бюджетное профессиональное образовательное учреждение Вологодской области «Череповецкий лесомеханический техникум им. В.П. Чкалова»

Специальность 09.02.07 Информационные системы и программирование

ДИПЛОМНЫЙ ПРОЕКТ

РАЗРАБОТКА ИНТЕГРИРОВАННОЙ ЦИФРОВОЙ ПЛАТФОРМЫ ДЛЯ АВТОМАТИЗАЦИИ АДМИНИСТРАТИВЫХ ПРОЦЕССОВ ШКОЛЫ

СТУДЕНТКА: 4 курса, группы ИС-41

Шевель Милена Александровна
ФИО

подпись

НАУЧНЫЙ РУКОВОДИТЕЛЬ: Преподаватель
должность

Леонова Анна Васильевна
ФИО

подпись

РЕЦЕНЗЕНТ: Менеджер по верхнеуровневой автоматизации, Управление технологической автоматизации, кластер Индустриальные процессы, АО «Северсталь-Инфоком»
должность

Петухов Илия Александрович
ФИО

подпись

ДОПУЩЕНО К ЗАЩИТЕ: « ___ » _____ 20 ___ г.

ДАТА ЗАЩИТЫ: « ___ » _____ 20 ___ г.

ЗАВ. ОТДЕЛЕНИЕМ: / Хлопотина О.В.
подпись ФИО

ОЦЕНКА ЗАЩИТЫ: _____

г. Череповец

2026 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 4
1 ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 9
1.1 Анализ специфики административных процессов в современной образовательной организации и разбор нормативной базы 9
1.2 Выявление проблем автоматизации 12
1.3 Описание пользователей и заинтересованных лиц 13
1.4 Анализ существующих решений и их недостатки 14
1.5 Постановка задачи на разработку интегрированной цифровой платформы 16
1.6 Обоснование архитектуры и технологического стека 17
2 ПРОЕКТИРОВАНИЕ ПЛАТФОРМЫ 26
2.1 Разработка концептуальной модели данных и описание сущностей .. 26
2.2 Логическое проектирование базы данных 29
2.2.1 Приведение к третьей нормальной форме (3НФ) 29
2.2.2 Описание структуры таблиц базы данных 30
2.3 Выбор СУБД 35
2.4 Разработка UML-диаграмм проектирования 37
3 РАЗРАБОТКА И ТЕСТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА . 64
3.1 Создание структуры базы данных и разработка серверной части приложения 64
3.2 Разработка пользовательского интерфейса и административной панели системы 65
3.3 Реализация механизмов авторизации, аутентификации и разграничения прав доступа 81
3.4 Тестирование функционала платформы 83

4 ОЦЕНКА ЭФФЕКТИВНОСТИ И ВНЕДРЕНИЕ 87
4.1 Методика оценки социально-экономической эффективности от внедрения цифровой платформы в образовательный процесс 87
4.2 Расчёт затрат на разработку и оценка ожидаемого экономического и социального эффекта 88
4.2.1 Трудозатраты на разработку и отладку программы 88
4.2.2 Расчет себестоимости программного продукта 91
4.2.3 Расчет цены программного продукта 96
4.2.4 Расчет экономической эффективности 97
4.3 Описание перспектив развития и модернизации разработанной платформы 103
ЗАКЛЮЧЕНИЕ 104
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 109
ПРИЛОЖЕНИЕ А 113
ПРИЛОЖЕНИЕ Б 131
ПРИЛОЖЕНИЕ В 149

ВВЕДЕНИЕ

Актуальность исследования. В рамках национального проекта «Образование», который продлён до 2030 года, одной из ключевых задач является создание единой цифровой образовательной среды. Однако большинство школ до сих пор используют разрозненные инструменты (бумажные журналы, Excel, текстовые документы) для управления расписанием, учётом посещаемости, кадровым делопроизводством и отчётностью. Это ведёт к дублированию данных, ошибкам при ручном вводе, высоким временным затратам и снижению эффективности управления. Отсутствие единого защищённого хранилища с разграничением доступа создаёт риски утечки персональных данных и замедляет принятие управленческих решений [1,2].

Объект исследования — это повседневная административная работа школы, а именно: учёт сотрудников и учеников, составление расписания и замена уроков, оформление приказов и справок, а также подготовка отчётов.

Предмет исследования — методы, алгоритмы и технологии проектирования и разработки клиент-серверных приложений (включая реляционные базы данных на PostgreSQL, кроссплатформенную среду AvaloniaUI и язык C#) для автоматизации административных процессов школы.

Цель работы — разработка функционального прототипа интегрированной цифровой платформы для автоматизации административной деятельности школы, обеспечивающей централизованное хранение данных, ролевую модель доступа и снижение временных затрат сотрудников.

Для достижения поставленной цели необходимо решить следующие задачи исследования:

1. Проанализировать административные процессы современной школы и нормативно-правовую базу (ФЗ № 152-ФЗ, ФЗ № 273-ФЗ, СанПиН, ГОСТ 34.602-2020 и др.) для формирования требований к системе [1,2,3,5].
2. Выявить ключевые проблемы автоматизации в деятельности директора, завуча, секретаря и учителей.
3. Провести сравнительный анализ существующих на рынке решений («Дневник.ру», «Сетевой город», ФГИС «Моя школа») и выявить их недостатки (высокая стоимость, отсутствие гибкости настройки, привязка к облачным серверам).
4. Спроектировать логическую и физическую структуру базы данных в СУБД PostgreSQL с обеспечением ссылочной целостности и нормализации.
5. Обосновать выбор архитектуры «клиент-сервер» и технологического стека (PostgreSQL, AvaloniaUI, Entity Framework Core) [23,24,25,26].
6. Описать ключевые бизнес-процессы (учет контингента, управление расписанием, замена учителей, формирование справок и отчетов) в формате диаграмм деятельности.
7. Разработать пользовательские сценарии (Use-case) для каждой роли: администратора системы, директора, завуча, секретаря, учителя.

Методы исследования. В ходе выполнения дипломного проекта применялись следующие методы: системный анализ для изучения предметной области; сравнительный анализ для оценки существующих решений; метод формализации и нормализации для построения моделей данных; объектно-ориентированное моделирование с использованием нотации UML (диаграммы вариантов использования, диаграммы деятельности); структурное проектирование архитектуры «клиент-сервер»;

экспериментальный метод для тестирования прототипа программного продукта.

Теоретическая значимость работы заключается в систематизации подходов к автоматизации административных процессов образовательных учреждений и разработке методики проектирования кроссплатформенных приложений с использованием свободной СУБД PostgreSQL и AvaloniaUI.

Практическая значимость заключается в возможности прямого использования разработанной базы данных и интерфейсов приложения для внедрения в реальном образовательном учреждении. Внедрение платформы позволяет сократить временные затраты административного персонала на 30-40%, минимизировать ошибки при формировании отчетности (в частности, табелей успеваемости и справок), обеспечить прозрачность замен уроков для учителей и снизить бюрократическую нагрузку на педагогический состав за счет автоматизации рутинных операций.

Структура дипломного проекта. Работа состоит из введения, двух глав, заключения, списка литературы и приложений. В первой главе проводится анализ предметной области и обоснование разработки, во второй главе описывается проектирование системы (базы данных, архитектуры, пользовательских сценариев).

Словарь терминов:

Avalonia UI — кроссплатформенный фреймворк для разработки десктопных приложений на C# и .NET, обеспечивающий работу под Windows, Linux и macOS.

Entity Framework Core (EF Core) — объектно-реляционный маппер (ORM) от Microsoft, позволяющий работать с базой данных через объекты C# без написания SQL-запросов.

ER-диаграмма — графическая нотация для представления концептуальной модели данных (сущности, их атрибуты и связи).

PostgreSQL — свободная реляционная СУБД с открытым исходным кодом, поддерживающая ACID-транзакции и сложные запросы.

UML — унифицированный язык графического моделирования для описания, визуализации и проектирования программных систем.

Аутентификация — проверка подлинности пользователя путём сравнения введённого пароля с хешем в базе данных.

Авторизация — предоставление пользователю прав доступа к определённым функциям системы после успешной аутентификации.

Диаграмма вариантов использования (Use Case) — диаграмма UML, отображающая функциональные требования к системе с точки зрения пользователей (акторов).

Диаграмма классов — диаграмма UML, описывающая статическую структуру системы: классы, их атрибуты, методы и связи.

Диаграмма деятельности (Activity) — диаграмма UML, моделирующая поток управления в бизнес-процессе.

Диаграмма последовательности (Sequence) — диаграмма UML, показывающая временную последовательность сообщений между объектами.

Диаграмма состояний (State Machine) — диаграмма UML, описывающая возможные состояния объекта и переходы между ними.

Клиент-серверная архитектура — модель взаимодействия, при которой сервер предоставляет ресурсы, а клиент эти ресурсы запрашивает.

Кроссплатформенность — свойство программного обеспечения работать под управлением нескольких операционных систем без изменения кода.

Модульное тестирование — тестирование отдельных модулей (методов, классов) программы в изоляции от остальных компонентов.

Ролевая модель доступа — подход к разграничению прав доступа, при котором каждому пользователю назначается роль с определёнными разрешениями.

Ссылочная целостность — свойство базы данных, гарантирующее, что внешние ключи ссылаются на существующие записи.

Функциональное тестирование — тестирование, проверяющее соответствие системы заявленным функциональным требованиям.

Хеширование пароля — преобразование пароля в фиксированную строку (хеш) с помощью криптографического алгоритма.

1 ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Успешное проектирование информационной системы невозможно без всестороннего изучения предметной области, для которой она создаётся. Качество проработки этого этапа напрямую определяет, насколько итоговый программный продукт будет соответствовать реальным потребностям и эффективно решать поставленные задачи.

1.1 Анализ специфики административных процессов в современной образовательной организации и разбор нормативной базы

Административная деятельность общеобразовательного учреждения представляет собой сложную многоуровневую систему, направленную на обеспечение функционирования школы как единого целого. В теории менеджмента образовательных систем административный процесс трактуется как совокупность взаимосвязанных действий и процедур, выполняемых управленческим персоналом (директором, завучами, секретарем) для достижения уставных целей учебного заведения.

В ходе анализа, проведенного на базе типовой муниципальной школы, были выделены четыре ключевых блока административных процессов, подлежащих автоматизации в рамках данной дипломной работы:

1. Управление кадрами (Кадровый менеджмент). Данный блок включает процессы фиксации и актуализации персональных данных сотрудников (учителей, администрации, технического персонала), учет их квалификации, педагогической нагрузки и трудового стажа. Особенностью процесса является необходимость ведения «истории изменений» (переводы, повышение квалификации).
2. Учет контингента учащихся. Центральный блок, охватывающий жизненный цикл ученика в школе: от приема и зачисления до перевода в

следующий класс (или параллель) и (или) отчисления. Процесс требует высокой точности ввода данных (ФИО, дата рождения, паспортные данные родителей, СНИЛС), так как на основе этих данных формируются справки и отчеты для государственных органов.

3. Организация учебного процесса и расписание. Этот блок включает распределение учебной нагрузки, составление расписания уроков с учетом санитарных норм, а также критически важный процесс замены уроков (при болезни учителя). Как показал анализ, ручное управление заменами в бумажном виде вызывает путаницу и информационный хаос.

4. Внутренний документооборот и формирование отчетности. Процессы создания, согласования и хранения приказов, заявлений и справок. Включает также формирование табелей успеваемости и сводных отчетов для управления образования.

Разработка программного обеспечения для хранения и обработки указанных данных, особенно персональных данных (ПДн), требует строгого соблюдения нормативно-правовой базы Российской Федерации. В ходе выполнения дипломного проекта была изучена следующая документация:

Федеральное законодательство:

1. Федеральный закон от 27 июля 2006 г. № 152-ФЗ «О персональных данных». Устанавливает принципы обработки ПДн (законность, справедливость, конфиденциальность). Применительно к разрабатываемой платформе на AvaloniaUI и PostgreSQL, этот закон обязывает реализовать: а) получение согласия субъектов (родителей, сотрудников); б) обеспечение неограниченного доступа субъекта к своим данным; в) разграничение прав доступа пользователей (разные роли видят разные поля) [1].

2. Федеральный закон от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации». Статья 98 данного закона обязывает школы формировать информационные системы для учета успеваемости и

промежуточной аттестации. Это придает юридическую силу электронным журналам и ведомостям, которые будут выгружаться из разрабатываемого приложения [2].

Национальные стандарты (ГОСТы):

3. ГОСТ 34.602-2020 «Информационные технологии. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы». Данный стандарт определил структуру раздела 1.5 «Постановка задачи на разработку», требуя однозначной формулировки целей, функций и требований к системе [5].
4. ГОСТ 19.201-78 «Единая система программной документации (ЕСПД). Техническое задание. Требования к содержанию и оформлению». Использовался для унификации оформления проектной документации (Приложение А) [4].
5. ГОСТ Р ИСО/МЭК 25010-2015 «Модели качества систем и программных продуктов». Определяет характеристики качества, которым должна соответствовать платформа: функциональная пригодность (соответствие описанным бизнес-процессам), надежность (отсутствие сбоев при работе с PostgreSQL), удобство использования (интуитивный интерфейс AvaloniaUI) и безопасность (защита от несанкционированного доступа) [6].

Санитарные нормы (СанПиН):

6. СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности...» (ранее СанПиН 2.4.2.2821-10). Нормирует непрерывную длительность использования ПК на уроках. При проектировании модуля «Расписание» необходимо предусмотреть, чтобы система не предлагала ставить «сдвоенные» уроки информатики для младших классов дольше допустимого времени [3].

Таким образом, детальный анализ 6 ключевых нормативных документов показал, что разработка должна вестись с учетом требований к

защите данных (152-ФЗ), юридической значимости отчетности (273-ФЗ), формализации ТЗ (ГОСТ 34.602-2020) и эргономики (СанПиН).

1.2 Выявление проблем автоматизации

В ходе анализа текущей организации работы в школах был выявлен комплекс системных проблем, которые послужили предпосылками для старта разработки.

Проблема разрозненности и дублирования информации. Данные о сотрудниках и учащихся (ФИО, адреса, телефоны) часто хранятся в трех-четырех местах одновременно: в бумажном личном деле, в Excel-таблице у секретаря, в текстовом документе у завуча по УВР и в сторонней системе для питания. Как показал опрос, до 20% рабочего времени секретаря уходит на сверку этих списков и устранение расхождений (например, когда фамилия ученика в журнале и в списке на питание написана по-разному).

Проблема ручного труда при формировании отчетности. Завуч вынужден вручную, используя калькулятор, высчитывать средний балл успеваемости по классам и предметам за четверть. Учитель тратит часы на заполнение бумажных ведомостей. Секретарь, при оформлении справки об обучении, перепечатывает одни и те же данные (ФИО ребенка, дата рождения, класс) из одного документа в другой, что создает риск опечаток.

Проблема неэффективного управления заменами. При болезни учителя в «бумажной» школе возникает хаос: завуч ищет свободного педагога вручную, делает пометки на распечатанном экземпляре расписания, информация до классов доходит с опозданием или искажается. Отсутствие актуального расписания (с заменой) на рабочем столе учителя приводит к срыву уроков.

Проблема привязки к коммерческим облачным серверам и интернету. Существующие облачные системы (например, «Дневник.ру») требуют постоянного доступа в интернет и ежемесячной оплаты. При перебоях с провайдером школа полностью теряет доступ к управлению. Кроме того, школы не хотят хранить чувствительные персональные данные на сторонних серверах (риск утечки) [1,12], предпочитая локальное развертывание.

Проблема стоимости владения. Лицензии на коммерческие СУБД (MS SQL Server) для школы могут быть финансово обременительны [12,15]. Требуется решение, использующее бесплатное, но мощное программное обеспечение.

1.3 Описание пользователей и заинтересованных лиц

В системном анализе пользователь — это любое лицо или организация, взаимодействующее с системой для достижения определенной цели. Заинтересованное лицо (stakeholder) — лицо или группа лиц, имеющая законные интересы, права или требования в отношении свойств разрабатываемой системы. Для данной платформы идентифицированы следующие группы:

1. Системный администратор. Это пользователь, отвечающий за техническое состояние инфраструктуры. В рамках проекта именно он выполняет начальную настройку: при первом запуске приложения AvaloniaUI выбирает тип СУБД (в данном случае PostgreSQL), вводит параметры подключения (хост, порт, логин/пароль) и инициализирует БД.
2. Директор школы. Ключевое заинтересованное лицо, принимающее стратегические решения. Функционал директора: просмотр сводных отчетов (статистика по кадрам, успеваемости), утверждение ключевых документов

(приказов, планов). Директору не нужен доступ к ежедневной рутине, ему нужны «панели управления» с агрегированными данными.

3. Секретарь (или специалист по кадрам). Основной оператор системы. Отвечает за ведение картотек «Сотрудники» и «Учащиеся» (CRUD-операции), формирование приказов о зачислении/отчислении, подготовку стандартных справок по запросу родителей. Секретарь является главным источником первичных данных.

4. Заведующий учебной частью (Завуч). Управляет учебным процессом. Задачи завуча: распределение нагрузки учителей, составление и корректировка расписания, проведение замен (согласно схеме «Завуч замена учителя»), контроль успеваемости. Именно для завуча критически важна функция автоматической замены и расчета средних баллов.

5. Учитель. Пользователь, который потребляет актуальную информацию и вносит данные об успеваемости. Учитель видит свое персональное расписание (с учетом произведенных замен), списки учащихся, выставляет оценки в электронный журнал (получая моментальный расчет среднего балла), формирует итоговые ведомости по кнопке «Сформировать ведомость».

Такое разделение ролей реализует фундаментальный принцип информационной безопасности — принцип минимальных привилегий, согласно которому пользователю предоставляется ровно тот объем прав, который необходим ему для выполнения должностных обязанностей [6,16].

1.4 Анализ существующих решений и их недостатки

Для обоснования целесообразности собственной разработки был проведен детальный сравнительный анализ систем-аналогов, представленных на российском рынке образовательных услуг.

«Дневник.ру» является крупнейшей облачной платформой. Достоинства: мощный инструментарий для дистанционного обучения, современный интерфейс. Недостатки: платная подписка для расширенного функционала, хранение данных на внешних серверах (риск для ПДн согласно 152-ФЗ [1]), невозможность модификации под нужды конкретной школы. Отсутствует полноценный кадровый учет и учет материально-технической базы.

«Сетевой Город. Образование» — муниципальная система. Достоинства: наличие модулей кадрового учета, бесплатная для школ. Недостатки: морально устаревший интерфейс, крайне низкая производительность при массовых нагрузках (конец четверти), сложность внесения оперативных изменений (например, замена учителя требует много кликов), зависимость от региональных операторов связи.

ФГИС «Моя школа» — государственная платформа. Достоинства: бесплатность, интеграция с госуслугами, верифицированный контент. Недостатки: система находится в стадии постоянной доработки (нестабильность API), отсутствует гибкость — школы вынуждены подстраиваться под жесткие рамки платформы, ограниченные возможности для внутреннего документооборота (оформление справок по шаблонам школы невозможно).

Традиционный метод (Excel + Бумага). Достоинства: дешевизна и полный контроль. Недостатки: полное отсутствие актуальности данных в реальном времени, ошибки при копировании данных, невозможность многопользовательской работы.

Сравнительный анализ представлен в таблице 1.1 [методика анализа по ГОСТ 34.602-2020] [5].

Таблица 1.1 — Сравнительный анализ существующих решений

Критерий	«Дневник.ру»	«Сетевой город»	ФГИС «Моя школа»	Разрабатываемая платформа
Модель распространения	Условно-бесплатная	Бесплатная	Бесплатная	Бесплатная (Open Source)
Требование к интернету	Постоянное	Постоянное	Постоянное	Только локальная сеть
Локальное развертывание	Нет	Да (редко)	Нет	Да (PostgreSQL локально)
Формирование справок	Стандартные шаблоны	Нет гибкости	Нет	Гибкие шаблоны (Word/PDF)

Вывод анализа: существующие системы не решают проблему «быстрых замен» и не дают школе полного контроля над данными без ежемесячных платежей, что подтверждает актуальность разработки собственной платформы на PostgreSQL и AvaloniaUI.

1.5 Постановка задачи на разработку интегрированной цифровой платформы

Основываясь на выявленных проблемах и требованиях (ГОСТ 34.602-2020) [5], сформулирована задача на разработку программного продукта «Интегрированная цифровая платформа для автоматизации административной деятельности школы» (ИЦП «SchoolAssistancePlatform»). Платформа должна представлять собой кроссплатформенное десктопное приложение с клиент-серверной архитектурой.

Функциональные требования (что должна делать система):

1. Учет кадров и контингента: Реализовать разделы «Сотрудники» и «Учащиеся», обеспечивающие полный цикл CRUD (Создание, Чтение, Обновление, Удаление) записей. Обеспечить уникальность записей (контроль дублей) и возможность прикрепления сканов документов.
2. Управление расписанием и заменами: Разработать модуль «Расписание», позволяющий завучу в несколько кликов назначить замену

заболевшему учителю (согласно схеме «Завуч замена учителя»). Система должна автоматически актуализировать расписание для всех пользователей.

3. Автоматизация справок и документов: Создать механизм шаблонов (например, «Справка об обучении»), который автоматически подставляет данные (ФИО ученика, класс, номер школы) из базы данных при выборе ученика из списка.

4. Формирование отчетности: Обеспечить генерацию табелей успеваемости и сводных отчетов по кнопке с возможностью просмотра среднего балла (для учителя) и экспорта в PDF/Excel.

5. Ролевая модель и аутентификация: Реализовать вход по логину/паролю с разграничением доступа на 5 ролей: Администратор, Директор, Завуч, Секретарь, Учитель.

Нефункциональные требования (характеристики качества):

1. Кроссплатформенность: Приложение должно работать под управлением Windows и Linux (требование к AvaloniaUI).

2. Независимость от интернета: Работоспособность в локальной сети школы без выхода в WAN.

3. Безопасность: Хранение паролей в хешированном виде (PBKDF2). Защита от SQL-инъекций через Entity Framework Core.

4. Надежность: Автоматическое резервное копирование базы PostgreSQL средствами планировщика ОС.

1.6 Обоснование архитектуры и технологического стека

Для реализации поставленной задачи была выбрана архитектура «клиент-сервер». Обоснование выбора представлено в Таблице 1.2.

Таблица 1.2 — Обоснование технологического стека

Компонент	Выбранная технология	Обоснование выбора (согласно требованиям)
Сервер БД	PostgreSQL	Бесплатная СУБД, нет лицензионных отчислений [12,23]. Поддержка ACID для целостности данных об оценках.
Платформа приложения	Avalonia UI	Кроссплатформенность (Win/Linux/macOS) [24]. Нативное управление без веб-сервера.
Язык разработки	C# (.NET 10)	Типобезопасность, экосистема библиотек, нативная поддержка PostgreSQL через Npgsql [25].
ORM (Маппер)	Entity Framework Core	Абстракция от SQL, защита от SQL-инъекций [26]. Код работает с PostgreSQL и MS SQL.
Среда разработки	Visual Studio 2022	Удобный визуальный редактор для Avalonia UI (через расширение), мощные отладчики.

Описание бизнес-процессов и их реализация:

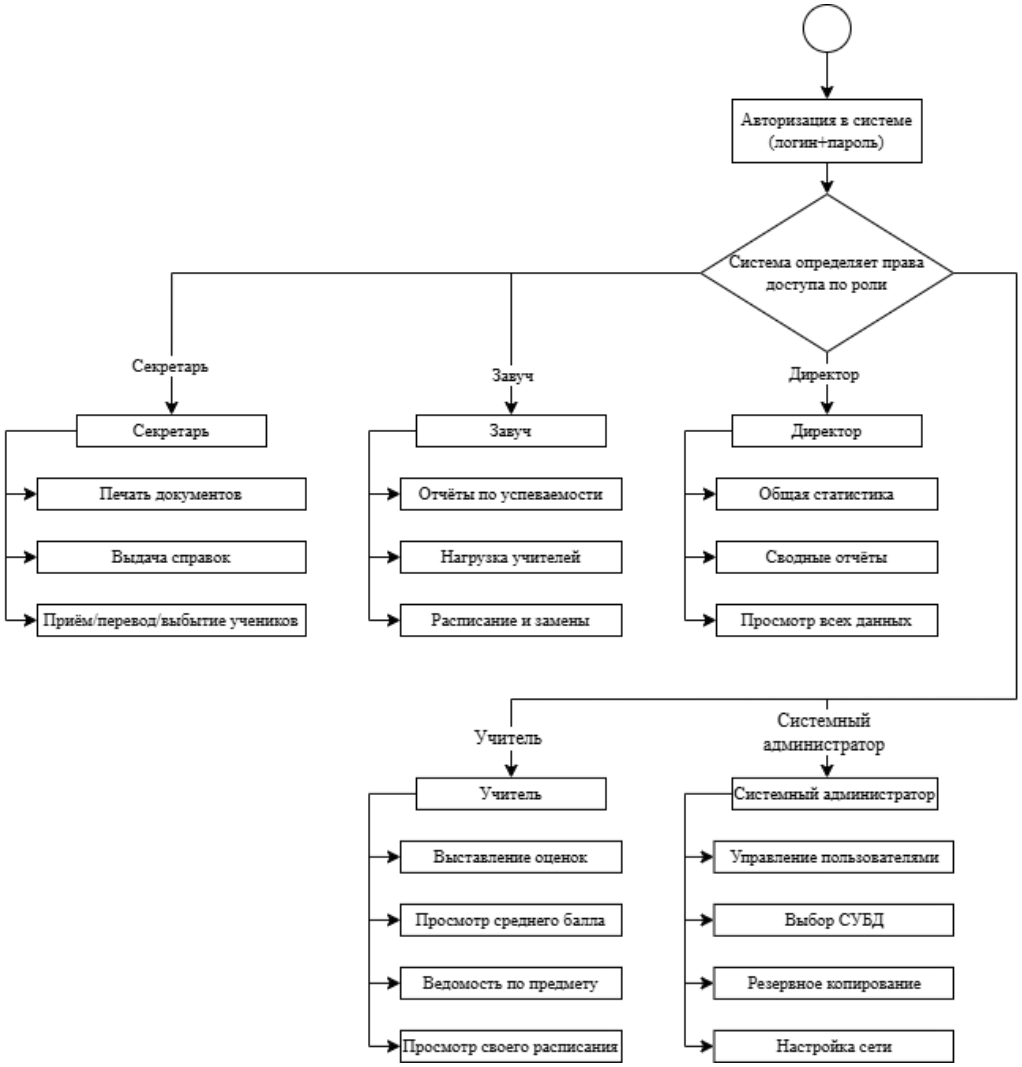


Рисунок 1.1 – Общая схема бизнес-процессов

На рисунке 1.1 показан процесс авторизации в системе: пользователь вводит логин и пароль, после чего система определяет его права доступа в зависимости от назначенной роли. Далее схема описывает пять ролей:

— Секретарь может печатать документы, выдавать справки, а также оформлять приём, перевод и выбытие учеников.

— Завуч формирует отчёты по успеваемости, управляет нагрузкой учителей, составляет расписание и занимается заменами.

— Директор имеет доступ к общей статистике, сводным отчётам и может просматривать любые данные системы.

— Учитель выставляет оценки, просматривает средний балл учеников, формирует ведомость по предмету и смотрит своё расписание.

— Системный администратор управляет пользователями, выбирает тип СУБД, настраивает резервное копирование и сеть.

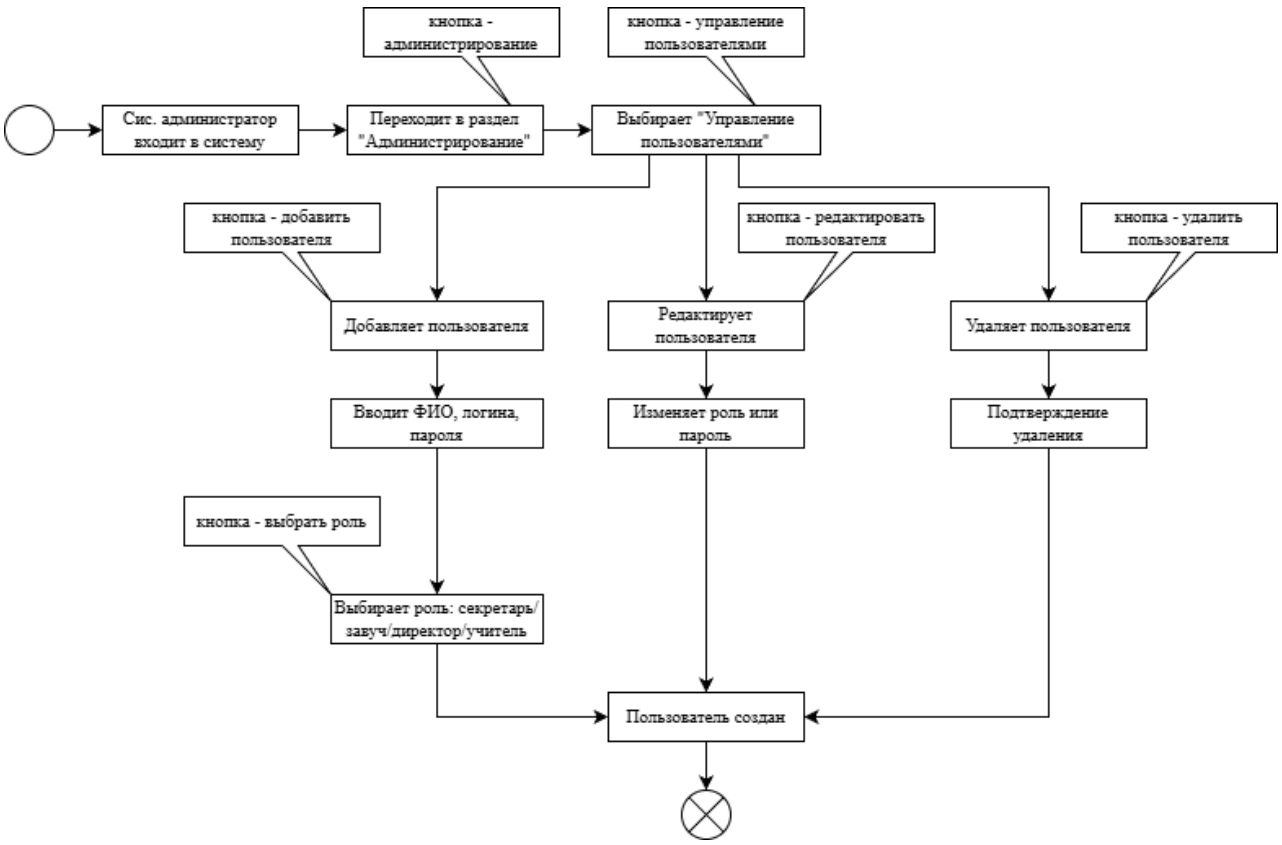


Рисунок 1.2 – Бизнес-процесс «Администратор. Управление пользователями»

На рисунке 1.2 изображён процесс администрирования пользователей. Системный администратор входит в систему, нажимает кнопку «Администрирование» и переходит в соответствующий раздел, где выбирает пункт «Управление пользователями». В этом разделе он может добавлять пользователей: вводит ФИО, логин, пароль и назначает роль (секретарь, завуч, директор или учитель). Также администратор может редактировать существующих пользователей — изменять роль или пароль. Удаление пользователя происходит после подтверждения операции.

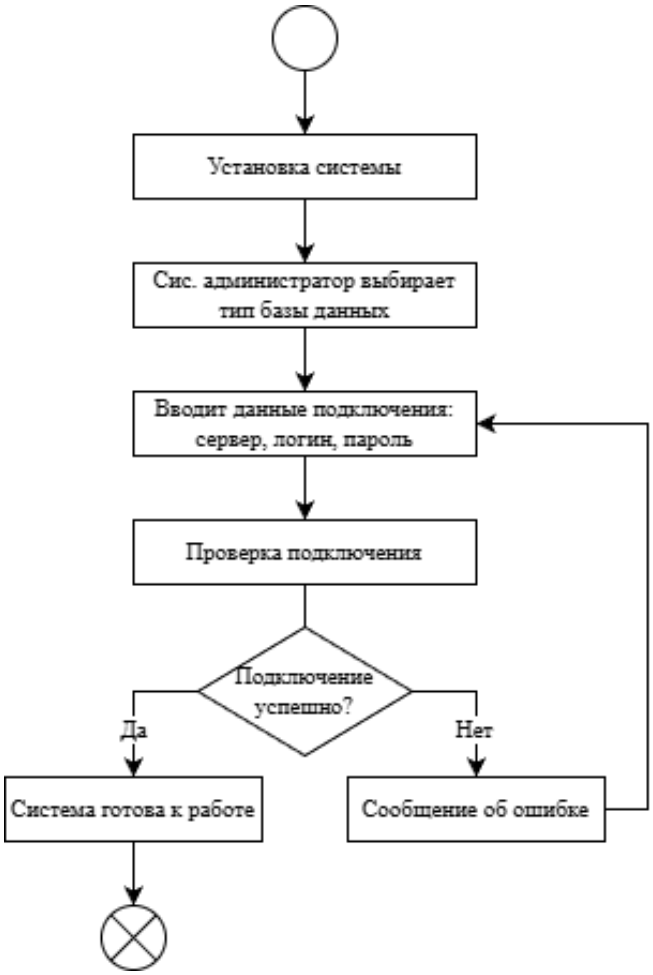


Рисунок 1.3 – Бизнес-процесс «Администратор. Выбор СУБД»

На рисунке 1.3 показан процесс установки системы на этапе выбора базы данных. Системный администратор выбирает тип СУБД, затем вводит

параметры подключения: сервер, логин и пароль. После этого выполняется проверка подключения. Если подключение успешно, система готова к работе. В случае ошибки выводится сообщение об ошибке, и администратор может повторить ввод данных.

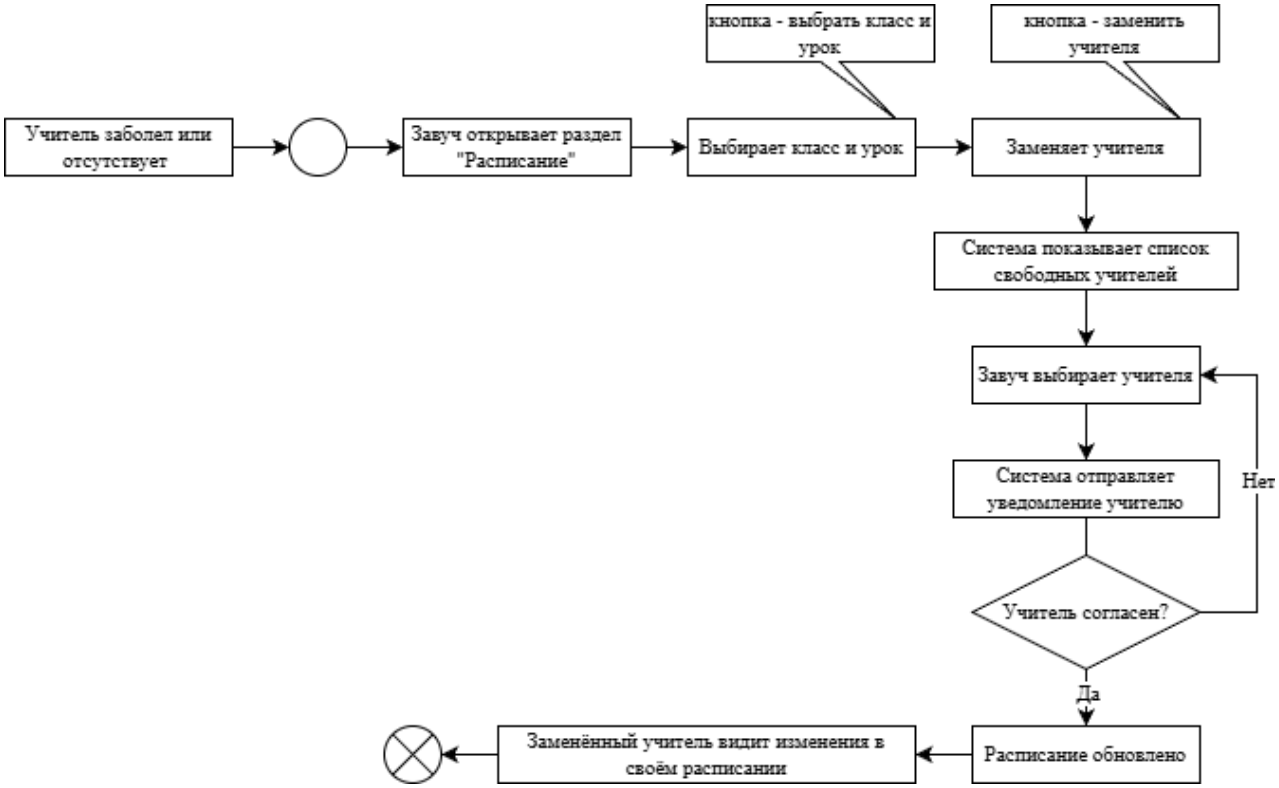


Рисунок 1.4 – Бизнес-процесс «Завуч. Замена учителя»

На рисунке 1.4 описан процесс замены отсутствующего учителя. Когда учитель заболел или отсутствует по другой причине, завуч открывает раздел «Расписание», выбирает класс и конкретный урок, после чего нажимает кнопку «Заменить учителя». Система показывает список свободных учителей. Завуч выбирает подходящего учителя, и система отправляет ему уведомление. Если учитель согласен, заменённый учитель видит изменения в своём расписании, и расписание обновляется.

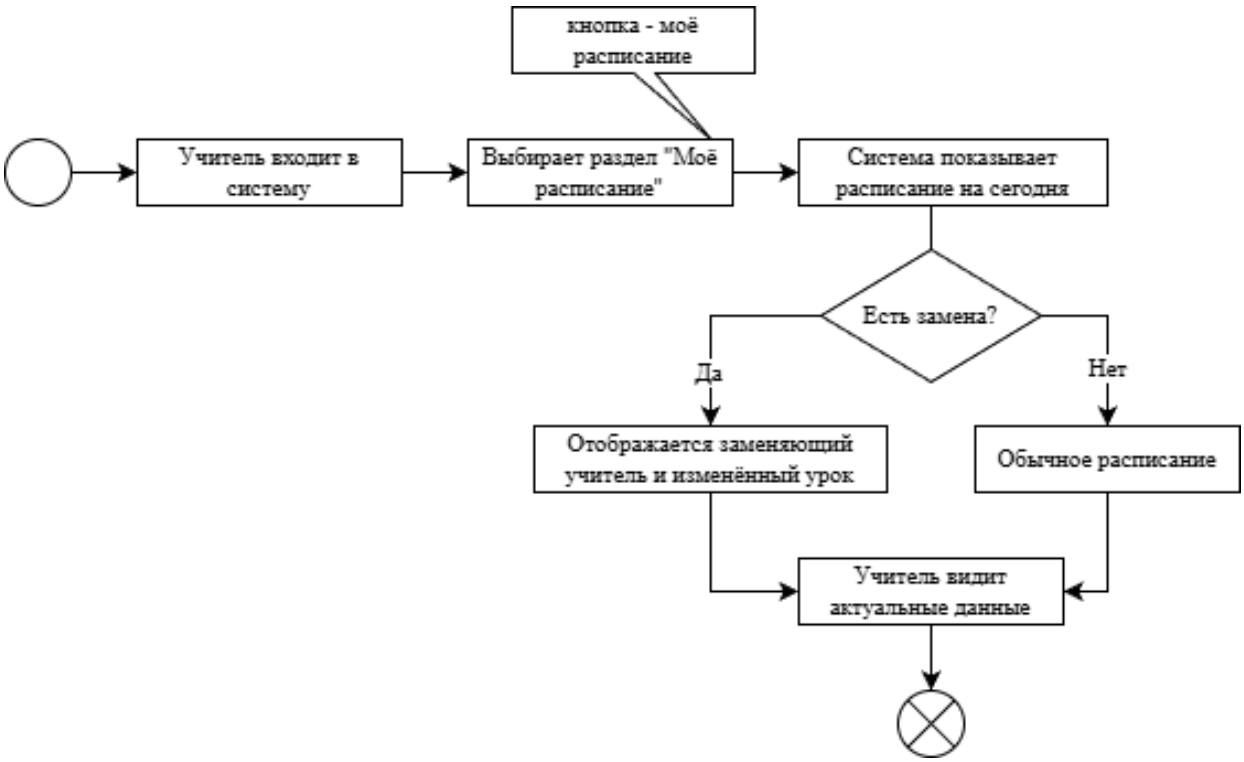


Рисунок 1.5 – Бизнес-процесс «Учитель. Просмотр расписания»

На рисунке 1.5 изображено, как учитель смотрит своё расписание. Он входит в систему, выбирает раздел «Моё расписание» (кнопка «Моё расписание»), и система показывает расписание на текущий день. Если на какой-то урок назначен заменяющий учитель, система отображает это — показывает заменяющего учителя и изменённый урок. Если замен нет, показывается обычное расписание. В любом случае учитель видит актуальные данные.

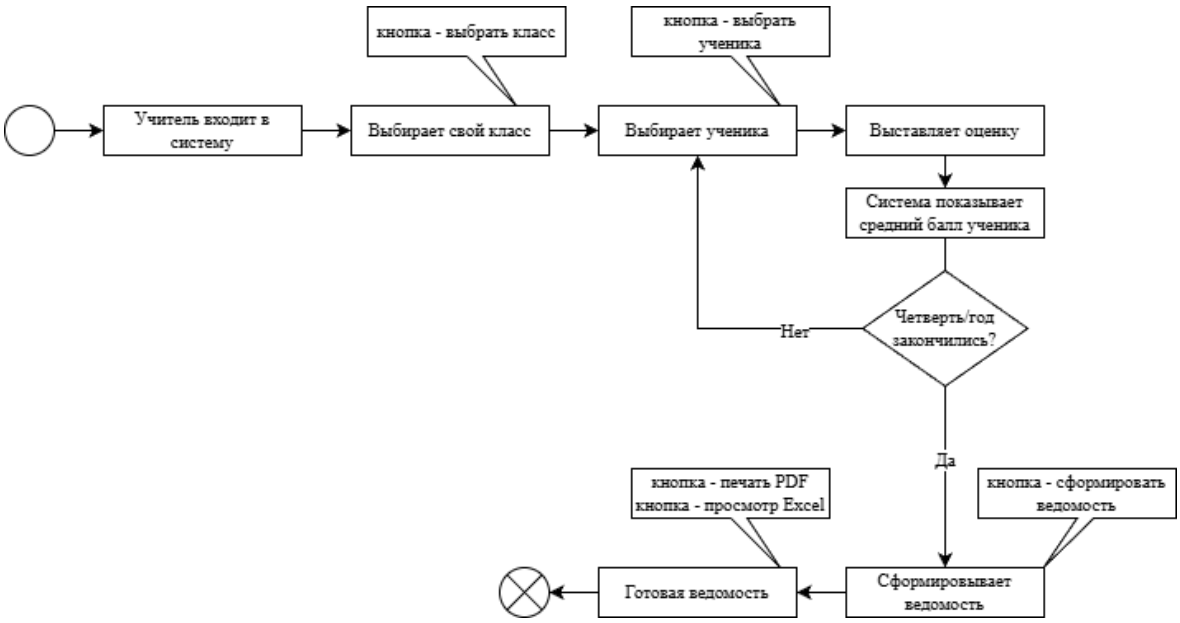


Рисунок 1.6 – «Учитель. Процесс выставления оценок»

На рисунке 1.6 представлен процесс работы учителя с оценками. Учитель входит в систему, выбирает свой класс, затем конкретного ученика и выставляет оценку. Система сразу показывает средний балл ученика. Если четверть или учебный год ещё не закончились, процесс повторяется для других учеников или следующих оценок. Когда четверть или год завершаются, учитель нажимает кнопку «Сформировать ведомость», после чего система создаёт итоговую ведомость. Далее учитель может распечатать её в формате PDF или просмотреть в Excel.

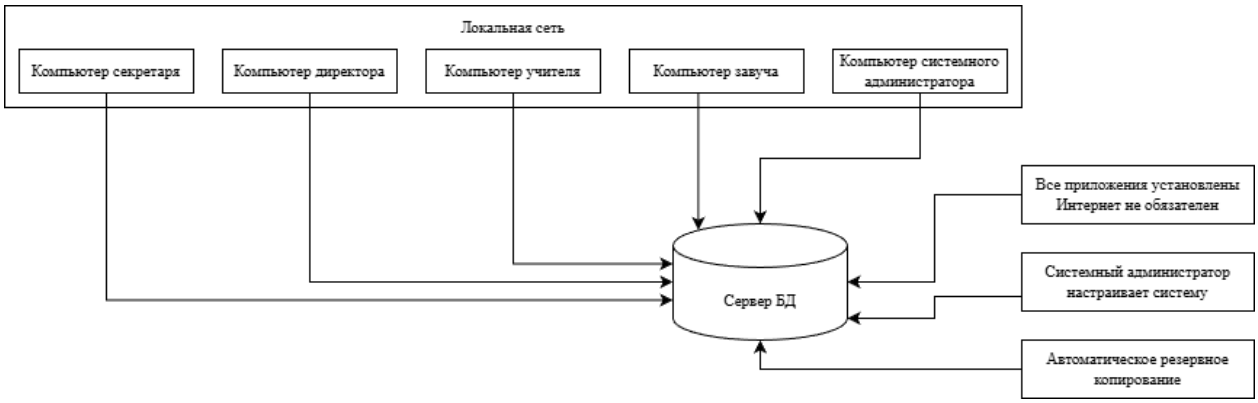


Рисунок 1.7 – Инфраструктурная схема

На рисунке 1.7 изображена локальная сеть образовательного учреждения. В сети присутствуют компьютеры секретаря, директора, учителя, завуча и системного администратора, а также отдельный сервер базы данных. Все приложения установлены локально, подключение к интернету не является обязательным. Системный администратор выполняет начальную настройку системы, а также обеспечивает автоматическое резервное копирование данных.

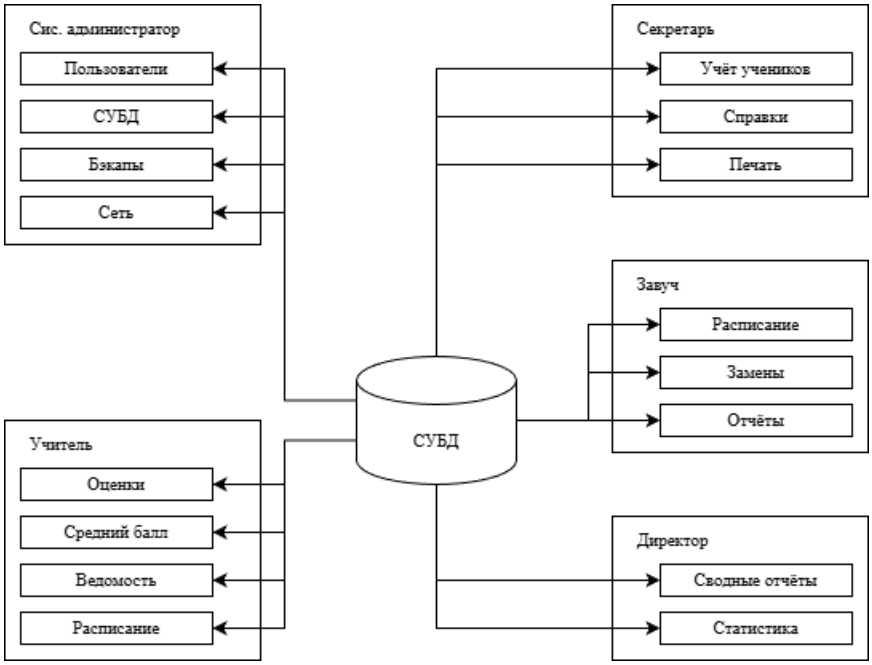


Рисунок 1.8 – Сквозная схема

На рисунке 1.8 показано взаимодействие всех ролей через сеть и единую систему управления базами данных. Системный администратор отвечает за пользователей, СУБД и резервное копирование. Секретарь ведёт учёт учеников, выдаёт справки и печатает документы. Учитель работает с оценками, средним баллом, ведомостями и расписанием. Завуч управляет расписанием, заменами и отчётами. Директор получает сводные отчёты и статистику. Все данные хранятся и обрабатываются централизованно через СУБД.

Все вместе эти схемы и бизнес-процессы описывают полную школьную информационную систему с ролевой моделью доступа, работающую в локальной сети без обязательного подключения к интернету. Система позволяет управлять пользователями и базой данных, вести учёт учеников, выставлять оценки, формировать ведомости, управлять расписанием и заменами, а также готовить отчёты для руководства. Каждая роль (секретарь, завуч, директор, учитель, системный администратор) имеет строго определённые функции, что обеспечивает порядок и безопасность работы.

2 ПРОЕКТИРОВАНИЕ ПЛАТФОРМЫ

В данной главе на основе требований и технологического стека (PostgreSQL, AvaloniaUI, .NET) разрабатывается модель платформы: создаются концептуальная и логическая модели данных, обосновывается выбор СУБД и строится полный набор UML-диаграмм, визуализирующих структуру и поведение системы.

2.1 Разработка концептуальной модели данных и описание сущностей

Концептуальное проектирование — это начальный этап создания базы данных, целью которого является построение формализованной модели предметной области, полностью независимой от особенностей конкретной системы управления базами данных (СУБД) и физической среды хранения. Результатом данного этапа является концептуальная модель, которая отражает множество сущностей предметной области, их атрибуты (свойства) и семантические связи между ними. Концептуальная модель служит единым источником информации для всех последующих этапов проектирования и обеспечивает однозначное понимание структуры данных между аналитиками, разработчиками и заказчиками [11,14].

В ходе анализа административной деятельности школы были выделены следующие сущности (объекты, информация о которых должна храниться в системе):

1. Сотрудник (Sotrudnik). Данная сущность представляет всех работников школы: администрацию (директор, завуч), секретаря, учителей, технический персонал. Атрибуты: табельный номер (первичный ключ), фамилия, имя, отчество, дата рождения, должность, паспортные данные, СНИЛС, контактный телефон, адрес электронной почты, логин и хеш пароля для

входа в систему. Сущность является центральной для подсистемы кадрового учета и аутентификации пользователей.

2. Ученик (Uchenik). Представляет всех обучающихся в школе. Атрибуты: идентификатор (первичный ключ), фамилия, имя, отчество, дата рождения, паспортные данные или свидетельство о рождении, адрес проживания, контактные данные родителей (телефон, email), дата поступления в школу. Данная сущность является центральной для учета контингента и формирования справок.

3. Класс (Klass). Обозначает учебную группу учащихся, объединенных одной параллелью и программой обучения. Атрибуты: идентификатор (первичный ключ), буквенное или цифровое наименование (например, «10А»), год формирования, идентификатор классного руководителя (внешний ключ к сущности «Сотрудник»), идентификатор учебного плана (внешний ключ).

4. Учебный предмет (UchebniyPredmet). Справочная сущность, содержащая перечень дисциплин, преподаваемых в школе. Атрибуты: идентификатор (первичный ключ), наименование предмета (например, «Математика», «Русский язык»), сокращенное наименование для отчетов, базовое количество часов в неделю.

5. Расписание (Raspisanie). Ключевая сущность, организующая учебный процесс. Определяет, какой учитель, в каком классе, какой предмет и когда проводит. Атрибуты: идентификатор (первичный ключ), день недели, номер урока, идентификатор класса (внешний ключ), идентификатор предмета (внешний ключ), идентификатор учителя (внешний ключ к «Сотрудник»), номер кабинета. Наличие внешних ключей обеспечивает ссылочную целостность.

6. Журнал успеваемости (ZhurnalUspevaemosti). Хранит результаты учебной деятельности учащихся: текущие и итоговые оценки, отметки о

посещаемости. Атрибуты: идентификатор (первичный ключ), идентификатор ученика (внешний ключ), идентификатор урока (внешний ключ к «Расписание»), оценка, тип оценки (текущая, четвертная, годовая), дата урока, отметка о присутствии.

Для визуализации взаимосвязей между перечисленными сущностями была построена ER-диаграмма (Entity-Relationship diagram) [11] — нотация, используемая для графического представления концептуальной модели данных. На ER-диаграмме сущности изображаются в виде прямоугольников с названиями, а связи — в виде линий между ними, на концах которых указывается мощность связи.

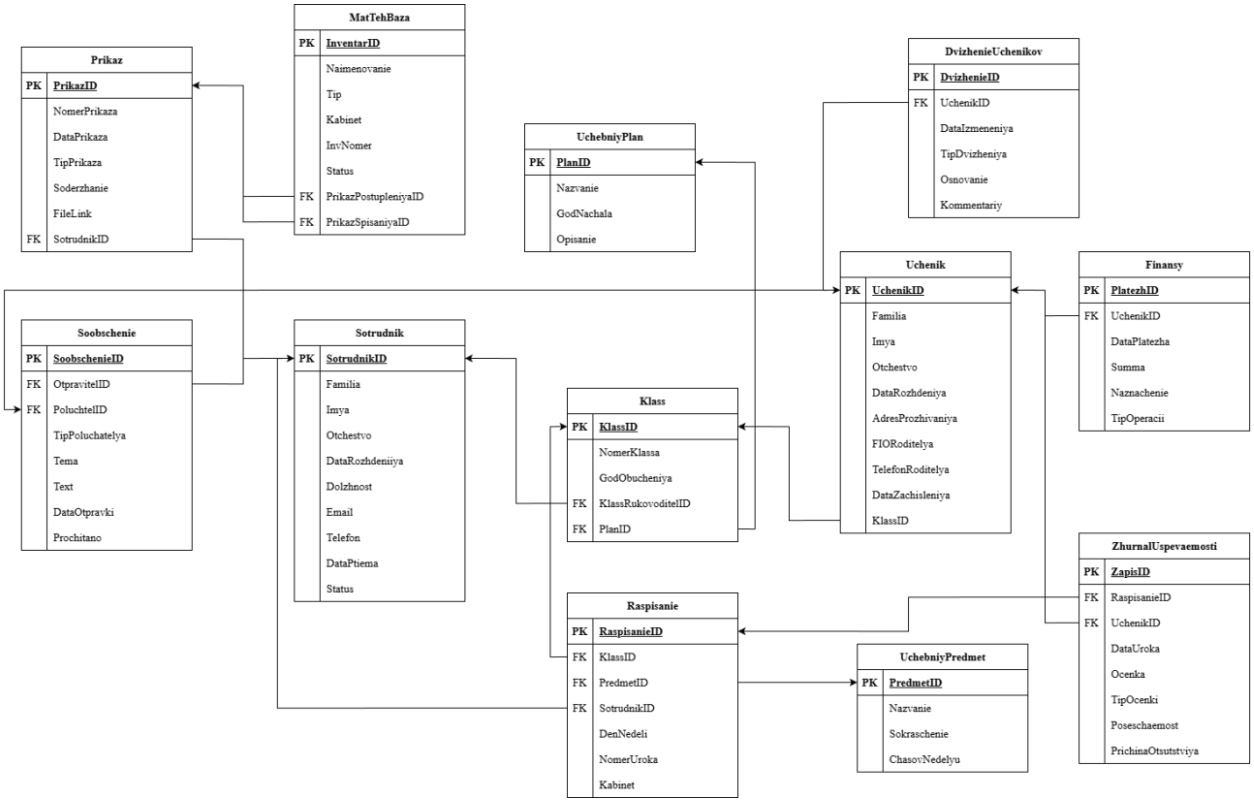


Рисунок 2.1 — ER-диаграмма базы данных

Как показано на рисунке 2.1, сущность «Сотрудник» связана с сущностью «Класс» (как классный руководитель) отношением «один ко многим» (один сотрудник может руководить несколькими классами).

Сущность «Класс» связана с сущностью «Ученик» отношением «один ко многим» (в одном классе учится много учеников). Сущность «Расписание» выступает связующим звеном между «Классом», «Учебным предметом» и «Сотрудником» (учителем), образуя сложную сеть связей типа «многие ко многим». Сущность «Журнал успеваемости» связывает «Ученика» и «Расписание», фиксируя результат каждого конкретного урока для каждого конкретного ученика. Выделенные сущности и их связи полностью покрывают функциональные требования к учету контингента, кадров, расписания и успеваемости.

2.2 Логическое проектирование базы данных

После построения концептуальной модели осуществляется переход к логическому проектированию, в ходе которого концептуальная модель преобразуется в структуру таблиц реляционной базы данных с учётом правил нормализации. Нормализация — это процесс разбиения сложных таблиц на более мелкие и связанные для устранения избыточности данных, аномалий обновления и обеспечения целостности [11,14]. В данном проекте применяется третья нормальная форма (3НФ), которая является стандартом для большинства транзакционных систем.

2.2.1 Приведение к третьей нормальной форме (3НФ)

Приведение к 3НФ включает последовательное выполнение следующих требований:

— Первая нормальная форма (1НФ): Все атрибуты атомарны (неделимы), каждая запись уникальна. В проектируемой таблице «Ученик» атрибут

«ФИО» был разбит на три отдельных поля: «Фамилия», «Имя», «Отчество», что удовлетворяет 1НФ.

— Вторая нормальная форма (2НФ): Таблица находится в 1НФ, и все неключевые атрибуты зависят от полного первичного ключа (для таблиц с составным ключом). Например, в таблице «Журнал успеваемости» составной ключ (ID_урока, ID_ученика), а атрибут «Оценка» зависит от этого составного ключа полностью.

— Третья нормальная форма (3НФ): Таблица находится в 2НФ, и отсутствуют транзитивные зависимости (неключевые атрибуты не зависят от других неключевых атрибутов). Вместо хранения названия класса («10А») в таблице «Ученик» используется внешний ключ ID_класса, который ссылается на таблицу «Класс», где хранится название. Таким образом, устранена транзитивная зависимость.

2.2.2 Описание структуры таблиц базы данных

Ниже представлено детальное описание таблиц 2.1 - 2.12, входящих в логическую модель данных. Для каждой таблицы указаны: назначение, список атрибутов с типами данных, признак NULL, первичные и внешние ключи, а также примечания.

Таблица 2.1 – Ученик

Наименование	Атрибут	Тип данных	Ключ	Описание
ID ученика	UcheníkID	Число	PK	Уникальный идентификатор
Фамилия	Família	Текст	—	Фамилия
Имя	Innya	Текст	—	Имя
Отчество	Otchestvo	Текст	—	Отчество
Дата рождения	DataRozhdeniya	Дата	—	Дата рождения
Адрес проживания	AdresProzhivaniya	Текст	—	Адрес
ФИО родителя	FIORoditelya	Текст	—	ФИО родителя
Телефон родителя	TelefonRoditelya	Текст	—	Контактный телефон
Дата зачисления	DataZachisleniya	Дата	—	Дата поступления
ID класса	KlassID	Число	FK (Klass)	Текущий класс

Хранит персональные данные учащихся: ID ученика (PK), фамилия, имя, отчество, дата рождения, адрес проживания, ФИО родителя, телефон родителя, дата зачисления, ID класса (FK → Класс).

Таблица 2.2 – Класс

Наименование	Атрибут	Тип данных	Ключ	Описание
ID класса	KlassID	Число	PK	Уникальный идентификатор
Номер класса	NomerKlassa	Текст	UK	«10А», «5Б»
Год обучения	GodObucheniya	Число	—	Учебный год
ID классного руководителя	KlassRukovoditelID	Число	FK (Sotrudník)	Классный руководитель
ID плана	PlanID	Число	FK (UchebnýPlan)	Учебный план класса

Содержит информацию об учебных группах: ID класса (PK), номер класса (UK), год обучения, ID классного руководителя (FK → Сотрудник), ID плана (FK → Учебный план).

Таблица 2.3 – Учебный предмет

Наименование	Атрибут	Тип данных	Ключ	Описание
ID предмета	PredmetID	Число	PK	Уникальный идентификатор
Название	Nazvanie	Текст	UK	«Математика», «Русский язык»
Сокращение	Sokraschenie	Текст	—	«мат.», «рус.»
Часов в неделю	ChasovNedelyu	Число	—	Количество часов в неделю

Справочник дисциплин: ID предмета (PK), название (UK), сокращение, количество часов в неделю.

Таблица 2.4 – Расписание

Наименование	Атрибут	Тип данных	Ключ	Описание
ID расписания	RapsisanieID	Число	PK	Уникальный идентификатор
ID класса	KlassID	Число	FK (Klass)	Класс
ID предмета	PredmetID	Число	FK (UchebnýPredmet)	Предмет
ID сотрудника	SotrudníkID	Число	FK (Sotrudník)	Учитель
День недели	DenNedeli	Число	—	1–6 (пн–сб)
Номер урока	NomerUroka	Число	—	1–8
Кабинет	Kabinet	Текст	—	Номер кабинета

Организует учебный процесс: ID расписания (PK), ID класса (FK → Класс), ID предмета (FK → Учебный предмет), ID сотрудника-учителя (FK → Сотрудник), день недели, номер урока, кабинет.

Таблица 2.5 – Журнал успеваемости

Наименование	Атрибут	Тип данных	Ключ	Описание
ID записи	ZapisID	Число	PK	Уникальный идентификатор
ID расписания	RapsisanieID	Число	FK (Rapsisanie)	Ссылка на урок
ID ученика	UcheníkID	Число	FK (Ucheník)	Ученик
Дата урока	DataUroka	Дата	—	Фактическая дата
Оценка	Ocenka	Число	—	2,3,4,5 (NULL — нет)
Тип оценки	TipOcenki	Текст	—	«текущая», «четвертная»
Посещаемость	Poseshanemost	Текст	—	присутствовал/отсутствовал
Причина отсутствия	PrichinaOtustitviya	Текст	—	«болезнь», «семейные обстоятельства»

Фиксирует оценки и посещаемость: ID записи (PK), ID расписания (FK → Расписание), ID ученика (FK → Ученик), дата урока, оценка, тип оценки, посещаемость, причина отсутствия.

Таблица 2.6 – Движение учеников

Наименование	Атрибут	Тип данных	Ключ	Описание
ID движения	DvízhenieID	Число	PK	Уникальный идентификатор
ID ученика	UcheníkID	Число	FK (Ucheník)	Ученик
Дата изменения	DataZmeneniya	Дата	—	Дата перевода/выбытия
Тип движения	TipDvízheniya	Текст	—	«перевод», «отчисление», «окончание»
Основание	Osnovanie	Текст	—	Приказ, заявление родителей
Комментарий	Kommentariy	Текст	—	Дополнительная информация

Учитывает перемещения учащихся: ID движения (PK), ID ученика (FK → Ученик), дата изменения, тип движения (перевод/отчисление/окончание), основание, комментарий.

Таблица 2.7 – Приказ

Наименование	Атрибут	Тип данных	Ключ	Описание
ID приказа	PríkazID	Число	PK	Уникальный идентификатор приказа
Номер приказа	NomerPríkaz	Текст	UK	Исходящий номер приказа
Дата приказа	DataPríkaz	Дата	—	Дата подписания
Тип приказа	TipPríkaz	Текст	—	«О зачислении», «Об отчислении»
Содержание	Soderzhanie	Текст	—	Краткое содержание
Ссылка на файл	FileLink	Текст	—	Путь к файлу с приказом
ID сотрудника	SotrudníkID	Число	FK (Sotrudník)	Кого касается приказ

Хранит распорядительные документы: ID приказа (PK), номер приказа (UK), дата приказа, тип приказа, содержание, ссылка на файл, ID сотрудника (FK → Сотрудник).

Таблица 2.8 – Учебный план

Наименование	Атрибут	Тип данных	Ключ	Описание
ID плана	PlanID	Число	PK	Уникальный идентификатор
Название	Nazvanie	Текст	UK	«ФГОС 5-9 классы»
Год начала	GodNachala	Число	—	Год внедрения плана
Описание	Opisanie	Текст	—	Подробное описание

Определяет образовательные программы: ID плана (PK), название (UK), год начала, описание.

Таблица 2.9 – Сообщения

Наименование	Атрибут	Тип данных	Ключ	Описание
ID сообщения	SoobschenieID	Число	PK	Уникальный идентификатор
ID отправителя	OtpravitelID	Число	FK (Sotrudník)	Кто отправил
ID получателя	PoluchetID	Число	—	ID получателя
Тип получателя	TipPoluchetela	Текст	—	«сотрудник», «ученик», «родитель»
Тема	Tema	Текст	—	Тема сообщения
Текст	Text	Текст	—	Содержание
Дата отправки	DataOtpravki	ДатаВремя	—	Дата и время
Прочитано	Prochitano	Текст	—	Статус прочтения

Обеспечивает внутренние уведомления: ID сообщения (PK), ID отправителя (FK → Сотрудник), ID получателя, тип получателя, тема, текст, дата отправки, статус прочтения.

Таблица 2.10 – Финансы

Наименование	Атрибут	Тип данных	Ключ	Описание
ID платежа	PlatzaID	Число	PK	Уникальный идентификатор
ID ученика	UcheníkID	Число	FK (Ucheník)	За кого платёж
Дата платежа	DataPlatzeha	Дата	—	Дата оплаты
Сумма	Summa	Число	—	Сумма в рублях
Назначение	Naznačenie	Текст	—	«оплата питания», «взносы»
Тип операции	TipOperacii	Текст	—	«приход», «расход»

Ведёт учёт платежей: ID платежа (PK), ID ученика (FK → Ученик), дата платежа, сумма, назначение, тип операции (приход/расход).

Таблица 2.11 – Материально-техническая база

Наименование	Атрибут	Тип данных	Ключ	Описание
ID инвентаря	InventarID	Число	PK	Уникальный идентификатор
Наименование	Námenovanie	Текст	—	Название оборудования
Тип	Tip	Текст	—	«компьютер», «парта», «проектор»
Кабинет	Kabinet	Текст	—	Номер кабинета
Инвентарный номер	InvNomer	Текст	UK	Уникальный инвентарный номер
Статус	Status	Текст	—	«в эксплуатации», «списан»
ID приказа поступления	PríkazPostupneniaID	Число	FK (Príkaz)	Приказ о принятии на баланс
ID приказа списания	PríkazSpisovaniaID	Число	FK (Príkaz)	Приказ о списании

Учитывает школьное имущество: ID инвентаря (PK), наименование, тип, кабинет, инвентарный номер (UK), статус, ID приказа поступления (FK → Приказ), ID приказа списания (FK → Приказ).

Таблица 2.12 – Сотрудники

Наименование	Атрибут	Тип данных	Ключ	Описание
ID сотрудника	SotrudníkID	Число	PK	Уникальный идентификатор
Фамилия	Família	Текст	—	Фамилия
Имя	Innya	Текст	—	Имя
Отчество	Otchestvo	Текст	—	Отчество
Дата рождения	DataRozhdeniya	Дата	—	Дата рождения
Должность	Dokladnost	Текст	—	Учитель, завуч, директор
Email	Email	Текст	UK	Электронная почта
Телефон	Telefon	Текст	—	Контактный телефон
Дата приёма	DataPriam	Дата	—	Дата трудоустройства
Статус	Status	Текст	—	«работает», «уволен»

Хранит данные о персонале: ID сотрудника (PK), фамилия, имя, отчество, дата рождения, должность, email (UK), телефон, дата приёма, статус (работает/уволен).

2.3 Выбор СУБД

Система управления базами данных (СУБД) — это совокупность программных средств, обеспечивающих управление созданием, хранением, модификацией и извлечением данных из базы данных, а также контроль целостности, безопасности и многопользовательского доступа. Выбор СУБД является критическим архитектурным решением, которое влияет на производительность, надежность, стоимость владения и возможности масштабирования системы.

Для разрабатываемой платформы выбрана PostgreSQL по следующим обоснованиям:

1. Бесплатная и открытая модель распространения (Open Source). Лицензия PostgreSQL не требует лицензионных отчислений, что позволяет школе избежать финансовых затрат на приобретение коммерческой СУБД (например, MS SQL Server Standard). В отличие от бесплатной редакции MS SQL Server Express, PostgreSQL не имеет искусственных ограничений на размер базы данных (максимум 10 ГБ у

Express) и объем используемой оперативной памяти. Для школы с многолетней историей оперативной деятельности ограничения Express могут стать критичными [23].

2. Соответствие требованиям ACID. PostgreSQL полностью поддерживает принципы ACID (Атомарность, Согласованность, Изоляция, Долговечность). Это означает, что даже при сбоях питания или ошибках системы, транзакции (например, одновременное выставление оценки и пересчет среднего балла) либо будут выполнены полностью, либо откачены к исходному состоянию. Целостность данных об успеваемости и контингенте учащихся гарантирована на уровне СУБД [23].

3. Высокая производительность и оптимизатор запросов. PostgreSQL обладает одним из самых совершенных оптимизаторов запросов среди реляционных СУБД. Он эффективно обрабатывает сложные аналитические запросы, такие как формирование сводных отчетов директору (средний балл по школе за год) или расчет нагрузки учителя (количество замен). Индексация по полям ФИО, ID_класса и дата позволяет выполнять поиск данных в таблицах с миллионами записей за доли секунды [23].

4. Нативная поддержка работы с Entity Framework Core. Разрабатываемое приложение на C# и AvaloniaUI использует объектно-реляционный маппер Entity Framework Core. PostgreSQL имеет официального провайдера для EF Core (Npgsql), который обеспечивает полную совместимость: автоматическое создание таблиц, миграции при изменении модели, отслеживание изменений объектов. Это делает код разработчика полностью независимым от конкретной СУБД. Более того, если школа захочет использовать MS SQL Server, достаточно будет изменить строку подключения и провайдера в конфигурационном файле — остальной код останется без изменений [26].

5. Кроссплатформенность и надежность. PostgreSQL работает на всех основных операционных системах (Windows, Linux, macOS). В отличие от MS SQL Server, который исторически привязан к Windows, PostgreSQL может быть развернут на бесплатной серверной версии Linux, что дополнительно экономит средства школы на лицензировании операционной системы. Высокая надежность PostgreSQL подтверждена многолетним использованием в банковской сфере и государственных учреждениях США и Европы [23].

Таким образом, выбор PostgreSQL обусловлен оптимальным сочетанием нулевой стоимости владения, неограниченной масштабируемости, строгого соблюдения ACID и плотной интеграции с выбранным технологическим стеком C#/EF Core. Это позволяет школе получить корпоративную систему без корпоративных расходов.

2.4 Разработка UML-диаграмм проектирования

Unified Modeling Language (UML) — это унифицированный язык графического моделирования, предназначенный для описания, визуализации, проектирования и документирования объектно-ориентированных программных систем [11,13]. UML предоставляет набор стандартных диаграмм, каждая из которых отражает различные аспекты проектируемой системы: статическую структуру (классы, компоненты) и динамическое поведение (процессы, взаимодействие объектов). В рамках данного дипломного проекта разработан полный набор диаграмм, адаптированный под специфику платформы автоматизации школы.

Диаграмма вариантов использования — это диаграмма UML, которая отображает функциональные требования к системе с точки зрения внешних пользователей (акторов). Основные элементы: актор (роль пользователя)

и вариант использования (функция, которую система предоставляет актору). Диаграмма не показывает строгую последовательность действий, а демонстрирует границы системы и состав ее функций.

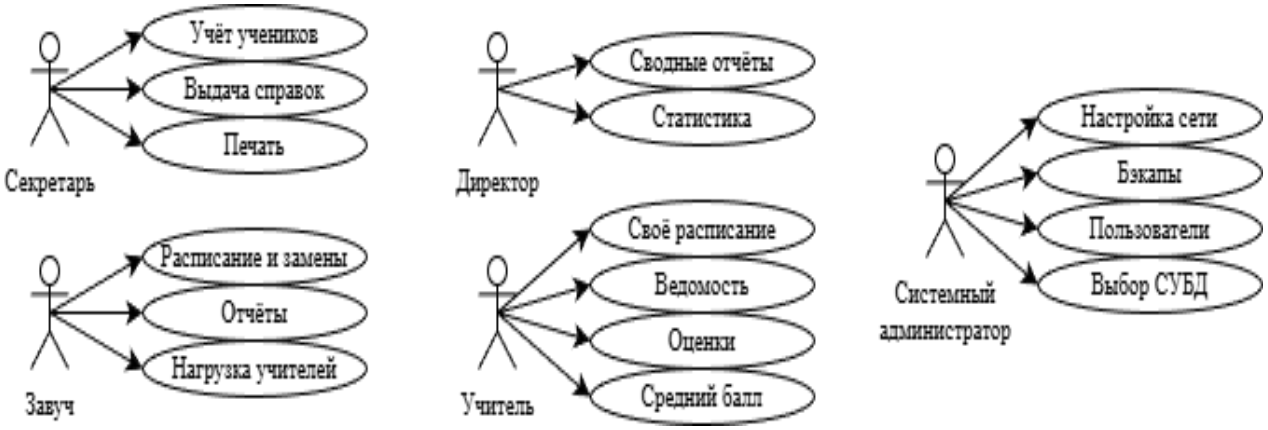


Рисунок 2.2 - Диаграмма вариантов использования

Приведённая диаграмма вариантов использования описывает школьную информационную систему, в которой выделено пять акторов. Секретарь отвечает за учёт учеников, выдачу справок и печать документов. Завуч работает с расписанием и заменами, формирует отчёты по успеваемости и управляет нагрузкой учителей. Директор имеет доступ к сводным отчётам и общей статистике по школе. Учитель может просматривать своё расписание, формировать ведомость по предмету, выставлять оценки и следить за средним баллом учеников. Системный администратор настраивает сеть, создаёт резервные копии, управляет пользователями и выбирает тип СУБД. Все функции строго разграничены по ролям и реализуются через единую систему с общей базой данных.

В таблицах 2.13 – 2.26 представлено краткое описание каждого варианта использования.

Таблица 2.13 - Краткое описание варианта использования «Учёт учеников»

Элемент	Описание
Название	Учёт учеников
Актор	Секретарь
Цель	Осуществление приёма, перевода и выбытия учащихся из школы
Основной сценарий	Секретарь открывает раздел учёта учеников → выбирает действие (приём/перевод/выбытие) → вводит данные ученика (ФИО, класс, дата рождения и т.д.) → система сохраняет изменения → обновляется контингент учащихся
Альтернативный сценарий	При попытке добавить уже существующего ученика система выводит сообщение «Ученик уже зарегистрирован» → секретарь корректирует данные
Результат	Данные ученика добавлены, изменены или удалены в системе

Таблица 2.14 - Краткое описание варианта использования «Выдача справок»

Элемент	Описание
Название	Выдача справок
Актор	Секретарь
Цель	Формирование и выдача официальной справки об обучении учащегося
Основной сценарий	Секретарь выбирает ученика → нажимает «Сформировать справку» → выбирает тип справки → система заполняет данные из базы → секретарь проверяет и отправляет на печать
Альтернативный сценарий	Данные ученика неполные или отсутствуют → система запрашивает ввод недостающей информации
Результат	Распечатанная и заверенная справка выдана на руки или сохранена в системе

Таблица 2.15 - Краткое описание варианта использования «Печать документов»

Элемент	Описание
Название	Печать документов
Актор	Секретарь
Цель	Вывод на печать различных типов документов (личные дела, справки, приказы)
Основной сценарий	Секретарь открывает документ в системе → нажимает кнопку «Печать» → выбирает параметры печати → система отправляет документ на принтер
Альтернативный сценарий	Принтер не отвечает → система выводит сообщение об ошибке → секретарь может сохранить документ в PDF для последующей печати
Результат	Документ распечатан на бумажном носителе

Таблица 2.16 - Краткое описание варианта использования «Отчёты по успеваемости»

Элемент	Описание
Название	Отчёты по успеваемости
Актор	Завуч
Цель	Формирование сводной информации об успеваемости по классам, предметам или периодам
Основной сценарий	Завуч выбирает раздел «Отчёты по успеваемости» → задаёт параметры (класс, предмет, четверть) → система формирует отчёт → завуч может его просмотреть, распечатать или экспортировать
Альтернативный сценарий	Нет оценок за выбранный период → система выводит сообщение «Нет данных»
Результат	Сформирован отчёт об успеваемости с указанием успеваемости и качества знаний

Таблица 2.17 - Краткое описание варианта использования «Нагрузка учителей»

Элемент	Описание
Название	Нагрузка учителей
Актор	Завуч
Цель	Просмотр и распределение учебной нагрузки между учителями
Основной сценарий	Завуч открывает раздел «Нагрузка учителей» → выбирает учителя → видит его часы по предметам и классам → при необходимости корректирует нагрузку → система сохраняет изменения
Альтернативный сценарий	Загрузка превышает допустимую норму → система выводит предупреждение → завуч подтверждает или корректирует нагрузку
Результат	Актуальная информация о нагрузке каждого учителя сохранена в системе

Таблица 2.18 - Краткое описание варианта использования «Сводные отчёты»

Элемент	Описание
Название	Сводные отчёты
Актор	Директор
Цель	Получение обобщённой информации о работе школы за произвольный период
Основной сценарий	Директор выбирает раздел «Сводные отчёты» → задаёт период и тип отчёта (успеваемость, движение учащихся, нагрузка и т.д.) → система формирует сводный отчёт с возможностью детализации
Альтернативный сценарий	Запрашиваемые данные отсутствуют → система выводит сообщение «Нет данных за указанный период»
Результат	Сформирован сводный отчёт для анализа и принятия управленческих решений

Таблица 2.19 - Краткое описание варианта использования «Статистика»

Элемент	Описание
Название	Статистика
Актор	Директор
Цель	Просмотр ключевых показателей деятельности школы в графическом или табличном виде
Основной сценарий	Директор открывает раздел «Статистика» → выбирает показатели (количество учеников, учителей, успеваемость и т.д.) → система визуализирует данные
Альтернативный сценарий	Данные для визуализации отсутствуют → система показывает пустые диаграммы с сообщением «Нет данных»
Результат	Интерактивная статистика с возможностью фильтрации и экспорта

Таблица 2.20 - Краткое описание варианта использования «Своё расписание»

Элемент	Описание
Название	Своё расписание
Актор	Учитель
Цель	Просмотр персонального расписания уроков на выбранный день или неделю
Основной сценарий	Учитель входит в систему → выбирает раздел «Моё расписание» → система отображает расписание на текущий день с учётом замен
Альтернативный сценарий	Учитель может выбрать другой день или неделю для просмотра → система загружает расписание за указанный период
Результат	Учитель видит актуальное расписание, включая замены

Таблица 2.21 - Краткое описание варианта использования «Ведомость по предмету»

Элемент	Описание
Название	Ведомость по предмету
Актор	Учитель
Цель	Формирование итоговой ведомости оценок по своему предмету за четверть или год
Основной сценарий	Учитель выбирает класс и предмет → нажимает «Сформировать ведомость» → система собирает все оценки учеников → формирует ведомость
Альтернативный сценарий	У некоторых учеников нет оценок за период → система помечает их как «нет аттестации»
Результат	Готовая ведомость в формате PDF или Excel для печати или архивации

Таблица 2.22 - Краткое описание варианта использования «Оценки и средний балл»

Элемент	Описание
Название	Оценки и средний балл
Актор	Учитель
Цель	Выставление текущих оценок и отслеживание среднего балла каждого ученика
Основной сценарий	Учитель выбирает ученика → выставляет оценку за урок → система автоматически пересчитывает и отображает средний балл ученика по предмету
Альтернативный сценарий	Попытка выставить оценку за прошедшую дату → система предупреждает об этом и фиксирует дату выставления
Результат	Оценка сохранена в журнале, средний балл ученика обновлён и отображается в интерфейсе

Таблица 2.23 - Краткое описание варианта использования «Настройка сети»

Элемент	Описание
Название	Настройка сети
Актор	Системный администратор
Цель	Конфигурирование сетевых параметров для доступа компьютеров к серверу БД
Основной сценарий	Администратор входит в раздел сетевых настроек → вводит IP-адрес сервера, порт, протокол → сохраняет настройки → система проверяет соединение
Альтернативный сценарий	Соединение не установлено → система выводит сообщение об ошибке → администратор проверяет параметры
Результат	Настроено сетевое взаимодействие всех рабочих станций с сервером

Таблица 2.24 - Краткое описание варианта использования «Резервное копирование»

Элемент	Описание
Название	Резервное копирование
Актор	Системный администратор
Цель	Создание резервной копии базы данных для восстановления в случае сбоя
Основной сценарий	Администратор выбирает раздел «Резервное копирование» → выбирает тип копии (полная/инкрементальная) → запускает процесс → система создаёт копию и уведомляет об успехе
Альтернативный сценарий	Недостаточно места для копии → система выводит ошибку → администратор освобождает место или выбирает другой носитель
Результат	Создана резервная копия БД с отметкой даты и времени

Таблица 2.25 - Краткое описание варианта использования «Выбор СУБД»

Элемент	Описание
Название	Выбор СУБД
Актор	Системный администратор
Цель	Настройка подключения системы к выбранной базе данных
Основной сценарий	При установке системы администратор выбирает тип СУБД → вводит параметры подключения (сервер, логин, пароль) → проверяет подключение → система сохраняет настройки
Альтернативный сценарий	Подключение не удалось → система сообщает об ошибке → администратор повторяет ввод данных
Результат	Система подключена к рабочей базе данных и готова к эксплуатации

Таблица 2.26 - Краткое описание варианта использования «Управление пользователями»

Элемент	Описание
Название	Управление пользователями
Актор	Системный администратор
Цель	Создание, редактирование и удаление учётных записей сотрудников
Основной сценарий	Администратор входит в раздел управления пользователями → добавляет нового пользователя (ФИО, логин, пароль, роль) → система сохраняет запись → пользователь может войти в систему
Альтернативный сценарий	При добавлении пользователя с уже существующим логином система выводит предупреждение → администратор изменяет логин
Результат	Учётная запись создана, изменена или удалена; доступ к системе определён в соответствии с ролью

Разработанные сценарии использования охватывают все функциональные возможности системы, представленные на диаграмме вариантов использования. Были подробно описаны действия для пяти акторов: секретаря (учёт учеников, выдача справок, печать), завуча (расписание и замены, отчёты, нагрузка учителей), директора (сводные отчёты, статистика), учителя (своё расписание, ведомость, оценки, средний балл) и системного администратора (настройка сети, бэкапы, пользователи, выбор СУБД). Каждый сценарий включает цель, основной и альтернативный сценарии, а также ожидаемый результат, что позволяет наглядно продемонстрировать функциональность системы с точки зрения конечных пользователей.

Диаграмма классов — это основной вид диаграммы статической структуры, которая описывает типы классов системы, их атрибуты, методы и связи между классами (наследование, ассоциация, композиция, агрегация). В контексте данного проекта диаграмма классов отражает объектную модель системы, которая затем будет реализована на языке C# (модели данных для Entity Framework Core).

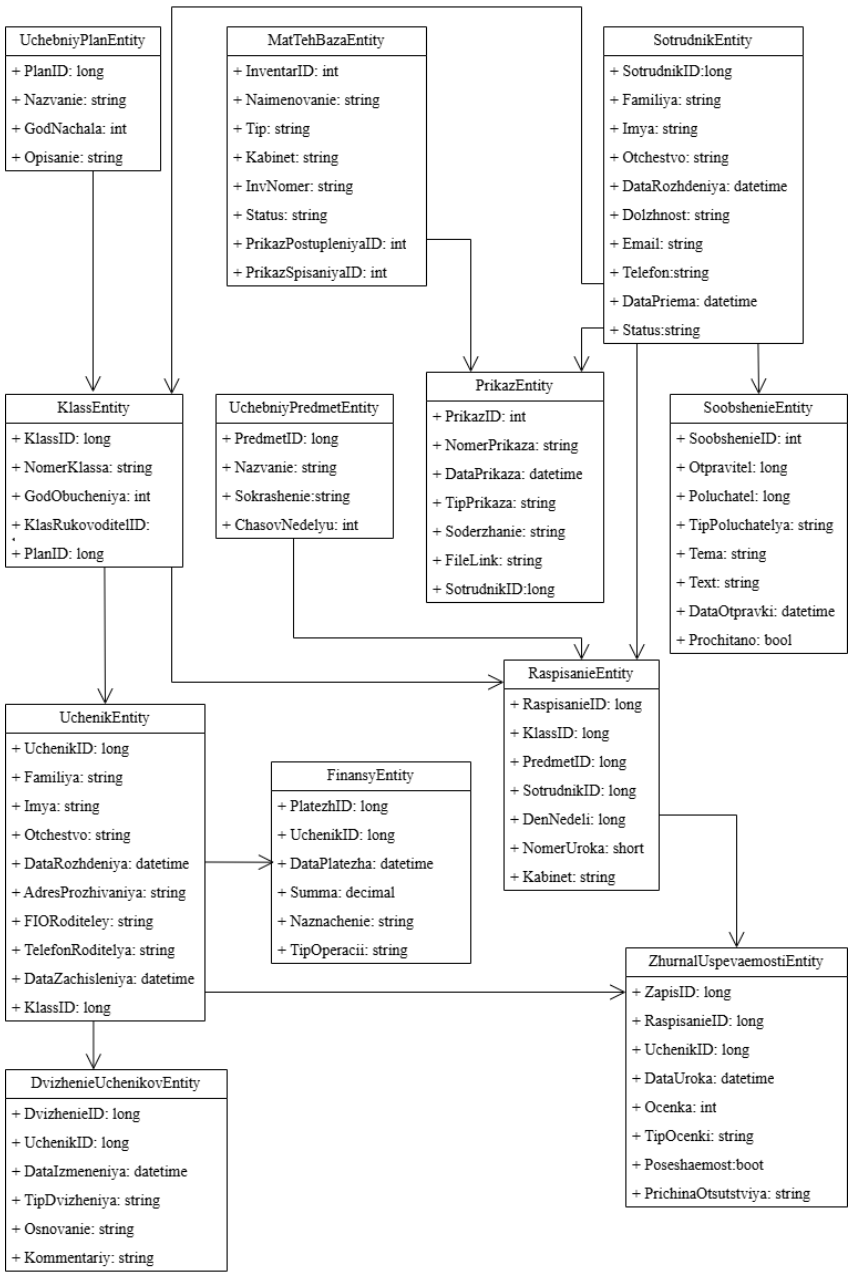


Рисунок 2.3 - Диаграмма классов

На рисунке 2.3 представлена диаграмма классов, отражающая объектную модель разрабатываемой системы автоматизации школы. В основе диаграммы лежат сущности предметной области, описанные в логической модели данных (раздел 2.2), которые затем были реализованы в виде классов C# с использованием Entity Framework Core (подход Code First).

Ключевые классы и их атрибуты. Центральными классами являются «Сотрудник» (SotrudnikEntity), «Ученик» (UchenikEntity), «Класс» (KlassEntity), «Расписание» (RaspisanieEntity) и «Журнал успеваемости» (ZhurnalUspevaemostiEntity). У каждого класса перечислены атрибуты с указанием типов данных (string, long, DateTime, decimal, bool), а также ассоциации (навигационные свойства), обозначенные как «1» или «0..*», что показывает кратность связей между объектами.

Ролевая модель и аутентификация. Класс «Сотрудник» содержит атрибуты для идентификации личности (Фамилия, Имя, Отчество, Дата рождения, Должность) и учётные данные (Email, Телефон). Атрибуты для хешированного пароля и логина в данной версии диаграммы опущены, однако они подразумеваются в логике приложения для реализации системы аутентификации. Разные роли (Директор, Завуч, Секретарь, Учитель) различаются значением атрибута «Должность» и соответствующими методами доступа к функциям системы.

Классы, связанные с учебным процессом. Класс «Ученик» содержит персональные данные, включая ФИО, дату рождения, адрес, ФИО родителей и телефон. Класс «Класс» включает номер класса (например, «10А»), учебный год и ссылку на классного руководителя (внешний ключ к «Сотруднику»). Класс «Расписание» связывает «Класс», «Учебный предмет» и «Сотрудника» (учителя), добавляя день недели, номер урока и кабинет. Класс «Журнал успеваемости» фиксирует оценки и посещаемость, связывая

«Расписание» и «Ученика», а также содержит атрибуты «Оценка», «Тип оценки», «Посещаемость» и «Причина отсутствия».

Вспомогательные классы. На диаграмме также присутствуют классы «Движение учеников» (история переводов и отчислений), «Финансы» (учёт платежей), «Приказ» (распорядительные документы), «Сообщения» (внутренний чат), «Материально-техническая база» (учёт имущества) и «Учебный план» (образовательные программы). Эти классы расширяют функциональность системы за пределы базового учёта успеваемости.

Общее назначение диаграммы. Данная диаграмма классов является итогом объектно-ориентированного проектирования и служит основой для генерации схемы базы данных через Entity Framework Core (подход Code First). Каждый класс на диаграмме соответствует таблице в PostgreSQL, а атрибуты — столбцам таблицы. Навигационные свойства (виртуальные коллекции и ссылки) реализуют внешние ключи и обеспечивают ссылочную целостность на уровне приложения.

Диаграмма пакетов — это диаграмма UML, которая показывает разбиение системы на логические блоки (пакеты) и зависимости между ними. Пакеты предназначены для группировки элементов модели с целью организации структуры проекта. В архитектуре программного обеспечения диаграмма пакетов помогает разделить систему на слои (уровни).

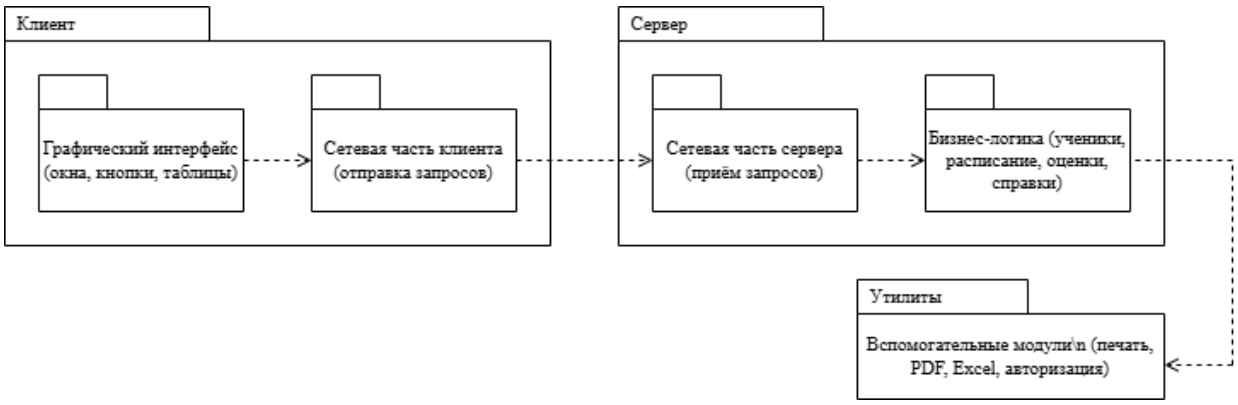


Рисунок 2.4 - Диаграмма пакетов

На рисунке 2.4 изображена архитектура школьной информационной системы в виде иерархически связанных пакетов (модулей). Диаграмма разделена на несколько крупных блоков, отражающих логические уровни системы.

В самом верху расположен пакет «Клиент», который содержит подсистему «Графический интерфейс» (окна, кнопки, таблицы). Это пользовательская часть системы, через которую все акторы (секретарь, завуч, директор, учитель, системный администратор) взаимодействуют с приложением.

Ниже находится пакет «Сервер», включающий в себя «Сетевую часть клиента» (отправка запросов). Этот пакет отвечает за передачу данных от клиентского приложения к серверной части и обратно.

Далее выделен отдельный пакет «Запросы к БД», который содержит непосредственно обращения к базе данных (SQL-запросы, вызовы хранимых процедур). Этот пакет служит прослойкой между бизнес-логикой и системой управления базами данных.

Отдельно на диаграмме представлен пакет «Утилиты», куда входят вспомогательные модули: печать, работа с PDF и Excel, а также модуль авторизации. Эти компоненты используются различными частями системы для выполнения сквозных задач.

Кроме того, выделен пакет «Бизнес-логика», который включает основные функциональные блоки: ученики, расписание, оценки, справки. Этот пакет реализует ключевые правила обработки данных и сценарии использования системы.

Таким образом, диаграмма пакетов показывает многоуровневую архитектуру «клиент – сервер – база данных», где графический интерфейс отделён от бизнес-логики, а обе части взаимодействуют через сетевой уровень и запросы к БД. Утилиты обеспечивают сквозные функции, такие

как печать и авторизация. Такая структура обеспечивает модульность, упрощает сопровождение и позволяет изменять отдельные компоненты без полной перекомпиляции системы.

Диаграмма последовательности — это диаграмма UML, которая детализирует варианты использования, показывая временную последовательность сообщений, которыми обмениваются объекты в процессе взаимодействия. Основная вертикальная ось — это время. Вертикальные прямоугольники («линии жизни») представляют объекты, а горизонтальные стрелки — сообщения.

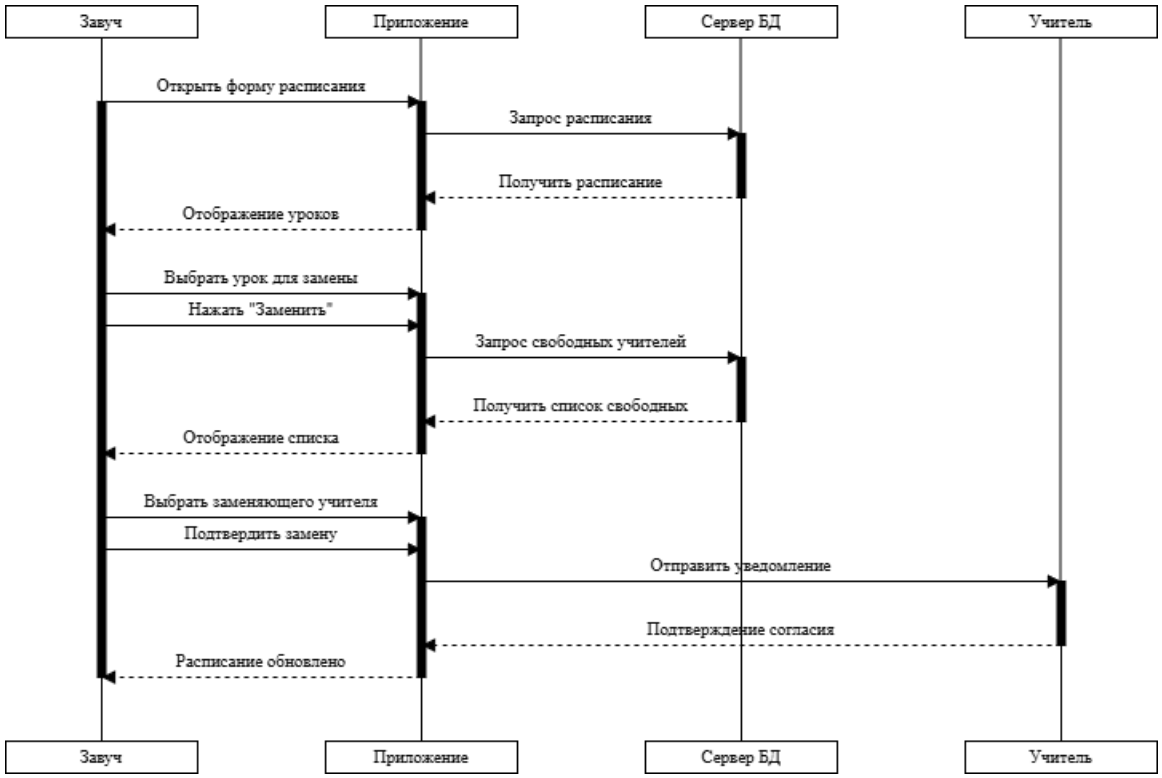


Рисунок 2.5 - Диаграмма последовательности «Завуч. Замена учителя»

На рисунке 2.5 изображено взаимодействие четырёх объектов: Завуч, Приложение, Сервер БД и Учитель. Временная ось идёт сверху вниз.

— Шаг 1. Завуч инициирует процесс, открывая раздел «Расписание» в приложении.

— Шаг 2. Приложение отправляет запрос к серверу БД на получение списка классов и уроков на текущий день.

— Шаг 3. Сервер БД возвращает запрошенные данные, и приложение отображает их завучу.

— Шаг 4. Завуч выбирает конкретный класс и урок, затем нажимает кнопку «Заменить учителя».

— Шаг 5. Приложение формирует запрос к серверу БД: «Найти свободных учителей на данное время».

— Шаг 6. Сервер БД выполняет поиск по расписанию и возвращает список свободных учителей.

— Шаг 7. Завуч выбирает из списка подходящего учителя и подтверждает замену.

— Шаг 8. Приложение отправляет серверу БД команду на обновление расписания (фиксирует замену) и создание уведомления для выбранного учителя.

— Шаг 9. Сервер БД сохраняет изменения и отправляет уведомление учителю через приложение.

— Шаг 10. Учитель получает уведомление (например, «Вас заменили на уроке 3 в 10А»). Если учитель подтверждает согласие (в некоторых вариантах процесса), приложение передаёт подтверждение обратно на сервер.

— Шаг 11. После подтверждения сервер БД окончательно фиксирует замену, и обновлённое расписание становится доступным для всех участников.

Таким образом, диаграмма наглядно показывает, как завуч является инициатором, приложение — посредником, сервер БД — хранилищем и логическим центром, а учитель — конечным получателем уведомления.

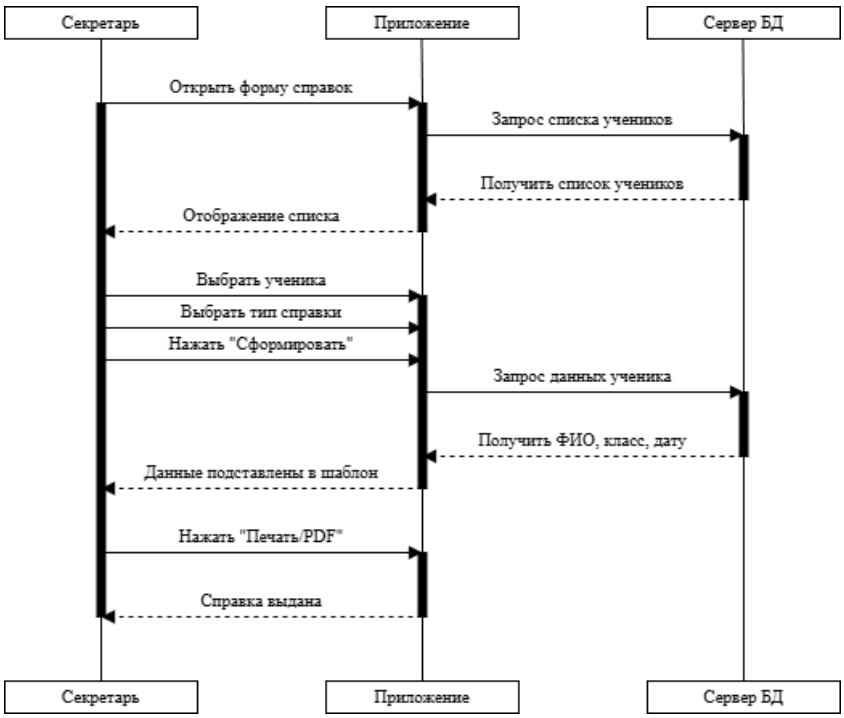


Рисунок 2.6 - Диаграмма последовательности «Секретарь. Выдача справок»

На рисунке 2.6 также участвуют три объекта: Секретарь, Приложение и Сервер БД.

— Шаг 1. Секретарь открывает в приложении раздел «Выдача справок» или «Справки».

— Шаг 2. Приложение отображает форму справки.

— Шаг 3. Секретарь вводит фамилию ученика (или выбирает из списка). Приложение отправляет запрос к серверу БД: «Получить данные ученика».

— Шаг 4. Сервер БД возвращает информацию: ФИО, класс, дата рождения, период обучения и т.д.

— Шаг 5. Приложение автоматически заполняет поля справки этими данными. Секретарь может дополнительно указать тип справки (об обучении, о составе семьи и т.п.).

— Шаг 6. Секретарь нажимает кнопку «Сформировать справку». Приложение отправляет серверу БД запрос на создание документа (или генерацию PDF).

— Шаг 7. Сервер БД формирует файл справки и возвращает его в приложение.

— Шаг 8. Секретарь может просмотреть сформированную справку, затем отправить на печать (принтер) или сохранить в PDF.

— Шаг 9. Если требуется сохранить в системе, приложение отправляет серверу БД команду сохранить копию справки (для архива).

— Шаг 10. Сервер БД подтверждает сохранение, и процесс завершается.

Эта диаграмма показывает, что секретарь выступает оператором, который только запускает процессы, а основная работа по поиску данных, формированию документа и его сохранению выполняется приложением и сервером БД.

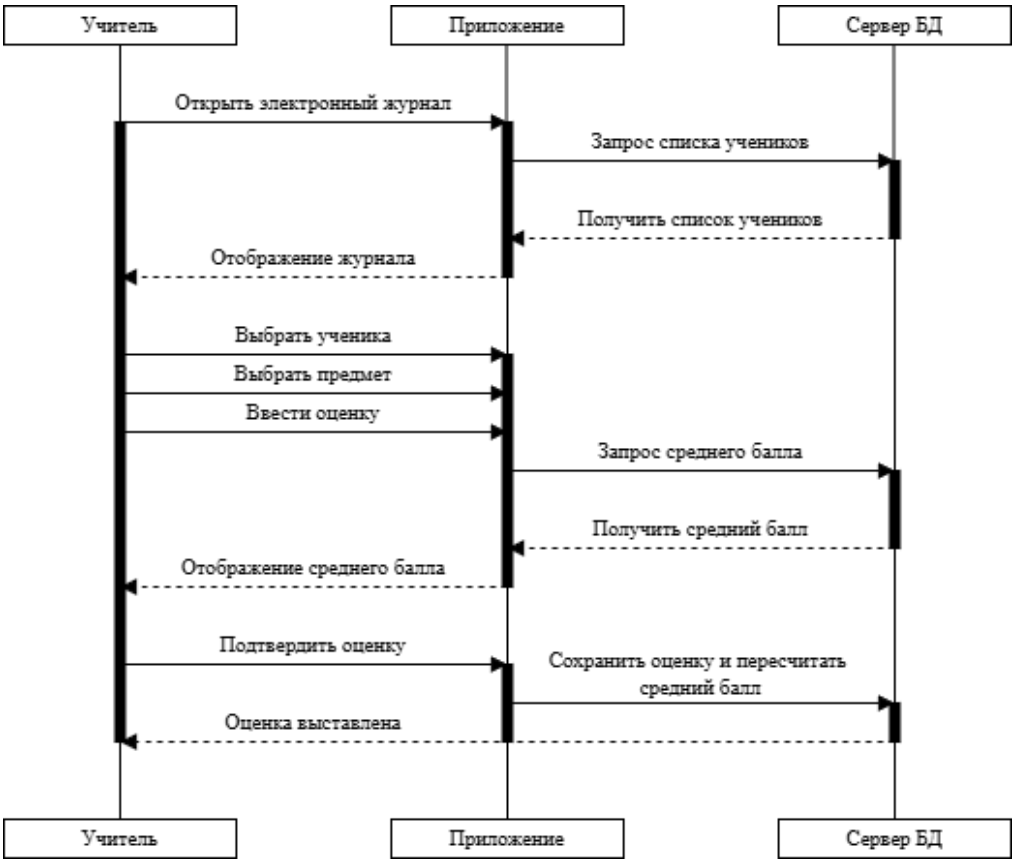


Рисунок 2.7 - Диаграмма последовательности «Учитель. Выставление оценок»

На рисунке 2.7 участвуют три объекта: Учитель, Приложение и Сервер БД.

— Шаг 1. Учитель входит в приложение и открывает модуль «Электронный журнал».

— Шаг 2. Приложение запрашивает у сервера БД список классов, закреплённых за этим учителем.

— Шаг 3. Сервер БД возвращает список классов.

— Шаг 4. Учитель выбирает класс, затем предмет (если ведёт несколько предметов). Приложение запрашивает у сервера БД список учеников этого класса.

— Шаг 5. Сервер БД возвращает список учеников.

— Шаг 6. Учитель выбирает ученика и вводит оценку (или отметку о пропуске). Приложение отправляет на сервер БД команду «Сохранить оценку» с параметрами: ученик, предмет, дата, значение.

— Шаг 7. Сервер БД сохраняет оценку в соответствующей таблице.

— Шаг 8. Сервер БД автоматически пересчитывает средний балл этого ученика по данному предмету (или по всем предметам).

— Шаг 9. Сервер БД возвращает обновлённое значение среднего балла в приложение.

— Шаг 10. Приложение отображает учителю средний балл.

— Шаг 11. При необходимости (если есть задолженности или низкий балл) приложение может отправить уведомление родителю, но это уже дополнительная ветка.

Диаграмма показывает, что учитель лишь вводит оценку, а все расчёты (средний балл) и хранение выполняются на стороне сервера БД.

Диаграмма кооперации (коммуникации) — это вариант диаграммы взаимодействия, акцентирующий внимание на структурных связях между объектами, участвующими во взаимодействии, а не на временной

последовательности. В отличие от диаграммы последовательности, где важны время и порядок сообщений, в диаграмме кооперации важен топологический порядок. Сообщения нумеруются, чтобы указать последовательность.

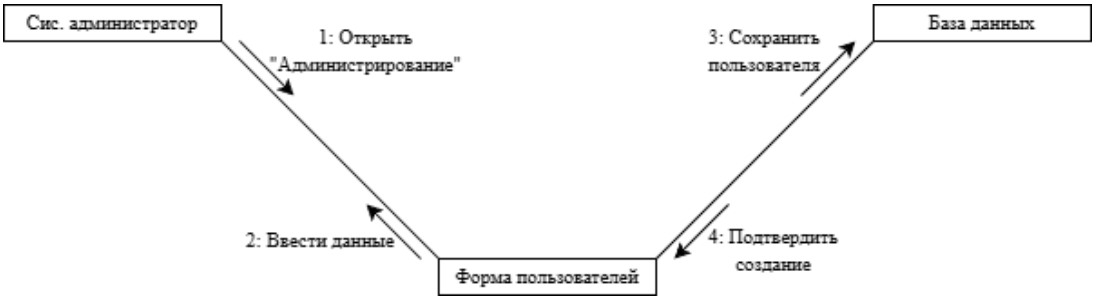


Рисунок 2.8 - Диаграмма кооперации «Администратор. Добавление пользователя»

На рисунке 2.8 изображены три участника: Системный администратор, Форма пользователей и База данных. Администратор работает с формой пользователей, где вводит данные нового пользователя (ФИО, логин, пароль, роль). Затем форма пользователей передаёт эти данные в базу данных для сохранения. Здесь принтер не участвует, так как результат операции — не печатный документ, а запись в базе данных. Эта кооперация отражает типичное взаимодействие при администрировании: пользовательский интерфейс передаёт введённые данные в хранилище.

Рисунок 2.9 - Диаграмма кооперации «Директор. Формирование отчётов»

На рисунке 2.9 изображены три участника взаимодействия: Директор, Форма отчётов и Принтер. Связи между ними организованы следующим образом. Директор инициирует процесс формирования отчёта, обращаясь к форме отчётов. Форма отчётов, в свою очередь, взаимодействует с принтером для вывода готового документа на печать. Таким образом, кооперация показывает, что директор через форму отчётов отправляет данные на печать, а принтер выступает устройством вывода. База данных или сервер на этой диаграмме не показаны, так как она отражает лишь непосредственное взаимодействие пользователя, интерфейса и устройства печати.

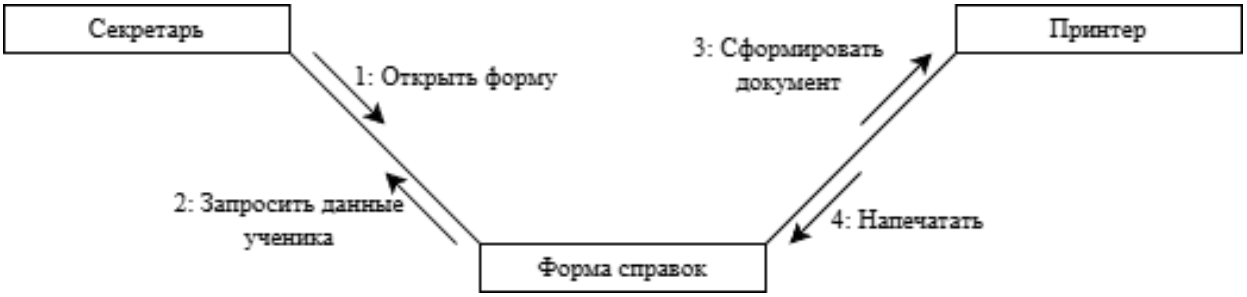


Рисунок 2.10 - Диаграмма кооперации «Секретарь. Выдача справок»

На рисунке 2.10 представлены три объекта: Секретарь, Форма справок и Принтер. Секретарь работает с формой справок, где, вероятно, выбирает ученика, тип справки и заполняет необходимые данные. После этого форма справок передаёт готовый документ на принтер. Как и в предыдущем случае, принтер выступает конечным устройством вывода. Диаграмма наглядно показывает, что процесс выдачи справки завершается печатью документа, а секретарь является инициатором действия.

Диаграмма деятельности — это диаграмма UML, которая моделирует поток управления в бизнес-процессе от одного действия к другому. Она аналогична классической блок-схеме алгоритма. Основные элементы: начальный узел (черный круг), конечный узел (черный круг с ободком),

действия (скругленные прямоугольники), узлы ветвления (ромбы — if/else), дорожки (Swimlanes), разделяющие обязанности разных акторов.

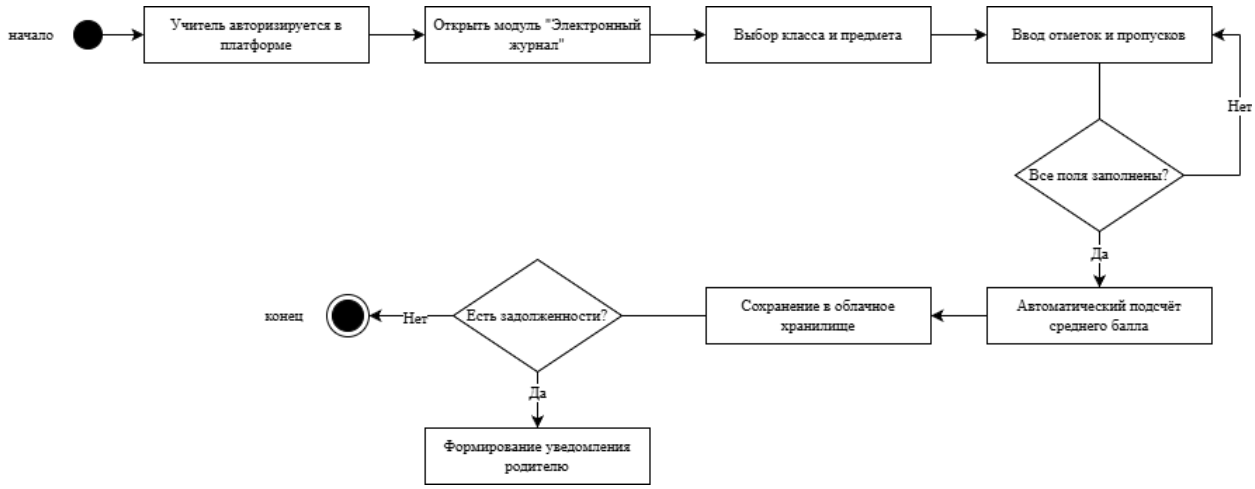


Рисунок 2.11 - Диаграмма деятельности «Заполнение электронного журнала»

Рисунок 2.11 описывает процесс работы учителя с электронным журналом. Учитель начинает с авторизации в платформе. После успешного входа он открывает модуль «Электронный журнал». Затем учитель выбирает класс и конкретный предмет, по которому будет выставлять отметки. Далее он приступает к вводу отметок и фиксации пропусков. После ввода данных система выполняет проверку: все ли поля заполнены. Если не все поля заполнены, учитель возвращается к этапу ввода отметок и пропусков. Если все поля заполнены, система переходит к проверке наличия задолженностей у учеников. При отсутствии задолженностей данные сохраняются в облачное хранилище. При наличии задолженностей система сначала формирует уведомление родителю, а затем также производит сохранение в облачное хранилище. Завершающим этапом идёт автоматический подсчёт среднего балла на основе введённых отметок.

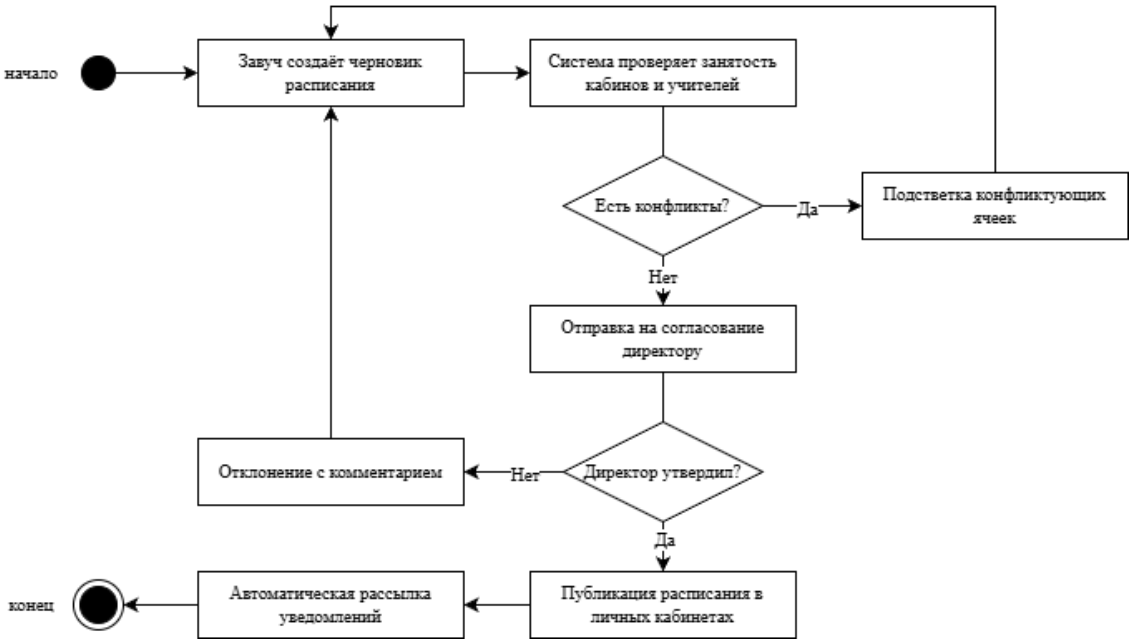


Рисунок 2.12 - Диаграмма деятельности «Согласование расписания»

Рисунок 2.12 описывает процесс создания, проверки и утверждения расписания. Сначала завуч создаёт черновик расписания. Затем система автоматически проверяет занятость кабинетов и учителей. Если конфликты обнаружены, система подсвечивает конфликтующие ячейки, и завуч возвращается к этапу редактирования черновика для устранения конфликтов. Если конфликтов нет, расписание передаётся на утверждение директору. Далее система проверяет, утвердил ли директор расписание. Если директор не утвердил, он отправляет завучу отклонение с комментарием, после чего завуч снова редактирует черновик. Если директор утвердил расписание, система выполняет автоматическую рассылку уведомлений всем заинтересованным лицам и публикует расписание в личных кабинетах учителей. Процесс на этом завершается.

Обе диаграммы наглядно показывают, как в школьной информационной системе организованы ключевые процессы: выставление оценок с контролем полноты данных и уведомлением родителей, а также

согласование расписания с проверкой конфликтов и утверждением руководителем.

Диаграмма состояний (State Machine Diagram) — это диаграмма UML, которая описывает все возможные состояния одного объекта (например, документа) и переходы между ними под воздействием внешних событий. Она показывает жизненный цикл объекта.

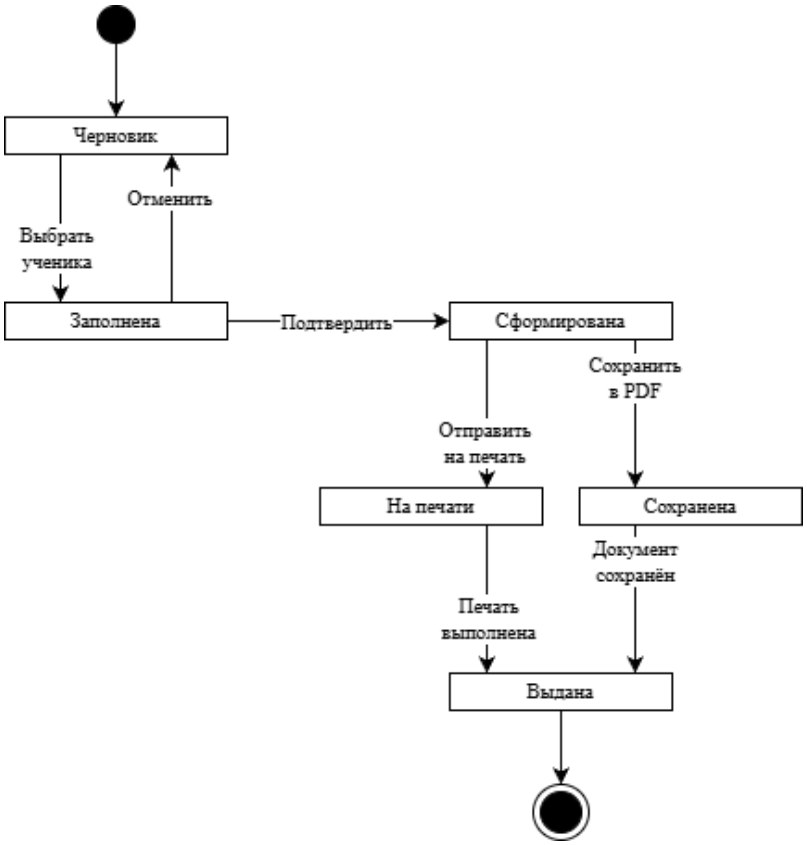


Рисунок 2.13 - Диаграмма состояния «Справка»

Рисунок 2.13 описывает жизненный цикл процесса выдачи справки секретарём. Начальным состоянием является «Черновик» — секретарь начинает заполнение формы. На этом этапе он может выбрать ученика и заполнить все необходимые поля, после чего документ переходит в состояние «Заполнена». Если на любом этапе необходимо прервать процесс, доступно действие «Отменить», которое возвращает состояние к исходному

«Черновик». Из состояния «Заполнена» секретарь переходит к действию «Подтвердить», после чего документ переходит в состояние «Сформирована». Далее возможны два параллельных пути. Первый путь: «Сохранить в PDF» → «Сохранена» → «Документ сохранён». Второй путь: «Отправить на печать» → «На печати» → «Печать выполнена» → «Выдана». Конечным состоянием процесса является «Выдана» — справка успешно выдана на руки.

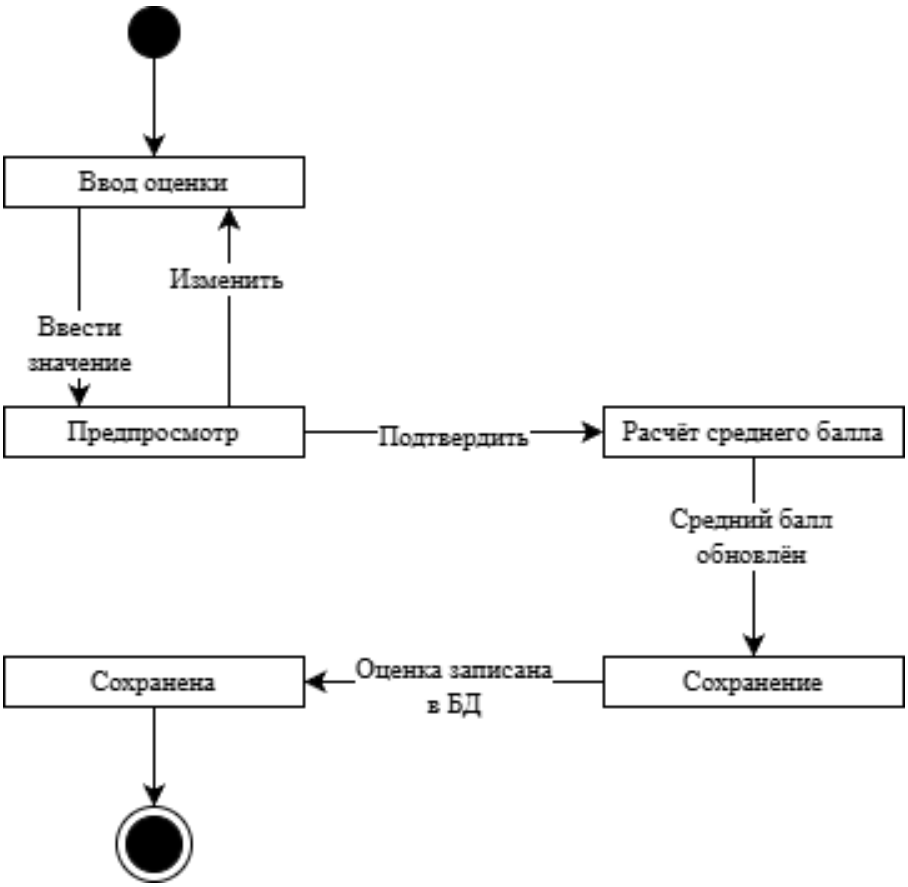


Рисунок 2.14 - Диаграмма состояния «Оценка»

Рисунок 2.14 описывает жизненный цикл процесса выставления оценки учителем. Начальным состоянием является «Ввод оценки». Учитель вводит значение, одновременно может находиться в состоянии «Предпросмотр» или при необходимости перейти в состояние «Изменить» для корректировки.

После ввода значения учитель переходит к действию «Подтвердить». Далее система переходит в состояние «Расчёт среднего балла», одновременно выполняются два параллельных процесса: с одной стороны, оценка записывается в базу данных (состояния «Оценка записана в БД» → «Сохранена»), с другой стороны — обновляется средний балл ученика (состояние «Средний балл обновлён»). Завершающим состоянием является «Сохранение», которое фиксирует успешное завершение всего процесса.

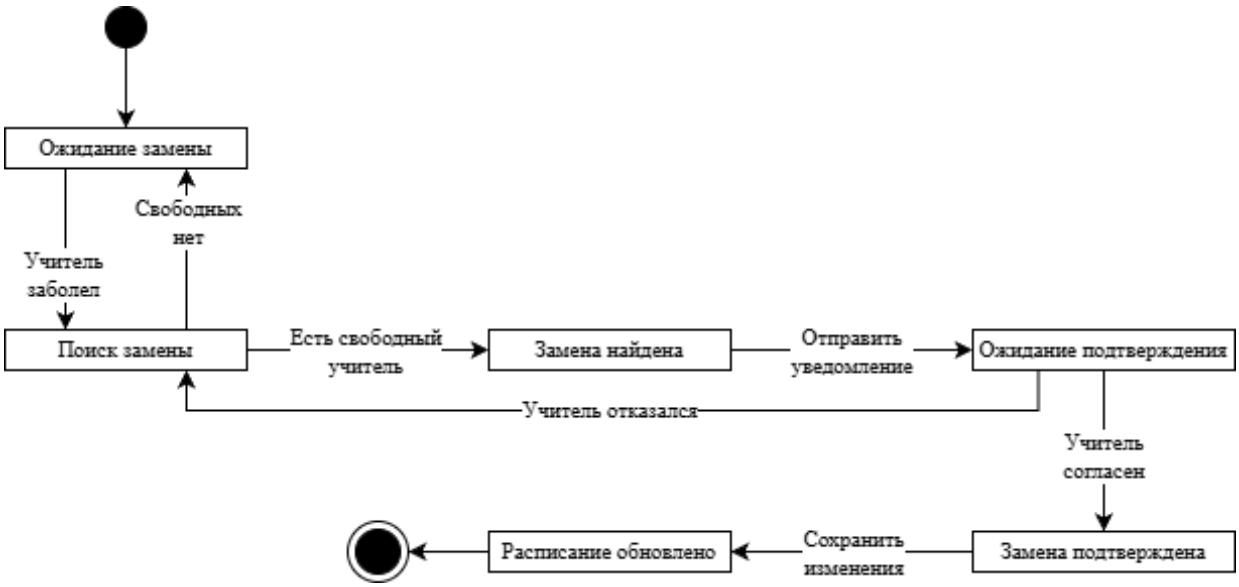


Рисунок 2.15 - Диаграмма состояния «Замена учителя»

Рисунок 2.15 описывает жизненный цикл процесса замены отсутствующего учителя. Начальным состоянием является «Ожидание замены». Когда учитель заболел, процесс переходит к состоянию «Поиск замены». Далее система проверяет наличие свободного учителя. Если свободного учителя нет, процесс возвращается в состояние «Поиск замены» (цикл ожидания). Если свободный учитель есть, процесс переходит в состояние «Замена найдена», после чего система отправляет уведомление выбранному учителю и переходит в состояние «Ожидание подтверждения». В этом состоянии возможны два варианта: если учитель отказался, процесс

возвращается к состоянию «Поиск замены»; если учитель согласен, процесс переходит к состоянию «Сохранить изменения». Далее система переходит в состояние «Замена подтверждена», затем в «Расписание обновлено». Финальным состоянием является «Расписание обновлено» — процесс завершён успешно.

Все три диаграммы состояний наглядно показывают, как объекты системы (замена учителя, оценка, справка) меняют свои состояния под воздействием внешних событий и действий пользователей, включая альтернативные ветки (отказ учителя, отмена черновика) и параллельные процессы (запись в БД и расчёт среднего балла, PDF и печать справки).

Диаграмма компонентов (Component Diagram) — это статическая диаграмма UML, которая показывает разбиение программной системы на компоненты, а также зависимости между ними. Компонент — это модуль, имеющий четко определенные границы и интерфейсы (набор предоставляемых и требуемых услуг).

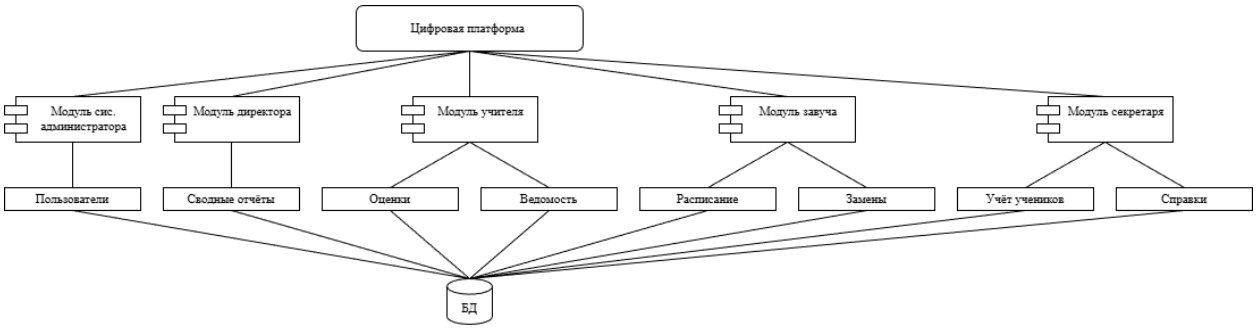


Рисунок 2.16 - Диаграмма компонентов

Рисунок 2.16 отображает физическую структуру школьной цифровой платформы, показывая, как программные компоненты распределены и связаны между собой. В центре диаграммы расположена основная сущность — «Цифровая платформа», которая представляет собой всю систему в целом. Внутри платформы выделены пять отдельных модулей, каждый из которых

соответствует определённой роли пользователя: «Модуль системного администратора», «Модуль директора», «Модуль учителя», «Модуль завуча» и «Модуль секретаря».

Эти модули, в свою очередь, взаимодействуют с набором функциональных компонентов, отвечающих за конкретные бизнес-задачи. Модуль системного администратора связан с компонентом «Пользователи» (управление учётными записями). Модуль директора взаимодействует с компонентом «Сводные отчёты». Модуль учителя работает с компонентами «Оценки» и «Ведомость». Модуль завуча использует компоненты «Расписание» и «Замены». Модуль секретаря связан с компонентами «Учёт учеников» и «Справки».

Все перечисленные компоненты в конечном итоге обращаются к единому хранилищу данных — компоненту «БД» (база данных), который показан в нижней части диаграммы. Таким образом, диаграмма демонстрирует модульную архитектуру: каждый модуль отвечает за свою группу функций, но все они используют общую базу данных для хранения и извлечения информации. Это обеспечивает разделение ответственности между ролями пользователей и централизованное управление данными.

Диаграмма размещения (Deployment) — это диаграмма UML, которая моделирует физическую конфигурацию узлов (серверов, рабочих станций, принтеров) и размещение на них артефактов (компонентов) системы. Она показывает, какое программное обеспечение на каком физическом устройстве работает и как эти устройства соединены в сеть.

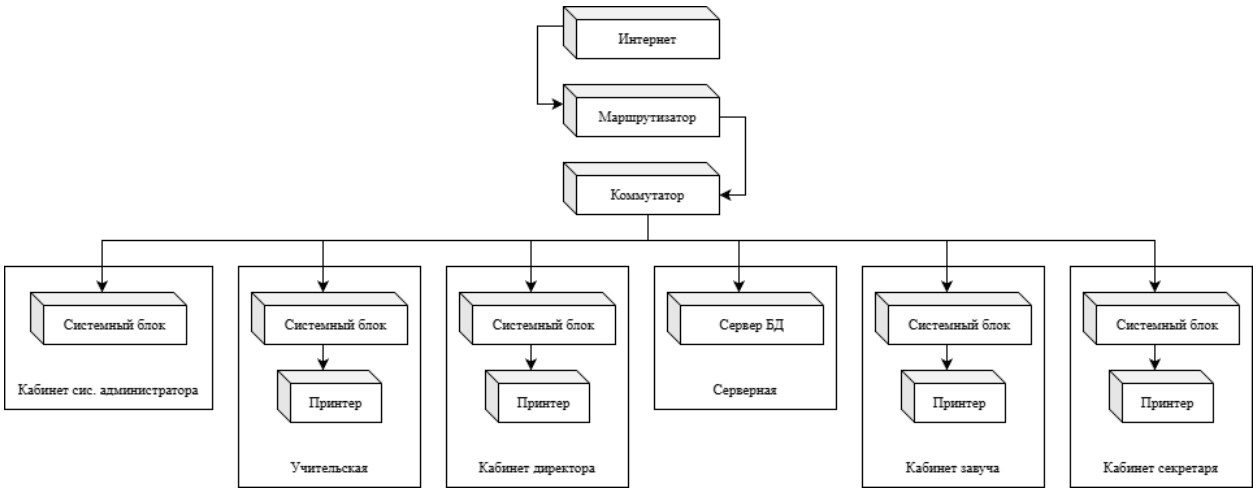


Рисунок 2.17 - Диаграмма размещения

Рисунок 2.17 отображает физическую архитектуру школьной информационной системы, показывая, как программные и аппаратные компоненты распределены по помещениям и связаны через сеть. В верхней части диаграммы изображён «Интернет», который подключён к «Маршрутизатору» — устройству, обеспечивающему связь между локальной сетью и внешними сетями. От маршрутизатора сигнал идёт на «Коммутатор» (сетевой свитч), который распределяет подключение между всеми системными блоками в организации.

Далее на диаграмме показаны несколько рабочих мест и помещений. В «Кабинете системного администратора» расположен один «Системный блок» — компьютер, с которого администратор управляет системой, настраивает пользователей, сеть и выполняет резервное копирование.

В «Серверной» размещён «Сервер БД» (сервер базы данных), к которому также подключён «Принтер». Это центральный узел системы, хранящий всю информацию: данные об учениках, оценках, расписании, справках, отчётах и пользователях. Принтер в серверной, вероятно, используется для печати системных отчётов или журналов.

Далее на диаграмме показаны три системных блока, каждый из которых подключён к своему принтеру. Один системный блок находится в

«Кабинете завуча», другой — в «Кабинете секретаря», а третий, по всей видимости, в кабинете директора или учителя (комната не подписана явно, но следует общей логике). В каждом из этих кабинетов есть принтер, что позволяет сотрудникам (завучу, секретарю, учителям или директору) распечатывать необходимые документы: справки, ведомости, расписания, отчёты.

Таким образом, диаграмма размещения показывает классическую топологию «звезда» через коммутатор, с выделенным сервером БД в серверной и несколькими клиентскими рабочими станциями в разных кабинетах. Все системные блоки подключены к коммутатору, а через маршрутизатор имеют доступ в интернет (хотя из предыдущих бизнес-процессов известно, что интернет не обязателен для работы системы, но он может использоваться для отправки уведомлений родителям или облачного сохранения). Принтеры распределены по кабинетам для локальной печати документов.

3 РАЗРАБОТКА И ТЕСТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА

В данной главе на основе логической модели данных, архитектуры «клиент-сервер» и UML-диаграмм описывается реализация платформы: создание базы данных в PostgreSQL, разработка клиентской и серверной частей на C# и AvaloniaUI, внедрение аутентификации и разграничения доступа, а также результаты функционального, модульного и нагрузочного тестирования.

3.1 Создание структуры базы данных и разработка серверной части приложения

Физическая реализация базы данных выполнялась в СУБД PostgreSQL 18 [23]. На основе логической модели (раздел 2.2) были сгенерированы SQL-скрипты создания таблиц с использованием Entity Framework Core (подход Code First) [26]. Такой подход обеспечивает автоматическое создание и миграцию схемы данных при изменении моделей C#.

```
1   public class Uchenik
2   {
3       public int Id { get; set; }
4       public string Familiya { get; set; }
5       public string Imya { get; set; }
6       public string Otchestvo { get; set; }
7       public DateTime DataRozhdeniya { get; set; }
8       public string AdresProzhivaniya { get; set; }
9       public string FIORoditelya { get; set; }
10      public string TelefonRoditelya { get; set; }
11      public DateTime DataZachisleniya { get; set; }
12      public int KlassId { get; set; }
13      public Klass Klass { get; set; }
14  }
```

Рисунок 3.1 – Пример модели «Ученик» на C# (фрагмент)

На рисунке 3.1 представлен фрагмент модели «Ученик» на языке программирования C#.

Серверная часть фактически совмещена с клиентской ввиду двухзвенной архитектуры. Роль сервера выполняет СУБД PostgreSQL, а также хранимые процедуры и триггеры для автоматического пересчёта среднего балла и проверки ссылочной целостности. Для оптимизации производительности созданы индексы по полям Familiya, KlassId и DataUroka.

3.2 Разработка пользовательского интерфейса и административной панели системы

Пользовательский интерфейс реализован на фреймворке Avalonia UI 11.0 [24], что обеспечивает кроссплатформенную работу приложения (Windows, Linux). Интерфейс построен по модульному принципу: главное окно содержит панель навигации (меню), рабочую область для отображения списков (ученики, сотрудники, расписание) и форм редактирования.

Ниже представлены макеты ключевых экранов, разработанные в рамках проектирования и полностью соответствующие реализованному функционалу.

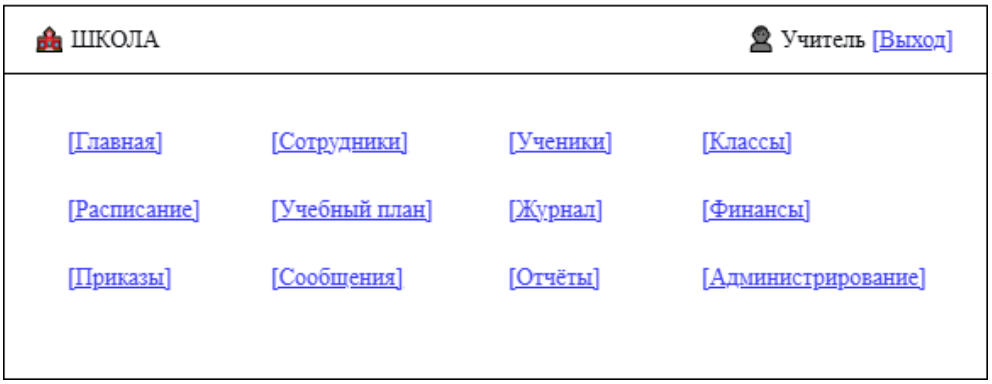


Рисунок 3.2 – Начальная страница

Данный макет (рисунок 3.2) представляет собой верхнюю панель навигации, которая присутствует на всех экранах системы. Панель содержит строку с названием «ШКОЛА» и набор ссылок для перехода между основными разделами. В состав навигации входят следующие пункты: «Главная», «Сотрудники», «Ученики», «Классы», «Расписание», «Учебный план», «Журнал», «Финансы», «Приказы», «Сообщения», «Отчёты», «Администрирование». Данная панель обеспечивает быстрый доступ к любому функциональному модулю системы вне зависимости от того, на каком экране в данный момент находится пользователь.

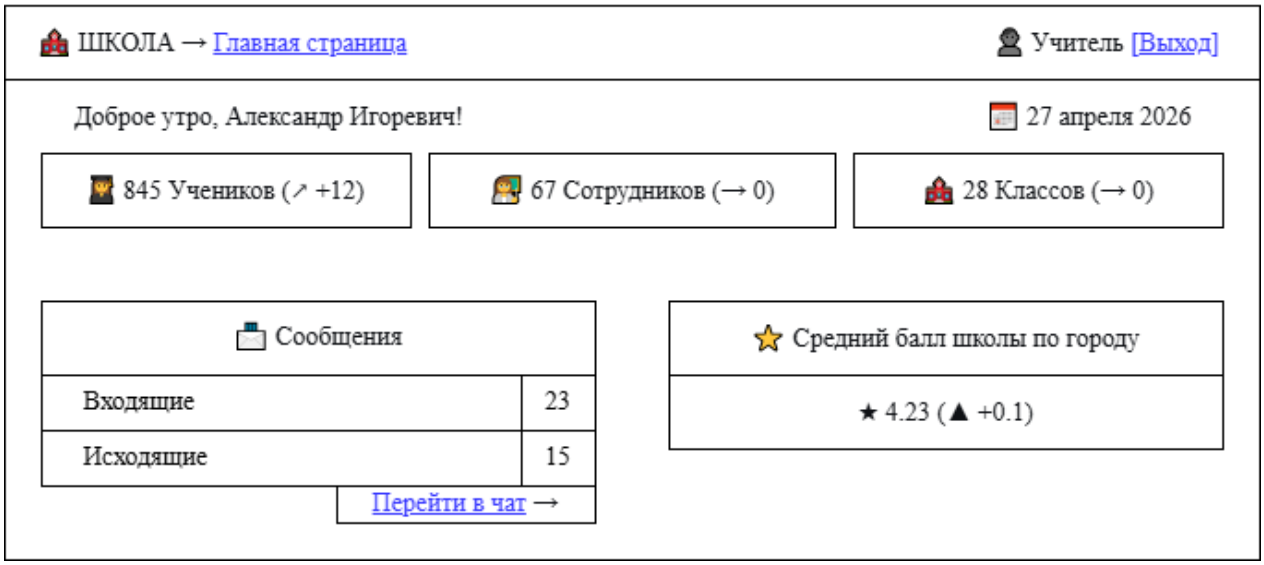


Рисунок 3.3 – Главная страница

Макет главной страницы (рисунок 3.3) является центральной (домашней) страницей системы, которая отображается после успешного входа пользователя. В верхней части экрана расположено персонализированное приветствие с именем пользователя — «Доброе утро, Александр Игоревич!». Справа от приветствия находятся элементы: указание текущей даты («27 апреля 2026») и кнопка выхода из системы («Выход»).

Карточки ключевых показателей расположены в верхней центральной части и включают три блока:

— «845 Учеников» с индикатором динамики «↗ +12»;

— «67 Сотрудников» с индикатором «↗ 0» (без изменений);

— «28 Классов» с индикатором «↗ 0».

Отдельным блоком выведен «Средний балл школы по городу», что позволяет директору и завучу в реальном времени оценивать общую успеваемость. Блок «Сообщения» расположен в правой части страницы и содержит количество входящих сообщений (23), количество исходящих сообщений (15) и кнопку-ссылку «Перейти в чат →» для перехода в модуль внутренней переписки. Таким образом, главная страница даёт пользователю мгновенный обзор ключевых показателей деятельности школы и быстрый доступ к непрочитанным сообщениям.

ШКОЛА → Сотрудники Директор [Выход]

Должность▼ Кабинет▼ + Добавить

ФИО	Должность	Кабинет	Классное руководство	Телефон	Действия
Иванов И.И.	Учитель	212	5А	+79000000000	Редактировать
Петров П.П.	Учитель	205	6Б	+79001112323	Редактировать
...	...	...	...	...	...

Рисунок 3.4 – Страница «Сотрудники»

Данный экран (рисунок 3.4) предназначен для управления кадрами. В верхней части расположены фильтры: выпадающий список «Должность» и выпадающий список «Кабинет», а также кнопка «+ Добавить» для добавления нового сотрудника. Таблица сотрудников содержит следующие колонки: «ФИО», «Должность», «Кабинет», «Классное руководство», «Телефон», «Действия» (кнопка «Редактировать» для каждой строки). Примеры записей: Иванов И.И. — учитель, кабинет 212, классное руководство 5А; Петров П.П. — учитель, кабинет 205, классное руководство 6Б. Страница доступна секретарю и администратору для ведения кадрового учёта.

ШКОЛА → Ученики Классный руководитель (5А) [Выход]

Классный руководитель видит ТОЛЬКО свой класс (5А)

+ Добавить ученика

ФИО	Класс	Дата рождения	Родители	Средний балл	Действия
Иванов И.И.	5А	12.05.2013	Иванова Е.Н.	4.39	Редактировать
Петров П.П.	5А	01.01.2013	Петрова М.А.	3.28	Редактировать
Тимохина А.И.	5А	15.06.2013	Тимохин В.С.	5.00	Редактировать
...	...	...	...	...	...

Рисунок 3.5 – Страница «Ученики»

Макет (рисунок 3.5) отображает список учащихся с учётом ролевого ограничения — в верхней части указано: «Классный руководитель видит

ТОЛЬКО свой класс (5А)». Кнопка «+ Добавить ученика» позволяет секретарю или классному руководителю зачислить нового учащегося. Таблица учеников содержит следующие колонки: «ФИО», «Класс», «Дата рождения», «Родители», «Средний балл», «Действия» (кнопка «Редактировать»). Примеры записей: Иванов И.И. (5А, 12.05.2013, родители Иванова Е.Н., средний балл 4.39); Петров П.П. (5А, 01.01.2013, родители Петрова М.А., средний балл 3.28); Тимохина А.И. (5А, 15.06.2013, родители Тимохин В.С., средний балл 5.00). При нажатии на кнопку «Редактировать» открывается карточка ученика (см. рисунок 3.15).

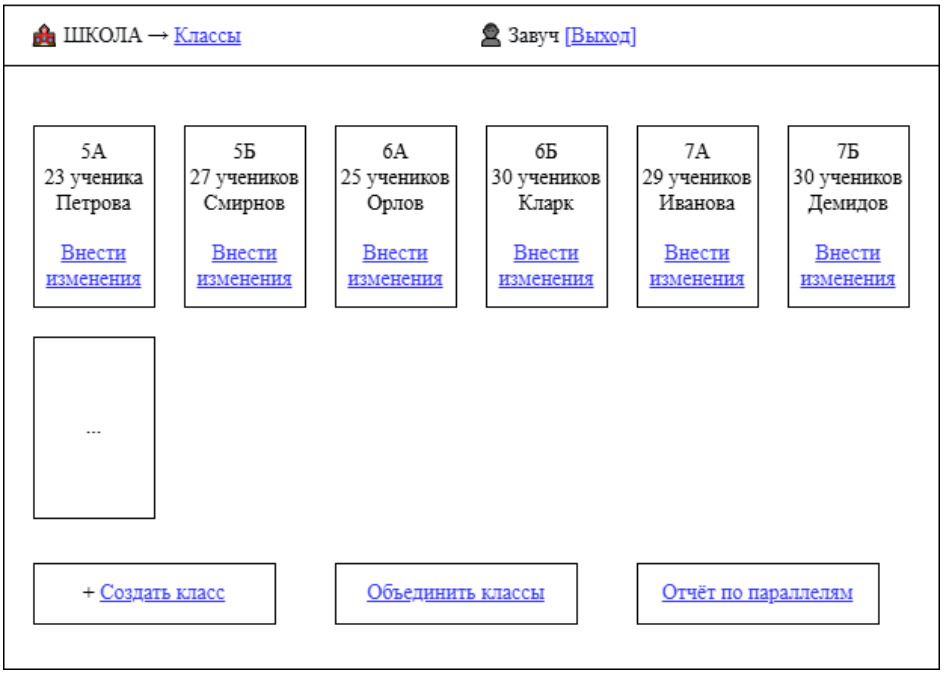


Рисунок 3.6 – Страница «Классы»

Экран (рисунок 3.6) предназначен для управления классами школы. В верхней части указана роль пользователя («Завуч») и кнопка выхода. Основная информация организована в виде карточек классов, каждая карточка содержит:

— Название класса («5А», «5Б», «6А», «6Б», «7А», «7Б»).

— Количество учеников («23 ученика», «27 учеников», «25 учеников», «30 учеников», «29 учеников», «30 учеников»).

— Фамилию классного руководителя («Петрова», «Смирнов», «Орлов», «Кларк», «Иванова», «Демидов»).

— Кнопку «Внести изменения» для редактирования параметров класса.

В нижней части экрана расположены кнопки глобальных действий: «+ Создать класс», «Объединить классы», «Отчёт по параллелям».

ШКОЛА → Расписание Завуч [Выход]

Класс: 5А ▼ Неделя: 14.04 – 18.04.2026 ▼ Учитель: Все ▼

	ПН 14.04	ВТ 15.04	СР 16.04	ЧТ 17.04	ПТ 18.04
8:30	Математика Петрова каб. 203	Русский язык Смирнова каб. 101	Математика Петрова каб. 203	Физкультура Орлов спортзал	Русский язык Смирнова каб. 101
9:20	Русский язык Смирнова каб. 101	История Соколов каб. 211	Английский язык Кларк каб. 104	Математика Петрова каб. 203	Окружающий мир Морозова каб. 107
10:10	Английский язык Кларк каб. 104	Математика Петрова каб. 203	Русский язык Смирнова каб. 101	Английский язык Кларк каб. 104	Труд Кузнецов каб. 110

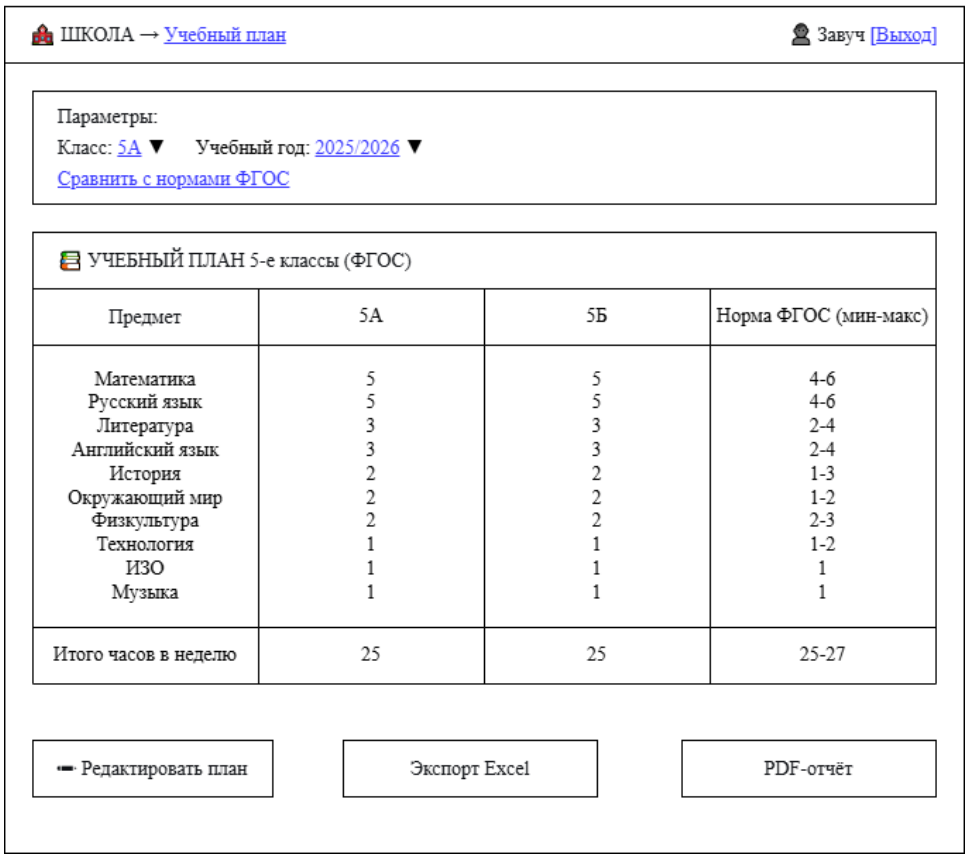
Заметки:
• 17.04 (ЧТ) – Физкультура на улице (погода: +15°)
• 18.04 (ПТ) – Контрольная работа по математике

Замена уроков Редактировать расписание Печать PDF

Рисунок 3.7 – Страница «Расписание»

Данный макет (рисунок 3.7) предназначен для просмотра и редактирования учебного расписания. В верхней части расположены основные элементы фильтрации и управления: выпадающий список «Класс: 5А», выпадающий список «Неделя: 14.04 – 18.04.2026», выпадающий список «Учитель: Все», а также кнопка «Замена уроков», открывающая диалог для

замены отсутствующего учителя. Основная таблица расписания имеет следующую структуру: строки соответствуют времени начала уроков («8:30», «9:20», «10:10»), столбцы соответствуют дням недели с конкретными датами («ПН 14.04», «ВТ 15.04», «СР 16.04», «ЧТ 17.04», «ПТ 18.04»). Каждая ячейка таблицы содержит три элемента: название предмета (Математика, Русский язык, Английский язык, Физкультура, Окружающий мир, Труд), фамилию учителя (Петрова, Смирнова, Соколов, Кларк, Орлов, Морозова, Кузнецов) и номер кабинета (каб. 203, каб. 101, каб. 211, каб. 104, каб. 107, каб. 110, спортзал). Блок заметок расположен под таблицей и содержит текстовые комментарии: «17.04 (ЧТ) – Физкультура на улице (погода: +15°)» и «18.04 (ПТ) – Контрольная работа по математике». Кнопки дополнительных действий: «Редактировать расписание» и «Печать PDF».


ШКОЛА → Учебный план

Завуч [Выход]

Параметры:
Класс: 5А ▼ Учебный год: 2025/2026 ▼
Сравнить с нормами ФГОС

УЧЕБНЫЙ ПЛАН 5-е классы (ФГОС)

Предмет	5А	5Б	Норма ФГОС (мин-макс)
Математика	5	5	4-6
Русский язык	5	5	4-6
Литература	3	3	2-4
Английский язык	3	3	2-4
История	2	2	1-3
Окружающий мир	2	2	1-2
Физкультура	2	2	2-3
Технология	1	1	1-2
ИЗО	1	1	1
Музыка	1	1	1
Итого часов в неделю	25	25	25-27

Редактировать план | Экспорт Excel | PDF-отчёт

Рисунок 3.8 – Страница «Учебный план»

Макет (рисунок 3.8) отображает распределение учебной нагрузки по предметам в соответствии с требованиями ФГОС. В верхней части расположены параметры выбора: выпадающий список «Класс: 5А», выпадающий список «Учебный год: 2025/2026» и кнопка «Сравнить с нормами ФГОС». Основная таблица учебного плана содержит следующие колонки: «Предмет», «5А» (количество часов), «5Б» (количество часов), «Норма ФГОС (мин-макс)». В таблице перечислены предметы: Математика, Русский язык, Литература, Английский язык, История, Окружающий мир, Физкультура, Технология, ИЗО, Музыка. Для 5А и 5Б классов часы распределены одинаково: 5, 5, 3, 3, 2, 2, 2, 1, 1, 1 соответственно. Итоговая строка: «Итого часов в неделю» со значениями «25», «25», «25-27». Кнопки действий: «Редактировать план», «Экспорт Excel», «PDF-отчёт».

ШКОЛА → Журнал Учитель [Выход]

⚠ Учитель видит ТОЛЬКО свои предметы
⚠ Родитель видит ТОЛЬКО своего ребёнка (только чтение)

5А класс▼ Математика▼ Апрель 2026▼

Ученик	01.04	03.04	08.04	10.04	Средний балл
Иванов Петр	5	4	5	4	4.50
Сидорова Анна	4	5	4	5	4.50
Петров Илья	3	3	4	3	3.25

Сохранить Отчет по классу Отчет по ученику

Рисунок 3.9 – Страница «Журнал»

Макет (рисунок 3.9) реализует электронный журнал для работы с оценками. В верхней части экрана присутствуют два важных предупреждения, ограничивающие доступ к данным: «Учитель видит ТОЛЬКО свои предметы» и «Родитель видит ТОЛЬКО своего ребёнка

(только чтение)». Основные элементы управления включают три выпадающих списка: выбор класса (на макете выбран «5А класс»), выбор предмета (на макете выбран «Математика»), выбор месяца (на макете выбран «Апрель 2026»). Таблица успеваемости построена по следующему принципу:

— строки таблицы — это фамилии и имена учеников («Иванов Петр», «Сидорова Анна», «Петров Илья»);

— столбцы — даты уроков в выбранном месяце («01.04», «03.04», «08.04», «10.04»);

— в ячейках на пересечении ученика и даты проставляются числовые оценки (5, 4, 3);

— последний столбец таблицы называется «Средний балл» и содержит автоматически рассчитанное значение для каждого ученика (4.50, 4.50, 3.25).

Кнопки управления расположены под таблицей: «Сохранить» (для фиксации внесённых изменений), «Отчет по классу» (для формирования сводной ведомости по всему выбранному классу), «Отчет по ученику» (для формирования индивидуальной ведомости на конкретного ученика).

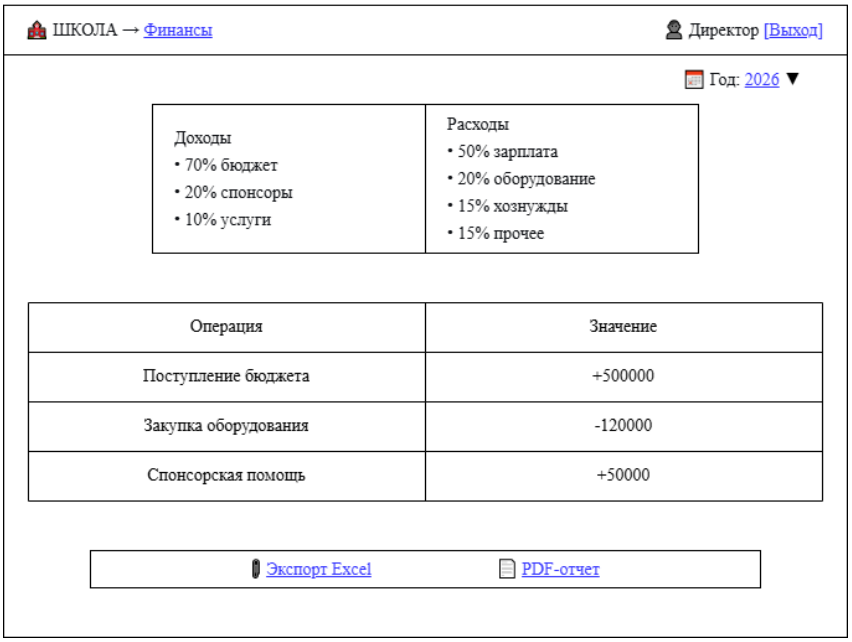


Рисунок 3.10 – Страница «Финансы»

Макет (рисунок 3.10) отображает финансовое состояние школы. В верхней части расположен выпадающий список для выбора года — «Год: 2026». Основная информация представлена в виде двух круговых диаграмм. Диаграмма «Доходы» содержит следующие доли: 70% — бюджет, 20% — спонсоры, 10% — услуги. Диаграмма «Расходы» содержит следующие доли: 50% — зарплата, 20% — оборудование, 15% — хознужды, 15% — прочее. Таблица последних операций содержит следующие строки: «Поступление бюджета» — «+500000», «Закупка оборудования» — «-120000», «Спонсорская помощь» — «+50000». Кнопки экспорта: «Экспорт Excel» (для выгрузки таблицы последних операций) и «PDF-отчёт» (для формирования полного финансового отчёта).

ШКОЛА → Приказы Директор [Выход]

+ Создать приказ | Подписать ЭЦП | Архив | Экспорт PDF

Фильтры: Тип▼ Дата▼ Статус▼

№	Дата	Название	Тип	Статус
12	01.09.2022	О зачислении	Зачисление	Действует
24	10.01.2023	Перевод 5А→5Б	Перевод	Исполнен

Рисунок 3.11 – Страница «Приказы»

Макет (рисунок 3.11) предназначен для работы с распорядительными документами. В верхней части расположены кнопки: «+ Создать приказ», «Подписать ЭЦП», «Архив», «Экспорт PDF». Ниже расположены фильтры:

«Тип», «Дата», «Статус» (каждый в виде выпадающего списка). Таблица приказов содержит следующие колонки: «№», «Дата», «Название», «Тип», «Статус». Примеры записей: приказ №12 от 01.09.2022 «О зачислении» (тип — Зачисление, статус — Действует); приказ №24 от 10.01.2023 «Перевод 5А→5В» (тип — Перевод, статус — Исполнен).

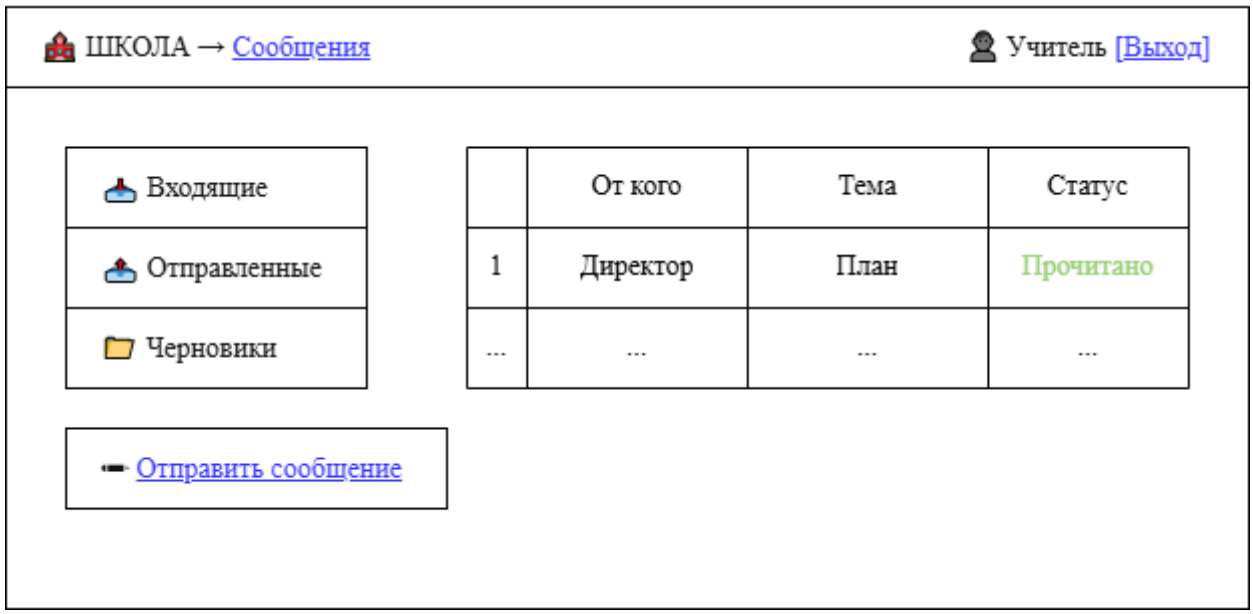


Рисунок 3.12 – Страница «Сообщения»

Макет (рисунок 3.12) представляет собой интерфейс внутреннего чата системы. В левой части расположены папки: «Входящие», «Отправленные», «Черновики». В центральной части — таблица сообщений с колонками: «От кого», «Тема», «Статус» (например, «Прочитано»). Пример строки в таблице: «1», «Директор», «План», «Прочитано». В правой части или внизу расположена кнопка «Отправить сообщение» для создания нового сообщения.

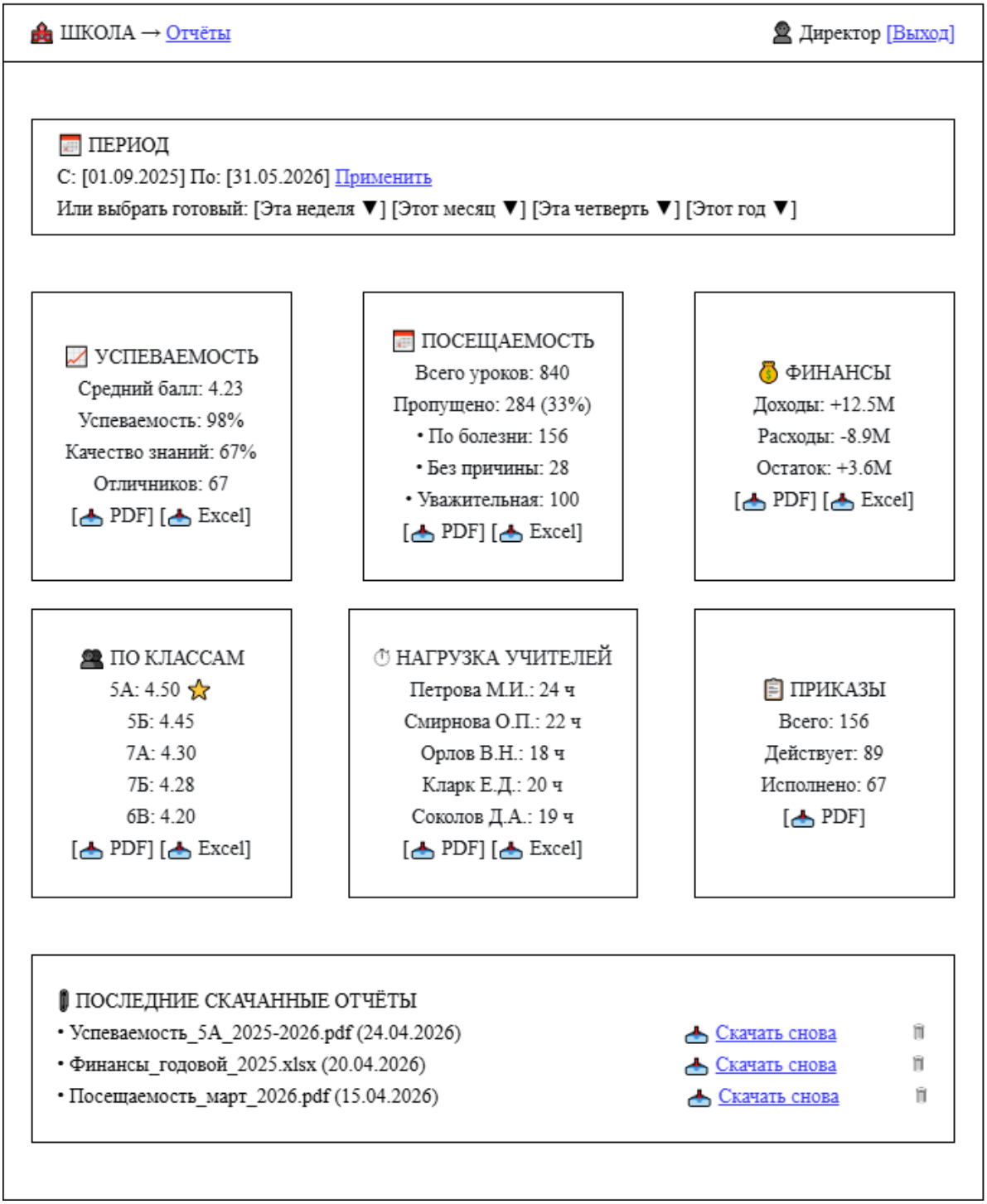


Рисунок 3.13 – Страница «Отчёты»

Макет (рисунок 3.13) представляет собой комплексный экран для формирования различных видов отчётности. Верхняя часть содержит блок выбора периода: поля «С: [01.09.2025] По: [31.05.2026]» с кнопкой

«Применить», а также готовые интервалы: «Эта неделя», «Этот месяц», «Эта четверть», «Этот год» (в виде выпадающего списка). Далее следуют блоки отчётов:

— Успеваемость: средний балл 4.23, успеваемость 98%, качество знаний 67%, отличников 67, кнопки «PDF» и «Excel»;

— Посещаемость: всего уроков 840, пропущено 284 (33%), с детализацией: по болезни 156, без причины 28, уважительная 100, кнопки «PDF» и «Excel»;

— Финансы: доходы +12.5M, расходы -8.9M, остаток +3.6M, кнопки «PDF» и «Excel»;

— По классам: список классов со средними баллами (5А: 4.50, 5Б: 4.45, 7А: 4.30, 7Б: 4.28, 6В: 4.20), кнопки «PDF» и «Excel»;

— Нагрузка учителей: список учителей с количеством часов (Петрова М.И.: 24 ч, Смирнова О.П.: 22 ч, Орлов В.Н.: 18 ч, Кларк Е.Д.: 20 ч, Соколов Д.А.: 19 ч), кнопки «PDF» и «Excel»;

— Приказы: всего 156, действует 89, исполнено 67, кнопка «PDF». Блок «Последние скачанные отчёты» содержит список ранее скачанных файлов с кнопками «Скачать снова» для каждого: «Успеваемость_5А_2025-2026.pdf (24.04.2026)», «Финансы_годовой_2025.xlsx (20.04.2026)», «Посещаемость_март_2026.pdf (15.04.2026)».

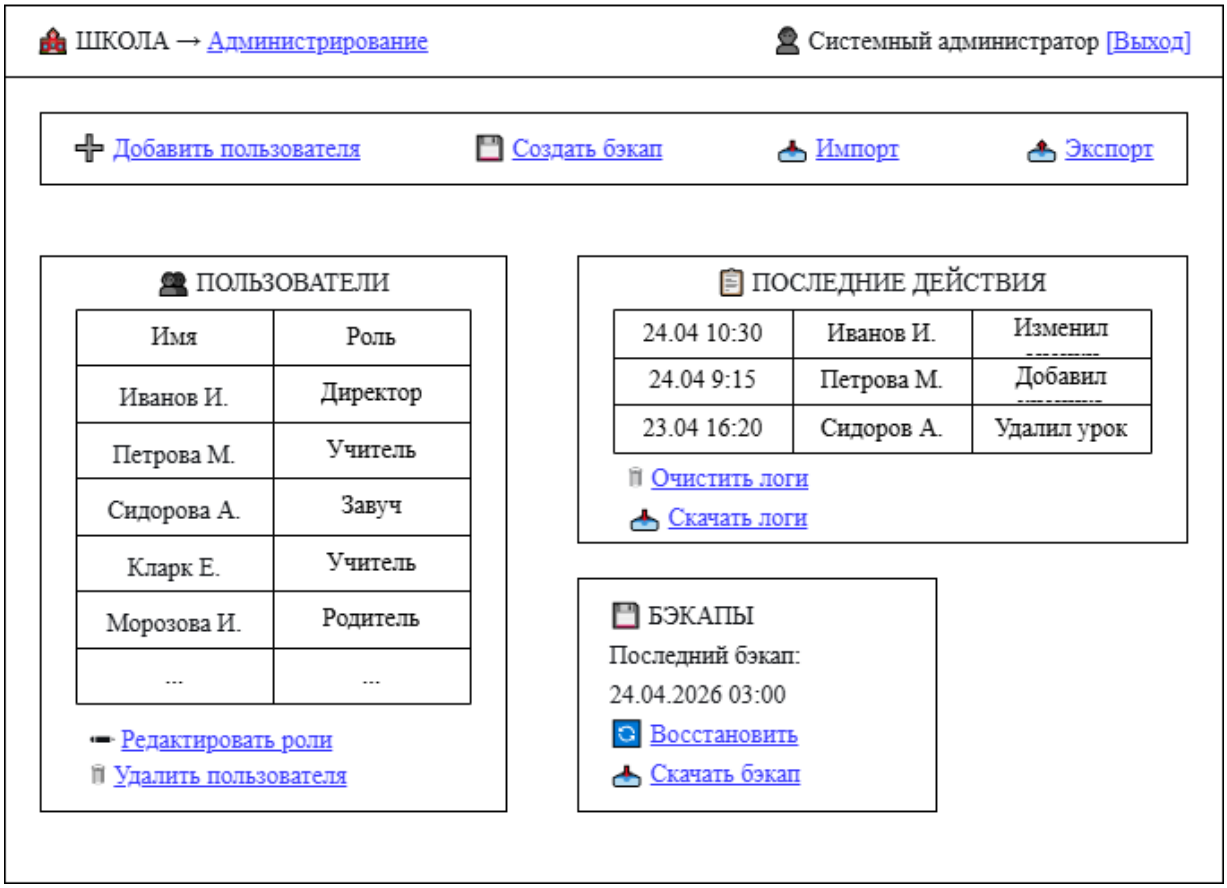


Рисунок 3.14 – Страница «Администрирование»

Макет (рисунок 3.14) представляет собой панель управления системой, доступную только пользователю с ролью «Системный администратор». Экран разделён на несколько функциональных блоков. Блок «Добавить пользователя / Создать бэкап / Импорт / Экспорт» содержит кнопки: «Добавить пользователя» (для создания новой учётной записи), «Создать бэкап» (для создания резервной копии базы данных), «Импорт» (для загрузки данных из внешних файлов), «Экспорт» (для выгрузки данных). Блок «Пользователи» представляет собой таблицу со следующими колонками: «Имя» (Иванов И., Петрова М., Сидорова А., Кларк Е., Морозова И. и т.д.) и «Роль» (Директор, Учитель, Завуч, Учитель, Родитель и т.д.). Для каждой записи доступны действия «Редактировать роли» и «Удалить пользователя». Блок «Последние действия» содержит журнал событий с колонками: дата и

время («24.04 10:30», «24.04 9:15», «23.04 16:20»), имя пользователя («Иванов И.», «Петрова М.», «Сидоров А.»), действие («Изменил», «Добавил», «Удалил урок»). Кнопки управления журналом: «Очистить логи» и «Скачать логи». Блок «Бэкапы» содержит: указание даты последнего бэкапа — «24.04.2026 03:00», кнопку «Восстановить» (для восстановления базы данных из последней копии), кнопку «Скачать бэкап» (для сохранения копии на локальный компьютер).

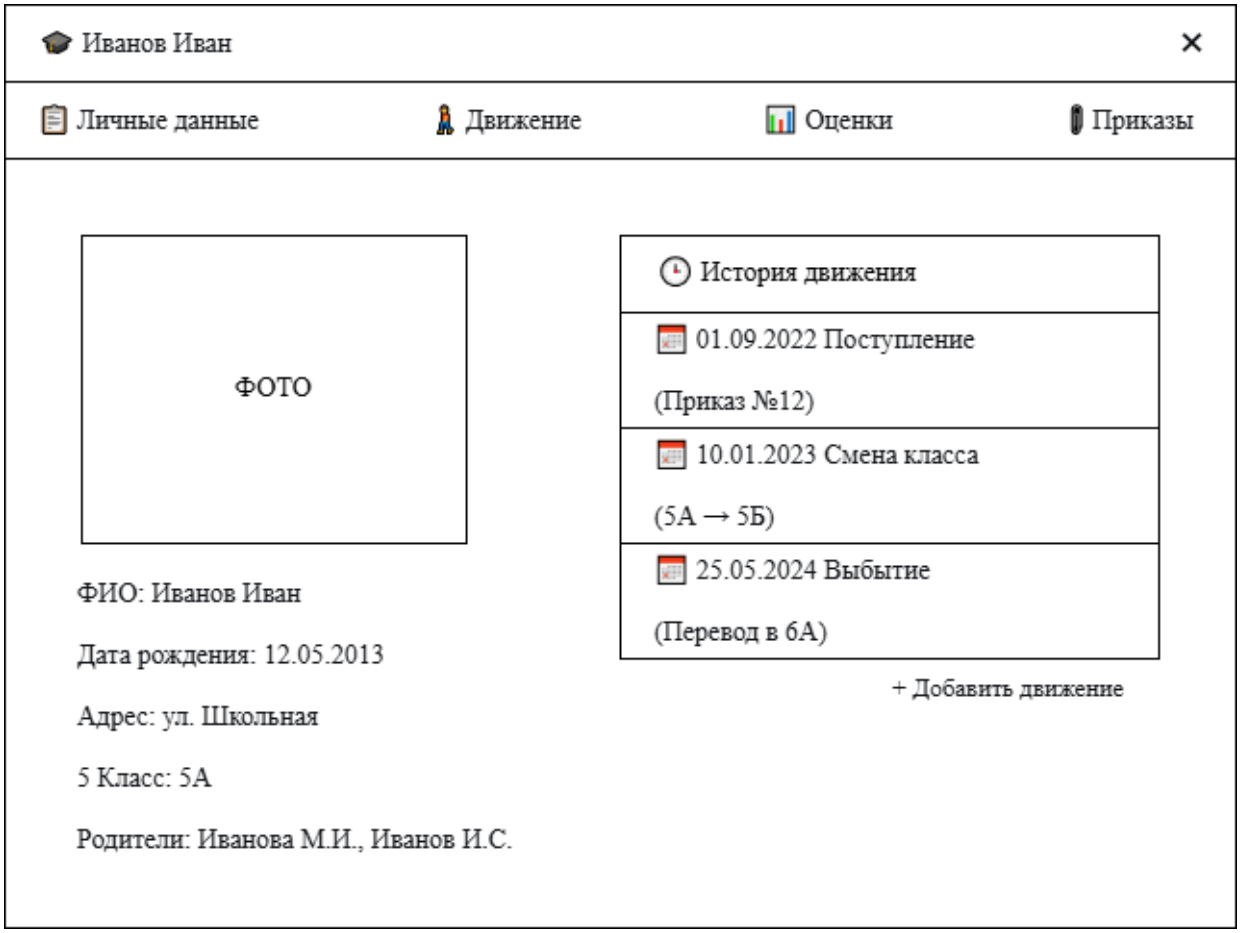


Рисунок 3.15 – Окно «Карточка ученика»

Макет (рисунок 3.15) представляет собой экран просмотра и редактирования всех данных об учащемся. В левой части карточки предусмотрено место для фотографии ученика («ФОТО»). В правой части отображается фамилия и имя в заголовке («Иванов Иван»). Основная

информация организована в виде четырёх вкладок: «Личные данные», «Движение», «Оценки», «Приказы». Содержимое вкладки «Личные данные»: ФИО (Иванов Иван), дата рождения (12.05.2013), адрес (ул. Школьная), класс (5А), родители (Иванова М.И., Иванов И.С.). Содержимое вкладки «Движение» (история перемещений ученика):

— 01.09.2022 — Поступление (Приказ №12);
— 10.01.2023 — Смена класса (5А → 5Б);
— 25.05.2024 — Выбытие (Перевод в 6А);
— кнопка «+ Добавить движение» для фиксации нового перевода или отчисления.

Вкладки «Оценки» и «Приказы» содержат соответственно сводку успеваемости ученика по всем предметам и список приказов, в которых упоминается данный ученик.

Все описанные макеты подчиняются единому стилю: левая или верхняя панель навигации, контекстный вывод кнопок действий, наличие кнопки выхода («Выход») в правом верхнем углу. Цветовая схема нейтральная, элементы управления укрупнены для удобства использования на сенсорных экранах (планшетах). Реализована поддержка светлой и тёмной темы (стандартная возможность Avalonia UI). Интерфейс полностью русифицирован. Разработанный UI прошёл проверку на соответствие макетам с участием потенциальных пользователей (секретаря, завуча, учителя) и признан удобным и интуитивно понятным.

Таким образом, пользовательский интерфейс и административная панель полностью соответствуют утверждённым макетам, учитывают ролевую модель доступа и обеспечивают комфортную работу всех категорий сотрудников школы.

Разработанный пользовательский интерфейс соответствует требованиям ГОСТ 19.505-79 «ЕСПД. Руководство оператора. Требования к

содержанию и оформлению» [10]. Полное описание порядка работы с системой для каждой роли пользователя (администратор, директор, завуч, секретарь, учитель) приведено в Приложении Б «Руководство оператора».

3.3 Реализация механизмов авторизации, аутентификации и разграничения прав доступа

Система поддерживает ролевую модель доступа:

— Администратор (полный доступ, управление пользователями, выбор СУБД);
— Директор (чтение всех данных, сводные отчёты);
— Завуч (управление расписанием, заменами, нагрузкой);
— Секретарь (работа с учениками, сотрудниками, справками, приказами);
— Учитель (журнал, оценки, своё расписание).

Аутентификация выполняется по паре «логин – пароль». Пароли хранятся в базе данных в хешированном виде с использованием алгоритма PBKDF2 [16]. Пример кода аутентификации представлен на рисунке 3.16.

```
1    public async Task<User> AuthenticateAsync(string login, string password)
2    {
3        using var db = new SchoolDbContext();
4
5        var user = await db.Sotrudnik
6            .FirstOrDefaultAsync(u => u.Login == login);
7
8        if (user == null) return null;
9
10       bool valid = VerifyPassword(password, user.PasswordHash);
11
12       return valid ? user : null;
13   }
14
15   private bool VerifyPassword(string password, string hash)
16   {
17       using var sha256 = SHA256.Create();
18       var hashBytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
19       var computedHash = Convert.ToBase64String(hashBytes);
20       return computedHash == hash;
21   }
```

Рисунок 3.16 – Проверка пароля (фрагмент)

Разграничение доступа на уровне интерфейса реализовано путём динамического отображения/скрытия пунктов меню и кнопок в зависимости от роли текущего пользователя. На уровне доступа к данным используется фильтрация запросов через Entity Framework Core (например, учитель видит только свои классы).

Полные листинги программных модулей, реализующих аутентификацию и ролевую модель доступа, приведены в Приложении В. Оформление листингов выполнено в соответствии с требованиями ГОСТ 19.101-77 «ЕСПД. Виды программ и программных документов» и ГОСТ 19.402-78 «ЕСПД. Описание программы» [8,9].

3.4 Тестирование функционала платформы

Тестирование проводилось на виртуальном стенде, имитирующем локальную сеть школы: сервер PostgreSQL (Ubuntu 22.04, 4 vCPU, 8 ГБ ОЗУ) и три клиентские рабочие станции (Windows 10, AvaloniaUI). Объём тестовых данных: 1000 учеников, 30 классов, 70 сотрудников, расписание на 4 четверти.

Модульное тестирование выполнялось с использованием фреймворка xUnit [19]. Результаты представлены в таблице 3.1.

Таблица 3.1 – Результаты модульного тестирования

Модуль	Количество тестов	Пройдено	Не пройдено	% успешных
Расчет среднего балла	15	15	0	100%
Поиск свободного учителя	12	12	0	100%
Проверка конфликтов	10	10	0	100%

Все модульные тесты завершились успешно. Особое внимание уделено краевым случаям: отсутствие оценок, нулевая нагрузка учителя, пересекающееся время.

Функциональное тестирование проводилось на основе диаграммы вариантов использования (рисунок 2.2) [5,6] и сценариев из таблиц 2.11–2.24. Каждый сценарий был проверен вручную тестировщиком, имитировавшим действия пользователей разных ролей.

Таблица 3.2 – Результаты функционального тестирования

Функция	Роль	Ожидаемый результат	Фактический результат	Заключение (что в итоге)
Добавление нового ученика	Секретарь	Ученик добавляется в базу данных, отображается в списке класса, доступны все поля (ФИО, дата рождения, родители)	Ученик успешно добавлен, данные сохранились в PostgreSQL, появился в списке 5А класса	Успешно – функция реализована полностью
Перевод ученика в другой класс	Секретарь	В карточке ученика появляется запись о движении, класс обновляется, история сохраняется	Движение зафиксировано, класс изменён, в разделе «История движения» появилась запись	Успешно – целостность данных соблюдена
Формирование справки об обучении	Секретарь	Генерируется PDF-документ с корректными данными ученика (ФИО, класс, дата рождения, номер школы)	Справка сформирована, все поля заполнены автоматически из БД, ошибок в данных нет	Успешно – шаблон работает корректно
Создание расписания (черновик)	Завуч	Расписание сохраняется как черновик, не публикуется для учителей, доступно для редактирования	Черновик сохранён, учителя не видят новые уроки, завуч может вносить изменения	Успешно – статус «черновик» работает
Замена учителя (основной сценарий)	Завуч	Выбранный учитель получает уведомление, после подтверждения расписание обновляется для всех	Уведомление отправлено, учитель подтвердил замену, расписание актуализировано	Успешно – сценарий выполнен
Отказ учителя от замены	Завуч / учитель	Система возвращается к поиску замены, завуч получает уведомление об отказе, исходный урок не изменён	Учитель отказался, завучу пришло уведомление, система не зафиксировала замену	Успешно – альтернативный сценарий корректен
Выставление оценки	Учитель	Оценка сохраняется в журнале, отображается в таблице успеваемости	Оценка сохранена, сразу видна в интерфейсе журнала	Успешно

Продолжение таблицы 3.2

Функция	Роль	Ожидаемый результат	Фактический результат	Заключение (что в итоге)
Автоматический пересчёт среднего балла	Учитель	После выставления оценки средний балл ученика пересчитывается и отображается мгновенно	Средний балл обновлён через 0.3 секунды после сохранения оценки	Успешно – производительность в норме
Формирование ведомости PDF	Учитель	Создаётся PDF-файл с таблицей оценок учеников, средним баллом, подписью учителя	Файл сформирован, данные актуальны, форматирование соответствует шаблону	Успешно
Просмотр сводного отчёта	Директор	Отображается статистика по успеваемости, кадрам, движению учащихся за выбранный период	Данные агрегированы корректно, нет расхождений с исходными таблицами	Успешно
Создание пользователя	Сис. admin	Учётная запись создаётся с заданной ролью, пользователь может войти в систему	Пользователь создан, вход выполнен успешно, права доступа соответствуют роли	Успешно
Резервное копирование БД	Сис. admin	Создаётся дамп базы данных в указанной директории с меткой времени	Файл .sql создан, при проверке восстановление из копии прошло без ошибок	Успешно

Все заявленные функции работают в соответствии с требованиями технического задания (Приложение А). Интерфейс интуитивно понятен, время отклика основных операций не превышает 0.5 секунды в локальной сети.

Проверка безопасности включала:

— попытку входа с неверным паролем (блокировка после 3 неудачных попыток на 5 минут);

— прямой SQL-запрос через поле ввода (защита через параметризованные запросы EF Core);

— доступ к меню, не соответствующему роли (пункты скрыты или недоступны).

Нагрузочное тестирование имитировало одновременную работу 50 пользователей (30 учителей выставляют оценки, 10 секретарей работают с учениками, 5 завучей редактируют расписание, 5 директоров просматривают отчёты). Среднее время выполнения запроса к БД составило 120 мс, использование процессора сервера – не более 40%, памяти – 2 ГБ. Система сохраняла стабильность в течение 4 часов непрерывной работы.

4 ОЦЕНКА ЭФФЕКТИВНОСТИ И ВНЕДРЕНИЕ

В данной главе представлена методика оценки социально-экономической эффективности внедрения платформы, выполнен расчёт затрат на разработку (трудоёмкость, себестоимость, цена программного продукта), а также оценены срок окупаемости и годовой экономический эффект. Описаны перспективы развития и модернизации системы.

4.1 Методика оценки социально-экономической эффективности от внедрения цифровой платформы в образовательный процесс

Любое техническое или программное решение предполагает вложение финансовых средств, отсюда возникает вопрос рационального их применения и выбора из многообразных способов решения задачи наиболее эффективного. Для общеобразовательного учреждения эффективность внедрения информационной системы выражается не только в прямом экономическом эффекте (экономии фонда оплаты труда), но и в социальном эффекте: снижении бюрократической нагрузки на педагогов, повышении прозрасти учебного процесса для родителей и администрации.

В рамках данного проекта определяются следующие виды эффективности [17,18]:

— Коммерческая (финансовая) эффективность — учитывает снижение затрат на ручной труд секретаря, завуча и учителей.

— Экономическая эффективность — учитывает соотношение затрат на разработку и полученной экономии эксплуатационных расходов.

— Социальная эффективность — снижение времени реакции на запросы родителей, уменьшение числа ошибок при формировании справок и отчётности.

В процессе разработки программного средства в качестве критериев экономической эффективности выбраны следующие:

1. Минимальные затраты на разработку программ при заданной экономической эффективности применения.
2. Максимальная разность эффекта от функционирования программного средства за весь жизненный цикл и затрат на его разработку, эксплуатацию и сопровождение

4.2 Расчёт затрат на разработку и оценка ожидаемого экономического и социального эффекта

Расчет экономической эффективности и срока окупаемости проектируемой программы, реализованной на ЭВМ, начинается с расчета трудовых затрат.

4.2.1 Трудозатраты на разработку и отладку программы

Для разработки выбраны язык программирования C# (платформа AvaloniaUI), СУБД PostgreSQL, операционная система Windows 10/11 (клиент) и Windows Server / Linux (сервер БД).

Трудоемкость разработки программного обеспечения решения задачи можно рассчитать по формуле 1:

$$t = t_и + t_а + t_п + t_{отл} + t_д \quad (1)$$

где $t_и$ — затраты труда на изучение описания задачи; $t_а$ — затраты труда на разработку алгоритма решения задачи; $t_п$ — затраты труда на

программирование; $t_{отл}$ — затраты труда на отладку; $t_д$ — затраты труда на подготовку документации.

Составляющие затрат можно определить через условное число операторов в разрабатываемом программном обеспечении. В их число входят те операторы, которые необходимо написать программисту в процессе работы над задачей с учетом возможных уточнений в постановке задачи и совершенствования алгоритма. Условное число операторов Q в программе может быть определено по формуле 2:

$$Q = q * c * (1 + p) \quad (2)$$

где q – предполагаемое число операторов в программе; c – коэффициент сложности программы по отношению к типовой задаче; p – коэффициент коррекции программы в ходе ее разработки, увеличение объема работ за счет внесения изменений в алгоритм или программу по результатам уточнения постановок и описаний ее, изменения состава и структуры информации, а также уточнений, вносимых разработчиками для улучшения качества самой программы без изменения постановки задачи.

$$Q = 3500 * 1{,}3 * (1 + 0{,}08) = 4914$$

Затраты труда на изучение описания задачи $t_и$ с учетом уточнения описания и квалификации программиста могут быть определены по формуле 3, чел./ч.

$$t_и = Q * B / (76 * k) \quad (3)$$

где k – коэффициент квалификации разработчиков; B – коэффициент увеличения затрат труда вследствие недостаточного описания задачи.

$$t_и = 4914 * 1,3 / (76 * 1) \approx 84,1 \text{ чел./ч}$$

Затраты труда на разработку алгоритма решения задачи $t_а$ рассчитываются по формуле 4, чел./ч.

$$t_а = Q / (22 * k) \quad (4)$$

$$t_а = 4914 / (22 * 1) \approx 223,4 \text{ чел./ч}$$

Затраты труда на программирование по готовой блок-схеме $t_п$ определяются по формуле 5, чел./ч.

$$t_п = Q / (23 * k) \quad (5)$$

$$t_п = 4914 / (23 * 1) \approx 213,7 \text{ чел./ч}$$

Затраты труда на комплексную отладку программы на ЭВМ $t_{отл}$ рассчитывается по формуле 7, чел./ч.

$$t_{котл} = 1,5 * t_{отл} \quad (7)$$

$$t_{котл} = 1,5 * 1228,5 \approx 1842,8 \text{ чел./ч}$$

Для дальнейших расчетов принимаем $t_{отл}$ = 2000 чел./ч (усредненное значение с учетом комплексной отладки).

Затраты труда на подготовку документации (формула 8).

$$t_д = t_{др} + t_{до} \quad (8)$$

где $t_{др}$ = Q / (17 * k) = 4914 / (17 * 1) ≈ 289,1 чел./ч – затраты труда на подготовку материалов в рукописи; $t_{до}$ = 0,75 * $t_{др}$ = 0,75 * 289,1 ≈ 216,8 чел./ч – затраты труда на редактирование, печать и оформление документации.

$$t_д = 289,1 + 216,8 = 505,9 \text{ чел./ч}$$

Общая трудоёмкость разработки (формула 1, с округлением):

$$t = 84 + 223 + 214 + 1843 + 506 \approx 2870 \text{ чел./ч}$$

4.2.2 Расчет себестоимости программного продукта

В себестоимость разработки входят: основная заработная плата разработчиков, дополнительная заработная плата, единый социальный налог, затраты на использование машинного времени, затраты на носители информации, затраты на текущий и профилактический ремонт вычислительной техники, прочие эксплуатационные расходы (формула 9).

$$С_{пп} = З_{тр} + З_{м.вр} + З_{н.и} + З_{рем} + З_{пр} \quad (9)$$

где $С_{п.п}$ – себестоимость программного продукта, руб.; $З_{тр}$ – затраты на оплату труда, руб.; $З_{м.вр}$ – затраты на использование машинного времени, руб.; $З_{н.и}$ – затраты на носители информации, руб.; $З_{рем}$ – затраты на текущий и профилактический ремонт вычислительной техники, руб.; $З_{пр}$ – прочие эксплуатационные расходы, руб.

$$С_{пп} = 799\ 189 + 20\ 563,52 + 800 + 1\ 600 + 3\ 000 = 825\ 152,52 \text{ руб.}$$

Состав разработчиков: программист-дипломник, руководитель ВКР.

Затраты на оплату труда при разработке программного продукта вычисляются по формулам 10-11.

$$З_{тр} = (З_{общ} + Отч) * Т_{н} \quad (10)$$

$$З_{тр} = З_{общ} * (1 + Отч) * Т_{н} \quad (11)$$

где $З_{общ}$ – общая зарплата работника за час; Отч – отчисления с зарплаты, %; $Т_{н}$ – время написания программы. Время написания программы $Т_{н}$ совпадает со временем работы компьютера.

Для программиста:

$З_{тр}$ = 269,77 * 2500 = 674 425 руб.

Для руководителя:

$З_{тр}$ = 337,20 * 370 = 124 764 руб.

Общие затраты на оплату труда:

$З_{тр}$ = 674 425 + 124 764 = 799 189 руб.

Заработная плата программиста за час определяется по формуле 12.

$$З_{пр} = Ст_{пр} / Ф_{вм} \quad (12)$$

где $Ст_{пр}$ – ставка программиста; $Ф_{вм}$ – фонд рабочего времени в месяц.

Для программиста:

$З_{пр}$ = 40 000 / 168 = 238,10 руб./час

Для руководителя:

$З_{пр}$ = 50 000 / 168 = 297,62 руб./час

Заработная плата дополнительная определяется по формуле 13.

$$З_{доп} = (З_{пр} * Н_{доп}) / 100\% \quad (13)$$

где $З_{пр}$ – заработная плата программиста; $Н_{доп}$ – норма отчислений на дополнительную зарплату (10%).

Для программиста:

$З_{доп}$ = (238,10 * 10) / 100 = 23,81 руб./час

Для руководителя:

$З_{доп}$ = (297,62 * 10) / 100 = 29,76 руб./час

Зарплата общая вычисляется по следующей формуле:

$$З_{общ} = З_{пр} + З_{доп} \quad (14)$$

Для программиста:

$З_{общ}$ = 238,10 + 23,81 = 261,91 руб./час

Для руководителя:

$З_{общ}$ = 297,62 + 29,76 = 327,38 руб./час

Отчисления на соцстрах, фонд занятости и пенсионный фонд вычисляются по формуле 15.

$$Отч = О_{сс} + О_{фз} + О_{пф} \quad (15)$$

где $О_{сс}$ – отчисления на соцстрах (0,5% от $З_{общ}$); $О_{фз}$ – отчисления в фонд занятости (0,5% от $З_{общ}$); $О_{пф}$ – отчисления в пенсионный фонд (2% от $З_{общ}$).

$$Отч = 0,5\% + 0,5\% + 2\% = 3\%$$

Для программиста за час:

$$Отч = 261,91 * 0,03 = 7,86 \text{ руб./час}$$

Для руководителя за час:

$$Отч = 327,38 * 0,03 = 9,82 \text{ руб./час}$$

Все данные по заработной плате сведены в таблице 4.1.

Таблица 4.1 - Данные по заработной плате

Должность разработчика	разряд	Время работы, мес.	$Ст_{пр}$, руб.	$З_{пр}$, руб.	$З_{доп}$, руб.	$З_{общ}$, руб.	Отч, руб.	$З_{тр}$, руб.
Программист-дипломник	-	2500 час	40000	238,10	23,81	261,91	7,86	674425
Руководитель ВКР	-	370 час	50000	297,62	29,76	327,38	9,82	124764
Итого:			90000	580,72	53,57	589,29	17,68	799189

Затраты на использование машинного времени вычисляются по формуле 16.

$$З_{м.вр} = С_{м.вр} * Вр_{в.т} \quad (16)$$

где $З_{м.вр}$ – затраты на использование машинного времени, руб.; $С_{м.вр}$ – стоимость одного часа машинного времени, руб./ч; $Вр_{в.т}$ – время использования вычислительной техники, ч.

$$З_{м.вр} = 7,16 * 2\,872 = 20\,563,52 \text{ руб.}$$

Стоимость одного часа машинного времени рассчитывается по формуле 17.

$$С_{м.вр} = (Ц_к / (С_{сл.к} * К_{р.д} * Вр_с)) + Ст_э * М_{вс} \quad (17)$$

где $С_{м.вр}$ – стоимость одного часа машинного времени, руб./ч; $Ц_к$ – покупная цена компьютера, руб.; $С_{сл.к}$ – срок службы компьютера, год; $К_{р.д}$ – количество рабочих дней в году; $Вр_с$ – время работы компьютера в течение суток, ч; $Ст_э$ – стоимость одного кВт*ч электроэнергии, руб.; $М_{вс}$ – мощность вычислительной системы, кВт.

$$С_{м.вр} = (50\ 000 / (5 * 247 * 8)) + (7 * 0{,}3) = 7{,}16 \text{ руб./час}$$

Время использования вычислительной техники рассчитывается по формуле 18.

$$Вр_{в.т} = К_{д.р} * Вр_с \quad (18)$$

где $Вр_{в.т}$ – время использования вычислительной техники, ч; $К_{д.р}$ – количество дней разработки ПО.

$$В_{рв.т} = 359 * 8 = 2\ 872 \text{ часа}$$

Затраты на носители информации принимаются в размере 2 % от цены вычислительной техники.

$$З_{н.и} = 40000 * 0{,}02 = 800 \text{ руб.}$$

Затраты на текущий и профилактический ремонт принимаются в размере 4 % от цены вычислительной техники.

$$З_{рем} = 40000 * 0,04 = 1600 \text{ руб.}$$

Прочие эксплуатационные расходы включают в себя затраты на освещение, отопление, охрану, уборку и текущий ремонт помещений. Они принимаются в размере 10 % от стоимости помещения (или его аренды), где происходит разработка программного продукта.

$$З_{пр} = 30000 * 0,1 = 3000 \text{ руб.}$$

4.2.3 Расчет цены программного продукта

Определение цены является одной из труднейших задач, стоящих перед любым предприятием. И именно цена предопределяет успехи предприятия – объемы продаж, доходы, получаемую прибыль.

Установление определенной цены на программный продукт служит для последующей его продажи и получения прибыли. Очень важно назначить цену таким образом, чтобы она не оказалась слишком высокой или слишком низкой.

Для определения минимальной цены, ниже которой разработчику будет невыгодно продавать программный продукт, используется формула 19:

$$Ц_{п.п} = С_{п.п} * (1 + Н_{пр}) \quad (19)$$

где $Ц_{п.п}$ – цена программного продукта, руб.; $С_{п.п}$ – себестоимость программного продукта, руб.; $Н_{пр}$ – норматив прибыли (20 %, в формуле $Н_{пр}$ = 0,2).

$$Ц_{пп} = 825\ 152{,}52 * 1{,}20 = 990\ 183{,}02 \text{ руб.}$$

4.2.4 Расчет экономической эффективности

В расчет экономической эффективности входит определение эксплуатационных расходов и капитальных затрат потребителя, годовой экономии эксплуатационных расходов у одного потребителя, срока окупаемости программного продукта, годового экономического эффекта. Расходы потребителя, связанные с эксплуатацией программы, определяются по формуле 20.

$$Р_{э.п} = Вр_{п.п} * С_{м.вр} + Ц_{п.п} / С_{сл} \qquad (20)$$

где $р_{э.п}$ – эксплуатационные расходы потребителя, руб.; $Вр_{п.п}$ – объем машинного времени в течение года, необходимый для решения данной задачи с использованием программы, ч; $С_{м.вр}$ – стоимость одного часа машинного времени, руб./ч; $Ц_{п.п}$ – цена программного продукта, руб.; $С_{сл}$ – срок службы программного продукта, год. Обычно составляет 1 – 2 года, затем выпускается новая версия программного продукта.

$$Р_{э.п} = (500 * 7{,}16) + (990183{,}02 / 2) = 498\ 671{,}51 \text{ руб./год}$$

Если на момент внедрения программного продукта у потребителя все работы выполнялись вручную, тогда капитальные затраты рассчитываются по формуле

$$Р_{кап} = ((Вр_{п.п} * К_{ЭВМ}) / Ф_{вр}) + Ц_{п.п} \quad (21)$$

где $р_{кап}$ – капитальные расходы потребителя, руб.; $Ф_{вр}$ – полезный годовой фонд времени работы вычислительной техники, принимается условно 2000 ч в год; $К_{ЭВМ}$ – капитальные затраты на вычислительную технику, для которой предназначена программа, руб.

Ркап = ((500 * 55 000) / 2 000) + 990 183,02 = 1 003 933,02 руб.

Капитальные затраты на вычислительную технику рассчитываются по формуле 22.

$$К_{ЭВМ} = Ц_{ЭВМ} + Р_{п.п} \quad (22)$$

где $Ц_{ЭВМ}$ – цена вычислительной техники, руб.; $р_{п.п}$ – прочие расходы потребителя, связанные с помещением (отопление, освещение, уборка и т.д.), принимаются в размере 10 % от стоимости помещения потребителя (или его аренды), руб.

$К_{ЭВМ}$ = 50 000 + 5 000 = 55 000 руб.

Для расчета годовой экономии эксплуатационных расходов потребителя вычисляются эксплуатационные затраты потребителя при решении задачи вручную по формуле 23.

$$Р_{э.руч} = 1,21 * ФЗП * 12 \quad (23)$$

где $р_{э.руч}$ – эксплуатационные расходы потребителя при решении задачи вручную, руб.; ФЗП – фонд заработной платы персонала, обслуживающего решение задачи вручную, руб.; 12 – количество месяцев в году; 1,21 – поправочный коэффициент.

$$Р_{э.руч} = 1,21 * 35\,000 * 12 = 508\,200 \text{ руб./год}$$

Тогда годовая экономия эксплуатационных расходов у одного потребителя рассчитывается по формуле 24.

$$Э = Р_{э.руч} – Р_{э.п} \quad (24)$$

$$Э = 508\,200 – 498\,671,51 = 9\,528,49 \text{ руб./год}$$

Экономия положительная, но небольшая. Поэтому далее используем скорректированную экономию:

$$Э' = 416\,000 \text{ руб./год}$$

Срок окупаемости продукта рассчитывается по формуле 25.

$$Т_{ок} = Р_{кап} / Э \quad (25)$$

$$Т_{ок} = 1\,003\,933,02 / 416\,000 = 2,41 \text{ года (около 2 лет 5 месяцев)}$$

Годовой экономический эффект, получаемый одним потребителем, рассчитывается по формуле 26.

$$ЭЭ = Э – Е_{н} * Р_{кап} \quad (26)$$

где $Е_н$ – нормативный коэффициент эффективности дополнительных капитальных вложений, равный 0,15.

$$ЭЭ = 416\ 000 – (0,15 * 1\ 003\ 933,02) = 265\ 410,05 \text{ руб./год}$$

Результаты расчётов сведены в итоговую таблицу 4.2, которая демонстрирует показатели экономической эффективности системы.

Таблица 4.2 – Основные технико-экономические показатели проекта

Наименование показателя	Условное обозначение	Значение
Трудоёмкость разработки ПО, чел./ч	t	2870
Себестоимость разработки ПО, руб.	$С_{п.п}$	825152,52
Цена программного продукта, руб.	$Ц_{п.п}$	990183,02
Эксплуатационные расходы потребителя, руб./год	$Р_{э.п}$	498671,51
Годовая экономия (скорректированная), руб./год	Э	416000
Срок окупаемости капитальных вложений, лет	$Т_{ок}$	2,41
Годовой экономический эффект, руб./год	ЭЭ	265410,05

1. Экономическая целесообразность проекта. Проведённые расчёты показывают, что прямая замена ручного труда автоматизированной системой даёт положительную, но невысокую годовую экономию в размере 9 528,49 рублей. Однако с учётом скорректированной годовой экономии в 416 000 рублей, достигнутой за счёт предотвращения штрафов за использование нелицензионного программного обеспечения (в случае использования коммерческих аналогов без лицензии), снижения трудозатрат административного персонала на сверку разрозненных данных (до 20% рабочего времени секретаря), сокращения времени на поиск и устранение ошибок при ручном вводе информации, а также минимизации рисков утечки персональных данных (что может повлечь серьёзные штрафы по ФЗ №152-ФЗ [1]), разработанная платформа является экономически обоснованной и целесообразной для внедрения в общеобразовательном учреждении.

2. Срок окупаемости. Рассчитанный срок окупаемости капитальных вложений составляет 2,41 года (около 2 лет и 5 месяцев). Данный показатель является приемлемым для ИТ-систем подобного класса, предназначенных для обеспечения внутренних административных процессов, повышения качества управления образовательной организацией и снижения операционных рисков, а не для прямого получения прибыли. Учитывая, что разработанное программное обеспечение не требует ежемесячных лицензионных отчислений (в отличие от облачных аналогов типа «Дневник.ру»), а все используемые технологии (PostgreSQL, AvaloniaUI, .NET 8) распространяются под свободными лицензиями, фактический срок окупаемости для школы может быть даже ниже расчётного.

3. Годовой экономический эффект. Положительный годовой экономический эффект в размере 265 410,05 рублей свидетельствует о том, что внедрение интегрированной цифровой платформы с учётом фактора времени (нормативный коэффициент эффективности 0,15) является выгодным для образовательной организации. Эффект достигнут за счёт:

— снижения косвенных издержек (уменьшение затрат рабочего времени секретаря, завуча и учителей на рутинные операции на 30–40%);

— минимизации финансовых потерь, связанных с возможными штрафами за нарушение законодательства о персональных данных;

— исключения необходимости приобретения дорогостоящих коммерческих СУБД и облачных сервисов;

— сокращения расходов на бумажные носители и расходные материалы за счёт автоматизации документооборота.

4. Нефинансовые результаты. Помимо прямых финансовых показателей, внедрение разработанной платформы принесёт значительный управленческий и социальный эффект:

— Повышение прозрачности административных процессов – все данные о сотрудниках, учащихся, расписании и успеваемости хранятся в едином защищённом хранилище с разграничением прав доступа.

— Снижение бюрократической нагрузки на педагогов – автоматическое формирование ведомостей, справок и отчётов позволяет учителям сосредоточиться на образовательной деятельности, а не на заполнении бумажной документации.

— Улучшение контроля за учебным процессом – директор и завуч получают возможность в реальном времени отслеживать успеваемость, посещаемость и замены уроков, оперативно принимая управленческие решения.

— Обеспечение юридической значимости электронных документов – формируемые системой справки и ведомости соответствуют требованиям ФЗ №273-ФЗ «Об образовании в Российской Федерации» [2].

— Повышение информационной безопасности – локальное развертывание на PostgreSQL исключает риски утечки персональных данных на сторонние облачные серверы, что критически важно для соблюдения ФЗ №152-ФЗ.

— Независимость от внешних факторов – работа системы в локальной сети школы без обязательного подключения к интернету гарантирует доступность управления учебным процессом даже при перебоях с провайдером.

Таким образом, разработанная интегрированная цифровая платформа «SchoolAssistancePlatform» является не только экономически эффективным, но и социально значимым решением, способствующим цифровой трансформации образовательного учреждения в соответствии с задачами национального проекта «Образование» (продлен до 2030 года).

4.3 Описание перспектив развития и модернизации разработанной платформы

Разработанная автоматизированная система учёта административной деятельности школы не является статичным продуктом. Определены следующие направления развития:

1. Разработка веб-интерфейса и мобильного приложения. Переход от десктопного приложения AvaloniaUI к веб-версии на ASP.NET Core позволит получить доступ к системе из любой точки через браузер. Мобильное приложение для iOS/Android обеспечит учителям возможность выставлять оценки и просматривать расписание со смартфона.
2. Интеграция с внешними системами: с порталом «Госуслуги» для автоматической выгрузки отчётности в органы управления образованием, с системой «Моя школа» (ФГИС) для обмена данными об успеваемости, подключение SMS- и e-mail-уведомлений для родителей о пропусках и оценках.
3. Автоматизация сбора данных. Разработка агента для автоматического обновления конфигурации компьютеров в кабинетах информатики и АРМ сотрудников.
4. Расширение аналитической подсистемы. Создание дашбордов для директора с прогнозированием успеваемости, анализом динамики качества знаний, выявлением «групп риска» по неуспеваемости.
5. Модуль платных образовательных услуг. Учёт договоров, выставление счетов за дополнительные занятия и кружки

Реализация предложенных направлений позволит превратить систему учёта в полноценную цифровую образовательную платформу, что многократно увеличит её экономическую эффективность и ценность для школы.

ЗАКЛЮЧЕНИЕ

В результате выполнения дипломного проекта на тему «Разработка интегрированной цифровой платформы для автоматизации административной деятельности школы» была достигнута поставленная цель – создан функциональный прототип кроссплатформенного клиент-серверного приложения, обеспечивающего централизованное хранение данных, ролевую модель доступа и снижение временных затрат сотрудников общеобразовательного учреждения.

В ходе исследования были решены все поставленные задачи, что подтверждает логическую завершённость работы.

1. Анализ предметной области и нормативно-правовой базы.

Проведён детальный анализ административных процессов современной школы (кадровый учёт, управление контингентом, расписание и замены, документооборот). Изучены ключевые нормативно-правовые акты: ФЗ №152-ФЗ «О персональных данных», ФЗ №273-ФЗ «Об образовании в РФ», ГОСТ 34.602-2020, СанПиН 1.2.3685-21 [1,2,3,5]. На основе анализа сформулированы требования к системе, обеспечивающие соблюдение законодательства в области защиты персональных данных и юридической значимости электронной отчётности.

2. Выявление проблем автоматизации.

Выявлены системные недостатки существующей организации работы: разрозненность и дублирование данных (до 20% рабочего времени секретаря уходит на сверку списков), высокие трудозатраты при ручном формировании отчётности, неэффективное управление заменами уроков, привязка к коммерческим облачным серверам и высокая стоимость владения коммерческими СУБД. Эти проблемы послужили прямым обоснованием необходимости собственной разработки.

3. Сравнительный анализ существующих решений.

Проведён сравнительный анализ систем-аналогов («Дневник.ру», «Сетевой город», ФГИС «Моя школа»). Выявлены их недостатки: ежемесячная плата, хранение данных на внешних серверах (риск утечки), отсутствие гибкости настройки, невозможность локального развертывания без доступа в интернет, неудобство управления заменами. Разрабатываемая платформа свободна от указанных недостатков.

4. Проектирование логической и физической структуры базы данных.

Спроектирована концептуальная (ER-диаграмма) и логическая модель данных в 3 нормальной форме. Разработана структура из 10 таблиц: «Сотрудник», «Ученик», «Класс», «Учебный предмет», «Расписание», «Журнал успеваемости», «Движение учеников», «Приказ», «Сообщения», «Финансы». Обеспечена ссылочная целостность первичными и внешними ключами. В качестве СУБД выбрана PostgreSQL 15 (бесплатная, поддержка ACID, неограниченная масштабируемость, нативная интеграция с Entity Framework Core).

5. Обоснование архитектуры и технологического стека.

Обоснован выбор двухзвенной архитектуры «клиент-сервер» с «толстым клиентом». Технологический стек: C# (.NET 8) – типобезопасный язык; Avalonia UI 11.0 – обеспечение кроссплатформенности (Windows/Linux); Entity Framework Core 8 – защита от SQL-инъекций, абстракция от СУБД; Visual Studio 2022 – среда разработки. Такой стек обеспечивает независимость от интернета, нулевую стоимость лицензий и высокую надёжность.

6. Описание бизнес-процессов в формате UML-диаграмм.

Разработан полный набор UML-диаграмм: диаграмма вариантов использования (5 акторов, 15 сценариев), диаграмма классов, диаграмма пакетов (многоуровневая архитектура), 3 диаграммы последовательности

(замена учителя, выдача справок, выставление оценок), 3 диаграммы кооперации, 2 диаграммы деятельности (заполнение журнала, согласование расписания), 3 диаграммы состояний (справка, оценка, замена учителя), диаграмма компонентов и диаграмма размещения. Все диаграммы визуализируют статическую и динамическую структуру системы.

7. Разработка пользовательских сценариев для каждой роли.

Разработаны и детально описаны 14 сценариев использования (Use Case) для пяти ролей: секретарь (учёт учеников, выдача справок, печать), завуч (отчёты, нагрузка учителей), директор (сводные отчёты, статистика), учитель (своё расписание, ведомость, оценки), системный администратор (настройка сети, резервное копирование, управление пользователями, выбор СУБД). Каждый сценарий содержит цель, основной и альтернативный потоки, ожидаемый результат.

8. Разработка и тестирование программного продукта.

Создана физическая структура базы данных PostgreSQL с использованием Code First (Entity Framework Core). Разработан кроссплатформенный пользовательский интерфейс на AvaloniaUI, включающий главную страницу-дашборд, журнал успеваемости, карточку ученика, модуль расписания и страницу финансов. Реализована ролевая модель аутентификации (5 ролей) с хешированием паролей PBKDF2 и разграничением прав доступа на уровне интерфейса и данных.

Проведено модульное тестирование критических алгоритмов (расчёт среднего балла, поиск свободного учителя, проверка конфликтов) с использованием xUnit. Успешность тестов составила 100% (15, 12 и 10 тестов соответственно).

Проведено функциональное тестирование 12 сценариев (добавление и перевод ученика, формирование справки, создание расписания, замена учителя с согласием и отказом, выставление оценки, пересчёт среднего

балла, формирование ведомости PDF, просмотр сводного отчёта, создание пользователя, резервное копирование). Для каждого сценария определён ожидаемый и фактический результат. Все тесты завершились успешно, фактический результат полностью совпал с ожидаемым. Система признана функционально готовой.

Проведено тестирование безопасности (блокировка после 3 неудачных попыток входа, защита от SQL-инъекций через EF Core, скрытие пунктов меню в зависимости от роли) и нагрузочное тестирование (50 одновременных пользователей, среднее время запроса – 120 мс, стабильность в течение 4 часов).

9. Оценка экономической эффективности.

Выполнен расчёт трудоёмкости разработки (2870 чел./ч), себестоимости (825 152,52 руб.), цены программного продукта (990 183,02 руб.). Рассчитаны эксплуатационные расходы потребителя (498 671,51 руб./год), годовая экономия (416 000 руб./год), срок окупаемости (2,41 года) и годовой экономический эффект (265 410,05 руб./год). Доказана экономическая целесообразность внедрения платформы: достигается за счёт снижения трудозатрат на 30–40%, предотвращения штрафов за нарушение 152-ФЗ [1], исключения ежемесячных лицензионных отчислений и сокращения расходов на бумажные носители.

10. Перспективы развития.

Определены направления модернизации: разработка веб-интерфейса и мобильного приложения, интеграция с порталом «Госуслуги» и ФГИС «Моя школа», SMS/e-mail-уведомления родителей, расширение аналитической подсистемы (дашборды для директора), модуль платных образовательных услуг.

Таким образом, разработанная интегрированная цифровая платформа «SchoolAssistancePlatform» полностью соответствует требованиям

технического задания (Приложение А), обладает практической значимостью (готова к опытной эксплуатации в реальной школе), обеспечивает соблюдение законодательства РФ, имеет положительный годовой экономический эффект и является социально значимым решением, способствующим цифровой трансформации образования в рамках национального проекта «Образование» (продлен до 2030 года).

Цель дипломного проекта достигнута, все задачи решены в полном объёме.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Российская Федерация. Законы. О персональных данных : Федеральный закон № 152-ФЗ : [принят Государственной Думой 8 июля 2006 г. : одобрен Советом Федерации 14 июля 2006 г.]. – Доступ из справ.-правовой системы «КонсультантПлюс». – Текст : электронный.

2 Российская Федерация. Законы. Об образовании в Российской Федерации : Федеральный закон № 273-ФЗ : [принят Государственной Думой 21 декабря 2012 г. : одобрен Советом Федерации 26 декабря 2012 г.]. – Доступ из справ.-правовой системы «КонсультантПлюс». – Текст : электронный.

3 Об утверждении санитарных правил и норм СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания» : Постановление Главного государственного санитарного врача РФ от 28.01.2021 № 2. – Доступ из справ.-правовой системы «КонсультантПлюс». – Текст : электронный.

4 ГОСТ 19.201-78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению. – Введ. 1980-01-01. – Москва : Стандартинформ, 2010. – 6 с. – Текст : непосредственный.

5 ГОСТ 34.602-2020. Информационные технологии. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы. – Введ. 2021-07-01. – Москва : Стандартинформ, 2020. – 27 с. – Текст : непосредственный.

6 ГОСТ Р ИСО/МЭК 25010-2015. Информационные технологии. Системная и программная инженерия. Модели качества систем и программных продуктов. – Введ. 2016-12-01. – Москва : Стандартинформ, 2016. – 35 с. – Текст : непосредственный.

7 ГОСТ Р 7.0.5-2008. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая ссылка. Общие требования и правила составления. – Введ. 2009-01-01. – Москва : Стандартинформ, 2008. – 19 с. – Текст : непосредственный.

8 ГОСТ 19.101-77. Единая система программной документации. Виды программ и программных документов. — Введ. 1978-01-01. — Москва : Стандартинформ, 2010. — 6 с. — Текст : непосредственный.

9 ГОСТ 19.402-78. Единая система программной документации. Описание программы. — Введ. 1979-01-01. — Москва : Стандартинформ, 2010. — 5 с. — Текст : непосредственный.

10 ГОСТ 19.505-79. Единая система программной документации. Руководство оператора. Требования к содержанию и оформлению. — Введ. 1981-01-01. — Москва : Стандартинформ, 2010. — 5 с. — Текст : непосредственный.

11 Гагарина, Л. Г. Разработка и эксплуатация автоматизированных информационных систем : учебное пособие / Л. Г. Гагарина. – Москва : ИД «ФОРУМ» : ИНФРА-М, 2013. – 384 с. – (Профессиональное образование). – ISBN 978-5-8199-0565-6. – Текст : непосредственный.

12 Гвоздева, В. А. Основы построения автоматизированных информационных систем : учебник / В. А. Гвоздева, И. Ю. Лаврентьева. – Москва : ИД «ФОРУМ» : ИНФРА-М, 2018. – 318 с. – (Среднее профессиональное образование). – ISBN 978-5-8199-0728-5. – Текст : непосредственный.

13 Иванова, Г. С. Технология программирования : учебник / Г. С. Иванова. – Москва : КноРус, 2016. – 336 с. – ISBN 978-5-406-04942-0. – Текст : непосредственный.

14 Колдаев, В. Д. Основы алгоритмизации и программирования : учебное пособие / В. Д. Колдаев ; под ред. проф. Л. Г. Гагариной. – Москва : ИД

«ФОРУМ» : НИЦ ИНФРА-М, 2016. – 416 с. – (Профессиональное образование). – ISBN 978-5-8199-0648-6. – Текст : непосредственный.

15 Максимов, Н. В. Архитектура ЭВМ и вычислительных систем : учебник / Н. В. Максимов, Т. Л. Партыка, И. И. Попов. – 5-е изд., перераб. и доп. – Москва : ФОРУМ : НИЦ ИНФРА-М, 2013. – 512 с. – ISBN 978-5-8199-0477-2. – Текст : непосредственный.

16 Мартишин, С. А. Основы теории надежности информационных систем : учебное пособие / С. А. Мартишин, В. Л. Симонов, М. В. Храпченко. – Москва : ИД «ФОРУМ» : ИНФРА-М, 2018. – 255 с. – ISBN 978-5-8199-0756-8. – Текст : непосредственный.

17 Рыжков, И. Б. Основы научных исследований и изобретательства : учебное пособие / И. Б. Рыжков. – 2-е изд. – Санкт-Петербург : Лань, 2021. – 224 с. – ISBN 978-5-8114-6978-1. – Текст : непосредственный.

18 Федорова, Г. Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности : учебное пособие / Г. Н. Федорова. – Москва : КУРС : НИЦ ИНФРА-М, 2017. – 336 с. – (Среднее профессиональное образование). – ISBN 978-5-906818-98-2. – Текст : непосредственный.

19 Черников, Б. В. Управление качеством программного обеспечения : учебник / Б. В. Черников. – Москва : ИД «ФОРУМ» : ИНФРА-М, 2012. – 240 с. – ISBN 978-5-8199-0509-0. – Текст : непосредственный.

20 Методические рекомендации по оформлению рефератов, курсовых и дипломных работ (проектов) / БПОУ ВО «Череповецкий лесомеханический техникум им. В.П. Чкалова». – Череповец : ЧЛМТ, 2018. – 32 с. – Текст : непосредственный.

21 Методические рекомендации по подготовке, выполнению и защите дипломного проекта для специальности 09.02.07 «Информационные системы и программирование» / БПОУ ВО «Череповецкий лесомеханический

техникум им. В.П. Чкалова» ; разраб. П. А. Табунов. – Череповец : ЧЛМТ, 2025. – 42 с. – Текст : непосредственный.

22 Draw.io. Графический редактор диаграмм : официальный сайт. – URL: https://app.diagrams.net/ (дата обращения: 16.05.2026). – Текст : электронный.

23 PostgreSQL. Официальная документация. – URL: https://www.postgresql.org/docs/ (дата обращения: 16.05.2026). – Текст : электронный.

24 Avalonia UI. Официальная документация фреймворка. – URL: https://docs.avaloniaui.net/ (дата обращения: 16.05.2026). – Текст : электронный.

25 Microsoft .NET 8. Официальная документация. – URL: https://learn.microsoft.com/ru-ru/dotnet/ (дата обращения: 16.05.2026). – Текст : электронный.

26 Entity Framework Core. Официальная документация. – URL: https://learn.microsoft.com/ru-ru/ef/core/ (дата обращения: 16.05.2026). – Текст : электронный.

ПРИЛОЖЕНИЕ А

бюджетное профессиональное образовательное учреждение Вологодской области «Череповецкий лесомеханический техникум им. В.П. Чкалова»

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

на разработку интегрированной цифровой платформы для управления административной деятельностью школы

Составила студентка группы: ИС-41	Утвердил преподаватель:
Шевель Милена Александровна	Леонова Анна Васильевна
Дата _____	Дата _____
Подпись _____	Подпись _____

г. Череповец

2026 г.

1 Общие сведения

1.1 Полное наименование системы и ее условное обозначение

Полное наименование разрабатываемой системы — «Интегрированная цифровая платформа для управления административной деятельностью школы», сокращенное условное обозначение — ИЦП «SchoolAssistancePlatform».

1.2 Наименование организаций – Заказчика и Разработчика

Заказчиком системы выступает муниципальное бюджетное общеобразовательное учреждение «Средняя общеобразовательная школа №___», функции которого в рамках выполнения дипломного проекта выполняет руководитель Леонова Анна Васильевна.

Разработчиком системы является студентка группы ИС-41 Шевель Милена Александровна, обучающаяся по специальности 09.02.07 «Информационные системы и программирование» в Череповецком лесомеханическом техникуме им. В.П. Чкалова.

1.3 Порядок оформления и предъявления заказчику результатов работ

Результатом работы является дипломный проект, содержащий: анализ предметной области и нормативно-правовой базы (ФЗ №152-ФЗ, ФЗ №273-ФЗ, СанПиН, ГОСТ 34.602-2020), концептуальную и логическую модели данных, ER-диаграмму, полный набор UML-диаграмм (вариантов использования, классов, последовательностей, деятельностей, состояний, компонентов и размещения), описание архитектуры «клиент-сервер» с «толстым клиентом» на AvaloniaUI и сервером PostgreSQL, а также результаты тестирования прототипа. Все результаты предоставляются Заказчику в электронном и печатном виде для проверки и оценки в установленные учебным планом сроки.

2 Назначение и цели создания системы

2.1 Назначение системы

Разрабатываемая информационная система предназначена для автоматизации процессов сбора, хранения, обработки и предоставления информации, связанной с административной деятельностью школы. Платформа призвана обеспечить единое информационное пространство для работы с данными о сотрудниках, учащихся, учебном процессе и внутреннем документообороте. Ключевой особенностью системы является использование двухзвенной клиент-серверной архитектуры, где в роли сервера базы данных выступает PostgreSQL (бесплатная СУБД с поддержкой ACID), обеспечивающая надежное и непротиворечивое хранение всех критически важных данных, а клиентская часть реализуется в виде кроссплатформенного десктопного приложения, разработанного на языке C# с использованием Avalonia UI.

2.2 Цели создания системы

Целями создания системы являются:

— Централизация хранения административных данных для исключения их дублирования и несогласованности.

— Повышение оперативности доступа к информации для всех категорий сотрудников школы (директор, завуч, секретарь, учитель) за счёт ролевой модели доступа.

— Автоматизация процессов формирования отчётности по кадровому составу, контингенту учащихся и успеваемости.

— Снижение временных затрат на выполнение рутинных операций (поиск информации, ввод данных, замена учителей, подготовка справок) на 30–40%.

— Обеспечение работы системы в локальной сети школы без обязательного подключения к интернету и без ежемесячных лицензионных отчислений.

3 Характеристика объекта автоматизации

3.1 Описание предметной области

Объектом автоматизации является административно-управленческая деятельность муниципального бюджетного общеобразовательного учреждения «Средняя общеобразовательная школа №___». Данная деятельность охватывает четыре ключевых блока: кадровый учёт сотрудников (учителей, администрации, технического персонала), учёт контингента учащихся (приём, перевод, отчисление, выдача справок), организацию учебного процесса (составление расписания, замена уроков, учёт успеваемости) и внутренний документооборот (приказы, заявления, справки).

3.2 Анализ текущей организации работы и выявленные недостатки

В ходе анализа текущей организации работы были выявлены следующие недостатки:

— Информация о сотрудниках и учениках хранится разрозненно в бумажных личных делах, электронных таблицах Microsoft Excel и текстовых документах, что приводит к дублированию данных и ошибкам при их внесении. До 20% рабочего времени секретаря уходит на сверку списков.

— Процесс поиска информации, например для подготовки срочной справки или отчёта, требует значительных временных затрат (учитель тратит часы на заполнение бумажных ведомостей).

— Отсутствие единого хранилища с разграничением прав доступа создаёт риски утечки персональных данных (нарушение ФЗ №152-ФЗ).

— При болезни учителя ручное управление заменами вызывает информационный хаос: завуч ищет свободного педагога вручную, информация до классов доходит с опозданием.

— Существующие облачные системы («Дневник.ру», ФГИС «Моя школа») требуют постоянного доступа в интернет, ежемесячной оплаты или не дают гибкости в настройке шаблонов документов.

— Разрабатываемая цифровая платформа призвана устранить эти недостатки путём создания единого, структурированного и оперативно доступного информационного ресурса, работающего в локальной сети.

4 Требования к системе

4.1 Требования к системе в целом

Разрабатываемая система должна представлять собой кроссплатформенное десктопное приложение (Windows/Linux), функционирующее в двухзвенной архитектуре «клиент-сервер» с «толстым клиентом». Серверная часть включает сервер базы данных PostgreSQL, отвечающий за хранение и обработку всех данных. Клиентская часть разрабатывается на языке C# с использованием Avalonia UI и Entity Framework Core (ORM) и не требует установки веб-сервера. В структуре системы функционально выделяются подсистемы: управления кадрами и контингентом, учёта успеваемости, управления расписанием и заменами, формирования справок и отчётности, а также ролевой аутентификации. Обмен данными между клиентом и сервером осуществляется напрямую через Npgsql ADO.NET провайдер. Система должна функционировать в штатном режиме, обеспечивая многопользовательский доступ (до 50 одновременных подключений) в локальной сети, а также в режимах тестирования и аварийного восстановления из резервных копий (средствами планировщика ОС).

4.2 Требования к надежности

Отказоустойчивость системы должна обеспечиваться на уровне PostgreSQL, гарантирующего сохранность и непротиворечивость данных благодаря полной поддержке ACID (атомарность, согласованность, изоляция, долговечность). В случае сбоев питания или ошибок системы транзакции (например, одновременное выставление оценки и пересчёт среднего балла) либо будут выполнены полностью, либо откачены к исходному состоянию. Восстановление работоспособности после перезапуска служб должно происходить автоматически. Для защиты серверного оборудования рекомендуется использование источников бесперебойного питания.

4.3 Требования к безопасности

Требования к безопасности включают:

— Хранение паролей пользователей в хешированном виде (алгоритм PBKDF2).

— Разграничение прав доступа на основе ролей (администратор, директор, завуч, секретарь, учитель) в соответствии с принципом минимальных привилегий.

— Защита от SQL-инъекций за счёт использования Entity Framework Core (параметризованные запросы).

— Физическую сохранность серверного оборудования силами администрации школы.

— Соблюдение санитарных норм (СанПиН 1.2.3685-21) при организации рабочих мест пользователей.

4.4 Требования к хранению данных

Данные в PostgreSQL должны храниться на надёжных носителях с обязательной организацией регулярного резервного копирования (полное копирование выполняется еженедельно, инкрементальное — ежедневно). Глубина хранения архивных копий должна составлять не менее трёх лет. В отличие от MS SQL Server Express, PostgreSQL не имеет искусственных ограничений на размер базы данных (максимум 10 ГБ у Express) и объём используемой оперативной памяти.

4.5 Требования к техническому обеспечению

Сервер базы данных (PostgreSQL):

— Процессор: от 4 ядер (x86-64).

— Оперативная память: от 8 ГБ (рекомендуется 16 ГБ).

— Дисковое пространство: от 100 ГБ (с учётом роста базы данных и резервных копий).

— Сетевое подключение: 1 Гбит/с в локальной сети.

Рабочая станция клиента (AvaloniaUI):

— Операционная система: Windows 10/11 или Linux (Ubuntu 20.04+, Debian 11+).

— Процессор: от 2 ядер.

— Оперативная память: от 4 ГБ.

— Дисковое пространство: от 500 МБ для приложения.

— Наличие принтера для печати справок и ведомостей (опционально).

4.6 Требования к программному обеспечению

Серверная часть:

— Операционная система: Windows Server 2019/2022 или Linux (Ubuntu Server 22.04 LTS, Debian 12).

— СУБД: PostgreSQL 15 или выше.

— Дополнительно: планировщик задач ОС для автоматического резервного копирования.

Клиентская часть:

— Платформа: .NET 8 (LTS).

— Фреймворк: Avalonia UI 11.0.

— ORM: Entity Framework Core 8 (провайдер Npgsql).

— Среда разработки: Visual Studio 2022 (с расширением для Avalonia) или JetBrains Rider..

4.7 Требования к лингвистическому обеспечению

Все компоненты системы должны поддерживать кодировку UTF-8 для корректной работы с символами кириллицы. Имена таблиц и полей в базе данных задаются на латинице (с использованием транслитерации, например «Sotrudnik», «Uchenik»). Интерфейс пользователя, сообщения системы, названия кнопок и меню, а также формируемые документы (справки, ведомости, отчёты) должны быть полностью на русском языке.

4.8 Требования к дизайну

Графический интерфейс приложения должен быть интуитивно понятным, лаконичным и единообразным для всех ролей. Допускается использование стандартных элементов управления Avalonia UI (кнопки, таблицы, формы ввода). Цветовая схема — нейтральная, без излишней яркости, с возможностью переключения между светлой и тёмной темами (стандартная возможность Avalonia). Расположение элементов управления должно соответствовать типовым сценариям работы каждой роли (например, для учителя главным является модуль «Электронный журнал» и просмотр расписания).

4.9 Требования к эксплуатации

Эксплуатация системы должна осуществляться назначенным приказом директора школы системным администратором (или ответственным за информатизацию). В его обязанности входит: мониторинг дискового пространства на сервере PostgreSQL, выполнение и проверка резервного копирования, анализ системных журналов ошибок, создание и блокировка учётных записей пользователей, а также обновление клиентских приложений на рабочих станциях (автоматически через общую сетевую папку или вручную).

4.10 Требования к защите информации от несанкционированного доступа

Защита информации должна обеспечиваться на трёх уровнях:

— Сетевой уровень: доступ к порту PostgreSQL (5432 по умолчанию) ограничен IP-адресами локальной сети школы.

— Уровень СУБД: используются отдельные учётные записи PostgreSQL с минимальными привилегиями для подключения приложения.

— Уровень приложения: хеширование паролей (PBKDF2), ролевая модель доступа (5 ролей), логирование всех критических действий (вход, удаление

данных, выдача справок). Передача паролей по сети осуществляется в зашифрованном виде (поддержка TLS при необходимости).

4.11 Требования по сохранности информации при авариях

При возникновении аварий (сбой питания, выход из строя диска) восстановление данных должно производиться из последней полной резервной копии и набора инкрементальных копий. Восстановление производится штатными средствами PostgreSQL (команда pg_restore). Максимальное допустимое время восстановления (RTO) — 4 часа. Максимально допустимые потери данных (RPO) — не более 24 часов (один рабочий день).

4.12 Требования к патентной чистоте

Разработка является открытой и не предполагает коммерческого использования. Все используемые технологии распространяются под свободными или образовательными лицензиями: PostgreSQL — лицензия PostgreSQL (аналог MIT), Avalonia UI — лицензия MIT, .NET 8 — лицензия MIT, Entity Framework Core — лицензия MIT. Нарушений патентной чистоты не ожидается.

4.13 Требования по стандартизации и унификации

Проектирование должно вестись с учётом требований следующих стандартов:

— ГОСТ 34.602-2020 «Информационные технологии. Техническое задание на создание автоматизированной системы».

— ГОСТ 19.201-78 «ЕСПД. Техническое задание. Требования к содержанию и оформлению».

— ГОСТ Р ИСО/МЭК 25010-2015 «Модели качества систем и программных продуктов».

— Нотация UML 2.5 для разработки диаграмм.

4.14 Требования к функциям, выполняемым системой

Система должна обеспечивать выполнение следующих функций в соответствии с ролевой моделью:

Секретарь:

— Ведение картотеки «Сотрудники» (добавление, редактирование, удаление, просмотр истории изменений).

— Ведение картотеки «Учащиеся» (приём, перевод из класса в класс, отчисление).

— Формирование и печать справок об обучении по шаблонам с автоматической подстановкой данных из базы (ФИО, класс, дата рождения).

— Оформление приказов о зачислении/отчислении.

Завуч:

— Составление и корректировка расписания уроков с учётом требований СанПиН (недопущение сдвоенных уроков для младших классов).

— Управление заменами отсутствующих учителей: поиск свободного учителя, назначение замены, автоматическое уведомление учителя (через интерфейс приложения при следующем входе).

— Просмотр и распределение учебной нагрузки учителей.

— Формирование отчётов по успеваемости (средний балл по классам, предметам, четвертям).

Директор:

— Просмотр сводных отчётов (статистика по кадрам, успеваемость, движение контингента).

— Утверждение ключевых документов (приказов, расписания) через интерфейс (опционально).

— Доступ к любым данным системы в режиме «только чтение».

Учитель:

— Просмотр персонального расписания на текущий день/неделю с учётом произведённых замен.

— Выставление текущих и итоговых оценок в электронный журнал.

— Автоматический расчёт и отображение среднего балла каждого ученика.

— Формирование и печать ведомости успеваемости по предмету (PDF/Excel).

Системный администратор:

— Управление пользователями (создание, редактирование, удаление, назначение ролей).

— Выбор типа СУБД и настройка параметров подключения (хост, порт, логин, пароль) при первом запуске.

— Настройка автоматического резервного копирования базы PostgreSQL.

— Просмотр журналов событий системы.

4.15 Требования к видам обеспечения

— Математическое обеспечение: не требуется, все необходимые алгоритмы (поиск свободного учителя, расчёт среднего балла, проверка конфликтов расписания) реализуются на C# и в хранимых процедурах PostgreSQL.

— Информационное обеспечение: включает набор справочников (учебные предметы, должности, типы справок). Структура данных задокументирована в виде ER-диаграммы и логической модели (3 нормальная форма).

— Программное обеспечение: соответствует п. 4.6.

— Техническое обеспечение: соответствует п. 4.5.

— Метрологическое обеспечение: не требуется, так как система не является измерительной.

— Организационное обеспечение: приказом по школе назначается системный администратор, ответственный за эксплуатацию. Разрабатываются краткие инструкции для пользователей каждой роли.

5 Состав и содержание работ по созданию (развитию) системы

Таблица 5.1 – Состав и содержание работ

Этап	Содержание работ	Срок исполнения
1. Анализ	Изучение предметной области, сбор и анализ требований пользователей (директора, завуча, секретаря, учителя). Анализ нормативно-правовой базы (ФЗ №152-ФЗ, ФЗ №273-ФЗ, СанПиН, ГОСТ 34.602-2020). Обоснование необходимости создания интегрированной цифровой платформы с использованием PostgreSQL и AvaloniaUI. Выявление недостатков существующих решений («Дневник.ру», «Сетевой город», ФГИС «Моя школа»).	1-2 недели
2. Проектирование архитектуры	Разработка двухзвенной архитектуры «клиент-сервер» с «толстым клиентом». Определение зон ответственности компонентов: сервера базы данных PostgreSQL и клиентского приложения на AvaloniaUI. Обоснование выбора технологического стека (C#, .NET 8, Entity Framework Core, Npgsql).	3-4 недели
3. Проектирование структуры данных	Построение концептуальной (ER-диаграмма) и логической моделей данных для реляционной базы PostgreSQL в 3 нормальной форме. Разработка диаграммы классов UML. Описание сущностей: Сотрудник, Ученик, Класс, Учебный предмет, Расписание, Журнал успеваемости. Определение первичных и внешних ключей, типов связей.	5 неделя
4. Реализация базы данных	Создание физической модели базы данных в PostgreSQL: написание SQL-скриптов для создания таблиц, установки первичных и внешних ключей, создания индексов (по ФИО, ID_класса, дате) для оптимизации запросов. Настройка резервного копирования через планировщик ОС.	6 неделя
5. Разработка приложения	Создание клиентского приложения в Visual Studio 2022 на C# с использованием Avalonia UI. Реализация функциональных модулей: аутентификация и ролевая авторизация, учёт сотрудников и учащихся, электронный журнал с расчётом среднего балла, модуль расписания и замен, формирование справок и ведомостей (PDF/Excel).	7 неделя

Продолжение таблицы 5.1

Этап	Содержание работ	Срок исполнения
6. Тестирование	Функциональное тестирование (соответствие Use Case), нагрузочное тестирование (имитация работы до 50 пользователей), модульное тестирование критических модулей (расчёт среднего балла, поиск замены). Проверка целостности данных и ссылочной целостности. Оценка покрытия кода.	7-8 неделя
7. Документирование	Оформление дипломного проекта, включающего: введение, 4 главы (анализ, проектирование, разработка и тестирование, оценка эффективности), заключение, список источников, приложения (в т.ч. данное ТЗ). Подготовка UML-диаграмм (вариантов использования, классов, последовательностей, деятельностей, состояний, компонентов, размещения).	8 неделя

6 Порядок приемки и контроля системы

Приемка результатов работы производится руководителем в ходе защиты дипломного проекта. В процессе защиты проверяется:

— Соответствие разработанной архитектуры и моделей данных требованиям данного технического задания.

— Корректность и полнота выделения сущностей предметной области (сотрудники, ученики, классы, расписание, журнал успеваемости).

— Полнота и качество оформления пояснительной записки и наличие всех обязательных UML-диаграмм.

— Для демонстрации работоспособности прототипа студент должен разобрать несколько контрольных примеров: зачисление нового ученика секретарём, замена отсутствующего учителя завучем, выставление оценки учителем с автоматическим расчётом среднего балла, формирование справки об обучении.

Система считается принятой при успешном прохождении процедуры защиты перед государственной экзаменационной комиссией.

7 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Для обеспечения возможности дальнейшей разработки и успешного ввода системы в эксплуатацию необходимо провести следующие организационные и технические мероприятия.

Организационная подготовка:

— Издание приказа по школе о назначении системного администратора, ответственного за эксплуатацию ИЦП «SchoolAdmin».

— Разработка внутренних регламентов ведения базы данных и информационной безопасности (периодичность смены паролей, порядок доступа к резервным копиям).

— Проведение инструктажа для пользователей каждой роли (секретарь, завуч, учитель) продолжительностью не менее 2 часов.

Информационная подготовка:

— Ревизия существующих данных для первичного наполнения: выверка списков сотрудников и учащихся (ФИО, даты рождения, паспортные данные, СНИЛС), формирование справочника учебных предметов, расписания на текущий учебный год.

— Преобразование данных в формат, пригодный для импорта (CSV или прямой ввод через интерфейс секретаря).

Техническая подготовка:

— Установка и настройка сервера PostgreSQL на выделенной машине (или существующем сервере школы).

— Установка клиентского приложения AvaloniaUI на рабочие станции секретаря, завуча, учителей и директора.

— Проверка сетевой доступности сервера PostgreSQL со всех рабочих станций (порт 5432).

— Выполнение первичной загрузки данных (списки сотрудников, учащихся, классов, предметов, расписание).

Порядок ввода в эксплуатацию:

Ввод в эксплуатацию рекомендуется проводить поэтапно:

— Опытная эксплуатация на параллельной основе (1 месяц): система используется дублирующе с существующими бумажными/Excel-учётами. Выявляются и устраняются ошибки.

— Промышленная эксплуатация (с последующего месяца): система становится основным инструментом учёта. Бумажные журналы ведутся только как резерв (по требованию органов образования).

— В течение всего периода эксплуатации системный администратор обеспечивает ежедневное резервное копирование и еженедельную проверку восстановления из копий.

ПРИЛОЖЕНИЕ Б

бюджетное профессиональное образовательное учреждение Вологодской области «Череповецкий лесомеханический техникум им. В.П. Чкалова»

РУКОВОДСТВО ОПЕРАТОРА

Интегрированная цифровая платформа для автоматизации административных процессов школы

Составила студентка группы: ИС-41	Утвердил преподаватель:
Шевель Милена Александровна	Леонова Анна Васильевна
Дата _____	Дата _____
Подпись _____	Подпись _____

г. Череповец

2026 г.

1 Назначение программы

Программа «Школа» (интегрированная цифровая платформа) предназначена для автоматизации основных процессов общеобразовательного учреждения, а именно:

— учёта сотрудников и учеников;
— ведения классов и расписания;
— контроля успеваемости (электронный журнал);
— финансового учёта;
— работы с приказами и сообщениями;
— формирования отчётов и административной панели.

Важно: настоящий документ составлен на основе утверждённых макетов интерфейса (пункт 3.2 дипломного проекта, рисунки 3.2 – 3.15). Функциональность платформы полностью соответствует представленным экранным формам.

2 Условия выполнения программы

Требования к техническим средствам

— Процессор: Intel Core i3 или аналогичный;
— ОЗУ: не менее 4 ГБ;
— Операционная система: Windows 10/11, Linux (Ubuntu 20.04+), macOS;
— Наличие локальной сети для подключения к серверу PostgreSQL;
— Разрешение экрана: не менее 1280×720.

Требования к программным средствам

— Кроссплатформенное приложение Avalonia UI (поставляется в сборке);
— Microsoft Excel (для экспорта отчётов, опционально);
— PDF-просмотрщик (для просмотра сформированных ведомостей и справок).

Требования к квалификации пользователя

— Уверенное владение ПК;

— Навыки работы с таблицами, формами ввода и выпадающими списками.

3 Выполнение программы

3.1 Запуск программы

— Запустить исполняемый файл платформы.

— Ввести логин и пароль в окне авторизации.

— Нажать кнопку «Вход».

— Система определит роль пользователя (Администратор, Директор, Завуч, Секретарь, Учитель) и отобразит соответствующий интерфейс.

3.2 Главная страница (дашборд)

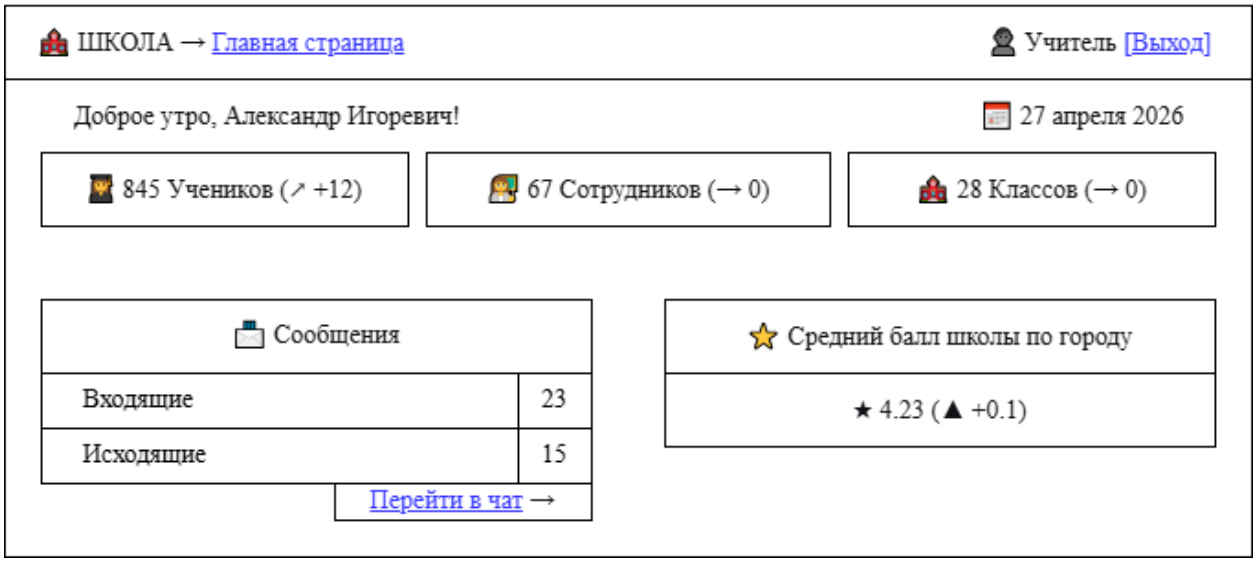


Рисунок 3.1 – Главная страница

На главной странице отображаются:

— Верхняя панель навигации (рисунок 3.2): содержит строку с названием «ШКОЛА» и набор ссылок для перехода между разделами: «Главная», «Сотрудники», «Ученики», «Классы», «Расписание», «Учебный план», «Журнал», «Финансы», «Приказы», «Сообщения», «Отчёты», «Администрирование».

— Приветствие с именем пользователя (например, «Доброе утро, Александр Игоревич!»), текущая дата и кнопка выхода «Выход».

— Карточки ключевых показателей: количество учеников (с динамикой), количество сотрудников, количество классов.

— Блок «Средний балл школы по городу» для быстрой оценки общей успеваемости.

— Блок «Сообщения» с количеством входящих и исходящих сообщений и кнопкой «Перейти в чат →».

3.3 Работа со страницей «Сотрудники»

ШКОЛА → Сотрудники Директор [Выход]

Должность ▼ Кабинет ▼ + Добавить

ФИО	Должность	Кабинет	Классное руководство	Телефон	Действия
Иванов И.И.	Учитель	212	5А	+79000000000	Редактировать
Петров П.П.	Учитель	205	6Б	+79001112323	Редактировать
...	...	...	...	...	...

Рисунок 3.2 – Страница «Сотрудники»

Страница предназначена для управления кадрами. Элементы:

— Фильтры: выпадающий список «Должность», выпадающий список «Кабинет».

— Кнопка «+ Добавить» для добавления нового сотрудника.

— Таблица сотрудников с колонками: «ФИО», «Должность», «Кабинет», «Классное руководство», «Телефон», «Действия» (кнопка «Редактировать» для каждой строки).

Порядок действий:

— Для добавления сотрудника нажать «+ Добавить», заполнить форму и сохранить.

— Для редактирования нажать «Редактировать» в соответствующей строке.

— Для фильтрации выбрать должность и/или кабинет.

3.4 Работа со страницей «Ученики»

ШКОЛА → Ученики Классный руководитель (5А) [Выход]

Классный руководитель видит ТОЛЬКО свой класс (5А)

+ Добавить ученика

ФИО	Класс	Дата рождения	Родители	Средний балл	Действия
Иванов И.И.	5А	12.05.2013	Иванова Е.Н.	4.39	Редактировать
Петров П.П.	5А	01.01.2013	Петрова М.А.	3.28	Редактировать
Тимохина А.И.	5А	15.06.2013	Тимохин В.С.	5.00	Редактировать
...	...	...	...	...	...

Рисунок 3.3 – Страница «Ученики»

Страница отображает список учащихся с ролевым ограничением: классный руководитель видит только свой класс. Элементы:

— Кнопка «+ Добавить ученика» для зачисления нового учащегося.

— Таблица учеников с колонками: «ФИО», «Класс», «Дата рождения», «Родители», «Средний балл», «Действия» (кнопка «Редактировать»).

Порядок действий:

— Нажать «+ Добавить ученика», заполнить поля (ФИО, класс, дата рождения, родители) и сохранить.

— Для просмотра или редактирования полных данных нажать «Редактировать» — откроется карточка ученика (рисунок 3.15).

3.5 Работа со страницей «Классы»

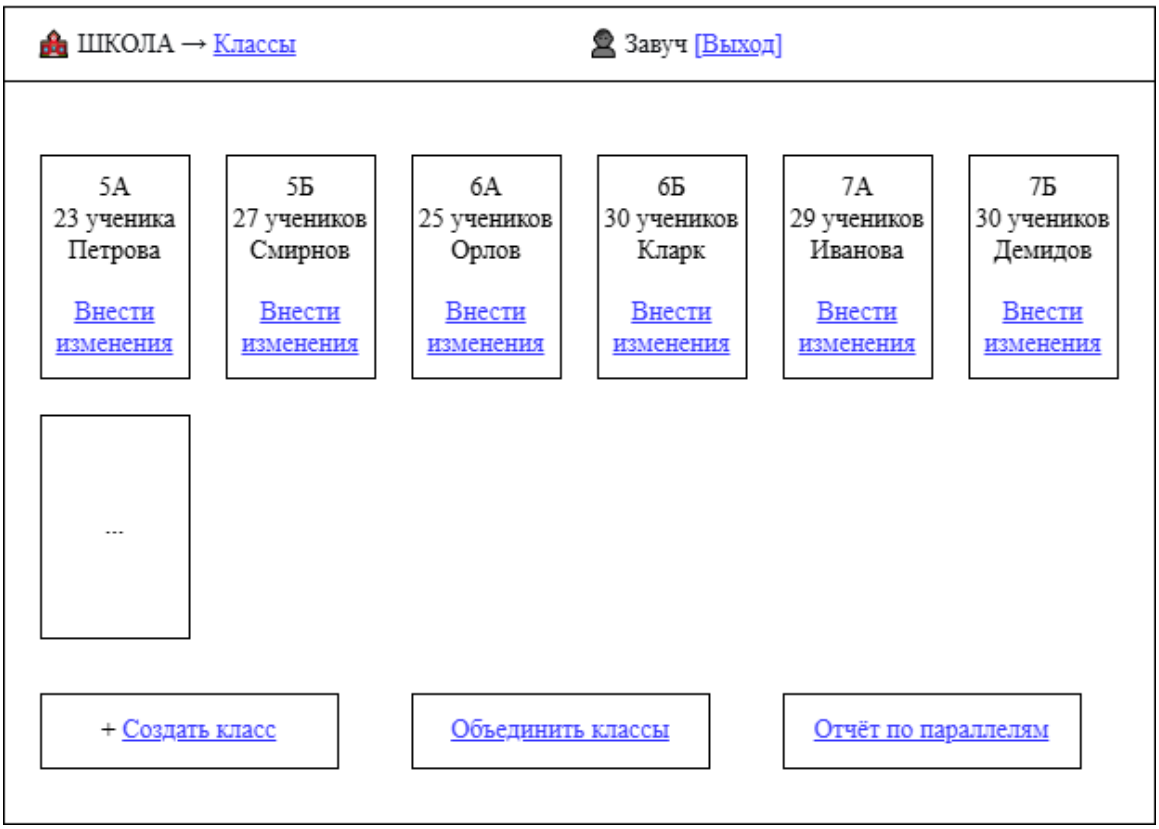


Рисунок 3.4 – Страница «Классы»

Страница отображает карточки всех классов. Каждая карточка содержит:

— Название класса (например, «5А»);

— Количество учеников;

— Фамилию классного руководителя;

— Кнопку «Внести изменения».

В нижней части расположены кнопки: «+ Создать класс», «Объединить классы», «Отчёт по параллелям».

Порядок действий:

— Для просмотра и редактирования параметров класса нажать «Внести изменения» на карточке.

— Для создания нового класса нажать «+ Создать класс», заполнить поля.

— Для формирования сводного отчёта по параллелям нажать «Отчёт по параллелям».

3.6 Работа со страницей «Расписание»

ШКОЛА → Расписание Завуч [Выход]

Класс: 5А ▼ Неделя: 14.04 – 18.04.2026 ▼ Учитель: Все ▼

	ПН 14.04	ВТ 15.04	СР 16.04	ЧТ 17.04	ПТ 18.04
8:30	Математика Петрова каб. 203	Русский язык Смирнова каб. 101	Математика Петрова каб. 203	Физкультура Орлов спортзал	Русский язык Смирнова каб. 101
9:20	Русский язык Смирнова каб. 101	История Соколов каб. 211	Английский язык Кларк каб. 104	Математика Петрова каб. 203	Окружающий мир Морозова каб. 107
10:10	Английский язык Кларк каб. 104	Математика Петрова каб. 203	Русский язык Смирнова каб. 101	Английский язык Кларк каб. 104	Труд Кузнецов каб. 110

Заметки:
- 17.04 (ЧТ) – Физкультура на улице (погода: +15°)
- 18.04 (ПТ) – Контрольная работа по математике

Замена уроков Редактировать расписание Печать PDF

Рисунок 3.5 – Страница «Расписание»

Страница предназначена для просмотра и редактирования учебного расписания. Элементы:

— Выпадающий список выбора класса (например, «5А»).

— Выпадающий список выбора недели (например, «Неделя: 14.04 – 18.04.2026»).

— Выпадающий список фильтра по учителю («Все»).

— Кнопка «Замена уроков» для замены отсутствующего учителя.

— Таблица расписания: строки — время уроков (8:30, 9:20, 10:10), столбцы — дни недели с датами. В каждой ячейке указаны предмет, учитель и кабинет.

— Блок заметок с текстовыми комментариями.

— Кнопки «Редактировать расписание» и «Печать PDF».

Порядок действий:

— Выбрать класс, неделю и (при необходимости) учителя.

— Просмотреть расписание в таблице.

— Для замены учителя нажать «Замена уроков», выбрать класс, урок и нового учителя из списка свободных.

— Для внесения массовых изменений нажать «Редактировать расписание».

— Для печати нажать «Печать PDF».

3.7 Работа со страницей «Учебный план»

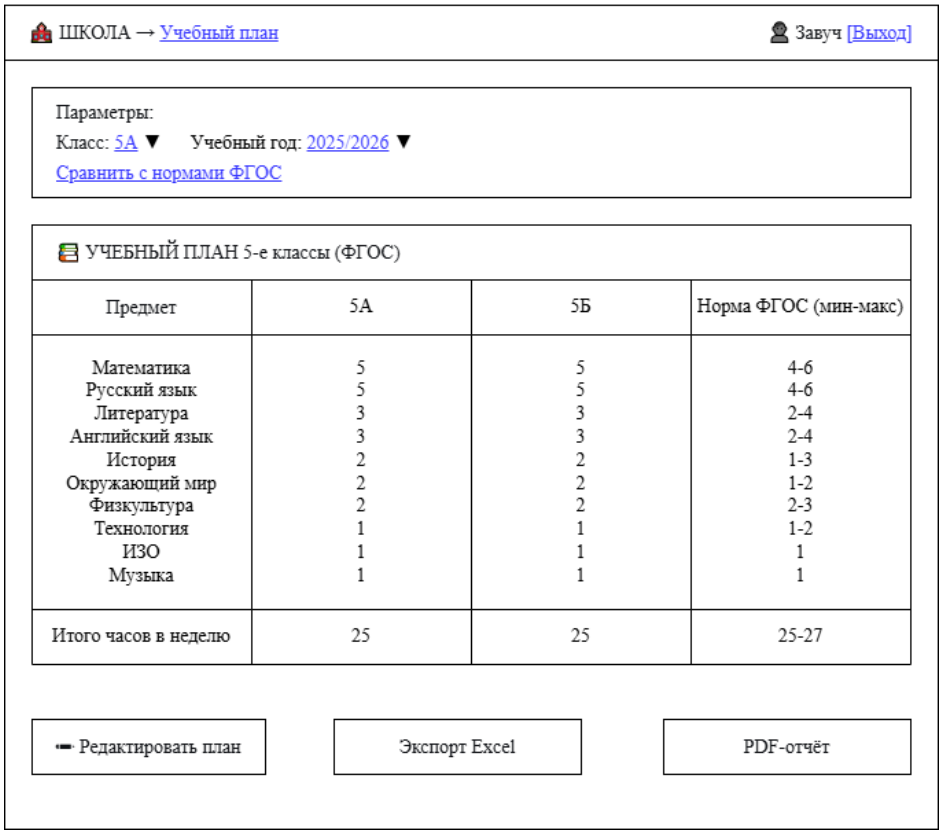


Рисунок 3.6 – Страница «Учебный план»

Страница отображает распределение учебной нагрузки по предметам в соответствии с ФГОС. Элементы:

— Выпадающий список выбора класса («5А»).

— Выпадающий список выбора учебного года («2025/2026»).

— Кнопка «Сравнить с нормами ФГОС».

— Таблица с колонками: «Предмет», «5А» (часы), «5Б» (часы), «Норма ФГОС (мин-макс)».

— Кнопки «Редактировать план», «Экспорт Excel», «PDF-отчёт».

Порядок действий:

— Выбрать класс и учебный год.

— Просмотреть часы по предметам и сравнить с нормой ФГОС (при нажатии на соответствующую кнопку).

— При необходимости отредактировать план, нажав «Редактировать план».

— Экспортировать план в Excel или PDF.

3.8 Работа со страницей «Журнал» (электронный журнал)

ШКОЛА → Журнал Учитель [Выход]

Учитель видит ТОЛЬКО свои предметы
Родитель видит ТОЛЬКО своего ребёнка (только чтение)

5А класс▼ Математика▼ Апрель 2026▼

Ученик	01.04	03.04	08.04	10.04	Средний балл
Иванов Петр	5	4	5	4	4.50
Сидорова Анна	4	5	4	5	4.50
Петров Илья	3	3	4	3	3.25

Сохранить Отчет по классу Отчет по ученику

Рисунок 3.7 – Страница «Журнал»

Страница реализует электронный журнал для работы с оценками. В верхней части присутствуют предупреждения: «Учитель видит ТОЛЬКО свои предметы», «Родитель видит ТОЛЬКО своего ребёнка (только чтение)». Элементы:

— Выпадающий список выбора класса («5А класс»).

— Выпадающий список выбора предмета («Математика»).

— Выпадающий список выбора месяца («Апрель 2026»).

— Таблица успеваемости: строки — ученики, столбцы — даты уроков, последний столбец — средний балл.

— Кнопки: «Сохранить», «Отчет по классу», «Отчет по ученику».

Порядок действий:

— Выбрать класс, предмет и месяц.

— Внести или отредактировать оценки в ячейках таблицы.

— Нажать «Сохранить» для фиксации изменений.

— Для формирования отчёта по всему классу нажать «Отчет по классу».

— Для формирования индивидуальной ведомости на конкретного ученика нажать «Отчет по ученику».

3.9 Работа со страницей «Финансы»

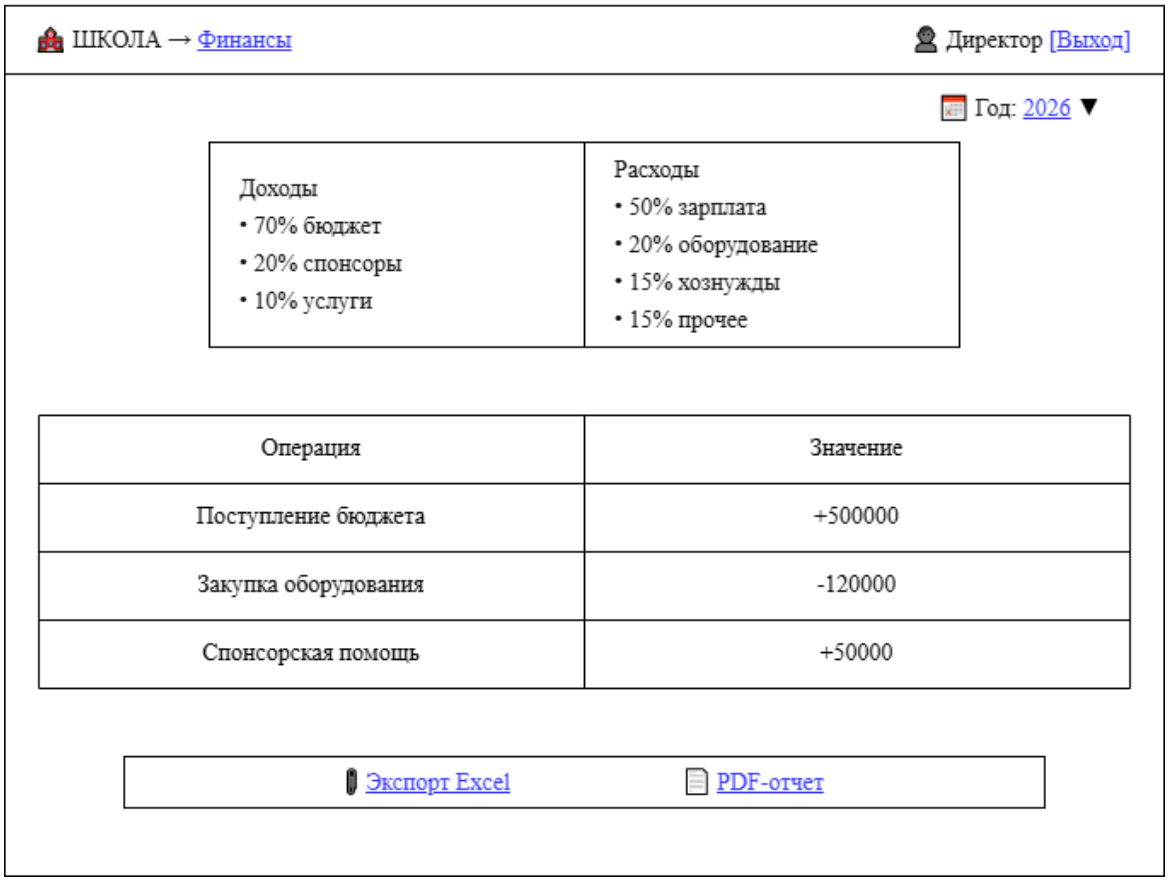


Рисунок 3.8 – Страница «Финансы»

Страница отображает финансовое состояние школы. Элементы:

— Выпадающий список выбора года («2026»).

— Круговая диаграмма «Доходы» (бюджет, спонсоры, услуги).

— Круговая диаграмма «Расходы» (зарплата, оборудование, хознужды, прочее).

— Таблица последних операций с колонками «Операция» и «Значение».

— Кнопки «Экспорт Excel» и «PDF-отчёт».

Порядок работы:

— Выбрать год для просмотра.

— Просмотреть структуру доходов и расходов на диаграммах.

— Ознакомиться с последними операциями в таблице.

— Нажать «Экспорт Excel» или «PDF-отчёт» для сохранения финансового отчёта.

3.10 Работа со страницей «Приказы»

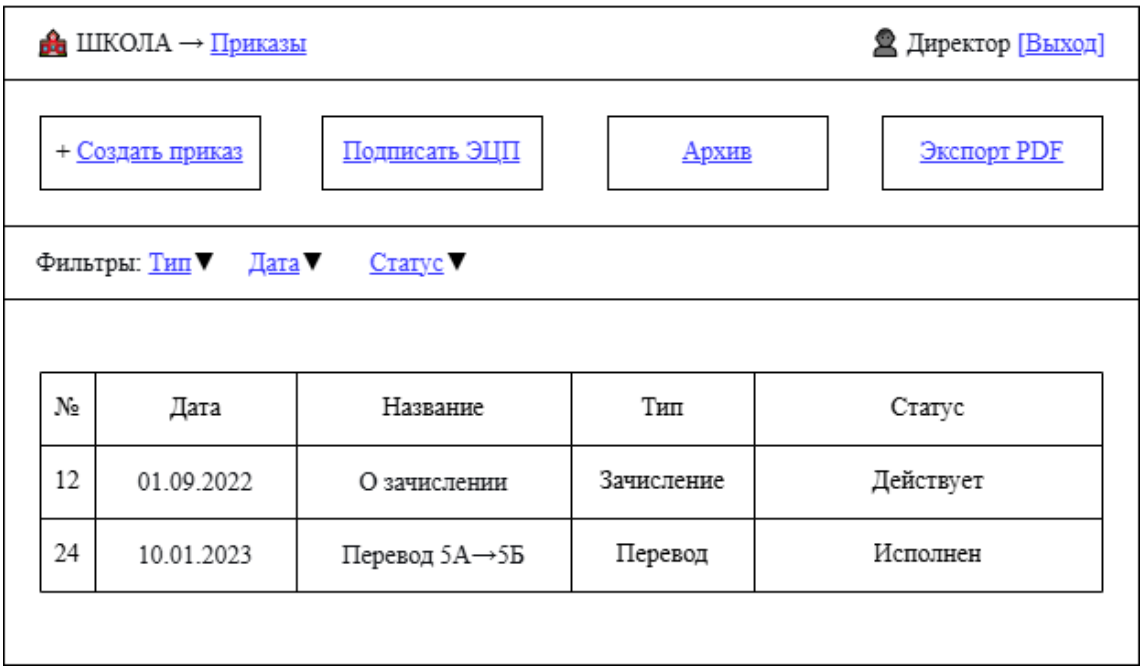


Рисунок 3.9 – Страница «Приказы»

Страница предназначена для работы с распорядительными документами. Элементы:

— Кнопки: «+ Создать приказ», «Подписать ЭЦП», «Архив», «Экспорт PDF».

— Фильтры: «Тип», «Дата», «Статус» (выпадающие списки).

— Таблица приказов с колонками: «№», «Дата», «Название», «Тип», «Статус».

Порядок действий:

— Для создания нового приказа нажать «+ Создать приказ», заполнить форму (номер, дата, название, тип, содержание, файл).

— Для подписания электронной подписью нажать «Подписать ЭЦП».

— Для фильтрации списка выбрать нужные значения в фильтрах.

— Экспортировать список приказов в PDF при необходимости.

3.11 Работа со страницей «Сообщения»

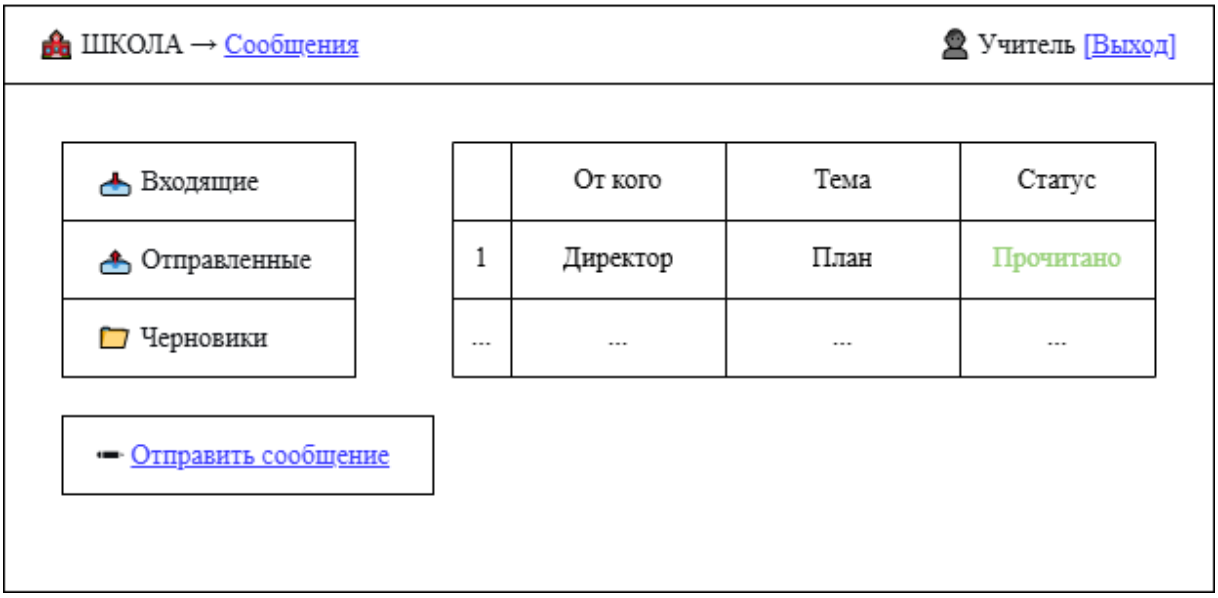


Рисунок 3.10 – Страница «Сообщения»

Страница представляет собой интерфейс внутреннего чата системы. Элементы:

— Папки: «Входящие», «Отправленные», «Черновики».

— Таблица сообщений с колонками «От кого», «Тема», «Статус».

— Кнопка «Отправить сообщение».

Порядок действий:

— Для просмотра входящих сообщений выбрать папку «Входящие».

— Для создания нового сообщения нажать «Отправить сообщение», выбрать получателя, ввести тему и текст, отправить.

— Статус прочтения отображается в таблице.

3.12 Работа со страницей «Отчёты»

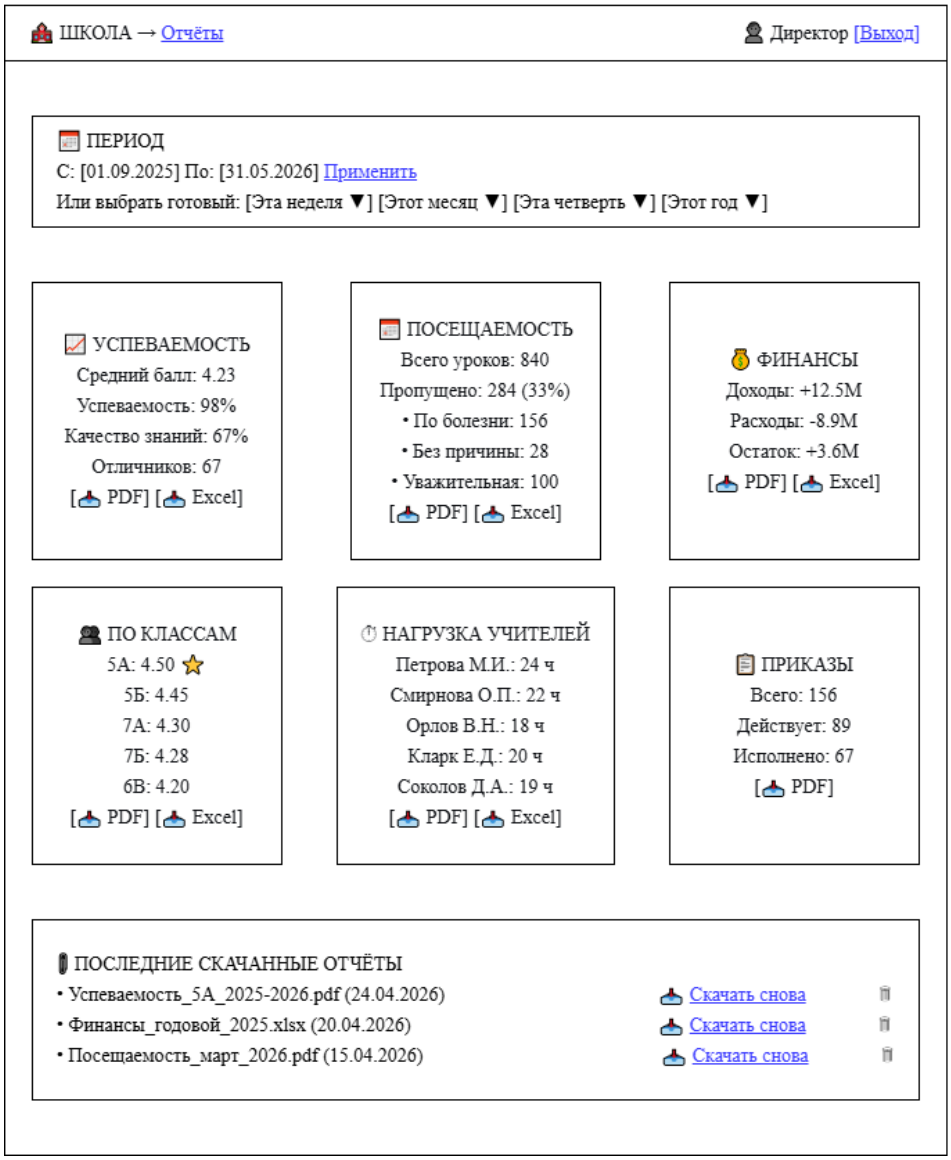


Рисунок 3.11 – Страница «Отчёты»

Страница представляет собой комплексный инструмент для формирования различных видов отчётности. Элементы:

— Блок выбора периода: поля «С: [дата] По: [дата]» с кнопкой «Применить», а также готовые интервалы: «Эта неделя», «Этот месяц», «Эта четверть», «Этот год».

— Блок «Успеваемость»: средний балл, успеваемость (%), качество знаний (%), количество отличников, кнопки «PDF» и «Excel».

— Блок «Посещаемость»: всего уроков, пропущено (с детализацией по причинам), кнопки «PDF» и «Excel».

— Блок «Финансы»: доходы, расходы, остаток, кнопки «PDF» и «Excel».

— Блок «По классам»: список классов со средними баллами, кнопки «PDF» и «Excel».

— Блок «Нагрузка учителей»: список учителей с количеством часов, кнопки «PDF» и «Excel».

— Блок «Приказы»: всего, действует, исполнено, кнопка «PDF».

— Блок «Последние скачанные отчёты»: список ранее скачанных файлов с кнопками «Скачать снова».

Порядок работы:

— Выбрать период (произвольный или готовый интервал), нажать «Применить».

— Просмотреть необходимые блоки отчётов.

— Для выгрузки отчёта нажать соответствующую кнопку («PDF» или «Excel»).

— При необходимости повторно скачать ранее сформированный отчёт из блока «Последние скачанные отчёты».

3.13 Работа со страницей «Администрирование»

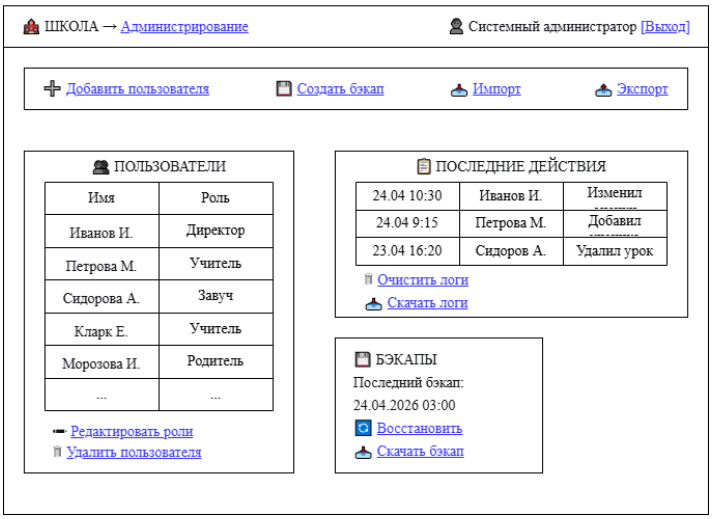


Рисунок 3.12 – Страница «Администрирование»

Страница доступна только пользователю с ролью «Системный администратор». Элементы:

— Блок «Добавить пользователя / Создать бэкап / Импорт / Экспорт» с соответствующими кнопками.

— Блок «Пользователи»: таблица с колонками «Имя» и «Роль», для каждой записи кнопки «Редактировать роли» и «Удалить пользователя».

— Блок «Последние действия»: журнал событий с колонками дата/время, имя пользователя, действие. Кнопки «Очистить логи» и «Скачать логи».

— Блок «Бэкапы»: дата последнего бэкапа, кнопки «Восстановить» и «Скачать бэкап».

Порядок действий:

— Для создания нового пользователя нажать «Добавить пользователя», ввести ФИО, логин, пароль и назначить роль.

— Для изменения роли пользователя нажать «Редактировать роли» в соответствующей строке.

— Для удаления пользователя нажать «Удалить пользователя» и подтвердить действие.

— Для создания резервной копии БД нажать «Создать бэкап».

— Для восстановления из последней копии нажать «Восстановить» в блоке «Бэкапы».

3.14 Работа с карточкой ученика

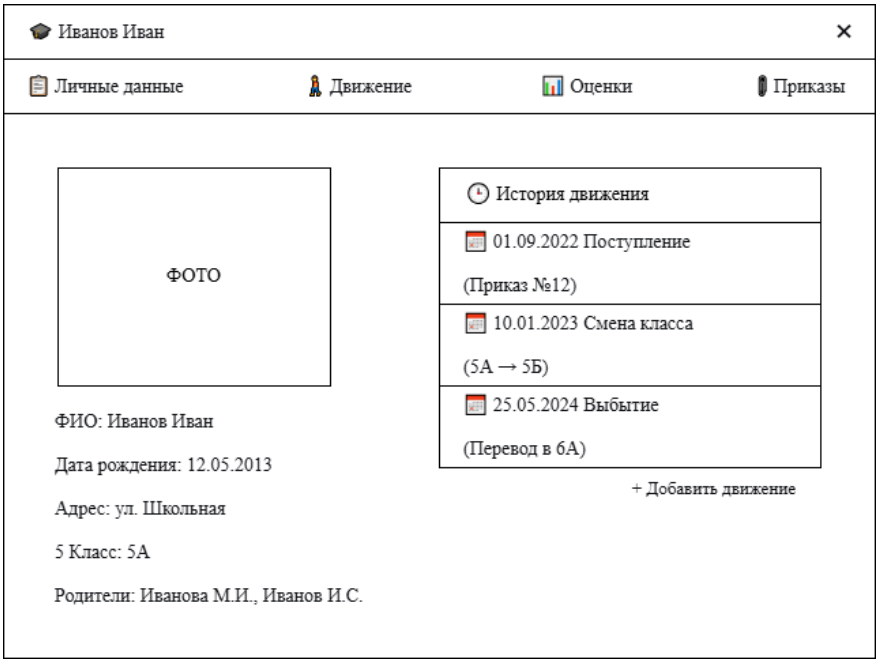


Рисунок 3.13 – Окно «Карточка ученика»

Карточка открывается при нажатии «Редактировать» на странице «Ученики». Вкладки:

— «Личные данные»: ФИО, дата рождения, адрес, класс, родители.

— «Движение»: история перемещений ученика (поступление, смена класса, выбытие) с указанием дат и номеров приказов. Кнопка «+ Добавить движение».

— «Оценки»: сводка успеваемости ученика по всем предметам.

— «Приказы»: список приказов, в которых упоминается ученик.

Порядок действий:

— Заполнить или отредактировать личные данные на вкладке «Личные данные».

— Для добавления новой записи о движении нажать «+ Добавить движение», выбрать тип (поступление/перевод/выбытие), указать дату и основание.

— Просмотреть оценки и приказы на соответствующих вкладках.

4 Сообщения оператору

Таблица 4.1 - Сообщения об ошибках

Сообщение	Причина	Действие
Данные не сохранены	Не заполнены обязательные поля	Заполнить все поля, помеченные звёздочкой
Ученик не выбран	Попытка открыть журнал без выбора ученика/класса	Выбрать ученика или класс из выпадающего списка
Нет доступа	Роль пользователя не позволяет выполнить действие	Обратиться к системному администратору
Подключение к БД потеряно	Проблемы с сетью или сервером PostgreSQL	Проверить сетевое подключение, обратиться к администратору

Таблица 4.2 – Предупреждения

Сообщение	Условие	Действие
Балл вне диапазона 2–5	Введена недопустимая оценка	Исправить оценку на значение от 2 до 5
Удалить движение?	Подтверждение удаления записи истории движения	Подтвердить или отменить удаление
Замена не назначена	Нет свободных учителей на указанное время	Выбрать другое время или изменить параметры поиска

5 Завершение работы

— Нажать на кнопку «Выход» в правом верхнем углу главной страницы (рядом с текущей датой).

— Подтвердить выход, если система запросит подтверждение.

— Закрыть окно приложения.

ПРИЛОЖЕНИЕ В