Server-Side Project Documentation

Background

Our project’s backend is running Laravel, a PHP framework, and Socket.IO, a JavaScript real-time event engine. We chose these frameworks due to their abilities and robustness.

Laravel has been around for a few years and has proven itself to be a very capable and power framework for handling PHP data. It interfaces well with MySQL and utilizes reusable classes, similar to Java. Laravel is constantly being updated, which makes it a perfect candidate for a future-proof backend. The documentation is invaluable as you can find many resources on Laravel’s main website (<https://www.laravel.com/docs>) or on forums and tutorial sites like [Laracasts.com](https://www.laracasts.com/).

Socket.IO was chosen because it can trigger real-time events very easily without much trouble. It has a simple implementation but can take a few times debugging a problem before coming to a solution. The documentation is vast, but it is hard to find data on specific use cases outside of a chat application. You can find the main documentation at <https://socket.io/docs>.

Installation

Ideally you should be running the backend in three locations. You should have a local environment, a developer environment running on a server, and the live production environment. It is ideal to have these “containers” so you have a buffer when things go wrong.

Local Environment

Local installations of Laravel have a lot of choices and deciding largely depends on what device you are using. If you are using a MacOS/OS X device your choices are quite broad. You can opt to do an installation with a virtual machine using Homestead or a local server setup using Valet. Both of these have their pros and cons however.

Homestead is good if everyone on the team is running on different operating systems, you have many projects you want to run, or you want to clear your computer of functioning as a server. Homestead does require a bit more setup and can use more disk space. To install Laravel using Homestead I recommend following the steps outlines in Laravel’s official documentation: <https://laravel.com/docs/5.5/homestead>.

If you are working on a **Mac** Valet is a great choice if you are running just one or two projects and want a quick way to get up and running. Valet isn’t as robust as Homestead and can leave your computer open to security risks since you are effectively running a server all the time on your computer. Valet can be installed using the documentation here: <https://laravel.com/docs/5.5/valet>.

For **Windows** and **Linux** users it is recommended to use Homestead. XAMPP or a similar setup is not recommended as tools like Node and Composer are harder to configure on these systems (and both are required to effectively use Laravel). I recommend following the steps outlines in Laravel’s official documentation: <https://laravel.com/docs/5.5/homestead>.

Project Files

After you have installed your environment you should begin looking at the project files. We have hosted our files on Github and you can clone the repository here: <https://github.com/vehcklox/eps-laravel>. All you need to do is navigate to the place you want to install your code and clone the files to your local machine.

Project Installation

Now that you have the files, we can start coding. I will be assuming you have chosen a Homestead environment, but most of these steps should work equally well for Valet.

First you will want to bootup Homestead and ssh into the virtual machine. Then you will use the terminal to navigate to the project files (cd /path/to/project/folder). Your first step will be to copy the ‘.env.example’ file and save it as ‘.env’. If you cannot see this file, enable hidden files or use the terminal (`cp .env.example .env`). With that done you should run `composer install` and get all the dependencies installed. Next you will need to generate an app key by typing `php artisan key:generate`. After that do `npm install`. Finally you will run `npm run dev` which will compile all the files you need.

To get the database running, you will need to run ` php artisan migrate:refresh --seed`. This makes the tables and populates them with default data which will apply to every starting game.

If you are not using Homestead you will need to edit the ‘.env’ file you created and change the MIX\_APP\_IP variable to 127.0.0.1 or whatever the local ‘web server’s’ IP address is. You will also need to change the variable if you are using Homestead and changed the IP address for some reason.

You can now go to the browser and visit the URL you assigned the project in the environment setup. Doing an internet search error codes is the best way to debug Laravel. And because there are a lot of different parts, it can be hard to write every little thing that can go wrong. Therefore, please use StackOverflow, Laracasts, and Google for anything that has gone wrong so far.

Completing Setup

As mentioned before we are running Laravel and Socket.IO. The Socket.IO part is very straight forward. The only file you need to worry about is the Socket.js file located in the root directory. There are two ways to run this file. You can either run it in a process where you can see console logs, or in a process that perpetually runs. To run it in a single instance with the console log simply type `node Socket.js` while ssh’d in to your Homestead installation. You should see a declaration of the port you are using, and other console logs we left that might be useful in debugging. Otherwise you can install a package called ‘forever’ and then run `forever start socket.js` from the root directory.

Laravel also needs one more step before being fully functional. You will need to make sure Passport is running. We are using Passport to authenticate the Tablets via API tokens. The first step will be to type ` php artisan passport:keys` which generates the app keys. Next, type ` php artisan passport:client --password` to get a client id and secret. You will need to give these to the App developer on the team so they can place them in their files.

That is all for a local environment setup. It takes a few steps, but it shouldn’t take too long before you are up and running. Again, any problems you run into should be researched on the internet. The documentation for Laravel is vast and support is out there.

Live Server

Our live server is located at <http://ppc-game.fhstp.ac.at/>. Most of the setup has already been completed and should be ready to go. Here are some things to keep in mind for the live server. The web files are located in the `/ var/www/html` directory. The server has git installed and you can simply do git pull on the master branch to get the changes. The server also should have its own ‘.env’ file which is in place. You may want to save the ‘.env’ for future reference. The server also already has Passport running. You should just need the ID which is `1` and the client secret which is ` fFlnWPBxdVxvcHsycglqU8zYPaaGPW2UAbbxZ554`. This is in case the App Developer changes their parameters or loses the previous settings.

Development Server

As mentioned before, it can be helpful to run a development server so that your production server remains live. I highly recommend getting a Laravel Forge server setup for this. You will need to get a license for both Forge and a Linode server, but it can save you a lot of heartache. It has an autodeploy script which will pull your repository down every time you push a change and it takes care of most of your dev ops without much hassle. Otherwise you will need to set up another server on your own which can be time consuming.

Conclusion

Using Laravel for the backend takes some setup. But it is a powerful tool that will allow you to run the web app efficiently. The setup will involve two or three different environments which will allow you to test and debug the code. A simplified checklist style of steps follows:

Local:

1. Follow the Documentation to Install Homestead (MacOS, Linux, Windows) or Laravel Valet (MacOS).
2. Clone the repository
3. If using Homestead, SSH into the virtual machine
4. navigate to the directory with the files
5. copy the .env.example file to .env
6. Change MIX\_APP\_IP if it is different
7. type `composer install`
8. type `php artisan key:generate`
9. type `npm install`
10. type `npm run dev`
11. type `php artisan migrate:refresh --seed
12. type `php artisan passport:keys`
13. type `php artisan passport:client --password`
14. type `node socket.js`

Server:

1. ssh into <http://ppc-game.fhstp.ac.at/>
2. navigate to `/ var/www/html`
3. type `git pull` to get the latest release
4. type `forever start socket.js` if the socket file isn’t running

Server login:

url: ppc-game.fhstp.ac.at

username: admin

password: peru-cache-wrote-riley-ok-putt

Server MySQL logins:

Username: root

Password: xG.nwaa8ft^Bu6n2atAq

Username eps\_user

Password: EPS!SGR347

Server Passport Data:

Client ID: 1

Client Secret: fFlnWPBxdVxvcHsycglqU8zYPaaGPW2UAbbxZ554

Client ID: 2

Client Secret: MKOEdse28bjJMOZ6oiXG1jG7p6483NiNkyNc3hpN