



RAPPORT DE PROJET INFO 303

PHP, SQL ET FRAMEWORK LARAVEL

ANTOINE THEBAULT, BOISSIER SEBASTIEN

**L2 - S3F3A
209 - M.ALIN**

Sommaire

Sommaire	2
Cahier des Charges et Objectifs	3
Description.....	3
Maquettes	3
Architecture	9
Gestion de l'administration.....	9
Organisation dans Laravel	9
Modélisation et interactions	10
Base de Données	14
Eléments graphiques.....	14
Exemples de scripts	15
Lancement du projet	18
Date de rendu	18
Déroulement.....	18
Suivi du projet.....	18
Git.....	18
Difficultés	19
Installation.....	19
Conclusions.....	20
Bibliographie.....	21

Cahier des Charges et Objectifs

Description

« Le but du projet est de développer une application à destination de propriétaires de bateaux. Elle doit leur permettre de gérer l'entretien de leur flotte. Un bateau possède un ou plusieurs propriétaires »

L'objectif du site est d'utiliser les technologies PHP et SQL, à l'aide d'un framework Laravel en respectant le modèle MVC. Celui-ci devra permettre de créer, écrire et modifier une base de données contenant notamment les informations suivantes :

- Des Utilisateurs
- Des Bateaux
- Des Equipements
- Des Pièces (de bateaux ou d'équipements)
- Des Fournisseurs
- Des Entretiens

Nous avons réfléchi à utiliser une table « Caractéristiques » pour un bateau, permettant de rendre génériques certains attributs. Cependant, les différences de type (String, Int, Booléen, Float) nous ont empêchés de procéder ainsi, rendant la table Bateau assez conséquente.

Afin de simplifier l'utilisation par les utilisateurs, nous avons décidé de reprendre l'organisation de l'administration issue du projet0304, touchant à la même association et ayant donné lieu à la base de données qui sera utilisée ici. Les utilisateurs ne rentreront ainsi pas directement les informations (évitant d'éventuelles failles de sécurité) et une vérification devra être faite par un administrateur avant inscription via des formulaires dans la base.

L'utilisateur pourra accéder à ses bateaux (et uniquement les siens), les consulter, demander une mise à jour ou ajouter de nouveaux bateaux. Il pourra également voir si ses bateaux peuvent naviguer et obtenir une liste de fournisseurs pouvant fournir les pièces ou équipements désirés.

Maquettes

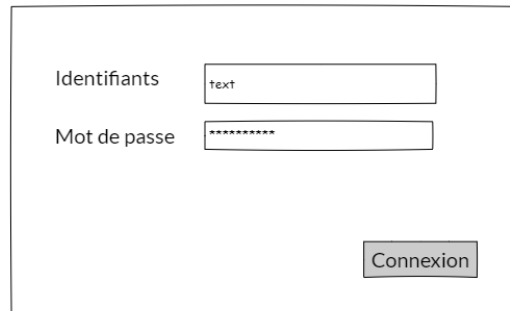
Afin de mieux représenter les éléments et fonctionnalités à développer, nous avons conçu plusieurs maquettes à l'aide du logiciel « Pencil ».

Une partie des utilisations et besoins est également précisée. L'ensemble sera établi dans la partie « Architecture ».

Page de connexion

Générée automatiquement par Laravel, la page permet de rentrer ses identifiants (« login » sur la base de données, unique par utilisateur) et son mot de passe (« password » sur la base de données). Le site appartenant à une association (« Ohmonbatôô »), l'adhérent doit avoir été au préalable inscrit par l'administrateur ou la secrétaire/helper.

Logo



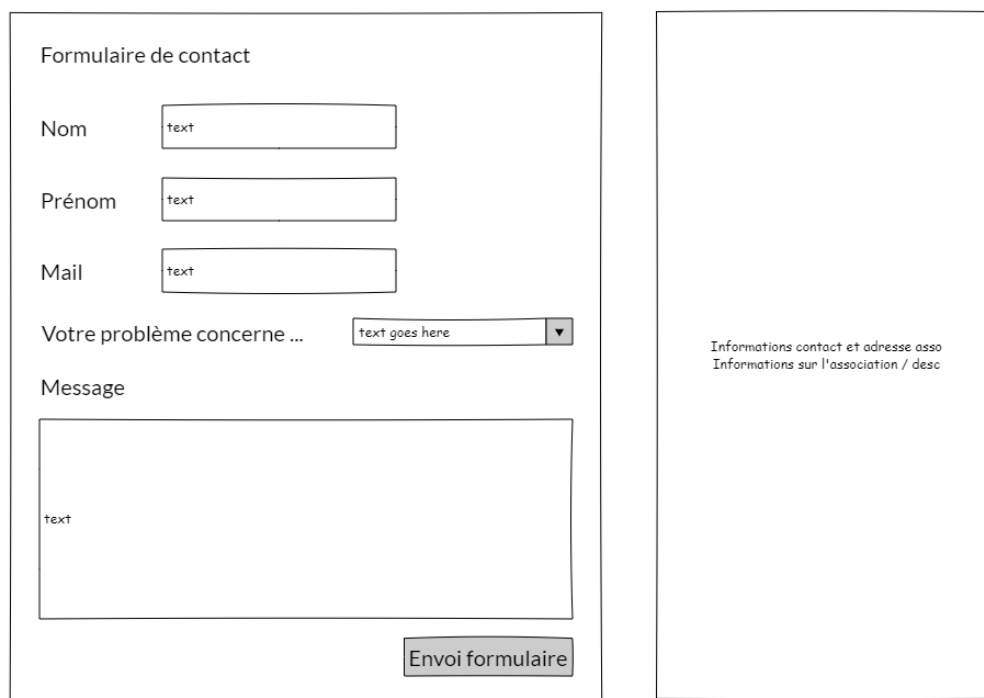
A login form diagram with a rectangular border. Inside, on the left, are the labels 'Identifiants' and 'Mot de passe'. To the right of 'Identifiants' is a text input field containing the placeholder 'text'. To the right of 'Mot de passe' is a password input field containing eight asterisks '*****'. In the bottom right corner of the form is a button labeled 'Connexion'.

Page de Contact

Cette page permet aux adhérents non-inscrits de rentrer en contact avec l'administrateur via des Messages (qui devront être ajoutés à la base de données). Les visiteurs pourront aussi utiliser le formulaire.

NOM_SITE

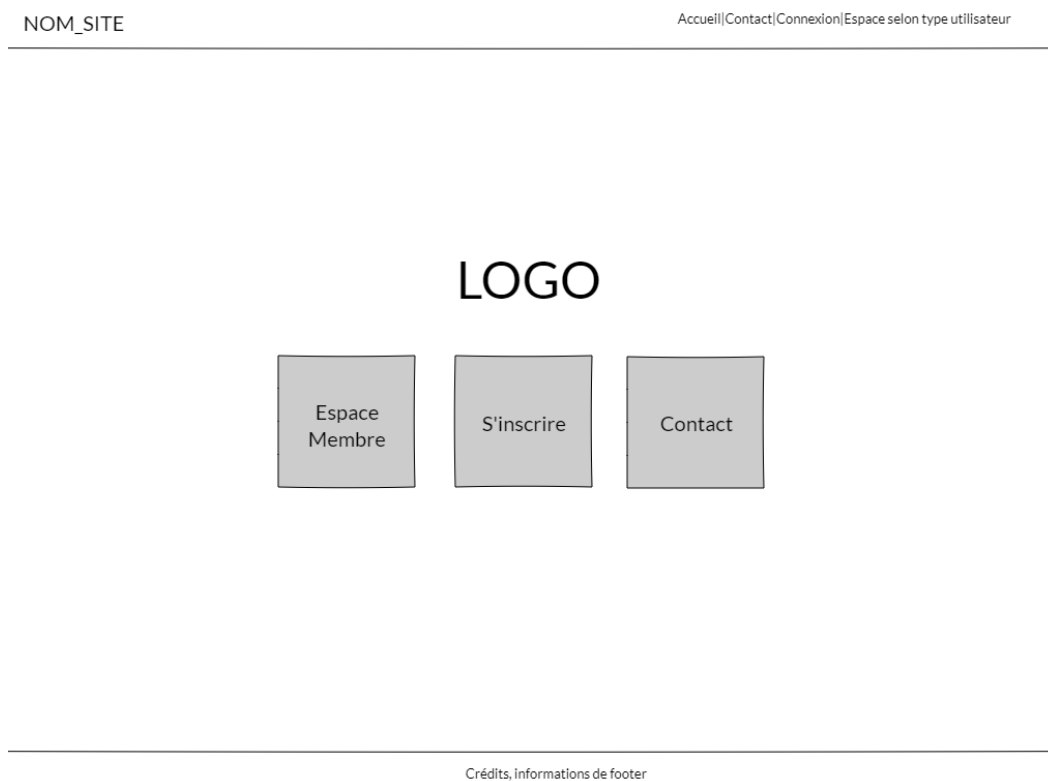
Accueil|Contact|Connexion|Espace selon type utilisateur



The contact page layout is divided into two main vertical sections. The left section is titled 'Formulaire de contact' and contains several input fields: 'Nom' (text), 'Prénom' (text), 'Mail' (text), and 'Votre problème concerne ...' (a dropdown menu with the placeholder 'text goes here'). Below these is a 'Message' section with a large text area containing the placeholder 'text'. At the bottom of this section is a button labeled 'Envoi formulaire'. The right section is a vertical rectangle containing the text 'Informations contact et adresse asso' and 'Informations sur l'association / desc'.

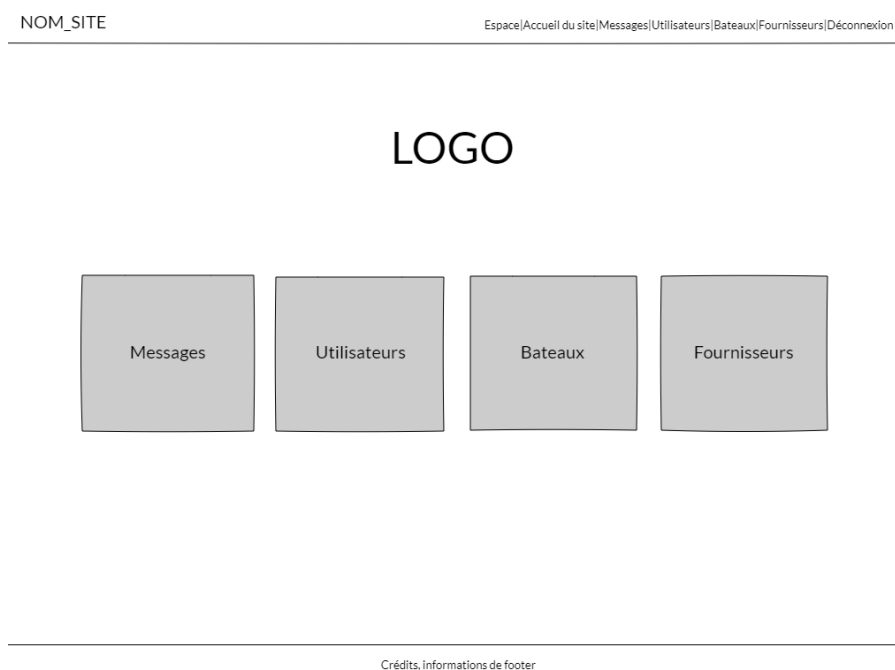
Page d'Accueil

L'interface est très simple et permet d'accéder à toutes pages du site ne nécessitant pas de connexion.



Panel

Reprenant l'organisation de la page d'accueil, le panel s'adapte au type d'utilisateur permettant par exemple d'accéder aux messages, utilisateurs ou à ses bateaux.



NOM_SITE

Espace/Accueil du site/Fournisseurs/Mes Bateaux/Déconnexion

LOGO



Crédits, informations de footer

Pages de Liste

Les Utilisateurs, Bateaux, Messages et Fournisseurs sont présentés sous forme de liste, permettant de voir rapidement les informations principales d'un élément, y accéder ou le supprimer. Les boutons devront s'adapter selon le type d'Utilisateur.

NOM_SITE

Espace/Accueil du site/Messages/Utilisateurs/Bateaux/Déconnexions

Bateaux

Nom	Propriétaire	DATE D'INSCRIPTION	
Navire	John Doe	16/08/2019	 



Crédits, informations de footer

Fournisseurs

NOM	DATE D'INSCRIPTION	
Doe	16/08/2019	 

Crédits, informations de footer

Utilisateurs

NOM	PRENOM	DATE D'INSCRIPTION	ID ADHERENT	
Doe	John	16/08/2019	1	 

Crédits, informations de footer

Pages de description


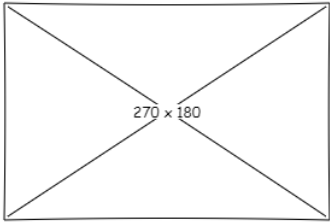
Elles permettent d'afficher les caractéristiques et sous-éléments d'un objet donné. La page d'un Bateau listera donc ses équipements, pièces, éventuel moteur et entretiens, tout en indiquant s'il lui est possible ou non de naviguer.

Des boutons dépendant du rôle de l'utilisateur seront disponibles (Mettre à jour un bateau, Ajouter un bateau, Supprimer)

NOM_SITE

Nav_Bar

Bateau : NOM_BATEAU

NOM_proprio	Etat 
Date ajout	
	Prochaine révision : XX/XX/XXXX Dernière révision : XX/XX/XXXX
Caractéristiques	
=> Catégorie : A	
=> Distance d'éloignement : X km	
=> Nombre d'accidents (etc.)	
Equipements et pièces	
=> L'équipement (Dernière révision : XX/XX/XXXX, prochaine révision : XX/XX/XXXX)	
=> Pièce X : (Dernière révision : XX/XX/XXXX, prochaine révision : XX/XX/XXXX)	
=> Pièce X (Dernière révision : XX/XX/XXXX, prochaine révision : XX/XX/XXXX)	
=> Moteur (Dernière révision : XX/XX/XXXX, prochaine révision : XX/XX/XXXX)	

Crédits, informations de footer

Architecture

Gestion de l'administration

Les utilisateurs du site seront définis en 3 catégories :

- Les adhérents (Type 0)
- Les helpers/secrétaires (Type 1) : Chargés d'inscrire les utilisateurs.
- Les administrateurs (Type 2) : Chargés d'inscrire et valider les bateaux, ainsi que les fournisseurs.

Cette division permet d'éviter d'avoir des accès directs à l'inscription dans la base de données par les utilisateurs. Laravel donne une certaine sécurité, mais il est risqué de permettre l'édition d'autant de données, d'autant que le but du site est de gérer facilement sa flotte.

Organisation dans Laravel

Nous avons besoin pour le développement du projet de différentes fonctionnalités du Framework dont entre autres :

- La gestion des Routes
- La gestion des Views
- La gestion de la base de données, par des migrations, Seeders, classes de Table et Controllars
- La gestion de l'authentification
- La gestion du stockage des fichiers envoyés par l'utilisateur

Views

Elles correspondent à la partie Front-end de l'application, utilisant de l'HTML et du CSS. Il s'agira d'utiliser les données traitées dans les controllers pour afficher les informations à l'utilisateur.

Routes

Elles permettent la redirection et le transfert des informations via les méthodes POST et DELETE. Regroupées dans le fichier *web.php*, elles seront divisées en 3 catégories, selon le type d'utilisateur et ses besoins.

Elles devront être protégées par un middleware, empêchant un accès sans connexion ou sans le rang d'administration requis.

Migrations

Cette fonctionnalité permet la mise en place de la base de données en tant que structure et correspond aux « CREATE TABLE » en SQL.

Seeders

Via l'utilisation du module Faker, ils permettent de générer de fausses informations de test dans les bases de données obtenues avec l'opération « migrate ».

Classes de Table ou Models

Elles permettent de lier plus efficacement et sans passer par des Query directe Tables et Programmation PHP, ainsi que d'établir les relations via les méthodes de classe.

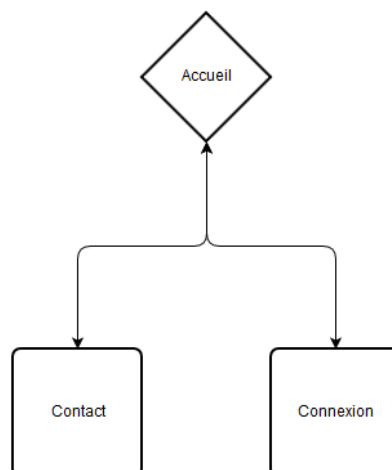
Controllers

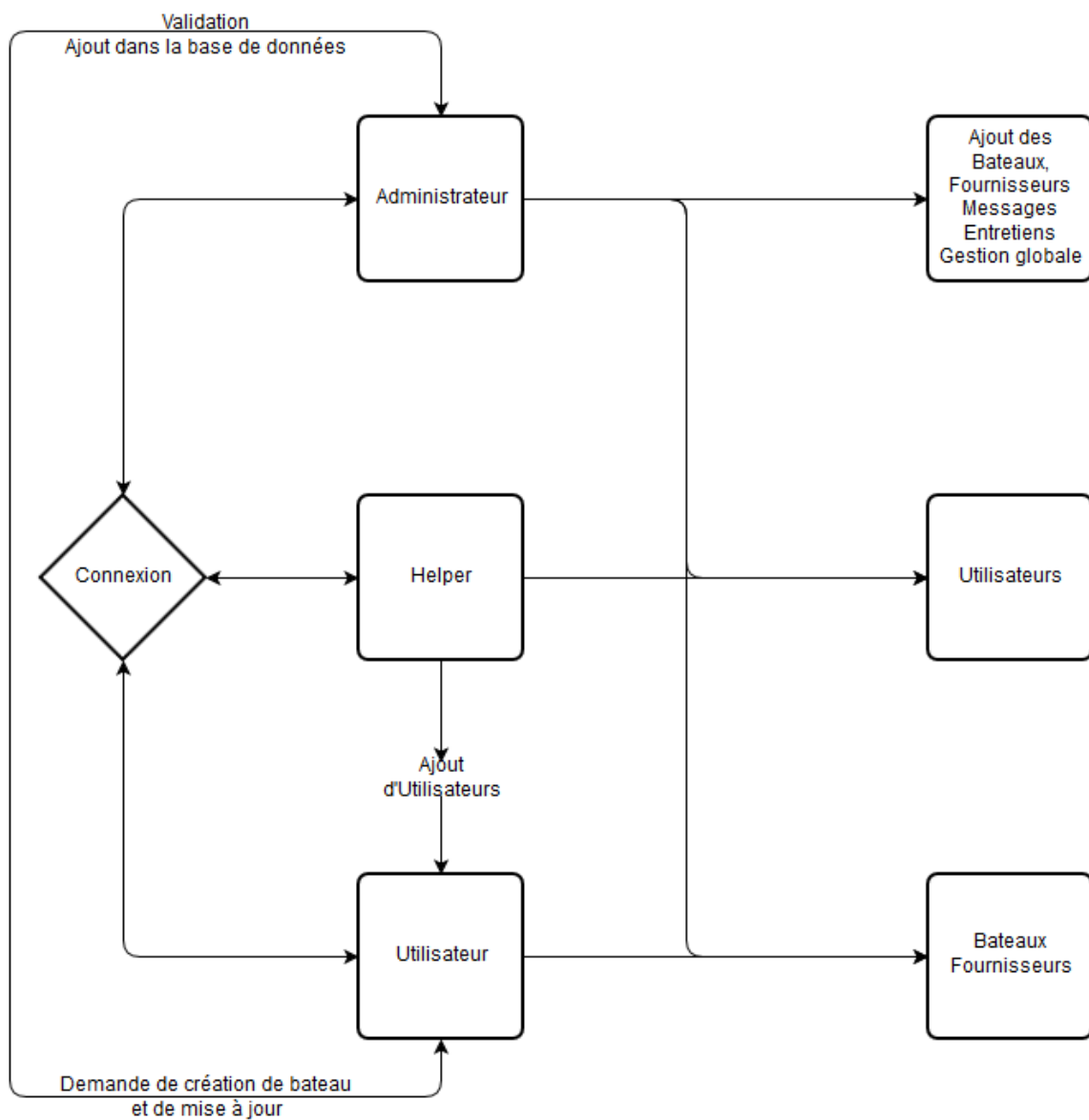
Appelés par les Routes, ils permettent de récupérer les données via les classes de Table et d'envoyer les données traitées vers les views ou la base de données. Plutôt que d'avoir un grand nombre de Controllers, nous avons regroupés les actions dans un nombre plus réduit, comptant les Tables principales :

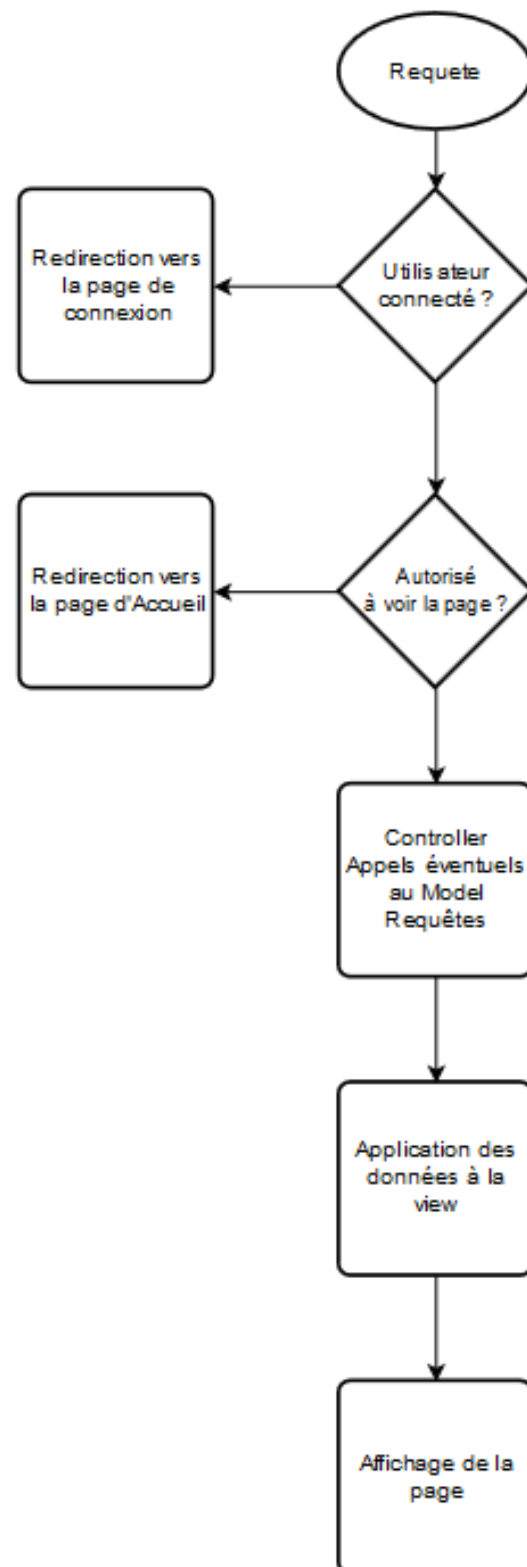
- BateauController => Génération, Traitement ou suppressions des données autour des bateaux
- EntretienController => Génération, Traitement ou suppressions des données autour des entretiens
- EquipementController => Génération, Traitement ou suppressions des données autour des Equipements
- FormController => Traitement des données issues de formulaires
- FournisseurController => Génération, Traitement ou suppressions des données autour des Fournisseurs
- ListesControllers => Traitement des données pour générer les views de listing.
- MessageController => Génération, Traitement ou suppressions des données autour des Messages
- MoteurController => Génération, Traitement ou suppressions des données autour des Moteurs
- PieceController => Génération, Traitement ou suppressions des données autour des Pieces
- UtilisateurController => Génération, Traitement ou suppressions des données autour des Utilisateurs

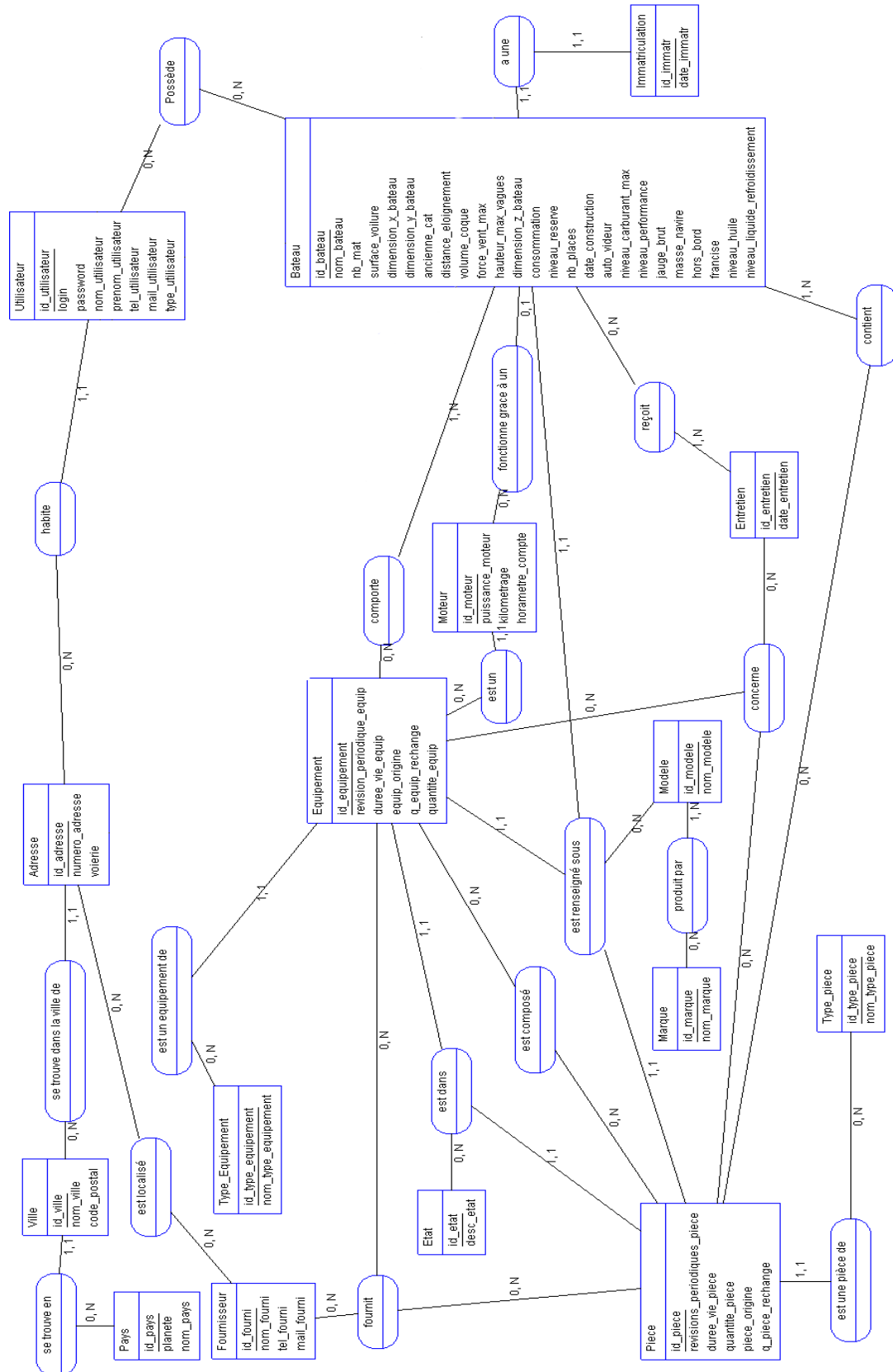
Modélisation et interactions

Partie du site accessible sans connexion



Accès des différents utilisateurs

Gestion de la génération des pages

MCD corrigé

Base de Données

La base de données utilisée pour ce projet est issue d'un projet parallèle en Info0304, traitant du même sujet.

Tous les détails du projet peuvent être trouvés sur le github suivant :

https://github.com/Angom8/ohmonbateau/tree/master/rendu_final

Celle-ci ont cependant été adaptés :

- Les Accidents ne seront pas gérés, puisqu'intégrés implicitement par les Etats et les Entretiens.
- Les Permis ne seront pas gérés, puisqu'il sera demandé à l'enregistrement d'un bateau une copie du permis de l'adhérent.
- La Relation utilise_couramment faisait doublon avec Possède
- Par des soucis de compatibilité, id_utilisateur a été renommé id.
- La notion de Voyage a été supprimée, ainsi que l'emplacement exact (Port) d'un bateau.

Eléments graphiques

L'interface du site utilise les éléments Bootstrap de base, le projet étant surtout Technique. Il est organisé de sorte à être pratique à être utilisé (couleurs représentant les actions effectuées, uniformité de l'apparence), mais l'objectif n'est pas de créer un design particulier. Deux images principales ont été utilisées en tant qu'icône d'onglet (favicon) et logo.



La date rendu du projet étant en hiver, le bateau de l'illustration est gelé.

©Illustration AnkamaGames

Exemples de scripts

Modèle : App\Bateau.php

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;
class Bateau extends Model {
    protected $table = 'Bateau';

    protected $fillable = ['id_bateau', 'nom_bateau', 'nb_mat', 'surface_voilure',
'dimension_x_bateau', 'dimension_y_bateau', 'ancienne_cat',
'distance_eloignement', 'volume_coque', 'force_vent_max', 'hauteur_max_vagues',
'dimension_z_bateau', 'consommation', 'niveau_reserve', 'nb_places', 'auto_videur',
'niveau_carburant_max', 'niveau_performance', 'url_photo', 'jauge_brut',
'masse_navire', 'hors_bord', 'franchise', 'niveau_huile',
'jauge_liquide_refroidissement', 'id_immatr', 'id_moteur', ];

    public function immatriculation(){
        return $this->hasOne(Immatriculation::class);
    }

    public function utilisateurs() {
        return $this->belongsToMany('App\Utilisateur', 'possède', 'id_bateau',
'id_utilisateur');
    }

    public function moteur() {
        return $this->hasOne(Moteur::class);
    }

    public function modele(){
        return $this->belongsTo('App/Modele', 'est_renseigné_sous',
'id_bateau', 'id_modele');
    }

    public function utilisateurscourant() {
        return $this->belongsToMany('App\Utilisateur', 'utilise_couramment',
'id_bateau', 'id_utilisateur');
    }

    public function equipements() {
```

```

        return $this->belongsToMany('App\Equipement', 'comporte',
'id_bateau', 'id_equipement');
    }

    public function pieces() {
        return $this->belongsToMany('App\Equipement', 'contient', 'id_bateau',
'id_piece');
    }

    public function entretients() {
        return $this->belongsToMany('App\Entretien', 'reçoit', 'id_bateau',
'id_entretien');
    }
}

```

Routes : web.php – Pages de bateau pour un utilisateur

```

Route::get('panel/boats', ['uses' => 'ListesController@getBateaux',
'as'=>'boats'])->middleware('auth');//done

Route::get('panel/boats/{id}', ['uses' => 'ListesController@getBateaux',
'as'=>'boatspagex'])->middleware('auth');//done

Route::get('panel/boat/{id}', ['uses' => 'BateauController@show',
'as'=>'boat.show'])->middleware('auth');//done

Route::get('panel/update-boat/{id}', ['uses' => 'BateauController@update',
'as'=>'boat.update'])->middleware('auth');//done

Route::post('panel/update-boat/send', ['uses'
=> 'FormController@updateboat', 'as'=>'boat.update.send'])->
middleware('auth');//done

Route::get('panel/send-boat', ['uses' => 'BateauController@send',
'as'=>'boat.send'])->middleware('auth');//done

Route::post('panel/send-boat/send', ['uses' => 'FormController@sendboat',
'as'=>'boat.send.send'])->middleware('auth');//done

```


Controller : BateauController

```
/**
 * Get all the users to add a first owner, then go to the form
 *
 * @return Create a boat form
 */
public function add()
{
    $users = Utilisateur::get()->pluck('id');
    return view('add-boat', ['users' => $users]);
}

/**
 * Return the send a boat form for Users
 */
public function send()
{
    return view('send-boat');
}
```

Lancement du projet

Date de rendu

Le projet devra être rendu avec ce présent rapport avant le 14 décembre 2019, 23h59. Le premier commit a été le 5 Novembre. L'essentiel du développement s'est déroulé du 25 Novembre au 14 Décembre 2019.

Déroulement

Avant de débiter le projet, nous avons établi une liste ordonnée des étapes de développement :

- Conception de la base de données (Info0304)
- Réalisation du squelette du site : Routes, Éléments PHP, HTML, JS et CSS, ne nécessitant pas d'accès à la base de données
- Création de la base de données : Passage de SQL à la génération Laravel, Adaptation du MCD.
- Peuplement de la base de données : Création de Seeders pour les différentes tables.
- Intégration du lien entre la base de données et le site : Création de requêtes, des Models et des pages de Liste.
- Ajout du système de connexion
- Ajout des actions de mise à jour de la base de données par les utilisateurs (Secrétaire, Admin)
- Restriction des accès aux sections du site selon l'utilisateur

Nous avons globalement suivi l'ordre proposé, mais en traitant parfois plusieurs étapes à la fois (Peuplement de la base de données et intégration du lien et des requêtes). A l'aide d'un fichier todo.txt que nous mettions à jour régulièrement, nous suivons pas à pas notre progression.

Suivi du projet

Git

Le projet a été dès sa création ajouté à la fois sur Github et Gitlab. Ceci nous a permis d'avoir une sauvegarde en ligne lors des problèmes de connexion du second, tout en permettant un bon suivi de l'avancement du projet.

Vers la fin cependant, seul Antoine T. faisait des commits afin d'éviter de perdre du temps à merge les branches (Sébastien B. contribuant tout de même).

Difficultés

Nous avons rencontré plusieurs difficultés lors du développement :

- **Mise en place de Laravel**

Sur Windows, l'installation du framework est un vrai parcours du combattant. Nous nous sommes également rendu compte que nos versions de Laravel étaient différentes (5/6) et avons été obligés de le réinstaller, causant des pertes de temps assez inutiles. Il n'y avait aucun problème sur Linux et nous utilisons une base de données distantes pour pouvoir se « baser » sur les mêmes données.

- **Mise en place du Git**

La plateforme gitlab a été indisponible pendant un certain temps, conduisant à un seul git utilisable. Or, avec des versions de laravel différentes, celui-ci est vite devenu inutilisable et nous avons du réinstaller totalement le framework des deux côtés, en plus de créer un nouveau Git.

- **Un nouvel outil**

Malgré les TP, Laravel restait un outil relativement nouveau pour nous et même après ce projet, nous ne sommes pas sûr de tout avoir compris, ni d'avoir utilisé correctement ses fonctionnalités. Ne pas avoir la main sur tout contrairement aux développements habituels (Info 0102, Info 0202, PPRO 0303) est également une chose à laquelle s'habituer.

Installation

Le site pourra être déployé sur un serveur web dont la version PHP est supérieure à 7.2 (Laravel 6) et possédant localement ou non une base de données MySQL. Un accès FTP sera nécessaire pour insérer les différents fichiers.

Migration : `php artisan migrate :fresh`

Seeders (pour d'éventuels tests) : `php artisan db:seed --class=DatabaseSeeder`

Il sera aussi important de modifier les permissions d'accès, réglés à 777 pour le rendu.

Conclusions

Le projet nous a permis d'apprendre à utiliser un framework et d'appliquer le modèle MVC. Il est fonctionnel, mais nécessiterait d'un travail de design. Certaines requêtes (et leur quantité) pourraient aussi être optimisées. Les objectifs établis dans le cahier des charges ont été appliqués.

Le sujet des Bateaux était particulièrement complexe et malgré notre approche reposant sur une gestion administrative, notre base de données et le site se retrouvent assez lourds et cela devient très vite rébarbatif, en sachant que les instructions données dans le sujet d'INFO0304, qui nous avait été annoncé comme très lié au sujet d'INFO303, étaient en fait bien différentes, surtout au niveau des utilisateurs.

Nous avons aussi géré les équipements et entretiens de façon très générique pour ne pas alourdir la base de données, mais il serait très bien possible d'implémenter des types d'équipement plus strict (De Sécurité, de Navigation) ainsi que des types d'entretien (Mécanique, de Sécurité, Vidange).

Bibliographie

- Documentation Laravel <https://laravel.com/docs/6.x>
- Laravel Sillo <https://laravel.sillo.org/>
- Projet INFO0304 <https://github.com/Angom8/ohmonbateau>
- TP d'INFO0303 (Moodle)