

APPLICATION DE GESTION DE STOCK



PRÉSENTÉ PAR :

Alssadig Ali Hassan Bukhari Et Sonia Ouendessere

CNAM d'Evry -Courcouronnes
(Conservatoire national des arts et métiers)

09 Septembre 2024

PLAN DE LA PRÉSENTATION

1. Introduction
3. Contexte du projet
3. Objectifs principaux
4. Architecture de l'application
5. Choix des Technologies utilisées
6. Schéma de Base de Données
7. Contraintes fonctionnelles
8. API / Fonctionnalités
9. Utilisateurs
10. Test
11. Défis rencontrés
12. Améliorations futures
13. Conclusion

Introduction:

Ce projet vise à développer une application web intuitive qui permet de gérer les stocks et les inventaires d'une entreprise, en optimisant la gestion de ses stocks.

Contexte du Projet

- **Problème :**

Gestion inefficace des stocks, erreurs fréquentes, dans la recherche de composants, perte de temps et de finances.

- **Solution :**

Développer une application qui permet de suivre précisément les composants dans l'entrepôt à travers une interface conviviale, avec une API pour faciliter l'intégration.

Objectifs Principaux

- **Optimisation Gestion des Stocks des accessoires informatiques pour améliorer l'efficacité opérationnelle :**

Rendre les processus de stockage et de retrait plus rapides et plus efficaces.

- **Automatisation :**

Automatiser les tâches répétitives comme l'inventaire des stocks, réduisant ainsi les erreurs humaines.

- **Traçabilité :**

Faciliter la recherche et la traçabilité des références via des fonctionnalités de recherche par référence et par emplacement.

Architecture de l'Application

Frontend : HTML, CSS, JS.

Backend : Utilise Flask pour gérer les requêtes API et envoyer les données à partir de la base de données.

Base de données : SQLite

Choix des Technologies

Backend : Python 3.x .

Base de données : SQLite en développement, pour gérer les bases de données et des mises à jour de manière efficace.

Frontend : Framework React pour un frontend réactif et interactif.

CI/CD : GitHub Actions pour l'intégration continue et Selenium pour les tests automatiques, exécute sur Docker.

Schéma de Base de Données

Les principales tables de la base de données incluent :

Allées : stocke les informations sur les allées disponibles.

Emplacements : stocke les emplacements disponibles dans chaque allée.

Composants : contient les détails sur les composants stockés (nom, référence, date d'arrivée).

Inventories : enregistre l'état des stocks et des emplacements des composants.

- Chaque table est liée par des relations clés étrangères pour assurer l'intégrité des données.

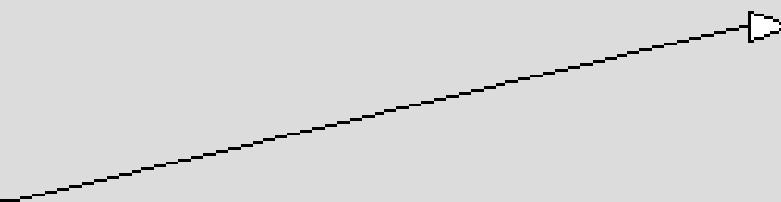
Contraintes fonctionnelles

- Référence unique dans le système : une même référence ne peut pas être ajoutée deux fois.
- Emplacement à usage unique : deux références ne peuvent pas être stocké dans le même emplacement.

SCHUTZ/1

INVENTAIRE	
<u>REF</u>	
Date	
<u>ALLEE_ID</u>	
ID	
id: REF	
acc	
id: Date	
acc	
id: ALLEE_ID	
ID	
ref acc	

EMPLACEMENTS	
<u>ALLEE_ID</u>	
ID	
id: ALLEE_ID	
ID	
acc	



API

API simple permettant de gérer des composants et des emplacements dans une plateforme de gestion.

Diagramme Cas d'Utilisation

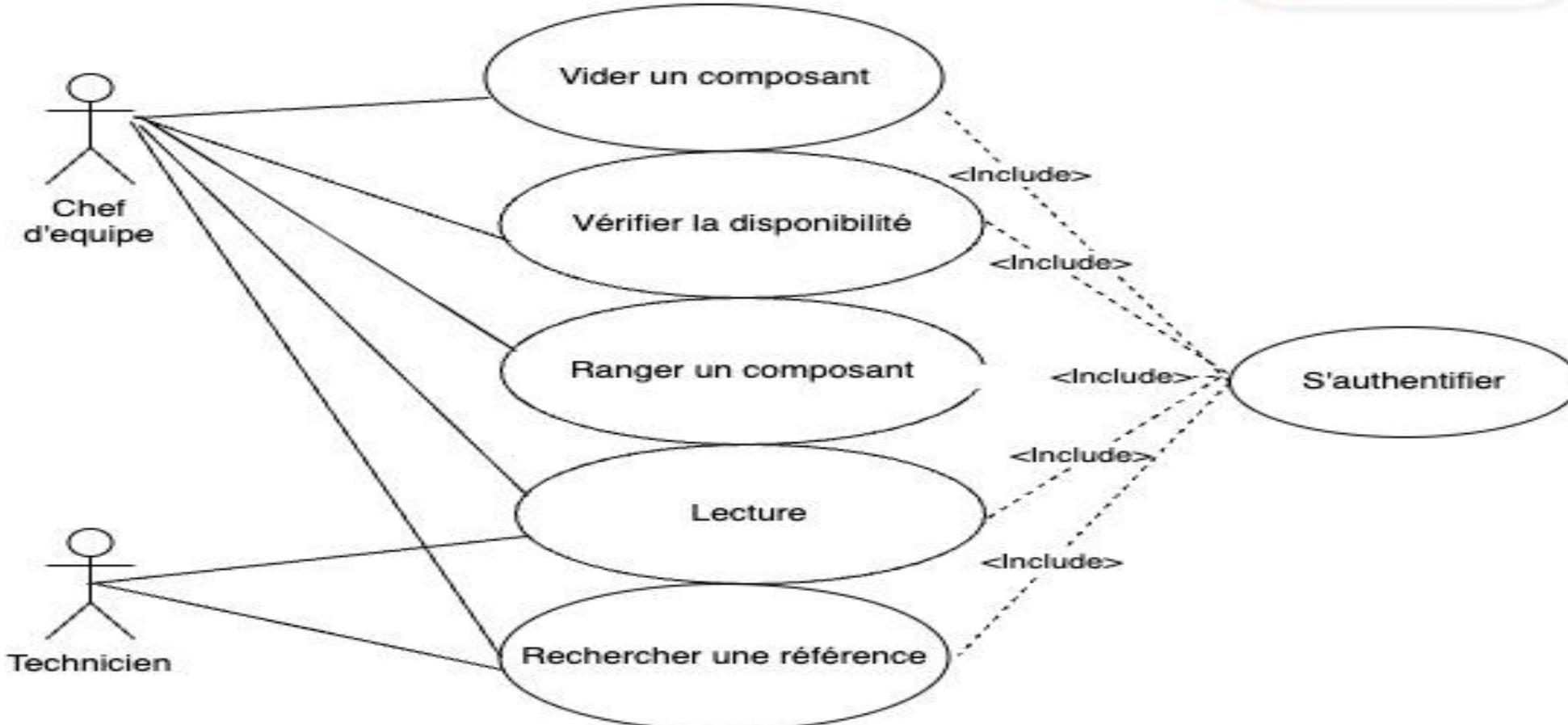
Rôles utilisateurs:

- **Chef d'équipe** : Gestion complète des composants, incluant la recherche, le rangement, la vérification, la suppression, et la mise à jour des stocks.
- **Technicien** : La lecture, et la recherche des composants, mais sans modifier les informations.

USE CASE GESTION DE STOCK

USE CASE GESTION DE STOCK

Made with
Visual Paradigm
For non-commercial use



Fonctionnalités Principales

Gestion des Allées

Affichage de la liste des allées disponibles dans l'entrepôt.

- **Fonctionnalité associée :** Ajout de nouvelles allées.
- **Exemple d'utilisation :** Un utilisateur peut consulter toutes les allées disponibles pour déterminer où stocker un nouveau composant.

Gestion des Emplacements

Liste des emplacements associés à chaque allée.

Fonctionnalité associée : Ajout de nouveaux emplacements dans une allée spécifique.

Exemple d'utilisation : Permet de voir et de modifier les emplacements disponibles dans une allée donnée.

PRINCIPALES PAGES



Bonjour, veuillez vous **identifier** via la page d'authentification.

AUTHENTIFICATION

Authentification

chef

.....

Se connecter

PAGE DES EMPLACEMENTS

Allées Disponibles

Allée A

Allée B

Allée C

Allée D

Allée E

Allée F

Allée G

Allée H

Allée I

PAGE LISTANT LES ALLÉES DISPONIBLES

Inventaire

Référence	Date	ID d'allée	ID d'emplacement
3017935	28/01/2024	UN	209
3017936	29/01/2024	UN	101
3017937	30/01/2022	UN	102
3017938	31/01/2024	B	200
3017939	01/02/2023	B	201
3017940	02/02/2021	B	202
3017941	03/02/2024	C	300
3017942	04/02/2024	C	301
3017943	05/02/2007	C	302
3017944	06/02/2024	D	400
3017945	07/02/2024	D	401
3017946	08/02/2025	D	402
3017947	09/02/2024	E	500
3017948	10/02/2024	E	501
3017949	2024-02-11	E	502
3017950	12/02/2023	F	600
3017951	13/02/2024	F	601
3017952	14/02/2024	F	602
3017953	15/02/2024	G	700
3017954	16/02/2024	G	701
3017955	17/02/2024	G	702
3017956	18/02/2024	H	800
3017957	19/02/2023	H	801
3017958	20/02/2024	H	802
3017959	21/02/2021	ie	900

Fonctionnalités

RANGER UN COMPOSANT



Bienvenue sur la plateforme SCHUTZ

Ranger un Composant

Allée

Emplacement

Référence

Date

Valider

VÉRIFIER LA DISPONIBILITÉ



Vérifier la disponibilité

Allée

E

Emplacement

500

Vérifier

A form interface for checking availability. It has two dropdown menus: one for the aisle ('Allée') containing the value 'E', and another for the location ('Emplacement') containing the value '500'. Below these is a large green button labeled 'Vérifier'.

RECHERCHE UNE REFERENCE

ACCESSIBLE PAR LE CHEF ET LES TECHNICIENS.

The screenshot shows a search interface titled "Rechercher une référence". It features a search bar containing the number "5800" and a green "Rechercher" button below it. A "Retour" link is located at the bottom of the search box. To the left of the search box is a large circular graphic with a gradient from blue to yellow, containing the text "Bienvenue sur la plateforme SCHUTZ". The background of the page is a dark blue gradient.

Bienvenue sur la plateforme SCHUTZ

Rechercher une référence

Référence

5800

Rechercher

Retour

VIDER UN EMPLACEMENT



Bienvenue sur la plateforme SCHUTZ

Vider un emplacement

Allée

H

Emplacement

700

Vider

UTILISATEURS

DEUX RÔLES UTILISATEURS :

- CHEF D'ÉQUIPE

ID : CHEF

MDP : EQUIPE

IL A TOUT LE DROIT POUR TOUTES LES FORMULAIRES

- TECHNICIENS

ID : TECHOS

MDP : TECH1

IL A LE DROIT DE LA FORMULAIRE RECHERCHE ET LA FORMULAIRE LECTURE

CHEF D'ÉQUIPE

Bienvenue, Chef !

Voici les actions disponibles :

[Ranger un composant](#)

[Vider un emplacement](#)

[Rechercher un composant](#)

[État de l'inventaire](#)

[Déconnexion](#)

TECHNICIEN

Bienvenue dans l'équipe !

Voici les actions disponibles :

[Rechercher un composant](#)

[État de l'inventaire](#)

[Déconnexion](#)

TESTS SÉLÉNIUM DANS UN DOCKER

```
node1] (local) root@10.0.22.2 ~
  nano testrech_OK.py
node1] (local) root@10.0.22.2 ~
  nano run_selenium.sh
node1] (local) root@10.0.22.2 ~
  chmod +x testrech_OK.py
node1] (local) root@10.0.22.2 ~
  chmod +x run_selenium.sh
node1] (local) root@10.0.22.2 ~
  ./run_selenium.sh
lateforme SCHUTZ
echerche réussie.
node1] (local) root@10.0.22.2 ~
```

```
node1] (local) root@10.0.23.2 ~
  nano testrech_NOK.py
node1] (local) root@10.0.23.2 ~
  chmod +x testrech_NOK.py
node1] (local) root@10.0.23.2 ~
  ./run_selenium.sh
lateforme SCHUTZ
as de resultat.
node1] (local) root@10.0.23.2 ~
  █
```

SCRIPT

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.firefox.service import Service
from selenium.webdriver.firefox.options import Options
from selenium.webdriver.common.by import By
import time
# Spécifiez le chemin vers l'exécutable geckodriver
service = Service(executable_path='/usr/local/bin/geckodriver')
# Initialisez les options Firefox
options = Options()
options.add_argument('--headless') # Pour exécuter Firefox en mode headless
# Initialisez le pilote Firefox avec le service et les options
driver = webdriver.Firefox(service=service, options=options)
# Ouvrir le site Python
driver.get("https://jeksos.alwaysdata.net/recherche")
# Afficher le titre de la page
print(driver.title)
# Trouver le champ reference
ref = driver.find_element(By.NAME, "reference")
# Effacer le contenu s'il y a du texte dedans
ref.clear()
# Entrer la requête de recherche dans la barre de recherche
ref.send_keys("87654321")
# Trouver l'élément bouton de recherche
envoie_val = driver.find_element(By.NAME, "recherche")
# Simuler la pression de la touche Entrée
envoie_val.send_keys(Keys.RETURN)
# attendre le résultat du serveur
time.sleep(1)
# Trouver le champ resultat nok
try:
```

DÉFIS RENCONTRÉS

Défi :

1) Synchronisation des données entre frontend et backend :

Essentielle pour garantir que les utilisateurs aient accès aux informations les plus récentes sans latence excessive ou incohérences.

- **Latence** entre la mise à jour des données sur le backend et leur affichage côté frontend.
- **Gestion des erreurs** en cas de défaillance du réseau
- **Consistance des données** lorsque plusieurs utilisateurs modifient simultanément les mêmes enregistrements

Solutions :

- WebSockets ou Server-Sent Events (SSE)
- Gestion des conflits de données
- Notifications d'erreur claires

Améliorations Futures

- Intégration avec d'autres systèmes ERP (automatiser la mise à jour des stocks en temps réel, Meilleure visibilité, Réduction des doublons)
- Ajout de fonctionnalités avancées comme la gestion des alertes de stock (Proactivité, Automatisation, Optimisation des coûts)
- Implémentation de fonctionnalités de code-barres pour les opérations de stockage/retrait (Gain de temps, Réduction des erreurs, Traçabilité améliorée)

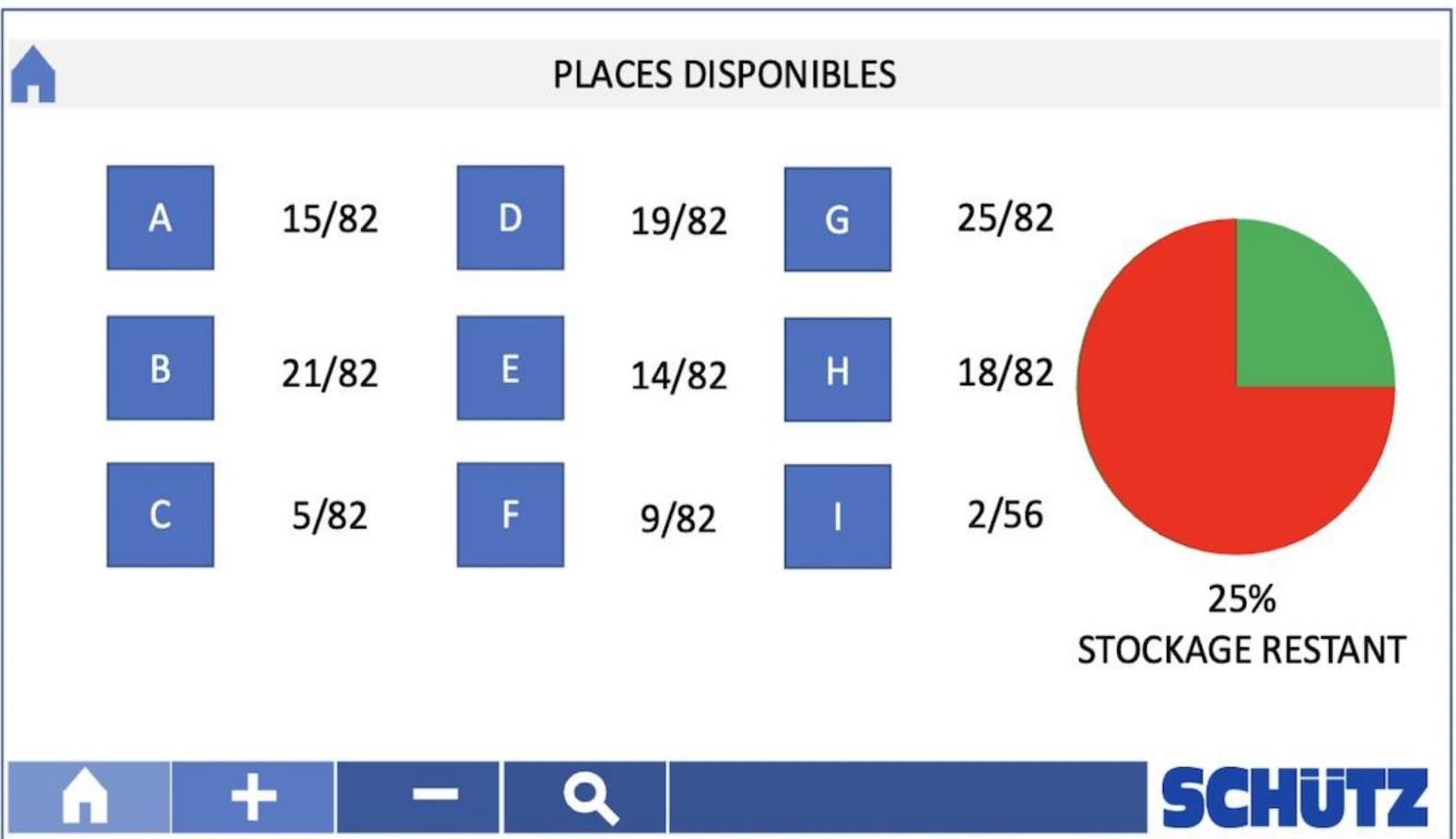
Points forts du projet

Collaboration : La réussite du projet a été en grande partie due à une collaboration efficace entre les membres de l'équipe, facilitée par des communications ouvertes et une coordination claire des tâches.

Conclusion

En conclusion, ce projet a abouti à une plateforme performante pour la gestion des stocks, offrant une interface conviviale, des fonctionnalités automatisées et une synchronisation fluide des données. L'application facilite la gestion efficace des composants en réduisant les erreurs humaines, améliorant la précision des inventaires et optimisant les opérations logistiques. L'impact attendu sur les entreprises est significatif, avec une meilleure visibilité sur les stocks, une réduction des coûts opérationnels et une amélioration de la productivité, contribuant ainsi à une gestion plus intelligente et anticipative des ressources.

Page principale avec vision des places disponible par allée



Liste déroulante des allées (De A à I) et liste déroulantes des emplacements (De 100 à 111 , 200 à 211 , 300 à 311 , 400 à 411 , 500 à 511 , 600 à 611 et 700 à 711)

RANGER UN COMPOSANT

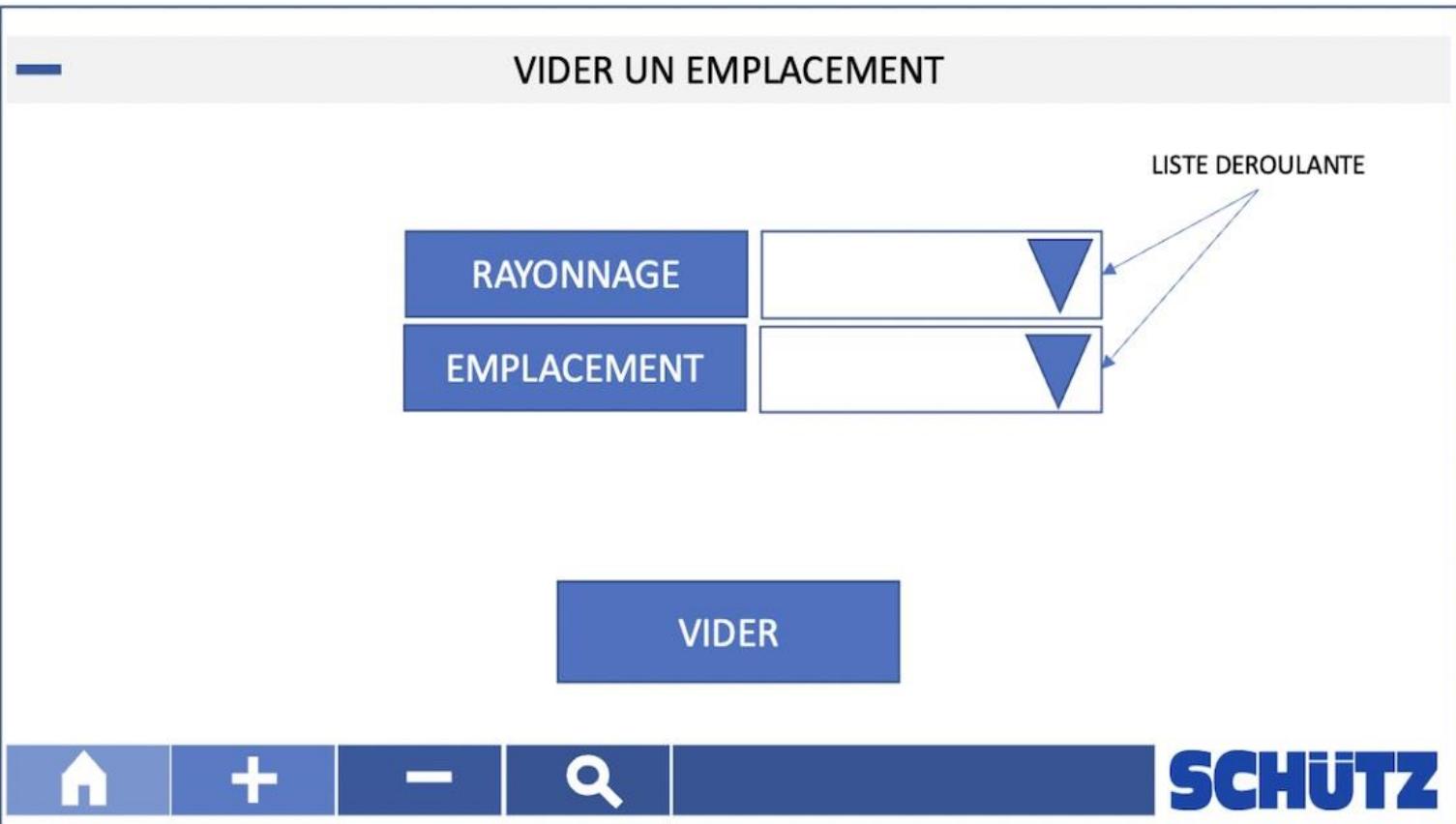
ALLÉE	<input type="text"/>	<i>Exemple A</i>
EMPLACEMENT	<input type="text"/>	<i>209</i>
REFERENCE	<input type="text"/>	<i>3017935</i>
DATE	<input type="text"/>	<i>28/01/24</i>

VALIDER

SCHÜTZ

Liste déroulante des allées (De A à I) et liste déroulantes des emplacements (De 100 à 111 , 200 à 211 , 300 à 311 , 400 à 411 , 500 à 511 , 600 à 611 et 700 à 711)

Liste déroulante des allées (De A à I) et liste déroulantes des emplacements (De 100 à 111 , 200 à 211 , 300 à 311 , 400 à 411 , 500 à 511 , 600 à 611 et 700 à 711)





RECHERCHER UNE REFERENCE

Exemple

REFERENCE

3017935

VALIDER



SCHÜTZ

Recherche d'une référence rangé en stock en prenant en compte la cage la plus ancienne = Respect du FIFO

