Eliminar archivos del disco

$path = 'credit/' . $study\_credi->personal\_id . '/' . $study\_credi->id;

                // Elimina el archivo del disco

                if (Storage::disk('public')->exists($path)) {

                    $files = Storage::disk('public')->files($path);

                    foreach ($files as $file) {

                        Storage::disk('public')->delete($file);

                    }

                    Storage::disk('public')->deleteDirectory($path);

                }

**Correr una migración en específico**

php artisan migrate --path=/database/migrations/2022\_06\_02\_143953\_create\_study\_credits\_table.php

Guardar archivos

 $directorio = 'inability/' . $inability->personal\_id . '/'  . $inability->id . '/';

//guardar los archivos en el servidor

                $fileAnexo = $request->annexes;

                if (!empty($fileAnexo)) {

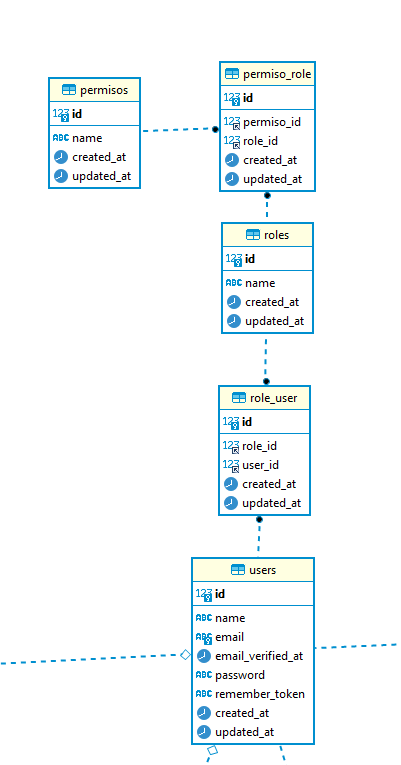
                    foreach ($fileAnexo as $files) {

                        $fileName = $files->getClientOriginalName();

                        Storage::putFileAs('public/' . $directorio, $files, $fileName);

                    }

                }

**Relaciones de muchos a muchos en Laravel**

**Migración User**

public function up()

    {

        Schema::create('users', function (Blueprint $table) {

            $table->id();

            $table->string('name');

            $table->string('email')->unique();

            $table->timestamp('email\_verified\_at')->nullable();

            $table->string('password');

            $table->rememberToken();

            $table->timestamps();

        });

    }

**Migración roles**

public function up()

    {

        Schema::create('roles', function (Blueprint $table) {

            $table->id();

            $table->string('name', 45);

            $table->timestamps();

        });

    }

**Migración Roles\_User**

 public function up()

    {

        Schema::create('role\_user', function (Blueprint $table) {

            $table->id();

            $table->unsignedBigInteger('role\_id');

            $table->unsignedBigInteger('user\_id');

            $table->foreign('role\_id')->references('id')->on('roles')->onDelete('cascade');

            $table->foreign('user\_id')->references('id')->on('users')->onDelete('cascade');

            $table->timestamps();

        });

    }

**Modelo User**

//relación de muchos a muchos con roles

    public function roles(){

        return $this->belongsToMany(Role::class);

    }

**Modelo Roles**

//relacion de muchos a muchos con usuarios

    public function users()

    {

        return $this->belongsToMany(User::class);

    }

**Insertar datos en tabla intermedia probando con tinker**

use App\Models\User;

$user = User::find(8);

**Asignar un rol al usuario**

$user->roles()->attach(1);

**Eliminar un rol**

$user->roles()->detach(1);

**Eliminar varios roles**

$user->roles()->detach([1, 2]);

**Agregar más de un rol**

$user->roles()->attach([1 ,2, 3]);

Agregar roles con el método sync. Lo que realiza es elimina

Todos los roles que ha tenido y agrega los nuevos

$user->roles()->sync([1 ,2])