МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ОСТРОЗЬКА АКАДЕМІЯ"
Кафедра економіко-математичного моделювання та інформаційних технологій

ЗВІТ
Лабораторна робота №7
CRUD-застосунку на React і Redux для Product

здобувачки вищої освіти
першого (бакалаврського) рівня
третього року навчання групи КН-31
спеціальності 122 Комп'ютерні науки
ОПП «Комп'ютерні науки»

Лайтер Ярини Семенівни
(прізвище, ім'я, по батькові)

База лабораторної - кафедра економіко-математичного моделювання та інформаційних технологій

Керівник лабораторної від університету

_____
(науковий ступінь, учене звання керівника)

_____
(Власне ім'я та ПРІЗВИЩЕ)

Члени комісії:

_____

Острог - 2023

Зміст

Опис

Тема: CRUD-застосунку на React і Redux для сутності Product
Мета: Ознайомити студентів зі створенням додатків на React з використанням Redux та редукторів (reducers), реалізація CRUD операцій

Завдання

1. Створення структури проекту:

    a. Встановлення React та Redux.
    b. Створення компонентів для перегляду, додавання, редагування та видалення продуктів.

2. Модель даних:

    c. Визначення моделі продукту з полями: назва (string), опис (string), кількість (number), ціна (number), знижка (number).

3. Redux:

    d. Створення дій (actions) для CRUD-операцій: додавання, оновлення, видалення, отримання продуктів.
    e. Створення редукторів (reducers) для обробки дій та оновлення стану.

4. Компоненти:

    f. Список продуктів: Відображення всіх продуктів з можливістю видалення та редагування.
    g. Форма продукту: Форма для додавання та редагування продукту. Має включати валідацію для перевірки коректності введених даних.

5. Валідація:

    h. Перевірка на заповнення всіх полів.
    i. Перевірка, що кількість та ціна є позитивними числами.
    j. Можливо, додаткові правила валідації залежно від бізнес-логіки.

6. UI/UX:

    k. Дизайн інтерфейсу для зручного користування.
    l. Переконатися, що форма є зрозумілою та легкою в користуванні.

Виконання

1. **Створення структури проекту:**

Відкриваємо наш попередній проект з лабораторної роботи №6 та створюємо нові компоненти:

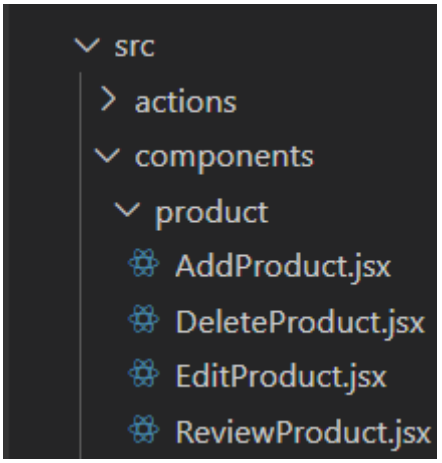
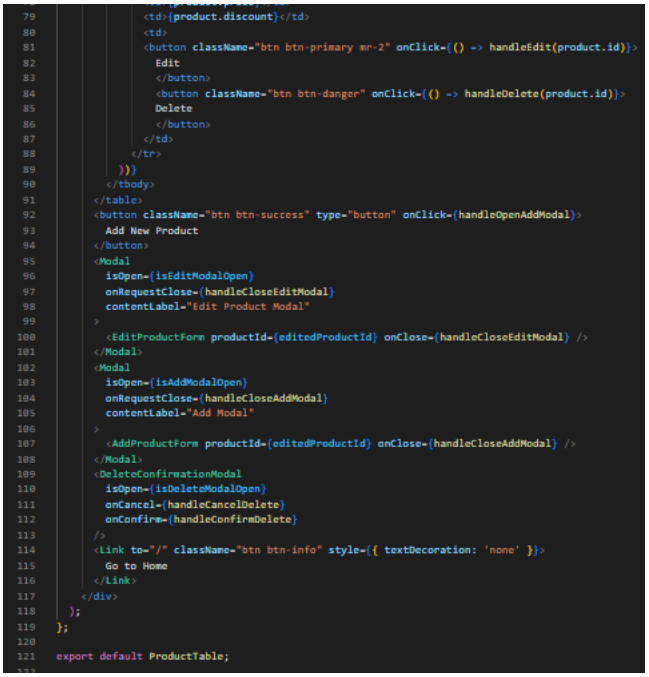


Рис 1.1 Удосконалена структура проекту

Створюємо компонент ReciewProduct.jsx:

```jsx
// components/product/ProductTable.jsx
import React, { useState } from 'react';
import { useSelector, useDispatch } from 'react-redux';
import { deleteProduct } from '../../actions/productActions';
import Modal from 'react-modal';
import { Link } from 'react-router-dom';
import { toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
import '../../style/UserTable.css';
import EditProductForm from './EditProduct';
import AddProductForm from './AddProduct';
import DeleteConfirmationModal from './DeleteProduct';

Modal.setAppElement('#root');

const ProductTable = () => {
  const products = useSelector((state) => state.products);
  const dispatch = useDispatch();

  const [isEditModalOpen, setIsEditModalOpen] = useState(false);
  const [isAddModalOpen, setIsAddModalOpen] = useState(false);
  const [isDeleteModalOpen, setIsDeleteModalOpen] = useState(false);
  const [editedProductId, setEditedProductId] = useState(null);

  const handleEdit = (productId) => {
    setEditedProductId(productId);
    setIsEditModalOpen(true);
  };


  const handleCloseEditModal = () => {
    setIsEditModalOpen(false);
    setEditedProductId(null);
  };

  const handleDelete = (productId) => {
    setEditedProductId(productId);
    setIsDeleteModalOpen(true);
  };

  const handleConfirmDelete = () => {
    dispatch(deleteProduct(editedProductId));
    toast.success('Product deleted successfully', { position: toast.POSITION.BOTTOM_RIGHT });
    setIsDeleteModalOpen(false);
  };

  const handleCancelDelete = () => {
    setIsDeleteModalOpen(false);
  };

  const handleOpenAddModal = () => {
    setIsAddModalOpen(true);
  };

  const handleCloseAddModal = () => {
    setIsAddModalOpen(false);
  };

  return (
    <div className="container mt-4">
      <table className="table">
        <thead>
          <tr>
            <th scope="col">Name</th>
            <th scope="col">Description</th>
            <th scope="col">Number</th>
            <th scope="col">Price</th>
            <th scope="col">Discount</th>
            <th scope="col">Actions</th>
          </tr>
```

Рис 1. 2 Код сторінки ReciewProduct*.jsx, частина 1*



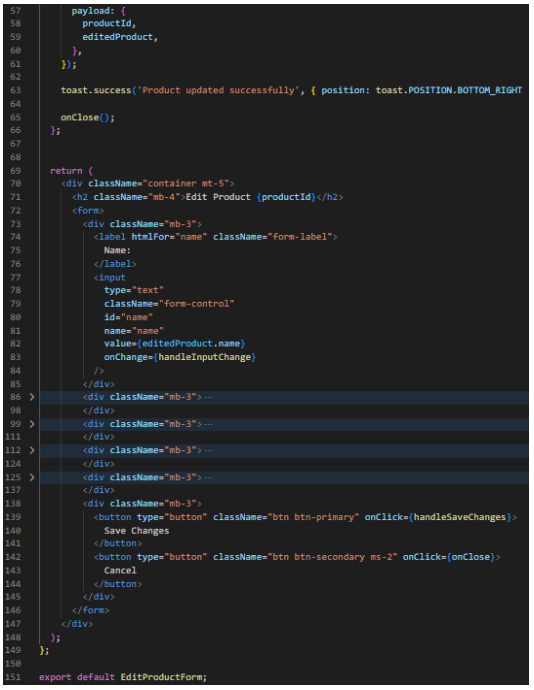
```
            <td>{product.discount}</td>
            <td>
            <button className="btn btn-primary mr-2" onClick={() => handleEdit(product.id)}>
              Edit
              </button>
              <button className="btn btn-danger" onClick={() => handleDelete(product.id)}>
              Delete
              </button>
            </td>
          </tr>
        ))}
      </tbody>
    </table>
    <button className="btn btn-success" type="button" onClick={handleOpenAddModal}>
      Add New Product
    </button>
    <Modal
      isOpen={isEditModalOpen}
      onRequestClose={handleCloseEditModal}
      contentLabel="Edit Product Modal"
    >
      <EditProductForm productId={editedProductId} onClose={handleCloseEditModal} />
    </Modal>
    <Modal
      isOpen={isAddModalOpen}
      onRequestClose={handleCloseAddModal}
      contentLabel="Add Modal"
    >
      <AddProductForm productId={editedProductId} onClose={handleCloseAddModal} />
    </Modal>
    <DeleteConfirmationModal
      isOpen={isDeleteModalOpen}
      onCancel={handleCancelDelete}
      onConfirm={handleConfirmDelete}
    />
    <Link to="/" className="btn btn-info" style={{ textDecoration: 'none' }}>
      Go to Home
    </Link>
  </div>
 );
};

export default ProductTable;
```

Рис 1.3 Код сторінки ReciewProduct*.jsx, частина 2*

Створюємо компонент EditProduct.jsx:

```
// components/product/EditProductForm.jsx
import React, { useState, useEffect } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import { toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';

const EditProductForm = ({ productId, onClose }) => {
  const dispatch = useDispatch();
  const products = useSelector((state) => state.products);

  const productToEdit = products.find((product=> product.id === productId));

  const [editedProduct, setEditedProduct] = useState({
    name: '',
    description: '',
    number: '',
    price: '',
    discount: '',
  });

  useEffect(() => {
    if (productToEdit) {
      setEditedProduct({
        name: productToEdit.name || '',
        description: productToEdit.description || '',
        number: productToEdit.number || '',
        price: productToEdit.price || '',
        discount: productToEdit.discount || '',
      });
    }
  }, [productToEdit]);

  const handleInputChange = (e) => {
    const { name, value } = e.target;
    setEditedProduct((prevProduct) => ({
      ...prevProduct,
      [name]: value,
    }));
  };

  const handleSaveChanges = () => {
    const changesMade =
    editedProduct.name !== productToEdit.name ||
      editedProduct.description !== productToEdit.description ||
      editedProduct.number !== productToEdit.number ||
      editedProduct.price !== productToEdit.price ||
      editedProduct.discount !== productToEdit.discount;

    if (!changesMade) {
      toast.info('No changes made', { position: toast.POSITION.BOTTOM_RIGHT });
      onClose();
      return;
    }

    dispatch({
      type: 'EDIT_PRODUCT',
```

Рис 1. 4 Код сторінки EditProduct*.jsx, частина 1*



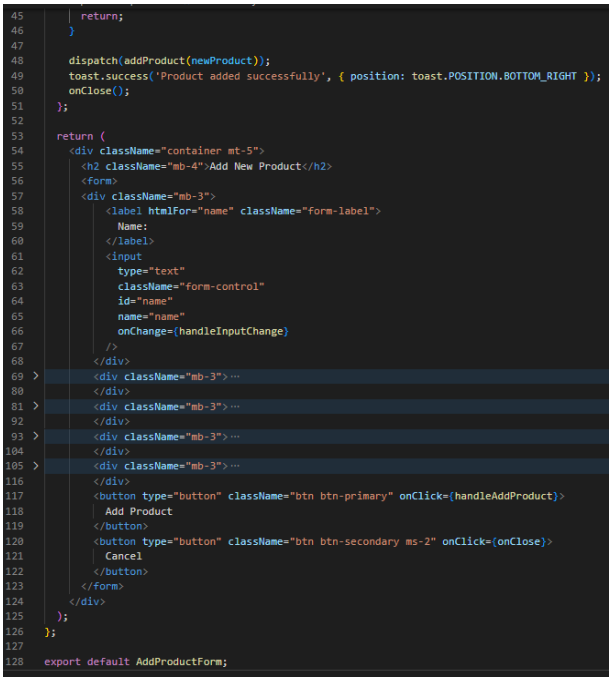
```
      payload: {
        productId,
        editedProduct,
      },
    });

    toast.success('Product updated successfully', { position: toast.POSITION.BOTTOM_RIGHT

    onClose();
  };


  return (
    <div className="container mt-5">
      <h2 className="mb-4">Edit Product {productId}</h2>
      <form>
        <div className="mb-3">
          <label htmlFor="name" className="form-label">
            Name:
          </label>
          <input
            type="text"
            className="form-control"
            id="name"
            name="name"
            value={editedProduct.name}
            onChange={handleInputChange}
          />
        </div>
        <div className="mb-3">···
        </div>
        <div className="mb-3">···
        </div>
        <div className="mb-3">···
        </div>
        <div className="mb-3">···
        </div>
        <div className="mb-3">
          <button type="button" className="btn btn-primary" onClick={handleSaveChanges}>
            Save Changes
          </button>
          <button type="button" className="btn btn-secondary ms-2" onClick={onClose}>
            Cancel
          </button>
        </div>
      </form>
    </div>
  );
};

export default EditProductForm;
```

Рис 1.5 Код сторінки ReciewProduct*.jsx, частина 2*

Створюємо компонент AddProduct.jsx:

```
// components/product/AddProductForm.jsx
import React, { useState } from 'react';
import { useDispatch } from 'react-redux';
import { addProduct } from '../../actions/productActions';
import { v4 as uuidv4 } from 'uuid';
import { toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';

const AddProductForm = ({ onClose }) => {
  const dispatch = useDispatch();

  const [newProduct, setNewProduct] = useState({
    id: uuidv4(),
    name: '',
    description: '',
    number: '',
    price: '',
    discount: '',
    email: '',
  });

  const handleInputChange = (e) => {
    const { name, value } = e.target;
    setNewProduct((prevProduct) => ({
      ...prevProduct,
      [name]: value,
    }));
  };

  const handleAddProduct = () => {
    if (!newProduct.name || !newProduct.description || !newProduct.number || !newProduct.price || !newProduct.discount) {
      toast.error('Please fill in all fields', { position: toast.POSITION.BOTTOM_RIGHT });
      return;
    }

    if (isNaN(newProduct.number) || isNaN(newProduct.price) || isNaN(newProduct.discount)) {
      toast.error('Number, Price, and Discount must be valid numbers', { position: toast.POSITION.BOTTOM_RIGHT });
      return;
    }

    if (Number(newProduct.number) <= 0 || Number(newProduct.price) <= 0) {
      toast.error('Number and Price must be positive numbers', {
        position: toast.POSITION.BOTTOM_RIGHT,
      });
```

Рис 1.6 Код сторінки AddProduct*.jsx, частина 1*



```
45      return;
46    }
47
48    dispatch(addProduct(newProduct));
49    toast.success('Product added successfully', { position: toast.POSITION.BOTTOM_RIGHT });
50    onClose();
51  };
52
53  return (
54    <div className="container mt-5">
55      <h2 className="mb-4">Add New Product</h2>
56      <form>
57      <div className="mb-3">
58          <label htmlFor="name" className="form-label">
59            Name:
60          </label>
61          <input
62            type="text"
63            className="form-control"
64            id="name"
65            name="name"
66            onChange={handleInputChange}
67          />
68        </div>
69 >      <div className="mb-3">···
80        </div>
81 >      <div className="mb-3">···
92        </div>
93 >      <div className="mb-3">···
104       </div>
105 >      <div className="mb-3">···
116       </div>
117       <button type="button" className="btn btn-primary" onClick={handleAddProduct}>
118         Add Product
119       </button>
120       <button type="button" className="btn btn-secondary ms-2" onClick={onClose}>
121         Cancel
122       </button>
123     </form>
124   </div>
125 );
126 };
127
128 export default AddProductForm;
```

Рис 1.7 Код сторінки AddProduct*.jsx, частина 2*

Створюємо компонент DeleteProduct.jsx:

```jsx
import React from 'react';
import Modal from 'react-modal';

Modal.setAppElement('#root');

const DeleteConfirmationModal = ({ isOpen, onCancel, onConfirm }) => {
  return (
    <Modal
      isOpen={isOpen}
      onRequestClose={onCancel}
      contentLabel="Delete Confirmation Modal"
    >
      <h2>Confirm Deletion</h2>
      <p>Are you sure you want to delete this product?</p>
      <button onClick={onConfirm}>Yes, Delete</button>
      <button onClick={onCancel}>Cancel</button>
    </Modal>
  );
};

export default DeleteConfirmationModal;
```

Рис 1.8 Код сторінки DeleteProduct*.jsx, частина 1*

2. **Визначення моделі продукту :**

Створюємо модель продукту з відповідними полями:

```
const initialState = [
    {
      id: 1,
      name: 'Яблука',
      description: 'Солодкі та соковиті',
      number: 10,
      price: 2.5,
      discount: 0.1,
    },
    {
       id: 2,
      name: 'Хліб',
      description: 'Пшеничний хліб',
      number: 5,
      price: 1.2,
      discount: 0.05,
    },
    {
       id: 3,
      name: 'Молоко',
      description: 'Натуральне коров"яче молоко',
      number: 3,
      price: 3,
      discount: 0,
    },
  ];
```

Рис 2.1 Створення масиву

3. **Створюємо редуктори (reducers) та дії (actions):**

Створюємо редуктор productReducer.jsx, який визначає логіку для обробки дій, які можуть виникнути у Redux-застосунку, пов'язаному з користувачами.

```
// reducers/productReducer.js
const initialState = [···
];


const productReducer = (state = initialState, action) => {
  switch (action.type) {
    case 'ADD_PRODUCT':
      return [...state, action.payload];

    case 'EDIT_PRODUCT':
      const { productId, editedProduct } = action.payload;
      const updatedProducts = state.map((product) =>
      product.id === productId ? { ...product, ...editedProduct} : product
      );
      return updatedProducts;

    case 'DELETE_PRODUCT':
      return state.filter((product) => product.id !== action.payload.productId);

    default:
      return state;
  }
};

export default productReducer;
```

Рис 3.1 Код сторінки productReducer.*jsx*

Та створюємо productActions.*jsx*, який відповідає дії, які використовуються в компонентах або діяльностях для відправки запитів до Redux-стору. Коли дія відправляється, відповідний редуктор обробляє цю дію і вносить відповідні зміни в стан додатку*:*

```jsx
// actions/productActions.js
export const editProduct = (productId, updatedProductData) => ({
    type: 'EDIT_PRODUCT',
    payload: { productId, updatedProductData },
  });

  export const deleteProduct = (productId) => ({
    type: 'DELETE_PRODUCT',
    payload: { productId },
  });

  export const addProduct = (product) => ({
    type: 'ADD_PRODUCT',
    payload: product,
  });
```

Рис 3.2 Код сторінки productActions*.jsx*

4. **Відображення списку продуктів:**

Відображення всіх продуктів з можливістю видалення та редагування. :

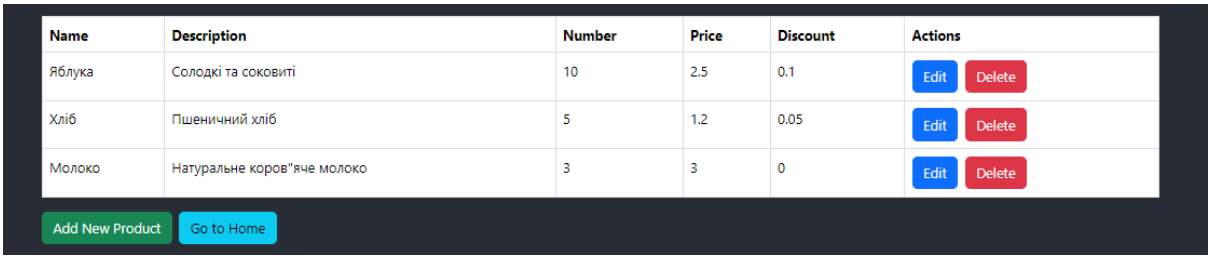


Рис 4.1 Відображення таблиці з продуктами

Форма для додавання та редагування продукту:

Рис 4.1 Форма для редагування

5. Додаємо валідацію

Додаємо валідацію в компонети EditProduc.jsxt AddProduct.jsx:

```
const handleAddProduct = () => {
  if (!newProduct.name || !newProduct.description || !newProduct.number || !newProduct.price || !newProduct.discount) {
    toast.error('Please fill in all fields', { position: toast.POSITION.BOTTOM_RIGHT });
    return;
  }

  if (isNaN(newProduct.number) || isNaN(newProduct.price) || isNaN(newProduct.discount)) {
    toast.error('Number, Price, and Discount must be valid numbers', { position: toast.POSITION.BOTTOM_RIGHT });
    return;
  }

  if (Number(newProduct.number) <= 0 || Number(newProduct.price) <= 0) {
    toast.error('Number and Price must be positive numbers', {
      position: toast.POSITION.BOTTOM_RIGHT,
    });
    return;
  }

  dispatch(addProduct(newProduct));
  toast.success('Product added successfully', { position: toast.POSITION.BOTTOM_RIGHT });
  onClose();
};
```

Рис 5.1 Валідація у компоненті AddProduct.jsx

6. UI/UX:

Створюємо зручний в користуванні дизайн. В результаті отримуємо:

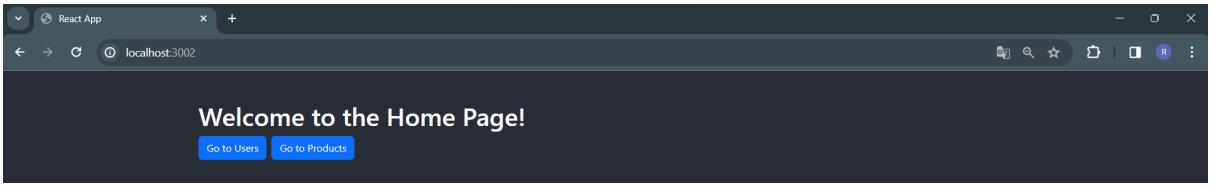


Рис 6.1 Результат запуску програми

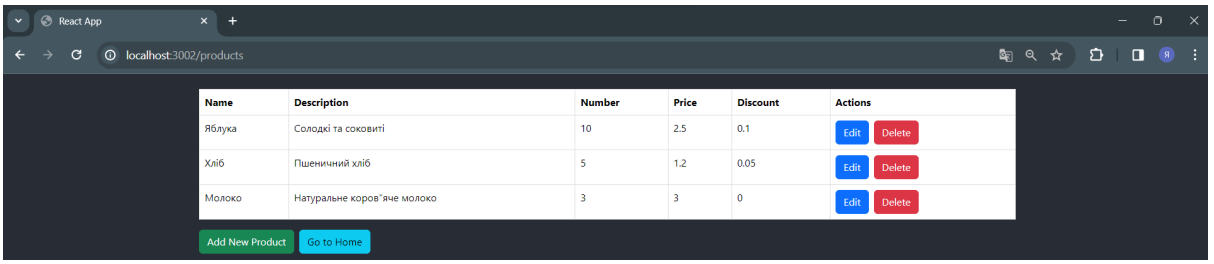


Рис 6.2 Результат запуску програми

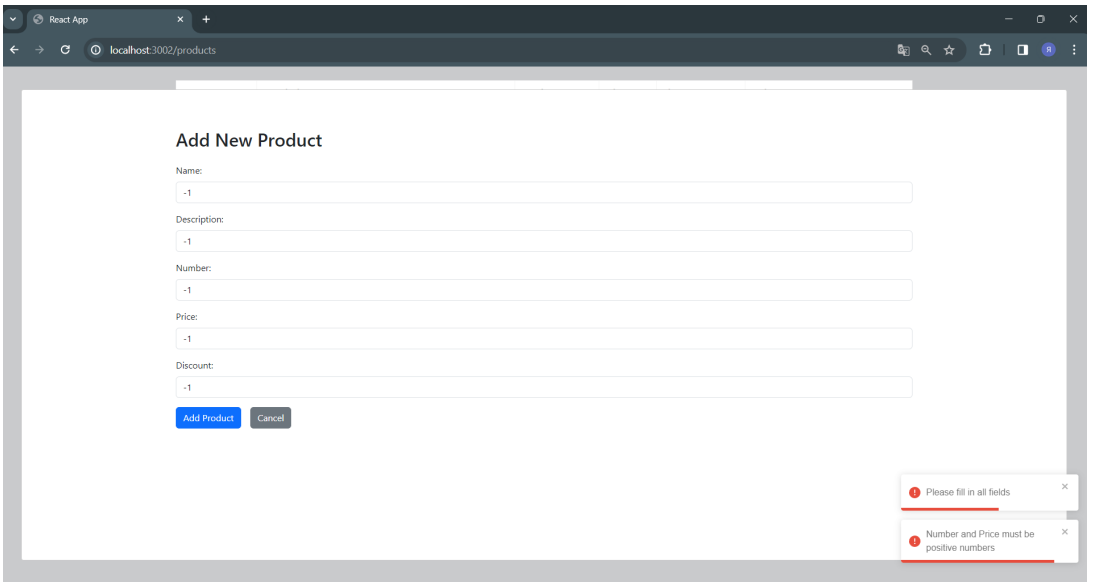


Рис 6.3 Результат запуску програми

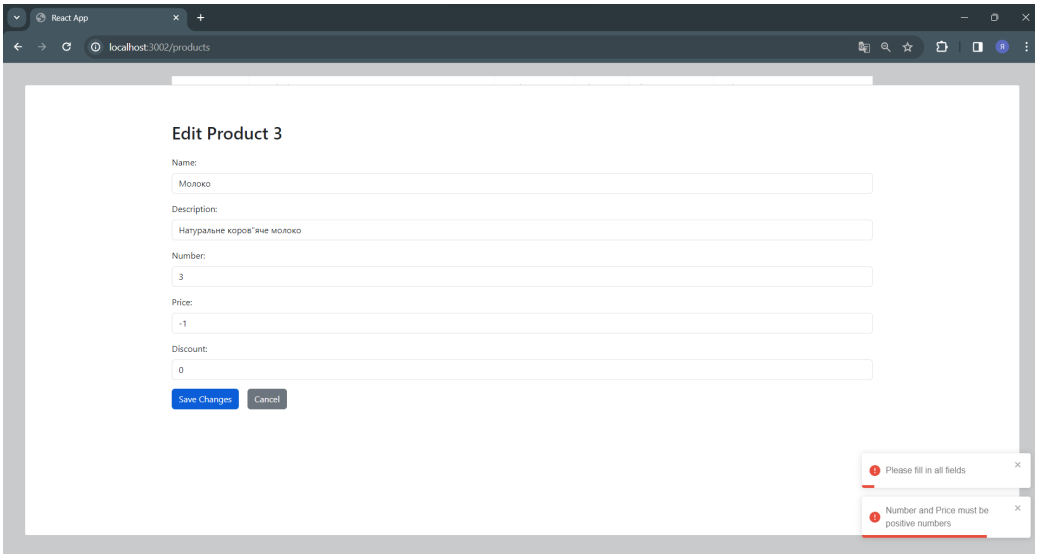


Рис 6.4 Результат запуску програми

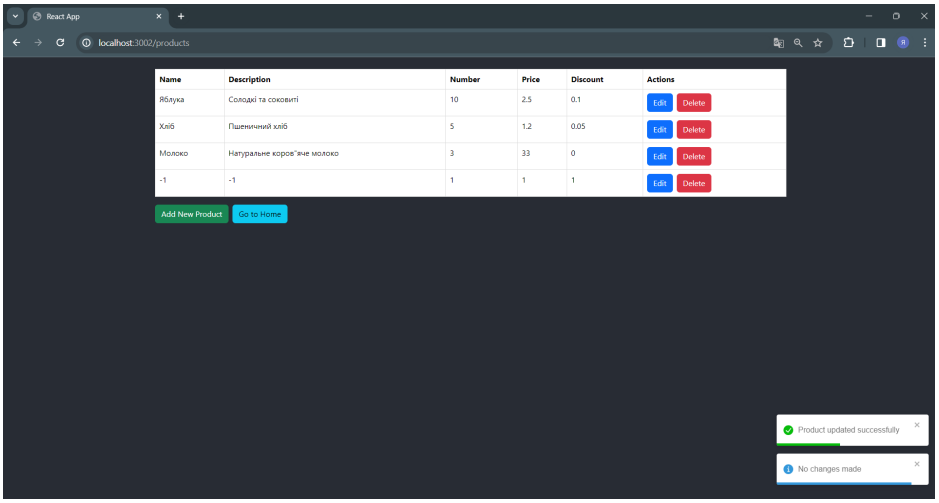


Рис 6.5 Результат запуску програми

Висновок:

На цій лабораторній роботі ми створили CRUD-застосунок на React і Redux з сутністю "Продукт", який містить поля: назва, опис, кількість, ціна, знижка, та з валідацією. Налаштували структуру каталогів, підключили React Router. Окрім того, створили компоненти для сторінок "/home", "/users" та “/products”, які відображають різні дані на кожній сторінці.

Також ми додали стилізація до проекту для поліпшення зовнішнього вигляду та завершено проект, переконавшись у його працездатності на локальному сервері.