PAC2. Desenv. back-end PHP

Josep Vicent Monjo Agut

1 Instal · lació de CodeIgniter 4

He instal · lat CodeIgniter 4 mitjançant composer

```
composer create-project codeigniter4/appstarter project-root
```

El procés d'instal · lació ha sigut prou senzill però menys intuïtiu que amb Wordpress ja que no hi ha una GUI d'instal · lació.

2 Base de dades

He creat una base de dades amb phpmyadmin i dins d'ella he creat les següents taules:

- categories: id, title
- news: id, title, author, date, content, image, category_id

He definit categoy_id com a foreign key de categories.id a la base de dades usant la pestanya relacions de phpmyadmin.

[TODO] parlar de la relació creada mitjançant codeigniter

3 Desenvolupament del backend

Primer he editat el fitxer app/config/Database.php amb les credencials de la base de dades.

Després he creat els models per a News i per a Categories.

A continuació he creat el controlador per llistar les news dins de app/Controllers i he afegit la ruta al fitxer app/config/Routes.php

En el controlador he creat una funció per gestionar les diferents respostes retornant el codi corresponent i les dades o un missatge d'error.

```
private function genericResponse($data, $msg, $code){

        if ($code == 200) {
            return $this->respond(array(
```

```
                'data' => $data,
                'code' => $code
            ));
        } else {
            return $this->respond(array(
                'msg' => $msg,
                'code' => $code
            ));
        }
    }
```

També he creat les diferents funcions que necessitarem per fer CRUD:

- index()
- show()
- create()
- update()
- delete()

He creat algunes validacions bàsiques a app/Config/Validation.php per a l'operació de create()

```
public $news =[
        'title' => 'required|min_length[3]|max_length[255]',
        'description' => 'min_length[3]|max_length[5000]'
    ];
```

3.1 Grocery CRUD

Per a instal · lat Grocery CRUD he seguit el *vídeo explicatiu de l'autor* i després he adaptat el controlador i la vista als nostres models (news i categories).

Ell reconama desactivar les rutes automàtiques si et dona error i afegir-les manualment. En el meu cas, m'ha funcionat amb les autorutes en true.

3.2 Paginació de resultats

Per a la part de la API que opté les notícies d'una categoria específica amb paginació opcional he creat estes rutes:

```
$routes->get('api/(:segment)', 'ApiCategory::show/$1');

$routes->get('api/(:segment)/(:any)', 'ApiCategory::show/$1/$2');
```

i he creat aquesta funció dins del controlador:

```
public function show($category = null, $page = null)
    {
        $db      = \Config\Database::connect();
```

```
        // obtenim la id de la categoria
        $categories = $db->table('categories');

        $categories->where('title', $category);

        $queryCategory   = $categories->get();

        $catResult = $queryCategory->getResult();

        if (!$catResult) {
            return $this->genericResponse(null, "Category doesn't exist", 404);
        }

        $id = $catResult[0]->id;


        // obtenim les notícies amb eixa id de categoria
        $news = $db->table('news');

        $news->where('category_id', $id);

        if ($page) {
            $news->limit(10, ($page - 1) * 10);
        }

        $queryNews   = $news->get();

        $newsResult = $queryNews->getResult();


        return $this->genericResponse($newsResult, null, 200);
    }
```

3.3 Frontend

Per a la part de frontend he creat dues pàgines la Home i la News-details.

Per a cadascuna d'aquestes he creat un fitxer a Controller i un fitxer a views.
En el cas de la home he optat per renderitzar usant parser.

```
public function index()
    {

        $newsModel = new NewsModel();
        $newsArray = $newsModel->findAll();
        $data['baseurl'] = getenv('app.baseURL');
        $data['news'] = $newsArray;
```

```
        $parser = \Config\Services::parser();


        return $parser->setData($data)
             ->render('home');

    }
```

Això m'ha permés usar el següent format a la vista:

```
    <ul>
        {news}
            <li>
                <a href="{baseurl}/news/{id}" class="card">
                <h2>{title}</h2>
                <p>{date}</p>
                </a>
            </li>
        {/news}
    </ul>
```

Per a la vista de detalls he usat el renderitzat que proporciona la funció view:

```
public function show($id)
    {

        $newsModel = new NewsModel();
        $news = $newsModel->find($id);

        $data = $news;
        $data['baseurl'] = getenv('app.baseURL');
        $data['date'] = date("d-m-Y", strtotime($data['date']));

        return view('news-details', $data);

    }
```

Per al correcte funcionament dels enllaços tant en l'entorn de producció com en el de desenvolupament he fet ús de la variable `app.baseURL` del fitxer `.env` que he cridat al controlador usant `getenv('app.baseURL');`

4 URL Pública

4.1 Front end

Home: https://eimtcms.eimt.uoc.edu/~josepmonjo/

4.2 API

- Get news: [GET] https://eimtcms.eimt.uoc.edu/~josepmonjo/api/news
- Post news: [POST] https://eimtcms.eimt.uoc.edu/~josepmonjo/api/news
- Edit news: [PUT, PATCH] https://eimtcms.eimt.uoc.edu/~josepmonjo/api/news/{ID}
- Delete news: [DELETE] https://eimtcms.eimt.uoc.edu/~josepmonjo/api/news/{ID}

5 Anotacions finals

Aquest document l'he generat a partir del README.md del projecte amb el programa *Pandoc* amb la següent instrucció:

```
pandoc -s -N --template=template.latex README.md -o Josep-Vicent-Monjo-Agut-PAC-2.pdf
```