ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ПО УЧЁТУ ПОСЕЩАЕМОСТИ СТУДЕНТОВ ИНСТИТУТА ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

Выпускная квалификационная работа
обучающегося по направлению подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем
очной формы обучения,
группы 07001402
Варданяна Александра Георгиевича

Научный руководитель
ст. п. Ерошенко Я.Б.

БЕЛГОРОД 2018

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Учебная посещаемость всегда являлась и является одним из самых острых вопросов образовательного процесса. Недобросовестное посещение студентами учебных занятий влечёт за собой ряд серьёзных проблем, как и для обучающегося, так и для общества в целом. Например, падает академическая успеваемость студента и, как следствие, снижается качество подготовки специалистов. Это, в свою очередь, негативно сказывается на конкурентоспособности выпускника при дальнейшем его трудоустройстве.

Из этого вытекает нерациональная трата преподавательского и административного времени, следовательно, увеличиваются расходы на обеспечение образовательного процесса.

С развитием информационных технологий данная проблема требует пересмотра традиционных подходов к её решению. Существующие на данный момент журналы учёта посещаемости, которые по-прежнему ведутся бумажной форме имеют ряд недостатков:

- трудоёмкость процесса подсчёта пропущенных занятий;
- возможность преднамеренного искажения учащимися информации в журнале;
- затруднительность проведения анализа причин отсутствия студентов на занятиях и, как следствие, невозможность проведения коррекции учебного процесса с целью повышения посещаемости.

В 21 декабря 2012 году Министерство образования и науки Российской Федерации разработало федеральный закон “Об образовании в Российской Федерации” [1].

Где одним из главных положений является создание электронного журнала с целью повышение качества образования за счёт:

- повышение уровня прозрачности учебного процесса;
- автоматизации учётных функций;

- комфортности ведения учёта и анализа учебной деятельности;
- повышения надёжности хранения информации.

На протяжении шести лет с момента публикации официальных рекомендаций по внедрению систем электронного учёта было разработано большое количество различных комплексов с целью проведения мониторинга посещаемости занятий учащимися.

В качестве примера одной из систем, предоставляющих возможность учёта посещаемости учащихся, можно привести популярный веб-сервис компании Первый Бит “БИТ.ВУЗ.Учебная часть”. Данный сервис в полной мере обеспечивает электронный учёт посещаемости, как и другие его аналоги. Но поскольку такие сервисы, как правило, реализованы на платформе 1С, то они имеют завышенную цену эксплуатации.

Именно поэтому разработка информационной системы для учёта посещаемости студентов института Инженерных технологий и естественных наук является актуальной на сегодняшний день.

Целью данной выпускной квалификационной работы является проектирование и разработка информационной системы для учёта посещаемости студентами учебных занятий.

Для достижения указанной цели необходимо было решить следующие задачи:

1. Анализ деятельности института, требующей информационного сопровождения.
2. Определение требований к системе.
3. Изучение фреймворка Laravel.
4. Проектирование базы данных для информационной системы.
5. Проектирование веб-интерфейса для информационной системы.
6. Разработать веб-приложение по учёту посещаемости студентов.
7. Тестирование.

Первая глава выпускной квалификационной работы содержит анализ предметной области, теоретические аспекты, анализ и постановку требований к результирующему продукту.

Во второй главе описывается выбор программного обеспечения для реализации поставленной задачи и проектирование базы данных.

Третья глава выпускной квалификационной работы посвящена подробному описанию процесса разработки информационной системы.

В четверной главе выпускной квалификационной работы описывается тестирование разработанной информационной системы согласно заранее подготовленным тестовым данным.

Данная выпускная квалификационная работа включает в себя 92 страницу, 38 иллюстраций, 2 таблицы и приложение.

ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Описание структуры института Инженерных технологий и естественных наук

Белгородский государственный национальный исследовательский университет – это одно из старейших высших учебных заведений города Белгорода, а также крупнейший вуз Белгородской области.

В 2017 году БелГУ занял семьдесят шестую позицию из ста возможных в одном из главных рейтингов – Шанхайском предметном рейтинге университетов Global Ranking of Academic Subjects (ARWU) [2].

Белгородский национальный исследовательский университет является многопрофильным научно-образовательным комплексом, включающим в себя семь институтов и четыре отдельных факультета, а также Медицинский колледж и один филиал, расположившийся в Старом Осколе.

На основании приказа ректора университета от 7.07.2014 № 583-ОД был создан Институт инженерных технологий и естественных наук, объединивший три факультета: информационных технологий и прикладной математики, инженерно-физический и биолого-химический.

Это объясняется промежуточным положением технического знания между естественными и науками гуманитарного профиля в общей системе научного знания. Анализ различных сторон этого взаимодействия становится всё более актуальным для осмысления характерных особенностей научно-технического прогресса, специфики его современного этапа. Таким образом, главной задачей Института объективно стала необходимость достижения синергетического эффекта от объединения инженеров, биологов, физиков, химиков, программистов и математиков. Эта тенденция проявляется Естественнонаучное знание вскрывает закономерности, свойства и связи предметов и явлений окружающей нас природной среды, объясняет их;

научно-техническое же знание выявляет способы, средства и методы для возможного эффективного использования этих закономерностей в практических целях, а также изучает так называемые «вторичные» качества и связи природных явлений и предметов, возникающие в процессе их технического освоения.

Обучение студентов осуществляется в соответствии с европейскими стандартами непрерывного трехуровневого высшего образования: бакалавриат, магистратура, аспирантура.

Основные усилия работы Института направлены на подготовку кадров в области инженерных и естественных наук.

На основании решения Ученого совета от 25.05.2015 «О внесении изменений в структуру образовательных подразделений НИУ «БелГУ» (приказ ректора НИУ «БелГУ» №467-ОД от 23.06.2015) внесены изменения в структуру института Инженерных технологий и естественных наук, важнейшими элементами которой являются советы по областям научных знаний, определяющие вектор развития соответствующих направлений:

- Направление математики и информатики;
- Инженерно-физическое направление;
- Биолого-химическое направление.

В Институте инженерных технологий и естественных наук для анализа профессиональной компетенции студентов в рамках освоения учебных дисциплин используется информационная система “Гермес-КНИТ БелГУ”. Данная система позволяет вести учёт, производить контроль и служит вспомогательным средством для проверки лабораторных работ. Также в этой системе можно вести учёт посещаемости студентами Института занятий. Но данная система используется только на направлении математики и информатики. И каких-либо других информационных систем для контроля посещаемости студентов на данный момент не имеется. Именно поэтому разработка новой информационной системы для учёта посещаемости учащихся Института являет актуальной задачей [3].

1.2 Теоретические основы разработки и проектирования информационных систем. Основные модели разработки

Информационная система – это система, предназначенная для хранения, поиска и обработки информации, и соответствующие организационные ресурсы, которые обеспечивают и распространяют информацию [4].

Такие системы служат для своевременного обеспечения надлежащей информацией определённого круга людей, если говорить иначе, то для удовлетворения конкретных информационных потребностей в рамках определённой предметной области.

Информационная продукция – это результат функционирования информационной системы. Информационной продукцией являются: юридические, нормативно-технические и другие виды документов, информационные услуги, базы данных, а также информационные массивы.

Говоря о моделях разработки информационной системы понимают различные состояния, начиная с момента в потребности данной системы и заканчивая моментом выхода из эксплуатации.

Обычно жизненный цикл ИС составляет 4 стадии:

1. Разработка концепции системы;
2. Составление задания для дальнейшего проектирования;
3. Проектирование ИС, а также разработка блок-схемы для решения поставленной задачи;
4. Составление модели жизненного цикла ИС.

Существует множество моделей жизненного цикла информационных систем, но на данный момент наибольшее распространение получили:

- Спиральная;
- Инкрементная;
- Каскадная.

1.2.1 Спиральная модель

Спиральная стратегия подразумевает разработку в виде последовательности версий, но в начале проекта определены не все требования. Требования уточняются в результате разработки версий.

Спиральная модель жизненного цикла характерна при разработке нетиповых информационных систем. В начале работы над проектом у заказчика и разработчика нет четкого видения итогового продукта или стопроцентной уверенности в успешной реализации проекта. Именно поэтому принимается решение разработки информационной системы по частям с возможностью изменения требований или отказа от ее дальнейшего развития. На рисунке 1.2. изображена спиральная модель.

Рис. 1.2. Спиральная модель

Достоинства данной модели:

1. Позволяет в короткие сроки получить пользователям системы работоспособный продукт, тем самым, активизировать процесс уточнения и дополнения требований;
2. Допускает изменение требований при разработке информационной системы, что характерно для большинства разработок, в том числе и типовых;
3. Обеспечивает большую гибкость в управлении проектом;
4. Позволяет получить более надежную и устойчивую систему. По мере развития системы ошибки и слабые места обнаруживаются и исправляются на каждой итерации;
5. Позволяет совершенствовать процесс разработки – анализ, проводимый в каждой итерации, позволяет проводить оценку того, что должно быть изменено в организации разработки, и улучшить ее на следующей итерации;
6. Уменьшаются риски заказчика, который может с минимальными для себя финансовыми потерями завершить развитие неперспективного проекта.

К недостаткам спиральной модели разработки ИС можно отнести:

1. Увеличение неопределенности у разработчика в перспективах развития проекта. Этот недостаток вытекает из предыдущего достоинства модели;
2. Затруднение операции временного и ресурсного планирования всего проекта в целом. Для решения этой проблемы необходимо ввести временные ограничения на каждую из стадий жизненного цикла. Переход осуществляется в соответствии с планом, даже если не вся запланированная работа выполнена. План составляется на основе статистических данных, полученных в предыдущих проектах и личного опыта разработчиков[5].

1.2.2 Инкрементная модель

Инкрементная стратегия подразумевает разработку ИС с линейной последовательностью стадий, но в несколько этапов для дальнейшего улучшения продукта.

В начале работы над проектом определяются все основные требования к системе, после чего выполняется ее разработка в виде последовательности версий. При этом каждая версия является законченным и работоспособным продуктом. Первая версия реализует часть запланированных возможностей, следующая версия реализует дополнительные возможности и т. д., пока не будет получена полная система. На рисунке 1.3. изображена инкрементная стратегия.

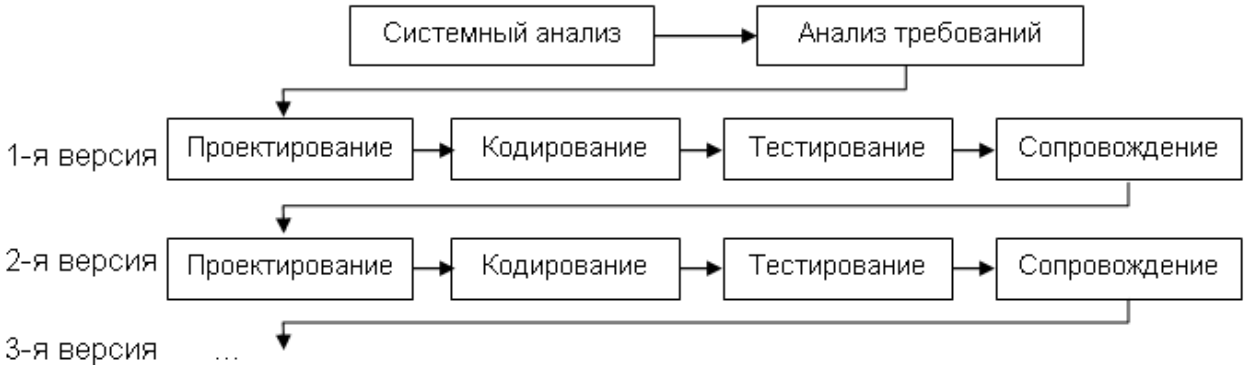


Рис. 1.4. Инкрементная модель

К достоинствам данной модели можно отнести то, что:

1. На каждой стадии формируется законченный набор документации, программного и аппаратного обеспечения, отвечающий критериям полноты и согласованности;
2. Выполняемые в четкой последовательности стадии позволяют уверенно планировать сроки выполнения работ и соответствующие ресурсы.

К недостаткам икрементной модели разработки ИС можно отнести:

1. Реальный процесс разработки информационной системы редко полностью укладывается в такую жесткую схему. Особенно это относится к разработке нетиповых и новаторских систем;
2. Основана на точной формулировке исходных требований к информационной системе. Реально в начале проекта требования заказчика определены лишь частично;
3. Основной недостаток – результаты разработки доступны заказчику только в конце проекта. В случае неточного изложения требований или их изменения в течение длительного периода создания ИС заказчик получает систему, не удовлетворяющую его потребностям[6].

1.2.3 Каскадная модель

Для реализации поставленной задачи применялась каскадная модель. Данная стратегия подразумевает линейную последовательность выполнения стадий создания информационной системы.

На рисунке 1.4. изображена линейная последовательность прохождения этапов создания.

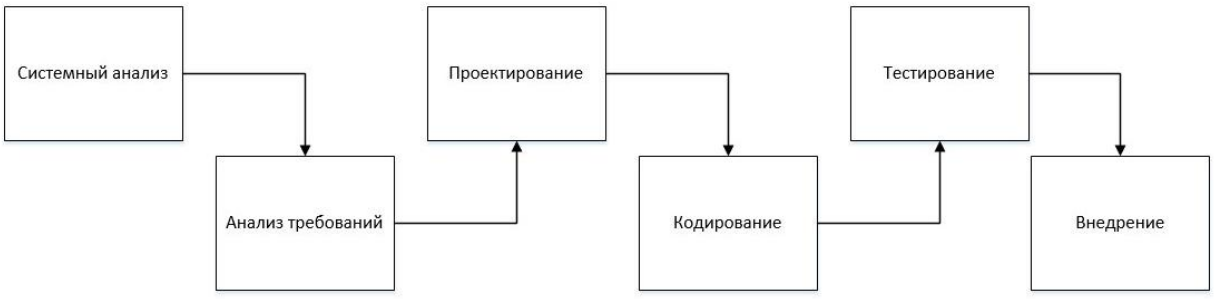


Рис. 1.4. Этапы создания информационной системы

Данная модель используется при разработке информационных систем, для которых в самом начале проектирования становится возможным достаточно точно и полно сформулировать все требования.

Достоинства и недостатки каскадной модели такие же, как и у инкрементной модели. Но в отличие от инкрементной стратегии заказчик может позже увидеть результаты.

1.3 Требования к информационной системе

После изучения теоретического материала по разработке и проектированию ИС необходимо установить требования к информационной системе в целом. Система должна обеспечивать:

- Полноту информации для каждого звена системы;
- Полезность и ценность информации;
- Точность и достоверность информации;
- Актуальность информации;
- Экономичность и эффективность информации.

Также информационная система должна удовлетворять таким техническим требованиям как:

- Быстродействие;
- Надёжная защита;
- Удобный пользовательский интерфейс;
- Возможность развития системы;
- Высокая надёжность работы системы.

Подводя итоги первой главы, можно сделать вывод, что результирующая информационная система должна:

- Осуществлять дополнение, удаление, изменение и хранение информации. Данный функционал должен быть доступен в зависимости от роли пользователя в информационной системе;
- Реализовать простой и понятный любому пользователю интерфейс;
- Реализовать функционал, позволяющий получать отчётность по заданным критериям, например по фамилии, имени, отчеству студента или группе студента. Данный функционал должен быть доступен только для управляющего персонала Института.

1.4 Интегрированные среды разработки

Интегрированная среда разработки (Integrated development environment - IDE) – это комплекс программных средств, для разработки программы или множества программ, используемых для управления компьютером[7].

Среда разработки включает в себя:

1. Программу для изменения и создания текстовых данных.
2. Программу, выполняющую компиляцию или, а иногда и, интерпретацию.
3. Средства автоматизации сборки.
4. Программу для поиска ошибок. Например, в других программах, ядрах операционных систем, SQL-запросах и других видах кода.

Некоторые интегрированные среды предназначены только для одного определённого языка программирования – такие как Visual Basic, но большинство из них предназначены для нескольких языков программирования – такие как Microsoft Visual Studio или Atom.

Atom или более раннее название Atomicity – бесплатная интегрированная среда разработки с открытым исходным кодом. Данная IDE

поддерживает независимо компилируемые программные модули, которые представлены в виде библиотек и служат для расширения возможностей среды разработки.

К таким библиотекам относятся:

- TypeScript & JavaScript - ide-typescript;
- Flow - ide-flowtype;
- C# - ide-csharp;
- Java - ide-java;
- PHP - ide-php.

1.5 Системы управления базами данных

Развитие технических средств и средств программирования сделало возможным накопление и обработку больших объемов данных. В настоящее время определяющим направлением организации и обработки файлов стала концепция баз данных.

База данных (БД) – это совокупность взаимосвязанных данных, используемых группой пользователей и хранящаяся с регулируемой избыточностью. Хранимые данные не зависят от программ пользователя. Именно то, что на сегодняшний день информация о структуре данных способна хранится непосредственно в файлах данных, при этом формат файла является стандартизированным, составляет основное положение концепции БД [8].

Широкое использование БД обусловлено оперативностью, гибкостью в получении ответов на любые запросы, наличии эффективных методов модификации данных и внесении изменений, безопасностью, целостностью, доступностью (возможность использования всей совокупности данных в БД для каждого пользователя).

При проектировании БД предметная область рассматривается в виде реального представления объекта, его описания в формальном виде и данных, которые отражают это представление. Информационная модель считается базовым понятием[9].

Система управлениями базами данных (СУБД) – это совокупность программных и языковых средств назначения, обеспечивающих управление базами данных[10].

Основными функциями СУБД являются:

- Управление данными во внешней и оперативной памяти;
- Сохранение информации для восстановления базы данных после сбоев в работе;
- Поддержка языков баз данных.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ

2.1 Выбор программного обеспечения для реализации

Главным преимуществом любого веб-приложения принято считать отсутствие надобности установки дополнительного программного обеспечения, так как нынешние операционные системы имеют заранее установленный браузер. Например ОС Windows поставляется вместе с браузером Internet Explorer.

Для реализации проекта был выбран один из лидеров среди языков, применяющихся для разработки веб-сайтов, PHP.

2.1.1 Выбор фреймворка

Для разработки программно-аппаратной части веб-приложения был выбран фреймворк Laravel.

Laravel – это бесплатный, кроссплатформенный, написанный на языке программирования PHP, веб-фреймворк с открытым кодом, предназначенный для разработки веб-приложений с использованием архитектурной модели Model-View-Controller.

Model-View-Controller (MVC) – это схема разделения приложения на три отдельных компонента таким образом, что изменение любого из компонентов может осуществляться независимо от остальных.

1. Model (Модель) – обработка данных и логика приложения;
2. View (Представление) – представление данных пользователю в любом поддерживаемом формате;
3. Controller (Контроллер) – обработка запросов пользователя и вызов соответствующих ресурсов.

На рисунке 1.3. изображена структура MVC.

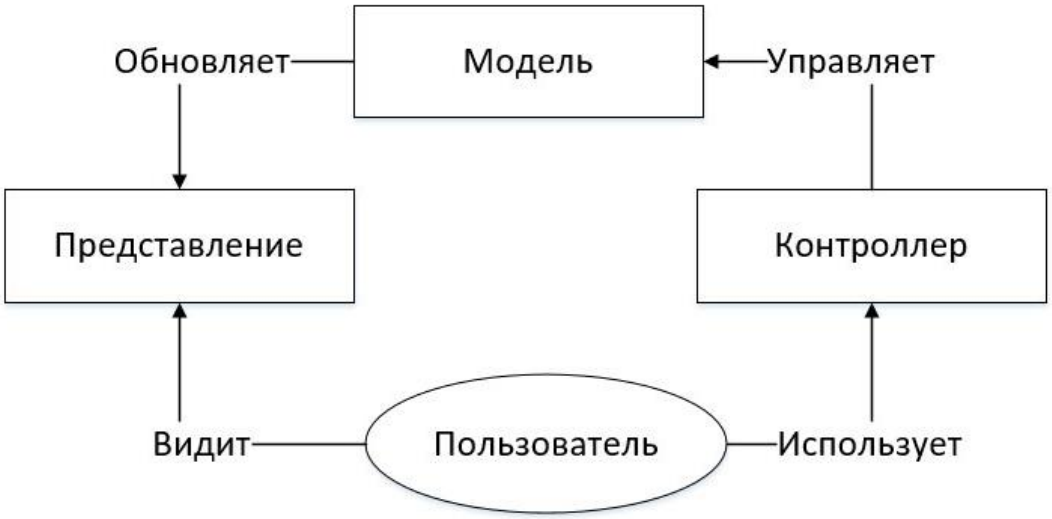


Рис. 2.1. Структура MVC

В архитектуре фреймворка Laravel присутствуют ключевые особенности.

Пакеты позволяют создавать и подключать модули в формате Composer к приложению на Laravel. Многие дополнительные возможности уже доступны в виде этих модулей.

Eloquent ORM – реализация шаблона проектирования ActiveRecord на PHP. Позволяет строго определить отношения между объектами базы данных. Стандартный для Laravel построитель запросов Fluent поддерживается ядром Eloquent.

Логика приложения – часть разрабатываемого приложения, объявленная либо при помощи контроллеров, либо маршрутов (функций-замыканий). Синтаксис объявлений похож на синтаксис, используемый в каркасе Sinatra.

Обратная маршрутизация связывает между собой генерируемые приложением ссылки и маршруты, позволяя изменять последние с автоматическим обновлением связанных ссылок. При создании ссылок с помощью именованных маршрутов Laravel автоматически генерирует конечные URL.

REST-контроллеры – дополнительный слой для разделения логики обработки GET- и POST-запросов HTTP.

Автозагрузка классов – механизм автоматической загрузки классов PHP без необходимости подключать файлы их определений в include. Загрузка по требованию предотвращает загрузку нетребуемых компонентов; загружаются только те из них, которые действительно используются.

Составители представлений – блоки кода, которые выполняются при генерации представления (шаблона).

Инверсия управления позволяет получать экземпляры объектов по принципу обратного управления. Также может использоваться для создания и получения объектов-одиночек.

Миграции – система управления версиями для баз данных. Она позволяет связывать изменения в коде приложения с изменениями, которые требуется внести в структуру БД, что упрощает развёртывание и обновление приложения.

Модульное тестирование (юнит-тесты) играет очень большую роль в Laravel, который сам по себе содержит большое число тестов для предотвращения регрессий (ошибок вследствие обновления кода или исправления других ошибок).

Страничный вывод упрощает генерацию страниц, заменяя различные способы решения этой задачи единым механизмом, встроенным в Laravel.

Laravel является достаточно гибким фреймворком и позволяет решать нестандартные задачи, структурировать веб-сайт в соответствии с существующей логикой и поставленными целями.

Преимуществами данного фреймворка является:

1. Обширный функционал фреймворка;
2. Возможность создания гибкой админпанели;
3. Безопасность данных.

Получить нелегальный доступ к БД крайне сложно. В данном фреймворке реализована надёжная защита от SQL-инъекций, от внедрения в

выдаваемые веб-системой страницы вредоносного кода, а также от межсайтовой подделки запросов.

4. Масштабируемость.

Для разработки интерфейса веб-приложения был выбран фреймворк Bootstrap.

Bootstrap – это бесплатный, кроссплатформенный, написанный на языках JavaScript, CSS и HTML языках программирования фреймворк, предназначенный для front-end разработки веб-приложений.

После того как Bootstrap был выложен в открытый доступ, он стал одним из самых удобных и мощнейших инструментов для разработки пользовательских интерфейсов любой сложности.

К преимуществам данного фреймворка относится:

1. Простота в использовании, что влечёт за собой сокращение времени разработки.
2. Кросс-браузерность и адаптивность. Интерфейс, написанный с использованием данного фреймворка будет одинаково отобраться как в разных браузерах, так и на различных устройствах.

2.1.2 Выбор СУБД

Для реализации проекта была выбрана СУБД PHPMyAdmin. Данная система управления базами данных написана на языке веб-программирования PHP. СУБД даёт возможность через браузер и не только осуществлять администрирование сервера MySQL.

На сегодняшний день PHPMyAdmin широко применяется на практике, в связи с тем, что данная СУБД сочетает в себе надёжность и простоту в использовании, а также отвечает всем требованиям современного программного обеспечения.

2.2 Проектирование базы данных

Проектирование баз данных – это процесс создания схемы БД, а также определение необходимых ограничений соответствий информации её внутренней логике, структуре и всем заданным правилам.

Основными этапами проектирования являются:

- Концептуальное проектирование;
- Логическое проектирование;
- Физическое проектирование.

2.2.1 Концептуальное проектирование базы данных

Концептуальное проектирование включает в себя описание информационных объектов или понятий предметной области и связей между ними, а также описание ограничений целостности.

Результатом данного этапа является семантическая модель предметной области.

Исходя из предметной области можно выделить 9 основных сущностей и атрибуты, описывающие их:

- Тип пользователя (идентификатор типа, тип пользователя);
- Пользователи (идентификатор пользователя, идентификатор учётной записи, пароль, тип пользователя);
- Группа (идентификатор группы, номер группы);
- Студенты (идентификатор студента, идентификатор пользователя, фамилия, имя, отчество, пол, номер студенческого билета, идентификатор группы);
- Преподаватели (идентификатор преподавателя, фамилия, имя, отчество, пол, возраст);

- Кафедры (идентификатор кафедры, номер кафедры, название кафедры);
- Предметы (идентификатор предмета, название предмета);
- Тип пропуска (идентификатор пропуска, тип пропуска);
- Журнал (идентификатор журнала, идентификатор студента, тип пропуска, дата пропуска, кол-во пропущенных академических часов, идентификатор преподаватель-дисциплина-кафедра).

Поскольку каждый преподаватель может совмещать работу на нескольких кафедрах и вести несколько дисциплин, то понадобится 2 промежуточных сущности, чтобы реализовать связь “многие ко многим”.

- Преподаватель-кафедра (идентификатор преподаватель-кафедра, идентификатор преподаватель, идентификатор кафедра);
- Преподаватель-дисциплина-кафедра (идентификатор преподаватель-кафедра, идентификатор предмет).

Для реализации функционала восстановления пароля и создания всех вышеперечисленных сущностей независимо от выбора СУБД понадобится ещё две дополнительных сущности:

- Сброс пароля (почтовый адрес, токен);
- Миграции (идентификатор миграции, миграция, серия-группа).

2.2.2 Логическое проектирование базы данных

На этапе логического проектирования осуществляется создание схемы базы данных на основе конкретной модели данных. Другими словами, осуществляется преобразование концептуальной модели в логическую.

Логический уровень подразумевает точку зрения пользователя, т.е. описание данных, которые используются в конкретном бизнес-процессе.

На рисунке 2.2. представлена логическая модель базы данных.

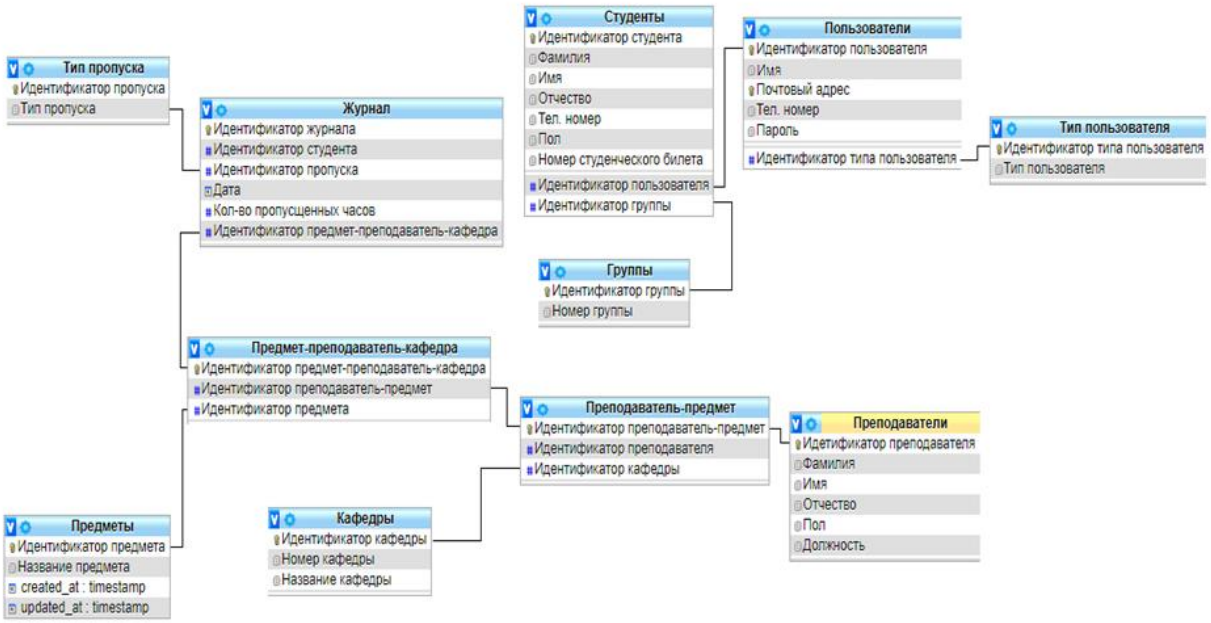


Рис. 2.2. Логическая модель базы данных

2.2.3 Физическое проектирование базы данных

Разработка физической модели БД - процесс подготовки описания реализации базы данных на вторичных запоминающих устройствах. Физическая модель БД отображает таблицы и их названия, поля таблиц, их типы и размеры, и связи между таблицами.

Процесс физического проектирования осуществляется для конкретной СУБД, что означает возможность накладывания ограничений по типу данных, по размерности данных, по наименованию объектов и т.д.

При таком проектирование необходимо указывать название поля, тип данных этого поля и его размерность.

На рисунке 2.3. представлена физическая модель базы данных.

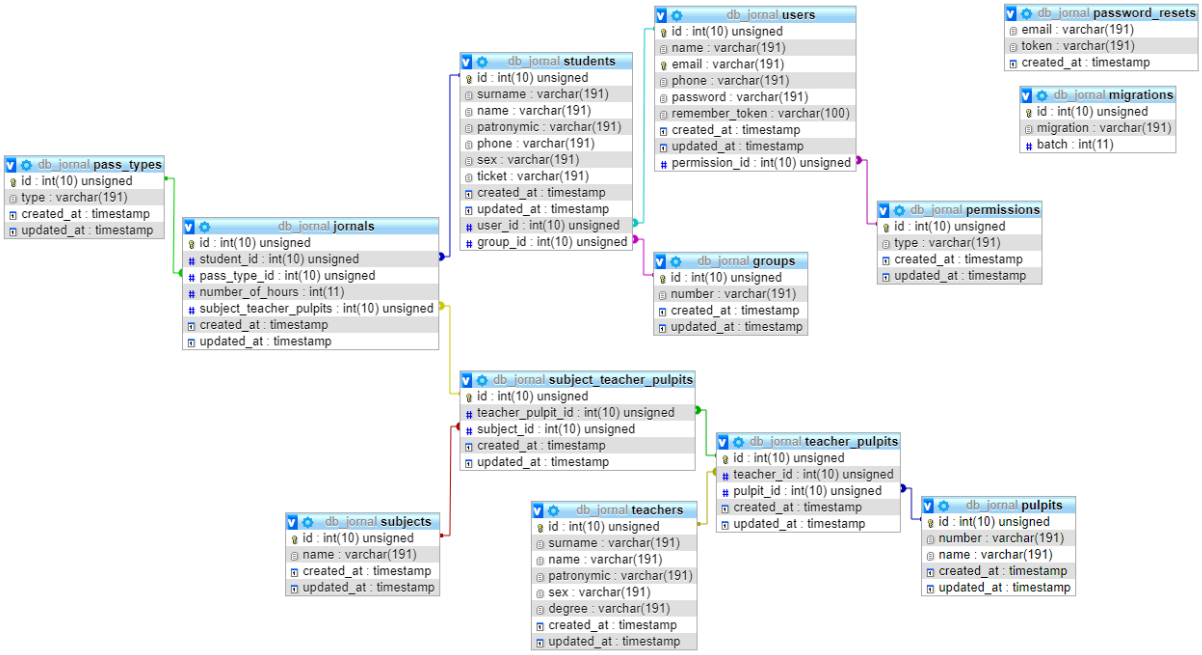


Рис. 2.3. Физическая модель базы данных

Ключевыми полями в таблицах являются поля id. Для первичных ключей и внешних ключей используется тип данных int. Для имен используется тип varchar. Для атрибута отвечающего за дату добавления используется тип данных timestamp.

ГЛАВА 3. РАЗРАБОТКА БАЗЫ ДАННЫХ И ВЕБ-ПРИЛОЖЕНИЯ

После инфологического и даталогического проектирования базы данных, определение функциональных возможностей проекта, можно приступить к этапу разработки информационной системы.

Разработку информационной системы можно разбить на несколько этапов:

- Установка фреймворка Laravel;
- Разработка информационного обеспечения;
- Программирование на стороне сервера;
- Разработка веб-приложения.

3.1 Установка фреймворка Laravel

Перед установкой самого фреймворка понадобиться установить пакетный менеджер для языка разработки PHP. Для фреймворка Laravel таким пакетным менеджером является Composer.

Для его установки в командной строке PowerShell необходимо написать следующее:

Листинг 3.1. Установка Composer

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php -r "if (hash_file('SHA384', 'composer-setup.php') === '544e09ee996cdf60ece3804abc52599c22b1f40f4323403c44d44fdfdd586475ca9813a858088ffbc1f233e9b180f061') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
php composer-setup.php
php -r "unlink('composer-setup.php');"
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
```

Результат и процесс установки можно увидеть на рисунке 3.1.

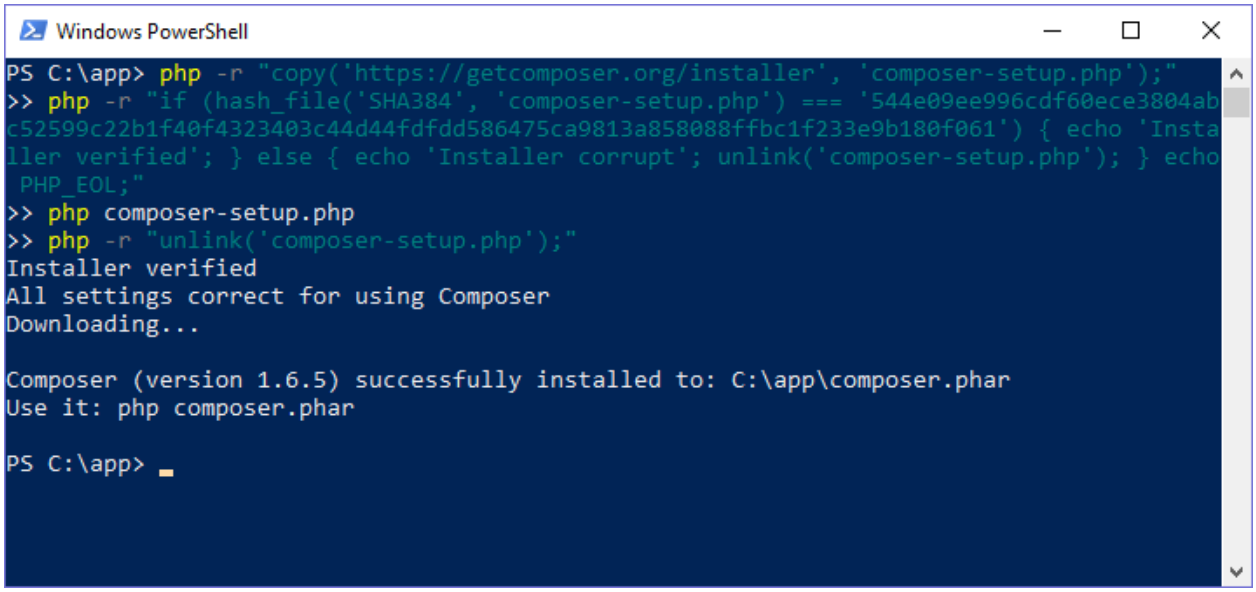


Рис. 3.1. Установка Composer

После установки файлового менеджера становится доступной установка фреймворка. Для этого понадобится в командной строке PowerShell написать следующий код.

Листинг 3.2. Установка Laravel

```
composer global require "laravel/installer"
```

При этом нужно обязательно разместить каталог поставщика в $PATH, чтобы исполняемый файл Laravel был установлен.

Результат и процесс установки файлового менеджера composer можно увидеть на рисунке 3.2.

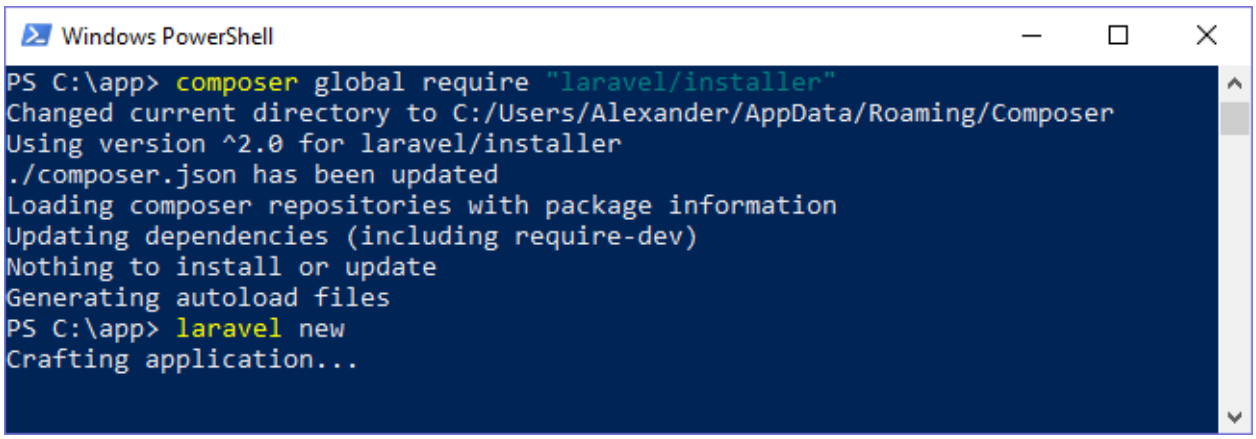


Рис. 3.2. Установка фреймворка Laravel

Далее был создан проект, применением команды “laravel new”.

В результате был получен макет проекта, представленный на рисунке 3.3.

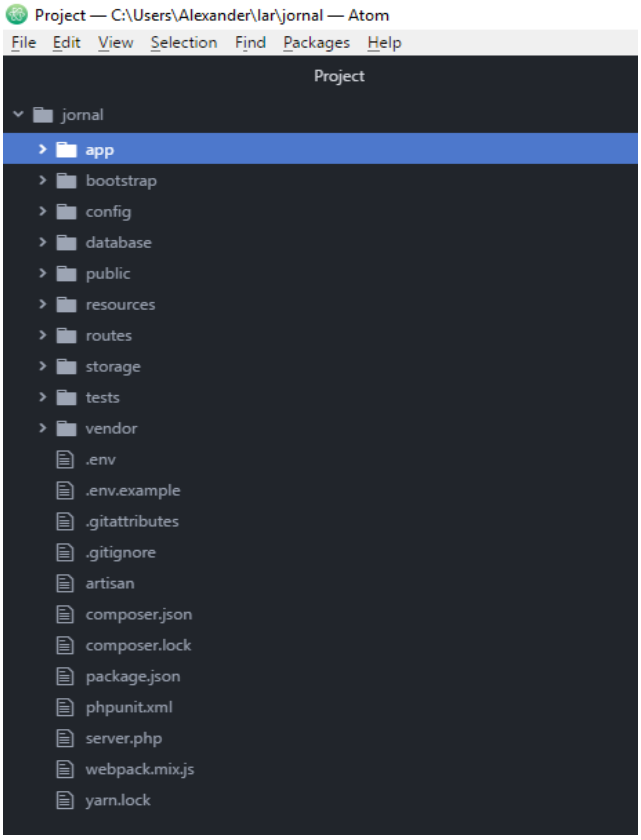


Рис. 3.3. Макет проекта

3.2 Разработка информационного обеспечения

Для хранения информации была создана база данных в СУБД PHPMyAdmin “jornal”, содержащая 13 таблиц. На рисунке 3.4. показано создание базы данных. Также для корректного отображения кириллицы заранее надо выбирать кодировку. Такой кодировкой является utf_8_general_ci.

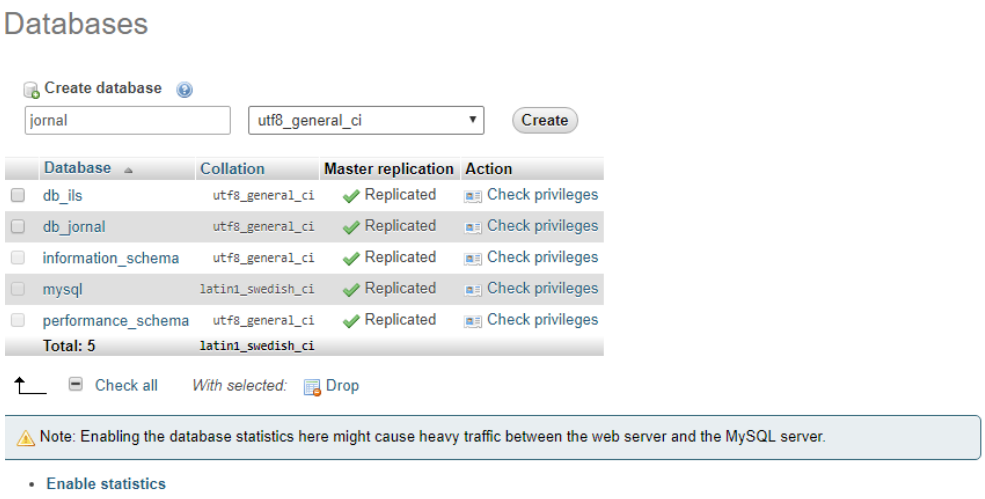


Рис. 3.4. Создание базы данных

Подключение созданной базы данных к проекту происходит в файле “.env”.

Листинг 3.4. Подключение БД

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=db_jornal
DB_USERNAME=root
DB_PASSWORD=root
```

Для создания таблиц, для данной базы данных было создано 13 миграций. Это позволит переносить веб-приложения на другие системы управления базами данных.

Листинг 3.5. Создание миграции create _users_table

```
<?php
use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;
class CreateUsersTable extends Migration {
   public function up() {
      Schema::create('users', function (Blueprint $table) {
         $table->increments('id');
         $table->string('name');
         $table->string('email')->unique();
         $table->string('password');
         $table->rememberToken();
         $table->timestamps();
      });
   }
   public function down() {
      Schema::dropIfExists('users');
   }
}
```

Данная миграция будет создавать таблицу “users”, а в случае пересоздания данной миграции или дополнения полей, то миграция будет удалена и перезаписана заново. Этот функционал выполняет функция down().

Результат миграции create_users_table можно увидеть на рисунке 3.5.

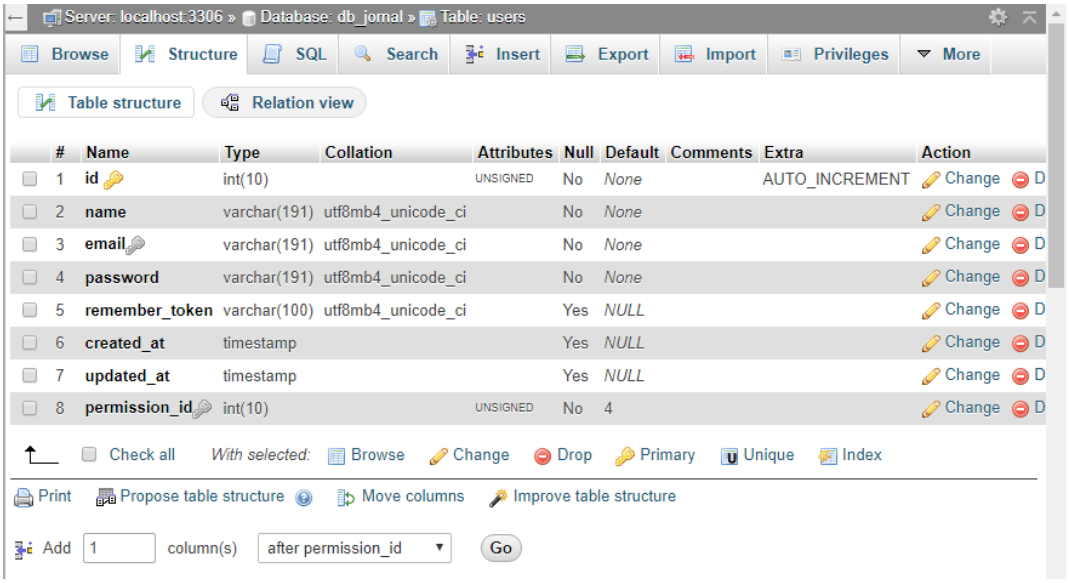


Рис. 3.5. Результат миграции create_users_table

Для создания связей между таблицами были созданы миграции, добавляющие поля в уже имеющие таблицы, после чего данное поле получало дополнительный атрибут зависимости к полю главного ключа другой таблицы.

Листинг 3.6. Добавление вторичного ключа в таблицу users

```
Schema::table('users', function($table)
    {
      $table->integer('permission_id')->unsigned();
      $table->foreign('permission_id')->references('id')->on('permission')-
>onDelete("NO ACTION");
    });
```

Также стоить отметить что для некоторых таблиц вторичный ключ имеет свойство при удалении Cascade. Это дополняет функционал имеющейся программы.

3.3 Программирование на стороне сервера

Особенностью Laravel является то, что данный фреймворк работает не напрямую с имеющейся базой данных, а с объектами.

Для реализации программного продукта были созданы контролеры и модели. Контроллеры наследуют один класс BaseController.

Листинг 3.7. Родительский контроллер

```
<?php
namespace App\Http\Controllers;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Routing\Controller as BaseController;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
class Controller extends BaseController {
use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}
```

Для реализации функционала аутентификации пользователей были созданные такие контролеры как:

- ForgotPasswordController
- LoginController
- RegisterController
- ResetPasswordController

Эти контроллеры осуществляют такие функции как:

- Регистрация в системе
- Вход зарегистрированного пользователя в систему
- Восстановление пароля по адресу электронной почты

Листинг 3.8. Контроллер, осуществляющий регистрацию

```
public function __construct() {
    $this->middleware('guest');
  }
  protected function validator(array $data) {
    return Validator::make($data, [
      'name' => 'required|string|max:255',
      'email' => 'required|string|email|max:255|unique:users',
      'password' => 'required|string|min:6|confirmed',
    ]);
  }
  protected function create(array $data) {
    return User::create([
      'name' => $data['name'],
      'email' => $data['email'],
      'password' => Hash::make($data['password']),
    ]);
  }}
```

Также были разработаны контроллеры, модели и запросы для добавления, удаления и обновления таких таблиц как:

- Предметы;
- Преподаватели;
- Кафедры;
- Группы;

- Пользователи;
- Журналы.

В фреймворке Laravel разработка данного модуля веб-приложения реализуется в консольной строке PowerShell.

Процесс создания контроллеров, моделей и запросов можно увидеть на рисунке 3.6.

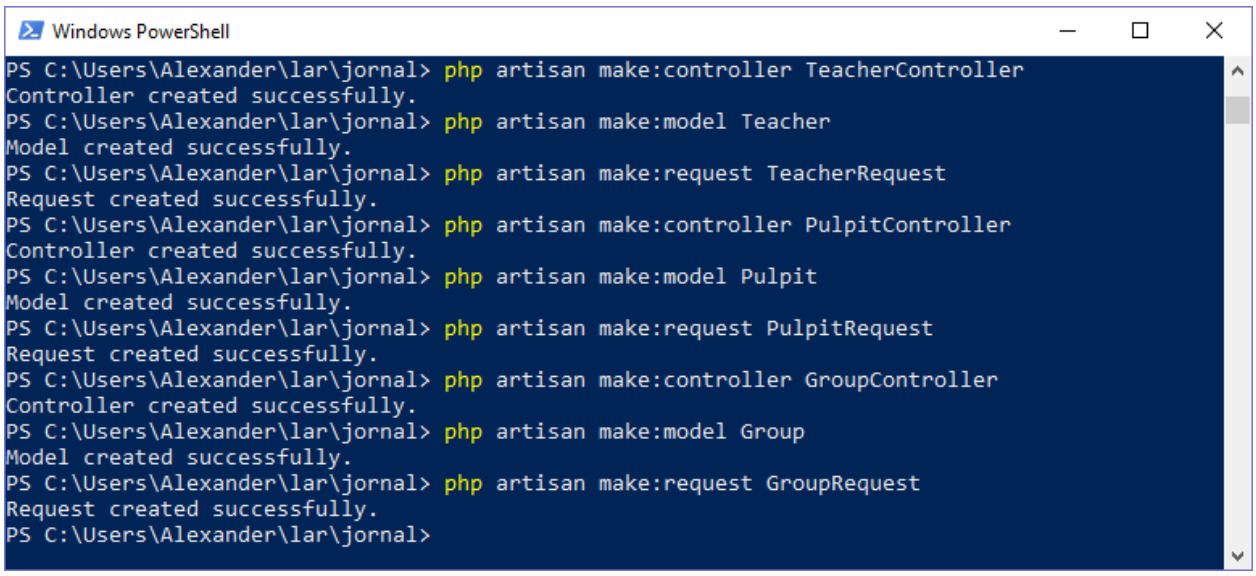


Рис. 3.6. Создание контроллеров, моделей и запросов

По умолчанию модели будут использовать основное соединение с БД, настроенное для веб-приложения. Поэтому свойство $connection указывать не потребуется.

Модели хранят в себе информацию о возвращаемых полях таблицы из базы данных. На листинге 3.9. представлен код модели Teacher.

Листинг 3.9. Код модели Teacher

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;
```

```
class Teacher extends Model {
    protected $fillable = array('surname', 'name' ,'patronymic', 'sex', 'degree');
}
```

При реализации запросов, для каждого из них, была реализована функция rules(), позволяющая устанавливать правила, при которых запрос будет реализован.

Листинг 3.10. Пример функции rules() в запросе SubjectRequest

```
public function rules() {
    return [
        'name'=>'required|min:10'
    ];
}
```

Также в веб-приложении разработан механизм, отображения ошибок. Он выполняет проверки валидации форм и соблюдения правил реализации запросов.

Листинг 3.11. Механизм проверки корректности информации

```
@if(count($errors) > 0)
  <div class="alert alert-dismissible alert-primary">
    <button type="button" data-dismiss="alert" class="close">&times;</button>
    <ul>
      @foreach($errors->all() as $error)
        <li>{{$error}}</li>
      @endforeach
    </ul>
  </div>
@endif
```

3.4 Пользовательский интерфейс

Для реализованной информационной системы был разработан пользовательский веб-интерфейс при помощи фреймворка Bootstrap. При запуске данного веб-приложения пользователь попадает на главную страницу. Интерфейс которой можно увидеть на рисунке 3.7.

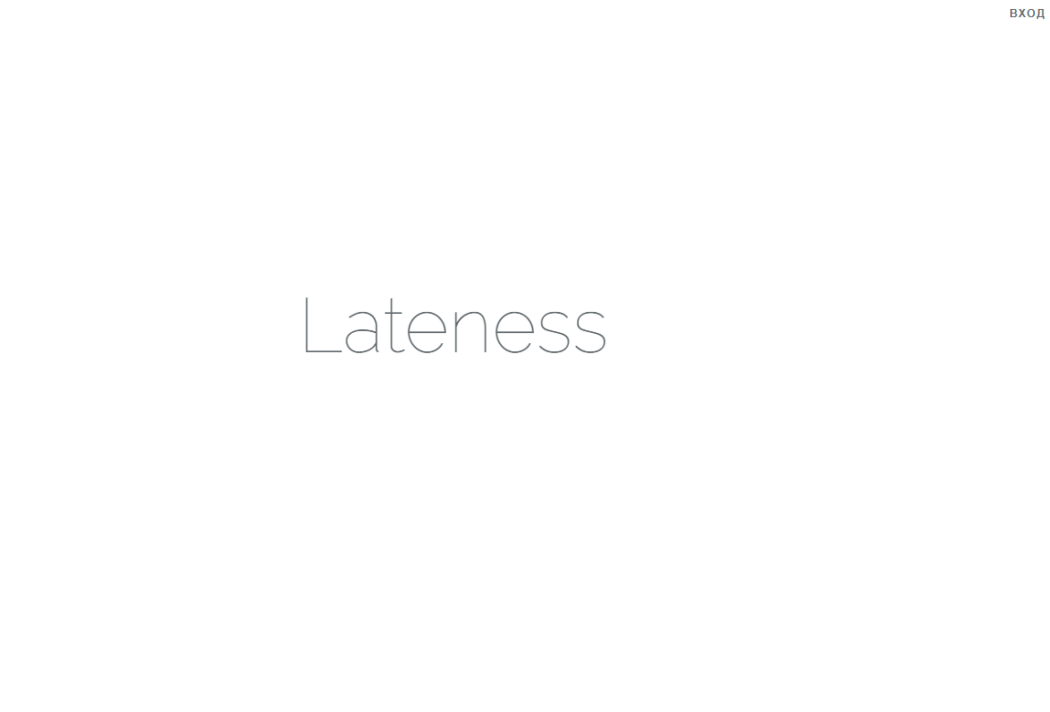


Рис. 3.7. Главная страница веб-приложения

На веб-станице расположены:

- Основная информация веб-приложения (название);
- Модуль аутентификации (кнопка входа в систему).

Далее пользователь может перейти на страницу входа в систему. На данной странице расположены поля для ввода email-адреса и пароля. Также имеется поле для запоминания введённой ранее информации. Также для данной страницы была разработана проверка корректности вводимых данных. На рисунке 3.8. представлен интерфейс входа в систему.

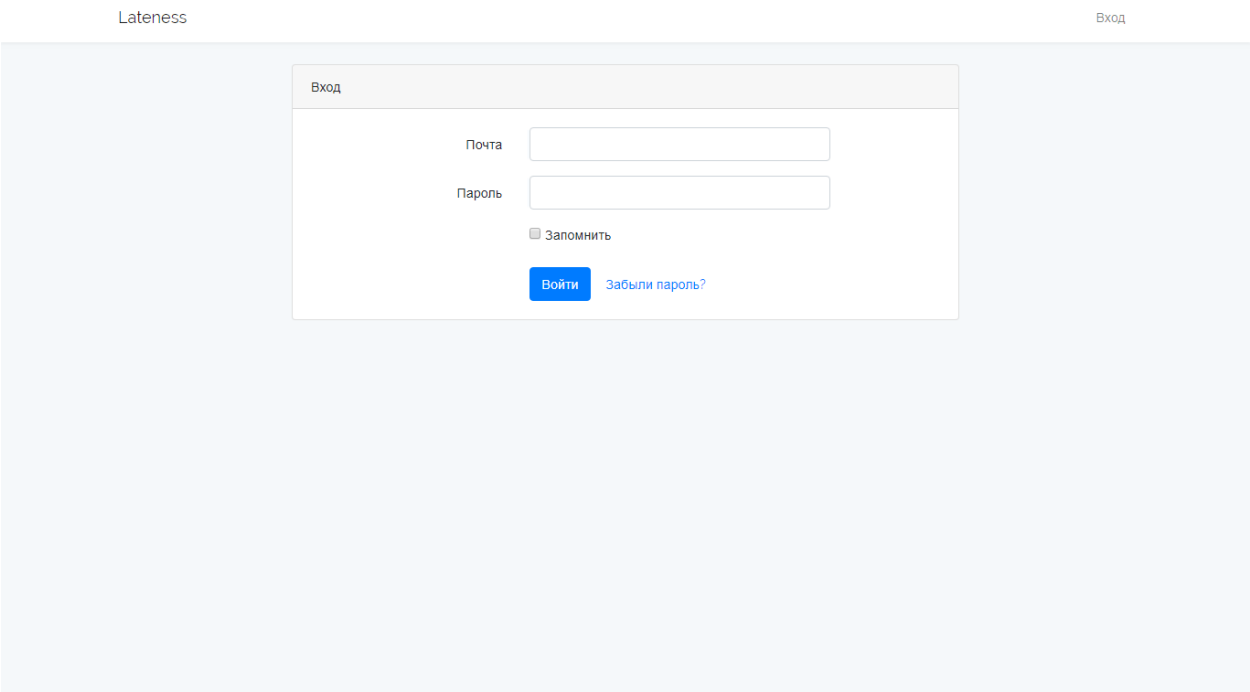


Рис 3.8. Вход в систему

Также в системе предусмотрена возможность восстановления пароля. При нажатии на кнопку “забыл пароль”, пользователю будет предложено ввести свой e-mail адрес. Интерфейс данного функционала представлен на рисунке 3.9.

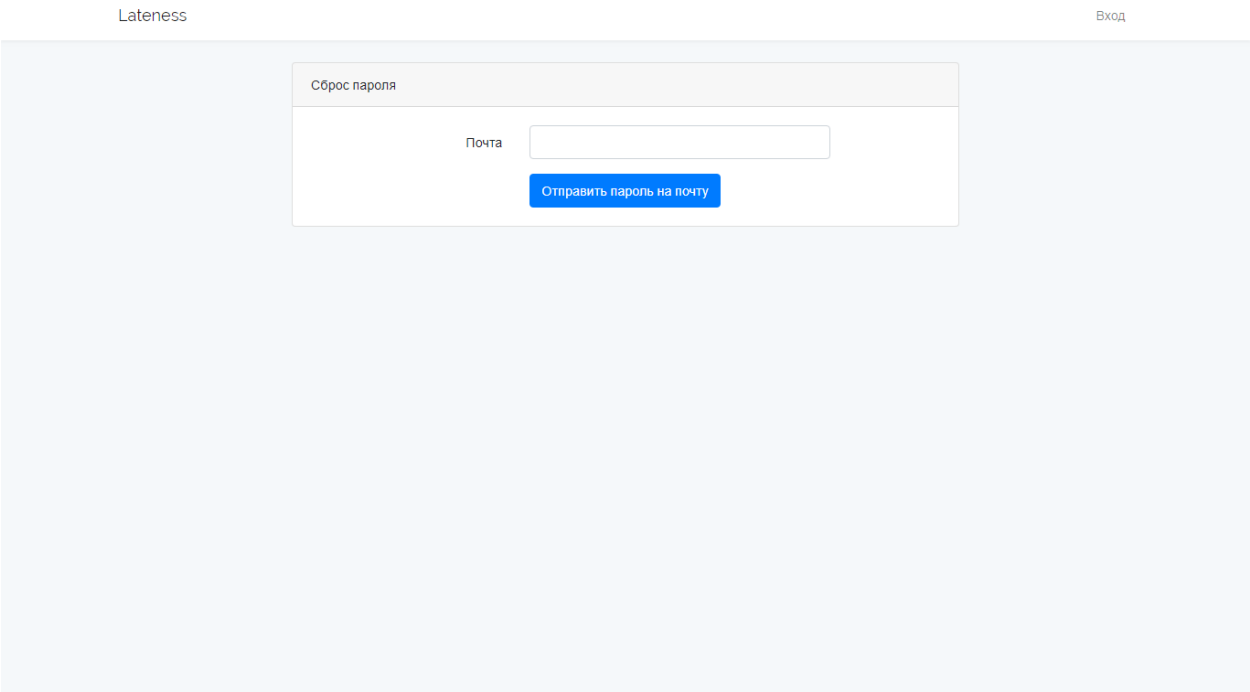


Рис 3.9. Сброс пароля

После чего ему на почту будет отправлено письмо с инструкциями по смене пароля. Пример письма можно увидеть на рисунке 3.10.

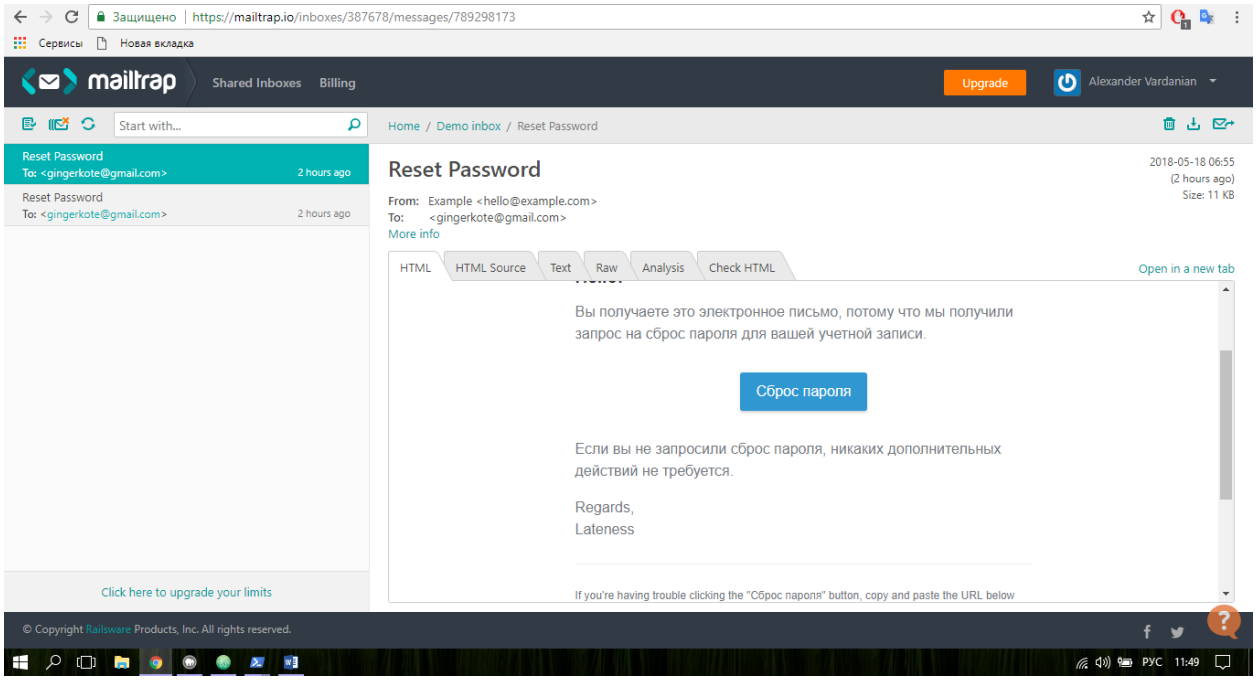


Рис. 3.10. Письмо о смене пароля

В зависимости от прав доступа, после входа в систему пользователь увидит разные интерфейсы, с разными функциональными возможностями. На рисунке 3.11. представлен интерфейс администратора системы.

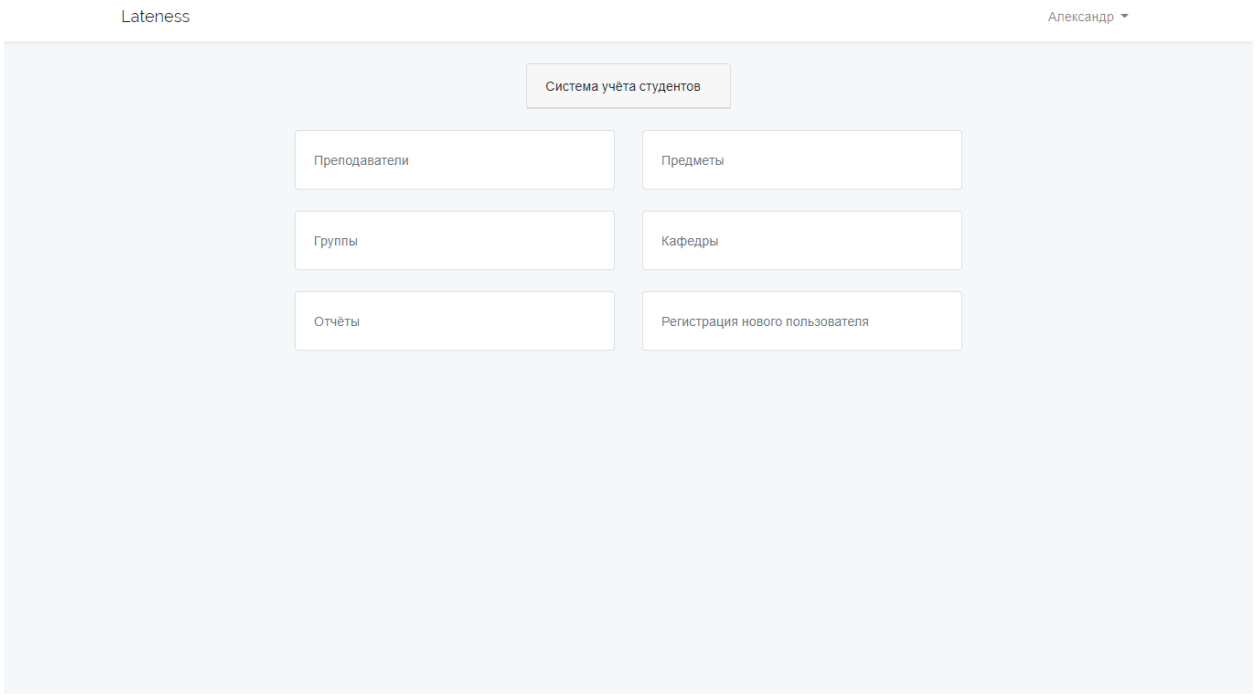


Рис. 3.11. Интерфейс администратора системы

При переходе во вкладку “Кафедры” пользователь увидит список существующих кафедр. Напротив каждой кафедры добавлена кнопка редактирования и удаления информации.

На рисунке 3.12. представлен интерфейс данной веб страницы.

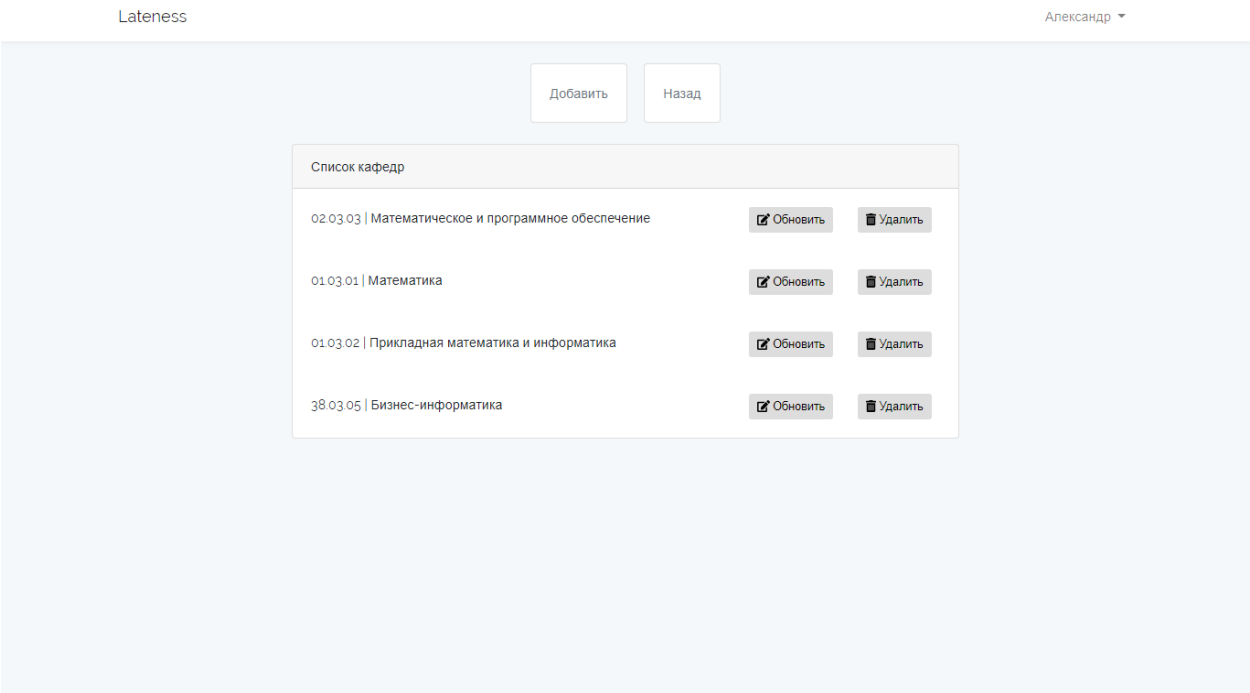


Рис. 3.12. Интерфейс веб страницы “Кафедры”

Также в данной вкладке расположен функционал добавления новых кафедр. На рисунке 3.13. представлен интерфейс добавления кафедры.

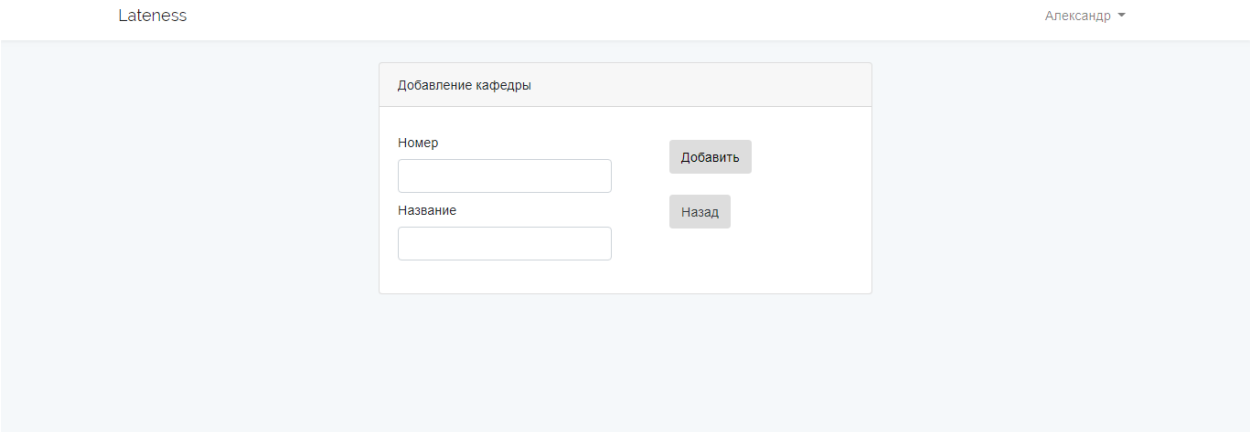


Рис. 3.13. Интерфейс добавления кафедры

При переходе во вкладку “Преподаватели” пользователь увидит список существующих преподавателей. Также в данной вкладке расположен функционал добавления новых преподавателей. Напротив каждого преподавателя добавлена кнопка редактирования и удаления информации.

На рисунке 3.14. представлен интерфейс данной веб страницы.

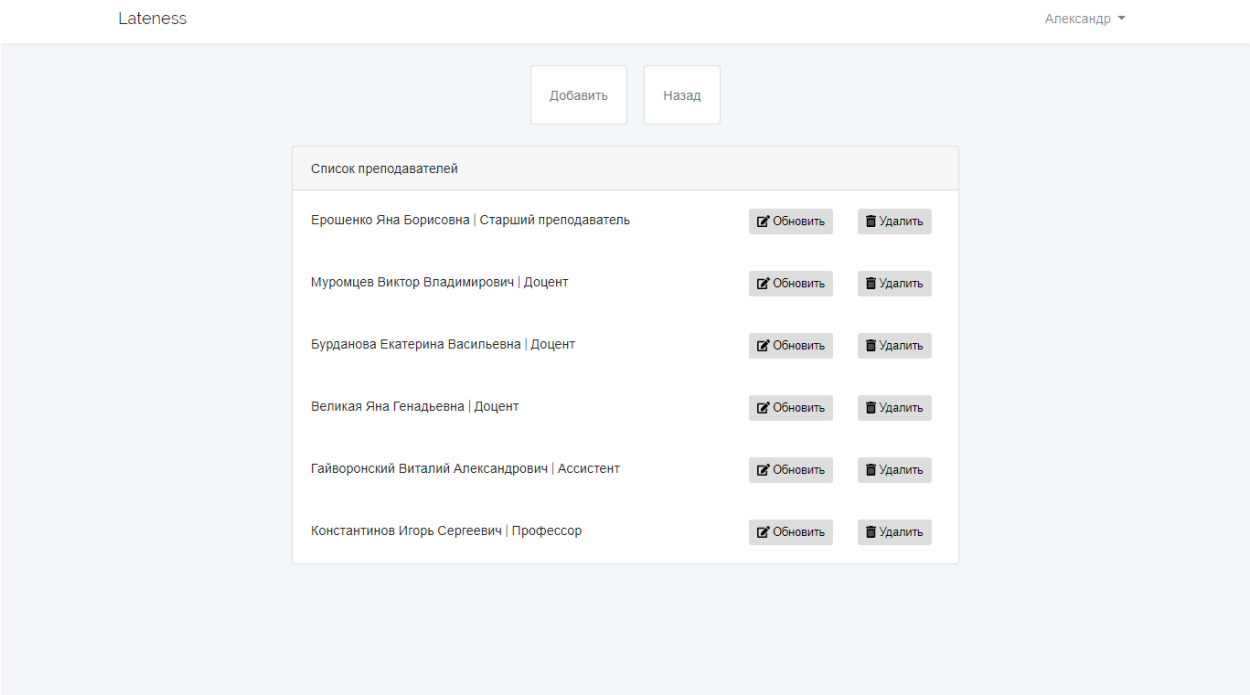


Рис. 3.14. Интерфейс веб страницы “Преподаватели”

Также в данной вкладке расположен функционал добавления новых преподавателей. Поскольку преподаватель не может не принадлежать какой-либо кафедре, вначале пользователю будет предложено выбрать кафедру из уже существующих. Этот функционал представлен на рисунке 3.15.

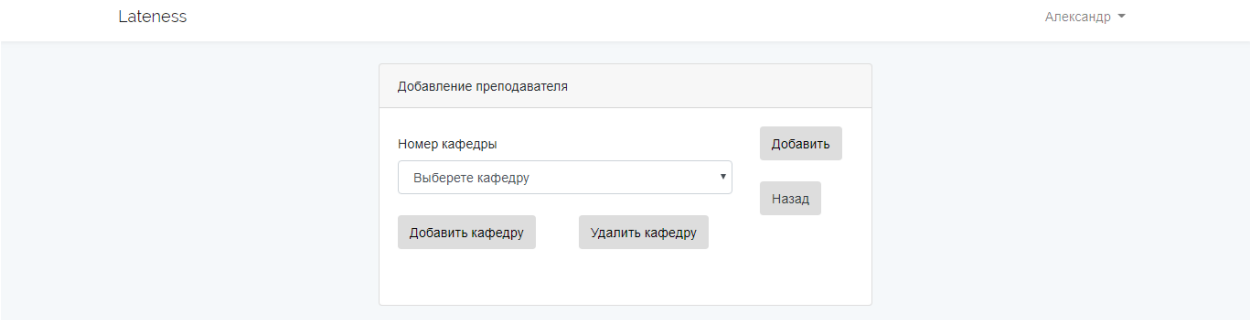


Рис. 3.15. Интерфейс выбора кафедры преподавателя

На рисунке 3.15. можно увидеть кнопку добавления кафедры. Это сделано для того, чтобы можно было назначить преподавателя на несколько кафедр. После выбора кафедры пользователь увидит интерфейс добавления преподавателя.

На рисунке 3.16. представлен интерфейс добавления преподавателей.

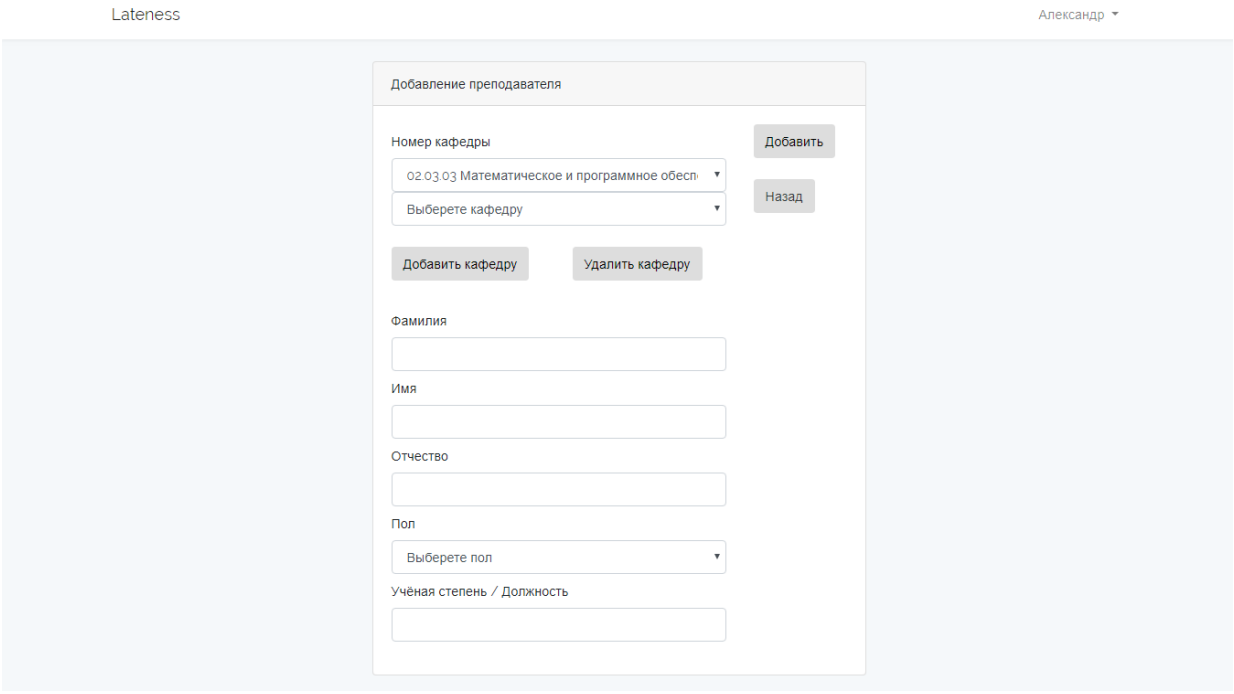


Рис. 3.16. Интерфейс добавления преподавателя

При переходе во вкладку “Предметы” пользователь увидит список существующих предметов. Также в данной вкладке расположен функционал добавления новых предметов и назначения предмета преподавателю.

Напротив каждого предмета добавлена кнопка редактирования и удаления информации.

На рисунке 3.17. представлен интерфейс данной веб страницы.

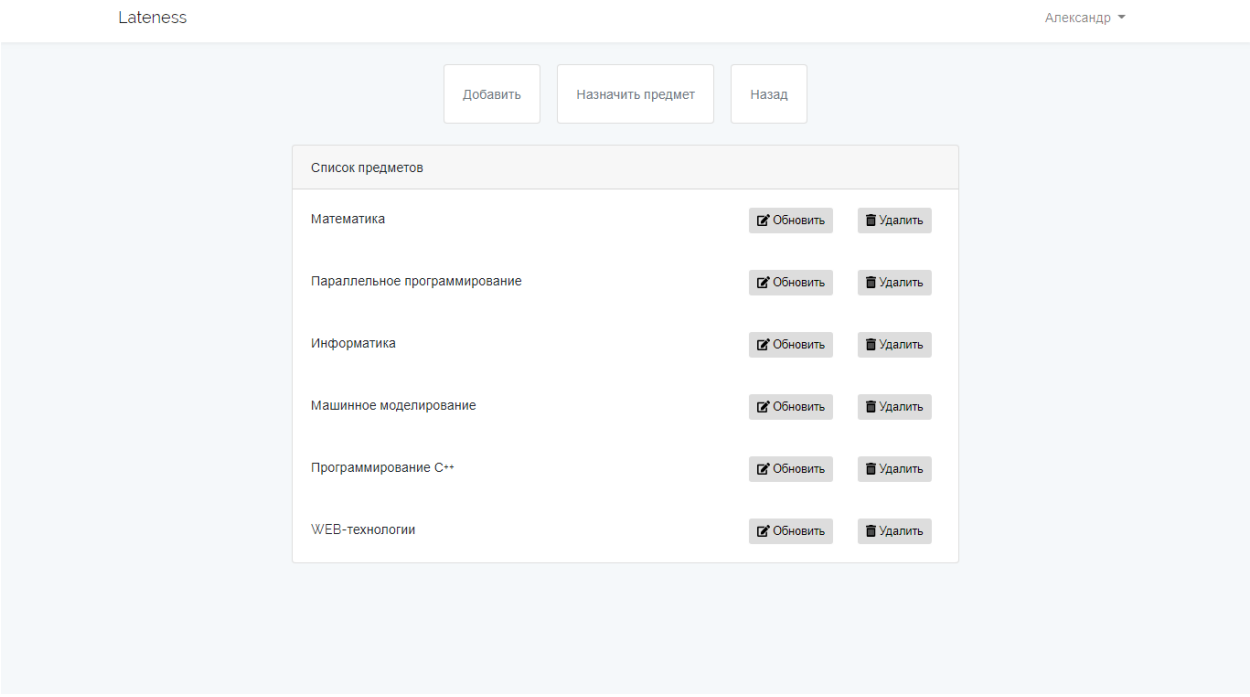


Рис. 3.17. Интерфейс веб страницы “Предметы”

Также в данной вкладке расположен функционал добавления новых предметов.

На рисунке 3.18. представлен интерфейс добавления кафедры.

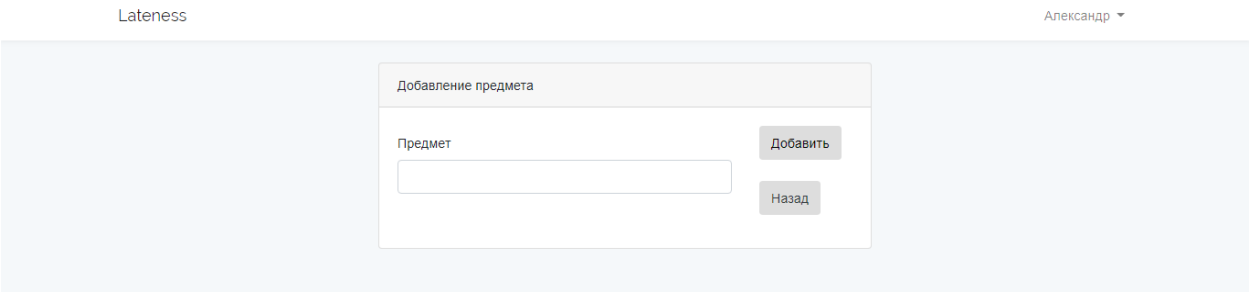


Рис. 3.18. Интерфейс добавления предмета

Поскольку предмет не может не принадлежать какой-либо кафедре и какому-либо преподавателю, вначале пользователю будет предложено выбрать кафедру и предмет из уже существующих. Этот функционал представлен на рисунке 3.19.

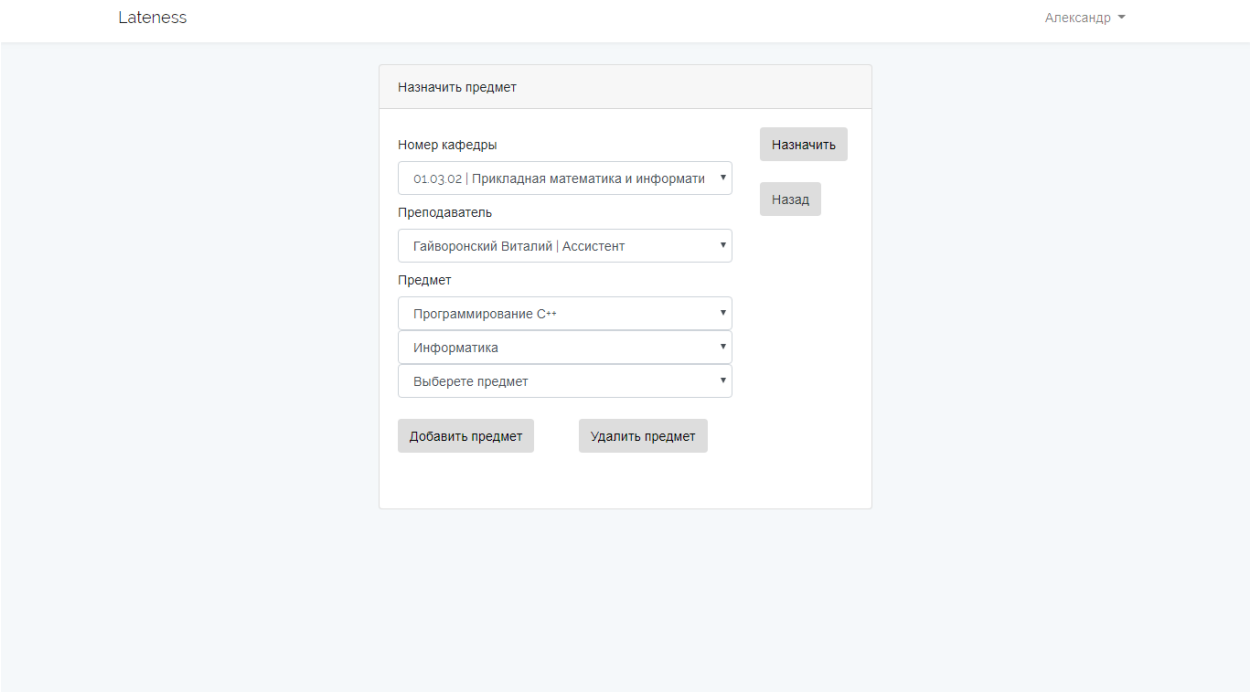


Рис. 3.19. Интерфейс назначения предмета

При переходе во вкладку “Группы” пользователь увидит список существующих групп. Напротив каждой группы добавлена кнопка редактирования и удаления информации.

На рисунке 3.20. представлен интерфейс данной веб страницы.

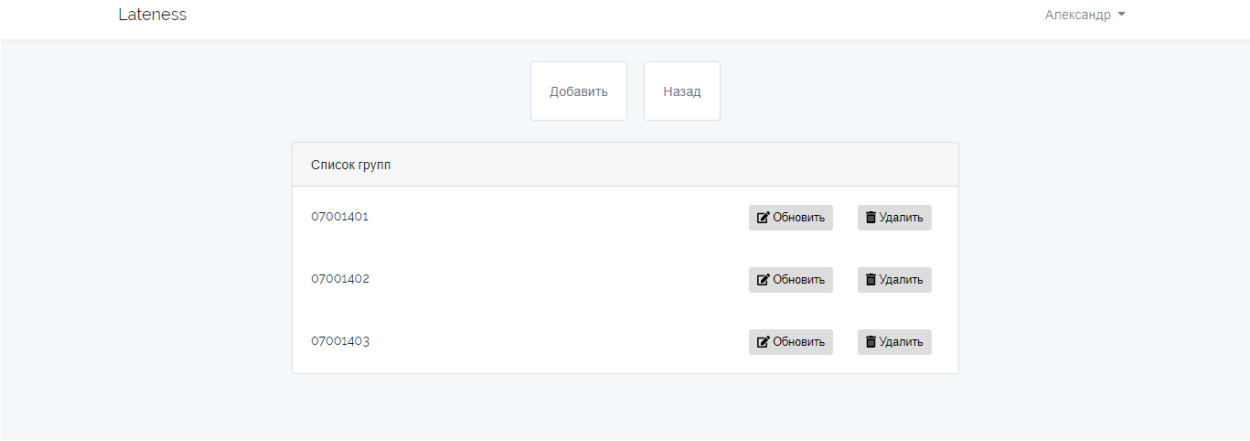


Рис. 3.20. Интерфейс веб страницы “Группы”

Также в данной вкладке расположен функционал добавления новых групп. На рисунке 3.20. представлен интерфейс добавления групп.

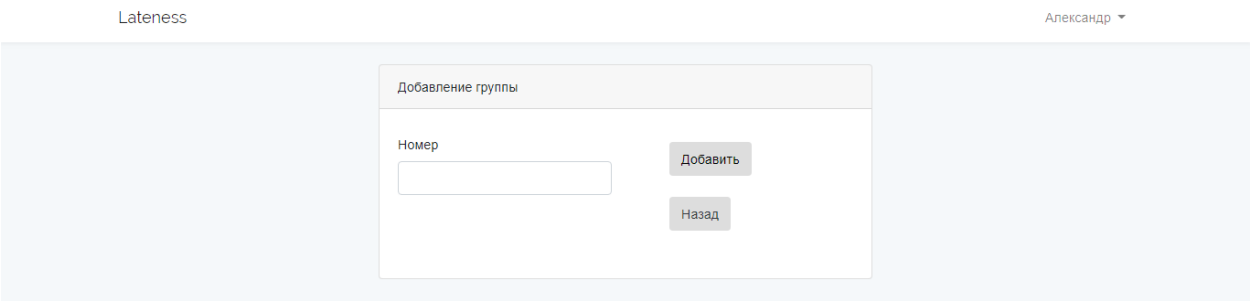


Рис. 3.20. Интерфейс добавления групп

При переходе во вкладку “Регистрация нового пользователя” пользователь увидит список пользователей. Данная вкладка доступна только администратору. Напротив каждой пользователя добавлена кнопка редактирования и удаления информации. На рисунке 3.21. представлен интерфейс данной веб страницы.

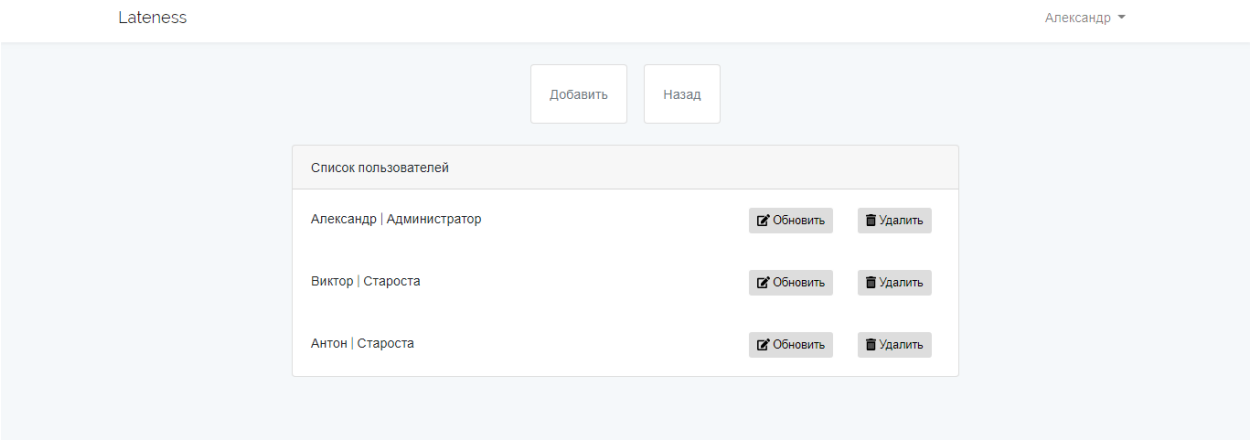


Рис. 3.21. Интерфейс веб страницы “Пользователи”

Также в данной вкладке расположен функционал добавления новых пользователей. Вначале администратору системы будет предложено выбора роли нового пользователя. В зависимости от этого ему будет предложен разный интерфейс добавления.

На рисунке 3.22. представлен интерфейс добавления пользователя “староста”.

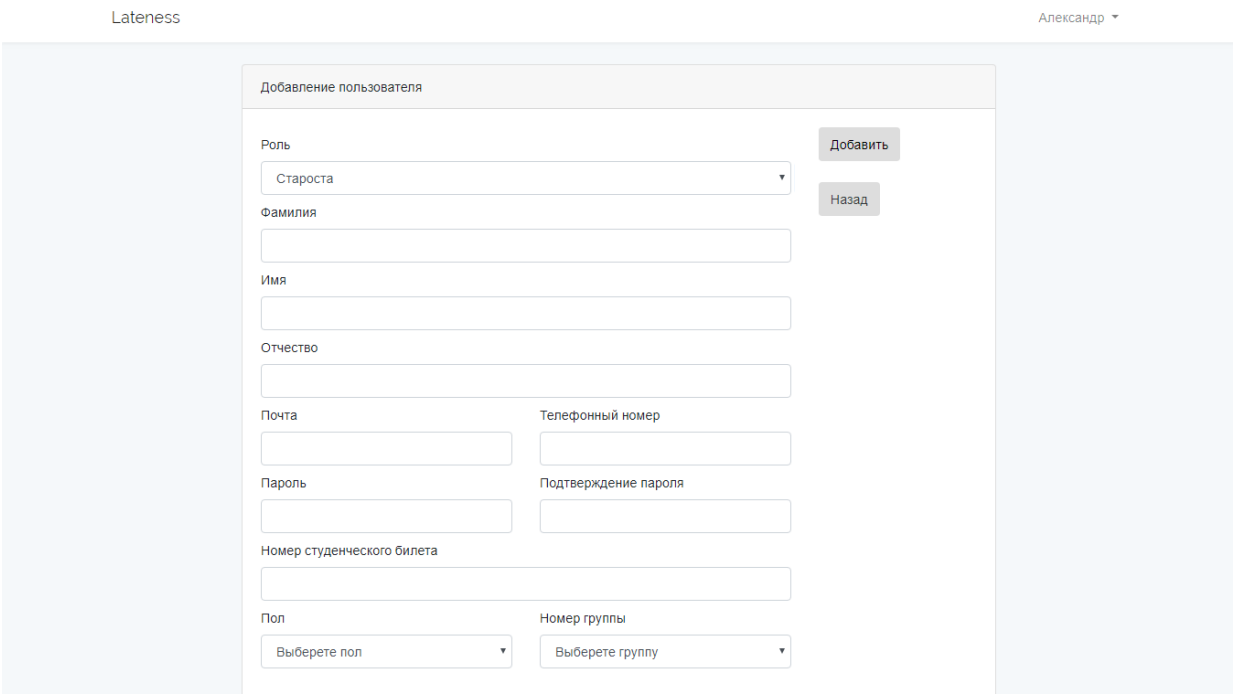


Рис. 3.22. Интерфейс добавления пользователя “староста”

При входе в систему с правами пользователя “староста” пользователь увидит следующий интерфейс, представленный на рисунке 3.23.

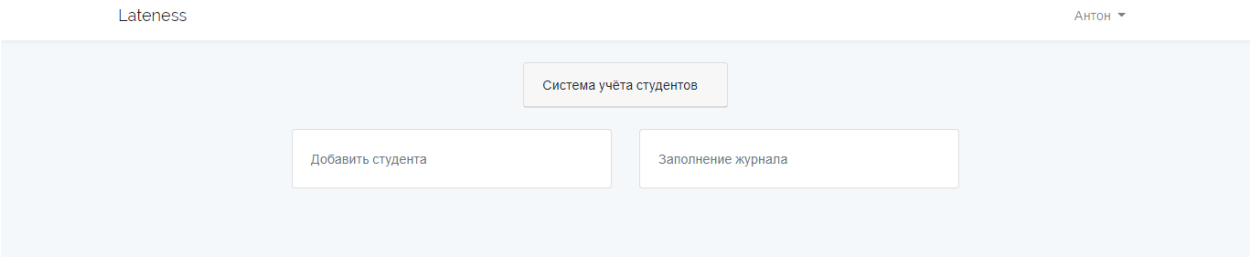


Рис. 3.23. Интерфейс пользователя “староста”

При переходе во вкладку “Добавить студента” пользователь увидит список существующих студентов данной группы. Напротив каждого студента добавлена кнопка редактирования и удаления информации.

На рисунке 3.24. представлен интерфейс данной веб страницы.

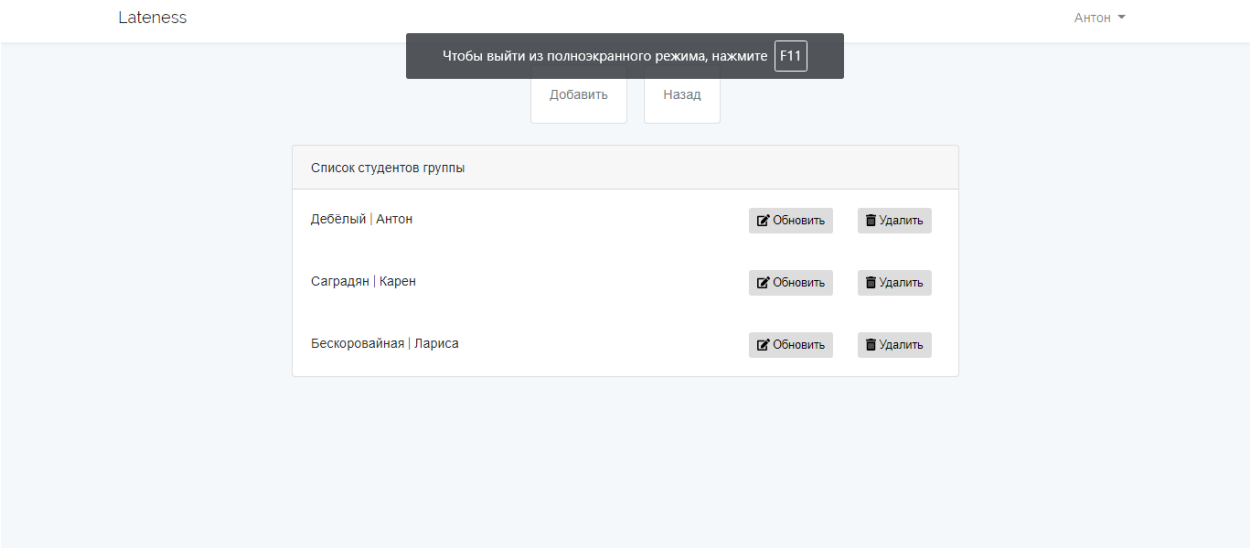


Рис. 3.24. Интерфейс веб страницы “Студенты”

Также в данной вкладке расположен функционал добавления новых студентов. Напротив каждого студента добавлена кнопка редактирования и удаления информации.

На рисунке 3.25. представлен интерфейс данной веб страницы.

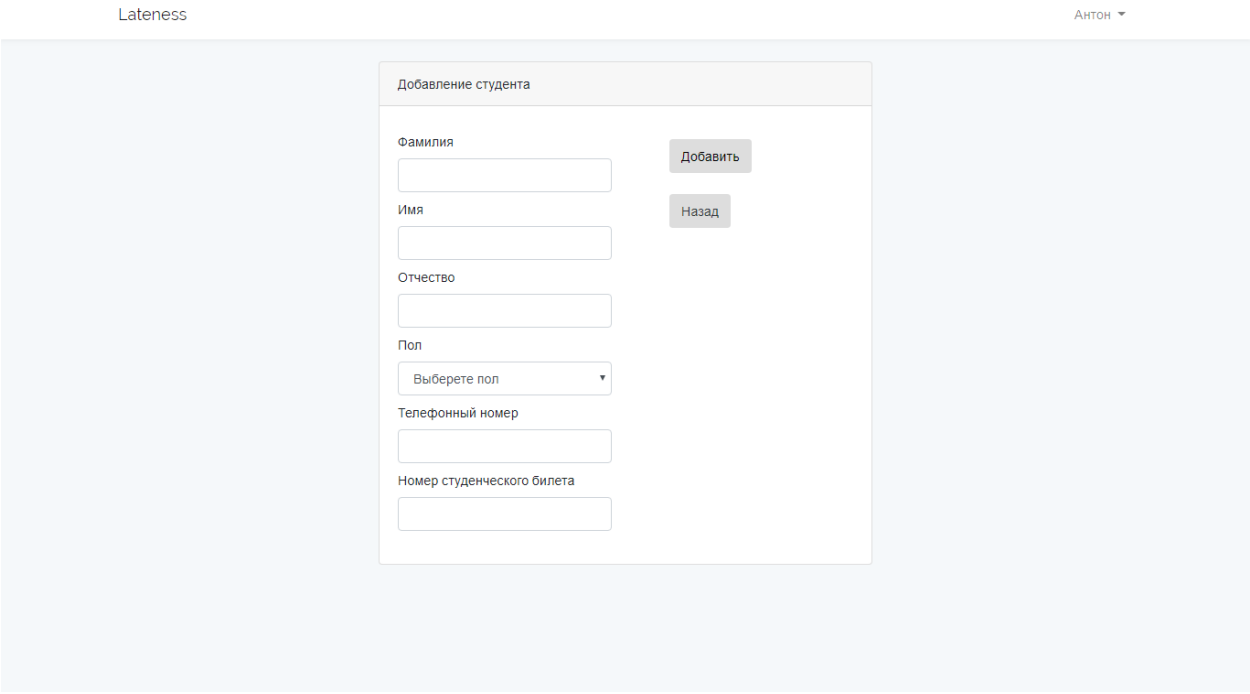
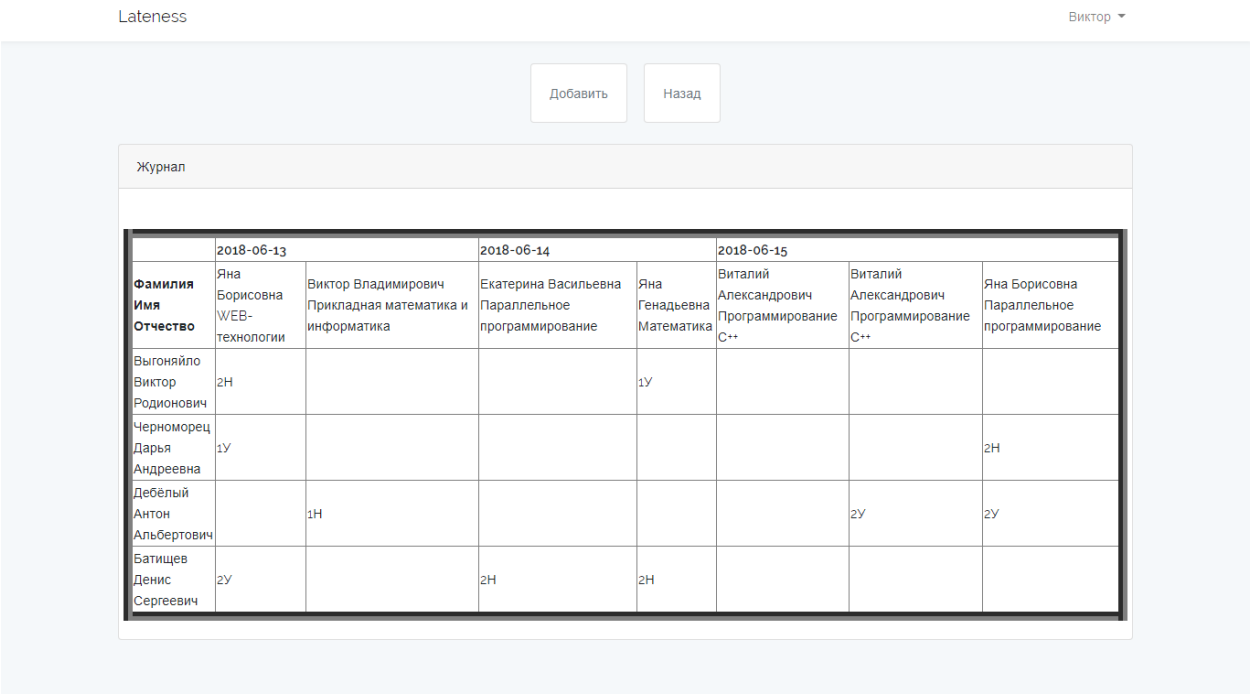


Рис. 3.24. Интерфейс добавление студентов

При переходе во вкладку “Заполнение журнала” староста увидит уже заполненный им ранее журнал на определённые даты.

На рисунке 3.25. представлен интерфейс данной веб страницы.



	2018-06-13		2018-06-14		2018-06-15		
Фамилия Имя Отчество	Яна Борисовна WEB-технологии	Виктор Владимирович Прикладная математика и информатика	Екатерина Васильевна Параллельное программирование	Яна Генадьевна Математика	Виталий Александрович Программирование C++	Виталий Александрович Программирование C++	Яна Борисовна Параллельное программирование
Выгоняйло Виктор Родионович	2Н			1У			
Черноморец Дарья Андреевна	1У						2Н
Дебёлый Антон Альбертович		1Н				2У	2У
Батищев Денис Сергеевич	2У		2Н	2Н			

Рис. 3.25. Интерфейс веб страницы “Заполнение журнала”

Также в данной вкладке расположен функционал добавления новых предметов по данной дате и присвоения студенту статуса “присутствующего” или “отсутствующего” на данной дисциплине.

На рисунке 3.26. представлен интерфейс данной веб страницы.

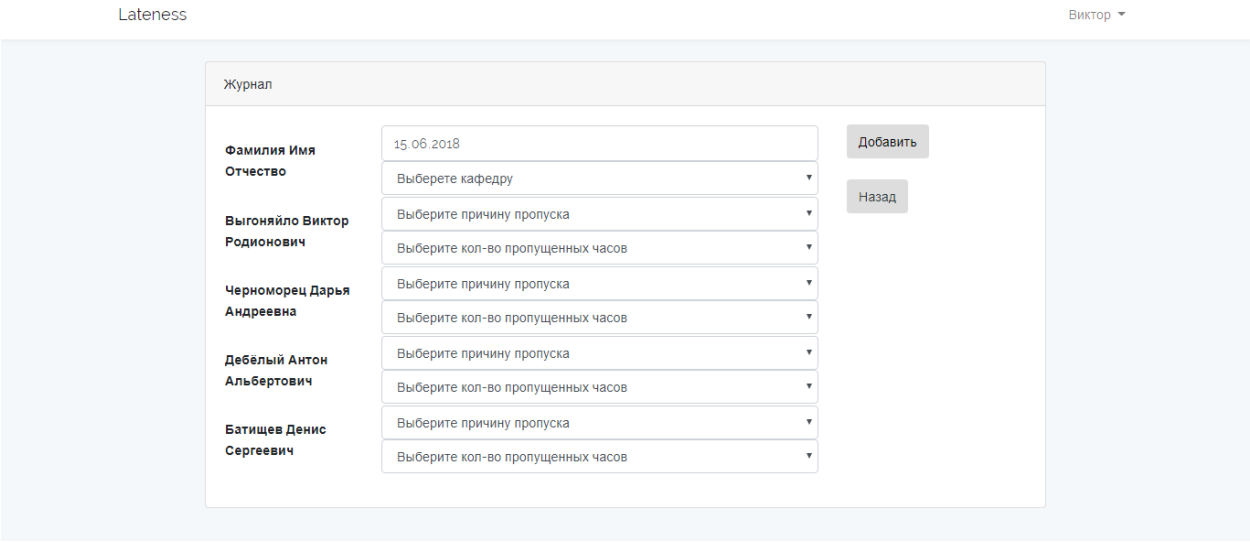


Рис. 3.26. Интерфейс веб страницы “Заполнение журнала”

Разработанный интерфейс обладает адаптивным свойством, что позволяет корректно отображать его на устройствах с разным форматом разрешения.

ГЛАВА 4. ТЕСТИРОВАНИЕ СИСТЕМЫ

В соответствии с установленными требованиями к разрабатываемой информационной системе, будет осуществлена проверка веб-приложения. Для этого необходимо подготовить тестовые данные и провести серию испытаний.

Испытания включают в себя:

- Проверку отображения информации из таблиц базы данных;
- Проверку добавления и удаления информации из базы данных;
- Проверку заполнения журнала.

4.1 Тестовые данные

Для проверки отображения данных в веб-интерфейсе разработанной системы мы подготовили данные заполнив таблицу “Преподаватели” в базе данных, при помощи разработанного модуля добавления. Тестовые данные для проверки отображения данных представлены в таблице 4.1.

Таблица 4.1.

Тестовые данные для таблицы “Преподаватели”

№	Фамилия Имя Отчество	Пол	Должность
1	Ерошенко Яна Борисовна	Женский	Старший преподаватель
2	Муромцев Виктор Владимирович	Мужской	Доцент
3	Бурданова Екатерина Васильевна	Женский	Доцент
4	Великая Яна Генадьевна	Женский	Доцент
5	Гайворонский Виталий Александрович	Мужской	Ассистент
6	Константинов Игорь Сергеевич	Мужской	Профессор

Тестовые данные для проверки заполнения данных журнала представлены в таблице 4.2.

Таблица 4.1.

Тестовые данные для заполнения журнала

№	ФИО студента	Дата пропуска	Тип пропуска
1	Выгоняйло Виктор Родионович	2018-06-13 2018-06-14	2Н 1У
2	Черноморец Дарья Андреевна	2018-06-13 2018-06-15	1У 2Н
3	Дебёлый Антон Альбертович	2018-06-13 2018-06-15	2Н 2У, 2У
4	Батищев Денис Сергеевич	2018-06-13 2018-06-14	2У 2Н, 2Н

Преподаватели и предметы были выбраны случайным образом. Это обусловлено тем, что данная система не привязана ни к какому расписанию занятий.

4.2 Тестовые испытания

Выполняем тестирование согласно установленным требованиям и подготовленным тестовым данным.

Для начала проведём проверку отображения информации из таблицы базы данных “Teachers”, перейдя на панели в пункт “Преподаватели”. Результат отображения данных приведён на рисунке 4.1.

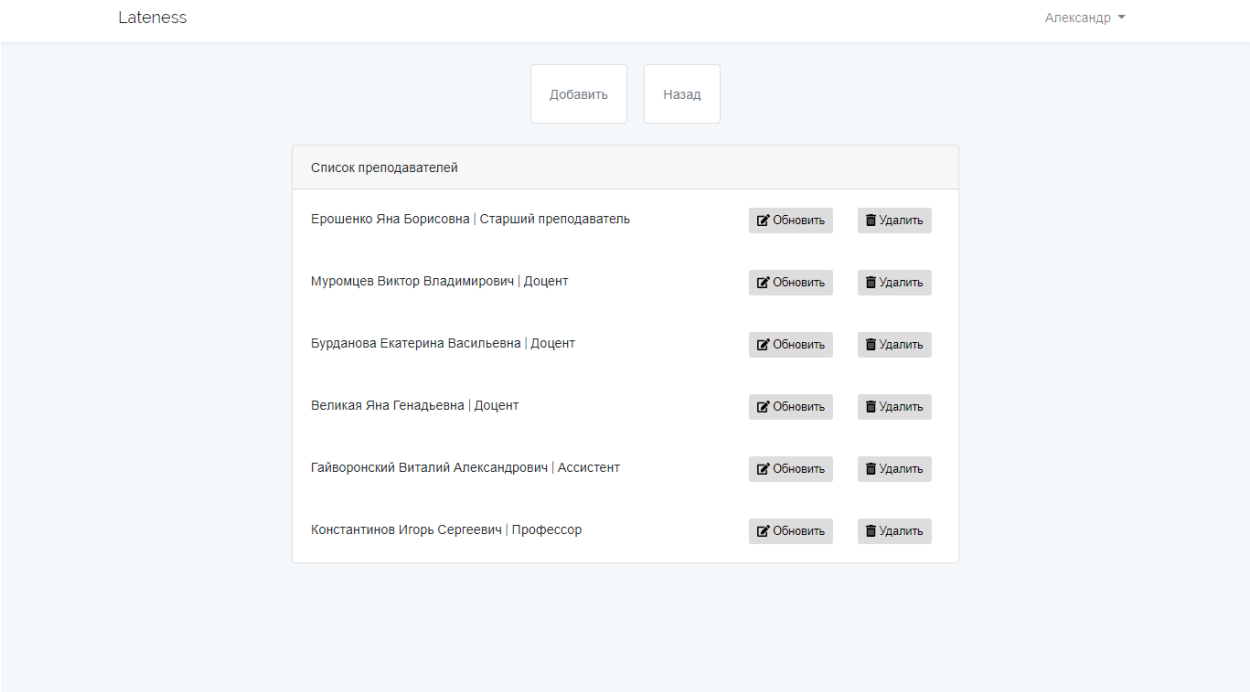


Рисунок 4.1. Вывод информации из таблицы

Процесс удаления данных происходит по средствам выбора, например, последнего преподавателя и нажатия на кнопку удаления.

На рисунке 4.2. показан результат проведения процесса удаления данных.

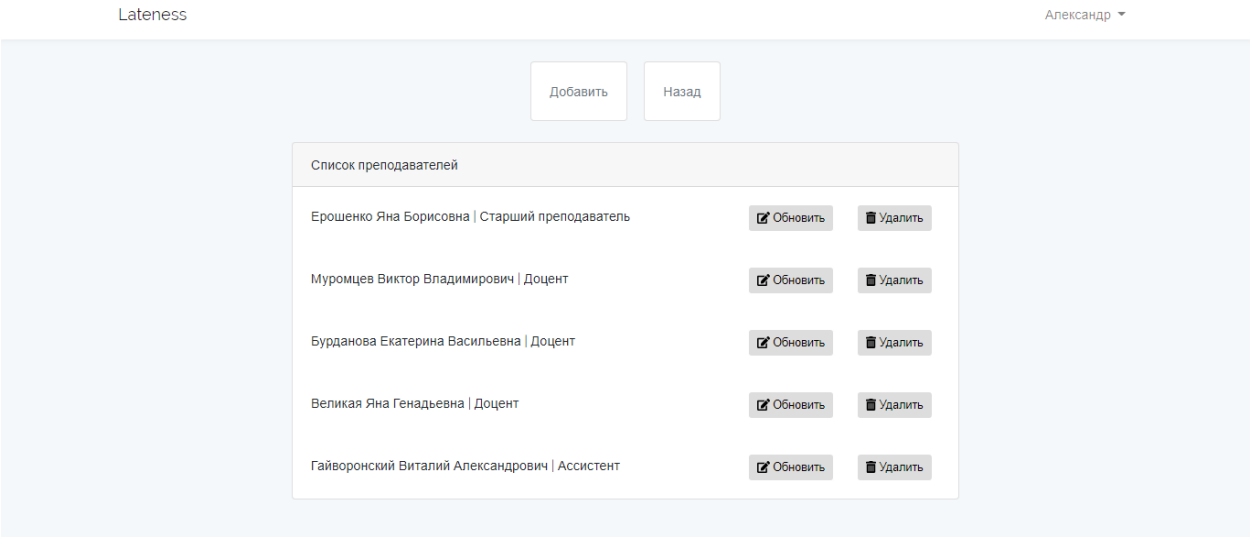


Рисунок 4.2. Удаление информации из таблицы

Процесс добавления данных происходит по средствам ввода информации о преподавателе и нажатия на кнопку добавления.

На рисунке 4.3. показана форма ввода нового преподавателя и на рисунке 4.4. представлен результат процесса добавления данных.

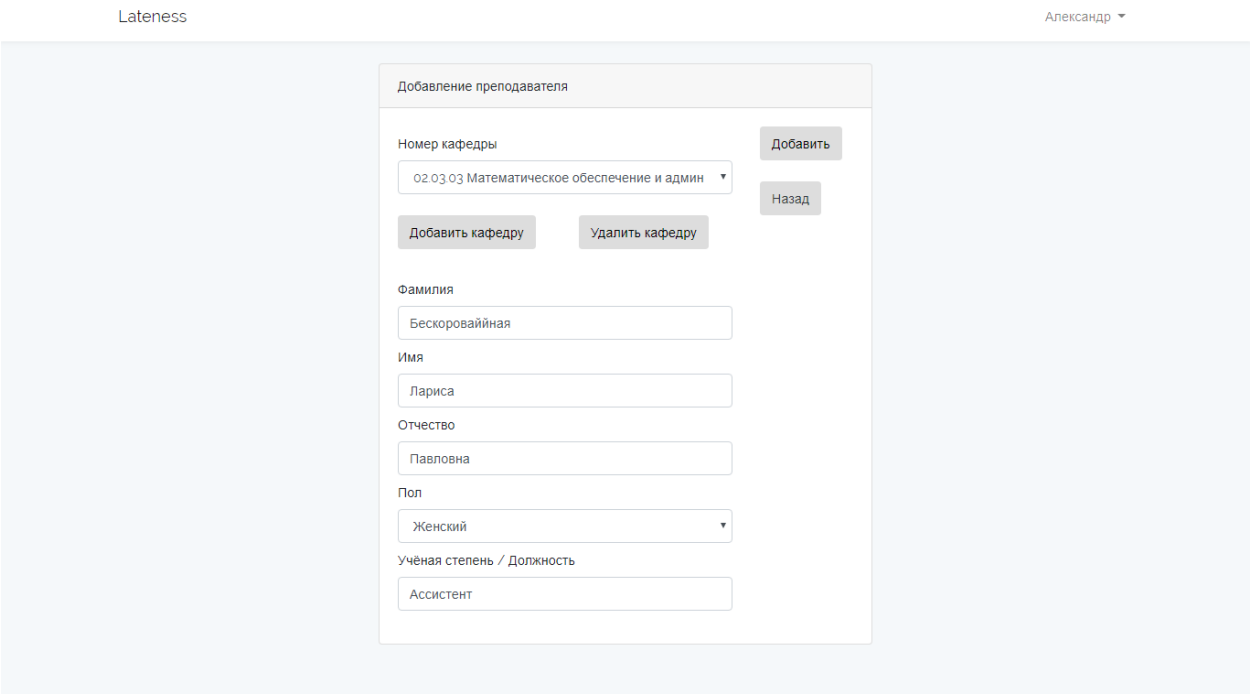


Рисунок 4.3. Форма добавления информации в таблицу

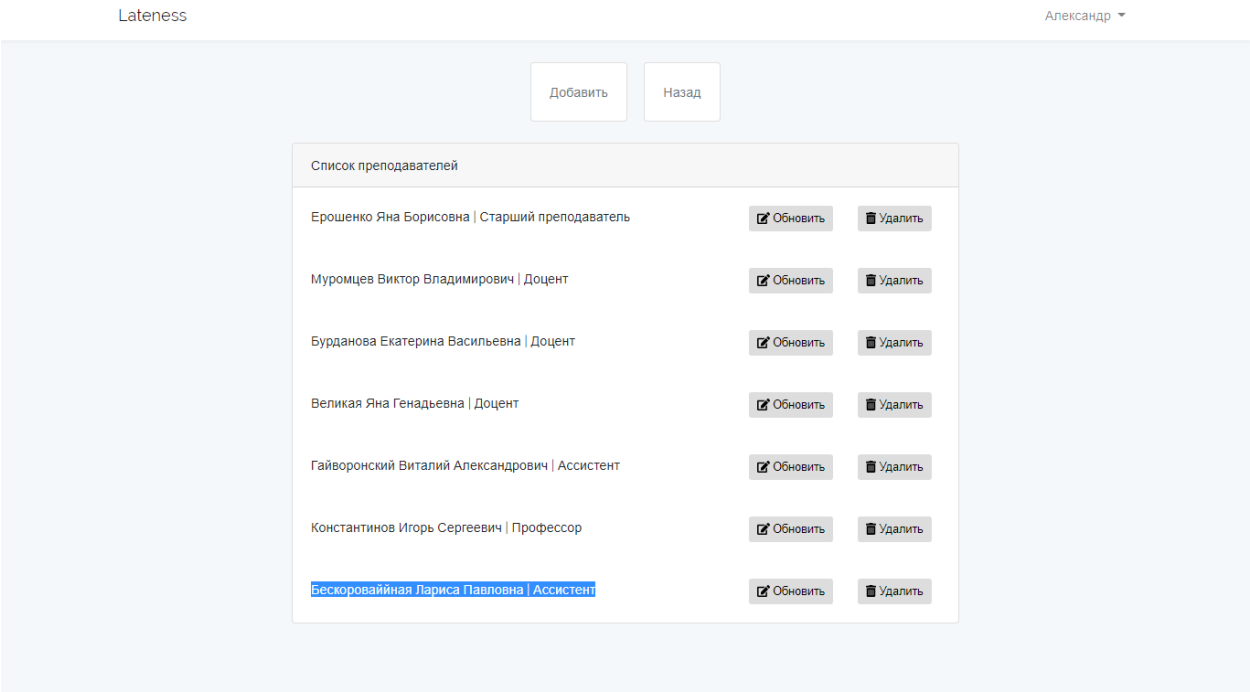
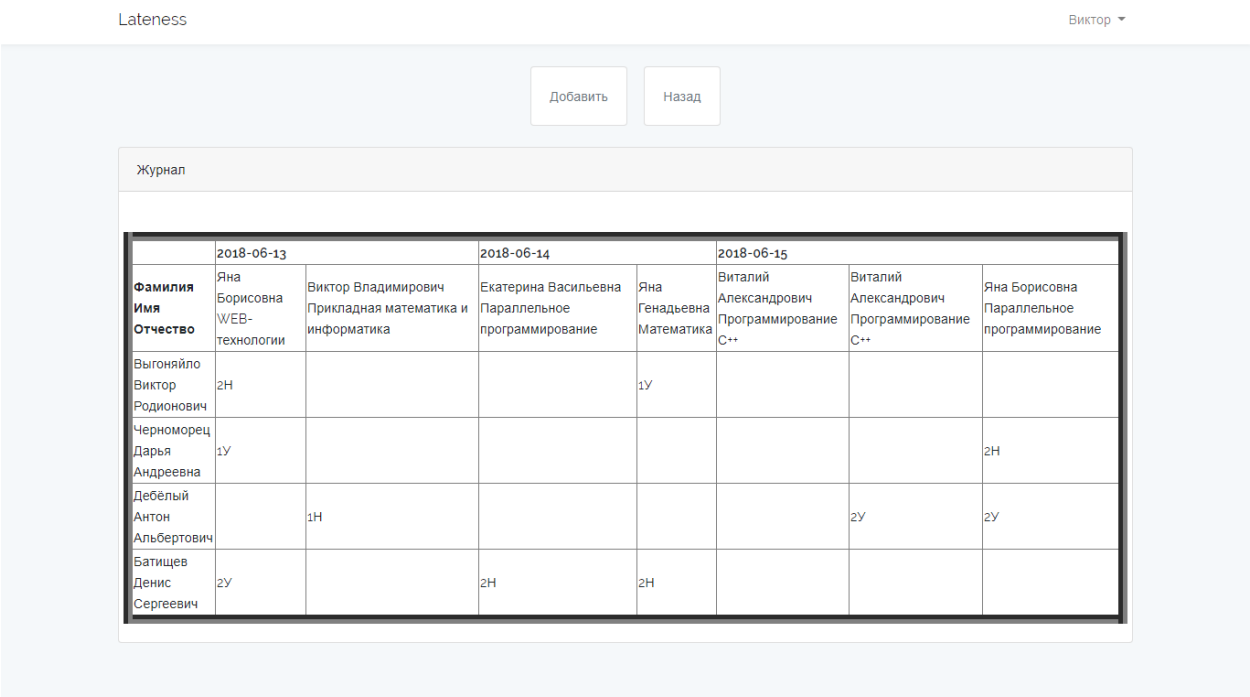


Рисунок 4.4. Добавление информации в таблицу

Далее проведём проверку заполняемости журнала. Для этого согласно тестовым данным добавим отметки об отсутствии студентов согласно заранее подготовленным тестовым данным.

Результат заполнения журнала можно увидеть на рисунке 4.5.



	2018-06-13		2018-06-14		2018-06-15		
Фамилия Имя Отчество	Яна Борисовна WEB-технологии	Виктор Владимирович Прикладная математика и информатика	Екатерина Васильевна Параллельное программирование	Яна Генадьевна Математика	Виталий Александрович Программирование C++	Виталий Александрович Программирование C++	Яна Борисовна Параллельное программирование
Выгоняйло Виктор Родионович	2Н			1У			
Черноморец Дарья Андреевна	1У						2Н
Дебёлый Антон Альбертович		1Н				2У	2У
Батищев Денис Сергеевич	2У		2Н	2Н			

Рисунок 4.5. Результат заполнения журнала

ЗАКЛЮЧЕНИЕ

В ходе работы представлены теоретические сведения об информационных системах, рассмотрены сильные и слабые стороны их построения, а также о разработке веб-приложений с использованием фреймворка Laravel.

В рамках дипломной работы поставленная цель достигнута: разработана информационная система по учёту посещаемости студентов института Инженерных технологий и естественных наук.

Результатом работы стало разработанное веб-приложение, которое способствует решению следующих задач:

- Дополнение, удаление, изменение и хранение информации;
- Создание журналов посещаемости студентов;
- Получение отчётности о посещаемости по заданным критериям.

Разработанное приложение протестировано на предмет безопасности хранимой информации. Данное приложение является готовым программным продуктом, пригодным для использования в качестве инструмента учёта посещаемости студентов.

В качестве возможных путей дальнейшего развития приложения можно предложить интеграцию дополнительных модулей, например, для учёта успеваемости студентов.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Федеральный закон от 29 декабря 2012 г. № 273-ФЗ "Об образовании в Российской Федерации" [Электронный ресурс]. – URL: http://минобрнауки.рф.
2. Рейтинги вузов на 2017 год [Электронный ресурс]. – URL: https://ru.wikipedia.org.
3. Об институте [Электронный ресурс]. – URL: http://iten.bsu.edu.ru/iten/info/about/.
4. ГОСТ 33707-2016 (ISO/IEC 2382:2015) Информационные технологии. Словарь.
5. phpMyAdmin [Электронный ресурс]: режим доступа: https://ru.wikipedia.org/wiki/phpMyAdmin
6. Дейт К. Дж. Введение в системы баз данных – «Вильямс», 2001. – 485 с.
7. Хелен Борри. Firebird. Руководство разработчика баз данных – СПб: БХВ-Петербург, 2007. – 1104 с.
8. Маклаков С. BPWin ErWin – Case-средства разработки информационных систем – Диалог-МИФИ, 2001.
9. Мартин Грубер. SQL – М.: Лори, 2007. – 672 с.
10. Построение инфологической модели [Электронный ресурс]. – Режим доступа: http://citforum.ru/database/dbguide/5-2.shtml , свободный.

ПРИЛОЖЕНИЕ

Листинг home.blade.php

```
@extends('layouts.app')

@section('content')
<div class="container">
   <div class="row justify-content-center">
      <div class="col-md-8">

        <div class="row justify-content-center">
          <div class="col-md-4"> </div>
          <div class="col-md-4">
            <div class="card">
              <div class="card-header">Система учёта студентов</div>
                @if (session('status'))
                  <div class="alert alert-success">
                    {{ session('status') }}
                  </div>
                @endif
            </div>
          </div>
          <div class="col-md-4  "> </div>
        </div>

        @if(Auth::user()->permission_id == 1 || Auth::user()->permission_id == 2)
        <div class="row justify-content-center">
          <div class="col-md-6">
            <!-- Преподаватели -->
```

```
            <br />
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/teacher') }}"> Преподаватели
</a>
              </div>
            </div>
          </div>

          <div class="col-md-6">
            <!-- Предметы -->
            <br />
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/subject') }}"> Предметы </a>
              </div>
            </div>
          </div>
        </div>

        <div class="row justify-content-center">
          <div class="col-md-6">
            <!-- Группы -->
            <br />
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/group') }}"> Группы </a>
              </div>
            </div>
          </div>
```

```
  <div class="col-md-6">
    <!-- Кафедры -->
    <br />
    <div class="card">
      <div class="card-body">
        <a class="text-muted" href="{{ url('/pulpit') }}"> Кафедры </a>
      </div>
    </div>
  </div>
</div>

<div class="row justify-content-center">

  <div class="col-md-6">
    <!-- Формирование отчёта -->
    <br />
    <div class="card">
      <div class="card-body">
        <a class="text-muted" href="#"> Отчёты </a>
      </div>
    </div>
  </div>
  <!-- Регистрация пользователя системы -->
  @if(Auth::user()->permission_id == 1)
  <div class="col-md-6">
    <br />
    <div class="card">
      <div class="card-body">
        <a class="text-muted" href="{{ url('/newuser') }}"> Регистрация
нового пользователя </a>
```

```
            </div>
          </div>
        </div>
        @endif

      </div>
      @endif

      @if(Auth::user()->permission_id == 3)
      <div class="row justify-content-center">
        <div class="col-md-6">
          <!-- Добавление студентов -->
          <br />
          <div class="card">
            <div class="card-body">
              <a class="text-muted" href="{{ url('/student') }}"> Добавить студента </a>
            </div>
          </div>
        </div>
        <div class="col-md-6">
          <!-- Заполнение журнала -->
          <br />
          <div class="card">
            <div class="card-body">
              <a class="text-muted" href=" {{ url('/jornal') }} "> Заполнение журнала </a>
            </div>
          </div>
        </div>
```

```
            </div>
            @endif

        </div>
    </div>
</div>
@endsection
```

Листинг welcome.blade.php

```
<!doctype html>
<html lang="{{ app()->getLocale() }}">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1">

        <title>Laravel</title>

        <!-- Fonts -->
        <link href="https://fonts.googleapis.com/css?family=Raleway:100,600"
rel="stylesheet" type="text/css">

        <!-- Styles -->
        <style>
            html, body {
                background-color: #fff;
                color: #636b6f;
                font-family: 'Raleway', sans-serif;
                font-weight: 100;
```

```
    height: 100vh;
    margin: 0;
}

.full-height {
    height: 100vh;
}

.flex-center {
    align-items: center;
    display: flex;
    justify-content: center;
}

.position-ref {
    position: relative;
}

.top-right {
    position: absolute;
    right: 10px;
    top: 18px;
}

.content {
    text-align: center;
}

.title {
    font-size: 84px;
```

```
            }

            .links > a {
                color: #636b6f;
                padding: 0 25px;
                font-size: 12px;
                font-weight: 600;
                letter-spacing: .1rem;
                text-decoration: none;
                text-transform: uppercase;
            }

            .m-b-md {
                margin-bottom: 30px;
            }
        </style>
    </head>
    <body>
        <div class="flex-center position-ref full-height">
            @if (Route::has('login'))
                <div class="top-right links">
                    @auth
                        <a href="{{ url('/home') }}">Домашняя страница</a>
                    @else
                        <a class="text-muted" href="{{ route('login') }}">Вход</a>
                    <!--                <a class="text-muted" href="{{ route('register') }}">Регистрация</a> -->
                    @endauth
                </div>
            @endif
```

```
            <div class="content">
                <div class="title m-b-md">
                    Lateness
                </div>

            </div>
        </div>
    </body>
</html>
```

Листинг assignsubject.blade.php

```
@if(Auth::user()->permission_id == 1 || Auth::user()->permission_id == 2)

@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-6">
          @include('layouts.parts.errors')
            <div class="card">
                <div class="card-header"> Назначить предмет </div>

                <div class="card-body">

                  <form method="POST" action="{{route('assignsubject.store')}}">
                   {{csrf_field()}}
```

```
<div class="form-group row">

<div class="col-md-9">

<label for="number" class="col-form-label text-center">{{ __('Номер кафедры') }}</label>
<select class="form-control" name="pulpit_id" id="pulpit_id" onchange="selectfunpul()">
<option value="0" disabled selected> Выберете кафедру </option>
@foreach($pulpits as $pulpit)
<option value="{{$pulpit->id}}"> {{ $pulpit->number }} | {{ $pulpit->name }} </option>
@endforeach
</select>

<div id="first" hidden>
<label for="number" class="col-form-label text-center">{{ __('Преподаватель') }}</label>
<select class="form-control" name="teacher_id" id="teacher_id" onchange="selectfunpulteach()">

</select>
</div>

<div id="second" hidden>

<label for="name" class="col-form-label text-center">{{ __('Предмет') }}</label>
```

```
                <select class="form-control" name="input_sub" id="input_sub">
                 <option value="0" disabled selected> Выберете предмет </option>
                 @foreach($subjects as $subject)
                  <option value="{{$subject->id}}"> {{ $subject->name }} </option>
                 @endforeach
                </select>

                <br />

                <div class="form-group row">

                 <div class="col-md-6">
                  <input class="btn btn-default" type="button" id="button1" value="Добавить предмет">
                 </div>

                 <br /> <br />

                 <div class="col-md-6">
                  <input class="btn btn-default" type="button" id="button2" value="Удалить предмет">
                 </div>

                </div>

                <input type="text" value="1" class="form-control" name="coll" id="coll" hidden>
```

```
                </div>

            </div>

            <br />

            <div class="col-md-3">
                <button type="submit" class="btn btn-default"> Назначить </button>
                <br /> <br />
                <button class="btn btn-default"> <a class="text-dark" href="{{ url('/subject') }}"> Назад </a> </button>
            </div>

        </div>

    </form>

</div>

</div>
</div>
</div>
@endsection

@endif

@if(Auth::user()->permission_id == 3)
@section('content')
```

```
<div class="container">
   <div class="row justify-content-center">
      <div class="col-md-8">
        <div class="card">
          <div class="card-body">
            Данная страница Вам не доступна. Обратитесь к администратору для смены прав доступа.
          </div>
        </div>
        <br/ >
        <div class="row justify-content-center">
          <div class="col-md-5"></div>
          <div class="col-md-2">
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/home') }}"> Назад </a>
              </div>
            </div>
          </div>
          <div class="col-md-5"></div>
        </div>
      </div>
      </div>
   </div>
</div>
@endsection
@endif

@section('js')
  <script src="{{ asset('js/select-subj.js') }}" defer></script>
```

```
<script src="{{ asset('js/selectsubject.js') }}" defer></script>

<script>
  $(document).ready(function () {
    $("#pulpit_id")
      .change(function() {
        if ($("#pulpit_id").val() != null) {

          var token = '{{ csrf_token()}}';
          var val = $("#pulpit_id").val();

          $.ajax({
            url: "{{ route('selectOption') }}",
            dataType: 'json',
            type: 'POST',
            data: { pulpit_id: val, _token: token},
            success:function(response) {
              $('#teacher_id').html(response.html);
            }
          });

        }
      })
      .trigger( "change" );
  });
</script>

@endsection
```

Листинг AssignSubjectController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Subject;
use App\Pulpit;
use App\Teacher;
use DB;

class AssignSubjectController extends Controller
{

  public function create()
  {

    $pulpits = Pulpit::all();
    $subjects = Subject::all();

    return view('subject.assignsubj', compact('pulpits'), compact('subjects'));

  }

  public function index()
  {

  }

  public function store(Request $request)
```

```
  {

    $num = $request['coll'];
    $pulpit_id = $request['pulpit_id'];
    $teacher_id = $request['teacher_id'];

    $input_sub[0] = $request['input_sub'];

    $teacher_pulpit_id = DB::select("
      SELECT
        teacher_pulpits.id
      FROM
        teacher_pulpits
      WHERE
        teacher_pulpits.teacher_id = ".$teacher_id." AND teacher_pulpits.pulpit_id =
".$pulpit_id
    );

    for ( $i = 1; $i < $num; $i++ ) {
      $input_sub[$i] = $request['input_sub'.$i];
    }

    for ( $i = 0; $i < $num; $i++ ) {
      DB::table('subject_teacher_pulpits')->insert([
        'teacher_pulpit_id'=>$teacher_pulpit_id[0]->id,
        'subject_id'=>$input_sub[$i]
      ]);
    }

    return back();
```

```
  }

  public function destroy(Request $request)
  {

  }

  public function selectOption(Request $request)
  {

    $teachers = DB::select("
    SELECT
      teachers.id, teachers.name, teachers.surname, teachers.degree
    FROM
      teachers
    WHERE
      teachers.id IN (SELECT teacher_pulpits.teacher_id FROM teacher_pulpits
WHERE teacher_pulpits.pulpit_id = ". $request->pulpit_id .")"
    );

    $view = view("subject.selectsub", compact('teachers'))->render();

    return response()->json(['status'=>'success', 'html'=>$view]);
  }
}
```

Листинг group/create.blade.php

```
@if(Auth::user()->permission_id == 1 || Auth::user()->permission_id == 2)
```

```
@extends('layouts.app')

@section('content')
<div class="container">
   <div class="row justify-content-center">
      <div class="col-md-6">
       @include('layouts.parts.errors')
         <div class="card">
            <div class="card-header"> Добавление группы </div>

            <div class="card-body">

             <form method="POST" action="{{route('group.store')}}">
              {{csrf_field()}}

              <div class="form-group row">

               <div class="col-md-6">
                <label for="number" class="col-form-label text-center">{{ __('Номер') }}</label>
                <input type="text" class="form-control" name="number">
               </div>

               <div class="col-md-6 py-3 px-5">
                <button type="submit" class="btn btn-default"> Добавить </button>
                <br /> <br />
                <button class="btn btn-default"> <a class="text-dark" href="{{ url('/group') }}"> Назад </a> </button>
               </div>
```

```
                </div>

              </form>

            </div>

          </div>
        </div>
    </div>
</div>
@endsection

@endif

@if(Auth::user()->permission_id == 3)
@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
          <div class="card">
            <div class="card-body">
              Данная страница Вам не доступна. Обратитесь к администратору для смены прав доступа.
            </div>
          </div>
          <br/ >
          <div class="row justify-content-center">
            <div class="col-md-5"></div>
            <div class="col-md-2">
```

```
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/home') }}"> Назад </a>
              </div>
            </div>
          </div>
          <div class="col-md-5"></div>
        </div>
      </div>
      </div>
   </div>
</div>
@endsection
@endif
```

Листинг group/idex.blade.php

```
@if(Auth::user()->permission_id == 1 || Auth::user()->permission_id == 2)

@extends('layouts.app')

@section('content')
<div class="container">
   <div class="row justify-content-center">
      <div class="col-md-8">

            <div class="row justify-content-center">

              <div col-md-2> </div>
              <div col-md-3>
```

```
            <div class="card">
              <div class="card-body">
                <a  class="text-muted"  href="{{route('group.create')}}">  Добавить
</a>
              </div>
            </div>
          </div>
          <div col-md-2>
                 
          </div>
          <div col-md-3>
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/home') }}"> Назад </a>
              </div>
            </div>
          </div>
          <div col-md-2> </div>

          </div>

        <br />
        <div class="card">
          <div class="card-header"> Список групп </div>

          @forelse($groups as $group)
          <div class="card-body">
            <div class="row">

              <div class="col-md-8">
```

```
                {{ $group->number }}
              </div>

              <div class="col-md-2">
                <form action="#" method="post">
                  {{csrf_field()}}
                  <button type="submit" class="btn btn-sm btn-default"><i class="fas fa-edit"></i> Обновить </button>
                </form>
              </div>
              <div class="col-md-2">
                <form action="{{ route('group.destroy', $group->id) }}" method="post">
                  {{csrf_field()}}
                  {{method_field('DELETE')}}
                  <button type="submit" class="btn btn-sm btn-default"><i class="fas fa-trash-alt"></i> Удалить </button>
                </form>
              </div>

            </div>
          </div>

          @empty

          <p> Нет предметов </p>

          @endforelse

        </div>
```

```
          </div>
        </div>
    </div>
</div>
@endsection

@endif

@if(Auth::user()->permission_id == 3)
@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
          <div class="card">
            <div class="card-body">
              Данная страница Вам не доступна. Обратитесь к администратору для смены прав доступа.
            </div>
          </div>
          <br/ >
          <div class="row justify-content-center">
            <div class="col-md-5"></div>
            <div class="col-md-2">
              <div class="card">
                <div class="card-body">
                  <a class="text-muted" href="{{ url('/home') }}"> Назад </a>
                </div>
              </div>
            </div>
```

```
            <div class="col-md-5"></div>
          </div>
        </div>
        </div>
    </div>
</div>
@endsection
@endif
```

Листинг teacher/create.blade.php

```
@if(Auth::user()->permission_id == 1 || Auth::user()->permission_id == 2)

@extends('layouts.app')

@section('content')

<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-6">
          @include('layouts.parts.errors')
            <div class="card">
                <div class="card-header"> Добавление преподавателя </div>

                <div id="div">

                </div>


                <div class="card-body">
```

```
<form method="POST" action="{{route('teacher.store')}}">
 {{csrf_field()}}

 <div class="form-group row">

  <div class="col-md-9">

   <label for="number" class="col-form-label text-center">{{ __('Номер кафедры') }}</label>
   <select class="form-control" name="pulpit_id" id="pulpit_id" onchange="selectfunpul()">
    <option value="0" disabled selected> Выберете кафедру </option>
    @foreach($pulpits as $pulpit)
     <option value="{{$pulpit->id}}"> {{ $pulpit->number }} {{ $pulpit->name }} </option>
    @endforeach
   </select>

   <br />

   <div class="form-group row">

    <div class="col-md-6">
     <input class="btn btn-default" type="button" id="button1" value="Добавить кафедру">
    </div>

    <br /> <br />
```

```
                <div class="col-md-6">
                  <input class="btn btn-default" type="button" id="button2" value="Удалить кафедру">
                </div>

              </div>

              <input type="text" value="1" class="form-control" name="coll" id="coll" hidden>

              <div id="teachers" hidden>
                <label for="surname" class="col-form-label text-center">{{ __('Фамилия') }}</label>
                <input type="text" class="form-control" name="surname">

                <label for="name" class="col-form-label text-center">{{ __('Имя') }}</label>
                <input type="text" class="form-control" name="name">

                <label for="patronymic" class="col-form-label text-center">{{ __('Отчество') }}</label>
                <input type="text" class="form-control" name="patronymic">

                <label for="name" class="col-form-label text-center">{{ __('Пол') }}</label>
                <select class="form-control" name="sex" id="sex">
                  <option value="0" disabled selected> Выберете пол </option>
                    <option value="Мужской"> Мужской </option>
                    <option value="Женский"> Женский </option>
```

```
                    </select>

                    <label for="degree" class="col-form-label text-center">{{ __('Учёная степень / Должность') }}</label>
                    <input type="text" class="form-control" name="degree">
                  </div>
                </div>

                <div class="col-md-3">
                  <button type="submit" class="btn btn-default"> Добавить </button>
                  <br /> <br />
                  <button class="btn btn-default"> <a class="text-dark" href="{{ url('/teacher') }}"> Назад </a> </button>
                </div>

              </div>

            </form>

          </div>

        </div>
      </div>
    </div>
</div>
@endsection

@endif
```

```
@if(Auth::user()->permission_id == 3)
@section('content')
<div class="container">
   <div class="row justify-content-center">
      <div class="col-md-8">
        <div class="card">
          <div class="card-body">
            Данная страница Вам не доступна. Обратитесь к администратору для смены прав доступа.
          </div>
        </div>
        <br/ >
        <div class="row justify-content-center">
          <div class="col-md-5"></div>
          <div class="col-md-2">
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/home') }}"> Назад </a>
              </div>
            </div>
          </div>
          <div class="col-md-5"></div>
        </div>
      </div>
      </div>
   </div>
</div>
@endsection
@endif
```

```
@section('js')
  <script src="{{ asset('js/select-pulpit.js') }}" defer></script>
  <script src="{{ asset('js/select-asd.js') }}" defer></script>
@endsection
```

Листинг teacher/index.blade.php

```
@if(Auth::user()->permission_id == 1 || Auth::user()->permission_id == 2)

@extends('layouts.app')

@section('content')

<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">

            <div class="row justify-content-center">

              <div col-md-2> </div>
              <div col-md-3>
                <div class="card">
                  <div class="card-body">
                    <a  class="text-muted"  href="{{route('teacher.create')}}">  Добавить
</a>
                  </div>
                </div>
              </div>
              <div col-md-2>
                     
```

```
  </div>
  <div col-md-3>
    <div class="card">
      <div class="card-body">
        <a class="text-muted" href="{{ url('/home') }}"> Назад </a>
      </div>
    </div>
  </div>
  <div col-md-2> </div>

  </div>

<br />
<div class="card">
    <div class="card-header"> Список преподавателей </div>

    @forelse($teachers as $teacher)

    <div class="card-body">
      <div class="row">

        <div class="col-md-8">
          {{ $teacher->surname }}
          {{ $teacher->name }}
          {{ $teacher->patronymic }}
          |
          {{ $teacher->degree }}
        </div>

        <div class="col-md-2">
```

```
                <form action="#" method="post">
                  {{csrf_field()}}
                  <button type="submit" class="btn btn-sm btn-default"><i class="fas fa-edit"></i> Обновить </button>
                </form>
              </div>
              <div class="col-md-2">
                <form action="{{ route('teacher.destroy', $teacher->id) }}" method="post">
                  {{csrf_field()}}
                  {{method_field('DELETE')}}
                  <button type="submit" class="btn btn-sm btn-default"><i class="fas fa-trash-alt"></i> Удалить </button>
                </form>
              </div>

            </div>
          </div>

          @empty

          <p> Нет преподавателей </p>

          @endforelse

        </div>

      </div>
    </div>
```

```
</div>
@endsection

@endif

@if(Auth::user()->permission_id == 3)
@section('content')
<div class="container">
   <div class="row justify-content-center">
      <div class="col-md-8">
        <div class="card">
          <div class="card-body">
            Данная страница Вам не доступна. Обратитесь к администратору для смены прав доступа.
          </div>
        </div>
        <br/ >
        <div class="row justify-content-center">
          <div class="col-md-5"></div>
          <div class="col-md-2">
            <div class="card">
              <div class="card-body">
                <a class="text-muted" href="{{ url('/home') }}"> Назад </a>
              </div>
            </div>
          </div>
          <div class="col-md-5"></div>
        </div>
      </div>
      </div>
```

```
    </div>
</div>
@endsection
@endif
```

Листинг web.php

```
<?php

Route::get('/', function () {
    return view('welcome');
});

Auth::routes();

Route::get('/home', 'HomeController@index')->name('home');

Auth::routes();

Route::get('/home', 'HomeController@index')->name('home');

Route::resource('/subject', 'SubjectController');
Route::resource('/teacher', 'TeacherController');
Route::resource('/pulpit', 'PulpitController');
Route::resource('/group', 'GroupController');
Route::resource('/newuser', 'NewUserController');
Route::resource('/student', 'StudentController');
Route::resource('/jornal', 'JornalController');

Route::resource('/assignsubject', 'AssignSubjectController');
```

Route::post('/assignsubject/select', 'AssignSubjectController@selectoption')->name('selectOption');

Листинг database\migrations\2018_05_17_155920_create_teacher_table.php

```
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreateTeacherTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('teachers', function (Blueprint $table) {
            $table->increments('id');
            $table->string('surname');
            $table->string('name');
            $table->string('patronymic');
            $table->string('sex');
            $table->string('degree');
            $table->timestamps();
        });
```

```
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('teachers');
    }
}
```

Листинг database\migrations\2018_05_17_154847_add_foreign_to_student_table.php

```
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class AddForeignToStudentTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
```

```
    public function up()
    {
      Schema::table('students', function($table)
      {
        $table->integer('user_id')->unsigned();
        $table->foreign('user_id')->references('id')->on('users')->onDelete("NO
ACTION");

        $table->integer('group_id')->unsigned();
        $table->foreign('group_id')->references('id')->on('groups')-
>onDelete('cascade');
      });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
      Schema::table('students', function($table)
      {
        $table->dropColumn('user_id');
        $table->dropColumn('group_id');
      });
    }
}
```

Листинг .env

```
APP_NAME=Lateness
APP_ENV=local
APP_KEY=base64:ohPvc3Y43GuapZ4XMhVJKMmL+Rb9kynyjJKkcjNiLjE=
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=db_jornal
DB_USERNAME=root
DB_PASSWORD=root

BROADCAST_DRIVER=log
CACHE_DRIVER=file
SESSION_DRIVER=file
SESSION_LIFETIME=120
QUEUE_DRIVER=sync

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_DRIVER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
```

```
MAIL_USERNAME=975896021598ba
MAIL_PASSWORD=5f8c0a650ea035
MAIL_ENCRYPTION=null

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_APP_CLUSTER=mt1

MIX_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
MIX_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```