|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ (ИУ5)

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

***«Информационная система сопровождения профессионально образования»***

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель ВКР **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Нормоконтролер **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2019 г.*

# АННОТАЦИЯ

Объектом квалификационной работы является информационная система сопровождения профессионального образования, которая позволит упростить процесс организацию образовательного процесса и коммуникации между преподавателями и студентами.

Целью данной работы является проектирование и реализация информационной системы, состоящей из серверной и клиентской части. Предусмотреть возможность работы большого количества пользователей и отказоустойчивость системы.

Система представляет собой клиент-серверное приложение, где клиентом выступает браузер пользователя. Система обеспечивает возможность одновременного подключения большого количества пользователей.

В первой части пояснительной записки содержится постановка задачи, описание предметной области, выбор критериев качества системы и анализ аналогов.

В конструкторско-технологической части пояснительной записки приведена архитектура проектируемой системы, приведены результаты анализа и выбора основных средств проектирования, приведены инфологическая, даталогическая модели и разработанный пользовательский интерфейс.

В исследовательской части приведено сравнение наиболее актуальных HTTP-серверов, на основании которого был выбран наиболее оптимальный сервер для использования в системе.

Разработка системы проводилась на основании задания на квалификационную работу бакалавра, подписанного руководителем квалификационной работы бакалавра и утвержденного заведующим кафедрой ИУ5 МГТУ им. Н. Э. Баумана и документа «Техническое задание», утвержденного руководителем квалификационной работы бакалавра.

# 

# СОДЕРЖАНИЕ

[АННОТАЦИЯ 2](#_Toc10468432)

[СОДЕРЖАНИЕ 3](#_Toc10468433)

[ВВЕДЕНИЕ 5](#_Toc10468434)

[1 Постановка задачи разработки 6](#_Toc10468435)

[1.1 Постановка задачи 6](#_Toc10468436)

[1.2 Описание предметной области 7](#_Toc10468437)

[1.3 Перечень процессов подлежащих автоматизации 8](#_Toc10468438)

[1.4 Функциональные задачи системы 8](#_Toc10468439)

[1.5 Выбор и обоснование критериев качества 9](#_Toc10468440)

[1.6 Анализ аналогов 11](#_Toc10468441)

[2 Конструкторско-технологическая часть 17](#_Toc10468442)

[2.1 Архитектура системы 17](#_Toc10468443)

[2.2 Выбор СУБД 19](#_Toc10468444)

[2.3 Выбор HTTP сервера 21](#_Toc10468445)

[2.4 Выбор средств разработки клиентской части системы 22](#_Toc10468446)

[2.5 Выбор средств разработки серверной части системы 24](#_Toc10468447)

[2.6 Проектирование базы данных 27](#_Toc10468448)

[2.6.1 Инфологическая модель 27](#_Toc10468449)

[2.6.2 Даталогическая модель 36](#_Toc10468450)

[2.7 Выбор программных средств 41](#_Toc10468451)

[2.8 Выбор аппаратных средств 42](#_Toc10468452)

[2.9 Разработка интерфейса взаимодействия с пользователем 43](#_Toc10468453)

[2.9.2 Интерфейс взаимодействия с модератором 45](#_Toc10468454)

[2.9.3 Интерфейс взаимодействия с преподавателем 52](#_Toc10468455)

[2.9.4 Интерфейс взаимодействия со студентом 52](#_Toc10468456)

[3 Исследовательская часть. Выбор HTTP-сервера 57](#_Toc10468457)

[3.1 Метод максимума взвешенной суммы 60](#_Toc10468458)

[3.2 Метод близости к идеалу 61](#_Toc10468459)

[3.3 Метод ранжирования Борда 62](#_Toc10468460)

[3.4 Метод гарантированного результата 64](#_Toc10468461)

[ЗАКЛЮЧЕНИЕ 66](#_Toc10468462)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 67](#_Toc10468463)

[ПРИЛОЖЕНИЕ А 69](#_Toc10468464)

# ВВЕДЕНИЕ

Квалификационная работа на тему «Информационная система сопровождения профессионального образования» посвящена разработке системы, позволяющей организовать и значительно упростить коммуникацию между преподавателями и студентами в рамках образовательного процесса.

По состоянию на 2016 год в России функционирует около 3,3 тыс. образовательных организаций среднего профессионального образования и около 950 образовательных организаций высшего профессионального образования.

В большей части образовательных организациях не существует централизованный системы коммуникации между студентами и преподавателями. Часто общение между ними осуществляется через электронную почту или средства обмена сообщениями.

Необходимость такой системы обусловлена тем, что объём информационного потока, проходящий через человека, увеличивается с каждым годом. Также, с ростом информационного потока увеличивается объёмы образовательных материалов, которые необходимо усваивать студентам.

Информационная система сопровождения профессионального образования «SkillArea» решает задачи создания единого информационного образовательного пространства, упрощения организации образовательного процесса и централизации обмена информацией между его участниками.

# 1 Постановка задачи разработки

## 1.1 Постановка задачи

Целью данной работы является проектирование и реализация информационной системы, состоящей из серверной и клиентской части. Предусмотреть возможность работы большого количества пользователей и отказоустойчивость системы.

В общем смысле под системой понимают совокупность объектов, компонентов или элементов произвольной природы, образующих некоторую целостность (система образования, пищеварительная система, солнечная система и т. п.). Определяющей предпосылкой выделения некоторой совокупности как системы является возникновение у нее новых свойств, которых не имеют составляющие ее элементы. Системы значительно различаются между собой как по составу, так и по целям функционирования.

Добавление к понятию «система» слова «информационная» отражает цель ее создания и функционирования. Назначение информационной системы — своевременное формирование и выдача достоверной информации для принятия решений. Информационные системы обеспечивают сбор, хранение, обработку, поиск, выдачу информации в задачах из любой области. Они помогают анализировать информацию, решать проблемы и создавать новые информационные продукты [1].

Информационная система сопровождения профессионального образования должна предоставлять возможность преподавателям размещать организационную информацию, образовательные материалы и расписание занятий, а также вести журнал оценок.

Таким образом, задачи проектирования могут быть сформулированы следующим образом:

* исследование предметной области;
* разработка инфологической и даталогической моделей данных;
* разработка структуры системы;
* разработка алгоритмов основных функций системы;
* разработка интерфейса системы;
* программная реализация;
* тестирование системы;
* разработка документации к системе.

## 1.2 Описание предметной области

Предметной областью данной информационной системы является совокупность сущностей, объединенных общим образовательным процессом и связи между ними.

Основными действующими лицами в этом процессе являются студенты, преподаватели и администрация образовательного учреждения. Исходя из этого можно выделить три основные группы пользователей, имеющих различные потребности и функциональные возможности:

1. Студент – пользователь системы, который может просматривать материалы дисциплин, расписание занятий и текущую успеваемость
2. Преподаватель – пользователь системы, который может назначать занятия, размещать материалы, связанные с занятием и ставить оценки студентам за активность на занятиях или выпаленные задания.
3. Модератор – пользователь системы, имеющий доступ ко всей информации, размещённой в ней. Кроме того, он способен редактировать данные групп и отдельных студентов

## 1.3 Перечень процессов подлежащих автоматизации

При создании информационной системы автоматизации подлежат следующие процессы:

* добавление/редактирование информации о студентах;
* добавление/редактирование информации о группах студентов;
* добавление/редактирование информации о преподаваемых дисциплинах;
* добавление/редактирование информации о проводимых занятиях;
* добавление/редактирование общих новостей;
* добавление/редактирование баллов за занятия;
* добавление/редактирование образовательных материалов;
* формирование расписания;
* формирование статистики о успеваемости.

## 1.4 Функциональные задачи системы

В информационной системе все пользователи должны иметь возможность и право:

* регистрироваться в системе;
* вносить и редактировать данные о себе;
* просматривать общую организационную информацию.

Помимо общих возможностей пользователь со статусом «Студент» должен иметь возможность и право:

* просматривать информацию и учебные материалы, связанные с занятиями по тем дисциплинам, которым он обучается;
* просматривать текущее расписание занятий;
* просматривать свою текущую успеваемость.

Помимо общих возможностей пользователь со статусом «Преподаватель» должен иметь возможность и право:

* назначать занятия и оставлять сопровождающие материалы;
* размещать общую организационную информацию;
* ставить и редактировать оценки студентов, полученные ими на занятии.

Помимо общих возможностей пользователь со статусом «Модератор» должен иметь возможность и право:

* добавлять, редактировать и удалять группы студентов;
* редактировать данные студентов;
* добавлять и удалять дисциплины;
* размещать, редактировать и удалять общую организационную информацию;
* назначать и отменять занятия;
* добавлять, редактировать и удалять материалы занятий;
* ставить, редактировать и удалять оценки студентов.

## 1.5 Выбор и обоснование критериев качества

Качество — это совокупность характеристик объекта, имеющая отношение к его способности удовлетворить установленные и предполагаемые требования потребителя. Под объектом качества может пониматься как собственно продукция (товары или услуги, информационная технология), процесс ее производства, так и производитель (организация, организационная структура или даже отдельный работник).

Качество информационной системы — обобщенная положительная характеристика системы, выражающая степень ее полезности пользователю.

Для проектируемого программного комплекса приоритетными являются следующие критерии качества:

* функциональность;
* надежность;
* практичность;
* эффективность;
* сопровождаемость;
* мобильность.

**Функциональность** характеризует способность информационной системы обеспечивать функции, соответствующие предметной области и удовлетворяющие установленным потребностям пользователей в заданных условиях.

**Надежность** характеризует устойчивость системы к большим нагрузкам, а также к обработке заведомо неверной или поддельной информации. Данное требование гарантирует наличие в системе только корректных данных. При возникновении всякого рода ошибок система должна оповещать пользователя о них и принимать возможные попытки по их устранению.

**Практичность** характеризует простоту и понятность работы с системой. Обеспечение удобства интерфейса позволяет минимизировать время, затрачиваемое при работе с системой и получать максимальную эффективность работы.

**Эффективность** включает требования к временным характеристикам. Можно определить как отношение получаемых с помощью информационной системы результатов к затрачиваемым на это ресурсам всех типов.

**Сопровождаемость** характеризует удобство и простоту администрирования, возможность расширения функционала и документированность системы.

**Мобильность** характеризует способность информационной системы к масштабированию или переносу на новое аппаратное обеспечение. Также важной характеристикой системы является её устойчивость к изменению конфигурации или архитектуры аппаратного обеспечения.

## 1.6 Анализ аналогов

Первые системы управления обучением создавались под конкретный проект и, чаще всего, той же организацией (либо её дочерней структурой), в которой затем использовались.

С появлением специализированных компаний-поставщиков системы управления обучением получили развитие универсальные решения, использовавшиеся в различных отраслях. Следующим этапом развития рынка стала специализация системы управления обучением по направлениям использования, с реализацией функционала под соответствующие требования.

Наиболее распространёнными аналогами являются:

* система управления курсами «Moodle»;
* система управления обучением «Canvas».

Критерии сравнения приведены в таблице 1.

Таблица 1 – Критерии сравнения

|  |  |  |
| --- | --- | --- |
| Критерий | Обозначение | Описание |
| Функциональность | K1 | Способность удовлетворить установленные потребностям пользователей в заданных условиях |
| Надежность | К2 | Способность информационной системы поддерживать определенную работоспособность в заданных условиях |
| Практичность | K3 | Свойство информационной системы, характеризующееся сложностью ее понимания, изучения и использования, а также привлекательность для пользователя при применении в указанных условиях |
| Эффективность | K4 | Свойство системы обеспечивать требуемую производительность с учетом количества используемых вычислительных ресурсов в установленных условиях |
| Сопровождаемость | K5 | Приспособленность информационной системы к модификации и изменению конфигурации, а также наличие документации |
| Мобильность | К6 | Приспособленность информационной системы к внедрению системы в новой аппаратно-операционной среде. |

Варианты систем приведены в таблице 2.

Таблица 2 – Варианты сравнения

|  |  |  |
| --- | --- | --- |
| Вариант | Обозначение | Описание |
| Система управления курсами «Moodle» | В1 | Свободное (распространяющееся по лицензии GNU GPL) веб-приложение, предоставляющее возможность создавать сайты для онлайн-обучения, модернизируется и поддерживается с 2001 года. |
| Система управления обучением «Canvas» | В2 | Коммерческое (распространяющееся по лицензии GNU AGPLv3) веб-приложение с открытым исходным кодом, модернизируется и поддерживается с 2010 года. |
| ИС «SkillArea» | В3 | Веб-приложение, предоставляющее единое информационной пространство для организации образовательного процесса и коммуникации между его участниками |

В таблицах 3 и 4 приведён перевод качественных характеристик к количественные. Эффективность, приведённая в таблице, была рассчитана на производительность эквивалентную одному ядру 64х битного процессора архитектуры x86 с частотой 2,0 ГГц и оперативной памяти объёмом 2 Гб и частотой 1,6 ГГц.

Таблица 3 – Перевод качественных характеристик в количественные

|  |  |  |  |
| --- | --- | --- | --- |
| Значение | Функциональность | Надежность | Практичность |
| 1 | Полное соответствие требованиям | Работоспособность при максимально возможных нагрузках | Интуитивно понятный интерфейс |
| 0,75 | Соответствие требованиям выше 90% | Работоспособность при нагрузках значительно выше среднего | Время освоения основного функционала менее 20 минут |
| 0,5 | Соответствие требованиям выше 80% | Работоспособность при средних нагрузках | Время освоения основного функционала менее 1 часа |
| 0,25 | Соответствие требованиям выше 50% | Работоспособность при минимальных нагрузках | Не требуются специальные знания для освоения |
| 0 | Несоответствие требованиям | Отсутствие работоспособности при минимальных нагрузках | Требуются специальные знания для освоения |

Таблица 4 – Перевод качественных характеристик в количественные

|  |  |  |  |
| --- | --- | --- | --- |
| Значение | Эффективность | Сопровождаемость | Мобильность |
| 1 | Способность обработать более 50 запросов в секунду | Использованы актуальные средства разработки, документация на русском или английском языке | Автоматизированное внедрение системы занимает менее 20 минут |
| 0,75 | Способность обработать более 25 запросов в секунду | Использованы не актуальные средства разработки, документация на русском или английском языке | Автоматизированное внедрение системы занимает менее 2х часов |
| 0,5 | Способность обработать более 10 запросов в секунду | Использованы не актуальные средства разработки, присутствует документация | Автоматизированное внедрение системы занимает более 2х часов |
| 0,25 | Способность обработать более 5 запросов в секунду | Использованы устаревшие средства разработки, присутствует документация | Ручное внедрение занимает менее суток |
| 0 | Не способность обработать более 5 запросов в секунду | Прочее | Ручное внедрение занимает более суток |

В таблице 5 приведены результаты сравнения аналогов с учётом весовых коэффициентов

Таблица 5 – Сравнение аналогов с учетом весовых коэффициентов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерий | Коэффициент | В1 | В2 | В3 |
| К1 | 0,2 | 1 | 1 | 1 |
| К2 | 0,15 | 0,75 | 1 | 1 |
| К3 | 0,2 | 0,5 | 0,75 | 1 |
| К4 | 0,2 | 1 | 0,5 | 0,75 |
| К5 | 0,15 | 0,75 | 1 | 1 |
| К6 | 0,1 | 0,25 | 1 | 1 |
| Итог |  | 0,75 | 0,85 | 0,95 |

АИС «SkillArea» набрал наибольший коэффициент, а значит выполняет поставленные задачи лучше остальных, поэтому разработка системы является целесообразной.

# 2 Конструкторско-технологическая часть

## 2.1 Архитектура системы

В качестве архитектуры была выбрана трёхуровневая архитектура клиент-серверная архитектура. Она включает себя клиентскую часть, серверную часть и базу данных.

Клиентом системы выступает браузер пользователя, на котором запускается приложение. Клиентское приложение выполняет роль пользовательского интерфейса системы и осуществляет взаимодействие с серверной частью системы. Общение с серверной частью происходит по протоколу HTTP в формате JSON.

В серверную часть приложения входит HTTP-сервер и серверное приложение.

* HTTP-сервер выполняет следующие функции:
* выступает в роле файл сервера для статических и медиа файлов;
* проектирует приходящие запросы на сервера приложения;
* балансирует нагрузку между серверами приложений;
* в случае ошибок в запросе, возвращает страницу с ошибкой;
* принимает на себя нагрузку по работе с плохими и медленными клиентами.

Серверное приложение отвечает за авторизацию и бизнес-логику работы системы. Оно может располагаться как на одной серверной машине, так и на нескольких.

Хранение данных осуществляется в кластере баз данных. В кластере баз данных обязательно должна присутствовать только одна мастер-реплика, в которую будет осуществляться запись данных. Кроме неё в кластере могут находиться слейв-реплики, с которых будет производиться чтение данных. Количество слейв-реплик ограничено максимальным количеством соединение, которое поддерживается мастер-репликой. Однако стоит учитывать, что каждое соединение замедляет работу сервера мастер-реплики. В случае, когда слейв-реплики отсутствуют чтение данных должно происходить с мастер-реплики.

Репликация — механизм синхронизации содержимого нескольких копий объекта (например, содержимого базы данных). Репликация — это процесс, под которым понимается копирование данных из одного источника на множество других и наоборот. При репликации изменения, сделанные в одной копии объекта, могут быть распространены в другие копии. Репликация может быть синхронной или асинхронной [2].

В случае синхронной репликации, если данная реплика обновляется, все другие реплики того же фрагмента данных также должны быть обновлены в одной и той же транзакции. Логически это означает, что существует лишь одна версия данных. В большинстве продуктов синхронная репликация реализуется с помощью триггерных процедур (возможно, скрытых и управляемых системой). Но синхронная репликация имеет тот недостаток, что она создаёт дополнительную нагрузку при выполнении всех транзакций, в которых обновляются какие-либо реплики (кроме того, могут возникать проблемы, связанные с доступностью данных).

В случае асинхронной репликации обновление одной реплики распространяется на другие спустя некоторое время, а не в той же транзакции. Таким образом, при асинхронной репликации вводится задержка, или время ожидания, в течение которого отдельные реплики могут быть фактически неидентичными (то есть определение реплика оказывается не совсем подходящим, поскольку мы не имеем дело с точными и своевременно созданными копиями).

Архитектура системы показана на рисунке 1.

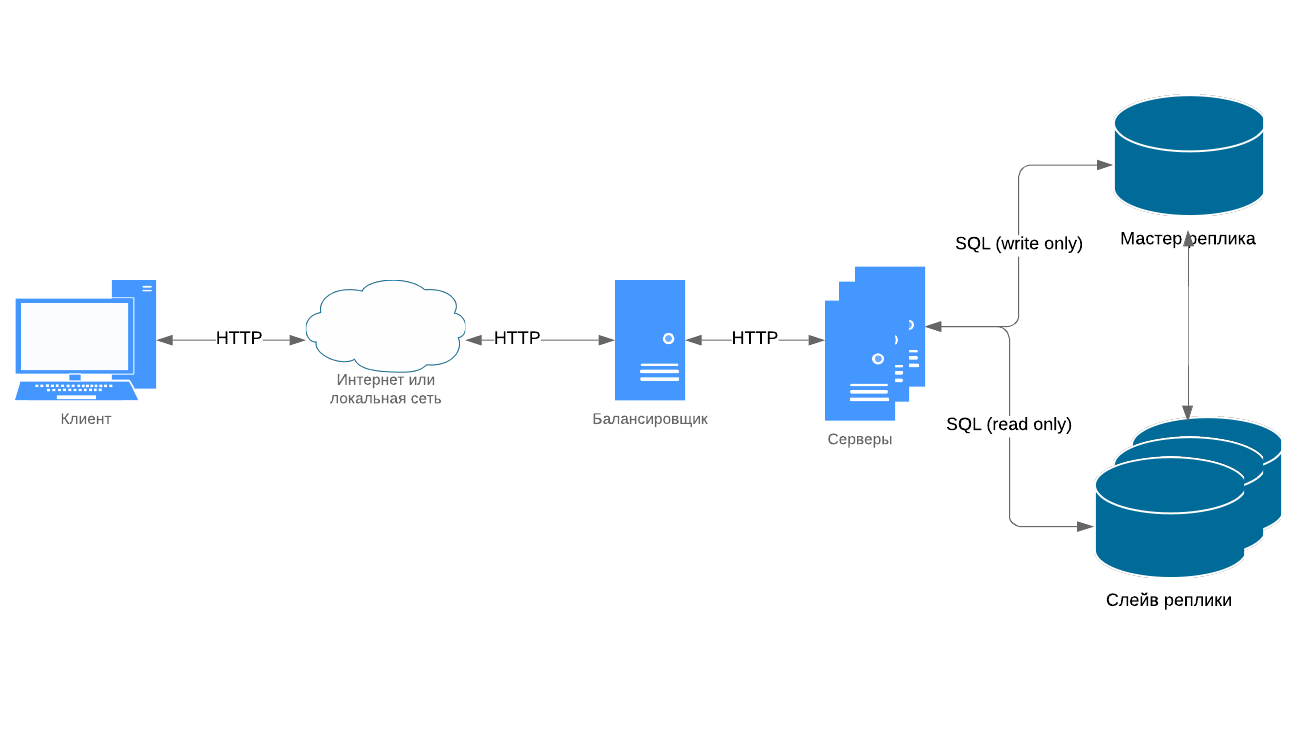


Рисунок 1 — Архитектура информационной системы

## 2.2 Выбор СУБД

Структура данных проектируемой информационной системы предполагает сложную иерархию. Поэтому следует выбрать реляционную базу данных. Подавляющее большинство баз данных работает с языком запросов sql.

Язык SQL — это непроцедурный язык, который является стандартным средством работы с данными во всех реляционных СУБД. Операторы (команды), написанные на этом языке, лишь указывают СУБД, какой результат должен быть получен, но не описывают процедуру получения этого результата. СУБД сама определяет способ выполнения команды пользователя. В языке SQL традиционно выделяются группа операторов определения данных (Data Definition Language — DDL), группа операторов манипулирования данными (Data Manipulation Language — DML) и группа операторов, управляющих привилегиями доступа к объектам базы данных (Data Control Language — DCL) [3].

Для выбора СУБД, используемой в системе, был проведен сравнительный анализ наиболее популярных продуктов по следующим критериям:

* Стоимость использования
* Поддержка технологии клиент-сервер
* Наличие средств администрирования
* Широкое распространение и поддержка

Результаты сравнения вариантов СУБД представлены в таблице 6.

Таблица 6 – Сравнительный анализ СУБД

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Microsoft SQL server | PostgreSQL | MySQL | SQLite |
| Стоимость использования | От 60 тыс. рублей | Бесплатно | Бесплатно | Бесплатно |
| Поддержка технологии клиент-сервер | да | да | да | нет |
| Наличие средств администрирования | да | да | да | да |
| Широкое распространение и поддержка | да | да | да | да |

Несмотря на то, что при сравнительном анализе СБУД «MySQL» и «PostgreSQL» показали себя равными, мной была выбрана база данных PostgreSQL как наиболее знакомая в использовании

## 2.3 Выбор HTTP сервера

Одной из самых больших проблем для веб-архитектора всегда был параллелизм. С момента появления веб-сервисов уровень параллелизма постоянно растет. Нередко популярный веб-сайт обслуживает сотни тысяч и даже миллионы одновременных пользователей. Десять лет назад основной причиной параллелизма были медленные клиенты - пользователи с ADSL или модемными соединениями.

В настоящее время параллелизм вызван сочетанием мобильных клиентов и новых архитектур приложений, которые обычно основаны на поддержании постоянного соединения, которое позволяет клиенту обновляться с помощью новостей, друзей и т. д.

Другим важным фактором, способствующим повышению параллелизма, является измененное поведение современных браузеров, которые открывают четыре-шесть одновременных подключений к веб-сайту для повышения скорости загрузки страницы [4].

Важно правильно выбрать HTTP-сервер который может обрабатывать множество соединений параллельно и не будет зависеть от медленных клиентов, проблем с сетью и других факторов.

В качестве HTTP-сервера был выбран nginx поскольку способен обрабатывать большое количество соединений одновременно, является свободно распространяемым продуктом и использует асинхронную систему ввода-вывода информации, что на практике означает экономию ресурсов системы и выгодно сказывается в случае больших нагрузок, обладает встроенной устойчивостью к многим видам DoS и DDos атак. Сравнительный анализ nginx и некоторых других HTTP-серверов приводится в исследовательской части.

**Описание nginx.**

Nginx — это HTTP-сервер и обратный прокси-сервер, почтовый прокси-сервер, а также TCP/UDP прокси-сервер общего назначения.

В nginx рабочие процессы обслуживают одновременно множество соединений. Рабочие процессы выполняют цикл обработки событий от дескрипторов. Полученные от клиента данные разбираются с помощью конечного автомата. Разобранный запрос последовательно обрабатывается цепочкой модулей, задаваемой конфигурацией. Ответ клиенту формируется в буферах, которые хранят данные либо в памяти, либо указывают на отрезок файла. Буферы объединяются в цепочки, определяющие последовательность, в которой данные будут переданы клиенту. Для чтения и отправки файлов клиенту используются отдельные потоки или системные вызовы (в зависимости от операционной системы).

Конфигурация HTTP-сервера nginx разделяется на виртуальные серверы (директива server). Виртуальные серверы разделяются на локешены (location) [5].

Для эффективного управления памятью nginx использует пулы. Пул — это последовательность предварительно выделенных блоков динамической памяти. Длина блока варьируется от 1 до 16 килобайт.

Nginx содержит модуль географической классификации клиентов по IP-адресу. В его основу входит база данных соответствия IP-адресов географическому региону.

Nginx является одним из немногих серверов, который решил проблему 10 тысяч одновременных соединенй.

## 2.4 Выбор средств разработки клиентской части системы

В качестве языка программирования клиентской части системы был выбран «JavaScript» как наиболее гибкий и распространённый. Для разработки пользовательского интерфейса была выбрана библиотека «React» поскольку является наиболее популярным и гибким инструментом для данной решения данной задачи на языке JavaScript.

**Описание JavaScript.**

JavaScript – это язык программирования для Веб. Подавляющее большинство веб-сайтов используют JavaScript, и все современные веб-браузеры – для настольных компьютеров, игровых приставок, электронных планшетов и смартфонов – включают интерпретатор JavaScript, что делает JavaScript самым широкоприменимым языком программирования из когда-либо существовавших в истории [6].

JavaScript (часто сокращенно JS) - динамический, слабо типизированный, прототипный и мультипарадигмальный язык программирования. JavaScript наряду с HTML и CSS является одной из трех основных технологий World Wide Web. JavaScript позволяет создавать интерактивные веб-страницы и, таким образом, является неотъемлемой частью веб-приложений. Для исполнения программного обеспечения написанного на JavaScript используется среда Node.js или web-браузер.

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

JavaScript имеет компактный, но очень гибкий синтаксис. Сообществом разработчиков написано большое количество инструментов, которые предоставляют доступ к большому количество дополнительных функций.

**Описание React.**

React — популярная библиотека, используемая для создания пользовательских интерфейсов. Она была создана в Facebook с целью решить ряд проблем, связанных с крупномасштабными сайтами, управляемыми данными. В момент выпуска библиотеки в 2013 году этот проект воспринимался с некоторой долей скептицизма, поскольку соглашения по React довольно необычны.

React имеет не большой размер в сравнении с аналогами и используется только для одной части всей работы. Она не содержит инструментария, ожидаемого от традиционного фреймворка JavaScript. Основные решения о том, какими средствами экосистемы воспользоваться, принимают разработчики. Кроме того, постоянно появляются новые наборы инструментов, а старые отходят на второй план. При таком количестве библиотек совершенно не удивляет ощущение, что за ними всеми невозможно угнаться [7].

React использует виртуальный DOM (англ. virtual DOM), он создает кэш структуру в памяти, что позволяет вычислять разницу между предыдущим и текущим состояниями интерфейса для оптимального обновления DOM браузера. Таким образом программист может работать со страницей, считая, что она обновляется вся, но библиотека самостоятельно решает, какие компоненты страницы необходимо обновить.

## 2.5 Выбор средств разработки серверной части системы

В качестве языка программирования серверной части был выбран Python. Для разработки серверного приложения был выбран фреймворк Django, поскольку он является наиболее функциональным и распространенным инструментом для решения данной задачи на языке Python.

**Описание Python**

Python — это интерпретируемый язык программирования общего назначения.

Для многих основное преимущество языка Python заключается в удобочитаемости, ясности и более высоком качестве, отличающими его от других инструментов в мире языков сценариев. Программный код на языке Python читается легче, а значит, многократное его использование и обслуживание выполняется гораздо проще, чем использование программного кода на других языках сценариев. Единообразие оформления программного кода на языке Python облегчает его понимание даже для тех, кто не участвовал в его создании. Кроме того, Python поддерживает самые современные механизмы многократного использования программного кода, каким является объектно-ориентированное программирование (ООП) [8].

По сравнению с компилирующими или строго типизированными языками, такими как C, C++ и Java, Python во много раз повышает производительность труда разработчика. Объем программного кода на языке Python обычно составляет треть или даже пятую часть эквивалентного программного кода на языке C++ или Java. Это означает меньший объем ввода с клавиатуры, меньшее количество времени на отладку и меньший объем трудозатрат на сопровождение. Кроме того, программы на языке Python запускаются сразу же, минуя длительные этапы компиляции и связывания, необходимые в некоторых других языках программирования, что еще больше увеличивает производительность труда программиста.

Большая часть программ на языке Python выполняется без изменений на всех основных платформах. Перенос программного кода из операционной системы Linux в Windows обычно заключается в простом копировании файлов программ с одной машины на другую. Более того, Python предоставляет массу возможностей по созданию переносимых графических интерфейсов, программ доступа к базам данных, веб-приложений и многих других типов программ. Даже интерфейсы операционных систем, включая способ запуска программ и обработку каталогов, в языке Python реализованы переносимым способом.

В составе Python поставляется большое число собранных и переносимых функциональных возможностей, известных как стандартная библиотека. Эта библиотека предоставляет массу возможностей, востребованных в прикладных программах, начиная от поиска текста по шаблону и заканчивая сетевыми функциями. Кроме того, Python допускает расширение как за счет ваших собственных библиотек, так и за счет библиотек, созданных сторонними разработчиками. Из числа сторонних разработок можно назвать инструменты создания веб-сайтов, программирование математических вычислений, доступ к последовательному порту, разработку игровых программ и многое другое. Например, расширение NumPy позиционируется как свободный и более мощный эквивалент системы программирования математических вычислений Mathlab.

Сценарии Python легко могут взаимодействовать с другими частями приложения благодаря различным механизмам интеграции. Эта интеграция позволяет использовать Python для настройки и расширения функциональных возможностей программных продуктов. На сегодняшний день программный код на языке Python имеет возможность вызывать функции из библиотек на языке C/C++, сам вызываться из программ, написанных на языке C/C++, интегрироваться с программными компонентами на языке Java, взаимодействовать с такими платформами, как COM и .NET, и производить обмен данными через последовательный порт или по сети с помощью таких протоколов, как SOAP, XML-RPC и CORBA. Python – не обособленный инструмент.

**Описание Django**

Django — это веб-среда Python высокого уровня, которая способствует быстрой разработке и чистому, прагматичному дизайну. Созданный опытными разработчиками, он берет на себя большую часть хлопот веб-разработки, поэтому вы можете сосредоточиться на написании своего приложения без необходимости повторять уже несписанные программы [9]. Django распространяется бесплатно и имеет открытый исходный код.

В составе Django присутствует собственный веб-сервер для разработки. Сервер автоматически определяет изменения в файлах исходного кода проекта и перезапускается, что ускоряет процесс разработки на Python, но пригоден только для процесса разработки и отладки приложения.

Некоторые возможности Django:

* ORM с поддержкой транзакций для доступа к БД;
* встроенный интерфейс администратора, с уже имеющимися переводами на многие языки;
* диспетчер URL на основе регулярных выражений;
* система кеширования;
* встроенная автоматическая документация по тегам шаблонов и моделям данных, доступная через административное приложение.

## 2.6 Проектирование базы данных

### 2.6.1 Инфологическая модель

**Теоретическая часть**

На этапе анализа предметной области была проведена предварительная структуризация объектов предметной области.

Следующим шагом является решение вопроса, какая информация об объектах должна храниться в базе данных и каким образом. Цель концептуального этапа проектирования базы данных – установление соответствия между состоянием предметной области, его восприятием и представлением в базе данных.

Одним из наиболее известных и получивших широкое распространение методов семантического моделирования является метод построения модели «сущность-связь» (или ER-модели). Этот подход основан на использовании модели «сущность-связь», предложенной в 1976 году и с тех пор неоднократно дополнявшейся многими исследователями [10].

Для описания инфологической схемы используются диаграммы «сущность – связь» (ER-диаграмма). Эта схема предоставляет возможность моделировать объекты предметной области и связи между ними. Для диаграммы «сущность – связь» существуют формальные механизмы преобразования в понятия реляционной модели данных, поддерживаемой подавляющим большинством современных СУБД.

Для построения ER-диаграмма используются три основных конструктивных элемента – сущность, атрибут и связь.

Сущность – именованная конструкция для моделирования объектов предметной области. Это собирательное понятие, некоторая абстракция реально существующего объекта, процесса или явления, информацию о котором необходимо хранить в базе данных. Каждый экземпляр сущности должен быть уникален. Для этого вводят первичные и вторичные ключи.

Первичный ключ (Primary Key) – это атрибут (или группа атрибутов), однозначно идентифицирующий экземпляр сущности. При этом его значение находится во взаимно-однозначном соответствии с экземплярами сущности. Первичный ключ должен удовлетворять ряду требований:

* Уникальность. Два экземпляра сущности не должны иметь одинаковых значений ключа.
* Минимум атрибутов в ключе. Составной первичный ключ (группа атрибутов) не должен содержать ни одного атрибута, удаление которого не приводило бы к утрате уникальности.

Вторичный (или внешний) ключ (Foreign Key) – появляется в процессе установления связей между сущностями. Это описательный атрибут, который назначается в дополнение к идентифицирующему атрибуту и служит не для идентификации этого уникального экземпляра, а для выделения из набора тех объектов, которые обладают интересующими нас свойствами. Этот ключ часто называют "ключом поиска". Ограничения на состав вторичных ключей определяются логикой запроса

Атрибут – именованная конструкция для моделирования свойств объекта предметной области (сущности). Основное назначение атрибута – описание свойства сущности, а также идентификация экземпляра сущностей.

Связь – именованная конструкция для моделирования отношений между объектами (сущностями). При анализе связей между сущностями могут встречаться унарные (рекурсивные) связи, когда экземпляр определенного типа сущности связан с другим экземпляром той же самой сущности, бинарные (между двумя сущностями), тернарные (между тремя сущностями) и, в общем случае n-арные связи. Наиболее часто встречаются бинарные связи. При проведении классификации видов связей обычно выделяют следующие виды связей:

* отображение вида 1:1 (связь "один-к-одному");
* отображение вида 1:М (связь "один-ко-многим");
* отображение вида М:1 (связь "многие-к-одному");
* отображение вида М:М (связь "многие-ко-многим").

Различают зависимые и независимые сущности. Тип сущности

определяется связью между сущностями.

Независимая сущность – обычная сущность, которая не зависит при

идентификации от других сущностей в модели, или просто независимая, если каждый экземпляр может быть однозначно идентифицирован без определения его отношений с другими объектами.

Зависимая сущность – это дочерняя сущность, уникальность которой зависит от атрибута внешнего ключа, или просто объект, у которой однозначная идентификация экземпляра одной сущности зависит от его отношения к другой сущности.

Зависимые сущности также делятся на сущности, которые не могут существовать без родительской сущности, и сущности, которые не могут быть идентифицированы без использования ключа родителя (сущности, зависящие от идентификации).

Идентифицирующая связь – это связь, которая характеризуется глагольной формой со сказуемым "должен", где подлежащим является имя дочерней таблицы. Она применяется, когда необходимо, чтобы внешний ключ передавался в дочернюю сущность (и, в результате, создавал зависимую сущность). Идентифицирующая связь устанавливается между независимой (родительской – начало связи) и зависимой (дочерней – конец связи) сущностями.

Вважно также обратить внимание на то, что при установлении идентифицирующей связи атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности. В дочерней сущности такие атрибуты помечаются как внешний ключ (foreign key).

Неидентифицирующая связь – это связь, которая характеризуется глагольной формой со сказуемым "может", где подлежащим является имя дочерней таблицы. Она применяется, когда необходимо отобразить другой тип передачи атрибутов внешних ключей – передача в область данных дочерней сущности.

Важно также обратить внимание на то, что при установлении неидентифицирующей связи происходит миграция атрибутов первичного ключа родительской сущности не в состав первичного ключа дочерней сущности, а в состав неключевых атрибутов дочерней сущности.

Существенно, что при идентифицирующей связи внешний ключ в дочерней сущности не может принимать значение NULL. Если внешнийключ должен существовать, то это означает, что запись в дочерней сущности может существовать только при наличии ассоциированной с ним родительской записи.

В таблицах 7-15 приведены сущности и их атрибуты

Таблица 7 – Сущность «Новость»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Заголовок | Текстовый |  |
| Содержание | Текстовый |  |
| Дата добавления | Дата |  |
| Идентификатор автора | Числовой | Внешний |

Таблица 8 – Сущность «Группа студентов»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Название | Текстовый |  |
| Дата начала обучения | Дата |  |

Таблица 9 – Сущность «Дисциплина»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Название | Текстовый |  |
| Сокращённое название | Текстовый |  |
| Идентификатор группы | Числовой | Внешний |
| Идентификатор преподавателя | Числовой | Внешний |

Таблица 10 – Сущность «Занятие»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Название | Текстовый |  |
| Описание | Текстовый |  |
| Дата проведения | Дата |  |
| Место проведения | Текстовый |  |

Таблица 11 – Сущность «Оценка»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Идентификатор студента | Числовой | Внешний |
| Идентификатор занятия | Числовой | Внешний |
| Значение | Числовой |  |
| Дата добавления | Дата |  |

Таблица 12 – Сущность «Студент»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Идентификатор группы | Числовой | Внешний |
| Идентификатор пользователя | Числовой | Внешний |

Таблица 13 – Сущность «Преподаватель»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Должность | Текстовый |  |
| Идентификатор пользователя | Числовой | Внешний |

Таблица 14 – Сущность «Пользователь»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Фамилия | Текстовый |  |
| Имя | Текстовый |  |
| Отчество | Текстовый |  |
| Электронная почта | Текстовый |  |
| Пароль | Текстовый |  |

Таблица 15 – Сущность «Сессия»

|  |  |  |
| --- | --- | --- |
| Логическое имя атрибута | Тип данных | Ключ |
| Идентификатор | Числовой | Первичный |
| Ключ доступа | Текстовый |  |
| Идентификатор пользователя | Числовой | Внешний |
| Дата создания | Дата |  |

**Связи между сущностями.**

1. «Публикует»: связь между сущностями «Пользователь» и «Новость» с типом связи 1:М (связь «один-ко-многим»);
2. «Преподаётся»: связь между сущностями «Группа» и «Дисциплина» с типом связи 1:М (связь «один-ко-многим»);
3. «Соответствует»: связь между сущностями «Дисциплина» и «Занятие» с типом связи 1:М (связь «один-ко-многим»);
4. «Преподаёт»: связь между сущностями «Преподаватель» и «Дисциплина» с типом связи 1:М (связь «один-ко-многим»);
5. «Получает»: связь между сущностями «Студент» и «Оценка» с типом связи 1:М (связь «один-ко-многим»);
6. «Относится»: связь между сущностями «Занятие» и «Оценка» с типом связи 1:М (связь «один-ко-многим»);
7. «Входит»: связь между сущностями «Студент» и «Группа» с типом связи 1:М (связь «один-ко-многим»);
8. «Является»: связь между сущностями «Студент» и «Пользователь» с типом связи 1:1 (связь «один-ко-многим»);
9. «Является»: связь между сущностями «Преподаватель» и «Пользователь» с типом связи 1:1 (связь «один-ко-многим»);
10. «Относится»: связь между сущностями «Сессия» и «Пользователь» с типом связи 1:1 (связь «один-ко-многим»).

Инфологическая модель изображена на рисунке 2.

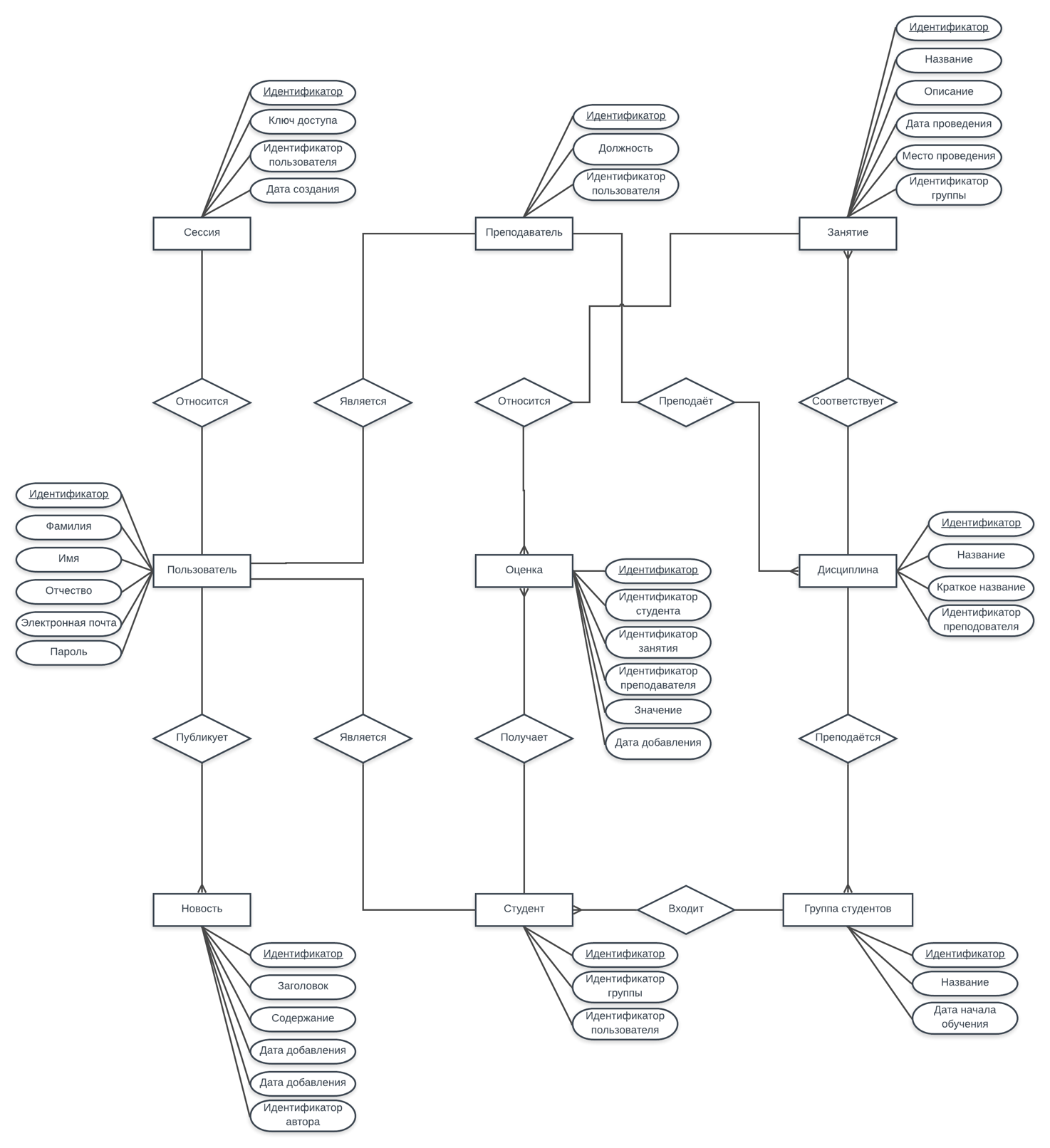


Рисунок 2 — Инфологическая модель

### 2.6.2 Даталогическая модель

Даталогическая модель (ДЛМ) базы данных является моделью логического уровня и представляет собой отображение логических связей между элементами [11].

данных безотносительно к среде хранения. Эта модель строится в терминах информационных единиц, допустимых в той конкретной СУБД, в среде которой мы проектируем

базу данных. Этап создания ДЛМ называется даталогическим проектированием. Описание логической структуры базы данных на языке СУБД называется схемой.

Построение даталогической модели основано на разработанной ранее инфологической модели.

Для обеспечения целостности данных в соответствующие сущности инфологической модели добавляются ключевые атрибуты, которые функционально определены и зависят только от первичного ключа, и не имеют связи между собой. При отображении инфологической модели в даталогическую, сущности соответствуют таблицами, атрибуты — полям таблиц. В даталогическую модель вводятся связи, аналогичные связям инфологической модели.

В таблицах 16-24 приведены названия и типы полей сущностей

Таблица 16 – Сущность «news»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Заголовок | title | text | Not null |  |
| Содержание | content | text | Not null |  |
| Дата добавления | created\_at | datetime | Not null |  |
| Идентификатор автора | author\_id | int | Not null | Foreign Key |

Таблица 17 – Сущность «group»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Название | name | text | Not null |  |
| Дата начала обучения | start\_at | date | Not null |  |

Таблица 18 – Сущность «discipline»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Название | name | text | Not null |  |
| Сокращённое название | short\_name | text | Null |  |
| Идентификатор группы | group\_id | int | Not null | Foreign Key |
| Идентификатор преподавателя | teacher\_id | int | Not null | Foreign Key |

Таблица 19 – Сущность «activity»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Название | name | text | Not null |  |
| Описание | description | text | Not null |  |
| Дата проведения | date | datetime | Not null |  |
| Место проведения | place | text | Not null |  |
| Идентификатор дисциплины | disciplin\_id | int | Not null | Foreign Key |

Таблица 20 – Сущность «mark»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Идентификатор студента | student\_id | int | Not null | Foreign Key |
| Идентификатор занятия | activity\_id | int | Not null | Foreign Key |
| Значение | value | int | Not null |  |
| Дата добавления | date | datetime | Not null |  |

Таблица 21 – Сущность «student»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Идентификатор группы | group\_id | int | Null | Foreign Key |
| Идентификатор пользователя | user\_id | int | Not null | Foreign Key |

Таблица 22 – Сущность «teacher»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Должность | position | text | Not null |  |
| Идентификатор пользователя | user\_id | int | Not null | Foreign Key |

Таблица 23 – Сущность «user»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | id | Not null | Primary Key |
| Фамилия | last\_name | text | Not null |  |
| Имя | first\_name | text | Not null |  |
| Отчество | patronymic | text | Not null |  |
| Электронная почта | email | text | Not null |  |
| Пароль | password | text | Not null |  |

Таблица 24 – Сущность «session»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логическое имя атрибута | Поле | Тип данных | Null/Not null | Ключ |
| Идентификатор | id | int | Not null | Primary Key |
| Ключ доступа | access\_key | uuid | Not null |  |
| Идентификатор пользователя | user\_id | id | Not null | Foreign Key |
| Дата создания | create\_at | datetime | Not null |  |

На рисунке 3 изображена датологическая модель.

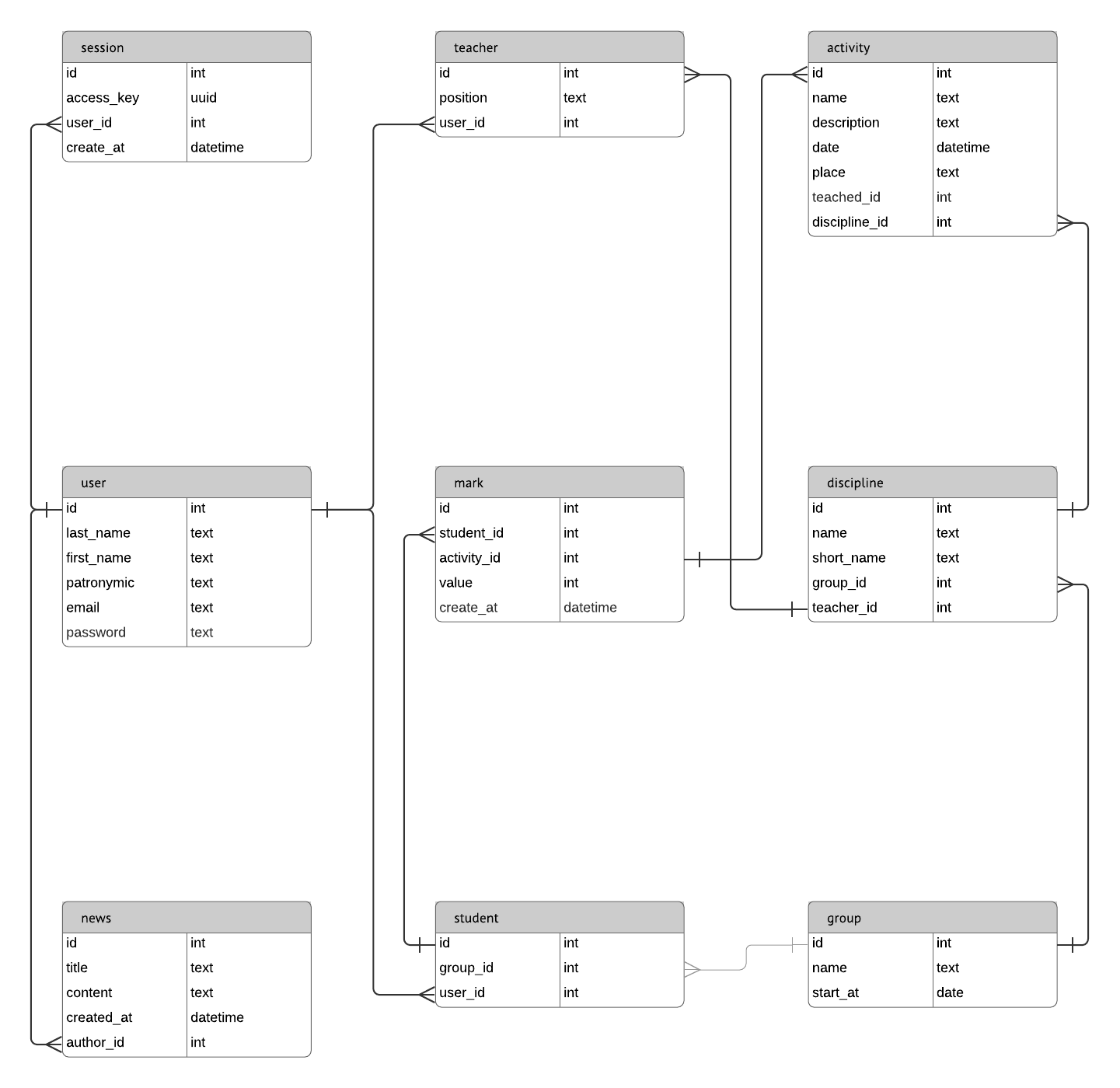


Рисунок 3 – Даталогическая модель

## 2.7 Выбор программных средств

Для корректной работы серверной части данного программного продукта требуется следующее программное обеспечение:

* CУБД PostgreSQL 10.6;
* веб-сервер NginX;
* вредство разработки Python 3.7;
* фрейморк для web-сервера Django 2.1;
* операционная система Ubuntu 16.04.

В качестве операционной системы была выбрана Ubuntu поскольку она имеет следующие преимущества:

* свободно распространяемая операционная система;
* срок поддержки 5 лет;
* легка в настройке.

Для корректной работы системы на компьютере или ноутбуке пользователя требуется следующее программное обеспечение:

* ОС (не регламентируется);
* веб-браузер Opera 58.0.

## 2.8 Выбор аппаратных средств

Для корректной работы серверной части данного программного продукта требуется следующее минимальное аппаратное обеспечение:

* процессор Intel Core 2 Quad (с тактовой частотой 2.5 ГГц);
* оперативную память объемом 8 Гб;
* жёсткий диск объемом 500 Гб, интерфейс: SATA;
* сетевая карта с выходом в сеть Интернет.

Для корректной работы системы на компьютере или ноутбуке пользователя требуется следующее минимальное аппаратное обеспечение:

* процессор Intel Pentium 4 / Athlon 64;
* оперативную память объемом 512 Мб;
* свободное место на диске 350 Мб;
* сетевая карта с выходом в сеть;
* монитор с разрешением 1365×768.

## 2.9 Разработка интерфейса взаимодействия с пользователем

Поскольку система выполнена в виде полноценного клиент серверного приложения то переход между странницами, получение данных с сервера отправка данных форм не требуют перезагрузки вкладки браузера.

Граф диалога с неавторизованным пользователем показан на рисунке 4. Возможные действия — регистрация и вход в систему.

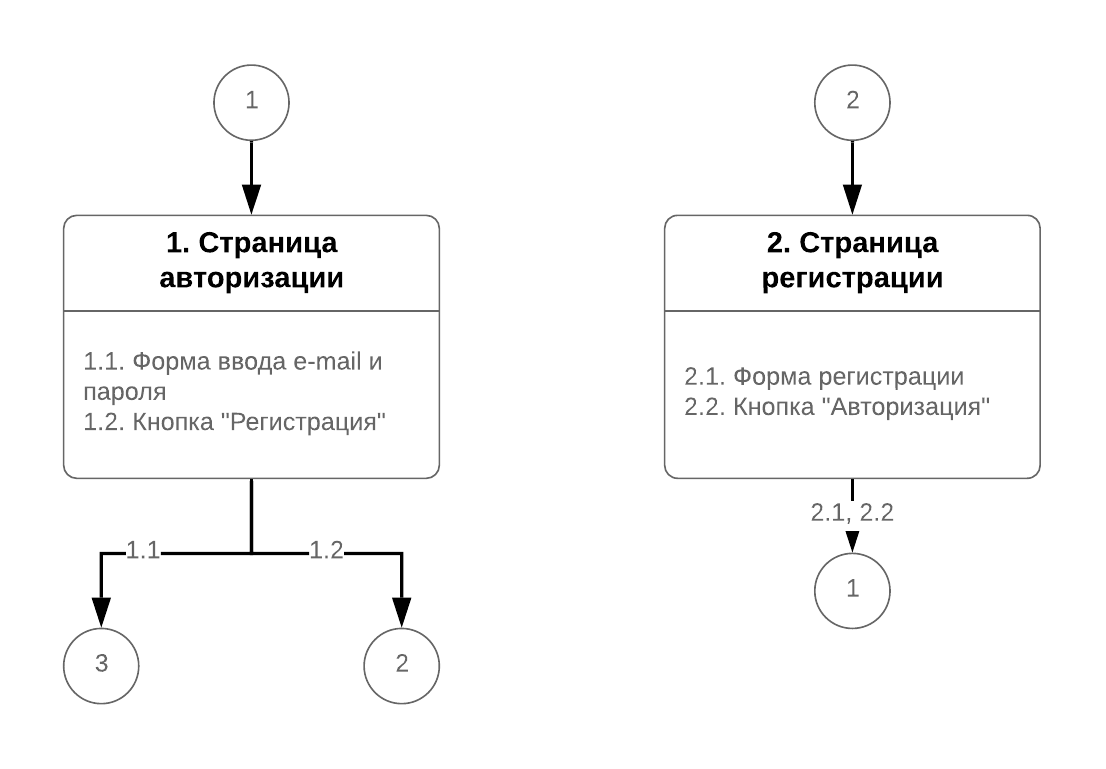


Рисунок 4 - граф диалога с неавторизованным пользователем

Интерфейс авторизации пользователя показан на рисунке 5.

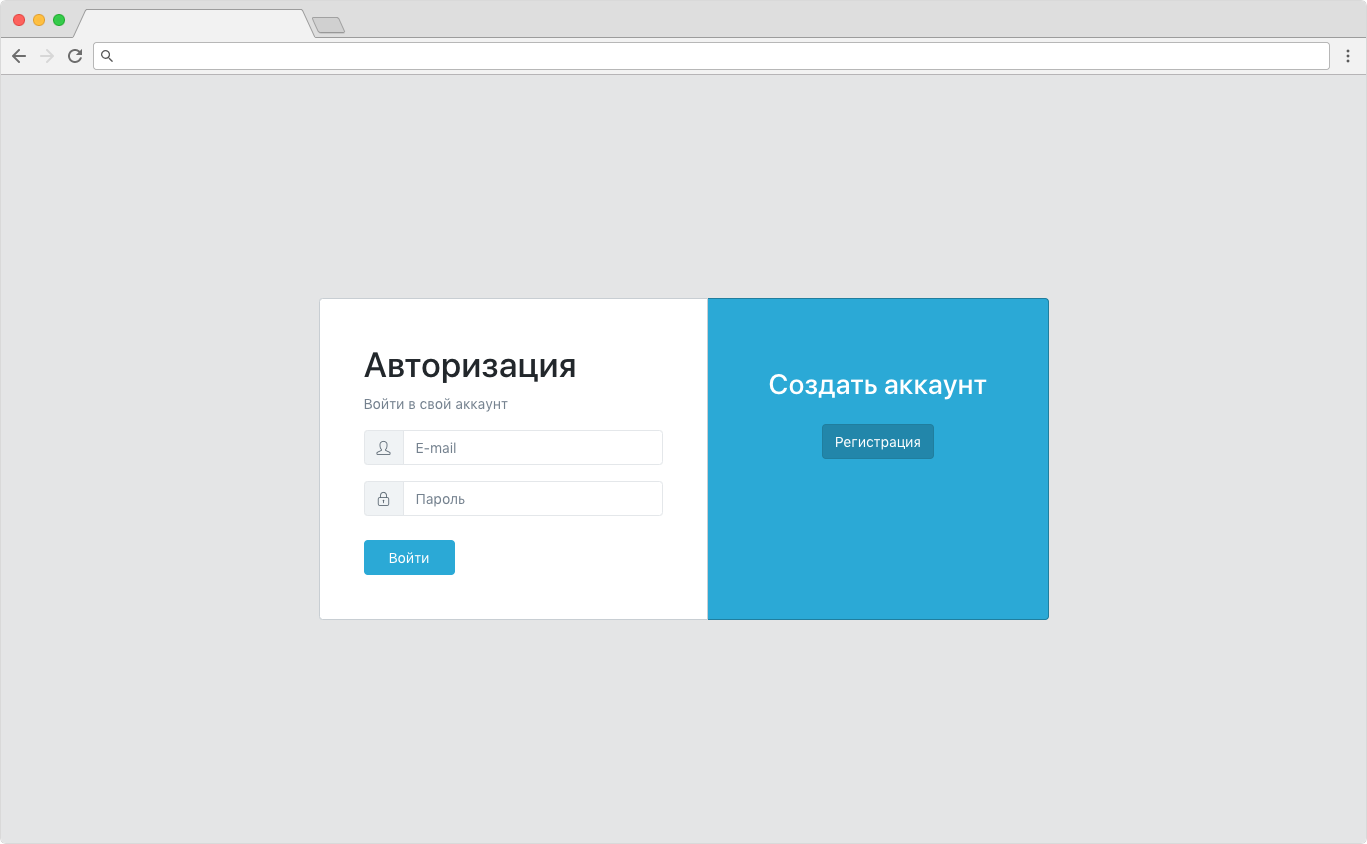


Рисунок 5 – авторизация пользователя

Интерфейс регистрации пользователя показан на рисунке 6.

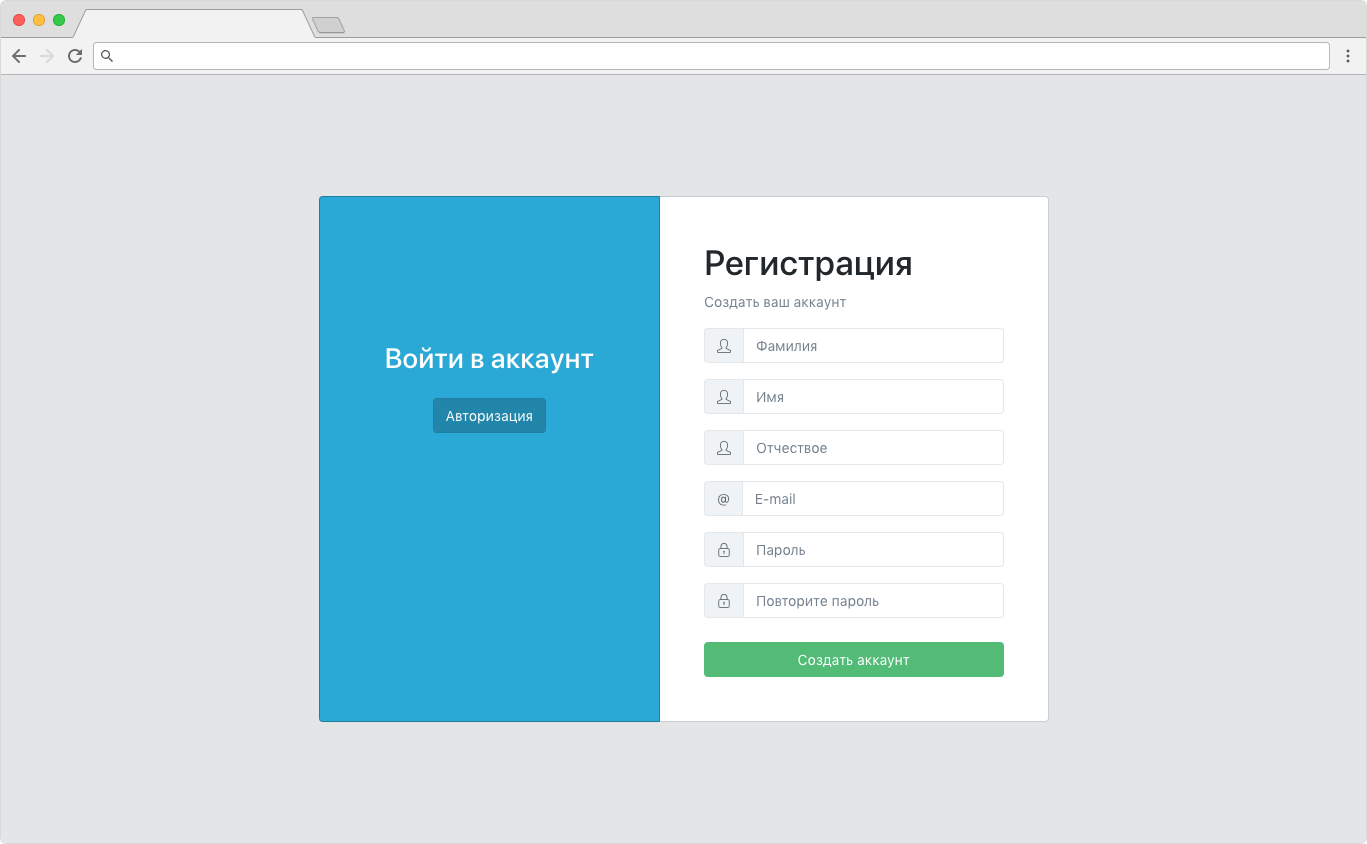


Рисунок 6 – регистрация пользователя

### 2.9.2 Интерфейс взаимодействия с модератором

Граф диалога с модератором системы показан на рисунке 7.

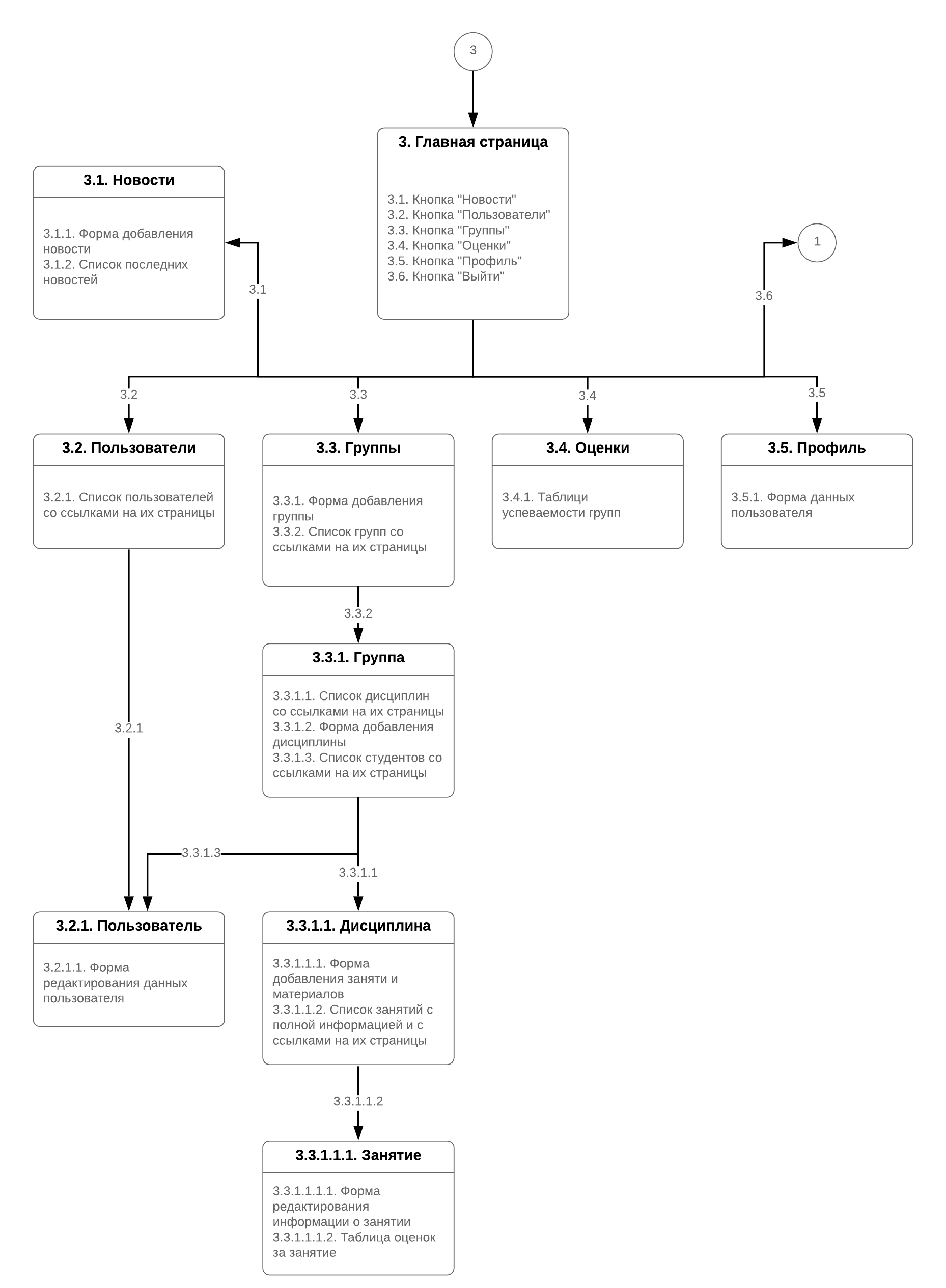


Рисунок 7 - граф диалога с модератором

Кнопки главной страницы доступны на всех подстранницах.

Интерфейс страницы новостей показан на рисунке 8. На странице есть возможность добавлять и просматривать новости.

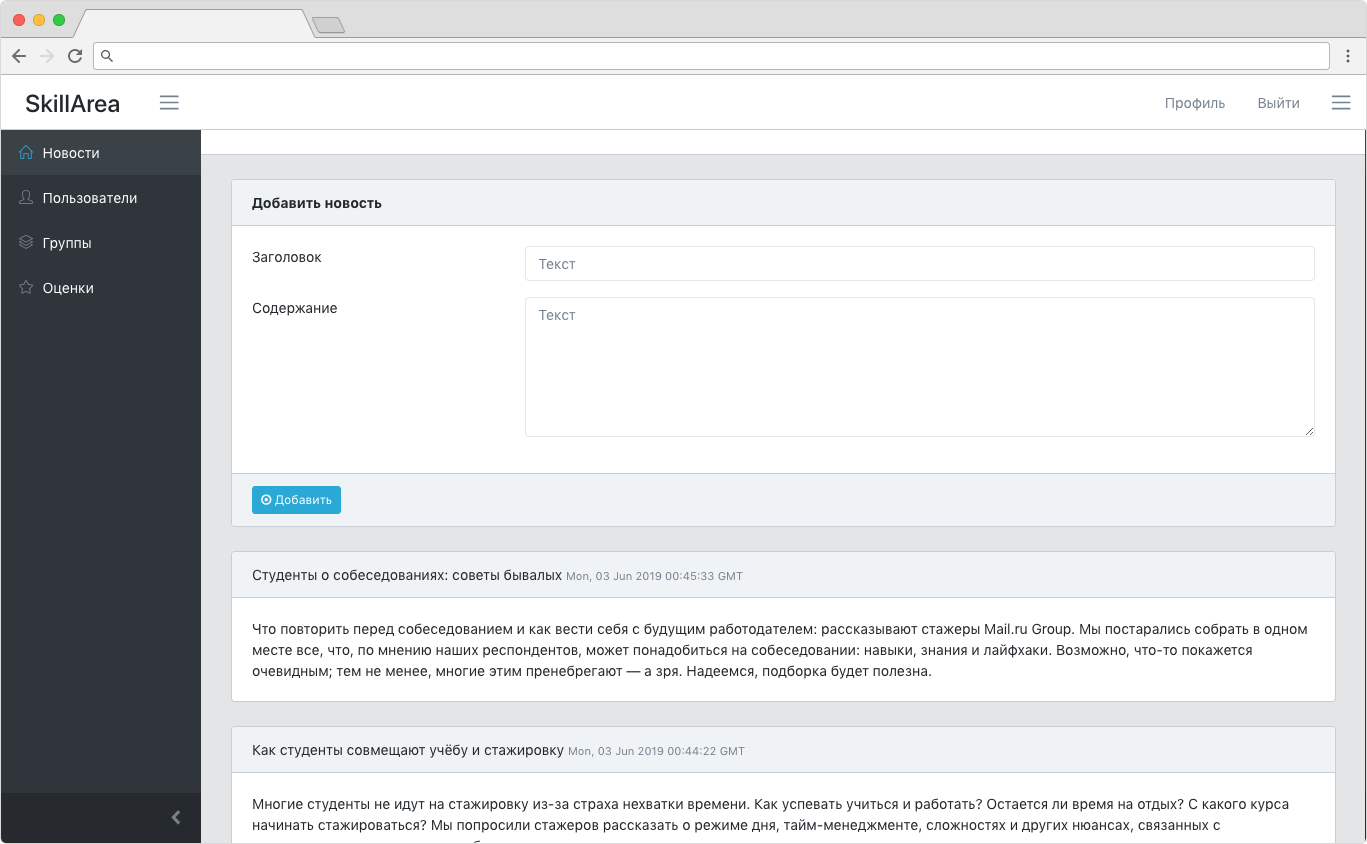


Рисунок 8 – страница новостей

Интерфейс страницы пользователей показан на рисунке 9. На странице представлен список всех пользователей

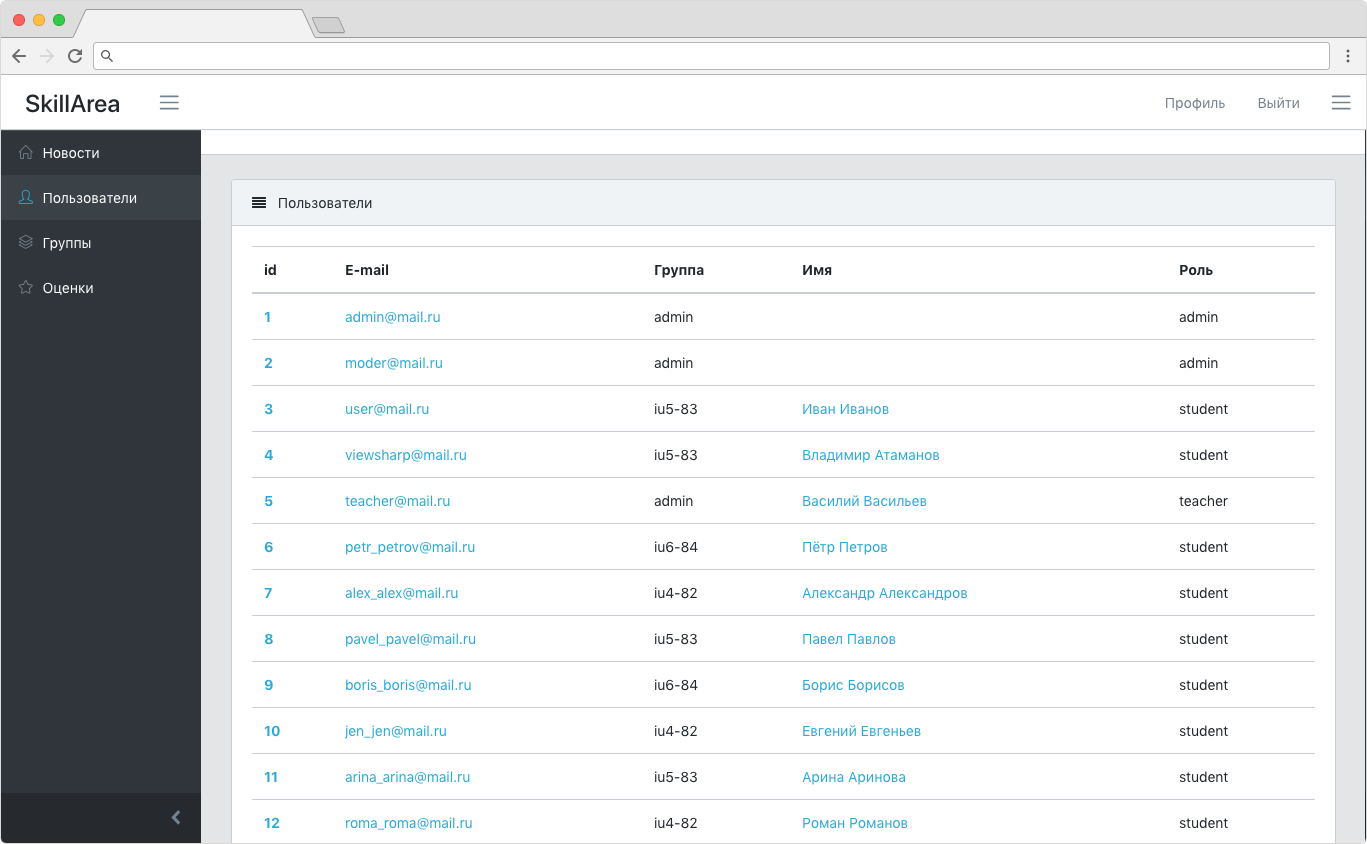


Рисунок 9 – страница пользователей

Интерфейс страницы пользователя показан на рисунке 10. На странице есть возможность редактирования данных пользователя.

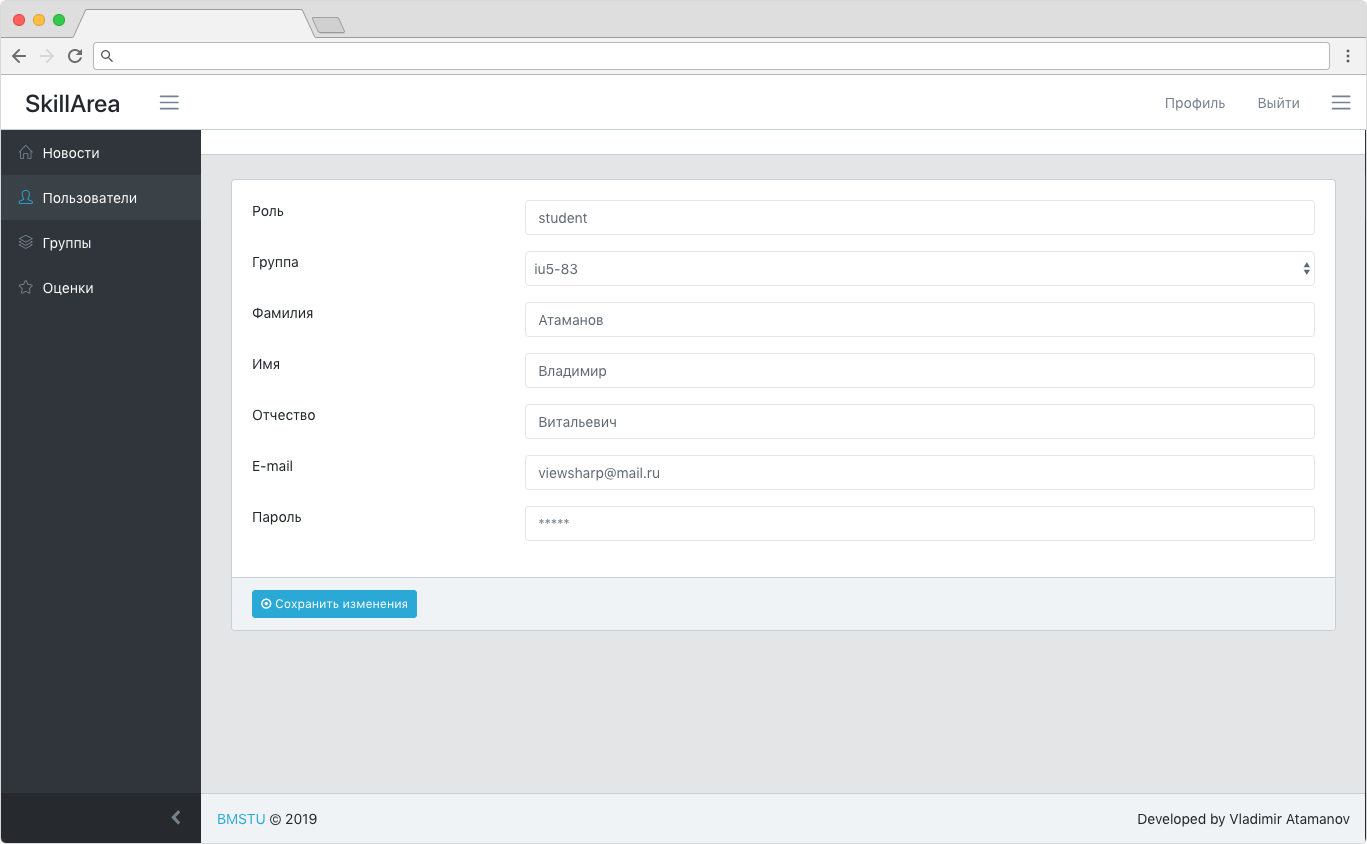


Рисунок 10 – страница пользователя

Интерфейс страницы групп показан на рисунке 11. На странице есть возможность просматривать список групп и добавлять новые.

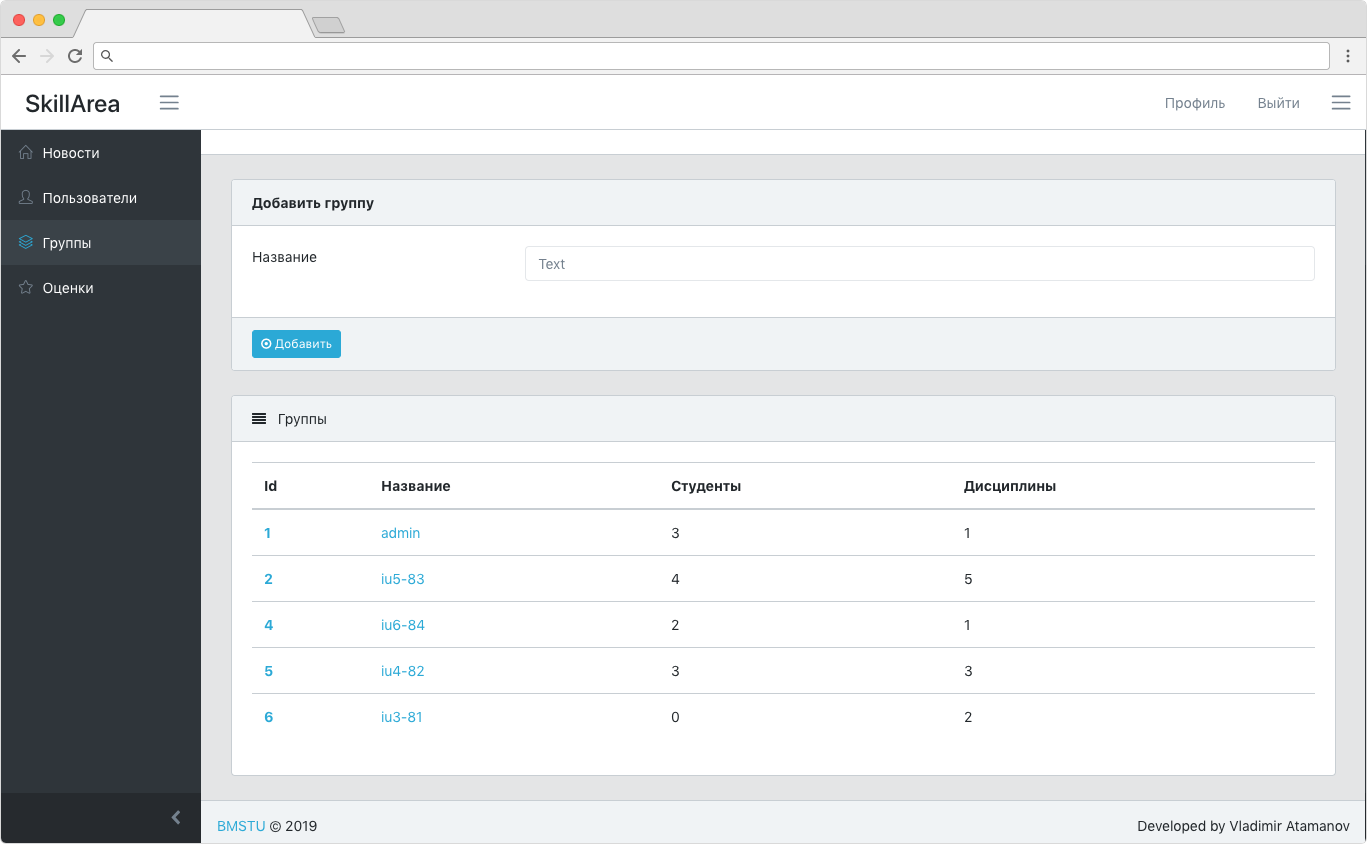


Рисунок 11 – страница групп

Интерфейс страницы группы показан на рисунке 12. На странице есть возможность просматривать список студентов, список преподаваемых дисциплин и добавлять новые.

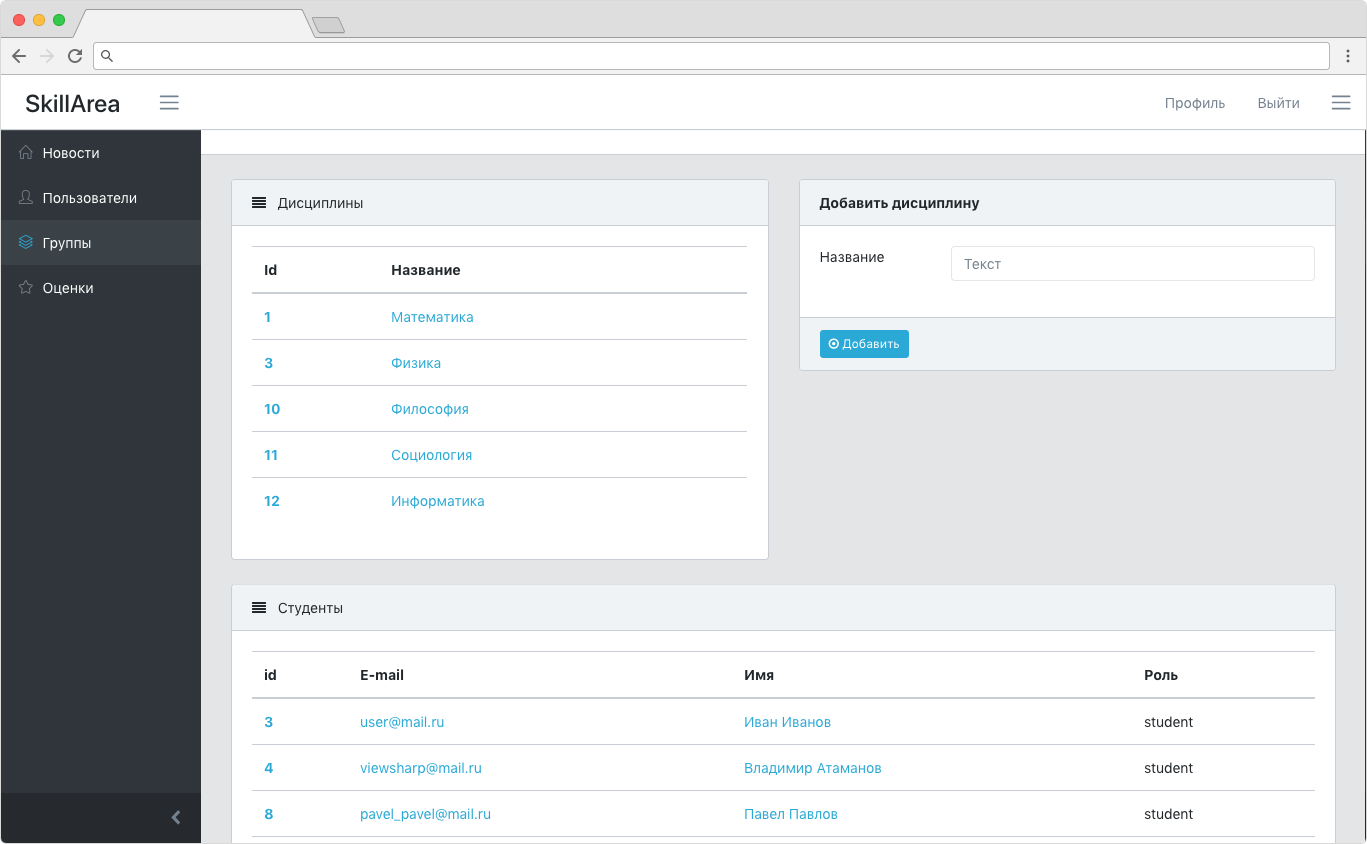


Рисунок 12 – страница групп

Интерфейс страницы дисциплины показан на рисунке 13. На странице есть возможность просматривать список занятий и их материалы и добавлять новые.

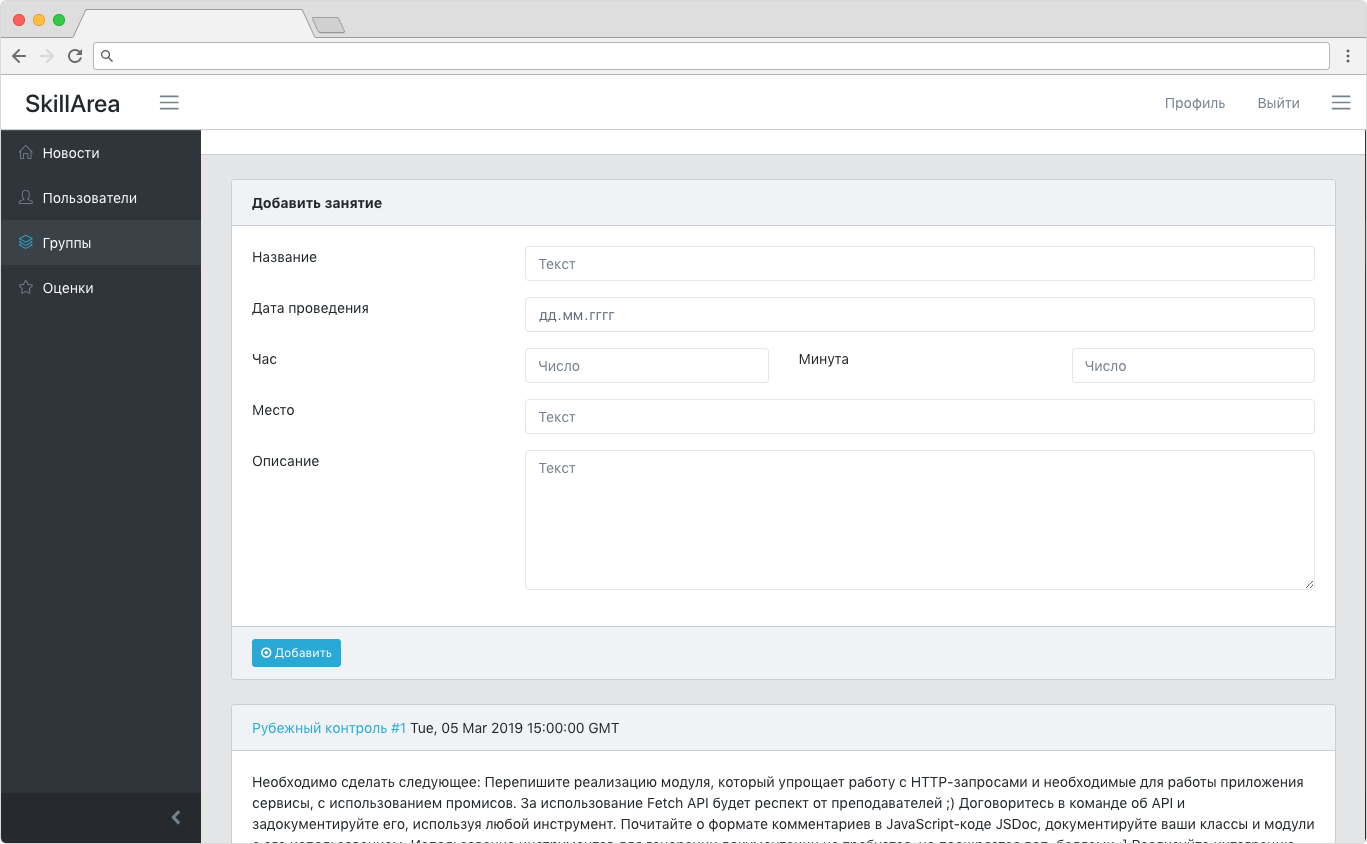


Рисунок 13 – страница дисциплины

Интерфейс страницы занятия показан на рисунке 14. На странице есть возможность редактировать информацию о занятии и выставлять оценки студентам.

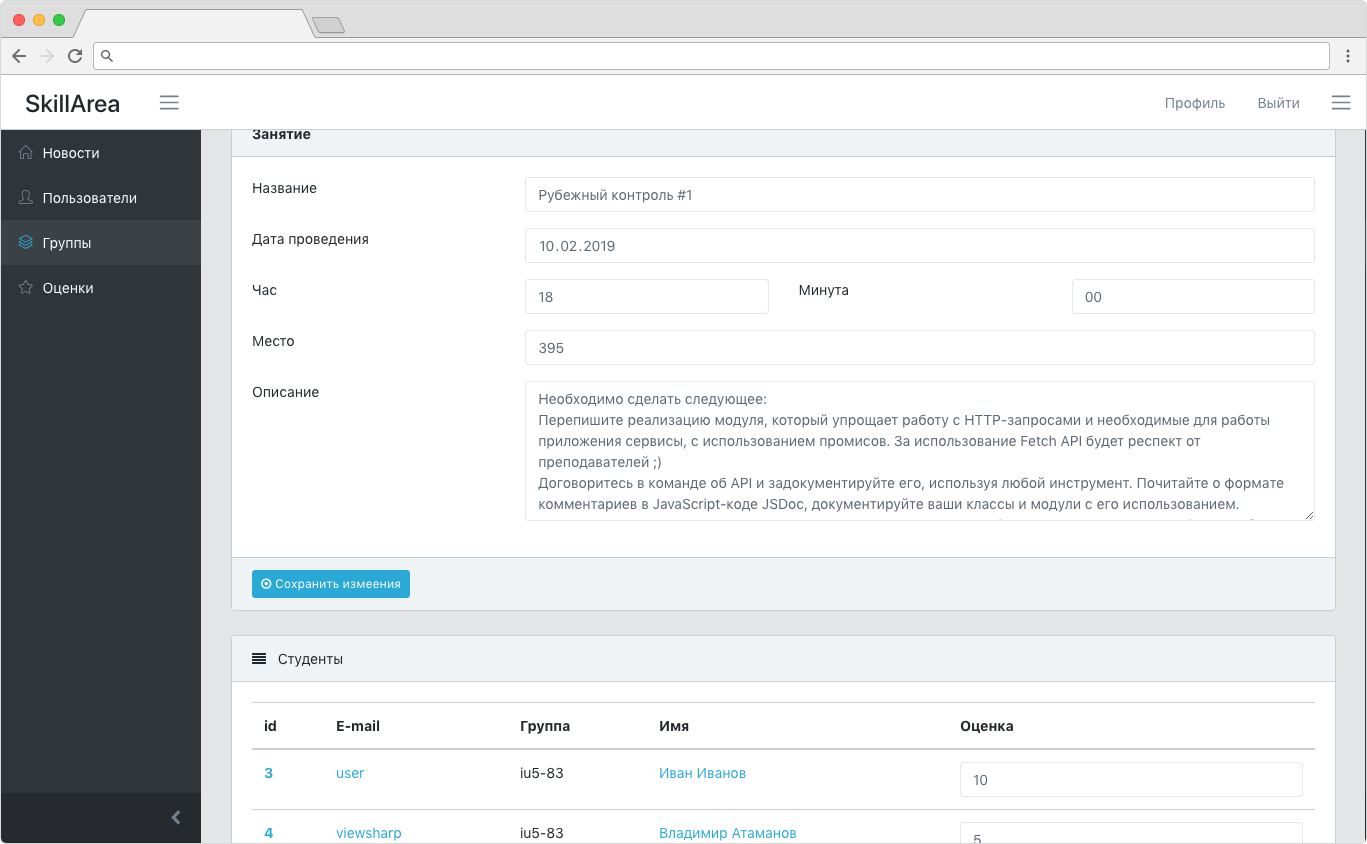


Рисунок 14 – страница занятия

Интерфейс страницы оценок показан на рисунке 15. На странице представлены таблицы успеваемости студентов.

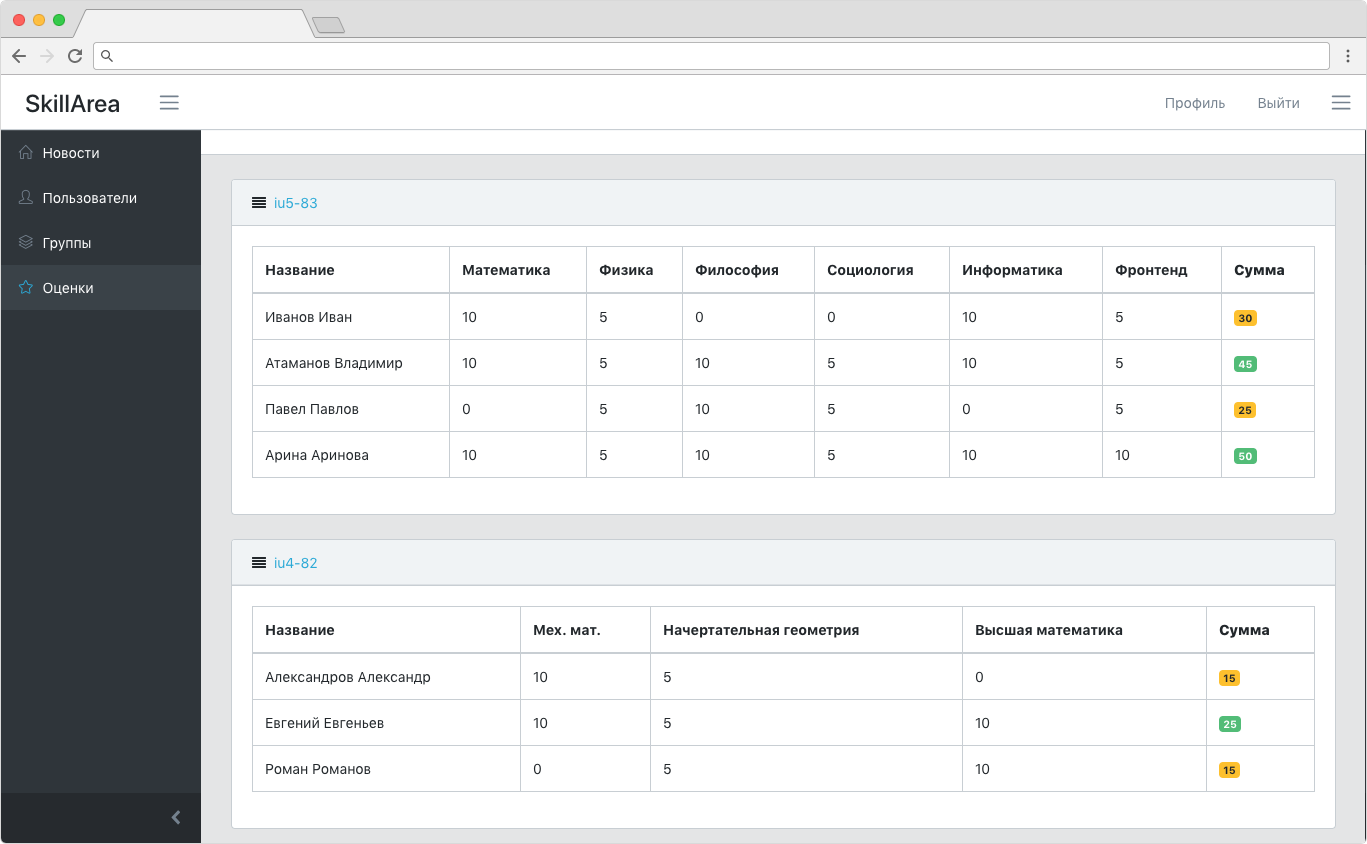


Рисунок 15 – страница оценок

Интерфейс страницы профиля показан на рисунке 16. На странице есть возможность редактирования данных профиля.

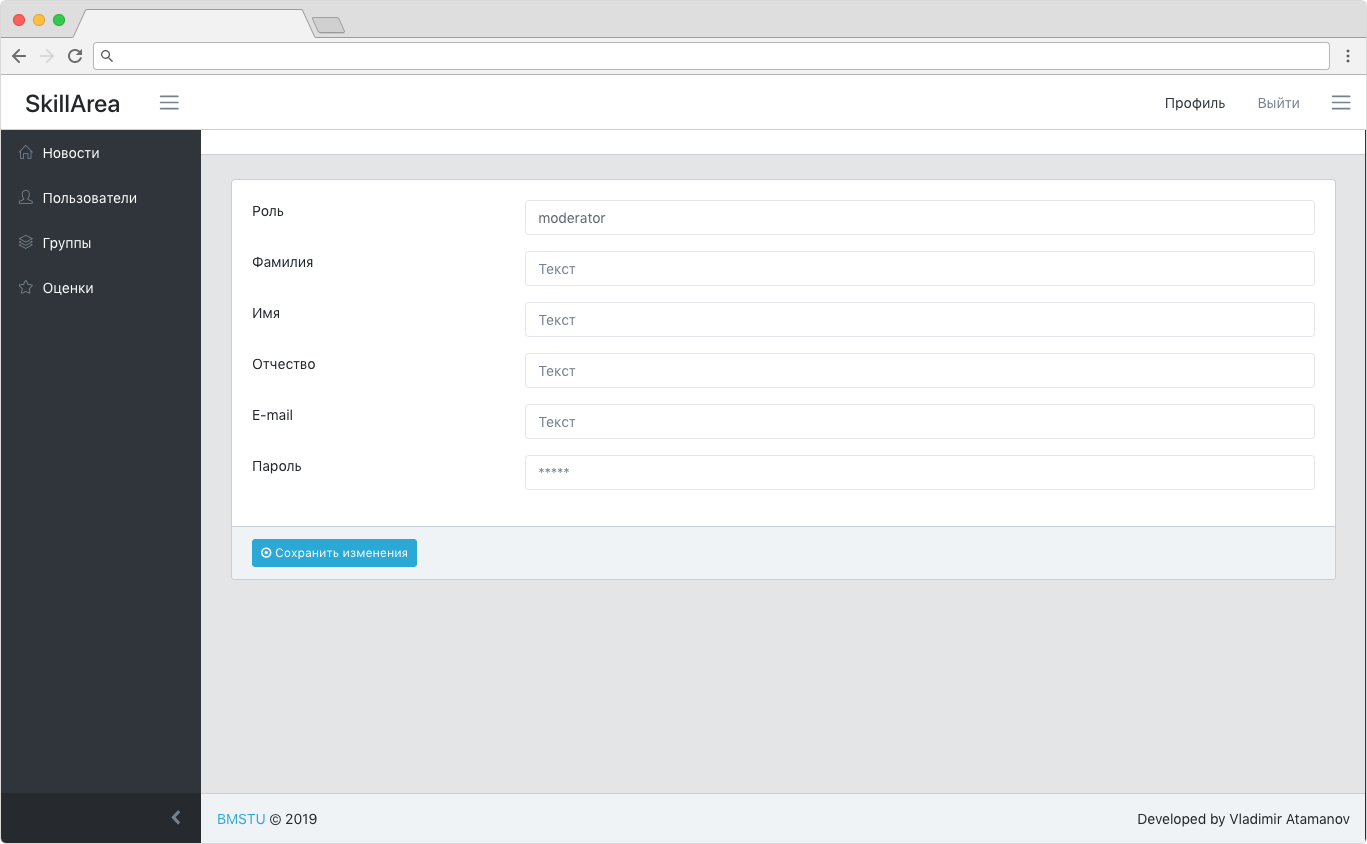


Рисунок 16 – страница профиля

### 2.9.3 Интерфейс взаимодействия с преподавателем

Граф диалога с преподавателем полностью соответствует графу диалога с модератором. Главные отличия между этими ролями заключаются в том, что преподаватель не имеет права на редактирование данных чужих профилей, не имеет права менять свою роль и не имеет доступа к дисциплинам, который не преподаёт.

Преподаваемые дисциплины назначаются модератором на странице редактирования информации пользователя преподавателя.

Поскольку у преподавателя нет прав на редактирование чужих профилей, страница пользователя для него выглядит иначе. Интерфейс страницы пользователя представлен на рисунке 17.

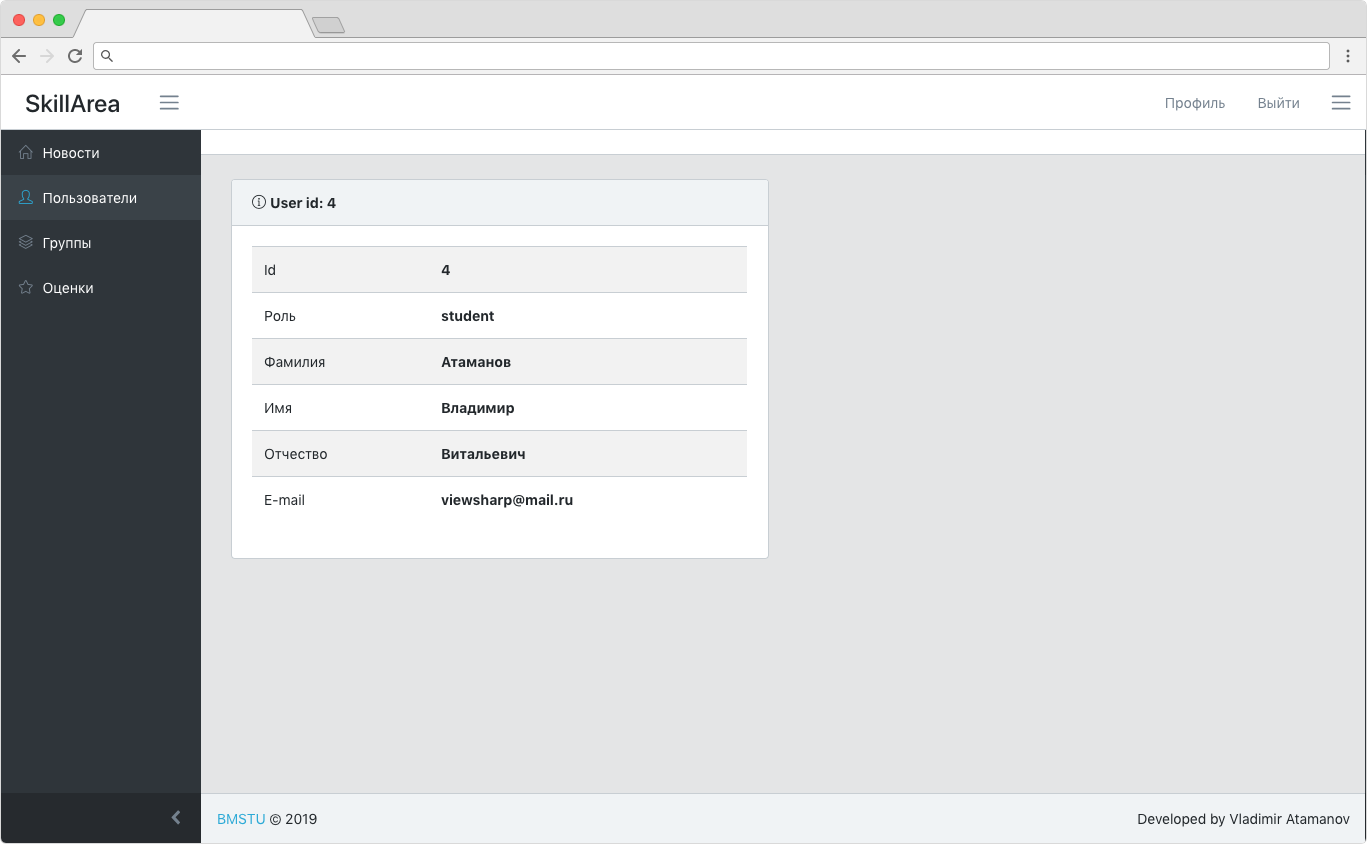


Рисунок 17 – страница пользователя

### 2.9.4 Интерфейс взаимодействия со студентом

Граф диалога со студентом показан на рисунке 18.

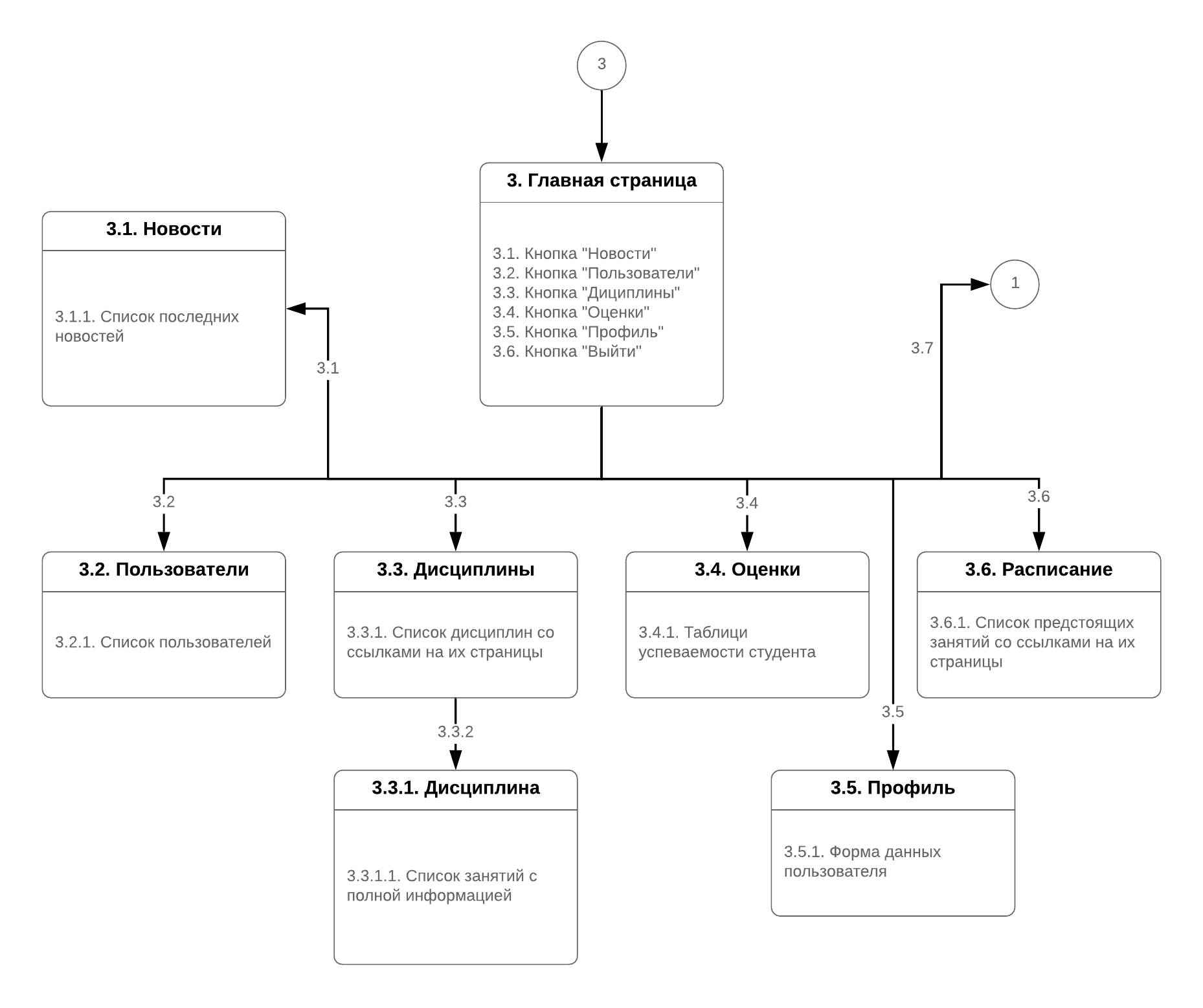


Рисунок 18 – граф диалога со студентом

Интерфейс страницы новостей показан на рисунке 19. На странице есть возможность просматривать последние новости.

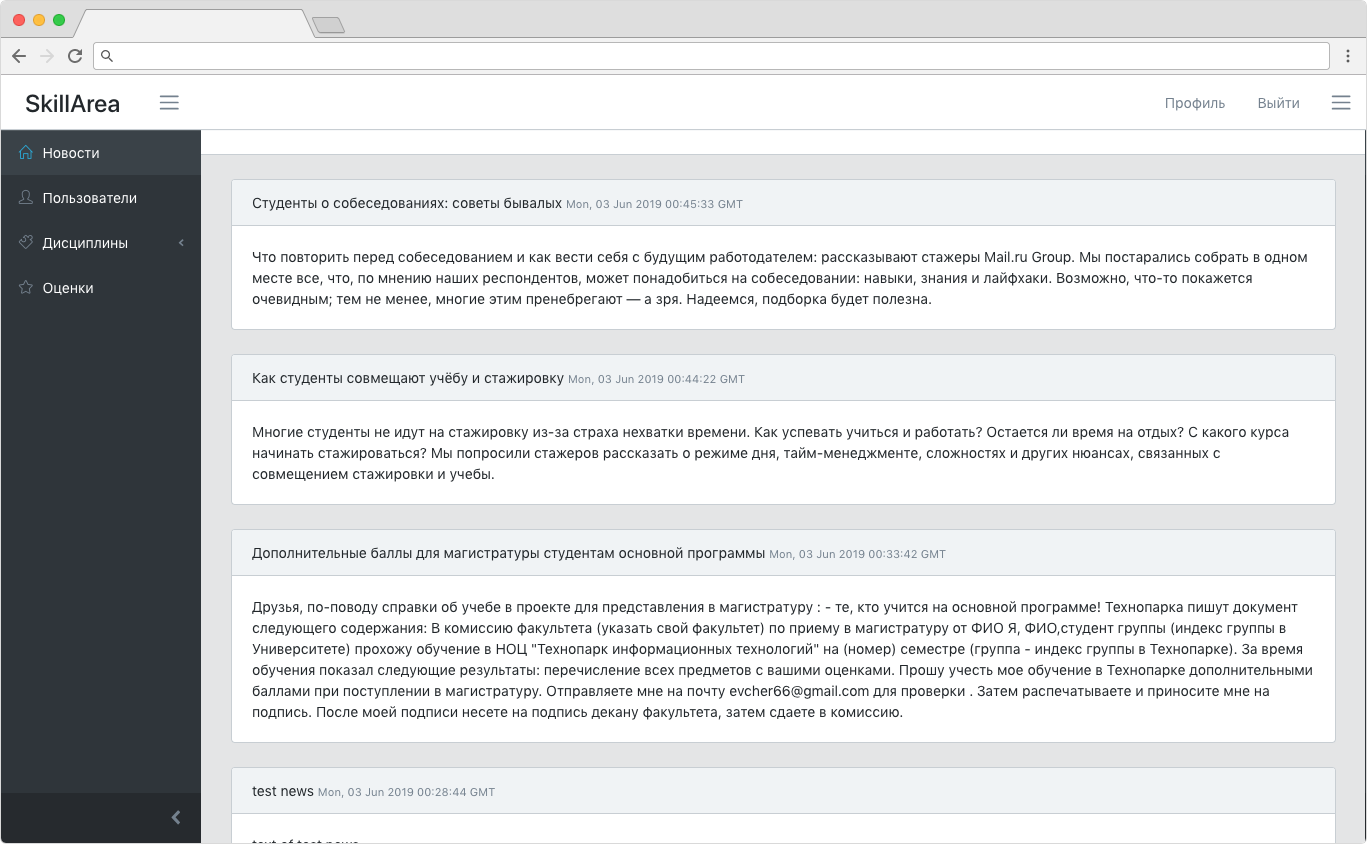
****

Рисунок 19 – страница новостей

Страница пользователей для студента выглядит точно также, как и для преподавателя и модератора, за исключением отсутствия ссылок на профили пользователей.

Интерфейс страницы новостей показан на рисунке 20. На странице есть возможность просматривать последние новости.

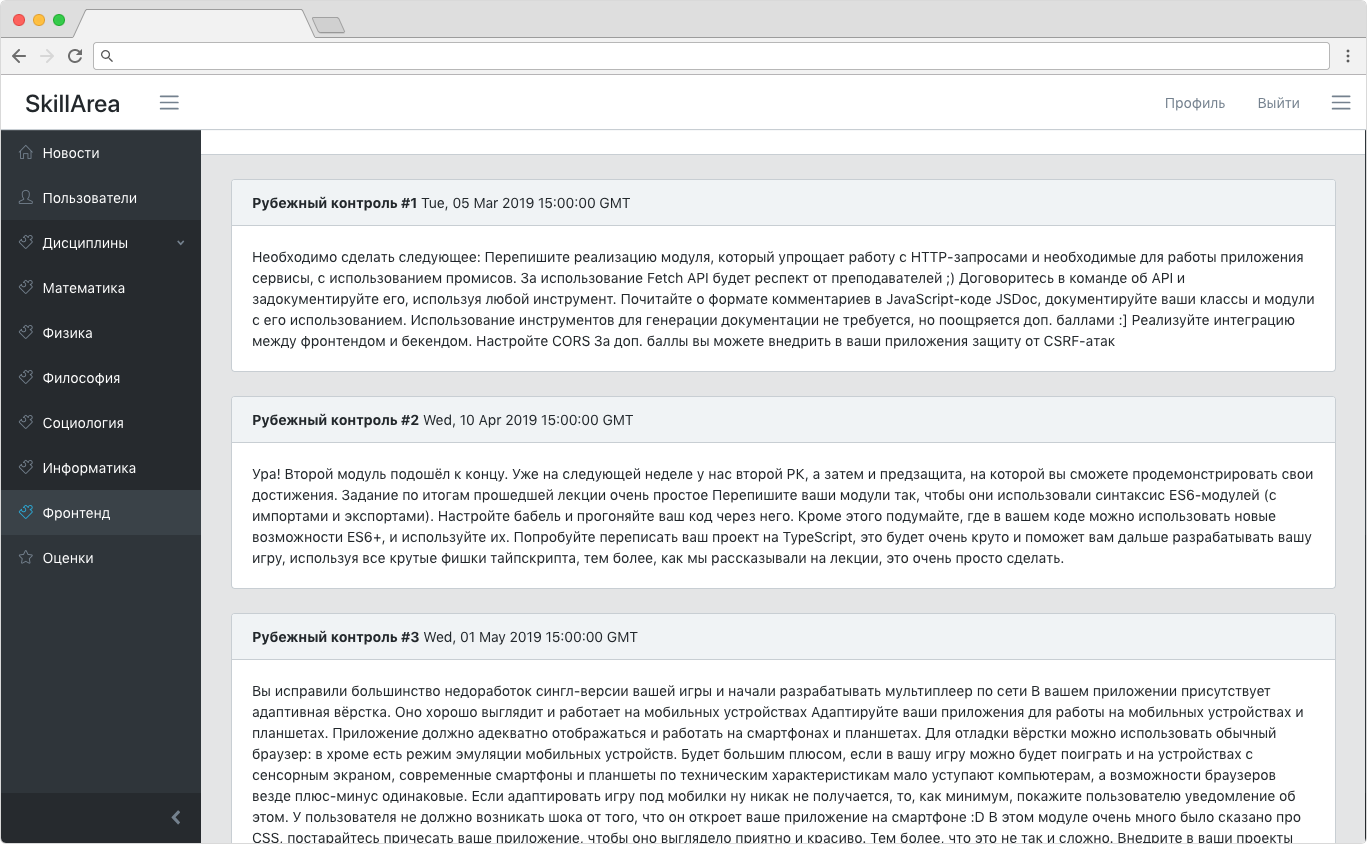


Рисунок 20 – страница дисциплины

Интерфейс страницы новостей показан на рисунке 21. На странице есть возможность просматривать оценки студента.

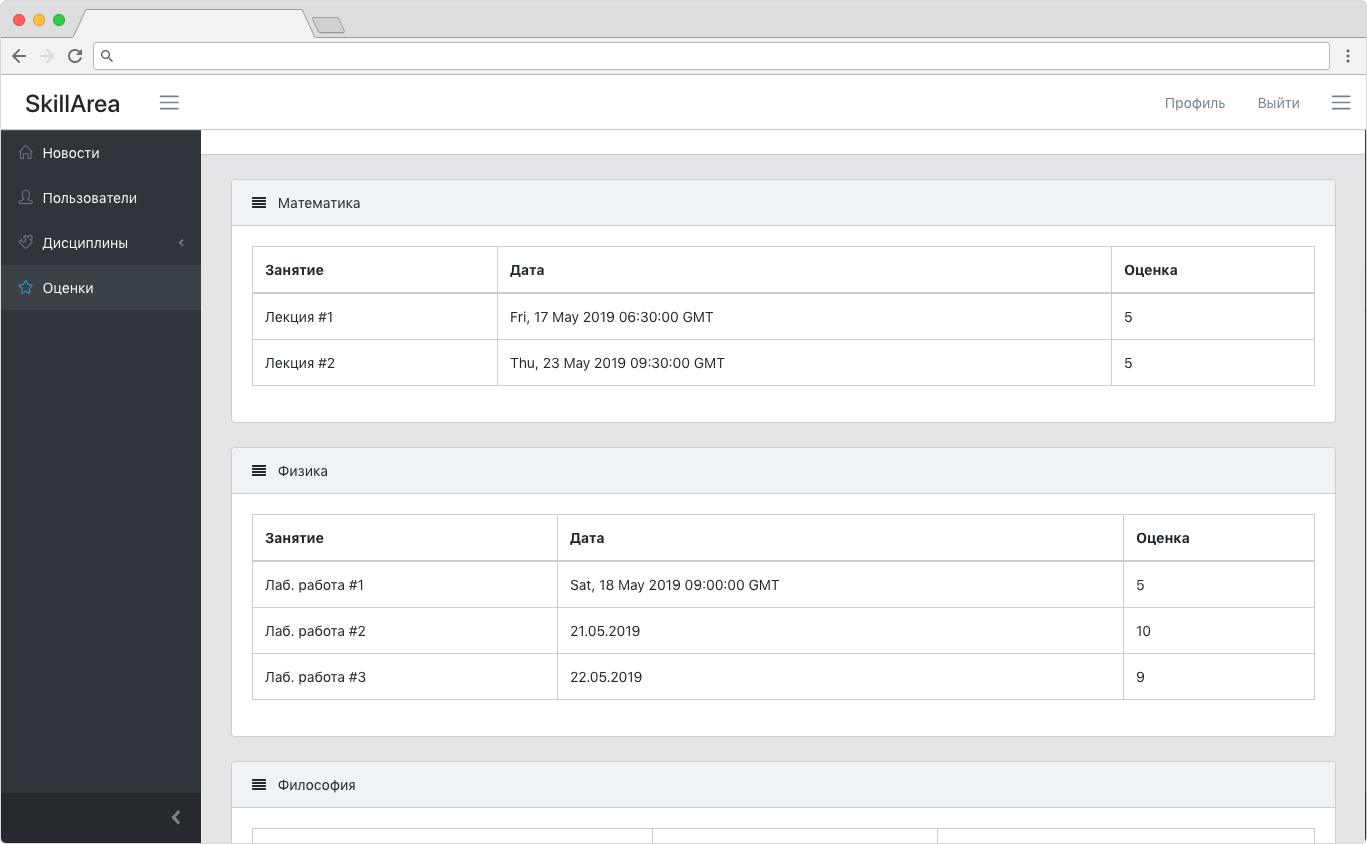


Рисунок 21 – страница дисциплины

Страница профиля для студента выглядит точно также, как и для преподавателя и модератора, но, как и преподаватель, он не имеет права менять свою роль

Интерфейс расписания показан на рисунке 22.

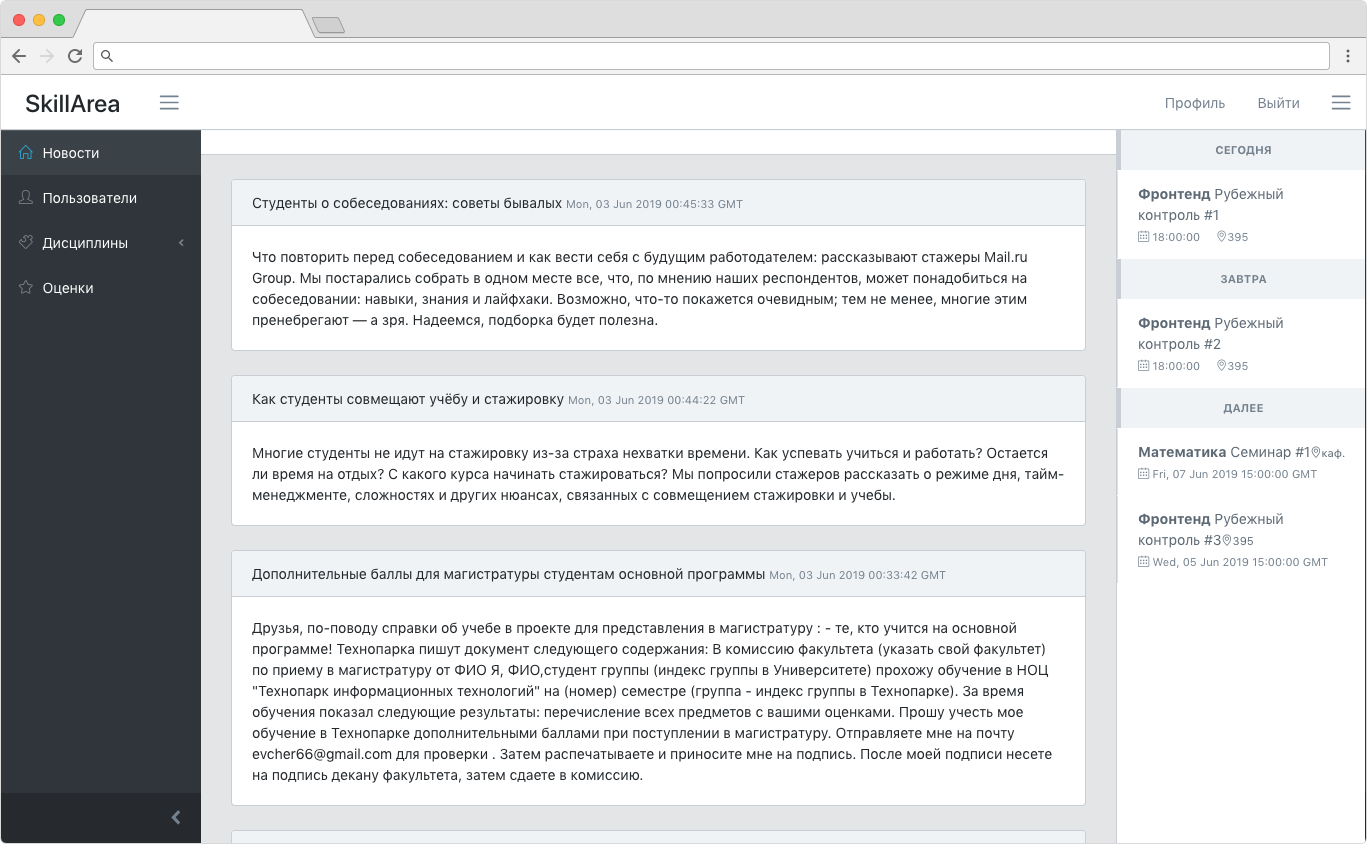
****

Рисунок 22 – расписание

# 3 Исследовательская часть. Выбор HTTP-сервера

Одним из наиболее важных элементов информационной системы является HTTP-сервер. Необходимо провести исследование и сравнить наиболее популярные варианты.

Сервера участвовавшие в исследовании:

1. **Nginx (В1).** Свободно распространяемый HTTP-сервер и обратный прокси-сервер, почтовый прокси-сервер, а также TCP/UDP прокси-сервер общего назначения.
2. **Apache (В2)**. Apache HTTP Server Project — это попытка разработать и поддерживать HTTP-сервер с открытым исходным кодом для современных операционных систем, включая UNIX и Windows. Цель этого проекта - предоставить безопасный, эффективный и расширяемый сервер, который предоставляет службы HTTP в синхронизации с текущими стандартами HTTP [12].
3. **HAProxy (В3).** HAProxy — это бесплатное, очень быстрое и надежное решение, обеспечивающее высокую доступность, балансировку нагрузки и прокси для приложений на базе TCP и HTTP. Он особенно подходит для веб-сайтов с очень высоким трафиком и поддерживает множество самых посещаемых сайтов в мире. За прошедшие годы он стал де-факто стандартным балансировщиком нагрузки с открытым исходным кодом, теперь поставляется с большинством распространенных дистрибутивов Linux и часто развертывается по умолчанию на облачных платформах [13].
4. **Lighttpd (В4).** Безопасность, скорость, соответствие и гибкость - все они описывают lighttpd (пр. Lighty), который быстро переопределяет эффективность веб-сервера; так как он разработан и оптимизирован для высокопроизводительных сред. Имея небольшой объем памяти по сравнению с другими веб-серверами, эффективное управление загрузкой процессора и расширенный набор функций (FastCGI, SCGI, Auth, сжатие вывода, перезапись URL-адресов и многие другие), lighttpd является идеальным решением для любого сервера [14].

Замеры метрик серверов проводилось на тестовом стенде со следующими характеристиками:

* количество ядер процессора: 2;
* частота процессора: 2,2 ГГц;
* объём оперативной памяти: 4 Гбайт;
* частота оперативно памяти: 1,6 ГГц.

Тестирование проводилось на операционной системе Ubuntu. Версии и даты выхода серверов, принимавших участие в тестировании:

* nginx 1.17.0 (21 мая 2019);
* apache HTTP Server 2.4.39 (27 марта 2019);
* HAProxy 1.8.10 (22 июня 2018);
* lighttpd 1.4.53 (27 января 2019).

Замеры использования процессора и оперативной памяти производились при нагрузке в 1000 запросов в секунду и 10 одновременных соединениях.

Тесты состояли из 2х типов запросов:

1. Запрос на отдачу статического файла весом 10 Кбайт;
2. Запрос, проектирующийся на сервер приложения, вес тела ответа на запрос составлял 1 Кбайт.

Критерии для сравнения вариантов приведены в таблице 25.

Таблица 25 – критерии для выбора HTTP-сервера.

|  |  |  |
| --- | --- | --- |
| Обозначение | Название | Описание |
| К1 | Среднее использование процессора, % | Средний процент использования процессора в тестах |
| К2 | Максимальное использование процессора, % | Процент максимального использования процессора в тестах |
| К3 | Среднее использование памяти, Мб | Средний объём используемой памяти в тестах |
| К4 | Максимальное использование памяти, Мб | Максимальный объём используемой памяти в тестах |
| К5 | Количество запросов в секунду | Среднее количество запросов в секунду, которое может обработать сервер |
| К6 | Поддержка HTTP/2 | Поддержка протокола HTTP/2. Протокол HTTP/2 является бинарным. По сравнению с предыдущим стандартом изменены способы разбиения данных на фрагменты и транспортирования их между сервером и клиентом. |

В таблице 26 приведены замеренные значения по заданным критериям.

Таблица 26 – значения вариантов по заданным критериям

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | В1 | В2 | В3 | В4 |
| К1 | 21,73 | 77,76 | 41,02 | 21,98 |
| К2 | 29,66 | 91,49 | 44,23 | 25,78 |
| К3 | 2,10 | 14,52 | 2,83 | 1,96 |
| К4 | 2,11 | 14,64 | 2,85 | 2,09 |
| К5 | 4213 | 2164 | 3547 | 4256 |
| К6 | + | + | + | - |

Для выбора лучшего варианта воспользуемся методами:

* метод анализа иерархии;
* метод максимума взвешенной суммы локальных критериев;
* метод ранжирования Борда;
* метод близости сравниваемых критериев к идеальному;
* метод гарантированного результата.

## 3.1 Метод максимума взвешенной суммы

По методу нахождения максимума взвешенной суммы нужно нормировать значения и рассчитать взвешенное значения для каждого варианта по формуле:

|  |  |
| --- | --- |
|  | (1) |

где αi – коэффициент критерия i;

kij – нормированное значение критерия i для варианта j.

Лучший вариант находится по формуле:

|  |  |
| --- | --- |
|  | (2) |

Рассчитанные значения приведены в таблице 27.

Таблица 27 – нормированные значения, коэффициенты и взвешенная сумма

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | В1 | В2 | В3 | В4 | α |
| К1 | 1 | 0,279 | 0,529 | 0,989 | 0,2 |
| К2 | 0,869 | 0,282 | 0,583 | 1 | 0,15 |
| К3 | 0,933 | 0,134 | 0,629 | 1 | 0,15 |
| К4 | 0,990 | 0,143 | 0,733 | 1 | 0,1 |
| К5 | 0,989 | 0,508 | 0,833 | 1 | 0,25 |
| К6 | 1 | 1 | 1 | 0 | 0,15 |
| Yj | 0,966 | 0,409 | 0,719 | 0,848 |  |

Предпочтительность вариантов согласно полученным значениям:

В1 > В4 > В3 > В2

По методу максимума взвешенной суммы лучшим является **вариант 1.**

## 3.2 Метод близости к идеалу

По методу близости сравниваемых критериев к идеалу нужно найти показатель близости по следующей формуле:

|  |  |
| --- | --- |
|  | (3) |

где αi – коэффициент критерия i;

kij – нормированное значение критерия i для варианта j.

Лучший вариант находится по формуле:

|  |  |
| --- | --- |
|  | (4) |

Рассчитанные значения приведены в таблице 28.

Таблица 28 – Нормированные значения, коэффициенты и показатель близости

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | В1 | В2 | В3 | В4 | α |
| К1 | 1 | 0,279 | 0,529 | 0,989 | 0,2 |
| К2 | 0,869 | 0,282 | 0,583 | 1 | 0,15 |
| К3 | 0,933 | 0,134 | 0,629 | 1 | 0,15 |
| К4 | 0,990 | 0,143 | 0,733 | 1 | 0,1 |
| К5 | 0,989 | 0,508 | 0,833 | 1 | 0,25 |
| К6 | 1 | 1 | 1 | 0 | 0,15 |
| Yj | 0,182 | 0,768 | 0,529 | 0,390 |  |

Предпочтительность вариантов согласно полученным значениям:

В1 > В4 > В3 > В2

По методу близости сравниваемых критериев к идеалу лучшим является **вариант 1.**

## 3.3 Метод ранжирования Борда

По методу ранжирования Борда каждому каждому критерия всех вариантов присваивается ранг [15]. После чего рассчитывается сумма рангов для каждого варианта по формуле:

|  |  |
| --- | --- |
|  | (5) |

где rij – ранг критерия i для варианта j.

Лучший вариант находится по формуле:

|  |  |
| --- | --- |
|  | (6) |

Рассчитанные значения приведены в таблице 29.

Таблица 29 – Ранги критериев

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | В1 | В2 | В3 | В4 |
| К1 | 1 | 4 | 2 | 1 |
| К2 | 2 | 4 | 3 | 1 |
| К3 | 1 | 4 | 2 | 1 |
| К4 | 1 | 5 | 2 | 1 |
| К5 | 1 | 3 | 2 | 1 |
| К6 | 1 | 1 | 1 | 2 |
| Yj | 7 | 21 | 12 | 7 |

Предпочтительность вариантов согласно полученным значениям:

В1 = В4 > В3 > В2

По методу близости сравниваемых критериев к идеалу лучшими является **вариант 1** и **вариант 4.**

## 3.4 Метод гарантированного результата

По методу гарантированного результата необходимо рассчитать значения для каждого варианта по следующей формуле:

|  |  |
| --- | --- |
|  | (7) |

где αi – коэффициент критерия i;

kij – нормированное значение критерия i для варианта j.

Лучший вариант находится по формуле:

|  |  |
| --- | --- |
|  | (8) |

Рассчитанные значения приведены в таблице 30.

Таблица 30 – Нормированные значения, коэффициенты и расчётные значения вариантов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | В1 | В2 | В3 | В4 | α |
| К1 | 1 | 0,279 | 0,529 | 0,989 | 0,2 |
| К2 | 0,869 | 0,282 | 0,583 | 1 | 0,15 |
| К3 | 0,933 | 0,134 | 0,629 | 1 | 0,15 |
| К4 | 0,990 | 0,143 | 0,733 | 1 | 0,1 |
| К5 | 0,989 | 0,508 | 0,833 | 1 | 0,25 |
| К6 | 1 | 1 | 1 | 0 | 0,15 |
| Yj | 0,1 | 0,0143 | 0,0733 | 0 |  |

Предпочтительность вариантов согласно полученным значениям:

В1 > В3 > В2 > В4

По методу близости сравниваемых критериев к идеалу лучшими является **вариант 1.**

**Вывод**: для выбора используемой в информационной системе «SkillArea» было произведено четыре метода выбора лучшего варианта HTTP-сервера. На основании того, что согласно всем методам, первое место занимает сервер Nginx был сделан вывод, что этот сервер действительно оптимальным и отвечающим всем необходимым требованиям проекта.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была разработана информационная система сопровождения профессионального образования, для чего были выполнены следующие задачи:

* поставлена задача разработки.
* исследована предметная область;
* разработана инфологической моделей данных
* разработана даталогической моделей данных;
* разработана структура системы;
* разработаны алгоритмы основных функций системы;
* разработан интерфейс системы;
* реализована программная часть;
* система протестирована;
* разработана документации к системе.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Федорова Г. H. Информационные системы: учебник для студ. учреждений сред. проф. образования / Г. Н. Федорова. — 3-е изд., стер. М.: Издательский центр «Академия», 2013.
2. Васильев А. Ю. Работа с PostgreSQL: настройка и масштабирование. — М.: Москва, 2017.
3. Моргунов Е. П. Язык SQL. Базовый курс. — М.: Москва, 2017.
4. Amy Brown and Greg Wilson. The architecture of an open source application. — lulu.com, 2011.
5. Айвалиотис Д.А. Администрирование сервера NGINX - М.: МК Пресс, 2013.: ил.
6. Флэнаган Д. JavaScript. Подробное руководство. СПб.: — Пер. с англ. — СПб: СимволПлюс, 2008.: ил.
7. Бэнкс Алекс, Порселло Ева. React и Redux: функциональная веб-разработка. — СПб.: Питер, 2018.: ил.
8. Марк Лутц. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с., ил.
9. Официальный сайт Django [Электронный ресурс] // djangoproject.com URL: https://www.djangoproject.com (дата обращения: 04.05.2019)
10. Дейт, К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом "Вильяме", 2005.: ил.
11. Диго С.М. БАЗЫ ДАННЫХ. ПРОЕКТИРОВАНИЕ И СОЗДАНИЕ: Учебнометодический комплекс. – М.: Изд. центр ЕАОИ. 2008.
12. Официальный сайт Apache [Электронный ресурс] // httpd.apache.org URL: https://httpd.apache.org (дата обращения: 09.05.2019)
13. Официальный сайт HAProxy [Электронный ресурс] // www.haproxy.org URL: http://www.haproxy.org (дата обращения: 09.05.2019)
14. Официальный сайт lighttpd [Электронный ресурс] // www.lighttpd.net URL: https://www.lighttpd.net (дата обращения: 09.05.2019)
15. Постников В. Основы эксплуатации автоматизированных систем обработки информации и управления. Краткий курс: учеб. пособие.

# ПРИЛОЖЕНИЕ А