



Rapport de Stage

**STAGE EFFECTUÉ DU 02 DÉCEMBRE 2024
AU 24 JANVIER 2025**

**LYCÉE POLYVALENT TURGOT PARIS 75003 BTS SIO
OPTION
SLAM (SOLUTIONS LOGICIELLES ET APPLICATIONS
MÉTIERS)**

**TUTEUR ACADEMIQUE: M.LE GUERN
TUTEUR DU STAGE : M.TABA'A**

SOMMAIRE

Introduction et remerciements :	3
Présentation de l'entreprise Digital Car :	
• Contexte et secteur d'activité	4
Présentation de stage :	5
Description du projet:	7
Organisation et Méthodologie de travail :	8
Mes contributions et le processus de développement:	9
Formulaire pour l'upload des fichiers docx et pdf:	12
Modèle conceptuel de données de l'application avec Vue.js (frontend) et node.js +	
Express + PostgreSQL (backend):	15
Évolution du projet et bascule vers le framework Django:	16
Mise en œuvre de la nouvelle solution:	17
Rôles des utilisateurs :	21
Liens vers les ressources du projet avec le backend Django :	22
Publication sur OVH :	23
Bilan personnel du stage:	24

1. Introduction et Remerciements

Dans le cadre de ma deuxième année de BTS Services Informatiques aux Organisations, option Solutions Logicielles et Applications Métier (SLAM), j'ai effectué un stage de 8 semaines au sein de l'entreprise Digital Car, spécialisée dans le calibrage des systèmes d'aide à la conduite.

Ce stage avait pour objectif de me permettre de mettre en pratique les connaissances acquises au cours de ma formation, tout en développant de nouvelles compétences techniques et professionnelles.

La principale mission qui m'a été confiée consistait à concevoir et développer une application Full Stack interne à l'entreprise, destinée aux techniciens et aux administrateurs. Cette application vise à faciliter la gestion des sélections de marques, modèles, et composants ADAS, ainsi qu'à automatiser l'affichage des procédures associées sous forme de documents PDF.

Je tiens à remercier mon tuteur, Amine TABAA et mon professeur M.Le Guern pour leurs conseils et son accompagnement tout au long du stage.

2. Présentation de Digital Car

1. Historique et Développement

Digital Car a été fondée dans le but de répondre à une demande croissante en services de calibrage pour les systèmes ADAS. Depuis sa création, l'entreprise n'a cessé de se développer et d'innover pour répondre aux besoins des constructeurs automobiles et des ateliers de réparation.

Parmi les étapes marquantes de son évolution :

- Crédit : (Insère ici l'année de création de l'entreprise).
- Développement des services : Adoption des dernières technologies de calibrage ADAS.
- Expansion : Couverture de plus de 95 % des véhicules disponibles sur le marché.

Expansion géographique

Aujourd'hui, Digital Car est présente sur plus de 70 % du territoire français, avec une équipe de plus de quarante techniciens répartis à travers le pays. Cette présence nationale permet de répondre efficacement aux besoins des réparateurs et des donneurs d'ordre de la réparation automobile.



3. Équipe dirigeante

Voici les membres clés de l'équipe dirigeante de Digital Car :

- **Jean-Christophe Canavesio** : Gérant - Associé
• (Email : jccanavesio@digitalcar.fr)
- **Amine Tabaa** : Ingénieur Automobile Spécialiste ADAS - Associé
• (Email : atabaa@digitalcar.fr)
- **Benjamin Everaere** : Associé
• (Email : beveraere@digitalcar.fr)

2. Présentation du stage

1. Objectifs du stage

Mon stage chez Digital Car avait pour objectif principal de développer mes compétences en développement Full Stack tout en contribuant à la création d'une application dédiée à la gestion des procédures ADAS.

Ce stage avait également pour but de me familiariser avec :

- La gestion de projets dans un environnement professionnel.
- Les méthodologies de travail utilisées en entreprise, notamment Agile.
- L'utilisation des outils collaboratifs pour optimiser la productivité.

Missions confiées

- Ma principale mission consistait en la conception et le développement d'une application interne destinée aux techniciens et aux administrateurs de Digital Car. Cette application vise à simplifier et centraliser la gestion des procédures liées aux systèmes ADAS."
- Ajouter un aperçu des autres missions que tu as pu réaliser.



Choix des technologies

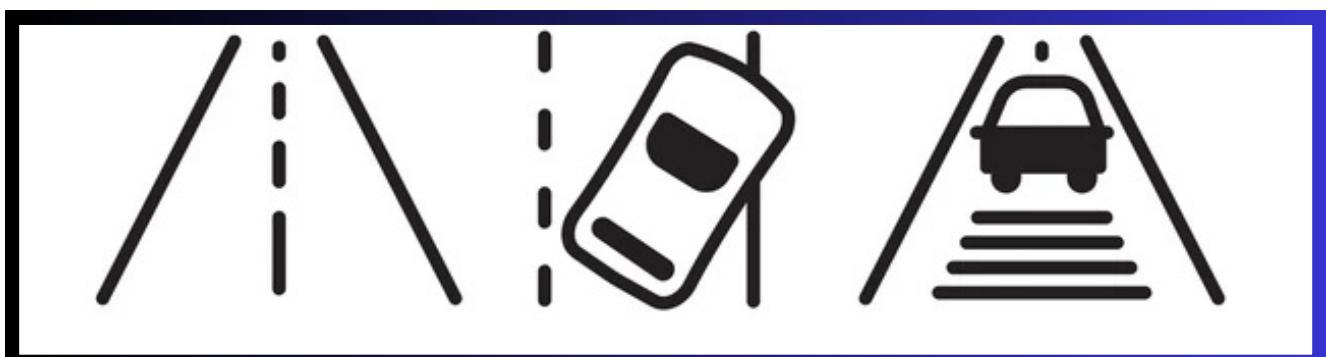
Pendant mon stage, j'ai été amené à utiliser les outils suivants :

- **Développement - Frontend :** Vue.js, Bootstrap
- **Développement - Backend :** Node.js, Express.js, Django
- **Base de données :** PostgreSQL
- **Environnement de développement :** Visual Studio Code et SQL Shell (psql)
- **Gestion des versions :** GitHub

Pour le frontend, nous avons longuement hésité entre React et Vue.js, deux frameworks modernes et performants. Après réflexion et des tests réalisés par mon camarade, nous avons choisi Vue.js, en grande partie pour sa facilité d'apprentissage et sa documentation claire, ce qui me permettait de progresser rapidement.

Pour le backend, nous avons opté pour Node.js avec Express et PostgreSQL, car ces technologies répondaient parfaitement aux besoins de l'application :

- Node.js + Express offraient une grande flexibilité pour développer une API robuste et rapide.
- PostgreSQL permettait une gestion efficace des données relationnelles avec un haut niveau de fiabilité.



3. Description du projet

Développement initial de l'application avec Vue.js et Node.js

Durant mon stage, mon camarade et moi avions pour mission de développer une application full stack destinée à un usage interne dans l'entreprise. L'objectif principal était de faciliter le travail des techniciens et de permettre à un administrateur de gérer les documents nécessaires à leur activité. L'application devait être composée de plusieurs pages, notamment :

- Une page d'accueil pour présenter l'application.
- Une page formulaire permettant de sélectionner une marque, un modèle associé, puis un composant spécifique. À partir de cette sélection, un bouton "Afficher le PDF" permettait d'accéder directement au document de procédure correspondant.

La page formulaire était centrale pour notre projet, car elle offrait une solution rapide et intuitive aux techniciens. Lors de mon premier jour de stage, mon tuteur m'a montré une première version de ce formulaire qu'il avait créée avec Python et des fichiers JSON. Cette version servait de point de départ pour définir nos objectifs et répondre aux besoins exprimés dans le cahier des charges.

Exigences du cahier des charges

Mon camarade et moi avons rédigé le cahier des charges en collaboration avec notre tuteur, qui a spécifié plusieurs attentes importantes :

1. L'application devait être responsive pour s'adapter à différents écrans.
2. Les mots de passe devaient être hachés pour garantir la sécurité.
3. Il fallait implémenter une gestion stricte des rôles (administrateur et technicien).
4. La gestion de l'authentification devait être robuste et sécurisée.
5. L'application devait être hautement sécurisée pour prévenir les attaques et protéger les données.
6. Nous avons également proposé d'ajouter une dimension d'accessibilité numérique pour rendre l'application utilisable par un plus large public.

En complément, notre tuteur souhaitait pouvoir :

- Ajouter des documents PDF ou DOCX depuis l'interface administrateur.
- Modifier directement les documents DOCX, avec une conversion automatique en PDF pour mise à jour immédiate dans l'interface technicien.

Mon tuteur m'a ensuite précisé que nous étions libres d'utiliser n'importe quel framework ou outil, à condition de répondre à ses exigences, ce qui lui convenait parfaitement.

Lien du document pdf des cahiers des charges :
https://raw.githubusercontent.com/ton_user/pdfs_projet_stage/main/dependances/list_dependencies.pdf

Liens vers les ressources du projet avec le backend Node.js + Express + PostgreSQL

- **Liste des dépendances et packages installés - Contient la liste des dépendances et packages utilisés dans le projet, avec leur rôle respectif :**
https://raw.githubusercontent.com/nirzara13/pdfs_stage_digital_car/main/dependances/Document_expliquatif_des_dependances_Vue.js_Express.pdf
- **Schéma applicatif du projet :**
https://raw.githubusercontent.com/nirzara13/pdfs_stage_digital_car/main/Mon%20Schema%20Applicatif%20de%20l'application%20Vue.js%20%2B%20Express.pdf
- **Tableau comparatif pour choisir entre React et Vue.js :**
https://raw.githubusercontent.com/nirzara13/pdfs_stage_digital_car/main/tableau_comparatifs/Tableau_comparatif_de%20_framework_2.pdf

Organisation et Méthodologie de travail

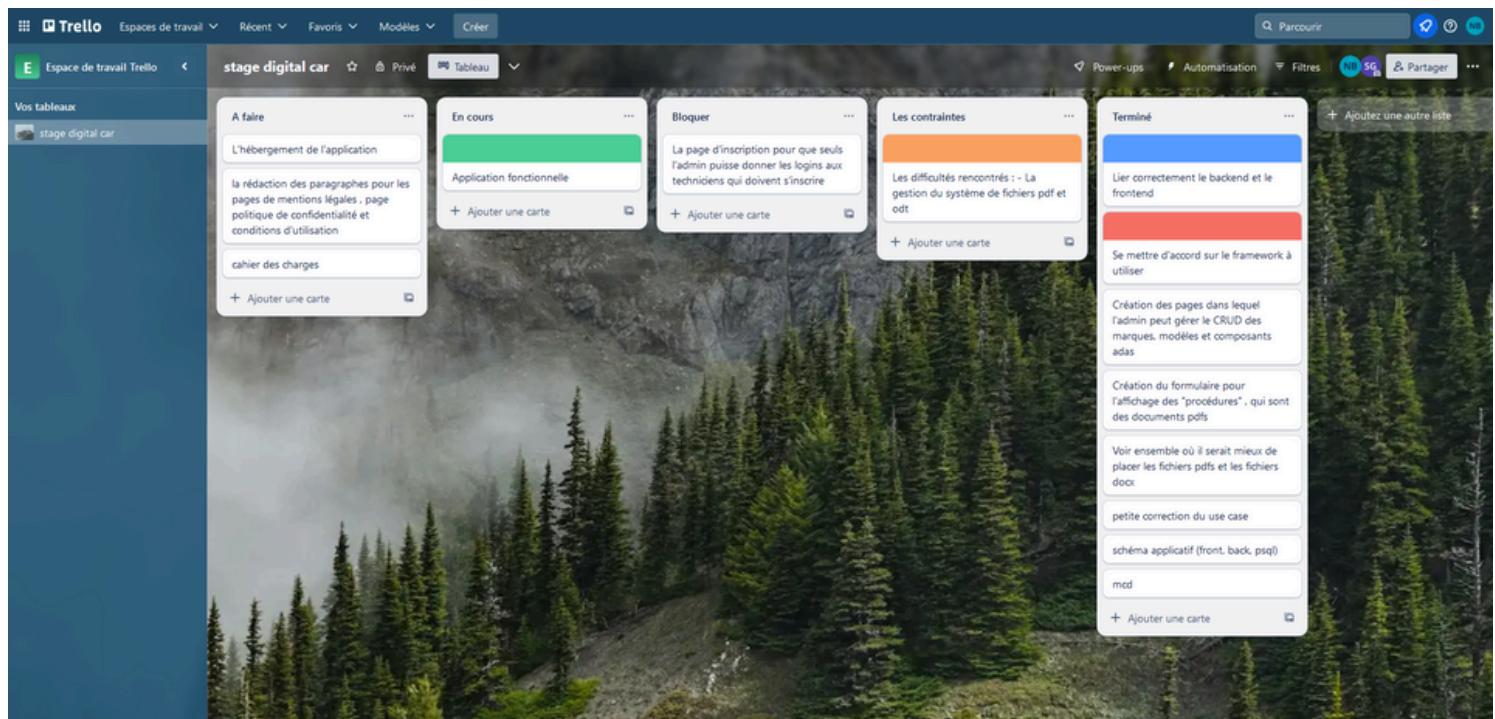
Méthodologie

- Organisation du travail en sprints hebdomadaires.
- Priorisation des tâches via des outils de gestion comme Trello
- Retours réguliers lors des réunions d'équipe pour ajuster les objectifs.
- Outils de gestion de projets :
- GitHub pour la gestion des versions du code.
- Slack pour la communication et la collaboration.
- Postman pour les tests d'API.
- Notion pour le suivi des documents et des tâches.

Répartition des tâches :

Nous avons réparti les tâches de manière équilibrée : si je travaillais sur le backend, mon camarade se concentrait sur le frontend, et inversement. Nous nous corrigions mutuellement et collaborions étroitement, notamment lors d'appels réguliers pour résoudre les problèmes rencontrés. Cette méthode nous a permis d'avancer rapidement tout en assurant une bonne qualité de code.

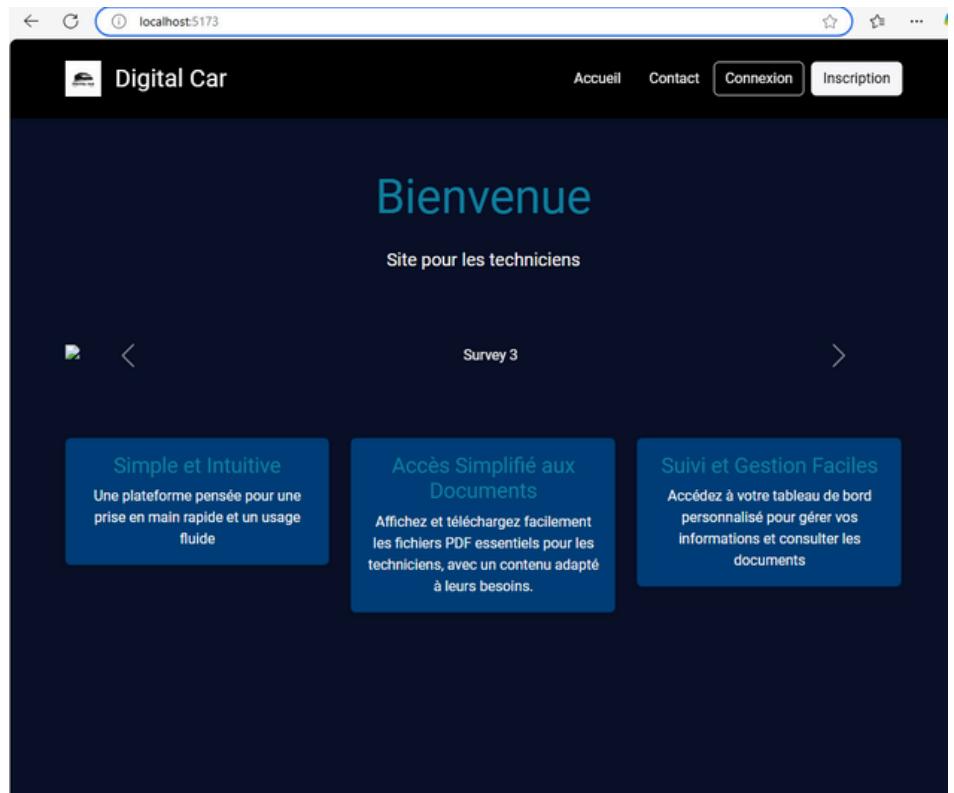
Outils de gestion de projets -Trello



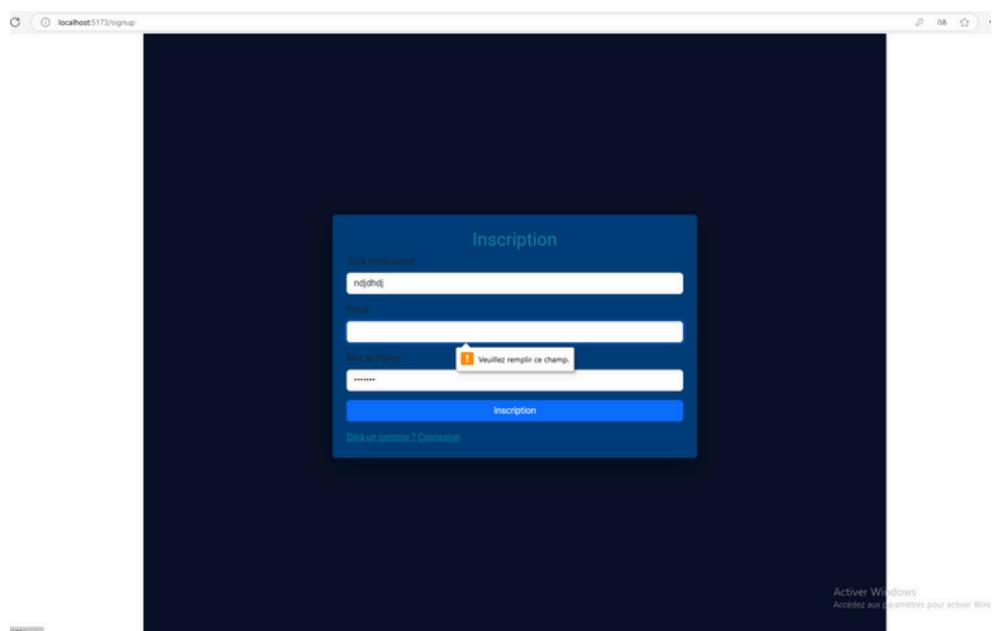
Mes contributions et le processus de développement

Processus de développement

Mon tuteur ne prêtait pas une attention particulière au design CSS, ce qui m'a permis de me concentrer sur la fonctionnalité. J'ai donc commencé par développer cette page d'accueil. Nous n'avons pas réalisé de maquettes préalables, car j'étais en phase de découverte d'un nouveau framework, réputé pour sa simplicité et son efficacité.



Après avoir développé une page d'inscription et une page de connexion fonctionnelles, j'ai intégré un système de hachage pour sécuriser les mots de passe. Lorsqu'un utilisateur (technicien) s'inscrit, il est automatiquement redirigé vers la page de connexion. Une fois connecté, l'utilisateur est orienté soit vers son tableau de bord, soit vers la page dédiée au formulaire de sélection des marques, modèles et composants avec l'affichage des documents PDF associés.



Voici l'arborescence de mon projet sur Visual Studio Code

```
+-----+-----+-----+
| public | Users | table | user1 |
(1 ligne)

dcdb=# SELECT * FROM users;
ERREUR: la relation « users » n'existe pas
LIGNE 1 : SELECT * FROM users;
^

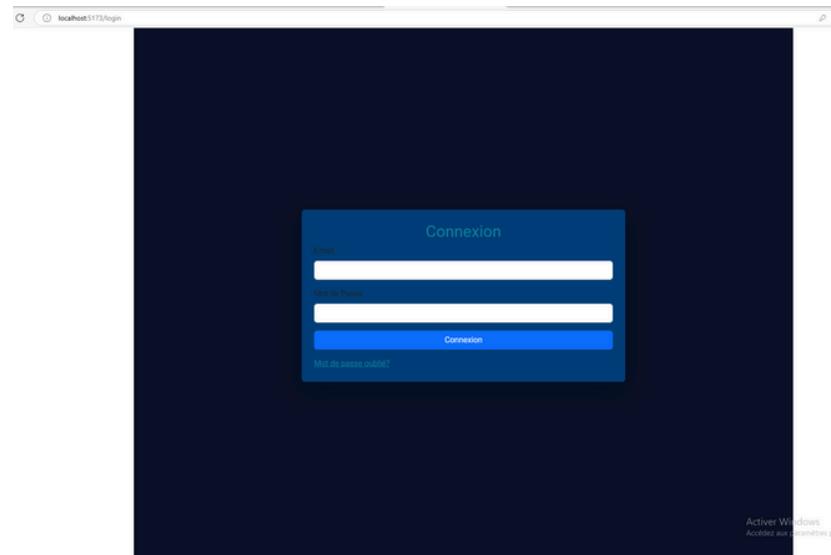
dcdb=# SELECT * FROM Users;
ERREUR: la relation « users » n'existe pas
LIGNE 1 : SELECT * FROM Users;
^

dcdb=# SELECT * FROM "Users";
 id | username |           email           |          password          |      createdAt
    |          |           updatedAt          |          |
+-----+-----+-----+-----+-----+
  1 | emmal   | emma123@gmail.com | $2a$10$ld8iwRq7kjNVNFD8h7hEf0agXTwQb4C.S11YlCzy/jgJbF5BZBAC. | 2024-12-06 16:19:34
.345+01 | 2024-12-06 16:19:34.345+01
  2 | sarah   | sarah123@gmail.com | $2a$10$UkllLU0DSB3oq/z.2hHruvSSU4qertUu2hB36bAGJX2KA13oI8T2 | 2024-12-06 16:23:28
.114+01 | 2024-12-06 16:23:28.114+01
  3 | clara12 | clara12@gmail.com | $2a$10$X3YIrBsAc0ZCkLzGFj53A0rQSIexcNupBE/JsbAjqkJN7wCL/58G | 2024-12-06 16:26:31
.777+01 | 2024-12-06 16:26:31.777+01
  4 | clara14 | clara14@gmail.com | $2a$10$XLxcK0Xq5YY8.jx/oPjRg0SCEz9nIcT/yf0YQuaEI6njU4f8r0Ofa | 2024-12-06 16:36:47
.702+01 | 2024-12-06 16:36:47.702+01
(4 lignes)

dcdb=#

```

Pour gérer les bases de données et les tables, j'ai utilisé SQL Shell (psql). Comme le montre l'exemple ci-dessous, les mots de passe des utilisateurs sont correctement hachés, garantissant ainsi une meilleure sécurité des données.



Lors de la connexion au formulaire, voici un aperçu du **backend** affiché dans le terminal de Visual Studio Code. On peut y voir les différentes données exécutées et traitées en temps réel

Utilisation et test avec Postman.

Pour tester la connexion avec l'API et assurer son bon fonctionnement, j'ai utilisé Postman, mais seulement au début du stage. Cet outil m'a permis de simuler des requêtes HTTP telles que GET et POST, ce qui est essentiel dans une application full stack. Par exemple, on peut envoyer des données à un serveur pour les enregistrer dans une base de données via une requête POST, ou récupérer des informations avec une requête GET pour vérifier que l'API renvoie les bonnes données. Postman m'a aidé à interagir avec le backend en envoyant des données et en récupérant des informations pour m'assurer que les échanges entre le frontend et le backend se déroulaient correctement. Grâce à cet outil, j'ai pu vérifier que l'API répondait comme prévu aux différentes requêtes, garantissant ainsi le bon déroulement des fonctionnalités de l'application.

The screenshot shows the Postman interface with a successful API call. At the top, the URL is set to `http://localhost:5000/login` and the method is `POST`. The `Body` tab is selected, showing raw JSON data:

```
1 {
2   "username": "testuser",
3   "password": "password123"
4 }
```

Below the request, the response status is `200 OK`, with a response time of `100 ms` and a size of `481 B`. The response body is displayed in the `Pretty` tab:

```
1 {
2   "message": "Connexion réussie"
3 }
```

J'ai également mis en place une fonctionnalité permettant à l'admin de télécharger des documents .docx, qui sont ensuite automatiquement convertis en PDF et stockés dans un dossier spécifié. J'ai testé cette fonctionnalité avec Postman en envoyant une requête POST pour simuler l'upload du fichier et vérifier que l'API gérait correctement le processus.

The screenshot shows the Postman interface with a successful API call. At the top, the URL is set to `http://localhost:3000/api/upload` and the method is `POST`. The `Body` tab is selected, showing form-data:

Key	Value	Description
<input checked="" type="checkbox"/> file	File	Prompt_pour_bolt...
Key	Text	Description

Below the request, the response status is `200 OK`, with a response time of `26 ms` and a size of `425 B`. The response body is displayed in the `Pretty` tab:

```
1 {
2   "message": "Fichier uploadé avec succès.",
3   "filePath": "uploads\\1734436468018.odt"
4 }
```

Formulaire pour l'upload des fichiers docx et pdf.

Sélection des Marques et Composants

Marques :

Toyota ▾

Modèles :

Corolla ▾

Composants ADAS :

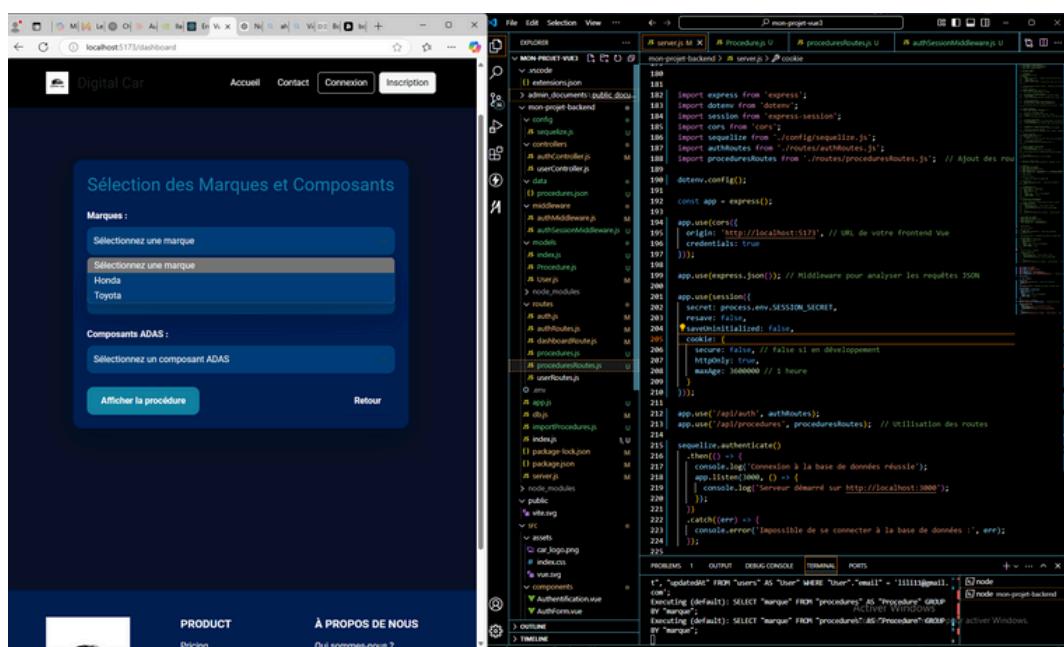
Radar avant ▾

Uploader un PDF

Retour

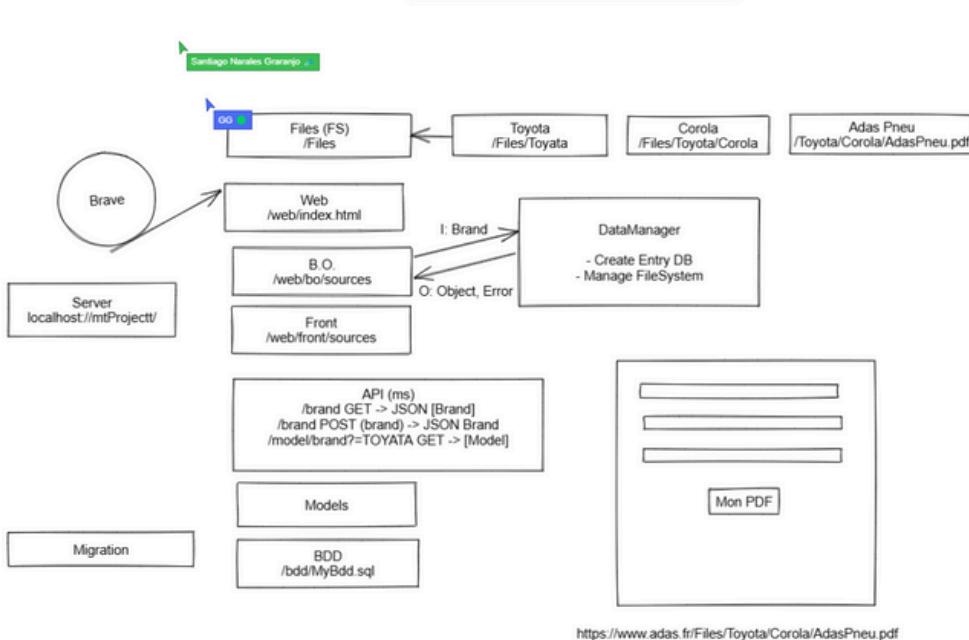
J'ai créé une page permettant à l'admin de télécharger des documents au format .docx. Une fois le fichier téléchargé, il est automatiquement placé dans le dossier uploads avec un chemin et un numéro de fichier généré. Ensuite, j'ai implémenté une requête POST vers l'URL du backend api/convert pour convertir le fichier .docx en PDF. Le document converti est ensuite enregistré dans le dossier pdf que j'avais préalablement spécifié dans le code.

Formulaire d'affichage des pdfs



Lors de mes échanges avec un développeur senior iOS, il m'a expliqué, à l'aide d'un schéma, comment afficher le PDF correspondant à une marque, un modèle et un composant sélectionnés par un technicien. L'idée a été de créer un dossier pour chaque marque, par exemple "Toyota", et un sous-dossier pour chaque modèle, comme "Corolla". Ensuite, nous avons renommé les fichiers PDF selon le nom des composants, par exemple "Adaptive Cruise Control.pdf". L'objectif était de mettre en place un chemin permettant d'accéder directement au fichier PDF en fonction des critères de sélection du technicien.

Schéma dessiné par le développeur senior IOS



```
dcdb=# SELECT * FROM adas_components;
 id |          name          | model_id |           file_path
---+---------------------+-----+-----
 1 | Adaptive Cruise Control |      1 | adaptive_cruise_control.pdf
(1 ligne)
```

Lors de l'ajout d'un composant dans le formulaire, j'ai testé l'insertion d'un fichier PDF dans la table adas_components, en y enregistrant le chemin du fichier dans la colonne filePath. Lors de la saisie du nom du composant, comme par exemple "Lane_Departure_Warning", le système était capable de retrouver le chemin et d'afficher le PDF correspondant. Cependant, lorsqu'il n'y avait pas d'insertion dans la table et que le fichier PDF était directement placé dans le dossier et sous-dossier correspondant, j'ai constaté un comportement inattendu : en remplaçant le champ du composant sans utiliser de tirets (par exemple "LaneDepartureWarning" au lieu de "Lane_Departure_Warning"), le PDF ne s'affichait pas correctement. Cela a montré que le nom du composant, y compris les tirets, était essentiel pour que le système puisse retrouver et afficher le fichier PDF associé.

The screenshot shows a developer's environment with multiple windows open:

- Browser:** Shows a dashboard with sections for Marque (audi), Modèle (a1), Composant ADAS (lane_departure_warning), and a search bar.
- Code Editor:** Displays a component template named "Dashboardview.vue" with HTML and CSS code for selecting a brand, model, and component, and a file input field for PDF uploads.
- Terminal:** Shows a MySQL command-line interface (dcdb) displaying the contents of the "adas_components" table, which includes an entry for "Adaptive Cruise Control" with file path "adaptive_cruise_control.pdf".
- Bottom Navigation:** Includes links for PRODUCT (Pricing, Changelog, Terms), À PROPOS DE NOUS (Qui sommes-nous ?, Contact, Nous rejoindre), and footer links for © 2024 Digital Car, All rights reserved.

Voici un aperçu des exécutions qui se déroulaient en temps réel dans le terminal de Visual Studio Code (VSC) lors du projet. Ces logs montrent les actions exécutées par le backend et les processus en cours, tels que la gestion des requêtes, les recherches de fichiers et les traitements effectués sur les données. Ce retour en temps réel était essentiel pour valider le bon fonctionnement du processus d'affichage des PDF et de la gestion des composants.

```

Marque : audi
Modèle : a1
Composant : lane_departure_warning
Chemin du dossier recherché : C:\Users\Amine Perso\mon-projet-vue3\mon-projet-backend\files\audi\a1
Fichiers trouvés : [
{
  file_name: 'lane_departure_warning.pdf',
  file_url: 'http://localhost:3000/files/audi/a1/lane_departure_warning.pdf'
}
]
Requête reçue avec les paramètres :
Marque : audi
Modèle : a1
Composant : lane departure warning
Chemin du dossier recherché : C:\Users\Amine Perso\mon-projet-vue3\mon-projet-backend\files\audi\a1
Fichiers trouvés : []
Requête reçue avec les paramètres :
Marque : audi
Modèle : a1
Composant : lane departure warning
Chemin du dossier recherché : C:\Users\Amine Perso\mon-projet-vue3\mon-projet-backend\files\audi\a1
Fichiers trouvés : []
Requête reçue avec les paramètres :
Marque : audi
Modèle : a1
Composant : lane departure warning
Chemin du dossier recherché : C:\Users\Amine Perso\mon-projet-vue3\mon-projet-backend\files\audi\a1
Fichiers trouvés : []
{
  file_name: 'lane_departure_warning.pdf',
  file_url: 'http://localhost:3000/files/audi/a1/lane_departure_warning.pdf'
}
]

```

J'ai effectué de nombreux tests et exploré différentes alternatives pour implémenter correctement cette fonctionnalité, car elle était cruciale pour le projet à ce moment-là. L'objectif était de garantir que le processus de recherche et d'affichage des PDF fonctionne de manière fluide et sans erreurs, ce qui a demandé plusieurs ajustements et expérimentations jusqu'à obtenir le résultat attendu.

Branches GitHub – Collaboration sur le projet

The screenshot shows the GitHub repository 'Digital-Car-vue3'. At the top, it displays '3 Branches' and '0 Tags'. On the left, there's a sidebar for 'Switch branches/tags' with options for 'main', 'nirzara', and 'santiago'. Below this are lists for '.utils', '.ignore', 'README.md', 'index.html', 'package-lock.json', 'package.json', 'tailwind.config.js', and 'vite.config.js'. The main area shows a list of commits for the 'main' branch:

- 9382212 - 2 days ago (4 Commits)
 - Premier commit - projet Vue.js version 3 (last week)
 - Ajout des nouveaux fichiers (2 days ago)
 - Premier commit - projet Vue.js version 3 (last week)
 - Ajout des nouveaux fichiers (2 days ago)
- 9382212 - 2 days ago (4 Commits)
 - Ajout des nouveaux fichiers (2 days ago)
 - Résolution des conflits (last week)
 - Premier commit - projet Vue.js version 3 (last week)
 - Ajout des nouveaux fichiers (2 days ago)
- 9382212 - 2 days ago (4 Commits)
 - Ajout des nouveaux fichiers (2 days ago)
 - Premier commit - projet Vue.js version 3 (last week)
 - Ajout des nouveaux fichiers (2 days ago)
 - Premier commit - projet Vue.js version 3 (last week)

On the right side, there are sections for 'About' (No desc), 'Release' (No release), 'Package' (No packag), 'Language' (Vue 4, CSS 6), and 'Suggest' (Based on).

Voici un aperçu des branches sur lesquelles j'ai collaboré avec mon camarade de stage. La branche principale était main, et nous avons créé chacun une branche à notre nom pour travailler sur nos fonctionnalités respectives. Nous avons effectué nos commits et pushes sur nos branches respectives. Par la suite, nous avons décidé de fusionner nos codes lorsque nous aurions terminé nos développements.

Modèle conceptuel de données de l'application avec Vue.js (frontend) et node.js + Express + PostgreSQL (backend)

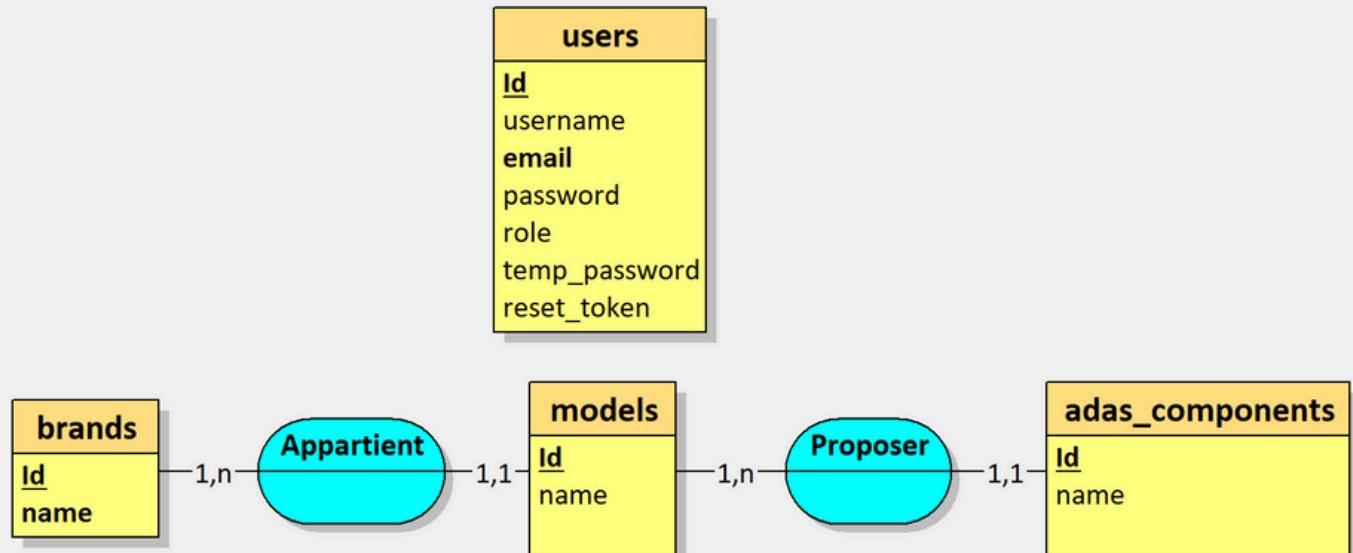
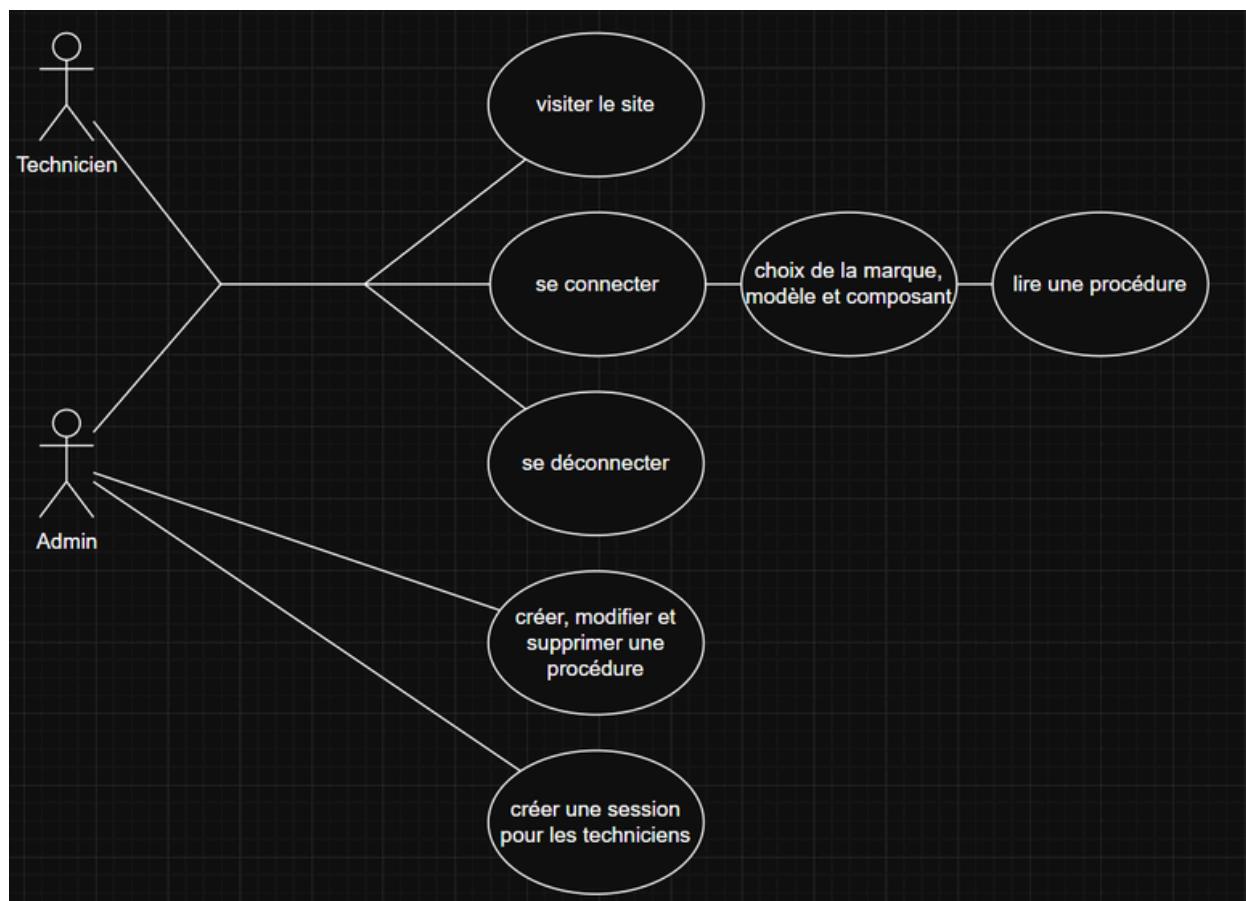


Diagramme de Use case



Évolution du projet et bascule vers le framework Django

Contexte des nouvelles exigences

Après avoir obtenu l'aide d'un camarade de classe travaillant sur un autre projet, j'ai organisé une réunion avec mon tuteur pour lui présenter les fonctionnalités que nous avions déjà développées. Bien qu'il ait apprécié notre travail, il a exprimé de nouveaux besoins fonctionnels qui n'étaient pas prévus initialement. Il souhaitait que, lorsqu'il se connecte en tant qu'administrateur sur le frontend, seul lui puisse accéder à certaines fonctionnalités spécifiques. Par exemple, les boutons "Marques", "Modèles", "Composants ADAS" et "Charger PDF" devaient être accessibles uniquement pour l'admin afin de lui permettre de gérer ces données.

De plus, il voulait revoir la logique de gestion des techniciens inscrits :

- L'administrateur devait avoir la possibilité de créer des identifiants et des mots de passe provisoires, qu'il enverrait aux techniciens. Ces derniers recevraient un code par mail pour une authentification à deux facteurs lors de leur première connexion.
- Une alternative était de conserver le formulaire d'inscription tel quel, mais avec un système où chaque inscription serait soumise à validation par l'administrateur. Une fois validée, le technicien recevrait un mot de passe provisoire par mail et pourrait ensuite le réinitialiser.
- Il a également demandé la création d'un tableau de bord pour chaque technicien. Ce dernier devait pouvoir modifier ses informations personnelles, tout en garantissant que l'administrateur reçoive une notification par mail de ces modifications, sans divulguer les détails des données modifiées. Enfin, pour une meilleure gestion des rôles, les techniciens connectés ne devaient voir que les boutons essentiels dans le menu (comme "Accueil", "Connexion", "Inscription", "Afficher un PDF" et "Contact"). Ces nouvelles exigences nous ont conduit à réévaluer notre choix initial de framework backend.

Pourquoi avoir choisi de basculer vers Django ?

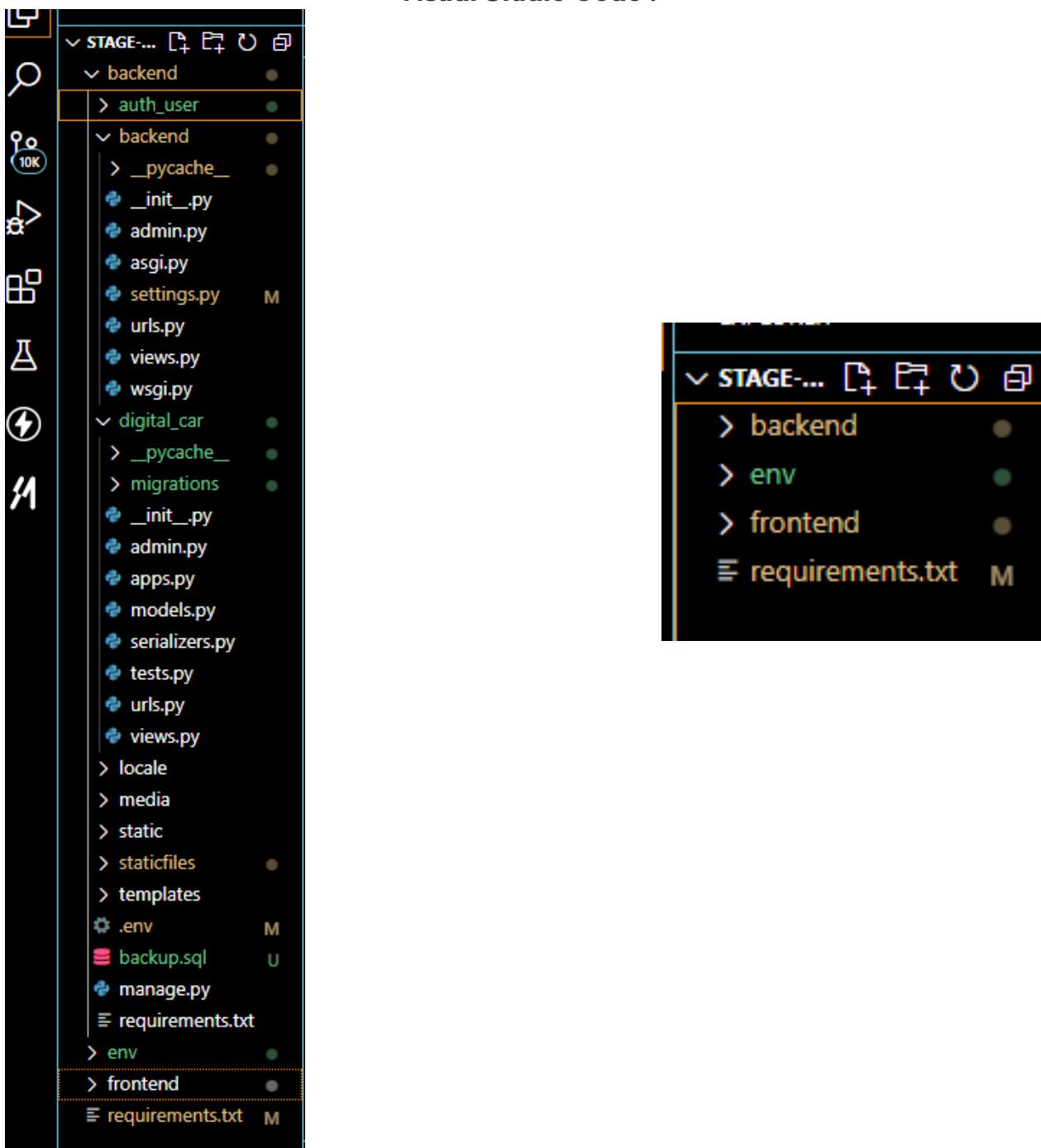
Suite à la réunion avec mon tuteur, j'ai discuté des nouvelles fonctionnalités demandées avec mes camarades. L'un d'eux m'a expliqué qu'il avait déjà développé une application avec des fonctionnalités similaires en utilisant Django. Cela nous a ouvert la possibilité de réutiliser certaines parties de son backend Django et de les adapter à notre projet avec Vue.js.

Étant donné que mon tuteur souhaitait présenter rapidement l'application à son entreprise et la publier sur Internet dans les meilleurs délais, nous avons décidé de changer de framework backend et d'opter pour Django. Cette solution nous permettait de répondre plus efficacement aux nouvelles exigences tout en optimisant le temps de développement.

Mise en œuvre de la nouvelle solution

Mon camarade de classe, qui nous a aidés, nous a présenté à moi et mon camarade de stage le code qu'il allait intégrer grâce au framework Django. Il avait déjà configuré une interface d'administration dans le backend. Lors d'un appel visio, il a ajouté les nouvelles fonctionnalités demandées par notre tuteur. Comme nous avions tous les trois accès au projet via GitHub, j'ai continué à travailler sur l'application en local sur mon PC.

Voici l'arborescence du projet sur Visual Studio Code :



Pour travailler avec Django, il est nécessaire de lancer un environnement virtuel. J'ai utilisé la commande suivante sur mon système Windows car la commande Linux/MacOS ne fonctionnait pas :

C:\Users\User\AppData\Local\Programs\Python\Python312\python.exe -m venv env.

Un environnement virtuel permet d'isoler les dépendances spécifiques d'un projet Python, pour éviter les conflits entre différents projets ou versions de bibliothèques.

Ensuite, j'ai activé l'environnement virtuel avec cette commande :

```
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs (main)
● $ source env/Scripts/activate
(env)
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs (main)
○ $ []
```

Dans le projet, il était nécessaire d'installer les dépendances spécifiées dans le fichier requirements.txt.

Cela se fait avec la commande :

pip install -r requirements.txt

Cette commande lit le fichier requirements.txt, qui liste toutes les bibliothèques nécessaires au projet, et installe automatiquement ces bibliothèques.

Après avoir configuré les dépendances, j'ai effectué une migration avec la commande :

python manage.py migrate
Cette commande applique les modifications des modèles Python à la base de données, comme la création ou la mise à jour des tables.
pip install -r requirements.txt

Cette commande lit le fichier requirements.txt, qui liste toutes les bibliothèques nécessaires au projet, et installe automatiquement ces bibliothèques.

```
(env)
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs/backend (main)
● $ python manage.py migrate
Operations to perform:
  Apply all migrations: account, admin, auth, auth_user, auth_token, contenttypes, digital_car, sessions, sites
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002.Alter_permission_name_max_length... OK
  Applying auth.0003.Alter_user_email_max_length... OK
  Applying auth.0004.Alter_user_username_opts... OK
  Applying auth.0005.Alter_user_last_login_null... OK
  Applying auth.0006.Require_contenttypes_0002... OK
  Applying auth.0007.Alter_validators_add_error_messages... OK
  Applying auth.0008.Alter_user_username_max_length... OK
  Applying auth.0009.Alter_user_last_name_max_length... OK
  Applying auth.0010.Alter_group_name_max_length... OK
  Applying auth.0011.Update_proxy_permissions... OK
  Applying auth.0012.Alter_user_first_name_max_length... OK
  Applying auth_user.0001_initial... OK
  Applying account.0001_initial... OK
  Applying account.0002_email_max_length... OK
  Applying account.0003.Alter_emailaddress_create_unique_verified_email... OK
  Applying account.0004.Alter_emailaddress_drop_unique_email... OK
  Applying account.0005_emailaddress_idx_upper_email... OK
  Applying account.0006_emailaddress_lower... OK
  Applying account.0007_emailaddress_idx_email... OK
  Applying account.0008_emailaddress_unique_primary_email_fixup... OK
  Applying account.0009_emailaddress_unique_primary_email... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying auth_user.0002_remove_userprofile_user_delete.usermodel_and_more... OK
  Applying auth_user.0003_initial... OK
  Applying auth_user.0004_remove.usermodel.username... OK
  Applying auth_user.0005.usermodel.username... OK
  Applying auth_user.0006.Alter.usermodel.managers.usermodel_approval_code_and_more... OK
  Applying auth_user.0007.Alter.usermodel.status... OK
  Applying auth_user.0008_remove.usermodel.approval_code_and_more... OK
  Applying auth_user.0009.usermodel.approval_code... OK
  Applying auth_token.0001_initial... OK
  Applying auth_token.0002_auto_20160226_1747... OK
```

Ensuite, j'ai exécuté la commande :

```
python manage.py
collectstatic
```

Elle collecte tous les fichiers statiques (CSS, JavaScript, images, etc.) utilisés dans l'application et les place dans un seul répertoire pour les servir plus facilement.

```
Amine Perso@DESKTOP-14FT85M MINGW64 ~/stage-vue/backend (main)
$ python manage.py collectstatic
● You have requested to collect static files at the destination
location as specified in your settings:
C:\Users\Amine Perso\stage-vue\backend\staticfiles
This will overwrite existing files!
Are you sure you want to do this?

Type 'yes' to continue, or 'no' to cancel: yes
Found another file with the destination path 'admin\js\cancel.js'. It will be ignored since only the first encountered file is collected. If this is not what you want, make sure every static file has a unique path.
Found another file with the destination path 'admin\js\popup_response.js'. It will be ignored since only the first encountered file is collected. If this is not what you want, make sure every static file has a unique path.

222 static files copied to 'C:\Users\Amine Perso\stage-vue\backend\staticfiles', 16 unmodified.
(env)
Amine Perso@DESKTOP-14FT85M MINGW64 ~/stage-vue/backend (main)
$
```

```
❖(env)
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs/backend (main)
$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
January 10, 2025 - 10:58:25
Django version 5.1.4, using settings 'backend.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

Not Found: /
[10/Jan/2025 10:58:32] "GET / HTTP/1.1" 404 10129
Not Found: /favicon.ico
[10/Jan/2025 10:58:32] "GET /favicon.ico HTTP/1.1" 404 10180
[10/Jan/2025 10:58:41] "GET /admin/login HTTP/1.1" 302 0
Could not reverse url from auth.user
[10/Jan/2025 10:58:41] "GET /admin/login/?next=/admin/login HTTP/1.1" 200 3462
[10/Jan/2025 10:58:42] "GET /static/vendor/Fontawesome-free/css/all.min.css HTTP/1.1" 200 183009
[10/Jan/2025 10:58:42] "GET /static/vendor/bootstrap/js/bootstrap.min.js HTTP/1.1" 200 62448
[10/Jan/2025 10:58:42] "GET /static/vendor/adminlte/js/adminlte.min.js HTTP/1.1" 200 44244
[10/Jan/2025 10:58:42] "GET /static/jazzmin/css/main.css HTTP/1.1" 200 16969
[10/Jan/2025 10:58:42] "GET /static/css/custom_admin.css HTTP/1.1" 200 0
[10/Jan/2025 10:58:42] "GET /static/vendor/bootstrap/darkly/bootstrap.min.css HTTP/1.1" 200 164202
[10/Jan/2025 10:58:42] "GET /static/admin/js/vendor/jquery/jquery.js HTTP/1.1" 200 285314
[10/Jan/2025 10:58:42] "GET /static/vendor/adminlte/css/adminlte.min.css HTTP/1.1" 200 1382975
[10/Jan/2025 11:00:11] "GET /static/vendor/bootstrap/darkly/bootstrap.min.css HTTP/1.1" 200 164202
[10/Jan/2025 11:00:11] "GET /static/admin/js/vendor/jquery/jquery.js HTTP/1.1" 200 285314
[10/Jan/2025 11:00:11] "GET /static/vendor/adminlte/img/AdminLTELogo.png HTTP/1.1" 200 2632
```

Enfin, pour démarrer le backend, j'ai lancé la commande :

```
python manage.py
runserver
```

Cette commande démarre un serveur local permettant de tester et d'interagir avec l'application en backend.

Pour accéder à l'interface d'administration de Django, j'ai créé un superutilisateur en utilisant la commande suivante :

```
python manage.py
createsuperuser
```

Cette commande permet de créer un compte administrateur avec des priviléges élevés pour gérer les données et les fonctionnalités du site directement depuis l'interface dédiée.

```
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs (main)
$ source env/Scripts/activate
(env)
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs (main)
$ cd backend/
(env)
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs/backend (main)
$ python manage.py createsuperuser
Email: admin@example.com
Nom d'utilisateur: admin
First name: Admin
Last name: Admin
Phone number: 0123456789
Password:
Password (again):
Le mot de passe est trop semblable au champ « nom d'utilisateur ».
Le mot de passe est trop court. Il doit contenir au minimum 8 caractères.
Le mot de passe est trop courant.
Bypass password validation and create user anyway? [y/N]: y
Creating superuser with status 'approved'...
Superuser created successfully.
(env)
Amine Perso@DESKTOP-14FT85M MINGW64 ~/app_django_vuejs/backend (main)
```

Une fois connecté avec ce compte, voici un aperçu de l'interface d'administration, où toutes les fonctionnalités demandées par le tuteur ont été correctement mises en place.

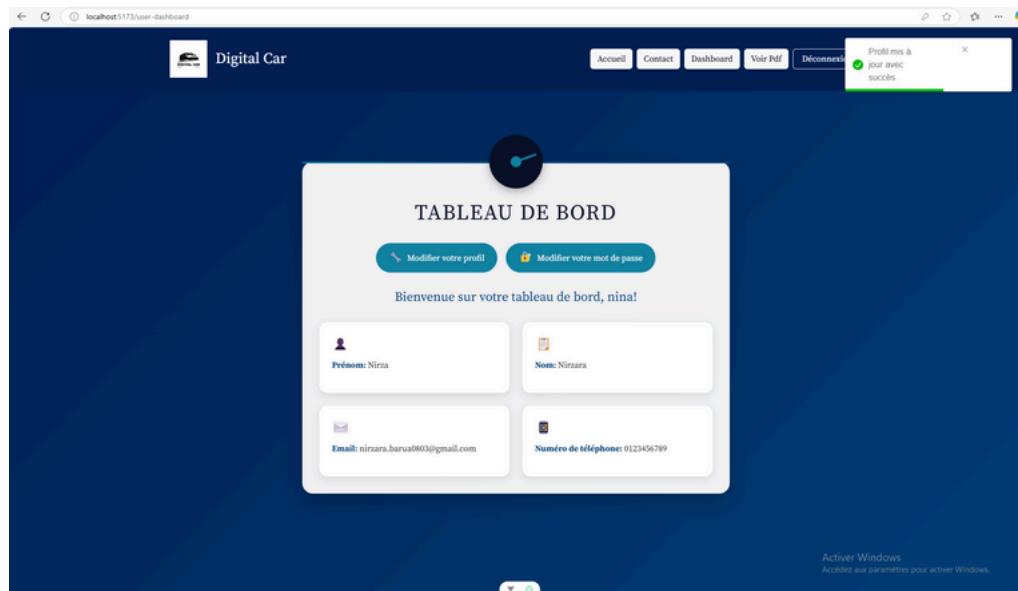
The screenshot shows the 'Tableau de bord' (Dashboard) of the 'Digital Car' application. The left sidebar contains navigation links for 'Authentification et autorisation', 'Auth_User', 'Comptes', 'Digital_Car', 'Jeton d'authentification', and 'Sites'. The main content area is divided into several sections: 'Authentification et autorisation' (Groups, Utilisateurs, Adresses e-mail), 'Digital_Car' (Adas components, Brands, Car models, Procedure pdfs), 'Actions récentes' (empty), 'Jeton d'authentification' (Jetons), and 'Sites' (Sites). Each section includes 'Ajouter' and 'Modification' buttons.

Je me suis aussi occupé du frontend, notamment du **CSS**, du design et des animations, pour rendre l'application plus agréable à utiliser. J'ai également fait en sorte que toutes les pages soient bien adaptées aux différents appareils (responsive). Une fois les modifications terminées, je lançais la commande **npm run build**, qui sert à créer une version optimisée et prête à être utilisée. Cette commande permet de compiler et optimiser le code frontend, générant ainsi une version prête pour la mise en production.

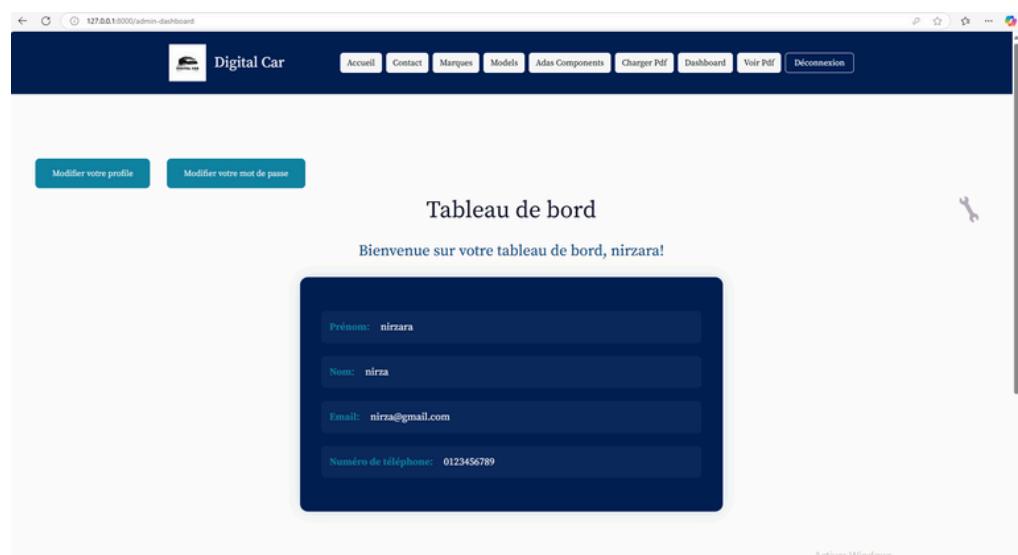
Rôles des utilisateurs

Les utilisateurs sont répartis en deux catégories : Technicien et Admin. En fonction du rôle attribué lors de la connexion, l'interface et les options disponibles varient.

Voici la navbar lorsque c'est un technicien qui se connecte :



Et voici celle d'un admin :

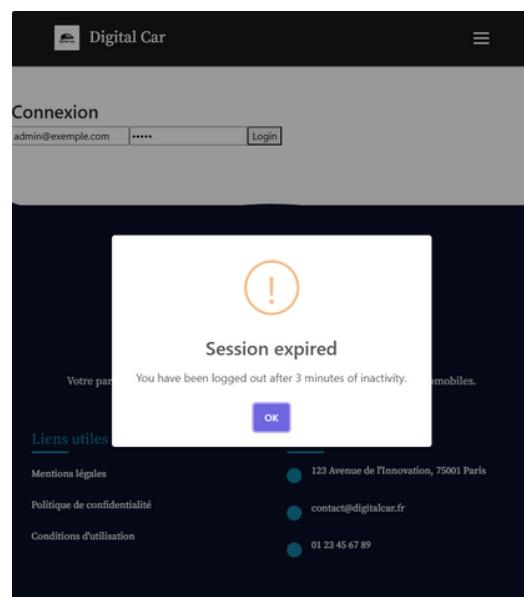


Gestion de la session

Le système utilise deux tokens :

- Access Token : Identifie l'utilisateur pour chaque requête, valide ses actions.
- Refresh Token : Permet de renouveler l'Access Token lorsque celui-ci expiré.

De plus, pour la sécurité, un Inactivity Timer est mis en place. Si l'utilisateur reste inactif pendant 3 minutes, il est automatiquement déconnecté. Si l'utilisateur reprend l'activité, le timer redémarre.



Liens vers les ressources du projet avec le backend Django

- **Liste des dépendances et packages installés :**
 - **Schéma applicatif du projet :** Lien vers le PDF sur github
-

Publication sur OVH

Après l'intégration des nouvelles fonctionnalités, nous allons publier l'application sur OVH. Et nous vérifierons que tout fonctionne correctement et que l'application est prête à être utilisée par l'entreprise.

1

2

3

4

Bilan personnel du stage

Compétences acquises :

C'était la première fois que je travaillais sur une application full-stack destinée à une utilisation en entreprise. Je me suis investi pleinement pour m'assurer que chaque fonctionnalité fonctionne correctement. J'ai également découvert et utilisé pour la première fois des technologies telles que Vue.js pour le frontend, Node.js avec Express et PostgreSQL pour le backend, avant de passer à Django. Cette expérience m'a permis d'acquérir des compétences pratiques et de découvrir des outils qui me seront utiles dans mes futurs projets.

Difficultés rencontrées et solutions apportées :

L'intégration des nouvelles fonctionnalités demandées a été un véritable défi. Après avoir échangé avec un développeur expérimenté, j'ai pu mettre en place l'une des principales fonctionnalités requises. Cependant, le manque de temps avant la fin du stage nous a poussés à changer notre approche en passant à Django. Travailler avec de nouvelles technologies a impliqué beaucoup d'essais et d'erreurs. Bien que j'aie utilisé l'intelligence artificielle pour m'aider à résoudre certains problèmes, il a souvent été nécessaire d'identifier les erreurs, de les corriger et de tester à nouveau jusqu'à obtenir le résultat attendu. Le passage à Django a nécessité une adaptation rapide et une collaboration efficace avec mes camarades. Grâce à une communication régulière et à des recherches approfondies, nous avons su répondre aux attentes tout en respectant les délais.

Apports du stage dans la formation BTS SIO SLAM :

Ce stage a parfaitement complété ma formation. Il m'a permis d'appliquer les notions théoriques vues en cours, comme la conception de bases de données et les schémas applicatifs, dans un contexte professionnel. J'ai également renforcé ma capacité à m'adapter rapidement aux imprévus et à proposer des solutions concrètes.

Conclusion :

Ce stage a été une expérience à la fois enrichissante et formatrice, tant sur le plan technique que personnel. J'ai pu renforcer mes compétences, explorer de nouveaux outils et développer ma capacité à travailler en équipe sur des projets ambitieux. Cette expérience m'a permis de mieux me préparer pour ma future carrière dans le domaine de l'informatique.