2031927 – CSF302: Project Planning and Management – Coursework 23

Ashley Eatly

2031927

Table of Contents

Ashley Eatly 2031927Lab 3..

CSF304 Web Application Development– Lab Class

Randell Gaya

CSF304 Web App Development – Lab Class 4 12/10/2022

This Lab follows on from Lab 3. Please complete Lab 3 first.

Last week you created a `Person` model and a corresponding table (using a migration) named

`people`. This week you will add more tables/models and practise using relationships.

You can choose whether or not you wish to create factories and seeds. Casey's personal advise is to at least create the seeds.

Hint: The artisan command

```
sail artisan make:model NameOfModel -m
```

will create both the model `NameOfModel` and the corresponding migration (named correctly).

To see all available options use the command

```
sail artisan help make:model
```

Task 4.1

Create a car model and a corresponding migration. Each car has a make and a model. Ad- ditionally each car has a single registered owner (i.e., a person). Set up the relationships to capture this.

Use `tinker` to create some cars and check your relationships work correctly. Take some screen- shots for signing off.

Task 4.2

Create a nationality model and a corresponding migration. Nationalities only have a name, such as "British", "French", etc.

Each person can have multiple nationalities. Normally, people have only one or two nationalities, but people can have, for example, duel nationalities. For the purposes of this lab they can have unlimited nationalities. Set up the relationships to capture this.

Use `tinker` to create some nationalities and check your relationships work correctly. Take some screenshots for signing off.

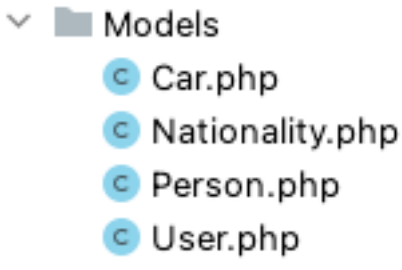
Challenge Task 4.3

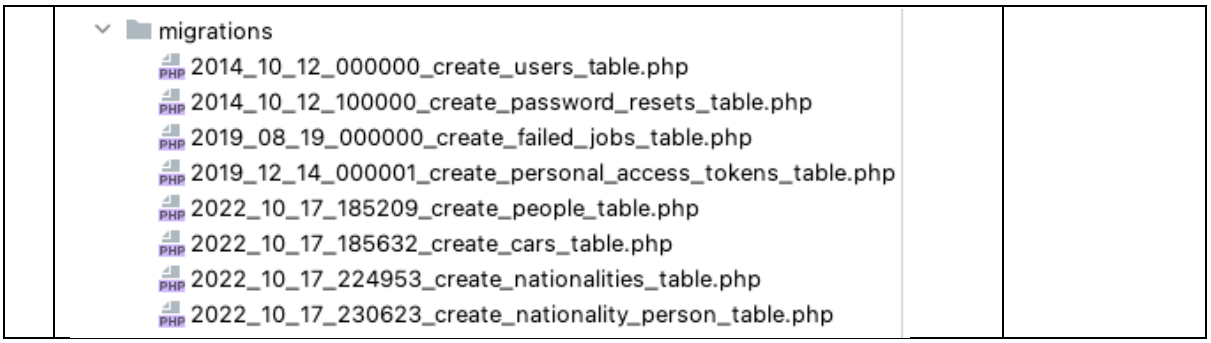
Link your models together randomly in your seed files (or factory files).

Task 4.1

Create a car model and a corresponding migration. Each car has a make and a model. Ad- ditionally each car has a single registered owner (i.e., a person). Set up the relationships to capture this.

Use `tinker` to create some cars and check your relationships work correctly. Take some screen- shots for signing off.

	Create a car model and a corresponding migration	Completed Y/N
1		
	sail up -d [+] Running 7/7 ⠿ Network lab4_sail Created 0.1s ⠿ Container lab4-mailhog-1 Started 1.1s ⠿ Container lab4-mysql-1 Started 1.2s ⠿ Container lab4-selenium-1 Started 1.2s ⠿ Container lab4-redis-1 Started 1.2s ⠿ Container lab4-meilisearch-1 Started 1.3s ⠿ Container lab4-laravel.test-1 Started	
	sail artisan make:model Car -m	
2		


migrations
2014_10_12_000000_create_users_table.php
2014_10_12_100000_create_password_resets_table.php
2019_08_19_000000_create_failed_jobs_table.php
2019_12_14_000001_create_personal_access_tokens_table.php
2022_10_17_185209_create_people_table.php
2022_10_17_185632_create_cars_table.php
2022_10_17_224953_create_nationalities_table.php
2022_10_17_230623_create_nationality_person_table.php

	Each car has a make and a model.	
	Additionally, each car has a single registered owner (i.e., a person).	
	Set up the relationships to capture this.	
	1,2014_10_12_000000_create_users_table,1 2,2014_10_12_100000_create_password_resets_table,1 3,2019_08_19_000000_create_failed_jobs_table,1 4,2019_12_14_000001_create_personal_access_tokens_table,1 5,2022_10_17_185209_create_people_table,1 6,2022_10_17_185632_create_cars_table,1 7,2022_10_17_224953_create_nationalities_table,1 8,2022_10_17_230623_create_nationality_person_table,1	

	Current status is	
	This is the Car and one to many relationship between car and person. In cars it references the person table via personid here.. which is related to the id in the person table (references id on person. The referential integrity is maintained by adding onDelete and onUpdate cascade.	

	<?php
	
	use Illuminate\Database\Migrations\Migration;
	use Illuminate\Database\Schema\Blueprint;
	use Illuminate\Support\Facades\Schema;
	
	return new class extends Migration
	{
	/**
	* Run the migrations.
	*
	* @return void
	*/
	public function up()
	{
	Schema::create('cars', function (Blueprint $table) {
	$table->bigIncrements('id');
	$table->string('make');
	$table->string('model');
	$table->unsignedBigInteger('person_id');
	// or just one car to each person
	// $table->unsignedBigInteger('person_id')->unique();
	$table->timestamps();
	$table->foreign('person_id')
	->references('id')
	->on('people')
	->onDelete('cascade')
	->onUpdate('cascade');
	});
	}
	
	/**
	* Reverse the migrations.
	*
	* @return void
	*/
	public function down()
	{
	Schema::dropIfExists('cars');
	}
	};	

Setting up relationships in Car and person models

Car

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Car extends Model
{
    use HasFactory;

    public function person():
\Illuminate\Database\Eloquent\Relations\BelongsTo
    {
        return $this -> belongsTo('App\Models\Person');
    }
}
```

Meaning a car belongs to person

Person

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Person extends Model
{
    use HasFactory;

    public function cars():
\Illuminate\Database\Eloquent\Relations\HasMany
    {
        return $this->hasMany('App\Models\Car');
    }

}
```

Meaning person has many cars potentially

I would rather create cars in seeder than use tinker but the syntax is the same…

Sail artisan make:seeder CarsTableSeeder

```php
<?php

namespace Database\Seeders;

use App\Models\Car;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class CarsTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run(): void
    {
        $c = new Car();
        $c->make = "Audi";
        $c->model = "Q5";
        $c->person_id = 1;
        $c->save();

        $c2 = new Car();
        $c2->make = "Renault";
        $c2->model = "Meganne";
        $c2->person_id = 1;
        $c2->save();

        $c3 = new Car();
        $c3->make = "Renault";
        $c3->model = "Nova";
        $c3->person_id = 2;
        $c3->save();
    }
}
```

Add to the DatabaseSeeder

```php
<?php

namespace Database\Seeders;

// use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class DatabaseSeeder extends Seeder
{
    /**
     * Seed the application's database.
     *
     * @return void
     */
    public function run(): void
    {
        $this->call([
            PeopleTableSeeder::class,
            CarsTableSeeder::class,
```

```
                        ]);
    }
}
```

```
sail artisan migrate:fresh --seed
```

(updated again to get ionfo here)

	id	make	model	person_id	created_at	updated_at
1	1	Audi	Q5	1	2022-10-27 12:03:13	2022-10-27 12:03:13
2	2	Renault	Meganne	1	2022-10-27 12:03:13	2022-10-27 12:03:13
3	3	Renault	Nova	2	2022-10-27 12:03:13	2022-10-27 12:03:13

```
sail artisan tinker
```

```
HOW MANY CARS ETC. DOES first Person have
```

```
>>> App\Models\Person::find(1)
=> App\Models\Person {#3989
     id: 1,
     title: "Mr",
     first_name: "Ashley",
     surname: "Eatly",
     address: "Ferryside 19",
     created_at: "2022-10-27 12:03:13",
     updated_at: "2022-10-27 12:03:13",
   }
```

```
>>>  App\Models\Person::find(1)->cars
=> Illuminate\Database\Eloquent\Collection {#4601
     all: [
       App\Models\Car {#3662
         id: 1,
         make: "Audi",
         model: "Q5",
         person_id: 1,
         created_at: "2022-10-27 12:03:13",
         updated_at: "2022-10-27 12:03:13",
       },
       App\Models\Car {#4608
         id: 2,
         make: "Renault",
         model: "Meganne",
         person_id: 1,
         created_at: "2022-10-27 12:03:13",
         updated_at: "2022-10-27 12:03:13",
       },
     ],
   }
```

```
HOW MANY CARS ETC. DOES second Person have
```

```
>>> App\Models\Person::find(2)->cars
=> Illuminate\Database\Eloquent\Collection {#3659
     all: [
       App\Models\Car {#4607
         id: 3,
         make: "Renault",
         model: "Nova",
         person_id: 2,
         created_at: "2022-10-27 12:03:13",
         updated_at: "2022-10-27 12:03:13",
       },
     ],
   }
```

```
(END)
```

```
HOW MANY CARS ETC. DOES third Person have (none)
```

```
>>> App\Models\Person::find(3)->cars
=> Illuminate\Database\Eloquent\Collection {#3660
     all: [],
   }
```

```
PERSON THAT owns Car 1
```

```
>>> App\Models\Car::find(1)->person
=> App\Models\Person {#4614
     id: 1,
     title: "Mr",
     first_name: "Ashley",
     surname: "Eatly",
     address: "Ferryside 19",
     created_at: "2022-10-27 12:03:13",
     updated_at: "2022-10-27 12:03:13",
   }
```

```
>>> App\Models\Car::find(2)->person
=> App\Models\Person {#3660
     id: 1,
     title: "Mr",
     first_name: "Ashley",
     surname: "Eatly",
     address: "Ferryside 19",
     created_at: "2022-10-27 12:03:13",
     updated_at: "2022-10-27 12:03:13",
   }
```

```
>>> App\Models\Car::find(3)->person
=> App\Models\Person {#4605
     id: 2,
     title: "Mrs",
     first_name: "Annette",
     surname: "Eatly",
     address: "Ferryside 19 too",
     created_at: "2022-10-27 12:03:13",
     updated_at: "2022-10-27 12:03:13",
   }
```

```
>>> App\Models\Car::find(4)->person
PHP Warning:  Attempt to read property "person" on null in Psy Shell
code on line 1
=> null
```

Task 4.2

Create a nationality model and a corresponding migration. Nationalities only have a name, such as “British”, “French”, etc.

Each person can have multiple nationalities. Normally, people have only one or two nationalities, but people can have, for example, duel nationalities. For the purposes of this lab they can have unlimited nationalities. Set up the relationships to capture this.

Use `tinker` to create some nationalities and check your relationships work correctly. Take some screenshots for signing off.

	Create a nationality model and a corresponding migration	Completed Y/N
1	sail artisan make:model Nationality -m	

	<?php
	
	use Illuminate\Database\Migrations\Migration;
	use Illuminate\Database\Schema\Blueprint;
	use Illuminate\Support\Facades\Schema;
	
	return new class extends Migration
	{
	/**
	* Run the migrations.
	*
	* @return void
	*/
	public function up()
	{
	Schema::create('nationalities', function (Blueprint $table) {
	$table->bigIncrements('id');
	$table->string('name');
	$table->timestamps();
	});
	}
	
	/**
	* Reverse the migrations.
	*
	* @return void
	*/
	public function down()
	{
	Schema::dropIfExists('nationalities');
	}
	};	

	Only has name as string	
	sail artisan make:seeder NationalitiesTableSeeder	
	class NationalitiesTableSeeder extends Seeder {	

```php
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        $n = new Nationality();
        $n->name = 'British';
        $n->save();

        $n1 = new Nationality();
        $n1->name = 'French';
        $n1->save();

        $n2 = new Nationality();
        $n2->name = 'Italian';
        $n2->save();

    }
}
```

Add seeder to the Database Seeder

```php
<?php

namespace Database\Seeders;

// use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class DatabaseSeeder extends Seeder
{
    /**
     * Seed the application's database.
     *
     * @return void
     */
    public function run(): void
    {
        $this->call([
            PeopleTableSeeder::class,
            CarsTableSeeder::class,
            NationalitiesTableSeeder::class,
        ]);

    }
}
```

sail artisan migrate:fresh --seed

	id	name	created_at	updated_at
1	1	British	2022-10-27 12:37:26	2022-10-27 12:37:26
2	2	French	2022-10-27 12:37:26	2022-10-27 12:37:26
3	3	Italian	2022-10-27 12:37:26	2022-10-27 12:37:26

	However skipping ahead each person can have multiple nationality this means we have a one to many relationship and we will need a link or pivot table.	
	sail artisan make:migration create_nationality_person_table	
	Convention is in pivot table to have them in alphabetical order – so here its nationality person. Note a person can have many nationalities and A nationality has many people.	
	Setting Up the relationships first (Person.php class)	

	<?php
	
	namespace App\Models;
	
	use Illuminate\Database\Eloquent\Factories\HasFactory;
	use Illuminate\Database\Eloquent\Model;
	
	class Person extends Model
	{
	use HasFactory;
	
	public function cars():
	\Illuminate\Database\Eloquent\Relations\HasMany
	{
	return $this->hasMany('App\Models\Car');
	}
	
	public function nationalities():
	\Illuminate\Database\Eloquent\Relations\BelongsToMany
	{
	return $this-
	>belongsToMany('App\Models\Nationality');;
	}
	}	

Now the create_nationality_person_table.php

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('nationality_person', function
(Blueprint $table) {
            $table-
>primary(['nationality_id','person_id']);
            $table->unsignedBigInteger('nationality_id');
            $table->unsignedBigInteger('person_id');
            $table->timestamps();

            $table->foreign('nationality_id')
                ->references('id')
                ->on('nationalities')
                ->onDelete('cascade')
                ->onUpdate('cascade');
            $table->foreign('person_id')
                ->references('id')
                ->on('people')
                ->onDelete('cascade')
                ->onUpdate('cascade');

        });
    }

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

We also have to add the relationship in Person model

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Person extends Model
{
    use HasFactory;

    public function cars():
\Illuminate\Database\Eloquent\Relations\HasMany
    {
        return $this->hasMany('App\Models\Car');
    }

    public function nationalities():
\Illuminate\Database\Eloquent\Relations\BelongsToMany
    {
       return $this->belongsToMany('App\Models\Nationality');;
    }
}
```

And the Nationality Model

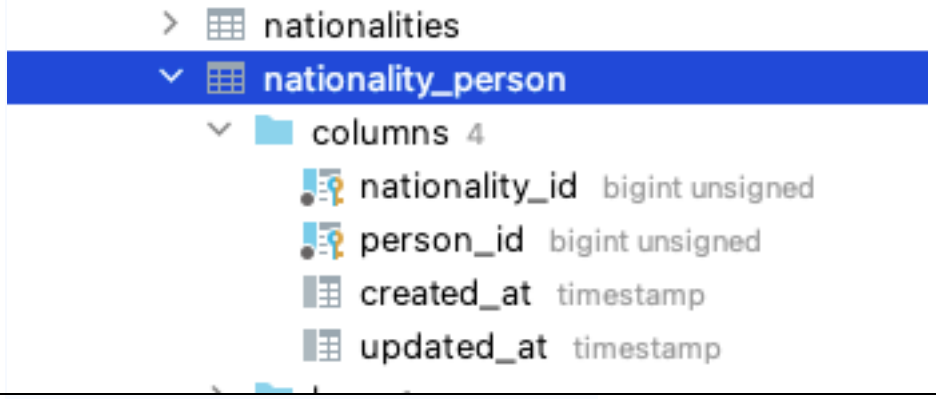
```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Nationality extends Model
{
    use HasFactory;

    public function people():
\Illuminate\Database\Eloquent\Relations\BelongsToMany
    {
        return $this->belongsToMany('App\Models\Person');
    }
}
```

Change the NationalityTablesSeeder too to attach people to Nationalities

```php
class NationalitiesTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        $n = new Nationality();
        $n->name = 'British';
        $n->save();

        $n1 = new Nationality();
```

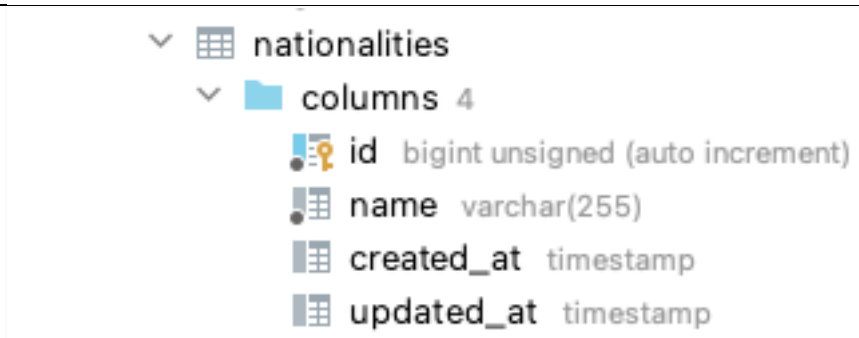
```
        $n1->name = 'French';
        $n1->save();

        $n2 = new Nationality();
        $n2->name = 'Italian';
        $n2->save();

        $n->people()->attach(1);
        $n->people()->attach(2);
        $n2->people()->attach(2);
    }
}
```

```
sail artisan migrate:fresh --seed
```



We probably don't need the timestamps

	nationality_id	person_id
1	1	1
2	1	2
3	3	2



	id	name	created_at	updated_at
1	1	British	2022-10-27 12:55:37	2022-10-27 12:55:37
2	2	French	2022-10-27 12:55:37	2022-10-27 12:55:37
3	3	Italian	2022-10-27 12:55:37	2022-10-27 12:55:37

```
Decided to revisit and update as I had items from 2022-10-17 etc.
```

```
=> Illuminate\Database\Eloquent\Collection {#4603
    all: [
      App\Models\Nationality {#4604
        id: 1,
        name: "British",
        created_at: "2022-10-17 23:27:46",
        updated_at: "2022-10-17 23:27:46",
        pivot: Illuminate\Database\Eloquent\Relations\Pivot {#4601
          person_id: 1,
          nationality_id: 1,
        },
      },
:
>>> App\Models\Person::find(2)->nationalities
=> Illuminate\Database\Eloquent\Collection {#4606
    all: [
      App\Models\Nationality {#4609
        id: 1,
        name: "British",
        created_at: "2022-10-17 23:27:46",
        updated_at: "2022-10-17 23:27:46",
        pivot: Illuminate\Database\Eloquent\Relations\Pivot {#4605
          person_id: 2,
          nationality_id: 1,
        },
      },
      App\Models\Nationality {#4610
        id: 3,
        name: "Italian",
        created_at: "2022-10-17 23:27:46",
        updated_at: "2022-10-17 23:27:46",
        pivot: Illuminate\Database\Eloquent\Relations\Pivot {#3662
          person_id: 2,
          nationality_id: 3,
        },
      },
    ],
  }
```

	sail artisan tinker	
Person 1		
	>>> App\Models\Person::find(1)->nationalities => Illuminate\Database\Eloquent\Collection {#4605 all: [App\Models\Nationality {#4606 id: 1, name: “British”, created_at: “2022-10-27 12:55:37”, updated_at: “2022-10-27 12:55:37”, pivot: Illuminate\Database\Eloquent\Relations\Pivot {#4603 person_id: 1, nationality_id: 1, }, },],	
	Person 2	
	App\Models\Person::find(2)->nationalities => Illuminate\Database\Eloquent\Collection {#4608 all: [App\Models\Nationality {#4611 id: 1, name: “British”, created_at: “2022-10-27 12:55:37”, updated_at: “2022-10-27 12:55:37”, pivot: Illuminate\Database\Eloquent\Relations\Pivot {#4607 person_id: 2, nationality_id: 1, }, }, App\Models\Nationality {#4612 id: 3, name: “Italian”, created_at: “2022-10-27 12:55:37”, updated_at: “2022-10-27 12:55:37”, pivot: Illuminate\Database\Eloquent\Relations\Pivot {#3664 person_id: 2, nationality_id: 3, }, },], } (END)	

	Person 3 not setup	

	>>> App\Models\Person::find(3)->nationalities
	=> Illuminate\Database\Eloquent\Collection {#4619
	all: [],
	}	

	Find people who are Britsh	

	>>> App\Models\Nationality::find(1)->people
	=> Illuminate\Database\Eloquent\Collection {#4621
	all: [
	App\Models\Person {#4626
	id: 1,
	title: "Mr",
	first_name: "Ashley",
	surname: "Eatly",
	address: "Ferryside 19",
	created_at: "2022-10-27 12:55:36",
	updated_at: "2022-10-27 12:55:36",
	pivot: Illuminate\Database\Eloquent\Relations\Pivot {#4620
	nationality_id: 1,
	person_id: 1,
	},
	},
	App\Models\Person {#4624
	id: 2,
	title: "Mrs",
	first_name: "Annette",
	surname: "Eatly",
	address: "Ferryside 19 too",
	created_at: "2022-10-27 12:55:36",
	updated_at: "2022-10-27 12:55:36",
	pivot: Illuminate\Database\Eloquent\Relations\Pivot {#4623
	nationality_id: 1,
	person_id: 2,
	},
	},
	],
	}	

	Find people who are French none	

	>>> App\Models\Nationality::find(2)->people
	=> Illuminate\Database\Eloquent\Collection {#4630
	all: [],
	}	

	Find people who are Italian	
	>>> App\Models\Nationality::find(3)->people => Illuminate\Database\Eloquent\Collection {#4614 all: [	

	App\Models\Person {#4635 id: 2, title: "Mrs", first_name: "Annette", surname: "Eatly", address: "Ferryside 19 too", created_at: "2022-10-27 12:55:36", updated_at: "2022-10-27 12:55:36", pivot: Illuminate\Database\Eloquent\Relations\Pivot {#4629 nationality_id: 3, person_id: 2, }, },], }	
	YES THE RELTIONSHIPS WORK	

Challenge Task 4.3

Link your models together randomly in your seed files (or factory files).