СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний момент современное общество все больше ведет свое развитие к тому, чтобы как можно сильнее упростить повседневную жизнь, собственную работу. Одним из способов является внедрение компьютерных технологий в любую деятельность человека, начиная от написания документа и заканчивая проведением сложнейших математических расчетов. Внедрение компьютеров значительно облегчает большинство трудоемких операций, на которые раньше уходили часы, сейчас уходит всего пару минут.

Автоматизированные системы в настоящее время широко используются во всех звеньях жизнедеятельности человека.

Для хранения и работы с информацией об оценках студентов необходимо иметь систему, благодаря которой можно будет перенести письменный труд по учету результатов в приложение.

Целью данной работы является разработка автоматизированной системы учета успеваемости студентов вуза.

Для достижения данной цели необходимо выполнить следующий ряд задач:

- Описать предметную область
- Определить требования к системе
- Обозреть методы решения поставленной задачи
- Разработать модели представления системы и описать их
- Описать информационную модель системы
- Описать и реализовать алгоритмы работы приложения
- Написать руководство для пользователя

1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И ОПРЕДЕЛЕНИЕ ТРЕБОВАНИЙ К СИСТЕМЕ

Высшее учебное заведение организационно состоит из факультетов. В свою очередь факультет включается в себя несколько кафедр. На кафедрах работают как штатные преподаватели, так и преподаватели-совместители. Существует три основных формы обучения: очная. заочная, вечерняя [1].

Каждая выпускающая кафедра обучает студентов по нескольким специальностям. Каждый год на каждую специальность, на основе конкурсного отбора по результатам вступительных экзаменов зачисляются студенты. На каждого студента заводится личное дело, которое хранится в течении всего срока обучения, а затем передается в архив. Для правильной организации процесса обучения, студентов каждой специальности делят на группы. В каждой группе деканом назначается староста.

Процесс обучения осуществляется на основе учебных планов, которые разрабатываются на выпускающих кафедрах и затем утверждаются деканом и начальником учебного управления. В учебных планах по семестрам отражаются наименования преподаваемых дисциплин, вид аттестации и ФИО преподавателя, его должность и звание.

Различают следующие виды аттестации:

Контроль знаний студентов осуществляется путем проведения различных видов аттестации: промежуточной текущей, итоговой. Аттестация выставляется на основании баллов, которые устанавливаются преподавателем по каждой дисциплине на основании разработанной им процедуры мониторинга успеваемости студентов.

Текущая аттестация проводится по трем контрольным точкам в течение семестра. Отметка в контрольной точке аккумулирует в себе результат выполнения всех видов работ студентом за данный период в соответствии с учебно-методическим комплексом дисциплины. Данная система оценки знаний является базой для перехода вуза к зачету учебных единиц (кредитов)

в соответствии с положениями Болонского соглашения в области образования.

По окончании каждого семестра проводится промежуточный контроль в форме экзамена, зачета, защиты курсового проекта или курсовой работы.

По результатам текущей и промежуточной аттестации студенту выставляется итоговая оценка по каждой изучаемой дисциплине.

За систематическую неудовлетворительную аттестацию к студентам в установленном порядке могут быть применены меры дисциплинарного воздействия вплоть до отчисления за академическую неуспеваемость.

В последнем семестре обучения студенты сдают государственный экзамен и защищают дипломный проект. Перед сдачей государственного экзамена по специальности на каждого студента высчитывается средний балл по итогам полученных оценок в течение всего периода обучения. Данная оценка является критерием при сдаче студентом государственного экзамена по специальности. В дальнейшем, при переходе вуза к зачету учебных единиц, при накопительной модели начисления балов по учебным дисциплинам каждый студент обязан набрать необходимое количество баллов за весь период обучения.

Дипломный проект является заключительной самостоятельной работой студента на последнем году обучения. По своему содержанию он должен предоставлять соьой оригинальное научное исследование, освещающее одну из актуальных проблем в области будущей профессиональной деятельности выпускника.

Основными отчетными документом по успеваемости студентов являются отчетные документы по результатам текущей, промежуточной и итоговой аттестации, которые представляются в учебное управление и используются деканом и его заместителями для анализа и принятия управленческих решений.

Таким образом, учет успеваемости студентов - сложная задача, требующая большой обработки данных. Следовательно, эта информация важна не только для деканата, но и для кафедры.

Использование компьютерного подхода в решении данной проблемы позволяет значительно сократить временные и трудовые затраты на выполнение поиска и предоставление информации о результатах успеваемости групп, а также отдельных студентов, выявление отстающих студентов. Возникает необходимость автоматизировать учет успеваемости студенческих групп и учет персональных данных.

Функциональная модель учета успеваемости представлена на рисунке 1.1.

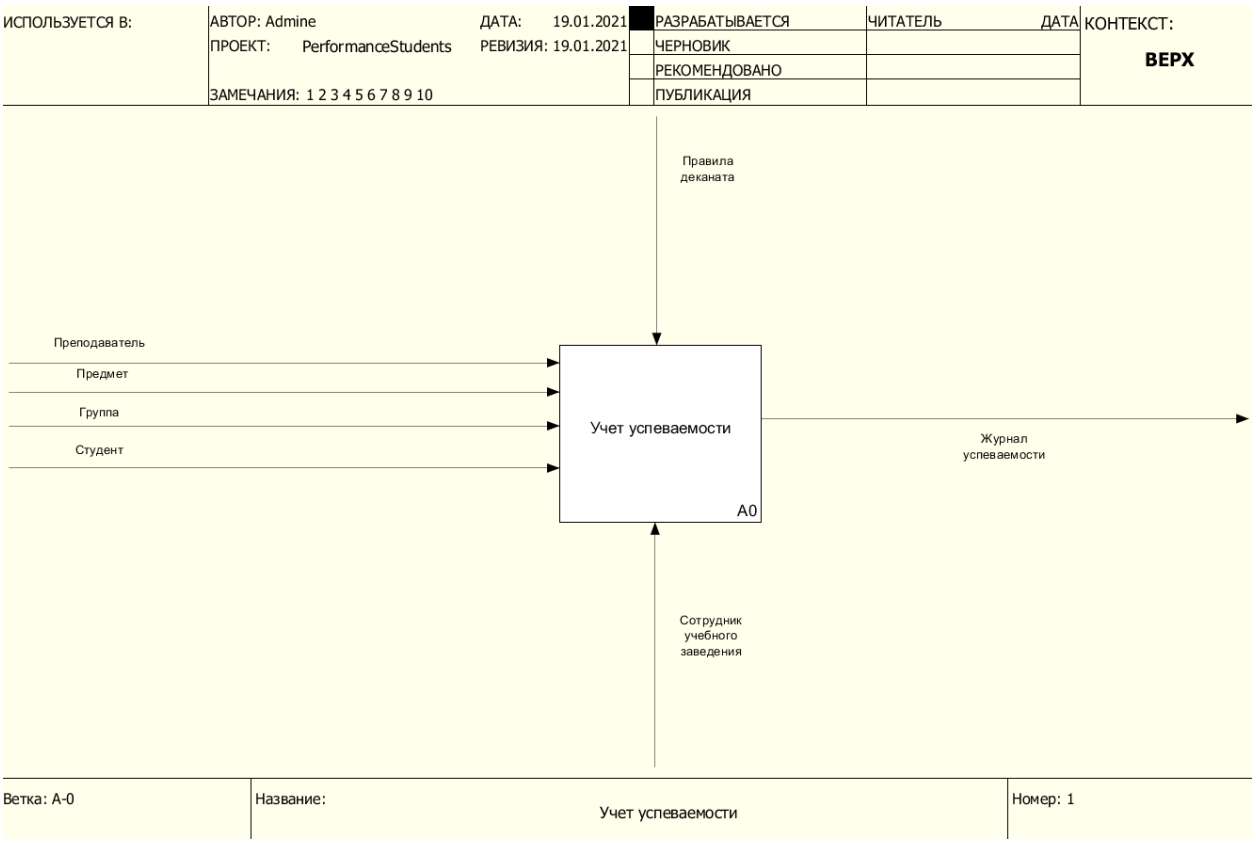

Рисунок 1.1 – Функциональная модель предметной области

Декомпозиция функциональной модели представлена на рисунке 1.2 [2]. Модель разбивается на 3 подпроцесса: заполнение информации о предмете и

учебных группах, ввод данных о студентах и преподавателях и заполнение таблицы учета успеваемости.

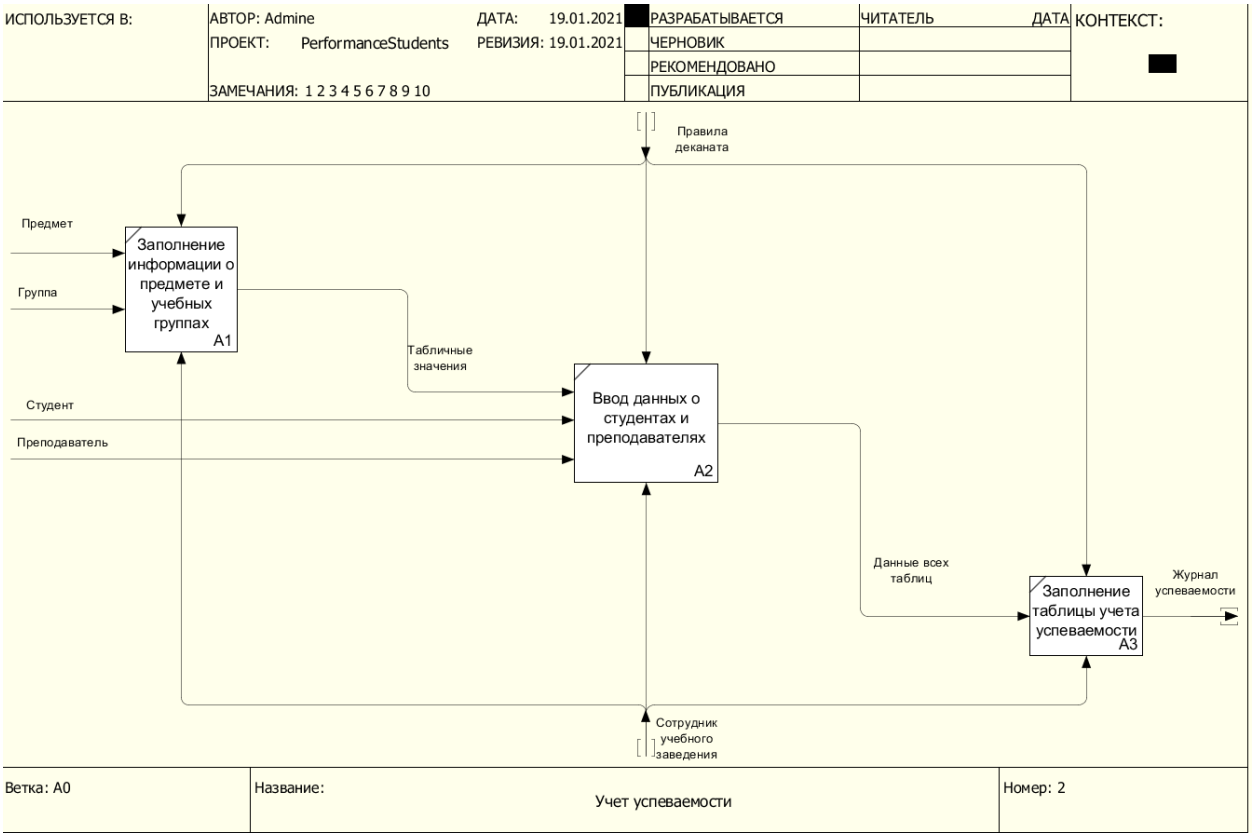

Рисунок 1.2 – Декомпозиция функциональной модели

Поскольку система предназначена для учета результатов розыгрышей, а не проведения розыгрышей, к ней определены следующие требования:

- Хранение информации о преподавателях
- Хранение информации о студентах
- Хранение информации о группах студентов
- Хранение информации о предметах
- Хранение оценок студентов по предметам
- Интуитивно понятный интерфейс
- Проект должен представлять из себя веб-приложение

2. ПОСТАНОВКА ЗАДАЧИ И ОБЗОР МЕТОДОВ ЕЁ РЕШЕНИЯ

Разработать веб-приложение для учета успеваемости студентов вуза.

В системе предполагается наличие следующих функций:

− Ввод, редактирование и удаление информации о преподавателях

− Ввод, редактирование и удаление информации о предметах

− Ввод, редактирование и удаление информации о группах

− Ввод, редактирование и удаление информации о студентах

− Ввод, редактирование и удаление информации о оценках

Информационная система должна быть реализована в виде web-приложения на языке Java с использованием технологий Servlets , JSP, HTML 5, CSS 3, JavaScript. Архитектура приложения должна быть выполнена в архитектуре паттерна MVC. Интерфейс приложения должен быть оформлен с использованием каскадных таблиц стилей(CSS).

Доступ к данным в СУБД должен осуществлять через драйвер JDBC, поставляемый производителем СУБД.

Для запуска приложения использовать Apache Tomcat, JDK 8/ JRE8 и выше.

Необходимо создать интерфейс, который был бы прост и понятен для пользователя. Среда разработки должна позволять с наименьшими затратами справиться с поставленной на данном этапе задачей. Этим требованиям соответствует интегрированная среда разработки, такая как Eclipse.

Интерфейс программы и данные должны быть только на русском языке.

3. СПЕЦИФИКАЦИЯ СИСТЕМЫ

При запуске приложения, пользователь может перейти на сайт на главную страницу, с помощью которой может перейти на любую страницу, используя главное меню.

Благодаря меню, пользователь может перейти на страницу групп, на которой будет выведен список всех групп в системе. При необходимости, пользователь может редактировать это список.

Следующая страница, на которую пользователь может перейти, это студенты. На этой странице пользователь также видит список всех студентов и может его изменять или добавлять новых, указывая их принадлежность к группе.

После добавления групп и студентов, пользователь может перейти на страницу преподавателей, на которой добавляет новых преподавателей в систему.

Как только преподаватели созданы, пользователь может перейти на страницу предметов, на которой можно создавать предметы, которые ведет определенный преподаватель, указывая его в выпадающем списке.

Когда сделаны все перечисленные работы, пользователь может перейти на страницу учета оценок, на которой будет записывать какому студенту по какому предмету и в каком семестре какую оценку поставили.

4. МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ

В данном разделе будет продемонстрировано моделирование информационной системы с помощью стандарта UML, который использует графические обозначения для создания абстрактной модели системы и предназначен для определения, визуализации, проектирования и документирования в основном программных систем. UML позволяет описать систему практически со всех возможных точек зрения и разные аспекты поведения системы [3].

Для данной курсовой работы были построены такие диаграммы, как диаграммы вариантов использования, последовательности, состояний, классов, развертывания и компонентов.

4.1 Диаграмма вариантов использования

Диаграмма вариантов использования состоит из актеров, для которых система производит действие и собственно действия Use Case, которое описывает то, что актер хочет получить от системы [4].

В данной диаграмме вариантов использования в роли актёров выступает администратор сайта. Он может работать с каждой сущностью используя все стандартные операции. Все это видно из диаграммы на рисунке 4.1.

Рисунок 4.1 – Диаграмма вариантов использования

4.2 Диаграмма состояний

Диаграмма состояний предназначена для отображения состояний объектов системы, имеющих сложную модель поведения [5]. Она показывает пространство состояний системы или ее элементов, события, которые влекут переход из одного состояния в другое, действия, которые происходят при изменении состояния. Объекты меняют своё состояние в ответ на

происходящие события и стечением времени. Диаграмма состояний представляет состояния объекта и переходы между ними, а также начальное и конечное состояние объекта

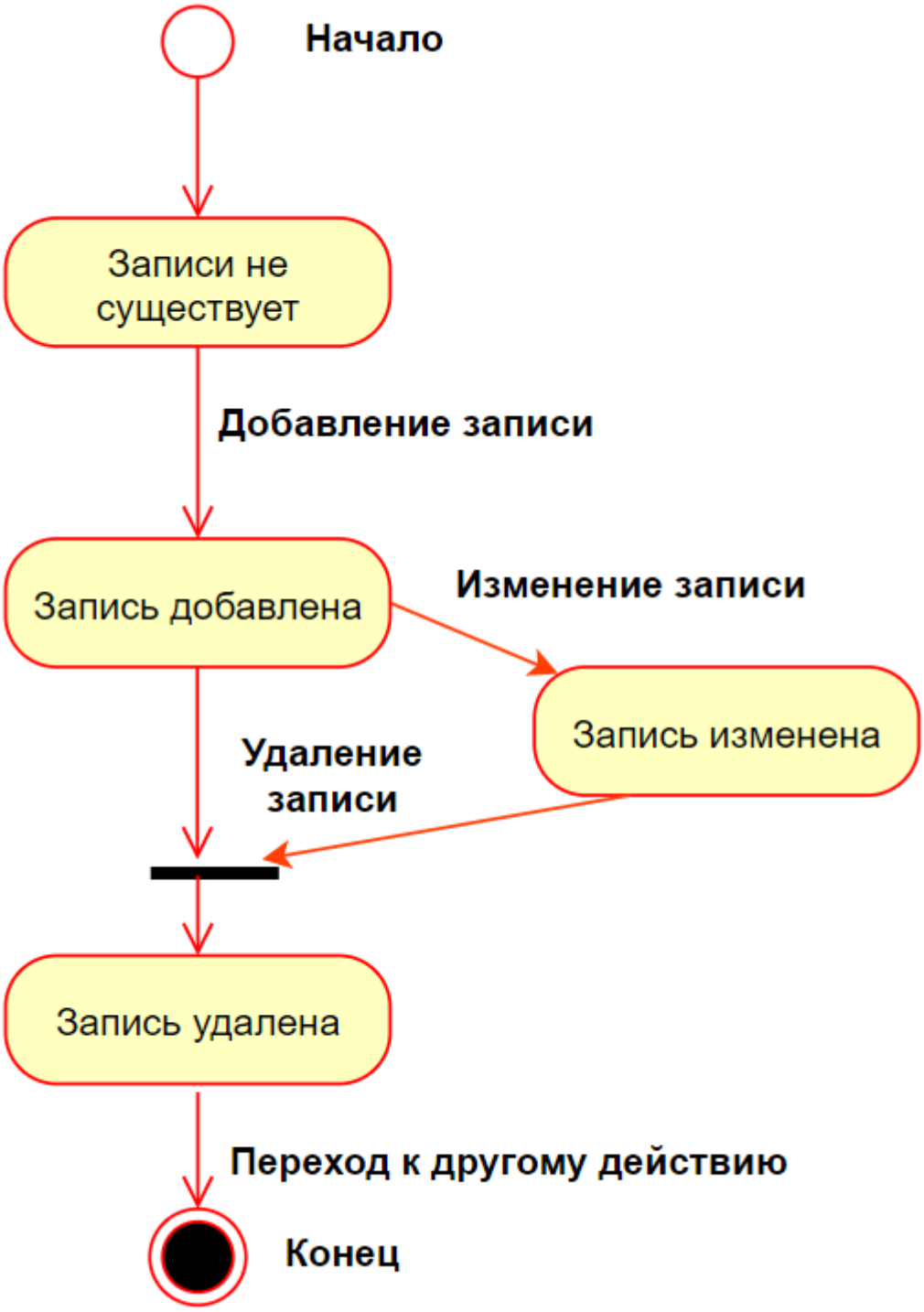

Рисунок 4.2 – Диаграмма состояний.

4.3 Диаграмма последовательностей

Для моделирования взаимодействия объектов во времени в языке UML используются диаграммы последовательностей [6]. Для демонстрации диаграммы последовательностей рассмотрим диаграмму, представленную в на рисунке 4.3.

Действие начинается с того, что клиент запрашивает какую-либо информацию. Программа обращается к сервлетам, которые обращаются к соответствующим сервисам, которые вызывают соответствующие методы из dao – уровня. Последние методы обращаются к базе данных, формируют информацию для отправки и возвращают на уровень выше.

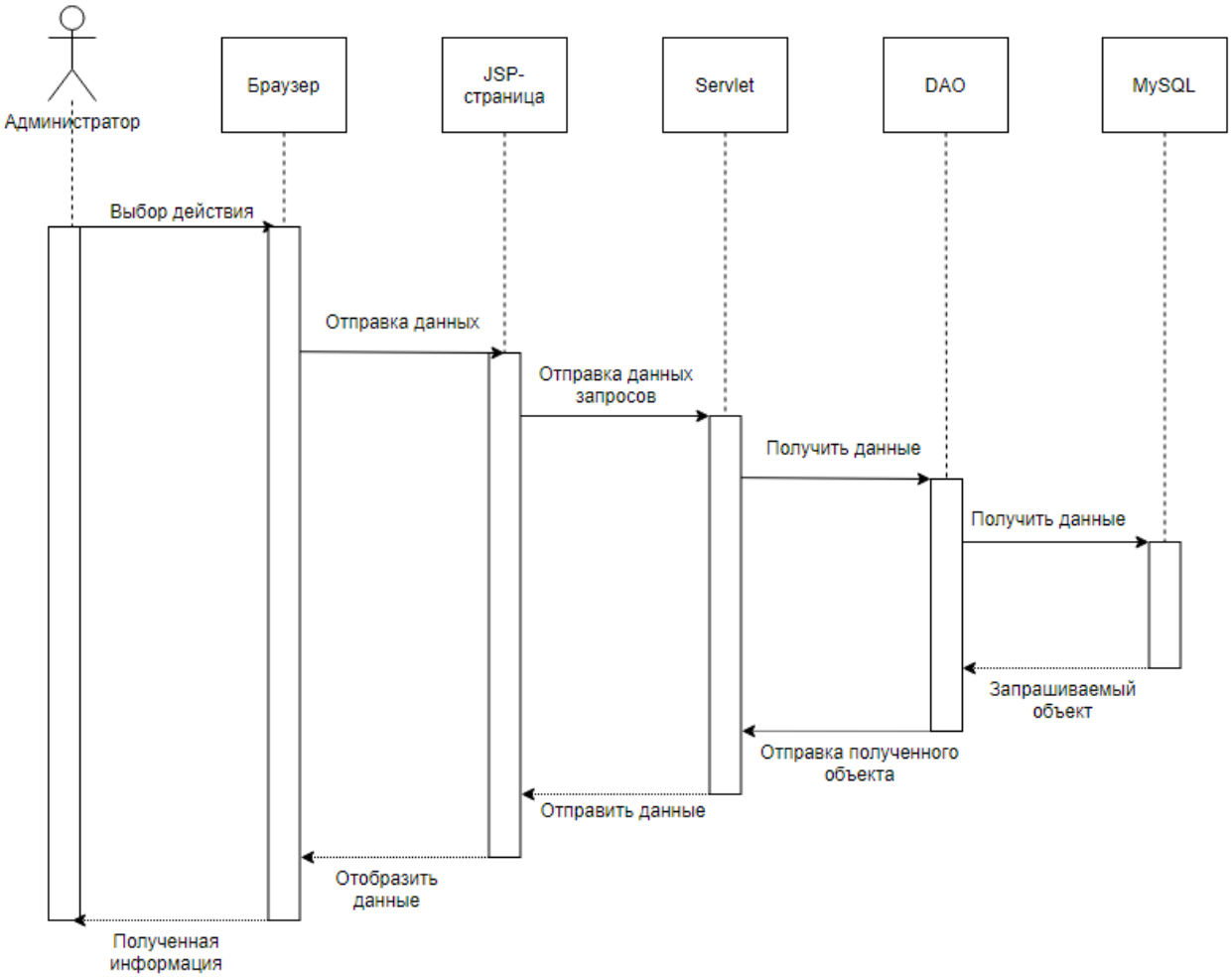

Рисунок 4.3 – Диаграмма последовательности

4.4 Диаграмма классов

Диаграмма классов описывает структуру системы, показывая её классы, их атрибуты и операторы, а также взаимосвязи этих классов.

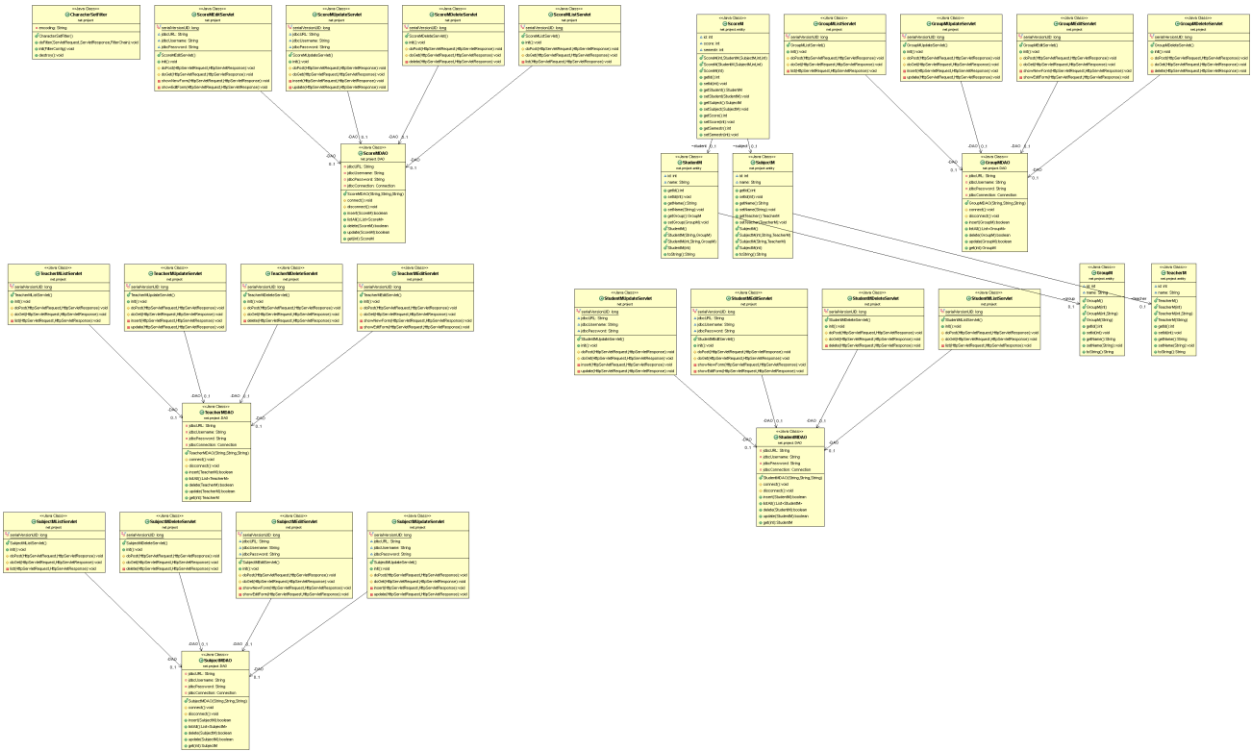

Рисунок 4.4. – Диаграмма классов

4.5 Диаграмма развёртывания

Диаграмма развёртывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения [7].

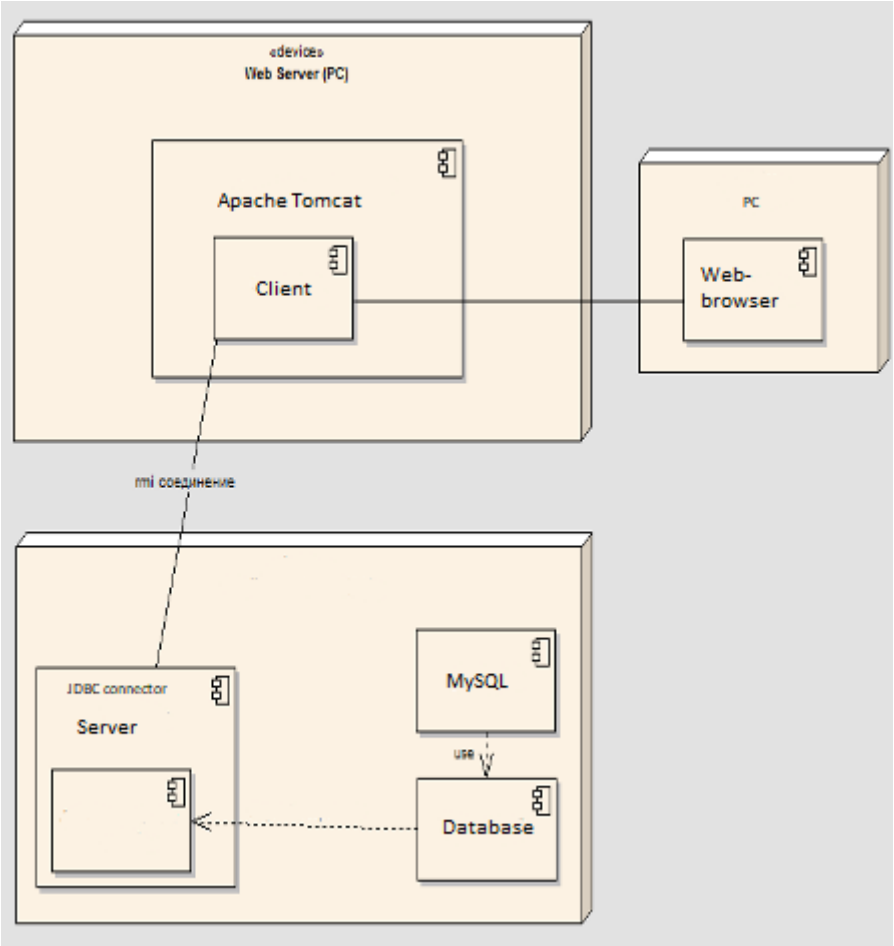

Рисунок 4.5 – Диаграмма развертывания

5. ИНФОРМАЦИОННАЯ МОДЕЛЬ СИСТЕМЫ И ЕЁ ОПИСАНИЕ

Процесс построения информационной модели состоит из следующих шагов:

- определение сущностей;
- определение зависимостей между сущностями;
- задание первичных и альтернативных ключей;
- определение атрибутов сущностей;
- составление логической (logical) модели;
- переход к физическому (physical) описанию модели.

Логический уровень означает прямое отображение фактов из реальной жизни [8]. На логическом уровне не рассматривается использование конкретной СУБД, не определяются типы данных и не определяются индексы для таблиц.

На рисунке 5.1 представлена логическая модель данной системы

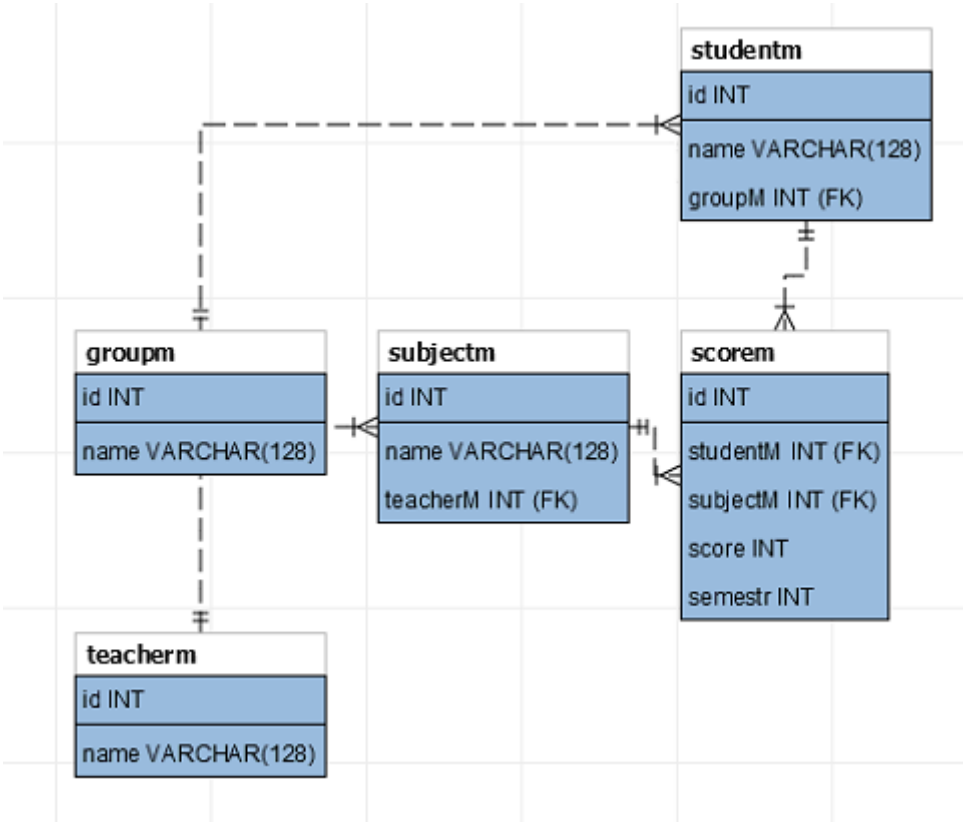

Рисунок 5.1 – логическая модель системы

Проанализировав предметную область, в проекте было решено создать следующие сущности: группа, студент, преподаватель, предмет, оценки. Ниже будут описаны атрибуты каждой сущности

Группа (groupm)

- Id – уникальный номер (первичный ключ)
- Name – название группы

Студент(studentm)

- Id – уникальный номер (первичный ключ)
- Name – ФИО студента
- groupm – идентификатор группы

Преподаватель (teacherm)

- Id – уникальный номер (первичный ключ)
- Name – ФИО преподавателя

Предмет (subjectm)

– Id – уникальный номер (первичный ключ)

– name – название предмета

– teacherm – идентификатор преподавателя

Оценки (score)

– Id – уникальный номер (первичный ключ)

– studentm– идентификатор студента

– subjectM – идентификатор предмета

– score – оценка

– semestr– номер семестра

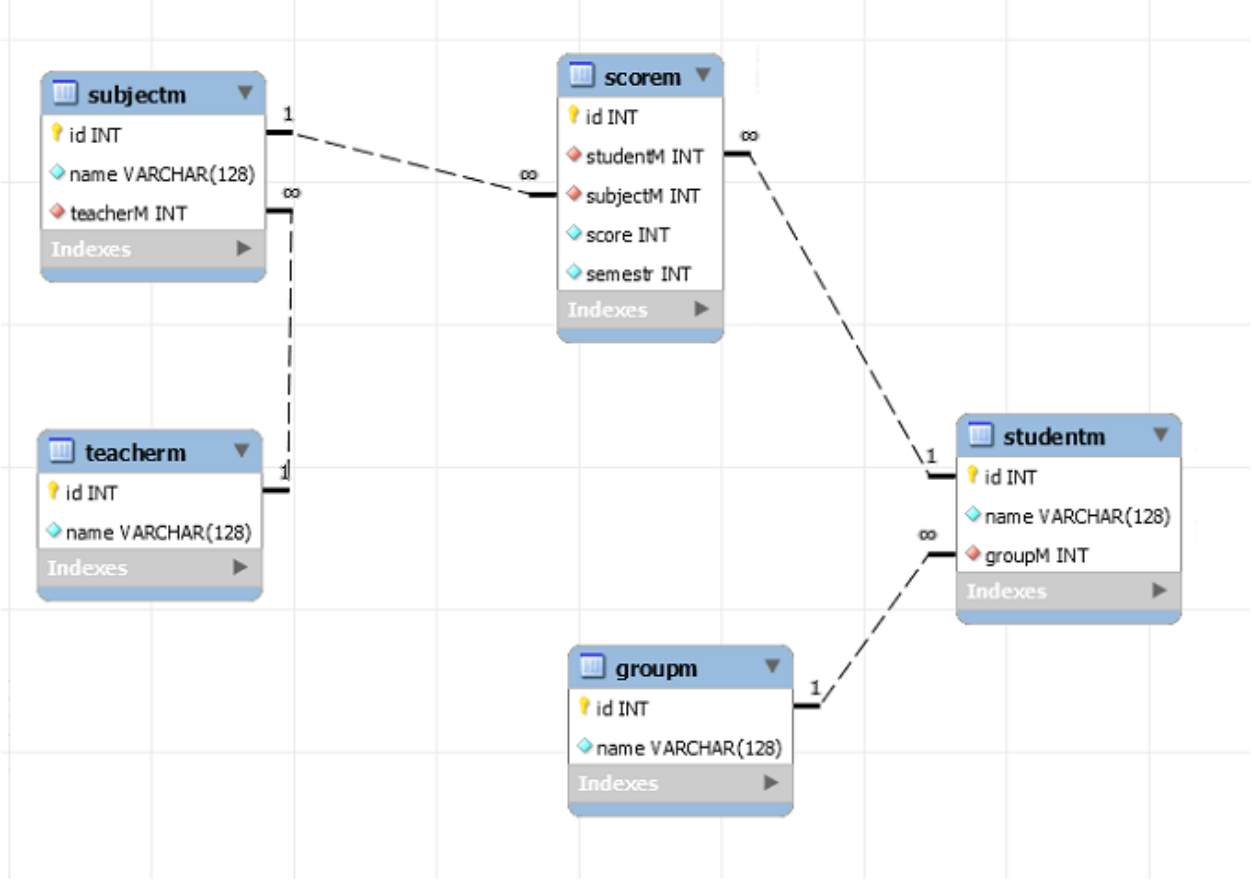

Рисунок 5.2 – физическая модель системы

6. ОБОСНОВАНИЕ ОРИГИНАЛЬНЫХ РЕШЕНИЙ ПО ИСПОЛЬЗОВАНИЮ ТЕХНИЧЕСКИХ И ПРОГРАММНЫХ СРЕДСТВ

При реализации данной программы были использованы следующие решения:

Использование шаблона MVC (Model-View-Controller), реализация, которого позволила чётко и структурировано разграничить части программы, что способствует удобной и быстрой расширяемости [9].

Шаблон MVC позволяет разделить данные, представление и обработку действий пользователя на три отдельных компонента:

- Модель (Model). Модель предоставляет данные (обычно для View), а также реагирует на запросы (обычно от контроллера), изменяя своё состояние.

- Представление (View). Отвечает за отображение информации (пользовательский интерфейс).

- Поведение (Controller). Интерпретирует данные, введённые пользователем, и информирует модель и представление о необходимости соответствующей реакции.

Важно отметить, что как представление, так и поведение зависят от модели. Однако модель не зависит ни от представления, ни от поведения. Это одно из ключевых достоинств подобного разделения. Оно позволяет строить модель независимо от визуального представления, а также создавать несколько различных представлений для одной модели.

Использование шаблона DAO (Data Access Object), реализация которого удобна для абстрагирования и инкапсулирования доступа к источнику данных. Он представляет собой объект, который предоставляет абстрактный интерфейс к какому-либо типу базы данных или механизму хранения. Определённые возможности предоставляются независимо от того, какой механизм хранения используется и без необходимости специальным образом соответствовать этому механизму хранения.

7. ОПИСАНИЕ ОБОБЩЕННОГО АЛГОРИТМА И АЛГОРИТМОВ ПРОГРАММНЫХ МОДУЛЕЙ.

Бизнес-логика данного проекта, как было написано в требованиях, сосредоточена в серверной части. Там реализованы такие операции, как соединение с базой данных, добавление данных в базу, удаление записей из неё, чтение и редактирование данных.

Когда пользователь выбирает определенную операцию, клиент вызывает соответствующие методы сервисов, передавая в качестве параметров необходимые данные. После того, как клиент вызвал метод, сервер получит информацию, он обрабатывает данный запрос, связывается с базой данных, если ему это требуется, и посылает клиенту результат его работы.

Ниже представлен алгоритм работы пользователя.

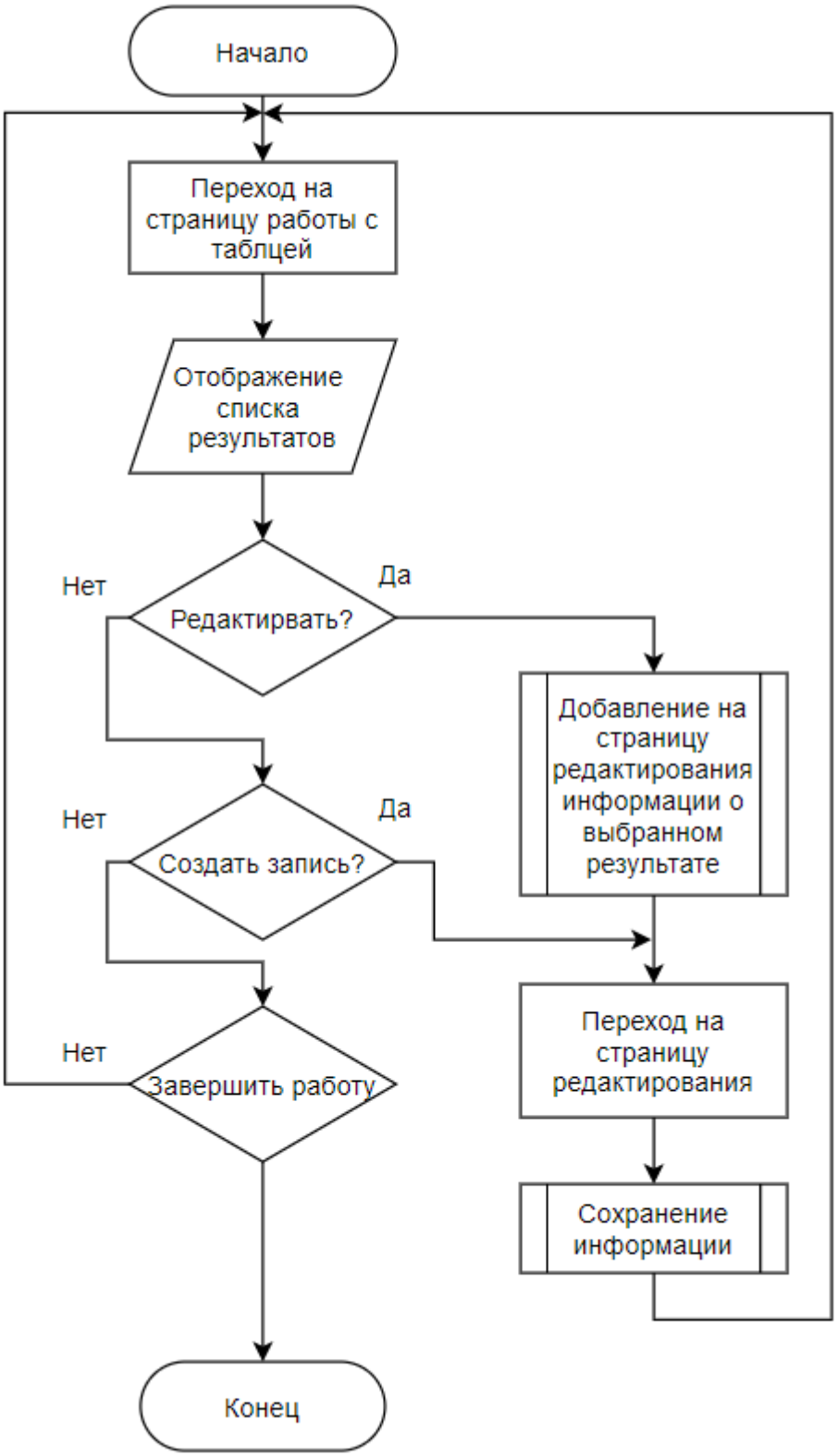

Рисунок 7.1 – Алгоритм работы с результатами

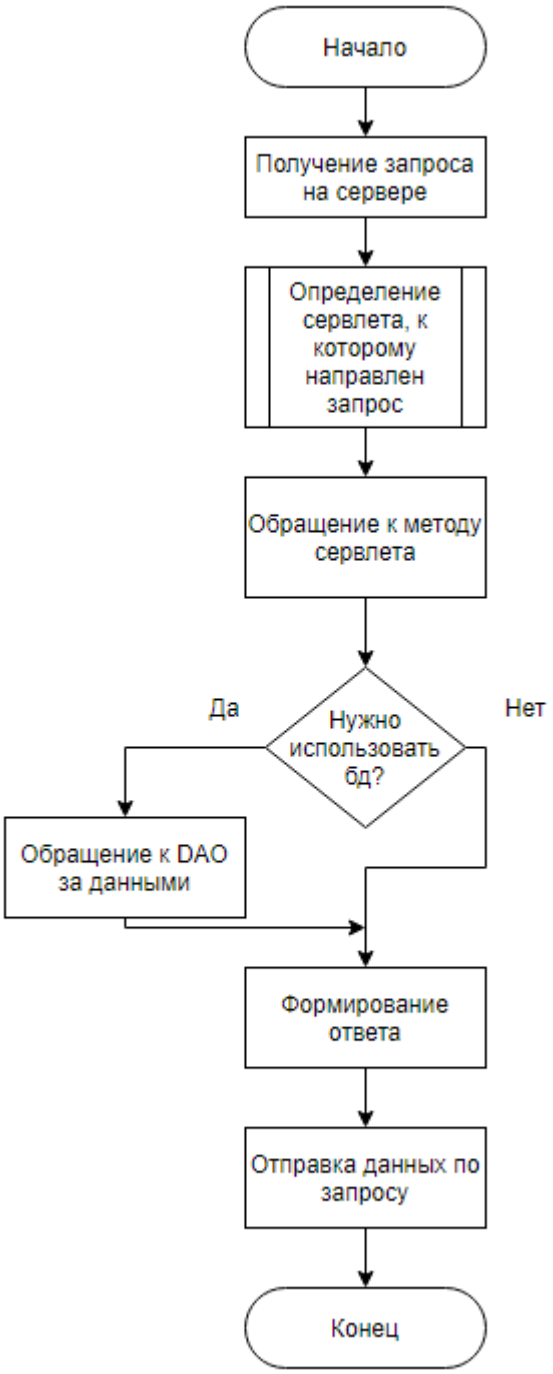

Рисунок 7.2 – Алгоритм работы серверной части

8. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Для запуска через IDE (к примеру Eclipse), нужно лишь открыть в нем проект и запустить приложение как серверное приложение (Run as – Server application).

Затем в самой IDE откроется страница приложения. Или уже после запуска можно перейти в браузер по адресу http://localhost:8080/PerformanceStudents/ и будет то же запущенное приложение.

Если запускать приложение без IDE (вне среды разработки), то нужно экспортировать проект в war-файл для запуска через Tomcat.

Файл экспортировать можно куда угодно, но в конце его нужно положить в папку webapp Tomcat-a. Затем в папке томката в папке bin запустить приложение соответствующим файлом

И все, снова открывать в браузере уже запущенное приложение

Приложение имеет простой, интуитивно-понятный интерфейс, доступный обычному пользователю. При запуске данного приложения вы увидите главную страницу (рисунок 8.1).

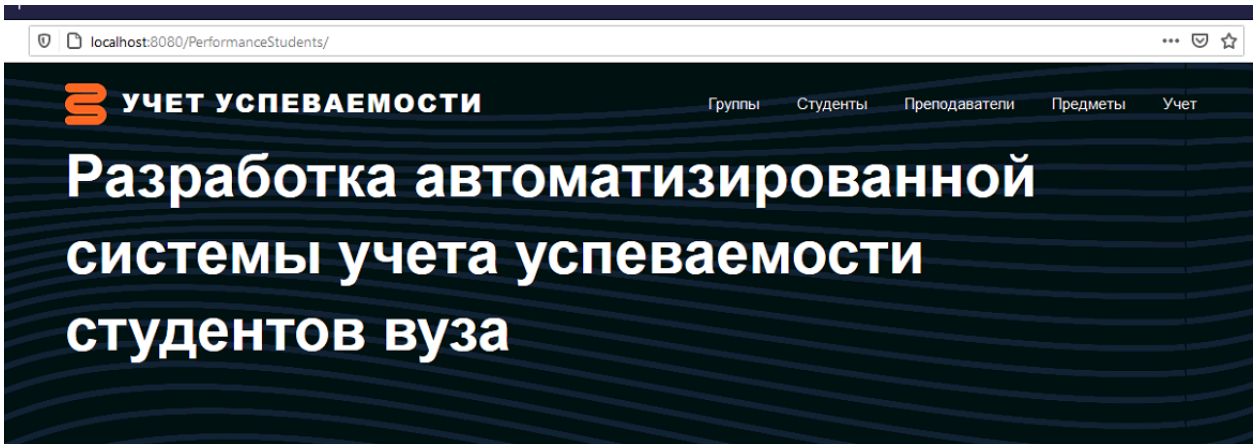

Рисунок 8.1 – Главная страница

Данная страница является главной. Она не наделена какой-либо лишней информации, чтобы можно было спокойно разобраться со всеми пунктами меню. С помощью нее можно перейти к работе с любой сущностью, используя меню.

Для работы с призами достаточно нажать на кнопку меню «Группы». Вид этой странице представлен ниже.

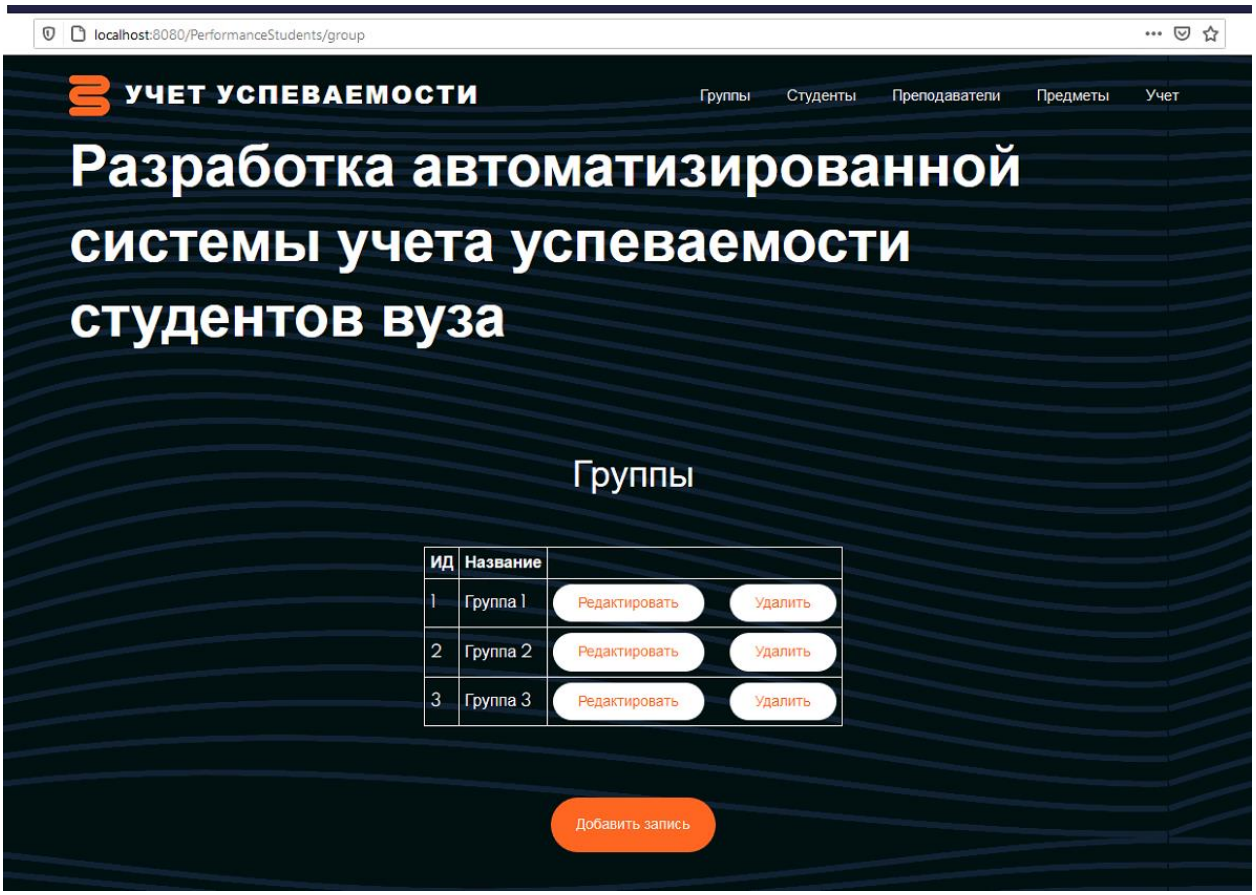

Рисунок 8.2 – Группы

Для того, чтобы добавить новую группу, необходимо нажать соответствующую кнопку «Добавить запись». Для редактирования – кнопку «Редактировать» напротив нужного элемента.

Вид страницы добавления показан ниже.

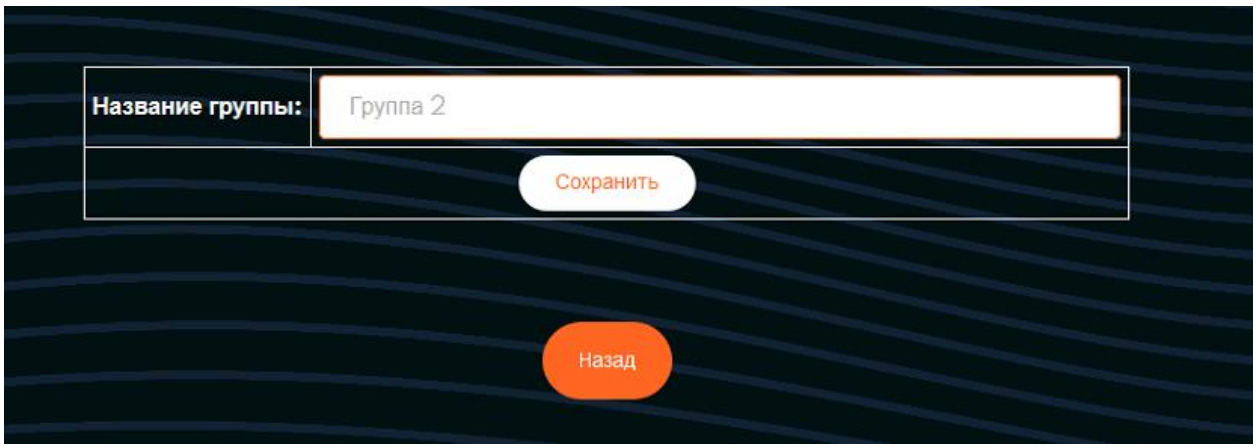

Рисунок 8.3 – Добавление группы

Для добавления информации нужно заполнить поля и нажать кнопку «Сохранить». Если вдруг вы передумали, достаточно нажать кнопку «Назад», и вы вернетесь на предыдущую страницу.

Такой же принцип работы и с другими страницами.

К примеру, вид страницы работы с студентами показан ниже.

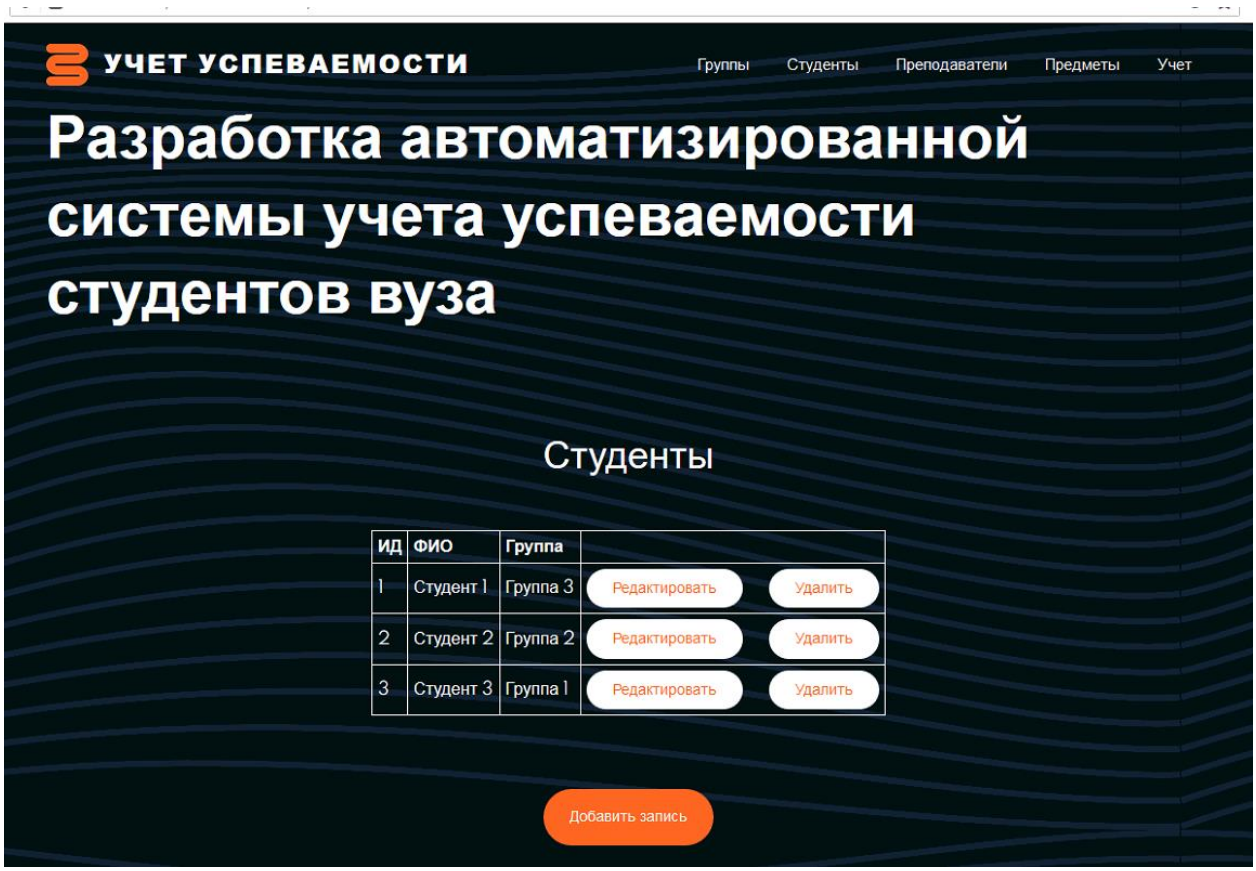

Рисунок 8.4 – Работа с студентами

При добавлении студента нужно из списка всех групп выбрать ту, к которой будет относиться студент.

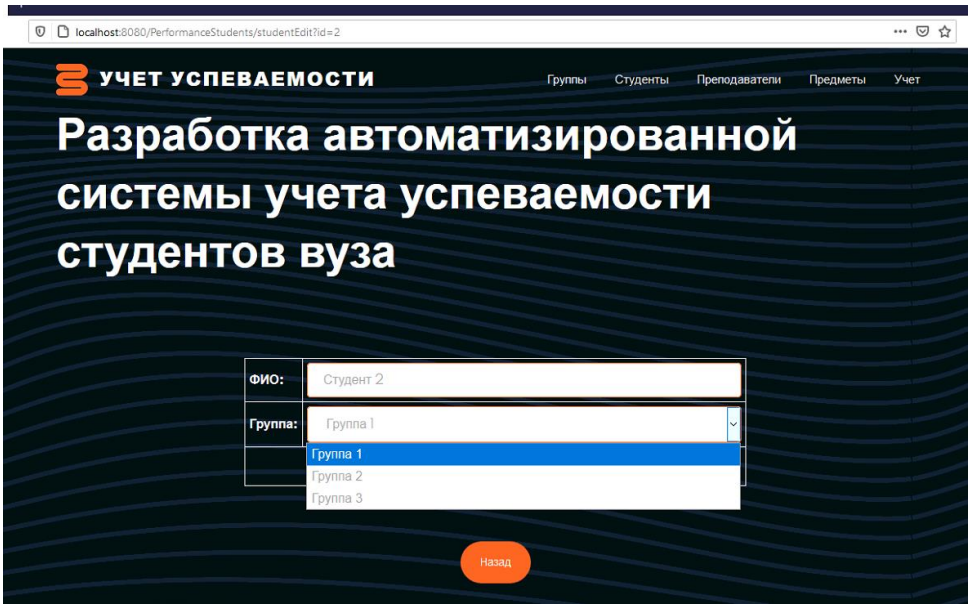

Рисунок 8.5 – Добавление студента

Страница отображений оценок показана ниже.

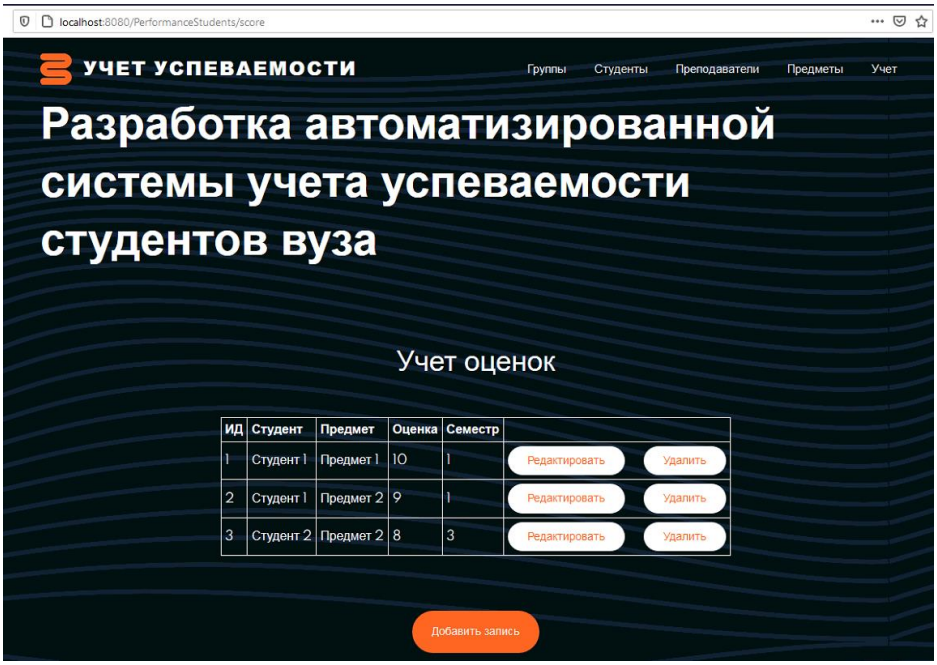

Рисунок 8.6 – Работа с оценками

Данное приложение является лёгким в использовании, понятным и не требует дополнительных затрат для осваивания .

9.РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ РАЗРАБОТАННОЙ СИСТЕМЫ И ОЦЕНКА ВЫПОЛНЕНИЯ ЗАДАЧ.

Все основные моменты исполнения данного проекта были показаны в разделе Руководство пользователя.

Во всей программе присутствуют проверки на ввод корректных данных.

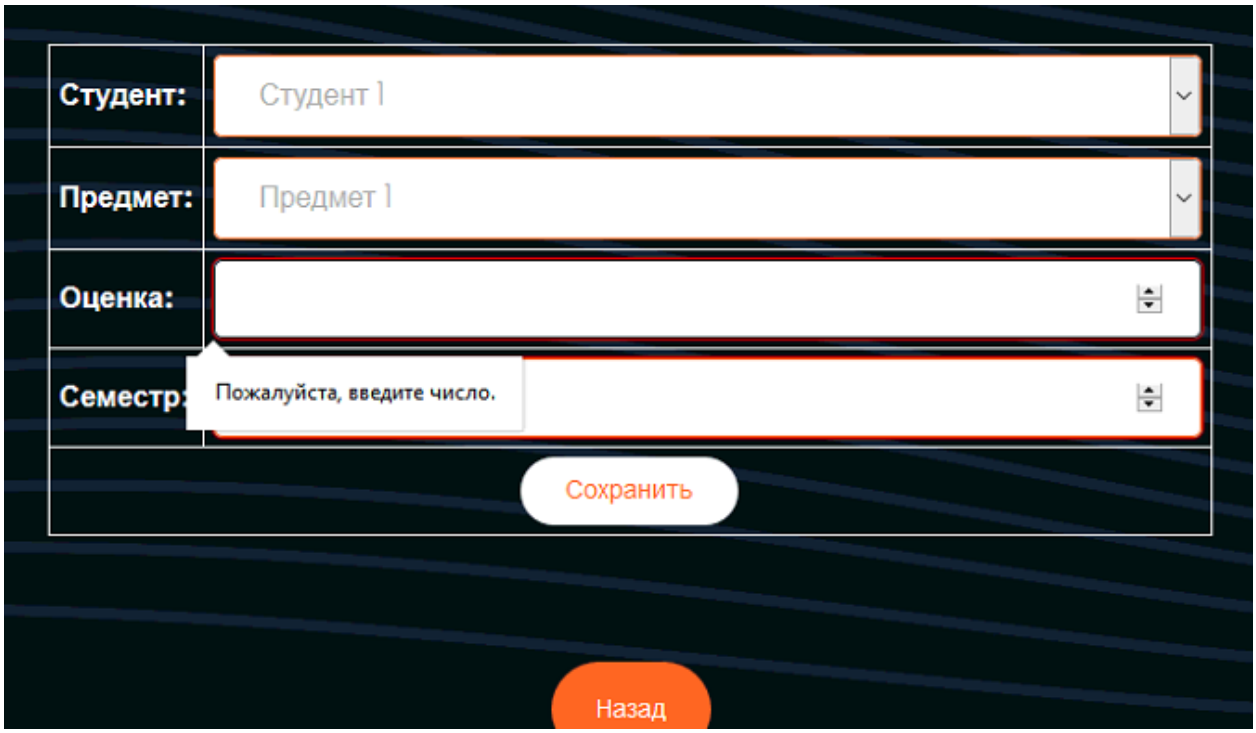

Рисунок 9.1 – Проверка на заполнение полей

Тест-кейс – это профессиональная документация тестировщика, последовательность действий, направленная на проверку какого-либо функционала, описывающая как прийти к фактическому результату [10].

Тестирование в виде тест-кейсов показано в таблице 1.

Таблица 1 – Тестирование приложения в виде тест-кейсов

Req. No.	Test Case Description	Expected Results
1	2	3
1	1. Запуск	После запуска приложения и

	приложения	открытия браузера по адресу http://localhost:8080/ PerformanceStudents/ отображается главная страница
2	1. Перейти по адресу / PerformanceStudents/group 2. Нажать кнопку «Добавить запись» 3. Заполнить поля 4. Нажать кнопку «Сохранить»	Произойдет переадресация на страницу со списком групп, на котором будет отображена только что добавленная группа.
3	1. Перейти по адресу / PerformanceStudents /group 2. Нажать кнопку «Добавить запись» 3. Ничего не заполнять 4. Нажать кнопку «Сохранить»	Переадресации не произойдет. Пользователю будет показано уведомлении о необходимом заполнении всех полей.
4	1. Перейти по адресу / PerformanceStudents /group 2. Удалить все группы 3. Перейти по адресу / PerformanceStudents /score 4. Нажать кнопку «Добавить запись» 5. Заполнить поля 6. Нажать кнопку «Сохранить»	Переадресации не произойдет. Пользователю покажется сообщении о необходимости заполнить все поля, даже если выпадающие списки пустые

Описанные выше тесты работают без проблем.

ВЫВОДЫ И ЗАКЛЮЧЕНИЯ.

В данной курсовой работе было разработано веб-приложение с использованием базы данных и доступом к ней через JDBC и драйвером поставщика базы данных на тему «Разработка автоматизированной системы учета успеваемости студентов вуза».

Были соблюдены все требования, предъявляемые к данному курсовому проекту, а также все задачи, которые были поставлены для разработки данной системы.

Была создана информационная и функциональная модель, приведённые выше, которые в полной мере позволяют оценить данное приложение и понять, как с точки зрения раработчика, так и с точки зрения пользователя.

Одним из достоинств разработанной системы является стильный и удобный для пользователя интерфейс. Таким образом, любой пользователь (работник ресторана, просто пользователь), когда-либо работавший в Internet, без труда сможет работать и с этой системой.

Для создания данного проекта был использован язык программирования Java, сервлеты, html и css.

Также использовался шаблон MVC, который позволил структурировать архитектуру приложения и сделать его удобным для расширяемости.

ЛИТЕРАТУРА

1. Структура высшего учебного заведения [Электронный ресурс]. – Режим доступа: https://studopedia.su/13_130056_struktura-visshego-uchebnogo-zavedeniya-i-urovni-upravleniya.html. – Дата доступа: 19.01.2021.

2. Основные методологии обследования организаций. Стандарт IDEF0 [Электронный ресурс]. – Режим доступа: https://www.cfin.ru/vernikov/idef/idef0.shtml. – Дата доступа: 19.01.2021.

3. UML [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/UML. – Дата доступа: 19.01.2021.

4. Use Case [Электронный ресурс]. – Режим доступа: https://systems.education/use-case. – Дата доступа: 19.01.2021.

5. Диаграмма состояний [Электронный ресурс]. – Режим доступа: http://caseclub.ru/articles/rose2.html?next=0. – Дата доступа: 19.01.2021.

6. Диаграмма последовательности [Электронный ресурс]. – Режим доступа: https://flexberry.github.io/ru/fd_sequence-diagram.html. – Дата доступа: 19.01.2021.

7. Диаграмма развертывания [Электронный ресурс]. – Режим доступа: http://khpi-iip.mipk.kharkiv.edu/library/case/leon/gl11/gl11.html. – Дата доступа: 19.01.2021.

8. Проектирование логической и физической моделей базы данных [Электронный ресурс]. – Режим доступа: https://studfile.net/preview/7227021/page:5/. – Дата доступа: 19.01.2021.

9. MVC [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Model-View-Controller. – Дата доступа: 19.01.2021.

10. Тест-кейсы [Электронный ресурс]. – Режим доступа: https://training.qatestlab.com/blog/technical-articles/test-case-topic/. – Дата доступа: 19.01.2021.