

**SESSION
05**

THE ELOQUENT ORM

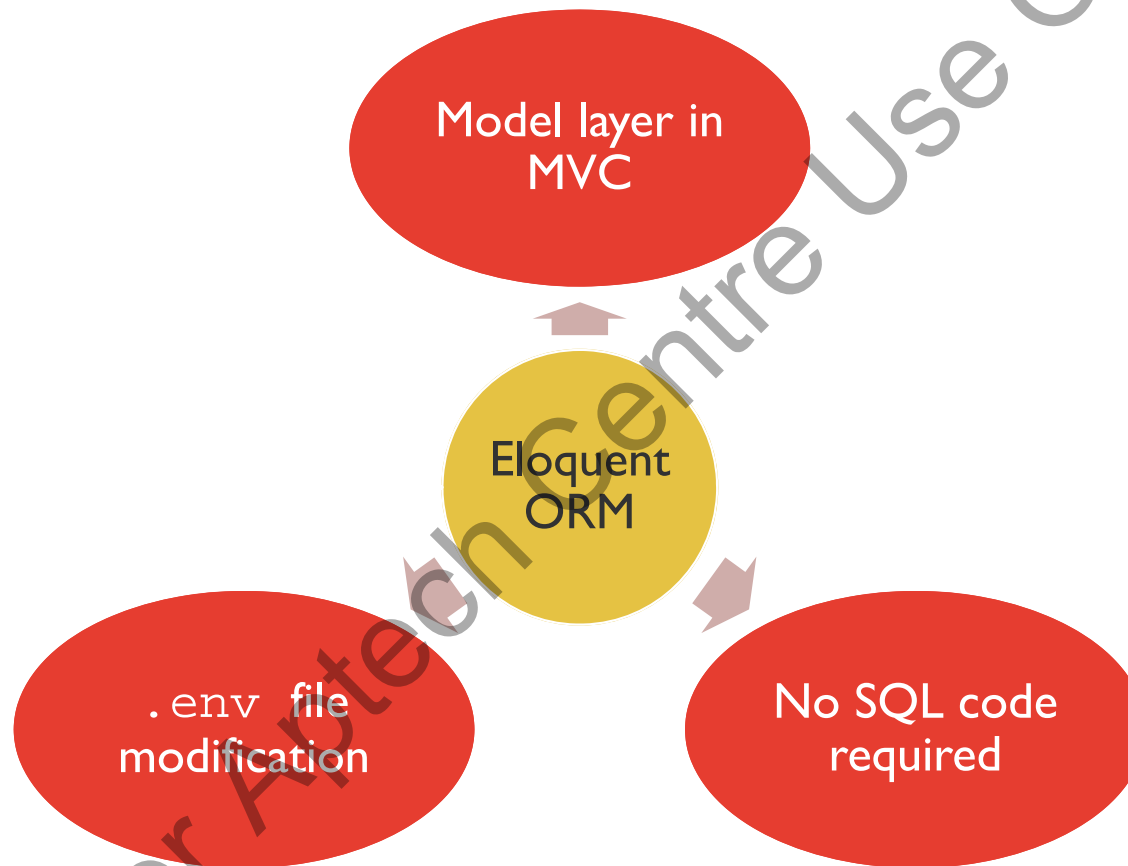


OBJECTIVES

- Describe the functionalities offered by Eloquent Object Relational Mapping (ORM)
- Learn to perform database operations using Eloquent ORM
- Understand database testing using Routes and Eloquent ORM



WHAT IS ELOQUENT ORM?



ELOQUENT CONVENTIONS AND THE MIGHTY MODEL (1-2)

Basic conventions of Eloquent

- Studly-case
- Snake-case



ELOQUENT CONVENTIONS AND THE MIGHTY MODEL (2-2)

```
php artisan make:model Employee
```

```
<?php
namespace HRM;
use Illuminate\Database\Eloquent\Model;
class Employee extends Model {
    //
}
```

```
<?php
use HRM\Employee;
Route::get('/', function () {
    return Employee::all();
});
```

INSERTING, RETRIEVING, SAVING, AND DELETING DATA (1-9)

Creating a Record:

```
$data = ['name' => 'Albert', 'salary' => 90000, 'designation' => 'Senior Engineer'];
```

```
use HRM\Employee;  
$employee = new Employee;
```

```
$employee->create($data);
```

```
<?php  
use HRM\Employee;  
  
Route::get('/', function () {  
    $data = ['name' => 'Albert', 'salary' => 90000, 'designation' => 'Senior Engineer'];  
    $employee = new Employee;  
    $employee->create($data);  
});
```

INSERTING, RETRIEVING, SAVING, AND DELETING DATA (2-9)

```
<?php

namespace HRM;

use Illuminate\Database\Eloquent\Model;

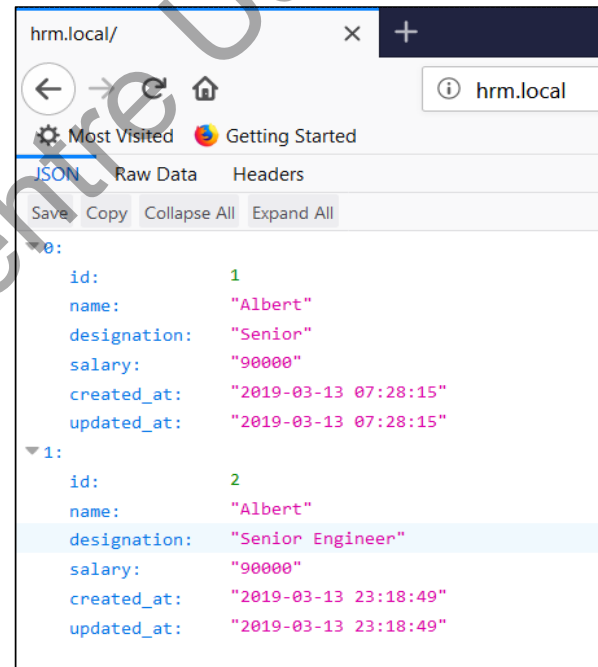
class Employee extends Model
{
    protected $fillable = ['name', 'salary', 'designation'];
}
```



INSERTING, RETRIEVING, SAVING, AND DELETING DATA (3-9)

Retrieving Records:

```
<?php
use HRM\local;
Route::get('/', function() {
    return Employee::all();
});
```



INSERTING, RETRIEVING, SAVING, AND DELETING DATA (4-9)

all() Method - Method to retrieve records.

find() Method - Used to view a specific record.

where Method - Displays same results as find () and provides the option to select the column name.

get method - Helps in extracting specific columns.



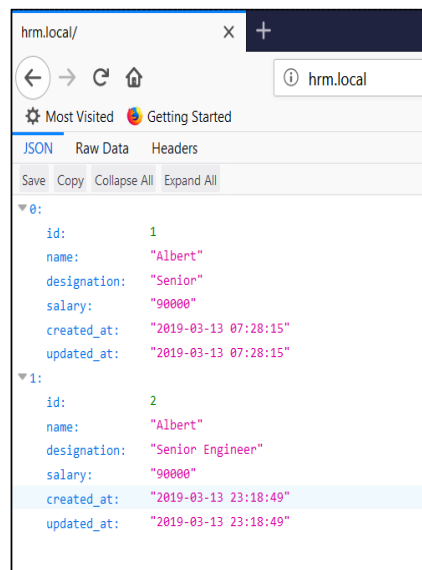
INSERTING, RETRIEVING, SAVING, AND DELETING DATA (5-9)

Modifying and Saving Records:

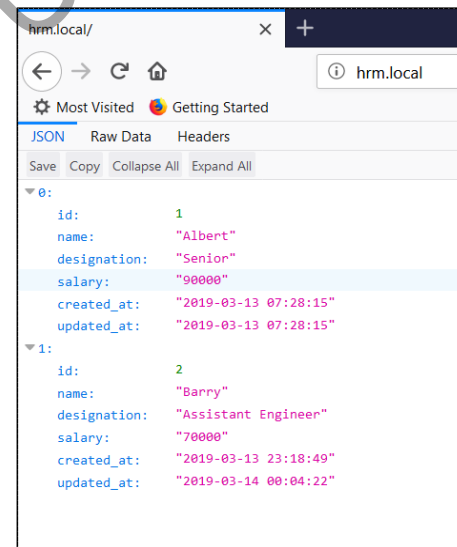
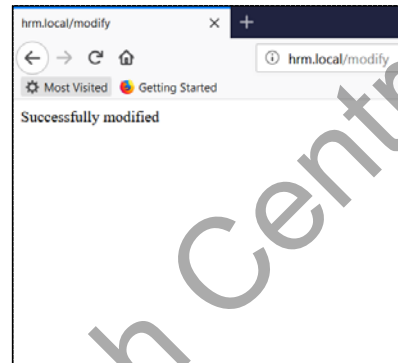
```
$employee = Employee::find(1);  
$id = $employee->id;  
$name = $employee->name;
```

```
<?php  
  
use HRM\Employee;  
  
Route::get('modify',function () {  
    $employee = Employee::find(2);  
    $employee->name = 'Barry';  
    $employee->salary = 70000;  
    $employee->designation = "Assistant Engineer";  
    $employee->save();  
    return "Successfully modified";  
});  
  
Route::get('/', function() {  
    return Employee::all();  
});
```

INSERTING, RETRIEVING, SAVING, AND DELETING DATA (6-9)



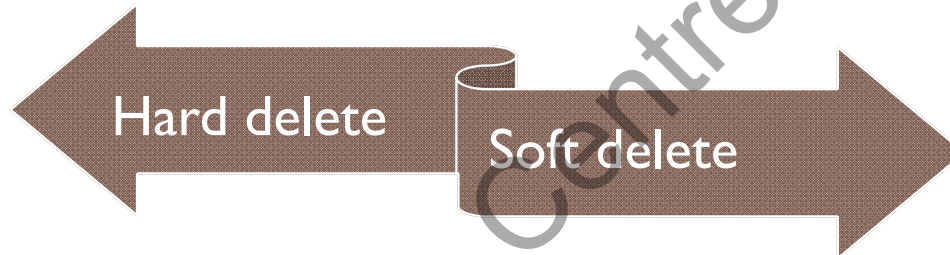
```
hrm.local/  
JSON Raw Data Headers  
Save Copy Collapse All Expand All  
0:  
  id: 1  
  name: "Albert"  
  designation: "Senior"  
  salary: "90000"  
  created_at: "2019-03-13 07:28:15"  
  updated_at: "2019-03-13 07:28:15"  
1:  
  id: 2  
  name: "Albert"  
  designation: "Senior Engineer"  
  salary: "90000"  
  created_at: "2019-03-13 23:18:49"  
  updated_at: "2019-03-13 23:18:49"
```



```
hrm.local/  
JSON Raw Data Headers  
Save Copy Collapse All Expand All  
0:  
  id: 1  
  name: "Albert"  
  designation: "Senior"  
  salary: "90000"  
  created_at: "2019-03-13 07:28:15"  
  updated_at: "2019-03-13 07:28:15"  
1:  
  id: 2  
  name: "Barry"  
  designation: "Assistant Engineer"  
  salary: "70000"  
  created_at: "2019-03-13 23:18:49"  
  updated_at: "2019-03-14 00:04:22"
```

INSERTING, RETRIEVING, SAVING, AND DELETING DATA (7-9)

Deleting Records:



Two ways to delete records:

- ⑩ Retrieving the record and using the delete method.
- ⑩ Using the destroy () method from the Model itself.

INSERTING, RETRIEVING, SAVING, AND DELETING DATA (8-9)

```
//using the delete method  
$employee = Employee::find(1);  
$employee->delete();
```

```
Employee::destroy(2);
```



INSERTING, RETRIEVING, SAVING, AND DELETING DATA (9-9)

```
<?php

use HRM\Employee;

Route::get('modify',function () {
    $employee = Employee::find(2);
    $employee->name = 'Barry';
    $employee->salary = 70000;
    $employee->designation = "Assistant
Engineer";
    $employee->save();
    return "Successfully modified";
});

Route::get('/', function() {
    return Employee::all();
});
```

```
Route::get('delete', function() {
    $employee = Employee::find(1);
    $employee->delete();
    return 'The record has been deleted';
});

Route::get('destroy', function(){
    Employee::destroy(2);
    return 'The record has been destroyed
';
});
```

QUERY SCOPES

Query scopes are defined in the Model and replace the long chain with a simple one.

```
$employee->where('salary','>',80000)->where('designation','=','Senior Engineer');
```

```
public function highPaidHighRank($query)
{
    return $query->where('salary','>',80000)->where('designation','=','Senior Engineer');
}
```

```
<?php

namespace HRM;

use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Database\Eloquent\Model;

class Employee extends Model
{
    use SoftDeletes;
    protected $fillable = ['name','salary','designation'];
    protected $dates = ['deleted_at'];
    public function highPaidHighRank($query)
    {
        return $query->where('salary','>',80000)->where('designation','=','Senior Engineer');
    }
}
```

MODEL EVENTS (1-3)

List of methods that Eloquent models can fire:

- creating
- created
- saving
- saved
- deleting
- deleted
- updating
- updated
- restoring
- restored



MODEL EVENTS (2-3)

Event Listener

- Internal function, in computer programs, that waits for an event to occur.
- `EventServiceProvider` located in `app/providers/EventServiceProvider`
- All events (keys) and their listeners (values) are included in the `listen` property.

MODEL EVENTS (3-3)

```
<?php

namespace simpleApp\Providers;

use Illuminate\Support\Facades\Event;
use Illuminate\Auth\Events\Registered;
use Illuminate\Auth\Listeners\SendEmailVerificationNotification;
use Illuminate\Foundation\Support\Providers\EventServiceProvider as ServiceProvider;

class EventServiceProvider extends ServiceProvider
{
    /**
     * The event listener mappings for the application.
     *
     * @var array
     */
    protected $listen = [
        Registered::class => [
            SendEmailVerificationNotification::class,
        ],
    ];
}
```

```
/**
 * Register any events for your application.
 *
 * @return void
 */
public function boot()
{
    parent::boot();

    Employee::deleting(function($employee){
        // the employee variable will get the employee record that is being deleted
        // Write more code to do some validation before deleting
    })
}
}
```

RELATIONSHIPS AND COLLECTIONS

Relationships – Common fields

Collections – Chaining of methods



SUMMARY

- Eloquent is a tool to write Models for databases without writing any SQL code.
- Models is part of the MVC design pattern.
- Eloquent has basic conventions that is required to be followed to make operations simpler.
- It follows the CRUD structure of operation.
- Query scopes are functions that are defined inside a Model; they can replace a long-chained method with a simple method.
- There are numerous events in Model, such as creating, created, deleted, and saving.
- The collection object in Eloquent ORM allows the chaining of methods, which facilitates interaction with the database without using a SQL query.

