МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
  
 к курсовой работе по дисциплине «Технологии сети интернет»

по теме: «Веб-приложение обучения по математике»

Обучающийся И.Г. Горбачев

Руководитель М.В. Додонов

Самара 2024

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc185859122)

[1 Описание и анализ предметной области 4](#_Toc185859123)

[1.1 Основные понятия и определения 4](#_Toc185859124)

[1.2 Описание систем-аналогов 5](#_Toc185859125)

[1.2.1 Веб-приложение «Math Playground» 5](#_Toc185859126)

[1.2.2 Веб-приложение «Khan Academy» 7](#_Toc185859127)

[1.2.3 Веб-приложение «Moodle» 8](#_Toc185859128)

[1.3 Постановка задачи 12](#_Toc185859129)

[2 Проектирование web-сайта 13](#_Toc185859130)

[2.1 Функциональная модель 13](#_Toc185859131)

[2.1.1 Входные и выходные данные 13](#_Toc185859132)

[2.1.2 Описание внешней информационной среды 13](#_Toc185859133)

[2.1.3 Перечень исключительных ситуаций 14](#_Toc185859134)

[2.1.4 Диаграмма вариантов использования 14](#_Toc185859135)

[2.2 Логическая схема данных 17](#_Toc185859136)

[2.2.1 Перечень таблиц и атрибутов 18](#_Toc185859137)

[2.3 Схемы алгоритмов обработки данных 19](#_Toc185859138)

[2.3.1 Алгоритм 1 20](#_Toc185859139)

[2.3.2 Алгоритм 2 20](#_Toc185859140)

[2.4 Прототип пользовательского интерфейса 22](#_Toc185859141)

[3 Реализация web-сайта 25](#_Toc185859142)

[3.1 Разработка физической модели БД 25](#_Toc185859143)

[3.2 Разработка серверной части 26](#_Toc185859144)

[3.3 Разработка клиентской части 27](#_Toc185859145)

[3.4 Тестирование сайта 28](#_Toc185859146)

[ЗАКЛЮЧЕНИЕ 32](#_Toc185859147)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc185859148)

[ПРИЛОЖЕНИЕ А Листинг программы 35](#_Toc185859149)

ВВЕДЕНИЕ

В современном мире интернет играет ключевую роль в упрощении и ускорении многих аспектов нашей жизни, включая образование. Одной из быстрорастущих областей является онлайн-обучение, где веб-приложения становятся удобным и востребованным инструментом для изучения различных предметов, включая математику.

Веб-приложение для обучения математике – это цифровая платформа, позволяющая пользователям изучать математические концепции, решать задачи и улучшать свои навыки в интерактивной форме. Оно предоставляет пользователям доступ к разнообразным учебным материалам, таким как теоретические уроки, видеоролики, практические задания и тесты. Основной задачей такого приложения является создание удобной и эффективной среды для самостоятельного или сопровождённого обучения.

Пользователи могут выбирать темы, соответствующие их уровню знаний, следить за своим прогрессом и получать мгновенную обратную связь. Веб-приложение для обучения математике обеспечивает доступность образовательного контента в любое время и из любой точки мира, что делает процесс обучения гибким и удобным. Для преподавателей такие приложения становятся мощным инструментом для создания курсов, отслеживания успеваемости учеников и персонализации образовательного процесса. Для учеников это возможность углубить свои знания и улучшить навыки в области математики через современные и интерактивные подходы.

1. Описание и анализ предметной области

Предметная область ­– это область знаний, которая изучает, описывает и анализирует определенный сегмент реальности или явление, фокусируясь на определенных объектах, процессах или проблемах. Она определяет предмет и структуру исследования, определяет основные понятия и термины, а также ограничения и рамки изучаемой области [2].

* 1. Основные понятия и определения

Веб-приложение – это цифровая платформа, представленная в виде веб-сайта, которая предоставляет пользователям доступ к интерактивным функциям и сервисам через интернет. Основой веб-приложения является пользовательский интерфейс, который позволяет взаимодействовать с различными функциями, такими как ввод данных, выполнение вычислений, отображение информации или управление ресурсами. Все процессы, включая обработку запросов, хранение данных и обеспечение безопасности, происходят на сервере или с использованием облачных технологий.

База данных – это упорядоченная совокупность данных, которая организована и хранится в компьютерной системе для удобного управления и обработки. Она представляет собой логически связанную структуру, предназначенную для эффективного хранения, организации и доступа к информации с использованием специализированного программного обеспечения. База данных позволяет выполнять запросы, обновлять данные и управлять ими, обеспечивая структурированный подход к работе с текстовой, числовой, графической, аудиовизуальной и другой информацией. Такой инструмент широко применяется для обработки больших объемов данных и их упорядоченного использования в различных сферах. [4].

Объектный анализ:

* учитель – объект, представляющий учителя в системе. Может иметь свойства, такие как имя, электронная почта и пароль;
* ученик – это объект, представляющий ученика в системе. Может иметь свойства, такие как имя, электронная почта и пароль;
* учебный материал – это объект, представляющий собой материал для обучения в виде текста, файла или видеоурока;
* домашнее задание – это объект, представляющий собой материал для выполнения заданий учеником в качестве домашнего задания.
  1. Описание систем-аналогов

Перед созданием интернет-магазина необходимо провести сравнительный анализ с аналогичными системами. Этот анализ позволит выявить преимущества и недостатки конкурирующих решений, а также определить, какие функциональные возможности можно дополнить или усовершенствовать по сравнению с уже существующими.

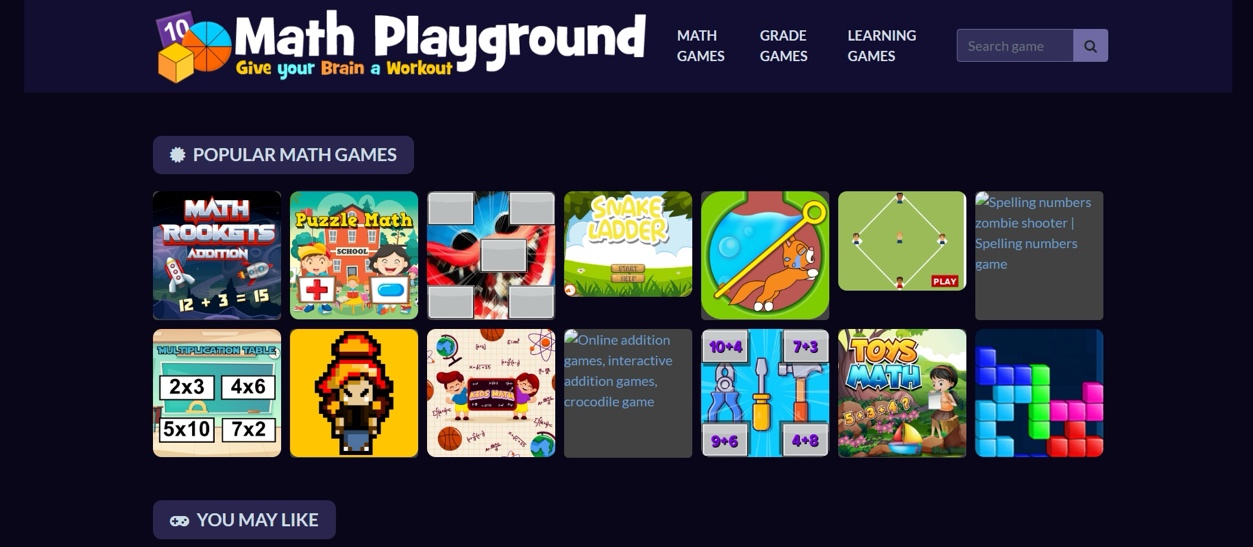
* + 1. Веб-приложение «Math Playground»

«Math Playground» – это специализированная платформа, предназначенная для изучения математики через интерактивные игры, упражнения и обучающие ресурсы. Сервис предоставляет пользователям широкий выбор математических задач и активностей, охватывающих темы от арифметики до геометрии, предназначенные как для самостоятельного обучения, так и для использования в образовательных учреждениях. [5]

Пользователи могут выбирать задания в зависимости от своего уровня подготовки, возраста и интересов. Для удобства навигации платформа предлагает разделы с классификацией по темам, уровню сложности и типу задач. Дополнительно доступны инструменты для отслеживания прогресса, включая подсчёт набранных баллов, выполнение тестов и прохождение уровней.

Платформа предоставляет гибкие возможности взаимодействия: ученики могут решать задачи и получать мгновенную обратную связь, учителя – назначать задания, создавать учебные планы и отслеживать успехи учеников через личный кабинет.

На рисунке 1 продемонстрирована главная страница на сайте «Math Playground».

  
Рисунок 1 – Главная страница на сайте «Math Playground»

К преимуществам платформы «Math Playground» можно отнести:

* разнообразие интерактивных заданий, игр и упражнений для разных возрастов и уровней подготовки;
* доступность учебных материалов в любое время и с любого устройства;
* возможность отслеживать прогресс и получать мгновенную обратную связь;
* наличие разделов для самостоятельного обучения и использования в образовательных учреждениях;
* интуитивно понятный интерфейс, подходящий для детей.

К недостаткам можно отнести:

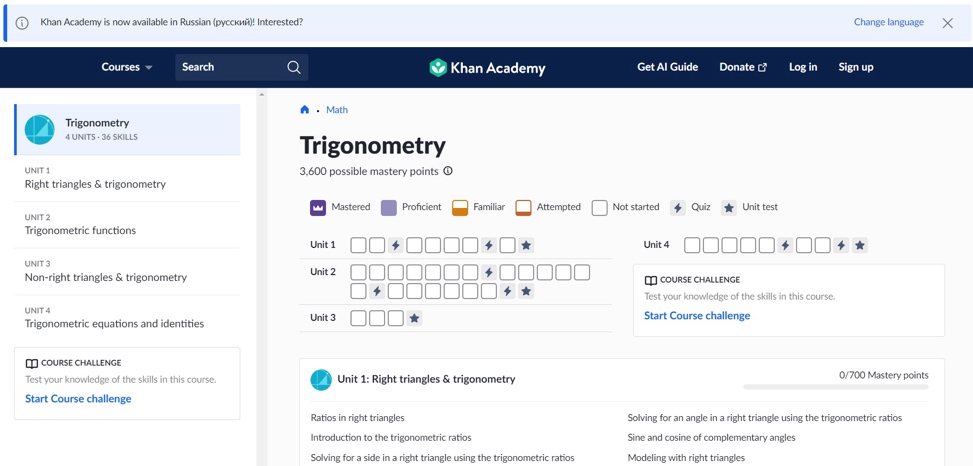
* отсутствие возможности работать без подключения к интернету;
* ограниченный доступ к некоторым функциям и материалам без платной подписки.
  + 1. Веб-приложение «Khan Academy»

«Khan Academy» – это популярный сайт и мобильное приложение, ориентированные на предоставление образовательных материалов по различным предметам, включая математику, естественные науки, историю и искусство. Сервис позволяет пользователям бесплатно изучать курсы, просматривать видеоуроки, выполнять задания и тесты, а также отслеживать свои достижения. [6]

Пользователи могут выбирать подходящие курсы, охватывающие широкий спектр тем, и изучать их в удобном темпе. Для этого платформа предлагает структурированные уроки, пошаговые объяснения, интерактивные задания и встроенные инструменты, такие как калькулятор или графический редактор. Разделы с персонализированными рекомендациями помогают направить обучающегося на наиболее актуальные темы.

Платформа также предоставляет преподавателям возможность создавать учебные планы, отслеживать успеваемость учеников и назначать им задания. Ученики, в свою очередь, могут улучшать свои знания и навыки через доступный и интуитивно понятный интерфейс, что делает обучение эффективным и увлекательным.

На рисунке 2 представлена главная страница на сайте «Khan Academy».

  
Рисунок 2 – Главная страница на сайте «Khan Academy»

К преимуществам системы «Khan Academy» можно отнести:

* доступ к широкому спектру образовательных материалов по различным предметам;
* бесплатное использование всех основных функций платформы;
* персонализированные рекомендации для улучшения процесса обучения;
* интерактивные задания с мгновенной обратной связью;
* возможность отслеживания прогресса и создания индивидуального плана обучения;
* доступность с любых устройств, включая компьютеры, планшеты и смартфоны.

К недостаткам можно отнести:

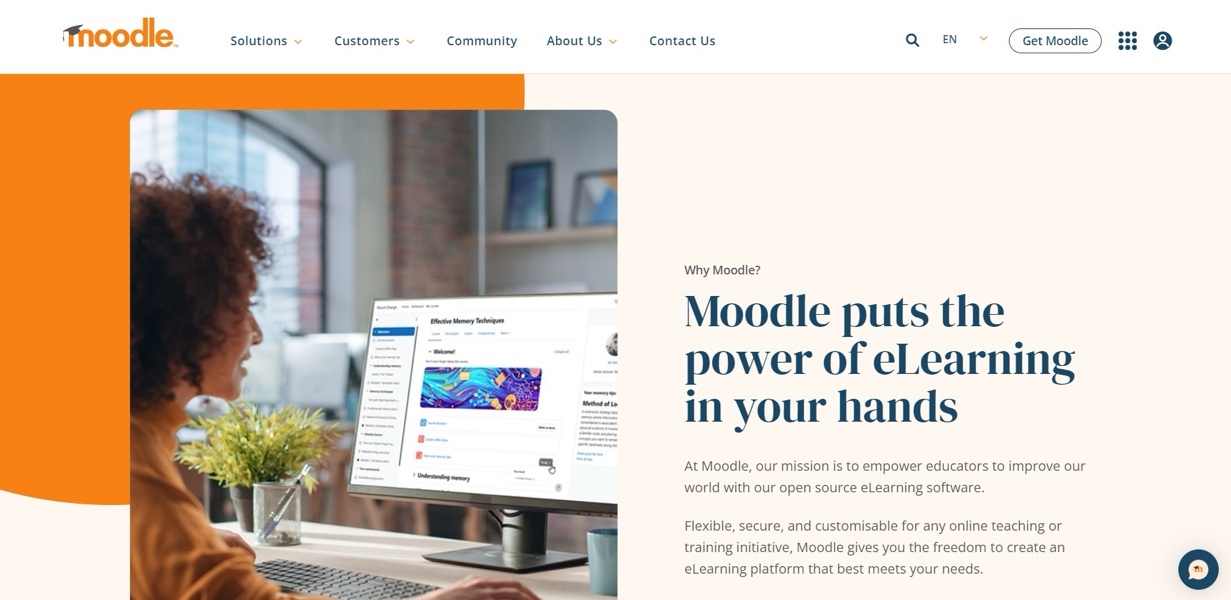
* отсутствие оффлайн-доступа к материалам без предварительного скачивания;
* необходимость базового уровня владения языком, на котором представлен контент, для его полного понимания;
* ограниченные возможности для углублённого изучения некоторых специализированных тем.
  + 1. Веб-приложение «Moodle»

«Moodle» – это популярная платформа предназначенное для управления образовательным процессом и создания интерактивных онлайн-курсов. Сервис объединяет преподавателей и студентов, предоставляя удобные инструменты для организации, прохождения и мониторинга учебных программ. [7]

На платформе «Moodle» пользователи могут создавать курсы, загружать учебные материалы, добавлять задания, тесты и форумы для обсуждений. Преподаватели могут детализировать содержание курса, включая лекции, презентации, видеоматериалы и дополнительные ресурсы. Для студентов предусмотрены удобные инструменты для выполнения заданий, прохождения тестов и отслеживания своего прогресса.

Сервис поддерживает функции для взаимодействия в реальном времени, такие как чаты и видеоконференции. Преподаватели также могут использовать дополнительные модули и плагины для расширения функционала курсов, включая автоматическую оценку, индивидуальные настройки доступа и интеграцию с внешними системами.

На рисунке 3 представлена главная страница на сайте «Moodle».

  
Рисунок 3 – Главная страница на сайте «Moodle»

К преимуществам системы «Moodle» можно отнести:

* возможность создания индивидуальных курсов с разнообразными образовательными материалами;
* поддержка широкого спектра инструментов для взаимодействия, включая форумы, чаты и видеоконференции;
* гибкость в настройке доступа и структуры курсов под разные образовательные цели;
* доступность системы с любых устройств через веб-интерфейс или мобильное приложение;
* открытый исходный код, что позволяет адаптировать платформу под конкретные нужды образовательных учреждений.

К недостаткам можно отнести:

* необходимость технической поддержки для настройки и администрирования платформы;
* потенциальная сложность интерфейса для начинающих пользователей;
* ограниченный функционал без подключения сторонних плагинов и модулей.

Таблица 1 – Сравнительные характеристики систем-аналогов

|  |  |  |  |
| --- | --- | --- | --- |
| Название системы  Название показателя | Math Playground | Khan Academy | Moodle |
| Связь репетитора и ученика | - | - | + |
| Возможность добавления материалов | + | + | + |
| Создание индивидуальных тестов | - | + | + |
| Автоматическая проверка тестов | - | + | + |
| Личный кабинет учащегося | - | + | + |
| Личный кабинет репетитора | - | + | + |
| Простота использования | + | + | - |
| Доступность и бесплатность | + | + | + |

* 1. Постановка задачи

Во время курсового проектирования необходимо разработать веб-приложение обучения математике, которое позволяет пользователям решать задания, смотреть каталог заданий и личный профиль, отправлять задания на проверку и записываться на занятия.

Требования к информационному, техническому и программному обеспечению:

* клиентская часть должна предоставлять удобный интерфейс;
* клиентская часть сайта должна быть разработана на JavaScript;
* серверная часть сайта должна быть разработана на Express.js;
* для хранения данных должна использоваться реляционная база данных;
* в качестве СУБД должна использоваться MySQL;
* для стилизации пользовательского интерфейса можно использовать CSS.

1. Проектирование web-сайта

На этапе проектирования системы нужно решить ряд задач:

* определить требования к системе;
* создать структуру и архитектуру системы;
* разработать пользовательский интерфейс;
* выбрать технологию разработки;
* выбрать платформу разработки.
  1. Функциональная модель

Функциональная модель – это структурированное представление функций в смоделированной системе или предметной области [8].

* + 1. Входные и выходные данные

Входные данные:

* электронная почта;
* пароль;

Выходные данные:

* задания от учителя;
* материалы для обучения;
* данные об ошибках.
  + 1. Описание внешней информационной среды

Внешняя информационная среда включает в себя следующие компоненты:

* интернет-подключение: интернет-магазин требует стабильного подключения к Интернету для обеспечения быстрой передачи данных между пользовательскими устройствами и облачным сервером;
* устройства доступа: различные устройства, такие как компьютеры, ноутбуки, смартфоны и планшеты, на которых пользователи могут получить доступ к интернет-магазину с помощью различных операционных систем;
* браузеры: пользователи могут обращаться к интернет-магазину через веб-браузеры;
* протоколы и стандарты: интернет-магазин может использовать различные протоколы для передачи данных, такие как HTTP, FTP и протоколы, обеспечивающие безопасное шифрование и аутентификацию;
  + 1. Перечень исключительных ситуаций

Исключительная ситуация – это ситуация, при которой система не может выполнить возложенных на нее функций или которая может привести к денормализации работы системы.

В таблице 2 приведен перечень исключительных ситуаций для разрабатываемой системы и описаны реакции системы на их возникновение.

Таблица 2 – Перечень исключительных ситуаций

|  |  |  |
| --- | --- | --- |
| Название  подсистемы | Название исключительной ситуации | Реакция системы |
| 1 Регистрация и авторизация | 1.1 Некорректный логин | Выдача сообщения «Некорректный email» |
| 1.2 Некорректный пароль | Выдача сообщения «Некорректный пароль» |
| 2 Взаимодействие с фильтрами | 3.3 Невозможно найти подходящее задание | Выдача сообщения «Неверные фильтры» |
| 3.4 Невозможно загрузить файл из-за ошибки на сервере | Выдача сообщения «Ошибка сервера» |

* + 1. Диаграмма вариантов использования

Диаграмма вариантов использования представляет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм. На ней изображаются отношения между актерами и вариантами использования [9].

На рисунке 4 приведена диаграмма вариантов использования (пользователя). На диаграмме представлен пользователь, который может взаимодействовать с системой следующим образом:

* «войти в систему» – пользователь может войти в систему. Для этого ему необходимо создать свой профиль в процессе регистрации и успешно пройти авторизацию;
* «открыть расписание» – пользователь может просмотреть расписание занятий, на которые он записался или отменить занятие;
* «открыть задание» – пользователь может открыть задание, а также отправить задание на проверку учителю;
* «открыть страницу профиля» – пользователь может открыть страницу своего профиля, где ему будет доступна возможность изменения личных данных;
* «открыть учебник» – пользователь может открыть страницу с учебником, где он может изучить необходимый материал;

  
Рисунок 4 – Диаграмма вариантов использования

* 1. Логическая схема данных

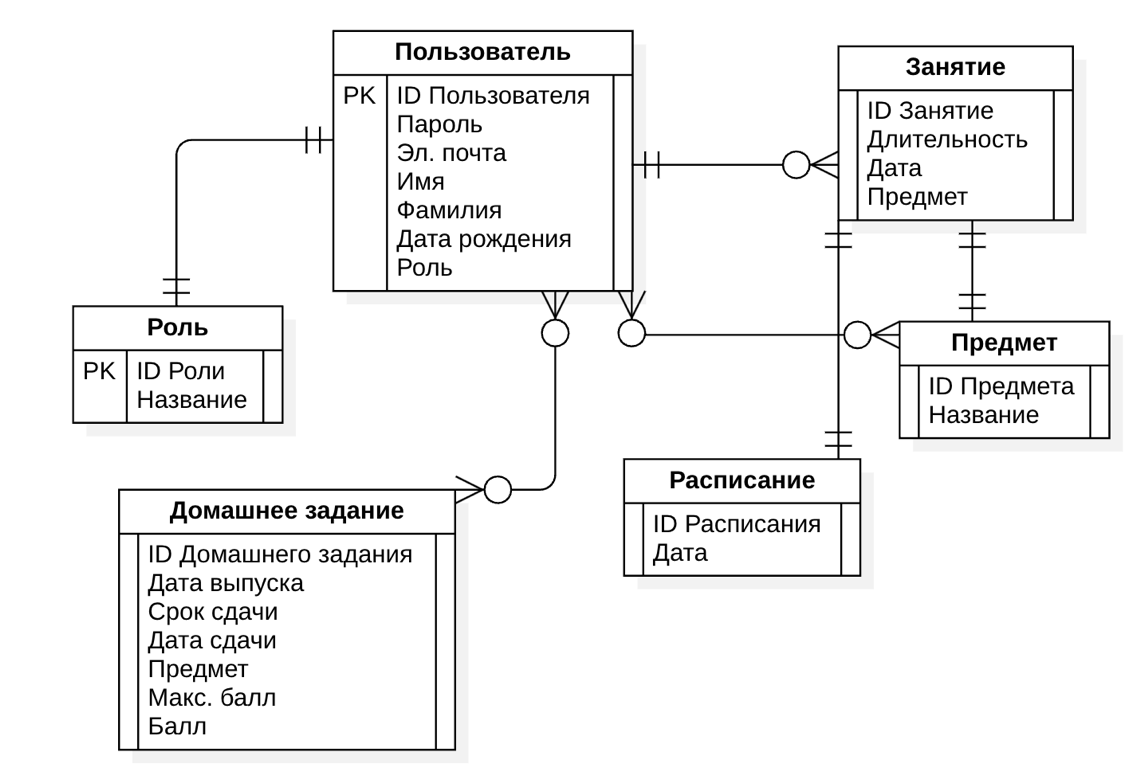
Логическая информационная модель – модель данных, в которой учитывается способ логического хранения данных в памяти ЭВМ. При построении модели базы данных (БД) используются следующие понятия.

Сущность – объект предметной области, который можно отличить от других понятий по некоторым признакам. Сущность состоит из множества своих экземпляров. Каждая сущность обладает свойствами – атрибутами.

Атрибут – определенное свойство сущности. Именно набор атрибутов, в общем случае уникальный для каждой сущности, позволяет выделить ее среди других объектов и назвать уникальным именем.

Атрибут или набор атрибутов, используемый для идентификации экземпляра сущности, называется ключом сущности. В случае если для идентификации экземпляра используется один атрибут, ключ называется простым; в противном случае ключ составной. Каждый экземпляр сущности однозначно определяется ключом [10].

Логическая модель БД разрабатываемой системы приведена на рисунке 5.

  
Рисунок 5 – Логическая модель данных

* + 1. Перечень таблиц и атрибутов

Описание объектов рассматриваемой предметной области, которые хранятся в базе данных, приведено в таблицах 3-8.

Таблица 3 – Сущность «Пользователь»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| ID пользователя | Целый | Уникальный идентификатор пользователя |
| Пароль | Символьный [10] | Пароль пользователя, преобразованный в захешированную строку |
| Эл. почта | Символьный [50] | Электронная почта, указанная пользователем при регистрации |
| Имя | Символьный [10] | Имя пользователя |
| Фамилия | Символьный [12] | Фамилия пользователя |
| Дата рождения | Дата | Дата рождения пользователя |
| Роль | Символьный [6] | Роль пользователя в системе |

Таблица 4 – Сущность «Домашнее задание»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| ID Домашнего задания | Целый | Уникальный идентификатор домашнего задания |
| Дата выпуска | Дата | Дата, когда задание появилось в системе |
| Срок сдачи | Дата | Дата, до которой нужно сдать задание |
| Дата сдачи | Дата | Дата, когда задание было сдано |
| Предмет | Символьный [10] | Название предмета |
| Макс. балл | Целый | Максимальный балл за задание |
| Балл | Целый | Полученный балл |

Таблица 5 – Сущность «Роль»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| ID Роли | Целый | Уникальный идентификатор роли |
| Название | Символьный [10] | Название роли |

Таблица 6 – Сущность «Расписание»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| ID Расписания | Целый | Уникальный идентификатор расписания |
| Дата | Дата | Дата расписания |

Таблица 7 – Сущность «Предмет»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| ID Предмет | Целый | Уникальный идентификатор предмета |
| Название | Символьный [10] | Название предмета |

Таблица 8 – Сущность «Занятие»

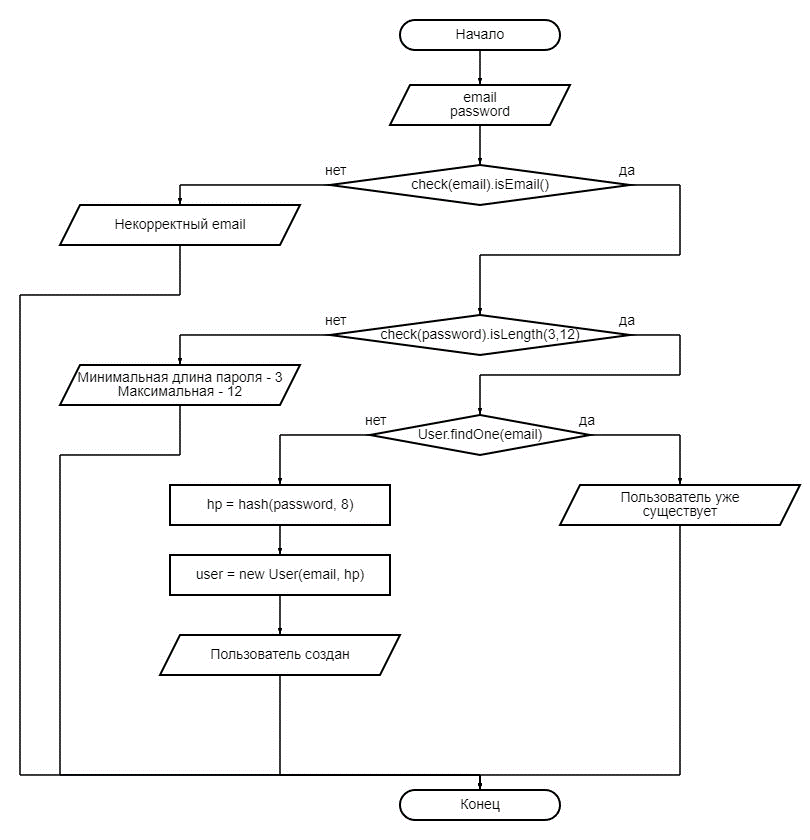
|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| ID Занятия | Целый | Уникальный идентификатор занятия |
| Длительность | Дата | Длительность занятия |
| Дата | Дата | Дата занятия |
| Предмет | Символьный [10] | Название предмета |

* 1. Схемы алгоритмов обработки данных

Алгоритмом обработки данных называют метод решения задачи, который возможно реализовать в выбранной среде программирования. Тщательная разработка алгоритма является весьма эффективной частью процесса решения задачи в любой области применения [11].

* + 1. Алгоритм 1

На рисунке 6 приведена схема алгоритма регистрации пользователя

  
Рисунок 6 – Схема алгоритма регистрации пользователя

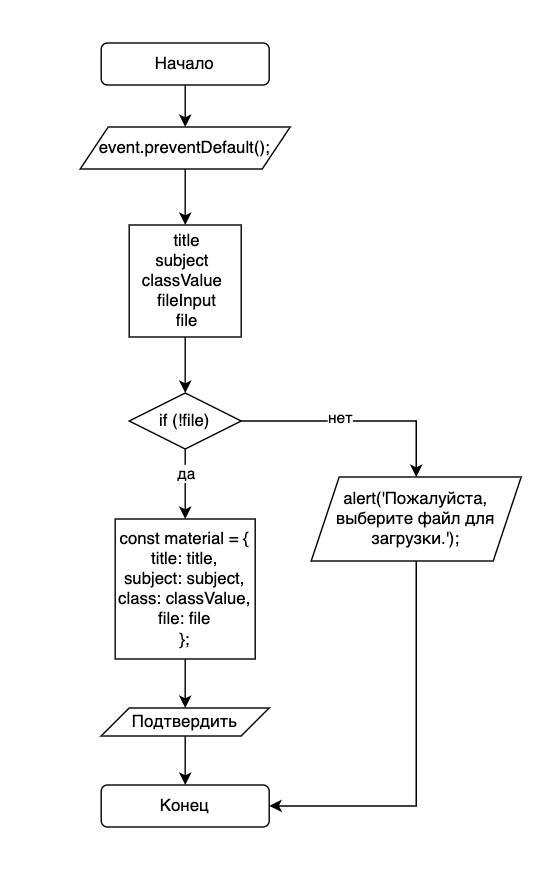
Описание алгоритма.

Пользователь вводит логин и пароль. Система осуществляет проверку корректности введенных данных. В случае ошибки система выводит сообщение об ошибке.

Если введенные данные корректны, то они отправляются на сервер, где осуществляется проверка уникальности электронной почты. Если введенная почта находится в БД, то выводится сообщение, что пользователь уже существует. Когда введенная почта уникальна, то система хеширует пароль, создаёт аккаунт и выводит сообщение, что пользователь создан.

* + 1. Алгоритм 2

На рисунке 7 приведена схема алгоритма загрузки файла в хранилище.

  
Рисунок 7 – Схема алгоритма добавления учебных материалов

Описание алгоритма.

Алгоритм отправки формы заявления начинается с обработки события отправки. Стандартное действие браузера по умолчанию отменяется, чтобы предотвратить нежелательную перезагрузку страницы. Далее собираются данные из полей формы, включая название, класс, предмет и приложенный файл. Все введённые данные проходят проверку на корректность. Если данные не соответствуют установленным требованиям, процесс прерывается, и пользователю предлагается внести исправления.

После успешной проверки формируется итоговая информация, содержащая ключевые детали: название, предмет, класс и файл. Этот текст добавляется в специальный блок на странице для отображения списка учебных материалов.

* 1. Прототип пользовательского интерфейса

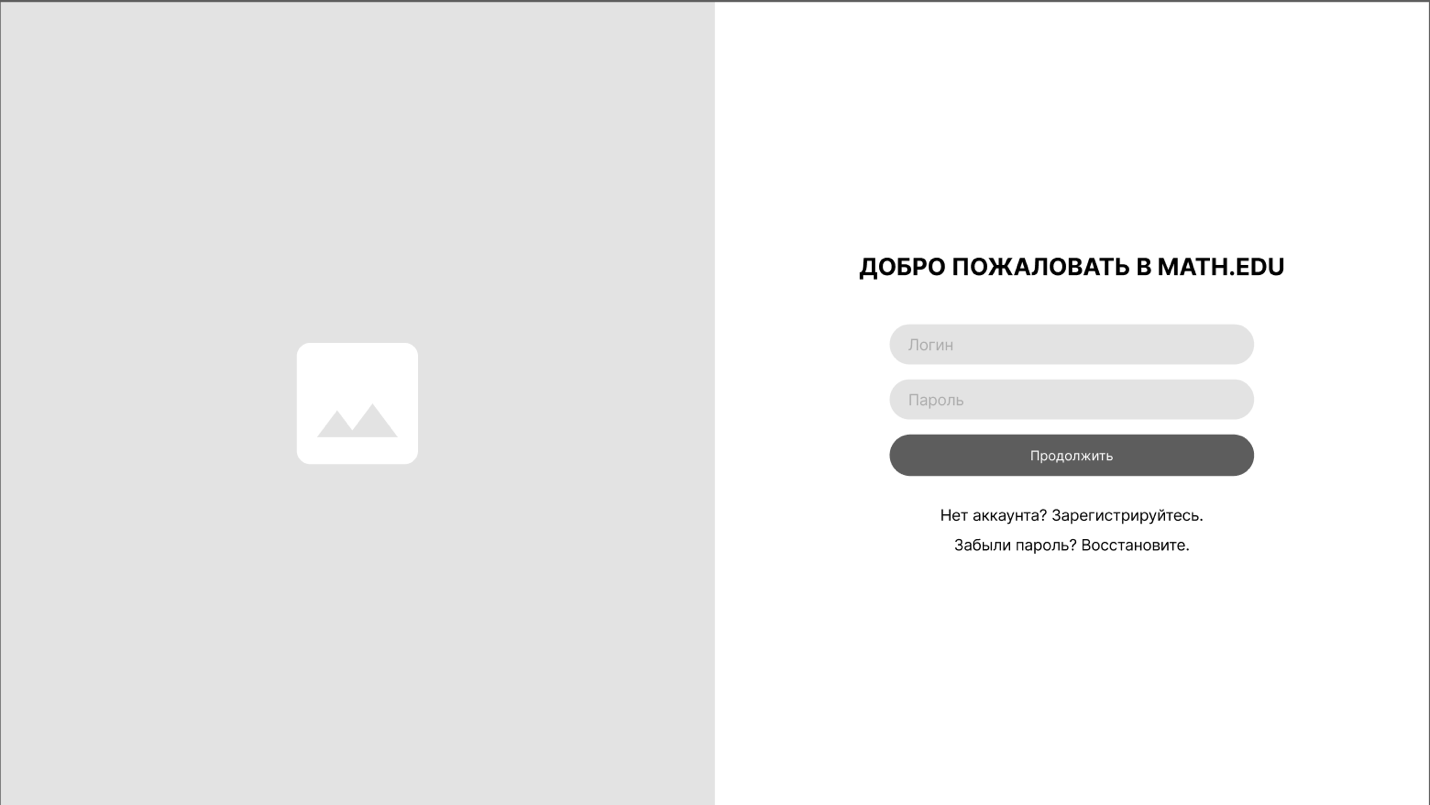
Интерфейс – это совокупность средств и механизмов, обеспечивающих взаимодействие пользователя с программой или устройством. Он представляет собой связующее звено между человеком и технической системой, предоставляя набор инструментов, элементов управления и визуальных компонентов, которые упрощают работу с программным обеспечением. Интерфейс может быть графическим, текстовым или голосовым, а его основная задача – сделать использование программы интуитивно понятным, эффективным и комфортным для пользователя. Хорошо продуманный интерфейс позволяет минимизировать количество ошибок, ускорить выполнение задач и обеспечить положительный пользовательский опыт.

Прототипирование помогает выявить потенциальные проблемы или недостатки интерфейса, что дает возможность внести корректировки на ранних стадиях разработки [12].

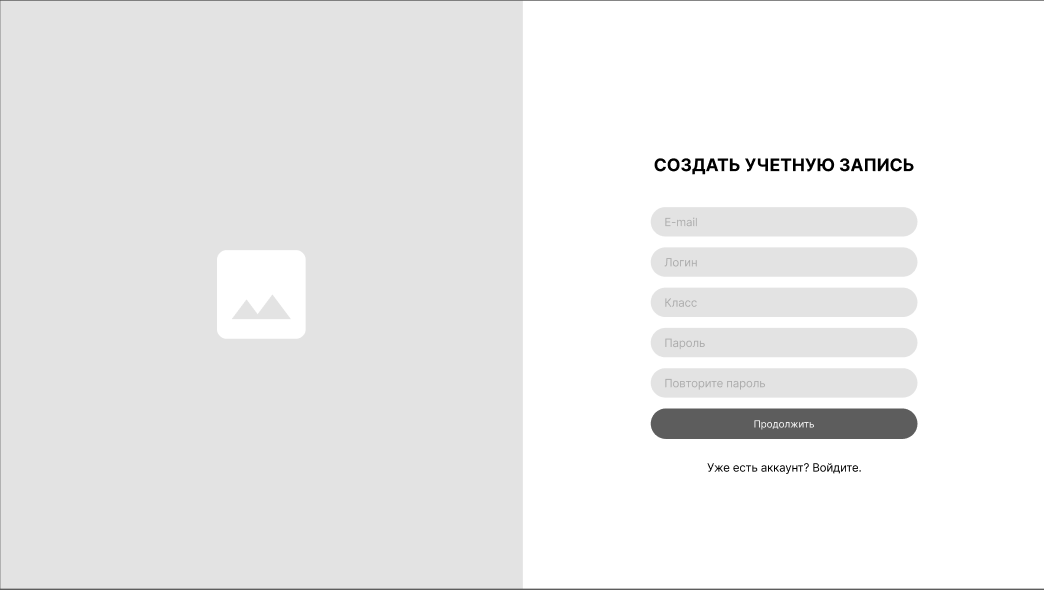
Основными требованиями к интерфейсу являются:

* функциональность: набор возможностей, операций и действий, предоставляемых пользователю или другим системам через данный интерфейс;
* понятность: интуитивно понятный интерфейс.

На рисунке 8 приведен прототип формы авторизации. При запуске приложения будет открываться форма авторизации. Здесь пользователь сможет ввести адрес электронной почты и пароль для входа в систему.

  
Рисунок 8 – Прототип экранной формы авторизации

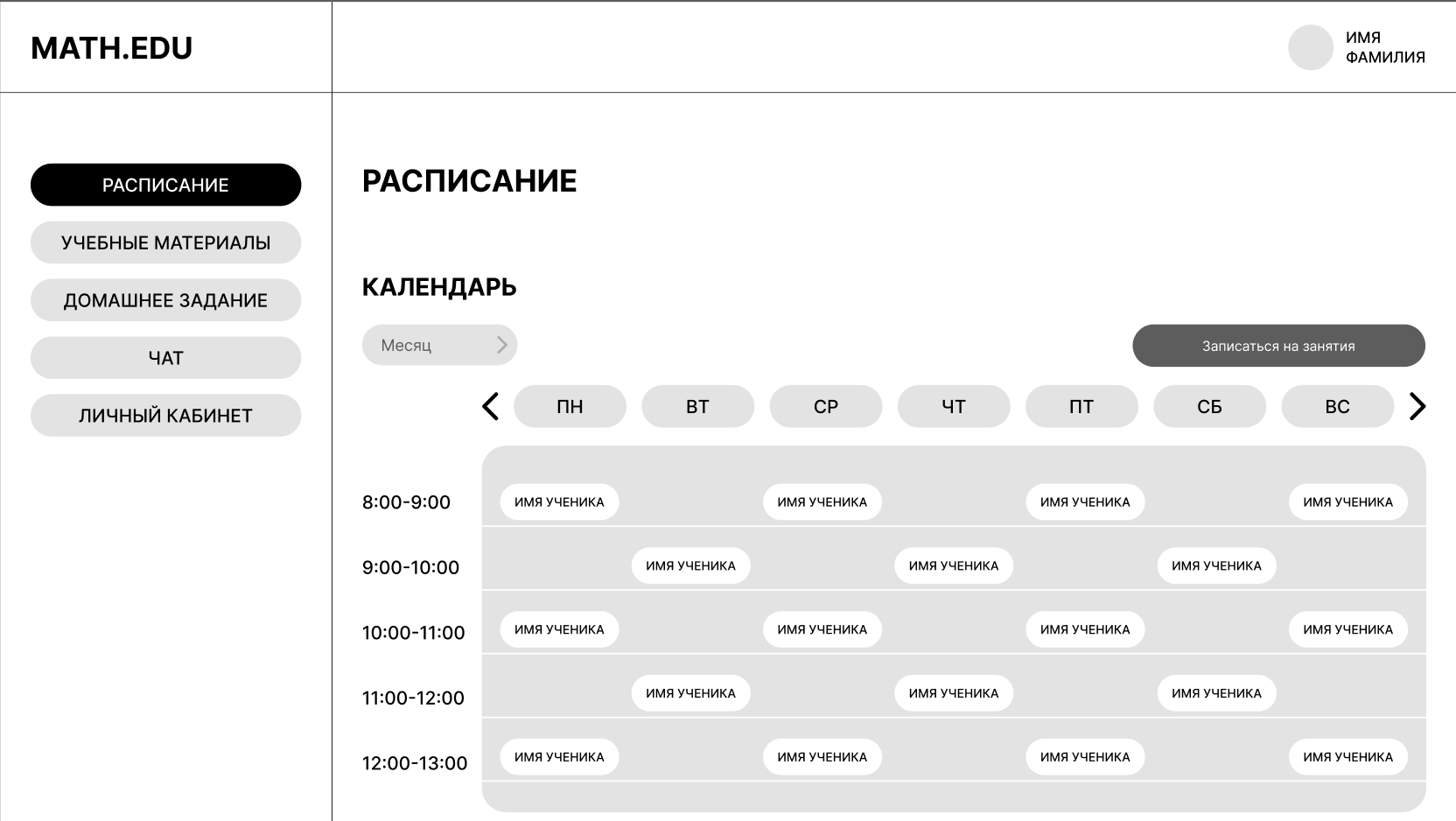
При отсутствии профиля пользователь сможет перейти на форму регистрации, где ему потребуется ввести адрес электронной почты, логин, класс и придумать пароль. На рисунке 9 приведён прототип формы регистрации.

  
Рисунок 9 – Прототип экранной формы регистрации

При успешном входе в систему откроется экранная форма главной страницы (рисунок 8). На странице будут представлены:

* кнопка «Записаться на занятие» – при нажатии на кнопку система должна будет открыть модальное окно для подтверждения записи на занятие;
* кнопка «Чат» – при нажатии на кнопку система должна будет открыть страницу со списком чатов пользователей системы;
* кнопка «Домашнее задание» – при нажатии на кнопку система должна будет открыть страницу с информацией о домашнем задании для учеников;
* кнопка «Учебные материалы» – при нажатии на кнопку система должна будет открыть страницу со списком учебных материалов;
* кнопка «Личный кабинет» – при нажатии на кнопку система должна будет открыть страницу личного профиля пользователя;

На рисунке 10 представлен прототип экранной формы главной страницы.

  
Рисунок 10 – Прототип экранной формы главной страницы

1. Реализация web-сайта
   1. Разработка физической модели БД

Физическое проектирование базы данных – это завершающий этап её разработки, на котором принимаются решения о методах реализации созданной структуры. На этом этапе определяется, как база данных будет функционировать на конкретной аппаратной и программной платформе. В процессе логического проектирования была сформирована логическая модель базы данных, включающая описание связей и ограничений, отражающих особенности рассматриваемой предметной области.

Физическая модель базы данных содержит все детали, необходимые конкретной СУБД для создания базы: наименования таблиц и столбцов, типы данных, определения первичных и внешних ключей [13].

В качестве СУБД была выбрана MySQL. Среди главных ее достоинств, необходимых для решения поставленной задачи, можно отметить:

* + - * оптимизацию связей с присоединением многих данных за один проход;
      * гибкую поддержку форматов чисел, строк переменной длины и меток времени;
      * быструю поддержку транзакций через механизм InnoDB [14].

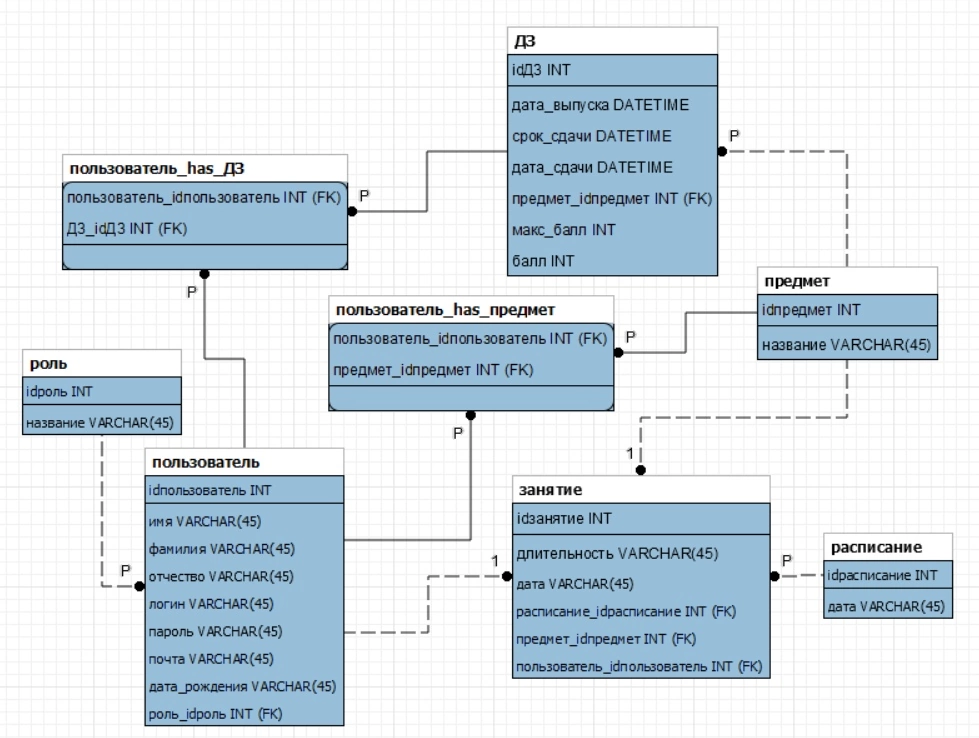
Кроме указанных достоинств, в MySQL также есть поддержка хранимых процедур и функций.

Хранимые процедуры – это набор инструкций, которые компилируются и сохраняются в базе данных. Они могут быть вызваны из других программ или запросов к базе данных. Хранимые функции – это подобные процедурам, но они всегда возвращают значение, которое может быть использовано в других запросах или выражениях [15].

В облачном хранилище используются хранимые процедуры для реализации различных операций, таких как:

* создание нового пользователя: хранимая процедура принимает на вход логин, пароль и другую необходимую информацию о пользователе, а затем создает новую запись в таблице пользователей;
* удаление пользователя: хранимая процедура принимает на вход идентификатор пользователя и удаляет соответствующую запись из базы данных;
* создание домашнего задания: хранимая процедура принимает на вход название предмета, а затем создает новую запись в таблице домашних заданий.
* создание занятия: хранимая процедура принимает на вход дату, а затем создает новую запись в таблице занятий.

На рисунке 11 представлена физическая модель данных системы.

  
Рисунок 11 – Физическая модель базы данных

* 1. Разработка серверной части

Разработка серверной части является важнейшим аспектом веб-разработки и разработки программного обеспечения. Это включает в себя создание и поддержку серверных компонентов приложения, которые отвечают за обработку данных и управление ими, а также за обработку скрытой логики [16].

Серверная часть облачного хранилища разработана на языке JavaScript (JS). Использование языка JavaScript на сервере и на клиенте упрощает разработку.

Рассмотрим реализацию серверной части.

Регистрация пользователя:

* сервер получает запрос, хеширует пароль и вызывает хранимую процедуру в БД.

Создание занятия:

* пользователь отправляет запрос на создание нового занятия: дата, предмет, длительность;
* сервер получает запрос и вызывает хранимую процедуру.

Удаление пользователя:

* администратор отправляет запрос на удаление пользователя;
* сервер выполняет удаление записи из таблицы.
  1. Разработка клиентской части

Разработка клиентской части веб-приложения является важным этапом создания любого веб-проекта. Клиентская часть, также известная как front-end, отвечает за то, как приложение будет выглядеть и взаимодействовать с пользователями [17].

Клиентская часть облачного хранилища написана на языке JavaScript (JS). Рассмотрим её реализацию.

Регистрация пользователя:

* пользователь заполняет форму регистрации, вводит email и пароль;
* клиент отправляет POST-запрос на сервер с учетными данными пользователя.

Аутентификация пользователя:

* пользователь вводит свои учетные данные (email и пароль) в форму авторизации;
* клиент отправляет POST-запрос на сервер с учетными данными.

Создание занятия:

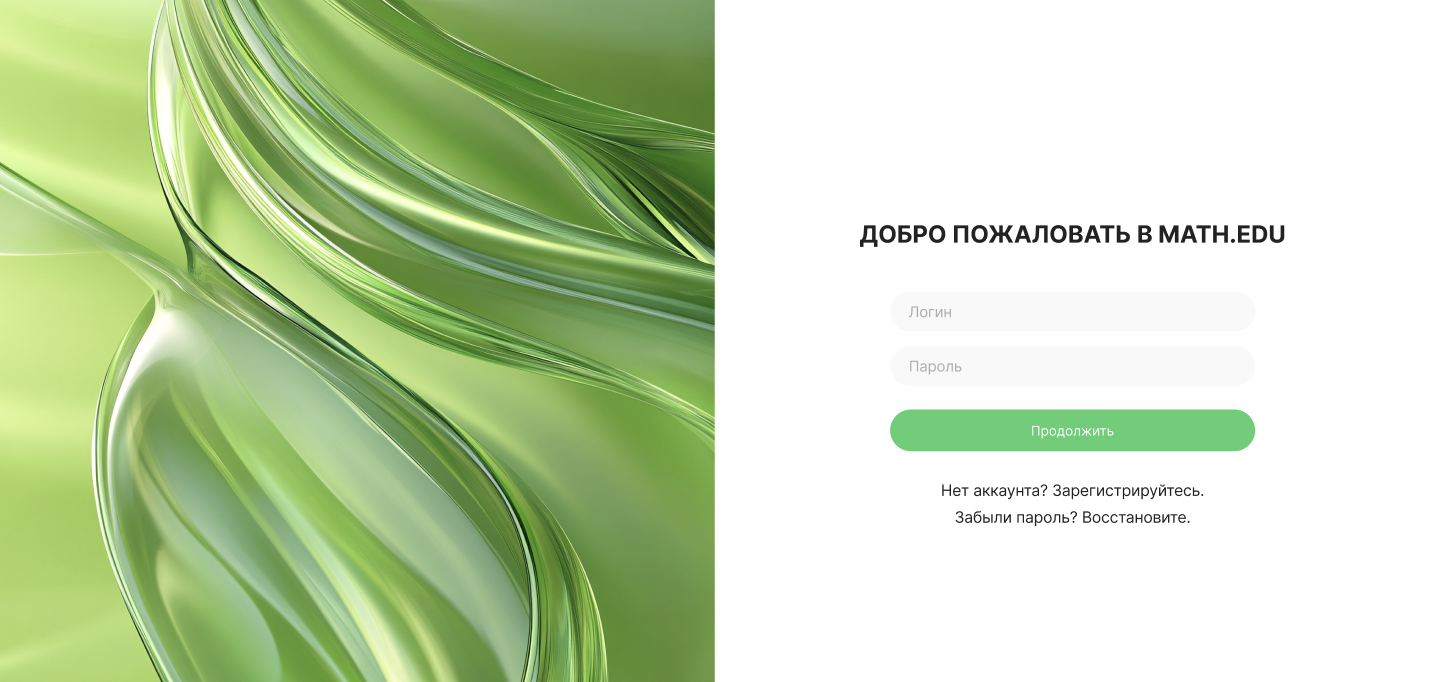
* пользователь заполняет поля для создания объявления: дата, предмет, длительность;
* клиент отправляет POST-запрос на сервер с введенной информацией

Удаление пользователя:

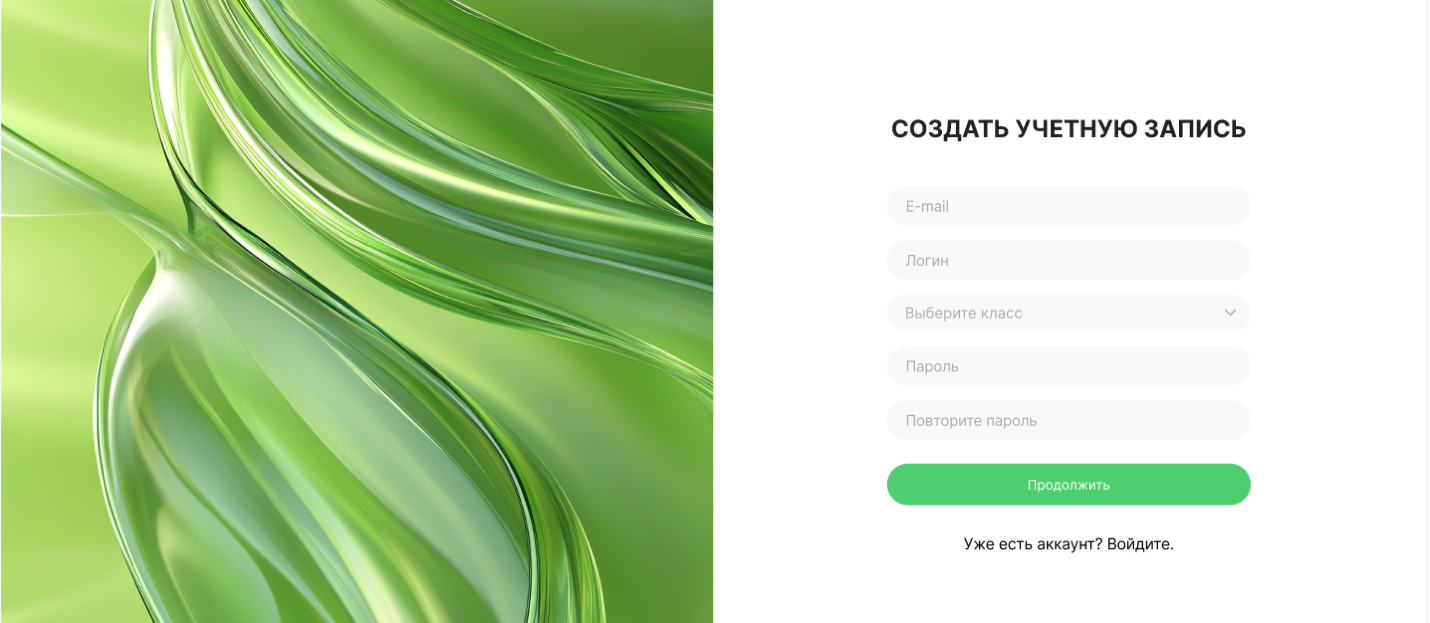
* администратор выбирает пользователя и нажимает кнопку "Удалить";
* клиент отправляет DELETE-запрос на сервер с ID удаляемого пользователя.
  1. Тестирование сайта

Веб-тестирование – это процесс проверки и оценки качества веб-приложений или сайтов, чтобы обеспечить их надежность, производительность, безопасность и совместимость с различными браузерами, устройствами и операционными системами [18].

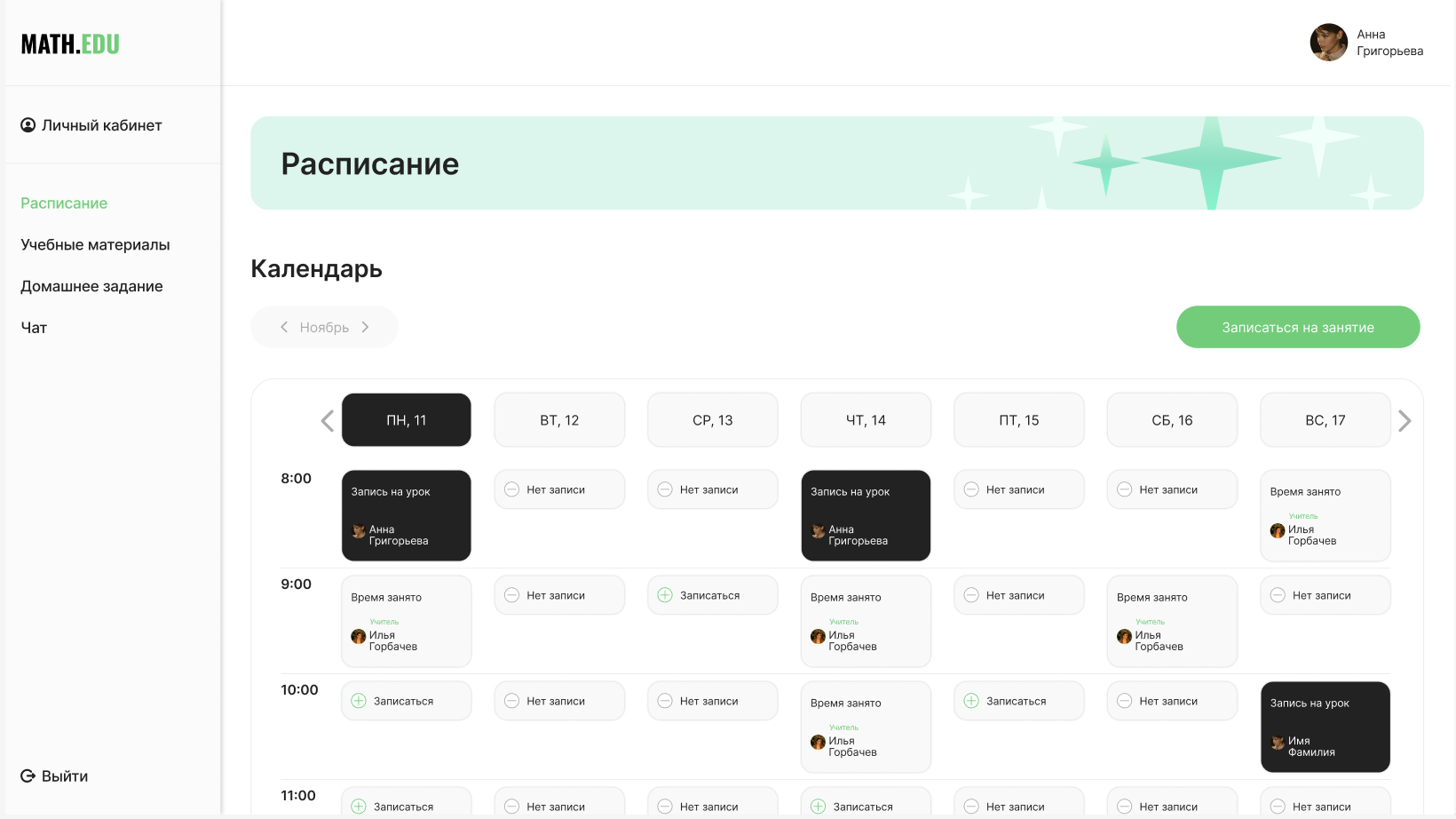
На рисунке 12 приведена страница авторизации, где пользователю предоставляется возможность войти в свой аккаунт.

  
Рисунок 12 – Страница авторизации

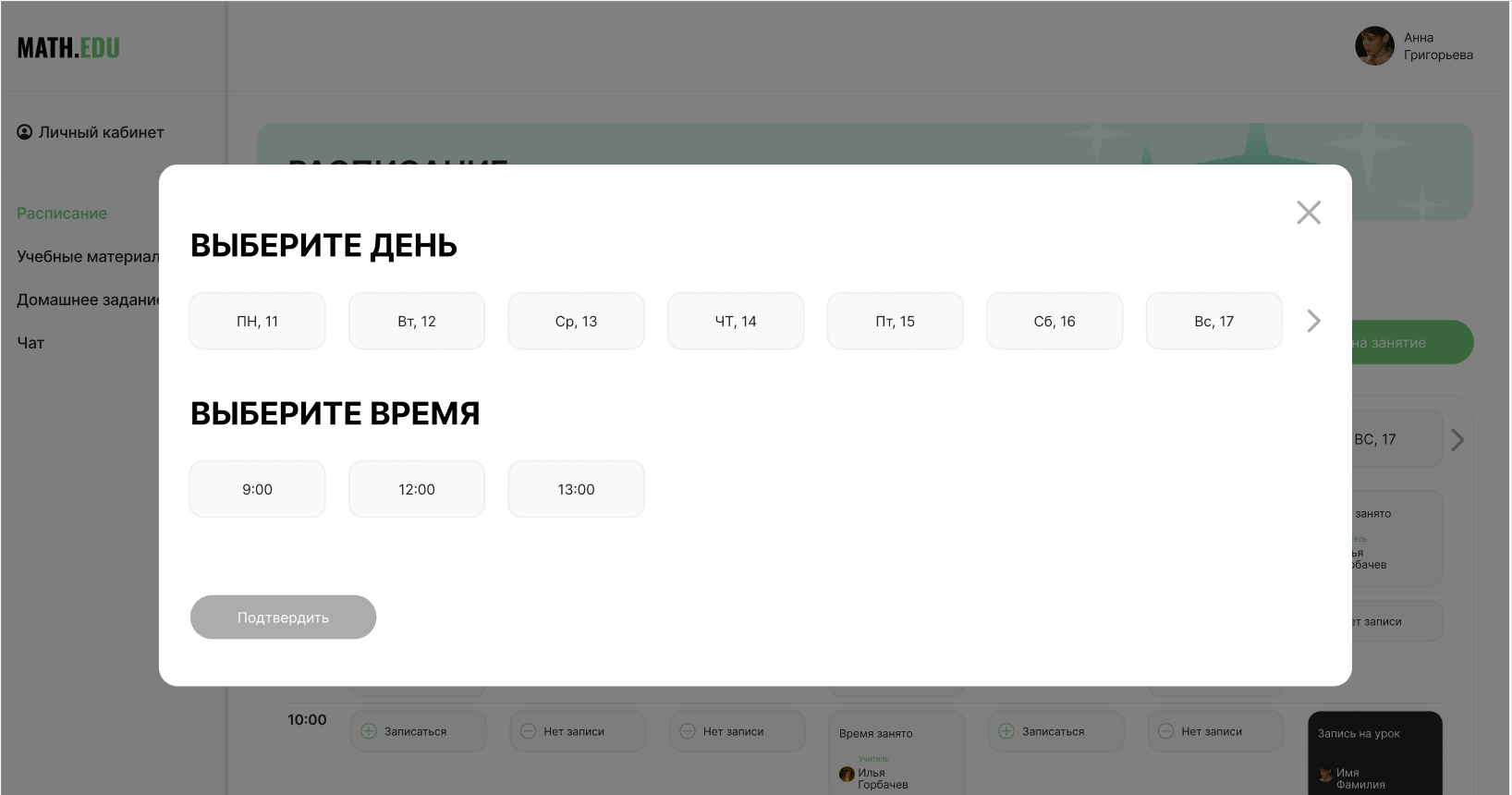
Если у пользователя нет аккаунта, он может перейти на страницу регистрации, где представлена возможность создать профиль. Страница регистрации приведена на рисунке 13.

  
Рисунок 13 – Страница регистрации

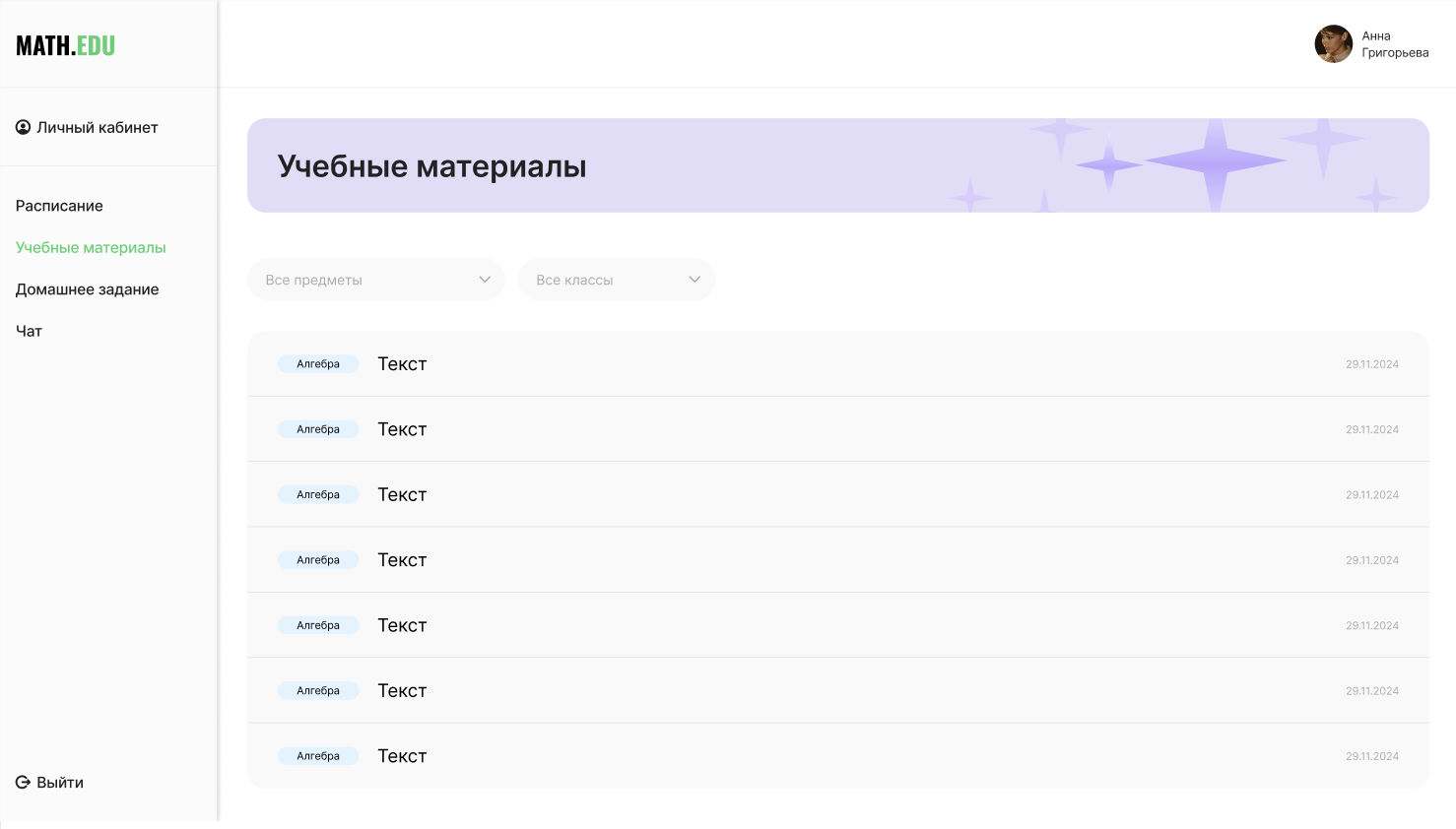
После успешного входа в систему открывается главная страница интернет-магазина, которая приведена на рисунке 14. На этой странице отображаются фотографии салона и основные разделы интернет-магазина.

  
Рисунок 14 – Главная страница

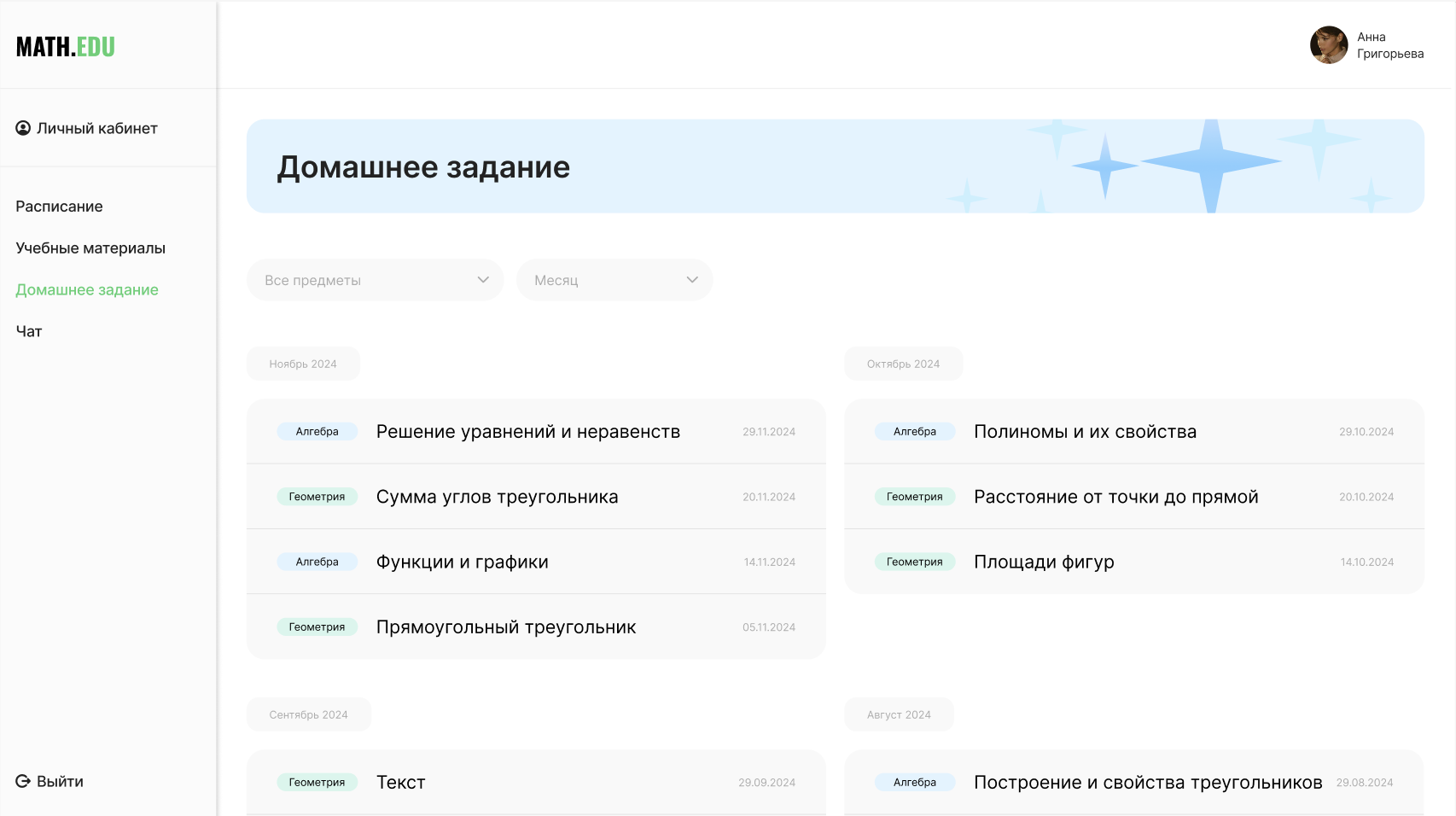
При нажатии на кнопку «Записаться на занятие» открывается модальное окно, в котором пользователю нужно выбрать дату и время планируемого занятия. На рисунке 15 показано модальное окно записи на занятие.

  
Рисунок 15 – Модальное окно записи на занятие

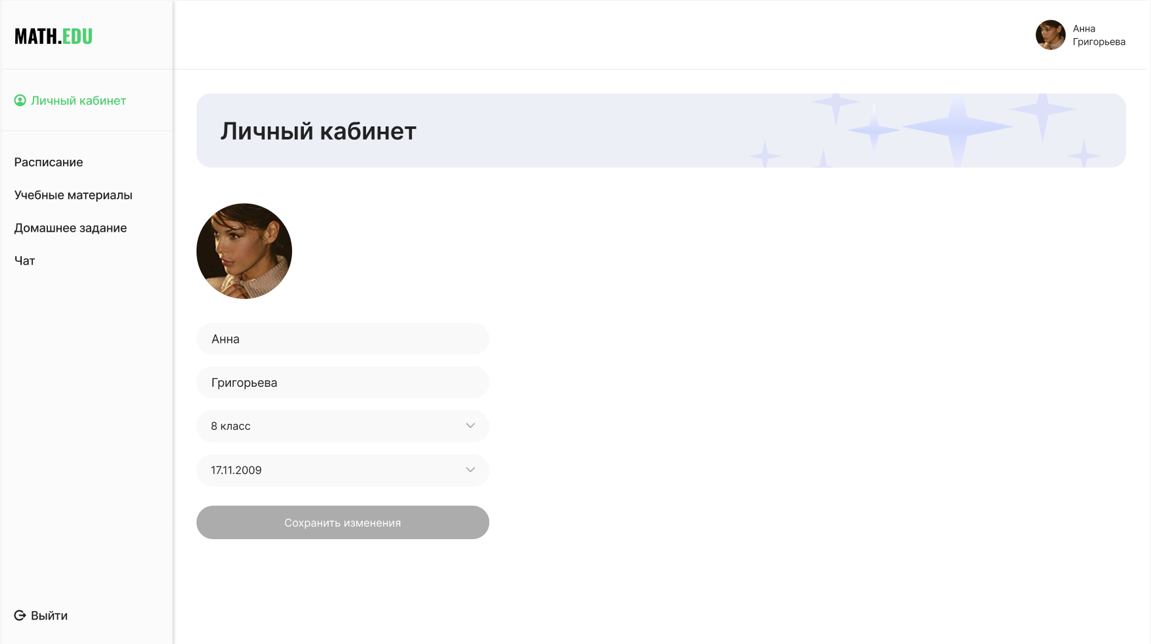
При нажатии на кнопку «Учебные материалы» открывается соответствующая страница. На странице отображаются доступные пользователю учебные материалы, которые он может отфильтровать по предмету и классу. На рисунке 16 приведена страница «Учебные материалы».

  
Рисунок 16 – Страница «Учебные материалы»

При нажатии на кнопку «Домашнее задание» открывается соответствующая страница. Пользователь может отфильтровать задания по предмету и сроку выполнения. На рисунке 17 показана страница «Домашнее задание».

  
Рисунок 17 – Страница «Домашнее задание»

При нажатии на кнопку «Личный кабинет» открывается страница с личными данными пользователя. Здесь он может изменить имя, фамилию. класс и дату рождения. На рисунке 18 приведена страница «Личный профиль».

  
Рисунок 18 – Работа функции сортировки

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта было разработан интернет-магазин для продажи подержанных автомобилей.

В первом разделе приведены основные понятия предметной области, описаны характеристики систем-аналогов, приведен их сравнительный анализ. На основе проведенного анализа сформулирована постановка задачи.

В разделе «Проектирование web-сайта» была построена логическая модель данных, описаны входные и выходные данные, перечислены исключительные ситуации, построена диаграмма вариантов использования и разработаны прототипы интерфейса пользователя.

В разделе «Реализация web-сайта» была описана разработка физической модели БД, реализация клиентской и серверной частей, и проведено тестирование системы.

Разработанная система может использоваться для удобного поиска и приобретения автомобилей. Она позволяет настраивать фильтры просмотра объявлений, оставлять заявки на приобретение и просматривать личный профиль.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Что такое интернет-магазин [Электронный ресурс]. URL: https://unisender.com/ru/glossary/chto-takoe-internet-magazin-vidy-struktura-celi/ (дата обращения: 10.12.2024).

Предметная область: понятие и основные принципы [Электронный ресурс]. URL: https://uchet-jkh.ru/i/predmetnaya-oblast-ponyatie-i-osnovnye-principy/ (дата обращения: 10.12.2024).

Интернет-магазин [Электронный ресурс]. URL: https://www.intervolga.ru/blog/projects/e-shop-terminy/ (дата обращения: 10.12.2024).

Как работают базы данных [Электронный ресурс]. URL: https://practicum.yandex.ru/blog/chto-takoe-bazy-dannyh/ (дата обращения: 22.12.2024).

Math playground – официальный сайт [Электронный ресурс]. URL: https://www.mathplayground.com/ (дата обращения: 14.12.2024).

Khan Academy – официальный сайт [Электронный ресурс]. URL: https://ru.khanacademy.org/ (даты обращения: 12.12.2024).

Moodle – официальный сайт [Электронный ресурс]. URL: https://moodle.org/?lang=ru (дата обращения: 14.12.2024).

Функциональная модель [Электронный ресурс]. URL: https://wiki5.ru/wiki/Function\_model (дата обращения: 14.12.2024).

Диаграмма вариантов использования [Электронный ресурс]. URL: https://itonboard.ru/analysis/629diagramma\_variantov\_ispolzovanija\_use\_case\_diagram/ (дата обращения: 14.12.2024).

Логическая схема данных [Электронный ресурс]. URL: https://docs.arenadata.io/adtm/Admin/Logical.html (дата обращения: 14.12.2024).

Построение и анализ алгоритмов обработки данных [Электронный ресурс]. URL: https://elar.urfu.ru/bitstream/10995/34782/1/978-5-7996-1489-8 (дата обращения: 14.12.2024).

Зачем нужно прототипирование интерфейсов [Электронный ресурс]. URL: https://inostudio.com/blog/articles-design/zachem-nuzhno-prototipirovanie-interfeysov/# (дата обращения: 14.12.2024).

Физическая модель базы данных [Электронный ресурс]. URL: https://neerc.ifmo.ru/wiki/index.php?title (дата обращения: 14.12.2024).

MySQL: основные достоинства и недостатки [Электронный ресурс]. URL: https://timeweb.cloud/blog/mysql-preimushchestva-i-nedostatki (дата обращения 14.12.2024).

Хранимые процедуры и функции [Электронный ресурс]. URL: https://wiki.dieg.info/xranimye\_procedury\_funkcii\_i\_triggery (дата обращения: 16.12.2024).

Разработка серверной части [Электронный ресурс]. URL: https://dzen.ru/a/ZPJt3skCSEafwlkq (дата обращения: 16.12.2024).

Разработка клиентской части [Электронный ресурс]. URL: https://science-engineering.ru/ru/article/view?id=1278 (дата обращения: 16.12.2024).

Что такое web-тестирование и как его проводить [Электронный ресурс]. URL: https://sky.pro/media/chto-takoe-veb-testirovanie-i-kak-ego-provodit/ (дата обращения: 16.12.2024).

ПРИЛОЖЕНИЕ А  
Листинг программы

def index(request):

data = {

'title': 'Главная страница',

'menu': menu,

}

return render(request, 'mainapp/index.html', context=data)

class MaterialCreate(LoginRequiredMixin, CreateView):

model = Material

form\_class = MaterialForm

template\_name = 'mainapp/create\_material.html'

success\_url = reverse\_lazy('mainapp:create\_material')

def form\_valid(self, form):

form.instance.teacher = self.request.user

return super().form\_valid(form)

class MaterialListView(LoginRequiredMixin, ListView):

model = Material

form\_class = MaterialForm

template\_name = 'mainapp/list\_materials.html'

success\_url = reverse\_lazy('mainapp:list\_materials')

def form\_valid(self, form):

form.instance.teacher = self.request.user

return super().form\_valid(form)

def get\_context\_data(self, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

if self.request.user.role == 'teacher':

context['materials'] = Material.objects.filter(teacher=self.request.user)

else:

context['materials'] = Material.objects.filter(teacher\_\_isnull=False)

context['message'] = "Вы не имеете доступа к загрузке материалов."

context['categories'] = Category.objects.all()

return context

class MaterialUpdateView(LoginRequiredMixin, UpdateView):

model = Material

form\_class = MaterialUpdateForm

template\_name = 'mainapp/material\_update.html'

success\_url = reverse\_lazy('mainapp:list\_materials')

def form\_valid(self, form):

form.instance.teacher = self.request.user

return super().form\_valid(form)

def get\_context\_data(self, \*\*kwargs):

context = super().get\_context\_data(\*\*kwargs)

if self.request.user.role == 'teacher':

context['materials'] = Material.objects.filter(teacher=self.request.user)

else:

context['materials'] = Material.objects.filter(teacher\_\_isnull=False)

context['message'] = "Вы не имеете доступа к загрузке материалов."

return context

class CategoryMaterials(View):

template\_name = 'mainapp/list\_materials.html'

def get(self, request, slug):

category = get\_object\_or\_404(Category, slug=slug)

materials = category.posts.all()

context = {

'category': category,

'materials': materials,

'categories': Category.objects.all(),

'cat\_selected': category.pk,

}

return render(request, self.template\_name, context)

@login\_required

def delete\_material(request, pk):

material = Material.objects.get(pk=pk)

if request.user.role == 'teacher':

material.delete()

return redirect('mainapp:list\_materials')

class LoginUser(LoginView):

form\_class = LoginUserForm

template\_name = 'users/login.html'

extra\_context = {'title': 'Авторизация'}

def logout\_user(request):

logout(request)

return HttpResponseRedirect(reverse('users:login'))

class RegisterUser(CreateView):

form\_class = RegisterUserForm

template\_name = 'users/register.html'

extra\_context = {'title': "Регистрация"}

success\_url = reverse\_lazy('users:login')

class ProfileUser(LoginRequiredMixin, UpdateView):

model = get\_user\_model()

form\_class = ProfileUserForm

template\_name = 'users/profile.html'

extra\_context = {

'title': "Профиль пользователя",

'default\_image': settings.DEFAULT\_USER\_IMAGE,

}

def get\_success\_url(self):

return reverse\_lazy('users:profile')

def get\_object(self, queryset=None):

return self.request.user

class UserPasswordChange(PasswordChangeView):

form\_class = UserPasswordChangeForm

success\_url = reverse\_lazy("users:password\_change\_done")

template\_name = "users/password\_change\_form.html"