Частное учреждение

профессиональная образовательная организация

ТЕХНИКУМ «БИЗНЕС И ПРАВО»

ЗАЩИЩЕНО С ОЦЕНКОЙ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и ФИО руководителя)

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

**ОТЧЕТ О ВЫПОЛНЕНИИ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ**

**ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

|  |  |
| --- | --- |
| Тема | Разработка сайта «Электронный журнал» |
| По МДК 09.01 | Проектирование и разработка веб-приложений |
| Выполнил | Грибенюков К.Н. |
| Курс, группа | 4 курс, группа ИС-4 |
| Специальность | 09.02.07 Информационные системы и программирование |
| Научный руководитель | Мартыненко В.А. |

г. Белореченск

2024 г.

Содержание

[ВВЕДЕНИЕ 3](#_Toc161226262)

[1. АНАЛИТИЧЕСКАЯ ЧАСТЬ 4](#_Toc161226263)

[1.1 Анализ предметной области 4](#_Toc161226264)

[1.2 Анализ существующих решений 4](#_Toc161226264)

[1.3 Проектирование базы данных 5](#_Toc161226265)

[1.4 Цели и задачи выполняемой системы 5](#_Toc161226266)

[1.5 Метод реализации процесса проектирования 5](#_Toc161226267)

[1.6 Обоснование проектных решений 6](#_Toc161226268)

[2. ПРОЕКТНАЯ ЧАСТЬ 30](#_Toc161226269)

[2.1 Техническое задание 30](#_Toc161226270)

2.2 Наименование предприятий разработчика и заказчика системы, их реквизиты 30

[2.3 Программное обеспечение информационной системы 32](#_Toc161226271)

[2.4 Руководство пользователя 33](#_Toc161226272)

[2.5 Руководство администратора 33](#_Toc161226273)

[ЗАКЛЮЧЕНИЕ 36](#_Toc161226274)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 37](#_Toc161226275)

[ПРИЛОЖЕНИЕ 38](#_Toc161226276)

# ВВЕДЕНИЕ

Электронный журнал является важным инструментом для систематизации и хранения информации о процессе обучения и успеваемости студентов. Он позволяет участникам образовательного процесса в реальном времени отслеживать прогресс студентов, а также анализировать их достижения и проблемы.

Целью данного электронного журнала является создание удобной и эффективной среды для ведения учета успеваемости и активности студентов. Журнал предоставляет возможность записи оценок, комментариев преподавателей, а также другой информации, необходимой для полного отражения учебного процесса.

В этом журнале мы собираемся отображать информацию о студентах, их успехах и проблемах, а также предоставлять рекомендации по повышению качества образования и эффективности учебного процесса.

Объект исследования – целью данного исследования является веб-сайт "Электронный журнал", разработанный с использованием PHP, CSS, Bootstrap, HTML.

Цель исследования – разработка и анализ веб-сайта " Электронный журнал", который облегчит доступ к просмотру оценок.

Методы исследования – для достижения поставленной цели исследования были использованы следующие методы:

1. Анализ существующих решений - был проведен анализ существующих веб-сайтов, связанных с журналом, для выявления их преимуществ и недостатков.
2. Разработка прототипа - был разработан прототип веб-сайта, который был использован на целевой аудитории для получения данных о его эффективности и удобстве использования.
3. Разработка и тестирование - была проведена разработка и тестирование сайта с использованием различных технологий и инструментов, чтобы обеспечить его функциональность и надежность.
4. Анализ результатов - был проведен анализ результатов, полученных в ходе исследования, для выявления улучшений и возможных направлений дальнейшего развития “Электронного дневника”

Начало формы

# 1. АНАЛИТИЧЕСКАЯ ЧАСТЬ

## Анализ предметной области

Общие сведения

Был проведен анализ предметной области индивидуально образовательного учреждения IT лицей лидер по адресу Краснодарский край, г. Белореченск, ул. Гоголя, д. 53 (2 этаж).

1.1.1 Определение концепций и принципов электронного журнала

Первым этапом анализа было изучение основных концепций и принципов централизованного хранения информации. Это помогло выявить возможные проблемы с безопасности информацией и доступе к спискам оценок, а также способы их решения с использованием данного веб-ресурса.

1.1.2 Определение требований к системе

На основе изученных концепций и принципов литературной сферы, а также анализа существующих решений, были определены требования к платформе "Электронный журнал", которые были реализованы в данном веб-приложении.

1.2 Анализ существующих решений

Был проведен анализ существующих электронных журналов. Это помогло понять, какие проблемы уже решаются в данной доступа к оценкам, а также какие функции могут быть полезны для разработки нового электронного журнала.

## 

## 1.3 Проектирование базы данных

Также было описано проектирование базы данных для электронного журнала. Это включает определение структуры базы данных, выбор соответствующих типов данных и структур данных, а также проектирование связей между таблицами.

## 

## 1.4 Цели и задачи выполняемой системы

Основные цели системы:

* Разработать удобный веб-сайт для ознакомления с информацией о оценках и информацией о журнале.
* Реализовать функциональность регистрации и авторизации пользователей.
* Для работы в режиме администратора реализовать возможность добавления и удаления списка групп, подгрупп, а также добавление и редактирование оценок.

Задачами являются:

* Достичь поставленных целей;
* Создать отчет о выполненной работе и результатах обучения.

## 1.5 Метод реализации процесса проектирования

Метод реализации процесса проектирования Электронного журнала может включать следующие этапы:

1. Анализ требований:
2. Определение функциональных возможностей электронного журнала для просмотра, включая управление доступом к группам, регистрацию пользователей.
3. Определение структуры базы данных для хранения информации о оценках, пользователях и группах.
4. Проектирование базы данных:
5. Создание структуры базы данных, отражающей связи между оценками, пользователями и другими сущностями электронного журнала.
6. Определение таблиц, их атрибутов и связей между ними для эффективного хранения и управления данными.
7. Разработка бэкенда:
8. Использование PHP для создания серверной части электронного журнала
9. Реализация функциональности управления доступом к оценкам, регистрации пользователей и других операций, необходимых для работы электронного журнала.
10. Разработка фронтенда:
11. Использование HTML, CSS и Bootstrap для создания пользовательского интерфейса электронного журнала.
12. Создание страниц с формами для регистрации пользователей, просмотра оценок, поиска материалов и других функциональных элементов.
13. Тестирование и отладка:
14. Проведение тестирования электронного журнала на различных уровнях, включая модульное тестирование бэкенда и функциональное тестирование фронтенда.
15. Выявление и исправление ошибок и недочетов в процессе разработки для обеспечения стабильной работы приложения.

## 1.6 Обоснование проектных решений

Выбор MySQL для хранения информации о пользователях, оценок. Требования к системе разработаны в соответствии со стандартом качества программного обеспечения ISO 9126:2001, который определяет оценочные характеристики качества программного обеспечения. Этот стандарт описывает многоуровневую модель характеристик качества и соответствующий им набор атрибутов, что помогает обосновать выбор конкретного способа хранения данных.

Требования к системе (электронный журнал) представлены в таблице 1.1.

Таблица 1.1 – Требования к системе электронного журнала

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибуты | Метки | | Требования |
| 1 | 2 | | 3 |
| Удобство | | | |
| Простота использования | Среднее время, необходимое  пользователю, чтобы найти  необходимый элемент в программе | | Поиск отдельного элемента интерфейса не должен занимать более 1 секунды |
| Привлекательность | Соответствие интерфейса  требованиям | | 99% элементов должны быть работоспособны для  пользователя |
| Обучаемость | Показатель, затрачиваемый  пользователями на обучение | | Пользователь должен изучить работу ПО за первые 3-5 минут |
| Производительность | | | |
| Временная эффективность | Время выполнения компонента программы | - Приложение должно  обслуживать любую функцию не дольше 1 сек  (не включая задержки в сети); | |
| Эффективность использования ресурсов | Объемы ресурсов требуемых для выполнения задач | - Постоянная память,  используемая программой – не более 10МБ;  - Приложение должно потреблять не более 3МБ  памяти на каждый неактивный сеанс с  пользователем; | |
| Переносимость | | | |
| Удобство установки | Легкость установки | Необходима установка:  xampp, mysql, admin php | |
| Способность к сосуществованию | Способность ПО сосуществовать с другими  программами в общем окружении | Программа не должна  занимать большие объемы памяти устройства (максимум 3МБ) | |
| Удобство замены другого ПО данным | Совместимость структур данных | - Должна поддерживать  внедрение новых версий;  - Система должна быть применима вместо других  программных систем для решения тех же задач в определенном окружении. | |

Требования к численности и квалификации персонала системы и режиму его работы:

В системе существуют 2 группы пользователей: Пользователь и Администратор. Пользователи выполняют функцию: просмотра.

Администратор имеет возможность добавления, редактирования и удаления контента веб-сайта.

Пользователи системы должны иметь опыт работы с персональным компьютером на базе операционных систем Microsoft Windows на уровне пользователя.

В таблице 1.2 представлены требования к надежности.

Таблица 1.2 – Требования к надёжности

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Атрибуты | Метрики | | Требования | |
| 1 | | 2 | | 3 |
| Надёжность | | | | |
| Устойчивость к отказу | Коэффициент аварийных отказов | | - Не менее 90% ошибок | |
|  |  | | в программе должны  обрабатываться без  экстренного завершения;  - Обеспечение бесперебойного питание  активного сетевого  оборудования; | |
| Способность к восстановлению | Отношение количества числа успешного восстановление работы системы к ее аварийному завершению за определённый промежуток времени | | - Восстановление  работоспособности  системы не должно превышать 5 минут; | |
| Уровень зрелости | Среднее время работы без сбоев | | - Система должна  работать 24 часа в сутки;  - Система должна оповещать  - Система должна оповещать пользователя при некорректных действиях в системе;  - Данные, которые вводит пользователь, должны сохраняться. | |

Средняя доступность приложения должна составлять не менее 99%

Среднее время между сбоями — это среднее время, за которое компонент или модуль может выполнять свои функции без перерыва.

Измеряется от начала работы до момента следующего сбоя.

Среднее, временя работы без сбоев должно составлять не менее 1500 часов.

В таблице 1.3 представлен перечень и критерии отказов для каждой функции, по которой задаются требования по надежности.

Таблица 1.3 – Перечень и критерии отказов для каждой функции

|  |  |
| --- | --- |
| Функции | Критерии отказа |
| 1 | 2 |
| Отображение данных | Отсутствие соединения с базой данных |
| Добавление | Отсутствие соединения с базой данных, неверно заполненные поля |
| Редактирование | Отсутствие соединения с базой данных, неверно заполненные поля |
| Удаление | Отсутствие соединения с базой данных |

Требования к функциям web-приложения, выполняемым программным обеспечением

Имя: Регистрация / авторизация

Описание: Пользователь авторизуется или регистрируется на сайте

Действующий субъект: Пользователь

Предусловие: пользователь зашел на страницу регистрации или авторизации

Сценарий:

* Пользователь зарегистрировался или авторизовался на сайте
* Пользователь перенаправлен в свой личный кабинет

Расширения: в случае неверно введенных данных при авторизации или регистрации будет выводиться ошибка «Поля заполнены некорректно».

Имя: Регистрация / авторизация

Описание: Пользователь авторизуется или регистрируется на сайте

Действующий субъект: Пользователь

Предусловие: пользователь зашел на страницу регистрации или авторизации

Сценарий:

* Пользователь зарегистрировался или авторизовался на сайте
* Пользователь перенаправлен в свой личный кабинет

Расширения: в случае неверно введенных данных при авторизации или регистрации будет выводиться ошибка «Поля заполнены некорректно».

Имя: Добавление оценки

Описание: Пользователь с ролью администратора добавляет пользователю его оценку по предмету

Действующий субъект: Пользователь с ролью администратора

Предусловие: Пользователь авторизовался в системе как администратор

Сценарий:

* Пользователь авторизовался на сайте как администратор
* Пользователь перенаправлен на главную страницу
* Пользователь нажал на поле с оценкой
* Пользователь попал на страницу журнала, заполнил поля и вписал нужную оценку

Имя: Редактирование оценки

Описание: Пользователь с ролью администратора редактирует информацию о оценке

Действующий субъект: Пользователь с ролью администратора

Предусловие: Пользователь авторизовался в системе как администратор

Сценарий:

* Пользователь авторизовался на сайте как администратор
* Пользователь перенаправлен в журнал
* Пользователь нажал на поле с оценкой
* Изменил на нужную ему оценку и обновил страницу

Имя: Удаление оценки

Описание: Пользователь с ролью администратора удаляет оценку

Действующий субъект: Пользователь с ролью администратора

Предусловие: Пользователь авторизовался в системе как администратор

Сценарий:

* Пользователь авторизовался на сайте как администратор
* Пользователь переходит на страницу журнала
* Пользователь стер оценку в поле и обновил страницу

Расширения: в случае отсутствия соединения с базой данных выведется сообщение об ошибке.

База данных информационной системы должна состоять из следующих связанных таблиц:

– grade – таблица, содержащая данные страницы «Оценок»

– groups – таблица, содержащая данные страницы «Групп»

* student – таблица, содержащая данные страницы «Пользователей»
* subject – таблица, содержащая данные страницы «Предмет»
* users – таблица, содержащая данные страницы «Пользователи в журнале»

Данные о пользователях представлены в таблице users 1.4.

Таблица 1.4 – Данные о пользователях

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | int | 11 | Идентификатор пользователя | Первичный ключ |
| Username | Varchar | 255 | Имя | NOT NULL |
| Lastname | Varchar | 255 | Отчество | NOT NULL |
| Surname | Varchar | 255 | Фамилия | NOT NULL |
| name | Varchar | 255 | Логин | NOT NULL |
| password | Long | 255 | Пароль пользователя | NOT NULL |
| role | Varchar | 255 | Роль | NOT NULL |

Данные со страницы «Оценок» должны размещаться в таблице grade 1.5.

Таблица 1.5 – Данные со страницы «Оценок»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | Int | 11 | Идентификатор  Оценки | Первичный ключ |
| student\_id | Varchar | 255 | Идентификатор пользователя | NOT NULL |
| grade | Varchar | 255 | Оценка | NOT NULL |

Данные со страницы «Групп» должны быть в таблице groups 1.6.

Таблица 1.6. – Данные со страницы «Групп».

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничения |
| id | Int | 11 | Индентификатор группы | Первичный ключ |
| name | Varchar | 255 | Название группы | NOT NULL |

Данные со страницы «Пользователи» должны размещаться в таблице student 1.7.

Таблица 1.7. – Данные со страницы «Пользователи».

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничения |
| id | INT | 11 | Идентификатор контента | Превичный ключ |
| name | Varchar | 255 | имя | NOT NULL |
| Lastname | Varchar | 255 | Отчество | NOT NULL |
| Surname | Varchar | 255 | фамилия | NOT NULL |
| group\_id | int | 13 | Группа id | NOT NULL |

Данные со страницы «Предмет» должны размещаться в таблице subject 1.8.

Таблица 1.8. – Данные со страницы «Предмет».

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничения |
| id | INT | 11 | Идентификатор предмет | Превичный ключ |
| name | Varchar | 255 | название | NOT NULL |
| Group\_id | int | 255 | Идентификатор группы | NOT NULL |

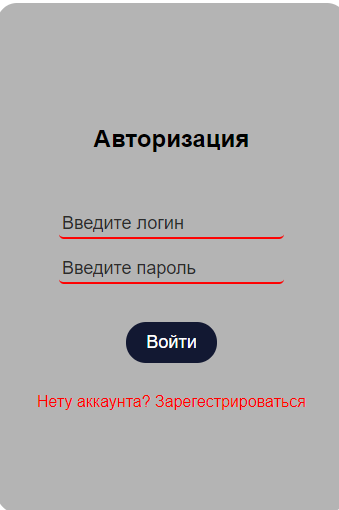


Рисунок 1.2. – окно авторизации.

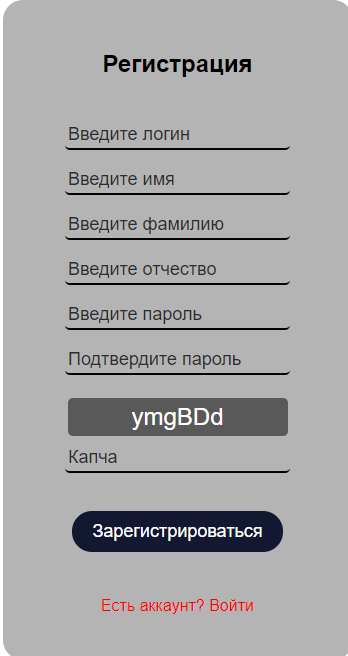


Рисунок 1.4. – окно регистрации.

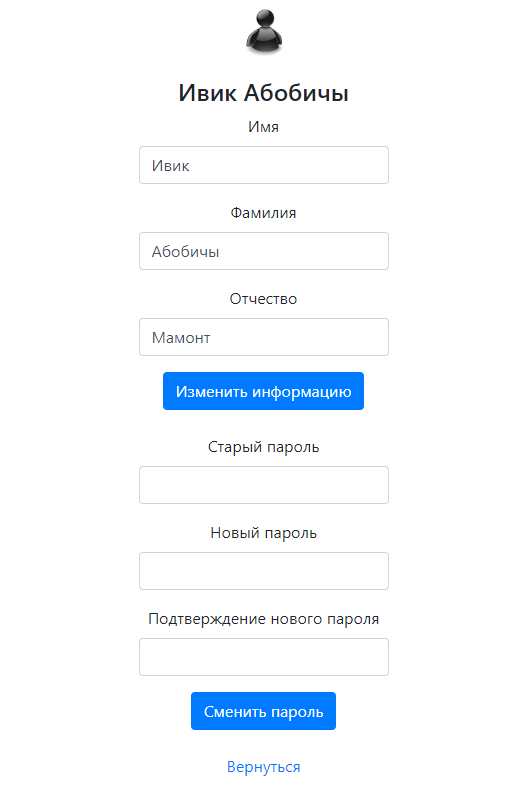


Рисунок 1.6 – страница личного кабинета.

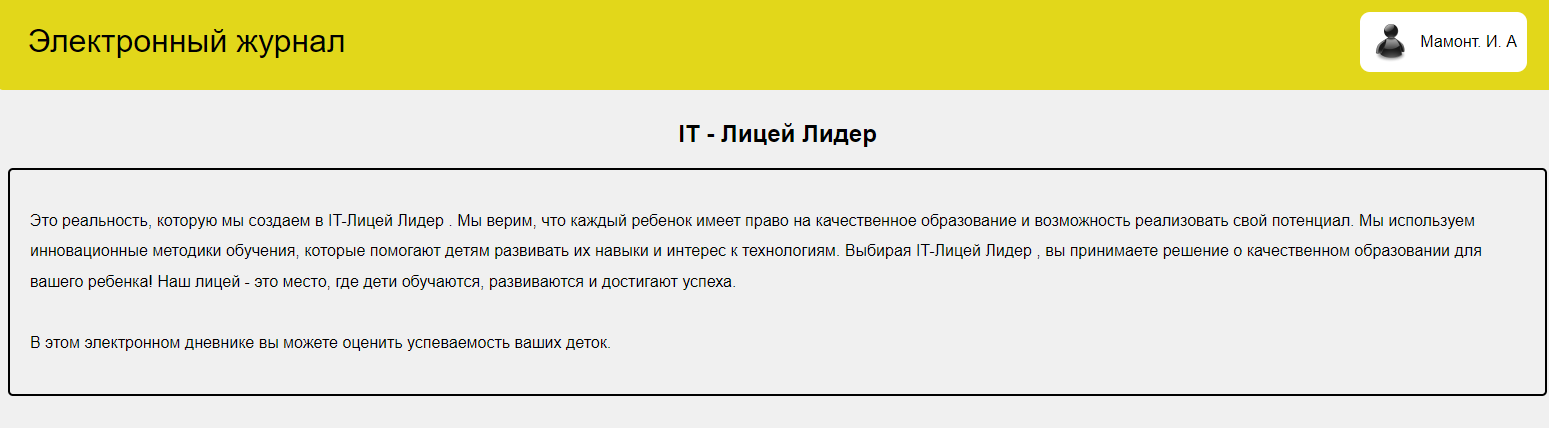


Рисунок 1.8 – страница главная.

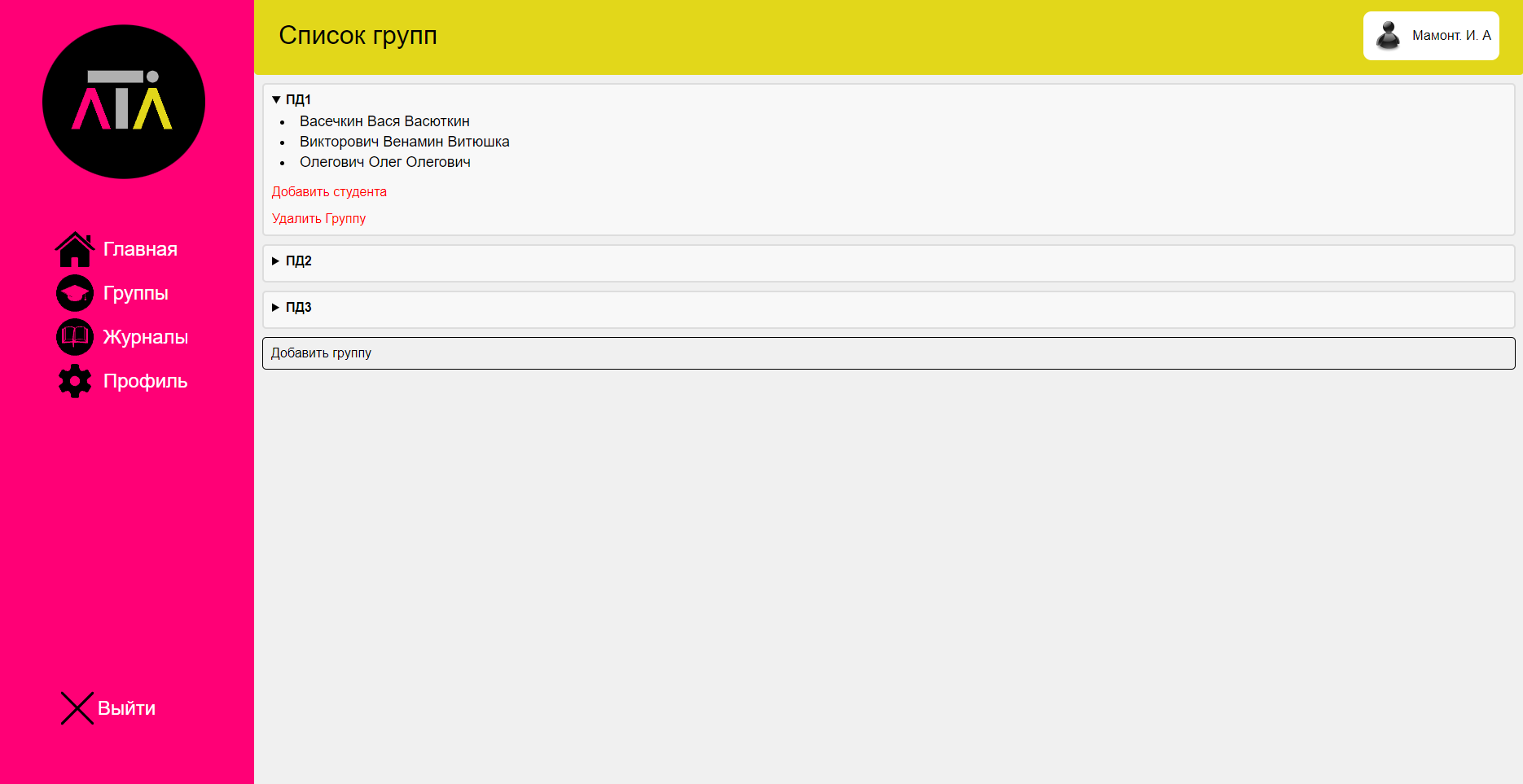


Рисунок 1.10 – страница групп.

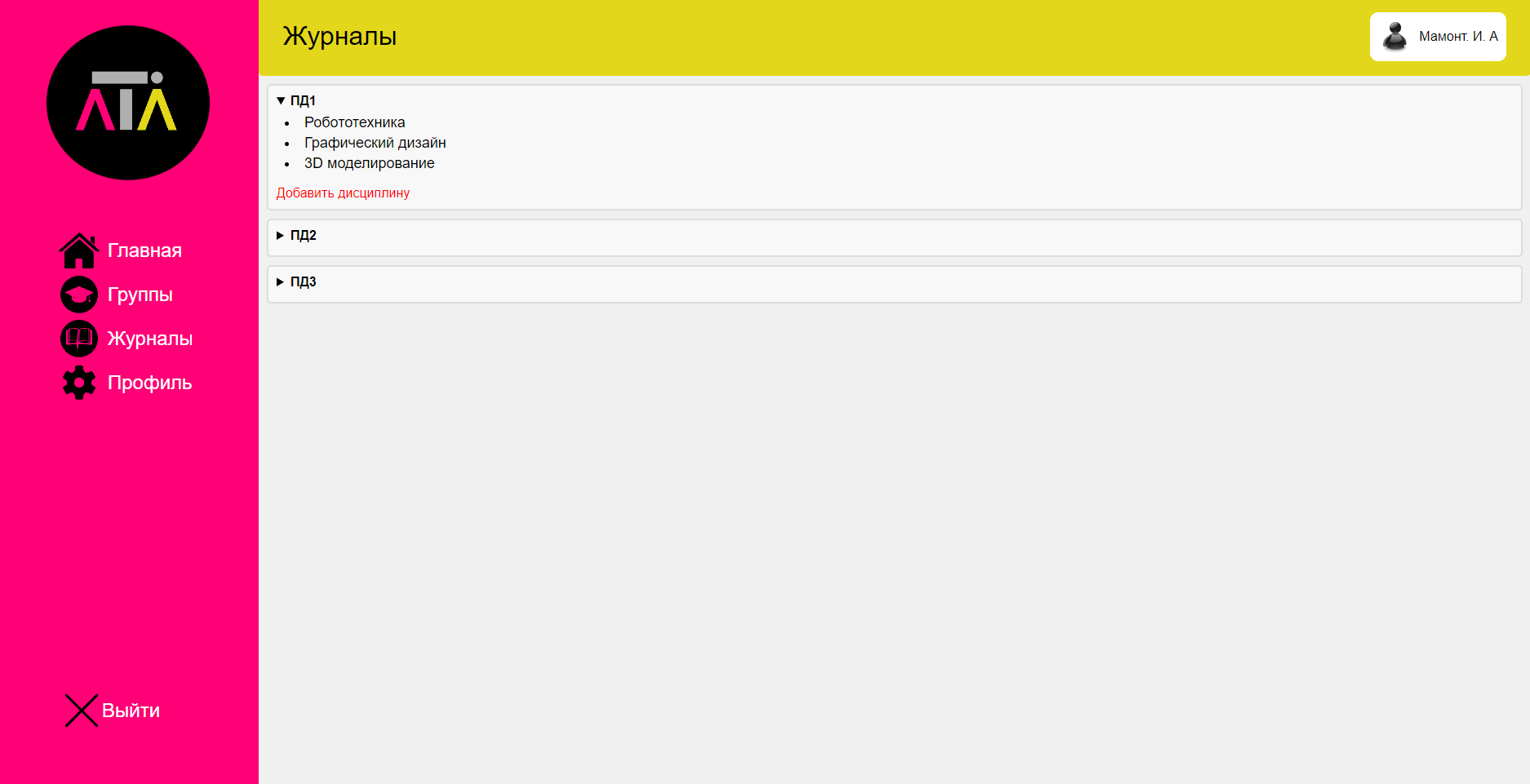


Рисунок 1.12 – страница журнала.



Рисунок 1.14 – страница «Cам журнал».

На данном этапе разберем, то как будем вести себя система при выполнение выделенных модулей. Для этого мы будем использовать диаграммы состояний в нотации UML.

На рисунке 1.16 представлена общая диаграмма состояний. При переходе пользователем в программу пользователь будет находиться в том, состоянии, когда ему необходимо авторизоваться в системе, чтобы продолжить работу с системой и осуществлять деятельность согласно выделенным вариантам использования.

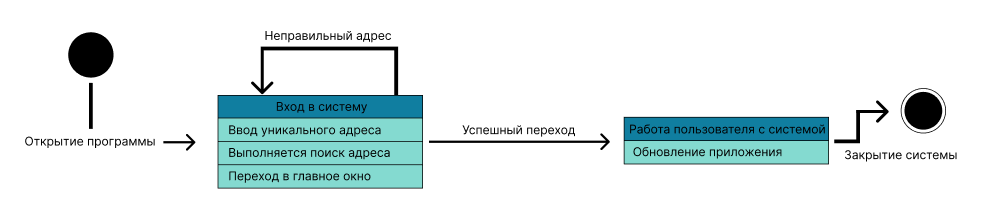


Рисунок 1.16 – Диаграмма состояний добавление данных

На рисунке 1.17 изображена декомпозиция состояния «Редактирования».

Сотрудник должен нажать на кнопку «Редактировать». После этого приложение обновится и обновит данные в базе данных. Далее сотрудник может просмотреть все изменённые данные.

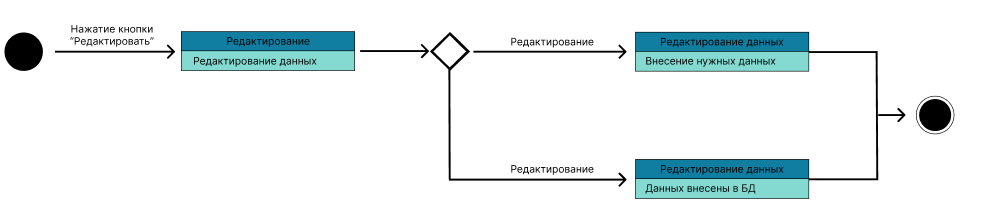


Рисунок 1.17 – Диаграмма состояний редактирования данных

# 2. ПРОЕКТНАЯ ЧАСТЬ

## 2.1 Техническое задание

2.2 Наименование предприятий разработчика и заказчика системы, их реквизиты

Полное наименование заказчика: IT лицей лидер по ул. Гоголя, 53, Белореченск • этаж 2

Сокращенное название: IT лицей лидер

Юридический и почтовый адрес: 352625, ул. Гоголя, 53, Белореченск

Контактный телефон: +7-(800)-770-03-07

ИНН: 236800273267

ОГРНИП: 323237500516223

ОКАТО 03208804001

ОКПО 98815633

ОКОГУ 49007

ОКФС 14

ОКОПФ 72

ФСС 2322725732

ПФР 033-033-016153

ОКТМО 03608404

Полное наименование разработчика: Студент группы ИС-4, ЧУПОО Техникума «Бизнес и Право», Грибенюков К.Н.

Краткое наименование: Грибенюков К.Н.

Юридический и почтовый адрес: 352625, Краснодарский край, город Белореченск, ул. Чапаева, 48.

Контактный телефон: +7 (988) 559-35-31

Наполнение различных сайтов, функционирование которых поддерживается одной и той же инсталляцией системы, должно храниться под управлением единой СУБД.

Система управления контентом должна иметь стандартный для Windows интерфейс, отвечающий следующим требованиям:

* реализация в графическом оконном режиме;
* единый стиль оформления;
* интуитивно понятное назначение элементов интерфейса;
* отображение на экране только тех возможностей, которые доступны конкретному пользователю;
* отображение на экране только необходимой для решения текущей прикладной задачи информации;
* отображение на экране хода длительных процессов обработки;
* диалог с пользователем должен быть оптимизирован для выполнения типовых и часто используемых операций;
* для операций по массовому вводу информации должна быть предусмотрена минимизация количества нажатий на клавиатуру для выполнения стандартных действий.

HTML (от английского HyperText Markup Language) — это язык гипертекстовой разметки текста. Он нужен, чтобы размещать на веб-странице элементы: текст, картинки, таблицы и видео.

HTML состоит из тегов — команд, которые указывают браузеру, как отображать помещённый в них текст. Это и есть элементы веб-страницы. У каждого тега есть имя, которое заключается в угловые скобки <и>.

Для реализации интерактивных элементов сервероной части должны использоваться языки php, javascript.

Для реализации динамических страниц должен использоваться язык PHP.

Объем одной стандартной загружаемой страницы сайта в среднем не должен превышать 170 kb.

Объем flash-заставки не должен превышать 300 Kb.

CSS (Cascading Style Sheets, каскадные таблицы стилей) — язык описания внешнего вида HTML-документа. Это одна из базовых технологий в современном интернете. Практически ни один сайт не обходится без CSS, поэтому HTML и CSS действуют в единой связке.

Каскадные таблицы стилей работают с HTML, но это совершенно другой язык. HTML структурирует документ и упорядочивает информацию, а CSS взаимодействует с браузером, чтобы придать документу оформление.

## 2.3 Программное обеспечение информационной системы

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире. Данная СУБД подходит для самых различных проектов: от небольших приложений до больших высоконагруженных проектов.

SQL Server был создан компанией Microsoft. Первая версия вышла в 1987 году. А текущей версией является версия 2022, которая вышла в ноябре 2022 году и которая будет использоваться в текущем руководстве.

SQL Server долгое время был исключительно системой управления базами данных для Windows, однако начиная с версии 16 эта система доступна и на Linux.

Необходимое программное обеспечение для персонального компьютера:

- Операционная система: Windows 10/Linux/macOS

Сайт должен быть доступен для полнофункционального просмотра с помощью следующих браузеров:

– Opera 6.0 и выше;

– Google;

– Yandex.

JavaScript - предназначен для написания сценариев для активных HTML-страниц. Язык JavaScript не имеет никакого отношения к языку Java. Java разработан фирмой SUN. JavaScript - фирмой Netscape Communication Corporation. Первоначальное название - LiveScript. После завоевания языком Java всемирной известности LiveScript из коммерческих соображений переименовали в JavaScript.

2.4 Руководство пользователя

Авторизация и Регистрация: для доступа к профилю пользователя и просмотра контента сайта необходимо авторизироваться, после этого вы сможете просматривать контент сайта.

Выход из Аккаунта:

В главном меню в левом поле находится кнопка «Выйти». Чтобы выйти из аккаунта нужно нажать на нее.

2.5 Руководство администратора

Авторизация: для доступа к профилю администратора и редактирования данных сайта необходимо авторизироваться, после этого вы сможете просматривать и редактировать контент сайта.

Добавление оценок:

Чтобы добавить оценку, нажмите на поле с оценкой и впишите нужную

Редактирование оценки:

Чтобы редактировать оценки, на странице «Журнала» выберите нужное поле с оценкой и нажмите после впишите нужную и обновите страницу

# ЗАКЛЮЧЕНИЕ

При разработке электронного журнала для IT Лицей лидер были достигнуты следующие цели производственной практики:

1. Разработан интерфейс для сайта «IT лицей лидер», обеспечивающий легкость в использовании для пользователей.
2. Реализована функциональность регистрации и авторизации пользователей, для администратора также имеется возможность удаления, добавления и редактирования новостей, контактов, книг, информации о нас, удаления пользователей.
3. Создана трёхуровневая архитектура приложения, обеспечивающая эффективное управление данными и логикой приложения.

Цель работы достигнута, а именно создан веб-сайт для электронного журнала «IT лицей лидер», который предоставляет следующие функции:

* Возможность создания и редактирования оценок.
* Просмотр контента веб-сайта.

Этот веб-сайт включает следующие технологии:

* Языки программирования: PHP, HTML, Bootstrap, CSS, JavaScript.
* Локальный сервер xampp
* Система управления реляционными базами данных: MySQL.

Вся информация, материалах и пользователях хранится в базе данных, обеспечивая легкость доступа и управления контентом электронного журнала.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Вандюк, Джон К. Руководство по разработке системы управления сайтом / Вандюк, Джон К., Мэтт Вестгейт,. - М.: Вильямс, 2019. - 400 c.
2. Гаевский, А.Ю. 100% самоучитель. Создание Web-страниц и Web-сайтов. HTML и JavaScript / А.Ю. Гаевский, В.А. Романовский. - М.: Триумф, 2020. - 464 c.
3. Дакетт, Джон HTML и CSS. Разработка и дизайн веб-сайтов / Джон Дакетт. - Москва: Наука, 2019. - 480 c.
4. Дронов, В. JavaScript в Web-дизайне / В. Дронов. - М.: СПб: БХВ, 2019. - 880 c.
5. Дронов, В. PHP, MySQL и Dreamweaver. Разработка интерактивных Web-сайтов / В. Дронов. - М.: БХВ-Петербург, 2019. - 480 c.
6. Дронов, В.А. PHP 5/6, MySQL 5/6 и Dreamweaver CS4. Разработка интерактивных Web-сайтов / В.А. Дронов. - М.: БХВ-Петербург, 2019. - 820 c.
7. Дронов, Владимир JavaScript и AJAX в Web-дизайне / Владимир Дронов. - Москва: Высшая школа, 2019. - 736 c.
8. Дронов, Владимир PHP, MySQL и Dreamweaver MX 2004. Разработка интерактивных Web-сайтов / Владимир Дронов. - М.: "БХВ-Петербург", 2019. - 448 c.
9. Колисниченко, Д. PHP 5/6 и MySQL 6. Разработка Web-приложений / Д. Колисниченко. - М.: БХВ-Петербург, 2021. - 560 c.
10. Колисниченко, Д.Н. PHP 5/6 и MySQL 6. Разработка Web-приложений / Д.Н. Колисниченко. - М.: БХВ-Петербург, 2021. - 364 c.

# ПРИЛОЖЕНИЕ

Functions.php

<?php

include('../Bd/pdo.php');

include('../Bd/brain.php');

if(isset($\_POST['submit\_student'])){

global $connection;

$name = $\_POST['name'];

$lastname = $\_POST['lastname'];

$surname = $\_POST['surname'];

$group\_id = getIdGroup($\_POST['group']);

$query = $connection->prepare("INSERT INTO student (name, lastname, surname, group\_id) VALUES ( :name, :lastname, :surname, :groupId)");

$query->bindParam(":name", $name, PDO::PARAM\_STR);

$query->bindParam(":lastname", $lastname, PDO::PARAM\_STR);

$query->bindParam(":surname", $surname, PDO::PARAM\_STR);

$query->bindParam(":groupId", $group\_id, PDO::PARAM\_STR);

$query->execute();

header("Location: Group.php");

exit;

} else if(isset($\_POST['submit\_group'])){

global $connection;

$groupName = $\_POST['group\_name'];

$query = $connection->prepare("INSERT INTO groups (name) VALUES ( :name )");

$query->bindParam(":name", $groupName, PDO::PARAM\_STR);

$query->execute();

header("Location: Group.php");

exit;

} else if(isset($\_POST['submit\_subject'])) {

global $connection;

$name = $\_POST['name'];

$group\_id = getIdGroup($\_POST['group']);

$query = $connection->prepare("INSERT INTO subject (name, group\_id) VALUES ( :name, :group)");

$query->bindParam(":name", $name, PDO::PARAM\_STR);

$query->bindParam(":group", $group\_id, PDO::PARAM\_INT);

$query->execute();

header("Location: Tables.php");

exit;

} else if(isset($\_POST['student\_id'])){

global $connection;

$id = $\_POST['student\_id'];

try{

$sql = "DELETE FROM grade WHERE student\_id = :studentId;";

$query = $connection->prepare($sql);

$query->bindParam(":studentId", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM student WHERE id = :studentId;";

$query = $connection->prepare($sql);

$query->bindParam(":studentId", $id, PDO::PARAM\_INT);

$query->execute();

} catch (PDOException $e) {

header('Content-Type: application/json');

echo json\_encode(['message' => $e->getMessage()]);

}

exit;

} else if(isset($\_POST['group'])) {

global $connection;

$id = getIdGroup($\_POST['group']);

try {

$sql = "DELETE FROM grade WHERE student\_id IN (SELECT student\_id FROM student WHERE group\_id = :group\_id)";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM student WHERE group\_id = :group\_id";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM subject WHERE group\_id = :group\_id";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM groups WHERE id = :group\_id";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

} catch (PDOException $e) {

header('Content-Type: application/json');

echo json\_encode(['message' => $e->getMessage()]);

}

exit;

} else {

echo '<script type="text/javascript">window.location.href = document.referrer;</script>';

}

GradeBd.php

<?php

include('../Bd/pdo.php');

include('../Bd/brain.php');

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

$jsonData = json\_decode($\_POST['jsonData'], true);

try {

global $connection;

$gradesData = $jsonData;

$subjectName = $gradesData['subject'];

$subjectId = getIdSubject($subjectName);

$studentId = getIdStudent($gradesData['name']);

$key = $gradesData['day'];

$grade = $gradesData['grade'];

if(!checkRepeat($studentId, $subjectId, $key)) {

$query = $connection->prepare("INSERT INTO grade (grade, subject\_id, student\_id, lesson\_num) VALUES ( :grade, :subject\_id, :student\_id, :lesson\_num)");

$query->bindParam(":grade", $grade, PDO::PARAM\_STR);

$query->bindParam(":subject\_id", $subjectId, PDO::PARAM\_INT);

$query->bindParam(":student\_id", $studentId, PDO::PARAM\_INT);

$query->bindParam(":lesson\_num", $key, PDO::PARAM\_INT);

$query->execute();

} else {

$id = getIdRepeat($studentId, $subjectId, $key);

$query = $connection->prepare("UPDATE grade SET grade = :grade WHERE id = :id");

$query->bindParam(':grade', $grade);

$query->bindParam(':id', $id);

$query->execute();

}

} catch (PDOException $e) {

$error = $e->getMessage();

$response = [

'data' => $gradesData,

'test' => 1,

'error' => $error,

];

header('Content-Type: application/json');

echo json\_encode($response);

}

} else {

echo 'Неверный метод запроса';

}

?>

Group.php

<?php

session\_start();

if(!isset($\_SESSION['user\_id']) || isset($\_GET['logout'])){

$\_SESSION['user\_id'] = null;

header('Location: login.php');

exit;

} else {

include('../Bd/pdo.php');

include('../Bd/brain.php');

$role = checkRoleUser($\_SESSION['user\_id']);

$\_SESSION['role'] = $role;

}

?>

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link rel="stylesheet" href="../css/Style.css">

<title>Главная</title>

</head>

<body>

<style>

.main\_podval{

display: flex;

align-items: center;

color: white;

font-size: 24px;

margin: 350px 48px;

fill: white;

width: 50px;

padding: 5 10px;

}

</style>

<div class="main-design container">

<div class="left-part" id="left\_part">

<a href="Main.php"><div class="logo"><img src="../img/Group.svg" alt=""></div></a>

<div class="main\_menu">

<!-- <a href="Main.php"><div><img src="../data/Домашняя.svg" alt="">Главная</div></a> -->

<a href="Main.php"><div><img src="../img/Главная.svg" alt="">Главная</div></a>

<a href="Group.php"><div><img src="../img/Группы.svg" alt="">Группы</div></a>

<a href="Tables.php"><div><img src="../img/Журнал.svg" alt="">Журналы</div></a>

<a href="Profile.php"><div><img src="../img/Настройки.svg" alt="">Профиль</div></a></div>

<a href="?logout=1" class="logout"><div class="main\_podval"><img src="../img/Выйти.svg" alt="">Выйти</div></a>

</div>

<div class="right-part" id="right\_part">

<div class="part\_header">

<?php

global $connection;

$info = getUserInfoById($\_SESSION['user\_id']);

?>

<span>Список групп</span>

<div class="logUserInformation">

<img src="../img/user\_img.png" alt="" width="40">

<label><?php echo $info['lastname'].'. '.mb\_substr($info['name'], 0, 1).'. '.mb\_substr($info['surname'], 0, 1)?> </label>

</div>

</div>

<div class="part\_bottom">

<aside>

<?php

$groupData = getAllGroup();

foreach ($groupData as $group){

echo "

<details class='group-details'>

<summary class='group-summary'>".$group['name']."</summary>";

echo "<div class=details\_list>";

$studentData = getAllStudentByGroup($group['name']);

foreach ($studentData as $student){

echo "<li class='delete-student' title='Удалить' data-student-id=".$student['id'].">".$student['lastname']." ".$student['name']." ".$student['surname']."</li>";

}

if($role) {

echo "<a onclick='toggleAddStudentBlock()' class='add\_things'>Добавить студента</a>";

echo "<a onclick='toggleDeleteGroup()' class='add\_things'>Удалить Группу</a>";

}

echo "</div></details>";

}

if($role){

echo '<a onclick="toggleAddGroupBlock()" class="groupAdd">Добавить группу</a>';

}

?>

</aside>

</div>

</div>

<div id="addGroupBlock" class="add-block">

<form action="functions.php" method="post" class="form-add">

<div class="form-group">

<label for="groupName">Название группы:</label>

<input type="text" id="groupName" name="group\_name" required>

</div>

<div>

<button type="submit" name="submit\_group">Добавить группу</button>

<button onclick="toggleAddGroupBlock()">Отмена</button>

</div>

</form>

</div>

<!-- Добавление студентов -->

<div id="addStudentBlock">

<form action="functions.php" method="post" class="form-add">

<div class="form-group">

<label for="firstName">Имя:</label>

<input type="text" id="firstName" name="name" required>

</div>

<div class="form-group">

<label for="lastName">Фамилия:</label>

<input type="text" id="lastName" name="lastname" required>

</div>

<div class="form-group">

<label for="middleName">Отчество:</label>

<input type="text" id="middleName" name="surname" required>

</div>

<div class="form-group">

<label for="group">Группа:</label>

<select id="group" name="group" required>

<?php

// Пример массива групп

$groups = getAllGroup();

// Заполнение выпадающего списка групп

foreach ($groups as $group) {

echo '<option value="' . $group['name'] . '">' . $group['name'] . '</option>';

}

?>

</select>

</div>

<button type="submit" name="submit\_student">Добавить</button>

<button onclick="toggleAddStudentBlock()">Отмена</button>

</form>

</div>

</div>

<script>

function confirmDeleteStudent(link) {

var studentId = link.dataset.studentId;

var confirmDelete = confirm('Вы уверены, что хотите удалить студента? ');

if (confirmDelete) {

deleteStudent(studentId);

} else {

}

}

function deleteStudent(studentId) {

// Отправьте POST-запрос на сервер с использованием fetch или другого метода

fetch('functions.php', {

method: 'POST',

headers: {

'Content-Type': 'application/x-www-form-urlencoded',

},

body: 'student\_id=' + encodeURIComponent(studentId),

})

.then(response => response.json())

.then(data => {

// Обработка ответа от сервера (если необходимо)

alert(data.message);

})

.catch(error => {

console.error('Ошибка:', error);

});

setTimeout(function() {

window.location.href = 'functions.php';

}, 500);

}

<?php

if($role){

echo "document.addEventListener('DOMContentLoaded', function () {

var deleteLinks = document.querySelectorAll('.delete-student');

deleteLinks.forEach(function (link) {

link.addEventListener('click', function (event) {

event.preventDefault();

confirmDeleteStudent(link);

});

});

});";}

?>

</script>

<script>

var detailsElements = document.querySelectorAll('.group-details');

var selected = document.getElementById('group');

var group;

detailsElements.forEach(function(detailsElement) {

var summaryText = detailsElement.querySelector('.group-summary');

detailsElement.addEventListener('toggle', function() {

if (detailsElement.open) {

var text = summaryText.textContent;

group = text;

console.log(group);

selected.value = group;

}

});

});

function toggleAddStudentBlock() {

var addStudentBlock = document.getElementById("addStudentBlock");

var blur1 = document.getElementById("left\_part");

var blur2 = document.getElementById("right\_part");

// Изменяем стиль фильтра для эффекта размытия

blur1.style.filter = (blur1.style.filter === "blur(3px)") ? "none" : "blur(3px)";

blur2.style.filter = (blur2.style.filter === "blur(3px)") ? "none" : "blur(3px)";

// Изменяем видимость блока

addStudentBlock.style.display = (addStudentBlock.style.display === "block") ? "none" : "block";

}

function toggleAddGroupBlock() {

var addGroupBlock = document.getElementById("addGroupBlock");

var blur1 = document.getElementById("left\_part");

var blur2 = document.getElementById("right\_part");

// Изменяем стиль фильтра для эффекта размытия

blur1.style.filter = (blur1.style.filter === "blur(3px)") ? "none" : "blur(3px)";

blur2.style.filter = (blur2.style.filter === "blur(3px)") ? "none" : "blur(3px)";

// Изменяем видимость блока

addGroupBlock.style.display = (addGroupBlock.style.display === "block") ? "none" : "block";

}

function toggleDeleteGroup(){

var confirmDelete = confirm('Вы уверены, что хотите удалить группу?' + ' ' + group + '?');

if (confirmDelete) {

fetch('functions.php', {

method: 'POST',

headers: {

'Content-Type': 'application/x-www-form-urlencoded',

},

body: 'group=' + group,

});

setTimeout(function() {

window.location.href = 'functions.php';

}, 500);

}

}

</script>

</body>

</html>

Login.php

<?php

session\_start();

if(isset($\_SESSION['user\_id'])){

header('Location: Group.php');

exit;

}?>

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link rel="stylesheet" href="../css/Style.css">

<title>Авторизация</title>

<?php

if(isset($\_SESSION["error"])){

echo "<style>.log\_input input{border-bottom:2px solid red;}</style>";}?>

</head>

<body>

<div class="log\_background">

<div>

<form action="" method="post" class="log\_form">

<div><h2>Авторизация</h2></div>

<div class="log\_input">

<input type="text" name="username" placeholder="Введите логин" required>

<input type="password" name="password" placeholder="Введите пароль" required>

</div>

<div>

<button type="submit" name="login">Войти</button>

</div>

<div><a href="registration.php">Нету аккаунта? Зарегестрироваться</a></div>

</form>

</div>

<?php

include('../Bd/pdo.php');

include('../Bd/brain.php');

if (isset($\_POST['login'])) {

$error = null;

$username = $\_POST['username'];

$password = $\_POST['password'];

$query = $connection->prepare("SELECT \* FROM users WHERE username=:username");

$query->bindParam("username", $username, PDO::PARAM\_STR);

$query->execute();

$result = $query->fetch(PDO::FETCH\_ASSOC);

if ($result) {

if (password\_verify($password, $result['password'])) {

$\_SESSION['user\_id'] = $result['id'];

header('Location: Group.php');}else{

$error = "password\_error";

$\_SESSION['error'] = $error;}}?></div></body>

</html>

Main.php

<?php

session\_start();

if(!isset($\_SESSION['user\_id']) || isset($\_GET['logout'])){

$\_SESSION['user\_id'] = null;

header('Location: login.php');

exit;

} else {

include('../Bd/pdo.php');

include('../Bd/brain.php');

$role = checkRoleUser($\_SESSION['user\_id']);

$\_SESSION['role'] = $role;

}?>

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link rel="stylesheet" href="../css/Style.css">

<title>Главная</title>

</head>

<body>

<style>

.main\_podval{

display: flex;

align-items: center;

color: white;

font-size: 24px;

margin: 350px 48px;

fill: white;

width: 50px;

padding: 5 10px;

}

</style>

<div class="main-design container">

<div class="left-part">

<a href="Main.php"><div class="logo"><img src="../img/Group.svg" alt=""></div></a>

<div class="main\_menu">

<!-- <a href="Main.php"><div><img src="../img/Домашняя.svg" alt="">Главная</div></a> -->

<a href="Main.php"><div><img src="../img/Главная.svg" alt="">Главная</div></a>

<a href="Group.php"><div><img src="../img/Группы.svg" alt="">Группы</div></a>

<a href="Tables.php"><div><img src="../img/Журнал.svg" alt="">Журналы</div></a>

<a href="Profile.php"><div><img src="../img/Настройки.svg" alt="">Профиль</div></a> </div>

<a href="?logout=1" class="logout"><div class="main\_podval"><img src="../img/Выйти.svg" alt="">Выйти</div></a>

</div>

<div class="right-part">

<div class="part\_header">

<span>Электронный журнал</span>

<?php

$info = getUserInfoById($\_SESSION['user\_id']);

?>

<div class="logUserInformation">

<img src="../img/user\_img.png" alt="" width="40">

<label><?php echo $info['lastname'].'. '.mb\_substr($info['name'], 0, 1).'. '.mb\_substr($info['surname'], 0, 1)?> </label>

</div>

</div>

<div class="part\_bottom">

<div class="bottom\_div">

</div>

<h2 align=center>IT - Лицей Лидер</h2>

<div class="description center">

<p>Это реальность, которую мы создаем в IT-Лицей Лидер . Мы верим, что каждый ребенок имеет право на качественное образование и возможность реализовать свой потенциал. Мы используем инновационные методики обучения, которые помогают детям развивать их навыки и интерес к технологиям. Выбирая IT-Лицей Лидер , вы принимаете решение о качественном образовании для вашего ребенка! Наш лицей - это место, где дети обучаются, развиваются и достигают успеха.

<br><br>В этом электронном дневнике вы можете оценить успеваемость ваших деток.

</div>

</div>

</body>

</html>

PostFunction.php

<?php

include('../Bd/pdo.php');

include('../Bd/module\_global.php');

if(isset($\_POST['submit\_student'])){

global $connection;

$name = $\_POST['name'];

$lastname = $\_POST['lastname'];

$surname = $\_POST['surname'];

$group\_id = getIdGroup($\_POST['group']);

$query = $connection->prepare("INSERT INTO student (name, lastname, surname, group\_id) VALUES ( :name, :lastname, :surname, :groupId)");

$query->bindParam(":name", $name, PDO::PARAM\_STR);

$query->bindParam(":lastname", $lastname, PDO::PARAM\_STR);

$query->bindParam(":surname", $surname, PDO::PARAM\_STR);

$query->bindParam(":groupId", $group\_id, PDO::PARAM\_STR);

$query->execute();

header("Location: Group.php");

exit;

} else if(isset($\_POST['submit\_group'])){

global $connection;

$groupName = $\_POST['group\_name'];

$query = $connection->prepare("INSERT INTO groups (name) VALUES ( :name )");

$query->bindParam(":name", $groupName, PDO::PARAM\_STR);

$query->execute();

header("Location: Group.php");

exit;

} else if(isset($\_POST['submit\_subject'])) {

global $connection;

$name = $\_POST['name'];

$group\_id = getIdGroup($\_POST['group']);

$query = $connection->prepare("INSERT INTO subject (name, group\_id) VALUES ( :name, :group)");

$query->bindParam(":name", $name, PDO::PARAM\_STR);

$query->bindParam(":group", $group\_id, PDO::PARAM\_INT);

$query->execute();

header("Location: Tables.php");

exit;

} else if(isset($\_POST['student\_id'])){

global $connection;

$id = $\_POST['student\_id'];

try{

$sql = "DELETE FROM grade WHERE student\_id = :studentId;";

$query = $connection->prepare($sql);

$query->bindParam(":studentId", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM student WHERE id = :studentId;";

$query = $connection->prepare($sql);

$query->bindParam(":studentId", $id, PDO::PARAM\_INT);

$query->execute();

} catch (PDOException $e) {

header('Content-Type: application/json');

echo json\_encode(['message' => $e->getMessage()]);

}

exit;

} else if(isset($\_POST['group'])) {

global $connection;

$id = getIdGroup($\_POST['group']);

try {

$sql = "DELETE FROM grade WHERE student\_id IN (SELECT student\_id FROM student WHERE group\_id = :group\_id)";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM student WHERE group\_id = :group\_id";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM subject WHERE group\_id = :group\_id";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

$sql = "DELETE FROM groups WHERE id = :group\_id";

$query = $connection->prepare($sql);

$query->bindParam(":group\_id", $id, PDO::PARAM\_INT);

$query->execute();

} catch (PDOException $e) {

header('Content-Type: application/json');

echo json\_encode(['message' => $e->getMessage()]);}exit;

} else {

echo '<script type="text/javascript">window.location.href = document.referrer;</script>';}

Profile.php

<?php

session\_start();

if (!isset($\_SESSION['user\_id']) || isset($\_GET['logout'])) {

$\_SESSION['user\_id'] = null;

header('Location: login.php');

exit;

} else {

include('../Bd/pdo.php');

include('../Bd/brain.php');

function clearSessionMessages() {

unset($\_SESSION['error\_message']);

unset($\_SESSION['success\_message']);

}

$role = checkRoleUser($\_SESSION['user\_id']);

$\_SESSION['role'] = $role;

$info = getUserInfoById($\_SESSION['user\_id']);

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

// Изменение пароля

if (isset($\_POST['change\_password'])) {

$old\_password = $\_POST['old\_password'];

$new\_password = $\_POST['new\_password'];

$confirm\_password = $\_POST['confirm\_password'];

if (empty($old\_password) || empty($new\_password) || empty($confirm\_password)) {

$\_SESSION['error\_message'] = 'Все поля должны быть заполнены';

header('Location: profile.php');

exit;

}

if ($new\_password !== $confirm\_password) {

$\_SESSION['error\_message'] = 'Новый пароль и подтверждение нового пароля не совпадают';

header('Location: profile.php');

exit;

}

$user\_id = $\_SESSION['user\_id'];

$user\_info = getUserInfoById($user\_id); // Предполагается, что эта функция возвращает массив с данными пользователя, включая пароль

if (!password\_verify($old\_password, $user\_info['password'])) {

$\_SESSION['error\_message'] = 'Старый пароль неверен';

header('Location: profile.php');

exit;

}

$new\_password\_hash = password\_hash($new\_password, PASSWORD\_DEFAULT);

$stmt = $pdo->prepare('UPDATE users SET password = :new\_password WHERE id = :user\_id');

$stmt->execute(['new\_password' => $new\_password\_hash, 'user\_id' => $user\_id]);

$\_SESSION['success\_message'] = 'Пароль успешно изменен';

header('Location: profile.php');

exit;

}

// Изменение имени, фамилии и отчества

if (isset($\_POST['change\_info'])) {

$name = $\_POST['name'];

$surname = $\_POST['surname'];

$lastname = $\_POST['lastname'];

$stmt = $pdo->prepare('UPDATE users SET name = :name, surname = :surname, lastname = :lastname WHERE id = :user\_id');

$stmt->execute(['name' => $name, 'surname' => $surname, 'lastname' => $lastname, 'user\_id' => $\_SESSION['user\_id']]);

$\_SESSION['success\_message'] = 'Информация о пользователе успешно обновлена';

header('Location: profile.php');

exit;

}

}

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Профиль</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">

</head>

<style>

body {

background-image: url('img/background.jpg');

background-size: cover; /\* Покрывает весь экран, сохраняя пропорции \*/

background-position: center; /\* Центрирует изображение \*/

background-repeat: no-repeat; /\* Предотвращает повторение изображения \*/

}

.form-control {

display: block;

margin: 0 auto;

width: 250px;

}

.profile-container {

margin-top: 50px;

text-align: center;

}

.profile-image {

width: 50px;

height: 50px;

border-radius: 50%;

object-fit: cover;

display: block;

margin: 0 auto;

}

.profile-info {

margin-top: 20px;

}

.profile-info h2 {

font-size: 24px;

}

.profile-info p {

font-size: 16px;

}

.alert {

margin-top: 10px;

padding: 10px;

border: 1px solid transparent;

border-radius: 5px;

width: 300px;

margin: 0 auto;

}

.alert-danger {

background-color: #f8d7da;

color: #721c24;

border-color: #f5c6cb;

}

.alert-success {

background-color: #d4edda;

color: #155724;

border-color: #c3e6cb;

}

a:hover {

color: blue;

}

.close {

position: absolute;

top: 50%;

right: 10px;

transform: translateY(-50%);

cursor: pointer;

}

</style>

</style>

<body>

<?php if (isset($\_SESSION['error\_message'])): ?>

<div class="alert alert-danger">

<?php echo $\_SESSION['error\_message']; ?>

<span class="close">&times;</span>

</div>

<?php clearSessionMessages(); ?>

<?php endif; ?>

<?php if (isset($\_SESSION['success\_message'])): ?>

<div class="alert alert-success" id="successAlert">

<?php echo $\_SESSION['success\_message']; ?>

<span class="close">&times;</span>

</div>

<?php clearSessionMessages(); ?>

<?php endif; ?>

<div class="container profile-container">

<img class="profile-image" src="../img/user\_img.png" alt="Profile Image">

<div class="profile-info">

<h2><?php echo $info['name'] . ' ' . $info['surname']; ?></h2>

<?php if (isset($\_SESSION['error\_message'])): ?>

<div class="alert alert-danger">

<?php echo $\_SESSION['error\_message']; ?>

<span class="close">&times;</span>

</div>

<?php endif; ?>

<?php if (isset($\_SESSION['success\_message'])): ?>

<div class="alert alert-success" id="successAlert">

<?php echo $\_SESSION['success\_message']; ?>

<span class="close">&times;</span>

</div>

<?php endif; ?>

<form method="post" action="profile.php">

<div class="form-group">

<label for="name">Имя</label>

<input type="text" class="form-control" id="name" name="name" value="<?php echo $info['name']; ?>">

</div>

<div class="form-group">

<label for="surname">Фамилия</label>

<input type="text" class="form-control" id="surname" name="surname" value="<?php echo $info['surname']; ?>">

</div>

<div class="form-group">

<label for="lastname">Отчество</label>

<input type="text" class="form-control" id="lastname" name="lastname" value="<?php echo $info['lastname']; ?>">

</div>

<button type="submit" class="btn btn-primary" name="change\_info">Изменить информацию</button>

</form>

<br>

<form method="post" action="profile.php">

<div class="form-group">

<label for="old\_password">Старый пароль</label>

<input type="password" class="form-control" id="old\_password" name="old\_password" required>

</div>

<div class="form-group">

<label for="new\_password">Новый пароль</label>

<input type="password" class="form-control" id="new\_password" name="new\_password" required>

</div>

<div class="form-group">

<label for="confirm\_password">Подтверждение нового пароля</label>

<input type="password" class="form-control" id="confirm\_password" name="confirm\_password" required>

</div>

<button type="submit" class="btn btn-primary" name="change\_password">Сменить пароль</button>

</form><br>

<a href="main.php"><div>Вернуться</div></a>

</div>

</div>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<script>

$(document).ready(function() {

$('.alert .close').click(function() {

$(this).closest('.alert').hide();

});

});

</script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"></script>

</body>

</html>

Registratuin.php

<?php

session\_start();

if(isset($\_SESSION['user\_id'])){

header('Location: main.php');

exit;

}

include('../Bd/pdo.php');

include('../Bd/brain.php');

if (isset($\_POST['register'])) {

if ($\_POST['captcha'] !== $\_SESSION['captcha\_result']) {

echo '<script>alert("Неправильная капча!");</script>';

generateCaptcha();

} else {

$username = $\_POST['username'];

$name = $\_POST['name'];

$lastname = $\_POST['lastname'];

$surname = $\_POST['surname'];

$password = $\_POST['password'];

$password\_hash = password\_hash($password, PASSWORD\_BCRYPT);

// Validate username (at least 4 characters)

if (!preg\_match('/^.{4,}$/u', $username)) {

echo '<script>alert("Имя пользователя должно содержать не менее 4 символов!");</script>';

generateCaptcha();

exit;

}

// Validate name, lastname, and surname (only Russian characters)

if (!preg\_match('/^[а-яА-ЯёЁ\s]+$/u', $name) ||

!preg\_match('/^[а-яА-ЯёЁ\s]+$/u', $lastname) ||

!preg\_match('/^[а-яА-ЯёЁ\s]+$/u', $surname)) {

echo '<script>alert("Имя, фамилия и отчество должны содержать только кириллицу!");</script>';

generateCaptcha();

exit;

}

// Validate password (at least 4 characters)

if (!preg\_match('/^.{4,}$/u', $password)) {

echo '<script>alert("Пароль должен содержать не менее 4 символов!");</script>';

generateCaptcha();

exit;

}

// Rest of your code...

// Check if the username already exists

$query = $connection->prepare("SELECT \* FROM users WHERE username=:username");

$query->bindParam("username", $username, PDO::PARAM\_STR);

$query->execute();

if ($query->rowCount() > 0) {

echo '<script>alert("Этот адрес уже зарегистрирован!");</script>';

generateCaptcha();

exit;

}

// If the username is unique, insert the new user

$query = $connection->prepare("INSERT INTO users(username, name, lastname, surname, password) VALUES (:username, :name, :lastname, :surname, :password\_hash)");

$query->bindParam("username", $username, PDO::PARAM\_STR);

$query->bindParam("name", $name, PDO::PARAM\_STR);

$query->bindParam("lastname", $lastname, PDO::PARAM\_STR);

$query->bindParam("surname", $surname, PDO::PARAM\_STR);

$query->bindParam("password\_hash", $password\_hash, PDO::PARAM\_STR);

$result = $query->execute();

if ($result) {

header('Location: LogIn.php');

exit;

}

}

} else {

generateCaptcha();

}

function generateRandomString($length = 6)

{

$characters = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz';

$randomString = '';

for ($i = 0; $i < $length; $i++) {

$randomString .= $characters[rand(0, strlen($characters) - 1)];

}

return $randomString;

}

function generateCaptcha()

{

$captcha\_string = generateRandomString(6);

$\_SESSION['captcha\_result'] = $captcha\_string;

}

?>

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link rel="stylesheet" href="../css/Style.css">

<title>Регистрация</title>

</head>

<body>

<div class="log\_background">

<div>

<form action="" method="post" class="log\_form">

<h2>Регистрация</h2>

<div class="log\_input">

<input type="text" name="username" placeholder="Введите логин" required>

<input type="text" name="name" placeholder="Введите имя" required>

<input type="text" name="lastname" placeholder="Введите фамилию" required>

<input type="text" name="surname" placeholder="Введите отчество" required>

<input type="password" name="password" placeholder="Введите пароль" required>

<input type="password" name="verify\_password" placeholder="Подтвердите пароль" required>

<div class="log\_input">

<label for="captcha" class="captcha\_text"><?php echo $\_SESSION['captcha\_result']; ?></label>

<input type="text" name="captcha" id="captcha" placeholder="Капча" required>

</div>

<div>

<button type="submit" name="register">Зарегистрироваться</button>

</div>

</div>

<div><a href="Login.php">Есть аккаунт? Войти</a></div>

</form>

</div>

</div>

</body>

</html>

Table.php

<?php

include('../Bd/pdo.php');

include('../Bd/brain.php');

$group = $\_GET['group'];

$subject = $\_GET['subject'];

$subjectId = getIdSubject($subject);

$query = $connection->prepare("SELECT id FROM groups WHERE name = :group");

$query->bindParam(":group", $group, PDO::PARAM\_STR);

$query->execute();

$groupId = $query->fetchColumn();

$query = $connection->prepare("SELECT \* FROM student WHERE group\_id = :groupId");

$query -> bindParam(":groupId", $groupId, PDO::PARAM\_STR);

$query -> execute();

$data = $query -> fetchAll(PDO::FETCH\_ASSOC);

session\_start();

$role = $\_SESSION['role'];

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>

<link rel="stylesheet" href="../css/Style.css">

<title><?php echo $group?></title>

</head>

<body>

<style>

.main\_podval{

display: flex;

align-items: center;

color: white;

font-size: 24px;

margin: 350px 48px;

fill: white;

width: 50px;

padding: 5 10px;

}

</style>

<div class="main-design container">

<div class="left-part">

<div class="logo"><img src="../img/Group.svg" alt=""></div>

<div class="main\_menu">

<!-- <a href="Main.php"><div><img src="../data/Домашняя.svg" alt="">Главная</div></a> -->

<a href="Main.php"><div><img src="../img/Главная.svg" alt="">Главная</div></a>

<a href="Group.php"><div><img src="../img/Группы.svg" alt="">Группы</div></a>

<a href="Tables.php"><div><img src="../img/Журнал.svg" alt="">Журналы</div></a>

<a href="Profile.php"><div><img src="../img/Настройки.svg" alt="">Профиль</div></a> </div>

<a href="?logout=1" class="logout"><div class="main\_podval"><img src="../img/Выйти.svg" alt="">Выйти</div></a>

</div>

<div class="right-part">

<div class="part\_header">

<?php

$info = getUserInfoById($\_SESSION['user\_id']);

?>

<span>Группа : <?php echo $group?></span>

<span>Предмет : <?php echo $subject?></span>

<div class="logUserInformation">

<img src="../img/user\_img.png" alt="" width="40">

<label><?php echo $info['lastname'].'. '.mb\_substr($info['name'], 0, 1).'. '.mb\_substr($info['surname'], 0, 1)?> </label>

</div>

</div>

<div class="part\_bottom">

<form id="gradeForm" action="<?= htmlspecialchars($\_SERVER["PHP\_SELF"]) ?>" method="post">

<table border="1">

<thead>

<tr>

<th>ФИО</th>

<?php for ($day = 1; $day <= 31; $day++): ?>

<th><?= $day ?></th>

<?php endfor; ?>

<th>Средний бал</th>

</tr>

</thead>

<tbody>

<?php

foreach ($data as $name) {

echo '<tr> <td>'.$name['lastname'].'</td>';

$gradeData = getGradesByStudentIdAndSubjectId($name['id'], $subjectId);

$totalGrades = 0;

$numGrades = 0;

for ($day = 1; $day <= 31; $day++) {

$found = false;

foreach ($gradeData as $grade) {

if ($grade['lesson\_num'] == $day) {

if($role){

echo '<td contenteditable="true" class="editable-cell" data-original-value="" oninput="limitInput(this, 1)">' . $grade['grade'] . '</td>';

} else {

echo '<td data-original-value="">' . $grade['grade'] . '</td>';

}

$totalGrades += intval($grade['grade']);

$numGrades++;

$found = true;

break;

}

}

if (!$found) {

if($role){

echo '<td contenteditable="true" class="editable-cell" data-original-value="" oninput="limitInput(this)"></td>';

} else {

echo '<td data-original-value=""></td>';

}

}

}

$averageGrade = $numGrades > 0 ? round($totalGrades / $numGrades, 2) : 0;

echo '<td>'.$averageGrade.'</td>';

echo '</tr>';

}

?>

</tbody>

</table>

</form>

</div>

</div>

<script>

function limitInput(element) {

var allowedCharactersRegex = /[2345НнуУ]/g;

var sanitizedText = element.textContent.match(allowedCharactersRegex);

if (sanitizedText !== null) {

element.textContent = sanitizedText.join('');

} else {

element.textContent = '';

}

var maxLength = 1;

if (element.textContent.length > maxLength) {

element.textContent = element.textContent.substring(0, maxLength);

}

}

$(document).ready(function () {

// Initialize original values

$('.editable-cell').each(function () {

$(this).attr('data-original-value', $(this).text());

});

function sendData(element) {

var cell = $(element);

var originalValue = cell.attr('data-original-value');

var currentValue = cell.text();

subjectValue = <?php echo '"'.$\_GET['subject'].'"'?>;

console.log(subjectValue);

if (originalValue !== currentValue) {

var studentDataObject = {

'name': cell.closest('tr').find('td:first-child').text(),

'day': cell.index(),

'grade': currentValue,

'subject': subjectValue

};

var jsonData = JSON.stringify(studentDataObject);

$.ajax({

type: 'POST',

url: 'GradeBd.php',

data: {jsonData: jsonData},

success: function (response) {

console.log(response);

},

error: function (error) {

console.error('Error:', error);

}

});

// Update original value after sending

cell.attr('data-original-value', currentValue);

}

}

$('.editable-cell').on('input', function () {

sendData(this);

});

});

</script>

</body>

</html>

Tables.php

<?php

session\_start();

if(!isset($\_SESSION['user\_id']) || isset($\_GET['logout'])){

$\_SESSION['user\_id'] = null;

header('Location: login.php');

exit;

} else {

include('../Bd/pdo.php');

include('../Bd/brain.php');

$role = $\_SESSION['role'];

}

?>

<!doctype html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link rel="stylesheet" href="../css/Style.css">

<title>Таблицы</title>

</head>

<body>

<style>

.main\_podval{

display: flex;

align-items: center;

color: white;

font-size: 24px;

margin: 350px 48px;

fill: white;

width: 50px;

padding: 5 10px;

}

</style>

<div class="main-design container" id="mainDesign">

<div class="left-part">

<a href="Main.php"><div class="logo"><img src="../img/Group.svg" alt=""></div></a>

<div class="main\_menu">

<!-- <a href="Main.php"><div><img src="../img/Домашняя.svg" alt="">Главная</div></a> -->

<a href="Main.php"><div><img src="../img/Главная.svg" alt="">Главная</div></a>

<a href="Group.php"><div><img src="../img/Группы.svg" alt="">Группы</div></a>

<a href="Tables.php"><div><img src="../img/Журнал.svg" alt="">Журналы</div></a>

<a href="Profile.php"><div><img src="../img/Настройки.svg" alt="">Профиль</div></a> </div>

<a href="?logout=1" class="logout"><div class="main\_podval"><img src="../img/Выйти.svg" alt="">Выйти</div></a>

</div>

<div class="right-part">

<div class="part\_header">

<?php

global $connection;

$info = getUserInfoById($\_SESSION['user\_id']);

?>

<span>Журналы</span>

<div class="logUserInformation">

<img src="../img/user\_img.png" alt="" width="40">

<label><?php echo $info['lastname'].'. '.mb\_substr($info['name'], 0, 1).'. '.mb\_substr($info['surname'], 0, 1)?> </label>

</div>

</div>

<div class="part\_bottom">

<aside>

<?php

$groupData = getAllGroup();

foreach ($groupData as $group){

echo "

<details class='group-details'>

<summary class='group-summary'>".$group['name']."</summary>";

echo "<div class=details\_list>";

$subjectData = getAllSubjectsByGroup($group['name']);

foreach ($subjectData as $student){

echo "<a href='Table.php?subject=".$student['name']."&group=".$group['name']."'><li>".$student['name']."</li></a>";

}

if($role) {

echo "<a onclick='toggleAddStudentBlock()' class='add\_things'>Добавить дисциплину</a>";

}

echo "</div></details>";

}

?>

</aside>

</div>

</div>

</div>

<!-- Добавление дисциплины -->

<div id="addStudentBlock">

<form action="functions.php" method="post" class="form-add">

<div class="form-group">

<label for="firstName">Название:</label>

<input type="text" id="firstName" name="name" required>

</div>

<div class="form-group">

<label for="group">Группа:</label>

<select id="group" name="group" required>

<?php

// Пример массива групп

$groups = getAllGroup();

// Заполнение выпадающего списка групп

foreach ($groups as $group) {

echo '<option value="' . $group['name'] . '">' . $group['name'] . '</a></option>';

}

?>

</select>

</div>

<button type="submit" name="submit\_subject">Добавить</button>

<button onclick="toggleAddStudentBlock()">Отмена</button>

</form>

</div>

<script>

var detailsElements = document.querySelectorAll('.group-details');

var selected = document.getElementById('group');

detailsElements.forEach(function(detailsElement) {

var summaryText = detailsElement.querySelector('.group-summary');

detailsElement.addEventListener('toggle', function() {

if (detailsElement.open) {

var text = summaryText.textContent;

console.log(text);

selected.value = text;

}

});

});

function toggleAddStudentBlock() {

var addStudentBlock = document.getElementById("addStudentBlock");

var body = document.getElementById("mainDesign");

// Изменяем стиль фильтра для эффекта размытия

body.style.filter = (body.style.filter === "blur(3px)") ? "none" : "blur(3px)";

// Изменяем видимость блока

addStudentBlock.style.display = (addStudentBlock.style.display === "block") ? "none" : "block";

}

</script>

</body>

</html>

Pdo.php

<?php

$pdo = new PDO('mysql:host=localhost;dbname=gradebook', 'Admin', '12345');

define('USER', 'Admin');

define('PASSWORD', '12345');

define('HOST', 'localhost');

define('DATABASE', 'gradebook');

try {

$connection = new PDO("mysql:host=".HOST.";dbname=".DATABASE, USER, PASSWORD);

} catch (PDOException $e) {

echo $e->getMessage();

exit("Error: " . $e->getMessage());

}

$stmt = $connection->prepare("SELECT \* FROM users");

$stmt->execute();

?>

Brain.php

<?php

function getUserInfoById($id){

global $connection;

$stmt = $connection->prepare("SELECT \* FROM users WHERE id = :id");

$stmt->bindParam(':id', $id, PDO::PARAM\_INT);

// Выполнение запроса

$stmt->execute();

// Получение результата

$result = $stmt->fetch(PDO::FETCH\_ASSOC);

return $result;

}

function checkRepeat($studentId, $subjectId, $key){

global $connection;

$query = $connection->prepare("SELECT id FROM grade WHERE subject\_id = :subjectId AND student\_id = :studentId AND lesson\_num = :lesson\_num");

$query->bindParam(":subjectId", $subjectId, PDO::PARAM\_INT);

$query->bindParam(":studentId", $studentId, PDO::PARAM\_INT);

$query->bindParam(":lesson\_num", $key, PDO::PARAM\_INT);

$query->execute();

if(empty($query -> fetchColumn())){

return false;

} else {

return true;

}

}

function getIdRepeat($studentId, $subjectId, $key){

global $connection;

$query = $connection->prepare("SELECT id FROM grade WHERE subject\_id = :subjectId AND student\_id = :studentId AND lesson\_num = :lesson\_num");

$query->bindParam(":subjectId", $subjectId, PDO::PARAM\_INT);

$query->bindParam(":studentId", $studentId, PDO::PARAM\_INT);

$query->bindParam(":lesson\_num", $key, PDO::PARAM\_INT);

$query->execute();

return $query->fetchColumn();

}

function getIdSubject(string $subject)

{

global $connection; // Предполагается, что $connection - это глобальная переменная или передана как аргумент

$query = $connection->prepare("SELECT id FROM subject WHERE name = :subject");

$query->bindParam(":subject", $subject, PDO::PARAM\_STR);

$query->execute();

return $query->fetchColumn();

}

function getIdStudent(string $student)

{

global $connection; // Предполагается, что $connection - это глобальная переменная или передана как аргумент

$query = $connection->prepare("SELECT id FROM student WHERE lastname = :student");

$query->bindParam(":student", $student, PDO::PARAM\_STR);

$query->execute();

$id = $query->fetchColumn();

return $id;

}

function getIdGroup(string $groupName){

global $connection;

$query = $connection->prepare("SELECT id FROM groups WHERE name = :groupName");

$query->bindParam(":groupName", $groupName, PDO::PARAM\_STR);

$query->execute();

$id = $query->fetchColumn();

return $id;

}

function getGradesByStudentIdAndSubjectId($studentId, $subjectId){

global $connection;

$query = $connection->prepare("SELECT \* FROM grade WHERE student\_id = :studentId AND subject\_id = :subjectId ORDER BY lesson\_num ASC");

$query->bindParam(":studentId", $studentId, PDO::PARAM\_STR);

$query->bindParam(":subjectId", $subjectId, PDO::PARAM\_STR);

$query->execute();

return $query->fetchAll();

}

function getAllGroup(){

global $connection;

$query = $connection->prepare("SELECT \* FROM groups");

$query->execute();

return $query->fetchAll();

}

function getAllStudentByGroup($groupName){

global $connection;

$GroupId = getIdGroup($groupName);

$query = $connection->prepare("SELECT \* FROM student WHERE group\_id = :groupId");

$query->bindParam(":groupId", $GroupId, PDO::PARAM\_STR);

$query->execute();

return $query->fetchAll();

}

function getAllSubjectsByGroup($groupName){

global $connection;

$GroupId = getIdGroup($groupName);

$query = $connection->prepare("SELECT \* FROM subject WHERE group\_id = :groupId");

$query->bindParam(":groupId", $GroupId, PDO::PARAM\_STR);

$query->execute();

return $query->fetchAll();

}

function checkRoleUser($userId){

$user = getUserInfoById($userId);

if($user['role'] == 'admin'){

return true;

} else {

return false;

}

}

AddColumnBd.php

<?php

include('../Bd/pdo.php');

include('../Bd/module\_global.php');

global $connection;

$sql = '

CREATE DATABASE IF NOT EXISTS ваша\_база\_данных CHARACTER SET utf8 COLLATE utf8\_general\_ci;

-- Создание таблицы "users" с полем "name"

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(255) NOT NULL,

name VARCHAR(255) NOT NULL,

lastname VARCHAR(255) NOT NULL,

surname VARCHAR(255) NOT NULL,

password VARCHAR(255) NOT NULL,

role VARCHAR(255) NOT NULL

) CHARACTER SET utf8 COLLATE utf8\_general\_ci;

-- Создание таблицы "groups"

CREATE TABLE groups (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL

) CHARACTER SET utf8 COLLATE utf8\_general\_ci;

-- Создание таблицы "subject"

CREATE TABLE subject (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

group\_id INT,

FOREIGN KEY (group\_id) REFERENCES groups(id)

) CHARACTER SET utf8 COLLATE utf8\_general\_ci;

-- Создание таблицы "student"

CREATE TABLE student (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

lastname VARCHAR(255) NOT NULL,

surname VARCHAR(255) NOT NULL,

group\_id INT,

FOREIGN KEY (group\_id) REFERENCES groups(id)

) CHARACTER SET utf8 COLLATE utf8\_general\_ci;

-- Создание таблицы "grade"

CREATE TABLE grade (

id INT AUTO\_INCREMENT PRIMARY KEY,

grade VARCHAR(255) NOT NULL,

subject\_id INT,

student\_id INT,

lesson\_num INT,

FOREIGN KEY (subject\_id) REFERENCES subject(id),

FOREIGN KEY (student\_id) REFERENCES student(id)

) CHARACTER SET utf8 COLLATE utf8\_general\_ci;

';

$stmt = $connection->prepare($sql);

$stmt->execute();

$sql = "

INSERT INTO users (username, name, lastname, surname, password, role)

VALUES ('admin', 'Admin', 'Admin', 'Admin', :password, 'admin');";

$stmt = $connection->prepare($sql);

$password\_hash = password\_hash('123', PASSWORD\_BCRYPT);

$stmt->bindParam(":password", $password\_hash, PDO::PARAM\_STR);

$stmt->execute();

$sql = "

INSERT INTO users (username, name, lastname, surname, password, role)

VALUES ('user', 'User', 'User', 'User', :password, 'user');";

$stmt = $connection->prepare($sql);

$stmt->bindParam(":password", $password\_hash, PDO::PARAM\_STR);

$stmt->execute();

$group = 'Ис-4';

$sql = "INSERT INTO groups (name) VALUES (:group)";

$stmt = $connection->prepare($sql);

$stmt->bindParam(":group", $group, PDO::PARAM\_STR);

$stmt->execute();

$id = getIdGroup($group);

echo $id;

$sql = "

INSERT INTO student (name, lastname, surname, group\_id) VALUES

('Виктор', 'Перл', 'Сергеевич', :id);";

$stmt = $connection->prepare($sql);

$stmt->bindParam(":id", $id, PDO::PARAM\_INT);

$stmt->execute();

header('Location: ../Web/Login.php');