Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Операционные среды и системное программирование»

|  |  |
| --- | --- |
|  | *К защите допустить:* |
|  | И. о. заведующего кафедрой информатики  *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* С. И. Сиротко |
|  |  |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

**ПРОГРАММНОЕ СРЕДСТВО ДЛЯ КОММУНИКАЦИИ ПОЛЬЗОВАТЕЛЕЙ ПРИНАДЛЕЖАЩИХ ФУТБОЛЬНОМУ СООБЩЕСТВУ**

БГУИР КП 1-40 04 01 026 ПЗ

Студент             П. С. Павлюткин

Руководитель             А. В. Давыдчик

Нормоконтролер             А. В. Давыдчик

Минск 2024

# СОДЕРЖАНИЕ

[Введение…………………………………………………………………………5](#_Toc166070974)

[1 Архитектура программного обеспечения………..………………………….7](#_Toc166070975)

[2 Платформа программного обеспечения 13](#_Toc166070976)

[2.1 Структура и архитектура платформы 13](#_Toc166070977)

[2.2 История и достоинства платформы программного обеспечения 15](#_Toc166070978)

[2.3 Обоснование выбора платформы программного обеспечения 16](#_Toc166070979)

2.4 Среда разработки, ее особенности и достоинства ……………………17

2.5. Анализ базы данных

[3 Теоретическое обоснование разработки программного продукта..............19](#_Toc166070980)

3.1 [Обоснование необходимости разработки……………………………...19](#_Toc166070981)

[4 Проектирование функциональных возможностей программы…………...23](#_Toc166070982)

* 1. [Обоснование и описание функций программного обеспечения……...23](#_Toc166070983)

4.2 Описание функциональных возможностей программы………………24

4.3 Выполнение программы……………………………….………………..26

4.4 Возможные улучшения функционала………………………………….28

5 Проектирование разрабатываемой базы данных программного обеспечения ………………………………...................................................30

[5.1 Общая структура программы…………………………………………..](#_Toc152750694).30

[5.2 Функциональная схема программы.……………………………………](#_Toc152750699)32

[5.3 Блок-схема алгоритма программы.……………………………………..33](#_Toc152750700)

[Заключение.……….](#_Toc152750702).............................................................................................34

[Список используемых источников](#_Toc152750703).…………………………………………...35

[Приложение А (обязательное) Исходный код программы.………………….37](#_Toc152750704)

[Приложение Б (обязательное) Функциональная схема алгоритма, реализующего программное средство](#_Toc152750705)………………………47

[Приложение В (обязательное) Ведомость документов](#_Toc152750708)……………………...48

# ВВЕДЕНИЕ

В современном мире коммуникация играет ключевую роль во многих аспектах жизни, будь то личные отношения, рабочие взаимодействия или общественная деятельность. С развитием информационных технологий и распространением интернет-сервисов, средства для общения пользователей становятся все более разнообразными и сложными. Программные средства для коммуникации пользователей позволяют объединить людей в рамках разнообразных сообществ и форумов, независимо от их географического положения, и предоставляют возможности для обмена информацией в реальном времени. Эти инструменты становятся неотъемлемой частью повседневной жизни, влияя на то, как мы взаимодействуем друг с другом, как мы ведем бизнес и как организуем наши социальные связи.

Актуальность разработки программного средства для коммуникации пользователей обусловлена ростом потребностей в эффективных и надежных инструментах для общения. Сегодня, когда многие сферы деятельности переходят в онлайн-формат, требуется программное обеспечение, которое не только обеспечивает стабильную и безопасную связь, но и предоставляет широкий функционал для совместной работы, обмена файлами и организации видеоконференций. Важность таких средств особенно возросла в условиях глобальных вызовов, таких как пандемия, которая подтолкнула многие компании и организации к удаленному режиму работы.

Целью данной работы является разработка программного средства, которое обеспечит пользователей возможностью эффективной и безопасной коммуникации. В рамках работы будут рассмотрены современные технологии и подходы, используемые для создания таких средств. Проектирование и реализация программного средства включат анализ требований пользователей, выбор оптимальной архитектуры и технологий, а также разработку и тестирование функциональных модулей. Итогом работы станет готовое программное средство, которое сможет удовлетворить потребности пользователей в рамках футбольного сообщества в современных средствах коммуникации.

# АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

**1.1 Структура программного обеспечения**

Архитектура клиент-серверных приложений является одним из наиболее распространенных и проверенных временем подходов к построению распределенных систем. В основе клиент-серверной архитектуры лежит разделение задач между клиентской частью (Frontend) и серверной частью (Backend), что позволяет оптимально распределять нагрузку и обеспечивать высокий уровень производительности и безопасности. Схема клиент-серверной архитектуры представлено на рисунке 1.1.

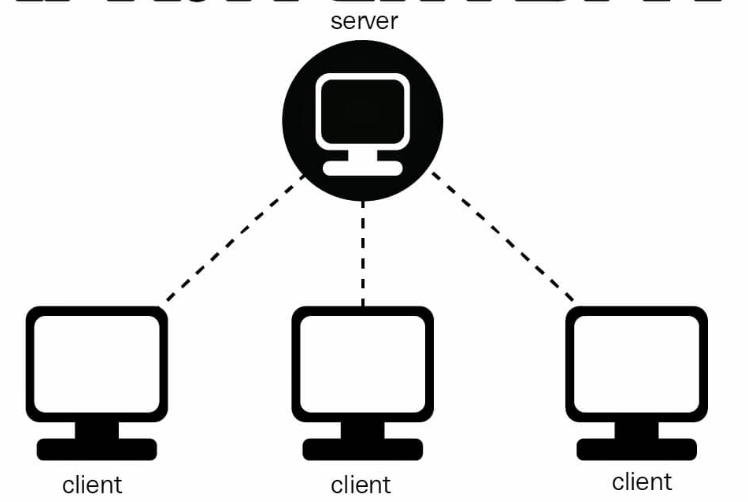


Рисунок 1.2 – Архитектура клиент-серверных приложений

Клиент-серверная архитектура — это достаточно большая сеть, которая может состоять из неограниченного числа клиентов и одного или нескольких серверов, при этом неважно, по какой сети происходит коммуникация: по глобальной или по локальной. Клиент-серверное взаимодействие происходит даже тогда, когда пользователи отправляют друг другу электронные письма или общаются через мессенджер. Даже в этих случаях сообщение или письмо вначале отправляется на сервер, где оно проходит небольшую обработку, а потом — получателю. При этом сервер может сохранить отправленный файл, чтобы отправитель и получатель всегда имели к нему доступ: скачали, отредактировали или удалили.

В глобальном смысле весь интернет — это один большой сервер, а в роли клиента выступает каждое устройство, выходящее в сеть.

Большинство обычных пользователей даже не догадываются, что благодаря архитектуре клиент-сервер обслуживается любой их запрос в поисковой системе. О каждом своем клиенте серверы хранят определенную информацию, создавая обезличенный облик своего клиента. Некоторые особенности:

* основная работа при такой архитектуре лежит на мощных серверах, а не на клиенте, что снижает нагрузку на последнего;
* основной объем данных хранится на серверах, так как они лучше защищены, чем многие клиенты;
* клиент-сервер — это общая архитектура отношений, где уровни отношений регулируются протоколами, что дает возможность разграничивать уровни доступа клиентов к серверам;
* с сервером может работать любое устройство, вне зависимости от его операционной системы;
* все команды от клиента обрабатываются сервером, что снижает нагрузку на саму сеть;
* важно сохранять работоспособность именно серверов, так как их выход из строя грозит отсутствием работоспособности многих клиентов;
* легкая масштабируемость, то есть при увеличении числа клиентов расширить количество серверов, которые их обслуживают, — не большая проблема.

**1.2 Клиентская часть**

Клиентская часть реализуется с использованием HTML, CSS и JavaScript для создания веб-интерфейса. Современные фреймворки, такие как React или Angular, используются для создания динамичных и отзывчивых веб-приложений. Эти фреймворки позволяют легко создавать сложные пользовательские интерфейсы, обеспечивая высокий уровень взаимодействия и удобства.

Клиентская часть отправляет запросы на сервер, отображает результаты, полученные от сервера, и обеспечивает взаимодействие пользователя с системой. Это включает в себя отображение страниц, форм для ввода данных, кнопок и других элементов интерфейса.

**1.3 Серверная часть**

Серверная часть реализована на Java с использованием фреймворка Spring Boot. Spring Boot предоставляет модульную архитектуру, автоматическую конфигурацию и встроенные инструменты для разработки и развертывания приложений.

Серверная часть обрабатывает запросы от клиента, выполняет бизнес-логику, взаимодействует с базой данных и возвращает результаты клиенту. Важными компонентами серверной части являются контроллеры, сервисы и репозитории.

**1.4 База данных, используемая в приложении**

В качестве базы данных для нашего проекта используется реляционная база данных PostgreSQL. PostgreSQL известна своей высокой производительностью и надежностью, что делает ее отличным выбором для хранения и управления большими объемами данных. Она поддерживает сложные запросы и транзакции, обеспечивая целостность и согласованность данных. PostgreSQL предоставляет широкий спектр возможностей, включая поддержку индексов, хранимых процедур, триггеров и полнотекстового поиска, что позволяет эффективно обрабатывать различные запросы пользователей.

База данных используется для хранения информации о пользователях, их профилях, списках контактов, группах общения, а также истории сообщений. Это позволяет обеспечить оперативное получение необходимой информации и повышает производительность системы. При этом данные хранятся в таблицах, каждая из которых отвечает за определенный аспект системы, что упрощает управление данными и их обновление. Важно отметить, что база данных также обеспечивает высокую степень безопасности данных благодаря встроенным механизмам шифрования и управления доступом.

**1.5 Безопасность приложения**

Безопасность приложения обеспечивается с использованием Spring Security, мощного фреймворка для реализации аутентификации и авторизации в приложениях на базе Spring. Spring Security поддерживает различные методы аутентификации, такие как базовая аутентификация, форм-логин, OAuth2 и JSON Web Token (JWT). Использование JWT позволяет безопасно передавать данные между клиентом и сервером, обеспечивая целостность и подлинность информации.

Контроль доступа к различным функциям системы осуществляется путем назначения ролей и прав пользователям. Это предотвращает несанкционированный доступ к важным данным и функциям приложения. Для защиты данных при передаче используются протоколы шифрования, такие как HTTPS, что обеспечивает безопасность информации на всех этапах взаимодействия. Кроме того, Spring Security предоставляет инструменты для защиты от атак типа CSRF (Cross-Site Request Forgery) и XSS (Cross-Site Scripting), что делает приложение устойчивым к различным угрозам безопасности.

**1.6 Операционная система**

Система разрабатывается и развертывается на операционной системе Windows. Windows предоставляет стабильную и надежную платформу для выполнения всех компонентов системы. Операционная система обеспечивает совместимость с широким спектром корпоративного программного обеспечения и инфраструктуры, что упрощает интеграцию нашего приложения в существующие ИТ-системы организаций.

Кроме того, Windows обеспечивает удобные инструменты для управления ресурсами, мониторинга и поддержки инфраструктуры. В частности, это включает средства для управления сетью, хранения данных и обеспечения безопасности. Windows также поддерживает контейнеризацию с помощью Docker, что позволяет легко развертывать и масштабировать приложение. Это особенно важно для обеспечения высокой доступности и отказоустойчивости системы. Благодаря поддержке различных инструментов разработки и развертывания процесс разработки и развертывания приложения становится более эффективным и организованным.

# ПЛАТФОРМА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

**2.1 Структура и архитектура платформы**

[Платформа программного обеспечения — это среда, в которой работают](https://bing.com/search?q=%d0%bf%d0%bb%d0%b0%d1%82%d1%84%d0%be%d1%80%d0%bc%d0%b0%2B%d0%bf%d1%80%d0%be%d0%b3%d1%80%d0%b0%d0%bc%d0%bc%d0%bd%d0%be%d0%b3%d0%be%2B%d0%be%d0%b1%d0%b5%d1%81%d0%bf%d0%b5%d1%87%d0%b5%d0%bd%d0%b8%d1%8f) [приложения](https://bing.com/search?q=%d0%bf%d0%bb%d0%b0%d1%82%d1%84%d0%be%d1%80%d0%bc%d0%b0%2B%d0%bf%d1%80%d0%be%d0%b3%d1%80%d0%b0%d0%bc%d0%bc%d0%bd%d0%be%d0%b3%d0%be%2B%d0%be%d0%b1%d0%b5%d1%81%d0%bf%d0%b5%d1%87%d0%b5%d0%bd%d0%b8%d1%8f). [Она включает в себя операционную систему, которая управляет](https://bing.com/search?q=%d0%bf%d0%bb%d0%b0%d1%82%d1%84%d0%be%d1%80%d0%bc%d0%b0%2B%d0%bf%d1%80%d0%be%d0%b3%d1%80%d0%b0%d0%bc%d0%bc%d0%bd%d0%be%d0%b3%d0%be%2B%d0%be%d0%b1%d0%b5%d1%81%d0%bf%d0%b5%d1%87%d0%b5%d0%bd%d0%b8%d1%8f) [аппаратными ресурсами компьютера, а также программные инструменты и](https://bing.com/search?q=%d0%bf%d0%bb%d0%b0%d1%82%d1%84%d0%be%d1%80%d0%bc%d0%b0%2B%d0%bf%d1%80%d0%be%d0%b3%d1%80%d0%b0%d0%bc%d0%bc%d0%bd%d0%be%d0%b3%d0%be%2B%d0%be%d0%b1%d0%b5%d1%81%d0%bf%d0%b5%d1%87%d0%b5%d0%bd%d0%b8%d1%8f) [библиотеки, которые используются для разработки приложений](https://bing.com/search?q=%d0%bf%d0%bb%d0%b0%d1%82%d1%84%d0%be%d1%80%d0%bc%d0%b0%2B%d0%bf%d1%80%d0%be%d0%b3%d1%80%d0%b0%d0%bc%d0%bc%d0%bd%d0%be%d0%b3%d0%be%2B%d0%be%d0%b1%d0%b5%d1%81%d0%bf%d0%b5%d1%87%d0%b5%d0%bd%d0%b8%d1%8f). Для разработки данного проекта была выбрана ОС Windows. В качестве платформы программного обеспечения для программного средства для коммуникации пользователей, принадлежащих футбольному сообществу, были выбраны Java, Spring и Spring Boot. Этот выбор обусловлен их мощностью, гибкостью и широкими возможностями для создания надежных и масштабируемых веб-приложений.

Java была выбрана из-за своей надежности, платформенной независимости и возможности обработки больших объемов данных. Spring и Spring Boot дополняют Java, предоставляя мощный фреймворк для разработки корпоративных приложений с минимальными настройками. Вот основные причины выбора этих технологий:

1. Простота использования и читаемость кода: Java имеет строгий и понятный синтаксис, который делает код легким для понимания и поддержки. Это особенно важно для разработки приложения, в котором необходимо обеспечить высокую производительность и надежность.
2. Богатая экосистема библиотек: Spring и Spring Boot предлагают множество встроенных инструментов и библиотек, которые упрощают разработку и расширение функциональности приложения. Это включает в себя средства для работы с базами данных, реализации безопасности, управления транзакциями и многое другое.
3. Переносимость: Java-приложения могут работать на различных операционных системах, включая Windows, macOS и Linux, без необходимости внесения значительных изменений в код. Это обеспечивает гибкость и удобство при развертывании приложения.
4. Поддержка различных платформ: Spring Boot поддерживает разработку как для настольных систем, так и для веб-приложений, что позволяет расширить функциональность нашей утилиты и интегрировать её с другими приложениями и сервисами.
5. Активное сообщество и обширная документация: Java и Spring имеют активное сообщество разработчиков и обширную документацию, что облегчает решение проблем и получение поддержки.

**2.2 История, версии и достоинства**

Клиент-серверная архитектура, на которой построено наше приложение, имеет долгую историю и зарекомендовала себя как надежное и эффективное решение для разработки сложных систем. Клиент-серверная модель была разработана в 1980-х годах и с тех пор стала стандартом де-факто для построения распределенных систем.

Java была представлена в середине 90-х годов и быстро стала одним из самых популярных языков программирования благодаря своей платформенной независимости, безопасности и широкому сообществу разработчиков. С тех пор Java претерпела множество изменений и обновлений, сохраняя свою актуальность и востребованность в мире разработки ПО.

Spring Framework, впервые представленный в 2002 году, стал стандартом для разработки корпоративных приложений на Java. Spring Boot, выпущенный в 2014 году, значительно упрощает создание и развертывание Spring-приложений благодаря встроенным настройкам и автоматической конфигурации.

Основные достоинства клиент-серверной архитектуры включают:

1. Разделение обязанностей: Клиентская часть отвечает за пользовательский интерфейс, а серверная часть — за обработку данных и выполнение бизнес-логики, что облегчает разработку и поддержку системы.
2. Масштабируемость: Серверная часть может быть масштабирована в ответ на увеличивающуюся нагрузку, обеспечивая высокую производительность системы.
3. Безопасность: Центральное управление доступом и защитой данных на сервере позволяет лучше контролировать безопасность.
4. Управление и поддержка: Централизованное хранение данных и логики упрощает администрирование и обновление системы.

**2.3 Обоснование выбора платформы**

Выбор Java и Spring Boot для реализации данного проекта был обусловлен несколькими ключевыми факторами:

1. Надежность и производительность: Java обеспечивает высокую производительность и стабильность приложений, что критично для систем с большим количеством пользователей.
2. Масштабируемость: Spring Boot поддерживает микросервисную архитектуру, что позволяет легко масштабировать приложение и добавлять новые функции.
3. Безопасность: Spring Security обеспечивает мощные механизмы для аутентификации и авторизации, включая поддержку OAuth2 и JWT, что обеспечивает надежную защиту данных.
4. Разработческая среда: Широкий выбор инструментов для разработки, тестирования и развертывания Java-приложений, таких как IntelliJ IDEA, Eclipse, Maven и Gradle.

**2.4 Среда разработки, ее особенности и достоинства**

IntelliJ IDEA – это мощная и универсальная интегрированная среда разработки (IDE), разработанная компанией JetBrains. Она доступна для Windows, Linux и macOS, что делает ее идеальным инструментом для кроссплатформенной разработки. IntelliJ IDEA предназначена для разработки на различных языках программирования, но особенно популярна среди Java-разработчиков благодаря своим обширным возможностям и удобству использования.

IntelliJ IDEA поддерживает подсветку синтаксиса для множества языков программирования, что облегчает чтение и понимание кода. Функция автодополнения кода IntelliSense (в IntelliJ IDEA аналогичная функция называется Code Completion) предлагает варианты завершения кода на основе контекста, что ускоряет написание кода и уменьшает количество ошибок. Кроме того, IntelliJ IDEA автоматически проверяет синтаксис и выделяет ошибки в реальном времени, что помогает разработчикам быстро находить и исправлять проблемы.

Встроенная поддержка системы контроля версий Git позволяет разработчикам легко управлять версиями кода, создавать ветки, сливать изменения и отслеживать историю изменений. IntelliJ IDEA также поддерживает другие системы контроля версий, такие как SVN и Mercurial, что делает ее универсальным инструментом для управления проектами.

Инструменты профилирования и отладки в IntelliJ IDEA помогают разработчикам анализировать производительность кода, находить узкие места и оптимизировать приложения. Среда разработки предоставляет возможность пошаговой отладки, установки точек останова, анализа стеков вызовов и просмотра переменных, что упрощает процесс отладки и улучшает качество кода.

IntelliJ IDEA имеет обширный маркетплейс с множеством плагинов, которые добавляют новые функции и возможности. Это позволяет разработчикам настроить среду разработки под свои нужды, добавляя поддержку дополнительных языков программирования, фреймворков и инструментов. IntelliJ IDEA поддерживает широкий спектр языков программирования, включая Java, Kotlin, Python, JavaScript, TypeScript и многие другие. Это делает ее универсальным инструментом для разработки различных типов приложений.

Среда разработки работает на всех популярных операционных системах, что позволяет разработчикам создавать приложения для разных платформ. Установка IntelliJ IDEA проста и возможна на всех популярных операционных системах. IntelliJ IDEA – это мощная и быстрая IDE, которая подходит для работы даже на не очень мощных компьютерах. Среда разработки имеет простой и интуитивно понятный интерфейс, что делает ее доступной для начинающих разработчиков. IntelliJ IDEA обладает всеми необходимыми функциями для профессиональной разработки, что делает ее полезным инструментом для опытных программистов. Среда разработки можно настроить под свои нужды с помощью плагинов и настроек, что позволяет оптимизировать процесс разработки и повысить производительность.

Интерфейс программы приведен на рисунке 2.1.

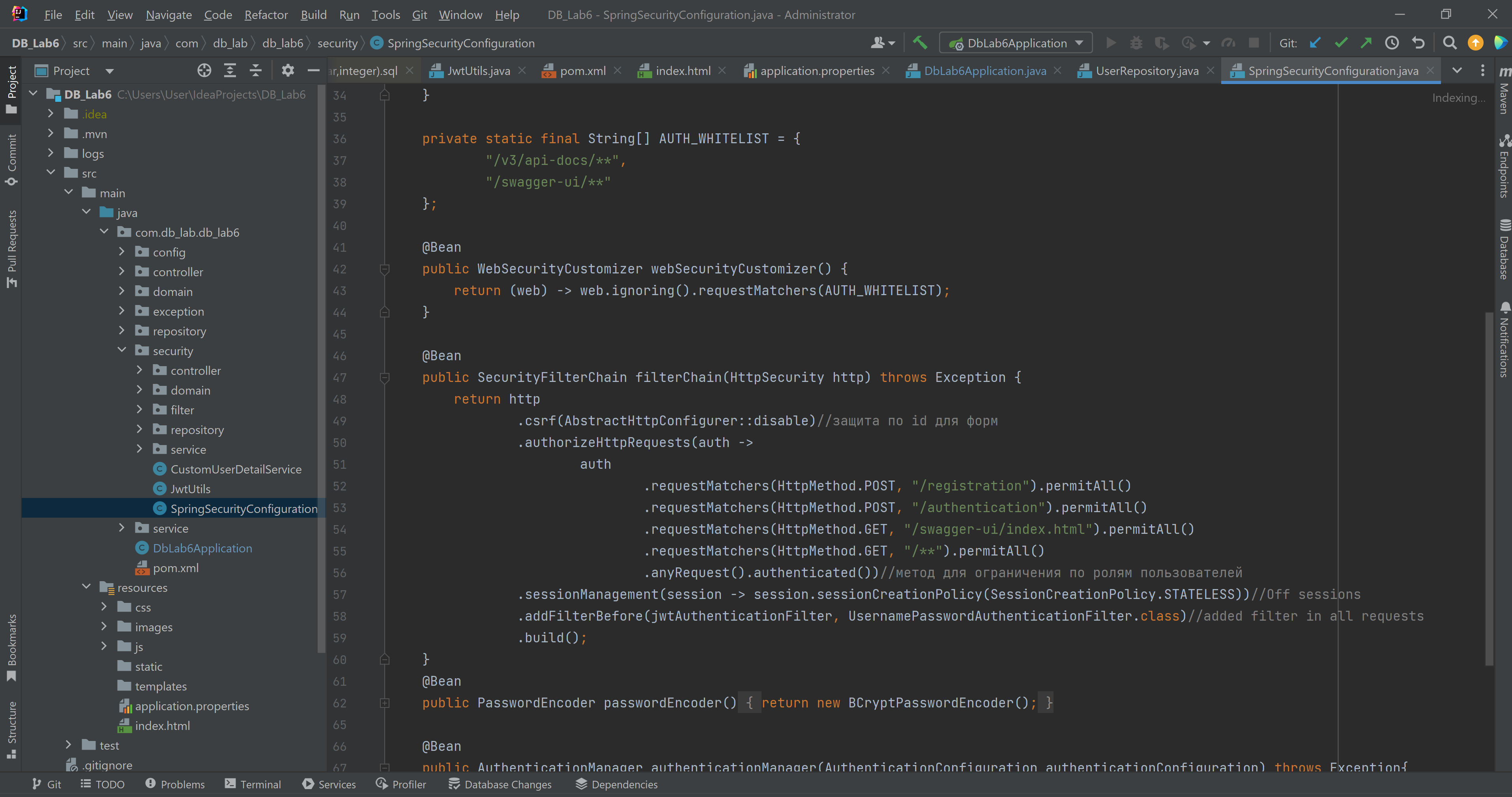


Рисунок 2.3 – Интерфейс редактора Visual Studio Code

Выбранная платформа программного обеспечения использует очень популярные и проверенные временем качественные решения.

**2.5 Анализ базы данных для написания программы**

В рамках данного программного обеспечения была выбрана реляционная база данных PostgreSQL. Этот выбор обусловлен несколькими важными факторами:

1. Надежность и безопасность данных: PostgreSQL поддерживает транзакции, соответствующие стандарту ACID, что гарантирует сохранение данных даже в случае сбоев. Это особенно важно при работе с финансовыми данными и заказами.
2. Масштабируемость и производительность: База данных PostgreSQL позволяет обрабатывать параллельные запросы и включает инструменты для оптимизации работы с большими объемами данных. Это особенно актуально при увеличении базы клиентов и заказов.
3. Мощный SQL-синтаксис и поддержка расширений: PostgreSQL поддерживает выполнение сложных запросов и может быть дополнена различными расширениями, которые позволяют интегрировать её с другими системами и добавлять специфические функции, такие как аналитика и машинное обучение.
4. Сообщество и документация: PostgreSQL — популярная СУБД, обладающая обширной документацией и активным сообществом, что облегчает получение помощи и консультаций.
5. Поддержка JSON и гибридных структур данных: PostgreSQL позволяет работать с JSON-данными, что делает её гибридной реляционно-документной СУБД. Это полезно для гибкого хранения данных, таких как информация о продуктах с уникальными атрибутами или история изменений в заказах.
6. Расширенная поддержка географических данных: PostgreSQL включает расширение PostGIS, которое добавляет возможность работы с географическими данными (координаты, маршруты и т.д.). Это особенно полезно для приложений, связанных с логистикой и доставкой, позволяя рассчитывать расстояния и зоны обслуживания.
7. Контроль версий данных: PostgreSQL поддерживает функции для отслеживания версий записей. Например, расширение temporal tables позволяет сохранять историю изменений записей, что важно для аудита и отслеживания изменений в заказах, ценах и запасах.
8. Асинхронная репликация и отказоустойчивость: PostgreSQL поддерживает асинхронную репликацию данных, что позволяет создавать резервные копии базы в реальном времени и балансировать нагрузку на серверы. Это повышает надежность системы и обеспечивает быстрый переход на резервный сервер в случае сбоя основного.
9. Поддержка хранимых процедур и триггеров: Хранимые процедуры и триггеры позволяют автоматизировать процессы на уровне базы данных, такие как проверка наличия товара на складе перед заказом, расчет скидок и начисление бонусов. Это снижает нагрузку на приложение, размещая логику бизнеса на стороне базы данных.
10. Защита на уровне строк и колонок: PostgreSQL предоставляет возможность настроить доступ к данным на уровне строк и колонок, что важно для защиты конфиденциальной информации, такой как финансовые данные или личные данные клиентов. Это позволяет разграничивать доступ в зависимости от ролей пользователей.
11. Параллельные индексы и оптимизация запросов: PostgreSQL позволяет создавать индексы параллельно и оптимизировать выполнение запросов, что снижает время отклика при сложных операциях. Для системы с растущей базой данных это помогает улучшить производительность и минимизировать задержки при работе с данными клиентов, продуктов и заказов.

Анализ базы данных PostgreSQL показывает, что она обладает широкими возможностями и функциями для разработки программного обеспечения. PostgreSQL обеспечивает интеграцию с различными инструментами разработки и поддержку разнообразного аппаратного обеспечения, что делает её отличным выбором для создания надежного и эффективного программного обеспечения для коммуникации пользователей, принадлежащих футбольному сообществу.

**3 ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ   
            ПРОГРАММНОГО ПРОДУКТА**

**3.1 Обоснование необходимости разработки**

В современном мире футбольное сообщество включает миллионы участников, среди которых болельщики, игроки, тренеры и организаторы. Эффективная коммуникация между всеми этими участниками является ключевым фактором для успешной работы и координации мероприятий. С развитием цифровых технологий возникла необходимость создания программного средства, которое бы обеспечивало удобные и функциональные инструменты для общения внутри футбольного сообщества.

Традиционные средства коммуникации, такие как электронная почта или мессенджеры, не всегда удовлетворяют специфическим потребностям футбольных сообществ. Требуется решение, которое позволит эффективно организовывать матчи, тренировки, обмениваться важной информацией и сохранять историю взаимодействий. Программное средство, разработанное специально для футбольного сообщества, способно решить эти проблемы и повысить эффективность взаимодействия всех участников.

Кроме того, создание такого программного средства позволяет улучшить организацию событий, уменьшить количество ошибок и недоразумений, а также предоставить удобные инструменты для анализа и оценки деятельности. Это позволит футбольным клубам и сообществам более эффективно управлять своими ресурсами и достигать лучших результатов.

**3.2 Технологии программирования, используемые для решения поставленных задач**

Для разработки программного средства для коммуникации пользователей, принадлежащих футбольному сообществу, были выбраны следующие технологии:

Язык программирования Java, обладающий высокой производительностью, надежностью и кроссплатформенностью. Позволяет создавать масштабируемые и эффективные приложения, поддерживает большое количество библиотек и фреймворков для упрощения разработки.

Spring – фреймворк для разработки корпоративных приложений на языке Java. Обеспечивает гибкость, модульность и простоту конфигурации приложения. Включает в себя множество модулей для работы с данными, безопасности, веб-приложений и интеграции с внешними системами.

Spring Boot – расширение Spring, предназначенное для упрощения процесса создания и развертывания приложений. Обеспечивает автоматическую конфигурацию, встроенные средства мониторинга и управления, а также поддержку микросервисной архитектуры.

PostgreSQL – реляционная база данных с открытым исходным кодом. Высокая производительность, надежность и поддержка сложных запросов и транзакций. Возможность работы с JSON-данными и расширенная поддержка географических данных.

**3.3 Связь архитектуры базы данных с разрабатываемым программным обеспечением**

Архитектура базы данных играет ключевую роль в разработке программного средства для коммуникации пользователей футбольного сообщества. Выбор PostgreSQL как СУБД обусловлен следующими факторами:

1. Сохранность и целостность данных: PostgreSQL обеспечивает транзакции, соответствующие стандарту ACID, что гарантирует сохранность данных даже в случае сбоев. Это особенно важно для хранения информации о пользователях, сообщениях и других важных данных.
2. Масштабируемость и производительность: PostgreSQL позволяет обрабатывать параллельные запросы и включает инструменты для оптимизации работы с большими объемами данных. Это обеспечивает высокую производительность системы при увеличении количества пользователей и объема данных.
3. Поддержка JSON и гибридных структур данных: PostgreSQL позволяет работать с JSON-данными, что делает её гибридной реляционно-документной СУБД. Это полезно для хранения информации о пользователях, профилях и настройках, а также для сохранения истории взаимодействий.
4. Расширенная поддержка географических данных: PostgreSQL включает расширение PostGIS, которое позволяет работать с географическими данными. Это может быть полезно для организации матчей и тренировок, позволяя рассчитывать расстояния и зоны обслуживания

**4 ПРОЕКТИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ   
            ВОЗМОЖНОСТЕЙ ПРОГРАММЫ**

**4.1 Обоснование и описание функций программного обеспечения**

Программное средство для коммуникации пользователей, принадлежащих футбольному сообществу, разработано для удовлетворения потребностей в эффективном взаимодействии, обмене информацией и управлении контентом внутри сообщества. Ниже приведены обоснование и описание ключевых функций программного обеспечения.

В современном цифровом мире коммуникативные платформы стали неотъемлемой частью повседневной жизни, особенно в контексте спортивных сообществ, таких как футбольные клубы и фан-группы. Возможность оперативного обмена информацией, координации мероприятий и обсуждения новостей является важной для поддержания активности и вовлеченности участников сообщества. Программное средство, предоставляющее все эти функции, не только упрощает взаимодействие пользователей, но и повышает общий уровень организованности и информированности.

Пользователи могут зарегистрироваться в системе, указав свое имя, адрес электронной почты и пароль. После регистрации они могут аутентифицироваться с использованием логина и пароля. Обеспечение безопасного доступа к системе и предотвращение несанкционированного использования учетных записей.

Пользователи могут создавать новые темы для обсуждения и публиковать сообщения в существующих темах. Функционал включает возможность редактирования и удаления собственных постов. Модераторы управляют контентом форума. Форум является основным средством для общения и обмена мнениями среди участников сообщества, что способствует активному участию и взаимодействию.

Администраторы или авторизованные пользователи могут публиковать новости о футбольном клубе, указывая дату публикации. Пользователи могут редактировать и удалять собственные новости. Новости обеспечивают актуальную информацию о событиях и достижениях клуба, поддерживая интерес и информированность участников сообщества.

Функциональность включает отображение расписания предстоящих матчей и результатов прошедших матчей, включая счет. Администраторы могут обновлять результаты матчей. Информирование пользователей о предстоящих матчах и результатах помогает им оставаться в курсе последних событий и планировать свои действия.

Отображение информации о футбольном клубе, включая его название, местоположение, достижения, стадион, главного тренера и игроков. Предоставление полной информации о клубе способствует лучшему пониманию его структуры и достижений, а также укреплению связи между клубом и его поклонниками.

Администраторы имеют право на управление форумом, новостями, расписанием и результатами матчей, а также на назначение главного тренера и редактирование информации о футбольном клубе. Центральное управление контентом обеспечивает его актуальность и соответствие правилам сообщества.

Пользователи могут оставлять комментарии под новостями и сообщениями на форуме, а также оценивать их (например, лайки). Комментарии и оценки способствуют активному обсуждению и взаимодействию, помогая пользователям выражать свои мнения и реакции.

**4.2 Описание функциональных возможностей программы**

Программное средство для коммуникации пользователей, принадлежащих футбольному сообществу, обладает множеством функций, которые обеспечивают удобное взаимодействие и обмен информацией между пользователями.

Пользователи могут зарегистрироваться в системе, указав свое имя, адрес электронной почты и пароль. После регистрации пользователи могут аутентифицироваться, вводя логин и пароль. Это позволяет им получить доступ ко всем функциям системы и обеспечивается безопасностью с использованием JSON Web Token (JWT).

На форуме пользователи могут создавать новые темы для обсуждения и публиковать сообщения в существующих темах. Функционал включает возможность редактирования и удаления собственных постов. Модераторы имеют возможность управлять контентом форума, обеспечивая его соответствие правилам сообщества.

Администраторы или авторизованные пользователи могут публиковать новости о футбольном клубе. Эти новости доступны для всех пользователей и включают дату публикации. Пользователи могут редактировать и удалять собственные новости, поддерживая актуальность информации.

Функциональность включает отображение расписания предстоящих матчей футбольного клуба, а также результатов прошедших матчей, включая счет. Администраторы имеют возможность обновлять результаты матчей, чтобы пользователи всегда были в курсе последних событий.

Система предоставляет информацию о футбольном клубе, включая его название, местоположение, достижения и стадион. Пользователи также могут просматривать информацию о главном тренере и игроках футбольного клуба, что способствует лучшему пониманию состава и достижений команды.

Администраторы имеют широкие полномочия по управлению контентом на платформе. Они могут модерировать форум, управлять новостями, расписанием и результатами матчей, а также назначать главного тренера и редактировать информацию о футбольном клубе.

Пользователи могут оставлять комментарии под новостями и сообщениями на форуме, что способствует активному обсуждению и обмену мнениями. Возможность оценивать новости, сообщения и комментарии (например, лайки) позволяет пользователям выражать свое одобрение и участвовать в формировании рейтинга контента.

Функциональные требования:

1. Авторизация пользователя: Система должна обеспечивать надежную и безопасную авторизацию пользователей.
2. Управление пользователями (CRUD): Администраторы могут создавать, редактировать, обновлять и удалять пользователей.
3. Система ролей: Различные уровни доступа для пользователей в зависимости от их ролей (например, пользователь, модератор, администратор).
4. Журналирование действий пользователя: Система ведет журнал действий пользователей, что позволяет отслеживать изменения и обеспечивать безопасность.

Эти функциональные возможности обеспечивают комплексное решение для взаимодействия пользователей футбольного сообщества, предоставляя им все необходимые инструменты для общения, обмена информацией и управления контентом.

Код программы представлен в приложении А.

**4.3 Выполнение программы**

**4.4 Возможные улучшения функционала**

**5 ПРОЕКТИРОВАНИЕ РАЗРАБАТЫВАЕМОЙ БАЗЫ ДАННЫХ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

**5.1 Общая структура программы**

**5.2 Описание функциональной схемы программы**

**5.3 Блок-схема алгоритма программы**

# ЗАКЛЮЧЕНИЕ

# CПИСОК ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ

# ПРИЛОЖЕНИЕ А

**(обязательное)**

**Листинг программного кода**

Листинг 1 – Программный код разработанного приложения

# 

# ПРИЛОЖЕНИЕ Б

**(обязательное)**

**Функциональная схема алгоритма, реализующего программное средство**

# ПРИЛОЖЕНИЕ В

**(обязательное)**

**Ведомость**