Tester son projet avec PHPUnit

SOMMAIRE

Installation de PHPUnit

PHPUnit est un framework open source de tests unitaires dédié au langage de programmation PHP. Il va nous permettre de tester certaines fonctionnalités de notre site internet.

Pour l'installer nous allons utiliser le gestionnaire de package Composer. Ouvrez donc votre terminal au niveau de votre projet et entrez ces deux lignes de commande :

- composer require --dev symfony/phpunit-bridge
- composer require --dev symfony/browser-kit symfony/css-selector

Tester les routes et les controllers

Tous d'abord nous allons tester le bon fonctionnement de nos controllers en vérifiant que toutes nos routes mènent bien à une page.

Pour cela, dans le dossier "Test", créez un dossier "Controllers" puis créez un fichier "nomEntitéControllerTest".

```php
<?php
// tests/Controller/ProductsControllerTest.php
namespace App\Tests\Controller;

use Symfony\Bundle\FrameworkBundle\Test\WebTestCase;

class ProductControllerTest extends WebTestCase
{
    public function testNewProducts()
    {
        $client = static::createClient();

        $client->request('GET', '/products/new');

        $this->assertEquals(200, $client->getResponse()->getStatusCode());
    }
}
```

Ici nous vérifions que la page de création des produits existe.

Une fois notre fichier paramétré, nous devons lancer le test. Pour cela entrez la ligne de commande suivante:

- php bin/phpunit

En fonction du résultat de celle-ci, vous saurez si le test a fonctionné.

Tester les formulaires

Les formulaires sont souvent primordiaux dans le bon fonctionnement de vos projets. C'est pour cela qu'il est très important de veiller à ce que ceux-ci fonctionnent correctement et qu'ils ne présentent pas de failles de sécurité. Ils donc les tester.

Pour ce faire, dans le dossier "Tests", nous allons créer un dossier "Forms" dans lequel nous allons créez un fichier "nomEntitéFormTest".

```php
<?php
// tests/Form/ProductsFormTest.php
namespace App\Tests\Form;

use App\Entity\Products;
use Symfony\Bundle\FrameworkBundle\Test\KernelTestCase;



class ProductsFormTest extends KernelTestCase{

    public function testNewCategory(){
        $products=(new Products())
        ->setNameProduct('77')
        ->setDescriptionProduct('77')
        ->setPriceProduct('77')
        ->setStockProduct('77');

        self::bootKernel();
        $error = self::$container->get('validator')->validate($products);
        $this->assertCount(0,$error);
    }
}
```

Comme vous pouvez le constater, j'ai volontairement développé mon test avec des données farfelues afin de savoir si cela peut fonctionner.

Je lance mon test et voici les résultats que j'obtiens :

```
λ php bin/phpunit
PHPUnit 8.5.14 by Sebastian Bergmann and contributors.

Testing Project Test Suite
....                                                                4 / 4 (100%)

Time: 589 ms, Memory: 32.00 MB

OK (4 tests, 4 assertions)
```

Mon test est réussi malgré mes données complètement à côté de ce que l'on attend dans ce formulaire.
Nous allons donc devoir modifier nos formulaires grâce aux contraintes de validation.

Les contraintes de validation

Pour pouvoir utiliser ces contraintes de validation dans vos formulaires, il faut tout d'abord installer un nouveau package à votre projet Symfony grâce Composer.

- composer require symfony/validator doctrine/annotations

Ensuite je vous invite à aller regarder la documentation de Symfony : *https://symfony.com/doc/current/reference/constraints.html*