**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МОЭВМ**

**ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ**

**по дисциплине «Введение в нереляционные базы данных»**

**Тема: Сервис хранения экспериментов инструмента Sumo**

| Студент гр. 0382 |  | Крючков А.М. |
| --- | --- | --- |
| Студент гр. 0383 |  | Смирнов И.А. |
| Студентка гр. 0383 |  | Пустовалова Е.М. |
| Преподаватель |  | Заславский М.М. |

Санкт-Петербург

2023**ЗАДАНИЕ**

**НА КУРСОВУЮ РАБОТУ**

| Студенты  Крючков А.М. | | |
| --- | --- | --- |
| Группа 0382  Пустовалова Е.М.  Смирнов И.А.  Группа 0383 | | |
| Тема проекта: Сервис хранения экспериментов инструмента Sumo | | |
| Исходные данные:  Необходимо реализовать приложение для импорта / хранения / поиска / визуализации экспериментов инструмента Sumo с помощью СУБД Neo4j. | | |
| Содержание пояснительной записки:  «Содержание», «Введение», «Сценарий использования», «Модель данных», «Разработанное приложение», «Выводы», «Приложения», «Литература» | | |
| Предполагаемый объем пояснительной записки:  Не менее 10 страниц. | | |
| Дата выдачи задания: 20.09.2023 | | |
| Дата сдачи реферата: 23.12.2023 | | |
| Дата защиты реферата: 23.12.2023 | | |
| Студент гр. 0382 |  | Крючков А.М. |
| Студент гр. 0383 |  | Смирнов И.А. |
| Студент гр. 0383 |  | Пустовалова Е.М. |
| Преподаватель |  | Заславский М.М. |

**АННОТАЦИЯ**

В рамках данного курса предполагалось разработать какое-либо приложение в команде на одну из поставленных тем. Была выбрана тема создания приложения для импорта / хранения / поиска / визуализации экспериментов инструмента Sumo с помощью СУБД Neo4j. Найти исходный код и всю дополнительную информацию можно по ссылке: https://github.com/moevm/nosql2h23-sumo

**ANNOTATION**

As part of this course, it was supposed to develop an application in a team on one of the set topics. The topic of creating an application for importing/storing/searching/visualizing experiments of the Sumo tool using the Neo4j DBMS was chosen. You can find the source code and all additional information at the link:

https://github.com/moevm/nosql2h23-sumo

**СОДЕРЖАНИЕ**

**ЗАДАНИЕ................................................................................................................. 2**

**АННОТАЦИЯ.......................................................................................................... 3**

**СОДЕРЖАНИЕ........................................................................................................ 4**

**1. ВВЕДЕНИЕ........................................................................................................... 5**

1.1. Актуальность решаемой проблемы ……………………............................ 6

1.2. Постановка задачи ……………………........................................................ 6

1.3. Предлагаемое решение ................................................................................. 6

1.4. Качественные требования к решению ........................................................ 6

**2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ.................................................................... 7**

2.1. Макеты UI....................................................................................................... 7

2.2. Сценарии использования для задачи ......................................................... 9

2.2.1. Импорт данных ………........................................................................ 9

2.2.1.1. Сценарий использования - “массовый импорт данных”..... 9

2.2.1.2. Сценарий использования - “ручной импорт данных” ........ 9

2.2.2. Представление данных ........................................................................ 10

2.2.3. Анализ данных ………......................................................................... 10

2.2.4. Экспорт данных ………………………............................................... 10

2.2. Вывод относительно нашего функционала ............................................... 11

**3. МОДЕЛЬ ДАННЫХ........................................................................................... 12**

3.1. Нереляционная модель данных - Neo4j ..................................................... 12

3.1.1. Графическое представление данных ………………………………. 12

3.1.2. Описание назначений коллекций, типов данных и сущностей …... 12

3.1.3. Оценка удельного объема информации, хранимой в модели .......... 13

3.1.4. Запросы к модели, с помощью которых реализуются сценарии использования ……………………………………………………................ 14

3.2. Реляционные модели данных...................................................................... 14

3.2.1. Графическое представление данных…………………………….…. 14

3.2.2. Описание назначений коллекций, типов данных и сущностей....... 15

3.2.3. Оценка удельного объема информации, хранимой в модели.......... 16

3.2.4. Запросы к модели, с помощью которых реализуются сценарии использования………………………………................................................ 17

3.3. Сравнение моделей................................................................................... 18

3.3.1. Удельный объем информации ......................................................... 18

3.3.2. Запросы по отдельным юзкейсам ................................................... 18

3.3.3. Вывод....................................... ......................................................... 18

**4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ...........................................................**

4.1. Краткое описание......................................................................................

4.2. Использованные технологии...................................................................

4.3. Снимки экрана приложения.....................................................................

**5. ВЫВОД...............................................................................................................**

5.1. Достигнутые результаты..........................................................................

5.2. Недостатки и пути для улучшения полученного решения...................

5.3. Будущее развитие решения......................................................................

**6. ПРИЛОЖЕНИЯ................................................................................................**

6.1. Документация по сборке и развертыванию приложения......................

6.2. Инструкция для пользователя.................................................................

**7. ЛИТЕРАТУРА...................................................................................................**

1. **ВВЕДЕНИЕ**
2. **Актуальность решаемой проблемы**

Цель работы - создать быстрое и удобное веб-приложения для импорта / хранения / поиска / визуализации экспериментов инструмента Sumo.

1. **Постановка задачи**

Сервис должен предоставлять функционал для решения следующих задач:

* Позволять пользователям загружать эксперименты инструмента Sumo
* Предоставлять пользователям инструменты для визуализации и работы со статистиками дорожных сетей экспериментов

1. **Предлагаемое решение**

Для решения поставленной задачи необходимо разработать web приложение для импорта / хранения / поиска / визуализации экспериментов инструмента Sumo.

1. **Качественные требования к решению**

Требуется разработать веб-приложение с использованием СУБД neo4j

**2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ**

**2.1. Макет UI**

1. Главная страница (Рис. 1)

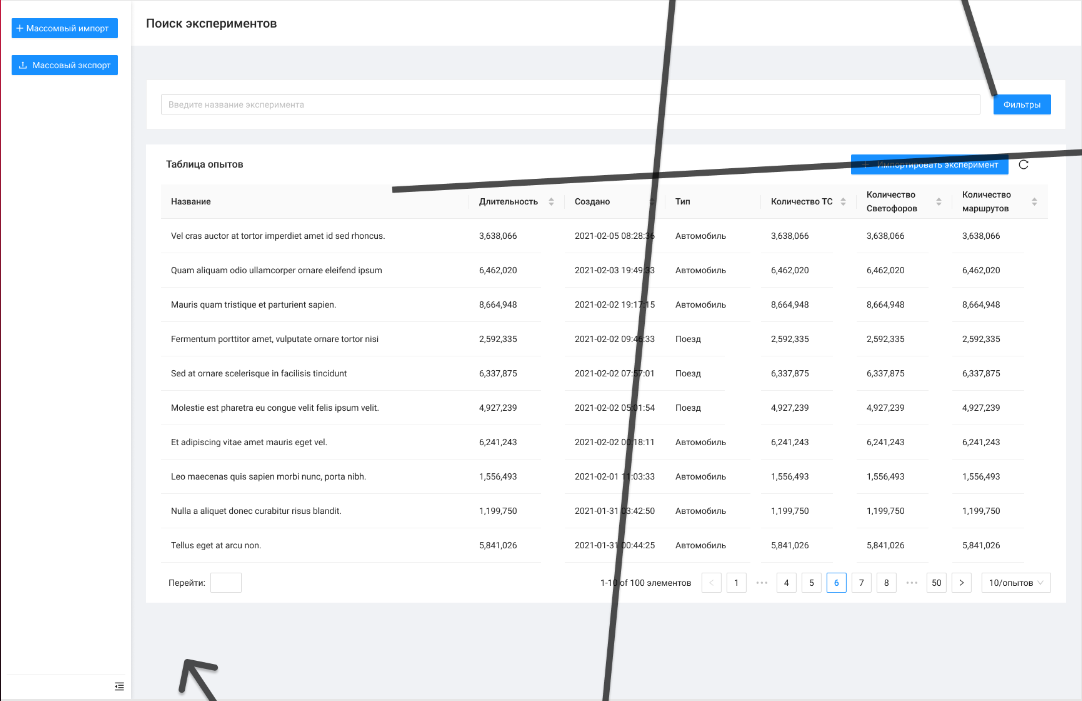


Рисунок 1 - главная страница

1. Страница просмотра эксперимента (Рис. 2)

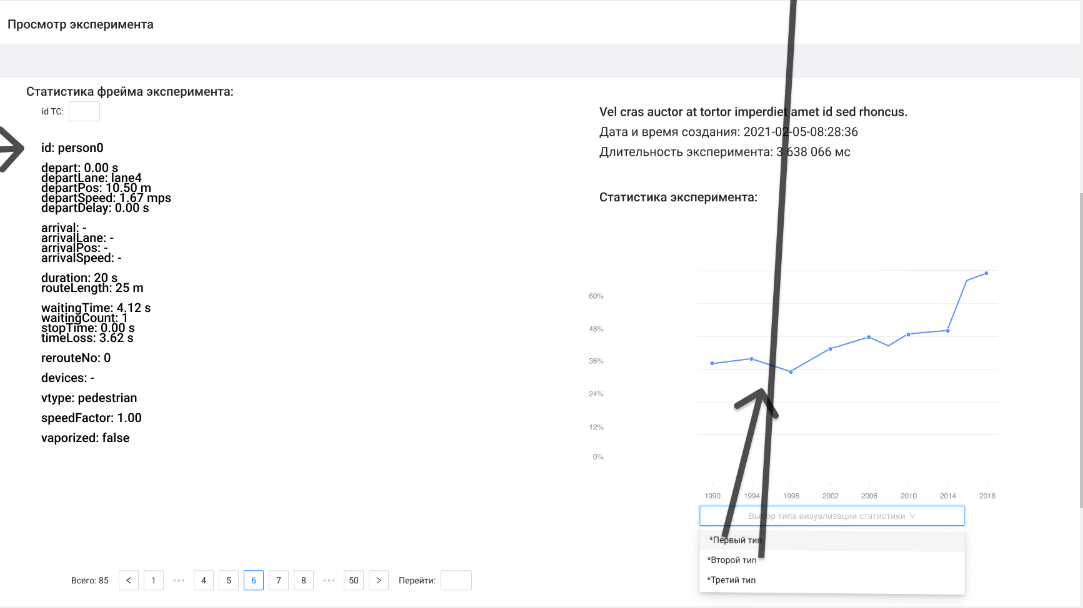


Рисунок 2 - Страница просмотра эксперимента

1. Экран настройки фильтрации (Рис. 3)

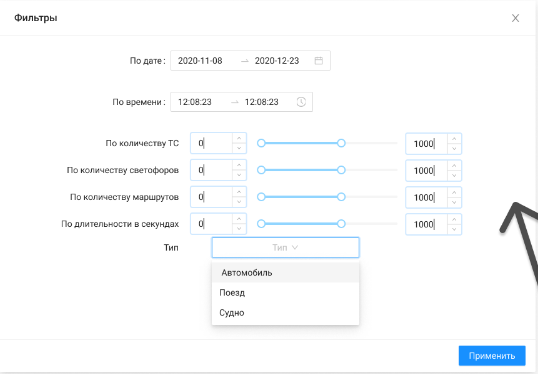


Рисунок 3 - Экран настройки фильтрации

1. Страница общей статистики экспериментов (Рис. 4)

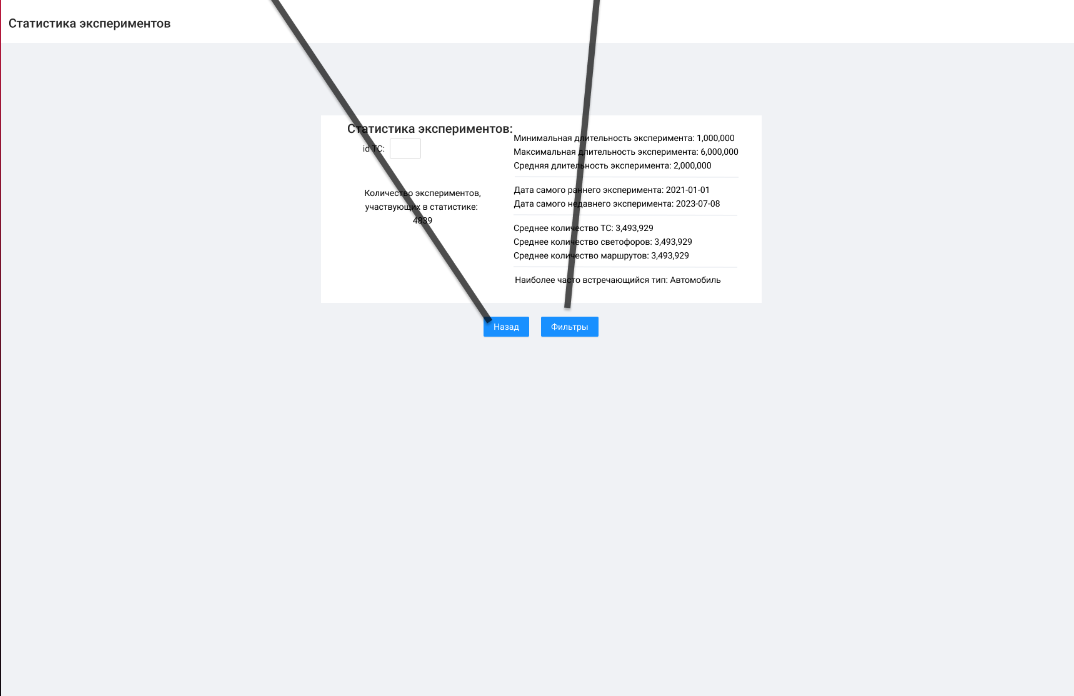


Рисунок 4 - Страница общей статистики экспериментов

**2.2. Сценарии использования для задачи**

**2.2.1. Импорт данных**

**2.2.1.1. Сценарий использования - “массовый импорт данных”**

Действующее лицо - пользователь

Предусловие:

* Пользователь находится на главной странице сервиса

Основной сценарий:

1. Пользователь видит кнопку “Массовый импорт”
2. Пользователь нажимает на иконку “Массовый импорт”
3. Появляется окно с выбором файла для импорта данных
4. Пользователь выбирает файл
5. Данные отправляются на серверную часть и успешно импортируются

Альтернативный сценарий:

1. Пользователь выбрал некорректный файл
2. Выводится сообщение о некорректном файле

**2.2.1.2. Сценарий использования - “ручной импорт данных”**

Действующее лицо - пользователь

Предусловие:

* Пользователь находится на главной странице сервиса

Основной сценарий:

1. Пользователь видит кнопку Импорт эксперимента
2. Пользователь нажимает кнопку Импорт эксперимента
3. Пользователь вводит данные дорожной сети эксперимента
4. Пользователь нажимает “ОК”
5. Данные отправляются на серверную часть и успешно импортируются

**2.2.2. Представление данных**

Действующее лицо - пользователь

Предусловие:

* Пользователь находится на главной странице сервиса

Основной сценарий:

1. Пользователь видит таблицу, в которой представлены строки-эксперименты
2. Пользователь нажимает на строку эксперимента
3. Пользователь переходит на страницу просмотра эксперимента
4. Пользователь видит визуализацию и статистику эксперимента

**2.2.3. Анализ данных**

Действующее лицо - пользователь

Предусловие:

* Пользователь находится на главной странице сервиса

Основной сценарий:

1. Пользователь нажимает на кнопку навигации “статистика”
2. Пользователь успешно переходит на страницу статистики, на которой отображена общая статистика экспериментов с возможностью настройки через страницу фильтрации

**2.2.4. Экспорт данных**

Действующее лицо - пользователь

Предусловие:

* Пользователь находится на главной странице сервиса

Основной сценарий:

1. Пользователь видит кнопку “Массовый экспорт”
2. Пользователь нажимает на кнопку
3. Пользователь выбирает эксперимент
4. Система формирует соответствующий файл в формате JSON с информацией об эксперименте
5. Файл скачивается на устройство пользователя.

**2.3. Вывод относительно нашего функционала**

В результате анализа сценариев использования и самой задачи приложения был сделан вывод, что в задаче преобладают операции чтения, поскольку импорт представляет собой запись информации в базу данных, а все остальные операции направлены на чтение данной информации.

**3. МОДЕЛЬ ДАННЫХ**

**3.1. Нереляционная модель данных - Neo4j**

1. **Графическое представление данных**

Графическое представление нереляционной базы данных представлено на рис. 5

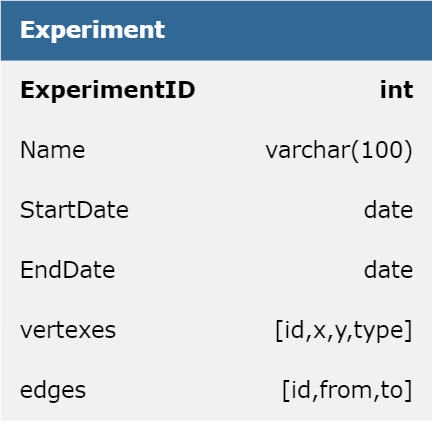


Рисунок 5 - Схема нереляционной бд

Рисунок 9 - графическое представление нереляционной базы данных

1. **Описание назначений коллекций, типов данных и сущностей**

Experiment - таблица экспериментов

оболочка над дорожной сетью с некоторой метаинформацией

Name - строковое название эксперимента

StartDate, EndDate - date, когда эксперимент начался и когда закончился

vertexes - узлы сети:

-x, y - float координаты узлов в координатной плоскости дорожной сети

-type - строковый тип узла (висячая вершина или нет)

edges - пути между связанными напрямую дорогой узлами дорожной сети:

-from, to - id вершин (откуда ребро выходит/куда ребро входит)

Experiment - таблица экспериментов

оболочка над дорожной сетью с некоторой метаинформацией

Name - строковое название эксперимента

StartDate, EndDate - date, когда эксперимент начался и когда закончился

vertexes - узлы сети:

-x, y - float координаты узлов в координатной плоскости дорожной сети

-type - строковый тип узла (висячая вершина или нет)

edges - пути между связанными напрямую дорогой узлами дорожной сети:

-from, to - id вершин (откуда ребро выходит/куда ребро входит)

1. **Оценка удельного объема информации, хранимой в модели**

Общий объем информации (O) можно оценить по формуле:

где - объемы памяти, необходимые для хранения одной вершины в vertex, одного ребра в esges и общей информации эксперимента соответственно  
V, M, N - количество вершин, количество рёбер и количество экспериментов соответственно  
предположим, что в среднем дорожная сеть представима полным графом, тогда

Тогда общий объем информации можно оценить через количество экспериментов как

Рассчитаем :

Тогда зависимость общего объема от числа экспериментов следующая:

1. **Запросы к модели, с помощью которых реализуются сценарии использования**

Получить эксперимент по имени

db.exps.find({name: \_name})

Получить эксперименты начатые в диапазоне дат

db.exps.find({start\_date: {$gt: \_start\_date}}, {end\_date: {$lt: \_end\_date}})

Получить представление эксперимента (потребуется получить эксперимент, все вершины и ребра)

db.exps.find({experiment\_id: \_experiment\_id})

Получить эксперименты, с числом вершин не менее заданного

db.exps.find({vertexes: {$size:{$gt: \_value}}})

**3.2. Реляционная модель данных**

1. **Графическое представление данных**

Графическое представление реляционной базы данных представлено на рис. 6

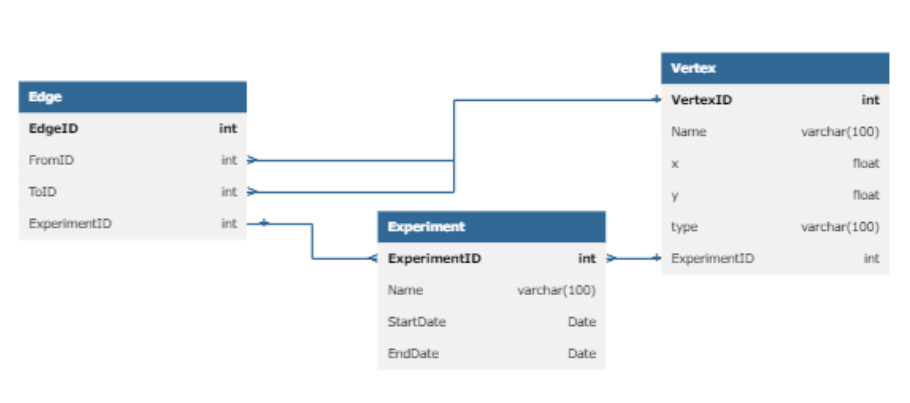


Рисунок 6 - графическое представление реляционной базы данных

1. **Описание назначений коллекций, типов данных и сущностей**

Experiment - таблица экспериментов сущность, объединяющая вершины и рёбра в единый объект - схему (граф) дорожной сети

StartDate - EndDate - date когда был проведен эксперимент (начало, конец)

Vertex - таблица вершин сущность для схематического отображения узлов дорожной сети

поля:

x, y - float расположение на координатной плоскости отображения графа type - строковый тип (тупик или узел)

Edges - таблица направленных рёбер сущность для схематического отображения путей между связанными напрямую дорогой узлами дорожной сети

основные поля:

FromID - id вершины, откуда ребро выходит

ToID - id вершины, куда ребро входит

1. **Оценка удельного объема информации, хранимой в модели**

Общий объем информации (O) можно оценить по формуле:

где - объемы памяти, необходимые для хранения одной сущности вершины, одной сущности ребра и одной сущности эксперимента соответственно  
V, M, N - количество вершин, количество рёбер и количество экспериментов соответственно  
предположим, что в среднем дорожная сеть представима полным графом, тогда

Тогда общий объем информации можно оценить через количество экспериментов как

Рассчитаем объёмы памяти для хранения отдельно взятых сущностей:

Тогда зависимость общего объема от числа экспериментов следующая:

1. **Запросы к модели, с помощью которых реализуются сценарии использования**

Получить эксперимент по имени

SELECT \* FROM EXPERIMENT

WHERE name = $name

Получить эксперименты начатые в диапазоне дат

SELECT \* FROM EXPERIMENT

WHERE start\_date BETWEEN $start\_date\_min AND $start\_date\_max

Получить представление эксперимента (потребуется получить эксперимент, все вершины и ребра)

SELECT \* FROM EXPERIMENT

WHERE experiment\_id = $experiment\_id

SELECT \* FROM VERTEX

WHERE experiment\_id = $experiment\_id

SELECT \* FROM EDGES

WHERE experiment\_id = $experiment\_id

Получить эксперименты, с числом вершин не менее заданного

SELECT \* FROM EXPERIMENT

WHERE (SELECT COUNT(\*) FROM VERTEX WHERE VERTEX.experiment\_id = EXPERIMENT.experiment\_id) >= $min\_v

**3.3. Сравнение моделей**

1. **Удельный объем информации**

Нереляционная модель предоставляет лучшее решение по памяти (N\*430b против N\*476b) и меньшее значение избыточности (1.21 против 1.28). Направления роста моделей в целом схожи

1. **Запросы по отдельным юзкейсам**

Нереляционная модель затрачивает меньшее или равное количество запросов и также затрагивает меньшее или равное число коллекций в рамках реализации пользовательских сценариев

1. **Вывод**

В задаче обработки графовой структуры данных дорожной сети лучше использовать нереляционную модель данных**4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ**

**4.1. Краткое описание**

Приложение было разработано на фреймворке Nuxt.js, работающем на Node.js. Для реализации бэкенд-логики использовались API маршруты Nuxt.js, в то время как для фронтенда использовался Vue.js.

**4.2. Использованные технологии**

База данных:

* Neo4j

Back-end:

* Node.js
* Nuxt.js API Routes

Front-end:

* Vue.js
* Nuxt.js

**4.3. Снимки экрана приложения**

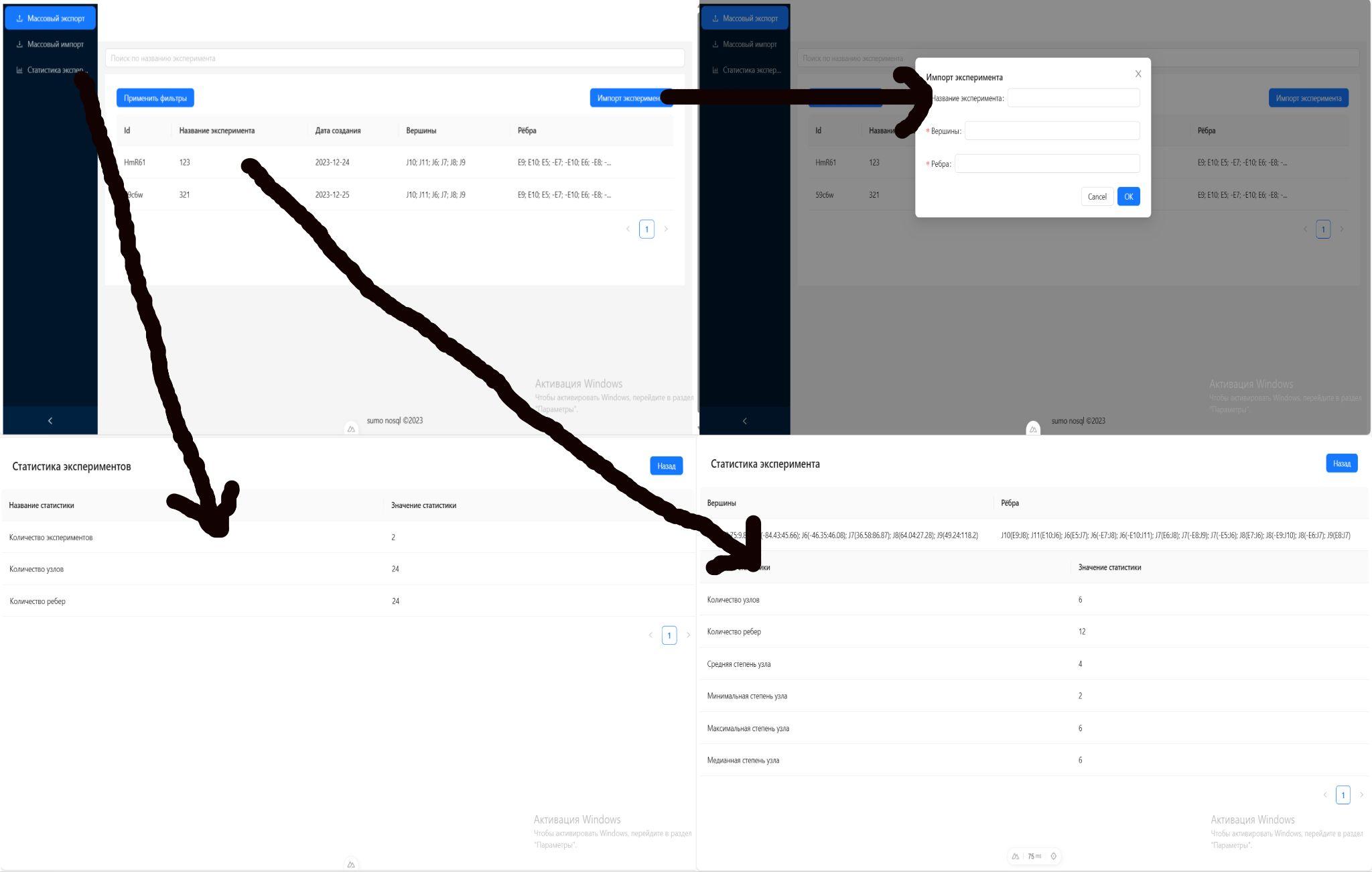


Рисунок 1 - Схема экранов приложения

**5. ВЫВОДЫ**

**5.1. Достигнутые результаты**

На текущем этапе развития приложения, мы успешно реализовали функционал импорта сетей для экспериментов, а также получения статистики как для отдельных экспериментов, так и для всех. Кроме того, для большинства параметров эксперимента реализованы фильтры.

**5.2. Недостатки и пути для улучшения полученного решения**

В данный момент, мы столкнулись с некоторыми проблемами производительности при массовом импорте и экспорте данных, особенно при работе с большими объемами данных. Один из потенциальных путей улучшения — использование потокового обработчика данных и рабочих потоков (worker threads) для оптимизации этих операций.

**5.3.** **Будущее развитие решения**

В перспективе планируется расширение функционала приложения, включая поддержку большего количества атрибутов из данных экспериментов SUMO.

**6. ПРИЛОЖЕНИЯ**

**6.1. Документация по сборке и развертыванию приложения**

1. Скачать проект из репозитория
2. Запустить команду: docker compose up

**6.2. Инструкция для пользователя**

1. Открыть приложение: http://localhost:3000/

**7. ЛИТЕРАТУРА**

1. Документация к Neo4j: <https://neo4j.com/docs/>
2. Официальная документация Nuxt: https://nuxt.com/docs/