Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Модели данных и системы управления базами данных»

ОТЧЕТ
к лабораторной работе №6
на тему:
«РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ИСПОЛЬЗОВАНИЕМ БАЗЫ ДАННЫХ»
БГУИР 6-05-0612-02

Выполнил студент группы 353503
КОХАН Артём Игоревич
РЯЗАНЦЕВ Алексей Владимирович

_____
(дата, подпись студента)

Проверил ассистент каф. информатики
КОЖЕМЯКО Евгения Александровна

_____
(дата, подпись преподавателя)

Минск 2025

1 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Реализация запросов по журналированию событий, реализация интерфейсов приложения. Результат работы: набор SQL-запросов с описанием, иллюстрации проволочных интерфейсов.

2 КРАТКИЕ ТЕОРИТИЧЕСКИЕ СВЕДЕНИЯ

Система журналирования событий является важным компонентом современных приложений, обеспечивающим отслеживание действий пользователей и системных процессов. В базах данных журналирование реализуется через создание специализированных таблиц для записи событий, которые фиксируют информацию о действиях пользователей, изменениях данных и системных операциях.

Основой для журналирования служит оператор INSERT, который добавляет записи в таблицу логов. Каждая запись обычно содержит идентификатор пользователя, тип выполненного действия, временную метку, описание события и дополнительную контекстную информацию. Для классификации событий по уровням важности используются ENUM-типы или справочные таблицы, определяющие категории событий такие как информационные, предупреждения и ошибки.

Реализация пользовательских интерфейсов приложения строится на основе проволочных прототипов (wireframes), которые представляют собой схематичное отображение интерфейса без детализации визуального оформления.

События журналирования могут включать авторизацию пользователей, выполнение критических операций с данными, ошибки ввода и другие значимые действия. Собранная информация используется для анализа работы системы, устранения проблем и улучшения пользовательского опыта.

3 ВЫПОЛНЕНИЕ РАБОТЫ

Разработка пользовательского интерфейса приложения велась на основе проволочных прототипов (wireframes), которые представляют собой схематичное отображение структуры и основных элементов интерфейса без детальной проработки визуального оформления. В рамках работы был создан проволочный интерфейс в Swagger (OpenAPI) для документирования и тестирования REST API.

Рисунок 3.1 – Swagger

Этот интерфейс описывает все доступные эндпоинты, их параметры, форматы запросов и ответов, а также позволяет интерактивно тестировать методы API непосредственно из браузера. Swagger-интерфейс включает описание операций для всех сущностей системы: управления пользователями, работы с абонементами, бронирования шкафчиков и записи на персональные тренировки. Для каждого эндпоинта определены HTTP-методы (GET, POST, PUT, DELETE), ожидаемые параметры запроса, структуры тел запросов и ответов, а также возможные коды HTTP-ответов. Такой подход обеспечивает четкую спецификацию API и упрощает интеграцию фронтенд-части приложения с бэкенд-сервисами.

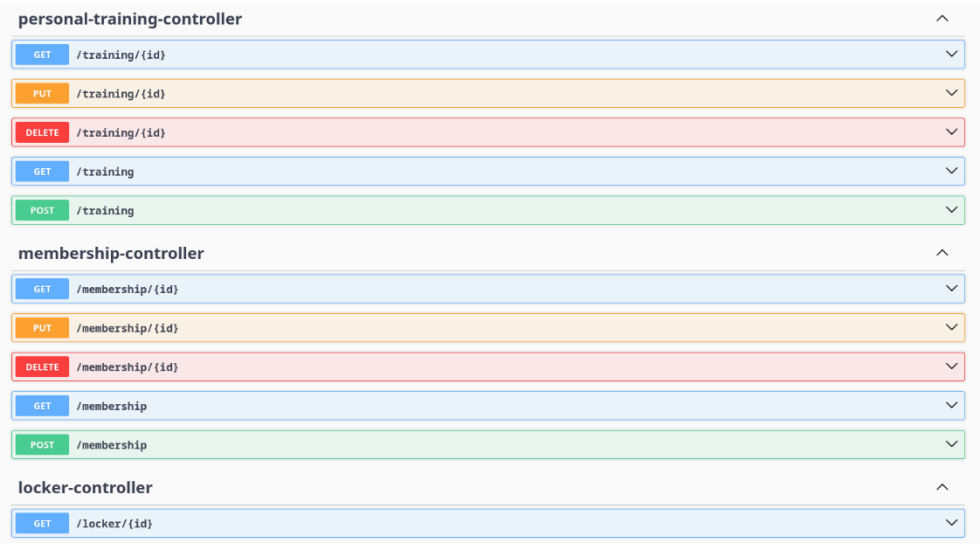

Рисунок 3.2 – Часть списка возможных запросов в swagger

В качестве примера воспользуемся GET-запросом получения всех возможных тренировок в системе.

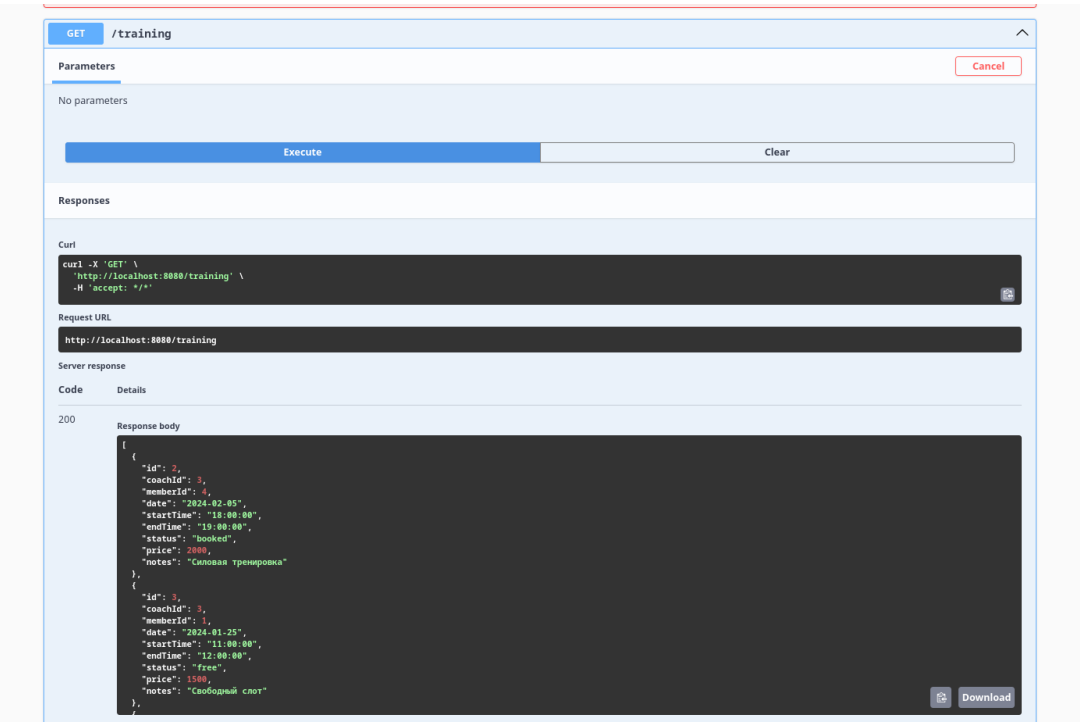

Рисунок 3.3 – GET-запрос тренировок спортивного зала

Реализация журналирования событий, представленная в листинге 3.1. Для записи событий в систему используется оператор INSERT, который добавляет записи в таблицу user_log. Данный запрос параметризован и принимает следующие значения: идентификатор пользователя (person_id), тип выполненного действия (action_type), описание события (action_description), временная метка (action_time), IP-адрес (ip_address) и уровень важности события (severity). Особенностью реализации является использование приведения типа ?::log_severity_enum для поля severity, что обеспечивает соответствие значения предопределенному ENUM-типу в базе данных, который классифицирует события по уровням важности такие как информационные, предупреждения и ошибки. Для извлечения записей журнала используется SELECT-запрос, который получает все поля таблицы user_log с сортировкой по времени события в порядке убывания (ORDER BY action_time DESC), что позволяет анализировать последние события в первую очередь.

Раз уж мы познакомились со swagger, то в качестве примеры выполним GET-запрос для получения логов со стороны админа в swagger.

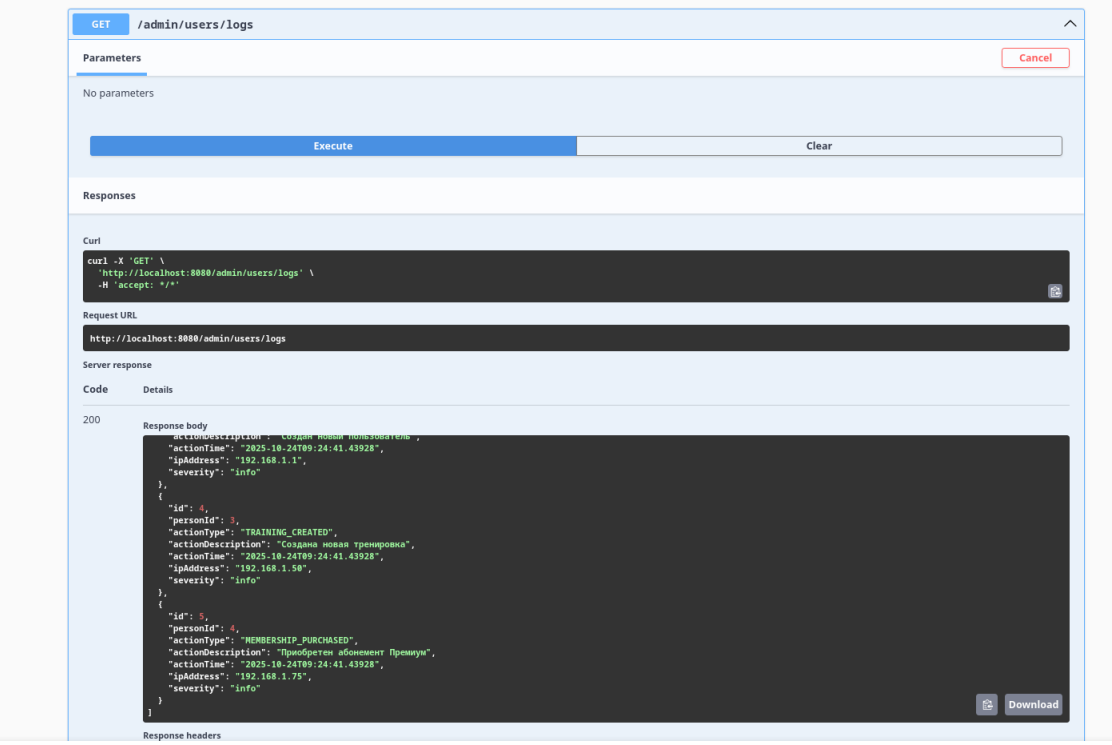

Рисунок 3.4 – GET-запрос логов

Листинг 3.1 – Журналирование событий

```
INSERT INTO user_log (person_id, action_type, action_description, action_time,
ip_address, severity)
            VALUES (?, ?, ?, ?, ?, ?::log_severity_enum)

SELECT id, person_id, action_type, action_description, action_time, ip_address,
severity FROM user_log ORDER BY action_time DESC
```

ВЫВОД

В рамках работы были созданы механизмы записи и отслеживания событий системы, а также разработан проволочный интерфейс с помощью Swagger для документирования и тестирования API.

Реализованная система журналирования обеспечивает надежную фиксацию всех значимых событий через оператор INSERT в таблицу user_log. Каждая запись журнала содержит идентификатор пользователя, тип действия, описание события, временную метку, IP-адрес и уровень важности. Использование ENUM-типа log_severity_enum позволяет классифицировать события по уровням важности (информационные, предупреждения, ошибки), что упрощает последующий анализ и фильтрацию записей.

Для извлечения записей журнала реализован SELECT-запрос с сортировкой по времени события в порядке убывания, что обеспечивает удобный просмотр последних событий системы.

Разработанный проволочный интерфейс в Swagger предоставляет полную документацию REST API системы, включая описание всех эндпоинтов для управления пользователями, абонементами, бронированием шкафчиков и персональными тренировками. Интерфейс позволяет интерактивно тестировать каждый метод API, проверять форматы запросов и ответов, а также анализировать коды HTTP-ответов.