Manual del Programador

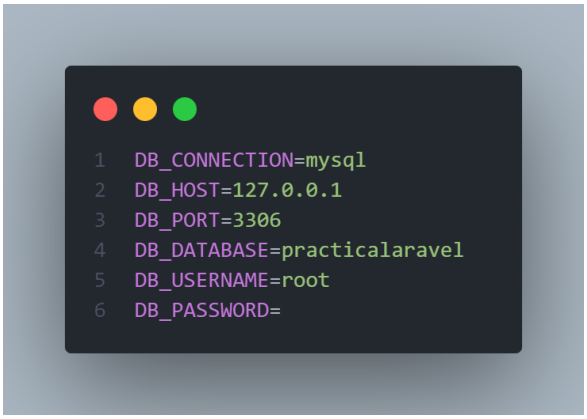
Primero instalamos estos componentes:

- Composer
- Artisan
- Xampp
- Node.js

Luego vamos a la carpeta htdocs desde consola y colocamos el siguiente comando para poder crear el proyecto laravel:

c:\xampp\htdocs\proyecto>composer create-project laravel/laravel nombre_proyecto

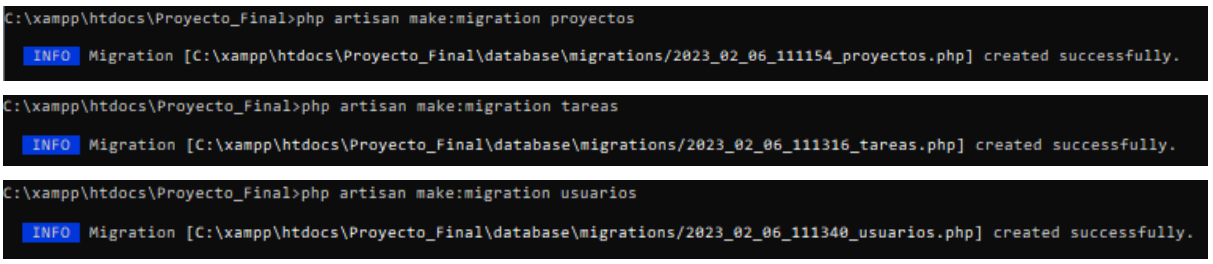
Ahora abrimos el proyecto y vamos al archivo .env y colocamos la base de datos a la que nos queremos conectar.



En la parte de DB_DATABASE quitamos la que está por defecto y agregaremos la nuestra.

Luego ejecutamos el proyecto para asegurarnos de que se creó correctamente: \proyecto> php artisan serve

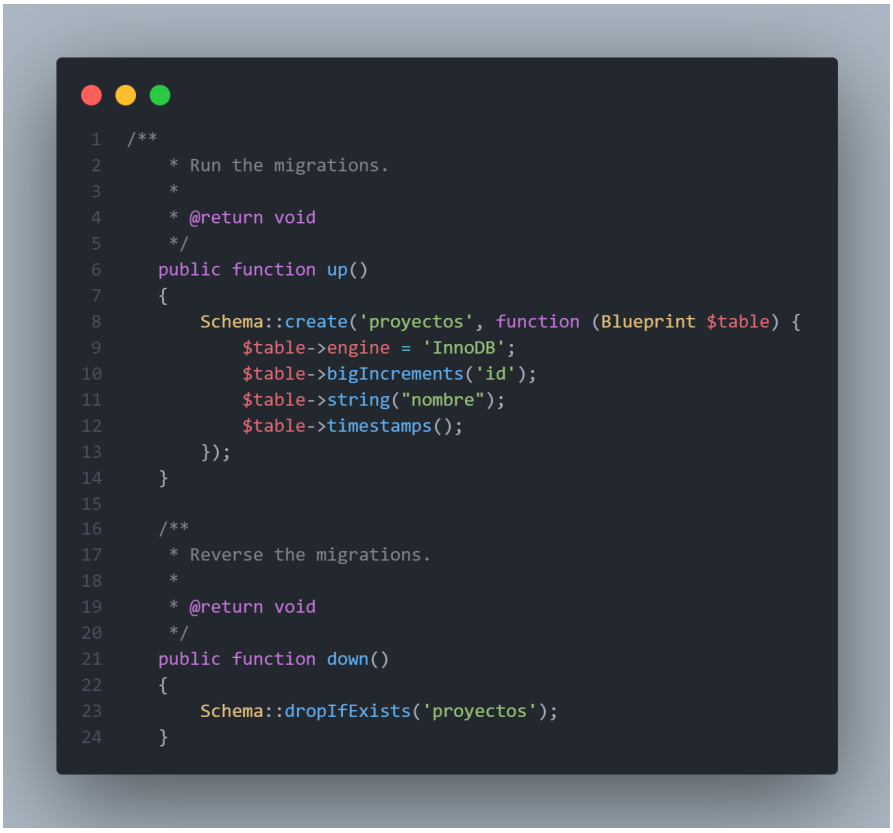
Luego vamos a crear las migraciones desde la consola en la carpeta del proyecto escribimos:



De esta forma si se han creado correctamente tendremos las tres tablas creadas en el proyecto.

Andres Alexander Licona Tome

Ahora agregamos los campos que tendrán nuestras tablas:



```php
/**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('proyectos', function (Blueprint $table) {
            $table->engine = 'InnoDB';
            $table->bigIncrements('id');
            $table->string("nombre");
            $table->timestamps();
        });
    }

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

```php
/**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('tareas', function (Blueprint $table) {
            $table->engine = 'InnoDB';
            $table->bigIncrements('id');
            $table->bigInteger('proyectos_id')->unsigned();
            $table->foreign('proyectos_id')->references('id')->on('proyectos')->cascadeOnDelete();
            $table->string('usuario');
            $table->string("descripcion");
            $table->timestamps();
        });
    }

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

La función uno es para crear la tabla con los campos que especificamos y la función down es para que en caso de que exista eliminemos la tabla.

Además asociamos los campos de la tabla tareas que hará referencia a la tabla proyectos.

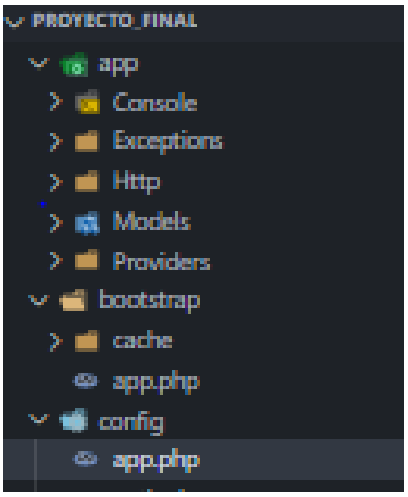
Ahora instalamos el método de autenticación:

proyecto>composer require laravel/ui
proyecto>php artisan ui bootstrap --auth
proyecto>npm install
proyecto>npm run dev

Ahora vamos a traducir los mensajes de validación al español.
Primero ejecutamos los siguiente comandos desde la carpeta del proyecto:

proyecto>composer requiere laraveles/spanish
proyecto>php artisan vendor:publish --tag=lang

Luego en nos dirigimos al archivo app/config/app.php



Dentro del archivo buscamos las variables locale y cambiamos el valor “en” a “es”.

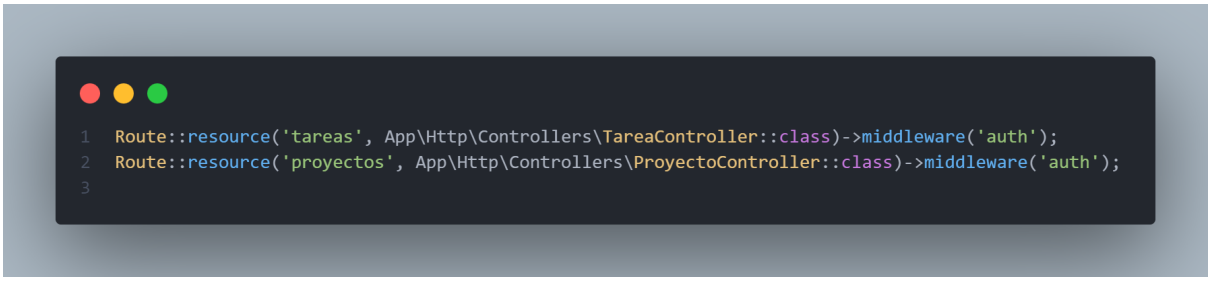
Además dejamos el de abajo el “fallback_locale” con “en” para que se muestre el mensaje en inglés en caso de que falle la traducción.

```php
/*
|--------------------------------------------------------------------------
| Application Locale Configuration
|--------------------------------------------------------------------------
|
| The application locale determines the default locale that will be used
| by the translation service provider. You are free to set this value
| to any of the locales which will be supported by the application.
|
*/

'locale' => 'es',

/*
|--------------------------------------------------------------------------
| Application Fallback Locale
|--------------------------------------------------------------------------
|
| The fallback locale determines the locale to use when the current one
| is not available. You may change the value to correspond to any of
| the language folders that are provided through your application.
|
*/

'fallback_locale' => 'en',
```

Ahora vamos a crear los CRUDs.
Lo hacemos con los siguientes comandos desde la carpeta del proyecto en la consola:

proyecto>composer require ibex/crud-generator --dev
proyecto>php artisan vendor:publish --tag=crud
proyecto>php artisan make:crud proyectos
proyectos>php artisan make:crud tareas

Luego agregamos las rutas de acceso a los CRUDs

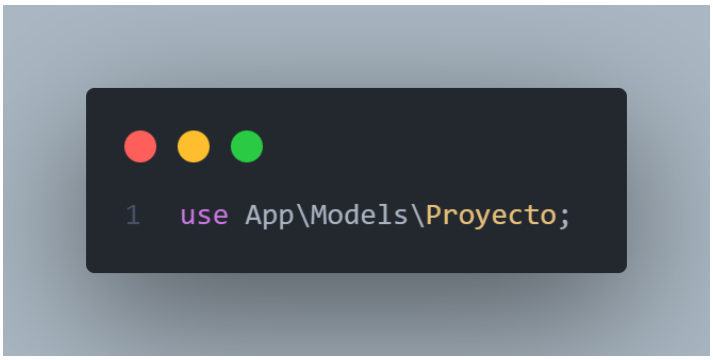


```
Route::resource('tareas', App\Http\Controllers\TareaController::class)->middleware('auth');
Route::resource('proyectos', App\Http\Controllers\ProyectoController::class)->middleware('auth');

```

Y agregamos los enlaces de las tablas a nuestra página principal:

```
<!-- Left Side Of Navbar -->
        @if (Auth::check())
            <ul class="navbar-nav me-auto">
                <li class="nav-item">
                    <a class="nav-link" href="{{ route('proyectos.index') }}">{{ __('Proyectos') }}</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="{{ route('tareas.index') }}">{{ __('Tareas') }}</a>
                </li>
            </ul>
        @endif
```

Ahora vamos a hacer unos cambios para poder acceder a los datos de las tablas. Primero nos vamos a la carpeta app\Http\Controllers\ y seleccionamos el controlador de la Tarea.
Luego agregamos el siguiente código:



```
use App\Models\Proyecto;
```

Para usar el modelo de Proyecto

```php
public function create()
{
    $tarea = new Tarea();
    $proyectos = Proyecto::pluck('nombre', 'id');
    return view('tarea.create', compact('tarea', 'proyectos'));
}
```

Para acceder a los datos de proyecto cada vez que creemos una tarea y poder asignar esa tarea en dicho proyecto, solo necesitamos agregar la segunda línea de código.

Además agregar el modelo en el compact.

```php
public function edit($id)
{
    $tarea = Tarea::find($id);
    $proyectos = Proyecto::pluck('nombre', 'id');
    return view('tarea.edit', compact('tarea','proyectos'));
}
```

Y hacemos lo mismo con la función edit, para que se haga lo mismo cada vez que se modifique una de las tareas de la tabla.

A continuación agregaremos un select para que sea más fácil identificar el proyecto en el que queremos agregar la tarea.

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

        <div class="form-group">
            {{ Form::label('proyectos_id') }}
            {{ Form::select('proyectos_id', $proyectos, ['class' => 'form-control' .
 ($errors->has('proyectos_id') ? ' is-invalid' : ''), 'placeholder' => 'Proyectos Id'
]) }}
            {!! $errors->first('proyectos_id', '<div class="invalid-feedback">:messag
e</div>') !!}
        </div>
        <div class="form-group">
            {{ Form::label('usuario') }}
            {{ Form::text('usuario', $tarea->usuario, ['class' => 'form-control' . (
$errors->has('usuario') ? ' is-invalid' : ''), 'placeholder' => 'Usuario']) }}
            {!! $errors->first('usuario', '<div class="invalid-feedback">:message</di
v>') !!}
        </div>
        <div class="form-group">
            {{ Form::label('descripcion') }}
            {{ Form::text('descripcion', $tarea->descripcion, ['class' => 'form-contr
ol' . ($errors->has('descripcion') ? ' is-invalid' : ''), 'placeholder' => 'Descripci
on']) }}
            {!! $errors->first('descripcion', '<div class="invalid-feedback">:message
</div>') !!}
        </div>

    </div>
    <div class="box-footer mt20">
        <button type="submit" class="btn btn-primary">Submit</button>
    </div>
</div>
```

En el elemento que se muestre el id de la tabla proyectos, cambiamos la segunda línea de código que estaba a text por un select, y cambiamos la asociación de la variable tarea por el nombre del proyecto.

Ahora nos vamos a al index.blade.php y en la parte de la tabla en la que muestre $tarea->proyecto_id lo cambiamos por $tarea->proyecto->nombre para que muestre el nombre del proyecto en vez del número del proyecto.

```
<thead class="thead">
    <tr>
        <th>No</th>

        <th>Proyectos Id</th>
        <th>Usuario</th>
        <th>Descripcion</th>

        <th></th>
    </tr>
</thead>
<tbody>
    @foreach ($tareas as $tarea)
        <tr>
            <td>{{ ++$i }}</td>

            <td>{{ $tarea->proyecto->nombre }}</td>
            <td>{{ $tarea->usuario }}</td>
            <td>{{ $tarea->descripcion }}</td>
```

Y hacemos lo mismo en show.blade.php para que cuando muestre la tarea lo haga con el nombre en vez del id del proyecto.

```
<div class="form-group">
    <strong>Proyectos Id:</strong>
    {{ $tarea->proyecto->nombre }}
</div>
<div class="form-group">
    <strong>Usuario:</strong>
    {{ $tarea->usuario }}
</div>
<div class="form-group">
    <strong>Descripcion:</strong>
    {{ $tarea->descripcion }}
</div>
```

Andres Alexander Licona Tome