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

Отчет
По лабораторной работе №4
по дисциплине «ПрИС»

Выполнила
студентка 3 курса
группы ПО-3
Гаврилкович Е.В.
Проверил
Лаврущик А.И.

Брест 2021

Лабораторная работа №4

Модульное тестирование

ВАРИАНТ 5

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 No1. Постарайтесь добиться 100% покрытия кода тестами. При написании постарайтесь учитывать, что в дальнейшем части этого кода вам могут пригодиться при тестировании доменной модели (ЛР5) и сервисов приложения (ЛР6).

Предметная область. Промышленная робототехника.

Ход работы

В ходе данной лабораторной работы логика была переписана с использованием Symfony, были получены 3 репозитория: ProgrammerRepository, RobotRepository, RobotSessionRepository.

RobotTest.php - тесты модели данных роботов

```
<?php namespace App\Tests;

use PHPUnit\Framework\TestCase;
use Symfony\Bundle\FrameworkBundle\Test\KernelTestCase;

class RobotTest extends KernelTestCase
{
    private $entityManager;

    protected function setUp(): void
    {
        $kernel = self::bootKernel();

        $this->entityManager = $kernel->getContainer()->get('doctrine')-
>getManager();
    }

    public function testInsert()
    {
        $repository = $this->entityManager-
>getRepository(\App\Entity\Robot::class);
        $original_count = $repository->getRobotsCount();
        $repository->addRobot(«Robot-php", 1000, 10);
        $new_count = $repository->getRobotsCount();
        $this->assertEquals($original_count + 1, $new_count);
    }

    public function testFindAll()
    {
        $repository = $this->entityManager-
>getRepository(\App\Entity\Robot::class);
        $our_count = $repository->getRobotsCount();
```

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

ProgrammerTest.php - тесты модели данных программистов

```
<?php namespace App\Tests;

use PHPUnit\Framework\TestCase;
use Symfony\Bundle\FrameworkBundle\Test\KernelTestCase;

class ProgrammerTest extends KernelTestCase
{
    private $entityManager;

    protected function setUp(): void
    {
        $kernel = self::bootKernel();

        $this->entityManager = $kernel->getContainer()->get('doctrine')-
>getManager();
    }

    public function testInsert()
    {
        $repository = $this->entityManager-
>getRepository(\App\Entity\Programmer::class);
        $original_count = $repository->getProgrammersCount();
        $repository->addProgrammer("Ivan", 50505);
        $new_count = $repository->getProgrammersCount();
        $this->assertEquals($original_count + 1, $new_count);
    }

    public function testFindAll()
    {
        $repository = $this->entityManager-
>getRepository(\App\Entity\Programmer::class);
        $our_count = $repository->getProgrammersCount();
        $get_all_count = count($repository->findAll());
        $this->assertEquals($our_count, $get_all_count);
    }
}
```

RobotSessionTest.php - тесты модели данных сессии создания роботов

```
<?php namespace App\Tests;

use PHPUnit\Framework\TestCase;
use Symfony\Bundle\FrameworkBundle\Test\KernelTestCase;

class RobotSessionTest extends KernelTestCase
{
    private $entityManager;

    protected function setUp(): void
```

```
    {
        $kernel = self::bootKernel();
        $this->entityManager = $kernel->getContainer()->get('doctrine')-
>getManager();
    }

    public function testInsert()
    {
        $repository = $this->entityManager-
>getRepository(\App\Entity\RobotSession::class);
        $putchase_repository = $this->entityManager-
>getRepository(\App\Entity\Robot::class);
        $Programmer_repository = $this->entityManager-
>getRepository(\App\Entity\Programmer::class);
        $new_Robot = $putchase_repository->addRobot("test Robot", 10,
10);
        $new_Programmer = $Programmer_repository->addProgrammer("test
Programmer", 11111);
        $original_count = $repository->getRobotSessionCount();
        $repository->addRobotSession(new \DateTime(), $new_Robot,
$new_Programmer);
        $new_count = $repository->getRobotSessionCount();
        $this->assertEquals($original_count + 1, $new_count);
    }

    public function testFindAll()
    {
        $repository = $this->entityManager-
>getRepository(\App\Entity\RobotSession::class);
        $our_count = $repository->getRobotSessionCount();
        $get_all_count = count($repository->findAll());
        $this->assertEquals($our_count, $get_all_count);
    }
}
```

Результат работы тестов:

```
Testing tests
...... 6 / 6 (166%)

Time: 170 ms, Memory: 15.00 МБ

OK (6 tests, 6 assert )
```

Вывод

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