CERTIF ACADEMY

Laravel testing

Comprendre la mise en oeuvre des tests sous Laravel

CERTIF ACADEMY

Compétence demandée :

Comprendre la mise en oeuvre de tests sous Laravel

CERTIF ACADEMY

1. Base de données de tests
2. Mocker les objets sous Laravel
3. Les tests
4. Les unit tests
5. Les feature tests

CERTIF ACADEMY

ATELIER

Prendre connaissance du projet

CERTIF ACADEMY

Base de données de tests

CERTIF ACADEMY

Afin de tester correctement notre application et de ne pas avoir de données de tests, c'est à dire de mauvaises données dans notre base de données de production. Nous allons émuler une base de données de test.

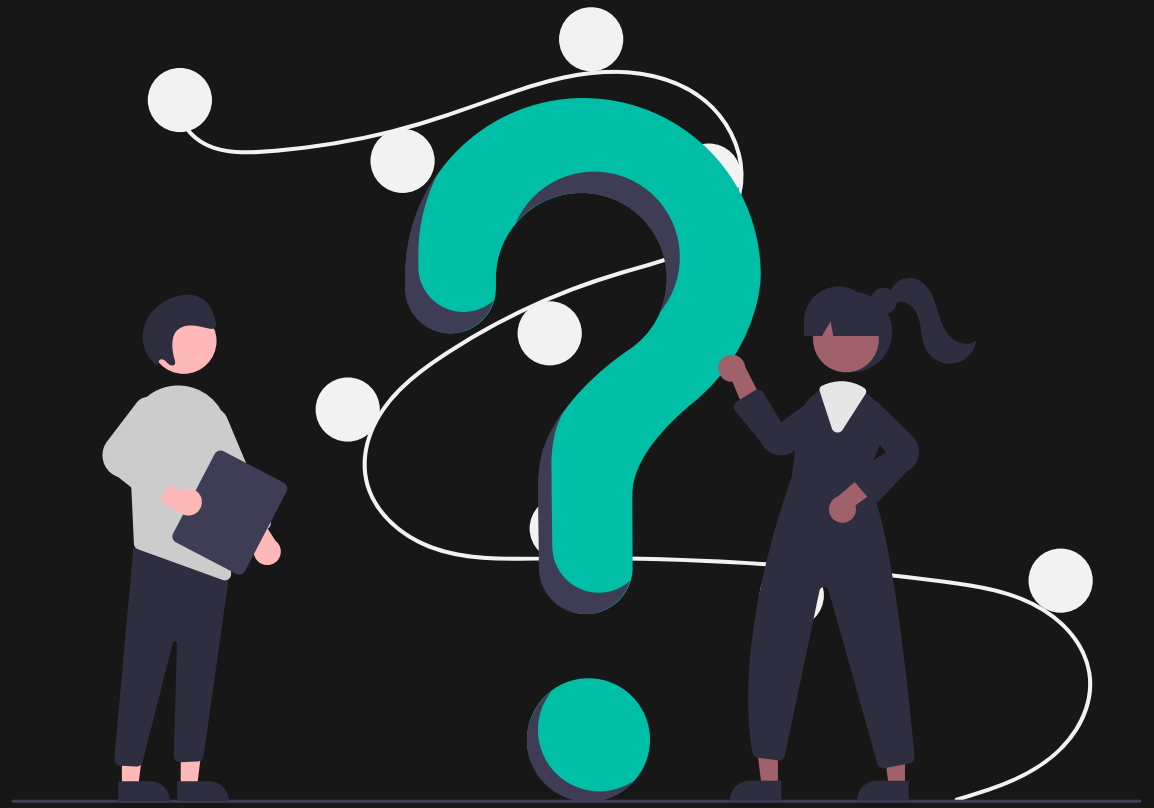
CERTIF ACADEMY

Afin de créer une base de données des tests, nous allons ajouter un nouveau fichier .env.testing.

Les variables d'environnement de ce fichier viendront supplanter les variables d'environnement, du fichier d'environnement initial.

Il sera ainsi possible de définir une nouvelle base de données spécifiquement pour les tests.

CERTIF ACADEMY



CERTIF ACADEMY

ATELIER

Créer une base de données de test pour le projet

CERTIF
ACADEMY

CERTIF ACADEMY

1. Base de données de tests
2. Mocker les objets sous Laravel
3. Les tests
4. Les unit tests
5. Les feature tests

CERTIF ACADEMY

Mocker les objets sous Laravel

CERTIF ACADEMY

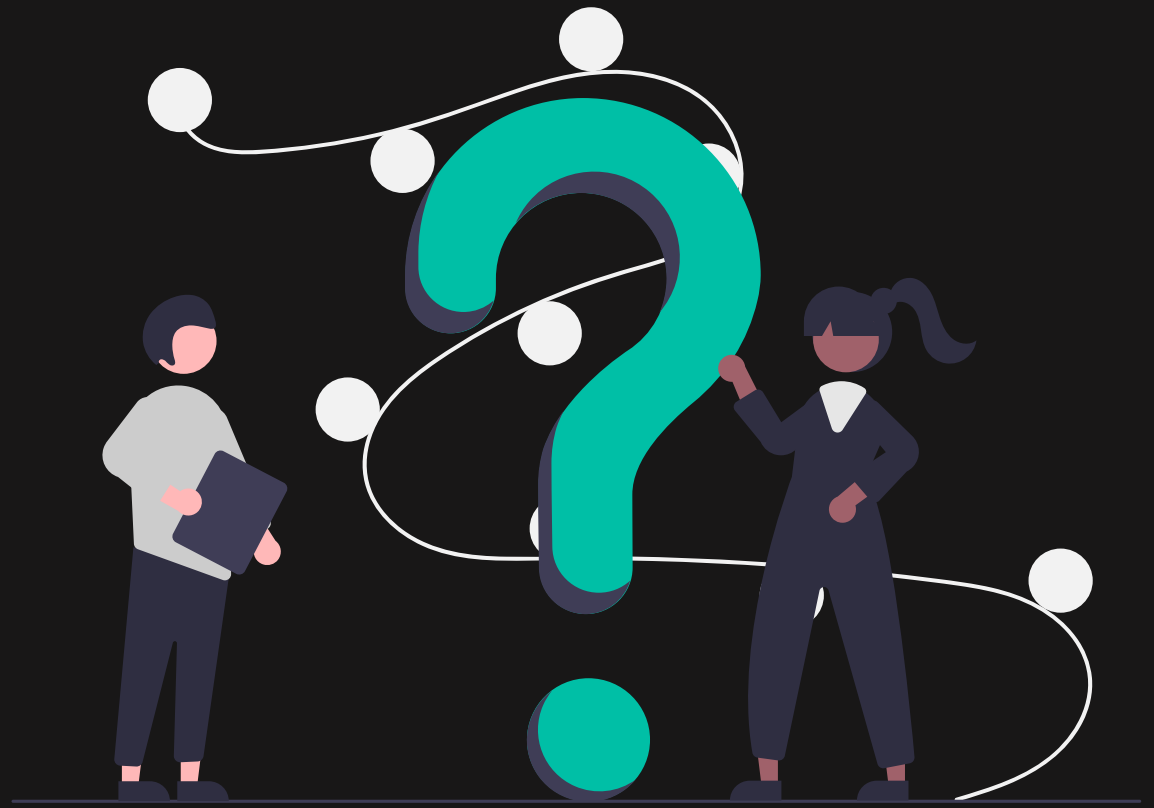
Mockery permet de moquer les objets utilisés dans laravel.

Il sera également possible de moquer les objets dans le container de service de laravel.

CERTIF ACADEMY

```php
protected function setUp(): void
    {
        parent::setUp();
        $repositoy = Mockery::mock(BookingRepository::class, function (MockInterface $mock) {
            $mock->shouldReceive('findOverlapingBooking')->andReturn([]);
            $mock->shouldReceive('create')->andReturn(new Booking());
        });
        $this->bookingService = new BookingService($repositoy);
    }
```

CERTIF ACADEMY



CERTIF
ACADEMY

CERTIF ACADEMY

1. Base de données de tests
2. Mocker les objets sous Laravel
3. Les tests
4. Les unit tests
5. Les feature tests

CERTIF ACADEMY

Les tests

CERTIF ACADEMY

Dans un projet Laravel, les tests sont généralement stockés dans le répertoire tests. Ce répertoire se trouve à la racine du projet et contient tous les fichiers de test de l'application.

CERTIF ACADEMY

```
php artisan make:test
```

```
php artisan make:test --unit
```

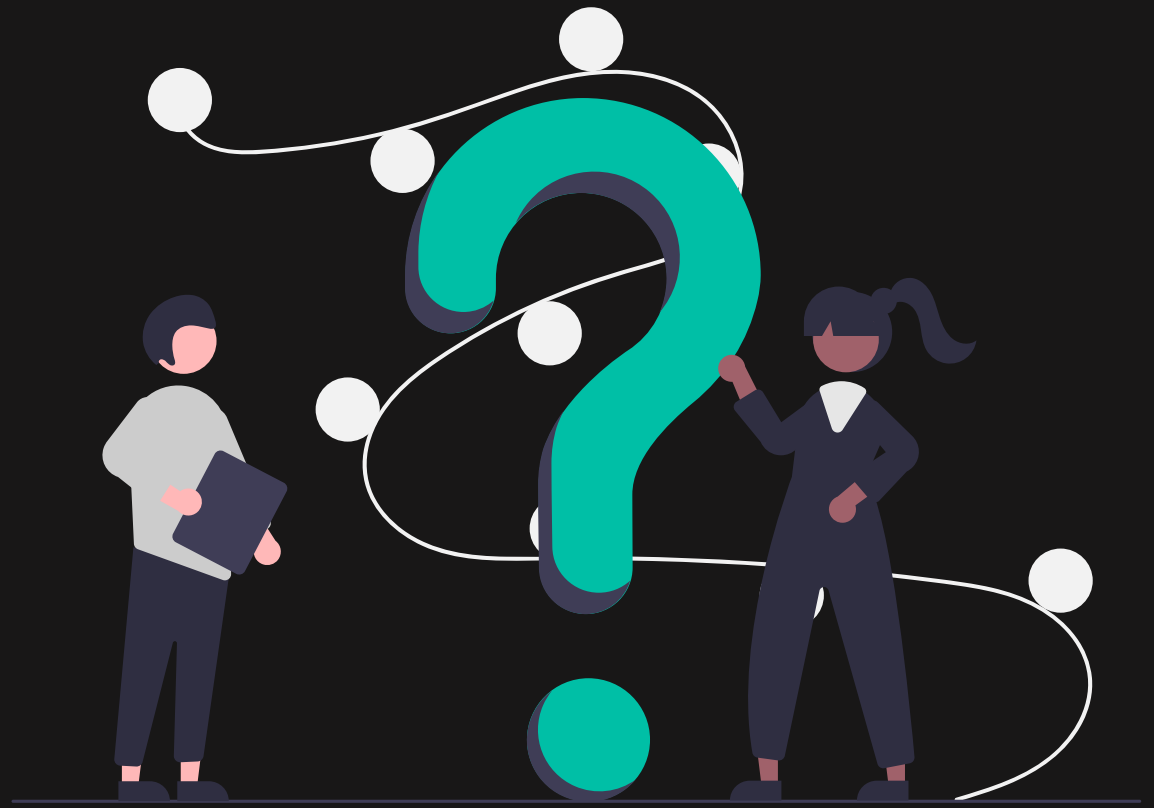
```
php artisan make:feature-test
```

CERTIF ACADEMY

```
php artisan test
```

Cette commande va exécuter tous les tests de votre projet Laravel en utilisant PHPUnit.

CERTIF ACADEMY



CERTIF
ACADEMY

CERTIF ACADEMY

1. Base de données de tests
2. Mocker les objets sous Laravel
3. Les tests
4. Les unit tests
5. Les feature tests

CERTIF ACADEMY

Les unit tests

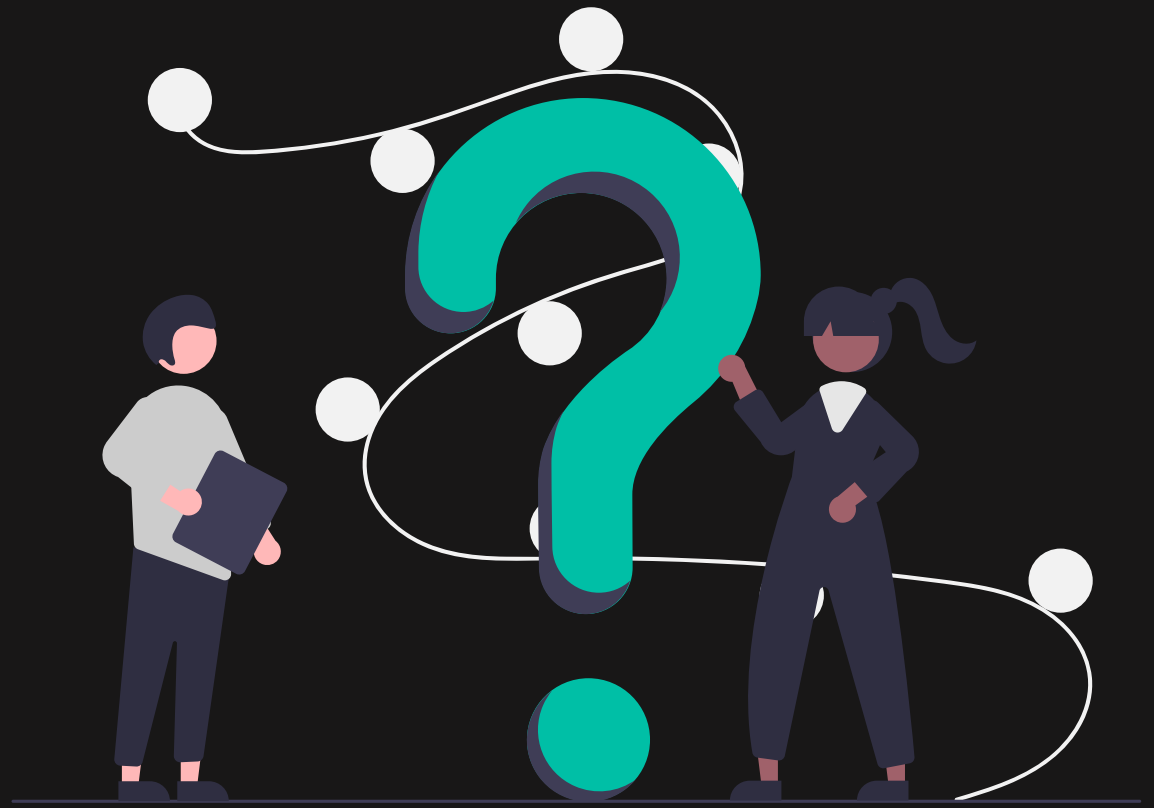
CERTIF ACADEMY

Les tests unitaires dans Laravel reste des tests unitaires comme vu dans la section précédente.

CERTIF ACADEMY

```php
public function test_book_return_success()
    {
        $user = User::create([
            'name' => 'user',
            'email' => 'user@user.com',
            'password' => 'useruser'
        ]);
        $hotel = Hotel::create([
            'name' => 'test',
            'address' => 'test'
        ]);
        $room = Room::create([
            'number' => '493',
            'capacity' => 10,
            'type' => 'suite',
            'hotel_id' => $hotel->id
        ]);
        $booking = $this->bookingService->book($user, $room, new \DateTimeImmutable('2022-12-29'),
new \DateTimeImmutable('2022-12-30'));
        $this->assertNotNull($booking);
    }
```

CERTIF ACADEMY



CERTIF
ACADEMY

CERTIF ACADEMY

1. Base de données de tests
2. Mocker les objets sous Laravel
3. Les tests
4. Les unit tests
5. Les feature tests

CERTIF ACADEMY

Les feature tests

CERTIF ACADEMY

Avec laravel, on a toujours la distinction entre les tests unitaires et les tests d'intégration.

Contrairement aux tests unitaires qui s'appelleront toujours unit test, les tests d'intégration s'appelleront les feature tests. À noter que les futurs tests peuvent également dénoter les tests e2e.

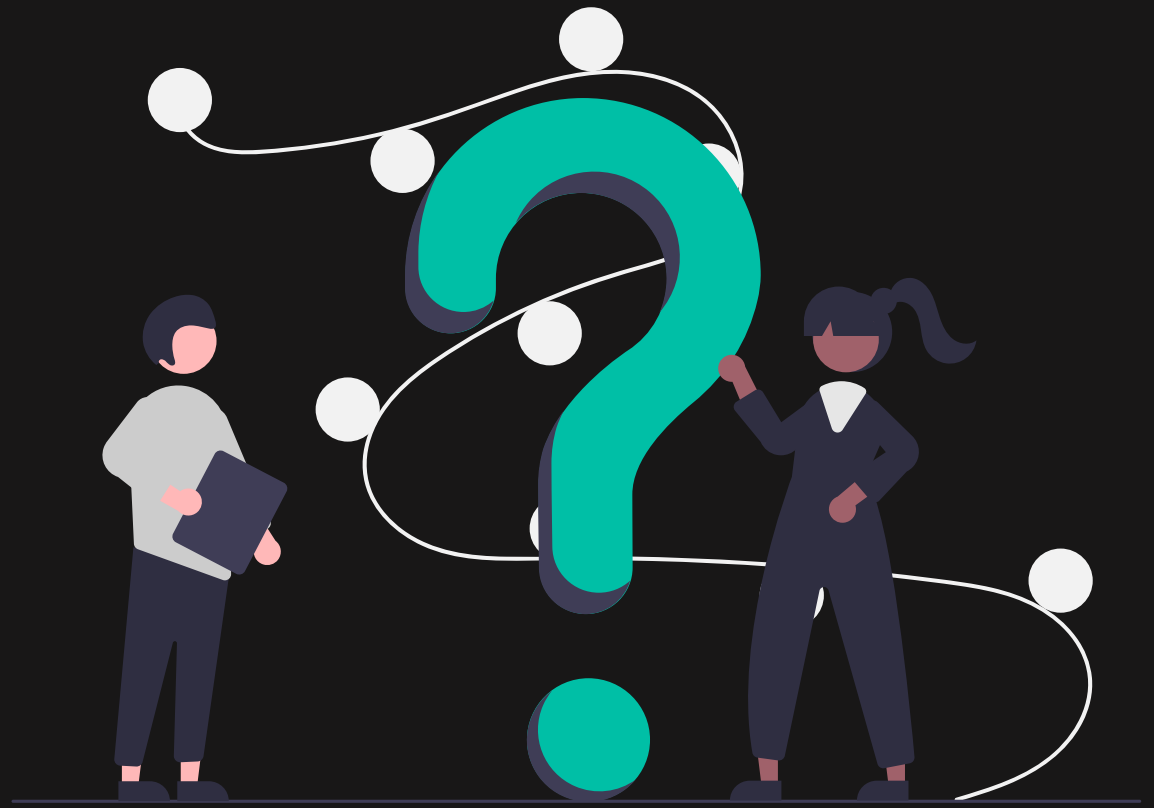
CERTIF ACADEMY

```php
public function test_book_room_return_201()
    {
        $user = User::create([
            'name' => 'user',
            'email' => 'user@user.com',
            'password' => 'useruser'
        ]);
        $hotel = Hotel::create([
            'name' => 'test',
            'address' => 'test'
        ]);
        $room = Room::create([
            'number' => '493',
            'capacity' => 10,
            'type' => 'suite',
            'hotel_id' => $hotel->id
        ]);

        $response = $this->actingAs($user)->post('/api/rooms/' . $room->id . '/bookings', [
            'start_at' => '2022-12-29',
            'end_at' => '2022-12-30'
        ]);


        $response->assertStatus(201);
    }
}
```

CERTIF ACADEMY



CERTIF ACADEMY

ATELIER

Implémenter les tests unitaires