# Initial Project Setup Guide:

> Requirements:

* PHP = 5.5.3

*user@trojanxxx:~/laravel/install$ php --version*

*PHP 5.5.30-1+deb.sury.org~precise+1 (cli) (built: Oct 4 2015 16:15:40)*

*Copyright (c) 1997-2015 The PHP Group*

*Zend Engine v2.5.0, Copyright (c) 1998-2015 Zend Technologies*

*with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2015, by Zend Technologies*

* Laravel installer = 1.2.1

*user@trojanxxx:~/laravel/install$ laravel --version*

*Laravel Installer version 1.2.1*

* Laravel = 5.1

*user@trojanxxx:~/laravel/dev/inzaana\_cms$ php artisan --version*

*Laravel Framework version 5.1.24 (LTS)*

* mysql = 14.14 (for PHP version 5.5.x)

*user@trojanxxx:~/laravel/install$ mysql --version*

*mysql Ver 14.14 Distrib 5.5.43, for debian-linux-gnu (i686) using readline 6.2*

* Homestead – a VM for laravel package environment (If you want to work on a secured environment – it's OPTIONAL)

[*http://laravel.com/docs/5.1/homestead*](http://laravel.com/docs/5.1/homestead)

* *Or, Host machine (Windows/ Linux/ Mac) choose your own.*

> Laravel Project Environment:

* Creates new laravel application project:

[*user@trojanxxx*](mailto:user@trojanxxx)*:~/laravel/dev$ laravel new inzaana\_cms*

* Project directory structure looks like below:

*user@trojanxxx:~/laravel/dev/inzaana\_cms$ ls -1 -a*

*.*

*..*

*app*

*artisan*

*bootstrap*

*composer.json*

*composer.lock*

*config*

*database*

*.env*

*.env.example*

*.gitattributes*

*.gitignore*

*gulpfile.js*

*.idea*

*package.json*

*phpspec.yml*

*phpunit.xml*

*public*

*readme.md*

*resources*

*server.php*

*storage*

*tests*

*vendor*

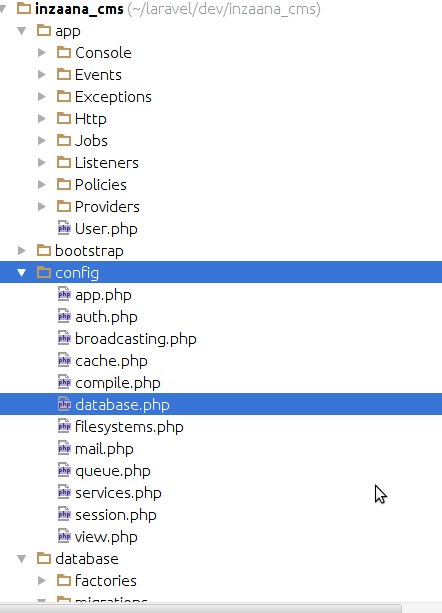
* *For more info please go to:* [*http://laravel.com/docs/5.1/installation*](http://laravel.com/docs/5.1/installation)

> Database setup:

* Create an empty database (e.g. webbuilderdb).
* Prepare your DB schema for database.
* Laravel expects each table name as pluralized.

*Example: users, categories, menus, employees …. like this.*

* From Project root directory go to: *config/database.php*



* Change default database connection name to 'mysql' if it is not like this:

*/\**

*|--------------------------------------------------------------------------*

*| Default Database Connection Name*

*|--------------------------------------------------------------------------*

*|*

*| Here you may specify which of the database connections below you wish*

*| to use as your default connection for all database work. Of course*

*| you may use many connections at once using the Database library.*

*|*

*\*/*

**'default'** => env(**'DB\_CONNECTION'**, **'mysql'**),

* Change mysql connection properties like below

*/\**

*|--------------------------------------------------------------------------*

*| Database Connections*

*|--------------------------------------------------------------------------*

*|*

*| Here are each of the database connections setup for your application.*

*| Of course, examples of configuring each database platform that is*

*| supported by Laravel is shown below to make development simple.*

*|*

*|*

*| All database work in Laravel is done through the PHP PDO facilities*

*| so make sure you have the driver for your particular database of*

*| choice installed on your machine before you begin development.*

*|*

*\*/*

**'connections'** => [

**'sqlite'** => [

**'driver'** => **'sqlite'**,

**'database'** => database\_path(**'database.sqlite'**),

**'prefix'** => **''**,

],

**'mysql'** => [

**'driver'** => **'mysql'**,

**'host'** => env(**'DB\_HOST'**, **'localhost'**),

**'database'** => env(**'DB\_DATABASE'**, **'webbuilderdb'**),

**'username'** => env(**'DB\_USERNAME'**, **'root'**),

**'password'** => env(**'DB\_PASSWORD'**, **'root1097'**),

**'charset'** => **'utf8'**,

**'collation'** => **'utf8\_unicode\_ci'**,

**'prefix'** => **''**,

**'strict'** => **false**,

],

* Now we have to decide IF we want to choose “database migration” or “SQL scripts” to prepare out database table schema.
  + ***Migration:*** Can keep the db schema of a project lifecycle under a version control which very hard to maintain to sql script – not easily readable. Can be controlled by php script.

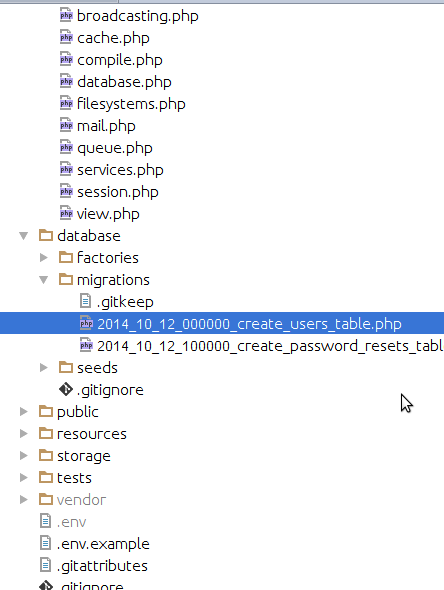
From Laracast: “*Migrations are like version control for your database, allowing a team to easily modify and share the application's database schema. Migrations are typically paired with Laravel's schema builder to easily build your application's database schema.*

*The Laravel Schema* [*facade*](http://laravel.com/docs/5.1/facades) *provides database agnostic support for creating and manipulating tables. It shares the same expressive, fluent API across all of Laravel's supported database systems.”*

* + ***SQL script:*** Already we have a script ready to go. Easy export but its not easy or readable enough to manage under version control.
* You can import SQL script from phpMyAdmin, but for migrating schema from php script you have to run couples of artisan commands:
  + When you are at project root run the command below:

[*user@trojanxxx*](mailto:user@trojanxxx)*:~/laravel/dev/inzaana\_cms$ php artisan make:migration create\_users\_table*

* You will see a script generated under “database/migrations/” using timestamp like below:



* The script will look like below:

**<?php**

**use** Illuminate\Database\Schema\Blueprint;

**use** Illuminate\Database\Migrations\Migration;

**class** CreateUsersTable **extends** Migration

{

*/\*\**

*\* Run the migrations.*

*\**

*\** ***@return*** *void*

*\*/*

**public function** up()

{

Schema::*create*(**'users'**, **function** (Blueprint $table) {

$table->increments(**'id'**);

$table->string(**'name'**);

$table->string(**'email'**)->unique();

$table->string(**'password'**, 60);

$table->rememberToken();

$table->timestamps();

});

}

*/\*\**

*\* Reverse the migrations.*

*\**

*\** ***@return*** *void*

*\*/*

**public function** down()

{

Schema::*drop*(**'users'**);

}

}

* For more info about migrations please got to: <http://laravel.com/docs/5.1/migrations>

> Relationship:

* Relationship like one to many , one to one, many to many can be done by laravel to its models
* For more info please go to: <http://laravel.com/docs/5.1/eloquent-relationships>

> For demonstration:

* We can use Seeding and Faker package for random valid data of any real life object.

> For Testing:

* We can automate laravel application testing with its test suites:
* Please got to: <http://laravel.com/docs/5.1/testing>

ALL THESE ABOVE ARE FOR BASIC KNOWING OF LARAVEL APPLICATIONS.

* To startup and build a basic application structure we have to take decisions which way we have to go through. If we follow the laravel way then it has a pretty cool architecture to go through.
* The proper decision will cost less time to develop.