**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc481447850)

[1 ОПИСАНИЕ ПРОЦЕССА УЧЕТА УСПЕВАЕМОСТИ ШКОЛЬНИКОВ 7](#_Toc481447851)

[2 ПОСТАНОВКА ЗАДАЧИ И ОБЗОР МЕТОДОВ ЕЁ РЕШЕНИЯ. 13](#_Toc481447852)

[3 ФУНКЦИОНАЛЬНАЯ МОДЕЛЬ 16](#_Toc481447853)

[4 ОПИСАНИЕ ИНФОРМАЦИОННОЙ МОДЕЛИ СИСТЕМЫ 19](#_Toc481447854)

[5 МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ 21](#_Toc481447855)

[6 ОПИСАНИЕ АЛГОРИТМОВ РЕАЛИЗУЮЩИХ БИЗНЕС-ЛОГИКУ СЕРВЕРНОЙ ЧАСТИ ПРОЕКТИРУЕМОЙ СИСТЕМЫ И ОПИСАНИЕ ПРИМЕНЕНИЯ ПАТТЕРНОВ ПРЕКТИРОВАНИЯ 24](#_Toc481447856)

[7 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 29](#_Toc481447857)

[8 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ РАЗРАБОТАННОЙ СИСТЕМЫ 38](#_Toc481447858)

[ЗАКЛЮЧЕНИЕ 40](#_Toc481447859)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 42](#_Toc481447860)

[ПРИЛОЖЕНИЕ А 43](#_Toc481447861)

[(обязательное) 43](#_Toc481447862)

# **ВВЕДЕНИЕ**

В современном обществе все большее значение приобретает умение учиться, формирующееся еще в детстве. Важным показателем этого умения является школьная успеваемость, которая у определенного количества детей не достигает минимального уровня. Практика показывает, что слабая успеваемость - многогранное явление, имеющее сложную структуру и происхождение.

Несмотря на накопленный школой опыт, а в ряде случаев и ликвидации отставания детей в учебе, эта проблема остается актуальной в педагогике. Теоретический анализ состояния проблемы показал, что существуют различные виды слабой успеваемости. Их выделяют по тем или иным параметрам, из которых основными являются глубина и стойкость. Различают в зависимости от стойкости кратковременную, относительно стойкую и длительную, а в зависимости от глубины неглубокую, среднюю и глубокую слабую успеваемость.

Анализ психолого-педагогической литературы показал, что, несмотря на большое число исследований, некоторые ключевые вопросы обучения слабоуспевающих учащихся остаются в тени и требуют более полного изучения. Недостаточно раскрыты формы и методы, приводящие слабоуспевающих детей к достижению успехов в условиях совместного обучения с более сильными учащимися. Не уделяется внимание вопросам профилактики слабой успеваемости, требует более глубокой проработки проблема факторов, определяющих успешность обучения.

В настоящий момент по всему миру во всех школах стоит вопрос повышения уровня знаний школьников, а также серьезной проблемой обучения становится оптимизация учебного процесса.

Чтобы добиться этого, необходимо внедрение механизма, способного решить эту задачу. В роли этого механизма может выступать автоматизированная информационная система учета и контроля успеваемости школьников. Чтобы лучше понять необходимость таких систем, необходимо посмотреть на это с точки зрения концепции научных революций.

Как и в науке, в которой на смену одной парадигмы приходит другая, в связи с тем, что нормальная наука уже не справляется с задачами, решаемыми в рамках старой парадигмы, так и педагогике приходится изменять некоторые правила. Иначе, если все время решать “задачи-головоломки”, как это делает нормальная наука в рамках одной парадигмы, можно прийти в тупик. Также дела обстоят и в учебно-воспитательной программе школы, если все время пытаться решать проблемы, применяя старые методы в учебно-воспитательной работе.

Ранее учёт успеваемости школьников вёлся мануально. Этот вариант является неэффективным и трудоемким. С распространением различных информационных технологий и способов взаимодействия с ними, человечество смогло автоматизировать различные процессы в жизни общества. Внедрение системы учета успеваемости школьников позволит повысить производительность учителей школ, освобождая их от выполнения простых действий и предоставляя им больше времени уделять ученикам.

Целью данной курсовой работы является сокращение временных затрат учителей школ за счёт автоматизации процесса учёта успеваемости школьников.

Для достижения поставленной цели необходимо решить следующие задачи:

* изучить предметную область;
* проанализировать логическую и физическую модель представления данных;
* создать базу данных;
* реализовать серверную часть, которая позволит работать с базой данных и выполнить ряд функций, поставленных целью разработки курсового проекта;
* разработать программное обеспечение;
* протестировать программный продукт.

# **1** **ОПИСАНИЕ ПРОЦЕССА УЧЕТА УСПЕВАЕМОСТИ ШКОЛЬНИКОВ**

Первоначально греческое σχολή означало «досуг, свободное времяпровождение», затем стало применяться как «занятия на досуге» (напр. Платоном, он собирал людей в свободное время и передавал им свои опыт и знания), затем – «учебные занятия», «беседа философов» (напр. у Плутарха). В русский язык слово пришло не напрямую, а через польск. szkoła от лат. schola.

Школы служат для получения:

* начального образования (начальная школа) (в Беларуси с 1 по 4 классы)
* основного базового образования (средняя школа) (в Беларуси с 5 по 9 классы)
* среднего общего образования (средняя школа) (в Беларуси с 10 по 11 классы)

Школьное время, в общем и целом распределяется на три этапа: урок, перемена и «продлёнка». Урок и перемена чередуются между собой некоторое количество раз, а «продлёнка» завершает учебный день.

Урок – занятие школьников по предметам.

Перемена – отдых между уроками.

«Продлёнка» (полн. группа продлённого дня, ГПД) – нахождение ученика в школе после окончания уроков (обеспечивается питание, возможность подготовки уроков), если нет возможности нахождения ребёнка дома.

Элективы (элективные курсы), факультативы (как правило, проводятся после основных занятий).

В системе общего образования могут также быть специализированные средние школы или отдельные классы (профильные): с углублённым изучением ряда предметов – иностранного языка, физико-математические, химические, биологические и т. п. От обычных отличаются дополнительной учебной нагрузкой по предметам специализации.

В конце последнего класса, а также в конце 9 класса, ученики сдают экзамены по части предметов. По результатам этих экзаменов и годовых оценок выставляются оценки в аттестат. По тем предметам, по которым нет экзаменов, в аттестат ставится годовая оценка.

Обязательно обучение до 9 класса. Обучение в 10 и 11 классе не является обязательным для всех детей. После окончания 11-го класса ученик получает свидетельство об общем среднем образовании. После успешного окончания 9-го класса выпускник получает аттестат об общем базовом образовании. Для поступления в высшее учебное заведение обычно требуется полное среднее образование: аттестат о среднем общем образовании, либо диплом профессионально-технического училища, либо диплом техникума, а также результаты ЦТ.

В отечественном обоаховании, поъоже, может наступить новая эра. Правительство всерьез задумалось над возможностью заменить традиционные бумажные учебники на интерактивные медиаприложения. Носителями электронных учебных пособий должны стать планшеты – самый перспективный на сегодняшний день вариант персональных компьютеров.

Стоит отметить, что в России уже протестировали данную методику. «Весной 2013 года в 75 российских школах был проведен эксперимент длительностью в 60 дней, на основании которого учителя смогли сделать вывод о положительной динамике использования электронных учебников в образовательном процессе. Однако было отмечено, что проект нуждается в доработке. До 2015 года было проведено ещё несколько стадий тестирования, по результатам которых согласно принятому закону с 1 января 2015 года все российские школы обязаны перейти на те учебники, для которых будет выпущена электронная версия. В настоящее время проект является пилотным – по данным на май 2015 года в нём участвует 2500 учащихся и 60 преподавателей. Электронный учебник как часть «Электронной Образовательной Среды» – это конструктор, используя который учитель собирает в единое целое текст учебника, интерактивное содержание и дополнительные материалы. Как и бумажный, он является индивидуальным инструментом каждого ученика. Посредством интернета, электронный учебник автоматически обновляется, а на сервер могут отправляться результаты тестов и контрольных работ[1]».

Современный этап развития образования характеризуется широким внедрением в учебный процесс компьютерных технологий. Они позволяют выйти на новый уровень обучения, открывают ранее недоступные возможности как для учителя, так и для учащегося. Применение информационных технологий в процессе обучения в начальной школе дает возможность активизировать познавательную и мыслительную деятельность учащихся.

Информационные технологии могут не только изменить формы и методы учебной работы, но и существенным образом трансформировать и обогатить образовательные парадигмы. Изменению подвергаются даже такие фундаментальные навыки, прививаемые начальной школой, как умение читать и писать[2].

Информационные технологии находят свое применение в различных предметных областях на всех возрастных уровнях, помогая лучшему усвоению как отдельных тем, так и изучаемых дисциплин в целом. Персональный компьютер помогает учащимся реализовывать свои знания в практической деятельности, творчески решать учебные задания. Сегодня целесообразность применения компьютерных технологий в среднем и старшем звеньях школьной системы очевидна. Навыки пользователя ПК и рынок обучающих программ, мультимедийных справочников, Интернет открывают большие возможности самообразования. Облегчение доступности базы знаний за счет применения постоянно совершенствующихся средств мультимедиа (цветовое и звуковое оформление, анимация) и методик дистанционного обучения (обучающе-контролирующие программы, диалоговый режим, различные уровни сложности) изменяет ролевую функцию учителя в образовательном процессе. Роль учителя как носителя и распространителя информации отходит на второй план, а доминирующей становится роль интерпретатора знаний. Главным становится научить пользоваться новыми знаниями, правильно внедрить их в интеллектуальную среду обучающихся, акцентировать тематические и межпредметные связи, сформировать устойчивые навыки практического применения знаний, развить на их основе мыслительные и творческие способности учеников, обеспечить выход на более высокий уровень образовательного процесса. [3].

Новая грамотность предполагает овладение умением ориентироваться в информационных потоках, в среде мультимедиа, создавать гипермедиа объекты. Современный человек еще в школе должен научиться читать и писать применительно к мировому информационному пространству.

В некоторых средних общеобразовательных школах уже сегодня создаются свои сайты, это становится для школы важным и престижным делом. Однако, это, к сожалению, во многом зависит от финансовых возможностей образовательного учреждения. Образовательными стандартами и программами это пока не предусматривается[4].

Однако, в новый век тысячелетие образование вошло более свободно, проявляя инициативу и пытающееся самостоятельно формировать свою образовательную политику, искать новые формы организации учебного процесса, оказания дополнительных образовательных услуг и привлечения внебюджетных средств финансирования. Образовательные учреждения, энергично внедряющие новые информационные технологии, демонстрируют желание обеспечить современный уровень преподавания и высокое качество обучения, привлекают внимание родителей.

Сегодня говорят об изменении содержания образования, о необходимости овладения учащимися информационной культурой – одним из слагаемых общей культуры, понимаемой как высшее проявление образованности, включая личностные качества человека и его профессиональную компетентность[5].

Исследования в области использования информационных образовательных технологий в профессиональное образование, ведутся достаточно давно. За это время в учебных заведениях США, Франции, Японии, России и ряда других стран было разработано множество компьютерных систем учебного назначения. Однако, сфера применения таких систем гораздо шире. Это крупные промышленные предприятия, военные и гражданские организации, ведущие самостоятельную подготовку и переподготовку кадров. Кроме того, становится уже стандартом снабжать новые сложные машины и технологии компьютерными обучающими системами, ускоряющими процесс их освоения и внедрения. За рубежом разработку компьютерного продукта учебного назначения (методических и программно-информационных средств) считают необходимым делом в силу его высокой наукоемкости и необходимости совместной работы высококвалифицированных специалистов: психологов, преподавателей-предметников, компьютерных дизайнеров, программистов. Многие крупные зарубежные фирмы финансируют проекты создания компьютерных учебных систем в образовательных учреждениях и ведут собственные разработки в данной области.

Организация информационных процессов в рамках информационных образовательных технологий предполагает выделение таких базовых процессов, как передача, обработка, организация хранения и накопления данных, формализация и автоматизация знаний[6].

Совершенствование методов решения функциональных задач и способов организации информационных процессов приводит к совершенно новым информационным технологиям, среди которых применительно к обучению выделяют следующие:

* компьютерные обучающие программы, включающие в себя электронные учебники, тренажеры, тьюторы, лабораторные практикумы, тестовые системы;
* обучающие системы на базе мультимедиа-технологий, построенные с использованием персональных компьютеров, видеотехники, накопителей на оптических дисках;
* интеллектуальные и обучающие экспертные системы, используемые в различных предметных областях;
* распределенные базы данных по отраслям знаний;
* средства телекоммуникации, включающие в себя электронную почту, телеконференции, локальные и региональные сети связи, сети обмена данными и т.д.;
* электронные библиотеки, распределенные и централизованные издательские системы.

Компьютер как средство повышения эффективности педагогической деятельности - компонент образовательной системы, который не только способен внести коренные преобразования в само понимание категории “средство” применительно к процессу образования, но и существенно повлиять на все остальные компоненты той или иной локальной образовательной системы: цели, содержание, методы и организационные формы обучения, воспитания и развития обучающихся в учебных заведениях любого уровня и профиля.

Компьютер – средство повышения эффективности научно-исследовательской деятельности в образовании. Современные научные исследования, тем более исследования междисциплинарные, комплексные, уже не могут быть успешными без всестороннего информационного обеспечения. Такое обеспечение предполагает поиск источников наиболее актуальной информации, соответствующей современному уровню содержания образования, отбор и избирательную оценку этой информации, ее хранение, обеспечивающее должный уровень классификации информации и свободу доступа к ней со стороны потенциальных потребителей, наконец оперативное представление необходимой информации пользователю по его запросам.

Компьютер и информатика – компонент системы образовательно-педагогического управления. Это направление информатизации связано с процессом принятия управленческих решений на всех уровнях образовательной деятельности – от повседневной работы по управлению учебным заведением до управления всей отраслью на федеральном и региональном уровнях. Для принятия оптимальных управленческих решений необходима самая разнообразная информация как фонового характера о тенденциях развития внешней социально-экономической и социокультурной среды, так и собственно образовательного характера.

Компьютер – сложное техническое устройство. Его образовательно-педагогические возможности во многом предопределяются техническими факторами, теми реальными достижениями в научно-технической сфере, которые придают компьютеру определенные свойства и позволяют ему выполнять с должным эффектом заданные функции, в том числе и функции, ориентированные на запросы системы образования.

Информационные образовательные технологии, на наш взгляд, действительно являются эффективными, способствуют реализации известных дидактических принципов организации учебного процесса, наполняют деятельность учителя, преподавателя принципиально новым содержанием, позволяя им сосредоточиваться на своих главных – обучающей, воспитательной и развивающей – функциях.

В процессе обучения детей с помощью информационных технологий, они учатся работать с текстом, создавать графические объекты и базы данных, использовать электронные таблицы. Ребенок узнает новые способы сбора информации и учится пользоваться ими, расширяется его кругозор. При использовании информационных образовательных технологий на занятиях повышается мотивация учения и стимулируется познавательный интерес учащихся, возрастает эффективность самостоятельной работы. Компьютер вместе с информационными технологиями открывает принципиально новые возможности в области образования, в учебной деятельности и творчестве учащегося. Возникает такая ситуация, когда информационные технологии становятся и основными инструментами дальнейшей профессиональной деятельности человека[7].

# **2 ПОСТАНОВКА ЗАДАЧИ И ОБЗОР МЕТОДОВ ЕЁ РЕШЕНИЯ.**

Задачей курсового проекта является создание клиент-серверного программного приложения, позволяющего оптимизировать процесс учёта успеваемости школьников.

Программный код должен быть написан на объектно-ориентированном языке Java. Java представляет собой язык программи-рования и платформу вычислений, которая была впервые выпущена Sun Microsystems в 1995 г. Существует множество приложений и веб-сайтов, которые не работают при отсутствии установленной Java, и с каждым днем число таких веб-сайтов и приложений увеличивается. Java отличается быстротой, высоким уровнем защиты и надежностью. От портативных компьютеров до центров данных, от игровых консолей до суперкомпью-теров, используемых для научных разработок, от сотовых телефонов до сети Интернет[8].

Как правило компьютеры и программы, входящие в состав инфор-мационной системы, не являются равноправными. Некоторые из них вла-деют ресурсами (файловая система, процессор, принтер, база данных и т.д.), другие имеют возможность обращаться к этим ресурсам. Компьютер (или программу), управляющий ресурсом, называют сервером этого ре-сурса (файл-сервер, сервер базы данных, вычислительный сервер). Клиент и сервер какого-либо ресурса могут находится как в рамках одной вычис-лительной системы, так и на различных компьютерах, связанных сетью.

Основной принцип технологии "клиент-сервер" заключается в разде-лении функций приложения на три группы:

* ввод и отображение данных (взаимодействие с пользователем);
* прикладные функции, характерные для данной предметной области;
* функции управления ресурсами (файловой системой, базой данных и т.д.).

Поэтому, в любом приложении выделяются следующие компоненты:

* компонент представления данных;
* прикладной компонент;
* компонент управления ресурсом.

Таким образом, клиент-сервер (англ. сlient-server) – вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Фактически клиент и сервер – это программное обеспечение. Программы-сервера, ожидают от клиентских программ запросы и предоставляют им свои ресурсы в виде данных (например, загрузка файлов посредством HTTP, FTP, BitTorrent, потоковое мультимедиа или работа с базами данных) или сервисных функций (например, работа с электронной почтой, общение посредством систем мгновенного обмена сообщениями, просмотр web-страниц во всемирной паутине) [9].

Преимущества клиент-серверных систем:

* клиент-серверный подход модульный, причем серверные программные компоненты компактны и автономны;
* сбой сервера не повлияет на остальные компоненты операционной системы;
* автономность компонентов делает возможным их выполнение на нескольких процессорах на одном компьютере (симметричная многопроцессорная обработка) или на нескольких компьютерах сети (распределенные вычисления);
* обязанность клиента, как правило, – предоставлять пользовательские сервисы и, прежде всего, пользовательский интерфейс, то есть средства для приема, отображения и редактирования данных, введенных пользователем, которые служат основой для запроса серверу [10].

Cвязь между сервером и клиентом в приложении осуществляется при помощи TCP/IP. Также в реализации системы были использованы следующие техники:

* разработаны и использованы собственной иерархии, расширение базовых классов, предоставляемых JDK;
* реализованы два паттерна проектирования;
* использовано сокрытие данных (инкапсуляция), перегрузка методов, переопределение методов, сериализация, абстрактные типы данных (интерфейсы, абстрактные классы), статические методы, обработка исключительных ситуаций.

Клиент-сервер позволит удаленно работать с данными, что очень важно в современных условиях. Администратор будет осуществлять авторизацию пользователей. Это позволит защитить данные от несанкционированного доступа. Требуется предусмотреть возможность просмотра, редактирования, удаления данных, создания новых записей. Сервер должен обладать возможностью параллельной обработки запросов, так как учётом успеваемости может заниматься не один, а несколько учителей. Связь с базой данных должна осуществляться с помощью SQL-запросов через сервер.

В качестве основной среды разработки в проекте используется NetBeans IDE 8.1. В качестве СУБД в курсовом проекте используется MySQL Workbench. Для создания UML-диаграмм в работе было использовано средство Enterprise Architect, которое позволяет осуществлять создание диаграмм вариантов использования, состояний, последовательность с помощью просто моделирования. Для выполнения UML-моделей в стандарте IDEF0 использовалось CASE-средство CA AllFusion Process Modeler r7 (BPwin). Для информационного моделирования применялось средство CA AllFusion ERwin Data Modeler r7 (ERwin).

# **3 ФУНКЦИОНАЛЬНАЯ МОДЕЛЬ**

После проведенного анализа предметной области представим функциональную модель процесса учёта успеваемости школьников. Входными данными запросы, которые поступают в школу из министерств.

Ресурсами, необходимыми для выполнения процесса являются:

* учитель;
* секретарь.

Основным процессом данного курсового проекта является учёт успеваемости школьников. В соответствии с этим на рисунке 3.1 представлена контекстная диаграмма функциональной модели.

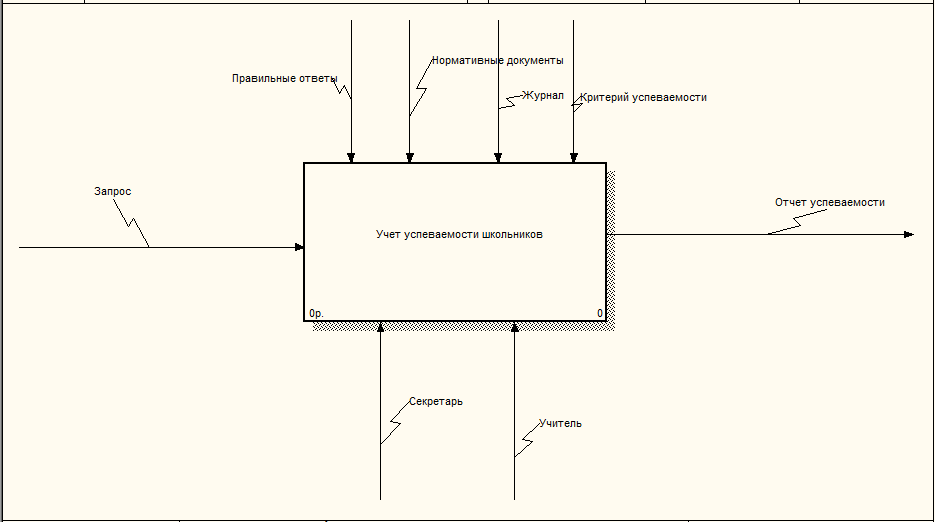


Рисунок 3.1 – Контекстная диаграмма верхнего уровня

Декомпозиция контекстной диаграммы, показанная на рисунке 3.2, описывает разделение процесса учёта успеваемости школьников на следующие этапы:

* провести самостоятельные работы;
* проверить самостоятельные работы;
* выставить оценки;
* составить отчет.

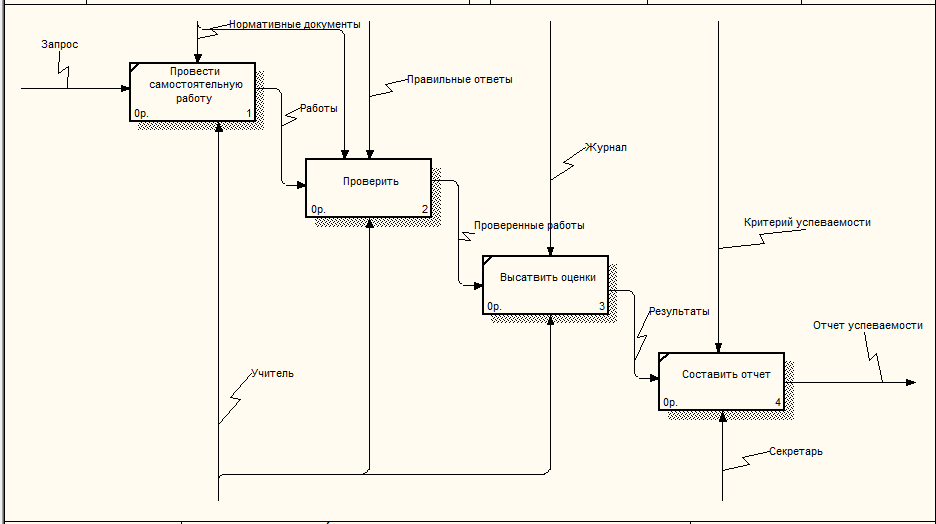


Рисунок 3.2 – Декомпозиция диаграммы верхнего уровня

Можно провести декомпозицию блока «Проверить самостоятельные работы» (см. рисунок 3.3). Данный блок можно разбить следующим образом:

сопоставить с правильными ответами;

подсчитать правидьные ответы;

определение количества “положительных» работ.

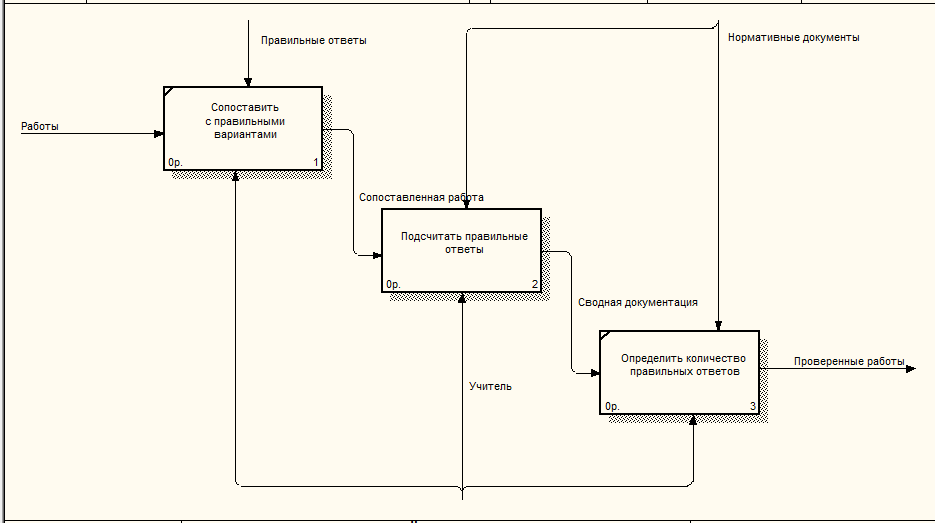


Рисунок 3.3 – Декомпозиция блока «Проверить самостоятельные работы»

Следующим блоком для декомпозиции является «Подсчитать правильные ответы». Данный блок состоит из:

* сравнить все ответы;
* отсев неудовлетворительных работ;
* выставление удовлетворительных оценок.

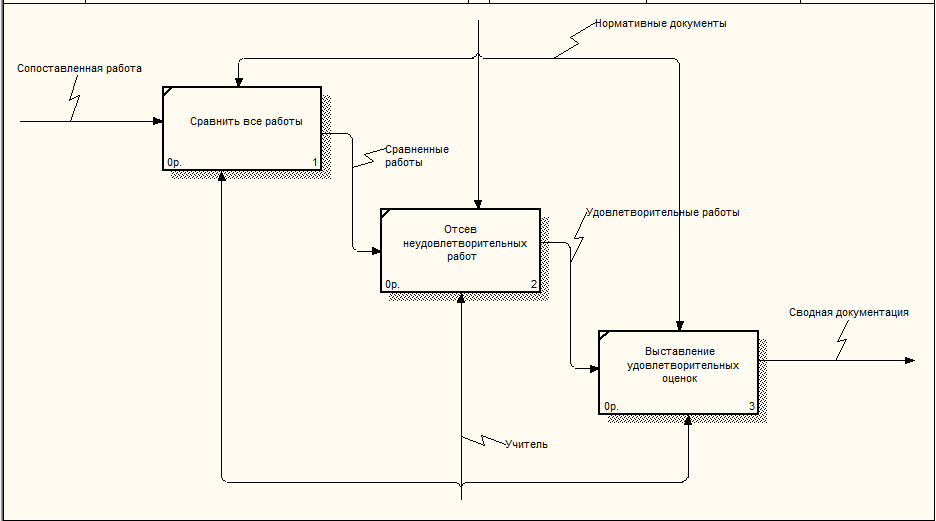


Рисунок 3.4 – Декомпозиция блока «Подсчитать правильные ответы»

С помощью представленной IDEF-0 диаграммы можно более подробно и тщательно рассмотреть процесс оформления выдачи книги и его подпроцессы.

# **4 ОПИСАНИЕ ИНФОРМАЦИОННОЙ МОДЕЛИ СИСТЕМЫ**

В результате информационного моделирования мною были выделены следующие сущности:

* ученики;
* пользователи;
* администраторы;
* заявления.

Для хранения личной информации об учениках необходимо использовать сущность «Ученики». В ней учителя школ будут заносить все данные об учениках имя, фамилия, отчество, класс, отметки по предметам.

Сведения об учениках должны вноситься администратором (учителем), в следствии чего введем сущность «Администратор». Данная таблица будет хранить в себе идентификационный номер, логин и пароль. Логично, что связь между сущностями будет многие-ко-многим.

При входе в систему каждый пользователь должен вводить пользовательские данные, которые необходимо где-то хранить, с этой целью создадим сущность «Пользователи». Также сущность «Пользователи» необходима для авторизации учеников в системе. В таком случае эта таблица будет содержать идентификационный номер приказа, пароль и логин. Связь в этом случае будет реализована с таблицей «Заявления», один-к-одному.

Таблица «Заявления» необходима для учета, а также статистики выбора учениками факультативных курсов. Она будет содержать такие поля как идентификационный номер заявления, идентификационный номер ученика, логин пользователя, фамилию ученика.

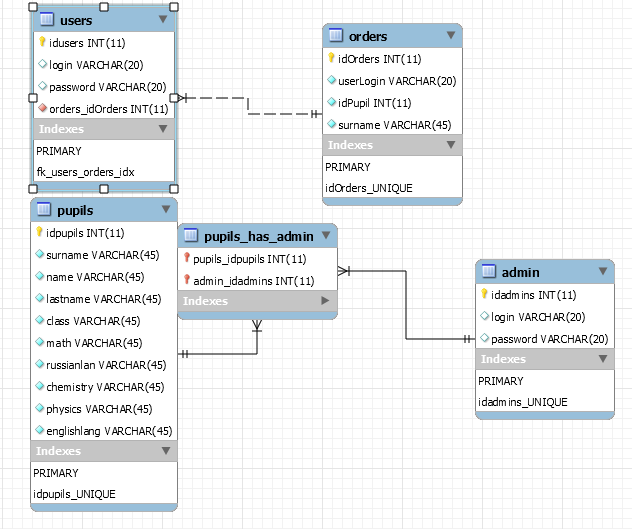


Рисунок 4.1 – Информационная модель базы данных

С учетом обозначенного взаимодействия сущностей смоделируем их взаимодействие в формате IDEF1.X и приведем эту модель к третьей нормальной форме. В результате последовательного приведения модели к третьей нормальной форме, получаем информационную модель соответствующую условиям третьей нормальной формы – не ключевой атрибут сущности функционально зависит только от всего первичного ключа и ни от чего другого.

# **5 МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ**

Для более подробного изучения и рассмотрения системы программы необходимо рассмотреть такие модели как:

* диаграмма состояний;
* диаграмма последовательностей;
* диаграмма классов;
* диаграмма компонентов;
* диаграмма развертывания.

Система управления работой библиотеки будет реализована на операционной системе Windows 8.1. На диаграмме развертывания (рисунок 5.1), показана конфигурация обрабатывающих узлов, на которых выполняется система, и компонентов, размещенных в этих узлах.

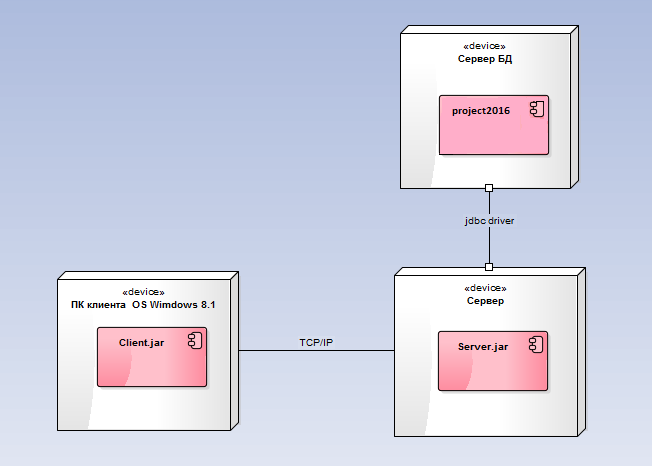
****

Рисунок 5.1 – Диаграмма развёртывания

Диаграмма состояний представлена ниже (рисунок 5.2).

****

Рисунок 5.2 – Диаграмма состояний при добавлении данных об ученике

Для входа в систему, ученику требуется ввести логин и пароль.После выполнения необходимих действий, клиент отправляет запрос серверу, а сервер уже сверяет введенные данные с БД. В БД формируется ответ серверу, который формирует из него ответ для клиента. Клиентская часть предоставляет результат авторизации ученику (рис. 5.3).

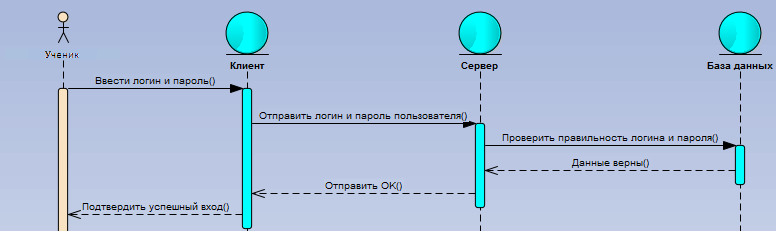
****

Рисунок 5.3 – Диаграмма последовательности процесса авторизации ученика

Полный проект программной системы представляет собой совокупность моделей разных уровней, которые должны быть согласованы между собой. Диаграмма компонентов отражает общие зависимости между компонентами. Она позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный и исполняемый код [5]. Диаграмма компонентов для данного проекта представлена ниже (рис. 5.4). В разделе Client находятся компоненты клиентской части систем. То, как будет выглядеть то или иное окно, определяется именно здесь. В разделе Server находятся компоненты серверной части системы. Здесь осуществляется запуск сервера, создаются сокеты и потоки ввода/вывода.

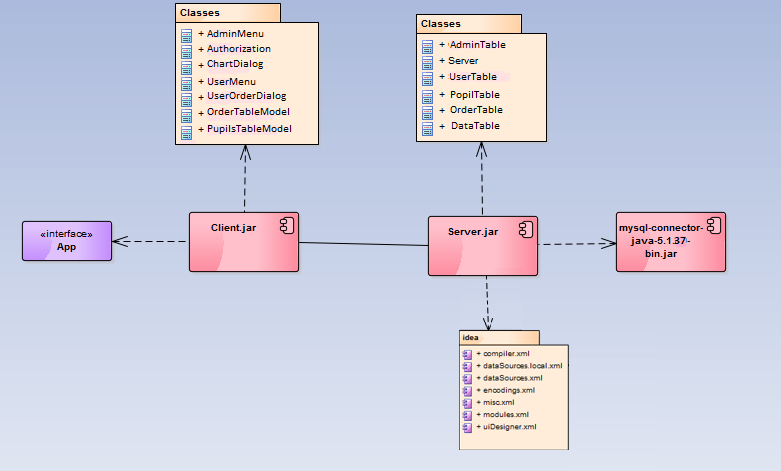
****

Рисунок 5.4 – Диаграмма компонентов

# **6 ОПИСАНИЕ АЛГОРИТМОВ РЕАЛИЗУЮЩИХ БИЗНЕС-ЛОГИКУ СЕРВЕРНОЙ ЧАСТИ ПРОЕКТИРУЕМОЙ СИСТЕМЫ И ОПИСАНИЕ ПРИМЕНЕНИЯ ПАТТЕРНОВ ПРЕКТИРОВАНИЯ**

Часто в новом программном проекте не удается повторно использовать уже существующий код. Например, имеющиеся классы могут обладать нужной функциональностью, но иметь при этом несовместимые интерфейсы. В таких случаях следует использовать паттерн Adapter (адаптер).

Паттерн Adapter, представляющий собой программную обертку над существующими классами, преобразует их интерфейсы к виду, пригодному для последующего использования.

Пусть класс, интерфейс которого нужно адаптировать к нужному виду, имеет имя Adaptee. Для решения задачи преобразования его интерфейса паттерн Adapter вводит следующую иерархию классов:

* Виртуальный базовый класс Target. Здесь объявляется пользовательский интерфейс подходящего вида. Только этот интерфейс доступен для пользователя.
* Производный класс Adapter, реализующий интерфейс Target. В этом классе также имеется указатель или ссылка на экземпляр Adaptee. Паттерн Adapter использует этот указатель для перенаправления клиентских вызовов в Adaptee. Так как интерфейсы Adaptee и Target несовместимы между собой, то эти вызовы обычно требуют преобразования.Паттерн Адаптер можно увидеть на рисунке 6.1.

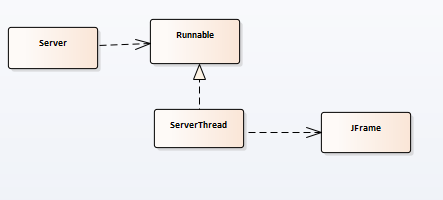


Рисунок 6.1 –Паттерн Адаптер

Достоинства паттерна Adapter:

* Паттерн Adapter позволяет повторно использовать уже имеющийся код, адаптируя его несовместимый интерфейс к виду, пригодному для использования.

Недостатки паттерна Adapter:

* Задача преобразования интерфейсов может оказаться непростой в случае, если клиентские вызовы и (или) передаваемые параметры не имеют функционального соответствия в адаптируемом объекте [9].

Медиатор выступает в качестве посредника в общении между различными модулями, инкапсулируя их взаимодействие. Кроме того, этот шаблон проектирования, предотвращая прямое взаимодействие различных компонентов системы, способствует ослаблению связей в коде.

Назначение паттерна Mediator:

* Паттерн Mediator определяет объект, инкапсулирующий взаимодействие множества объектов. Mediator делает систему слабо связанной, избавляя объекты от необходимости ссылаться друг на друга, что позволяет изменять взаимодействие между ними независимо.
* Паттерн Mediator вводит посредника для развязывания множества взаимодействующих объектов.
* Заменяет взаимодействие "все со всеми" взаимодействием "один со всеми".

Достоинства паттерна Mediator:

* Уменьшает связывание модулей, добавляя посредника – центральный элемент управления. Это позволяет модулям отправлять и слушать сообщения, не затрагивая остальной части системы. Сообщения могут быть обработаны любым количеством модулей сразу.
* Благодаря слабой связанности кода, внедрение новой функциональности происходит существенно легче.

Недостатки паттерна Mediator:

* Модули больше не могут взаимодействовать напрямую. Использование медиатора приводит к небольшому падению производительности – такова природа слабой связанности – становится достаточно трудно определить реакцию системы, отталкиваясь только от событий, происходящих в ней.
* Системы с высокой связанностью кода являются обычно источником всевозможных проблем, решением которых может стать уменьшение связанности [9]. Графическое изображение его можно увидеть на рисунке 6.2.

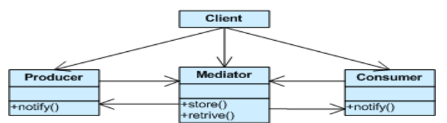


Рисунок 6.2 – Паттерн Mediator

Изобразим алгоритм функции авторизации пользователя.

Схема данного алгоритма приведена на рисунке 6.3. Сначала выполняется выборка необходимых данных посредствам выполнения sql-запроса. Выбранные данные обрабатываются в цикле, в котором для каждой записи выборки происходит сравнение логина и пароля с теми, которые были переданы с клиента. По достижению последней записи выборки осуществляется выход из тела цикла. Если совпадение было найдено, функция возвратит «success», если нет, то «fail».

Описанная функция реализована в данном приложении следующим образом:

public ResultSet getResultFromTable() {

ResultSet rs = null;

try {

rs = stmt.executeQuery("SELECT \* FROM users");

} catch (SQLException e) {

}

return rs;

}

public String checkLogin(String login, String password) {

ResultSet rs = getResultFromTable();

int flag = 0;

String tableLogin = "";

String tablePassword = "";

try {

while (rs.next()) {

tableLogin = rs.getString("login");

tablePassword = rs.getString("password");

if (tableLogin.equals(login) && tablePassword.equals(password)) {

flag = 1;

break;

}

}

if (flag == 1) {

return "success";

} else {

return "fail";

}

} catch (Exception e) {

System.out.println("Exception in Table of users");

return "";

} finally {

mdbc.close(rs);

}

}

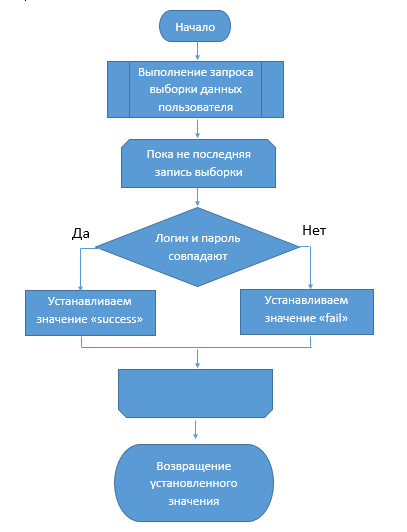


Рисунок 6.3 – Алгоритм функции авторизации пользователя

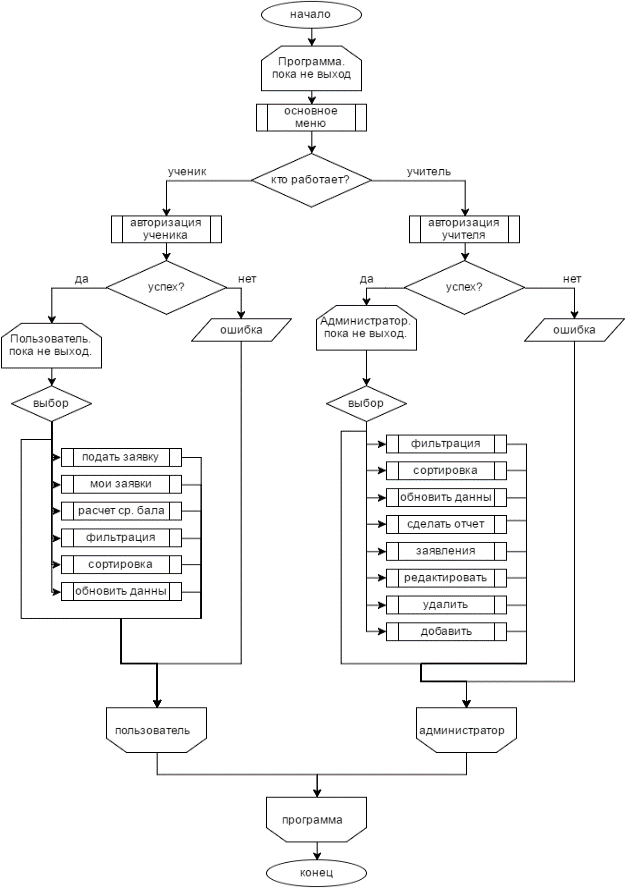


Рисунок 6.4 – Алгоритм работы всей программы

# **7 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

Данное программное приложение представляет собой автоматизированную систему учета успеваемости школьников. Приложение включает в себя два интерфейса: интерфейс учителя и ученика. Возможности использования у каждого из них различны.

Чтобы войти как учитель, нужно выбрать соответствующий radio button, ввести логин и пароль и нажать кнопку «войти». (Рисунок 7.1)

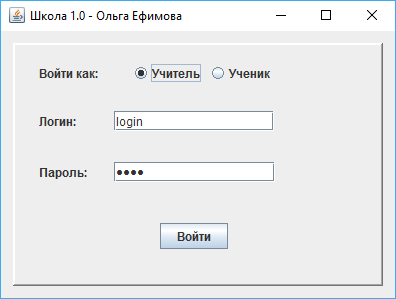


Рисунок 7.1 – Вход от имени учителя

Открывается меню учителя. (Рисунок 7.2)

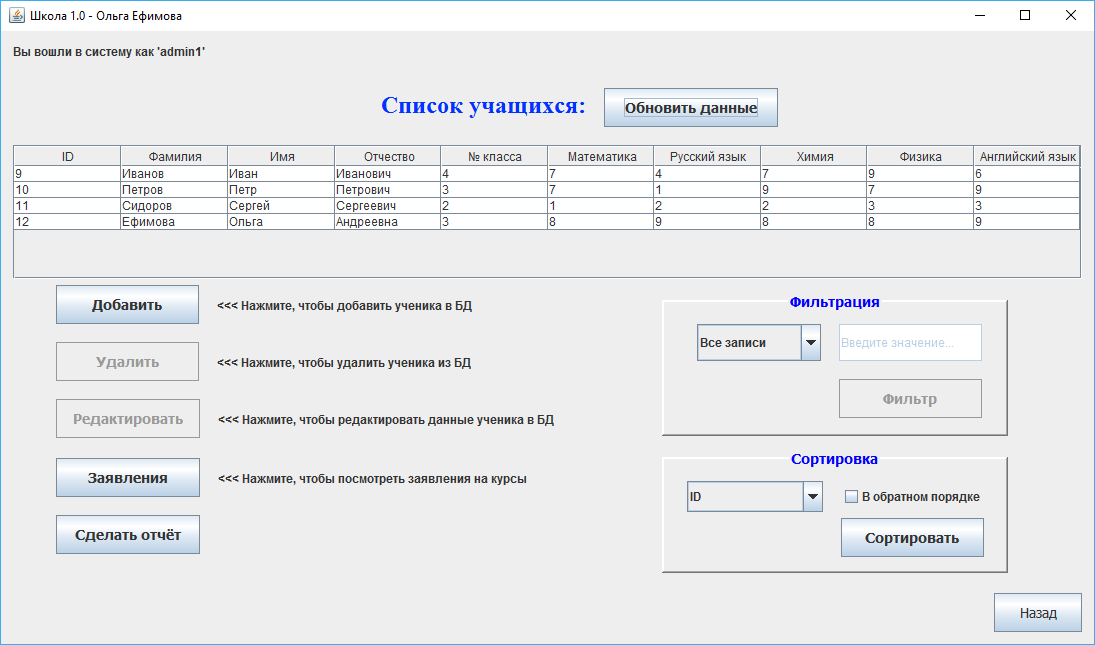


Рисунок 7.2 – Меню учителя

Чтобы добавить нового ученика в базу данных, нужно нажать кнопку «Добавить», после чего откроется диалоговое окно с формой для заполнения. (Рисунок 7.3) После заполнения формы и нажатия кнопки «Ок» новый ученик добавляется в базу данных.

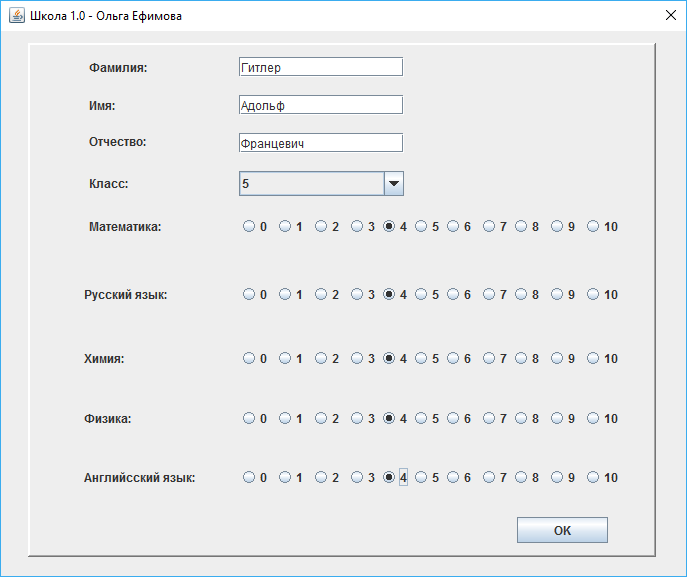


Рисунок 7.3 – Добавление нового ученика в базу данных

Чтобы отредактировать добавленную запись, можно для начала отыскать её в базе данных с помощью фильтрации. Для этого в пункте фильтрации нужно выбрать по какой характеристики будет происходить фильтрация, например, по фамилии ученика. После нажатия на выпадающий список и выбора нужного пункта, по которому должна происходить фильтрация, вводится значение в текстовое поле (в данном случае «Куц»). После этих действий нажимается кнопка фильтр. В таблице отображаются все записи, которые попали под данный критерий. (Рисунок 7.4)

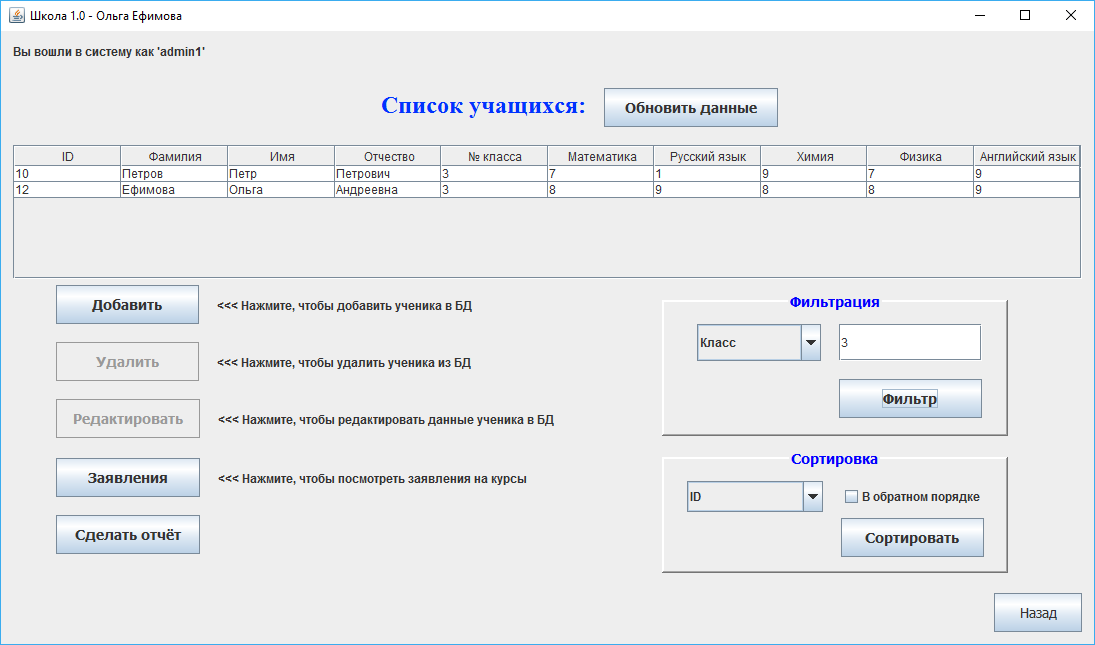
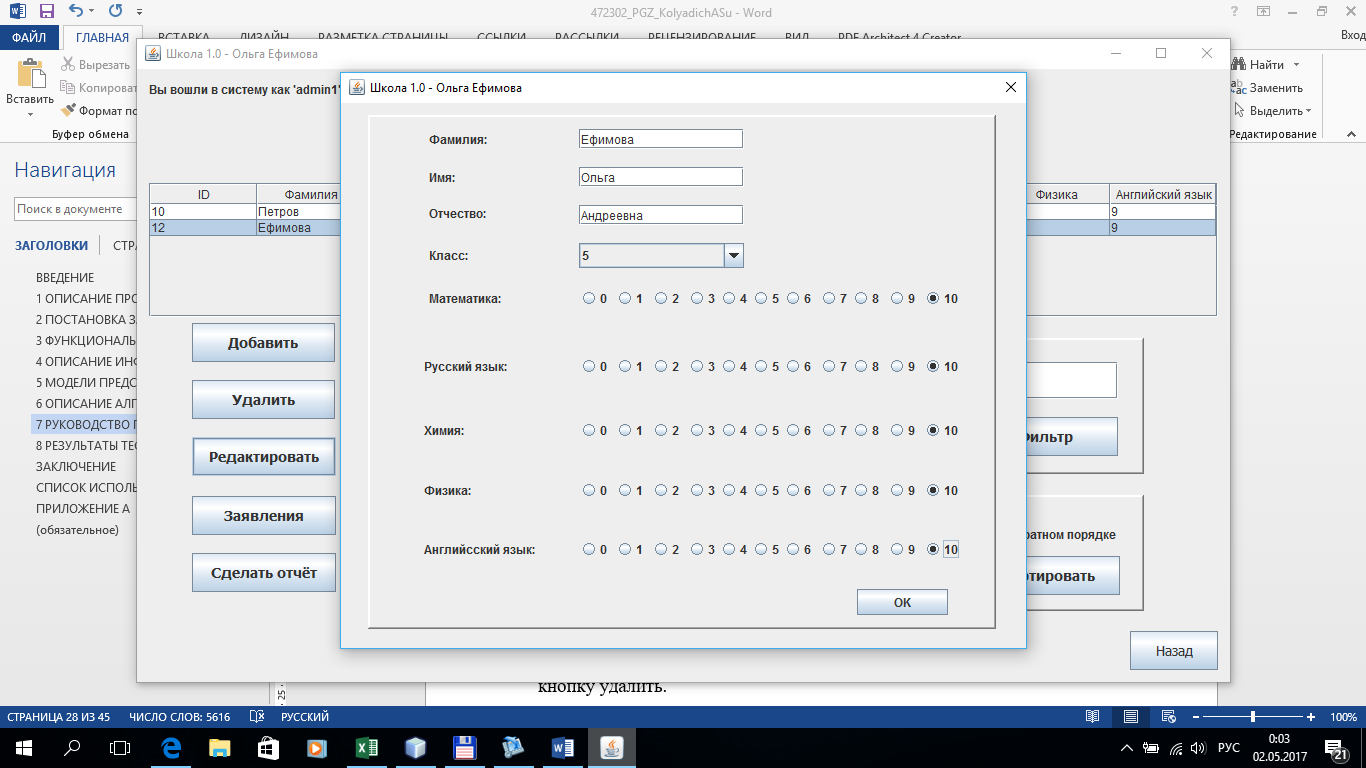


Рисунок 7.4 – Демонстрация работы фильтрации

После выбора соответствующего ученика в таблице можно отредактировать данную запись. Для этого нажимается кнопка «Редактировать», после этого открывается диалоговое окно, такое же, как и при добавление нового ученика. После редактирования записи (например поля «имя») нажимается кнопка «Ок». В таблице отображается запись в обновлённом варианте. (Рисунок 7.5)



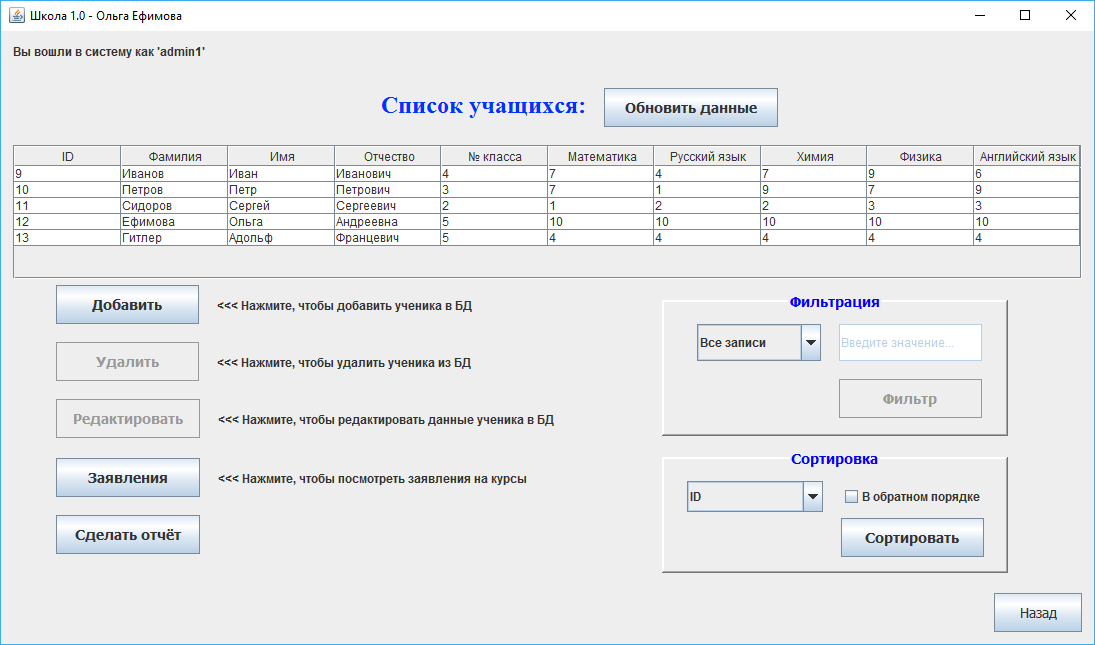


Рисунок 7.5 – Обновлённая запись

Чтобы удалить запись, нужно её выбрать в таблице и нажать на кнопку удалить.

Для сортировки всех записей, нужно нажать на выпадающий список в разделе «Сортировка», выбрать по какому параметру производить сортировку (Например, по фамилии) и нажать кнопку «Сортировать». По умолчанию сортировка проводится по возрастанию. Чтобы отсортировать в обратном порядке, нужно поставить галочку в check box, справа от выпадающего списка. После нажатия на кнопку «Сортировать», все данные будут отображаться в таблице в отсортированном виде (Рисунок 7.6).

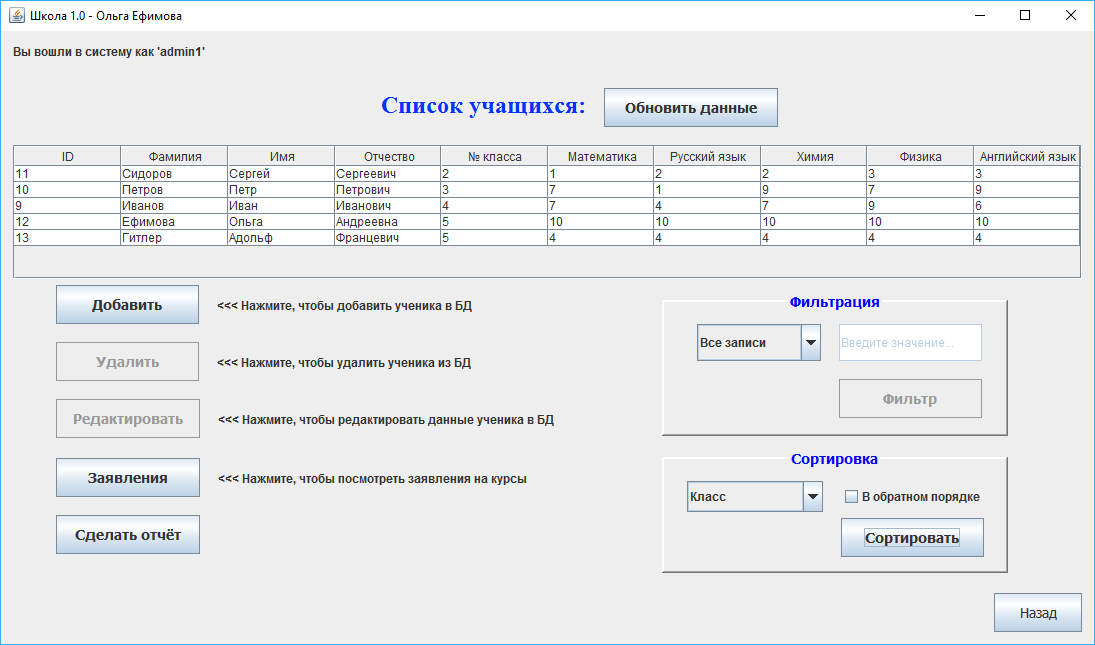


Рисунок 7.6 – Отсортированные записи

Чтобы посмотреть заказы всех учеников, нужно нажать кнопку «Заявления», после чего откроется диалоговое окно с заявлениями всех учеников (Рисунок 7.7).

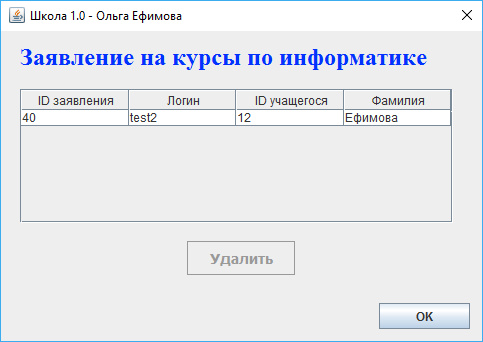


Рисунок 7.7 – Окно заявлений

Если нужно сохранить в файл информацию о всех учениках, находящихся в базе данных, необходимо нажать на кнопку «Сделать отчёт». Файл с данными можно найти на жёстком диске (Рисунок 7.8).

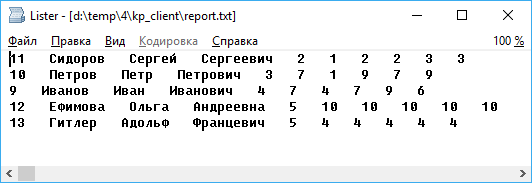


Рисунок 7.8 – Файл отчёта

Чтобы войти как ученик, нужно выбрать соответствующий radio button, ввести логин и пароль и нажать кнопку «войти». (Рисунок 7.9)

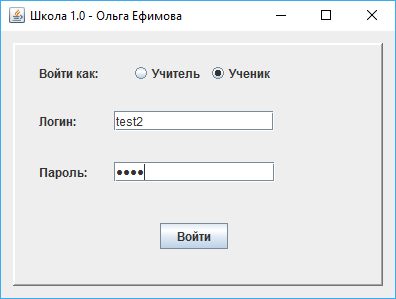


Рисунок 7.9 – Вход ученика

Открывается меню ученика. (Рисунок 7.10)

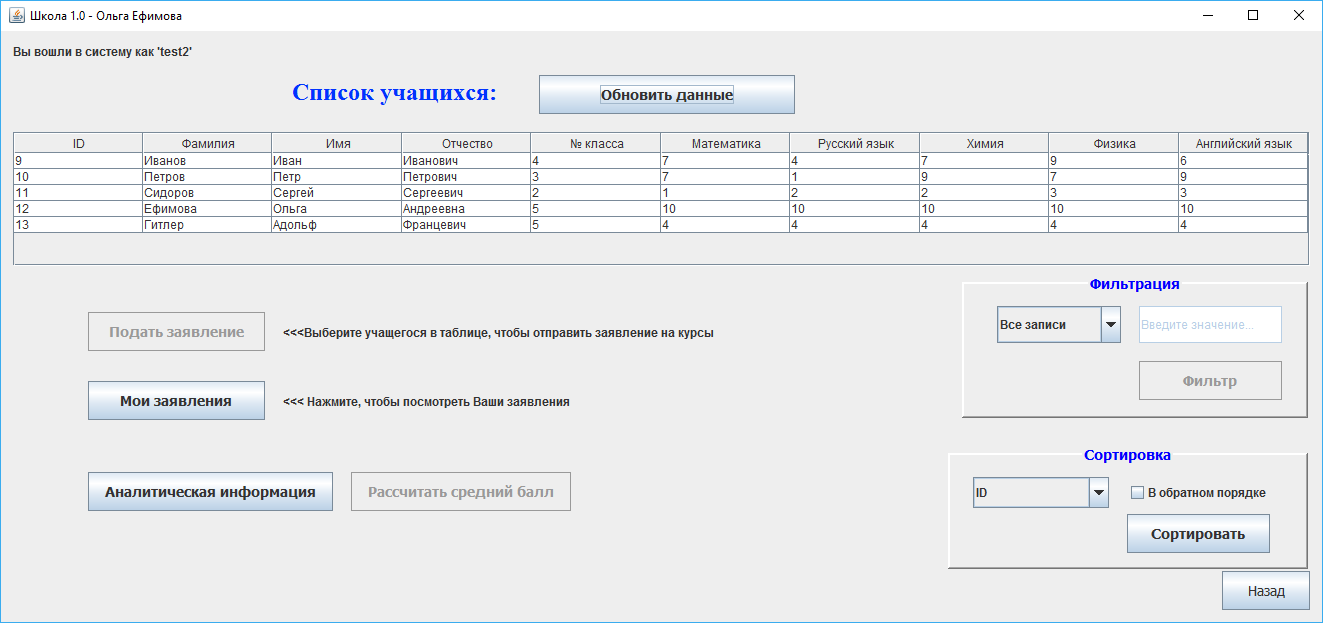


Рисунок 7.10 – Меню ученика

Ученик так же, как и учитель может фильтровать и сортировать данные в таблице. (Можно посмотреть выше)

Чтобы сделать заявление на курсы (Рисунок 7.11), необходимо нажать на соответствующую запись в таблице и кнопку «Подать заявление», после чего заявление попадает в раздел «Мои заявления».

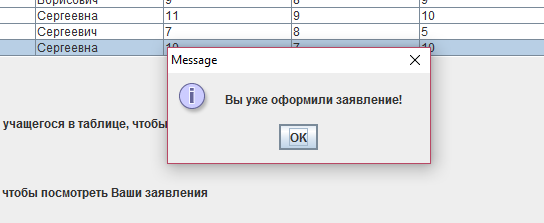


Рисунок 7.11 – Отправка заявления

Чтобы посмотреть заявления, необходимо нажать на кнопку «Мои заявления». (Рисунок 7.12)

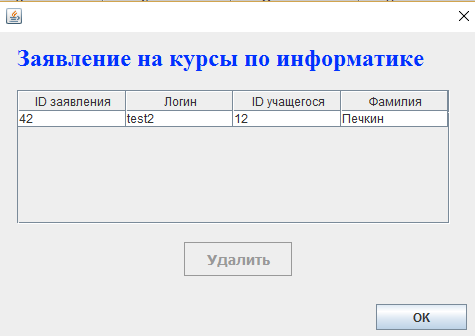


Рисунок 7.12 – Раздел «Мои заявления»

При необходимости можно удалить заказ, для этого нужно выбрать заказ в таблице и нажать кнопку удалить.

Для просмотра аналитической информации, а конкретно графика, который отображает количество учащихся и их средний балл, требуется нажать на кнопку «Аналитическая информация». Откроется диалоговое окно с графиком. (Рисунок 7.13)

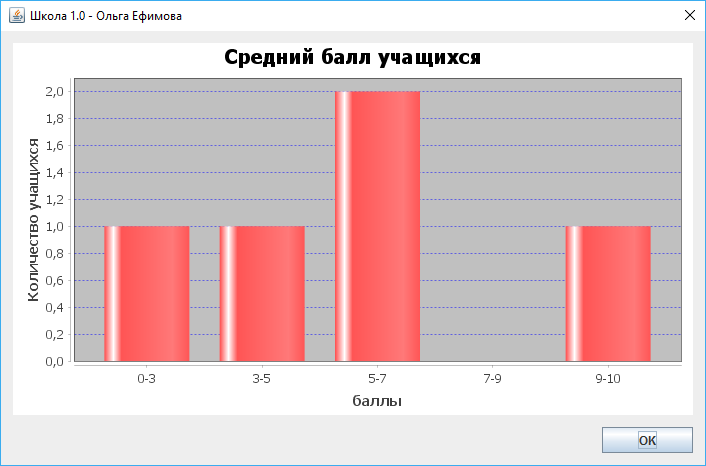


Рисунок 7.13 – Аналитическая информация

Вывод по разделу: в данном разделе была полностью рассмотрена функциональная сторона приложения. Реализованы функции, которые помогут с лёгкостью управлять базой данных ученикам и учителям школы.

# **8 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ РАЗРАБОТАННОЙ СИСТЕМЫ**

В ходе выполнения курсового проекта было разработано клиент-серверное приложение автоматизации учета успеваемости школьников.

Необходимо проверить реакцию системы на подключение нескольких клиентов одновременно. Если к серверу подключается несколько клиентов, то в консоли выводится количество активных подключений. (Рисунок 8.1)

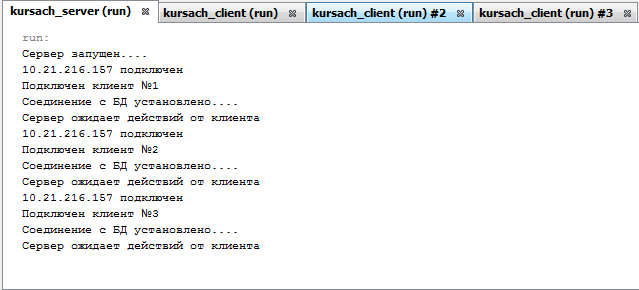


Рисунок 8.1 – Статистика активных клиентов сервера

Если на этапе авторизации не ввести логин или пароль, вы увидите соответствующую подсказку. (Рисунок 8.2 и Рисунок 8.3)

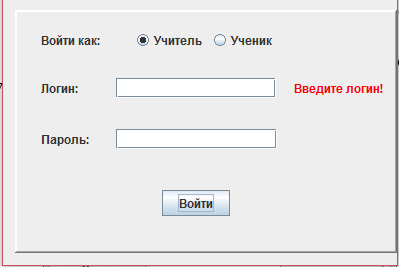


Рисунок 8.2 – Незаполненное поле ввода логина

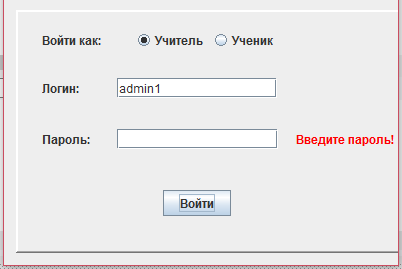


Рисунок 8.3 – Незаполненное поле ввода пароля

При попытке войти под логином не зарегистрированного пользователя или при вводе неверного пароля, система выдаст предупреждение. (Рисунок 8.4)

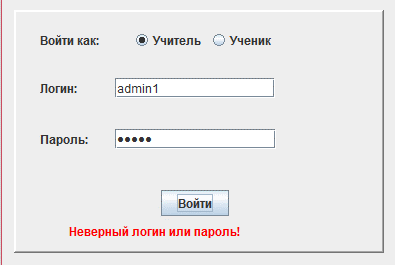


Рисунок 8.4 – Неверный логин или пароль

При попытке подать заново заявление появится предупреждение. (Рисунок 8.5)

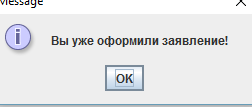


Рисунок 8.5 – Предупреждение

# **ЗАКЛЮЧЕНИЕ**

Информационные технологии не стоят на месте, и порой невозможно проследить за выпуском нового программного обеспечения. Но именно это призвано значительно упростить жизнь человека, максимально автоматизируя всевозможные процессы.

Данное приложение создано для выполнения различных функции и запросов. Также оно сокращает время из-за автоматизации различных процессов и выявляет дополнительные ресурсы.

Конечно данное приложение может быть модернизировано для решения задач в более узком направлении. Также возможно расширение функций как у администратора, так и у пользователя. В будущем данный продукт будет совершенствоваться посредством добавления новых функций, улучшения уже существующих функций, добавления новых данных, подключением различных библиотек, улучшением дизайна и внедрением новых возможностей.

Вся информация хранится в базе данных.

В программе предусмотрено использование следующих аспектов:

* исполняемые файлы работают в среде 32х разрядной ОС Windows 7 и выше;
* приложение запускается без использования интегрированных средств разработки;
* интерфейс программы русскоязычный.

Выполнено моделирование с использованием стандартов IDEF0 (функциональное моделирование процессов предметной области решаемой задачи (не менее чем 4 уровня)). Использовано CASE средство allfusion process modeler. Использовано CASE средство allfusion erwin data modeler. UML 2.0 (модели представления системы на основе UML).

Для описания работы приложения представлен следующий графический материал:

* диаграмма вариантов использования (Use Case);
* диаграмма состояний (Statechart);
* диаграмма последовательностей (Sequence diagram);
* диаграмма компонентов (component diagram);
* диаграмма развертывания (deployment diagram);
* информационная модель;
* блок-схемы алгоритмов, реализующих бизнес-логику
* IDEF0 – диаграмма операции выдачи книги.

Итогом курсового проекта является разработанное графическое приложение, которое позволит автоматизировать учет успеваемости школьников. Обеспечит школьникам и учителям быстрый доступ к данным.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Школа [Электронный ресурс]. – Электронные данные. – Режим доступа: https://ru.wikipedia.org

[2] Информационные технологии [Электронный ресурс]. – Электронные данные. – Режим доступа: http://festival.1september.ru

[3] Апатова Н.В. Информационные технологии в школьном образовании.- М.: изд-во РАО., 1994.- 228 с.Библиотека

[4] Громов Г.Р. Очерки информационной технологии.- М., 1993

[5] Новые педагогические и информационные технологии в системе образования: Учеб. пособие для студ. пед. вузов и системы повыш. квалиф. пед. кадров / Е.С. Полат, М.Ю. Бухаркина, М.В. Моисеева, А.Е. Петров; Под ред. Е.С. Полат. - М.: Издательский центр “Академия”, 1999. – 224 с.

[6] Проблемы учета успеваемости [Электронный ресурс]. – Электронные данные. – Режим доступа: http://www.cs-alternativa.ru/text/2198/3

[7] Внедрение электронных учебников в Беларуси [Электронный ресурс]. – Электронные данные. – Режим доступа: http://pleschenitsy.by/news/belarus/804-elektronnie-uchebniki

[8] Немного о Java [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.java.com/ru/download/faq/whatis\_java

[9] Клиент-сервер [Электронный ресурс]. – Электронные данные. – Режим доступа: http://www.mstu.edu.ru/study/materials/zelenkov/ch

[10] Понятие клиент-серверных систем [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://bourabai.kz/dbt/client1.htm>

[11] [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://www.bytemag.ru/articles/detail.php?ID=6547>

# **ПРИЛОЖЕНИЕ А**

# **(обязательное)**

**Листинг кода**

**Файл Server.java**

package server;

import java.io.\*;

import java.net.\*;

import java.sql.\*;

public class Server {

public static void main(String[] args) {

try {

int counter = 0;

ServerSocket serverSocket = new ServerSocket(8071);

System.out.println("Сервер запущен....");

while (true) {

// ожидание клиента

Socket socket = serverSocket.accept();

System.out.println(socket.getInetAddress().getHostName() + " подключен");

counter++;

/\*

создание отдельного потока для обмена данными

с соединившимся клиентом

\*/

System.out.println("Подключен клиент №" + counter);

ServerThread thread = new ServerThread(socket, counter); // запуск потока

thread.start();

}

} catch (IOException e) {

System.err.println(e);

}

}

}

class ServerThread extends Thread {

private InetAddress addr; // адрес клиента

private ObjectOutputStream os; // передача

private ObjectInputStream is; // прием

private String clientMessage;

private DBConnection mdbc;

private java.sql.Statement stmt;

private int counter;

private AdminTable adminTable;

private UserTable userTable;

private PupilTable pupilTable;

private OrderTable orderTable;

public ServerThread(Socket s, int counter) throws IOException {

this.counter = counter;

os = new ObjectOutputStream(s.getOutputStream());//поток вывода

is = new ObjectInputStream(s.getInputStream());//поток ввода

addr = s.getInetAddress();

mdbc = new DBConnection();

mdbc.init();

Connection conn = mdbc.getMyConnection();

try {

stmt = conn.createStatement();

adminTable = new AdminTable(stmt, mdbc);

userTable = new UserTable(stmt, mdbc);

pupilTable = new PupilTable(stmt, mdbc);

orderTable = new OrderTable(stmt, mdbc);

} catch (SQLException ex) {

System.out.println(ex);

}

}

public void writeObj(String str) {

clientMessage = str;

try {

os.writeObject(clientMessage);

} catch (IOException e) {

System.err.println("ошибка I/О потока" + e);

}

}

public void run() {

int i = 0;

String messageToClient = "";

String str;

String ThreadStop = "";

try {

System.out.println("Сервер ожидает действий от клиента");

while (!ThreadStop.equals("Exit")) {

str = (String) is.readObject();

String messageParts[] = str.split(",");

switch (messageParts[0]) {

case "checkUserLogin":

String UserLogin = messageParts[1];

String UserPassword = messageParts[2];

messageToClient = userTable.checkLogin(UserLogin, UserPassword);

writeObj(messageToClient);

break;

case "checkAdminLogin":

String AdminLogin = messageParts[1];

String AdminPassword = messageParts[2];

messageToClient = adminTable.checkLogin(AdminLogin, AdminPassword);

writeObj(messageToClient);

break;

case "initTable":

messageToClient = pupilTable.ReadAllRecord();

writeObj(messageToClient);

break;

case "addInPupilsTable":

messageToClient = pupilTable.AddInTable(messageParts);

writeObj(messageToClient);

break;

case "deleteFromPupilsTable":

String ID = messageParts[1];

messageToClient = pupilTable.DeleteFromTable(ID);

writeObj(messageToClient);

break;

case "editInPupilsTable":

messageToClient = pupilTable.EditInTable(messageParts);

writeObj(messageToClient);

break;

case "filterInPupilTable":

String FilterColumn = messageParts[1];

String FilterValue = messageParts[2];

messageToClient = pupilTable.FilterInTable(FilterColumn, FilterValue);

writeObj(messageToClient);

break;

case "sortInPupilTable":

String sortColumn = messageParts[1];

String ifDesc = messageParts[2];

messageToClient = pupilTable.SortInTable(sortColumn, ifDesc);

writeObj(messageToClient);

break;

case "addInOrdersTable":

messageToClient = orderTable.AddInTable(messageParts);

writeObj(messageToClient);

break;

case "showUserOrders":

String loginValue = messageParts[1];

messageToClient = orderTable.FilterInTable("", loginValue);

writeObj(messageToClient);

break;

case "initTableOrders":

messageToClient = orderTable.ReadAllRecord();

writeObj(messageToClient);

break;

case "deleteFromOrdersTable":

String id = messageParts[1];

messageToClient = orderTable.DeleteFromTable(id);

writeObj(messageToClient);

break;

case "Exit":

ThreadStop = "Exit";

break;

default:

System.err.println("Неизвестная команда");

}

}

} catch (Exception e) {

System.err.println("Соединение закрыто");

} finally {

disconnect(); // уничтожение потока

}

}

public void disconnect() {

try {

if (os != null) {

os.close();

}

if (is != null) {

is.close();

}

System.out.println(addr.getHostName() + " Закрытие соединения " + counter + "го клиента");

} catch (IOException e) {

e.printStackTrace();

} finally {

this.interrupt();

}

}

}