Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER-PROJET

Nom de naissance Grange

Nom d'usage

Prénom Estelle

Adresse 11 rue du 19 mars 1962

42510 NERONDE

Titre professionnel visé

Titre professionnel développeur web et web mobile - Niveau III

Sommaire

Article I. REMERCIEMENTS

Je remercie tout particulièrement l'Ecole Simplon de Roanne qui m'a accompagnée tout au long de mon projet du configurateur automobile. Pedro et Gael, nos deux formateurs, pour leur pédagogie, grâce à eux j'ai appris le métier de développeur web et j'ai ainsi plusieurs "codes" à mon arc ! La communauté d'OpenClassRoom pour leurs cours Open Source et l'entraide.

Je remercie également Céline Guillot qui m'a aidée et soutenue dans la mise en place de toutes mes idées pour la création de ce site.

Ainsi que tous les Apprenants de la 1ère promotion, notre directeur Ludovic Cabanne, et Faustine Joly.

Article II. RESUME DU PROJET EN ANGLAIS

Cette partie n'est plus obligatoire. La partie anglaise concernera la description d'une situation de travail ayant nécessité une recherche, effectuée durant le projet, à partir de sites anglophones.

Article III. LISTE DES COMPETENCES DU REFERENTIEL

Intitulé de l'activité type	Compétences professionnelles
Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité	1. Maquetter une application 2. Réaliser une interface utilisateur web statique et adaptable 3. Développer une interface utilisateur web dynamique 4. Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce
▸ L'activité 1 est maîtrisée :	OUI ☐ NON ☐
Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité	1. Créer une base de données 2. Développer les composants d'accès aux données 3. Développer la partie back-end d'une application Web ou Web Mobile 4. Elaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce
▸ L'activité 2 est maîtrisée :	OUI ☐ NON ☐

Le document ci-joint : Livret d'évaluation, contient plus de détails sur la liste de compétences du référentiel.

Le Guide de mise en œuvre des Evaluations passées en cours de formation est à télécharger sur le site du ministère de l'emploi : http://travail-emploi.gouv.fr/ (rubrique Documents techniques).

Il comporte un mode d'emploi du présent livret d'évaluations passées en cours de formation.

Article IV. Cahier des charges ou expressions des besoins de l'application à développer

A) Présentation du projet : Configurateur automobile

Le projet a fait l'ébauche d'une première version dans le cadre du projet chef d'œuvre : fin de Front-end lors de ma formation développeur web. J'ai voulu le reprendre, l'améliorer ainsi qu'intégrer une partie Back-end avec la gestion d'utilisateur et une base de données.

1. Le contexte

Le constructeur mythique français, Alpine, commercialise son nouveau modèle A110, disponible en deux versions. Il souhaite développer un site ou une application Web permettant à ses utilisateurs de découvrir ou de redécouvrir la marque à travers une page d'accueil. Le projet doit refléter l'image de marque en empruntant ses codes techniques et stylistiques.

2. L'Expression de Besoin

La Page d'accueil

Le point d'entrée est la page d'accueil. Elle doit refléter l'image de marque à travers sa navigation, son graphisme et ses fonctionnalités. Les différentes fonctionnalités sont les suivantes :

- La page d'accueil est composée de 9 sections :
    - Design avec une image en fond
    - Conception
    - Motorisation
    - Technologie
    - Intérieur
    - Caractéristiques Techniques
    - Versions
    - Galerie
- Une barre de navigation fixe permet de naviguer entre les différentes sections
- Les images et vidéos des différentes sections doivent être intégrées de façon immersive et interactive.
- L'utilisateur peut voir le détail d'une photo de la galerie photo.
- L'utilisateur peut accéder au configurateur via la barre de navigation fixe ou via la section des versions des véhicules.

La page Configurateur

Le configurateur est le cœur du projet, le parcours client doit être sans faille et sans impasse. L'utilisation du configurateur et de la navigation doivent être fluides et instinctives. Les différentes fonctionnalités attendues sont les suivantes :

- Le configurateur est composé de 7 pages/sections.
    - Versions
    - Couleurs
    - Jantes
    - Sellerie
    - Equipements
    - Accessoires

- Le configurateur présente le véhicule et ses configurations sous différents angles de vue (utilisation de carousels, sliders ou autres techniques de visualisation).
- Les différentes étapes du configurateur contiennent plusieurs options et déclinaisons affichées de façon claire et intuitive (image/descriptif/prix).
- Des indicateurs visuels signifient à l'utilisateur quels et combien d'équipements ou d'accessoires il a déjà sélectionné (effet de survol, de sélection et compteurs).
- Un fil d'ariane permet à l'utilisateur de se situer à l'étape où il se trouve.
- L'utilisateur peut revenir à/aux étape(s) précédente(s)/suivante(s).
- Un récapitulatif tarifaire est affiché en permanence et se met à jour en fonction du choix et des modifications faites par l'utilisateur.
- La dernière étape du configurateur est un récapitulatif visuel et tarifaire

Un soin particulier apporté à l'Interface Utilisateur ainsi qu'à l'Expérience Utilisateur. Les zones d'actions devront être claires et ordonnées, les interactions de l'utilisateur devront être simples et intuitives, et les différents états des composants de l'application devront être signifiés visuellement à l'utilisateur.

La visualisation d’utilisateur et de la page profils

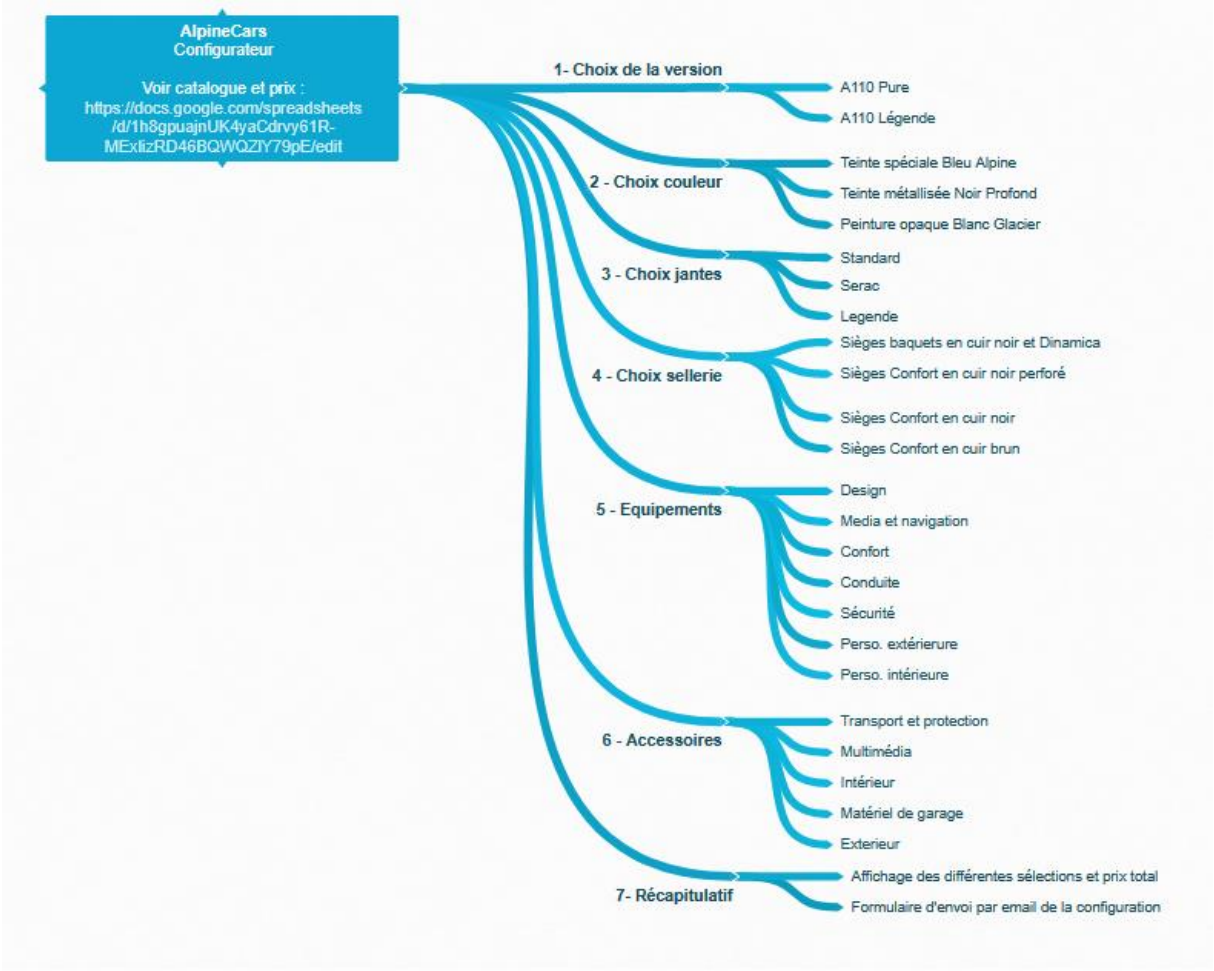
La page profils détient les informations sur l’utilisateur. L’Utilisateur peut consulter, modifier facilement ces informations personnelles, elles doivent s’afficher dynamiquement sur la page. Une partie administrateur permet de créer, supprimer, ajouter et modifier un utilisateur.

La partie Back-end.

Seul un membre connecté peut acheter une voiture et faire ainsi un devis. L’admin peut facilement accéder aux informations utilisateurs ainsi qu’au devis. L’authentification et l’enregistrement en base de données est primordiale.

3. Le Mapping de la home Page et du configurateur

Pour mieux visualiser le site web, notamment les pages principales : la home page et le configurateur, ce document technique a été réalisé avec l’outil Coggle:



B) Partie Front-End : Maquette, typographie et CSS

1. Les Maquettes de la Home Page et du Configurateur Auto

Le lien de partage : https://app.moqups.com/9xfevXA4xT/view/page/ad64222d5

Les maquettes de la home Page et du configurateur sont réalisées avec le site Moqups. La maquette me permet facilement de visualiser mes ‘’row ‘’ lors de l’intégration web. La class « row » contient le system de grid utilisé par le Framework Bootstrap, elle permet d’englober les différents éléments ainsi que de les positionner. Ainsi on gère plus facilement le responsive Design. Les maquettes Moqups ont été réalisées pour un écran de type LG , la taille d’un écran standard d’un ordinateur.

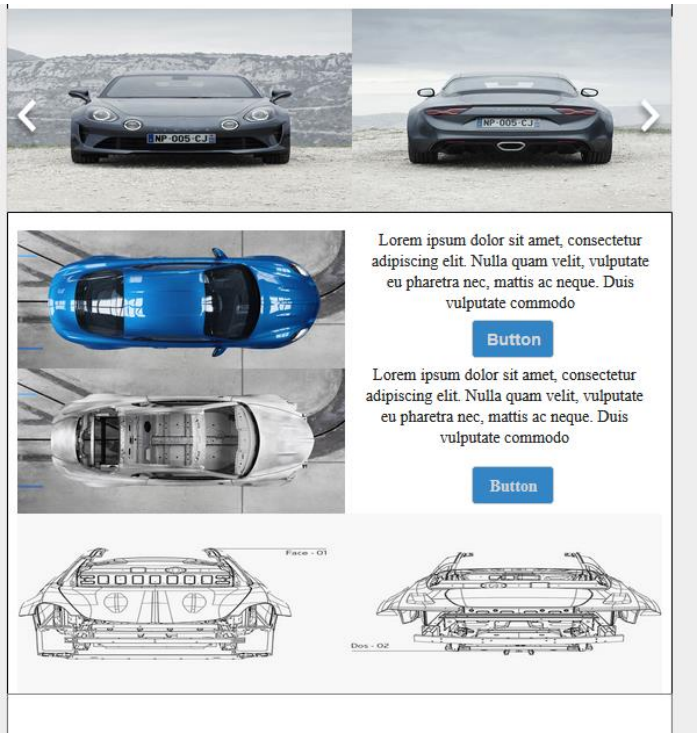
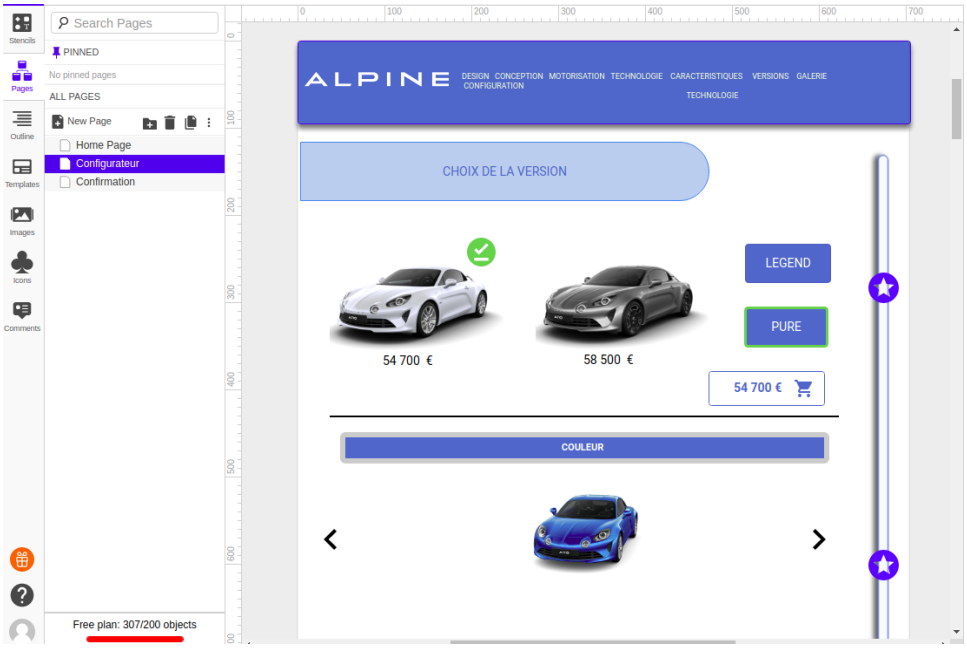


Figure 1Maquette de la home Page : la partie conception



2. La typographie et les couleurs du site web

Le site utilise le thème Lux de Bootswatch : https://bootswatch.com/lux/ notamment pour la typographie du texte, il s'intègre parfaitement au caractère exigent du client. Avec le site Frontify, j'ai créé mes propres contraintes graphiques.

Le lien de partage :

https://company-146779.frontify.com/document/258731#/basics/typography

Pour les couleurs du site, voici la palette de couleur :

Colors

Cette palette de couleur : Alpine Broom Broom . Elle inspire le bleu de la montagne et le blanc de la neige dans un style moderne. Cette palette s'intègre parfaitement avec le Logo Alpin

	White	HEX #FFFFFF	RGB 255, 255, 255
	Mine Shaft	HEX #232323	RGB 35, 35, 35
	Zircon	HEX #F8FBFF	RGB 248, 251, 255
	Zumthor-4	HEX #EAF4FF	RGB 234, 244, 255
	Zumthor-2	HEX #D6EAFF	RGB 214, 234, 255
	Zumthor-1	HEX #ADD6FF	RGB 173, 214, 255
	Malibu	HEX #84C1FF	RGB 132, 193, 255

Figure 2 Frontify palette de couleur

J'ai créé cette palette pour le côté clair et moderne. Cependant d'autres nuances de bleu ont été utilisées sur le site du configurateur automobile, notamment un bleu foncé pour la partie administration. Le bleu est la caractéristique principale de notre site.

Concernant les boutons, nous retrouvons les mêmes pour la partie navigateur et la home page. Ce sont les boutons principaux du site. Nous retrouvons leurs formes dans les maquettes.

3. Utilisation de SASS pour la page inscription et connexion

Le design tient une place fondamentale au sein du site. Pour cela nous avons utilisé le package webpack encore pour intégrer SASS dans un projet Symfony . La page d'inscription ainsi que celle de la connexion utilisent cette technologie.

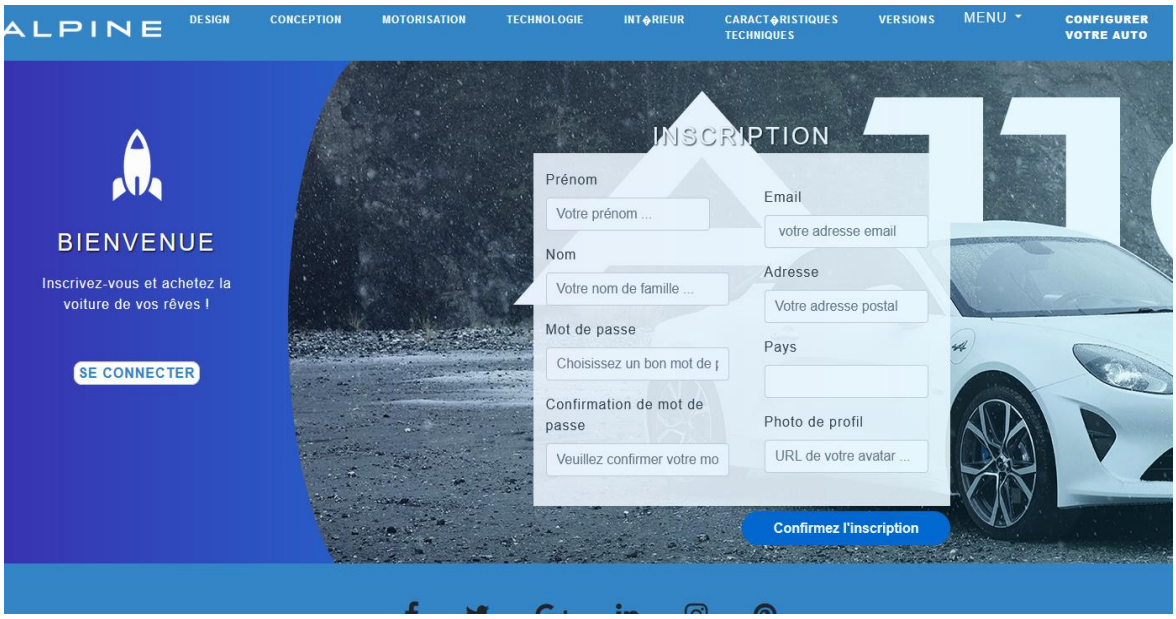


Figure 3 page d'inscription

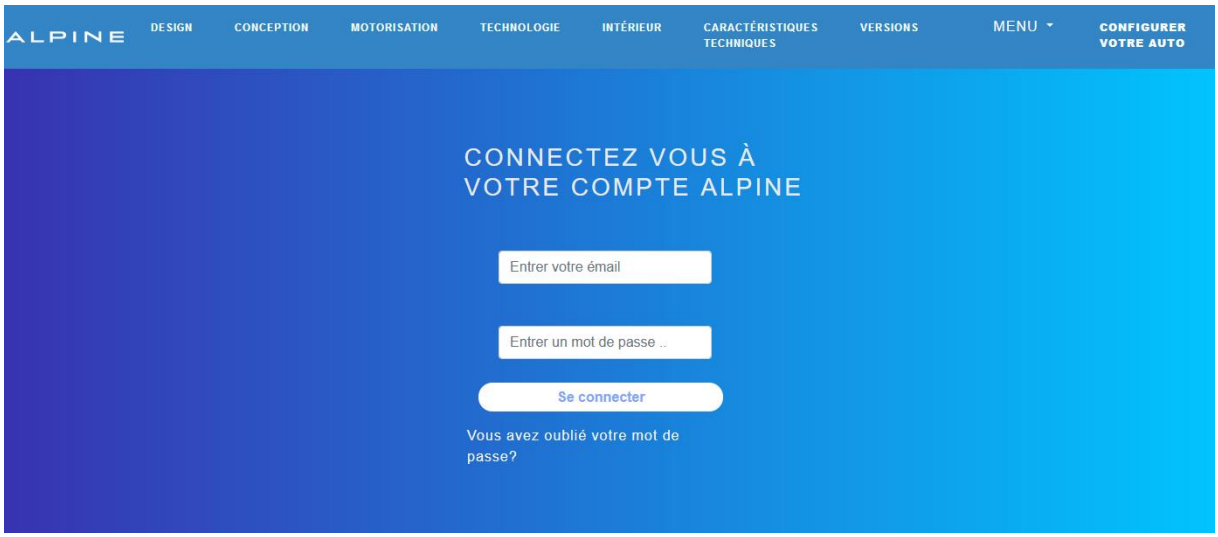


Figure 4 la page de connexion

Grâce au animations CSS ou à libraire AOS, le site contient de nombreuses animations.

```css
.register-left img{
    margin-top: 15%;
    margin-bottom: 5%;
    width: 25%;
    -webkit-animation: mover 2s infinite  alternate;
    animation: mover 1s infinite  alternate;
}
@-webkit-keyframes mover {
    0% { transform: translateY(0); }
    100% { transform: translateY(-20px); }
}

@keyframes mover {
    0% { transform: translateY(0); }
    100% { transform: translateY(-20px); }
}
```

Cette animation mover permet d'animer facilement sur l'axe verticale, on la retrouve sur la page d'inscription sur le côté gauche.

Webpack Encore et AOS sont plus détaillée dans la partie technique.

C) Gestion du temps avec le Trello et le versioning avec Git

1. La gestion du temps avec le Trello

Trello est une application permettant de gérer des projets et des tâches. Pour planifier et organiser mon application, mon trello comporte plusieurs catégories :

- La liste des tâches
- En cours
- En validation
- Terminer
- à refaire

Chaque tâche est avancée au fur à mesure du projet, elle commence de la « Liste de tâches » et finit dans la catégorie « A terminer ». L'État en cours décrit la tâche actuelle et ne contient généralement pas plus de deux tâches. Une fois finie, elle est tout d'abord vérifiée puis transportée dans la catégorie « terminer » ou « à refaire ». Le Trello permet aussi d'afficher des dates limites, grâce à cette fonctionnalité j'ai pu gérer le timing.

2. Le versioning avec Git

L'utilisation d'un dépôt Git m'a permis de partager mon code, et de voir l'avancement du projet en fonction des « commit ». Le versionning sauvegarde mon ancien code, l'extension Git de visual studio code compète cette fonctionnalité directement dans l'éditeur de code à l'aide d'un visuel.

D) Gestion des utilisateurs : Authentification et affichage

1. L'authentification avec Symfony

Lien direct de la documentation du Bundle Symfony :

https://symfony.com/doc/current/security.html

De base un visiteur a accès seulement à la home page, page d'inscription, page de connexion, et la page du configurateur automobile. Il ne peut pas faire un devis sans s'inscrire, d'où la nécessité d'une authentification dans mon application web.

Le controller AccountController ainsi que le Bundle Security gère l'authentification. Pour intégrer ce dernier composant à notre application web :

composer require symfony/security-bundle

```yaml
security:
    encoders:
        App\Entity\User:
            #hashage utiliser pour crypter le mot de passe #
            algorithm: bcrypt

    # https://symfony.com/doc/current/security.html#where-do-users-come-from-user-providers
    providers:
        in_memory: { memory: ~ }
        #on cree un nouveau priver qui provient de la base de donnée #
        in_database:
            entity:
                class: App\Entity\User
                property: email
    firewalls:
        dev:
            pattern: ^/(_(profiler|wdt)|css|images|js)/
            security: false
        main:
            anonymous: true

            provider: in_database

            form_login:
                login_path: account_login
                check_path: account_login
```

Une fois la commande entrée, j'ai créé un nouveau provider provenant de la base de données dans le fichier security.yaml , ainsi lorsque l'utilisateur se connecte, ou s'enregistre Symfony va vérifier les informations correspondant bien à la base de données. Lorsque j'utilise ce provider, je dois rajouter dans l'Entity User, l'inferface UserInterface qui doit contenir les 5 méthodes suivantes : getRoles(), getPassword(), getSalt(), getUsername(), eraseCredentials() .

Dans Security.yaml , l'hashage du mot de passe que j'utilise est l'algorithm bcrypt. Je ne peux pas rentrer un mot de passe utilisateur non crypté dans la base de données, Symfony le refusera lors de l'authentification.

Dans notre controller Account, nous avons trois méthodes pour la partie authentifications : login, logout, register.

La gestion :

```php
/**
 * @Route("/login", name="account_login")
 */
public function login(AuthenticationUtils $utils)
{
    $error = $utils->getLastAuthenticationError();
    $username = $utils->getLastUsername();


    return $this->render('account/login.html.twig' ,[
        'hasError' => $error !== null,
        'username' => $username]);
}
```

Grâce au bundle Security et à la classe AuthenficationUtils, je peux récupérer les erreurs liées à l'authentification, ce qui me permet côté clients d'inviter l'utilisateur à ressaisir son email et son mot de passe en cas d'erreur.

```php
/**
 * le formulaire d'inscription
 *
 * @Route("/register", name="account_register")
 *
 * @return Response
 */
public function register(Request $request, ObjectManager $manager, UserPasswordEncoderInterface $encoder) {
    //instance de l'objet user
    $user = new User();

    // on cree le formulaire à l'aide du form RegistrationType
    $form = $this->createForm(RegistrationType::class, $user);

    // récupère les donnée de type request au sein du formlaire
    $form->handleRequest($request);

    //on vérifie si le form à bien été envoyer et valider
    if($form->isSubmitted() && $form->isValid()) {

        // on hash le code
        $hash = $encoder->encodePassword($user, $user->getPassword());
        $user->setPassword($hash);

        //on enregistre les données
        $manager->persist($user);
        $manager->flush();

        //message flash
        $this->addFlash(
            'success',
            "Votre compte a bien été créé ! Vous pouvez maintenant vous connecter !"
        );

        // si l'utilisateur à bien remplie les conditions alors on le redirige vers la page de connexion
        return $this->redirectToRoute('account_login');
    }

    //sinon on le renvoie à la page d'inscription
    return $this->render('account/registration.html.twig', [
        'form' => $form->createView()
    ]);
}
```

Illustration 1: méthode register : inscription

La méthode register enregistre les différents utilisateurs du site. Une fois authentifié, cela donne à l'utilisateur accès à son profil ainsi à l'autorisation de pouvoir faire un devis.

```
/**
 * Se déconnecter
 *
 * @Route("/logout", name="account_logout")
 *
 * @return void
 */
public function logout() {
    //  besoin de rien d'autre.
}
```

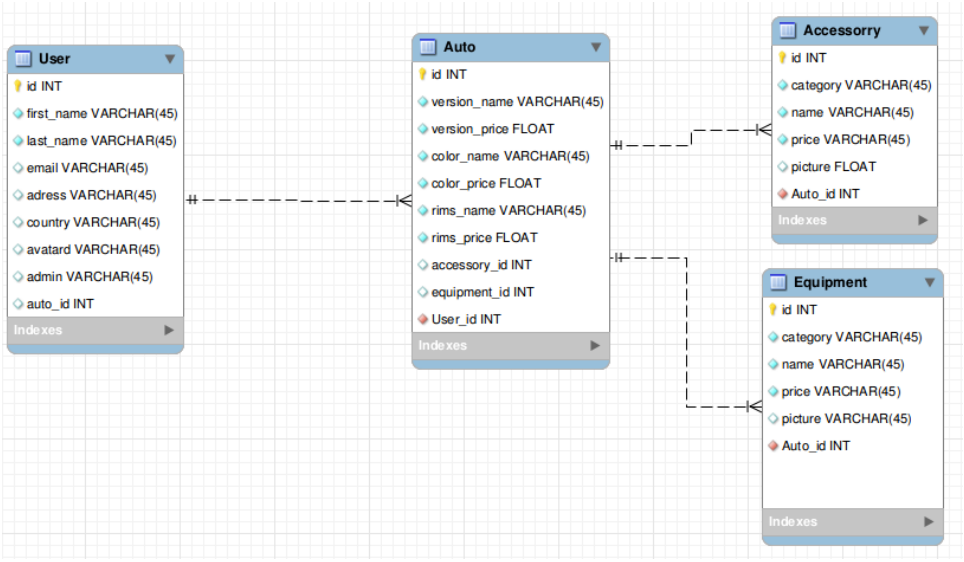
La méthode de déconnexion est très simple, elle est gérée par Symfony. Une fois déconnecté l'utilisateur devra s'authentifier pour accéder à ses informations.

3. Visualisation des Users dans la partie admin

La partie administration est très importante, elle permet à l'administrateur de pouvoir modifier, ajouter, supprimer un utilisateur, une configuration automobile, un accessoire, un équipement, en clair à partir d'une administration on gère toutes les données de l'application. Pour cette partie j'utilise le bundle esayAdmin. Pour l'affichage et les différentes options proposées, la modification se fait principalement dans le fichier easy_admin.yaml .

E) Base de données

1. Schéma de la base de données et relation entre les différentes entités



Tous les sites web ont besoin d'enregistrer des informations, comme la liste de leurs utilisateurs, des messages qui ont été échangés, etc. C'est là qu'une base de données intervient : c'est un logiciel dédié au stockage de données. La base de données de mon projet s'appelle autoProjet, elle comporte 4 tables avec leurs propriétés :

- La table User définie l'utilisateur. Chaque utilisateur peut acheter plusieurs voitures, et une voiture n'est attribuée à qu'un seul utilisateur. La relation entre ces deux tables est donc One To Many .
- La table Auto définie l'automobile. Chaque automobile peut avoir plusieurs accessoires et plusieurs équipements, mais l'équipement ou l'accessoire ne peut être attribué qu'à une automobile. La relation est donc de type de One to Many .
- La table Accessory represente les accessoires. Chaque automobile peut avoir plusieurs accessoires.
- La table Equipment représente les équipements. Chaque automobile peut avoir plusieurs équipements.

Les relations entre les tables dans mon projet sont faites grâce à Symfony au moment de saisir l'instance. Voici l'exemple en français de la création de la relation User->Auto lors de l'insistance avec Symfony:

"Php bin/console make:entity"

Choix du nouveau champ :

"Auto" (nom de la table ou Class que nous voulons reliée par la relation")

Choix du type (exemple : float, int, relation, string ..)

relation

Qu'elle type de relation voulez-vous définir (OneToMany, ManyToOne..) ?

OneToMany

Comment voulez-vous nommer ce champ ?

buy

Voulez-vous supprimer les orphelins lors de la supprimation de la class ?

No

(Je veux garder les données automobiles choisies par l'utilisateur, même si l'utilisateur supprime son compte)

2. L'ORM de Symfony

Dans Symfony, afin d'interagir avec une base de données, on utilise un ORM, Object-Relating Mapping. L'ORM, par défaut dans Symfony, est Doctrine. Doctrine crée des entités qui permettent de manipuler la base de données à travers des objets. Ces entités correspondent aux tables de notre base de données. Elles sont utilisées dans des contrôleurs pour faire des requêtes qui donnent un résultat affiché dans la vue.

F) Point de blocage et solution

Au cours de mon projet j'ai eu plusieurs points de blocages : google est notre ami. En recherchant, je remarque que mes problèmes ont déjà été rencontrés par d'autres développeurs. Le forum StackOveOflow est une référence pour tout programmeur et une aide précieuse en plus de la documentation officielle.

Article V. SPECIFICATIONS FONCTIONNELLES

Article VI.

1- Fonctionnalité global du site :

- Responsive
- Ergonomique

Le site est Responsive et Ergonomique sur les pages principales.

2- La Page d'accueil

- Une barre de navigation fixe permet de naviguer entre les différentes sections
- L'utilisateur peut accéder au configurateur via la barre de navigation fixe ou via la section des versions des véhicules.

3- Formulaire de création de compte/inscription

Un utilisateur lambda peut naviguer sur plusieurs pages mais pour faire un devis de sa configuration automobile, ou avoir accès à son profil il doit s'inscrire si ce n'est pas fait puis se connecter.
L'inscription est faite via un formulaire d'inscription, qui va récupérer les informations requises pour la création d'un compte utilisateur.
Le formulaire vérifie côté client la qualité des informations à enregistrer dans la base de données.

4. Page Profil de l'utilisateur inscrit

L'utilisateur une fois connecté peut modifier son profil via un formulaire de modification "edit" puis valider ces changements qui seront mis à jour dans la base de données.

5. Enregistrement du configurateur Automobile dans la base de données

Après la création du profil, l'utilisateur peut alors commencer à faire un devis pour sa voiture. C'est un formulaire.

7. Fonction CRUD (create, read, update, delete):

Chaque Entity (Auto, Equipment, User, Accessoire) possède une page qui te permet de faire chaque méthode.

Article VII. SPECIFICATIONS TECHNIQUES

Ce projet est développé par le biais du framework Symfony 4 et Vue Js pour le front-end. Ce choix se porte sur plusieurs points :

- Il est basé sur le langage php
- Le développement et l'arborescence sont normés
- Une structure MVC (Modèle Vue Contrôleur)
- Une excellente communauté et d'une documentation à jour
- Un nombre important de bundle prêt à l'emploi
- Il dispose de mesure de sécurité vis-à-vis des failles et attaques XSS, CSRF et injection SQL.

De plus le projet Symfony a été créé par des français.

Au sein du framework différents composants sont utilisés :

- Le bundle Asset pour intégrer le fichier asset
- Vue JS
- Le bundle Security pour l'authentification et la sécurité
- Easy bundle pour le back office
- Le framework dispose déjà du moteur de template Twig qui sera conservé

L'utilisation de l'IDE visual Studio pour réaliser ce développement, car il est gratuit ! Il fournit aussi un ensemble de service et d'extension simplifiant l'appréhension de la structure des fichiers et l'apport d'un Terminal facilitant l'usage des commandes prévues par le framework et l'accès au versionning.

Pour réaliser les maquettes : Moqups .

Article VIII. REALISATIONS

La commande de Creation de symfony :

symfony/website-skeleton

La partie fonctionnalité comporte également la partie Réalisation.

Article IX. CONCLUSION

J'ai beaucoup aimé l'ambition du projet et le côté professionnel, faire un configurateur automobile demande beaucoup de travail, la gestion du temps a donc été la partie la plus difficile. J'ai mis environ 1 mois pour concrétiser le site, mais si j'avais eu plus de temps, mon site aurait peut-être vendu du rêve.

Au niveau technique, j'ai pu varier en utilisant le framework Symfony. J'utilise Webpack avec lequel j''ai fait du SASS ou du CSS, et je minifiais mes fichiers css afin d'optimiser le site. J'ai aussi utilisé différents bundles créés par Symfony. J'ai apprécié leur simplification et l'aide apportée par Symfony et ces différents bundles.

Il reste beaucoup de bugs à améliorer, que ce soit au niveau Back-end, notamment pour la gestion de données, plus contrôler au niveau de la saisie d'utilisateur. Ou le côté Front-end, au niveau des couleurs, ou du responsive sur certaines images.

Au cours de ce projet, j'ai énormément appris du coté Back-end aussi bien que du Front-end .