Kot-Line

Table des matières

Source

Réaliser par l'équipe RedSuns (POIRIER Alexandre & BELLIOT Killian) le 21/11/2022.

- Lien du projet: https://ldv-melun.github.io/projet-kot-line/
- Lien du repository : https://gitlab.com/alexandrepoirier69/kot-line

1. Contexte

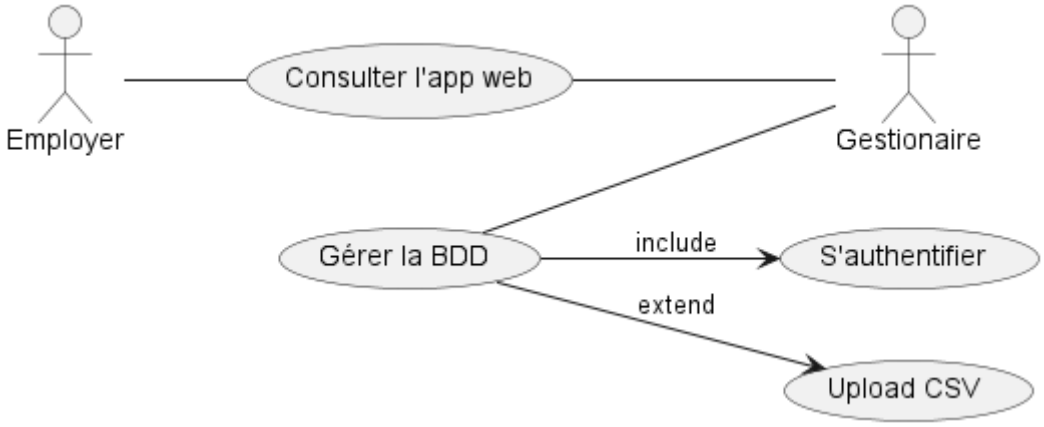
La société de service SO-DEV dans laquelle vous travaillez à comme client l'entreprise Kot-Line, créatrice de ligne de vêtements.

La société Kot-Line reçoit de la part de ses partenaires, tous les mois, 2 fichiers clients au format CSV.

Afin d'exploiter ces fichiers partenaires, Kot-Line souhaite disposer d'un outil (une application web), disponible qu'aux gestionnaires, lui permettant d'intégrer les données de ces fichiers dans une base de données relationnelle.

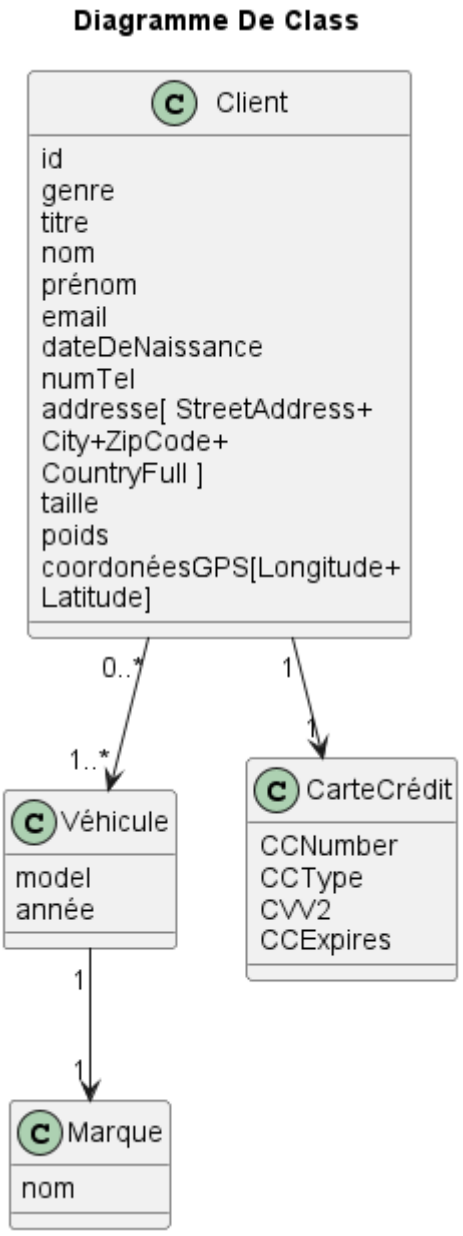
L'entreprise Kot-Line envisage de se lancer dans des produits en liens avec l'automobile.

Le département R&D vous demande de concevoir un premier prototype d'application.



2. Analyse

2.1 Diagramme de classe



Noter toutes les questions qu'on se pose !

2.2 Choix librairie utilisée pour lire un fichier CSV

Nous avons choisi d'utilisée la bibliothèque FasterXML Jackson CSV Library. En effet malgrès le fait que cette bibliothèque nécessite une certaine configuration et que l'instance de mappeur est lourde et doit être mis en cache ce qui fait qu'elle peut être lourd par rapport aux autres solutions, il faut définir le mappeur et le schéma de notre fichier ce qu'il peut être plus légèrement plus exigeant. FasterXML Jackson CSV Library possède l'avantage d'être stable, il s'intègre bien avec le typage JVM, faut baliser la classe grâce à JsonProperty car les noms peuve ne pas correspondre aux noms de champ de la classe de données. On peut utiliser un constructeur sans argument manuellement ou avec un plugin. Enfin l'écriture est assez simple ce qui rend que la lecture et l'analyse facile à gérer. D'autant plus que comme tous les mois l'entreprise Kot-Line aurait de nouveaux fichiers CSV ils nous faut être polyvalent

2.3 Réflexion d'mmplémentation des contraintes de sélection

2.3.1 La contrainte d'âge des individus

On crée une méthode qui convertit le fichier CSV en fichier JSON et qui récupère la date de naissance d'une personne et on l'a soustrait avec la date d'aujourd'hui ce qui nous donnes son âge ce qui nous permets de faire la sélection en ne choissisant que des peronnes majeurs (18 ans) et des personnes qui n'ont pas atteint l'âge de 88 ans.

2.3.2 La contrainte de taille des individus :

Pour pallier les incohérences de valeurs entre la taille en inch et celle en cm on décide de choisir comme mesure principale de prendre celle en centimètre car le système international d'unités associer à la taille est le metre donc dans notre cas les centimètres. Pour ce faire on crée une méthode qui convertit le fichier CSV en fichier JSON et qui calcule la valeur en inch à partir de la valeur en cm et on vérifie si elle est différente à celle renseigner et si elle est différente on rentre l'individus dans un fichier anomalie avec la cause de son entré dans ce fichier.

2.3.3 La contrainte de poids des individus :

Lorsque nous avons analysé les fichiers CSV nous avons remarqué que nous avons les mêmes incohérences que lors de la taille des individus. C'est pour cela que pour pallier les incohérences de valeurs entre le poids en pounds et en kilogamme on décide de choisir comme mesure principale de prendre celle en kilogramme car le système international d'unités associer à le poids est le kilogramme donc dans notre cas les kilogrammes. Pour ce faire on crée une méthode qui convertit le fichier CSV en fichier JSON et qui calcule la valeur en pounds à partir de la valeur en kilogramme et on vérifie si elle est différente à celle renseigner et si elle est différente on rentre de l'invidus dans un fichier "anomalie" avec la cause de son entré dans ce fichier.

2.3.4 La contrainte des cartes de crédits :

Pour éviter de se retrouver avec des doublons (une carte de crédit pour plusieurs utilisateurs). On convertit le fichier CSV en fichier JSON puis en le parcourant on test si une carte de crédit appartient à plus d'une personne et si celle-ci appartient à plus d'une personne on mets dans un fichier "anomalie" avec la cause de son entré dans ce fichier. Le nombre de personne qui auront mis la même carte de crédit.

2.4 Stratégie gestion des données à problèmes :

Pour la gestion des données à problèmes on a décidé de créer un fichier "anomalie" en CSV pour faciliter le traitement des données dans lequel on retrouve l'id de l'individus son nom, prénom et la raison de son problème(âge, taille, poids, carte de crédit) donc ils ne seront pas ajouter à la base de donnée.

2.5 Gestion des données utilisateurs en double

Afin d'éviter d'inscrire plusieurs fois les mêmes données dans la base de données. Lorsque nous avons notre fichier CSV on le convertit en fichier JSON qu'on parcoure et grâce à une méthode on vérifie si les données d'un individus qu'on essaie de rajouter ne corresponde pas déjà à un individus existant et s'il existe déjà on le rajoute dans le fichier anomalie et on ne touche pas à la base de donnée

2.6 Identification des Evil User Story et Abuser Story

En tant qu'attaquant, je veux essayer d'accéder aux comptes d'un gestionnaire, car le gestionnaire à plus de droits qu'un employé. Ce qui me permettrait directement de modifier la base de données.