MANUAL DE PROGRAMADOR

1- Dependencias del sistema

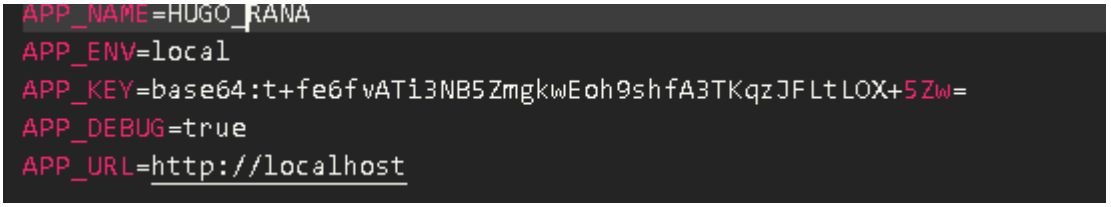
- Xampp
- Node.js
- Composer

2- Creación del proyecto

Nos situaremos en la carpeta htdocs situada en xampp (xampp/htdocs).

Ejecutaremos el siguiente comando:
htdocs> composer create-project laravel/laravel Laravel_Hugo_Raña

Nos dirigimos al archivo .env, en el cual introduciremos el nombre de la base de datos creada previamente.



Para comprobar que funciona ejecutaremos el siguiente comando:
htdocs> php artisan serv

3- Creación de migraciones

Ejecutaremos los siguientes comandos, en:

> php artisan make:migration categorias
> php artisan make:migration juegos

Para la creación de tablas según unos valores concretos nos dirigiremos a database/migrations.

```php
public function up(): void
{
    Schema::create('juegos',function(Blueprint $table){
        $table->engine="InnoDB";
        $table->id();
        $table->bigInteger('categoria_id')->unsigned();
        $table->string('nombre');
        $table->integer('valoracion');
        $table->integer('precio');
        $table->timestamps();
        $table->foreign('categoria_id')->references('id')->on('categorias')->onDelete('cascade');
    });
}
```

```php
public function up(): void
{
    Schema::create('categorias',function(Blueprint $table){
        $table->engine="InnoDB";
        $table->id();
        $table->string('nombre');
        $table->timestamps();
    });
}
```

De la misma manera crearemos la funcionalidad para destruir las tablas en caso de que existan.

```php
public function down(): void
{
    Schema::dropIfExists('juegos');
}
```

```php
public function down(): void
{
    Schema::dropIfExists('categorias');
}
```

4- Creación de sistema de autentificación

En este caso utilizamos el sistema de ui con bootstrap de laravel. Para ello ejecutaremos los siguiente comandos:

Laravel_Hugo_Raña> composer require laravel/ui
> php artisan ui bootstrap –auth
> npm install
> npm run dev

Para su correcto funcionamiento el último comando debe ser lanzado en una consola diferente, porque se mantendrá en ejecución. Mientras tanto ejecutaremos el proyecto en una consola diferente y deberá aparecer el “log in” y “register”.

5- Creación de CRUDs

Se lanzarán los siguientes comandos:

Laravel_Hugo_Raña> composer require ibex/crud-generator –dev
> php artisan vendor:publish –tag=crud
> php artisan make:crud categorias
> php artisan make:crud juegos

6- Acceso a los CRUD

Para enlazar las vistas de los crud con el sistema de autentificación deberemos añadir las siguientes líneas de código en el archivo routes/web.php.

```
Route::resource('juegos',App\Http\Controllers\JuegoController::class)->middleware('auth');
Route::resource('categorias',App\Http\Controllers\CategoriaController::class)->middleware('auth');
```

7- Creación de los enlaces

En el archivo app/resources/views/layouts/app.blade.php añadiremos las líneas de código para los enlaces.

```
@if(Auth::check())
<ul class="navbar-nav me-auto">
    <li class="nav-item">
        <a class="nav-link" href="{{ route('juegos.index')}}">{{ __('Juegos') }}</a>
    </li>

    <li class="nav-item">
        <a class="nav-link" href="{{ route('categorias.index')}}">{{ __('Categorias') }}</a>
    </li>
</ul>
@endif
```

8- Enlace de tablas

Nos dirigimos a http/Controllers/JuegoController.php añadimos las siguientes líneas de código en las funciones create() y edit().

```php
public function create()
{
    $juego = new Juego();
    $categorias = Categoria::pluck('nombre', 'id');
    return view('juego.create', compact('juego', 'categorias'));
}
```

```php
public function edit($id)
{
    $juego = Juego::find($id);
    $categorias = Categoria::pluck('nombre', 'id');

    return view('juego.edit', compact('juego','categorias'));
}
```

9- Creación de select

Añadiremos la opción de seleccionar la categoría dentro de la creación del propio juego, añadiendo las siguientes lineas de código en resources/views/juego/form.blade.php

```
<div class="box box-info padding-1">
    <div class="box-body">

        <div class="form-group">
            {{ Form::label('categoria_id') }}
            {{ Form::select('categoria', $categorias,$juego->categoria_id, ['class' => 'form-control' . ($errors->has('categori
            {!! $errors->first('categoria_id', '<div class="invalid-feedback">:message</div>') !!}
```

10- Modificación de nombre de los formularios

Laravel autoimplementa los nombre de las columnas de la tabla como campos del formulario, para cambiarlos nos dirigimos a app/resources/views/juego/index.blade.php

```
<div class="card-body">
    <div class="table-responsive">
        <table class="table table-striped table-hover">
            <thead class="thead">
                <tr>
                    <th>No</th>

                    <th>Categoria</th>
                    <th>Nombre</th>
                    <th>Valoracion</th>
                    <th>Precio</th>

                    <th></th>
```

También podemos cambiar si queremos el campo en form.blade.php

```
<div class="form-group">
    {{ Form::label('categoria') }}
```

11- Instalación de español

Para ello cambiaremos el idioma base de laravel en app/config/app.php

```
'locale' => 'es',

/*
|--------------------------------
| Application Fallback Locale
|--------------------------------
|
| The fallback locale determines
| is not available. You may cha
| the language folders that are
|
*/

'fallback_locale' => 'es',

/*
|--------------------------------
| Faker Locale
|--------------------------------
|
| This locale will be used by th
| data for your database seeds.
| localized telephone numbers,
|
*/

'faker_locale' => 'es_ES',
```

Así mismo accederemos al archivo app/resources/lang/es/validation.php, y añadiremos las siguientes líneas de código.

```php
<?php
return [
    //TODO: terminar validaciones
    /*
    |--------------------------------------------------------------------------
    | Validation Language Lines
    |--------------------------------------------------------------------------
    |
    | The following language lines contain the default error messages used by
    | the validator class. Some of these rules have multiple versions such
    | as the size rules. Feel free to tweak each of these messages here.
    |
    */

    'accepted' => 'El campo :attribute debe ser aceptado.',
    'active_url' => 'El campo :attribute no es una URL válida.',
    'after' => 'El campo :attribute debe ser una fecha posterior a :date.',
    'after_or_equal' => 'El campo :attribute debe ser una fecha posterior o igual a :date.',
    'alpha' => 'El campo :attribute solamente puede contener letras.',
    'alpha_dash' => 'El campo :attribute solamente puede contener letras, números, guiones y guiones bajos.',
    'alpha_num' => 'El campo :attribute solamente puede contener letras y números.',
    'array' => 'El campo :attribute debe ser un arreglo.',
    'before' => 'El campo :attribute debe ser una fecha anterior a :date.',
    'before_or_equal' => 'El campo :attribute debe ser una fecha anterior o igual a :date.',
    'between' => [
        'numeric' => 'El campo :attribute debe estar entre :min y :max.',
        'file' => 'El campo :attribute debe estar entre :min y :max kilobytes.',
        'string' => 'El campo :attribute debe tener entre :min y :max caracteres.',
        'array' => 'El campo :attribute debe tener entre :min y :max elementos.',
    ],
    'boolean' => 'El campo :attribute debe ser falso o verdadero.',
    'confirmed' => 'El campo de confirmación :attribute no coincide.',
    'date' => 'El campo :attribute no es una fecha válida.',
    'date_equals' => 'El campo :attribute debe ser una fecha igual a :date.',
    'date_format' => 'El campo :attribute no coincide con el formato :format.',
    'different' => 'El campo :attribute y :other deben ser diferentes.',
    'digits' => 'El campo :attribute debe tener :digits dígitos.',
    'digits_between' => 'El campo :attribute debe estar entre :min y :max dígitos.',
    'dimensions' => 'El campo :attribute tiene dimensiones de imagen inválidas.',
    'distinct' => 'El campo :attribute tiene un valor duplicado.',
    'email' => 'El campo :attribute debe ser un correo electrónico válido.',
    'exists' => 'El campo seleccionado :attribute es inválido.',
    'file' => 'El campo :attribute debe ser un archivo.',
    'filled' => 'El campo :attribute debe tener un valor.',
    'gt' => [
        'numeric' => 'El campo :attribute debe ser mayor que :value.',
        'file' => 'El campo :attribute debe ser mayor que :value kilobytes.',
        'string' => 'El campo :attribute debe ser mayor que :value caracteres.',
        'array' => 'El campo :attribute must have more than :value items.',
    ],
    'gte' => [
        'numeric' => 'El campo :attribute debe ser mayor que or equal :value.',
        'file' => 'El campo :attribute debe ser mayor que or equal :value kilobytes.',
        'string' => 'El campo :attribute debe ser mayor que or equal :value caracteres.',
        'array' => 'El campo :attribute must have :value items or more.',
    ],
    'image' => 'El campo :attribute debe ser una imagen.',
    'in' => 'El campo seleccionado :attribute es inválido.',
    'in_array' => 'El campo :attribute field does not exist in :other.',
    'integer' => 'El campo :attribute must be an integer.',
    'ip' => 'El campo :attribute must be a valid IP address.',
    'ipv4' => 'El campo :attribute must be a valid IPv4 address.',
    'ipv6' => 'El campo :attribute must be a valid IPv6 address.',
    'json' => 'El campo :attribute must be a valid JSON string.',
```

```
    'lt' => [
        'numeric' => 'El campo :attribute must be less than :value.',
        'file' => 'El campo :attribute must be less than :value kilobytes.',
        'string' => 'El campo :attribute must be less than :value caracteres.',
        'array' => 'El campo :attribute must have less than :value items.',
    ],
    'lte' => [
        'numeric' => 'El campo :attribute must be less than or equal :value.',
        'file' => 'El campo :attribute must be less than or equal :value kilobytes.',
        'string' => 'El campo :attribute must be less than or equal :value caracteres.',
        'array' => 'El campo :attribute must not have more than :value items.',
    ],
    'max' => [
        'numeric' => 'El campo :attribute no debería ser más grande que :max.',
        'file' => 'El campo :attribute no debería pesar más de :max kilobytes.',
        'string' => 'El campo :attribute no debería tener más de :max caracteres.',
        'array' => 'El campo :attribute no debería tener más de :max elementos.',
    ],
    'mimes' => 'El campo :attribute debe ser un archivo de tipo: :values.',
    'mimetypes' => 'El campo :attribute debe ser un archivo de tipo: :values.',
    'min' => [
        'numeric' => 'El campo :attribute must be at least :min.',
        'file' => 'El campo :attribute must be at least :min kilobytes.',
        'string' => 'El campo :attribute must be at least :min caracteres.',
        'array' => 'El campo :attribute must have at least :min items.',
    ],
    'not_in' => 'El campo seleccionado :attribute es inválido.',
    'not_regex' => 'El formato de :attribute es inválido.',
    'numeric' => 'El campo :attribute debe ser numérico',
    'present' => 'El campo :attribute field debe estar presente.',
```

```
    'regex' => 'El campo :attribute format es inválido.',
    'required' => 'El campo :attribute es obligatorio.',
    'required_if' => 'El campo :attribute es obligatorio cuando :other es :value.',
    'required_unless' => 'El campo :attribute es obligatorio al menos que :other esté en :values.',
    'required_with' => 'El campo :attribute es obligatorio cuando :values está presente.',
    'required_with_all' => 'El campo :attribute es obligatorio cuando :values están presentes.',
    'required_without' => 'El campo :attribute es obligatorio cuando :values no está presente',
    'required_without_all' => 'El campo :attribute es obligatorio cuando ninguno de los valores: :values está presente.',
    'same' => 'El campo :attribute y :other deben coincidir.',
    'size' => [
        'numeric' => 'El campo :attribute must be :size.',
        'file' => 'El campo :attribute must be :size kilobytes.',
        'string' => 'El campo :attribute must be :size caracteres.',
        'array' => 'El campo :attribute debe contener :size elementos.',
    ],
    'starts_with' => 'El campo :attribute con uno de los siguientes valores: :values',
    'string' => 'El campo :attribute debe ser una cadena.',
    'timezone' => 'El campo :attribute debe ser una zona horaria válida.',
    'unique' => 'El campo :attribute ya existe.',
    'uploaded' => 'Error al subir :attribute.',
    'url' => 'El formato de :attribute es inválido.',
    'uuid' => 'El campo :attribute debe ser un UUID válido.',
```

```php
/*
|--------------------------------------------------------------------------
| Custom Validation Language Lines
|--------------------------------------------------------------------------
|
| Here you may specify custom validation messages for attributes using the
| convention "attribute.rule" to name the lines. This makes it quick to
| specify a specific custom language line for a given attribute rule.
|
*/

'custom' => [
    'attribute-name' => [
        'rule-name' => 'custom-message',
    ],
],

/*
|--------------------------------------------------------------------------
| Custom Validation Attributes
|--------------------------------------------------------------------------
|
| The following language lines are used to swap our attribute placeholder
| with something more reader friendly such as "E-Mail Address" instead
| of "email". This simply helps us make our message more expressive.
|
*/

'attributes' => [],
```

Finalmente realizaremos los ajustes que queramos en la pantalla inicial en el archivo resouerces/views/welcome.blade.php