



TPI

# Gestion de parc automobile d'entreprise

Eliott Deriaz

Chef de projet : M. Gaël Sonney

Expert 1 : M. Roger Malherbe

Expert 2 : M. Daniel Berney

Du 30.04.2025 au 26.05.2025

## TABLE DES MATIÈRES

Introduction.....	5
Matériel à disposition .....	5
Prérequis.....	5
Livrables .....	5
Analyse .....	5
Objectifs.....	5
Administrateurs.....	5
Employés .....	5
Choix de technologie .....	6
Planification .....	6
Horaire de travail .....	6
Planification initiale.....	6
Conception.....	8
Schéma de navigation .....	8
Base de données .....	9
Authentification .....	9
Listes.....	10
Ajout.....	11
Modification .....	12
Détail .....	14
Validation d’une expertise .....	15
Stratégie de test.....	15
Risques techniques .....	17
Réalisation .....	17
Connexion à la base de données .....	17
Authentification et gestion des rôles.....	17
Gestion des véhicules .....	20
Liste .....	20

Ajout.....	20
Modification.....	21
Détail.....	21
Suppression.....	22
Contrôle du type de fichier .....	23
Gestion des expertises.....	23
Liste .....	24
Ajout.....	24
Modification .....	25
Suppression.....	26
Validation d'une expertise .....	26
Implémentation de l'api .....	27
Connexion à l'api.....	27
Sélection de la marque .....	28
Sélection du modèle .....	29
Tableau de bord.....	30
Déploiement de l'application .....	30
Description des tests effectués.....	31
Erreurs restantes .....	32
Conclusion .....	33
Bilan des fonctionnalités.....	33
Bilan personnel .....	33
Bilan planification .....	34
Améliorations possibles.....	34
Annexes .....	35
Journal de travail .....	35
Résumé du rapport.....	36
Situation initiale .....	36
Implémentation .....	36
Conclusion .....	36

Repository GitHub .....	37
Table des illustrations .....	37
Glossaire .....	38
API .....	38
ASP.NET .....	38
ASP.NET Identity .....	38
CarData API .....	38
Code First .....	38
DataTables.....	38
Entity Framework.....	38
MVC.....	38
Nginx .....	38
Reverse proxy.....	38
SweetAlert2.....	38
Seeding.....	38
Sources.....	38

## INTRODUCTION

Ce projet s'inscrit dans le cadre du TPI et se déroule du 30 avril 2025 au 26 mai 2025. Il vise à créer une application web pour la gestion d'un parc automobile d'entreprise. Puis de déployer cette application sur l'infrastructure de l'Etml.

Le but est à la fois d'appliquer mes connaissances en développement d'application ainsi que mes connaissances système pour le déploiement d'application web.

## MATÉRIEL À DISPOSITION

- Un PC en configuration standard ETML avec un accès à internet
- Visual Studio avec ASP.NET installé
- Un outil de gestion de version (GitHub)
- Suite Microsoft 365 pour la documentation

## PRÉREQUIS

- Connaître les notions de la programmation orientée objet
- Connaître les notions de développement web MVC
- Savoir utiliser Visual Studio
- Savoir coder en C#
- Maîtriser les Modules ICT suivants : 104, 105, 120, 133, 153, 226, 326, 403, 404 et 411

## LIVRABLES

- Rapport de projet
- Journal de travail
- Code source
- Scripts de base de données

## ANALYSE

Cette partie décrit toute la réflexion préliminaire à la réalisation afin d'avoir un plan pour le développement.

## OBJECTIFS

Le but de ce projet est de développer une application web pour la gestion d'un parc automobile d'entreprise. Il y a deux types d'utilisateur, les administrateurs et les employés.

### ADMINISTRATEURS

Les administrateurs doivent pouvoir gérer les véhicules, c'est-à-dire ajouter, supprimer, modifier, consulter la liste de tous les véhicules ainsi que consulter le détail des véhicules.

Ils doivent aussi pouvoir gérer les expertises, ajouter, supprimer, modifier et accéder au détail des expertises.

### EMPLOYÉS

Les employés auront accès uniquement à la liste et au détail des véhicules ainsi qu'aux expertises qui leur sont attribuées. Ils auront aussi le droit de changer le statut d'une expertise.

## CHOIX DE TECHNOLOGIE

Pour réaliser le projet, le framework donné est ASP.NET, mais je vais préciser ASP.NET 9 pour une prise en charge à long terme.

Pour réaliser la connexion à la base de données, j'utilise Entity Framework qui permet de faire du Code First en ASP.NET. Il est présent de base dans une application ASP.NET donc entièrement compatible et j'ai déjà de l'expérience avec.

Pour la gestion de l'authentification ainsi que la gestion des rôles et des autorisations, j'utilise Identity. Il donne aussi accès à une interface utilisateur basique pour la gestion des comptes qui me permet d'avoir une base solide pour effectuer les modifications de style afin que cela corresponde au style de l'application.

Pour l'affichage des listes, j'utilise Datatables qui est une librairie JS open source pour la génération de tables complexes. Il permet d'avoir automatiquement une pagination et de filtrer les colonnes. Il y a aussi beaucoup de documentation utilisateurs sur le forum de l'extension.

Pour l'affichage des alertes, j'utilise SweetAlert2 qui permet de faire des alertes personnalisables en JS.

Enfin, pour l'API que je vais utiliser pour récupérer les marques et les modèles de véhicule lors de l'ajout et la modification, c'est Car Data que j'ai trouvé sur Rapid API. Elle est la seule API gratuite qui est capable de récupérer des modèles en fonction du manufacturier, du nom de modèle et du type. Son seul défaut est que les modèles les plus récents répertoriés sont de 2020.

## PLANIFICATION

Le projet se déroule du 30.04.2024 au 26.05.2025.

### HORAIRE DE TRAVAIL

En raison des cours que je dois suivre en dehors du projet, je ne peux pas travailler tous les jours. Ci-dessous, l'horaire attribué au projet.

	Lundi	Mardi	Mercredi	Jeudi	Vendredi
Matin	08h00-11h25	-	08h00-12h15	-	08h00-12h15
Après-midi	12h20-14h45	-	13h10-16h35	13h10-16h35	13h10-16h35

### PLANIFICATION INITIALE

Pour la planification, j'utilise la méthode en cascade avec un Gantt, car elle permet de définir les dates limites pour chacune des fonctionnalités.

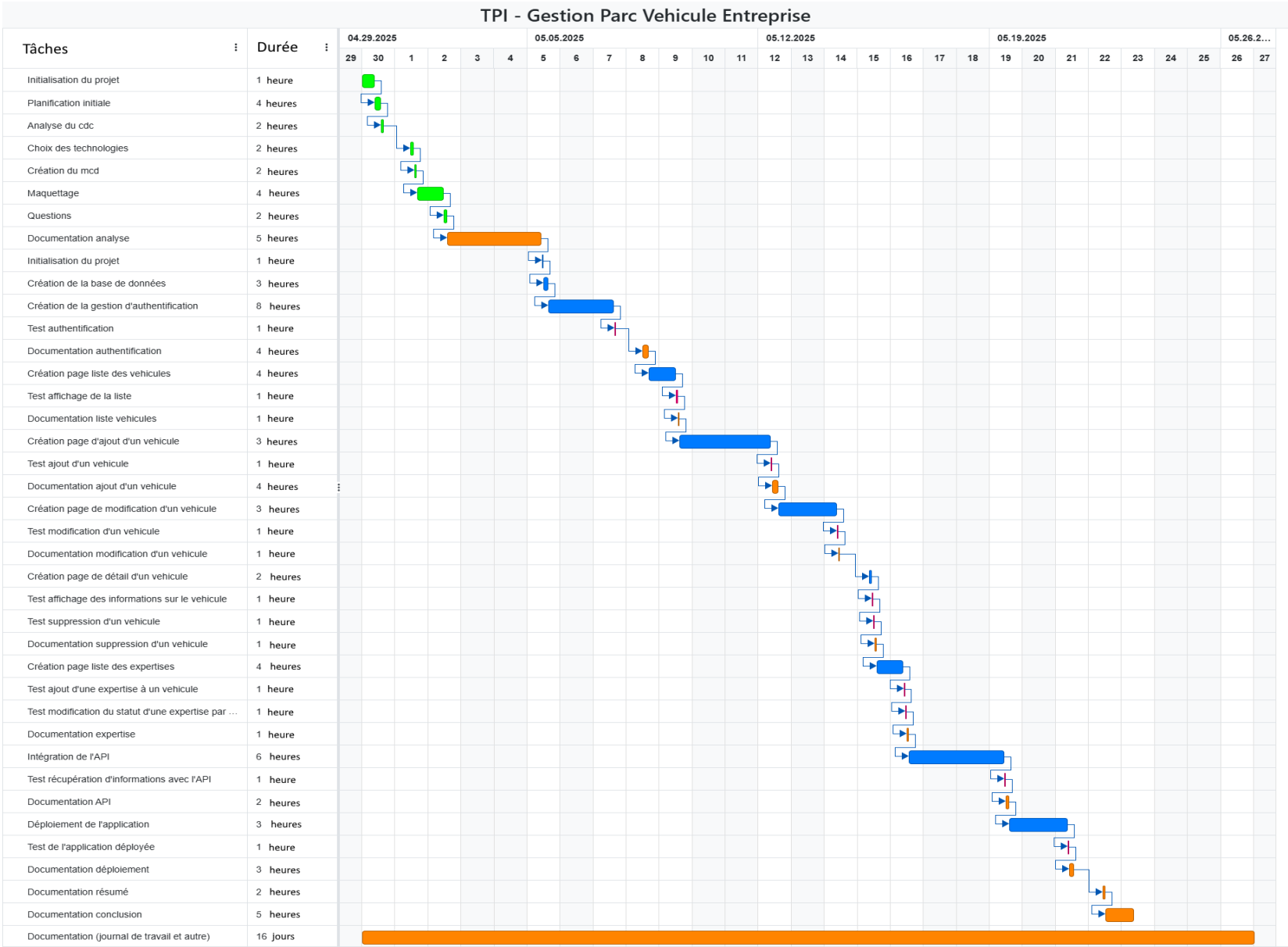


FIGURE 1 PLANIFICATION INITIALE





## BASE DE DONNÉES

Ici le schéma (MLD) montre comment les différentes tables interagissent entre elles

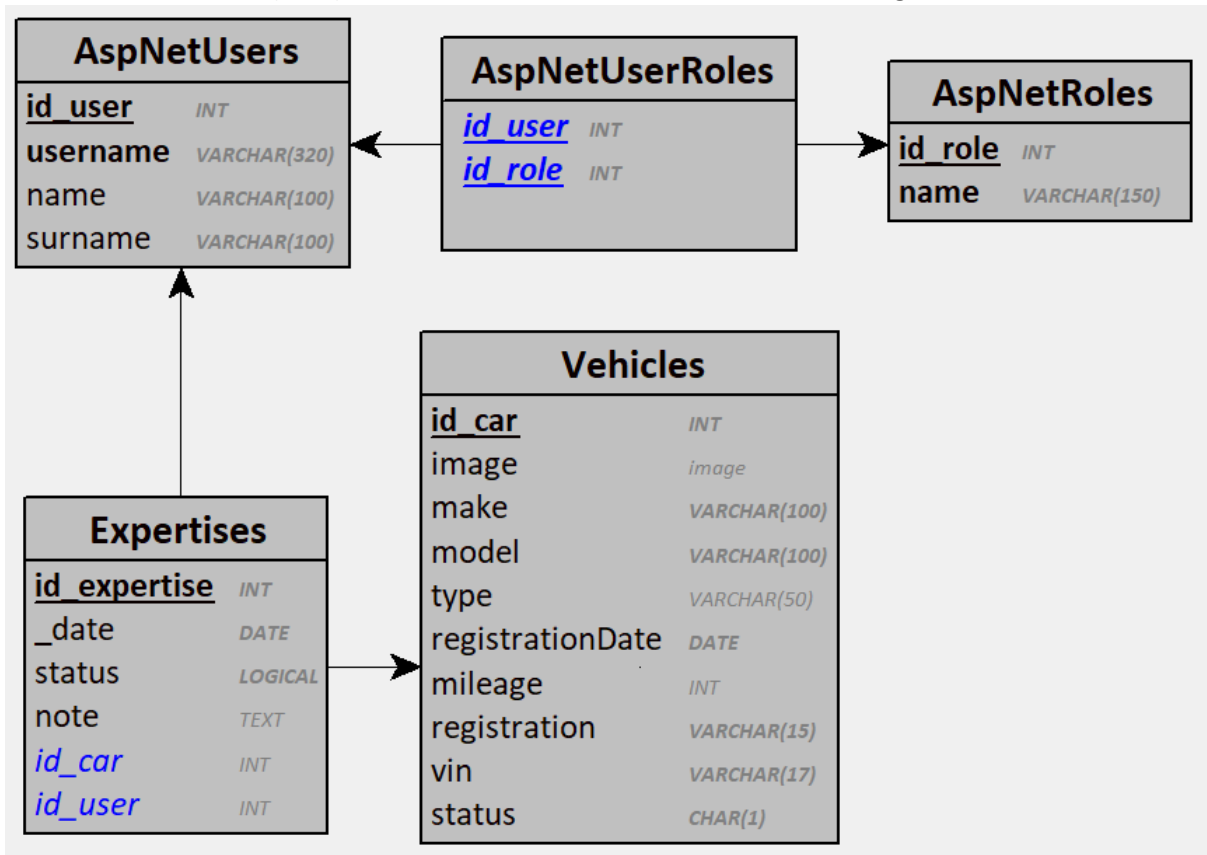


FIGURE 3 SCHÉMA DE BASE DE DONNÉES MLD

## AUTHENTIFICATION

La maquette suivante montre à quoi devrait ressembler la page de connexion à l'application.

Gestion du parc automobile

The mockup shows a login form titled "Connexion". It contains two input fields: "e-mail" and "mot de passe". Below the fields is a blue button labeled "Connexion".

FIGURE 4 MAQUETTE PAGE AUTHENTIFICATION

LISTES

Ici les maquettes servent à illustrer le contenu des deux pages listes (véhicule et expertise), elles serviront de modèle lors de la réalisation.

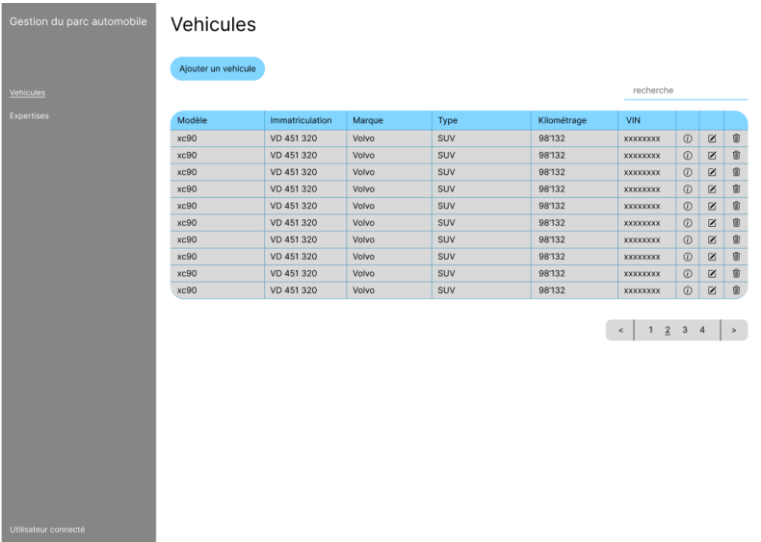


FIGURE 5 MAQUETTE PAGE LISTE DE VÉHICULES

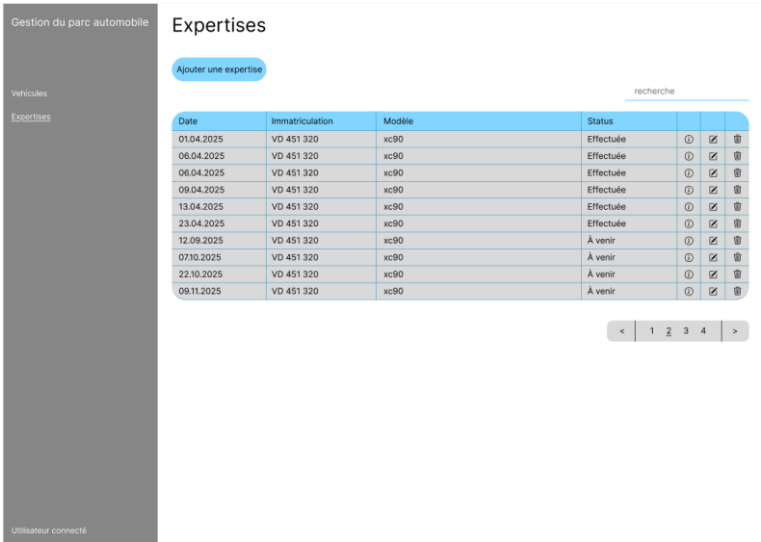


FIGURE 6 MAQUETTE PAGE LISTE D'EXPERTISES

## AJOUT

Les maquettes ci-dessous serviront de modèles pour les pages d'ajout de l'application.

Maquette de la page 'Ajouter un véhicule'. Le formulaire est structuré en trois colonnes. La première colonne contient les champs 'Marque', 'Année' et 'Immatriculation'. La deuxième colonne contient 'Modèle', 'Kilométrage' et 'VIN'. La troisième colonne contient 'Type', 'Photo - glisser et déposer' et 'Status' avec une flèche vers le bas. Un bouton 'Ajouter' est situé en bas à droite. À gauche, une barre latérale grise contient le menu 'Gestion du parc automobile' avec les sous-menus 'Vehicules' et 'Expertises', et 'Utilisateur connecté' en bas.

FIGURE 7 MAQUETTE PAGE AJOUT DE VÉHICULE

Maquette de la page 'Ajouter une expertise'. Le formulaire est structuré en trois colonnes. La première colonne contient le champ 'Date'. La deuxième colonne contient 'Voiture' avec une flèche vers le bas. La troisième colonne contient 'Employé' avec une flèche vers le bas. Un bouton 'Ajouter' est situé en bas à droite. À gauche, une barre latérale grise contient le menu 'Gestion du parc automobile' avec les sous-menus 'Vehicules' et 'Expertises', et 'Utilisateur connecté' en bas.

FIGURE 8 MAQUETTE PAGE AJOUT D'EXPERTISE

La maquette suivante montre le popup qui s'affiche sur l'écran d'ajout d'un véhicule lorsque l'utilisateur entre une marque de véhicule pour choisir le modèle.

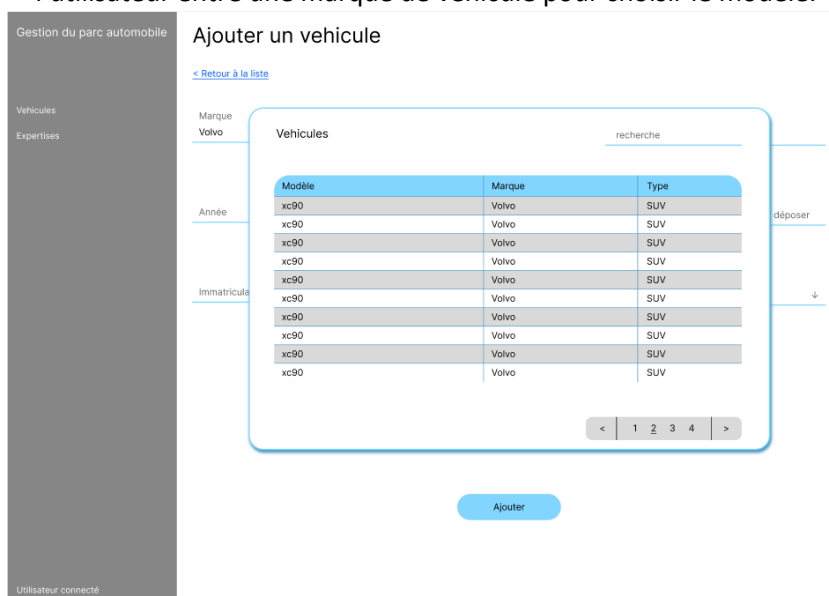


FIGURE 9 POPUP AJOUT D'UN VÉHICULE

## MODIFICATION

Les maquettes suivantes représentent les pages de modifications dans l'application.

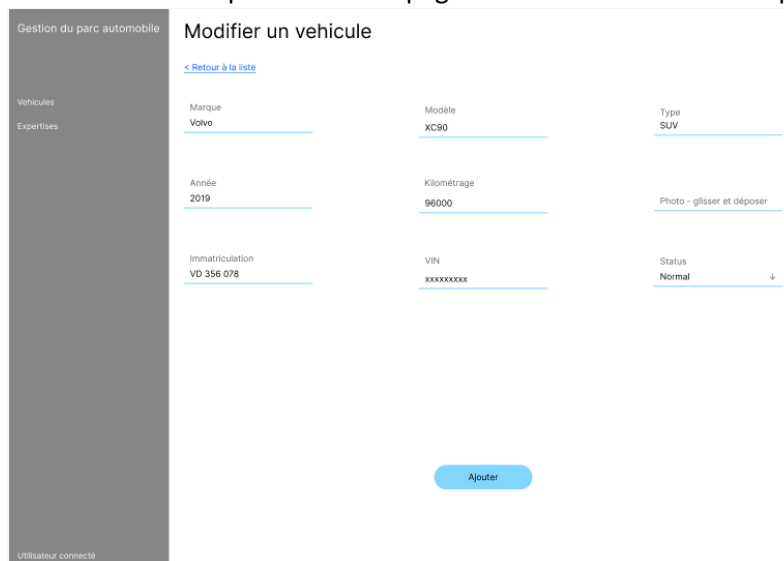


FIGURE 10 MAQUETTE PAGE MODIFICATION DE VÉHICULE

Maquette de la page 'Modifier une expertise'. Le formulaire est divisé en sections : une barre de navigation latérale, un titre, un lien de retour, des champs de saisie pour la date, le véhicule, l'employé et le statut, un champ de note, et un bouton 'Ajouter'.

**Gestion du parc automobile**

Véhicules  
Expertises

### Modifier une expertise

[< Retour à la liste](#)

Date: 01.12.2025

Véhicule: XC90 - VD 389 530

Employé: Elliott Deriaz

Status: Effectué

Note:

[Ajouter](#)

Utilisateur connecté

FIGURE 11 MAQUETTE PAGE MODIFICATION D'EXPERTISE

La maquette suivante montre le popup qui s'affiche lorsque l'utilisateur modifie la marque du véhicule afin de choisir un modèle.

Maquette du popup 'Modifier un véhicule'. Le popup est divisé en sections : une barre de navigation latérale, un titre, un lien de retour, des champs de saisie pour la marque, l'année et l'immatriculation, une table de véhicules, et un bouton 'Ajouter'.

**Gestion du parc automobile**

Véhicules  
Expertises

### Modifier un véhicule

[< Retour à la liste](#)

Marque: Volvo

Année: 2019

Immatriculation: VD 356 076

**Véhicules**

Modèle	Marque	Type
xc90	Volvo	SUV
xc90	Volvo	SUV
xc90	Volvo	SUV
xc90	Volvo	SUV
xc90	Volvo	SUV
xc90	Volvo	SUV
xc90	Volvo	SUV
xc90	Volvo	SUV
xc90	Volvo	SUV

[Ajouter](#)

Utilisateur connecté

FIGURE 12 POPUP SÉLECTION MODÈLE VÉHICULE

DÉTAIL

Ici les maquettes serviront de modèles pour les pages de détail de l’application.

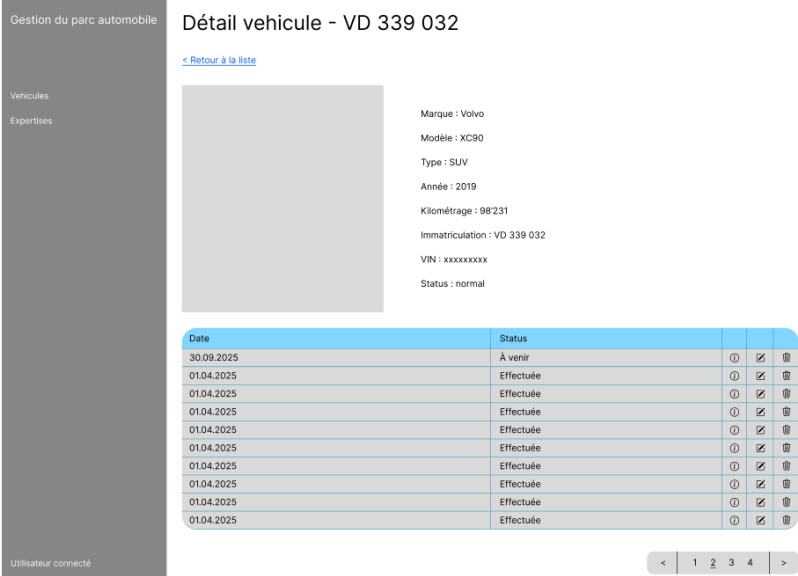


FIGURE 13 MAQUETTE PAGE DÉTAIL VÉHICULE

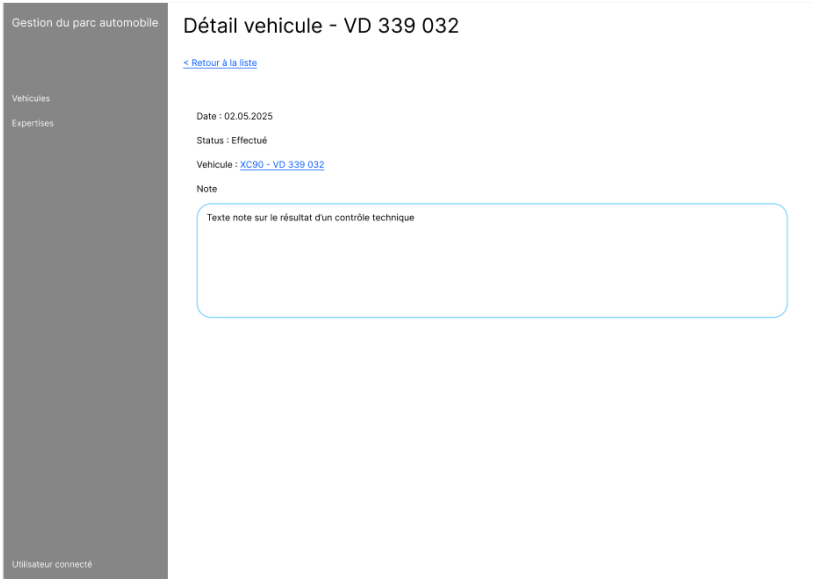


FIGURE 14 MAQUETTE PAGE DÉTAIL EXPERTISE

## VALIDATION D'UNE EXPERTISE

Sur la liste des expertises pour les employés, lorsqu'ils cliqueront sur l'icône de validation, un popup s'affichera pour entrer la potentielle note et valider l'expertise.

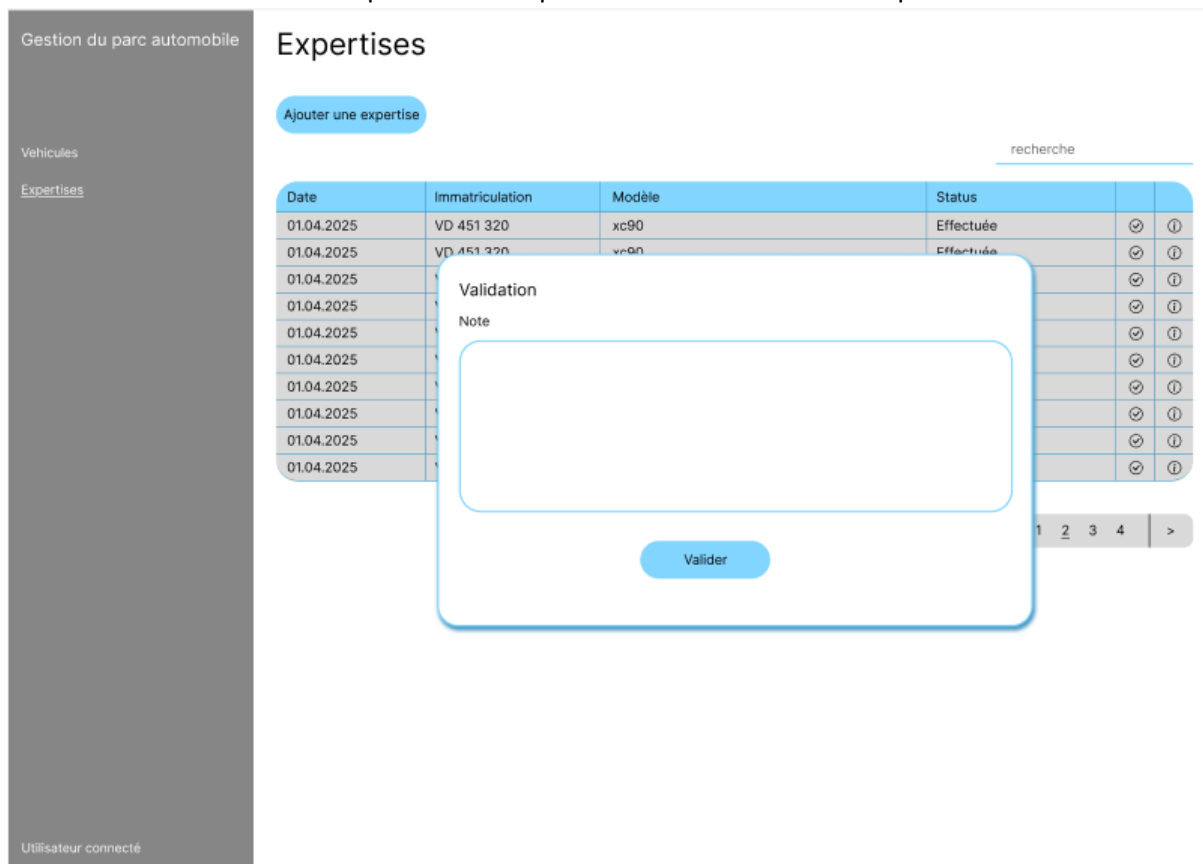


FIGURE 15 MAQUETTE POPUP VALIDATION EXPERTISE

## STRATÉGIE DE TEST

ID	Fonctionnalité	Définition	Résultat attendu
T01	Authentification	Sur la page d'authentification, je m'authentifie avec des informations correctes.	Redirection sur la page liste des véhicules.
T02	Affichage des véhicules	Sur la page de liste des véhicules, je veux afficher une liste des véhicules.	À la fin du chargement un tableau s'affiche avec tous les véhicules.
T03	Ajout d'un véhicule	Sur la page d'ajout d'un véhicule, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur ajouter.	L'administrateur est redirigé sur la liste des véhicules et un message lui indique que le véhicule a été ajouté.
T04	Sélection marque	Sur la page d'ajout d'un véhicule.	Un dropdown sous l'input marque permet de sélectionner

			la marque sur la base de la réponse API.
T05	Sélection modèle	Sur la page d'ajout d'un véhicule.	Un dropdown sous l'input modèle permet de sélectionner le modèle sur la base de la réponse API.
T06	Modification d'un véhicule	Sur la page de modification d'un véhicule, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur modifier.	L'administrateur est redirigé sur la liste des véhicules et un message lui indique que le véhicule a été modifié.
T07	Sélection marque	Sur la page de modification d'un véhicule.	Un dropdown sous l'input marque permet de sélectionner la marque sur la base de la réponse API.
T08	Sélection modèle	Sur la page de modification d'un véhicule.	Un dropdown sous l'input modèle permet de sélectionner le modèle sur la base de la réponse API.
T09	Affichage détails véhicule	Dans la liste, lorsqu'un utilisateur clique sur l'icône des détails.	Je suis redirigé sur une page avec les détails du véhicule
T10	Suppression véhicule	Dans la liste lorsque, l'administrateur clique sur l'icône corbeille d'un véhicule et qu'il confirme.	Le véhicule est supprimé et un message lui indique qu'il a bien été supprimé.
T11	Affichage des expertises	Lorsqu'un administrateur va sur la page des expertises.	Un tableau s'affiche avec toutes les expertises.
T12	Affichage des expertises	Lorsqu'un employé va sur la page des expertises.	Un tableau s'affiche avec toutes les expertises qui lui sont attribuées.
T13	Ajout d'une expertise	Sur la page d'ajout d'une expertise, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur ajouter.	L'administrateur est redirigé sur la liste des expertises et un message lui indique que l'expertise a été ajoutée.
T14	Modification d'une expertise	Sur la page de modification d'une expertise, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur modifier.	L'administrateur est redirigé sur la liste des expertises et un message lui indique que l'expertise a été modifiée.



T15	Suppression d'une expertise	Dans la liste, lorsque l'administrateur clique sur l'icône corbeille d'une expertise et qu'il confirme.	L'expertise est supprimée et un message lui indique qu'elle a bien été supprimée.
T16	Changement statut d'une expertise	Dans la liste, lorsque l'employé clique sur l'icône pour changer le statut, qu'il remplit ou pas la note dans le popup et qu'il confirme.	L'expertise passe en statut effectué.
T17	Déploiement	Lorsque j'accède à l'URL du site.	Le site s'ouvre et toutes les fonctionnalités sont disponibles.

## RISQUES TECHNIQUES

Selon moi, les principaux risques techniques résident dans l'intégration de l'API où je pourrais rencontrer des erreurs, puisque je n'ai pas beaucoup d'expérience et dans le déploiement de l'application du fait que je ne suis pas très rapide en système et donc je pourrais perdre du temps.

## RÉALISATION

Dans cette partie, je décris la mise en œuvre du projet.

### CONNEXION À LA BASE DE DONNÉES

Dans mon projet, le fichier appsettings.json contient le string de connexion à la base de données, c'est ce texte que j'utilise dans program.cs pour initialiser le context de l'application.

Au lancement de l'application, après que la connexion à la base de données a réussi, j'utilise la méthode `UseAsyncSeeding()` de la classe `DbContextOptionsBuilder` pour contrôler que les informations nécessaires au bon fonctionnement de l'application existent et les ajouter si ce n'est pas le cas.

### AUTHENTIFICATION ET GESTION DES RÔLES

Pour la gestion des rôles et des utilisateurs, j'utilise ASP.NET Core Identity. C'est une API qui me permet de gérer les utilisateurs et les rôles plus simplement.

Pour créer les rôles, j'ai un script qui s'exécute au lancement de l'application qui va contrôler si les rôles existent bien sinon il les crée et si aucun utilisateur n'existe va créer un administrateur de base.

Cette API crée aussi les pages de bases pour la gestion du compte donc j'ai simplement dû modifier les pages existantes pour qu'elles correspondent au style de l'application.

Ci-dessous l'avant et l'après modification.

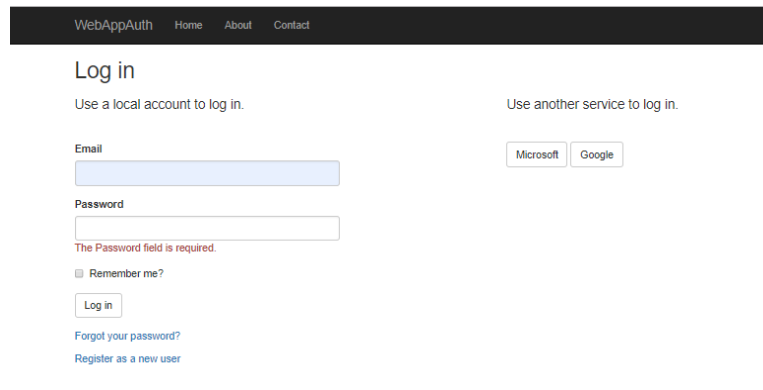


FIGURE 16 PAGE D'AUTHENTIFICATION DE BASE

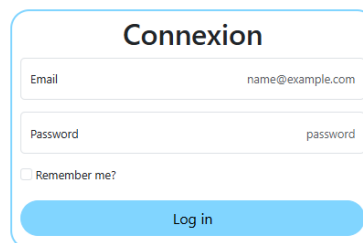


FIGURE 17 PAGE D'AUTHENTIFICATION MODIFIÉE

Avec Identity, je peux aussi gérer les autorisations pour que seulement les administrateurs puissent accéder aux pages de gestion des rôles et des utilisateurs.

Pour limiter l'accès à une page ou un contrôleur, il suffit de mettre cet attribut sur la classe/méthode :  
[Authorize(Roles = "Admin")]

Celle-ci autorise uniquement l'accès aux utilisateurs dans le rôle « Admin ».

Pour simplifier la gestion des utilisateurs et des rôles pour les administrateurs, j'ai créé une gestion des utilisateurs qui leur permet d'ajouter de modifier et de supprimer les utilisateurs ainsi qu'une gestion des rôles pour attribuer les rôles aux utilisateurs.

**Gestion des utilisateurs**

Ajouter un utilisateur

Afficher 10 entrées Rechercher :

Nom	Email	Rôles
Eliott Deriaz	eliott.deriaz@gmail.com	Employee
Eliott Deriaz	admin@GestParc.local	Admin

Affichage de 1 à 2 sur 2 entrées

Première Précédente 1 Suivante Dernière

FIGURE 18 GESTION DES UTILISATEURS

**Rôles de l'application**

10 entries per page Search:

Nom
Admin
Employee

Showing 1 to 2 of 2 entries

<< < 1 > >>

FIGURE 19 GESTION DES RÔLES

Lors de l'ajout d'un utilisateur, l'administrateur peut choisir de forcer le changement de mot de passe. L'utilisateur sera redirigé après sa prochaine authentification sur une page pour modifier son mot de passe.

**Modifier le mot de passe**

Ancien mot de passe

Nouveau mot de passe

Confirmer le mot de passe

Modifier

FIGURE 20 PAGE DE MODIFICATION DU MOT DE PASSE

## GESTION DES VÉHICULES

Dans cette section. Je décris toute la réalisation en rapport avec la gestion des véhicules.

### LISTE

Pour l’affichage des données, j’utilise DataTables. Je récupère les données avec via Ajax puis je définis les colonnes avec un titre, la donnée, et pour les colonnes dans lesquelles il y a des liens, j’utilise la fonction de rendu pour afficher un <a> avec une redirection sur la bonne page. Je peux aussi définir la visibilité des colonnes, fonction que j’utilise pour cacher les colonnes pour les administrateurs aux employés.

Gestion de parc

- Véhicules
- Expertises
- Gestion des utilisateurs
- Gestion des rôles

admin@GestParc.local

**Liste des Véhicules**

Ajouter un véhicule

Afficher 10 entrées

Rechercher :

Modèle	Immatriculation	Marque	Type	Kilométrage	VIN
CR-Z	VDScuz	Honda	Coupé sport	261000	jspWllh
XC90	VD111023	Volvo	SUV	0	871728371828374712

Affichage de 1 à 2 sur 2 entrées

Première Précédente 1 Suivante Dernière

FIGURE 21 LISTE DES VÉHICULES VUE ADMINISTRATEUR

### AJOUT

Sur cette page, j’utilise le véhicule comme modèle pour que lors de l’envoi, je puisse lier directement les données de la requête POST au paramètre de la méthode. Pour gérer l’envoi de l’image, je récupère de la requête un objet IFormFile que je vais ensuite transformer en tableau de byte (byte[]) puisqu’elle est enregistrée dans ce type dans la base de données.

Pour la sélection du statut du véhicule, ASP.NET possède un type SelectList que l’on peut lier à un select qui crée automatiquement les options du select.

Gestion de parc

Véhicules

Expertises

Gestion des utilisateurs

Gestion des rôles

admin@GestParc.local

## Ajout d'un véhicule

[Retour à la liste](#)

Marque	Nissan	Modèle	Modèle	Type	Type
Date d'immatriculation	jj.mm.aaaa	Kilométrage	Kilométrage	Image	Choisir un fichier   Aucun fichier choisi
Immatriculation	Immatriculation	VIN	VIN	Statut	▼

Créer

FIGURE 22 PAGE D'AJOUT D'UN VÉHICULE

## MODIFICATION

Sur cette page, j'utilise le même principe que sur la page d'ajout, mais je donne à la vue le véhicule concerné.

Gestion de parc

Véhicules

Expertises

Gestion des utilisateurs

Gestion des rôles

admin@GestParc.local

## Modification d'un véhicule

[Retour à la liste](#)

Marque	Volvo	Modèle	XC90	Type	SUV
Date d'immatriculation	01.05.2025	Kilométrage	0	Image	Choisir un fichier   Aucun fichier choisi
Immatriculation	VD111023	VIN	871728371828374712	Statut	Normal ▼

Modifier

FIGURE 23 PAGE DE MODIFICATION D'UN VÉHICULE

## DÉTAIL

Sur cette page, j'utilise à nouveau l'objet véhicule comme modèle pour afficher les informations.

Pour afficher l'image, je dois utiliser comme source l'image convertie en texte Base64 :

`src="data:image/png;base64,@Convert.ToBase64String(Model.Image ?? new byte[0])"`

Sur cette page, j'affiche aussi une liste des expertises liées au véhicule. Il est possible de gérer les expertises du véhicule depuis cette table.

Gestion de parc

Véhicules

Expertises

Gestion des utilisateurs

Gestion des rôles

admin@GestParc.local

## Détails véhicule - VD11023

[Retour à la liste](#)

Marque : Volvo

Modèle : XC90

Type : SUV

Date d'immatriculation : 01.05.2025

Immatriculation : VD11023

VIN : 871728371828374712

Statut : N

Afficher 10 entrées

Rechercher :

Date	Statut	Employé
Aucune donnée disponible dans le tableau		

Affichage de 0 à 0 sur 0 entrées

Première Précédente Suivante Dernière

FIGURE 24 PAGE DE DÉTAIL D'UN VÉHICULE

## SUPPRESSION

La suppression des véhicules se fait depuis la page liste des véhicules, il s'agit simplement d'un bouton avec une confirmation qui envoie ensuite une requête Ajax. Si la suppression est un succès, un message s'affiche pour confirmer la suppression.

Gestion de parc

Véhicules

Expertises

Gestion des utilisateurs

Gestion des rôles

admin@GestParc.local

## Liste des Véhicules

Ajouter un véhicule

Afficher 10 entrées

Rechercher :

Modèle	Immatriculation	Marque	Type	Kilométrage	VIN
CR-Z	VDSuz	Honda	Coupé sport	261000	jspWlh
XC90	VD11023	Volvo	SUV	0	871728371828374712

Affichage de 1 à 2

Précédente 1 Suivante Dernière

FIGURE 25 POPUP CONFIRMATION SUPPRESSION VÉHICULE

Gestion de parc

Véhicules

Expertises

Gestion des utilisateurs

Gestion des rôles

admin@GestParc.local

## Liste des Véhicules

Ajouter un véhicule

Afficher 10 entrées

Rechercher :

Modèle	Immatriculation	Marque	Type	Kilométrage	VIN
CR-Z	VDSuz	Honda	Coupé sport	261000	jspWlh
XC90	VD11023	Volvo	SUV	0	871728371828374712

Affichage de 1 à 2

Précédente 1 Suivante Dernière

FIGURE 26 CONFIRMATION SUPPRESSION VÉHICULE

## CONTRÔLE DU TYPE DE FICHIER

Afin de contrôler que le fichier envoyé à l'application par l'utilisateur est bel et bien une image, j'ai ajouté deux contrôles ; l'un est sur la page directement en HTML en utilisant l'attribut accept de l'input.

```
<input asp-for="Image" class="form-control" aria-required="true"
placeholder="Image" type="file" accept="image/jpeg, image/png, image/svg+xml,
image/gif"/>
```

L'autre est situé dans le backend, il existe un attribut de validation pour contrôler l'extension de fichier cependant, il ne marche qu'avec le type string. J'ai donc dû créer une classe me permettant de faire ce contrôle.

```
public class AllowExtensionsAttribute : ValidationAttribute
{
    private string _extensions = "png";

    public AllowExtensionsAttribute(string? extensions)
    {
        this._extensions = extensions ?? this._extensions;
    }
    protected override ValidationResult IsValid(object value, ValidationContext validationContext)
    {
        var file = value as IFormFile;
        var extension = Path.GetExtension(file.FileName).Trim('.');
        if (file != null)
        {
            if (!_extensions.Split(',').Contains(extension.ToLower()))
            {
                return new ValidationResult(this.ErrorMessage);
            }
        }
        return ValidationResult.Success;
    }
}
```

Ci-dessus, la classe qui me permet de faire cette validation. Elle contient en plus des propriétés de base, une chaîne de caractères contenant les extensions autorisées, par défaut que le type png. Lorsque la classe est appelée pour réaliser le contrôle, elle va exécuter la méthode IsValid(). Dans celle-ci, je récupère le fichier envoyé, je récupère uniquement l'extension de fichier et je contrôle que la propriété contenant les extensions contient bien l'extension du fichier. Sinon, je renvoie le message d'erreur.

Il s'utilise de la manière suivante :

```
[AllowExtensions("pjp,jpg,pjpeg,jpeg,jfif,png,svgz,svg,gif", ErrorMessage = "Le
type de fichier sélectionné n'est pas valide.")] IFormFile? image
```

## GESTION DES EXPERTISES

Ici, je détaille l'implémentation de la partie gestion des expertises.

## LISTE

Sur cette page, j'initialise une table avec les expertises, elle contient la date, l'immatriculation, la marque, le statut ainsi que l'employé responsable de l'expertise. Pour un affichage optimal des expertises, il est possible de trier par colonne en ordre croissant ou décroissant ainsi que de rechercher une entrée. Les utilisateurs peuvent accéder à la page de détail de l'expertise, les employés peuvent en plus de cela valider ou dévalider l'expertise en fonction du statut et les administrateurs peuvent modifier et supprimer les expertises. Depuis cette page, il est aussi possible aux administrateurs d'accéder à la page d'ajout d'une expertise.

**Gestion de parc**

- Véhicules
- Expertises
- Gestion des utilisateurs
- Gestion des rôles

**Liste des expertises**

Ajouter une expertise

Afficher 10 entrées

Rechercher :

Date	Immatriculation	Marque	Statut	Employé
07.01.2026 10:30	VD111023	Volvo	À faire	Maël Gétain
24.05.2025 13:40	VDTheo	Honda	Effectué	Elliott Deriaz

Affichage de 1 à 2 sur 2 entrées

Première Précédente 1 Suivante Dernière

admin@GestParc.local

FIGURE 27 PAGE LISTE DES EXPERTISES VUE ADMINISTRATEURS

**Gestion de parc**

- Véhicules
- Expertises

**Liste des expertises**

Afficher 10 entrées

Rechercher :

Date	Immatriculation	Marque	Statut
23.07.2025 16:50	VD111023	Volvo	Effectué
24.05.2025 13:40	VDTheo	Honda	Effectué
30.12.2025 17:43	VD2002	Kawa Jérém	À faire

Affichage de 1 à 3 sur 3 entrées

Première Précédente 1 Suivante Dernière

elriott.deriaz@gmail.com

FIGURE 28 LISTE DES EXPERTISES VUE EMPLOYÉ

## AJOUT

Il est possible aux administrateurs d'ajouter des expertises à deux endroits ; sur la liste des véhicules ou sur la page d'ajout d'une expertise accessible depuis la page liste des expertises. Cela permet aux administrateurs d'effectuer leurs tâches plus rapidement en fonction de leur emplacement.



The screenshot shows the 'Ajout d'une expertise' page. On the left is a dark sidebar with the following menu items: 'Gestion de parc', 'Véhicules', 'Expertises', 'Gestion des utilisateurs', and 'Gestion des rôles'. The main content area has the title 'Ajout d'une expertise' with a link 'Retour à la liste' below it. There are three input fields: 'Date' with a placeholder 'jj.mm.aaaa --:--', a 'Véhicule' dropdown menu, and an 'Employé' dropdown menu. At the bottom center is a blue 'Créer' button. The footer shows the user 'admin@GestParc.local'.

FIGURE 29 PAGE AJOUT D'UNE EXPERTISE

Comme précisé précédemment, il est aussi possible d'ajouter une expertise sur la page liste des véhicules. Lorsque l'administrateur clique sur l'icône d'ajout d'une expertise un modal s'ouvre dans lequel il est possible de préciser la date ainsi que l'employé responsable de l'expertise.

The screenshot shows a modal titled 'Ajouter une expertise VD111023'. It contains a 'Date' field with a calendar icon and a placeholder 'jj.mm.aaaa --:--', and an 'Employé' dropdown menu. The dropdown is open, showing a search bar and a list of email addresses: 'eliott.deriaz@gmail.com', 'mael.getain@gestparc.local', and 'testexp@local.local'. The 'Employé' label is highlighted in blue. In the background, a table with columns 'Nom', 'Marque', 'Modèle', and 'Type' is partially visible, showing rows for 'Volvo' and 'SUV'.

FIGURE 30 MODAL AJOUT D'UNE EXPERTISE

Les selects pour le véhicule et pour l'employé contiennent une searchbox pour permettre aux administrateurs de choisir facilement les entrées concernées.

## MODIFICATION

S'il y a une erreur dans les informations d'une expertise (p. ex. date erronée ou validation maladroite) les administrateurs ont accès à une page de modification des expertises accessibles depuis la page liste des expertises.

The screenshot shows the 'Modifier une expertise' page. On the left is a dark sidebar with navigation links: 'Gestion de parc', 'Véhicules', 'Expertises', 'Gestion des utilisateurs', and 'Gestion des rôles'. The main content area has a title 'Modifier une expertise' and a link 'Retour à la liste'. Below the title, there are two dropdown menus: 'Date' (showing '07.01.2026 10:30') and 'Immatriculation' (showing 'VD111023'). To the right of these is a dropdown for 'Marque' (showing 'mael.getain@gestparc.local'). Below these is a checkbox labeled 'Statut'. A large text area for 'Note' is present, with a 'Marque' label on the right. At the bottom right is a blue 'Modifier' button. The user's email 'admin@GestParc.local' is visible in the bottom left of the sidebar.

FIGURE 31 PAGE MODIFICATION D'UNE EXPERTISE

## SUPPRESSION

La suppression se fait sur la page liste des expertises en cliquant sur l'icône corbeille d'une expertise, il faut ensuite confirmer sur le popup qui s'affiche comme pour les véhicules. Seuls les administrateurs ont accès à cette fonctionnalité.

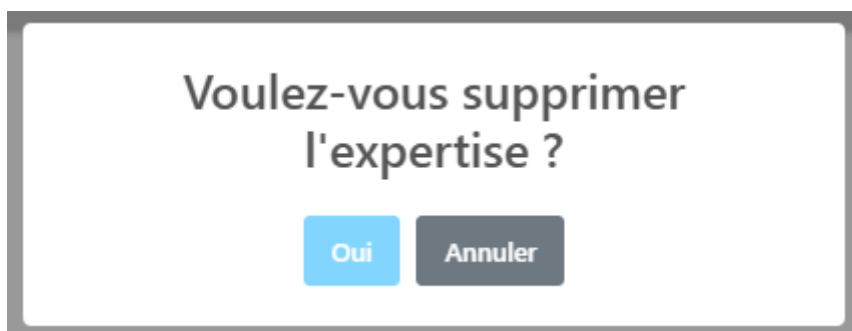


FIGURE 32 POPUP SUPPRESSION D'UNE EXPERTISE

## VALIDATION D'UNE EXPERTISE

Les employés ont accès à un bouton permettant de valider une expertise. Lorsqu'ils cliquent dessus un popup s'affiche afin qu'ils puissent entrer ou non une note (texte afficher sur un résultat d'expertise).

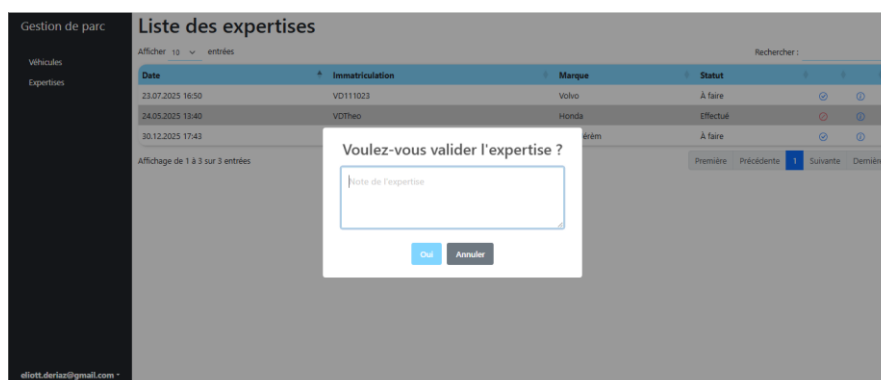


FIGURE 33 POPUP VALIDATION EXPERTISE

## IMPLÉMENTATION DE L'API

J'avais dans un premier temps prévu de réaliser un popup permettant de sélectionner le modèle sur la base de la marque entrée (Figure 9). Cependant, je ne trouvais pas cette manière de faire satisfaisante et intuitive, c'est pourquoi j'ai décidé d'inclure sous les inputs de marque et de modèle un dropdown avec les options disponibles.

## CONNEXION À L'API

L'application est mise à disposition sur rapidapi.com et à un plan gratuit permettant d'effectuer jusqu'à 1000 requêtes par mois, ce qui est amplement suffisant pour la réalisation de ce projet. Cependant, si un jour cette application venait à être déployée à grande échelle, il faudrait considérer passer à un plan payant pour bénéficier de plus de requêtes par mois.

Pour réaliser la connexion de base, j'ai créé une classe API qui permet de construire les requêtes et une classe CarDataAPI héritée de la classe API dans laquelle j'ai mis les informations relatives à la connexion (URL, Headers) ainsi que les méthodes pour envoyer les requêtes API.

```
public static HttpRequestMessage BuildRequest(Dictionary<string, string>? parameters, Dictionary<string, string>? body, Dictionary<string, string>? headers, string defaultUrl, string action, string urlComplement, HttpMethod method)
{
    //Build Url
    string url = defaultUrl + action;

    if (parameters != null)
    {
        foreach (var item in parameters)
        {
            if (parameters.ToList().IndexOf(item) == 0)
                url += "?";
            else
                url += "&";
            url += $"{item.Key}={item.Value}";
        }
    }

    url += urlComplement;

    //Build Body
    string bodyString = "{";
    if (body != null)
    {
        foreach (var item in body)
        {
            bodyString += $"{item.Key}={item.Value},";
        }
    }
    bodyString = "}";
    JsonContent content = JsonContent.Create(bodyString);
```

```
//Build Request With Info gotten
HttpRequestMessage req = new HttpRequestMessage()
{
    RequestUri = new Uri(url),
    Method = method,
    Content = content
};

foreach (var item in headers)
{
    req.Headers.Add(item.Key, item.Value);
}

return req;
}
```

La méthode ci-dessus me permet de construire les requêtes API avec toutes les informations nécessaires.

### SÉLECTION DE LA MARQUE

L'API me renvoie une liste complète des marques disponibles. Je renvoie donc cette liste à la vue et à l'aide d'un foreach, j'initialise les options dans une datalist que j'attribue à l'input marque.

```
<datalist id="makes">
    @foreach (string i in ViewBag.MakeSelectList){
        <option>@i</option>
    }
</datalist>
```

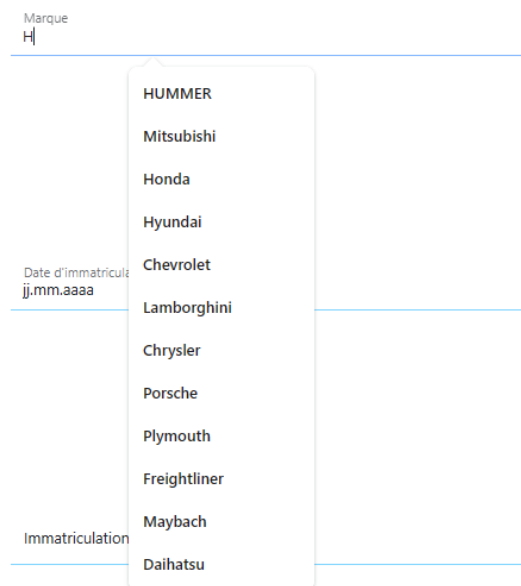


FIGURE 34 DROPDOWN SÉLECTION MARQUE

## SÉLECTION DU MODÈLE

Ici, je ne peux pas renvoyer directement la liste de tous les modèles à la vue. J'utilise donc ajax pour récupérer la liste de modèles, j'envoie la marque ainsi que la modèle entrée pour avoir une sélection précise. Pour éviter d'avoir une erreur à cause du nombre de requêtes ou d'envoyer trop de requêtes à l'API, j'utilise une fonction de debounce que j'ai trouvée sur internet.

```
const updateModelDataList = debounce(() => getDataModelDetaList(), 1000)

function debounce(func, timeout = 300){
  let timer;
  return (...args) => {
    clearTimeout(timer);
    timer = setTimeout(() => { func.apply(this, args); }, timeout);
  };
}

function getDataModelDetaList(){
  var model = $('#Model').val();
  var make = $('#Make').val();

  $.ajax({
    url: '/Vehicles/GetModelList',
    data: {
      make: make,
      model: model,
    },
    type: "GET",
    success: function (data){
      $('#models').empty();
      data.forEach((elem) => $('#models').append("<option
onclick='attributeVal(this)' value='" + elem.model + "'>" + elem.type +
"</option>"));
    }
  });
}
```

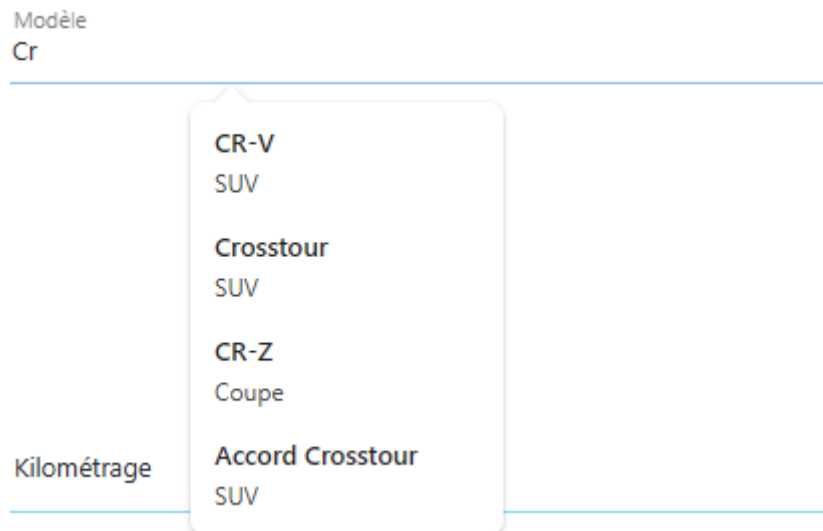


FIGURE 35 DROPDOWN SÉLECTION MODÈLE

À la sélection d'une option, le champ type de véhicule est aussi rempli.

## TABLEAU DE BORD

Puisque j'étais assez en avance par rapport à la planification, j'ai décidé de réaliser un tableau de bord pour les administrateurs leur permettant d'accéder facilement aux actions d'ajout d'expertise et de véhicule cette page contient aussi un aperçu des prochaines expertises et des derniers véhicules ajoutés. Selon moi, cet aperçu permettrait d'accélérer les actions que doivent exécuter les administrateurs.

**Gestion de parc**

- Tableau de bord
- Véhicules
- Expertises
- Gestion des utilisateurs
- Gestion des rôles

admin@GestParc.local

### Tableau de bord

[Ajouter une expertise](#) [Ajouter un véhicule](#)

#### Prochaines expertises

Date	Marque	Modèle	Immatriculation	Employé
24.05.2025 10:00	Kawa Jérém	Versys650	VD2002	Maël Gétain
23.07.2025 16:50	Volvo	XC90	VD111023	Eliott Deriaz
30.12.2025 17:43	Kawa Jérém	Versys650	VD2002	Eliott Deriaz
16.06.2028 12:40	Kawa Jérém	Versys650	VD2002	Eliott Deriaz

#### Derniers véhicules

Marque	Modèle	Immatriculation	Kilométrage	Prochaine expertise
Dodge	Charger		2	16.05.2025 11:25
Honda	Jazz	VDTheo	0	24.05.2025 13:40
Kawa Jérém	Versys650	VD2002	5500	16.06.2028 12:40
Volvo	XC90	VD111023	0	23.07.2025 16:50

FIGURE 36 TABLEAU DE BORD

## DÉPLOIEMENT DE L'APPLICATION

Pour déployer mon application, je commence par créer un profil de publication vers un dossier connu puis j'effectue la publication qui me donne un dossier publish avec toute l'application. Je crée à partir de ce dossier une archive zip que je mets dans ma release github qui me permettra de récupérer le projet sur le serveur.

Sur le serveur, j'ai dû installer les choses suivantes :

- Dotnet 9
- Dotnet-sdk-9.0
- Aspnetcore-runtime-9.0
- Dotnet-runtime-9.0
- Nginx

J'ai, après cela, configuré Nginx en reverse pour faire en sorte qu'il écoute le port 80 et qu'il passe localhost:5000 l'emplacement de l'application.

Après avoir récupéré le projet, je l'ai dézippé et j'ai contrôlé qu'il fonctionne bien en exécutant la commande suivante dans le dossier :

dotnet GestionParcAuto.dll
----------------------------

Puisque je pouvais y accéder, j'ai ajouté dans le dossier /etc/systemd/system/ un fichier de service pour que l'application soit lancée à tout moment.

## DESCRIPTION DES TESTS EFFECTUÉS

ID	Définition	Résultat attendu	Statut
T01	Sur la page d'authentification, je m'authentifie avec des informations correctes.	Redirection sur la page liste des véhicules.	Réussi
T02	Sur la page de liste des véhicules, je veux afficher une liste des véhicules.	À la fin du chargement un tableau s'affiche avec tous les véhicules.	Réussi
T03	Sur la page d'ajout d'un véhicule, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur ajouter.	L'administrateur est redirigé sur la liste des véhicules et un message lui indique que le véhicule a été ajouté.	Réussi
T04	Sur la page d'ajout d'un véhicule.	Un dropdown sous l'input marque permet de sélectionner la marque sur la base de la réponse API.	Réussi
T05	Sur la page d'ajout d'un véhicule.	Un dropdown sous l'input modèle permet de sélectionner le modèle sur la base de la réponse API.	Réussi
T06	Sur la page de modification d'un véhicule, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur modifier.	L'administrateur est redirigé sur la liste des véhicules et un message lui indique que le véhicule a été modifié.	Réussi
T07	Sur la page de modification d'un véhicule.	Un dropdown sous l'input marque permet de sélectionner la marque sur la base de la réponse API.	Réussi

T08	Sur la page de modification d'un véhicule.	Un dropdown sous l'input modèle permet de sélectionner le modèle sur la base de la réponse API.	Réussi
T09	Dans la liste, lorsqu'un utilisateur clique sur l'icône des détails.	Je suis redirigé sur une page avec les détails du véhicule	Réussi
T10	Dans la liste lorsque, l'administrateur clique sur l'icône corbeille d'un véhicule et qu'il confirme.	Le véhicule est supprimé et un message lui indique qu'il a bien été supprimé.	Réussi
T11	Lorsqu'un administrateur va sur la page des expertises.	Un tableau s'affiche avec toutes les expertises.	Réussi
T12	Lorsqu'un employé va sur la page des expertises.	Un tableau s'affiche avec toutes les expertises qui lui sont attribuées.	Réussi
T13	Sur la page d'ajout d'une expertise, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur ajouter.	L'administrateur est redirigé sur la liste des expertises et un message lui indique que l'expertise a été ajoutée.	Réussi
T14	Sur la page de modification d'une expertise, lorsque les informations nécessaires sont remplies et que l'administrateur clique sur modifier.	L'administrateur est redirigé sur la liste des expertises et un message lui indique que l'expertise a été modifiée.	Réussi
T15	Dans la liste, lorsque l'administrateur clique sur l'icône corbeille d'une expertise et qu'il confirme.	L'expertise est supprimée et un message lui indique qu'elle a bien été supprimée.	Réussi
T16	Dans la liste, lorsque l'employé clique sur l'icône pour changer le statut, qu'il remplit ou pas la note dans le popup et qu'il confirme.	L'expertise passe en statut effectué.	Réussi
T17	Lorsque j'accède à l'URL du site.	Le site s'ouvre et toutes les fonctionnalités sont disponibles.	Réussi

## ERREURS RESTANTES

La seule erreur restante est que le https n'est pas supporté cela est lié au fait que dans l'infrastructure, un autre reverse proxy fait la redirection vers le serveur sur lequel est hébergée l'application. Il faudrait ajouter le certificat sur le premier proxy, mais je n'y ai pas accès. L'infrastructure ressemble au schéma ci-dessous.



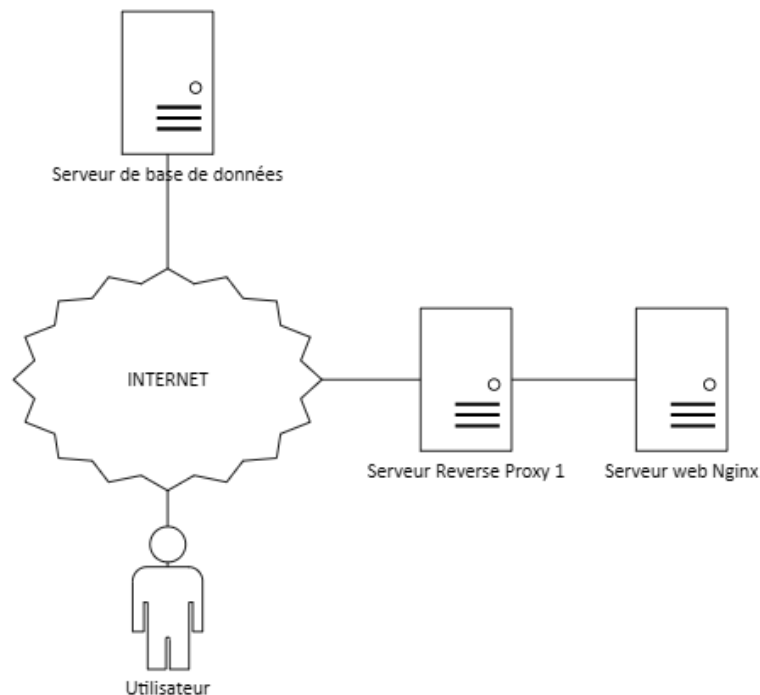


FIGURE 37 SCHÉMA INFRASTRUCTURE

## CONCLUSION

Cette partie conclue le rapport avec une réflexion sur le déroulement du projet.

## BILAN DES FONCTIONNALITÉS

Du côté des fonctionnalités, le cahier des charges demandait :

- Une gestion correcte de l'authentification
- Une liste des véhicules claire et intuitive
- Des opérations de base sur les véhicules (ajouter, modifier, supprimer et afficher le détail)
- Un aperçu optimal et rapide des expertises passées et futures
- La possibilité d'attribuer une expertise à un employé par l'administrateur
- Intégrer une API pour récupérer les marques et modèles des véhicules
- L'application doit être déployée sur l'infrastructure de l'ETML

Toutes ces fonctionnalités ont été implémentées. En plus de celles-ci, j'ai ajouté un tableau de bord pour les administrateurs. L'ajout d'une expertise peut se faire à deux endroits, depuis la page d'ajout d'une expertise et depuis la liste des véhicules pour que les administrateurs puissent réaliser l'ajout rapidement.

## BILAN PERSONNEL

Ce projet m'aura permis d'approfondir mes connaissances en développement d'application web et plus précisément en ASP.NET C#. Il m'a aussi permis de mieux comprendre les implications au niveau de la sécurité lors de la création de ce type d'application. De plus, le fait de mettre en ligne cette application aura permis d'ajouter un point de réalisme, dans le sens où c'est un type de projet que je pourrais retrouver en entreprise.

## BILAN PLANIFICATION

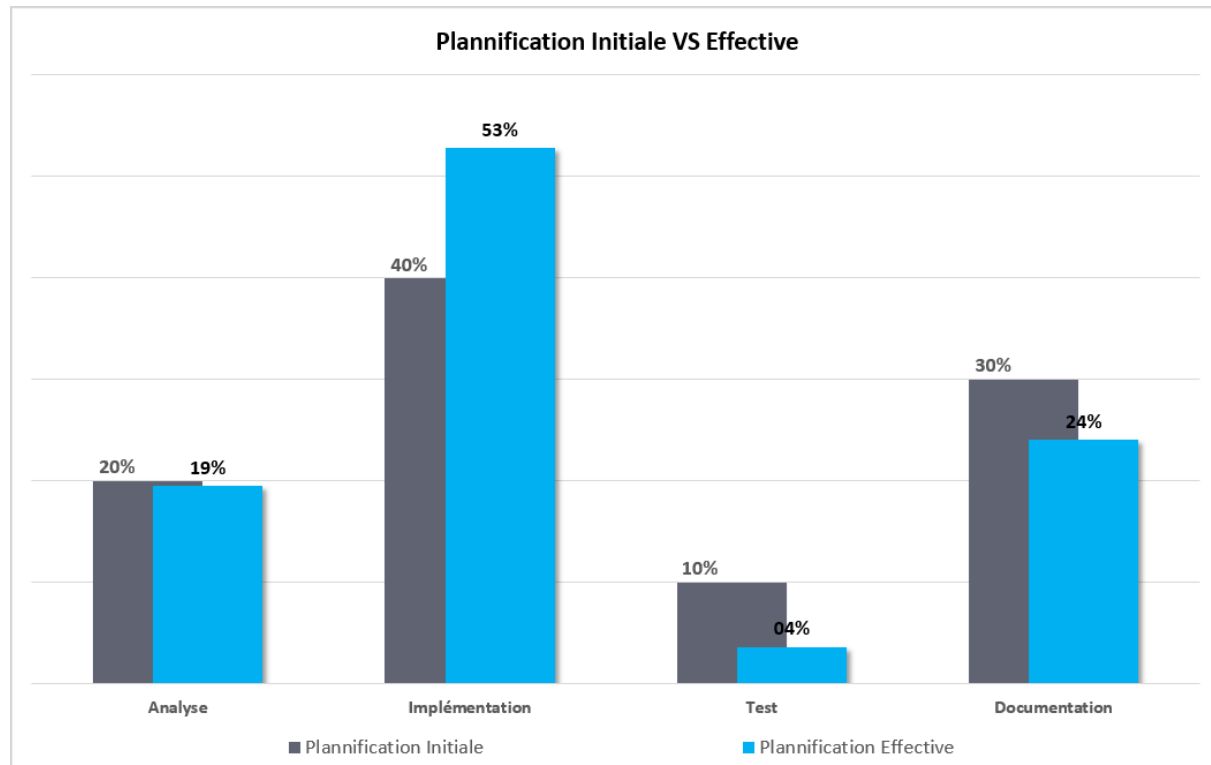


FIGURE 38 PLANIFICATION INITIALE VS EFFECTIVE

On peut voir, grâce au tableau ci-dessus, que c'est principalement la partie test qui n'est pas respectée, et cela, pour plusieurs raisons ; la première étant que le temps planifié pour les tests a été fait avant l'analyse complète du cahier des charges et donc je ne savais pas les tests que je réaliserai. Aussi, lors de l'implémentation, j'ai testé continuellement donc lorsque j'arrive à la fin de l'implémentation, je suis quasiment sûr que le test est réussi et cela me prend moins de temps.

Aussi grâce à ce temps gagné sur les tests, j'ai pu avancer sur l'implémentation et la documentation qui sont les parties les plus importantes du TPI.

Selon moi, le type de planification en cascade était tout à fait en adéquation avec le type de projet et les contraintes du TPI. L'ordre des tâches a été respecté bien que la plupart des tâches m'aient finalement pris moins de temps que prévu.

Maintenant par rapport à la méthodologie de travail, j'ai parfois passé trop de temps à régler des petits problèmes au lieu de passer à la tâche suivante, ce qui ne m'aura finalement pas pénalisé, mais qui pourrait dans un cas où la planification serait serrée.

Je pense aussi avoir réussi à bien gérer le fait de faire la documentation en parallèle de l'application. Cela me permet de mettre à l'écrit à chaud et de ne pas oublier des parties importantes que j'aurais voulu mettre dans ce rapport.

## AMÉLIORATIONS POSSIBLES

Avec un peu plus de temps, j'aurais pu une gestion des documents, le fait d'ajouter des documents à un véhicule ou une expertise afin d'en avoir le détail, cela servirait aussi d'historique. On pourrait imaginer attribuer à un véhicule la carte grise et les factures du garage par exemple.

L'un des points qui pourrait être amélioré par rapport à cette application, serait d'ajouter un moyen de savoir qui a le véhicule en ce moment et garder un historique de cela.

Une autre chose qui pourrait avoir du sens à cette application serait d'utiliser un active directory pour la gestion des comptes et des accès afin d'avoir la gestion des comptes d'entreprise centralisée.

Pour un usage réel, il pourrait être intéressant d'avoir un moyen d'importer des véhicules dans l'application à partir d'un fichier CSV par exemple.

Enfin, il pourrait être intéressant d'avoir un troisième type d'utilisateur qui pourrait accéder uniquement à la liste des véhicules et qui pourrait réserver ou voir la disponibilité d'un véhicule. On pourrait imaginer une application mobile qui récupérerait par API les informations de l'application.

## ANNEXES

### JOURNAL DE TRAVAIL

## RÉSUMÉ DU RAPPORT

### SITUATION INITIALE

Ce projet s'inscrit dans le cadre de mon TPI. Il vise à créer une application web pour la gestion de parc automobile d'entreprise. Cette application sera développée en ASP.NET et devra avoir les fonctionnalités suivantes : une gestion de l'authentification, une liste des véhicules, opération de gestion de base des véhicules (CRUD), une liste pour consulter la liste des expertises, les administrateurs doivent pouvoir attribuer les expertises à des employés, l'application devra intégrer une api pour récupérer une liste de marques et de modèles de véhicules. L'application devra aussi être déployée sur l'infrastructure locale de l'ETML.

### IMPLÉMENTATION

Pour la réalisation, j'ai séparé les tâches en fonction des fonctionnalités demandées. J'ai commencé par réaliser la gestion de l'authentification, en utilisant ASP.NET Identity, j'ai simplement dû modifier les pages existantes. En plus de la modification des pages, j'ai ajouté une gestion des utilisateurs et une gestion des rôles pour permettre aux administrateurs d'ajouter, supprimer des utilisateurs et gérer leurs rôles. Ensuite, je me suis attaqué à la réalisation de la gestion des véhicules. Pour ce faire, j'ai d'abord créé le modèle qui devait contenir au moins une photo, une marque, un modèle, un kilométrage et un numéro de plaque. Puis j'ai fait les pages liste, ajout, modification et détail puis j'ai ajouté la fonction de suppression des véhicules. Après cela, j'ai réalisé la partie de gestion des expertises. Elles sont caractérisées par une date, un véhicule et un employé responsable. J'ai décidé d'ajouter à cela une remarque qui correspond au champ remarques du résultat d'un contrôle technique. Après cela, j'ai créé les pages de la gestion et j'ai ajouté la possibilité de valider les expertises. Une fois la gestion des expertises finie, j'ai commencé l'intégration de l'API avec la création de la classe qui fait la connexion avec l'API. Puis, j'ai créé les méthodes qui permettent de récupérer les listes de marques et de modèles dans le contrôleur des véhicules. Enfin, j'ai ajouté aux pages de création et de modification de véhicules les listes permettant de sélectionner la marque et le modèle. À ce moment-là, j'avais un peu d'avance sur le planning, j'ai donc décidé d'ajouter une page qui n'était pas demandé dans le cahier des charges, un tableau de bord pour les administrateurs. Pour leur permettre d'accéder rapidement aux pages d'ajout de véhicule et d'expertise ainsi que leur afficher un aperçu des prochaines expertises et des derniers véhicules ajoutés. Enfin, une fois que j'étais satisfait des fonctionnalités, j'ai publié l'application sur un serveur sur l'infrastructure de l'ETML. En parallèle, j'ai écrit un guide de déploiement de l'application.

### CONCLUSION

En conclusion, ce projet m'aura permis d'accumuler des compétences dans la création d'application web en ASP.NET C#. L'ordre des tâches dans la planification a été respecté malgré le fait que la plupart des tâches m'aient pris moins de temps que planifié. Ce temps supplémentaire m'aura permis de faire des petites corrections qui perfectionnent l'application et la rendent plus agréable et intuitive.

## REPOSITORY GITHUB

Le repo du projet est disponible à l'adresse suivante :

<https://github.com/ederiazl/TPI-Gestion-de-parc-automobile-d-entreprise>

## TABLE DES ILLUSTRATIONS

Figure 1 Planification initiale .....	7
Figure 2 Schéma de navigation dans l'application .....	8
Figure 3 Schéma de base de données MLD.....	9
Figure 4 Maquette page authentification .....	9
Figure 5 Maquette page liste de véhicules.....	10
Figure 6 Maquette page liste d'expertises .....	10
Figure 7 Maquette page ajout de véhicule .....	11
Figure 8 Maquette page ajout d'expertise .....	11
Figure 9 Popup ajout d'un véhicule.....	12
Figure 10 Maquette page modification de véhicule .....	12
Figure 11 Maquette page modification d'expertise.....	13
Figure 12 Popup sélection modèle véhicule .....	13
Figure 13 Maquette page détail véhicule.....	14
Figure 14 Maquette page détail expertise .....	14
Figure 15 Maquette popup validation expertise.....	15
Figure 16 Page d'authentification de base .....	18
Figure 17 Page d'authentification modifiée .....	18
Figure 18 Gestion des utilisateurs .....	19
Figure 19 Gestion des rôles .....	19
Figure 20 Page de modification du mot de passe .....	20
Figure 21 Liste des véhicules vue administrateur .....	20
Figure 22 Page d'ajout d'un véhicule .....	21
Figure 23 Page de modification d'un véhicule .....	21
Figure 24 Page de détail d'un véhicule.....	22
Figure 25 Popup confirmation suppression véhicule .....	22
Figure 26 Confirmation suppression véhicule.....	22
Figure 27 Page liste des expertises vue administrateurs .....	24
Figure 28 Liste des expertises vue employé.....	24
Figure 29 Page ajout d'une expertise .....	25
Figure 30 Modal ajout d'une expertise .....	25
Figure 31 Page modification d'une expertise.....	26
Figure 32 Popup suppression d'une expertise .....	26
Figure 33 Popup validation expertise.....	26
Figure 34 Dropdown sélection marque.....	28
Figure 35 Dropdown sélection modèle .....	30
Figure 36 Tableau de bord .....	30
Figure 37 Schéma infrastructure .....	33
Figure 38 Planification initiale vs effective.....	34

## GLOSSAIRE

### API

« Interface de programmation applicative » est un ensemble d'objets de programmation qui des services à d'autres logiciels.

### ASP.NET

ASP.NET est un framework open source pour la création d'application et de service web en C#.

### ASP.NET IDENTITY

C'est un système de gestion des utilisateurs, rôles et authentification intégré à ASP.NET.

### CARDATA API

C'est l'API que j'utilise pour récupérer les marques et les modèles.

### CODE FIRST

C'est une approche de conception de base de données dans laquelle le modèle est utilisé pour générer la base de données.

### DATATABLES

DataTables est un plugin JQuery permettant de créer des tableaux avancés avec des fonctions de tri, de filtrage et de pagination.

### ENTITY FRAMEWORK

Entity Framework est un ORM (Object-Relational Mapping) permettant de manipuler une base de données via des objets C#.

### MVC

Modèle vue contrôleur, c'est une architecture logicielle qui sépare les données (modèle), l'affichage (vue) et la logique (contrôleur).

### NGINX

Serveur reverse proxy que j'utilise pour rediriger les requêtes vers mon application web.

### REVERSE PROXY

C'est un serveur qui redirige les requêtes http vers d'autres serveurs.

### SWEETALERT2

SweetAlert2 est une librairie JS pour créer de jolis popups et facilement personnalisables.

### SEEDING

C'est le fait d'insérer des données au démarrage de l'application.

## SOURCES

- <https://www.youtube.com/watch?v=ESpp3uVmKhU&t=5s> pour le seeding de la base de données

- <https://learn.microsoft.com/en-us/aspnet/core/mvc/models/file-uploads?view=aspnetcore-7.0> pour l'ajout d'image aux véhicules
- <https://stackoverflow.com/questions/49865677/display-image-from-byte-array-in-asp-net-mvc-core> pour afficher l'image du véhicule
- <https://rapidapi.com/principalapis/api/car-data> API pour la récupération des marques et modèles
- <https://www.freecodecamp.org/news/javascript-debounce-example/> pour le debounce de la fonction d'actualisation de la liste des modèles
- <https://medium.com/@ameer.dev22/deploy-asp-net-core-application-on-ubuntu-using-nginx-20f71d70cedb> pour le déploiement
- <https://ubuntuhandbook.org/index.php/2024/11/install-net-9-or-8-ubuntu/> pour le déploiement
- <https://learn.microsoft.com/en-us/dotnet/core/install/linux-ubuntu-install?tabs=dotnet9&pivots=os-linux-ubuntu-2404> pour le déploiement
- <https://stackoverflow.com/questions/64518949/data-annotation-for-iformfile-so-it-only-allows-files-with-png-jpg-or-jpeg-e> pour le contrôle du type de fichier