**Modèle Conceptuel de la Base de Donnée**

Dans ce modèle, chaque entité a un identifiant unique (ID) pour la référencer de manière unique. Les relations entre les entités sont établies à l'aide de ces identifiants, formant ainsi des relations un-à-un ou un-à-plusieurs selon les besoins du système.

1. Utilisateur (**User**) :

- Identifiant (ID) : id

- Nom : nom

- Prénom : prenom

- Adresse e-mail : email

- Mot de passe : password

- Numéro de téléphone : phone

- Sexe : genre

2. Conducteur (**Driver**) :

- Identifiant (ID, lié à l'utilisateur) : id

- Véhicule (ID du véhicule, lié à l'entité Véhicule) : vehicleId

- Statut (disponible, en trajet, etc.) : status

3. Passager (**Passenger**) :

- Identifiant (ID, lié à l'utilisateur) : id

- Statut (en attente, en trajet, etc.) : status

4. Véhicule (**Vehicle**) :

- Identifiant (ID)

- Propriétaire (ID du propriétaire ) : driverId

- Marque : mark

- Numéro de la plaque : plateNumber

- Numéro du moteur : enginNumber

- Photo du véhicule : pictures

- Nombre de places : placeNumber

5. Trajet (**Trip**) :

- Identifiant (ID) : id

- Conducteur (ID du conducteur, lié à l'entité Conducteur) : driverId

- Date et heure de départ : startTime

- Lieu de départ : startingPlace

- Lieu d'arrivée : destinationPlace

- Nombre de places disponibles : freePlace

- Statut (en cours, terminé, annulé, etc.) : status

6. Réservation (**Reservation**) :

- Identifiant (ID) : id

- Passager (ID du passager, lié à l'entité Passager) : passengerId

- Trajet (ID du trajet, lié à l'entité Trajet) : tripId

- Date et heure de réservation : reservationDate

- Statut (confirmé, en attente, annulé, etc.) : status

7. Avis et Commentaires (**RatingAndComment**) :

- Identifiant (ID) : id

- Auteur (ID de l'utilisateur, lié à l'entité Utilisateur) : passengerId

- Cible (ID du conducteur ou du passager concerné) : driverId

- Note : rating

- Commentaire : comment

- Date de publication : createdAt

Pour ce projet d’application de covoiturage, une base de données NoSQL comme Firestiore de Firebase est un option appropriée pour plusieurs raisons :

* **Facilité d'utilisation** : Firestore est facile à configurer et à utiliser, ce qui est idéal pour les développeurs qui cherchent à démarrer rapidement avec leur application.
* **Temps réel** : Firestore offre la synchronisation en temps réel des données entre les clients et la base de données. Cela signifie que les utilisateurs peuvent voir les mises à jour en temps réel sans avoir besoin de rafraîchir l'application.
* **Facilité d'intégration** : Firestore s'intègre facilement avec d'autres produits Firebase tels que l'authentification, le stockage cloud, les fonctions cloud, etc., ce qui simplifie le développement et la gestion de l'ensemble de votre application.
* **Sécurité** : Firestore offre des règles de sécurité flexibles qui vous permettent de contrôler précisément qui peut accéder à vos données et ce qu'ils peuvent faire avec ces données.