1)Laravel Installation

Composer create-project laravel/laravel nomApplication –prefer-dist

2)Actionner le server

Cd nomApplication

Php artisan serve

3)Routing

Attention ceci change selon la version. La structure des fichiers change selon les versions utilisées

nomApplication/app/http/routes.php

->Créer des routes

```
Route ::get('/about',function(){
    Return 'Le contenu est ici';
});
```

->Passer une variable dans la route

```
Route::get('about/{theSubject}', function($theSubject)
{
    return $theSubject.' content goes here.';
});

Route::get('/about/classes/{theSubject}', function($theSubject)
{
    return "Content about {$theSubject} classes goes here.";
});
```

4) Reenvoyer un html view

Les templates sont stockés dans le dossier

Resources/views/

```
Route::get('/', function()
{
   return View::make('hello');
});
```

5) Les resources css peuvent être stockés dans le dossier

Public si le dossier css ou javacript n'existe pas créez les dans public

Essayer avec <link rel='stylesheet' href='< ?php echo asset('css/main.css') ; ?>' />

templating

```
<p>
    {{ $theLocation }} on {{ date('M d, Y') }}
</p>
@if($weather == 'sunny')
    <p>It's a {{$weather}} day.</p>
@else
    <p>No forecast available</p>
@endif
<p>Don't miss:</p>
<ul style="text-align:left">
@foreach($theLandmarks as $landmark)
```

```
    @unless($landmark == 'Central Park')
    <li>{{$landmark}}</li>
    @endunless
@endforeach
</ul>
```

6)Création de base de données

Créer une base de données manuellement avec phpMyAdmin
Changer les identifiants dans le fichier
nomApplication/app/config/databse.php0
pour une étrange raison il faut updater aussi le fichier .env qui définie quelques variables locales

```
Schema::create('art',function($newTable){
    $newTable->increments('id');
    $newTable->string('artist');
    $newTable->string('title',500);
    $newTable->text('description');
    $newTable->date('created');
    $newTable->date('exhibition_date');
    $newTable->timestamps();
});
```

```
//update
Schema::table('art',function($newTable){
    $newTable->boolean('alumni');
    $newTable->dropColumn('exhibition_date');
});
```

7) Migrations

Générer des fichiers de migration
Dans le dossier database/migrations

>php artisan make :migration create_art

Préparer l'objet de migration

```
/**
 * Run the migrations.
 *
 * @return void
 */
public function up()
{
    Schema::create('art2',function($newTable){
        $newTable->increments('id');
        $newTable->string('artist');
        $newTable->string('title',500);
        $newTable->text('description');
```

```
            $newTable->date('created');
            $newTable->date('exhibition_date');
            $newTable->timestamps();
        });
}

/**
 * Reverse the migrations.
 *
 * @return void
 */
public function down()
{
    Schema::drop('art2');
}
```

Ensuite lancez la ligne de commande

>php artisan migrate :install
>php artisan migrate

Pour défaire la migration migrate :rollback
>php artisan migrate :rollback

8) Générer un model avec Laravel : Ce qui nous permettra de faire des requêtes en base de données
> php artisan make :model Painting

Une fois que le fichier est généré dans firstapp/app/
On peut appeler l'objet pour insérer des informations
A partir du routeur

```
$painting = new Painting();
$painting->title = 'Joconde';
$painting->artist = 'Mona Lisa';
$painting->year=2014;
$painting->save();
```

Pour appeler des données en provenance de la base de données :
Notez que implicitement on cherche par id

```
$painting = Painting::find(1);
return $painting->title;
```

Pour updater l'instance:

```
$painting = Painting::find(1);
$painting->title='Do No Wrong - Just Do Right';
$painting->save();
return $painting->title;
```

Pour effacer utiliser “delete”

```
$painting = Painting::find(1);
$painting->delete();
return “deleted”;
```

10) Seed the database

Aller dans le dosser database>seeds
Rajouter le nom du fichier faisant l’opération de seeding dans DatabaseSeeders

```
$this->call('PaintingsTableSeeder');
```

➔ Console
Composer require fzaninotto/faker

In the seeder file, for exemple PaintingsTableSeeder

```
use App\Painting;
use Illuminate\Database\Seeder;

Class PaintingsTableSeeder extends Seeder
{
    public function run()
    {
        $faker = Faker\Factory::create();

        Painting::truncate();

        for($i=0; $i<10;$i++)
        {
            $painting = Painting::create(array(
               'title'=>$faker->realText(rand(20,40)),
                'author'=>$faker->name,
                'description'=>$faker->realText(100)
            ));
        }
    }
}
```

ligne de commande:
php artisan db:seed

11) Forms
Creation d'un formulaire sign up et redirection a une autre page "merci beaucoup"

Il faut un setup au préalable :
Rajouter un composer

```
"require": {
    "laravelcollective/html": "5.*"
}
```

Alors composer update
Ensuite,
Rajouter dans config/app.php

```
'providers' => [
Collective\Html\HtmlServiceProvider::class,
],
```

```
'aliases' => [
'Form' => Collective\Html\FormFacade::class,
'Html' => Collective\Html\HtmlFacade::class,
],
```

Rajouter encore dans config/app.php

```
'Input' => Illuminate\Support\Facades\Input::class,
```

Tapper dans le template

```
{{ Form::open(array('url'=>'thanks')) }}
    {{Form::label('email','Email Address')}}
    {{Form::text('email')}}
    {{Form::label('os','Operating System')}}
    {{Form::select('os',array(
        'linux'=>'Linux',
        'mac'=>'MAC OS X',
        'windows' => 'Windows'
    )) }}

    {{Form::label('comment','Comments')}}
    {{Form::textarea('comment','',array('placeholder'=>'What are you
interests?'))}}

    {{Form::checkbox('agree','yes',false)}}
    {{Form::label('agree','I agree to your terms of service')}}
    {{Form::submit('Sign up')}}
{{ Form::close() }}
```

Aller au controlleur

```php
Route::get('/signup',function(){
   return View::make('signup');
});

Route::post('/thanks',function(){
    $theEmail = Input::get('email');
    return View::make('thanks')->with('theEmail',$theEmail);
});
```

Exercice:

12) Créer un controlleur

Peut être créé par le biais de
>> php artisan make :controller [Name]Controller

13) Créer un repositoire

```
"require": {
    "bosnadev/repositories": "0.*"
}
```

Ou >>composer require "bosnadev/repositories: 0.*"

Etape 2

Create repository
Example

```php
<?php

namespace App\Repositories;

use Bosnadev\Repositories\Contracts\RepositoryInterface;
use Bosnadev\Repositories\Eloquent\Repository;

class PaintingsRepository extends Repository {

    public function model() {
        return 'App\Painting';
    }
}
```

Update the Model

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Painting extends Model
{
    protected $primaryKey='id';
    protected $table='paintings';


}
```

And finally prepare the Controller

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Http\Requests;
// ad Repo with alias
use App\Repositories\PaintingsRepository as Painting;

class HomeController extends Controller
{
    //add var
    private $painting;
    //prepare the repo to be used in the contructor
    public function __construct(Painting $painting)
    {
        $this->painting=$painting;
    }

    public function myMethod()
    {
        $p = $this->painting->all();

        dd($p);
        die("here in my method");
    }
}
```

13) Créer un système d'authentication

>>`php artisan make:auth`