МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Кафедра вычислительных технологий**

### ДОПУСТИТЬ К ЗАЩИТЕ В ГЭК

Заведующий кафедрой, доктор физ. - мат. наук, профессор \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. И. Миков \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2016 г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

**Разработка системы безопасности веб-сайта**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.В. Митрофанов

Факультет компьютерных технологий и прикладной математики курс 4 Направление 02.03.02 «Фундаментальная информатика и информационные технологии»

Научный руководитель, преп. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С. С. Ермоленко

Нормоконтролер\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю. С. Фисун

Краснодар 2016

СОДЕРЖАНИЕ

# Введение…...........................................................................................................3

1. Безопасность веб-сайта....................................................................................5
   1. Защита от SQL-инъекций................................................................5
   2. Привилегии различных типов пользователей..............................9
   3. Хеширование с использованием соли.....................9
   4. Аудит……………...................................................................11
2. Электронный дневник и его задачи.............................................................13
   1. Актуальность электронных дневников…....................13
   2. Сравнительный анализ существующих продуктов………....18
3. Реализация......................................................................19
   1. Средства для реализации веб-сайта.................................19
   2. Авторизация и страница с выводом ошибок.................22
   3. Интерфейс ученика...........................................................25
   4. Интерфейс родителя................................................................30
   5. Интерфейс преподавателя..................................................31
   6. Интерфейс администратора.....................................................35 3.7 Интерфейс директора…....................................................45

Заключение…….................................................................................................48

Список использованных источников…………...............................................49 Приложение А Исходный код..........................................................................50

ВВЕДЕНИЕ

Как известно, цифровые внедрения нынче повсеместны: школы, колледжи и университеты заводят собственные странички в социальных сетях и блогах для информирования о происходящем, редкое учебное заведение не имеет собственного сайта, а электронные журналы вот-вот окончательно вытеснят бумажные. По постановлению администрации Краснодарского края с 1 сентября 2016 года каждая школа края должна будет иметь свой электронный дневник. В других регионах подобное постановление либо уже было приведено в действие, либо в скором времени это произойдет. Таким образом, эпоху бумажных дневников можно считать законченной. И это лишь часть тех перемен, которые происходят у нас на глазах. Конечно же, в информационный век этому уже не приходится удивляться: люди привыкли, что вся интересующая их информация находится Интернете.

Очевидно, эти процессы инкапсулируют в себе и немалую ответственность, как со стороны пользователей, так и со стороны разработчиков. Теперь недостаточно, уходя из кабинета, запирать дверь на ключ, оставляя позади нее всё важное. В цифровом мире физическая недоступность не является гарантией безопасности (однако, несомненно, этот аспект крайне важен в плане обеспечения комплексной безопасности). Создание безопасного относительно внешних угроз и неумелого поведения пользователей электронного дневника является целью данной работы. В ходе работы были изучены средства, позволяющие реализовать веб-сайт, были изучены проблемы, касающиеся безопасности веб-сайта, был создан электронный дневник школы и были предприняты меры, решающие проблемы безопасности сайта.

Основная часть данной выпускной работы состоит из трех разделов. В первом разделе детально рассматривается всевозможные угрозы для безопасности веб-сайта, способы их избегания и средства необходимые для этого. Во втором разделе дается подробная информация о самом понятии электронного дневника, его задачах, свойствах и особенностях. В третьем разделе приводится подробное описание реализации поставленной задачи средствами языков PHP, HTML, CSS, JavaScript и SQL и рассматривается использование созданного электронного дневника с точки зрения различных типов пользователей (школьник, родитель, преподаватель, администратор и директор). Код реализации находится в приложении данной работы. В конце данной работы, приводится заключение, в котором формулируются выводы о проделанной работе и даются заключительные оценки о соответствии полученных результатов поставленным задачам.

1 Безопасность веб-сайта

## 1.1 Защита от SQL-инъекций

Многие веб-разработчики даже не догадываются, что SQL-запросы могут быть подделаны, и считают, что SQL-запросы всегда достоверны. На самом деле поддельные запросы могут обойти ограничения доступа, стандартную проверку авторизации, а некоторые виды запросов могут дать возможность выполнять команды операционной системы.

Прямое внедрение вредоносных инструкций в SQL-запросы - это методика, в которой взломщик создает или изменяет текущие SQLзапросы для отображения скрытых данных, их изменения или даже выполнения опасных команд операционной системы на сервере базы данных. Атака выполняется на базе приложения, строящего SQL-запросы из пользовательского ввода и статических параметров.

Благодаря отсутствию проверки пользовательского ввода и соединению с базой данных под учетной записью суперпользователя (или любого другого пользователя, наделенного соответствующими привилегиями), взломщик может создать еще одного пользователя БД с правами суперпользователя.

Хотя по-прежнему очевидно, что взломщик должен обладать, по крайней мере, некоторыми знаниями о структуре базы данных, чтобы провести успешную атаку, получить эту информацию зачастую очень просто. Например, если база данных является частью open-source или другого публично доступного программного пакета с инсталляцией по умолчанию, эта информация является полностью открытой и доступной. Эти данные также могут быть получены из закрытого проекта, даже если он закодирован, усложнен, или скомпилирован, и даже из вашего личного кода через отображение сообщений об ошибках. К другим методам относится использование распространенных (легко угадываемых) названий таблиц и столбцов [1].

Из всего сказанного можно, наконец, сформулировать определение инъекции.

### SQL-инъекции - это способ хакерской атаки, когда передаваемые

скрипту данные модифицируются таким образом, что запрос, формируемый в этом скрипте, начинает выполнять совсем не то, для чего он предназначался [6].

Правила защиты от таких атак можно разделить на два пункта:

* Работа с данными (не помещать в базу данных данные без обработки);
* Работа с управляющими элементами запроса (не помещать в запрос управляющие структуры и идентификаторы, введенные пользователем, а заранее прописывать в скрипте список возможных вариантов, и выбирать только из них).

Что касательно второго пункта, то тут все понятно: не следует помещать в запрос различные if, else и так далее. Поэтому следует поподробнее рассмотреть правила, относящие к первому пункту.

При подстановке строковых данных в запрос, следует придерживаться двух правил:  все вставляемые строковые данные должны быть заключены в кавычки (одинарные или двойные, но удобнее и чаще используются одинарные).  в них должны быть экранированы слешами спецсимволы.

### К примеру, если написать «SELECT \* FROM table WHERE name =

Bill», то база решит, что Bill - это имя другого поля, не найдёт его, и выдаст ошибку. Поэтому подставляемые данные (в данном случае имя Bill) надо заключать в кавычки - тогда база сочтет его строкой, значение которой надо присвоить полю name: «SELECT \* FROM table WHERE name

= 'Bill'».

Однако и в самих данных могут встречаться кавычки тоже. К примеру, «SELECT \* FROM table WHERE name = 'Д'Артаньян'».

Здесь база данных решит, что 'Д' - это данные, а Артаньян - команда, которую она не знает, и тоже выдаст ошибку. Поэтому и надо прослешивать все данные, чтобы объяснить базе, что встречающиеся в них кавычки (и некоторые другие спецсимволы) относятся к данным.

В результате мы получим правильный запрос, который ошибок не вызовет: «SELECT \* FROM table WHERE name = 'Д\'Артаньян'».

Всё вышесказанное относится к данным строкового типа и датам. Числа можно вставлять, не прослешивая и не окружая кавычками. Если вы так делаете, то обязательно приводите данные к нужному типу перед вставкой в запрос (например, $id=intval($id)).

Однако для простоты и надёжности можно и с числами работать, как со строками (поскольку MySQL всё равно преобразует их к нужному типу). Соответственно, мы будем любые данные, вставляемые в запрос, прослешивать и заключать в кавычки [7].

Так же, есть ещё одно правило - необязательное, но его следует придерживаться во избежание появления ошибок:

Имена полей и таблиц следует заключать в обратные одинарные кавычки - "`" (клавиша с этим символом находится на стандартной клавиатуре слева от клавиши "1") Ведь имя поля может совпадать с ключевыми словами MySQL, но если мы используем обратную кавычку, то MySQL поймёт всё правильно: «SELECT \* FROM `table` WHERE `date` = '2006-04-04'».

Следует различать эти кавычки и не путать одни с другими. Следует также помнить, что обратные кавычки слешами не экранируются.

Пойдем далее. Один из самых популярных способов отправлять запросы в БД называется "подготовленными выражениями" (prepared statements).

Суть его заключается в том, что подготавливается шаблон запроса, со специальными маркерами, на место которых будут подставлены динамические компоненты. Пример такого шаблона: «SELECT \* FROM table WHERE name=?».

Знак вопроса здесь - это тот самый маркер. По-другому он называется плейсхолдером (placeholder). Весь секрет в том, что данные на его место подставляет специальная функция, которая "привязывает" переменную к запросу.

Вот как выглядит код в таком случае:

$stmt = $mysqli->prepare("SELECT District FROM City WHERE Name=?");

$stmt->bind\_param("s", $city);

$stmt->execute();

В первой строчке мы подготавливаем шаблон запроса.

Во второй - привязываем к маркеру значение переменной $city.

В третьей строчке выполняем подготовленный таким образом запрос.

При этом запрос и данные идут в базу не вместе, а по отдельности, исключая возможность какой-либо ошибки или злонамеренной манипуляции.

## 1.2 Привилегии различных типов пользователей

Одной из основ безопасности сайта является разделение полномочий (привилегий) пользователей, то есть понятие того, что можно и чего нельзя определенному типу пользователя.

На разрабатываемом в данной работе сайте имеются пять типов пользователей (пять ролей): Ученик, Родитель, Преподаватель, Администратор и Директор. У каждого из них свои собственные функции и полномочия, которых нет у других пользователей. У кого-то их больше, у кого-то – меньше, но ни в этом суть. У каждой роли свои личные страницы, на которые не сможет зайти пользователей с другой ролью.

Таким образом, возможности и полномочия пользователя строго ограничены в соответствии с системой привилегий, используемой на данном сайте, что значительно увеличивает его безопасность.

## 1.3 Хеширование с использованием соли

Время от времени, серверные базы данных похищают. Учитывая это, важно убедиться, что некоторые важные пользовательские данные, такие как пароли, могут быть не восстановлены. Идеальным решением данной проблемы является такое понятие, как хеширования, помогающее защитить пароли в веб-приложениях.

Хеширования преобразует данные (большого или малого размеров), в относительно короткий отрезок данных, таких как строка или целое число. Это осуществляется с помощью односторонней хеш-функции. “Односторонняя” означает, что очень трудно (практически невозможно) получить первоначальные данные.

Криптографическая соль представляет собой данные, которые применяются в процессе хэширования для предотвращения возможности разгадать оригинальный ввод с помощью поиска результата хэширования в списке заранее вычисленных пар ввод-хеш, известном также как "радужная" таблица.

Более простыми словами, соль - это кусочек дополнительных данных, которые делают ваши хеши намного более устойчивыми к взлому. Существует много онлайн-сервисов, предоставляющих обширные списки заранее вычисленных хешей вместе с их оригинальным вводом. Использование соли делает поиск результирующего хеша в таком списке маловероятным или даже невозможным.

При использовании функции password\_hash(), возвращаемое значение уже содержит соль как часть созданного хеша. Это значение нужно хранить как есть в вашей базе данных, так как оно содержит также информацию о хеширующей функции, которая использовалась, и может быть непосредственно передано в функцию password\_verify() при проверке пароля.

Таким образом, можно сформулировать следующие алгоритмы регистрации и авторизации.

Процесс регистрации пользователя:

* Пользователь заполняет регистрационную форму, в том числе и поле пароль.
* Веб-скрипт сохраняет всю информацию в базу данных. Однако пароль будет передан хеш-функции, перед сохранением. Не зашифрованный пароль нигде не сохраняется.

И процесс входа в систему:

* Пользователь вводит имя пользователя и пароль.
* Скрипт передаёт пароль той же хеш-функции.
* Скрипт находит записи пользователя в базе данных, и читает хранящийся в базе данных хэшированный пароль.
* Оба эти значения сравниваются, и, если они совпадают, то пользователь авторизуется.

Стоит заметить, что оригинальный пароль нигде не хранится. Если базу данных украли, то пользовательские данные не могут быть скомпрометированы.

## 1.4 Аудит

Как отслеживать действия пользователей на сайте? Как определить, почему произошли эти или иные изменения? Ответ только один – провести исследование веб-сайта.

Аудит сайта заключается в выявлении сильных и слабых сторон данного сайта и разработке предложений по повышению его эффективности. Комплексный аудит сайта дает всю необходимую информацию для выявления поведения пользователей и для принятия решения о возможной коррекции сайта.

Аудит, представленный в данной работе, предоставляет возможность отслеживать, какие изменения произвел конкретный пользователь со своими и чужими персональными данными, оценками, домашними заданиями и предметами.

После проведения аудита сайта предоставляется подробный отчет о действиях всех пользователей, на основании которого можно выявить нарушения или ошибки в функционале сайта.

# 2. Электронный дневник и его задачи

## 2.1 Актуальность электронных дневников

С внедрением технологических новинок в век всеобщей компьютеризации возникла идея создания электронного журнала (дневника) школы. Как показало время, данная инновация принесла выгоду всем участникам образовательного пространства. Разберемся, какую именно.

Основная задача внедрения в школьную практику электронного журнала - удобство во взаимодействии его пользователей. В чем это удобство осуществляется?

* Родители и учащиеся имеют возможность узнать расписание уроков, домашнее задание, намечающиеся в школе мероприятия;
* Родители могут контролировать успеваемость детей, опоздания, прогулы;
* Система электронного дневника помогает составлять отчеты по успеваемости, просчитывать рейтинги и даже провести электронные тесты с учениками, отсутствующими в школе.

Работа со школьным журналом уже не такое нудное занятие. А система электронного журнала с разделением прав доступа для пользователей сохраняет конфиденциальную информацию. То есть, родители увидят только ту информацию, которая касается их детей.

Отпадает необходимость так называемой «бумажной волокиты», а также можно теперь не бояться, что с кражей классного журнала пропадет вся информация об успеваемости.

Что дает система электронного журнала конкретно каждому участнику образовательного пространства?

* Учитель

Учитель может в любую минуту получить необходимую ему информацию или оставить сообщение для родителей, назначить домашнее задание, посмотреть итоги успеваемости, даже не выходя из дома, так как для работы с электронным дневником достаточно всего лишь зайти в Интернет.

Также немаловажный момент для учителя, который в силу каких-то обстоятельств (болезнь, курсы или что-либо другое) не может выполнять свои прямые обязанности. Тогда, имея под рукой такой электронный дневник, он сможет контролировать весь учебный процесс, получать информацию о работе его учеников и их оценках. Несомненным плюсом такой системы является минимизация бумажных носителей, которая вместе с тетрадями детей составляет непосильную ношу на плечи наших учителей. К тому же исключается их утрата или использование закрытой информации недобросовестными учениками.

* Ученик

Многие учащиеся против внедрения электронных дневников в образовательную систему. Они не хотят, чтобы родители контролировали их учебный процесс. Но не зря в народе говорят: "Тайное всегда становится явным", и в итоге родители узнают про их оценки в школе, прогулы или нарушение дисциплины.

Поэтому пусть лучше родители узнают про двойку за невыполненное задание от самих детей, чем от кого-то другого. Значит, польза от введения электронных дневников есть. Она, как минимум, воспитает в наших детях честность и открытость по отношению к другим людям.

К тому же с момента зарождения идеи о создании электронных дневников прошло ужу немало времени, и они перестали быть электронной версией привычных нам бумажных дневников. Теперь у детей есть уникальная возможность пообщаться друг с другом в чатах вместо того, чтобы проводить время за просмотром "ненужных" сайтов или компьютерными играми.

В наиболее продвинутых электронных дневниках есть возможность хранить информацию в так называемых библиотеках, где содержатся необходимые материалы, книги или учебники.

* Родители

Звание "родитель" накладывает на вас определенные обязанности, а с поступлением ребенка в школе к этим обязанностям добавляются новые (родительские собрания, контроль за успеваемостью, выполнением домашнего задания и другие нюансы). Но как все это совместить с работой? Тут на помощь приходит электронный дневник ученика. Стоит только заглянуть туда, и вы получите всю необходимую для вас информацию, озвученную выше. Помимо этого, вы можете взаимодействовать с учителями или администрацией школы. Выделите, допустим, на работе несколько минут и посмотрите, как успехи у ребенка в школе. Если обнаружите двойку за невыполненное домашнее задание, то у вас хотя бы будет возможность все взвесить и не наломать дров, как если бы вы это узнали дома из дневника ребенка.

* Администратор

Роль администратора электронного дневника, пожалуй, самая важная и ответственная. Именно от этого человека зависит регистрация всех пользователей, добавление расписания, предметов и уроков. По сути, это первый человек, к которому можно обратиться, если что-то пошло не так.

Выполнение этой роли требует от человека внимание к всевозможным деталям, усидчивости и пунктуальности.

Вдобавок ко всему выше перечисленному, в электронном дневнике, разрабатываемом в данной дипломной работе, добавлена функция аудита сайта. Именно при ее помощи администратор сможет отслеживать действия всех пользователей (кто и когда изменил пароль, поставил оценку, добавил домашнее задание и так далее). Аудит сайта поможет предотвратить целое множество нежелательных последствий.

* Директор

Директор – это человек, который, грубо говоря, контролирует все. Ему важно, сколько в подвластном ему учебном заведении неуспевающих учеников, сколько отличников, троечников и так далее.

Чтобы раньше он смог получить эту информацию, ему или его помощникам необходимо было сверять и сопоставлять громадное количество классных журналов и прочей бумажной документации. Теперь же, при помощи электронного дневника, получить всю эту статистику можно за считанные секунды, не прилагая к этому никаких усилий.

Подводя итог под вышесказанным, хочется отметить, что сейчас почти каждая школа имеет свой сайт, который доступен всем. Электронный дневник тоже своего рода сайт школы, но доступ к нему имеют только учителя, администрация школы, учащиеся и родители. Таким образом, электронный дневник - это внутренний сайт школы, где размещена вся информация о жизни школы, множество полезной информации для всех участников образовательного процесса и возможность общения друг с другом. Он гораздо гибче своего «бумажного» предшественника, и, безусловно, облегчает весь учебный процесс в школах.

Основными источниками угрозы для безопасности электронного дневника являются невнимательность людей и вредоносные действия злоумышленников. Другими словами: самая большая уязвимость любой информационной системы — это её пользователь.

С этим высказыванием трудно поспорить — человек есть человек. Именно эта проблема является ключевой в плане надежности и безопасности информационных ресурсов, как со стороны пользователей, так и разработчиков. Эта проблема актуальна ровно столько, сколько существует сам человек, а поэтому она не относится к цифровому миру, как таковому. Ведь и во времена, предшествующие цифровым, например, бумажный журнал было доверено переносить конкретному человеку (старосте или кому либо другому), который, в общем-то, мог бы пользоваться своим положением или быть использованным другими своими одногруппниками, что, в конечном счете, вполне могло привести к подделке оригинальной информации, ее компрометации.

Другой пример, когда виновниками выступают уже не доверенные лица, а непосредственные «владельцы» информации. Если рассматривать это на конкретном примере, то легко представить себе обычного, скажем, секретаря, не подумавшего не только забрать, но и вообще закрыть журнал с важной информацией на время своего отсутствия в кабинете. Такое бывает и нередко. Это также может привести к утечке/подделке важных для учреждения сведений.

В современное время примеры неаккуратности человеческого фактора остаются идентичными, разве что бумажки заменяются компьютерами. На самом деле это несет еще большую угрозу. Если запрограммировать и «заразить» бумажку было нельзя, то с компьютерами все обстоит иначе.

В связи с этим можно выделить следующие наиболее важные направления, которые при должной их разработке увеличат надежность рассматриваемого электронного дневника:

* Защита базы данных от несанкционированных попыток ее изменения или получения из нее информации;
* Жесткое разграничение возможностей различных типов пользователей;
* Система, помогающая отслеживать действия всех пользователей. Решение поставленных задач представлено в первом разделе данной работы.

## 2.2 Сравнительный анализ существующих продуктов

В ходе выполнения работы были изучены различные уже существующие и внедренные в учебные заведения электронные дневники. В некоторых из них были выявлены недостатки, которые были исправлены в данной работе. Среди них можно выделить следующие:

* Отсутствие роли «Директор»;
* Возможность просмотра оценок других учеников в классе Родителем.

# 3. Реализация

## 3.1 Средства для реализации веб-сайта

Как уже было сказано во введении, для реализации данного дипломного проекта были использованы языки PHP, HTML, CSS, JavaScript и SQL. Ниже приводится краткое описание и функциональные возможности каждого из них.

PHP (от англ. «PHP: Hypertext Preprocessor» — «PHP: препроцессор гипертекста») — скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.

Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки вебприложений, среди которых можно выделить наиболее значимые:

* автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы;
* взаимодействие с большим количеством различных систем управления базами данных (MySQL, MySQLi, SQLite, Oracle, Microsoft SQL Server, Sybase, ODBC, IBM DB2, Cloudscape и Apache

Derby, Informix, Ovrimos SQL, Lotus Notes, DBX, FrontBase и другие);  автоматизированная отправка HTTP-заголовков;  работа с HTTP-авторизацией;  работа с cookies и сессиями;  работа с локальными и удалёнными файлами;  обработка файлов, загружаемых на сервер.

HTML (от англ. HyperText Markup Language — «язык гипертекстовой разметки») — стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML).

Текстовые документы, содержащие разметку на языке HTML (такие документы традиционно имеют расширение .html или .htm), обрабатываются специальными приложениями, которые отображают документ в его форматированном виде. Такие приложения, называемые «браузерами» или «интернет-обозревателями», обычно предоставляют пользователю удобный интерфейс для запроса веб-страниц, их просмотра (и вывода на иные внешние устройства) и, при необходимости, отправки введённых пользователем данных на сервер [2].

CSS (от англ. Cascading Style Sheets — каскадные таблицы стилей) — формальный язык описания внешнего вида документа, написанного с использованием языка разметки. Преимущественно используется как средство описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но может также применяться к любым XML-документам, например, к SVG или XUL.

CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось разделение описания логической структуры вебстраницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими шрифт Брайля [4].

JavaScript (JS) — прототипно-ориентированный сценарный язык программирования. JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам [3].

SQL (от англ. structured query language — «язык структурированных запросов») — формальный непроцедурный язык программирования, применяемый для создания, модификации и управления данными в произвольной реляционной базе данных, управляемой соответствующей системой управления базами данных [5]. SQL является основным способом работы пользователя с базой данных и позволяет выполнять следующий набор основных операций:

* создание в базе данных новой таблицы;  добавление в таблицу новых записей;  изменение записей;  удаление записей;
* выборка записей из одной или нескольких таблиц (в соответствии с заданным условием);
* изменение структур таблиц.

Операторы SQL делятся на операторы определения данных и операторы манипуляции данными. Ниже представлен список самых основных операторов:

* CREATE создает объект БД (саму базу, таблицу, представление, пользователя и т. д.);
* ALTER изменяет объект;
* DROP удаляет объект;
* SELECT выбирает данные, удовлетворяющие заданным условиям;
* INSERT добавляет новые данные;  UPDATE изменяет существующие данные;
* DELETE удаляет данные.

## 3.2 Авторизация и страница с выводом ошибок

Первая страница, которую пользователь видит, заходя на сайт – это страница авторизации, где по логину и паролю он может зайти на свою личную страницу. У каждого типа пользователя она, разумеется, имеет свои отличные от других визуальные и функциональные особенности.

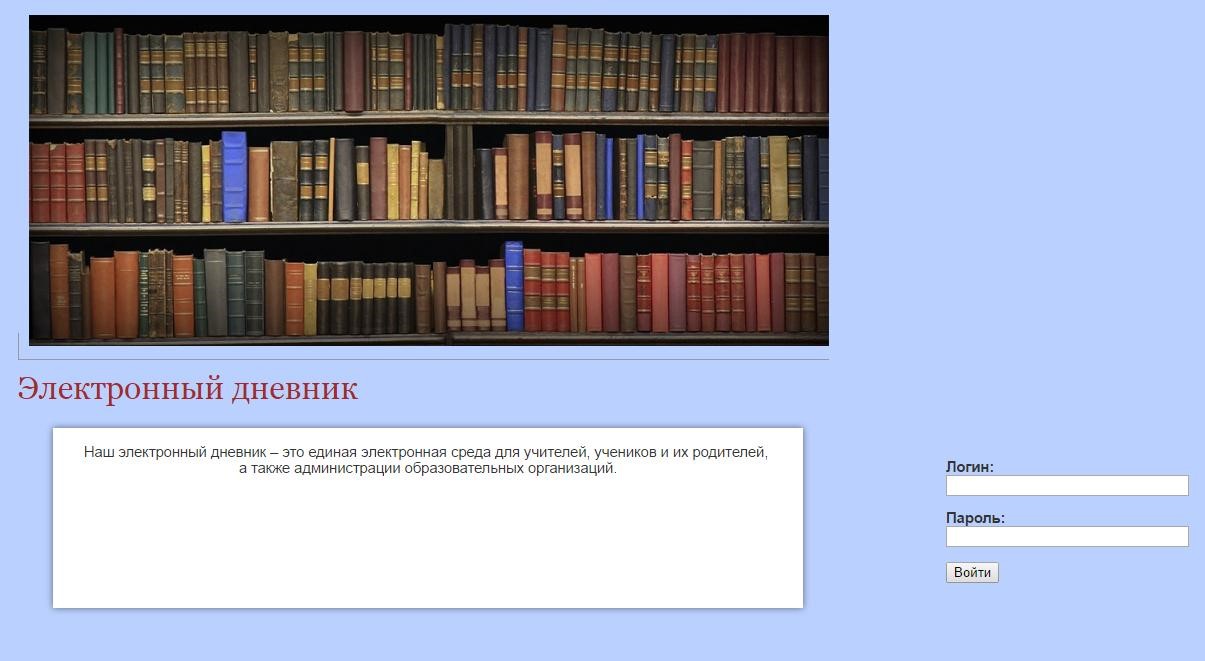


Рисунок 1 – Страница авторизации

Если неавторизованный пользователь попытается зайти на какуюлибо другую страницу сайта, то он будет автоматически перенаправлен на страницу авторизации.

Зарегистрированный же пользователь, попытаясь зайти на страницу пользователя другого типа, будет переведен на страницу, на которой будет написано, что у него нет доступа к этой странице.

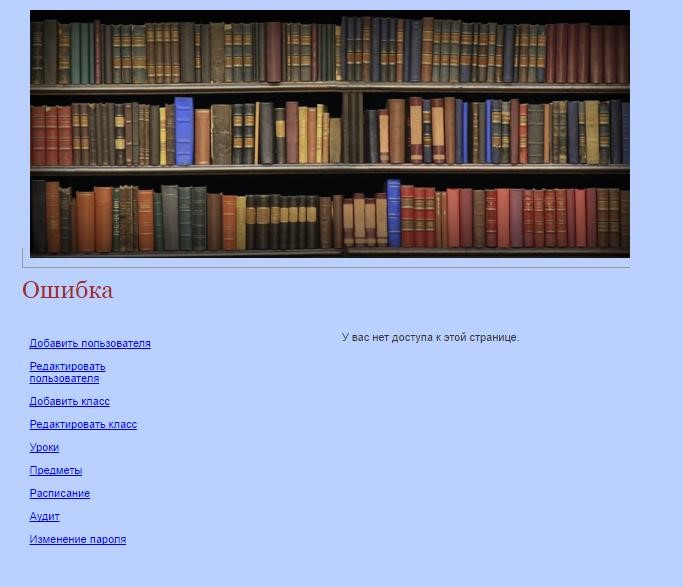


Рисунок 2 – Попытка зайти на страницу другого пользователя

Также во всевозможных формах, при попытке отправить информацию, не соответствующую заранее определенным правилам, пользователь будет перенаправлен на страницу ошибки с указанием того, что он сделал неправильно. Если же пользователь не заполнен какое-либо обязательное поле в форме, он не сможет отправить информацию на сервер, и будет выведено сообщение с просьбой заполнить это поле.

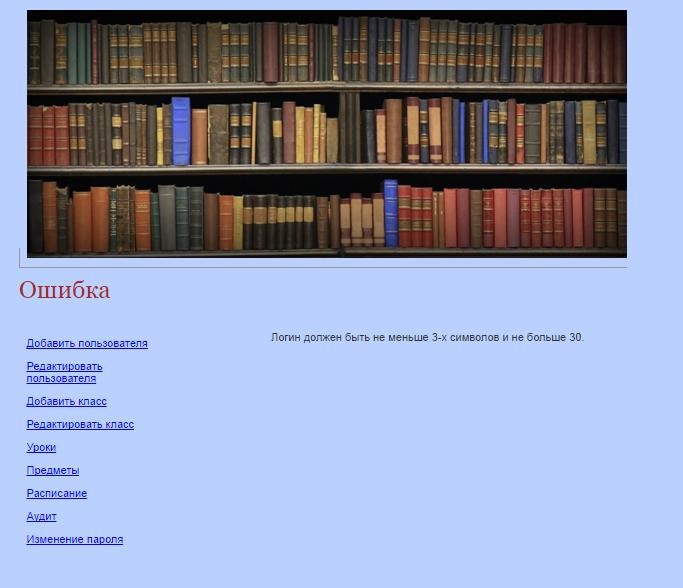


Рисунок 3 – Страница с сообщение о неправильном заполнении поля

«Логин»

## 3.3 Интерфейс ученика

На домашней странице Ученика, на которую он попадает после авторизации, содержится информация о классе, в котором он учится – это название класса, ФИО его классного руководителя, а также последние новости класса.

Картинка с книжными полками является ссылкой на домашнюю страницу пользователя. Справа от нее располагается наименование учебного заведения, в котором учится школьник. Справа по центру (также как и у других пользователей) выводится фамилия и имя авторизованного пользователя. Там же располагается кнопка «Выйти», после нажатия на которую пользователь перенаправляется на страницу авторизации.

Слева располагается меню, в котором содержатся ссылки на другие страницы, на которые может перейти данный пользователь.



Рисунок 4 – Домашняя страница ученика

Переходя на вкладку «Домашнее задание», Ученик должен выбрать дату того дня, на которую он хочет посмотреть домашнее задание, и нажать кнопку «Далее».

После этого он переходит на страницу, изображенную на рисунке 5, на которой отображается само домашнее задание.

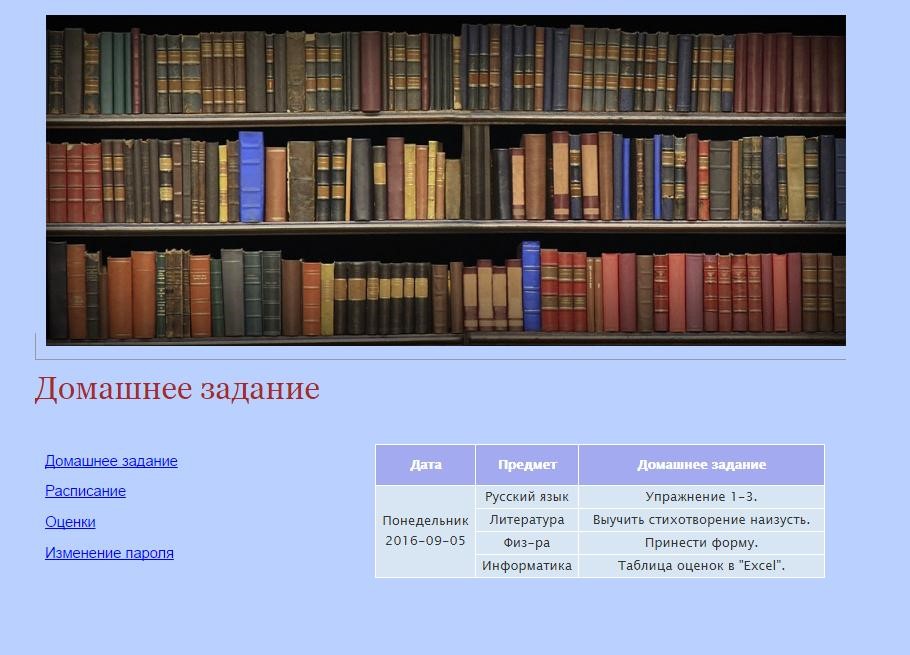


Рисунок 5 – Домашнее задание класса

На странице «Расписание» находится таблица с расписанием класса на неделю в течение четверти (полугода). В таблице указаны расписание уроков, названия предметов, ФИО преподавателя, который ведет этот предмет и номер аудитории, где проходит урок.

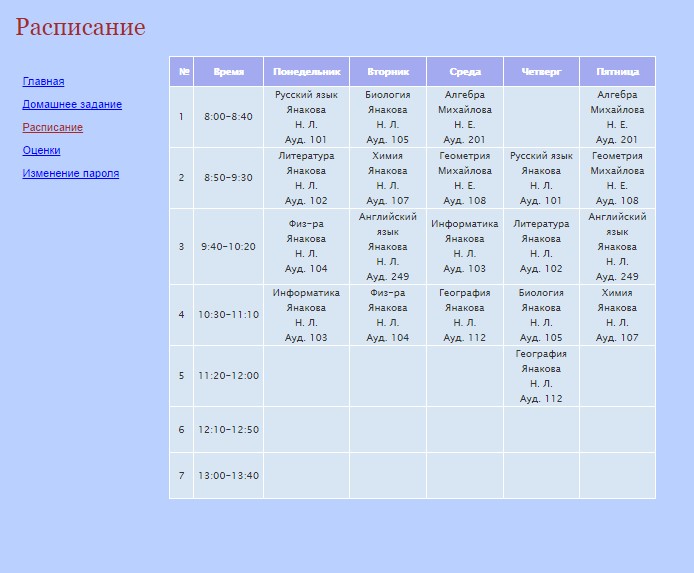


Рисунок 6 – Расписание класса

Переходя на вкладку «Оценки», Ученик выбирает предмет, оценки по которому он хочет посмотреть, и нажимает кнопку «Далее».

После этого он переходит на страницу, изображенную на рисунке 7, на которой отображается оценки всех учеников класса. Вывод оценок всех учеников, а не оценок конкретно одного пользователя, обуславливается тем, что иногда полезно сравнивать успеваемость одного человека с успеваемостью всего класса.

Последними столбцами в таблице является столбы с годовыми оценками учеников, столбы с четвертными оценками или полугодовыми, в зависимости от того, о каком классе идет речь (1-9 или 10-11 соответственно).



Рисунок 7 – Оценки класса

И последняя страница во вкладке меню отвечает за самостоятельное изменение пароля пользователем. На ней ему необходимо ввести свой текущий пароль и новый, и, если текущий пароль введен правильно, изменение произойдет успешно.

Такая возможность есть у каждого типа пользователя, поэтому дальше о ней упоминаться не будет.

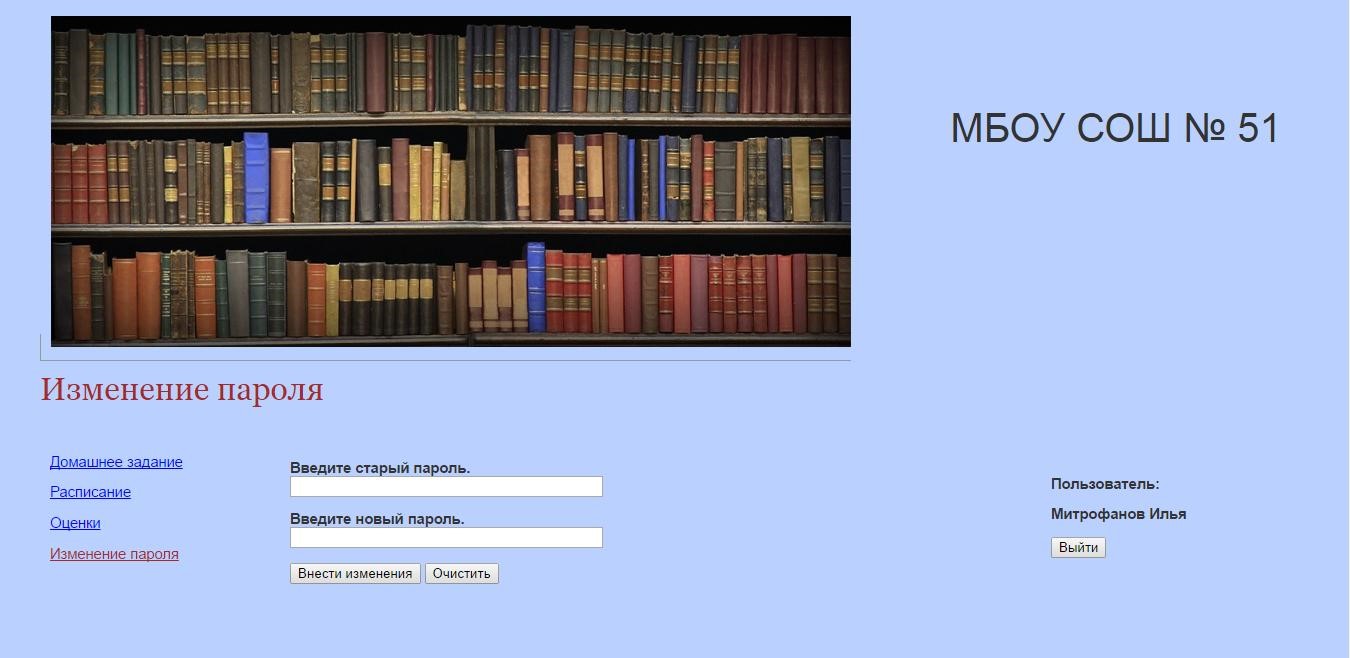


Рисунок 8 – Страница для изменения пароля

## 3.4 Интерфейс родителя

Страницы, которые видит Родитель, ничем не отличаются от тех, которые видит Ученик. Родителю доступны все страницы его ребенка. Но есть одно исключение.

В силу того, что у одного Родителя в одной и той же школе могут учиться несколько детей, после авторизации первая страница, которая предлагается ему, содержит выбор, информацию о каком именно своем ребенке он хочет посмотреть. Выбрав одного из них, далее он переходит на страницу класса и так далее, как у Ученика.

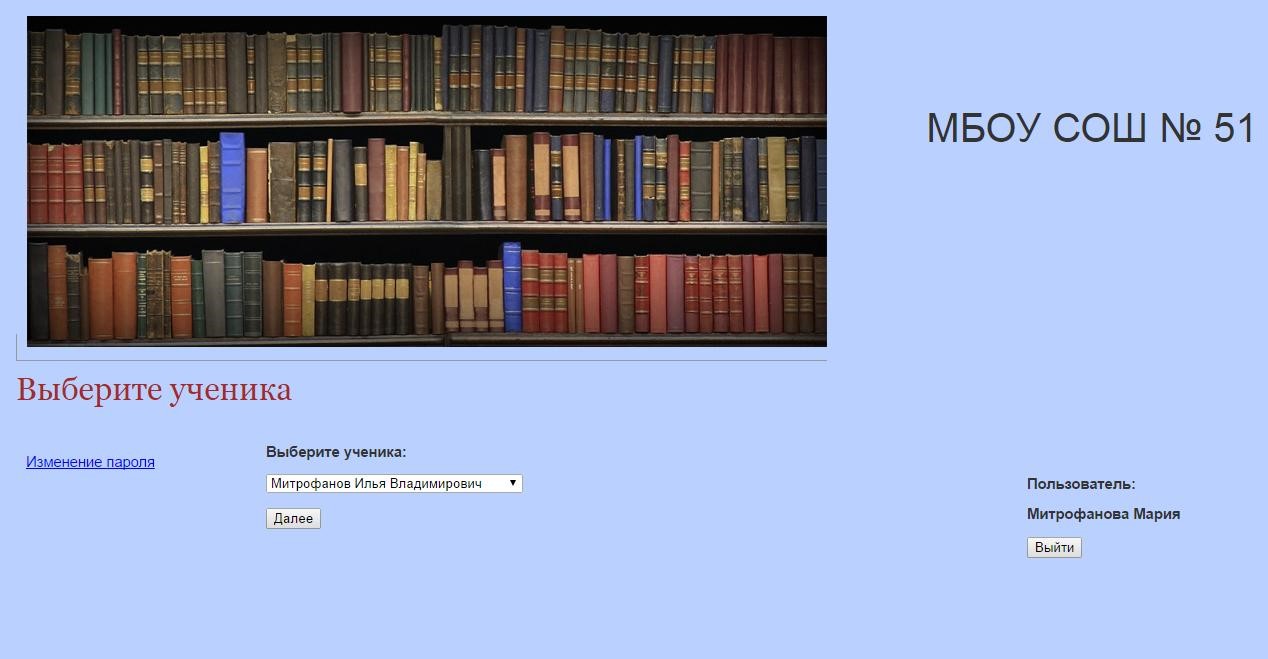


Рисунок 9 – Выбор родителем своего ребенка

В случае если родители ребенка либо разведены, либо не живут вместе, в данном электронном дневнике предусмотрена возможность создание у одного Ученика двух Родителей, что позволяет предотвратить возможные нежелательные препятствия и конфликты в семье ребенка.

## 3.5 Интерфейс преподавателя

У типа пользователя «Преподаватель» имеются две функции – выставление оценок и добавление домашнего задания.

При переходе на вкладку «Домашнее задание», Преподавателю открывается страница с формой, на которой ему необходимо выбрать класс, предмет, дату и ввести текст домашнего задания.

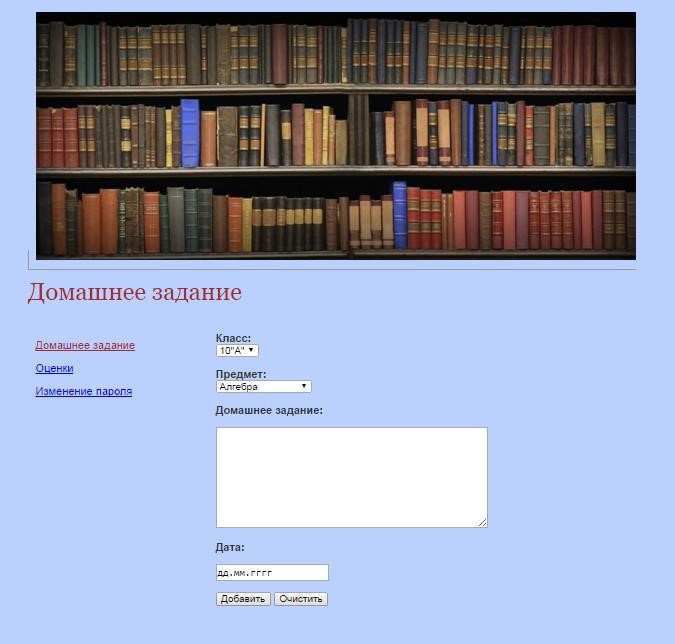


Рисунок 10 – Добавление домашнего задания

При переходе на вкладку «Оценки» Преподавателю открывается страница с формой, в которой ему необходимо выбрать класс, предмет, дату нового урока, на котором он хочет поставить оценки и нажать кнопку «Далее».



Рисунок 11 – Выбор класса, предмета и даты для выставления оценок

После этого Преподаватель переходит на страницу, где сможет расставить оценки ученикам класса. Там же он может проставить итоговые оценки за четверть (полугодие) и год. Как только он закончит выставление, ему необходимо нажать кнопку «Добавить», чтобы новые оценки сохранились в базе данных.

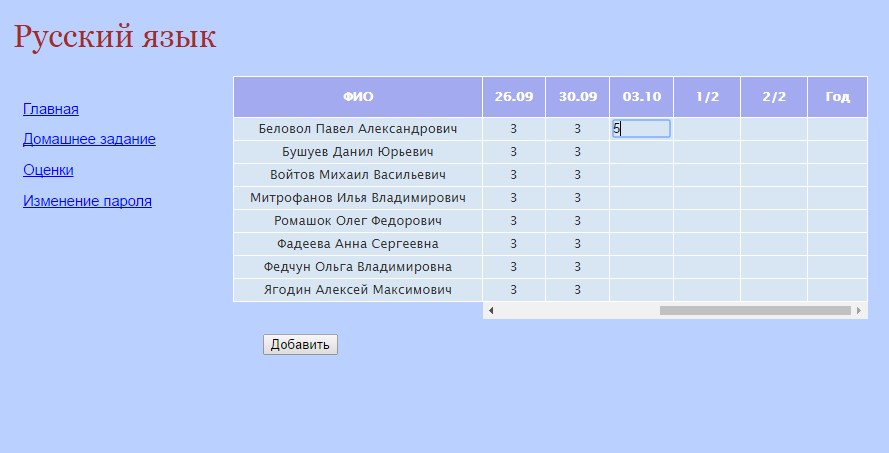


Рисунок 12 – Выставление оценок

Также у Преподавателя есть возможность довести до сведения других информацию о том, какие события происходят в классах, в которых они преподают или у которых они являются классными руководителями. Для этого Преподавателю необходимо перейти на страницу «Новости», выбрать класс и написать текст новости. Данное сообщение будет видно всем ученикам и родителям класса в колонке «Последние новости» на домашней странице класса.

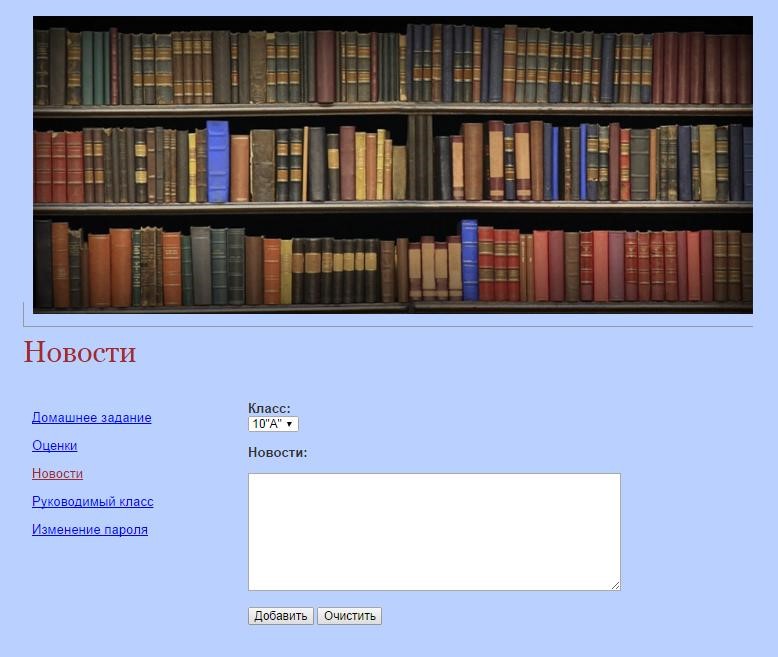


Рисунок 13 – Страница «Новости»

Также у Преподавателя есть возможность, в случае если он является классный руководителем, посмотреть всю информацию о руководимом им классе. Для этого ему необходимо перейти на страницу «Руководимый класс». Преподавателю будут доступны все страницы Ученика.

## 3.6 Интерфейс администратора

Как уже оговаривалось выше, у Администратора больше, чем у коголибо другого, функций на этом сайте. Перечислим их все.

Первым делом Администратор должен зарегистрировать само образовательное учреждение, для чего есть соответствующая страница на сайте, где необходимо ввести наименование учреждения и выбрать его директора.

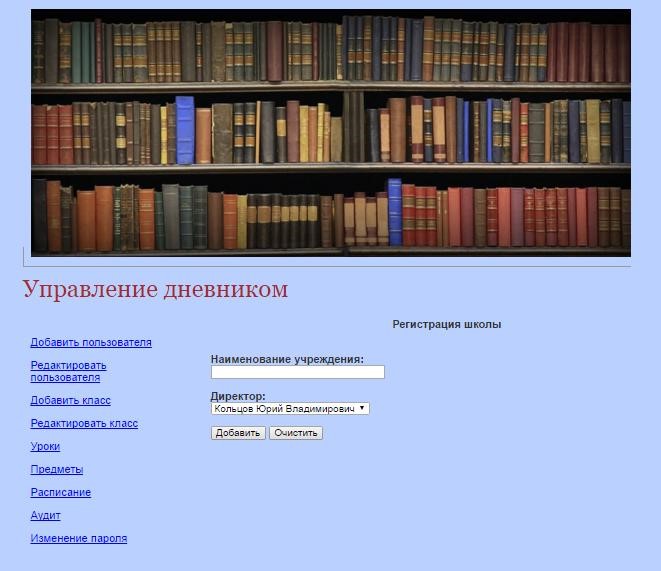


Рисунок 14 – Регистрация школы

Добавление пользователей – одна из самых важных функций Администратора. На соответствующей странице в форме ему необходимо ввести логин пользователя, адрес его электронной почты, ФИО и выбрать его тип – Школьник, Родитель, Преподаватель, Администратор или Директор.

Причем в зависимости от типа пользователя при помощи JavaScript на форме появляются или убираются некоторые элементы. Например, выборка класса присутствует только у Школьника, поскольку только ему необходимо иметь принадлежность к какому-либо классу, а выборка ребенка присутствует только при выборе типа «Родитель».

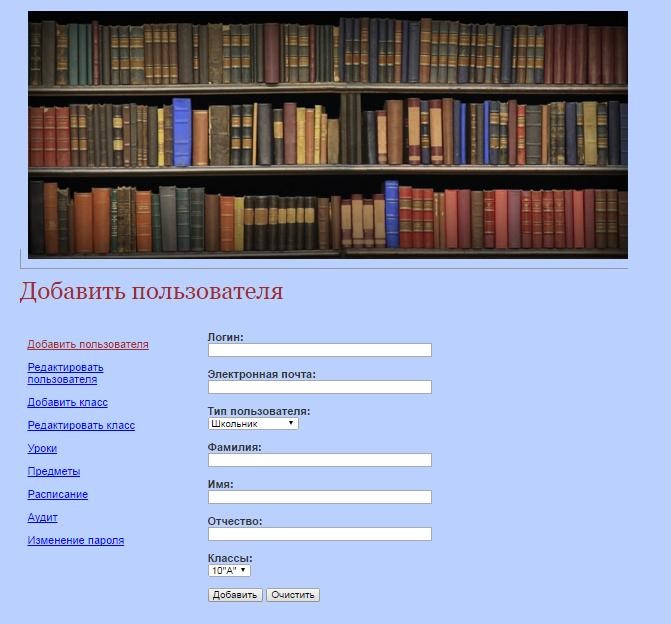


Рисунок 15 – Добавление пользователя

Так как есть возможность добавление пользователя, так и должна быть возможность редактировать его или удалить, что и позволяет сделать страница «Редактировать пользователя». Ее форма точно такая же, как и у страницы добавления, только лишь за тем изменением, что вместо ввода логина пользователя стоит поле, в котором вы можете выбрать того пользователя, над учетной записью которого вы хотите произвести изменения.

На каждую функцию добавления в данном электронном дневнике есть соответствующие функции редактирования и удаления, будь то пользователь, класс, предмет и так далее.



Рисунок 16 – Редактирование пользователя

Чтобы добавить новый класс, Администратору следует пройти на соответствующую страницу и всего лишь ввести название класса и выбрать ему классного руководителя, как это показано на рисунке ниже.

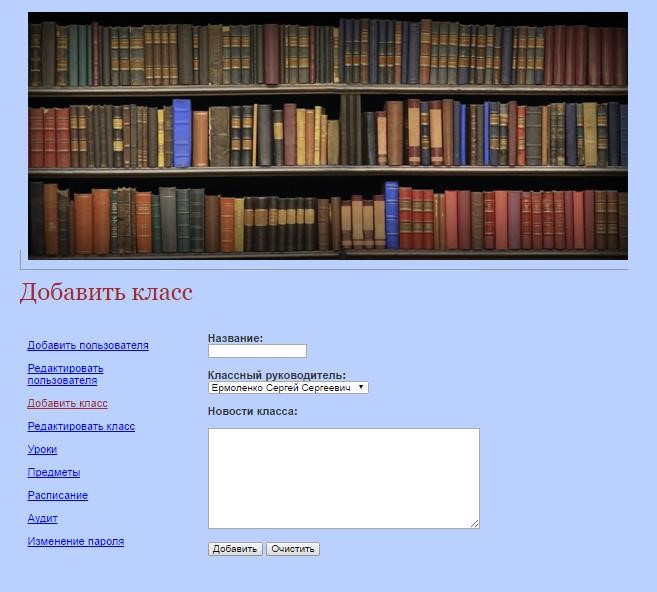


Рисунок 17 – Добавление класса

По окончанию учебного года все классы должны перевестись в новый год, и было бы крайне неудобно создавать каждый раз новые классы. Поэтому у Администратора есть возможность переводить классы, при помощи который создается новый класс, наследуемый от уже существующего, в который переносятся все ученики из «старого» класса.

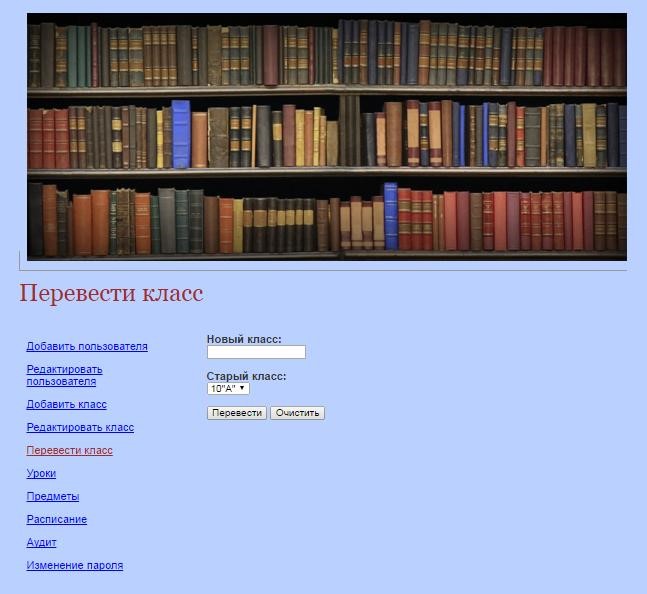


Рисунок 18 – Переведение класса в новый учебный год

Далеко не у всех школ расписание уроков совпадают друг с другом с точностью до минуты. У одних школ урок длится на 5 минут больше, а у другой школы перемена в два раза короче.

Учитывая все это, напрашивается вывод, что каждой школе нужно задавать свои собственные временные ограничения, что и позволяет сделать вкладка «Уроки», на которой надо всего лишь ввести номер урока, его начало и окончание.

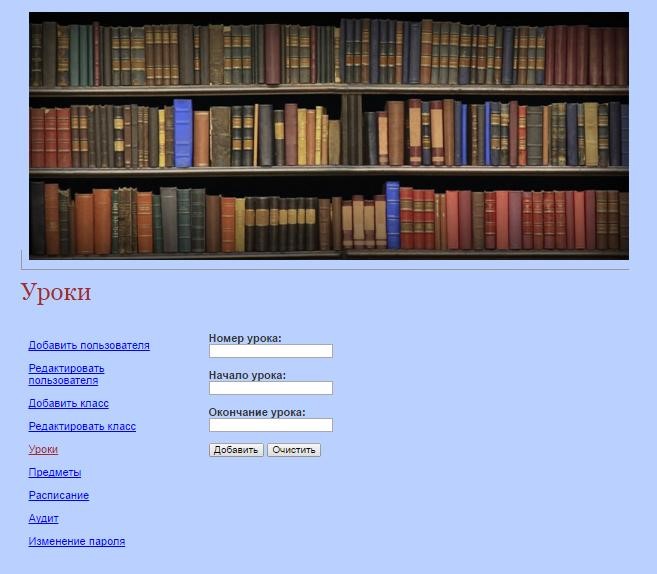


Рисунок 19 – Добавление уроков

На странице «Предметы» Администратор может задавать предметы, которые будут преподавать в учебной заведении, указывать у какого класса он будет вестись, и кто будет его преподаватель.

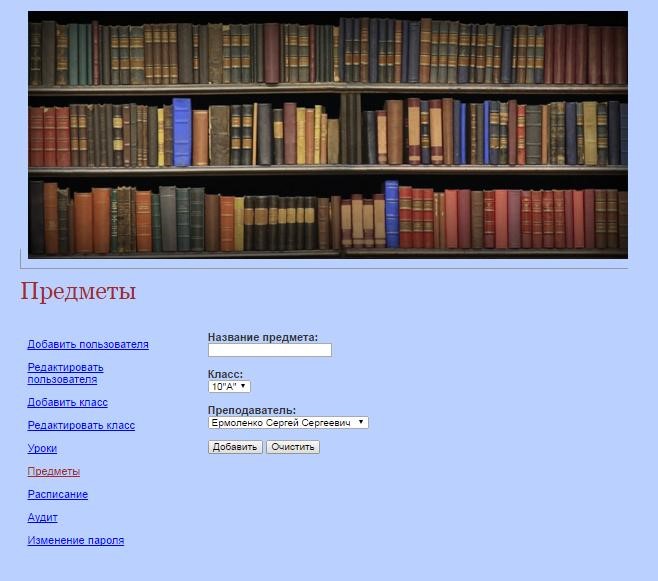


Рисунок 20 – Добавление предметов

Также у Администратора есть возможность задавать расписание уроков у каждого конкретного класса. Для этого на странице «Расписание» ему следует выбрать класс, предмет, номер урока, день недели, номер аудитории и нажать кнопку «Добавить».

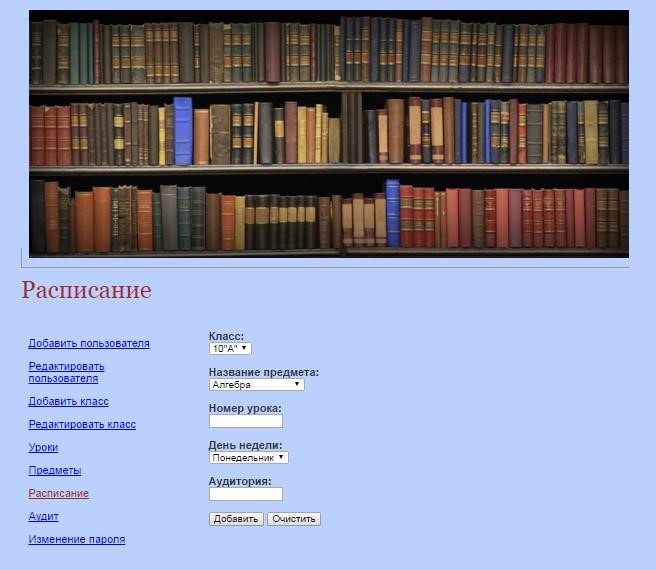


Рисунок 21 – Добавление расписания

Ну и напоследок самая главная функция Администратора с точки зрения безопасности веб-сайта – его аудит. На соответствующей странице Администратору достаточно лишь выбрать пользователя и интересующий его аудит, нажать кнопку «Далее» и перед ним появится вся необходимая ему информация.

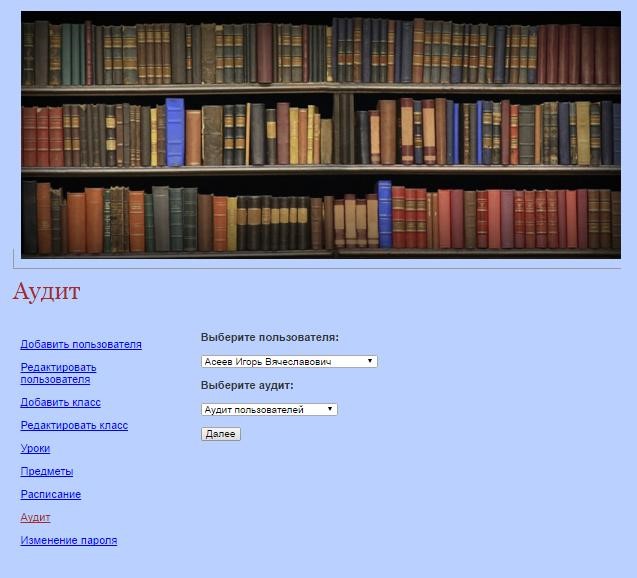


Рисунок 22 – Выбор аудита

На рисунке ниже изображен результат аудита пользователей. На нем показаны все изменения, который делал конкретный пользователей (в данном случае, администратор) за все время.

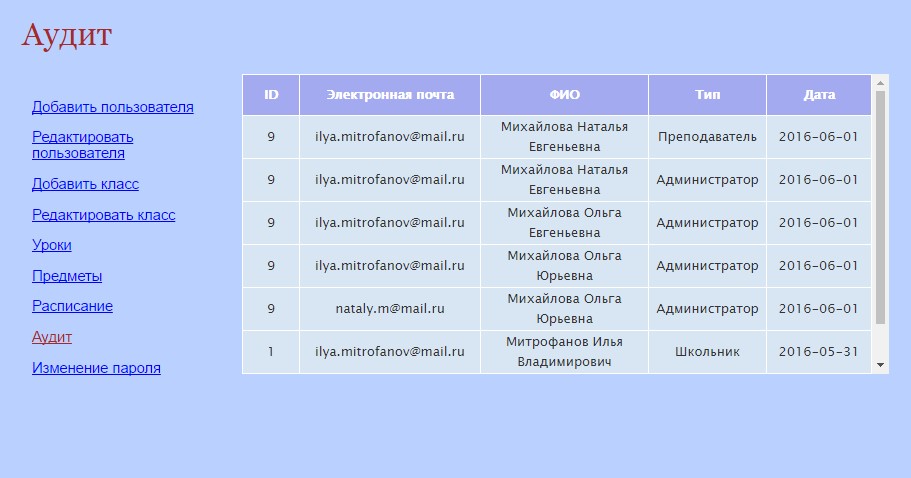


Рисунок 23 – Аудит пользователей

## 3.7 Интерфейс директора

Любому Директору школы необходимо знать, что происходит в подвластном ему учебном заведении. Зайдя на вкладку «Статистика», он сможет найти всю интересующую его информацию. Там ему необходимо будет выбрать класс (или несколько классов) и интересующую его статистику. Их в дневнике пять: «Отличники», «Хорошисты», «Троечники», «Неуспевающие» и «Посещаемость». Посещаемость считается по количеству пропущенных уроков, а остальные статистики – по промежуточным оценкам (четверти, полугодия).



Рисунок 24 – Выбор интересующей статистики

Нажав на кнопку «Далее», он переходит на новую страницу, где и отображаются результаты запроса. На рисунке ниже изображен результат запроса всех отличников в 11«А» классе.

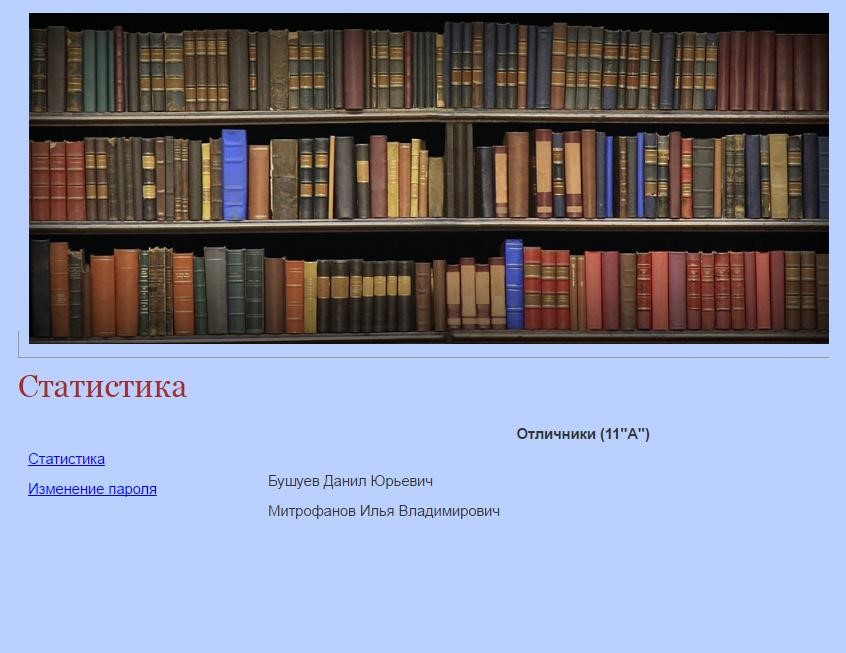


Рисунок 25 – Статистика отличников в 11«А» классе

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы бакалавра были изучены средства для веб-программирования (языки PHP, HTML, CSS, JavaScript и SQL) и создан веб-сайт, представляющий собой электронный дневник школы. Данный сайт был оснащен такими функциональными возможностями, как регистрация школы, добавление пользователей различных типов (Ученик, Родитель, Преподаватель, Администратор и Директор), добавление уроков (их времени и аудитории), добавления классов, предметов, расписания, сообщений в новостную ленту и добавления, редактирования и удаления оценок. Были изучены потенциально слабые места веб-сайта с точки зрения его безопасности и предприняты меры для предотвращения нежелательных явлений. В комплекс мер по безопасности сайта вошли защита от SQL-инъекций, разделение привилегий различных типов пользователей, хеширование с использованием соли персональных данных пользователей (паролей), а также создание аудита сайта, показывающего различные

«подозрительные» действия пользователей. Поставленные задачи были полностью решены, результаты отражены в третьем разделе основной части работы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. PHP: Hypertext Preprocessor. [Электронный ресурс], URL: www.php.net, [Дата обращения: 05 мая 2016].
2. HTMLBOOK: Для тех, кто делает сайты. [Электронный ресурс], URL: www.htmlbook.ru, [Дата обращения: 21 апреля 2016].
3. Современный учебник JavaScript. [Электронный ресурс], URL: www.javascript.ru, [Дата обращения: 10 мая 2016].
4. Руководство по CSS для начинающих. [Электронный ресурс], URL:

www. technologyweb.org, [Дата обращения: 21 мая 2016].

1. Интерактивный учебник по SQL. [Электронный ресурс], URL: www.sql-tutorial.ru, [Дата обращения: 07 мая 2016].
2. Введение в SQL. [Электронный ресурс], URL: www.mysql.ru, [Дата обращения: 07 мая 2016].
3. PhpMyAdmin. [Электронный ресурс], URL: www.phpmyadmin.net,

[Дата обращения: 19 апреля 2016].

ПРИЛОЖЕНИЕ А

# Исходный код

style.css

body {

font-family: Arial, Verdana, sans-serif; font-size: 11pt; background-color: #BAD0FF;

color: #333; margin-left: 30px; margin-top: 2px;

}

h1 { color: #a52a2a; margin-left: 10px; font-size: 24pt; font-family: Georgia, Times, serif; font-weight: normal;

}

p { text-align: justify; margin-left: 10px; margin-right: 10px;

}

p.active a {

color: #a52a2a;

}

p0 { text-align: justify; margin-left: 10px; margin-right: 10px; border-left: 1px solid #999; border-bottom: 1px solid #999; padding-left: 10px; padding-bottom: 10px;

}

p1

{ text-align: left; margin-left: 10px; margin-right: 10px; border-left: 1px solid #999; border-bottom: 1px solid #999; padding-left: 10px; padding-bottom: 10px;

}

p2 { margin-left: 0px; margin-right: 0px;

}

aside

{ background: #BAD0FF; padding: 10px;

width: 200px; float: left; }

#content1 { text-align: center; float:left; width:750px; height:180px; margin-left: 45px; box-shadow: 0 0 10px rgba(0,0,0,0.5); background: #fff;

}

#content2 { text-align: center; float:left; width:650px; height:180px; box-shadow: 0 0 10px rgba(0,0,0,0.5); background: #fff;

}

#content3 { text-align: center; float:left; margin-left: 30px; width:650px; height:250px;

background: #BAD0FF;

}

#error1 { text-align: center; float:left; width:850px; height:250px; background: #BAD0FF;

}

#error2 { text-align: center;

float:left; width:670px; height:250px;

background: #BAD0FF;

}

#homework

{ text-align: center;

float:left; margin-left: 130px; width:500px; height:250px;

background: #BAD0FF;

}

#container1 { float:left; margin-left: 10px; width: 250px; }

#container2 { width: 385px; margin-top: 20px; overflow: auto; }

#right1 { width:400px; height:100px; float:right;

background: #BAD0FF;

}

#right2 { width:300px; height:100px;

float:right;

background: #BAD0FF;

}

#school

{ width:350px; height:200px; font-size: 30pt; float:right; text-align: center; margin-left: 30px; margin-right: 50px;

background: #BAD0FF;

}

table {

font-family: "Lucida Sans Unicode", "Lucida Grande", Sans-Serif; font-size: 13px; border-collapse: collapse; border: 1px solid white; width: 250px; text-align: center;

}

th

{

background: #A4AAF0; color: white; padding: 10px 12px;

}

th, td { border-style: solid; border-width: 0 1px 1px 0; border-color: white;

}

td {

background: #D8E6F3;

}

A {

color: #0000FF;

}

A:visited {

color: ##0000FF;

}

header.css

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"

"http://www.w3.org/TR/html4/strict.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<link rel="stylesheet" href="style.css">

<?

if ($\_SESSION['type'] == 'Школьник')

{

|  |  |
| --- | --- |
| ?> |  |
|  | <title>Ученик</title> |
|  | </head> |
|  | <body> |
| src="books.jpg"></a></p1>  <?  } | <br><p1><a href="parent.php" title="Главная"><img |

if ($\_SESSION['type'] == 'Родитель')

{

?>

|  |  |
| --- | --- |
|  | <title>Родитель</title> |
|  | </head> |
|  | <body> |
| src="books.jpg"></a></p1>  <?  } | <br><p1><a href="child.php" title="Главная"><img |

if ($\_SESSION['type'] == 'Преподаватель')

{

?>

|  |  |
| --- | --- |
|  | <title>Преподаватель</title> |
|  | </head> |
|  | <body> |
| src="books.jpg"></a></p1>  <?  } | <br><p1><a href="teacher.php" title="Главная"><img |

if ($\_SESSION['type'] == 'Администратор')

{

?>

|  |  |
| --- | --- |
|  | <title>Администратор</title> |
|  | </head> |
|  | <body> |
| src="books.jpg"></a></p1>  <?  } | <br><p1><a href="admin.php" title="Главная"><img |

if ($\_SESSION['type'] == 'Директор')

{

?>

<title>Директор</title>

</head> <body>

<br><p1><a href="director.php" title="Главная"><img src="books.jpg"></a></p1>

<?

}

?>

<div id="school"><br><br><?echo $\_SESSION['school']?></div>

<script type="text/javascript" src="jquery-1.11.1.js"></script>

menu.css

<aside id="menu">

<?

if ($\_SESSION['type'] == 'Родитель')

{

?>

<p><a href="child.php">Ребенок</a></p>

<ul>

<li>

<a href="parent.php">Класс</a>

</li>

<li>

<a href="homework.php">Домашнее задание</a>

</li>

<li>

<a href="timetable.php">Расписание</a>

</li>

<li>

<a href="valuation.php">Оценки</a>

</li>

</ul>

<p><a href="edit.php">Изменение пароля</a></p>

<?

}

if ($\_SESSION['type'] == 'Преподаватель')

{

?>

<p><a href="teacher\_homework.php">Домашнее задание</a></p>

<p><a href="teacher\_mark.php">Оценки</a></p>

<p><a href="teacher\_news.php">Новости</a></p> <p><a href="parent.php">Руководимый класс</a></p>

<ul>

<li>

<a href="parent.php">Класс</a>

</li>

<li>

<a href="homework.php">Домашнее задание</a>

</li>

<li>

<a href="timetable.php">Расписание</a>

</li>

<li>

<a href="valuation.php">Оценки</a>

</li>

</ul>

<p><a href="edit.php">Изменение пароля</a></p>

<?

}

if ($\_SESSION['type'] == 'Школьник')

{

?>

<p><a href="homework.php">Домашнее задание</a></p>

<p><a href="timetable.php">Расписание</a></p>

<p><a href="valuation.php">Оценки</a></p> <p><a href="edit.php">Изменение пароля</a></p>

<?

}

if ($\_SESSION['type'] == 'Администратор')

{

?>

<p><a href="add\_user.php">Добавить пользователя</a></p>

<p><a href="edit\_user.php">Редактировать пользователя</a></p>

<p><a href="add\_class.php">Добавить класс</a></p> <p><a href="edit\_class.php">Редактировать класс</a></p>

<p><a href="transfer\_class.php">Перевести класс</a></p>

<p><a href="add\_lesson.php">Уроки</a></p>

<p><a href="add\_subject.php">Предметы</a></p>

<p><a href="add\_timetable.php">Расписание</a></p>

<p><a href="audit.php">Аудит</a></p>

<p><a href="edit.php">Изменение пароля</a></p>

<?

}

if ($\_SESSION['type'] == 'Директор')

{

?>

<p><a href="statistics1.php">Статистика</a></p> <p><a href="edit.php">Изменение пароля</a></p>

<?

}

?>

</aside>

<script language="JavaScript" type="text/javascript">

$("aside#menu p").each(function ()

{

if (this.getElementsByTagName("a")[0].href == location.href)

this.className = "active";

}

);

</script>

login1.php

<div id="right2">

<br><br>

<p><b>Пользователь:</b><br>

<p><b><?echo $\_SESSION['surname'], ' ', $\_SESSION['name']?></b></p> <p><button onclick="location.href = 'home.php'">Выйти</button></p>

</div>

add\_user.php

<?

session\_start(); unset($\_SESSION['error']);

if (!isset($\_SESSION['id']))

{

header("Location: home.php"); exit();

}

if ($\_SESSION['type'] != "Администратор")

{

$\_SESSION['error'] = "У вас нет доступа к этой странице."; header("Location: error.php"); exit();

}

$link=mysqli\_connect("localhost", "root", "123", "1");

$class = mysqli\_query($link,"SELECT id, name FROM class ORDER BY name"); $child = mysqli\_query($link,"SELECT id, surname, name, patronymic FROM users WHERE type='Школьник' ORDER BY surname");

require\_once '/template/header.php';

?>

<h1>Добавить пользователя</h1>

<? require\_once '/template/menu.php';

?>

<div id="content3">

<form method="POST" action="register\_user.php">

<p><b>Логин:</b><br>

<input name="login" type="text" size="40" required> </p>

<p><b>Электронная почта:</b><br>

<input name="email" type="email" size="40" required> </p>

<p><b>Тип пользователя:</b><br>

<select size="1" name="type[]" required>

<option value='Школьник'>Ученик</option>

<option value='Родитель'>Родитель</option>

<option value='Преподаватель'>Преподаватель</option>

<option value='Администратор'>Администратор</option>

<option value='Директор'>Директор</option>

</select>

<script language="JavaScript" type="text/javascript">

$('select').change(function ()

{

if (($(this).val() === 'Преподаватель')||($(this).val() === 'Администратор')||($(this).val() ===

'Директор')||($(this).val() === 'Родитель'))

{

$("#ghost1").hide();

} else

{

$("#ghost1").show();

}

});

</script>

<script language="JavaScript" type="text/javascript">

$("#ghost2").hide();

$('select').change(function ()

{

if (($(this).val() === 'Преподаватель')||($(this).val() === 'Администратор')||($(this).val() === 'Директор')||($(this).val() === 'Школьник'))

{

$("#ghost2").hide();

} else

{

$("#ghost2").show();

}

});

</script>

<p><b>Фамилия:</b><br>

<input name="surname" type="text" size="40" required> </p>

<p><b>Имя:</b><br>

<input name="name" type="text" size="40" required> </p>

<p><b>Отчество:</b><br>

<input name="patronymic" type="text" size="40" required> </p>

<div id='ghost1'>

<p><b>Классы:</b><br>

<select size="1" name="class[]" required>

<?

while ($row = mysqli\_fetch\_assoc($class))

{

?>

<option value="<? echo $row['id'] ?>"><?echo $row['name'] ?></option>

<?

}

?>

</select>

</div>

<div id='ghost2'>

<p><b>Выберите ребенка:</b><br>

<select size="3" multiple name="child[]" required>

<?

while ($row = mysqli\_fetch\_assoc($child))

{

?>

<option selected value="<? echo $row['id'] ?>"><? echo $row['surname'], ' ', $row['name'], ' ', $row['patronymic'] ?></option>

<? }

?>

</select>

</div>

<script language="JavaScript" type="text/javascript">

$("#ghost2").hide();

</script>

<p><input type="submit" value="Добавить">

<input type="reset" value="Очистить"></p>

</form><br>

</div>

<? require\_once '/template/login.php';

?>

<?

if (isset($\_SESSION['success']))

{

?>

<script language="JavaScript" type="text/javascript">

alert("Пользователь успешно добавлен."); </script> <?

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

}

?>

</body>

</html>

register\_user.php

<?

session\_start();

$link=mysqli\_connect("localhost", "root", "123", "1");

if(!preg\_match("/^[a-zA-Z0-9]+$/",$\_POST['login']))

{

$\_SESSION['error'] = "Логин может состоять только из букв английского алфавита и цифр.";

}

if(strlen($\_POST['login']) < 3 or strlen($\_POST['login']) > 30)

{

$\_SESSION['error'] = "Логин должен быть не меньше 3-х символов и не больше 30.";

}

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

$query = mysqli\_query($link, "SELECT \* FROM users WHERE login='$login'"); if(mysqli\_num\_rows($query) != 0)

{

$\_SESSION['error'] = "Пользователь с таким логином уже существует в базе данных.";

}

if (!isset($\_SESSION['error']))

{

$pas = generate\_pas(10);

$password = password\_hash($pas, PASSWORD\_DEFAULT);

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

$message = "Ваш пароль - $password"; mail($email, 'Регистрация', $message);

$type = $\_POST['type'][0];

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

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

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

$class = $\_POST['class'][0];

mysqli\_query($link,"INSERT INTO users SET login='$login', password='$password', email='$email', type='$type', surname='$surname', name='$name', patronymic='$patronymic', class='$class', salt='$salt'");

$\_SESSION['success'] = "true"; header("Location: add\_user.php"); exit();

} else

{

header("Location: error.php"); exit();

}

?>

login.php

<?

session\_start();

if (!isset($\_SESSION['id']))

{

$link=mysqli\_connect("localhost", "root", "123", "1");

$query = mysqli\_query($link,"SELECT id, password, type, surname, name, salt FROM users

WHERE login='".mysqli\_real\_escape\_string($link,$\_POST['login'])."' LIMIT 1");

$data = mysqli\_fetch\_assoc($query);

$school1 = mysqli\_query($link,"SELECT school\_name FROM settings");

$school2 = mysqli\_fetch\_assoc($school1);

if($data['password'] == md5(md5($\_POST['password']))) //if (password\_verify($\_POST['password'], $data['password']))

{

$\_SESSION['id']=$data['id'];

$\_SESSION['surname']=$data['surname'];

$\_SESSION['name']=$data['name'];

$\_SESSION['school']=$school2['school\_name'];

$\_SESSION['type']=$data['type'];

if ($data['type'] === "Школьник")

{

header("Location: parent.php"); exit();

}

if ($data['type'] === "Родитель")

{

header("Location: child.php"); exit();

}

if ($data['type'] === "Преподаватель")

{

header("Location: teacher.php"); exit();

}

if ($data['type'] === "Администратор")

{

header("Location: admin.php"); exit();

}

if ($data['type'] === "Директор")

{

header("Location: director.php"); exit();

} } else {

$\_SESSION['error'] = "Вы ввели неправильный логин/пароль."; header("Location: error.php"); exit();

}

}

?>

add\_class.php

<?

session\_start();

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

if (!isset($\_SESSION['id']))

{

header("Location: home.php"); exit();

}

if ($\_SESSION['type'] != "Администратор")

{

$\_SESSION['error'] = "У вас нет доступа к этой странице."; header("Location: error.php"); exit();

}

$link=mysqli\_connect("localhost", "root", "123", "1");

$res = mysqli\_query($link,"SELECT id, surname, name, patronymic FROM users WHERE type='Преподаватель' ORDER BY surname");

require\_once '/template/header.php';

?>

<h1>Добавить класс</h1>

<? require\_once '/template/menu.php';

?>

<div id="content3">

<form method="POST" action="register\_class.php">

<p><b>Название:</b><br>

<input name="name" type="text" size="15" required>

</p>

<p><b>Классный руководитель:</b><br> <select size="1" name="form\_master[]" required>

<?

while ($row = mysqli\_fetch\_assoc($res))

{

?>

<option value="<? echo $row['id'] ?>"><? echo $row['surname'], ' ', $row['name'], ' ', $row['patronymic'] ?></option>

<?

}

?>

</select>

<p><b>Новости класса:</b><br>

<p><textarea rows="8" cols="50" name="news" required></textarea></p>

<p><button>Добавить</button>

<input type="reset" value="Очистить"></p>

</form><br>

</div>

<? require\_once '/template/login.php';

?>

<?

if (isset($\_SESSION['success']))

{

?>

<script language="JavaScript" type="text/javascript"> alert("Класс успешно добавлен."); </script> <?

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

}

?>

</body>

</html>

register\_class.php

<?

session\_start();

$link=mysqli\_connect("localhost", "root", "123", "1");

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

$query = mysqli\_query($link, "SELECT \* FROM class WHERE name='$name'"); if(mysqli\_num\_rows($query) != 0)

{

$\_SESSION['error'] = "Класс с таким именем уже существует.";

}

if (!isset($\_SESSION['error']))

{

$form\_master = $\_POST['form\_master'][0];

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

mysqli\_query($link,"INSERT INTO class SET name='$name',

classroom\_teacher\_id='$form\_master', news='$news'");

$\_SESSION['success'] = "true";

header("Location: add\_class.php"); exit();

} else {

header("Location: error.php"); exit();

}

?>

register\_news.php

<?

session\_start();

$link=mysqli\_connect("localhost", "root", "123", "1");

if (!isset($\_SESSION['error']))

{

$class = $\_POST['class'][0];

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

$stmt = $link->prepare("UPDATE class SET news=? WHERE id=?");

$stmt->bind\_param("si", $news, $class);

$stmt->execute();

$\_SESSION['success'] = "true";

header("Location: teacher\_news.php"); exit();

} else

{

header("Location: error.php"); exit();

}

?>