

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт (технический университет)»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| УГНС | | 09.00.00 | Информатика и вычислительная техника | | |
| Направление подготовки | | 09.03.01 | Информатика и вычислительная техника | | |
| Направленность (профиль) | |  | Системы автоматизированного проектирования | | |
| Форма обучения | |  | очная | | |
|  | |  |  | | |
| Факультет | |  | Информационных технологий и управления | | |
| Кафедра | |  | Систем автоматизированного проектирования и управления | | |
| Учебная дисциплина | |  | Разработка программных систем | | |
| Курс | 2 | | | Группа | 423 |

Отчёт по лабораторной работе №3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнитель: |  |  |  |  |
| обучающийся группы 423 |  |  |  | Мухаметов Данил Ильнурович  Маскаев Иван Сергеевич  Цуркан Владимир Сергеевич |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |
| Проверил: |  |  |  | Соболевский Владислав Алексеевич |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |

СОДЕРЖАНИЕ

[1. Постановка цели 3](#_Toc153938007)

[2. Описание хода выполнения 3](#_Toc153938008)

[3. Выводы по заданию 12](#_Toc153938009)

## Постановка цели

Разработка графического пользовательского интерфейса для программы и интеграция базы данных для веб-приложений.

## Описание хода выполнения

* 1. Создание ветки в системе контроля

Для выполнения задания было необходимо в системе контроля GitHub создать новую ветку разработки, в которую вносились все модификации 2 лабораторной работы, при этом не изменяя основную ветку разработки. Для этого была создана ветка “RPS project”.

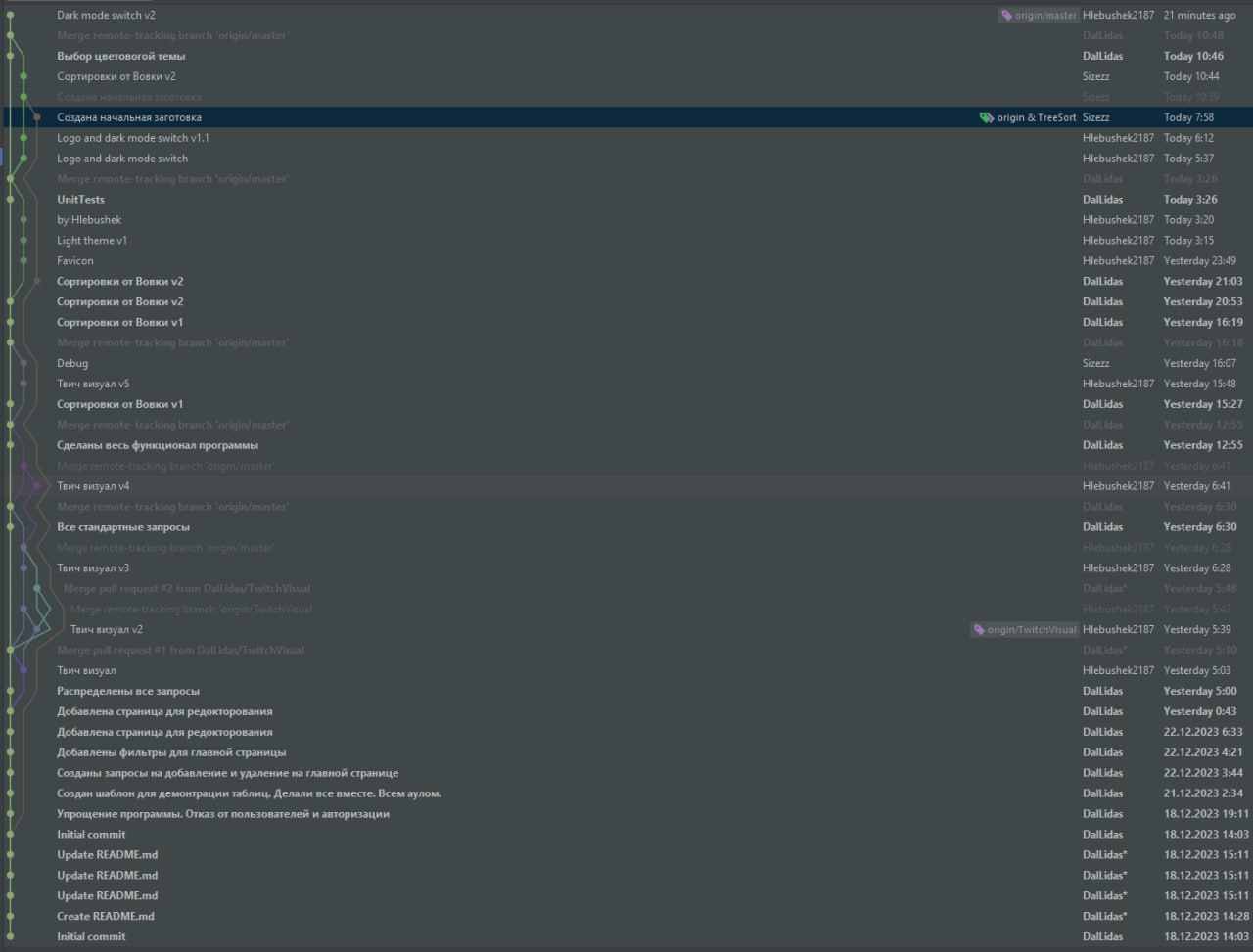


Рисунок 1 – Экранная копия окна всех веток проекта с данными об их изменениях

* 1. Тестирование

Для разработанной программы были разработаны модульные тесты добавления случайно сгенерированных массивов в базу данных, выгрузки и сортировки 100 случайных массивов из базы данных и очистки базы данных. Каждый тест должен быть проведен для базы данных на 100, 1000 и 10000 записей. Для разработки тестов были использованы ОRМ «Alchemy» и коннектор «psycopg2». Итоги тестирования приведены в Таблице 1.

Таблица 1. Итоги тестирования программы

|  |  |  |  |
| --- | --- | --- | --- |
| Кол-во массивов | Создание, с | Сортировка, с | Удаление, с |
| 100 | 5.1337 | 7.9269 | 6.8144 |
| 1000 | 49.3073 | 80.4122 | 74.1749 |
| 10000 | 489.0411 | 1025.3201 | 674.3454 |
| **Общее** | 543.4821 | 1113.6592 | 755.3347 |

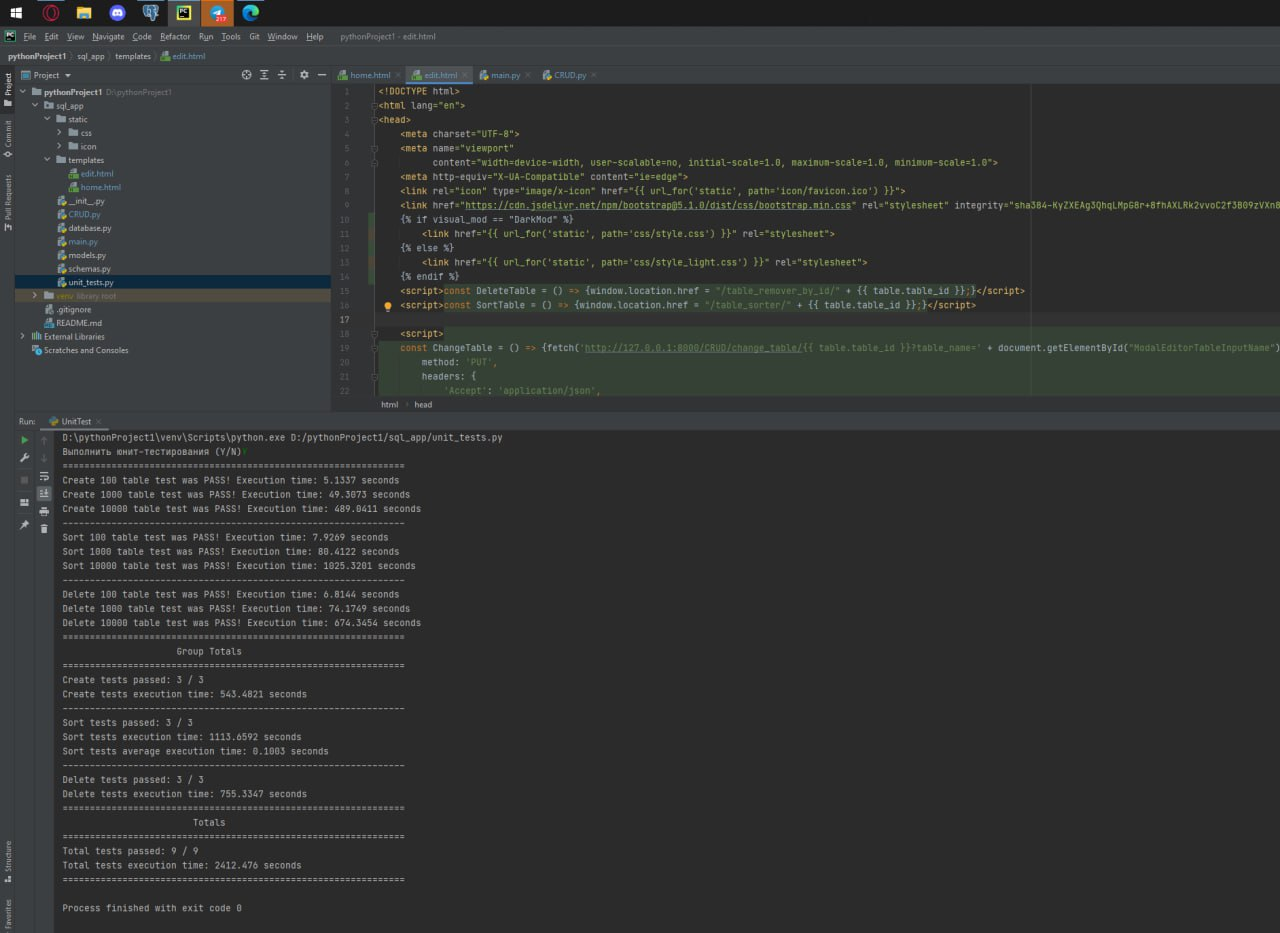


Рисунок 2 – Экранная копия окна консоли с результатами тестирования

* 1. Серверная оболочка

Для программы, реализованной в рамках 2 лабораторной работы, была разработана серверная оболочка на языке Python при помощи фреймворка «FastAPI». При помощи созданной оболочки создается связь между приложением и базой данных, а также возможность использовать функцию сортировки из второй лабораторной, которая была написана на языке С++. Так, в выполненной программе при помощи FastAPI будет прослушиваться порт 8000 на подключения и поступающие запросы.

* 1. База данных

В выполненной работе было необходимо развернуть базу данных для хранения результатов работы программы. Для этого была использована база данных «PostgreSQL» работающая со структурами данных типа «ключ — значение». Данная база данных была выбрана из-за ее высокой скорости работы и надёжности. В качестве коннектора была использована СУБД «Alchemy».

В базе данных таблицы хранятся в формате: «индекс таблицы, имя таблицы, флаг сортировки, данные в JSON формате».

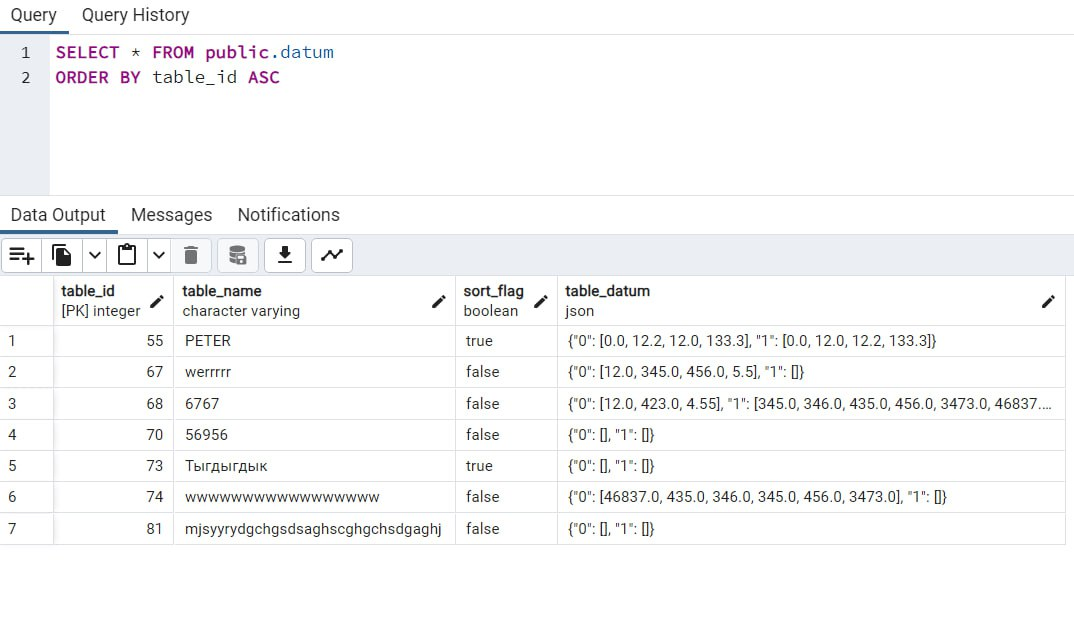


Рисунок 3 – Экранная копия хранения ключей в базе данных «PostgreSQL».

Обращения к базе данных проходят на серверной части программы через запросы POST и GET. Пользователь передает необходимый запрос по URL в формате:

«http://127.0.0.1:8000/CRUD/название\_запроса?offset=отступ\_по\_записям&limit=количество\_записей». Далее, на сервере этот URL обрабатывается и делается требуемый запрос базе данных.

* 1. Реализация графического интерфейса

Для программы сортировки деревом, выполненной в лабораторной работе 2 был реализован графический пользовательский веб-интерфейс при помощи языка Python, серверной оболочки FastAPI и шаблонизатора Ginger.

В графическом интерфейсе были реализованы функции:

1. Ввода и редактирования массива для сортировки с клавиатуры (рис. 4);
2. Сохранения введенного в интерфейсе массива в базу данных (рис. 5, 6);
3. Вывода списка всех массивов хранящихся в базе данных (без возможности редактирования и сортировка) (рис. 7);
4. Выгрузки массива из базы для редактирования (рис. 8);
5. Сортировки массива по алгоритму сортировки слиянием, реализованному во 2 лабораторной работе (рис. 9);
6. Удаления массива из базы данных (рис. 10).

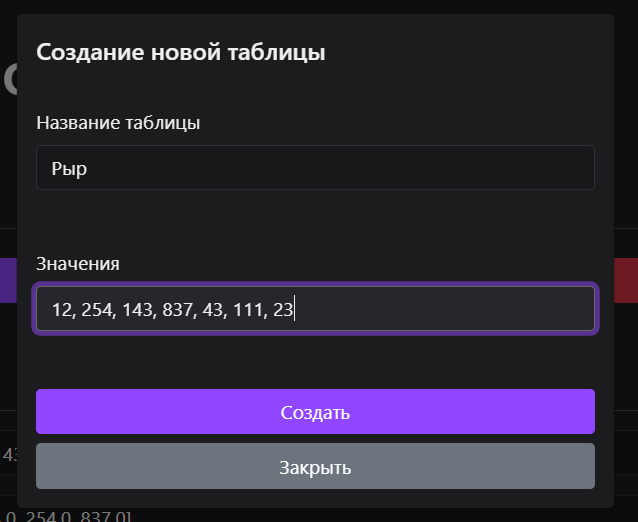


Рисунок 4 – Экранная копия реализации возможности ввода массива с клавиатуры

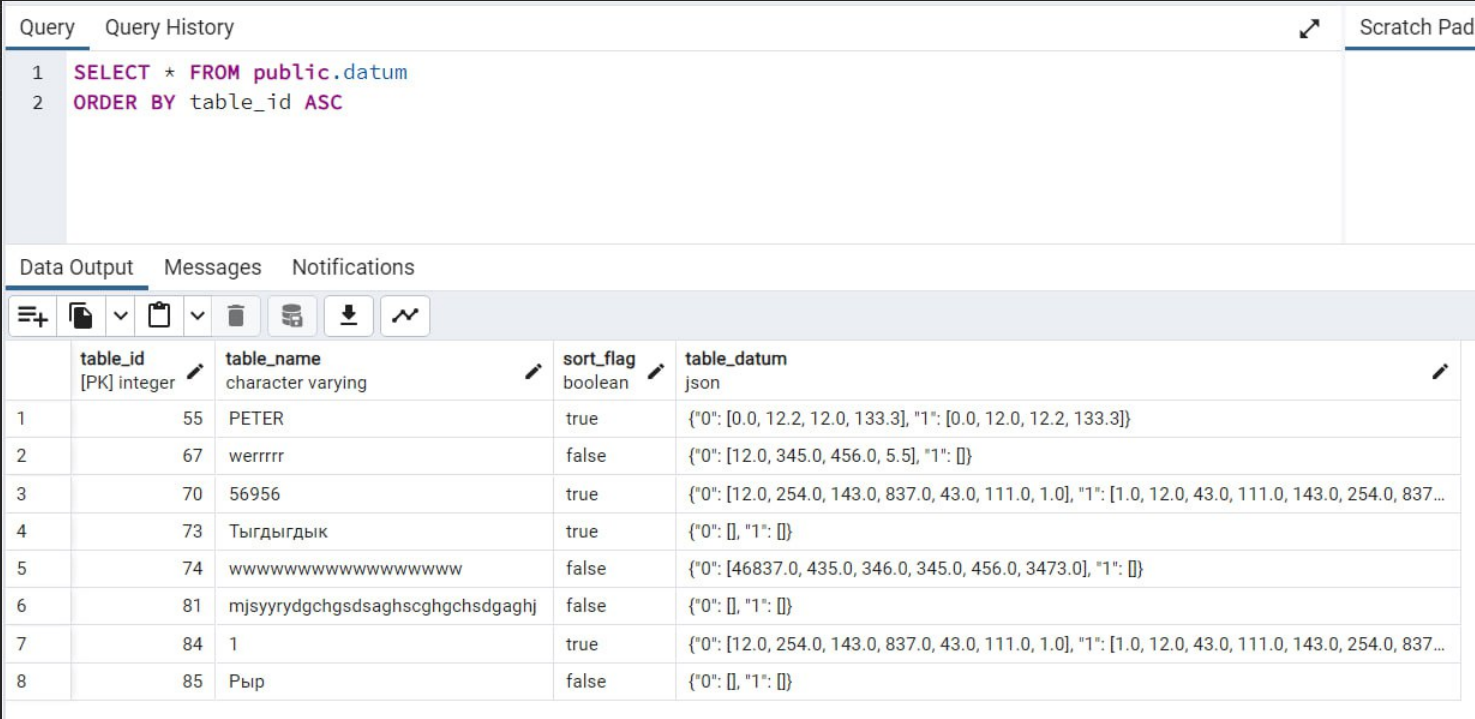


Рисунок 5 – Экранная копия реализации возможности сохранения массивов в БД на примере массива «Рыр»

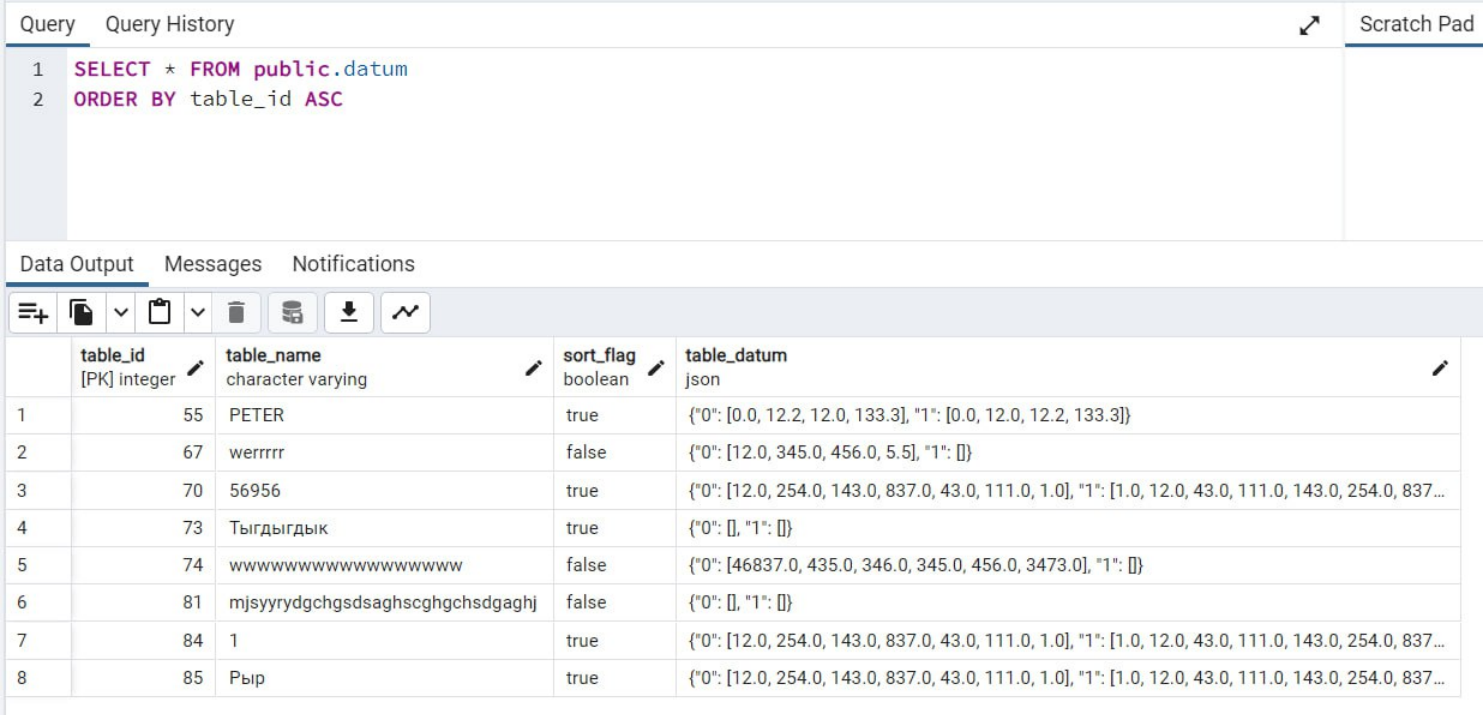


Рисунок 6 – Экранная копия реализации возможности сохранения массивов в БД на примере массива «Рыр»

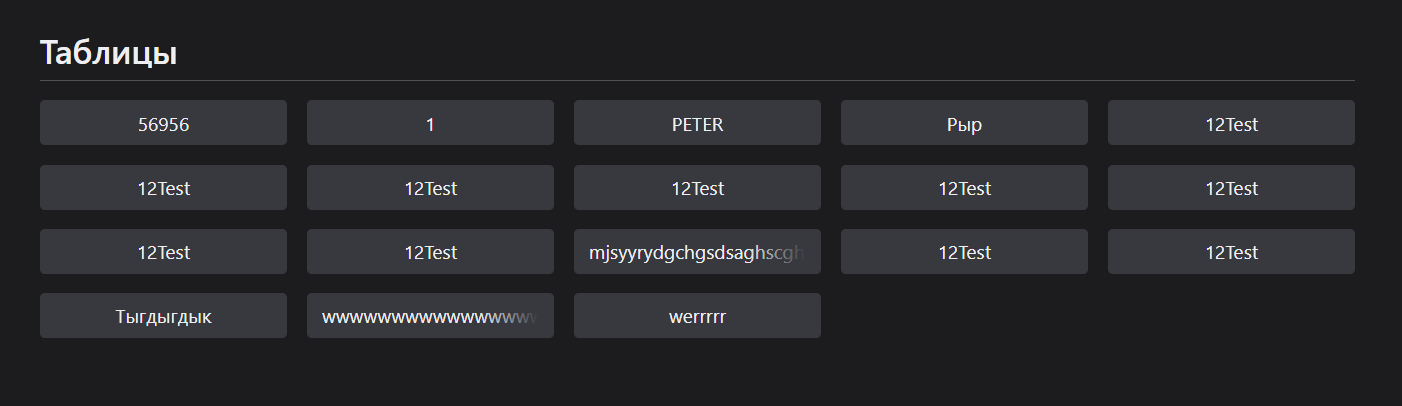


Рисунок 7 – Экранная копия реализации возможности вывода списка всех массивов (без возможности редактирования)

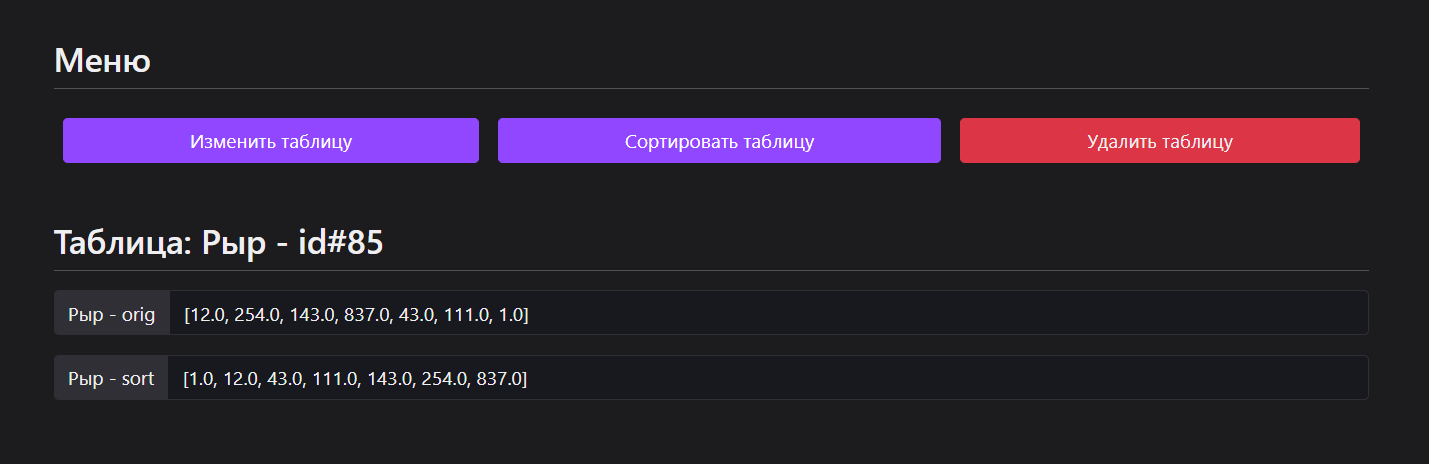


Рисунок 8 – Экранная копия реализации возможности вывода массива для дальнейшего редактирования и сортировки

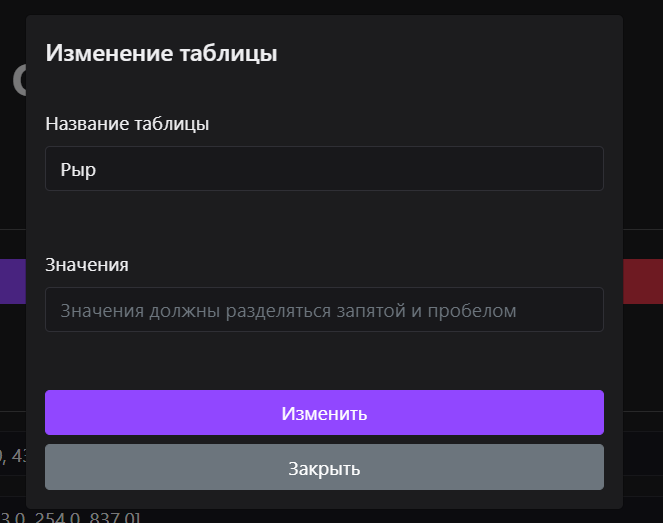


Рисунок 9 – Экранная копия реализации возможности редактирования массива

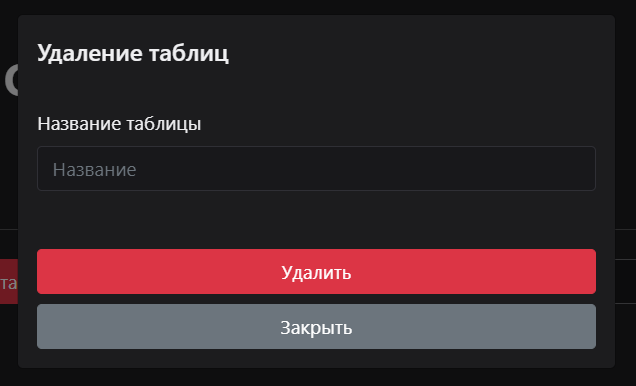


Рисунок 10 – Экранная копия реализации возможности удаления массивов

Также, при создании пользовательского интерфейса учитывалось его соответствие эвристикам ui/ux.

1. Видимость статуса системы: при наведении или нажатии соответствующие кнопки подсвечиваются, а также, при выводе массивов из БД и сохранении массивов, пользователю всегда дается информация о его действиях. (рис. 11)

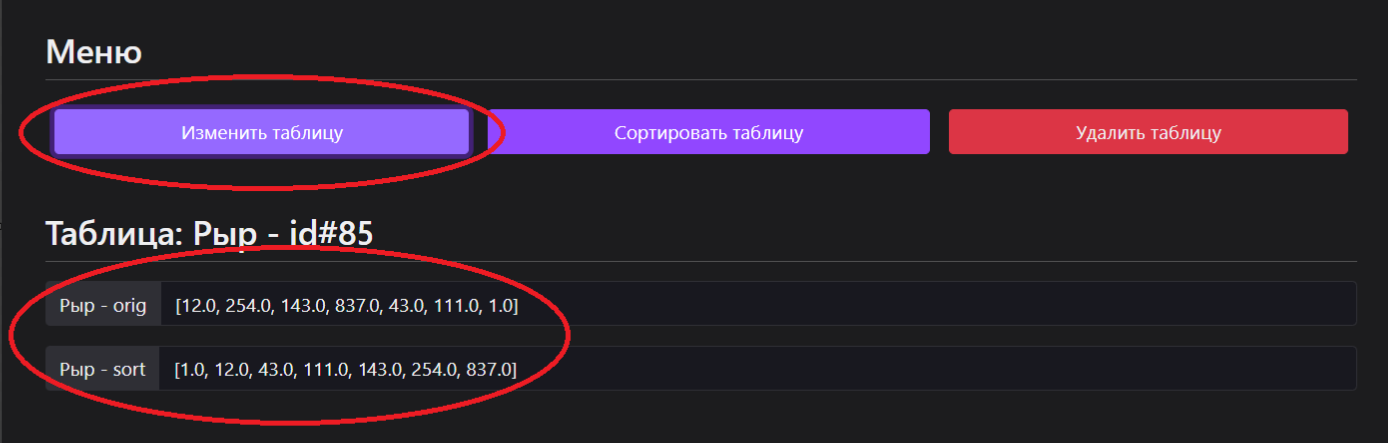


Рисунок 11 – Экранная копия реализации видимости статуса системы.

1. Соответствие между системой и реальным миром: в программе используются элементы реального мира, дающие пользователю контекст о происходящем, например, иконка луны или солнца (в зависимости от текущего оформления), которая обозначает функцию смены оформления страницы. (рис. 12)

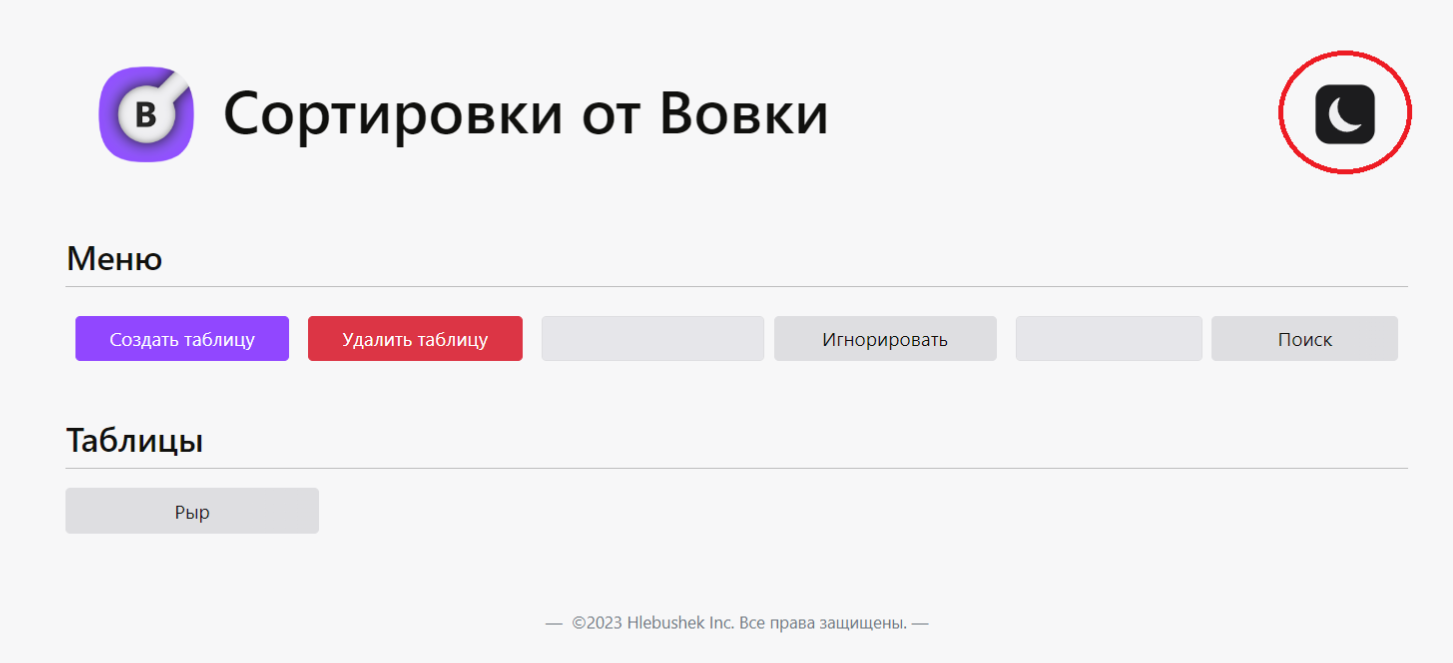


Рисунок 12 – Экранная копия реализации соответствия между системой и реальным миром.

1. Управляемость: в любой момент работы программы пользователь может выбрать функцию вывода или ввода данных, заново ввести числа, пересоздать числа и так далее. В моменте, когда у пользователя открывается окно подтверждения, у него также есть возможность отказаться и вернуться к предыдущему окну. (рис. 13)

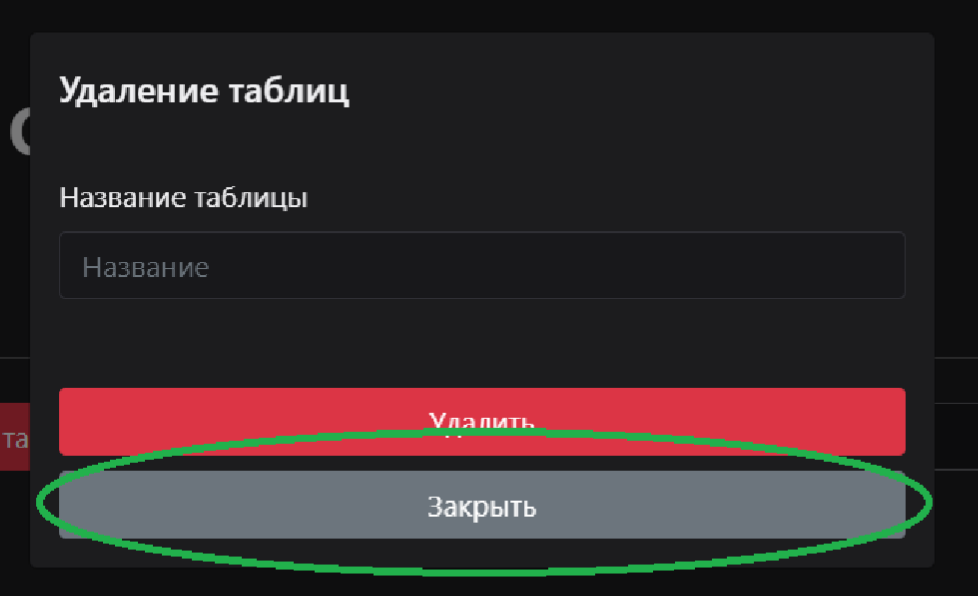


Рисунок 13 – Экранная копия реализации управляемости. (зеленый – кнопка отмены, позволяющая пользователю вернуться обратно).

1. Согласованность: пользователь будет взаимодействовать с привычным по виду интерфейсом, состоящим из простых кнопок. При нажатии на некоторые кнопки система выводит окна с более подробной информацией о действии, которое хочет совершить пользователь.
2. Гибкость: программа создана с предусмотренной возможностью некорректных действий пользователя. Так, при попытке добавления элементов, не представляющих собой числа система не добавит данные в массив. Также в данном поле ввода подписаны условия корректности значений (рис. 14).

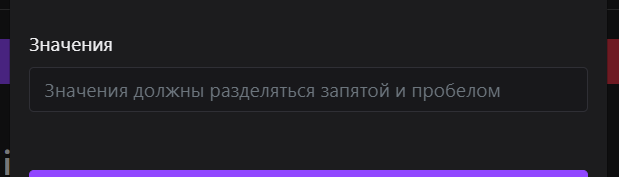


Рисунок 14 – Экранная копия реализации гибкости.

1. Минимализм в дизайне: весь интерфейс содержит минимальное количество различных цветов, которые несут, в основном, функциональный характер. Например, кнопки, выполняющие основные функции выделены яркими цветами, а дополнительные – нет (рис. 15), однако при наведении они становятся более яркими для наглядности выбора пользователя. Также, интерфейс содержит исключительно необходимую информацию, не перегружая пользователя множеством окон, подокон и другой ненужной информацией.

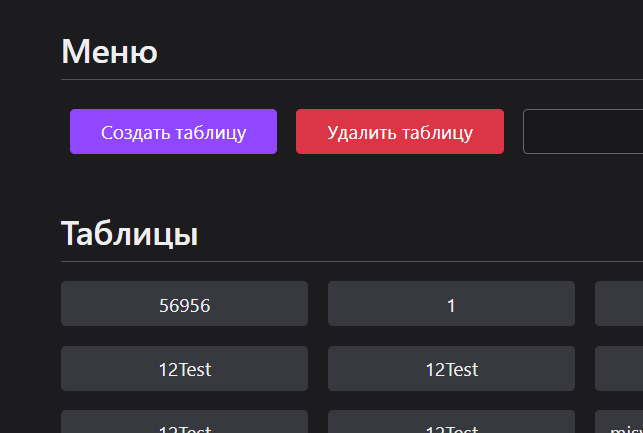


Рисунок 15 – Экранная копия реализации минимализма в дизайне.

## Выводы по заданию

В ходе выполнения задания были получены теоретические и практические навыки по разработке пользовательского интерфейса, веб-приложений, работы с базами данных, а также связывании программ, написанных на различных языках программирования.