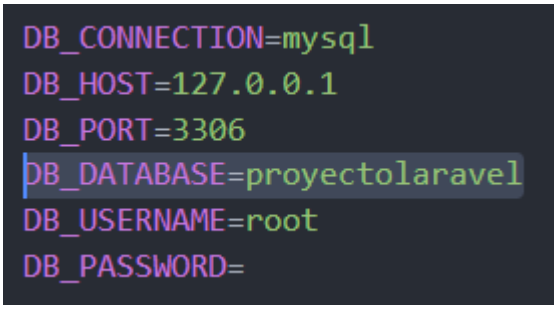
Manual Programador

Una vez creada la base de datos y el proyecto laravel se debe acceder al archivo “.env”, y allí modificar el campo “DB_DATABASE” indicando nuestra base de datos



Por requisitos para subir a github el proyecto, antes necesitaremos ejecutar los siguientes comandos en la carpeta del proyecto:

>composer require laravel/ui
>npm install

Para lanzar el proyecto se necesita ejecutar dos comandos en dos terminales distintos y dejarlas funcionando.

>npm run dev

>php artisan serve

Migraciones

En la carpeta “database/migrations” crearemos los campos que contendrán nuestras tablas. Las migraciones están formadas por dos métodos: “up()” donde especificaremos los campos de nuestra tabla y “down()” con el que podremos eliminar la tabla.

Tabla Estilos:

```php
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create("estilos", function (Blueprint $table) {
            $table->engine="InnoDB";
            $table->id();
            $table->string("nombre");
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists("estilos");
    }
};
```

Tabla Posiciones:

```php
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create("posiciones", function (Blueprint $table) {
            $table->engine="InnoDB";
            $table->id();
            $table->string("nombre");
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists("posiciones");
    }
};
```

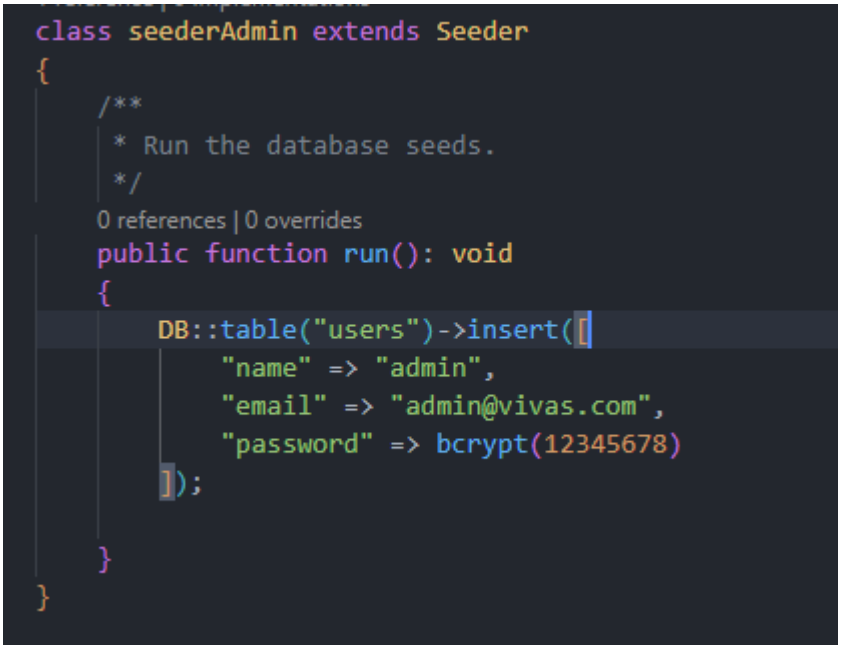
Tabla Campeones:

```php
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create("campeones", function (Blueprint $table) {
            $table->engine="InnoDB";
            $table->id();
            $table->string("nombre");
            $table->bigInteger("region");
            $table->bigInteger("estilo_ID")->unsigned()->nullable();
            $table->bigInteger("posicion_ID")->unsigned()->nullable();
            $table->timestamps();
            $table->foreign("estilo_ID")->references("id")->on("estilos")->onDelete("set null");
            $table->foreign("posicion_ID")->references("id")->on("posiciones")->onDelete("set null");
            });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
       Schema::dropIfExists("campeones");
    }
};
```

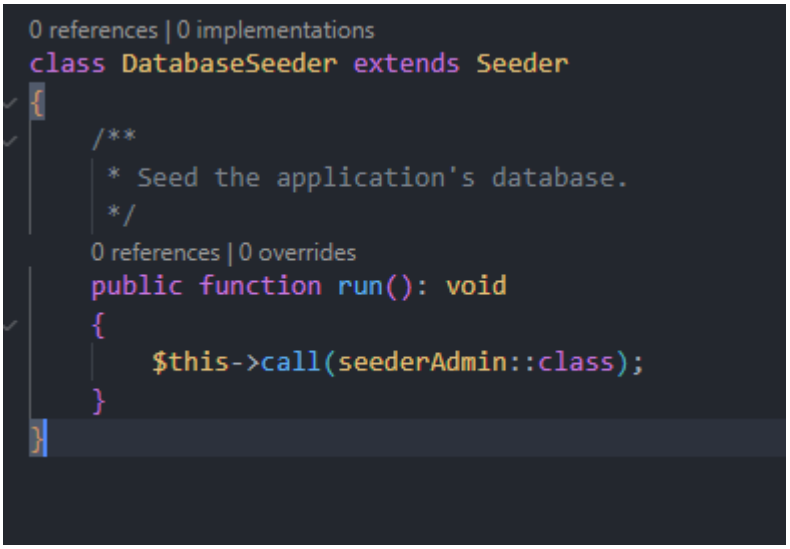
Seeder:

Se configura el seeder para crear los datos del administrador al crear las tablas



```php
class seederAdmin extends Seeder
{
    /**
     * Run the database seeds.
     */
    0 references | 0 overrides
    public function run(): void
    {
        DB::table("users")->insert([
            "name" => "admin",
            "email" => "admin@vivas.com",
            "password" => bcrypt(12345678)
        ]);

    }
}
```



```php
0 references | 0 implementations
class DatabaseSeeder extends Seeder
{
    /**
     * Seed the application's database.
     */
    0 references | 0 overrides
    public function run(): void
    {
        $this->call(seederAdmin::class);
    }
}
```

Configuración de rutas:

Posiciones y estilos serán solo usables para usuarios autentificados gracias a “->middleware(“auth”)”

```php
Route::resource("campeones",App\Http\Controllers\CampeoneController::class);
Route::resource("posiciones",App\Http\Controllers\PosicioneController::class)->middleware("auth");
Route::resource("estilos",App\Http\Controllers\EstiloController::class)->middleware("auth");
```

Enlaces:

Para poder acceder a las distintas tablas utilizaremos los botones de la barra de navegación implementados con:

```php
<div class="collapse navbar-collapse" id="navbarSupportedContent">
    <!-- Left Side Of Navbar -->
    <ul class="navbar-nav me-auto">

        <li class="nav-item">
            <a class="nav-link" href="{{ route("campeones.index")}}">{{__("Campeones")}}</a>
        </li>
        @if(Auth::check())
        <li class="nav-item">
            <a class="nav-link" href="{{ route("posiciones.index")}}">{{__("Posiciones")}}</a>
        </li>

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

Gracias al “@if(Auth::check())” los botones de Posiciones y Estilos solo serán visibles para usuarios autentificados.

Acceso de Datos Entre Tablas.

Para poder ver los datos de las tablas Posiciones y Estilos en la de Campeones se va a modificar el Create y Edit del controlador de Campeones:

```php
public function edit($id)
{
    $campeone = Campeone::find($id);
    $estilo = Estilo::pluck("nombre","id");
    $posicion = Posicion::pluck("nombre","id");

    return view('campeone.edit', compact('campeone',"estilo","posicion"));
}
```

```php
0 references | 0 overrides
public function create()
{
    $campeone = new Campeone();
    $estilo = Estilo::pluck("nombre","id");
    $posicion = Posicion::pluck("nombre","id");
    return view('campeone.create', compact('campeone',"estilo","posicion"));
}
```

Para mostrar el valor y no el ID en los distintos campos se modificará la vista Index de Campeones

```blade
<tbody>
    @foreach ($campeones as $campeone)
        <tr>
            <td>{{ ++$i }}</td>

            <td>{{ $campeone->nombre }}</td>
            <td>{{ $campeone->estilo->nombre }}</td>
            <td>{{ $campeone->posicion->nombre}}</td>

            <td>
```

Para cambiar el idioma, una vez instalado el paquete de idiomas cambiaremos el locale en “Config/app.php”

```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',
```

Modificación de Apariencia.

Se ha modificado la apariencia del index de campeones.

```
<tbody>
    @php($i = 0)
        <tr>
    @foreach ($campeones as $campeone)
        <td>
            <div class="card" style="width: 18rem;">
                <div class="card-body">
                    <h5 class="card-title">{{ $campeone->nombre }}</h5>
                    <h6 class="card-subtitle mb-2 text-muted">
                        {{ $campeone->estilo->nombre ?? 'Ningun Estilo' }}</h6>
                    <p class="card-text">
                        {{ $campeone->posicion->nombre ?? 'Ninguna Posicion' }}</p>
                    <form action="{{ route('campeones.destroy', $campeone->id) }}"
                        method="POST">
                        <a class="btn btn-outline-primary "
                            href="{{ route('campeones.show', $campeone->id) }}"><i
                                class="fa fa-fw fa-eye"></i> {{ __('Mostrar') }}</a>
                        <a class="btn btn-outline-success"
                            href="{{ route('campeones.edit', $campeone->id) }}"><i
                                class="fa fa-fw fa-edit"></i> {{ __('Editar') }}</a>
                        @csrf
                        @method('DELETE')
                        <button type="submit" class="btn btn-outline-danger"><i
                                class="fa fa-fw fa-trash"></i> {{ __('Borrar') }}</button>
                    </form>
                </div>
            </div>
        </td>
        @php($i++)
        @if ($i >= 3)
            @php($i = 0)
            </tr>
            <tr>
        @endif
    @endforeach
</tbody>
```