Оглавление

[Введение 6](#_Toc193708772)

[1 Информационные системы анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств 8](#_Toc193708773)

[1.1 Анализ предметной области 8](#_Toc193708774)

[1.2 Аналоги информационной системы 9](#_Toc193708775)

[1.3 Выводы и постановка задач на дипломное проектирование 14](#_Toc193708776)

[2 Разработка информационной системы 16](#_Toc193708777)

[2.1 Описание технологий разработки программного средства 16](#_Toc193708778)

[2.2 Структура системы 18](#_Toc193708779)

[2.3 Алгоритм работы программы 19](#_Toc193708780)

[2.4 Структура базы данных программного средства 22](#_Toc193708781)

[2.5 Выводы и оценка результатов разработки 23](#_Toc193708782)

[3 Расчет надежности программного средства 25](#_Toc193708783)

[3.1 Расчет надежности программного средства по модели сложности 25](#_Toc193708784)

[3.2 Расчет надежности программного средства по модели Джелинского – Моранды 29](#_Toc193708785)

[4.3 Расчет по модели Муса 31](#_Toc193708786)

[4 Экономическое обоснование разработки и использования приложения для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств 33](#_Toc193708787)

[4.1 Характеристика разработанного программного средства по индивидуальному заказу 33](#_Toc193708788)

[4.2 Расчет основных затрат на разработку 34](#_Toc193708789)

[4.3 Расчет результата от разработки и реализации программного средства 38](#_Toc193708798)

[4.4 Расчет показателей экономической эффективности разработки и реализации программного средства 40](#_Toc193708799)

[5 Охрана труда. организация охраны труда на предприятии руп «Центр цифрового развития» 44](#_Toc193708800)

[Заключение 49](#_Toc193708801)

[Список использованных источников 50](#_Toc193708802)

# ВВЕДЕНИЕ

В современном мире качество питьевой воды остается одним из ключевых факторов, влияющих на здоровье населения. Контроль ее состава и своевременное информирование граждан о возможных отклонениях от нормы — важная задача для городских служб. В Минске данные о качестве воды публикуются на сайте Минскводоканала в табличном формате, что затрудняет их оперативный анализ и визуальное восприятие. Пользователям приходится вручную сравнивать значения с предельно допустимыми концентрациями (ПДК), что снижает удобство и скорость интерпретации данных..

Актуальность разработки обусловлена необходимостью улучшения способа визуализации информации о качестве воды для жителей Минска. Существующий табличный формат не позволяет быстро оценить общую ситуацию по городу, выявить районы с отклонениями от нормы или отследить динамику изменений. Внедрение интерактивной карты с цветовой дифференциацией показателей позволит сделать данные более наглядными и доступными для широкой аудитории.

Анализ текущего решения, предоставляемого сайтом Минсководоканала, показал, что табличный формат требует знаний в соответствующей области, что затрудняет понимаем о том, какое качество воды в заданном доме. Это создает нишу для разработки специализированного приложения, которое сможет улучшить наглядность данных.

Цель дипломного проектирования — разработка веб-приложения для усовершенствованной визуализации данных о качестве воды в Минске на основе гексагональной тепловой карты. Приложение должно автоматически загружать данные с сайта Минскводоканала, анализировать их на соответствие санитарным нормам и отображать результаты в формате, который не требует знаний норм предельно допустимых значений.

Объектом исследования являются методы визуализации пространсвенных данных, таких как качество воды в определнном районе, с помощью гексагональных сеток и тепловых карт. Предмет исследования — разработка алгоритмов обработки и отображения данных о качестве воды с учетом их динамики и территориального распределения.

Задачи проекта включают проектирование backend, отвечающую за автоматизацию сбора и обработки данных с сайта Минскводоканала. разработка алгоритма классификации качества воды по районам города (алгоритм объединение группы домов в один гексагон соответсвующего размера), оптимизацию запросов к PostgreSQL, реализация интерактивной карты с гексагональной сеткой и тестирование на реальных данных. Каждый из этапов разработки направлен на создание системы, которая будет не только функциональной, но и удобной в использовании.

Данный дипломный проект выполнен мной лично, проверен на заимствования, процент оригинальности составляет 72% (отчет о проверке на заимствования прилагается).

# 1 ИНФОРМАЦИОННЫЕ СИСТЕМЫ АНАЛИЗА РАБОЧЕГО ВРЕМЕНИ И ПОКАЗАТЕЛЕЙ ПРОИЗВОДИТЕЛЬНОСТИ ВОДИТЕЛЕЙ КРУПНОГАБАРИТНЫХ ТРАНСПОРТНЫХ СРЕДСТВ

## Анализ предметной области

В современном мире, где скорость и эффективность становятся ключевыми факторами успеха, транспортная и логистическая отрасли сталкиваются с необходимостью внедрения современных технологий для автоматизации процессов. Управление крупногабаритными транспортными средствами, контроль за рабочим временем водителей и анализ их производительности — это задачи, которые требуют высокой точности и оперативности. В условиях растущих объемов перевозок и ужесточения требований к безопасности и экологичности, компании вынуждены искать новые способы оптимизации своих процессов.

Разработка информационной системы анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств направлена на удовлетворение этих потребностей. Система позволяет автоматизировать учет рабочего времени, анализировать производительность водителей, формировать отчеты и визуализировать маршруты с помощью интеграции с картами. Это решение не только упрощает контроль за водительским составом, но и помогает транспортным компаниям снизить затраты, минимизировать риски и повысить прозрачность управления[2].

Разработка информационной системы анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств направлена на удовлетворение потребностей транспортных компаний в автоматизации учета и контроля деятельности водителей. Основные концептуальные требования включают:

1 Учет рабочего времени: необходимость фиксации времени начала и окончания поездок, перерывов и простоев.

2 Анализ производительности: оценка эффективности водителей на основе данных о пройденных маршрутах, времени в пути и других показателях.

3 Формирование отчетов: автоматизация создания отчетов для анализа деятельности водителей и принятия управленческих решений.

4 Интеграция с картами: возможность использования карт для визуализации маршрутов и расчета расстояний.

Целевая аудитория системы включает транспортные компании, диспетчерские службы и водителей. Транспортные компании заинтересованы в контроле рабочего времени водителей и оптимизации логистических процессов. Диспетчерские службы используют систему для мониторинга поездок и оперативного реагирования на отклонения от графика. Водители получают возможность самостоятельно вводить данные о поездках и просматривать свою статистику, что повышает прозрачность и вовлеченность.

Система состоит из трех основных структурных частей: мобильного приложения, сервера и базы данных. Мобильное приложение служит интерфейсом для взаимодействия с пользователями. Оно включает функции для ввода данных о поездках, интеграции с картами для визуализации маршрутов и отображения статистики. Серверная часть системы отвечает за обработку данных, выполнение бизнес-логики и управление взаимодействием между мобильным приложением и базой данных. Она включает API для приема и отправки данных, алгоритмы расчета рабочего времени и производительности, а также модуль управления пользователями. База данных, построенная на PostgreSQL, обеспечивает хранение информации о поездках, пользователях и результатах аналитики.

Взаимодействие между модулями организовано следующим образом: мобильное приложение отправляет данные о поездках на сервер через API. Сервер обрабатывает данные, применяет бизнес-логику и сохраняет результаты в базе данных. При необходимости сервер извлекает данные из базы и отправляет их обратно в мобильное приложение для отображения статистики или отчетов. Таким образом, система обеспечивает комплексный подход к учету и анализу рабочего времени и производительности водителей, упрощая процессы контроля и повышая прозрачность управления транспортными парками.

## 1.2 Аналоги информационной системы

Анализ существующих решений показал, что на рынке представлено несколько категорий систем, которые частично покрывают функционал, необходимый для анализа рабочего времени и производительности водителей крупногабаритных транспортных средств. Однако ни одно из них не предлагает комплексного подхода, сочетающего ручной ввод данных, интеграцию с картами и глубокую аналитику производительности. Рассмотрим наиболее значимые аналоги.

Toggl — это система учета рабочего времени, которая позволяет сотрудникам вручную вводить данные о времени, затраченном на задачи. Она широко используется в различных отраслях благодаря своей простоте и функциональности. Приложение поддерживает формирование отчетов, что позволяет менеджерам и владельцам бизнеса анализировать производительность сотрудников и оптимизировать рабочие процессы. Кроме того, Toggl интегрируется с популярными инструментами для управления проектами, такими как Trello, Asana и Jira, что делает ее удобным решением для команд, работающих над множеством задач одновременно [3].

Основные достоинства Toggl включают простоту использования, поддержку мобильных устройств и доступность на белорусском рынке. Интерфейс приложения интуитивно понятен, что позволяет пользователям быстро освоить его без дополнительного обучения. Мобильная версия Toggl доступна для iOS и Android, что делает ее удобной для использования в дороге или вне офиса. Это особенно важно для сотрудников, которые часто находятся в разъездах, например, для менеджеров по продажам или консультантов. Однако, несмотря на свои преимущества, система не адаптирована для транспорта и не предлагает функций для учета маршрутов или интеграции с картами. Это ограничивает ее применение в логистической отрасли, где важно учитывать не только время, но и географические данные.

Стоимость использования Toggl начинается от $9 за пользователя в месяц, что делает ее доступной для небольших и средних компаний. Бесплатная версия приложения также доступна, но она имеет ограниченный функционал, например, отсутствие возможности создавать детализированные отчеты или использовать интеграции с другими инструментами. Для крупных транспортных предприятий Toggl не подходит из-за ограниченного функционала, связанного с учетом специфики транспортной отрасли. Например, система не поддерживает учет времени вождения, анализ маршрутов или интеграцию с биометрическими датчиками, что делает ее непригодной для задач, связанных с управлением водительским составом [4].

Интерфейс приложения Toggl представлен на рисунке 1.1.

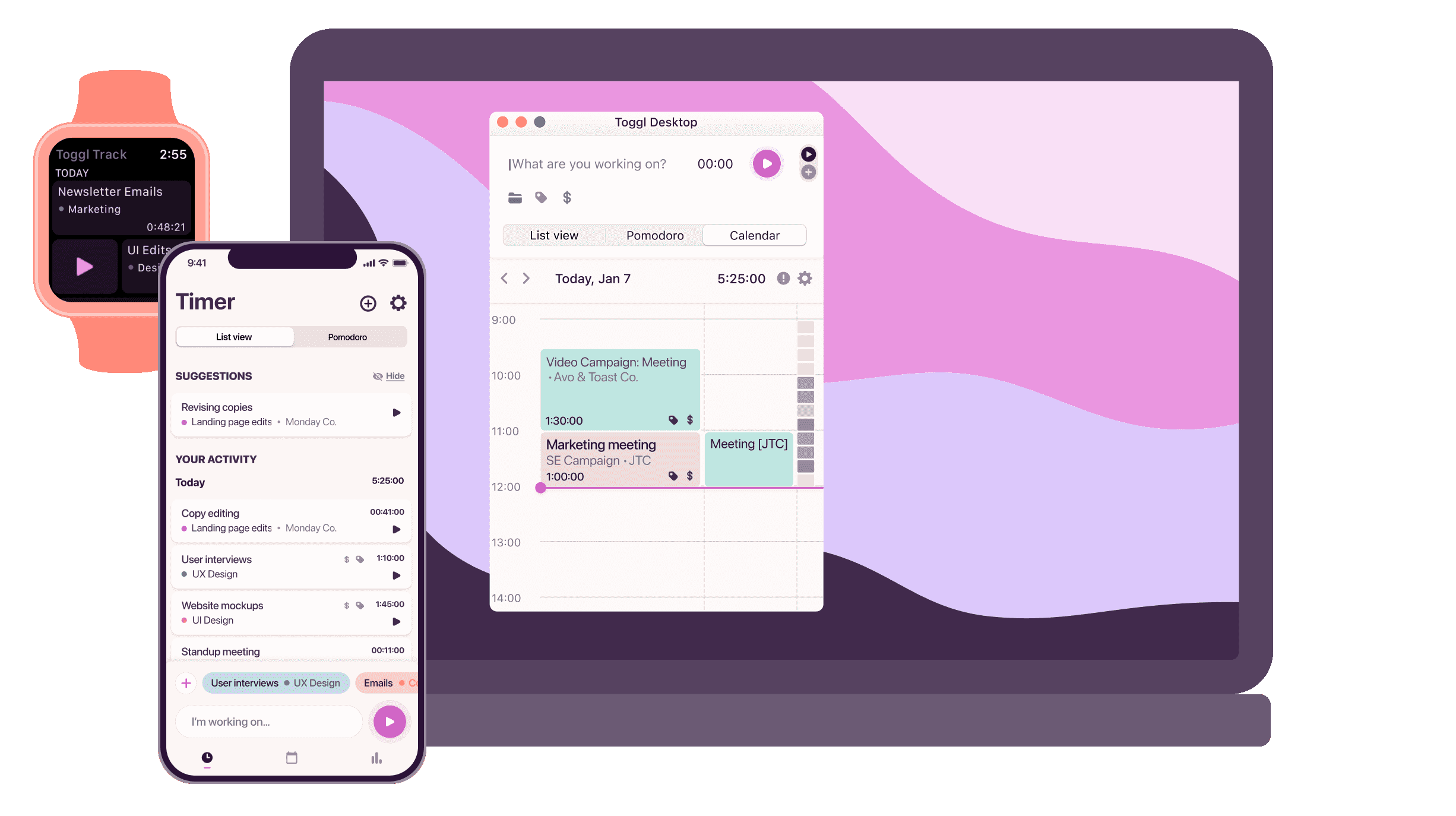


Рисунок 1.1 – Интерфейс приложения Toggl

1С:Логистика — это система для управления транспортными операциями, которая поддерживает ручной ввод данных о поездках, грузах и маршрутах. Она широко используется логистическими компаниями и транспортными предприятиями благодаря своему функционалу, включающему учет поездок, управление грузами и формирование отчетов. Система позволяет автоматизировать такие процессы, как планирование маршрутов, контроль загрузки транспортных средств, учет топлива и расчет себестоимости перевозок. Это делает ее популярным решением для компаний, которые стремятся оптимизировать свои логистические процессы и снизить затраты на транспортировку [5].

Основные достоинства 1С:Логистика — это широкая функциональность для логистики и доступность на белорусском рынке. Система поддерживает интеграцию с другими решениями на базе 1С, что позволяет создавать единую экосистему для управления бизнесом. Например, она может быть интегрирована с системами бухгалтерского учета, что упрощает процесс формирования финансовой отчетности. Кроме того, 1С:Логистика предлагает гибкие настройки, которые позволяют адаптировать систему под специфику конкретного предприятия.

Однако система сложна в адаптации под анализ производительности водителей и не предлагает глубокой аналитики по утомляемости. Это связано с тем, что 1С:Логистика ориентирована в первую очередь на управление транспортными операциями, а не на анализ данных о водителях. Например, система не поддерживает интеграцию с биометрическими датчиками или картографическими сервисами для анализа маршрутов и оценки утомляемости водителей. Это ограничивает ее применение в задачах, связанных с управлением водительским составом.

Стоимость лицензии начинается от $500, что делает ее дорогостоящей для небольших компаний. Кроме того, внедрение и обучение требуют значительных затрат, что ограничивает ее масштабируемость для крупных предприятий. Например, для настройки системы под нужды конкретной компании может потребоваться привлечение специалистов по 1С, что увеличивает общие затраты на внедрение. Также стоит учитывать, что система требует регулярного обновления и технической поддержки, что также связано с дополнительными расходами.

Пример интерфейса решения 1С:Логистика представлен на рисунке 1.2. Интерфейс системы выполнен в классическом стиле 1С, что делает его знакомым для пользователей, уже работающих с другими продуктами этой платформы. Однако для новых пользователей интерфейс может показаться сложным и перегруженным, что требует дополнительного времени на обучение.

Таким образом, 1С:Логистика является мощным инструментом для управления транспортными операциями, но не подходит для задач, связанных с анализом производительности водителей. Это подчеркивает необходимость разработки специализированного решения, которое сможет удовлетворить потребности транспортных компаний в анализе рабочего времени и производительности водителей. Интерфейс программы представлен на рисунке 1.2.

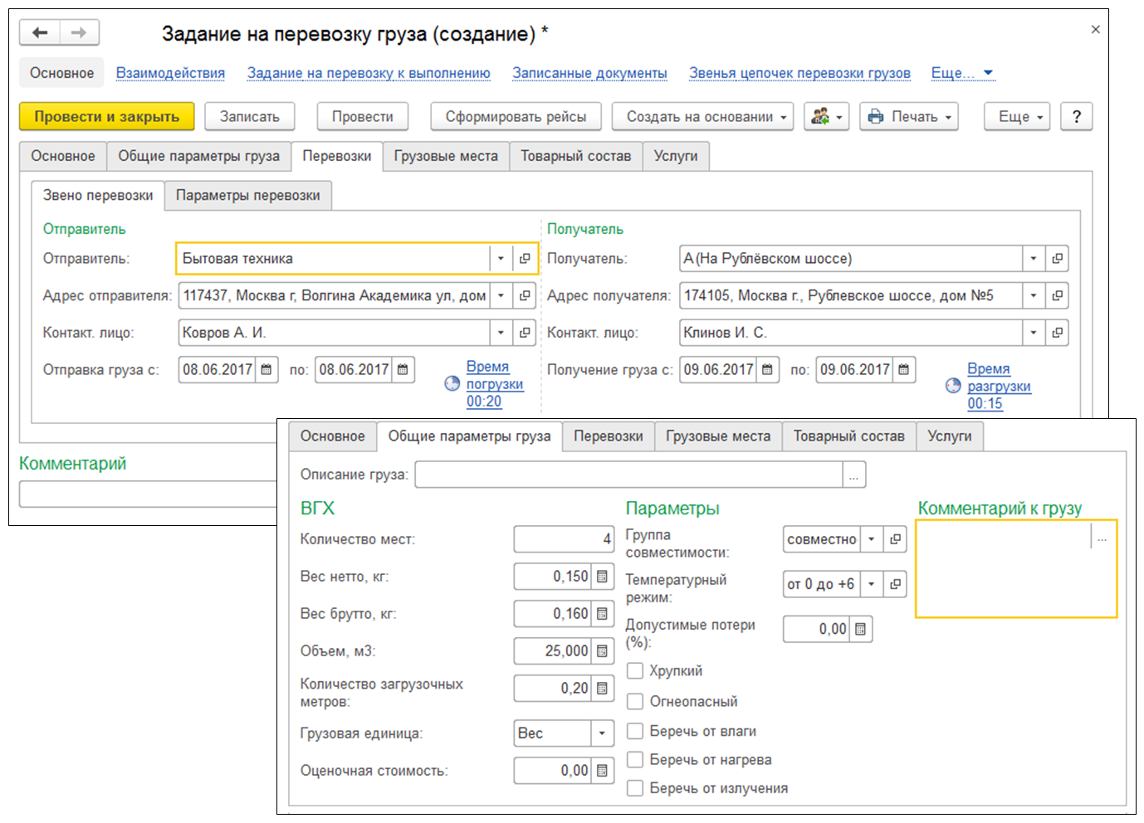


Рисунок 1.2 – Интерфейс создания задания на перевозку груза

Fleetio — это система для управления транспортными активами, поддерживающая ручной ввод данных о поездках и техническом обслуживании. Она широко используется транспортными компаниями благодаря функционалу, включающему учет поездок, управление активами и формирование отчетов. Основные достоинства — широкая функциональность для управления транспортом и поддержка мобильных устройств. Однако система ориентирована на управление активами, а не на анализ производительности водителей [6].

Стоимость использования начинается от $8 за транспортное средство в месяц, что делает ее дорогостоящей для небольших компаний. Внедрение требует значительных затрат, включая настройку и обучение, что ограничивает масштабируемость. Интерфейс Fleetio современный и интуитивно понятный, но может показаться перегруженным для новых пользователей [7].

Таким образом, Fleetio эффективна для управления автопарком, но не подходит для анализа производительности водителей, что подчеркивает необходимость разработки специализированного решения. Интерфейс данной программы представлен на рисунке 1.3.

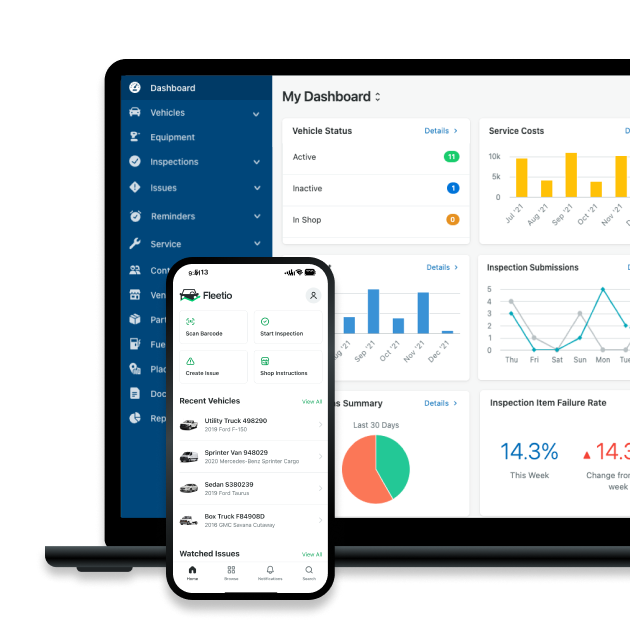


Рисунок 1.3 – Интерфейс приложения Fleetio

Driver’s Diary — это мобильное приложение, которое позволяет водителям вручную вводить данные о поездках, рабочем времени и расходах. Оно поддерживает интеграцию с картами для расчета расстояний, что делает его удобным для индивидуальных водителей и небольших транспортных компаний. Основные достоинства Driver’s Diary — это простота использования и мобильность. Однако приложение ограничено в функционале для анализа производительности и не поддерживает интеграцию с backend-системами, что делает его непригодным для крупных предприятий. Помимо этого, приложение недоступно на ОС Android. Базовая версия приложения бесплатна, а премиум-функции доступны от $2 в месяц [8].

Hubstaff — это система для анализа производительности сотрудников на основе данных о рабочем времени и активности. Она поддерживает формирование отчетов и интеграцию с другими инструментами, что делает ее популярной среди компаний, заинтересованных в повышении эффективности сотрудников. Основные достоинства Hubstaff — это глубокая аналитика производительности и поддержка мобильных устройств. Однако система не адаптирована для транспорта и не предлагает функций для учета маршрутов и поездок. Стоимость использования начинается от $7 за пользователя в месяц, что делает ее доступной для небольших и средних компаний. Однако для транспортных предприятий Hubstaff не подходит из-за ограниченного функционала [9].

Итог анализа аналогов показывает, что существующие системы частично покрывают функционал, необходимый для анализа рабочего времени и производительности водителей крупногабаритного транспорта. Однако ни одно из решений не предлагает комплексного подхода, сочетающего ручной ввод данных, интеграцию с картами и глубокую аналитику производительности. Большинство аналогов либо слишком дороги для небольших компаний (1С:Логистика, Fleetio), либо ограничены в функционале (Toggl, Driver’s Diary, Hubstaff). Разрабатываемая система будет выгодно отличаться доступностью для белорусского рынка, низкой стоимостью внедрения и специализированным функционалом для анализа производительности водителей. Это делает ее уникальным решением для транспортных компаний, заинтересованных в повышении эффективности управления водительским составом.

## 1.3 Выводы и постановка задач на дипломное проектирование

Разработка информационной системы анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств направлена на удовлетворение потребностей транспортных компаний в автоматизации учета и контроля деятельности водителей. Основная цель системы — обеспечить эффективный мониторинг рабочего времени, анализ производительности и формирование отчетов для принятия управленческих решений. Практическое назначение системы заключается в повышении прозрачности управления водительским составом, оптимизации логистических процессов и снижении затрат на эксплуатацию транспортных средств.

Требования к разрабатываемой системе включают создание трех основных компонентов: мобильного приложения для водителей, серверной части для обработки данных и базы данных для хранения информации. Мобильное приложение, разработанное на Android, обеспечивает ввод данных о поездках, интеграцию с картами для визуализации маршрутов и отображение статистики. Серверная часть, реализованная на Spring Boot, отвечает за обработку данных, выполнение бизнес-логики и управление взаимодействием между мобильным приложением и базой данных. База данных на PostgreSQL обеспечивает надежное хранение информации о поездках, пользователях и результатах аналитики.

Обмен данными между мобильным приложением и сервером осуществляется через REST API, что обеспечивает гибкость и масштабируемость системы. Сервер взаимодействует с базой данных через ORM (Object-Relational Mapping), что позволяет оптимизировать запросы и упростить работу с данными. Система должна работать в режиме реального времени, обеспечивая актуальность данных для пользователей. Также предусмотрена поддержка офлайн-режима для мобильного приложения с последующей синхронизацией данных при восстановлении соединения.

Взаимодействие со смежными системами включает интеграцию с картографическими сервисами, такими как Google Maps, для расчета расстояний и визуализации маршрутов. В перспективе система может быть расширена за счет интеграции с биометрическими датчиками и другими внешними системами, что позволит добавить функции анализа утомляемости водителей и прогнозирования производительности с использованием методов машинного обучения.

Эргономика системы играет важную роль в ее успешном внедрении. Интерфейс мобильного приложения должен быть интуитивно понятным и удобным для использования водителями, а серверная часть — обеспечивать быстрый доступ к данным и формирование отчетов в удобном для диспетчеров формате.

Разрабатываемая система обладает рядом преимуществ. Во-первых, она предлагает специализированный функционал для анализа рабочего времени и производительности водителей, что делает ее уникальной для транспортных компаний. Во-вторых, использование современных технологий, таких как Java, Spring Boot, Android и PostgreSQL, позволяет минимизировать затраты на разработку и поддержку. В-третьих, модульная архитектура системы обеспечивает ее масштабируемость, что позволяет легко расширять функционал в будущем. Наконец, система разрабатывается с учетом потребностей белорусского рынка, что делает ее доступной для локальных транспортных компаний.

Для администратора система предоставляет возможности управления пользователями, настройки системы, доступа к аналитическим отчетам и формирования рекомендаций по оптимизации рабочих процессов. Для водителей мобильное приложение становится удобным инструментом для ввода данных о поездках, просмотра статистики и получения уведомлений.

Выбор технологий для разработки системы обоснован их надежностью, производительностью и доступностью. Java и Spring Boot обеспечивают высокую производительность и безопасность серверной части, Android позволяет создать удобное мобильное приложение, а PostgreSQL — надежную и масштабируемую базу данных.

Таким образом, разрабатываемая система представляет собой специализированное решение, которое сочетает в себе простоту использования, глубокую аналитику и доступность для транспортных компаний. Это делает ее уникальным инструментом для повышения эффективности управления водительским составом и оптимизации логистических процессов.

# 2 РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 2.1 Описание технологий разработки программного средства

Разработка программного средства для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств потребовала выбора современных и надежных технологий, которые обеспечили бы высокую производительность, безопасность и масштабируемость системы. В основе разработки лежат три ключевых компонента: backend, мобильное приложение и база данных. Каждый из них был реализован с использованием технологий, которые наилучшим образом соответствуют поставленным задачам.

Backend-часть системы была разработана на языке программирования Java с использованием фреймворка Spring Boot. Этот выбор обусловлен несколькими причинами. Во-первых, Java является одним из самых популярных языков для разработки enterprise-приложений благодаря своей надежности, производительности и широкой поддержке сообщества [10]. Во-вторых, Spring Boot значительно упрощает создание RESTful API, обеспечивая встроенную поддержку безопасности, управления зависимостями и интеграции с базами данных. Одной из ключевых особенностей Spring Boot является его модульность. Для реализации функционала системы были использованы такие модули, как Spring Security для обеспечения безопасности, Spring Data JPA для работы с базой данных и Spring Web для создания REST API. Это позволило быстро и эффективно разработать backend, который отвечает за обработку данных, выполнение бизнес-логики и взаимодействие с мобильным приложением [11].

Мобильное приложение было разработано на платформе Android, что обеспечило его доступность для большинства пользователей. Android был выбран благодаря своей кроссплатформенности, широкой поддержке устройств и наличию мощных инструментов разработки, таких как Android Studio [12]. Приложение позволяет водителям вручную вводить данные о поездках, просматривать статистику и получать уведомления. Для обеспечения удобства пользователей интерфейс приложения был разработан с учетом принципов Material Design, что делает его интуитивно понятным и удобным в использовании. Также была реализована интеграция с картографическим сервисом OpenStreetMap, что позволяет бесплатно использовать его API для проектов [14].

Для хранения данных была выбрана реляционная база данных PostgreSQL. Этот выбор обусловлен ее надежностью, поддержкой сложных запросов и масштабируемостью. PostgreSQL позволяет эффективно хранить и обрабатывать большие объемы данных, что особенно важно для системы, которая анализирует рабочее время и производительность водителей. Для работы с базой данных был использован ORM (Object-Relational Mapping) инструмент Hibernate, который интегрирован с Spring Data JPA. Это позволило упростить процесс работы с данными и минимизировать количество ручных SQL-запросов [15].

В процессе разработки использовались следующие инструменты: IntelliJ IDEA и Android Studio в качестве основных сред разработки для написания кода на Java и Kotlin, Bruno для тестирования API благодаря его простоте и удобству использования, Git для контроля версий и эффективного управления кодом, а также Docker для контейнеризации приложения и упрощения процесса развертывания.

Одним из ключевых аспектов разработки стало обеспечение безопасности системы. Для аутентификации и авторизации пользователей была реализована схема OAuth 2.0 с использованием JWT (JSON Web Tokens). OAuth 2.0 — это протокол авторизации, который позволяет пользователям предоставлять доступ к своим данным третьим сторонам без передачи паролей [16]. В системе данный протокол используется для управления доступом к API и обеспечения безопасности данных. JWT — это компактный и самодостаточный способ передачи информации между сторонами в виде JSON-объекта. JWT используется для создания токенов, которые содержат информацию о пользователе и его правах доступа. Эти токены подписываются с использованием секретного ключа, что обеспечивает их защиту от подделки [17].

Для повышения производительности и обеспечения быстрого доступа к токенам была использована система кэширования Redis. Redis — это высокопроизводительная in-memory база данных, которая идеально подходит для хранения временных данных, таких как access и refresh токены [18]. При аутентификации пользователя refresh token сохраняется в Redis, что позволяет быстро проверять его валидность при обновлении access token. Это значительно снижает нагрузку на основную базу данных и ускоряет процесс аутентификации.

Реализация OAuth 2.0 и JWT в Spring Security, а также использование Redis для кэширования токенов позволили создать надежную систему аутентификации, которая обеспечивает безопасный доступ к данным и защищает систему от несанкционированного доступа [19].

Выбор технологий для разработки системы был основан на их надежности, производительности и доступности. Java и Spring Boot обеспечили высокую производительность и безопасность backend-части, Android — удобство и доступность мобильного приложения, а PostgreSQL — надежное хранение данных. Использование OAuth 2.0, JWT и Redis позволило создать безопасную и производительную систему аутентификации, что является критически важным для защиты данных пользователей. Таким образом, выбранные технологии и инструменты обеспечили эффективную разработку системы, которая соответствует всем поставленным требованиям и готова к дальнейшему масштабированию.

## 2.2 Структура системы

Система анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств обрабатывает данные, которые вводятся водителями вручную через мобильное приложение. Основные данные, которые обрабатываются системой, включают: название или код рейса в системе учета предприятия, дату начала и окончания поездки, место начала и место окончания маршрута, а также время, затраченное на сон. Время сна проверяется на соответствие нормам, установленным законодательством или внутренними правилами компании. На основе этих данных сервер рассчитывает общую скорость поездки, проверяет соответствие введенных данных нормам и вычисляет итоговую эффективность водителя.

Система состоит из нескольких модулей, которые взаимодействуют между собой через REST API [20]. Это обеспечивает масштабируемость и модульность системы, позволяя легко добавлять новые функции или изменять существующие без необходимости переписывания всей системы. Основные модули включают:

1 Модуль ввода данных (мобильное приложение) предназначенный для водителей и позволяющий им вводить данные о поездках. Водитель указывает название или код рейса, дату начала и окончания поездки, места начала и окончания маршрута, а также время, затраченное на сон. Данные отправляются на сервер через REST API.

2 Модуль обработки данных и формирования отчетов (серверная часть) отвечающий за обработку данных, полученных от мобильного приложения. Сервер рассчитывает общую скорость поездки на основе расстояния и времени, проверяет соответствие времени сна установленным нормам и вычисляет эффективность водителя. На основе этих данных формируется отчет, который отправляется обратно в мобильное приложение или веб-интерфейс для диспетчеров.

3 Модуль хранения данных (база данных) обеспечивает хранение всех данных, включая информацию о поездках, пользователях и результатах анализа. База данных построена на PostgreSQL, что обеспечивает надежность и масштабируемость.

Взаимодействие между модулями организовано следующим образом: мобильное приложение отправляет данные о поездках на сервер через REST API. Сервер обрабатывает данные, применяет бизнес-логику и сохраняет результаты в базе данных. При необходимости сервер извлекает данные из базы и формирует отчеты, которые отправляются обратно в мобильное приложение или веб-интерфейс.

Основные варианты взаимодействия пользователей с системой включают действия водителей, диспетчеров и администраторов. Водители используют мобильное приложение для ввода данных о поездках, просмотра статистики и отчетов о своей производительности, а также получения уведомлений о нарушениях норм или рекомендаций по улучшению эффективности. Диспетчеры используют веб-интерфейс для просмотра отчетов о производительности водителей, анализа данных о поездках и выявления тенденций, а также формирования рекомендаций для водителей на основе анализа данных. Администраторы управляют пользователями, настраивают систему и следят за ее работой, устраняя возможные неполадки.

Таким образом, система обеспечивает комплексный подход к учету и анализу рабочего времени и производительности водителей, упрощая процессы контроля и повышая прозрачность управления транспортными парками. Взаимодействие между модулями через REST API обеспечивает гибкость и масштабируемость системы, что позволяет легко адаптировать ее под нужды конкретных предприятий.

## 2.3 Алгоритм работы программы

Алгоритм работы программы включает несколько ключевых этапов, начиная с аутентификации пользователя и заканчивая формированием отчетов. Одним из важнейших аспектов системы является алгоритм аутентификации, который обеспечивает безопасный доступ к данным и защиту от несанкционированного доступа.

В приложении применяется следующий алгоритм авторизации пользователя в системе:

1 Пользователь (водитель, диспетчер или администратор) отправляет запрос на аутентификацию, предоставляя свои учетные данные (логин и пароль).

2 Сервер проверяет учетные данные в базе данных. Если данные верны, сервер генерирует два токена: access token и refresh token.

3 Access token имеет ограниченный срок действия (15 минут) и используется для доступа к защищенным ресурсам системы.

4 Refresh token имеет более длительный срок действия (7 дней) и используется для обновления access token после истечения его срока.

5 Оба токена сохраняются в базе данных Redis для быстрого доступа и кэширования.

Для наилучшего описания алгоритма действий всех участников системы была выбрана диаграмма взаимодействия, которая представлена на рисунке 2.1. Данные диаграммы используются, чтобы продемонстрировать взаимодействие актёров во времени и выделить этапы работы системы.

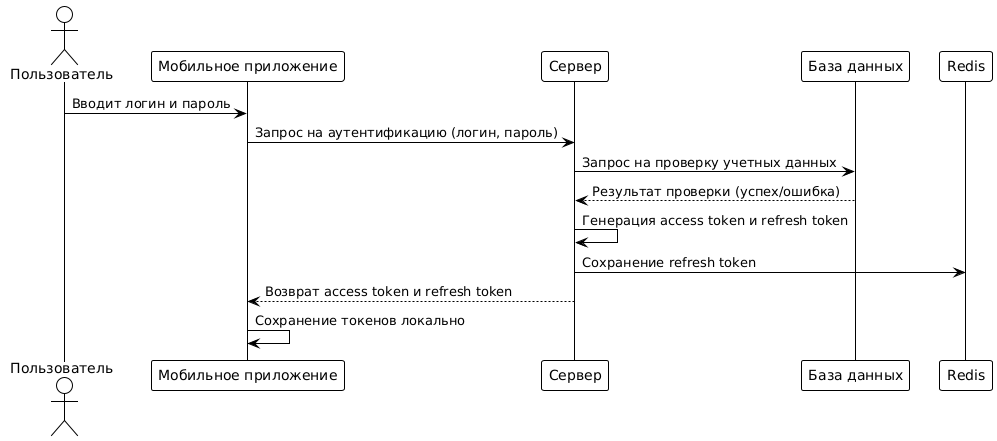


Рисунок 2.1 – Диаграмма взаимодействия при авторизации в системе

Алгоритм получения доступа к защищенным ресурсам системы:

1 При каждом запросе к защищенным ресурсам (например, ввод данных о поездке или запрос отчета) клиентское приложение отправляет access token в заголовке запроса.

2 Сервер проверяет валидность access token, используя секретный ключ для расшифровки. Если токен валиден, запрос обрабатывается.

3 Если access token истек, клиентское приложение отправляет refresh token на сервер для получения нового access token.

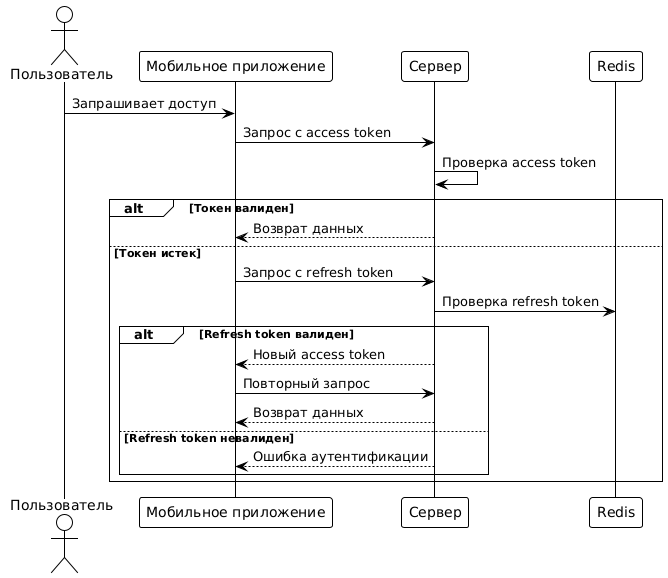


Рисунок 2.2 – Диаграмма взаимодействия при попытке доступа к системе

Алгоритм обновления токенов:

1 Если access token истек, клиентское приложение отправляет refresh token на сервер для получения нового access token.

2 Сервер проверяет refresh token в базе данных Redis. Если токен валиден, генерируется новый access token и отправляется клиенту.

3 Если refresh token также истек, пользователь должен пройти процедуру аутентификации заново.

Алгоритм ввода данных водителем:

1 Водитель вводит данные о поездке через мобильное приложение: название или код рейса, дату начала и окончания поездки, места начала и окончания маршрута, а также время, затраченное на сон.

2 Данные отправляются на сервер через REST API. Сервер проверяет access token в заголовке запроса для подтверждения прав доступа.

3 Если токен валиден, сервер сохраняет данные в базе данных PostgreSQL и начинает обработку.

Алгоритм обработки данных и расчет эффективности водителя:

1 Сервер рассчитывает общую скорость поездки на основе расстояния между точками начала и окончания маршрута и времени, затраченного на поездку.

2 Время сна проверяется на соответствие нормам, установленным законодательством или внутренними правилами компании.

3 На основе этих данных сервер вычисляет итоговую эффективность водителя.

Алгоритм формирования отчета:

1 По запросу от клиента (водителя или диспетчера) сервер извлекает данные из базы данных и формирует отчет.

2 Отчет включает информацию о поездках, скорости, времени сна и эффективности водителя.

3 Отчет отправляется обратно в мобильное приложение или веб-интерфейс для отображения.

Таким образом, алгоритм работы программы обеспечивает безопасный и эффективный процесс аутентификации, обработки данных и формирования отчетов. Использование access и refresh токенов, а также кэширование в Redis позволяют обеспечить высокую производительность и безопасность системы.

## 2.4 Структура базы данных программного средства

База данных системы анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств построена на PostgreSQL и включает несколько таблиц, которые обеспечивают хранение и обработку данных. Существующая структура базы данных уже включает таблицы для управления пользователями, их ролями, поездками, метриками поездок и уведомлениями.

Таблица u\_roles хранит роли пользователей, такие как водитель, диспетчер и администратор. Каждая роль имеет уникальный идентификатор, название, а также временные метки создания и обновления.

Таблица users содержит информацию о пользователях системы. В ней хранятся данные об уникальном имени пользователя, электронной почте, имени, фамилии, хэшированном пароле, а также временные метки создания и обновления записи.

Связь между пользователями и их ролями реализована через таблицу m2m\_users\_roles, которая обеспечивает отношение многие-ко-многим. Эта таблица содержит внешние ключи на таблицы users и u\_roles, что позволяет назначать пользователям несколько ролей.

Таблица trips хранит данные о поездках, совершенных водителями. Каждая запись включает уникальный идентификатор поездки, код или название рейса, дату и время начала и окончания поездки, места начала и окончания маршрута, а также время, затраченное на сон. Временные метки создания и обновления записи также сохраняются.

Расчетные показатели поездок, такие как средняя скорость, оценка эффективности водителя и соответствие времени сна нормам, хранятся в таблице trip\_metrics. Эта таблица связана с таблицей trips через внешний ключ, что позволяет хранить метрики для каждой поездки.

Таблица notifications используется для хранения уведомлений, отправляемых пользователям. Каждое уведомление содержит текст сообщения, статус прочтения и временные метки создания и обновления. Уведомления связаны с пользователями через внешний ключ на таблицу users.

Таким образом, база данных системы обеспечивает гибкость и масштабируемость, позволяя эффективно хранить и обрабатывать данные о пользователях, поездках, метриках и уведомлениях. Такая структура позволяет системе анализировать рабочее время и производительность водителей, а также предоставлять пользователям актуальную информацию и рекомендации.

## 2.5 Выводы и оценка результатов разработки

Разработка программного средства для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств завершена успешно. Система соответствует всем поставленным требованиям и обеспечивает эффективное решение задач учета, анализа и отчетности.

Основным результатом разработки стало создание модульной и масштабируемой системы, которая включает мобильное приложение для водителей, серверную часть для обработки данных и базу данных для хранения информации. Использование современных технологий, таких как Java, Spring Boot, Android и PostgreSQL, позволило обеспечить высокую производительность, безопасность и удобство использования системы.

Одним из ключевых преимуществ системы является ее специализированный функционал, ориентированный на анализ рабочего времени и производительности водителей. Система позволяет водителям вручную вводить данные о поездках, включая время, маршруты и время сна, а также получать отчеты о своей производительности. Для диспетчеров и администраторов система предоставляет инструменты для анализа данных, формирования отчетов и управления пользователями.

Безопасность системы обеспечивается за счет реализации протокола OAuth 2.0 и использования JWT (JSON Web Tokens) для аутентификации и авторизации пользователей. Кэширование токенов в Redis позволяет повысить производительность и обеспечить быстрый доступ к данным.

База данных системы, построенная на PostgreSQL, обеспечивает надежное хранение и обработку данных. Структура базы данных включает таблицы для управления пользователями, ролями, поездками, метриками поездок и уведомлениями. Такая структура позволяет системе эффективно анализировать данные и предоставлять пользователям актуальную информацию.

В процессе разработки были успешно решены задачи проектирования архитектуры системы, реализации бизнес-логики, интеграции с картографическими сервисами и обеспечения безопасности. Система прошла тестирование на реальных данных, подтвердив свою работоспособность и соответствие требованиям.

Таким образом, разработанное программное средство представляет собой эффективное решение для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств. Оно обеспечивает прозрачность управления водительским составом, оптимизацию логистических процессов и снижение затрат на эксплуатацию транспортных средств. Система готова к внедрению и дальнейшему масштабированию, что делает ее уникальным инструментом для транспортных компаний.

# 3 РАСЧЕТ НАДЕЖНОСТИ ПРОГРАММНОГО СРЕДСТВА

## 3.1 Расчет надежности программного средства по модели сложности

Модель сложности программного обеспечения основывается на предположении, что уровень безошибочности продукта может быть предсказан с использованием метрик сложности. Это справедливо для непреднамеренных уязвимостей, так как, чем сложнее и больше программа, тем выше вероятность ошибок при её написании и модификации. Для расчета надежности программного средства по модели сложности используются метрики размера, сложности потока управления, сложности потока данных, а также объектно-ориентированные метрики.

К метрикам размера относятся объем программы V и потенциальный объем программы V\*.

Объем программы V рассчитывается по формуле 3.1:

(3.1)

V=(N1​+N2​) ⋅log2​(n1​+n2​),

где:

– n1 = 8 (число уникальных операторов);

– n2 =16 (число уникальных операндов);

– N1 =150 (общее число операторов);

– N2 =75 (общее число операндов).

Потенциальный объем программы V\* определяется по формуле 3.2:

(3.2)

V∗=n∗⋅log2​(n∗),

где n\* – теоретический словарь программы (словарный запас, необходимый для написания программы с учетом того, что необходимая функция уже реализована в языке программирования). Для разработанной программы n\* =

Для разрабатываемого программного средства значения объема и потенциального объема равны:

V = (150+75) ⋅log2​ (8+16) =225⋅log2​ (24) = 1031,6

V\*=128⋅log2(128) = 896.

Для определения метрик сложности потока управления используются следующие характеристики:

– CL – абсолютная сложность программы, характеризующаяся количеством операторов условия;

– cl – относительная сложность программы, определяемая как отношение CL к общему числу операторов;

– CLI – максимальный уровень вложенности операторов условия.

Для разрабатываемого программного средства значения этих характеристик равны: CL =200, cl = 25, CLI = 41.

Метрика сложности потока данных, предложенная Чепиным, оценивает информационную прочность программного модуля на основе анализа использования переменных из списка ввода-вывода. Все переменные делятся на четыре группы:

– P = 30 (вводимые переменные для расчетов и обеспечения вывода);

– M = 40 (модифицируемые или создаваемые внутри программы переменные);

– C = 5 (управляющие переменные);

– T = 0 (не используемые в программе переменные).

Метрика Чепина Q рассчитывается по формуле 3.3:

(3.3)

Q = P + 2M + 3C + 0,5T.

Для разрабатываемого программного средства значение метрики Чепина равно Q = 125.

Теоретическая длина программы N рассчитывается по формуле 3.4:

(3.4)

где n1 – словарь операторов, n2– словарь операндов.

Для разрабатываемого программного средства N = 88.

Уровень качества программирования L определяется как отношение потенциального объема к фактическому по формуле 3.5:

(3.5)

Для разрабатываемого программного средства L = 0,87.

Аппроксимированный уровень качества программирования L^ рассчитывается по формуле 3.6:

(3.6)

​

Для разрабатываемого программного средства L^ = 0,052.

Интеллектуальные усилия на разработку программы E рассчитываются по формуле 3.7:

Для разрабатываемого программного средства E = 1187,5.

(3.7)

Для оценки объектно-ориентированных характеристик программного средства используются метрики Чидамбера и Кемерера:

– WMC – суммарная сложность всех методов класса;

– DIT – глубина дерева наследования;

– NOC – количество потомков;

– CBO – сцепление между классами;

– RFC – отклик для класса;

– LCOM – недостаток сцепления методов.

Для разрабатываемого программного средства значения этих метрик равны: WMC = 201, DIT = 5, NOC = 15, CBO = 9, RFC = 125, LCOM = 20.

Результаты расчетов представлены в таблице 3.1.

Таблица 3.1 – Результаты расчетов

|  |  |  |  |
| --- | --- | --- | --- |
| Метрики ПС | amin | amax | ai |
| V | 108 | 1364 | 1031,6 |
| V\* | 540 | 83362,41 | 896 |
| CL | 7 | 368 | 200 |
| Cl | 25 | 1278 | 25 |
| CLI | 41 | 4213 | 41 |
| Q | 67 | 3589 | 125 |
| N^ | 6 | 186 | 88 |
| L | 0,0054 | 2 | 0,87 |
| L^ | 0 | 3 | 0,052 |
| E | 29 | 967 | 1187,5 |
| WMC | 13 | 299 | 201 |
| DIT | 1 | 8 | 5 |
| NOC | 1 | 32 | 15 |
| CBO | 1 | 27 | 9 |
| RFC | 1 | 163 | 125 |
| LCOM | –39 | 387 | 20 |

Исходя из полученных метрик рассчитаны нормативный и фактический уровни каждой метрики (формулы (3.8) и (3.9) соответственно):

(3.8)

(3.9)

По полученным результатам определен дискриминант каждой метрики, используя формулу (3.10):

(3.10)

Результаты вычислений по формулам (3.8 – 3.10) представлены в таблице 3.2.

Таблица 4.2 – Результаты вычисления

|  |  |  |  |
| --- | --- | --- | --- |
| Метрики ПС | xmin | xфi | di |
| V | 0,079179 | 0,756305 | 0,027707 |
| V\* | 0,006478 | 0,010748 | 0,600088 |
| CL | 0,019022 | 5,095109 | -0,01558 |
| Cl | 0,019562 | 0,019562 | 1 |
| CLI | 0,009732 | 0,009732 | 1 |
| Q | 0,018668 | 0,034829 | 0,527173 |
| N^ | 0,032258 | 0,473118 | 0,037121 |
| L | 0,0027 | 0,435 | 0,003516 |
| L^ | 0 | 0,017333 | 0 |
| E | 0,02999 | 1,228025 | -0,00574 |
| WMC | 0,043478 | 0,672241 | 0,022162 |
| DIT | 0,125 | 0,625 | 0,085714 |
| NOC | 0,03125 | 0,46875 | 0,036559 |
| CBO | 0,037037 | 0,333333 | 0,076923 |
| RFC | 0,006135 | 0,766871 | 0,001877 |
| LCOM | -0,10078 | 0,05168 | -1,67993 |

Риск снижения надежности рассчитывается по формуле 3.11:

(3.11)

Где λi – весовые коэффициенты для каждой метрики. Для упрощения расчетов можно считать, что все метрики вносят равный вклад в надежность, т.е. λi = 1/16.

Для разрабатываемого программного средства вероятность безотказной работы равна 3.12:

(3.12)

Таким образом, вероятность безотказной работы программного средства по модели сложности составляет приблизительно 0,8.

## 3**.2 Расчет надежности программного средства по модели Джелинского – Моранды**

Модель Джелинского – Моранды используется для оценки надежности программного обеспечения на основе данных о времени между обнаружением ошибок. Она предполагает, что интенсивность обнаружения ошибок зависит от текущего количества оставшихся ошибок в программе. В данном разделе проведем расчет надежности программного средства на основе модели Джелинского – Моранды. Общее время тестирования программы составило 15 дней. За это время было обнаружено 20 ошибок. Время между обнаружением ошибок (Xi​) приведено в таблице 3.3.

Таблица 3.3 – Время между ошибками (Xi​) в часах:

|  |  |
| --- | --- |
| Номер ошибки (i) | Время между ошибками (Xi) |
| 1 | 4 |
| 2 | 6 |
| 3 | 5 |
| 4 | 7 |
| 5 | 3 |
| 6 | 8 |
| 7 | 6 |
| 8 | 4 |
| 9 | 5 |
| 10 | 7 |
| 11 | 5 |
| 12 | 6 |
| 13 | 4 |
| 14 | 5 |
| 15 | 7 |
| 16 | 3 |
| 17 | 6 |
| 18 | 4 |
| 19 | 5 |
| 20 | 6 |

Для оценки общего числа ошибок B*B* в программном средстве используется метод, основанный на минимизации разности функций f(m) и g(m,A)*.* Функции f(m) и g(m,A) определяются следующим образом по формулам 4.13 и 4.14:

(3.13)

f(m)=n/m,

(3.14)

g(m, A)=A/m,

где A — константа, зависящая от данных. В данном случае предположим, что A=12> (20 + 1)/2.

Для нахождения B выполняется поиск такого значения m, при котором разность f(m)−g(m,A) минимальна. Результаты расчетов для различных значений m показывают, что минимальная разность достигается при m=26, что соответствует общему числу ошибок B=25 (так как B=m−1).

Коэффициент K*K* рассчитывается по формуле 3.14:

(3.14)

K=nB⋅T−∑i=1ni⋅Xi.

Для расчета суммы ∑i=1ni⋅Xi∑*i*=1*n*​*i*⋅*Xi*​ используются времена между отказами Xi​. Подставляя значения, получаем:

K=2025⋅120−1260=203000−1260=201740≈0.0115.

Интенсивность отказов λ(t)после обнаружения 20 ошибок рассчитывается по формуле:

λ(t)=K⋅(B−20) =0.0115⋅(25−20)=0.0115⋅5=0.0575.

Вероятность безотказной работы P(t)*P*(*t*) на интервале t=1*t*=1 час определяется по формуле 3.15:

(3.15)

P(1)=e−λ(t)⋅t=e−0.0575⋅1≈0.944.

Для времени t=1 час:

P(t)=e−0,1196⋅1≈e−0,1196≈0,887.

Среднее время до появления следующей ошибки рассчитывается по формуле 3.16:

(3.16)

X21=1K⋅(B−20) =10.0115⋅5≈17.39 часов.

Время до окончания тестирования оценивается как сумма времени до обнаружения оставшихся ошибок (формула 3.17):

(3.17)

Предположив, что ошибки будут находиться через среднее значение времени, получим t = 90 часов.

Таким образом, модель Джелинского – Моранды позволяет оценить надежность программного средства и спрогнозировать время, необходимое для обнаружения и исправления всех ошибок.

## 4.3 Расчет по модели Муса

Проведем расчет надежности по модели Муса. В этой модели надежность программного средства на этапе эксплуатации оценивается по результатам тестирования. Пусть Т – суммарное время тестирования, М – число отказов, произошедших за время тестирования. Тогда по модели Муса средняя наработка до отказа после тестирования определяется по формуле (3.18).

|  |  |
| --- | --- |
|  | (3.18) |

В этой формуле τ0 – средняя наработка до отказа до начала тестирования, С – коэффициент, учитывающий уплотнение тестового времени по сравнению с временем реальной эксплуатации. Например, если один час тестирования соответствует 12 часам работы в реальных условиях, то С = 12. Неизвестный параметр τ0 можно оценить из соотношения (4.19).

|  |  |
| --- | --- |
|  | (3.19) |

В этой формуле N – первоначальное число ошибок в программной средстве, которое можно оценить с помощью другой модели, позволяющей определить N на основе статистических данных, полученных при тестировании. К – коэффициент проявления ошибок, который определяется эмпирическим путем по однотипным программам, обычно это значение изменяется от 1,5·10–7 до 4·10–7. f – средняя скорость исполнения программы (A), деленная на число команд (операторов (B)).

Средняя скорость исполнения одного оператора программного средства определяется по формуле (3.20).

|  |  |
| --- | --- |
|  | (3.20) |

Длительности этапов тестирования составляют t1 = 50 ч, t2 = 60 ч, t3 = 10 ч. Число отказов на первом этапе m1 = 10, на втором – m2 = 8, на третьем – m3 = 2. Средняя скорость исполнения программного средства = 107 операторов/ч, количество операторов в программном средстве = 1050. Определим надежность системы для периода эксплуатации t = 90 ч.

Найдем среднюю скорость выполнения одного оператора по формуле (3.21):

(3.21)

Первоначальное количество ошибок в программном средстве N равно 20. Коэффициент проявления ошибок K примем равным 3·10-7. Найдем формуле (3.22).

(3.22)

Примем значение коэффициента С = 8. Тогда средняя наработка до отказа после тестирования на этапе эксплуатации программного обеспечения рассчитывается по формуле (3.18).

Найдем надежность программного средства для периода эксплуатации t равному 90 часам по формуле (3.23).

|  |  |
| --- | --- |
|  | (3.23) |

Надежность равна:

# 4 Экономическое обоснование разработки и использования приложения для АНАЛИЗА РАБОЧЕГО ВРЕМЕНИ И ПОКАЗАТЕЛЕЙ ПРОИЗВОДИТЕЛЬНОСТИ ВОДИТЕЛЕЙ КРУПНОГАБАРИТНЫХ ТРАНСПОРТНЫХ СРЕДСТВ

## 4.1 Характеристика разработанного программного средства по индивидуальному заказу

Данный раздел посвящен экономическому обоснованию дипломного проекта на тему «Приложение для анализа рабочего времени и показателей производительности крупногабаритных транспортных средств». Проект относится к типу разрабатываемых по индивидуальному заказу и направлен на автоматизацию учета рабочего времени и формирования отчетов по производительности водителей в логистических компаниях.

Так как разработка программного обеспечения ведется для сторонней организации, для экономического обоснования выбрана методика обоснования разработки по индивидуальному заказу.

Основная цель проекта – автоматизация рутинных процессов сбора данных и анализа профессиональных показателей водителей.

В настоящее время учет рабочего времени и производительности водителей в логистических компаниях часто ведется вручную, что приводит к ошибкам, задержкам в формировании отчетов и увеличению затрат на персонал. Разрабатываемое приложение позволит автоматизировать процесс учета рабочего времени и формирования отчетов, что повысит точность данных, сократит время на обработку информации и снизит затраты на ручной труд.

Область применения проекта — логистическая отрасль. Основными пользователями приложения будут менеджеры и диспетчеры, ответственные за учет рабочего времени и контроль производительности водителей, а также водители, которые смогут отслеживать свои показатели в режиме реального времени. Для менеджеров приложение станет инструментом для оперативного контроля и анализа данных, а для водителей — удобным способом отслеживания своей производительности и рабочего времени.

Основным конкурентным преимуществом приложения является удобство внесения данных водителями и простота анализа данных за счет автоматизации бизнес-процессов. Помимо этого, приложение позволит формировать отчеты об эффективности водителей для менеджмента ресурсов компании.

Экономическая оценка целесообразности инвестиций в разработку и использование программного средства осуществляется на основе расчета и оценки следующих показателей: чистый дисконтированный доход, рентабельность инвестиций и срок окупаемости инвестиций.

## 4.2 Расчет основных затрат на разработку

## 4.2.1 Расчет затрат на основную заработную плату разработчикам

Для расчета затрат на разработку программного средства в первую очередь необходимо рассчитать основную заработную плату команды разработчиков.

Расчет осуществляется исходя из состава и численности команды, размера месячной заработной платы каждого участника команды, а также трудоемкости работ, выполняемых при разработке программного средства отдельными исполнителями по формуле:

(4.1)

где Кпр ‒ коэффициент премий (равный 1,5);

*n* ‒ категории исполнителей, занятых разработкой программного средства;

Зчi ‒ часовая заработная плата исполнителя *i*-й категории, р.;

*ti* ‒ трудоемкость работ, выполняемых исполнителем *i*-й категории, определяется исходя из сложности разработки программного обеспечения и объема выполняемых им функций, ч.

На 2025 год расчетная норма рабочего времени для пятидневной рабочей недели составляет 168 часов, 8 часов работы в день, среднемесячная расчетная норма рабочего времени – 21 день.

В разработке были задействованы следующие сотрудники: бизнес-аналитик, программист, тестировщик. Часовая заработная плата каждого сотрудника определена как результат деления месячной заработной платы (оклада) на количество рабочих часов в месяце. Расчет основных затрат представлен в таблице 4.1.

Таблица 4.1 ‒ Расчет затрат на основную заработную плату разработчиков

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Категория исполнителя | Месячный оклад, р | Часовой оклад, р | Трудоемкость работ, Р | Итого, р |
| Бизнес-аналитик | 2943 | 17 | 25 | 437 |
| Программист | 1962 | 11 | 450 | 5251 |
| Тестировщик | 2125 | 12 | 120 | 1518 |
| Итого | | | | 7206 |
| Премия и иные стимулирующие выплаты (50%) | | | | 3603 |
| Основная заработная плата разработчиков | | | | 10809 |

## 4.2.2 Расчет затрат на дополнительную заработную плату разработчикам

Дополнительная заработная плата ‒ это оплата за сверхурочный труд, различные трудовые успехи и надбавки за особые условия труда команды и включает выплаты, предусмотренные законодательством о труде, и определяется по нормативу в процентах (составляет 20%) к основной заработной плате по следующей формуле:

(4.2)

где Зо ‒ затраты на основную заработную плату;

Нд ‒ норматив дополнительной заработной платы, 20%.

Подставим значение в формулу (4.2) и вычислим Зд:

Согласно расчетам, затраты на дополнительную заработную плату разработчикам составит 2162 рубля.

## 4.2.3 Расчет отчислений на социальные нужды

В расчете отчислений на социальные нужды учитываются обязательные платежи по установленным законодательством тарифам в фонд социальной защиты населения, а также затраты предприятия на обязательное социальное медицинское страхование для определенных категорий работников в соответствии с законодательством. Размер отчислений в фонд социальной защиты населения и на обязательное страхование определяется в соответствии с действующими законодательными актами Республики Беларусь и рассчитывается по установленной формуле:

(4.3)

где Нсоц ‒ норматив отчислений на социальные нужды, %.

Согласно законодательству Республики Беларусь, отчисления на социальные нужды составляют 29% в фонд социальной защиты и 6% на обязательное страхование. Подставим результаты вычислений в формулу (4.3) и вычислим Рсоц:

Согласно расчетам, размер отчислений в фонд социальной защиты и на обязательное страхование составляет 4539 рублей.

## 4.2.4 Расчет затрат на прочие расходы

Прочие расходы связаны с функционированием организации-разработчика в целом, например: затраты на аренду офисных помещений, отопление, освещение, амортизацию основных производственных фондов и так далее. При расчете данной статьи затрат учитывается норматив прочих затрат в целом по организации. В данном случае норматив прочих затрат равен 20 %. Размер затрат на прочие расходы рассчитывается по формуле:

(4.4)

где Ннз ‒ норматив прочих затрат в целом по организации, 20 %.

Подставим значение из выражения в формулу (4.4) и произведем расчет Рпр:

Согласно расчетам, размер затрат на прочие расходы составляет 2162 рубля.

## 4.2.5 Расчет суммы затрат на разработку

Общая сумма затрат на разработку рассчитывается путем суммирования основной заработной платы, дополнительной заработной платы, отчислений на социальные нужды, прочих затрат. Формула расчета имеет следующий вид:

(4.5)

Подставим результаты вычислений в формулу (4.5) и произведем расчет Зр:

Согласно расчетам, сумма затрат на разработку составляет 19672 рубля.

## 4.2.6 Расчет плановой прибыли, включаемой в цену программного средства

Плановая прибыль, включаемая в цену программного средства, рассчитывается по формуле:

(4.6)

В данном случае рентабельность затрат на разработку программного средства установили на уровне 35%. Подставим значение из выражения в формулу (4.6) и произведем расчет Пп.с.:

Исходя из расчетов, плановая прибыль, включаемая в цену программного средства, составляет 6885,2 рублей.

## 4.2.7 Расчет отпускной цены программного средства

Отпускная цена программного продукта представляет собой сумму затрат на заработную плату и плановой прибыли. Рассмотрим формулу расчета отпускной цены программного средства:

(4.7)

Подставим результат вычислений и произведем расчет Цп.с.:

Исходя из расчетов, отпускная цена программного средства составляет 26557,2 рублей.

## 4.2.8 Результаты расчета затрат на разработку и цены приложения для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств

В данном подразделе были рассчитаны необходимые статьи для расчета затрат на разработку и для расчета цены программного средства, а именно: основная заработная плата разработчиков, дополнительная заработная плата разработчиков, отчисления на социальные нужды, прочие расходы и плановая прибыль. Результаты расчетов представлены в таблице 4.2.

Таблица 4.2 ‒ Результаты расчета цены на разработку программного средства

|  |  |
| --- | --- |
| Наименование статьи затрат | Сумма, р. |
| 1 Основная заработная плата разработчиков | 10809 |
| 2 Дополнительная заработная плата разработчиков | 2162 |
| 3 Отчисления на социальные нужды | 4488 |
| 4 Прочие расходы | 2162 |
| 5 Всего затраты на разработку | 19672 |
| 6 Плановая прибыль | 6885,2 |
| 7 Цена программного средства | 26557,2 |

## 4.3 Расчет результата от разработки и реализации программного средства

Для организации-разработчика экономическим эффектом является прирост чистой прибыли, полученной от разработки и реализации программного средства заказчику. Так как программное средство будет реализовываться организацией-разработчиком по отпускной цене, сформированной на основе затрат на разработку, то экономический эффект, полученный организацией-разработчиком, в виде прироста чистой прибыли от его разработки, определяется по формуле:

(4.8)

где Пп.с. ‒ прибыль, включаемая в цену программного средства, р;

Нп   ‒ ставка налога на прибыль согласно действующему законодательству, (по состоянию на 01.01.2024 г. – 20%).

Подставим результат вычисления в формулу (4.8) и произведем расчет ΔПч:

Исходя из расчетов, экономический эффект составляет 5508,16 рублей.

Для организации-заказчика расчет экономического эффекта от использования программного обеспечения, разработанного по индивидуальному заказу сторонней организацией, осуществляется в соответствии с методикой расчета основных видов экономического эффекта.

Экономия на заработной плате и начислениях на заработную плату сотрудников за счет снижения трудоемкости работ определяется по формуле:

(4.9)

где Кпр – коэффициент премий (по фактическим данным предприятия или в диапазоне 1,5-2);

*tр*без п.с, *t*рс п.с – трудоемкость выполнения работ сотрудниками до и после внедрения программного средства, ч;

*Т*ч – часовой оклад (часовая тарифная ставка) сотрудника, использующего программное средство, р;

*N*п – плановый объем работ, выполняемых сотрудником;

Нд – норматив дополнительной заработной платы;

Нсоц – ставка отчислений от заработной платы, включаемых в себестоимость (34,6%).

Подставим результат вычисления в формулу (4.9) и произведем расчет Эз.п:

Экономия на заработной плате и начислениях на заработную плату в результате сокращения численности работников определяется по формуле:

(4.10)

где *n* – категории работников, высвобождаемых в результате внедрения программного средства;

ΔЧ*i*  – численность работников *i*-й категории, высвобожденных после внедрения программного средства, чел.;

З*i*    – годовая заработная плата высвобожденных работников *i*-й категории после внедрения программного средства, р.;

Нсоц – норматив отчислений от заработной платы в соответствии с законодательством, %.

Экономия на заработной плате и начислениях на заработную плату в результате сокращения численности работников составляет 0 р., поскольку сотрудники не были уволены.

Экономия на материальных ресурсах в результате внедрения программного средства определяется по формуле:

(4.11)

где Кт.р. – коэффициент транспортных расходов (по данным предприятия или 1,05-1,2);

Нрбез п.с, Нрсп.с – норма расхода материальных ресурсов при выполнении работ сотрудниками до и после внедрения программного средства, нат. ед.;

Цм – цена за единицу материального ресурса, р.;

*N*п – плановый объем работ, выполняемых сотрудником с использованием программного средства.

Экономия на материальных ресурсах так же равно 0 р., поскольку расход материальных ресурсов не изменился.

Экономическим эффектом при использовании программного средства является прирост чистой прибыли, полученной за счет экономии на текущих затратах предприятия, который рассчитывается по формуле:

(4.12)

где Этек – экономия на текущих затратах при использовании программного средства, р.;

ΔЗтекп.с– прирост текущих затрат, связанных с использованием программного средства, р;

Нп    – ставка налога на прибыль согласно действующему законодательству.

Таким образом экономический эффект при использовании программного средства составит

## 4.4 Расчет показателей экономической эффективности разработки и реализации программного средства

Для организации-разработчика программного средства оценка экономической эффективности разработки осуществляется с помощью расчета рентабельности затрат на разработку программного средства. Рентабельность является одним из основных показателей эффективности предприятия с точки зрения использования привлеченных средств. Она представляет собой отношение суммы чистой приведенной прибыли, полученной за весь расчетный период, к суммарным приведенным затратам за этот же период и определяется по формуле:

(4.13)

где ΔПч ‒ прирост чистой прибыли, полученной от разработки программного средства организацией-разработчиком по индивидуальному заказу, р;

Зр ‒ затраты на разработку программного средства организацией-разработчиком, р.

Подставим результат вычисления в формулу (4.9) и произведем расчет Рз:

Рассчитанный показатель отображает, сколько чистой прибыли компания-разработчик получит от вложенных денег в разработку программного средства.

Так как сумма инвестиций больше суммы годового прироста чистой прибыли, для организации-заказчика рассчитывается несколько показателей экономической эффективности.

Для приведения доходов и затрат к настоящему моменту времени определяется коэффициент дисконтирования по формуле:

(4.14)

где *d* –   требуемая норма дисконта, которая по своему смыслу соответствует устанавливаемому инвестором желаемому уровню рентабельности инвестиций, доли единицы;

*t* – порядковый номер года, доходы и затраты которого приводятся к расчетному году;

*tp* – расчетный год, к которому приводятся доходы и инвестиционные затраты.

Норму дисконта принимаем равным ставке рефинансирования Национального банка Республики Беларусь – 9,5%. Расчетный период составит четыре года.

Таким образом, коэффициенты дисконтирования за каждый год составляют:

В течение первого года осуществляется разработка приложения, поэтому в первый год экономический эффект будет меньше планируемого. Для того, чтобы учесть этот факт, необходимо выяснить, сколько времени будет затрачено на разработку приложения.

Так как работа команды разработчиков осуществляется поэтапно, то затраченное время будет равно сумме трудоемкости работ команды, и составит 595 часов.

Таблица 4.3 – Расчет эффективности инвестиций (затрат) в реализацию проектного решения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Показатель | Значение расчетного периода по годам | | | |
| 1-й год | 2-й год | 3-й год | 4-й год |
| 1 Прирост чистой прибыли, р. | 12687,2 | 15859 | 15859 | 15859 |
|  |  |  |  |  |
| 2 Дисконтированный резуль­тат, р | 12687,2 | 14431,7 | 13163 | 12052,84 |
| 3 Инвестиции в разработку, р. | 19672 | 0 | 0 | 0 |
| 4 Дисконтированные инве­стиции, р. | 19672 | 0 | 0 | 0 |
| 5 Чистый дисконтированный доход по годам, р. | -6984,8 | 14431,7 | 13163 | 12052,84 |
| 6 Чистый дисконтированный доход нарастающим итого, р. | -6984,8 | 7446,9 | 20609,9 | 32662,74 |
| 7 Коэффициент дисконтиро­вания, доли единицы | 1,00 | 0,91 | 0,83 | 0,76 |

В данном случае дисконтированный эффект нарастающим итогом превысит дисконтированные инвестиции на второй год. Дисконтированный срок окупаемости рассчитывается по формуле:

(4.15)

Таким образом, дисконтированный срок окупаемости равен

Индекс доходности инвестиций рассчитывается по формуле:

(4.16)

Таким образом, индекс доходности инвестиций равен:

В результате проведения расчетов была определена необходимость разработки программного обеспечения, а также получен экономический эффект от использования данного программного продукта. По результатам проведенного экономического обоснования были получены следующие результаты:

1 Стоимость заказа на разработку программного средства автоматизации бэк-офисных процессов организации составила 26557,2 рублей.

2 Прирост чистой прибыли составил 15859 рублей.

3 Данная разработка имеет положительный экономический эффект в размере 80%.

4 По результатам проведенных расчетов, вложенные инвестиции должны окупится за 1,3 года, чистую прибыль стоит ожидать на второй год реализации проекта.

Таким образом, разработка и реализация по индивидуальному заказу приложения для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств экономической точки зрения целесообразна.

# 5 **Охрана труда. организация охраны труда на предприятии руп «Центр цифрового развития»**

Организация охраны труда на предприятии РУП "Центр цифрового развития" является важным элементом обеспечения безопасности и здоровья сотрудников, особенно в условиях активного использования компьютерной техники и офисного оборудования. В процессе разработки мобильного приложения для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств сотрудники сталкиваются с рядом опасных и вредных производственных факторов, которые могут негативно влиять на их здоровье и работоспособность. Основные нормативные документы, регулирующие вопросы охраны труда на предприятии, включают Инструкцию по охране труда при использовании офисного оборудования (№3), Инструкцию по оказанию первой (доврачебной) помощи пострадавшим (№7), Инструкцию по охране труда при работе с персональными компьютерами (№8), Инструкцию по пожарной безопасности (№9) и Инструкцию для ответственного по пожарной безопасности (№13).

Согласно Инструкции №3 и Инструкции №8, к основным опасным и вредным факторам, связанным с работой за компьютером и использованием офисного оборудования, относятся:

1 Повышенный уровень электромагнитных и ионизирующих излучений, которые могут оказывать негативное воздействие на нервную систему и общее состояние здоровья.

2 Статические и динамические перегрузки костно-мышечного аппарата, вызванные длительным пребыванием в одной позе и повторяющимися движениями.

3 Перенапряжение зрительного анализатора из-за длительной работы с экраном монитора, что может приводить к ухудшению зрения.

4 Повышенная яркость света и блесткость, которые вызывают усталость глаз и снижение концентрации внимания.

5 Эмоциональные перегрузки и монотонность труда, связанные с высокой умственной нагрузкой и выполнением однообразных задач.

Эти факторы могут приводить к хронической усталости, снижению продуктивности и увеличению риска профессиональных заболеваний. Для минимизации их воздействия Инструкция №8 устанавливает требования к организации рабочего места, включая оптимальное расстояние от экрана монитора до глаз (не менее 500 мм), регулировку высоты стула и спинки, а также обеспечение достаточной освещенности (300–500 люкс). Кроме того, предусмотрены регламентированные перерывы для снижения нагрузки на нервную систему и восстановления концентрации внимания.

Таким образом, существующая система охраны труда на предприятии направлена на создание безопасных условий работы, минимизацию воздействия вредных факторов и обеспечение готовности персонала к действиям в чрезвычайных ситуациях. Внедрение и соблюдение данных нормативов является важным условием для успешной реализации дипломного проекта и обеспечения здоровья сотрудников.

В процессе разработки мобильного приложения для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств на предприятии РУП "Центр цифрового развития" одним из наиболее значимых вредных факторов является умственное переутомление. Этот фактор связан с длительной концентрацией внимания, необходимостью обработки большого объема информации, а также монотонностью труда при написании кода, тестировании и отладке приложения. Умственное переутомление возникает в результате интенсивной интеллектуальной деятельности, которая требует постоянного сосредоточения, анализа данных и принятия решений. В контексте разработки программного обеспечения это проявляется в длительной работе за компьютером, связанной с написанием и отладкой кода, а также в необходимости постоянного анализа требований к приложению и их реализации. Монотонность задач, таких как тестирование и устранение ошибок, может вызывать снижение мотивации и усталость.

Умственное переутомление оказывает негативное воздействие на здоровье и работоспособность разработчиков. Основные последствия включают снижение концентрации внимания, что может привести к ошибкам в коде, увеличению времени на выполнение задач и снижению качества продукта. Длительная умственная нагрузка вызывает чувство усталости, что снижает продуктивность. Монотонность труда и высокие требования к качеству работы могут привести к стрессу и снижению интереса к профессиональной деятельности. Кроме того, умственное переутомление часто сопровождается проблемами со сном, что усугубляет общее состояние здоровья. В условиях разработки мобильного приложения умственное переутомление является одним из наиболее значимых вредных факторов. Это связано с тем, что процесс разработки требует высокой концентрации внимания и длительной работы за компьютером. Например, при написании сложных алгоритмов или отладке кода разработчики могут проводить за компьютером по 6-8 часов в день, что превышает рекомендуемые нормы для интенсивной умственной работы.

Максимальное значение умственного переутомления может быть оценено с помощью субъективных и объективных показателей. К субъективным показателям относятся жалобы разработчиков на усталость, снижение концентрации, головные боли. Объективные показатели включают увеличение количества ошибок в коде, снижение скорости выполнения задач, ухудшение качества работы. Согласно нормативным документам, таким как СанПиН 2.2.2/2.4.1340-03 и Трудовой кодекс Республики Беларусь, продолжительность работы за компьютером не должна превышать 4-6 часов в день для интенсивной умственной деятельности. При этом рекомендуется делать перерывы каждые 1,5-2 часа для снижения нагрузки на нервную систему и восстановления концентрации. В случае разработки мобильного приложения, где разработчики часто работают сверхурочно или без достаточных перерывов, уровень умственного переутомления может значительно превышать допустимые нормы. Это подтверждается жалобами на усталость и снижение продуктивности, что требует внедрения соответствующих мер для минимизации данного фактора [1].

Умственное переутомление, возникающее в процессе разработки мобильного приложения, требует строгого соблюдения нормативных требований, установленных в Республике Беларусь. Согласно Инструкции по охране труда при работе с персональными компьютерами (№8 от 16.10.2024), работа с ПК должна быть организована таким образом, чтобы минимизировать негативное воздействие на здоровье работников. В частности, инструкция регламентирует продолжительность работы за компьютером, частоту и длительность перерывов, а также требования к организации рабочего места. Для разработчиков программного обеспечения, чья деятельность относится к категории III (творческая работа в режиме диалога с ПК), суммарное время регламентированных перерывов при 8-часовой рабочей смене составляет 70 минут. Перерывы должны распределяться следующим образом: через 1,5–2 часа от начала рабочей смены и через 1,5–2 часа после обеденного перерыва продолжительностью 20 минут каждый или 15 минут через каждый час работы. Это позволяет снизить нагрузку на нервную систему, восстановить концентрацию внимания и предотвратить накопление усталости.

В процессе разработки мобильного приложения для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств на предприятии РУП "Центр цифрового развития" одним из наиболее значимых вредных факторов является умственное переутомление. Этот фактор связан с длительной концентрацией внимания, необходимостью обработки большого объема информации, а также монотонностью труда при написании кода, тестировании и отладке приложения. Умственное переутомление возникает в результате интенсивной интеллектуальной деятельности, которая требует постоянного сосредоточения, анализа данных и принятия решений. В контексте разработки программного обеспечения это проявляется в длительной работе за компьютером, связанной с написанием и отладкой кода, а также в необходимости постоянного анализа требований к приложению и их реализации. Монотонность задач, таких как тестирование и устранение ошибок, может вызывать снижение мотивации и усталость.

Умственное переутомление оказывает негативное воздействие на здоровье и работоспособность разработчиков. Основные последствия включают снижение концентрации внимания, что может привести к ошибкам в коде, увеличению времени на выполнение задач и снижению качества продукта. Длительная умственная нагрузка вызывает чувство усталости, что снижает продуктивность. Монотонность труда и высокие требования к качеству работы могут привести к стрессу и снижению интереса к профессиональной деятельности. Кроме того, умственное переутомление часто сопровождается проблемами со сном, что усугубляет общее состояние здоровья. В условиях разработки мобильного приложения умственное переутомление является одним из наиболее значимых вредных факторов. Это связано с тем, что процесс разработки требует высокой концентрации внимания и длительной работы за компьютером. Например, при написании сложных алгоритмов или отладке кода разработчики могут проводить за компьютером по 6-8 часов в день, что превышает рекомендуемые нормы для интенсивной умственной работы.

Максимальное значение умственного переутомления может быть оценено с помощью субъективных и объективных показателей. К субъективным показателям относятся жалобы разработчиков на усталость, снижение концентрации, головные боли. Объективные показатели включают увеличение количества ошибок в коде, снижение скорости выполнения задач, ухудшение качества работы. Согласно нормативным документам, таким как СанПиН 2.2.2/2.4.1340-03 и Трудовой кодекс Республики Беларусь, продолжительность работы за компьютером не должна превышать 4-6 часов в день для интенсивной умственной деятельности. При этом рекомендуется делать перерывы каждые 1,5-2 часа для снижения нагрузки на нервную систему и восстановления концентрации. В случае разработки мобильного приложения, где разработчики часто работают сверхурочно или без достаточных перерывов, уровень умственного переутомления может значительно превышать допустимые нормы. Это подтверждается жалобами на усталость и снижение продуктивности, что требует внедрения соответствующих мер для минимизации данного фактора.

Для снижения уровня умственного переутомления и обеспечения безопасности персонала при работе с персональными компьютерами необходимо внедрить комплекс мер, направленных на информирование и обучение сотрудников нормам и требованиям охраны труда. В соответствии с Инструкцией по охране труда при работе с ПК (№8 от 16.10.2024), а также с учетом Приложения 2, где приведены комплексы упражнений для глаз и гимнастических упражнений для снятия напряжения, основное внимание должно быть уделено повышению осведомленности разработчиков о правильной организации рабочего процесса и соблюдении регламентированных перерывов.

Первым шагом является проведение регулярного обучения персонала правилам работы с ПК. Это включает в себя ознакомление с требованиями Инструкции, в частности, с нормами продолжительности работы за компьютером, частотой и длительностью перерывов, а также с правилами организации рабочего места. Разработчики должны быть проинформированы о том, что несоблюдение этих норм может привести к серьезным последствиям для здоровья, таким как ухудшение зрения, хроническая усталость и снижение продуктивности. Обучение должно проводиться как в формате вводных инструктажей для новых сотрудников, так и в виде периодических семинаров для всего коллектива.

Важным элементом обучения является ознакомление персонала с комплексами упражнений для глаз и гимнастическими упражнениями, приведенными в Приложении 2 Инструкции. Эти упражнения направлены на снятие напряжения с глаз, улучшение кровообращения и расслабление мышц, что особенно важно при длительной работе за компьютером. Разработчики должны быть обучены выполнению этих упражнений во время регламентированных перерывов, что позволит снизить нагрузку на зрение и предотвратить накопление усталости. Для удобства сотрудников комплексы упражнений могут быть размещены на информационных стендах или в электронном виде на корпоративном портале.

Кроме того, необходимо внедрить систему контроля за соблюдением норм труда. Это может включать в себя регулярный мониторинг продолжительности работы за компьютером и соблюдения перерывов, а также проведение опросов сотрудников для выявления случаев несоблюдения установленных правил. В случае выявления нарушений следует проводить дополнительные разъяснительные беседы и, при необходимости, вводить меры дисциплинарного характера.

Таким образом, внедрение комплекса мер, направленных на информирование и обучение персонала, позволит снизить уровень умственного переутомления, обеспечить соблюдение норм охраны труда и повысить эффективность работы разработчиков. Это также будет способствовать созданию безопасных и комфортных условий труда, что положительно скажется на здоровье и мотивации сотрудников.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения проекта разработана информационная система для анализа рабочего времени и показателей производительности водителей крупногабаритных транспортных средств. Реализованное программное обеспечение предоставляет возможность эффективного контроля за соблюдением рабочего графика, анализа производительности и формирования отчетов для принятия управленческих решений.

Проведенный анализ предметной области подтвердил актуальность разработки, обусловленную потребностью транспортных компаний в автоматизированных решениях, обеспечивающих минимизацию человеческих ошибок, соблюдение нормативов труда и повышение прозрачности управления. Существующие системы мониторинга зачастую ограничены в функционале и не интегрируются с современными аналитическими инструментами, что создает необходимость в специализированном программном решении.

Архитектура разработанной системы включает серверную часть на основе Spring Boot, мобильное приложение для Android и базу данных PostgreSQL, что гарантирует гибкость, масштабируемость и надежность работы. Использование REST API позволяет обеспечивать обмен данными в режиме реального времени, а применение JWT-аутентификации и Redis-кэширования повышает безопасность и производительность системы.

Проведенные расчеты надежности по различным моделям подтвердили устойчивость программного обеспечения к отказам, а экономический анализ показал, что внедрение системы позволяет снизить затраты на управление водительским составом, повысить эффективность использования ресурсов и обеспечить возврат инвестиций в течение 1,3 года.

Система готова к внедрению и дальнейшему расширению. В перспективе возможно добавление интеграции с биометрическими датчиками для анализа состояния водителей, а также внедрение алгоритмов машинного обучения для прогнозирования утомляемости и оптимизации маршрутов. Таким образом, разработка подтверждает свою практическую значимость и потенциал для масштабирования, обеспечивая современные решения для управления логистическими процессами.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Научный студенческий форум — статья об анализе усталости водителей. [Электронный ресурс]. – Режим доступа: scienceforum.ru. – Дата доступа: 20.02.2025.

[2] ISO 45001:2018 - Occupational Health and Safety Management Systems. Requirements. [Электронный ресурс]. – Режим доступа: https://www.iso.org/standard/63787.html. – Дата доступа: 15.12.2024.

Закон Республики Беларусь 5 января 2016 г. № 354-З О промышленной безопасности [Электронный ресурс]. – Режим доступа: https://pravo.by/document/?guid=12551&p0=H11600354. – Дата доступа: 15.12.2024.

[3] Toggl Track [Электронный ресурс]. – Режим доступа: https://toggl.com/. – Дата доступа: 23.02.2025.

[4] Toggl Track pricing [Электронный ресурс]. – Режим доступа: https://toggl.com/track/pricing/. – Дата доступа: 23.02.2025.

[5] 1C:Предприятие. TMS [Электронный ресурс]. – Режим доступа: https://solutions.1c.ru/catalog/tms/features. – Дата доступа: 23.02.2025.

[6] Fleetio app [Электронный ресурс]. – Режим доступа: https://www.fleetio.com/. – Дата доступа: 23.02.2025.

[7] Fleetio app pricing [Электронный ресурс]. – Режим доступа: https://www.fleetio.com/pricing. – Дата доступа: 23.02.2025.

[8] Drivers diary app [Электронный ресурс]. – Режим доступа: https://apps.apple.com/us/app/driver-diary/id396984974. – Дата доступа: 23.02.2025.

[9] Hubstaff systems [Электронный ресурс]. – Режим доступа: https://hubstaff.com/. – Дата доступа: 23.02.2025.

[10] Сайт Java [Электронный ресурс]. – Режим доступа: https://www.java.com/ru/. – Дата доступа: 25.02.2025.

[11] Сайт Spring [Электронный ресурс]. – Режим доступа: https://spring.io/. – Дата доступа: 25.02.2025.

[12] Сайт Android [Электронный ресурс]. – Режим доступа: https://www.android.com/. – Дата доступа: 25.02.2025.

[13] Material Design [Электронный ресурс]. – Режим доступа: https://m3.material.io/. – Дата доступа: 25.02.2025.

[14] OpenStreetMap API [Электронный ресурс]. – Режим доступа: https://wiki.openstreetmap.org/wiki/API. – Дата доступа: 25.02.2025.

[15] Сайт PostgreSQL [Электронный ресурс]. – Режим доступа: https://www.postgresql.org/. – Дата доступа: 25.02.2025.

[16] What is OAuth 2.0? [Электронный ресурс]. – Режим доступа: https://auth0.com/intro-to-iam/what-is-oauth-2. – Дата доступа: 25.02.2025.

[17] JSON Web Tokens [Электронный ресурс]. – Режим доступа: https://auth0.com/docs/secure/tokens/json-web-tokens. – Дата доступа: 25.02.2025.

[18] Redis database [Электронный ресурс]. – Режим доступа: https://redis.io/. – Дата доступа: 25.02.2025.

[19] Реализация OAuth2 на Spring Security [Электронный ресурс]. – Режим доступа: https://docs.spring.io/spring-security/reference/servlet/oauth2/index.html. – Дата доступа: 25.02.2025.

[20] Rest api examples [Электронный ресурс]. – Режим доступа: https://restful-api.dev/. – Дата доступа: 25.02.2025.

[21] Буч, Г. Язык UML. Руководство пользователя / Г. Буч, Дж Рамбо, И. Якобсон. – Москва : ДМК-Пресс, 2015. – 127-200 с.

Тулякова, Т. В. Основы проектирования систем менеджмента безопасности / Т. В. Тулякова, Е. В. Крюкова. – Москва : Издательство ЛАНЬ, 2010. – 91-94 с.