ФЕДЕРАЛЬНОЕ АГЕНТСТВО МОРСКОГО И РЕЧНОГО ТРАНСПОРТА

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Государственный университет морского и речного флота имени адмирала С.О. Макарова»
(ФГБОУ ВО «ГУМРФ имени адмирала С.О. Макарова»)
Воронежский филиал

Кафедра математики, информационных систем и технологий
Направление подготовки 09.03.02 Информационные системы и технологии
(код, наименование направления подготовки/специальности)
Форма обучения очная

«К ЗАЩИТЕ ДОПУЩЕН(А)»
Заведующий кафедрой
_____
(подпись)
Черняева С. Н.
(ФИО)
_____ 20___

Выпускная квалификационная работа

Обучающегося Уланова Владислава Евгеньевича
(фамилия, имя, отчество)
Вид работы выпускная квалификационная работа бакалавра
(выпускная квалификационная работа бакалавра, специалиста, магистра)

Пояснительная записка

Тема Разработка приложения оптимизации транспортных потоков доставки продукции (на примере ГБУ ГЦ «Юго-Западный»)
(полное название темы квалификационной работы, в соответствии с приказом об утверждении тематики ВКР)

Руководитель работы к.т.н., доцент Матыцина И. А.
(должность, подпись, фамилия, инициалы, дата)
Консультант _____
(при наличии) (должность, подпись, фамилия, инициалы, дата)
Консультант _____
(должность, подпись, фамилия, инициалы, дата)
Обучающийся Уланов В. Е.
(подпись, фамилия, инициалы, дата)

Воронеж
2024

ФЕДЕРАЛЬНОЕ АГЕНТСТВО МОРСКОГО И РЕЧНОГО ТРАНСПОРТА

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Государственный университет морского и речного флота имени адмирала С.О. Макарова»
(ФГБОУ ВО «ГУМРФ имени адмирала С.О. Макарова»)
Воронежский филиал

Кафедра математики, информационных систем и технологий
Направление подготовки 09.03.02 Информационные системы и технологии
(код, наименование направления подготовки/специальности)
Форма обучения очная

УТВЕРЖДАЮ
Заведующий кафедрой
_____
(подпись)
Черняева С. Н.
(ФИО)
_____ 2024

Задание
на выпускную квалификационную работу

Вид работы ВКР бакалавра
(ВКР бакалавра, ВКР специалиста, ВКР магистра)

Обучающемуся Уланову Владиславу Евгеньевичу
(фамилия, имя, отчество)

Тема Разработка приложения оптимизации транспортных потоков доставки продукции (на примере ГБУ ГЦ «Юго-Западный»)

Утверждена приказом ректора университета от _____ 20___, № _____

Срок сдачи законченной работы _____ 20___

Исходные данные (или цель ВКР):
Целью данной дипломной работы является разработка информационной подсистемы оптимизации транспортных потоков доставки продукции.

Перечень подлежащих исследованию, разработке, проектированию вопросов (краткое содержание ВКР):
(актуальность темы, цели и задачи ВКР; аналитический обзор литературных источников; постановка задачи исследования, разработки, проектирования; содержание процедуры исследования, разработки, проектирования; обсуждение результатов; дополнительные вопросы, подлежащие разработке; заключение – выводы по работе в целом, оценка степени решения поставленных задач, практические рекомендации; и др.)

– Введение. Актуальность выбранной темы, цель и задачи ВКР

(наименование вопроса, раздела и его краткое содержание)

– Исследовательски йраздел. Задачи коммивояжера и их вариации. Алгоритмы оптимизации маршрутов. Современные подходы и технологии в оптимизации логистических маршрутов. Обзор существующих программных решений.

(наименование вопроса, раздела и его краткое содержание)

– Проектный раздел. Общая архитектура системы. Модели данных. Взаимодействие компонентов системы. Выбор технологий и инструментов разработки. Реализация приложения. Описание среды разработки. Разработка пользовательского интерфейса. Реализация алгоритмов оптимизации маршрутов. Интеграция с внешними сервисами и базами данных. Тестирование приложения

(наименование вопроса, раздела и его краткое содержание)

– Заключение. Выводы по работе в целом. Оценка степени решения поставленных задач

(наименование вопроса, раздела и его краткое содержание)

Практические рекомендации

Перечень графического материала (или презентационного материала):

1. Титульный лист
2. Цель и задачи ВКР
3. Описание предметной области
4. Проектирование системы
5. Проектирование системы (продолжение)
6. Разработка системы
7. Результаты ВКР

Консультанты по разделам ВКР (при наличии):

1. _____

(наименование раздела, ученая степень, ученое звание и должность, ФИО консультанта)

2. _____

(наименование раздела, ученая степень, ученое звание и должность, ФИО консультанта)

3. _____

(наименование раздела, ученая степень, ученое звание и должность, ФИО консультанта)

Дата выдачи задания: _____ 20____

Задание согласовано и принято к исполнению: _____ 20____

Руководитель ВКР: к.т.н., доцент Матыцина Ирина Александровна _____

(должность, ученая степень, ученое звание, ФИО) (подпись)

Обучающийся: ИТ-4-1 Уланов Владислав Евгеньевич _____

(учебная группа, ФИО) (подпись)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Логистика - ключ к эффективности в мировой экономике, гарантирует скорость и продуктивность перемещений товаров, услуг. Эра растущего соперничества, глобального взаимодействия заставляет отрасль стремиться к более высокой работоспособности процедур. Создание идеальных сценариев перемещения – это шаг к уменьшению расходов на перевозку, к сокращению сроков доставки, к полноте обслуживания клиентов.

Технологический прогресс, увеличение грузооборота требуют новых программ, выбирающих наилучшие пути для доставки. Опыт, интуиция логистов уже не всегда справляются из-за сложной инфраструктуры транспорта, разностью влияющих элементов, как состояние дорог, сроки, загруженность транспорта.

Современные оптимизационные алгоритмы, методики, подкрепленные программным обеспечением, трансформируют и обогащают планирование маршрутов. Такое ПО становится необходимым инструментом, помогает сократить затраты, укрепить маркетинговую позицию компании в острой конкурентной борьбе.

Изучение разработки программ для определения оптимальных маршрутов доставки является сложной, но критически значимой задачей для логистики, влияя на повышение управленческой эффективности. Целью данной дипломной работы является создание приложения, обеспечивающего глубокий анализ различных факторов и эффективное распределение ресурсов. В процессе предстоит изучить и оценить существующие методики и алгоритмы для выбора наиболее подходящих для повышения точности и скорости планирования потоков.

Особое внимание при разработке уделяется нуждам пользователей и интуитивности интерфейса, что ведет к созданию удобной системы взаимодействия. Глубокий анализ программной структуры, в том числе архитектуры баз данных и связей между компонентами приложения,

является важным для обеспечения стабильности системы. Отбор инструментов и оборудования для поддержания эффективной работы системы при высоких нагрузках играет ключевую роль.

В фокусе исследования – разработка приложения с интуитивным интерфейсом и включение алгоритмов для улучшения маршрутизации, учитывающих разные ограничения и переменные. Планируется интеграция с различными внешними сервисами и базами данных, расширяя функциональные возможности системы. Основной фазой проекта является детальное тестирование всех компонентов приложения, что гарантирует их безошибочную работу и соответствие предъявленным требованиям. Такой методологический подход направлен на достижение цели - создание высококачественного продукта, способствующего эффективной оптимизации логистических процессов.

Изучаемым объектом в рамках дипломного проекта выступает механизм разработки и применения оптимальных маршрутов в условиях логистических организаций. Акцент ставится на всестороннем анализе начиная с первичных потребностей заказчиков и заканчивая контролем за реализацией маршрутов, что способствует повышению общей эффективности процессов.

Изучаемый объект - создание приложения для автоматизации и улучшения процесса составления маршрутов доставки, учитывающее различные аспекты: состояния дорог, времени доставки, грузоподъемности транспорта, пожеланий заказчиков и стандартов обслуживания.

Основная цель исследования - разработка инструмента, повышающего эффективность и качество планирования маршрутов, что позволит логистическим компаниям уменьшить расходы на перевозки, сократить сроки доставки и повысить уровень сервиса.

Для достижения цели предусматривается изучение разнообразных методов и алгоритмов для оптимизации маршрутов, анализ требований

пользователей к приложению, акцент на проектировании его архитектуры и реализации с помощью новейших технологий программирования.

В исследовании рассматривается задача анализа потенциальных рисков и разработки методик для их снижения, что необходимо для успешной реализации и функционирования создаваемого приложения. Исследуемый объект включает в себя комплексную систему логистических процедур, объемлющую разработку маршрутов доставки, организацию складских процессов, управление транспортными средствами, поддержание стандартов обслуживания и прочее. Целью разработки приложения является оптимизация важнейшего аспекта логистики — эффективного планирования маршрутов. Так, предмет и объект исследования тесно переплетены и формируют интегральную структуру, в рамках которой осуществляются разнообразные действия и взаимодействия. Создание программного обеспечения для определения наилучших путей доставки грузов представляется эффективным методом повышения эффективности логистических процессов, укрепления позиций логистических фирм на рынке. Для достижения целей исследования будут применены разнообразные методы анализа, совмещающие теоретические знания и практический опыт, что позволит получить всестороннее и достоверное знание об исследуемом предмете.

Аналитический подход играет ключевую роль в изучении методик и алгоритмов для эффективного маршрутизирования доставок, обеспечивая анализ применимости тех или иных решений в реальности и отбор оптимальных для интеграции в создаваемое приложение. Дополнительно, использование методов моделирования и проектирования критично для конструирования архитектуры приложения, определения его структурных элементов и установления между ними связей, что способствует гибкости, масштабируемости и удобству последующего обслуживания системы.

В процессе разработки немаловажна роль методов программирования и инженерии ПО, охватывающих выбор языков программирования,

фреймворков, библиотек и разработку качественного кода с применением современных технологий и практик. Важным этапом является также тестирование, в котором применяются разнообразные методики, в том числе модульное, интеграционное и пользовательское тестирование, обеспечивающие проверку надежности компонентов системы и их бесперебойную функциональность.

В рамках данной дипломной работы предвидится достижение конкретных итогов, которые способствуют реализации поставленных задач и адресации вопросов исследования.

Основным ожидаемым итогом станет создание функционального и легко управляемого приложения для определения наилучшего пути перевозки грузов. Предполагается, что приложение будет включать в себя множество функциональных возможностей, таких как учет множества параметров, оптимизация маршрутов и предоставление подробных сведений о доставке.

Дополнительным результатом станет разработка сопутствующей документации для приложения, в которой будет содержаться всестороннее описание его функций, архитектурных решений, использованных технологий и подходов к оптимизации. Это даст возможность другим специалистам изучить разработанный продукт и при необходимости внести в него корректировки или улучшения.

Также в числе ожидаемых итогов - выполнение тестирования созданного приложения и получение подтверждений его безошибочной и стабильной работы. Это предусматривает проведение как функционального тестирования, так и проверки эффективности и надежности программы в разнообразных условиях эксплуатации.

1. ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ

В разделе "Теоретические основы" рассмотрены ключевые идеи и подходы, критические для создания приложения, направленного на вычисление наилучшего пути доставки грузов. Приводится обобщение по логистике, анализируются способы усовершенствования маршрутизации и перечисляются технологии, актуальные для указанной сферы.

1.1. Задачи коммивояжера и их вариации

Задача коммивояжера, относящаяся к области комбинаторной оптимизации, представляет поиск наименее длинного пути, который позволяет посетить каждый город из заданного набора ровно однажды и вернуться в исходную точку. Эта проблема акцентирует внимание на минимизации общего пройденного расстояния или времени путешествия, исходя из известных дистанций между всеми парами городов.

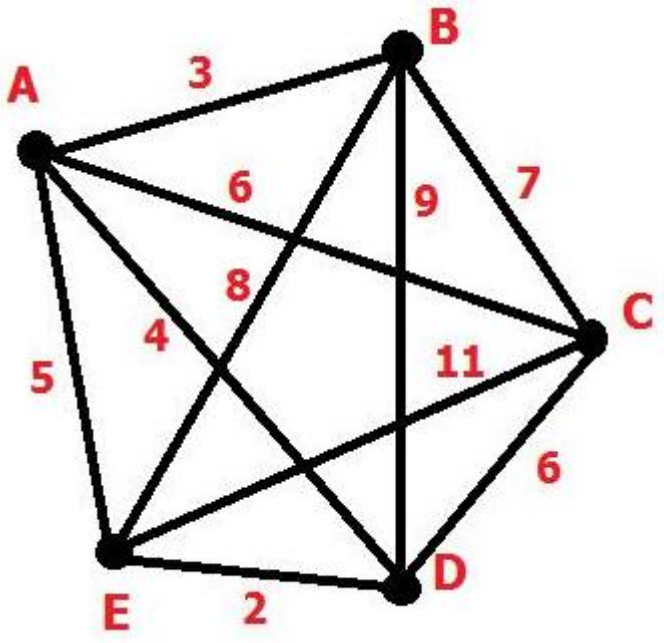

Рисунок 1.1 – Пример задачи коммивояжера

Задача коммивояжера находит широкое применение в различных областях, включая организацию доставки, планирование маршрутов в телекоммуникационных сетях и оптимизацию процессов производства. Ее суть сводится к поиску наиболее эффективного пути среди заданного набора пунктов, минимизируя затраты или пройденное расстояние.

Уникальность задачи заключается в ее высокой вычислительной сложности: с ростом числа пунктов, время, требуемое для вычисления идеального маршрута, увеличивается экспоненциально, классифицируя ее как NP-полную задачу. Это означает, что нахождение абсолютно точного решения в больших масштабах оказывается весьма затруднительным.

Вместе с тем, существуют различные приблизительные методы, которые обеспечивают достаточно хорошие результаты в практически приемлемые сроки. Среди них - полный перебор, муравьиные и генетические алгоритмы, методы динамического программирования. Эти подходы играют ключевую роль в обработке и решении задачи коммивояжера в современных условиях.

Исследование задачи коммивояжёра и её специфических разновидностей актуально в контексте разработки дипломного проекта, направленного на создание приложения для оптимизации маршрутов грузоперевозок. Вариативность этой задачи, включая временные ограничения, лимиты вместимости транспортных средств и сценарии с множественными исполнителями, требует индивидуализированного подхода к выбору и адаптации алгоритмов для достижения наилучших результатов. Такое исследование обеспечивает ключевую теоретическую основу, важную для последующего этапа - внедрения алгоритмов в программное обеспечение. Оно делает возможным глубокое понимание принципов и методик, подходящих для решения различных спецзадач в сфере логистики и транспорта, и помогает учитывать все необходимые дополнительные условия и ограничения при формировании эффективных маршрутов доставки.

Изучение коммивояжера и его вариаций играет ключевую роль в разработке дипломного проекта, обеспечивая тщательный подход к выбору методов оптимизации. Это способствует созданию эффективного инструмента для решения задач логистики и управления транспортом.

1.2. Алгоритмы оптимизации маршрутов

Работа "Алгоритмы оптимизации маршрутов" фокусируется на изучении разнообразных стратегий и методик, применяемых для усовершенствования процедур доставки. Ключевое внимание уделяется набору методов – от проверенных временем алгоритмов поиска эффективного пути до современных эвристических решений, нацеленных на улучшение маршрутизации в транспортной сфере.

Примером классической техники оптимизации является алгоритм Дейкстры, назначение которого – обнаружение наиболее короткого пути между узлами в графе. Эта методика находит широкое применение в составлении маршрутов транспортных и логистических сетей.

В числе распространенных подходов выделяется также алгоритм A* (A-star), в основе которого лежит сочетание эвристических оценок с поиском оптимального пути. Благодаря своей эффективности в обработке объемных графов, данный алгоритм занимает важное место в сферах планирования логистики и управления транспортными потоками.

Разнообразие эвристических подходов, включая генетические и муравьиные алгоритмы, открывает перспективы для нахождения приблизительных решений задач оптимизации маршрутов при ограничениях по времени или ресурсам. Исследование и применение этих методик в создании приложений для планирования оптимального пути доставки грузов позволит определить наиболее эффективные стратегии для специфических задач, способствуя улучшению работы приложений в реальных условиях. Помимо традиционных подходов к оптимизации, таких как алгоритмы

Дейкстры и A*, целесообразно также уделить внимание новейшим методам, в том числе алгоритмам машинного обучения. Применение технологий машинного обучения, например нейронных сетей и методов глубокого обучения, для анализа обширных наборов данных дает возможность прогнозирования оптимальных маршрутов на основании предыдущих данных о доставках.

Применение техник машинного обучения для прогнозирования уровня загруженности дорог позволяет сформировать наилучший маршрут, учитывая различные параметры: скорость передвижения, время дня, погодные условия. Оптимальный путь предоставляется на основе актуальной информации.

Облачные вычисления и распределенные системы также играют значимую роль в сфере оптимизации маршрутов. Задачи по поиску путей решаются силами разных узлов сети, что дает преимущество в обработке большого массива данных без задержек.

Изучение передовых методов усовершенствует классические подходы и способствует созданию адаптивных, высокоэффективных методик для вычисления маршрутов, что особенно актуально для логистической сферы.

1.2.1. Жадные алгоритмы

Методы жадной оптимизации относятся к категории эффективных стратегий для вычисления оптимальных путей в разнообразных задачах [4]. Основой данных методов служит принцип "жадности", когда на каждой отдельной стадии принимается решение, наилучшим образом подходящее в данной ситуации, что предполагается привести к наилучшему итогу в глобальном масштабе.

Примером жадного метода, применяемого для определения маршрутов, является алгоритм, основанный на выборе ближайшего пункта назначения. Этот метод заключается в последовательном выборе ближайшего узла

маршрута на каждом этапе его формирования. Несмотря на то что данный подход может обеспечить удовлетворительные результаты при работе с ограниченными объемами данных, он не предоставляет гарантии идентификации абсолютно оптимального маршрута и может закончиться созданием маршрутов с суб-оптимальными параметрами.

Жадные методы активно применяются для решения прикладных задач оптимизации маршрутов благодаря своей простоте и скорости выполнения. Вместе с тем их эффективность сильно зависит от особенностей решаемой задачи и может оказаться несостоятельной при увеличении количества пунктов или наличии дополнительных условий.

Осознание механизмов функционирования жадных методов критически важно для определения их пригодности в различных условиях и их совместного использования с другими оптимизационными подходами при стремлении к достижению наиболее высоких показателей эффективности.

1.2.2. Алгоритмы на основе теории графов

Алгоритмы, базирующиеся на теории графов, являются ключевым инструментом для усовершенствования логистики доставки. Представляя собой графическую схему городов и связей между ними, они обеспечивают оптимальное решение задач выбора пути. В числе применяемых алгоритмов выделяется алгоритм Дейкстры, который определяет наиболее короткий маршрут от одной точки к другим, способствуя тем самым повышению эффективности доставки на основе точных измерений расстояний между населенными пунктами.

Алгоритм A* представляет собой значимый инструмент в категории методик решения задач по поиску оптимального пути, умело сочетая эвристическую стратегию с традиционными подходами к поиску кратчайшего пути, что демонстрирует его высокую эффективность в контексте обширных графов. Теория графов, в свою очередь, предлагает

широкий спектр инструментов для улучшения процессов планирования доставки, включая, к примеру, использование алгоритмов для создания минимального остовного дерева, способствующих минимизации расходов при организации доставки через комплексные дорожные сети. Таким образом, внедрение алгоритмических подходов, основанных на теории графов, играет ключевую роль в оптимизации логистических и транспортных операций.

1.2.3. Метод ветвей и границ

Метод ветвей и границ представляет собой эффективный алгоритм для решения задач оптимизации в области комбинаторики, в том числе и для определения наиболее выгодного пути доставки товаров. Принцип работы метода заключается в систематическом разделении исходной задачи на упрощенные подзадачи, далее происходит подробный перебор потенциальных решений. В ходе реализации метода, задача разделяется на более мелкие сегменты - подзадачи, для каждой из которых оценивается минимально возможный показатель оптимального решения. Исключение из рассмотрения ветвей, чьи минимальные показатели превышают значение уже найденного оптимального решения, или тех, которые не способны способствовать его улучшению, позволяет сократить время и ресурсы, затрачиваемые на поиск наилучшего варианта.

Метод ветвей и границ, применяемый к решению задачи оптимизации маршрутизации доставки, позволяет систематизировать поиск наилучшего пути через анализ всех потенциальных маршрутов. На каждом этапе анализа происходит выбор следующего пункта назначения, при этом для каждой возможности рассчитывается минимально возможная стоимость маршрута. Варианты, стоимость которых превышает уже найденное оптимальное решение, исключаются из дальнейшего рассмотрения. Основное преимущество этого метода заключается в его способности эффективно

справляться с заданиями, которые имеют большое пространство для поиска и множество ограничений. Впрочем, значительная вычислительная сложность при работе с объемными данными и необходимость тщательного выбора оценок для минимальных границ могут стать препятствием для его практического использования.

1.2.4. Методы динамического программирования

Динамическое программирование играет центральную роль в оптимизационных задачах, включая вычисление оптимальных путей доставки. Методика разбиения крупных вопросов на меньшие части облегчает процесс нахождения решения за счет уменьшения необходимых вычислений.

Особое место занимает алгоритм Флойда-Уоршелла, который определяет минимальные расстояния между всеми вершинами в взвешенном графе. Эта особенность ценится при планировании маршрутов, поскольку позволяет учитывать различные варианты движения по сети.

Также значим алгоритм Беллмана-Форда, который нацелен на поиск кратчайших путей от одной вершины к остальным, допуская отрицательные веса ребер. Эта особенность делает его незаменимым в адаптации путей доставки с учетом разнообразных ограничений и предпочтений, способствуя прогрессу в логистических системах.

Динамическое программирование выделяется среди алгоритмов оптимизации. Его эффективность обусловлена способностью идентифицировать оптимальные пути в условиях множества альтернатив и наличия ограничений. Сильная сторона метода – детальный анализ каждого возможного варианта и оценка связанных с ним затрат. Тем не менее, следует учитывать, что при работе с обширными данными нагрузка на временные и вычислительные ресурсы может быть существенной.

1.3. Современные подходы и технологии в оптимизации логистических маршрутов

Оптимизация маршрутов и логистических процессов — ключ к повышению эффективности распределения. В эпоху быстро эволюционирующих цифровых технологий и аналитических методов открываются передовые возможности для улучшения логистической системы. Особую ценность приобретают инновационные методы и инструменты, направленные на оптимизацию логистических маршрутов.

Нельзя переоценить значимость информационного анализа в процессе пересмотра логистических маршрутов. Сегодня логистика активно использует аналитические методы для создания стратегии. Благодаря передовым технологиям сбора и обработки данных, специалисты получают доступ к данным о доставке, построении маршрутов и запросах потребителей.

Аналитические методы играют ключевую роль в определении трендов, создании прогнозов и учёте разнообразных факторов. Возможность анализировать эти данные предоставляет компаниям основу для разработки рациональных стратегий оптимизации маршрутов

Применение искусственного интеллекта и машинного обучения в логистике приводит к значительным изменениям. “Глубокий анализ данных помогает выбирать оптимальные маршруты, принимая во внимание погодные условия, качество дорог и интенсивность движения” [9]. В результате сокращается время и снижаются издержки на транспортировку.

“Интернет вещей и интеллектуальные устройства оказывают значительное влияние на логистику” [7]. “Транспортные средства и контейнеры, оснащённые датчиками, осуществляют мониторинг грузов, отслеживая температуру и уровень влажности” [7].

“Применение методов, основанных на принципах агентных систем, играет ключевую роль в модернизации и повышении эффективности

логистических процессов” [4]. Эти инновационные решения могут адаптироваться к изменяющимся обстоятельствам. Они обеспечивают всесторонний анализ различных аспектов, включая текущую ситуацию на дорогах, объёмы транспортируемых товаров и расположение ключевых логистических узлов. “Благодаря обработке данных в реальном времени, агентные системы и оптимизационные алгоритмы способствуют правильному распределению грузов, выбору наиболее подходящих маршрутов и рациональному использованию ресурсов” [5].

“В наше время, инновации, основанные на последних научных достижениях, значительно меняют логистику в бизнес-сфере” [6]. Улучшение способов доставки ускоряет процессы и повышает их эффективность. Оптимизация маршрутов перевозок снижает логистические затраты, и улучшает распределение ресурсов.

Развитие прогнозного анализа усиливает конкурентоспособность компаний, акцентируя внимание на использовании инновационных инструментов, для адаптации логистических стратегий к изменчивости рынка.

“Такие компании, как Amazon, наглядно демонстрируют преимущества интеграции инноваций в логистику, поскольку они успешно применяют новые технологии и подходы для оптимизации своей деятельности” [4]. Внедрение систем машинного обучения для анализа цепочек поставок повысило качество процедур упаковки и распределения продукции, что сократило временные затраты и улучшило сервис. Компания UPS, используя возможности Интернета вещей, добилась точности в отслеживании грузов, чем улучшила управление доставками и минимизировала возможные задержки.

Эти практические примеры лишь подтверждают значимость новаторских наработок в сфере логистики, подчеркивая, как значительно они могут способствует процветанию отрасли. Многообещающие направления, включающие современные стратегии и технологические инновации, без

сомнения, играют ключевую роль в эволюции логистической индустрии и открывают для компаний возможности на пути к сокращению издержек, ускорению процессов и повышению стандартов обслуживания.

Применение статистики, искусственных интеллектуальных систем, инноваций Интернета вещей и методологий на основе агентного подхода — всё это способствует созданию эластичных схем планирования, эффективному использованию ресурсов и беспрепятственной адаптации к изменениям на рынке. Прогресс в области технологий, направленных на совершенствование и повышение эффективности логистических процессов, продолжит оставаться ключевым фактором для сохранения и укрепления конкурентного положения компаний, а также будет способствовать их развитию в условиях постоянно изменяющегося рынка.

1.4. Обзор существующих программных решений

Изучение программ, предназначенных для совершенствования логистических путей, занимает ведущую позицию в процессе создания передового приложения. Понимание ассортимента доступных на рынке инструментов и возможностей позволяет выделить сильные и слабые стороны различных подходов, а также наметить возможности для дальнейшего совершенствования и инноваций.

Анализ ведущих программных продуктов

На рынке представлено много программных продуктов, направленных на оптимизацию логистических маршрутов. В числе лидеров можно отметить такие системы, как Route4Me, OptimoRoute и Vanguard System. Эти программные решения предоставляют широкий спектр функциональных возможностей, включая оптимизацию маршрутов с учетом разнообразных параметров (время, дистанция, затраты), учитывая ограничения (например, грузоподъемность транспорта) и обеспечивая интеграцию с другими системами, такими как системы мониторинга грузов, бухгалтерские системы.

Исследование ключевых атрибутов программных систем несет высокую значимость при их оценивании. Аналитик должен рассмотреть ряд параметров: от функционала и эффективности до стоимости и настраиваемости. К примеру, определенные программные продукты выделяются обширным спектром опций, однако могут быть более затратными и сложными в настройке. Выбор оптимального варианта требует сопоставления предложенных функций с уникальными нуждами организации.

Преимущества нынешних программных решений многочисленны: от скорости внедрения и разнообразия функций до высокой стабильности и производительности. Облачные технологии, представленные многими системами, расширяют доступность данных, обеспечивая их доступ из любой точки мира. В то же время, стоимость лицензий и технической поддержки, ограниченная адаптивность и проблемы интеграции с другими платформами могут выступать как существенные недостатки.

В настоящее время наблюдается активное развитие технологий, что приводит к изменению запросов на рынке, особенно заметно это в сфере оптимизации логистических маршрутов. Среди прогрессивных направлений особое внимание уделяется повышению автоматизации и самообучающихся возможностей систем. Также акцентируется внимание на расширении функционала по интеграции с различными системами, включая управление складами и отслеживание перемещения грузов. Особую роль играет разработка облачных сервисов, способствующих повышению доступности сервисов и их масштабированию.

2. ПРОЕКТНЫЙ РАЗДЕЛ

Проектирование архитектуры приложения является ключевым аспектом в разработке эффективного программного продукта. В данном контексте акцентируется внимание на принципах и подходах к созданию архитектуры для усовершенствования маршрута доставки груза. Анализируются разнообразные архитектурные решения, освещаются ключевые элементы и модули приложения, а также рассматриваются способы их взаимодействия. Цель — разработать простую, но при этом масштабируемую архитектуру, отвечающую требованиям проекта и обеспечивающую его эффективность. Привлекается внимание к вопросам безопасности, управлению данными и интеграции с внешними сервисами, что способствует созданию надежного приложения. Данная работа подчеркивает, что проектирование архитектуры — это фундаментальный этап, определяющий успешность и конкурентоспособность программного продукта.

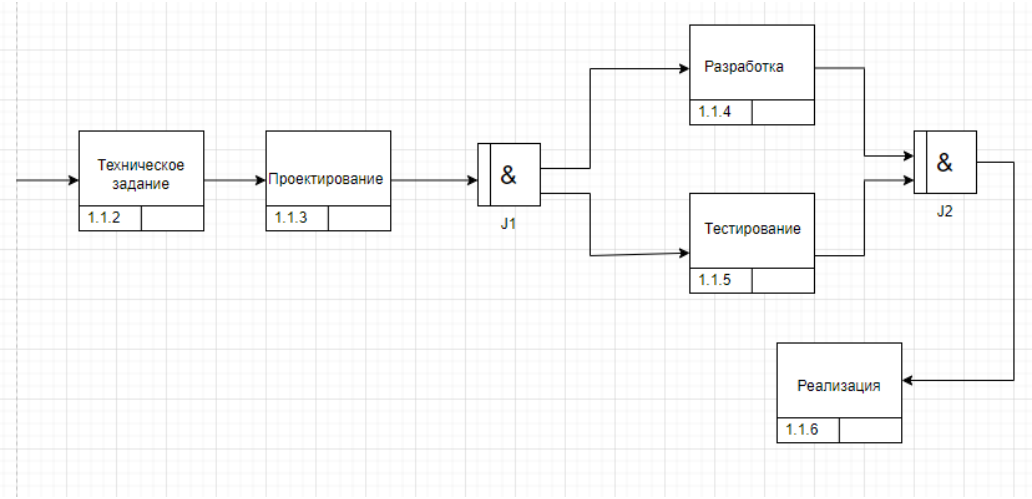

Рисунок 2.1 – Диаграмма IDF3

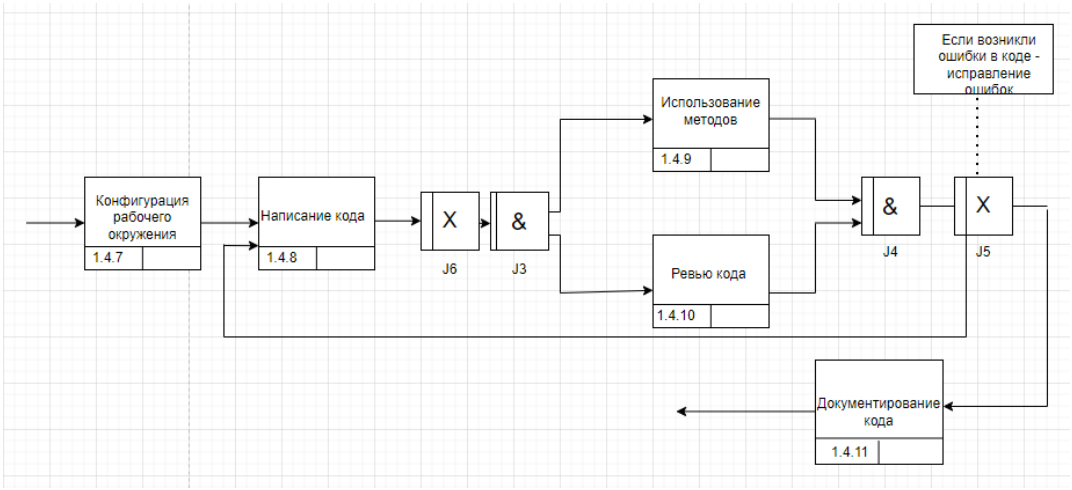

Рисунок 2.2 – Декомпозиция диаграммы IDF3

2.1. Общая архитектура системы

Сердцем данного приложения является его архитектура, которая служит каркасом для создания системы, цель которой — нахождение наилучших путей доставки товаров. Здесь рассматриваются основные элементы и методы проектирования, которые задают как структуру, так и способы взаимодействия всех частей приложения между собой.

Ключевые составляющие архитектуры включают в себя:

Интерфейс пользователя. Этот фрагмент является мостом между приложением и его использованием человеком. Через пользовательский интерфейс осуществляется ввод информации о пунктах назначения и выводятся результаты оптимизации маршрутов. Задача состоит в том, чтобы сделать этот процесс максимально простым и понятным для каждого пользователя, облегчая его взаимодействие с системой.

Бизнес-логика. Эта часть отвечает за обработку поступающей информации и принятие решений на основе установленных бизнес-правил. В рамках бизнес-логики происходит анализ данных, вычисление оптимальных путей для доставки, учитывая различные параметры (время, дистанция, затраты) и подготовка советов для пользователя.

Специализированное хранилище данных является ключевым элементом для сохранения всех необходимых сведений. В рамках данного хранилища производится регистрация информации о местах доставки, транспортных единицах и результатах совершенствования маршрутов. Приоритет отводится гарантированию надежности сохранения данных с возможностью быстрого доступа к ним.

Использование сервисов и API обеспечивает синергию компонентов в приложении [15]. Применяемые элементы содействуют выполнению операций с данными, доступу к ним, а также привносят интегративный компонент с внешними сервисами и системами.

Выбор структурного паттерна системы оказывает влияние на организацию и кооперацию ее компонентов. В соответствии с конкретными задачами и требованиями к расширяемости, гибкости и скорости работы приложения, возможно использование сложной многоуровневой структуры, структуры типа клиент-сервер или системы, построенной на основе микросервисов.

Общий дизайн системы внимательно подходит к аспектам безопасности, расширяемости и производительности. Это охватывает выбор подходов к проверке подлинности и правам доступа пользователей, защите информации и улучшению работы с базами данных. Структура и взаимосвязь всех элементов приложения, заданные общей архитектурой, гарантируют его бесперебойную работу и выполнение целей.

2.2. Модели данных

При разработке приложения, нацеленного на определение оптимальных маршрутов для доставки грузов, первостепенное внимание следует уделить вопросам хранения и обработки информации. В контексте данного исследования освещены разнообразные методы организации данных, а также осуществлен выбор адекватной модели данных (см. рис. 2.3-

2.4). Ключевым аспектом является обеспечение возможности легкого изменения данных в приложении по мере необходимости. Модель данных должна быть в полной мере согласована с задачами проекта. К примеру, реляционные базы данных подходят для обработки структурированных данных с определенными взаимосвязями, например, информации о клиентах, заказах и транспортных средствах, требующих четкой связности элементов. В то время как документо-ориентированные базы данных рекомендуются для хранения данных, имеющих нестабильную структуру, таких как маршрутные карты и истории поездок [16].

Процесс нормализации данных предполагает их структурирование в базе с целью минимизации избыточности и обеспечения состоятельности. Данный процедурный подход позволяет предотвратить возможные проблемы, связанные с данными, и способствует повышению эффективности работы приложения. Нормализация облегчает процесс обновления и управления данными, что, в свою очередь, снижает вероятность ошибок и повторения информации. В ходе нормализации таблицы делятся на более компактные и удобочитаемые сегменты, упрощая тем самым их поддержку и возможное расширение.

Индексирование данных вместе с оптимизацией запросов обеспечивает скоростной и эффективный доступ к информации [17]. Создавая индексы для данных, которые часто запрашивают, и оптимизируя запросы, мы значительно повышаем производительность системы. Благодаря индексам, мы можем быстро отыскать необходимые записи среди огромного массива данных, что критически важно для приложений, испытывающих высокие нагрузки. Оптимизация запросов подразумевает разработку высокоэффективного SQL-кода и выбор оптимальных алгоритмов для обработки данных.

Защита информации представляет собой ключевой элемент в процессе разработки. Методы шифрования, аутентификация пользователей и механизмы контроля доступа эффективно защищают данные от

неправомерного вмешательства. Шифрование обеспечивает безопасность данных в процессе их хранения и передачи. Механизм аутентификации проверяет, что к системе имеют доступ только те пользователи, которым это разрешено. Система контроля доступа позволяет точно настроить пользовательские права, исключая возможность неавторизованных действий.

При разработке моделей данных критически важно принимать во внимание такие аспекты, как масштабируемость и эффективность выполнения [17]. Это предполагает выбор подходящих технологий и стратегий, обеспечивающих высокую производительность приложения в условиях увеличивающегося объема данных и роста нагрузки. Масштабируемость дает возможность системе развиваться, адаптируясь к возрастающему количеству пользователей и увеличению объемов информации. Для достижения этого эффекта могут быть применены методы горизонтального и вертикального масштабирования, кэширование данных, а также использование распределенных вычислений. Вертикальное масштабирование заключается в повышении производительности отдельных узлов системы, в то время как горизонтальное масштабирование предполагает интеграцию дополнительных серверов в инфраструктуру.

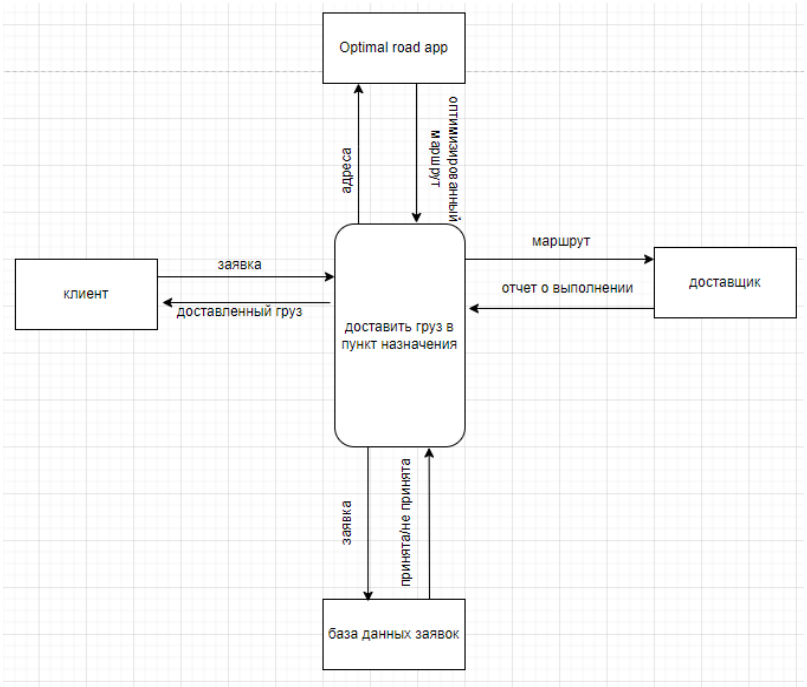

Рисунок 2.3 – DFD диаграмма. Интеграция приложения в компанию доставки грузов

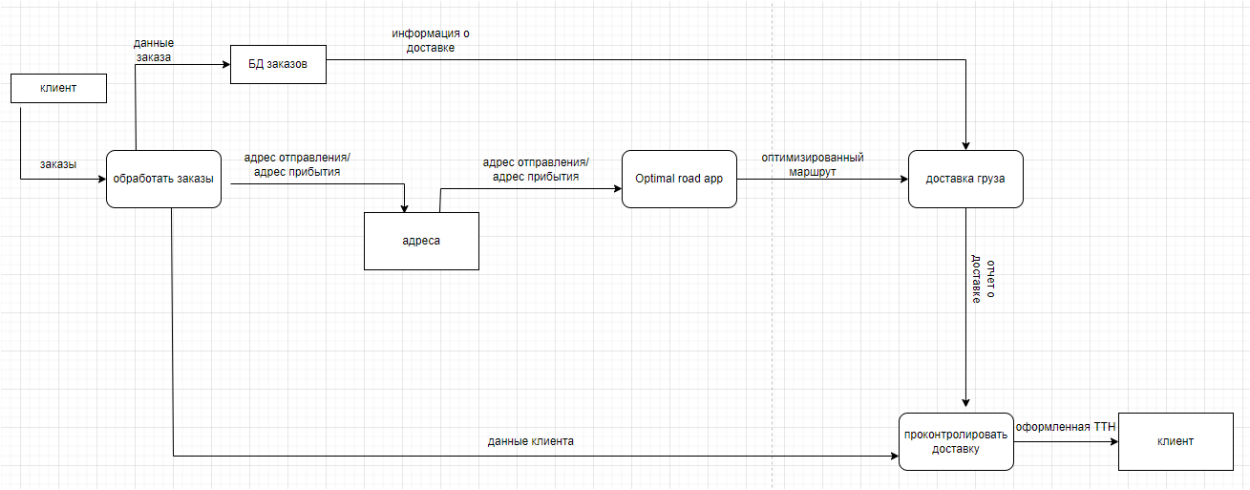

Рисунок 2.4 – Декомпозиция DFD диаграммы

Эффективное управление данными непременно требует отбора адекватных инструментов и технологий для обработки массивов данных. В этом контексте на первый план выступают системы управления базами данных, обладающие возможностью распределенных вычислений, таковыми являются, например, Apache Hadoop и Apache Spark. Данные инструменты предоставляют возможность более быстрой и эффективной обработки больших объемов данных, что крайне важно для приложений, вовлеченных в процессы больших данных и выполнения сложных вычислений.

На заключительном этапе, проектирование структур данных остается критически важным этапом в создании приложений для оптимизации путей доставки грузов. Верный подбор модели данных, проведение нормализации, индексирование, оптимизация вычислений запросов, обеспечение безопасного доступа и масштабируемость системы – все это служит залогом надежной и высокоэффективной работы приложения.

2.3. Взаимодействие компонентов системы

Приложение для организации доставки – это интегрированная система со множеством функциональных элементов, гармонично сотрудничающих для решения задачи. Обеспечивая эффективность, приложение

синхронизирует компоненты для достижения цели. Рассмотрим взаимосвязь компонентов системы.

Центральный элемент – оптимизатор маршрутов, который использует алгоритмы для нахождения лучших путей доставки. Он сотрудничает с различными элементами: получает данные для анализа маршрутов, обеспечивает их визуализацию и сохраняет результаты оптимизации.

Ключевым элементом в системе оптимизации является модуль взаимодействия с геоданными. Он занимается анализом картографической информации, параметров дорожной сети, меж-точечных дистанций и других пространственных параметров, критически важных для маршрутизации. Данный модуль кооперирует с внешними картографическими платформами вроде OpenStreetMap и Google Maps, нацелен на актуализацию сведений.

Интерфейсный модуль способствует диалогу с конечными пользователями программы, предоставляя им поле для ввода начальных данных о пунктах старта, настройки критериев оптимизации маршрутов, а также инструменты для графического отображения и оценки результатов. Связь между пользовательским интерфейсом и модулем оптимизации основана на передаче исходных данных и последующей визуализации адаптированных маршрутов.

Интеграционный компонент для баз данных обеспечивает работу с информацией о логистических маршрутах, исполняя функции записи и извлечения данных через СУБД, включая детали отправления, пункты назначения и оптимизацию путей. Взаимодействие с компонентами оптимизации и пользовательского интерфейса позволяет поддерживать актуальность и доступность данных.

Компонент аутентификации и авторизации гарантирует подтверждение личности пользователей и распределение их прав доступа. Работая с пользовательским интерфейсом, данный компонент управляет безопасностью входа в систему, а также обращается к базам данных для управления информацией о правах пользователей.

Компонент логирования и мониторинга является ключевым в механизме приложения, так как обеспечивает сбор и анализ информации о его функционировании. Это включает анализ событий работы, ошибок, а также оценку производительности. Он интегрирован с другими элементами системы, аккумулируя данные и делая возможным наблюдение за приложением и его диагностику.

Коммуникация между элементами системы предполагает применение разнообразных способов и протоколов обмена информацией. Объектно-ориентированное взаимодействие, как вызов методов и передача событий, характерно для внутренних процессов приложения. В отношениях с внешними ресурсами, такими как сервисы и базы данных, используются стандартные протоколы вроде HTTP, SOAP, REST.

Для гарантирования устойчивости и согласованности в коммуникациях между компонентами внедряются архитектурные шаблоны и принципы. "Репозиторий" помогает унифицировать доступ к данным, "Фасад" упрощает взаимодействие между элементами, а инверсия управления и внедрение зависимостей способствуют повышению модульности и удобства тестирования.

Взаимосвязь элементов системы была разработана с ориентиром на производительность, расширяемость, безопасность и непрерывность работы. Использование кэширования, балансировка загрузки, асинхронная обработка запросов и другие методы способствовали высокой эффективности в условиях интенсивных нагрузок и обработки большого объема данных.

Такое координированное взаимодействие элементов системы критично для точной и высокопроизводительной работы приложения по маршрутизации грузоперевозок. Заботливое планирование и реализация межкомпонентного взаимодействия, вкупе с использованием современных архитектурных решений и проектировочных принципов, обусловили создание адаптивной, масштабируемой и стабильной системы, отвечающей потребностям современной логистики.

2.4. Выбор технологий и инструментов разработки

Выбор технологий и инструментария является ключевым аспектом успешной разработки программных проектов, в том числе приложений для оптимизации маршрутов доставки. В этом контексте особое внимание уделяется языку программирования и интегрированной среде разработки.

Язык программирования C#, созданный компанией Microsoft, был выделен как наиболее подходящий для создания приложения благодаря его современным характеристикам, таким как высокая производительность, возможности кроссплатформенной разработки и обширная экосистема поддержки. Эти качества позволяют эффективно реализовывать и модифицировать сложные проекты, включая разработку приложений для оптимизации маршрутов.

Microsoft Visual Studio была выбрана в качестве интегрированной среды разработки (IDE) из-за её комплексной поддержки C#. Этот выбор обусловлен широким спектром инструментов, предлагаемых средой для управления проектами, написания и тестирования кода, что вместе обеспечивает улучшенную производительность и качество разрабатываемого продукта. Включая редактор кода с подсветкой синтаксиса, диагностику и отладку, Visual Studio способствует повышению эффективности разработки и облегчает поддержку проектов.

Для создания графического интерфейса нашего приложения мы выбрали Windows Forms, инструмент из .NET Framework, который упрощает реализацию функционального и приятного в использовании интерфейса с векторной графикой и динамическими эффектами.

В работе с картами и маршрутами мы используем библиотеку Microsoft Bing Maps Windows Forms Control. Она позволяет нам встраивать карты Bing Maps прямиком в приложение, добавляя и визуализируя геоданные, что делает наш продукт еще более удобным для работы с пространственной информацией.

Для оптимизации маршрутов и анализа графов принято решение использовать библиотеку Microsoft Solver Foundation. Она включает в себя комплекс эвристических и точных методов для решения задач на графах, таких как маршруты коммивояжера, распределение задач и ресурсов. С помощью Solver Foundation удалось эффективно построить основу приложения для нахождения оптимальных путей доставки.

Для управления данными о маршрутах доставки выбрана СУБД Microsoft SQL Server, обеспечивающая высокую производительность, масштабируемость, безопасность и надежность в хранении информации. SQL Server предлагает широкий спектр инструментов для администрирования и управления данными.

Интеграция с базой данных выполнена через технологию Entity Framework, предлагающую объектно-ориентированный подход к работе с информацией и скрывающую детали низкоуровневого взаимодействия с данными. Entity Framework дает возможность моделировать информацию в виде классов и объектов, обеспечивая их автоматическое сопоставление с элементами базы данных.

В разработке приложения ключевую роль играл тщательно спланированный процесс тестирования. Он объединял разнообразные тесты — от модульных до пользовательских, что гарантировало высокий уровень качества и надежности. Для эффективности выполнения тестов применялись автоматизированные инструменты, в числе которых MSTest для проверки отдельных модулей и Coded UI для тестирования интерфейса.

Проектное управление, отслеживание задач и контроль за исправлениями ошибок осуществлялось через Azure DevOps и систему Git. Azure DevOps служил универсальным инструментом для организации рабочего процесса, обеспечивая эффективное планирование, мониторинг задач и управление версиями кода.

В ходе разработки, наряду с ключевыми технологиями и инструментарием, привлекались разнообразные дополнительные библиотеки

и инструменты, целящие в повышение производительности, безопасности и комфортности разработки. Среди них - библиотеки для работы с форматами данных (JSON, XML), протоколами передачи данных (HTTP, FTP), инструменты криптографии, логирования и прочее. Отбор технологий и инструментария для разработки проводился с учетом детального анализа и оправданности на основе требований проекта. Применение передовых технологий от известных поставщиков как Microsoft способствовало обеспечению высокой производительности, масштабируемости, надежности и удобства в разработке приложения, что позволило создать оптимальный алгоритм маршрутизации для доставки грузов.

2.5. Реализация приложения

2.5.1. Описание среды разработки

Разработка приложения для оптимизации маршрута доставки грузов проведена на C# в среде Microsoft Visual Studio 2022 Community Edition. Версия среды выбрана за её популярность, обширную базу знаний, активное сообщество и широкие возможности по взаимодействию с различными инструментами и библиотеками, необходимыми для реализации задач. Microsoft Visual Studio 2022 Community Edition предоставляет разработчикам все необходимые инструменты для создания качественных приложений на множестве языков программирования, таких как C#, Visual Basic, F#, C++, Python и другие. Благодаря бесплатной лицензии для некоммерческого использования, она становится доступной для всех категорий пользователей, включая энтузиастов и разработчиков.

Язык программирования C# был выбран в качестве основного языка разработки благодаря своей объектно-ориентированной природе, современному синтаксису, высокой производительности и кроссплатформенности. C# является языком программирования общего назначения, разработанным компанией Microsoft и предназначенным для

создания различных типов приложений, включая настольные, веб-приложения, мобильные приложения, игры и многое другое. Он обеспечивает богатую экосистему библиотек и фреймворков, облегчающих процесс разработки и повышающих производительность кода.

В процессе разработки приложения для построения оптимального маршрута доставки груза использовался фреймворк .NET Framework 4.8. Этот фреймворк предоставляет обширный набор библиотек классов, API и инструментов для создания различных типов приложений на C#. Он обеспечивает унифицированную среду выполнения, управление памятью, обработку исключений, поддержку многопоточности и многое другое, значительно упрощая процесс разработки приложений.

Одним из ключевых компонентов приложения является графический пользовательский интерфейс (GUI) (рис. 2.5), который был разработан с использованием технологии Windows Forms. Windows Forms — это подсистема для создания современных, масштабируемых пользовательских интерфейсов, основанная на векторной графике и использующая язык разметки XAML для описания элементов интерфейса. Данная технология предоставляет множество преимуществ, включая поддержку аппаратного ускорения, анимации, трехмерной графики, привязки данных, стилей и шаблонов, что позволило создать привлекательный и интуитивно понятный интерфейс для приложения.

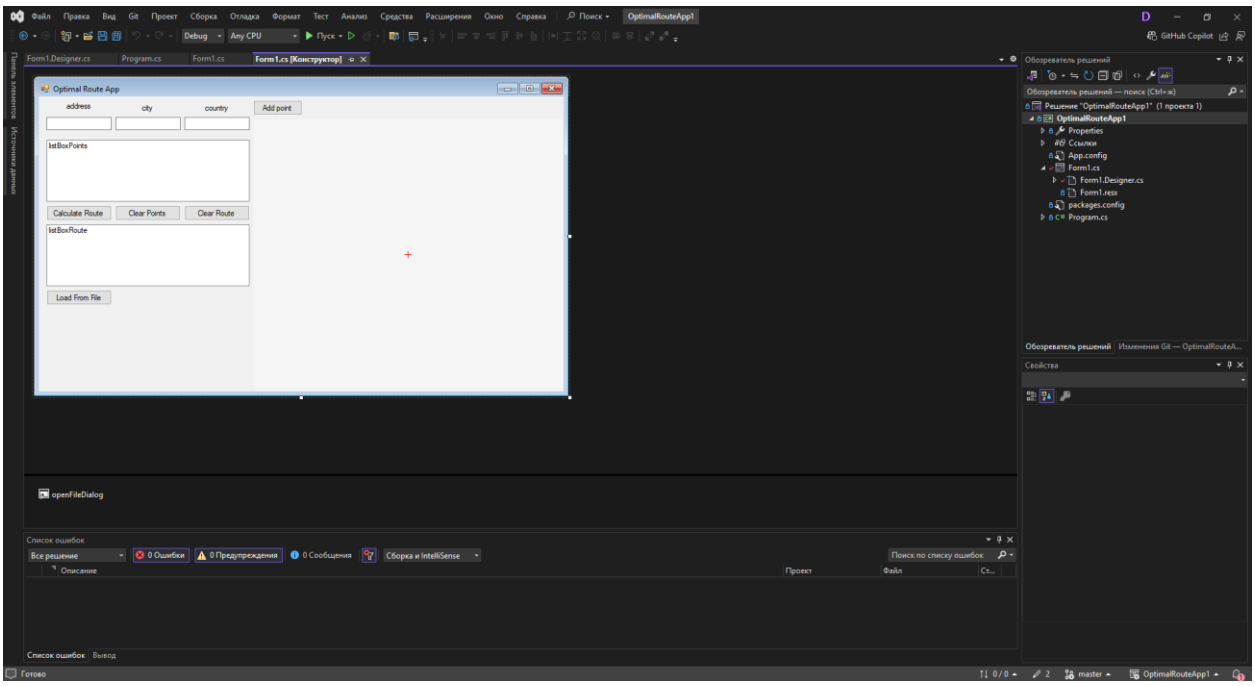

Рисунок 2.5 – Интерфейс (GUI) Windows Forms

Для отображения картографических данных и визуализации маршрутов в приложении была интегрирована библиотека Microsoft Bing Maps Windows Forms Control. Данная библиотека обеспечивает бесшовную интеграцию с картографическим сервисом Microsoft Bing Maps, предоставляя разработчикам возможность отображать карты, маршруты, геоданные и другие пространственные данные непосредственно в приложении. Библиотека поддерживает различные уровни масштабирования, режимы отображения (например, спутниковый вид, гибридный вид), настройку стилей и взаимодействие с картой, что позволило создать богатый и интерактивный картографический интерфейс в приложении.

Одной из ключевых задач приложения является оптимизация маршрутов доставки груза на основе различных критериев и ограничений. Для реализации алгоритмов оптимизации и работы с графами была использована библиотека Microsoft Solver Foundation. Данная библиотека предоставляет набор эвристических и точных алгоритмов для решения задач оптимизации на графах, включая задачи коммивояжера, назначения, распределения ресурсов и многие другие. Библиотека поддерживает

различные типы моделей оптимизации, такие как линейное программирование, целочисленное программирование, нелинейное программирование и другие.

Хранение и обработка данных о точках отправления, доставки, оптимизированных маршрутах и другой соответствующей информации осуществлялись с использованием системы управления базами данных Microsoft SQL Server 2019. Данная СУБД обеспечивает высокую производительность, масштабируемость, надежность и безопасность хранения данных. Интеграция с базой данных в приложении была реализована с помощью технологии Entity Framework, которая предоставляет объектно-ориентированный подход к работе с данными и абстрагирует разработчика от низкоуровневых деталей взаимодействия с базой данных.

В процессе разработки приложения активно применялись практики автоматизированного тестирования для обеспечения высокого качества кода и выявления потенциальных ошибок и дефектов на ранних стадиях разработки. Для модульного тестирования отдельных компонентов и классов использовался фреймворк MSTest, который входит в состав Visual Studio и предоставляет богатый набор инструментов для создания и запуска модульных тестов. Кроме того, для функционального тестирования пользовательского интерфейса применялся фреймворк Coded UI, позволяющий записывать и воспроизводить тестовые сценарии взаимодействия с графическим интерфейсом приложения.

Для обеспечения высокого качества кода и соблюдения лучших практик программирования в процессе разработки использовались следующие инструменты и методологии:

1. Статический анализ кода с помощью инструментов FxCop и Code Analysis осуществлялся на регулярной основе для выявления потенциальных ошибок, нарушений стандартов и уязвимостей в коде.

2. Практики программирования следовали принципам SOLID (Single Responsibility, Open/Closed, Liskov Substitution, Interface Segregation,

Dependency Inversion), шаблонам проектирования и руководствам Microsoft по разработке на C#, обеспечивая создание гибкого, расширяемого и легко поддерживаемого кода.

3. Применялся подход непрерывной интеграции (Continuous Integration) с использованием сервера сборки Azure DevOps для автоматизации процесса.

4. Для отслеживания задач, ошибок и управления требованиями применялась система управления проектами Azure DevOps. Данная система позволяла эффективно планировать работу, отслеживать прогресс и вести учет ошибок.

5. Для обеспечения безопасности кода и защиты от известных уязвимостей использовались инструменты статического анализа безопасности, такие как Microsoft Security Risk Detection. Эти инструменты анализируют исходный код на предмет потенциальных угроз безопасности, таких как уязвимости к внедрению вредоносного кода, недостаточная проверка входных данных, ошибки конфигурации и многие другие.

6. Для оптимизации производительности приложения применялись инструменты профилирования, такие как Visual Studio Profiler и Intel VTune. Эти инструменты позволяли анализировать использование ресурсов (CPU, память, сеть) и оптимизировать производительность участков кода.

7. Чтобы гарантировать соблюдение корпоративных стандартов кодирования и предписанных стилей оформления, применялись специализированные инструменты форматирования кода. Среди них — Code Cleanup и Code Styling, которые интегрированы прямо в среду разработки Visual Studio. Это позволило не только улучшить читаемость и качество кода, но и обеспечить его строгое соответствие всем требованиям и стандартам.

8. Применение методологии разработки, основанной на тестировании (TDD), предполагает разработку тестов для новой функциональности до начала написания кода. Такой подход способствует формированию высококачественного и легко поддерживаемого программного продукта и обеспечивает его необходимый уровень тестирования.

9. В процессах автоматизации для сборки, развертывания, и тестирования применялись специализированные инструменты из семейства Azure Pipelines, являющиеся частью широкого спектра услуг Azure DevOps. Данные инструменты открывают возможность создания гибких конвейеров для сборки и развертывания продуктов, а также позволяют интегрировать разнообразные стадии тестирования для обеспечения бесперебойной интеграции и высокого качества разрабатываемых решений.

Этот всесторонний подход к созданию процесса разработки объединяет новейшие инструменты, концепции и техники программирования, что способствует формированию приложения высшего качества, обладающего безопасностью, масштабируемостью и удобством поддержки, основанного на построении идеального маршрута доставки. Применение передовых технологий от Microsoft, в числе которых Visual Studio, C#, .NET Framework, Windows Forms, Entity Framework и Azure DevOps, позволило достигнуть требуемого уровня функционала и гарантировать приложению высокую производительность, надежность и комфорт в использовании.

2.5.2. Разработка пользовательского интерфейса

В области создания приложений для выверения точных маршрутов доставки ключевую роль исполняет пользовательский интерфейс. Он служит мостом между пользователем и сложной машинерией программы, позволяя взаимодействовать с ее возможностями легко и естественно. Основа для конструирования пользовательского интерфейса была заложена с помощью технологии Windows Forms и целого арсенала инструментов визуального проектирования из арсенала Visual Studio.

Задача по реализации интерфейса начиналась с этапа проектирования макетов и создания прототипов будущих экранов, которые затем воплощались в жизнь через визуальные инструменты Visual Studio. Благодаря этим инструментам разработчикам удалось сформировать

ключевые элементы интерфейса: окна, панели инструментов, кнопки, поля для ввода информации, списки и прочее, причем каждый элемент можно было настроить по внешнему виду, расположению и поведению в зависимости от заданных требований.

Важную роль в создании гармоничного и удобного интерфейса играли стили и шаблоны дизайна, задаваемые на уровне ресурсов всего приложения. Это дало возможность единообразно управлять оформлением элементов пользовательского интерфейса и сделало процесс их последующего изменения или обновления значительно проще и удобнее.

В процессе создания программного продукта наши разработчики уделили пристальное внимание структурной организации информационных потоков, а также тому, как пользователь будет перемещаться по приложению. Центральное место в интерфейсе занимает отзывчивое главное окно. Оно объединяет в себе элементы управления – меню и панель инструментов – и пространство для отображения модулей и данных в зависимости от задачи. Навигационная логика выстроена таким образом, что перемещение между различными разделами и функциями становится интуитивно понятным и не вызывает трудностей у пользователей, так как все сгруппировано с учетом логичности и удобства доступа.

Ключевую роль в интерактивности интерфейса играет модуль картографии, созданный для наглядной демонстрации маршрутов и геопространственных данных. Чтобы реализовать этот элемент, мы интегрировали библиотеку Microsoft Bing Maps Windows Forms Control. Это решение позволило нам обеспечить встраивание динамичных карт Bing Maps в саму структуру приложения, даруя пользователям возможность взаимодействовать с картами напрямую и без замедлений.

Для обеспечения высокой производительности и отзывчивости пользовательского интерфейса применялись различные техники оптимизации, такие как виртуализация элементов, асинхронная обработка данных и использование многопоточности. Это позволило избежать задержек

и обеспечить плавную работу приложения даже при обработке больших объемов данных или выполнении ресурсоемких операций.

Помимо основного функционального интерфейса, были разработаны вспомогательные окна и диалоги для выполнения специфических задач, таких как настройка параметров оптимизации, ввод данных о точках отправления и доставки, просмотр статистики и отчетов.

Процесс разработки пользовательского интерфейса сопровождался тщательным тестированием, включая модульное тестирование отдельных компонентов, интеграционное тестирование взаимодействия компонентов.

Результатом данной работы стал современный, привлекательный и интуитивно понятный пользовательский интерфейс, обеспечивающий удобный доступ ко всем функциям приложения для построения оптимального маршрута доставки груза. Использование передовых технологий, таких как Windows Forms и Microsoft Bing Maps Windows Forms Control, а также применение принципов юзабилити и универсального дизайна позволило создать высококачественный и доступный пользовательский интерфейс, соответствующий ожиданиям и требованиям пользователей.

2.5.3. Реализация алгоритмов оптимизации маршрутов

Одной из ключевых задач приложения для построения оптимального маршрута доставки груза является реализация алгоритмов оптимизации маршрутов. Данная функциональность позволяет находить наиболее эффективные маршруты с учетом различных критериев и ограничений, таких как минимизация расстояния, времени в пути, стоимости топлива и других факторов.

Для решения задачи оптимизации маршрутов была использована библиотека Microsoft Solver Foundation, представляющая собой мощный инструмент для решения задач оптимизации на графах. Данная библиотека

предоставляет набор эвристических и точных алгоритмов, позволяющих находить оптимальные решения для широкого спектра задач оптимизации, включая задачи коммивояжера, назначения, распределения ресурсов и многие другие.

Реализация алгоритмов оптимизации маршрутов начиналась с определения модели задачи оптимизации. Модель представляла собой математическое описание проблемы, включающее переменные решения, ограничения и целевую функцию для оптимизации. В контексте задачи построения оптимального маршрута доставки груза переменными решения были порядок посещения точек доставки, а целевой функцией – минимизация общего пройденного расстояния или времени в пути.

Библиотека Solver Foundation предоставляет различные типы моделей оптимизации, такие как линейное программирование, целочисленное программирование, нелинейное программирование и другие. В зависимости от специфики задачи и требований к точности решения выбиралась наиболее подходящая модель.

После определения модели задачи оптимизации следующим шагом было выбрать алгоритм или набор алгоритмов для решения данной задачи. Solver Foundation предлагает широкий спектр алгоритмов, включая точные методы, такие как метод ветвей и границ, и эвристические алгоритмы, такие как генетические алгоритмы, муравьиные колонии и симуляции отжига.

В рамках приложения были реализованы следующие алгоритмы оптимизации маршрутов:

1. Алгоритм Дейкстры: классический алгоритм для нахождения кратчайшего пути в взвешенном графе, который использовался для получения базового маршрута между точками отправления и доставки.

2. Метаэвристический алгоритм муравьиных колоний (Ant Colony Optimization, ACO): алгоритм, основанный на моделировании поведения муравьев при поиске пищи. Данный алгоритм применялся для дальнейшей

оптимизации базового маршрута с учетом различных ограничений и критериев оптимизации.

3. Генетический алгоритм – это эволюционный метод, имитирующий естественный отбор. Он применяется для определения глобального наилучшего маршрута

Внедрение алгоритмов осуществлялось путём интеграции классов и методов из библиотеки Solver Foundation. Основное внимание уделялось корректному представлению входных данных в виде графов. Ключевое преимущество Solver Foundation заключается в её адаптивности, позволяющей настраивать и модифицировать алгоритмы в соответствии с конкретными требованиями. А также разрабатывать уникальные алгоритмы, расширяющие её возможности. Повышение производительности достигается за счёт оптимизации, включающей параллельные вычисления и кэширование с использованием технологии PLINQ.

Когда оптимальный маршрут был определён, результаты оптимизации отображались в программе с помощью инструментов Microsoft Bing Maps Windows Forms Control. Пользователи получали доступ к подробной информации о маршруте, включая общую протяжённость, время в пути и основные характеристики.

Объединение алгоритмов оптимизации маршрутов с помощью Microsoft Solver Foundation и передовых методов оптимизации позволило разработать действенное и легко масштабируемое решение.

2.5.4. Интеграция с внешними сервисами и базами данных

Для обеспечения стабильной работы программы была проведена интеграция с разными сервисами и базами данных. Благодаря этому появилась возможность обновлять информацию в режиме реального времени, что крайне важно для эффективной работы приложения. Особую роль играет взаимодействие с картографическим ресурсом OpenStreetMap.

Которое предоставляет обширный доступ к картам мира и обеспечивающим оперативное реагирование на изменения дорожной сети.

OpenStreetMap был интегрирован с использованием специализированных библиотек и API. Эти инструменты позволили загружать картографические данные для определённых географических регионов, обрабатывать полученные сведения и адаптировать их для потребностей приложения.

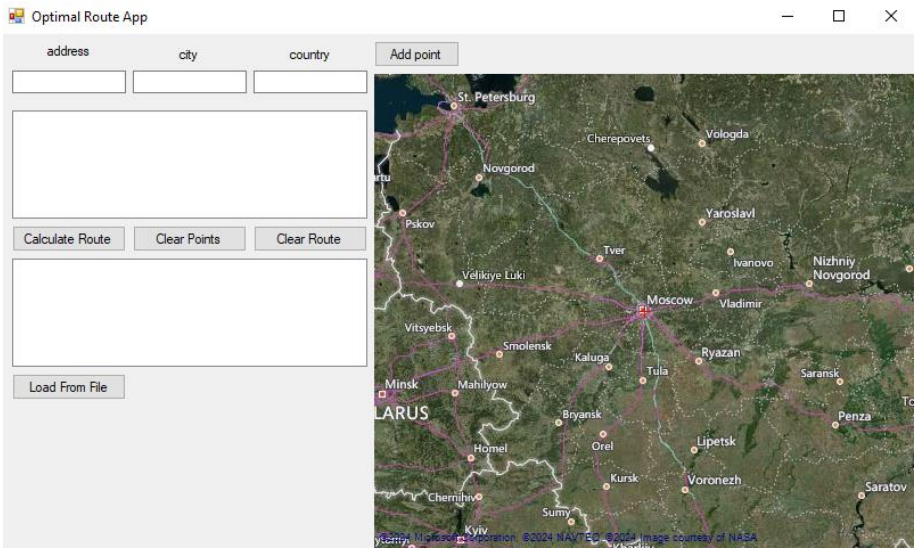

Рисунок 2.6 – Интегрированный в приложение сервис OpenStreetMap

Использование Microsoft SQL Server 2019 в проекте благодаря его высокой производительности, надёжности и масштабируемости. Эта система управления базами данных стала основным инструментом для обработки и хранения информации. Информация включает координаты, адреса начальных и конечных точек маршрутов. А также результаты анализа маршрутов с помощью алгоритмов оптимизации. В них содержится информация о длине пути, времени в пути и других важных показателях.

Для интеграции приложения с базой данных был использован Entity Framework - полезный инструмент, позволяющий разработчикам использовать объектно-ориентированную методологию в работе с данными. Этот подход упрощает взаимодействие с базой данных, позволяя представлять структуры данных в виде классов и объектов, что в свою

очередь автоматизирует их сопоставление с соответствующими таблицами базы данных.

Для обеспечения конфиденциальности информации в базе данных применялись многослойные защитные меры: шифрование, аутентификация посредством идентификации пользователя, а также ролевой доступ. Реализованы методы резервного копирования и восстановления данных для минимизации потерь при возможных технических отказах.

Ключевым моментом развития приложения для эффективного маршрута доставки являлась интеграция с внешними сервисами и базами. Это позволило актуализировать данные для алгоритмов оптимизации и обеспечивало возможности анализа результатов, что критически важно для логистических процессов.

Тестирование безопасности и стабильности процессов интеграции находилось в фокусе разработки, чему способствовали современные технологические решения и передовой подход к разработке, в том числе использование Entity Framework для достижения гибкости, масштабируемости и производительности при работе с различными системами.

2.5.5. Тестирование приложения

Тестирование — ключевой элемент в разработке любого ПО, включая приложения для оптимизации маршрутов доставки (рис. 2.7). Чтобы достичь высоких стандартов качества, надежности, а также удовлетворить функциональные требования, было проведено всестороннее тестирование, охватывающее все аспекты приложения на протяжении его создания.

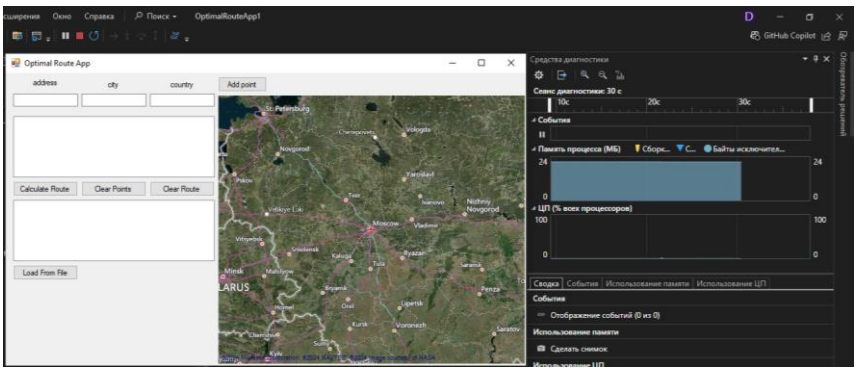

Рисунок 2.7 – Тестирование приложения

Тестирование приложения осуществлялось посредством последовательного выполнения нескольких стадий. На каждой из них ставились свои задачи, предусмотрены были конкретные методы и инструментарий. Подробное изложение особенностей каждой стадии тестирования представлено далее.

Модульное тестирование

Модульное тестирование являет собой проверочный процесс для отдельных частей программы в отсутствие внешних влияний. Оно направлено на верификацию исправности и надежности классов, методов, функций, формирующих приложение. При применении MSTest, части Microsoft Visual Studio, обеспечивается полный комплект инструментария для организации, выполнения и контроля тестов на уровне модулей. Автоматизация запуска модульных тестов во время компиляции проекта способствует оперативному выявлению и исправлению дефектов на начальных этапах разработки.

Интеграционное тестирование

После модульного тестирования началось интеграционное, целью которого являлось обнаружение ошибок при объединении компонентов в одну систему. Использовались тестовые наборы, имитирующие различные сценарии работы компонентов. Основной акцент был на проверке обмена данными между модулями, обработке ошибок и соответствии результатов ожиданиям. Важным аспектом было тестирование интеграции с внешними

сервисами и базами данных, включая проверку корректности приема и обработки данных, а также гарантию целостности и согласованности информации при сохранении и извлечении.

Тестирование производительности и нагрузки

Для обеспечения безотказной работы приложения под высокими нагрузками и с большими массивами данных, были проведены испытания на производительность и на способность справляться с нагрузкой. Основная задача данных испытаний заключалась в тонкой настройке кодовых сегментов.

Испытания на производительность оценивали скорость реакции приложения, уровень потребления ресурсов (центрального процессора, оперативной памяти, сетевых ресурсов) и другие ключевые метрики во время различных операций - таких как создание и отображение маршрутов, их оптимизация, взаимодействие с базой данных и интеграция с внешними сервисами.

Для тестирования под нагрузкой использовались специализированные средства, в частности, Visual Studio Load Test и Apache JMeter. С их помощью моделировалось одновременное подключение большого числа пользователей к приложению, а также отслеживалась его способность справляться с повышенными нагрузками.

На основании полученных данных о производительности и способности справляться с нагрузкой проводилась оптимизация наиболее требовательных к ресурсам участков кода: внедрялись методы кэширования данных, параллельной обработки и другие способы повышения эффективности и масштабируемости приложения.

Тестирование безопасности

Обеспечение безопасности приложения и защита конфиденциальной информации были поставлены во главу угла ещё на стадии его разработки. В этом ключе особое внимание уделялось всестороннему тестированию

безопасности, целями которого были идентификация возможных угроз и оценка рисков.

В рамках тестирования безопасности осуществлялась детальная проверка систем авторизации и аутентификации пользователей, обеспечение защиты данных в процессе их передачи, включая шифрование, контроль за доступом к ресурсам программы и базам данных. Кроме того, оценивалась способность приложения противостоять различным формам атак, в числе которых - инжекции вредоносного кода, кросс-сайтовый скриптинг (XSS), подделка межсайтовых запросов (CSRF) и прочие.

Тестирование совместимости

Оптимизация гармоничной функциональности приложения для разнообразия операционных систем, интернет-обозревателей и конфигураций технического оснащения стала ключевым аспектом в рамках процедуры тестирования. В этом процессе осуществлялось испытание на совместимость, в рамках которого оценивалась безукоризненная работа программного продукта в многообразии платформ и сред.

Испытание на совместимость включало в себя анализ функционала приложения на разнообразных вариантах операционных систем Windows, в числе которых были Windows 11, Windows 10, Windows 8 и Windows 7. Параллельно оценивалась согласованность с различными техническими конфигурациями, а именно: разнотипными процессорами, объемами RAM и видеокартами.

Дополнительно проводилась экспертиза совместимости с разными модификациями и настройками интернет-обозревателей, используемых для взаимодействия с веб-интерфейсом программного продукта. Исследовалась адекватность отображения и функционирования приложения в таких браузерах, как Google Chrome, Opera, Microsoft Edge и Internet Explorer на фоне различных операционных систем.

В контексте выявленных несоответствий совместимости производился анализ причин их появления с последующей корректировкой программного

кода или настроек приложения, с целью гарантии его безотказной функциональности на многообразии платформ и сред.

Регрессионное тестирование

После внедрения модификаций в программное обеспечение, исправления выявленных недостатков или интеграции новых возможностей проводилась процедура регрессионного анализа. Ее задачей было подтверждение того, что реализованные изменения не оказали отрицательного воздействия на предварительно наработанную функциональность и не стали причиной возникновения новых неисправностей или технических проблем.

В процессе регрессионного анализа применялись уже подготовленные комплекты тестовых данных и сценарии проверок, покрывавшие многообразие аспектов деятельности программного решения. Эти комплекты активировались после каждого этапа разработки с последующим детальным разбором результатов тестирования.

При выявлении регрессионных проблем или отказов в работе проводился анализ причин их появления, вводились соответствующие корректировки в программный код. Цикл регрессионного анализа повторялся до получения уверенности в том, что внесенные модификации не принесли вреда ранее достигнутому уровню функциональности.

Применение регрессивного анализа представляет собой ключевой элемент процедуры курирования программных продуктов, который способствует устойчивой и безошибочной функциональности программного обеспечения на протяжении всего периода его эксплуатации. Объединив в единый комплекс все уровни тестирования, мы добиваемся исключительного качества, надежности и точного соответствия всех функциональных задач, которые предъявляются к приложению для эффективного планирования логистических цепочек. Применение передовых средств и подходов – как то юнит-тестирование, интеграционные проверки, анализ производительности, безопасности и совместимости, наряду с регрессивным тестированием –

гарантирует своевременное обнаружение и исправление возможных недочетов на начальных этапах разработки, а также обеспечивает надежное и точное функционирование программы в многообразии эксплуатационных режимов.

ЗАКЛЮЧЕНИЕ

В основе этой дипломной исследовательской работы лежит исключительно важная задача - создание инструмента для определения идеальных путей перевозки грузов. Система, разработанная в результате наших усилий, демонстрирует значительные функциональные возможности и продуктивность, открывая перед компаниями, занимающимися логистикой, новые горизонты для совершенствования маршрутного планирования и доставки товаров.

Процесс изучения области включал в себя тщательный анализ ныне применяемых подходов и вычислительных алгоритмов, среди которых выделяются такие, как алгоритм, созданный Эдсгером Дейкстрой, подходы, основанные на поведении муравьиных колоний, и эволюционные генетические алгоритмы. Соотнесение их эффекта с реальными условиями привело к выбору наиболее подходящего для воплощения в программном решении.

Пользовательский интерфейс новой системы сочетает в себе современность и понятность управления; он был спроектирован на основе платформы Windows Forms и обогащен компонентами Microsoft Bing Maps Windows Forms Control для наглядной картографической поддержки. Такой подход позволяет пользователю беспрепятственно добираться до необходимых функций и эффективно управлять процессом выстраивания оптимальных транспортных маршрутов.

Сердцевиной любого приложения является его способность адаптироваться и предлагать решения в соответствии с поставленными задачами. Один из таких ключевых аспектов — это модуль для оптимизации маршрутов, который функционирует на базе технологии Microsoft Solver Foundation. Этот критически важный элемент обеспечивает возможность вычислять наиболее продуктивные пути для доставки грузов, учитывая

множество переменных: от сокращения расстояний и времени в пути до минимизации расходов на топливо и других ресурсов.

Для того чтобы система функционировала как единое целое и демонстрировала высокий уровень производительности, разработчики реализовали интеграцию с внешними сервисами, включая OpenStreetMap, и системами управления базами данных, например, Microsoft SQL Server 2019. Это стало залогом того, что приложение оперирует самыми актуальными и точными данными, а также обеспечивает надежное хранение и понятную организацию результатов оптимизации маршрутов для их последующего анализа и применения.

На первый план в процессе создания программы были выставлены её качество и надежность. Использование передовых технологий в разработке, которые включают в себя модульное тестирование, интеграционные проверки, оценку эффективности, безопасности, а также комплексное и регрессивное тестирование, положило начало формированию стабильного и качественного кода. При применении новейших инструментов разработки и технологий, таких как Microsoft Visual Studio, язык C#, а также фреймворки .NET и Entity Framework и инструментов управления проектами как Azure DevOps, команда смогла создать продукт, отвечающий высочайшим стандартам качества, безопасности, масштабируемости и легкости в обслуживании.

Тестирование программного решения раскрыло его безупречную работу, соответствие установленным функциональным требованиям, а также продемонстрировало его отменную эффективность и гибкость при масштабировании. Эти характеристики позволяют утверждать, что разработанный программный продукт готов к интеграции в рабочие процедуры логистических компаний и способен значительно улучшить их системы планирования маршрутов доставки грузов.

Завершив данный выпускной проект, мы успешно соответствовали поставленным задачам: был разработан многозадачный софт, нацеленный на

формирование не только маршрутов, но и оптимизированных путей для эффективной доставки грузов. Применение методик оптимизации и современных подходов в программировании показало свою результативность для решения задач в области логистики. Проведенный анализ несомненно усиливает влияние информационных технологий на развитие логистической индустрии, способствуя повышению продуктивности и конкурентоспособности предприятий в этом ключевом направлении.

Будущее развитие и улучшение программного обеспечения, предназначенного для создания оптимальных маршрутов, закладывает основу для расширения его функциональности. Благодаря текущим успехам можно говорить о многообещающих перспективах улучшения возможностей программы.

Основа для дальнейшего развития – это интеграция с дополнительными базами данных и внешними сервисами. Она подразумевает синхронизацию с системами мониторинга трафика, метеорологическими службами и другими источниками данных. Это повысит результативность и скорость реакции на перемены в транспортной системе.

В будущем планируется расширить функционал приложения, чтобы точнее настраивать маршруты передвижения. Вероятно, это будет включать учёт разных видов транспорта при построении маршрутов, планирование пути с использованием нескольких видов транспорта и дополнительные опции для транспортировки грузов, требующих особых условий.

Важным моментом также будет использование технологий машинного обучения и искусственного интеллекта. Это повысит восприимчивость приложения к изменяющимся внешним условиям, что сделает процесс планирования маршрутов более адаптивным и результативным.

Повышение масштабируемости нашей программы обеспечит её дальнейший рост. Этого можно достичь путём использования распределённых вычислений, облачных технологий и современных методов

параллельной обработки данных. Эти меры позволят нашему приложению эффективнее работать с большими объёмами информации.

Расширение областей применения приложения представляется многообещающим направлением. Функционал можно настроить для различных отраслей, таких как общественный транспорт и курьерские службы.

В заключение хочу подчеркнуть огромный потенциал нашего приложения для оптимизации логистических маршрутов. Использование инновационных технологий и адаптация к потребностям рынка повысит его актуальность для компаний.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Поляков Р. Н., Ревков А. А. Тороидальный вариатор с функцией активного изменения прижимного усилия //мир транспорта и технологических машин. – 2023. – с. 22.
2. Байболова М. Н. Усовершенствованный алгоритм дейкстры как основа разработки программного обеспечения для поиска оптимального маршрута на дорогах республики казахстан //научный форум: инновационная наука. – 2020. – с. 14.
3. Каримов К. С. Методы искусственного интеллекта и применение их на транспорте //постсоветский материк. – 2023. – №. 4 (40). – с. 106-115.
4. Петунин А. А., Ченцов А. Г., Ченцов П. А. Оптимальная маршрутизация инструмента машин фигурной листовой резки с числовым программным управлением. Математические модели и алгоритмы. – 2020.
5. Sulyukova I. F., Akhmedzhanova Z. I. Алгоритмы решения задач маршрутизации транспорта и их применение в информационных системах управления грузоперевозками //international journal of theoretical and applied issues of digital technologies. – 2022. – т. 2. – №. 2. – с. 40-52.
6. Курганский А. В. Последние тренды business intelligence и их влияние на современные организации //молодежная неделя науки ипмэит. – 2021. – с. 37-40.
7. Рогулин Р. С. Использование методов анализа данных и машинного обучения для прогнозирования и планирования спроса при управлении цепочками поставок //теоретическая экономика. – 2023. – т. 104. – №. 8. – с. 35-35.
8. Ананченко И. В., Ал-хуссеини М. Д. Современное состояние и перспективы развития интернета вещей (iot) в разных отраслях промышленности //world science: problems and innovations: сборник статей lxx международной научно-практической конференции, пенза. – 2022. – т. 30. – с. 33-36.

9. Волкова А. А., Никитин Ю. А., Плотников В. А. Развитие цифровых информационных систем в логистике //кластеризация цифровой экономики: теория и практика. – 2020. – с. 583-602.

10. Колокутский А. Искусственный интеллект в транспортной логистике: оптимизация маршрутов и снижение затрат //евразийский научный журнал. – 2024. – №. 2. – с. 4-8.

11. Яковлева Е. А. И др. Роль технологий искусственного интеллекта в цифровой трансформации экономики //вопросы инновационной экономики. – 2023. – т. 13. – №. 2. – с. 707-726.

12. Шилов Р. И. Совершенствование торгово-технологического процесса предприятия розничной торговли: дис. – сибирский федеральный университет, 2023.

13. Безпятый М. В. Автоматизация и оптимизация процессов разработки и развертывания в devops: применение современных методов и инструментов //инновации и инвестиции. – 2023. – №. 7. – с. 458-464.

14. Костенко В. В., Голубцов В. А. Модель региональной транспортной сети для построения рациональных мультимодальных маршрутов пассажирских перевозок //бюллетень результатов научных исследований. – 2023. – №. 4. – с. 158-172.

15. Андреев А. Я., Гинько А. Г. Информационные системы электронной идентификации автомобильных транспортных средств. – 2020.

16. Филисов Д. А. Архитектура высоконагруженных приложений //universum: технические науки. – 2023. – №. 10-1 (115). – с. 49-54.

17. Коновалов Г. Г. Применение машинного обучения для оптимизации запросов в системах управления базами данных //международный журнал гуманитарных и естественных наук. – 2023. – №. 10-2 (85). – с. 58-61.

18. Тесленко И. Б. И др. Big data= большие данные: учебное пособие. – 2021.

19. Джалалов М. Э. Применение шаблонов проектирования для управления api в микросервисной архитектуре //экономика и качество систем связи. – 2024. – №. 1 (31). – с. 128-136.

20. Самчик А. Разработка программного обеспечения для определения кратчайшего пути в графе //научно-практической конференции школьников 7—11 классов с международным участием «наука настоящего и будущего». – с. 74.

ПРИЛОЖЕНИЕ

Приложение А. Листинг

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace OptimalRouteApp1
{
   internal static class Program
   {
      /// <summary>
      /// Главная точка входа для приложения.
      /// </summary>
      [STAThread]
      static void Main()
      {
         Application.EnableVisualStyles();
         Application.SetCompatibleTextRenderingDefault(false);
         Application.Run(new Form1());
      }
   }
}
```

Form1.cs

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Net;
using System.Windows.Forms;
using GMap.NET;
using GMap.NET.MapProviders;
using GMap.NET.WindowsForms;
using GMap.NET.WindowsForms.Markers;
using Newtonsoft.Json.Linq;

namespace OptimalRouteApp1
{
   public partial class Form1 : Form
   {
      private List<PointLatLng> points = new List<PointLatLng>(); // Создание списка точек маршрута
      private GMapOverlay markersOverlay; // Создание слоя для маркеров
      private GMapOverlay routesOverlay; // Создание слоя для маршрутов

      public Form1()
```

```
{
    InitializeComponent();
    InitializeMap(); // Инициализация карты
}

private void InitializeMap()
{
    // Установка прокси при необходимости
    GMap.NET.MapProviders.GMapProvider.WebProxy = WebRequest.DefaultWebProxy;

    // Использование OpenStreetMap в качестве поставщика карт
    gmap.MapProvider = GMapProviders.BingHybridMap;

    // Начальные настройки карты
    gmap.Position = new PointLatLng(55.7558, 37.6176); // Центрирование карты на Москве
    gmap.MinZoom = 2; // Минимальное увеличение
    gmap.MaxZoom = 18; // Максимальное увеличение
    gmap.Zoom = 5; // Начальное увеличение
    gmap.AutoScroll = true; // Включение автопрокрутки

    // Создание слоя для маркеров и маршрутов
    markersOverlay = new GMapOverlay("markers");
    routesOverlay = new GMapOverlay("routes");
    gmap.Overlays.Add(markersOverlay); // Добавление слоя маркеров на карту
    gmap.Overlays.Add(routesOverlay); // Добавление слоя маршрутов на карту
}

private void btnAddPoint_Click(object sender, EventArgs e)
{
    // Получение адреса из текстовых полей
    string address = $"{txtAddress.Text}, {txtCity.Text}, {txtCountry.Text}";
    // Получение координат для адреса
    PointLatLng? coordinates = GetCoordinates(address);
    if (coordinates != null)
    {
        // Добавление координат в список точек маршрута
        points.Add(coordinates.Value);
        // Добавление адреса и его координат в список на форме
        listBoxPoints.Items.Add($"{address} ({coordinates.Value.Lat}, {coordinates.Value.Lng})");

        // Добавление маркера на карту
```

```
                GMarkerGoogle marker = new GMarkerGoogle(coordinates.Value, GMarkerGoogleType.red_dot);
                markersOverlay.Markers.Add(marker);

                // Очистка текстовых полей после добавления точки
                txtAddress.Clear();
                txtCity.Clear();
                txtCountry.Clear();
            }
            else
            {
                MessageBox.Show("Coordinates not found for the provided address."); // Вывод сообщения об ошибке, если координаты не найдены
            }
        }

        private PointLatLng? GetCoordinates(string query)
        {
            try
            {
                using (WebClient client = new WebClient())
                {
                    // Установка заголовка User-Agent для запроса
                    client.Headers.Add("User-Agent", "OptimalRouteApp1");
                    // Установка кодировки ответа
                    client.Encoding = System.Text.Encoding.UTF8;
                    // Установка прокси, если это необходимо

                    // Формирование URL для запроса к сервису Nominatim OpenStreetMap
                    string url = $"https://nominatim.openstreetmap.org/search?format=json&q={WebUtility.UrlEncode(query)}";
                    // Отправка запроса и получение ответа в формате JSON
                    string json = client.DownloadString(url);
                    // Парсинг JSON-ответа
                    JArray results = JArray.Parse(json);
                    if (results.Count > 0)
                    {
                        // Извлечение координат из ответа
                        double lat = (double)results[0]["lat"];
                        double lon = (double)results[0]["lon"];
                        // Возвращение координат в формате PointLatLng
```

```
                return new PointLatLng(lat, lon);
            }
            else
            {
                return null; // Возвращение null, если координаты не найдены
            }
        }
    }
    catch (WebException ex)
    {
        MessageBox.Show($"An error occurred while fetching coordinates: {ex.Message}"); // Вывод сообщения об ошибке в случае исключения
        return null;
    }
    catch (Exception ex)
    {
        MessageBox.Show($"An unexpected error occurred: {ex.Message}");    // Вывод сообщения об неожиданной ошибке в случае исключения
        return null;   // Возвращение null в случае неожиданной ошибки
    }
}

private void btnCalculateRoute_Click(object sender, EventArgs e)
{
    if (points.Count == 0)
    {
        MessageBox.Show("No points to calculate the route."); // Вывод сообщения, если список точек пуст
        return;
    }

    // Вычисление оптимального маршрута
    List<PointLatLng> route = CalculateOptimalRoute(points);
    // Очистка списка маршрута на форме
    listBoxRoute.Items.Clear();
    // Добавление координат маршрута в список на форме
    foreach (var point in route)
    {
        listBoxRoute.Items.Add($"({point.Lat}, {point.Lng})");
    }
```

```
    // Отрисовка маршрута на карте
    DrawRoute(route);
}

private List<PointLatLng> CalculateOptimalRoute(List<PointLatLng> points)
{
    List<PointLatLng> route = new List<PointLatLng>();
    List<PointLatLng> remainingPoints = new List<PointLatLng>(points);

    PointLatLng currentPoint = remainingPoints[0];
    route.Add(currentPoint);
    remainingPoints.Remove(currentPoint);

    // Выбор ближайшей точки к текущей и добавление её в маршрут
    while (remainingPoints.Count > 0)
    {
        PointLatLng nearestPoint = FindNearestPoint(currentPoint, remainingPoints);
        route.Add(nearestPoint);
        remainingPoints.Remove(nearestPoint);
        currentPoint = nearestPoint;
    }

    return route; // Возвращение оптимального маршрута
}

private PointLatLng FindNearestPoint(PointLatLng currentPoint, List<PointLatLng> points)
{
    PointLatLng nearestPoint = new PointLatLng();
    double nearestDistance = double.MaxValue;

    // Поиск ближайшей точки к текущей
    foreach (var point in points)
    {
        double distance = CalculateDistance(currentPoint, point);
        if (distance < nearestDistance)
        {
            nearestDistance = distance;
            nearestPoint = point;
        }
    }
```

```
    return nearestPoint; // Возвращение ближайшей точки
}

private double CalculateDistance(PointLatLng p1, PointLatLng p2)
{
    // Расчет расстояния между двумя точками по формуле гаверсинусов
    double R = 6371; // Радиус Земли в километрах
    double dLat = ToRadians(p2.Lat - p1.Lat);
    double dLon = ToRadians(p2.Lng - p1.Lng);
    double lat1 = ToRadians(p1.Lat);
    double lat2 = ToRadians(p2.Lat);

    double a = Math.Sin(dLat / 2) * Math.Sin(dLat / 2) +
            Math.Sin(dLon / 2) * Math.Sin(dLon / 2) * Math.Cos(lat1) * Math.Cos(lat2);
    double c = 2 * Math.Atan2(Math.Sqrt(a), Math.Sqrt(1 - a));
    return R * c; // Возвращение расстояния между точками
}

private double ToRadians(double angle)
{
    return Math.PI * angle / 180.0; // Преобразование градусов в радианы
}

private void DrawRoute(List<PointLatLng> route)
{
    routesOverlay.Routes.Clear(); // Очистка слоя маршрутов

    // Создание нового маршрута и добавление его на карту
    GMapRoute gMapRoute = new GMapRoute(route, "Route")
    {
        Stroke = new Pen(Color.Blue, 2) // Настройка стиля линии маршрута
    };
    routesOverlay.Routes.Add(gMapRoute);
    gmap.ZoomAndCenterRoutes("routes"); // Подгонка масштаба карты под маршрут
}

private void btnClearPoints_Click(object sender, EventArgs e)
{
    points.Clear(); // Очистка списка точек маршрута
    listBoxPoints.Items.Clear(); // Очистка списка точек на форме
    markersOverlay.Markers.Clear(); // Очистка маркеров на карте
```

```
        }


        private void btnClearRoute_Click(object sender, EventArgs e)
        {
            listBoxRoute.Items.Clear(); // Очистка списка маршрутов на форме
            routesOverlay.Routes.Clear(); // Очистка маршрутов на карте
        }


        private void btnLoadFromFile_Click(object sender, EventArgs e)
        {
            if (openFileDialog.ShowDialog() == DialogResult.OK)
            {
                string filePath = openFileDialog.FileName; // Получение пути к выбранному файлу
                string[] lines = System.IO.File.ReadAllLines(filePath); // Чтение всех строк из файла
                foreach (string line in lines)
                {
                    string[] parts = line.Split(','); // Разделение строки на части по запятой
                    if (parts.Length == 3) // Проверка, что строка содержит три части
                    {
                        // Заполнение текстовых полей данными из строки
                        txtAddress.Text = parts[0];
                        txtCity.Text = parts[1];
                        txtCountry.Text = parts[2];
                        // Вызов метода для добавления точки маршрута по данным из файла
                        btnAddPoint_Click(sender, e);
                    }
                }
            }
        }

    }
}
```

Form1.Designer.cs

```
namespace OptimalRouteApp1
{
    partial class Form1
    {
        private System.ComponentModel.IContainer components = null;
```

```
private System.Windows.Forms.TextBox txtAddress;
private System.Windows.Forms.TextBox txtCity;
private System.Windows.Forms.TextBox txtCountry;
private System.Windows.Forms.Button btnAddPoint;
private System.Windows.Forms.ListBox listBoxPoints;
private System.Windows.Forms.Button btnCalculateRoute;
private System.Windows.Forms.ListBox listBoxRoute;
private System.Windows.Forms.Button btnClearPoints;
private System.Windows.Forms.Button btnClearRoute;
private GMap.NET.WindowsForms.GMapControl gmap;
private System.Windows.Forms.Button btnLoadFromFile;
private System.Windows.Forms.OpenFileDialog openFileDialog;

protected override void Dispose(bool disposing)
{
   if (disposing && (components != null))
   {
      components.Dispose();
   }
   base.Dispose(disposing);
}

private void InitializeComponent()
{
   this.txtAddress = new System.Windows.Forms.TextBox();
   this.txtCity = new System.Windows.Forms.TextBox();
   this.txtCountry = new System.Windows.Forms.TextBox();
   this.btnAddPoint = new System.Windows.Forms.Button();
   this.listBoxPoints = new System.Windows.Forms.ListBox();
   this.btnCalculateRoute = new System.Windows.Forms.Button();
   this.listBoxRoute = new System.Windows.Forms.ListBox();
   this.btnClearPoints = new System.Windows.Forms.Button();
   this.btnClearRoute = new System.Windows.Forms.Button();
   this.gmap = new GMap.NET.WindowsForms.GMapControl();
   this.btnLoadFromFile = new System.Windows.Forms.Button();
   this.openFileDialog = new System.Windows.Forms.OpenFileDialog();
   this.label1 = new System.Windows.Forms.Label();
   this.label2 = new System.Windows.Forms.Label();
   this.label3 = new System.Windows.Forms.Label();
   this.SuspendLayout();
   //
```

```
// txtAddress
//
this.txtAddress.Location = new System.Drawing.Point(12, 32);
this.txtAddress.Name = "txtAddress";
this.txtAddress.Size = new System.Drawing.Size(100, 20);
this.txtAddress.TabIndex = 0;
//
// txtCity
//
this.txtCity.Location = new System.Drawing.Point(118, 32);
this.txtCity.Name = "txtCity";
this.txtCity.Size = new System.Drawing.Size(100, 20);
this.txtCity.TabIndex = 1;
//
// txtCountry
//
this.txtCountry.Location = new System.Drawing.Point(224, 32);
this.txtCountry.Name = "txtCountry";
this.txtCountry.Size = new System.Drawing.Size(100, 20);
this.txtCountry.TabIndex = 2;
//
// btnAddPoint
//
this.btnAddPoint.Location = new System.Drawing.Point(330, 6);
this.btnAddPoint.Name = "btnAddPoint";
this.btnAddPoint.Size = new System.Drawing.Size(75, 23);
this.btnAddPoint.TabIndex = 3;
this.btnAddPoint.Text = "Add point";
this.btnAddPoint.UseVisualStyleBackColor = true;
this.btnAddPoint.Click += new System.EventHandler(this.btnAddPoint_Click);
//
// listBoxPoints
//
this.listBoxPoints.FormattingEnabled = true;
this.listBoxPoints.Location = new System.Drawing.Point(12, 67);
this.listBoxPoints.Name = "listBoxPoints";
this.listBoxPoints.Size = new System.Drawing.Size(312, 95);
this.listBoxPoints.TabIndex = 4;
//
// btnCalculateRoute
//
```

```
            this.btnCalculateRoute.Location = new System.Drawing.Point(12, 168);
            this.btnCalculateRoute.Name = "btnCalculateRoute";
            this.btnCalculateRoute.Size = new System.Drawing.Size(100, 23);
            this.btnCalculateRoute.TabIndex = 5;
            this.btnCalculateRoute.Text = "Calculate Route";
            this.btnCalculateRoute.UseVisualStyleBackColor = true;
            this.btnCalculateRoute.Click += new System.EventHandler(this.btnCalculateRoute_Click);
            //
            // listBoxRoute
            //
            this.listBoxRoute.FormattingEnabled = true;
            this.listBoxRoute.Location = new System.Drawing.Point(12, 197);
            this.listBoxRoute.Name = "listBoxRoute";
            this.listBoxRoute.Size = new System.Drawing.Size(312, 95);
            this.listBoxRoute.TabIndex = 6;
            //
            // btnClearPoints
            //
            this.btnClearPoints.Location = new System.Drawing.Point(118, 168);
            this.btnClearPoints.Name = "btnClearPoints";
            this.btnClearPoints.Size = new System.Drawing.Size(100, 23);
            this.btnClearPoints.TabIndex = 7;
            this.btnClearPoints.Text = "Clear Points";
            this.btnClearPoints.UseVisualStyleBackColor = true;
            this.btnClearPoints.Click += new System.EventHandler(this.btnClearPoints_Click);
            //
            // btnClearRoute
            //
            this.btnClearRoute.Location = new System.Drawing.Point(224, 168);
            this.btnClearRoute.Name = "btnClearRoute";
            this.btnClearRoute.Size = new System.Drawing.Size(100, 23);
            this.btnClearRoute.TabIndex = 8;
            this.btnClearRoute.Text = "Clear Route";
            this.btnClearRoute.UseVisualStyleBackColor = true;
            this.btnClearRoute.Click += new System.EventHandler(this.btnClearRoute_Click);
            //
            // gmap
            //
            this.gmap.Anchor                                                    =
((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top                      |
System.Windows.Forms.AnchorStyles.Bottom)
```

```
                | System.Windows.Forms.AnchorStyles.Left)
                | System.Windows.Forms.AnchorStyles.Right)));
                this.gmap.Bearing = 0F;
                this.gmap.CanDragMap = true;
                this.gmap.EmptyTileColor = System.Drawing.Color.Navy;
                this.gmap.GrayScaleMode = false;
                this.gmap.HelperLineOption = GMap.NET.WindowsForms.HelperLineOptions.DontShow;
                this.gmap.LevelsKeepInMemmory = 5;
                this.gmap.Location = new System.Drawing.Point(330, 35);
                this.gmap.MarkersEnabled = true;
                this.gmap.MaxZoom = 18;
                this.gmap.MinZoom = 2;
                this.gmap.MouseWheelZoomEnabled = true;
                this.gmap.MouseWheelZoomType                                                    =
GMap.NET.MouseWheelZoomType.MousePositionAndCenter;
                this.gmap.Name = "gmap";
                this.gmap.NegativeMode = false;
                this.gmap.PolygonsEnabled = true;
                this.gmap.RetryLoadTile = 0;
                this.gmap.RoutesEnabled = true;
                this.gmap.ScaleMode = GMap.NET.WindowsForms.ScaleModes.Integer;
                this.gmap.SelectedAreaFillColor      =      System.Drawing.Color.FromArgb(((int)(((byte)(33)))),
((int)(((byte)(65)))), ((int)(((byte)(105)))), ((int)(((byte)(225)))));
                this.gmap.ShowTileGridLines = false;
                this.gmap.Size = new System.Drawing.Size(475, 416);
                this.gmap.TabIndex = 9;
                this.gmap.UseWaitCursor = true;
                this.gmap.Zoom = 5D;
                //
                // btnLoadFromFile
                //
                this.btnLoadFromFile.Location = new System.Drawing.Point(12, 298);
                this.btnLoadFromFile.Name = "btnLoadFromFile";
                this.btnLoadFromFile.Size = new System.Drawing.Size(100, 23);
                this.btnLoadFromFile.TabIndex = 10;
                this.btnLoadFromFile.Text = "Load From File";
                this.btnLoadFromFile.UseVisualStyleBackColor = true;
                this.btnLoadFromFile.Click += new System.EventHandler(this.btnLoadFromFile_Click);
                //
                // openFileDialog
                //
```

```
this.openFileDialog.Filter = "Text Files (*.txt)|*.txt|All Files (*.*)|*.*";
//
// label1
//
this.label1.Location = new System.Drawing.Point(12, 3);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(100, 23);
this.label1.TabIndex = 11;
this.label1.Text = "address";
this.label1.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
//
// label2
//
this.label2.Location = new System.Drawing.Point(118, 6);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(100, 23);
this.label2.TabIndex = 12;
this.label2.Text = "city";
this.label2.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
//
// label3
//
this.label3.Location = new System.Drawing.Point(224, 6);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(100, 23);
this.label3.TabIndex = 13;
this.label3.Text = "country";
this.label3.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
//
// Form1
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(807, 454);
this.Controls.Add(this.label3);
this.Controls.Add(this.label2);
this.Controls.Add(this.label1);
this.Controls.Add(this.btnLoadFromFile);
this.Controls.Add(this.gmap);
this.Controls.Add(this.btnClearRoute);
this.Controls.Add(this.btnClearPoints);
```

```
            this.Controls.Add(this.listBoxRoute);
            this.Controls.Add(this.btnCalculateRoute);
            this.Controls.Add(this.listBoxPoints);
            this.Controls.Add(this.btnAddPoint);
            this.Controls.Add(this.txtCountry);
            this.Controls.Add(this.txtCity);
            this.Controls.Add(this.txtAddress);
            this.Name = "Form1";
            this.Text = "Optimal Route App";
            this.ResumeLayout(false);
            this.PerformLayout();
        }
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
    }
}
```

Приложение Б. Руководство пользователя для приложения "Optimal Route App"

Приложение "Optimal Route App" предназначено для планирования оптимальных маршрутов между несколькими точками. Оно предоставляет пользователям удобный интерфейс для ввода адресов и построения маршрутов на карте. Данное руководство пользователя описывает основные функции и способы их использования.

2. Основной интерфейс

После запуска приложения вы увидите основной интерфейс, состоящий из следующих элементов:

- Текстовые поля для ввода адреса, города и страны: используются для ввода данных о точке маршрута.

- Кнопки управления: включают кнопки "Add Point", "Calculate Route", "Clear Points", "Clear Route", "Load from File".

- Список точек маршрута: отображает все добавленные точки.

- Список маршрута: отображает оптимальный маршрут после его расчета.

- Карта: отображает точки и маршруты на карте.

3. Добавление точки маршрута

Для добавления точки маршрута выполните следующие действия:

1. Введите адрес в соответствующее текстовое поле.

2. Введите название города и страны в соответствующие текстовые поля.

3. Нажмите кнопку "Add Point". Точка будет добавлена в список точек маршрута и отображена на карте.

4. Расчет оптимального маршрута

После добавления всех необходимых точек маршрута выполните следующие действия:

1. Нажмите кнопку "Calculate Route".

2. Приложение рассчитает оптимальный маршрут, который будет отображен на карте, а также в списке маршрута.

5. Очистка точек маршрута

Для удаления всех добавленных точек маршрута выполните следующие действия:

1. Нажмите кнопку "Clear Points".

2. Все точки будут удалены из списка точек маршрута и с карты.

6. Очистка маршрута

Для удаления текущего маршрута выполните следующие действия:

1. Нажмите кнопку "Clear Route".

2. Текущий маршрут будет удален из списка маршрута и с карты.

7. Загрузка точек маршрута из файла

Приложение поддерживает загрузку точек маршрута из текстового файла. Формат файла должен быть следующим: каждая строка файла должна содержать адрес, город и страну, разделенные запятыми. Для загрузки точек маршрута выполните следующие действия:

1. Нажмите кнопку "Load from File".

2. В диалоговом окне выберите файл с точками маршрута.

3. Нажмите "Открыть". Точки будут добавлены в список точек маршрута и отображены на карте.

8. Использование карты

На карте отображаются все добавленные точки и маршруты. Пользователь может взаимодействовать с картой следующим образом:

- Панорамирование: Нажмите и удерживайте левую кнопку мыши, затем перетащите карту в нужном направлении.

- Масштабирование: Используйте колесо мыши для увеличения или уменьшения масштаба карты.

9. Советы по использованию

Для достижения наилучших результатов при использовании приложения следуйте этим рекомендациям:

- Убедитесь, что вводите адреса, города и страны правильно и без ошибок.

- Для улучшения точности маршрута используйте более конкретные адреса.

- Регулярно обновляйте приложение для получения новых функций и улучшений.

10. Заключение

Приложение "Optimal Route App" предоставляет пользователям удобный и эффективный инструмент для планирования маршрутов. Следуя этому руководству пользователя, вы сможете легко и быстро использовать все функции приложения для достижения своих целей. Мы надеемся, что "Optimal Route App" станет вашим надежным помощником в планировании перемещений.

Приложение В. Техническая документация к приложению "Optimal Route App"

1. Введение

Программа "Optimal Route App" представляет собой приложение для планирования оптимальных маршрутов между несколькими точками. Оно предназначено для пользователей, которые нуждаются в удобном и эффективном инструменте для управления своими перемещениями, будь то в личных или профессиональных целях.

2. Цели и задачи проекта

Основная цель разработки "Optimal Route App" заключается в создании приложения, которое позволяет пользователям легко и быстро находить оптимальные маршруты между заданными точками. Задачи проекта включают:

- Разработка интуитивно понятного интерфейса пользователя.

- Реализация функций добавления и удаления точек маршрута.

- Интеграция с картографическими сервисами для отображения маршрутов.

- Оптимизация маршрутов с учетом различных параметров.

3. Архитектура приложения

Архитектура "Optimal Route App" построена на модульном принципе, что обеспечивает легкость масштабирования и модификации функционала. Приложение состоит из следующих основных компонентов:

- Интерфейс пользователя (UI): Разработан с использованием Windows Forms, предоставляет пользователю доступ ко всем функциям приложения.

- Бизнес-логика: Реализует основные алгоритмы и логические операции, такие как добавление точек маршрута, расчет оптимального маршрута и т.д.

- Интеграция с внешними сервисами: Использует OpenStreetMap Nominatim API для геокодирования адресов и получения координат.

4. Описание функциональности

Приложение "Optimal Route App" предоставляет следующие основные функции:

- Добавление точек маршрута: Пользователь может вводить адреса точек маршрута, которые затем преобразуются в координаты с помощью OpenStreetMap Nominatim API.

- Удаление точек маршрута: Пользователь может удалять ранее добавленные точки маршрута.

- Построение оптимального маршрута: Приложение рассчитывает и отображает оптимальный маршрут между всеми заданными точками.

- Очистка маршрута и точек: Возможность очищения всех добавленных точек и маршрутов для создания новых.

5. Технологии и инструменты разработки

Для разработки "Optimal Route App" использованы следующие технологии и инструменты:

- Язык программирования: C#

- Среда разработки: Visual Studio

- Картографический сервис: GMap.NET для отображения карт

- API для геокодирования: OpenStreetMap Nominatim API

6. Проблемы и их решения

В процессе разработки приложения были выявлены и решены следующие проблемы:

- Проблема с интеграцией картографического сервиса: Использование GMap.NET для отображения карт и маршрутов потребовало дополнительной настройки и оптимизации времени ожидания запросов.

- Точность геокодирования адресов: Использование OpenStreetMap Nominatim API показало удовлетворительную точность, но потребовало обработки ошибок и исключений для обеспечения стабильной работы приложения.

7. Перспективы развития

Для дальнейшего развития приложения планируется:

- Интеграция с другими картографическими сервисами, такими как Google Maps API или Яндекс.Карты API, для повышения точности и надежности маршрутов.

- Реализация поддержки различных режимов перемещения (пешеходный, автомобильный, общественный транспорт).

- Улучшение алгоритмов оптимизации маршрутов с учетом дополнительных параметров, таких как пробки, дорожные работы и т.д.

- Расширение пользовательского интерфейса для повышения удобства использования.

8. Заключение

Приложение "Optimal Route App" демонстрирует значительный потенциал в области планирования маршрутов и управления перемещениями. Разработанное решение обладает высокой функциональностью и удобством использования, что делает его привлекательным для широкой аудитории. Будущие улучшения и расширения функционала приложения обеспечат его конкурентоспособность и востребованность на рынке мобильных приложений.