IUT ANNECY UNIVERSITÉ SAVOIE MONT BLANC

INFO INFORMATIQUE

BMW MOTORRAD

Procédure D'Installation

Le 14/01/2024

DULHOSTE Maxime | LAMY Evan | MIHOUBI Marouane | NUNES Ricardo| RAGUIN Hugo

Groupe 1 Sujet 5

Table des matières

Introduction :

Nous sommes ravis de vous présenter ce guide détaillé pour installer configurer Laravel et une base de donnée. Ce document vise à vous fournir toutes les informations nécessaires pour une installation réussie. Pour mener à bien l'installation, veuillez vérifier que PHP est installé sur votre système. Laravel nécessite PHP 7.3 ou supérieur. Ensuite, veuillez installer Composer, le gestionnaire de dépendances pour PHP. Pour vous faciliter la tâche **installer l'extension VsCode SSH** afin de pouvoir réaliser ce guide.

Partie 1 : procédure d'installation l'environnement de développement et sa configuration :

1.1 Installation de Laravel :

- Ouvrez votre terminal ou console de commande.
- Exécutez la commande suivante :

```
nunric@ns3146606:~/public_html/Screen pour marouane/[NomDuFichier]$ composer global require laravel/installer
```

1.2 Création d'un nouveau projet Laravel :

- Dans votre terminal, naviguez jusqu'au répertoire souhaité pour votre projet.
- Tapez la commande suivante :

```
$ composer create-project laravel/laravel [NomDuFichier]
```

Remplacez « NomDuFichier » par le nom souhaité pour votre projet.

1.3 Lancement du serveur de déploiement :

- Accédez au répertoire de votre projet que vous venez de créer :

```
nunric@ns3146606:~/public_html/Screen pour marouane$ cd [NomDuFichier]
```

- Démarrez le serveur de développement intégré (remplacer XXXX par le port voulu)

-
```
nunric@ns3146606:~/public_html/Screen pour marouane/[NomDuFichier]$ php -S 0.0.0.0:XXXX -t public
```

- Si vous avez le message "Address already in use", changez le port.
- Pour vérifier, chargez **http://localhost:8080** dans votre navigateur. Remplacer "localhost" par le nom du serveur si vous n'êtes pas en local.
- Votre application Laravel sera accessible à cette adresse.
- Vous pouvez maintenant commencer à explorer Laravel et développer votre application. Pour plus d'informations, consultez la documentation officielle de Laravel. (Laravel Documentation - Laravel 10.x - The PHP Framework For Web Artisans)

1.4 Mise au point sur les dossiers :

- **config** : les fichiers de configs
- **public** : point d'accès + toutes les ressources statiques (css, js, images...)
- **app** : le coeur du système, avec les fichiers du modèle et les contrôleurs
- **app/http/Controllers** : vos contrôleurs
- **app/Models** : vos modèles
- **resources/views** : les vues (*.blade.php)
- **resources/views/layouts** : les modèles de page

1.5 Connexion de Laravel à PostgreSQL :

- Dans le fichier ".env", supprimez toutes les lignes qui commencent par DB_ :

```
//LIGNE à Supprimer
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laravel
DB_USERNAME=root
DB_PASSWORD=
```

  - Dans le fichier "config/database.php" :
  - On change la ligne 18 qui est le type de connexion par default(sqlite, mysql ,pgsql, sqlsrv).

```
18      'default' => env('DB_CONNECTION', '[Types de base de données]'),
```

- Dans la connexion, plus bas, remplir les champs (Ne pas oublier le schéma) :

```
'pgsql' => [
    'driver' => 'pgsql',
    'url' => env('DATABASE_URL'),
    'host' => env('DB_HOST', '[ADRESSE IP DE LA BASE]'),
    'port' => env('DB_PORT', '5432'),
    'database' => env('DB_DATABASE', '[nom de la base ]'),
    'username' => env('DB_USERNAME', '[nom utilisateur de la base de donnees]'),
    'password' => env('DB_PASSWORD', '[Mot de passe de la base de donnees]'),
    'charset' => 'utf8',
    'prefix' => '',
    'prefix_indexes' => true,
    'search_path' => '[Nom du schéma]',
    'sslmode' => 'prefer',
],
```

1.6 Modèle :

- Sur le site, les modèles représentent les différentes tables de la base de données. Il s'agit d'un fichier qui fait le lien entre les contrôleurs et la base de données, permettant d'accéder aux différentes tables et de gérer les liaisons :

```
nunric@ns3146606:~/public_html/Screen pour marouane/[NomDuFichier]$ php artisan make:model [NomDuModel]
```

- Ajoutez le nom de la table pour une connexion automatique, avec le nom de la clé primaire et l'annulation des timestamps :

[NomDuFichier] > app > Models > Model.php

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class NomDuModel extends Model
{
    protected $table = "[NomDeTable]";
    protected $primaryKey = "ClefPrimaire";
    public $timestamps = false;

}
```

1.7 Routes :

- Le fichier Route permet d'établir un chemin avec les liens pour se rendre dans les modèles
- Il va par la suite recherche une correspondance avec les fins de lien remplie dans le fichier WEB.php pour lancer la méthode qui se trouve dans le contrôleur.

```php
Route::get("/Modeles",[MotoController :: class , "index" ]);
Route::get('/Modeles/{id}', [MotoController::class, 'one']);
Route::get('/Modeles/{id}/Equipement/{id2}', [MotoController::class, 'two']);
```

1.8 Contrôleurs :

- Les contrôleurs sont responsables de lancer plusieurs modèles, qui établissent des liens avec la base de données afin de récupérer les informations nécessaires. Ces informations sont ensuite envoyées aux différentes vues, chacune représentant une page web :

```php
public function index()
{
    $motos = Moto::all();
    $illustrers = Illustrer::all();
    $apourvaleurs = APourValeur::all();
    $gammemotos = GammeMoto::all();
    $peutequipers = PeutEquiper::all();
    $peutcontenirs = PeutContenir::all();
    $posseders = Posseder::all();


    return view ("moto-list", compact('motos',
    'illustrers','peutequipers','apourvaleurs', 'gammemotos','peutcontenirs','posseders'));
}
```

1.9 Vues :

- Une 'vue' est un élément essentiel du modèle architectural MVC (Modèle-Vue-Contrôleur). Elle est responsable de la présentation des données et est typiquement utilisée pour déterminer la manière dont ces données sont affichées à l'utilisateur. Voici, par exemple, un extrait de code HTML de notre vue :

```blade
{{-- Héritage du layout moto --}}
@extends('layouts.moto')

{{-- Inclusion du fichier CSS pour le style du formulaire --}}
<link rel="stylesheet" type="text/css" href="{{ asset('styleCB.css') }}" />

{{-- Définition du titre de la section --}}
@section('title', 'Reseaux')

{{-- Contenu principal de la page --}}
@section('content')
    <h1>Bienvenue sur Ma Petite Page</h1>
    <p>Ceci est un exemple de petite page HTML.</p>
    <ul>
        <li>Elément 1</li>
        <li>Elément 2</li>
        <li>Elément 3</li>
    </ul>

@endsection
```

1.10 Authentification :

- Dans la classe AuthController, insérez ce code pour gérer les connexions des comptes autorisés. La première partie du code définit les comptes clients valides avec leurs informations. Ensuite, dans la seconde partie, si le compte est présent dans la base de données, l'authentification est autorisée ; sinon, un message 'aucun compte trouvé' sera affiché :

```php
public function login(Request $request)
{
    $motDePasseChiffre = hash(self::Cypte, $request->input("password"));

    $user = CompteClient::where('email', $request->input("email"))
        ->where('password', $motDePasseChiffre)
        ->first();

    if ($user) {
        Auth::login($user);
        $request->session()->regenerate();

        return redirect()->intended('/');
    }

    return back()->withErrors([
        'email' => 'Aucun compte trouvé',
    ])->withInput();
}
```

- Fonction à rajouter pour la déconnexion :

```php
public function logout(Request $request)
{
    Auth::logout();
    $request->session()->invalidate();
    $request->session()->regenerateToken();

    return redirect('/');
}
```

- Pour créer des comptes, se rendre dans la classe register :

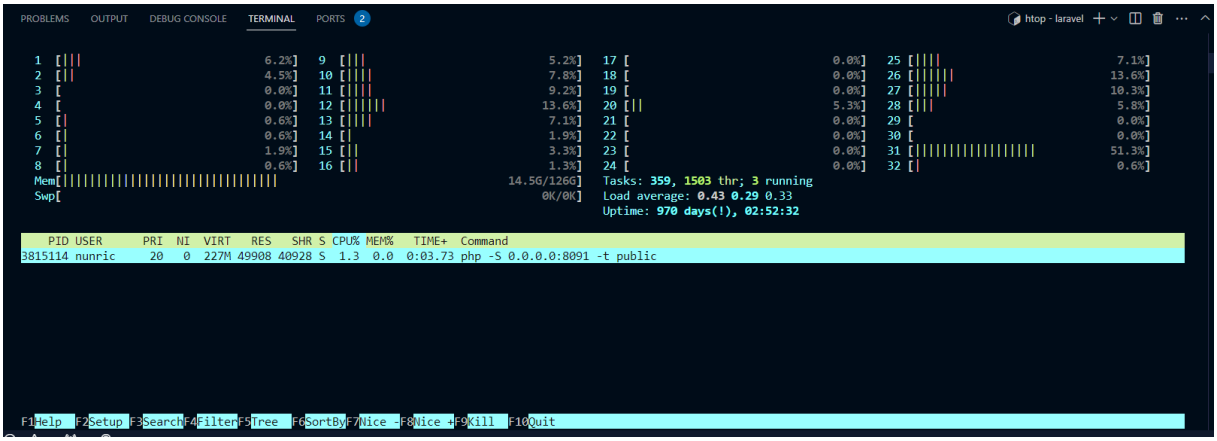
```php
$motDePasseChiffre = hash(self::Cypte, $request->string("new_password"));

// Create a new user account
$user = CompteClient::create([
    'email' => $request->input('new_email'),
    'password' => $motDePasseChiffre, // Utilise Hash::make pour crypter
    'nomclient' => $request->input('nom'),
    'prenomclient' => $request->input('prenom'),
    'civiliteclient' => $request->input('civilite'),
    'numeroclient' => $request->input('numero_client'),
    'datenaissanceclient' => $request->input('datenaissance'),
]);
```

Partie 2 : Réalisation des tests de performances réseaux du site :

2.1 HTOP :

Les performances de base lors du chargement de votre site pour la première fois :

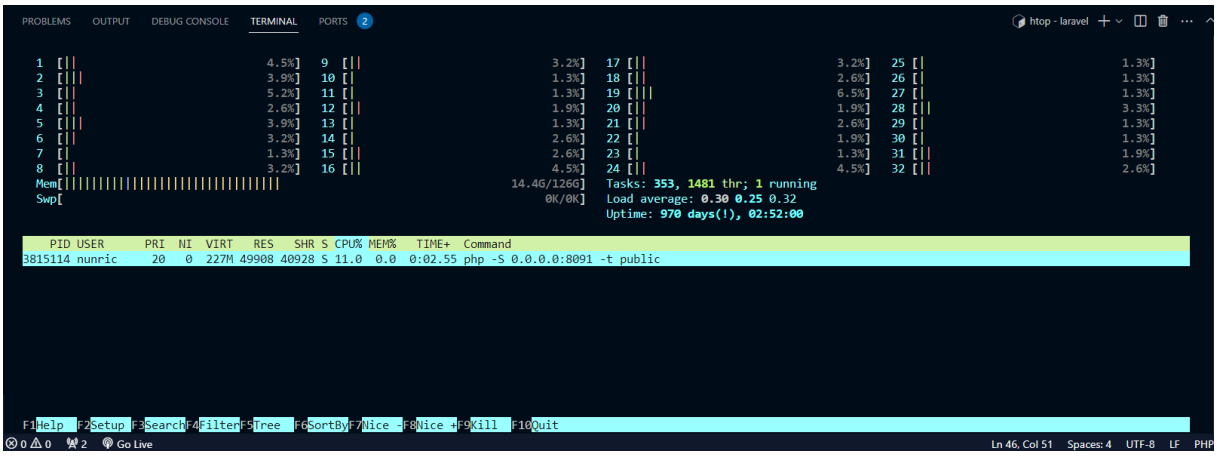


- Charge CPU : Les barres indiquent une utilisation de faible à modérée, avec les cœurs les plus sollicités affichant environ 6,2 % et 7,1 % d'utilisation. Les autres cœurs sont beaucoup moins utilisés ou restent inactifs.
  En bas de l'écran, le processus mis en évidence est un serveur PHP intégré, exécutant la commande 'php -S' pour héberger une application Laravel sur le port 8091. Ce processus consomme 5,2 % de la capacité du CPU et n'utilise ni mémoire virtuelle ni swap, ce qui est une charge raisonnable pour un serveur web gérant trois utilisateurs simultanément.

- Mémoire (RAM) : Il y a une utilisation significative de la mémoire avec la barre de progression presque complètement remplie, indiquant que la majorité des 126 Go de mémoire est utilisée.

- Stockage (Swap) : L'utilisation du swap est minime, ce qui indique que la mémoire vive (RAM) est suffisante pour les besoins actuels du système.

Les performances de base lors du chargement, avec une charge importante de requêtes clients :



- Charge CPU : L'utilisation du CPU semble avoir augmenté par rapport à la première image, avec un cœur particulièrement à 11,0%. Cependant, dans l'ensemble, l'utilisation reste modérée.
- Mémoire (RAM) : L'utilisation de la mémoire est très similaire à la première image, avec une grande partie de la mémoire de 126 Go utilisée.
- Stockage (Swap) : Comme dans la première image, l'utilisation du swap est négligeable.

Pour chaque image, le système a également été en fonctionnement pendant 970 jours, ce qui indique une très longue période sans redémarrage. Cela peut être courant pour les serveurs, mais il est souvent recommandé de redémarrer périodiquement pour appliquer les mises à jour de sécurité et garantir un fonctionnement optimal, cela n'est pas de notre ressort.

Partie 3 : Installer et configurer Botman dans un projet Laravel :

3.1 Installer Botman :

- Dans le répertoire racine de votre projet Laravel, installer les prérequis du chatbot Botman :

```
nunric@ns3146606:~/public_html/SAE/bmwMoto/laravel$ composer require botman/botman
```

- Ensuite, installer le driver web de Botman :

```
nunric@ns3146606:~/public_html/SAE/bmwMoto/laravel$ composer require botman/driver-web
```

- Configurer le fichier cache de Botman en rajoutant un dossier « botman » dans le dossier « config » puis créer un fichier « config.php » dedans. Coller le code suivant dans le fichier « config.php » du dossier « botman » :

```php
<?php
return [
    'conversation_cache_time' => 40,
    'user_cache_time' => 30,
];
?>
```

- Configurer le driver web de Botman en rajoutant un fichier « web.php » dans le dossier « botman » précédent. Coller le code suivant dans ce fichier :

```php
<?php

return [

    'matchingData' => [
        'driver' => 'web',
    ],
];
?>
```

- Créer un nouveau Controller pour botman :

```
nunric@ns3146606:~/public_html/SAE/bmwMoto/laravel$ php artisan make:controller BotManController
```

- Créer une nouvelle route pour le Controller botman : copier le code suivant dans le fichier web.php dans le dossier « routes » de votre projet.

```
use App\Http\Controllers\BotManController;
Route::match(['get', 'post'], '/botman',
'App\Http\Controllers\BotManController@handle');
```

- Rajouter le chatbot Botman à la page d'accueil de votre projet : dans votre fichier php d'accueil, rajoutez le code suivant qui permet de créer un bouton « chatbot ». Vous pouvez modifier le code d'accueil de votre chatbot en fonction des spécificités de votre site (charte de couleurs, image d'arrière-plan, message d'accueil, etc .)

```
<script>
        var botmanWidget = {
            aboutText: '',
            introMessage: "Bienvenue dans notre site web"
        };
    </script>

    <script src='https://cdn.jsdelivr.net/npm/botman-web-
widget@0/build/js/widget.js'></script>
```

- Cette partie du code montre un contrôleur de BotMan. Dans la méthode **handle()**, le bot écoute les messages. Si un message reçu correspond à l'un des mots-clés ("compte", "connexion", "deconnexion", "moto", "configuration", "panier" ou "commande"), il appelle une fonction spécifique pour traiter la question correspondante.

```php
<?php
namespace App\Http\Controllers;

use BotMan\BotMan\BotMan;
use Illuminate\Http\Request;
use BotMan\BotMan\Messages\Incoming\Answer;

class BotManController extends Controller
{

    public function handle()
    {
        $botman = app('botman');

        $botman->hears('{message}', function($botman, $message) {

            if ($message == 'compte' || $message == 'connexion' || $message == 'deconnexion'|| $message == 'COMPTE') {
                $this-> askQuestionCompte($botman);
            }
            else if ($message == 'configuration' || $message == 'moto'|| $message == 'CONFIGURATION') {
                $this-> askQuestionConfiguration($botman);
            }
            else if ($message == 'panier'|| $message == 'PANIER') {
                $this-> askQuestionPanier($botman);
            }
            else if ($message == 'commande'|| $message == 'COMMANDE') {
                $this-> askQuestionCommande($botman);
            }

        });

        $botman->hears('.*salut.*|coucou.*|bonjour.*|Bonjour.*', function (BotMan $bot) {
            $response = "Bonjour à toi, tappez 'aide'";
            $bot->reply($response);
        });
```

- Ce morceau de code définit une écoute pour le mot 'aide'. Lorsqu'un utilisateur envoie 'aide', le bot répond avec des instructions sur comment poser des questions sur différents sujets, en utilisant des mots-clés pour chaque catégorie de question.

```
    $botman->hears('aide', function (BotMan $bot) {
        $response = "Tu peux me poser des questions sur différents Sujets, pour cela il suffit de tapper la commande que vous souhaitez<br>
        -'COMPTE' pour avoir des informations sur la création, connexion, suppression etc d'un compte<br>
        -'CONFIGURATION' pour avoir des informations sur la création, suppression etc d'une configuration de moto<br>
        -'COMMANDE' pour avoir des informations sur la création, suppression etc d'une commande<br>
        -'PANIER' pour avoir des informations sur votre panier,sa suppression, modification etc<br>";
        $bot->reply($response);
    });


    $botman->listen();
}
```

- Cette partie du code s'occupe des questions liées au panier. Elle fournit des instructions sur comment ajouter des articles au panier, les supprimer ou les visualiser.

```
public function askQuestionPanier($botman)
{
    $botman->ask('Si vous avez des problèmes pour ajouter, visualiser ou supprimer dans votre panier, précisez votre problème en répondant avec un des chiffres suivant:<br>
    1. Ajouter au panier.<br>
    2. Supprimer du panier.<br>
    3. Visualiser votre panier.<br>
    4. Modifier la quantité d\'un article dans votre panier.<br>', function(Answer $answer) {
        $question = $answer->getText();

        if (stripos($question, '1') !== false) {
            $responseMessage = 'Pour ajouter au panier un article il suffit de cliquer sur le bouton "Equipements" dans la barre des tâches en haut de votre écrant puis de selecti
        } elseif (stripos($question, '2') !== false) {
            $responseMessage = 'Pour supprimer un article du panier il suffit de ciquer sur le bouton "Panier" pour y avoir accès, ensuite cliquer "Supprimer" sur l\'article que v
        } elseif (stripos($question, '3') !== false) {
            $responseMessage = 'Pour visualiser un article du panier il suffit de ciquer sur le bouton "Panier" pour y avoir accès';
        } elseif (stripos($question, '4') !== false) {
            $responseMessage = 'Pour modifier la quantitée d\'un article du panier il suffit de cliquer sur le bouton "Panier" pour y avoir accès, ensuite sélectionner la nouvelle
        }
        else {
            $responseMessage = 'Désolé, je n\'ai pas compris. Peux-tu reformuler ?';
        }

        $this->say($responseMessage);
    });
}
```

- Cette fonction gère les questions liées à la configuration des motos. Elle fournit des réponses basées sur l'option choisie par l'utilisateur pour créer, visualiser ou supprimer des configurations.

```
public function askQuestionConfiguration($botman)
{
    $botman->ask('Si vous avez des problèmes pour créé, visualiser ou supprimer une Configuration de moto, précisez votre problème en répondant avec un des chiffres suivant:<
    1. Crée une configuration.<br>
    2. Supprimer une configuration.<br>
    3. Visualiser une configuration.<br>', function(Answer $answer) {
        $question = $answer->getText();

        if (stripos($question, '1') !== false) {
            $responseMessage = 'Pour créé une configuration il suffit de ciquer sur le bouton de "Moto" dans la barre des tâches en haut de votre écrant puis de selectionner
        } elseif (stripos($question, '2') !== false) {
            $responseMessage = 'Pour supprimer une configuration il suffit de ciquer sur le bouton "Profil" pour y avoir accès, ensuite cliquer sur "Mes Configuration" puis "
        } elseif (stripos($question, '3') !== false) {
            $responseMessage = 'Pour visualiser une configuration il suffit de ciquer sur le bouton "Profil" pour y avoir accès, ensuite cliquer sur "Mes Configuration"';
        } else {
            $responseMessage = 'Désolé, je n\'ai pas compris. Peux-tu reformuler ?';
        }

        $this->say($responseMessage);
    });
}
```

- C'est une fonction qui gère les questions liées aux comptes. Si l'utilisateur a des problèmes pour créer, visualiser ou supprimer des comptes, cette fonction guide l'utilisateur à travers les étapes pour accomplir ces tâches en utilisant le chatbot.

```php
public function askQuestionCompte($botman)
{
    $botman->ask('Si vous avez des problèmes de connexion ou sur votre compte, précisez votre problème en répondant avec un des chiffres suivant:<br>
    1. Crée un compte.<br>
    2. Supprimer mon compte.<br>
    3. Se connecter<br>
    4. Se deconnecter', function(Answer $answer) {
        $question = $answer->getText();

        if (stripos($question, '1') !== false) {
            $responseMessage = 'Pour créé un compte il suffit de ciquer sur le bouton de "Connexion" dans la barre des tâches en haut de votre écrant puis de selectionner "Inscr
        } elseif (stripos($question, '2') !== false) {
            $responseMessage = 'Pour supprimer son compte il suffit de ciquer sur le bouton "Profil" pour y avoir accès, ensuite cliquer sur "Modifier Mon Profil" puis "Supprime
        } elseif (stripos($question, '3') !== false) {
            $responseMessage = 'Pour créé un compte il suffit de ciquer sur le bouton de "Connexion" dans la barre des tâches en haut puis remplir vos informations de Connexion"
        } elseif (stripos($question, '4') !== false) {
            $responseMessage = 'Pour se déconnecter de son compte il suffit de ciquer sur le bouton de "Deconnexion" dans la barre des tâches en haut de votre écrant';
        } else {
            $responseMessage = 'Désolé, je n\'ai pas compris. Peux-tu reformuler ?';
        }

        $this->say($responseMessage);
    });
}
```

- C'est une fonction qui gère les questions liées aux commandes
- Le chatbot pose la question : "Si vous avez des problèmes pour visualiser ou supprimer des commandes, précisez votre problème en répondant avec un des chiffres suivants :". Deux options sont données à l'utilisateur, créer ou supprimer une commande.

```php
public function askQuestionCommande($botman)
{
    $botman->ask('Si vous avez des problèmes pour visualiser ou supprimer des commandes, précisez votre problème en répondant avec un des chiffres suivant:<br>
    1. Création d\'une commande.<br>
    2. Supprimer une commande.<br>', function(Answer $answer) {
        $question = $answer->getText();

        if (stripos($question, '1') !== false) {
            $responseMessage = 'Pour visualiser un article du panier il suffit de ciquer sur le bouton "Panier" pour y avoir accès puis "Passer Commande", entrez maintenant votre
        } elseif (stripos($question, '2') !== false) {
            $responseMessage = 'Pour supprimer une commande il suffit de ciquer sur le bouton "Profil" pour y avoir accès, ensuite cliquer sur "Mes Commandes" puis "Supprimer" sur
        }
        else {
            $responseMessage = 'Désolé, je n\'ai pas compris. Peux-tu reformuler ?';
        }

        $this->say($responseMessage);
    });
}
```

- Démonstration d'une demande d'aide pour créer un compte :
- J'écris le mot clé « aide » :