

MyCovoit

MYCOVOIT

Mail : Adresse mail

Mot de passe : Mot de passe



[MOT DE PASSE OUBLIÉ ?](#)

CONNEXION

[Vous n'êtes pas inscrit ?](#)
Cliquez sur le bouton ci-dessous.

INSCRIPTION

Présentation de l'application :

MyCovoit est une application mobile développée avec le framework Ionic. C'est une application de covoiturage pour les élèves, étudiants, professeurs et personnels du lycée Jean Rostand de Caen.

L'application comporte 8 pages principales : Inscription, Connexion, Tableau de bord, Historique, Rechercher, Proposer, Messages et Profil. Nous avons également d'autres pages avec d'autres fonctionnalités (Noter, Conversation, ...).

L'application est disponible sur serveur en lançant la commande "ionic serve" afin que Ionic construise les pages et les affiche sur un navigateur web, ou nous avons une APK, installable sur mobile.

La navigation des pages se fait à travers une toolbar (barre en bas de l'application) afin d'accéder aux pages Historique, Rechercher, Proposer, Messages et Profil. Nous avons également un menu accessible en glissant la page de plus à gauche vers la droite, ou en allant dans Profil. Ce menu nous donne accès aux pages Tableau de bord, Modifier le profil, Conditions générales et Déconnexion.

compte google:

adresse mail: inforostand14.net@gmail.com

mot de passe : F1r3b@s3

Déploiement sur un serveur:

Démarrer un nouveau projet sur le serveur distant:

Installation Guide

01

Install Ionic

Install [Node.js](#), then install the latest Ionic command-line tools in your terminal. Follow the [Android](#) and [iOS](#) platform guides to install their required tools.

```
npm install -g @ionic/cli
```

New to command line? Read our [Terminal tutorial](#) →

02

Start an app

Create an app using one of our ready-made app templates, or a blank one to start fresh. Check out the [Market](#) for more designs.

```
ionic start myApp tabs
```

03

Run your app

Much of your app can be built in the browser with **ionic serve**. We suggest starting with this workflow. When you're ready, check out our [Deploying](#) guide.

```
cd myApp
ionic serve
```

Générer l'APK:

- Brancher votre appareil Android sur votre ordinateur.
 - Lancez la commande: **ionic cordova run android --prod** (ATTENTION : certains antivirus peuvent bloquer l'installation sur mobile)
- Si vous n'avez pas d'appareil mobile :
 - Lancez la commande : **ionic cordova platform add android** (avec au préalable android studio d'installé)

- **ionic cordova run android --prod**

L'APK, une fois générée, sera dans le répertoire : **platforms\android\app\build\outputs\apk\debug**

Vous pouvez donc copier cette APK et la glisser dans le dossier de votre smartphone sans passer par les commandes.

Page Connexion

Présentation :

La page de connexion (voir ci-dessus), comme son nom l'indique, permet à l'utilisateur de se connecter à l'application via son adresse mail et son mot de passe. Le champ mot de passe est masqué lors de l'écriture mais peut être rendu visible grâce au bouton présent à droite de celui-ci (oeil barré). Un lien cliquable "mot de passe oublié ?" est présent juste en dessous du champ mot de passe et le bouton "connexion" se trouve juste en dessous de celui-ci. Dans la partie basse de la page nous retrouvons le bouton "inscription".

Fonctionnalité :

La "connexion" :

Pour la partie connexion tout est géré avec la fonction **login()** où le mail et le mot de passe de l'utilisateur sont comparés et s'ils correspondent à un utilisateur présent dans la base celui-ci est directement connecté et envoyé vers son tableau de bord personnel. revanche si cela ne correspond pas, un message d'erreur s'affiche grâce à la fonction **errormail()** et l'utilisateur doit saisir à nouveau son email ainsi que son mot de passe.

Page Inscription

Présentation :

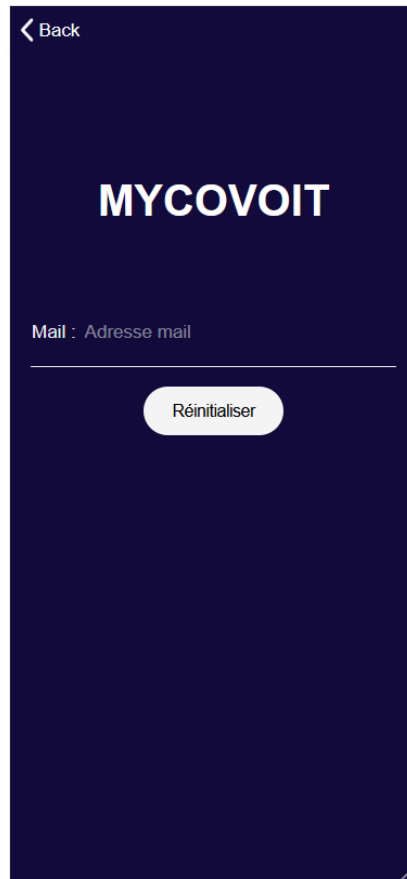
La page Inscription permet à un utilisateur de s'inscrire à l'application. Elle est composée de 4 champs (3 obligatoires et 1 champ pour l'instant non fonctionnel mais nous avons anticipé la possibilité qu'un code soit distribué lors de la distribution de l'application afin de limiter l'utilisation de l'application qu'aux personnes du lycée.), elle est composée également de deux boutons, l'un si un utilisateur est déjà inscrit et donc nous revenons sur la page de connexion, et l'autre permettant après avoir inscrit les données dans les champs, nous dirige vers la page Infos-Perso afin d'inscrire des informations personnelles.

Fonctionnalité :

La fonction principale dans cette page est ***signUp()*** prenant les valeurs dans les champs et vérifiant si les champs sont vides, si les mots de passe sont identiques, si les mots de passe contiennent au moins 1 chiffre, 1 majuscule et 1 minuscule. Nous avons également mis obligatoire une taille de 8 caractères dans le fichier html (min=8).

Une fois les conditions vérifiées, nous pouvons utiliser une méthode que Firestore qui est `createUserWithEmailAndPassword()` qui crée un utilisateur dans la base de donnée de Firestore avec un email comme identifiant et un mot de passe. Nous nous connectons aussitôt et changeons de page ensuite.

Page Mot de passe oublié



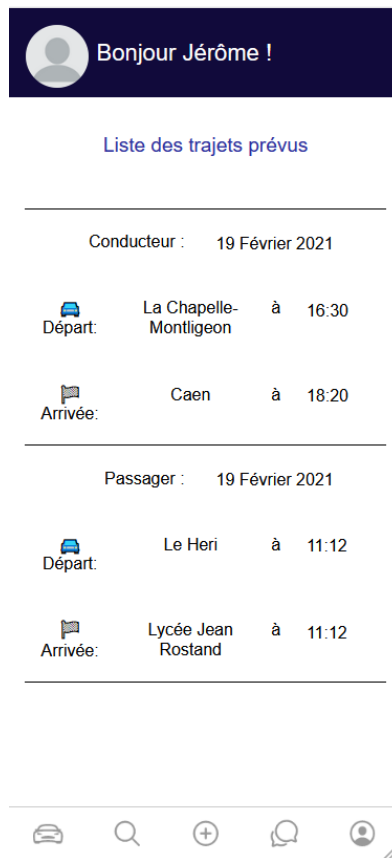
Présentation :

Cette page permet tout simplement à l'utilisateur de réinitialiser son mot de passe en saisissant son adresse mail, qui sera vérifiée. Un mail lui sera alors envoyé contenant un lien lui permettant de changer son mot de passe et il sera redirigé vers la page de connexion.

Fonctionnalité :

Le fonction ***resetEmail()*** contient une fonction de Firestore qui est ***sendPasswordResetEmail()*** qui s'assure que le mail correspond à un utilisateur et envoie le mail a l'adresse indiqué.

Page Tableau de bord



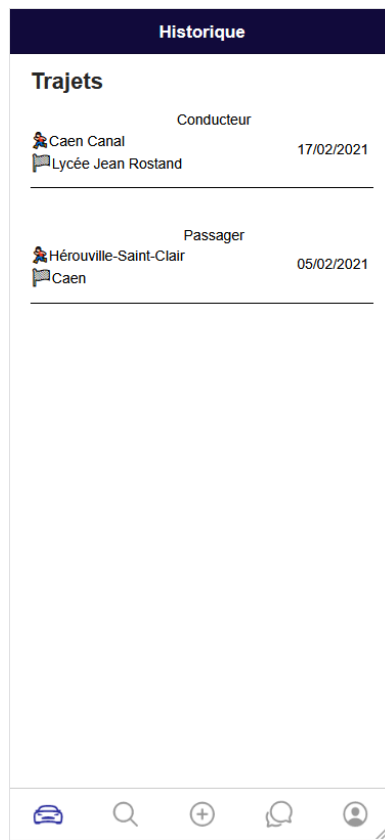
Présentation :

Cette page répertorie tous les trajets prévus, que ce soit en temps que passager ou conducteur. Il est possible de voir le rôle, la date, le lieu d'arrivée et de départ ainsi que l'heure pour les deux. Il est aussi possible d'appuyer sur chaque trajet afin d'avoir des informations complémentaires sur celui-ci. L'en-tête contient la photo de profil de l'utilisateur ainsi que son prénom. Si l'utilisateur n'a pas de trajets prévus deux liens sont affichés renvoyant respectivement vers les pages rechercher et proposer.

Fonctionnalité :

En appuyant sur notre photo en haut de la page nous sommes redirigés vers notre page de profil grâce à une simple fonction **goProfil()**. Le fait d'afficher plus d'information sur les trajets en appuyant dessus est géré par la fonction **goDetail()** qui contient l'id du trajet pour afficher les informations du bon trajet. La récupération des trajets concernant notre utilisateur est effectuée par la fonction **getTrajet()** qui s'occupe de faire une liste des trajets de l'utilisateur parmi tous ceux présents en base en comparant plusieurs informations sur le trajet et l'utilisateur.

Page Historique



Présentation :

Cette page répertorie tous les trajets déjà passés de l'utilisateur et permet d'afficher, comme sur la page tableaubord, le détail des trajets en appuyant dessus.

Fonctionnalité :

Le fait d'afficher plus d'information sur les trajets en appuyant dessus est géré par la fonction **goDetail** qui contient l'id du trajet pour afficher les informations du bon trajet. La récupération des trajets concernant notre utilisateur est effectué par la fonction **getTrajet()** qui s'occupe de faire une liste des trajets de l'utilisateur parmi tous ceux présent en base en comparant plusieurs informations sur le trajet et l'utilisateur. Les pages tableaubord et historique fonctionnent de la même façon.

Page trajet-detail



Présentation :

Cette page permet d'obtenir plus de détails sur le trajet que l'on souhaite. Elle nous permet, si nous venons de la page historique, de noter le trajet auquel nous avons participé. Par contre si la page précédente est le tableau de bord il sera soit possible de supprimer le trajet ou d'annuler sa participation au trajet selon son rôle. Elle

permet aussi de rentrer en contact avec le conducteur du trajet lorsqu'on est passager. Il est possible d'avoir accès au profil du conducteur.

Fonctionnalité :

Les informations concernant le trajet sont récupérées grâce à la fonction **ionViewWillEnter()**. La possibilité de contacter le conducteur est possible avec la fonction **envoyerMessage()** qui récupère l'ID de l'utilisateur ainsi que celui du destinataire pour pouvoir s'adresser à la bonne personne. L'envoi vers la page de notation est gérée par **showTrajet()** qui récupère le trajet correspondant et ouvre un popup permettant de sélectionner la personne que l'on souhaite noter. Enfin la redirection vers le profil du conducteur est gérée avec `[routerLink]='[/profil-autre']` dans le html.

Page infos-trajet


[< Back](#) Rechercher un trajet

Vendredi 26 février 2021


- 07:15 → Rue Louise Michel
↳ (Dives-sur-Mer)
- 07:50 → Lycée Jean Rostand
↳ (Caen)


Etape :
→ Ranville


Conducteur :


 Thibault Brebion
4.67★ / 5 - 9 avis

Contacter Thibault

 Les animaux sont autorisés.

 Fumer n'est pas autorisé.

 Il y a de la musique.

 Le conducteur aime bien parler.

Voiture

Réserver

Présentation :

Cette page permet d'obtenir plus de détails sur le trajet que l'on souhaite. Elle nous permet, si nous venons de la page "rechercher", de réserver le trajet sur lequel nous avons appuyé.. Elle permet aussi de rentrer en contact avec le conducteur du trajet. Il est possible d'avoir accès au profil du conducteur. Une fois la réservation faite, ce trajet est affiché dans le tableau de bord.

Fonctionnalité :

Ici tout se passe à l'identique de la page "trajet-detail" à l'exception de la fonction **reserver()** qui ajoute l'utilisateur au trajet et le passe en passager pour le trajet.

Page Rechercher

Rechercher un trajet

Où allez-vous ?

Lieu de départ :
Ville de départ

Lieu d'arrivée :
Ville d'arrivée

Date de départ :

RECHERCHER

Présentation :

La page Rechercher est divisée en deux parties, une première avec 3 champs (lieu de départ, lieu d'arrivée, date du départ) et un bouton pour exécuter la fonction de recherche, et une autre partie listant les trajets correspondant aux informations que vous avez inscrits avec un bouton retour afin de revenir sur la recherche.

L'application est destinée à effectuer des trajets d'une ville vers le lycée ou inversement. Pour cela, dès lors qu'un champ (départ ou arrivée) est renseigné, le second est directement renseigné par "Lycée Jean Rostand".

De plus, afin d'uniformiser le nom des villes recherchées, les champs pour inscrire la ville de départ ou d'arrivée proposent une liste de trois villes maximum, générée par une API, ayant les mêmes lettres que les informations inscrites.

Fonctionnalité :

Cette page est gérée par deux fonctions principales, en plus des autres fonctions concernant les champs de saisie. La première, **recherche()**, permet de retrouver les trajets correspondants à la recherche saisie. La deuxième, **getImageStorage()**, s'occupe de faire le tri dans les trajets récupérés par la fonction **recherche()**, en n'affichant pas par exemple les trajets déjà réservés.

Page Proposer

Proposer un trajet

Gestion Des Propositions

Itinéraire

D'où partez-vous ? :

Ville de départ

Où allez-vous ? :

Ville d'arrivée

Etapas ?

+

Nombre de place :

Date et horaire

Date de départ :

jj / mm / aaaa

Heure de départ

Heure d'arrivée

PROPOSER

+

Présentation :

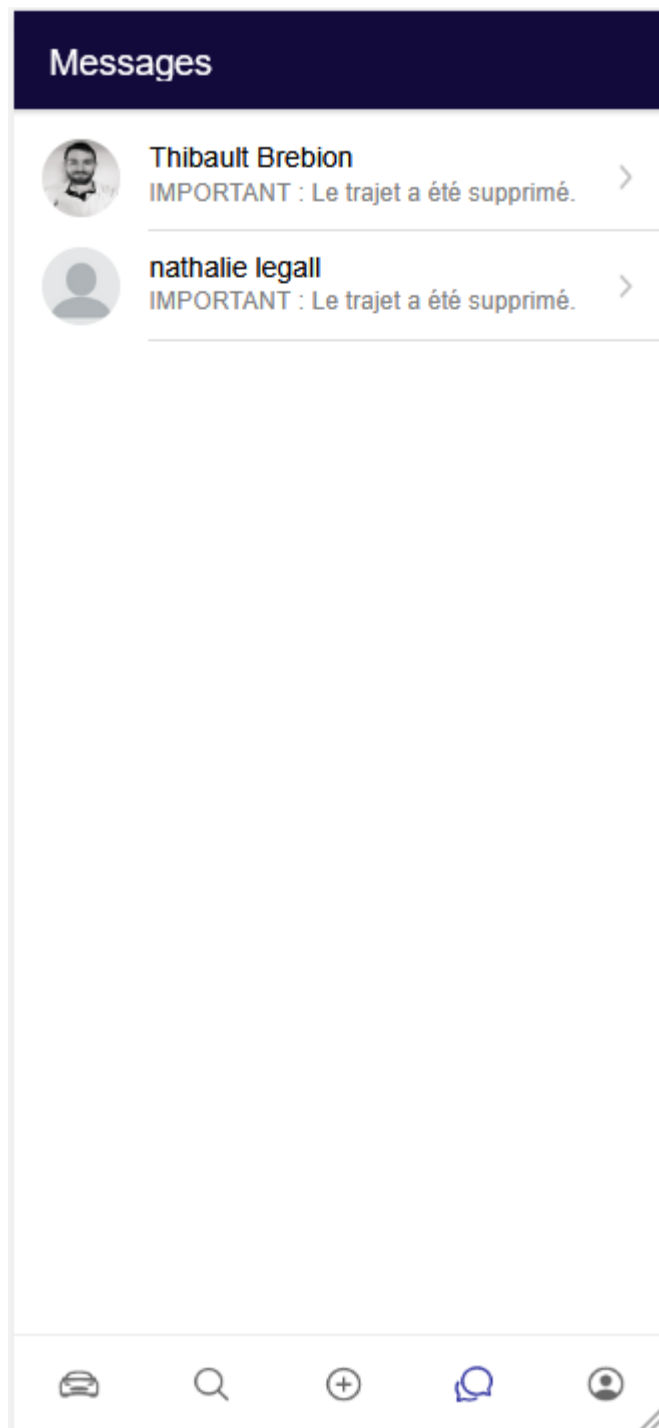
Sur cette page il est possible pour l'utilisateur connecté de proposer un trajet, et comme expliqué dans la présentation de la page de recherche, les trajets ont le 'lycée jean rostand' pour lieu de départ ou d'arrivée et les saisies proposent 3 villes grâce à l'API.

Fonctionnalité :

Cette page est l'une de celles contenant le plus de fonctions. La fonction **presentAlertConfirm()** récupère les informations du formulaire pour les afficher dans un popup de confirmation. Cette précédente fonction lance une autre fonction

nommée **LancerFonction()** qui lancera à son tour plusieurs fonctions qui créeront les informations concernant le trajet dans la base de données.

Page Messages



Présentation :

Cette page rassemble toutes les conversations que l'utilisateur a eu avec les autres utilisateurs avec qui il a eu au moins un trajet en commun.

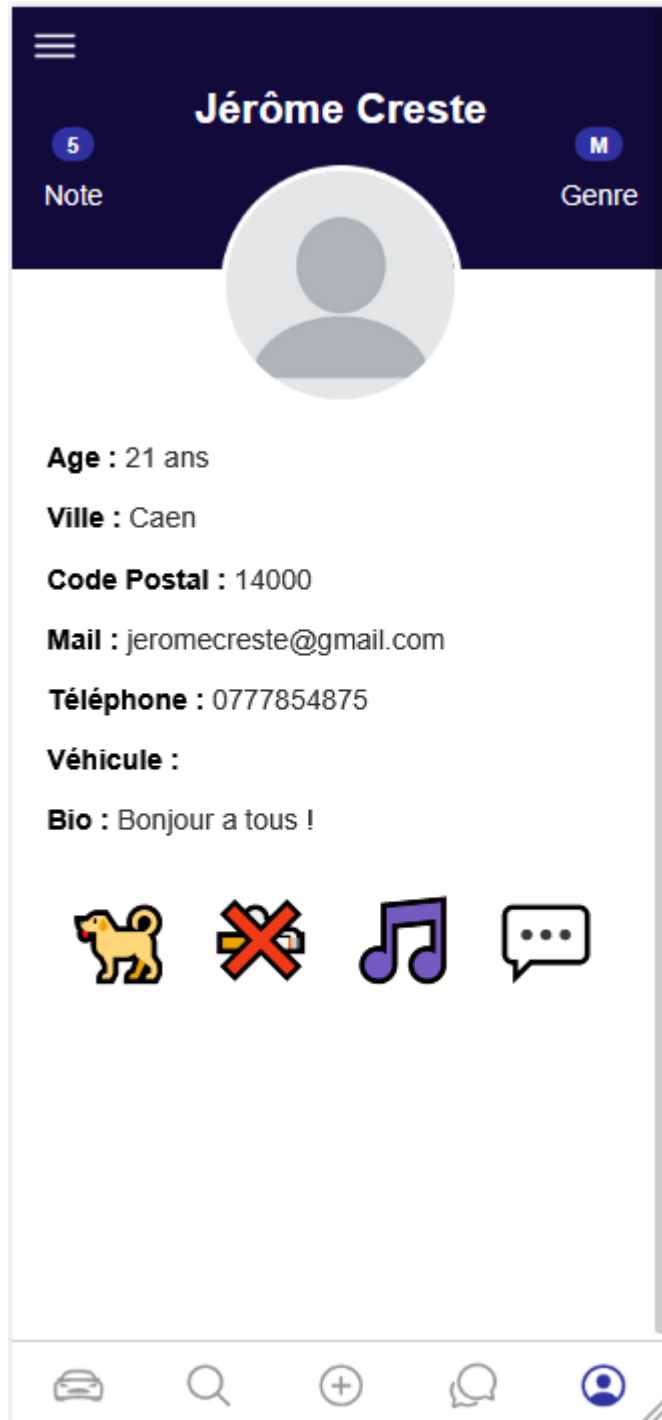
Page Messages



Présentation :

Cette page permet de converser avec la personne de son choix en instantané. Elle offre la possibilité de supprimer un message en maintenant une pression sur le message voulu. Il est possible d'accéder au profil de la personne en appuyant sur la photo présente en haut à droite de la page. Enfin un bouton retour est présent en haut à gauche et nous renvoie sur la page messages.

Page Profil



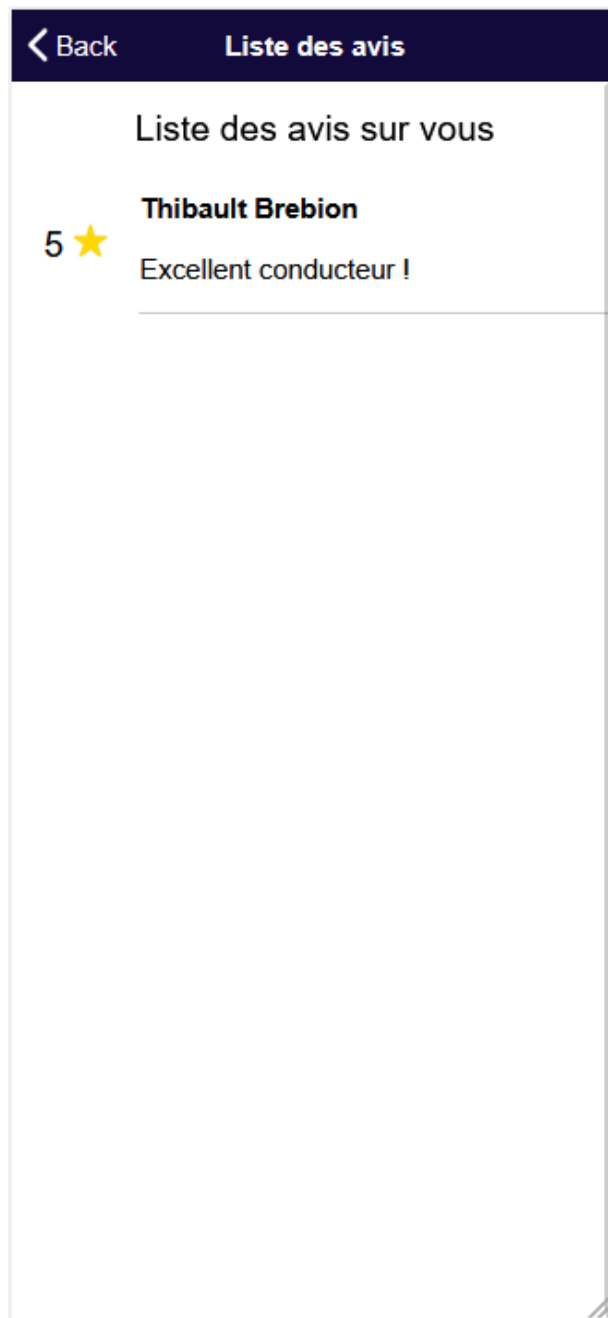
Présentation :

Sur cette page nous pouvons retrouver toutes les informations concernant l'utilisateur étant connecté. Elle regroupe toutes les informations de première nécessité tel que le prénom, l'âge ou la ville par exemple.

Fonctionnalité :

La fonction **getUtilisateur()** définit les informations saisies lors de l'inscription dans des variables afin de les afficher. Une fonction s'occupe de récupérer les avis laissés par les autres utilisateurs afin de pouvoir calculer la moyenne des notes laissées, celle-ci est **getAvis()**.

Page Liste-avis



Présentation :

Pour ce qui est de cette page, on peut y retrouver tous les avis laissés par les utilisateurs ayant eu un trajet en commun avec l'utilisateur concerné, que ce soit un passager ou un conducteur.

Fonctionnalité :

Cette page récupère simplement les avis laissés par les utilisateurs grâce à la fonction **getAvis()**, on se sert ensuite de cette fonction pour afficher les informations souhaitées sur la page .

Page ModifProfil





[Changer la photo](#)

Prénom : 

Nom : 

Naissance :

Genre : 

Ville : 

Code Postal : 

Téléphone :

Véhicule :

Bio :



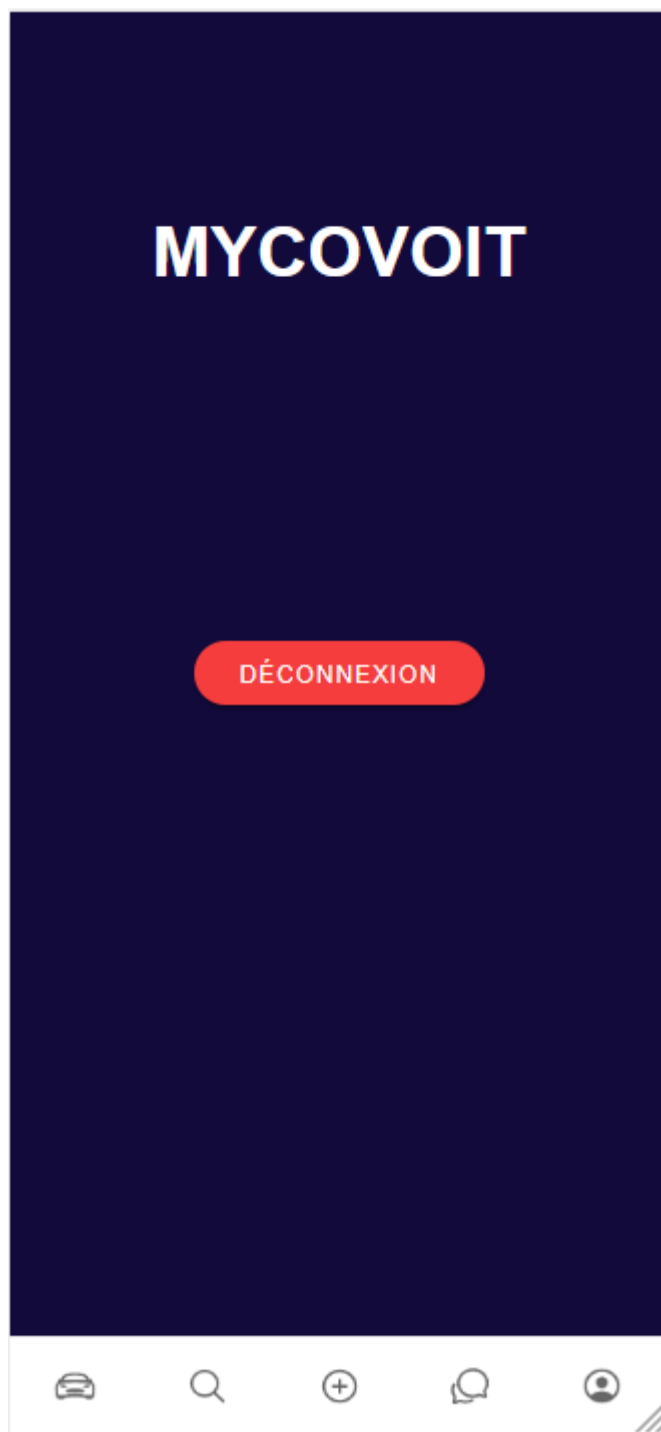
[Modifier](#)



Présentation :

Grâce à cette page il est possible de modifier les informations nous concernant afin qu'elle soit mise à jour sur notre profil.

Page Deconnexion



Présentation :

Cette page sert tout simplement à se déconnecter de l'application afin de revenir à la page de connexion. Elle contient simplement un bouton.

Fonctionnalité :

La fonction **logout()** gère tout ce qui concerne la déconnection de l'utilisateur, c'est-à-dire qu'elle réinitialise les variables de connexions.

MCD:

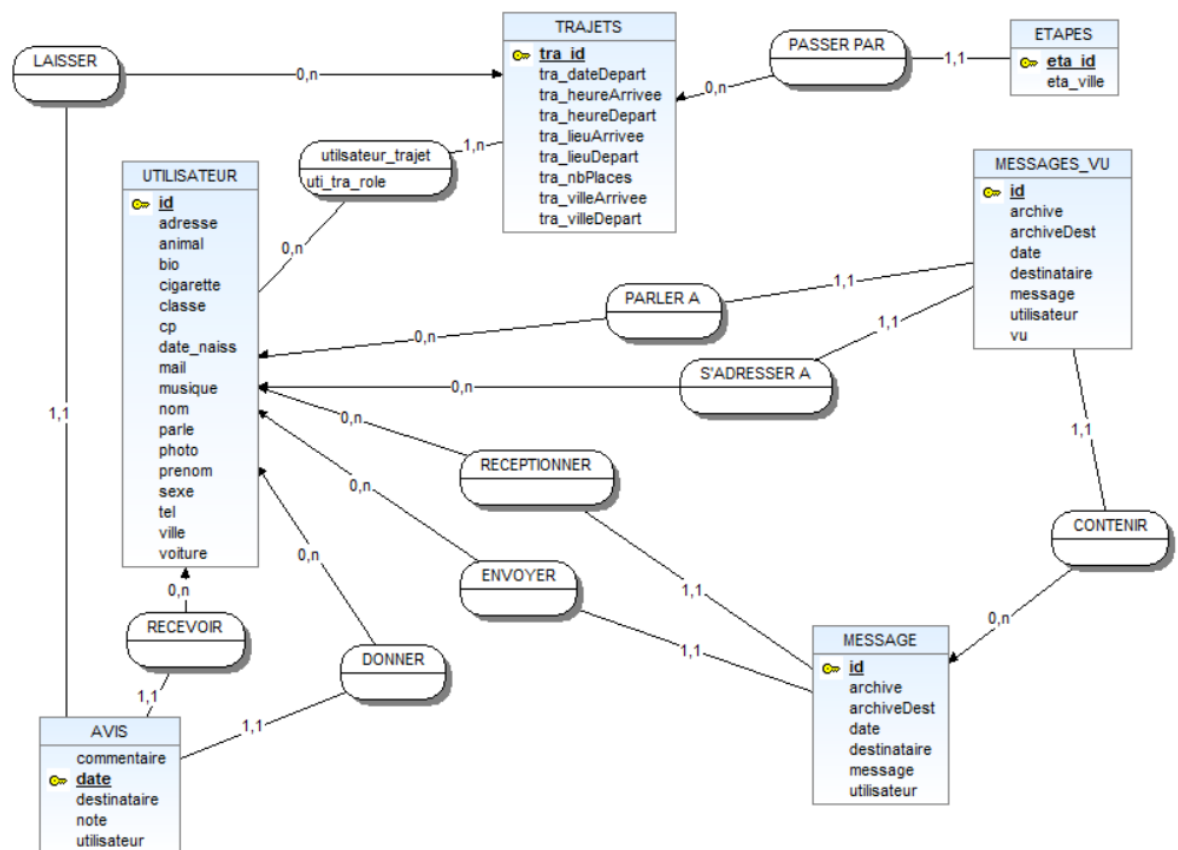


Diagramme de classe:

