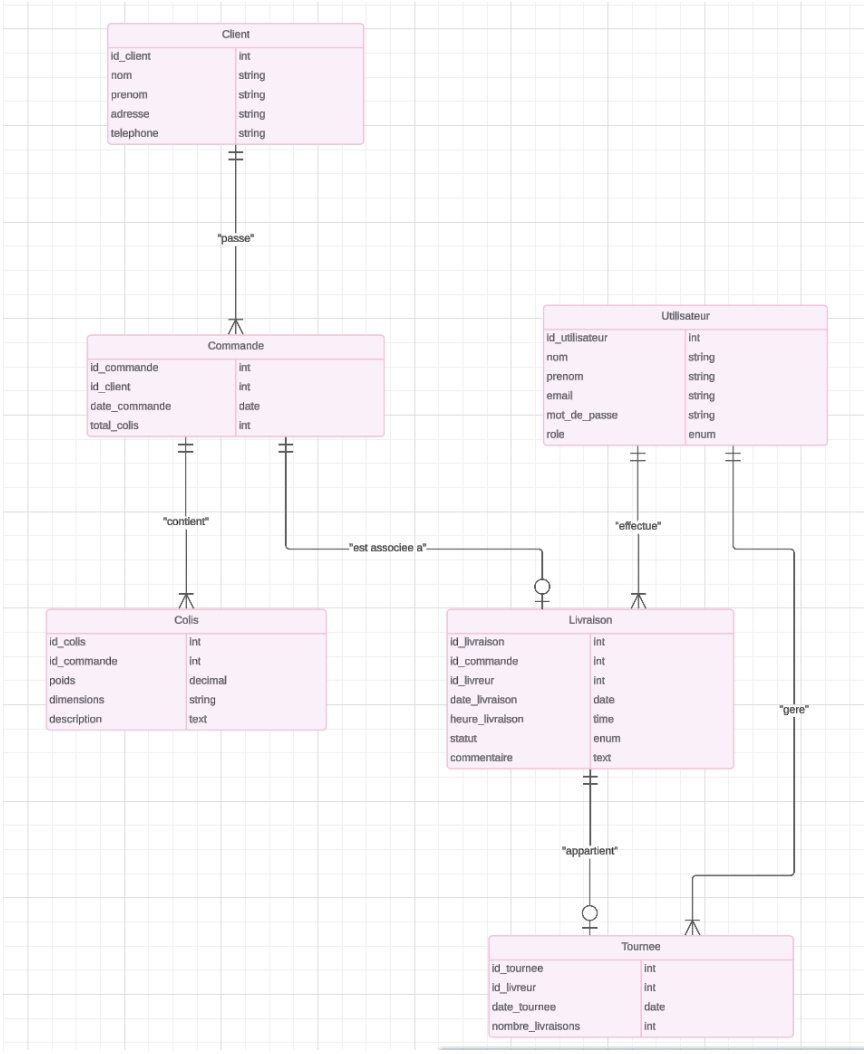
Livrable Deliver'easy

Dictionnaire de données.

Nom de la Table	Nom de l'Attribut	Type	Description
Utilisateur			
	id_utilisateur	INT	Identifiant unique de l'utilisateur
	nom	VARCHAR(50)	Nom de l'utilisateur
	prenom	VARCHAR(50)	Prénom de l'utilisateur
	email	VARCHAR(100)	Adresse e-mail de l'utilisateur, doit être unique
	mot_de_passe	VARCHAR(255)	Mot de passe hashé de l'utilisateur
	role	ENUM	Rôle de l'utilisateur (administrateur, livreur)
Client			
	id_client	INT	Identifiant unique du client
	nom	VARCHAR(50)	Nom du client
	prenom	VARCHAR(50)	Prénom du client
	adresse	VARCHAR(255)	Adresse complète du client
	telephone	VARCHAR(20)	Numéro de téléphone du client
Livraison			
	id_livraison	INT	Identifiant unique de la livraison
	id_commande	INT	Identifiant de la commande associée à la livraison
	id_livreur	INT	Identifiant du livreur assigné pour la livraison
	date_livraison	DATE	Date prévue de la livraison
	heure_livraison	TIME	Heure prévue de la livraison
	statut	ENUM	Statut de la livraison (en cours, livrée, à replanifier)
	commentaire	TEXT	Commentaire laissé par le livreur
Commande			
	id_commande	INT	Identifiant unique de la commande
	id_client	INT	Identifiant du client qui a passé la commande
	date_commande	DATE	Date de la commande
	total_colis	INT	Nombre total de colis dans la commande
Colis			
	id_colis	INT	Identifiant unique du colis
	id_commande	INT	Identifiant de la commande à laquelle appartient le colis
	poids	DECIMAL(5,2)	Poids du colis en kilogrammes
	dimensions	VARCHAR(50)	Dimensions du colis (ex: 10x10x10 cm)
	description	TEXT	Description du contenu du colis
Tournee			
	id_tournee	INT	Identifiant unique de la tournée de livraison
	id_livreur	INT	Identifiant du livreur assigné pour la tournée

date_tournee	DATE	Date de la tournée
nombre_livraisons	INT	Nombre de livraisons planifiées dans la tournée

MCD/MLD




```
CREATE TABLE `Utilisateur` (
  `id_utilisateur` INT PRIMARY KEY,
  `nom` VARCHAR(50),
  `prenom` VARCHAR(50),
  `email` VARCHAR(100) UNIQUE,
  `mot_de_passe` VARCHAR(255),
  `role` ENUM('administrateur', 'livreur')
);

CREATE TABLE `Client` (
  `id_client` INT PRIMARY KEY,
  `nom` VARCHAR(50),
  `prenom` VARCHAR(50),
```

```
  `adresse` VARCHAR(255),
  `telephone` VARCHAR(20)
);

CREATE TABLE `Livraison` (
  `id_livraison` INT PRIMARY KEY,
  `id_commande` INT,
  `id_livreur` INT,
  `date_livraison` DATE,
  `heure_livraison` TIME,
  `statut` ENUM('en cours', 'livrée', 'à replanifier'),
  `commentaire` TEXT,
  FOREIGN KEY (`id_commande`) REFERENCES `Commande`(`id_commande`),
  FOREIGN KEY (`id_livreur`) REFERENCES `Utilisateur`(`id_utilisateur`)
);

CREATE TABLE `Commande` (
  `id_commande` INT PRIMARY KEY,
  `id_client` INT,
  `date_commande` DATE,
  `total_colis` INT,
  FOREIGN KEY (`id_client`) REFERENCES `Client`(`id_client`)
);

CREATE TABLE `Colis` (
  `id_colis` INT PRIMARY KEY,
  `id_commande` INT,
  `poids` DECIMAL(5,2),
  `dimensions` VARCHAR(50),
  `description` TEXT,
  FOREIGN KEY (`id_commande`) REFERENCES `Commande`(`id_commande`)
);

CREATE TABLE `Tournee` (
  `id_tournee` INT PRIMARY KEY,
  `id_livreur` INT,
  `date_tournee` DATE,
  `nombre_livraisons` INT,
  FOREIGN KEY (`id_livreur`) REFERENCES `Utilisateur`(`id_utilisateur`)
);
```

Dépendances fonctionnelles par table

1. **Utilisateur**
    - id_utilisateur → nom, prenom, email, mot_de_passe, role

- Chaque utilisateur est identifié par un id_utilisateur unique qui détermine toutes les informations associées à cet utilisateur : nom, prénom, email, mot de passe, et rôle.

- email → id_utilisateur
  - L'email doit être unique et peut être utilisé pour retrouver l'id_utilisateur.

2. **Client**

- id_client → nom, prenom, adresse, telephone
  - Chaque client est identifié par un id_client unique qui permet de récupérer les informations associées telles que le nom, prénom, adresse, et téléphone.

3. **Commande**

- id_commande → id_client, date_commande, total_colis
  - Chaque commande est identifiée par un id_commande unique, qui permet d'obtenir les informations de la commande, y compris l'identifiant du client associé (id_client), la date de la commande et le nombre total de colis.

4. **Colis**

- id_colis → id_commande, poids, dimensions, description
  - Chaque colis est identifié par un id_colis unique qui permet de retrouver la commande associée ainsi que les caractéristiques du colis, telles que le poids, les dimensions et la description.

- id_commande → total_colis
  - Une commande contient un nombre total de colis, qui est déterminé par l'identifiant de la commande (id_commande).

5. **Livraison**

- id_livraison → id_commande, id_livreur, date_livraison, heure_livraison, statut, commentaire
  - Chaque livraison est identifiée par un id_livraison unique. Ce dernier détermine les informations de la livraison, telles que l'identifiant de la commande associée, l'identifiant du livreur, la date et l'heure de livraison, le statut de la livraison, et les éventuels commentaires.

- id_commande → id_client
  - L'identifiant de la commande permet d'obtenir le client qui a passé cette commande.

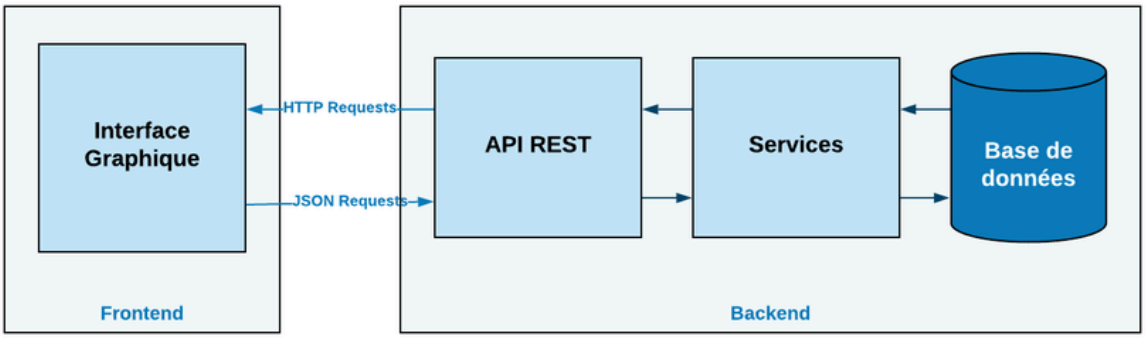
6. **Tournée**

- id_tournee → id_livreur, date_tournee, nombre_livraisons

- Chaque tournée de livraison est identifiée par un id_tournee unique, permettant de récupérer l'identifiant du livreur assigné, la date de la tournée, et le nombre total de livraisons prévues pour cette tournée.

- id_livreur, date_tournee → id_tournee
  - Une combinaison de l'identifiant du livreur et de la date de la tournée est unique et permet de retrouver l'identifiant de la tournée (id_tournee).

Schéma d'Architecture Applicative :



1. **Frontend** : C'est la partie de l'application avec laquelle les utilisateurs interagissent. Elle sert d'interface utilisateur, permettant l'affichage des données et l'envoi de requêtes pour effectuer des actions (par exemple, afficher des informations, mettre à jour un statut de livraison).(vue.js,css,html)

2.**Backend** : Ce composant traite les requêtes du frontend, exécute la logique de l'application (comme les opérations CRUD : créer, lire, mettre à jour et supprimer), et gère la communication avec la base de données. Il agit comme un pont entre le frontend et la base de données.(express,typescript)

3.**Database** : Cette base de données stocke toutes les informations nécessaires pour l'application (comme les informations des utilisateurs, des commandes, des livraisons, etc.). Elle reçoit des demandes de lecture et d'écriture de données en provenance du backend.(mysql,docker)