Cela permettrait d'utiliser un « prompt », un texte regroupant une série d'ordres, qui indique à l'IA ce qu'il faut faire et quels sont les résultats attendus. Idéalement, il faudrait utiliser ces modèles d'IA en local afin d'éviter d'envoyer des données sur des serveurs à l'autre bout du monde. Il s'agit d'une piste à explorer que ce soit pour l'élaboration de la solution ou bien lors de la rédaction des recommandations pour aller plus loin sur le sujet.

3.4. Accès aux bases de données et à l'ERP

La méthode du web scraping sera donc utile à partir du moment où je n'ai pas d'accès aux bases de données de l'entreprise pour en extraire directement les données. Ce point est encore en suspens et en discussion au sein de l'entreprise et devrait être réglé à priori lors des premières semaines de la phase de réalisation. Un accès à la base de données me permettrait de simplifier la méthode de récupération des produits et d'économiser du temps et des ressources. J'ai cependant pu obtenir des accès supplémentaires à l'ERP en plus de ceux que j'ai en tant qu'employé du Retail. J'ai à présent les mêmes accès qu'un manager de succursale adjoint, ce qui me donne des droits d'édition sur le stock cible.

3.5. Choix de l'environnement de développement

Maintenant que je sais comment procéder pour générer la liste et pour l'enregistrement des produits dans l'ERP, il me faut trouver une façon de mettre en commun toutes ces technologies. Bien que ma solution devrait être composée de HTML, CSS et JavaScript, il est rare de nos jours qu'un site utilise ces technologies seules. En effet, il est de plus en plus commun d'utiliser un framework pour développer une solution digitale à l'heure actuelle.

3.5.1. Framework des portails de vente

Un framework est une structure logicielle conçue pour servir de fondation et de guide dans le développement de logiciels. Il fournit une base standardisée sur laquelle construire, avec des outils préétablis, des bibliothèques, et des conventions qui facilitent le processus de développement. Les frameworks sont particulièrement utiles pour promouvoir des pratiques de codage cohérentes et efficaces et réduire le temps de développement en offrant des solutions réutilisables à des problèmes courants. Ils en existent pour divers domaines de programmation, y compris le développement web, les applications mobiles, et le développement de jeux.

Il se trouve que l'entreprise utilise un framework pour le développement des 2 plateformes de vente. Un rapide coup d'œil sur l'outil d'analyse de site web de wappalyzer.com[21] m'a permis de voir toutes les technologies employées sur galaxus.ch[22] et nous pouvons y voir notamment l'utilisation d'un framework web nommé Next.js.

[21] https://wappalyzer.com
[22] https://galaxus.ch

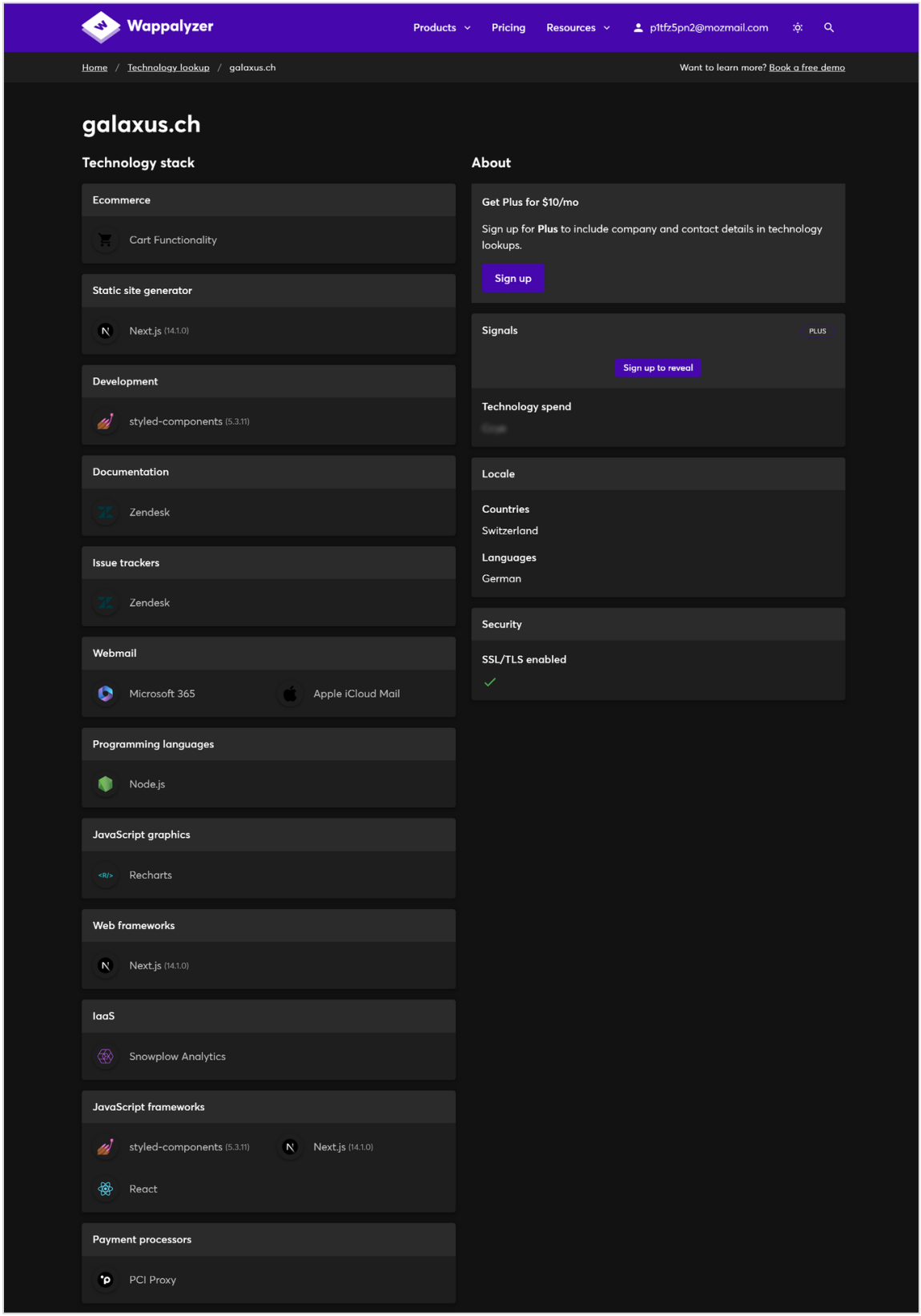


Figure 20 : Analyse technologique de galaxus.ch réalisée à l'aide de l'outil de wappalyzer.com (1/2)

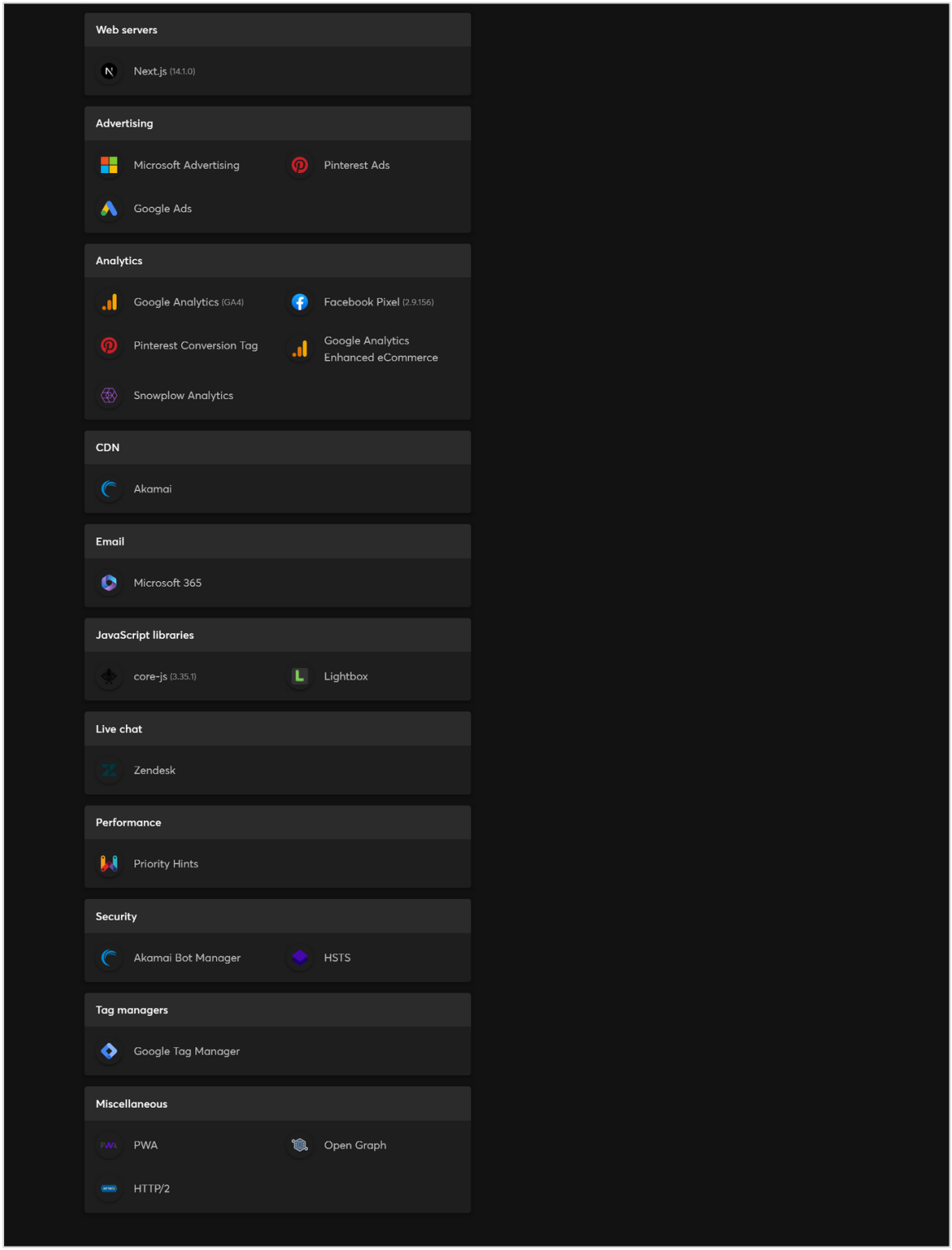


Figure 21 : Analyse technologique de galaxus.ch réalisée à l'aide de l'outil de wappalyzer.com (2/2)

Next.js est un framework de développement web open-source basé sur Node.js[23] qui permet aux développeurs de construire des applications web et des sites web statiques plus efficacement. Il est construit par-dessus React.js[24], apportant des fonctionnalités supplémentaires telles que le rendu côté serveur et la génération de sites statiques, qui améliorent les performances et l'expérience utilisateur. Next.js offre également des capacités de routage basées sur le système de fichiers, ce qui simplifie la configuration des routes dans une application. Conçu pour la production dès le départ, Next.js facilite le déploiement d'applications React avec moins de configuration et une optimisation automatique.

Étant donné qu'il s'agit d'une technologie utilisée par l'entreprise et avec laquelle je me suis déjà familiarisé avec au cours du dernier projet de semestre à la HEIG-VD et lors d'un projet personnel, il me parait pertinent d'utiliser ce framework pour mon projet. Bien entendu, il me faudra faire des tests au début de la phase de réalisation pour être sûr que le tout s'imbrique de façon optimale.

3.5.2. Environnement alternatif

Il est vrai que je pourrais également me tourner vers la technologie utilisée pour l'ERP de l'entreprise. Après une nouvelle analyse sur wappalyzer.com[25], j'ai pu noter que Microsoft ASP.NET est utilisé en tant que framework web.

[23] Node.js est une plateforme JavaScript côté serveur qui permet de créer des applications réseau rapides et évolutives.
[24] React.js est une bibliothèque JavaScript utilisée pour construire des interfaces utilisateur interactives et dynamiques, principalement pour les applications web.
[25] https://wappalyzer.com

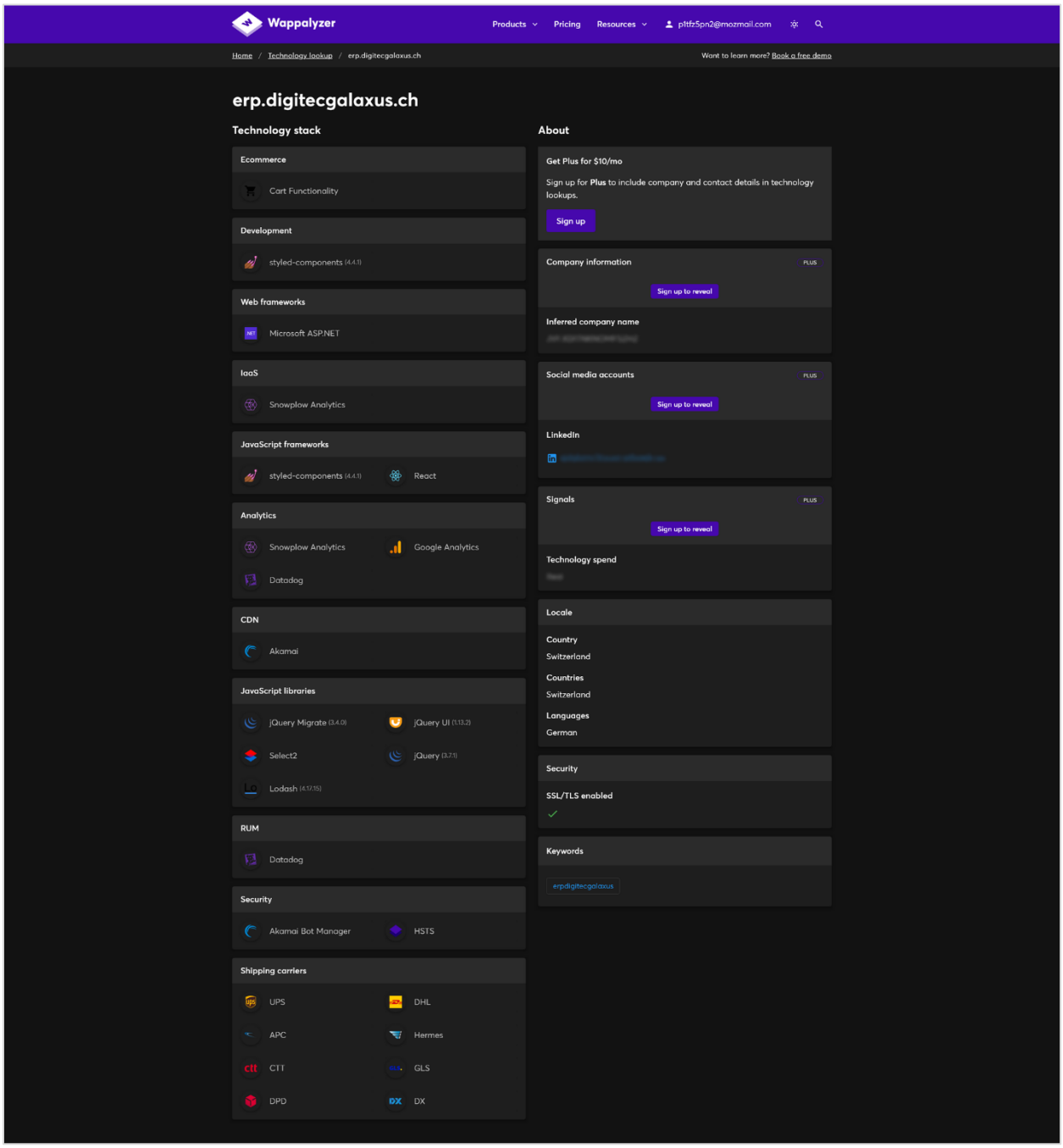


Figure 22 : Analyse technologique de l'ERP réalisée à l'aide de l'outil de wappalyzer.com

Microsoft ASP.NET est un framework pour le développement d'applications web, conçu par Microsoft. Il permet aux développeurs de construire des sites web, des applications web et des services web dynamiques et interactifs. ASP.NET supporte plusieurs langages de programmation, tels que C#, VB.NET, et F#, offrant ainsi une grande flexibilité dans le choix des outils de développement. Ce framework est remarquable pour son intégration profonde avec le .NET framework, facilitant l'accès à une large gamme de bibliothèques pour la sécurité, la gestion des données, et les communications réseau. ASP.NET est conçu pour améliorer la productivité en automatisant nombre de tâches de programmation, permettant ainsi de se concentrer sur la logique métier de l'application tout en assurant une performance, une évolutivité et une sécurité robustes.

Malgré toutes ces qualités, il me faudrait surmonter un problème de taille, mon absence de connaissances envers les langages de programmation utilisés par le framework de Microsoft. Comme pour le Machine Learning et Python, il me faudrait un temps considérable pour me mettre à niveau et assimiler toutes les connaissances nécessaires à la réalisation de mon projet. De plus, le but principal de ma solution étant de faire une sélection de produits, je trouve cela cohérent de se rapprocher de la façon de faire des portails de vente en intégrant les mêmes technologies. On pourrait assimiler la sélection de produits pour le stock cible aux produits qu'un client va choisir et mettre dans son panier. De ce fait, il fait sens de proposer quelque chose se reprochant de digitec.ch[26] et galaxus.ch[27] d'un point de vue stylistique également. Cela ne m'empêchera pas de proposer 2 styles pour ma solution, sous forme de maquettes. L'une se rapprochant des portails de vente et l'autre avec la mise en page de l'ERP pour une éventuelle intégration future.

3.6. Conclusion

En conclusion, grâce aux recherches et réflexions précédentes, il semble approprié de démarrer le projet avec Next.js comme base pour construire la solution, en utilisant différentes bibliothèques pour élaborer les principales fonctions. Le choix de JavaScript, associé à HTML et CSS, permet de créer une solution web intégrée et cohérente avec les technologies utilisées par l'entreprise. Les différentes méthodes envisagées pour le web scraping et l'automatisation ne sont certes pas les plus optimales, mais devraient me permettre de proposer un prototype utilisable.

4. Analyse et description des objectifs et des besoins

À la suite des éléments mis en lumière au point précédent, je suis en mesure de définir les objectifs qui guideront mon travail tout au long de ce projet. Ces objectifs résultent d'une analyse des besoins des utilisateurs finaux et de recherches sur le sujet.

4.1. Besoins

Pour commencer, mon projet devra répondre aux besoins suivants :

[26] https://digitec.ch
[27] https://galaxus.ch

4.1.1. Besoins fonctionnels

- La solution doit être capable de générer une liste de produits à partir de critères prédéfinis, tels que les meilleures ventes, les nouveautés, les produits saisonniers, etc. Cette liste doit être ajustable en fonction des tendances du marché et des besoins spécifiques des succursales.
- La solution doit pouvoir enregistrer automatiquement chaque produit de la liste dans l'ERP, en définissant les quantités, les dates de début et de fin pour chaque article.

4.1.2. Besoins des utilisateurs

- La solution doit être conçue pour être facilement utilisable par des personnes sans compétences techniques particulières. L'interface utilisateur doit être claire, intuitive et permettre une prise en main rapide.

4.2. Objectifs

À partir des besoins énoncés précédemment et de la documentation recueillie, mon projet aura comme objectifs les points suivants :

4.2.1. Automatisation complète du processus

L'objectif principal est de développer une solution permettant d'automatiser entièrement la sélection et l'enregistrement des produits dans le stock cible des succursales de Digitec Galaxus AG. Cela comprend la génération d'une liste d'articles basée sur des critères définis, et l'enregistrement automatique de ces articles dans l'ERP de l'entreprise.

4.2.2. Simplification et accessibilité de la solution

La solution doit être accessible à tous les utilisateurs, sans prérequis techniques spécifiques. Cela signifie qu'elle doit être intuitive, user-friendly et ne nécessiter aucune installation ou configuration complexe. L'interface utilisateur doit être conçue pour être conviviale et facilement compréhensible par les utilisateurs finaux.

4.2.3. Optimisation du temps et de l'efficacité

Le processus actuel, en partie manuel, est long et répétitif. La nouvelle solution doit permettre de réduire considérablement le temps consacré à la sélection et à l'enregistrement des produits. Cela inclut la minimisation des interventions humaines et la réduction des erreurs potentielles liées aux manipulations.

5. Organisation et description des tâches et des livrables

Afin d'organiser au mieux mon travail, ce dernier sera découpé en phases. Cela permettra d'avancer pas à pas dans ce projet en établissant pour chaque phase des tâches distinctes. En parallèle des tâches mentionnées ci-dessous, je vais devoir également produire, au fur et à mesure de l'avancement du projet, le rapport final du TB ainsi que l'affiche et le résumé publiable.

5.1. Découpage en phases

5.1.1. Phase 1 : Conception de la solution

- Finalisation des spécifications fonctionnelles et techniques.
- Conception de l'architecture de la solution, choix des technologies et des outils.
- Création des maquettes de l'interface utilisateur (deux styles : portail de vente et ERP).
- Test et validation des maquettes par le mandant et des représentants des succursales.
- Adaptation des maquettes suite aux tests utilisateurs.

5.1.2. Phase 2 : Développement des fonctionnalités

- Développement de la fonctionnalité de génération automatique de la liste de produits basée sur des critères définis.
- Développement de la fonctionnalité d'enregistrement automatique des produits dans l'ERP.
- Développement des fonctionnalités annexes (tableau de bord, etc.)
- Développement de tests unitaires pour vérifier le bon fonctionnement des principales fonctionnalités.
- Point de situation avec le mandant sur la faisabilité du projet (ce qui a été fait, ce qui pourrait être fait et inversement)

5.1.3. Phase 3 : Intégration des fonctionnalités

- Intégration des différentes fonctionnalités développées.
- Ajustements à la suite de la mise en commun.
- Préparation au déploiement de la solution.

5.1.4. Phase 4 : Tests et optimisation

- Organisation de sessions de tests utilisateurs avec le mandant et des représentants des succursales.
- Optimisation et corrections de la solution suite aux tests utilisateurs.

5.1.5. Phase 5 : Finalisation et recommandations

- Déploiement de la solution.
- Présentation de la solution finale au mandant.
- Rédaction de recommandations pour aller plus loin dans le sujet.
- Rédaction d'un guide de prise en main de l'outil.

5.2. Livrables

Ces phases s'accompagneront également d'un certain nombre de livrables dont voici la liste :

- Maquettes de la solution
- Code source de l'outil
- Liste de recommandations
- Guide de prise en main

6. Planning

La réalisation de ce projet se déroule sur 11 semaines, du 27 mai au 9 août 2024. La planification, tout comme le reste du travail, est organisée sur Notion. Il s'agit d'une application de productivité tout-en-un qui combine des fonctionnalités de prise de notes, de gestion de projets, de bases de données et de collaboration en équipe. Cet outil me permet d'organiser facilement et de manière centralisée mon travail. J'ai pu également donner accès à mon projet sur Notion au mandant afin qu'il puisse aisément suivre l'avancée du travail.

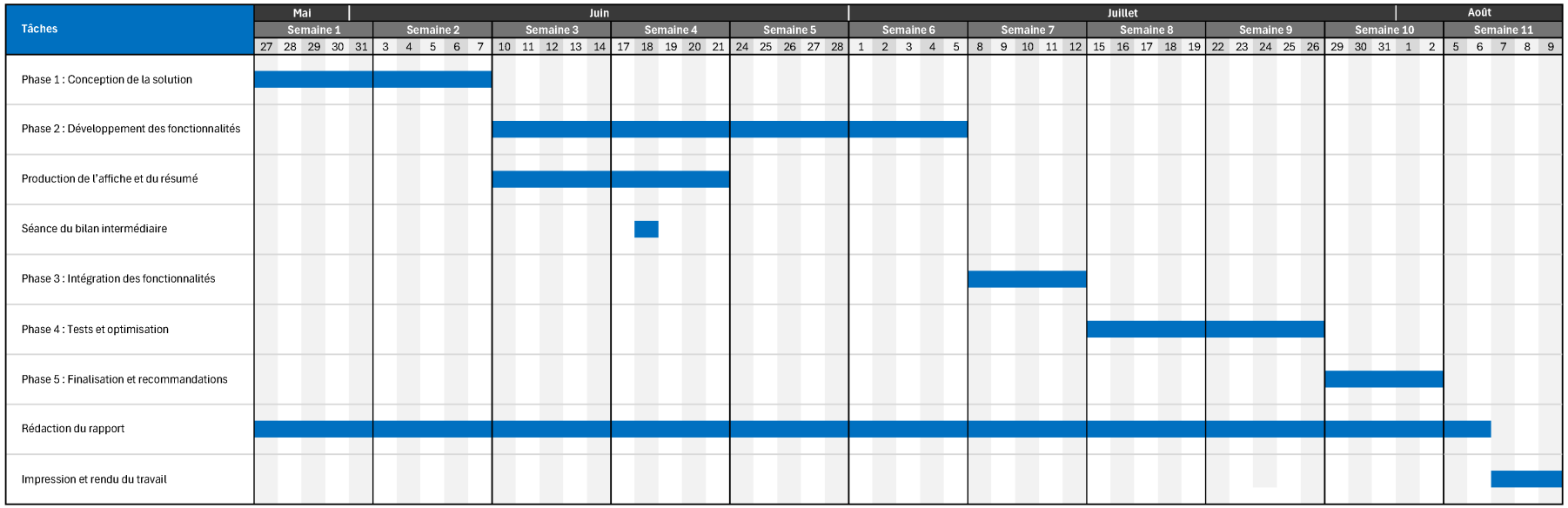


Figure 23 : Planification du projet au format Gantt Chart

La planification se découpe en plusieurs phases comme décrites dans la description des tâches. Le temps attribué par phase est proportionnel à la quantité et la complexité des tâches qui y sont rattachées. Ainsi, le développement des fonctionnalités est la phase qui me prendra le plus de temps, car il s'agit du cœur du travail. En parallèle des principales phases, il me faudra également rédiger le rapport, le résumé et concevoir l'affiche en plus d'organiser la séance pour le bilan intermédiaire. Le travail se terminera par l'impression du rapport et le rendu de ce dernier.

7. Gestion des risques

Chaque projet, quelle que soit sa taille, comporte un certain nombre de risques qui peuvent impacter sur différents niveaux le bon déroulement du travail. Il est donc important d'identifier ces risques et de prévoir des mesures pour les contrer ou limiter leur impact.

7.1. Risques et mesures

Mon projet n'échappe pas à la règle et comporte plusieurs risques. Le tableau ci-dessous les recense ainsi que les mesures correspondantes. De plus, deux critères sont utilisés pour calculer le facteur de risque (F). Le premier est la probabilité que le risque se réalise (P) et le deuxième est le niveau de gravité du risque (G). Ces deux critères sont notés de 1 à 5, allant du moins au plus probable/grave. Cela permet de calculer le facteur de risque, qui varie entre 1 et 25, en multipliant la probabilité par la gravité du risque. Plus le facteur de risque est élevé, plus il faudra surveiller le risque et prendre des mesures significatives pour le contrer ou l'atténuer.

N°	Risque	P	G	F	Mesure
1	Perte de données	2	5	10	Faire des copies de toutes les ressources à la fois sur un support de stockage externe et dans le cloud.
2	Absence (maladie ou travail)	3	3	9	Prévoir de la marge dans les tâches et, si nécessaire, rattraper le travail le weekend et limiter l'ajout de jours de travail en semaine.
3	Personnel interne non joignable	2	2	4	Dépendre le moins possible du personnel interne, anticiper les interactions pour mieux les planifier.
4	Manque de temps pour tout développer	3	4	12	Se concentrer sur la fonctionnalité principale en priorité (sélection des produits), car c'est elle qui représente le vrai défi et agir en fonction du temps restant pour le reste.
5	Manque de connaissances techniques	2	1	2	Limiter l'usage de technologies peu ou pas connues, se concentrer sur les acquis et leurs dérivés.
6	Manque d'accès au système, base de données, etc.	4	2	8	Limiter le plus possible le besoin d'accéder aux ressources internes (faire plus avec moins).
7	Erreurs dans le système suite à des manipulations	1	4	4	Travailler avec l'environnement de test pour éviter toutes erreurs.
8	Incompatibilité des dépendances (librairies, framework, etc.)	3	5	15	Réaliser les tests avec les outils choisis au début du travail pour rebondir rapidement et trouver des alternatives en cas de problèmes.

Figure 24 : Tableau des risques et mesures

7.2. Matrice des risques

La matrice ci-dessous permet de représenter visuellement les différents risques liés à mon projet. Cela me permet ainsi d'évaluer chaque risque et de voir lesquels sont à surveiller en priorité.

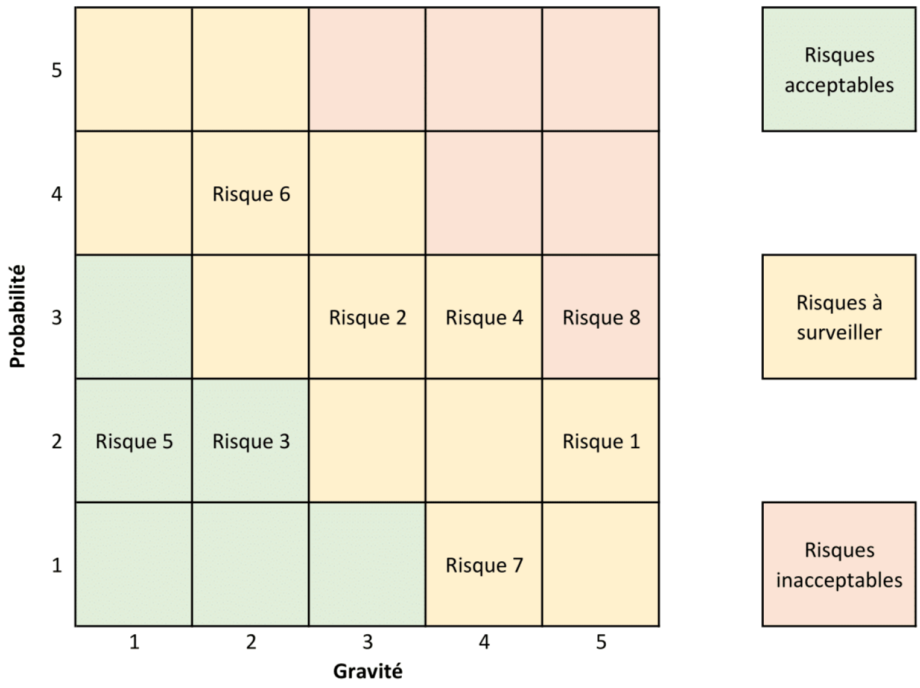


Figure 25 : Matrice des risques

8. La conclusion

Cette pré-étude m'a permis de m'immerger dans la problématique pour mieux préparer le travail à venir. Bien que le sujet ne me soit pas inconnu, étant donné que je travaille dans l'entreprise depuis plus de trois ans, j'ai pu en apprendre davantage sur ce qu'il représente. En tant qu'employé, je connaissais l'importance du stock cible, mais je ne m'étais pas rendu compte des tenants et aboutissants de ce sujet en coulisses avant d'entamer ce travail.

Nombreux sont les projets que l'on aimerait voir démarrer et prendre vie en interne, mais par manque de ressources ou en raison de la priorisation d'autres projets, ils ne peuvent malheureusement pas tous voir le jour dans un avenir proche. Ce constat est similaire dans de nombreuses entreprises, et le sujet du stock cible fait partie de ces projets en attente pour une durée indéterminée.

Ma problématique m'a permis d'explorer différentes façons de procéder pour obtenir le résultat escompté, mais je me suis vite rendu compte qu'un résultat optimal demanderait plus de ressources et de connaissances que je n'en possède actuellement. C'est pour cette raison que je vais à la fois proposer un prototype de la solution, mais également des recommandations sur la façon dont on pourrait traiter le sujet avec davantage de temps et de ressources. Il ne s'agira certes pas de la solution idéale, mais cela permettra d'entrouvrir une porte et d'exposer une méthode parmi tant d'autres.

Il reste encore des éléments en suspens qu'il faudra régler lors des premières semaines de la phase de réalisation. Premièrement, le niveau d'accès qui m'est octroyé n'est pas encore complètement défini. J'ai pu obtenir des droits supplémentaires dans l'ERP, équivalents à ceux d'un manager de succursale adjoint, mais je n'ai pas encore d'accès aux bases de données. Ce dernier point m'indiquera vers quelles technologies me tourner pour notamment récupérer des listes de produits.

Lors de cette pré-étude, j'ai pu faire face aux premiers problèmes liés à la dépendance envers différentes personnes pour avancer dans le projet au sein d'une grande entreprise. Que ce soit pour valider des démarches ou obtenir de l'aide, il faut anticiper un maximum et prévoir des alternatives en cas de problèmes.

Avec le recul, je n'ai pas consacré assez de temps aux tests d'un point de vue technique pour la future solution. En théorie, les différentes méthodes devraient fonctionner, car elles ont déjà été utilisées par le passé lors de mon cursus à la HEIG-VD, mais les tests seront déterminants pour la viabilité de ce projet. J'ai pris beaucoup de temps pour effectuer des recherches et rédiger le présent rapport, en plus des longues réflexions en dehors. C'est pourquoi la phase de réalisation commencera par des tests ainsi que la mise en place de l'environnement de développement.

En conclusion, je me sens prêt à démarrer cette prochaine phase et j'ai hâte de découvrir ce que j'arriverai à produire en onze semaines. Tout comme durant cette pré-étude, il y aura sûrement des moments de doute, des blocages, des problèmes, mais cela fait partie de l'exercice et constitue un bon défi avant de se lancer dans la vie active.

Glossaire

A

API

Une API (Application Programming Interface) est un ensemble de règles qui permet à différentes applications de communiquer entre elles.

B

Balises

Une balise est un élément de code en HTML utilisé pour définir et structurer des parties spécifiques du contenu sur une page web.

C

C#

C# est un langage de programmation orienté objet développé par Microsoft, principalement utilisé pour le développement d'applications Windows et de logiciels d'entreprise.

Cheerio

Cheerio est une bibliothèque Node.js qui permet de manipuler et d'analyser facilement des documents HTML.

Cookie de session

Un cookie de session est un petit fichier stocké sur le navigateur d'un utilisateur qui permet de maintenir l'état de la session active et de suivre l'activité de l'utilisateur pendant la durée de sa visite sur un site web.

CSS

Le CSS (Cascading Style Sheets) est un langage de feuille de style utilisé pour décrire la présentation et la mise en forme des pages web écrites en HTML.

E

ERP

Un ERP (Enterprise Resource Planning) est un système intégré de gestion qui permet de centraliser et d'automatiser les processus métier essentiels d'une organisation, tels que la comptabilité, la gestion des stocks, la production et les ressources humaines.

F

F#

F# est un langage de programmation fonctionnel conçu pour le développement sur la plateforme .NET, favorisant la concision et la robustesse du code.

Framework

Un framework est une structure de support préétablie qui facilite le développement d'applications en fournissant des composants réutilisables et des outils standardisés.

H

HTML

Le HTML (HyperText Markup Language) est un langage de balisage utilisé pour créer et structurer le contenu des pages web.

I

IA

L'IA (Intelligence Artificielle) est la simulation des capacités humaines par des systèmes informatiques, notamment l'apprentissage, le raisonnement et la résolution de problèmes.

Invite de commande

Un invite de commande est une interface textuelle qui permet aux utilisateurs d'exécuter des commandes système en tapant des instructions directement dans un terminal ou une console.

J

JavaScript

JavaScript est un langage de programmation script qui permet de créer des pages web interactives et dynamiques.

M

Machine Learning

Le Machine Learning est une branche de l'intelligence artificielle qui permet aux systèmes d'apprendre et de s'améliorer automatiquement à partir de données et d'expériences sans être explicitement programmés.

Microsoft ASP.NET

Microsoft ASP.NET est un framework pour développer des applications web dynamiques et interactives, supporté par la plateforme .NET.

N

Next.js

Next.js est un framework React qui permet de créer des applications web optimisées avec rendu côté serveur et génération de sites statiques.

Notion

Notion est une application de productivité tout-en-un qui combine des fonctionnalités de prise de notes, de gestion de projets, de bases de données et de collaboration en équipe.

P

Prompt

Un prompt est une instruction ou une entrée donnée à un système informatique ou à une intelligence artificielle pour générer une réponse ou exécuter une action.

Puppeteer

Puppeteer est une bibliothèque Node.js qui permet d'automatiser les actions sur des navigateurs web, notamment Chrome, pour le scraping et les tests d'interface utilisateur.

Python

Python est un langage de programmation polyvalent et de haut niveau, apprécié pour sa syntaxe simple et sa lisibilité, utilisé dans divers domaines tels que le développement web, l'analyse de données, l'intelligence artificielle et l'automatisation des tâches.

R

React.js

React.js est une bibliothèque JavaScript utilisée pour construire des interfaces utilisateur interactives et dynamiques, principalement pour les applications web.

Retail

Le Retail ou "commerce de détail" est la branche de l'entreprise qui s'occupe de la présence physique de cette dernière (les succursales).

S

Script

Un script est un fichier contenant une série d'instructions exécutées par un programme ou un interpréteur pour automatiser des tâches spécifiques.

Selenium

Selenium est un outil d'automatisation de tests pour les applications web, permettant de contrôler les navigateurs web par le biais de scripts.

T

Terminal

Un terminal est une interface utilisateur textuelle qui permet d'interagir avec le système d'exploitation en saisissant et en exécutant des commandes.

U

URL

Une URL (Uniform Resource Locator) est l'adresse web qui spécifie l'emplacement d'une ressource sur Internet.

User-friendly

User-friendly désigne une interface ou un produit facile à utiliser et intuitif pour les utilisateurs.

V

VB.NET

VB.NET est un langage de programmation orienté objet développé par Microsoft pour créer des applications sur la plateforme .NET.

W

Web Scraping

Le web scraping est une technique d'extraction de données à partir de sites web en utilisant des scripts pour collecter et analyser le contenu automatiquement.

Table des abréviations et acronymes

API - Application Programming Interface (Interface de programmation d'application)

CSS - Cascading Style Sheets (Feuilles de style en cascade)

ERP - Enterprise Resource Planning (Planification des ressources de l'entreprise)

HTML - HyperText Markup Language (Langage de balisage hypertexte)

IA - Intelligence Artificielle

TB - Travail de Bachelor

W3C - World Wide Web Consortium

Bibliographie

AIT ABDELLAH, Mohammed, 2023. Matrice des risques: Modèle Excel et tutoriel vidéo. *Blog Gestion de Projet* [en ligne]. 15 novembre 2023. Disponible à l'adresse : https://blog-gestion-de-projet.com/matrice-gestion-des-risques/ [consulté le 17 mai 2024].

ASANA, 2024. Modèle de matrice des risques : évaluer efficacement les risques d'un projet pour assurer sa réussite [2024] • Asana. *Asana* [en ligne]. 21 février 2024. Disponible à l'adresse : https://asana.com/fr/resources/risk-matrix-template [consulté le 17 mai 2024].

BATAVIYA, Avani, 2023. Office 365 login in Next.js using NextAuth.js. *Simform Engineering* [en ligne]. 2 mai 2023. Disponible à l'adresse : https://medium.com/simform-engineering/office-365-login-in-next-js-using-nextauth-js-beb1f18e05d8 [consulté le 13 mai 2024].

BETTENBUK, Zoltan, 2022. Cheerio vs Puppeteer for Web Scraping - Which One Is for You. *ScraperAPI* [en ligne]. 31 mars 2022. Disponible à l'adresse : https://www.scraperapi.com/blog/cheerio-vs-puppeteer/ [consulté le 21 mai 2024].

BILLETER, Tobias, 2023. What Galaxus stands for. *Galaxus* [en ligne]. 11 janvier 2023. Disponible à l'adresse : https://www.galaxus.de/en/page/what-galaxus-stands-for-25759 [consulté le 21 mai 2024].

BORCHERS, Daniel, 2024. Une fonctionnalité d'IA montre les avantages et inconvénients des produits. *Galaxus* [en ligne]. 16 mai 2024. Disponible à l'adresse : https://www.galaxus.ch/fr/page/une-fonctionnalite-dia-montre-les-avantages-et-inconvenients-des-produits-33091 [consulté le 21 mai 2024].

DIGITEC GALAXUS AG, [sans date]. Digitec Galaxus AG - digitec. *digitec* [en ligne]. [sans date]. Disponible à l'adresse : https://www.digitec.ch/fr/wiki/528 [consulté le 30 avril 2024].

GOOGLE, 2023. ChromeDriver - WebDriver for Chrome - Downloads. *Chrome for Developpers* [en ligne]. 15 novembre 2023. Disponible à l'adresse : https://chromedriver.chromium.org/downloads [consulté le 1 mai 2024].

GOOGLE, 2024a. Puppeteer | Puppeteer. *Puppeteer* [en ligne]. 2024. Disponible à l'adresse : https://pptr.dev/ [consulté le 10 mai 2024].

GOOGLE, 2024b. Chrome for Testing availability. *Chrome for Testing* [en ligne]. 22 mai 2024. Disponible à l'adresse : https://googlechromelabs.github.io/chrome-for-testing/ [consulté le 1 mai 2024].

GOOGLE CLOUD, [sans date]. Qu'est-ce que l'analyse prédictive et comment fonctionne-t-elle ? | Google Cloud. *Google Cloud* [en ligne]. [sans date]. Disponible à l'adresse : https://cloud.google.com/learn/what-is-predictive-analytics?hl=fr [consulté le 7 mai 2024].

JAMESHORAB, 2023. The chromedriver version cannot be discovered selenium python. *Stack Overflow* [en ligne]. 1 novembre 2023. Disponible à l'adresse : https://stackoverflow.com/q/77401357 [consulté le 1 mai 2024].

KINSTA, 2023. Qu'est-ce que le Web Scraping ? Comment extraire légalement du contenu web. *Kinsta* [en ligne]. 30 août 2023. Disponible à l'adresse : https://kinsta.com/fr/base-de-connaissances/web-scraping/ [consulté le 10 mai 2024].

LAUSSON, Julien, 2023. Au fait, c'est quoi un prompt ? *Numerama* [en ligne]. 10 février 2023. Disponible à l'adresse : https://www.numerama.com/tech/1264452-au-fait-cest-quoi-un-prompt.html [consulté le 21 mai 2024].

LO CARPENA, Maria Cecilia, 2023. Comment utiliser Google Trends pour l'optimisation des moteurs de recherche afin d'augmenter le trafic sur le site. *SEO.com* [en ligne]. 9 novembre 2023. Disponible à l'adresse : https://www.seo.com/fr/tools/google-trends/ [consulté le 7 mai 2024].

MICROSOFT, [sans date]. ASP.NET Core | Open-source web framework for .NET. *Microsoft* [en ligne]. [sans date]. Disponible à l'adresse : https://dotnet.microsoft.com/en-us/apps/aspnet [consulté le 14 mai 2024 a].

MICROSOFT, [sans date]. Outils de génération Microsoft C++. *Visual Studio* [en ligne]. [sans date]. Disponible à l'adresse : https://visualstudio.microsoft.com/fr/visual-cpp-build-tools/ [consulté le 1 mai 2024 b].

ORACLE, [sans date]. Qu'est-ce que le machine learning ? *Oracle Suisse* [en ligne]. [sans date]. Disponible à l'adresse : https://www.oracle.com/ch-fr/artificial-intelligence/machine-learning/what-is-machine-learning/ [consulté le 7 mai 2024].

PERRIER, Alexis, 2024. Initiez-vous au Machine Learning. *OpenClassrooms* [en ligne]. 23 avril 2024. Disponible à l'adresse : https://openclassrooms.com/fr/courses/8063076-initiez-vous-au-machine-learning [consulté le 7 mai 2024].

PYTHON SOFTWARE FOUNDATION, [sans date]. Python Releases for Windows | Python.org. *python* [en ligne]. [sans date]. Disponible à l'adresse : https://www.python.org/downloads/windows/ [consulté le 1 mai 2024].

RAMADHAN, Hilman, 2023. Web Scraping with Javascript and Nodejs (2024 Guide). *SerpApi* [en ligne]. 25 octobre 2023. Disponible à l'adresse : https://serpapi.com/blog/web-scraping-in-javascript-complete-tutorial-for-beginner/ [consulté le 10 mai 2024].

RICHARDSON, Leonard, [sans date]. Beautiful Soup Documentation — Beautiful Soup 4.4.0 documentation. *Beautiful Soup* [en ligne]. [sans date]. Disponible à l'adresse : https://beautiful-soup-4.readthedocs.io/en/latest/ [consulté le 2 mai 2024].

ROBERT, Jérémy, 2020. Machine Learning : Définition, fonctionnement, utilisations. *Formation Data Science | DataScientest.com* [en ligne]. 18 novembre 2020. Disponible à l'adresse : https://datascientest.com/machine-learning-tout-savoir [consulté le 7 mai 2024].

SELENIUM, 2023a. The Selenium Browser Automation Project. *Selenium* [en ligne]. 17 novembre 2023. Disponible à l'adresse : https://www.selenium.dev/documentation/ [consulté le 22 mai 2024].

SELENIUM, 2023b. Unable to Locate Driver Error. *Selenium* [en ligne]. 17 novembre 2023. Disponible à l'adresse : https://www.selenium.dev/documentation/webdriver/troubleshooting/errors/driver_location/ [consulté le 1 mai 2024].

SERPAPI. Google Search Engine Results API. *SerpApi* [en ligne]. Disponible à l'adresse : https://serpapi.com/search-api [consulté le 13 mai 2024].

SERPAPI, 2024a. SerpApi: Google Search API. *SerpApi* [en ligne]. 2024. Disponible à l'adresse : https://serpapi.com/ [consulté le 13 mai 2024].

SERPAPI, 2024b. SerpApi: Plans and Pricing. *SerpApi* [en ligne]. 2024. Disponible à l'adresse : https://serpapi.com/pricing [consulté le 13 mai 2024].

SHAWN, 2023. Answer to « The chromedriver version cannot be discovered selenium python ». *Stack Overflow* [en ligne]. 1 novembre 2023. Disponible à l'adresse : https://stackoverflow.com/a/77401527 [consulté le 1 mai 2024].

STUDER, Xavier, 2024. IA: test de Claude 3, Mistral Large, GPT-4 Turbo par Perplexity. *Le blog high-tech & telecom de Xavier Studer* [en ligne]. 11 mars 2024. Disponible à l'adresse : https://www.xavierstuder.com/2024/03/ai-le-test-comparatif-de-claude-3-mistral-large-chatgpt-turbo-avec-perplexity/ [consulté le 21 mai 2024].

THE CHEERIO CONTRIBUTORS, 2024. The industry standard for working with HTML in JavaScript | cheerio. *Cheerio* [en ligne]. 2024. Disponible à l'adresse : https://cheerio.js.org/ [consulté le 10 mai 2024].

THOMAS, Gaël, 2023. Web Scraping in JavaScript – How to Use Puppeteer to Scrape Web Pages. *freeCodeCamp.org* [en ligne]. 31 janvier 2023. Disponible à l'adresse : https://www.freecodecamp.org/news/web-scraping-in-javascript-with-puppeteer/ [consulté le 10 mai 2024].

VERCEL, 2024. Next.js by Vercel - The React Framework. *Next.js* [en ligne]. 2024. Disponible à l'adresse : https://nextjs.org/ [consulté le 14 mai 2024].

W3SCHOOLS, [sans date]. HTML Tables. *W3Schools* [en ligne]. [sans date]. Disponible à l'adresse : https://www.w3schools.com/html/html_tables.asp [consulté le 9 mai 2024 a].

W3SCHOOLS, [sans date]. Python Dictionaries. *W3Schools* [en ligne]. [sans date]. Disponible à l'adresse : https://www.w3schools.com/python/python_dictionaries.asp [consulté le 1 mai 2024 b].

WALULYA FRANCIS, 2024. Answer to « The chromedriver version cannot be discovered selenium python ». *Stack Overflow* [en ligne]. 31 mars 2024. Disponible à l'adresse : https://stackoverflow.com/a/78252438 [consulté le 1 mai 2024].

WAPPALYZER, 2024a. Technologies used on erp.digitecgalaxus.ch - Wappalyzer. *Wappalyzer* [en ligne]. 13 mai 2024. Disponible à l'adresse : https://www.wappalyzer.com/lookup/erp.digitecgalaxus.ch/ [consulté le 13 mai 2024].

WAPPALYZER, 2024b. Technologies used on galaxus.ch - Wappalyzer. *Wappalyzer* [en ligne]. 13 mai 2024. Disponible à l'adresse : https://www.wappalyzer.com/lookup/galaxus.ch/ [consulté le 13 mai 2024].

WIKIPÉDIA, 2023a. Commerce de détail. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Commerce_de_d%C3%A9tail&oldid=210943227 [consulté le 23 mai 2024]. Page Version ID: 210943227

WIKIPÉDIA, 2023b. Framework. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Framework&oldid=211016426 [consulté le 14 mai 2024]. Page Version ID: 211016426

WIKIPÉDIA, 2024a. Visual Basic .NET. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Visual_Basic_.NET&oldid=211175153 [consulté le 23 mai 2024]. Page Version ID: 211175153

WIKIPÉDIA, 2024b. ASP.NET. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=ASP.NET&oldid=212101147 [consulté le 14 mai 2024]. Page Version ID: 212101147

WIKIPÉDIA, 2024c. Framework Web. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Framework_Web&oldid=212375435 [consulté le 14 mai 2024]. Page Version ID: 212375435

WIKIPÉDIA, 2024d. Next.js. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Next.js&oldid=212857168 [consulté le 14 mai 2024]. Page Version ID: 212857168

WIKIPÉDIA, 2024e. Langage de script. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Langage_de_script&oldid=213367546 [consulté le 23 mai 2024]. Page Version ID: 213367546

WIKIPÉDIA, 2024f. F Sharp. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=F_Sharp&oldid=213578286 [consulté le 23 mai 2024]. Page Version ID: 213578286

WIKIPÉDIA, 2024g. Terminal (informatique). *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Terminal_(informatique)&oldid=213607401 [consulté le 23 mai 2024]. Page Version ID: 213607401

WIKIPÉDIA, 2024h. Node.js. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Node.js&oldid=213696620 [consulté le 23 mai 2024]. Page Version ID: 213696620

WIKIPÉDIA, 2024i. Uniform Resource Locator. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Uniform_Resource_Locator&oldid=213685665 [consulté le 23 mai 2024]. Page Version ID: 213685665

WIKIPÉDIA, 2024j. Apprentissage automatique. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Apprentissage_automatique&oldid=214475606 [consulté le 7 mai 2024]. Page Version ID: 214475606

WIKIPÉDIA, 2024k. C Sharp. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=C_Sharp&oldid=214559489 [consulté le 23 mai 2024]. Page Version ID: 214559489

WIKIPÉDIA, 2024l. Progiciel de gestion intégré. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Progiciel_de_gestion_int%C3%A9gr%C3%A9&oldid=214597674 [consulté le 23 mai 2024]. Page Version ID: 214597674

WIKIPÉDIA, 2024m. Cookie (informatique). *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Cookie_(informatique)&oldid=214773115 [consulté le 23 mai 2024]. Page Version ID: 214773115

WIKIPÉDIA, 2024n. Interface de programmation. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Interface_de_programmation&oldid=214840327 [consulté le 23 mai 2024]. Page Version ID: 214840327

WIKIPÉDIA, 2024o. Cmd. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Cmd&oldid=214863970 [consulté le 23 mai 2024]. Page Version ID: 214863970

WIKIPÉDIA, 2024p. Python (langage). *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Python_(langage)&oldid=215128850 [consulté le 23 mai 2024]. Page Version ID: 215128850

WIKIPÉDIA, 2024q. Feuilles de style en cascade. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Feuilles_de_style_en_cascade&oldid=215257022 [consulté le 23 mai 2024]. Page Version ID: 215257022

WIKIPÉDIA, 2024r. *Hypertext Markup Language*. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Hypertext_Markup_Language&oldid=215256315 [consulté le 23 mai 2024]. Page Version ID: 215256315

WIKIPÉDIA, 2024s. *Web scraping*. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Web_scraping&oldid=215252374 [consulté le 23 mai 2024]. Page Version ID: 215252374

WIKIPÉDIA, 2024t. JavaScript. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=JavaScript&oldid=215279721 [consulté le 23 mai 2024]. Page Version ID: 215279721

WIKIPÉDIA, 2024u. React. *Wikipédia* [en ligne]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=React&oldid=215284403 [consulté le 23 mai 2024]. Page Version ID: 215284403

Annexes

Annexe 1 : Sondage vierge

Annexe 2 : Résultats du sondage

Annexe 1 : Sondage vierge

Automatisation du stock cible

Hello there !

Je me présente, je m'appelle Nicolas Meuwly. Cela fait maintenant un peu plus de trois ans que je travaille à la succursale de Lausanne, tout en poursuivant mes études en Ingénierie des médias à la HEIG-VD à Yverdon-les-Bains. Je suis actuellement en train de réaliser mon travail de Bachelor, qui constitue la dernière étape avant l'obtention de mon diplôme.

Dans le cadre de ce projet, je développe une solution visant à automatiser la sélection des produits destinés au stock cible et à enregistrer ces produits dans l'ERP de l'entreprise. Ce travail représente pour moi l'opportunité de m'attaquer à une problématique concrète et de tenter de trouver une solution adéquate. Je tiens à souligner que, bien que je m'efforce de proposer une solution optimale, je n'ai pas la certitude, à ce stade, de pouvoir offrir la solution idéale et définitivement prête à l'emploi à la fin de ce projet. Mon travail se concentre avant tout sur l'exploration, la recherche et l'expérimentation, et j'aspire à développer un prototype testable.

Le concept de la solution envisagée consiste à générer une liste de produits en fonction de divers critères, puis à automatiser l'enregistrement de ces produits comme stock cible dans l'ERP, après validation par l'utilisateur. Par le biais de ce sondage, je souhaite affiner ma compréhension de la problématique et mieux cerner vos besoins ainsi que les fonctionnalités que vous attendez d'une telle solution.

Pour les questions qui vont suivre, je vous invite à imaginer la solution idéale selon vous et à choisir les réponses qui vous semblent les plus appropriées. Vos réponses sont anonymes et seront utilisées exclusivement dans le cadre de ce projet.

Ce sondage est également disponible en versions allemande et anglaise, traduites à l'aide de DeepL.

Description de la solution

1. De quelle manière devrait être géré le stock cible ? *

- ◯ Spécifique par succursale (chaque succursale gère son propre stock cible)
- ◯ Commun pour toutes les succursales (le stock cible est géré de façon globale et est le même pour tout le monde)
- ◯ Dynamique (le stock cible peut être initialement créé pour tout le monde mais chaque succursale peut avoir sa gestion interne par la suite)

2. Comment préféreriez-vous interagir avec la solution ? *

Les images sont là à titre d'exemples et ne reflètent pas la solution finale.

- ◯ En tapant des commandes dans un terminal
- ◯ En effectuant des actions à partir d'une interface

3. Comment la solution doit-elle atteindre son but ? *

- ◯ En un clic ou en une commande, tout faire et vous n'avez qu'à valider à la fin.
- ◯ Aller pas à pas et vous laisser le contrôle sur chaque étape du processus.
- ◯ Autre

Sélection des articles

4. Quel(s) produit(s) serait(ent) envisageable(s) pour le stock cible ? *

Triez les types de produits du plus au moins important.

- Produits relatifs à la saison
- Produits issus des meilleures ventes
- Produits issus des offres du jour
- Produits faisant partis des nouveautés
- Produits étant en tendances
- Produits ayant de bonnes évaluations

5. Avez-vous d'autres suggestions de produits ?

6. Serait-il pertinent d'inclure le client dans le processus de sélection des produits ? (Via des sondages sur le site par exemple) *

Choisissez le niveau de pertinence entre 1 (pas pertinent) et 5 (très pertinent)

1	2	3	4	5

Fonctionnalités et suggestions

7. Quelles fonctionnalités faudrait-il inclure dans la solution ? *

Ce sont, pour la plupart, des fonctionnalités liées à la sélection de produits pour le stock cible.

- [ ] Choisir les catégories de produits (catégories, sous-catégorie, etc)
- [ ] Choisir les critères de sélection (bestseller, nouveautés, etc)
- [ ] Sélectionner la date de début, de fin et la quantité
- [ ] Possibilité de choisir une période de disponibilité et une quantité différentes en fonction des produits
- [ ] Possibilité d'ajouter des articles après coup (le processus est terminé et enregistré dans l'ERP mais vous souhaitez rajouter d'autres articles)
- [ ] Possibilité de modifier la liste pendant sa création et après sa finalisation
- [ ] Choisir entre un mode de sélection automatique ou manuel (automatique prends tous les critères et propose une liste et manuel vous accompagne étape par étape dans la sélection)
- [ ] Possibilité de choisir les succursales dans le but d'avoir un stock cible indépendant par succursale
- [ ] Avoir un tableau de bord regroupant différentes données (état du stock cible, temps restant, etc)
- [ ] Toutes les fonctionnalités ci-dessus
- [ ] Autre

8. Avez-vous des remarques ou des suggestions concernant ce sujet ?

Tests utilisateurs

9. Souhaitez-vous être contacté ultérieurement pour tester un prototype et/ou maquette de la solution ?

   Si oui, merci de d'inscrire votre adresse e-mail dans le champ ci-dessous.

Ce contenu n'a pas été créé ni n'est approuvé par Microsoft. Les données que vous soumettez sont envoyées au propriétaire du formulaire.

Microsoft Forms

Annexe 2 : Résultats du sondage

Automatisation du stock cible

6 Réponses 04:08 Durée moyenne de remplissage Actif État

1. De quelle manière devrait être géré le stock cible ?

- Spécifique par succursale (chaq... 1
- Commun pour toutes les succur... 1
- Dynamique (le stock cible peut ... 4

2. Comment préféreriez-vous interagir avec la solution ?

- En tapant des commandes dans... 0
- En effectuant des actions à parti... 6

3. Comment la solution doit-elle atteindre son but ?

- En un clic ou en une commande... 5
- Aller pas à pas et vous laisser le ... 1
- Autre 0

4. Quel(s) produit(s) serait(ent) envisageable(s) pour le stock cible ?

1 Produits issus des meilleures ve...
2 Produits étant en tendances
3 Produits relatifs à la saison
4 Produits faisant partis des nouv...
5 Produits ayant de bonnes évalu...
6 Produits issus des offres du jour

5. Avez-vous d'autres suggestions de produits ?

1
Réponses

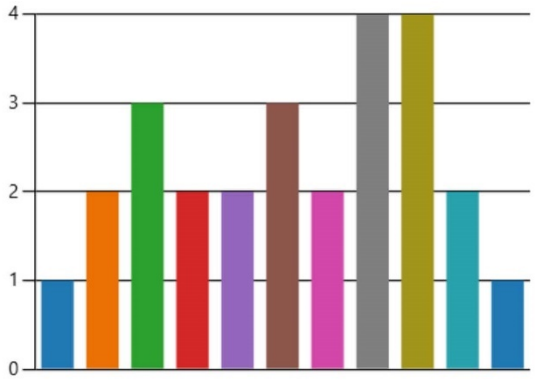
Dernières réponses

6. Serait-il pertinent d'inclure le client dans le processus de sélection des produits ? (Via des sondages sur le site par exemple)

3.50
Évaluation moyenne

7. Quelles fonctionnalités faudrait-il inclure dans la solution ?

- Choisir les catégories de produit... 1
- Choisir les critères de sélection (... 2
- Sélectionner la date de début, d... 3
- Possibilité de choisir une périod... 2
- Possibilité d'ajouter des articles ... 2
- Possibilité de modifier la liste pe... 3
- Choisir entre un mode de sélecti... 2
- Possibilité de choisir les succurs... 4
- Avoir un tableau de bord regrou... 4
- Toutes les fonctionnalités ci-des... 2
- Autre 1



8. Avez-vous des remarques ou des suggestions concernant ce sujet ?

0
Réponses

Dernières réponses

9. Souhaitez-vous être contacté ultérieurement pour tester un prototype et/ou maquette de la solution ?

3
Réponses

Dernières réponses

Annexe 6

Reproduction de l'affiche

HEIG VD HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Travail de Bachelor 2024

Automatisation du stock cible des succursales de Digitec Galaxus AG

Contexte

Digitec Galaxus AG, fondée en 2001, opère via digitec.ch et galaxus.ch, offrant une vaste gamme de produits. Elle possède 10 succursales en Suisse où les clients peuvent retirer leurs commandes, obtenir des conseils, retourner des produits et recevoir de l'aide. Chaque succursale dispose d'un stock permanent appelé « stock cible ».

Problématique

La sélection et l'enregistrement des articles du stock cible sont manuels, nécessitant une intervention humaine pour examiner les données et les enregistrer dans l'ERP. Ce processus est long, répétitif et techniquement contraignant, rendant la gestion du stock difficile pour tous les employés.

Objectifs

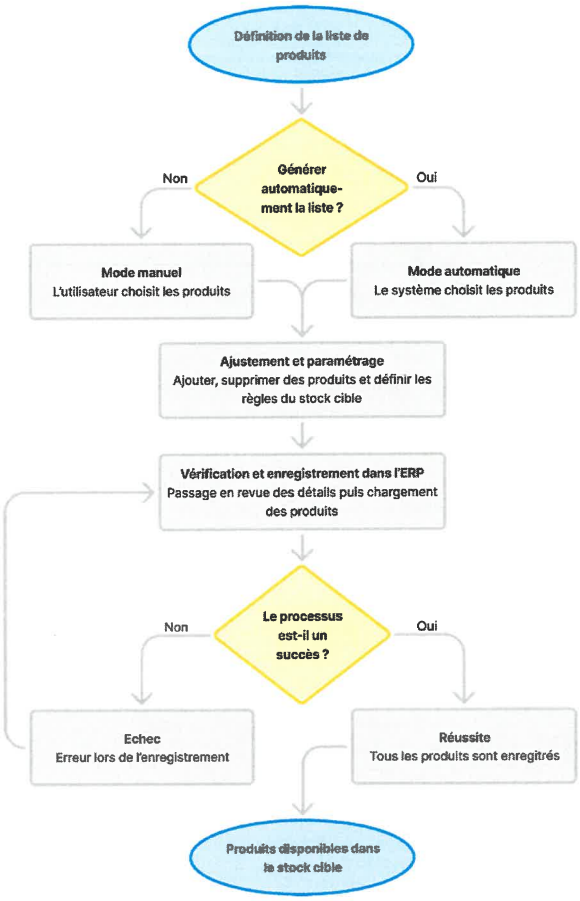
L'objectif est d'automatiser le processus lié au stock cible, de la sélection des produits en fonction de critères prédéfinis à leur enregistrement dans l'ERP. Cela vise à réduire les tâches répétitives et les erreurs humaines, tout en offrant une interface intuitive et accessible à tous les utilisateurs, même sans compétences techniques.

Méthodologie

Le développement a commencé par une analyse des besoins et des processus existants, suivie de recherches sur les technologies disponibles. Une première version de l'interface a été créée et testée auprès des utilisateurs, permettant d'affiner l'interface pour une expérience optimale. Des tests techniques ont vérifié la faisabilité du projet, en utilisant des technologies comme Puppeteer et Cheerio pour le web scraping, et en privilégiant une solution web.

Solution

La réponse à la problématique réside dans une maquette interactive développée avec Figma. Cette maquette simule la gestion du stock cible, de sa définition à son enregistrement dans l'ERP, en générant une liste d'articles basée sur des critères prédéfinis. En complément, une première partie du développement a commencé avec Next.js, le même framework web utilisé pour les plateformes de vente. Enfin, une série de recommandations a été rédigée pour améliorer les résultats futurs, ainsi qu'un guide d'utilisation pour accompagner les futurs utilisateurs.



Auteur : Nicolas Meuwly
Répondant externe : Steve Cosandey
Prof. responsable : Laurent Bolli
Sujet proposé par : Digitec Galaxus AG

Hes·so Haute Ecole Spécialisée de Suisse occidentale

HEIG-VD © 2024, filière Ingénierie des Médias

Annexe 7

Copie du formulaire de Clause de confidentialité

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Confidentialité liée au Travail de Bachelor (TB)

Diplômant :	Nicolas Meuwly
Titre du travail de Bachelor :	Automatisation du stock cible des succursales de Digitec Galaxus AG
Enseignant responsable du TB :	Laurent Bolli
Entreprise partenaire :	Digitec Galaxus AG

Tous les TB sont déposés à la Bibliothèque de la HEIG-VD qui en gère l'archivage et la consultation. Quel que soit le niveau de confidentialité du TB, le nom du diplômant, le nom de l'enseignant responsable, le titre du TB et le résumé publiable figurent dans tous les documents de présentation des TB ainsi que sur la plateforme de consultation des TB (http://tb.heig-vd.ch). L'enseignant responsable veille à ce que le titre du TB et le résumé publiable soient rédigés conformément au niveau de confidentialité voulu.
Les TB peuvent être soumis à un logiciel anti-plagiat. Dans ce cas, leur contenu sera traité de manière confidentielle.

⊗ **Le TB n'est pas confidentiel**

Outre les informations mentionnées ci-dessus, les documents de présentation du TB contiennent également le nom de l'entreprise partenaire, le résumé publiable et une affiche. Le TB peut être consulté sur la plateforme des TB.

○ **Le TB est confidentiel.**

Les conditions suivantes de diffusion des informations sont appliquées :

*Aucune consultation ou emprunt du TB n'est permis hormis par l'enseignant responsable du TB et le diplômant qui s'engagent à ne pas faire usage des informations mises à leur disposition. Le TB porte la mention « **confidentiel** ».*

○ Oui ○ Non *Nous acceptons que **le nom de l'entreprise partenaire** figure dans les documents publiés (titre, résumé, affiche, etc.), ainsi que dans la plateforme de consultation des TB.*

○ Oui ○ Non *Nous acceptons que **l'affiche** du TB figure sur la plateforme de consultation des TB (l'affiche est au préalable validée par l'entreprise partenaire).*

Dans tous les cas, un accord de confidentialité doit être signé par le diplômant, l'expert et toutes les personnes participant à l'évaluation du TB.

Nous déclarons accepter les conditions de diffusion du Travail de Bachelor indiquées.

Diplômant

Date	Nom et signature
07.08.2024	Meuwly Nicolas N. Meuwly

Enseignant responsable

Date	Nom et signature
07.08.2024	BOLLI LAURENT

Entreprise partenaire

Date	Nom et signature
03.08.2024	Cosandey

N.B.: Ce document fait partie intégrante du cahier des charges du TB.
La forme masculine est utilisée comme genre neutre et désigne à la fois les hommes et les femmes.

Annexe 8

Copie du procès-verbal du bilan intermédiaire

Bilan intermédiaire

Procès-verbal de la séance

Date : 18.06.2024

Lieu : Digitec Galaxus AG, Rue du Grand-Pré 2B, 1007 Lausanne

Début de la séance : 09h00

Fin de la séance : 10h25

Durée de la séance : 1h25

Participants :

- Steve Cosandey, répondant externe
- Laurent Bolli, enseignant responsable
- Nicolas Meuwly, étudiant

Ordre du jour :

1. Approbation de l'ordre du jour
2. Discussion autour de la clause de confidentialité
3. Présentation de l'avancement du travail
4. Explication des problèmes rencontrés
5. Présentation des adaptations suite aux problèmes
6. Passage en revue des prochaines étapes
7. Proposition d'affiche et de résumé du TB
8. Discussion sur le déroulement du TB
9. Remarques et questions

1. Approbation de l'ordre du jour

L'ordre du jour a été communiqué au préalable par e-mail et est approuvé par l'ensemble des participants.

2. Discussion autour de la clause de confidentialité

Le travail sera-t-il confidentiel ? Dans l'état actuel, il n'expose pas de données sensibles de l'entreprise, mais avec l'enregistrement des produits dans l'ERP, la structure de ce dernier sera exposée. Sous quelles conditions le travail pourrait-il être non confidentiel ?

Steve Cosandey n'est pas opposé à ce que le travail ne soit pas confidentiel, cela dépendra de l'évolution du projet et de son orientation. Tant que des données internes ne sont pas exposées, que les éventuelles captures d'écran masquent les éléments confidentiels, et que les noms ou les adresses e-mail des collaborateurs de l'entreprise ne soient pas affichés, le travail pourrait être non confidentiel.

Laurent Bolli est d'accord avec ce qui précède et mentionne qu'il serait préférable que le travail ne soit pas confidentiel, ce que recommande également l'école. Il est tout à fait envisageable de mettre en place des mesures pour que cela soit possible.

Ainsi, il a été décidé de reporter ce point à plus tard, lorsque le projet sera plus avancé et que la majorité des points importants auront été réglés.

3. Présentation de l'avancement du travail

Revue de l'avancement du travail. Les tâches suivantes ont été réalisées jusqu'à présent :

- Architecture du projet avec des fonctionnalités de test
- Maquette interactive (version 1)
- Deux tests utilisateurs sur la maquette
- Rédaction du rapport (phase 1)
- Affiche et résumé du TB (version 1)

Laurent Bolli rappelle l'importance des tests utilisateurs dans ce genre de projet. Ils permettent notamment de confirmer ou d'infirmer les interrogations que l'on pourrait avoir.

4. Explication des problèmes rencontrés

Un des premiers problèmes rencontrés concernait l'accès au système de l'entreprise, avec l'accès aux bases de données encore en cours de discussion. Pour ne pas perdre de temps, une solution alternative via le web scraping pour récupérer des listes de produits depuis galaxus.ch a été mise en place. Les premiers tests ont montré que les requêtes pour récupérer les données prennent environ 20 minutes pour obtenir plus de 3000 liens, et plus d'une heure pour obtenir des listes de produits par catégorie.

Le temps est un facteur décisif, car la phase 1, qui prévoyait deux semaines pour préparer la solution techniquement, produire deux maquettes et réaliser des tests utilisateurs, a débordé sur la phase 2, avec une seule maquette produite et les tests utilisateurs non terminés. De plus, le

sondage de la pré-étude ayant reçu peu de réponses, il a fallu relancer les participants pour obtenir des données plus précises.

Lors des tests utilisateurs, la gestion du temps était également un problème. Sur les conseils de Steve Cosandey et Laurent Bolli, il faudra faire attention de bien recadrer les tests et de revoir un peu la dynamique de ces derniers, surtout s'ils se font à distance et dans une autre langue que le français. Il faudra à l'avenir prévoir également plus de temps et mieux anticiper les prochains tests en convenant déjà de dates avec les futurs participants pour éviter d'avoir trop d'absents dus aux vacances d'été. Il serait également intéressant d'avoir l'avis d'autres personnes qui ne connaissent pas nécessairement le sujet afin d'obtenir un regard extérieur.

5. Présentation des adaptations suite aux problèmes

N'ayant pas d'autres solutions actuellement pour récupérer les données, le web scraping sera conservé malgré le temps de traitement relativement long. Le processus pourra être optimisé en stockant certaines données, comme les liens des catégories, pour éviter de les récupérer à chaque fois et ainsi accélérer le processus.

Concernant la maquette de la solution, une seule version sera finalement proposée. L'idée initiale de créer deux maquettes, l'une reprenant le design de l'ERP et l'autre celui des sites e-commerce de l'entreprise, a été abandonnée. En effet, l'ERP va être refondu pour se rapprocher du design des sites e-commerce de l'entreprise, rendant la création de deux maquettes superflues. Cette décision, combinée à un manque de temps, a conduit à la production d'une seule maquette.

Steve Cosandey propose de relancer les personnes concernées pour l'accès à la base de données. Il est essentiel de déterminer au plus vite si cet accès pourra être obtenu afin de faciliter le processus de développement et d'améliorer le résultat final. Il est nécessaire de repenser la stratégie sur ce point et de définir une date butoir pour prendre une décision définitive.

Laurent Bolli souligne qu'il ne faut pas hésiter à le contacter en cas de questions. Le cas des deux maquettes aurait été un bon exemple, car il aurait pu être décidé plus rapidement de ne pas prévoir deux maquettes complètes, mais de rester sur une seule, étant donné que l'intérêt d'en avoir deux était limité. Il rappelle qu'il est normal de prendre des libertés et de faire des adaptations pour un tel outil, surtout lorsque l'on se base sur un design system prévu pour de l'e-commerce à la base.

6. Passage en revue des prochaines étapes

Revue des prochaines étapes du travail. Les tâches suivantes restent à réaliser :

- 2 autres tests utilisateurs sur la maquette
- Adaptation de la maquette à la suite des tests utilisateurs
- Développement des fonctionnalités principales (génération d'une liste de produits et enregistrement dans l'ERP)
- Développement des fonctionnalités secondaires
- Intégration des fonctionnalités avec l'interface et développement de cette dernière
- Implémentation de tests unitaires

- Sessions de tests utilisateurs sur l'outil
- Adaptation et corrections à la suite des tests utilisateurs
- Déploiement de la solution
- Rédaction des recommandations
- Rédaction du guide utilisateur
- Rédaction du rapport

Il a été souligné durant la séance qu'il ne faut pas absolument se fixer l'objectif d'avoir un produit "vendable" à la fin. Le cheminement et les réflexions qui mènent à la solution sont aussi importants que cette dernière. Par exemple, le déploiement de la solution ne doit pas être une priorité, mais plutôt envisager comme un bonus si le temps le permet.

L'idée d'explorer ce que fait la concurrence a été émise durant la séance, que ce soient les produits qu'elle vend le plus ou son fonctionnement de manière générale. Cela pourrait être intégré à la solution afin d'avoir en quelques clics un aperçu de ce que vend la concurrence. Il s'agit également de penser au parcours du client et à la manière dont il recherche des produits.

La maquette a été présentée à Laurent Bolli afin de recueillir ses conseils et avis. Il a notamment suggéré de déplacer certaines options, telles que la langue et la localisation du magasin, dans le menu apparaissant au clic sur l'icône du compte utilisateur afin de regrouper les options générales en un seul endroit. Le placement initial de ces options s'inspirait de ce que l'on retrouve sur les sites e-commerce de l'entreprise. Il a rappelé l'importance de prendre certaines libertés et de ne pas se restreindre à ce qui est présent sur les sites e-commerce, en particulier si les tests révèlent qu'il serait plus judicieux de faire différemment des plateformes de vente.

La question de l'intelligence artificielle et de son utilisation a également été soulevée. Bien que son utilisation puisse être bénéfique à certains moments, il est important de garder un esprit critique quant aux résultats obtenus. Il est préférable d'éviter de l'utiliser pour la rédaction de texte brut, car le résultat est souvent générique et plat, avec une tendance à exagérer en utilisant des mots forts où tout devient crucial ou vital pour l'IA. Il est important de se rappeler qu'il s'agit d'un outil qui peut nous assister sur certains points, mais qui ne peut pas tout faire à notre place.

7. Proposition d'affiche et de résumé du TB

Une ébauche de l'affiche et du résumé du TB est présentée à l'enseignant responsable et au répondant externe.

La différence entre le résumé et l'affiche a été questionnée, et Laurent Bolli a apporté quelques précisions à ce sujet. Il s'avère que le contenu des deux documents est partiellement similaire, mais ils n'ont pas le même objectif. Le résumé est intégré directement au rapport et sera également disponible publiquement, peu importe le niveau de confidentialité du projet, et résume le travail dans les grandes lignes. L'affiche, quant à elle, pourra potentiellement être imprimée en grand format et présente le travail différemment du résumé, avec notamment l'ajout d'images pour illustrer les propos.

Pour Laurent Bolli, le résumé est correct dans son état actuel, mais il suggère de restructurer l'affiche pour inclure également la méthodologie du travail. Il propose de raccourcir la partie sur le contexte pour laisser plus de place à la méthodologie et à la solution. Il émet également l'idée d'ajouter un diagramme ou un schéma expliquant le processus du stock cible.

Ces deux documents seront modifiés et adaptés au fur et à mesure de l'avancement du projet, avant d'être validés par les différentes parties prenantes.

8. Discussion sur le déroulement du TB

Le travail se passe relativement bien jusqu'à présent malgré les problèmes rencontrés. Cependant, il est possible que certains points aient été sous-estimés quant au temps nécessaire pour réaliser ces points. Il a fallu revoir certaines choses à la baisse comme le nombre de maquettes réalisées par exemple même si ce choix était justifié.

Laurent Bolli rappelle que le Travail de Bachelor est une bonne occasion pour s'essayer sur différents domaines et commettre des erreurs. C'est encore une phase d'expérimentation. Il donne également quelques précisions sur la défense orale du travail. Dans la mesure du possible, il faudrait éviter les démonstrations techniques qui dépendent de nombreux facteurs extérieurs (réseau, ordinateur, etc.) et privilégier l'explication de la méthodologie, du cheminement et l'évolution du projet avec en complément, par exemple, des captures d'écran représentant la solution. Cela permet de ne pas perdre l'attention du public et du temps en cas de problèmes techniques.

Concernant le rapport final, il recommande d'être direct dans le contenu et de ne pas s'éparpiller en allant trop dans les moindres détails. Le but est de présenter la démarche et les résultats de ce travail.

9. Remarques et questions

Tout a été traité lors des points précédents, il n'y a pas eu de remarques ou de questions supplémentaires.