УДК 004.4
DOI: 10.26102/2310-6018/2025.51.4.033

Архитектура и реализация клиент-серверного решения для организации мобильного доступа к учебным данным

А.М. Абрамов, И.В. Аникин✉, А.А. Бурдина, Д.Р. Гайфуллин, А.А. Грудцин, И.С. Добрынин, Б.И. Лупанов, Р.А. Мулюков, А.Э. Плотник

Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация

Резюме. Представлена архитектура и реализация клиент-серверного решения, обеспечивающего мобильный доступ к учебным данным университета. Целью работы являлось повышение эффективности реализации ряда процессов образовательной деятельности, обеспечивающих работу с такими учебными данными, как расписание занятий, результаты текущего и промежуточного контроля, информация по преподавателям и учебной группе, новостной портал. Проект не только устраняет ряд недостатков, свойственных текущей ИТ-инфраструктуре университета, но и создает основу для дальнейшей цифровой трансформации образовательного процесса. Описана архитектура решения и особенности ее реализации, в том числе выбор стека технологий. Подробно рассмотрены основные компоненты архитектуры приложения, такие как gRPC-мост для интеграции с устаревшими системами, REST API для взаимодействия с внешними сервисами и система мониторинга на основе Prometheus и Grafana. Применение современных технологий, таких как Golang, PostgreSQL, gRPC и облачных сервисов Yandex Cloud, обеспечило высокую производительность, масштабируемость и безопасность. Интеграция с существующими системами университета посредством gRPC-моста обеспечила совместимость и эффективный обмен данными. Представлены результаты внедрения приложения в реальную IT-инфраструктуру вуза, демонстрирующие повышение доступности и удобства работы с учебными данными и снижение нагрузки. Разработанный подход может быть успешно адаптирован и применен для других высших учебных заведений. В перспективе планируется расширение функциональности приложения за счет интеграции с AI-алгоритмами для прогнозирования академических рисков и формирования индивидуальных образовательных траекторий.

Ключевые слова: цифровизация образовательного процесса, микросервисная архитектура, gRPC, REST API, PostgreSQL.

Для цитирования: Абрамов А.М., Аникин И.В., Бурдина А.А., Гайфуллин Д.Р., Грудцин А.А., Добрынин И.С., Лупанов Б.И., Мулюков Р.А., Плотник А.Э. Архитектура и реализация клиент-серверного решения для организации мобильного доступа к учебным данным. *Моделирование, оптимизация и информационные технологии.* 2025;13(4). URL: https://moitvivt.ru/ru/journal/pdf?id=2048 DOI: 10.26102/2310-6018/2025.51.4.033

Architecture and design of client-server application for mobile access to educational data

A.M. Abramov, I.V. Anikin✉, A.A. Burdina, D.R. Gaifullin, A.A. Grudtsin, I.S. Dobrynin, B.I. Lupanov, R.A. Mulyukov, A.E. Plotnik

Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation

Abstract. The paper presents the architecture and implementation of a client-server solution that provides mobile access to university educational data. The aim of the work was to improve the efficiency of a number of educational activity processes that provide work with educational data such as class schedules, current and midterm assessment results, information on teachers and the study group, and a

© Абрамов А.М., Аникин И.В., Бурдина А.А., Гайфуллин Д.Р., Грудцин А.А., Добрынин И.С., Лупанов Б.И., Мулюков Р.А., Плотник А.Э., 2025

news portal. The project not only bypasses a number of shortcomings inherent in the university's current IT infrastructure, but also creates a basis for further digital transformation of the educational process. The paper describes the architecture of the solution and its implementation features, including the choice of technology stack. The main components of the application architecture are considered in detail, such as the gRPC bridge for integration with legacy systems, REST API for interaction with external services, and a monitoring system based on Prometheus and Grafana. The use of modern technologies such as Golang, PostgreSQL, gRPC, and Yandex Cloud services ensured high performance, scalability, and security. Integration with existing university systems via the gRPC bridge ensured compatibility and efficient data exchange. The results of the application implementation in the real IT infrastructure of the university are presented, demonstrating the increase in accessibility and convenience of working with educational data and the reduction of the load. The developed approach can be successfully adapted and applied to other higher education institutions. In the future, it is planned to expand the functionality of the application through integration with AI algorithms for predicting academic risks and forming individual educational trajectories.

Keywords: digitalization of education, microservice architecture, gRPC, REST API, PostgreSQL.

For citation: Abramov A.M., Anikin I.V., Burdina A.A., Gaifullin D.R., Grudtsin A.A., Dobrynin I.S., Lupanov B.I., Mulyukov R.A., Plotnik A.E. Architecture and design of client-server application for mobile access to educational data. *Modeling, Optimization and Information Technology*. 2025;13(4). (In Russ.). URL: https://moitvivt.ru/ru/journal/pdf?id=2048 DOI: 10.26102/2310-6018/2025.51.4.033

Введение

В условиях динамично развивающегося рынка образовательных услуг и возрастающих требований к качеству образования, цифровизация учебного процесса становится одной из важнейших целей для современных университетов. Их конкурентоспособность напрямую зависит от эффективности работы цифровых сервисов, обеспечивающих поддержку реализации ключевых бизнес-процессов университета, к основным из которых относятся процессы реализации образовательной деятельности. К типовым процессам данного вида относятся процессы организации довузовской подготовки, приемной кампании, управления контингентом, нагрузкой, расписанием, текущим и промежуточным контролем, практиками, государственной итоговой аттестацией и др.

Наряду с актуальностью цифровизации процессов образовательной деятельности, решение данных задач во многих университетах осложняется рядом сложившихся исторически обстоятельств, к основным из которых следует отнести следующие:

– применение большого количества разнотипных и изначально плохо взаимодействующих между собой систем для цифровизации различных сервисов;

– IT-инфраструктура многих университетов характеризуется фрагментарностью, сложностью интеграции различных систем, ограниченной масштабируемостью;

– отсутствие открытых интерфейсов для взаимодействия различных сервисов;

– не отчуждаемость применяемых решений по цифровизации.

Данные обстоятельства приводят к снижению эффективности реализации процессов образовательной деятельности, проблемам с доступностью и удобством работы с данными, увеличению нагрузки на преподавательский состав.

Цель данной работы заключается в повышении эффективности реализации ряда процессов образовательной деятельности с учетом выше представленных сложностей за счет разработки и реализации архитектуры серверного и мобильного приложений, интегрированных с существующей IT-инфраструктурой вуза и обеспечивающих доступ к таким учебным данным, как расписание занятий, результатам текущего и промежуточного контроля, информации по преподавателям и учебной группе, новостному порталу.

Для достижения поставленной цели в одном из университетов, в качестве которого был выбран КНИТУ-КАИ, были решены следующие задачи:

– проведен анализ существующей IT-инфраструктуры и зафиксированы текущие особенности цифровизации процессов образовательной деятельности;

– разработана и реализована архитектура приложений, обеспечивающих доступ к обозначенным учебным данным;

– проведена интеграция приложений с существующими решениями в вузе;

– проведена интеграция с внешними сервисами: ВКонтакте, голосовым помощником «Яндекс Алиса»;

– проведено тестирование и оценка эффективности работы разработанной архитектуры.

Разработанное решение является масштабируемым и отчуждаемым, может быть эффективно использовано для решения задач цифровизации других высших учебных заведений.

Материалы и методы

Анализ текущей ситуации. В настоящее время в КНИТУ-КАИ для решения задач управления учебной нагрузкой, расписанием занятий, результатами текущего и промежуточного контроля, используются следующие информационные системы:

– автоматизированная система управления учебным процессом АСУ «Деканат» — собственная разработка университета, обеспечивающая учет контингента обучающихся, фиксацию хода образовательного процесса, результатов текущей успеваемости (балльно-рейтинговая система), промежуточной и итоговой аттестации и другие задачи управления образовательным процессом;

– информационная система «Авторасписание» внешней разработки (Лаборатория ММИС), позволяющая формировать расписание без «окон» для учебных групп, оптимизировать в расписании «окна» преподавателей, учитывать характер работы и пожелания каждого преподавателя, учитывать характер работы и пожелания каждого преподавателя, оптимально размещать занятия по аудиториям с учетом количества студентов, учитывать разновидности занятий и учебных дисциплин, приоритеты и пожелания преподавателей и кафедр, вместимость аудиторий, оптимизировать количество переходов из аудитории в аудиторию, и из корпуса в корпус.

В университете осуществлена интеграция АСУ Деканат и Авторасписания в части экспорта и импорта информации. Схема интеграции представлена на Рисунке 1.

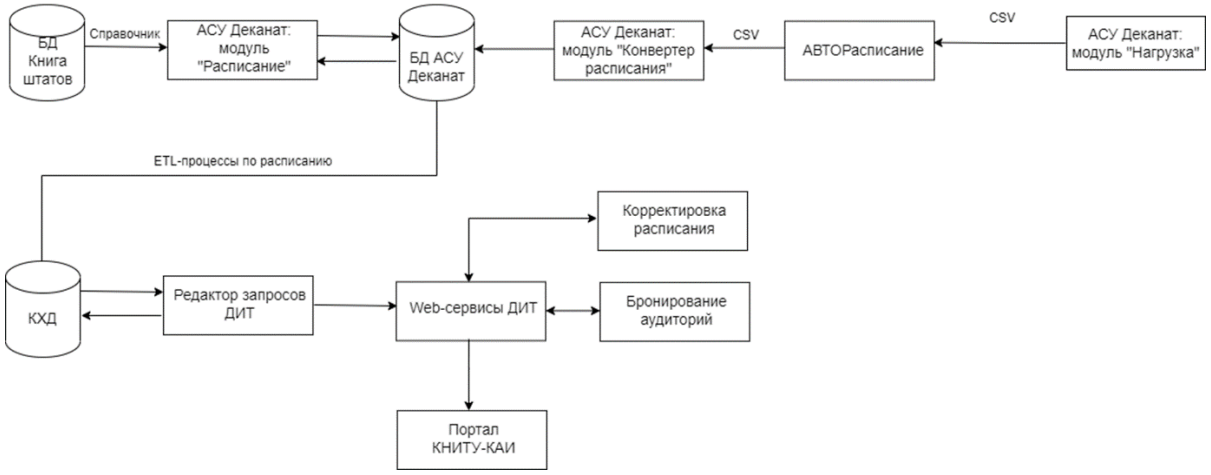

Рисунок 1 – Схема интеграции АСУ Деканат и Авторасписания
Figure 1 – Scheme of integration of the ACS of the Dean's Office and Auto-writing

Проведенный анализ позволил выделить следующие недостатки текущей реализации:

1) ограниченная доступность актуальной информации о расписании занятий – процесс получения расписания сопряжен с необходимостью ручной идентификации подгруппы и определения четности недели, что создает дополнительные когнитивные нагрузки на пользователей и увеличивает вероятность ошибок;

2) несоответствие пользовательских интерфейсов современным требованиям эргономики;

3) периодическая недоступность сервисов, особенно в периоды повышенной нагрузки, оказывает негативное влияние на непрерывность учебного процесса.

Для преодоления указанных недостатков был предложен подход, включающий:

1) обеспечение удобного и беспрепятственного доступа к актуальной информации о расписании занятий посредством разработки специализированных мобильного и веб-приложения;

2) предоставление доступа к данным об аттестациях, обеспечивающего возможность оперативного контроля успеваемости;

3) формирование структурированной информации о задолженностях, облегчающей процесс планирования и организации учебной деятельности;

4) предоставление унифицированного API [1] для интеграции с внешними сервисами, обеспечивающего возможность расширения функциональности IT-инфраструктуры;

5) реализацию механизмов для работы с заданиями, обеспечивающих возможность эффективного планирования и контроля выполнения учебных, а также внеучебных задач;

Предложенный подход был реализован в приложении «КапиПара», обеспечивающего доступ к учебным данным. При проектировании приложения были учтены следующие требования:

1) оптимизация производительности с учетом пиковых нагрузок, обеспечивающая стабильную работу приложения в периоды максимальной активности пользователей;

2) обеспечение кроссплатформенности, гарантирующее возможность использования приложения на различных типах устройств и операционных системах;

3) масштабируемость архитектуры, позволяющая адаптировать систему к растущим потребностям пользователей и увеличению объема данных;

4) интеграция с существующими сервисами вуза, обеспечивающая совместимость и возможность обмена данными между различными IT-системами;

5) обеспечение независимости архитектуры, гарантирующее возможность модификации и расширения функциональности приложения без влияния на другие компоненты системы.

Для достижения поставленных целей была выбрана микросервисная архитектура, обеспечивающая гибкость и масштабируемость системы. Использование отдельных независимых компонентов упрощает разработку, тестирование и развертывание, а также позволяет оперативно реагировать на изменяющиеся потребности пользователей.

Архитектура разработанного решения. Архитектура разработанного решения «КапиПара» представлена на Рисунке 2.

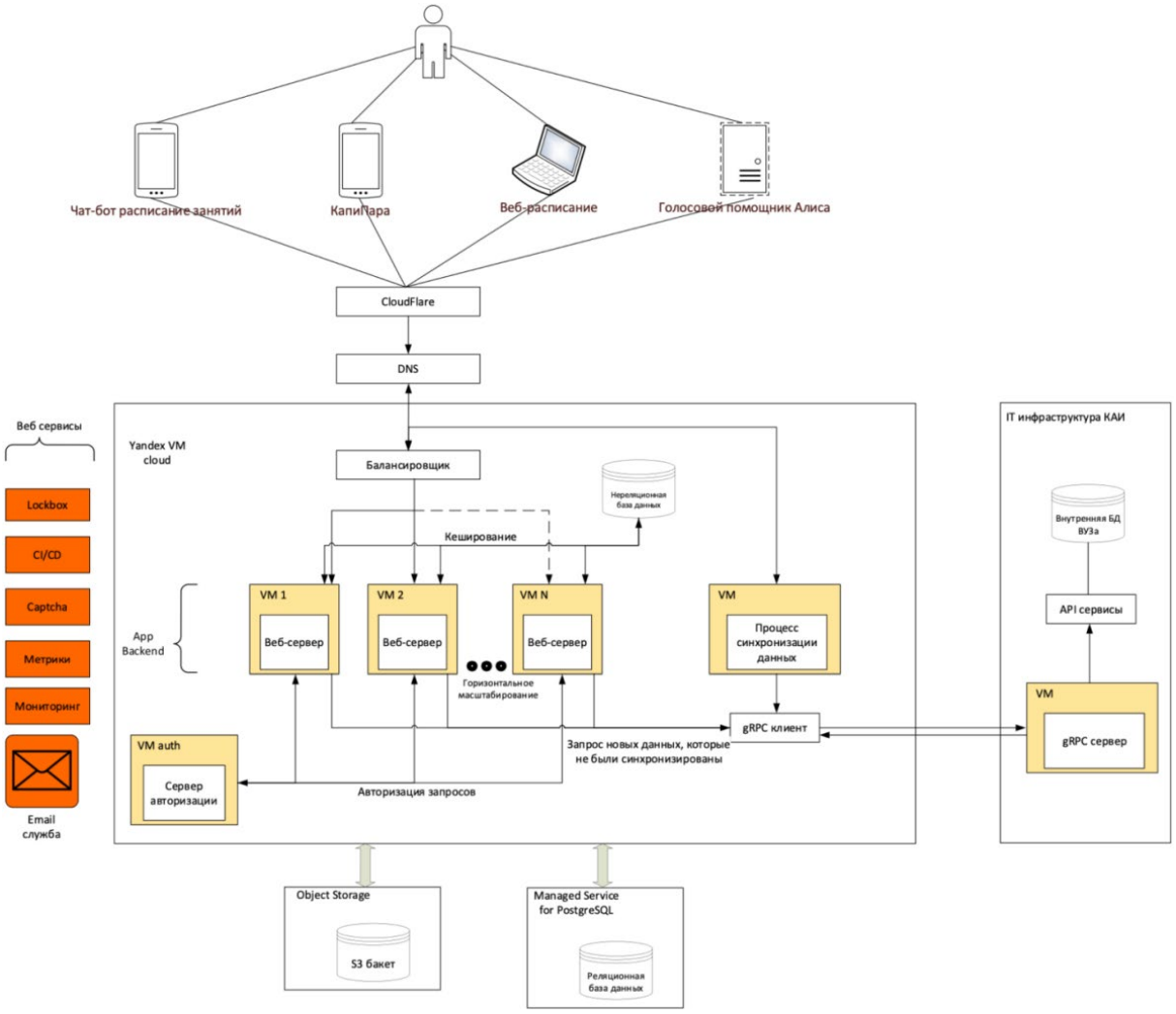

Рисунок 2 – Архитектура решения «КапиПара», обеспечивающего доступ к учебным данным
Figure 2 – Architecture of the Kapipara solution providing access to educational data

Решение разработано с использованием программного обеспечения для автоматизации развертывания и контейнеризации приложений Docker [2], языка программирования Go и Dart, фреймворк Flutter, СУБД PostgreSQL. Для обмена данными реализована REST API. Реализованы автоматические миграции [3]. Реализована авторизация через Keycloak с поддержкой SSO и внешних OpenID провайдеров, что позволяет организовать авторизацию, используя учетные записи Яндекс, ВКонтакте и т. п. Для автоматизации сборки и доставки кода используются CI/CD пайплайны. Для сбора и визуализации метрик используются системы с открытым исходным кодом Prometheus и Grafana, мониторинг и диагностика проблем – средствами Firebase. Специалистами департамента информационных технологий (ДИТ) реализован API с авторизацией, используя который по протоколу gRPC, происходит синхронизация данных с «внешним» по отношению к ИС вуза контуром. Внешний контур реализован на облачных технологиях Яндекса. Кэширование данных позволило улучшить надежность системы т. к. была исключена прямая зависимость получения данных от работоспособности БД ИС ВУЗа, а также увеличить скорость реакции мобильного приложения.

Для автоматической генерации документации API был использован инструмент Swagger, что обеспечило унификацию взаимодействия между backend- и frontend-

разработчиками. Автоматически генерируемая документация в формате OpenAPI упрощает интеграцию новых сервисов и обеспечивает возможность тестирования API [4].

Для обеспечения безопасности кода был применен статический анализатор уязвимостей gosec, интегрированный в CI/CD-пайплайн (конвейер непрерывной интеграции и непрерывной поставки) [5]. Это позволило автоматически проверять исходный код на типовые ошибки безопасности, такие как SQL-инъекции, небезопасное использование криптографических функций и утечки чувствительных данных, а также исключить попадание небезопасного кода в «боевую» систему.

Дополнительные компоненты инфраструктуры:

1) Valkey (Redis-совместимая СУБД). Использовалась в качестве кэширующего слоя для часто запрашиваемых данных, таких как расписание на текущий день, новостная лента и результаты аттестаций. Это позволило значительно снизить нагрузку на основную базу данных и обеспечить быстрый доступ к информации для пользователей [6].

2) gRPC. Протокол gRPC был применен для обеспечения высокопроизводительного взаимодействия между микросервисами при синхронизации данных с сервисами вуза. Использование gRPC позволило сократить объем передаваемых данных и повысить скорость обмена информацией.

3) Yandex Object Storage. Для хранения медиаконтента был выбран Yandex Object Storage, обеспечивающий масштабируемость и надежность хранения данных. Интеграция с Yandex Object Storage была реализована через S3-совместимый API, что позволило упростить разработку и развертывание приложения.

4) Yandex Lockbox. Для централизованного управления секретами приложения (ключами API, паролями базы данных, сертификатами TLS) использовался сервис Yandex Lockbox. Это исключило необходимость хранения чувствительных данных в коде или конфигурационных файлах и повысило безопасность системы [7].

5) Keycloak (с Keycloakify). Для реализации функций аутентификации и авторизации был выбран Keycloak [8]. Поддержка протоколов OAuth 2.0 и OpenID Connect обеспечила возможность единого входа (SSO) для различных сервисов приложения, а использование Keycloakify позволило кастомизировать интерфейс Keycloak в соответствии с фирменным стилем университета.

6) Prometheus и Grafana. Для мониторинга и визуализации работы приложения были применены системы Prometheus и Grafana. Prometheus собирает метрики работы приложения (время ответа API, загрузка CPU, частота ошибок), а Grafana визуализирует эти данные в виде наглядных дашбордов, что позволяет оперативно отслеживать состояние системы и выявлять потенциальные проблемы.

Результаты и обсуждение

Научная новизна полученных результатов заключается в том, что в отличие от классических решений, основанных на одноконтурном размещении всех компонентов приложения внутри одной сети (контура) вуза, разработанная архитектура позволила нивелировать проблемы периодического отказа сети и/или отдельных критически важных компонентов путем выделения 2 контуров, главный из которых вынесен в облачную инфраструктуру. Основная задача главного контура заключается в том, чтобы справляться с высокими нагрузками за счет отсутствия жесткой зависимости от неустойчивых компонентов сети и сервисов вуза. Облачные провайдеры гарантируют высокий показатель uptime, который не может сравниться с показателями локальной информационной сети вуза.

Ранее, при реализации классической одноконтурной архитектуры, запрос пользователя попадал прямиком на применяемые в вузе сервисы, построенные на устаревших технологиях, которые не способны выдерживать большие нагрузки с большим количеством пользователей в реальном времени. Разработанная архитектура реорганизует иным способом подход к хранению данных, обеспечению бесперебойного доступа к ресурсам за счет «ленивой» синхронизации данных между старым и новым решением. В данном случае нет острой необходимости быстрой модернизации существующих и разрабатываемых достаточно давно legacy-систем вуза, что потребовало бы больших ресурсных затрат. В рамках предложенной архитектуры достаточно лишь «обернуть» старые информационные системы в новый фасад и развивать новые приложения уже на современном отказоустойчивом стеке.

В качестве канала синхронизации между двумя контурами можно использовать современные решения в области построения API. Авторами статьи использован gRPC. В данном случае также нет необходимости делать поправки на множество ограничений уже существующих в вузе решений.

Еще одной новизной предложенного подхода является то, что для дополнительного снижения нагрузки на старый стек технологий вуза, в используемом механизме «ленивой» синхронизации учтен фактор приоритезации необходимых к обновлению данных. Сначала обновляются данные тех пользователей, которые пользуются сервисом чаще всего, а остальные – по остаточному принципу. Так, если система фиксирует «усталость» старых сервисов от многочисленных запросов на обновление данных, она уходит в задержку по экспоненциальному принципу [9]. В этом случае, с каждой неудачей время ожидания перед повторной попыткой передачи данных увеличивается экспоненциально. Это дает возможность выполнить все запросы на актуализацию данных пользователей, и отработать другим внутренним сервисам вуза, которые также могут выполнять обращения к той же информации. В данном подходе устаревшие сервисы вуза не уходят в отказ для всех пользователей, а для приоритетных из них данные остаются в максимально актуальном состоянии.

Как итог, разработанное решение позволяет оперативно получать множеству пользователей доступ к следующим учебным данным, предоставляемым разноплановыми сервисами вуза:

- расписанию занятий;
- аттестациям;
- мониторингу академических задолженностей;
- информации о преподавателях и академических группах;
- агрегатору новостей.

В предложенном решении реализована интеграция с голосовым помощником Алиса.

Проведено тестирование и оценка эффективности работы разработанной архитектуры. Синхронизация данных с существующими сервисами университета осуществлялась в автоматическом режиме, обеспечивая ежедневное обновление расписания, аттестаций и других необходимых данных, несмотря на ограничения, связанные с низким быстродействием и периодической недоступностью этих сервисов.

Анализ загрузки центрального процессора показал, что пики нагрузки системы приходятся на периоды непосредственно перед началом занятий (09:40, 11:15, 13:25). Полученные данные свидетельствуют о высокой востребованности разработанного приложения «КапиПара» среди студентов и позволяют оптимизировать распределение ресурсов сервера с целью обеспечения максимальной производительности в периоды наибольшей активности пользователей. На Рисунке 3 представлен график загрузки CPU в течение учебного дня, иллюстрирующий пиковые нагрузки.

Рисунок 3 – График загрузки CPU
Figure 3 – CPU load

Данные, полученные в результате анализа загрузки CPU, позволяют применить методы динамического масштабирования ресурсов (например, с использованием Kubernetes), обеспечивая автоматическое увеличение вычислительной мощности в периоды пиковой нагрузки и снижая затраты на содержание инфраструктуры в периоды низкой активности [10].

Анализ полученных метрик выявил четкую корреляцию пиков нагрузки на ЦПУ с академическими ритмами (длительностью занятий и расписанием перерывов). Несмотря на пиковые значения, система демонстрировала стабильную работу, что подтверждает корректность выбранной архитектуры и ее готовность к горизонтальному масштабированию.

Для оценки эффективности разработанного решения был проведен сравнительный анализ и сбор объективных метрик. Было измерено время, необходимое на ответ среднестатистическому студенту для получения ответа на вопрос «Какие будут пары завтра?». Полученные результаты представлены на Рисунке 4.

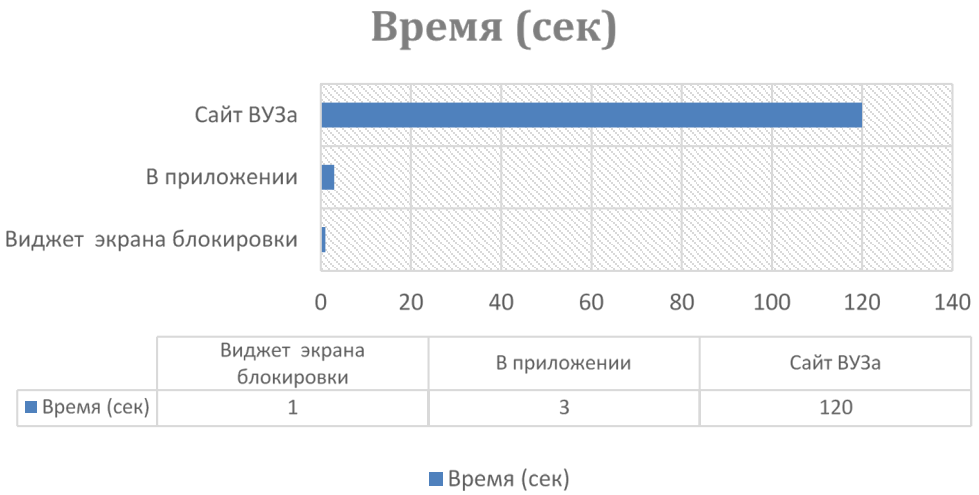

Рисунок 4 – Время получения доступа к расписанию
Figure 4 – Time per sec to schedule access

Совокупность реализованных решений позволила сократить время доступа к информации с 120 до 1–3 секунд, в зависимости от варианта использования системы.

По результатам проведенного тестирования, время доступа к расписанию с использованием разработанного приложения сократилось в среднем на 40 %. Эти показатели свидетельствуют о значительном повышении удобства и доступности информации для студентов и преподавателей.

Заключение

В статье представлено разработанное приложение, предназначенное для предоставления эффективного доступ к таким учебным данным, как расписание занятий, результатам текущего и промежуточного контроля, информации по преподавателям и учебной группе, новостному порталу. Проведенное исследование выявило ряд существенных сложностей в существующей IT-инфраструктуре, которые были устранены в предложенной архитектуре приложения и ее реализации.

Реализация проекта позволила не только устранить выявленные недостатки, но и создать основу для дальнейшей цифровой трансформации образовательного процесса. Применение современных технологий, таких как Golang, PostgreSQL, gRPC и облачных сервисов Yandex Cloud, обеспечило высокую производительность, масштабируемость, безопасность и удобство разработки приложения. Интеграция с существующими системами университета посредством gRPC-моста позволила обеспечить совместимость и эффективный обмен данными.

Результаты проведенного тестирования показали значительное улучшение показателей доступности и удобства работы с учебными данными.

Разработанный подход может быть успешно адаптирован и применен для других высших учебных заведений. В перспективе, планируется расширение функциональности приложения за счет интеграции с AI-алгоритмами для прогнозирования академических рисков и формирования индивидуальных образовательных траекторий, что позволит повысить эффективность обучения и улучшить результаты освоения образовательных программ.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Fielding R.T., Taylor R.N. Principled Design of the Modern Web Architecture. *ACM Transactions on Internet Technology*. 2002;2(2):115–150. https://doi.org/10.1145/514183.514185
2. Brazil B. *Prometheus: Up & Running*. London: O'Reilly Media, Inc.; 2018. 386 p.
3. Boettiger C. An Introduction to Docker for Reproducible Research. *ACM SIGOPS Operating Systems Review*. 2015;49(1):71–79. https://doi.org/10.1145/2723872.2723882
4. Dua R., Raja A.R., Kakadia D. Virtualization vs Containerization to Support PaaS. In: *Proceedings of the 2014 IEEE International Conference on Cloud Engineering, 11–14 March 2014, Boston, MA, USA.* IEEE; 2014. P. 610–614. https://doi.org/10.1109/IC2E.2014.41
5. Jani Ya. Unified Monitoring for Microservices: Implementing Prometheus and Grafana for Scalable Solutions. *Journal of Artificial Intelligence, Machine Learning and Data Science*. 2024;2(1):848–852. https://doi.org/10.51219/JAIMLD/yash-jani/206
6. Turcotte A., Gokhale S., Tip F. Increasing the Responsiveness of Web Applications by Introducing Lazy Loading. In: *Proceedings of the 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE), 11–15 September 2023, Luxembourg, Luxembourg.* IEEE; 2023. P. 459–470. https://doi.org/10.1109/ASE56229.2023.00192

7. Christie M.A., Bhandar A., Nakandala S., et al. Managing Authentication and Authorization in Distributed Science Gateway Middleware. *Future Generation Computer Systems*. 2020;111:780–785. https://doi.org/10.1016/j.future.2019.07.018
8. Shanmughan V.K. Keycloak Implementation for Identity Management in SASE Architectures: A Regional Hub Approach. *Sarcouncil Journal of Multidisciplinary*. 2025;5(7):1001–1007. https://doi.org/10.5281/zenodo.16498126
9. Divyabharathi D.N., Cholli N.G. A Review on Identity and Access Management Server (KeyCloak). *International Journal of Security and Privacy in Pervasive Computing*. 2020;12(3):46–53. https://doi.org/10.4018/IJSPPC.2020070104
10. Bunn C.D.S., Miers Ch.C. Evaluating Performance Impacts in Identity Management Based on Keycloak and OpenID Connect. In: *2024: Companion Proceedings of the 24th Brazilian Symposium on Information and Computational Systems Security, 16–19 September 2024, São José dos Campos/SP, Brazil.* Porto Alegre: Sociedade Brasileira de Computação; 2024. P. 197–200. https://doi.org/10.5753/sbseg_estendido.2024.243372

ИНФОРМАЦИЯ ОБ АВТОРАХ/ INFORMATION ABOUT THE AUTHORS

Абрамов Алексей Михайлович, аспирант, руководитель Академии ИТ-парка, технопарк в сфере высоких технологий «ИТ-парк», Казань, Российская Федерация.
e-mail: am.abramov1@yandex.ru

Alexey M. Abramov, Postgraduate, Head of the Academy of the IT Park, Technopark in the field of high technologies "IT Park", Kazan, the Russian Federation.

Аникин Игорь Вячеславович, доктор технических наук, профессор, заведующий кафедрой систем информационной безопасности, Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация.
e-mail: anikinigor777@mail.ru
ORCID: 0000-0001-9478-4894

Igor V. Anikin, Doctor of Engineering Sciences, Professor, Head of Information Security Systems Department, Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation.

Бурдина Алена Алексеевна, магистр, Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация.
e-mail: aliona2011ujl@yandex.ru

Alena A. Burdina, Master, Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation.

Гайфуллин Дамир Равильевич, бакалавр, программист, Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация.
e-mail: gaifullindr@gmail.com

Damir R. Gaifullin, Bachelor, Programmer, Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation.

Грудцин Андрей Андреевич, бакалавр, графический дизайнер, Университет Иннополис, Казань, Российская Федерация.
e-mail: grudtcinaa@ya.ru

Andrey A. Grudtsin, Bachelor, Graphic Designer, Innopolis University, Kazan, the Russian Federation.

Добрынин Илья Сергеевич, бакалавр, ведущий программист, Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация.
e-mail: me@dobryninilya.ru

Ilya S. Dobrynin, Bachelor, Leading Programmer, Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation.

Лупанов Богдан Игоревич, бакалавр, ведущий программист, Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация.
e-mail: bogdanlupanov@yandex.ru

Bogdan I. Lupanov, Bachelor, Leading Programmer, Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation.

Мулюков Рустам Азатович, директор центра «Цифровая кафедра», Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация.
e-mail: r.mulyukov@yandex.ru

Rustam A. Mulyukov, Director of the Digital Department Center, Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation.

Плотник Артур Эдуардович, специалист, ведущий программист, Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ, Казань, Российская Федерация.
e-mail: artur.plotnik@outlook.com

Arthur E. Plotnik, Specialist, Lead Programmer, Kazan National Research Technical University named after A.N. Tupolev-KAI, Kazan, the Russian Federation.

Статья поступила в редакцию 25.08.2025; одобрена после рецензирования 22.10.2025; принята к публикации 05.11.2025.

The article was submitted 25.08.2025; approved after reviewing 22.10.2025; accepted for publication 05.11.2025.