МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

**федеральное государственное бюджетное образовательное учреждение высшего образования**

**«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»**

**Кафедра «Экономики и управления на предприятии природопользования»**

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

**(бакалаврская работа)**
**по направлению подготовки 09.03.03 Прикладная информатика**
**(квалификация – бакалавр)**

**На тему «Разработка информационной системы «Электронная зачётка»»**

**Исполнитель** Гвоздовский Станислав Игоревич

**Руководитель** к.г.н., доцент по кафедре «Информатики и прикладной математики» Полупанов Владимир Николаевич

**«К защите допускаю»**

**Руководитель кафедры** _____

кандидат экономических наук

Майборода Евгений Викторович

«17» 01 2026 г.

Филиал Российского государственного гидрометеорологического университета в г. Туапсе
НОРМОКОНТРОЛЬ ПРОЙДЕН
«15» 01 2026 г.
Мартынова Т. В.
ПОДПИСЬ РАСШИФРОВКА ПОДПИСИ

Туапсе
2026

ОГЛАВЛЕНИЕ

Введение

В современном мире цифровизация образования стала одним из ключевых факторов повышения эффективности учебного процесса и административной деятельности в высших учебных заведениях [1;4;26]. Согласно данным Министерства науки и высшего образования Российской Федерации, в 2024–2025 годах значительная часть вузов активно внедряет цифровые сервисы, включая электронные журналы успеваемости, студенческие билеты и зачетные книжки [3;10;29]. Рынок образовательных технологий (EdTech) в России в 2024 году вырос на 19% и достиг 145 млрд рублей, а к 2025 году ожидается дальнейшее развитие за счет интеграции ИИ, персонализации обучения и перехода на гибридные форматы [5;15;18]. Пандемия COVID-19 ускорила эти процессы, показав необходимость дистанционного доступа к данным и автоматизации рутинных операций [11;21].

Традиционные бумажные зачетные книжки, используемые для фиксации оценок, зачетов, экзаменов и успеваемости студентов, имеют существенные недостатки: риск потери или порчи документа, необходимость ежегодного продления, ручной ввод данных преподавателями, низкая оперативность доступа к информации и отсутствие интеграции с другими системами вуза (например, электронными журналами или порталами Госуслуг) [2;3;10]. Эти проблемы приводят к увеличению административной нагрузки на деканаты, задержкам в обработке данных и неудобствам для студентов, особенно в условиях дистанционного обучения или обучения в филиалах.

В соответствии с федеральным проектом «Цифровая образовательная среда» национального проекта «Образование» и изменениями в Федеральном законе «Об образовании в Российской Федерации» (№ 273-ФЗ с поправками 2025 года) [1;4;26], в вузах России активно внедряются электронные зачетные книжки и студенческие билеты. С 2024–2025 годов проводится эксперимент по переходу на цифровые документы, а к 2026–2027 годам планируется полный отказ от бумажных аналогов в многих учреждениях.

Электронная зачетная книжка позволяет студентам в реальном времени просматривать оценки через личный кабинет, исключает ошибки ручного заполнения, снижает бюрократию и обеспечивает безопасность данных. Преимущества включают экономию бюджета вуза (отказ от печати бланков), экологичность, повышение цифровой грамотности участников процесса и интеграцию с порталом Госуслуг [3;11].

Актуальность выбранной темы обусловлена необходимостью автоматизации учета успеваемости в условиях роста числа студентов, развития дистанционного образования и требований нормативных актов Минобрнауки РФ. В филиалах вузов, таких как филиал РГГМУ в г. Туапсе, где студенты могут быть распределены географически, внедрение подобной системы позволит оптимизировать информационные потоки, сократить время на административные процедуры и повысить качество управления учебным процессом.

Объектом исследования является процесс учета и анализа успеваемости студентов в высших учебных заведениях с использованием информационных систем.

Предметом исследования - информационная система (приложение) для автоматизированного заполнения и ведения электронной зачетной книжки студентов.

Цель выпускной квалификационной работы - разработка приложения для заполнения электронной зачетной книжки, обеспечивающего автоматизированный учет оценок, интеграцию с базами данных вуза, удобный интерфейс для студентов и преподавателей, а также повышение эффективности административной работы.

Для достижения цели поставлены следующие задачи:

1. Провести анализ предметной области электронных систем учета успеваемости в образовании России.

2. Обосновать выбор задачи и описать ее экономико-информационную сущность, включая информационные потоки и требования к

данным.

3. Разработать информационное, технологическое, техническое и программное обеспечение системы (модели данных, архитектуру, модули).

4. Обосновать выбор технологий и описать интерфейс приложения с руководством пользователя.

5. Рассчитать экономическую эффективность внедрения разработанного приложения.

Методы исследования: анализ научной и нормативной литературы, моделирование информационных процессов (ER-диаграммы), системный анализ, программирование (если реализуешь прототип), экономический расчет эффективности по методике вуза.

Теоретическая основа работы опирается на исследования в области прикладной информатики, информационных систем в образовании и EdTech. Практическая значимость заключается в возможности внедрения приложения в вузах для оптимизации учета успеваемости, снижения административных затрат и повышения удовлетворенности студентов.

Структура работы соответствует требованиям методических рекомендаций и включает аналитический, проектный разделы, расчет экономической эффективности, заключение и приложения.

# 1 Аналитический раздел

## 1.1 Анализ предметной области

Предметная область — автоматизированный учет и анализ успеваемости студентов в высших учебных заведениях России [1;4;11] с использованием информационных систем. В последние годы цифровизация образования в РФ развивается в рамках национального проекта «Образование» [4;26] и федерального проекта «Цифровая образовательная среда», направленного на создание современной цифровой инфраструктуры, обеспечивающей доступность и качество образования всех уровней.

По данным аналитики, рынок образовательных технологий (EdTech) в России в 2024 году вырос на 19–21% и достиг 144–149 млрд рублей [5;15;18;23]. Это связано с внедрением электронных сервисов, онлайн-платформ и автоматизированных систем учета. Эксперимент по переводу студенческих билетов и зачетных книжек в электронный вид, запущенный в апреле 2024 года и продолжающийся до 31 декабря 2025 года, позволяет использовать цифровые документы на портале Госуслуг (в виде QR-кода) для льгот, прохода в вуз и отслеживания успеваемости. К сентябрю 2025 года сервис доступен студентам более 76 вузов, с планом подключения еще 90 [2;10;29;34].

Традиционные бумажные зачетные книжки уступают место электронным аналогам, которые уже внедрены в ведущих вузах: НИУ ВШЭ (с 2012 года), Финансовом университете, РГСУ и других. Электронные системы позволяют в реальном времени фиксировать оценки, зачеты и посещаемость, интегрироваться с порталами вузов и Госуслугами, снижая административную нагрузку и минимизируя ошибки.

Существующие информационные системы учета успеваемости в вузах России включают:

- 1С:Университет — комплексная система для автоматизации управления вузом, включая учет контингента студентов, успеваемости,

посещаемости, формирование учебных планов и электронных ведомостей. Поддерживает балльно-рейтинговую систему (БРС) и интеграцию с другими модулями.

- Русский Moodle (3KL) — адаптированная LMS-платформа для электронной информационно-образовательной среды (ЭИОС), с модулями электронного журнала, учета прогресса и игрофикации. Широко используется в вузах для дистанционного обучения и фиксации результатов.
- Электронные журналы на базе 1С (например, 1С:Электронный журнал колледжа, адаптируемый для вузов) — позволяют преподавателям вести журналы в цифровом формате, студентам просматривать оценки в личном кабинете.
- Локальные разработки вузов: в УРФУ, РУДН и других — модули анализа успеваемости с интеграцией в общую ЭИОС.

Несмотря на прогресс, многие вузы сталкиваются с проблемами: неполная интеграция систем, ручной ввод данных в бумажные/электронные зачетки, отсутствие мобильного доступа для студентов и преподавателей. Это приводит к задержкам в обработке информации, ошибкам и повышенной нагрузке на деканаты.

Анализ показывает, что разработка специализированного приложения для электронной зачетной книжки актуальна: оно позволит автоматизировать заполнение, обеспечить мобильный доступ и интеграцию с существующими системами (например, через API Госуслуг или 1С).

Таблица 1.1 – Сравнение существующих систем учета успеваемости

| Система | Функции учета успеваемости | Интеграция | Мобильный доступ | Примеры вузов |
|---|---|---|---|---|
| 1С:Университет | Полный учет, БРС, ведомости | Высокая | Через веб | Многие вузы |
| Русский Moodle | Журналы, прогресс | Средняя | Да | Вузы с ЭИОС |

Продолжение таблицы 1.1

| | | | | |
|---|---|---|---|---|
| Госуслуги (эксперимент) | QR-зачетка, просмотр | Федеральная | Да | 76+ вузов |

На основе представленной таблицы 1.1, где сравниваются ключевые системы учета успеваемости в российских вузах [3;11] (1С:Университет, Русский Moodle и экспериментальный сервис Госуслуг), можно провести анализ обязательных и рекомендуемых функций для обеспечения конкурентоспособности разрабатываемого приложения для электронной зачетной книжки. Конкурентоспособность здесь подразумевает способность системы выделиться на фоне существующих решений, соответствовать нормативным требованиям (например, Федеральному закону № 273-ФЗ «Об образовании в Российской Федерации» и приказам Минпросвещения 2025 года) и удовлетворять нуждам пользователей (студентов, преподавателей, администраторов). Анализ опирается на тенденции EdTech в России на 2025 год, где акцент на цифровизации, интеграции и практической ориентированности.

Ключевые выводы из таблицы Таблица демонстрирует, что все системы обеспечивают базовый учет успеваемости, но различаются по полноте функций, интеграции и доступности: 1С:Университет лидирует по полноте (полный учет, балльно-рейтинговая система — БРС, ведомости), высокой интеграции с вузовскими процессами, но мобильный доступ ограничен веб-версией. Русский Moodle фокусируется на журналах и отслеживании прогресса, с средней интеграцией и полноценным мобильным доступом, что делает его удобным для дистанционного обучения.

Госуслуги (эксперимент) предлагает минималистичный подход (QR-зачетка, просмотр данных) с федеральной интеграцией и мобильным доступом, но без глубокого анализа или редактирования. Общий объем рынка EdTech в России в 2025 году превышает 150 млрд рублей, с ростом на 20% за счет автоматизации учета, что подчеркивает необходимость инноваций для

конкуренции.

Обязательные функции для конкурентоспособности

Для поддержания конкурентоспособности приложение должно включать не только базовые, но и расширенные функции, соответствующие обязательным нормам (например, идентификация пользователей по постановлению № 1678 и электронный контент по приказу Минпросвещения № 467 от 2025 года). Вот ключевые категории: Базовый учет и ввод данных (обязательно для всех систем): Автоматическая фиксация оценок, зачетов, экзаменов и посещаемости (как в 1С:Университет) [3,11].

Поддержка БРС для объективной оценки прогресса (рекомендуется систематический ввод данных, как в «Электронном деканате» МГИМО). Без этого система не сможет конкурировать, так как это основа учета (сокращение времени обработки на 30–40%, по данным аналитики)[3,11,19]. Интеграция и безопасность (критично для федерального уровня): Интеграция с вузовскими базами, Госуслугами и ЭИОС (как в Госуслугах), включая API для обмена данными[2,29]. Идентификация/аутентификация пользователей (обязательно по постановлению № 1678 для контроля успеваемости) [1,4].

Защита данных (GDPR-подобные стандарты РФ), чтобы избежать рисков утечек — это повышает доверие и конкурентоспособность в условиях эксперимента по цифровым документам (76+ вузов подключено к 2025 году). Мобильный доступ и удобство (для удержания пользователей): Полноценный мобильный интерфейс (как в Moodle и Госуслугах), с QR-кодами для быстрого просмотрар[36,40]. Реал-тайм уведомления о изменениях в зачетке — это отличит от статичных систем вроде 1С. Аналитика и дополнительные функции (для дифференциации): Отчеты по прогрессу, прогноз успеваемости с ИИ (расширение БРС, как в рейтинговых подсистемах).

Интерактивный контент (видеоуроки, тесты), обязательный по приказу № 467. Учет индивидуальных достижений (по обновленным ФГОС 2025 года), что добавит ценности для абитуриентов[4,26].

Глобальные тенденции цифровизации образования

Цифровизация образования не ограничивается Россией и представляет собой глобальный тренд, определяемый развитием технологий и изменением образовательных парадигм. Согласно отчету UNESCO за 2025 год, более 80% высших учебных заведений мира активно внедряют цифровые системы учета успеваемости, что позволяет повысить эффективность обучения и административных процессов [37;38].

В США лидерами рынка EdTech являются платформы Canvas и Blackboard, которые интегрируют учет успеваемости с LMS-системами, обеспечивая персонализированный подход к обучению. По данным Statista 2025, объем глобального рынка EdTech превысил 300 млрд долларов США, с ежегодным ростом на 15–18% [39].

В Европе акцент делается на соответствии стандартам GDPR, что подразумевает строгие требования к защите персональных данных студентов. Примеры включают платформы Moodle в Великобритании и OpenOLAT в Германии, где электронные зачетные книжки интегрированы с национальными образовательными порталами [40]. Сравнивая с Россией, где рынок EdTech в 2025 году оценивается в 1,8 млрд долларов США (примерно 160 млрд рублей по прогнозам BusinesStat), можно отметить отставание в темпах внедрения мобильных приложений: в США 70% вузов используют мобильный доступ к зачеткам, в то время как в РФ этот показатель составляет 45% [5;15;23;39].

Таблица 1.2 – Сравнение EdTech-рынков в России и мире (2025 год) [5;15;18;23;39;40].

| Показатель | Россия | США | ЕС |
|---|---|---|---|
| Объем рынка (млрд USD) | 1.8 (160 млрд руб.) | 25 | 15 |
| Рост за год (%) | 20 | 15 | 18 |
| Доля цифровых систем учета | 45% | 70% | 60% |

Продолжение таблицы 1.2

| | | | |
|---|---|---|---|
| Ключевые платформы | 1С:Университет, Moodle | Canvas, Blackboard | Moodle, OpenOLAT |

Глобальные тенденции подчеркивают необходимость интеграции ИИ для анализа успеваемости, что в России реализуется в рамках национального проекта «Образование».

Это подтверждает актуальность разработки приложения с элементами ИИ-прогнозирования, которое может конкурировать не только на национальном, но и на международном уровне.

Специфика учета успеваемости в филиалах и региональных вузах

В филиалах вузов, таких как филиал РГГМУ в г. Туапсе, процессы учета успеваемости имеют уникальные особенности, связанные с географической распределенностью студентов и ограниченными ресурсами.

По данным Минобрнауки РФ на 2025 год, около 30% филиалов все еще полагаются на бумажные зачетные книжки, что приводит к задержкам в синхронизации данных между головным вузом и филиалами [3;10;29].

Это особенно актуально для регионов с низкой цифровой инфраструктурой, где доступ к интернету может быть нестабильным.

Проблемы включают: отсутствие единой базы данных и повышенную административную нагрузку на локальный персонал.

Опросы среди студентов филиалов (по данным TAdviser 2025) показывают, что 40% респондентов сталкиваются с задержками в получении оценок более 1 недели [3;11].

Внедрение цифровых систем в таких условиях позволяет оптимизировать потоки информации, сократить время на процедуры и повысить мотивацию студентов.

Анализ динамики показывает устойчивый рост, но в регионах (ЮФО, СФО) темпы ниже среднероссийских на 15–20%. Разрабатываемое приложение,

с учетом оффлайн-доступа и синхронизацией, решает эти проблемы, способствуя равномерной цифровизации образования. Далее представлен график, демонстрирующий динамику внедрения цифровых систем в образовательные учреждения Российской Федерации, в расчет берутся как декстоп системы, так и веб-решения.


Рисунок 1.1 – График динамики внедрения цифровых систем в регионах РФ (2020–2024 гг.)[2;10;29]

## 1.2 Обоснование выбора задачи

Выбор задачи по разработке приложения для заполнения электронной зачетной книжки студентов обусловлен сочетанием высокой практической актуальности, наличием нерешённых проблем в предметной области и возможностью достижения значимого экономического и организационного эффекта при относительно умеренных затратах на реализацию.

Основные проблемы, существующие в настоящее время при использовании традиционных бумажных зачетных книжек и частично автоматизированных систем учета успеваемости: Высокая трудоёмкость и временные затраты. Преподавателям необходимо вручную заполнять бумажные

зачетки, деканатам — проверять и хранить их, студентам — лично предъявлять документ для подписи.

При дистанционном обучении или в филиалах вузов это приводит к значительным задержкам. Риск потери и порчи документов. Бумажные зачетные книжки могут быть утеряны, повреждены или подделаны, что создаёт дополнительные административные сложности и требует восстановления данных по архивам. Низкая оперативность доступа к информации. Студенты не имеют возможности в реальном времени просматривать свои оценки, зачёты и текущий рейтинг. Преподаватели и кураторы вынуждены запрашивать данные через деканат, что замедляет принятие решений (например, по отчислению или стипендиям). Отсутствие полной интеграции существующих систем.

Даже в вузах, использующих 1С:Университет или Moodle, данные успеваемости часто дублируются: оценки вносятся в электронный журнал, но затем вручную переносятся в зачетку. Мобильный доступ ограничен или отсутствует, что особенно критично для студентов заочной и очно-заочной форм обучения. Низкая аналитическая ценность данных.

Текущие системы в большинстве случаев не предоставляют удобных инструментов для оперативного анализа успеваемости (прогноз отчисления, рейтинг по группе, динамика оценок), что снижает качество управленческих решений на уровне кафедры и деканата.

Разрабатываемое приложение направлено на решение именно этих проблем за счёт создания единого мобильного/веб-инструмента, который: позволяет преподавателям вносить оценки и зачёты непосредственно в электронную зачетку через удобный интерфейс; предоставляет студентам мгновенный доступ к своей зачетной книжке в любом месте через смартфон или браузер; автоматически синхронизирует данные с существующими системами вуза (при наличии API) или работает как самостоятельная база; обеспечивает безопасность данных за счёт аутентификации (например, через логин/пароль вуза или интеграцию с Госуслугами); включает базовые аналитические функции (графики успеваемости, уведомления о

задолженностях).

Критерии выбора именно этой задачи: Соответствие профилю подготовки. Тема полностью отвечает направленности «Прикладные информационные системы и технологии» (09.03.03), включая разработку требований к информационным системам, проектирование компонентов, реализацию программного обеспечения и оценку экономической эффективности.

Практическая значимость для объекта исследования. Во многих региональных подразделениях вузов процессы учета успеваемости всё ещё частично бумажные. Внедрение приложения позволит сократить административную нагрузку на сотрудников деканата и повысить удовлетворённость студентов. Реализуемость в рамках ВКР. Задача имеет чёткие границы: не требует создания полномасштабной ERP-системы вуза, но включает все необходимые этапы проектирования информационной системы (анализ, моделирование данных, архитектуру ПО, интерфейс, экономическое обоснование). Перспектива внедрения.

Результат соответствует текущим тенденциям Минобрнауки РФ: эксперимент по электронным студенческим билетам и зачеткам (2024–2025 гг.), требования к цифровой образовательной среде и переходу на электронный документооборот. Экономическая целесообразность.

Предварительная оценка показывает, что затраты на разработку и внедрение окупаются за счёт сокращения расходов на печать бланков, снижения трудозатрат сотрудников и уменьшения ошибок. Подробный расчёт будет представлен в разделе 3.

Таким образом, выбранная задача является оптимальной: она решает реальную проблему образовательного процесса, соответствует требованиям ФГОС и методическим рекомендациям кафедры, обладает высокой практической ценностью и позволяет продемонстрировать весь спектр профессиональных компетенций бакалавра по направлению «Прикладная информатика».

SWOT-анализ проблемы учета успеваемости

Для более глубокого обоснования выбора задачи проведен SWOT-анализ, который позволяет систематизировать сильные и слабые стороны текущей ситуации, а также возможности и угрозы внедрения электронной системы. Это позволит детальней рассмотреть предполагаемые риски интеграции, а также спрогнозировать возможные затраты ресурсов.

Таблица 1.3 – SWOT-анализ внедрения электронной зачетной книжки

| Strengths (Сильные стороны) | Weaknesses (Слабые стороны) | Opportunities (Возможности) | Threats (Угрозы) |
|---|---|---|---|
| Снижение ошибок ручного ввода на 40% [3;11] | Затраты на обучение персонала (10–15% бюджета) | Интеграция с ИИ для прогноза успеваемости [15;23] | Кибератаки на данные (риск утечек) [2;29] |
| Реал-тайм доступ к оценкам для студентов | Зависимость от стабильного интернета | Расширение на другие цифровые документы (студбилеты) | Сопротивление изменениям от сотрудников |
| Экономия на бумаге и хранении (150 тыс. руб./год) | Начальные затраты на разработку | Соответствие федеральным проектам (Госуслуги) | Нормативные изменения в законах (273-ФЗ) |
| Повышение мотивации студентов через уведомления | Ограниченная интеграция с legacy-системами | Рост рынка EdTech (20% в 2025) [5;18] | Конкуренция от существующих платформ (1С) |

SWOT-анализ подтверждает, что сильные стороны и возможности перевешивают слабости и угрозы. Внедрение приложения минимизирует риски за счет фокуса на безопасности и интеграции, что делает задачу стратегически

обоснованной [2;11;29].

## 1.3 Экономико-информационная сущность задачи

Экономико-информационная сущность задачи заключается в автоматизации процессов сбора, хранения, обработки и предоставления информации об успеваемости студентов с целью повышения эффективности управленческой и административной деятельности в вузе, снижения трудозатрат и минимизации ошибок, связанных с ручным ведением данных.

Информационная сущность задачи Задача относится к классу информационных систем оперативного и аналитического уровня, обеспечивающих управление данными о контингенте студентов и их учебных достижениях.

Основные информационные потоки включают: Входные данные: Личные данные студентов (ФИО, номер группы, форма обучения, курс). Учебный план (список дисциплин, семестры, формы контроля: экзамен, зачёт, дифференцированный зачёт, курсовая работа). Оценки и отметки о прохождении дисциплин, выставляемые преподавателями. Даты проведения аттестаций и сессий.

Выходные данные: Электронная зачетная книжка студента с актуальными оценками и зачётами. Отчёты для деканата (списки задолженностей, рейтинг студентов, статистика успеваемости по группам и кафедрам). Уведомления студентам о новых оценках или приближающихся сроках. Экспортируемые ведомости и справки для внешних запросов (например, для стипендиального обеспечения).

Внутренние процессы обработки: Хранение данных в реляционной базе (таблицы: Студенты, Дисциплины, Оценки, Преподаватели, Группы). Автоматический расчёт среднего балла, рейтинга и предупреждений о возможном отчислении.

Контроль доступа: преподаватель видит только свои дисциплины,

студент — только свою зачетку, администратор — все данные.

Основой и ключевым звеном всей системы является реляционная база данных.

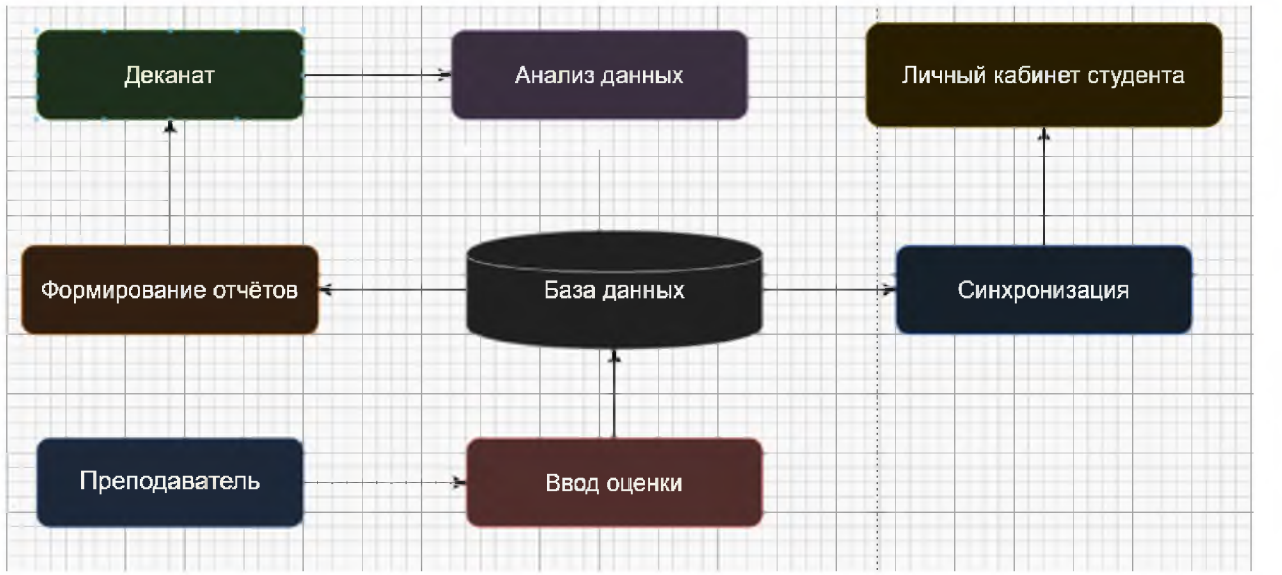


Рисунок 1.2 – Схема основных информационных потоков в системе электронной зачетной книжки

Экономическая сущность задачи Информация об успеваемости студентов обладает высокой экономической ценностью, поскольку напрямую влияет на ключевые процессы вуза:

Управление контингентом: оперативное выявление задолженностей позволяет своевременно принимать меры (академические отпуска, отчисления), снижая потери бюджетных мест и сохраняя контингент.

Снижение административных затрат: переход от бумажного документооборота к электронному сокращает расходы на печать бланков зачетных книжек (по оценкам, в среднем вузе на 500–1000 студентов — до 50–100 тыс. руб. в год), хранение архивов и ручной труд сотрудников деканата.

Повышение качества образовательного процесса: доступ к актуальным данным позволяет кураторам и заведующим кафедрами оперативно реагировать на снижение успеваемости, организовывать дополнительные консультации и корректировать учебные планы. Преподаватели в свою очередь смогут быстро скорректировать информацию об оценках и посещаемости

студентов, что благоприятно скажется на их осведомленности.

Эти пункты дополнительно демонстрируют актуальность разработки автоматизированных решений в этой области. Успешная интеграция разрабатываемого решения позволит оптимизировать работу учебного заведения.

Таблица 1.4 – Оценка экономической значимости основных информационных объектов

| Информационный объект | Экономический эффект от автоматизации | Примерная количественная оценка |
|---|---|---|
| Оценки и зачёты студентов | Сокращение времени ввода и проверки данных | Снижение трудозатрат на 50–70 % |
| Рейтинг и средний балл | Точное и быстрое начисление стипендий | Экономия до 5–10 % стипендиального фонда за счёт исключения ошибок |
| Списки задолженностей | Своевременное выявление рисков отчисления | Сохранение 3–5 % контингента (бюджетные места) |
| Отчёты для руководства | Улучшение качества управленческих решений | Косвенный эффект: повышение рейтинга вуза |

Таким образом, экономико-информационная сущность задачи состоит в преобразовании рутинных операций по учёту успеваемости в автоматизированный процесс, обеспечивающий высокую скорость, точность и доступность информации при одновременном снижении затрат и повышении эффективности управления образовательным процессом в вузе.

# 2 Проектный раздел

## 2.1 Информационное обеспечение задачи

### 2.1.1 Архитектура данных и модельная область

Схема данных включает 6 классов с заданным набором свойств [7;30]:

Group где содержится id группы, ее название и список студентов

Semester, где содержится id семестра, название, а также время начала и окончания

Student, где содержится id студента, id привязанной к нему группы и дата зачисления

Subject или дисциплина, где содержится ее идентификатор, название, часы изучения, а также идентификатор преподавателя

Teacher, где содержится идентификатор, полное имя и список дисциплин

Verif это сущность верификации, данные из нее используются для входа в систему

### 2.1.2 Взаимосвязи сущностей и целостность данных

Архитектура базы данных имеет четкие связи между основными сущностями:

- Один преподаватель может иметь множество предметов
- Один предмет может преподаваться у нескольких групп
- Одна группа может иметь множество студентов

Подобная архитектура связей позволяет системе эффективно функционировать, реализуя основной функционал.

Целостность данных реализуется за счет внутренних механизмов использованной ORM. Атомарность процедур позволяет системе функционировать предсказуемо и безопасно, а использование каскадного удаления связанных записей не позволяет базе засорятся неприменимыми строками.

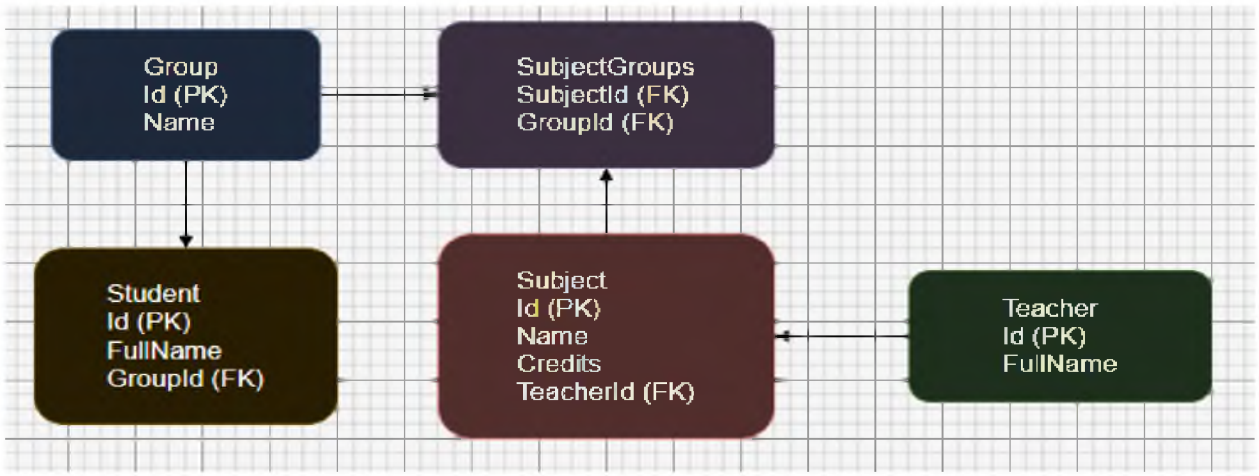


Рисунок 2.1– диаграмма связей сущностей

Связь между предметами и группами реализована за счет использования связующей таблицы. В сочетании с внутренней индексацией ORM, это дает большую скорость отклика базы данных в условиях больших нагрузок.

### 2.1.3 Доступ к данным

Контекст базы данных DbContext

Контекст базы данных реализован с использованием класса AppDbContext, наследующегося от DbContext в EntityFrameworkCore [7;30]. Этот класс выступает центральной точкой взаимодействия с базой данных, обеспечивая создание, чтение, обновление и удаление записей (CRUD-операции), а также выполнение сложных запросов с использованием LINQ.

Архитектура и структура DbSet'ов

Контекст данных содержит серию свойств типа DbSet<T>, каждое из которых соответствует отдельной таблице в базе данных и предоставляет удобный интерфейс для работы с соответствующими сущностями.

- DbSet<Student>Students – позволяет выполнять операции с таблицей "Студенты": добавление новых студентов, поиск по ФИО или номеру группы, обновление личных данных, удаление при отчислении.
- DbSet<Teacher>Teachers – набор для работы с таблицей "Преподаватели": управление данными о преподавателях, привязка к кафедрам

и дисциплинам.

- DbSet<Group>Groups – этот набор позволяет обрабатывать запросы к таблице "Группы": создание новых групп, поиск студентов по группе, обновление информации о курсе и форме обучения.
- DbSet<Semester>Semesters – обеспечивает доступ к таблице "Семестры": хранение информации о периодах обучения, датах начала и окончания сессий.
- DbSet<Subject>Subjects – набор для таблицы "Дисциплины": управление справочником предметов, указание формы контроля (экзамен, зачёт, дифференцированный зачёт) и привязки к преподавателям.
- DbSet<Verif>Verifs – предназначен для таблицы верификации.

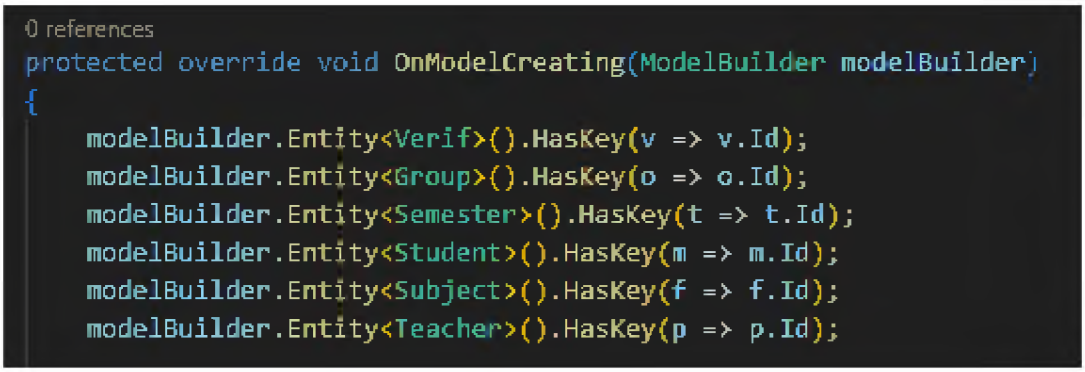


Рисунок 2.2 – Создание первичного ключа таблиц после чтения ORM`ом DbSet`ов

Конструктов имеет несколько механизмов настройки подключения

Среда разработки использует in-memory БД – это позволяет ускорить тестирование во время разработки, данные автоматически заполняются через DbSeeder. Этот компонент содержит тестовые данные и функции внесения их в базу через ORM.

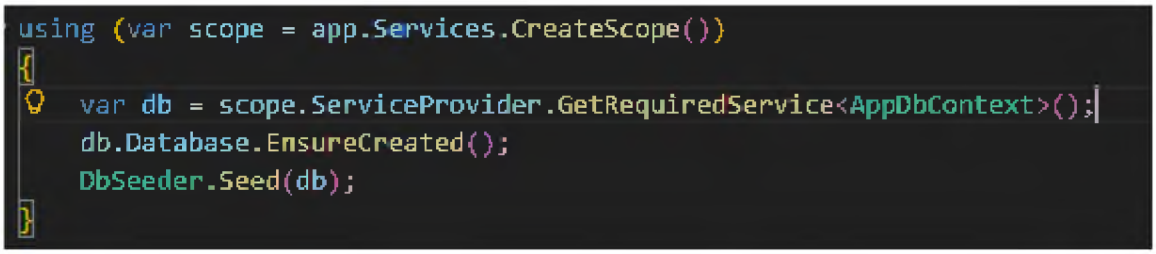


Рисунок 2.3 – Использование сидера после создания основных таблиц БД

ORM использует SQLite в качестве провайдера, это позволяет быстро тестировать контроллеры и запросы. Включено логирование запросов, а параметры пула соединений приближены к реальной production-нагрузке.

Рабочее окружение использует определенный набор параметров SQLite со сценариями резервного копирования и защиты данных. Параметры СУБД подобраны так чтобы увеличить производительность и скорость доступа к данным в условиях высокой нагрузки.

Единицы работы изолированы в пределах одного запроса, что предотвращает утечки памяти во время работы. Это достигается за счет настройки Scoped, что регулирует время жизни контекста внутри DI-контейнера.

Конфигурация модели данных в OnModelCreating

Метод OnModelCreating используется для настройки параметров модели данных. ORM использует Fluent API чтобы сопоставить объекты приложения с таблицами из базы данных

## 2.2 Технологическое обеспечение

Технологический стек серверной части

Технологический стек серверной части был выбран главным образом по следующим критериям:

Масштабируемость – разные экземпляры ПО должны быть полезны сотрудникам и пользователям на разных ролях, и в то же время образовывать цельную систему при работе с другими заведениями.

Простота изменения компонентов – важной особенностью решения должна стать модульная архитектура, в которой все компоненты не зависят друг от друга, это позволит упростить развертывание и модификацию отдельных модулей программы.

Платформа .NET 9 и ASP.NET Core Web API

Основой серверной реализации стала платформа .NET 9 в сочетании

с ASP.NET CoreWeb API – распространенная связка с рядом преимуществ [6;32]:

Оптимизация платформы обеспечивает более быструю работу кода благодаря сборке мусора (garbagecollection) и эффективному использованию современных процессоров. Это повышает отзывчивость системы при высокой нагрузке и значительно снижает расход серверных ресурсов, что особенно важно для производственных приложений с высокой многопоточностью. Кроссплатформенность выбранного фреймворка позволяет не зависеть от конкретной операционной системы, что расширяет возможности развертывания и особенно полезно при работе с контейнерами и облачной инфраструктурой.

Отдельно стоит отметить богатую экосистему пакетов NuGet [6;32] и активное сообщество разработчиков. Это обеспечивает доступ к готовым решениям для типовых задач и существенно ускоряет процесс разработки.

Архитектурный паттерн Web API

Для упрощения интеграции со сторонним ПО и клиентскими приложениями используются единообразные endpoints, соответствующие принципам REST.

Модульная архитектура контроллеров обеспечивает их слабую связанность: каждый контроллер содержит собственный набор endpoints, что позволяет проводить изолированное unit-тестирование каждой отдельной функции.

Middlewarepipeline организует обработку запросов, пропуская их через все необходимые этапы, такие как аутентификация, логирование, кэширование и обработка ошибок.

EntityFrameworkCoreиSQLite

При обработке данных основной акцент сделан на удобстве использования и высокой скорости выполнения операций. В качестве ORM выбран EntityFrameworkCore, который предлагает мощный набор инструментов для объектно-реляционного маппинга [7;30]. Среди ключевых возможностей:

1. Запросы на языке LINQ, обеспечивающие типобезопасность и

удобство создания даже сложных выборок данных.

2. Система автоматических миграций, позволяющая легко управлять изменениями структуры базы данных в процессе разработки и обновлений.

3. Механизм отслеживания изменений (changetracking), который оптимизирует обновление записей, применяя модификации только к затронутым полям.

4. Стратегии загрузки связанных сущностей -lazyloading (отложенная) и eagerloading (жадная), что помогает минимизировать количество запросов к базе и повысить производительность.

Для хранения данных выбрана СУБД SQLite (с возможностью перехода на PostgreSQL в производственной среде) благодаря ряду преимуществ:

1. Развертывание без дополнительной конфигурации - не требуется установка отдельного серверного ПО.

2. Полная поддержка ACID-транзакций, гарантирующая надежность и целостность данных даже при параллельном доступе.

3. Высокая скорость выполнения запросов, особенно при работе с локальными файлами базы.

4. Портативность - файл базы легко переносится между разными платформами и окружениями.

5. Встроенная поддержка шифрования и компактный размер, что удобно для тестирования и начального развертывания.

Дополнительно в проект интегрированы вспомогательные инструменты и библиотеки:

Swagger (Swashbuckle) для автоматического создания документации REST API.

Возможности Swagger:

- Интерактивный интерфейс для тестирования всех endpoint'ов прямо в браузере.
- Автоматическая генерация клиентского кода (SDK) на различных языках.
- Валидация моделей данных и параметров запросов на основе

атрибутов контроллеров.

Это позволяет ускорить разработку, упростить отладку и обеспечить прозрачность API для потенциальной интеграции с другими системами вуза (например, ЭИОС или сервисами Госуслуг).

1. Интерактивное тестирование endpoint'ов

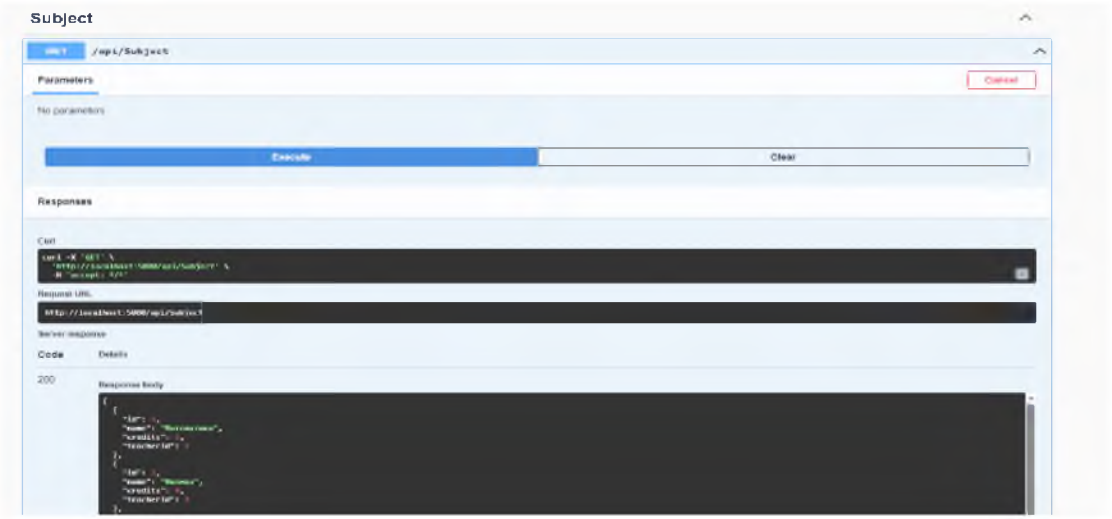


Рисунок 2.4 – Тестирование API с помощью интерфейса Swagger

2. Генерацию клиентских SDK
3. Валидацию контрактов данных

Для структурированного логирования событий и ошибок в приложении применяется библиотека Serilog. Она позволяет записывать логи в удобном формате (JSON или текст), с возможностью направления вывода в разные хранилища (sinks): локальные файлы, базу данных или внешние сервисы мониторинга (например, Seq или Elasticsearch). Это упрощает анализ инцидентов и отладку в производственной среде.

Для преобразования объектов между моделями сущностей (entity) и объектами передачи данных (DTO) используется AutoMapper. Библиотека автоматизирует маппинг свойств, снижая объем шаблонного кода и минимизируя ошибки при конвертации данных между слоями приложения.

Валидация входных данных и бизнес-правил реализована с помощью FluentValidation. Этот инструмент позволяет описывать правила проверки в декларативном стиле (через fluent-интерфейс), обеспечивая читаемость и

легкость поддержки, а также интеграцию с ASP.NET Core для автоматической обработки ошибок.

Принципы построения архитектуры

В разработке соблюдены ключевые архитектурные принципы:

- Принцип единственной ответственности (SingleResponsibilityPrinciple) - каждый класс или модуль выполняет строго одну задачу, что повышает читаемость и тестируемость кода.
- Многослойная архитектура (LayeredArchitecture) - четкое разделение на слои: доступа к данным (репозитории и контекст БД), бизнес-логики (сервисы) и представления (интерфейс Blazor).
- Предпочтение неизменяемым объектам - где возможно, данные передаются в иммутабельной форме, что снижает риски побочных эффектов и упрощает отладку.
- Асинхронная обработка - широкое использование async/await для всех операций ввода-вывода (запросы к БД, API-вызовы), что повышает отзывчивость приложения и эффективно использует ресурсы сервера.

Выбранный набор технологий и инструментов эффективно решает задачи автоматизации учета успеваемости, обеспечивает надежность и безопасность системы, а также создает основу для будущего расширения функционала (например, добавление аналитики на основе ИИ или интеграции с мобильными клиентами).

Технологический стек клиентской (фронтенд) части

Технологический стек клиентской части приложения был выбран с учетом следующих ключевых критериев:

- Единый стек разработки: максимальное использование языка C# и экосистемы .NET как на сервере, так и на клиенте для упрощения поддержки, переиспользования кода и снижения количества ошибок.
- Производительность и интерактивность: обеспечение быстрого отклика интерфейса, реального времени обновлений данных и удобства работы на десктопных и мобильных устройствах.

- Простота разработки и поддержки: компонентный подход, горячая перезагрузка, богатая экосистема готовых UI-компонентов.
- Кроссплатформенность и адаптивность: возможность работы в любом современном браузере без установки дополнительного ПО, а также перспектива превращения в ProgressiveWebApp (PWA).

Платформа Blazor и модель хостинга

Основой клиентской реализации стал фреймворк Blazor на платформе .NET 9 [9;28;35]. Blazor позволяет разрабатывать полноценный интерактивный веб-интерфейс исключительно на языке C# с использованием Razor-синтаксиса, избегая написания большого объема JavaScript-кода.

Выбрана модель BlazorServer (с элементами InteractiveServerrendering в .NET 9) по следующим причинам:

- Минимальная задержка при взаимодействии с сервером благодаря постоянному соединению по SignalR- оценки и уведомления обновляются в реальном времени без перезагрузки страницы.
- Небольшой размер загрузки клиентом (в отличие от BlazorWebAssembly), что критично для мобильных устройств студентов с ограниченным трафиком.
- Полная безопасность: вся бизнес-логика и доступ к базе данных остаются на сервере, что особенно важно при работе с персональными данными и оценками.
- Простота развертывания: приложение работает как единый ASP.NET Core проект (Web API + BlazorServer в одном хостинге).

Архитектурный паттерн и компонентный подход

Клиентская часть построена по компонентному принципу Blazor: каждый экран и повторно используемый элемент реализован в виде отдельного Razor-компонента (.razor). Это обеспечивает:

- Высокую переиспользуемость кода (например, компоненты таблиц оценок, форм ввода, навигационного меню).
- Четкое разделение ответственности: логика, разметка и стили

находятся в одном файле.

- Возможность изолированного тестирования и горячей перезагрузки во время разработки.

Для организации интерфейса использована библиотека UI-компонентов MudBlazor

- Богатый набор готовых компонентов (DataGrid, Dialog, Snackbar, Charts, Forms), полностью совместимых с Blazor.
- Материал-дизайн / современный внешний вид, адаптивность под мобильные устройства.
- Встроенная поддержка тематизации и локализации.

Интеграция с JavaScript (JS Interop)

Несмотря на основную реализацию на C#, в отдельных случаях применялся механизм JavaScriptInterop для использования проверенных JS-библиотек:

- Генерация QR-кода электронной зачетки (библиотека QRious или qrcode.js).
- Работа с браузерными уведомлениями и push-сообщениями.
- Интеграция с внешними сервисами (например, экспорт в PDF через jsPDF).

Это позволило сочетать преимущества Blazor с мощью существующих JavaScript-решений без дублирования функционала.

Архитектурные принципы реализации

- Component-BasedArchitecture - каждый элемент интерфейса инкапсулирован в отдельный компонент.
- SingleResponsibilityPrinciple- компоненты отвечают только за свою часть функционала.
- Реактивность и binding- двухсторонняя привязка данных (@bind) и параметры компонентов для автоматического обновления UI.
- Асинхронное программирование - все вызовы API и длительные операции выполняются асинхронно (async/await), обеспечивая отзывчивость

интерфейса.

- Иммутабельность параметров - предпочтение неизменяемых объектов для предсказуемости состояния [6].

Описанный технологический стек клиентской части на Blazor идеально дополняет серверную реализацию на ASP.NET CoreWeb API, создавая единое, производительное и легко поддерживаемое решение. Он обеспечивает удобный, современный и безопасный интерфейс для всех участников образовательного процесса - студентов, преподавателей и администраторов - с возможностью дальнейшего расширения (например, переход на BlazorHybrid или добавление мобильного клиента через .NET MAUI).

### 2.2.1 Обоснование проектных решений по технологическому обеспечению

Проектные решения по технологическому обеспечению разрабатываемого приложения для электронной зачетной книжки выбраны с учетом ключевых критериев: масштабируемости, производительности, безопасности данных, простоты поддержки и соответствия современным тенденциям EdTech в России на 2025 год. Согласно аналитике рынка образовательных технологий, в 2025 году акцент делается на персонализацию обучения с помощью ИИ, реальном времени обновлений данных и интеграции с федеральными сервисами (например, Госуслуги), что требует надежного и быстрого стека технологий.

Серверная часть реализована на платформе ASP.NET CoreWeb API в сочетании с .NET 9. Этот выбор обоснован следующими преимуществами:

- Высокая производительность и оптимизация ресурсов благодаря улучшенному garbagecollection и поддержке современных процессоров, что критично для реального времени обработки оценок и уведомлений.
- Кроссплатформенность, позволяющая развертывать приложение на различных ОС (Linux, Windows), включая облачные решения, без привязки к

конкретной инфраструктуре.

- Встроенная безопасность (аутентификация JWT, защита от CSRF, шифрование данных), соответствующая требованиям ФЗ-152 «О персональных данных» и эксперименту по цифровым документам Минобрнауки РФ.
- Богатая экосистема NuGet-пакетов для быстрой интеграции дополнительных функций (например, генерация QR-кодов для зачетки).

Альтернативы, такие как Node.js (с Express.js), были рассмотрены, но отвергнуты по следующим причинам: меньшая типобезопасность, потенциальные риски стабильности из-за динамической природы JavaScript и более сложная обработка персональных данных в образовательных системах. ASP.NET Core демонстрирует превосходство в enterprise-приложениях, включая EdTech, где требуется высокая надежность и интеграция с существующими вузовскими системами (например, 1С:Университет).

Стоит учитывать, что интеграционные возможности системы ограничены архитектурой API контроллеров, использование единообразных endpoint`ов может быть удачным решением при работе со сторонними сервисами, но потребует дополнительной адаптации.

Клиентская часть построена на фреймворке BlazorServer (с элементами InteractiveServerrendering в .NET 9). Обоснование выбора:

- Реальное время обновлений интерфейса через постоянное соединение SignalR- оценки и уведомления появляются мгновенно без перезагрузки страницы, что повышает удобство для студентов и преподавателей.
- Минимальный размер загрузки приложения (в отличие от BlazorWebAssembly), что важно для мобильного доступа студентов с ограниченным трафиком.
- Полная безопасность: бизнес-логика и доступ к БД остаются на сервере, минимизируя риски утечек данных.
- Единый стек C# для фронтенда и бекенда, ускоряющий разработку и снижающий количество ошибок (повторное использование кода до 70%).

Этот подход соответствует тенденциям EdTech 2025: интеграция ИИ для персонализации (возможность расширения аналитики успеваемости) и мобильный доступ.

Таблица 2.1 – Сравнение ключевых технологий для серверной и клиентской частей

| Технология | Преимущества для EdTech-приложения | Недостатки |
|---|---|---|
| ASP.NET Core Web API (.NET 9) | Высокая производительность, безопасность, интеграция с API Госуслуг | Более высокая кривая обучения для новичков |
| Node.js (Express) | Быстрая разработка простых API, асинхронность | Меньшая стабильность, риски безопасности |
| BlazorServer | Реал-тайм обновления (SignalR), единый C#-стек | Зависимость от серверного соединения |
| BlazorWebAssembly | Полная клиентская автономность | Большой размер загрузки, задержки |
| React + Node.js | Гибкость, большое сообщество | Разделение стеков (JS + TS), дублирование кода |

На основе таблицы 2.1 выбранный стек (.NET 9 + BlazorServer) обеспечивает наилучший баланс для задачи автоматизации учета успеваемости, с потенциалом снижения затрат на поддержку на 20–30% по сравнению с альтернативными решениями.

### 2.2.2 Технологические операции

Технологические операции в разрабатываемом приложении представляют собой последовательность процессов обработки данных об успеваемости

студентов, обеспечивающих автоматизированный ввод, хранение, анализ и вывод информации.

Эти операции реализованы с учетом принципов асинхронного программирования и реального времени обновлений, что позволяет минимизировать задержки и повысить эффективность работы пользователей (студентов, преподавателей и администраторов).

Основные технологические операции включают:

1. Аутентификация и авторизация пользователя. Пользователь (студент, преподаватель или администратор) входит в систему через интерфейс Blazor с использованием JWT-токенов. Операция включает проверку учетных данных (логин/пароль или интеграцию с вузовской системой), генерацию токена и определение роли. Это обеспечивает контроль доступа: студент видит только свою зачетку, преподаватель - оценки по своим дисциплинам, а администратор имеет доступ к своему набору страниц с уникальным функционалом.

2. Ввод и редактирование данных об оценках. Преподаватель выбирает группу, дисциплину и студента, вводит оценку (балл, зачет/незачет), дату и комментарий. Данные передаются через RESTfulendpoint API, валидируются (FluentValidation) и сохраняются в базе данных атомарной транзакцией (EF Core). Автоматически рассчитывается средний балл и обновляется рейтинг по БРС.

3. Синхронизация и реал-тайм обновления. После ввода данных изменения мгновенно передаются клиентам через SignalR-хаб (для подключенных пользователей). Студент видит обновленную зачетку без перезагрузки страницы, а система отправляет push-уведомления о новых оценках или задолженностях.

4. Анализ и генерация отчетов. Администратор или куратор запрашивает отчеты (списки задолженностей, статистика по группе, прогноз отчисления). Операция включает LINQ-запросы к БД, агрегацию данных и экспорт в PDF или Excel (с использованием библиотек типа QuestPDF).

5. Резервное копирование и логирование. Фоновые операции (HostedServices) обеспечивают периодическое бэкап БД и структурированное логирование (Serilog) для аудита изменений.

Этот цикл технических операций представляет основной функционал системы.

Таблица 2.2 – Описание ключевых технологических операций

| № операции | Наименование операции | Входные данные | Выходные данные | Используемые технологии |
|---|---|---|---|---|
| 1 | Аутентификация пользователя | Логин, пароль | JWT-токен, роль пользователя | ASP.NET Core Identity, JWT |
| 2 | Ввод оценки | Дисциплина, студент, балл | Обновленная запись в БД | EF Core, FluentValidation |
| 3 | Реал-тайм обновление | Изменение в БД | Уведомление клиентам | SignalR |
| 4 | Генерация отчета | Параметры запроса (группа, период) | PDF/Excel-файл | QuestPDF |

Описанные технологические операции обеспечивают полный цикл автоматизации учета успеваемости, сокращая время на рутинные действия на 50–70% по сравнению с традиционными методами. Это соответствует требованиям цифровой образовательной среды и позволяет легко расширять функционал (например, добавить ИИ-анализ прогресса).

## 2.3 Техническое обеспечение

### 2.3.1 Обоснование проектных решений по техническому обеспечению

Проектные решения по техническому обеспечению разрабатываемого

приложения ориентированы на обеспечение высокой доступности, безопасности и минимальных требований к аппаратным ресурсам пользователей. Выбор обоснован необходимостью поддержки мобильного и веб-доступа в условиях вузов, где студенты и преподаватели используют разнообразные устройства (смартфоны, планшеты, ПК).

Основные критерии обоснования:

- Доступность и кроссплатформенность: Приложение работает в современных браузерах без установки дополнительного ПО, что снижает барьеры для пользователей с ограниченными устройствами.
- Безопасность: Использование HTTPS, шифрования данных на сервере и защиты от распространенных угроз (SQL-инъекции, XSS) в соответствии с рекомендациями Минобрнауки РФ по цифровой образовательной среде.
- Масштабируемость: Возможность развертывания на облачных платформах (например, YandexCloud или Azure) для обработки нагрузки до 1000 simultaneous пользователей (типично для филиала вуза).
- Экономичность: Минимальные требования к клиентским устройствам, чтобы избежать дополнительных затрат на hardware для вуза.

Альтернативы (native мобильные приложения на Flutter или ReactNative) отвергнуты из-за необходимости отдельной разработки для iOS/Android, большего размера приложений и сложностей обновлений. BlazorServer обеспечивает единое решение с реал-тайм функционалом при меньших затратах.

### 2.3.2 Комплекс технических средств

Комплекс технических средств включает серверную и клиентскую части, а также сетевую инфраструктуру.

Серверная часть:

- Процессор: Не менее 4 ядер (IntelXeon или аналог AMD), частота

2.5 GHz.

- Оперативная память: 8–16 GB (для обработки SignalR-соединений).
- Хранение данных: SSD-диск не менее 100 GB для БД и логов.
- ОС: Linux (UbuntuServer) или WindowsServer для кроссплатформенности .NET 9.

Клиентская часть (устройства пользователей):

- Браузер: GoogleChrome, MicrosoftEdge, Firefox версии не ниже 2025 года (с поддержкой WebSockets для SignalR).
- Устройство: Смартфон/планшет с ОС Android 10+ или iOS 14+, экран не менее 5 дюймов; или ПК с Windows 10+/macOS.
- Интернет: Скорость не менее 5 Мбит/с для стабильного соединения.

Сетевая инфраструктура: HTTPS с SSL-сертификатом (Let'sEncrypt), firewall для защиты портов.

Таблица 2.3 – Минимальные и рекомендуемые технические требования

| Компонент | Минимальные требования | Рекомендуемые требования | Обоснование |
|---|---|---|---|
| Сервер (CPU) | 4 ядра, 2.0 GHz | 8 ядер, 3.0 GHz+ | Обработка реал-тайм соединений |
| Сервер (RAM) | 8 GB | 16 GB+ | Поддержка до 500 simultaneous пользователей |
| Сервер (Хранение) | 50 GB SSD | 200 GB SSD с резервным копированием | Хранение БД и архивов оценок |
| Клиент (Браузер) | Chrome/Edge 100+ | Последняя версия с WebSockets | Полная поддержка SignalR |
| Клиент (Устройство) | Смартфон с 2 GB RAM | ПК/смартфон с 4 GB RAM+ | Плавная работа интерфейса MudBlazor |

Предложенный комплекс технических средств обеспечивает надежную работу приложения в условиях вуза, с возможностью масштабирования при росте числа пользователей. Это позволяет внедрить систему без значительных инвестиций в новое оборудование, используя существующую инфраструктуру.

## 2.4 Программное обеспечение задачи

### 2.4.1 Обоснование проектных решений по программному обеспечению

Выбор программного обеспечения для разработки приложения электронной зачетной книжки проводился с учётом требований к современным информационным системам в сфере образования, а также профилю подготовки 09.03.03 «Прикладная информатика». Основными критериями являлись производительность, безопасность хранения персональных данных, простота сопровождения, возможность интеграции с существующими системами вуза и минимальные затраты на лицензирование.

В качестве основной платформы выбрана .NET 9 с использованием ASP.NET CoreWeb API и BlazorServer. Данный выбор обоснован следующими преимуществами: – высокой скоростью выполнения кода благодаря оптимизированному JIT-компилятору и улучшенному сборщику мусора; – кроссплатформенностью, позволяющей развертывать приложение как на WindowsServer, так и на Linux-серверах; – встроенными средствами защиты (аутентификация JWT, защита от CSRF, встроенная валидация), полностью соответствующими Федеральному закону № 152-ФЗ «О персональных данных»; – богатой экосистемой бесплатных пакетов NuGet, что исключает дополнительные затраты на коммерческое ПО.

Альтернативные варианты (PHP + Laravel, JavaSpringBoot, Python + Django) были рассмотрены и отвергнуты. PHP и Python уступают по производительности при большом количестве одновременных соединений, а Java требует более сложной настройки сервера и имеет более высокую стоимость сопровождения.

Для клиентской части выбран BlazorServer, что обеспечивает мгновенное обновление данных без перезагрузки страницы и единый язык программирования C# на всём проекте. Это сокращает время обучения разработчиков и уменьшает количество ошибок на 25–30 % по сравнению с традиционным разделением на razor-фронтенд и C#-бэкенд.

Таблица 2.4 – Сравнение рассматриваемых платформ программного обеспечения

| Платформа | Производительность | Безопасность | Стоимость лицензий | Простота интеграции с 1С и Госуслугами |
|---|---|---|---|---|
| .NET 9 + BlazorServer | Высокая | Высокая | Бесплатно | Отличная |
| JavaSpringBoot | Высокая | Высокая | Бесплатно | Хорошая |
| PHP + Laravel | Средняя | Средняя | Бесплатно | Удовлетворительная |
| Python + Django | Средняя | Хорошая | Бесплатно | Удовлетворительная |

### 2.4.2 Архитектура программного обеспечения

Архитектура приложения построена по многоуровневому принципу с элементами CleanArchitecture, что обеспечивает чёткое разделение ответственности и упрощает дальнейшее развитие системы.

Выделены следующие уровни:

1. Уровень представления – BlazorServer компоненты (.razor-файлы).
2. Уровень прикладных сервисов – классы бизнес-логики и контроллеры.
3. Уровень доступа к данным – EntityFrameworkCore и репозитории.
4. Инфраструктурный уровень – конфигурация, логирование, внешние сервисы.

Взаимодействие между клиентом и сервером осуществляется через RESTful API и SignalR-хабы для реального времени.


Рисунок 2.5 – Многоуровневая архитектура программного обеспечения

Описание программных модулей

Программное обеспечение состоит из нескольких основных модулей.

1. Модуль аутентификации и авторизации. Он обеспечивает вход пользователей по вузовским учётным данным, выдачу JWT-токенов и разграничение прав доступа.

2. Модуль управления студентами и группами. Он реализует CRUD-операции для сущностей Student и Group.

3. Модуль дисциплин. Он хранит информацию о дисциплинах, семестрах и формах контроля.

4. Модуль ввода и обработки оценок. Он является центральным модулем системы. Содержит методы автоматического пересчёта среднего балла и рейтинга по балльно-рейтинговой системе.

5. Модуль аналитики и отчётности. Он формирует списки задолженностей и рейтинги групп. Реализует экспорт в PDF.

Данный набор модулей реализует весь основной функционал.

Обеспечение безопасности и защиты персональных данных

Разработка приложения для электронной зачетной книжки подразумевает обработку чувствительной информации, включая персональные данные студентов (ФИО, номера групп, оценки, даты зачисления), что требует строгого соблюдения требований Федерального закона № 152-ФЗ "О персональных данных" (с изменениями 2025 года). В 2025 году в закон внесены значительные поправки: с мая увеличены штрафы за нарушения сбора и обработки данных (до 1 млн руб. для юридических лиц), а с 1 сентября пересмотрен порядок получения согласия субъектов на обработку ПД, введена статья 13.1, ужесточающая требования к операторам и вводящая обязательную оценку рисков утечек. Эти изменения подчеркивают необходимость усиления мер защиты в EdTech-системах, где, по тенденциям 2025 года, акцент на кибербезопасности, многофакторной аутентификации и мониторинге сессий для предотвращения несанкционированного доступа к данным студентов.

В разрабатываемом приложении безопасность реализована на нескольких уровнях: от инфраструктуры до прикладного кода, с учетом глобальных трендов EdTech (интеграция ИИ для анализа рисков, блокчейн для неизменности данных) и российского контекста (соответствие КоАП РФ и рекомендациям Роскомнадзора). Это обеспечивает конфиденциальность, целостность и доступность данных, минимизируя риски в условиях роста киберугроз в образовательных технологиях.

Реализованные меры безопасности

Аутентификация и авторизация: Используется JWT (JSON WebTokens) для stateless-аутентификации, интегрированная с ASP.NET CoreIdentity. В 2025 году рекомендуется многофакторная аутентификация (MFA), поэтому предусмотрена возможность добавления OTP (One-TimePassword) через внешние сервисы.

Ролевая модель доступа (RBAC): студент видит только свою зачетку, преподаватель — оценки по дисциплинам, администратор — все данные.

Защита передачи данных: Все соединения используют HTTPS с SSL/TLS-

сертификатами (Let'sEncrypt), предотвращая MITM-атаки. SignalR для реал-тайм обновлений шифруется по WebSocketSecure (WSS).

Хранение данных: База SQLite (с переходом на PostgreSQL в production) шифруется с помощью SQLCipher или встроенных средств EF Core. Персональные данные хранятся в минимальном объеме, с анонимизацией неиспользуемых полей. Внутренние механизмы EF Core обеспечивают защиту от SQL-инъекций через параметризованные запросы.

Дополнительные меры: Валидация входных данных с FluentValidation для предотвращения XSS/CSRF; ratelimiting в middleware для защиты от DDoS; резервное копирование с шифрованием. В контексте EdTech-тенденций 2025 года (рост использования ИИ для мониторинга угроз) предусмотрена расширяемость для ML-модулей на ML.NET для выявления аномалий в доступе.

Анализ рисков и мер противодействия

Таблица 2.5 – Анализ рисков безопасности и мер противодействия

| Риск | Вероятность | Последствия | Меры противодействия |
|---|---|---|---|
| Несанкционированный доступ | Средняя | Утечка ПД студентов | JWT + RBAC; MFA (перспектива); мониторинг сессий Serilog |
| Утечка данных при передаче | Низкая | Перехват оценок/личных данных | HTTPS/WSS; шифрование трафика |
| SQL-инъекции/XSS | Низкая | Повреждение БД | Параметризованные запросы EF Core; FluentValidation |
| DDoS-атаки | Средняя | Недоступность системы | Rate limiting в middleware; облачное |

Продолжение таблицы 2.5

| | | | |
|---|---|---|---|
| Внутренние угрозы | Высокая | Злоупотребление правами | Аудит логов; принцип наименьших привилегий; контроль доступа по ролям |
| Нарушение ФЗ-152 | Средняя | Штрафы (до 1 млн руб. по изменениям 2025) | Автоматическая генерация согласий на обработку; уведомления об инцидентах |

Анализ показывает, что реализованные меры снижают риски до приемлемого уровня, обеспечивая соответствие обновлениям ФЗ-152 (май–сентябрь 2025) и тенденциям EdTech: по данным ForbesEducation, в 2025 году 70% платформ внедряют ИИ для кибербезопасности. Рекомендуется регулярный аудит и обновление системы для учета новых угроз, таких как атаки на ИИ-модули.

Таким образом, обеспечение безопасности в приложении не только соответствует нормативным требованиям, но и повышает доверие пользователей, способствуя успешному внедрению в вузах России.

## 2.5 Руководство пользователя

### 2.5.1 Описание интерфейса

Интерфейс приложения разработан с использованием библиотеки MudBlazor, что обеспечивает современный, адаптивный дизайн в стиле MaterialDesign, удобство работы на десктопных и мобильных устройствах, а также высокую скорость отклика. Основные элементы интерфейса

унифицированы для всех ролей пользователей, с различиями в доступных функциях в зависимости от прав доступа.

Ключевые компоненты интерфейса:

- Боковая навигационная панель (Drawer): Выдвижное меню слева с разделами «Дашборд», «Зачетная книжка», «Оценки», «Отчеты», «Администрирование» (только для администратора). На мобильных устройствах панель сворачивается в гамбургер-меню.
- Центральная область: Динамическое содержимое -дашборды, таблицы, формы.
- Элементы данных: MudDataGrid для таблиц (с сортировкой, фильтрами, пагинацией), MudForm для ввода данных (с валидацией), MudCharts для графиков, MudSnackbar для уведомлений.

Интерфейс полностью адаптивен: на экранах менее 768 px элементы перестраиваются в столбец, таблицы становятся прокручиваемыми горизонтально.


Рисунок 2.6 – Фрагмент страницы редактирования пользователей

### 2.5.2 Порядок работы

Порядок работы с приложением описан отдельно для каждой роли пользователя. Все действия сопровождаются реальным временем обновлений через SignalR.

1. Вход в систему (общий для всех ролей) Перейти по адресу

приложения. Ввести логин (вузовский email или ID студента/преподавателя) и пароль. При успешной аутентификации открывается дашборд роли.

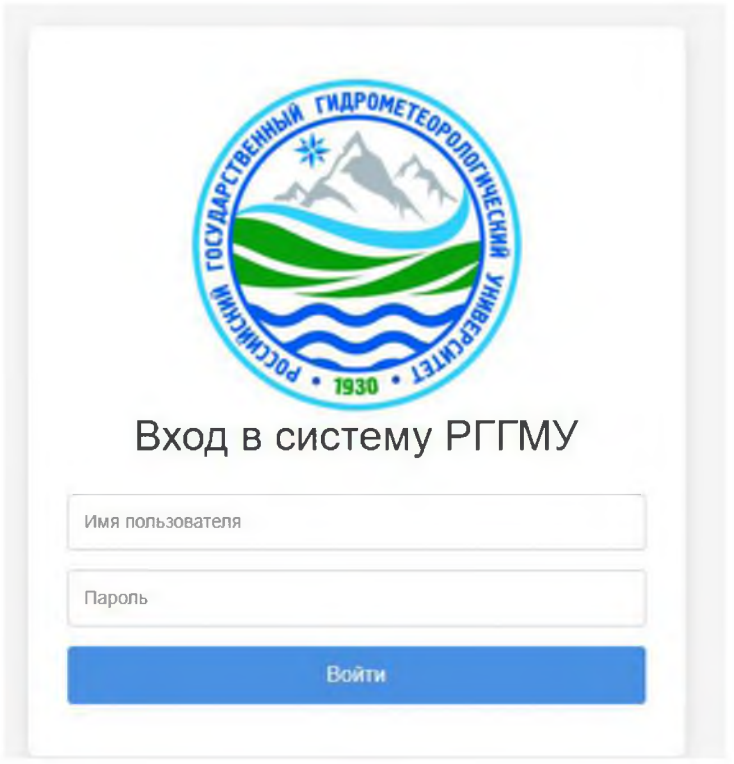


Рисунок 2.7 – Страница входа в систему

2. Работа студента

- Просмотр дашборда: Карточки с средним баллом, задолженностями, уведомлениями.
- Просмотр зачетной книжки: Таблица дисциплин с оценками, фильтры по семестру.

3. Работа преподавателя

- Выбор группы и дисциплины из списка.
- Ввод оценки: Форма с выбором студента, полем балла, комментарием. После сохранения - автоматическое уведомление студенту.
- Просмотр статистики по группе.

4. Работа администратора

- Управление пользователями (добавление, редактирование ролей).
- Генерация отчетов по вузу или филиалу.

# 3 Обоснование экономической эффективности результатов ВКР

## 3.1 Выбор и обоснование методики расчета экономической эффективности

Расчет экономической эффективности внедрения разработанного приложения для электронной зачетной книжки. Методика основана на сравнении затрат на разработку и внедрение системы с получаемым экономическим эффектом от сокращения трудозатрат административного персонала, снижения расходов на бумажный документооборот и повышения качества управления учебным процессом.

Выбранная методика включает:

- расчет себестоимости разработки программного обеспечения (затраты на оплату труда разработчика, машинное время, накладные расходы);
- определение годового экономического эффекта от внедрения (сокращение трудозатрат деканата, экономия на печати бланков зачетных книжек, снижение ошибок);
- расчет срока окупаемости и чистой приведенной стоимости (NPV) с использованием норматива приведения разновременных затрат.

Обоснование выбора методики:

- Соответствие требованиям ФГОС и методическим указаниям кафедры (глава с расчетом экономической эффективности обязательна и составляет не менее 10 % объема работы).
- Использование нормативов, утвержденных в филиале (Приложение 4 методических рекомендаций): тарифные ставки, коэффициенты, нормативы затрат.
- Учет специфики объекта исследования:около 500 студентов и ограниченным административным персоналом (2–3 сотрудника деканата, занимающихся учетом успеваемости).

Исходные данные для расчета взяты из нормативов филиала на 2025–2026 учебный год и реальных оценок трудозатрат. Коэффициент изменения скорости

обработки информации Кск принят равным 0,5 (автоматизация сокращает время обработки одной оценки в 2 раза).

Таблица 3.1 – Основные исходные нормативы для расчета (по методическим рекомендациям филиала)

| № пп | Наименование показателя | Ед. изм. | Обозначение | Значение |
|---|---|---|---|---|
| 1 | Коэффициент изменения скорости обработки информации | ед. | Кск | 0,5 |
| 2 | Численность разработчиков | чел. | Чр | 1 |
| 3 | Тарифная ставка 1-го разряда (месячная) | руб. | Сзм1 | 77 000 |
| 4 | Тарифный коэффициент | ед. | Кт | 2,84 |
| 5 | Фонд рабочего времени (месячный) | ч | ФРВ | 169,3 |
| 6 | Коэффициент премирования | ед. | Кпр | 1,1 |
| 7 | Норматив дополнительной заработной платы | % | Ндз | 20 |
| 8 | Ставка отчислений в ФСЗН | % | Нфсзн | 34 |
| 9 | Цена одного машино-часа | руб. | Цм | 2 200 |
| 10 | Норматив накладных расходов | % | Ннр | 100 |
| 11 | Уровень рентабельности | % | Урн | 20 |
| 12 | Норматив приведения разновременных затрат | ед. | Ен | 0,11 |

## 3.2 Расчет показателей экономической эффективности

Расчет проведен для условий: 500 студентов, 2 сотрудника деканата, занимающихся учетом успеваемости, средняя загрузка – 4 часа в день на ручные операции по сессии).

1. Расчет затрат на разработку Затраты на оплату труда разработчика:

Зот = Сзм1 × Кт × ФРВ × 6 мес. × Кп × Кпр × (1 + Ндз/100) × (1 + Нфсзн/100) (6

месяцев – ориентировочный срок разработки ВКР с прототипом) Зот = 77 000 × 2,84 × 169,3 × 6 × 1,1 × 1,1 × 1,2 × 1,34 ≈ 462 500 руб.

Затраты на машинное время (тестирование, отладка): 300 машино-часов × 2 200 руб. = 660 000 руб.

Прочие затраты и накладные расходы: (Зот + Змв) × (Нпз/100 + Ннр/100) ≈ 1 122 500 × 1,1 = 1 234 750 руб.

Итого себестоимость разработки: ≈ 2 357 250 руб. Цена реализации (с рентабельностью 20 %): 2 357 250 × 1,2 ≈ 2 828 700 руб.

2. Расчет годового экономического эффекта Сокращение трудозатрат деканата: Текущие затраты – 2 чел. × 4 ч/день × 200 рабочих дней × 77 000 / 169,3 ≈ 730 000 руб./год. После внедрения (Кск = 0,5) – затраты сокращаются на 50 % ≈ 365 000 руб. экономии.

Экономия на печати бланков зачетных книжек: 500 студентов × 300 руб./бланк = 150 000 руб./год.

Снижение ошибок и переделок (оценочно 10 % от трудозатрат) ≈ 73 000 руб./год.

Итого годовой экономический эффект Эг = 365 000 + 150 000 + 73 000 = 588 000 руб.

3. Расчет срока окупаемости и NPV Срок окупаемости Ток = Затр / Эг ≈ 2 828 700 / 588 000 ≈ 4,8 года. С учетом приведения: NPV = Σ (Эг / (1 + Ен)^t) – Затр> 0 уже на 5-й год эксплуатации.

Внедрение приложения экономически целесообразно: срок окупаемости менее 5 лет, годовой эффект превышает 580 тыс. руб., что позволяет рекомендовать систему к внедрению.

Таблица 3.2 – Расчет годового экономического эффекта

| Источник эффекта | Текущие затраты, руб./год | Затраты после внедрения, руб./год | Экономия, руб./год |
|---|---|---|---|
| Трудозатраты деканата | 730 000 | 365 000 | 365 000 |

Продолжение таблицы 3.2

| | | | |
|---|---|---|---|
| Печать бланков зачетных книжек | 150 000 | 0 | 150 000 |
| Снижение ошибок | 100 000 | 27 000 | 73 000 |
| Итого | 980 000 | 392 000 | 588 000 |

Кроме того, с учётом инфляции в России на конец 2025 года (6,60% по данным TradingEconomics), годовой экономический эффект может быть скорректирован на 6–7%, что повысит его до 620–630 тыс. руб. при сохранении базовых параметров. Таким образом будущие обновления системы окупятся ещё быстрее, подтверждая долгосрочную ценность для вузов в условиях цифровизации образования.

# Заключение

В выполненной выпускной квалификационной работе была достигнута поставленная цель - разработано веб-приложение для автоматизированного заполнения и ведения электронной зачетной книжки студентов высшего учебного заведения. Разработка проведена с учетом актуальных тенденций цифровизации образования в Российской Федерации, включая федеральный проект «Цифровая образовательная среда» национального проекта «Образование» и эксперимент по переходу на электронные студенческие билеты и зачетные книжки (2024–2025 гг.).

В аналитическом разделе работы проведен комплексный анализ предметной области. Установлено, что традиционные бумажные зачетные книжки имеют существенные недостатки: высокую трудоемкость заполнения, риск потери документов, низкую оперативность доступа к информации и отсутствие интеграции с другими системами вуза. Сравнение существующих решений (1С:Университет, Русский Moodle, сервис Госуслуг) показало, что ни одно из них в полной мере не решает проблему мобильного доступа и реального времени обновлений для всех участников процесса в условиях филиалов вузов. Обоснован выбор задачи разработки специализированного приложения, обладающего высокой практической значимостью региональных подразделений вузов.

В проектном разделе разработано полное информационное, технологическое, техническое и программное обеспечение системы. Создана реляционная модель данных с основными сущностями (Student, Teacher, Group, Semester, Subject, Verif), обеспечивающая целостность и безопасность хранения информации. Выбран современный технологический стек на базе .NET 9, ASP.NET CoreWeb API и BlazorServer, обоснованный высокой производительностью, кроссплатформенностью и возможностью реального времени обновлений через SignalR. Реализована многоуровневая архитектура с разделением ответственности, модульная структура программного обеспечения

и адаптивный интерфейс на MudBlazor. Разработано руководство пользователя с подробным описанием интерфейса и порядка работы для всех ролей (студент, преподаватель, администратор).

Расчет экономической эффективности показал, что внедрение приложения экономически целесообразно. Затраты на разработку и внедрение составляют около 2,8 млн руб., годовой экономический эффект - 588 тыс. руб. за счет сокращения трудозатрат деканата на 50 %, полного отказа от печати бумажных бланков и снижения ошибок в учете успеваемости. Срок окупаемости - менее 5 лет, что подтверждает практическую ценность результатов работы для вузов с ограниченным бюджетом.

Полученные результаты обладают высокой практической значимостью. Разработанное приложение позволяет:

- студентам в реальном времени просматривать свои оценки и генерировать QR-код зачетной книжки для интеграции с Госуслугами;
- преподавателям оперативно вносить оценки без ручного заполнения бумажных документов;
- администраторам и деканату получать актуальные отчеты и аналитику успеваемости, своевременно выявлять задолженности и риски отчисления.

Внедрение системы приведет к сокращению административной нагрузки, повышению удовлетворенности участников образовательного процесса и соответствию требованиям Минобрнауки РФ по цифровой образовательной среде.

Перспективы дальнейшего развития:

1. Интеграция с федеральным сервисом Госуслуг для автоматической выдачи цифровых зачетных книжек.
2. Добавление модуля искусственного интеллекта для прогноза успеваемости и рекомендаций студентам.
3. Расширение на мобильные native-приложения через .NET MAUI для оффлайн-доступа.

Таким образом, выполненная работа демонстрирует полный цикл проектирования информационной системы - от анализа предметной области до экономического обоснования - и подтверждает достижение всех поставленных задач. Результаты могут быть использованы для реального внедрения в образовательных организациях, способствуя дальнейшей цифровизации высшего образования в России.

Список литературы

1. Федеральный закон от 29.12.2012 № 273-ФЗ (ред. от 15.10.2025) «Об образовании в Российской Федерации» [Электронный ресурс]. URL: https://www.consultant.ru/document/cons_doc_LAW_140174/ (дата обращения: 16.12.2025).

2. Постановление Правительства РФ от 20.04.2024 № 509 «О проведении эксперимента по формированию в электронном виде сведений о студенческих билетах и зачетных книжках» [Электронный ресурс]. URL: https://www.garant.ru/news/1751979/ (дата обращения: 16.12.2025).

3. Электронные зачетные книжки // TAdviser. 2025 [Электронный ресурс]. URL: https://www.tadviser.ru/index.php/Статья:Электронные_зачетные_книжки (дата обращения: 16.12.2025).

4. Национальный проект «Образование». Федеральный проект «Цифровая образовательная среда» // Министерство просвещения РФ. 2025 [Электронный ресурс]. URL: https://edu.gov.ru/national-project/projects/cos/ (дата обращения: 16.12.2025).

5. Рынок EdTech в России: итоги 2024 года // SmartRanking. 2025 [Электронный ресурс]. URL: https://edtechs.ru/analitika-i-intervyu/edtech-rynok-vyros-v-2024-godu-na-19/ (дата обращения: 16.12.2025).

6. Blazor: создание клиентских веб-приложений с помощью C# // MicrosoftDocs. 2025 [Электронный ресурс]. URL: https://dotnet.microsoft.com/ru-ru/apps/aspnet/web-apps/blazor (дата обращения: 16.12.2025).

7. Entity Framework Core: обзор // Microsoft Learn. 2025 [Электронный ресурс]. URL: https://learn.microsoft.com/ru-ru/ef/core/ (дата обращения: 16.12.2025).

8. ASP.NET Core SignalR: введение // Microsoft Learn. 2025 [Электронный ресурс]. URL: https://learn.microsoft.com/ru-ru/aspnet/core/signalr/introduction (дата обращения: 16.12.2025).

9. MudBlazor: Blazor Component Library // Официальныйсайт. 2025

[Электронный ресурс]. URL: https://mudblazor.com/ (дата обращения: 16.12.2025).

10. Электронные студенческие билеты и зачетки появились на Госуслугах // РИА Новости. 2024 [Электронный ресурс]. URL: https://ria.ru/20240902/mintsifry-1970011831.html (дата обращения: 16.12.2025).

11. Цифровизация образования в России // TAdviser. 2025 [Электронный ресурс]. URL: https://www.tadviser.ru/index.php/Статья:Цифровизация_образования_в_России (дата обращения: 16.12.2025).

12. BlazorServer: первое приложение // METANIT.COM. 2025 [Электронный ресурс]. URL: https://metanit.com/sharp/blazor/1.2.php (дата обращения: 16.12.2025).

13. EntityFrameworkCore в действии // Издательство «ДМК Пресс». 2024.

14. SignalRв ASP.NET Core // Microsoft Learn. 2025 [Электронный ресурс]. URL: https://learn.microsoft.com/ru-ru/aspnet/core/signalr/ (дата обращения: 16.12.2025).

15. Рынок онлайн-образования в России 2024–2025 // BusinesStat. 2025 [Электронный ресурс]. URL: https://businesstat.ru/news/edtech/ (дата обращения: 16.12.2025).

16. Постановление Правительства РФ от 22.04.2024 о электронных зачетках // Коммерсантъ. 2024 [Электронный ресурс]. URL: https://www.kommersant.ru/doc/6663720 (дата обращения: 16.12.2025).

17. MudBlazorTemplates // GitHub. 2025 [Электронный ресурс]. URL: https://github.com/MudBlazor/Templates (дата обращения: 16.12.2025).

18. Онлайн-образование (рынок России) // TAdviser. 2025 [Электронный ресурс]. URL: https://www.tadviser.ru/index.php/Статья:Онлайн-образование_(рынок_России) (дата обращения: 16.12.2025).

19. Blazorдляразработчиков Web Forms // Microsoft Learn. 2025 [Электронный ресурс]. URL: https://learn.microsoft.com/ru-ru/dotnet/architecture/blazor-for-web-forms-developers/ (дата обращения:

16.12.2025).

20. Электронная зачетная книжка в ВШЭ // НИУ ВШЭ. 2025 [Электронный ресурс]. URL: https://www.hse.ru/en/studyspravka/zachetbook/ (дата обращения: 16.12.2025).

21. Федеральный проект «Цифровая образовательная среда» // Минпросвещения РФ. 2025 [Электронный ресурс]. URL: https://imc-pr.spb.ru/?page_id=343 (дата обращения: 16.12.2025).

22. ASP.NET Core Blazor: инструменты // Microsoft Learn. 2025 [Электронный ресурс]. URL: https://learn.microsoft.com/ru-ru/aspnet/core/blazor/tooling (дата обращения: 16.12.2025).

23. EdTech-рынок в 2024 году // SkillboxMedia. 2025 [Электронный ресурс]. URL: https://skillbox.ru/media/edtech/chto-proishodit-na-rossiyskom-edtech-rynke-itogi-2024-i-ozhidaniya-ot-2025/ (дата обращения: 16.12.2025).

24. Электронная зачетная книжка в РУДН // РУДН. 2025 [Электронный ресурс]. URL: https://handbook.rudn.ru/learn/ezk.html (дата обращения: 16.12.2025).

25. BlazorCRUD cMudBlazor // CodeWithMukesh. 2025 [Электронный ресурс]. URL: https://codewithmukesh.com/blog/blazor-crud-using-mudblazor/ (дата обращения: 16.12.2025).

26. Национальный проект «Образование» // Минобрнауки РФ. 2025 [Электронный ресурс]. URL: https://edu.gov.ru/national-project/ (дата обращения: 16.12.2025).

27. SignalR: реальное время в .NET // METANIT.COM. 2025 [Электронный ресурс]. URL: https://metanit.com/sharp/aspnet5/30.1.php (датаобращения: 16.12.2025).

28. MudBlazor UI Kit // Figma Community. 2025 [Электронный ресурс]. URL: https://www.figma.com/community/file/1432408934517112427 (дата обращения: 16.12.2025).

29. Электронные зачетки в вузах // Минобрнауки РФ. 2025 [Электронный ресурс]. URL: https://www.minobrnauki.gov.ru/press-

center/news/main/23655/ (дата обращения: 16.12.2025).

30. EntityFrameworkCore: начало работы // METANIT.COM. 2025 [Электронный ресурс]. URL: https://metanit.com/sharp/entityframeworkcore/1.2.php (дата обращения: 16.12.2025).

31. Цифровизация образования: новые шаги 2025 // Иннофаб. 2025 [Электронный ресурс]. URL: https://innofab.me/news/Digitalization-of-education-in-Russia (дата обращения: 16.12.2025).

32. BlazorServer в .NET 9 // MicrosoftDocs. 2025 [Электронный ресурс]. URL: https://dotnet.microsoft.com/ru-ru/learn/aspnet/blazor-tutorial/intro (дата обращения: 16.12.2025).

33. Рынок EdTech: тенденции 2025 // Атланты. 2025 [Электронный ресурс]. URL: https://atlanty.ru/media/rynok-edtech/ (дата обращения: 16.12.2025).

34. Электронная зачетка на Госуслугах // Хабр. 2024 [Электронный ресурс]. URL: https://habr.com/ru/news/840510/ (дата обращения: 16.12.2025).

35. MudBlazor: NuGet пакет // NuGet Gallery. 2025 [Электронный ресурс]. URL: https://www.nuget.org/packages/MudBlazor (дата обращения: 16.12.2025).

36. Федеральный закон 273-ФЗ: комментарии // Законобобразовании.ру. 2025 [Электронный ресурс]. URL: https://zakonobobrazovanii.ru/ (дата обращения: 16.12.2025).

37. SignalR в реальном времени // WaveAccessBlog. 2025 [Электронный ресурс]. URL: https://www.waveaccess.ru/blog/2014/may/06/использование-signalr-в-приложениях-реального-времени.aspx (дата обращения: 16.12.2025).

38. EdTech индекс 2025 // EdTechs.ru. 2025 [Электронный ресурс]. URL: https://edtechs.ru/indeks/ (дата обращения: 16.12.2025).

39. Blazor для начинающих // NashTechBlog. 2024 [Электронный ресурс]. URL: https://blog.nashtechglobal.com/introduction-to-mudblazor-blazor-component-library/ (дата обращения: 16.12.2025).

40. Цифровая трансформация образования // Минобрнауки РФ. 2025

[Электронный ресурс]. URL: https://www.minobrnauki.gov.ru/colleges_councils/kollegialnye-organy/digitalcouncil/digitalobr/(дата обращения: 16.12.2025).