

CAHIER DES CHARGES



Site de réservation Taxi

Zemni Sarra

Janvier 2023

Table de matière

1. Présentation de L'auteur	3
2. Présentation du projet.....	3
2.1 Entreprise	3
2.2 Représentation de site	4
2.3 Cibles de sites	5
2.4 Le benchmark concurrentiel	5
2.5 Personas	6
2.6 Planning prévisionnel	6
2.7 Budgétisation du projet	6
2.8 Wireframe	7
2.9 Arborescence du site	9
2.9.1 Plan du site.....	9
2.9.2 Organisation des fichiers	10
3. Développement du site	12
3.1 Langages et Framework	13
3.2 Traitement des données	14
3.3 Insertion des données dans la base	16
3.4 La Sécurités des comptes User.....	19
3.5 BackOffice L'espace Administratif “Dashboard Admin”.....	23
3.6 Controller, Templates et Formulaires	24
3.6.1 Page d'Accueil	24
3.6.2 Page Taxi	25
3.6.3 Page Réservation.....	27
3.6.4 Page Account.....	30
3.6.5 Page Contact	32
4. Référencement	34
5. LA MISE EN LIGNE	34
6. Design	36
6.1 Charte graphique	36
6.2 Responsive Web Design.....	36

1. Présentation de L'auteur



Passionné par la réalisation et le développement, je me suis tout naturellement tourné vers une carrière de développeuse web. Je suis ravi de suivre cette formation. J'étais étudiante à l'université René Descartes à rue Saints **Pères**, 75006 **Paris** en mathématique Informatique et j'ai suivi une formation ERP\GPE dans le domaine de développement java JEE et Java Swing... Avec Une bonne base en Algorithme.

En effet, Pour compléter mes connaissance, pôle emploi mon proposé cette formation "Développeur Web et Web Mobile" pour exploiter le domaine de travail et faciliter mon intégration.

Je mettrai mon savoir-faire en pratique, ainsi que mes bonnes connaissances en HTML, javascript, SQL, PHP, Symfony...

Estimant posséder les compétences et qualités essentielles d'une bonne développeuse web, je suis persuadé que je saurai dûment compléter mes connaissances avec cette formation.

Travailleuse, efficace, sociable, organisé et doté d'un esprit d'analyse, je prendrai soin d'assimiler les différents besoins et saurai facilement m'intégrer.

2. Présentation du Projet

2.1 Entreprise

Il s'agit d'une société de Transport de Voyageurs par Taxis Nommé Kassas Taxi 92 immatriculée sous le numéro Siret :44984634400041.

C'est un service de transport pour les Handicapes, malade, les hommes d'affaire, les Ministère, les journalistes....

Pour exercer leur profession, les chauffeurs de taxi doivent passer un examen pour obtenir une carte professionnelle, se soumettre régulièrement à une visite médicale et présenter périodiquement leur véhicule à un contrôle technique.

Les Taxis travaille avec un système de réservation on l'appelle les standards de réservation. Actuellement, trois principaux opérateurs sur Le marché des centrales d'appel et de réservation des taxis parisiens. Elles se partagent le marché parisien : - Alpha taxis - Les Taxis bleus - Les Taxis G7.

D'autres standards se sont spécialisés au niveau communal ou départemental. Le prix de revient pour un chauffeur est approximativement de 400 euros par mois pour la location et l'utilisation du matériel. La borne fait également partie des

moyens pour faire appel à un taxi. Le point fort de ce moyen est le choix du client pour la proximité des stations, c'est une permission livrée par la préfecture par un numéro de plaque spécifique pour chaque taxi.

Aussi l'assurance maladie c'est une grande société publique est associe avec les taxis CPAM. Elle fait appel aux taxis pour les Handicapes et les malades grâce à une l'ordonnance délivrée par un médecin.

Kassas Taxi 92 mon proposer de leur crée un site de réservation pour réduire leur frais. Et facilite les démarches de leur travail.

2.2 Représentation de site :

The screenshot shows the homepage of the Kassas TAXI 92 website. At the top, there are language and currency dropdown menus, a search bar, and user account links. Below the header is the Kassas TAXI 92 logo. The main content area features a section titled "Exclusive Taxi" with four taxi options: "Ledoux Auguste" (black car with green leaf icon), "Meunier Nath" (black van), "Didier Antoinette" (black sedan with multilingual icons and flags), and "Pascal Valérie" (silver car). Below this are three smaller thumbnail images of different taxi types. A red URL bar at the bottom left shows "https://127.0.0.1:8000/taxis12".

C'est un site vitrine qui publie leur service de déplacement. L'utilisateur peut passer via le site web pour Prendre des informations des Taxis et Chercher les offres de déplacement, ces informations peuvent les trouverz dans la page d'accueil ou la page Taxi.

L'utilisateur peut faire une inscription, une réservation d'un taxi tous dépend de besoin et envoyer des dossiers CPAM ou recevoir des factures lors de déplacement,

Et il peut envoyer des messages à l'administrateur dans l'onglet Contact.

- Administrateur de site : Gérer une base de données avec toutes les informations nécessaires pour les clients et leurs réservations et les taxis avec leurs catégories.

Ces informations permettent au responsable de contacter les clients pour confirmer ou les annulé les réservations, et contacter les Chauffeurs pour leur attribuer les réservations.

2.3 Cibles de sites :



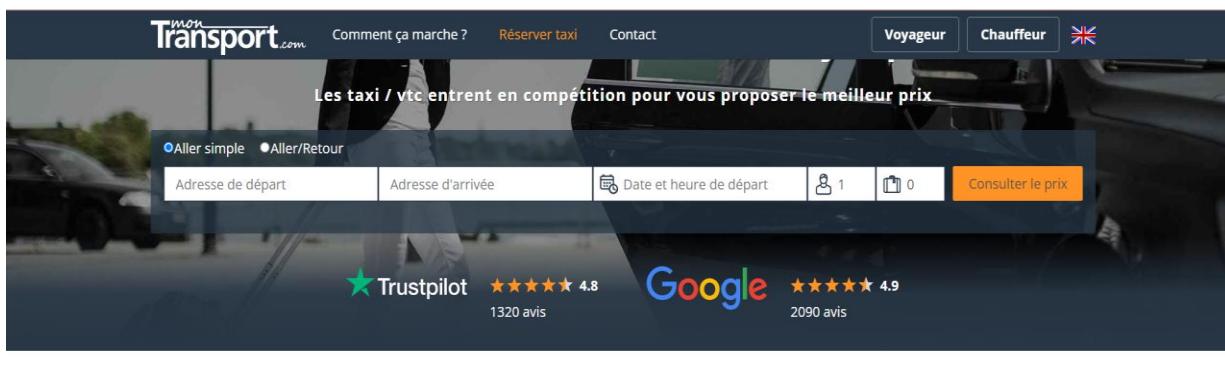
Ce site est ciblé pour tous les Français qui souhaitent réserver un Taxi.

- Les Hommes d'affaires, Les Ministères, les journalistes ...
- Les Voyageurs ...
- Les Handicapées et les malades qu'ils ont une ordonnance de médecin.

Il aura trois types d'utilisateurs de sites :

- Administrateur : celui qui peut gérer dans le BackOffice, les réservations, ajouter les taxis et leurs attribuer les catégories qui leur convient, contacter les utilisateurs, répondre au message (contact) et modifier les offres.
- Visiteur : il peut visualiser et accéder au site et il peut créer un compte.
- Client ou utilisateur : il peut accéder au site, faire une réservation et connecter à son espace personnel (Dashboard). Déjà il a un compte ou il peut voir et modifier ses informations personnelles et ses réservations.

2.4 Le benchmark concurrentiel :



Il y a beaucoup de site concurrent sur le marché. Qui nous a aider à finaliser nos besoins et de trouve ce qui nous correspondions.

2.5 Personas :



- **Nom et Prénom :** Ketati Lotfi
- **Age :** 46 ans
- **Métier :** Chauffeur de Taxi
- **Personnalité :** les modalités de réservation soient par stationnement ou radio G7 parfois c'est compliqué car c'est par rang donc l'attente est fatigante et Le coût du carburant est chère j'aimerai bien avoir une réservation dès ma sortie de la maison. Avec un prix et offre qui me convient et qui convient au client.



- **Nom et Prénom :** Quentin berger.
- **Age :** 39 ans
- **Métier :** Client
- **Personnalité :** Réserver par le numéro de téléphone 3607 c'est le numéro de services qui coutre 0.45euro par minute et les réservations par application est plus simple que l'appel.

2.6 Planning prévisionnel

Entre la réflexion de la réalisation du projet et la finalisation du site web, la durée de la conception est estimée à 5 mois. La livraison est prévue pour début février 2022.

2.7 Budgétisation du projet :

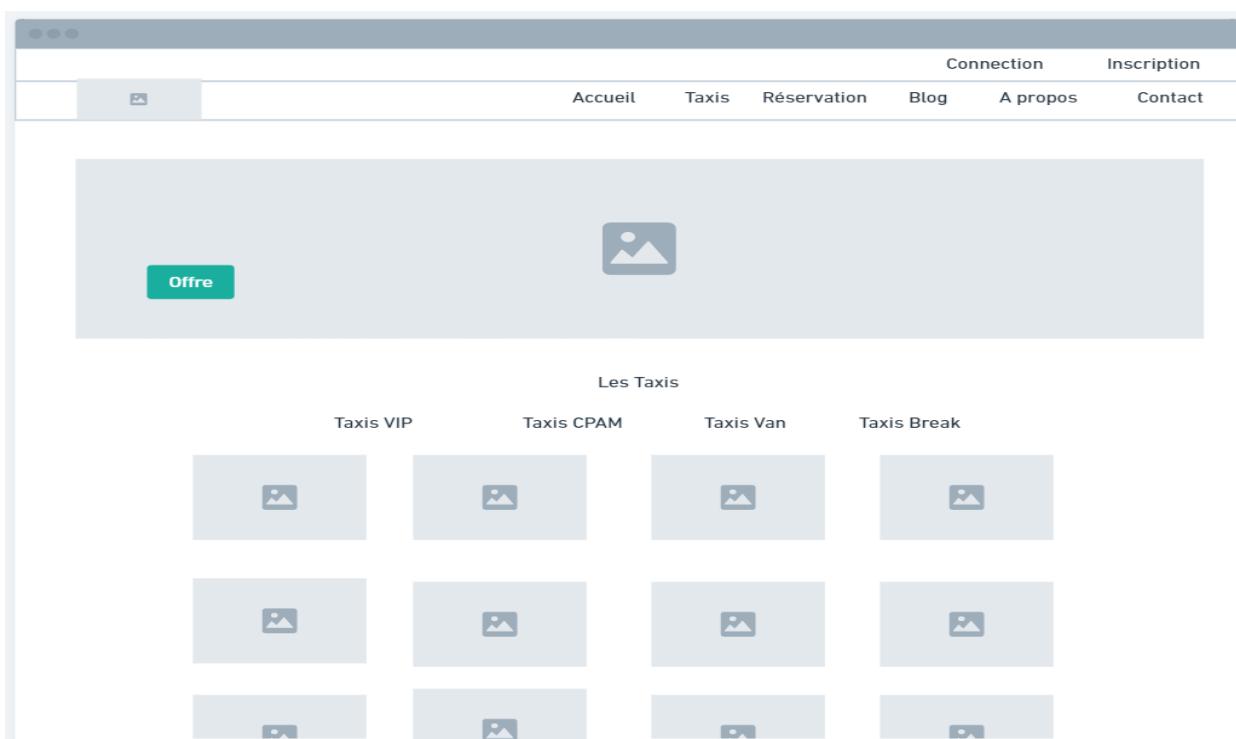
J'ai mis en place un budget prévisionnel concernant le site Sans l'hébergement, le coût du site reviendrait à de 4000 euros.

Le devis prend en considération les éléments suivants du site :

- Réalisé en PHP/Symfony
- Création de logo
- Interphase d'administration
- Interphase de réservation
- Interphase du Client

2.8 Wireframe

- La page d'Accueil :



- La page Taxis :



Déconnection Compte

Accueil Taxis Réservation Blog A propos Contact

Recherche

Place

Bagage

Catégories

S'abonner à notre Newsletter Entrer Votre adresse Email S'abonner

- La page Réservation :

Déconnection Compte

Accueil Taxis Réservation Blog A propos Contact

Réserver

Adresse de départ

Adresse D'arrivé

Date

Jour	Mois	Année
Heure		Minute

Nombre de place

Nombre de Bagage

Choisir un fichier

Recherche

S'abonner à notre Newsletter Entrer Votre adresse Email S'abonner

- La page D'enregistrement ou création de compte :

Connection Inscription

Accueil Taxis Réservation Blog A propos Contact

Création de Compte :

Create an Account

UserName

Firstname Lastname

Password

Repeat Password

Email

Téléphone

Agree terms
[Read Conditions](#)

Registrar

or

Facebook **Google**

S'abonner à notre Newsletter Entrer Votre adresse EMail S'abonner

- La page Contact :

Déconnection Compte

Accueil Taxis Réservation Blog A propos Contact

Nous Contacter :

Quando ambulabat agendis admonere te qualis actio. Si ad corpus, quae plerumque imaginare tecum in balineo quidam aquam fundes aliquod discrimen

Contact

Nom et Prénom

Email

Téléphone

Objet

Nombre de place

Contenu de Message

Envoyer

S'abonner à notre Newsletter Entrer Votre adresse EMail S'abonner

- La page Connection :

Connection

Email

Password

Rappeler moi Mot de passe Oublier

Se Connecter

or

Facebook Google

S'abonner à notre Newsletter

Entrer Votre adresse EMail

S'abonner

- La page Compte Client et Administrateur (Dashboard et BackOffice) :

Connection Inscription

Accueil Taxis Réservation Blog A propos Contact

Dashbaord

Mes Réervations

Réserver Maintenant

Details de Compte

Déconnection

Hello Chihab kassas
Et omnia in potestate nostra esse natura liber, libera, libere valeant; sed illis non est in nostra potestate sunt infirmi, servilis, licet, lex pertinet.

Access the Backoffice

S'abonner à notre Newsletter

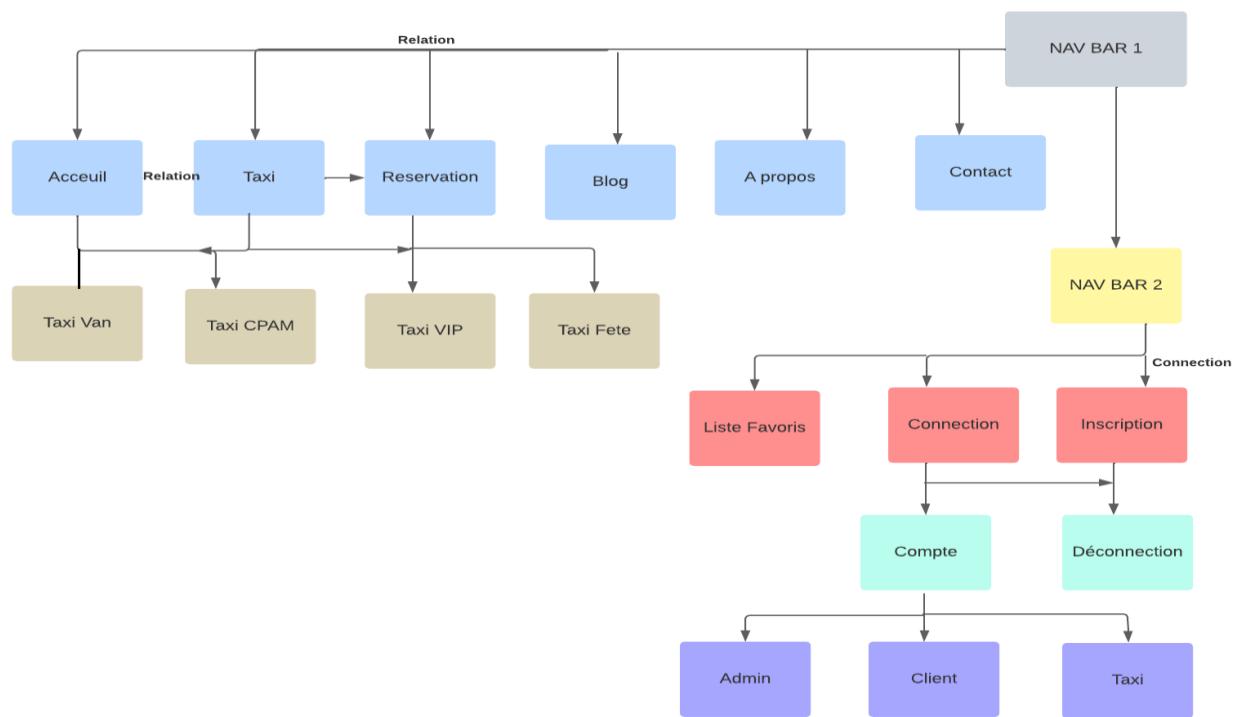
Entrer Votre adresse EMail

S'abonner

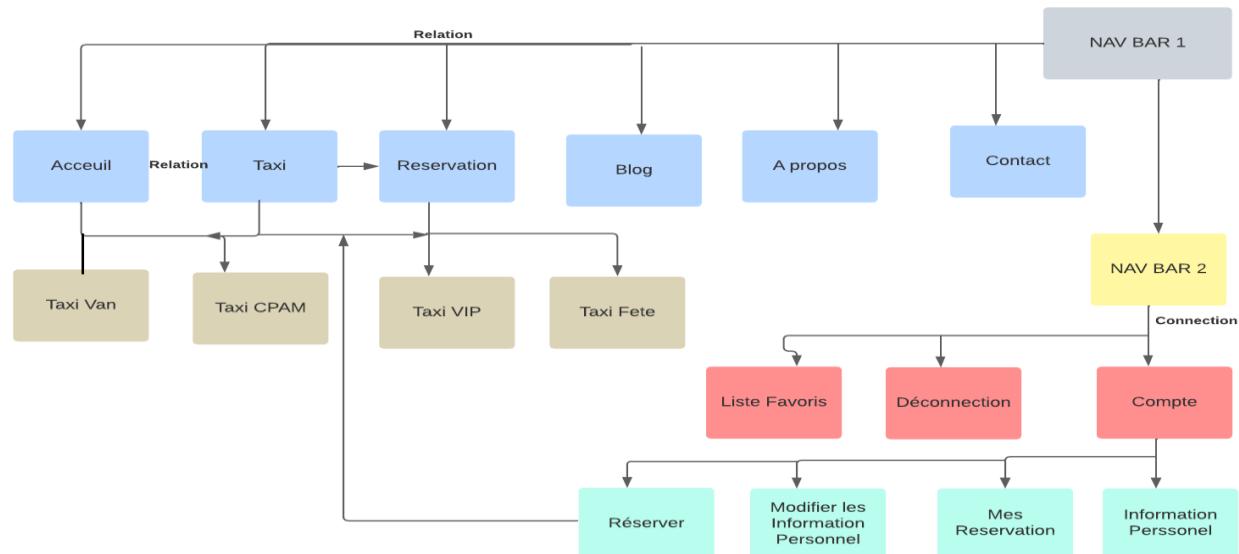
2.9 Arborescence du site

2.9.1 Plan du site

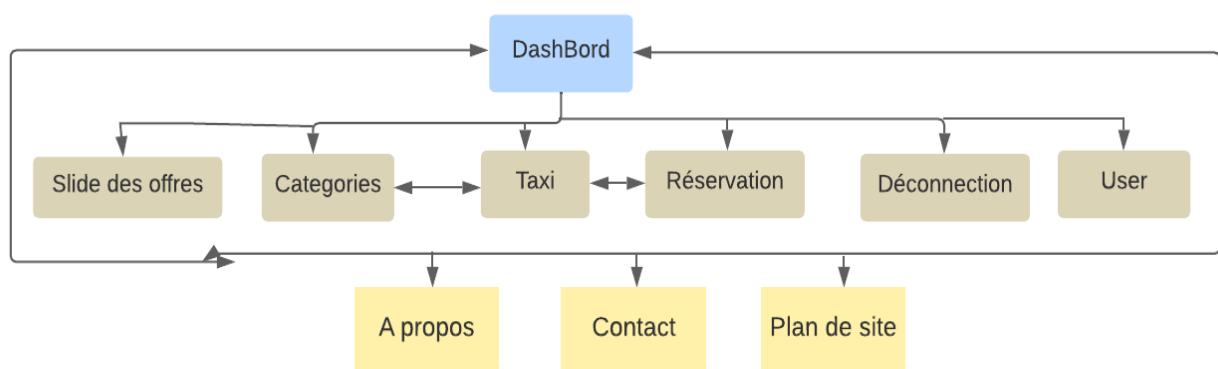
A. Visiteur non-inscrit



B. Utilisateur Client



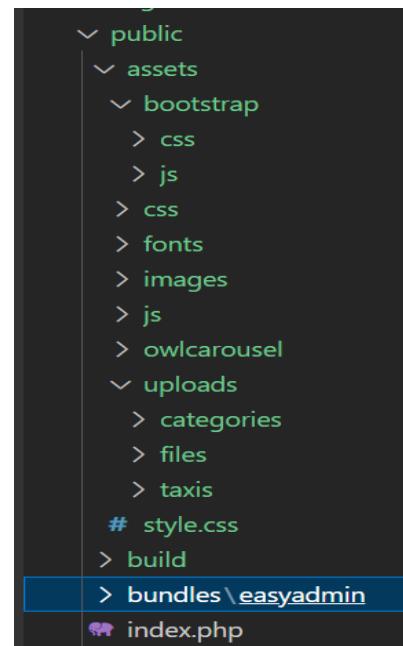
C. Utilisateurs Admin



2.9.2 Organisation des fichiers

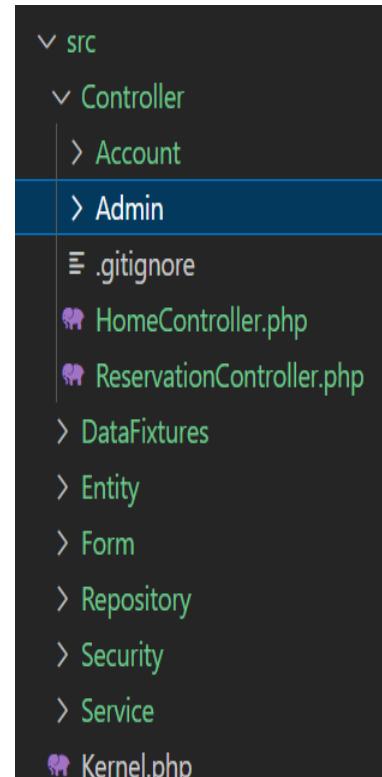
Le Dossier “ Public “ contient les Fichiers :

- bootstrap : c'est le scripte de style bootstrap.
- CSS : feuilles de styles CSS.
- JS : contient les scripts de JavaScript
- uploads : c'est un fichier de stockage des Images et des fichiers.



Le dossier « SRC » contient les dossiers :

- **Controller** : c'est un chef d'orchestre qui se Contente de faire la liaison entre tout le monde. Retourner une réponse avec la classe Response. return \$response.
- **Entity** : Du point de vue de Doctrine,c'est un objet complété avec des informations *Mapping* qui lui permettent d'enregistrer Correctement l'objet en base de données.
- **Form** : Un formulaire Web est composé des Champs de saisie (texte, liste déroulante, cases à cocher, etc.) Le composant Formulaire Permet de créer, traiter et réutiliser des formulaires.



```

    < templates>
      > account
      > cgu
      < home>
        .< index.html.twig
        .< single_taxi.html.twig
      < partials>
        .< footer.html.twig
        .< header.html.twig
        .< taxi-item.html.twig
        .< taxi.html.twig
        .< title_section.html.twig
      > registration
      > reservation
      > security
    .< base.html.twig

```

- **Security** : Le contrôle de la sécurité sous Symfony est très avancé mais également très simple. Pour cela Symfony distingue : L'authentification et L'autorisation
- **Template** : HTML.TWIG C'est un script qui permet d'utiliser des templates, c'est-à-dire des fichiers qui ont pour but d'afficher le contenu de la page HTML de façon dynamique, mais sans PHP.

3. DEVELOPPEMENT DU SITE

3.1 Langages et Framework

Pour la réalisation de ce projet, plusieurs technologies ont été utilisés :

Environnement de développement :

 **Visual Studio Code** : est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS. Les fonctionnalités incluent la prise en charge du débogage, la complétion intelligente du code, la refactorisation du code et Git intégré.

Langages :

 **HTML** : « *HyperText Markup Language* » traduit par « langage de balises pour l'hypertexte ». Il est utilisé afin de créer et de représenter le contenu d'une page web et sa structure.



CSS : « Cascading Style Sheets » ce qui signifie « feuille de style en cascade ». Le CSS correspond à un langage informatique permettant de mettre en forme des pages web (HTML).



JavaScript : un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web. Est un langage de script léger, orienté objet, principalement connu comme le langage de script des pages web.



Bootstrap : est une bibliothèque de **composants** frontend qui permet de **prototyper** vos idées et de créer un site web entier à l'aide de HTML, CSS et JavaScript. Est utile pour le développement **rapide** de prototypes des sites web et applications web.



PHP : (officiellement, ce sigle est un acronyme récursif pour PHP Hypertext Preprocessor) est un langage de scripts généraliste et Open Source, spécialement conçu pour le développement d'applications web. Il peut être intégré facilement au HTML.



Symfony : est un ensemble de composants PHP ainsi qu'un framework MVC libre écrit en PHP. Il fournit des fonctionnalités modulables et adaptables qui permettent de faciliter et d'accélérer le développement d'un site web.

Base de données :



MySQL : est un système de gestion de bases de données relationnelles. Il est distribué sous une double licence GPL et propriétaire.



PhpMyAdmin : est un outil logiciel gratuit écrit en PHP, destiné à gérer l'administration de MySQL sur le Web.

Logiciels :



LOGO MAKER Designer : Logo Designer & Logo Creator, Create Logos and Design Free est un créateur de logo gratuit pour les entrepreneurs, les petites entreprises, les pigistes et les organisations pour créer des logos d'aspect professionnel en quelques minutes. Obtenez un logo gratuit pour les sites Web, les cartes de visite ou les correspondances.

3.2 Traitements de données :

J'ai créé le projet qui est nommé taxi92 :

```
SORTIE      CONSOLE DE DÉBOGAGE      PROBLÈMES      TERMINAL      GITLENS
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony new taxi92 --full
```

Dans ce projet, j'ai travaillé avec 8 tables différentes :

Serveur : 127.0.0.1 » Base de données : taxi92

Filtres

Contenant le mot :

Table	Action	Lignes	Type	Interclassement	Taille	Perte
categories	Parcourir Structure Rechercher Insérer Vider Supprimer	12	InnoDB	utf8mb4_unicode_ci	16,0 kio	-
contact	Parcourir Structure Rechercher Insérer Vider Supprimer	17	InnoDB	utf8mb4_unicode_ci	16,0 kio	-
doctrine_migration_versions	Parcourir Structure Rechercher Insérer Vider Supprimer	7	InnoDB	utf8_unicode_ci	16,0 kio	-
home_slider	Parcourir Structure Rechercher Insérer Vider Supprimer	3	InnoDB	utf8mb4_unicode_ci	16,0 kio	-
messenger_messages	Parcourir Structure Rechercher Insérer Vider Supprimer	6	InnoDB	utf8mb4_unicode_ci	64,0 kio	-
reservation	Parcourir Structure Rechercher Insérer Vider Supprimer	22	InnoDB	utf8mb4_unicode_ci	48,0 kio	-
reset_password_request	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_unicode_ci	32,0 kio	-
reviews_taxi	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_unicode_ci	48,0 kio	-
taxi	Parcourir Structure Rechercher Insérer Vider Supprimer	30	InnoDB	utf8mb4_unicode_ci	64,0 kio	-
taxi_categories	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	utf8mb4_unicode_ci	48,0 kio	-
user	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	utf8mb4_unicode_ci	32,0 kio	-
11 table(s)	Somme	107	InnoDB	utf8mb4_general_ci	400 kio	0 o

Console de requêtes SQL

Signtes Optic

Les Relations entre les tables :

- Une table « user » pour toutes les inscriptions des client, administrateur du site et les chauffeure de taxi.
- Une table « reset_password_request» concernant les modifications des mots de passes des utilisateurs. Cette table a une relation avec la table User.
- Une table « taxi » concernant les taxis, leurs services et leurs informations personnels.
- Une table «categories » concernant les catégories de chaque taxi alors c'est une relation ManyToMany entre ses deux tables (Taxi et Categories) , et qu'elles sont identifiés et modifiés par l'administrateur.
- Une table « reservation » concernant les réservations des Taxis. Admet une relation entre les deux tables Taxi et User.
- (reservation –taxi) : (*ManyToOne Each reservation relates to (has) one Taxi.*

Each Taxi can relate to (can have) many reservation objects.)

- (*reservation - user*) : (*ManyToOne Each reservation relates to (has) one User.*

Each User can relate to (can have) many reservation objects.)

- Une table « review_taxi » concerne les avis les notes et les commentaires des clients. C'est une table est en relation avec la table user et la table taxi
- (review_taxi –taxi) : (*ManyToOne Each review_taxi relates to (has) one Taxi.*

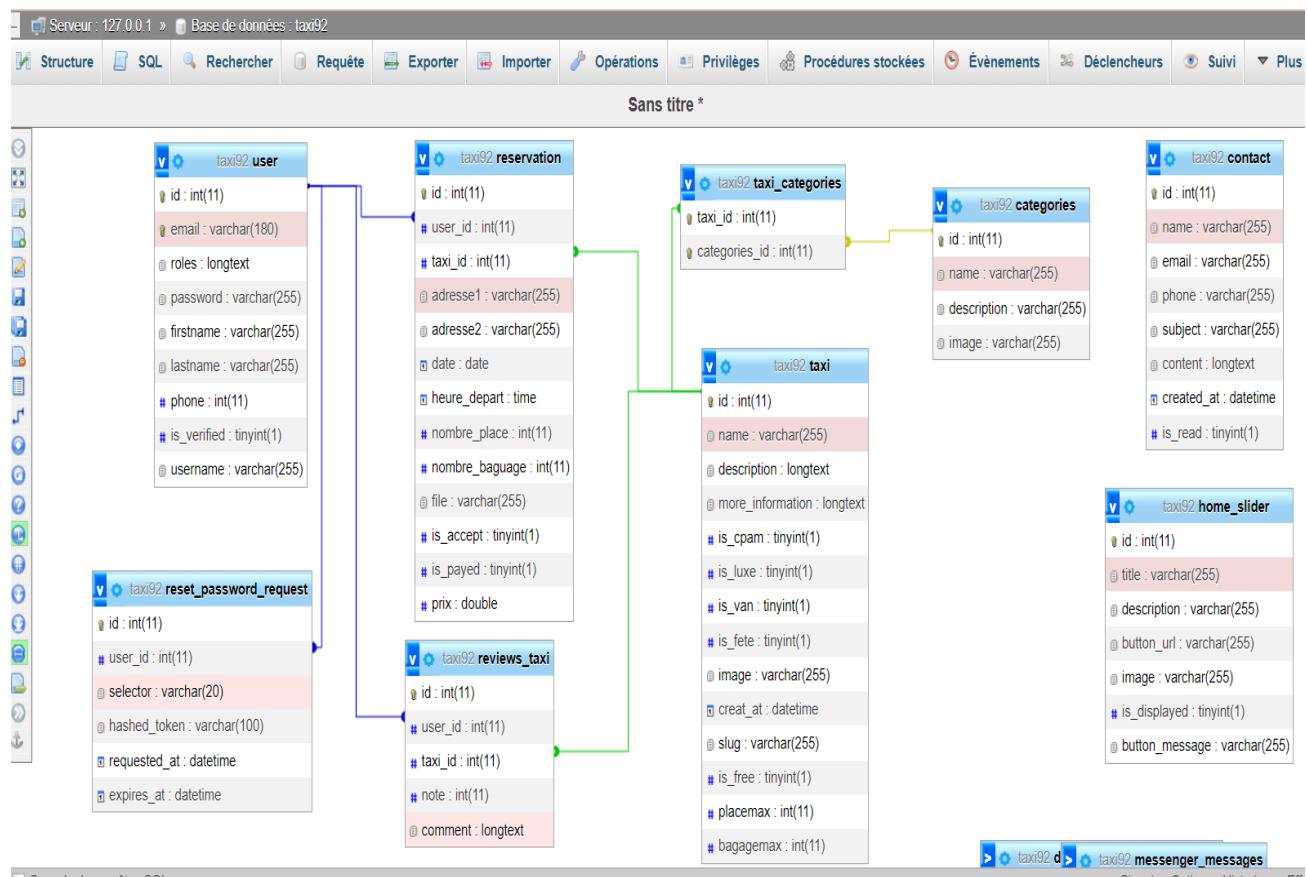
Each Taxi can relate to (can have) many review_taxi objects.)

- (review_taxi - user) : (*ManyToOne Each review_taxi relates to (has) one User.*

Each User can relate to (can have) many review_taxi objects.)

- Une table “Contact” concernant les Contacts des clients qui vont les envoyer à l'espace Administrative.
- Une table “home_slider” concernant les offres de site Kassas Taxi 92 qui vont être publier sur la page d'accueil et elle est gérer par l'administrateur.

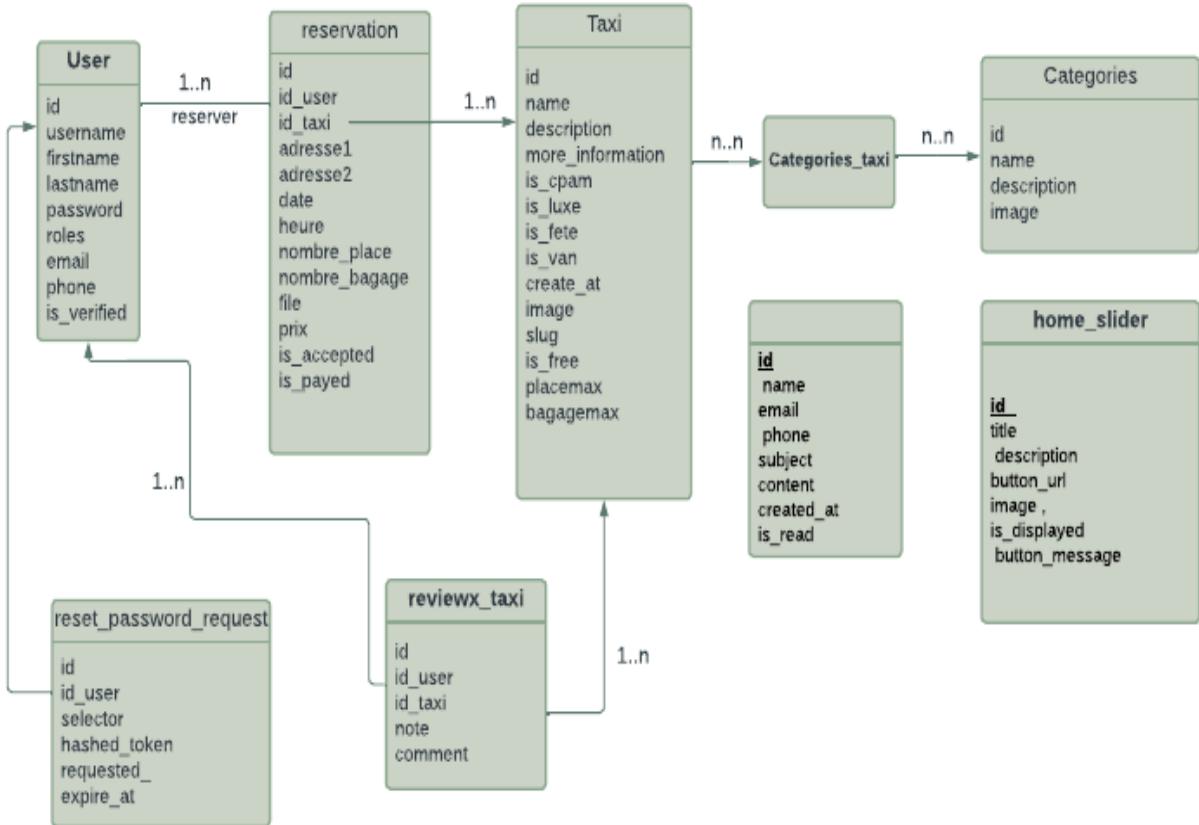
Le diagramme de classes :



Model Relationnel (MLD):

- **user(id_user(pk))**, username, firstname, lastname, password ,roles, email, phone, is_verified;
- **reset_password_request(id_reset (pk))** selector hashed_token requested_ expire_at , **id_user(fk)**;
- **Reservation(id_reservation (pk))** , adresse1, adresse2, date, heure, nombre_place ,nombre_bagage file, prix, is_accepted , is_payed, **id_user(fk)** , **id_taxi(fk)**;
- **Taxi(id_taxi (pk))** , name , description, more_information, is_cpam, is_luxe , is_fete , is_van, create_at , image , slug , is_free , placemax , bagagemax);
- **Categories(id_categories (pk))**, name , description, image);
- **Taxi_categories(id_taxi(fk) , id_categories(fk))**;
- **Reviews_taxi(id_reviews (pk))** , note comment , **id_user(fk)** , **id_taxi(fk)**);
- **Contact(id_contact(pk))**, name , email , phone ,subject , content ,created_at, is_read);
- **Home_slider(id_slider(pk))**,title , description , button_url ,image , is_displayed , button_message);

Model conceptuel des données (MCD):



3.3 Insertion des données dans la base

Pour mener à bien ce projet, J'ai utilisé le framework Symfony 6 qui, comme vu plus haut, est un ensemble de composant PHP.

Et parmi, Symfony fournit tous les outils dont j'ai besoin pour utiliser des bases de données dans cette applications grâce à Doctrine, le meilleur ensemble de bibliothèques PHP pour travailler avec des bases de données.

Pour l'insertion des Classes au niveau de la base de données :

1 - J'ai installé le support de Doctrine via le ORM (Object-Relational Mapping) pack Symfony, ainsi que le MakerBundle, qui m'aidera à générer du code :

```

composer require symfony/orm-pack
composer require --dev symfony/maker-bundle
  
```

A pour but de faciliter ses différentes interactions avec la base de données.

Une fois Doctrine installé, nous mettons en place un fichier ".env.local" qui permettra de Configurer notre application avec la base de données pour un développement en « local », Contrairement au fichier (.env) qui sera configuré pour la mise en production.

The terminal window shows the configuration file .env.local for the taxi application. The file contains the following content:

```

Atteindre Exécuter Terminal Aide .env.local - taxi
base.html.twig M .env.local X
taxi92 > .env.local
1 DATABASE_URL="mysql://root:@127.0.0.1:3306/taxi92"
2 MAILER_DSN=null://null
  
```

Voici les configurations apportées par rapport à mon système :

- db_user = root
- db_password = pas de mot de passe



- db_name = taxi92

Maintenant que les paramètres de connexion sont configurés, Doctrine peut créer la base de données taxi92

```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console doctrine:database:create
```

Une fois que la base de données était créée j'ai commencé à créer les classes "entity" qui représente les tables de la base de données,

```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:user
```

```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:entity Taxi
```

De la même façon j'ai ajouté la classe "Catégories", "Contact" et "HomeSlider", "ReviewTaxi" et "Search".

Une fois que les classes ont été créées, « Maker » va stocker les classes dans le dossier "src -> Entity".

```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:migration
```

Et pour les intégrer dans la base de données et dans phpMyAdmin j'ai utilisé cette commande pour la création de la migration. La migration des Classes "entity" sous forme de tables.

```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console doctrine:migrations:migrate
```

Maker va également créer une classe PHP dans le dossier "src -> Repository".

3.4 La Sécurité des comptes User :

Symfony fournit de nombreux outils pour sécuriser ma application. Certains outils de sécurité liés à HTTP, tels que les cookies de session sécurisés et la protection CSRF, sont fournis par défaut. J'ai utilisé le "SecurityBundle", qui fournit toutes les fonctionnalités d'authentification et d'autorisation nécessaires pour sécuriser ma application.

Pour commencer, j'ai installé le "SecurityBundle" :

```
composer require symfony/security-bundle
```

Et j'ai édité le fichier "security.yaml" (# config/packages/security.yaml) afin de déclarer le chemin :



```

You, il y a 5 secondes | l'auteur (You)
security:
    # https://symfony.com/doc/current/security.html#registering-the-user-hashing-passwords
    password_hashers:
        Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface: 'auto'
    # https://symfony.com/doc/current/security.html#loading-the-user-the-user-provider
    providers:
        # used to reload user from session & other features (e.g. switch_user)
        app_user_provider:
            entity:
                class: App\Entity\User
                property: email
    firewalls:
        dev:
            pattern: ^/(_(profiler|wdt)|css|images|js)/
            security: false
        main:
            lazy: true
            provider: app_user_provider
            custom_authenticator: App\Security\LoginAuthenticator
            logout:
                path: app_logout
                # where to redirect after logout

# Note: Only the *first* access control that matches will be used
access_control:
    - { path: ^/admin, roles: ROLE_ADMIN }           You, il y a 1 seconde • Uncommitted
    - { path: ^/account, roles: ROLE_USER }

when@test:
    security:
        password_hashers:
            # By default, password hashers are resource intensive and take time. This is
            # important to generate secure password hashes. In tests however, secure hash
            # are not important, waste resources and increase test times. The following
            # reduces the work factor to the lowest possible values.
            Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface:
                algorithm: auto
                cost: 4 # Lowest possible value for bcrypt

```

Après la création de classe User, j'ai tapé cette commande suivante

The screenshot shows a terminal window with several tabs at the top: SORTIE, CONSOLE DE DÉBOGAGE, PROBLÈMES, TERMINAL (which is selected), and GITLENS. Below the tabs, the terminal output shows:

```

> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> composer require symfonycasts/verify-email-bundle

```

C'est un système d'enregistrement sécurisé entièrement fonctionnel avec vérification par e-mail.

```

PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi> symfony console make:registration
Creating a registration form for App\Entity\User

Do you want to add a @UniqueEntity validation annotation on your User class to make sure duplicate accounts aren't created? (yes/no) [yes]:
> no

Do you want to send an email to verify the user's email address after registration? (yes/no) [yes]:
> no

Do you want to automatically authenticate the user after registration? (yes/no) [yes]:
> no

What route should the user be redirected to after registration?
[0] _preview_error
[1] _wdt
[2] _profiler_home
[3] _profiler_search
[4] _profiler_search_bar
[5] _profiler_phpinfo
[6] _profiler_xdebug
[7] _profiler_search_results
[8] _profiler_open_file
[9] _profiler
[10] _profiler_router
[11] _profiler_exception
[12] _profiler_exception_css
[13] admin
[14] app_login
[15] app_logout
> 13

updated: src/Entity/User.php
created: src/Form/RegistrationFormType.php
created: src/Controller/RegistrationController.php
created: templates/registration/register.html.twig

```

TERMINAL

```

Fixtures
u\taxi\taxi> symfony console make:registration -form

```

Les fichiers suivants vont être créés automatiquement :

- “src/Security/EmailVerifier.php”: Pour la sécurité et les vérifications des Emails.

EmailVerifier.php - taxi92 - Visual Studio Code

```

private EntityManagerInterface $entityManager
}

public function sendEmailConfirmation(string $verifyEmailRouteName, UserInterface $user, TemplatedEmail $email):
{
    $signatureComponents = $this->verifyEmailHelper->generateSignature(
        $verifyEmailRouteName,
        $user->getId(),
        $user->getEmail(),
        ['id' => $user->getId()]
    );
    $context = $email->getContext();
    $context['signedUrl'] = $signatureComponents->getSignedUrl();
    $context['expiresAtMessageKey'] = $signatureComponents->getExpirationMessageKey();
    $context['expiresAtMessageData'] = $signatureComponents->getExpirationMessageData();
    $email->context($context);
    $this->mailer->send($email);
}

/**
 * @throws VerifyEmailExceptionInterface
 */
public function handleEmailConfirmation(Request $request, UserInterface $user): void
{
    $this->verifyEmailHelper->validateEmailConfirmation($request->getUri(), $user->getId(), $user->getEmail());
    $user->setIsVerified(true);
    $this->entityManager->persist($user);
    $this->entityManager->flush();
}

```

- “templates/registration/register.html.twig” et “confirmation_email.html.twig”: c'est la page HTML qu'elle va afficher le formulaire d'inscription et de confirmation.
- “src/Form/RegistrationFormType.php”: c'est les inputs du formulaire d'inscription qui va permettre de détecter les erreurs d'entrée grâce à des classes : ” EmailType” va

inspecter les emails, “NumberType” envoie un message d’erreur si l’utilisateur tape des lettres au lieu des chiffres et mot de passe doit contenir au moins une majuscule une minuscule et un chiffre grâce à “new Regex()”

```
new Regex(['pattern'=>'^(?=.*?[A-Z])(?=.*?[a-z])(?=.*?[0-9]).{6,}$'],
'message'=>'Votre mot de passe doit contenir au moins une majuscule une minuscule
et un chiffre'],
```

```
mType.php M X LoginAuthenticator.php M ReviewsTaxiType.php U ReviewsTaxi.php index.html.t
RegistrationFormType.php > PHP Intelephense > RegistrationFormType > buildForm
    'label'=>'Prenom'])
    ->add('email', EmailType::class, [
        'label'=>'Email'])
    ->add('phone', NumberType::class, [
        'label'=>'Téléphone',])
    ->add('agreeTerms', CheckboxType::class, [
        'mapped' => false,
        'label'=>'Accepter les Conditions',
        'constraints' => [
            new IsTrue([
                'message' => 'Vous devez accepter nos conditions.',

            ],], 1)
    ->add('plainPassword', RepeatedType::class, [ // instead of being set onto the object directly,//
        'type'=>PasswordType::class,
        'mapped' => false,
        'attr' => ['autocomplete' => 'new-password'],
        'constraints' => [
            new NotBlank([
                'message' => 'Veuillez entrer un mot de passe',]),
            new Regex([
                'pattern'=>'^(?=.*?[A-Z])(?=.*?[a-z])(?=.*?[0-9]).{6,}$',
                'message'=>'Votre mot de passe doit contenir un majuscule un minuscule et un chiffre']),
            new Length([
                'min' => 6,
                'minMessage' => 'Your password should be at least {{ limit }} characters',
                // max length allowed by Symfony for security reasons
                'max' => 4096,
            ],),
            'first_options' => ['label' => 'Mot de Passe'],
            'second_options' => ['label' => 'Confirmer le Mot de Passe '],
        ],
    ],

```

- “src/Controller/RegistrationController.php”: c’est le Controller

```

gle_taxi.html.twig M EmailVerifier.php 6. M RegistrationController.php M X LoginAuthenticator.php M
Controller > Account > RegistrationController.php > PHP Intelephense > RegistrationController > verifyUserEmail
    #[Route('/register', name: 'app_register')]
    public function register(Request $request, UserPasswordHasherInterface $userPasswordHasher,
    {
        $user = new User();
        $form = $this->createForm(RegistrationFormType::class, $user);
        $form->handleRequest($request);
        if ($form->isSubmitted() && $form->isValid()) {
            // encode the plain password
            $user->setPassword(
                $userPasswordHasher->hashPassword(
                    $user,
                    $form->get('plainPassword')->getData()  )
            );
            $entityManager->persist($user);
            $entityManager->flush();
            // generate a signed url and email it to the user
            $this->emailVerifier->sendEmailConfirmation('app_verify_email', $user,
                (new TemplatedEmail())
                    ->from(new Address('contact@taxikassas92.com', 'Kassas Taxi 92'))
                    ->to($user->getEmail())
                    ->subject('Confirmer votre Email')
                    ->htmlTemplate('registration/confirmation_email.html.twig')
            );
            // do anything else you need here, like send an email
            return $this->redirectToRoute('home');
        }
        return $this->render('registration/register.html.twig', [
            'registrationForm' => $form->createView(),
        ]);
    }

    #[Route('/verify_email', name: 'app_verify_email')]
    public function verifyUserEmail(Request $request, TranslatorInterface $translator, UserRepository $userRepository): Response
    {
        $id = $request->get('id');
        if (null === $id) {
            return $this->redirectToRoute('app_register');
        }
        $user = $userRepository->find($id);
        if (null === $user) {
            return $this->redirectToRoute('app_register');
        }
        // validate email confirmation link, sets User::isVerified=true and persists
        try {
            $this->emailVerifier->handleEmailConfirmation($request, $user);
        } catch (VerifyEmailExceptionInterface $exception) {
            $this->addFlash('verify_email_error', $translator->trans($exception->getReason(), [], 'VerifyEmailBundle'));
            return $this->redirectToRoute('app_register');
        }
        // @TODO Change the redirect on success and handle or remove the flash message in your templates
        $this->addFlash('success', 'Votre adresse électronique a été vérifiée.');
        return $this->redirectToRoute('app_register');
    }

```

Le Controller va vérifier si l'email il n'existe pas dans la base de données et si l'utilisateur a bien écrit un email sans erreur grâce au "try et catch". Il aura l'affichage d'un message d'erreur ou un message de succès d'inscription.

Email

hindahotmail.fr



There is already an account with this email





```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> composer require symfony/google-mailer
```

Une fois le formulaire d’inscription est étais créé, je lance la commande :

Symfony console make:controller CGU, qui va afficher les conditions générales d’inscription grâce au fichier Template/cgu/index.html.twig .

CGUController est un router simple

```
#[Route ('/cgu', name: 'cgu_conditions')]

public function index(): Response
{
    return $this->render('cgu/index.html.twig', [
        'controller_name' => 'CGUController ']);}
```

```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:auth
```

La création d'un formulaire de connexion puissant peut être amorcée avec la “make:auth” commande de “MakerBundele” . Cette commande va créer les fichiers suivants :

- “Src/Security/LoginFormAuthenticator.php”

Les Token d'accès ou Token API sont utilisés comme mécanisme d'authentification dans les contextes d'API. Le Token d'accès est une chaîne, obtenue lors de l'authentification (à l'aide de l'application ou d'un serveur d'autorisation). Le rôle du Token d'accès est de vérifier l'identité de l'utilisateur et de recevoir le consentement avant d'émission du Token .



```

e_taxi.html.twig M EmailVerifier.php 6. M LoginAuthenticator.php M X ReviewsTaxiType.php U ReviewsTaxi...
security > LoginAuthenticator.php > PHP Symbols > LoginAuthenticator
class LoginAuthenticator extends AbstractLoginFormAuthenticator
{
    use TargetPathTrait;

    public const LOGIN_ROUTE = 'app_login';
    private UrlGeneratorInterface $urlGenerator;

    public function __construct(UrlGeneratorInterface $urlGenerator)
    {
        $this->urlGenerator = $urlGenerator;
    }

    public function authenticate(Request $request): Passport
    {
        $email = $request->request->get('email', '');
        $request->getSession()->set(Security::LAST_USERNAME, $email);
        return new Passport(
            new UserBadge($email),
            new PasswordCredentials($request->request->get('password', '')),
            [new CsrfTokenBadge('authenticate', $request->request->get('_csrf_token'))]
        );
    }

    public function onAuthenticationSuccess(Request $request, TokenInterface $token, string $firewallName): void
    {
        if ($targetPath = $this->getTargetPath($request->getSession(), $firewallName)) {
            return new RedirectResponse($targetPath);
        }
        return new RedirectResponse($this->urlGenerator->generate('account'));
        throw new \Exception('TODO: provide a valid redirect inside '.__FILE__);
    }

    protected function getLoginUrl(Request $request): string
    {
        return $this->urlGenerator->generate(self::LOGIN_ROUTE);
    }
}

```

- “Config/packages/security.yaml”
- “Src/Controller/SecurityController.php”

```

class SecurityController extends AbstractController
{
    #[Route(path: '/login', name: 'app_login')]
    public function login(AuthenticationUtils $authenticationUtils): Response
    {
        /*if ($this->getUser()) {//
            return $this->redirectToRoute('target_path');// }*/ // get the login error
        $error = $authenticationUtils->getLastAuthenticationError();
        $lastUsername = $authenticationUtils->getLastUsername();
        return $this->render('security/login.html.twig', ['last_username' => $lastUsername, 'error' => $error]);
    }

    #[Route(path: '/logout', name: 'app_logout')]
    public function logout(): void
    {
        throw new \LogicException('This method can be blank - it will be intercepted by the logout key on your f
    }
}

```

Si le mot de passe oublier “[Mot de passe oublier]({{ path('app_forgot_password_request') }})” permet de la récupérer grâce à la méthode suivante :

```

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS

> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> composer require symfonycasts/reset-password-bundle

```

La façon la plus simple et la plus préférée est d'utiliser le MakerBundle de Symfony. Le Maker s'occupera de tout, de la création de la configuration à la génération de mes modèles, contrôleurs et entités.

```

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:reset-password

```

3.5 BackOffice L'espace Administratif "Dashboard Admin"

```

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> composer require easycorp/easyadmin-bundle

```

Le Tableau de bord d'administration Web" qui sera le principal point d'entrée pour gérer les données du site :

```

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:admin:dashboard

```

Dans le contrôleur de tableau de bord, la méthode `configureMenuItems()` qui contient un commentaire sur l'ajout de liens vers les "CRUD". **CRUD** est un acronyme pour "Créer, Lire, Mettre à jour et Supprimer", les quatre opérations de base que je souhaite effectuer sur n'importe quelle entité...

Générons un CRUD avec cette commande :

```

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:admin:crud

```

```

public function configureDashboard(): Dashboard
{
    return Dashboard::new()
        ->setTitle('Kassas Taxi 92');
}

public function configureMenuItems(): iterable
{
    yield MenuItem::linkToDashboard('Dashboard', 'fa fa-home');
    yield MenuItem::linkToCrud('Taxi', 'fa-solid fa-car', Taxi::class);
    yield MenuItem::linkToCrud('Categories', 'fas fa-list', Categories::class);
    yield MenuItem::linkToCrud('Reservation', 'fa-regular fa-calendar-days', Reservation::class);
    yield MenuItem::linkToCrud('Home Slider', 'fas fa-images', HomeSlider::class);
    yield MenuItem::linkToCrud('Contact', 'fas fa-envelope', Contact::class);
    yield MenuItem::section("Administration")->setPermission("ROLE_ADMIN");
    yield MenuItem::linkToCrud('Client', 'fas fa-user', User::class)->setPermission("ROLE_ADMIN");
}

```

Puisque la relation entre la table Taxi et la table Categories est une relation ManyToMany donc j'ai utilisé AssociationFieled .

ImageFieled c'est pour télécharger l'image du Taxi

Kassas Taxi 92

Search

sarraouf@hotmail.com

Taxi

Add Taxi

ID	Name	Slug	Taxi CPAM	Taxi Luxe	Taxi Van	Taxi Fête	Taxi est Libre	Category	Image
8	Ledoux Auguste	Ledoux-Auguste	Off	On	On	On	Off	0	
9	Chauvin Paul	Chauvin-Paul	Off	Off	On	Off	Off	0	
10	Meunier Nath	Meunier-Nath	Off	On	Off	Off	Off	0	
11	Blanchard Dorothee	Blanchard-Dorothee	On	Off	Off	On	Off	0	
12	Didier Antoinette	Didier-Antoinette	Off	On	Off	Off	Off	0	
13	Coste Camille	Coste-Camille	On	Off	Off	On	Off	0	
14	Rodrigues Renée	Rodrigues-Renée	On	Off	Off	Off	Off	0	
15	Pascal Valérie	Pascal-Valérie	Off	On	Off	Off	On	0	

```

DashboardController.php U CategoriesCrudController.php U TaxiCrudController.php U
src > Controller > Admin > TaxiCrudController.php > PHP Symbols > TaxiCrudController

public function configureFields(string $pageName): iterable
{
    return [
        IdField::new('id')->hideOnForm(),
        TextField::new('name'),
        SlugField::new('slug')->setTargetFieldName('name'),
        TextEditorField::new('description')->hideOnIndex(),
        TextEditorField::new('moreInformation')->hideOnIndex(),
        TextField::new('tags'),
        DateField::new('createdAt')->setFormat('dd-MM-yyyy'),
        BooleanField::new('isCPAM', 'Taxi CPAM'),
        BooleanField::new('isLuxe', 'Taxi Luxe'),
        BooleanField::new('isVan', 'Taxi Van'),
        BooleanField::new('isFete', 'Taxi Fête'),
        BooleanField::new('isFree', 'Taxi est Libre'),
        AssociationField::new('category'),
        ImageField::new('image')->setBasePath('/assets/uploads/taxis/')
            ->setUploadDir('/public/assets/uploads/taxis/')
            ->setUploadedFileNamePattern('[randomhash].[extension]')
            ->setRequired(false),
    ];
}

```

3.6 Controller, Templates et Formulaires

3.6.1 Page d'Accueil :

J'ai Cree "HomeController" qui va donner le chemin au "template" "home/index.html.twig"

```

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> symfony console make:controller Home

```

La page d'accueil va afficher tous les taxis grâce à une fonction dans "RepositoryTaxi"
"\$taxiRepository->findAll();"

Et elle contient un filtre pour les options qu'elles sont des entités Booliennes de la classe taxi, ce filtre est grâce à des fonctions dans le “RepositoryTaxi” également.

```
$taxiCpam=$taxiRepository->findByIsCpam(1) ;
```

```
$taxiVan=$taxiRepository->findByIsVan(1) ;
```

Les taxis

```
$homeSlider=$repoHomeSlider->findBy(['isDisplayed'=>true]) ;
```

La classe “HomeSlider” qui est gérer par l'administrateur où on peut afficher toutes les offres sur la page d'Accueil grâce au “RepositoryHomeSlider”.

```
$homeSlider=$repoHomeSlider->findBy(['isDisplayed'=>true]) ;
```

3.6.2 Page Taxi :

J'ai créé également le Controller Taxi et j'ai créé une classe Nome “Search.php” qu'elle contient les “entity” suivante :

- \$minplace- \$maxplace-\$minbagage- \$maxbagage: ils se sont des entiers
- \$categories : c'est un tableau

J'ai créé ensuite un formulaire de recherche “SearchType.php” grâce à la commande `symfony console make:form SearchType`

Et J'ai ajouté une fonction dans la RepositoryTaxi.php une fonction `findWithSearch($search)`

Qui permet de prendre en valeur les variables de Search.php et les comparer avec les variables des Taxi.php .

“p” contient les valeurs de Taxi grâce à `$query=$this->createQueryBuilder` , et \$search contient les valeurs de Search.php.

Et j'ai fait appel à cette fonction dans le ControllerTaxi.php

```
[  
#[Route('/taxi', name: 'taxi', methods:["GET","POST"])]  
public function index(TaxiRepository $taxiRepository, Request $request, PaginatorInterface $paginator): Response  
{  
    $search=new Search();  
    $form=$this->createForm(SearchType::class,$search);  
    $form->handleRequest($request);  
  
    if($form->isSubmitted() && $form->isValid())  
    {  
        $taxis=$taxiRepository->findWithSearch($search);  
    }  
    else  
    {  
        $taxis = $taxiRepository->findBy([]);  
        $pagination = $paginator->paginate(  
            $taxis,  
            $request->query->getInt('page', 1), 8 );  
  
        return $this->render('taxi/index.html.twig', [  
            'controller_name' => 'HomeController',  
            'pagination'=> $pagination,  
            'search'=>$form->createView()]);  
    }  
}
```

- La Création de la pagination dans cette page est grâce à la commande suivante :

```
advisories.
(taxi\taxi92> composer require knplabs/knp-paginator-bundle
```

- J'ai ajouté et configurer le fichier paginator.yaml dans config/packages :

```
config > packages > ! paginator.yaml > {} knp_paginator
1 knpPaginator:
2     page_range: 5                      # number of links shown in the pagination menu (e.g: you have 10 pages, a page_range of 5 will show links for pages 1 to 10)
3     default_options:
4         page_name: page                # page query parameter name
5         sort_field_name: sort          # sort field query parameter name
6         sort_direction_name: direction # sort direction query parameter name
7         distinct: true               # ensure distinct results, useful when ORM queries are using GROUP BY statements
8         filter_field_name: filterField # filter field query parameter name
9         filter_value_name: filterValue # filter value query parameter name
10    template:
11        pagination: '@KnpPaginator/Pagination/bootstrap_v5_pagination.html.twig'      # sliding pagination controls template
12        sortable: '@KnpPaginator/Pagination/bootstrap_v5_bi_sortable_link.html.twig' # sort link template
13        filtration: '@KnpPaginator/Pagination/filtration.html.twig' # filters template
```

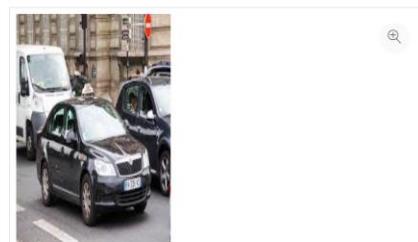
Et dans le fichier templates/taxi/index.html.twig :

```
<div class="row shop_container">
    {% for taxi in pagination %}
        | {{ include('partials/taxi.html.twig',{'taxi':taxi}) }}
    {% endfor %}
</div>
<div class="navigation mx-auto" style="width:max-content;">
    {{ knp_pagination_render(pagination) }}
```

```
$pagination = $paginator->paginate( $taxis, $request->query->getInt('page', 1), 8);
```

Après le filtrage et la recherche des taxis \$taxi, la pagination affiche 8 taxis par page.

The screenshot shows the Kassas TAXI 92 website. At the top, there are language and currency dropdowns (English, USD), a search bar, and navigation links for Compare, Liste Favoris, Connection, and Incription. The main header features a car icon and the text "KASSAS TAXI 92". Below the header, there are three search/filter sections: "Place" with "min ..." and "max ...", "Baguage" with "min ..." and "max ...", and "Catégories" with checkboxes for "Langue Anglais", "Langue Espagnole", "Electrique", "Break", and "Siège Bébé". To the right, there is a grid of eight taxi profiles, each with a thumbnail, the driver's name, and a small description. A pagination bar at the bottom indicates page 1 of 4.



Jacob Jean

Minus est enim qui et alias laudantium. Ad cupiditate ab praesentium molitia modi qui. Non maxime magni et consequatur aut unde. Aliquam tempora sequi fugit error praesentium sequi est. Voluptas quas quia placeat officia doloribus qui qui. Laudantium praesentium magnam voluptatem molestias. Nihil voluptas iure et eaque commodi sed hic saepe. Eaque cupiditate sed sint placeat et. Laudantium accusamus quas non voluptatem animi.

► la suite...

Réserver

Category:

Partager:

DESCRIPTION

3.6.3 Page Réservation :



adresse de départ

adresse d'arrivée

Date
 Jan 1 2018

Heure départ
 00 : 00

Nombre place

Nombre bagage

Télécharger Votre CPAM (PDF)
 Choisir un fichier | Aucun fichier choisi

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS

```
> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> php bin/console make:crud Reservation
```

Cette commande a créé “ReservationController.php” , le “ReservationRepository.php” et “templates/reservation” .



Il y a deux types de réservations :

- La première, si on réserve un taxi, alors la réservation est faite de la page d'accueil ou de la page Taxis, on peut les trouver sur le NavBar .

Dans ce cas on va chercher deux types d'information,

User et Taxis : \$user=\$this->getUser() ; \$taxi=\$this->getTaxi();

Grace aux jointures entre les trois tables User, Reservation et Taxi

- Un user peu avoir plusieurs Reservation et une Reservation pour un seul utilisateur (ManyToOne).
- Une Reservation pour un seul Taxi et un Taxi peu avoir plusieurs Reservation (OneToMany).

Après je vais insérer ses deux informations dans une nouvelle “reservation” :

```
$reservation= new Reservation();  
$reservation ->setTaxi($taxi);  
$reservation ->setUser($user);
```

- La deuxième si on réserve directement sur l'onglet Reservation , c'est une reservation sans taxi donc l'administrateur il va gérer le chauffeur qui va prendre cette réservation mais on va retourner les informations de user avec \$reservation ->setUser(\$user) ;



ACCEUIL TAXIS RESERVATION BLOG APROPOS CONTACT

- Pour la connaissance d'adresse j'ai utilisé un code JQuery (JavaScript, json) qu'il va aider l'utilisateur à compléter son adresse avec un système d'autocomplète de l'adresse API : “<https://api-adresse.data.gouv.fr/search/?label=>”
- Il s'agit d'une fonction Ajax abrégée, qui équivaut à :

```
$.ajax({ dataType: "json", url: url, data: data, success: success });
```

‘Label’ va permettre d'afficher l'adresse avec la ville, le code et la commune.

Le site principal : <https://api-adresse.data.gouv.fr>

```

$(".adresse1").autocomplete({
    source: function (request, response) {
        $.ajax({
            url: "https://api-adresse.data.gouv.fr/search/?label="+$(".adresse1").val(),
            data: { q: request.term },
            dataType: "json",
            success: function (data) {
                response($.map(data.features, function (item) {
                    return { label: item.properties.label, value: item.properties.label};
                }));
            }
        });
    }
});

```

- La page “Reservation” permet de télécharger un fichier ou une image de l’ordonnance fournie par un médecin pour la réservation d’un taxi conventionné.

The screenshot shows the Kassas TAXI 92 website's reservation page. At the top, there is a logo and a navigation bar with links: ACCUEIL, TAXIS, RESERVATION, BLOG, A PROPOS, and CONTACT. Below the navigation, the page title is "Reserver". The form has fields for "adresse de départ" (85 Rue Nationale 59000 Lille) and "adresse d'arrivée" (58 boulevard). A dropdown menu lists several addresses in Paris and Toulouse. There is also a "Nombre place" field with a dropdown menu set to 00. At the bottom of the page, a portion of the reservation edit controller code is displayed:

```

#[Route('/reservation{id}_edit', name: 'app_reservation_edit', methods: ['GET', 'POST'])]
public function edit(Request $request, int $id/*Reservation $reservation*/, ReservationRepository
ManagerRegistry $manager): Response
{
    $reservation = $manager->getRepository(Reservation::class)->find($id);
    $form = $this->createForm(ReservationType::class, $reservation);
    $form->handleRequest($request);
    if ($form->isSubmitted() && $form->isValid()) {
        $file = $form->get('file')->getData();
        if($file)
        {
            $oldfile = $reservation->getFile();
            if ($oldfile) {
                unlink($this->getParameter(self::PARAMETER) . '/' . $reservation->getFile());
            }
            $fileName = md5(uniqid()). '.' . $file->guessExtension();
            $file->move(
                $this->getParameter(self::PARAMETER),
                $fileName);
            $reservation ->setFile($fileName);
        }
        $om = $manager->getManager();
        $om->persist($reservation);
        $om->flush();
        return $this->redirectToRoute('success', [], Response::HTTP_SEE_OTHER);
    }
}

```

```

    if($user)
    {
        $form = $this->createForm(ReservationType::class, $reservation);
        $form->handleRequest($request);
        if ($form->isSubmitted() && $form->isValid()) {
            $reservation ->setTaxi($taxi);
            $reservation ->setUser($user);
            $file = $form->get('file')->getData();
            if($file)
            {
                $fileName = md5(uniqid()). '.' . $file->guessExtension();
                $file->move(
                    $this->getParameter(self::PARAMETER),
                    $fileName);
                $reservation ->setFile($fileName);
            }
            $om = $manager->getManager();
            $om->persist($reservation);
            $om->flush();
            return $this->redirectToRoute('success', [], Response::HTTP_SEE_OTHER);
        }
        return $this->renderForm('reservation/new.html.twig', [
            'reservation' => $reservation, 'form' => $form,]);
    }
}

```

3.6.4 Page Account:

J'ai créé un Controller "Account" La connexion à l'espace personnelle

```
{ {app.user.firstname|upper} }{ {app.user.lastname|upper} };
```

Contient les onglets suivants :

- Visualiser et éditer les réservations personnelles
- Visualiser et éditer les informations personnelles
- Déconnexion.

- **1. Visualiser et éditer les réservations :**

La jointure entre la table User et Réservation et Grace a cette commande qui va parcourir et afficher les réservations d'un utilisateur,

```
{% if app.user.reservations %}
{% set index = 0 %}

{% for reservation in app.user.reservations %}

    {% set index = index + 1 %}
```



HELLO HINDA BABOURA

From your account dashboard, you can easily check & view your recent orders, manage your shipping and billing addresses and edit your password and account details.

Subscribe Our Newsletter

Enter Email Address

Subscribe

Les chemin URL permet de modifier ou annulé la réservation

Modifier
Annuler

- **2. Visualiser et éditer les informations personnelles :**

J'ai ajouté les informations personnelles dans les inputs :

First Name <input value="{{ app.user.firstname }}">
Last Name <input value="{{ app.user.lastname }}">
User Name <input value="{{ app.user.username }}">
Email <input value="{{ app.user.email }}"> Phone <input value="{{ app.user.phone }}">

- **3. L'utilisateur peut être l'administrateur de site donc :**

```
{% if is_granted('ROLE_ADMIN') %}  
<a href="{{ path('admin') }}" >Access the backoffice </a>{% endif %}
```

Permet à l'administrateur de connecter au Dashboard de BackOffice

3.6.5 Page Contact :

The screenshot shows the Kassas TAXI 92 website. At the top, there is a logo of a taxi and the text "KASSAS TAXI 92". Below the logo, a navigation bar includes links for "ACCEUIL", "TAXIS", "RÉSERVATION", "BLOG", "À PROPOS DE", and "CONTACTER". On the left, a sidebar menu has a green header tab labeled "Tableau De Bord" which is currently selected. Other menu items include "Mes Réservations", "Réserver Maintenant", "Détails Du Compte", and "Déconnexion". The main content area has a title "BONJOUR SARRA ZEMNI KASSAS" and a message about viewing recent commands, managing delivery addresses, and changing the password and account details. A button "Accéder Au Back-Office" is present. Below this, a green banner encourages newsletter subscription with the text "Abonnez-Vous À Notre Newsletter", an input field for an email address, and a "S'abonner" button.

Symfony console make:controller Contact : Trois fichiers vont être créé le “ContactController.php” “ContactRepository.php” et “templates/contact”

\$this->addFlash("contact_success",'Votre message a été envoyé. Un conseiller vous répondra très rapidement !');

En cas d'envoi de contact. Un message de confirmation va être afficher grâce au addFlash()

Et le message va être enregistrer dans la base de données et tous les contacts vont être afficher sur le BackOffice de l'administrateur.

A screenshot of a code editor showing the `ContactController.php` file. The code is as follows:

```
new.html.twig _form.html.twig ContactType.php ContactController.php Contact.php M
c > Controller > ContactController.php > ...
22     $contact = new Contact();
23     $form = $this->createForm(ContactType::class, $contact);
24     $form->handleRequest($request);
25
26     if ($form->isSubmitted() && $form->isValid()) {
27         // $contactRepository->save($contact, true);
28         $om = $manager->getManager(); █
29         $om->persist($contact);
30         $om->flush();
31
32         $user=(new User())
33             ->setEmail('sarraraouf@hotmail.com')
34             ->setFirstname('Taxi Kassas 92')
35             ->setLastname('Réservation Taxi');
36         $emailSender->sendEmailByMailJet($user); █
37
38         $contact=new Contact();
39         $form=$this->createForm(ContactType::class,$contact);
40
41         $this->addFlash("contact_success",'Votre message a été envoyé. Un conseiller vous r
42     }
43
44     if($form->isSubmitted() && !$form->isValid())
45     {
```



ACCUEIL TAXIS RESERVATION BLOG A PROPOS CONTACT



Adresse :
33 Rue Léon Gambetta 22222 Canvas



Email :
contact@kassastaxi92.com



Téléphone :
(+33) 7 22 11 00 60

Nous Contacter :

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus blandit massa enim. Nullam id varius nunc id varius nunc.

Votre message a été envoyé. Un conseiller vous répondra très rapidement ! ×

Nom et Prénom

Email

Téléphone

Objet

Content

Les informations des taxis, les nom des chauffeurs et les réservations sont très confidentielles, c'est pour ça j'ai utilisé la bibliothèque FakerPHP pour remplir des fausses informations dans la base de données.

```
SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS

> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> composer require fakerphp/faker
```

A crée le dossier “DataFixtures”

```
SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS

> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> composer require --dev orm-fixtures

SORTIE CONSOLE DE DÉBOGAGE PROBLÈMES TERMINAL GITLENS

> purging database
> loading App\DataFixtures\AppFixtures
PS C:\Users\ksgka\OneDrive\Bureau\taxi\taxi\taxi92> php bin/console doctrine:fixtures:load
```

Permet de remplir les informations dans les tables de la base de données grâce à la fonction load()

```

> DataFixtures > AppFixtures.php > PHP Intelephense > AppFixtures > load > $name
5     public function load(ObjectManager $manager): void
6     {
7         $faker = Factory::create('fr_FR');
8         for ($i=1; $i <= 30; $i++) {
9             $name=$faker->lastName() . '.' . $faker->firstName();
10            $placemax = $faker->numberBetween(3,7);
11            $bagagemax = $faker->numberBetween(3, 9);
12            $taxi= (new Taxi)
13                ->setName($name)
14                ->setDescription($faker->paragraphs(3, true))
15                ->setIsCpam(true)
16                ->setIsLuxe(@var \Faker\Generator $faker)
17                ->setIsVan($faker->boolean())
18                ->setIsFete($faker->boolean())
19                ->setIsFree($faker->boolean())
20                ->setImage('car' . $i . '.jpg')
21                ->setCreatedAt(new DateTimeImmutable($faker->date()))
22                ->setSlug(str_replace(" ", "-", $name))
23                ->setMoreInformation($faker->paragraphs(3, true))
24                ->setPlacemax($placemax)
25                ->setBagagemax($bagagemax);
26            $manager->persist($taxi);
27        }
28        $manager->flush();

```

4. LE REFERENCEMENT



Pour la partie « référencement », Les stratégies de SEO et d'accessibilité doivent être établis en amont du projet.

Néanmoins, l'optimisation basiques du HTML, partagées par le SEO et l'accessibilité, peuvent déjà améliorer grandement l'exploration du contenu.

En effet, le SEO et l'accessibilité poursuivent le même objectif : un site facilement accessible et compréhensible, et passent par une navigation claire et simple, un temps de chargement rapide, une hiérarchisation des contenus, des titres de pages éloquents et enfin du contenu alternatif lorsque le media ne contient pas de texte. Bien évidemment, **ces exemples ne sont pas exhaustifs**, une mise en place de stratégie d'accessibilité et / ou de SEO dépend des objectifs et du budget alloué, et nécessite une mise en œuvre complète.

5. LA MISE EN LIGNE

Ce site n'est pas encore en ligne ... dès que la société décide de le publier on va choisir un hébergeur.

The screenshot shows the IONOS website homepage. At the top, there's a navigation bar with links like "Domaines & Hébergement", "Cloud computing", "0970 808 911", "Programmes IONOS", and "Connexion". Below the navigation is a main banner with the text "Acheter un site Internet et l'héberger au meilleur prix". It lists three ways to get a website: "Acquérir un site Web existant", "Créer votre propre site avec MyWebsite", and "Faire créer votre site par des professionnels". A button "Voir les packs" is present. To the right of the text is a circular portrait of a smiling man holding a smartphone displaying a website for "Meubles Fabre". Below the banner, a section titled "Il existe de nombreuses façons d'acheter un site Internet : laquelle vous convient ?" provides four options with icons: "Pourquoi acheter ?" (magnifying glass), "Acheter un site Internet existant" (server), "Créer votre site Internet vous-même" (person at computer), and "Faire créer votre site Internet" (person with laptop).

6. DESIGN

6.1 Charte graphique (respect du choix des couleurs, police et choix du logo)

La Charte graphique de site Taxi Kassas 92 suit le modèle et les couleurs des GreenCart écologiques pour être au norme des lois Françaises et la restriction d'utiliser les voitures hybrides et électriques. Et les taxis parisiens en France sont noire ou blanc

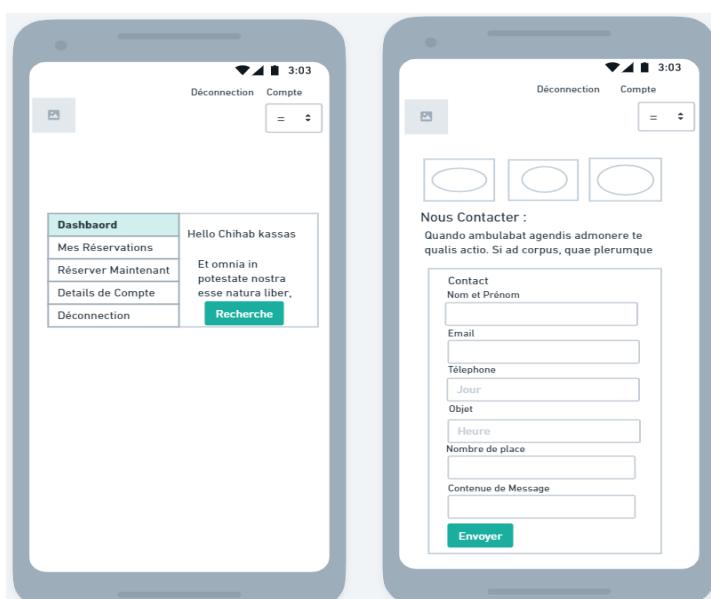
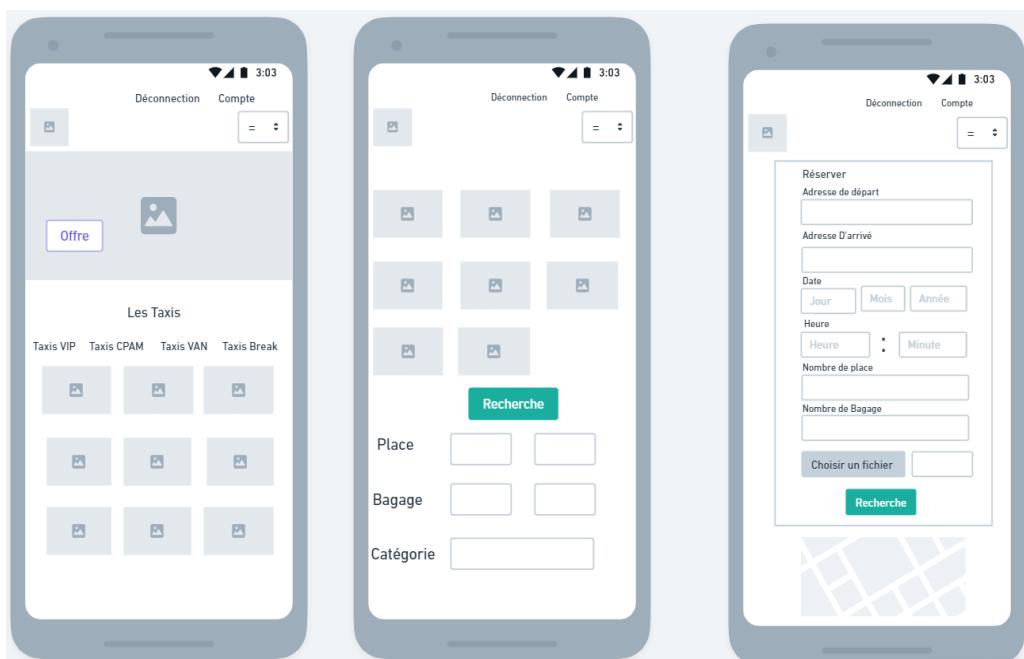
Couleur :



Logo :



6.2 Conclusion avec remerciement et évolution du projet web ou mobile



Mon Taxi Conventionné VSL

[Accueil](#) [Taxis](#) [Réservation](#) [Blog](#) [A propos](#) [Contact](#)

Les Taxis

Taxis VIP	Taxis CPAM	Taxis Van	Taxis Break

The screenshot displays a responsive web design for a taxi service. At the top, there's a header with a logo, a search bar labeled "Recherche", and navigation links for "Déconnection" and "Compte". Below the header are menu items: Accueil, Taxis, Réservation, Blog, A propos, and Contact. The main content area features a grid of vehicle images, including a minivan, a sedan, a black car, and a van with a wheelchair accessibility icon. There are also sections for "Place" (with two input fields), "Bagage" (with two input fields), and "Catégories" (with one input field). At the bottom, there's a teal-colored footer bar with a "S'abonner à notre Newsletter" button, an "Entrer Votre adresse EMail" input field, and a "S'abonner" button.

Le Responsive Web Design permet au site web de s'adapter à la taille de tous les écrans (ordinateur de bureau, ordinateur portable, tablette et téléphone portable).

Pour être sûre que les internautes naviguent correctement sur le site et sur n'importe quel support, nous avons opté pour le framework Bootstrap ainsi que le Module de boîte flexibles (Flexbox).