Ministère de l'Enseignement Supérieur et de la recherche scientifique
Direction Générale des Etudes Technologiques
Institut Supérieur des Etudes Technologiques de Sfax
Département Technologies de l'Informatique

TP JEE DEMO DOCUMENT

Année Universitaire : 2022/2023

Sommaire

Liste des figures

Introduction

Nous essayons de réaliser les fonctionnalités classiques d'une solution de gestion des stocks : la création de fiches individuelles, le blocage d'un article afin de le vendre ou de réaliser un mouvement, le contrôle des stocks entrants et sortants, l'obtention d'un historique des mouvements de stock, l'émission des bons de livraison et la possibilité d'utiliser les données en réseau

1. Creation de DATABASE

1.1. Requette

```
CREATE TABLE User (
  id INT PRIMARY KEY AUTO_INCREMENT,
  prenom VARCHAR(255),
  login VARCHAR(255),
  password VARCHAR(255)
);
CREATE TABLE SystemeGestionStock (
  id INT PRIMARY KEY AUTO_INCREMENT,
  listeArticles VARCHAR(255),
  commandesAchat VARCHAR(255),
  commandesVente VARCHAR(255)
);

CREATE TABLE Article (
  id INT PRIMARY KEY AUTO_INCREMENT,
  nom VARCHAR(255),
  quantite INT
);

CREATE TABLE CommandeAchat (
  id INT PRIMARY KEY AUTO_INCREMENT
);

CREATE TABLE CommandeAchat_Article (
  id INT PRIMARY KEY AUTO_INCREMENT,
  commandeAchatId INT,
  articleId INT,
  FOREIGN KEY (commandeAchatId) REFERENCES CommandeAchat(id),
  FOREIGN KEY (articleId) REFERENCES Article(id)
);
```

```
CREATE TABLE CommandeVente (
  id INT PRIMARY KEY AUTO_INCREMENT,
  client VARCHAR(255)
);

CREATE TABLE CommandeVente_Article (
  id INT PRIMARY KEY AUTO_INCREMENT,
  commandeVenteId INT,
  articleId INT,
  FOREIGN KEY (commandeVenteId) REFERENCES CommandeVente(id),
  FOREIGN KEY (articleId) REFERENCES Article(id)
);

CREATE TABLE MouvementStock (
  id INT PRIMARY KEY AUTO_INCREMENT,
  articleId INT,
  typeMouvement ENUM('ENTREE', 'SORTIE'),
  FOREIGN KEY (articleId) REFERENCES Article(id)
);

CREATE TABLE BonLivraison (
  id INT PRIMARY KEY AUTO_INCREMENT,
  commandeVenteId INT,
  dateLivraison DATE,
  FOREIGN KEY (commandeVenteId) REFERENCES CommandeVente(id)
);
CREATE TABLE `donnees_commande` (
  `id` INT AUTO_INCREMENT PRIMARY KEY,
  `client` VARCHAR(100) NOT NULL
);
CREATE TABLE `donnees_article` (
  `id` INT AUTO_INCREMENT PRIMARY KEY,
```

`nom` VARCHAR(50) NOT NULL,

`quantite` INT NOT NULL

);

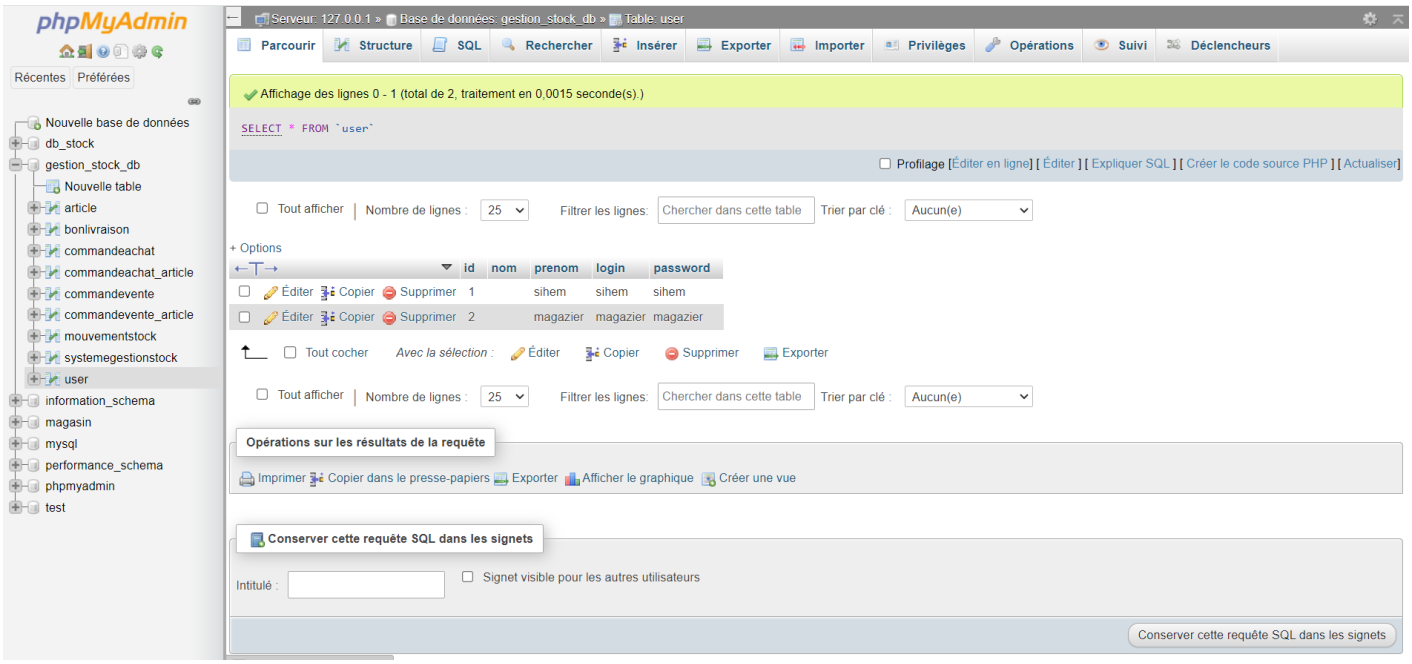

Figure 1. DB

1.2. Diagramme relationnel :

1. Table: SystemeGestionStock
   - Columns:
     - id (Primary Key)

2. Table: Article
   - Columns:
     - id (Primary Key)
     - nom
     - quantite

3. Table: CommandeAchat
   - Columns:

- id (Primary Key)

4. Table: CommandeVente
   - Columns:
     - id (Primary Key)
     - client

5. Table: MouvementStock
   - Columns:
     - id (Primary Key)
     - articleId (Foreign Key referencing Article.id)
     - typeMouvement

6. Table: BonLivraison
   - Columns:
     - id (Primary Key)
     - commandeVenteId (Foreign Key referencing CommandeVente.id)
     - dateLivraison

7. Table: DonneesArticle
   - Columns:
     - id (Primary Key)
     - nom
     - quantite

8. Table: TypeMouvement
   - Columns:
     - id (Primary Key)
     - type (e.g., ENTREE, SORTIE)

9. Table: DonneesCommande
   - Columns:
     - id (Primary Key)

- commandeId (Foreign Key referencing either CommandeAchat.id or CommandeVente.id)
- articleId (Foreign Key referencing Article.id)
- quantite

10. Table: CommandeAchat_Article
    - Columns:
      - commandeAchatId (Foreign Key referencing CommandeAchat.id)
      - articleId (Foreign Key referencing Article.id)

11. Table: CommandeVente_Article
    - Columns:
      - commandeVenteId (Foreign Key referencing CommandeVente.id)
      - articleId (Foreign Key referencing Article.id)

2. Conception

2.1. Diagramme de classs

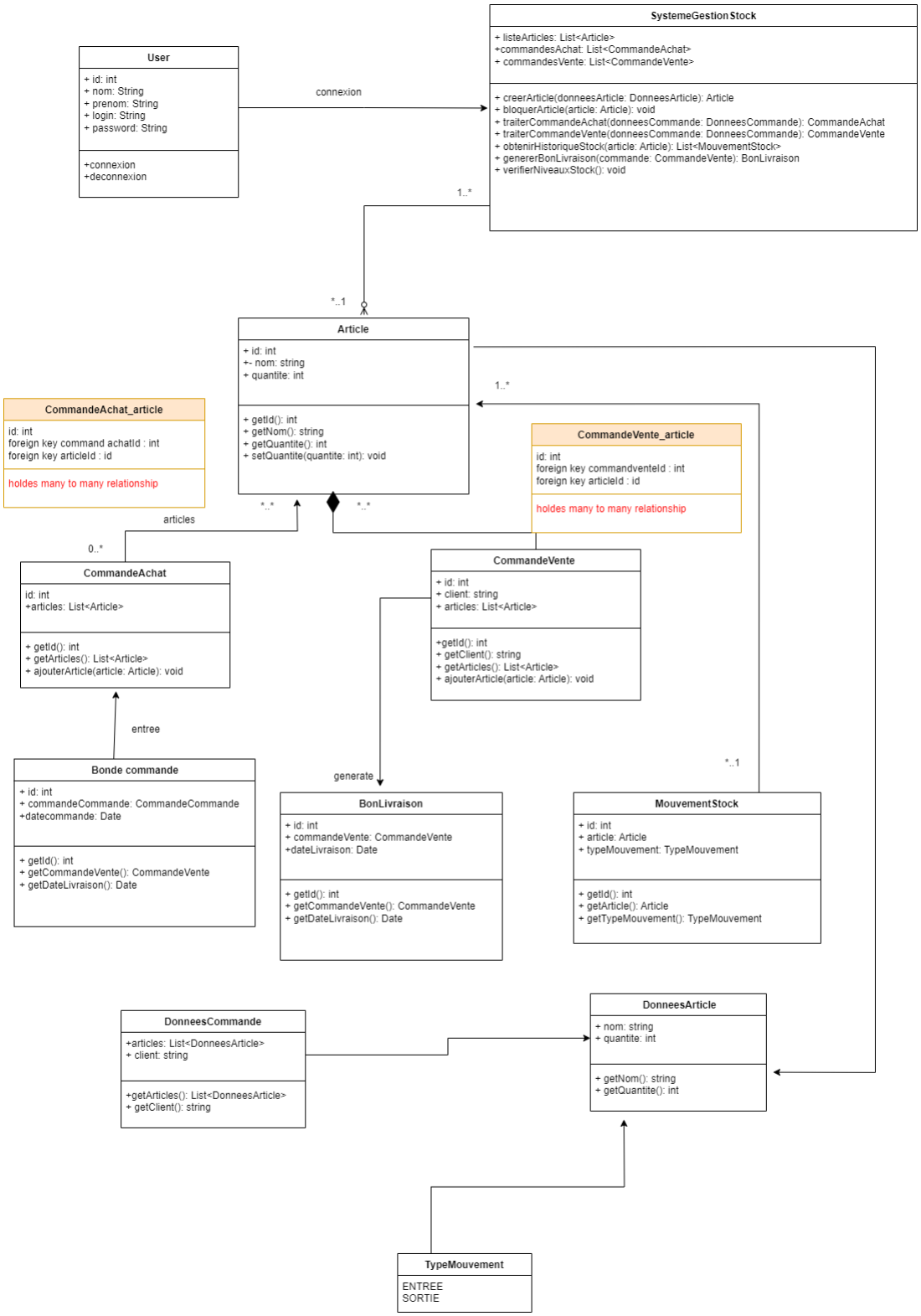

Figure 2-Diagramme de classe

3. Etude de l’architecture:

3.1. Classes de domain

1. Article:
   - Attributes:
     - id (int):
     - nom (String):
     - quantite (int): Quantity of the article in stock.

2. BonLivraison:
   - Attributes:
     - id (int): Unique identifier of the delivery note.
     - commandeVente (CommandeVente): The associated
     - dateLivraison (Date): Date of delivery.

3. CommandeAchat:
   - Attributes:
     - id (int):

- nomFournisseur (String):
- dateCommande (Date):
- articleId (int):

     - articles (List<Article>): List of articles

4. CommandeVente:
   - Attributes:
     - id (int):
     - client (String):
     - articles (List<Article>): List of articles

5. DonneesArticle:
   - Attributes:
     - nom (String):

- quantite (int): Quantity of the article.

6. DonneesCommande:
   - Attributes:
     - id (int):
     - client (String):

7. MouvementStock:
   - Attributes:
     - id (int):
     - article (Article): The associated article.
     - typeMouvement (TypeMouvement): The type of stock movement (ENTREE or SORTIE).

8. TypeMouvement (enum):
   - Constants:
     - ENTREE: stock entry movement.
     - SORTIE: stock exit movement.

3.2. Classes de metier :

metier (business logic) the stock management application:

1. `ArticleMetierInterface`:
   - Methods:
     - `List<Article> listArticles()`: Get a list of all articles.
     - `void addArticle(Article a)`: Add a new article.
     - `Article getArticleById(int id)`: Get an article by its ID.
     - `void updateArticle(Article a)`: Update an existing article.
     - `void deleteArticle(int id)`: Delete an article by its ID.
   - Implementation class: `ArticleMetierImpl`

2.
CommandeAchatMetierInterface`:
   - Methods:

- `void traiterCommandeAchat(PurchaseOrder purchaseOrder)`: Process a purchase order by adding it to the system.
- `void addArticleToCommandeAchat(PurchaseOrder purchaseOrder, Article article)`: Add an article to a purchase order.
- `void removeArticleFromCommandeAchat(PurchaseOrder purchaseOrder, Article article)`: Remove an article from a purchase order.
- `List<CommandeAchat> getCommandeAchats()`: Get a list of all purchase orders.

3.`CommandeVenteMetierInterface`:

- Methods:
- `void traiterCommandeVente(CommandeVente commandeVente)`: Process a sales order by adding it to the system.
- `void addArticleToCommandeVente(CommandeVente commandeVente, Article article)`: Add an article to a sales order.
- `void removeArticleFromCommandeVente(CommandeVente commandeVente, Article article)`: Remove an article from a sales order.
- `List<CommandeVente> getCommandeVentes()`: Get a list of all sales orders.
- `BonLivraison genererBonLivraison(CommandeVente commande)`: Generate a delivery note for a sales order.

4. `StockMetierInterface`:

- Methods:
- `void verifierNiveauxStock()`: Check stock levels and take necessary actions.
- `List<MouvementStock> obtenirHistoriqueStock(Article article)`: Get the stock movement history for an article.
- `void bloquerArticle(Article article)`: Block an article from further transactions.
- Implementation class: `StockMetierImpl`

5.

`BonLivraisonMetierInterface`:

- `List<BonLivraison> listBonsLivraison()`: Get a list of all delivery notes.
- `void addBonLivraison(BonLivraison bonLivraison)`: Add a new delivery note.
- `BonLivraison getBonLivraisonById(int id)`: Get a delivery note by its ID.
- `void updateBonLivraison(BonLivraison bonLivraison)`: Update an existing delivery note.
- `void deleteBonLivraison(int id)`: Delete a delivery note by its ID.

6.BonDeCommandeMetierInterface`:

- Methods:
  - `List<BonDeCommande> listBonDeCommande()`: Get a list of all order data.
  - `void addBonDeCommande(BonDeCommande bonDeCommande)`: Add new order data.
  - `BonDeCommande getBonDeCommandeById(int id)`: Get order data by its ID.
  - `void updateBonDeCommande(BonDeCommande bonDeCommande)`: Update existing order data.
  - `void deleteBonDeCommande(int id)`: Delete order data by its ID.
- Implementation class: `BonDeCommandeMetierImpl`

7. `DonneesArticleMetierInterface`:

- Methods:
  - `List<DonneesArticle> listDonneesArticles()`: Get a list of all article data.
  - `void addDonneesArticle(DonneesArticle donneesArticle)`: Add new article data.
  - `DonneesArticle getDonneesArticleById(int id)`: Get article data by its ID.
  - `void updateDonneesArticle(DonneesArticle donneesArticle)`: Update existing article data.
  - `void deleteDonneesArticle(int id)`: Delete article data by its ID.
- Implementation class: `DonneesArticleMetierImpl`

8. `TypeMouvementMetierInterface`:

- Methods:
  - `List<TypeMouvement> listTypesMouvement()`: Get a list of all stock movement types.
  - `TypeMouvement getTypeMouvementById(int id)`: Get a stock movement type by its ID.
- Implementation class: `TypeMouvementMetierImpl`

3.3. Classes servlets :

Les classes servlet nécessaires pour l'application de gestion des stocks sont les suivantes :

1. `ListeArticlesController` : Un servlet chargé de gérer la requête de listing des articles. Il interagit avec la classe `ArticleMetierImpl` pour récupérer la liste des articles et transfère la requête à la page JSP appropriée.

2. `AjoutModificationArticleController` : Un servlet chargé de gérer la requête d'ajout ou de modification d'un article. Il interagit avec la classe `ArticleMetierImpl` pour effectuer les opérations nécessaires et transfère la requête à la page JSP appropriée.

3. `CommandeAchatController` : Un servlet chargé de gérer la requête de traitement d'une commande d'achat. Il interagit avec la classe `CommandeMetierImpl` pour traiter la commande d'achat et transfère la requête à la page JSP appropriée.

4. `CommandeVenteController` : Un servlet chargé de gérer la requête de traitement d'une commande de vente. Il interagit avec la classe `CommandeMetierImpl` pour traiter la commande de vente et transfère la requête à la page JSP appropriée.

5. `ControleurMouvementsStock` : Un servlet chargé de gérer la requête de visualisation des mouvements de stock. Il interagit avec la classe `StockMetierImpl` pour récupérer les mouvements de stock et transfère la requête à la page JSP appropriée.

6. `ControleurCommandes` : Un servlet chargé de gérer la requête de visualisation et de gestion des commandes. Il interagit avec la classe `CommandeMetierImpl` pour récupérer et traiter les commandes, puis transfère la requête à la page JSP appropriée.

7. `ControleurRapports` : Un servlet chargé de gérer la requête de génération de rapports. Il interagit avec les classes métier pertinentes et les classes d'accès aux données (DAO) pour récupérer les données nécessaires à la génération des rapports, puis transfère la requête à la page JSP appropriée.

8. `ControleurParametres` : Un servlet chargé de gérer la requête de gestion des paramètres de l'application. Il interagit avec les classes métier pertinentes et les classes d'accès aux données (DAO) pour effectuer des opérations liées aux paramètres de l'application, puis transfère la requête à la page JSP appropriée.

3.4. Vue :

Vues pouvant être générées pour l'application de gestion des stocks :

1. Vues des articles :

- Liste des articles : Affichage de tous les articles avec leurs détails tels que l'ID, le nom et la quantité.
- Ajouter un article : Formulaire pour ajouter un nouvel article.
- Modifier un article : Formulaire pour mettre à jour un article existant.
- Supprimer un article : Confirmation de suppression d'un article.
- Voir un article : Affichage des détails d'un article spécifique.

2. Vues des commandes d'achat :

- Liste des commandes d'achat : Affichage de toutes les commandes d'achat avec leurs détails.
- Ajouter une commande d'achat : Formulaire pour créer une nouvelle commande d'achat.
- Voir une commande d'achat : Affichage des détails d'une commande d'achat spécifique.
- Ajouter un article à une commande d'achat : Formulaire pour ajouter un article à une commande d'achat.
- Supprimer un article d'une commande d'achat : Confirmation de suppression d'un article d'une commande d'achat.

3. Vues des commandes de vente :

- Liste des commandes de vente : Affichage de toutes les commandes de vente avec leurs détails.
- Ajouter une commande de vente : Formulaire pour créer une nouvelle commande de vente.
- Voir une commande de vente : Affichage des détails d'une commande de vente spécifique.
- Ajouter un article à une commande de vente : Formulaire pour ajouter un article à une commande de vente.
- Supprimer un article d'une commande de vente : Confirmation de suppression d'un article d'une commande de vente.
- Générer un bon de livraison : Formulaire pour générer un bon de livraison pour une commande de vente.

4. Vues de gestion des stocks :

- Vérifier les niveaux de stock : Affichage des niveaux de stock actuels et des actions nécessaires à entreprendre.
- Historique des mouvements de stock : Affichage de l'historique des mouvements d'un article spécifique.
- Bloquer un article : Confirmation de blocage d'un article pour de futures transactions.

5. Vues des bons de livraison :

- Liste des bons de livraison : Affichage de tous les bons de livraison avec leurs détails.
- Ajouter un bon de livraison : Formulaire pour créer un nouveau bon de livraison.
- Voir un bon de livraison : Affichage des détails d'un bon de livraison spécifique.
- Modifier un bon de livraison : Formulaire pour mettre à jour un bon de livraison existant.
- Supprimer un bon de livraison : Confirmation de suppression d'un bon de livraison.

6. Vues des données de commande :
- Liste des données de commande : Affichage de toutes les données de commande avec leurs détails.
- Ajouter des données de commande : Formulaire pour ajouter de nouvelles données de commande.
- Voir des données de commande : Affichage des détails de données de commande spécifiques.
- Modifier des données de commande : Formulaire pour mettre à jour des données de commande existantes.
- Supprimer des données de commande : Confirmation de suppression de données de commande.

7. Vues des données d'article :
- Liste des données d'article : Affichage de toutes les données d'article avec leurs détails.
- Ajouter des données d'article : Formulaire pour ajouter de nouvelles données d'article.
- Voir des données d'article : Affichage des détails de données d'article spécifiques.
- Modifier des données d'article : Formulaire pour mettre à jour des données d'article existantes.
- Supprimer des données d'article : Confirmation de suppression de données d'article.

8. Vues des types de mouvement de stock :
- Liste des types de mouvement de stock : Affichage de tous les types de mouvement de stock.
- Voir un type de mouvement de stock : Affichage des détails d'un type de mouvement de stock spécifique.

4. Implimentation:

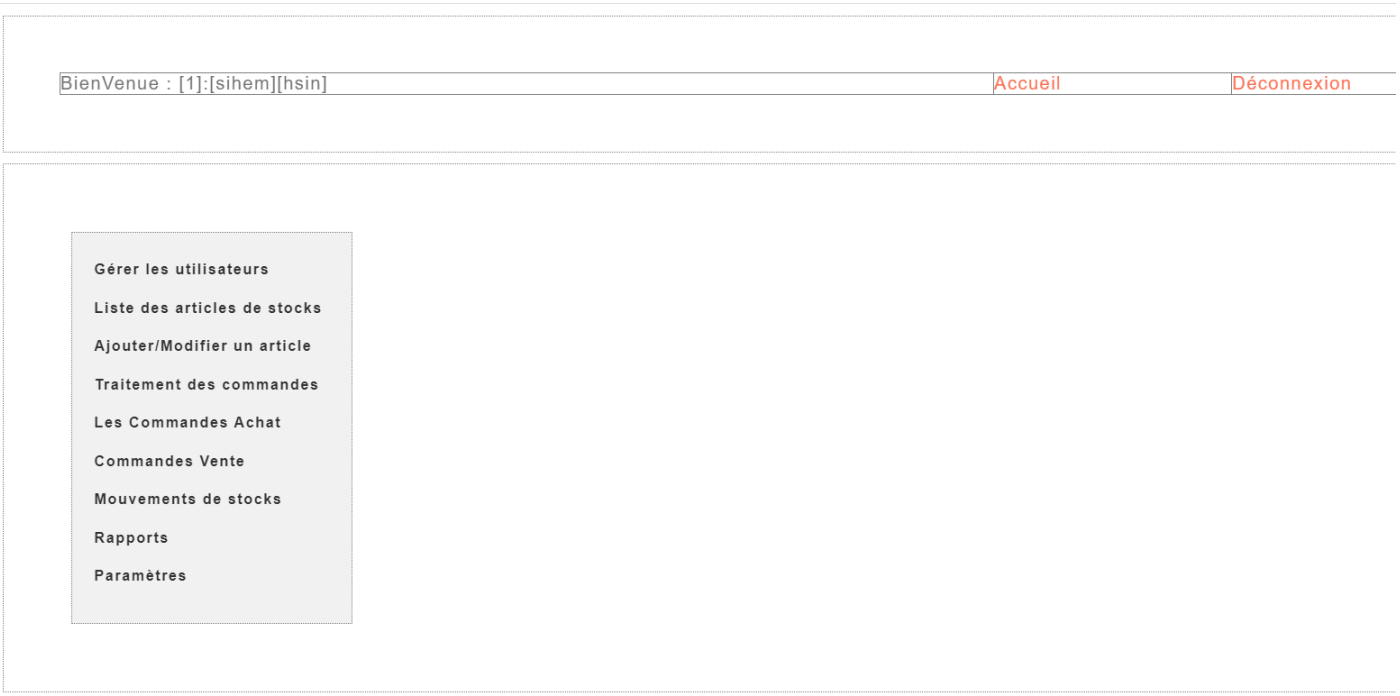

Figure 3-user interface

1. Dashboard :

- Le tableau de bord fournit un aperçu des informations clés et des métriques liées à la gestion des stocks, telles que la quantité totale de stocks, les alertes de stocks bas, les mouvements de stocks récents et les commandes en attente.

2. Liste des stocks : Articles en stocks

- Cette vue affiche une liste de tous les articles en stock. Elle comprend des colonnes pour le nom de l'article, la quantité et d'autres détails pertinents. Les utilisateurs peuvent effectuer des recherches, trier et filtrer la liste pour trouver rapidement des articles spécifiques.

3. Ajouter/Modifier un article :

- Les utilisateurs peuvent accéder à cette vue pour ajouter un nouvel article au stock ou modifier les détails d'un article existant. Le formulaire comprend des champs pour le nom de l'article, la description, la quantité, les informations du fournisseur et d'autres attributs supplémentaires spécifiques à vos besoins commerciaux.

4, . Mouvements de stocks :

- Cette vue fournit un journal de tous les mouvements de stocks, y compris les entrées (réceptions) et les sorties (ventes, retours, etc.). Chaque entrée comprend des détails tels que le nom de l'article, la quantité, le type de mouvement, la date et d'éventuelles notes associées.

5. Commandes : Orders

- Dans cette vue, les utilisateurs peuvent gérer les commandes pour l'achat de nouveaux articles en stock. Elle affiche une liste des commandes en attente avec des informations pertinentes telles que le fournisseur, la date de commande, la date de livraison prévue et l'état de la commande. Les utilisateurs peuvent créer de nouvelles commandes, suivre l'avancement des commandes existantes et mettre à jour leur statut.

6. Rapports :

- La section des rapports permet aux utilisateurs de générer divers rapports liés à la gestion des stocks. Cela peut inclure des rapports sur les niveaux de stock, les mouvements de stocks, l'historique des commandes, la performance des fournisseurs, etc. Les utilisateurs peuvent sélectionner des paramètres spécifiques et des filtres pour personnaliser la sortie du rapport.

7. Paramètres :

- La vue des paramètres offre des options pour personnaliser les préférences et les configurations de l'application. Cela peut inclure la gestion des utilisateurs, les notifications, l'intégration avec des systèmes externes et d'autres paramètres spécifiques au système.

Conclusion

Travail inachevé mais nous faisons de notre mieux.