Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти й газу

Кафедра ІПЗ

Лабораторна робота №6

ПЕРЕДАВАННЯ ДАНИХ МІЖ СТОРІНКАМИ (КОМПОНЕНТАМИ ОДНОГО РІВНЯ)

Виконав:

ст. гр. ІП-22-4

Пархоменко М.А.

Перевірила:

Гобир Л. М.

Івано-Франківськ

2025р.

**Мета роботи:** Метою лабораторної роботи є освоєння механізму передавання даних між компонентами одного рівня за допомогою localStorage у React-додатку, а також демонстрація методів життєвого циклу компонентів за допомогою хука useEffect.

**Теоретичні відомості**

* **Передавання даних**: Для передавання даних між компонентами одного рівня використовується вбудований механізм браузера localStorage, який дозволяє зберігати дані без терміну дії між сеансами.
* **Синтаксис localStorage**:
  + Збереження: localStorage.setItem('ключ', 'значення');
  + Зчитування: const значення = localStorage.getItem('ключ');
  + Видалення: localStorage.removeItem('ключ');
  + Очищення: localStorage.clear();
* **Життєвий цикл компонентів**: Хук useEffect у React імітує методи життєвого циклу (componentDidMount, componentDidUpdate, componentWillUnmount).

**Завдання:**

**1. Опис реалізації**

**1.1. Структура проєкту**

* Створено папку src/pages для однорівневих компонентів, які викликаються через роутер.
* Структура файлів:
  + src/pages/VacancyListPage.jsx – головна сторінка зі списком вакансій.
  + src/pages/AppliedVacanciesPage.jsx – сторінка відгукнутих вакансій.
  + src/pages/VacancyDetailPage.jsx – сторінка деталей вакансії.
  + src/components/ – містить дочірні компоненти (VacancyList, VacancyItemComponent, VacancyStatus тощо).
  + src/models/constants.js – файл із константою для localStorage.
  + src/App.js – конфігурація роутів.

**1.2. Використання localStorage**

* Дані вакансій зберігаються в localStorage під ключем LOCALSTORE\_VACANCIES.
* Приклад зчитування та запису в VacancyListPage.jsx:
  + Зчитування:

const getLocalStore = () => {

const storedVacancies = window.localStorage.getItem(LOCALSTORE\_VACANCIES);

if (storedVacancies) {

const parsedVacancies = JSON.parse(storedVacancies);

if (Array.isArray(parsedVacancies)) {

setVacancies(parsedVacancies);

}

}

};

Запис:

const handleToggleApply = (id) => {

const updatedVacancies = vacancies.map((v) =>

v.id === id ? { ...v, applied: !v.applied } : v

);

setVacancies(updatedVacancies);

window.localStorage.setItem(LOCALSTORE\_VACANCIES, JSON.stringify(updatedVacancies));

};

**3.3. Роутеризація**

* Використовується бібліотека react-router-dom для навігації між сторінками.
* Приклад конфігурації в App.js:

<Routes>

<Route path="/" element={<VacancyListPage />} />

<Route path="/vacancy/:id" element={<VacancyDetailPage />} />

</Routes>

**3.4. Демонстрація useEffect**

* Створено компонент VacancyStatus.jsx для імітації методів життєвого циклу:
  + componentDidMount:

javascript

Copy

useEffect(() => {

console.log(`Компонент для вакансії ${name} змонтовано`);

});

**componentWillUnmount:**

useEffect(() => {

return () => {

console.log(`Компонент для вакансії ${name} розмонтовано`);

};

}, []);

* + componentDidUpdate:

useEffect(() => {

console.log(`Статус applied для ${name} змінився: ${applied}`);

}, [applied]);

* Зчитування даних при монтуванні в VacancyListPage.jsx:

useEffect(() => {

getLocalStore();

}, []);

**1.5. Покращення коду**

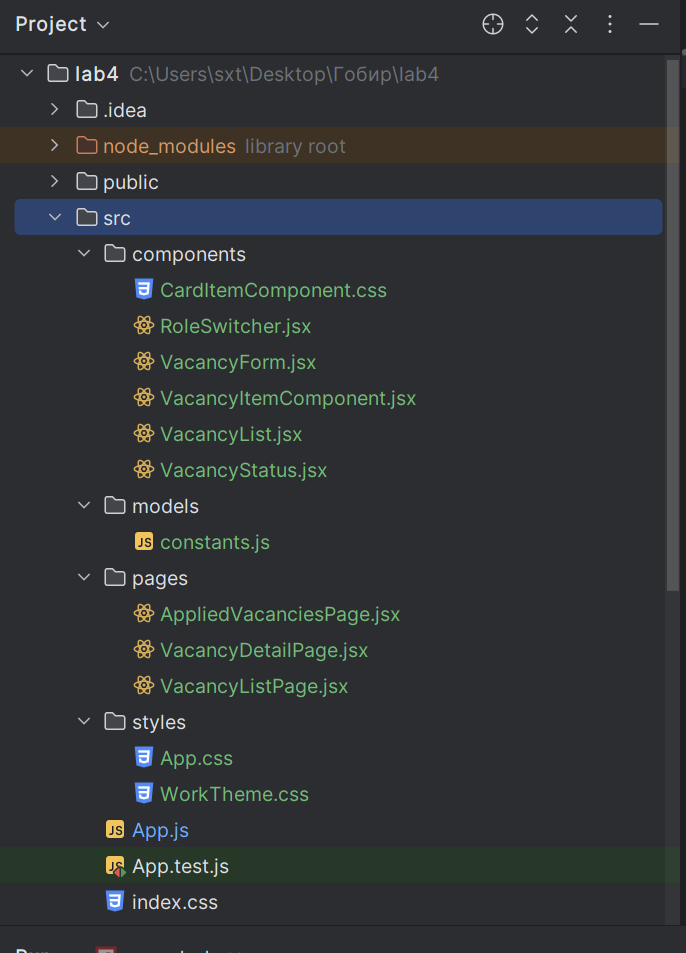
* Константа LOCALSTORE\_VACANCIES винесена в окремий файл constants.js:

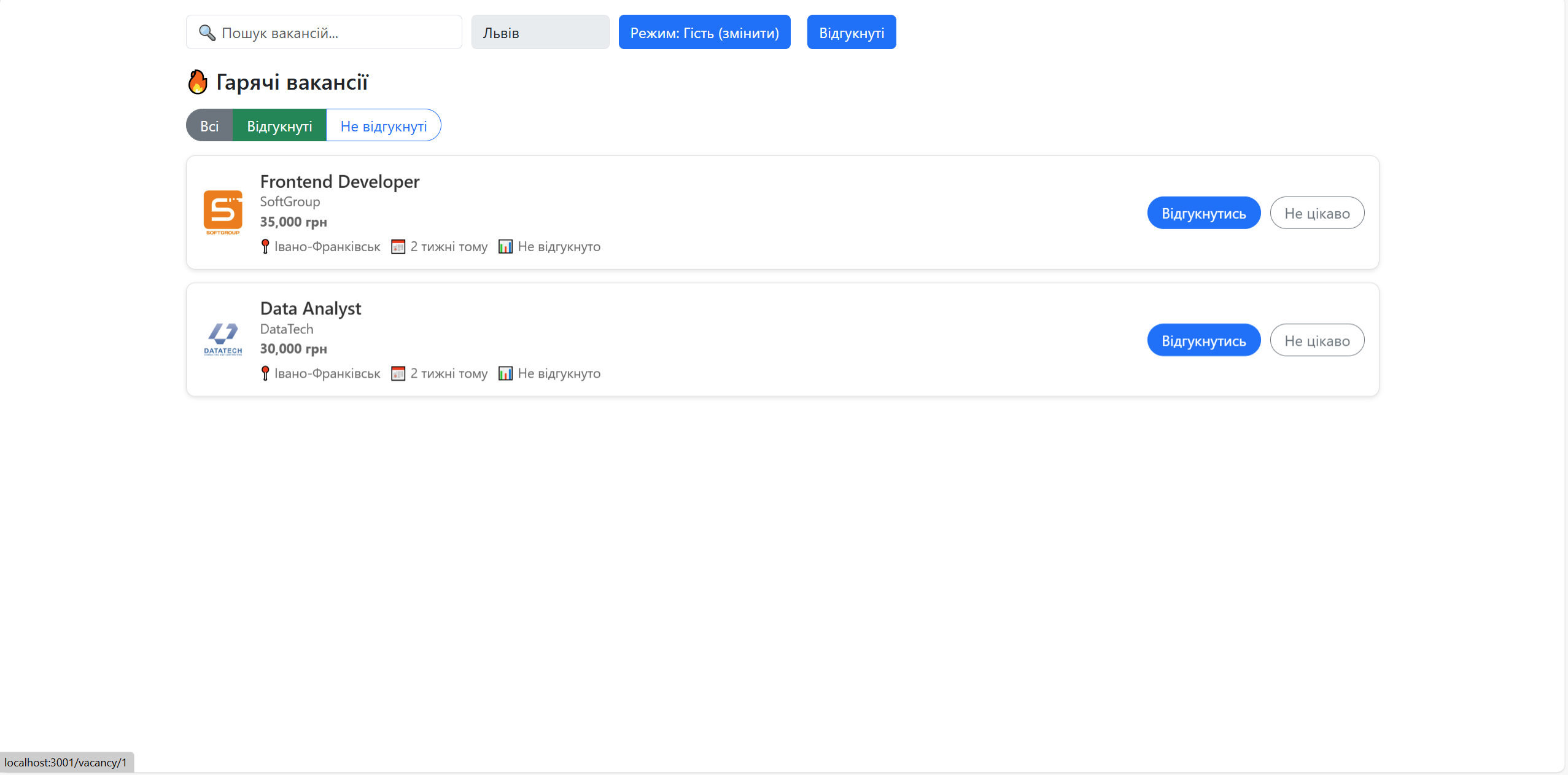
export const LOCALSTORE\_VACANCIES = "LOCALSTORE\_VACANCIES";

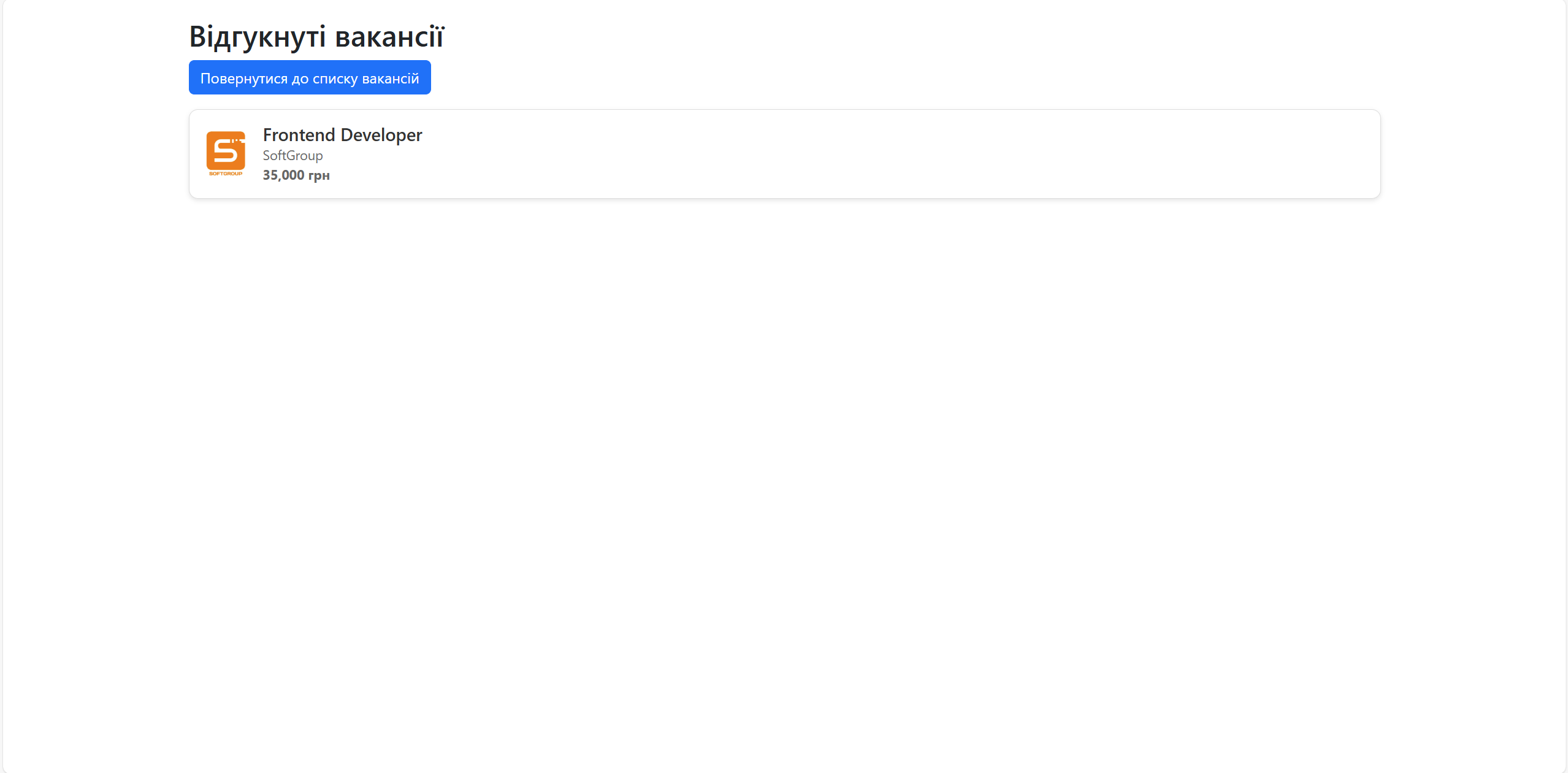
* Імпорт у компонентах:

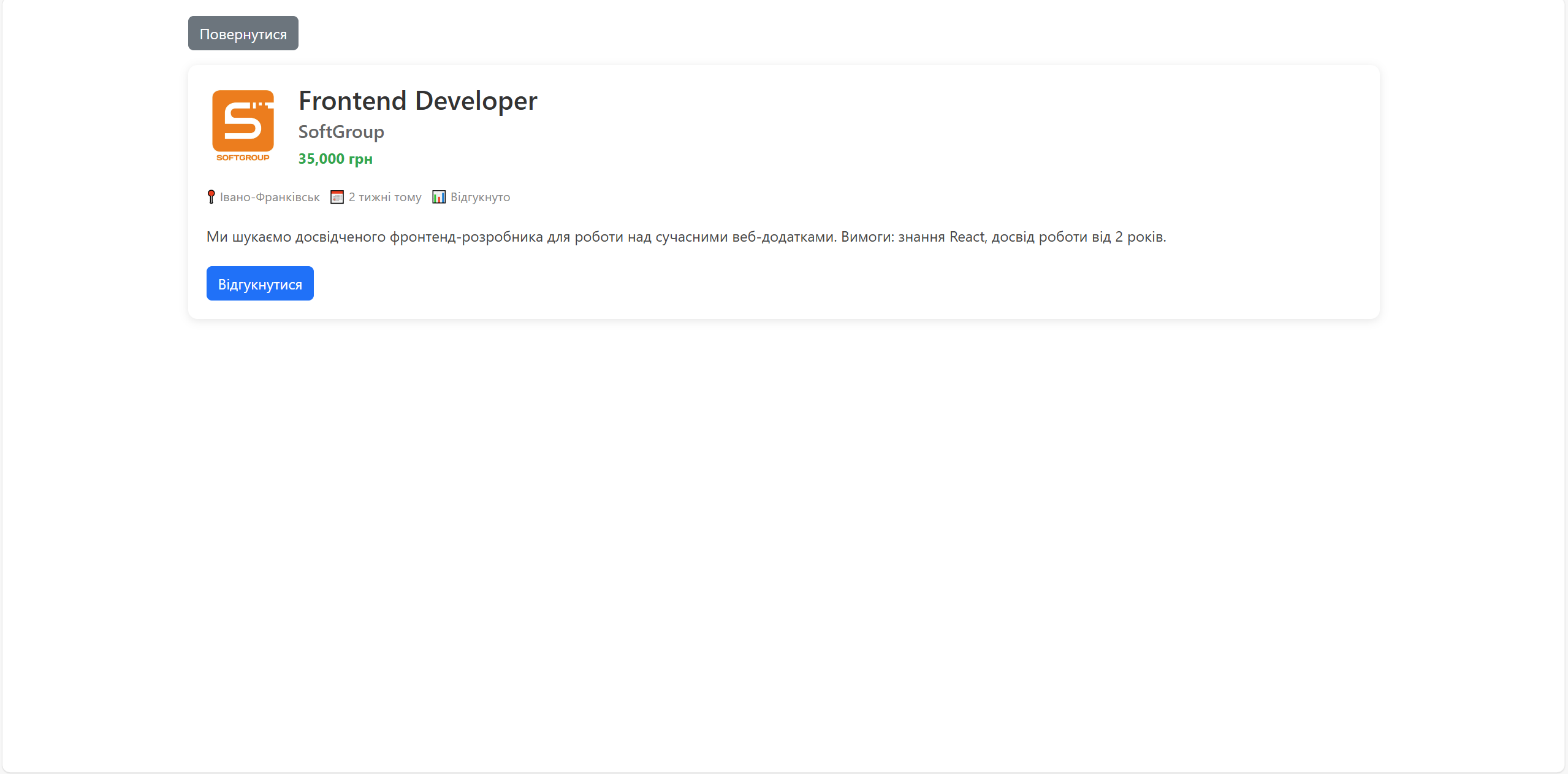
import { LOCALSTORE\_VACANCIES } from "../models/constants";

Результати роботи:









Висновок: В даній лабораторній роботі я навчився використовувати компоненти та передавати дані між ними.