

Министерство образования Республики Беларусь  
Учреждение образования  
«Брестский государственный технический  
университет» Кафедра ИИТ

**Отчет по лабораторной работе 4**  
**Дисциплина “ПриС”**

**Выполнила:**

Студентка группы ПО-3

Ковалева А. И.

**Проверил:**

Лаврущик А. И.

Брест 2021

## Вариант 12

### Цель работы:

Познакомиться с механизмами модульного тестирования веб-приложений, построенных на гексагональной архитектуре.

### Задание для выполнения:

Установите и настройте PHPUnit. Напишите модульные тесты для сценария транзакции из ЛР №1. Постарайтесь добиться 100% покрытия кода тестами. При написании постарайтесь учитывать, что в дальнейшем части этого кода вам могут пригодиться при тестировании доменной модели (ЛР №5) и сервисов приложения (ЛР №6).

Предметная область: блог о фермерском хозяйстве.

### Ход работы:

```
composer require --dev symfony/phpunit-bridge
```

./bin/phpunit для финала настройки тестового окружения. Команда успешно выполнялась и теперь мы можем перейти к выполнению написания тестов. Тесты будут храниться в папке tests и представляют из себя классы с функциями которые выполняют различные элементы логики приложения и проверяют её на целостность и корректность выполнения.

Для наших задач подойдет два файла тестов на php:

1. UserTest — тесты модели данных пользователя.
2. NoteTest — тесты модели данных заметки.

#### UserTest.php

```
<?php namespace App\Tests;

use Symfony\Bundle\FrameworkBundle\Test\KernelTestCase;

class UserTest extends KernelTestCase
```

```

{

    private $manager;

    function configure(): void
    {

        $kernel = self::bootKernel();

        $this->manager = $kernel->getContainer()

            ->get('doctrine')

            ->getManager();

    }

    public function testInsert()
    {

        $repository = $this->manager

            ->getRepository(\App\Entity\User::class);

        $original_count = $repository->getUserCount();

        $repository->addUser("Nastya", 19);

        $new_count = $repository->getUsersCount();

        $this->assertEquals($original_count + 1,
$хew_count);

    }

    public function testFindAll()
    {

        $repository = $this->manager

            ->getRepository(\App\Entity\User::class);

        $our_count = $repository->getUsersCount();

        $get_all_count = count($repository->findAll());

```

```
        $this->assertEquals($our_count, $get_all_count);
    }
}
```

### NoteTest.php

```
<?php namespace App\Tests;

use Symfony\Bundle\FrameworkBundle\Test\KernelTestCase;

class NotesTest extends KernelTestCase
{
    private $manager;

    function configure(): void
    {
        $kernel = self::bootKernel();
        $this->manager = $kernel->getContainer()
            ->get('doctrine')
            ->getManager();
    }

    public function testInsert()
    {
        $repository = $this->manager
            ->getRepository(\App\Entity>Note::class);

        $original_count = $repository->getShipsCount();

        $repository->addNote("University", 8);

        $new_count = $repository->getNotesCount();
    }
}
```

```
        $this->assertEquals($original_count + 1,
$new_count);
    }

    public function testFindAll()
    {
        $repository = $this->manager
            ->getRepository(\App\Entity\Note::class);

        $our_count = $repository->getNotesCount();
        $get_all_count = count($repository->findAll());
        $this->assertEquals($our_count, $get_all_count);
    }
}
```

## Выводы:

В данной лабораторной работе я познакомилась с механизмами модульного тестирования веб-приложений, построенных на гексагональной архитектуре.