Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий

Кафедра инструментального и прикладного программного обеспечения

Дисциплина «Разработка клиент-серверных приложений»

КУРСОВАЯ РАБОТА

Разработка и развёртывания клиент-серверного приложения для доставки продуктов

Студент: Сулейманов А.В.

Группа: ИКБО-30-22

Руководитель: ст. преп. Ермаков С.Р.

Москва 2025

Цель

Создание клиент-серверного программного приложения веб-сервиса заказа продуктов.

Задачи

1. Провести анализ предметной области для выбранной темы
2. Выбрать клиент-серверную архитектуру для разрабатываемого приложения и дать её детальное описание с помощью UML
3. Выбрать программный стек для реализации фуллстек CRUD приложения
4. Разработать клиентскую и серверную части приложения, реализовать авторизацию и аутентификацию пользователя, обеспечить работу с базой данных, заполнить тестовыми данными, валидировать ролевую модель на некорректные данные
5. Провести фаззинг-тестирование
6. Разместить исходный код клиент-серверного приложения в репозитории GitHub с наличием Dockerfile и описанием структуры проекта в readme файле
7. Развернуть клиент-серверное приложение в облаке
8. Разработать презентацию с графическими материалами.

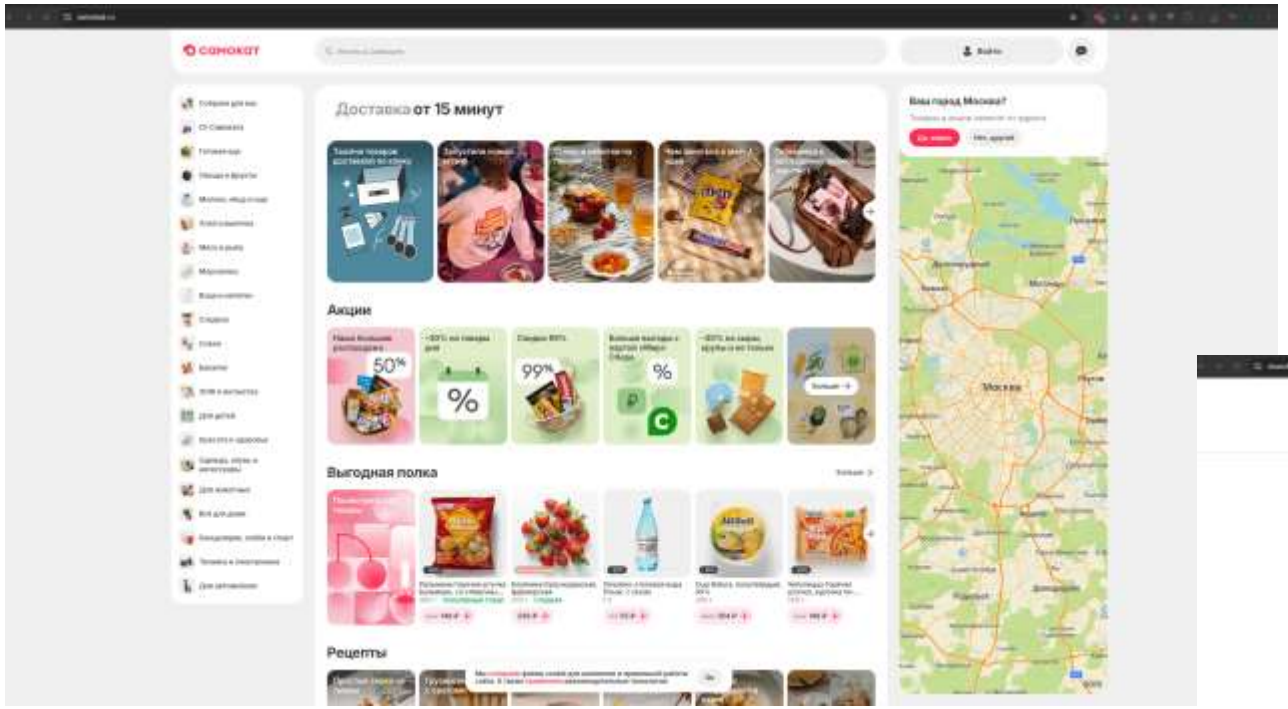
Технологии разработки

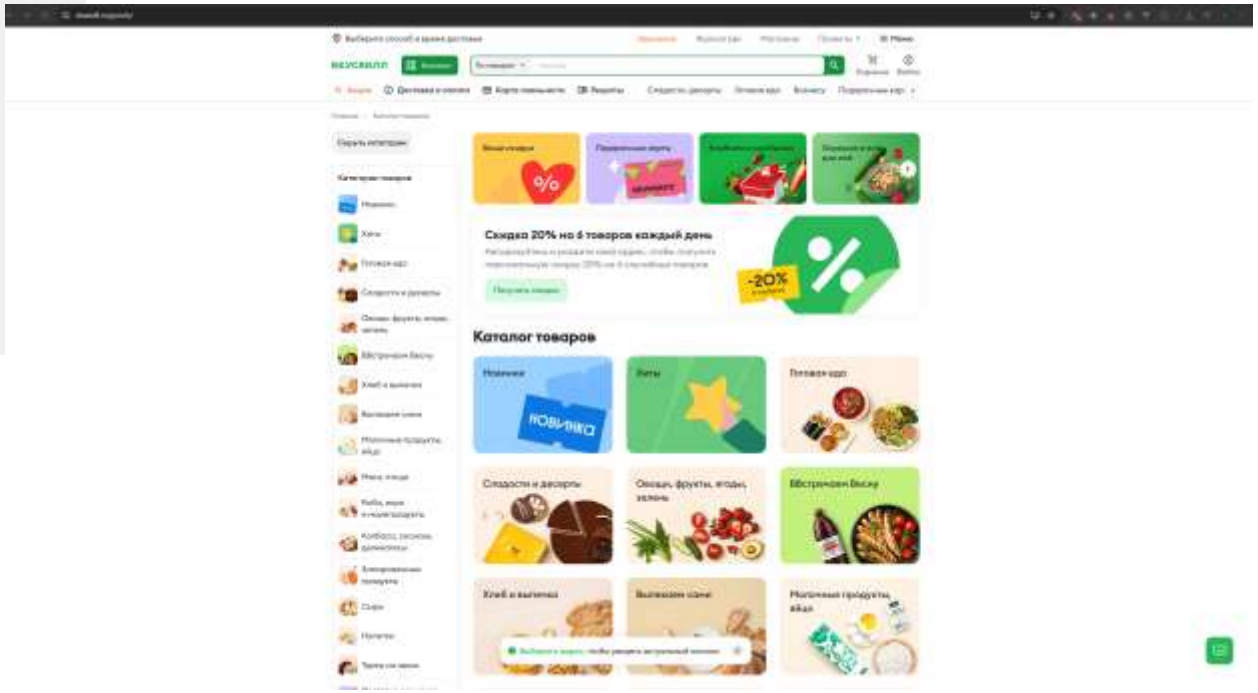
При разработке приложения был использован язык программирования **JavaScript**. Серверная часть приложения написана с помощью **Node.js** и библиотеки **Express**, для базы данных использовались **SQLite** и **SequalizeORM**. На клиентской части был использован фреймворк **React.js**, для стилей был использован **Bootstrap**.

Сравнительный анализ

При проведении анализа источников и аналогов для было обращено внимание на такие интернет-ресурсы, как «Яндекс Лавка», «Самокат», «Вкусвилл». В каждом приложении присутствует возможность просмотра имеющихся товаров с названием, картинкой и ценой, возможность добавить их в корзину, отображение количества товаров в корзине и возможность совершения заказа с указанием адреса и контактных данных.

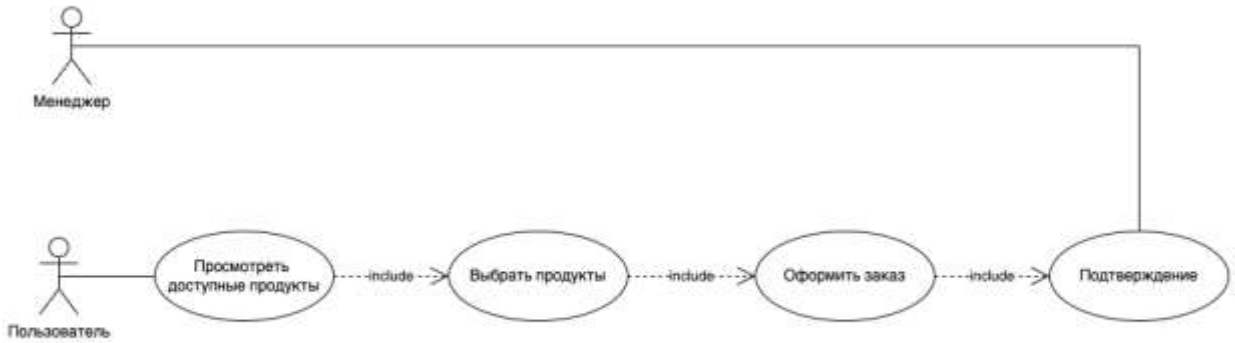
Сравнительный анализ



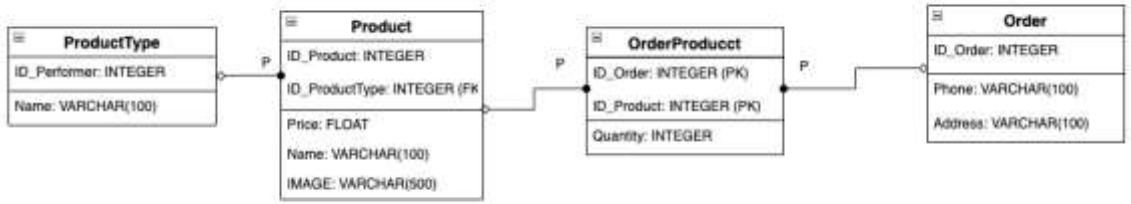


Структура приложения

Use-case диаграмма



Структура базы данных



Разработка серверной части

server > index.js > ...

```
const express = require('express');
const cors = require('cors');
const {sequelize} = require('./models');

const app = express();
const port = 8000;

const corsOptions = {
  origin: '*',
};

app.use(cors(corsOptions));
app.use(express.json());

// Роуты
const productRoutes = require('./routes/productRoutes');
const orderRoutes = require('./routes/orderRoutes');
const productTypeRoutes = require('./routes/productTypeRoutes');

app.use('/api/products', productRoutes);
app.use('/api/orders', orderRoutes);
app.use('/api/product-types', productTypeRoutes);

sequelize.sync().then(() => {
  app.listen(port, () => {
    console.log(`Server running on port ${port}`);
  });
});
```

server > Dockerfile

```
# Бэкенд
FROM node:18-alpine AS backend

WORKDIR /app

COPY package*.json ./
RUN npm ci --only=production

COPY . .

EXPOSE 8000
CMD ["node", "index.js"]
```

Разработка клиентской части

client > src > App.jsx > ...

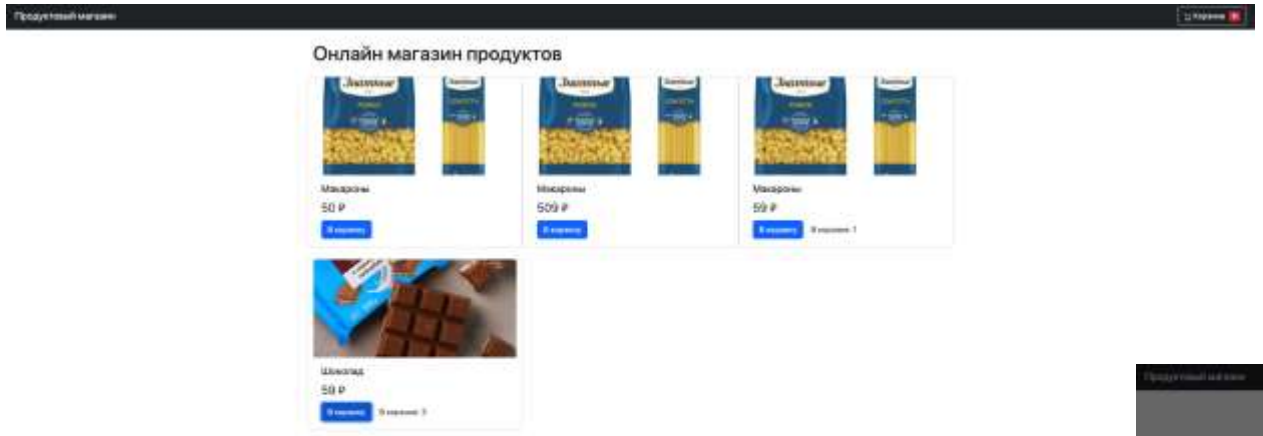
```jsx
function App() {
  const addToCart = (product) => {
    });
  };

  const handleOrder = async (phone, address) => {
    try {
      const productsData = cart.map(item => ({
        id: item.id,
        quantity: item.quantity
      }));

      await axios.post(`${API_URL}/orders`, {
        phone,
        address,
        products: productsData
      });

      setCart([]);
      alert('Заказ успешно оформлен!');
    } catch (error) {
      alert('Ошибка при оформлении заказа');
    }
  };

  return (
    <>
      <Header cartCount={cart.length} onOrder={handleOrder} />
      <Container className="mt-4">
        <h1 className="mb-4">Онлайн магазин продуктов</h1>
        <ProductList products={products} onAddToCart={addToCart} cart={cart} />
      </Container>
    </>
  );
}

export default App;
```

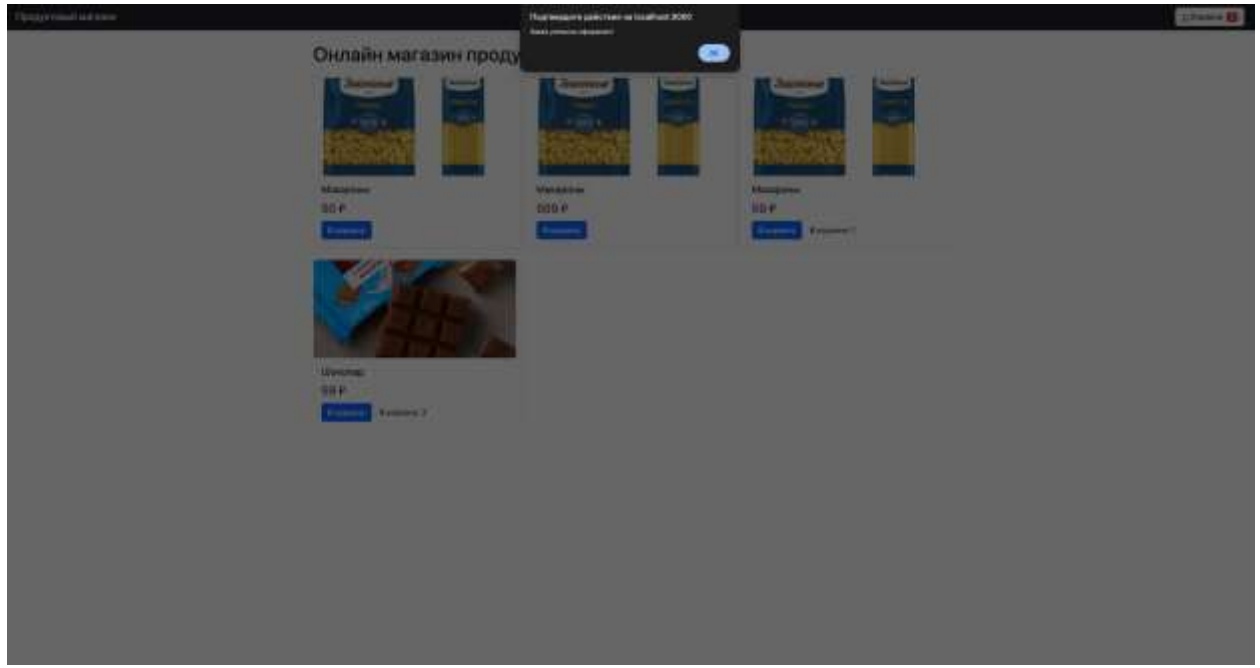
```jsx
useEffect(() => {
  axios.get(`${API_URL}/products`)
    .then(res => setProducts(res.data))
    .catch(err => console.error(err));
}, []);
```

client > Dockerfile

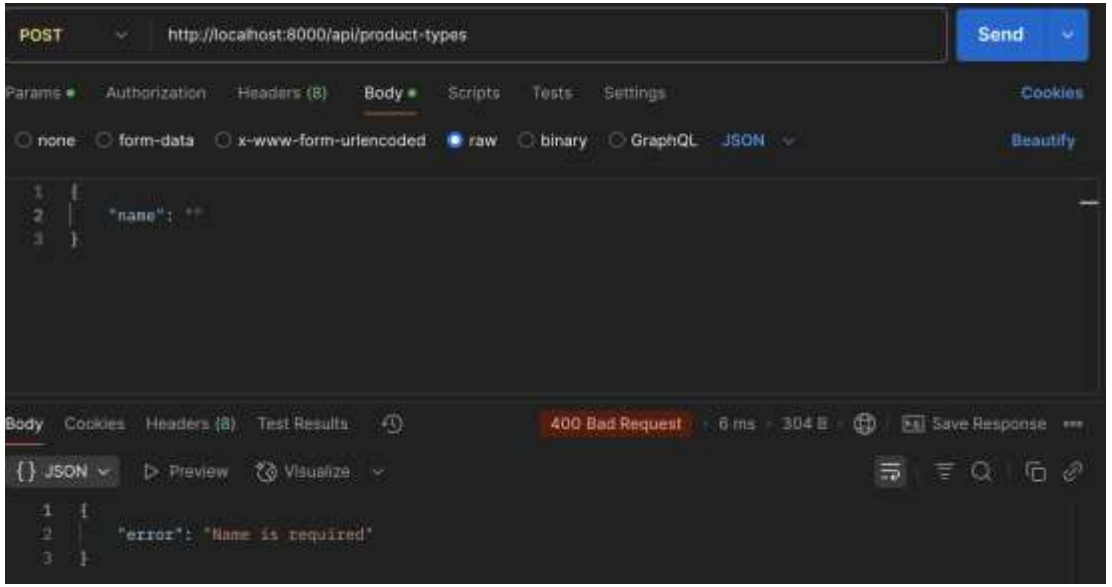
```dockerfile
# Фронтенд
FROM node:18-alpine AS frontend

WORKDIR /app

COPY package*.json ./
RUN npm ci

COPY . .
RUN npm run build

FROM nginx:alpine
COPY --from=frontend /app/build /usr/share/nginx/html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

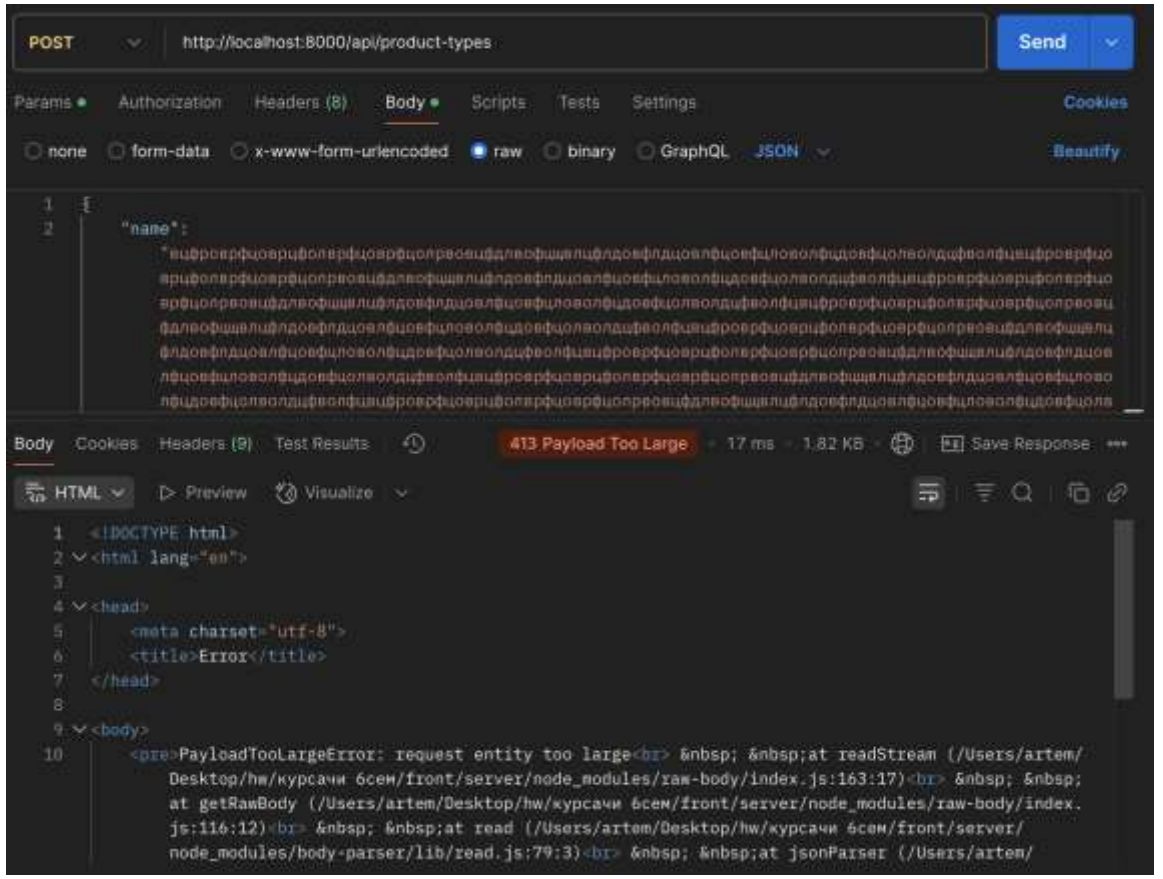
Тестирование приложения





Фаззинг-тестирование приложения





Результаты

- Разработано приложение обширным функционалом: возможность просмотра имеющихся товаров со всей информацией, оформление заказа ;
- Приложение стабильно работает и обеспечивает хорошую производительность.

URL хранилища с кодом:

https://github.com/stufade/front-kursovaya

СПАСИБО ЗА ВНИМАНИЕ!