МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

РАЗРАБОТКА СИСТЕМЫ КОММУНИКАЦИИ МЕЖДУ СТУДЕНТАМИ И ПРЕПОДАВАТЕЛЯМИ В РАМКАХ УЧЕБНОГО ПРОЦЕССА

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 441 группы
направления 02.03.03 Математическое обеспечение и администрирование информационных систем
факультета компьютерных наук и информационных технологий
Филиппова Максима Владимировича

Научный руководитель:
ст. преп. кафедры ИиП _____ А. А. Казачкова

подпись, дата

Зав. кафедрой:
к. ф.-м. н., доцент _____ М. В. Огнева

подпись, дата

Саратов 2023

ВВЕДЕНИЕ

Актуальность темы.

Современное образование в университетах требует эффективных инструментов и систем, способных облегчить работу и взаимодействие между студентами и преподавателями, а также оптимизировать организацию учебного процесса. В свете этой потребности разработка и реализация системы помощи студентам и преподавателям становится актуальной задачей, позволяющей создать удобную и эффективную платформу для получения информации и взаимодействия в университете.

В эпоху цифровизации современная технологическая сфера предлагает множество инструментов и платформ для создания сложных и эффективных приложений. Развитие информационных технологий привело к тому, что способы общения и взаимодействия людей значительно изменились. Непрерывное совершенствование и внедрение новых технологий позволяют нам обмениваться данными и информацией в режиме реального времени. Такие технологии, как архитектура микросервисов, Docker, RabbitMQ, Spring Framework, Java, Redis и PostgreSQL, служат основой для создания таких современных программных решений, которые улучшают эффективность обработки, хранения и обмена информацией в онлайн-среде.

Ключевой особенностью разрабатываемой системы станет применение передовых технологий и подходов, обеспечивающих высокую производительность, масштабируемость и удобство использования. Решение будет ориентировано на реализацию эффективного клиент-серверного взаимодействия и использование брокеров сообщений для коммуникации.

Ожидается, что система помощи студентам и преподавателям будет способствовать улучшению качества учебного процесса, упрощению взаимодействия между его участниками и повышению эффективности образовательного процесса в университете.

Цель бакалаврской работы: проектирование и реализация веб-приложения, предназначенного для студентов и преподавателей

Саратовского государственного университета, с целью облегчить взаимодействие и организацию учебного процесса.

Поставленная цель определила следующие задачи:

- Исследование существующих систем и приложений для взаимодействия студентов и преподавателей в университете.
- Выбор и обоснование технологического стека.
- Исследование современных технологий и инструментов для разработки веб-приложений.
- Проектирование архитектуры системы.
- Реализация компонентов системы, включая получение и отображение расписания, рассылку сообщений и управление домашним заданием.
- Реализация взаимодействия с Telegram Bot API для обеспечения коммуникации через телеграм-бота.

Методологические основы.

Разработка систем коммуникации между студентами и преподавателями в рамках учебного процесса представлены в работах Bonk C. J., Давиденко П.В., Давиденко Л.М.

Практическая значимость бакалаврской работы.

Практическая значимость бакалаврской работы заключается в использовании разработанного приложения студентами и преподавателями университета для коммуникации друг с другом и получения актуального расписания занятий.

Структура и объём работы.

Бакалаврская работа состоит из введения, 8 разделов, заключения, списка использованных источников и 7 приложений. Общий объем работы – 65 страницы, из них 42 страницы – основное содержание, включая 6 рисунков и 1 таблицу, список использованных источников информации – 22 наименования.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел "Анализ аналогов" посвящен исследованию трех существующих решений, предоставляющих расписание для студентов.

В данном разделе основное внимание уделяется анализу недостатков каждого из рассмотренных аналогов, особенно в контексте отсутствия функциональности для общения со студентами. Выводы по данному разделу подчеркивают, что все текущие решения имеют свои недостатки и не предлагают полного набора необходимых функций. В свою очередь, предлагаемое решение нацелено на объединение этих функций, создание универсальной платформы для эффективного взаимодействия студентов и преподавателей, предоставляя доступ к расписанию и обмен сообщениями.

Второй раздел "Архитектура приложения" посвящен анализу и сравнению двух основных подходов к архитектуре приложений: монолитной и микросервисной.

Монолитная архитектура предполагает построение приложения как единого неделимого блока, где все компоненты сильно связаны между собой. Микросервисная архитектура, напротив, предполагает разделение приложения на отдельные сервисы, каждый из которых выполняет свою конкретную функцию. В процессе анализа были рассмотрены преимущества и недостатки каждого подхода. В конечном итоге, для реализации приложения была выбрана микросервисная архитектура. Этот выбор был обусловлен несколькими критически важными факторами.

Масштабируемость является ключевым критерием: с учетом потенциального роста числа пользователей и функциональности системы, микросервисная архитектура позволяет легко адаптироваться к этим изменениям. Устойчивость – другой важный аспект: создание системы, которая останется работоспособной даже при возникновении сбоев в отдельных сервисах, становится возможным благодаря микросервисной

архитектуре. Технологическая гетерогенность также имеет значение: микросервисная архитектура дает возможность использовать различные технологии и языки программирования в рамках разных сервисов. И, наконец, разделение обязанностей обеспечивает более простой и понятный процесс разработки и поддержки.

С учетом всех этих факторов, микросервисная архитектура была признана наиболее подходящим выбором для данного проекта.

Третий раздел "Фреймворки для разработки на Java" посвящен подробному анализу Spring Framework, его основных принципов и компонентов.

Рассмотрены концепции внедрения зависимостей и инверсии управления, которые лежат в основе этого фреймворка и определяют его гибкость и мощность. Кроме того, дан раздел уделён подробному изучению Spring Boot и Spring Data, которые предоставляют упрощённые способы создания стандартных приложений и взаимодействия с базами данных. Основными преимуществами этих фреймворков, указанными в разделе, являются инверсия управления, внедрение зависимостей, упрощенная конфигурация и управление зависимостями, а также удобный доступ к данным через Spring Data.

Выводы по третьему разделу сводятся к решению использовать Spring Framework и Spring Boot для разработки веб-приложения. Этот выбор обусловлен тем, что данные фреймворки предоставляют мощные инструменты и преимущества, позволяющие эффективно реализовать поставленные функциональные задачи, а также обеспечивают гибкость и масштабируемость разрабатываемого приложения.

Четвертый раздел "Межсервисное взаимодействие" посвящен анализу способов взаимодействия между сервисами в приложении с микросервисной архитектурой.

Основное внимание уделено Apache Kafka и RabbitMQ, включая их сравнение, а также выявление преимуществ и недостатков каждого из этих инструментов. Кроме того, раздел описывает такой подход к межсервисному взаимодействию, как RPC (вызов удаленной процедуры), активно применяемый в RabbitMQ.

Выводы по четвертому разделу подчеркивают, что использование RPC с RabbitMQ обладает высокой производительностью и надежностью. Эти качества делают его предпочтительным механизмом взаимодействия для реализации межсервисной коммуникации в микросервисной архитектуре.

Пятый раздел "Интерфейс Telegram Bot API" посвящен взаимодействию с Telegram Bot API.

В этом разделе изначально рассматривается общая концепция API, затем детально изучаются возможности, которые предоставляет API Telegram Bot. В частности, описываются и сравниваются два способа взаимодействия: Web Hooks и Long Polling.

Выводы по пятому разделу подчеркивают, что с учетом рассмотренных факторов, выбор метода Long Polling для взаимодействия с серверами Telegram является обоснованным. Этот выбор позволяет достичь необходимой функциональности приложения.

Шестой раздел "Хранение данных" посвящен механизмам хранения данных в приложении.

В данном разделе подробно изучается система управления базами данных PostgreSQL, предназначенная для обработки широкого спектра задач, включая хранение, обновление и извлечение данных. Также исследуется и описывается инструмент Redis, который представляет собой систему управления базами данных, основанную на хранении данных в памяти и предназначенную для кэширования.

Выводы по шестому разделу указывают, что PostgreSQL была выбрана в качестве СУБД для данного приложения из-за ее преимуществ, включая гибкость, надежность и возможности масштабирования, которые обеспечивают эффективное хранение и управление данными. Кроме того, Redis используется для эффективного кэширования запросов на получение расписания с сайта университета, что позволяет снизить общее количество запросов и увеличить скорость предоставления данных.

Седьмой раздел "Дополнительные инструменты и технологии" посвящен обзору дополнительных инструментов, используемых в разработке приложения.

В разделе рассматриваются технологии Docker и Docker Compose, которые используются для контейнеризации приложений. Docker - это инструмент, который позволяет разработчикам создавать, развертывать и запускать приложения внутри контейнеров, обеспечивая их изоляцию от окружающей системы. Docker Compose - это инструмент, позволяющий определить и управлять многоконтейнерными приложениями Docker. Также обсуждаются способы извлечения данных с веб-страниц и API Tracto, используемого для взаимодействия с веб-сайтом университета для получения расписания.

Вывод по этому разделу подчеркивает, что использование Docker и Docker Compose в микросервисной архитектуре упрощает процесс разработки, тестирования и развертывания приложений. В то же время, несмотря на различные доступные библиотеки для извлечения данных с веб-страниц, в данном проекте выбран сервис Tracto. Этот выбор обусловлен удобством использования API Tracto для доступа к актуальному расписанию, его RESTful структурой, возможностью легко и быстро извлекать информацию, использованием формата JSON для обмена данными, полным функционалом для получения данных о расписании, а также бесплатностью и отсутствием необходимости регистрации.

Восьмой раздел "Разработка приложения" посвящен полному циклу разработки приложения, начиная от архитектуры и заканчивая демонстрацией его работы.

В этом разделе подробно описаны все этапы разработки, включая взаимодействие с Telegram Bot API, реализацию сервисов обработки данных пользователей, обработки сообщений и получения расписания, а также процесс контейнеризации приложения с помощью Docker. В конечном итоге, было создано веб-приложение, направленное на помощь студентам и преподавателям университета в их взаимодействии. Разработанная система обеспечивает следующие функциональности:

- Получение расписания студентов и преподавателей на текущий день или на текущую неделю.
- Возможность для преподавателей рассылать сообщения группам студентов.

Технологический стек приложения включает в себя Java и фреймворк Spring Boot, микросервисную архитектуру, асинхронное взаимодействие между клиентом и сервером через RabbitMQ, хранение данных в PostgreSQL, кэширование с помощью Redis и использование Telegram Bot API для взаимодействия с пользователями.

ЗАКЛЮЧЕНИЕ

В ходе данной дипломной работы было разработано веб-приложение для студентов и преподавателей университета. Целью работы было создание системы, которая помогала бы студентам и преподавателям эффективно взаимодействовать друг с другом.

Для достижения данной цели были реализованы следующие функциональности в приложении:

1. Получение расписания студентов и преподавателей на день а также возможность получения расписания на текущую неделю.
2. Возможность рассылки сообщений преподавателями группам студентов.

Для реализации приложения был выбран технологический стек, включающий Java с использованием фреймворка Spring Boot, микросервисную архитектуру, асинхронное клиент-серверное взаимодействие с помощью RabbitMQ, хранение данных в базе данных PostgreSQL, кэширование с помощью Redis, а также использование Telegram Bot API для интерфейса взаимодействия с пользователями.

В результате проведенной работы были достигнуты поставленные цели. Разработанное веб-приложение предоставляет студентам и преподавателям удобный инструмент для взаимодействия и обмена информацией. Студенты могут легко получать актуальное расписание и домашнее задание. Преподаватели же могут удобно осуществлять рассылку сообщений и домашнего задания студентам.

Таким образом, разработанное приложение представляет собой полезный инструмент для улучшения коммуникации и взаимодействия между студентами и преподавателями в университете.

Основные источники информации:

1. Bonk, C. J. "The Wiley Learning Technology Series: The Rise of Educational Technology as a Sociocultural and Ideological Phenomenon" / Curtis J. Bonk – Wiley-Blackwell, 2021. - 320 с.
2. Давиденко П.В., Давиденко Л.М. ИНФОРМАТИЗАЦИЯ ПРОЦЕССА ОБУЧЕНИЯ: ИССЛЕДОВАНИЕ LMS-СИСТЕМ // Grand Altai Research & Education. 2021. №2.
3. Richards, M. Software Architecture Patterns / Mark Richards – O'Reilly Media, 2018. - 210 с.
4. Newman, S. Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith / Sam Newman – O'Reilly Media, 2019. - 347 с.
5. Stenzel, K. Asynchronous Programming in Java / Klaus Stenzel – Apress, 2020. - 281 с.
6. Volkenstein, H. PostgreSQL 13 High Performance / Hans-Jürgen Schönig, Enrico Pirozzi – Packt Publishing, 2021. - 798 с.
7. Carlson, D. Redis in Action / David Carlson – Manning Publications, 2021. - 320 с.
8. Miell, I., Docker in Practice / Ian Miell, Aidan Hobson Sayers – Manning Publications, 2019. - 384 с.
9. Walls, C. Spring in Action / Craig Walls – Manning Publications, 2020. - 520 с.