Université de Rouen

Projet Languages WEB
2018

Réalisé par Rani TOTONJI

Table des matières

Laravel

I. Présentation

Le langage de programmation utilisé dans ce projet (PHP) a été fixé au préalable par **M. Olivier Mallet**. En revanche le choix du *framework* web a été laissé libre. Le *framework* choisi est **Laravel 5.6** sous **Nginx** comme serveur web.

Laravel est un framework web open-source écrit en PHP2 respectant le principe MVC (modèle-vue-contrôleur), et entièrement développé en programmation orientée objet. Laravel est distribué sous licence MIT, avec ses sources hébergées sur GitHub. Ce framework a été créé en juin 2011 et il est rapidement devenu l'un des frameworks PHP les plus référencés, utilisés et documentés.

Laravel est un framework d'application web avec une syntaxe expressive et élégante. Le créateur de ce framework croit que le développement doit être une expérience agréable et créative pour être vraiment épanouissant. Laravel tente donc à éviter le coté désagréable de la programmation en facilitant les tâches courantes utilisées dans la majorité des projets Web, tels que l'authentification, le routage, les sessions et la mise en cache. Il vise à rendre le processus de développement agréable pour le développeur sans sacrifier la fonctionnalité de l'application. Les développeurs heureux font le meilleur code. À cette fin, il a tenté de combiner le meilleur de ce qu'il a vu dans d'autres frameworks web, y compris des frameworks implémentés dans d'autres langages, tels que Ruby on Rails, ASP.NET MVC et Sinatra.

Laravel, dont l'installation est basée sur le gestionnaire de paquets Composer, est Open source, accessible, gratuit, puissant, fournissant tous les outils nécessaires au développement des grandes applications robustes en termes de routage de requête, de mapping objet-relationnel (un système baptisé Eloquent implémentant Active Record), d'authentification, de vue (avec **Blade**), de migration de base de données, de gestion des exceptions et de test unitaire.

II. Architecture et arboresence des répertoires

L'architecture du framework est très simple et intuitive. Respectant le principe MVC, ce framework sépare les fichiers de code selon leur fonction. L'architecture peut être shématisée de la façon suivante :

- Les fichiers contenant les modèles (un fichier .php par modèle) sont placés dans le répertoire «app/».
- Toujours dans ce répertoire, les fichiers contenant les contrôleurs (un fichier .php par controleur) sont placés dans «app/Http/Controllers/».

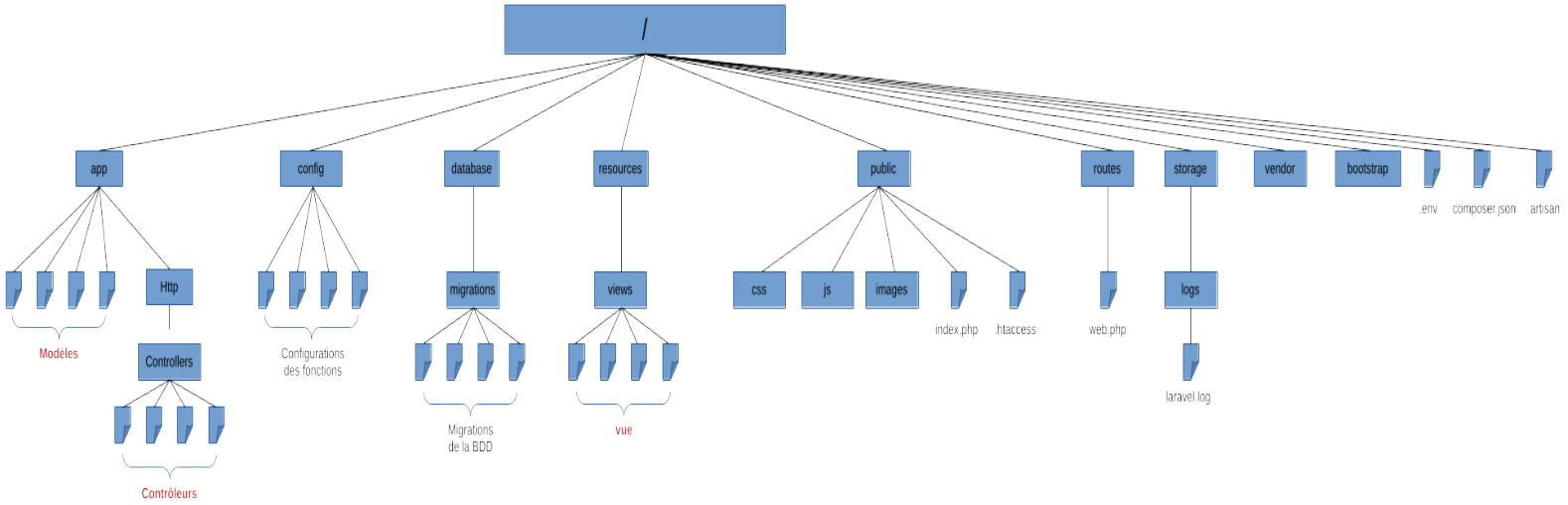
- Les fichiers correspondant à la vue (un fichier .php par page) sont placés dans «resources/views/».
- La liste des paramètres de l'application sont placés dans un fichier caché «.env». Cette liste contient les clés de connexion à la base de données.
- Les configurations des fonctions de l'application sont placées dans «config/» (un fichier .php par fonction).
- La racine de l'arboresence est située dans «public/». Ce répertoire contient le fichier «index.php», les feuilles de style («public/css/»), les scripts JavaScript («public/js»), les images de l'application («public/images»), le fichier caché de configuration du serveur web («public/.htaccess»), ainsi que l'icon de l'application. Seulement le contenu de ce répertoire est accessible aux utilisateurs, ce qui permet d'avoir un système bein sécurisé quand aux fonctionnalités de l'application (modèles et contrôleurs) en les plaçant en dehors de ce répertoire.
- La liste des urls de routage (POST et GET) est placée dans le repertoire «routes/» et précisement dans le fichier «web.php»
- Les logs (erreurs) durant l'exécution de l'application sont incrémentées dans un fichier texte placé dans le répertoire «storage/logs/laravel.log»
- L'ensemble des modules PHP nécessaires au lancement du framework sont placés dans le répertoire «vendor/»

III. Création et migration de la base de données

La connexion avec la base de donnée est assuré grâce à la fonctionnalité **Eloquent ORM**. La configuration de cette connexion est paramétrée dans les fichier «.env» et «config/database.php». Ce module permet également de définir la structuration de la base de données, les contraintes, les clés primaires et secondaires… Chaque entité (table) est définei par un fichier .php situé dans «database/migrations». Une fois cette connexion avec la base de données est établie, la migration de la structure des entités est réalisée grâce à la fonctionnalité **Laravel Migration**.

IV. Fonctionnalités front/back

Laravel possède son propre système de template : **Blade**. Ce dernier fonctionne avec des vues ayant l'extension blade.php, dans lesquelles il affiche le contenu des variables PHP. Il peut générer des formulaires et il peut créer des layouts (modèle de base pour les pages html). Blade contient des instructions conditionnelles et itératives. Les formulaires génèrent des tokens pour empêcher les attaques **CSRF**. Le principe de ces attaques **C**ross-**S**ite **R**equest **F**orgery est de contourner l'authentification du site pour effectuer des actions malveillantes, et ces actions peuvent être envoyés par un formulaire..



Architecture du framework Laravel

Développement du projet

Le code du projet est divisé en deux parties : la partie visuelle (vue) appelée « Frontend » et la partie fonctionnelle (modèles et contrôleurs) appelée « Backend »

I. Backend (PHP)

Comme demandé par M. Mallet, la partie Backend, codée en PHP, sert à établir la connexion avec la base de données, charger la carte du jeu, identifier les utilisateurs et sauvegarder leurs sessions de jeu.

1) Créatation de la base de données

À la première connexion, le contrôleur « HomeController » vérifie l'état d'installation du jeu en vérifiant la valeur de la variable d'envirement « INSTALLED » dans le fichier « .env ». Si cette valeur vaut 0, l'utilisateur sera redirigé vers l'interface d'installation où il choisira le nom de la base de donnée souhaité, le nom de l'utilisateur, le mot de passe, et éventuellement un préfixe. L'application créera ensuite la base de données avec le contrôleur « ConfigController » qui lancera un script MySQL placé dans « scripts » et utilisant les données renseignées par l'utilisateur. Grâce à la fonctionnalité « Migration », les tables seront générés et migrés (schéma ci-dessous).

2) Authentification *et création de comptes utilisateurs*

La gestion des comptes utilisateurs (création et authentification) est totalement gérée par le framework Laravel. En effet, deux contrôleurs pré-définis ont été placés dans le répertoire « app/Http/Controllers/Auth » : « LoginController »et « RegisterController ». Ces deux contrôleurs permettent de valider les données entrées par l'utilisateur : adresse mail,

mot de passe, nom et prénom… et communiquer avec le modèle « User » pour l'authentifier ou lui créer un compte.

3) Chargement des cartes

Pour commencer le jeu, choisis une des deux cartes disponibles. Au chargement de la carte (contrôleur « HomeController »), les paramètres du jeu seront initialisés dans le code JavaScript, grâce à l'abilité de la vue (Blade) à intégrer des variables PHP.

4) Sauvegarde et chargement des sessions de jeu

Une des fonctionnalité de l'application c'est la possibilité de sauvegarder la session de jeu actuelle, et pouvoir la récupérer et la reprendre par la suite. Au cours du jeu, l'utilisateur connecté avec son compte, sera capable d'envoyer au serveur (contrôleur « UserController ») les données du jeu (les variables JavaScript) à l'aide d'une requête AJAX. La particularité de cette méthode est d'établir une connexion étroite et rapide avec le serveur, sans devoir changer de page ou d'URL, ce qui permet d'avoir une meilleur expérience utilisateur.

II. Frontend (Blade + JavaScript)

Le déroulement du jeu est totalement codé en Javascript, sur le navigateur de l'utilisateur. Une fois la carte téléchargée à partir de la base de données, 5 fichiers JS sont appelés :

- **mainLoop.js** permet de :
    1. définir et initialiser un «canvas» dans lequel il sera ajouter les éléments du jeu (enemies, canons, tires…)
    2. Actualiser régulièrement le contenu du «canvas» chaque lapse de temps (FPS) et mettre à jour les statistiques
- **towerUnits.js :** définie une classe «Tower» avec les attributs : coordonées du centre, rayon, portée, puissance, fréquence de tire, couleur, prix, ainsi que les méthodes nécessaires aux fonctionnement du canon : ajout dans le canvas, recherche de cible, processus du tire. Cinq types de canons ont été créés à partir de ce protoype «Tower».
- **attackerUnits.js** : définie une classe «Enemy» avec les attributs : coordonées, points de vie, vitesse, couleur; ainsi que les méthodes nécessaires au déplacement et à la durée de vie : ajout dans le canvas, déplacement sur le trajet, vérification des points de vie, vérification de la perte. Trois types d'enemies ont été créés à partir de ce protoype «Enemy».
- **bullets.js** : définie une classe «Bullet» qui correspond aux missiles tirés par les canons, avec les attributs : coordonées, rayon, puissance, vitesse, cible (de type

Enemy), couleur; ainsi que les méthodes nécessaires au déplacement et à la durée de vie : ajout dans le canvas, déplacement vers la cible en mouvement, vérification de la collision avec la cible. La cible étant en mouvement, les missiles suivent tout le temps leurs cibles en prenant des trajectoires .

- **mouseClick.js** : permet de défnir les actions de la souris dans le « canvas » :
    1. En déplacement de la souris : autorisation ou non de la construction d'un nouveau canon
    2. En cliquant : construction et ajout d'un nouveau canon dans le « canvas »

III. Fonctionnalités de l'application

- Plusieurs types cartes

- Plusieurs types d'enemies (fragile, rapide, blindé)

- Plusieurs types de canons (faible, rapide, moyen, puissant, ralentisseur)

- Possibilité de vendre des conons

- Possibilité de suspendre ou recommencer une session de jeu

- Possibilité de sauvegarder une session, et de la retrouver dans le compte utilisateur

- Possibilité de reprendre une session déjà commencée

- Ajout des statistitiques (nombre d'enemies arrêtés, nombre d'enemies passés, niveau progressif des enemies, cash money, temps de jeu)

IV. Modules Javascript supplémentaires utilisés

- toastr : permet d'afficher des messages de notification dans une page HTML.

- sweetalert2 : permet d'appliquer des feuilles de style et des fonctionnalités aux alerts JavaScript traditionnelles.

V. Autres modules

- fontawesome : permet d'inclure des icons au format texte dans une page HTML

Installation

I. Système d'exploitation et prérequis

Le système d'exploitation requis pour l'installation de l'application est Linux. Le système d'exploitation a été choisi par rapport à la version utilisée du framework Laravel. Les modules indispensables au lancement du framework :

- Mysql server
- PHP >= 7.1.3
- OpenSSL PHP Extension
- PDO PHP Extension
- Mbstring PHP Extension
- Tokenizer PHP Extension
- XML PHP Extension
- Ctype PHP Extension
- JSON PHP Extension

II. Étapes de l'installation

Comme décrit au premier chapitre dans la rubrique "Architecture et arboresence des répertoires », les répertoires et les fichiers placés en dehors du répertoire « public » sont soumis à des droits d'accès privés. Pour pouvoir modifier les paramètres de l'application (nom de l'utilisateur de la BDD, son mot de passe, le nom de la BDD…), un changement des droits d'accès au fichier « .env » sera indispensable. Ainsi, les répertoires « vendor », « storage », « bootstrap » devront également être accessible par l'application. La première étape de l'installation consiste donc à lancer le script « install.sh » placé dans à la racine du projet. Ce script permet d'autoriser l'accès aux fichiers précédents. La deuxième étape sera la création de la base de données et la migrations des tables. Cette étape sera effectuée automatiquement dès la première connexion.

1. Modification des droits d'accès aux répertoires

   ```
   sudo ./install.sh
   ```

2. Première connexion : lancer l'application sur le localhost.
3. Remplir les données de connexion à la base de données
4. Installer et initialiser la base de données

Améliorations possibles :

- Ajouter un enemie Soldat, qui ne sera pas affecté par les missiles des canons
- Ajouter un canon Métrilleuse, qui sera effectif contre les soldats uniquement
- Possibilité de mise à niveau des canons : puissance, portée et fréquence de tire
- Ajout d’un score personnel à battre à chaque session de jeu (nombre de chars neutralisés)
- Ajout d’un défis entre les utilisateur pour avoir le meilleur score

Bibliographie

- Portail du framework Laravel, https://laravel.com/ *(consulté le 07/04/2018)*
- Documentation de la version Laravel 5.6 , https://laravel.com/docs/5.6 *(consulté le 07/04/2018)*
- Tower Defense – Wikipédia, https://en.wikipedia.org/wiki/Tower_defense *(consulté le 07/04/2018)*