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

**Лабораторная работа №6**
**По дисциплине ПрИС**

Выполнила: студентка группы ПО-3
Ковалева А.И.
Проверил: Лаврущик А.И.

Брест 2021

**Цель:** познакомиться с практической реализацией принципа инверсии зависимостей, принципа единственной ответственности и механизмом внедрения зависимостей.
**Предметная область:** блог о фермерском хозяйстве.

**Задание для выполнения:** реализуйте сервисы на уровне приложения, производящие с доменными моделями те же действия, что и сценарий транзакции из ЛР №1. Хотя бы одно действие должно реализовываться доменным сервисом.

## Ход работы

**Агрегаты:**
- Note
- User

**Сервисы:**
- NoteService

**Репозитории:**
- NoteRepository
- UserRepository
- SpecializationRepository
- CategoryRepository
- NoteChangeEventRepository

**NoteController:**

```php
<?php

namespace App\Controller;

use App\Entity\Note;
use App\Form\NoteType;
use App\Service\NoteService;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Component\HttpFoundation\Request;

class NoteController extends AbstractController
{
    private $noteService;

    public function __construct(NoteService $noteService)
    {
        $this->noteService=$noteService;
    }
```

```
    /**
     * @Route("/notes", name="notes")
     */
    public function getAllNotes()
    {
        return $this->render('note/notes.html.twig', [
            'notes' => $this->noteService->getAllNotes(),
        ]);
    }

    /**
     * @Route("/notes/delete/{id}", name="deleteNoteById")
     */
    public function deleteNoteById($id)
    {
        $this->noteService->deleteNoteById($id);
        return $this->redirectToRoute('notes');
    }

    /**
     * @Route("/notes/create", name="note")
     */
    public function createNote(Request $request)
    {
        $note = new Note();
        $form = $this->createForm(NoteType::class, $note);
        $form->handleRequest($request);

        if ($form->isSubmitted() && $form->isValid()) {
            $this->noteService->addNote($note);
            return $this->redirectToRoute('notes');
        }

        return $this->render('note/note_form.html.twig', [
            'form' => $form->createView(),
        ]);
    }
}
```

**NoteService:**

```
<?php

namespace App\Service;

use App\Entity\Note;
use App\Repository\NoteRepository;
```

```php
class NoteService {
    private $noteRepository;

    public function __construct(NoteRepository $noteRepository)
    {
        $this->noteRepository=$noteRepository;
    }

    public function getAllNotes() {
        return $this->noteRepository->findAll();
    }

    public function addNote(Note $note) {
        $this->noteRepository->add($note);
    }

    public function deleteNoteById($id) {
        $this->noteRepository->delete($id);
    }
}
```

**Вывод**: я познакомилась с практической реализацией принципа инверсии зависимостей, принципа единственной ответственности и механизмом внедрения зависимостей.