МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”**

Факультет компьютерных наук

Кафедра информационных технологий в управлении предприятием

Курсовая работа по курсу

«Технологии программирования»

«Система для составления расписания факультета»

Выполнили: студенты 3 курса, группы 3.2

Пастухова О.В.  
Дьяконова М.А.

Воронеж 2019

[**Введение**](#_gjdgxs) **3**

[**Глава 1. Постановка задачи**](#_30j0zll) **4**

[**Глава 2. Анализ предметной области**](#_1fob9te) **5**

[2.1 Глоссарий](#_3znysh7) 5

[2.2 Аналоги решения задачи](#_2et92p0) 6

[2.3 Методология](#_tyjcwt) 8

[2.4 Моделирование бизнес процессов составления расписания](#_t1015pmtj65r) 10

[2.5 Анализ требований](#_3dy6vkm) 12

[2.6 Анализ процесса получения списка аудиторий](#_1t3h5sf) 13

[2.7 Анализ процесса изменения преподавателя](#_4d34og8) 15

[2.8 Анализ процесса изменения расписания](#_2s8eyo1) 16

[2.9 Анализ работы приложения](#_17dp8vu) 17

[**Глава 3. Архитектура приложения**](#_26in1rg) **20**

[3.1 Анализ структуры данных](#_lnxbz9) 20

[3.2 Диаграмма классов](#_1ksv4uv) 22

[3.3 Дизайн интерфейса страниц](#_44sinio) 24

[3.4 Средства реализации](#_g3f746r70w01) 28

## Введение

Для того чтобы облегчить работу человека, в настоящее время во многие сферы общества внедряются различные программные продукты. Не исключением стала и выбранная нами область деятельности.

Одной из основных задач учебного процесса является грамотное составление расписания занятий. Оно должно обеспечивать равномерную загрузку студенческих групп и преподавательского состава, так как это напрямую влияет на качество обучения. При составлении расписания нужно учитывать требования по организации учебного процесса, пожелания преподавателей, наличие свободных аудиторий и при всем этом надо стремиться исключить большие перерывы между занятиями, как у преподавателей, так и у студентов.

Также немаловажно, что учебное расписание должно быть достаточно информативным. Там должны быть отражены не только аудитория и время проведения занятий, но и номер группы и курса, название предмета и имя преподавателя.

В итоге, задача составления расписания становится очень сложной, так как при ее реализации существует много нюансов. При его заполнении не всегда получается минимизировать затраты во времени. Также не всегда корректно подбираются аудитории и часто можно столкнуться с большими перерывами между предметами в расписании, что составляет некоторые неудобства как студентам, так и преподавателям.

Все это приводит к низкой результативности обучения, поэтому автоматизация данного процесса является одной из основных проблем в учебных заведениях.

Противоречие связано с тем, что проблема достижения положительных результатов, а точнее минимизация незадействованного для учебы времени, решена, однако методов для её решения недостаточно, ведь большинство подобных приложений имеют либо платную основу, либо непонятный интерфейс.

Отсюда вытекает актуальность работы. Именно поэтому в качестве направления деятельности была выбрана разработка приложения для пользователя с любыми навыками владения компьютером.

Цель работы – реализовать приложение для составления расписания занятий, которое позволит учесть все требования и пожелания, предъявляемые к нему, оптимизировать его и сократить время, потраченное на его составление.

Предмет исследования – учебный план и аудиторный фонд стандартного университета.

Объектом исследования является расписание учебного процесса для университета.

Рабочая гипотеза заключается в предположении, что организация занятий в течение учебного года может быть оптимизирована путем применения программного продукта, реализованного в рамках научной работы. Процесс оптимизации будет выполняться на основе автоматизации выбора либо аудиторий, либо времени. А также вывод результата в удобном для пользователя формате.

Практическая значимость состоит в возможности применения данного приложения для решения реальных задач, возникающих перед организациями. Оно призвано помочь заинтересованным в составлении расписания лицам в упрощении данного процесса.

## Глава 1. Постановка задачи

Целью данной работы является разработка системы для автоматизации процесса составления учебного расписания на факультете.

Для достижения поставленной цели необходимо решить следующие задачи **:**

* Определить способы составления корректного расписания
* Спроектировать БД, содержащую все необходимые данные
* Сформулировать основные методы, доступные для администратора системы
* Разработать программный продукт для автоматизации составления расписания

Необходимо, чтобы данная система удовлетворяла следующим требованиям:

* Возможность внесения изменений в существующую базу данных.
* Возможность вывода свободных аудиторий для занятия и осуществление подбора аудитории для лекции или для практического занятия.
* Возможность вывода списков всех групп на каждом курсе, всех преподавателей и предметов, соответствующих каждому преподавателю.
* Хранение всей необходимой информации о предметах, преподавателях и группах студентов в базе данных.
* Наличие логичного и понятного пользовательского интерфейса.
* Экспорт расписания в какой-либо распространенный формат(PDF, Microsoft Excel).

## Глава 2. Анализ предметной области

### 2.1 Глоссарий

Для конкретного понимания темы работы составлен небольшой справочник узкоспециализированных слов. Ниже представлены понятия и определения к ним.

1. Расписание - вид календаря (т.е. упорядоченность по времени), для которого указана информация о предстоящих (планируемых) событиях. Обычно представляется в виде таблицы.
2. Аудитория - помещение для проведения лекций. Не оборудовано компьютерной и специализированной техникой.
3. Лаборатория - помещение для проведения практических работ. Оборудовано специализированной техникой (компьютерами).
4. Временная решётка - время начала и окончания пар, согласно расписанию звонков.
5. Преподаватель - человек, педагог, который занимается преподаванием какого-либо предмета. Работник вуза.
6. Читаемый курс (предмет) – учебная дисциплина, преподаваемая педагогом.
7. Лабораторные занятия (практические) - занятия в отдельных группах. Проведение учащимися по заданию преподавателя опытов с использованием спец оборудования. Проводятся в лабораториях.
8. Потоковая лекция - лекция, читаемая для нескольких групп одновременно. Проводятся в аудиториях.
9. Группа студентов - учётная единица. Характеризуется названием (номером) и относительно постоянным составом.
10. Учебная пара- аудиторное занятие, происходит в виде двух совмещённых академических часов.

### 2.2 Аналоги решения задачи

Можно рассмотреть такое программное средство как 1С: Автоматизированное составление расписания. Университет.1. Данное программное обеспечение предназначается для решения задач по успешной автоматизации составления расписания учебных в ВУЗах. Система позволяет составлять расписания учебных занятий в смешанном, ручном и автоматическом режиме. Продукт предназначен для решения задач автоматизированного составления учебных расписаний и оперативного управления помещениями в ВУЗах.   Вид программы 1С:Автоматизированное составление расписания. Университет представлен на рисунке 1.

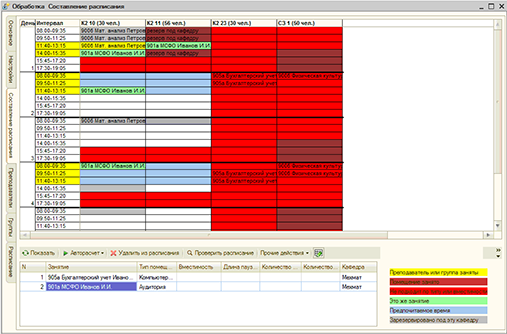


Рисунок 1 - Интерфейс программы 1С:Автоматизированное составление расписания

Следующее программное средство, которое используется для составления расписания - экспресс-расписание ВУЗ2. Экспресс-расписание ВУЗ2 - программа для автоматизации составления расписания учебных занятий в ВУЗах. Программа автоматически составляет основное расписание, позволяет вести ежедневные изменения расписания, учет выполненных часов, формирует разнообразные отчеты. Вид данной программы представлен на рисунке 2.

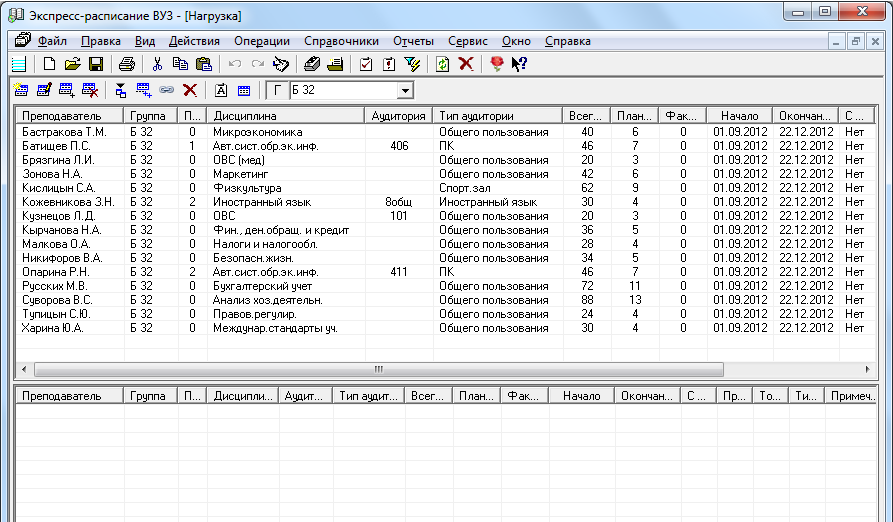


Рисунок 2 - Интерфейс программы Экспресс-расписание ВУЗ

Минус этих программных средств заключается в их стоимости. 1С: Автоматизированное составление расписания. Университет.1 стоит около семидесяти тысяч, а Экспресс-расписание ВУЗ – двадцать тысяч. А также их недостаток в сложном понимании для неуверенного пользователя ПК.

### 2.3 Методология

Модель Waterfall выделяет следующие стадии:

1. Анализ системных и программных требований, которые закреплены в документе.

2. Планирование всех этапов разработки. Важный пункт, так как вся последующая работа будет четко следовать составленному плану.

3. Проектирование. Разрабатывается внутренняя архитектура проекта, его внешний вид, структура, рассматриваются варианты реализации.

4. Реализация дизайна, верстки, программного продукта.

5. Интеграция. Проводятся необходимые работы по обмену данных и пишется код программы.

6. Тестирование. Готовый продукт проверяется на наличие программных ошибок, также выявляются недочеты функционала. После этого идет исправление нужных багов.

7. Выпуск продукта. Релиз готового проекта и окончание разработки. Возможна также работа по адаптации проекта к иным видам систем.

8. Техническая поддержка. Поддержание работоспособности ресурса и оперативное реагирование на возникающие вопросы или проблемы в системе.

Следуя модели водопада, разработчик переходит от одной стадии к другой строго последовательно. Сначала полностью завершается этап «определение требований», в результате чего получается список требований к ПО. После того как требования полностью определены, происходит переход к проектированию, в ходе которого создаются документы, подробно описывающие для программистов способ и план реализации указанных требований. После того как проектирование полностью выполнено, программистами выполняется реализация полученного проекта. На следующей стадии процесса происходит интеграция отдельных компонентов, разрабатываемых различными командами программистов. После того как реализация и интеграция завершены, производится тестирование и отладка продукта; на этой стадии устраняются все недочёты, появившиеся на предыдущих стадиях разработки. После этого программный продукт внедряется и обеспечивается его поддержка - внесение новой функциональности и устранение ошибок.

Тем самым, модель водопада подразумевает, что переход от одной фазы разработки к другой происходит только после полного и успешного завершения предыдущей фазы, и что переходов назад либо вперёд или перекрытия фаз - не происходит.

Достоинством этой модели явилось ограничение возможности возвратов на произвольный шаг назад, например, от тестирования – к анализу, от разработки – к работе над требованиями и т.д. Отмечалось, что такие возвраты могут катастрофически увеличить стоимость проекта и сроки его выполнения. Например, если при тестировании обнаруживаются ошибки проектирования или анализа, то их исправление часто приводит к полной переделке системы. Этой моделью допускались возвраты только на предыдущий шаг, например, от тестирования к кодированию, от кодирования к проектированию и т.д.

Эта модель не подходит для нашего проекта, так как пользователи и заказчик не могут ознакомиться с вариантами системы во время разработки, и видят результат только в самом конце; тем самым, они не могут повлиять на процесс создания системы.

Поэтому в разработке программного обеспечения мы используем методику Agile. Agile является современной методикой для создания качественного, соответствующего требованиям заказчика программного обеспечения, и за меньшее время. Agile делает проект менее громоздким и более прозрачным.

В первую очередь стоит отметить, что Agile - управление очень гибкое. Если, например, традиционная методология указывает на конкретные этапы работы, то Agile легко подстраивается под потребителя конечного продукта и требования заказчика.

В конечном продукте число дефектов минимизируется, ведь он является результатом тщательной проверки качества, которая проводится по завершении каждого этапа-спринта.

Кроме того, Agile быстро запускается, легко реагирует на изменения, позволяет команде разработчиков и клиентов поддерживать постоянную связь в реальном времени.

### 2.4 Моделирование бизнес процессов составления расписания

Для отображения функций системы, а также имеющихся связей между функциями и внешней средой была построена диаграмма IDEF0. Центральным бизнес-процессом является функция составления расписания, которая на схеме отображается в виде функционального блока на рисунке 3.

На вход (стрелки слева) подаются списки дисциплин, групп, преподавателей и аудиторий, которые преобразуются в процессе.

Вход сверху - это поток управления в виде учебного плана. Он не преобразуется в процессе, но нужен для его выполнения. На основе учебного плана будут производиться все бизнес процессы, продемонстрированные на рисунке 4.

Вход снизу - это механизм. Он показывает людей, технические средства, информационные системы и т.п., при помощи которых бизнес процесс реализуется. В данном случае - механизмом является администратор.

На выходе системы получается составленное учебное расписание.

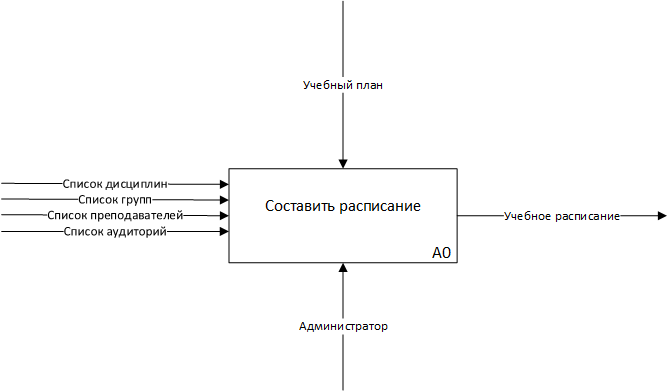


Рисунок 3 - Функциональный блок A0

На рисунке 4 представлена декомпозиция блока А0. Сложный процесс составления расписания разбит на составляющие его функции для понимания того, какие бизнес-процессы должна решать система и как они должны быть связаны между собой.

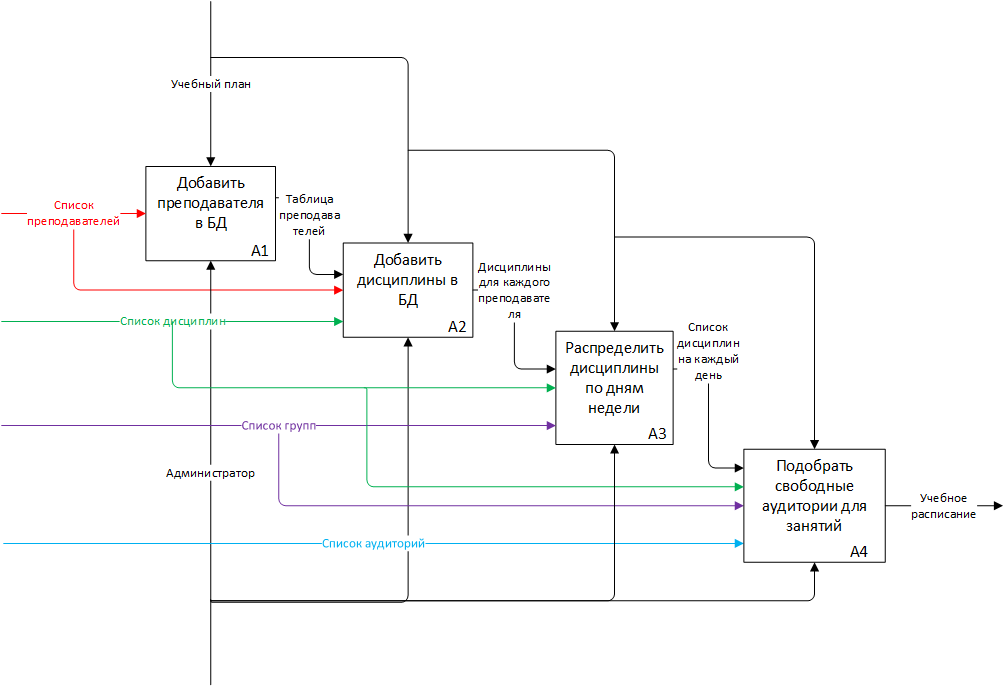


Рисунок 4 - Декомпозиция А0

### 2.5 Анализ требований

В данной работе общая идея проекта состоит из реализации некоторого количества заданий. Для уточнения этих требований рассмотрим диаграмму вариантов использования на рисунке 5.

На диаграмме видно, что есть такой актер, как администратор. Он выполняет все функции в системе. Эти функции называются вариантами использования. Для актера возможны следующие варианты использования:

1. Получить список преподавателей. Включает в себя получение предпочтений по расписанию от преподавателей. Расширяется вариантами: добавление преподавателя, удаление преподавателя и изменение преподавателя.
2. Получить список всех предметов. Имеет расширения: добавление предмета, изменение предмета и удаление предмета.
3. Составить расписание. Включает получение списка свободных аудиторий.
4. Получить список групп. Включает в себя получение списков предметов для каждой группы.

5. Внести изменения в расписания. Расширяется вариантами: изменить аудиторию и перенести занятие.

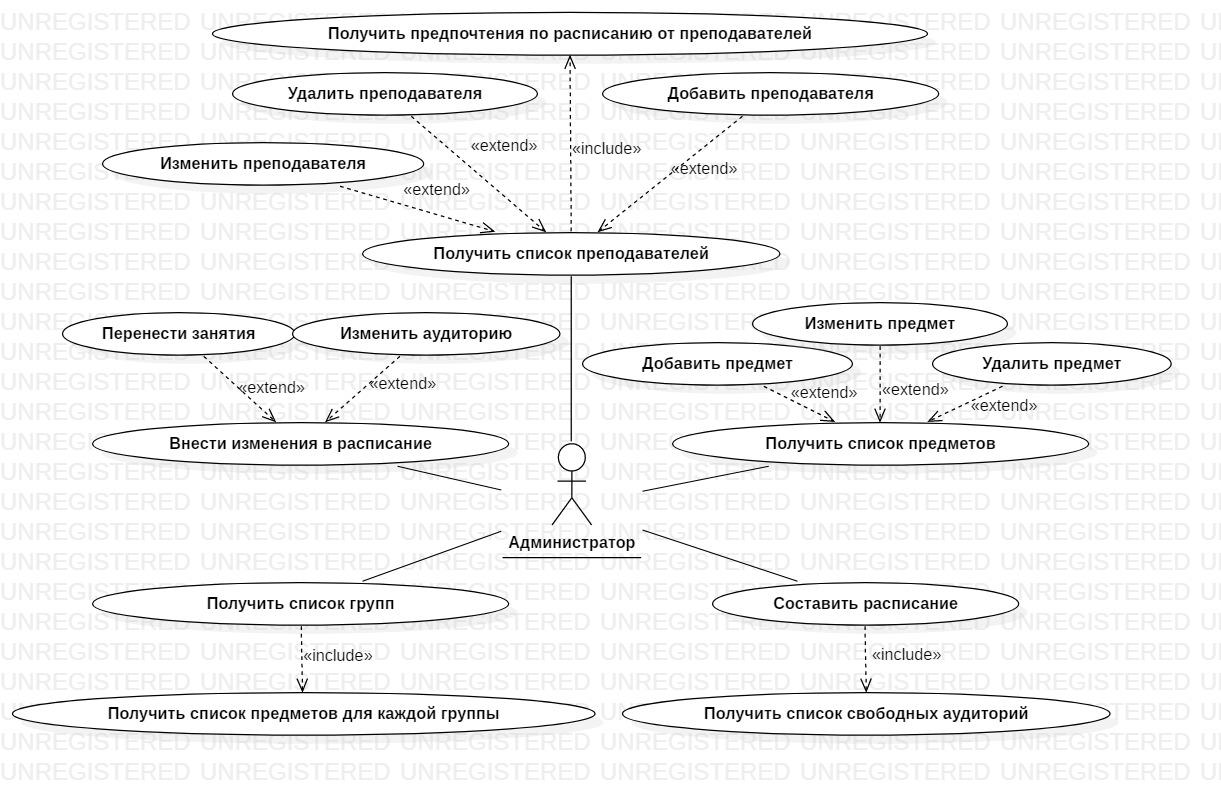
****

Рисунок 5 - Диаграмма вариантов использования

### 2.6 Анализ процесса получения списка аудиторий

Процесс получения списка аудиторий представлен диаграммой последовательности на рисунке 6.

Данная диаграмма последовательности соответствует варианту использования Получить список свободных аудиторий. Диаграмма содержит 2 объекта “Приложение” и “База данных” и одно действующее лицо - “Пользователь”. Для того чтобы получить список свободных аудиторий, пользователю необходимо перейти на вкладку Составления расписания. Далее необходимо выбрать пару, для которой нужно вывести свободные аудитории. Выведется информация о названии выбранной пары, ее преподавателя и список аудиторий. Далее пользователь нажимает на кнопку получения списка свободных аудиторий, обрабатывается запрос и выводятся все свободные аудитории на данное время.

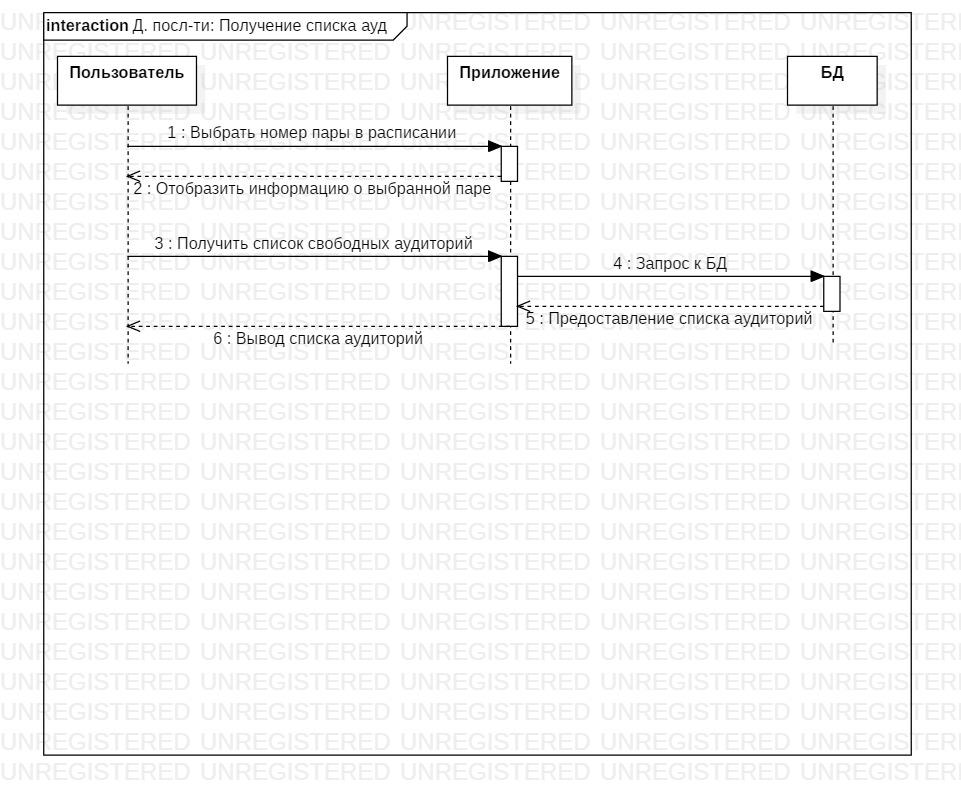


Рисунок 6 - Диаграмма последовательности “Получение списка аудиторий”

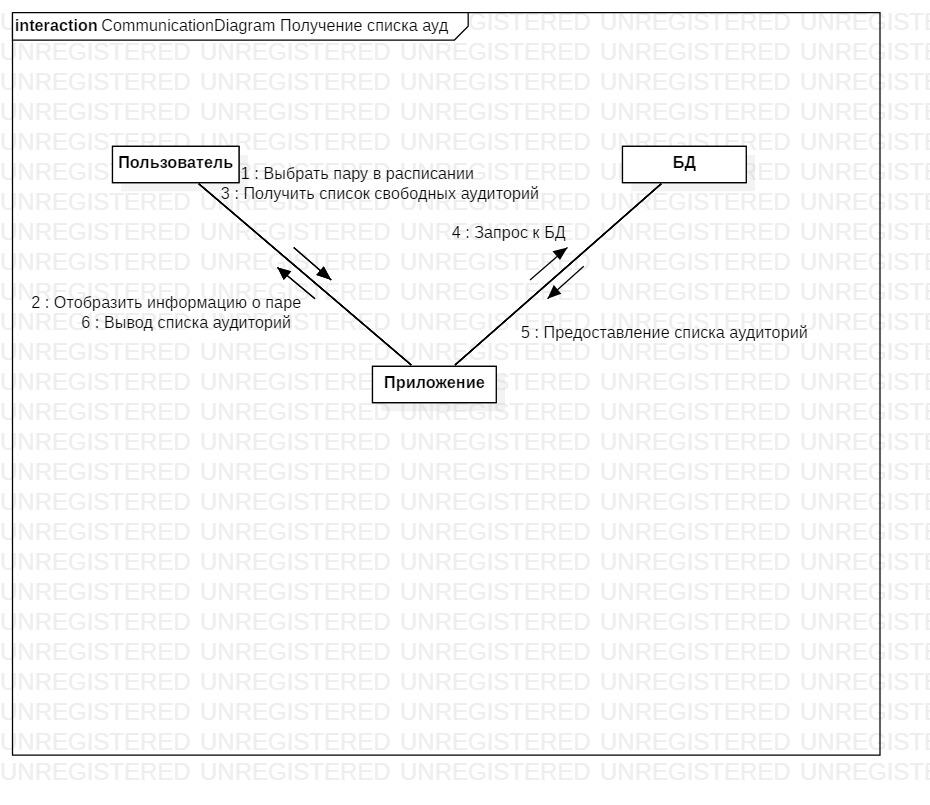
Диаграмма коммуникации для описанного процесса представлена на рисунке 7. Она явно указывает отношения между объектами в рамках процесса получения списка аудиторий, а время как отдельное измерение не используется (применяются порядковые номера вызовов)

Рисунок 7- Диаграмма коммуникации “Получение списка аудиторий”

### 2.7 Анализ процесса изменения преподавателя

На рисунках 8 и 9 показан процесс редактирования преподавателя пользователем. Для того чтобы отредактировать какого-либо преподавателя, на экран нужно вывести список всех преподавателей. Для вывода списка преподавателей, пользователь должен перейти на вкладку “Преподаватели”. При переходе на вкладку пользователю становится доступен список преподавателей, полученных из базы данных при запуске приложения. Далее пользователь выбирает нужную строку записи для редактирования и нажимает кнопку “Изменить”, после чего вносит необходимые изменения. Затем изменения заносятся в базу, и на экране отображается отредактированная запись в списке преподавателей.

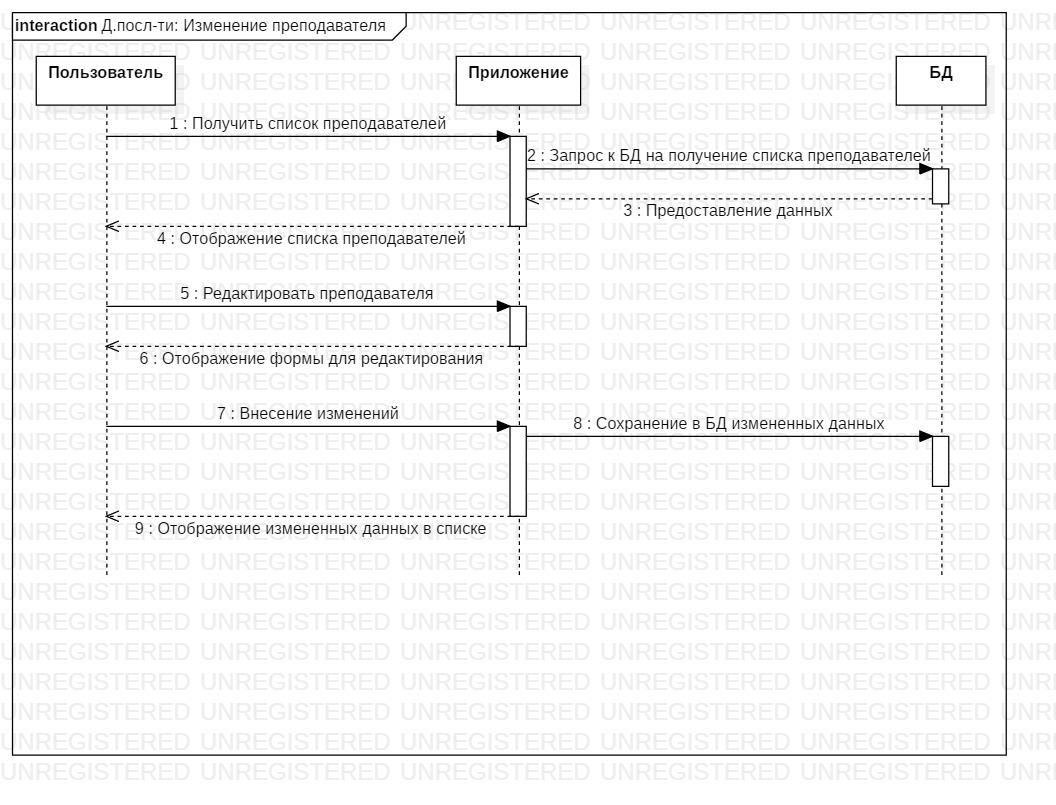


Рисунок 8 - Диаграмма последовательности “Изменение преподавателя”

Ниже на рисунке 9 представлена диаграмма коммуникации для процесса изменения преподавателя.

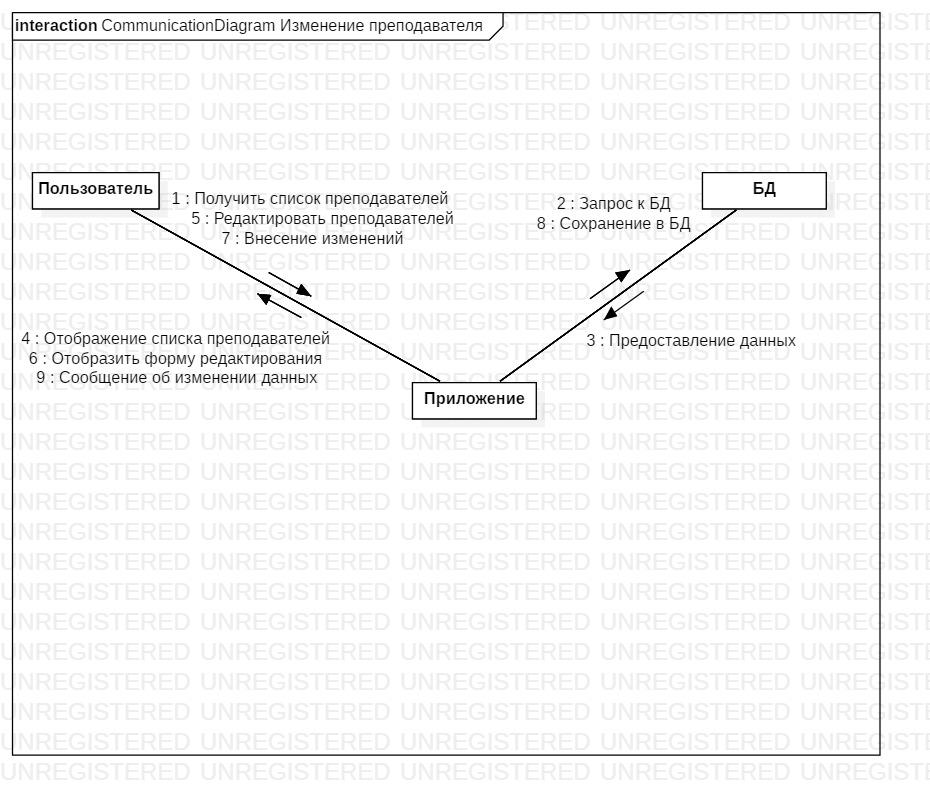


Рисунок 9 - Диаграмма коммуникации “Изменение преподавателя”

### 2.8 Анализ процесса изменения расписания

На рисунках 10 и 11 представлен процесс изменения расписания диаграммами последовательности и коммуникации. Действующим лицом является пользователь. Для изменения расписания необходимо перейти на вкладку составления расписания, нажать на кнопку изменения расписания. После обработки запроса появится таблица с существующим расписанием. Далее, пользователь вносит необходимые изменения, после этого они записываются в БД и отображаются в таблице расписания на форме.

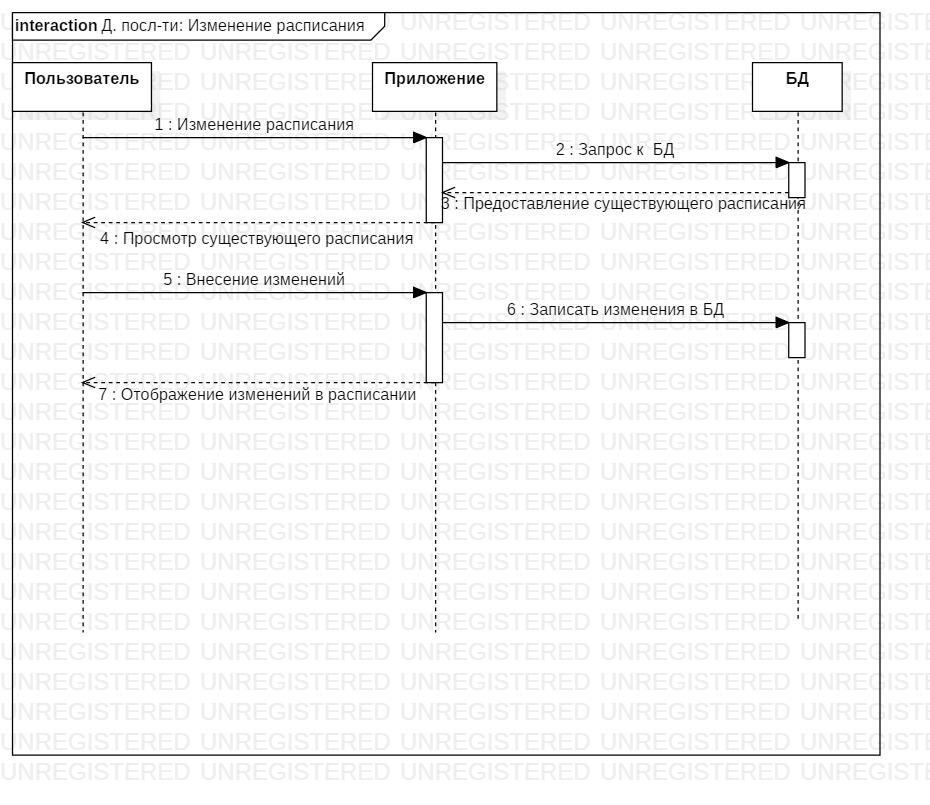
****

Рисунок 10 - Диаграмма последовательности “Изменение расписания”

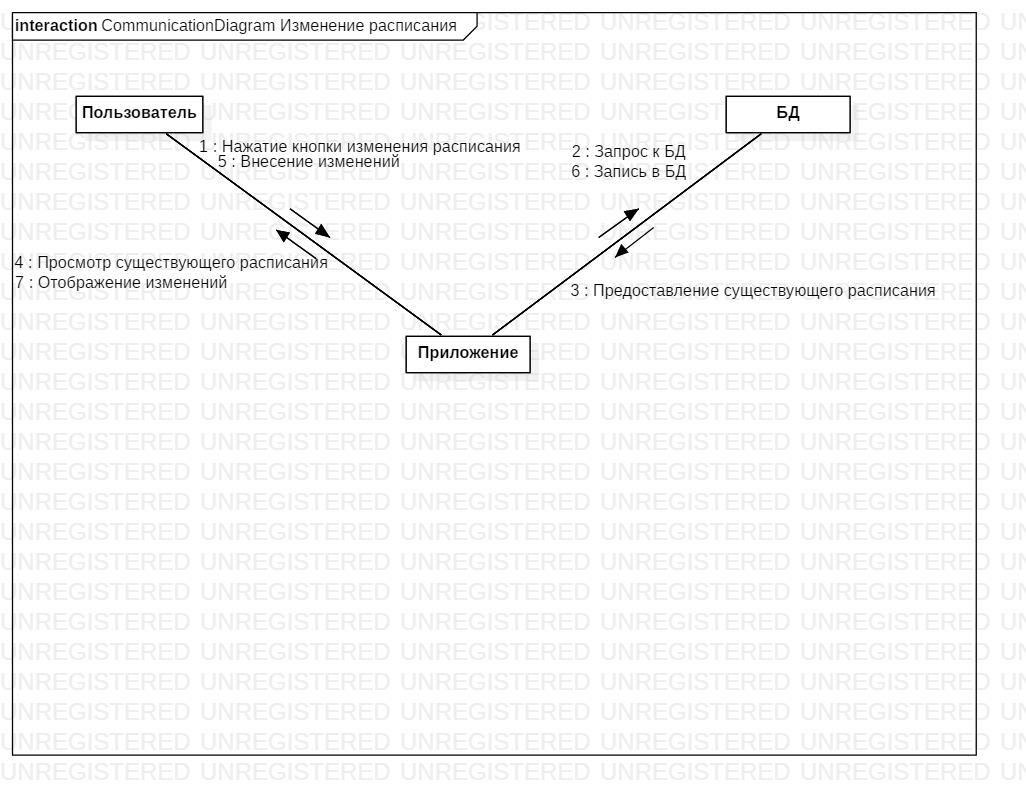


Рисунок 11 - Диаграмма коммуникации “Изменение расписания”

### 2.9 Анализ работы приложения

На рисунке 12 представлен процесс составления расписания. Изначально ожидается действие от пользователя, затем происходит операция сформировать расписание путем добавления для определенной группы соответствующих предметов на выбранное время. Если пользователь формирует новое расписание, то приложение получает список групп, предметов, преподавателей, аудиторий. Но если расписание уже существует, то пользователь может его отредактировать. После составления расписания, пользователь может вернуться в режим ожидания действия или выйти из системы.

### 

Рисунок 12 - Диаграмма деятельности

На рисунке 13 показан процесс смены состояний в результате составления расписания. Изначально приложение находится в состоянии ожидания действий от пользователя. Если расписание существует, то можно перейти в состояние редактирования, либо в формирование расписания, где система находится в стадии получения списка предметов, преподавателей и аудиторий. Заключительная стадия в системе - составление расписания.

****

Рисунок 13- Диаграмма состояния

На рисунке 14 показана диаграмма объектов для нашего приложения. Составление расписания требует взаимодействия нескольких различных объектов. Сценарий начинается с объекта класса Расписания. В свою очередь, объект Расписания может вызвать объекты других классов(Предмет, Группа, Аудитория, Преподаватель).

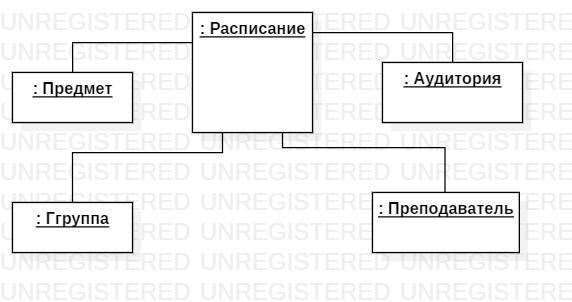


Рисунок 14- Диаграмма объектов

## Глава 3. Архитектура приложения

### 3.1 Анализ структуры данных

Для детального изучения работы можно рассмотреть ER-диаграмму, представленную на рисунке 15.

Модель «Сущность - Отношение» используется для проектирования баз данных с целью удаления всякой избыточности данных.

На диаграмме видно, что между сущностями выстраиваются различные отношения взаимодействия. Опишем связи между сущностями.

1. Преподаватель - Дисциплина. Степень связи m:m (многие ко многим). Класс принадлежности сущности «Преподаватель» к сущности «Дисциплина» обязательный, и наоборот. Каждый преподаватель может преподавать 1 или несколько дисциплин, каждая дисциплина может преподаваться одним или несколькими преподавателями.

2. Преподаватель – Пара. Степень связи 1:1. Класс принадлежности сущности «Преподаватель» к сущности «Пара» обязательный, и наоборот. Каждый преподаватель может преподавать только 1 пару в определенное время, а у каждой пары может быть только 1 преподаватель.

3. Дисциплина – Пара. Степень связи 1:1. Класс принадлежности сущности «Дисциплина» к сущности «Пара» обязательный, и наоборот. Каждая дисциплина преподается в на одной паре, а на каждой паре преподается только одна дисциплина.

4. Группа – Пара. Степень связи m:1 (многие к 1). Класс принадлежности сущности «Группа» к сущности «Пара» обязательный, и наоборот. На каждой паре могут присутствовать несколько групп, но у каждой группы может быть только 1 пара в определенное время.

5. Аудитория – Пара. Степень связи 1:m (1 ко многим). Класс принадлежности сущности «Аудитория» к сущности «Пара» обязательный, и наоборот. В каждой аудитории может проходить несколько пар, но каждая пара проходит в одной аудитории.

6. Дни\_недели – Пара. Степень связи 1:1. Класс принадлежности сущности «Дни\_недели» к сущности «Пара» обязательный, и наоборот. В каждый день недели проходит только одна конкретная пара, и одна конкретная пара проходит в определенный день недели.

7. Дисциплина – Тип\_занятия. Степень связи m:1 (многие к 1). Класс принадлежности сущности «Дисциплина» к сущности «Тип\_занятия» обязательный, и наоборот. У каждой дисциплины может быть только 1 тип занятия (лекция/практика), а у каждого типа занятия может быть несколько дисциплин.

8. Аудитория – Тип\_занятия. Степень связи m:1 (многие к 1). Класс принадлежности сущности «Аудитория» к сущности «Тип\_занятия» обязательный, и наоборот. В каждой аудитории проводятся занятия определенного типа, но для каждого типа занятий может быть несколько аудиторий.

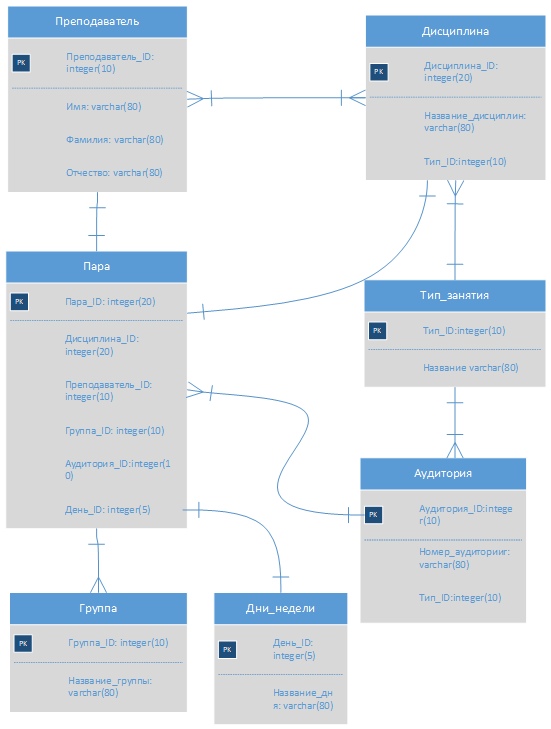


Рисунок 15 - ER-диаграмма системы

### 3.2 Диаграмма классов

Приложение создавалось на основе представленной на рисунке 16 диаграмме классов.

В состав приложения входят такие классы:

1. Queries. Класс, в котором реализованы все запросы, посылаемые к базе данных.

2. Connection. Класс, отвечающий за соединение с базой данных.

3. TimeTableForm. Данный класс представляет собой класс основной формы взаимодействия пользователя с программой. В нем содержится список всех занятий. Каждое занятие сочетает в себе преподавателя, дисциплину, которую данный преподаватель ведет, аудиторию, в которой проходит занятие, группу, которая присутствует на занятии и время проведения.

4. Классы EditDisciplineForm, EditLecturerForm, EditRoomForm, EditGroupForm представляют собой формы, возникающие при нажатии на кнопку редактирования на главной форме для дисциплины, преподавателя, аудитории и группы соответственно. Классы содержат в себе методы для добавления новых объектов, редактирования и удаления уже существующих.

5. Application. Класс, содержащий в себе вспомогательные методы для работы с базой данных и некоторыми элементами формы.

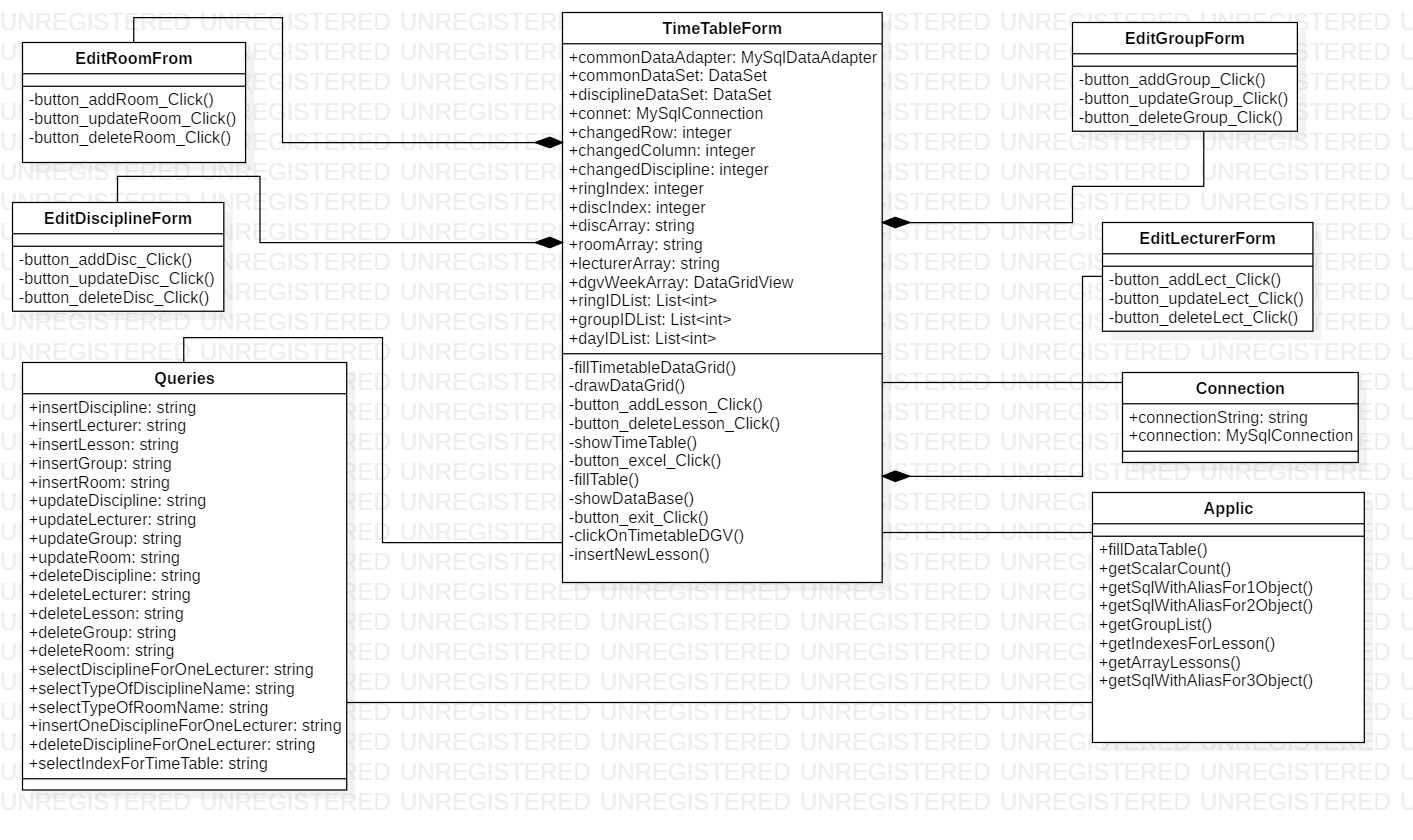
****

Рисунок 16 - Диаграмма классов

На первом этапе разработки приложения была создана база данных в СУБД MySQL. Для подключения базы данных к приложению использовался код, представленный на листинге 1.

Листинг 1 – Подключение БД.

static string connectionString = "server=localhost;user=root;database=facultytimetable; password=root;";

public static MySqlConnection connection = new MySqlConnection(connectionString);

В реализации программы был выделен класс (Queries), который содержит текст всех запросов к базе данных. Они необходимы для корректного вывода данных и дальнейшего их использования.

В программе присутствует основной класс, в котором наблюдается вся логика работы приложения. Методы выполняют основные действия при составлении и изменении расписания.

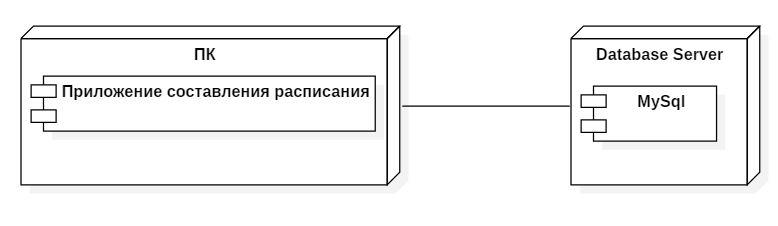
Для визуализации элементов и компонентов системы рассмотрим диаграмму развертывания, представленную на рисунке 17****

Рисунок 17 - Диаграмма развертывания

На персональном компьютере будет находиться разрабатываемое приложения составления расписания, в котором будет возможность хранить все данные и управлять ими на сервере базы данных MySql.

### 3.3 Дизайн интерфейса страниц

На главном экране разрабатываемого приложения будет возможность переключения между следующими вкладками:

1. «Расписание»

Данная вкладка откроется при запуске приложения. Она представляет собой экран, содержащий табличку с самим расписанием (оно может быть уже составлено, а может быть еще пустым) (Рисунок 18). На данном экране будут присутствовать кнопки «Сохранить» (сохраняет изменения в БД), «Вывести список свободных аудиторий» (выводится список всех свободных аудиторий на выбранное время в новом окне).



Рисунок 18

1. «Список преподавателей»

При выборе этой вкладки появится экран, соответствующий рисунку 19. Будет выведена таблица, показывающая расписание занятий для каждого преподавателя.

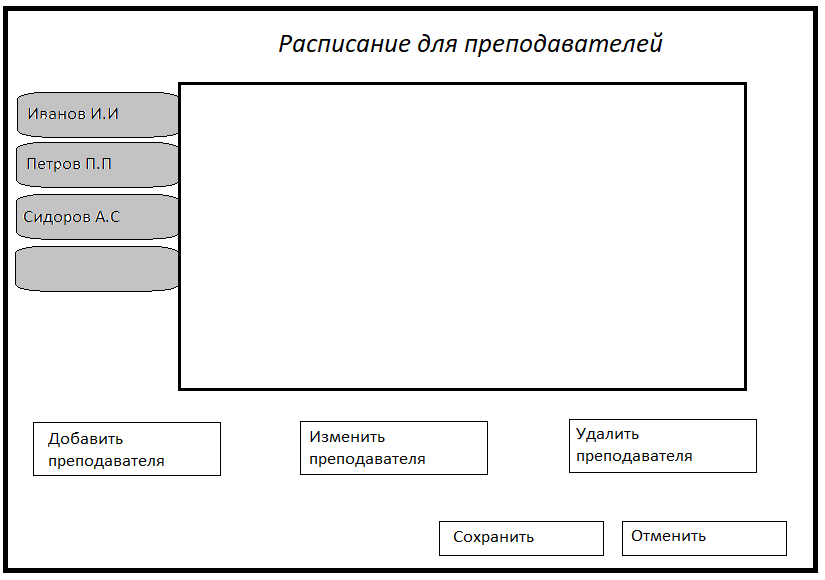


Рисунок 19

На рисунке 19 присутствуют кнопки «Сохранить» (сохраняет изменения в БД), «Отменить» (отменяет последнее действие), «Удалить преподавателя» (происходит удаление выбранного педагога), «Изменить преподавателя» (можно изменить личные параметры преподавателя) и «Добавить преподавателя», при нажатии на которую появится новый экран, изображенный ниже на рисунке 20.

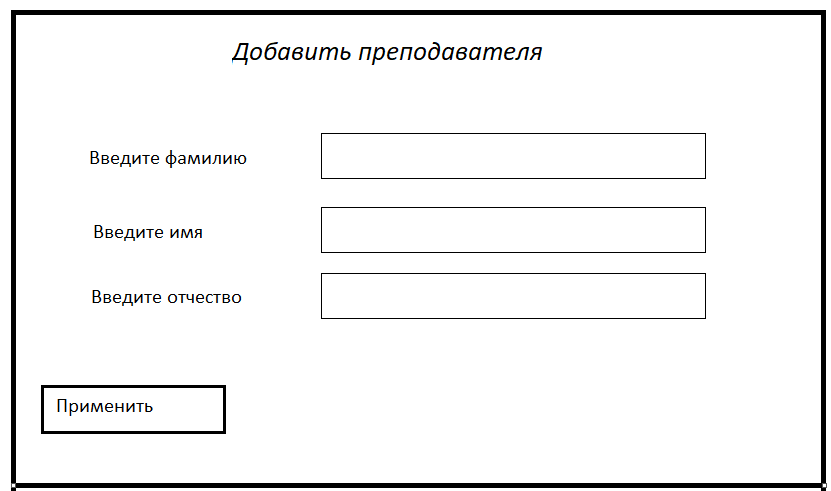


Рисунок 20

Появляются поля для заполнения (фамилия, имя, отчество) и кнопка «Применить», сохраняющая новые данные.

1. «Список учебных групп»

В программе появится экран, соответствующий рисунку 21. Выведется список учебных групп. Будут доступны пять кнопок: «Сохранить» (сохраняет изменения в БД), «Отменить» (отменяет последнее действие), «Удалить группу» (происходит удаление выбранной группы), «Редактировать» (можно изменить параметры группы) и «Добавить группу» (появляется новый экран по примеру рисунка 20 для добавления группы с полем для заполнения номера группы).

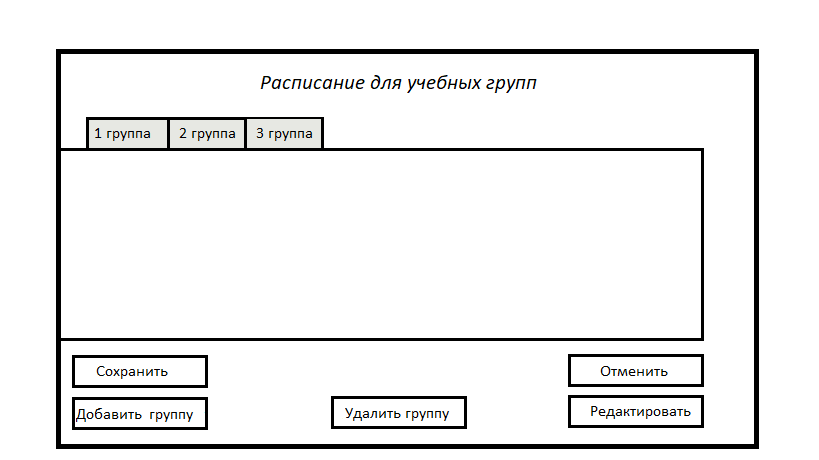


Рисунок 21

1. «Список предметов»

При выборе данной вкладки, появится экран, соответствующий рисунку 22. Выведется список предметов, которые преподаются на факультете. Будут доступны пять кнопок: «Сохранить» (сохраняет изменения в БД), «Отменить» (отменяет последнее действие), «Удалить предмет» (происходит удаление выбранного предмета), «Редактировать» (можно изменить параметры предмета) и «Добавление предмета» (появляется новый экран по примеру рисунка 20 для добавления предмета).

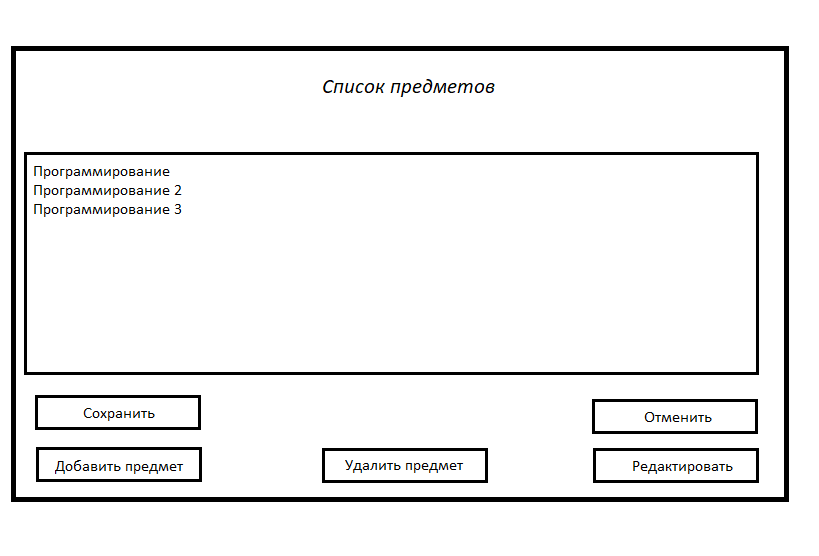


Рисунок 22

### 3.4 Средства реализации

В процессе разработки проекта были построены различные виды диаграмм с использованием такого программного продукта, как StarUML и Microsoft Visio.

Для написания приложения использовалась среда разработки Microsoft Visual Studio. Она основана на платформе .NET Framework.

Код проекта реализован на объектно-ориентированном языке программирования C#, который является языком с С-подобным синтаксисом.

Важно отметить, что мы использовали еще один программный продукт для написания проекта. В нашей задаче приходится работать с большим объемом данных и, как следствие, нам необходимо иметь хранилище для них. MySQL – это простая, надежная и эффективная система управления информацией, использующая традиционный язык SQL.

Для взаимодействия между программой и базой данных была использована технология ADO.NET