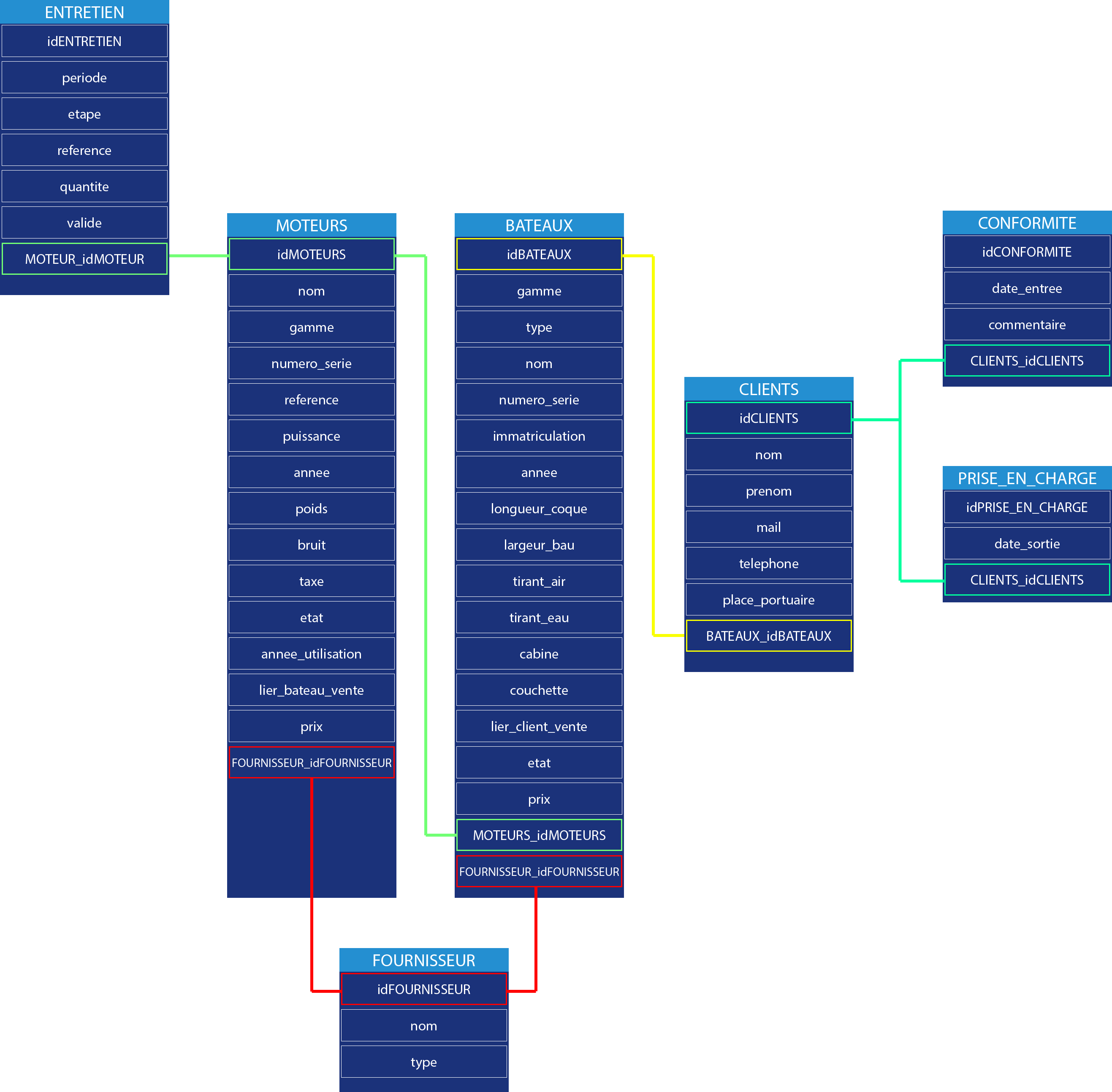
**DOCUMENTATION**

**–**

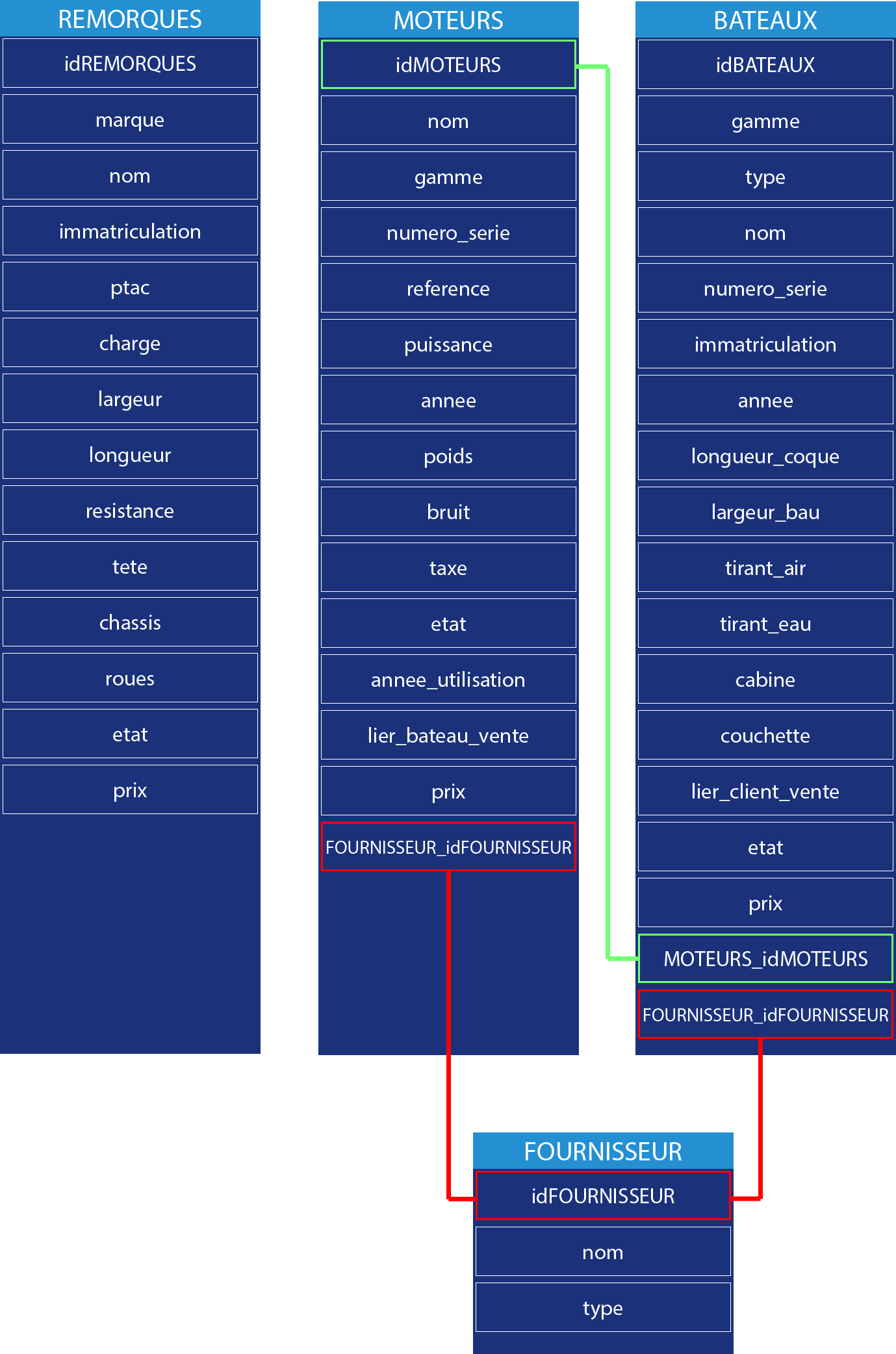
**LE HAVRE NAUTIC SITE APPLICATIF**



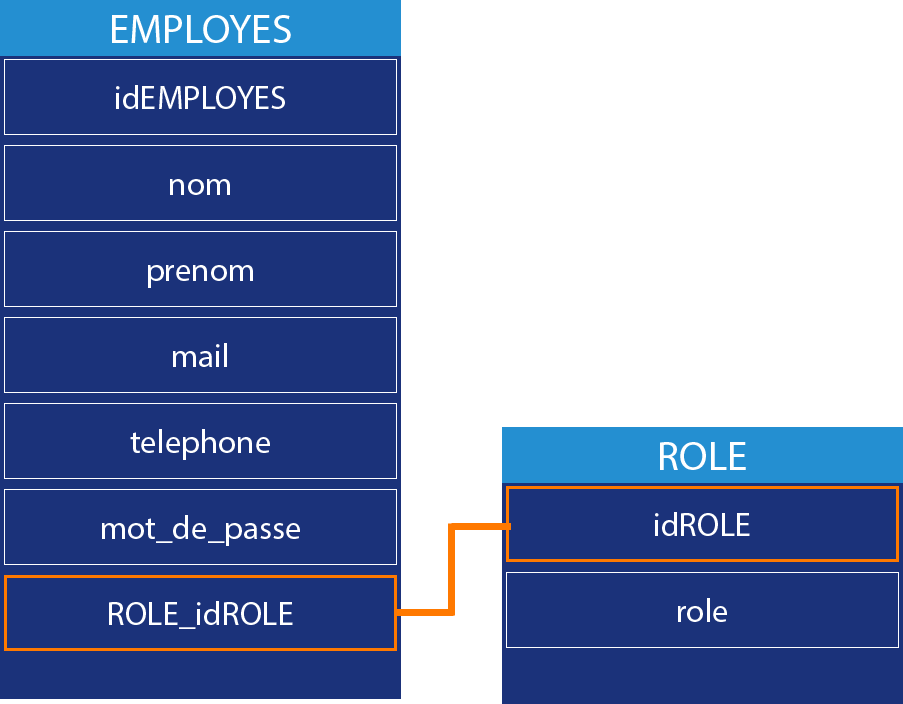
**LIAISONS POUR LES AFFICHAGES D’ENTRETIEN, ENVOI DES CONFORMITE VERS LA PRISE EN CHARGE ET FIN DE LA PRISE EN CHARGE**



**LIAISONS POUR AFFICHER LES NOMS DE FOURNISSEURS**



**LIAISONS POUR AFFICHER LES ROLES**



**FRAMEWORKS & LANGAGES**

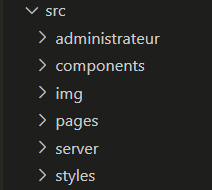
**Front-End**

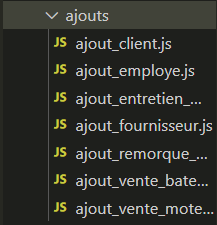
* REACTJS
* JAVASCRIPT
* HTML
* CSS

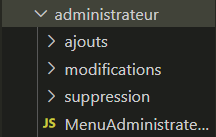
**Back-End**

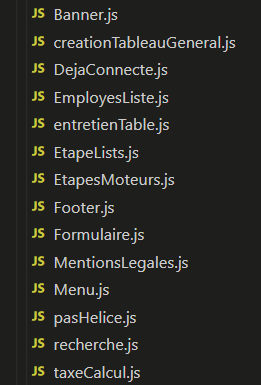
* NODEJS (API REST)
* EXPRESSJS
* MYSQL

**DOSSIERS DE LA STRUCTURE**

Le dossier **src**, qui avec le **framework** **ReactJS** permet de stocker les composants, les feuilles **JavaScript**, les différentes pages, les feuilles de styles ainsi que le dossier du **serveur API**.

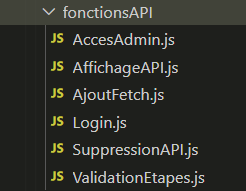
Ici, le dossier **administrateur** comporte le dossier concernant la partie d’ajout, de modification et de suppression vers la base de données.

Il y a également le composant qui permet de générer ce **menu administrateur**. Chaque dossier comporte le même nombre de feuilles **JS**.

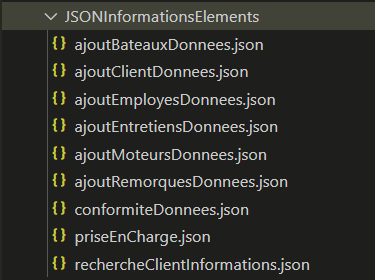
Dans le dossier **components** concerne les composants. On y retrouve le dossier **fonctionsAPI** qui retrouvera les fonctions génériques réutilisables dans les différentes feuilles afin d’éviter la réécriture. **JSONInformationsElements** concerne les informations de header tableau ou d’informations de formulaire afin d’éviter également la réécriture.

Ces composants sont réutilisés à travers les pages qui sont dans un autre dossier, ce qui facilite l’organisation. Voici leurs différentes utilités :

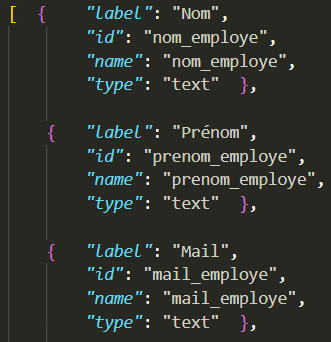
* **Banner.js :** Permet d’intégrer la bannière en entier. Il y a un lien avec la connexion ou déconnexion en fonction de si l’utilisateur est connecté ou non.
* **CreationTableauGeneral.js :** Composant générique qui permet de créer un tableau en réunissant des paramètres et permettant de donner un certain ordre dans le tableau, intégrer des données de base de données et ainsi le rendre générique plutôt que de devoir copier coller à chaque fois.
* **DejaConnecte.js**: Permet d’afficher un message à l’utilisateur sur le formulaire de connexion alors qu’il est déjà connecté.
* **EmployesListe**.js : Permet de lister les employés et le moyen de les contacter, sera réutilisé dans le footer pour les employés qui seront connectés.
* **EntretienTable**.**js :** Permet de mettre en place les tableaux regroupant tous les entretiens des moteurs qui seront affichés grâce à un menu cliquable généré par la base de données. Ainsi, nous pouvons le styliser en fonction des périodes et récupérer tous les entretiens importants.
* **EtapeLists**.**js :** Permet de lister toutes les étapes, faire afficher, lorsque les étapes sont toutes finies, un bouton qui permet de dire que les réparations sont terminées, un bouton qui permet d’afficher ou cacher les étapes également.
* **EtapesMoteurs**.**js** : Permet de lister les étapes liées à un moteur spécifique lui-même lié à un client.
* **Footer**.js : Composant qui permet de créer le footer.
* **Formulaire**.**js :** Composant générique de création de formulaire afin d’éviter les copier coller.
* **MentionsLegales**.js : Composant regroupant les mentions légales et autres informations nécessaires.
* **Menu**.**js :** Composant qui regroupe le menu général composé de :
  + **Client**
    - Recherche de client
    - Vérification client vers atelier
  + **Prise en charge**
  + **Vente** 
    - Bateaux
    - Moteurs
    - Remorques
  + **Moteurs**
    - Menu généré à partir des fournisseurs moteurs dans la base de données
* **PasHelice.js :** Feuille qui regroupe un algorithme qui permet à l’employé de calculer le pas de l’hélice et ainsi retourner des inputs lui permettant d’intégrer les valeurs et d’y voir le résultat, ainsi qu’un bouton pour cacher ou afficher la « calculatrice ».
* **Recherche.js** : Algorithme et composant permettant à l’utilisation d’une barre de recherche qui sera réutilisée dans tout le site et dans tous les sens.
* **TaxeCalcul.js :** Algorithme qui servira à calculer la taxe d’un moteur afin d’avoir le résultat lors de l’ajout d’un moteur dans la base de données.

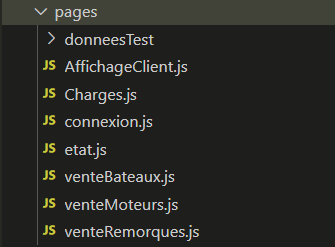


* **AccesAdmin.js :** Permet de récupérer l’accès de l’utilisateur s’il a le rôle d’administrateur afin d’afficher le menu administrateur montré plus haut.
* **AffichageAPI.js**: Fonction générique qui permet d’éviter la réécriture pour afficher les données, fonctionne avec l’id ou sans.
* **AjoutFetch**.**js :** Fonction générique d’ajout ou de modification de données.
* **Login**.**js** : Fonction de connexion.
* **SuppressionAPI**.**js** : Fonction générique de suppression.
* **ValidationEtapes.js :** Fonction de validation d’étapes pour la prise en charge.



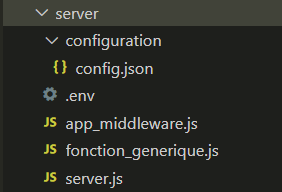
Feuilles JSON qui permet de regrouper des données afin d’éviter de les réécrire. On peut récupérer ainsi un label, un id ou autres. Exemple avec les informations des employés



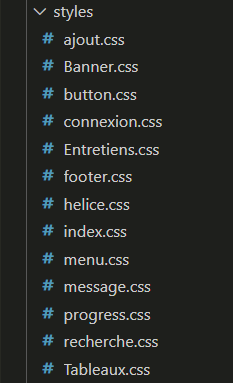


Dans le dossier pages nous pouvons retrouver les composants qui nous permettront à l’affichage de nos pages du site. On importe à l’intérieur les composants du dossier components en fonction de leurs utilités, et nous faisons l’affichage et la structure dans ces feuilles js.

* **AffichageClient.js :** Permet d’afficher le tableau des données client.
* **Charges.js :** Permet d’afficher le tableau de prise en charge
* **Connexion.js :** Permet d’afficher le formulaire de connexion.
* **Etat.js :** Permet d’afficher le tableau de conformité afin d’envoyer l’appareil vers la prise en charge, on y détermine quelle période d’entretien du moteur on doit faire et on y ajoute les commentaires supplémentaires au cas où.
* **VenteBateaux.js, VenteMoteurs.js, VenteRemorques.js :** Les pages qui permettent à la vente d’appareils sous forme de tableaux en regroupant leurs informations.



* **Config.json :** Feuille JSON qui regroupe les informations nécessaires à la liaison du serveur API et la base de données.
* **.env :** Permet de générer le token.
* **App\_middleware.js :** Feuille qui regroupe les fonctions middleware de manière exportée au serveur.
* **Fonction\_generique.js** : Fonctions générique nécessaires au serveur pour les ajouts, modifications, affichages et suppressions.
* **Server**.**js :** Serveur API REST qui regroupe toutes les requêtes SQL.

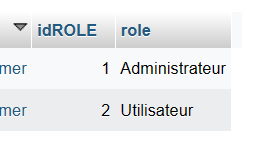


Voici les différents feuilles de CSS, les titres sont assez indicateurs sur le composant ou la page qu’elles touchent.

**Ces pages sont toutes commentées afin de mieux comprendre les lignes qui sont écrites.**

Respecter l’écriture du nom / prenom dans la table des employés en attribuant le rôle administrateur.





Respecter l’id et les écritures de ces lignes :

Table bateaux respecter la gamme, l’idBATEAUX et le lier\_client\_vente :

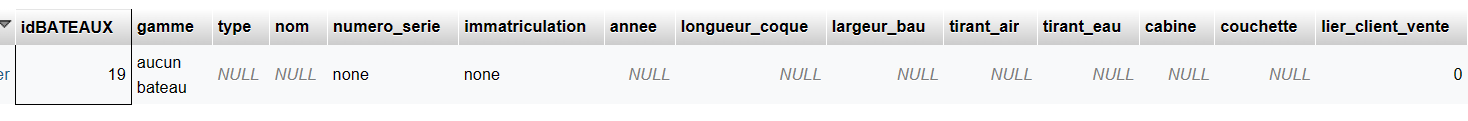
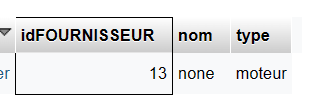
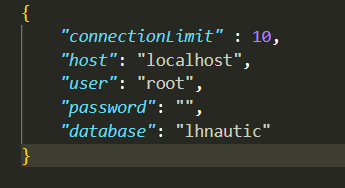


Table moteurs respecter le nom, l’idMOTEURS ainsi que le lier\_bateau\_vente :



Table fournisseur :





Mettre à jour ces configurations qui se trouvent dans le dossier src – server – configuration, en fonction des informations de la base de données réelle.