**INF3710 –Fichiers et Bases de données**

**Hiver 2023**

**TP No. 4**

**Groupe 4**

**2143023 – Bakashov, Marsel**

**2147174– Chowdhury, Rasel**

**2154453 – Hamadache, Wassim**

**Soumis à : Charles de Lafontaine**

**14 avril 2023**

# Introduction résumant le projet et contributions

Le projet consiste à modéliser une base données d’une coopérative de co-voiturage et à développer une application web permettant d’interroger cette base de données. Le projet comporte plusieurs étapes telles que la création d'un modèle entité-association, la conversion de ce modèle en un modèle relationnel avec la création d'une base de données SQL, la saisie de données et la création de requêtes SQL pour répondre à des besoins spécifiques, la création d'un déclencheur pour créer une table AVENDRE lorsque la valeur d'odomètre d'un véhicule spécifique est atteinte, l'analyse des dépendances fonctionnelles et la forme normale de la base de données, et enfin la création d'une application web pour interroger la base de données. La base de données a été modélisé avec PostgreSQL. L'objectif est de permettre aux membres de la coopérative de réserver et d'utiliser les véhicules appartenant à la coopérative. Les types de véhicules comprennent des voitures hybrides, des berlines et des minicamionnettes. Les véhicules doivent avoir une plaque d'immatriculation et une assurance automobile valide, et leur kilométrage doit être enregistré. Les emplacements de stationnement doivent être gérés pour chaque véhicule, ainsi que les informations sur les membres, y compris leur adhésion à la coopérative et leur permis de conduire valide. L'application web doit permettre la recherche d’information d’un membre, l’affichage et l’ajout d’une réservation.

En ce qui concerne les contributions, Marsel s’est occupé de la modélisation en notation UML, de la conversion en modèle relationnel et de l’application Web. Wassim, quant à lui, s’est occupé de créer les requêtes demandées, d’implémenter le déclencheur, de lister les tables, les dépendances fonctionnelles et la forme normale de la base de données relationnelles obtenue. Finalement, l’entrée de données et le rapport a été complété ensemble.

# Modèle conceptuel UML

AJOUTER IMAGE

# Dépendances fonctionnelles et explication de la forme normale de la BD

WASSIM

# Présentation de l’application développé avec copies d’écran permettant de démontrer toutes les fonctionnalités.

L’application Web utilisant la stack PEAN (PostgreSQL, Express, Angular et Node.js) doit permettre d’insérer et d’interroger les données de votre base de données. L’application doit permettre de rechercher l’information d’un membre et d’afficher les réservations ainsi que de permettre l’ajout d’un membre.

## Recherche d’information d’un membre

En ce qui concerne la fonctionnalité de recherche d'informations de membre, tous les membres sont affichés sur la page "Membres" de l'application Web. Pour réaliser cette fonctionnalité, nous utilisons un service appelé "communication.service" via sa méthode "getMembres()", qui envoie une requête GET au serveur pour récupérer la liste de tous les membres.

Du côté du serveur, la requête est interceptée par le "database.controller", qui à son tour appelle la méthode "getAllMembres" du "databaseService". Cette méthode exécute une requête SQL "SELECT \* FROM Coovoiturage\_schema.Membre;" pour récupérer la liste de tous les membres stockés dans la base de données. Cette liste est ensuite renvoyée au client qui l'affiche sur la page "Membres".

Pour la fonctionnalité de recherche, nous avons choisi de l'implémenter côté client plutôt que côté serveur, car la liste des membres était déjà affichée et il suffisait de la filtrer selon les entrées de l'utilisateur. Ainsi, lorsque l'utilisateur saisit un nom ou une partie de nom dans la barre de recherche, le client trie la liste des membres en utilisant les informations fournies par l'utilisateur et n'affiche que les résultats correspondants.

Cette approche offre une meilleure expérience utilisateur car les résultats de la recherche s'affichent instantanément et il n'y a pas de temps d'attente lié à une requête supplémentaire au serveur. Cependant, il est important de noter que cette approche peut devenir moins efficace si la liste des membres est très longue, auquel cas il pourrait être préférable de réaliser la recherche côté serveur.

AJOUTER IMAGE

## L’affichage et l’ajout d’un réservation

# En ce qui concerne la fonctionnalité d'affichage et d'ajout de réservations, elle se déroule sur la page "Réservations" de l'application Web. Pour afficher toutes les réservations existantes, nous utilisons le service "communication.service" via la méthode "getReservations()", qui envoie une requête GET au serveur pour récupérer la liste de toutes les réservations. Pour ajouter une nouvelle réservation, nous utilisons la méthode "insertReservation(reservation: Reservation)" qui envoie une requête POST au serveur pour ajouter la nouvelle réservation à la base de données.

# Côté serveur, la requête GET est interceptée par le "database.controller", qui appelle la méthode "getAllReservations()" du "databaseService". Cette méthode exécute une requête SQL "SELECT \* FROM Coovoiturage\_schema.Reservation;" pour récupérer la liste de toutes les réservations stockées dans la base de données. Cette liste est ensuite renvoyée au client qui l'affiche sur la page "Réservations".

# La requête POST est interceptée par le "database.controller", qui à son tour appelle la méthode "createReservation" du "databaseService". Cette méthode exécute une requête SQL "INSERT INTO Coovoiturage\_schema.Reservation VALUES($1,$2,$3,$4,$5,$6);" pour ajouter la nouvelle réservation à la base de données avec les entrées de l'utilisateur reçues côté client. Toutefois, avant d'ajouter la réservation, nous effectuons quelques vérifications pour nous assurer que les données sont valides. Nous vérifions que toutes les entrées ont été reçues, que la date de fin n'est pas antérieure à la date de début et que la voiture n'est pas réservée pendant la période demandée par l'utilisateur.

# Pour permettre à l'utilisateur d'ajouter une réservation, nous avons créé une section "Ajout" dans la page de Réservations. Cette section permet à l'utilisateur de remplir les champs nécessaires pour la création d'une nouvelle réservation. Une fois que l'utilisateur a rempli les champs et envoyé la demande, les données sont envoyées au serveur via la méthode "insertReservation" du "communication.service" et sont traitées comme expliqué précédemment. L'ID de la réservation n'est pas considéré comme une entrée, car il est incrémenté automatiquement. Nous avons également inclus des valeurs par défaut dans les champs du formulaire pour faciliter les tests du correcteur. Enfin, la page d'affichage de toutes les réservations permet à l'utilisateur de visualiser toutes les réservations existantes dans la base de données.

# AJOUTER IMAGE

# Guide d’installation et de configuration qui permette d’installer et exécuter l’application