МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

**ФГБОУ ВО «БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**Кафедра «Информатика и программное обеспечение»**

**«У Т В Е Р Ж Д А Ю»**

Зав. кафедрой «И и ПО», к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подвесовский А.Г.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

**Подсистема учёта результатов аттестации студентов и интеграционный программный интерфейс для автоматизированной системы мониторинга успеваемости студентов и посещаемости занятий «СУП»**

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

Всего \_\_\_\_ листов

**Научный руководитель**

к.т.н., доц. Булатицкий Д.И.

« \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

**Нормоконтролер**

к.т.н., доц. Лагерев Д.Г.

« \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

**Магистрант**

Исаев И.С.

« \_\_\_ »\_\_\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

БРЯНСК 2017

**АННОТАЦИЯ**

На основании задания на выполнение магистерской диссертации была спроектирована подсистема учёта результатов аттестации студентов и интеграционный программный интерфейс для автоматизированной системы мониторинга успеваемости студентов и посещаемости занятий «СУП».

В данном документе описаны решения, использованные при создании подсистемы, проведен анализ требований, а также другие аспекты разработки.

Анализ требований содержит подробное описание задач, которые должна решать разрабатываемая подсистема. Также, приведен анализ аналогичных работ в данной области и разработок. Основной задачей данной части является формулировка требований к разрабатываемой подсистеме.

В исследовательской части работы проанализированы способы взаимодействия распределенного приложения и выбран оптимальный из них. Спроектирован протокол взаимодействия с сервисом учета контингент. Также разработаны алгоритмы оценивания работы студентов в семестре.

Раздел «Разработка подсистемы учета успеваемости» содержит описание архитектуры подсистемы и проектные модели подсистемы. Также дано описание составляющих компонентов разработанного программного продукта.

В экспериментальной части описаны методики отладки и тестирования, примененные при разработке данного приложения.

**Содержание**

[ВВЕДЕНИЕ 6](#_Toc484679190)

[1. УЧЕТ РЕЗУЛЬТАТОВ АТТЕСТАЦИИ СТУДЕНТОВ В ВУЗАХ 8](#_Toc484679191)

[1.1. Виды аттестации студентов 8](#_Toc484679192)

[1.1.1. Итоговая аттестация 8](#_Toc484679193)

[1.1.2. Промежуточная аттестация 9](#_Toc484679194)

[1.1.3. Внутрисеместровая аттестация 10](#_Toc484679195)

[1.1.4. Текущая аттестация 10](#_Toc484679196)

[1.2. Обзор работ в области учета результатов аттестации студентов 11](#_Toc484679197)

[1.3. Обзор программ аналогов 16](#_Toc484679198)

[1.3.1. Электронный журнал успеваемости «Магеллан» 16](#_Toc484679199)

[1.3.2. Система учета абитуриентов и студентов. Компания Тауруна 16](#_Toc484679200)

[1.3.3. Программный комплекс «ЭЛЕКТРОННЫЕ ВЕДОМОСТИ» 17](#_Toc484679201)

[1.4. Формализация процесса учета результатов аттестации студентов 18](#_Toc484679202)

[1.5. Системы учета результатов аттестации студентов на кафедре 23](#_Toc484679203)

[1.6. Смежные информационные сервисы на кафедре 24](#_Toc484679204)

[1.7. Выводы по главе 25](#_Toc484679205)

[2. РАЗРАБОТКА АЛГОРИТМОВ ОЦЕНИВАНИЯ РАБОТЫ СТУДЕНТОВ В СЕМЕСТРЕ И ПУТЕЙ ИНТЕГРАЦИИ СИСТЕМЫ СО СМЕЖНЫМИ СЕРВИСАМИ 28](#_Toc484679206)

[2.1. Анализ технологий взаимодействия партнерских сервисов или компонентов распределенного приложения 29](#_Toc484679207)

[2.1.1. Архитектурный стиль REST API 29](#_Toc484679208)

[2.1.2. Архитектурный стиль JSON API 30](#_Toc484679209)

[2.1.3. Протокол XML-RPC 30](#_Toc484679210)

[2.1.4. Протокол SOAP 31](#_Toc484679211)

[2.2. Взаимодействие с сервисом учета контингента 31](#_Toc484679212)

[2.2.1. Программный интерфейс сервиса учета контингента 31](#_Toc484679213)

[2.2.2. Проектирование формата взаимодействия с сервисом учета контингента 32](#_Toc484679214)

[2.3. Разработка алгоритмов оценивания работы студентов в семестре 34](#_Toc484679215)

[2.4. Выводы по главе 36](#_Toc484679216)

[3. РАЗРАБОТКА ПОДСИСТЕМЫ УЧЕТА РЕЗУЛЬТАТОВ АТТЕСТАЦИИ СТУДЕНТОВ 38](#_Toc484679217)

[3.1. Этапы разработки подсистемы 38](#_Toc484679218)

[3.2. Формирование функциональных требований к веб-интерфейсу подсистемы учета результатов аттестации студентов 39](#_Toc484679219)

[3.2.1. Возможности роли «Гость» 39](#_Toc484679220)

[3.2.2. Возможности роли «Преподаватель» 40](#_Toc484679221)

[3.2.3. Возможности роли «Куратор» 40](#_Toc484679222)

[3.2.4. Возможности роли «Администратор» 40](#_Toc484679223)

[3.2.5. Функциональная модель подсистемы 41](#_Toc484679224)

[3.3. Формирование функциональных требований к программному интерфейсу для интеграции с другими сервисами 41](#_Toc484679225)

[3.4. Проектирование системы 42](#_Toc484679226)

[3.4.1. Архитектура системы 42](#_Toc484679227)

[3.4.2. Проектирование взаимодействия компонентов подсистемы учета результатов аттестации студентов 45](#_Toc484679228)

[3.4.3. Проектирование пользовательского интерфейса веб-приложения 46](#_Toc484679229)

[3.4.4. Разработка программного интерфейса для манипулирования данными в системе мониторинга посещаемости и успеваемости 53](#_Toc484679230)

[3.4.5. Разработка механизмов взаимодействия с сервисом учета контингента 55](#_Toc484679231)

[3.5. Выбор средств разработки 56](#_Toc484679232)

[3.6. Модель данных 57](#_Toc484679233)

[3.7. Программная реализация подсистемы учета результатов аттестации 59](#_Toc484679234)

[3.8. Программная реализация интеграционного программного интерфейса 60](#_Toc484679235)

[4. ТЕСТИРОВАНИЕ И ВЕРИФИКАЦИЯ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 61](#_Toc484679236)

[4.1. Проверка выполнения функциональных требований 61](#_Toc484679237)

[4.1.1. Проверка работы модуля учета результатов текущей аттестации 61](#_Toc484679238)

[4.1.2. Проверка работы модуля учета результатов внутрисеместровой аттестации 61](#_Toc484679239)

[4.1.3. Проверка работоспособности программного интерфейса 61](#_Toc484679240)

[4.1.4. Проверка модуля взаимодействия с сервисом учета контингента 62](#_Toc484679241)

[4.2. Опытная эксплуатация 62](#_Toc484679242)

[4.2.1. Работа с веб-интерфейсами подсистемы учета результатов аттестации 62](#_Toc484679243)

[4.2.2. Применение механизмов, основанных на алгоритмах оценивания работы студентов в семестре 63](#_Toc484679244)

[4.2.3. Наладка интеграции с сервисом учета контингента 64](#_Toc484679245)

[4.3. Выводы по главе 66](#_Toc484679246)

[ЗАКЛЮЧЕНИЕ 67](#_Toc484679247)

[СПИСОК ЛИТЕРАТУРЫ 68](#_Toc484679248)

[ПРИЛОЖЕНИЕ А 72](#_Toc484679249)

[ПРИЛОЖЕНИЕ Б 73](#_Toc484679250)

# ВВЕДЕНИЕ

Информационные и коммуникационные технологии успешно внедряются в различные сферы человеческой деятельности, в том числе в сферу образования. Развитие таких технологий в образовании призвано способствовать совершенствованию организации образовательного процесса, взаимодействию всех его участников и, в конечном счете, повышению успеваемости обучающихся и качества образования.

Как правило, в университетах данные о сдаче лабораторных работ, о результатах промежуточной и внутрисеместровой аттестации хранятся на бумажных носителях, которые необходимо заполнять вручную, что затрудняет процесс учета. Так же из-за этого у студентов отсутствует оперативный доступ к данным. Ведение данных об успеваемости в электронном виде позволит ускорить обработку данных, снизить вероятность ошибок.

**Целью данной работы** является расширение инструментария для учета успеваемости студентов в системе учета посещаемости и успеваемости и разработка программного интерфейса для интеграции данной системы со  сторонними сервисами.

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

* анализ процесса учета и видов аттестации студентов, а также способов взаимодействия различных сервисов друг с другом;
* формирование функциональных требований к веб-интерфейсу системы учета результатов аттестации и к программному интерфейсу для интеграции системы учета результатов аттестации с другими сервисами;
* разработка подсистемы учета результатов аттестации студентов;
* разработка программного интерфейса для интеграции системы мониторинга посещаемости и успеваемости студентов с другими сервисами;
* опытная эксплуатация системы и апробация разработанных моделей и методов на реальных данных.

**Объектом исследования** выступают процессы учета результатов текущей и промежуточной аттестации студентов, а также взаимодействие программных систем посредством программных интерфейсов.

**Предметом исследования** является автоматизация процесса учета результатов аттестации студентов и разработка программного интерфейса для интеграции со смежными сервисами.

**Методология и методы исследования.** При выполнении исследований и решении поставленных задач использовались основные положения и методы системного анализа, объектно-ориентированного проектирования, теории реляционных баз данных и методы проектирования клиент-серверной архитектуры системы.

**Теоретическая значимость** диссертационной работы заключается в разработке алгоритмов и формул для автоматического расчета оценок по внутрисеместровой и промежуточной аттестациям.

**Практическая ценность** работы заключается в следующем:

Разработано и внедрено программное обеспечение для учета результатов аттестации студентов, а также разработан программный интерфейс для интеграции со сторонними сервисами.

# УЧЕТ РЕЗУЛЬТАТОВ АТТЕСТАЦИИ СТУДЕНТОВ В ВУЗАХ

Успеваемость – степень успешности усвоения учебных предметов учащимися. Проверка успеваемости производится посредством аттестации студентов.

Аттестация – подтверждение уровня знаний, умений и навыков обучаемых по отношению к объёму и содержанию учебных дисциплин, представленных и утверждённых в учебных планах и учебных программах [15, 16].

## Виды аттестации студентов

Основные подходы и виды аттестации регулируются федеральным законом «Об образовании в Российской Федерации» и  федеральными государственными образовательными стандартами, а виды и перечень мероприятий и работ по контролю успеваемости выбираются по усмотрению учебных заведений [1, 15, 16].

### Итоговая аттестация

В качестве **итоговой аттестации** можно выделить государственный экзамен и выпускную квалификационную работу.

Государственный экзамен – это выпускная работа, которая сдается в  высших и среднеспециальных учебных заведениях. Государственный экзамен проводится в присутствии государственной аттестационной комиссии, в которую могут входить: ректор (или проректор) учебного заведения, декан соответствующего факультета (или его заместитель), заведующие кафедрами, профессора и доценты. Государственный экзамен вводится по усмотрению вуза.

Выпускная квалификационная работа – самостоятельная творческая работа студентов, выполняемая ими на последнем, выпускном курсе.

Данная работа обычно выполняется студентами, обучающимися по естественнонаучным, гуманитарным, экономическим и творческим специальностям, и имеет целью систематизацию, обобщение и проверку специальных теоретических знаний и практических навыков выпускников. После написания работы, студенту необходимо ее защитить перед комиссией, состав которой может варьироваться. По итогам защиты студенту выставляется оценка («отлично», «хорошо», «удовлетворительно» и «неудовлетворительно»), которая в совокупности с оценкой по государственному экзамену (если он есть), позволяет принять решение о выдаче диплома данному студенту.

По итогам государственного экзамена и защиты выпускной квалификационной работы студенту выставляется оценка «отлично», «хорошо» или «удовлетворительно».

### Промежуточная аттестация

**Промежуточная аттестация** студентов в вузах является формой контроля качества знаний студентов по дисциплине за семестр с целью комплексного определения соответствия уровня и качество знаний, умений и навыков студентов требованиям, установленным рабочей программы дисциплины. Студенты могут сдавать экзамены и зачеты по факультативным дисциплинам. Результаты сдачи вносятся в ведомость, зачетную книжку и выписку из зачетной книжки.

Видами промежуточной аттестации выступают зачет/дифференцированный зачет или экзамен. Конкретный вид промежуточной аттестации определяется учебным планом.

Зачеты, как правило, являются формой промежуточной аттестации дисциплин с трудоемкостью менее 3 зачетных единиц, но могут использоваться и для дисциплин с большей трудоемкостью и проводиться в форме дифференцированного зачета. Результаты сдачи зачетов оцениваются отметкой «зачтено». Зачеты дифференцированными оценками по пятибалльной шкале проставляются по курсовым проектам (работам), практикам и отдельным дисциплинам согласно учебному плану.

Экзамен предполагает проверку учебных достижений обучаемых, как правило, по всей программе дисциплины. Успеваемость студентов определяется оценками «отлично», «хорошо», «удовлетворительно» и «неудовлетворительно».

### Внутрисеместровая аттестация

**Внутрисеместровая аттестация** является обязательной формой контроля для студентов первого и второго курса высшего учебного заведения, обучающихся по очной форме. Для студентов старших курсов аттестации проводятся по представлению деканата/директората с целью контроля успеваемости в отдельных группах.

Как правило, аттестация проводится два раза в семестр. Студенты аттестуются путем выставления в соответствующую групповую ведомость дифференцированной оценки (от 1 до 5) по дисциплинам, вынесенным на аттестацию деканатом или директоратом.

При аттестации студентов учитываются результаты работы на занятиях, выполнения контрольных работ, домашних заданий, посещение студентами занятий, своевременная ликвидация задолженностей.

### Текущая аттестация

**Текущая аттестация** в вузе является формой контроля качества знаний студентов, осуществляемого в межсессионный период обучения с целью определения качества освоения основных образовательных программ высшего образования.

Данный вид контроля проводится, как правило, в устной (опрос, защита работы, доклад и т.д.), письменной (письменный опрос, выполнение определенных заданий и т.д.) или тестовой (устное, письменное, компьютерное тестирование) форме.

Результаты текущего контроля используются преподавателями в целях своевременного выявления отстающих студентов и оказания им содействия; анализа качества используемой рабочей программы и совершенствования методики ее преподавания; доведения до студентов и иных заинтересованных лиц информации о степени освоения студентами программы [1].

Существует множество разнообразных видов работ для учета текущей аттестации студентов, при этом практически на каждой кафедре есть свои особенности. Например, на математических кафедрах, помимо лабораторных работ или вместо них, для учета текущей аттестации используются самостоятельные и контрольные работы, на кафедре иностранных языков присутствует множество специфических работ для учета успеваемости такие, как пересказы, словарные диктанты, написание студентами различных эссе. Таким образом, для хранения списка всевозможных видов работ в базе целесообразно создать некий справочник, который будет периодически дополняться администратором или преподавателями.

Как различаются виды работ для учета текущей аттестации, так и разнятся способы оценки выполнения данных работ. Преподаватели могут использовать 5-ти балльную, 10-ти или 100-балльную систему оценок или же вообще отмечать этапы сдачи работы студентом графическими фигурами или символами. Данный фактор также необходимо учесть при разработке системы.

## Обзор работ в области учета результатов аттестации студентов

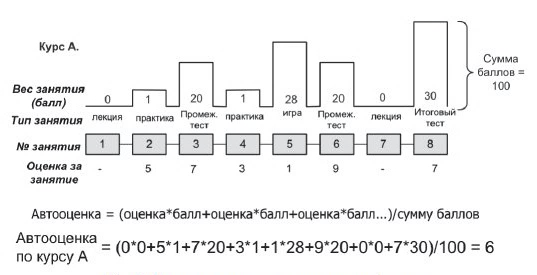
Чуйко О.И. в своей работе рассматривает балльно-рейтинговую систему оценки текущей аттестации студентов и предлагает следующее распределение баллов:

* посещение занятия – от 0 до 1 балла в зависимости от работы студента на занятии;
* защита лабораторных работ – от 1 до 5 баллов в зависимости от качества выполнения работы и ответов на вопросы преподавателя;
* контрольная работа – от 1 до 5 баллов в зависимости от правильности выполнения и ответов на вопросы;
* дополнительные баллы – от 0 до 10 баллов – назначаются за доклад, реферат, статью, презентацию и другие дополнительные задания [2].

При автоматическом заполнении аттестационных ведомостей такая система оценок позволяет учитывать посещаемость студента. Также она позволяет разделять работы по степени важности, отдавая больше дополнительных баллов более значимым видам работ.

В работе С.Р. Гурикова для учета успеваемости вводится понятие «система оценивания». С точки зрения автора, система оценивания – это технология, включающая в себя автоматический расчет оценок промежуточной аттестации обучающихся по  дисциплинам образовательной программы на основе трех параметров: посещаемости, текущих оценок и контрольных мероприятий.

Для расчета оценки по дисциплине предлагается схема, представленная на рисунке 1.1.



Риc. 1.1. Правила расчета автоматической оценки

Автооценка, рассчитанная программой, учитывает количество пропущенных студентом занятий, весовые коэффициенты занятий и, безусловно, текущую оцениваемую деятельность студента. Автооценка на уровне двух баллов в 10-балльной системе позволяет преподавателю оценить студента как удовлетворительно успевающего по дисциплине, что способствует повышению абсолютной успеваемости [3].

Кроме методики расчета автооценки, С.Р. Гуриков предложил формировать оценку промежуточной аттестации по дисциплине с учетом трех параметров: автооценки, посещаемости и контрольной экзаменационной оценки, но с применением коэффициентов k1, k2, k3. Их значение может варьироваться от нуля до единицы и зависит от важности вышеупомянутых параметров. Значения коэффициентов устанавливает преподаватель, ведущий дисциплину, причем сумма всех коэффициентов должна быть равна единице.

ИО = k1 \* АО + k2 \* АО + k3 \* АО, (1) где: ИО – итоговая оценка, АО – автооценка, k1, k2, k3 – коэффициенты.

Борисов З.В. при разработке системы мониторинга посещаемости и учета успеваемости студентов для расчетов показателей текущей и промежуточной аттестации использовал следующие формулы.

Формула расчета коэффициента текущей аттестации *Ucur* для студента:

|  |  |
| --- | --- |
| , | (Формула1.1) |

где *Uy* – подмножество оценок по текущей аттестации, *Wcount* – количество работ в семестре.

Коэффициент *Ucur* находится в диапазоне допустимых значений (0;5] и показывает коэффициент текущей аттестации студента. Чем ближе полученное значение к 5, тем лучше студент усвоил знания.

Формула расчета коэффициента промежуточной аттестации *Uinterim*для группы студентов:

|  |  |
| --- | --- |
| , | (Формула1.2) |

где *U1* – подмножество оценок за первую аттестацию, *U2* – подмножество оценок за вторую аттестацию, *Scount* – множество студентов.

Коэффициент *Uinterim*находится в диапазоне допустимых значений (0;5] и  показывает коэффициент промежуточной аттестации группы студентов. Чем ближе полученное значение к 5, тем лучше студенты усвоили знания [4, 7].

Также Борисов З.В. пишет о целесообразности разработки системы учета посещаемости и успеваемости с точки зрения ускорения и упрощения синхронизации информации, имеющейся у обучающегося и преподавателя.

Жетесова Г.С. и Ерназарова М.А. рассматривают проблему учета успеваемости студентов с точки зрения модульно-рейтинговой системы обучения.

Они пишут, что модульная система имеет целью поставить студента перед необходимостью регулярной учебной работы в течение всего семестра. Это достигается делением преподаваемой дисциплины на крупные блоки, по  завершении которых студент сдает промежуточные (модульные) экзамены. Полученные им баллы за все промежуточные экзамены суммируются и  составляют его рейтинг по данной дисциплине. Весомость этих баллов для студента повышается тем, что при получении достаточной суммы баллов за все промежуточные экзамены их результаты могут засчитываться ему как итоговый (семестровый) экзамен, т.е. модульные экзамены становятся такими же авторитетными, как и семестровые. Этим самым и решается задача приучения студента к систематической учебной работе.

Итоговая оценка по модулю (*Им*), состоящему из одной дисциплины, по которой предусмотрен только экзамен, рассчитывается по формуле [4]:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула1.3) |

где *А1* и *А2* – оценки первого и второго рубежного контроля; *Э* – оценка за экзамен; 0,4 и 0,6 – соответственно коэффициенты весомости экзаменационной оценки и оценок рубежного контроля.

Итоговая оценка по модулю, состоящему из одной дисциплины, (Имс) по которой предусмотрены экзамен и курсовая работа (проект), рассчитывается как среднее число по формуле:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула1.4) |

где *ИМК* – итоговая оценка по курсовой работе (проекту), рассчитывается по формуле:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула1.5) |

где *ЭК* – оценка защиты курсовой работы (проекта).

В случае если модуль состоит из нескольких дисциплин, то определяется доля каждого составного компонента в кредитном выражении к общему количеству кредитов, т.е. коэффициент весомости по каждой дисциплине.

Коэффициент весомости рассчитывается по формуле:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула1.6) |

где *αi* – коэффициент весомости *i*-й дисциплины; *ki* – количество кредитов *i*-й дисциплины; *n* – общее количество кредитов, из которых состоит модуль.

Коэффициенты весомости по каждой дисциплине, входящей в модуль, должны удовлетворять условию:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула1.7) |

где *αi* – коэффициент весомости *i*-й дисциплины.

В случае если модуль состоит из нескольких компонентов (дисциплин), то экзамен проводится по основному компоненту, а по другим компонентам формой итогового контроля являются курсовой проект или работа, тестовые задания, расчетно-графическая работа, расчетная работа, реферат.

Итоговая оценка по модулю, состоящему из нескольких дисциплин, рассчитывается как сумма произведений коэффициентов весомости на итоговую оценку по соответствующей дисциплине:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула1.8) |

где *αi* и *Иi* – соответственно коэффициент весомости и итоговая оценка по *i*-й дисциплине.

Итоговая оценка по *i*-й дисциплине, по которой не предусмотрен экзамен, рассчитывается по формуле (1.5). Итоговая оценка по основному компоненту, по которому предусмотрен экзамен или экзамен и защита курсовой работы (проекта), рассчитывается по формулам (1.3-1.8) [17].

В своей работе Овчаренков Э.А. также рассматривает модульно-рейтинговую систему учета [18]. Он предлагает выделить каждой дисциплине 100 баллов, которые будут делиться между модулями, на которые она разбивается, с учетом их весомости. Например, студент может получить максимум 40 баллов за работу на практических занятиях и 60 на экзамене. Автор также предлагает выставлять студенту определенную оценку, не проводя сложных вычислений, если он преодолел определенный порог по баллам.

## Обзор программ аналогов

### Электронный журнал успеваемости «Магеллан»

Электронный журнал успеваемости позволяет хранить и редактировать данные по текущей аттестации и посещаемости обучающихся, формировать на их основе фактически выполненную аудиторную нагрузку преподавателей и рейтинги обучающихся, выявлять прогульщиков [20].

С помощью модуля можно создавать и выбирать типы занятий (лекции, практические, лабораторные, зачеты, экзамены), отмечать успеваемость обучающихся в 5-балльной или 100-балльной системе, вычислять средний балл на основе выставленных оценок.

Внешний вид данного приложения представлен на рис. 1.2.

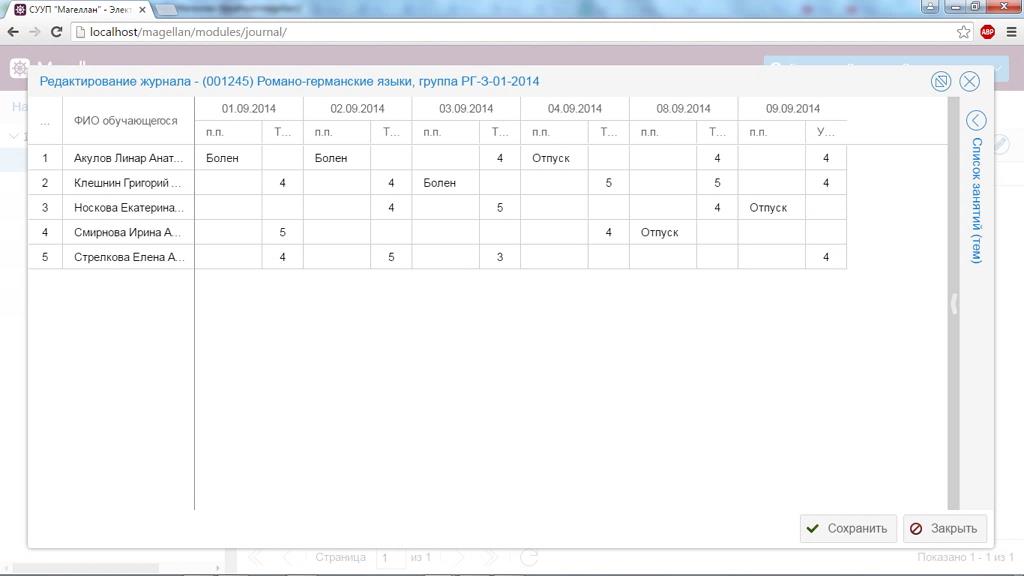


Рис 1.2. Окно редактирования журнала в веб-версии модуля

### Система учета абитуриентов и студентов. Компания Тауруна

Данная система разработана для улучшения качества и ускорения работы приемных комиссий и деканата.

Состоит из совместимых компонентов «Абитуриент» и «Студент». Позволяет вести личные карточки, электронный архив документов, мониторинг успеваемости. Упрощает отчетность, уменьшает «человеческий фактор», избавляет от дублирующей работы. Имеет простой интерфейс, поддерживает одновременные сеансы работы пользователей [21].

Система «Студент» – это набор инструментов для ведения базы данных учащихся и учета их динамики, мониторинга успеваемости в семестре и результатов экзаменационной сессии. Первичные данные могут использоваться для создания комбинированных отчетов по сложным условиям, в том числе – для подготовки сводных ведомостей по итогам года и вкладышей к дипломам.

Данные по текущей аттестации в систему могут вносить сами преподаватели. Также в системе «Студент» деканат может создавать и хранить документы.

Оригинальная разработка компании «Тауруна» с 2009 года в эксплуатации Ульяновского фармацевтического колледжа.

### Программный комплекс «ЭЛЕКТРОННЫЕ ВЕДОМОСТИ»

Программный комплекс разработан лабораторией математического моделирования и информационных систем (ММИС).

Информационная система «Электронные ведомости» предназначена для учета и анализа успеваемости студентов. Она позволяет проводить контроль как в течение семестра, так и по итогам сессии.

Подсистема «Электронные ведомости» позволяет использовать как 100-балльную рейтинговую систему, так и традиционную 5-балльную. Возможно использование до 6 контрольных точек (коллоквиумов) с несколькими видами работ [22].

Можно также использовать упрощенный вариант ведомости, если контрольные точки не проводятся. После экзамена/зачета предусмотрены поля для ввода трех пересдач.

В конце семестра электронная ведомость выводится на печать для подписи преподавателем. Учет успеваемости в единой базе данных позволяет производить оперативный текущий анализ и быстро подводить итоги сессии.

Для анализа успеваемости предусмотрена сводная ведомость по группе, в которой отображается рейтинг или итоговая оценка по каждой дисциплине и рассчитывается статус студента.

## Формализация процесса учета результатов аттестации студентов

Для описания процесса учета посещаемости занятий и успеваемости студентов, применим стандарт IDEF0. Данный стандарт используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающие эти функции.

Основной концептуальный принцип методологии IDEF0 заключается в представлении любой изучаемой системы в виде набора взаимодействующих и взаимосвязанных блоков, отображающих процессы, операции, действия, происходящие в изучаемой системе.

В IDEF0 все, что происходит в системе и ее элементах, принято называть функциями. Каждой функции ставится в соответствие блок.

В IDEF0 – диаграмме блок представляет собой прямоугольник. Интерфейсы, посредством которых блок взаимодействует с другими блоками или с внешней по отношению к моделируемой системе средой, представляются стрелками, входящими в блок или выходящими из него. Входящие стрелки показывают, какие условия должны быть одновременно выполнены, чтобы функция, описываемая блоком, осуществилась [19].

Функциональная модель рассматривается, как совокупность работ, каждая из которых оперирует некоторым набором данных. Работа изображается в виде прямоугольного блока, данные – в виде стрелок.

Процесс моделирования начинается с описания системы в целом. Функциональная модель по стандарту IDEF0 представляет собой совокупность иерархически упорядоченных и взаимосвязанных диаграмм.

Контекстная диаграмма функциональной модели мониторинга посещаемости занятий и успеваемости студентов, на которой определены все данные и объекты, участвующие в ходе рассматриваемого процесса представлена на рисРис. 1.3.

Выполнив декомпозицию процесса мониторинга посещаемости занятий и успеваемости студентов, получим диаграмму первого уровня функциональной модели процесса мониторинга посещаемости занятий и успеваемости студентов (рисРис. 1.4).

Процесс мониторинга посещаемости занятий и успеваемости студентов можно разделить на ряд этапов:

* мониторинг текущей посещаемости;
* учет результатов текущей аттестации;
* учет результатов внутрисеместровой аттестации;
* учет результатов промежуточной аттестации;
* учет результатов итоговой аттестации.

Рассмотрим диаграмму второго уровня процесса мониторинга посещаемости занятий и успеваемости студентов представленную на рис 1.5. На данной диаграмме произведена декомпозиция блока «Учет результатов текущей аттестации». Этот блок можно разбить на следующие элементы:

* формирование ведомостей;
* заполнение ведомостей;
* обработка ведомостей.

|  |  |
| --- | --- |
| D:\Cybran\YandexDisk\Практика\Фун1.png | Рис. 1.3. Функциональная модель мониторинга посещаемости занятий и успеваемости студентов |
| D:\Cybran\YandexDisk\Практика\Фун2.png | Рис. 1.4. Декомпозиция процесса мониторинга посещаемости занятий и успеваемости студентов |
| D:\Cybran\YandexDisk\Практика\Фун3.png | Рис. 1.5. Декомпозиция процесса учета результатов текущей аттестации студентов |

## Системы учета результатов аттестации студентов на кафедре

В мае 2014 года Борисовым З.В. под руководством Булатицкого Д.И. было разработано мобильное приложение и простой программный интерфейс (API). Приложение в рамках тестовой эксплуатации на 2-х группах первого курса кафедры «ИиПО» 14ИВТ1 и 14ИВТ2 общей численностью 36 студентов позволяло проставлять посещаемость и успеваемость. Большим минусом было то, что данные справочников, рабочих учебных планов вносились напрямую в базу данных через панель управления. В начале 2015 года разработка была продолжена и было решено создать веб-приложение, которое позволяло бы работать, как с посещаемостью и успеваемостью, так и с заполнением справочников и рабочих учебных планов. В сентябре 2015 года программный комплекс был размещен на кафедральном веб-сервере и предоставлял возможность мониторинга текущей посещаемости 19 групп кафедры «ИиПО» общей численностью 384 студента. К этому времени ввод данных для добавления рабочих учебных планов уже осуществлялся через веб-интерфейс, но трудоемкость для администратора по их добавлению большая. В процессе апробации программного комплекса, происходила его доработка, в частности была добавлена возможность работы с промежуточной аттестацией. Для помощи в заполнении посещаемости и добавления рабочих учебных планов была введена роль «Куратор», которая позволила существенно снизить нагрузку на  администратора по заполнению рабочих учебных планов. Главным недостатком оставалось то, что программный комплекс не умел хранить семестровую историю, и приходилось бы каждый год перезаполнять базу новыми данными рабочих учебных планов, на основании которых формируются ведомости посещаемости и успеваемости. В начале 2016 года в программный комплекс «СУП» был внесен ряд серьезных изменений и доработок, в частности было принято решение обеспечить семестровое хранение данных посещаемости и  успеваемости студентов, что потребовало добавления новых сущностей в базу данных. Так же была добавлена генерация рабочих учебных планов на основании учебных планов направлений, что позволило автоматизировать заполнение рабочих учебных планов и как следствие снижение с трудоемкости администратора и куратора. Помимо всего, была добавлена возможность анализа и визуализации данных с применением OLAP-технологий, которая позволяет выявлять проблемных студентов на основании собранных данных о  посещаемости и успеваемости студентов.

Также в 2015 году аспирантом Калевко В.В. под руководством Лагерева Д.Г. было разработано мобильное приложение для контроля успеваемости на кафедре и синхронизации данных с «АРМ Преподавателя». Данное приложение позволяет работать с электронными карточками групп, вести учет необходимых для преподавателя данных на карточке групп (напр., тема курсовой работы), работать с комплексом через мобильное приложение или веб-сайт.

## Смежные информационные сервисы на кафедре

Помимо системы мониторинга посещаемости и успеваемости студентов на кафедре разрабатываются или уже разработаны и введены в эксплуатацию многие сервисы. Основные из них – следующие:

1. Автоматизированное рабочее место (АРМ) преподавателя – сервис мониторинга посещаемости студентов и сдачи лабораторных работ. В отличие от сервиса мониторинга посещаемости и успеваемости данный сервис не позволяет просмотреть статистику по количеству пропусков в месяц/семестр, а также больше ориентирован на работу через мобильное приложение.

2. Сервис учета контингента. Предоставляет данные о группах и студентах на кафедре, а также возможность редактировать их. Разработан для централизации данных о группах и предоставлении их всем смежным сервисам через имеющееся API.

3. Сервис распределения студентов по дипломным и научным руководителям. Как правило, процесс распределения происходит в основном без участия студентов, из-за чего порой некоторые студенты остаются недовольны конечным результатом. Данный сервис позволяет увидеть какие студенты назначены какому преподавателю и к каким преподавателям еще можно записаться.

4. Сервис распределения тем дипломных и курсовых работ между студентами. Данный сервис предоставляет данные о том, какие темы выбраны студентами, что позволяет избежать коллизий в темах.

5. Сервис учета оборудования. На кафедре имеется множество технического оборудования, учет которого упрощает данный сервис.

6. Сервис учета программного обеспечения упрощает учет программного обеспечения, установленного на вычислительных машинах кафедры.

7. Сервис хранения курсовых и выпускных квалификационных работ. Данный сервис позволяет оперативно узнать какие темы, по котором уже когда-то были сданы работы, что позволяет избежать повторения тем и избавляет от необходимости поиска в бумажном архиве.

8. Сервис мониторинга прохождения практики студентами. Позволяет увидеть какие студенты сдали практику, на какую оценку, на каком предприятии они ее проходили, и какой отзыв был получен.

Разрабатываемый в рамках данной работы сервис мониторинга посещаемости и успеваемости студентов из перечисленных сервисов наиболее нуждается в интеграции с АРМ преподавателя и сервисом учета контингента.

В связи с тем, что АРМ преподавателя и системы мониторинга посещаемости и успеваемости студентов несколько похожи по своему функционалу было принято решение о разработке программного интерфейса для их интеграции.

Также для упрощения процесса актуализации списков групп и студентов целесообразно разработать модуль для взаимодействия с сервисом учета контингента через его API.

## Выводы по главе

На основе проведенного анализа предметной области можно сделать следующие выводы:

1. Автоматизация процесса учета результатов текущей и промежуточной аттестации является актуальной задачей. Так как системы для данного учета представленные на кафедре не имеют необходимого функционала, существует необходимость в их доработке.
2. На основе анализа рассмотренной литературы было установлено, что при развитии существующей системы целесообразно составить легко расширяемый справочник видов работ для аттестации, а также использовать балльно-рейтинговую систему для автоматического выставления оценок внутрисеместровой и промежуточной аттестации.
3. Так как разрабатываемая подсистема будет частью уже введенного в эксплуатацию сервиса, то использование в данном ключе рассмотренных аналогов будет крайне затруднительно. Однако их анализ поспособствовал формированию основных принципов работы и требований к будущей подсистеме.
4. Основными требованиями к системе учета результатов аттестации являются:

* удобный интерфейс для ввода данных об аттестации;
* разграничение доступа к системе;
* обработка больших объемом данных;
* автоматическое выставление оценок студентам по внутрисеместровой и промежуточной аттестации на основе данных о текущей аттестации и посещаемости;
* обеспечение удобных инструментов для создания и редактирования различных работ и контроля успеваемости.

**Целью данной работы** является расширение инструментария для учета успеваемости студентов в системе учета посещаемости и успеваемости и разработка программного интерфейса для интеграции данной системы со  сторонними сервисами.

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

* анализ процесса учета и видов аттестации студентов, а также способов взаимодействия различных сервисов друг с другом;
* формирование функциональных требований к веб-интерфейсу системы учета результатов аттестации и к программному интерфейсу для интеграции системы учета результатов аттестации с другими сервисами;
* разработка подсистемы учета результатов аттестации студентов;
* разработка программного интерфейса для интеграции системы мониторинга посещаемости и успеваемости студентов с другими сервисами;
* опытная эксплуатация системы и апробация разработанных моделей и методов на реальных данных.

**Объектом исследования** выступают процессы учета результатов текущей и промежуточной аттестации студентов, а также взаимодействие программных систем посредством программных интерфейсов.

**Предметом исследования** является автоматизация процесса учета результатов аттестации студентов и разработка программного интерфейса для интеграции со смежными сервисами.

# РАЗРАБОТКА АЛГОРИТМОВ ОЦЕНИВАНИЯ РАБОТЫ СТУДЕНТОВ В СЕМЕСТРЕ И ПУТЕЙ ИНТЕГРАЦИИ СИСТЕМЫ СО СМЕЖНЫМИ СЕРВИСАМИ

Как было показано в пункте 1.6 система учета посещаемости и успеваемости нуждается в интеграции с АРМ преподавателя и сервисом учета контингента. Интеграция с АРМ будет односторонней: предполагается, что АРМ преподавателя будет передавать данные в СУП. Для этого следует разработать программный интерфейс. Для интеграции с сервисом учета контингента следует изучить его возможности и предоставляемое им API.

Для упрощения процесса проведения внутрисеместровой и промежуточной аттестации задачу первичной оценки успеваемости студентов можно возложить на разрабатываемую систему. Но для этого необходимо разработать алгоритмы оценивания работы студентов в семестре.

Таким образом, **целью исследования** является разработка алгоритмов оценивания работы студентов в семестре и путей интеграции системы со смежными сервисами.

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

* проанализировать протоколы, технологии и стили взаимодействия распределенного приложения и выбрать более подходящий из них;
* рассмотреть документацию сервиса учета контингента, выявить среди предлагаемых сервисом возможностей полезные для СУП и спроектировать формат взаимодействия этих двух сервисов;
* разработать алгоритмы оценивания работы студентов в семестре.

## Анализ технологий взаимодействия партнерских сервисов или компонентов распределенного приложения

### Архитектурный стиль REST API

REST (Representational State Transfer) – архитектурный стиль взаимодействия компонентов распределённого приложения в сети.

REST представляет собой согласованный набор ограничений, учитываемых при проектировании распределённой системы. В определённых случаях (интернет-магазины, поисковые системы, прочие системы, основанные на данных) это приводит к повышению производительности и упрощению архитектуры. В широком смысле компоненты в REST взаимодействуют наподобие взаимодействия клиентов и серверов во Всемирной паутине [24].

Существует пять обязательных и одно факультативное ограничение для построения распределённых REST-приложений по Филдингу:

1) модель клиент-сервер;

2) отсутствие состояния;

3) кэширование;

4) единообразие интерфейса;

5) слои;

6) код по требованию (необязательное ограничение).

Если сервис-приложение нарушает любое из этих ограничительных условий, данную систему нельзя считать REST-системой.

Выполнение этих ограничительных требований обязательно для REST-систем. Накладываемые ограничения определяют работу сервера в том, как он может обрабатывать и отвечать на запросы клиентов. Действуя в рамках этих ограничений, система приобретает такие желательные свойства как производительность, масштабируемость, простота, способность к изменениям, переносимость, отслеживаемость и надежность.

### Архитектурный стиль JSON API

JSON API является спецификацией того, как клиент должен отправлять запросы на добавление и изменение сущностей, и как сервер должен на эти запросы отвечать [28].

Среди основных отличий от REST API можно выделить:

1. Использование только одного метода для передачи данных – POST для HTTP и SEND, в случае использования Web Sockets.

2. Механизм передачи и содержимое запроса полностью независимы. Все ошибки, предупреждения и данные передаются в теле запроса, в формате JSON

3. Используется лишь один код ответа, чтобы подтвердить успешную передачу, обычно это 200 ОК.

4. Все данные находятся в одном месте в легко-читаемом формате JSON

5. Переносимость на любой канал связи, например, HTTP/S, WebSockets, XMPP, telnet, SFTP, SCP, or SSH.

JSON API появилось в следствии осознания разработчиками того факта, что RESTful API не особо дружелюбно к браузерам. Разделение сообщения и способа передачи делает JSON API быстрым, надежным, простым в использовании, портировании и поиске ошибок.

### Протокол XML-RPC

XML-RPC (сокр. от англ. Extensible Markup Language Remote Procedure Call – XML-вызов удалённых процедур) – стандарт/протокол вызова удалённых процедур, использующий XML для кодирования своих сообщений и HTTP в качестве транспортного механизма. Является прародителем SOAP, отличается исключительной простотой в применении. XML-RPC, как и любой другой интерфейс Remote Procedure Call (RPC), определяет набор стандартных типов данных и команд, которые программист может использовать для доступа к функциональности другой программы, находящейся на другом компьютере в сети [26].

### Протокол SOAP

SOAP (от англ. Simple Object Access Protocol – простой протокол доступа к объектам; вплоть до спецификации 1.2) – протокол обмена структурированными сообщениями в распределённой вычислительной среде. Первоначально SOAP предназначался в основном для реализации удалённого вызова процедур (RPC). Сейчас протокол используется для обмена произвольными сообщениями в формате XML, а не только для вызова процедур. Официальная спецификация последней версии 1.2 протокола никак не расшифровывает название SOAP. SOAP является расширением протокола XML-RPC.

SOAP может использоваться с любым протоколом прикладного уровня: SMTP, FTP, HTTP, HTTPS и др. Однако его взаимодействие с каждым из этих протоколов имеет свои особенности, которые должны быть определены отдельно. Чаще всего SOAP используется поверх HTTP.

SOAP является одним из стандартов, на которых базируются технологии веб-служб [25].

## Взаимодействие с сервисом учета контингента

### Программный интерфейс сервиса учета контингента

На кафедре проводится ряд работ по объединению всех кафедральных сервисов в единую систему на основе корпоративных сервисных шин – особого класса сетевого программного обеспечения.

Одной из первых, внедренных в данную систему, служб является сервис учета контингента [27]. Данный сервис позволяет получить данные о студентах, группах и преподавателях через специализированное API, разработанное согласно архитектурному стилю RESTfull API. Все данные передаются в формате JSON. В каждом ответе API, помимо основных полей запрашиваемого объекта, также содержатся метаданные описывающие возможные действия с данным объектом.

Также стоит отметить, что API сервиса учета контингента позволяет подписываться на оповещения об изменении интересующих получателя сущностей. При изменении какого-либо объекта сервис генерирует системное событие, которое отправляется оконечным точкам всех заинтересованных в данном объекте сервисов.

Методы API сервиса учета контингента находятся по адресу http://82.179.88.27:8280/core/v1 и разделены на два основных блока – для работы с группами (/groups) и для работы со студентами и преподавателями (/people).

Модуль работы с группами имеет следующие методы:

* + - 1. /groups – GET запрос. Ответ содержит список всех групп на кафедре.
      2. /groups – POST запрос на создание группы.
      3. /groups/{id} – GET запрос. Ответ содержит данные о составе группы и названии.
      4. /groups/{id} – PATCH запрос на изменение данных о группе.
      5. /groups/{groupId}/students/{studentUid} – PUT запрос на добавление студента в группу.
      6. /groups/{groupId}/students/{studentUid} – DELETE запрос на исключение студента из группы.

Модуль работы со студентами имеет следующие методы:

* + - 1. /people – GET запрос. Ответ содержит список всех студентов и преподавателей на кафедре.
      2. /people – POST запрос на создание студента или преподавателя.
      3. /people{?uid,givenName,sn,initials,mail,title,cn} – GET запрос. Осуществляет поиск студента или преподавателя по переданным параметрам.
      4. /people/{uid}/profile – GET запрос. Ответ содержит полные данные о студенте или преподавателе.

### Проектирование формата взаимодействия с сервисом учета контингента

Предполагается, что начальная синхронизация уже существующих групп в сервисе мониторинга посещаемости и успеваемости с группами в сервисе учета контингента будет осуществлена с помощью однократного запуска специального метода, который сопоставит группы по названию и запишет идентификаторы групп в сервисе учета контингента в базу. Данный метод должен будет запускать программист, и наблюдать за ходом выполнения процесса синхронизации. Если какая-то из групп не будет найдена, программист должен будет проверить, в чем причина ошибки, и если выяснится, что она заключается в небольших различиях названия группы в двух сервисах, вручную исправить название и запустить метод заново.

В дальнейшем при вызове окна создания новой группы, будет произведен запрос к сервису учета контингента на получение списка групп и пользователю вместо ручного ввода названия будет достаточно выбрать нужную группу из выпадающего списка.

Также будет доработан веб-интерфейс редактирования группы и списков групп. При изменении данных о составе группы или о конкретном студенте сервис учета контингента будет отправлять оповещение, содержащее все данные об измененной сущности, на оконечную точку сервиса учета успеваемости и посещаемости. Таким образом, данные о списках групп будут автоматически поддерживаться в актуальном состоянии, следовательно, с интерфейса редактирования списка студентов группы необходимо будет удалить элементы для добавления, удаления и редактирования студента.

На основе описанных выше положений было принято решение о разработке трех основных модулей для организации взаимодействия сервиса мониторинга посещаемости и успеваемости с сервисом учета контингента:

* модуль получения данных о списке и составе групп по  непосредственному запросу с веб-интерфейса;
* модуль одноразовой сопоставления данных уже имеющихся в сервисе учета посещаемости и успеваемости с данными в сервисе учета контингента;
* модуль автоматической актуализации списков групп на основе оповещений, полученных от сервиса учета контингента.

## Разработка алгоритмов оценивания работы студентов в семестре

Чтобы облегчить и упростить процесс проведения внутрисеместровой и промежуточной аттестации, целесообразно разработать алгоритмы автоматического выставления оценок на основе данных о текущей аттестации и посещаемости.

Данный алгоритм будет работать следующим образом. При инициировании автоматической генерации данных на новый семестр администратором будут автоматически сгенерированы ведомости всех групп по всем предметам, которые выносятся на аттестацию, с пустыми полями. Преподаватель авторизуется в системе и переходит на страницу со списками электронных ведомостей внутрисеместровой аттестации всех групп, у которых он ведет какие-либо предметы. Преподаватель выбирает ведомость интересующей его группы по конкретному предмету и переходит на страницу редактирования этой ведомости. Далее он либо вносит данные вручную, либо инициирует автоматическое заполнение ведомости на основе оценок по текущей аттестации. Перед тем, как начать процесс автоматического проставления оценок, система проверяет, есть ли уже в ведомости какие-либо данные. Если они есть, преподавателю показывается окно с предупреждением о том, что все внесенные ранее данные будут утеряны, спрашивается, продолжить ли действие, и предлагается продолжить или отменить процесс. Если преподаватель отменил действие, то окно закрывается, и никаких данных в ведомость не вносится, в противном случае процесс продолжается и ведомость заполняется автоматически рассчитанными оценками. Далее преподаватель может изменить любую непонравившуюся ему автоматически выставленную оценку. Блок-схема данного алгоритма представлена на рис. 2.1.

Формула, по которой рассчитывается оценка по внутрисеместровой аттестации на основе данных о текущей аттестации и посещаемости имеет следующий вид:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула2.1) |

где *Ова* – оценка студента по предмету внутрисеместровой аттестации, *По* – общее количество занятий по предмету за период аттестации, *Пи* – количество посещенных занятий студентом, *n* – количество лабораторных работ, которые студенты должны сдать, за период аттестации, *Вi* – вес i-ой лабораторной работы (его определяет преподаватель), *Бi* – балл за выполнение i-ой работы, *Кiс* – коэффициент, который позволяет учитывать срок сдачи лабораторной работы, может принимать следующе значения: 1,5 – сдана досрочно, 1 – сдана вовремя, 0,5 – сдана с опозданием и 0 – не сдана.

Как видно из формулы значение *Ова* может превышать 5 баллов, это означает что данный студент проявил повышенную активность по данному предмету и его можно поощрить. В таком случае в поле ведомости все равно выставляется значение 5, но добавляется специальное обозначение.

Также схожий механизм должен быть заложен в интерфейс работы с ведомостями промежуточной аттестации. Оценка по промежуточной аттестации рассчитывается по следующей формуле:

|  |  |  |
| --- | --- | --- |
|  | , | (Формула2.2) |

где *Опа* – оценка студента по предмету промежуточной аттестации (то есть рекомендуемая оценка по экзамену), *Ова1, Ова2* – оценки студента по предмету по первой и второй внутрисеместровой аттестации (берутся из базы данных), *Ои* – оценка студента по предмету по итогам работы в семестре (рассчитывается по формуле 1 с учетом всех работ семестра, включая курсовой проект).



Рис. 2.1. Алгоритм автоматического расчета оценки по первой аттестации

## Выводы по главе

На основе проведенного исследования были сделаны следующие выводы:

1. Было принято решение использовать REST API при разработке программного интерфейса для взаимодействия системы мониторинга успеваемости и посещаемости студентов. Данный стиль упростит разработку и понимание принципов работы разрабатываемого API для последующих разработчиков, а также интеграцию в общую систему кафедры.

2. Разработан формат взаимодействия с программным интерфейсом сервиса учета контингента. Интеграция с данным сервисом позволит значительно упростить процесс актуализации списков групп в системе мониторинга посещаемости и успеваемости студентов, что снизит нагрузку на оператора сервиса.

3. Составлен алгоритм автоматического выставления оценок по внутрисеместровой и промежуточной аттестации на основе данных о текущей аттестации и посещаемости и разработана формула, по которой рассчитывается автоматическая оценка.

# РАЗРАБОТКА ПОДСИСТЕМЫ УЧЕТА РЕЗУЛЬТАТОВ АТТЕСТАЦИИ СТУДЕНТОВ

## Этапы разработки подсистемы

Процесс разработки подсистемы должен выполняться в соответствии с международным стандартом ISO/IEC 12207, регламентирующим общую структуру процессов жизненного цикла программного обеспечения [23]. В качестве модели жизненного цикла была выбрана каскадная модель, которая предполагает последовательное выполнение всех этапов проекта в строго фиксированном порядке. Переход на следующий этап производится после завершения всех работ на текущей стадии. В соответствии с выбранной моделью жизненного цикла были выделены следующие этапы разработки программного обеспечения:

1. *Формирование требований*. На данном этапе составляется список функциональных требований к системе. Происходит сбор, анализ и формализация требований к разрабатываемому программному обеспечению.
2. *Проектирование*. На данном этапе выполняется разработка общей архитектуры системы, функциональной схемы системы, отражающей основные функции подсистем и взаимосвязь их основных модулей. Проектирование осуществляется в соответствии с объектно-ориентированным подходом к проектированию программных систем. Основной задачей при такой методике является выделение объектов и определение их внутренней структуры и выполняемых ими функций.
3. *Выбор языка, средств разработки и СУБД*. На данной стадии производится выбор средств разработки программного продукта. Основными критериями являются: объектная ориентированность языка, типовая защищенность, поддержка совместимости версий, удобство интерфейса среды разработки, производительность и надежность СУБД, степень оснащенности системы инструментарием для администрирования, степень владения данными средствами разработки.
4. *Организация данных программной системы*. На данной стадии производится реализация спроектированной структуры данных программы. При ее реализации необходимо руководствоваться критериями скорости доступа и уровнем сложности взаимодействия с СУБД.
5. *Реализация программной системы*. На данном этапе происходит написание кода программной системы.
6. *Тестирование программной системы*. На данной стадии осуществляется отладка созданной системы, обеспечивающая выявление фактов расхождений с требованиями (ошибок).
7. *Внедрение*. Данный этап представляет собой процесс обучения персонала заказчика работе с программным обеспечением, предоставления помощи в настройке созданной системы.
8. *Эксплуатация и сопровождение*. На этом этапе основное внимание уделяется внесению изменений в программное обеспечение.

## Формирование функциональных требований к веб-интерфейсу подсистемы учета результатов аттестации студентов

Разрабатываемая подсистема должна взаимодействовать с уже существующей системой учета посещаемости, не нарушая ее работу.

Подсистема должна предоставлять доступ к просмотру хранимых данных всем заинтересованным лицам.

В разрабатываемой системе должны быть предусмотрены методы, с помощью которых будет осуществляться взаимодействие с мобильным приложением.

### Возможности роли «Гость»

Для пользователя роли «Гость» должны быть предоставлены следующие возможности:

* просмотр текущей аттестации любой группы за выбранный период;
* просмотр внутрисеместровой аттестации любой группы за выбранный период;
* просмотр промежуточной аттестации любой группы за выбранный период.

### Возможности роли «Преподаватель»

Для пользователя с ролью «Преподаватель» доступны все функции гостя, а также должны быть предоставлены следующие возможности:

* создание и редактирование курсов дисциплин для контроля успеваемости;
* заполнение ведомостей текущей аттестации по своим предметам;
* заполнение ведомостей внутрисеместровой аттестации по своим предметам;
* заполнение ведомостей промежуточной аттестации по своим предметам.

### Возможности роли «Куратор»

Для пользователя роли «Куратор» доступны все функции преподавателя, а также должны быть предоставлены следующие возможности:

* заполнение ведомостей внутрисеместровой аттестации курируемой группы;
* заполнение ведомостей промежуточной аттестации курируемой группы.

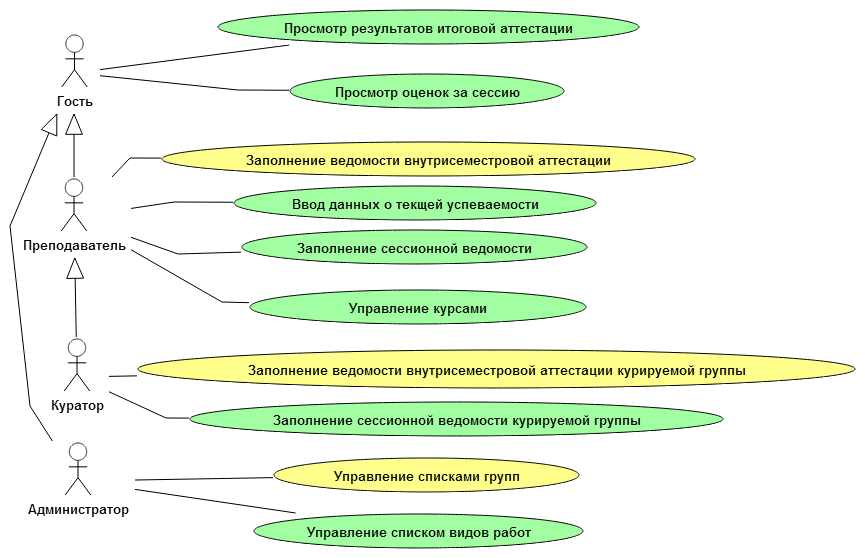
### Возможности роли «Администратор»

Для пользователя с ролью «Администратор» доступны все функции преподавателя, а также должны быть предоставлены следующие возможности:

* управление списками групп;
* управление списком видов работ.

### Функциональная модель подсистемы

Диаграмма вариантов использования, описываются функции, доступные каждому из ролей пользователей системы учета посещаемости представлена на рис. 3.1. Зеленым цветом выделены функции, разработанные с нуля, желтым – уже существовавшие, но претерпевшие ряд изменений.



**Рис. 3.1. Функциональная модель подсистемы учета результатов аттестации**

## Формирование функциональных требований к программному интерфейсу для интеграции с другими сервисами

Разрабатываемый программный интерфейс должен взаимодействовать с уже существующей системой учета посещаемости, не нарушая ее работу.

Данное API должно предоставлять методы для произведения следующих операций:

1. авторизация пользователя;
2. создание, редактирование, удаление видов работ;
3. создание, редактирование, удаление электронных ведомостей сдачи лабораторных работ;
4. создание, редактирование, удаление электронных ведомостей посещаемости;
5. получение данных о списках и составе групп в сервисе учета контингента;
6. обновление данных о списках и составе групп на основе оповещений от сервиса учета контингента.

Все запросы на создание, редактирование и удаление должны производиться только авторизованными пользователями. Доступ к методам, связанным с сервисом учета контингента, должен иметь только администратор или оператор.

## Проектирование системы

### Архитектура системы

Архитектурное проектирование – фаза «принятия решения». Цель этой фазы – определить совокупность компонент программного изделия и их интерфейсы, чтобы дать каркас для последующей разработки программного изделия. Архитектурный проект должен охватывать все требования, сформулированные на предыдущей фазе – системный анализ.

Архитектура системы, на которой показаны основные ее составляющие изображена на рис. 3.2. В архитектуру программного комплекса заложен шаблон MVC (Model, View, Controller) [29], где Model отвечает за обработку данных и логику, View отвечает за представление данных пользователю в любом поддерживаемом формате, а Controller отвечает за обработку запросов пользователя и вызов соответствующих ресурсов. При проектировании программного комплекса была использована трехуровневая архитектура, предполагающая наличие в нём трёх компонентов: клиентского приложения (обычно называемого «тонким клиентом» или терминалом), сервера приложений, к которому подключено клиентское приложение, и сервера базы данных, с которым работает сервер программного комплекса. На первом уровне располагается клиент, куда выносится простейшая бизнес-логика: интерфейс авторизации, алгоритмы шифрования, проверка вводимых значений на валидность, несложные операции (сортировка, группировка, подсчет значений) с уже загруженными данными. Взаимодействие с сервером берет на себя API.

Сервер программного комплекса располагается на втором уровне, где сосредоточена большая часть бизнес-логики. Вне его остаются фрагменты, экспортируемые на терминалы, а также погружённые в третий уровень хранимые процедуры и триггеры. Серверная часть предоставляет также программный интерфейс и отвечает за сохранение данных и предоставление их по запросам клиентских приложений. Сервер базы данных обеспечивает хранение данных и выносится на третий уровень.

Использование программного комплекса «СУП» происходит при помощи веб-браузера и мобильных приложений. При помощи сети Интернет, приложения получают необходимые данные с сервера программного комплекса для работы, а также происходит отправка данных. При помощи веб-интерфейса вносятся необходимые правки на сервер.

Программный комплекс обеспечивает интерфейс пользователя и включает в себя подсистемы:

* подсистема работы с учебными планами направлений;
* подсистема работы с рабочими учебными планами;
* подсистема учета результатов аттестации;
* подсистема работы с оповещениями;
* подсистема анализа и визуализации данных;
* программный интерфейс (API).

|  |  |
| --- | --- |
| D:\Cybran\YandexDisk\Диссертация\Arkhitektura_sistemy.png | Рис. 3.2. Архитектура разрабатываемой системы |

### Проектирование взаимодействия компонентов подсистемы учета результатов аттестации студентов

Подсистема учета результатов аттестации состоит из следующих компонентов:

* модуль учета текущей аттестации;
* модуль учета внутрисеместровой аттестации;
* модуль учета промежуточной аттестации;
* модуль управления курсами.

Диаграмма компонентов подсистемы учета результатов аттестации представлена на рис 3.3. Как видно из диаграммы, модуль учета текущей аттестации и модуль учета промежуточной аттестации зависят от модуля управления курсами. Это обусловлено тем, что электронные ведомости об успеваемости строятся на основе данных о курсах дисциплин, по которым обучается та или иная группа. Преподаватель сам указывает, какой группе назначить какой курс, и при заполнении курса различными работами (будь то лабораторные работы, контрольные работы, расчетно-графические работы и т.д.) также указывает, относится ли эта работа к текущей аттестации или промежуточной аттестации.

Также на диаграмму вынесена подсистема учета посещаемости, так как она участвует в алгоритме выставления оценок по внутрисеместровой и промежуточной аттестации. Для автоматического выставления оценки по внутрисеместровой аттестации подсистема учета результатов аттестации берет данные о сдаче работ студентов из модуля учета текущей аттестации и данные о пропусках пар из подсистемы учета успеваемости. Аналогично организовано автоматическое выставление оценок по промежуточной аттестации, но помимо данных о текущей успеваемости и посещаемости студента, также происходит запрос данных о внутрисеместровой аттестации.

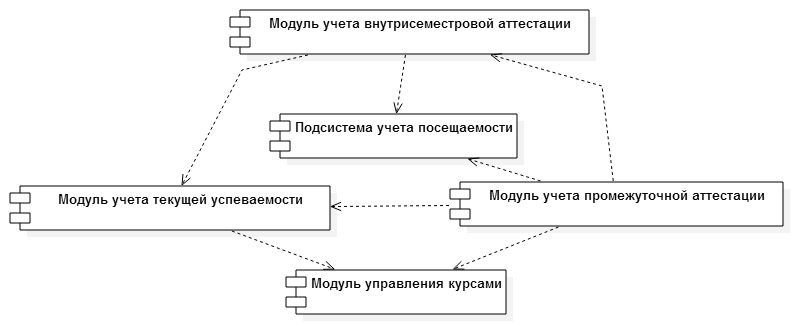


Рис. 3.3. Диаграмма компонентов подсистемы учета результатов аттестации

### Проектирование пользовательского интерфейса веб-приложения

#### Модуль работы с курсами дисциплин

Модуль предоставляет возможность редактирования курсов по дисциплинам, а также редактирования списка работ, связанных с курсом. Использовать данные возможности имеет право только пользователь в роли «Преподаватель».

Модуль состоит из нескольких веб-интерфейсов:

* список дисциплин;
* список курсов для отдельной дисциплины;
* список работ отдельного курса.

Пользователь видит функционал, который необходим для того, чтобы начать работу с курсами дисциплин:

* просмотр списка дисциплин, которые ведет преподаватель;
* просмотр списка курсов для выбранной дисциплины;
* добавление нового курса;
* редактирование курса;
* удаление курса;

Рассмотрим подробнее возможность добавления нового курса в список. На странице со списком дисциплин пользователь нажимает на кнопку «Редактировать курсы дисциплины» и переходит на страницу со списком курсов по выбранной дисциплине. Далее после нажатия на кнопку «Добавить курс» появляется всплывающее окно, в котором необходимо ввести информацию о новом курсе.

Макет страницы, на которой преподаватель может работать со списком дисциплин, представлен на рис. 3.4. Макет страницы, на которой преподаватель может работать со списком курсов дисциплины, представлен на рис. 3.5. Макет всплывающего окна добавления нового курса представлен на рис. 3.6.

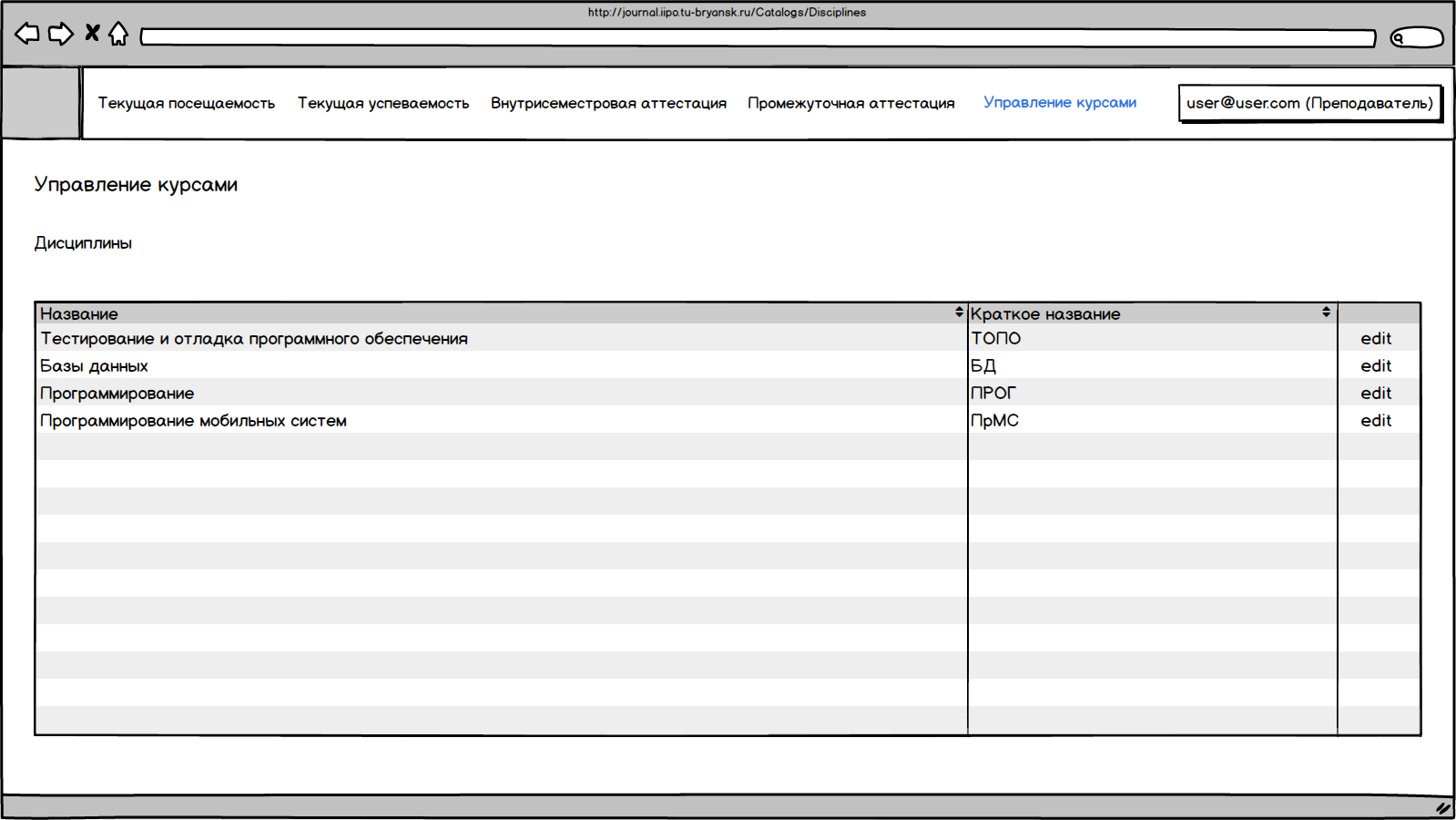


Рис. 3.4. Макет веб-страницы списка дисциплин

Веб-интерфейс список работ выбранного курса необходим для просмотра и редактирования информации внутри отдельного курса дисциплины. Преподавателю доступен следующий функционал:

* просмотр списка работ курса и информации о них;
* добавление новой работы;
* редактирование работы;
* удаление работы.

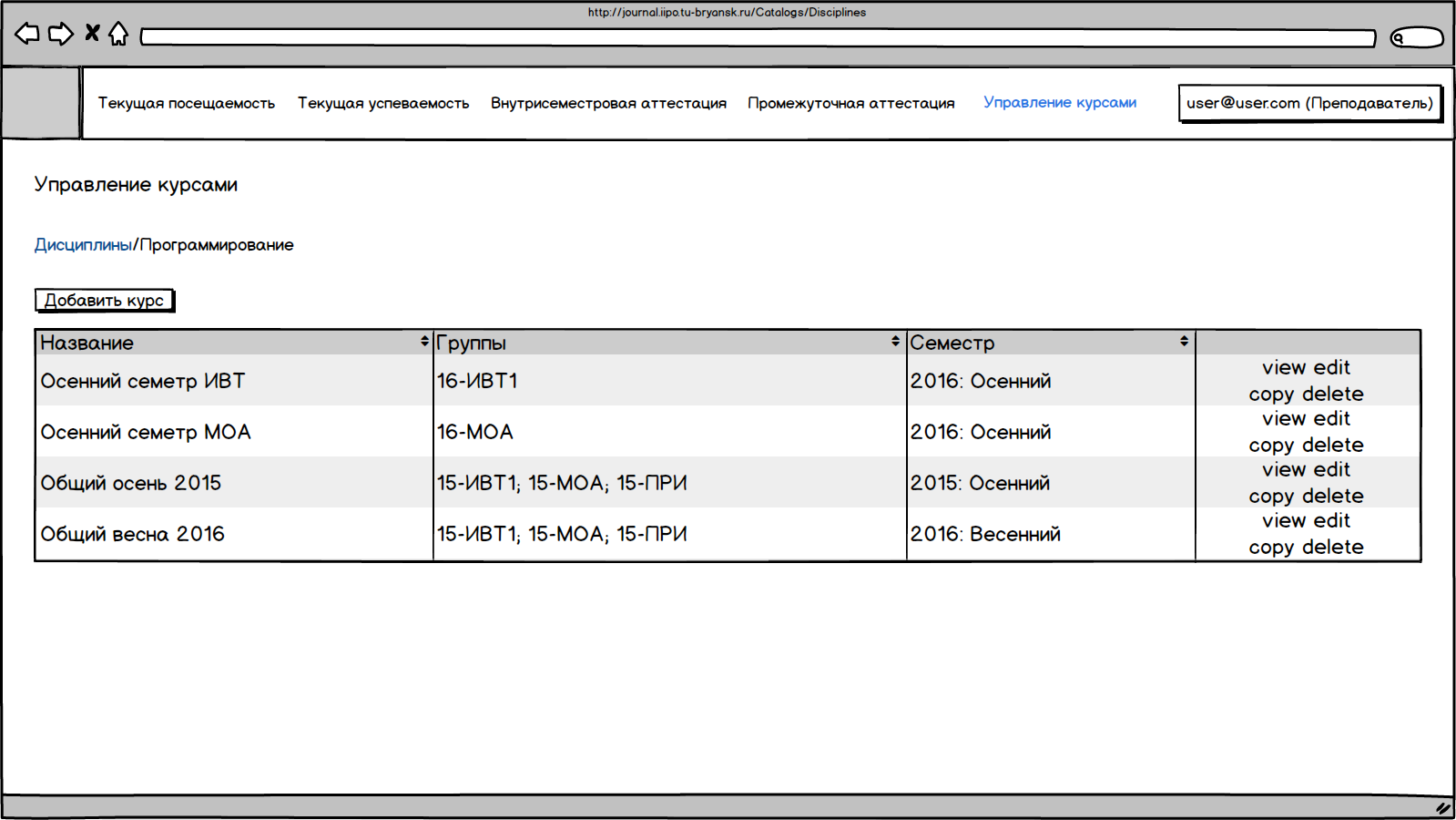


Рис. 3.5. Макет веб-страницы списка курсов дисциплины

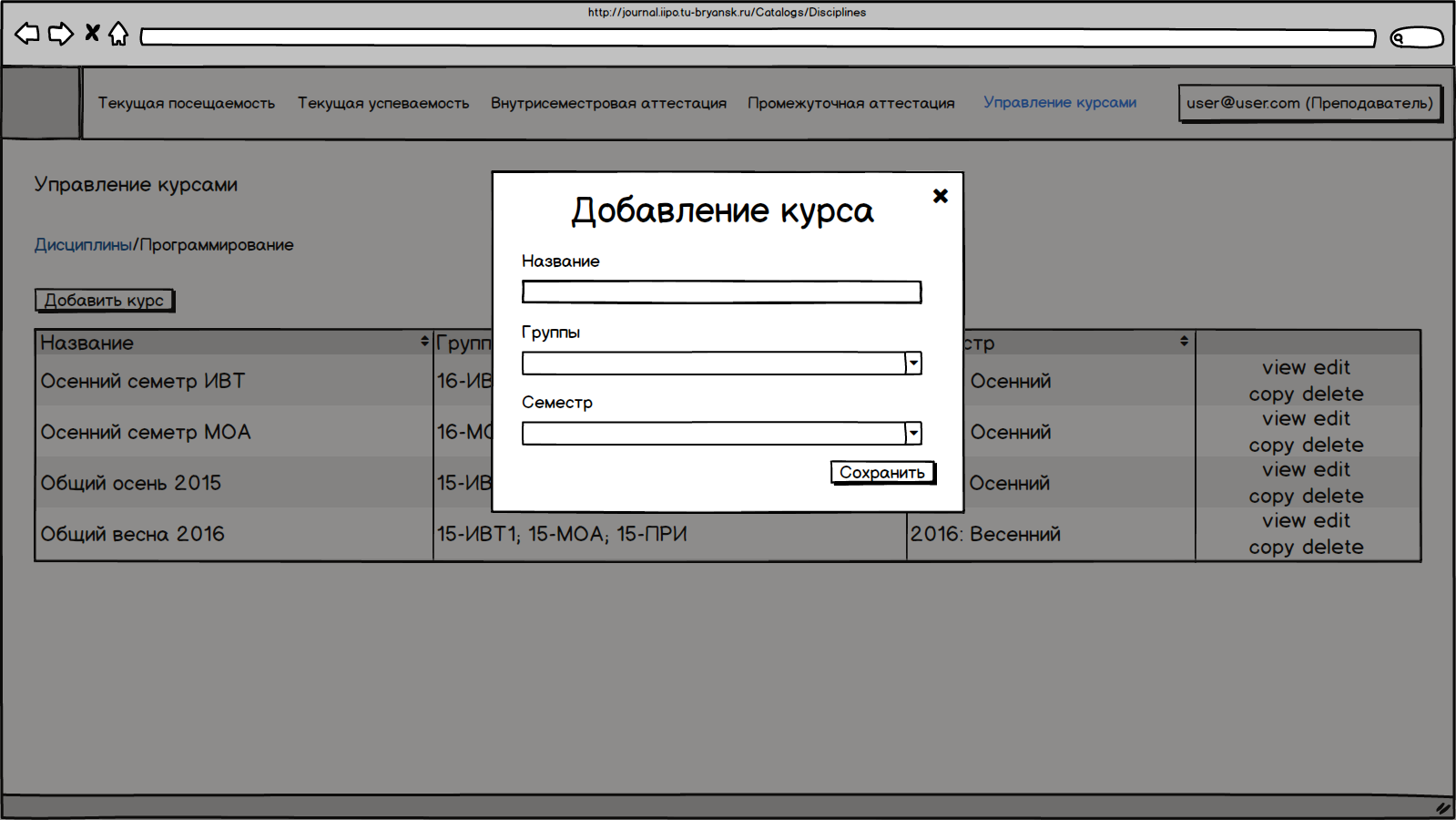


Рис. 3.6. Макет всплывающего окна «Добавление курса»

Макет веб-страницы, на которой преподаватель может работать с информацией внутри отдельного курса дисциплины, представлен на рис. 3.7. Макет всплывающего окна добавления работы представлен на рис. 3.8.

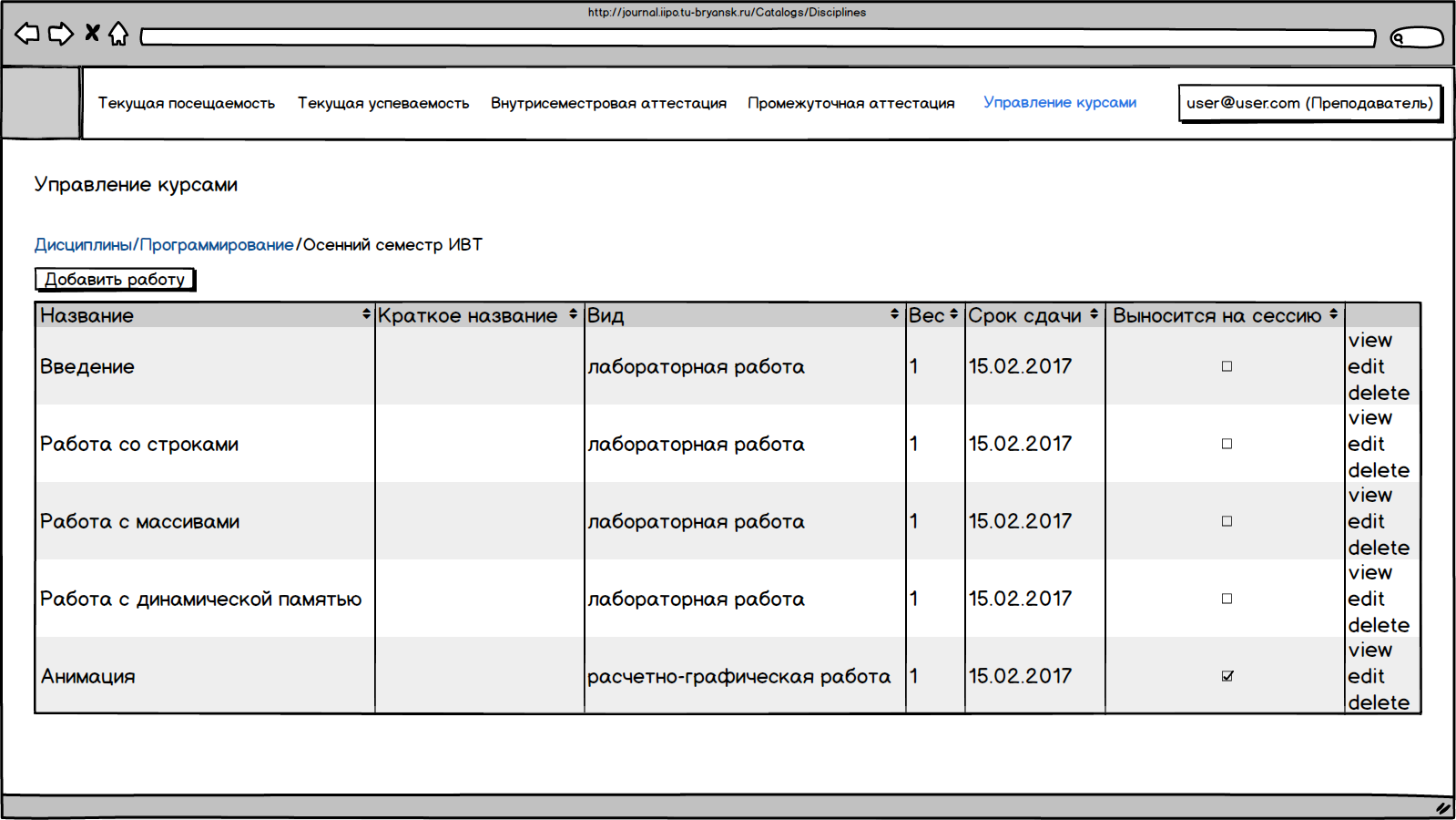


Рис. 3.7. Макет веб-страницы списка работ курса

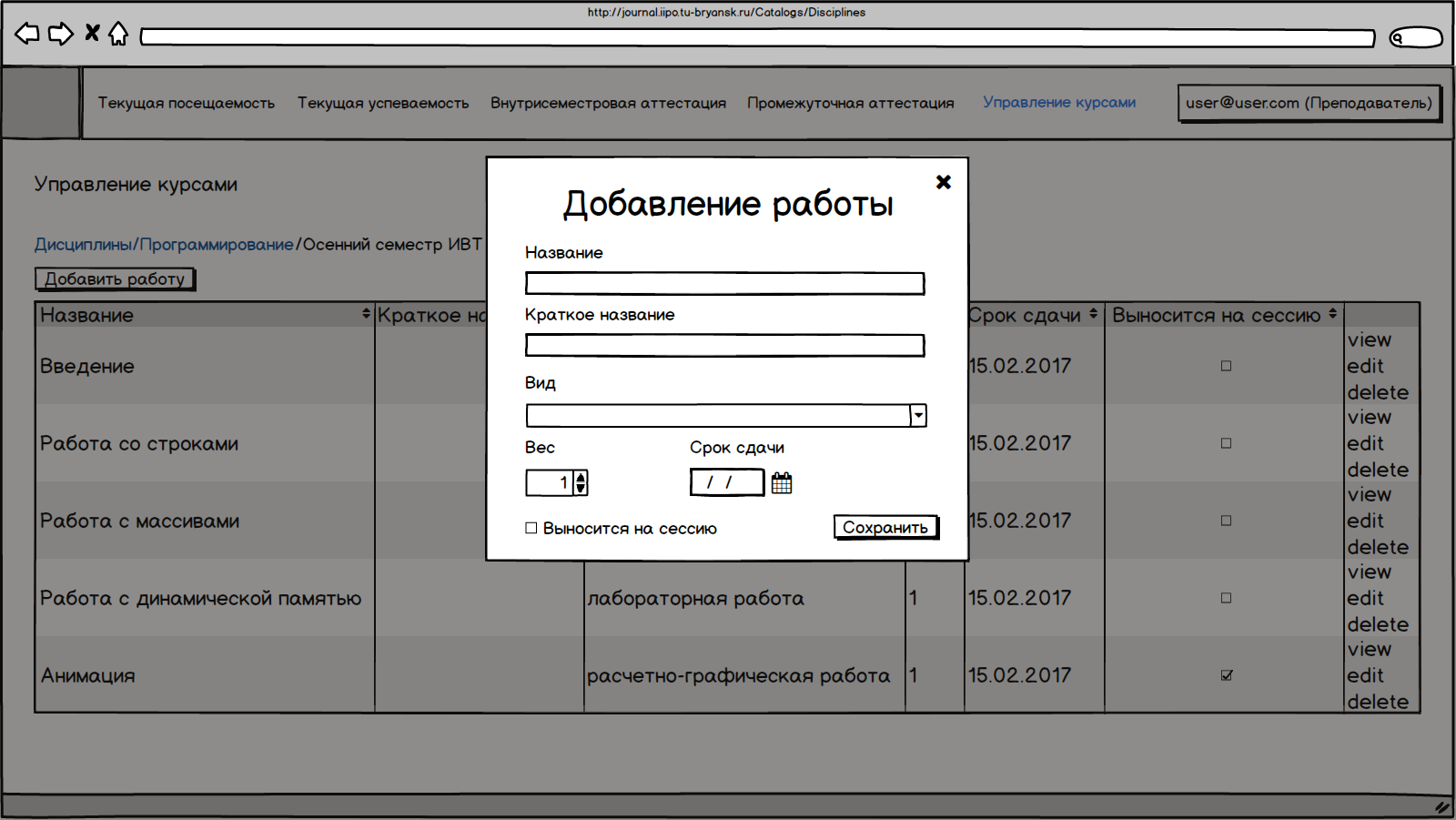


Рис. 3.8. Макет всплывающего окна «Добавление работы»

#### Модуль работы с текущей аттестацией

Модуль предоставляет любому пользователю возможность просматривать данные о текущей аттестации выбранной группы по конкретному предмету. Также пользователю в роли «Преподаватель» предоставляется возможность выставлять оценки по текущей аттестации.

Модуль состоит из нескольких веб-интерфейсов:

* список ведомостей;
* ведомость успеваемости группы по предмету;
* ведомость успеваемости группы с возможностью редактирования.

Рассмотрим подробнее возможность выставления оценок по текущей аттестации. На странице со списком ведомостей пользователь нажимает на кнопку с интересующей его ведомостью группы по некоторому предмету и переходит на страницу с ведомостью текущей аттестации выбранной группы по предмету. Далее после нажатия на ячейку таблицы появляется всплывающее окно, в котором необходимо указать оценку студента и дату сдачи работы.

Макет страницы, на которой преподаватель может работать со списком ведомостей, представлен на рис. 3.9. Макет страницы, на которой преподаватель может работать с ведомостью группы, представлен на рис. 3.10. Макет всплывающего окна выставления оценки студента по предмету представлен на рис. 3.11.

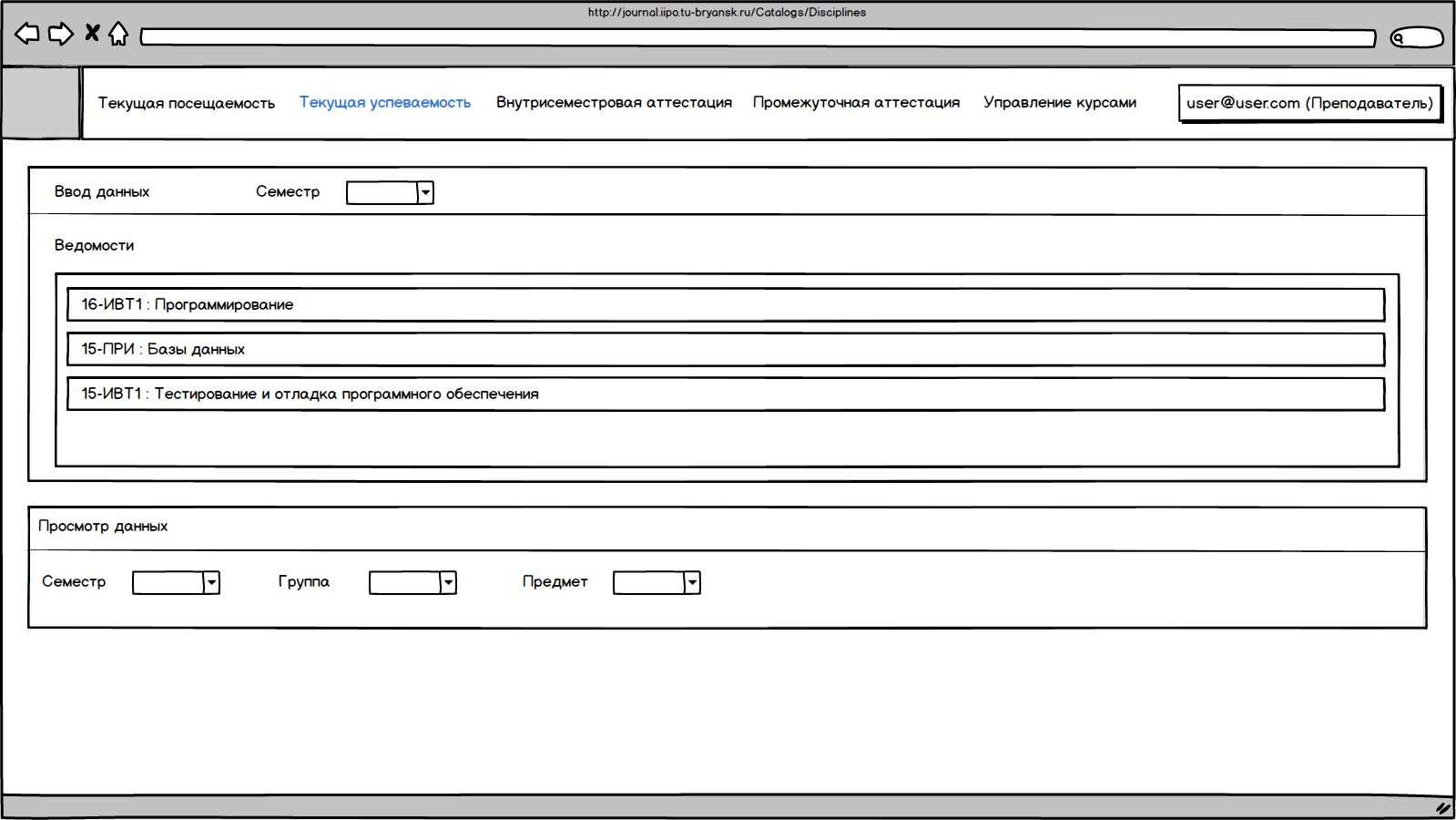


Рис. 3.9. Макет веб-страницы списка ведомостей

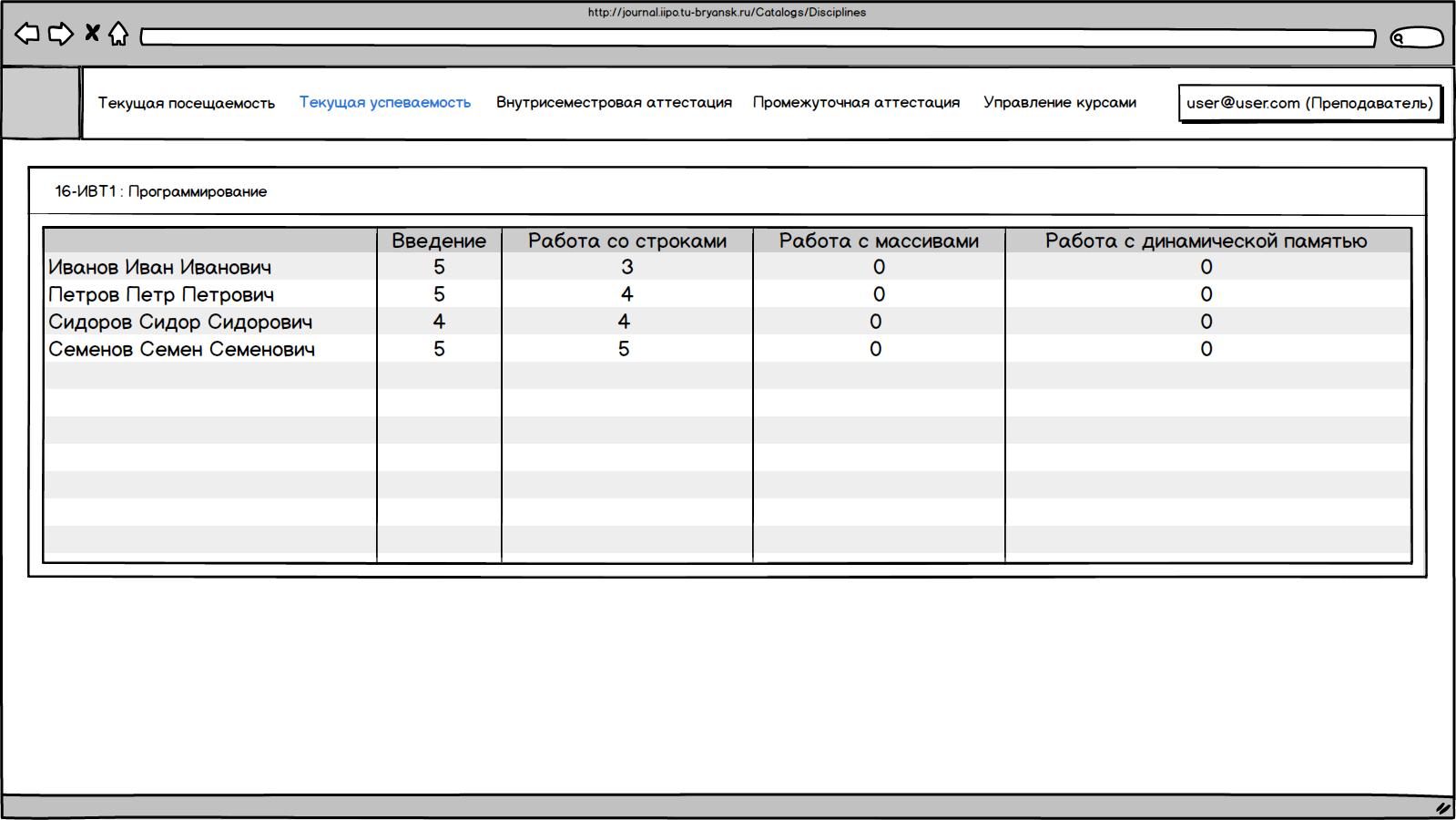


Рис. 3.10. Макет веб-страницы ведомости группы

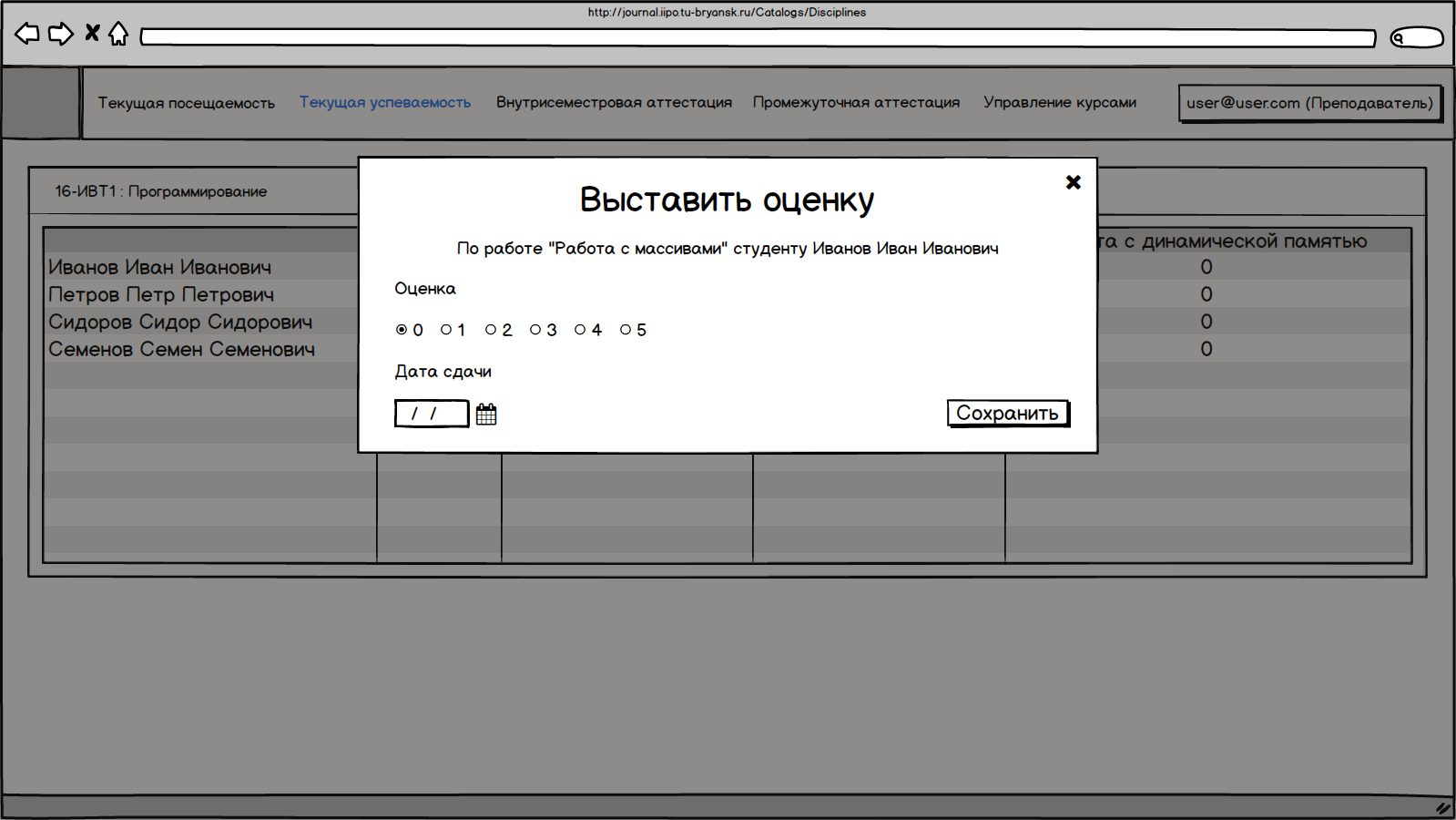


Рис. 3.11. Макет всплывающего окна выставления оценки

#### Модуль работы с промежуточной аттестацией

Модуль предоставляет любому пользователю возможность просматривать данные о промежуточной аттестации выбранной группы. Также пользователю в роли «Преподаватель» предоставляется возможность выставлять оценки по промежуточной аттестации.

Модуль состоит из нескольких веб-интерфейсов:

* список ведомостей;
* аттестационная ведомость группы;
* аттестационная ведомость группы с возможностью редактирования.

Рассмотрим подробнее возможность выставления оценок по промежуточной аттестации. На странице со списком ведомостей пользователь нажимает на кнопку с ведомостью интересующей его группы и переходит на страницу с ведомостью промежуточной аттестации выбранной группы. После нажатия на ячейку таблицы оценка в этой ячейке увеличивается на 1, а сама ячейка берется в фокус. Далее при нажатии на клавиши от 0 до 5, в выделенной ячейке выставляется соответствующая оценка, а фокус перемещается вниз на одну ячейку, или на первую ячейку следующего столбца, если в текущем столбце выбранная ранее ячейка была нижней. Также, преподаватель может запустить автоматическое выставление оценок по промежуточной аттестации на основе данных о текущей аттестации и посещаемости студентов.

Макет страницы, на которой преподаватель может работать со списком ведомостей, представлен на рис. 3.12. Макет страницы, на которой преподаватель может работать с ведомостью группы, представлен на рис. 3.13.

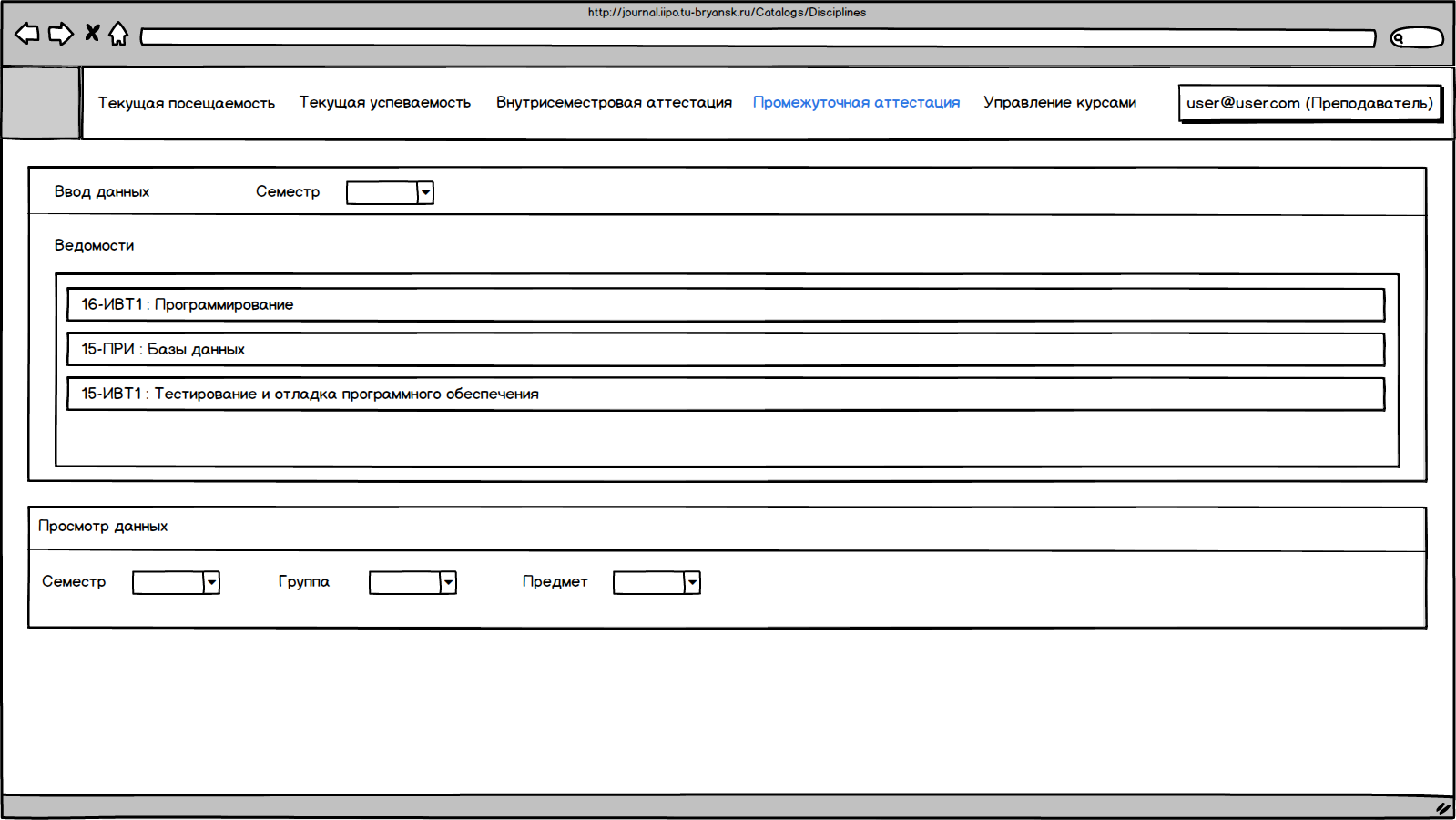


Рис. 3.12. Макет веб-страницы списка ведомостей

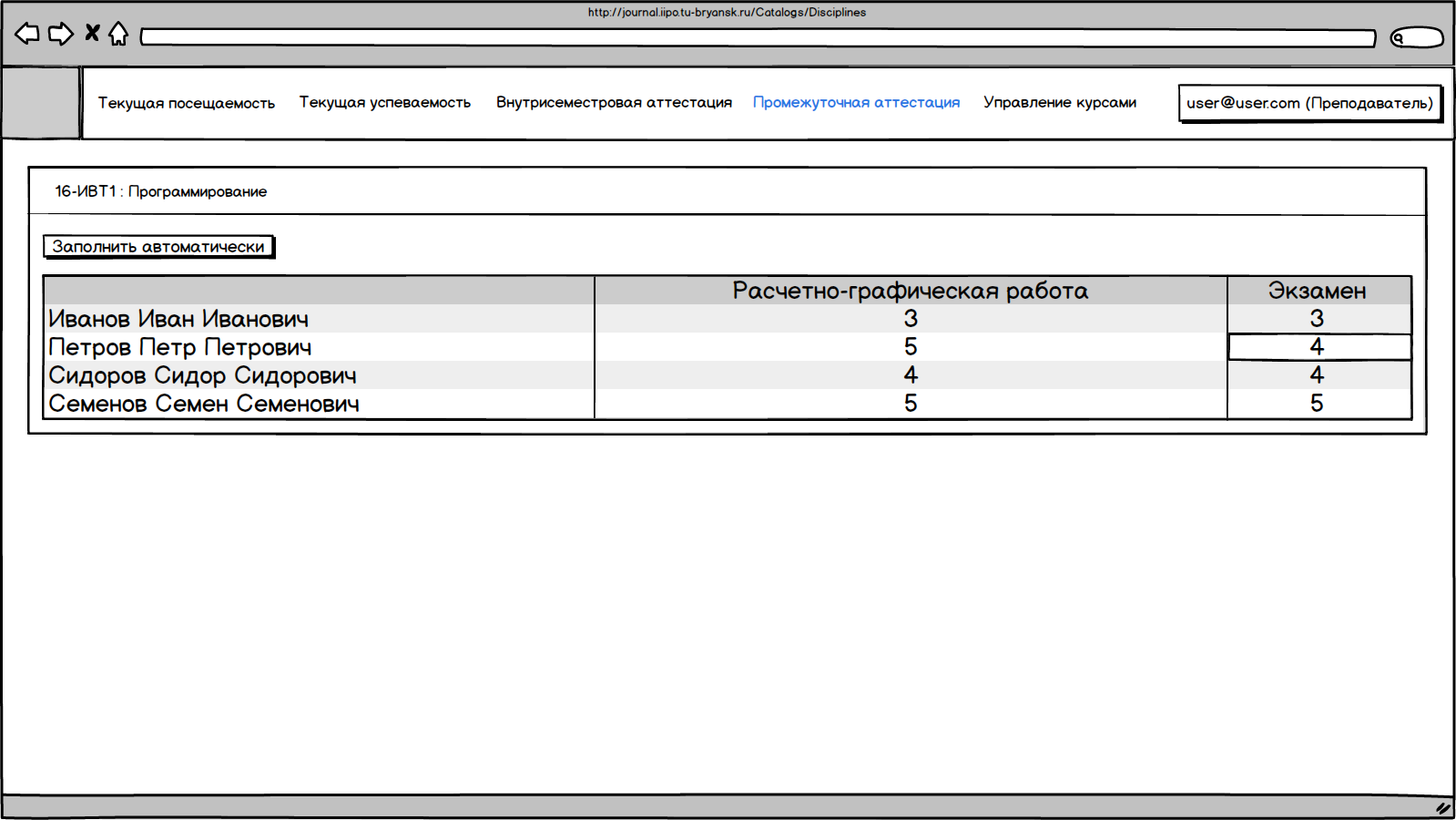


Рис. 3.13. Макет веб-страницы ведомости группы

### Разработка программного интерфейса для манипулирования данными в системе мониторинга посещаемости и успеваемости

Интерфейс будет разработан согласно архитектурному стилю REST API. Все общение с API будет происходить в зашифрованном виде по протоколу SSL. Это значит, что все ссылки к API должны будут содержать протокол HTTPS. Предусмотрены механизмы ограничения активности работы с API путем настройки в конфигурационном файле допустимой частоты запросов, а также в некоторых методах предусмотрены ограничения на количество возвращаемых за один запрос данных.

Для авторизации в системе через API, необходимо будет отправить запрос, содержащий API-ключ приложения, логин и пароль пользователя. Ответ будет содержать токен, с помощью которого будет происходить взаимодействие с остальными методами API.

Все данные в теле запроса к системе и ответа от нее будут представлены в формате JSON.

Далее представлен список некоторых методов разрабатываемого API c кратким описанием входных параметров и ответов.

Методы работы с API:

1. /attendance/{groupId} – GET метод. Позволяет получить данные о посещаемости. Входными параметрами являются id группы, номер месяца и id семестра. Ответ содержит данные о посещаемости каждого студента в группе в каждый день месяца.

3. /attendance/{studentId} – GET метод. Позволяет получить данные о посещаемости конкретного студента. Входными параметрами являются id студента, номер месяца и id семестра. Ответ содержит данные о посещаемости студента в каждый день месяца.

4. /attendance/{studentId} – POST метод. Позволяет изменить данные о посещаемости конкретного студента. Входными параметрами являются id студента, устанавливаемое значение, номер месяца, id предмета и id семестра. Ответ содержит данные об успешности выполнения операции или ошибку.

5. /attestation/{groupId} – GET метод. Позволяет получить данные об аттестации. Входными параметрами являются id группы, номер аттестации id семестра. Ответ содержит данные об аттестации каждого студента в группе.

6. /attendance/{studentId} – POST метод. Позволяет изменить данные об аттестации конкретного студента. Входными параметрами являются id студента, оценка, id предмета, номер аттестации и id семестра. Ответ содержит данные об успешности выполнения операции или ошибку.

7. /academicPlans – GET метод. Возвращает список учебных планов (УП) в системе. Входных параметров нет. Ответ содержит словарь ключ-значение, где в качестве ключа выступает id УП, а в качестве значения – название УП.

В случае возникновения ошибки для всех методов помимо стандартного кода ошибки, также возвращается описание ошибки в теле запроса.

### Разработка механизмов взаимодействия с сервисом учета контингента

Разработка механизмов взаимодействия с сервисом учета контингента включает в себя разработку методов для сопоставления уже имеющихся групп, получения списка групп из сервиса учета контингента и обновления состава всех групп или выбранной, а также внесение модификаций в веб-интерфейс списка групп, создания и редактирования группы.

В базе в таблицу Groups, будет добавлено новое поле ContingentId типа int. По этому полю систем будет связывать группу в сервисе мониторинга посещаемости и успеваемости с группой в сервисе учета контингента.

Для сопоставления уже имеющихся групп на интерфейсе списка групп будет добавлена кнопка «Сопоставить группы», видимая только пользователю с правами администратора. При нажатии на кнопку будет вызван метод SynchronizeGroups, который произведет GET запрос по адресу http://82.179.88.27:8280/core/v1/groups с целью получения списка всех групп из сервиса учета контингента. На основе ответа полученного в формате JSON система сгенерирует *Dictionary<int, string>*, содержащий связки *ContingentId*-название для всех групп. Далее система сделает выборку активных групп, не имеющих *ContingentId*, из базы и запишет в *List<Group>*. Программа в цикле сделает обход всех элементов получившегося *List<Group>* и попробует найти сопоставление в созданном ранее *Dictionary<int, string>* по названию. Если сопоставление найдется, то *ContingentId* из найденного элемента *Dictionary<int, string>* будет записано в соответствующее поле текущей группы в базе.

В окне создания новой группы при клике в поле названия будет произведен Ajax запрос к серверному методу *GetContingentGroupsList*. В данном методе система произведет GET запрос по адресу http://82.179.88.27:8280/core/v1/groups с целью получения списка всех групп из сервиса учета контингента. На основе ответа полученного в формате JSON система сгенерирует *Dictionary<int, string>*, содержащий связки *ContingentId*-название для всех групп. Далле из *Dictionary<int, string>* будут удалены все группы, которые уже имеются в системе. Полученный *Dictionary<int, string>* будет преобразован в JSON и отправлен на клиент в качестве ответа на Ajax запрос. На клиенте с помощью js полученный ответ будет преобразован в выпадающий список, который будет представлен пользователю для выбора возможного названия группы. При выборе какого-либо варианта из предложенного списка js запишет *ContingentId* выбранного элемента в соответствующий скрытый input формы. При получении данных из формы создания группы на сервере *ContingentId* будет записан в соответствующее поле созданной в базе группы.

## Выбор средств разработки

Выбор средств разработки определяется функциональными требованиями разрабатываемого программного комплекса, архитектурой, формой и видом данных. Также учитывается скорость и эффективность, с которыми могут разрабатываться отдельные компоненты и интерфейсы программного комплекса с помощью того или иного средства разработки.

В первую очередь следует установить, какую систему управления базами данных следует применить в программном комплексе с учетом требований и возможностей. В качестве СУБД выбрана MS SQL Server 2012 [30, 31] с учетом совместимости с программным обеспечением, установленным в Брянском государственном техническом университете.

Поскольку программный комплекс имеет клиент-серверную архитектуру, то в качестве веб-сервера был выбран (IIS) Internet Information Services [32], также с учетом совместимости с программным обеспечением, установленным в Брянском государственном техническом университете и в частности, в аудиториях кафедры «Информатика и программное обеспечение».

Следующим шагом является выбор языка программирования. Здесь главными критериями выбора являлись то, что бы язык программирования был довольно простой, современный, имел хорошую защищенность, а также хорошо взаимодействовал с выбранной СУБД. Поэтому был выбран язык программирования C# [33].

В качестве среды разработки выбрана Microsoft Visual Studio 2015 Community, поскольку это очень мощная и бесплатная среда разработки, которая предоставляет ряд преимуществ:

* мощные инструменты программирования;
* усовершенствованная отладка;
* веб-инструменты;
* развертывание и создание современных веб-приложений с помощью ASP.NET, Node.js, Python и JavaScript;
* возможность использования мощных веб-платформ, таких как AngularJS, jQuery, Bootstrap, Django и Backbone.js;
* поддержка нескольких языков, включая C#, Visual Basic, F#, C++, JavaScript, TypeScript, Python и множество других;
* помощь в отладке и тестировании кода независимо от того, какой язык выбран.

В качестве фреймворка выбрана ASP.NET MVC 4. Данный фреймворк используется для создания веб-приложений, которые реализуют шаблон Model-View-Controller.

## Модель данных

База данных для системы учета результатов аттестации студентов была спроектирована в ERwin Data Modeler, а затем импортирована в MS SQL 2012. Для управления базой данных используется приложение SQL Server Management Studio 2012 [31]. Логическая структура базы данных программного комплекса представлена на рис. 3.14. В процессе разработки программного комплекса база данных была приведена к 4 нормальной форме, определена предметная область со следующими основными сущностями:

* **Группа.** Справочник, содержащий информацию о группах, с которыми работает программный комплекс «СУП».
* **Студент.** Справочник, содержащий информацию, идентифицирующую студента.
* **Преподаватель.** Справочник, содержащий информацию, идентифицирующую преподавателя.
* **Предмет.** Справочник, содержащий идентифицирующую информацию о предмете.
* **Вид занятия.** Справочник, содержащий идентифицирующую информацию о виде занятия.
* **Семестр.** Справочник, содержащий информацию об идентификации учебного семестра.
* **Рабочий учебный план.** Отношение, содержащее информацию о рабочих учебных планах.
* **Текущая успеваемость**. Отношение, содержащее информацию о текущей успеваемости студентов.

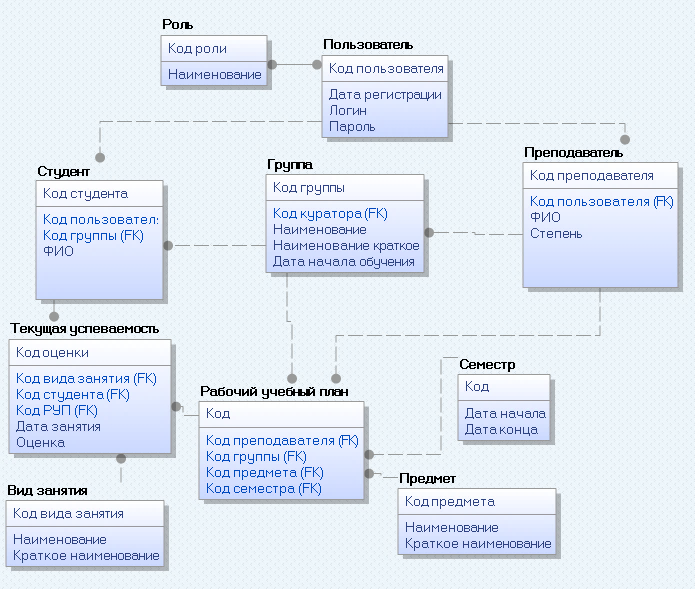


Рис. 3.14. Логическая структура сегмента базы данных

## Программная реализация подсистемы учета результатов аттестации

Для реализации модуля управления курсами были добавлены классы сущностей Сourse и Exercise, отражающие соответственно курсы дисциплин и работы. В уже существующий контроллер CatalogsController были добавлены методы для отображения, добавления, редактирования и удаления курсов и работ. Также были добавлены представления для отображения списка курсов, списка работ и всплывающих окон добавления и редактирования.

Для реализации модуля учета текущей аттестации был добавлен контроллер AcademicPerformanceController. В него были внесены методы для получения ведомости текущей аттестации конкретной группы по предмету, списка ведомостей преподавателя и ведомости текущей аттестации конкретной группы по предмету, которую может редактировать преподаватель. Также были добавлены представления для отображения аттестационной ведомости, списка ведомостей преподавателя и всплывающего окна выставления оценки студента по предмету.

Для реализации модуля учета промежуточной аттестации был добавлен контроллер SessionController. В него были внесены методы для получения ведомости промежуточной аттестации конкретной группы по предмету, списка ведомостей преподавателя и ведомости промежуточной аттестации конкретной группы по предмету, которую может редактировать преподаватель. Также были добавлены представления для отображения аттестационной ведомости и списка ведомостей преподавателя.

## Программная реализация интеграционного программного интерфейса

Для реализации программного интерфейса было создано отдельное ASP.NET Web API приложение. Вынесение API в отдельное приложение позволяет обновлять основное веб-приложение и приложение с API независимо друг от друга.

Для реализации модуля интеграции с сервисом учета контингента был добавлен контроллер ContingentController. В него был добавлен метод оконечной точки, который получает оповещения об изменении состава группы или данных студента и автоматически в фоновом режиме производит актуализацию данных. Также был добавлен метод для однократного сопоставления уже имеющихся в СУП данных о группах и студетах с данными в сервисе учета контингента.

Для реализации модуля интеграции с АРМ преподавателя были добавлены контроллеры AuthController, AttestatonDataController и AttendanceDataController.

Контроллер AuthController содержит методы, с помощью которых производится авторизация в API.

В контроллер AttestatonDataController были внесены методы для получения и обновления данных о текущей, внутрисеместровой и промежуточной аттестациями.

В контроллер AttendanceDataController были внесены методы для получения и обновления данных о посещаемости занятий студентами.

# ТЕСТИРОВАНИЕ И ВЕРИФИКАЦИЯ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## Проверка выполнения функциональных требований

### Проверка работы модуля учета результатов текущей аттестации

Необходимо произвести попытку создать ведомость текущей аттестации, отредактировать и удалить данные в ведомости какой-нибудь группы поочередно в роли администратора, гостя, старосты и преподавателя. Убедиться в том, что гость и администратор не имеют доступа к этим операциям и, что они корректно выполняются у старосты и преподавателя.

### Проверка работы модуля учета результатов внутрисеместровой аттестации

Необходимо произвести попытку создать ведомость внутрисеместровой аттестации, отредактировать и удалить данные в ведомости какой-нибудь группы поочередно в роли администратора, гостя, старосты и преподавателя. Убедиться в том, что гость, староста и администратор не имеют доступа к этим операциям и что они корректно выполняются у преподавателя.

### Проверка работоспособности программного интерфейса

Чтобы проверить работоспособности программного интерфейса необходимо будет с помощью АРМ преподавателя, либо какой-либо вспомогательной программы произвести вызовы всех методов API. Убедиться в том, что корректно работает механизм авторизации, а так же создание, редактирование и удаление видов работ, электронных ведомостей сдачи лабораторных работ и электронных ведомостей посещаемости.

### Проверка модуля взаимодействия с сервисом учета контингента

Для проверки данного модуля необходимо запустить процесс автоматической актуализации списков групп и студентов, а затем проверить сгенерированные данные с данными в ведомости.

## Опытная эксплуатация

### Работа с веб-интерфейсами подсистемы учета результатов аттестации

Подсистема, разрабатываемая в рамках магистерской работы, была введена в эксплуатацию в системе мониторинга посещаемости и успеваемости студентов на кафедре «Информатика и программное обеспечение».

Так как система была внедрена в конце семестра, а не вначале, то провести опытную эксплуатацию непосредственно в ходе учебного процесса не удалось. Однако проверка была произведена на реальных данных, полученных к концу семестра.

В качестве тестового набора данных были взяты данные о текущей аттестации группы 16-ИВТ1 по дисциплине «Программирование». Через интерфейс управления курсами для данной группы был создан курс по программированию, как показано на рис. 4.1.

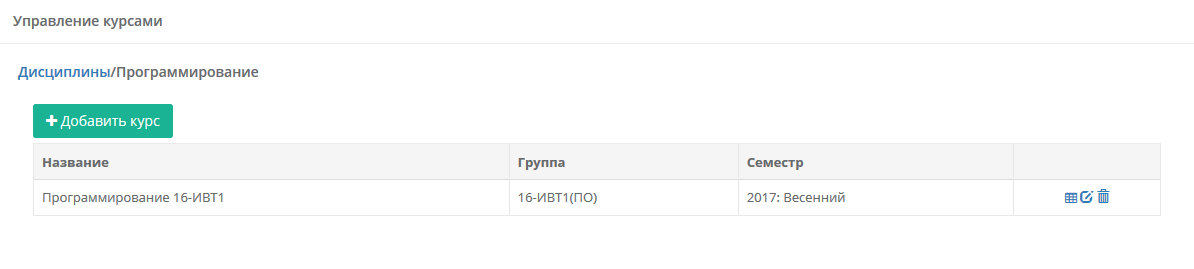


Рис. 4.1. Список курсов по дисциплине «Программирование»

Затем на странице со списком работ курса он был заполнен работами, часть из которых была вынесена на промежуточную аттестацию. Итоговый вид курса представлен на рис. 4.2.

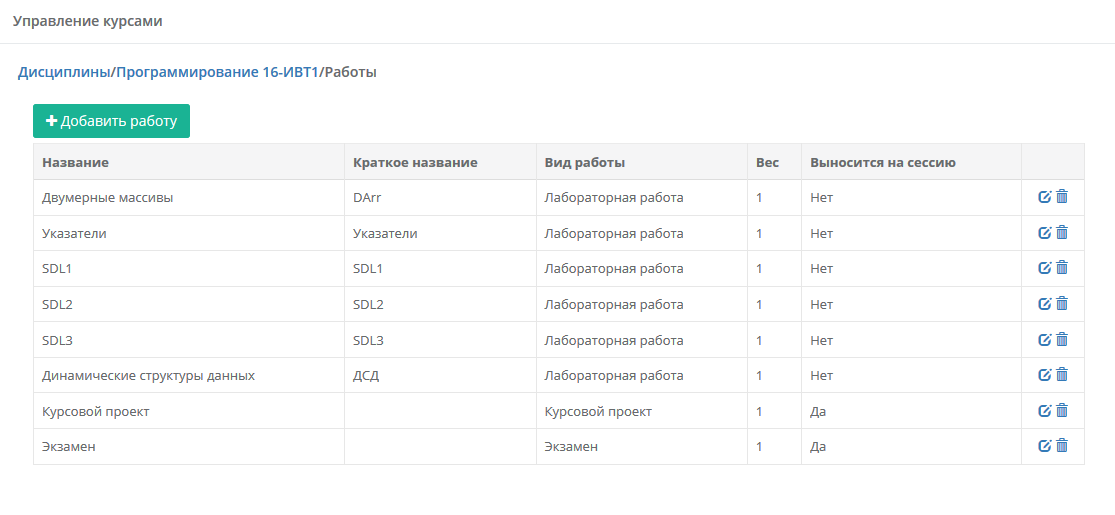


Рис. 4.2. Список работ курса «Программирование 16-ИВТ1»

Далее через интерфейс учета текущей аттестации ведомость группы была заполнена реальными данными, полученными к концу семестра в ходе учебного процесса.

### Применение механизмов, основанных на алгоритмах оценивания работы студентов в семестре

На странице ведомости по внутрисеместровой аттестации был запущен алгоритм автоматического оценивания работы студентов в семестре. Полученные таким образом оценки по первой и второй внутрисеместровой аттестациям, в большей мере совпали с реальными оценками полученными студентами. Результат работы алгоритма представлен на рис. 4.3.

Далее на странице ведомости по промежуточной аттестации был запущен алгоритм автоматического выставления рекомендуемой оценки за экзамен по программированию. В целом полученные таким образом оценки соответствовали реальным данным, полученным в ходе учебного процесса.

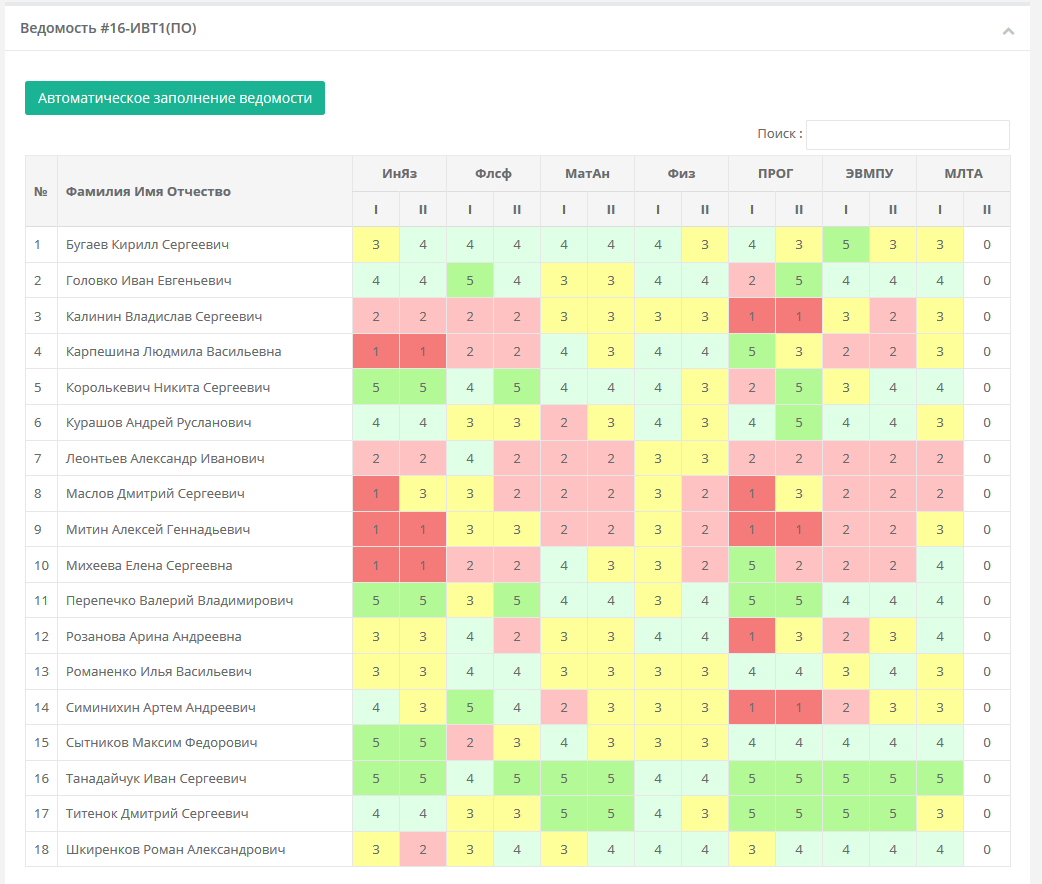


Рис. 4.3. Аттестационная ведомость группы 16-ИВТ1

### Наладка интеграции с сервисом учета контингента

Перед началом работы со списками группы администратором был запущен механизм однократного сопоставления данных в системе мониторинга посещаемости и успеваемости студентов с данными сервиса учета контингента.

Далее для проверки модуля автоматической актуализации списков групп на основе оповещений в сервисе учета контингента в группу 16-ИВТ1 был добавлен тестовый студент «Тест». Модуль корректно обработал пришедшее оповещение и добавил студента, как показано на рис. 4.4.

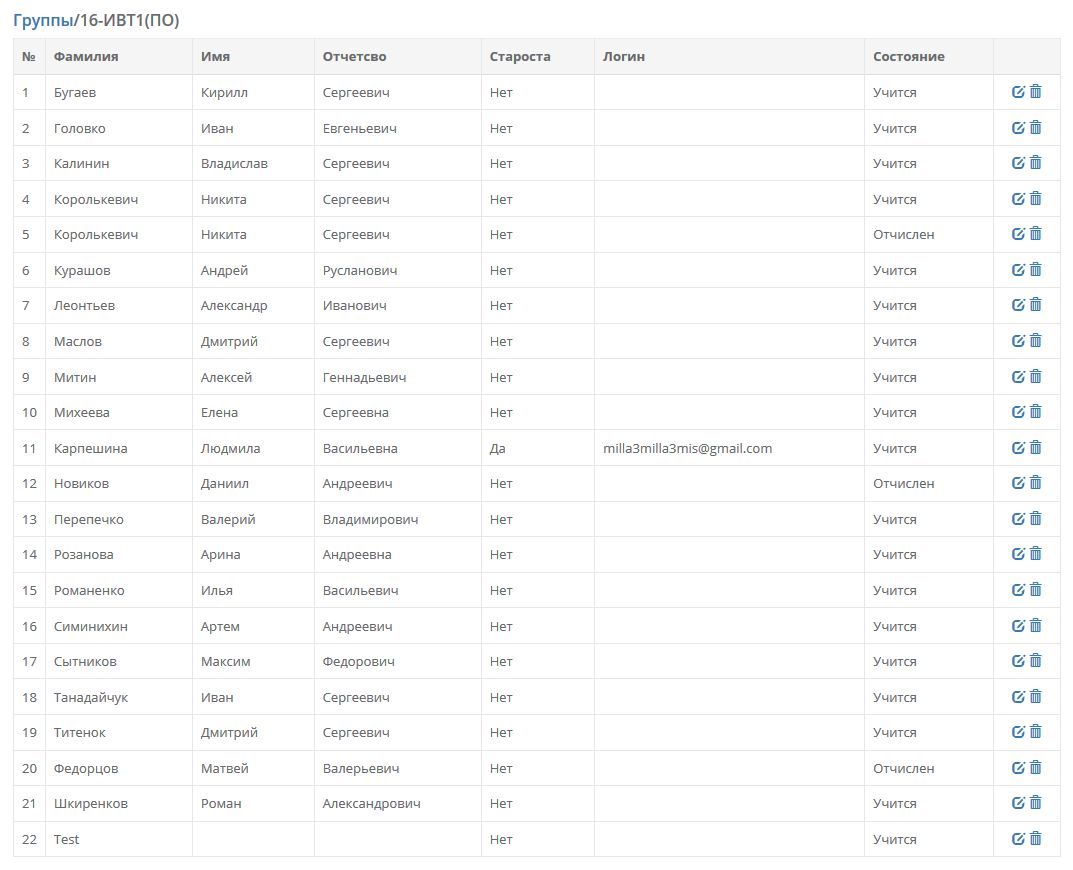


Рис. 4.4. Список студентов группы 16-ИВТ1 после добавления студента

Далее в сервисе учета контингента из группы 16-ИВТ1 был удален тестовый студент «Тест». Модуль корректно обработал пришедшее оповещение и перевел студента в статус «Отчислен», как показано на рис. 4.5.

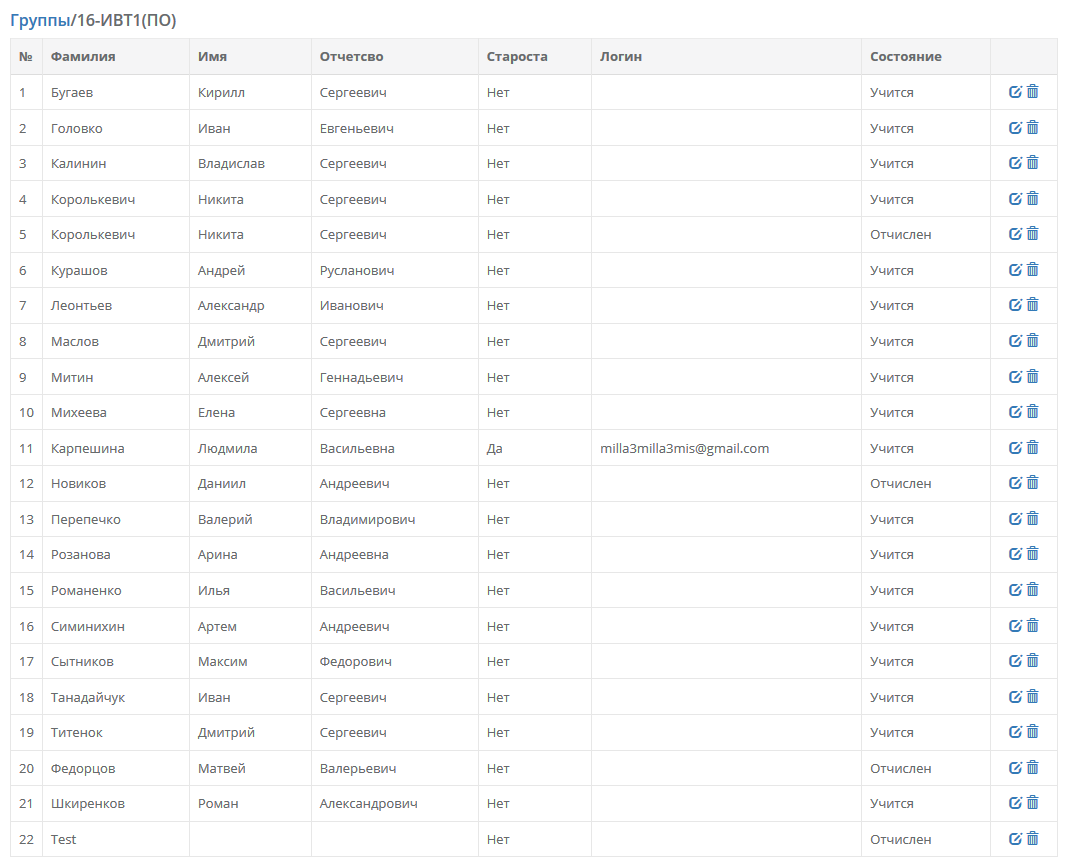


Рис. 4.5. Список студентов группы 16-ИВТ1 после удаления студента

## Выводы по главе

1. После отладочных и пусконаладочных работ финальное тестирование не выявило серьёзных ошибок.

2. Подсистема соответствует заявленным функциональным требованиям.

3. Разработанная подсистема отражает процесс аттестации студентов и позволяет эффективно вести учет результатов аттестации.

# ЗАКЛЮЧЕНИЕ

Диссертационная работа «Подсистема учёта результатов аттестации студентов и интеграционный программный интерфейс для автоматизированной системы мониторинга успеваемости студентов и посещаемости занятий «СУП» выполнена в Брянском государственном техническом университете на кафедре «Информатика и программное обеспечение».

При проведении научных исследований, связанных с темой работы, и решении поставленных в работе задач были достигнуты следующие результаты:

1. На основе анализа способов взаимодействия веб-приложений был сделан вывод о целесообразности применения REST API.
2. Разработаны алгоритмы и формулы для автоматического выставления оценок по внутрисеместровой и промежуточной аттестациям.
3. Разработан программный интерфейс для интеграции системы мониторинга посещаемости и успеваемости со смежными сервисами.
4. Разработана подсистема учета результатов аттестации студентов.

Основные результаты работы докладывались на 71-ой и 72-ой студенческих научных конференциях БГТУ и на VIII Международной научно-практической конференции «Вопросы современных научных исследований». Материалы докладов были опубликованы в виде следующих статей: «Система учета посещаемости и успеваемости студентов. Подсистема учета текущей успеваемости» [34], «Разработка программного интерфейса (API) для манипулирования данными в системе мониторинга успеваемости и посещаемости «СУП» [35], «Разработка алгоритмов выставления оценок по внутрисеместровой и промежуточной аттестации на основе данных о текущей успеваемости и посещаемости» [36].

Промежуточные результаты работы составили основу для веб-приложения учета текущей аттестации, которое было зарегистрировано в Роспатенте (ФИПС).

При дальнейшей работе были созданы еще два веб-приложения подсистемы учета результатов аттестации студентов и программного интерфейса, которые тоже готовы к регистрации.

# СПИСОК ЛИТЕРАТУРЫ

1. Положение о текущем контроле успеваемости и промежуточной аттестации студентов. ФГБОУ ВО «Брянский государственный технический университет». 2016 г. – 13 с.
2. Чуйко, О.И. Разработка информационной системы учета успеваемости студентов на основе облачных технологий. / О.И. Чуйко, С.И. Белозёрова // Интернет-журнал «НАУКОВЕДЕНИЕ». Том 7, №5 сентябрь-октябрь 2015 г. – 15 c.
3. Гуриков, С.Р. Использование электронного модуля успеваемости и посещаемости студентов в техническом университете. / С.Р. Гуриков // Политематический журнал научных публикаций «Дискуссия». Выпуск: №1 (53) январь 2015 г. – С. 96-100.
4. Борисов, З.В. Система мониторинга успеваемости и посещаемости в вузах: дис. магистра техн.:09.14.01 / З.В. Борисов. БГТУ – Б , 2016. – 93с.
5. Калевко, В.В. Особенности синхронизации данных между мобильным приложением «АРМ Преподавателя» и веб-сервисом. / В.В. Калевко // Сборник 70 студ конференции БГТУ. 2016 г. – С. 342-344.
6. Борисов, З.В. Разработка мобильного приложения для системы мониторинга успеваемости студентов и посещаемости занятий / З.В. Борисов // Материалы 70-ой студенческой научно-практической конференции, посвященной 70-летию победы в Великой Отечественной войне. – Брянск: БГТУ, 2015. – с. 314-315.
7. Булатицкий, Д.И. Разработка системы мониторинга посещаемости занятий и успеваемости студентов / З.В. Борисов, Д.И. Булатицкий // Научные перспективы XXI века. Достижения и перспективы нового столетия. – 2015. № X (17), ноябрь. с. 8-11. Режим доступа: http://edu-science.ru/wp-content/uploads/2016/03/edu-17\_p1\_6-158.pdf
8. Борисов, З.В. Анализ и визуализация данных в системе учёта посещаемости и успеваемости студентов / З.В. Борисов // Материалы 70-ой студенческой научно-практической конференции. – Брянск: БГТУ, 2016.
9. Булатицкий, Д.И. Применение технологий OLAP для анализа и визуализации данных в системе учёта посещаемости и успеваемости студентов / З.В. Борисов, Д.И. Булатицкий // III Международная студенческая конференция «Инновационные направления разработки и использования информационных систем и технологий» – Брянск: БГАУ, 2016.
10. Никитин, А.А. Разработка рабочего места заведующего кафедрой в системе мониторинга успеваемости студентов и посещаемости занятий /   
    А.А. Никитин // Материалы 70-й студенческой научной конференции / под ред. И.Г. Чернышовой. – Брянск: БГТУ, 2015.
11. Сергеенко, П.В. Разработка рабочего места преподавателя, старосты и студента в системе мониторинга успеваемости студентов и посещаемости занятий / П.В. Сергеенко // Материалы 70-й студенческой научной конференции / под ред. И.Г. Чернышовой. – Брянск: БГТУ, 2015.
12. Копелиович, Д.И. Автоматизированная система мониторинга успеваемости студентов университета с применением технологии OLAP/ Проблемы и перспективы развития образования в России: сборник материалов XI Международной научно-практической конференции: / Под общ.ред. С.С. Чернова. – Новосибирск: Издательство НГТУ, 2011. – с 152-156. Режим доступа: http://cyberleninka.ru/article/n/avtomatizirovannaya-sistema-monitoringa-uspevaemosti-studentov-universiteta-s-primeneniem-tehnologii-olap
13. Нормативное обеспечение деятельности университета: сб. нормативных документов: в 2 ч. Ч. 2 / Сост. А.В. Лагерев, О.А. Горленко. - Брянск: БГТУ 2007. - 203 с.
14. Мазорчук, М.С. Автоматизированная информационная система «Деканат» / М.С. Мазорчук, Ю.В. Тыжненко // Объектные системы. 2011. №1 (3) С.52-58. Режим доступа: <http://cyberleninka.ru/article/n/avtomatizirovannaya-informatsionnaya-sistema-dekanat>
15. Об образовании в Российской Федерации: Федер.закон от 29 декабря 2012 г. N 273-Ф3 // Собрание законодательства РФ. – 2012. – N 53. – ст. 7598.
16. Об утверждении порядка организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры: Приказ от 19 декабря 2013 г. № 1367 / Мин-во образования и науки РФ // Собрание законодательства РФ. – 2013 . – № 57. – ст. 7936.
17. Жетесова, Г.С. Автоматизация контроля знаний студентов при модульной форме обучения на основе программного обеспечения / Г.С. Жетесова, М.А. Ерназарова // Фундаментальные исследования. – 2014. – № 6-2. – С. 355-359.
18. Овчаренков, Э.А. Рейтинговая оценка знаний студентов / Э.А. Овчаренков // Современные проблемы науки и образования. – 2014. – № 5. – С. 201-203.
19. Методология функционального моделирования IDEF0. Режим доступа: <http://www.nsu.ru/smk/files/idef.pdf>
20. Электронный журнал успеваемости «Магеллан». Режим доступа: <https://magellanius.ru/journal/>
21. Система учета абитуриентов и студентов. Компания «Тауруна». Режим доступа: <http://www.tauruna.ru/products/students.html>
22. Программный комплекс «Электронные ведомости». ММИС. Режим доступа: <http://www.mmis.ru/Default.aspx?tabid=159>
23. ISO – International Organization for Standardization [Электронный ресурс]. – Режим доступа: http://www.iso.org/iso/catalogue\_detail? csnumber=43447.
24. Fielding, R. T. Architectural Styles and the Design of Network-based Software Architectures / R. T. Fielding, – Irvine, 2000.
25. SOAP Версия 1.2: Учебник для начинающих. Режим доступа: <https://www.w3.org/2002/07/soap-translation/russian/part0.html>
26. Официальный сайт XML-RPC. Режим доступа: <http://xmlrpc.scripting.com/>
27. Васин, А.В. Программная платформа автоматизации деятельности кафедры «Информатика и программное обеспечение» / А.В. Васиин // Материалы VIII Международной научно-практической конференции «Достижения молодых ученых в развитии инновационных процессов в экономике, науке и образовании». – Брянск: БГТУ, 2016 г. – С. 137-141.
28. Официальный сайт JSON API. Режим доступа: <http://jsonapi.org/>
29. ASP.NET MVC 5 – Режим доступа: http://www.asp.net/mvc/mvc5
30. Учебник. Среда SQL Server Management Studio. Режим доступа: <https://msdn.microsoft.com/ru-ru/library/bb934498(v=sql.120).aspx>
31. SQL Server Management Studio — единое средство управления и среда разработки в MS SQL Server 2012 – Режим доступа: <http://tavalik.ru/sql-server-management-studio/>
32. Веб-сервер (IIS). Режим доступа: <https://technet.microsoft.com/ru-ru/library/cc753433(v=ws.10).aspx>
33. Введение в язык C# и .NET Framework – Режим доступа: <https://msdn.microsoft.com/ru-ru/library/z1zx9t92.aspx>
34. Исаев, И.С. Система учета посещаемости и успеваемости студентов. Подсистема учета текущей успеваемости / И.С. Исаев // Материалы 71-ой студенческой научной конференции. – Брянск: БГТУ, 2016 г.
35. Исаев, И.С. Разработка программного интерфейса (API) для манипулирования данными в системе мониторинга успеваемости и посещаемости «СУП» / И.С. Исаев // Материалы 72-ой студенческой научной конференции. – Брянск: БГТУ, 2017 г., в печати.
36. Булатицкий, Д.И. Разработка алгоритмов выставления оценок по внутрисеместровой и промежуточной аттестации на основе данных о текущей успеваемости и посещаемости / Д.И. Булатицкий, И.С. Исаев // VIII Международная научно-практическая конференция «Вопросы современных научных исследований» – Омск: Научный центр «Орка», 2017 г., в печати.

# ПРИЛОЖЕНИЕ А



# ПРИЛОЖЕНИЕ Б

