МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра вычислительной техники

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ**

Тема:

**«АВТОМАТИЗАЦИЯ УЧЕБНОГО ПРОЦЕССА»**

**Цель работы:**

В соответствии с темой курсового проекта спроектировать и реализовать web-приложение с использованием платформы ASP.NET.

Оглавление

[ВВЕДЕНИЕ 4](#_Toc90460630)

[1 ПРЕДМЕТНАЯ ОБЛАСТЬ 5](#_Toc90460631)

[1.1 Анализ предметной области 5](#_Toc90460632)

[1.2 Требования к проекту 6](#_Toc90460633)

[2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 8](#_Toc90460634)

[2.1 Средства проектирования 8](#_Toc90460635)

[2.2 Информационные объекты системы 9](#_Toc90460636)

[2.3 Иерархия классов 12](#_Toc90460637)

[3 ПРАКТИЧЕСКАЯ ЧАСТЬ 14](#_Toc90460638)

[3.1 Структура информационной системы 14](#_Toc90460639)

[3.2 Реализация системы 18](#_Toc90460640)

[4 ТЕСТИРОВАНИЕ 21](#_Toc90460641)

[ЗАКЛЮЧЕНИЕ 25](#_Toc90460642)

[ИСТОЧНИКИ 27](#_Toc90460643)

[ПРИЛОЖЕНИЕ 28](#_Toc90460644)

[ПРИЛОЖЕНИЕ А 28](#_Toc90460645)

[ПРИЛОЖЕНИЕ Б 30](#_Toc90460646)

[ПРИЛОЖЕНИЕ В 30](#_Toc90460647)

[ПРИЛОЖЕНИЕ Г 32](#_Toc90460648)

[ПРИЛОЖЕНИЕ Д 36](#_Toc90460649)

[ПРИЛОЖЕНИЕ Е 37](#_Toc90460650)

# ВВЕДЕНИЕ

Развитие технологий способствует переходу всех сфер жизни человека в цифровой формат. Это связано с тем, что с ростом объема хранимой и обрабатываемой информации эффективное управление такой системой без использования электронных устройств становится невозможным. Физические носители информации хороши только на сравнительно небольших количествах данных, хранимых в системе. Примерами небольших по размерам систем могут быть: классный журнал, телефонный справочник и другое. Однако при росте размеров системы, например, работа библиотек, больниц, интернет-магазинов, которые могут содержать в себе тысячи записей и постоянно обновляться, хранение данных и осуществление поиска необходимой информации или сортировка по выбранному критерию становятся невыполнимыми задачами. Для оптимизации процесса управления системой, не зависимо от природы данных в ней, применяется более удобное и эффективное решение - хранение данных в базах данных. Работа с базами данных позволяет автоматизировать хранение и обработку данных, а также обеспечить их безопасность. К безопасности данных относятся: целостность – данные не будут искажены или утеряны, доступность – данные предоставляются по запросу и конфиденциальность – доступ к данным должны иметь только те лица, которым этот доступ разрешен. В то время как физический носитель не может гарантировать обеспечение безопасности данных: физический носитель может быть поврежден (нарушение доступности), в то время как БД могут иметь резервные копии для восстановления данных в случае сбоя системы; в физическом носителе могут быть исправления, которые могут привести к дальнейшему искажению информации (нарушение целостности), в то время как хранение информации в БД позволяет однозначно определять значения полей таблиц; физический носитель может оказаться в руках злоумышленника и быть прочитан (нарушение конфиденциальности), в то время как БД позволяют ограничить доступ к системе через задание пользователю уникальных логина и пароля и ограничений его прав на взаимодействие с системой. Так же БД позволяет оптимизировать поиск информации и выдавать справки в зависимости от требований пользователя, что является несомненным преимуществом БД по сравнению с физическими носителями информации.

Благодаря автоматизации и оптимизации работы с данными системы базы данных укоренились во многих бизнес-процессах как коммерческих, так и некоммерческих организаций.

# 1 ПРЕДМЕТНАЯ ОБЛАСТЬ

## 1.1 Анализ предметной области

Высшее учебное заведение предоставляет услуги получения высшего образования для людей, успешно прошедших вступительные экзамены или поступивших на основе результатов Единых государственных экзаменов (ЕГЭ). Для эффективного управления деятельностью ВУЗа необходимо разработать автоматизированную систему, которая позволит работникам бюро расписаний оптимизировать процесс составления графика учебных занятий с назначением преподавателя на конкретную дисциплину и группу в соответствии с учебным планом.

Ввод в эксплуатацию автоматизированной системы «Автоматизация учебного процесса» позволит хранить и обрабатывать информацию в электронном виде об ученических группах, о действующих преподавателях с учетом их стажа работы в данном учебном заведении и о преподаваемых дисциплинах. На основе хранимых в системе данных каждый период (семестр, год) будет составляться расписание занятий в соответствии с требованиями к реализации учебного процесса.

С введением автоматизированной системы организация учебного процесса будет занимать у работников бюро расписаний меньше времени, а также позволит снизить количество ошибок, связанных с человеческим фактором. Так как деятельность ВУЗа реализуется большим преподавательским составом (более 500 человек), который обучает более 14000 студентов, то внедрение информационной системы «Автоматизация учебного процесса» является необходимым инструментом по обеспечению автоматизации и оптимизации всех бизнес-процессов, а также по достижению эффективности в управлении деятельностью ВУЗа.

## 1.2 Требования к проекту

Информационная система «Автоматизация учебного процесса» должна обеспечивать хранение данных о преподаваемых дисциплинах в ВУЗе, преподавателях, группах студентов с целью повышения эффективности работников бюро расписаний.

Система должна содержать следующую информацию:

* Список преподавателей (ФИО, кафедра, должность, номер\_преп, дата приема на работу);
* Список групп (название, код, семестр, специальность);
* Список дисциплин (код, номер\_преп, количество часов).

Система должна предоставлять следующие возможности:

* учитывать возможность чтения одной дисциплины несколькими преподавателями и наоборот;
* учитывать возможность изучения группой нескольких дисциплин и наоборот;
* иметь средства для ввода и модификации всех перечисленных данных;
* выдавать справки по преподавателю - какие дисциплины он читает, для каких групп и сколько часов;
* выдавать список преподавателей, читающих выбранную дисциплину по запросу;
* в целях безопасности соединение с БД должно открываться непосредственно перед выполнением запроса пользователя и после завершения операции должно закрываться сразу;
* работа с содержимым таблиц БД должна осуществляться только через управляющие элементы, расположенные на странице и доступные пользователю;
* запросы к БД для поиска значения по критерию должны быть реализованы в виде пользовательских функций. Разработчик вправе расширить список пользовательских функций при необходимости.

Требования к графическому интерфейсу:

* необходимо обеспечить пользователю графический интерфейс для взаимодействия с объектами системы;
* система должна быть реализована в виде многостраничного веб-приложения;
* наличие Главной страницы;
* все страницы веб-сайта должны быть оформлены в одном стиле;
* выбор веб-страниц должен осуществляться через меню;
* страницы должны быть разделены по функционалу (просмотр, добавление новых записей, удаление записей);
* необходимо предусмотреть поля для ввода информации и функциональные кнопки для выполнения операций.

Требования к базе данных:

При проектировании реляционной базы данных необходимо помнить об эффективности структуры БД.

Эффективная БД должна:

* обеспечивать быстрый доступ к данным в базе;
* минимизировать избыточность данных;
* обеспечивать целостность данных в базе. Изменение одних объектов должно автоматически изменять связанные с ними другие объекты.

Таблицы эффективной БД должны находиться в третьей нормальной форме.

# 2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 2.1 Средства проектирования

В качестве средств проектирования жизненного цикла (ЖЦ) информационной системы «Автоматизация учебного процесса» были выбраны следующие инструменты:

* Диаграмма классов предметной области «Автоматизация учебного процесса» - общая структура иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними;
* Диаграмма состояний для сущностей системы - диаграмма состояний из теории автоматов со стандартизированными условными обозначениями, определяет состояние системы в динамике (в разные моменты времени);
* Диаграмма прецедентов - диаграмма, отражающая отношения между акторами (действующими лицами, пользователями системы) и прецедентами (функции, которые они могут выполнять);
* Диаграмма компонент - показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами.

## 2.2 Информационные объекты системы

К информационным объектам системы относятся таблицы и пользовательские функции, хранимые в базе данных. База данных содержит в себе следующие таблицы:

* таблица «disciplina» - список преподаваемых дисциплин в ВУЗе;
* таблица «groups» - список ученических групп, обучающихся в ВУЗе;
* таблица «teacher» - список преподавателей, работающих в ВУЗе;
* таблица «raspisanie» - назначает преподавателя для конкретной группы;
* таблица «ucheb\_plan» - учебный план на заданный период, с указанием семестра и количества часов, выделенных на освоение дисциплины.

Структура таблиц «disciplina», «groups», «teacher», «raspisanie» и «ucheb\_plan» представлена в таблицах 1, 2, 3, 4 и 5 соответственно.

Таблица 1. Структура таблицы «disciplina».

|  |  |  |
| --- | --- | --- |
| **Название поля** | **Тип поля** | **Допустимость NULL** |
| id | integer | not null |
| name | character varying(60) | not null |
| **Индексы:** | | |
| "disciplina\_pkey" PRIMARY KEY, btree (id) | | |
| **Ссылки извне:** | | |
| TABLE "ucheb\_plan" CONSTRAINT "ucheb\_plan\_id\_disciplina\_fkey" FOREIGN KEY (id\_disciplina) REFERENCES disciplina(id) MATCH FULL ON DELETE CASCADE | | |

Таблица 2. Структура таблицы «groups».

|  |  |  |
| --- | --- | --- |
| **Название поля** | **Тип поля** | **Допустимость NULL** |
| id | integer | not null |
| name | character varying(60) | not null |
| specialization | character varying(60) | not null |
| **Индексы:** | | |
| "groups\_pkey" PRIMARY KEY, btree (id) | | |
| **Ссылки извне:** | | |
| TABLE "raspisanie" CONSTRAINT "raspisanie\_id\_group\_fkey" FOREIGN KEY (id\_group) REFERENCES groups(id) MATCH FULL ON DELETE CASCADE | | |

Таблица 3. Структура таблицы «teacher».

|  |  |  |
| --- | --- | --- |
| **Название поля** | **Тип поля** | **Допустимость NULL** |
| id | integer | not null |
| fio | character varying(60) | not null |
| kafedra | character varying(60) | not null |
| status | character varying(60) | not null |
| date\_hiring | date | not null |
| **Индексы:** | | |
| "teacher\_pkey" PRIMARY KEY, btree (id) | | |
| **Ссылки извне:** | | |
| TABLE "raspisanie" CONSTRAINT "raspisanie\_id\_teacher\_fkey" FOREIGN KEY (id\_teacher) REFERENCES teacher(id) MATCH FULL ON DELETE CASCADE | | |

Таблица 4. Структура таблицы «raspisanie».

|  |  |  |
| --- | --- | --- |
| **Название поля** | **Тип поля** | **Допустимость NULL** |
| id | integer | not null |
| id\_teacher | integer | not null |
| id\_group | integer | not null |
| **Индексы:** | | |
| "raspisanie\_pkey" PRIMARY KEY, btree (id) | | |
| **Ограничения внешнего ключа:** | | |
| "raspisanie\_id\_group\_fkey" FOREIGN KEY (id\_group) REFERENCES groups(id) MATCH FULL ON DELETE CASCADE  "raspisanie\_id\_teacher\_fkey" FOREIGN KEY (id\_teacher) REFERENCES teacher(id) MATCH FULL ON DELETE CASCADE | | |
| **Ссылки извне:** | | |
| TABLE "ucheb\_plan" CONSTRAINT "ucheb\_plan\_id \_fkey" FOREIGN KEY (id) REFERENCES raspisanie(id) MATCH FULL ON DELETE CASCADE | | |

Таблица 5. Структура таблицы «ucheb\_plan».

|  |  |  |
| --- | --- | --- |
| **Название поля** | **Тип поля** | **Допустимость NULL** |
| id | integer | not null |
| id\_disciplina | integer | not null |
| semestr | integer | not null |
| hours | integer | not null |
| **Индексы:** | | |
| "ucheb\_plan\_pkey" PRIMARY KEY, btree (id\_record) | | |
| **Ограничения внешнего ключа:** | | |
| "ucheb\_plan\_id\_disciplina\_fkey" FOREIGN KEY (id\_disciplina) REFERENCES disciplina(id) MATCH FULL ON DELETE CASCADE  "ucheb\_plan\_id\_record\_fkey" FOREIGN KEY (id) REFERENCES raspisanie(id) MATCH FULL ON DELETE CASCADE | | |

Связи между таблицами изображены на рисунке 1.

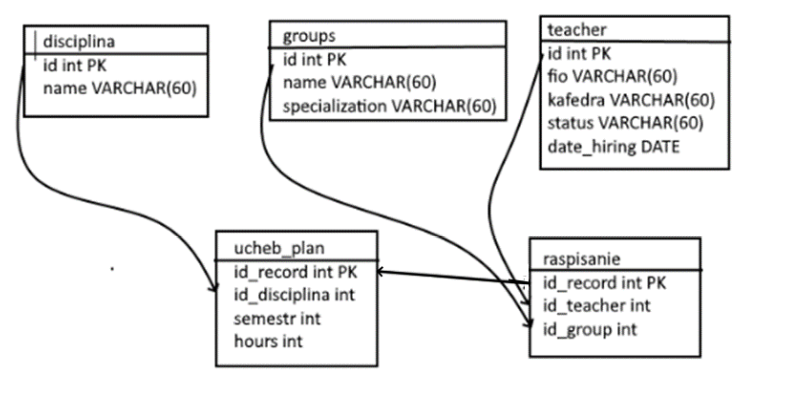


Рис. 1 - Связи между таблицами базы данных.

## 2.3 Иерархия классов

Диаграмма классов представлена на рисунке 2. Класс «пользователь системы» имеет доступ к работе со всеми классами системы. Классы «Преподаватель», «Группа» входят в состав класса «Расписание», который в свою очередь входит в состав класса «Учебный план». Так же класс «Дисциплина» составляет «Учебный план».

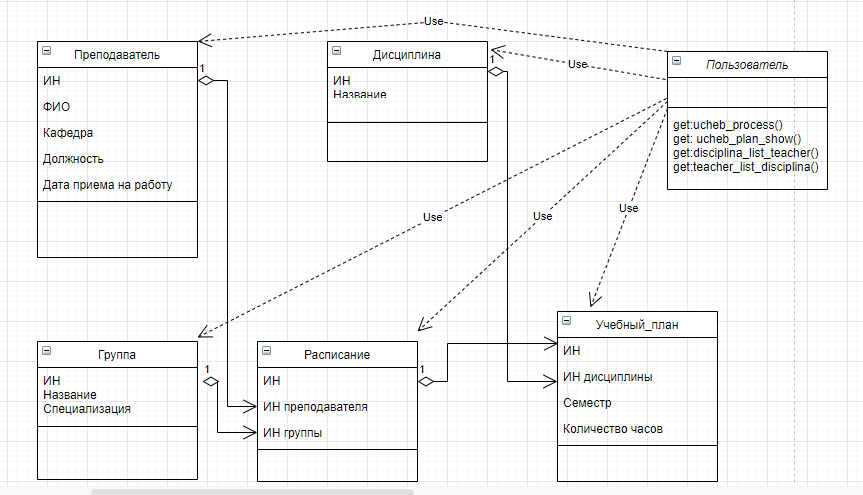


Рис. 2 – Диаграмма классов предметной области

Диаграмма состояний изображена на рисунке 3. Клиент выбирает функциональную страницу веб-сайта и вводит данные пока сервер ожидает запрос. Когда клиент заполнил форму и нажал кнопку «Ок» начинает формироваться SQL-запрос к БД. Когда SQL-запрос сформирован он передается серверу для выполнения. Когда запрос выполнен, то запрос уничтожается, и клиенту выводится результат сформированного им запроса в виде таблицы на веб-странице. Сервер переходит в состояние ожидания запроса.

Диаграмма прецедентов отображена на рисунке 4. Данная диаграмма содержит 3 актора: «Пользователь», «Сервер», «База данных». «Пользователь» взаимодействует с «Сервером» и ему доступны функции выбора страницы веб-сайта, ввода данных и получения ответа от «Сервера». «Сервер» формирует запрос и отправляет его «Базе данных». Взаимодействие с данными и выполнение запросов осуществляется на стороне «Базы данных». Таким образом результат запроса будет сформирован на стороне «Базы данных» и отправлен «Серверу», который в свою очередь осуществляет графическое отображение результата запроса «Пользователю».

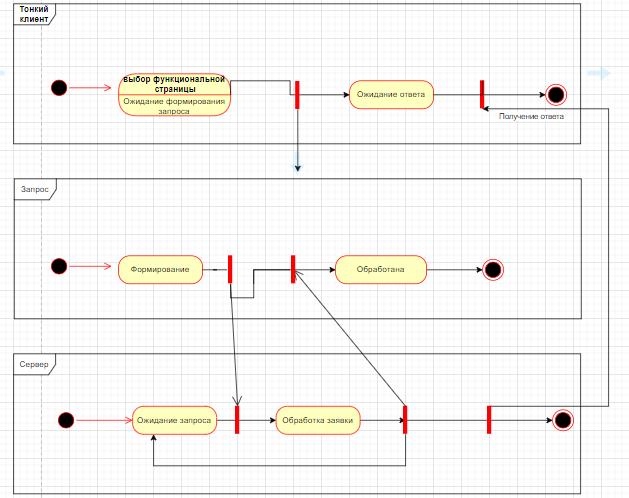


Рис. 3 – Диаграмма состояний

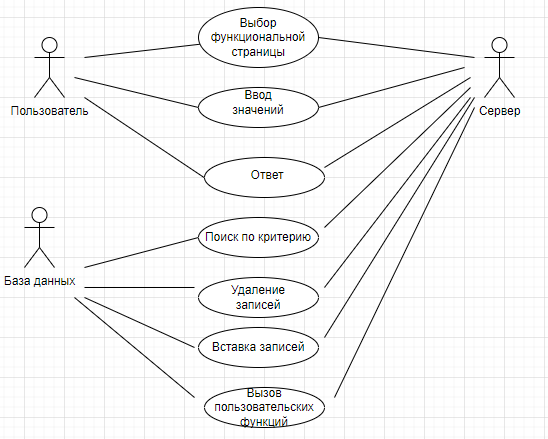


Рис. 4 – Диаграмма прецедентов

На рисунке 5 изображена диаграмма компонент реализации системы. Пользовательский интерфейс представлен в виде многостраничного сайта с разделением страниц по функционалу. В качестве языка реализации используется C#. Для пользовательского графического интерфейса используются каскадные таблицы стилей css. В качестве СУБД выбран PostgreSQL, так как он бесплатен, осуществляет функции хранения данных в виде реляционных БД и функции манипулирования ими посредством запросов SQL.

Реализация системы «Автоматизация учебного процесса» включает в себя:

Frontend – Пользовательский интерфейс представлен в виде тонкого клиента (веб-сайт).

Backend – Серверная часть и сервер БД.

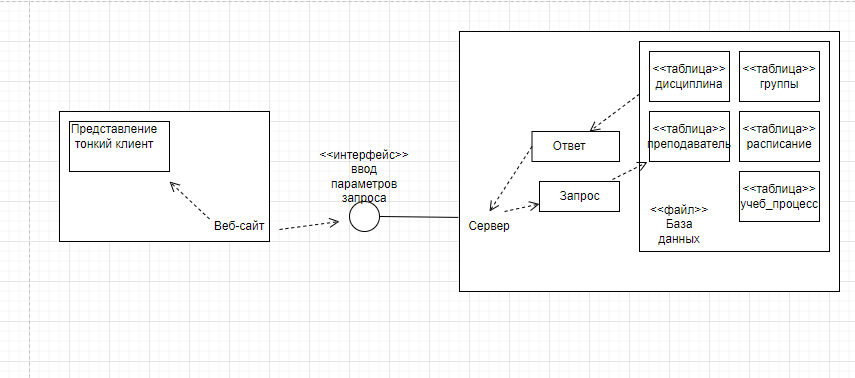


Рис. 5 – Диаграмма компонент реализации системы

# 3 ПРАКТИЧЕСКАЯ ЧАСТЬ

## 3.1 Структура информационной системы

Для хранения данных о деятельности ВУЗа была создана база данных «university», содержащая следующие таблицы «disciplina», «groups», «teacher», «raspisanie» и «ucheb\_plan». Подробная структура таблиц описана в разделе 2.2.

Заполним таблицы и выведем их содержимое (рисунки 6-10).

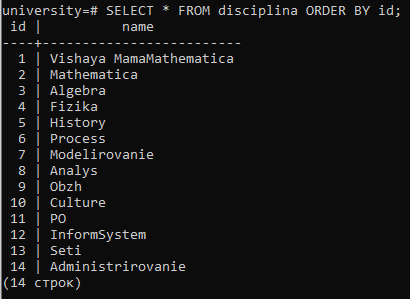


Рис. 6 – Содержимое таблицы «disciplina»

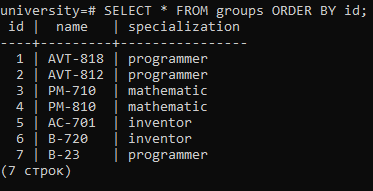


Рис. 7 – Содержимое таблицы «groups»

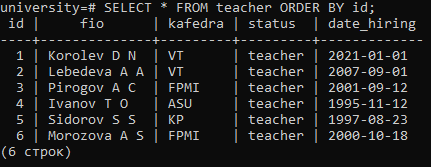


Рис. 8 – Содержимое таблицы «teacher»

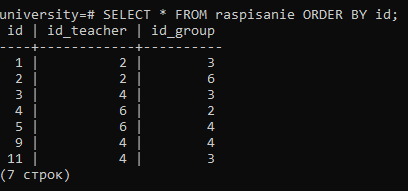


Рис. 9 – Содержимое таблицы «raspisanie»

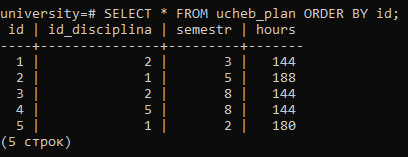


Рис. 10 – Содержимое таблицы «ucheb\_plan»

Для работы с данными БД создадим пользовательские функции:

1. Функция ucheb\_process() формирует расписание учебного процесса с подробной информацией (дисциплина, преподаватель, группа и тд)

CREATE or replace FUNCTION ucheb\_process() returns TABLE(id int, disciplina VARCHAR(60), semestr int, hours int, teacher VARCHAR(60), group\_name VARCHAR(60)) AS $$ BEGIN return query SELECT [ucheb\_plan.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fucheb_plan.id), [disciplina.name](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fdisciplina.name), ucheb\_plan.semestr, ucheb\_plan.hours, teacher.fio, [groups.name](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fgroups.name) FROM raspisanie JOIN teacher ON [raspisanie.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fraspisanie.id)\_teacher=[teacher.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fteacher.id) JOIN groups ON [raspisanie.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fraspisanie.id)\_group=[groups.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fgroups.id) JOIN ucheb\_plan ON [raspisanie.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fraspisanie.id)=[ucheb\_plan.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fucheb_plan.id) JOIN disciplina ON [ucheb\_plan.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fucheb_plan.id)\_disciplina=[disciplina.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fdisciplina.id) ORDER BY id; END; $$ language plpgsql;

2. Функция ucheb\_plan\_show() выводит учебный план в виде списка, который содержит в себе название дисциплины, количество часов и семестр.

CREATE or replace function ucheb\_plan\_show() returns table (id int, disciplina\_name VARCHAR(60), semestr int, hours int) AS $$ BEGIN return query SELECT ucheb\_plan.id, disciplina.name, ucheb\_plan.semestr, ucheb\_plan.hours FROM ucheb\_plan JOIN disciplina ON ucheb\_plan.id\_disciplina=disciplina.id ORDER BY ucheb\_plan.id; END; $$ language plpgsql;

3. Функция disciplina\_list\_teacher(id\_discpln int default 1) позволяет получить список преподавателей для выбранной дисциплины. Если значение не указано, то по умолчанию будет выведена информация для дисциплины с ИН=1.

CREATE or replace FUNCTION disciplina\_list\_teacher(id\_discpln int default 1) returns TABLE(id\_disciplina int, disciplina VARCHAR(60), semestr int, hours int, teacher VARCHAR(60), group\_name VARCHAR(60)) AS $$ BEGIN return query SELECT disciplina.id, disciplina.name, ucheb\_plan.semestr, ucheb\_plan.hours, teacher.fio, groups.name FROM raspisanie JOIN teacher ON raspisanie.id\_teacher=teacher.id JOIN groups ON raspisanie.id\_group=groups.id JOIN ucheb\_plan ON raspisanie.id=ucheb\_plan.id JOIN disciplina ON ucheb\_plan.id\_disciplina=disciplina.id WHERE disciplina.id=$1 ORDER BY id; END; $$ language plpgsql;

4. Функция teacher\_list\_disciplina(id\_teach int default 2) позволяет получить список читаемых дисциплин для выбранного преподавателя. Если параметр не задан, то по умолчанию будет выведена информация о всех дисциплинах, читаемых преподавателем с ИН=2.

CREATE or replace FUNCTION teacher\_list\_disciplina(id\_teach int default 2) returns TABLE(id\_teacher int, teacher VARCHAR(60), disciplina VARCHAR(60), semestr int, hours int, group\_name VARCHAR(60)) AS $$ BEGIN return query SELECT teacher.id,teacher.fio ,disciplina.name, ucheb\_plan.semestr, ucheb\_plan.hours, groups.name FROM raspisanie JOIN teacher ON raspisanie.id\_teacher=teacher.id JOIN groups ON raspisanie.id\_group=groups.id JOIN ucheb\_plan ON raspisanie.id=ucheb\_plan.id JOIN disciplina ON ucheb\_plan.id\_disciplina=disciplina.id WHERE teacher.id=$1 ORDER BY teacher.id; END; $$ language plpgsql;

Разработка информационной структуры завершена. При необходимости база данных может быть расширена новыми таблицами и пользовательскими функциями.

## 3.2 Реализация системы

Разработка системы осуществляется в среде разработки Visual Studio 2019 Enterprise. В качестве шаблона для реализации многостраничного веб-сайта использован «Веб-приложение ASP.NET (.NET Framework)». Данный проект реализован на платформе .NET Framework версии 4.7.2

Для обеспечения единого стиля страниц веб-сайта используется мастер-страница, на которой располагаются элементы, которые должны быть размещены на каждой странице. Мастер-страница имеет специальную область, называемую «placeholder», в которой будет отображаться содержимое страниц содержания, при этом область вне области «placeholder» остается неизменной. Такой подход позволяет создать шаблон, который упрощает внесение изменений в интерфейс веб-сайта: нет необходимости менять каждую страницу в отдельности, достаточно изменить только один файл мастер-страницы и изменения автоматически применятся ко всем связанным с этим шаблоном страницам содержания.

Для реализации графического интерфейса были также применены каскадные таблицы стилей, позволяющие установить внешний вид элементов (рисунок 11). Полный код файла StyleSheet1.css, который описывает внешнее представление сайта содержится в Приложении А.

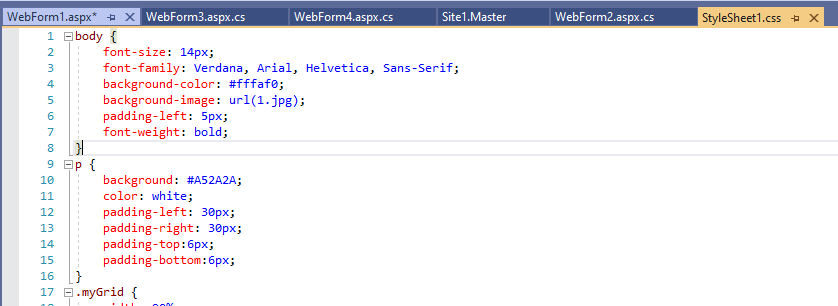


Рис. 11 – Описание представления сайта в файле StyleSheet1.css

При разработке мастер-страницы на форму было добавлено меню с элементами, реализующими навигацию по веб-сайту. Макет мастер-страницы представлен на рисунке 12.

При разработке страниц-содержания необходимо учитывать функционал страниц: наличие текстовых полей для ввода информации, кнопок для подтверждения выбора операции, средства для отображения таблиц с автоматическим обновлением в случае изменения. Макеты страниц содержания представлены на рисунках 13-16.

Для реализации системы «Автоматизация учебного процесса» для каждой страницы-содержания на форме были расположены необходимые элементы управления: TextBox – поля для ввода информации, Label – метка, для отображения справочной информации, ListBox – список, необходим для выбора таблицы или пользовательской функции , Button – кнопка, нажатие которой создает событие для обработки сервером, GridView – элемент управления, который предоставляет возможность отображения данных, представленных в табличном виде.

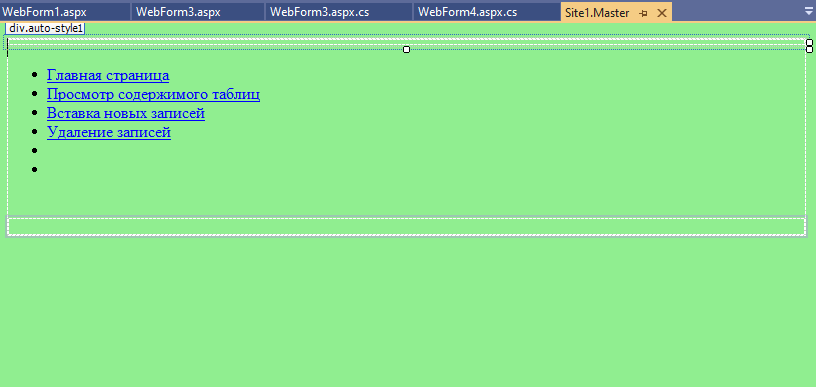


Рис. 12 – Содержимое мастер-страницы (конструктор)

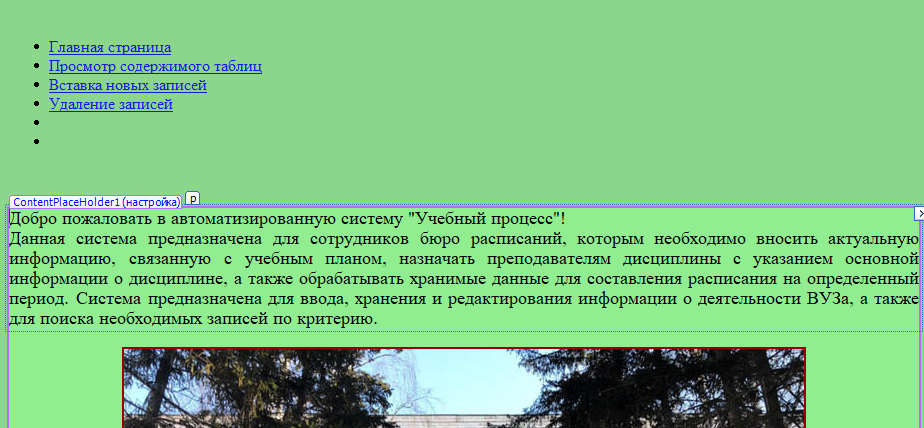


Рис. 12 – Содержимое главной страницы (конструктор)

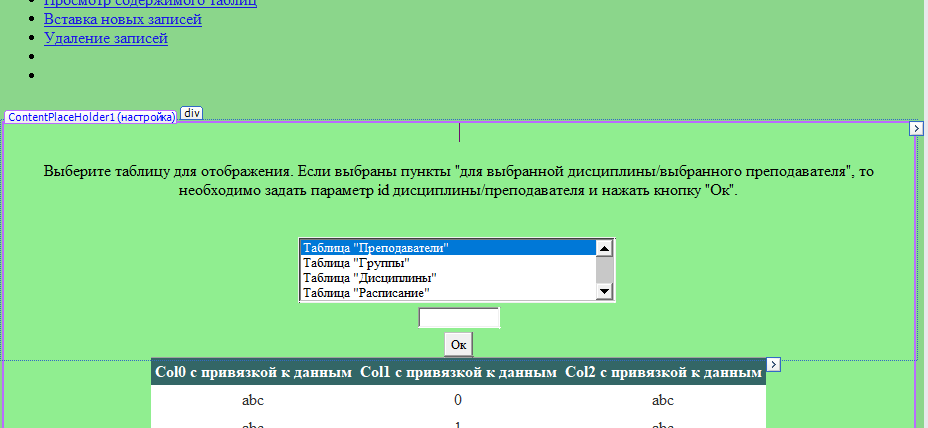


Рис. 13 – Содержимое страницы просмотра данных (конструктор)

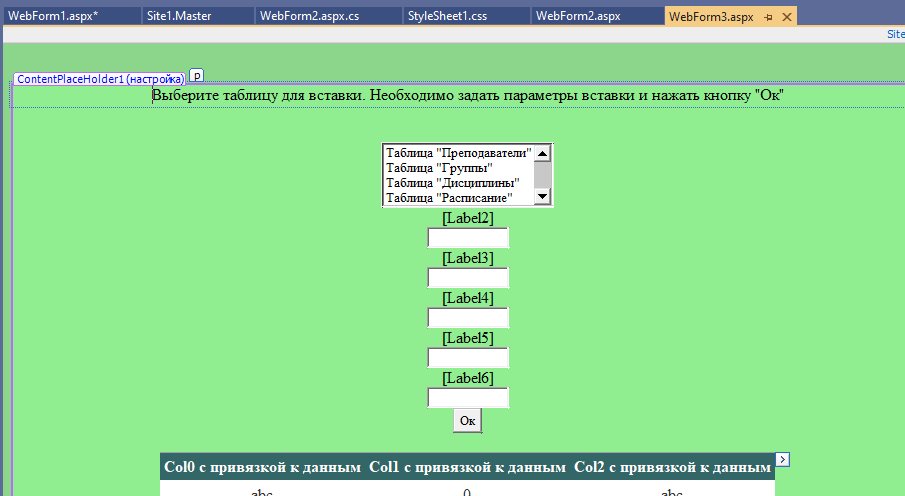


Рис. 14 – Содержимое страницы вставки данных (конструктор)

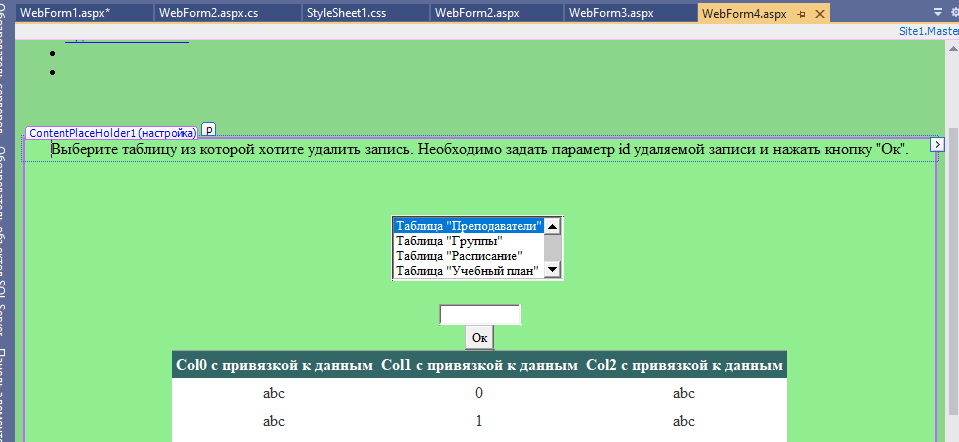


Рис. 15 – Содержимое страницы удаления данных (конструктор)

Так как были применены каскадные таблицы стилей, то итоговый вид разработанного веб-сайта можно увидеть непосредственно во время его запуска в веб-браузере.

# 4 ТЕСТИРОВАНИЕ

Запустим проект и протестируем систему «Автоматизация учебного процесса» (рисунок 16).

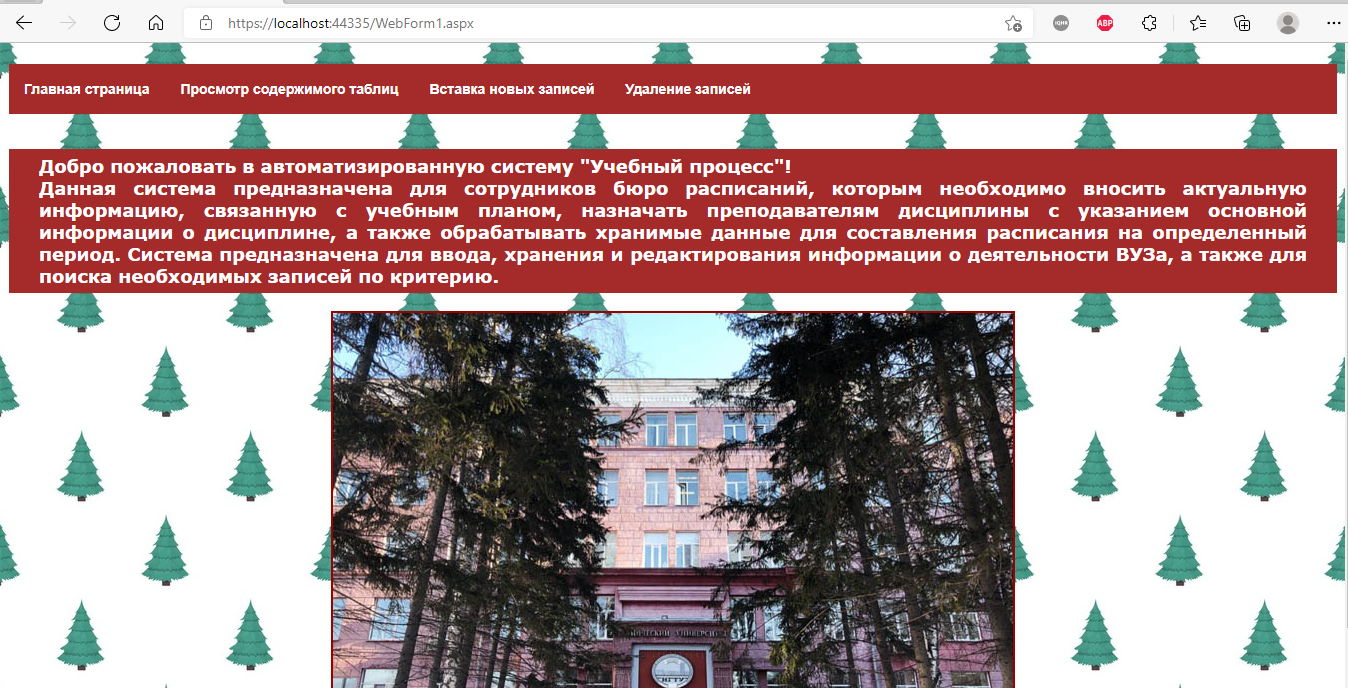


Рис 16. – Главная страница

Перейдем по кнопке меню «Просмотр содержимого таблиц» и просмотрим содержимое таблиц (рисунок 17).

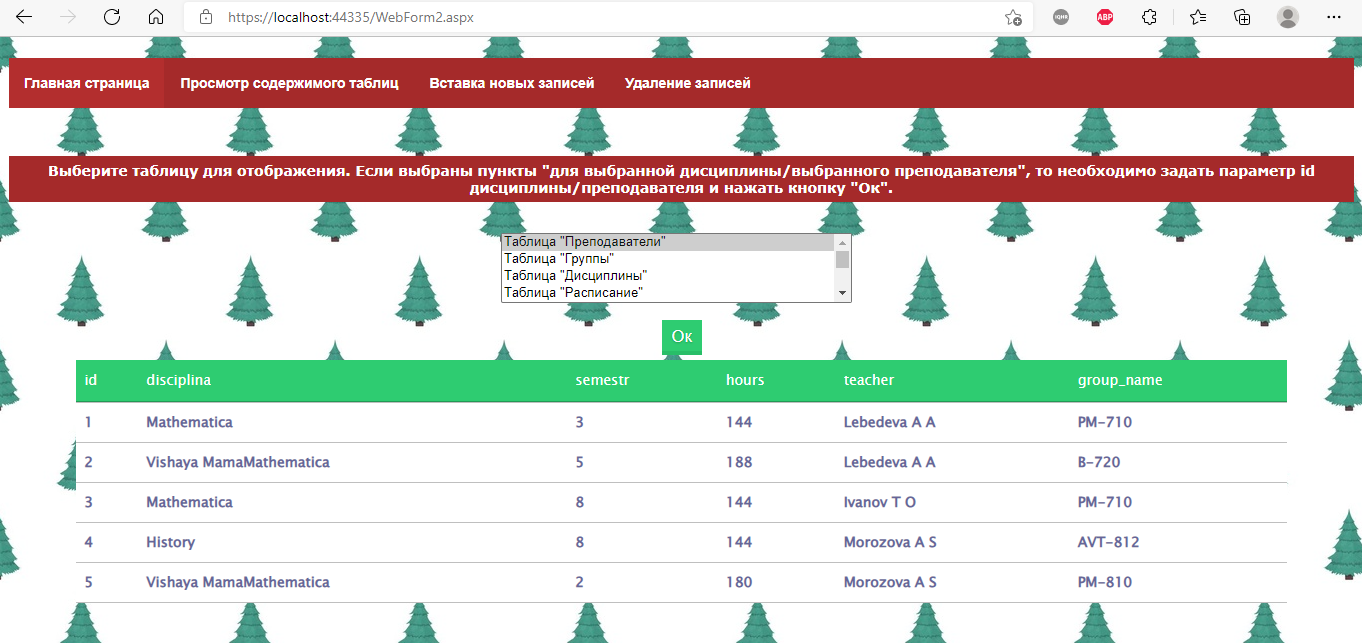


Рис. 17 – Страница просмотра данных

По умолчанию отображается таблица «Преподаватели», можно сменить таблицу для отображения или выбрать пользовательскую функцию. Если для использования функции требуется ввод параметров, то предусмотрено поле для ввода (рисунок 18).

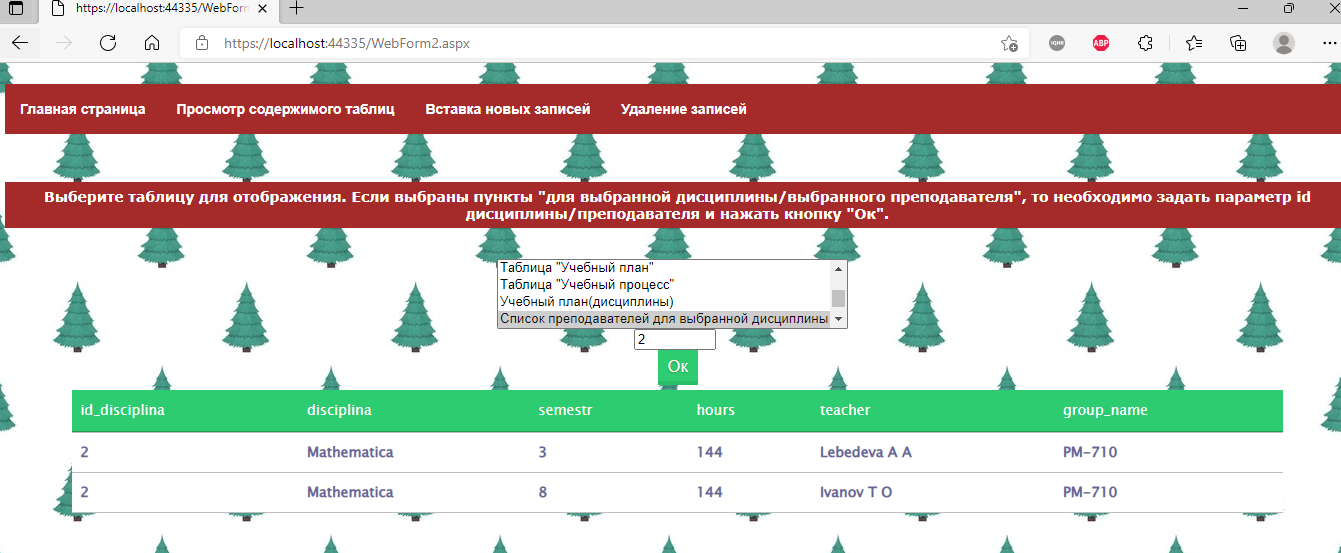


Рис. 18 – Результат работы функции вывода списка преподавателей для выбранной дисциплины (в данном случае дисциплины с id=2)

Получим расписание, составленное с учетом информации из всех таблиц БД с помощью пользовательской функции «Полное расписание(учебный процесс)» (ucheb\_process()). Функция не требует ввода параметров (рисунок 19).

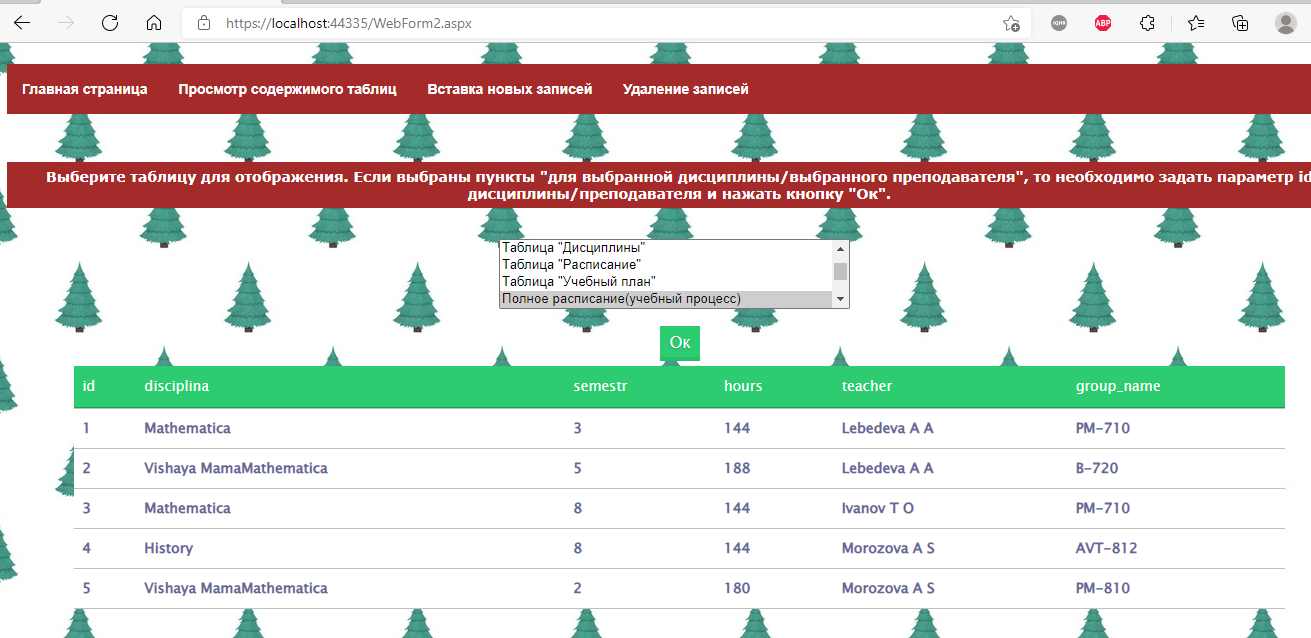


Рис. 19 – Вывод расписания с полной информацией

Вставим новую запись в таблицу. При смене таблицы из списка автоматически отображаются поля для ввода и подписи со справочной информацией (в соответствии с количеством столбцов таблицы). Для вставки была выбрана таблицы «Группы» (рисунок 20).

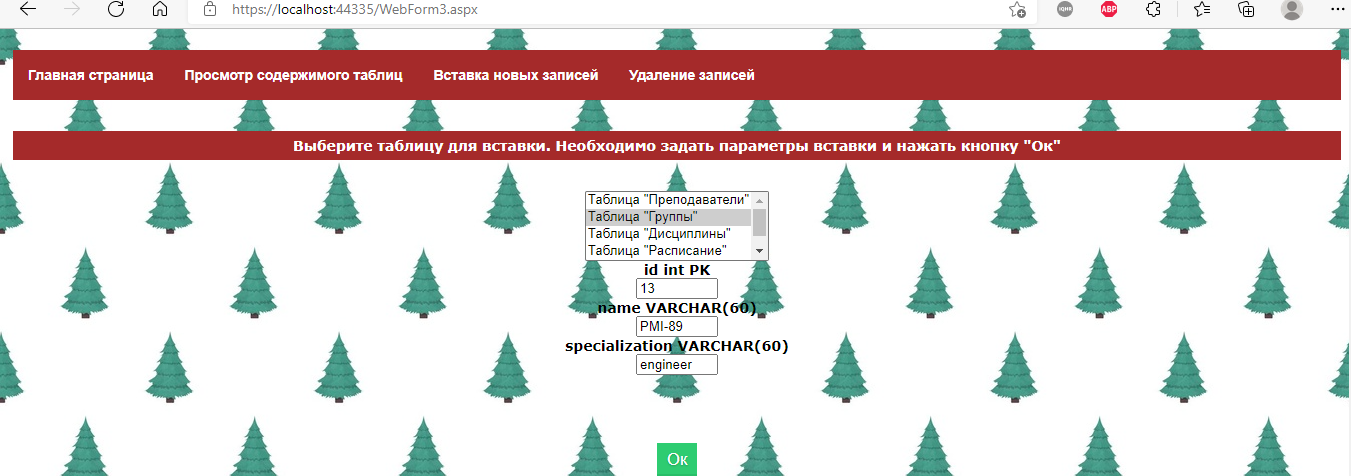


Рис. 20 – Ввод параметров для вставки

После нажатия на кнопку «Ок» происходит выполнение запроса на вставку к БД. Обновленное содержимое таблицы отображается на странице автоматически (рисунок 21).

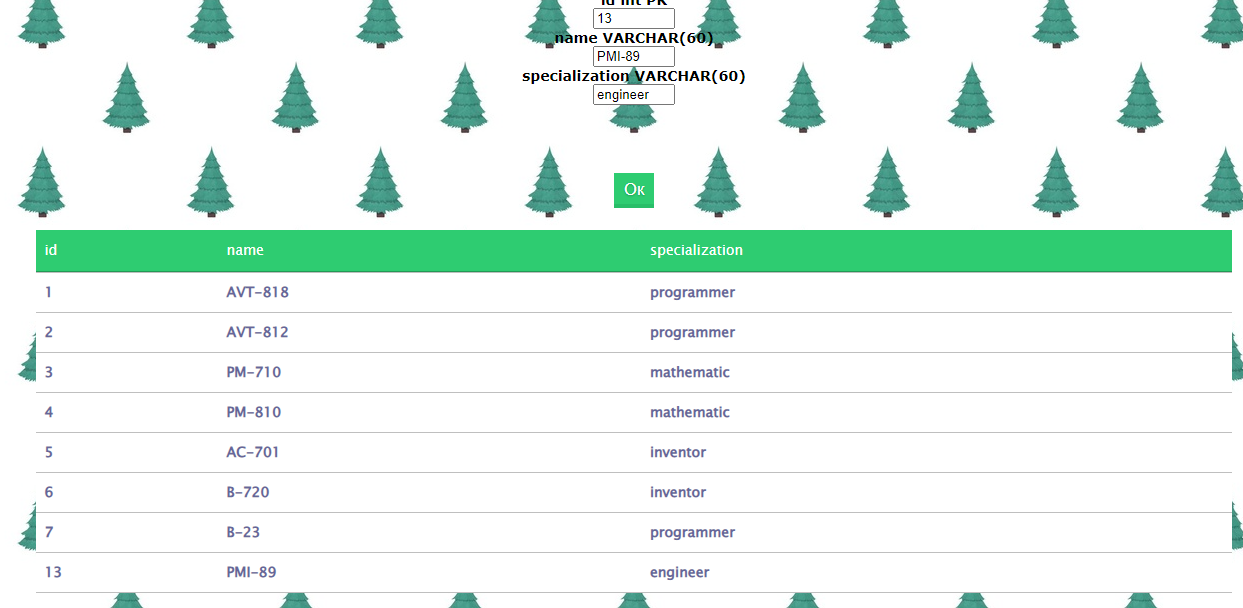


Рис. 21 – Результат работы функции вставки

Перейдем на страницу удаления данных и удалим запись из таблицы «Дисциплина» с id=2. Так как в системе предусмотрено каскадное удаление, то при удалении записи с id=2 из таблицы «Дисциплина» все записи других таблиц, ссылающихся на это же значение будут также удалены (рисунки 22 и 23). После удаления записи содержимое таблицы так же автоматически отображается на странице.

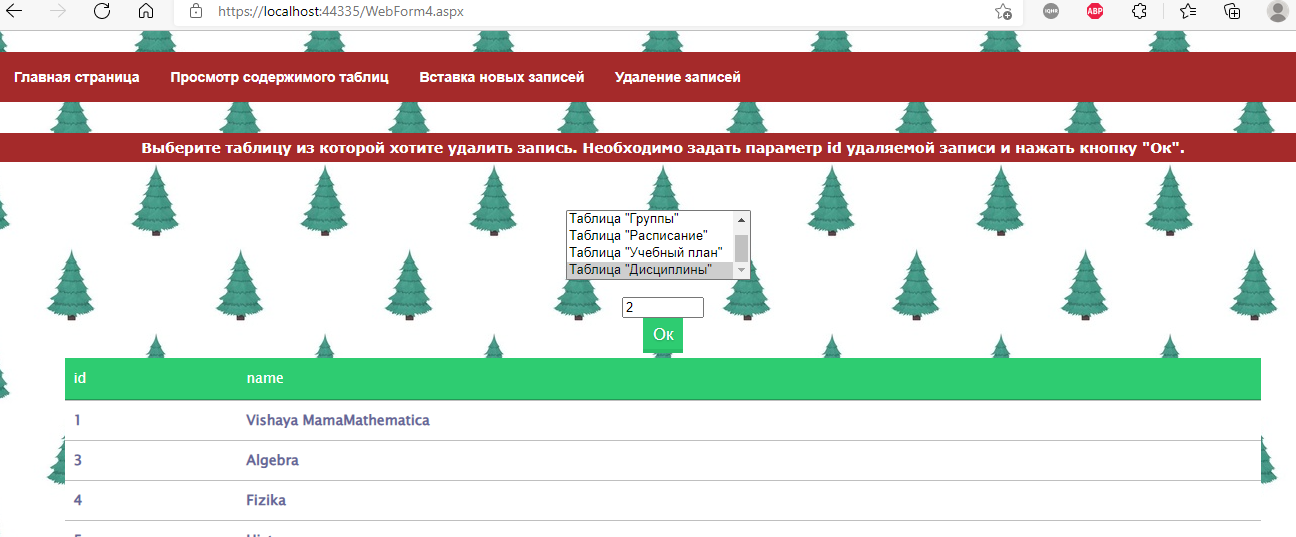


Рис. 22 – Содержимое таблицы «Дисциплина» после удаления записи с id=2

Убедимся, что в таблице «Полное расписание» так же отсутствуют записи с дисциплиной (id=2, name=mathematica)

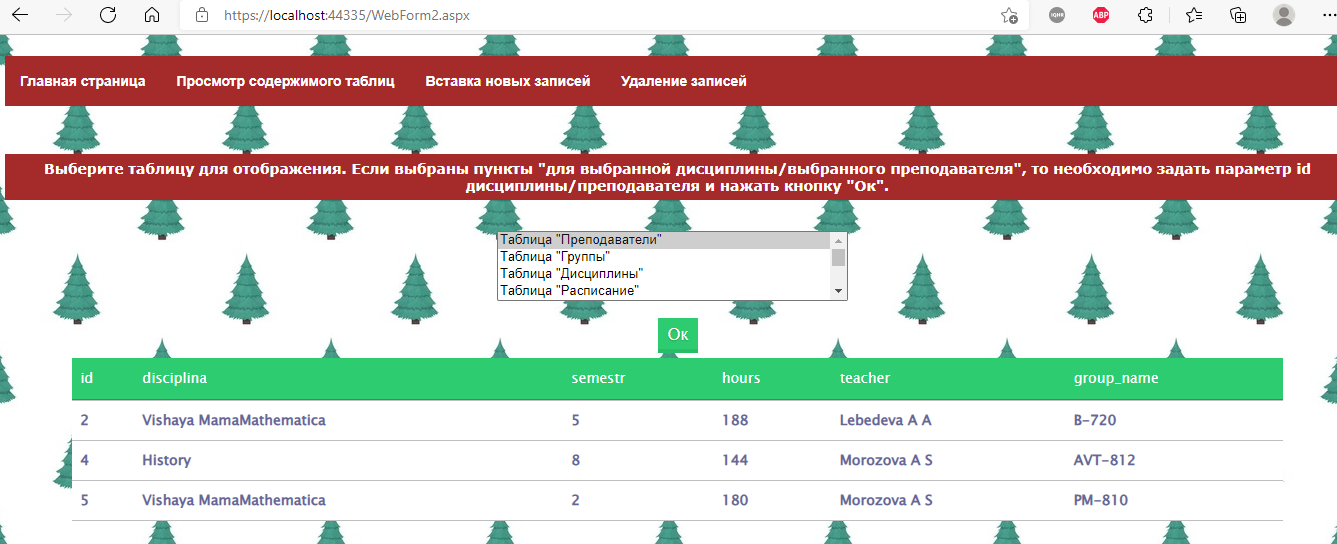


Рис. 23 – Содержимое связанной таблицы «Полное расписание» после удаления записи из таблицы «Дисциплина»

# ЗАКЛЮЧЕНИЕ

При внедрении системы «Автоматизация учебного процесса» управление деятельностью ВУЗа будет эффективнее, так как многие рутинные процессы стали автоматизированными. Получение полного расписания с помощью данной системы значительно снизит временные затраты работников бюро расписаний.

Технология разработки веб-приложений ASP.NET позволяет абстрагироваться от элементов программирования низкого уровня и использовать гибкий набор инструментов платформы .NET. Поэтому при создании веб-приложений разработчик имеет те же инструменты управления и технологии разработки, что и при создании десктопных приложений, сохраняя гибкость в принятии решений и получая возможность создавать полноценные приложения, которые могут быть развёрнуты на веб-сервере.

В ASP.NET возможно использование технологии для подключения к источникам данных ADO.NET. ADO.NET предоставляет единый интерфейс взаимодействия с источниками данных различной природы. Существуют реализации классов как для активного подключения к источникам данных (классы Connection, DataAdapter и т.д.), так и автономного хранения данных (DataSet, DataTable и др.). Таким образом, эта технология может применяться для взаимодействия с СУБД и их интеграции в прикладные приложения путём установления соединения для передачи данных между приложением и сервером баз данных.

Существует множество реализаций классов представления наборов данных. Некоторые из них (например, элемент управления DataGridView) могут самостоятельно подключаться к источникам данных и создавать адаптеры для непосредственного взаимодействия с источниками данных.

ADO.NET так же поддерживает работу с отсоединенными наборами данных, что необходимо при использовании масштабируемых веб-приложений. Для предотвращения утечки данных при каждом запросе к БД необходимо открывать соединение, а после обработки запроса закрывать соединение.

Гибкость разработки веб-приложения и возможность подключения источника данных независимо от поставщика делают технологию ASP.NET хорошим инструментом для разработки программного обеспечения информационных систем в виде веб-сайта.

# ИСПОЛЬЗОВАННЫЕ ИСТОЧНИКИ

# ПРИЛОЖЕНИЯ

## ПРИЛОЖЕНИЕ А