MODELOS

```
//nombre de la tabla en bd
protected $table = 'nombre_tabla';

//PK
protected $keyType = 'string'; //si no es int se identifica el tipo de dato
protected $primarykey = 'nombre_PK'; // si no es id, se especifica el nombre de la PK
public $incrementing = false; //la PK no autoincrementa.

public $timestamps = false; //no se utiliza timestamps en tabla equipos

use Illuminate\Database\Eloquent\SoftDeletes; //SI USA BORRADO LOGICO
        use HasFactory,SoftDeletes
```

```
    //relacion 1:N con equipo
    public function equipo():BelongsTo{
        return $this->belongsTo(Equipo::class);
    }
}
```

Controller Jugador

```
//relacion 1:N con jugadores
public function jugadores():HasMany{
    return $this->hasMany(Jugador::class);
}
```

Controller Equipo

```
public function run(): void
{
    $this->call([
        EquiposSeeder::class,
        JugadoresSeeder::class,
    ]);
}
```

```
//lleva a portada del sitio
public function index(){
    return view('home.index');
}
```

@foreach ($collection as $index => $item)

```
@foreach($jugadores as $num=>$jugador)
<tr>
    <td class="align-middle">{{$num+1}}</td>
    <td class="align-middle">{{$jugador->rut}}</td>
    <td class="align-middle">{{$jugador->apellido}}</td>
    <td class="align-middle">{{$jugador->nombre}}</td>
    <td class="align-middle">{{$jugador->numero_camiseta}}</td>
    <td class="align-middle">{{$jugador->posicion}}</td>
    <td class="align-middle">{{ $jugador->equipo->nombre }}</td>
    <td>
```

CONTROLLER

```
//index()
$jugadores = Jugador::all();
return view('jugadores.index',compact('jugadores'));

//create()
$equipos = Equipo::all(); //obtiene equipos en el orden que estan ingresados
$equipos = Equipo::orderBy('nombre')->get(); //los obtiene segun un orden en especifico
return view('jugadores.create',compact('equipos'));

//store(NombreRequest $request) //por ej JugadorRequest
$jugador = new Jugador(); //crear nuevo Jugador
$jugador ->rut = $request ->rut;//asignar valor a sus atributos
$jugador ->save();//guardar en la bd
$estadio->imagen = $request->file('imagen')->store('public/estadios');
return redirect()->route('jugadores.index');

//destroy()
$jugador ->delete();
return redirect()->route('jugadores.index');
```

Si tengo tabla Bodega(PK id) y Producto(FK id_bodega) no se necesita especificar la PK Y FK en relaciones. Si no, sí se debe específicar **REVISAR DOCUMENTACION**

```
$bodegas = Bodega::withCount('productos') //cuenta el num de productos en cada bodega
$bodega = Bodega::with('productos')->findOrFail($cod_bodega); // Carga los productos relacionados a cada bodega, findOrFail lanzara un error si no encuentra una bodega con ese código.
```

MIGRACIONES

```
$table-> string('rut',50)->primary(); //PK rut tipo string de largo 50.
$table->tinyInteger('edad');
$table->unsignedInteger('cod_asignatura');
$table->foreign('cod_asignatura')->references('cod')->on('asignaturas');//FK cod_asignatura
tinyInteger: Puede almacenar valores desde -128 hasta 127/unsigned solo valores positivos
Para PK compuesta primero se crean los campos sin primary() y luego
$table->primary(['equipo_id','partido_id']);//PK COMPUESTA

$table->softDeletes(); //BORRADO LOGICO
```

REQUEST

Los datos que se colocan para la validación son los que vienen del formulario 'name' / authorize() true
Integers: 'gte:1','lte:99:' / Strings: min & max / Rule::in(['item1','item2']) / exists:tabla,id

```
'nombre' => ['required','alpha','min:2','max:20'],
'apellido' => ['required','alpha','min:2','max:20'],
'rut' => ['required','min:9','max:10','unique:jugadores,rut'],
```

```
public function messages():array
{
    return [
        'nombre.required' => 'Indique el n
        'imagen.required' => 'Indique la i
    ];
```

1. Descargar código desde repositorio
2. Crear la BD en phpMyAdmin
3. Ejecutar composer install
4. Crear archivo .env y colocar datos de conexión a BD

DB_CONNECTION=mysql
DB_DATABASE= nombre_database

SESSION_DRIVER=file
SESSION_ENCRYPT=true

5. Ejecutar php artisan key:generate
6. Ejecutar migraciones (php artisan migrate)
7. Ejecutar seeders (si es que hay) (php artisan db:seed)
8. Ejecutar aplicación (php artisan serve)

Cuando las rutas llevan un parametro, el parametro se coloca como el segundo parametro de la funcion route
Route::delete('/jugadores**/{jugador}**',[JugadoresController::class,'destroy'])->name('jugadores.destroy');

action="{{route('jugadores.destroy'**,$jugador->rut)**}}

CREAR PROYECTO
laravel new nombre_proyecto

CONTROLADORES
//CREAR CONTROLADOR (ejemplo tabla productos)
php artisan make:controller ProductosController
//CREAR CONTROLADOR CON TODO JUNTO (menos migraciones)
php artisan make:controller ProductosController --model=Producto --resource

MIGRACIONES (tablas)
//CREAR MIGRACION
php artisan make:migration create_nombre_table
//DESHACER MIGRACION
php artisan migrate:rollback

SEEDING
//CREAR SEEDER
php artisan make:seeder NombreSeeder
//EJECUTAR SEEDER
php artisan db:seed --class=NombreSeeder

VISTAS
//CREAR VISTA
php artisan make:view productos.index (crear carpeta 'poductos' y dentro archivo index.blade.php)
LISTAR RUTAS
php artisan route:list
GENERAR RUTAS
Route::resource('/nombre',NombreCotroller::class]);
Route::get('/',[HomeController::class,'index'])->name('home.index');

REQUEST

php artisan make:request NombreRequest

TEMPLATES
master.blade.php

```
<div class="container-fluid">
    @yield('contenido-principal')
</div>
```

@extends('templates.master')
@section('contenido-principal')
@endsection

Para que el link del navbar se vea activo

```
<li class="nav-item">
    <a class="nav-link @if(Route::current()->getName()=='home.index') active @endif" aria-current="page" href="{{ route('home.index') }}">Inicio</a>
</li>
```

FORMS
<form method="POST" action=""><**@crsf**</form> //en los inputs y demás el 'name' es el q se usa para store dsp
action="{{route('jugadores.store')}}"
action="{{route('jugadores.destroy',$jugador->rut)}}
cuando se va a eliminar algo, además del *@crsf* se usa *@method('DELETE')*
//Mensaje de error

```
<div class="mb-3 col-12 col-md-6">
    <label for="nombre" class="form-label">Nombre</label>
    <input type="text" id="nombre" name="nombre" class="form-control @error('nombre') is-invalid @enderror" value="{{ old('nombre') }}">
    @error('nombre')
    <div id="nombreFeedback" class="invalid-feedback">
        {{$message}}
    </div>
    @enderror
</div>
```

//Llenar un dropdown

```
<select id="equipo" name="equipo" class="form-control @error('equipo') is-invalid @enderror">
    <option value="0">Seleccione</option>
    @foreach($equipos as $equipo)
    <option value="{{$equipo->id}}" @if(old('equipo')==$equipo->id) selected @endif>{{ $equipo->nombre }}</option>
    @endforeach
</select>
```