Lapié Joséphine
Li Yicong
Hagron Raphaël
Acker Manuel

NF18 P24

NF18: Société Vehicle Share

UML v2 et MLD

Groupe 4

Responsable: Alessandro Victorino

UML v2

Lien PlantUML:

https://www.plantuml.com/plantuml/uml/dLNFRji-3BxdAGIVVhzW2TeTWsBeK7Id0UjfLqEncOIe_dWILJJe-YwxxZdwOgEjA5RafS2IImoAvCVI8qNTUfAEWbQpHadlnPr2C2HUPe9_2z7jBDcLyEJGR9FDGsXB8u42TZQm4cZeO0GjKHqTNmVh0UU1KR49ZEOIcB5wbUVi7B19d0B9BOW1WjCTKdsrZII93iQgmMdqHOLFKc6Bn0MsaY1LwAaU3k2__yUC3yvs3Y4bFThl3JdW4pz2SOK5ZcmTUFOeuL6fYR6nBTITzIHLP0LRS2FUdJLjQ0Y2UmSlobGMiOsDoDfuGSToDSMxSeurVuTTtzTH8THQakjjaj2QKW79dAr3AEnQkXzXOhGjA9Xeo6Z1JVAwTN3IK2cCCG0ryu6wGPUAOd9xaExWyuZAQY1NY6Zm2MIeiymRPMKwRS4t3hj8aU2PiWbhqNVMu5gTJDctmVaeao6aa8r8Devox_ys029p0OUs7KLlOHtePD8tQCfMyNWnIQhvkoVKG0uoMikvsNUZQP9BsIOgcbBPWMMzwLFjq9VPF0HKTQdulx2u9uPbzMk549uPTooJjGGJzEfbIFI1Wr_VkH1dYZZJxsO7pQD2JrCjliY6WikQ8TCrkGX9spsOkfC86WnDheIXQMrlyMJNXxDyvkwivYIdsz9OEYd3uzP81NoZrjQgwNXF2NysnkRZm_pn2GbYXcB02mB6whM3Ee8U6B9hLbkfkpHRSFKqpaUYiyCJTlDpiSWszYnRtdC-gQwNotcLhzYLa5qd7SNeF4t-Misox_vmV7xAaqN-ZfpWLrVBvNKbiYhAsyCYsU1xSY3UVebFipmLHrNYyjWSE6rh3W9FeXiyttuFhXnvFAxcCOh_qgg94SDgE-V9UulTEWWauh85bxl6DbIJFPHoCmsDdZbaGR7m274rVCKrskDOxw7joqujkKZfYKUgCzrh3ccjONU54Cx4324Vgt8vCQ_D1ZWBd0sA1UVtT0eNxrlgvJplOXBXPdOBFDbQ_G40

UML:

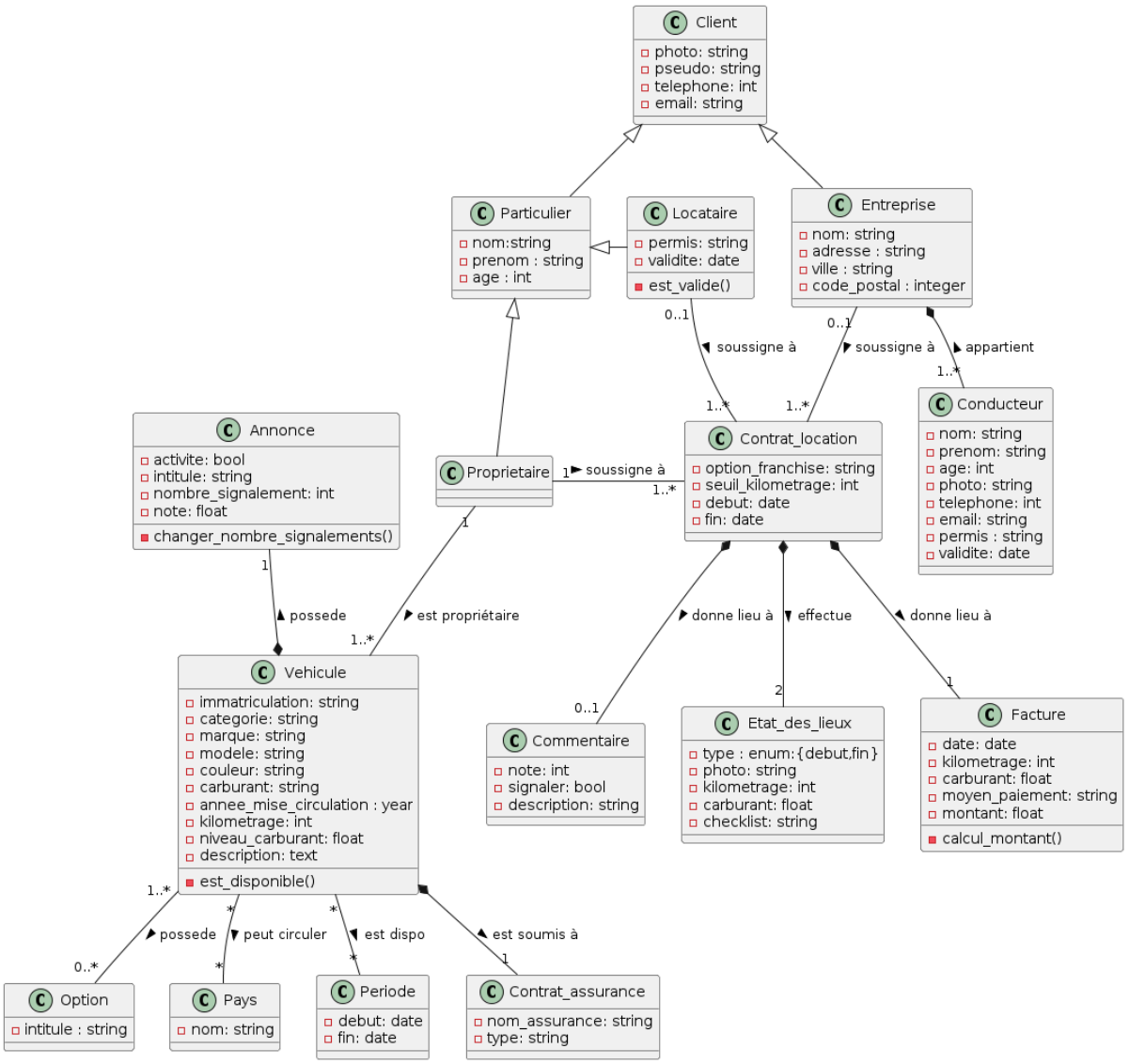
```
@startuml
class Client {
   - photo: string
   - pseudo: string
   - telephone: int
   - email: string
}

class Particulier {
   -nom:string
   -prenom : string
   -age : int
}

class Locataire {
   - permis: string
   - validite: date
   - est_valide()
}

class Proprietaire {
}
```

```
class Entreprise {
    - nom: string
    - adresse : string
    - ville : string
    - code_postal : integer
}

class Conducteur {
    - nom: string
    - prenom: string
    - age: int
    - photo: string
    - telephone: int
    - email: string
    - permis : string
    - validite: date
}

class Vehicule {
    - immatriculation: string
    - categorie: string
    - marque: string
    - modele: string
    - couleur: string
    - carburant: string
    - annee_mise_circulation : year
    - kilometrage: int
    - niveau_carburant: float
    - description: text
    - est_disponible()
}

class Option{
    - intitule : string
}

class Pays {
    - nom: string
}

class Periode {
    - debut: date
    - fin: date
}

class Contrat_assurance {
```

```
    - nom_assurance: string
    - type: string
}

class Contrat_location {
    - option_franchise: string
    - seuil_kilometrage: int
    - debut: date
    - fin: date
}

class Etat_des_lieux {
    - type : enum:{debut,fin}
    - photo: string
    - kilometrage: int
    - carburant: float
    - checklist: string
}

class Facture {
    - date: date
    - kilometrage: int
    - carburant: float
    - moyen_paiement: string
    - montant: float
    - calcul_montant()
}

class Commentaire {
    - note: int
    - signaler: bool
    - description: string
}

class Annonce {
    - activite: bool
    - intitule: string
    - nombre_signalement: int
    - note: float
    - changer_nombre_signalements()
}


Client <|-- Entreprise
Entreprise *-- "1..*" Conducteur : appartient <
Client <|-- Particulier
Particulier <|- Locataire
```

Particulier <|-- Proprietaire
Entreprise "0..1" -- "1..*" Contrat_location : soussigne à >
Proprietaire "1" - "1..*" Contrat_location : soussigne à >
Proprietaire "1" -- "1..*" Vehicule : est propriétaire >
Vehicule "*" -- "*" Periode : est dispo >
Vehicule "*" -- "*" Pays : peut circuler >
Vehicule *-- "1" Contrat_assurance : est soumis à >
Vehicule "1..*" -- "0..*" Option : possede >
Locataire "0..1" -- "1..*" Contrat_location : soussigne à >
Contrat_location *-- "1" Facture : donne lieu à >
Contrat_location *-- "2" Etat_des_lieux : effectue >
Contrat_location *-- "0..1" Commentaire : donne lieu à >
Annonce "1" --* Vehicule : possede <
@enduml

Visualisation:



MLD

Droits des utilisateurs

Les **administrateurs** système ont tous les droits.
Les locataires pourront consulter les tables servant à réserver un véhicule telles que : les vehicules qu'il a loués ou en cours de location (avec les caractéristiques associées : les options, les pays de circulation, la période de disponibilité et le contrat d'assurance), les annonces qui sont activées, les commentaires, les propriétaires, les états des lieux qui les concernent, les factures qui les concernent, et les contrats de location qui les concernent.

Les **entreprises** peuvent voir : toutes les annonces activées, tous les commentaires, les factures, contrats de location et états des lieux qui les concernent, les propriétaires, la liste de leurs conducteurs, les véhicules qu'elle a loués ou en cours de location (avec les caractéristiques associées : les options, les pays de circulation, la période de disponibilité et le contrat d'assurance)

Les **propriétaires** peuvent voir les mêmes tables ainsi que les locataires qui ont réservé une de leurs voitures, les voitures qu'ils détiennent et ses annonces (même désactivées)

Explications des transformations par héritage :

Nous avons décidé de faire des héritages par classes filles. En effet, la classe client n'est reliée à aucune autre classe et l'héritage est exclusif, ce qui nous permet ici de choisir l'héritage par classe fille. Il en va de même pour la classe Particulier, qui n'est reliée à aucune autre classe, ce qui nous permet de choisir également l'héritage par classes filles.

Généralités :

On considère que tous les attributs sont NOT NULL sauf si indication contraire.

Dans certaines classes, nous avons des cardinalités 1..*, qui correspondent à une minimalité de 1. Pour exprimer cette contrainte, nous avons choisi d'utiliser l'égalité des projections.

Classes générales:

Propriétaire(#pseudo:string, photo: string, telephone: int, email: string, nom: string, prenom: string, age: int)
Contraintes
- age > 18

Locataire(#pseudo:string, photo: string, telephone: int, email: string, permis: string, validité: date, nom: string, prenom: string, age: int)
Contraintes:
- age > 18
- validité > NOW()

Entreprise(#id_entreprise:int, nom: string, adresse: string, ville: string, code_postal: int)
Contraintes:
- code_postal > 0 AND code_postal < 99999

Conducteur(#id_conducteur: int, #entreprise => Entreprise.id_entreprise, nom: string, prenom: string, age: int, photo: string, pseudo: string, telephone: int, email: string, permis : string, validité: date)
Contraintes :

- Projection(Entreprise, id_entreprise) = Projection(Conducteur, entreprise)
- age > 18
- validité > NOW()

Contrat_location(#id_contrat:int, option_franchise: string, seuil_kilometrage: int, debut: date, fin: date, proprietaire => Propriétaire.pseudo, locataire => Locataire.pseudo, entreprise => Entreprise.id_entreprise)

Contraintes :

- (locataire AND NOT entreprise) OR (NOT locataire AND entreprise)
- Projection(Propriétaire, pseudo) = Projection(Contrat_location, propriétaire)
- Projection(Locataire, pseudo) = Projection(Contrat_location, locataire)
- Projection(Entreprise, id_entreprise) = Projection(Contrat_location, entreprise)
- Projection(Contrat_location, edl_debut) = Projection(Restriction(Etat_des_lieux, type = “début”), id_edl) et
- Projection(Contrat_location, edl_fin) = Projection(Restriction(Etat_des_lieux, type = “fin”), id_edl)
- commentaire peut être NULL
- seuil_kilometrage > 0

Etat_des_lieux (#id_edl:int, #contrat=>Contrat_location.id_contrat, type(enum:{debut,fin}), photo: string, kilometrage: int, carburant: float, checklist: string, contrat_location => Contrat_location.id_contrat)
Contraintes:

- Projection(Contrat_location, id_contrat) = Projection(Etat_des_lieux, contrat)
- kilometrage > 0
- carburant >0 AND carburant < 1

Facture (#id_facture:int, date: date, kilometrage: int, carburant: float, moyen_paiement: string, montant: float, contrat_location => Contrat_location.id_contrat)
Contraintes:

- kilometrage > 0
- carburant >0 AND carburant < 1
- montant > 0

Commentaire (#id_commentaire:int, note: int, signaler: bool, description: string, contrat_location => Contrat_location.id_contrat)
Contraintes:

- note > 0 AND note < 5

Véhicule(#immatriculation: string, categorie: string, marque: string, modele: string, couleur: string, carburant: string, annee_mise_circulation : year, kilometrage: int, niveau_carburant: float, description: text, propriétaire => Propriétaire.pseudo,
Contraintes :

- Projection(Véhicule, propriétaire) = Projection(Propriétaire, pseudo)
- kilometrage > 0
- niveau_carburant > 0 AND niveau_carburant < 1 (On considère que lorsque le niveau vaut 1, le réservoir est plein)

Pays(#nom: string)

Contrat_assurance (#id_assurance:int, nom_assurance: string, type: string, vehicule=> Véhicule.immatriculation)

Annonce(#id_annonce: int, activite: bool, intitule: string, nombre_signalement: int, note: float, vehicule=> Véhicule.immatriculation)
Contraintes :

- nombre_signalement > 0
- (note >= 0) AND (note <= 5)
- (nombre_signalement >= 3 AND activité = 0)

Option(#id_option:int, intitule: string)

Periode(#id_periode: int, debut: date, fin: date)
Contraintes :

- fin > debut

Classes associations:

Est_disponible(#vehicule=>Vehicule.id_vehicule, #periode=> Periode.id_periode)

Peut_circuler(#vehicule=>Vehicule.id_vehicule, #pays=> Pays.nom)

Possede(#vehicule=>Vehicule.id_vehicule, #option => Option.id_option) avec Projection(Option,id_option)=Projection(Possède,option)