Conceptes bàsics de Laravel

Conceptes bàsics	Passos a seguir per un projecte bàsic
1- Namespace i MVC 2- Migrations i Taules 3- Bastiments 4- Models i Controllers 5- Views i Controllers 6- Blades i Blade Templates 7- Routing 8- Request & Responses 9- Middleware 10- Seeders i Factories	1- Creació de base de dades, usuari i permisos amb un script sql 2- Creació d'un projecte amb l'ordre laravel 3- Configuració del projecte modificant l'arxiu .env 4- Creació de taules del projecte: • Creció dels fitxers de Migrations • Execució de les Migrations => Creació taules 5- [opcional] Afegiment de Middleware d'autenticació 6-Desenvolupament dels Models 7- Desenvolupament dels Controllers 8- Creació de Routes 9- Creació de Views (blades)

Treballant amb Controllers. Començant a utilitzar Routes i vistes

1- En el **patró de programació MVC** (Model-Vista-Controlador), un **Controlador (Controller):**

- Es la part de l'aplicació que rediregeix les peticions provinents de les **Vistes** cap el **Model** adequat tenint en compte el tipus de petició realitzada i rediregeix cap a les **Vista** adequada la resposta obtinguda des del **Model**.
- Com que en un fitxer de **Model** normalment té una classe amb mètodes per fer INSERT, SELECT, etc.. dins d'una taula, el **Controlador** s'encarrega de seleccionar el mètode correcte en funció de la petició rebuda des de la **Vista**.
- Un **Controlador** accepta la recepció dels mètodes HTTP GET, POST, PUT o DELETE i redirecciona les peticions cap el mètode correcte dins de la classe adequada per gestionar la petició.
- Mai s'encarrega d'interactuar directament amb les taules de les bases de dades i per tant, mai s'encarrega de fer INSERT, SELECT, UPDATE, DELETE sobre les taules.

2- Laravel:

- Defineix una classe abstracta anomenada **Controller** a partir de la qual, per herència, es poden crear noves classes de tipus **Controller**.
- Normalment crearem una classe de tipus **Controller** associada a una classe de tipus **Model**. Per exemple, per la classe **Treballador** tindrem associada un classe que es podria anomenar per exemple **ControladorTreballador** (de fet, no hi ha una convenció per el noms dels controladors).
- Normalment, hi ha un fitxer PHP diferent per cada classe de tipus Controller.

3- Els mètodes típics d'un controlador són:

- **index** → S'utilitzaria normalment per redirigir peticions que demanen veure tots els continguts de la taula cap el mètode adequat del Model i retornar cap a la Vista adequada el resultat obtingut. Aquest mètode ha de cridar al mètode del Model que permet recollir totes els registres de la taula i les envia a la vista que els mostra amb el navegador. També es pot utilitzar per mostrar una pàgina d'inici de l'aplicació.
- **show** → S'utilitzarà normalment per redirigir peticions que demanen veure el continguts de només un registre de la taula cap el mètode adequat del Model i retornar cap a la Vista adequada el resultat obtingut. Aquest mètode ha de cridar al mètode del Model que permet recollir un registre de la taula i les envia a la vista que els mostra amb el navegador.
- **create** → S'utilitzar per demanar i mostrar el formulari necessari per afegir les dades d'un nou recurs (o sigui, un nou registre en la taula) que es vol crear. No crea el recurs, només es crida per mostrar el formulari.
- **store** → S'utilitza per crear i emmagatzemar el recurs amb les dades proporcionades des del formulari mostrat amb **create**. És el mètode que crea el nou recurs (o sigui, un registre en la taula). Aquest mètode ha de cridar al mètode del Model que crea el recurs.
- **edit** → S'utilitzar per demanar i mostrar el formulari necessari per **actualitzar** les dades d'un recurs (o sigui, un nou registre en la taula) existent. No actualitza el recurs, només es crida per mostrar el formulari.
- **update** → S'utilitza per **actualitzar** el recurs amb les dades proporcionades des del formulari mostrat amb **edit**. És el mètode que **actualitza** el recurs (o sigui, un registre en la taula). Aquest mètode ha de cridar al mètode del Model que **actualitza** el recurs.
- **destroy** → S'utilitza per esborrar un recurs (o sigui, un registre en la taula).

- Els mètodes del controlador estan associats als següents mètodes HTTP:
  - GET/HEAD → index
  - GET/HEAD/{url relativa addicional} → show, edit, create (recorda que create no crea registres)
  - POST → store
  - PUT (i PATCH) → update
  - DELETE → destroy

4- Ara crearem el controlador necessari per interactuar amb la classe del model **Treballador** i amb les vistes que farem més endavant. Dins del projecte **empresa**, executa aquesta ordre per crear el controlador:

```
php artisan make:controller ControladorTreballador --resource
```

i comprova que es crea el fitxer **app/Http/Controllers/ControladorTreballador.php** dins del qual trobaràs la classe **ControladorTreballador** amb els mètodes **index**, **create**, **store**, **show**, **edit**, **update** i **destroy** ja preparats per ser desenvolupats.

5- A continuació, registrarem la ruta que apunta a la classe **ControladorTreballadors** dins de **routes/web.php** afegint les línies:

- Al final de la secció dels **use**:

  ```
  use App\Http\Controllers\ControladorTreballador;
  ```

  per assegurar-nos que es pot accedir a la classe **ControladorTreballador** des de **web.php**. Aquesta serà ara la línia **5**.

- A la secció de les **Route** afegeix:

  ```
  Route::resource('trebs', ControladorTreballador::class);
  ```

  a on **trebs** serà la ruta base necessària per arribar al controlador i els seus mètodes. És a dir, per arribar a un mètode de la classe **ControladorTreballador** s'utilitzarà la ruta base **http://<ip_equip>:<port>/trebs/**.

6- Ara indicarem el **Model** que volem utilitzar dins de **ControladorTreballadors.php**. A la secció dels **use** afegirem:

```
use App\Models\Treballador;
```

7- A continuació, podem començar a desenvolupar els mètodes del controlador. Per exemple, pel mètode **index** de la classe **ControladorTreballador** només ens cal el següent:

```
public function index()
{
	$dades_treballadors = Treballador::all();
	return view('llista', compact('dades_treballadors'));
	// Recollirà totes les entrades de la taula treballadors i les desarà dins d'una
	//variable de nom $dades_treballadors
	//Cridara a la vista llista.blade.php que es trobarà a resouces/views per mostrar
	//les dades dels treballadors
	//The compact() function creates an array from variables and their values.
}
```

Imaginem que l'equip servidor tingués l'adreça IP **192.168.1.40** i el servidor web escoltés pel port **8000**. En aquest cas, el mètode **index** de de la classe **ControladorTreballador** s'executaria quan féssim una petició amb el mètode **GET** a **http://192.168.1.38:8000/trebs**.

8- D'acord amb el codi del mètode **index.php** del punt anterior, ara hem de crear un fitxer de vistes de nom **llista.blade.php** dins de **resource/views**. A més a més, farem que aquesta vista tingui el següent codi per visualitzar totes les dades de tots els usuari de la taula **treballadors**:

```
@extends('disseny')
@section('content')
<h1>Llista d'empleats</h1>
<div class="mt-5">
  <table class="table">
    <thead>
        <tr class="table-primary">
          <td>tid</td>
          <td>Nom</td>
          <td>Cognoms</td>
          <td>NIF</td>
          <td>Data de naixement</td>
          <td>Sexe</td>
          <td>Adreça</td>
          <td>Telèfon fixe</td>
          <td>Telèfon mòbil</td>
          <td>Email</td>
          <td>Fotografia</td>
          <td>Treball a distància</td>
          <td>Tipus de contracte</td>
          <td>Data de contractació</td>
          <td>Categoria</td>
          <td>Nom de la feina</td>
          <td>Sou</td>
        </tr>
    </thead>
    <tbody>
        @foreach($dades_treballadors as $treb)
        <tr>
            <td>{{$treb->tid}}</td>
            <td>{{$treb->nom}}</td>
            <td>{{$treb->cognoms}}</td>
            <td>{{$treb->nif}}</td>
            <td>{{$treb->data_naixement}}</td>
            <td>{{$treb->sexe}}</td>
            <td>{{$treb->adressa}}</td>
            <td>{{$treb->tlf_fixe}}</td>
            <td>{{$treb->tlf_mobil}}</td>
            <td>{{$treb->email}}</td>
            <td>{{$treb->fotografia}}</td>
            <td>{{$treb->treball_distancia}}</td>
            <td>{{$treb->tipus_contracte}}</td>
            <td>{{$treb->data_contractacio}}</td>
            <td>{{$treb->categoria}}</td>
            <td>{{$treb->nom_feina}}</td>
            <td>{{$treb->sou}}</td>
        </tr>
        @endforeach
    </tbody>
  </table>
<div>
@endsection
```

i com que aquest codi utilitza el contingut de **disseny.blade.php** com es veu a la primera línia, en el següent punt veurem com s'ha de crear i el seu contingut.

9- Crea el fitxer **disseny.blade.php** dins de **resources/views** i fes que el seu codi sigui el següent:

```
<!DOCTYPE html>
```

```
<html>
   <head>
      <meta charset="utf-8">
      <meta name="viewport" content="width=device-width, initial-scale=1">
      <meta http-equiv="X-UA-Compatible" content="ie=edge">
      <title>Projecte d'accés a BD MySQL amb Laravel 10</title>
      <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">
   </head>
   <body>
     <div class="container-fluid">
         @yield('content')
     </div>

     <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"></script>
     <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"
       type="text/js"></script>
   </body>
</html>
```

10- Ara posa en marxa el servidor web intern de laravel executant des d'**empresa**:

php artisan serve --host=0.0.0.0 --port=8000

i estableix una connexió des de la màquina física a l'adreça URL **http://<ip_màquina_virtual>:8000/trebs** (a on has de canviar **<ip_màquina_virtual>** per la seva adreça IP real). Comprova que el resultat es aquest:

Llista d'empleats

tid	Nom	Cognoms	NIF	Data de naixement	Sexe	Adreça	Telèfon fixe	Telèfon mòbil	Email	Fotografia	Treball a distància	Tipus de contracte	Data de contractació	Categoria	Nom de la feina	Sou
1	Jaume	Pons	46108810Q	2000-03-15	d	Carrer de València, 690, 1-2	934567890	666339900	jpons@gmail.com		1	temporal	2018-03-15	3	desenvolupador junior	1404
2	Anna	Puig	47108810P	2000-03-15	d	Carrer de València, 680, 2-2	934567899	666339911	apuig@gmail.com		0	temporal	2021-03-15	2	desenvolupador senior	1704