Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОННИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Лабораторная работа №6

Создание приложения для базы данных

Студент: К.В. Горбачевский

Преподаватель: Д.В. Куприянова

МИНСК 2024

**СОДЕРЖАНИЕ**

[1 СОЗДАНИЕ СЕРВЕРНОЙ ЧАСТИ 4](#_Toc163390327)

[1.1 Подключение к базе данных 4](#_Toc163390328)

[1.2 Контроллер 4](#_Toc163390329)

[1.3 Сервер 5](#_Toc163390330)

[2 КЛИЕНТСКАЯ ЧАСТЬ 7](#_Toc163390331)

[2.1 Основные компоненты 7](#_Toc163390332)

[2.2 Скриншоты 9](#_Toc163390333)

[ЗАКЛЮЧЕНИЕ 11](#_Toc163390334)

# СОЗДАНИЕ СЕРВЕРНОЙ ЧАСТИ

Для серверной части приложения основанного на технологии REST API был выбран язык NodeJS. Помимо довольно широко функционала NodeJS позволяют использовать множество внешний модулей и библиотек.

Одной из таких библиотек явялется библиотека pg и pg-hstore для работы конкретно с базой данный PostgreSQL

## Подключение к базе данных

Для подключение к базе данных я использовал библиотеку pg, которая предоставляет для этого весь функционал.

Требуется указать имя базы данных, пароль, порт на которой запущена БД и другую конфигурационную информацию.

Скрипт подключения к базе данных представлен на рисунке 1.1

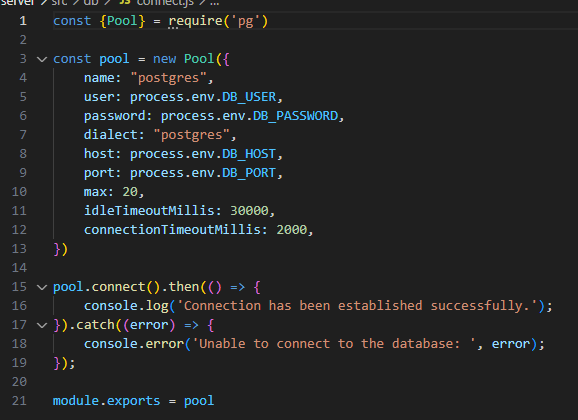


Рисунок 1.1 – Скрипт подключения к базе данных

## Контроллер

В папке controllers был создан файл, в котором описаны эндпоинты, реализующие запросы из четвертой и пятой лабораторной работы работы.

Эндпоинт представляет собой функцию, в которую попадает программа при вызове определенного метода с клиентской части приложения.

Внутри функции вызывается непосредственно запрос к БД. Пример API эндпоинтов представлены на рисунке 1.2



Рисунок 1.2 – API эндпоинты

## Сервер

Серверное приложение инициализируется через библиотеку express.js. Достаточно указать порт и запустить скрипт.

В строке 2 создается сущность app, которая является связующим звеном между API эндпоинтами и сервером.

На рисунке 1.3 представлен скрипт инициализации сервера

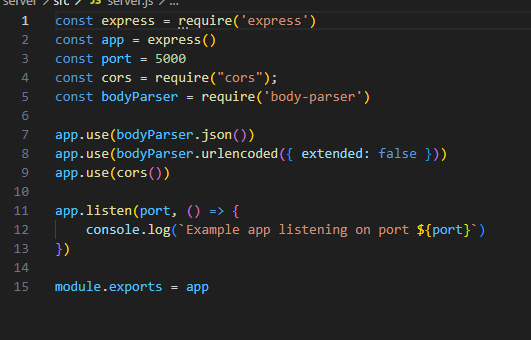


Рисунок 1.2 – API эндпоинты

# КЛИЕНТСКАЯ ЧАСТЬ

Для создания клиентской части приложения был выбран популярный фреймворк для JavaScript - React.

Приложение состоит из 3 простых компонентов:

1) Информационный компоннент. Содержит список доступных для вызова методов.

2) Форма. Компонент для ввода вызываемого метода и кнопка отправки.

3) Таблица. Компонент отображения ответа. В таблице отображается результат запроса от сервера

## 2.1 Основные компоненты

API запросы реализуются через библиотеку axios. В файле endpoints.ts описаны номера возможных эндопинтов для вызова (рис. 2.1.1)

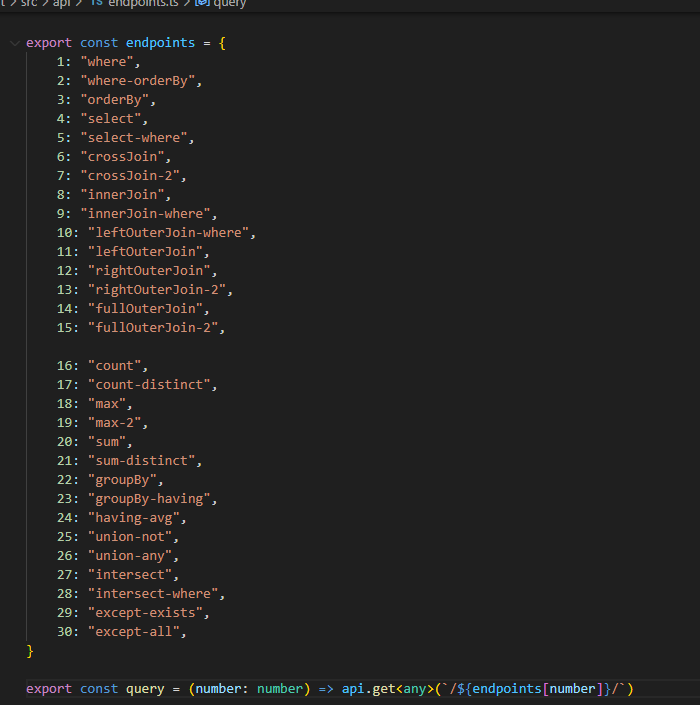


Рисунок 2.1.1 – API эндпоинты

Основной компонент приложения представлен на рисунке 2.1.2



Рисунок 2.1.2 – Содержимое файла Main.tsx

На рисунке 2.1.3 представлен основной компонент (Form.tsx), осуществляющий логику клиентского приложения (отправку запросов).



Рисунок 2.1.3 – Содержимое файла Form.tsx

На рисунке 2.1.4 представлен основной компонент таблицы, которая отвечает за отображение данных после ответа. Таблицу можно очистить нажав кнопку под таблице.

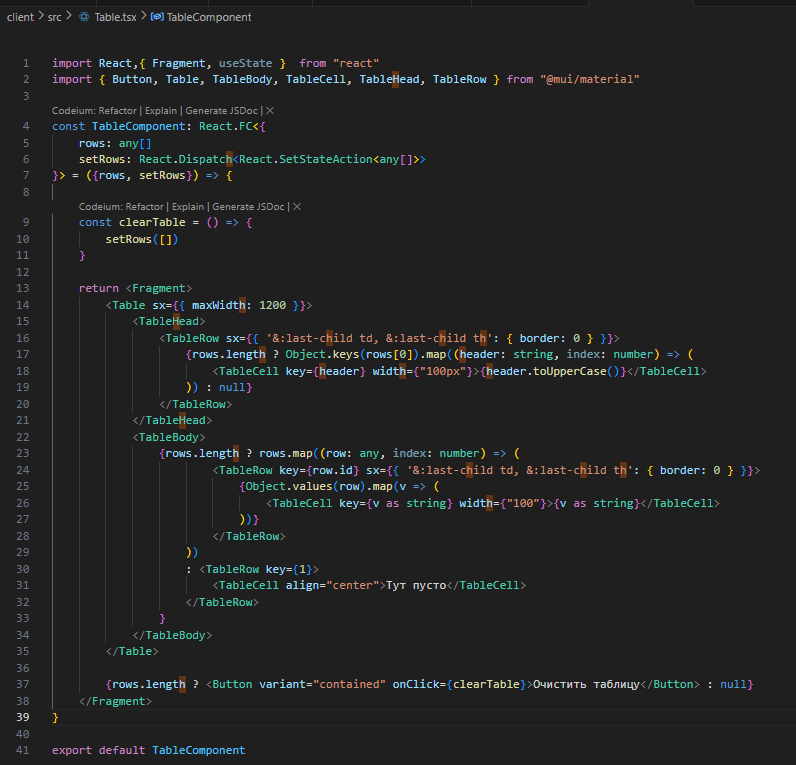


Рисунок 2.1.4 – Содержимое файла Table.tsx

## 2.2 Скриншоты

Приложение запускается в браузере на 3000 порту localhost (по умолчанию). На рисунке 2.2.1 показан вид работающего приложения на странице браузера.

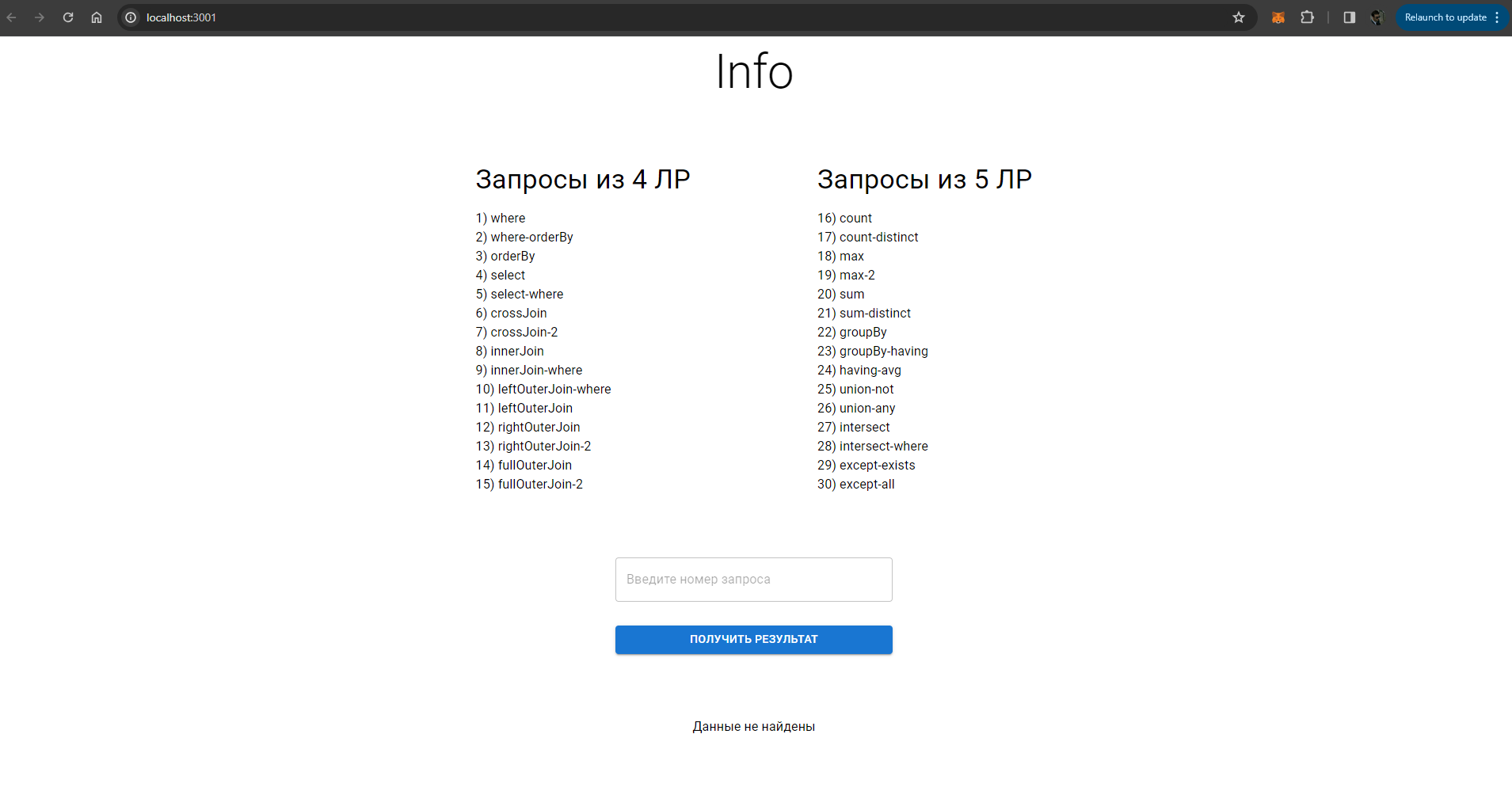


Рисунок 2.2.1 – Скриншот из браузера

На рисунке 2.2.2 показан пример возвращаемых данных

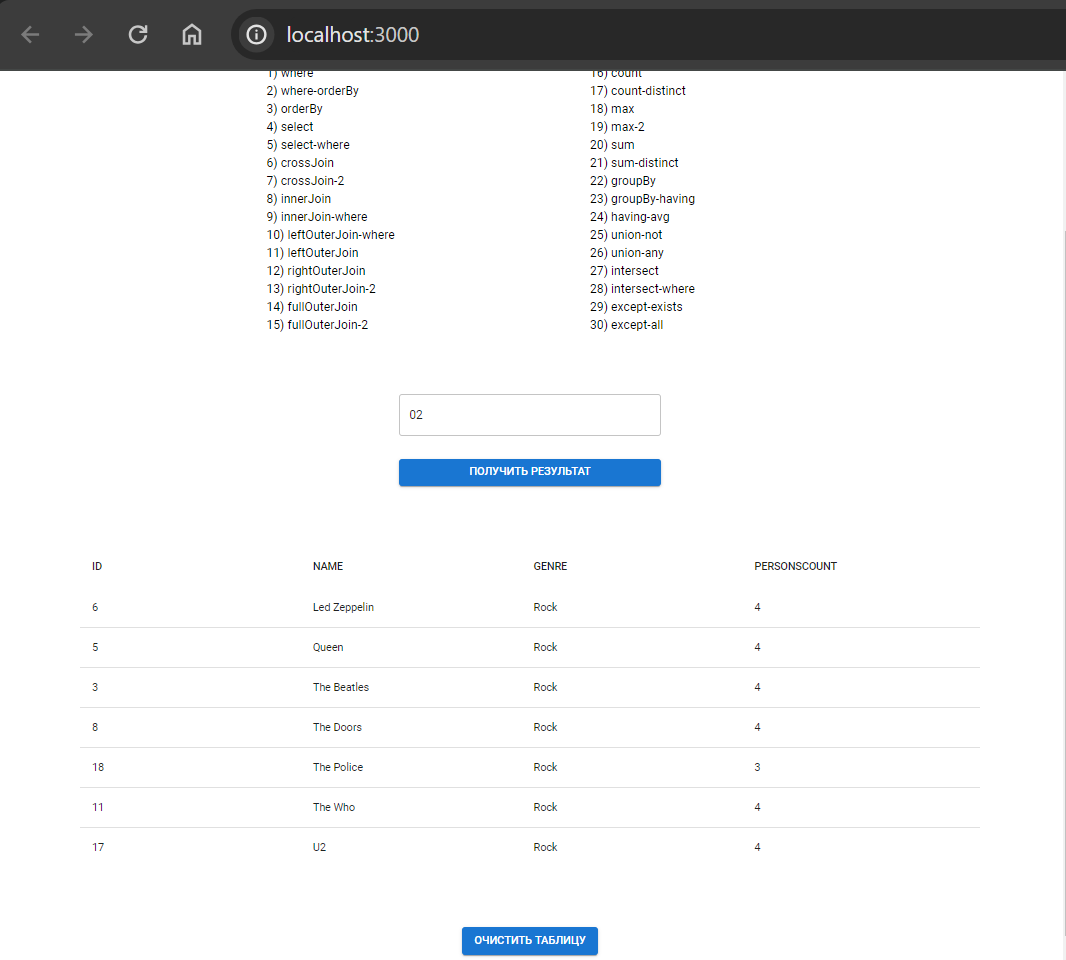


Рисунок 2.2.2 – Пример возвращаемых данные

# ЗАКЛЮЧЕНИЕ

В ходе данной лабораторной работы было создано простое клиент-серверное приложение на базе NodeJS, Typescript и React.

Был реализован функционал SQL-запросов из лабораторных работ 4 и 5. Создан сервер, который выполняет запросы и взаимодействует с базой данный, и клиент который направляет запросы на сервер посредством API вызовов.