LAPORAN 6

Integratif Programming

Leveling Up Responses

Disusun Oleh :

Nama	: Ichsanul Dwi Prayitno
Nim	: 4.33.20.0.14
Kelas	: TI-4A

PROGRAM STUDI S.TR TEKNOLOGI REKAYASA KOMPUTER

JURUSAN TEKNIK ELEKTRO

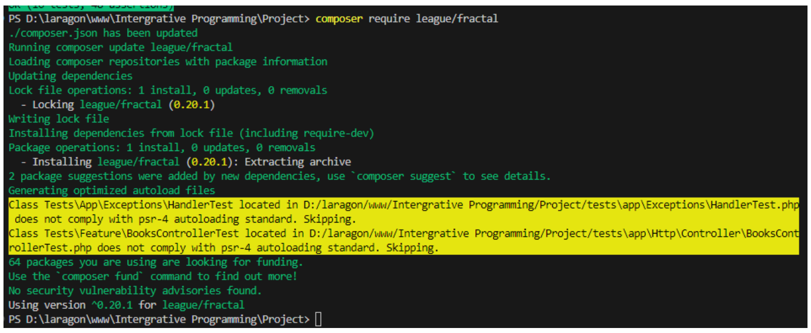
POLITEKNIK NEGERI SEMARANG

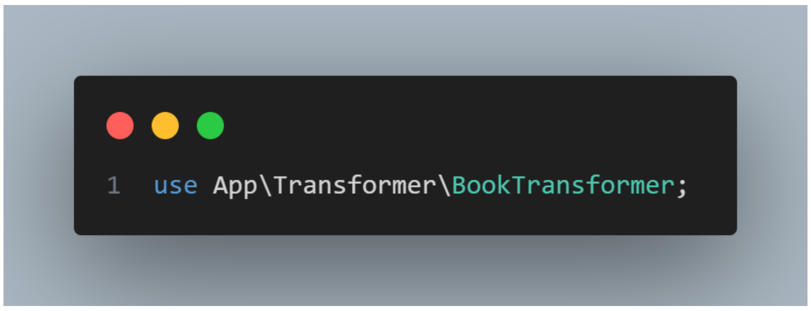
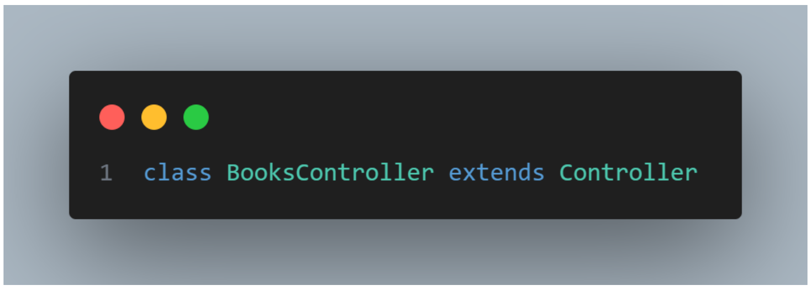
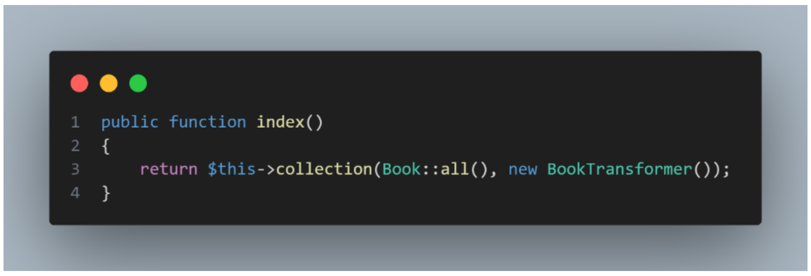
TAHUN 2024

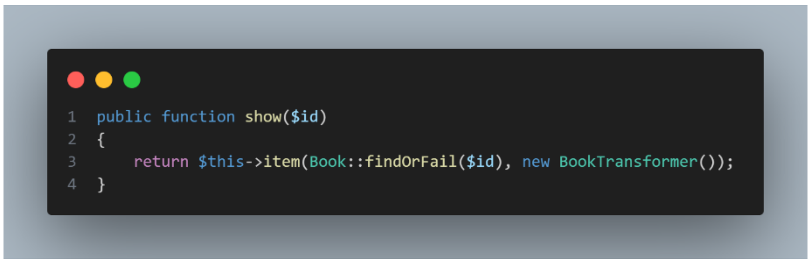
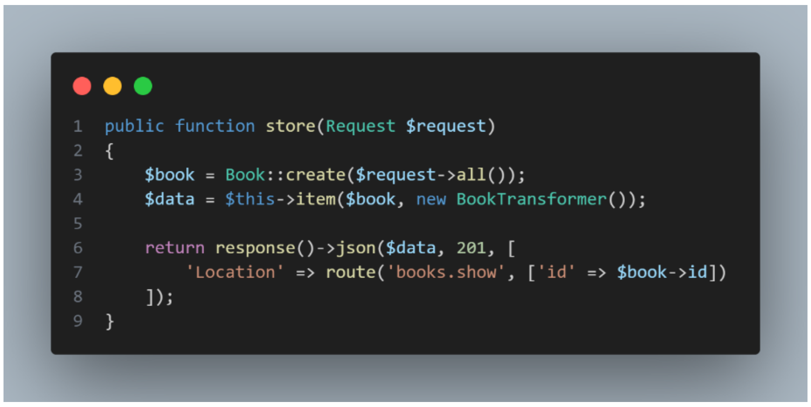
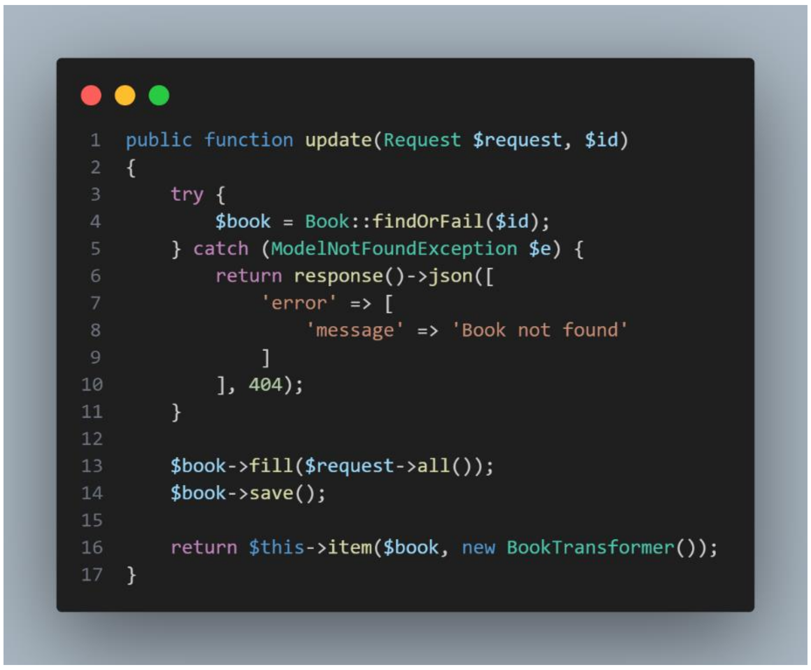
PRAKTEK

I. Lembar Kerja

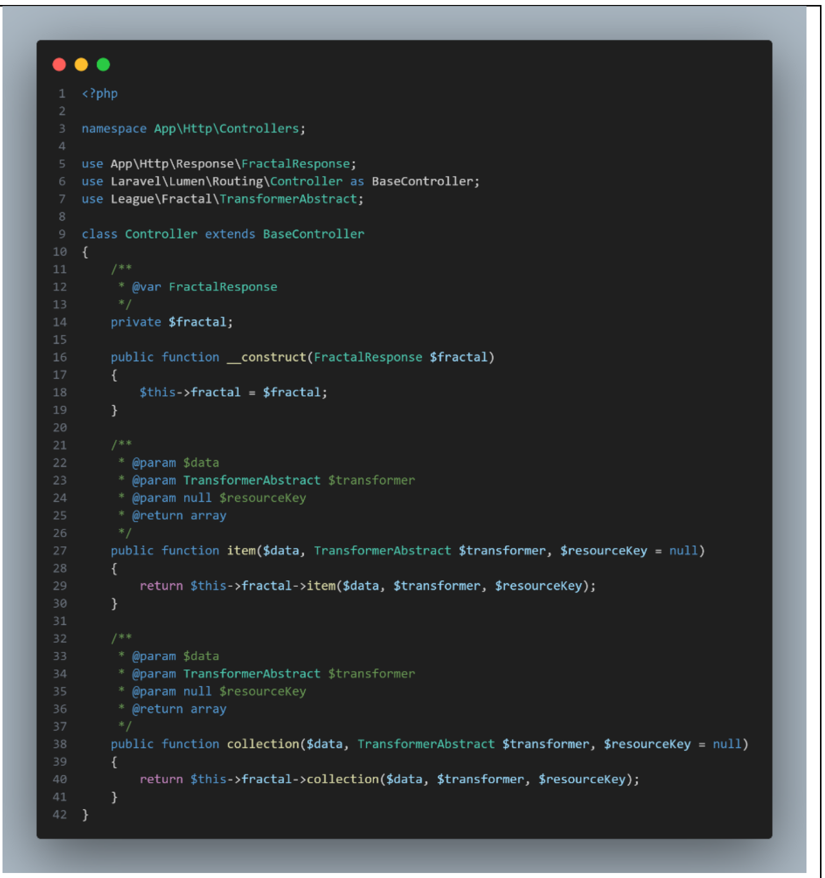
No	Praktik	Hasil Program

1	Installasi The Fractal Response Class	Pada terminal jalankan “composer require league/fractal” untuk installasi library fractal
		
		`PS D:\laragon\www\Intergrative Programming\Project> composer require league/fractal`
		`./composer.json has been updated`
		`Running composer update league/fractal`
		`Loading composer repositories with package information`
		`Updating dependencies`
		`Lock file operations: 1 install, 0 updates, 0 removals`
		`  - Locking league/fractal (0.20.1)`
		`Writing lock file`
		`Installing dependencies from lock file (including require-dev)`
		`Package operations: 1 install, 0 updates, 0 removals`
		`  - Installing league/fractal (0.20.1): Extracting archive`
		`2 package suggestions were added by new dependencies, use ``composer suggest`` to see details.`
		`Generating optimized autoload files`
		`Class Tests\App\Exceptions\HandlerTest located in D:/laragon/www/Intergrative Programming/Project/tests\app\Exceptions\HandlerTest.php`
		` does not comply with psr-4 autoloading standard. Skipping.`
		`Class Tests\Feature\BooksControllerTest located in D:/laragon/www/Intergrative Programming/Project/tests\app\Http\Controller\BooksCont`
		`rollerTest.php does not comply with psr-4 autoloading standard. Skipping.`
		`64 packages you are using are looking for funding.`
		`Use the ``composer fund`` command to find out more!`
		`No security vulnerability advisories found.`
		`Using version ^0.20.1 for league/fractal`
		`PS D:\laragon\www\Intergrative Programming\Project>`

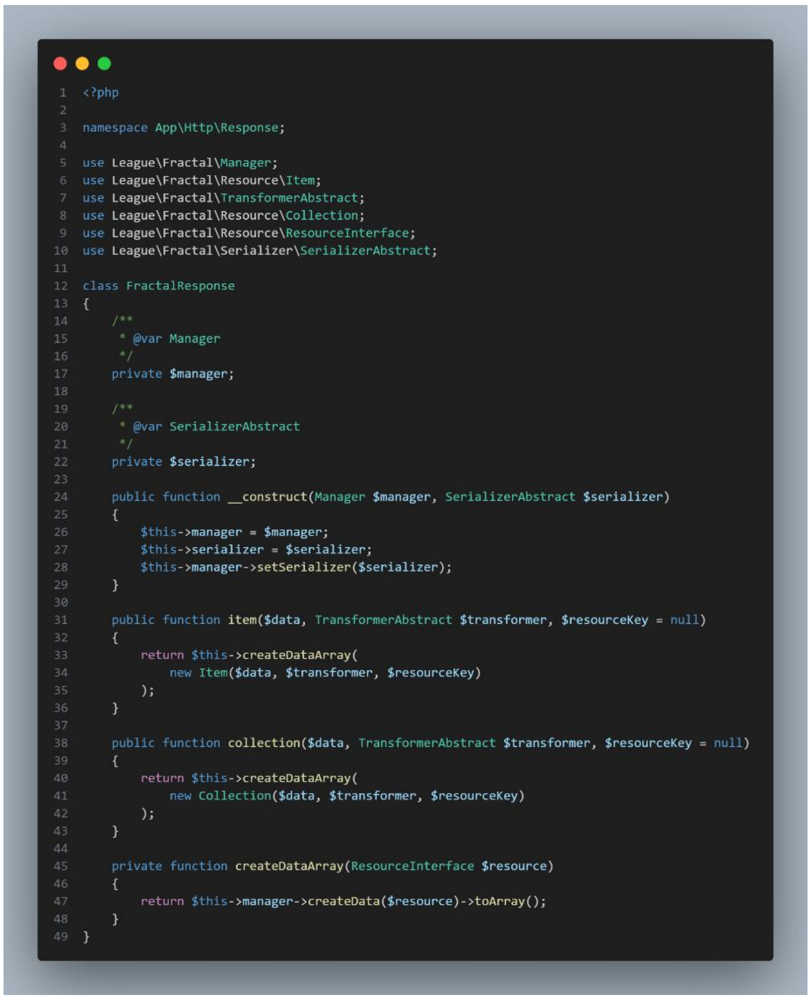
2	Perubahan pada file
	
	Project/app/Http/Controllers/BooksController.php
		Penambahan Include Transformasi
		
		`use App\Transformer\BookTransformer;`
		
		Extend base controller
		
		`class BooksController extends Controller`
		
		Function index
		
		`public function index()`
		`{`
		`    return $this->collection(Book::all(), new BookTransformer());`
		`}`

		Function show
		
		```
		public function show($id)
		{
		    return $this->item(Book::findOrFail($id), new BookTransformer());
		}
		```
		Function store
		
		```
		public function store(Request $request)
		{
		    $book = Book::create($request->all());
		    $data = $this->item($book, new BookTransformer());

		    return response()->json($data, 201, [
		        'Location' => route('books.show', ['id' => $book->id])
		    ]);
		}
		```
		Function update
		
		```
		public function update(Request $request, $id)
		{
		    try {
		        $book = Book::findOrFail($id);
		    } catch (ModelNotFoundException $e) {
		        return response()->json([
		            'error' => [
		                'message' => 'Book not found'
		            ]
		        ], 404);
		    }

		    $book->fill($request->all());
		    $book->save();

		    return $this->item($book, new BookTransformer());
		}
		```

3	Perubahan file pada	Banyak

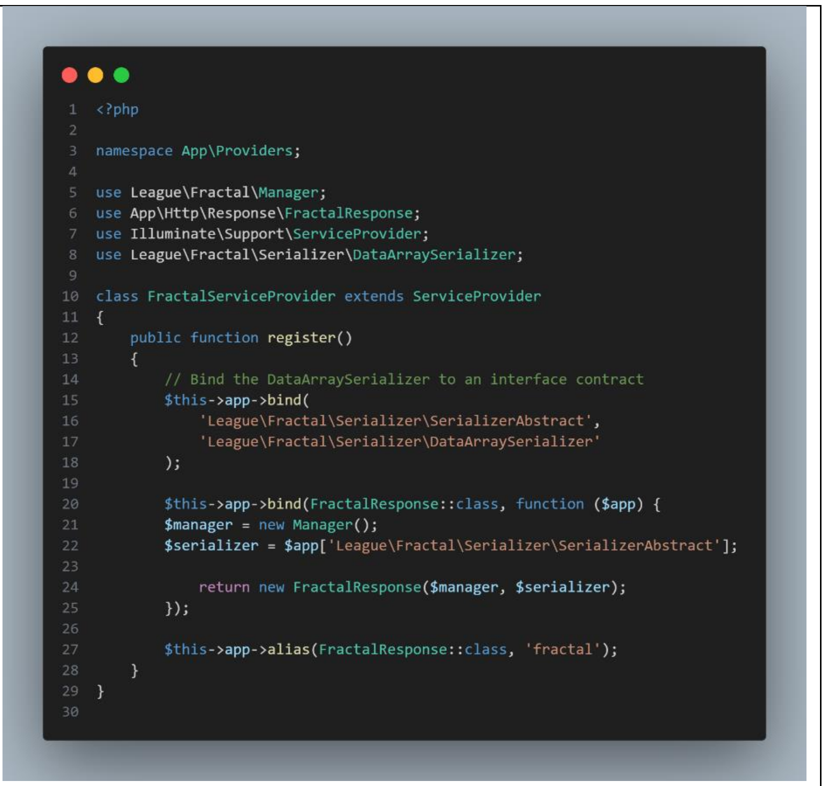
	Project/app/Http/Controllers/Controller.php	

```php
<?php

namespace App\Http\Controllers;

use App\Http\Response\FractalResponse;
use Laravel\Lumen\Routing\Controller as BaseController;
use League\Fractal\TransformerAbstract;

class Controller extends BaseController
{
    /**
     * @var FractalResponse
     */
    private $fractal;

    public function __construct(FractalResponse $fractal)
    {
        $this->fractal = $fractal;
    }

    /**
     * @param $data
     * @param TransformerAbstract $transformer
     * @param null $resourceKey
     * @return array
     */
    public function item($data, TransformerAbstract $transformer, $resourceKey = null)
    {
        return $this->fractal->item($data, $transformer, $resourceKey);
    }

    /**
     * @param $data
     * @param TransformerAbstract $transformer
     * @param null $resourceKey
     * @return array
     */
    public function collection($data, TransformerAbstract $transformer, $resourceKey = null)
    {
        return $this->fractal->collection($data, $transformer, $resourceKey);
    }
}
```

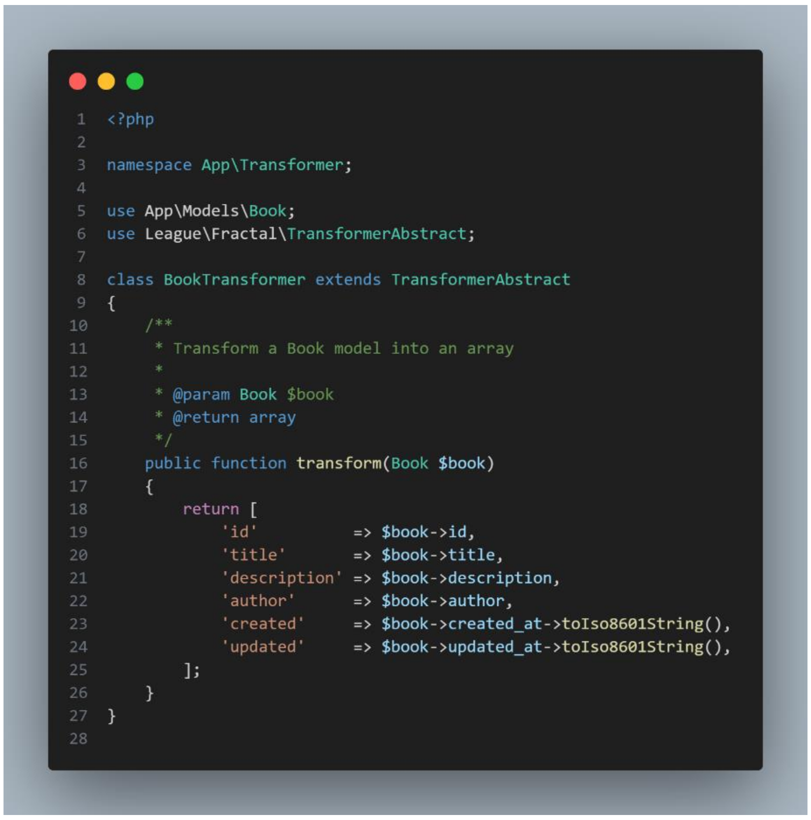
4	Penambahan file Project/app/Http/Response/FractalResponse.php	Dengan isi 

```php
<?php

namespace App\Http\Response;

use League\Fractal\Manager;
use League\Fractal\Resource\Item;
use League\Fractal\TransformerAbstract;
use League\Fractal\Resource\Collection;
use League\Fractal\Resource\ResourceInterface;
use League\Fractal\Serializer\SerializerAbstract;

class FractalResponse
{
    /**
     * @var Manager
     */
    private $manager;

    /**
     * @var SerializerAbstract
     */
    private $serializer;

    public function __construct(Manager $manager, SerializerAbstract $serializer)
    {
        $this->manager = $manager;
        $this->serializer = $serializer;
        $this->manager->setSerializer($serializer);
    }

    public function item($data, TransformerAbstract $transformer, $resourceKey = null)
    {
        return $this->createDataArray(
            new Item($data, $transformer, $resourceKey)
        );
    }

    public function collection($data, TransformerAbstract $transformer, $resourceKey = null)
    {
        return $this->createDataArray(
            new Collection($data, $transformer, $resourceKey)
        );
    }

    private function createDataArray(ResourceInterface $resource)
    {
        return $this->manager->createData($resource)->toArray();
    }
}
```

5	Penambahan file Project/app/Providers/FractalServiceProvider.php	Dengan isi

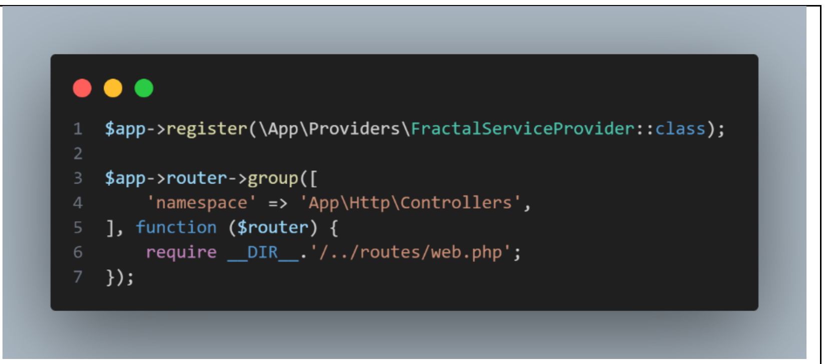
		

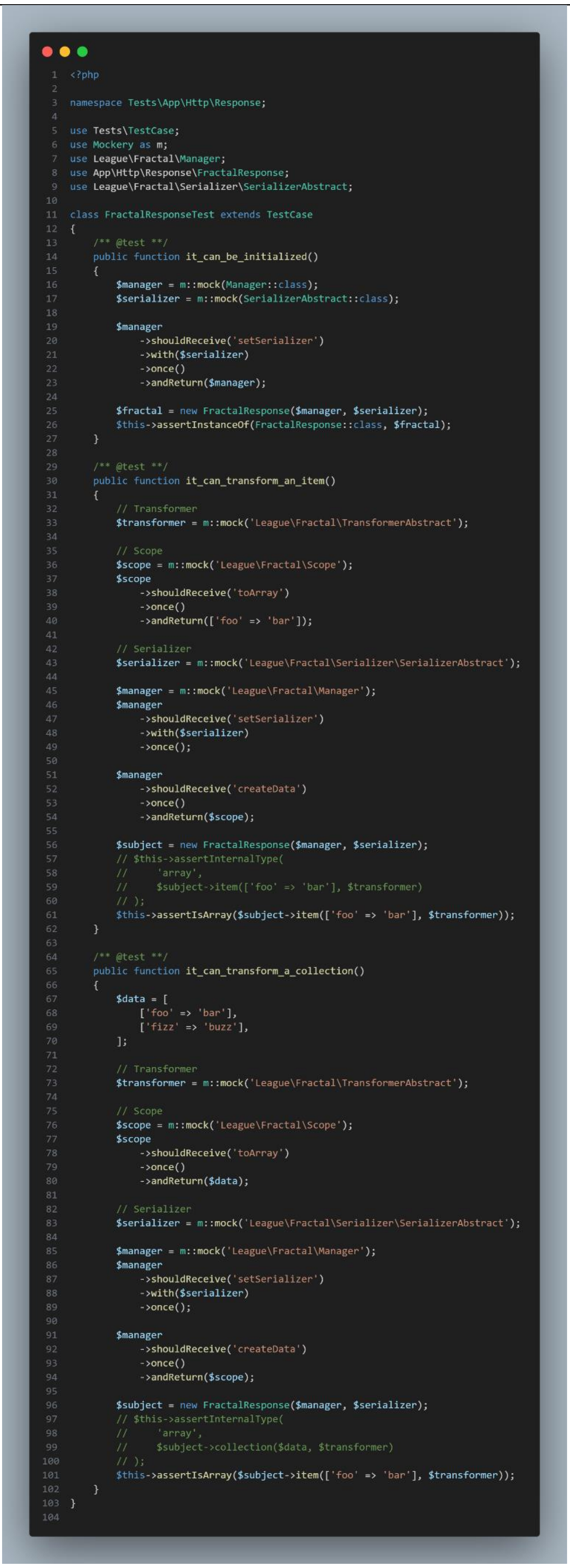
```php
<?php

namespace App\Providers;

use League\Fractal\Manager;
use App\Http\Response\FractalResponse;
use Illuminate\Support\ServiceProvider;
use League\Fractal\Serializer\DataArraySerializer;

class FractalServiceProvider extends ServiceProvider
{
    public function register()
    {
        // Bind the DataArraySerializer to an interface contract
        $this->app->bind(
            'League\Fractal\Serializer\SerializerAbstract',
            'League\Fractal\Serializer\DataArraySerializer'
        );

        $this->app->bind(FractalResponse::class, function ($app) {
        $manager = new Manager();
        $serializer = $app['League\Fractal\Serializer\SerializerAbstract'];

            return new FractalResponse($manager, $serializer);
        });

        $this->app->alias(FractalResponse::class, 'fractal');
    }
}

```

6	Penambahan file Project/app/Transformer/BookTransformer.php	Dengan isi 

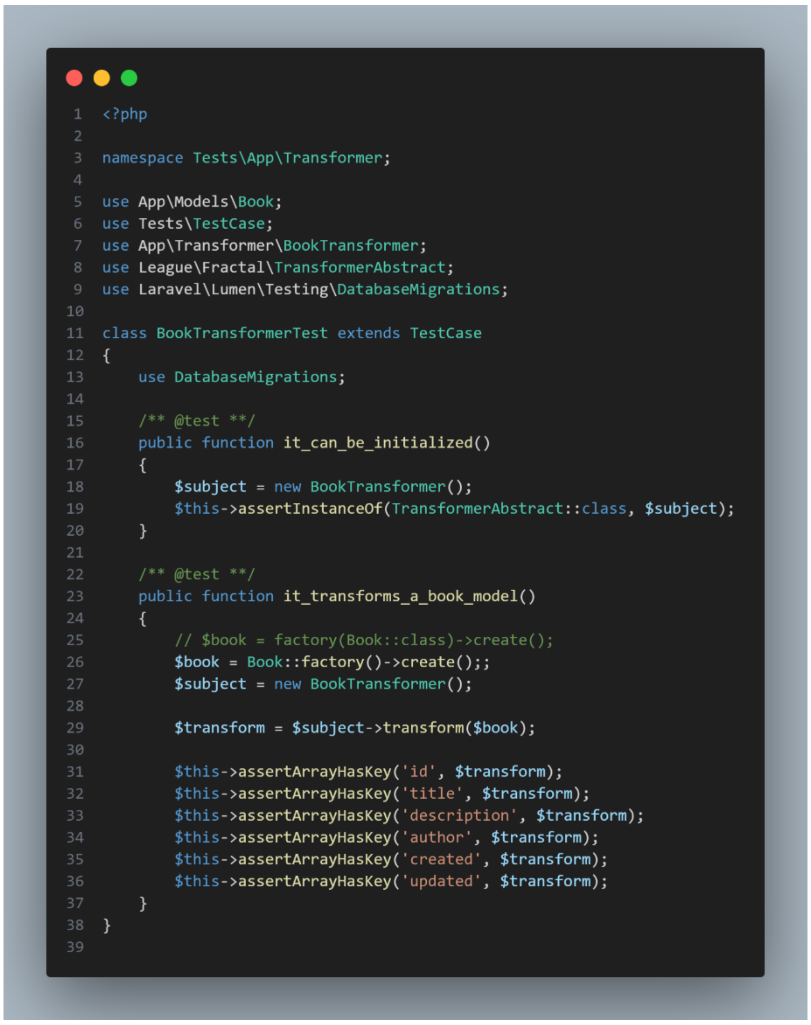
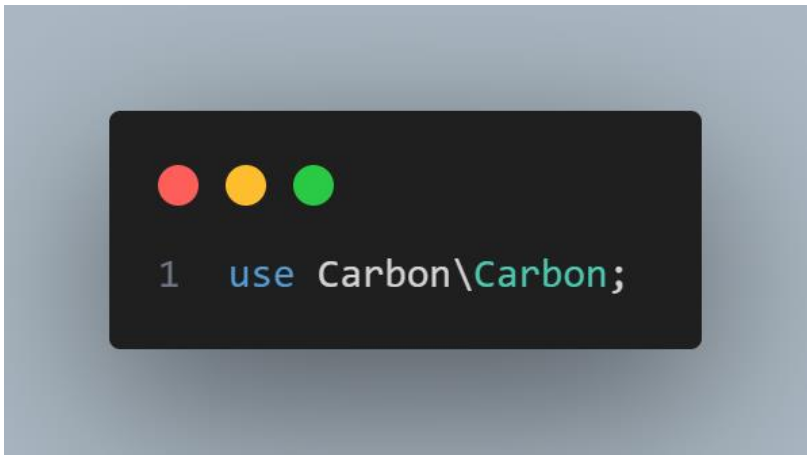
```php
<?php

namespace App\Transformer;

use App\Models\Book;
use League\Fractal\TransformerAbstract;

class BookTransformer extends TransformerAbstract
{
    /**
     * Transform a Book model into an array
     *
     * @param Book $book
     * @return array
     */
    public function transform(Book $book)
    {
        return [
            'id'          => $book->id,
            'title'       => $book->title,
            'description' => $book->description,
            'author'      => $book->author,
            'created'     => $book->created_at->toIso8601String(),
            'updated'     => $book->updated_at->toIso8601String(),
        ];
    }
}

```

7	Perubahan file pada	Tambahkan register sebelum di group

	Project/bootstrap/app.php	 ```php $app->register(\App\Providers\FractalServiceProvider::class); $app->router->group(['namespace' => 'App\Http\Controllers',], function ($router) { require __DIR__.'/../routes/web.php'; }); ```
8	Penambahan file Project/tests/App/Http/Response/FractalResponseTest.php	Dengan isi

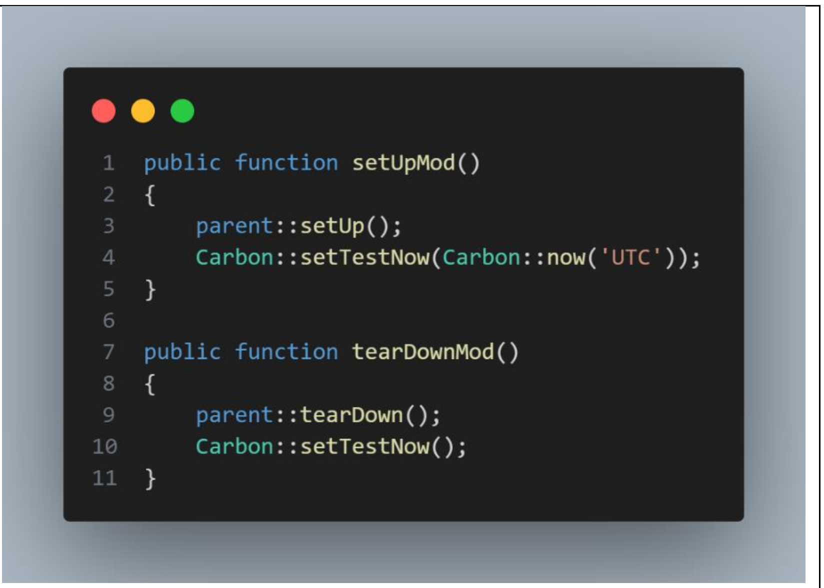


```php
<?php

namespace Tests\App\Http\Response;

use Tests\TestCase;
use Mockery as m;
use League\Fractal\Manager;
use App\Http\Response\FractalResponse;
use League\Fractal\Serializer\SerializerAbstract;

class FractalResponseTest extends TestCase
{
    /** @test **/
    public function it_can_be_initialized()
    {
        $manager = m::mock(Manager::class);
        $serializer = m::mock(SerializerAbstract::class);

        $manager
            ->shouldReceive('setSerializer')
            ->with($serializer)
            ->once()
            ->andReturn($manager);

        $fractal = new FractalResponse($manager, $serializer);
        $this->assertInstanceOf(FractalResponse::class, $fractal);
    }

    /** @test **/
    public function it_can_transform_an_item()
    {
        // Transformer
        $transformer = m::mock('League\Fractal\TransformerAbstract');

        // Scope
        $scope = m::mock('League\Fractal\Scope');
        $scope
            ->shouldReceive('toArray')
            ->once()
            ->andReturn(['foo' => 'bar']);

        // Serializer
        $serializer = m::mock('League\Fractal\Serializer\SerializerAbstract');

        $manager = m::mock('League\Fractal\Manager');
        $manager
            ->shouldReceive('setSerializer')
            ->with($serializer)
            ->once();

        $manager
            ->shouldReceive('createData')
            ->once()
            ->andReturn($scope);

        $subject = new FractalResponse($manager, $serializer);
        // $this->assertInternalType(
        //     'array',
        //     $subject->item(['foo' => 'bar'], $transformer)
        // );
        $this->assertIsArray($subject->item(['foo' => 'bar'], $transformer));
    }

    /** @test **/
    public function it_can_transform_a_collection()
    {
        $data = [
            ['foo' => 'bar'],
            ['fizz' => 'buzz'],
        ];

        // Transformer
        $transformer = m::mock('League\Fractal\TransformerAbstract');

        // Scope
        $scope = m::mock('League\Fractal\Scope');
        $scope
            ->shouldReceive('toArray')
            ->once()
            ->andReturn($data);

        // Serializer
        $serializer = m::mock('League\Fractal\Serializer\SerializerAbstract');

        $manager = m::mock('League\Fractal\Manager');
        $manager
            ->shouldReceive('setSerializer')
            ->with($serializer)
            ->once();

        $manager
            ->shouldReceive('createData')
            ->once()
            ->andReturn($scope);

        $subject = new FractalResponse($manager, $serializer);
        // $this->assertInternalType(
        //     'array',
        //     $subject->collection($data, $transformer)
        // );
        $this->assertIsArray($subject->item(['foo' => 'bar'], $transformer));
    }
}
```

9	Penambahan file Project/tests/App/Transformer/BookTransformerTest.php	Dengan isi 
10	Perubahan file pada Project/tests/app/Http/Controller/BooksControllerTest.php	Import library carbon  Function baru

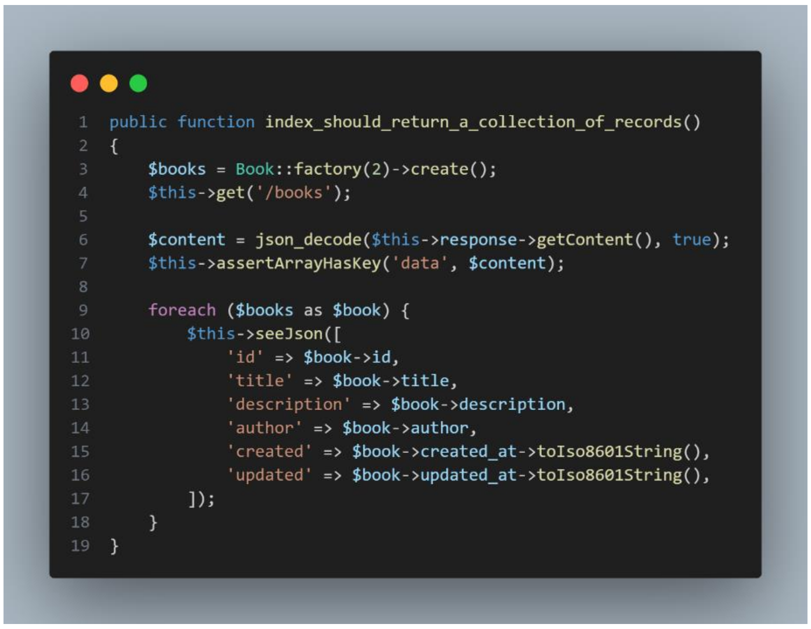
```php
<?php

namespace Tests\App\Transformer;

use App\Models\Book;
use Tests\TestCase;
use App\Transformer\BookTransformer;
use League\Fractal\TransformerAbstract;
use Laravel\Lumen\Testing\DatabaseMigrations;

class BookTransformerTest extends TestCase
{
    use DatabaseMigrations;

    /** @test **/
    public function it_can_be_initialized()
    {
        $subject = new BookTransformer();
        $this->assertInstanceOf(TransformerAbstract::class, $subject);
    }

    /** @test **/
    public function it_transforms_a_book_model()
    {
        // $book = factory(Book::class)->create();
        $book = Book::factory()->create();;
        $subject = new BookTransformer();

        $transform = $subject->transform($book);

        $this->assertArrayHasKey('id', $transform);
        $this->assertArrayHasKey('title', $transform);
        $this->assertArrayHasKey('description', $transform);
        $this->assertArrayHasKey('author', $transform);
        $this->assertArrayHasKey('created', $transform);
        $this->assertArrayHasKey('updated', $transform);
    }
}
```

```php
use Carbon\Carbon;
```



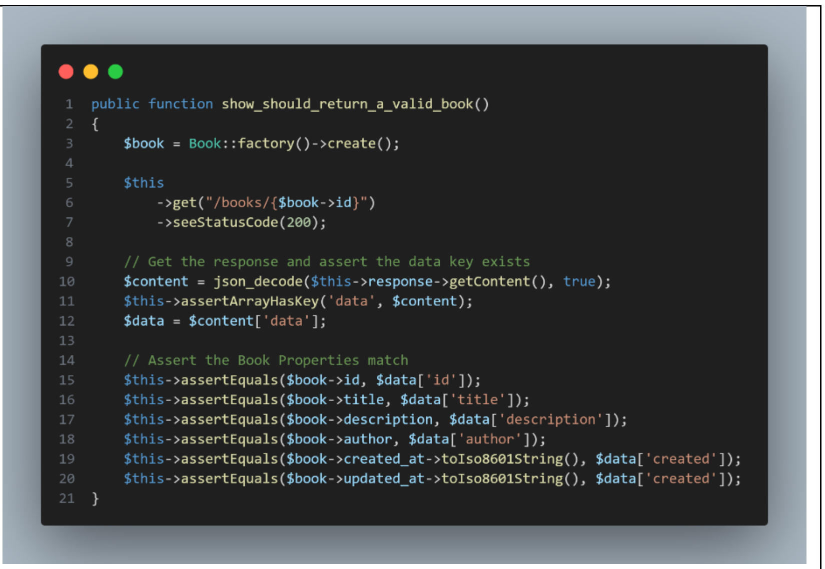
```php
public function setUpMod()
{
    parent::setUp();
    Carbon::setTestNow(Carbon::now('UTC'));
}

public function tearDownMod()
{
    parent::tearDown();
    Carbon::setTestNow();
}
```

Function index_should_return_a_collection_of_records



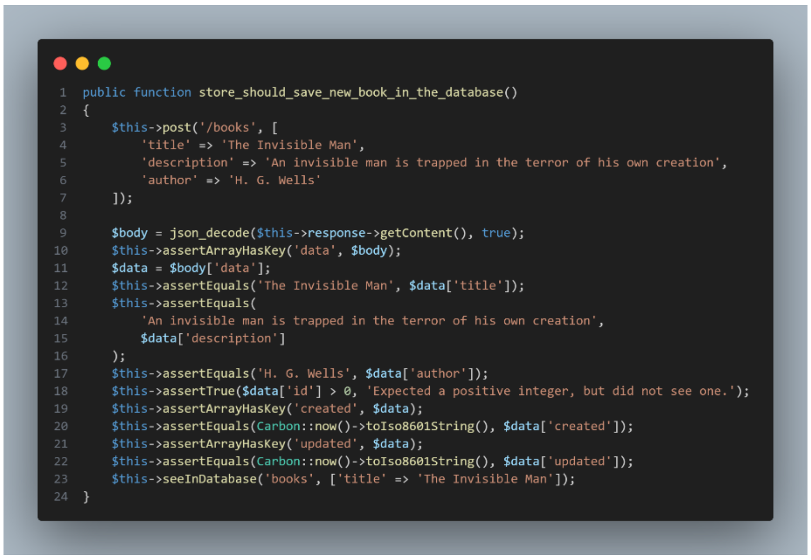
```php
public function index_should_return_a_collection_of_records()
{
    $books = Book::factory(2)->create();
    $this->get('/books');

    $content = json_decode($this->response->getContent(), true);
    $this->assertArrayHasKey('data', $content);

    foreach ($books as $book) {
        $this->seeJson([
            'id' => $book->id,
            'title' => $book->title,
            'description' => $book->description,
            'author' => $book->author,
            'created' => $book->created_at->toIso8601String(),
            'updated' => $book->updated_at->toIso8601String(),
        ]);
    }
}
```

Function show_should_return_a_valid_book



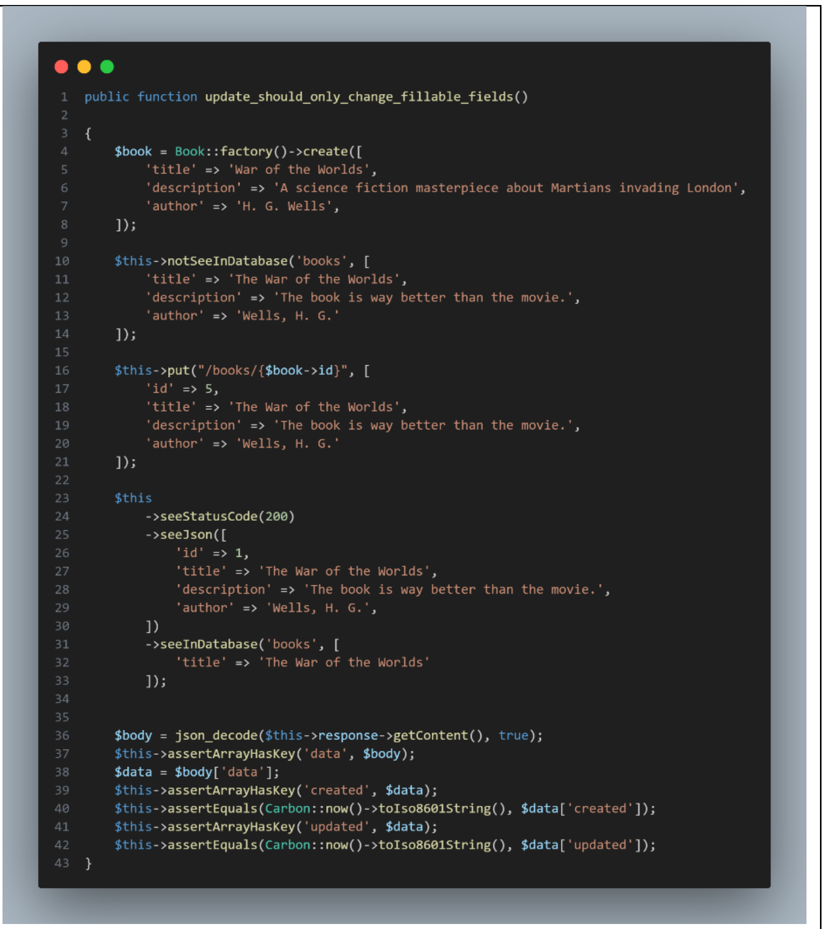
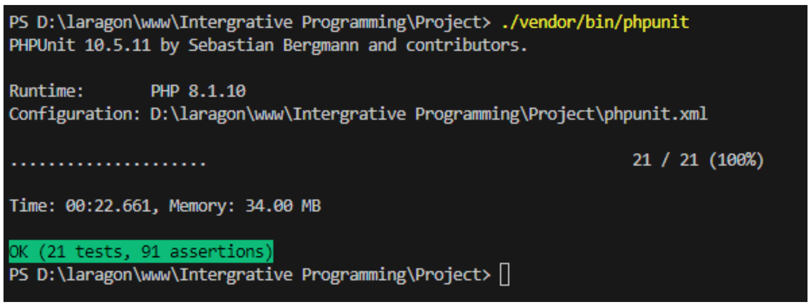
```php
public function show_should_return_a_valid_book()
{
    $book = Book::factory()->create();

    $this
        ->get("/books/{$book->id}")
        ->seeStatusCode(200);

    // Get the response and assert the data key exists
    $content = json_decode($this->response->getContent(), true);
    $this->assertArrayHasKey('data', $content);
    $data = $content['data'];

    // Assert the Book Properties match
    $this->assertEquals($book->id, $data['id']);
    $this->assertEquals($book->title, $data['title']);
    $this->assertEquals($book->description, $data['description']);
    $this->assertEquals($book->author, $data['author']);
    $this->assertEquals($book->created_at->toIso8601String(), $data['created']);
    $this->assertEquals($book->updated_at->toIso8601String(), $data['created']);
}
```

Function store_should_save_new_book_in_the_database



```php
public function store_should_save_new_book_in_the_database()
{
    $this->post('/books', [
        'title' => 'The Invisible Man',
        'description' => 'An invisible man is trapped in the terror of his own creation',
        'author' => 'H. G. Wells'
    ]);

    $body = json_decode($this->response->getContent(), true);
    $this->assertArrayHasKey('data', $body);
    $data = $body['data'];
    $this->assertEquals('The Invisible Man', $data['title']);
    $this->assertEquals(
        'An invisible man is trapped in the terror of his own creation',
        $data['description']
    );
    $this->assertEquals('H. G. Wells', $data['author']);
    $this->assertTrue($data['id'] > 0, 'Expected a positive integer, but did not see one.');
    $this->assertArrayHasKey('created', $data);
    $this->assertEquals(Carbon::now()->toIso8601String(), $data['created']);
    $this->assertArrayHasKey('updated', $data);
    $this->assertEquals(Carbon::now()->toIso8601String(), $data['updated']);
    $this->seeInDatabase('books', ['title' => 'The Invisible Man']);
}
```

Function update_should_only_change_fillable_fields

		
11	Testing phpunit	Hasil 

```php
public function update_should_only_change_fillable_fields()

{
    $book = Book::factory()->create([
        'title' => 'War of the Worlds',
        'description' => 'A science fiction masterpiece about Martians invading London',
        'author' => 'H. G. Wells',
    ]);

    $this->notSeeInDatabase('books', [
        'title' => 'The War of the Worlds',
        'description' => 'The book is way better than the movie.',
        'author' => 'Wells, H. G.'
    ]);

    $this->put("/books/{$book->id}", [
        'id' => 5,
        'title' => 'The War of the Worlds',
        'description' => 'The book is way better than the movie.',
        'author' => 'Wells, H. G.'
    ]);

    $this
        ->seeStatusCode(200)
        ->seeJson([
            'id' => 1,
            'title' => 'The War of the Worlds',
            'description' => 'The book is way better than the movie.',
            'author' => 'Wells, H. G.',
        ])
        ->seeInDatabase('books', [
            'title' => 'The War of the Worlds'
        ]);


    $body = json_decode($this->response->getContent(), true);
    $this->assertArrayHasKey('data', $body);
    $data = $body['data'];
    $this->assertArrayHasKey('created', $data);
    $this->assertEquals(Carbon::now()->toIso8601String(), $data['created']);
    $this->assertArrayHasKey('updated', $data);
    $this->assertEquals(Carbon::now()->toIso8601String(), $data['updated']);
}
```

```
PS D:\laragon\www\Intergrative Programming\Project> ./vendor/bin/phpunit
PHPUnit 10.5.11 by Sebastian Bergmann and contributors.

Runtime:       PHP 8.1.10
Configuration: D:\laragon\www\Intergrative Programming\Project\phpunit.xml

.....................                                             21 / 21 (100%)

Time: 00:22.661, Memory: 34.00 MB

OK (21 tests, 91 assertions)
PS D:\laragon\www\Intergrative Programming\Project> 
```

TEORI

I. Lembar Soal

N/A

II. Lembar Kerja

N/A