Pour cet ECF qui nous plonge dans la réalité de la vie d'un programmeur, la première chose qui m'a semblée la plus importante, c'est de se demander si on a bien compris tout ce qui nous était demandé par rapport au cahier des charges et c'est ce que je me suis efforcé de faire.

Ce projet est travaillé sous un environnement Linux (distribution : linux-mint), pour des raisons techniques, en particulier. Après avoir bien lu le cahier des charges et étant donné que je débute le code informatique dans la création de site web, je ne vois qu'une possibilité pour réussir ce grand projet de fin de formation, celui de choisir un framework. Il me vient tout de suite à l'esprit 'Symfony', car il est extrêmement puissant tant autant de la sécurité que celui de gérer la base de donnée avec Doctrine.

Pour la création de page, le moteur de template 'Twig' s'impose, car il est extrêmement bien assorti avec Symfony en raison de sa facilité d'utilisation, de sa sécurité et de ses performances.

Au niveau du style des pages, le framework Bootstrap sera utilisé parce qu'il nous fait gagner énormément de temps et le côté responsive des pages crées. Il sera bien sûr aidé par le CSS qui lui est indispensable quelque soit la technologie utilisée. Le Javascript sera aussi un peu utilisé pour l'interactivité avec les pages crées.

Une base mysql sera utilisé pour stocker les informations de la base de données. PHP est donc le langage utilisé pour ce projet car il est associé avec Symfony.

Ma configuration pour ce projet est donc la suivante :

1- Linux-mint 21.1
2- Php 8.1.2
3- mysql 8.0.33
4- Bootstrap 5.3.0
5- Symfony 6.2
6- Twig 3.6.0
7- JavaScript
8- HTML 5
9- CSS 3

Pour débuter ce projet, la première que je fais habituellement est celui de créer la base de données. Je suis habitué depuis plusieurs années à utiliser la méthode Merise . Je commence donc à créer le MCD (modèle conceptuel de donnée) , dont voici sa représentation qui correspond à la vision que j'ai par rapport à cet étude de cas.

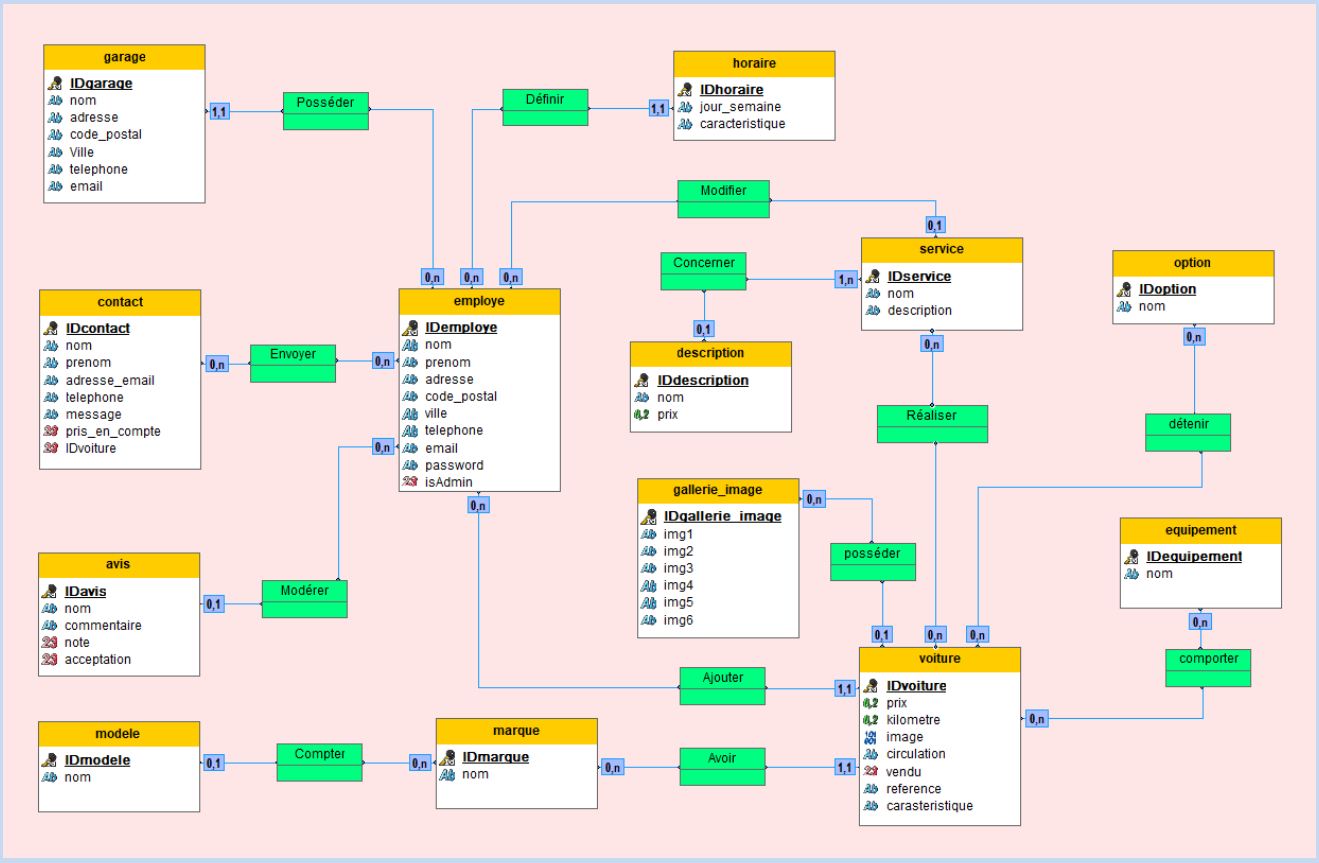
A noter que je ne voulais pas créer une table 'Administrateur' et une table 'Employé'. Comme il se doit, une base de donnée est performante si on fait tout pour éviter la redondance d'information.

J'ai donc décider d'inclure l'administrateur dans la table 'employé' en ajoutant une propriété 'IsAdmin' qui est un booléen et qui distinguera l'administrateur de l'employé.

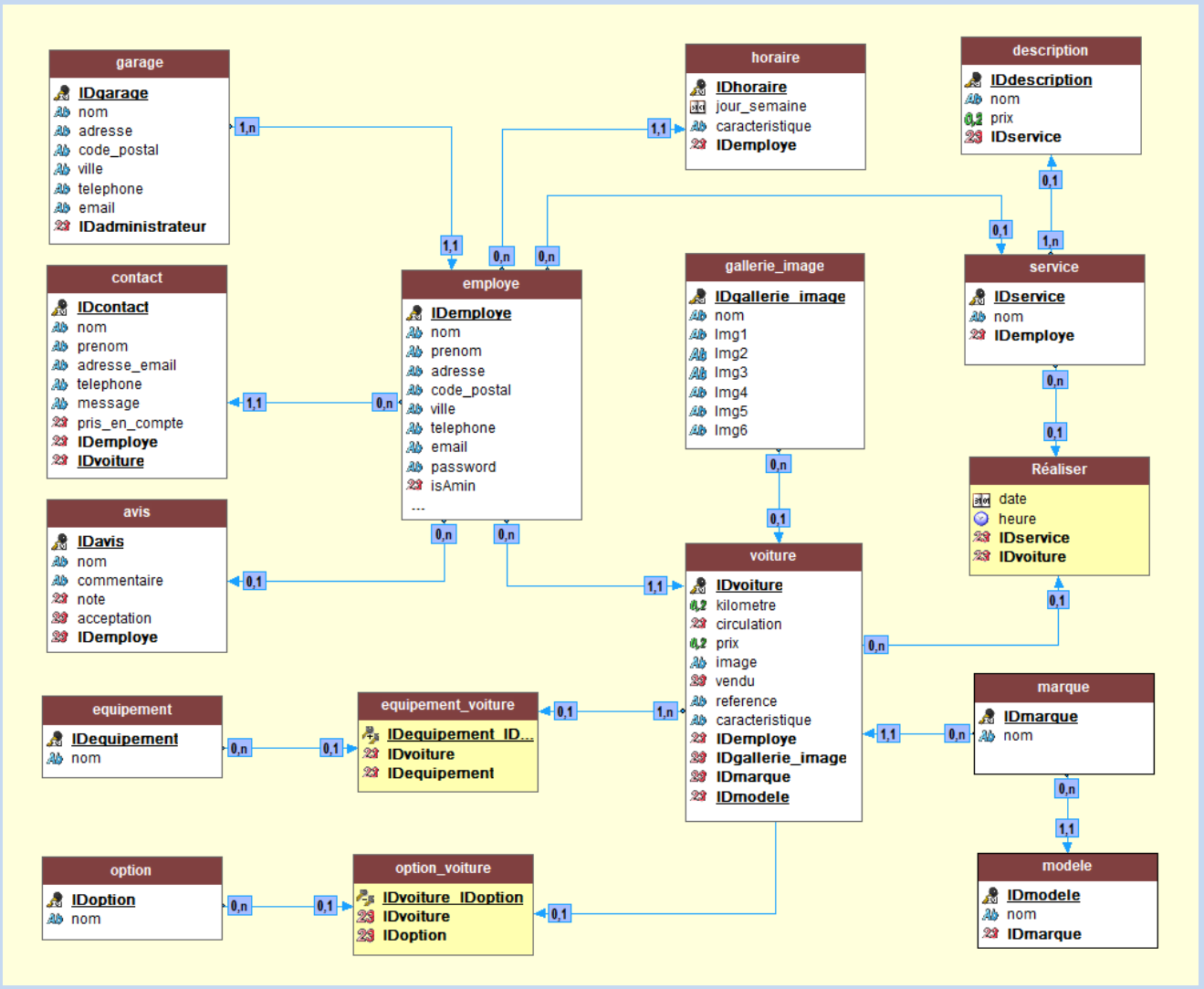
Je tiens à signaler que j'ai utilisé Windev (version 20) pour la création de la base de donnée, car c'est un logiciel que j'utilise depuis quelques années et que je trouve très performant pour ce type de travail. Je n'ai donc pas à chercher un autre logiciel pour la vision des tables et l'écriture des rubriques car cela se fait aisément.

La difficulté, si l'on peut dire, c'est de bien penser le mcd, car le mld, pour la suite de la modélisation ,va être traduit selon celui-ci en y ajoutant des clefs étrangères et s'il y a des erreurs de conceptions dès le départ, cela va causer des gros problèmes par la suite pour récupérer les informations de la base de donnée. En effet, il faudra traduire l'existence du MLD en langage sql et cette tache sera réservé à l'ORM 'Doctrine'.

Voici donc la conception du MCD.



On transforme ce mcd en mld (modèle logique de données)



A partir de ce schéma, on pourra construire nos tables et rubriques (ou entités et propriétés en UML) avec le langage SQL, si l'on veut utiliser une base de donnée mysql ou mariaDB.

Pour installer toutes les outils indispensables pour ce projet dans un système Linux, voici la syntaxe qui doit être tapé en ligne de commande avec le terminal de linux

```
    //Installation de PHP
1-  sudo apt update
2-  sudo apt install php php-cli php-fpm php-mysql php-curl php-mbstring php-xml php-
    zip
3-
    //Installation de mysql
4-  sudo apt install mysql-server

    //Installation du serveur apache
5-  sudo apt install apache2

    //Installation de Composer
6-  sudo apt install curl php-cli php-mbstring
7-  cd ~
8-  curl -sS https://getcomposer.org/installer -o composer-setup.php
9-  sudo php composer-setup.php --install-dir=/usr/local/bin --filename=composer
10-
    //Installation de npm
11- sudo apt install nodejs npm

    //Installation de Symfony cli → terminal pour executer des instructions à Symfony
12- wget https://get.symfony.com/cli/installer -O - | bash
13- sudo mv ~/.symfony/bin/symfony /usr/local/bin/symfony
```

Après cela, on vérifie si symfony a toutes les dépendances pour fonctionner correctement et on ajuste s'il le faut.

```
symfony check:requirements
```

On peut maintenant créer le programme de ce projet que j'ai nommé 'garage'

```
symfony new garage –full
```

Doctrine, dans symfony, peut créer maintenant la base de donnée. Une condition pour réaliser cela, c'est de bien paramétrer une ligne dans le fichier '.env', situé à la racine du projet, qui selon ma configuration et ma version de mysql donne ceci :

```
DATABASE_URL="mysql://admin@127.0.0.1:3306/garage?serverVersion=8.0.33-0ubuntu0.22.04.2"
```

Maintenant, la création des entités et des propriétés deviennent possible, après avoir lancé la syntaxe suivante dans le terminal :

```
Symfony serve -d
```

Pour ce devoir, il était demandé de créer un administrateur uniquement en ligne de commande. Avant cela, il faut créer la première table du projet dans Doctrine, qui sera nommée 'employe'. Ensuite, pour une authentification sécurisée, il faudra utiliser la commande très particulière de Symfony :

```
symfony console make:auth
```

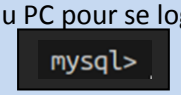
Cette commande générera automatiquement tout ce qui est nécessaire, notamment :

- Génération des fichiers d'authentification
- Configuration de la sécurité
- Cryptage des mots de passe
- Gestion des routes (Le controller)
- Gestion des vues (Le template)

Pour la création d'un administrateur, il faudra créer un mot de passe qui sera crypté et hasché pour des raisons de sécurité. Tout se fait dans le terminal de VScode.

Par défaut, Symfony utilise l'algorithme bcrypt avec un coût de 13.
Pour le projet d'évaluation du garage Parrot, le mot de passe a été bien simplifié --> 'parrot' .
Malgré sa simplicité, il sera illisible.
On insérera ce mot de passe dans une requête sql lors de la création de cet administrateur. Voici donc comment j'ai créé l'administrateur : Mr Parrot

//Tout se fait maintenant dans un terminal (celui de VScode par exemple) et en super utilisateur
`htpasswd -nbBC 13 USER parrot` // donne une suite de lettre et chiffre qui montre le cryptage
$2y$13$l1DxGvlIie9wH2uphUBtse84uxUPGc0m7MP2d3gP064Hfh8CjsMW2 (mot de passe 'parrot' : crypté)

`sudo su` //On rentre le mot de passe de l'administrateur du PC pour se loguer et ainsi avoir accès au langage 'sql'.
`mysql` // on est maintenant en invite de commande mysql 

`USE` garage; // Utilisation de la base de donnée 'garage'

//Requête d'insertion dans la table 'employe'

```
INSERT INTO employe (nom, prenom, email, roles, password, is_admin, adresse,
code_postal, ville, telephone) VALUES ('Vincent', 'Parrot', 'vincent@free.fr',
'{"role": "ROLE_ADMIN"}',
'$2y$13$l1DxGvlIie9wH2uphUBtse84uxUPGc0m7MP2d3gP064Hfh8CjsMW2', 1, '15 rue du garage',
'31000', 'Toulouse', '02.54.36.25.21');
```

L'administrateur est donc créé. Il est le premier enregistrement de la base de donnée :

id int	email varchar(180)	roles json	password varchar(255)	nom varchar	pren varcha	adresse varchar(150)	cod int	ville varchar(	telephone varchar(25)	is_admin tinyint
1	vincent@free.fr	{"role": "ROLE_ADMIN"}	$2y$13$l1DxGvlIie9wH2uphUBtse84uxUPGc0m7MP2d3gP064Hfh8CjsMW2	Vincent	Parrot	15 rue du garage	31000	Toulouse	02.54.36.25.21	1