

18 décembre 2019

Projet Web

BonColoc

Melvin AUBOURG, Jordan BOUTEILLÉ et Any LAFAYE
ENSEIRB-MATMECA
Réseaux et Systèmes d'Information

Table des matières

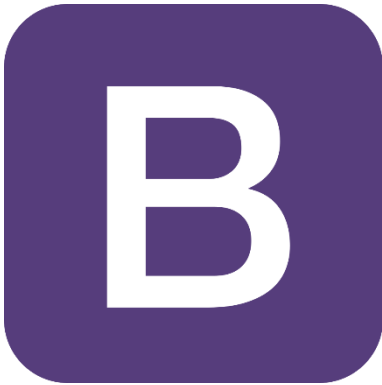
I. Développement Front-End : Bootstrap	4
II. Modèle Conceptuel des Données (MCD)	6
III. Développement Back-End : Laravel	7
A. Authentification	8
B. Eloquent	9
C. Contrôleurs.....	9
V. Fonctionnement de BonColoc.....	11

Introduction


Ce projet Web s'inscrit dans le cadre de notre formation à l'école d'ingénieurs ENSEIRB-MATMECA, en filière Réseaux et Systèmes d'Information.

BonColoc est un site Web dédié à la recherche de colocation ou de colocataire. Un utilisateur peut, soit publier une annonce de colocation, soit en rechercher une selon différents critères de recherche. Par exemple, le type de bien, le nombre de pièce(s), la localisation, etc. Afin de pouvoir bénéficier des services de BonColoc, il est nécessaire de créer un compte.

I. Développement Front-End : Bootstrap



Bootstrap est une boîte à outils open source et gratuite pour le développement de sites et d'applications Web avec HTML, CSS et JS. C'est un ensemble qui contient des formulaires, boutons, outils de navigation et autres éléments interactifs. Grâce à son système de grille réactif, il est possible de créer des présentations de formes et de tailles variées grâce à un système à douze colonnes, à cinq niveaux réactifs par défaut, ainsi qu'à des dizaines de classes prédéfinies. Bootstrap est en effet, « responsive », autrement dit, il s'adapte à la taille de l'écran. Les cinq tailles de grille possible sont les suivantes :

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl- 
# of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

Bootstrap fournit une feuille de style CSS qui contient des définitions de base de tous les composants HTML, ce qui permet de disposer d'une apparence uniforme pour les textes, tableaux et les éléments de formulaires. Le framework fournit également de nombreux éléments graphiques au format standardisé : boutons, libellés, icônes, miniatures, barres de progression, etc.

Prenons l'exemple de la partie « Centres d'intérêts » dans la page d'inscription. Les colonnes sont délimitées en rouge et les lignes, en bleu. Nous observons sur la figure ci-dessous que la partie titre est composée d'une ligne, elle-même décomposée en un titre. Nous avons divisé la ligne en trois sous colonnes afin d'y incorporer les commutateurs personnalisés. Nous avons donc décomposé nos pages en une suite logique de lignes et colonnes.

Figure 1 - Col et rows

Le code correspondant à la figure ci-dessus est le suivant :

```
<!-- ligne horizontale -->
<div class="row">
  <hr width="100%">
  <h4>Centres d'intérêts</h4>
</div>
<!-- commutateur sport -->
<div class="row mb-3">
  <div class="col-sm-4">
    <div class="custom-control custom-switch">
      <input type="checkbox" class="custom-control-input" id="switchSport">
      <label class="custom-control-label" for="switchSport">Sport</label>
    </div>
  </div>
</div>
```

Figure 2 - Cols et rows (inscription.html)

La figure 2 nous permet d'introduire la notion d'espacement entre div (col ou row). Nous pouvons voir que la div row possède plusieurs classes, notamment « mb-3 ». Dans notre cas, nous avons placé un margin sur le bottom de la ligne avec une valeur de 3 afin d'espacer le contenu. Il est possible en effet, de placer le contenu par rapport aux autres éléments grâce aux margins et paddings. Voici leur utilisation :

{property}{sides}-{size}

Où property vaut :

- m - pour les classes qui établissent le margin
- p - pour les classes qui établissent le padding

Où sides vaut :

- t - pour les classes qui établissent le margin-top ou padding-top
- b - pour les classes qui établissent le margin-bottom ou padding-bottom
- l - pour les classes qui établissent le margin-left ou padding-left
- r - pour les classes qui établissent le margin-right ou padding-right
- x - pour les classes qui établissent les deux *-left et *-right
- y - pour les classes qui établissent les deux *-top et *-bottom

Où size vaut :

- 0 - pour les classes qui éliminent le margin ou padding en le fixant à 0
- 1 à 5 - pour les classes qui établissent le margin ou padding à \$spacer * .25 à \$spacer * 3
- auto - ajustement automatique de margin

Comme dit précédemment Bootstrap embarque une multitude d'éléments personnalisés et personnalisables. Sur la figure 1, les commutateurs sont en réalité des checkboxes avec une apparence extérieur modifiée. De même pour les outils de sélection tels que les select (liste déroulante) qui donnent un aspect plus moderne au site. Pour une question de facilité, nous avons utilisé une librairie d'icônes gratuites FontAwesome que nous avons utilisé notamment dans le pied de page pour diriger les utilisateurs aux réseaux sociaux de BonColoc.

II. Modèle Conceptuel des Données (MCD)



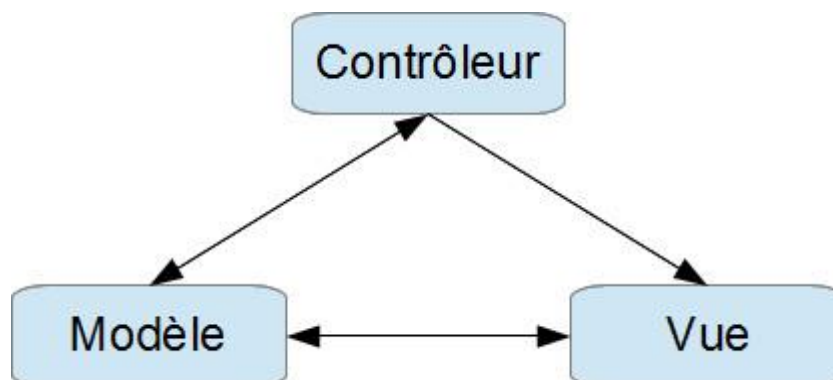
Une table “utilisateur” sert à stocker toutes les personnes inscrites sur notre site Web Boncoloc. Celle-ci comprend un champ “typeProfil” qui sert à identifier si une personne veut poster une annonce ou si elle en recherche une. De plus, nous pouvons connaître les centres d’intérêt d’un utilisateur grâce à un champ de type boolean. Par exemple, si l’utilisateur aime le sport, la valeur est “true”, sinon elle est “false”.

La table “location” permet quant à elle de lier les logements avec les utilisateurs. Elle possède ainsi deux clés étrangères. Un utilisateur peut posséder ou non une location. Effectivement, si un utilisateur recherche une colocation, il n’a pas besoin de publier une annonce.

Enfin, la table “logement” regroupe les informations clés de la colocation (la localisation, la description de l’annonce, la surface, etc).

III. Développement Back-End : Laravel

Laravel est un framework Web open-source écrit en PHP respectant le principe modèle-vue-contrôleur (MVC) et est entièrement développé en programmation orientée objet. Laravel est distribué sous licence MIT (“Massachusetts Institute of Technology”), et ses sources sont hébergées sur GitHub.



Le modèle représente les données (les utilisateurs et les logements).

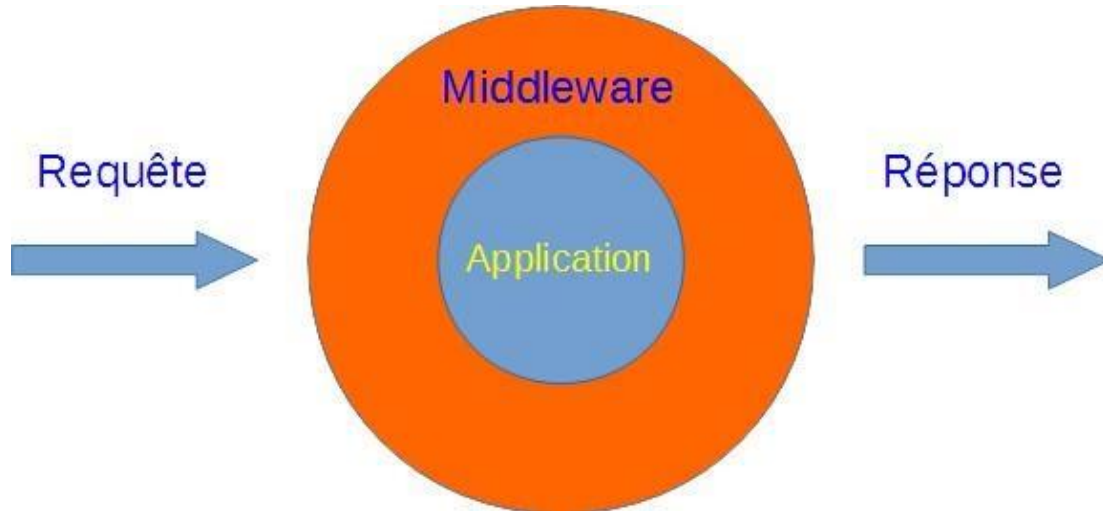
Le contrôleur sert à faire une gestion entre le modèle et la vue.

Les vues ont été créées à partir de l’outil Bootstrap. Ainsi, elles sont appelées lorsque le contrôleur les sollicite afin de les afficher sur le site Web.

Laravel propose plusieurs fonctions simples comme celles citées dans la suite de ce document avec des exemples extraits de notre code.

A. Authentification

Pour authentifier un utilisateur, un **middleware** est utilisé. Celui-ci effectue un traitement à l'arrivée d'une requête ou à son départ :



Pour l'authentification, le middleware va tout d'abord vérifier si le login et le mot de passe saisis sont corrects grâce à une fonction « `auth()` » de Laravel :

```
//Vérifier que l'utilisateur et le mdp est correct
$resultat = auth()->attempt([
    'login' => $request->input('login'),
    'password' => $request->input('mdp')
]);
```

Si le login et/ou le mot de passe sont/est incorrect, alors l'utilisateur est redirigé vers la page de connexion où seront affichées les erreurs (cf. figure ci-dessous).

```
//Si l'utilisateur et/ou mdp incorrect
if(! $resultat)
{
    //Renvoyer vers la page d'accueil avec des erreurs
    return redirect("/boncoloc")->withInput()->withErrors([
        'mdp' => 'Login et/ou mot de passe incorrect.',
        'login' => "Login et/ou mot de passe incorrecte."
    ]);
}
```

C'est la vue qui gère l'affichage des erreurs.

B. Eloquent

Avec Laravel, plus besoin d'utiliser des syntaxes SQL pour les requêtes vers sa base de données grâce à Eloquent. Prenons un exemple avec l'ajout d'un utilisateur :

```
//Préférences
$utilisateur->Sport = $this->IfEmpty($requete->input('sport'));
$utilisateur->Arts = $this->IfEmpty($requete->input('arts'));
$utilisateur->JeuxVideo = $this->IfEmpty($requete->input('j-v'));
$utilisateur->Fete = $this->IfEmpty($requete->input('fete'));
$utilisateur->Lecture = $this->IfEmpty($requete->input('lecture'));
$utilisateur->Musique = $this->IfEmpty($requete->input('musique'));

//Utilisateur
$utilisateur->Prénom = $requete->input('name');
$utilisateur->Nom = $requete->input('lastname');
$utilisateur->DateNaissance = $requete->input('date');
$mail = $requete->input('mail');
$utilisateur->Mail = $mail;
$utilisateur->Tel = $requete->input('phone');

//Identifiant
$log = $requete->input('login');
$utilisateur->Login = $log;
$utilisateur->Mdp = bcrypt($requete->input('mdp'));

//profil
$utilisateur->TypeProfil = $requete->input('profil');
```

Nous récupérons tout d'abord les centres d'intérêt de l'utilisateur. Pour cela, on utilise une fonction « IfEmpty » qui renvoie « true » ou « false » en fonction des données saisies. Puis, nous récupérons également ses coordonnées, et enfin ses identifiants de connexion pour qu'ils soient stockés dans la table utilisateur, (créée par un modèle présent dans le dossier laravel5/app/Models). Ensuite, nous enregistrons la ligne dans la base de données avec la fonction « save() » de Eloquent et nous dirigeons l'utilisateur vers la page de connexion en affichant un message (cf. figure ci-dessous).

```
$utilisateur->save();
session()->flash('inscription', 'Inscription avec succès !');

sleep(3);

return redirect("/boncoloc");
```

C. Contrôleurs

Tout d'abord, dans notre site, un contrôleur est appelé via un routage (fichier laravel5/routes/web.php) :

```
Route::get('/boncoloc/annonce/{id}', 'annonce@getPage')->middleware('App\Http\Middleware\TypeProfilRecherche');
```

Ici, lorsque l'url comporte « /boncoloc/annonce/{id} », Laravel appelle la fonction « getPage » du contrôleur « Annonce » en passant l'id du logement qui est dans l'url. Au préalable, le contrôleur utilise le middleware pour vérifier que l'utilisateur actuel a un profil « Chercheur » et qu'il est authentifié (c'est une sécurité pour éviter que n'importe qui accède au site sans se connecter).

Le contrôleur permet de gérer les données ainsi que la vue. Prenons en exemple le code ci-dessous (nouvel exemple de l'utilisation de Eloquent) :

```
//trouver le logement
$annonce = Logement::find($id);

//jointure avec la location
$location = Logement::find($annonce->IdLogement)->locations;

//trouver l'utilisateur qui a publié l'annonce
$user = Location::find($location->IdLocation)->utilisateur;

//vue de l'annonce avec les données du logement et de l'utilisateur
return view('annonce', [compact('annonce'), compact('user')]);
```

Nous récupérons le logement grâce à son id dans la base de données. Ensuite, grâce à une jointure avec la table « Location », nous trouvons l'utilisateur qui a publié le logement. Enfin, nous retournons une vue nommée « annonce » qui contient les données du logement (« compact('annonce') ») et celles de l'utilisateur (« compact('user') »).

V. Fonctionnement de BonColoc

Tout d’abord, une page d’accueil s’affiche indiquant deux possibilités : se connecter et s’inscrire.

Dans le cas de l’inscription, une vue « inscription » apparaît et n’importe qui peut créer un compte en renseignant les informations utiles. L’utilisateur a le choix entre deux profils : « Chercheur » pour rechercher des colocations ou « Annonceur » pour publier une annonce de colocation.

En ce qui concerne la connexion, il y a deux chemins possibles que nous allons décrire par la suite.

- Le chercheur

Si le login correspond à un profil “Chercheur”, l’utilisateur est dirigé vers la page dédiée à la recherche d’une colocation. Il peut ainsi orienter sa recherche en fonction de ses centres d’intérêt en spécifiant préalablement ses critères (surface, prix, etc).

En soumettant ses préférences, l’utilisateur est redirigé vers une page où plusieurs annonces sont affichées. Il peut alors choisir celle qui l’intéresse pour que toutes les données relatives au logement soient affichées dans une nouvelle vue.

- L’annonceur

En revanche, si le login correspond à un profil “Annonceur”, il y a 2 possibilités :

- L’utilisateur a déjà publié une annonce : il est dirigé vers la page « monAnnonce » qui permet de voir l’annonce publiée. Il peut la modifier ou la supprimer.
- L’utilisateur n’a jamais publié ou a supprimé une annonce : il est renvoyé vers une page pour ajouter une annonce.

Pour finir, un utilisateur peut se déconnecter en cliquant sur « Se déconnecter » en haut droite, et il sera automatiquement dirigé vers la page de connexion.

Conclusion

Grâce au développement de ce site web, nous avons pu apprendre à utiliser les framework Bootstrap et Laravel, à gérer une base de données et donc un serveur, les fonctions javascript, etc.

Malgré un bon travail sur notre site Web, nous souhaitons partager des axes d'amélioration que nous aurions pu ajouter sur notre page web :

- Mot de passe oublié : fonctionnalité qui permet, lorsqu'un utilisateur oublie son mot de passe, d'envoyer un mail de réinitialisation de ce dernier.
- La gestion de son compte : l'utilisateur peut modifier son adresse mail, son mot de passe, son numéro de téléphone et peut notamment supprimer son compte.