Projet info0303

Site web concernant les bateaux

Benjamin QUINET  
Nathan TONNELLE

Table des matières

[Introduction 3](#_Toc27286415)

[Objectifs de la réalisation 3](#_Toc27286416)

[Analyse des besoins 4](#_Toc27286417)

[Design du dispositif 6](#_Toc27286418)

[Choix de développement 7](#_Toc27286419)

[Problèmes rencontrés 8](#_Toc27286420)

[Conclusion 9](#_Toc27286421)

# Introduction

L’association de plaisanciers « ÔMONATÔÔ » regroupe des propriétaires de bateaux ayant décidé de mettre en relations des personnes souhaitant faire de la navigation avec ces derniers. Afin de posséder une sécurité optimale pour les sorties des bateaux, L’associations a décidé de mettre en place un outil qui permet d’informer les différents propriétaires si des équipements de sécurité obligatoire doivent être changés ou révisés dans les 15 jours suivant la notification, ainsi qu’à les aider à gérer l’entretien mécanique de leur bateau.

Pour le sujet présenté, nous mettrons en place l’interface web nécessaire à l’utilisation de l’outil dans le cadre de notre projet d’Info0303.

# Objectifs de la réalisation

L’objectif principal est de fournir aux utilisateurs un espace dans lequel ils pourront mettre les informations concernant leurs bateaux et recevoir les différentes notifications si le besoin d’entretien se fait voir.

L'application se fait avec Laravel 6 qui nous permet de faire énormément de choses tel que gérer des utilisateurs, la création d'une base de données, de faire des échanges entre cette base et le site internet, par le biais de formulaires. Ainsi nous pouvons créer des Bateaux, les modifier, les mettre dans une corbeille ou les supprimer définitivement de la base de données.

# Analyse des besoins

Lors du projet d'INFO0304, nous avons vu comment nous devions gérer la base de données de l'association. Nous avons vu quels liens devait être établi entre les différentes tables. Pour le site, nous avons fait le choix de faire les tables suivantes : Users, Boats, Categories, SecuriteObligs, RevisionPeriods, Moteurs, Reservoirs, EntretienCourants et Pieces.

Lorsqu'un utilisateur, autre qu'administrateur du site, se connecte, il ne doit pouvoir accéder à aucuns autres bateaux que les siens, et donc ne peut ni les voir, ni les modifier. Alors que l'administrateur, lui possède tous les droits sur les bateaux, ainsi il peut les voir, les modifier ou les supprimer, définitivement ou non.

Nous allons voir dans le schéma suivant comment les tables communiques ente elles :

Boats

Users

Categories

SecuriteObligs

RevisionPeriods

EntretienCourants

Pieces

Reservoirs



Moteurs

Ce schéma ne comporte des relations ni de n:n ni de 1:1, car nous en avons pas eu besoin.

Pour faire de manière plus écrite :

* Un utilisateur peut avoir plusieurs bateaux mais un bateau ne peut avoir qu'un seul utilisateur (nous n'avons réussi à faire une relation n:n)
* Un bateau possède une catégorie et une seule, tandis qu'une catégorie possède plusieurs bateaux
* Un bateau possède plusieurs objets de Sécurité obligatoires, ainsi que plusieurs réservoirs (dont un minimum de 2, le premier pour l'essence et le second pour l'eau)
* Un bateau possède également plusieurs pièces, qui elles-mêmes possèdent chacun au moins un entretien courant
* Un bateau possède aussi 1 ou plusieurs moteurs, qui ont eux-mêmes au moins une révision Périodique

# Design du dispositif

Parlons design, nous sommes restés sur des page naturelle, sans grande sophistication, juste l'essentiel pour la bonne compréhension de l'utilisateur. Ainsi nos pages sont restées blanche en fond et il y a que les informations essentielles dont les utilisateurs auront besoin.

Pour chacune des tables suivantes, nous avons créé plusieurs pages, pourquoi nous demanderiez-vous ? Car tout simplement cela facilite la compréhension autant pour le développeur que pour l'utilisateur.

Pour chacune de ces tables : boats, moteurs, pieces, reservoirs et securiteObligs, nous avons 4 pages, avec chacune ses fonctionnalités propres.

Les pages index servent à afficher tous les éléments que la table possède. Elle sert aussi a pouvoir créer, supprimer, modifier ou regarder les éléments de la table ne détails, car oui nous n'affichons pas toutes les informations sur cette page, tout cela gâcherai toute la simplicité de lecture des pages et ainsi les pages show n'existeraient pas. Vous pouvez voir au-dessus du tableau de valeurs de la table un menu déroulant nommé "Créations" qui vous permet d'accéder à la page de création de n'importe quel objet (bateau, piece, moteur ou reservoir), ce menu vous permet également d'accéder a la page création qui vous affiche les objets que vous pouvez créer, comme avec le menu déroulant.

Les pages create servent quant à elles, comme leurs noms l'indique à créer l'objet dont il est question. Ce sont des pages avec des formulaire pour faire entrer ledit objet, une fois tout les champs remplis, dans la base de données.

Les pages edit servent à modifier l'objet sur lequel le bouton était associé, vous verrez donc les informations apparaître lors du chargement de la page et ainsi vous pourrez les modifier, attention les modifications ne se confirmeront que si vous appuyez sur le bouton en bas du formulaire.

Les pages show servent à détailler l'objet sur lequel vous avez appuyés, il vous montre toutes ces caractéristiques, chaque colonne de la Base de données où il est situé. Par exemple pour un bateau vous verrez s'afficher les éléments suivants : le nom, son immatriculation, son modèle, son année de construction, ses dimensions, le nombre de personnes qui peuvent monter à bord, le poids en tonnes, son nombre de moteur, la quantité maximale qu'il peut accueillir en eau et en essence.

# Choix de développement

Nous avons fait le choix de faire le seeder a la main, c’est-à-dire sans les factory, bien qu'ils soient créés et développé, pour la simple et bonne raison de donner plus de réalisme à notre base de données, ainsi nous avons des pièces avec des noms cohérant avec la vraie vie, comme l'hélice, la poulie, etc.

Nous avons aussi fait le choix d'utiliser blade et un template pour toutes nos pages, car elles se ressembles toutes et cela allège énormément le code, par exemple pour afficher une valeur d'une case d'un objet nous faisons : *{{$objet->nomCase}}* ce qui est nettement plus court et plus compréhensible dans un code.

Nous avons fait également le choix du nombre d'objet présent en une seule page, ainsi lorsque nous voulons afficher une page index qui comporte plus de 5 objets, une pagination est créée et seulement 5 objets sont affichés.

# Problèmes rencontrés

Nous avons rencontré différents problèmes avec laravel 6 et les tables de migrations, notamment avec les noms des différentes tables car nous ne savions que laravel rajoutait un 's' a la fin des noms des tables.

La mise en œuvre des créations, modifications des données de la base de données.

Nous avons utilisé des méthodes pas tout le temps bien expliquer même dans le cours Larevel 6 de sillo.

# Conclusion

Pour conclure, ce projet fut très long à mener à bien, même s'il ne sera encore pas fini d'être debugger au moment où j'écris ces quelques lignes. Le sujet était simple et clair, et comme nous avions déjà dû rendre la partie INFO0304 avant de réellement commencer le projet nous avions déjà une idée de ce qu'il fallait faire.

Nous vous remercions, vous lecteur(lectrice) de ce rapport.

Nous remercions également Mr. Alin pour ces cours passionnant.