МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”**

Факультет компьютерных наук

*Кафедра программирования и информационных технологий*

*Разработка портала для практических занятий по SQL.*

*Отчет по производственной практике*

09.03.04 - Программная инженерия

Зав. кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_*С.Д. Махортов, д-р физ.-мат. наук, профессор*

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*Е.В. Елфимова, 4 курс, д/о*

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*Д.И. Соломатин, ст. преподаватель*

Воронеж 2021

Оглавление

[ВВЕДЕНИЕ 3](#_Toc64122535)

[1. ПОСТАНОВКА ЗАДАЧИ 4](#_Toc64122536)

[2. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc64122537)

[2.1. Обзор существующих аналогов приложения 5](#_Toc64122538)

[2.2. Глоссарий 7](#_Toc64122539)

[2.3. Сценарии использования 8](#_Toc64122540)

[2.3.1. Неавторизованная работа 8](#_Toc64122541)

[2.3.2. Авторизованная работа (студент) 9](#_Toc64122542)

[2.3.3. Авторизованная работа (преподаватель) 10](#_Toc64122543)

[2.3.4. Администрирование сайта 11](#_Toc64122544)

[3. СРЕДСТВА РЕАЛИЗАЦИИ 12](#_Toc64122545)

[4. РЕАЛИЗАЦИЯ 13](#_Toc64122546)

[4.1. Архитектура приложения 13](#_Toc64122547)

[4.2. Структура приложения 15](#_Toc64122548)

[**4.3. Диаграмма классов** 16](#_Toc64122549)

[5. ИНТЕРФЕЙС 20](#_Toc64122550)

# ВВЕДЕНИЕ

В настоящее время студентам факультета ФКН впервые знакомящимся с языком SQL не всегда бывает просто. Для настройки базы данных для студентов преподавателям приходится прикладывать немало усилий. Как известно, практики в классах не всегда бывает достаточно, и часто возникает потребность поупражняться дома на своем собственном ноутбуке в написании различных запросов, чтобы лучше понять тонкости языка запросов SQL. Для этого студентам так же нужно приложить немало усилий, устанавливать на своем устройстве сервер баз данных, будучи еще даже не знакомым с элементарными запросами выборки, согласитесь, будет очень сложно. У некоторых студентов все желание изучать язык SQL может быть потерянно при такой установке.

В связи с этими обстоятельствами были изучены существующие решения данной проблемы и предложен новый подход, освобождающий студентов от сложной для них предварительной подготовки базы данных и позволяющее преподавателям отслеживать прогресс своих учеников.

На основе данного подхода было реализовано веб-приложение для факультета для изучения языка SQL и проведения практических занятий по данной тематике.

# ПОСТАНОВКА ЗАДАЧИ

Задачей данной работы было создание веб приложения - портала для практических занятий по SQL.

Приложение должно быть доступно для студентов ФКН начинающих изучать язык SQL, а также преподавателей по практике, ведущих у них соответствующий предмет. Преподавателям приложение должно позволять создавать различные базы данных и создавать курсы с тестами, которые будут предлагаться для решения студентам. Студентам так же должно быть доступно написание и исполнение запросов к созданным преподавателями базам данных.

Сайт должен иметь адаптивную верстку и корректно отображаться как на мониторе, так и на мобильном устройстве.

# 2. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

# 2.1. Обзор существующих аналогов приложения

(В аналоги можно добавить лайв SQL, там можно выполнять запросы, но каждый раз надо создавать свою бд и тд)

Похожих аналогов на данный момент не существует. На факультете ФКН сейчас для тренировок в написании SQL запросов используется среда Oracle SQL Developer, а для проверки знаний студентов используется Электронный университет ВГУ.

Oracle SQL Developer - интегрированная среда разработки на языках SQL и PL/SQL, с возможностью администрирования баз данных, ориентированая на применение в среде Oracle Database.

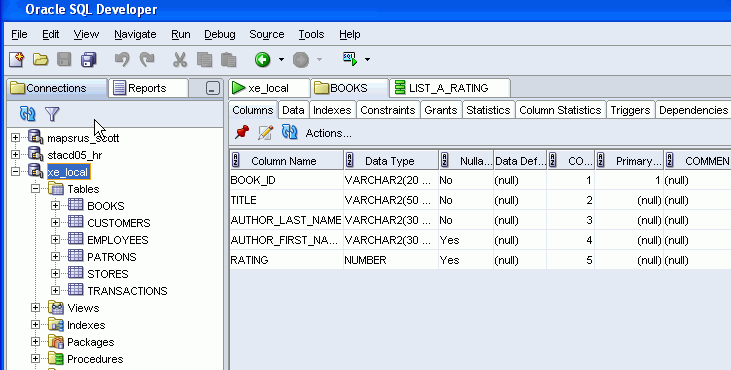


Рисунок 1 – Интерфейс Oracle SQL Developer

Преимущества:

* широкий спектр функциональности;
* полностью бесплатное приложение;
* может использоваться как под Windows, так и под Linux;

Недостатки:

* большую ресурсоемкость;
* необходимость иметь доступную базу данных для работы с ней.

Электронный университет ВГУ – образовательный портал, предназначенный для студентов и работников Воронежского государственного университета. В качестве заданий на сайте Электронного университета ВГУ для курсов по изучению SQL студентам предлагаются задания, требующие написания запросов. При этом сайт не позволяет студентам выполнять запросы и просматривать результат их выполнения. Такое ограничение хорошо подходит для контроля знаний, но не для еженедельных практических занятий по написаниям SQL запросов.

На рисунке 2 можно увидеть интерфейс главной страницы Электронного университета ВГУ.

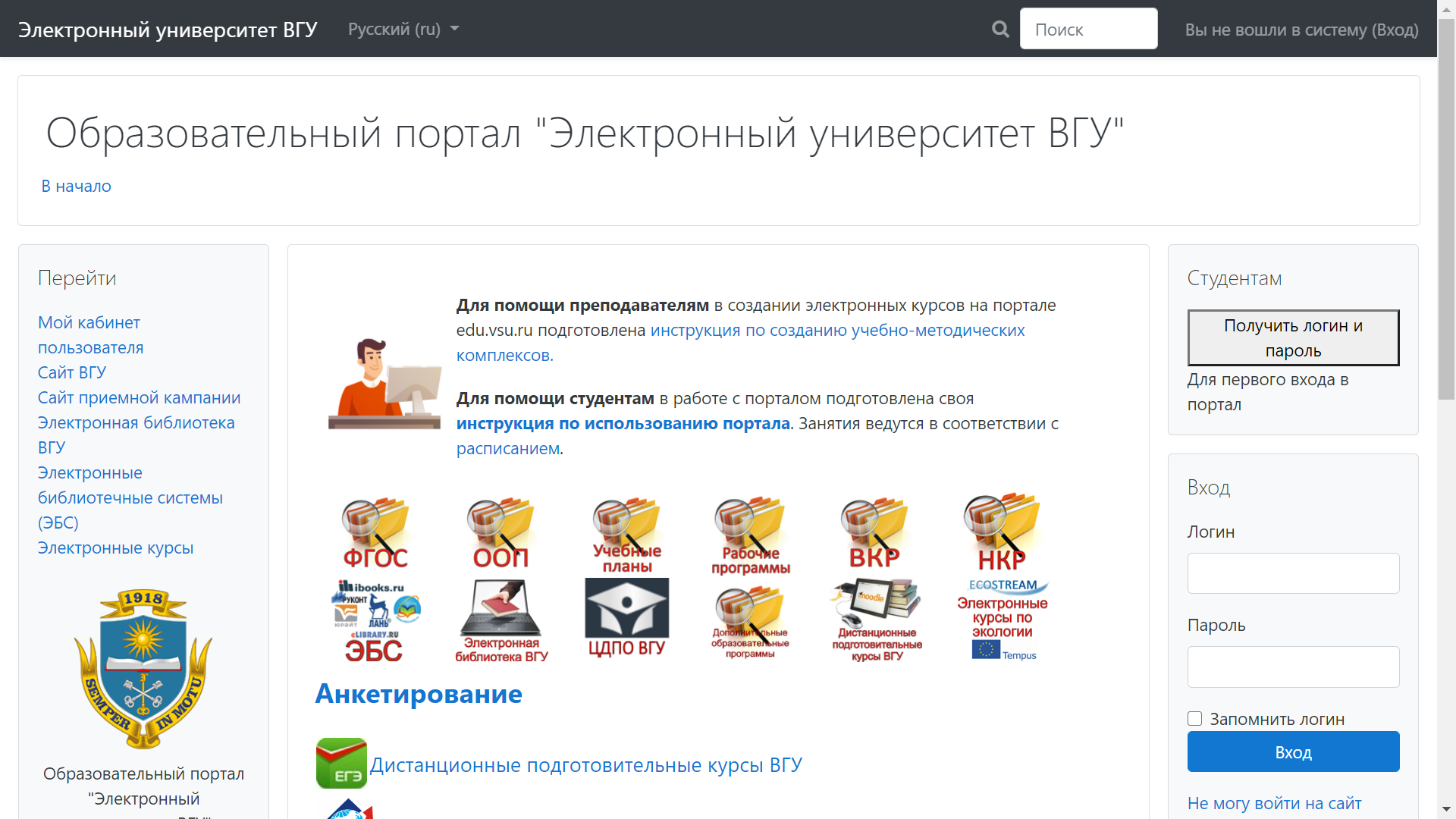


Рисунок 2 – Главная страница Электронного университета ВГУ

Преимущества:

* возможность создавать тесты и курсы для проверки знаний студентов;
* возможность просмотра успеваемости по курсам;
* наличие чатов с преподавателями.

Недостатки:

* отсутствует возможность взаимодействия с базой данных для исполнения запросов студентами.

# 2.2. Определение модели описания базы данных (может поменять на что то вроде описание процесса работы с базами данных)

Ключевым элементом легкости системы в обращении – отсутствие необходимости размещения локальных или распределенных СУБД. Такая легкость достигается за счет использования AlaSQL – легкой клиентской базы данных расположенной в оперативной памяти и работающей на веб-сервере Node.js. JavaScript используется при создании запросов, а сама базы быстрее других популярных JavaScript SQL баз данных: SQL.js, Linq.

Преподаватель для загрузки базы данных должен загрузить XLSX файл с определенной структурой, которую можно рассмотреть на рисунке ХХ.

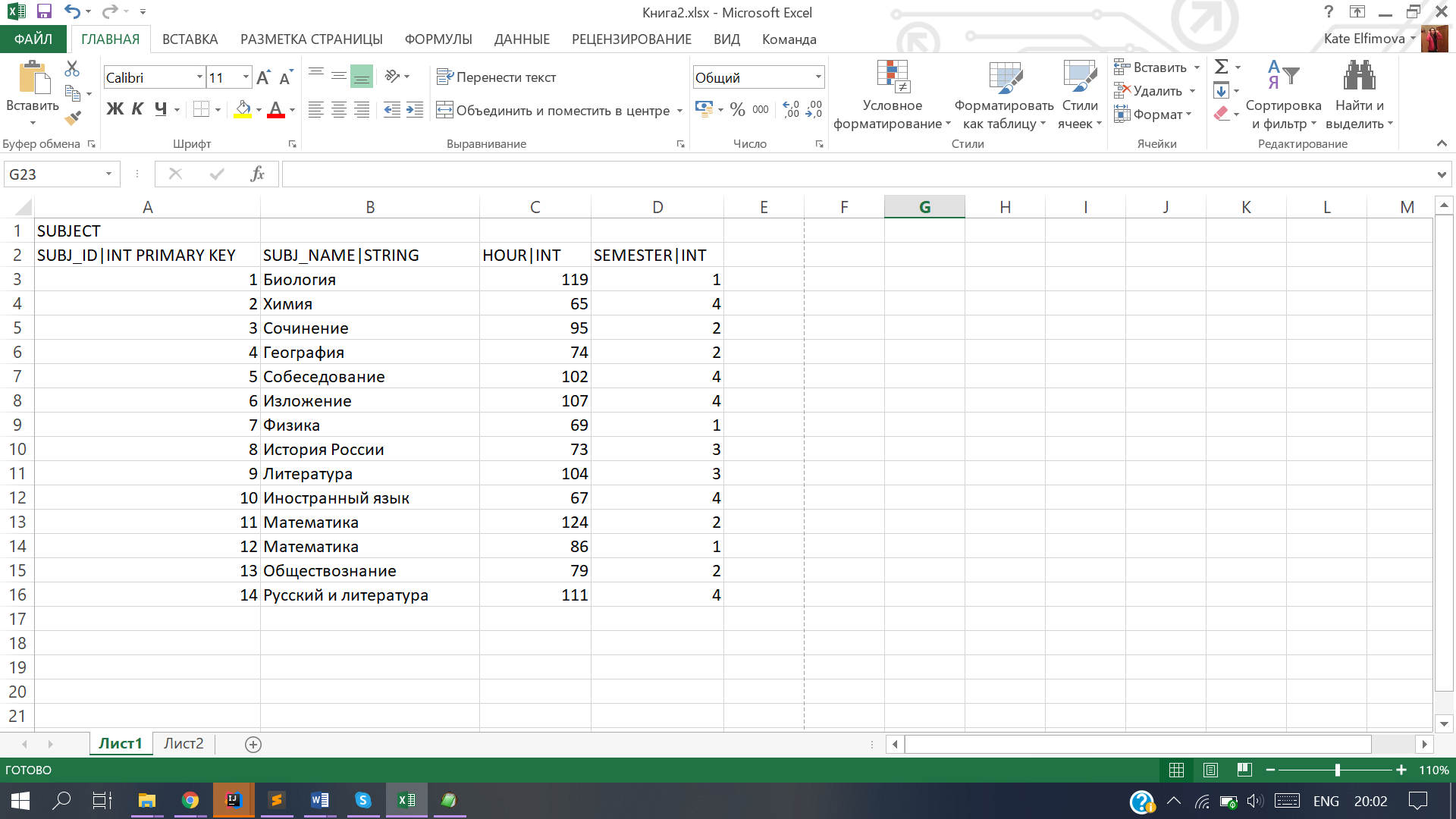


Рисунок ХХ – Пример файла с базой данных

На рисунке ХХ представлен пример форматирования файла с базой данных. На листе в файле должна располагаться одна таблица. В клетке А1 каждого листа файла должно быть записано название таблицы. В клетках линии 2 должны быть записаны названия столбцов таблицы. Для указания типа столбца необходимо поставить знак «|» без пробелов и написать тип столбца и необходимые ограничения. Ниже под соответствующими столбцами должны присутствовать записи таблицы. Название таблицы и названия колонок не должны содержать пробельных символов.

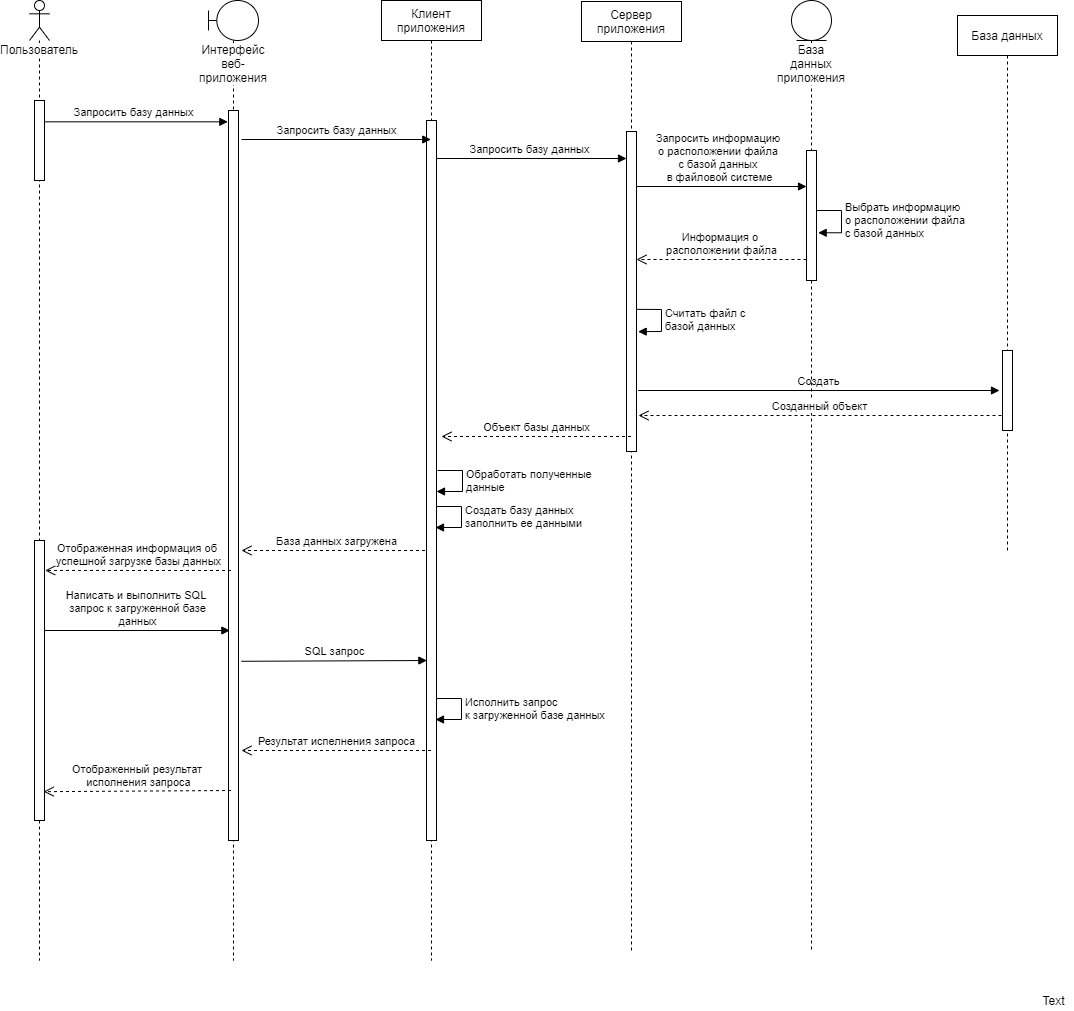
Описанный ранее файл должен быть загружен через интерфейс веб приложения. Перед загрузкой файла в приложение базу данных можно предварительно проверить. Для этого файл с базой данных будет считан, база данных будет создана и заполнена на клиенте приложения.

Для предоставления студентам или преподавателям возможности дальнейшей работы с базой данных, файл с базой данных должен быть загружен и передан на сервер приложения. На бекэнде файл сохраняется на файловой системе, а расположение файла и прочая необходимая информация сохраняются в базу данных.

При запросе работы с определенной базой бекэнд приложение находит необходимый файл, извлекает из него необходимые структуры, такие как таблицы, столбцы, строки, и, возвращает фронтэнд приложению в качестве ответа на запрос в формате json. Фронтэнд приложение разбирает структуру json структуру, выделяя необходимые элементы: таблицы, столбцы, записи, после чего создает для пользователя базу данных, выполняет создание таблиц с помощью исполнения запросов CREATE TABLE, и заполняет их записями с помощью исполнения операции вставки INSERT.

Для пользователя перешедшему на страницу базы данных, к которой он хочет написать и исполнить определенные запросы, все описанные ранее операции будет незаметны, но они будут исполняться каждый раз при перезагрузке страницы.

Пользователю будет доступно написание и исполнение DML запросов. При этом необходимо помнить, что результаты запросов вставки, обновления и удаления будут сохранены в базе до момента перезагрузки веб-страницы. Сохранение внесенных изменений в базу данных пользователя излишни для данной тематики, так как база данных используется для тренировки студентов в написании SQL-запросов. Намного более важным аспектом является сохранение исходной базы данных в первозданном виде, отсутствие возможности стереть записи на которых тренируются все студенты. Как уже было отмечено ранее, для отката к первоначальной версии базы данных, которая была загружена преподавателем, какие бы изменения не были внесены, перезагрузки страницы будет достаточно.



Написать про все возможные типы столбцов в базе и какими словами их называть, составить для этого таблицу.

# 2.2. Глоссарий

Неавторизованный пользователь – пользователь, который не произвел вход на сайте.

Авторизованный пользователь – пользователь, наделенный дополнительными правами и возможностями в системе по сравнению с неавторизованным пользователем.

Студент – авторизованный пользователь в системе имеющий возможность писать и исполнять запросы к базам данных, решать тесты по курсам и отправлять их на проверку преподавателю.

Преподаватель – авторизованный пользователь в системе имеющий права на создание курсов по SQL и тестов к ним. Преподаватель может загружать базы данных в приложение, а также проверять решенные студентами тесты.

Администратор – авторизованный пользователь, наделенный дополнительными правами и возможностями по администрированию системы.

Backend – программно-аппаратная часть сервиса.

Frontend – клиентская сторона пользовательского интерфейса к программно-аппаратной части сервиса.

Одностраничное приложение – это веб-приложение, использующее единственный HTML-документ как оболочку для всех веб-страниц и организующий взаимодействие с пользователем через динамически подгружаемые HTML.

Фреймворк - программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

API (Application Programming Interface) – интерфейс программирования приложений, позволяющий сервисам взаимодействовать, получать доступ и обмениваться данными.

DML (Data Manipulation Language) – язык управления, семейство компьютерных языков, используемых в компьютерных программах или пользователями баз данных для получения, вставки, удаления или изменения данных в базах данных.

Курс – ресурс, аккумулирующий в себе тесты по определенной тематике.

Тест – набор вопросов, созданных автором курса.

HTTPS –

REST -

# 2.3. Сценарии использования

На рисунке 3 можно увидеть диаграмму вариантов использования, на которой указаны все роли в системе, а также отношения между ними.

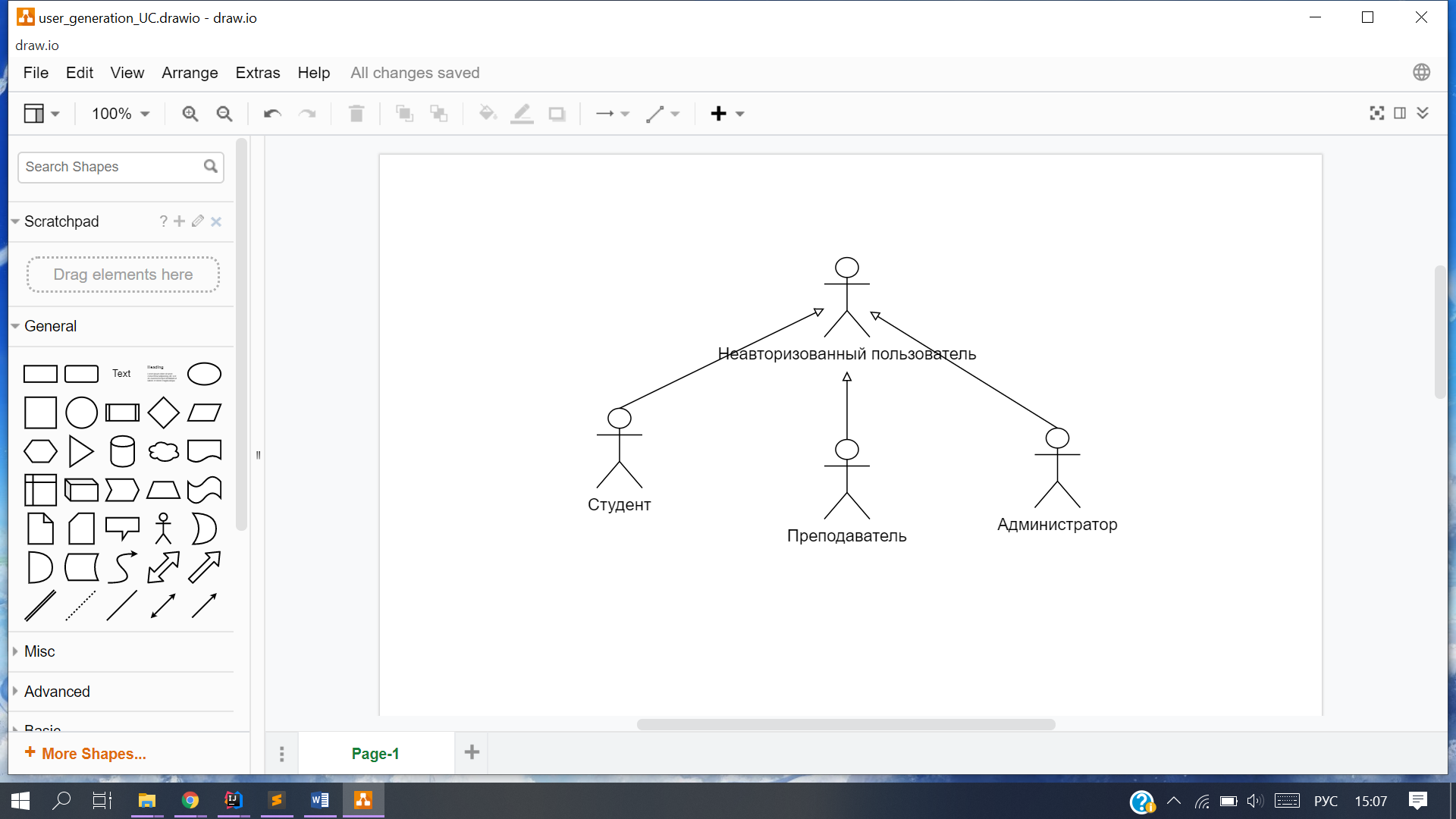


Рисунок 3 – Роли пользователей в системе

В системе предусмотрены следующие роли пользователей: неавторизованный пользователь, студент, преподаватель и администратор.

Рассмотрим более подробно все возможности, доступные для каждой роли в системе.

# 2.3.1. Неавторизованная работа

На рисунке 4 показаны варианты использования при неавторизованной работе.

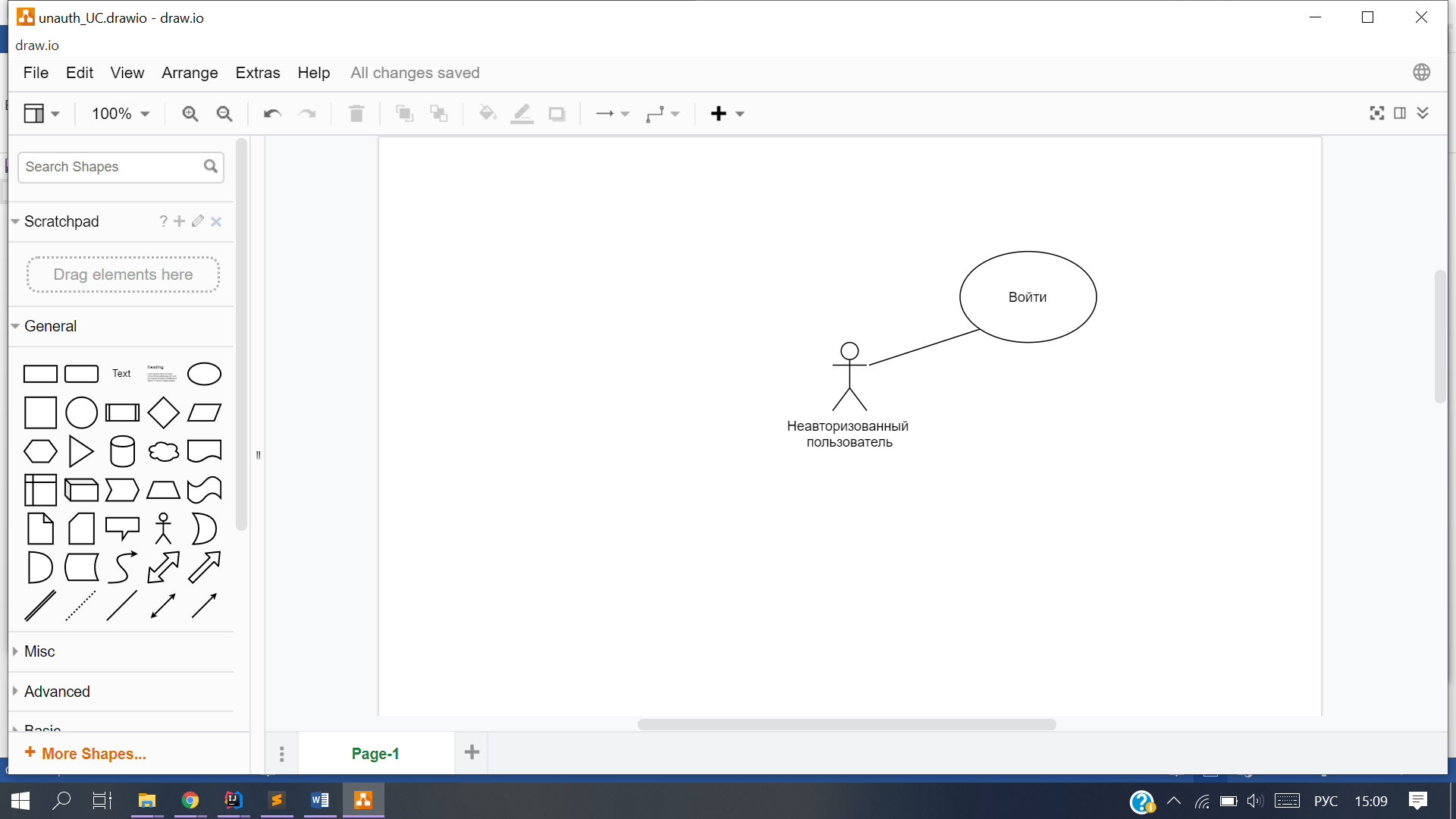


Рисунок 4 – Диаграмма вариантов использования для неавторизованного пользователя

Неавторизованный пользователь может произвести вход в систему, введя свой логин и пароль. Весь остальной контент сайта скрыт от пользователя до момента его успешной авторизации.

# 2.3.2. Авторизованная работа (студент)

На рисунке 5 представлена диаграмма вариантов использования при авторизованной работе пользователя с ролью студента в системе.

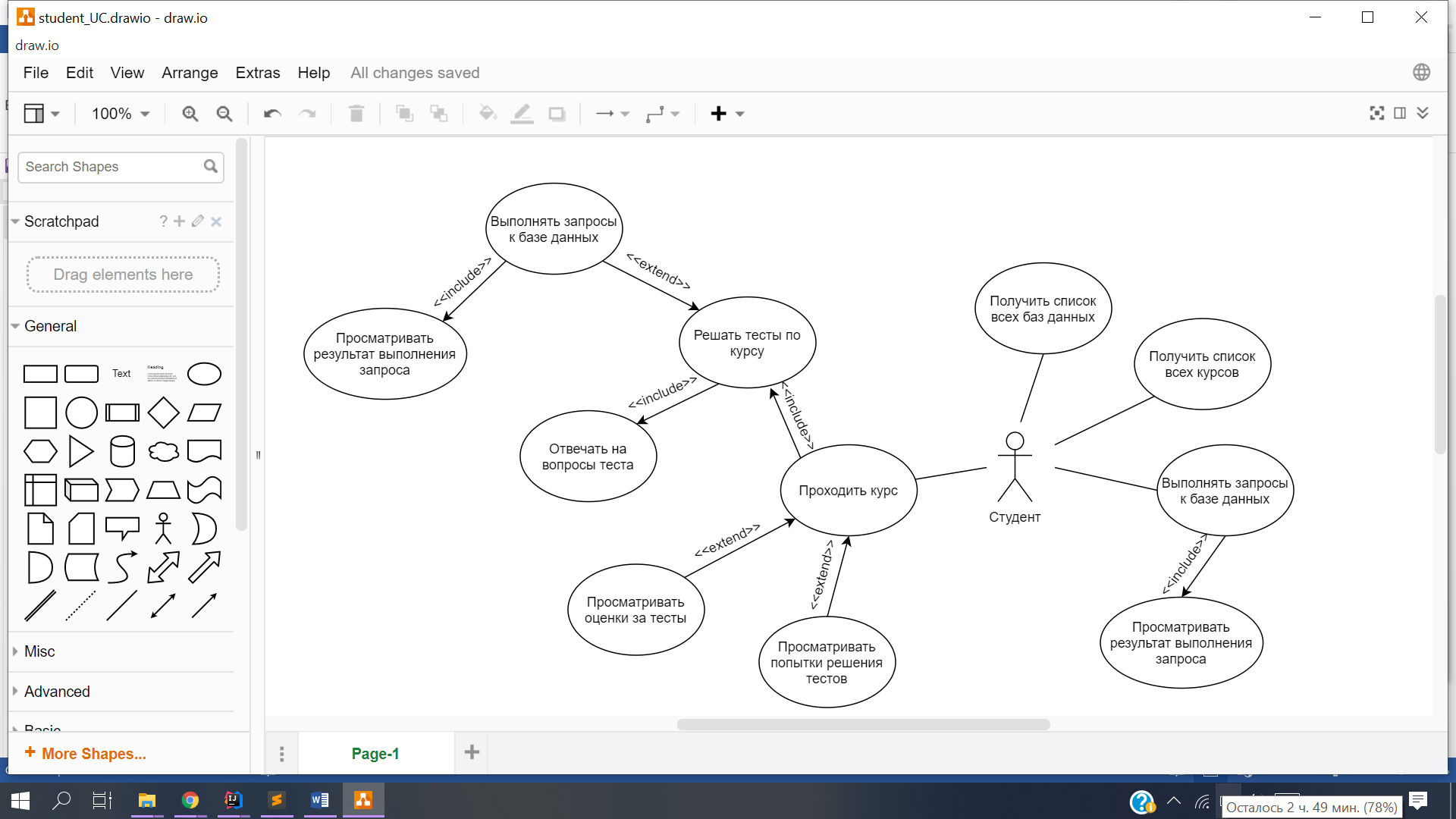


Рисунок 5 – Диаграмма вариантов использования для студента

Для студента доступен просмотр всех присутствующих в приложении баз данных к которым студент может выполнять запросы и просматривать результат их выполнения. Эти действия будут позволять студенту упражняться в написании SQL запросов без необходимости установки и запуска сервера баз данных на своем ноутбуке или компьютере.

Студент может просматривать список всех имеющихся в системе курсов и тестов которые содержатся в курсах. Студент может решать данные тесты, давая в качестве ответов на вопросы запросы к базе данных указанной в тесте. Для проверки качества запросов, студент может выполнить их там же на странице прохождения теста и проверить их результат.

Также студент может просматривать все свои попытки решения тестов, где также может увидеть поставленную оценку за определенную попытку преподавателем.

# 2.3.3. Авторизованная работа (преподаватель)

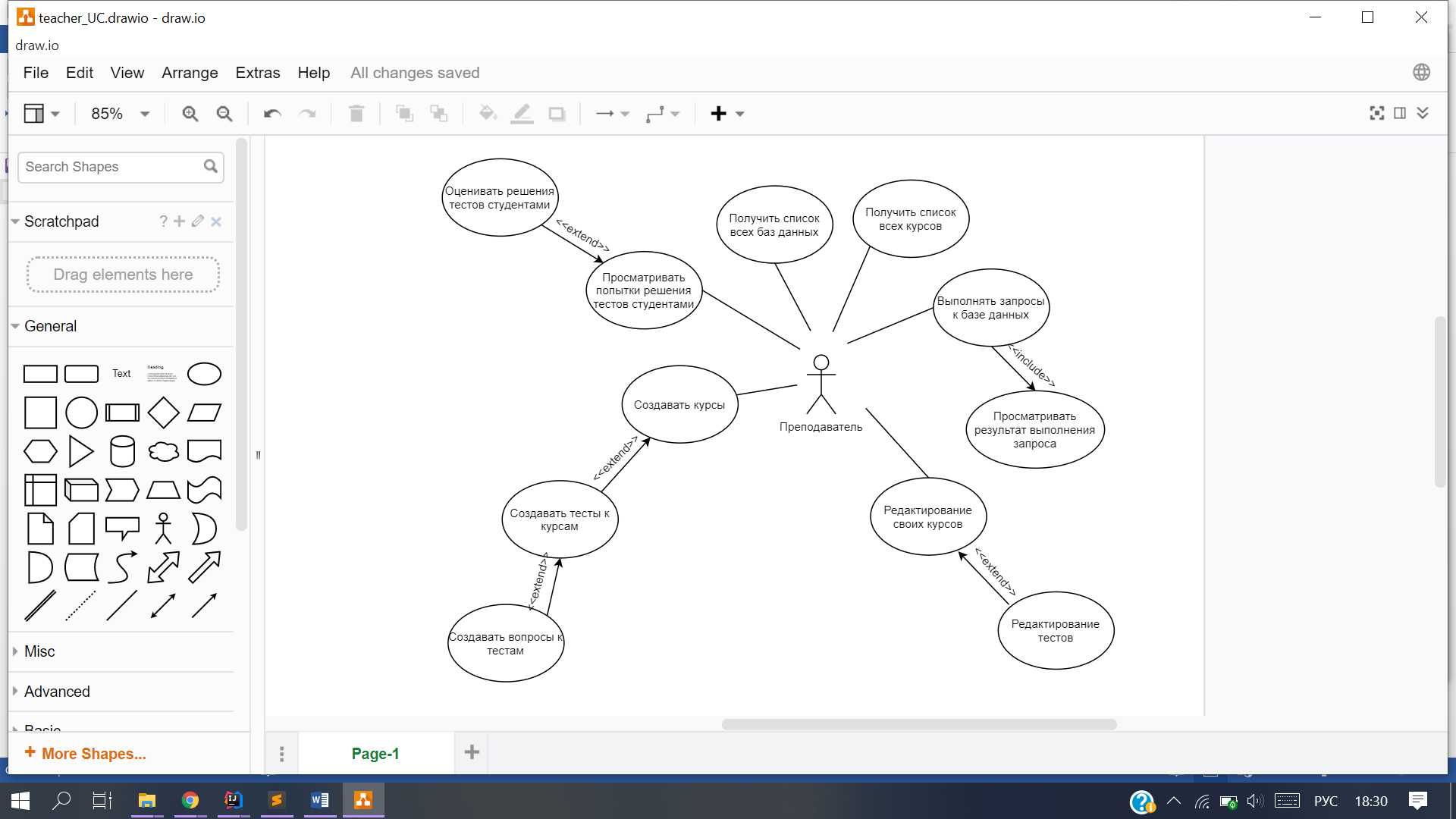


Рисунок 6 – Варианты использования для преподавателя

Преподаватель имеет права на загрузку в приложение базы данных, которая должна быть составлена по определенным описанных на сайте правилам в файле формата .xlsx. Преподаватель также как и студент может писать запросы к загруженным базам данных в приложения, исполнять их и просматривать результат выполнения.

Созданные курсы преподаватель может редактировать и удалять. Преподаватель может редактировать и удалять лишь курсы, автором которых он является. При создании тестов к курсам, автор создает набор вопросов и правильных ответов к ним.

После того как тесты курса преподавателя начнут решать студенты, преподаватель сможет просматривать их попытки решения тестов и выставлять за них баллы.

# 2.3.4. Администрирование сайта

На рисунке 7 представлена диаграмма вариантов использования для администратора.

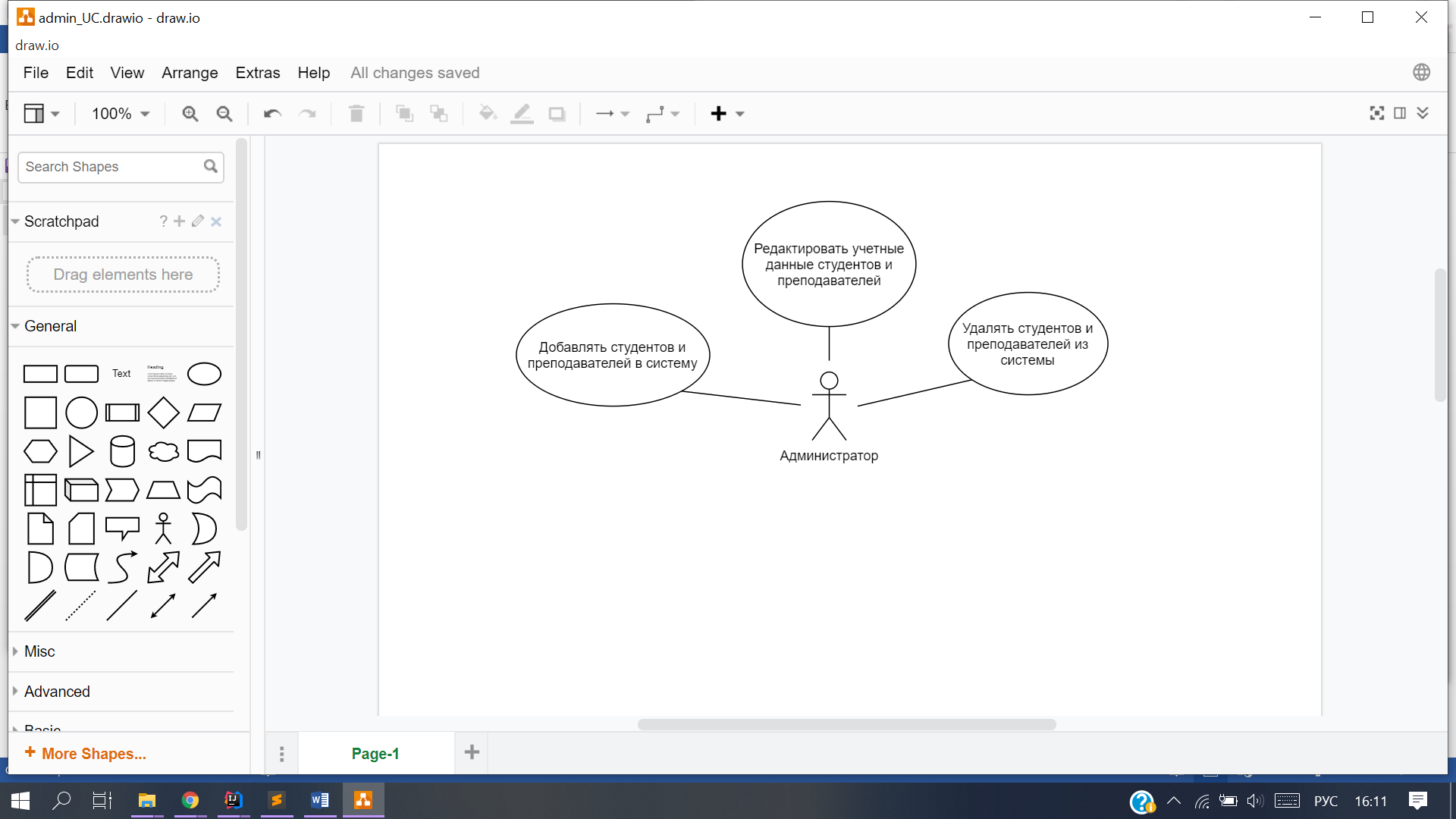


Рисунок 7 – Варианты использования для администратора

Каждый администратор может получить список всех пользователей, зарегистрированных в системе. Он может регистрировать в системе новых пользователей, указывая для каждого роль – студент или преподаватель. Администратор может удалять студентов или преподавателей из системы или редактировать их имена, логины, менять пароли.

# 3. СРЕДСТВА РЕАЛИЗАЦИИ

Рассмотрим стек используемых технологий, используемый для реализации данного проекта.

Для реализации frontend приложения использовались:

* Angular (версия 5) - открытая платформа для разработки веб-приложений;
* TypeScript (версия 2.5.3) - язык программирования, используемый для разработки веб-приложений, расширяющий возможности JavaScript.
* AlaSQL - легковесная клиентская база данных SQL в памяти, предназначенная для работы в браузере и на Node.js
* Ace - встраиваемый редактор кода, написанный на JavaScript, используемый для подсветки SQL синтаксиса при написании запросов.
* Bootstrap (версия 4) - открытый HTML, CSS и JS фреймворк для быстрой вёрстки адаптивных сайтов.

Для реализации backend приложения использовались:

* Java (версия 8) - строго типизированный объектно-ориентированный язык программирования;
* Spring Boot (версия 2.2.4) - фреймворк, позволяющий создавать приложения на основе Spring, для таких приложений требуется минимальная конфигурация;
* Spring Security - Java фреймворк, предоставляющий механизмы построения систем аутентификации и авторизации, а также другие возможности обеспечения безопасности;
* Hibernate - популярная реализация спецификации JPA для решения задач объектно-реляционного отображения (ORM);
* СУБД - PostgreSQL.

# 4. РЕАЛИЗАЦИЯ

# 4.1. Архитектура приложения

По выбранной теме было реализовано одностраничное клиент серверное приложение.

Важно отметить, что для взаимодействия распределенных приложений в сети (frontend и backend), использовался REST архитектурный стиль, то есть frontend приложение взаимодействует с backend по HTTP протоколу, и передает данные в формате JSON.

На рисунке 8 представлена схема архитектуры реализованного приложения.

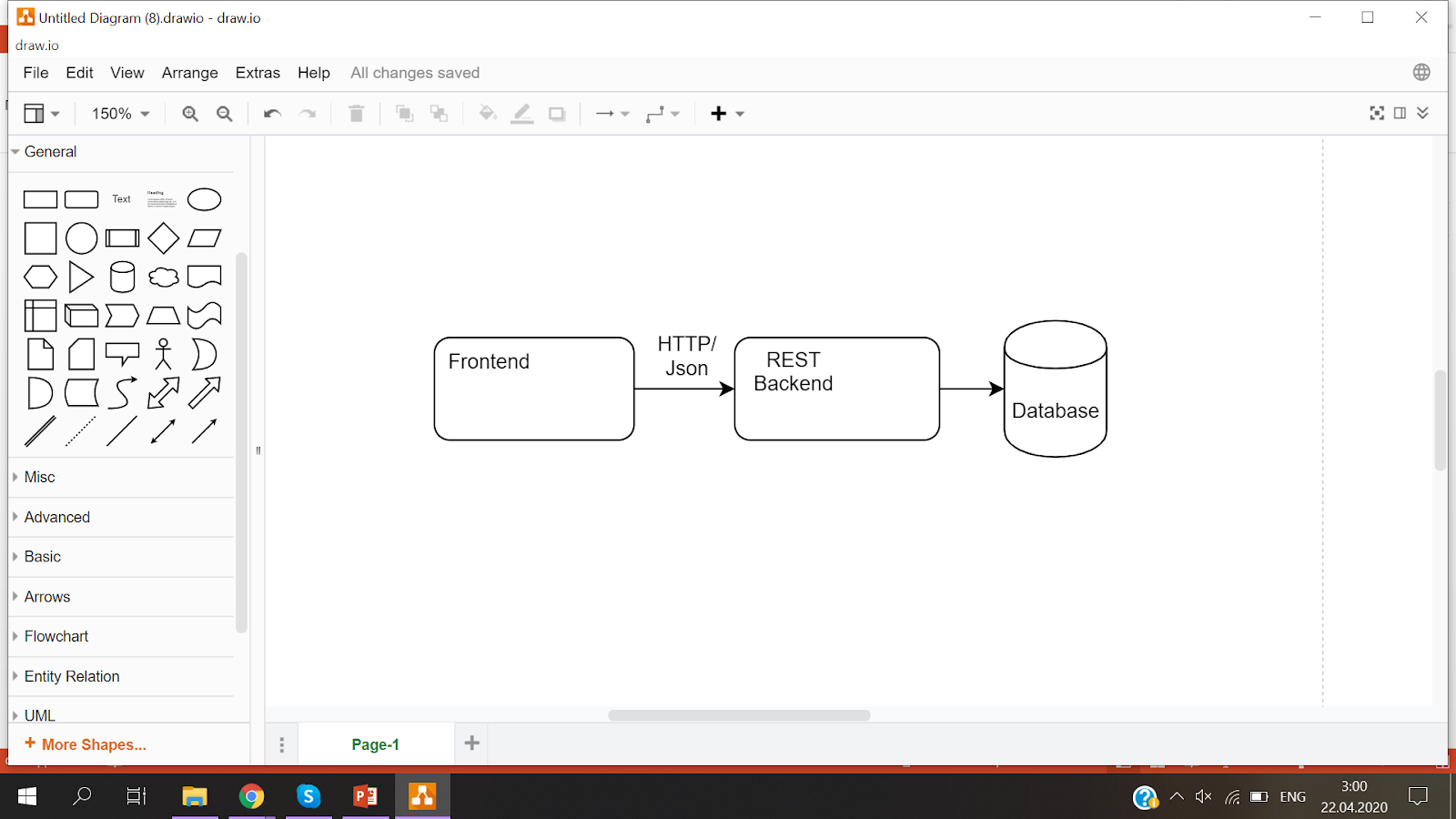


Рисунок 8 – Схема архитектуры приложения

Рассмотрим более подробно архитектуру frontend приложения, ее схема представлена на рисунке 9.

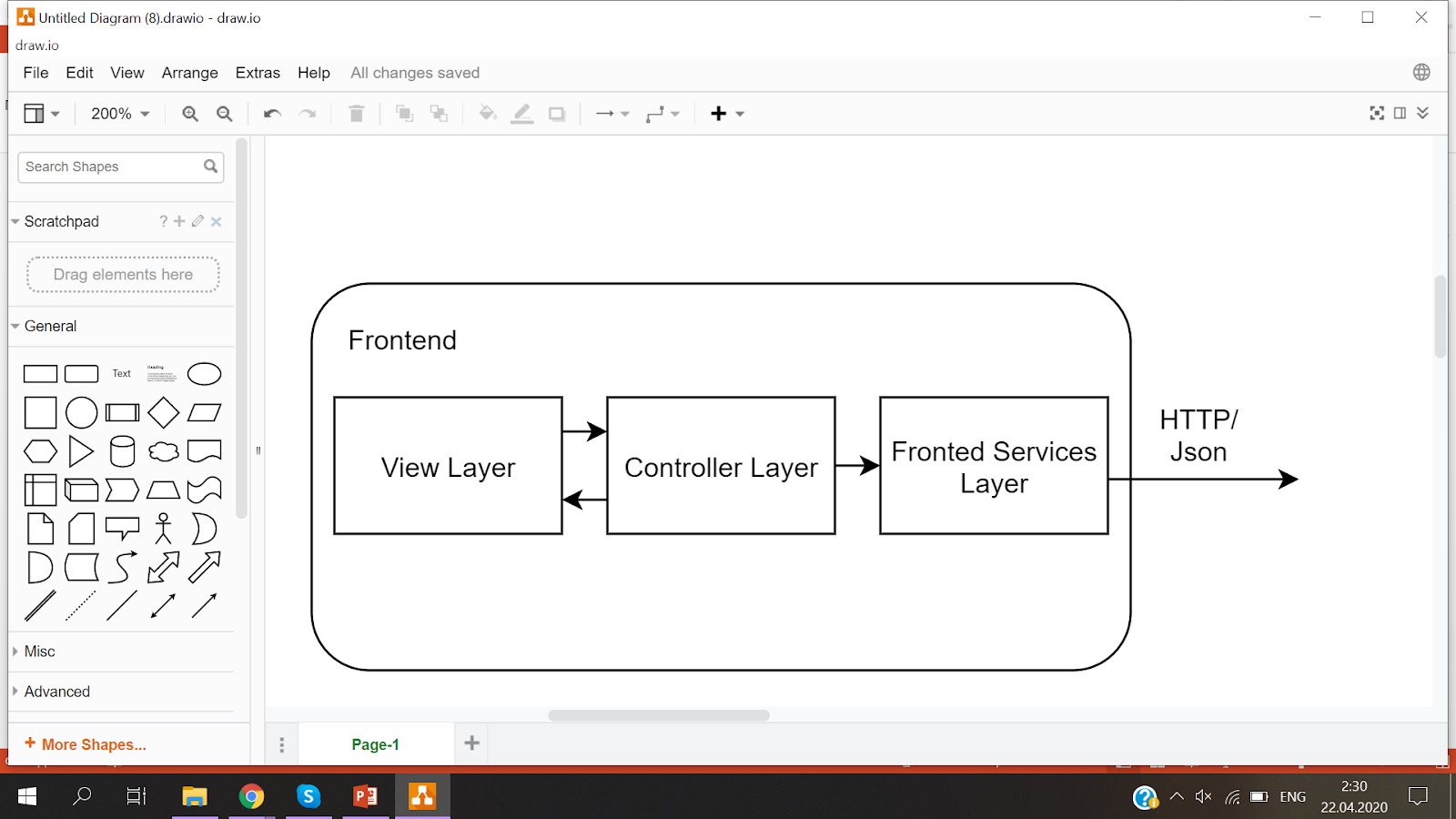


Рисунок 9 – Схема архитектуры frontend приложения

Как видно из рисунка frontend приложение реализует паттерн MVC. Рассмотрим более конкретно уровни frontend приложения:

* уровень представления (View Layer) состоит из HTML-шаблонов, таблиц CSS соответствующих различным компонентам пользовательского интерфейса;
* уровень управления (Controller Layer) состоит из контроллеров Angular, склеивающий вместе данные, извлекаемые, соответственно, с сервера и из представления. Контроллер инициализирует модель представления и определяет, как представление должно реагировать на изменения модели и наоборот;
* уровень сервисов (Frontend Services Layer) - набор сервисов Angular, которые могут взаимодействовать с REST API.

Далее, рассмотрим архитектуру backend приложения, ее схема представлена на рисунке 10.

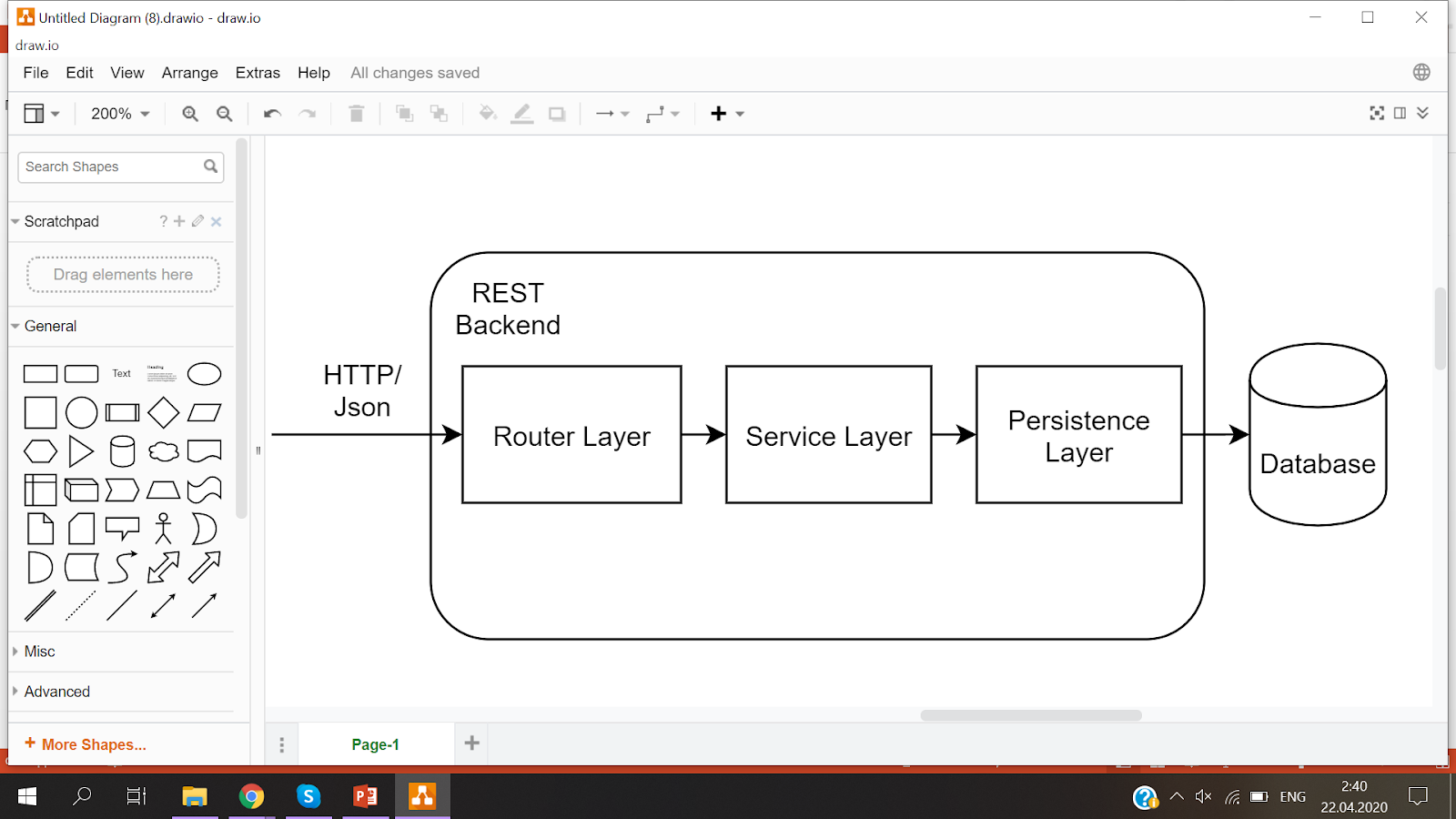


Рисунок 10 – Схема архитектуры backend приложения

Backend приложение представлено REST API, которое реализует многослойную архитектуру и содержит три уровня:

* уровень маршрутизации (Router Layer) определяет, какие точки входа сервисов соответствуют конкретным HTTP URL, и как будут считываться параметры из HTTP-запроса;
* уровень сервисов (Service Layer) содержит всю бизнес-логику приложения;
* уровень сохраняемости (Persistence Layer), который отображает базу данных на объекты предметной области, хранящиеся в памяти и наоборот. Для реализации уровня сохраняемости использовался Hibernate.

Для защиты интерфейса REST API и определения процессов авторизации и аутентификации на стороне backend использовалась конфигурация Spring Security.

# 4.2. Структура приложения

На рисунке 11 изображена диаграмма пакетов приложения, отражающая структуру программы.



Рисунок 11 – Диаграмма пакетов

На представленной диаграмме пакетов видны зависимости модулей в приложении. На ней четко видны три уровня, соответствующий шаблону MVC это пакеты: controller, service и repository. На уровнях service и repository используются классы из пакета model, который содержит классы для описания сущностей предметной области. Классы из пакета dto используемые для описания структуры запросов и ответов принимаемых и отправляемых rest бекэндом приложения используются на уровнях controller и service. Так же на диаграмме присутствует пакет пакет security в котором собраны настройки Spring Security и определен процесс аутентификации и авторизации пользователя.

**4.3. Диаграмма классов**

Рассмотрим более подробно структуру классов на уровне представления в backend приложении.

На рисунке 12 представлена диаграмма классов для уровня представления приложения. На данной диаграмме отражены классы, находящиеся в пакете entity и использующиеся для отображения данных из базы данных на объекты этих классов. Рассмотрим более подробно классы, изображенные на ней.

Класс User инкапсулирует информации о пользователе системы, содержит поля для хранения идентификатора, имени, фамилии, логина, пароля. Для хранения роли пользователя, обеспечивающей пользователя набором полномочий в системе, у класса User есть поле role типа Role, представленный типом enum и имеющий соответствующие значения: STUDENT, TEACHER, ADMIN. Также класс User имеет список courses, в котором содержатся все курсы созданные пользователем, список exhaustedDbs, содержащий загруженные пользователем записи о базах данных. (попытки + ответы)

Класс Courses используется для описании курса и имеет поля для хранения идентификатора, названия и описания курса. Для возможности определения автора курса присутствует поле author типа User. Также курс имеет поле tests для хранения списка относящихся к данному курсу тестов.

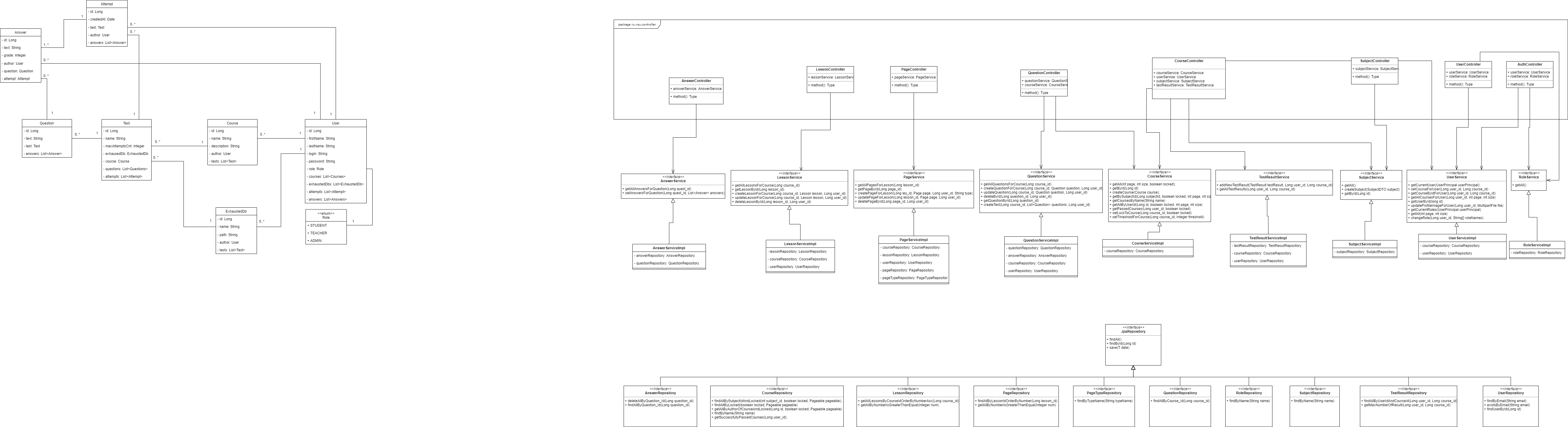
Расположение баз данных, загруженных в систему описывается в классе ExhaustedDb. Класс содержит поля для идентификатора, названия базы данных и поле для хранения относительного пути файла с базой данных. Для хранения информации об авторе базы в классе присутствует поле author, для хранения информации о тестах, в которых используется база данных поле tests.

Класс Test используется для описания тестов, содержит поля уникальный идентификатор, название теста, максимальное допустимое число попыток решений данного теста. С тестом связанна база данных которая записывается в поле exhaustedDb – это база данных запросы к которой студенты должны давать в качестве ответов на вопросы теста. Вопросы теста хранятся в поле questions. Для хранения всех попыток прохождения данного теста, предусмотрено поле attempts.

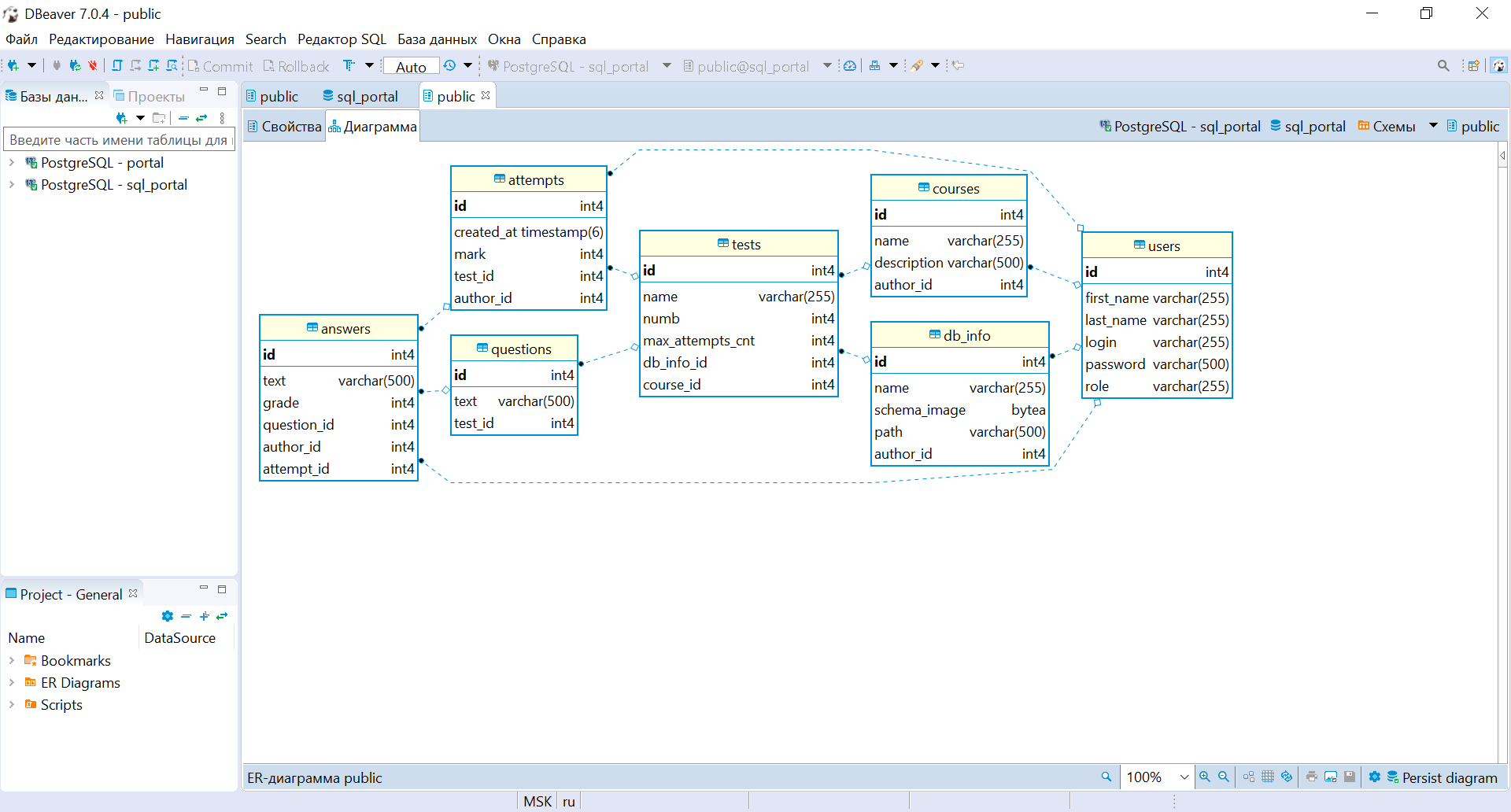
Класс Question описывает вопрос и содержит поля: уникальный идентификатор, поле text для записи текста вопроса, поле test указывающее на тест к которому относится вопрос и список ответов, данных на вопрос.

Класс Answer присутствует для описания ответов на вопросы теста. Класс содержит поля: идентификатор, поле text для текста ответа, grade для оценки данного ответа, поле author для автора ответа, поле question для указания вопроса на который давался ответ и поле attempt, указывающее к какой попытке решения теста относится данный ответ.

Класс Attempt используется для описания попыток решения тестов. Каждый экземпляр этого класса будет иметь поле идентификатора попытки, время сохранения попытки, поле test хранящее тест попытка решения которого была зафиксирована, поле для сохранения информации об авторе попытки и список ответов, относящихся к данной попытке.



Базы данных, которые преподаватели загружают в систему хранятся в каталоге проекта. ПОМЕНЯТЬ КАРТИНКУ СХЕМЫ БД



На рисунке представлена схема базы данных используемая в приложении.

Для сохранения информации о пользователях использовалась таблица users, содержащая необходимые для пользователя поля. Таблица users связана с таблицами courses и exhausted\_dbs отношениями один-ко-многим, по полю author\_id обоих таблиц. Таблица courses содержит основную информацию о курсе, а exhausted\_dbs хранит информацию о базах данных, которые преподаватели загружают в систему. Файлы с базами данных хранятся в каталоге проекта, а путь к файлам записываются в поле path таблицы exhausted\_dbs для соотвествующих записей. Таблица tests предназначена для хранения информации о тестах и определяет что у каждый тест может относиться к одному курсу и одной базе данных – это обусловлено отношениями многие-к-одному. Так же таблица tests связана с таблицами attempts и questions отношением один-ко-многим, что означает что у теста может быть несколько попыток его решения, и он может содержать несколько вопросов. Таблица attempts используется для сохранении истории взаимодействия пользователей с тестами – все данные ответы при решении теста объединяются в одну попытку.

# 5. ИНТЕРФЕЙС

Добавить картинки с примерами выполнения различных запросов, различной сложности. + запросы с ошибками

Рассмотрим основные страницы сайта. Для начала опишем страницы доступные для посещения неавторизованному пользователю.

На рисунке 12 можно увидеть интерфейс страницы входа. Страница входа – это единственная страница, доступная для посещения неавторизованным пользователям. Для доступа к другим страницам системы пользователю необходимо произвести вход в систему. Он должен ввести логин, пароль и нажать кнопку «Войти» для авторизации в системе. При отсутствии в системе пользователя с введенной парой логина и пароля на странице отобразится сообщение об ошибке: «Во время входа возникла ошибка. Проверьте свои учетные данные и повторите вход еще раз». Также в случае если пользователь оставит хотя бы одно поле незаполненным кнопка «Войти» не будет активна.

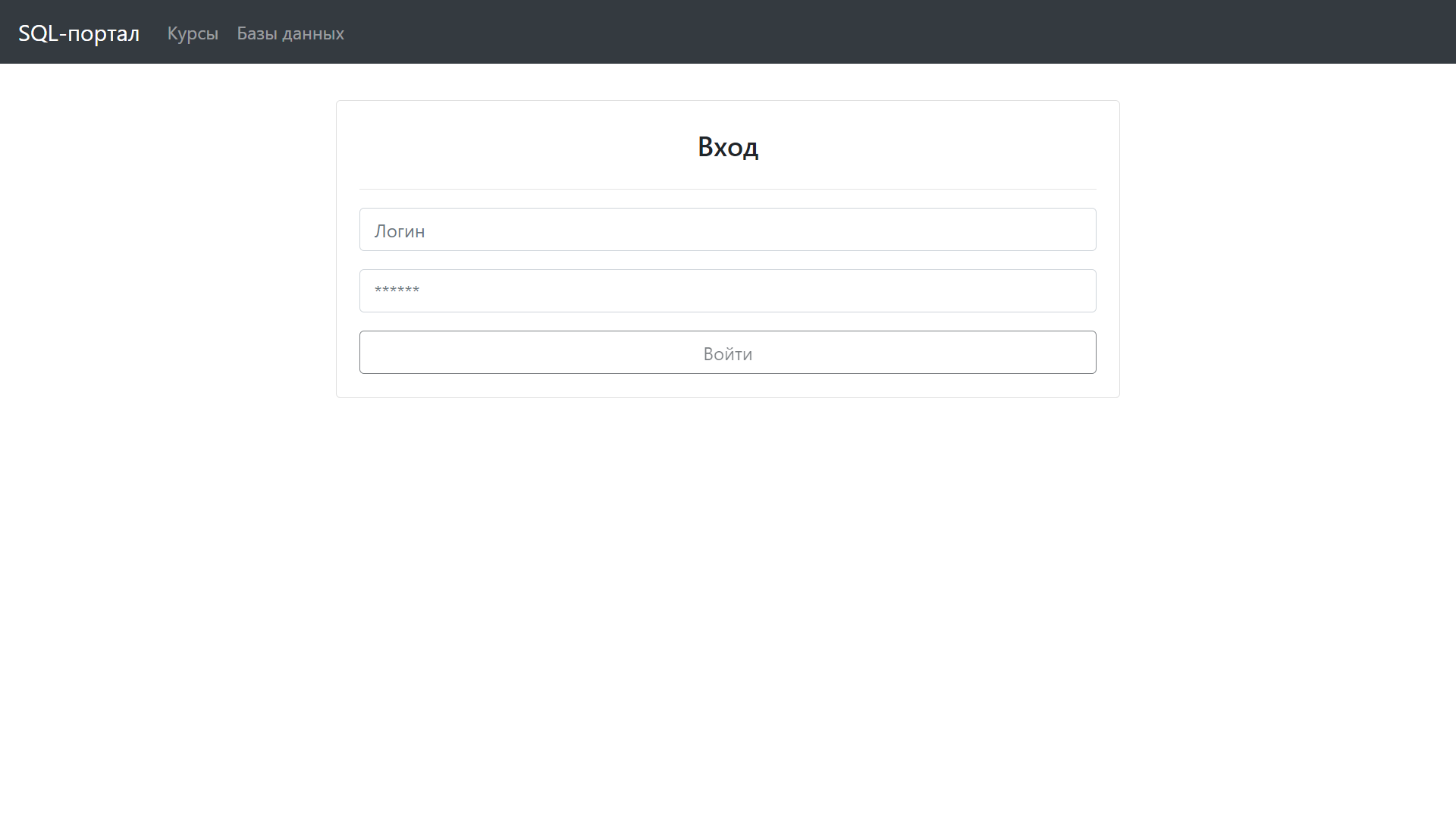


Рисунок 12 – Страница входа

После успешного входа пользователи (студенты и преподаватели) перенаправляются на страницу курсов, которую можно увидеть на рисунке 13. Здесь перечислены все созданные в системе курсы по SQL доступные для тренировок студентам. На странице предусмотрена пагинация для подгрузки курсов по необходимости. Для каждого курса указана минимальная информация о нем: его название, описание и автор курса. Преподаватели могут редактировать или удалять созданные ими курсы. Такие курсы будут иметь дополнительные кнопки-иконки для перехода к редактированию или удалению курса.

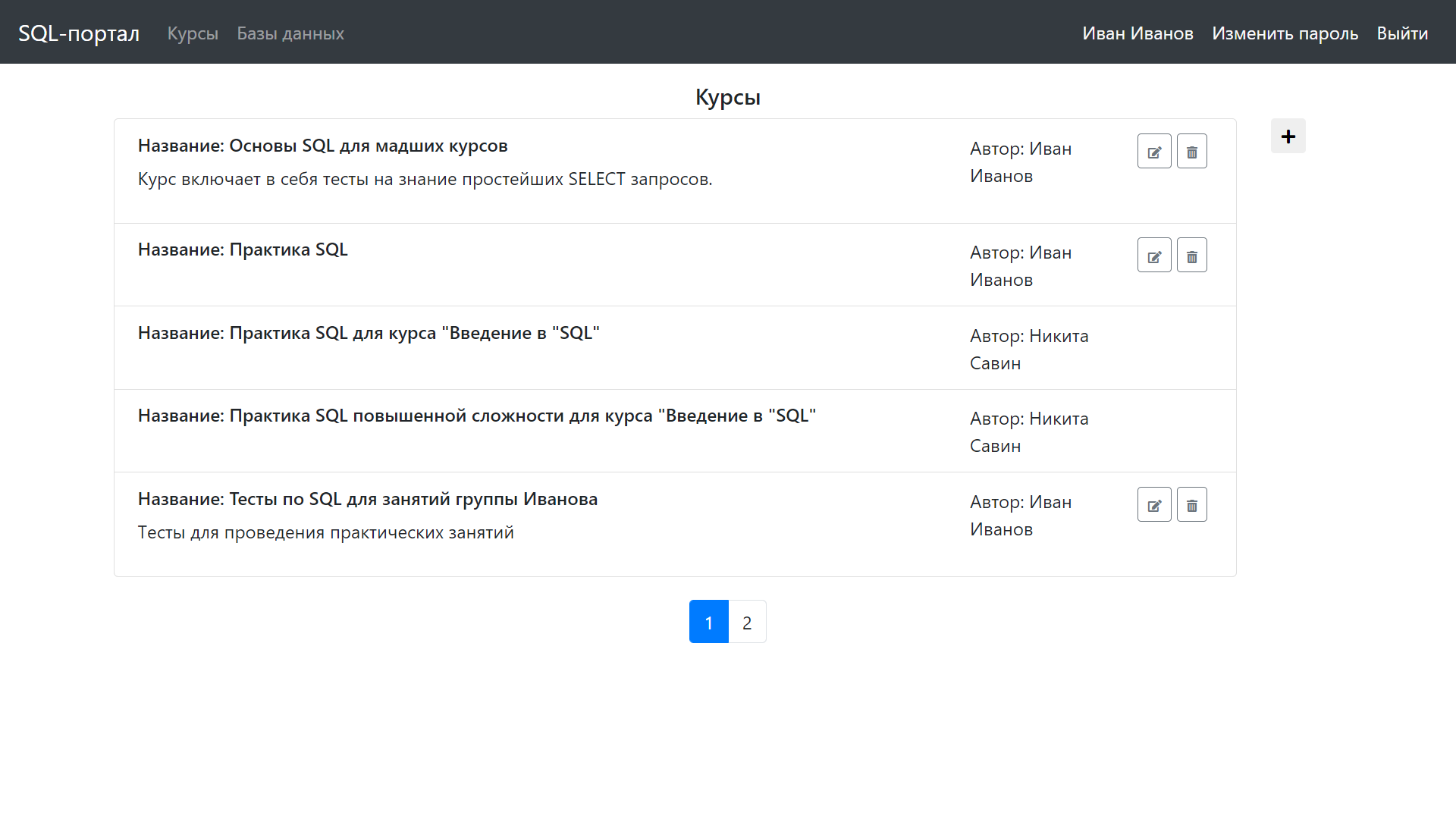


Рисунок 13 – Страница курсов

Для создания курса преподавателям доступна кнопка «+» - создать курс. Нажав на эту кнопку преподаватель перенаправляется на страницу создания курса, которую можно увидеть на рисунке 14. Эта страница доступна только для преподавателей. Пользователю предлагается ввести название создаваемого курса и по желанию добавить к нему короткое описание. Кнопка «Создать» будет активна только при условии, что поле Название курса не пустое.

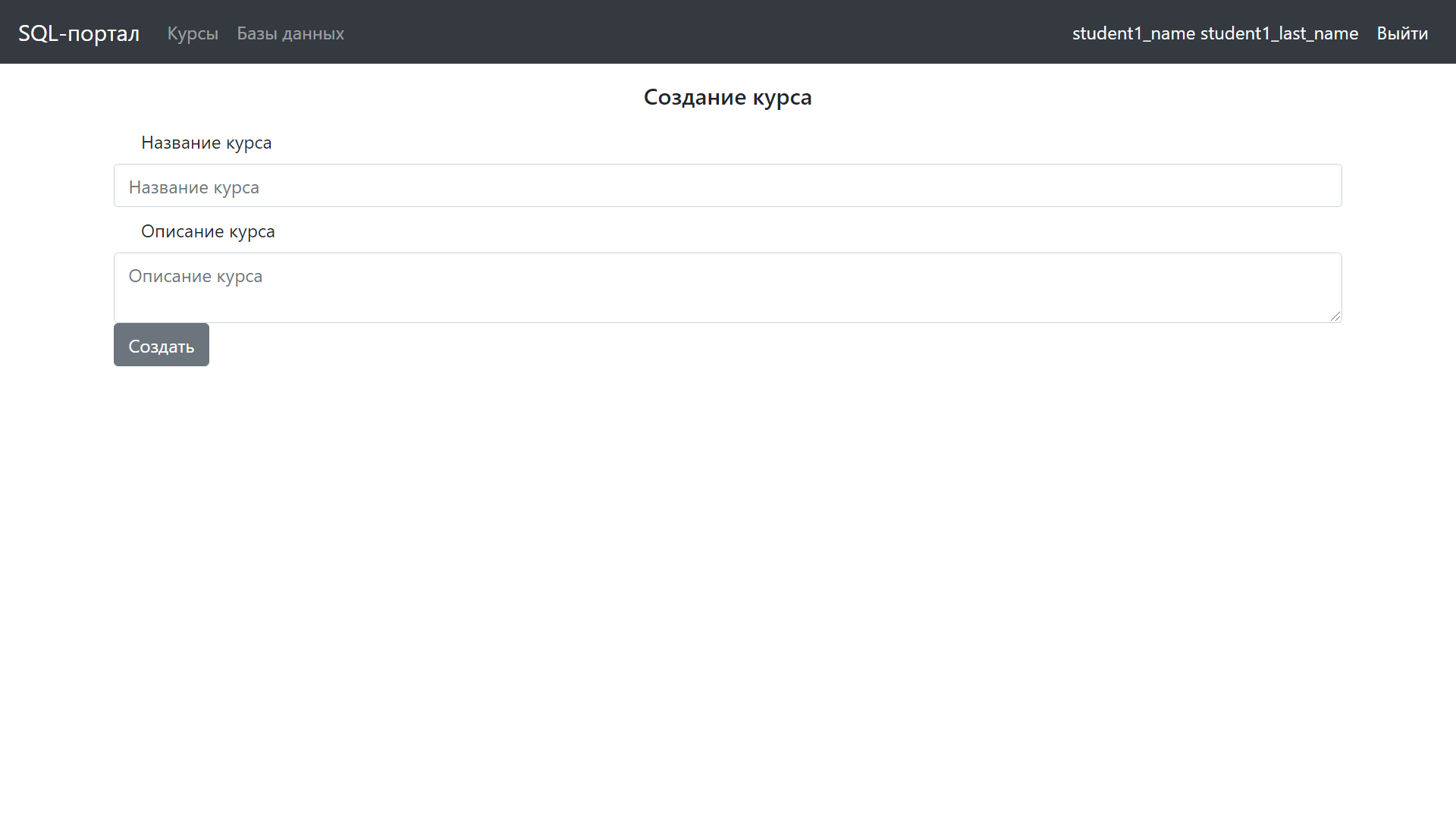


Рисунок 14 – Страница создания курса

Для создания полноценного курса, его автор может добавлять к курсу тесты. Для каждого теста может быть выбрана любая загруженная в систему база данных. В рамках одного курса для тестов базы могут отличаться. На рисунке 15 изображена страница создания теста курса доступная преподавателям. Для создания теста необходимо заполнить поля: название теста, максимальное число попыток, ограничивающее студентов в количестве попыток решить тест, и выбрать загруженную заранее в систему базу данных из списка для выполнения теста.

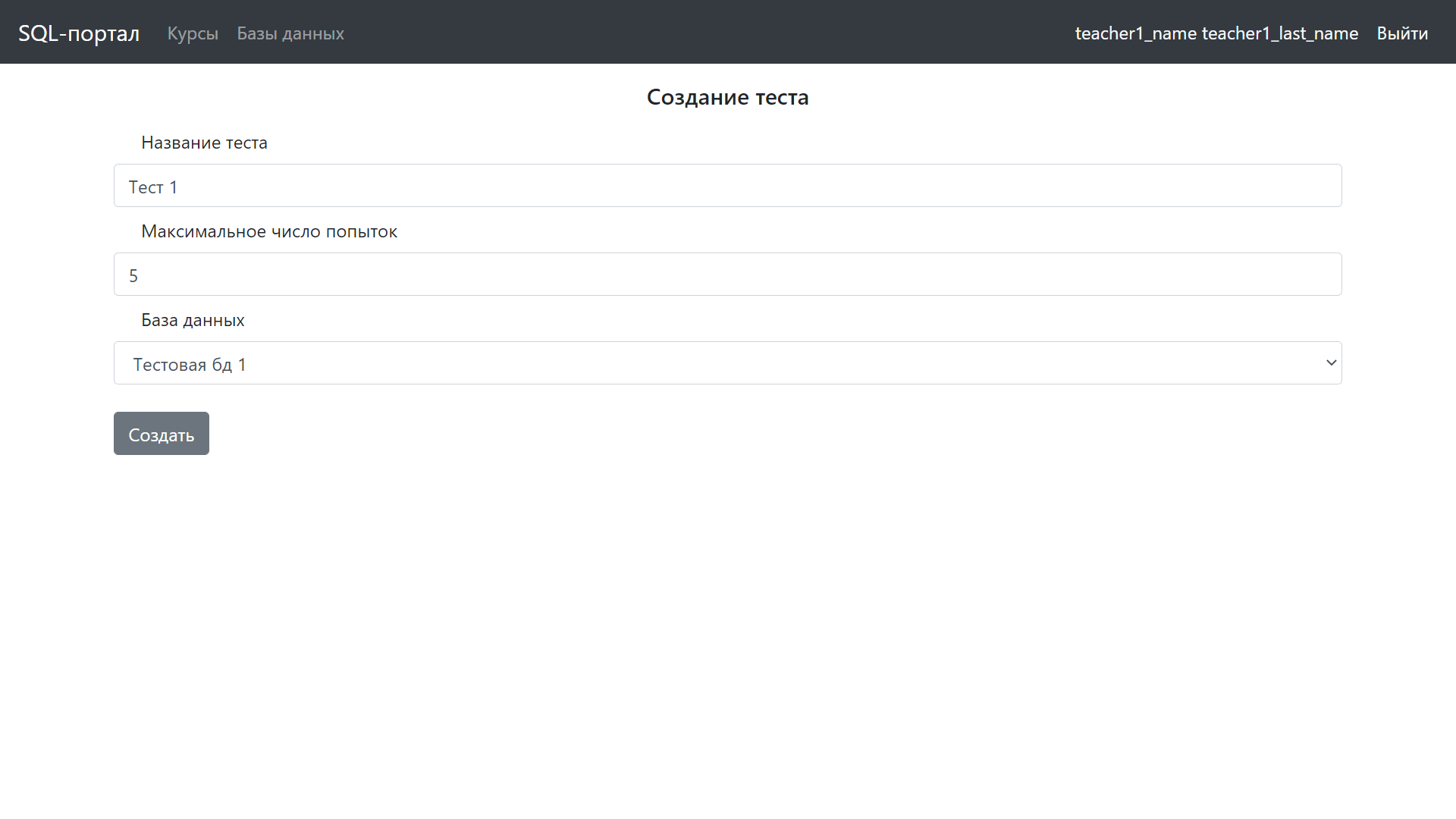


Рисунок 15 – Страница создания теста

На рисунке 16 представлена страница загрузки базы данных доступная для преподавателей. Для загрузки базы данных необходимо название для базы данных, выбрать файл с базой данных с описанной ранее структурой в формате .xlsx и нажать на кнопку «Загрузить».

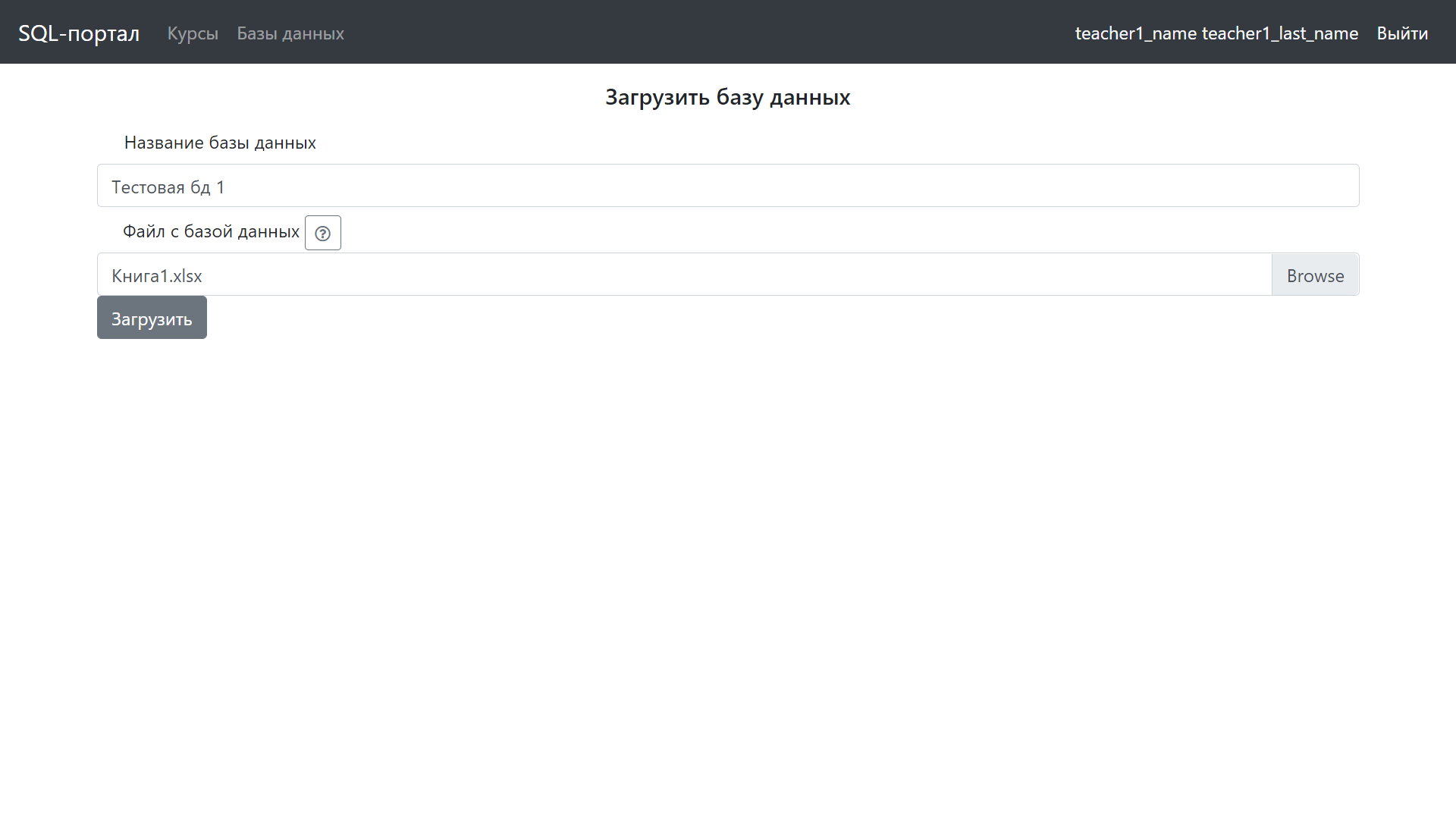


Рисунок 16 – Страница создания базы данных

После успешной загрузки базы данных она становится доступна для пользования студентам и преподавателям. На рисунке 17 показана страница выполнения запросов к базе данных. Пользователю необходимо ввести запрос в редактор кода и нажать на кнопку «Выполнить». Редактор кода будет подсвечивать синтаксис SQL-запросов. После выполнения запроса отобразится результат выполнения вашего запроса, будь то ошибка или выбранные записи из таблицы.

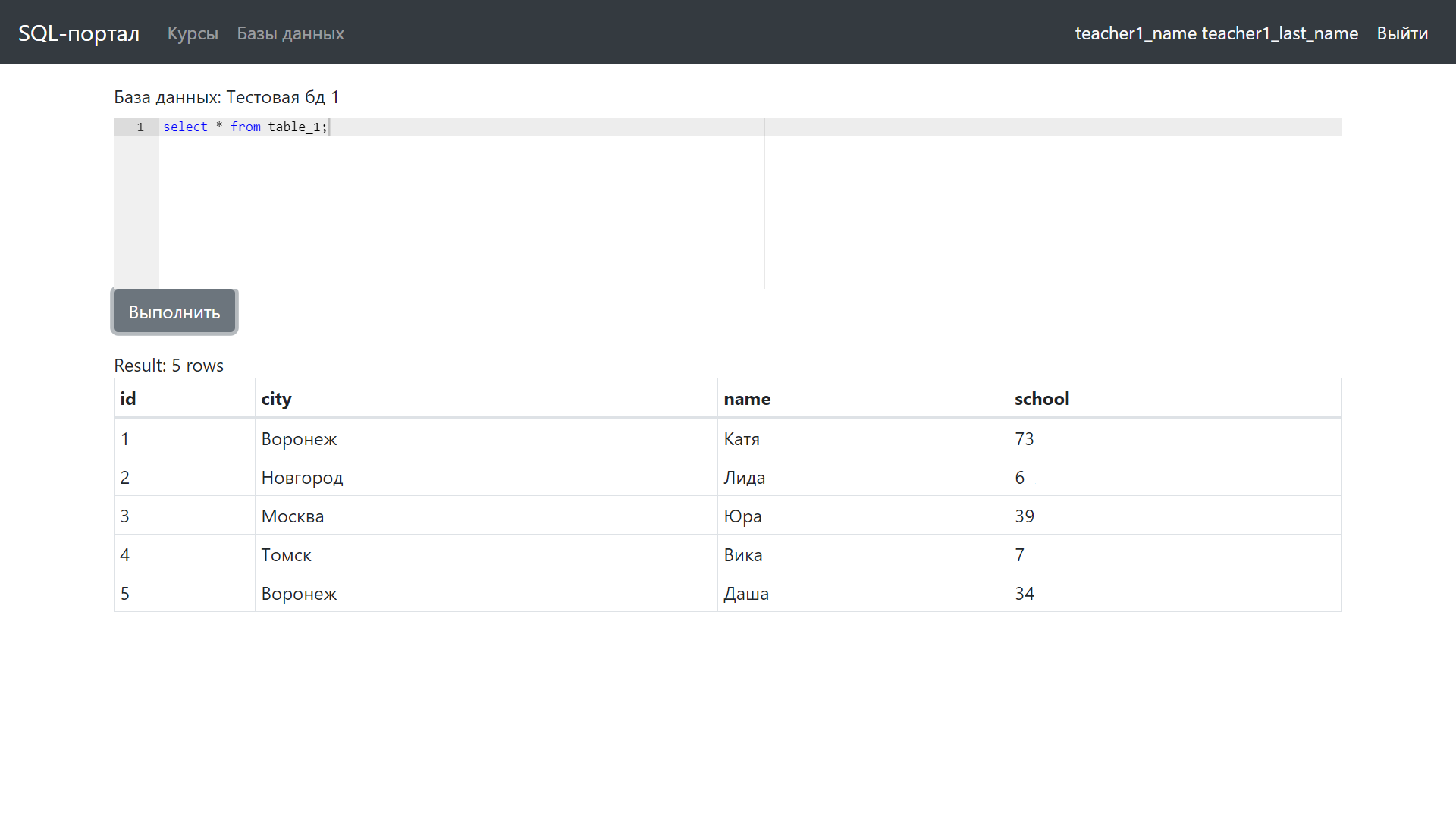


Рисунок 17 – Страница выполнения запросов к базе данных

На рисунке 18 показана страница курса, где можно увидеть название курса и список тестов относящихся к нему. Студенты могут выбирать нужные им тесты и переходить к их решению, а преподаватели могут редактировать тесты своих курсов или добавлять новые тесты нажав на иконку «+».

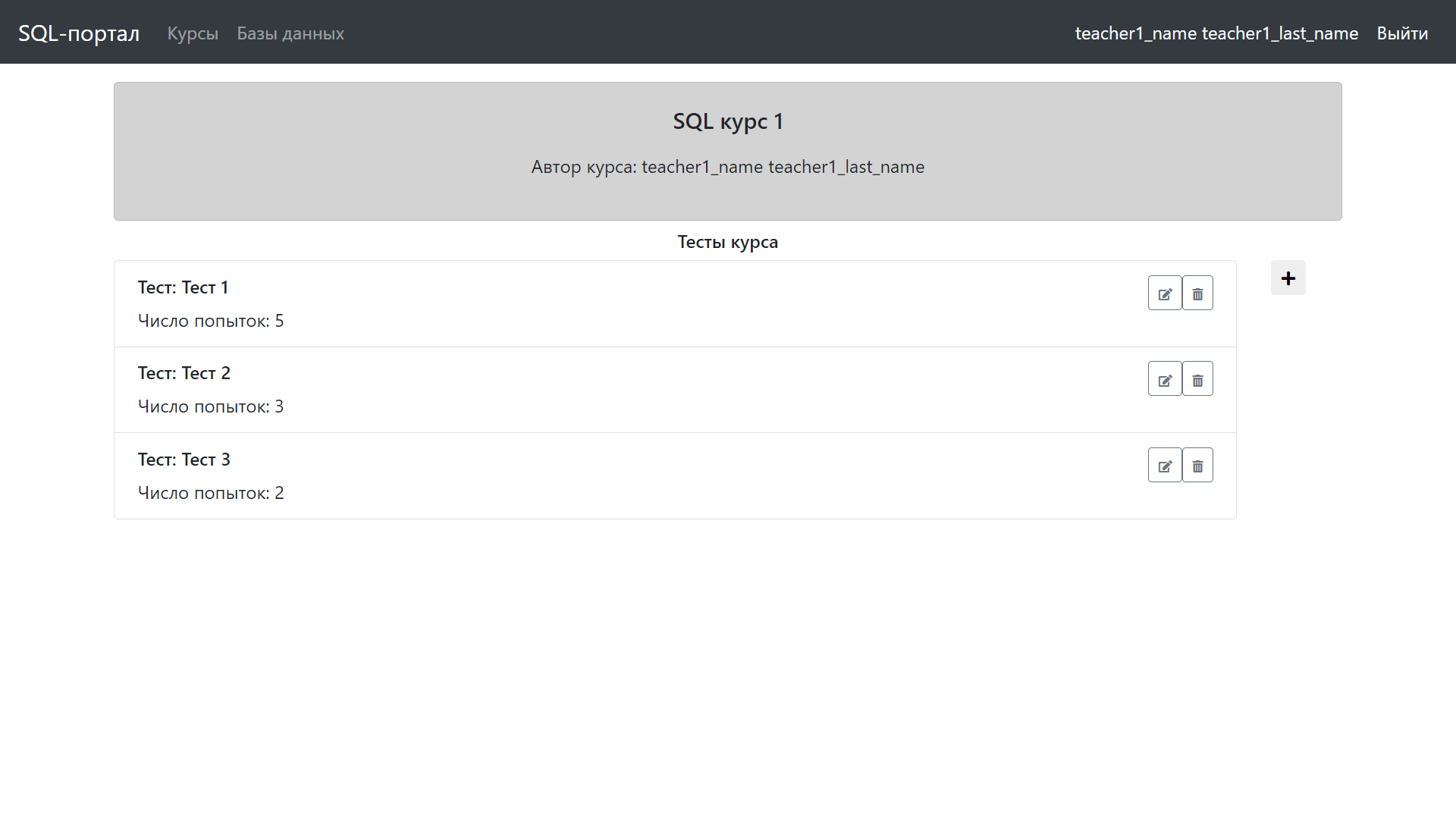


Рисунок 18 – Страница курса

На рисунке 19 изображена страница создания теста, доступная для преподавателей. Здесь преподаватель может создать необходимое ему количество вопросов с помощью кнопок «Добавить вопрос» и кнопок удаления для вопросов. В качестве вопроса преподаватель должен вписать задание, требующее составление запроса к базе данных, определенной в этом тесте. Под вопросом преподаватель должен дать ответ на свой вопрос, чтобы в последующем сравнивать ответы студентов со своими при проверке. Для сохранения теста, преподавателю необходимо нажать на кнопку «Сохранить тест».

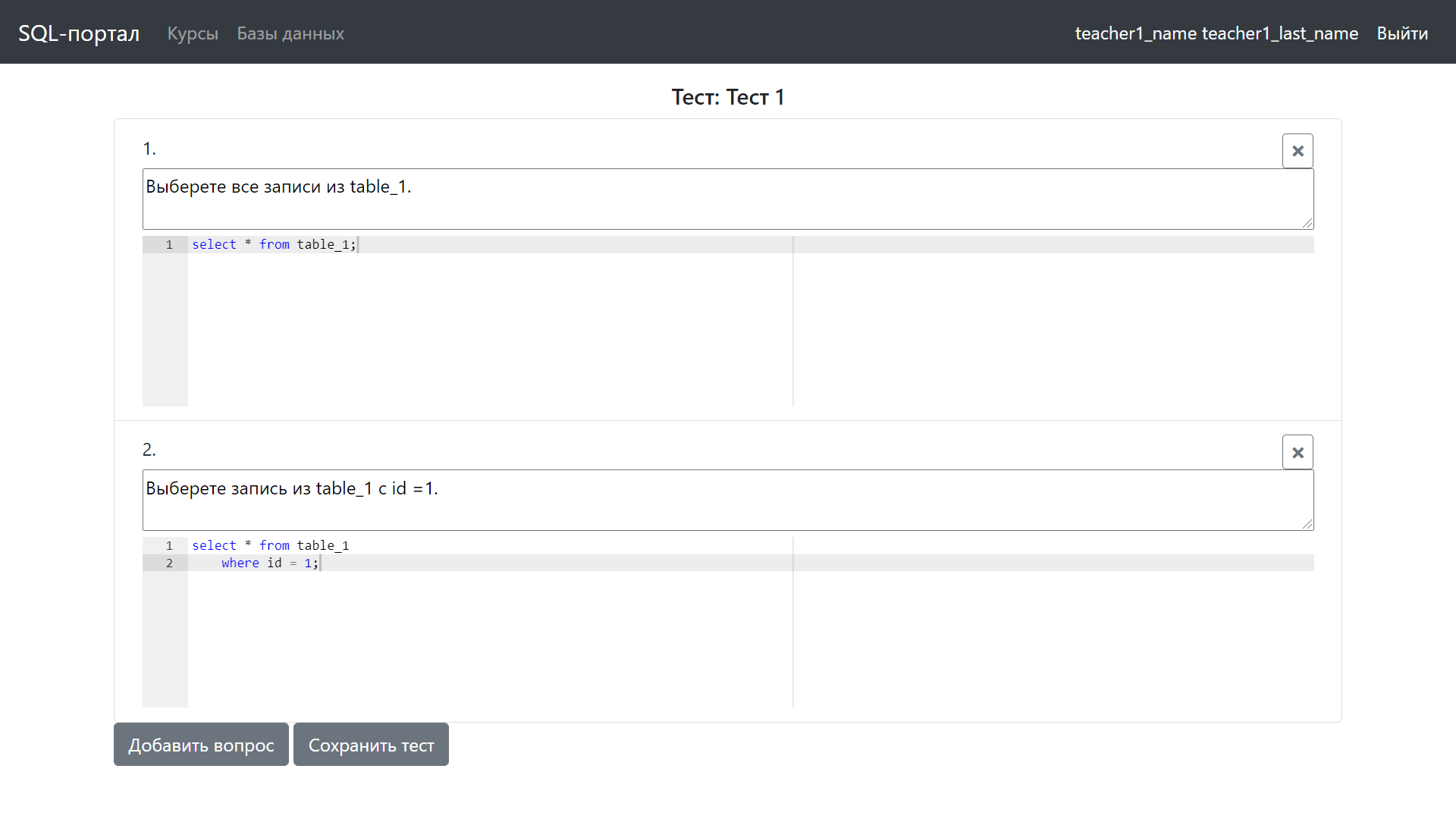


Рисунок 19 – Страница создания теста

Для проверки решения тестов студентами для каждого теста своего курса может нажать на кнопку иконку галочку для перехода к проверки решений студентами данного теста. По нажатию преподаватель будет перенаправлен на страницу представленную на рисунке 191. Данная страница поделена на две части – левую и правую. В левой части страницы преподаватель может увидеть список всех студентов зарегистрированных на сайте. Для просмотра списка студентов предусмотрена пагинация. При нахождении преподавателем нужного студента он должен нажать на него, после чего в правой части страницы появится список всех попыток решения выбранным студентом текущего теста. Для идентификации каждой попытки указывается время ее отправки студентом. Попытки выводятся отсортированные по этому времени. Для перехода к просмотру интересующей попытки преподавателю также необходимо нажать на отображающей ее элемент.

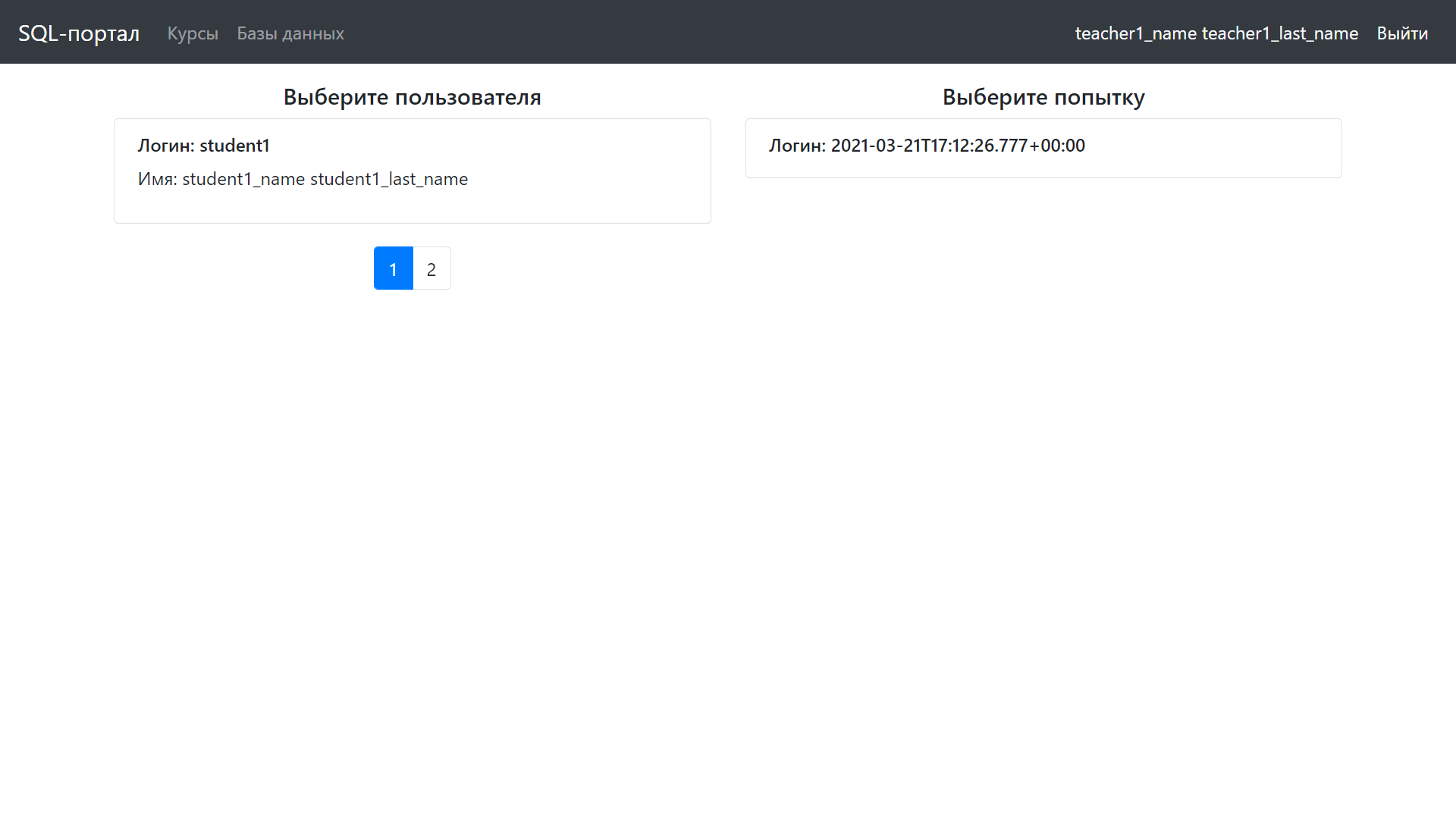


Рисунок 191 – Страница попыток решений теста студентами

Дописать здесь где выставление балла

На рисунке 192 можно увидеть страницу проверки попытки студента. В заголовке страницы указывается текущий тест, попытка решения которого проверяется. Для преподавателя представляются вопросы теста, для каждого из которых представлен ответ который дал на него студент и правильный ответ, который дал на него сам преподаватель при создании данного теста. Так как язык запросов очень разнообразен и чаще всего одну и ту же выборку данных можно сделать различными способами, то проверка требует знаний преподавателя, а правильный ответ лишь выполняет роль помощника при проверке данного теста. После проверки ответа преподаватель должен оценить его поставив для него оценку в поле «Оценка» какая должна быть оценка???? От 0-100 от 0-5??? После оценки преподавателем всей попытки общая оценка попытки будет выставлена как среднее арифметическое всех оценок за ответы на вопросы.

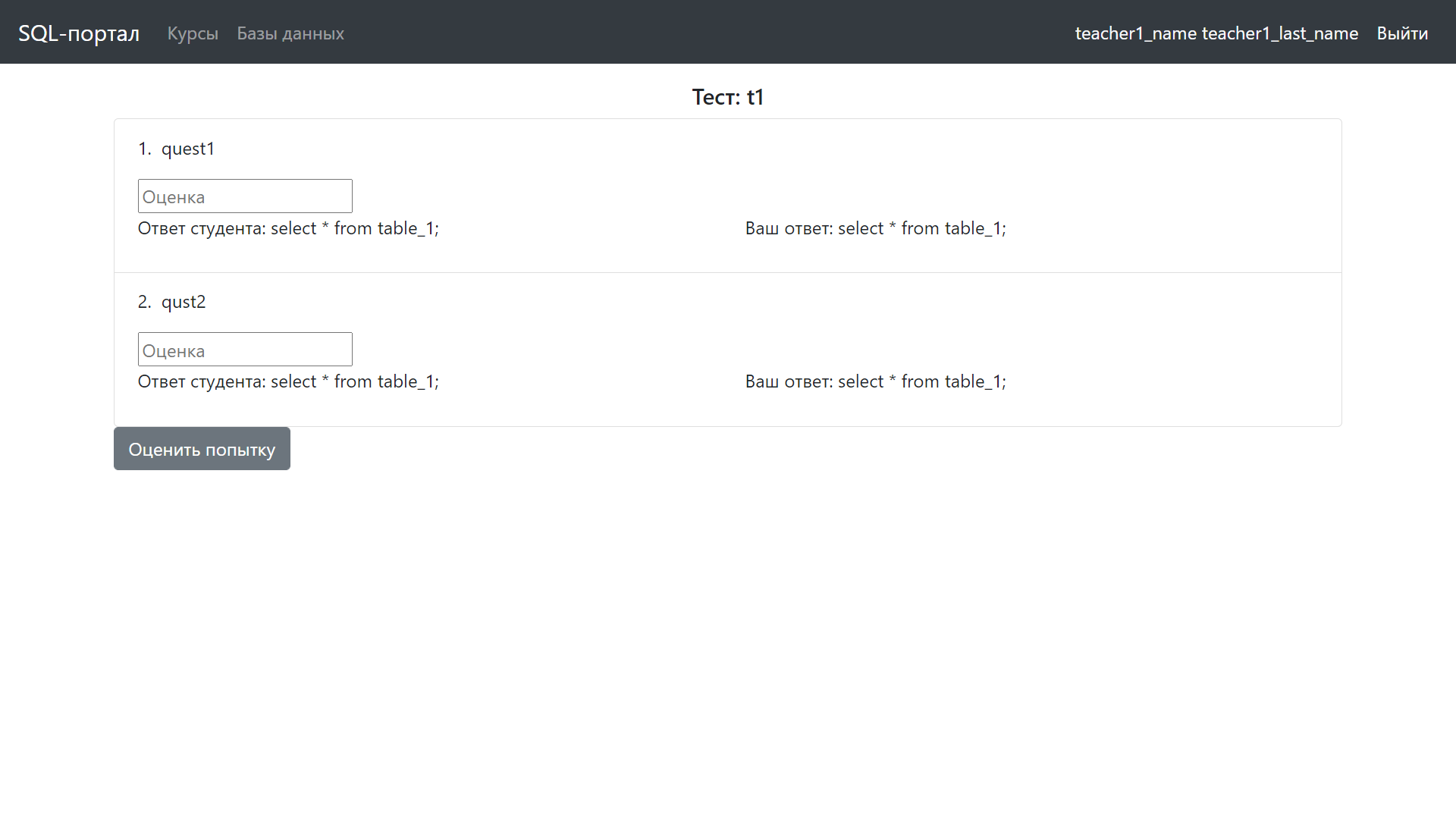


Рисунок 192 – Страница оценки попытки студента

На 20 рисунке можно увидеть страницу решения теста, доступную студенту.

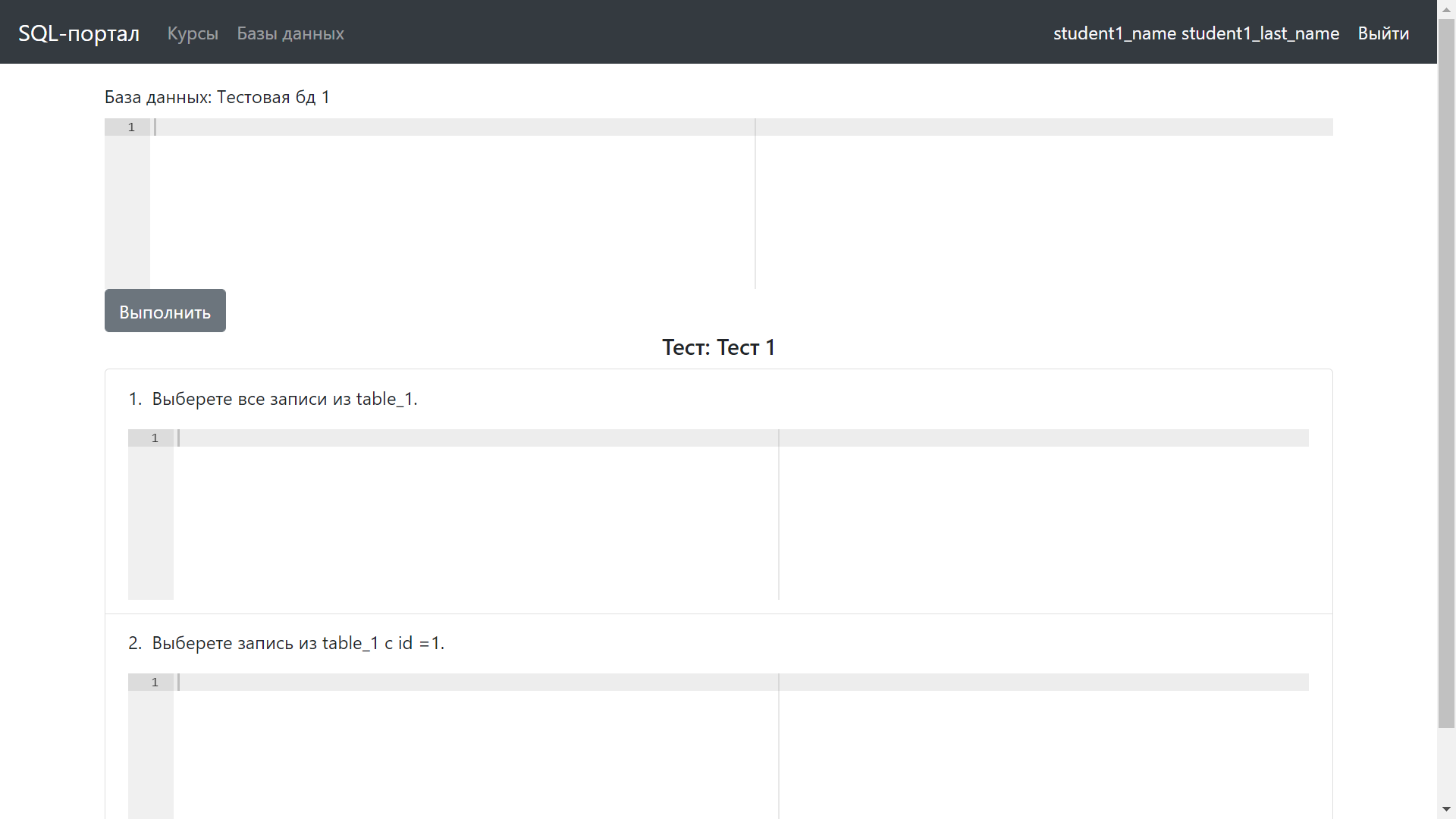


Рисунок 20 – Страница решения теста

(картинки с админ панели!!)

На рисунке 21 показана главная страница администратора, на которую он перенаправляется после успешного входа. На данной странице администратор имеет две вкладки: «Студенты» и «Преподаватели» со всеми зарегистрированными студентами и преподавателями соответственно. На каждой вкладке для удобной навигации предусмотрена пагинация для загрузки и отображения пользователей по необходимости. Список выводимых пользователей отсортирован по их фамилии.

Добавление пользователей в систему – обязанность администратора. Для добавления пользователя на главной странице администратор имеет кнопку «+», нажатие на которую перенаправит его на страницу создания нового пользователя в системе. Для редактирования существующих данных пользователей и их удаления из системы рядом с пользователями присутствуют соответствующие кнопки.

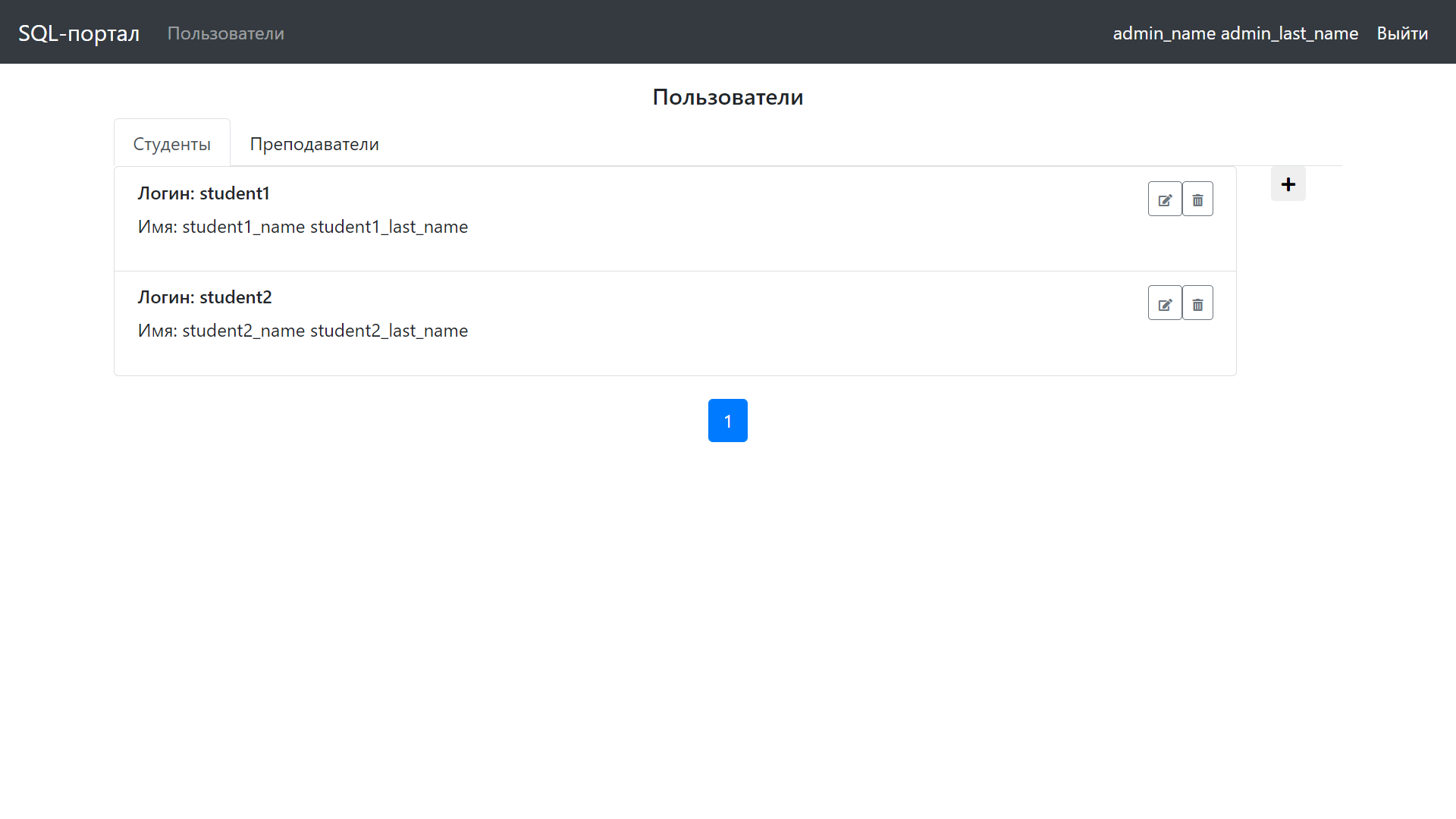


Рисунок 21 – Главная страница администратора

На рисунке 22 представлена страница добавления нового пользователя доступная для посещения только администраторам системы. На данной странице администратору необходимо заполнить следующие поля: Логин, Имя, Фамилия, Роль в системе, Пароль. Все поля являются обязательными для заполнения. При заполнении поля Роль в системе будет предложено два варианта: «Студент», «Преподаватель».

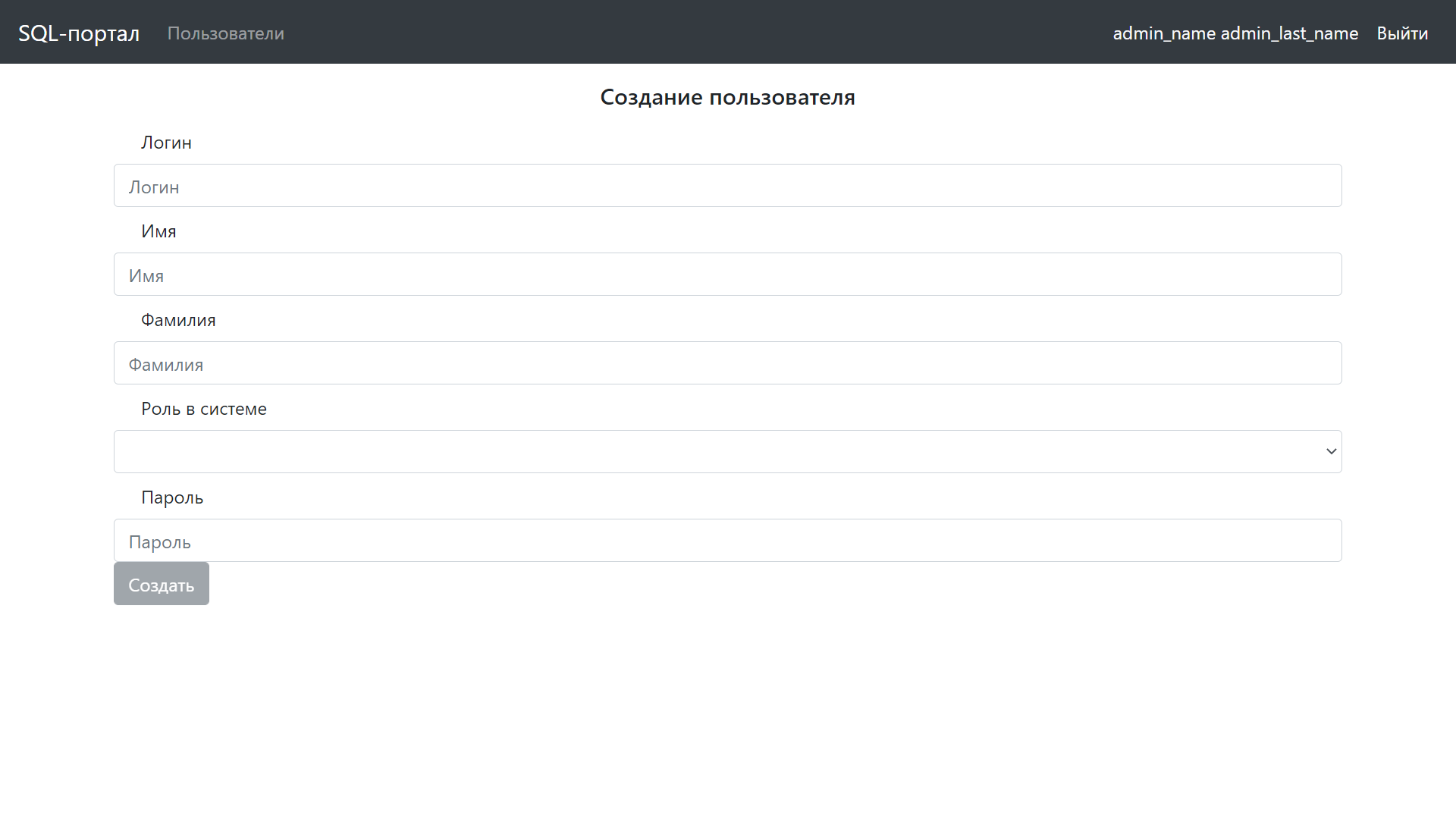


Рисунок 22 – Станица создания нового пользователя

При редактировании пользователя администратору можно изменять любые поля, приведенные на рисунке 23, при этом логин пользователя должен оставаться уникальным в разрезе всей системы.

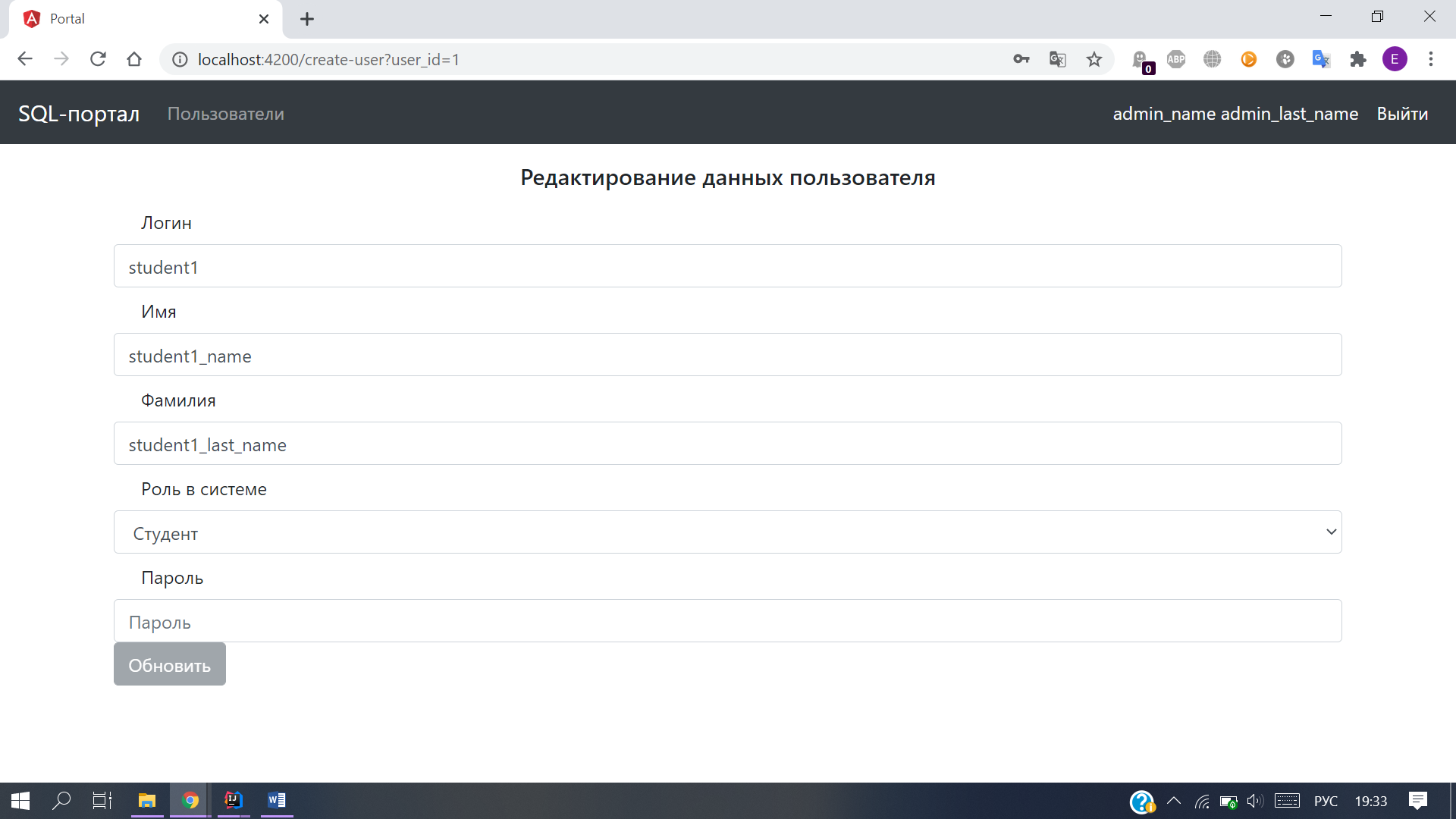


Рисунок 23 – Станица редактирования существующего пользователя

При удалении пользователя администратору будет необходимо подтвердить удаление во всплывающем окне. При удалении пользователя из системы вместе с ним будут удалены его курсы, базы данных, если пользователь преподаватель, и его попытки решения тестов, если пользователь студент.

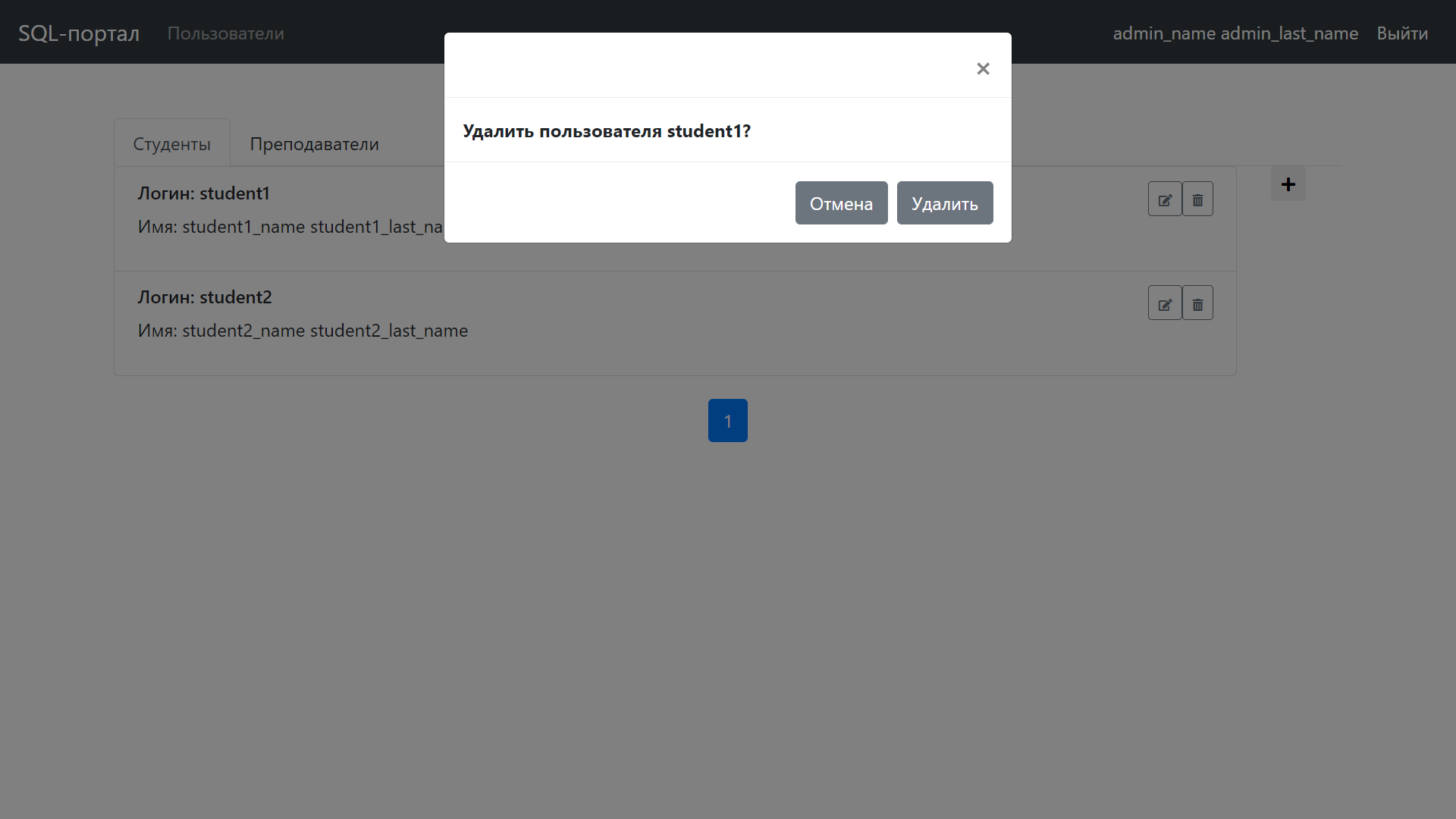


Рисунок 24 – Удаление пользователя из системы

**ЗАКЛЮЧЕНИЕ**

В ходе работы было разработано приложение, являющееся онлайн порталом для обучения.

Разработанное приложение предоставляет всю заявленную функциональность, для создания и управления курсами. Предусмотрены операции добавления, удаления и редактирования для элементов курса.

Кроме того, была разработана функциональность, для прохождения и изучения курсов, а также сохранение попыток сдачи курсов.

Для входа в систему через сторонних провайдеров авторизации была проведена интеграция приложения со сторонними сервисами.

Приложение адаптивно под различные устройства, такие как компьютер или мобильное устройство, за счет адаптивной разметки сайта.

Данный образовательный портал имеет преимущества над анализируемыми в ходе курсовой работы приложениями за счет отсутствия на сайте ограничений для создания курсов. Также преимуществом является свободная регистрация для любого желающего, возможность входа через сторонних провайдеров и то, что все курсы, представленные на сайте являются бесплатными, а значит свободными для изучения любыми пользователями системы.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Официальный сайт документации Spring Boot [сайт]. - URL: <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#boot-documentation> (дата обращения 24.05.2019)
2. Официальный сайт документации Hiberante [сайт]. - URL: <https://hibernate.org/orm/documentation/5.4/> (дата обращения 24.05.2019)
3. Официальный сайт документации OAuth2 [сайт]. - URL: <https://oauth.net/2/> (дата обращения 24.05.2019)
4. Официальный сайт руководства Google OAuth 2.0 [сайт]. - URL: <https://developers.google.com/identity/protocols/oauth2> (дата обращения 24.05.2019)
5. Официальный сайт руководства Github OAuth [сайт]. - URL: <https://developer.github.com/apps/building-oauth-apps/authorizing-oauth-apps/> (дата обращения 24.05.2019)