САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2: Взаимодействие с внешним API

Выполнила:

Мартьянова Ольга Артемьевна

K3344d

Проверил:
Добряков Д. И.

Санкт-Петербург

2023 г.

Задача

Необходимо привязать сайт, который был сверстан в лабораторной работе №1, к внешнему API средствами fetch/axios/xhr

Ход работы

Шаг 1. В начале выполнялся поиск подходящего внешнего API для реализации калькулятора калорий. Наиболее оптимальным вариантом показался https://calorieninjas.com/. Так как он поддерживает только англоязычные запросы, то сайт был переведен на английский язык.

Часть работы с авторизацией и сохранением данных пользователя была реализована с помощью json-server. Код регистрации пользователя:

```
server.post("/users", (req, res) => {
    const password = req.body.password1;
    const email = req.body.email;
    const existingUser = router.db.get('users').find({ email }).value();

    if (existingUser) {
        res.status(400).json({ er: "User with this email already exist" });
        return;
    } else {
        bcrypt.genSalt(saltRounds, function (err, salt) {
            bcrypt.hash(password, salt, function (err, hash) {
                const newUser = {
                    id: Date.now(),
                    name: req.body.name,
                    email: req.body.email,
                    password: hash,
                    gender: req.body.gender,
                    height: req.body.height,
                    weight: req.body.weight,
                    dateOfBirth: req.body.dateOfBirth,
                    favorites: [],
                    daily_rations: [],
                };
                router.db.get("users").push(newUser).write();

                if (err) {
                    console.log(err);
                    res.status(400).json({ er: err });
                } else {
                    const token = jwt.sign({ email }, secretKey, { expiresIn: '1h' });
                    res.json({ token });
                }
            })
        })
    }
});
```

При этом пароли шифруются с помощью bcrypt, а также проверяется не зарегистрирован пользователь с таким же адресом почты.

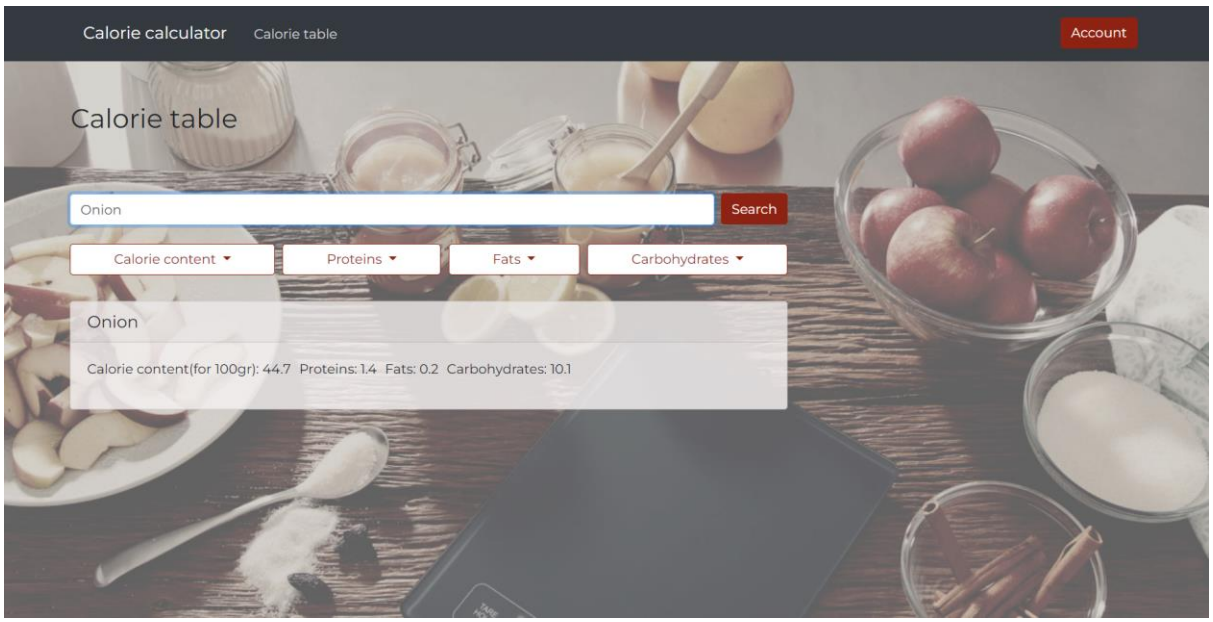
Код авторизации:

```
server.post('/login', (req, res) => {
    const email = req.body.email;

    const user = router.db
        .get('users')
        .find({ email })
        .value();

    if (user) {
        bcrypt.compare(req.body.password, user.password, function (err, result) {
            if (err) {
                res.status(401).json({ msg: "Error to compare passwords", er: err });
            } else if (result) {
                const token = jwt.sign({ email }, secretKey, { expiresIn: '1h' });
                res.json({ token });
            } else {
                res.status(401).json({ msg: "Passwords doesn't match" })
            }
        });
    } else {
        res.status(401).json({ message: 'User not found' });
    }
});
```

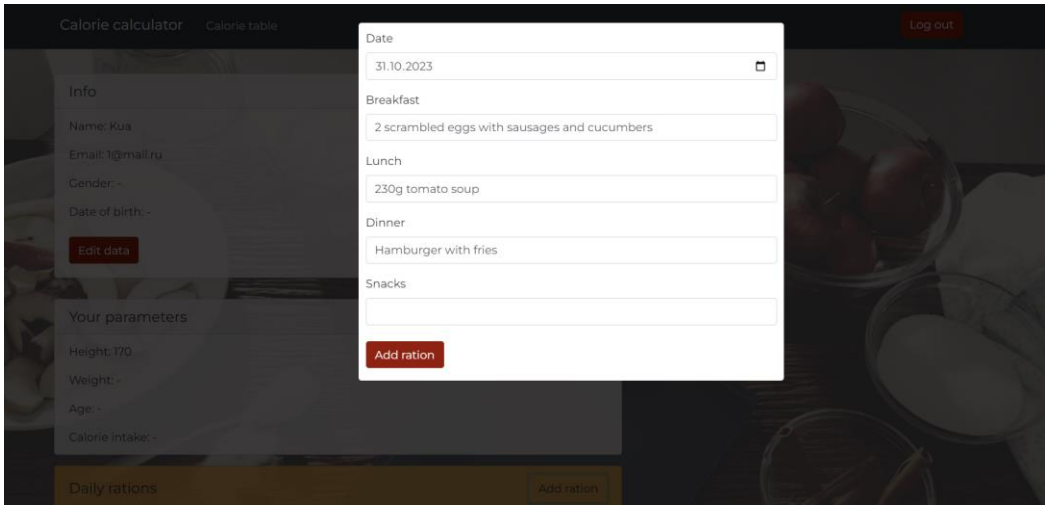
Были реализованы еще несколько методов, позволяющие сайту полноценно функционировать.

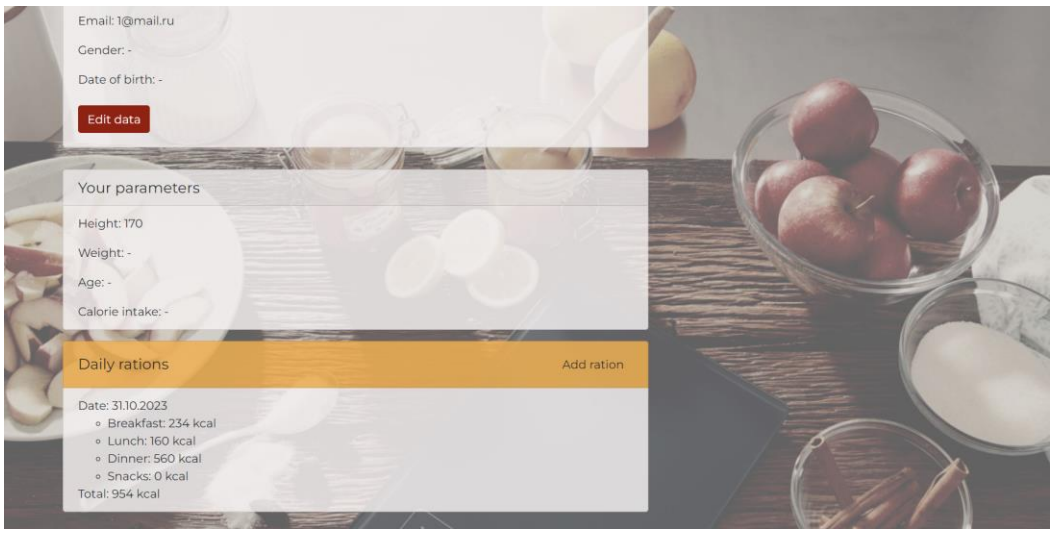
Шаг 2. Привязка к API.

Связь с внешним API выполнялась с помощью axios. Теперь пользователь в поиске может находить необходимые продукты и получать информацию о их энергетической ценности:



Также есть возможность рассчитать количество калорий исходя из потребленной за день пищи:





Дополнительно появилась возможность добавлять продукты в избранное. Список избранных продуктов отображается в “Calorie table”, где их можно фильтровать по количеству калорий, белков, жиров и углеводов.

Код получения данных из внешнего API:

```
async function fetchData(productNames) {
    try {
        let query = productNames.join(' ');
        const response = await axios.get(`https://api.calorieninjas.com/v1/nutrition?query=${query}`, {
            headers: {
                'X-Api-Key': 'yNHRx+yRMMHLnN907h/kNw==sYBu7FD1a0LtVn8P'
            }
        });

        if (response.status === 200) {
            const productData = response.data.items;
            return productData;
        } else {
            throw new Error(`Unable to fetch data. Status^ ${response.status}`);
        }
    } catch (error) {
        throw new Error(`Error to request API: ${error.message}`);
    }
}
```

Получение списка избранных продуктов:

```
async function getUser() {
    const userToken = localStorage.getItem('userToken');
    if (userToken) {
        try {
            const response = await axios.get('http://localhost:3000/users', {
                headers: { 'Authorization': `Bearer ${userToken}`, },
            });
            if (response.status === 200) {
                return response.data;
            } else {
                console.log(response.data);
                alert("Error to get user");
                return;
            }
        } catch (error) {
            console.error(error);
            return;
        }
    }
}

async function getFavorites() {
    const user = await getUser();
    return user.favorites;
}
```

Формирование калорийности дневного рациона:

```
function getListHTML(daily_rations) {
  var list = "";
  const options = { year: 'numeric', month: '2-digit', day: '2-digit' };
  daily_rations.forEach(ration => {
    const date = new Date(ration.date);
    var formattedDate = new Intl.DateTimeFormat('ru-RU', options).format(date);
    list += `<li class="list-group-item">
      Date: ${formattedDate}
      <ul>
        <li>Breakfast: ${ration.breakfast} kcal</li>
        <li>Lunch: ${ration.lunch} kcal</li>
        <li>Dinner: ${ration.dinner} kcal</li>
        <li>Snacks: ${ration.snacks} kcal</li>
      </ul>
      Total: ${ration.breakfast + ration.lunch + ration.dinner + ration.snacks} kcal
    </li>
`
  });
  return list;
}

const addForm = document.getElementById('addForm');

async function fetchDailyRations(query) {
  try {
    const response = await axios.get(`https://api.calorieninjas.com/v1/nutrition?query=${query}`, {
      headers: {
        'X-Api-Key': 'yNHRx+yRMMHLnN907h/kNw==sYBu7FD1a0LtVn8P'
      }
    });
    if (response.status === 200) {
      const productData = response.data.items;
      return productData;
    } else {
      throw new Error(`Unable to fetch data. Status^ ${response.status}`);
    }
  } catch (error) {
    throw new Error(`Error to request API: ${error.message}`);
  }
}
```

Регистрация пользователей и валидация данных:

```
const form = document.getElementById("registrationForm");

form.addEventListener("submit", function (event) {
    event.preventDefault();
    const formData = new FormData(form);
    const userData = {};
    formData.forEach((value, key) => {
        if (key === 'height' || key === 'weight') {
            if (typeof value === 'string') {
                userData[key] = parseFloat(value);
            } else {
                alert('Please enter numeric value for "${key}"');
                return;
            }
        } else if (key === 'dateOfBirth') {
            if (typeof value === 'string') {
                userData[key] = value;
            } else {
                alert('Please enter correct date of birth');
                return;
            }
        }
        userData[key] = value;
    });
    if (!userData.name || !userData.email || !userData.password1 || !userData.password2) {
        alert('Please fill reqiered information');
        return;
    }
    if (userData.password1.length < 6) {
        alert('Password should be longer than 6 symbols');
        return;
    }
    if (userData.password1 != userData.password2) {
        alert("Passwords don't match");
        return;
    }
    axios.post('http://localhost:3000/users', userData, {
        headers: { 'Content-Type': 'application/json', },
    })
        .then(response => {
            if (response.status === 200) {
                setUserLoggedIn(true);
                localStorage.setItem('userToken', response.data.token);
                window.location.href = "../account/account.html";
            } else {
                console.log(response.status);
                alert("Registration failed");
            }
        })
        .catch(error => {
            console.error('Error login user:', error);
            alert("Registration failed");
        })
});
```

Вывод

В ходе данной работы была выполнена привязка сайта к внешнему API так, что теперь он является функциональным и есть возможность полноценно его использовать.