

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
1 ОБЩАЯ ЧАСТЬ.....	6
1.1 Описание предметной области.....	6
1.2 Обзор аналогов.....	13
2 СПЕЦИАЛЬНАЯ ЧАСТЬ.....	16
2.1 Аналитическая часть.....	16
2.1.1 Постановка задачи.....	16
2.1.2 Разработка алгоритма веб-приложения.....	16
2.1.2.1 Разработка блок-схемы алгоритма.....	17
2.1.2.2 Разработка диаграммы вариантов использования.....	19
2.1.2.3 Разработка схемы базы данных.....	20
2.1.3 Обоснование выбора языков программирования.....	22
2.1.4 Обоснование выбора инструментальных средств.....	23
2.2 Практическая часть.....	29
2.2.1 Описание процесса разработки веб-приложения.....	29
2.2.1.1 Установка программного обеспечения.....	29
2.2.1.2 Настройка подключения к базе данных.....	33
2.2.1.3 Разработка серверной части.....	35
2.2.1.4 Разработка клиентской части.....	40
2.2.2 Интерфейс разработанного веб-приложения.....	45
2.2.3 Разработка руководства пользователя.....	46
ЗАКЛЮЧЕНИЕ.....	53
СПИСОК ЛИТЕРАТУРЫ.....	56
ПРИЛОЖЕНИЕ А.....	60
ПРИЛОЖЕНИЕ Б.....	63
ПРИЛОЖЕНИЕ В.....	66

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ		
Изм.	Лист	№ докум.	Подпись	Дата			
Разраб.		Ханнанов А. Р.			Разработка веб-приложения «Advanced Schedule»	Лит.	Лист
Провер.		Валеева Г. Р.					3
Реценз.		Маннанов А. К.					68
Н. Контр.		Ковалева А. А.				ГБПОУ УГКТИд	
Утверд.		Шепелева Н. Ю.					

ВВЕДЕНИЕ

В настоящее время наблюдается непрерывное развитие информационных технологий в организациях разного рода деятельности и рост темпов появления новой информации. Это обусловлено стремлением к сокращению времени обработки и анализа данных. В связи с тем, что объем рабочей и иной информации является обширным, в любой организации появляется необходимость использования автоматизированной обработки данных, которая обеспечила бы наиболее эффективную работу. Большая часть организаций выбирают компьютеризированные способы, позволяющие эффективно хранить, структурировать и систематизировать большие объемы разнообразных данных [33].

Одной из важнейших проблем качественной организации учебного процесса в высшем учебном заведении является задача создания автоматизированного учебного расписания. Многие люди считают, что опытные диспетчеры могут вручную составить расписание, оптимальное для учебного процесса и общественной жизни образовательного учреждения, но, учитывая увеличение количества студентов, укрупнение образовательных учреждений, необходимость использования и учёта большого количества данных, такое мнение, безусловно, можно считать ошибочным [17].

Потенциал вычислительной техники уже давно позволяет поставить и решить задачу создания автоматизированной системы составления расписания, которая упростила бы часть организации деятельности учебной части. Исследованию этой проблемы и ее разработке посвящены программы «1С:Автоматизированное составление расписания. Колледж», «Экспресс-расписание Колледж», «АВТОРасписание».

Однако в современный экономический кризис образовательному учреждению довольно проблематично приобрести готовую программную продукцию, которую без труда можно было бы перенести на почву учебного заведения. Более реалистичной является подготовка частной системы,

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		4

позволяющей планировать и составлять расписания занятий студентов и преподавателей [16]. Этим и обуславливается актуальность исследования.

Разработка такого рода программ требует учета специфики образовательного учреждения.

Объектом исследования дипломной работы является разрабатываемая автоматизированная информационная система составления расписания веб-приложение «Advanced Schedule» [8].

Предмет исследования: особенности автоматизированной информационной системы составления расписания.

Целью работы является разработка информационной системы веб-приложения «Advanced Schedule».

Для достижения данной цели необходимо решить следующие задачи:

- рассмотрение аналогов разрабатываемой информационной системы, их ключевых достоинств и недостатков;
- теоретический анализ и обработка практической и методической литературы по дипломной работе;
- рассмотрение классификации информационных систем и принципа автоматизации бизнес-процессов;
- выявление требований к информационной системе и постановка задачи на разработку информационной системы;
- проектирование и реализация приемлемой для учебного отдела автоматизированной системы составления расписания.

Практическая значимость данной работы заключается в том, что полученная, в результате исследования, система будет иметь возможность применения в деятельности Уфимского государственного колледжа технологии и дизайна, а также может быть использован и другими учебными заведениями для совершенствования анализа результатов образовательного процесса [4].

1. ОБЩАЯ ЧАСТЬ

1.1 Описание предметной области

Расписания можно использовать для организации и координации деятельности в различных условиях, таких как школа, бизнес и личная жизнь. Они могут помочь людям оставаться организованными и обеспечивать эффективное и своевременное выполнение задач.

Расписание - это план выполнения конкретной задачи или действия, часто в течение определенного периода времени. План включает в себя список задач или событий, которые необходимо выполнить, и время, в которое они должны произойти [20].

Существует несколько типов расписаний, которые можно использовать в зависимости от потребностей и целей человека или организации:

- **годовой график** - это план, в котором излагаются действия и задачи, которые необходимо выполнить в течение года. Может включать ежемесячные расписания, а также любые повторяющиеся события или действия, которые происходят ежегодно;

- **график проекта** - это план, в котором излагаются задачи и этапы, которые необходимо выполнить в рамках конкретного проекта. Включает график выполнения каждой задачи, а также может включать необходимые ресурсы, зависимости между задачами и крайние сроки;

- **сменный график** - это план, в котором указывается время, в которое должны работать разные сотрудники. Может включать разные смены для разных дней недели, а также может включать выходные и праздничные дни;

- **расписание занятий** - это план, в котором указаны время и дни, в которые запланированы занятия для различных занятий или разделов курса. Обычно используется в школах и университетах [21];

- **расписание собраний** - это план, в котором указаны время и место запланированных собраний. Это могут быть регулярные встречи, а также разовые или специальные встречи;

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

– график поездок - это план, в котором излагается маршрут поездки, включая время и место отправления/прибытия рейсов, гостиниц и других мероприятий, связанных с поездками;

– расписание событий - это план, в котором указаны сроки и детали события, например конференции, концерта или торговой выставки. Может включать время и место проведения различных сессий или мероприятий, а также любые специальные меры или приспособления, которые необходимо сделать в заданный срок;

– график технического обслуживания - это план, в котором излагаются задачи и действия, которые необходимо выполнить для технического обслуживания и ремонта оборудования или помещений. Может включать график выполнения этих задач, а также сведения о ресурсах и персонале, необходимых для их выполнения;

– производственный график - это план, в котором излагаются задачи и действия, связанные с производством товаров или услуг. Может включать график выполнения каждой задачи, а также сведения о ресурсах и персонале, необходимых для их выполнения.

Различия в задачах расписания и методах их решения определяются наличием и видом связей между заявками на формирование расписания [23]. Это позволяет ввести следующую классификацию:

– расписание является множеством независимых друг от друга действий. То есть, любое действие, являющееся элементом расписания, может быть расположено в любом таймслоте интервала расписания с учетом обязательных ограничений. К этому типу относятся расписания экзаменов, занятий и т.п. Для этого типа задач расписания характерна возможность произвольного выбора заявок при его формировании. То же самое относится к выбору элементов расписания при его оптимизации. Для визуализации в расписании одновременно происходящих событий используется двумерной представление времени;

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

– расписание является множеством независимых друг от друга векторов действий, являющихся элементами расписания. Например, расписание движения пассажирского транспорта. Расписание будет формироваться из векторов заявок, каждая из которых включает прохождение одного перегона между станциями и пребывание на конечной станции перегона. Формирование расписания основано на возможности произвольного выбора векторов заявок и их включении в расписание на произвольное время начала первого действия с учетом обязательных ограничений. Оптимизация расписания обеспечивается перестановкой элементов расписания – векторов действий, то есть изменением времени начала первого действия. Круговое представление расписания для любого ресурса системы в данном случае позволяет наглядно оценить возникающую неравномерность действий расписания во времени;

– расписание является множеством независимых друг от друга иерархий действий. К этому типу расписаний относится, например, календарный график малоэтажного строительства поселка, где каждая иерархия представляет необходимые работы для возведения и обустройства одного здания. При формировании расписания и его оптимизации должны рассматриваться иерархии заявок и действий. Элементами расписания будут иерархии действий. Наиболее удобными для работы с данными расписаниями являются диаграммы Ганта;

– расписание является множеством независимых друг от друга сетевых структур действий. К этому типу расписаний относится большинство календарных графиков мультипроектного планирования. Каждая сеть представляет отдельный проект. Проекты могут быть технологически независимыми, но объединенными по потребляемым ресурсам, прежде всего по возобновляемым ресурсам. При формировании расписания и его оптимизации должны рассматриваться сетевые структуры заявок и действий. Элементами расписания будут сетевые структуры действий. Поскольку

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		8

иерархические структуры являются частным случаем сетевых структур, то методы формирования расписания в обоих случаях очень близки.

Расписание занятий может включать следующую информацию:

- название и номер курса;
- имя преподавателя;
- дни и время встречи;
- местонахождение, например, номер кабинета;
- название предмета [9].

Расписания занятий могут также включать сведения о формате занятия, например, очная форма обучения, онлайн или их сочетание. Расписание занятий обычно распространяется среди студентов в начале семестра и может быть изменено из-за непредвиденных обстоятельств или обновлений [24].

Виды расписания по способу представления информации:

- расписание, которое распечатывается на бумаге и распространяется среди учащихся, учителей (преподавателей) и других заинтересованных лиц. Этот тип расписания удобен для предоставления физической ссылки, к которой можно легко получить доступ и обратиться к ней, но может потребоваться больше усилий для обновления и распространения;

- расписание, которое представлено в электронном виде либо на веб-сайте, либо через программное приложение. Этот тип расписания удобен для предоставления более динамичной и интерактивной справочной информации, которую можно легко обновлять и получать к ней доступ, но может потребоваться доступ к компьютеру или другому устройству;

- расписание, которое представлено через мобильное приложение или другую мобильную платформу, что позволяет получить к нему доступ со смартфона или другого мобильного устройства. Этот тип расписания удобен для предоставления удобного и переносимого справочника, к которому можно легко получить доступ на ходу, но для которого может потребоваться подключение для передачи данных или другие ресурсы;

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

– расписание, представленное в комбинации печатного и электронного форматов, что позволяет получить к нему доступ с помощью различных средств. Этот тип расписания удобен для обеспечения гибкости и доступности, но может потребовать больше усилий для координации и обслуживания расписания.

Расписание может быть представлено в различных форматах:

- таблица: расписание, представленное в виде таблицы с различными столбцами, такими как время, место, участники;
- график: расписание, представленное в виде графика, где каждое событие представлено в виде точки на осях времени и темы;
- список: расписание, представленное в виде простого списка событий с указанием времени и места каждого события;
- календарь: расписание, представленное в виде календаря, где каждое событие отображается в соответствующую дату;
- недельное расписание, где каждая колонка – учебная группа, строки это конкретный временной отрезок, группа строк – один день [21].

Расписание может быть составлено на различный период времени:

- еженедельное расписание: составляется каждую неделю;
- расписание семестра: это расписание, в котором излагаются занятия, которые необходимо выполнить в течение семестра или академического семестра. Это может включать время занятий, перерывы, экзамены и другие действия, такие как проекты или презентации;
- расписание курса: это расписание, в котором указаны занятия и задачи, которые необходимо выполнить в течении курса [25].

Составление расписания занятий в образовательных учреждениях среднего профессионального образования осуществляет учебная часть данной образовательной организации.

Работа по составлению расписания проводится заведующим учебной частью, либо курирующим учебную работу заместителем директора при

отсутствии должности заведующего учебной частью, на основе сведений о распределении учебной нагрузки между педагогическими работниками колледжа в пределах рабочей недели.

При составлении расписания учитывается участие педагогических работников в научной, учебно-методической и воспитательной работе структурного подразделения колледжа, реализующего образовательные программы СПО. Для внешних совместителей и педагогических работников, привлеченных на основе договора гражданско-правового характера, учитывается занятость по основному месту работы [26].

При составлении расписания учитываются факторы с точки зрения возможности проведения занятий в заданный промежуток времени, в конкретном учебном кабинете и с определенным преподавателем, учитывая учебную нагрузку на группы и количество часов по отдельным учебным предметам специальности [27].

Методы составления расписания:

- создание расписания вручную путем записи сведений о каждом занятии на листе бумаги или в электронной таблице;
- использование предварительно разработанного шаблона для создания расписания, например таблицы или шаблона календаря в текстовом процессоре или программе для работы с электронными таблицами;
- использование специализированного программного обеспечения, предназначенного для создания расписаний, такого как программное обеспечение для управления проектами или приложения-календари;
- использование онлайн-инструментов, позволяющих создавать расписание путем перетаскивания событий в представление календаря.

Среди многочисленных методов, применяющихся в автоматизированном формировании расписаний, наиболее распространёнными являются: метод моделирование отжига, метод штрафов и запретов и эволюционные алгоритмы. Перечисленные методы применяются для

оптимизации расписаний, а общим для них является введение критериев оптимальности и целевой функции оптимизации взамен требований к расписанию. Методы формирования начальных расписаний, к которым будут применены методы оптимизации, различны и определяются предметной областью метода [20].

При использовании специального программного обеспечения расписание составляется на основе базы данных, которую пользователь изначально наполняет необходимыми параметрами, выполнив следующие шаги:

- сбор информации: первым шагом в создании расписания занятий является сбор всей необходимой информации, включая названия и номера групп, имена преподавателей, дни и время собраний, аудитории и предметы;
- после того, как вся необходимая информация будет собрана, ее можно ввести в программное обеспечение. Может включать создание отдельных записей для каждой группы и ввод соответствующих сведений;
- установка ограничений и предпочтений: программное обеспечение может позволить пользователям устанавливать ограничения и предпочтения, которые помогают гарантировать, что расписание выполнимо и соответствует их потребностям. Например, пользователи могут указать, что определенные занятия должны или не должны быть запланированы на определенное время, или что они предпочитают, чтобы определенные занятия были запланированы на определенное время [29];
- настройка представления расписания: после ввода всей необходимой информации и настройки любых ограничений и предпочтений программное обеспечение может создать один или несколько вариантов расписания. Расписание может отображаться в виде календаря или списка, можно просматривать и сравнивать для определения наилучшего соответствия;
- доработка расписания: после того, как желаемое расписание определено, его можно просмотреть и доработать. Этот шаг может включать в

себя внесение любых необходимых корректировок или изменений и обеспечение правильности всей необходимой информации.

Когда расписание готово, оно публикуется, чтобы учащиеся, преподаватели и другие заинтересованные стороны могли получить к нему доступ. Публикация может включать экспорт расписания в другой формат, например, PDF или электронную таблицу, или предоставление доступа к нему в Интернете через веб-интерфейс [19].

1.2 Обзор аналогов

В настоящее время существует большое количество автоматизированных информационных систем, осуществляющих решение задачи автоматизации составления расписания в образовательном учреждении [35].

«Экспресс-расписание Колледж» - программа для автоматизации составления расписания учебных занятий в училищах, колледжах и профессиональных лицеях. Программа автоматически составляет основное расписание, позволяет вести учет выполненных часов, ежедневные изменения расписания, формирует разнообразные отчеты. Интерфейс окна программы представлен на рисунке 1.

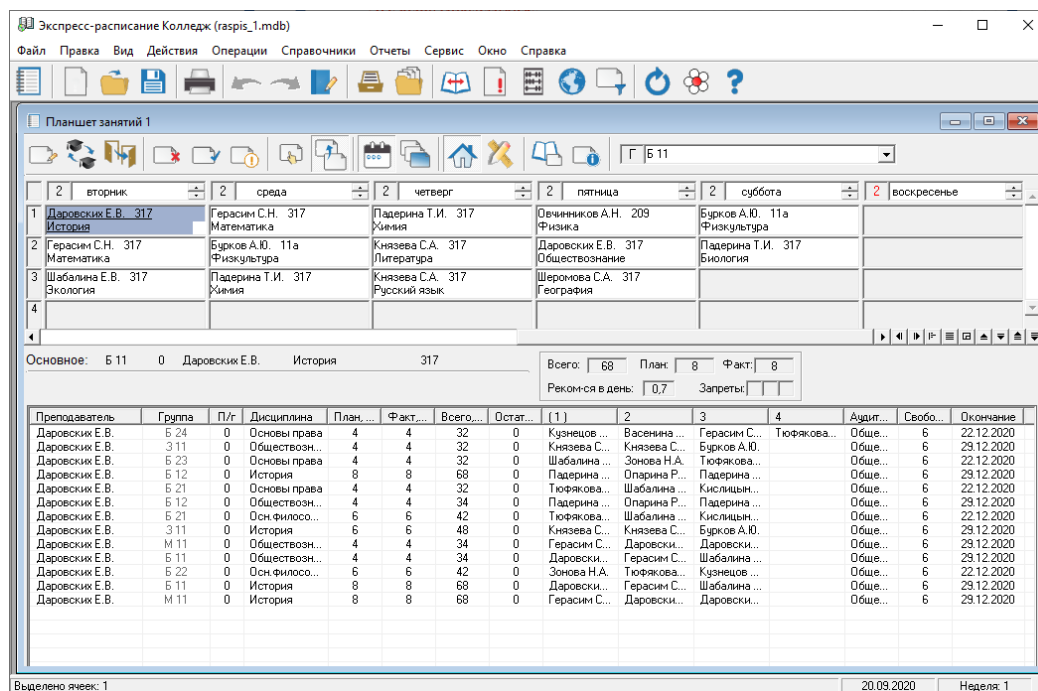


Рисунок 1 - Интерфейс окна программы «Экспресс-расписание Колледж»

Преимущества:

- при составлении расписания помогает избежать накладок;
- можно в любое время отследить нагрузку по преподавателю, по группе, по аудитории;
- недорогая и работает в сети.

Недостатки:

- не может учесть специфику некоторых учреждений, расписание приходится составлять вручную;
- программа может оставлять пустые окна в расписании, имеется необходимость в ручном режиме ликвидировать данный недостаток.

«1С:Автоматизированное составление расписания. Колледж» - продукт является самостоятельным программным продуктом, предназначенным для решения задач автоматизированного составления учебных расписаний [33]. Интерфейс окна программы представлен на рисунке 2.

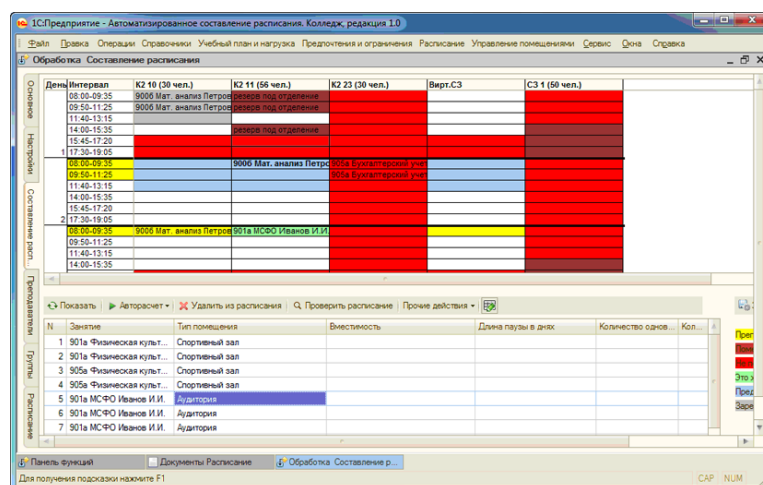


Рисунок 2 - Интерфейс окна программы «1С:Автоматизированное составление расписания. Колледж»

Преимущества:

- решение прорабатывается под конкретную организацию;
- интегрируется с другими решениями на базе 1С.

Недостатки:

- высокая стоимость;
- отсутствует программа разработки учебных планов.

Система «АВТОРасписание» предназначена для быстрого, удобного и качественного составления расписаний занятий и сопровождения их в течение всего учебного года. Программа достаточно проста в освоении. Имеется подробное руководство пользователя и справочная система, где описаны все возможности и способы работы с программой [16].

Программа отличается уникальным, мощным алгоритмом построения и оптимизации расписания. Этот алгоритм является оригинальной авторской разработкой. Он позволяет находить оптимальные решения даже при очень сложных исходных данных. Интерфейс окна программы «АВТОРасписание» представлен на рисунке 3.

Рисунок 3 - Интерфейс окна программы «АВТОРасписание»

Преимущества:

- сохранение расписание в различных форматах;
- возможность выбора режима составления расписания.

Недостатки:

- автоматически расписание составляется долго, и требует большой корректировки вручную;
- программа требует продвинутых способностей от пользователя.

2. СПЕЦИАЛЬНАЯ ЧАСТЬ

2.1 Аналитическая часть

2.1.1 Постановка задачи

Веб-приложение для составления расписания занятий «Advanced Schedule» будет основано на веб-сервере в операционной системе Linux [1].

Веб-приложение для составления расписания занятий «Advanced Schedule» принесёт революционный интерфейс в сфере программ по составлению расписания занятий, будет брать на себя задачи по проверке расписания на пересечения и накладки. Также благодаря веб-интерфейсу приложения, оно будет доступно на любых устройствах с выходом в сеть интернет и с браузером.

Веб-приложение «Advanced Schedule» будет иметь следующие основные функции:

- возможность просмотра и фильтрации информации в базе данных, например, группы, предметы, преподаватели (пользователи) и любые другие данные;
- добавление новой информации, составление расписания занятий, при этом система способна проверять расписание занятий на пересечение;
- просмотр расписания занятий любыми пользователями [3].

2.1.2 Разработка алгоритма веб-приложения

Алгоритм - совокупность заданных правил решения задачи или набор инструкций, описывающих порядок действий для решения задачи.

Разработка алгоритма веб-приложения «Advanced Schedule» включает следующие этапы [12]:

- разработка блок-схем;
- разработка диаграммы вариантов использования;
- разработка схемы базы данных.

2.1.2.1 Разработка блок-схемы алгоритма

Использование веб-приложения «Advanced Schedule» будет выполняться с двумя сценариями:

- просмотр расписания обычными пользователями, такими как студенты, преподаватели, родители студентов;
- составление расписания администраторами веб-приложения.

Для разработки алгоритмов веб-приложения будут использованы разветвляющиеся блок-схемы.

Блок-схема - распространённый тип графических моделей, описывающих алгоритмы или процессы, в которых отдельные шаги изображаются в виде блоков различной формы, соединённых между собой линиями, указывающими направление последовательности [15]. На рисунке 4 представлена блок-схема алгоритма просмотра расписания [7].

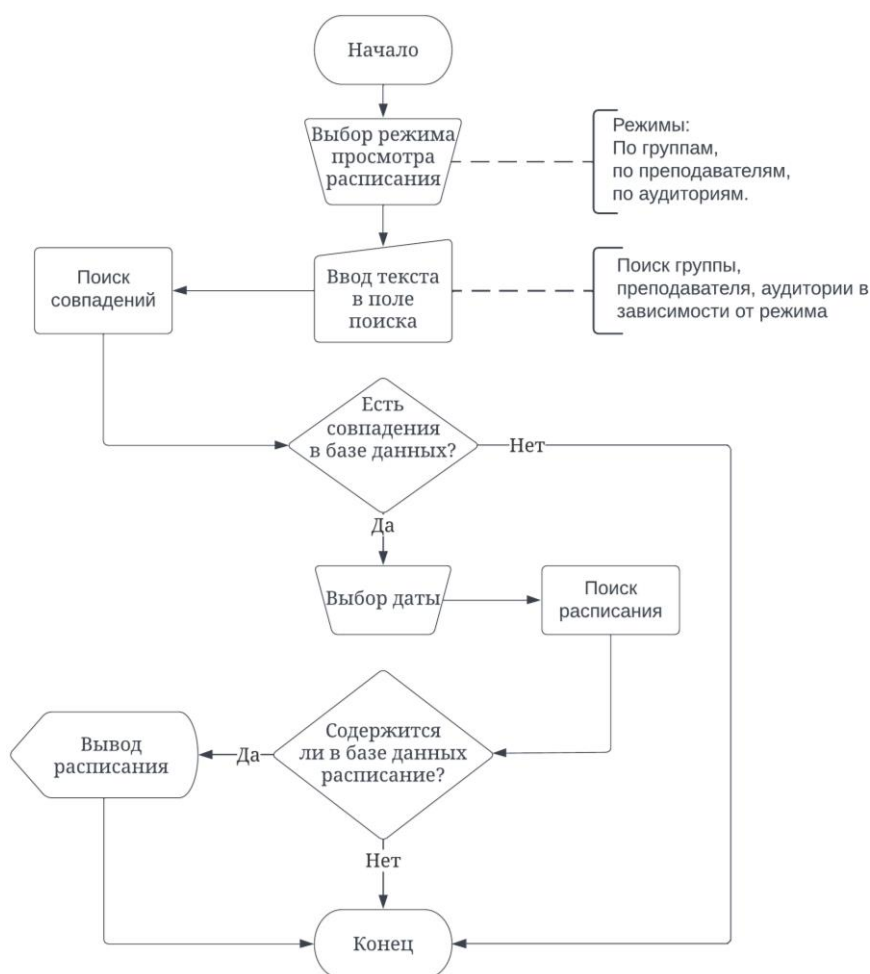


Рисунок 4 - Блок-схема алгоритма просмотра расписания

Пользователь:

- выбирает режим просмотра расписания, присутствует три режима - просмотр по группам/аудиториям/преподавателям;
- после выбора режима, пользователь вводит текст для поиска необходимой(-ого) группы/аудитории/преподавателя;
- система автоматически проводит выборку из базы данных по тексту;
- если выборка дала какой-либо результат - система выводит пользователю совпадения, в противном случае - система выводит пользователю сообщение «Ничего не найдено»;
- пользователь выбирает из списка совпадений необходимую(-ого) группу/аудиторию/преподавателя и дату;
- система выполняет поиск расписания, затем, в случае если выборка дала результаты, система выводит найденное расписание пользователю, в противном случае, ничего не выводится.

На рисунке 5 представлена блок-схема алгоритма составления учебного расписания [7].

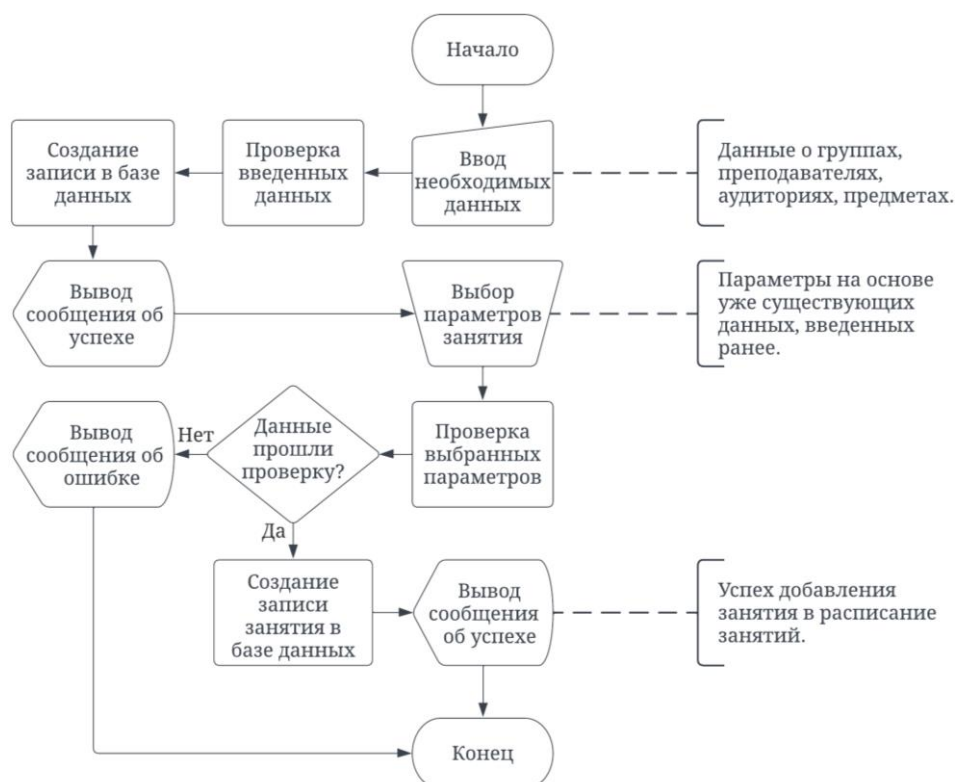


Рисунок 5 - Блок-схема алгоритма составления учебного расписания

Администратор:

- проводит заполнение системы, вводя названия/ФИО необходимых групп/аудиторий/преподавателей, в случае если эти данные еще не были введены в систему;
- после заполнения системы нужными данными, администратор составляет расписания, выбирая из списка необходимые параметры;
- система последовательно и автоматически проверяет корректность выбранных параметров, если при проверке была выявлена некорректность, то система сообщает об этом и администратор должен изменить набор параметров для составления расписания;
- после успешного процесса валидации данных, система вносит занятие в базу данных и сообщает пользователю об успешной операции.

2.1.2.2 Разработка диаграммы вариантов использования

Исходя из составленных блок-схем алгоритма можно выделить возможности использования веб-приложения «Advanced Schedule» различными типами пользователей. Для этого будет применена диаграмма вариантов использования.

Диаграмма вариантов использования - диаграмма, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей программного продукта [32].

На диаграммах вариантов использования отображается взаимодействие между вариантами использования, представляющими функции системы, и действующими лицами, представляющими людей или системы, получающие или передающие информацию в данную систему. Из диаграмм вариантов использования можно получить довольно много информации о системе. Этот тип диаграмм описывает общую функциональность системы.

На рисунке 6 представлена диаграмма вариантов использования веб-приложения «Advanced Schedule» [7].

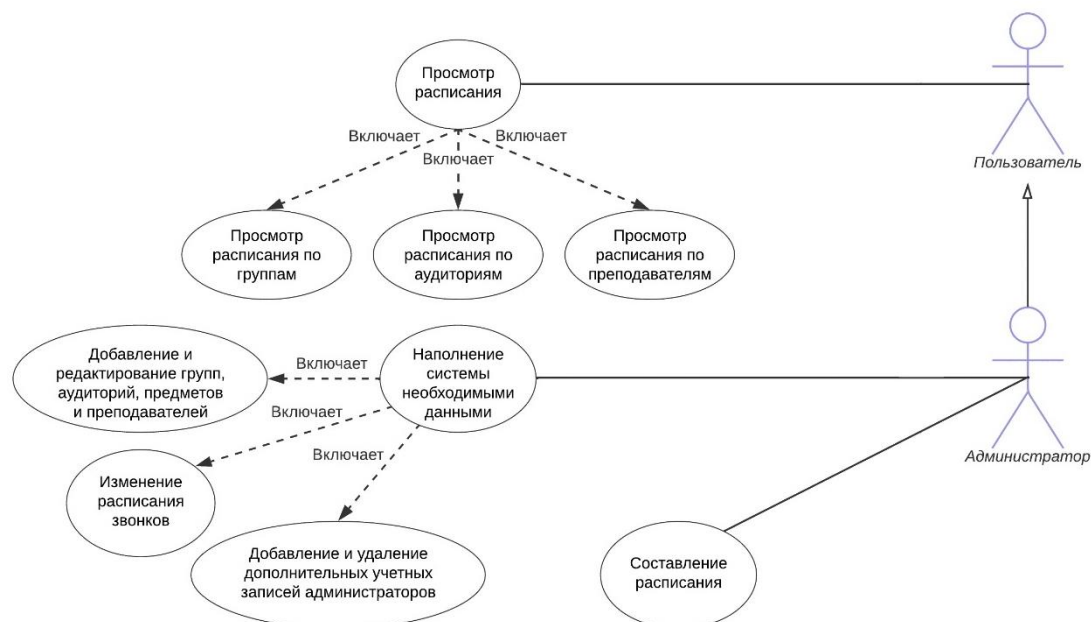


Рисунок 6 - Диаграмма вариантов использования веб-приложения «Advanced Schedule»

Обычный пользователь будет иметь возможность:

- просмотра расписания по группам;
- просмотра расписания по аудиториям;
- просмотра расписания по преподавателям.

Администратор будет иметь расширенные возможности:

- добавления и редактирования групп, аудиторий, предметов, списка преподавателей;
- редактирования расписания звонков;
- манипуляции учетными записями администраторов;
- составления расписания.

2.1.2.3 Разработка схемы базы данных

База данных, являющаяся одной из важнейших частей информационной системы, конечно же, представляет собой сложный объект, который также подлежит проектированию [18].

В процессе создания информационной системы проектирование базы данных имеет очень важную роль, так как база данных является фундаментом информационной системы. Проектирование базы данных выполняется после

анализа требований к будущей системе. А уже после того, как выработана общая схема базы данных, происходит процесс определения архитектуры будущей информационной системы. Так, решаются вопросы о том, какой будет база данных, производится соответствующая декомпозиция и другие необходимые работы [34].

Проектирование базы данных играет огромную роль в создании будущей информационной системы, являясь её фундаментом. На рисунке 7 представлена схема спроектированной базы данных.

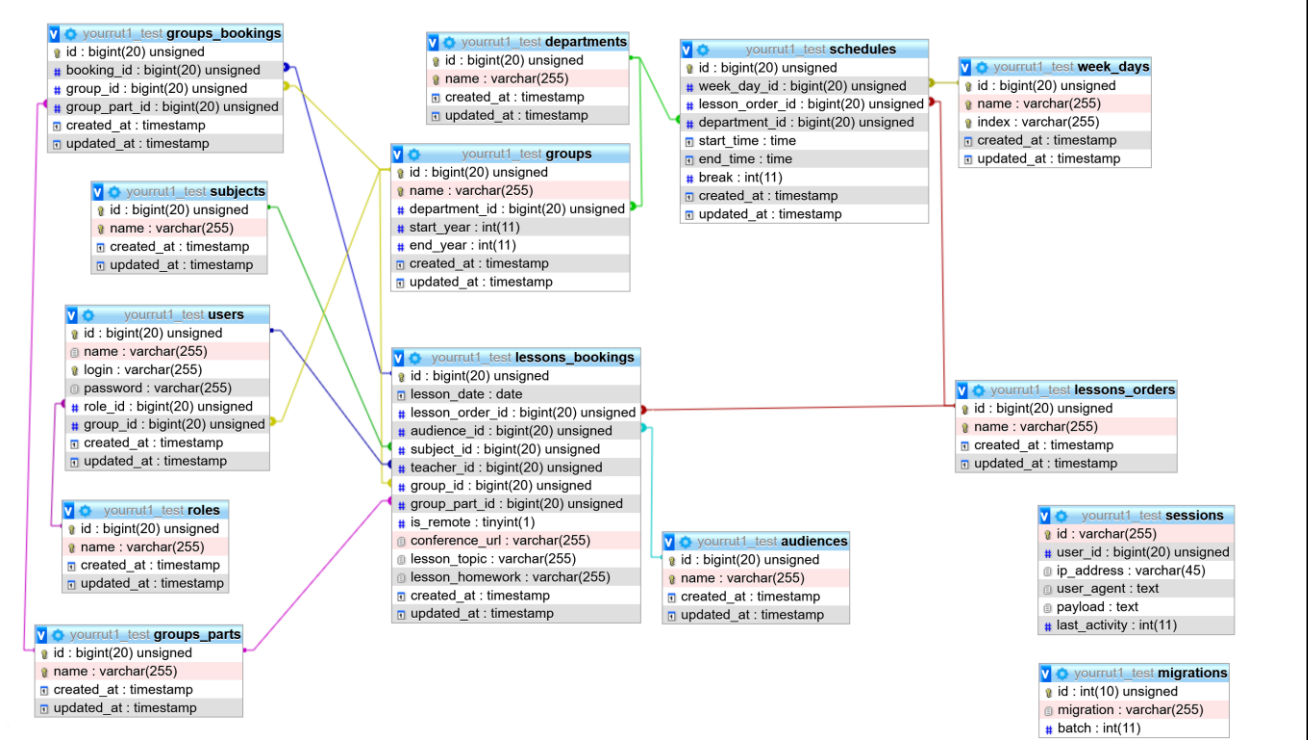


Рисунок 7 - Схема базы данных

Исходя из схемы спроектированной базы выделены данные, которые будут структурированно храниться:

- информация об учебных группах;
- информация о предметах, отделениях и аудиториях;
- информация о пользователях и ролях пользователей, предоставляющих различные полномочия;
- информация о расписании занятий и расписании звонков.

Благодаря связям достигается производительность базы данных и автоматизация процессов хранения информации [15].

2.1.3 Обоснование выбора языков программирования

В качестве языка программирования для разработки веб-приложения «Advanced Schedule» были выбраны языки PHP и JavaScript.

PHP - C-подобный 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 [13].

Основная область применения - разработка скриптов, которые работают на стороне сервера. Также можно создавать скрипты командной строки и приложения с графическим интерфейсом.

PHP - язык с динамической типизацией. Это означает, что переменная не определяется жестко и заранее. Динамическая типизация удобная и гибкая, но приводит к потреблению большого количества оперативной памяти и уменьшает скорость работы [28].

Достоинства:

- язык бесплатный, код находится в открытом доступе;
- язык гибкий и предоставляет много свободы;
- простой синтаксис.

Недостатки:

- для работы с PHP желательно знание HTML и CSS;
- из-за смеси PHP и HTML на многостраничных сайтах может быть затруднительно найти необходимую часть кода;
- в названиях функций стандартной библиотеки отсутствует четкая система: одни имеют сокращения и подчеркивания, другие - нет. Иногда в названиях функций для работы со строками встречаются обозначения str, а иногда их не бывает.

JavaScript - мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили.

Является реализацией спецификации ECMAScript. 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.

Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.

Достоинства:

- наличие полной интеграции с версткой страниц и серверной частью;
- производительность;
- комфортность использования пользовательских интерфейсов.

Недостатки:

- отсутствие возможности чтения и загрузки документов. Основная причина наличия этого минуса – соображения безопасности;
- нестрогая типизация;
- доступность для конкурентов, связано это с высокой читаемостью исходного программного кода JavaScript.

2.1.4 Обоснование выбора инструментальных средств

Для разработки веб-приложения с использованием языка программирования PHP необходимы:

- локальный веб-сервер;
- локальный сервер баз данных;
- среда разработки;
- интерпретатор языка программирования PHP;
- система управления базами данных (СУБД);
- фреймворки для серверной и клиентской части;

Open Server Panel - это портативная программная среда, созданная специально для веб-разработчиков [30].

Для отладки скриптов в различном окружении Open Server предлагает на выбор сразу два вида HTTP серверов, различные версии PHP и СУБД модулей, а так же возможность быстрого переключения между ними.

HTTP модули: Apache и Nginx;

СУБД модули: MySQL, MariaDB, MongoDB, PostgreSQL и Redis;

Достоинства:

- имеет все необходимые модули для разработки и позволяет быстро переключаться между ними;
- программный комплекс портативен;
- программный комплекс бесплатен.

Недостатки:

- занимает большое дисковое пространство, за счет большого пакета включенных модулей;
- в некоторых случаях может некорректно работать и требует перезапуск с использованием прав администратора;
- поддерживается только на операционной системе Windows.

MySQL - свободная реляционная система управления базами данных [28]. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется под лицензией GNU General Public License. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации [34].

Возможности СУБД MySQL:

- корректное распределенное хранение данных на сервере;
- идентификация и обработка отдельных данных, преобразование и отправка данных;
- создание, редактирование и удаление записей, которые есть в базе;

- отправка транзакций - «пакетов» из нескольких запросов к базе;
- возможность контролировать версии базы данных: делать резервные копии, обновлять базу или откатывать назад;
- контроль состояния базы данных.

MySQL имеет клиент-серверную архитектуру. Клиент-серверная архитектура делает хранение данных безопаснее: клиентские компьютеры не могут получить к ним бесконтрольный доступ [30]. Вся информация находится на сервере, а клиенты не перегружены, поэтому им не нужны большие вычислительные мощности.

Достоинства:

- универсальность;
- высокая производительность;
- безопасность;
- популярность.

Недостатки:

- медленное развитие;
- недостаток функций;
- снижение производительности при работе с большими проектами.

PhpStorm - коммерческая кросс-платформенная интегрированная среда разработки для языка программирования PHP. Разрабатывается компанией JetBrains на основе платформы IntelliJ IDEA [13].

PhpStorm представляет собой интеллектуальный редактор для PHP, HTML и JavaScript с возможностями анализа кода, предотвращения ошибок в коде и автоматизированными средствами рефакторинга для PHP и JavaScript.

Основные возможности PhpStorm:

- поддерживает PHP 5.3-8.0, генераторы, сопрограммы и все синтаксические улучшения;
- PHP рефакторинги, детектор дублируемого кода;

- поддержка Docker, Composer, встроенный REST клиент, Command Line Tools, SSH консоль;
- поддержка фреймворков и специализированные плагины для ведущих PHP фреймворков;
- HTML, CSS, JavaScript редактор. Отладка и модульное тестирование для языка JavaScript;
- инструменты работы с базами данных, SQL редактор;
- кросс-платформенность.

Достоинства:

- стабильная работа на разных устройствах;
- поддержка смешивания языков;
- верификация кода;
- полноценная поддержка PHP вместе с базами данных и SQL.

Недостатки:

- использование возможно после приобретения лицензии;
- высокие системные требования;
- сложность в освоении.

Laravel - это бесплатный PHP-фреймворк с открытым исходным кодом, специально разработанный для создания сложных сайтов и веб-приложений. Позволяет упростить аутентификацию, маршрутизацию, сессии, кэширование, архитектуру приложения, работу с базой данных [14].

Laravel пользуются backend-разработчики, которые пишут код на PHP. Он помогает определить структуру веб-приложения и служит для нее каркасом. Фреймворк написан на PHP и расширяет его возможности.

Назначение Laravel - создание веб-приложений и сайтов на основе архитектуры MVC. Это вариант архитектуры, при котором компоненты программы делятся на три части:

- модель предоставляет данные и методы работы с ними: запросы в базу данных, проверка на корректность;

- представление показывает пользователю эти данные и изменяется, если меняется модель;
- контроллер направляет данные от пользователя к системе и наоборот от системы к пользователю.

Возможности фреймворка Laravel:

- консоль Artisan;
- Eloquent ORM;
- шаблонизатор Blade;
- валидация;
- система контроля версий баз данных;
- аутентификация.

Laravel поддерживает noSQL-базы данных. Они отличаются более высокой скоростью работы. Информация хранится в оперативной памяти сервера, поэтому можно быстро получить к ней доступ.

В Laravel встроены механизмы защиты от SQL-инъекций и XSS-атак. SQL-инъекции не дает провести собственная ORM: она не позволяет обрабатывать посторонние SQL-запросы. А от XSS-атак защищает возможность экранировать теги.

Достоинства:

- универсальность;
- высокая производительность;
- обширное сообщество разработчиков.

Недостатки:

- разработчик должен обладать достаточной компетенцией;
- если сравнивать эту технологию с другими, Laravel имеет ограниченную встроенную поддержку. правда, этот недостаток устраняется с помощью благодаря большому сообществу и хорошей документации.

Vue.js - это прогрессивный фреймворк для разработки пользовательских интерфейсов и одностраничных веб-приложений на языке JavaScript [13]. Он

решает задачи уровня представления и упрощает работу с библиотеками. Vue.js можно внедрять постепенно, этим он отличается от других фреймворков на языке JavaScript.

Фреймворк Vue.js применяется при разработке:

- быстрых веб-сайтов и приложений, блогов небольшого размера;
- сайтов с высокой нагрузкой;
- адаптивных интерфейсов;
- разделов личных кабинетов и пользовательских страниц;
- интерфейсов авторизации, онлайн-чатов, форм заявки и других функциональных блоков.

Особенности Vue.js:

- Vue.js - это реактивный MVC-фреймворк. Представление (view) изменяется по мере изменения модели;
- ядро Vue.js идеально подходит для внедрения в существующий проект, сайт готового продукта может продолжать работать, например на jQuery, но часть модулей постепенно будет переписываться на Vue до полноценного перехода;
- фреймворк занимает около 20 кБ, поэтому реализованные на нем проекты быстрее загружаются и лучше ранжируются поисковыми роботами;
- шаблоны, множество документации и инструкций, широкое сообщество энтузиастов позволяют решить любую проблему, возникающую при создании проектов на Vue.js.

Достоинства:

- простота;
- поддерживается серверным фреймворком Laravel.

Недостатки:

- отсутствие полной англоязычной документации;
- Vue.js по-прежнему имеет довольно небольшую долю рынка.

2.2 Практическая часть

2.2.1 Описание процесса разработки веб-приложения

Для разработки веб-приложения «Advanced Schedule» необходимо выполнить следующие работы:

- загрузка и установка интегрированной среды разработки JetBrains PhpStorm для написания программного кода;
- настройка интегрированной среды разработки JetBrains PhpStorm;
- загрузка и установка программы Open Server, реализующей функции для локальной разработки веб-приложений, а именно: веб-сервер, сервер базы данных, интерпретатор языка программирования PHP;
- настройка модулей Open Server, выбор версии языка программирования PHP, выбор веб-сервера, выбор модуля СУБД;
- создание нового проекта;
- настройка необходимых зависимостей и установка программного обеспечения для разработки приложения с помощью терминального пакетного менеджера «composer»;
- конфигурирование проекта и фреймворка Laravel;
- настройка зависимостей и установка фреймворка Vue JS, вспомогательной библиотеки Vuex с помощью пакетных менеджеров «composer» и «npm»;
- разработка программного кода клиентской части и серверной части веб-приложения «Advanced Schedule»;
- тестирование и отладка приложения.

2.2.1.1 Установка программного обеспечения

Для загрузки интегрированной среды разработки JetBrains PhpStorm необходимо перейти на официальный сайт разработчиков, перейти в раздел загрузки и начать скачивание. Важно, что разработчики распространяют свою среду разработки на платной основе. Имеется 30-дневный пробный период,

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		29

чего вполне достаточно для разработки. На рисунке 8 представлен интерфейс страницы загрузки JetBrains PhpStorm.

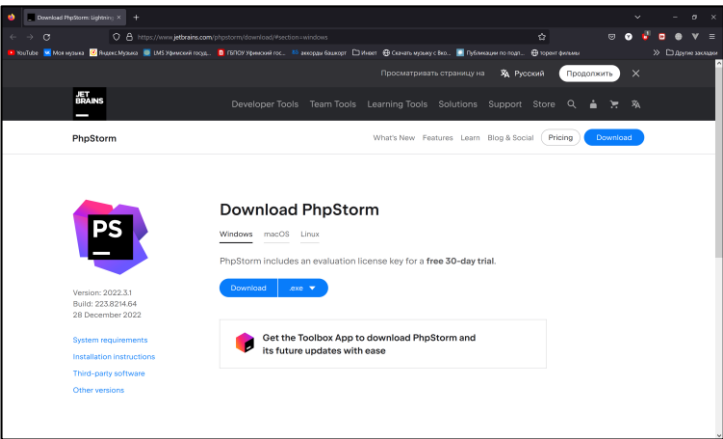


Рисунок 8 - Интерфейс страницы загрузки JetBrains PhpStorm

Необходимо произвести запуск установки интегрированной среды разработки JetBrains PhpStorm. На рисунках 9 и 10 представлены процессы настройки комплектации среды разработки и установки.

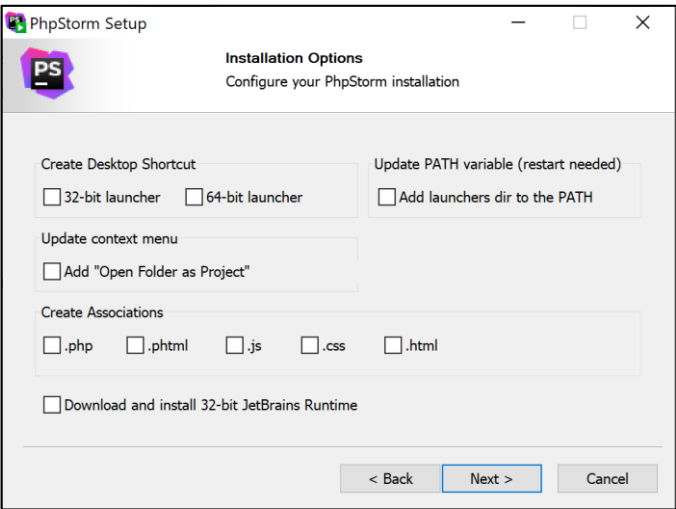


Рисунок 9 - Интерфейс окна настройка комплектации JetBrains PhpStorm

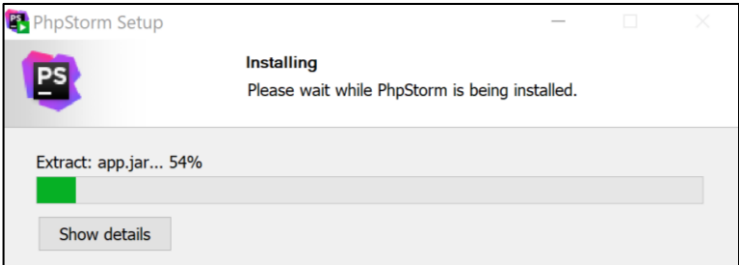


Рисунок 10 - Интерфейс окна установки JetBrains PhpStorm

Для разработки и тестирования веб-приложения требуется интерпретатор языка программирования PHP, веб сервер и сервер базы

данных. На рисунке 11 представлен интерфейс страницы загрузки установочного файла Open Server Panel на официальном сайте разработчиков.

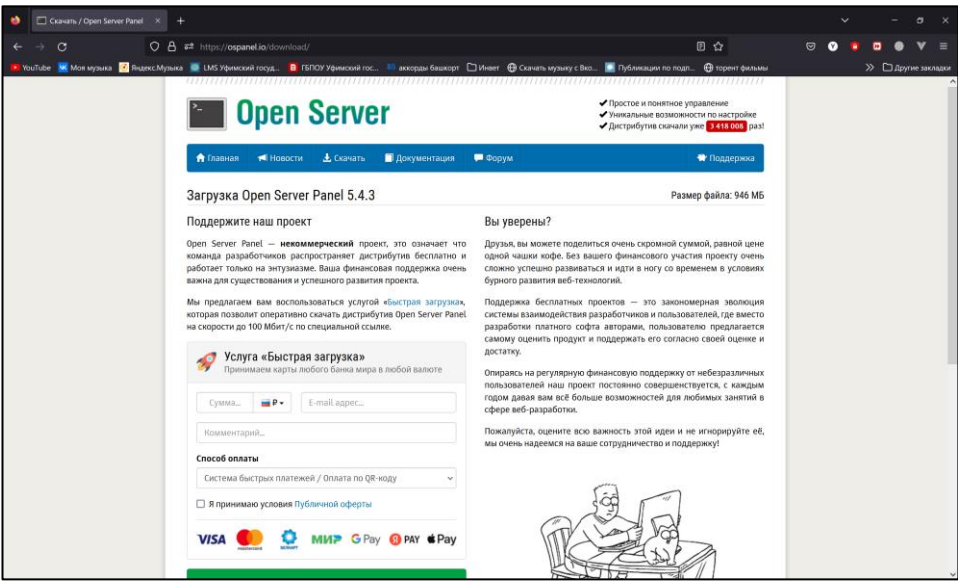


Рисунок 11 - Интерфейс страницы загрузки Open Server Panel

Необходимо произвести конфигурирование программы Open Server Panel и запустить процесс установки. Интерфейс окна конфигурирования Open Server Panel представлен на рисунке 12.

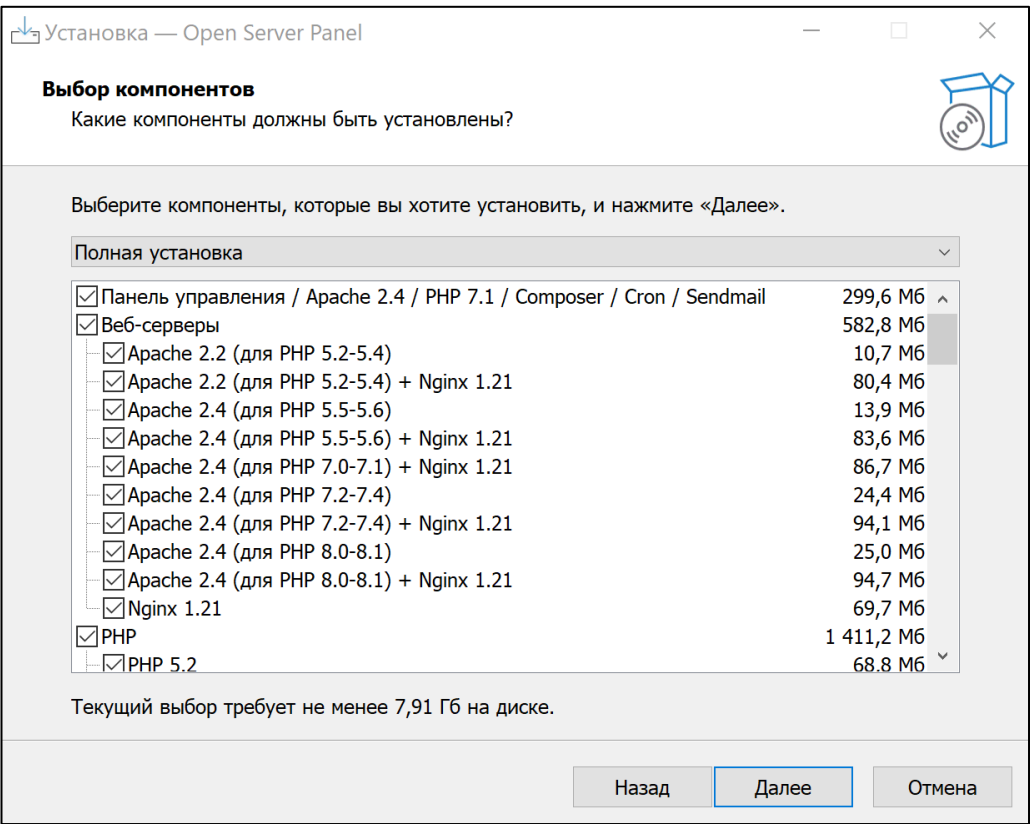


Рисунок 12 - Интерфейс окна конфигурирования Open Server Panel

На рисунке 13 представлен интерфейс окна процесса установки Open Server Panel.

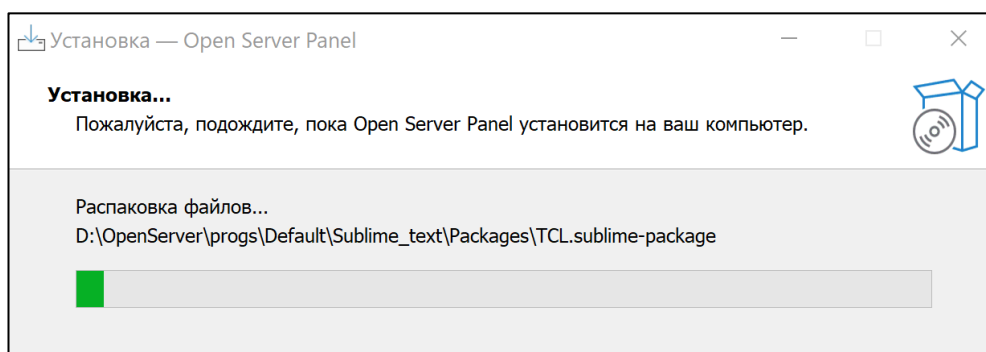


Рисунок 13 - Интерфейс окна процесса установки Open Server Panel

Требуется настройка требуемых версий модулей для разработки веб-приложения «Advanced Schedule», а именно:

- веб-сервер Apache версии 2.4;
- интерпретатор языка PHP версии 8.0;
- сервер реляционной базы данных MariaDB версии 10.5.

На рисунке 14 представлен интерфейс окна настройки Open Server.

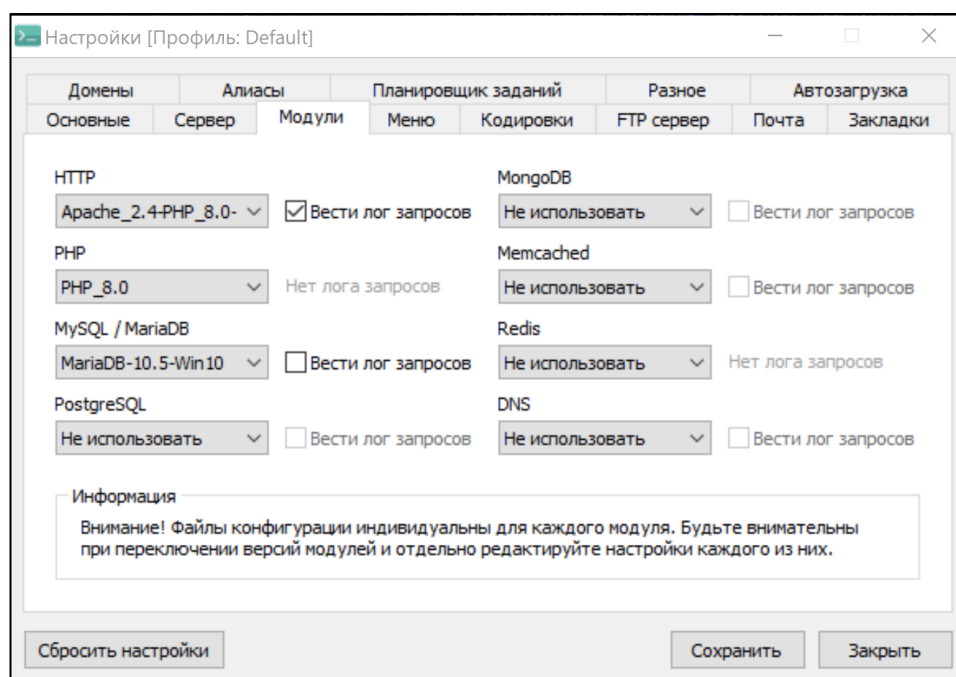


Рисунок 14 - Интерфейс окна настройки Open Server

Необходимо создать проект в директории «domains» программы Open Server с помощью пакетного менеджера Composer. На рисунке 15 представлен интерфейс окна терминала во время процесса создания нового проекта.

Рисунок 15 - Интерфейс окна терминала во время процесса создания нового проекта

Далее необходимо настроить конфигурацию проекта для дальнейшей работы. Необходимо создать файл «.htaccess» в корне проекта, и ввести в нем необходимый код, на рисунке 16 представлен интерфейс окна файла маршрутизации «.htaccess».

Рисунок 16 - Интерфейс окна файла маршрутизации «.htaccess»

2.2.1.2 Настройка подключения к базе данных

Для настройки подключения к базе данных отредактировать файл «database.php» в директории «config», данный файл определяет подключение к базе данных, в нем необходимо ввести информацию для доступа и аутентификации в базе данных, программный код файла представлен на рисунке 17.


```
'connections' => [

    'sqlite' => [
        'driver' => 'sqlite',
        'url' => env( key: 'DATABASE_URL'),
        'database' => env( key: 'DB_DATABASE', database_path( path: 'database.sqlite')),
        'prefix' => '',
        'foreign_key_constraints' => env( key: 'DB_FOREIGN_KEYS', default: true),
    ],

    'mysql' => [
        'driver' => 'mysql',
        'url' => env( key: 'DATABASE_URL'),
        'host' => env( key: 'DB_HOST', default: '127.0.0.1'),
        'port' => env( key: 'DB_PORT', default: '3306'),
        'database' => env( key: 'DB_DATABASE', default: 'rasp'),
        'username' => env( key: 'DB_USERNAME', default: 'root'),
        'password' => env( key: 'DB_PASSWORD', default: ''),
        'unix_socket' => env( key: 'DB_SOCKET', default: ''),
        'charset' => 'utf8mb4',
        'collation' => 'utf8mb4_unicode_ci',
        'prefix' => '',
        'prefix_indexes' => true,
        'strict' => true,
        'engine' => null,
        'options' => extension_loaded( extension: 'pdo_mysql') ? array_filter([
            PDO::MYSQL_ATTR_SSL_CA => env( key: 'MYSQL_ATTR_SSL_CA'),
        ]) : [],
    ],

    'pgsql' => [
        'driver' => 'pgsql',
        'url' => env( key: 'DATABASE_URL'),
```

Рисунок 17 - Программный код файла «database.php»

Первым этапом в разработке программного кода веб-приложения является составление структуры базы данных. Во фреймворке Laravel это осуществляется с помощью инструмента миграций. В файлах миграций задаётся структура таблиц и их связи. На рисунке 18 представлен программный код файла миграции.

```
class LessonsBookings extends Migration
{
    /** Run the migrations. ...*/
    public function up()
    {
        Schema::create( table: 'lessons_bookings', function (Blueprint $table) {
            $table->id();
            $table->date( column: 'lesson_date');
            $table->unsignedBigInteger( column: 'lesson_order_id');
            $table->foreign( columns: 'lesson_order_id')->references( columns: 'id')->on( table: 'lessons_orders')->onUpdate( action: 'CASCADE');
            $table->unsignedBigInteger( column: 'audience_id');
            $table->foreign( columns: 'audience_id')->references( columns: 'id')->on( table: 'audiences')->onUpdate( action: 'CASCADE');
            $table->unsignedBigInteger( column: 'subject_id');
            $table->foreign( columns: 'subject_id')->references( columns: 'id')->on( table: 'subjects')->onUpdate( action: 'CASCADE');
            $table->unsignedBigInteger( column: 'teacher_id');
            $table->foreign( columns: 'teacher_id')->references( columns: 'id')->on( table: 'users')->onUpdate( action: 'CASCADE');
            $table->unsignedBigInteger( column: 'group_id');
            $table->foreign( columns: 'group_id')->references( columns: 'id')->on( table: 'groups')->onUpdate( action: 'CASCADE');
            $table->unsignedBigInteger( column: 'group_part_id');
            $table->foreign( columns: 'group_part_id')->references( columns: 'id')->on( table: 'groups_parts')->onUpdate( action: 'CASCADE');
            $table->boolean( column: 'is_remote')->default( value: 0);
            $table->string( column: 'conference_url')->nullable();
            $table->string( column: 'lesson_topic')->nullable();
            $table->string( column: 'lesson_homework')->nullable();
            $table->timestamps();
        });
    }
}
```

Рисунок 18 - Программный код файла миграции

2.2.1.3 Разработка серверной части

После того, как структура базы данных задана, необходимо описать модели для работы с каждой таблицей, определить в них необходимые функции, которые позволят взаимодействовать с данными. На рисунке 19 представлен программный код модели.

```
class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable;

    /** The attributes that are mass assignable. ...*/
    protected $fillable = [...];

    /** The attributes that should be hidden for serialization. ...*/
    protected $hidden = [...];

    protected $table = 'users';

    public function isAdmin(): bool
    {
        return $this->role_id === 1;
    }

    public function isTeacher(): bool
    {
        return $this->role_id === 2;
    }

    public function isStarosta(): bool
    {
        return $this->role_id === 3;
    }
}
```

Рисунок 19 - Программный код модели

Возможности моделей:

- в случае, если какие-либо данные, которые используются системой, необходимо скрыть, то можно указать защищенное свойство «\$hidden» и передать в него массив, состоящий из строк с названиями колонок таблицы, которые необходимо скрыть;
- для автоматического удаления устаревших данных имеется специальный метод «prunable()», если его определить и указать, при каком условии данные необходимо удалить, то система будет автоматически удалять по данным условиям записи связанные с данной моделью.

Для того, чтобы сервер принимал и отвечал на запросы необходимы контроллеры и классы-валидаторы, наследующиеся от класса «form request». Чтобы соответствовать принципу единственной ответственности из набора

принципов SOLID, необходимо декомпозировать логику обработки данных в классы, отвечающие за одну конкретную задачу.

Структура класса-валидатора:

- метод «rules()» отвечает за правила валидации, он возвращает набор с правилами валидации для каждого параметра в запросе;
- метод «authorize()» отвечает за допуск к выполнению данного запроса, в нем описывается проверка пользователя на право доступа к выполнению данного действия;

метод «failedValidation()» позволяет указывать ответ, который будет возвращаться, если валидация не пройдена.

На рисунках 20-21 представлен программный код классов-валидаторов.

```
class UserRegisterRequest extends AdminRequest
{
    /** Get the validation rules that apply to the request. ...*/
    public function rules()
    {
        return [
            'name' => 'required|string|min:3',
            'login' => 'required|string|min:6|unique:users',
            'password' => ['required', 'string', Password::min( size: 6)->letters()->numbers()],
            'role_id' => 'required|integer|exists:roles,id',
            'group_id' => 'required_if:role_id,==,3|exists:groups,id|integer'
        ];
    }
}
```

Рисунок 20 - Программный код класса-валидатора полей

```
class AdminRequest extends MyRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        if (!Auth::check()) {
            return false;
        }
        return Auth::user()->isAdmin();
    }
}
```

Рисунок 21 - Программный код класса-валидатора авторизации

В случае если валидация прошла, то класс валидатор не выбрасывает исключение, соответственно программный код продолжает выполняться дальше в контроллере. Декомпозиция делает программный код более человеко-читаемым, на рисунке 22 представлен программный код декомпозированного метода контроллера.

```
class UserController extends Controller
{
    public function register(UserRegisterRequest $request): JsonResponse
    {
        $request = $request->validated();

        if ($request['role_id'] != 3 && !empty($request['group_id'])) {
            return $this->sendResp( title: 'Ошибка', text: 'Привязка группы возможна только к пользователям с ролью "Староста"', code: 422);
        }

        User::create([
            'name' => $request['name'],
            'login' => $request['login'],
            'password' => Hash::make($request['password']),
            'role_id' => $request['role_id'],
            'group_id' => $request['group_id'] ?? null
        ]);

        return $this->sendResp( title: 'Успешно', text: 'Пользователь был успешно создан', code: 200);
    }
}
```

Рисунок 22 - Программный код декомпозированного метода контроллера

Для обратного вывода информации от сервера к клиенту необходимо выполнить написание программного кода представлений. Данные файлы отвечают за создание начальных HTML разметок, которые в ходе работы будут использованы на клиентской части фреймворком Vue и преобразованы. На рисунке 23 изображен программный код представления.

```
<!doctype html>
<html lang="ru">
<head>
    <meta charset="UTF-8"/>
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge"/>
    <title>@yield('title')</title>
    <link rel="stylesheet" href="{{ asset('css/app.css') }}" />
    <link rel="icon" href="{{ asset('favicon.ico') }}" />
</head>
<body>
    @yield('content')
    <script src="{{ asset('js/app.js') }}"></script>
    <script>
        var time = new Date().getTime();
        $(document.body).bind("mousemove keypress", function(e) {
            time = new Date().getTime();
        });

        function refresh() {
            if(new Date().getTime() - time >= 31 * 60 * 1000)
                window.location.reload();
            else
                setTimeout(refresh, 10 * 1000);
        }

        setTimeout(refresh, 10 * 1000);
    </script>
</body>
</html>
```

Рисунок 23 - Программный код представления

Имеется необходимость при получении запроса конвертировать время в необходимый формат, чтобы это сделать нужно разработать «middleware». «Middleware» является дополнительным слоем между клиентом и сервером, который может выполнять преобразования или действия проверки перед тем, как отдать данные контроллеру. На рисунке 24 представлен программный код конвертора времени.

```
class TimeConvert
{
    /**
     * Handle an incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @param \Closure(\Illuminate\Http\Request): (\Illuminate\Http\Response|\Illuminate\Http\RedirectResponse) $next
     * @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse
     */
    public function handle(Request $request, Closure $next)
    {
        $input = $request->all();
        if (isset($input['start_time']) && isset($input['end_time'])) {
            $start_time = explode( separator: ':', $input['start_time']);
            $end_time = explode( separator: ':', $input['end_time']);

            $input['start_time'] = [
                'hours' => $start_time[0],
                'minutes' => $start_time[1]
            ];
            $input['end_time'] = [
                'hours' => $end_time[0],
                'minutes' => $end_time[1]
            ];

            $request->replace($input);
        }

        return $next($request);
    }
}
```

Рисунок 24 - Программный код конвертора времени

Для того, чтобы запросы проходили через программный код и попадали в нужные методы и в нужные контроллеры, необходимо определить маршруты запросов по программному коду. За маршруты отвечают специальные файлы для маршрутизации. Фреймворк Laravel предоставляет несколько маршрутизаторов для различных целей. В случае веб-приложения будет использован соответствующий маршрутизатор «web». На рисунке 25 представлен программный код маршрутизатора.

```

Route::get( uri: '/admin', function () {
    return view( view: 'layouts.admin');
})->middleware(['isAuth', 'isAdmin']);
Route::get( uri: '/', function () {
    return view( view: 'layouts.index');
});
Route::get( uri: '/login', function () {
    if (\Illuminate\Support\Facades\Auth::check()) {
        return redirect( to: '/');
    }
    return view( view: 'layouts.login');
});
Route::post( uri: '/login', [\App\Http\Controllers\UserController::class, 'login']);
Route::get( uri: '/logout', [\App\Http\Controllers\UserController::class, 'logout'])->middleware(['isAuth', 'isAdmin']);
Route::get( uri: '/search', [\App\Http\Controllers\MainReadController::class, 'search']);
Route::prefix( prefix: 'api')->group(function () {

    Route::get( uri: '/lessons', [\App\Http\Controllers\MainReadController::class, 'getLessons']);
    Route::get( uri: '/Group', [\App\Http\Controllers\MainReadController::class, 'getAllGroups']);
    Route::get( uri: '/Department', [\App\Http\Controllers\MainReadController::class, 'getAllDepartments']);
    Route::get( uri: '/Audience', [\App\Http\Controllers\MainReadController::class, 'getAllAudiences']);
    Route::get( uri: '/Week', [\App\Http\Controllers\MainReadController::class, 'getAllWeeks']);
    Route::get( uri: '/LessonsOrder', [\App\Http\Controllers\MainReadController::class, 'getAllLessonOrders']);
    Route::get( uri: '/groupsPart', [\App\Http\Controllers\MainReadController::class, 'getAllGroupParts']);
    Route::get( uri: '/Subject', [\App\Http\Controllers\MainReadController::class, 'getAllSubject']);
    Route::get( uri: '/Prepod', [\App\Http\Controllers\MainReadController::class, 'getAllPrepods']);
    Route::get( uri: '/Captain', [\App\Http\Controllers\MainReadController::class, 'getCaptains']);
    Route::get( uri: '/Schedule', [\App\Http\Controllers\MainReadController::class, 'getSchedules']);

    Route::get( uri: '/lessonBooking', [\App\Http\Controllers\MainReadController::class, 'getBookings']);
    Route::get( uri: '/lessonBooking/group', [\App\Http\Controllers\MainReadController::class, 'getGroupBooking']);
    Route::get( uri: '/lessonBooking/teacher', [\App\Http\Controllers\MainReadController::class, 'getTeacherBooking']);
    Route::get( uri: '/lessonBooking/audience', [\App\Http\Controllers\MainReadController::class, 'getAudienceBooking']);
});

```

Рисунок 25 - Программный код маршрутизатора

Аутентификация – это процесс проверки подлинности логина и пароля зарегистрированного пользователя, в данном случае администратора, путём сравнения введенных и зашифрованных данных в базе данных. После успешной аутентификации в сессии пользователя указывает, что данный пользователь аутентифицирован, и выдаются соответствующие права на доступ к другим частям веб-приложения.

Для реализации аутентификации необходимо воспользоваться вспомогательными классами в фреймворке Laravel – «Hash» и «Auth».

«Hash» выполняет сравнение введенных и зашифрованных данных. «Auth» выполняет процесс аутентификации пользователя в сессии.

На рисунке 26 представлен программный код метода аутентификации контроллера пользователей.

```

public function login(Request $request)
{
    $request = $request->all();
    $val = Validator::make($request, [
        'login' => 'required|exists:users,login',
        'password' => 'required'
    ]);

    if ($val->fails()) {
        return back()->with('errorTitle', 'Ошибка')->with('errorText', 'Неверный логин или пароль');
    }

    $user = User::where('login', $request['login'])->first();

    if (!Hash::check($request['password'], $user->password)) {
        return back()->with('errorTitle', 'Ошибка')->with('errorText', 'Неверный логин или пароль');
    }

    if ($user->role_id == 1) {
        Auth::login($user);
        $cookie = Cookie::make( name: 'isAdmin', value: 'true', minutes: 60, path: null, domain: null, secure: null, httpOnly: false);
        return redirect( to: '/admin' )->withCookie($cookie);
    }

    return redirect( to: '/' );
}

```

Рисунок 26 - Программный код метода аутентификации контроллера пользователей

2.2.1.4 Разработка клиентской части

После окончания разработки серверной части веб-приложения необходимо выполнить разработка клиентской части веб-приложения «Advanced Schedule», используя фреймворк «Vue», библиотеку CSS стилей «Bootstrap», библиотеку JS «jQuery».

Перед тем, как приступить к установке вышеперечисленных библиотек требуется установка программы «Node JS» и пакетного менеджера «npm». На рисунке 27 представлен интерфейс страницы загрузки установочного файла программы «Node JS».

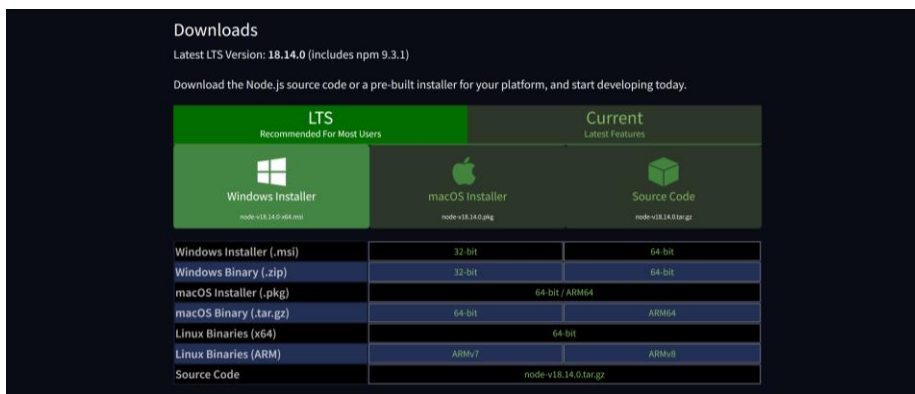


Рисунок 27 - Интерфейс страницы загрузки установочного файла программы «Node JS»

Необходимо запустить процесс установки программы «Node JS» и пакетного менеджера «npm», интерфейс окна установки представлен на рисунке 28.

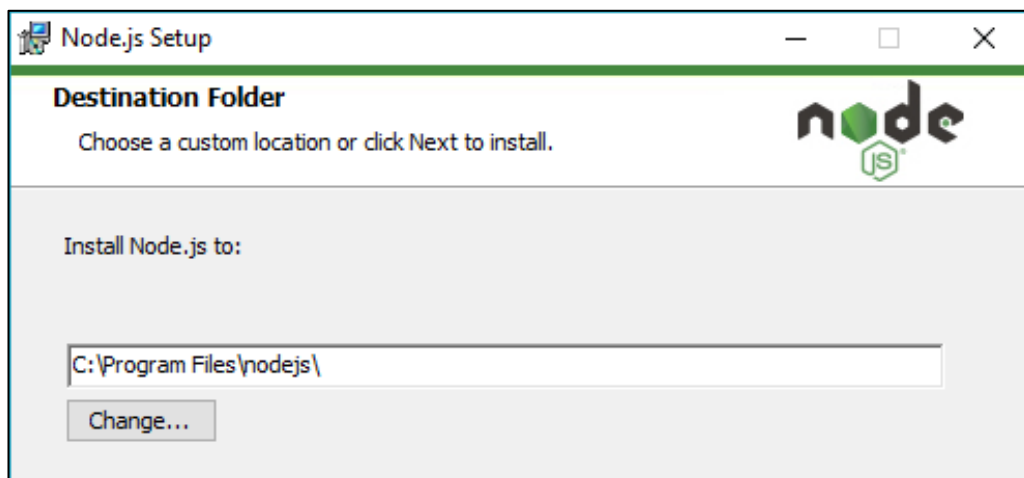


Рисунок 28 - Интерфейс окна установки «Node JS» и «npm»

Для установки набора библиотек для разработки необходимо ввести команды представленные на рисунках 29-31.



Рисунок 29 - Команды для установки фреймворка «Vue»

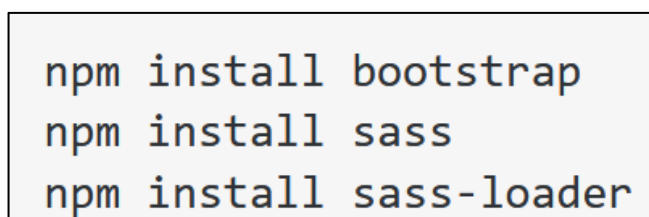


Рисунок 30 - Команды для установки библиотеки стилей «Bootstrap»

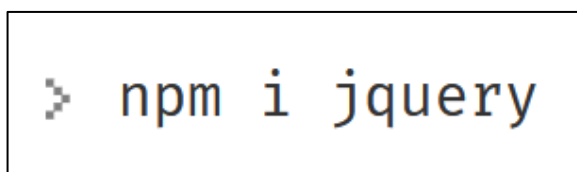


Рисунок 31 - Команда для установки библиотеки «jQuery»

После выполнения загрузки необходимо выполнить подключение в программном коде. На рисунках 32-34 представлен программный код подключения всех библиотек для дальнейшей работы с ними.

```
require('./bootstrap');
window.Vue = require('vue').default;
/** The following block of code may be used to automatically register your ...*/

// const files = require.context('./components', true...

/** Next, we will create a fresh Vue application instance and attach it to ...*/
import {BootstrapVue, IconsPlugin} from "bootstrap-vue";
import vueDebounce from 'vue-debounce';
import store from "./storage/storage";
import router from "./router.js";

Vue.use(vueDebounce, {
  trim: true
});
Vue.use(BootstrapVue);
Vue.use(IconsPlugin);

const app = new Vue({
  el: '#app',
  router,
  store
});
```

Рисунок 32 - Программный код подключения фреймворка «Vue»

```
window._ = require('lodash');
/** We'll load jQuery and the Bootstrap jQuery plugin which provides support ...*/
try {
  window.Popper = require('popper.js').default;
  window.$ = window.jQuery = require('jquery');
  require('bootstrap');
} catch (e) {}

/** We'll load the axios HTTP library which allows us to easily issue requests ...*/
window.axios = require('axios');
window.axios.defaults.headers.common['X-Requested-With'] = 'XMLHttpRequest';
```

Рисунок 33 - Программный код подключения библиотек JavaScript

```
// Fonts
@import url('https://fonts.googleapis.com/css?family=Nunito');
// Variables
@import 'variables';
// Bootstrap
@import '~bootstrap/scss/bootstrap';
@import "~bootstrap-vue/dist/bootstrap-vue.css";
```

Рисунок 34 - Программный код подключения стилей

Клиентская часть состоит из графического интерфейса, с которым можно взаимодействовать. Разработку клиентской части необходимо выполнить в несколько этапов:

- разметка HTML, вёрстка;
- применение CSS стилей для улучшения графического интерфейса;
- разработка взаимодействия с графическим интерфейсом;
- обеспечение работы клиентской части с программным интерфейсом серверной части приложения.

Фреймворк Vue предоставляет компонентный подход к разработке клиентской части приложений. Каждый компонент состоит из HTML, CSS и JS. Данные компоненты возможно переиспользовать и внедрять в другие компоненты, благодаря этому разработка становится эффективней.

На рисунке 35 представлен программный код компонента Vue.

```
<template>
  <div>
    <Header/>
    <Card/>
    <NotificationBootstrap/>
  </div>
</template>

<script>
  export default {
    name: "admin",
    mounted() {
      this.$store.dispatch( type: 'callAllData');
    }
  }
</script>
```

Рисунок 35 - Программный код компонента Vue

Внутри компонентов выполняются первые три этапа разработки клиентской части приложения. Последний этап выполняет модуль фреймворка «Vue» под названием «Vuex». Данный модуль позволяет отделить данные в общий контейнер, который отвечает за оперирование данными и взаимодействием с программным интерфейсом серверной части приложения

посредством сети «Internet». На рисунке 36 представлен программный код модуля «Vuex».

```
import Vue from 'vue';
import Vuex from 'vuex';

Vue.use(Vuex);

const store = new Vuex.Store({ options: {
  actions: {
    switchTab: (context, tabName) => context.commit('switchTab', tabName.target.id),
    switchLessonTab: (context, tabName) => context.commit('switchLessonTab', tabName.target.id),
    callAllData({commit, dispatch}) {
      dispatch('getAllAudiences');
      dispatch('getAllDepartments');
      dispatch('getGroups');
      dispatch('getAllGroupParts');
      dispatch('getAllLessonOrders');
      dispatch('getAllUsers');
      dispatch('getAllWeeks');
      dispatch('getRoles');
      dispatch('getSchedule');
      dispatch('getAllSubjects');
      dispatch('getAllBookings')
    },
    sendRequest({commit, dispatch}, request) {
      const instance = axios.create({
        baseURL: '/api/'
      })
      instance.request({ config: {
        url: request.entity,
        method: request.method ?? 'get',
        data: request.data ?? {},
        params: request.params ?? {}
      }}).then(response => {
        if (Array.isArray(request.todoComm) && request.todoComm.length !== 0) {
          if (request.method && request.method === 'get') {
            request.todoComm.map(item => {
```

Рисунок 36 - Программный код модуля «Vuex»

Впоследствии код необходимо скомпилировать с помощью «сборщика» в один общий файл, используя скрипты пакетного менеджера npm. На рисунке 37 представлен программный код файла настройки скриптов npm.

```
{
  "private": true,
  "scripts": {
    "dev": "npm run development",
    "development": "mix",
    "watch": "mix watch",
    "watch-poll": "mix watch -- --watch-options-poll=1000",
    "hot": "mix watch --hot",
    "prod": "npm run production",
    "production": "mix --production"
  },
```

Рисунок 37 - Интерфейс окна файла настройки скриптов npm

На рисунке 38 представлен интерфейс окна результата сборки кода.

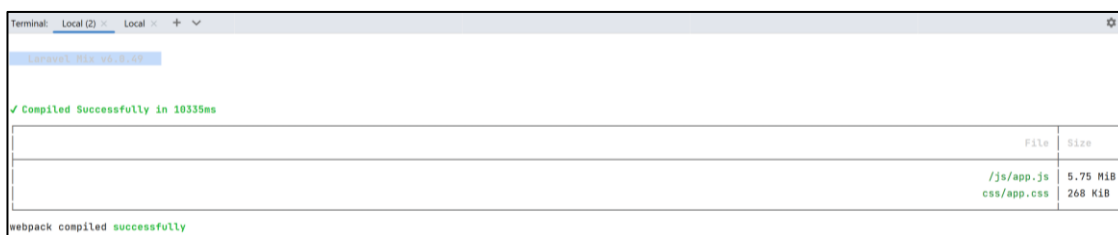


Рисунок 38 - Интерфейс окна результата сборки кода

2.2.2 Интерфейс разработанного веб-приложения

Графический интерфейс - система средств для взаимодействия пользователя с электронными устройствами, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана.

Графический интерфейс пользователя является обязательным компонентом большинства современных программных продуктов, ориентированных на работу конечного пользователя. Основными достоинствами графического интерфейса являются наглядность и интуитивная понятность для пользователя. На рисунках 39-40 представлены интерфейсы страниц обычного пользователя и администратора.

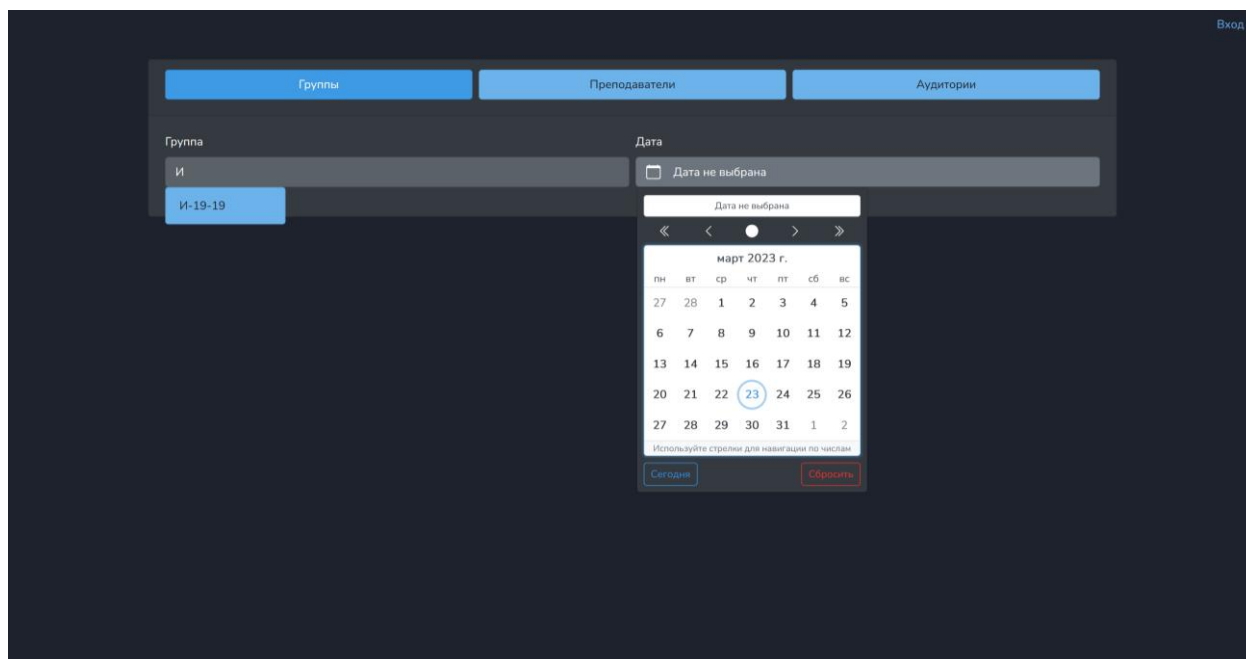


Рисунок 39 - Графический интерфейс страницы обычного пользователя

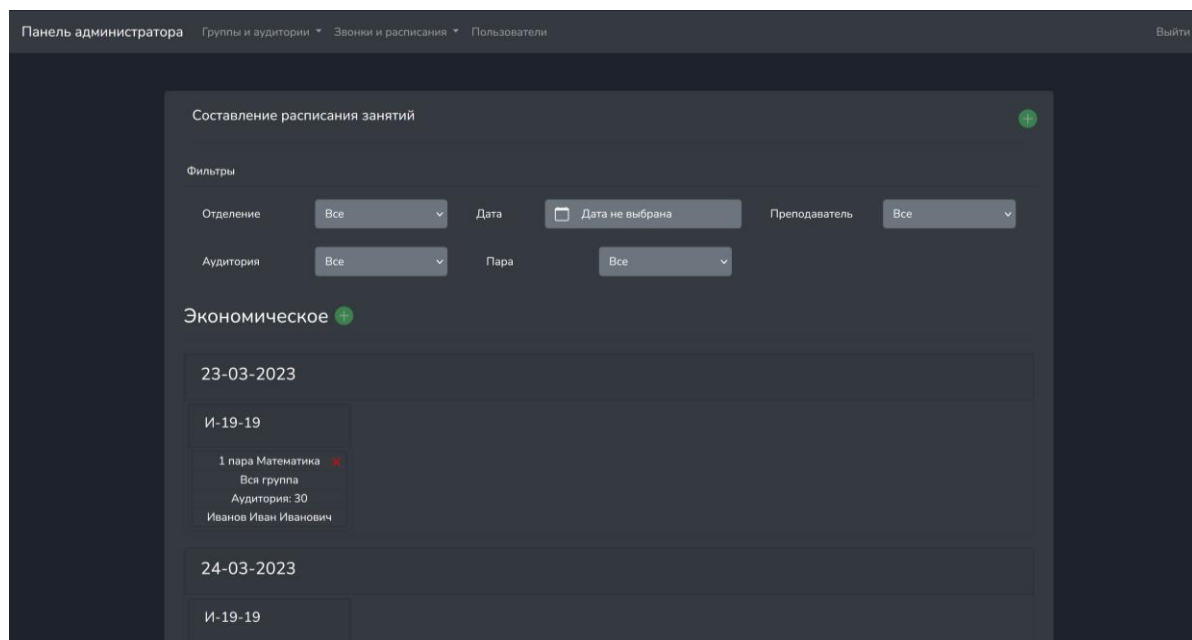


Рисунок 40 - Графический интерфейс страницы администратора

2.2.3 Разработка руководства пользователя

Для того, чтобы начать пользоваться и составлять расписание занятий в веб-приложении «Advanced Schedule» необходимо перейти по доменному имени, к которому будет привязано данное веб-приложение, введя доменное имя в адресную строку любого установленного браузера на текущем устройстве. На рисунке 41 представлен интерфейс адресной строки браузера «Mozilla Firefox» и пример доменного имени развернутого веб-приложения «Advanced Schedule».



Рисунок 41 - Интерфейс адресной строки браузера

Когда переход будет завершен, будет открыта начальная страница, которая предназначена для просмотра расписания занятий. Для просмотра расписания нужно выбрать режим, по которому будет происходить выборка расписания занятий. Доступные режимы выборки:

- поиск по названию группы;
- поиск по названию аудитории;
- поиск по ФИО преподавателя.

После выбора режима будут доступны два поля:

- поле поиска предмета выборки, в соответствии с выбранным режимом выборки;
- поле выбора даты поиска расписания занятий.

Для поиска необходимого предмета выборки необходимо ввести символы, совпадающие с названием/ФИО. Поиск происходит по любой части названия/ФИО. После ввода будет открыт выпадающий список с найденными совпадениями в базе данных, из предложенных вариантов необходимо выбрать нужный для пользователя. Нажав на поле ввода даты будет открыто окно календаря для выбора даты. Когда пользователь выберет нужную дату, поиск будет начат, впоследствии ниже будет выведено расписание занятий, в случае успешного поиска. Напротив, если поиск будет провален, то ниже ничего не будет выведено. На рисунках 42-43 представлены интерфейсы начальной страницы при заполнении полей и вывод расписания занятий.

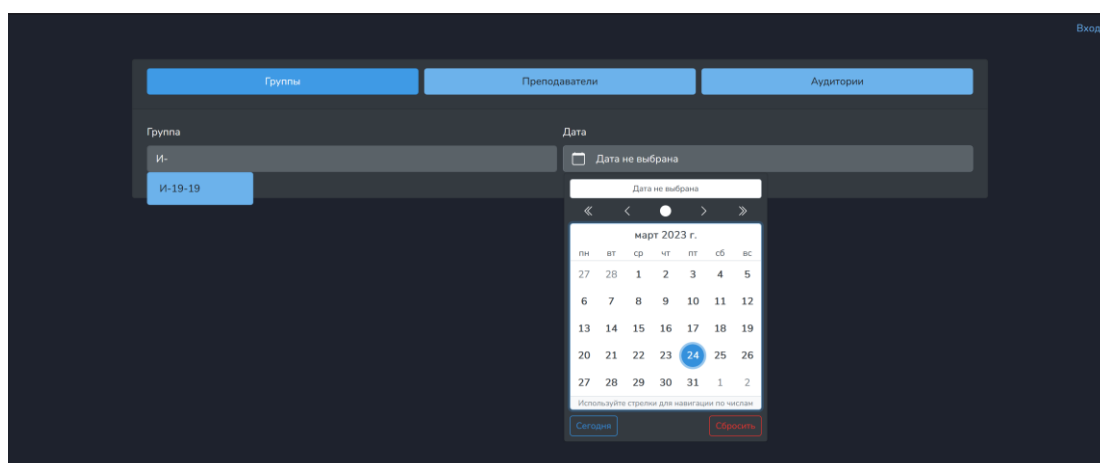


Рисунок 42 - Интерфейс страницы просмотра расписания при заполнении полей поиска расписания занятий

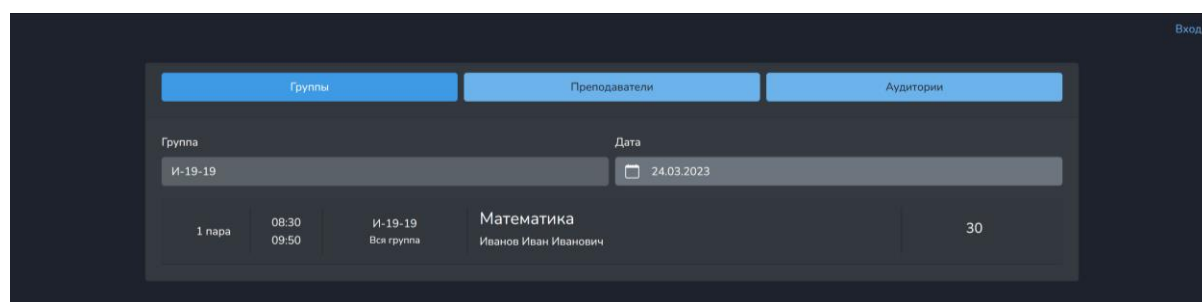


Рисунок 43 - Интерфейс страницы просмотра расписания после вывода найденного расписания

Составление расписания занятий требует авторизации. Для авторизации необходимо нажать «Вход» в правом верхнем углу страницы просмотра расписания занятий. Интерфейс страницы авторизации представлен на рисунке 44.

Рисунок 44 - Интерфейс страницы авторизации

После развертывания по умолчанию присутствует учетная запись администратора со следующими данными для авторизации:

- логин «spregesur»;
- пароль «123456».

Необходимо ввести данные для авторизации в соответствующие поля, затем нажать на кнопку «Войти», после этого произойдет авторизация и переход на страницу для составления расписания. Интерфейс страницы составления расписания представлен на рисунке 45.

Рисунок 45 - Интерфейс страницы администратора

После успешной авторизации рекомендуется переключиться на вкладку «Пользователи», нажав на верхней панели «Пользователи», и создать новую учетную запись администратора в целях безопасности с надежным паролем, комбинируя регистры, буквы, цифры и спецсимволы. Затем необходимо нажать на «Выйти» и авторизоваться под новой учетной записью, а учетную

запись по умолчанию следует удалить. Создание новой учетной записи происходит по нажатию кнопки «Добавить пользователя», в открывшемся модальном окне ввести необходимые данные и выбрать роль «Администратор», затем нажать на кнопку «Создать». На рисунке 46 представлен интерфейс модального окна создания нового пользователя.

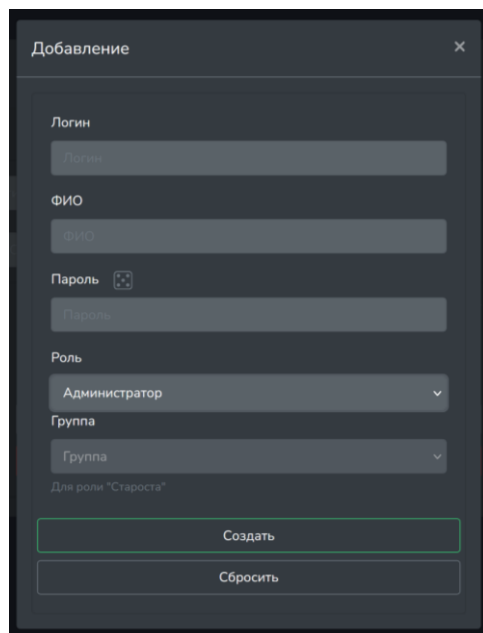


Рисунок 46 - Интерфейс модального окна создания нового пользователя

В случае, если необходимо отредактировать данные пользователя, нужно нажать на необходимого пользователя, откроется модальное окно. Для редактирования доступны ФИО и роль пользователя. После редактирование необходимо нажать кнопку «Сохранить». На рисунке 47 представлен интерфейс модального окна редактирование пользователя.

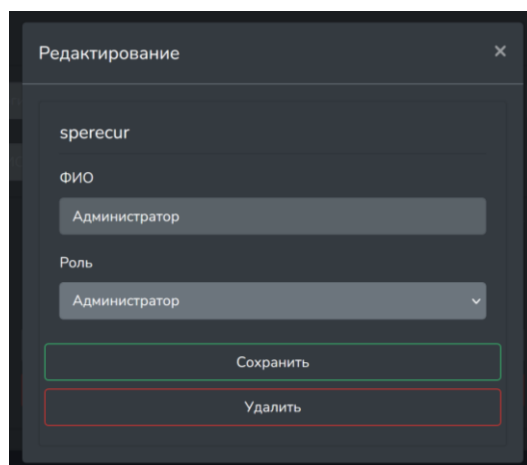


Рисунок 47 - Интерфейс модального окна редактирования пользователя

Для изменения пароля пользователя воспользуйтесь серой кнопкой «Изменить пароль».

Перед составлением расписания необходимо:

- заполнить списки отделений, аудиторий и предметов во вкладке «Основное», а также, в случае необходимости, отредактировать другие виды данных на данной вкладке;
- заполнить список групп во вкладке «Группы»;
- отредактировать расписание звонков во вкладке «Расписание звонков», по умолчанию в системе присутствует расписание звонков.

Для перехода во вкладку «Основное» и редактирования необходимых данных нужно нажать на верхней панели «Группы и аудитории» и выбрать в выпадающем списке «Основное».

Во вкладке «Основное» присутствует два типа форм:

- в левой части форма для редактирования и удаления;
- в правой части форма для создания.

Для редактирования или удаления сущности необходимо выбрать нужную вам сущность нажав на выпадающий список «Выбрать [тип сущности]» и выбрать в нем конкретную сущность, затем для редактирования изменить текст в поле и нажать кнопку «Сохранить», либо для удаления нажать на кнопку «Удалить». На рисунке 48 представлен графический интерфейс вкладки «Основное».

The screenshot displays the 'Основное' (Main) tab interface, which is divided into two main sections: 'Отделения' (Departments) and 'Аудитории' (Auditoriums). Each section contains a form for editing an existing entity and a form for creating a new one.

Отделения (Departments):

- Left Form (Edit/Delete):** Features a dropdown menu 'Выбрать отделение' (Select department) with 'Экономическое' (Economic) selected. Below it is a text input field 'Название' (Name) containing 'Экономическое'. At the bottom are two buttons: 'Сохранить' (Save) and 'Удалить' (Delete).
- Right Form (Create):** Features a text input field 'Название' (Name) and two buttons: 'Создать' (Create) and 'Сбросить' (Reset).

Аудитории (Auditoriums):

- Left Form (Edit/Delete):** Features a dropdown menu 'Выбрать аудиторию' (Select auditorium) with '30' selected. Below it is a text input field 'Название' (Name) containing '30'. At the bottom are two buttons: 'Сохранить' (Save) and 'Удалить' (Delete).
- Right Form (Create):** Features a text input field 'Название' (Name) and two buttons: 'Создать' (Create) and 'Сбросить' (Reset).

Рисунок 48 - Интерфейс вкладки «Основное»

Для перехода во вкладку «Группы» и заполнение списка групп нужно нажать на «Группы и аудитории» и выбрать в выпадающем списке «Группы». На рисунке 49 представлен интерфейс вкладки «Группы».

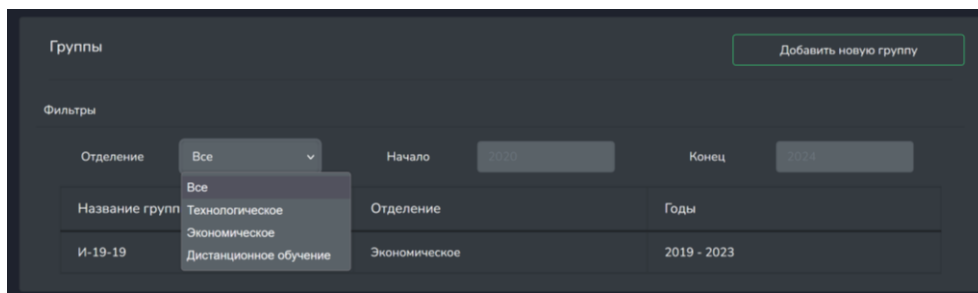


Рисунок 49 - Интерфейс вкладки «Группы»

На вкладке «Группы» необходимо нажать на кнопку «Добавить новую группу» и в открывшемся модальном окне ввести данные группы, затем нажать кнопку «Добавить». Интерфейс модального окна добавления группы аналогичен с модальным окном создания пользователя. Для редактирования данных о группе нажмите на группу в таблице, откроется модальное окно аналогичное с модальным окном редактирования пользователя.

Для перехода во вкладку «Расписание звонков» необходимо нажать на верхней панели «Звонки и расписания» и выбрать в выпадающем списке «Расписание звонков». На рисунке 50 представлен графический интерфейс вкладки «Расписание звонков».

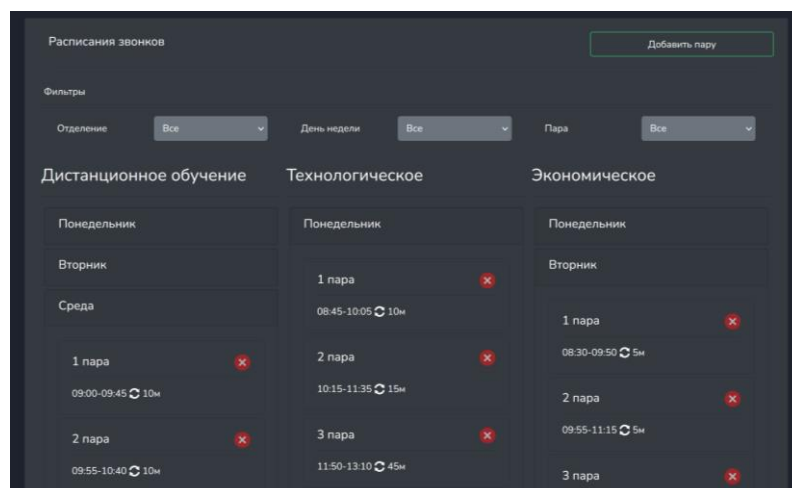


Рисунок 50 - Интерфейс вкладки «Расписание звонков»

Чтобы отредактировать временные отрезки пары нужно нажать на день недели, откроется список пар, а затем нажать на пару, откроется модальное

окно для редактирования данных. Интерфейс модального окна аналогичен интерфейсу модального окна редактирования пользователя. Для добавления пары необходимо нажать на кнопку «Добавить пару». Для удаления пары необходимо нажать на красную кнопку со знаком креста на необходимой паре.

Для составления расписания занятий необходимо перейти во вкладку «Расписание занятий», нажав на верхней панели «Звонки и расписание», затем выбрать в выпадающем списке «Расписание занятий». Интерфейс вкладки «Расписание занятий» представлен на рисунке 51.

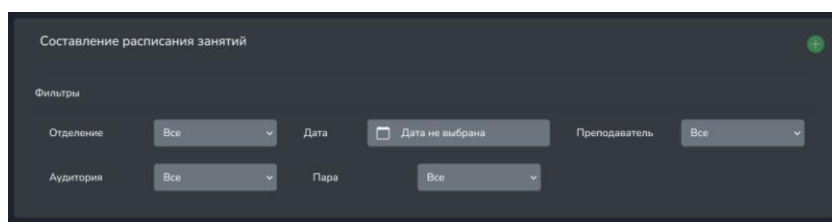


Рисунок 51 - Интерфейс вкладки «Расписание занятий»

Для создания занятия необходимо нажать на зеленую кнопку в правой верхней части вкладки, в открывшемся модальном окне, интерфейс которого представлен на рисунке 52, ввести необходимые данные и нажать на зеленую кнопку «Сохранить».

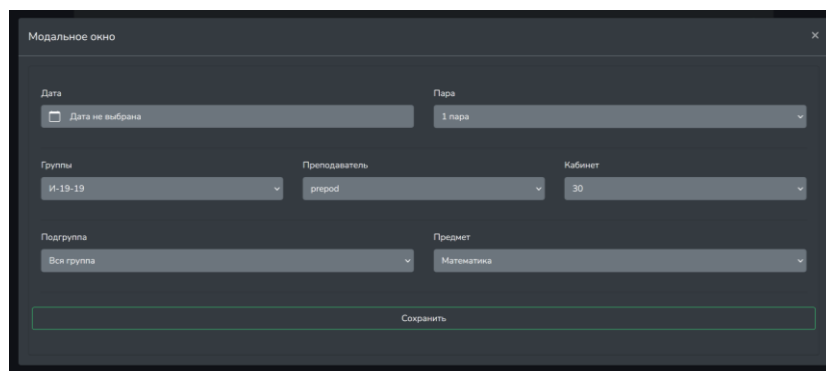


Рисунок 52 - Интерфейс модального окна для создания занятия

Для удаления занятия необходимо нажать на необходимый день и группу, затем нажать на красную кнопку-крестик.

В каждой вкладке, кроме вкладки «Основные» присутствуют фильтры по разным параметрам для более точного вывода необходимых данных. Пользователю рекомендуется использовать фильтры в целях выборки данных по требуемым параметрам [6].

ЗАКЛЮЧЕНИЕ

Веб-приложение - клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются межплатформенными службами.

Целью дипломной работы является разработка информационной системы веб-приложения «Advanced Schedule» для автоматизированного составления расписания и просмотра.

Для разработки веб-приложения «Advanced Schedule» использовались следующие программные средства:

- интегрированная среда разработки PhpStorm;
- программный комплекс Open Server Panel;
- серверный фреймворк Laravel;
- система управления базами данных MySQL;
- клиентский фреймворк Vue и модуль Vuex.

Интегрированная среда разработки PhpStorm была использована для написания программного кода. Open Server Panel использовался для развертывания локальных веб-сервера и системы управления базами данных. Фреймворк Laravel был использован для ускорения процесса разработки серверной части веб-приложения. Система управления базами данных MySQL использовалась для хранения данных, необходимых для работы веб-приложения «Advanced Schedule». Фреймворк Vue использовался для ускорения процесса разработки клиентской части веб-приложения, а модуль Vuex для оперирования данными на клиентской стороне.

В ходе выполнения дипломной работы были продемонстрированы теоретические и практические навыки и умения, полученные в рамках

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		53

обучения. Подводя ее итоги можно выделить следующие наиболее значимые достигнутые результаты:

- исследованы деятельность и организационная структура ГБПОУ Уфимского государственного колледжа технологии и дизайна, на основании чего был сделан вывод о необходимости повышения эффективности работы колледжа в части автоматизации информационного процесса составления расписания;

- проведен анализ существующей организации информационных процессов составления расписания в колледже и выявлены недостатки, устранение которых возможно за счет внедрения автоматизированной системы составления расписания;

- составлена спецификация и проведено обоснование функциональных и нефункциональных требований к разрабатываемой автоматизированной системе составления расписания, учитывающих нормативно-правовую документацию и специфические особенности учебной деятельности ГБПОУ Уфимского государственного колледжа технологии и дизайна;

- разработаны модели программного продукта, включающая в себя: разработку блок-схем алгоритма работы веб-приложения «Advanced Schedule», возможные сценарии взаимодействия с ней пользователя показанных в главе посвященной аналитической части дипломной работы;

- спроектированы база данных и автоматизированная система составления расписания в колледже;

- разработано программное обеспечение, внедрение которого позволяет обеспечить повышение эффективности и снижение трудозатрат персонала ГБПОУ Уфимского государственного колледжа технологии и дизайна посредством автоматизации информационных процессов составления расписания в школе;

– рассмотрены аналоги разрабатываемой информационной системы, были заимствованы ключевые достоинства и устранены основные выявленные недостатки.

При написании данной работы были продемонстрированы результаты разработки программы в виде изображений готового интерфейса программы, структуры ее базы данных, а также фрагментов кода, представленных как примеры.

Разработанная система реализована средствами современных сетевых web-технологий с целью автоматизации организационных процессов и обеспечения возможности одновременного удаленного доступа пользователей к информационным ресурсам. Обучающиеся, преподаватели и операторы составления расписания имеют доступ в любое время и из любого места при наличии интернета к актуальному расписанию занятий. Возможным следующим этапом развития веб-приложения «Advanced Schedule» является добавление расписания экзаменационной сессии. Исходя из технической характеристики проекта, можно сделать вывод, что разработанная ИС является современной и выполняет поставленные задачи без длительных задержек и ошибок, пользователям не следует загружать на свои устройства Excel-файлы.

В результате дипломной работы была разработана автоматизированная система составления расписания занятий образовательного заведения с целью ее рассмотрения для внедрения в учебный процесс образовательного учреждения. Данная система имеет удобный пользовательский интерфейс, позволяющий легко освоить работу в программе. Преследуемая изначально цель автоматизации процесса расписания занятий достигнута. Разработанная система позволит повысить скорость обработки информации, сократит сроки формирования расписания и сэкономит время работы пользователей [10].

СПИСОК ЛИТЕРАТУРЫ

- 1 ГОСТ 19.102-77. Стадии разработки. - Введ. 01.01.1980. - М.: Межгосударственный совет по стандартизации, метрологии и сертификации, 2010. - 2 с. - (Единая система программной документации).
- 2 ГОСТ 19.201-78. Техническое задание. Требования к содержанию и оформлению. - Введ. 01.01.1980. - М.: Межгосударственный совет по стандартизации, метрологии и сертификации, 2013. - 2 с. - (Единая система программной документации).
- 3 ГОСТ 19.402-78. Описание программы. Введ. 01.01.1980. - М.: Межгосударственный совет по стандартизации, метрологии и сертификации, 2013. - 2 с. - (Единая система программной документации).
- 4 ГОСТ 19.404-79. Пояснительная записка. Требования к содержанию и оформлению. - Введ. 01.01.1981. - М.: Межгосударственный совет по стандартизации, метрологии и сертификации, 2013. - 2 с. - (Единая система программной документации).
- 5 ГОСТ 19.504-79. Руководство программиста. Требования к содержанию и оформлению. - Введ. 01.01.1980. - М.: Межгосударственный совет по стандартизации, метрологии и сертификации, 2011. - 1 с. - (Единая система программной документации).
- 6 ГОСТ 19.505-79. Руководство оператора. Требования к содержанию и оформлению. - Введ. 01.01.1980. - М.: Межгосударственный совет по стандартизации, метрологии и сертификации, 2014. - 2 с. - (Единая система программной документации).
- 7 ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения. - Взамен ГОСТ 19.002-80, ГОСТ 19.003-80; введ. 01.01.1992. - М.: Межгосударственный совет по стандартизации, метрологии и сертификации, 2011. - 23 с. - (Единая система программной документации).

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		56

- 8 ГОСТ 2.105-95. Общие требования к текстовым документам. - Взамен ГОСТ 2.105-79, ГОСТ 2.906-71; введ. 01.07.1996. - Минск: Межгосударственный совет по стандартизации, метрологии и сертификации, 2011. - 19 с. - (Межгосударственный стандарт. Единая система конструкторской документации).
- 9 ГОСТ 7.1. – 2014 «Библиографическая запись. Библиографическое описание».
- 10 ГОСТ 7.32. – 2012 «Система стандартов по информации, библиотечному и издательскому делу «Отчет о научно-исследовательской работе».
- 11 ГОСТ 7.82. – 2010 «Библиографическая запись».
- 12 Бхаргава Адитья, А.Б. Грокаем алгоритмы. / А.Б. Бхаргава Адитья. – Санкт-Петербург : Издательский Дом ПИТЕР, 2022. – 288 с.: ил.
- 13 Дари, К. AJAX и PHP. Разработка динамических веб-приложений / К. Дари, Б. Бринзаре, Ф. Черчез-Тоза, М. Бусика. - СПб.: Символ-плюс, 2018. - 336 с.
- 14 Миковски, М.С. Разработка одностраничных веб-приложений / М.С. Миковски, Д.К. Пауэлл. - М.: ДМК, 2019. - 512 с.
- 15 Мюллер, Р.Д. Проектирование баз данных и UML / Р.Д. Мюллер; Пер. с англ. Е.Н. Молодцова. - М.: Лори, 2018. - 420 с.
- 16 Автоматизация составления расписания учебных занятий в вузе // Материалы Всероссийской научно-технической конференции «Наукоемкие технологии в приборо- и машиностроении и развитие инновационной деятельности в ВУЗе». - Т.2. - М., Издательство МГТУ им. Н.Э.Баумана, 2020. - 98 с.
- 17 Атрощенко В.А. К вопросу проектирования автоматизированной системы составления расписаний с учетом приоритетов заявок // Сборник международной научно- практической конференции «Научные исследования и их практическое применение. Современное состояние и

пути развития 2020». Том 5. / Атрощенко В.А., Семенюта И.С. - Одесса: Черноморье, 2020. – 58 с.

- 18 Афонин, А.М. Теоретические основы разработки и моделирования систем автоматизации: Учебное пособие - (ГРИФ) / А.М. Афонин. - М.: Форум, 2022. - 192 с.
- 19 Батищев, П.С. Опыт использования информационных технологий при составлении расписания учебных занятий. Текст. // Среднее профессиональное образование. - №11. - 2023 - 77 с.
- 20 Беленький, А.С. Применение моделей и методов теории расписаний в задачах оптимального планирования. Текст. / А.С. Беленький, Е.В. Левнер // Автоматика и телемеханика. 2019. -№1.- 77 с.
- 21 Брезгинов, А.Н. Обзор существующих методов составления расписаний / А.Н. Безгинов, С.Ю. Трегубов // Информационные технологии в программировании. - М., 2021. - 12 с.
- 22 Воробович, О.Н. Алгоритм формирования расписания занятий студенческих групп в высшем учебном заведении Текст. / О.Н. Воробович // Материалы межвузовской научной конференции / под ред. Е.А. Вейсова, Ю.А. Шитова, КГТУ. - Красноярск, 2023.- 29-35 с.
- 23 Воробович, О.Н. Информационная система формирования расписания занятий в высшем учебном заведении Текст. / О.Н. Воробович // Вестник СибГТУ, N1 / СибГТУ. Красноярск 2023. - 120-125 с.
- 24 Воробович, О.Н. Метод формирования расписания занятий студенческих групп в высшем учебном заведении Текст. / О.Н. Воробович, Н.П. Воробович // Вестник КГТУ. Выпуск 33. Математические методы и моделирование. / КГТУ. Красноярск:, 2018. - 166-176 с.
- 25 Галузин, К.С. Гибридный алгоритм решения задачи составления оптимального учебного расписания. Текст. / К.С. Галузин, В.Ю. Столбов // Информационные технологии в образовании: Сб. трудов XIII международной конференции-выставки. М., 2022. 130-131 с.

- 26 Галузин, К.С. Методика составления оптимального учебного расписания с учетом предпочтений Текст. / К.С. Галузин, В.Ю. Столбов // Теоретические и прикладные аспекты информационных технологий: Сб.науч.тр. /ГосНИИУМС. Вып. 53. - Пермь, 2019. – 50 с.
- 27 Кабальнов, Ю.С. Композиционный генетический алгоритм составления расписания учебных занятий / Ю.С. Кабальнов, Л.И. Шехтман, Г.Ф. Низамова, Н.А. Земченкова // Вестник Уфимского государственного авиационного технического университета. - 2020. – 9 с.
- 28 Колисниченко, Д.Н. PHP и MySQL. Разработка веб-приложений. Профессиональное программирование / Д.Н. Колисниченко. - СПб.: BHV, 2021. - 592 с.
- 29 Конвей, Р. В. Теория расписаний. Текст. / Р. В. Конвей, В. Л. Максвелл, Л. В. Миллер М.: Наука, 2021. - 360 с.
- 30 Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения. / Р. Мартин. – СПб : Питер, 2021. – 352 с.
- 31 Рылько, М. Компьютерные технологии в проектировании / М. Рылько. - М.: АСВ, 2022. - 326 с.
- 32 Соловьев, И.В. Проектирование информационных систем. Фундаментальный курс / И.В. Соловьев, А.А. Майоров. - М.: Академический проект, 2019. - 398 с.
- 33 Черноруцкий, И.Г. Методы оптимизации. Компьютерные технологии / И.Г. Черноруцкий. - СПб.: BHV, 2021. - 384 с.
- 34 Шпак, Ю.А. Проектирование баз данных. Просто как дважды два / Ю.А. Шпак. - М.: Эксмо, 2018. - 304 с.
- 35 Щитов, И.Н. Введение в методы оптимизации. / И.Н. Щитов. - М.: Высшая школа, 2018. - 206 с.

ПРИЛОЖЕНИЕ А
Техническое задание на программный продукт

Министерство образования и науки Республики Башкортостан
Государственное бюджетное профессиональное образовательное учреждение
Уфимский государственный колледж технологии и дизайна

Рассмотрено на заседании
ЦК математических и
естественнонаучных дисциплин
Протокол № _____
«_____» _____ 202__ г
Председатель ЦК
_____ О.Н. Заглядина

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА
ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ**

Ф. И. О. студента Ханнанова Алмаза Расиховича

Учебная группа И-19-19

Специальность Информационные системы (по отраслям)

Тема: Разработка веб-приложения «Advanced Schedule»

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		60

1. ВВЕДЕНИЕ

Настоящее техническое задание распространяется на разработку веб-приложения «Advanced Schedule». Предполагается, что использовать данную систему будут преподаватели колледжа и студенты.

Веб-приложение позволит преподавателям, студентам в упрощенной онлайн форме просматривать расписание занятий.

2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

Веб-приложение разрабатывается на основании приказа колледжа «Об утверждении тем, руководителей и консультантов выпускных квалификационных работ студентов» № 2/395а от 05.12.2022 г.

3. НАЗНАЧЕНИЕ

Разработка производится с целью создания программного продукта, с помощью которого можно составлять расписание в полу-автоматизированном режиме и просматривать его с любых браузеров.

4. ТРЕБОВАНИЯ К ПРОГРАММНОМУ ПРОДУКТУ

4.1 Требования к функциональным характеристикам

Добавление и редактирование данных:

- группы;
- аудитории;
- расписание звонков;
- предметы;
- преподаватели.

С возможностями: сортировки данных по различным параметрам, отображения результатов действий оператора по составлению расписания, вывода расписания по аудиториям, группам и преподавателям. Создание новых учетных записей с правами администратора. Незарегистрированный пользователь, администратор и преподаватель. Эта система – прикладная программа, которая должна быть полезна для пользователей в том, что они смогут просматривать расписание в любой точке мира при наличии интернета и браузера.

4.2 Требования к надежности

Действия пользователя не должны приводить к некорректной или неправильной работе программы.

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		61

4.3 Требования к составу и параметрам технических средств

Требования к составу и параметрам технических средств накладываются к требованиям операционной системы.

4.4 Требования к информационной и программной совместимости

Веб-приложение должно быть запущено в операционной среде Windows 10/11 / Linux Ubuntu/Fedora/Debian/Red OS.

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

Вместе с программным продуктом должны предоставляться пояснительная записка и руководство пользователя. Руководство должно содержать всю необходимую информацию по эксплуатации в отдельном разделе.

6. ЭТАПЫ РАЗРАБОТКИ

№	Название этапа	Срок	Отчетность
1	Разработка ТЗ	до 03.03.2023	Техническое задание
2	Разработка веб-приложения	до 16.04.2023	Код программы
3	Тестирование	до 21.05.2023	Программный продукт

Дата выдачи задания: 02.12.2022г.

Руководитель: Г.Р. Валеева _____
(ФИО, подпись преподавателя)

Задание принял(а) к исполнению: А.Р. Ханнанов _____
(ФИО, подпись студента)

ПРИЛОЖЕНИЕ Б
Программный код маршрутизатора

```
Route::get('/admin', function () {
    return view('layouts.admin');
})->middleware(['isAuth', 'isAdmin']);

Route::get('/', function () {
    return view('layouts.index');
});

Route::get('/login', function () {
    if (\Illuminate\Support\Facades\Auth::check()) {
        return redirect('/');
    }
    return view('layouts.login');
});

Route::post('/login', [\App\Http\Controllers\UserController::class, 'login']);
Route::get('/logout', [\App\Http\Controllers\UserController::class, 'logout'])->middleware(['isAuth', 'isAdmin']);

Route::get('/search', [\App\Http\Controllers\MainReadController::class, 'search']);

Route::prefix('api')->group(function () {
    Route::get('/lessons', [\App\Http\Controllers\MainReadController::class, 'getLessons']);

    Route::get('/Group', [\App\Http\Controllers\MainReadController::class, 'getAllGroups']);

    Route::get('/Department',
[\App\Http\Controllers\MainReadController::class, 'getAllDepartments']);

    Route::get('/Audience',
[\App\Http\Controllers\MainReadController::class, 'getAllAudiences']);

    Route::get('/Week', [\App\Http\Controllers\MainReadController::class, 'getAllWeeks']);
});
```

```

        Route::get('/lessonsOrder',
[\App\Http\Controllers\MainReadController::class, 'getAllLessonOrders']);

        Route::get('/groupsPart',
[\App\Http\Controllers\MainReadController::class, 'getAllGroupParts']);

        Route::get('/Subject', [\App\Http\Controllers\MainReadController::class,
'getAllSubject']);

        Route::get('/Prepod', [\App\Http\Controllers\MainReadController::class,
'getAllPrepods']);

        Route::get('/Captain', [\App\Http\Controllers\MainReadController::class,
'getCaptains']);

        Route::get('/Schedule', [\App\Http\Controllers\MainReadController::class,
'getSchedules']);

        Route::get('/lessonBooking',
[\App\Http\Controllers\MainReadController::class, 'getBookings']);

        Route::get('/lessonBooking/group',
[\App\Http\Controllers\MainReadController::class, 'getGroupBooking']);

        Route::get('/lessonBooking/teacher',
[\App\Http\Controllers\MainReadController::class, 'getTeacherBooking']);

        Route::get('/lessonBooking/audience',
[\App\Http\Controllers\MainReadController::class, 'getAudienceBooking']);

        Route::post('/User/register', [\App\Http\Controllers\UserController::class,
'register']);

        Route::post('/User/changePassword',
[\App\Http\Controllers\UserController::class, 'changePassword']);

        Route::delete('/User', [\App\Http\Controllers\UserController::class,
'deleteUser']);

        Route::patch('/User', [\App\Http\Controllers\UserController::class,
'editUser']);

```



```

        Route::get('/User',    [\App\Http\Controllers\MainReadController::class,
'getUsers']);

        Route::get('/Roles',    [\App\Http\Controllers\MainReadController::class,
'getRoles']);

        Route::post('/Audience',
[\App\Http\Controllers\AudienceController::class, 'createAudience']);

        Route::delete('/Audience',
[\App\Http\Controllers\AudienceController::class, 'deleteAudience']);

        Route::patch('/Audience',
[\App\Http\Controllers\AudienceController::class, 'editAudience']);

        Route::post('/Department',
[\App\Http\Controllers\DepartmentController::class, 'createDepartment']);

        Route::delete('/Department',
[\App\Http\Controllers\DepartmentController::class, 'deleteDepartment']);

        Route::patch('/Department',
[\App\Http\Controllers\DepartmentController::class, 'editDepartment']);

        Route::post('/groupsPart',
[\App\Http\Controllers\GroupsPartsController::class, 'createGroupsPart']);

        Route::delete('/groupsPart',
[\App\Http\Controllers\GroupsPartsController::class, 'deleteGroupsPart']);

        Route::patch('/groupsPart',
[\App\Http\Controllers\GroupsPartsController::class, 'editGroupsPart']);

        Route::post('/lessonsOrder',
[\App\Http\Controllers\LessonsOrdersController::class, 'createLessonsOrder']);

        Route::delete('/lessonsOrder',
[\App\Http\Controllers\LessonsOrdersController::class, 'deleteLessonsOrder']);

    });

```

ПРИЛОЖЕНИЕ В
Программный код контроллера

```
namespace App\Http\Controllers;

use App\Contracts\ScheduleCheckContract;
use App\Contracts\ScheduleEditContract;
use App\Http\Requests\Admin\ScheduleRequest;
use App\Models\Audience;
use App\Models\Schedule;
use Illuminate\Http\JsonResponse;
use Illuminate\Support\Facades\Validator;

class ScheduleController extends Controller
{
    private \Illuminate\Validation\Validator $val;
    private function validateTimes(array $request) {
        $this->val = Validator::make($request, [
            'start_time.hours' => 'numeric',
            'start_time.minutes' => 'numeric',
            'end_time.hours' => 'numeric',
            'end_time.minutes' => 'numeric',
        ]);
        return $this->val->fails();
    }

    public function createSchedule (ScheduleRequest $request,
ScheduleCheckContract $scheduleCheck, ScheduleEditContract $scheduleEdit):
JsonResponse
    {
        $request = $request->validated();
```

					09.02.04.ИС.И-19-19.2/395а.23.ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		66

```

        if ($this->validateTimes($request)) {
            return $this->sendError('Ошибка валидации', 'Проверьте
            правильность заполнения полей', $this->val->errors(), 422);
        }

        $hoursDifference = $request['end_time']['hours'] -
        $request['start_time']['hours'];
        $minutesDifference = $request['end_time']['minutes'] -
        $request['start_time']['minutes'];

        if ($hoursDifference<=0 && $minutesDifference<=0) {
            return $this->sendResp('Ошибка', 'Некорректно задано время,
            длительность пары составляет 0 минут', 422);
        }

        if ($scheduleCheck($request)) {
            return $this->sendResp('Ошибка', 'Для выбранного отделения, дня
            недели и пары уже существует расписание', 422);
        }

        $scheduleEdit($request, new Schedule)->save();
        return $this->sendResp('Успешно', 'Расписание было успешно
        добавлено', 200);
    }

    public function editSchedule (ScheduleRequest $request,
    ScheduleEditContract $scheduleEdit): JsonResponse
    {
        $request = $request->validated();
        if ($request['start_time'] && $request['end_time']) {
            if ($this->validateTimes($request)) {
                return $this->sendError('Ошибка валидации', 'Проверьте
                правильность заполнения полей', $this->val->errors(), 422);
            }
        }
    }

```

```

        }

        $hoursDifference      =      $request['end_time']['hours']      -
$request['start_time']['hours'];

        $minutesDifference    =      $request['end_time']['minutes']    -
$request['start_time']['minutes'];

        if ($hoursDifference<=0 && $minutesDifference<=0) {
            return $this->sendResp('Ошибка', 'Некорректно задано время,
длительность пары составляет 0 минут', 422);
        }
    }

    $schedule = $scheduleEdit($request, Schedule::find($request['id']));
    $schedule->save();

    return $this->sendResp('Успешно', 'Расписание было успешно
отредактировано', 200);
}

public function deleteSchedule (ScheduleRequest $request) {
    $request = $request->validated();
    return $this->deleteSomething(new Schedule(), $request['id'],
        $this->sendResp('Успешно', 'Расписание было успешно удалено',
200));
}
}

```