**Реферат**

Дипломный проект содержит !!!!! страницы пояснительной записки, !!! таблицы, !!!! рисунков, !!!! источников литературы, !!!!! приложения.

АВТОМАТИЗАЦИЯ,Учёт,ASP.NET, ВЕБ-СЕРВЕР, Деканат, Лабораторные работы, БАЗА ДАННЫХ

Целью данного дипломного проекта является создание приложения для автоматизации процесса учёта сдачи лабораторных работ и анализа успеваемости, включая систему авторизации с использованием механизмов ASP.NET.

Дипломный проект включает в себя пошаговое руководство по настройке веб-сервера IIS 7.0 и созданию самого веб-приложения.

В первой главе проводится обзор существующих технологий, удобных для написания дипломного проекта а также выбор из них тех, которые будут применяться впоследствии.

Во второй главе указаны данные о проектировании приложения, схемы базы данных и диаграммы работы

В третьей главе описываетсяструктура приложения, средства разработки, реализации основного функционала.

В четвёртой главе приведены скриншоты приложения с обзором его работы и руководством пользователя.

В пятой главе приводится расчет экономических параметров и себестоимость данного программного средства

В шестой главе данного дипломного проекта содержатся требования по охране труда и безопасности жизнедеятельности.

В заключении — анализ проделанной работы.

# Введение

В деканате информационных технологий Белорусского государственного технологического университета осуществляется учёт защиты лабораторных работ студентов, анализ текущей успеваемости студентов. Учёт защиты лабораторных работ производится путём получения от преподавателей отчетов по сданным студентами лабораторным работам в бумажном виде. Для создания сводной ведомости по успеваемости студентов группы, ответственный за успеваемость в конце каждого месяца (семестра) берет ведомость у заведующего отделения. Затем он, используя список студентов, список предметов и оценки по каждому предмету заполняет её.

Дипломный проект посвящён автоматизации процесса документооборота в деканате высшего учебного заведения.

**Актуальность темы:** часть полезного времени преподавателей тратится на составление отчёта по текущей успеваемости студентов, и, если заменить это взаимодействие на единую электронную систему, то производительность преподавателей улучшится, деканат будет иметь инструментарий для анализа успеваемости, для своевременного обнаружения проблем в какой-либо дисциплине, студенты будут иметь возможность просмотреть, какие лабораторные работы по каким предметам запланированы, и вовремя их выполнить.

**Цель дипломного проекта:** автоматизация процесса документооборота в деканате высшего учебного заведения.

Для достижения цели сформулированы следующие **задачи**.

1. Исследование процесса документооборота в деканате.
2. Проектирование архитектуры автоматизированной системы.
3. Проектирование программного обеспечения автоматизированной системы.
4. Проектирование логической схемы базы данных.
5. Разработка программного обеспечения.
6. Тестирование автоматизированной системы.

**Предмет исследования**:автоматизированная система документооборота деканата высшего учебного заведения.

**Практическая значимость:** внедрение автоматизированной системы позволит уменьшить трудозатраты по учёту успеваемости студентов.

Оглавление

[1. Введение 2](#_Toc480664742)

[**1.1.** **Аналитический обзор литературы** 5](#_Toc480664743)

[**1.1.1.** **СУБД** 5](#_Toc480664744)

[**1.1.2.** **Веб-сервер** 5](#_Toc480664745)

[1.1.3. Microsoft Visual Studio 6](#_Toc480664746)

[1.1.4. Microsoft SQL Server 6](#_Toc480664747)

[1.1.5. Oracle 12c 6](#_Toc480664748)

[**1.2.** **Стек технологий LAMP** 7](#_Toc480664749)

[**1.3.** **Стек технологий Java** 8](#_Toc480664750)

[**1.3.1. JEE** 8](#_Toc480664751)

[**1.3.2. Spring Framework** 9](#_Toc480664752)

[**1.4.** **Стек технологий MEAN** 14](#_Toc480664753)

[**1.5.** **Стек технологий .NET** 15](#_Toc480664754)

[**1.6.** **Патентный поиск и аналоги программного обеспечения** 16](#_Toc480664755)

[1.6.2. Модуль «Деканат» интегрированной автоматизированной информационной системы «ВУЗ» 19](#_Toc480664756)

[1.6.3. **Система управления учебным процессом «Магеллан»** 19](#_Toc480664757)

[1.6.4. **Система автоматизации учебного процесса «GS-Ведомости»** 20](#_Toc480664758)

[1.6.5. **Информационно-аналитическая система управления деканатом высшего учебного заведения** 20](#_Toc480664759)

[1.6.6. **Программный комплекс «Электронный документооборот деканата »** 21](#_Toc480664760)

[1.6.7. **Информационная система «Деканат »** 21](#_Toc480664761)

[1.6.8. **МОДУЛЬ "ДЕКАНАТ JTSQL"** 21](#_Toc480664762)

[**1.6.9.** **Программный комплекс «Деканат » КГПУ им. В.П. Астафьева** 22](#_Toc480664763)

[**1.7.** **Технологии и инструментарий для разработки автоматизированной системы «Электронный деканат»** 23](#_Toc480664764)

[1.7.1. OracleClient 23](#_Toc480664765)

[**1.7.2.** **OWIN** 23](#_Toc480664766)

[**1.7.3.** **EntityFramework** 24](#_Toc480664767)

[**1.7.4.** **Паттерн «репозиторий»** 25](#_Toc480664768)

[**1.7.5.** **bootstrap** 26](#_Toc480664769)

[**1.7.6.** **jQuerry** 27](#_Toc480664770)

[1.7.7. jQuery DataTables 27](#_Toc480664771)

[**1.8.** **Заключение** 28](#_Toc480664772)

[**2.** **Проектирование автоматизированной системы «Электронный деканат»** 31](#_Toc480664773)

[**2.1. Архитектура** 31](#_Toc480664774)

[**2.2. Диаграммы** 31](#_Toc480664775)

[**2.3. Структура базы данных** 32](#_Toc480664776)

[**2.4.** **Заключение** 37](#_Toc480664777)

[**3.** **Реализация программного обеспечения автоматизированной системы «Электронный деканат»** 37](#_Toc480664778)

[**3.1. Принципы построения по автоматизированной системы** 37](#_Toc480664779)

[**3.2. Контроллеры** 38](#_Toc480664780)

[**3.3. Модели** 39](#_Toc480664781)

[**3.4. Представления** 41](#_Toc480664782)

[3.4.1. Контроллер «Teacher»: 41](#_Toc480664783)

[3.4.2. Контроллер «Student»: 42](#_Toc480664784)

[3.4.3. Контроллер «Main»: 42](#_Toc480664785)

[3.4.4. Контроллер «Dekan»: 42](#_Toc480664786)

[3.4.5. Контроллер «Admin»: 43](#_Toc480664787)

[**3.5. Маршрутизация** 45](#_Toc480664788)

[**3.6. Заключение** 46](#_Toc480664789)

[**4.** **Описание контрольного примера** 46](#_Toc480664790)

[**4.1.** **Администратор** 46](#_Toc480664791)

[**4.2.** **Преподаватель** 54](#_Toc480664792)

[**4.3.** **Студент** 57](#_Toc480664793)

[**4.4.** **Деканат** 58](#_Toc480664794)

[**5. Экономический раздел** 60](#_Toc480664795)

[**6. Мероприятия по охране труда и безопасности жизнедеятельности** 60](#_Toc480664796)

[**7. Заключение** 60](#_Toc480664797)

[Список использованных источников 61](#_Toc480664798)

1. **Анализ и выбор технологий и инструментария для разработки автоматизированной системы «Электронный деканат»**
   1. **Аналитический обзор литературы**
      1. **СУБД**

Система управления базами данных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных [1].

Основные функции СУБД

* управление данными во внешней памяти (на дисках);
* управление данными в оперативной памяти с использованием дискового кэша;
* журнализация изменений, резервное копирование и восстановление базы данных после сбоев;
* поддержка языков БД (язык определения данных, язык манипулирования данными).

Обычно современная СУБД содержит следующие компоненты:

* ядро, которое отвечает за управление данными во внешней и оперативной памяти и журнализацию,
* процессор языка базы данных, обеспечивающий оптимизацию запросов на извлечение и изменение данных и создание, как правило, машинно-независимого исполняемого внутреннего кода,
* подсистему поддержки времени исполнения, которая интерпретирует программы манипуляции данными, создающие пользовательский интерфейс с СУБД
* а также сервисные программы (внешние утилиты), обеспечивающие ряд дополнительных возможностей по обслуживанию информационной системы.
  + 1. **Веб-сервер**

Понятие "Веб-сервер" может относиться как к железу, так и к программному обеспечению.

С точки зрения железа, `Веб-сервер` это компьютер, который хранит ресурсы сайта (HTML документы, CSS стили, JavaScript файлы и другое) и доставляет их на устройство конечного пользователя (веб-браузер и т.д.).

С точки зрения ПО, Веб-сервер включает в себя некоторые вещи, которые контролируют доступ веб-пользователей к размещенным на сервере файлам, это минимум HTTP сервера. HTTP сервер это часть ПО которая понимает URL’ы (веб-адреса) и HTTP (протокол который использует ваш браузер для просмотра веб-станиц).

Простыми словами, когда браузеру нужен файл размещенный на веб-сервере, браузер запрашивает его через HTTP. Когда запрос достигает нужного веб-сервера (железо), сервер HTTP (ПО) передает запрашиваемый документ обратно, также через HTTP [5].

### Microsoft Visual Studio

MicrosoftVisualStudio – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии WindowsForms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, WindowsMobile, Windows CE, .NET Framework, Xbox, WindowsPhone .NET CompactFramework и Silverlight [4].

В учреждении используется VisualStudioCommunity 2015, преимуществом которой является бесплатная лицензия на использование, а также полнофункциональная и расширяемая интегрированная среда разработки. На момент написания отчета актуальной версией данного ПО является VisualStudio 2017.

### Microsoft SQL Server

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка [6].

### Oracle 12c

**Oracle Database** - это объектно-реляционная система поддерживающая некоторые технологии, реализующие объектно-ориентированный подход, то есть обеспечивающих управление создания и использования баз данных [2].

**Ключевые возможности Oracle Database**

* Real Application Cluster (RAC) обеспечивает работу одного экземпляра базы данных на нескольких узлах grid, позволяя управлять нагрузкой и гибко масштабировать систему в случае необходимости.
* Automatic Storage Management (ASM) позволяет автоматически распределять данные между имеющимися ресурсами систем хранения данных, что повышает отказоустойчивость системы и снижает общую стоимость владения (TCO).
* Производительность. Oracle Database позволяет автоматически управлять уровнями сервиса и тиражировать эталонные конфигурации в рамках всей сети.
* Простые средства разработки. Новый инструмент разработки приложений HTML DB позволит простым пользователям создавать эффективные приложения для работы с базами данных в короткие сроки.
* Самоуправление. Специальные механизмы Oracle Database позволяют самостоятельно перераспределять нагрузку на систему, оптимизировать и корректировать SQL-запросы, выявлять и прогно- зировать ошибки.
* Большие базы данных. Теперь максимальный размер экземпляра базы данных Oracle может достигать 8 экзабайт.
* Недорогие серверные системы. Oracle Database может использовать недорогие однопроцессорные компьютеры или модульные системы из “серверов-лезвий”.
* В новой версии базы данных реализована поддержка переносимых табличных пространств, система управления потоками данных Oracle Streams и модель распределенных SQL-запросов. Для переноса существующих баз данных в среду Grid в них не потребуется вносить изменений, что позволяет быстро начать использовать все преимущества Oracle Database.
  1. **Стек технологий LAMP**

Рассмотрим свободно распространяемый и практически бесплатный набор программного обеспечения под названиям LAMP. Как правильно расшифровывается данная аббревиатура? Она состоит из 4-ех популярных технологий, которые состоят в данной связке: Linux — операционная система; Apache — веб-сервер; MySQL — СУБД; PHP — язык программирования, с помощью которого создают веб-ресурсы.

Данный набор ПО устанавливают на сервер для отображения динамических веб-сайтов и веб-приложений. Эта платформа позволяет установить (развернуть) большинство типов веб-сайтов и сетевого программного обеспечения на вашем сервере.

**Комбинация LAMP – это**:

1. Серверная **ОС Linux** для выполнения нужных задач;
2. веб-сервер **Apache**, для которого создано множество дополнительных модулей, которые решают вопрос совместной работы веб-сервера и сценариев, написанных на самых разных языках программирования;
3. серверный язык динамических сценариев **PHP** (либо Perl);
4. **СУБД** **MySQL**, которая демонстрирует отличную скорость выполнения SQL-запросов и идеально подходит для малых и средних проектов. MySQL работает на Unix и Windows, отличается особенным удобством в работе.

**Для чего может понадобиться LAMP?**

1. в случае необходимости окружения для веб-разработки и тестирования приложений, которые написаны для LAMP. Это как и для собственных приложений, так и для любых необходимых Вам CMS. Вы можете работать на Windows и на Ubuntu;
2. если Вам необходимо сделать на виртуальном VPS сервере или на выделенном сервере производительный веб-сервер для хостинга своих проектов;
3. если Вам нужен сервер для систем контроля версий;
4. для самообучения администрирования;
5. в экономических целях, для создания собственного сервера.

Как весомое преимущество данной сборки: она замечательно подходит для быстрого развертывания приложения, из-за простой конфигурации, но все же она дает мало функций в плане масштабируемости и изоляции компонентов.

Недостаток подобной сборки: приложение и база данных используют одни и те же ресурсы сервера (CPU, память, I/O и т.д.), что дает низкую производительность и затрудняет определение источника (приложение или база данных) этой проблемы. Так же есть помехи в осуществлении горизонтального масштабирования[7].

* 1. **Стек технологий Java**

**1.3.1. JEE**

Сервер приложений (англ. application server) — это программная платформа (фреймворк), предназначенная для эффективного исполнения процедур (программ, скриптов), на которых построены приложения. Сервер приложений действует как набор компонентов, доступных разработчику программного обеспечения через API (Интерфейс прикладного программирования), определённый самой платформой [8].

Для веб-приложений основная задача компонентов сервера — обеспечивать создание динамических страниц. Однако современные серверы приложений включают в себя и поддержку кластеризации, повышенную отказоустойчивость, балансировку нагрузки, позволяя таким образом разработчикам сфокусироваться только на реализации бизнес-логики.

В случае сервера приложений Java, сервер приложений ведёт себя как расширенная виртуальная машина для запуска приложений, прозрачно управляя соединениями с базой данных с одной стороны и соединениями с веб-клиентом с другой.

**Преимущества серверов приложений**

1. Целостность данных и кода

Выделяя бизнес логику на отдельный сервер, или на небольшое количество серверов, можно гарантировать обновления и улучшения приложений для всех пользователей. Отсутствует риск, что старая версия приложения получит доступ к данным или сможет их изменить старым несовместимым образом.

1. Централизованная настройка и управление

Изменения в настройках приложения, таких как изменение сервера базы данных или системных настроек, могут производиться централизованно.

1. Безопасность

Сервер приложений действует как центральная точка, используя которую, поставщики сервисов могут управлять доступом к данным и частям самих приложений, что считается преимуществом защиты. Её наличие позволяет переместить ответственность за аутентификацию с потенциально небезопасного уровня клиента на уровень сервера приложений, при этом дополнительно скрывая уровень базы данных.

1. Поддержка транзакций

Транзакция представляет собой единицу активности, во время которой большое число изменений ресурсов (в одном или различных источниках) может быть выполнено атомарно (как неделимая единица работы). Конечные пользователи при этом могут выиграть от стандартизованного поведения системы, от уменьшения времени на разработку и от снижения стоимости. В то время как сервер приложений выполняет массу нужного генерирования кода, разработчики могут сфокусироваться на бизнес-логике.

Примеры реализации

* + 1. Под сервером приложений в случае Java EE подразумевается комплекс программ, реализующих концепцию Java EE и позволяющих запускать в себе Java EE приложения. К классу серверов приложений относятся такие продукты как Sun GlassFish, IBM WebSphere, RedHat JBoss Application Server, Apple WebObjects (англ.), Oracle Weblogic Server и др.

1. Zope, развитый сервер web-приложений.
2. Терминальные серверы, например поставляемые компанией Citrix

**1.3.2. Spring Framework**

Spring Framework (или коротко Spring) — универсальный фреймворк с открытым исходным кодом для Java-платформы. Также существует форк для платформы .NET Framework, названный Spring.NET [10].

Первая версия была написана Родом Джонсоном, который впервые опубликовал её вместе с изданием своей книги «Expert One-on-One Java EE Design and Development

Фреймворк был впервые выпущен под лицензией Apache 2.0 license в июне 2003 года. Первый стабильный релиз 1.0 был выпущен в марте 2004. Spring 2.0 был выпущен в октябре 2006, Spring 2.5 — в ноябре 2007, Spring 3.0 в декабре 2009, и Spring 3.1 в декабре 2011. Текущая версия — 4.3.7.

Несмотря на то, что Spring не обеспечивал какую-либо конкретную модель программирования, он стал широко распространённым в Java-сообществе главным образом как альтернатива и замена модели Enterprise JavaBeans. Spring предоставляет бо́льшую свободу Java-разработчикам в проектировании; кроме того, он предоставляет хорошо документированные и лёгкие в использовании средства решения проблем, возникающих при создании приложений корпоративного масштаба.

Между тем, особенности ядра Spring применимы в любом Java-приложении, и существует множество расширений и усовершенствований для построения веб-приложений на Java Enterprise платформе. По этим причинам Spring приобрёл большую популярность и признаётся разработчиками как стратегически важный фреймворк.

Spring обеспечивает решения многих задач, с которыми сталкиваются Java-разработчики и организации, которые хотят создать информационную систему, основанную на платформе Java. Из-за широкой функциональности трудно определить наиболее значимые структурные элементы, из которых он состоит. Spring не всецело связан с платформой Java Enterprise, несмотря на его масштабную интеграцию с ней, что является важной причиной его популярности.

Spring, вероятно, наиболее известен как источник расширений (features), нужных для эффективной разработки сложных бизнес-приложений вне тяжеловесных программных моделей, которые исторически были доминирующими в промышленности. Ещё одно его достоинство в том, что он ввел ранее неиспользуемые функциональные возможности в сегодняшние господствующие методы разработки, даже вне платформы Java.

Этот фреймворк предлагает последовательную модель и делает её применимой к большинству типов приложений, которые уже созданы на основе платформы Java. Считается, что Spring реализует модель разработки, основанную на лучших стандартах индустрии, и делает её доступной во многих областях Java.

**Модули**

Spring может быть рассмотрен как коллекция меньших фреймворков или фреймворков во фреймворке. Большинство этих фреймворков может работать независимо друг от друга, однако они обеспечивают большую функциональность при совместном их использовании. Эти фреймворки делятся на структурные элементы типовых комплексных приложений:

* Inversion of Control-контейнер: конфигурирование компонентов приложений и управление жизненным циклом Java-объектов.
* Фреймворк аспектно-ориентированного программирования: работает с функциональностью, которая не может быть реализована возможностями объектно-ориентированного программирования на Java без потерь.
* Фреймворк доступа к данным: работает с системами управления реляционными базами данных на Java-платформе, используя JDBC- и ORM-средства и обеспечивая решения задач, которые повторяются в большом числе Java-based environments.
* Фреймворк управления транзакциями: координация различных API управления транзакциями и инструментарий настраиваемого управления транзакциями для объектов Java.
* Фреймворк MVC: каркас, основанный на HTTP и сервлетах, предоставляющий множество возможностей для расширения и настройки (customization).
* Фреймворк удалённого доступа: конфигурируемая передача Java-объектов через сеть в стиле RPC, поддерживающая RMI, CORBA, HTTP-based протоколы, включая web-сервисы (SOAP).
* Фреймворк аутентификации и авторизации: конфигурируемый инструментарий процессов аутентификации и авторизации, поддерживающий много популярных и ставших индустриальными стандартами протоколов, инструментов, практик через дочерний проект Spring Security (ранее известный как Acegi).
* Фреймворк удалённого управления: конфигурируемое представление и управление Java-объектами для локальной или удалённой конфигурации с помощью JMX.
* Фреймворк работы с сообщениями: конфигурируемая регистрация объектов-слушателей сообщений для прозрачной обработки сообщений из очереди сообщений с помощью JMS, улучшенная отправка сообщений по стандарту JMS API.
* Тестирование: каркас, поддерживающий классы для написания модульных и интеграционных тестов.

**Inversion of Control**

Центральной частью Spring является контейнер Inversion of Control, который предоставляет средства конфигурирования и управления объектами Java с помощью рефлексии. Контейнер отвечает за управление жизненным циклом объекта: создание объектов, вызов методов инициализации и конфигурирование объектов путём связывания их между собой.

Объекты, создаваемые контейнером, также называются управляемыми объектами (beans). Обычно конфигурирование контейнера осуществляется путём загрузки XML-файлов, содержащих определение bean’ов и предоставляющих информацию, необходимую для создания bean’ов.

Объекты могут быть получены одним из двух способов:

* Поиск зависимости — шаблон проектирования, в котором вызывающий объект запрашивает у объекта-контейнера экземпляр объекта с определённым именем или определённого типа.
* Внедрение зависимости — шаблон проектирования, в котором контейнер передает экземпляры объектов по их имени другим объектам с помощью конструктора, свойства или фабричного метода.

**MVC**

Spring имеет собственную MVC-платформу веб-приложений, которая не была первоначально запланирована. Разработчики Spring решили написать её как реакцию на то, что они восприняли как неудачность конструкции (тогда) популярного Apache Struts, а также других доступных веб-фреймворков. В частности, по их мнению, было недостаточным разделение между слоями представления и обработки запросов, а также между слоем обработки запросов и моделью.[4]

Класс DispatcherServlet является основным контроллером фрэймворка и отвечает за делегирование управления различным интерфейсам, на всех этапах выполнения HTTP-запроса. Об этих интерфейсах следует сказать более подробно.

Как и Struts, Spring MVC является фреймворком, ориентированным на запросы. В нем определены стратегические интерфейсы для всех функций современной запросно-ориентированной системы. Цель каждого интерфейса — быть простым и ясным, чтобы пользователям было легко его заново имплементировать, если они того пожелают. MVC прокладывает путь к более чистому front-end-коду. Все интерфейсы тесно связаны с Servlet API. Эта связь рассматривается некоторыми как неспособность разработчиков Spring предложить для веб-приложений абстракцию более высокого уровня. Однако эта связь оставляет особенности Servlet API доступными для разработчиков, облегчая все же работу с ним. Наиболее важные интерфейсы, определенные Spring MVC, перечислены ниже:

* HandlerMapping: выбор класса и его метода, которые должны обработать данный входящий запрос на основе любого внутреннего или внешнего для этого запроса атрибута или состояния.
* HandlerAdapter: вызов и выполнение выбранного метода обработки входящего запроса.
* Controller: включен между Моделью (Model) и Представлением (View). Управляет процессом преобразования входящих запросов в адекватные ответы. Действует как ворота, направляющие всю поступающую информацию. Переключает поток информации из модели в представление и обратно.
* View: ответственно за возвращение ответа клиенту в виде текстов и изображений. Некоторые запросы могут идти прямо во View, не заходя в Model; другие проходят через все три слоя.
* ViewResolver: выбор, какое именно View должно быть показано клиенту.
* HandlerInterceptor: перехват входящих запросов. Сопоставим, но не эквивалентен сервлет-фильтрам (использование не является обязательным и не контролируется DispatcherServlet-ом).
* LocaleResolver: получение и, возможно, сохранение локальных настроек (язык, страна, часовой пояс) пользователя.
* MultipartResolver: обеспечивает Upload — загрузку на сервер локальных файлов клиента.

Spring MVC предоставляет разработчику следующие возможности:

* Ясное и прозрачное разделение между слоями в MVC и запросах.
* Стратегия интерфейсов — каждый интерфейс делает только свою часть работы.
* Интерфейс всегда может быть заменен альтернативной реализацией.
* Интерфейсы тесно связаны с Servlet API.
* Высокий уровень абстракции для веб-приложений.
* В веб-приложениях можно использовать различные части Spring, а не только Spring MVC.

**Доступ к данным**

Spring предоставляет свой слой доступа к базам данных и поддерживает все популярные СУБД.

JDBC, iBatis / MyBatis, Hibernate, JDO, JPA, Oracle TopLink, Apache OJB, Apache Cayenne и т. д.

Для всех этих фреймворков Spring предоставляет такие особенности:

* Управление ресурсами — автоматическое получение и освобождение ресурсов базы данных
* Обработка исключений — перевод исключений при доступе к данным в исключения Spring-а
* Транзакционность — прозрачные транзакции в операциях с данными
* Распаковка ресурсов — получение объектов базы данных из пула соединений
* Абстракция для обработки BLOB и CLOB

**Управление транзакциями**

Фреймворк управления транзакциями в Spring привносит механизм абстракций для платформы Java. Основные возможности этих абстракций:

* работа с локальными и глобальными транзакциями
* работа с вложенными транзакциями
* работа с точками сохранения в транзакциях

**«Соглашение прежде конфигурации» в разработке приложения**

Spring Roo — фреймворк, поддерживающий принцип «соглашение прежде конфигурации» для быстрого создания бизнес-приложений.

Может использоваться со:

* Spring Framework
* Spring Security
* Spring Web Flow

**Интеграция**

Spring Integration — фреймворк для JEE, который предоставляет функции, необходимые для отправки сообщений или для построения событийно-ориентированной архитектуры.

* роутеры
* трансформеры
* адаптеры для интеграции с другими технологиями и системами (HTTP, AMQP, JMS, XMPP, SMTP, IMAP, FTP (FTPS/SFTP), файловые системы и т. д.)
* фильтры
* активаторы сервисов
* аудит и управление
  1. **Стек технологий MEAN**

В последнее время в веб-сообществе много говорят о новом стеке MEAN: MongoDB, Express, AngularJS, Node.js. Стек технологий MEAN отражает современный подход к веб-разработке: когда на каждом уровне приложения, от клиента до сервера и персистентности, применяется один и тот же язык (JavaScript).

MEAN — это больше, чем простая перестановка начальных букв и новых технологий. Смещение базовой платформы с ОС (Linux) к среде исполнения JavaScript (Node.js) несет с собой независимость от ОС: Node.js работает на Windows® и OS X так же, как и на Linux.

Node.js заменяет Apache из стека LAMP. Но Node.js — это гораздо больше, чем просто веб-сервер. На самом деле готовое приложение не развертывается на отдельном веб-сервере; вместо этого сам веб-сервер включается в приложение и автоматически устанавливается в составе стека MEAN. В результате процесс развертывания значительно упрощается, так как требуемая версия веб-сервера явно определена вместе с остальными зависимостями времени выполнения.

Переход от традиционной базы данных, такой как MySQL, к бессхемному, документо-ориентированному NoSQL-хранилищу, такому как MongoDB, представляет собой фундаментальный сдвиг в стратегии персистенции. Программист тратит меньше времени на написание операторов SQL и больше — на написание функций map/reduce на JavaScript. При этом исключаются огромные пласты логики преобразования, так как MongoDB изначально выдает формат JavaScript Object Notation (JSON). В результате предельно упрощается написание веб-сервисов REST.

Но главный сдвиг между LAMP и MEAN заключается в переходе от традиционного генерирования страниц на стороне сервера к ориентации на одностраничные приложения (SPA) на стороне клиента. Express позволяет управлять и маршрутизацией / генерацией страниц на стороне сервера, но теперь — благодаря AngularJS — упор делается на представления на стороне клиента. Эта перемена означает не просто перенос ваших артефактов модель-представление-контроллер (MVC) с сервера в клиентское устройство. Это также скачок от менталитета синхронности к менталитету, носящему событийно-управляемый, принципиально асинхронный характер. И, пожалуй, самое главное, — это движение от странично-ориентированных приложений к компонентно-ориентированным.

Стек MEAN не «заточен» на мобильные приложения — AngularJS одинаково хорошо работает на настольных компьютерах и ноутбуках, смартфонах и планшетах и даже на смарт-телевизорах, — но он и не относится к мобильным устройствам как к гражданам второго сорта. И тестирование больше не откладывается на потом: с помощью платформ тестирования мирового класса, таких как MochaJS, JasmineJS и KarmaJS, можно писать тщательные и всеобъемлющие наборы тестов для своих MEAN-приложений [9].

* 1. **Стек технологий .NET**

.NET Framework — это программная платформа, выпущенная компанией Microsoft, которая подходит для разных языков программирования. ASP.NET — технология создания веб-приложений и веб-сервисов от компании Microsoft. Она является составной частью платформы Microsoft. NET и развитием более старой технологии Microsoft ASP.

Считается, что платформа. NET Framework явилась ответом компании Microsoft на набравшую к тому времени большую популярность платформу Java. ASP.NET основывается на Common Language Runtime: разработчики могут писать код для ASP.NET, используя практически любые языки программирования, некоторые из которых входят в комплект. NET Framework (C#, Visual Basic.NET и JScript. NET), а другие могут быть установлены дополнительно (IronRuby, IronPython, PHP, Perl, Smalltalk, Haskell и др.). Тем не менее, нередко используется не вполне корректный термин «.NET-программист», который, видимо, характеризует разработчика на любом языке программирования, который может быть использован в. NET-проекте.

Некоторые особенности ASP.NET:

* Компилируемый код выполняется быстрее, а большинство ошибок отлавливается ещё на стадии разработки.
* Расширяемый набор элементов управления и библиотек классов, ускоряющий разработку.
* Возможность кэширования всей страницы, её частей или данных, используемых на странице.
* Возможность разделения визуальной части и бизнес-логики по разным файлам, есть возможность выделять часто используемые шаблоны пользовательских элементов управления, таких как меню сайта, наличие master-страниц для задания шаблонов оформления, поддержка AJAX (расширение ASP.NET AJAX).
* Расширяемые модели событий, обработки запросов и серверных элементов управления.
* Поддержка CRUD-операций при работе с таблицами через GridView.
* Возможно создание веб-приложений, которые реализуют шаблон Model-View-Controller (ASP.NET MVC Framework).

.NET достаточно широко распространён в сфере разработки внутрикорпоративных программных продуктов, но в веб-разработке всё же встречается относительно редко, как и другие программные продукты корпорации Microsoft. Поэтому найти разработчиков для веб-проекта бывает достаточно непросто. Использование. NET «тянет» за собой покупку и иного ПО от корпорации Microsoft (серверной ОС, СУБД и т.п.). Технология достаточно дорогая в разработке и сопровождении: кроме затрат на покупку лицензий на необходимое ПО существенный вклад в бюджет проектов вносят высокие зарплаты разработчиков. [11].

## **Патентный поиск и аналоги программного обеспечения**

Патентные исследования – это исследования технического уровня и тенденций развития объектов техники, их патентоспособности, патентной чистоты, конкурентоспособности на основе патентной и иной информации.

По своему характеру и содержанию патентные исследования относятся к прикладным научно-исследовательским работам. Патентные исследования проводятся как в виде самостоятельной научно-исследовательской работы, так и в составе работ субъекта хозяйствования.

Патентный поиск - это процесс отбора соответствующих запросу документов или сведений по одному или нескольким признакам из массива патентных документов или данных, при этом осуществляется процесс поиска из множества документов и текстов только тех, которые соответствуют теме или предмету запроса.

Предмет поиска определяют исходя из конкретных задач патентных исследований, особенностей объекта (устройство, способ, вещество), а так же из того, какие его элементы, параметры, свойства и другие характеристики предполагается исследовать.

Основные виды патентного поиска: предметный, именной (или фирменный), нумерационный, патентов-аналогов. Выбор типа патентного поиска определяется как необходимой глубиной поиска и временными ограничениями, так и поисковыми возможностями лица или организации, проводящих поиск.

Предметный поиск – является основным и чаще всего применяемым. При этом виде поиска формулируется техническая задача (предмет поиска), выбором рубрики (рубрик) патентной классификации ограничивается тематическая область поиска, выявляются и анализируются патентные материалы, относящиеся к ней за необходимый временной промежуток.

Именной (или фирменный) – поиск проводится в том случае, когда известны имя (имена) изобретателя (изобретателей) или названия фирм. Этот вид поиска дополняет предметный поиск.

Нумерационный поиск – осуществляется, когда известен номер охранного документа и по его номеру требуется узнать другие данные об изобретении, полезной модели, промышленном образце.

Поиск патентов-аналогов – проводится для выявления патентов, выданных в какой-либо стране и запатентованных затем в других странах, т.е. выявляются патенты, выданные в каждой стране патентования на одно и то же изобретение. К этому виду поиска целесообразно прибегать, если найден патент, интересующий специалиста, на редком языке, а патенты-аналоги позволяют ознакомиться с описанием этого изобретения на других более доступных языках. Кроме того, этот вид поиска дополняет предметный и проводится на стадии подробного ознакомления с полными описаниями к патентам.

Среди патентных исследований следует особо выделить исследование патентной чистоты. Патентная чистота – это юридическое свойство технического изделия или технологии, заключающееся в том, что они могут использоваться в определенной стране без нарушения действующих на ее территории патентов.

Установление патентной чистоты продукции позволяет избежать ответственности за возможное нарушение прав, вытекающих из патентов, действующих на территории той страны, в которой предполагается реализация данной продукции. Исследование на патентную чистоту заключается в отыскании действующих в данной стране патентов, имеющих отношение к продукции, их анализе, а также изучении обстоятельств, которые могли бы способствовать беспрепятственной реализации продукции на данном рынке [12].

В соответствии с темой дипломного проекта был проведен патентный поиск в области программных систем для анализа успеваемости студентов, и автоматизации отчётности в деканате высшего учебного заведения[19].

В качестве аналогов можно выделить:

1. Автоматизация процесса составления деканатом рейтинга студентов
2. Модуль «Деканат» интегрированной автоматизированной информационной системы «ВУЗ»
3. Система управления учебным процессом «Магеллан»
4. Система автоматизации учебного процесса «GS-Ведомости»
5. Информационно-аналитическая система управления деканатом высшего учебного заведения
6. Программный комплекс «Электронный документооборот деканата»
7. Информационная система «Деканат »
8. МОДУЛЬ "ДЕКАНАТ JTSQL"
9. Программный комплекс «Деканат » КГПУ им. В.П. Астафьева
   * 1. **Автоматизация процесса составления деканатом рейтинга студентов**

Язык программирования**:** Visual Basic for Applications.

Номер регистрации (свидетельства): 2015610242.

Дата регистрации: 12.01.2015.

Реферат: Программа служит для автоматизации процесса составления сотрудниками деканата сводных ведомостей успеваемости студентов факультета. Программа подготавливает формы для отчетности сотрудниками деканата о рейтинговых оценках студентов за определенные учебные периоды. Формы отчетности содержат списки групп, студентов, учебных дисциплин и рейтинговых оценок. На основании получаемых данных производится анализ, определяются проблемные места учебного процесса, формируются сводные таблицы, производятся сравнения различных структурных подразделений.

### Модуль «Деканат» интегрированной автоматизированной информационной системы «ВУЗ»

Язык программирования Java, framework Struts.

Номер регистрации (свидетельства): 2014611450.

Дата регистрации: 03.02.2014.

Реферат: Программа предназначена для ввода, хранения и модификации данных о студентах с целью дальнейшей обработки в ходе процесса обучения, а также для автоматизированного создания, хранения и редактирования документов, используемых сотрудниками деканатов в отчетах, для упрощения процедуры мониторинга успеваемости студентов. Область применения - программа может быть использована в высших учебных заведениях как средство мониторинга учебного процесса и автоматизации документооборота сотрудниками деканатов и отдела кадров. Программа является «клиент-серверным» приложением, выполняющим следующие функции: оператор системы. Функции оператора системы: ввод и корректировка личной карточки студента (учебная информация, личные данные, сведения о работе и образовании и прочее); поиск личных карточек по различным критериям, создание списков студентов по группам; создание и печать экзаменационных ведомостей; создание, хранение, редактирование и мониторинг оценок студентов в период сессии (по экзаменам, зачетам, курсовым работам и т.д.); оформление продления сессии студенту, создание и редактирование информации о дипломниках; получение информации о расписании занятий; формирование и учет приказов о перемещении и отчислении студентов; формирование, просмотр, удаление и повторная печать отчетов с данными; формирование бланков, ведомостей, справок, форм и сводок (по специальностям, факультетам и т.д.); формирование и просмотр выписок из дипломов; генерация текущей отчетной документации с целью упрощения контроля за успеваемостью студентов.

* + 1. **Система управления учебным процессом «Магеллан»**

Язык программирования Delphi, С++, PHP, JavaScript, P-SQL, SQL.

Номер регистрации (свидетельства): 2014615939.

Дата регистрации: 06.06.2014.

Реферат: Программа предназначена для автоматизации учебного процесса и административной деятельности в учебных заведениях среднего, высшего и дополнительного профессионального образования. Основные функциональные возможности программы: автоматизация работы сотрудников приёмной комиссии, деканата , отдела кадров, учебно-методического управления, кафедр, отдела составления расписаний (позволяет формировать расписание занятий в ручном и полностью автоматическом режимах), научных библиотек, общежитий; автоматизация работы, связанной с ведением НИОКР, договорной работы, ведения электронного документооборота, планирования деятельности сотрудников, проведения автоматизированного тестирования.

* + 1. **Система автоматизации учебного процесса «GS-Ведомости»**

Язык программирования Delphi, С++, PHP, SQL, P-SQL.

Номер регистрации (свидетельства): 2014610950.

Дата регистрации: 21.01.2014.

Реферат: Программа предназначена для автоматизации работы учебных заведений среднего профессионального, высшего профессионального и дополнительного профессионального образования. Функциональные возможности: автоматизация работы приёмной комиссии; подача заявлений абитуриентами в электронном виде; автоматизация работы учебной части и деканатов ; ведение базы данных по учащимся и их успеваемости; автоматизация работы отдела кадров образовательного учреждения; ведение базы данных по сотрудникам; автоматизация работы учебно -методического управления; автоматизация работы кафедр и предметно-цикловых комиссий; обеспечивает возможность распределения нагрузки по преподавателям; автоматизация процесса создания расписания занятий в учебном заведении ; автоматизация прохождение учащимися тестов для проверки уровня знаний; автоматизация работы с контрагентами учебного заведения ; реализует ведение базы данных по заключаемым договорам. Программа реализует простейшие функции системы электронного документооборота . Программа автоматизирует процесс формирования планов работы сотрудников учебного заведения , распределение заданий и поручений; реализует сетевое унифицированное хранилище файлов и документов в произвольных форматах; реализует возможность быстрого формирования и просмотра различных статистических и аналитических отчётов руководителями учебного заведения ; автоматизирует работу библиотеки учебного заведения , ведения библиотечного фонда; автоматизирует работу общежитий учебного заведения , вселения и выселения постояльцев; реализует функциональность по настройке системы, управления пользователями системы, назначения прав доступа; реализует функциональность по созданию и редактированию шаблонов печатных документов, формируемых из системы, а также созданию и редактированию статистических и аналитических отчётов, формируемых из системы; реализует внутрисистемный чат для общения пользователей между собой.

* + 1. **Информационно-аналитическая система управления деканатом высшего учебного заведения**

Язык программирования С#.

Номер регистрации (свидетельства): 2015660921.

Дата регистрации: 13.10.2015.

Реферат: Программа предназначена для управления деканатом высшего учебного заведения с реализацией ведения справочной информации и документов, на основе которых происходит составление статистических и аналитических отчетов об успеваемости и движении контингента учащихся.

* + 1. **Программный комплекс «Электронный документооборот деканата »**

Язык программирования РНР.

Номер регистрации (свидетельства): 2016614900.

Дата регистрации: 11.05.2016.

Реферат: Программный комплекс предназначен для автоматизации документооборота деканата вуза. Программный комплекс выполняет следующие основные функции: отражение основных документов, касающихся учебного процесса; формирование, вывод на экран и печать отчетов; регистрация входящей, исходящей и внутренней документаций; предварительное рассмотрение и распределение документов; оповещения о сроках выполнения того или иного приказа; контроль исполнения приказов.

* + 1. **Информационная система «Деканат »**

Язык программирования Object Pascal (в среде Delphi).

Номер регистрации (свидетельства): 2014616013.

Дата регистрации: 09.06.2014

Реферат: Программа предназначена для развития электронного документооборота вуза, связанного с успеваемостью студентов. Программа состоит из совокупности модулей, обеспечивающих ведение в базе данных информации об успешности освоения студентами образовательных программ, автоматического формирования ведомостей, направлений, личных карточек студентов. Специальный модуль, предназначенный для автоматического формирования академических справок, приложений к диплому, европейских приложений к диплому, основан на системе шаблонов, обеспечивающих гибкую настройку на различные принтеры. В программу включен модуль, автоматически формирующий итоговую справочную информацию по успеваемости контингента студентов в различных разрезах по внутренним запросам и данные для внешних мониторингов.

* + 1. **МОДУЛЬ "ДЕКАНАТ JTSQL"**

Язык программирования JAVA, XML, JSON

Номер регистрации (свидетельства): 2017612930

Дата регистрации: 06.03.2017

Реферат: Программа предназначена для автоматизированного сбора и обработки данных деканатов ВУЗа. Она представляет собой «тонкий» клиент, позволяющий создавать, открывать, редактировать и сохранять в базу данных информацию о студентах и о цикле обучения, а также в форматы JSON, XML. В состав программы входят компоненты по вводу и редактированию карточки студента, модуль миграции данных зачисленных абитуриентов в группы обучения студентов и компоненты ведения документооборота секретариатом деканатов. Программа поддерживает многопользовательский режим работы. Программа поддерживает следующие роли: наблюдатель, оператор, администратор.

* + 1. **Программный комплекс «Деканат » КГПУ им. В.П. Астафьева**

Язык программирования Python, Odoo ERP

Номер регистрации (свидетельства): 2016612958

Дата регистрации: 14.03.2016

Реферат: Программа предназначена для автоматизированного сбора и обработки данных деканатов ВУЗа. Она представляет собой «тонкий» клиент, позволяющий создавать, открывать, редактировать и сохранять в базу данных информацию о студентах и о цикле обучения, а также в форматы JSON, XML. В состав программы входят компоненты по вводу и редактированию карточки студента, модуль миграции данных зачисленных абитуриентов в группы обучения студентов и компоненты ведения документооборота секретариатом деканатов. Программа поддерживает многопользовательский режим работы. Программа поддерживает следующие роли: наблюдатель, оператор, администратор.

## **Технологии и инструментарий для разработки автоматизированной системы «Электронный деканат»**

### OracleClient

Пакет из репозитория пакетов Nuget, предназначен для взаимодействия с СУБД Oracle. Устанавливается командой в npm консоли:

*Install-Package Oracle.OracleClient*

Основной класс, с помощью которого устанавливается соединение, это OracleConnection из пространства имён Oracle.ManagedDataAccess.Client. Из экземпляра объекта этого класса можно создать объект OracleCommand, он используется для выполнения как sql запросов, так и вызова хранимых процедур. Поддерживает вставку параметров в плейсхолдеры, которые помечаются двоеточиями в строке sql запроса, и позднее, заполняются методом Add, в который передаются объекты класс OracleParameter, инициализированные необходимыми значениями. Это помогает обеспечить защиту от sql-иньекций, исключая конкатенацию строк. Если выполнение OracleCommand подразумевает получение результирующего набора данных, то вызывается метод ExecuteReader, если получение данных не подразумевается, то запрос выполняется методом ExecuteNonQuery. ExecuteReader возвращает объект OracleDataReader, который представляет из себя итератор. Считать из него данные можно как по названию столбца результирующего набора, так и по его номеру, для перемещения к следующей записи используется метод Read.

* + 1. **OWIN**

Вдохновленные Rack, несколько разработчиков .NET решили создать свою абстракцию между веб сервером и компонентами инфраструктуры. Разработчики преследовали две цели:

* Новые компоненты будут просты в разработке и использовании.
* Приложения могут быть легко переносимы с хоста на хост, возможно даже на разных платформах

И вот что вышло.

OWIN (The Open Web Interface for .NET) — это спецификация определяющая интерфейс и описывающая взаимодействие между всеми компонентами.

Основой работы OWIN является словарь IDictionary<string, object>, который используется для получения доступа к HTTP запросам, заголовку запроса и окружению хоста. Все ключи описаны в спецификации OWIN.

Owin-совместимый сервер отвечает за заполнение этого словаря данными.

Следующий элемент OWIN это делегат:

Func<IDictionary<string, object>, Task>;

В качестве входных данных он принимает выше описанный словарь, а возвращает объект Task по завершению процесса.

Katana — это OWIN-совместимый хост написанный Microsoft. Помимо реализации спецификации OWIN, Katana содержит различные вспомогательные классы и обертки для упрощения разработки, содержащиеся в библиотеке Owin.Types. Например два класса-адаптера упрощающие работу со словарём: OwinRequest и OwinResponse, реализация WebSocket для Owin — OwinWebSocket, вспомогательный класс для работы с заголовками и другими параметрами запросов — OwinHelpers.   
 Важный момент: Katana не завязана на использовании каких либо интерфейсов или базовых типов, вместо этого определены соглашения, которым должны следовать разработчики[13].

В данной работе используется класы GoogleAuthenticationExtensions и GoogleOAuth2AuthenticationOptions из пространства имён Owin и Microsoft.Owin.Security.Google соответственно, которые позволяют организовать OAuth2 авторизацию с помощью учётной записи Google.

* + 1. **EntityFramework**

Entity Framework представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами [14].

Центральной концепцией Entity Framework является понятие сущности или entity. Сущность представляет набор данных, ассоциированных с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их наборами.

Любая сущность, как и любой объект из реального мира, обладает рядом свойств. Например, если сущность описывает человека, то мы можем выделить такие свойства, как имя, фамилия, рост, возраст, вес. Свойства необязательно представляют простые данные типа int, но и могут представлять более комплексные структуры данных. И у каждой сущности может быть одно или несколько свойств, которые будут отличать эту сущность от других и будут уникально определять эту сущность. Подобные свойства называют ключами.

При этом сущности могут быть связаны ассоциативной связью один-ко-многим, один-ко-одному и многие-ко-многим, подобно тому, как в реальной базе данных происходит связь через внешние ключи.

Отличительной чертой Entity Framework является использование запросов LINQ для выборки данных из БД. С помощью LINQ мы можем не только извлекать определенные строки, хранящие объекты, из бд, но и получать объекты, связанные различными ассоциативными связями.

Другим ключевым понятием является Entity Data Model. Эта модель сопоставляет классы сущностей с реальными таблицами в БД.

Entity Data Model состоит из трех уровней: концептуального, уровень хранилища и уровень сопоставления (маппинга).

На концептуальном уровне происходит определение классов сущностей, используемых в приложении.

Уровень хранилища определяет таблицы, столбцы, отношения между таблицами и типы данных, с которыми сопоставляется используемая база данных.

Уровень сопоставления (маппинга) служит посредником между предыдущими двумя, определяя сопоставление между свойствами класса сущности и столбцами таблиц.

Таким образом, мы можем через классы, определенные в приложении, взаимодействовать с таблицами из базы данных.

Способы взаимодействия с БД

Entity Framework предполагает три возможных способа взаимодействия с базой данных:

* Database first: Entity Framework создает набор классов, которые отражают модель конкретной базы данных
* Model first: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере.
* Code first: разработчик создает класс модели данных, которые будут храниться в бд, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы
  + 1. **Паттерн «репозиторий»**

Основное преимущество репозиториев — это абстрактный механизм хранения для коллекций сущностей.

Предоставляя интерфейс MemberRepository мы развязываем руки разработчику, который уже сам решит, как и где хранить данные.

Посредничает между уровнями области определения и распределения данных (domain and data mapping layers), используя интерфейс, схожий с коллекциями для доступа к объектам области определения.

Система со сложной моделью области определения может быть упрощена при помощи дополнительного уровня, например Data Mapper, который бы изолировал объекты от кода доступа к БД. В таких системах может быть полезным добавление ещё одного слоя абстракции поверх слоя распределения данных (Data Mapper), в котором бы был собран код создания запросов. Это становится ещё более важным, когда в области определения множество классов или при сложных, тяжелых запросах. В таких случаях добавление этого уровня особенно помогает сократить дублирование кода запросов.

Паттерн Репозиторий посредничает между слоем области определения и слоем распределения данных, работая, как обычная колекция объектов области определения. Объекты-клиенты создают описание запроса декларативно и направляют их к объекту-репозиторию (Repository) для обработки. Объекты могут быть добавлены или удалены из репозитория, как будто они формируют простую коллекцию объектов. А код распределения данных, скрытый в объекте Repository, позаботится о соответсвующих операциях в незаметно для разработчика. В двух словах, паттерн Repository инкапсулирует объекты, представленыые в хранилище данных и операции, производимые над ними, предоставляя более объектно-ориентированное представление реальных данных. Repository также преследует цель достижения полного разделения и односторонней зависимости между уровнями области определения и распределения данных [15].

* + 1. **bootstrap**

Bootstrap — это CSS/HTML фреймворк для создания сайтов. Другими словами, это набор инструментов для вёрстки. В нём есть ряд преимуществ, благодаря которым BS считается самым популярным из себе подобных[16]. Преимущества бутстрапа:

* Скорость работы — благодаря множеству готовых элементов вёрстка с бутстрапом занимает значительно меньше времени;
* Масштабируемость — добавление новых элементов не нарушает общую структуру;
* Лёгкая настраиваемость — редактирование стилей производится путём создания новых css-правил, которые исполняются вместо стандартных. При этом не требуется использовать атрибуты типа “!important”;
* Большое количество шаблонов — о шаблонах Bootstrap я напишу далее;
* Огромное сообщество разработчиков;
* Широкая сфера применения — Bootstrap используется в создании тем для практически любой CMS (OpenCart, Prestashop, Magento, Joomla, Bitrix, WordPress и любые другие), в том числе для одностраничных приложений.

Особой популярностью пользуется Bootstrap для создания одностраничников или “лендингов” (landing page).

Шаблоны Bootstrap

Шаблоны в бутстрапе позволяют менять уже изменённые элементы под свои потребности. Множество разработчиков предлагают свои шаблоны (как платно, так и бесплатно).

Подключить шаблон в Bootstrap очень легко: после подключения бутстрапа нужно только добавить вызов CSS вашего шаблона.

* + 1. **jQuerry**

jQuery — библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX

Возможности:

* Движок кросс-браузерных CSS-селекторов Sizzle, выделившийся в отдельный проект;
* Переход по дереву DOM, включая поддержку XPath как плагина;
* События;
* Визуальные эффекты;
* AJAX-дополнения;
* JavaScript-плагины.

Точно так же, как CSS отделяет визуализацию от структуры HTML, JQuery отделяет поведение от структуры HTML. Например, вместо прямого указания на обработчик события нажатия кнопки, управление передаётся JQuery, которая идентифицирует кнопки и затем преобразует его в обработчик события клика. Такое разделение поведения и структуры также называется принципом ненавязчивого JavaScript.

Библиотека jQuery содержит функциональность, полезную для максимально широкого круга задач. Тем не менее, разработчиками библиотеки не ставилась задача совмещения в jQuery функций, которые подошли бы всюду, поскольку это привело бы к большому коду, бо́льшая часть которого не востребована. Поэтому была реализована архитектура компактного универсального ядра библиотеки и плагинов. Это позволяет собрать для ресурса именно ту JavaScript-функциональность, которая на нём была бы востребована[17].

### jQuery DataTables

**DataTables** это мощный плагин jQuery для создания таблиц и списков. Плагин jQuery DataTables способен сделать таблицу с поиском, сортировкой и разбитие на страницы без каких-либо особых настроек.

Сначала нужно скачать библиотеку DataTables. Если Вы не хотите, скачивать файлы, можете воспользоваться Microsoft CDN. Его необходимо подключить как и плагин jQuery и тоже желательно последней версии.

DataTables позволяет работать с данными напрямую в HTML, указать данные в виде массива во время инициализации. Или можем работать с данными, источник которых поступает из Ajax и MySQL.

Таблицу можно просматривать на маленьких экранах мобильных устройств. Для этого нужно дополнительно подключить файл CSS и JavaScript. Для таблицы HTML добавить необходимые классы[18].

* 1. **Заключение**

После обзора всех технологий, в качестве основной базы данных была выбрана Oracle, в качестве вебсервера был выбран IIS сервер и ASP.NET приложение

Преимущества ASP.NET:

* Надежность и стабильность вашего сайта

Огромное значение для современного бизнеса имеет отсутствие простоев, не важно, будет это на час или несколько часов в день. Простой приводит к серьезным убыткам, удару по репутации в деловом мире. Именно поэтому огромную роль в работе сайта играет его надежность и устойчивость к хакерским атакам.

Встроенная защита от различных видов нападений, предоставляет следующие возможности: SQL Injection, переполнение буфера, XSS, изменение скрытых полей и прочие. Технология ASP.NET повышает степень устойчивости к вредоносным действиям и различным видам хакерских атак сайтов, построенных на ней.

* Производительность и скорость работы сайта

Строение ASP.NET как технологии позволяет компилировать программный код и все страницы сайта. Код интерпретируется в PHP значительно медленнее и не дает нужного эффекта производительности.

Особенно это касается активного использования в разработке сайта концепции ООП. Благодаря тому, что в ASP.NET встроено функционирование сайта на кластере сервера, при увеличении посещаемости на сайте достигается масштабируемость.

* Интеграция с приложениями и прочими информационными системами

Платформа Microsoft .NET имеет множество встроенных технологий для интеграции информационных систем и приложений, таких как службы web, WCF, JSON, remoting, XML и пр., при этом ASP.NET существует как ее часть.

Наличие таких многочисленных решений дает возможность выбора оптимальной технологии для каждого отдельного случая. Это обеспечит отменную производительность, масштабируемость и, самое главное, - безопасность.

* Скорость разработки на высшем уровне

Для разработки на платформе ASP.NET предлагается среда MS Visual Studio от 2010 года, которая признана одним из лучших средств. Данная среда упростит и ускорит создание web-приложений за счет применения усовершенствованного конструктора web-форм, поддержки ASP.NET AJAX, неограниченного набора коммерческих и стандартных элементов управления.

В целях эффективной совместной работы группы и организации взаимодействия, возможна интеграция Visual Studio с TFS и SS - средствами коллективной разработки. При этом обеспечивается наглядность управления функционированием приложений и процесса разработки, а также повышается качество создаваемого программного обеспечения.

Надежная и безопасная работа приложений обеспечена и повышается также с помощью применения встроенного модульного тестирования. Для среды Microsoft Visual Studio придается встроенная поддержка продвинутых технологий AJAX, WCF, Silverlight, WPF, а также WWF [20].

Среди основных свойств СУБД Oracle следует отметить такие, как[21]:

1. Высочайшая надежность.
2. Возможность разбиения крупных баз данных на разделы (large-database partition), что дает возможность эффективно управлять гигантскими гигабайтными базами;
3. Наличие универсальных средств защиты информации;
4. Эффективные методы максимального повышения скорости обработки запросов;
5. Индексация по битовому отображению;
6. Свободные таблицы (в других СУБД все таблицы заполняются сразу при создании);
7. Распараллеливание операций в запросе;
8. Наличие широкого спектра средств разработки, мониторинга и администрирования;
9. Поддержка известных платформ: Windows, AIX, Compaq Tru64 UNIX, HP 9000 Series HP-UX, Linux Intel, Sun Solaris
10. Ориентация на интернет технологии.
11. Поддержка XML в хранимых процедурах, позволяющая разработчикам (традиционных) баз данных непосредственно использовать преимущества языка XML, применяя привычный механизм хранимых процедур
12. Доступ по протоколу HTTP, поддерживающий отправку SQL-запросов к БД с применением URL-адресов
13. Интегрированное средство выявления закономерностей, применяемое для отбора важной, но не обязательно очевидной бизнес-информации из больших наборов данных. Оно обеспечиваюет сбор, хранение, управление и анализ потока данных о действиях пользователей при посещении ими Web-узла
14. Связанные базы данных OLAP - это, в первую очередь, кубы OLAP, применяемые для реализации новых возможностей анализа данных; они позволяют повысить ценность данных за счет предоставления возможностей анализа OLAP через Web
15. Доступ к кубам данных по протоколу HTTP, обеспечивающий просмотр и управление многомерными кубами данных в Web и через брандмауэры при помощи встроенного в БД средства поддержки протокола HTTP
16. Средство передачи журналов, позволяющее синхронизировать отдельные системы посредством автоматизированного обновления журнала транзакций, что повышает надежность системы и устойчивость к авариям
17. Параллельные операции при создании индексов и выполнении поиска на компьютерах с симметричной многопроцессорной обработкой, ускоряющие работу приложений
18. Улучшенные средства отказоустойчивости, упрощающие создание и управление отказоустойчивыми кластерами, переключение между узлами кластера при сбое и восстановление узла кластера, не оказывающее влияния на остальные узлы
19. Поддержка большого объема памяти и симметричной многопроцессорной обработки,позволяющая управлять приложениями с высокими нагрузками, связанными с большим объемом транзакций
20. Поддержка протокола VI SAN (Virtual Interface System Area Network), позволяющая эффективно использовать сети SAN (System Area Network - системная сеть) со встроенным интерфейсом VI Consumer, который позволяет устанавливать прямую связь (в режиме пользователя) с дисковыми устройствами SAN
21. Поддержка службы единого каталога, позволяющая в крупных организациях упростить и централизовать управление именованием серверов, данными OLAP, репликациями и хронологией архивирования
22. Инструментальные средства, включающие графические средства разработки схем, генерации запросов и кода
23. **Проектирование автоматизированной системы «Электронный деканат»**

**2.1. Архитектура**

Приложение состоит из 3 частей: база данных Oracle, база данных MSSQL и IIS сервера, на котором работает ASP.NET приложение.

И базы данных, и IIS сервер работают в виртуальной машине под управлением Windows 7, доступной по внешнему IP адресу 195.50.2.89.

Само приложение развёрнуто на IIS сервере и доступно по адресу *http://195.50.2.89:40001/*

**2.2. Диаграммы**

Диаграмма вариантов использования разработанной системы:

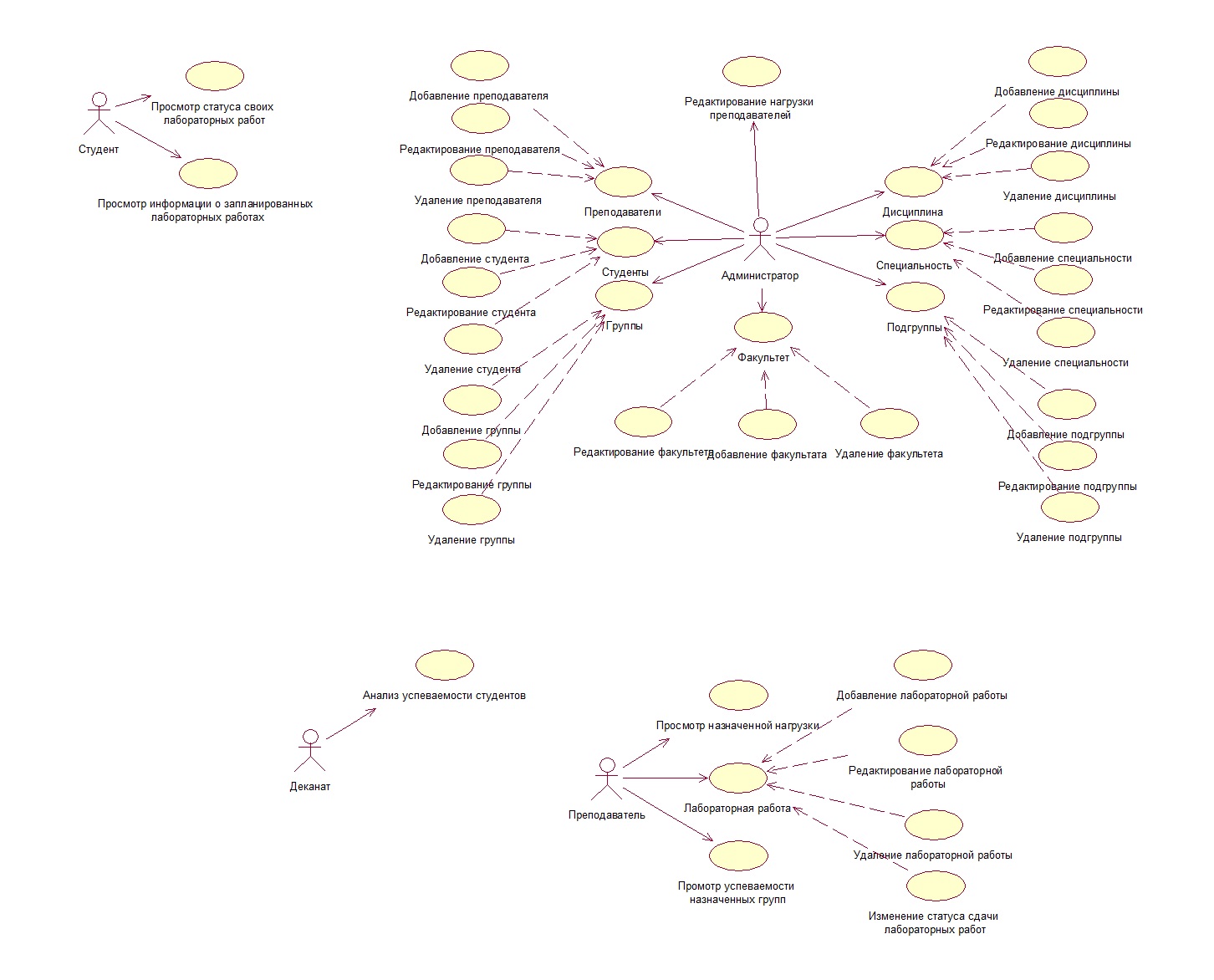
****

Рисунок –Диаграмма вариантов использования

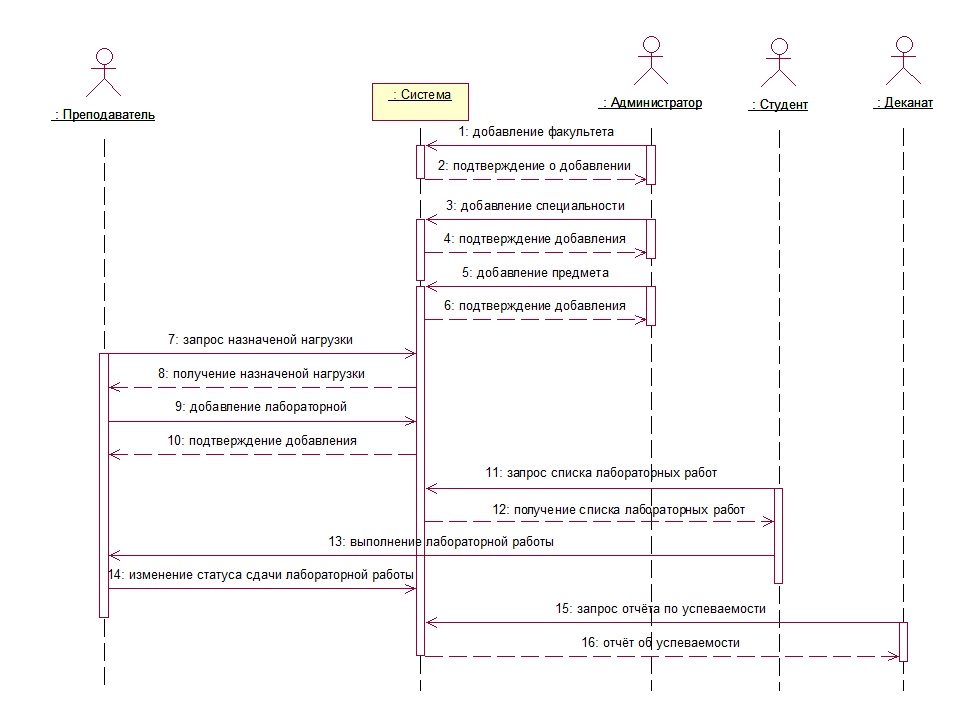
****

Рисунок – Диаграмма последовательностей

**2.3. Структура базы данных**

Таблицы в базе данных представлены в следующих отношениях:

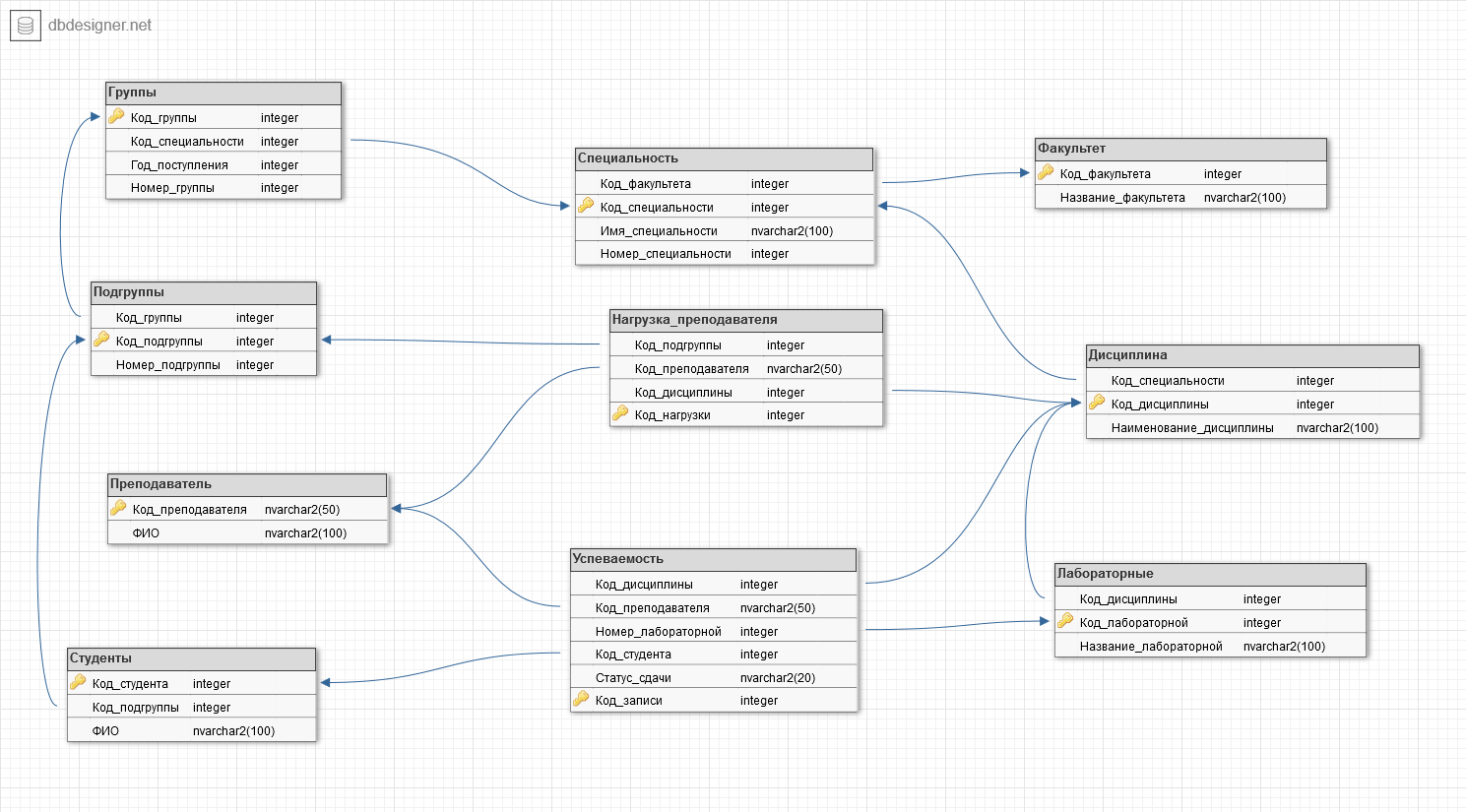


Рисунок – Схема базы данных

В соответствии со схемой, были созданы следующие таблицы:

Таблица – Таблицы в базе данных

|  |  |
| --- | --- |
| Наименование таблицы | Назначение |
| Факультеты | Перечень факультетов вуза |
| Специальность | Перечень кафедр вуза |
| Дисциплина | Перечень дисциплин, привязка их к кафедрам |
| Группы | Перечень групп, содержащий их специальность и год поступления |
| Подгруппы | Перечень подгрупп, содержащий их номера и привязку к группам |
| Студенты | Перечень студентов, содержащий ФИО студентов и их привязку к подгруппе |
| Преподаватель | Перечень преподавателей вуза |
| Нагрузка преподавателя | Таблица, в которой хранится закрепление подгруппы за преподавателем |
| Лабораторные | Перечень всех лабораторных с привязкой к дисциплине |
| Успеваемость | Таблица, которая хранит статус сдачи лабораторных студентами |

Назначение столбцов таблиц:

Таблица 2 – Факультеты

|  |  |
| --- | --- |
| Столбец | Назначение |
| Название\_факультета | Содержит название факультета в буквенном сокращении (ФИТ, ИДиП…) |
| Код\_факультета | Уникальный числовой идентификатор факультета |

Таблица – Специальность

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_факультета | Содержит название факультета в буквенном сокращении (ФИТ, ИДиП…) |
| Номер\_специальности | Цифровой код, описывающий эту специальность (например, 400501) |
| Имя\_специальности | Полное название этой специальности |
| Код\_специальности | Уникальный числовой идентификатор специальности |

Таблица – Дисциплина

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_специальности | Цифровой код, описывающий эту специальность (см. таблицу Специальность столбец 2) |
| Код\_дисциплины | Уникальный числовой идентификатор дисциплины |
| Наименование\_дисциплины | Полное название этой дисциплины |

Таблица – Группы

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_группы | Уникальный числовой идентификатор группы |
| Код\_специальности | Цифровой код, описывающий эту специальность (см. таблицу Специальность столбец 2) |
| Год\_поступления | Год зачисления студентов группы в вуз |
| Номер\_группы | Порядковый номер группы |

Таблица – Подгруппы

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_группы | Привязка подгруппы к группе |
| Код\_подгруппы | Уникальный числовой идентификатор подгруппы |
| Номер\_подгруппы | Порядковый номер подгруппы |

Таблица – Студенты

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_студента | Уникальный числовой идентификатор студента |
| Код\_подгруппы | Привязка студента к подгруппе |
| ФИО | Полное ФИО студента |

Таблица – Преподаватели

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_преподавателя | Уникальный числовой идентификатор преподавателя |
| ФИО | Полное ФИО преподавателя |

Таблица – Нагрузка преподавателя

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_подгруппы | Идентификатор обучаемой подгруппы |
| Код\_преподавателя | Идентификатор обучающего преподавателя |
| Код\_дисциплины | Идентификатор преподаваемой дисциплины |

Таблица – Лабораторные

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_дисциплины | Идентификатор дисциплины, за которой закреплена лабораторная работа |
| Код\_лабораторной | Уникальный числовой идентификатор лабораторной работы |
| Название\_лабораторной | Полное название лабораторной работы |

Таблица – Успеваемость

|  |  |
| --- | --- |
| Столбец | Назначение |
| Код\_дисциплины | Идентификатор дисциплины, за которой закреплена лабораторная работа |
| Код\_преподавателя | Уникальный числовой идентификатор лабораторной работы |
| Номер\_лабораторной | Полное название лабораторной работы |

Для каждого поля, в котором присутствует уникальный идентификатор, были созданы sequence и BeforeInsert триггер уровня строки, который получает идентификаторы из этого sequence.

Помимо таблиц, для удобного представления данных из базы, были созданы следующие представления:

Таблица – Представления

|  |  |
| --- | --- |
| Представление | Назначение |
| FACULTY\_LIST | Перечень названий всех факультетов |
| KOORS\_LIST | Перечень всех групп, с указанием факультета, кафедры и текущего курса обучения. |
| STUDENT\_LIST | Перечень всех студентов, с указанием специальности, номера группы и подгруппы |
| LAB\_LIST | Текущая успеваемость студентов, статус сдачи ими лабораторных, в удобном для восприятия виде |
| TEACHER\_DISCIPLINE\_LIST | Список привязок преподавателей к подгруппам, в удобном для чтения виде |
| DISCIPLINE\_LIST | Перечень информации о дисциплинах |
| LABS\_DISCIPLINE | Перечень лабораторных работ на дисциплинах |
| SPECIALITY\_LIST | Перечень специальностей на факультете |
| STUDENT\_DISCIPLINE | Перечень связки студент-дисциплина-преподаватель |
| STUDENT\_LAB\_DISCIPLINE | Успеваемость студентов, отфильтрованная по дисциплинам |
| STUDENT\_LABS | Успеваемость по студентам, суммарная по всем дисциплинам |
| STUDENT\_LIST | Перечень всех студентов |
| SUBGROUP\_LIST | Перечень всех подгрупп |
| TEACHER\_DISCIPLINE | Перечень связки дисциплина-преподаватель |

Помимо создания всех необходимых таблиц и представлений, для обеспечения эффективной работы с базой данных, были созданы функции и процедуры. Все процедуры, кроме SET\_PROGRESS, представлены парами ADD/REMOVE, позволяющие удалить или добавить соответствующий логический объект в базу данных.

Таблица – Процедуры

|  |  |
| --- | --- |
| Название | Назначение |
| REMOVE/EDIT/ADD\_FACULTY | Добавление/редактирование/ удаление факультета |
| REMOVE/ EDIT/  ADD\_SPECIALISTIS | Добавление/ редактирование/ удаление кафедры, и привязка её к факультету |
| REMOVE/ EDIT/  ADD\_DISCIPLINE | Добавление/ редактирование/ удаление дисциплины, и привязка её к кафедре |
| REMOVE/EDIT/ADD\_TEACHER | Добавление/ редактирование/ удаление преподавателя |
| REMOVE/ EDIT/ADD\_LAB | Добавление/ редактирование/ удаление лабораторной работы на указанную дисциплину |
| REMOVE/ADD\_GROUP | Добавление/удаление группы студентов, и привязка её за специальностью |
| REMOVE/ADD\_SUBGROUP | Добавление/удаление подгруппы студентов, и привязка её к конкретной группе |
| REMOVE/ EDIT/ADD\_STUDENT | Добавление/ редактирование/ удаление студента, привязка его к конкретной группе |
| REMOVE/ADD\_WORK | Добавление/удаление нагрузки преподавателя |
| SET\_PROGRESS | Изменение статуса сдачи лабораторной конкретного студента |
| LOAD\_LABS | Служебная процедура, вызываемая для студента, при его добавлении, и при добавлении лабораторной работы, проверяющая таблицу успеваемости, и добавляющая необходимые записи в неё. |
| LOAD\_LABS\_SUBGROUP | Процедура, вызывающая LOAD\_LABS для всех студентов в подгруппе. |

Для вычисления курса по году поступления была разработана функция GET\_KOORS\_BY\_YEAR, которая считает разницу между годом поступления студента и 1 августа текущего года, определяя курс, на котором студент сейчас находится.

* 1. **Заключение**

Всего было разработано 10 таблиц, 27 хранимых процедур, 9 триггеров, 1 функция, 13 представлений, которые обеспечивают согласованность данных, и предоставить весь необходимый функционал веб-серверу. Так же, была разработана Use Case диаграмма, которая позволила определиться, какие контроллеры, модели и представления будут в системе.

1. **Реализация программного обеспечения автоматизированной системы «Электронный деканат»**

**3.1. Принципы построения по автоматизированной системы**

* Система имеет следующие особенности реализации и технические решения:
* Используется принцип «тонкий клиент» - вся работа возложена на Oracle server, комбинация из процедур, триггеров, ограничений целостности, функций и представлений выполняют всю логику, и поддерживают базу в согласованном состоянии.
* Для доступа к данным применяется паттерн «репозиторий» который позволяет получить доступ к данным единообразным образом.
* Репозитории объединены с помощью паттерна UnitOfWork (связь включением), который обеспечивает единообразие получение данных из репозиториев.
* Так как необходимо отображать множество различных таблиц, для всех была применена JQuerry datatable библиотека, обеспечивающая разбивку на страницы, сортировку и поиск в каждой таблице.
* Для соединения с базой данных была написана обёртка в виде singleton-класса, с помощью которого все репозитории получают доступ, использую единый объект connection.
* Для большинства сущностей, применяемых в системе, реализованы CRUD операции создания, чтения, редактирования и удаления.
* Для большинства сущностей применяется схема, где для редактирования сущности применяется модель, наследуемая от базовой, с префиксом New, что позволяет корректно использовать внутренние механизмы ASP.NET, такие как построение страницы по модели, и валидация. Это необходимо, так как при редактировании сущности, необходимо показать как старые, так и новые значения, и базовый класс для этого не подходит.
* База ASP.NET аккаунтов физически отделена от базы сущностей предметной области, и находится на MSSQL сервере, что позволяет использовать встроенные в ASP.NET механизмы авторизации, аутентификации, ролей, OAuth2 авторизацию, защищённое хранилище паролей.
* Стандартная регистрация была доработана, что бы обеспечить соответствие между пользователями ASP.NET и преподавателями в системе.

## **3.2. Контроллеры**

Таблица – Контроллеры

|  |  |
| --- | --- |
| Контроллер | Описание |
| Account | Этот контроллер содержит действия по управлению аккаунтом, регистрация (по логину и паролю, и по OAuth2), авторизация, смена пароля. |
| Admin | Этот контроллер содержит все действия, соответствующие роли «администратор» в системе, а именно: создание, редактирование и удаление факультетов, кафедр, специальностей, групп, подгрупп, студентов, управление нагрузкой преподавателя. |
| Teacher | Этот контроллер содержит все действия, соответствующие роли «преподаватель» в системе, а именно: создание, редактирование и удаление лабораторных работ по преподаваемым им дисциплинам. Так же, ему доступна страница успеваемости, с помощью которой, преподаватель может отмечать факт защиты студентом лабораторной работы. |
| Dekan | Этот контроллер содержит все действия, соответствующие роли «декан» в системе, а именно: просмотр успеваемости студентов, анализ данных, содержащихся в базе данных, оперативное выявление проблем с успеваемостью. |
| Student | Этот контроллер содержит все действия, соответствующие роли «студент» в системе, а именно: просмотр запланированных и защищённых лабораторных работ. |
| Main | Содержит страничку приветствия. |

## **3.3. Модели**

Таблица – Модели

|  |  |
| --- | --- |
| Модель | Описание |
| Discipline | Модель, представляющая дисциплину на специальности |
| DisciplineStudy | Наследуется от модели Study, представляет успеваемость студента по конкретной дисциплине |
| Faculty | Модель, представляющая факультет |
| Group | Модель, представляющая группу |
| Lab | Модель, представляющая лабораторную работу |
| LabProgress | Модель, представляющая запись о статусе защиты студентом лабораторной работы. |
| NewDiscipline | Наследуется от модели Discipline, используется при редактировании дисциплины. |
| NewFaculty | Наследуется от модели Faculty, используется при редактировании факультета. |
| NewLab | Наследуется от модели Lab, используется при редактировании лабораторной работы. |
| NewSpeciality | Наследуется от модели Speciality, используется при редактировании специальности. |
| NewStudent | Наследуется от модели Student, используется при редактировании студента. |
| NewTeacher | Наследуется от модели Teacher, используется при редактировании преподавателя. |
| NewUserRole | Наследуется от модели Teacher, используется при добавлении роли пользователю. |
| NewWork | Наследуется от модели Work, используется при добавлении нагрузки преподавателю. |
| Speciality | Модель, представляющая специальность на факультете |
| Stydent | Модель, представляющая студента |
| Study | Модель, представляющая суммарную успеваемость студента |
| Subgroup | Модель, представляющая подгруппу |
| Teacher | Модель, представляющая преподавателя |
| TeacherWork | Модель, представляющая нагрузку преподавателя для преподавателя. |
| UserRole | Модель, представляющая роль пользователя |
| Work | Модель, представляющая нагрузку преподавателя для администратора. |

Диаграмма классов, показывающая отношения между моделями:

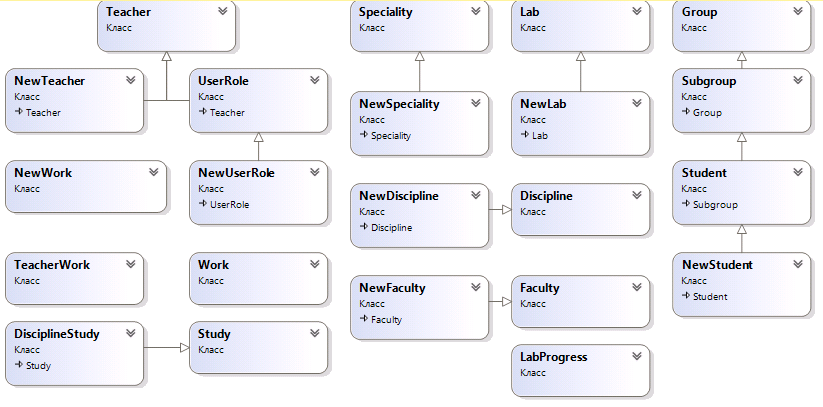


Рисунок – Диаграмма классов

## **3.4. Представления**

### 3.4.1. Контроллер «Teacher»:

Таблица – Представления контроллера Teacher

|  |  |
| --- | --- |
| Представление | Описание |
| AddLab | Форма для добавления лабораторной работы на дисциплину |
| DeleteLab | Форма для удаления выбранной лабораторной работы |
| EditLab | Форма для редактирования выбранной лабораторной работы |
| Index | Страница, позволяющая преподавателю выбрать дисциплину и подгруппу, для возможности последующего редактирования статуса защиты лабораторных работ у студентов выбранной подгруппы. |
| Labs | Страница, позволяющая выбирать студентов, и смотреть, какие лабораторные работы ими сданы, какие нет, и отмечать факт защиты работ. |
| LabsList | Страница, на которой преподаватель может просмотреть все преподаваемые им дисциплины, а так же, перейти к списку лабораторных работ на выбранной дисциплине. |
| LabOnDisciplineList | Страница, отображающая для выбранной ранее дисциплины лабораторные работы с возможностью удалять, добавлять, и редактировать их. |
| ChangeLabStatus | Форма, с помощью которой можно отметить факт сдачи лабораторной работы. |

### 3.4.2. Контроллер «Student»:

Таблица – Представления контроллера Student

|  |  |
| --- | --- |
| Представление | Описание |
| Index | Страница, позволяющая найти информацию о студенте в базе данных |
| Student | Страница, отображающая успеваемость студента по всем преподаваемым у него дисциплинам |

### 3.4.3. Контроллер «Main»:

Таблица – Представления контроллера Main

|  |  |
| --- | --- |
| Представление | Описание |
| Index | Страница приветствия, так же отображающая авторство этой дипломной работы. |

### 3.4.4. Контроллер «Dekan»:

Таблица – Представления контроллера Dekan

|  |  |
| --- | --- |
| Представление | Описание |
| Facultes | Страница со списком факультетов |
| Specialitis | Страница со списком специальностей на факультете |
| Subgroups | Страница со списком групп на специальности |
| Studing | Страница, отображающая список студентов подгруппы, суммарное количество защищённых, запланированных и не защищённых студентами лабораторных работ |
| DisciplineStuding | Страница, отображающая список дисциплин у студента, суммарное количество защищённых, запланированных и не защищённых им лабораторных работ по каждой дисциплине |

### 3.4.5. Контроллер «Admin»:

Таблица – Представления контроллера Admin

|  |  |
| --- | --- |
| Представление | Описание |
| Facultes | Страница со списком факультетов с возможностью удалять, добавлять, и редактировать их. |
| AddFaculty | Форма для добавления факультета |
| EditFaculty | Форма для переименования факультета |
| DeleteFaculty | Форма для удаления факультета |
| Specialitis | Страница со списком специальностей на факультете, с возможностью удалять, добавлять, и редактировать их. |
| AddSpeciality | Форма для добавления специальности на факультет |
| EditSpeciality | Форма для редактирования выбранной специальности |
| DeleteSpeciality | Форма для удаления выбранной специальности |
| Disciplines | Страница со списком дисциплин на специальности, с возможностью удалять, добавлять, и редактировать их. |
| AddDiscipline | Форма для добавления дисциплины на специальность |
| EditDiscipline | Форма для редактирования выбранной дисциплины |
| DeleteDiscipline | Форма для удаления выбранной дисциплины |
| Groups | Страница со списком групп на специальности, с возможностью добавлять и удалять их. |
| AddGroup | Форма для добавления группы на специальность |
| DeleteGroup | Форма для удаления выбранной группы |
| Subgroups | Страница со списком подгрупп в группе, с возможностью добавлять и удалять их. |
| AddSubgroup | Форма для добавления подгруппы в группу |
| DeleteSubgroup | Форма для удаления выбранной подгруппы |
| Students | Страница со списком студентов в подгруппе с возможностью удалять, добавлять, и редактировать их. |
| AddStudent | Форма для добавления студента в подгруппу |
| EditStudent | Форма для редактирования выбранного студента |
| DeleteStudent | Форма для удаления выбранного студента |
| Works | Страница, отображающая, кто и какую дисциплину преподаёт у выбранной подгруппы, с возможностью добавления и удаления нагрузки преподавателям |
| AddWork | Форма для добавления нагрузки преподавателя на выбранную подгруппу |
| DeleteWork | Форма для удаления нагрузки преподавателя |
| Teachers | Страница, отображающая информацию о зарегистрированных в системе преподавателях |
| EditTeacher | Форма для редактирования выбранного преподавателя |
| DeleteTeacher | Форма для удаления выбранного преподавателя |
| Users | Страница, отображающая информацию о зарегистрированных в системе пользователях, включая преподавателей, деканов и администраторов |
| Roles | Страница, отображающая информацию о ролях, назначенных пользователям, с возможностью добавить и удалить роли |
| AddRole | Форма для добавления пользователю роли |

## **3.5. Маршрутизация**

Маршрутизация стандартная, по шаблону {controller}/{action}/{id}, где:

controller – имя контроллера

action – имя действия в контроллере

id – необязательный параметр идентификатора

Контроллер по умолчанию Main, действие по умолчанию – Index

## **3.6. Заключение**

В приложении всего: 7 контроллеров, 74 представления

Таблица – Колличественные характеристики кода

|  |  |  |
| --- | --- | --- |
| расширение | файлов | Строк кода |
| .cs | 63 | 4667 |
| .cshtml | 146 | 6210 |
| .js | 28 | 34977(включая библиотеки) |
| .css | 10 | 13929(включая библиотеки) |

1. **Описание контрольного примера**

Пример состоит из 4 частей-вкладок меню: администратор, преподаватель, студент, деканат.

* 1. **Администратор**

Для администратора была создана учётная запись (Логин admin@gmail.com, пароль 12345). Администратор, помимо контроля за содержимым базы данных, может выдавать полномочия администратора или деканата любому зарегистрированному пользователю. Окно, открывающееся после авторизации:

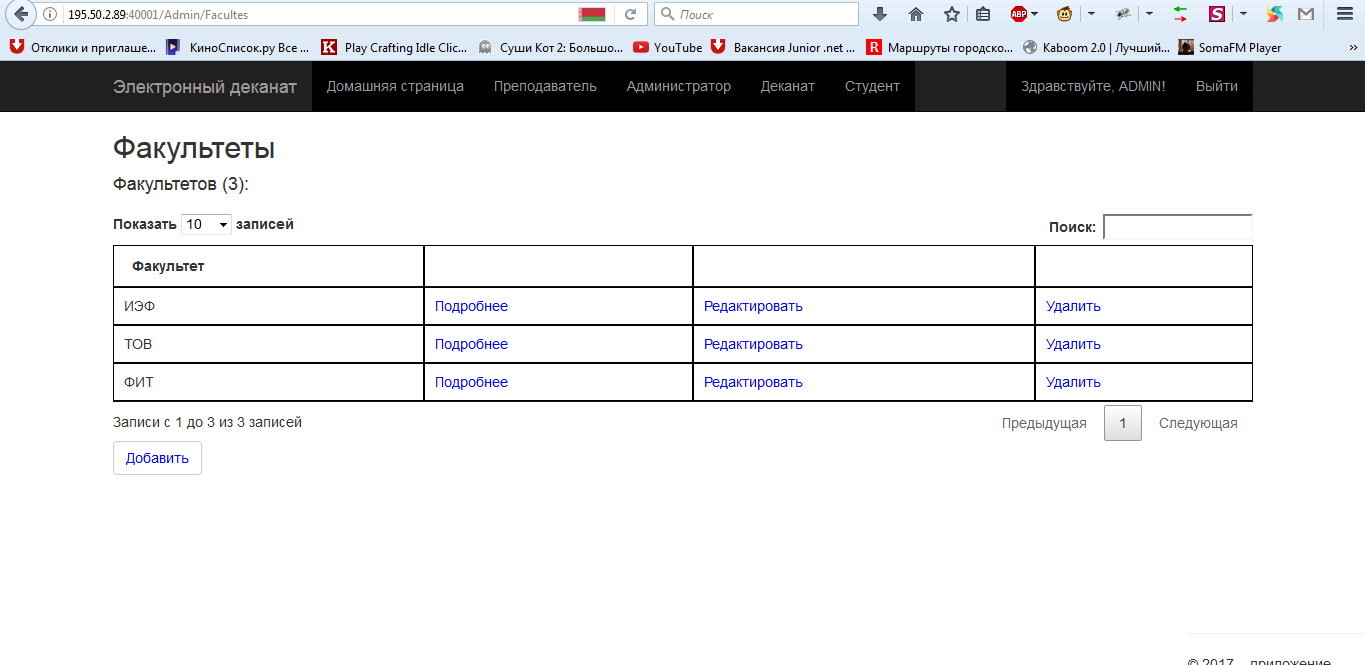


Рисунок – Администратор, страница факультетов

С этой страницы, можно создать, удалить или переименовать факультет, а так же зайти в подробную информацию о факультете.

Форма добавления факультета:

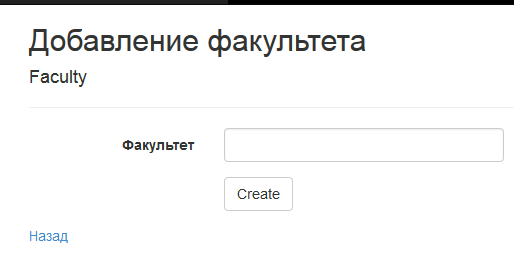


Рисунок – Администратор, добавление факультета

Форма удаления факультета:

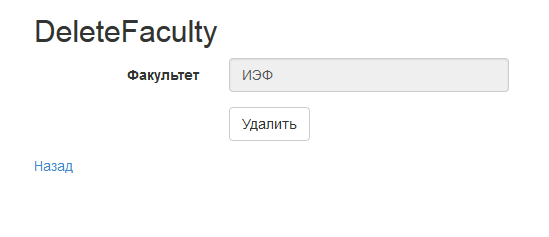


Рисунок – Администратор, удаление факультета

Форма редактирования факультета:

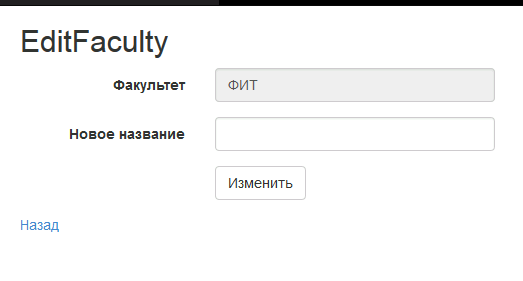


Рисунок – Администратор, редактирование факультета

Страница, после нажатия на «подробнее» напротив факультета ФИТ:

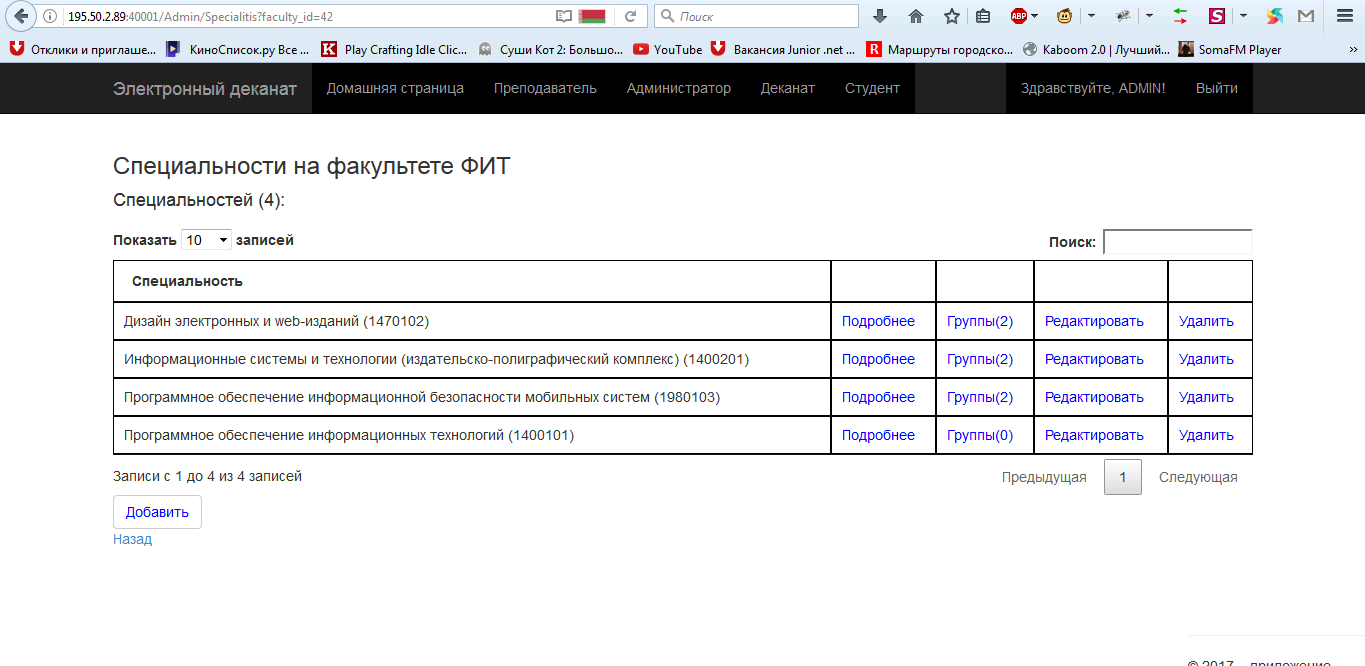


Рисунок – Администратор, страница специальностей

С этой страницы, можно создать, удалить или переименовать специальность, а так же зайти в подробную информацию о специальности, и посмотреть какие группы учатся на этой специальности.

Форма добавления специальности:

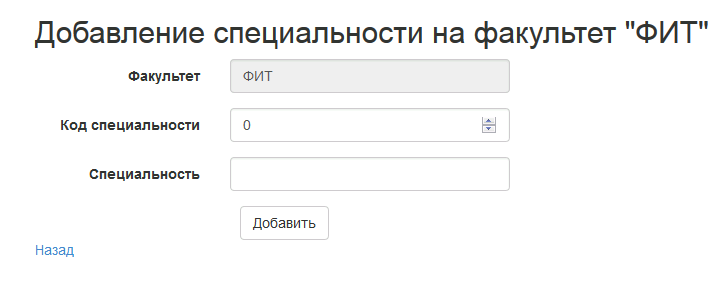


Рисунок – Администратор, добавление специальности

Форма редактирования специальности:

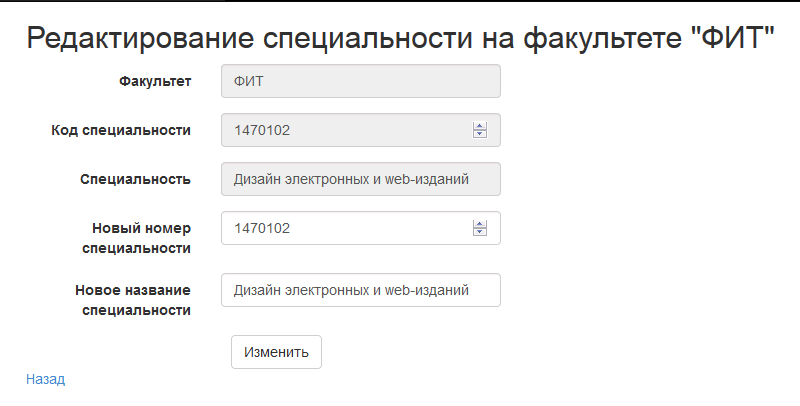


Рисунок – Администратор, редактирование специальности

Форма удаления специальности:

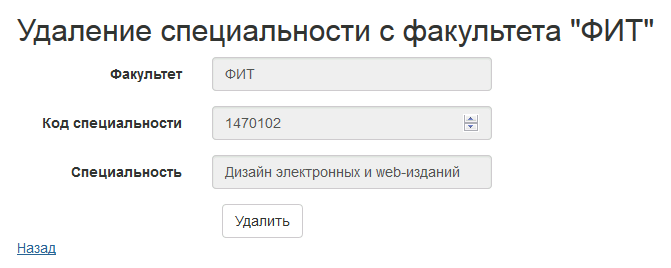


Рисунок – Администратор, удаление специальности

После нажатия на «подробнее» напротив специальности «ИСиТ»:

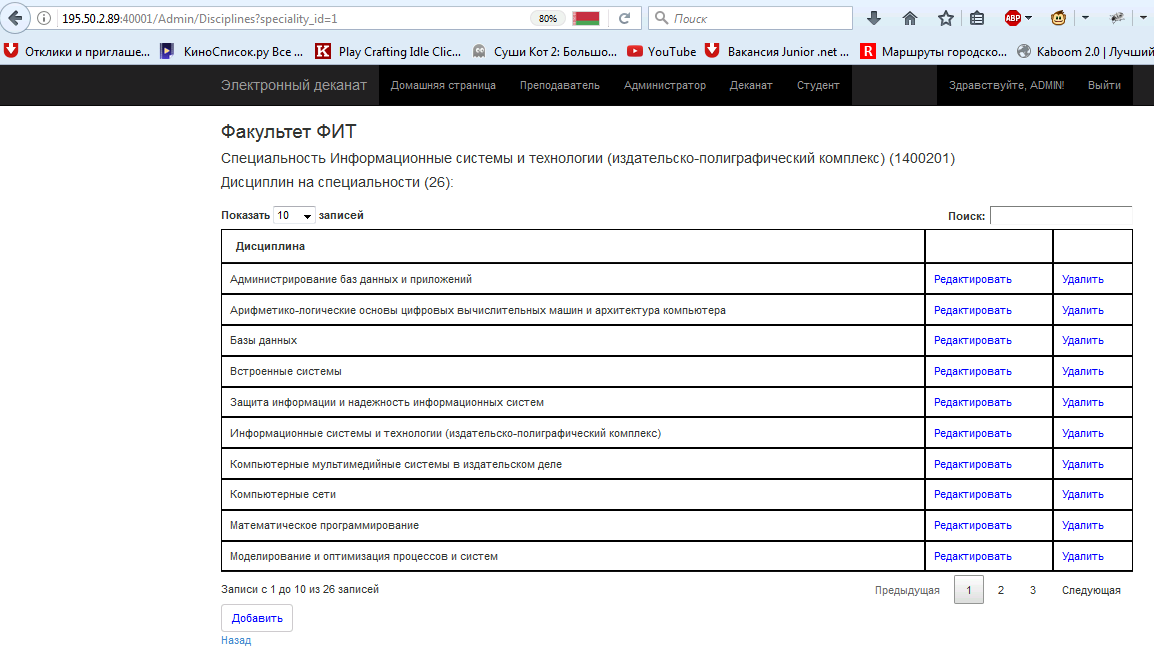


Рисунок – Администратор, дисциплины

С этой страницы, можно создать, удалить или переименовать дисциплины.

Вернёмся на экран «специальности», и нажмём на «группы» напротив специальности ИСиТ(число в скобках – количество групп на специальности).

Попадаем на экран «группы». С этой страницы, можно создать и удалить группы. На кнопку «подробнее» можно переместиться на экран «подгруппы» (число в скобках показывает количество подгрупп):

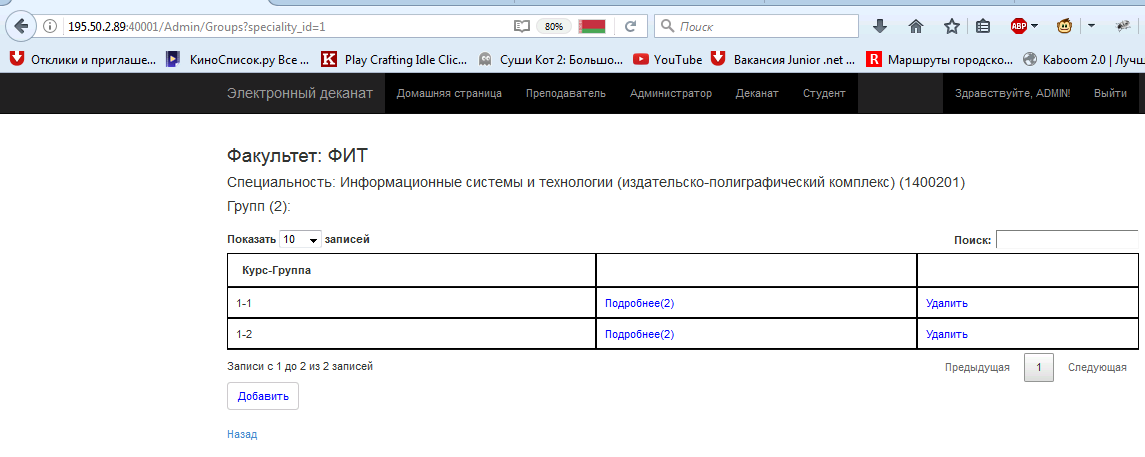


Рисунок – Администратор, страница групп

После нажатия на кнопку «подробнее» напротив группы 1-1:

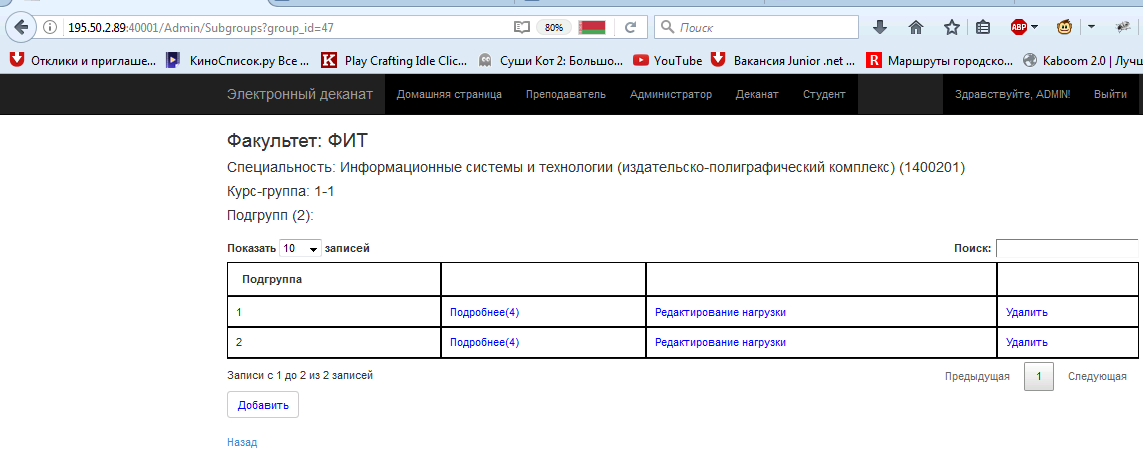


Рисунок – Администратор, страница подгрупп

Попадаем на экран «группы». С этой страницы, можно создать и удалить подгруппы. На кнопку «подробнее» можно переместиться на экран «студенты» (число в скобках показывает количество студентов), на кнопку «редактирование нагрузки» можно просмотреть, кто и что преподаёт у подгруппы, и отредактировать эти данные:

Нажимаем «подробнее» напротив подгруппы 1. С этой страницы, можно создать, редактировать и удалить студентов:

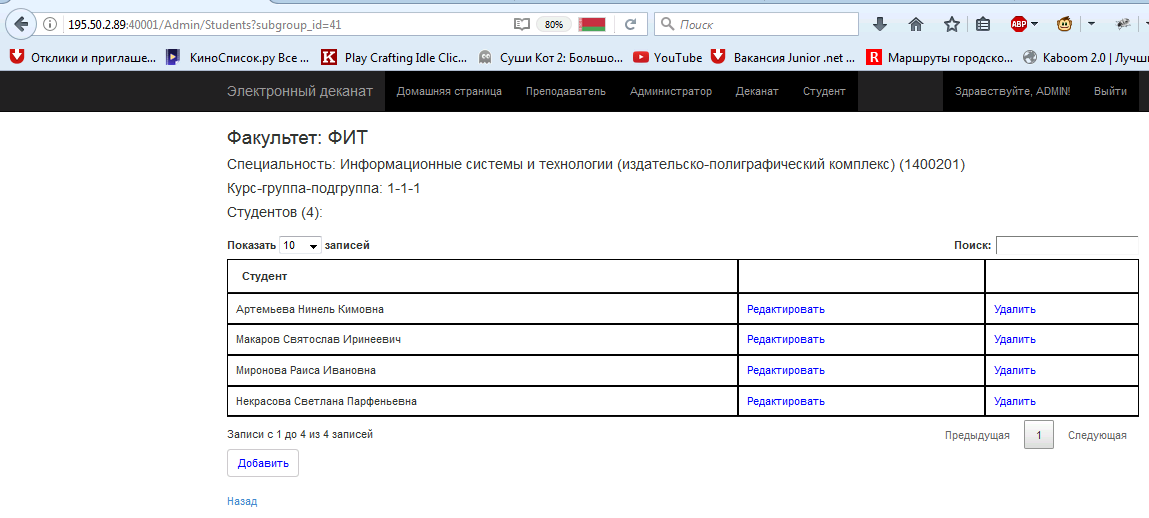


Рисунок – Администратор, страница студентов

Вернувшись назад, и нажав на «редактирование нагрузки», попадаем на следующую страницу:

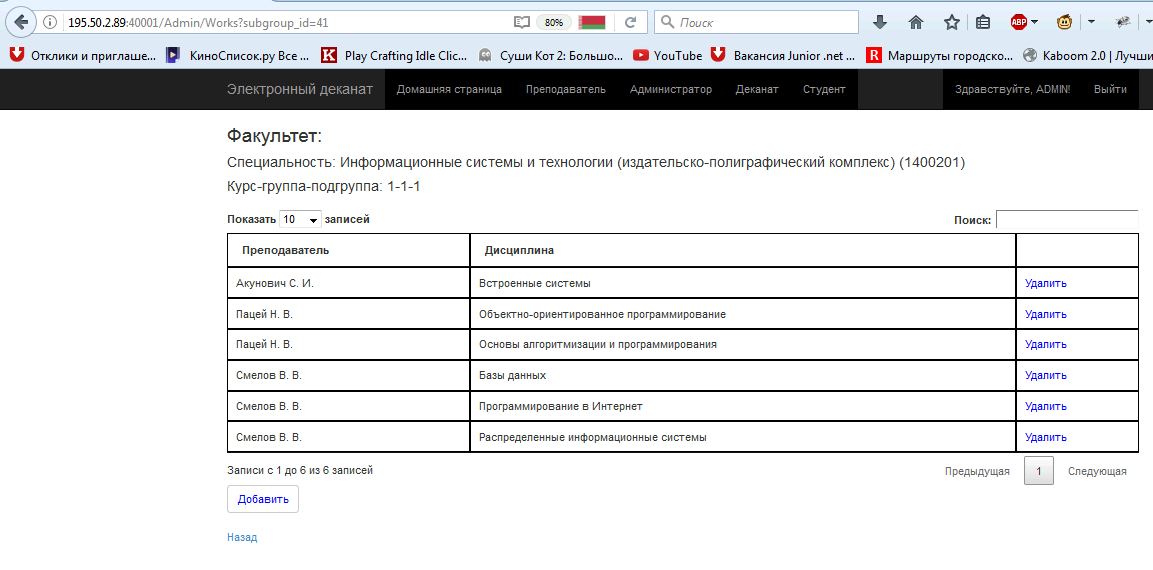


Рисунок – Администратор, список нагрузок преподавателей

На этой странице, можно добавить и удалить нагрузку преподавателей для этой подгруппы.

Форма добавления нагрузки:

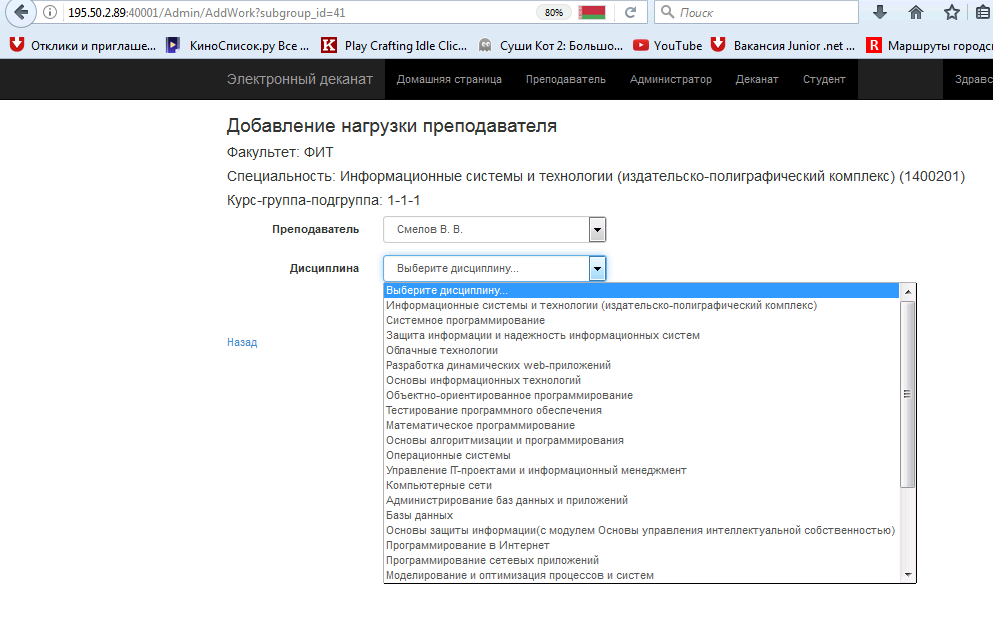


Рисунок – Администратор, добавление нагрузки преподавателя

Перейдём на вкладку меню «преподаватели»

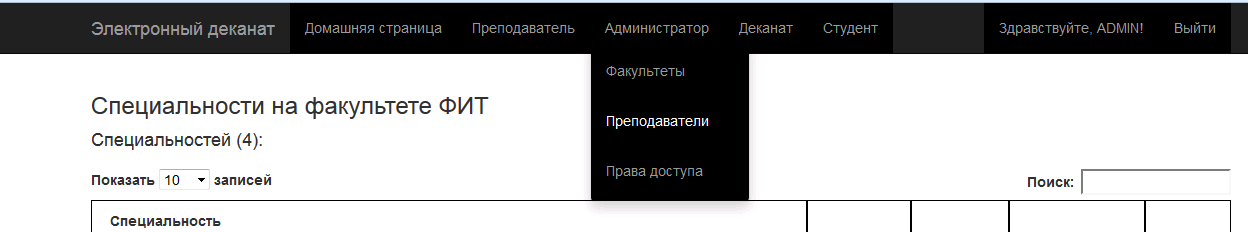


Рисунок – Администратор, горизонтальное меню

На этой странице, можно увидеть всех зарегистрированных преподавателей, редактировать и удалять записи.

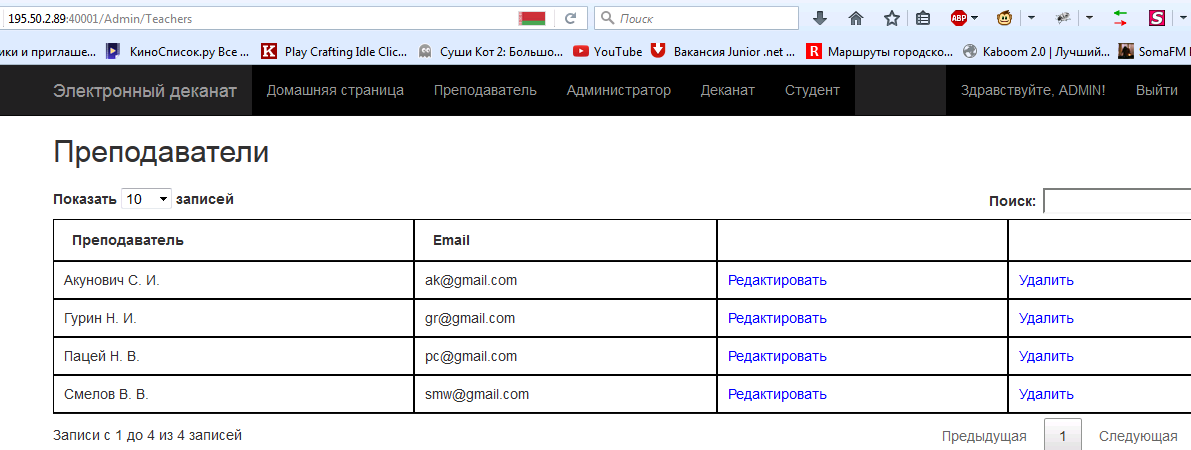


Рисунок – Администратор, список преподавателей

Перейдём на вкладку «права доступа»:

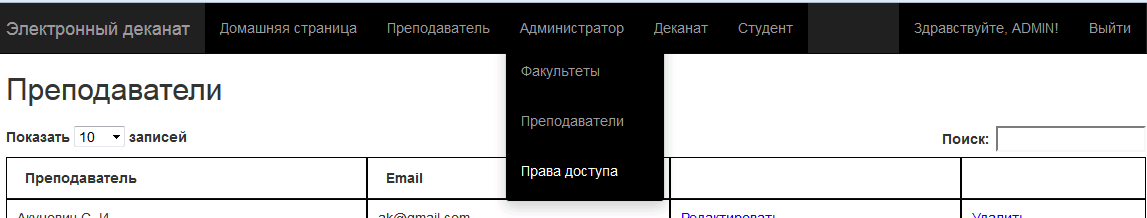


Рисунок – Администратор, горизонтальное меню

Посде нажатия:

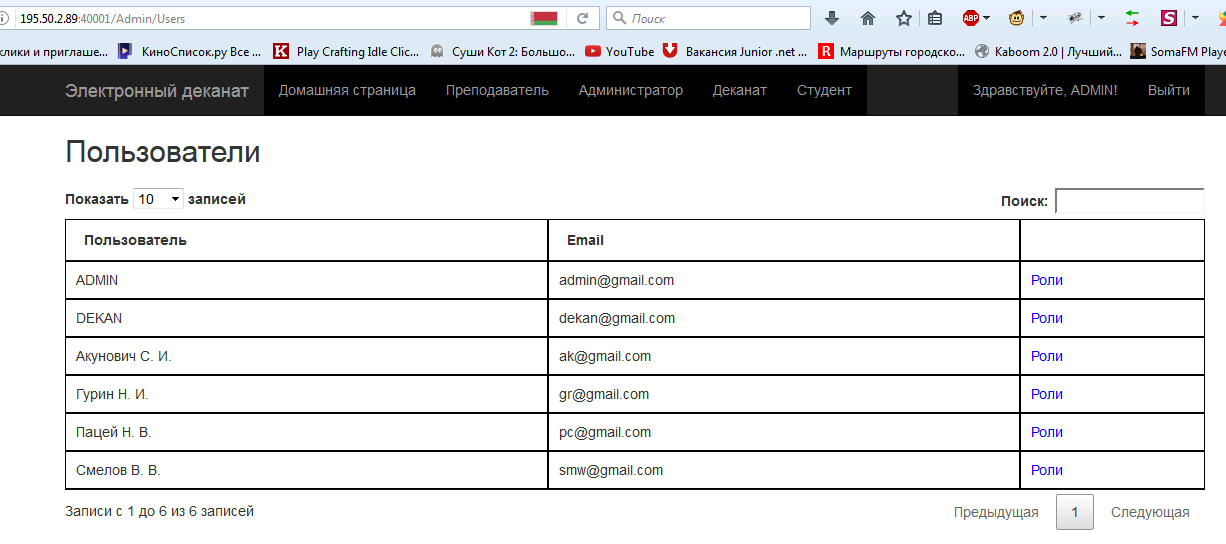


Рисунок – Администратор, страница пользователей

На этой станице, видно всех пользователей, включая деканов и администраторов, а так же, можно просмотреть роли каждого из них. Нажмём на ссылку «роли» напротив Смелов В.В.:

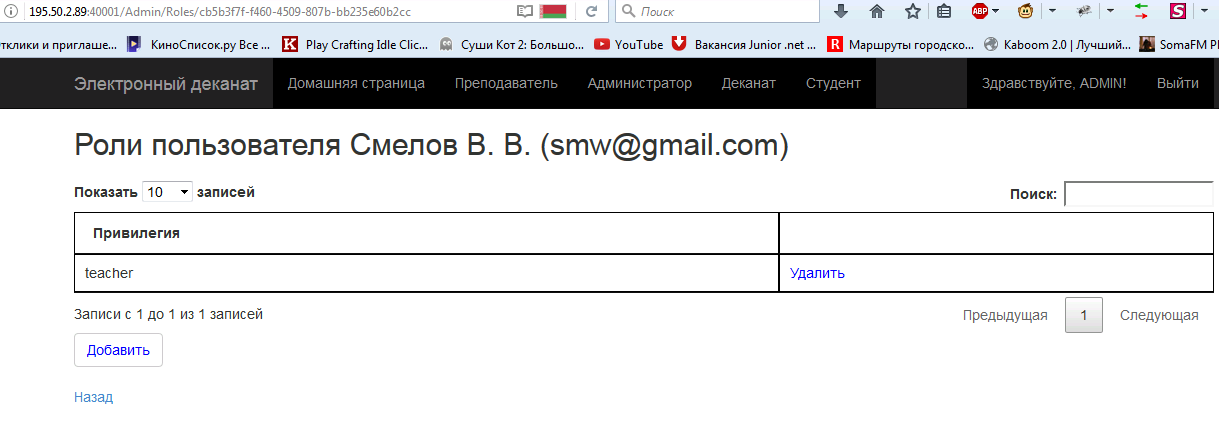


Рисунок – Администратор, страница ролей

На этой странице, мы видим, что единственная назначенная роль этому пользователю в системе на текущий момент, это «преподаватель», а так же, есть кнопки для добавления и удаления ролей.

* 1. **Преподаватель**

Для преподавателя была создана учётная запись (Логин smw@gmail.com, пароль 123). Преподаватель может добавлять, удалять и редактировать лабораторные работы на всех преподаваемых им дисциплинах, а так же, отмечать факт защиты студентом лабораторной работы. Окно, открывающееся после авторизации:

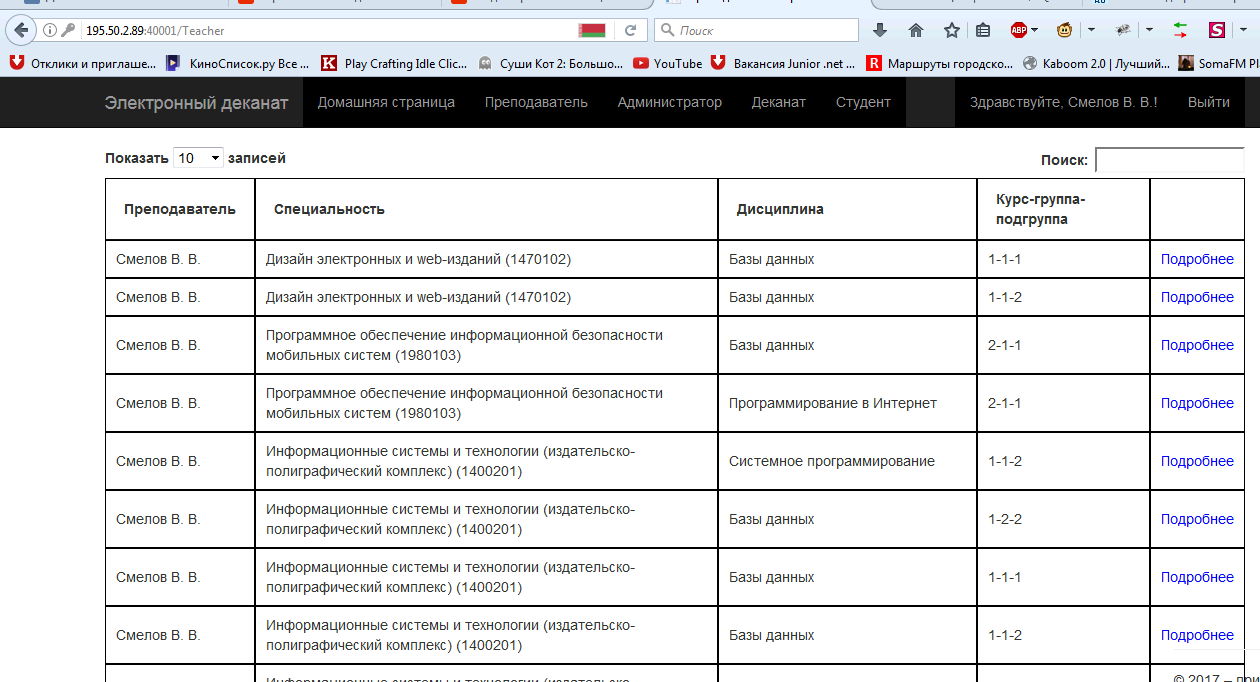
****

Рисунок – Преподаватель, страница выбора подгруппы и дисциплины

На этой странице можно увидеть, какие дисциплины, и у каких групп преподаются, с помощью поиска отфильтровать данные, а так же, перейти к подробному представлению успеваемости конкретной подгруппы. Нажимаем на одну из них, получаем вот такое окошко:

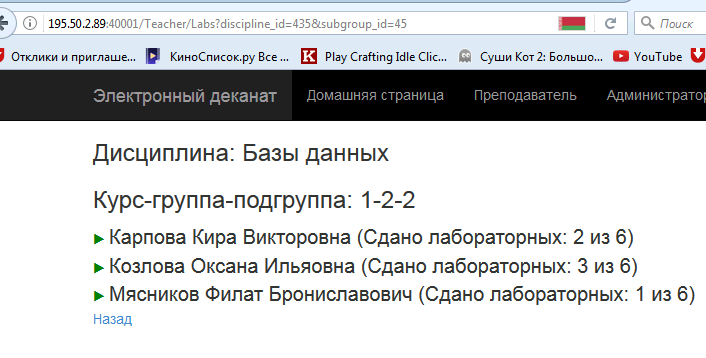


Рисунок – Преподаватель, выбранная подгруппа

Нажав на любую из записей, она разворачивается в следующую таблицу:

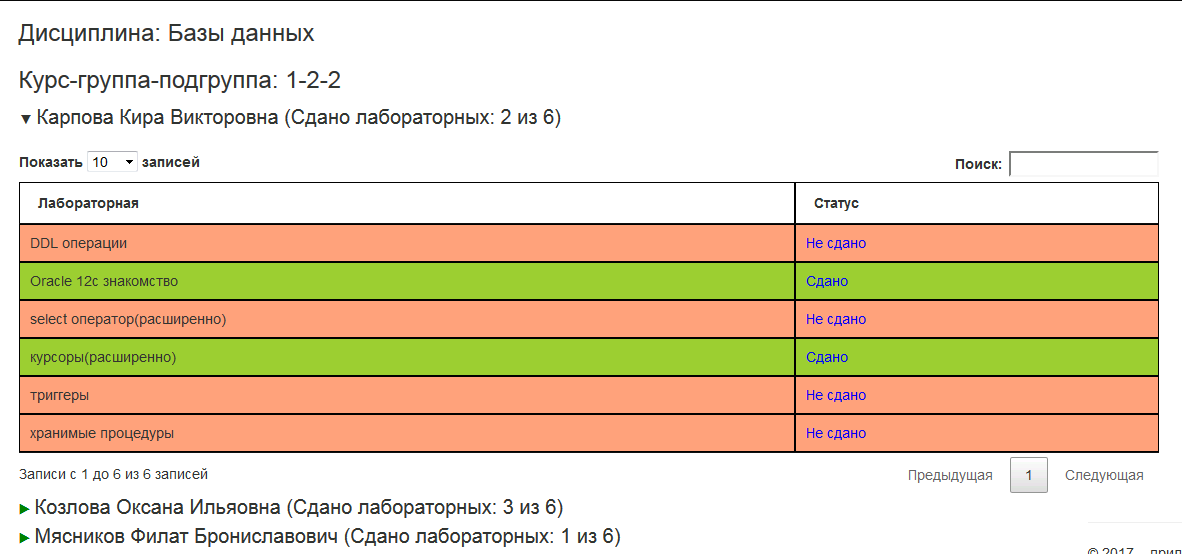


Рисунок – Преподаватель, информация о студенте

Нажав на «статус», переходим в следующее окно, в котором можно поменять статус защиты лабораторной для выбранного студента:

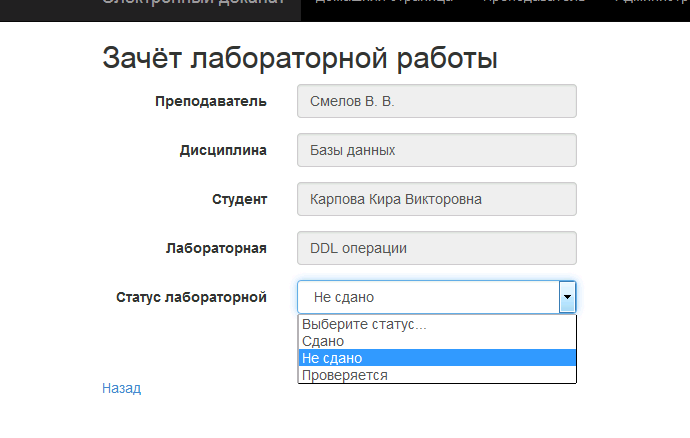
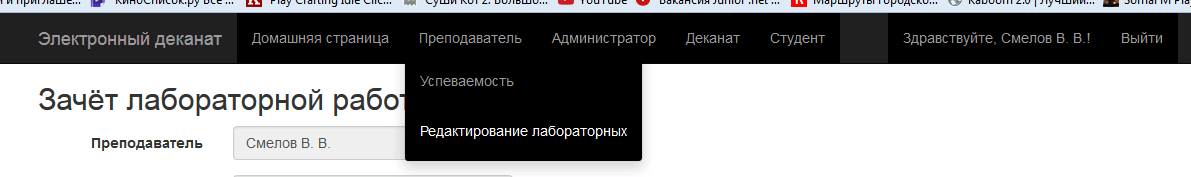


Рисунок – Преподаватель, защита лабораторной

Нажав на вкладку «редактирование лабораторных» попадаем на следующую страницу:



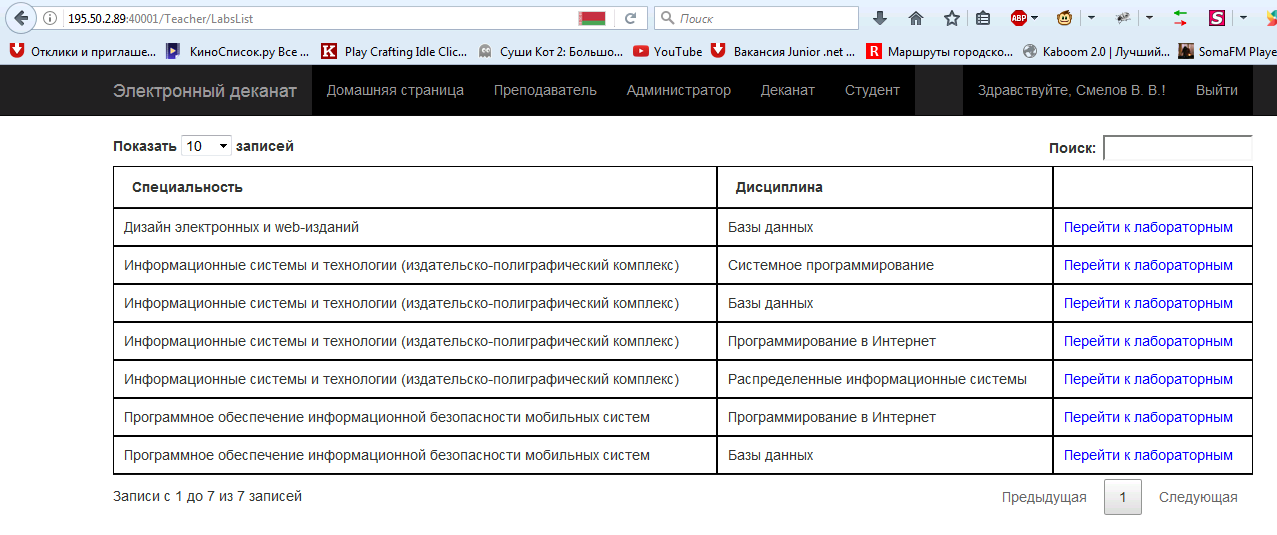


Рисунок – Преподаватель, список дисциплин

На этой странице, можно просмотреть, на каких специальностях какие дисциплины ведёт преподаватель, который авторизован на данный момент в системе, а так же, посмотреть лабораторные работы на выбранной дисциплине, нажав на «перейти к лабораторным»:

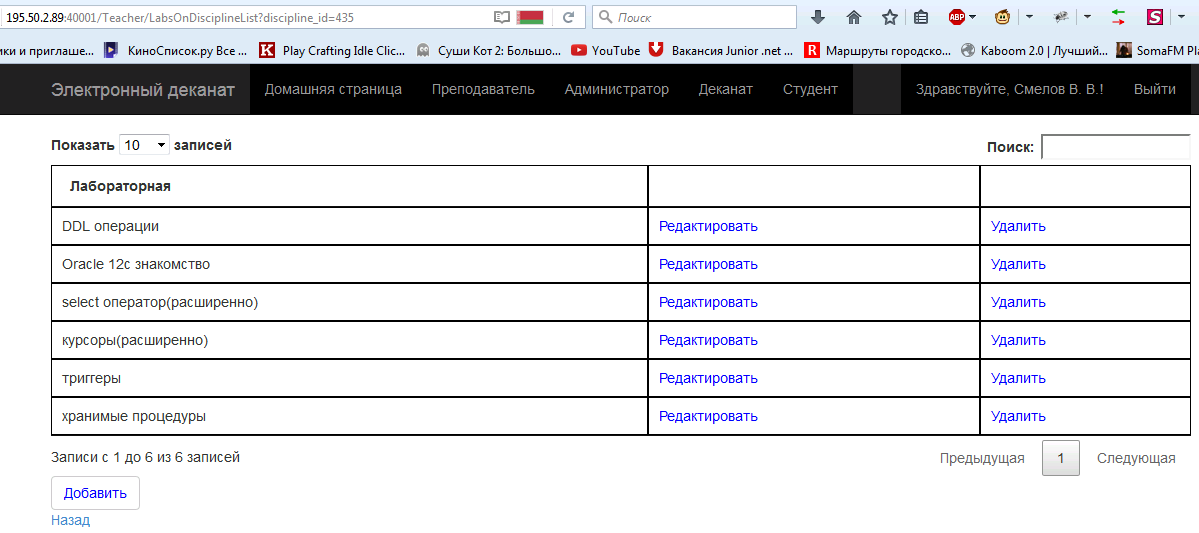


Рисунок – Преподаватель, список лабораторных работ

С этой страницы, можно создавать, добавлять и удалять лабораторные работы.

* 1. **Студент**

На странички, относящиеся к студентам, можно попасть без авторизации. Их всего 2, это поиск по студентам, и подробная информация о студенте. Страница поиска выглядит так:

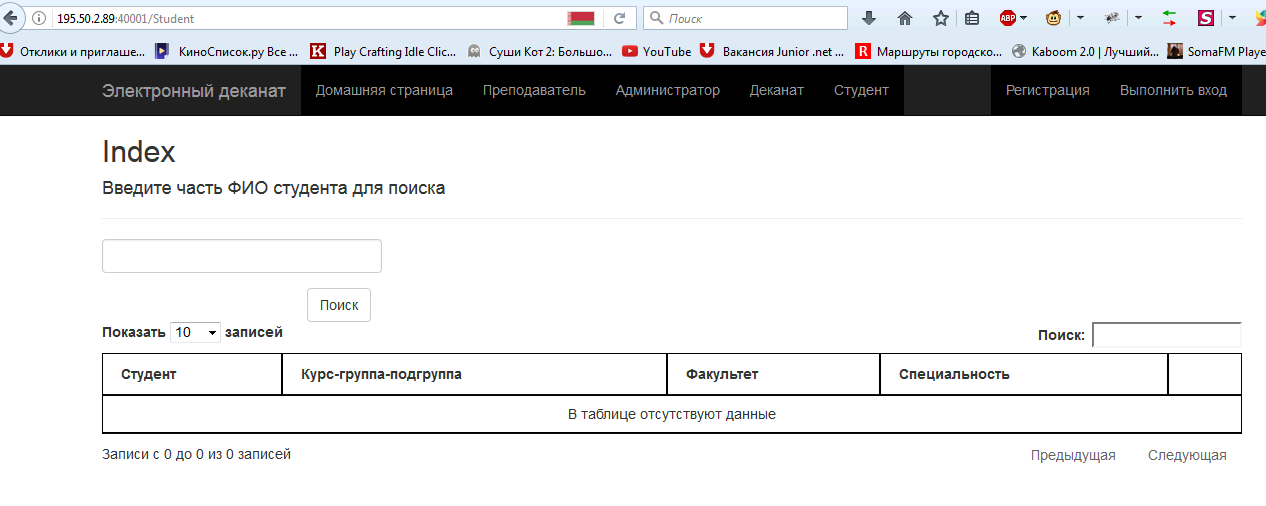


Рисунок – Форма поиска студентов

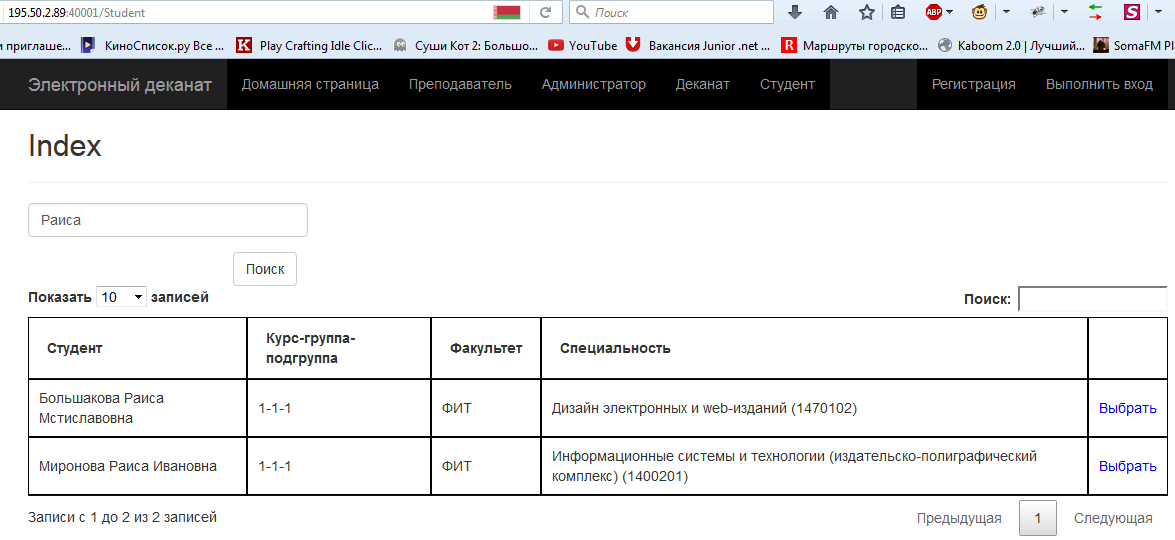


Рисунок – Список найденных студентов

Поиск осуществляется по части строки, например, введём «Раиса», видим, что нашло 2 студентов с таким именем. Нажав «выбрать» открывается страничка, на которой можно увидеть успеваемость выбранного студента, а так же, какие лабораторные нужно защитить этому студенту:

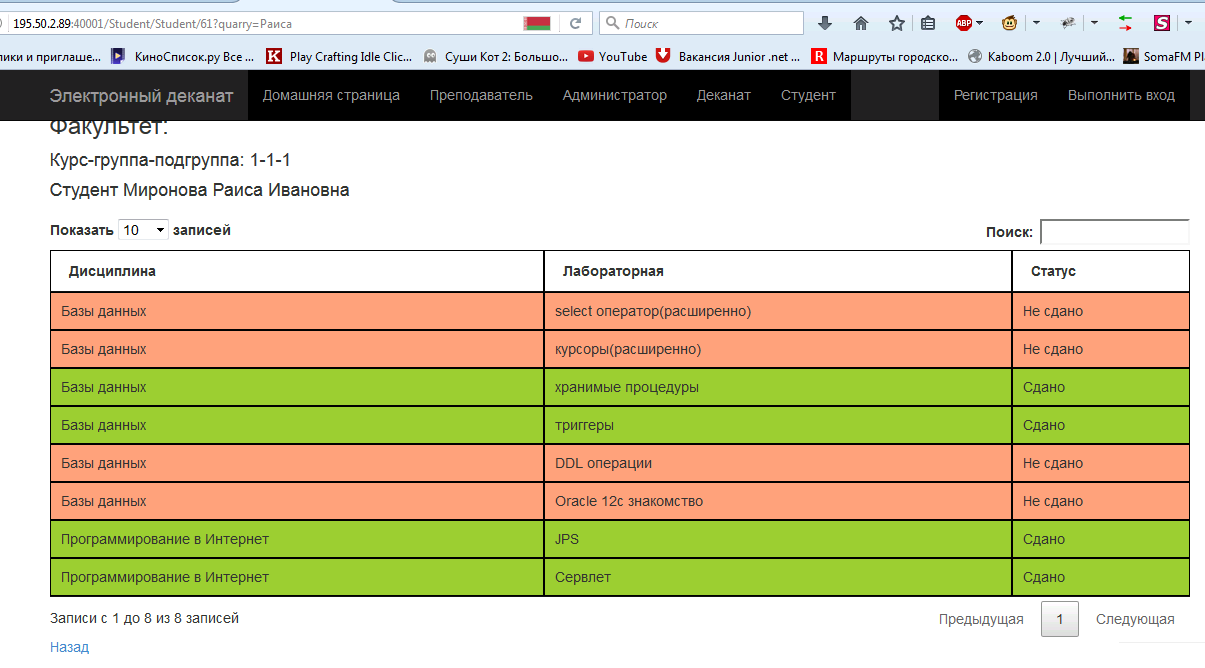


Рисунок – Информация о студенте

* 1. **Деканат**

Для деканата была создана учётная запись (Логин dekan@gmail.com, пароль 12345). Деканат может просматривать успеваемость как по всем группам, так и по конкретным студентам . Окно, открывающееся после авторизации:

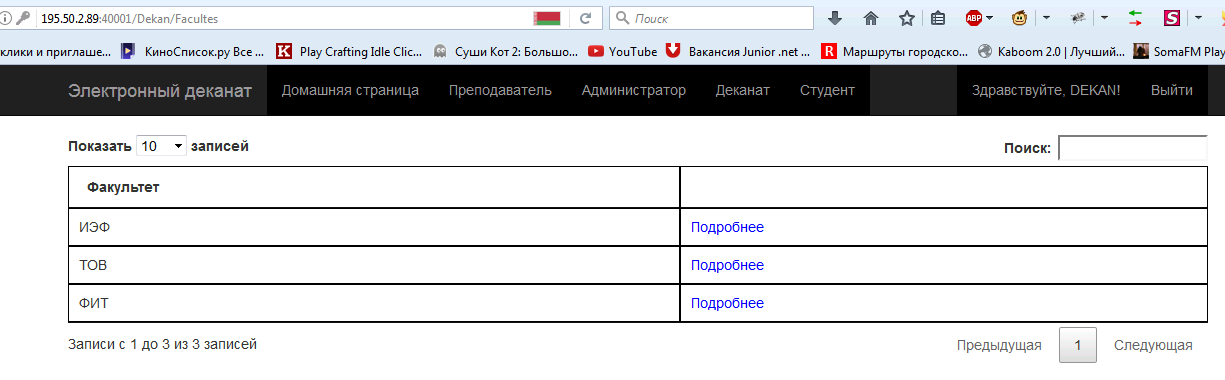
****

Рисунок – Деканат, список факультетов

На этой странице, мы выбираем факультет кнопкой «подробнее», нажимаем на ссылку напротив факультета «ФИТ»:

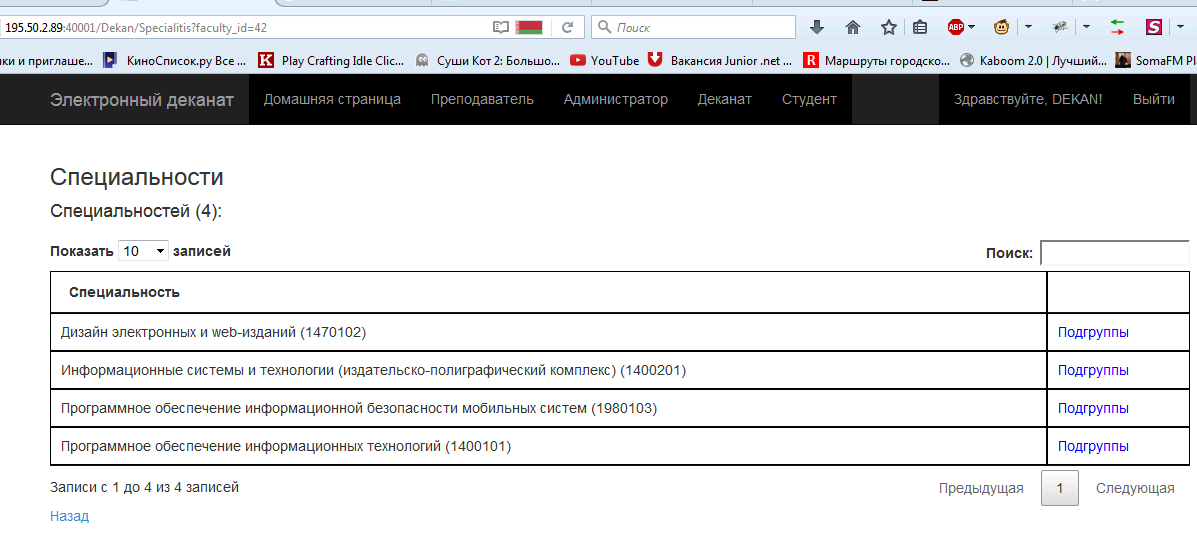


Рисунок – Деканат, список специальностей

На этой странице, видны все специальности на факультете, нажимаем на «подгруппы» напротив специальности «ИСиТ»:

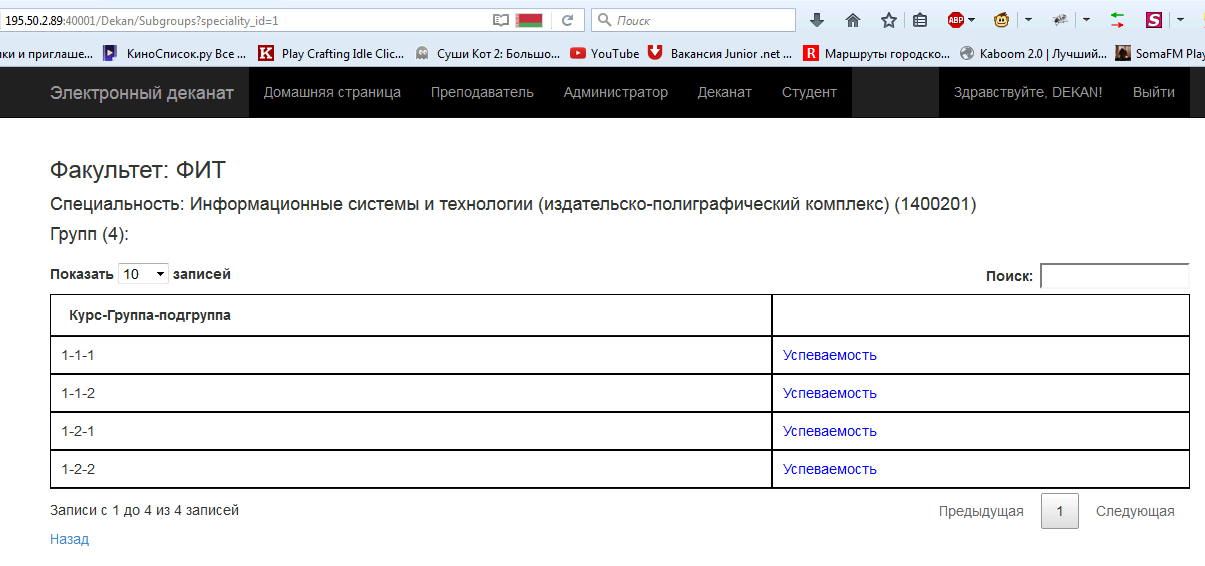


Рисунок – Деканат, список групп

На этой странице, видны все подгруппы на специальности, нажимаем на «успеваемость» подгруппы 1-1-2:

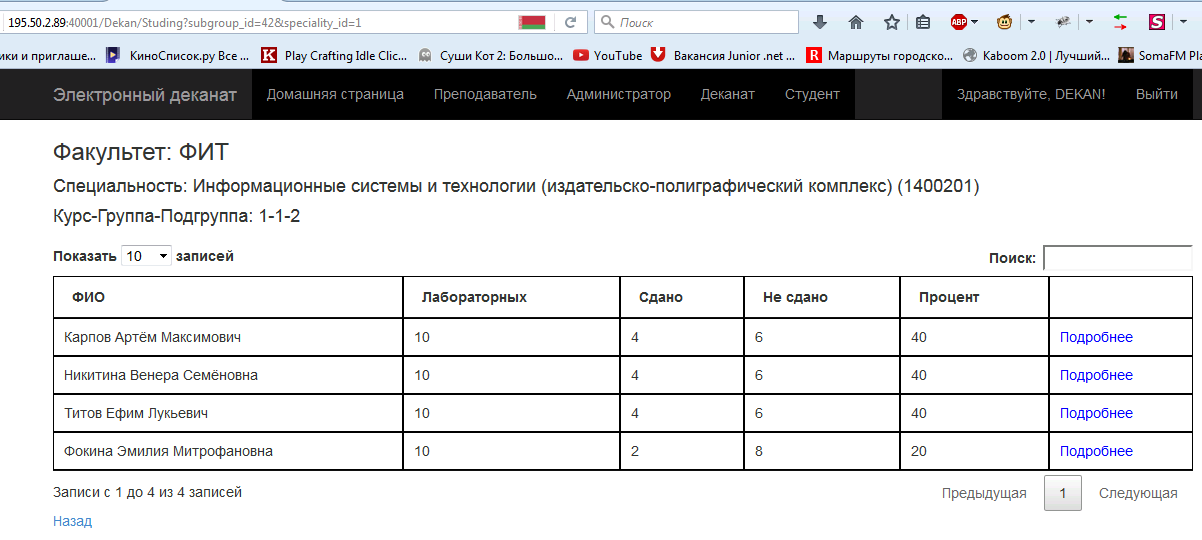


Рисунок – Деканат, список успеваемости студентов

На этой странице, видны все студенты в выбранной подгруппе, а так же, суммарная успеваемость каждого студента, при необходимости посмотреть подробнее, нажимаем на «успеваемость» нужного студента в списке:

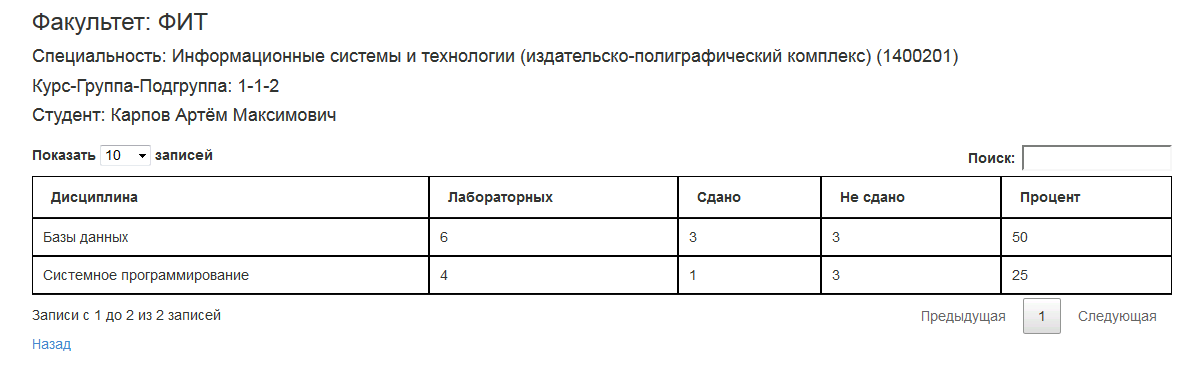


Рисунок – Деканат, список успеваемости студента

На этой странице мы видим успеваемость студента по каждой дисциплине по отдельности.

**5. Экономический раздел**

**6. Мероприятия по охране труда и безопасности жизнедеятельности**

**7. Заключение**

# Список использованных источников

1. Кафедра Интеллектуальных Информационных Технологий ИнФО УрФУ [Электронный ресурс] – Режим доступа: <http://lecturesdb.readthedocs.io/databases/dbms.html>. Дата доступа: 15.04.2017. 2. Частное Боровское исследовательское учреждение по внедрению новых технологий [Электронный ресурс] – Режим доступа: <http://bourabai.ru/dbt/servers/oracle2.htm>. Дата доступа: 15.04.2017.

3. Абитуриент БГТУ [Электронный ресурс] – Режим доступа: <https://abiturient.belstu.by/foreign-abiturient.html>. Дата доступа: 15.04.2017.

4. Википедия [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio>. Дата доступа: 15.04.2017.

5. Mozilla Developer Network [Электронный ресурс] – Режим доступа: <https://developer.mozilla.org/ru/docs/Learn/Что_такое_веб_сервер>. Дата доступа: 15.04.2017.

6. Википедия [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Microsoft_SQL_Server>. Дата доступа: 15.04.2017.

7. HyperHost хостинг [Электронный ресурс] – Режим доступа: <https://hyperhost.ua/info/?p=3007>. Дата доступа: 15.04.2017.

8. Википедия [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Сервер_приложений>. Дата доступа: 15.04.2017.

9. developerWorks [Электронный ресурс] – Режим доступа: <https://www.ibm.com/developerworks/ru/library/wa-mean1/>. Дата доступа: 15.04.2017.

10. Википедия [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Spring_Framework>. Дата доступа: 15.04.2017.

11. Web Creator [Электронный ресурс] – Режим доступа: <https://web-creator.ru/articles/dot_net_and_asp>. Дата доступа: 15.04.2017.

12. РегФармПатент[Электронный ресурс] – Режим доступа: <http://regpharmpat.by/ru/patentnye-issledovaniya>. Дата доступа: 15.04.2017.

13. Хабрахабр[Электронный ресурс] – Режим доступа: <https://habrahabr.ru/post/202018/>. Дата доступа: 15.04.2017.

14. metanit.com[Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/entityframework/1.1.php>. Дата доступа: 15.04.2017.

15. Справочник по паттернам проектирования [Электронный ресурс] – Режим доступа: <http://design-pattern.ru/patterns/repository.html>. Дата доступа: 15.04.2017.

16. dedushka.org[Электронный ресурс] – Режим доступа: <http://dedushka.org/uroki/6901.html>. Дата доступа: 15.04.2017.

17. Википедия [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/JQuery>. Дата доступа: 15.04.2017.

18. htmlhook.ru[Электронный ресурс] – Режим доступа: <https://htmlhook.ru/rabota-s-jquery-datatables.html>. Дата доступа: 15.04.2017.

19. Федеральный институт промышленной собственности [Электронный ресурс] – Режим доступа: <http://www1.fips.ru>. Дата доступа: 15.04.2017.

20. webdom.net [Электронный ресурс] – Режим доступа: <http://www.webdom.net/inf/asp>. Дата доступа: 15.04.2017.

21. orastack.narod.ru [Электронный ресурс] – Режим доступа: <http://orastack.narod.ru/index_file/onebd.html>. Дата доступа: 15.04.2017.