Présentation finale

PT1 - G17 – Gogniat Aaron

Sommaire

- Introduction et description de la problématique
- Analyse des Objectifs
- Conception
- Implémentation
- Outils utilisés
- Evaluation
- Démonstration finale
- Conclusion

Introduction au sujet

Projet de Design Science

«Ride-Quest : a proposal for a mobile application to push people to travel using a ride sharing and public transportation system »

Application mobile de covoiturage et transport en commun

Voiture – transport public, transport public – voiture, voiture

Gratuit

Projet n°10 : système de réservation de taxi

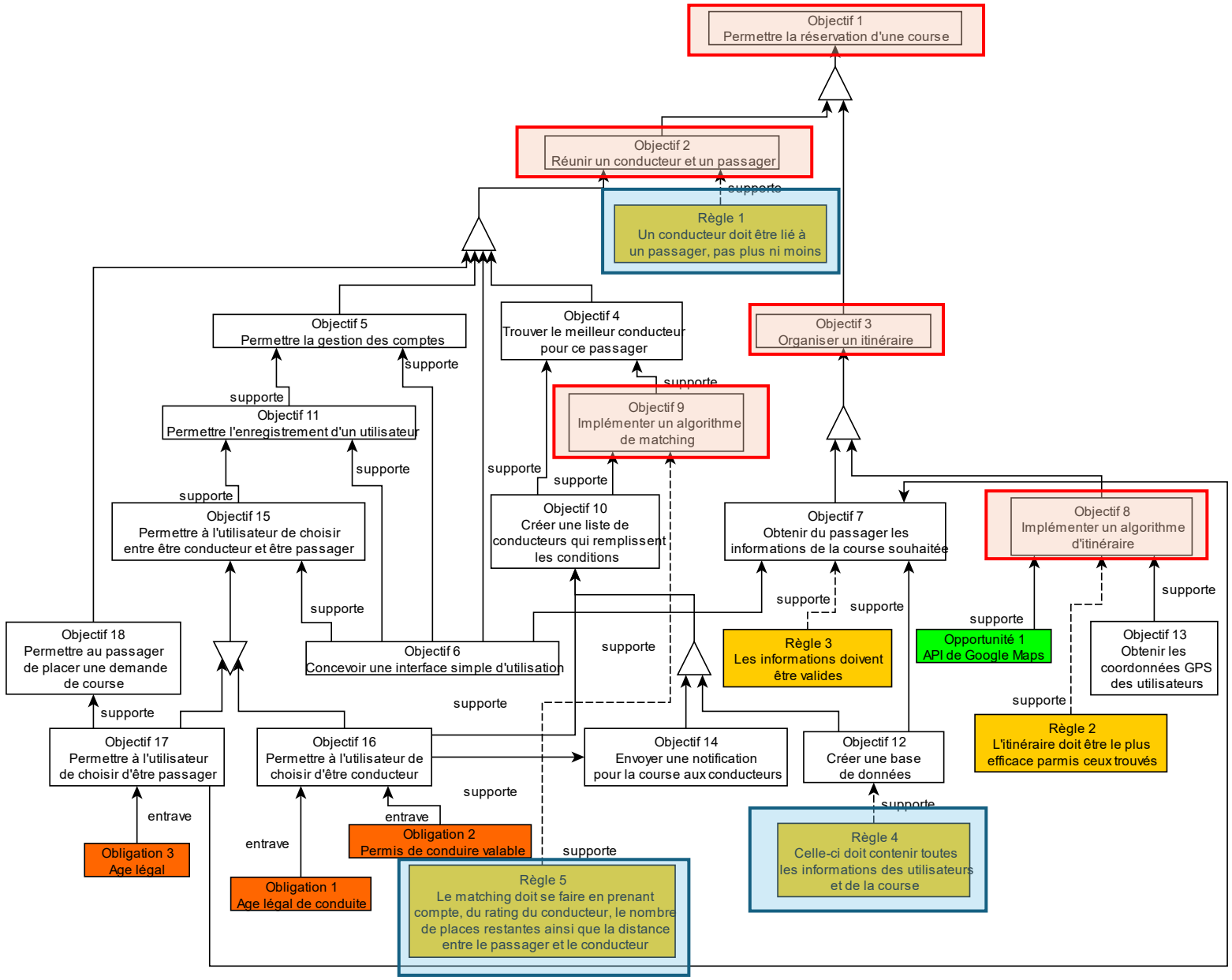
Page web, co-voiturage

Match conducteur et passager à l'aide d'un algorithme

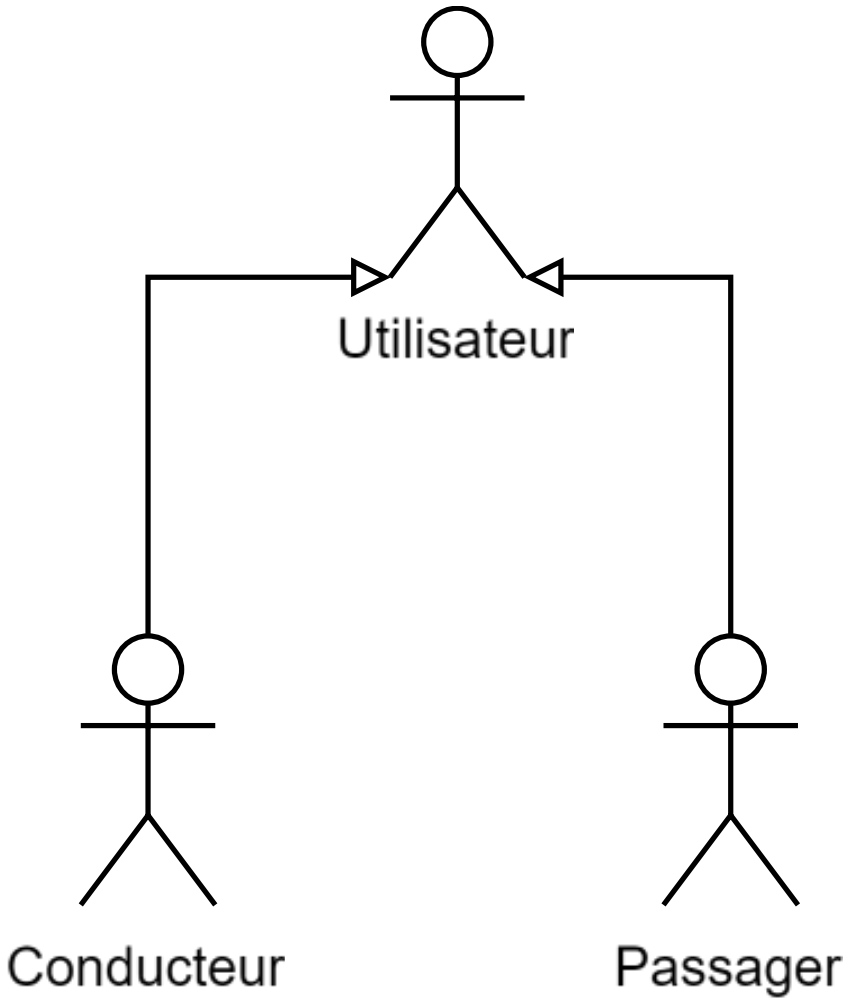
Itinéraire à l'aide des APIs

Analyse des Objectifs

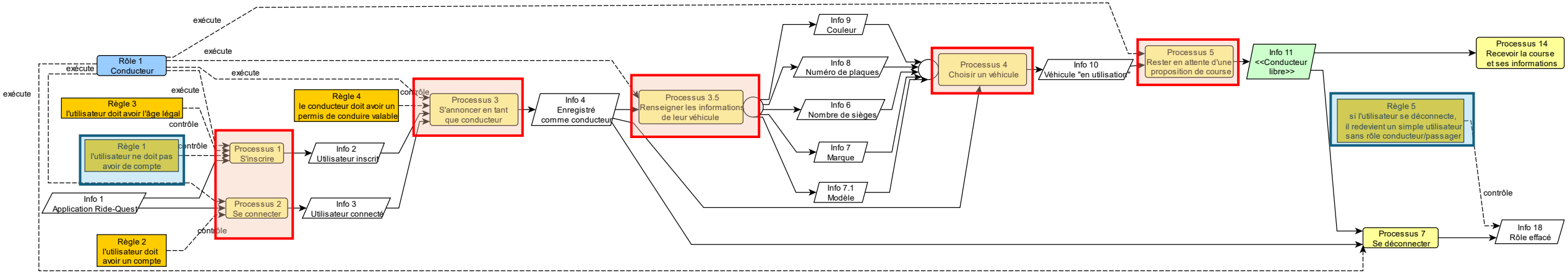
Modèle des Objectifs + Règles de gestion



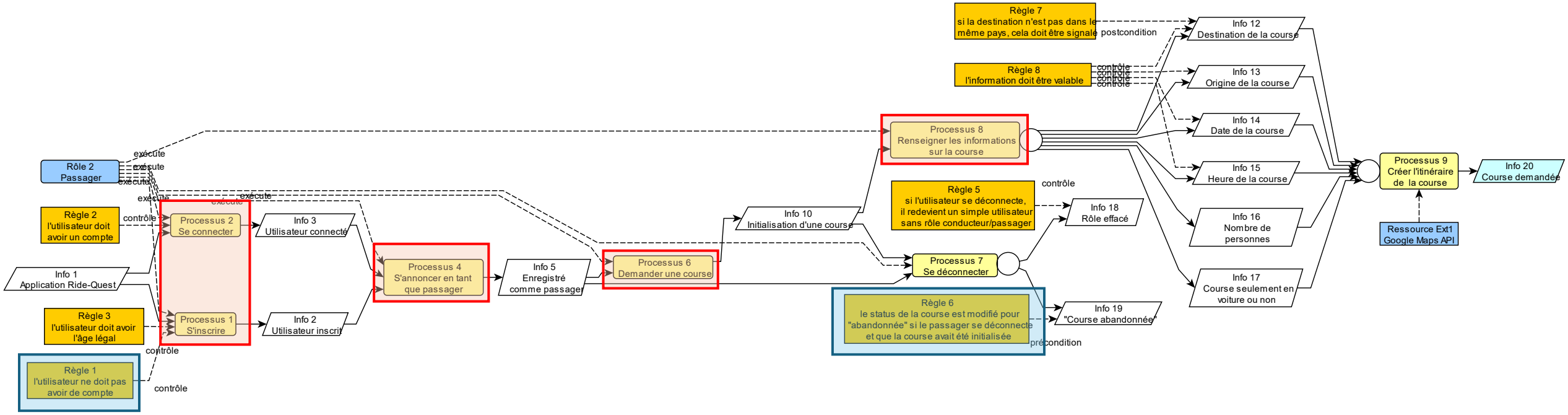
Liste des acteurs



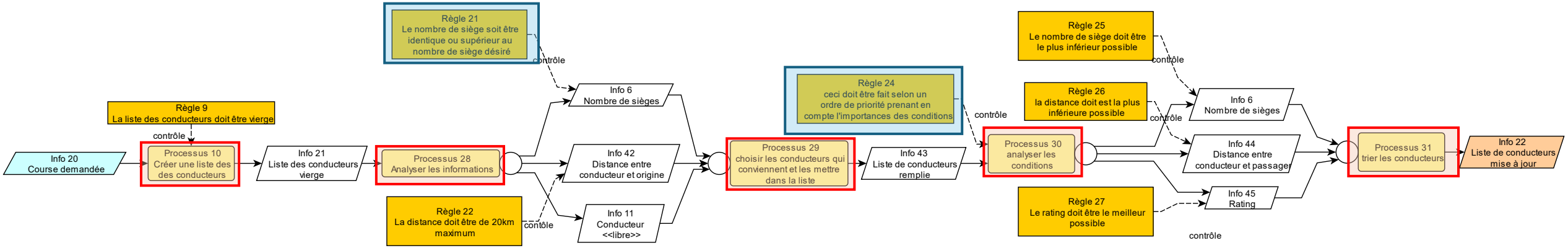
Modèle des Activités + Règles de gestion - Conducteur



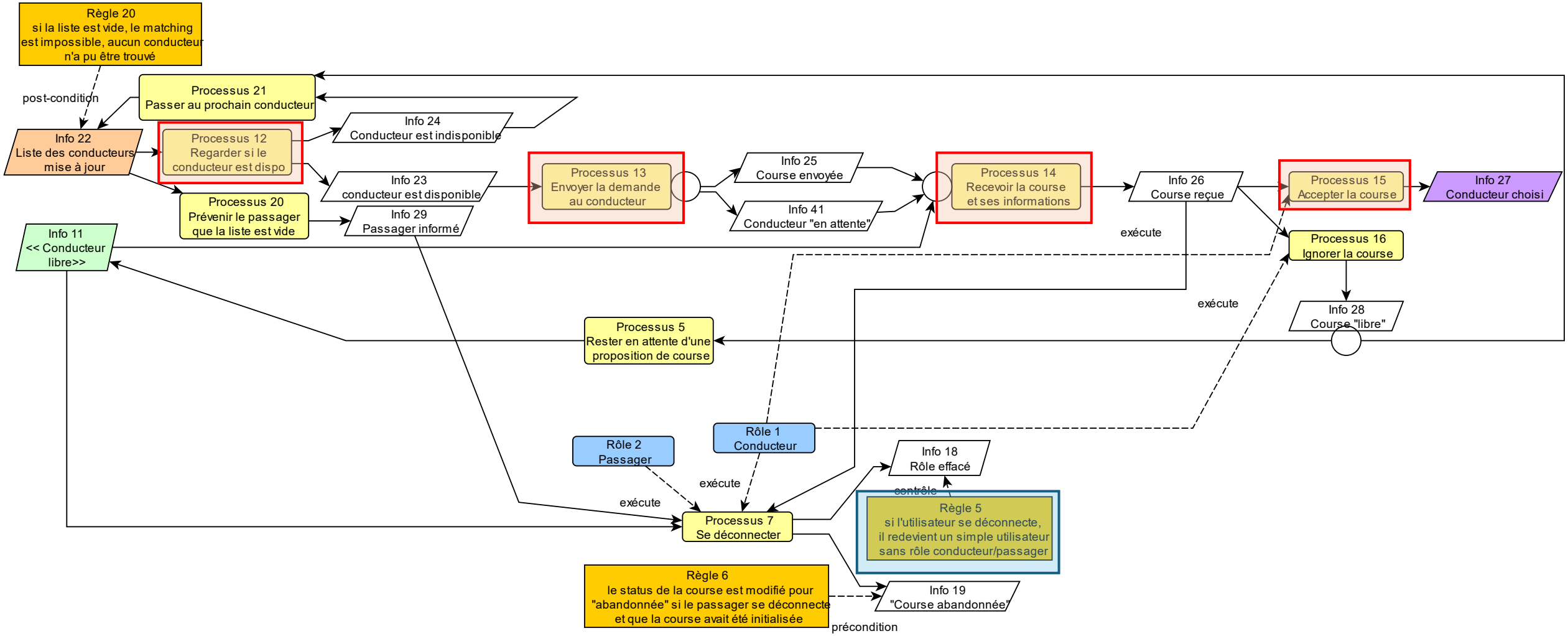
Modèle des Activités + Règles de gestion - Passager



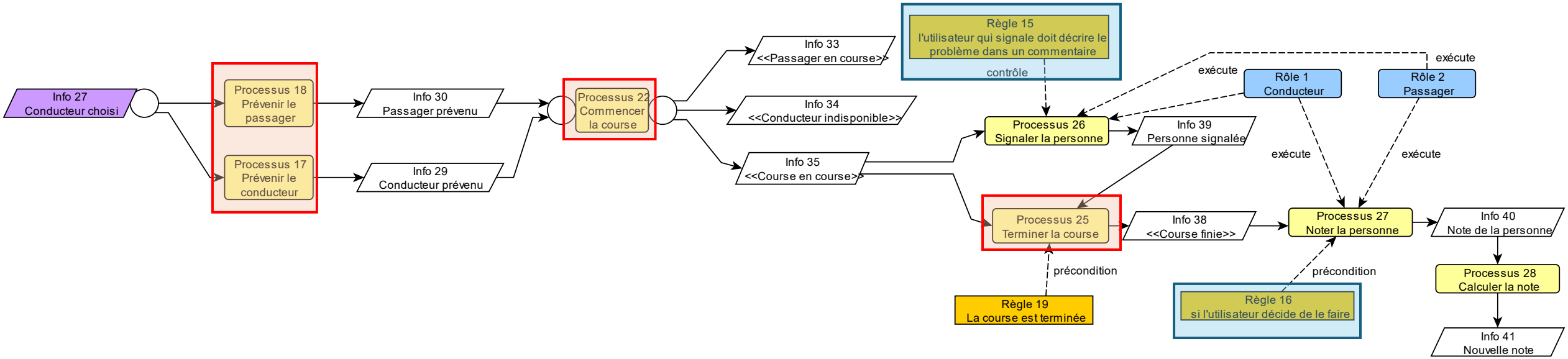
Modèle des Activités + Règles de gestion – Algorithme de sélection



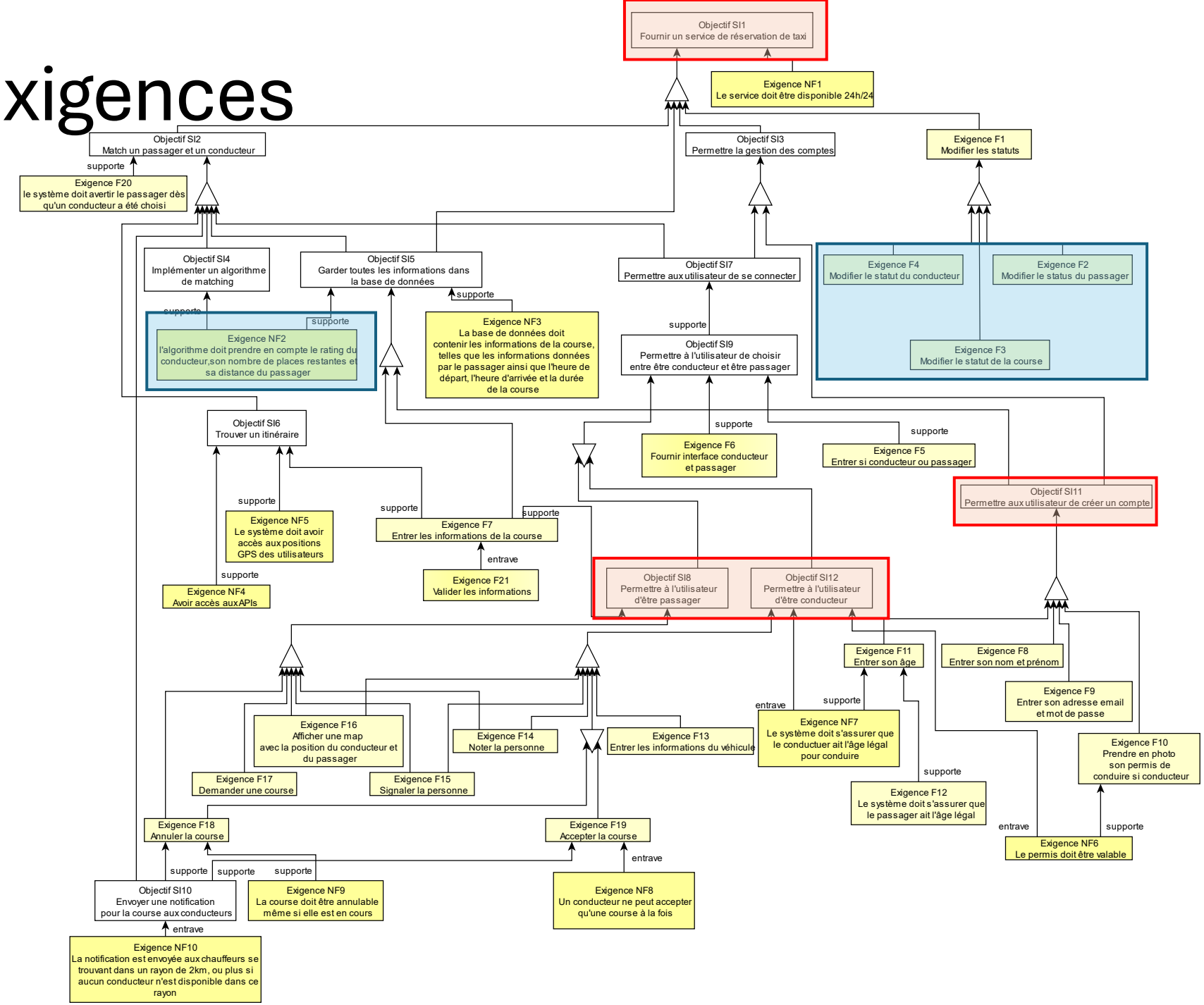
Modèle des Activités + Règles de gestion – Envoi de la course



Modèle des Activités + Règles de gestion - Course



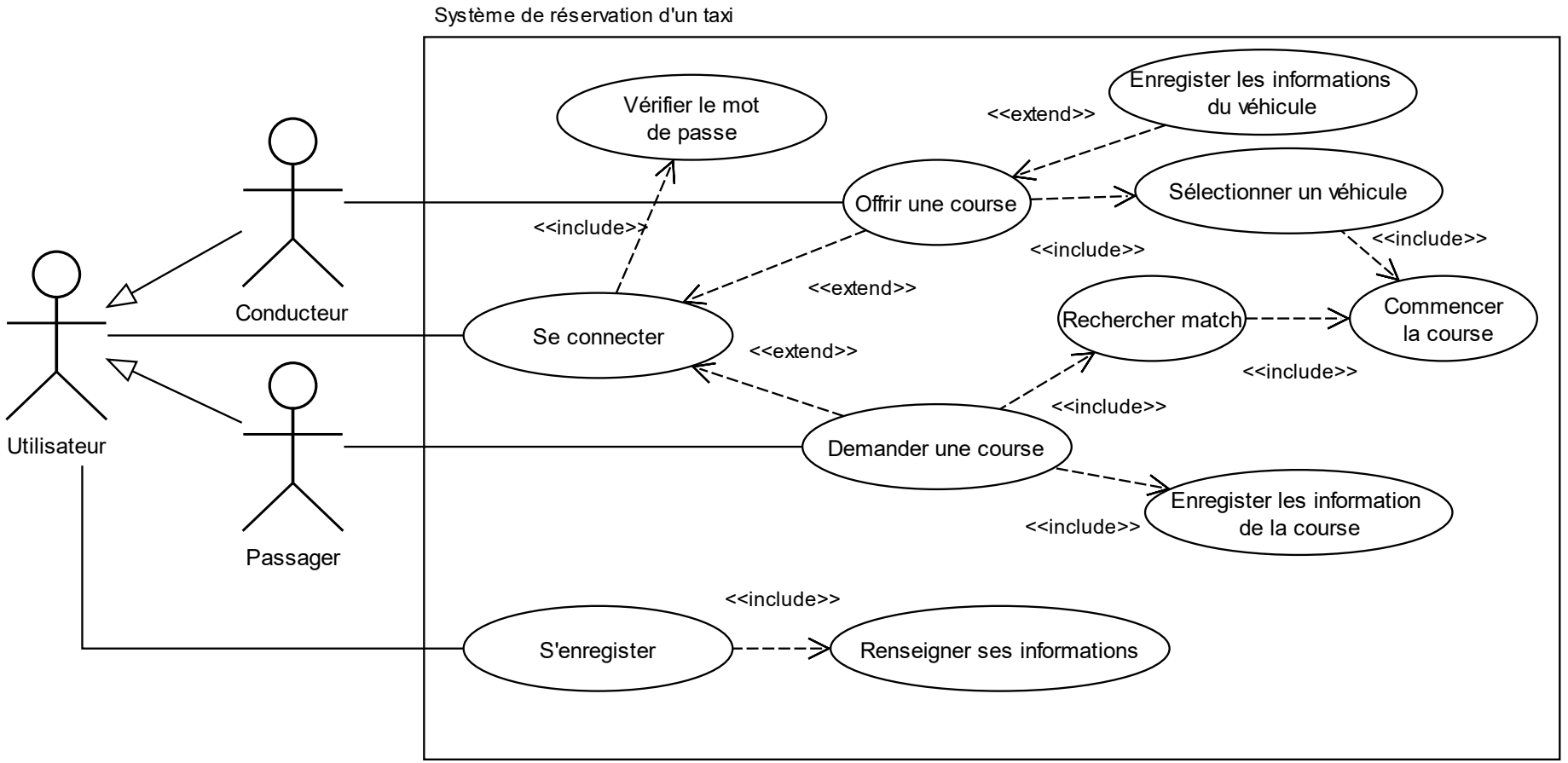
Modèle des Exigences



Conception

Modèles de la base de données

Use Case


Système de réservation d'un taxi
Vérifier le mot de passe
Enregistrer les informations du véhicule
<<extend>>
Offrir une course
Sélectionner un véhicule
<<include>>
<<include>>
<<include>>
<<extend>>
Conducteur
Se connecter
Rechercher match
Commencer la course
<<extend>>
<<include>>
Utilisateur
<<include>>
Demander une course
Enregistrer les information de la course
<<include>>
Passager
<<include>>
S'enregister
Renseigner ses informations

Use Case

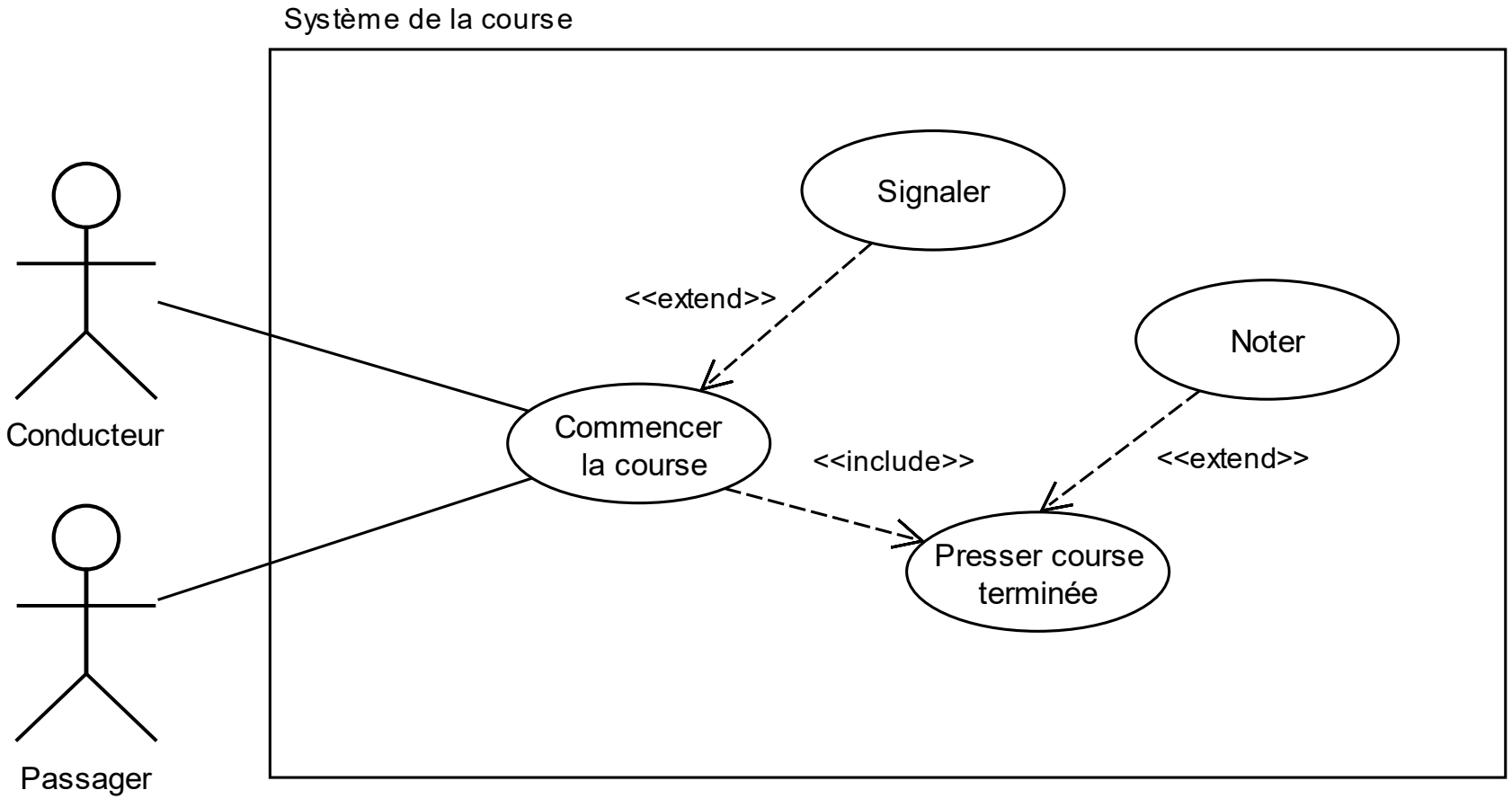

Système de la course
Signaler
<<extend>>
Noter
Conducteur
Commencer
la course
<<include>>
<<extend>>
Presser course
terminée
Passager

Schéma conceptuel

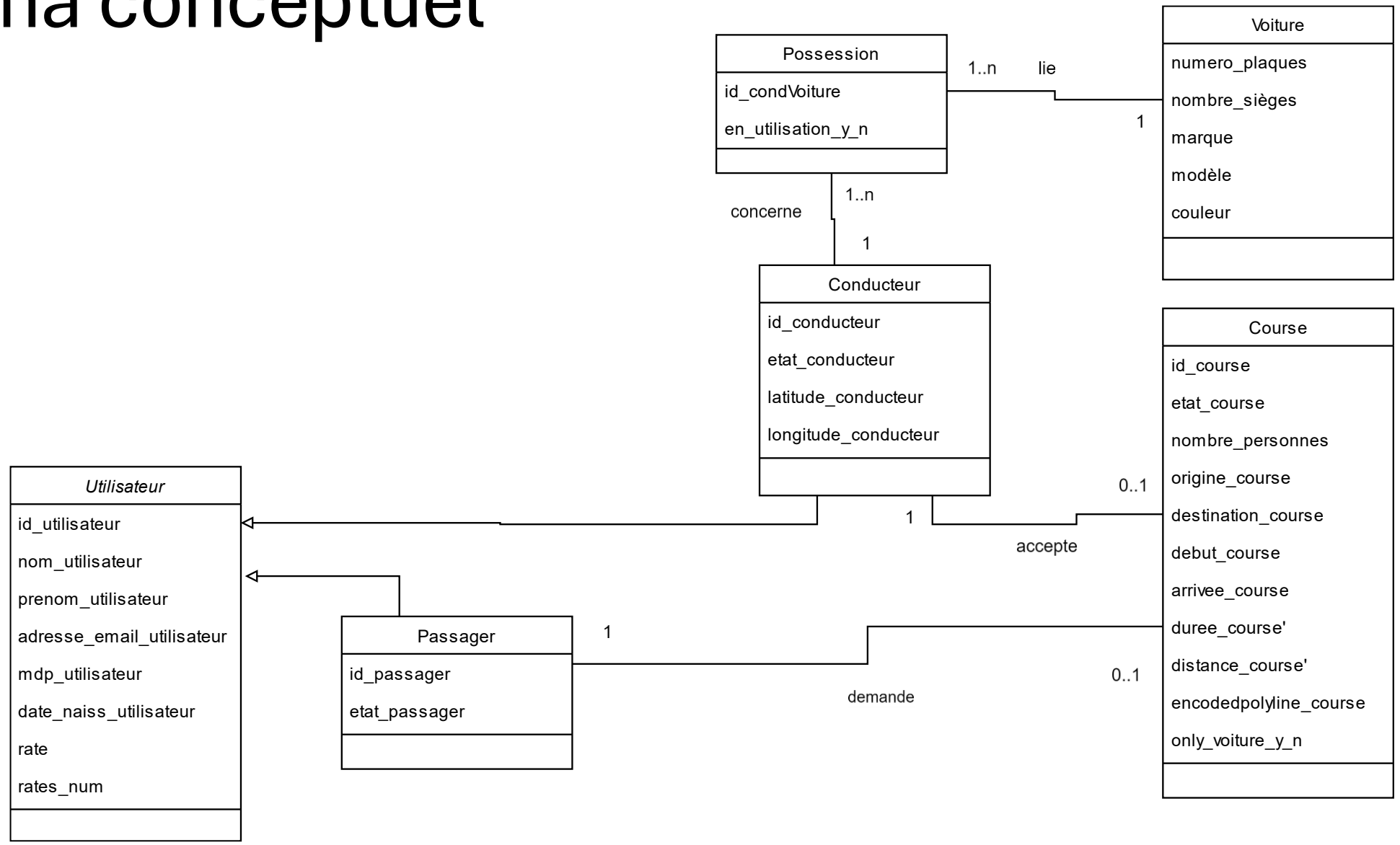
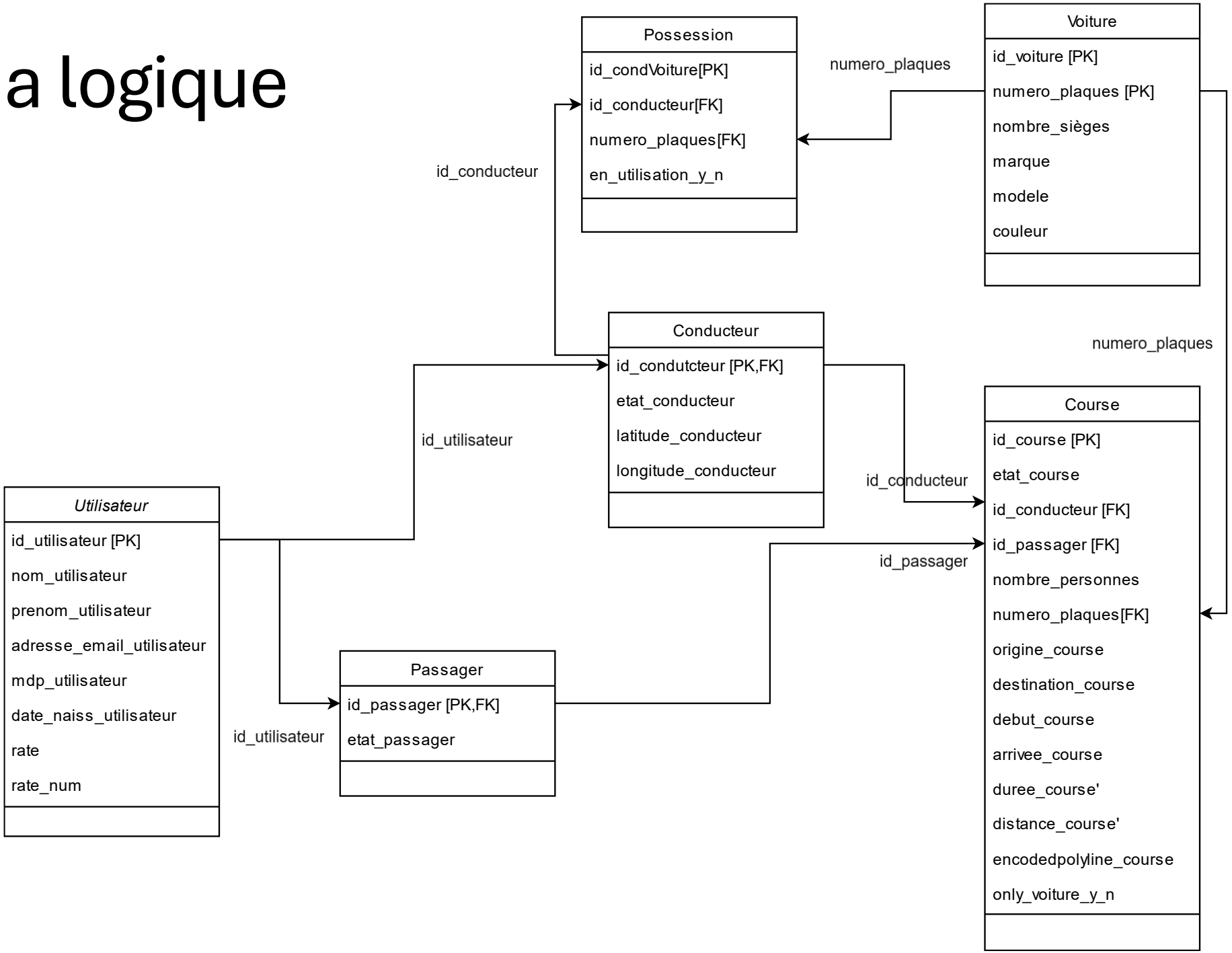


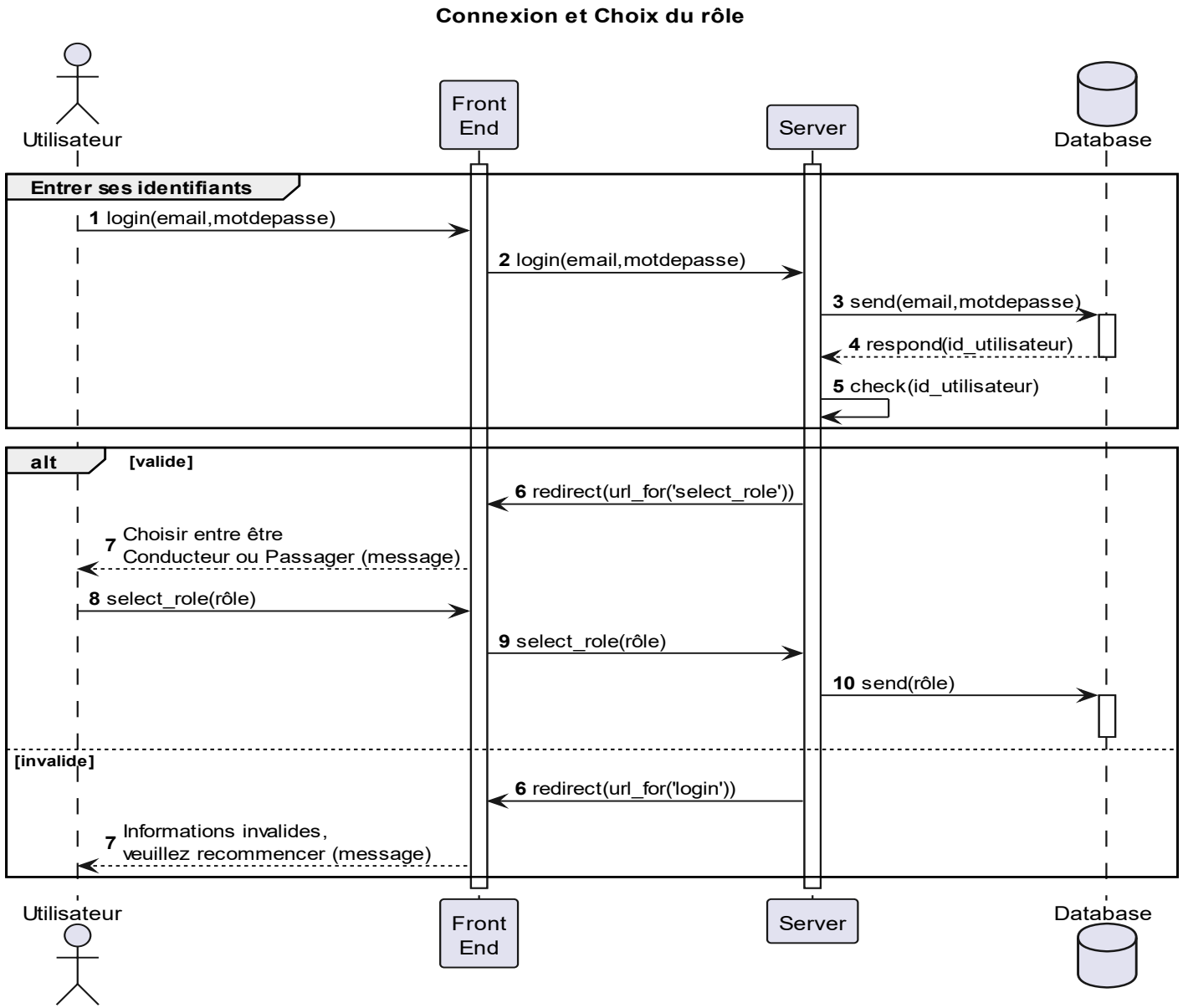
Schéma logique



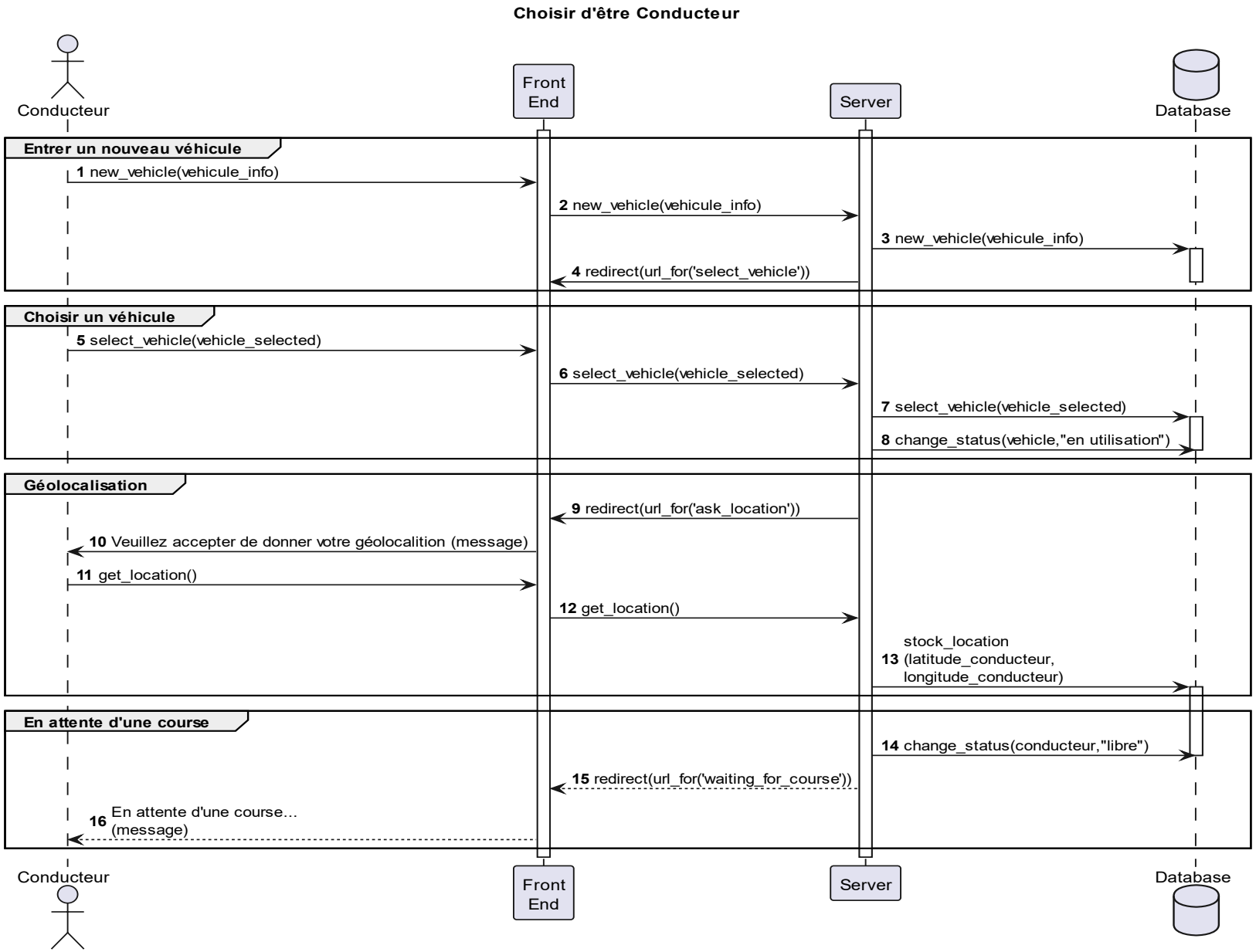
Conception

Modèles du système

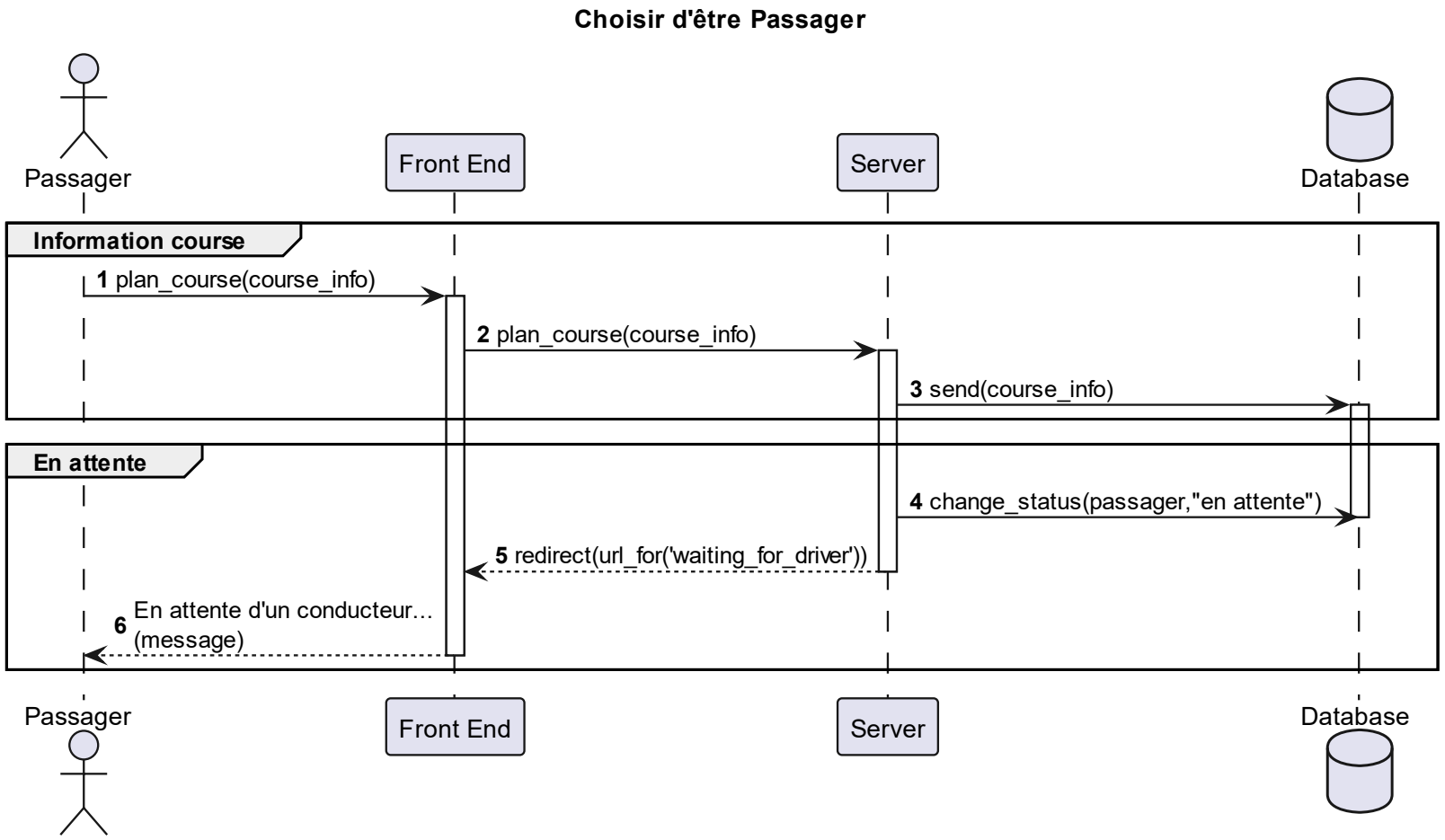
Sequence Diagram - Connexion



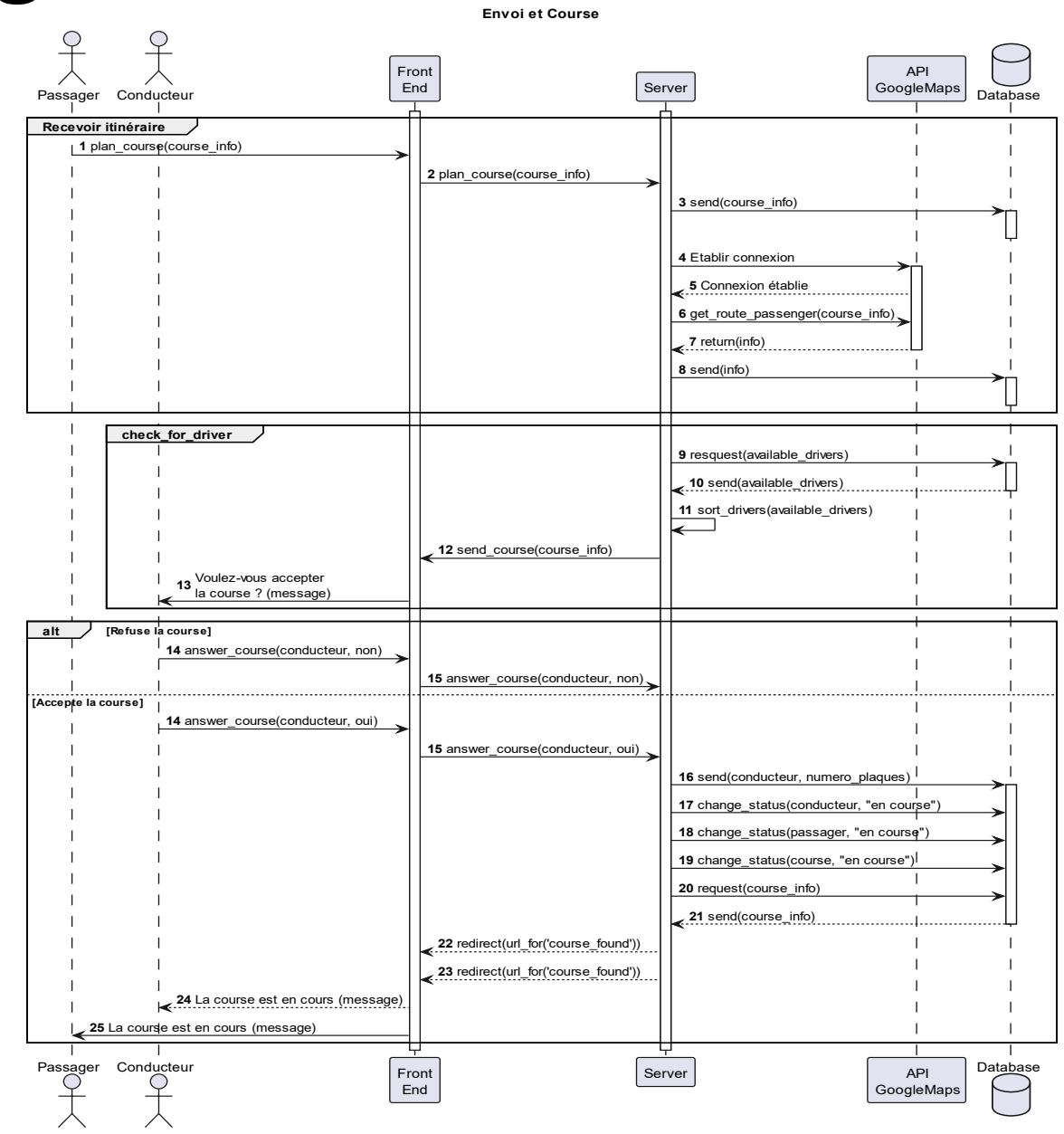
Sequence Diagram - Conducteur



Sequence Diagram - Passager



Sequence Diagram – Envoi de la course



Conception

Modèles algorithmiques

Modèle Algorithmique

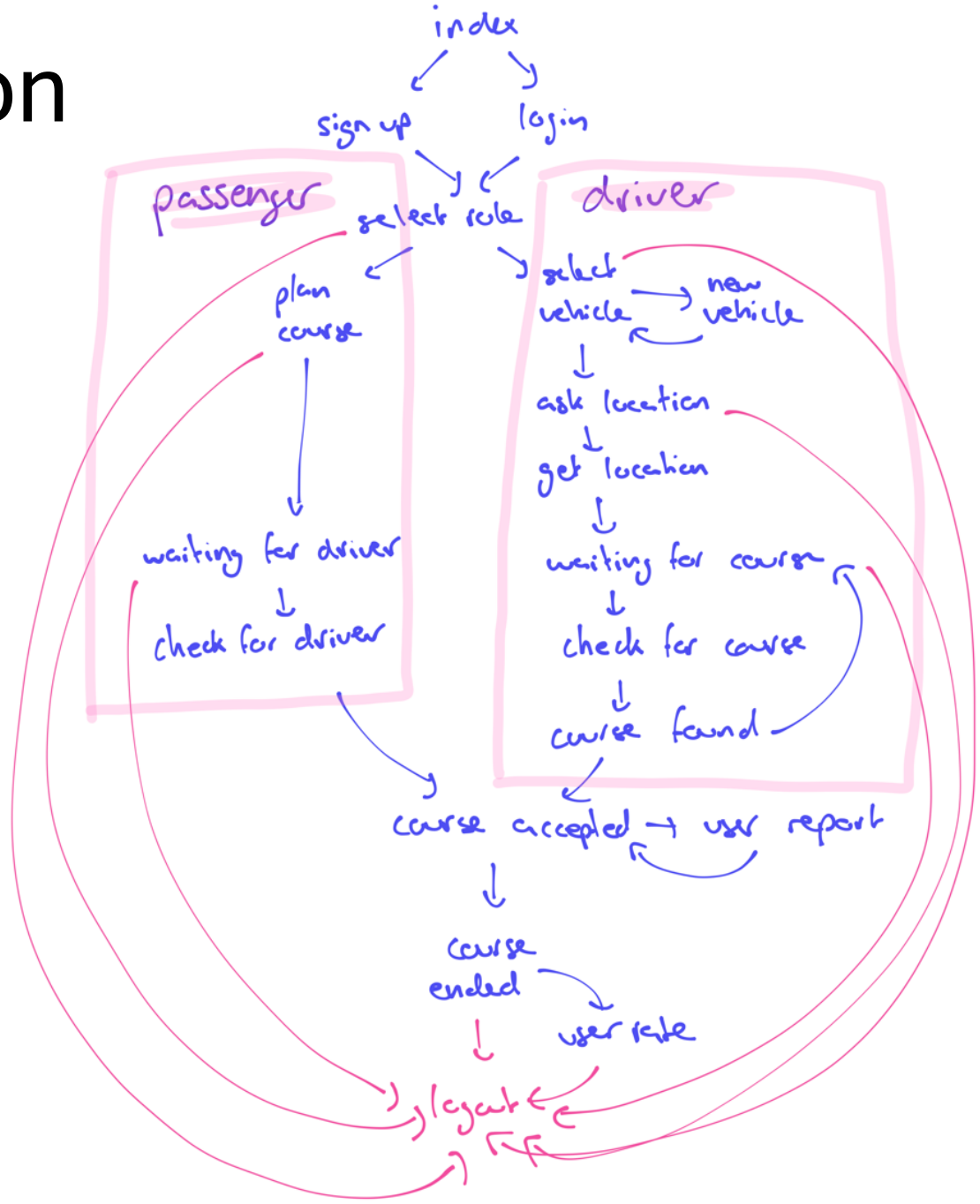
```
function sort_drivers
        list available_drivers (From database)

        if available_drivers isn't empty
                while table empty and rayon less or equal to 20km
                        for each driver in rayon + correct number or more seats asked from passenger, add to drivers_to_sort
                        if drivers_to_sort isn't empty, do :
                                sort by rating
                                sort by seat
                                sort by distance
                                sorted_drivers = drivers_to_sort
                        else, add 5 km to rayon

                if sorted_drivers is empty, return "no drivers available in a range of 20km"
                else return sorted_drivers
```

Modèle Algorithmique

```
send to the driver the course request
        get info of drivers that are free from database and add to a list

        for driver_id in driver_ids
                driver_info = []
                get and add the driver_id to driver_info
                get and add the number of seats to driver_info
                get and add the distance to driver_info
                add driver_info to available_drivers

        sorted_driver = sort_drivers(available_drivers)

        for driver in sorted_driver
                check if the driver is still free
                if free
                        update everything needed in the database
                        retries = 0
                        while retries is below 30
                                send course
                                if answer == course accepted
                                        return course accepted
                                elif answer == course free
                                        restart the loop for driver in sorted_driver
                                elif answer == course wait
                                        retries +1
                if not free do for driver in sorted_driver loop again
```

Implémentation

Implémentation



Implémentation

```python
@app.route('/signup/', methods=['GET', 'POST'])
def signup():
    if request.method == 'POST':
        # Get the information from the form
        new_email = request.form['email']
        new_password = request.form['password']
        new_firstname = request.form['firstname']
        new_lastname = request.form['lastname']
        dob_str = request.form['dob']
        dob_date = datetime.strptime(dob_str, '%Y-%m-%d')
        dob_db = dob_date.strftime('%Y-%m-%d')

        try:
            with mysql.connection.cursor() as cur:
                # Check if the email already exists => it should not exist
                cur.execute("SELECT id_utilisateur FROM utilisateur WHERE adresse_email_utilisateur = %s", [new_email])
                existing_user = cur.fetchone()

                if existing_user:
                    flash("This email address is already used, please try another one.", "error")
                    return redirect(url_for('signup'))

                # Insert the information into the utilisateur table
                cur.execute(
                    """INSERT INTO utilisateur (adresse_email_utilisateur, mdp_utilisateur, prenom_utilisateur,
                    nom_utilisateur, date_naiss_utilisateur) VALUES (%s, %s, %s, %s, %s)""",
                    [new_email, new_password, new_firstname, new_lastname, dob_db]
                )
                mysql.connection.commit()

                # Get user_id that will be used in the session and redirect to select_role
                cur.execute("SELECT id_utilisateur FROM utilisateur WHERE adresse_email_utilisateur = %s", [new_email])
                user_id = cur.fetchone()[0]
                session['user_id'] = user_id
                return redirect(url_for('select_role'))

        except MySQLdb.Error as e:
            print(f"MySQL Error: {e}")
            flash("An error occurred while signing up. Please try again.", "error")
        except Exception as e:
            print(f"Error: {e}")
            flash("An unexpected error occurred. Please try again.", "error")

    return render_template('signup.html')
```

Implémentation

templates > signup.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Ride Quest - Sign Up</title>
</head>
<body>
    <h1>Sign Up</h1>
    {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
            <ul>
            {% for category, message in messages %}
                <li class="{{ category }}">{{ message }}</li>
            {% endfor %}
            </ul>
        {% endif %}
    {% endwith %}
    <form action="/signup" method="post">
        <label for="email">Email address:</label>
        <input type="email" id="email" name="email" required><br>
        <label for="firstname">First Name:</label>
        <input type="text" id="firstname" name="firstname" required><br>
        <label for="lastname">Last Name:</label>
        <input type="text" id="lastname" name="lastname" required><br>
        <label for="dob">Date of Birth:</label>
        <input type="date" id="dob" name="dob" required><br>
        <label for="password">Password:</label>
        <input type="password" id="password" name="password" required><br>
        <button type="submit">Sign Up</button>
    </form>
</body>
</html>
```

Démonstration

Evaluation

Succès

Fonctionne

Base de données permet un stockage effectif

Interface simple

Code compréhensible et commenté

Amélioration à apporter

Ajouter les transports publics

Coordonnées GPS en temps réel

Meilleure gestion de la base de données et des données

Meilleure sécurité (vérification de l'âge, hashage de mot de passe, ...)

Créer une application mobile

Conclusion

Base robuste grâce aux modèles et schémas et à l’algorithme de tri

Les algorithmes ont permis une implémentation sans problème majeur

L’analyse des objectifs, la conception et l’implémentation ont permis de créer un système de réservation de taxi qui fonctionne

Des améliorations sont à apporter, mais le résultat nous satisfait