

GSB CARPOOLING

APPLICATION DE COVOITURAGE INTERNE

Section BTS SIO - Lycée Condorcet – 90000 BELFORT



Galaxy Swiss Bourdin
Carpooling



1 SOMMAIRE

2	Historique du document	5
3	Description du sujet du projet.....	6
3.1	Contexte du projet	6
3.1.1	Identification et activité de l'entreprise	6
3.1.2	Projet de l'entreprise	6
3.2	Domaine étudié.....	7
3.3	Fonctionnement du système	7
3.4	Périmètre du système	8
3.5	Extensions possibles.....	8
4	Conception du projet	9
4.1	Dictionnaire de données	9
4.2	Modèle Conceptuel de Données (MCD)	13
4.3	Explication des associations et des cardinalités	13
4.4	Modèle logique des données relationnel normalisé	15
4.5	Script de création de base de données.....	16
4.6	Modélisation UML.....	19
4.6.1	Diagramme de contexte.....	19
4.6.2	Diagramme de packages	19
4.6.3	Diagrammes de cas d'utilisation	20
4.6.3.1	Diagramme de cas d'utilisation : Gestion des trajets	20
4.6.3.2	Diagramme de cas d'utilisation : Gestion des données personnelles	22
4.6.3.3	Diagramme de cas d'utilisation : Gestion administrative	24
4.6.4	Diagrammes de séquences (niveau analyse)	26
4.6.5	Diagramme de classes.....	28
5	Finalisation du projet.....	29
5.1	IMH.....	29
5.1.1	Introduction	29
5.1.2	Ouverture du programme et connexion de l'utilisateur.....	29
5.1.3	Administration – Fenêtre d'accueil.....	29
5.1.4	Administration – Gestion des utilisateurs.....	30
5.1.5	Administration – Gestion des utilisateurs – Ajout/Modification d'un utilisateur	30
5.1.6	Administration – Gestion des véhicules.....	31
5.1.7	Administration – Gestion des véhicules – Ajout/Modification d'un véhicule	31
5.1.8	Administration – Gestion des véhicules.....	32
5.1.9	Administration – Gestion des trajets – Ajout/Modification d'un trajet	32
5.1.10	Fenêtre d'accueil.....	33

5.1.11	Fenêtre « Vos trajets »	33
5.1.12	Fenêtre « Rechercher un trajet »	34
5.1.13	Fenêtre « Proposer un trajet »	34
5.1.14	Fenêtre « Paramètre utilisateur »	34
5.2	Principales requêtes développées	35
5.3	Quelques exemples de code	36
5.3.1	Authentification à l'application	36
5.3.2	Procédure de hachage de mot de passe	38

2 HISTORIQUE DU DOCUMENT

Version	Date	Auteur	Description
1.0	11/09/2020	CHANOT Alexandre	Composition de la description du sujet du projet
1.1	20/09/2020	CHANOT Alexandre	Ajout de fonctionnalités dans la partie « fonctionnement du système » et « extensions possibles »
2.0	09/11/2020	CHANOT Alexandre	Conception du projet (MCD, MLD, Script de BDD et UML)
2.1	20/12/2020	CHANOT Alexandre	Mise à jour du SGDB utilisé. Mise à jour du script de la base de données, du MCD, du MLD.
3.0	15/01/2020	CHANOT Alexandre	Présentation de l'IHM. Présentation de diverses requêtes utilisées et exemples de code.

3 DESCRIPTION DU SUJET DU PROJET

3.1 CONTEXTE DU PROJET

3.1.1 Identification et activité de l'entreprise

La société Galaxy Swiss Bourdin, géant pharmaceutique, est issue de la fusion entre le géant américain Gakaxy et le conglomérat européen Swiss Bourdin.

Le siège social de l'entreprise se situe à Philadelphie, Pennsylvanie aux États-Unis. Paris a été choisi pour accueillir en son sein le siège administratif.

Une conséquence de cette fusion est la recherche d'une optimisation de l'activité du groupe ainsi constitué en réalisant des économies d'échelle dans la production et la distribution des médicaments, tout en prenant le meilleur des deux laboratoires sur les produits concurrents.

L'entreprise compte 480 visiteurs médicaux en France métropolitaine (Corse comprise), et de 60 dans les départements et territoires d'outre-mer.

3.1.2 Projet de l'entreprise

Après plusieurs réunions comprenant des hauts-cadres, le pôle comptable et un chef de projet, **la société souhaite mettre en place un système de covoiturage internet à l'entreprise** pour faciliter les déplacements de ses visiteurs médicaux. Les principaux arguments avancés sont :

- Financier : L'objectif principal est de limité le coût des frais de déplacement.
- Écologique : Permet de limiter l'empreinte carbone de la société dans un monde qui se veut plus vert, écologique.
- Social : Permet la rencontre entre collaborateurs, la sociabilisation, pour permettre une augmentation de la psychologique positive au sein de la société.
En effet, s'intéresser à la santé et au bien-être, à ce qui rend les collaborateurs résilients, heureux, optimistes, permet d'augmenter, en théorie et sur le long terme : la productivité.

3.2 DOMAINE ÉTUDIÉ

Afin de développer la solution, je me suis tourné vers plusieurs technologies :

La solution sera conçue avec les technologies :

- C#
- SQL Server (Base de données)

C# est un langage professionnel parfait pour réaliser la solution demandée. Il permet de créer facile des logiciels de gestion via son IDE Visual Studio 2019. C'est également le langage étudié lors de ma première année de formation.

Concernant le système de gestion de base de données, j'opte pour l'utilisation d'SQL Server. Pourquoi ? Car j'ai été amené à l'utiliser lors de différents TP et l'apprentissage de ce SGBD est un plus. Son implémentation dans l'IDE Visual Studio 2019 que j'utilise pour le développement me facilite également ce dernier.

3.3 FONCTIONNEMENT DU SYSTÈME

INSCRIPTION

Lors de l'installation du programme, un compte administrateur sera défini. Ce sera ce compte qui sera habilité à créer des comptes utilisateur. L'inscription étant privée car interne à l'entreprise. L'utilisateur recevra un mail avec ses identifiant et un lien pour changer son mot de passe.

CONNEXION

Un utilisateur lambda arrivera sur une page d'accueil utilisateur. L'administrateur arrivera sur son tableau de bord.

PARCOURS

Conducteur

Un utilisateur conducteur pourra :

- Proposer un trajet.
- Renseigner le point de départ, le point d'arrivée, la date du trajet, l'heure de départ et d'arrivée du trajet
- Renseigner des étapes intermédiaires.
- Renseigner divers éléments : le nombre de passagers possibles, la voiture utilisée (personnelle ou professionnelle).
- Notifier un commentaire informatif lors de la création d'un nouveau voyage qui sera visible par les utilisateurs en recherche d'un covoiturage.
- Voir la liste de ses trajets publiés et en cours.

Utilisateur

Un utilisateur passager pourra :

- Rechercher multicritères sur un trajet : la ville de départ, la ville d'arrivée, la date du trajet etc.
- S'inscrire / se désinscrire sur un trajet
- Mettre une note au conducteur une fois le trajet passé.

VEHICULE

Les voitures professionnelles pourront être ajoutées ou supprimées par et seulement par l'administrateur.

La liste des voitures sera alors accessible lors de la proposition d'un trajet.

ADMINISTRATION

L'administrateur pourra via un tableau de bord administrateur :

- Ajouter / modifier / supprimer des utilisateurs
- Supprimer / modérer des trajets ne respectant pas les règles d'utilisations.
- Voir quelques statistiques (ex : le nombre de trajets proposés, le nombre d'utilisateurs connectés etc.)
- Voir les informations des utilisateurs

3.4 PÉRIMÈTRE DU SYSTÈME

Les fonctionnalités du site seront celles présentées précédemment, elles pourront évoluer en fonction des demandes des utilisateurs du site.

3.5 EXTENSIONS POSSIBLES

1. Il sera possible d'établir des statistiques en fonction des trajets proposés, des notes utilisateurs.
2. Développer une MAP qui offrira différentes fonctionnalités comme indiquer le tracé du trajet.
3. Proposer un calcul automatique du temps de trajet d'un point A à un point B.
4. Créer un bouton « alerte » que l'utilisateur pourra utiliser pour être prévenu qu'un trajet sur ses critères a été ajouté.
5. **Utilisateur** : Envoyer une demande de covoiturage.
6. **Conducteur** : Voir / Accepter une demande de covoiturage.
7. Ajouter un système de messagerie.
 - Permettra aux utilisateurs de s'envoyer des messages.
 - Permettra à l'administrateur d'envoyer des messages à l'ensemble des utilisateurs.
8. Développer une fonctionnalité pour envoyer un mail de rappel quand l'heure du trajet approche.
9. Créer une liaison avec l'application de remboursement de frais de déplacement.
10. Développer une application Android (version simplifiée).

4 CONCEPTION DU PROJET

4.1 DICTIONNAIRE DE DONNÉES

TABLE	Nom conceptuel	Nom logique	Type (E, Ca, Co)	Nature	Longueur	ID ?	Exemple
UTILISATEUR							
	Identifiant de l'utilisateur	Utilisateur_Id	E	N	8	Oui	00001254
	Nom de l'utilisateur	Utilisateur_Nom	E	A	25		JEAN
	Prénom de l'utilisateur	Utilisateur_Prenom	E	A	20		Pierre
	Photo de profil de l'utilisateur	Utilisateur_Photo	E	Image	X		[UNE IMAGE]
	Pseudonyme de l'utilisateur	Utilisateur_Pseudonyme	E	AN	20		JeanP_90000
	Mot de passe de l'utilisateur	Utilisateur_HashPassword	E	AN	255		\$2y\$10\$CJ1ik/JTqArXq6f1yDaqm..pex9odGp00K/DvzET3fu.bYbudzCce
	Date de naissance de l'utilisateur	Utilisateur_DateNaissance	E	Date	10		29/08/1985
	Rue de l'utilisateur	Utilisateur_Rue	E	AN	100		5 Rue de la Rotonde
	E-mail de l'utilisateur	Utilisateur_Email	E	AN	50		jean.pierre.90000@gmail.com
	Portable de l'utilisateur	Utilisateur_TelMobile	E	N	10		0645264125

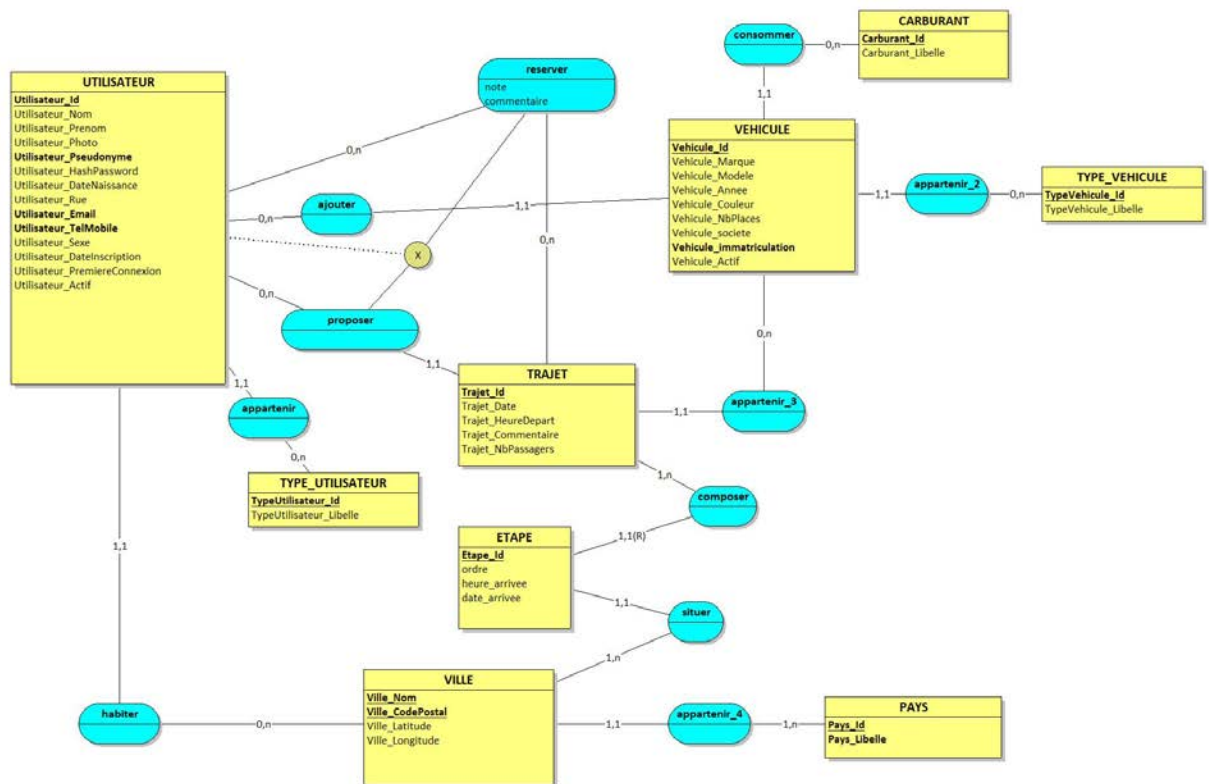
	Sexe de l'utilisateur	Utilisateur_Sexe	E	A	10		Homme
	Date d'inscription de l'utilisateur	Utilisateur_DateInscription	E	Date	10		10/10/2020
	Première connexion	Utilisateur_PremiereConnexion	E	Boolean	1		Oui
	Utilisateur actif	Utilisateur_Actif	E	Boolean	1		Oui
TYPE_UTILISATEUR							
	Identifiant du type utilisateur	TypeUtilisateur_Id	E	A	10	Oui	ADMIN
	Libellé du type utilisateur	TypeUtilisateur_Libelle	E	A	20		Administrateur
VEHICULE							
	Identifiant du véhicule	Vehicule_Id	E	N	8	Oui	00001254
	Marque du véhicule	Vehicule_Marque	E	A	20		AUDI
	Modèle du véhicule	Vehicule_Modele	E	A	20		A3
	Année du véhicule	Vehicule_Annee	E	N	4		2010
	Couleur du véhicule	Vehicule_Couleur	E	A	20		Noire
	Nombre de places dans le véhicule	Vehicule_NbPlaces	E	N	2		5
	Véhicule de société	Vehicule_societe	E	Boolean	1		Non
	Immatriculation du véhicule	Vehicule_immatriculation	E	AN	12		AA-219-AA
	Véhicule actif	Vehicule_Actif	E	Boolean	1		Oui

TYPE_VEHICULE							
	Identifiant du type véhicule	TypeVehicule_Id	E	N	2	Oui	1
	Libellé du type véhicule	TypeVehicule_Libelle	E	A	20		Voiture
CARBURANT							
	Identifiant du carburant	Carburant_Id	E	AN	5	Oui	E10
	Libelle du carburant	Carburant_Libelle	E	A	30		Sans Plomb 95 - E 10
TRAJET							
	Identifiant du trajet	Trajet_Id	E	N	10	Oui	0000002145
	Date du trajet	Trajet_Date	E	Date	10		15/10/2020
	Heure du départ	Trajet_HeureDepart	E / Ca	Time	5		15:00
	Commentaire du conducteur	Trajet_Commentaire	E	AN	255		Pas de gros bagages SVP
ETAPE	Nombre de passagers	Trajet_NbPassagers	E	N	2		3
	Id de l'étape	Etape_Id	E	N	2		2
VILLE							
	Nom de la ville	Ville_Nom	E	A	40	Oui	METZ
	Code postal de la ville	Ville_CodePostal	E	A	5	Oui	57000
PAYS	Latitude de la ville	Ville_Latitude	E	N	10		49.133331
	Longitude de la ville	Ville_Longitude	E	N	10		6.16667
	Identifiant du pays	Pays_Id	E	A	3	Oui	FR

reserver

Libellé du Pays	Pays_Libelle	E	A	20		France
Note du trajet réservé	note	E	N	1		5
Commen taire du trajet réservé	commentaire	E	AN	255		Le trajet s'est très bien déroulé ! :)

4.2 MODÈLE CONCEPTUEL DE DONNÉES (MCD)

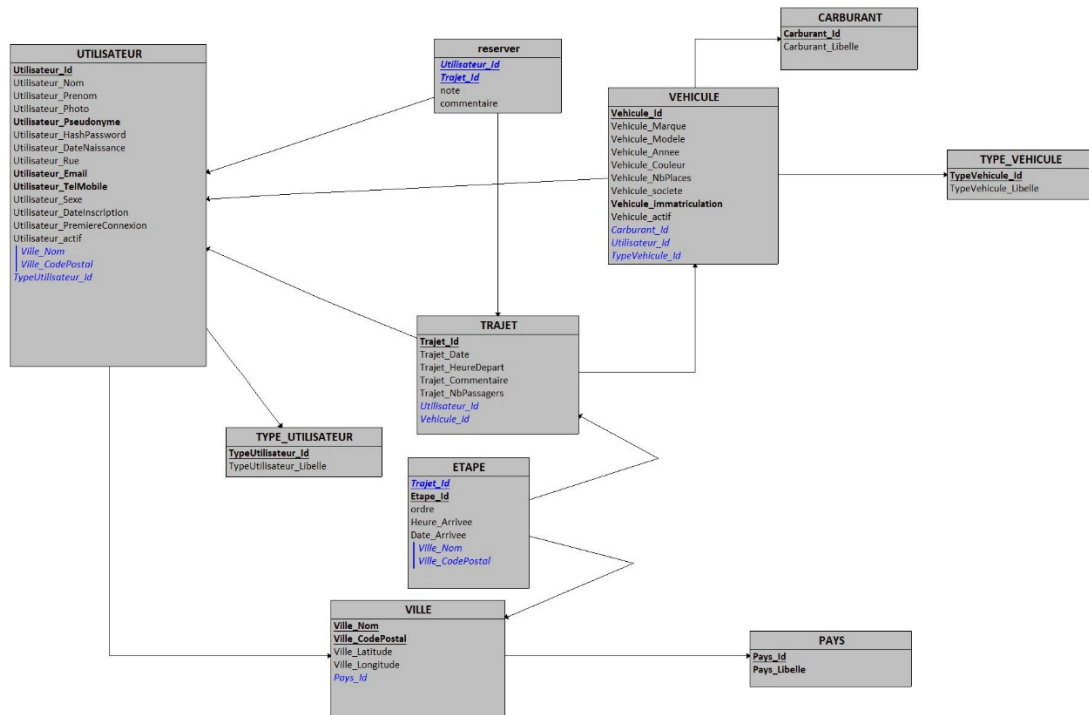


4.3 EXPLICATION DES ASSOCIATIONS ET DES CARDINALITÉS

Entités	Associations	Cardinalités	Justifications
UTILISATEUR	habiter	1,1	Un utilisateur habite dans une et une seule ville.
UTILISATEUR	appartenir	1,1	Un utilisateur appartient à un et un seul type d'utilisateur.
UTILISATEUR	proposer	0,n	Un utilisateur peut ne pas proposer de trajet ou en proposer plusieurs.
UTILISATEUR	réserver	0,n	Un utilisateur peut ne pas réserver de trajet ou en réserver plusieurs.
UTILISATEUR	ajouter	0,n	Un utilisateur peut ajouter aucun ou plusieurs véhicules.
TYPE_UTILISATEUR	appartenir	0,n	Un type utilisateur peut être appartenu par aucun ou plusieurs utilisateurs.
VEHICULE	ajouter	1,1	Un véhicule peut être ajouté que par un et un seul utilisateur.
VEHICULE	proposer	0,n	Un véhicule peut être proposé pour aucun ou plusieurs trajets.
VEHICULE	consommer	1,1	Un véhicule peut consommer un et un seul carburant.
VEHICULE	Appartenir_2	1,1	Un véhicule appartient à un et un seul type de véhicule.
VEHICULE	Appartenir_3	0,n	Un véhicule peut appartenir à aucun ou plusieurs trajets.
CARBURANT	consommer	0,n	Un carburant peut être consommé par aucun ou plusieurs véhicules.
TYPE_VEHICULE	Appartenir_2	0,n	Un type de véhicule peut être appartenu par aucun ou plusieurs véhicules.

TRAJET	proposer	1,1	Un trajet peut être proposé par un et un seul utilisateur
TRAJET	réserver	0,n	Un trajet peut être réservé par aucun ou plusieurs utilisateurs
TRAJET	Appartenir_3	1,1	Un trajet peut être lié qu'à un et un seul véhicule
TRAJET	composer	1,n	Un trajet peut être composé d'une ou plusieurs étapes
VILLE	habiter	0,n	Une ville peut être habitée par aucun ou plusieurs utilisateurs
VILLE	Appartenir_4	1,1	Une ville appartient à un et un seul pays
VILLE	situer	0,n	Une ville peut être située dans aucune ou plusieurs étapes
PAYS	Appartenir_4	1,n	Un pays peut avoir une ou plusieurs villes
ETAPE	composer	1,1(R)	Une étape peut composer un et un seul trajet. Une étape n'existe pas dans un trajet. Identifiant relatif
ETAPE	situer	1,1	Une étape peut être située dans une et une seule ville.

4.4 MODÈLE LOGIQUE DES DONNÉES RELATIONNEL NORMALISÉ



TYPE_UTILISATEUR (TypeUtilisateur Id VARCHAR(10), TypeUtilisateur_Libelle VARCHAR(20));

TYPE_VEHICULE (TypeVehicule Id BYTE, TypeVehicule_Libelle VARCHAR(20));

CARBURANT (Carburant Id VARCHAR(5), Carburant_Libelle VARCHAR(30));

PAYS (Pays Id VARCHAR(3), Pays_Libelle VARCHAR(20));

VILLE (Ville Nom VARCHAR(40), Ville CodePostal CHAR(5), Ville_Latitude REAL, Ville_Longitude INT, #Pays_Id);

UTILISATEUR (Utilisateur Id INT, Utilisateur_Nom VARCHAR(25), Utilisateur_Prenom VARCHAR(20), Utilisateur_Photo VARCHAR(255), Utilisateur_Pseudonyme VARCHAR(20), Utilisateur_HashPassword VARCHAR(255), Utilisateur_DateNaissance DATE, Utilisateur_Rue VARCHAR(100), Utilisateur_Email VARCHAR(50), Utilisateur_TelMobile INT, Utilisateur_Sexe VARCHAR(10), Utilisateur_DateInscription DATE, Utilisateur_PremiereConnexion LOGICAL, Utilisateur_actif LOGICAL, #(Ville_Nom, Ville_CodePostal), #TypeUtilisateur_Id);

VEHICULE (Vehicule Id INT, Vehicule_Marque VARCHAR(20), Vehicule_Modele VARCHAR(20), Vehicule_Annee BYTE, Vehicule_Couleur VARCHAR(20), Vehicule_NbPlaces BYTE, Vehicule_societe LOGICAL, Vehicule_immatriculation VARCHAR(12), Vehicule_actif VARCHAR(50), #Carburant_Id, #Utilisateur_Id, #TypeVehicule_Id);

TRAJET (Trajet Id INT, Trajet_Date DATE, Trajet_HeureDepart TIME, Trajet_Commentaire VARCHAR(255), Trajet_NbPassagers BYTE, #Utilisateur_Id, #Vehicule_Id);

ETAPE (#Trajet_Id, Etape Id BYTE, ordre BYTE, Heure_Arrivee TIME, Date_Arrivee DATE, #(Ville_Nom, Ville_CodePostal));

reserver (#Utilisateur_Id, #Trajet_Id, note BYTE, commentaire VARCHAR(350));

4.5 SCRIPT DE CRÉATION DE BASE DE DONNÉES

```
/* Création de la base de données GSBCarpooling */  
  
CREATE DATABASE IF NOT EXISTS GSBCarpooling CHARACTER SET 'utf8';  
  
/* Utilisation de la base de données */  
  
USE GSBCarpooling;  
  
/* Création des différentes tables de la base de données */  
  
CREATE TABLE TYPE_UTILISATEUR(  
    TypeUtilisateur_Id VARCHAR(10),  
    TypeUtilisateur_Libelle VARCHAR(20) NOT NULL,  
    PRIMARY KEY(TypeUtilisateur_Id)  
);  
  
CREATE TABLE TYPE_VEHICULE(  
    TypeVehicule_Id TINYINT,  
    TypeVehicule_Libelle VARCHAR(20) NOT NULL,  
    PRIMARY KEY(TypeVehicule_Id)  
);  
  
CREATE TABLE CARBURANT(  
    Carburant_Id VARCHAR(5),  
    Carburant_Libelle VARCHAR(30) NOT NULL,  
    PRIMARY KEY(Carburant_Id)  
);  
  
CREATE TABLE PAYS(  
    Pays_Id VARCHAR(3),  
    Pays_Libelle VARCHAR(20) NOT NULL,  
    PRIMARY KEY(Pays_Id),  
    UNIQUE(Pays_Libelle)  
);  
  
CREATE TABLE VILLE(  
    Ville_Nom VARCHAR(40),  
    Ville_CodePostal CHAR(5),  
    Ville_Latitude REAL NOT NULL,  
    Ville_Longitude REAL NOT NULL,  
    Pays_Id VARCHAR(3) NOT NULL,  
    PRIMARY KEY(Ville_Nom, Ville_CodePostal),  
    FOREIGN KEY(Pays_Id) REFERENCES PAYS(Pays_Id)  
);  
  
CREATE TABLE UTILISATEUR(  
    Utilisateur_Id INT,  
    Utilisateur_Nom VARCHAR(25) NOT NULL,  
    Utilisateur_Prenom VARCHAR(20) NOT NULL,  
    Utilisateur_Photo VARCHAR(255),  
    Utilisateur_Pseudonyme VARCHAR(20) NOT NULL,  
    Utilisateur_HashPassword VARCHAR(255) NOT NULL,  
    Utilisateur_DateNaissance DATE NOT NULL,  
    Utilisateur_Rue VARCHAR(100) NOT NULL,  
    Utilisateur_Email VARCHAR(50) NOT NULL,  
    Utilisateur_TelMobile INT NOT NULL,  
    Utilisateur_Sexe VARCHAR(10) NOT NULL,
```



```

Utilisateur_DateInscription DATE NOT NULL,
Utilisateur_PremiereConnexion BIT NOT NULL,
Ville_Nom VARCHAR(40) NOT NULL,
Ville_CodePostal CHAR(5) NOT NULL,
TypeUtilisateur_Id VARCHAR(10) NOT NULL,
Utilisateur_Actif BIT NOT NULL,
PRIMARY KEY(Utilisateur_Id),
UNIQUE(Utilisateur_Pseudonyme),
UNIQUE(Utilisateur_Email),
UNIQUE(Utilisateur_TelMobile),
FOREIGN KEY(Ville_Nom, Ville_CodePostal) REFERENCES VILLE(Ville_Nom, Ville_CodePostal),
FOREIGN KEY(TypeUtilisateur_Id) REFERENCES TYPE_UTILISATEUR(TypeUtilisateur_Id)
);

```

```

CREATE TABLE VEHICULE(
    Vehicule_Id INT,
    Vehicule_Marque VARCHAR(20) NOT NULL,
    Vehicule_Modele VARCHAR(20) NOT NULL,
    Vehicule_Annee TINYINT NOT NULL,
    Vehicule_Couleur VARCHAR(20) NOT NULL,
    Vehicule_NbPlaces TINYINT NOT NULL,
    Vehicule_societe BIT NOT NULL,
    Vehicule_immatriculation VARCHAR(12) NOT NULL,
    Carburant_Id VARCHAR(5) NOT NULL,
    Utilisateur_Id INT NOT NULL,
    TypeVehicule_Id TINYINT NOT NULL,
    Vehicule_Actif BIT NO NULL,
    PRIMARY KEY(Vehicule_Id),
    UNIQUE(Vehicule_immatriculation),
    FOREIGN KEY(Carburant_Id) REFERENCES CARBURANT(Carburant_Id),
    FOREIGN KEY(Utilisateur_Id) REFERENCES UTILISATEUR(Utilisateur_Id),
    FOREIGN KEY(TypeVehicule_Id) REFERENCES TYPE_VEHICULE(TypeVehicule_Id)
);

```

```

CREATE TABLE TRAJET(
    Trajet_Id INT,
    Trajet_Date DATE NOT NULL,
    Trajet_HeureDepart TIME NOT NULL,
    Ville_Depart VARCHAR(40) NOT NULL,
    Trajet_Commentaire VARCHAR(255),
    Trajet_NbPassagers TINYINT NOT NULL,
    Vehicule_Id INT NOT NULL,
    PRIMARY KEY(Trajet_Id),
    FOREIGN KEY(Vehicule_Id) REFERENCES VEHICULE(Vehicule_Id)
);

```

```

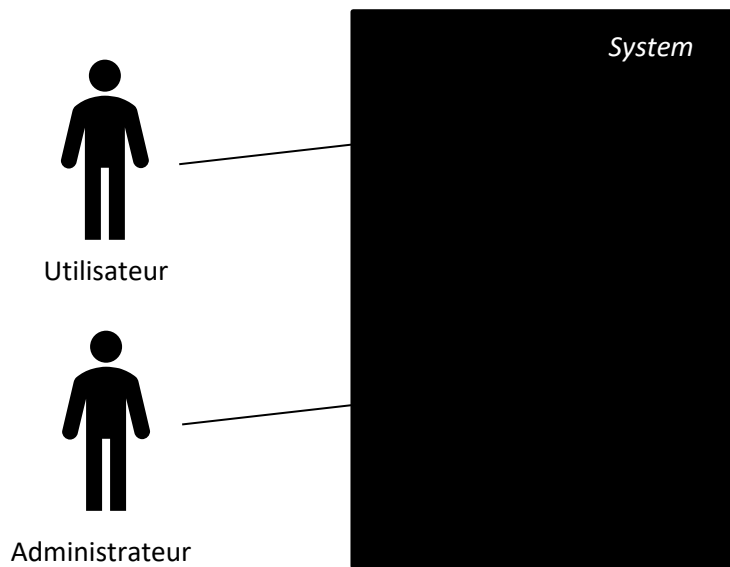
CREATE TABLE ETAPE(
    Trajet_Id INT,
    Etape_Id TINYINT,
    ordre INT,
    Heure_Arrivee TIME,
    Date_Arrivee DATE,
    Ville_Nom VARCHAR(40) NOT NULL,
    Ville_CodePostal CHAR(5) NOT NULL,
    PRIMARY KEY(Trajet_Id, Etape_Id),
    FOREIGN KEY(Trajet_Id) REFERENCES TRAJET(Trajet_Id),
    FOREIGN KEY(Ville_Nom, Ville_CodePostal) REFERENCES VILLE(Ville_Nom, Ville_CodePostal)
);

```

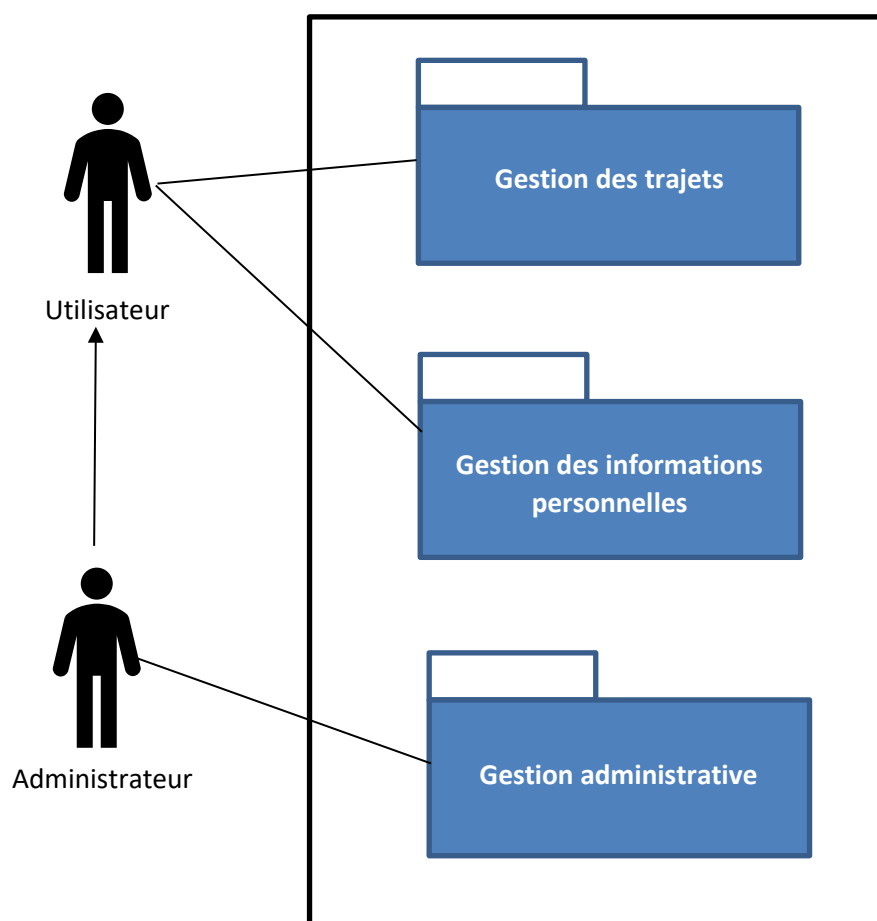
```
CREATE TABLE RESERVER(  
  Utilisateur_Id INT,  
  Trajet_Id INT,  
  note TINYINT,  
  commentaire VARCHAR(350),  
  PRIMARY KEY(Utilisateur_Id, Trajet_Id),  
  FOREIGN KEY(Utilisateur_Id) REFERENCES UTILISATEUR(Utilisateur_Id),  
  FOREIGN KEY(Trajet_Id) REFERENCES TRAJET(Trajet_Id)  
);
```

4.6 MODÉLISATION UML

4.6.1 Diagramme de contexte

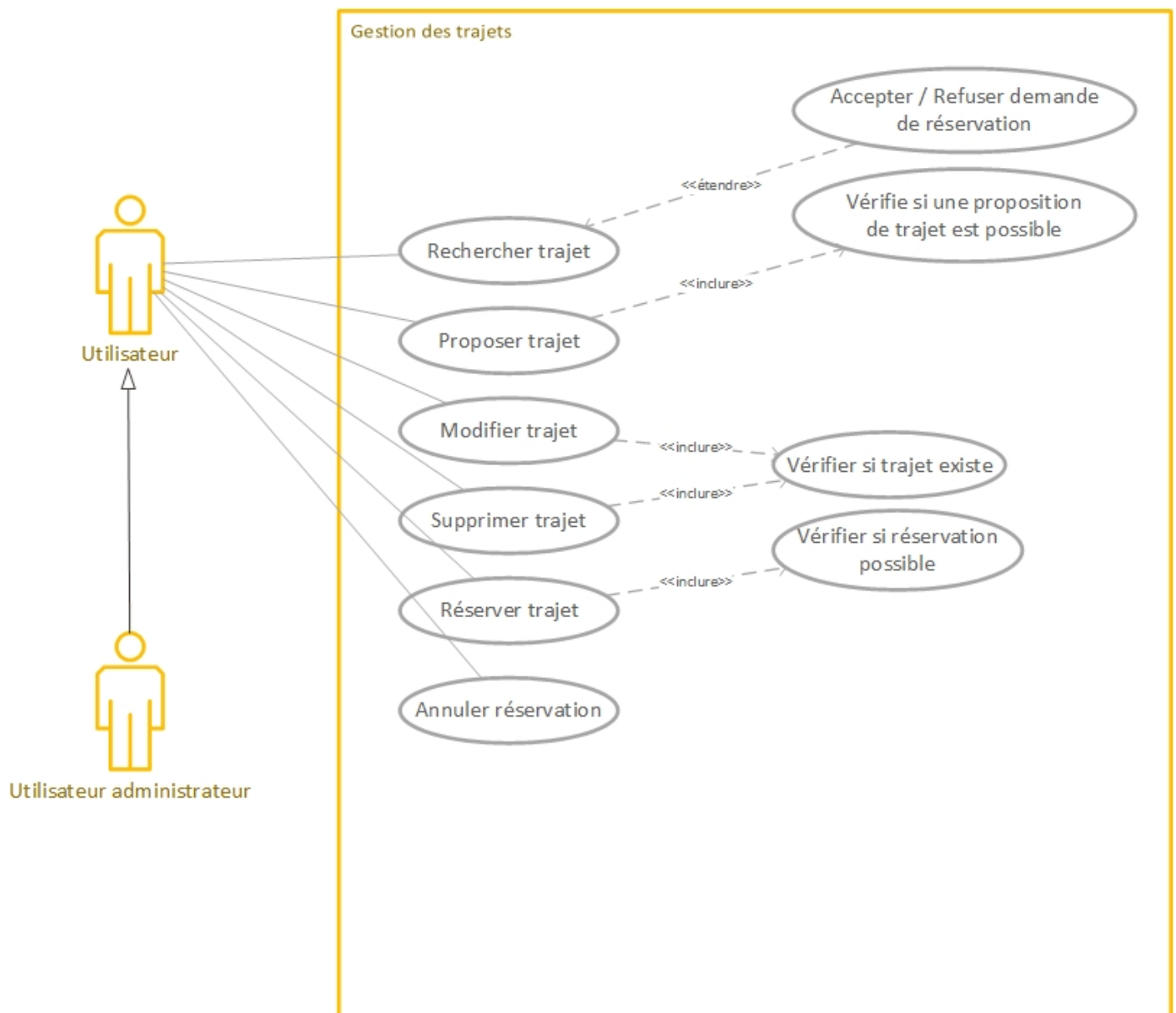


4.6.2 Diagramme de packages



4.6.3 Diagrammes de cas d'utilisation

4.6.3.1 Diagramme de cas d'utilisation : Gestion des trajets



Objectif

L'acteur principal pourra rechercher des trajets ; proposer des trajets ; modifier ou supprimer un de ses trajets en cours. Réserver un trajet ou annuler sa demande de réservation.

Préconditions

Pour que le cas d'utilisation puisse être déclenché, il faut que l'utilisateur soit connecté à l'application et à la base de données. Cette dernière doit être disponible.

Scénarios

- L'utilisateur pourra rechercher un trajet à une date voulue en spécifiant la ville de destination. Si un retour est positif alors l'utilisateur pourra faire une demande de réservation. Une réservation sera possible pour l'utilisateur seulement s'il n'a pas déjà une réservation en cours au même moment et s'il n'a pas proposé un trajet à la même date.

- Lorsque l'utilisateur voudra proposer un trajet, celui devra renseigner une date de départ, une heure de départ, une ville de destination et s'il le souhaite, il pourra ajouter des étapes au trajet. Pour que le trajet soit valide, il ne faut pas qu'un trajet du même utilisateur soit à la même période.
- L'utilisateur souhaite apporter une modification à son trajet. Changer la date ou l'heure du départ ne sera possible que s'il n'y a aucune réservation acceptée. Il pourra changer le commentaire à tout moment.
- L'utilisateur ne souhaite plus effectuer le trajet qu'il propose, il peut alors tout simplement supprimer son trajet.
- L'utilisateur a procédé à une réservation, il a la possibilité de se rétracter et annuler sa demande de réservation si la date et l'heure du trajet ne sont pas dépassées.

Post conditions

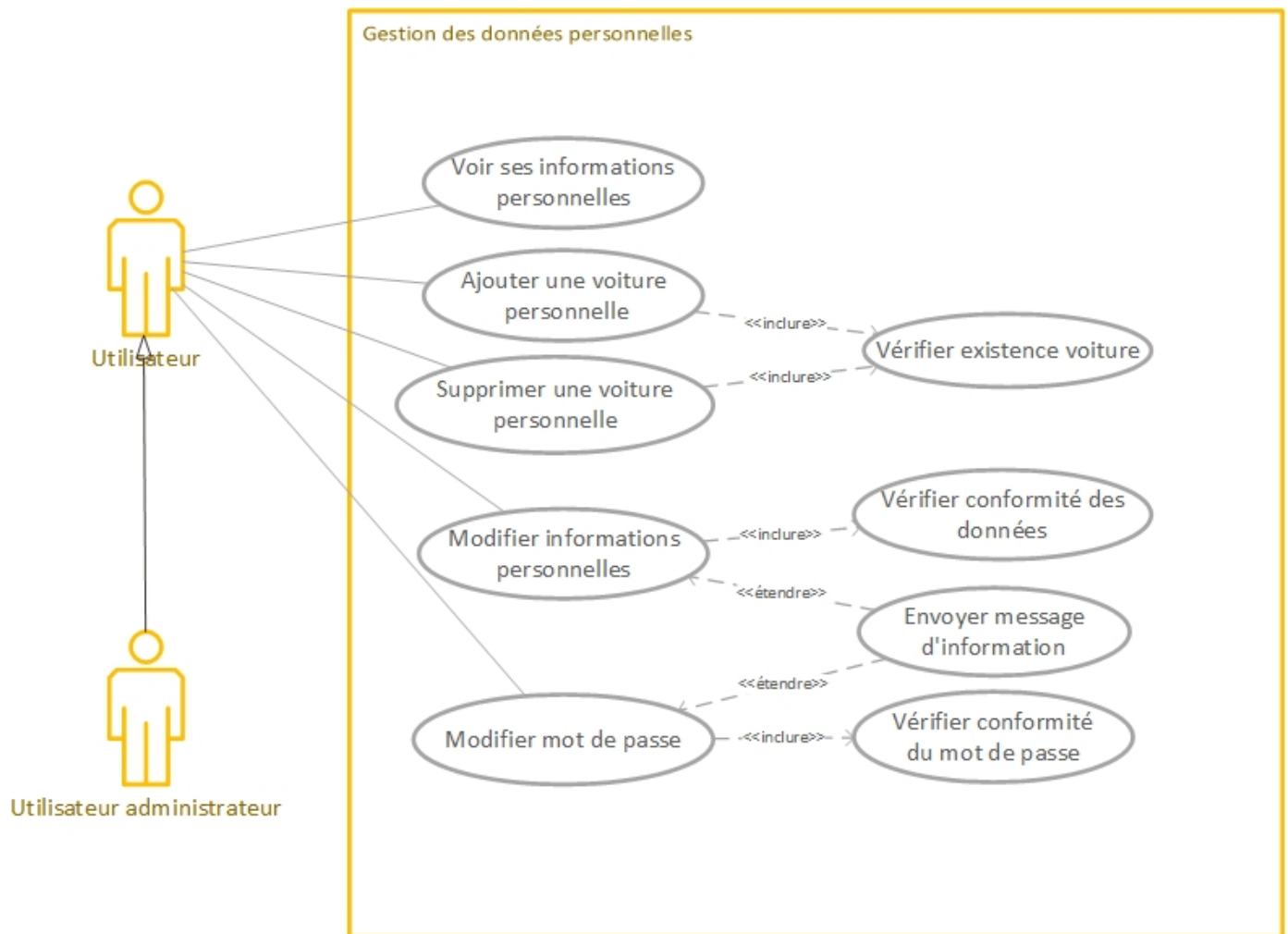
Après les actions « proposer un trajet », « modifier un trajet », « supprimer un trajet », « réserver un trajet », « annuler réservation », la base de données sera mise à jour. Dans le cas d'une simple consultation des trajets, le système n'aura reçu aucune modification.

Date de réalisation : 07/11/2020

Responsable : CHANOT Alexandre

Version : 1.0

4.6.3.2 Diagramme de cas d'utilisation : Gestion des données personnelles



Objectif

L'acteur principal pourra voir ses informations personnelles, ajouter ou supprimer une voiture personnelle, modifier ses informations personnelles, modifier son mot de passe.

Préconditions

Pour que le cas d'utilisation puisse être déclenché, il faut que l'utilisateur soit connecté à l'application et à la base de données. Cette dernière doit être totalement disponible.

Scénarios

- L'utilisateur clique sur sa fiche personnelle et accède à toutes ses informations personnelles à titre de consultation.
- En allant dans ses données personnelles, l'utilisateur peut ajouter une voiture qui sera spécifiée lors des trajets. Il devra saisir la marque du véhicule, son modèle, l'année, la couleur, le carburant utilisé ainsi que son numéro d'immatriculation. Si la voiture existe déjà dans la base de données, elle ne sera pas ajoutée et un message informe l'utilisateur.
- L'utilisateur a la possibilité de supprimer une voiture créée. Après confirmation, la voiture est tout simplement supprimée de la base de données.
- L'utilisateur a besoin de changer ses informations personnelles. Une fois sur la fenêtre de ses informations, il change tout simplement les informations qu'il souhaite changer. Une vérification de l'adresse mail, du numéro de téléphone et du pseudonyme sera effectuée

lors de la validation de la modification pour vérifier si un doublon est possible. Si c'est le cas alors un message alertera l'utilisateur et le système ne procédera pas à la modification.

- L'utilisateur souhaite modifier son mot de passe. Cette action s'effectue au même endroit que pour modifier ses données personnelles. Une fois le mot de passe modifier et la demande de modification effectuée, une vérification de la conformité du mot de passe est faite. En cas d'irrégularité, un message d'erreur apparaît et le système ne procède pas au changement de mot de passe.

Post conditions

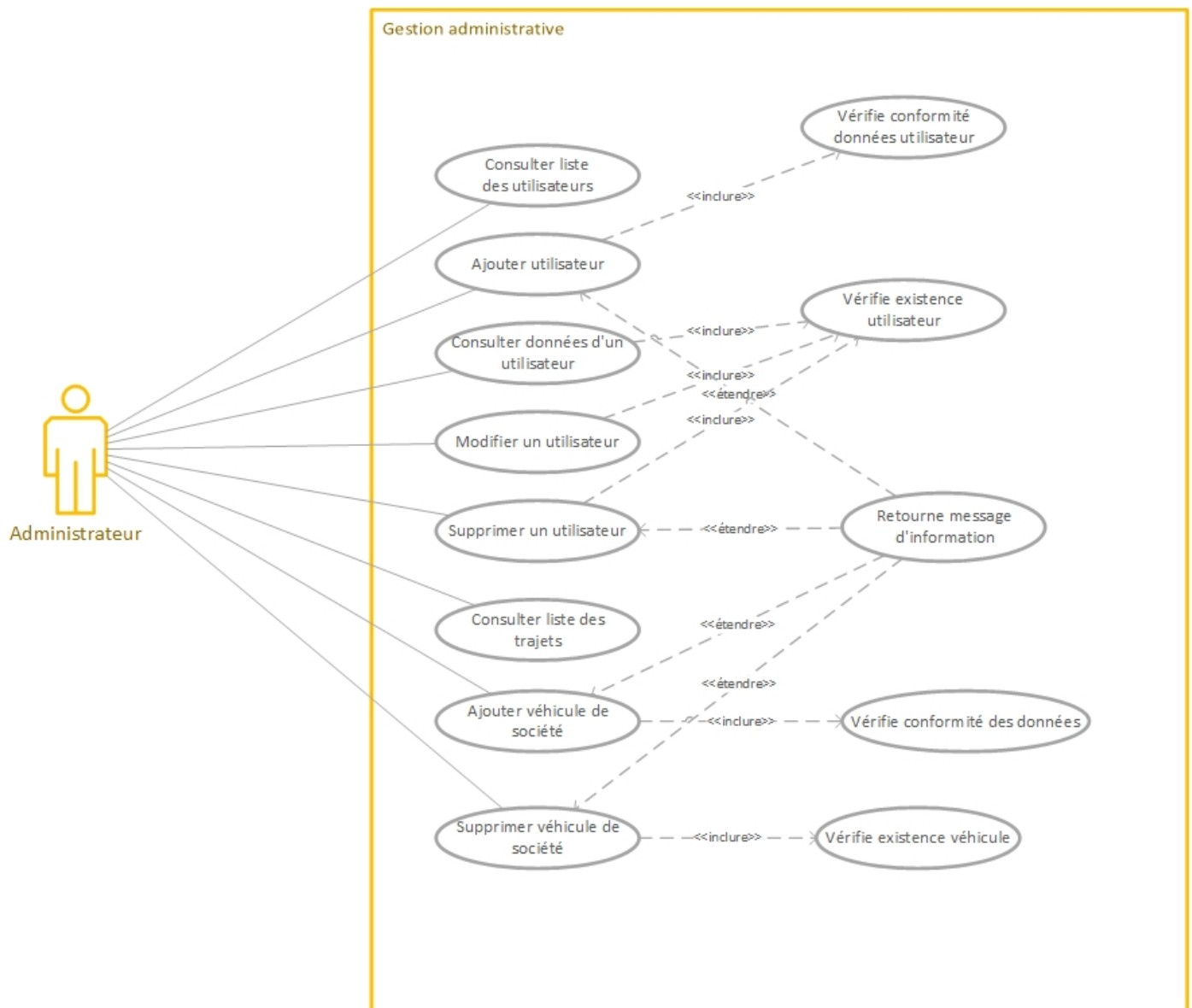
Après les actions « ajouter une voiture », « supprimer une voiture », « modifier informations personnelles » et « modifier mot de passe », la base de données sera mise à jour. Dans le cas de l'action de consultation, le système reste inchangé.

Date de réalisation : 07/11/2020

Responsable : CHANOT Alexandre

Version : 1.0

4.6.3.3 Diagramme de cas d'utilisation : Gestion administrative



Objectif

L'acteur principal pourra consulter la liste des utilisateurs, ajouter, modifier, supprimer un utilisateur. Il pourra également consulter la liste des trajets, modifier, supprimer un trajet. Ajouter, supprimer une voiture de société.

Préconditions

Pour que le cas d'utilisation puisse être déclenché, il faut que l'utilisateur soit connecté à l'application en tant qu'administrateur et être connecté à la base de données. Cette dernière doit être totalement disponible.

Scénarios

- L'utilisateur, administrateur souhaite consulter la liste des utilisateurs. Celle-ci s'affiche et n'infecte en rien le système.
- L'administrateur souhaite ajouter un nouvel utilisateur. Celui-ci devra renseigner un formulaire avec toutes les données nécessaires : nom, prénom, pseudonyme, mot de passe, date de naissance, rue,

ville, e-mail, téléphone, sexe, type utilisateur. Des vérifications seront effectuées lors de la validation du formulaire pour vérifier si l'utilisateur existe déjà ou s'il y a un risque de doublon avec le pseudonyme, le téléphone ou l'adresse mail. Si c'est le cas alors l'action de création s'arrête et un message d'avertissement est retourné.

- L'acteur souhaite consulter les données d'un utilisateur précis. Une vérification est donc effectuée pour vérifier si l'utilisateur existe bien (requête) et accéder à sa fiche utilisateur.

- L'acteur principal souhaite modifier les données d'un utilisateur. Après validation, des vérifications sont effectuées pour vérifier s'il y a un risque de doublon avec le pseudonyme, le téléphone ou l'adresse mail de l'utilisateur. Si c'est le cas alors l'action de création s'arrête et un message d'avertissement est retourné.

- L'administrateur souhaite supprimer un utilisateur. Le système vérifie si l'utilisateur existe bien et procède à tous les traitements nécessaires à la suppression de l'utilisateur.

Une fois le traitement terminé, un message de confirmation est retourné.

- L'utilisateur, administrateur souhaite consulter la liste des trajets. Celle-ci s'affiche et n'infecte en rien le système.

- L'acteur souhaite consulter les données d'un trajet précis. Une vérification est donc effectuée pour vérifier si le trajet existe bien et accéder à sa fiche.

- L'administrateur souhaite modifier un trajet. Celui-ci peut supprimer que le commentaire du trajet. Un message de validation est retourné une fois l'action effectué.

- L'administrateur souhaite supprimer un trajet. Le système vérifie si le trajet existe bien et procède à tous les traitements nécessaires à la suppression. Un message de confirmation est retourné une fois l'action effectué.

- L'utilisateur souhaite ajouter une voiture de société dans la base de données. Celui-ci devra renseigner toutes les informations de la voiture : marque, modèle, année, couleur, nombre de places, carburant, type de véhicule, immatriculation. Une fois le formulaire validé, une vérification sera effectuée et un message sera renvoyé à l'utilisateur pour l'informer de la bonne exécution de l'action.

- L'administrateur souhaite supprimer un véhicule. Le système vérifie si le véhicule existe bien et procède à tous les traitements nécessaires à la suppression. Un message de confirmation est retourné une fois l'action effectué.

Post conditions

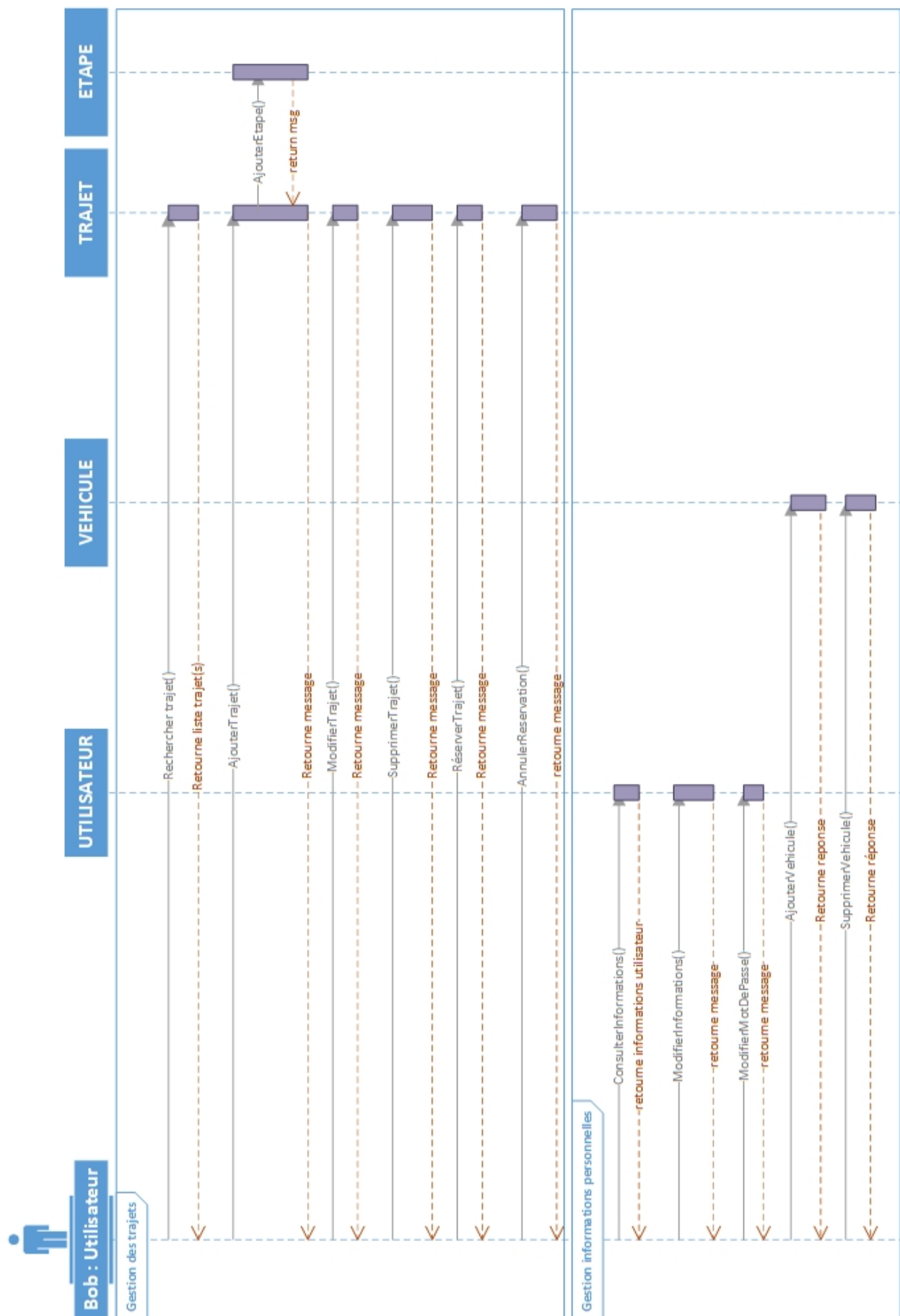
Après les actions « consulter utilisateur », « modifier utilisateur », « supprimer utilisateur », « consulter un trajet », « modifier un trajet », « supprimer un trajet », « ajouter une voiture » et « supprimer une voiture », la base de données sera mise à jour. Dans le cas de l'action de consultation d'un utilisateur, d'un trajet, de la liste des utilisateurs ou la liste des trajets, le système reste inchangé.

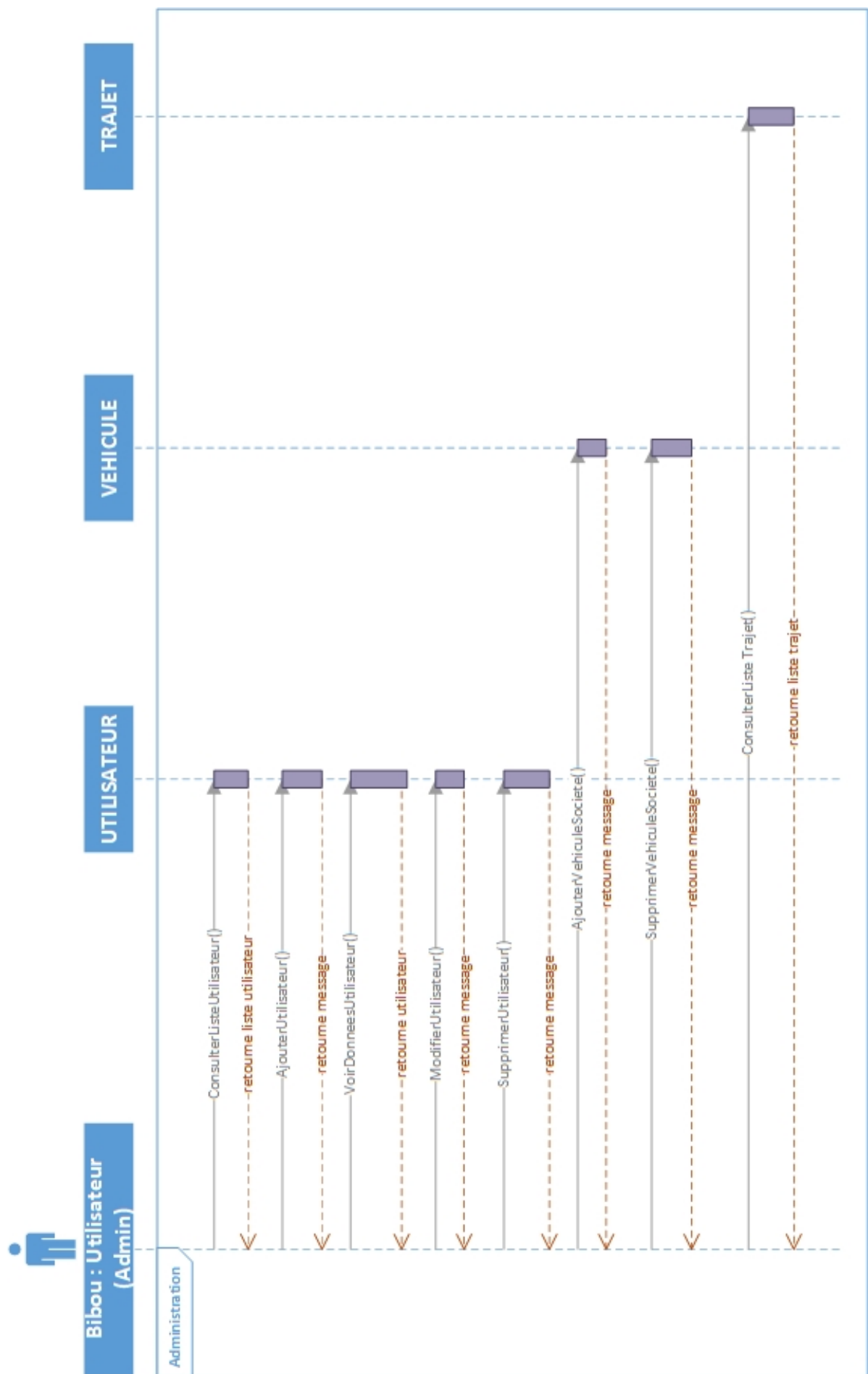
Date de réalisation : 07/11/2020

Responsable : CHANOT Alexandre

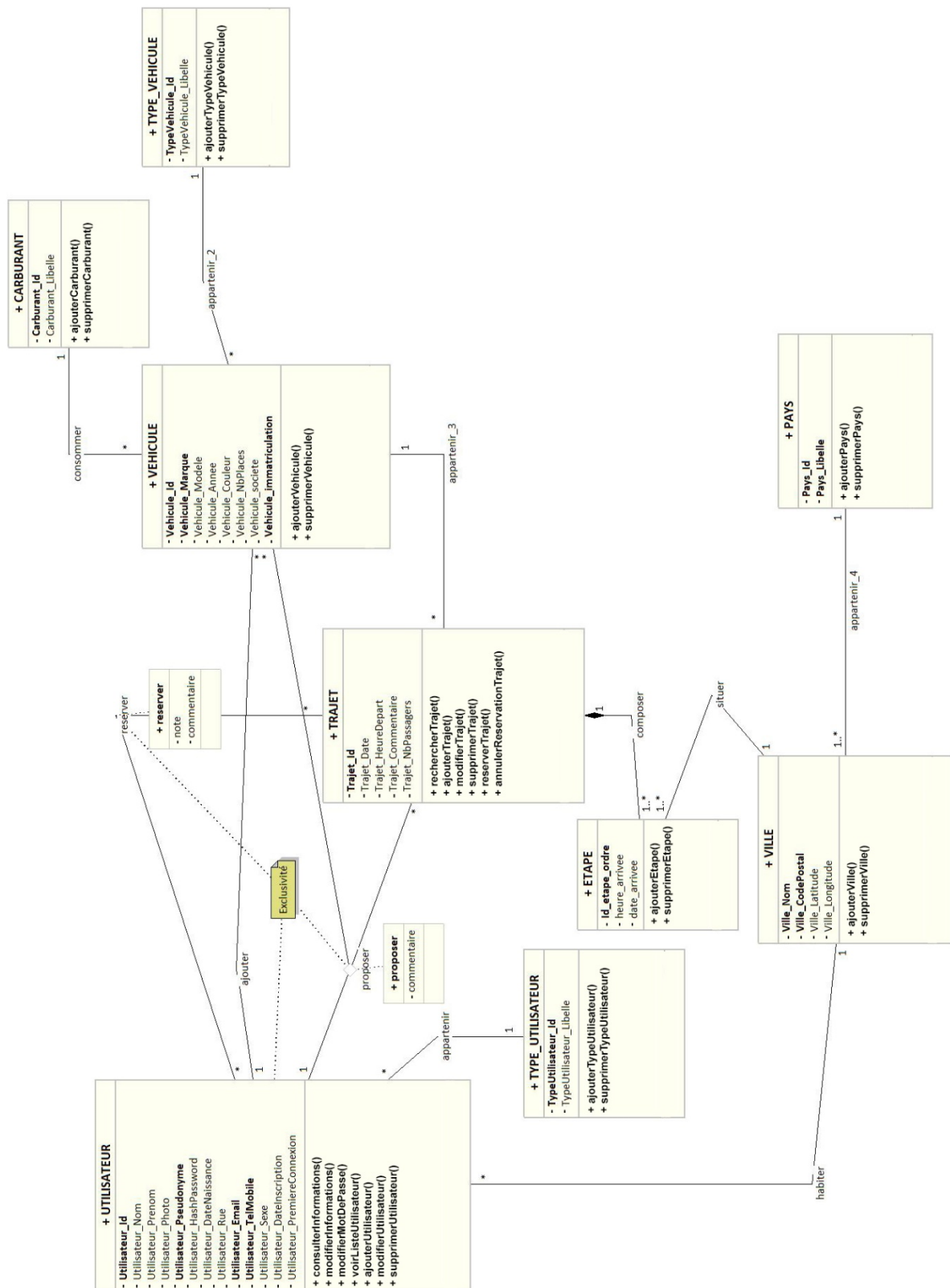
Version : 1.0

4.6.4 Diagrammes de séquences (niveau analyse)





4.6.5 Diagramme de classes



5 FINALISATION DU PROJET

5.1 IMH

5.1.1 Introduction

Pour développer cette application, il a été retenu d'avoir une partie utilisateur et une partie administration propre à elle-même. La partie utilisateur est pensée pour que l'utilisateur ait tout ce qu'il a besoin en très peu de clics. Les onglets et les fenêtres sont donc simples et intuitives.

5.1.2 Ouverture du programme et connexion de l'utilisateur



Le programme démarre donc sur la fenêtre d'authentification.

L'utilisateur renseigne alors son nom d'utilisateur et son mot de passe.

Si l'utilisateur est un administrateur, il lui sera demandé s'il souhaite accéder à l'espace d'administration ou non.

5.1.3 Administration – Fenêtre d'accueil



La page d'accueil d'administration se veut simple et épurée.

L'administrateur accède rapidement à un menu.

De gauche à droite il lui est possible de : « **Gérer les utilisateurs** », « **Gérer les véhicules** », « **Gérer les trajets** », « **Se déconnecter de l'espace d'administration** ».

5.1.4 Administration – Gestion des utilisateurs

ID	Nom	Prénom	Pseudonyme	E-Mail	Mobile	Adresse	Ville
1	CHANOT	ALEXANDRE	a.chanot	calex555@hotma...	0645210235	5 Rue de la Plèbe	BESAN
2	FONT	Clém	c.fonteyne	fndfhgf@fhfhf.fr	0652145410	5 Rue des étoiles	VALDA
7	GUERITEY	Sébastien	s.gueritey	sgueritey@peuge...	0685411275	5 Rue du Champ...	NANCY
8	PARISOT	Pierre	p.parisot	pierre@free.fr	0652412125	5 rue du test	PARIS

Cette fenêtre permet à l’administrateur de consulter une table contenant les différents utilisateurs. Il est possible de faire un tri sur chaque colonne de celle-ci.

5 boutons composent cette fenêtre. De gauche à droite et de haut en bas, les boutons servent à : « Créer un nouvel utilisateur », « modifier un utilisateur » sélectionné, « supprimer un utilisateur » sélectionné, « actualiser la table » et enfin, « fermer » la fenêtre.

5.1.5 Administration – Gestion des utilisateurs – Ajout/Modification d’un utilisateur

Nom

Prénom

Pseudonyme

Date de naissance

Sexe

Adresse

Ville

Code postal

Pays

Type utilisateur

Mot de passe

Confirmer mot de passe

E-Mail

Numéro téléphone mobile

Cette fenêtre est utilisée pour l’ajout et la modification. Dans le second, les champs sont préremplis avec les données de l’utilisateur à modifier.

On retrouve chaque champ nécessaire à l’enregistrement d’un utilisateur.

Deux boutons sont présents pour valider le formulaire ou pour fermer la fenêtre et annuler l’opération.

5.1.6 Administration – Gestion des véhicules

ID	Marque	Modèle	Année	Couleur	Nb Places	Societe ?	Immatr
1	Peugeot	207	2010	Bleu	5	<input checked="" type="checkbox"/>	AA-125

Cette fenêtre est identique à la fenêtre de gestion des utilisateurs. Celle-ci permet à l'administrateur de consulter une table contenant les différentes voitures. Il est possible de faire un tri sur chaque colonne de celle-ci.

5 boutons composent également cette fenêtre. De gauche à droite et de haut en bas, les boutons servent à : « Ajouter un nouveau véhicule », « modifier un véhicule » sélectionné, « supprimer un véhicule » sélectionné, « actualiser la table » et enfin, « fermer » la fenêtre.

5.1.7 Administration – Gestion des véhicules – Ajout/Modification d'un véhicule

Marque

Modèle

Année

Couleur

Nb Places

Immatriculation

Type de véhicule

Carburant

ID Utilisateur

☐ Voiture de société ?

Valider Fermer

Cette fenêtre est utilisée pour l'ajout et la modification. Dans le second, les champs sont préremplis avec les données du véhicule à modifier.





On retrouve chaque champ nécessaire à l'enregistrement d'un véhicule.

Deux boutons sont présents pour valider le formulaire ou pour fermer la fenêtre et annuler l'opération.

5.1.8 Administration – Gestion des véhicules

Gestion des trajets

GSBCarpooling

ID	Date	Heure du départ	Commentaire	Véhicule	Nom
10	08/01/2021 ...	02:58:28	Petite balade :)	3	FONT
11	08/01/2021 ...	03:00:49	Petite balade 2 ! :D	1	CHANOT
12	08/01/2021 ...	10:15:51	gggggggggggggggggggggggg	1	CHANOT
13	08/01/2021 ...	10:31:49	aaaaaaaaaaaaaaaaaaaa	1	CHANOT
14	08/01/2021 ...	10:50:37	aaaaaaaaaaaaaaaaaaaaaaaaaaaa	1	CHANOT
15	08/01/2021 ...	10:53:25	bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb	3	FONT
16	08/01/2021 ...	10:55:22	cccccccccccccccccccccccccccccccccccc	1	CHANOT
17	08/01/2021 ...	10:57:52	Petits bagages :)	3	FONT
1002	12/01/2021 ...	20:32:46	petit test comme ça wallah	1	CHANOT

Fermer

Cette fenêtre est identique aux autres. Cela permet une certaine uniformité et permet à l'utilisateur de se repérer facilement. Celle-ci permet à l'administrateur de consulter une table contenant les différents trajets publiés par les utilisateurs. Il est possible de faire un tri sur chaque colonne de celle-ci.

5 boutons composent également cette fenêtre. De gauche à droite et de haut en bas, les boutons servent à : « Ajouter un nouveau trajet », « modifier un trajet » sélectionné, « supprimer un trajet » sélectionné, « actualiser la table » et enfin, « fermer » la fenêtre.

5.1.9 Administration – Gestion des trajets – Ajout/Modification d'un trajet

Proposer un trajet

Date de départ: jeudi 14 janvier 2021

Heure de départ: 22:42

Ville de départ:

Ville d'arrivée:

Commentaire (250 caractères max):

Liste des étapes:

Villes étape:

Ajouter une étape

Supprimer étape sélectionnée

Véhicule utilisé:

Nb passagers: 0

Valider Fermer

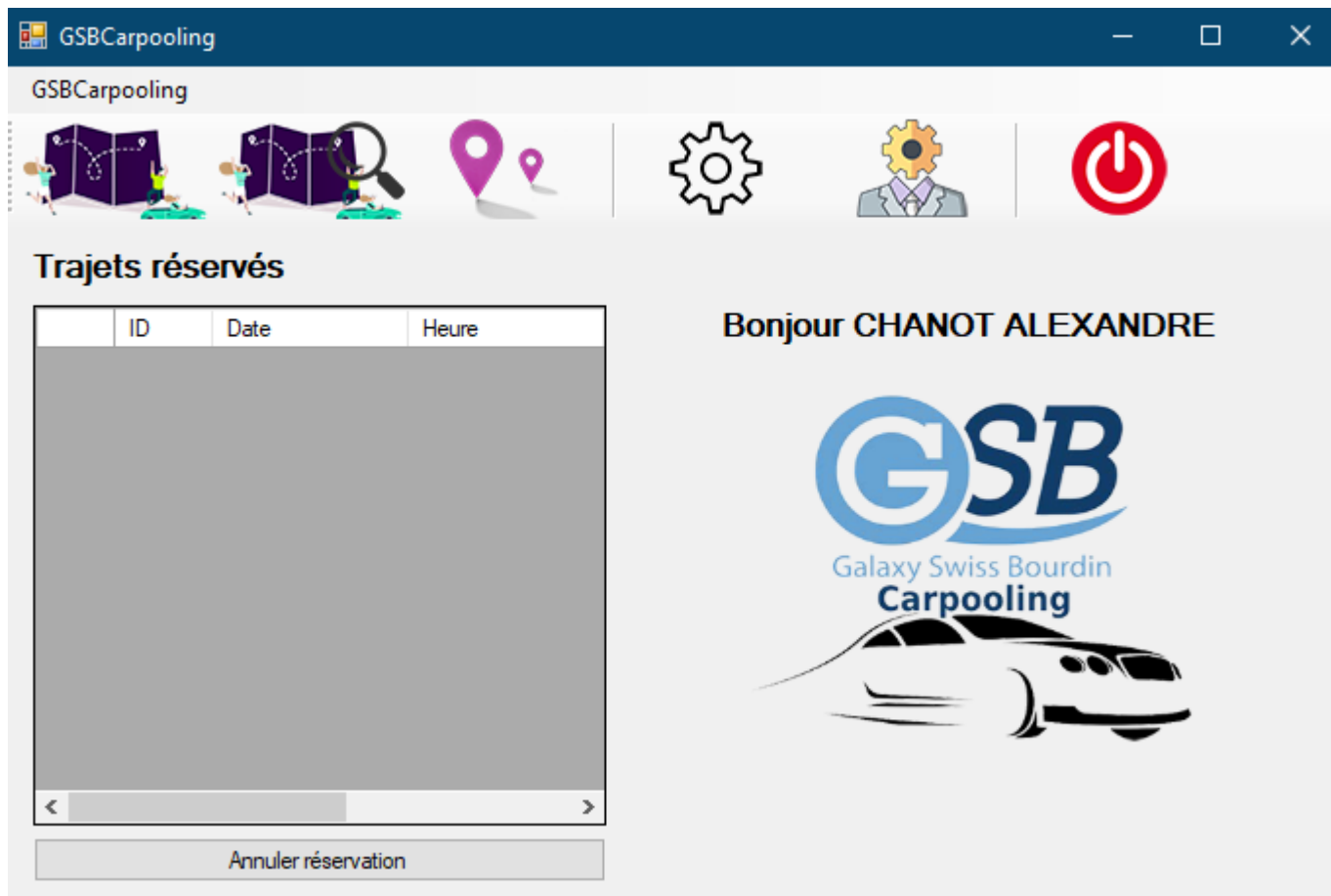
Cette fenêtre est utilisée pour l'ajout et la modification. Dans le second, les champs sont préremplis. Si aucune réservation n'a été faite, alors tout est modifiable. Sinon seul le commentaire le sera.

On retrouve chaque champ nécessaire à l'enregistrement d'un trajet.

L'utilisateur choisit une date et une heure de départ. Une ville de départ et d'arrivée. Il peut également ajouter des villes étape. Il doit renseigner le véhicule utilisé pour le trajet et le nombre de passager qu'il accepte. Enfin, il a la possibilité d'ajouter un commentaire.

Deux boutons sont présents pour valider le formulaire ou pour fermer la fenêtre et annuler l'opération.

5.1.10 Fenêtre d'accueil



Une fois l'identification passée, cette fenêtre s'ouvre à l'utilisateur.

Celle-ci se veut également simple et épurée. La liste des trajets réservés et à venir doit être visible directement pour une simple consultation. Connexion, consultation, fermeture de l'application... Un bouton est également disponible sous la table pour annuler rapidement la réservation d'un trajet sélectionné.

Un menu simple, épurée et découpé en 3 parties est disponible et propose donc à l'utilisateur, de gauche à droite de : « consulter ses trajets proposés », « rechercher un trajet », « Proposer un trajet », « consulter ses informations personnelles », « accéder à l'administration » (seulement disponible à un utilisateur administrateur), et enfin « quitter l'application ».

5.1.11 Fenêtre « Vos trajets »

#	Date départ	Heure départ	Ville départ	Ville arrivée
1002	12/01/2021 00:00...	20:32:46	1	4
17	08/01/2021 00:00...	10:57:52	1	4
15	08/01/2021 00:00...	10:53:25	1	3
14	08/01/2021 00:00...	10:50:37	1	4
13	08/01/2021 00:00...	10:31:49	1	4
11	08/01/2021 00:00...	03:00:49	1	3
*				

Cette fenêtre permet à l'utilisateur de visualiser les trajets qu'il a proposé et qui sont d'actualité.

2 boutons sont disponibles. De gauche à droite : « Modifier le trajet », « Supprimer le trajet »

Dans le cas d'une modification, l'utilisateur ouvre la même fenêtre vu précédemment : « Gestion des trajets – Ajout/Modification d'un trajet ».

5.1.12 Fenêtre « Rechercher un trajet »

ID	Date départ	Heure départ	Nom	Prénom
*				

Cette fenêtre permet à l'utilisateur de rechercher un trajet.

Il doit donc renseigner une date de départ, une heure minimale, une ville de départ et d'arrivée et cliquer sur « Rechercher un trajet » pour effectuer la recherche.

Si un ou des trajet(s) sont trouvés alors ils seront affichés dans la table et le bouton normalement caché « Réserver trajet » apparaît. L'utilisateur pourra réserver le trajet sélectionné dans la table.

Le bouton « Fermer » permet à l'utilisateur de fermer la fenêtre.

5.1.13 Fenêtre « Proposer un trajet »

Cette fenêtre est la même que la fenêtre vu précédemment dans l'administration : « *Gestion des trajets – Ajout/Modification d'un trajet* ».

5.1.14 Fenêtre « Paramètre utilisateur »

ID	Date départ	Heure départ	Nom	Prénom
*				

Cette fenêtre est la même que la fenêtre vu précédemment dans l'administration : « *Administration – Gestion des utilisateurs – Ajout/Modification d'un utilisateur* »

Celle-ci charge les données de l'utilisateur et lui permet de les modifier au besoin.

Un utilisateur non administrateur ne peut normalement pas modifier son nom, prénom, pseudonyme, date de naissance, sexe et type utilisateur n'est pas visible.

5.2 PRINCIPALES REQUÊTES DÉVELOPPÉES

Récupérer les données d'un utilisateur

```
SELECT Utilisateur_Nom, Utilisateur_Prenom, Utilisateur_Pseudonyme, Utilisateur_DateNaissance,  
Utilisateur_Sexe, Utilisateur_Email, Utilisateur_TelMobile, Utilisateur_Rue, U.Ville_Nom,  
U.Ville_CodePostal, Pays_Libelle, TypeUtilisateur_Id  
FROM UTILISATEUR U  
JOIN VILLE V ON V.Ville_Nom = U.Ville_Nom  
JOIN PAYS P ON P.Pays_Id = V.Pays_Id  
WHERE Utilisateur_Id = 1;
```

Modifier les données utilisateurs

```
UPDATE UTILISATEUR  
SET Utilisateur_Nom = "CHANOT", Utilisateur_Prenom = "Alexandre", Utilisateur_Pseudonyme = "a.chanot",  
Utilisateur_DateNaissance = "29/08/1993", Utilisateur_Rue = "4 Rue en France", Utilisateur_TelMobile =  
"0641254541", Utilisateur_Sexe = "Homme", Ville_Nom = "BESANCON", Ville_CodePostal = "25000",  
TypeUtilisateur_Id = "ADMIN"  
WHERE Utilisateur_Id = 1;
```

Recherche les véhicules de l'utilisateur

```
SELECT Vehicule_Id, Vehicule_Marque, Vehicule_Modele, Vehicule_NbPlaces  
FROM VEHICULE  
WHERE Utilisateur_Id = 1;
```

Ajout d'un trajet

```
INSERT INTO TRAJET (Trajet_Date, Trajet_HeureDepart, Trajet_Commentaire, Trajet_NbPassagers, Vehicule_Id,  
Utilisateur_Id)  
VALUES ("14/01/2021", "08:00:00", "Petit commentaire ici...", 4, 1, 1);
```

Ajout des étapes du trajet

```
INSERT INTO ETAPE (Trajet_Id, Ville_Nom, Ville_CodePostal, ordre)  
VALUES  
(1003, "BESANCON", "25000", 4),  
(1003, "NANCY", "54000", 3),  
(1003, "METZ", "57000", 2),  
(1003, "STRASBOURG", "67000", 1);
```

Recherche d'un trajet

```
SELECT DISTINCT Trajet_id, Trajet_Date, Trajet_HeureDepart, Utilisateur_Nom, Utilisateur_Prenom  
FROM TRAJET T  
JOIN ETAPE E ON E.Trajet_Id = T.Trajet_Id  
JOIN UTILISATEUR U ON U.Utilisateur_Id = T.Utilisateur_Id  
WHERE Trajet_Date = #14/01/2021#  
AND Trajet_HeureDepart >= #08:00:00#  
AND Ville_Depart = "BESANCON"  
AND E.Ville_Nom = "METZ"  
AND T.Utilisateur_Id != 1 ;
```

5.3 QUELQUES EXEMPLES DE CODE

5.3.1 Authentification à l'application

```
private void BTN_Login_Click(object sender, EventArgs e)
{
    //Si champ nom d'utilisateur est vide...
    if(SAI_username.Text == "")
    {
        L_Erreur.Visible = true;
        L_Erreur.Text = "Veuillez saisir un nom d'utilisateur...";
        return;
    }
    //Si champ mot de passe est vide...
    if (SAI_Password.Text == "")
    {
        L_Erreur.Visible = true;
        L_Erreur.Text = "Veuillez saisir votre mot de passe...";
        return;
    }

    //Connexion à la base de données
    SqlConnection db = new SqlConnection(Connection.conString);
    try
    {
        db.Open();
    }
    catch (SqlException)
    {
        L_Erreur.Visible = true;
        L_Erreur.Text = "Connexion au serveur impossible...";
        return;
    }

    // Vérifie le statut de la base de données
    if (db.State == System.Data.ConnectionState.Open)
    {
        // Sauvegarde la connexion à la base de données dans une global au système
        Global.dataBase = db;

        // Récupère les informations saisies
        string username = SAI_username.Text.ToString();
        string password = Security.Sha256Hash(SAI_Password.Text.ToString());

        // Vérifie si le compte de l'utilisateur est actif
        string rSQL = "SELECT * FROM UTILISATEUR WHERE Utilisateur_Pseudonyme='" + username
            + "' AND Utilisateur_HashPassword='" + password + "' AND Utilisateur_Actif = 0";
        SqlCommand cmd = new SqlCommand(rSQL, db);
        SqlDataReader data = cmd.ExecuteReader();

        // S'il y a des données alors l'utilisateur a été trouvé comme utilisateur
        // suspendu...
        if (data.HasRows)
        {
            L_Erreur.Visible = true;
            L_Erreur.Text = "Accès refusé...";
            MessageBox.Show("Accès refusé ! Votre compte a été suspendu... Veuillez
                contacter votre administrateur système.");
            this.fermetureRequete(cmd, data);
            return;
        }
        this.fermetureRequete(cmd, data);
    }
}
```

```

// Recherche l'utilisateur avec le nom d'utilisateur et le mot de passe saisi.
rSQL = "SELECT * FROM UTILISATEUR WHERE Utilisateur_Pseudonyme='" + username + "'
AND Utilisateur_HashPassword='" + password + "' AND Utilisateur_Actif = 1";
cmd = new SqlCommand(rSQL, db);
data = cmd.ExecuteReader();

// Si pas de données, l'utilisateur n'existe pas...
if (!data.HasRows) {
    L_Erreur.Visible = true;
    L_Erreur.Text = "Le nom d'utilisateur et/ou le mot de passe est incorrect...";
    this.fermetureRequete(cmd, data);
    return;
}

int i = 0;
while(data.Read()) {
    // Vérifie qu'il n'y ait pas un doublon de compte.
    // Si i > 2 alors i comptes trouvés. Erreur et arrêt de l'opération
    if (i > 0) {
        L_Erreur.Visible = true;
        L_Erreur.Text = "Veuillez contacter l'administrateur système. (Code erreur
: 0001)";
        this.fermetureRequete(cmd, data);
        return;
    }

    // Lecture des données utilisateur
    var dataRecord = (IDataRecord)data;
    // Stock les informations de l'utilisateur dans une global au système
    Global.user = new Utilisateur((int)dataRecord[0], (string)dataRecord[1],
(string)dataRecord[2], (string)dataRecord[4], (DateTime)dataRecord[6],
(string)dataRecord[7], (string)dataRecord[13], (string)dataRecord[14],
(string)dataRecord[8], (string)dataRecord[9], (string)dataRecord[10],
(DateTime)dataRecord[11], (string)dataRecord[15]);
    i++;
}

// L'utilisateur est vérifié. Log OK
Global.logOK = true;

this.fermetureRequete(cmd, data);

// Charge les données importantes au programme en cache pour travailler en mémoire
Cache.chargerCacheVilles();
Cache.chargerCachePays();
Cache.chargerCacheTypeVehicule();
Cache.chargerCacheCarburant();

// Tout est OK (Utilisateur et mot de passe OK && pas de doublon Ok : Connexion)
// Si utilisateur administrateur alors il lui est proposé de se rendre sur l'espace
d'administration
if (Global.user.getTypeUtilisateur() == "ADMIN")
{
    DialogResult dialogResult = MessageBox.Show("Voulez-vous accéder à
l'administration ?", "Message de confirmation", MessageBoxButtons.YesNo);
    if (dialogResult == DialogResult.Yes)
    {
        Form FEN_Admin = new FEN_Admin();
        this.Hide();
        FEN_Admin.ShowDialog();
        Close();
    }
    Close();
}
else

```

```

        {
            this.Hide();
            Close();
        }
    }
    else
    {
        L_Erreur.Visible = true;
        L_Erreur.Text = "Erreur de connexion au serveur.";
    }
}

```

5.3.2 Procédure de hachage de mot de passe

```

public class Security
{
    public static string Sha256Hash(string rawData)
    {
        // Créé une instance de SHA256
        using (SHA256 sha256Hash = SHA256.Create())
        {
            // Hachage de la chaine fournie en paramètre.
            byte[] bytes = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(rawData));

            // Instanciation d'un StringBuilder
            StringBuilder builder = new StringBuilder();

            // Lecture du tableau
            for (int i = 0; i < bytes.Length; i++)
            {
                // Formate chaque caractère du tableau "bytes" en deux caractères hexadécimaux
                // puis l'ajoute à la chaine "builder"
                builder.Append(bytes[i].ToString("x2"));
            }

            //Retourne la chaine construite.
            return builder.ToString();
        }
    }
}

```