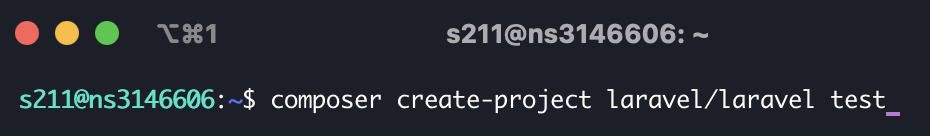
**PARTIE 1 : installation de l'environnement technique du projet**

1. Déploiement Laravel

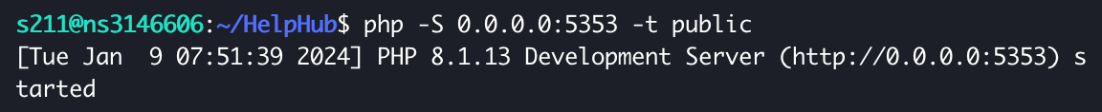
Pour commencer l’installation du framework laravel, il faut taper la commande suivante (dans le répertoire où se situera le projet) : **composer create-project laravel/laravel nom-du-projet.**



Lorsque l’installation est terminée, le projet est prêt à être utiliser. On peut également mettre le mode “debug” lors de la phase de développement : dans le fichier **config/app.php** mettre debug à **true**.



La dernière étape pour rendre le site accessible depuis un navigateur internet, est d’ouvrir un serveur “built-in”. Pour cela on peut utiliser la commande : **php –S 0.0.0.0:8080 –t public**. A savoir que le port peut être changé si celui de base est déjà occupé (s’assurer que les ports ne sont pas restreints par un pare feu).



Grâce au serveur built-in le projet sera accessible avec **adresse-serveur:port**. Chaque modification apportée au projet sera automatiquement enregistrée et déployer sur le serveur, donc inutile de le relancer à chaque fois.

1. Configuration de la base de données PostgreSQL

La deuxième étape de ce guide consiste à configurer le projet pour lui permettre la connexion avec une base de données externe.

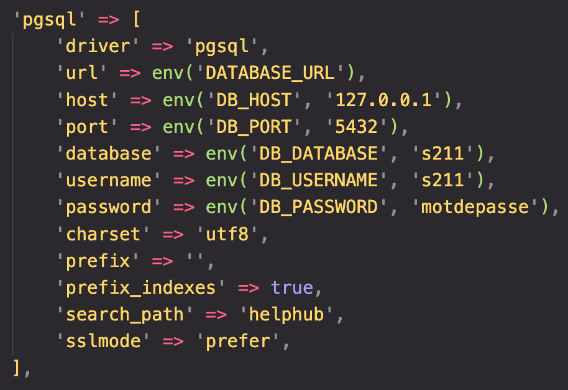
Pour commencer il faut déployer une base donnée MySQL ou PostgreSQL par exemple. Dans notre cas, nous utiliserons PostgreSQL.

Dans la racine du projet, il faut dans le fichier. env, supprimer toutes les lignes qui commencent par **DB\_**.

La suite de la configuration se trouve dans le fichier **config/database.php**. Dans le champ “default”, choisissez la connexion qui correspond à votre base. Dans notre cas nous utiliserons **pgsql** (La liste des connections se trouve dans le tableau connections en dessous).



Puis dans le tableau de connections, sur celle qui correspond à votre base de données, remplissez les informations requises à la connexion. Les champs les plus importants sont **database**, **username**, **password** et **search\_path** (qui correspondent aux informations de votre base).



1. Déploiement BotMan

Voilà comment, étape par étape, nous avons installé et configuré BotMan sur notre projet Laravel :

1. Exécuter la commande suivant dans le répertoire racine du projet Laravel :

**composer require botman/botman**

1. Exécuter cette commande au même endroit pour installer le driver web BotMan :

**Composer require botman/driver-web**

1. 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

return [

'conversation\_cache\_time' => 40,

'user\_cache\_time' => 30,

];

?>

1. 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

return [

'matchingData' => [

'driver' => 'web',

],

];

?>

1. Créer un nouveau Controller pour botman :

**php artisan make:controller BotManController**

1. 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');

1. 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 ».

<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](mailto:src='https://cdn.jsdelivr.net/npm/botman-web-widget@0/build/js/widget.js'%3e%3c/script)>

1. Copier le code suivant dans BotManController (ce code oblige les clients à commencer une conversation par « bonjour », puis récupère la question du client pour l’afficher par la suite) :

<?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 == 'Bonjour' || $message == 'bonjour') {

$this-> askQuestion($botman);

}

else{

$botman->reply("Commencez la conversasion par un Bonjour");

}

});

$botman->listen();

}

/\*\*

\* Place your BotMan logic here.

\*/

public function askQuestion($botman)

{

$botman->ask('Bonjour, comment puis-je vous aider aujourd’hui ?', function(Answer $answer) {

$question = $answer->getText();

$this->say('Votre question est bien la suivante : '.$question);

});

}

}

Voilà les réponses que nous avons implémenté :

$botman->hears('.\*(problème|problèmes).\*chargement.\*(page|pages).\*', function (BotMan $bot) {

$response = "Je suis désolé d'entendre que vous rencontrez des problèmes de chargement de pages. Voici quelques suggestions :

1. Assurez-vous d'avoir une connexion Internet stable.

2. Essayez de rafraîchir la page.

3. Vérifiez si d'autres navigateurs ou appareils fonctionnent correctement.";

$bot->reply($response);

});

$botman->hears('.\*(actions de bénévolat|dons|informations).\*ne se charge pas\*', function (BotMan $bot) {

$response = "Il semble y avoir des problèmes avec le chargement de la page des actions de bénévolat, des dons ou des informations postées par des associations.

Voici quelques suggestions :

1. Assurez-vous d'avoir une connexion Internet stable.

2. Essayez de rafraîchir la page.

3. Vérifiez si d'autres navigateurs ou appareils fonctionnent correctement.";

$bot->reply($response);

});

$botman->hears('.\*trouver.\*(association|recherche).\*', function (BotMan $bot) {

$response = "Si vous ne trouvez pas l'association que vous recherchez, assurez-vous qu'elle est bien inscrite sur le site. Vérifiez la liste des associations à gauche de l'écran sous 'Choisissez une association :'.";

$bot->reply($response);

});

$botman->hears('.\*(filtre|recherche|date).\*', function (BotMan $bot) {

$response = "Pour la recherche par date, actuellement, il n'existe pas de filtre spécifique, mais vous pouvez trier les résultats par date. Sur la gauche de l'écran, vous pouvez cocher l'option :

- Aucun triage

- Plus récents d'abord

- Plus anciens d'abord";

$bot->reply($response);

});

$botman->hears('.\*(proposer candidature|action bénévolat|participer).\*', function (BotMan $bot) {

$response = "Pour proposer votre candidature à une action bénévole, suivez ces étapes :

1. Trouvez l'action bénévole qui vous intéresse.

2. Cliquez sur 'Voir plus' en bas de la carte de l'action.

3. Sur la page de l'action, cliquez à droite sur 'Participer à l'action' si l'objectif de participation n'est pas déjà atteint.

4. Remplissez le formulaire requis.

5. Soumettez votre candidature en cliquant sur 'S'inscrire'.

Si tout est correct, vous verrez le message : 'Candidature ajoutée avec succès!'. Sinon, il peut y avoir un problème momentané, et dans ce cas, réessayez plus tard.";

$bot->reply($response);

});

$botman->hears('.\*(trouver|mes candidatures).\*', function (BotMan $bot) {

$response = "Pour trouver vos candidatures, suivez ces étapes :

1. Assurez-vous d'être connecté en tant qu'utilisateur. Si vous n'avez pas encore de compte, cliquez sur 'Inscription' en haut à droite, choisissez 'Utilisateur' et remplissez le formulaire.

2. Une fois connecté, cliquez sur 'Mon compte' en haut à droite.

3. Dans la barre de menu bleu, sélectionnez 'Mes candidatures'.

Si vous ne voyez aucune action, cela signifie que vous n'avez pas encore soumis de candidature. Sinon, vous verrez vos candidatures avec leur état précisé.";

$bot->reply($response);

});

$botman->hears('.\*(modifier|infos bancaires).\*', function (BotMan $bot) {

$response = "Pour modifier vos informations bancaires, suivez ces étapes :

1. Assurez-vous d'être connecté en tant qu'utilisateur. Si vous n'avez pas encore de compte, cliquez sur 'Inscription' en haut à droite, choisissez 'Utilisateur' et remplissez le formulaire.

2. Une fois connecté, cliquez sur 'Mon compte' en haut à droite.

3. Sélectionnez 'Mes informations bancaires'.

4. Vous verrez vos informations bancaires sur votre identité bancaire et votre carte bancaire.

5. Si les informations sont déjà enregistrées, vous pouvez les supprimer en cliquant sur le bouton rouge. Pour les modifier, éditez le texte et cliquez sur 'Enregistrer ma carte' ou 'Enregistrer mes infos bancaires'.

6. Si les informations ne sont pas encore enregistrées, remplissez les champs nécessaires, puis cliquez sur le même bouton.";

$bot->reply($response);

});

$botman->hears('.\*(mettre commentaire|commenter|action).\*', function (BotMan $bot) {

$response = "Pour mettre un commentaire sur une action, suivez ces étapes :

1. Assurez-vous d'être connecté en tant qu'utilisateur. Si vous n'avez pas encore de compte, cliquez sur 'Inscription' en haut à droite, choisissez 'Utilisateur' et remplissez le formulaire.

2. Une fois connecté, cliquez sur le logo HelpHub en haut à droite si vous êtes sur votre page personnelle.

3. Cherchez l'action que vous souhaitez commenter.

4. Faites défiler la page jusqu'à voir la section nommée 'Les commentaires'.

5. Cliquez sur 'Votre commentaire' puis commencez à écrire votre commentaire. Assurez-vous que le commentaire soit respectueux, sinon il pourrait être signalé et supprimé.

6. Cliquez ensuite sur 'Ajouter le commentaire'.

7. Vous verrez le message 'Commentaire ajouté avec succès!' en bas à droite pendant quelques secondes si cela a fonctionné. Vous pourrez ensuite voir votre commentaire avec les autres commentaires déjà présents, avec le nombre de likes indiqué.";

$bot->reply($response);

});

$botman->hears('.\*(voir|mes commentaires).\*', function (BotMan $bot) {

$response = "Pour voir vos commentaires, suivez ces étapes :

1. Assurez-vous d'être connecté en tant qu'utilisateur. Si vous n'avez pas encore de compte, cliquez sur 'Inscription' en haut à droite, choisissez 'Utilisateur' et remplissez le formulaire.

2. Une fois connecté, cliquez sur 'Mon compte' en haut à droite.

3. Sélectionnez 'Mes commentaires'.

4. Vous verrez alors vos commentaires avec le nombre de likes, le nom de l'action et un bouton pour aller sur la page de l'action.

5. Si vous n'avez encore fait de commentaire, cela sera indiqué par un message.";

$bot->reply($response);

});

$botman->hears('.\*(voir|page association).\*', function (BotMan $bot) {

$response = "Pour voir la page d'une association, suivez ces étapes :

1. Trouvez une action postée par l'association que vous recherchez.

2. Dans la barre de recherche à droite de l'écran, en dessous de 'Choisissez une association :', sélectionnez l'association dans la liste déroulante.

3. Cliquez ensuite sur le bouton 'Rechercher'.

4. Sur l'une des actions résultantes, cliquez sur le nom de l'association juste en dessous de l'image.

Vous serez alors redirigé vers la page de l'association où vous pourrez en apprendre davantage sur elle.";

$bot->reply($response);

});

$botman->hears('.\*(voir|situe|lieu|localisation).\*action bénévolat.\*', function (BotMan $bot) {

$response = "Pour voir où se situe une action de bénévolat, suivez ces étapes :

1. Recherchez l'action que vous souhaitez.

2. Vous pouvez voir directement le code postal et le département où se situe l'action.

3. Pour voir exactement où se situe l'action, cliquez sur 'Voir plus'.

4. À droite de l'écran, vous verrez l'adresse exacte indiquée, ainsi qu'une carte interactive avec un point montrant l'emplacement.

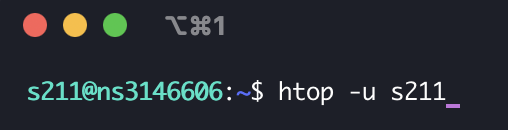
5. Si c'est écrit 'Pas de lieu spécifique', cela signifie que l'association n'a pas précisé l'endroit, ou que c'est une action en présentiel (ce sera alors précisé à droite de l'écran sur la page de l'action), ou encore que c'est une action de dons ou une information.";

$bot->reply($response);

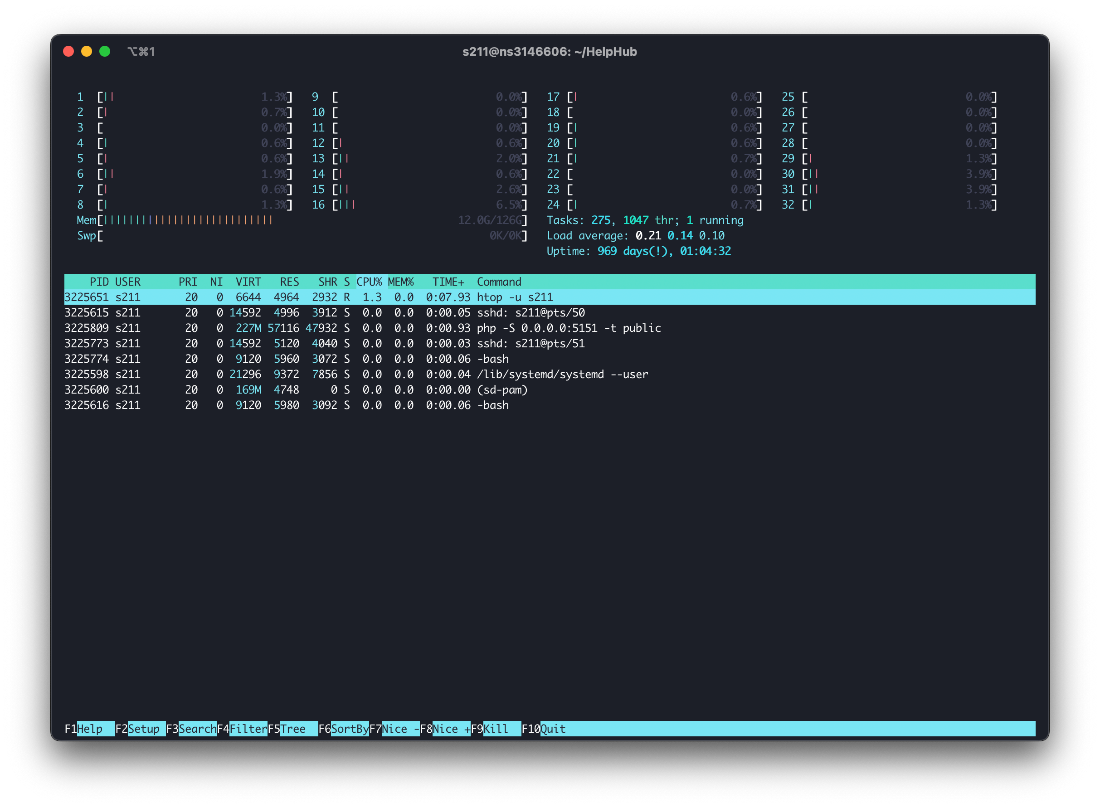
});

**PARTIE 2: La réalisation des tests de performances réseaux de votre site web**

Pour réaliser des tests de performance on peut utiliser htop, un outil utilisable en mode consol pour vérifier les statistiques de performance de la machine. Dans notre cas sachant que tous les groupes utilisent la même machine il faut donc séparer les résultats pour filtrer seulement les processus qui appartient à notre utilisateur. Pour faire cela il suffit simplement de rajouter un paramètre derrière la commande et d’indiquer l’utilisateur à cibler :

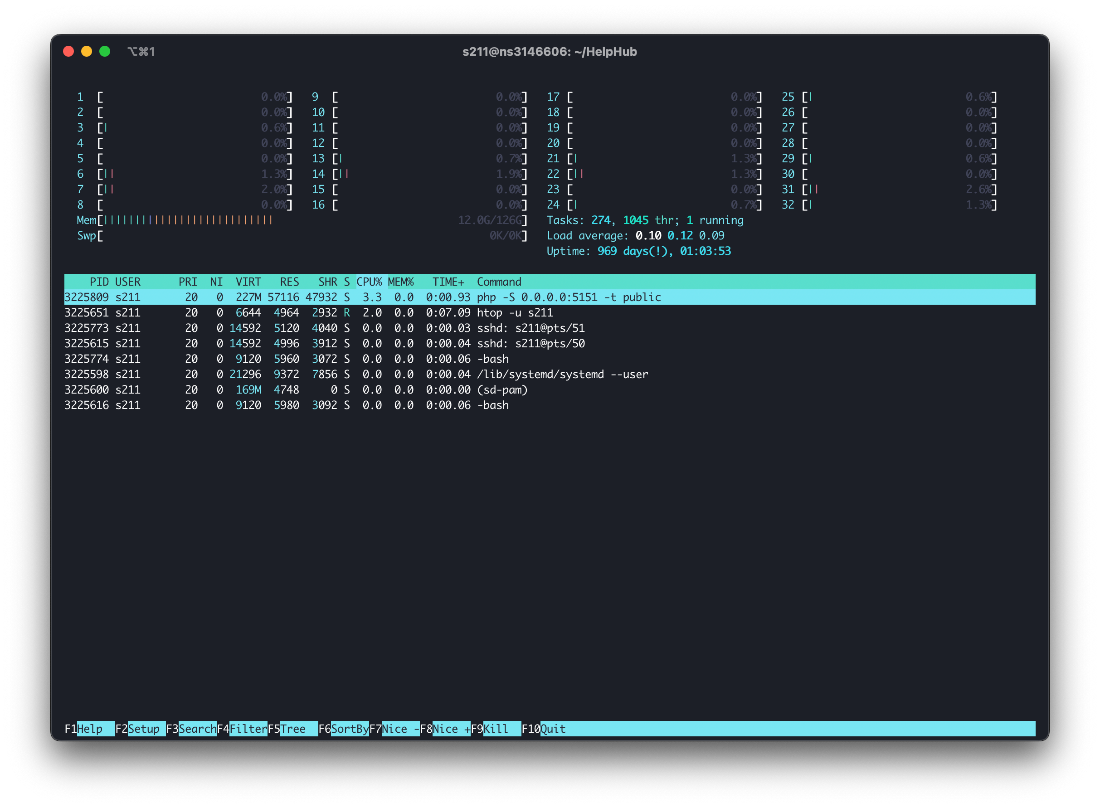


Comme on peut le voir sur la photo ci-dessous, seuls les processus appartenant à l’utilisateur passé en paramètre sont répertoriés :



Le processus qui s’occupe du projet laravel (php –S 0.0.0.0 ...) n’utilise presque aucune ressource lorsque le site est inactif (aucune requête effectuée).

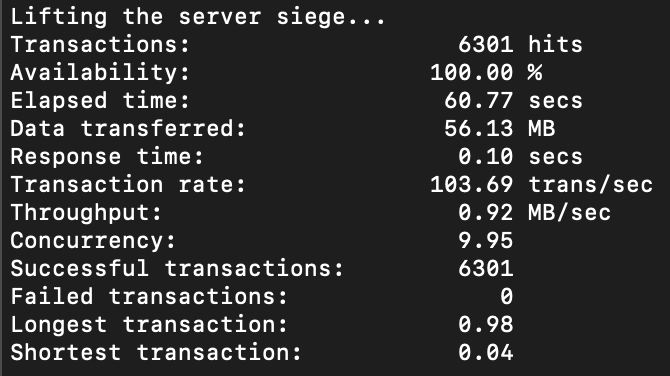
Lorsque l’on charge une page du projet, l’utilisation du CPU monte jusqu’à 3% ce qui est vraiment peu.



Nous avons simulé plusieurs connexion simultanée (100 connexions en même temps par seconde pendant 1 minute) comme on peut le voir sur la capture d’écran pendant la simulation, l’utilisation de la machine est très faible : seulement 25% du cpu est utilisé.



Voici les résultats de siege, l’outil que nous avons utilisé pour envoyer les différentes requêtes :



Sur 6300 requêtes, la totalité a réussi à accéder à la page en 0.1s ce qui est très bon (avec un interval de temps de réponse entre 0.04s et 0.98s)