МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Лабораторная работа №2
По дисциплине “Специализированные языки разметки документов”
Тема: “Дополнительные функции”

Выполнил:
студент группы ПО-11
Сымоник И.А.
Проверил:
Войцехович Г.Ю.

Брест 2024

Цель работы: реализовать дополнительные функции

Функция доставки:
Пользователь может ввести данные для доставки в форму
const { user, goods } = useContext(Context);

```
  const [user_, setUser] = useState(null);
  const [address, setAddress] = useState({
    street: '',
    building: '',
    entrance: '',
    city: '',
    phone: ''
  });



  const save = () => {
    axios.post("http://localhost:5000/" + "api/profile", {
      street: address.street,
      home: address.building,
      entranceway: address.entrance,
      city: address.city,
      telephone: address.phone
    }).then(response => {
      console.log(response.data);
    }).catch(error => {
      console.error('Error adding item to cart:', error);
    });
  }

  const handleAddressChange = (e) => {
    const { name, value } = e.target;
    setAddress((prevAddress) => ({
      ...prevAddress,
      [name]: value,
    }));
  };

  const handleAddressSubmit = (e) => {
    e.preventDefault();
    setUser((prevUser) => ({
      ...prevUser,
      address: address,
    }));
    setAddress({
```

```
            street: '',
            building: '',
            entrance: '',
            city: '',
            phone: ''
        });
    };


    useEffect(() => {
        axios.get("http://localhost:5000/" + "api/profile")
            .then(response => {

                setAddress({
                    street: response.data[0].street,
                    building: response.data[0].home,
                    entrance: response.data[0].entranceway,
                    city: response.data[0].city,
                    phone: response.data[0].telephone
                });
            }).catch(error => {
                console.error('Error adding item to cart:', error);
            });
    }, [])

    return (
        <div className="profile-page">
            <div className="profile-page-content">
                <h2>Профиль пользователя</h2>
                <div className="user-info">
                    <div className="user-info-header">

                    </div>
                    <div className="user-info-details">
                        <p>Email: {localStorage.getItem('email')}</p>
                        <p>Адрес: {address.street}, {address.building}, подъезд {address.entrance},
{address.city}</p>
                        <p>Телефон: {address.phone}</p>
                    </div>
                </div>
                <div className="address-form">
                    <h3>Изменить адрес доставки</h3>
                    <form onSubmit={handleAddressSubmit}>
                        <label>
                            Улица:
                            <input
                                type="text"
                                name="street"
```

```
            value={address.street}
            onChange={handleAddressChange}
          />
        </label>
        <label>
          Дом:
          <input
            type="text"
            name="building"
            value={address.building}
            onChange={handleAddressChange}
          />
        </label>
        <label>
          Подъезд:
          <input
            type="text"
            name="entrance"
            value={address.entrance}
            onChange={handleAddressChange}
          />
        </label>
        <label>
          Город:
          <input
            type="text"
            name="city"
            value={address.city}
            onChange={handleAddressChange}
          />
        </label>
        <label>
          Телефон:
          <input
            type="text"
            name="phone"
            value={address.phone}
            onChange={handleAddressChange}
          />
        </label>
        <button type="submit" onClick={save}>Сохранить</button>
      </form>
    </div>

    {
      user.isAdmin &&
      <AdminPanel></AdminPanel>
    }
```

```
            </div>
        </div>
    );
}

export default ProfilePage;

app.post('/api/profile', (req, res) => {
    const { street, home, entranceway, city, telephone } = req.body;
    const values = [street, home, entranceway, city, telephone];

    mysqlConnection.query('INSERT INTO userData(street, home, entranceway, city,
telephone) VALUES(?,?,?,?,?)', values, (error_, results_) => {
        if (error_) {
            console.error('Error writing data:', error_);
            res.status(500).json({ message: 'Произошла ошибка. Попробуйте позже' });
        }
        else {
            res.status(200).send('Item removed from cart');
        }
    })
})
```

Можно отправить запрос на доставку

```
import React, { useContext, useEffect, useState } from 'react';
import { Context } from '../index.js';
import './Basket.css';
import axios from 'axios';
import { GetBasket } from '../http/GoodsAPI.js';
import { toJS } from 'mobx';

const Basket = () => {
    const { user, goods } = useContext(Context);
    const [cartItems, setCartItems] = useState([]);
    const [selectedItem, setSelectedItem] = useState(null);

    const handleRemoveFromBasket = (productId) => {
        goods.deleteItemFromBasket(productId);
        console.log('Удалить товар из корзины:', productId);
        console.log(goods.itemsInBasket);

    };


    const fetchCartItems = async () => {
        axios.get(`http://localhost:5000/api/shopping-cart/${localStorage.getItem('email')}`)
            .then(response => {
                setCartItems(response.data);
                console.log(cartItems);
```

```
            })
            .catch(error => {
                console.error('Error fetching shopping cart items:', error);
            });
    }

    const handleRemove = async (itemId) => {

        console.log(itemId);
        axios.delete(`http://localhost:5000/api/shopping-
cart/${localStorage.getItem('email')}/${itemId.ItemID}`)
            .then(response => {
                if (response.status === 200) {
                    setCartItems(prevItems => prevItems.filter(item => item.ItemID !==
itemId.ItemID));
                    console.log(response.data.message);
                }
            })
            .catch(error => {
                console.error('Error removing item from cart:', error);
            });
    }



    useEffect(() => {
        fetchCartItems(cartItems);
        console.log(cartItems)
    }, []);

    useEffect(() => {
        if (!user.email)
            return;


    }, [cartItems]);



    const handleCheckout = () => {
        console.log('Оформить заказ');
        console.log(cartItems)
        cartItems.map((item) => {
            handleRemove(item);
        })
    };

    return (
```

```
        <div className="Basket">
            <h2>Корзина</h2>
            {cartItems.length === 0 ? (
                <p className="empty-Basket-message">Корзина пуста</p>
            ) : (
                <div className="Basket-items">
                    {cartItems.map((item, index) => (

                        <div key={item.id} className="Basket-item">
                            <img src={goods.categories[item.ProductID - 1].image}
alt={goods.categories[item.ProductID - 1].title} className="Basket-item-image" />
                            <div className="Basket-item-info">
                                <h3>{goods.categories[item.ProductID - 1].title}</h3>
                                <p>Цена: {goods.categories[item.ProductID - 1].price} р.</p>
                                <button onClick={() => handleRemove(item)}>Удалить</button>
                            </div>
                        </div>
                    ))}
                    <div className='cheakout-div'>
                        <button className="checkout-button"
onClick={handleCheckout}>Оформить заказ</button>
                    </div>
                </div>
            )}
        </div>
    );
}

app.post('/api/shopping-cart/add', async (req, res) => {
    const { email, productID, quantity } = req.body;

    const getUserIDQuery = `SELECT id FROM user WHERE email = ?`;
    mysqlConnection.query(getUserIDQuery, email, (err, userResult) => {
        if (err) {
            res.status(500).send('Internal Server Error');
            console.log(err);
            return;
        }
        if (userResult.length === 0) {
            res.status(404).send('User not found');
            return;
        }
        const userID = userResult[0].id;
        console.log([userID, productID, quantity])

        const addToCartQuery = `INSERT INTO shoppingcart (UserID, ProductID, Quantity)
VALUES (?, ?, ?)`;
        mysqlConnection.query(addToCartQuery, [userID, productID, quantity], (err, result) => {
```

```
            if (err) {
                res.status(500).send('Internal Server Error');
                console.log(err);
            }
            res.status(200).send('Item added to cart');
        });
    });
});

app.delete('/api/shopping-cart/:email/:itemID', async (req, res) => {
    const { email, itemID } = req.params;
    console.log(email);

    const getUserIDQuery = `SELECT id FROM user WHERE email = ?`;
    mysqlConnection.query(getUserIDQuery, email, (err, userResult) => {
        if (err) {
            res.status(500).send('Internal Server Error');
            throw err;
        }
        if (userResult.length === 0) {
            res.status(404).send('User not found');
            return;
        }
        const userID = userResult[0].id;
        const removeFromCartQuery = `DELETE FROM ShoppingCart WHERE UserID = ?
AND ItemID = ?`;
        mysqlConnection.query(removeFromCartQuery, [userID, itemID], (err, result) => {
            if (err) {
                res.status(500).send('Internal Server Error');
                console.log(err);
                throw err;
            }
            res.status(200).send('Item removed from cart');
        });
    });
});
export default Basket;
```

Вывод: После добавления функций фильтрации и процесса покупки приложение стало более функциональным.