2JAVA

Documentation Technique

ISTORE

Ryan Dordain & Séraphin Dubail

Binary

Table des matières :

# Gestion des données

Pour des raisons de portabilité et de sécurité, nous avons choisis d’héberger notre base de données sur un serveur distant.

L’avantage est que grâce à cette solution, tous les appareils disposant du logiciel IStore auront accès aux données peu importe leur emplacement mais l’inconvénient est que ces appareils doivent obligatoirement être connectés à internet.

Les codes d’accès à la DB sont dans le fichier AAAAAAA.java

## Schéma de la base de données :

Notre base de données est divisée en 4 tables :

-items

-shop

-users

-white\_list

Voici un la composition détaillée des table de la DB :

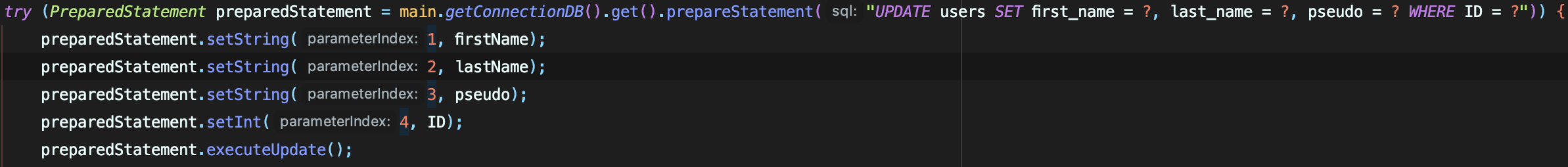
Une image contenant table

Description générée automatiquement

## Sécurité :

Pour sécuriser au maximum les données des utilisateurs, les mots de passes sont hachés dans le fichier *PassewordHasher.java,* pour y parvenir, nous avons utilisés la librairie native de java « Security ».

Nous avons également utilisé des requêtes préparées afin d’éviter les injections SQL évitant ainsi de forcer l’entrer ou la suppression de données dans la base de données.



Requête préparée

# Affichage Graphique

Pour réaliser la partie graphique de notre logiciel, nous avons utilisés la librairie native de java «Swing ». Son intégration native à notre IDE intelliJ permet d’organiser l’affichage graphique sans grande difficulté notamment grâce à la visualisation en temps réel de notre affichage.

L’instanciation des fenêtres de fait dans le fichier *Display.java*

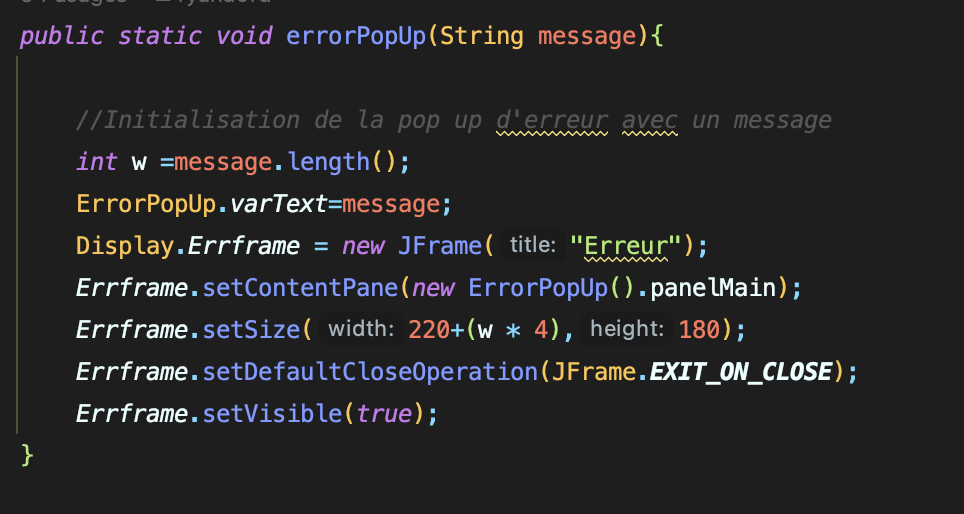
Une image contenant texte

Description générée automatiquement

instanciation de la fenêtre

Nous avons instancié une JFrame par défaut à laquelle nous venons modifier le contenu à afficher ainsi que la taille de la fenêtre en fonction des besoins.

Nous avons procédé de la même manière pour générer une fenêtre popup que l’utilisateur va fermer à chaque fois : si besoin, on appelle la fonction *errorPopUp* avec en paramètre le message qu’on souhaite afficher. Elle définit au passage sa taille en fonction du message à afficher.



Fonction errorPopUp