МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ государственное БЮДЖЕТНОЕ

образовательное учреждение

высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации

**

**ЛАБОРАТОРНАЯ РАБОТА №1**

по дисциплине: «*Технологии и методы программирования*»

|  |  |
| --- | --- |
| Выполнил:  Студент гр. «АБ-320», «АВТФ»  *Мартынова Е.Д.*  «\_\_\_» \_\_\_\_\_\_ 2025г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | Проверил:  *Старший преподаватель кафедры ЗИ*  *Медведев М.А.*  «\_\_\_» \_\_\_\_\_\_ 2025г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Новосибирск 2025

Цель работы:

1. Освоить процесс разработки одностраничного приложения (SPA) с помощью современного фреймворка или библиотеки (React/Vue/Angular).

2. Закрепить на практике концепции RESTful API: отправка запросов (GET/POST/PUT/DELETE), обработка JSON-ответов.

3. Реализовать базовый роутинг без перезагрузки страницы, управление состоянием и динамическую отрисовку интерфейса.

Задачи:

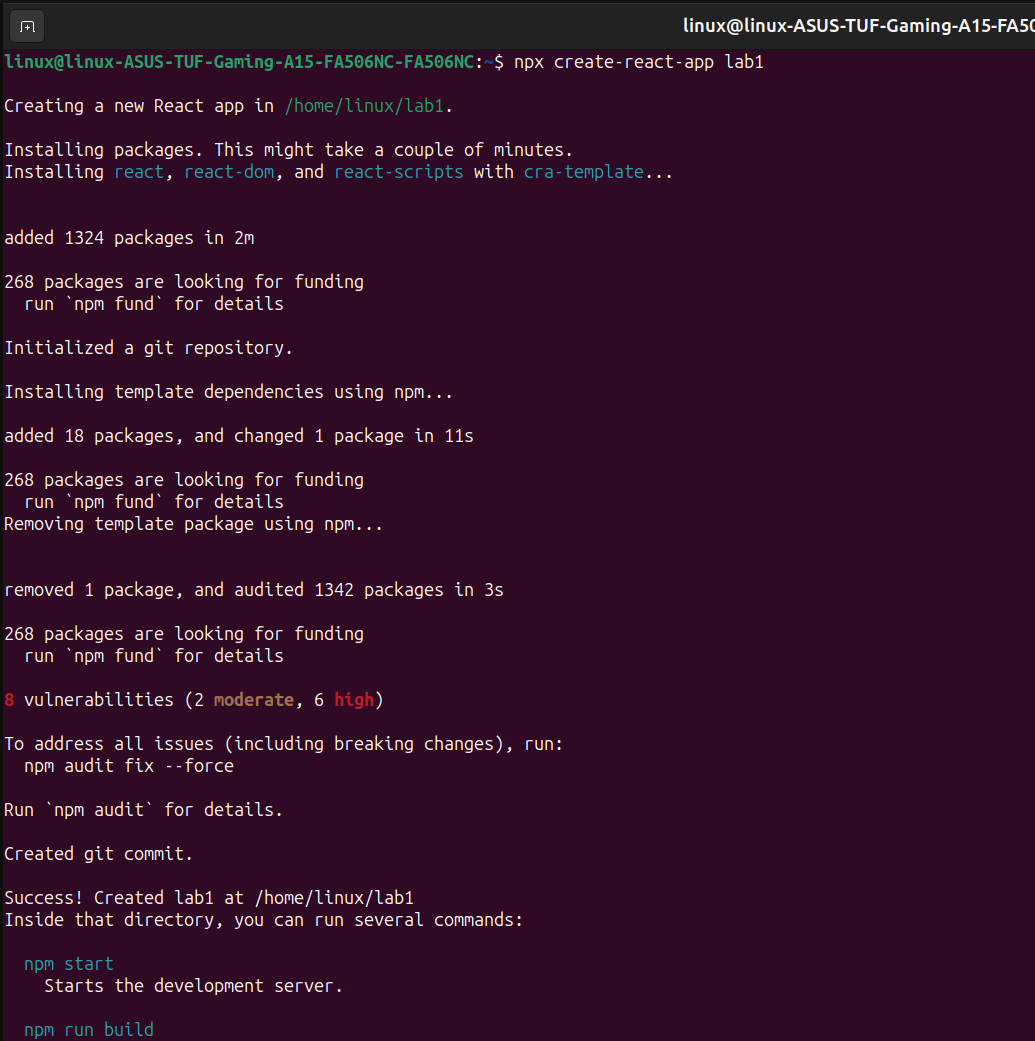
1. Подготовить проект по предметной области

2. Составить структуру и роутинг SPA

3. Организовать связь с сервером (RESTful API)

4. Провести работу с JSON-данными

**Задание 1**

Создадим новый проект «React» с помощью «Create React App»:Рисунок 1 – Создание проекта «SPA»

Запустим проект на порту 3000:

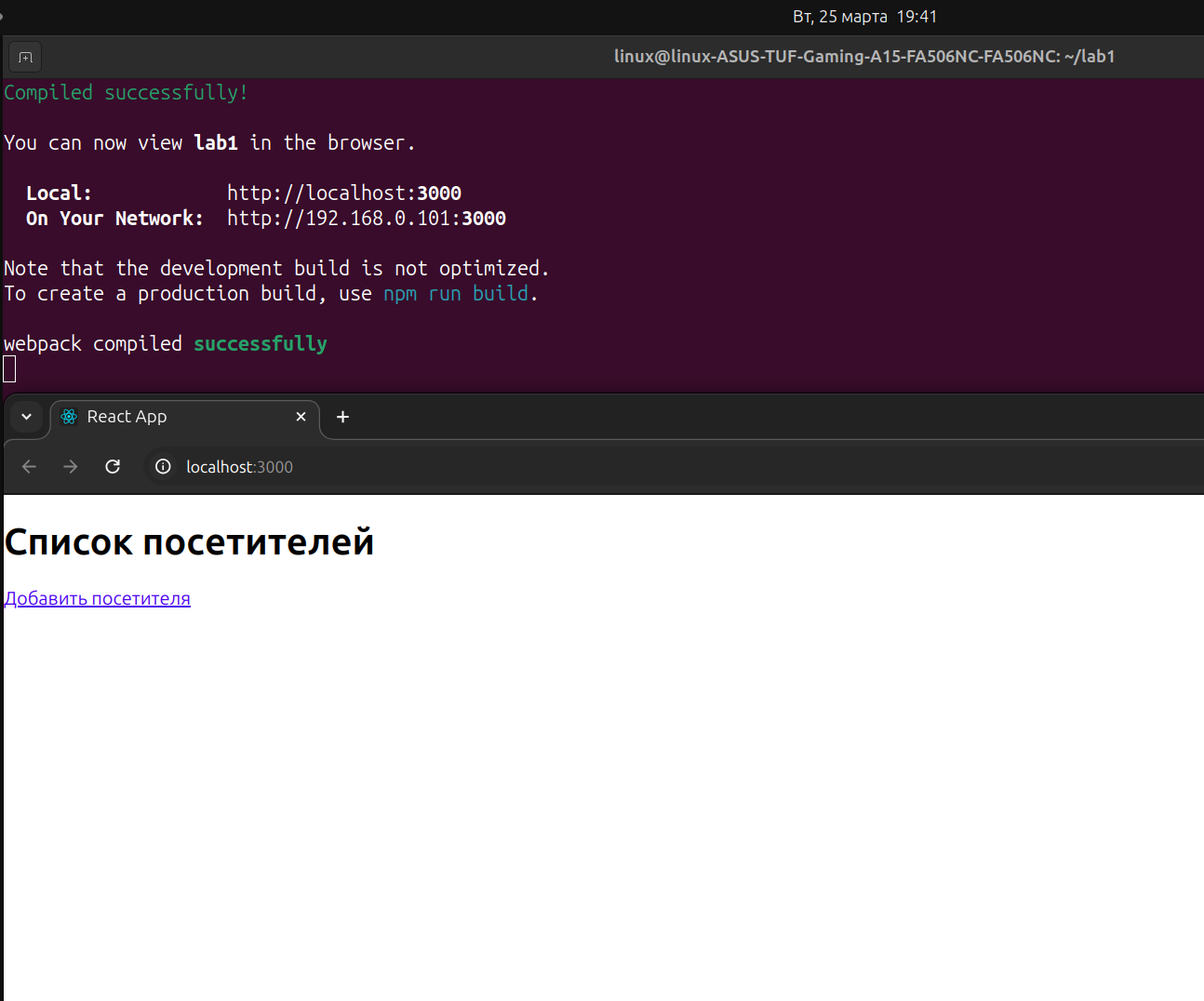


Рисунок 2 – Запуск проекта

# Задание 2

Создадим три основные страницы сайта на языке «JavaScript»:

* «Home» – начальная страница, отображающая список сущностей
* «Detail» – загрузка данных о сущности и редактирование
* «Form» – добавление новых сущностей

Полный код представлен в приложении 1.

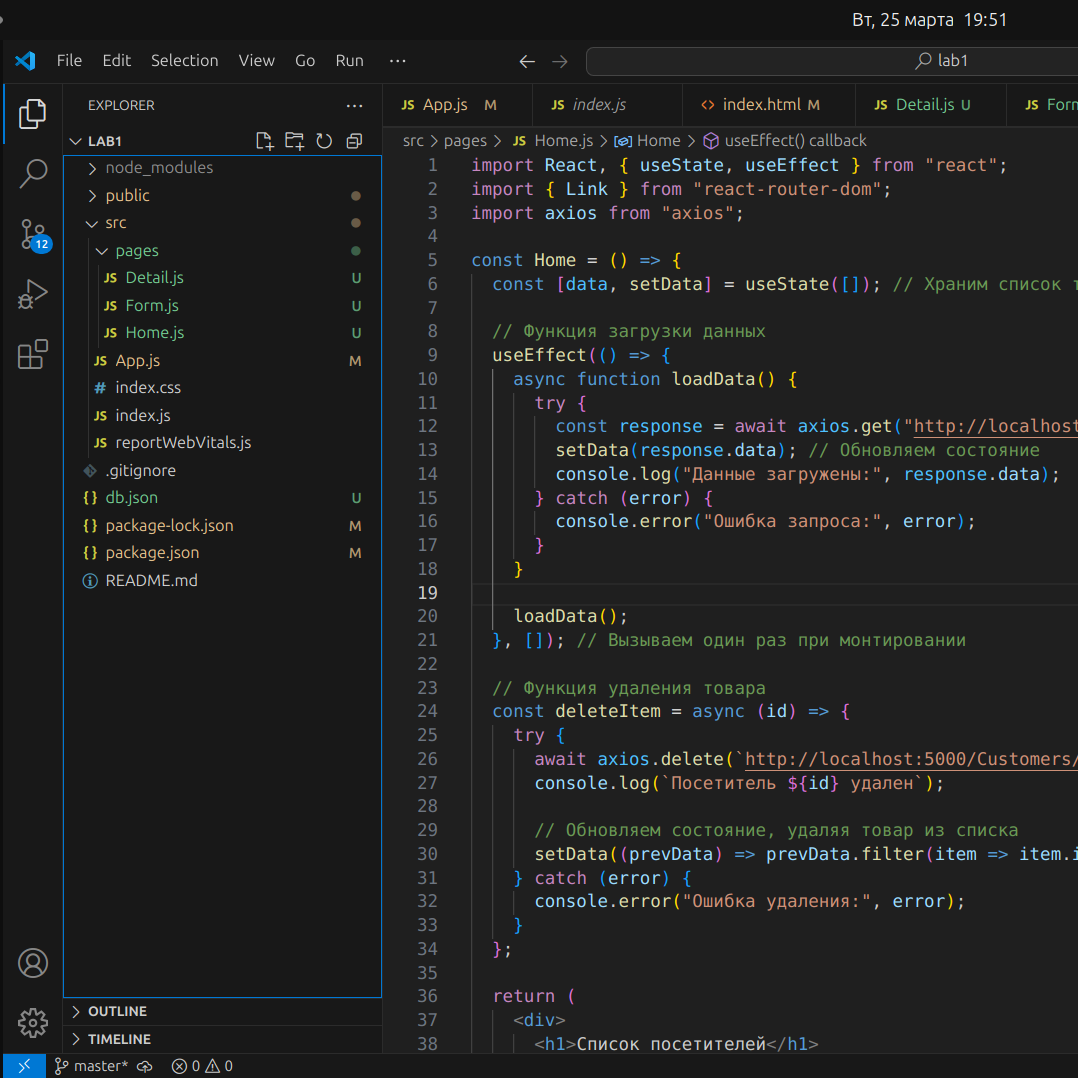


Рисунок 3 – Отображение созданных «.js» файлов

Настроим роутинг, изменив содержание файла «App.js»:

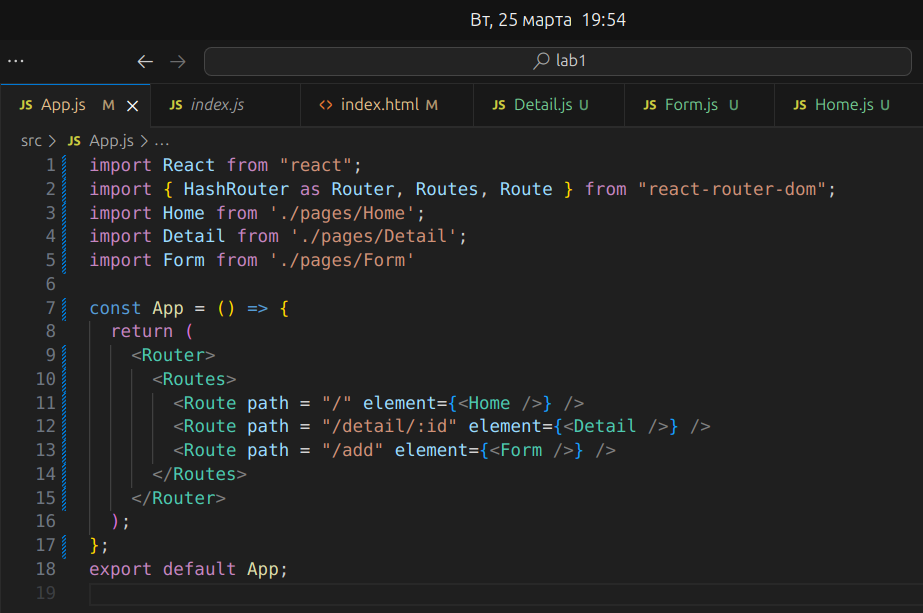


Рисунок 4 – Содержание файла «App.js»

# Задание 3

Создадим json-файл с базой данных сущностей:



Рисунок 5 – Содержание файла «db.json»

Запустим сервер на порту 5000:

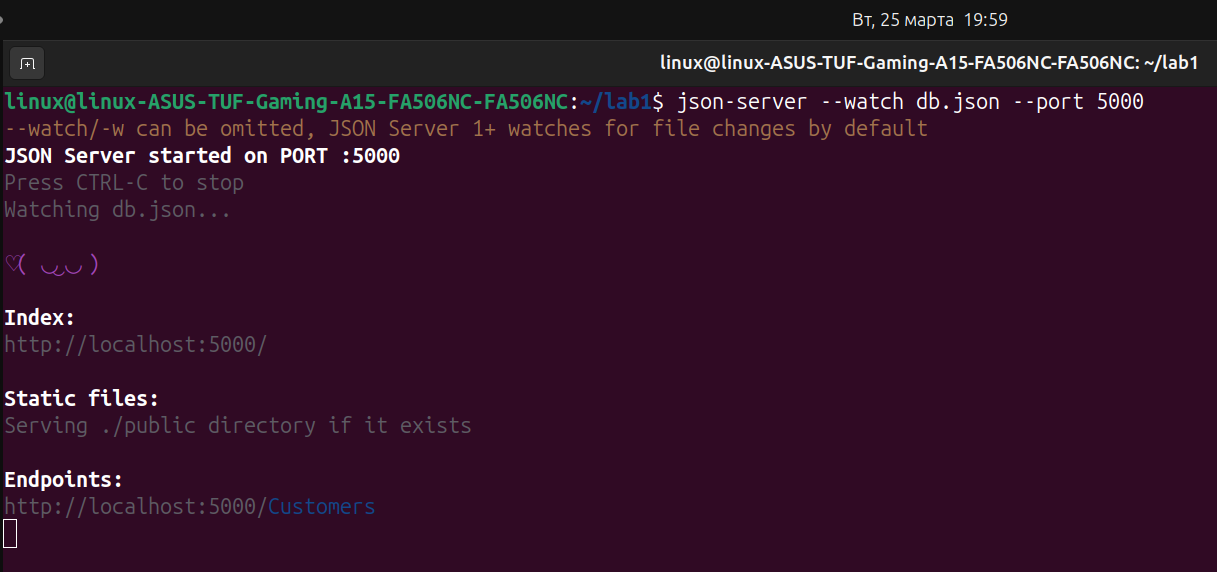


Рисунок 6 – Запуск json-сервера

Проверим работоспособность сервера с помощью стороннего программного обеспечения «Postman», отправив GET-запрос:

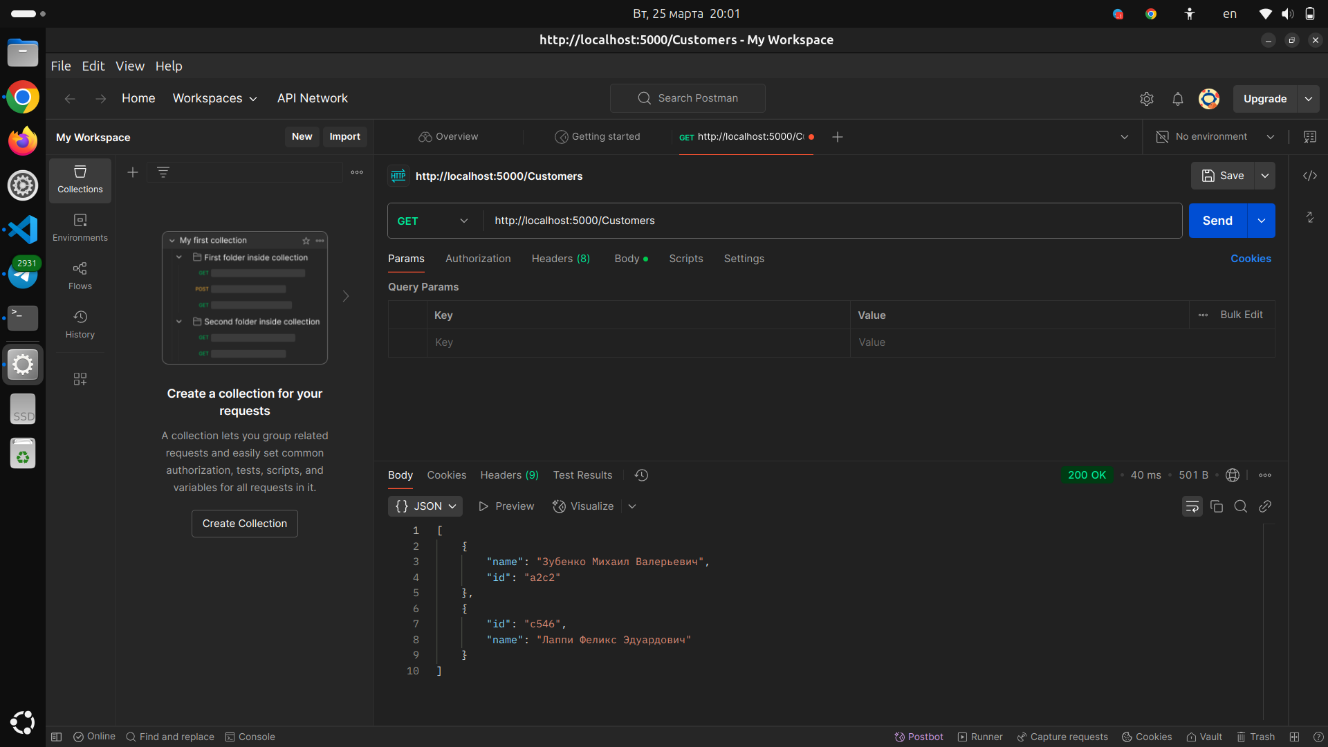


Рисунок 7 – Отправка GET-запроса на сервер через «Postman»

Реализация запросов находится в файлах:

* GET – «Home.js»
* POST – «Form.js»
* PUT – «Detail.js»
* DELETE – «Home.js»

Полный код представлен в приложении 1.

# Задание 4

Json-ответ от сервера преобразуется в объект «JavaScript» в 13 строке с помощью инструмента «axios»

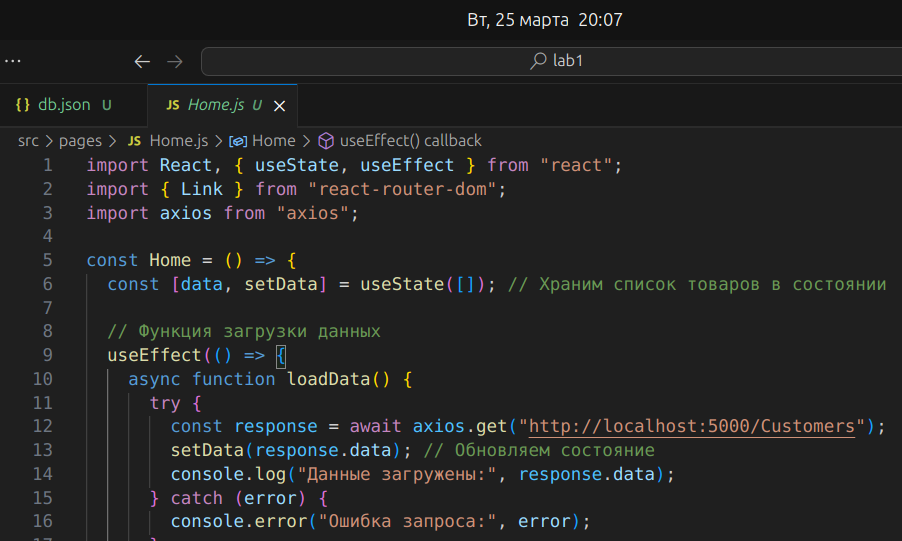


Рисунок 8 – Содержание файла «Home.js»

При создании/обновлении сущности отправляется JSON в теле запроса:

«Detail.js» - JSON отправляется в теле PUT-запроса в функции «handleSubmit», где объект «updatedItem» передаётся с заголовком «Content-Type»: «application/json».

«Form.js» - Аналогично, как и в «Detail.js».

Также добавим в «db.json» поле для описания сущности и в «Detail.js» код для его редактирования:

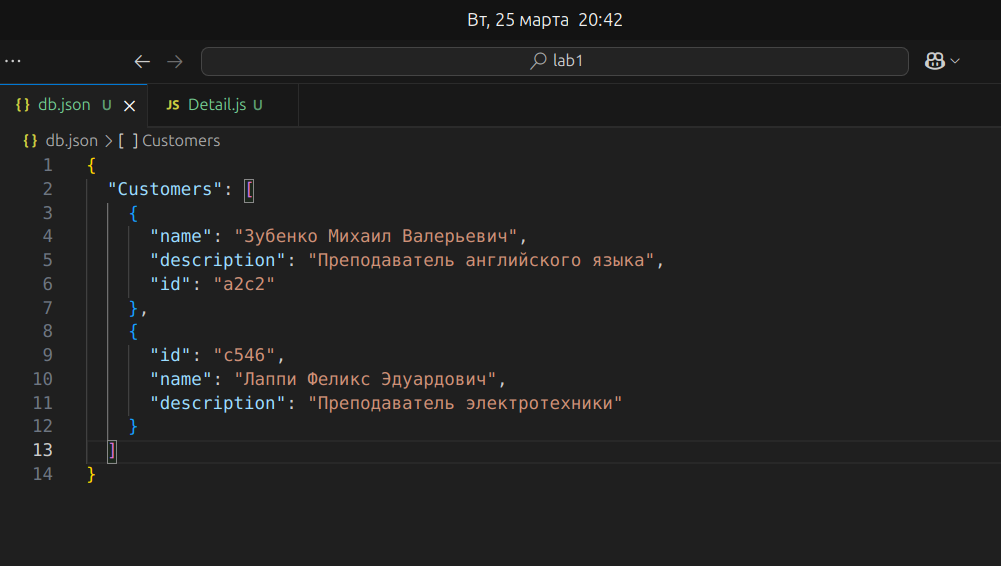


Рисунок 9 – Обновление файла «db.json»

# Задание 7

Запустим json-сервер:

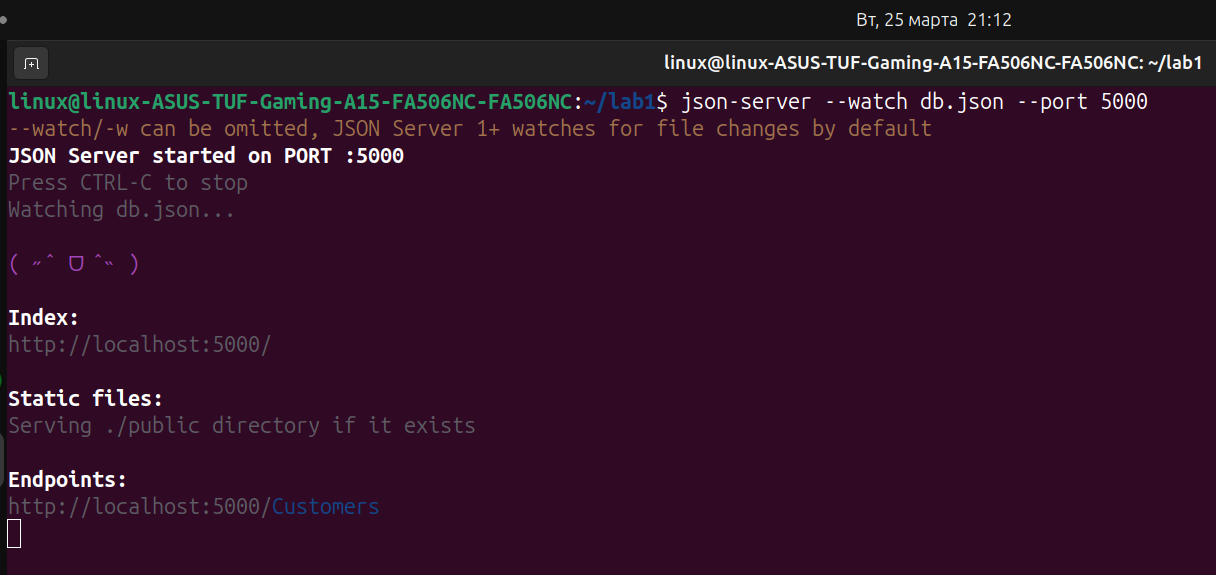


Рисунок 10 – Запуск json-сервера

Далее запустим React-приложение:

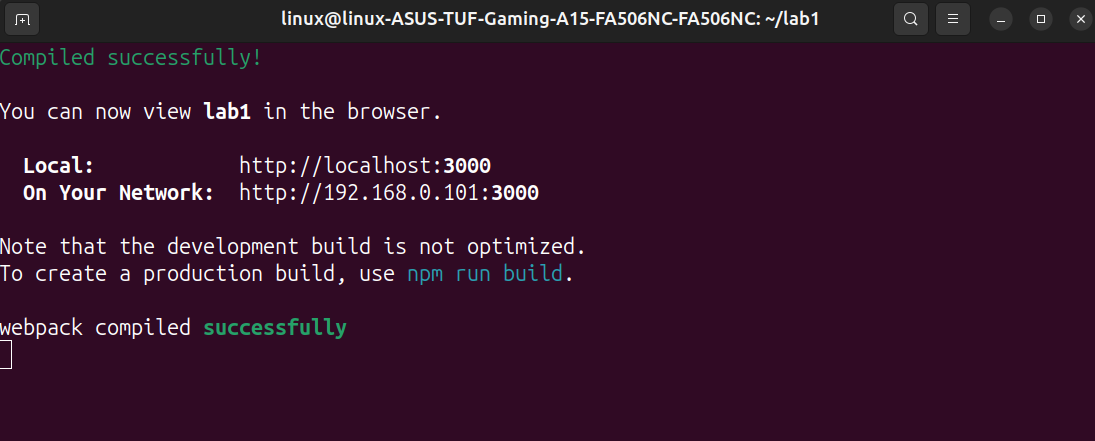


Рисунок 11 - Запуск React-приложения

Открывается главная страница со списком сущностей:

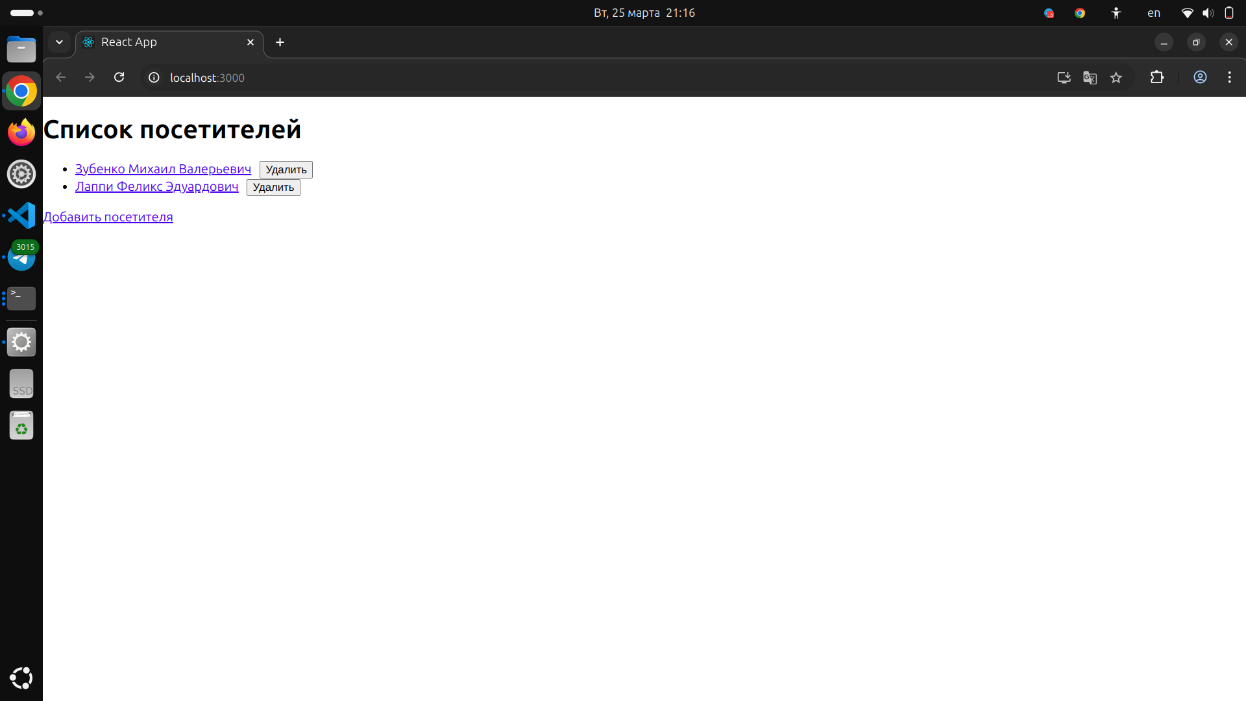


Рисунок 12 – Главная страница

Кликнем на детальную страницу первой сущности:

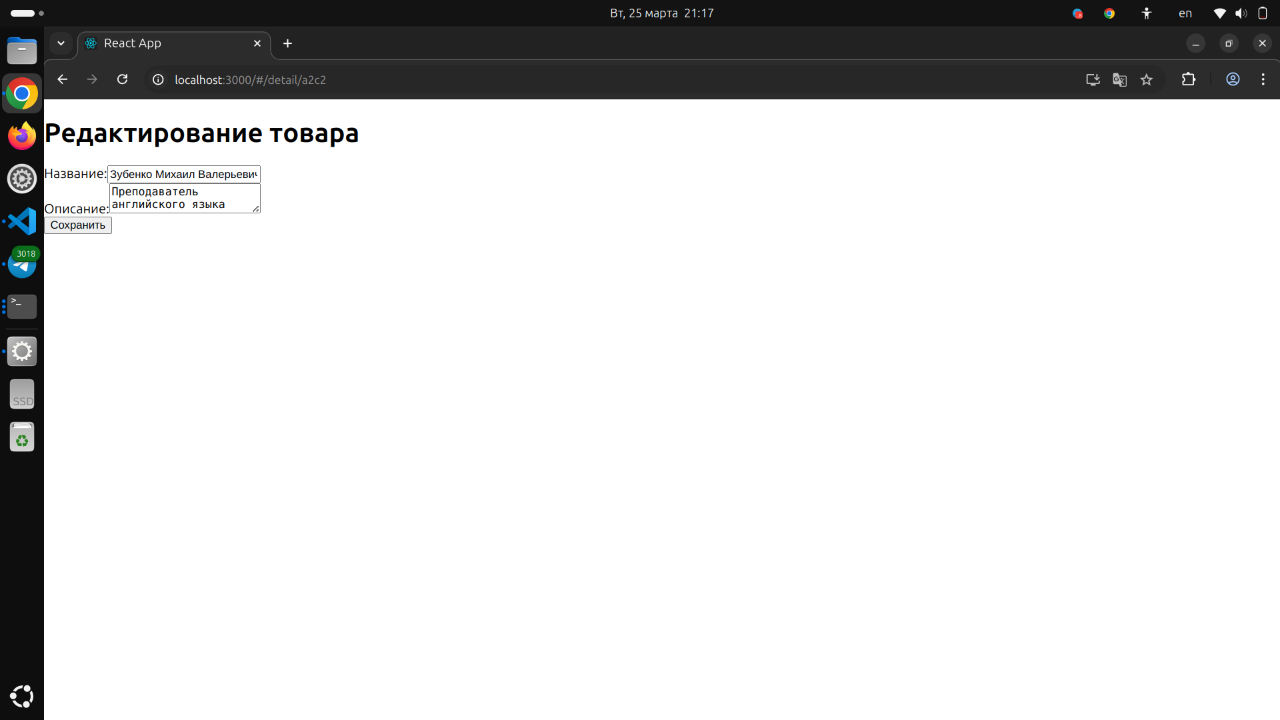


Рисунок 13 – Детальная страница

Возвращаемся на главную страницу и нажимаем кнопку «Добавить посетителя». Нас перекидывает на страницу для добавления сущности. Добавляем «Мартынова Екатерина Дмитриевна».

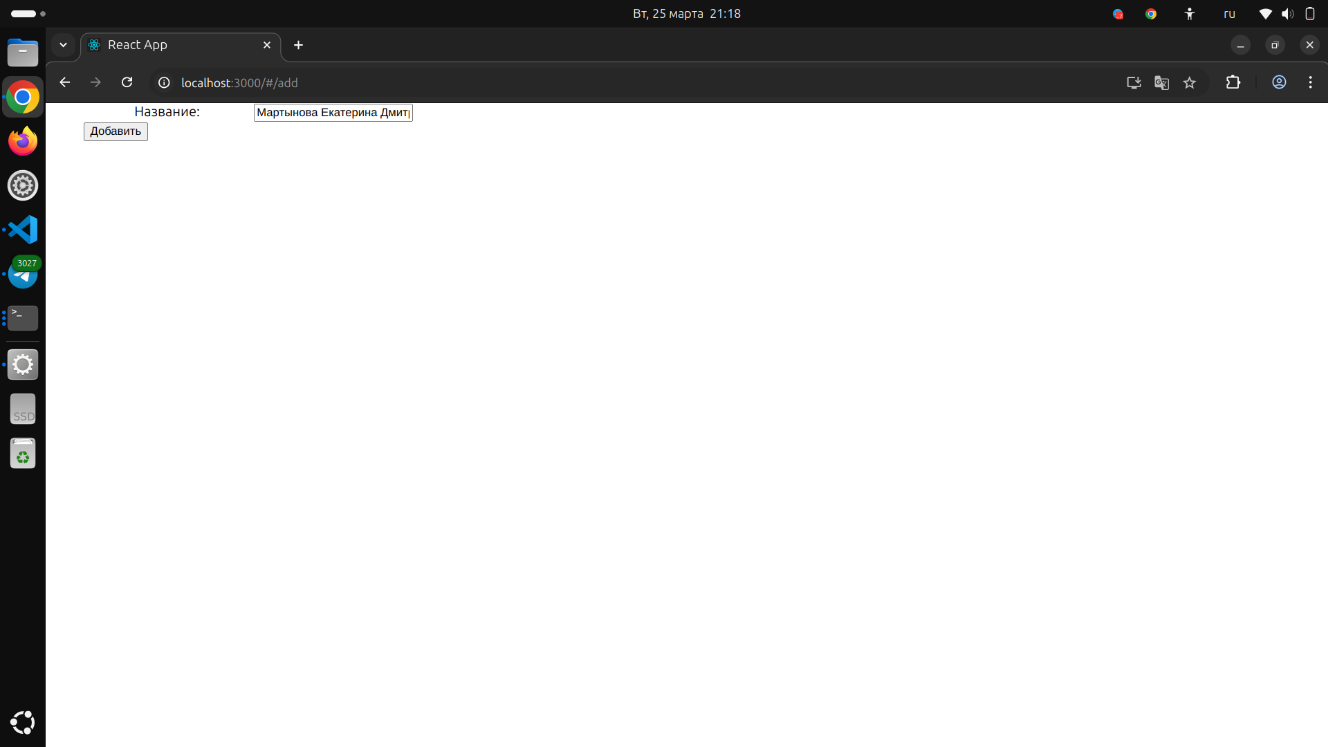


Рисунок 14 – Страница добавления сущности

После нажатия кнопки «Добавить» на главной странице обновляется список сущностей:

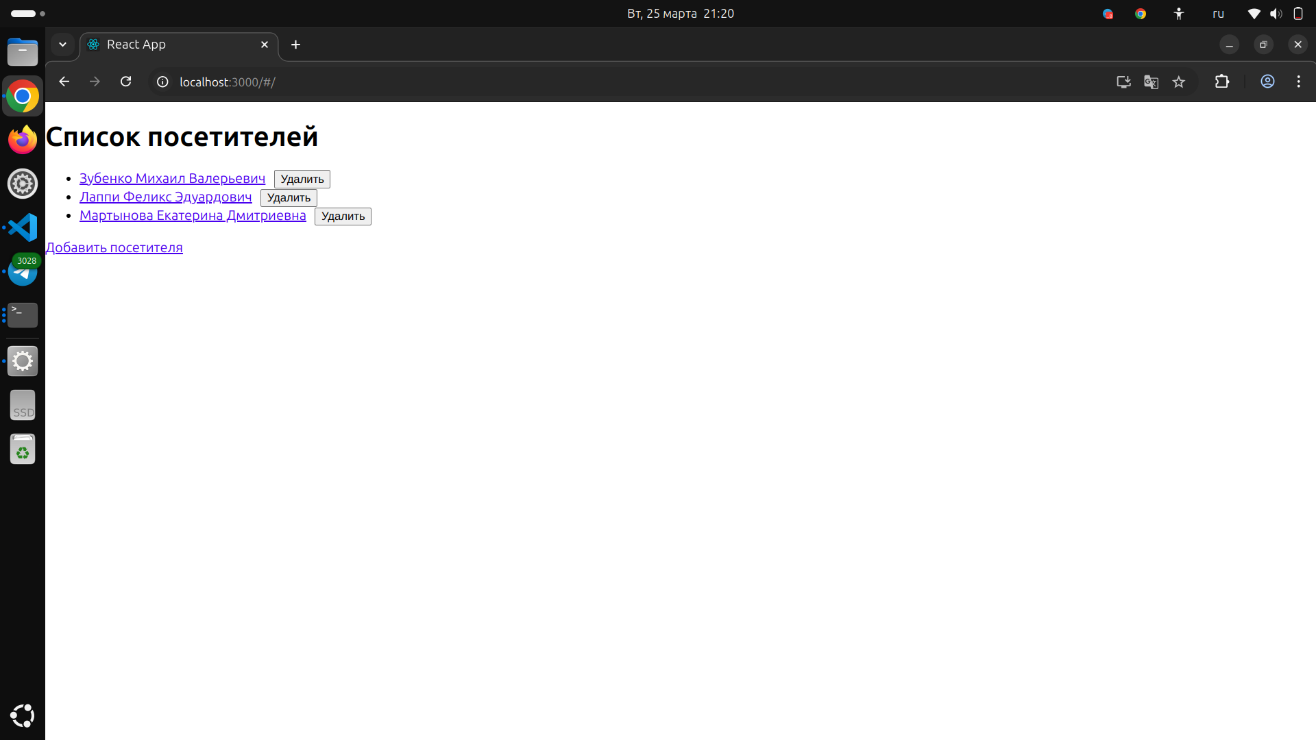


Рисунок 15 – Главная страница

Нажмем кнопку «Удалить» для последней добавленной сущности:

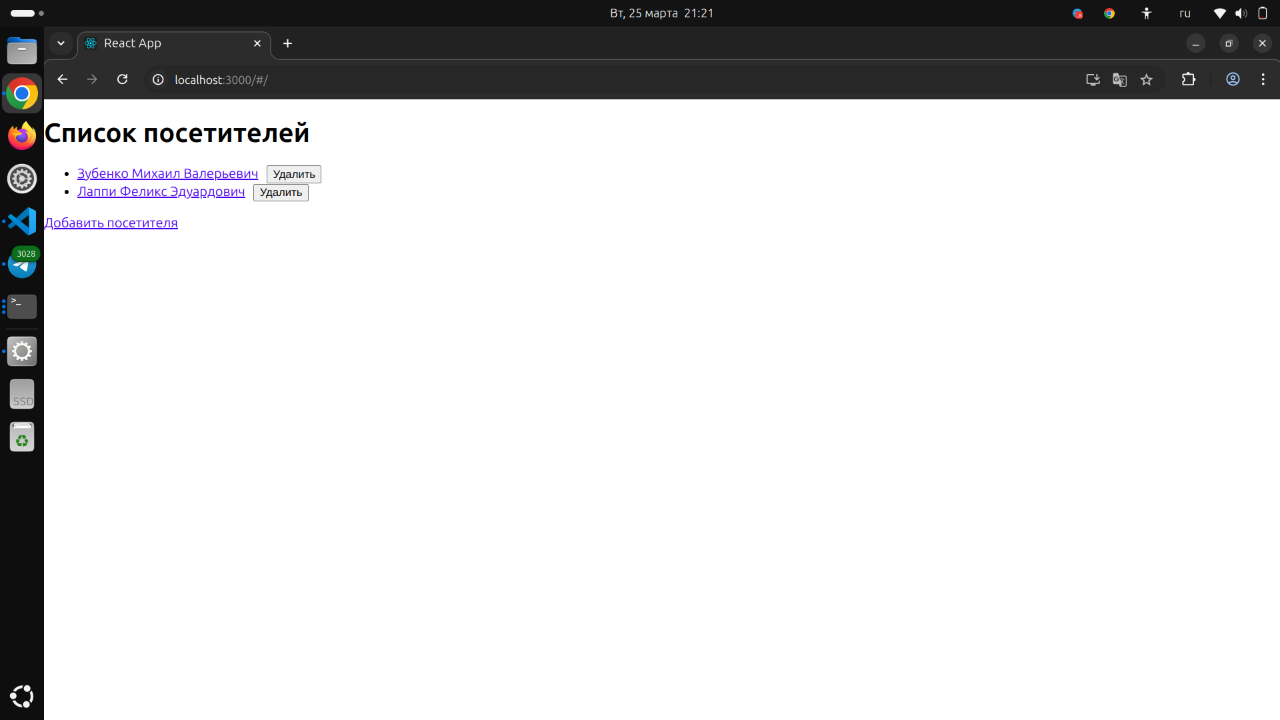


Рисунок 16 – Главная страница

Можно увидеть, что «Мартынова Екатерина Дмитриевна» была удалена из базы данных.

Открыв инструменты разработчика, убедимся, что при взаимодействии с сервером отправляются нужные запросы (GET, POST, PUT, DELETE):

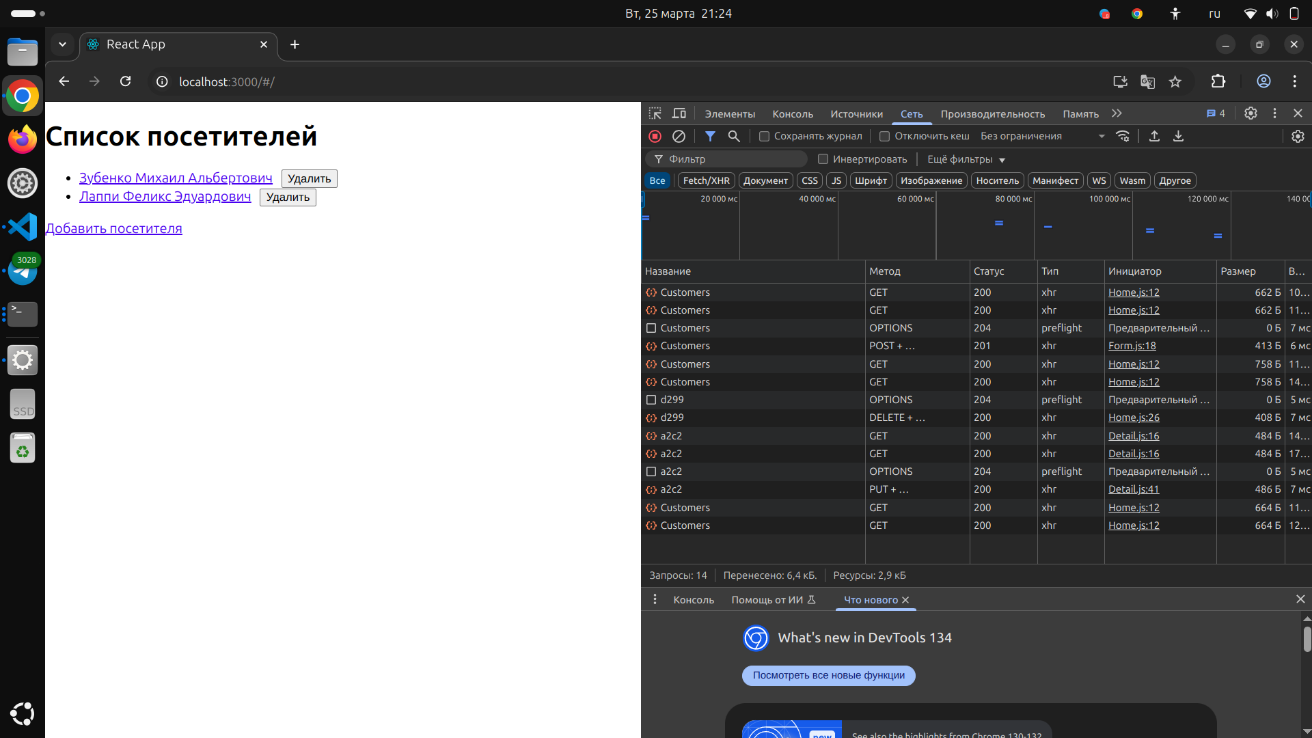


Рисунок 17 – Инструменты разработчика

Убедимся, что на GET-запрос сервер отвечает JSON:

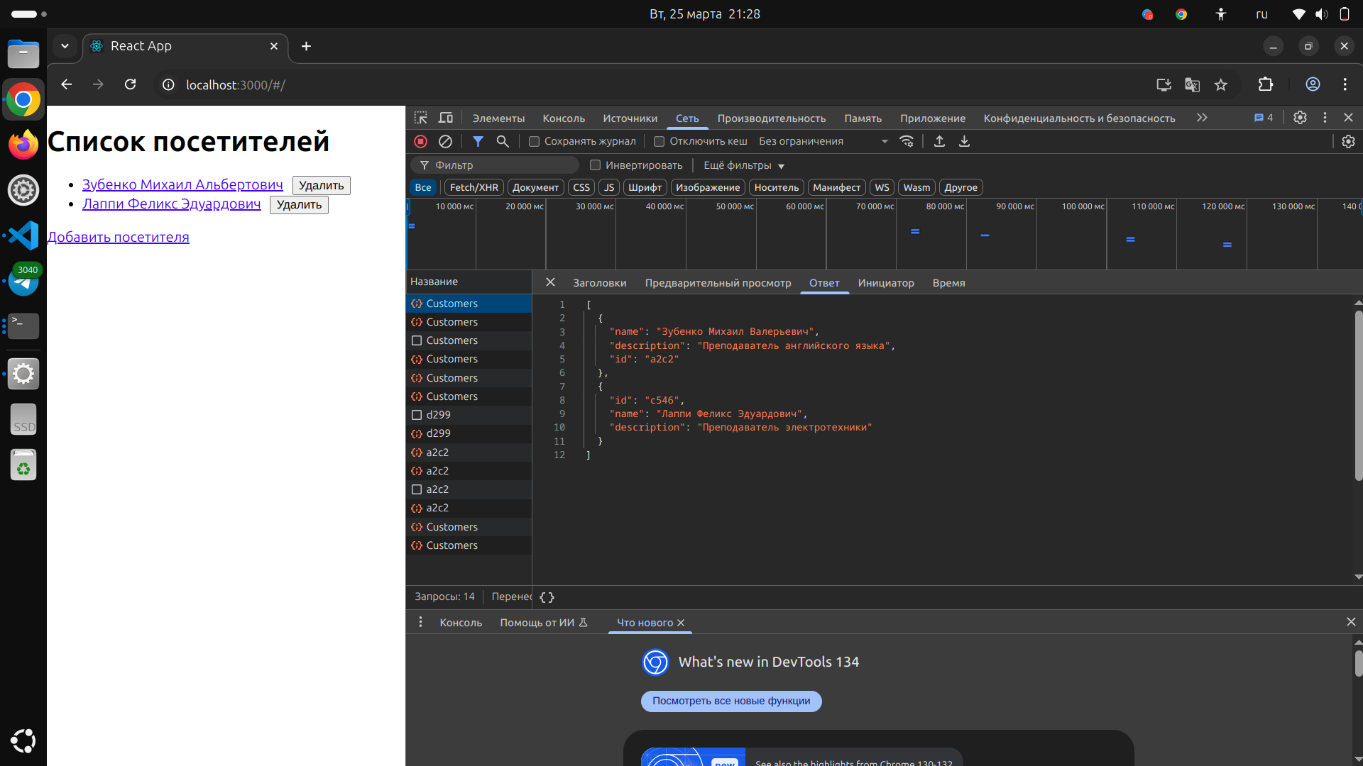


Рисунок 18 – Ответ сервера в инструментах разработчика

Далее продемонстрируем работу роутинга:

Рисунок 19 – Переход на главную страницу

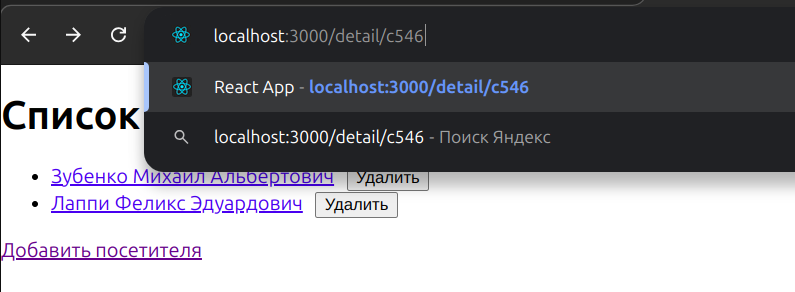
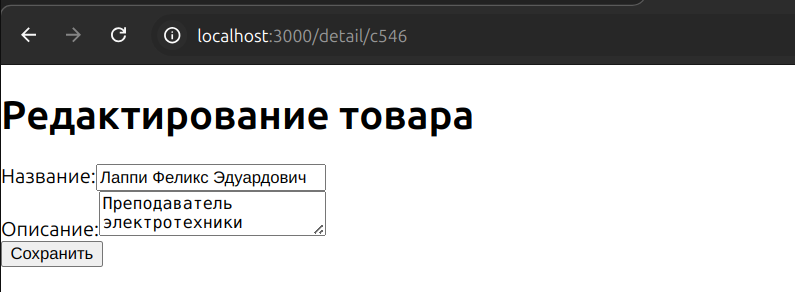
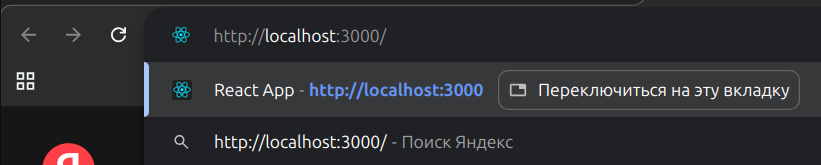
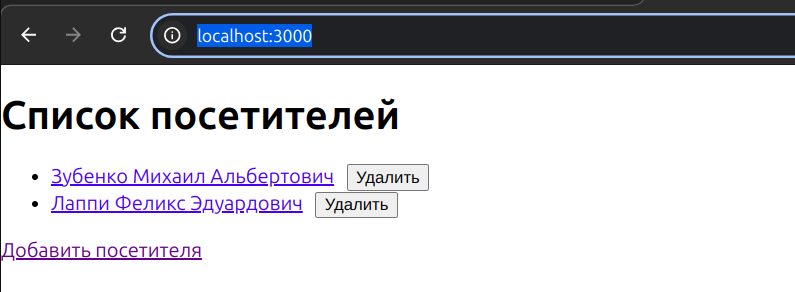


Рисунок 20 – Переход на детализацию сущности

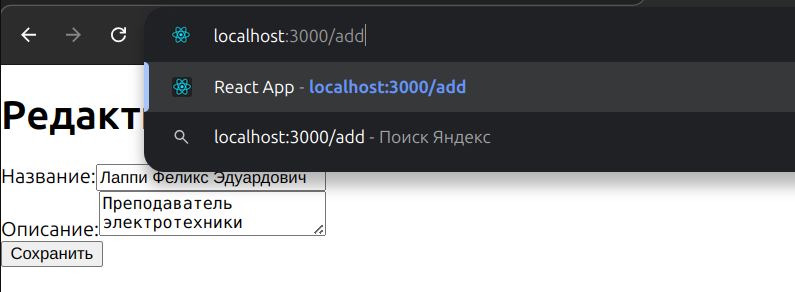
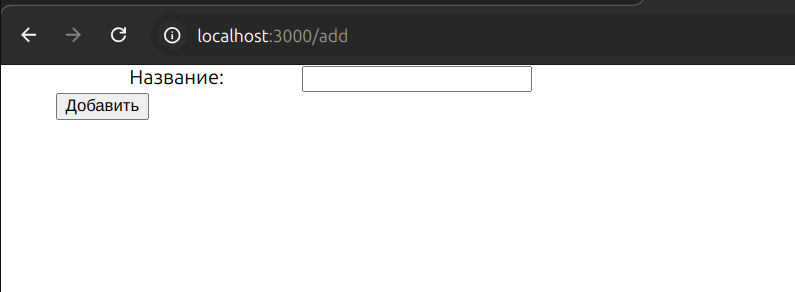


Рисунок 21 – Переход на страницу добавления сущности

Затем продемонстрируем работу CRUD-операций:

Read:

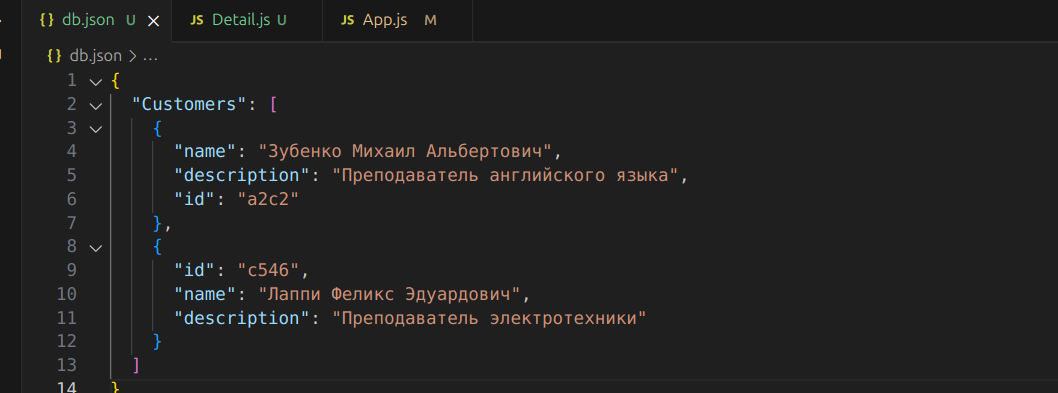
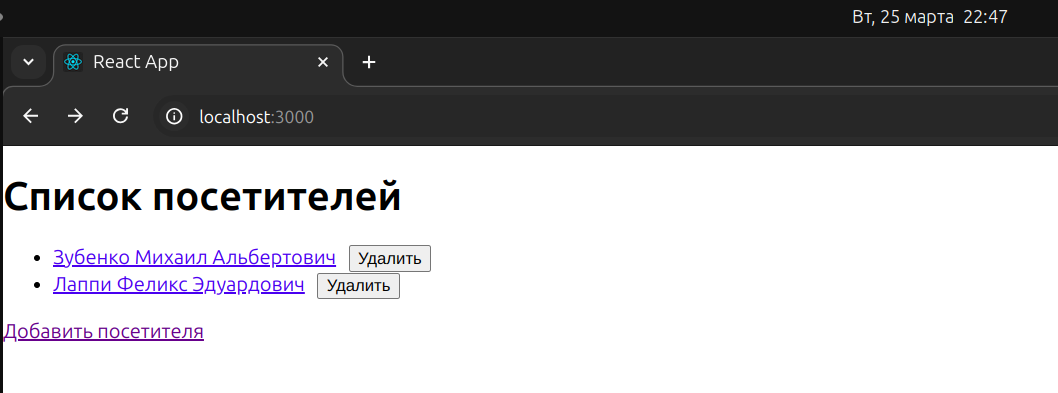


Рисунок 22 – Чтение данных

Create:

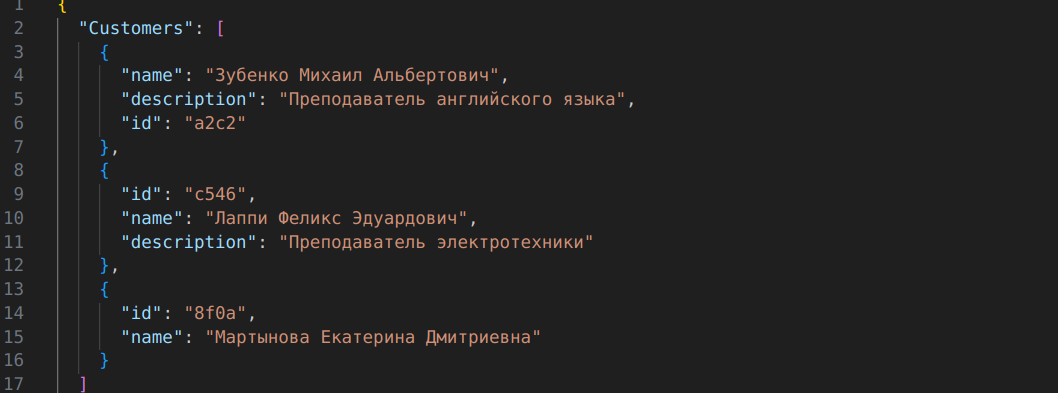
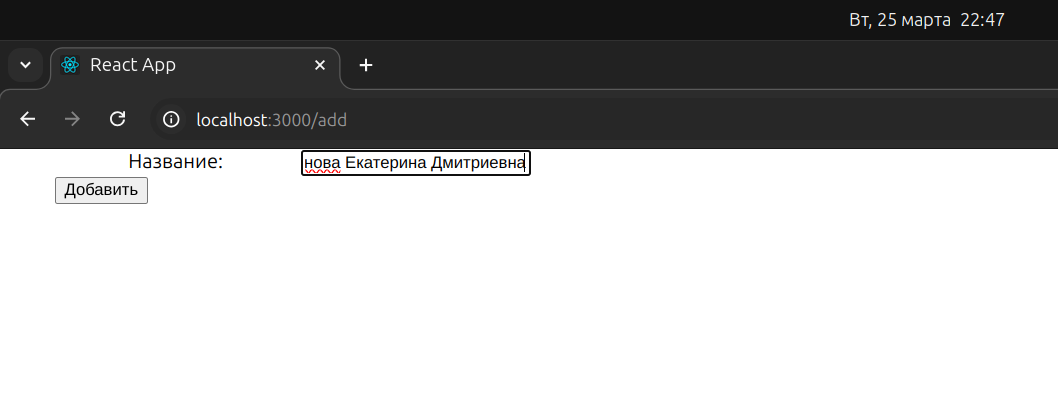


Рисунок 23 – Добавление данных

Update:

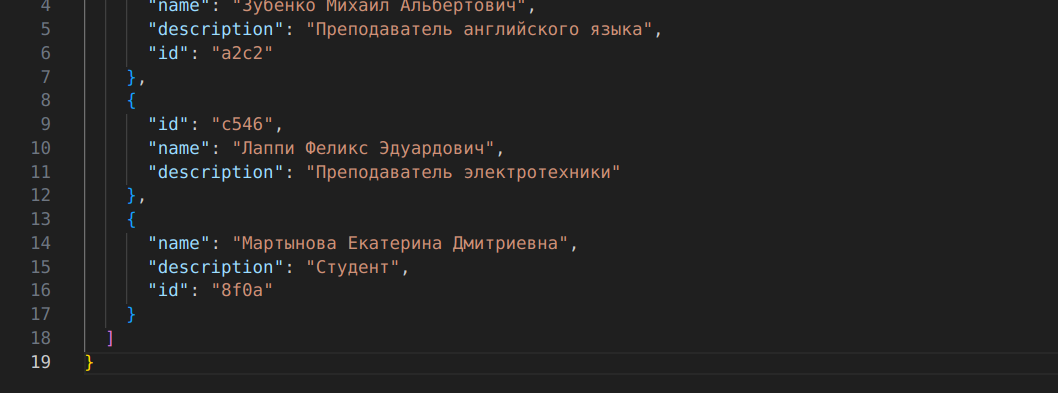
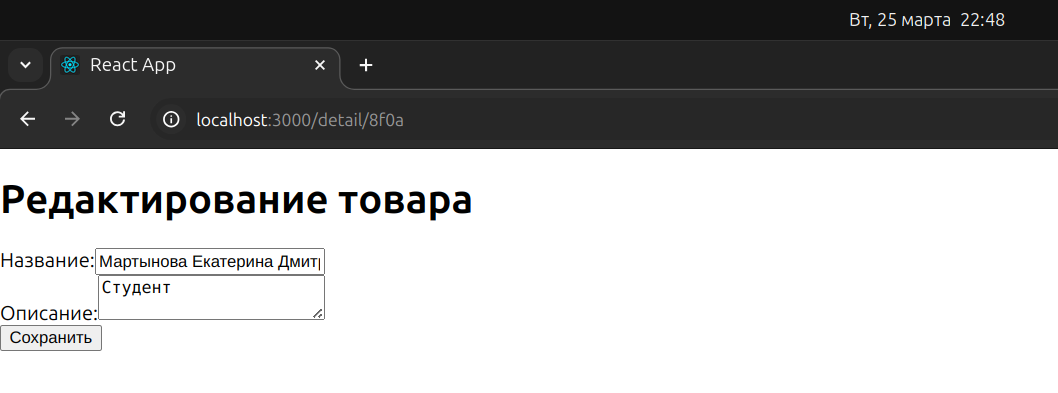


Рисунок 24 – Обновление данных

Delete:

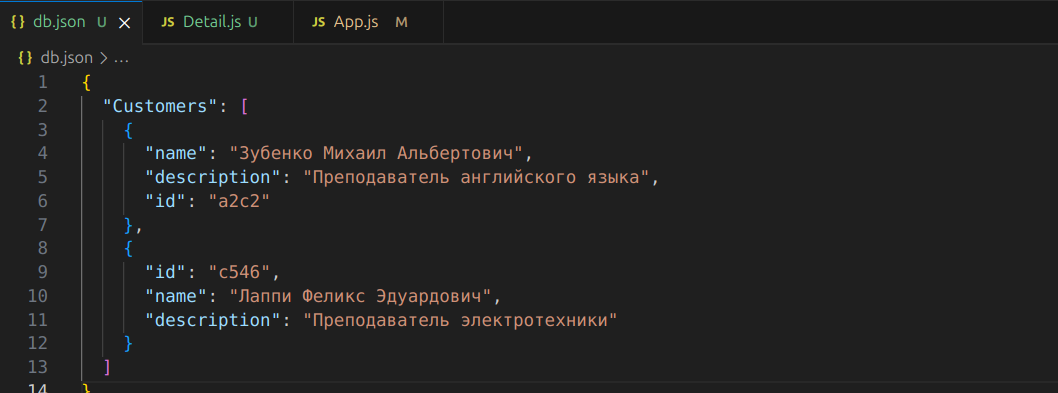
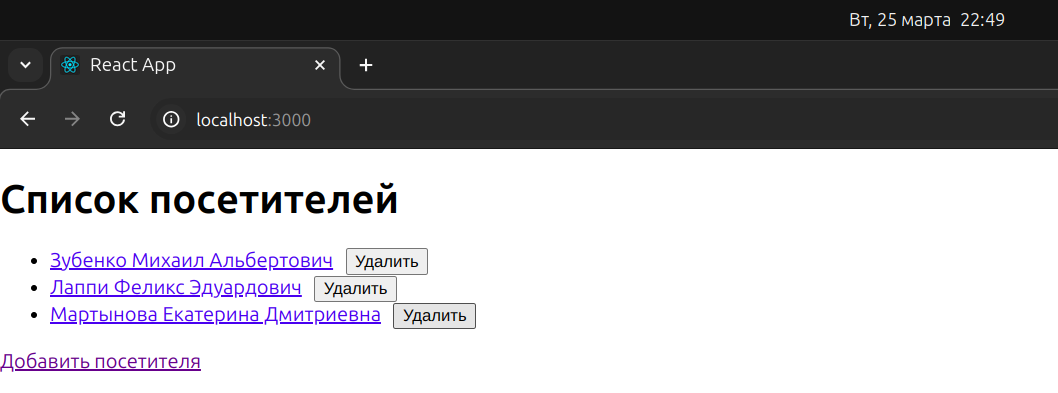


Рисунок 25 – Удаление данных

# Задание 8

В ходе выполнения работы было разработано SPA-приложение с использованием инструментов «React» и «json-server». Приложение реализует базовую работу с RESTful API, включая операции GET, POST, PUT, DELETE, и взаимодействует с сервером через JSON-данные.

Для запуска проекта локально необходимо в терминале запустить две команды:

1. json-server --watch db.json --port 5000
2. npm start

Первая команда запускает json-сервер на порту 5000, а вторая React-приложение.

Все скриншоты с демонстрацией работы роутинга, CRUD-операций и взаимодействия с сервером были предоставлены в предыдущих заданиях.

# Приложение

Приложение 1:

Ссылка на репозиторий: [Нажать](https://github.com/mamam111a/4-semester_laboratory-1)