

РЕФЕРАТ

Работа магистра содержит 101 страницу пояснительной записи формата А4, 44 рисунка, 6 таблиц, 54 используемых источника, 4 приложения.
ЭЛЕКТРОННЫЕ ЗАРЯДНЫЕ СТАНЦИИ, ЭЛЕКТРОКАРЫ, БРОНИРОВАНИЕ, РАСПИАНИЕ ЭЗС, ИНТЕЛЛЕКТУАЛЬНЫЙ СЕРВИС, ВЕБ-ПРИЛОЖЕНИЕ

Целью данной работы является разработка веб-приложения, представляющее собой интеллектуальный сервис по бронированию электронных зарядных станций для владельцев электрокаров и обладающего понятным интерфейсом, позволяющим любому пользователю без знаний этапов протекания бизнес-процесса и основ программирования получить качественную услугу.

Для достижения поставленной цели необходимо решить следующие задачи:

- изучить теоретические аспекты работы веб-приложений, а также технологии их разработки;
- провести обзор практического опыта и выбор метода интеллектуального управления эксплуатацией ЭЗС;
- сформировать перечень учитываемых факторов при эксплуатации ЭЗС и электрокаров;
- адаптировать метод интеллектуального управления эксплуатацией ЭЗС к текущим условиям задачи;
- разработать пользовательский интерфейс и архитектуру для функционирования веб-приложения;
- осуществить программную реализацию выбранного метода интеллектуального управления эксплуатацией ЭЗС;
- предложить экономическое обоснование разработанного веб-приложения.

Работа содержит три главы. В первой главе описываются аспекты работы веб-приложений, их траектория развития, существующие разновидности, а также роль в современном мире. Более того, теоретическая часть включает в себя основные технологии разработки интеллектуальных веб-сервисов той или иной части их архитектуры, которые при взаимной при интеграции и координации обеспечивают бесперебойные обработку введённых пользователем данных и предоставление ему её результатов или доступных альтернатив для выбора в рамках задач веб-приложения. Также в ней проводится обзор практического опыта и последующий выбор метода интеллектуального управления эксплуатацией ЭЗС, имеющего возможность быть применённым с учётом поставленных задач и доступных инструментов разработки.

Вторая глава включает в себя методологическую часть, которая предполагает анализ и отбор рассматриваемых в модели веб-приложения параметров электронных зарядных станций, а также внутренних и внешних факторов влияния на эксплуатацию электрокаров в рамках необходимости их подключения к ЭЗС во время прохождения маршрута. Помимо этого, во второй главе используются результаты из теоретической части относительно выбранного метода для его адаптации под выявленные условия для учёта в работе интеллектуального сервиса.

Третья глава посвящена практическим аспектам, а именно моделированию бизнес-процесса веб-приложения, проектированию архитектуры интеллектуального сервиса, разработке пользовательского интерфейса с использованием таких фронт-энд инструментов, как HTML, CSS и JavaScript, а также программной реализации адаптированного в методологической части метода с помощью бэк-энд инструментов: PHP и Python с его специализированными библиотеками, значительно облегчающими выполнения поставленных задач. Помимо этого, в ней будет затронуто экономическое обоснование разработанного ИТ-решения.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
ГЛАВА 1. ТЕОРИТИЧЕСКАЯ ЧАСТЬ	5
1.1. История развития веб-приложений.....	5
1.2. Основные технологии разработки интеллектуальных веб-приложений	11
1.3. Обзор практического опыта и выбор метода интеллектуального управления эксплуатацией ЭЗС.....	17
Выводы по главе 1.....	28
ГЛАВА 2. МЕТОДОЛГИЧЕСКАЯ ЧАСТЬ	30
2.1. Формирование перечня параметров эксплуатации ЭЗС	30
2.2. Формирование перечня внутренних и внешних факторов влияния на эксплуатацию электрокаров.....	35
2.3. Адаптация метода интеллектуального управления эксплуатацией ЭЗС к текущим условиям задачи	44
Выводы по главе 2.....	54
ГЛАВА 3. ПРАКТИЧЕСКАЯ ЧАСТЬ	56
3.1. Моделирование основного бизнес-процесса	56
3.2. Проектирование архитектуры интеллектуального сервиса	61
3.3. Создание пользовательского интерфейса сервиса	65
3.4. Программная реализация выбранного метода интеллектуального управления эксплуатацией ЭЗС.....	75
3.5. Экономическое обоснование разработанного веб-приложения	86
Выводы по главе 3.....	93
ЗАКЛЮЧЕНИЕ	94
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	95
ПРИЛОЖЕНИЕ А	102
ПРИЛОЖЕНИЕ Б.....	103
ПРИЛОЖЕНИЕ В	105
ПРИЛОЖЕНИЕ Г.....	112

ВВЕДЕНИЕ

Электрокары стремительно занимают своё место в российском авторынке. В 2022 наблюдался 11% рост в объемах продаж по сравнению с предыдущим годом, а прогноз на 2023 с увеличением числа проданных электромобилей на 15% также подтверждает явную экспоненциальную природу тренда в распространении данной технологии [1]. В связи с этим можно сделать вывод о том, что для эксплуатации такого электротранспортного парка необходима соответствующая инфраструктура в виде электронных зарядных станций, которая позволяет электрокарам получать ресурсы для выполнения своих прямых задач. Действительно, в Российской Федерации уже насчитывается около 1600 зарядных станций, что является пусть и небольших, но уже достаточным существенным числом, для того чтобы вызывать трудности в управлении ими традиционными инструментами учёта времени их использования пользователями [2].

В современном мире, где многие задачи решаются ИТ-решениями, отличающиеся высокими эффективностью и отказоустойчивостью, проблема контроля функционирования электрозарядной инфраструктуры так же может быть подвергнута цифровой трансформации с использованием специально разработанных программных продуктов. Именно этот вопрос и поднимается в данной работе. Уже внушительное число пользователей на территории всей страны, желающих забронировать часть мощностей электrozарядной инфраструктуры под свои нужды в конкретный период времени при совершении своих поездок, стало основным драйвером для возникновения стремлений в создании ИТ-решения, которое бы позволило принимать запросы пользователей на бронирование временных окон для зарядки своей техники, а также предлагать доступные альтернативы в расписании с учётом броней других водителей. Предполагаемая программная реализация предоставит возможность в управлении инфраструктурой ЭЗС наиболее комфортным образом для пользователей.

ГЛАВА 1. ТЕОРИТИЧЕСКАЯ ЧАСТЬ

1.1. История развития веб-приложений

Когда речь заходит о создании приложения, то сразу возникает вопрос о его дистрибуции. В таком случае на ум сразу приходит что-то вездесущее, что-то, к чему может иметь доступ практически каждый член общества. Данным местом в современном мире является сеть Интернет, которая предоставляет миллиардам людей обмениваться информацией и на её основе получать специализированные услуги. Именно по этой причине разрабатываемое ИТ-решение для выполнения задач, заявленных в рамках этой работы, будет представлять себя веб-приложение, традиционно представляющее собой клиентскую часть, доступную любому пользователю, а также серверную часть, в которой вдали от глаз клиента будет разворачиваться основная программная реализация поставленных им целей.

Тем не менее, прежде чем приступить к методологической работе, следует разобраться в истории развития веб-приложений и концепции сети Интернет в целом, как платформы, на которой они основываются. Это позволит ориентироваться, к какой стадии развития будет принадлежать разрабатываемое в рамках решения поставленных задач ИТ-решение и какие технологии стоит использовать при его создании.

На сегодняшний момент времени большинство исследователей выделяют 4 уровня развития глобальной Сети, к каждой из которых присуще своё взаимодействие веб-приложений с пользователями и сопутствующими веб-сервисами: [3], [4]

- WEB 1.0 (read-only Web);
- WEB 2.0 (participatory social Web);
- WEB 3.0 (semantic Web);
- WEB 4.0 (symbiotic Web).

Таким образом, веха «WEB 1.0» началась в 1989 году, когда рядовые люди постепенно начали получать доступ к сети Интернет. Данная эпоха

ознаменовалась статистическими HTML-страницами, с которыми пользователи не могли взаимодействовать: единственное, что они могли делать – это читать их содержимое. В первую очередь, на данном этапе сеть Интернет активно развивалась как инструмент маркетинга с онлайн-брошюрами и интернет-магазинами, предлагающие купить товар онлайн, нежели в оффлайн. Всё взаимодействие с пользователем строилась по толкающей модели (push), то есть пользователь не играл никакой роли в формировании контента на веб-сайте. Как правило, при создании веб-страницы использовалась простая технология по типу HTML (HyperText Markup Language), а коммуникационным протоколом был HTTP (HyperText Transfer Protocol).

Примерно в 2004 году веб-страницы стали динамичными, позволяя пользователям принимать участие в формировании контента веб-страницы, что стало началом эры «WEB 2.0». Как раз в это время начали зарождаться первые форумы, социальные сети, подкасты, влоги, обмен фотографиями и другие действия над контентом. Примерно в тогда появлялись первые веб-приложения, которые посредством обмена информацией с пользователем могли удовлетворять его нужды, используя вычислительные ресурсы, располагающиеся на хостинге. Данная эра охарактеризовалась проведённым экспериментом со интерактивностью, которая продемонстрировала огромный успех. Помимо этого, этот этап развития сети Интернет дал старт использованию таких новых технологий, которые широко используются и по сей день, как Ajax (Asynchronous Javascript and XML) для фонового обмена данных браузера с веб-сервером, язык программирования JavaScript для формирования статических и динамических компонентов пользовательского интерфейса, CSS (Cascading Style Sheets) для создания уникального дизайна веб-страницы, DOM (Document Object Model) для чтения и редактирования HTML- и XML-документов (Extensible Markup Language), JSON (JavaScript Object Notation) как удобный формат обмена данными, а также плагин Adobe Flash, позволяющий пользователю взаимодействовать с некоторыми

сложными программными элементами, находящимися на веб-странице. Таким образом, вышеперечисленные технологии позволили веб-приложениям предоставлять свои услуги идентично тому, как это делают обычные программы, установленные на жёсткий диск компьютера.

Существуют разные мнения по поводу того, находится ли современная сеть Интернет на стадии «WEB 3.0», либо всё ещё осуществляется переход к ней. Тем не менее, можно с уверенностью заявить, что обширная часть элементов данной эры уже активно используется во многих веб-сервисах. В этой концепции сеть Интернет представляется как единая распределённая база данных с семантическими связями между разными данными, которые обладают своими смыслами, понятными машинам через правила и логику. Таким образом, данные из разных ресурсов в Сети могут иметь связи между друг другом с целью взаимного дополнения друг друга. Хорошим примером таких семантических связей может служить ситуация, когда пользователь хочет купить билеты на полёт в другую страну, и, помимо расписаний рейсов, поисковик показывает погоду в столице, карты метками отелей на ней, к которым прилагаются доступные для брони даты комнат в них и другую сопутствующую информацию, что свидетельствует о том, что отдельные веб-сервисы могут обмениваться информацией между собой без прямых запросов человека путём использования машиночитаемых связей между ними. Также к WEB 3.0 относят такие технологии распределённого хранения данных, как блокчейн, который, в свою очередь, лежит в основе существование такого феномена, криптовалюты, а также технологий Интернета вещей, позволяющей объектами физического мира «общаться» с компьютерами путём использования сети Интернет. Данная веха ознаменовывается использованием RDF (Resource Description Framework), представляющая из себя технологию создания семантических связей между документами путём построения графов с рёбрами, являющимися связями объектами и определяющимися присвоенными им именами, что позволяет веб-приложениям логически связывать неструктурированные массивы информации из разных ресурсов

(Рисунок 1), URI (Uniform Resource Identifier) для идентификации объекта в семантической базе данных и SPARQL (Protocol and RDF Query Language) для составления запросов к данным по модели RDF.

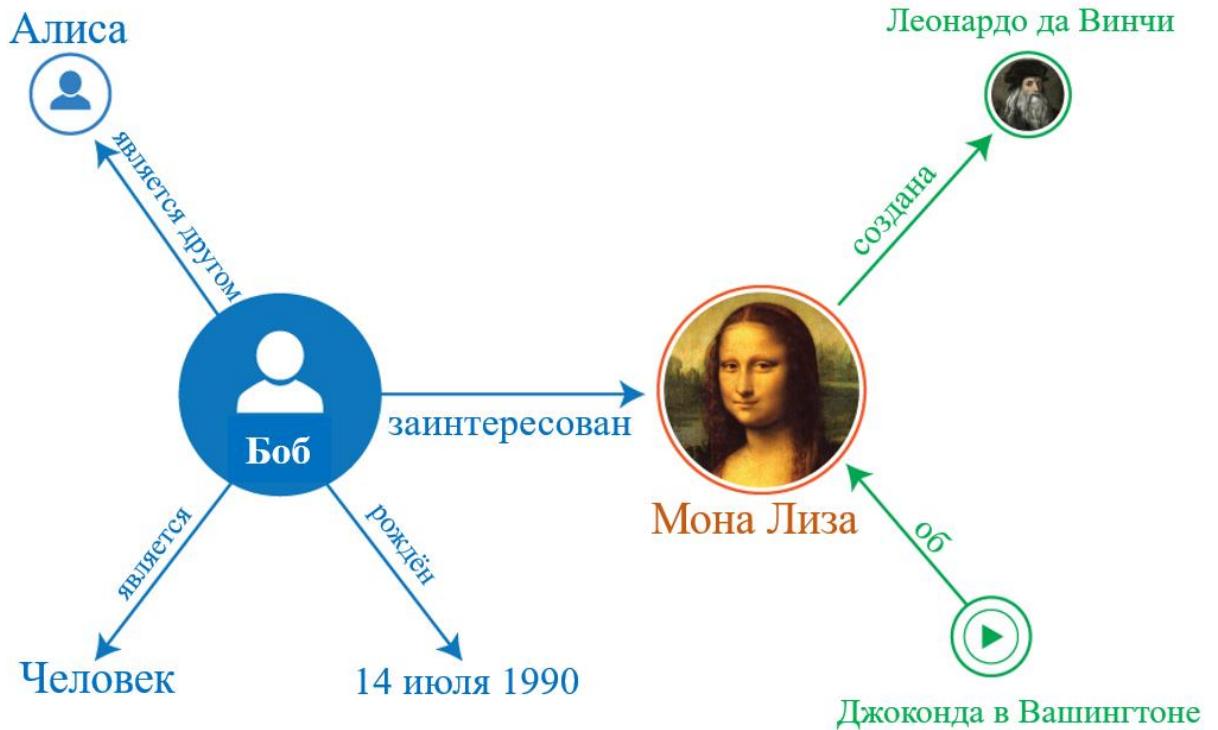


Рисунок 1. Пример RDF-схемы [5]

Касаемо этапа «WEB 4.0», можно сказать, что он является скорее предсказываемой стадией дальнейшего развития сети Интернет, где взаимодействие пользователя и веб-приложения будет осуществляться на принципах нейрокоммуникаций, опирающихся на уже существующие семантические связи между объектами. Считается, что технологии, которые позволяют совершить переход на данную веху, появятся только в 2030-2040-ых годах. На текущий момент времени только отдельные элементы описываемой технологии подвергаются экспериментам и в текущей сети Интернет они не присутствуют.

Таким образом, три стадии развития сети Интернета и веб-приложений как её существенной составляющей можно представить следующей таблицей (**Таблица 1**):

Таблица 1. Сравнительный анализ стадий развития WEB [4]

Критерий	WEB 1.0	WEB 2.0	WEB 3.0
Год начала	1994	2002	2006
Предложен от	Тим Бернерс-Ли	Дэйл Доэрти	Джон Маркоф
Поколения	Знание	Коммуникация	Кооперация
Известен как	«Сеть»	«Сеть документов»	«Сеть данных»
Взаимодействие с человеком	«Только чтение»	«Чтение-редактирование»	«Персонализация»
Представление данных	HTML	HTML, XHTML, XML	RDF, RDFa, Микроформаты
Семантика	Нет	Нет	Да
Ссылки	Гиперссылка	Гиперссылка	URI
Метаданные	Реляционная схема	XSD/DTD	Онтология
Модель данных	Реляционная	Реляционная, иерархическая	Граф
Машинное обучение	Нет	Нет	Да
Язык запросов	SQL	SQL, XPath	SPARQL
Ответы на вопросы	Где искать данные?	Как делить данными с другими?	Как искать, интегрировать и управлять данными?
Объём прилагаемых усилий для нахождения продукта/услуги/людей	Большой	Меньший в связи с отзывами и упоминаниями продукта	Намного меньший в связи с интеграцией данных через мобильных агентов
Примеры	Новости, информационные порталы, E-Commerce сайты	Вики, блоги, социальные сети	Google Squared, Zemanta, TripIt, Siri, Wolfram Alfa, Watson и т.д.
Области применения	Поиск, шоппинг и реклама	Социальные сети	«Умные» поисковые движки, семантические социальные сети

Исходя из развития сети Интернет в целом можно выделить несколько вех в развитии веб-приложений и добавляемые сопутствующие признаки в них. Их можно представить с помощью следующей схемы (*Рисунок 2*):



Рисунок 2. Стадии развития веб-приложений [6]

Таким образом, после рассмотрения стадий развития сети Интернета и изменения способов взаимодействия веб-приложений с пользователями и сопутствующими веб-сервисами на каждом из них можно сделать вывод о том, что разрабатываемый интеллектуальный сервис бронирования ЭЗС должен включать в себя элементы концепции «Web 3.0», чтобы получить необходимое конкурентное преимущество на рынке и удовлетворять современным требованиям разработки. Именно поэтому он должен обладать связями с другими веб-сервисами, быть способным к адаптации к использованию технологии Интернета вещей (в текущих условиях, с ЭЗС), а также предоставлять пользователю возможность во вводе данных и обрабатывать их, используя математические алгоритмы, необходимые в рамках решения поставленных задач.

1.2. Основные технологии разработки интеллектуальных веб-приложений

Для начала стоит разобраться, что делает сервис «интеллектуальными». Прежде всего, в современном понимании слово приложение обладает статусом «интеллектуальное», если в основе алгоритмов его работы лежит либо искусственный интеллект, либо Интернет вещей, либо аналитика больших данных. Помимо этого, такие приложения используют исторические данные и информацию в режиме реального времени от взаимодействия с пользователями или из других источников, чтобы составлять индивидуальные прогнозы или предложения, делая опыт взаимодействия клиента более адаптивным и персонализированным. [7]

Несмотря на некоторую абстрактность термина, можно выделить 3 основных признака, наличие элементов которых позволяют сервису считаться интеллектуальным: [8]

- адаптивность;

Такие приложения используют аналитику, машинное обучение и искусственный интеллект, чтобы предлагать прогнозы и рекомендации, обеспечивающие возможность для пользователя в принятии лучшего решения;

- контекстуальность;

Данные сервисы используют личные данные, информацию с датчиков и геолокацию, чтобы сделать опыт взаимодействия пользователя более персонализированным;

- проактивность.

Как правило, такие приложения сами обращаются к пользователю, а не ждут запросов от него. В них отмечается применение пуш-уведомлений, чат-ботов, сервисы сообщений и другие инструменты для взаимодействия с потребителем услуги.

Таким образом, работа интеллектуальных сервисов очень сильно опирается на обработку личных данных клиента, информацию, вводимую им, а также на связанные сведения из других источников, полезных для решения

поставленных задач. Именно по этой причине разрабатываемые алгоритмы работы веб-приложения должны опираться на данные пользователя и внешней среды, чтобы предоставить наиболее персонализированную услугу.

После изучения признака «интеллектуальности» стоит обратить внимание на общую архитектуру любых современных веб-приложений и инструменты, используемые для её разработки (*Рисунок 3*):

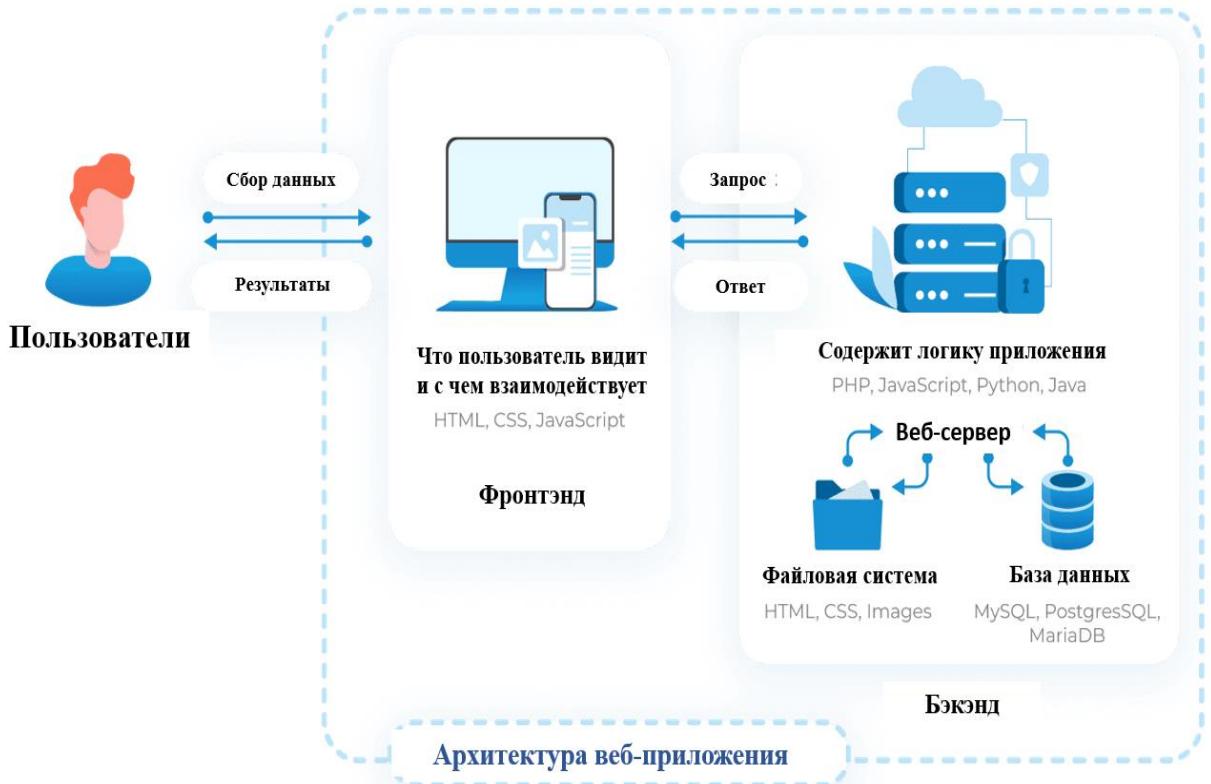


Рисунок 3. Структура веб-сервиса [9]

Фронт-энд — это презентационная часть информационной или программной системы, которая является пользовательским интерфейсом, та видимая часть, с которой взаимодействует пользователь. Можно выделить несколько типов фронтэнд-части приложения:

- односторонние приложения (SPAs - Single-Page Application);

Данный тип приложения функционирует в пределах только одной HTML-страницы, то есть пользователя нет необходимости в загрузке других веб-страниц, потому что обновлённый контент доставляется ему на одну и ту же страницу. Браузер сразу загружает весь код HTML-страницы, но демонстрирует только конкретный модуль, а при переходе на другой –

показывает уже загруженные данные для него. Такой тип фронт-энда веб-приложения крайне удобен при разработке маленьких сервисов, а также требует интуитивно понятного интерфейса при наличии только необходимых элементов управления.

- многостраничные приложения (MPAs – Multi-Page Application);

Этот тип веб-приложения функционируют в пределах нескольких страниц, то есть при запросе с сервера нового HTML-документа происходит полное обновление страницы. В отличие от первого типа он считается устаревшим, так как перегружает сервер и отличается меньшей скоростью.

- прогрессивные приложения (PWAs – Progressive Web Application).

Этот тип фронт-энда представляет из себя веб-сайт, установленный как приложение на устройство. То есть, разработчикам необязательно разрабатывать приложения для нескольких операционных систем – достаточно только поддерживать сайт и адаптировать одно общее приложение под работу с ним. Данный тип особо популярен в области социальных сетей из-за возможности работы в режиме онлайн и получения уведомлений, например, есть веб-приложения «Twitter», «Facebook», которые для которых браузер является виртуальной машиной для их работы.

Помимо этого, существуют различные подходы к загрузке контента веб-страниц:

- рендеринг на стороне сервера (SSR – Server-Side Rendering);

В таком случае сервер полностью загружает HTML-документ и отправляет веб-браузеру, с которым взаимодействует пользователь. При смене веб-страницы веб-сервер повторит работу. При получении готового HTML-документа браузер загружает сопутствующий JavaScript-код и выполняет его, после чего веб-страница становится интерактивной. При таком подходе HTML-документ создаётся на сервере, и поэтому может проиндексирован поисковой системой, что делает его поиск значительно легче. Также в таком случае веб-страница быстро загружается, но время

ответов на действия пользователя занимают большой объём времени в связи с частыми запросами на сервер. Данный тип используется, когда рейтинг веб-сайта для поиска в браузере важен и пользовательский интерфейс не обладает большим числом элементов управления в нём;

- рендеринг на стороне клиента (CSR – Client-Side Rendering).

При этом подходе веб-страница полностью загружается в браузере клиента: логика работы, контент, маршрутизация и т. д. Это позволяет уменьшить число запросов на сервер, но требует большего времени изначальной загрузки HTML-документа с сопутствующим JS-кодом, а также не позволяет поисковым системам проводить индексацию, что делает поиск веб-сайта сложнее для пользователей. Обычно такой подход используется, когда пользовательский интерфейс отличается большим числом элементов управления.

Рассмотрим основные инструменты, используемые при разработке фронтэнд-части веб-приложения и уже ранее упомянутые в первом пункте данного теоретического раздела:

- HTML – это язык разметки, используемый для построения структуры будущего веб-сайта или веб-приложения. Все абзацы, разделы, изображения, заголовки и текст написаны на HTML. Контент появляется на сайте в том порядке, в котором он написан в HTML;
- CSS – это язык стилевых правил, который используется для настройки дизайна элементов веб-страницы или веб-приложения, написанных на HTML. С помощью него как статичным, так и динамичным частям может задаваться текст, шрифт, форма, шрифт, тип анимации и т. д.
- JavaScript – это язык программирования, который используют разработчики для создания интерактивных веб-страниц. Он используется при визуализации данных, манипулированием DOM, построении форм, а также написании математических и текстовых функций. Код JavaScript интерпретируется, то есть непосредственно переводится в код машинного языка движком JavaScript. Язык JavaScript возник как технология на

стороне браузера, позволяющая сделать веб-приложения более динамичными. Используя его, браузеры могут реагировать на взаимодействие с пользователем и менять расположение контента, прописанного с помощью HTML и CSS, на веб-странице. На данный момент времени существует множество библиотек и фреймворков для JavaScript, которые позволяют разработчикам использовать уже готовые функции и шаблоны для построения пользовательского интерфейса, а также прочих элементов дизайна и управления с тратой наименьшего объёма временных ресурсов.

Бэк-энд – это разработка бизнес-логики какого-нибудь продукта (веб-сайта, интеллектуального сервиса и т. д.). Эта часть ИТ-решения отвечает за взаимодействие пользователя с внутренними данными, которые потом отображает фронт-энд. Как правило, она скрыта от глаз пользователя. Бэк-энд может включать в себя множество компонентов, но чаще всего во все программные продукты входят такие, как веб-сервера, файловые системы и базы данных. Рассмотрим каждый из них по-отдельности:

- Веб-сервер – это отдельное устройство, которое способно хранить ресурсы веб-приложения (HTML-документы, CSS-стили, JavaScript-скрипты, картинки, шрифты, медиафайлы и т. д.), а также доставлять их клиенту (как правило, веб-браузеру) по запросу от него с помощью сети Интернет и имеющегося у него доменного имени. Коммуникация сервера и клиента осуществляется путём такой части его программного обеспечения, как HTTP-сервер (HyperText Transfer Protocol), который при запросе веб-браузером какого-то файла в файловой системе веб-сервера передаёт его этому клиенту через этот протокол, в это же клиент так же может передавать входящие в него данные веб-серверу для их обработки путём этого же протокола (**Рисунок 4**). Наиболее популярными веб-серверами на данный момент являются Apache и Nginx. Разработка логики работы серверной части веб-приложения может быть осуществлена с помощью многих языков программирования, например, PHP, Ruby, Java, C, Python,

Perl и т. д., но более всего себя зарекомендовали PHP и Java: примерно 77.8% всех веб-сайтов и веб-приложений функционируют с PHP на стороне веб-сервера по состоянию на январь 2023 года [10].

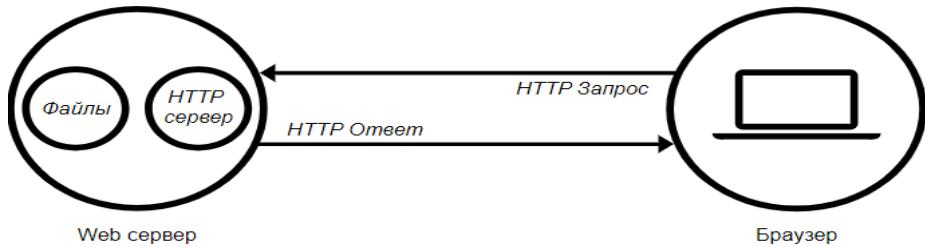


Рисунок 4. Коммуникация HTTP-сервера и клиента

- Файловая система – это инструмент, позволяющий веб-серверу обращаться к нужным файлам и работать с ними. Этими файлами могут компоненты веб-сайта или веб-приложения, часть программного обеспечения в виде взаимодействующих скриптов, написанных на множестве языков программирования, которые представляют собой основные алгоритмы, обрабатывающие входные данные от пользователя. Она позволяет хранить данные в определённом порядке, обращаться к ним с помощью их адреса, а также обновлять их, что, в принципе говорит о том, что файловая система позволяет управлять данными, хранящимися на веб-сервере.
- База данных – это инструмент, который позволяет хранить, добавлять, изменять и обновлять информацию о пользователях и сведений для реализации бизнес-логики. Другими словами, это центральный репозиторий со всей необходимой информацией для функционирования веб-приложения. Базы данных могут быть реляционными, когда информация представляется в виде набора таблиц, состоящих из столбцов и строк, каждая из которых может служить ключом для соединения с другой таблицей. В каждом столбце таблицы хранится определенный тип данных, в каждой ячейке – значение атрибута. В нереляционных базах данных не используется табличная схема строк и столбцов, а применяется модель хранения, наиболее соответствующая требованиям хранения взятого типа информации, например, с помощью графов, сетей и т.д.

Основным инструментом для взаимодействия с традиционными реляционными базами данных является SQL (Structured Query Language) и его множество диалектов по типу MySQL и PostgreSQL, которые позволяют управлять записями в базах данных: добавлять записи, модифицировать их, осуществлять их перемещение, удалять и выполнять иные функции.

Таким образом, в данном пункте теоретической части работы удалось познакомиться с наиболее традиционной структурой веб-приложения, а также некоторыми инструментами для разработки её компонентов. Данное исследование помогает сформировать базовые ориентиры для выполнения практической части этого проекта.

1.3. Обзор практического опыта и выбор метода интеллектуального управления эксплуатацией ЭЗС

С наличием необходимой теоретической базы для формирования требований к будущему веб-приложению, позволяющих ему обладать рядом конкурентных преимуществ, имеющих значимость в рамках текущих трендов, а также для отбора наиболее подходящих подходов к его разработке и инструментов, используемых во время неё, стоит приступить к рассмотрению основной задачи данной работы. В этом пункте главы будут рассмотрены некоторый практический опыт, направленный на решения задачи управления инфраструктурой электrozарядных станций при наличии заявок на их использование владельцами электрокаров в разные периоды времени и с разными требованиями к их характеристикам.

Обратим внимание на практическое исследование, проведённое индийскими специалистами, целью которого является предложение распределённую систему «S²RC» (Smart Search of Route and Charging) [11]. Она способна предложить гибкий подход к бронированию электронных зарядных станций с оптимальными маршрутами к их достижения с применением метода минимизации трёх параметров: количество электроэнергии, необходимое для

достижения выбранной зарядной станции, продолжительность ожидания доступности зарядной станции, а также стоимость процедуры зарядки электротранспортного средства на выбранной зарядной станции. Она может быть выражена следующим образом. Авторы отмечают, что в рамках решения задач по бронированию электронных зарядных станций используются 4 подхода: эвристический, коммерческие решения, применение машинного обучения, а также специализированных методов принятия решения, и связи с большой степенью практическости и экспериментальности методов для решения поставленной задачи исследователи прибегают именно к эвристическим инструментам.

Основной идеей системы является представление населённого пункта или города, в котором находится электротранспортное средство, точное местоположение которого определяется с помощью GPS-сервисов в системе, в качестве полного ориентированного графа с вершинами на перекрёстках дорог и зарядными станциями. Данная интерпретация городского пространства позволяет составлять массив допустимых маршрутов от точки местоположения электрокара до желаемым водителем пунктом назначения с учётом минимизации ранее перечисленных параметров (*Рисунок 5*). Таким образом, строятся один оптимальный маршрут и два субоптимальных, один из которых может быть выбран водителем при отказе от оптимального.



Рисунок 5. Построение маршрутов системой «S²RC» [11]

Построение оптимальных и субоптимальных маршрутов от точки местоположения электротранспортного средства до выбранной финальной точки назначения с включением зарядных станций на их протяжении происходит исходя из ограничений, которые содержатся в математической модели рассматриваемой системы, целью которой является минимизация трёх ранее упомянутых параметров: количество электроэнергии, необходимое для достижения выбранной зарядной станции, продолжительность ожидания доступности зарядной станции, а также стоимость процедуры зарядки электротранспортного средства на выбранной зарядной станции. Она может быть выражена следующим образом:

$$\min T_{ope} = E_{ope} + T_{wait} + R_{cost}, \text{ где} \quad (1.1)$$

T_{ope} – составная оценка для рассчитываемого маршрута;

E_{ope} – объём электроэнергии, необходимый для прибытия к зарядной станции, кВт;

T_{wait} – время, затрачиваемое на достижение зарядной станции, минуты;

R_{cost} – стоимость процедуры зарядки на станции, денежная единица.

Тем не менее, стоит обратиться к последующей декомпозиции данной формулы с целью расширения перечня факторов, оказывающих влияние на значения параметров в исследуемой системе. Более того, при многокритериальной оптимизации возможно возникновение частичного или полного конфликта, который, по мнению авторов исследования, может быть разрешён с помощью введения коэффициентов для каждого параметра, величина которых настраивается исходя из личных интересов и предпочтения пользователя. Они также используются в декомпозированной модели:

$$\begin{aligned} \min T_{ope} = & \varphi_1 \left(\sum_{arc=1}^k \sum_{a,b} E_c^{a,b} * J_{a,b} \right) + \\ & \varphi_2 \left(\sum_{b=1}^k \sum_{a=1}^k (T_{a,b} - T_{a,b}^e) * X_{a,b} \right) + \\ & \varphi_3 \left(\sum_{t=t_s}^{t_f} \mu(t) C_p \tau \right), \text{ где} \end{aligned} \quad (1.1.1)$$

T_{ope} – составная оценка для рассчитываемого маршрута;

φ_n – коэффициент предпочтения (вес), назначенный параметру;

k – число зарядных станций в рассматриваемой местности;

a, b – вершины полного ориентированного графа;

$E_c^{a,b}$ – количество потребляемой электроэнергии для перемещения от вершины a до вершины b , кВт;

$J_{a,b}$ – бинарная переменная выбора: {0;1} (путь между вершинами a и b не включён/включён в маршрут);

$T_{a,b}$ – момент времени начала процедуры зарядки электротранспортного средства, ч;

$T_{a,b}^e$ – расчётный момент времени прибытия к зарядной станции, ч;

$X_{a,b}$ – бинарная переменная выбора: {0;1} (путь между вершинами a и b не включён/включён в маршрут);

t_s – момент времени начала зарядки электротранспортного средства, ч

t_f – момент времени окончания зарядки электротранспортного средства, ч;

$\mu(t)$ – стоимостная функция зависимости стоимости потребления 1 кВт от продолжительности процедуры зарядки, денежная единица/кВт·ч;

C_p – мощность зарядной станции на вершине b , кВт;

τ – совокупная продолжительность процедуры зарядки электротранспортного средства, ч.

Предлагаемая система также подтверждает свою эффективность на реальном практическом опыте. Её применение для инфраструктуры электронных зарядных станций в городе на севере Индии под названием «Чандигарх» подтверждает ожидания исследователей. Хочется отметить, что рассматриваемый город является запланированным, что говорит о том, что он обладает чёткой структурой дорожной инфраструктуры, что делает задачу построения полного ориентированного графа на её основе более осуществимой. В статье рассматривается моделирование отдельных оптимальных и субоптимальных маршрутов с назначением каждого из трёх

параметров целевым, а также моделирование оптимального маршрута и субоптимальных маршрутов с применением коэффициентов предпочтения. В конечном счёте подход с построением полных ориентированных графов и идущая с ним математическая модель действительно работают, так как во множестве разных ситуациях предлагаются рациональные маршруты.

Обратимся к другому практическому опыту. В работе китайских исследователей используется подход, отличный от прошлого: в нём не используется разбиение местности на полный ориентированный граф с вершинами на перекрёстках дорог и ЭЗС, а также в нём используется не прямой перебор альтернатив маршрутов для поиска оптимального маршрута, а эволюционные алгоритмы, обеспечивающие более направленный и быстрый поиск оптимальной комбинации параметров [12]. В рамках данного практического опыта рассматривается ситуация, при которой электротраки, выезжающие из депо, расположенных в разных точках города, которые должны доставить заказы клиентам, так же находящихся в разных точках города. Основным вопросом, решаемым в этом исследовании, является оптимальное назначение электротракам клиентов для доставки и динамическое построение маршрутов для них с учётом необходимости получения заряда на станциях на их протяжении с последующим возвращением в депо после выполнения задачи.

Первой очередь авторы обращаются к гауссовым смесям, чтобы осуществить процедуру кластеризации клиентов исходя из их местоположения и окон доставки для последующего назначения каждого кластера для обслуживанияциальному депо. Результаты этой операции выглядят следующим образом (*Рисунок 6*):

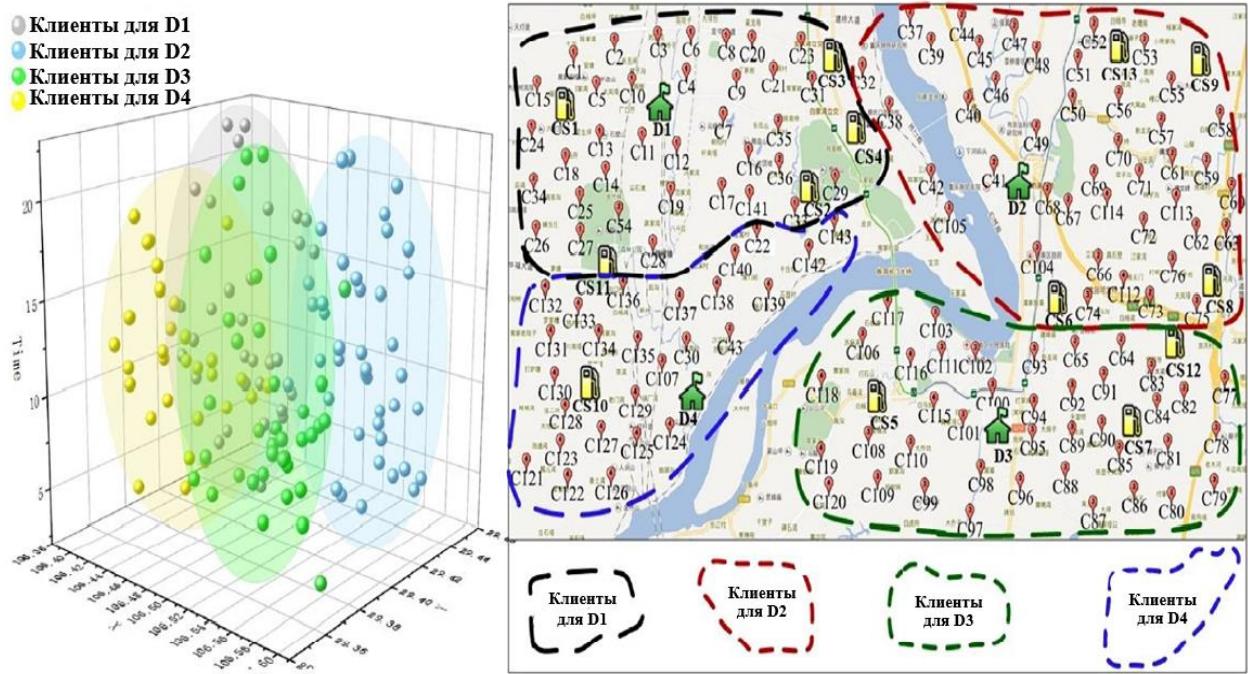


Рисунок 6. Кластеризация клиентов с помощью гауссовых смесей [12]

Следующим этапом идёт введение математической модели для многокритериальной оптимизации и предлагающимися к ней ограничений. Система уравнений математической модели и ограничений составляет функцию приспособленности, которая позволяет ввести более направленный отбор в рамках эволюционного программирования. Функция для многокритериальной оптимизации включает в себя 5 параметров, а также 21 ограничение, входящие вместе с ней в систему уравнений. В связи с обширным математическим аппаратом, используемым в данном исследовании, сфокусируемся только на верхней декомпозиции оптимизируемой функции, которая раскрывает основные параметры системы, с которыми проводятся манипуляции:

$$Z = Z_1 + Z_2 + Z_3 + Z_4 + Z_5, \text{ где} \quad (1.2)$$

Z - совокупная стоимость, денежная единица;

Z_1 - стоимость доставки в зависимости от потраченной электроэнергии на маршруте, денежная единица;

Z_2 - стоимость эксплуатации депо, денежная единица;

Z_3 - стоимость задержки доставки при несоблюдении временных окон,

денежная единица;

Z_4 - стоимость эксплуатации электротрака (общие электротранспортные издержки + стоимость аренды электротранспортного средства), денежная единица;

Z_5 - стоимость потребляемой электроэнергии на зарядной станции электротраком, денежная единица.

После введении функции приспособленности рассматривается подход к построению комбинаций маршрутов для проведения доставки клиентам, обслуживание которых распределено между отдельными депо согласно проведённой ранее процедуре кластеризации. В этом случае исследователи обращаются к эволюционным алгоритмам, чтобы найти оптимальную последовательность маршрутов. Таким образом, осуществляются итерации, направленные на комбинацию возможных вариантов точек назначения для удовлетворения требованиям клиентов, а также системы уравнений оптимизируемой модели и прилагающейся к ней ограничений, путём использования биологических операторов кроссовера, мутации, инверсии и других видов для изменения генов (точек маршрута) с целью получения новых хромосом (полных маршрутов), наиболее удовлетворяющих введённой функции приспособленности. Примером описанной процедуры может служить фрагмент из самой статьи, демонстрирующий работу оператора мутации для воспроизведения решения с отличной последовательностью пунктов назначения в маршруте (*Рисунок 7*):

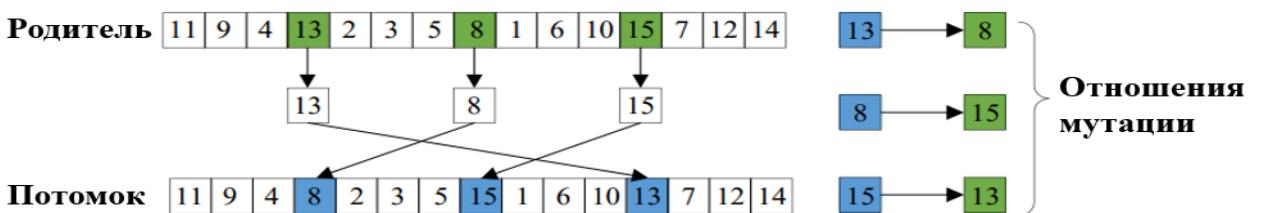


Рисунок 7. Пример работы оператора мутации [12]

Разработанная система тестируется с применением разных стратегий и, в конечном счёте, демонстрирует впечатляющие результаты. Для

осуществление всех поставленных задач потребовалось меньше арендованных электротраков, потребляемой электроэнергии, а также общих операционных издержек, нежели при использовании ранних традиционных методов. В связи с этим можно сделать вывод о том, что совокупность применяемых инструментов в рамках данного подхода является работоспособной и достойной внимания.

Две прошлые приведённые статьи главным образом затрагивали проблему построения маршрутов и резервации зарядных станций в течение какого-то временного окна на его протяжении, но не рассматривали детали, связанные с управлением разными бронями, оформленными ранее и назначенными на одну из зарядных станций. В рамках рассмотрения подходов для построения оптимального расписания резерваций сессий зарядки электрокаров для начала обратимся к исследованию румынских экспертов [13]. В данной статье упоминаются такая проблема как «наслоение» броней, что создаёт очереди перед зарядными станциями, ведущие к снижению уровня удовлетворения сервисами владельцев электрокаров. Во избежание таких случаев авторы предлагают простую математическую модель, применение которой позволит выстроить окна резервации таким образом, чтобы их периоды шли последовательно, а не параллельно друг другу.

Таким образом, выразить основную мотивацию, стоящую за данной статьёй, можно выразить следующей схемой (*Рисунок 8*):

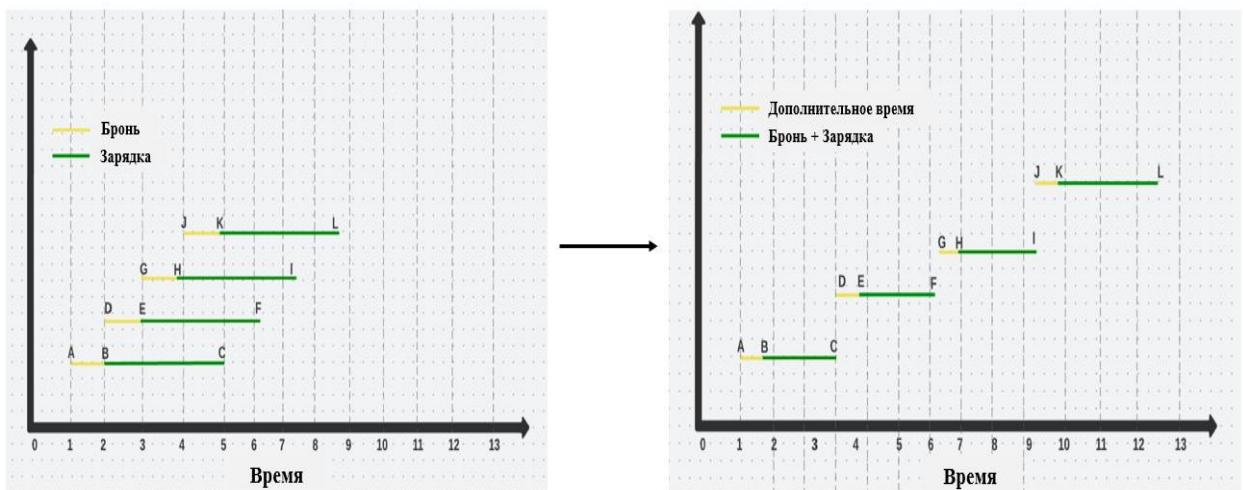


Рисунок 8. Сравнение алгоритмов резервации [13]

Ось абсцисс отображает продолжительность дня на обоих графиках. На первой диаграмме отсутствуют какие-либо ограничения, связанные с продолжительностью сессии зарядки электротранспортного средства, что может стать причиной возникновения «наслоения» резерваций и, следовательно, увеличения длительности времени ожидания в очередях. Во втором случае время брони состоит из буферного времени перед процедурой зарядки, в течение которой текущие клиенты получают уведомление о том, что они должны закончить мероприятие и покинуть позицию, и продолжительностью самого процесса получения электроэнергии. Это позволяет резервации начинаться в выбранной водителем время, а заканчиваться, когда потребитель прерывает процесс зарядки и покидает место. Помимо этого, на второй диаграмме, помимо ведения дополнительного времени, также ведётся расчёт оптимальной продолжительности резервации для каждого отдельного случая для избегания возникновения эффекта «наслоения» при полной зарядке электротранспортного средства:

$$hoursToAdd = finalCoefficient * maxExtraHours, \text{ где } \quad (1.3)$$

hoursToAdd – число дополнительных часов, добавляемых к началу брони, часы;

finalCoefficient – коэффициент продления брони;

maxExtraHours – максимально возможное число дополнительных часов, добавляемых к началу брони (зависит от текущего уровня заряда), часы.

Следующий уровень декомпозиции:

$$hoursToAdd = (0.5 * connectorRequestCoefficient + 0.5 * overlappingCoefficient) * maxExtraHours, \text{ где } \quad (1.3.1)$$

connectorRequestCoefficient – степень эксплуатации зарядной станции;

overlappingCoefficient – степень эксплуатации выбранного временного окна;

Можно приступить ещё к одному уровню декомпозиции, который будет последним:

$$hoursToAdd = \left(0.5 * \frac{lastWeekTransactions}{lastTwoWeeksTransactions} + 0.5 * \frac{lastWeekTransactions^*}{lastTwoWeekTransactions^*} \right) * maxExtraHours, \text{ где} \quad (1.3.2)$$

lastWeekTransactions – число транзакций для резервации зарядной станции на прошлой неделе, ед.;

lastTwoWeekTransactions – число транзакций для резервации зарядной станции в течение двух прошлых недель, ед.;

lastWeekTransactions^{}* – число транзакций с «соприкосновениями» броней на зарядной станции на прошлой неделе, ед.;

lastTwoWeekTransactions^{}* – число транзакций с «соприкосновениями» броней на зарядной станции в течение двух прошлых недель, ед.

Таким образом, водитель электрокара получает достаточное время для зарядки своего электротранспортного средства, в то время как другие потребители электроэнергии для своих средств перемещения будут приезжать в заранее рассчитанное время, которое сведёт к минимуму возникновение очередей. Помимо этого, в статье также проводится практический эксперимент, где разработанная система была протестирована на базе одной из уже существующих платформ. От начала до конца алгоритм успешно предлагал рациональные временные окна на основе уже сделанных броней: «наслоения» расписаний резерваций не наблюдались. В будущем исследователи рассматривают возможность автоматизации ввода таких данных, как бренд автомобиля, модель, тип штекера, текущий уровень заряда и область поиска, путём создания дополнительного программного обеспечения, которое путём регистрации электротранспортного средства сможет получить эти сведения с использованием одной из беспроводных технологий передачи информации.

Также стоит обратиться к опыту китайских учёных [14]. В данной статье рассматривается вопрос разработки системы планирования брони зарядной станции путём применения алгоритмов роя частиц для оптимизации. В рамках этого исследования наибольший интерес представляет часть математической

модели, которая описывает подход к расчёту полного времени сессии зарядки. Таким образом, предлагается следующее уравнение:

$$\hat{c}_{ij} = \begin{cases} t_{ij} + c_{ij}, & \text{если } x_{0,i}^j = 1 \\ \max(\hat{c}_{lj}, t_{ij}) + c_{ij}, & \text{если } x_{l,i}^j = 1 \end{cases}, \text{ где} \quad (1.4)$$

\hat{c}_{ij} – полное время сессии зарядки на j-ого слота зарядной станции для i-ого электротранспортного средства, часы;

t_{ij} – время необходимое для i-ого электротранспортного средства, чтобы достигнуть j-ого слота зарядной станции, часы;

c_{ij} – время необходимое для полной зарядки i-ого электротранспортного средства на j-ом слоте зарядной станции, часы;

\hat{c}_{lj} – время необходимое для уже заряжающегося l-ого электротранспортного для достижения полной зарядки аккумулятора на j-ого слота зарядной станции, часы;

$x_{0,i}^j$ – бинарная переменная условия достижения i-ого электротранспортного средства j-ого слота зарядной станции первым;

$x_{l,i}^j$ – бинарная переменная условия достижения l-ого по счёту электротранспортного средства j-ого слота зарядной станции первым перед i-ым электротранспортным средством.

Таким образом, можно заметить подход, который отличается о тех, что были представлены в предыдущих работах, и заключается в рассмотрении слотов для зарядки как отдельных объектов анализа. Этот взгляд вполне логичен: несколько электротранспортных средств могут заряжаться на одной зарядной станции при наличии одинаковых слотов для их штекеров или же разных слотов для моделей с разными штекерами. Помимо этого, одно из представленных математических ограничений выражено крайне лаконичным уравнением, вычисление которого не требует больших вычислительных ресурсов, что крайне полезно при большом числе объектов, участвующих в системе. Минимизация этого параметра поможет выбрать наиболее подходящую зарядную станцию с позиции минимизации совокупного времени

для её достижения и ожидания, что является одним из самых существенных критериев при проведении многокритериальной оптимизации.

В рамках данного пункта этой теоретической главы были рассмотрены 4 работы зарубежных исследователей, содержащие различные подходы к решению основной задачи разработки системы бронирования электронных зарядной станций. В первой из них [11] рассматривается решение вопроса с помощью разбития местности на полный ориентированный граф с последующим применением многокритериальной оптимизации, основанной на простом переборе возможных вариантов. Во второй работе [12] задача уже решается без введения графа, а также простой перебор заменяется эволюционными алгоритмами, а многокритериальная оптимизация сменяется на однокритериальную. В третьем исследовании [13] уже рассматриваются вопросы по определению оптимальной продолжительности брони после успешного выбора зарядной станции для использования с целью уменьшения времени ожидания пользователя. В четвёртой работе [14] наблюдается смена подхода от представления зарядной станции как объекта моделирования к рассмотрению отдельного слота зарядной станции как такового.

Каждая из рассмотренных статей обладает уникальным потенциалом, который может быть адаптирован в рамках решения поставленной задачи в этой работе. Отбор отдельных элементов из них, а также их интеграция будут выполнены в методологической части после рассмотрения конкретных реальных условий решаемой проблемы. Тем не менее, на данный момент времени можно заявить, что в связи с успехом прошлого рассмотренного опыта будет использован эвристический метод, опирающийся на введении грамотной математической модели.

Выводы по главе 1

Развитие веб-приложений тесно связано с распространением сети Интернет и её развитием. Большинство исследователей выделяют 3 стадии развития сети Интернет, в прогрессии которых веб-приложения получали

новые отличительные черты и функции, определяющие их конкурентные преимущества на рынке. Рассмотрение текущей совокупности возможностей веб-приложений позволяет ответить на вопрос о том, какими отличительными чертами они должны обладать, чтобы считаться имеющими свойства актуальности и интеллектуальности.

Процесс разработки веб-приложений включает в себя два основных компонента: фронт-энд (графический интерфейс для взаимодействия пользователя) и бэк-энд (основной инкапсулированный алгоритм, решающий задачи согласно требованием пользователя). Помимо этого, упомянутые элементы связываются между собой некоторыми виртуальными сущностями, например, такими как веб-сервер и база данных. Первая позволяет коммуницировать фронт-энду и бэк-энду между собой, в то время как вторая – позволять им работать с данными. Успешное веб-приложение успешно интегрирует и координирует в себе работу всех компонентов, что позволяет ему предлагать оптимальные решения пользователю с затратой на это наименьшего объёма ресурсов. Также для их разработки используются уже зарекомендовавшие себя инструменты, которые отличаются своей узкой направленностью.

Существует достаточный зарубежный практический опыт решения проблемы бронирования зарядных станций для электрокаров. Каждый из них предлагает свои эвристические методы для решения этой задачи. Именно разнообразие используемых подходов позволяет получить массив возможных альтернатив, элементы которых могут быть интегрированы между собой для построения уникальной системы со свойствами, представляющим возможность в удовлетворении воздвигаемых на неё требований в конкретно рассматриваемых условиях, присущих каждому отдельному практическому случаю.

ГЛАВА 2. МЕТОДОЛОГИЧЕСКАЯ ЧАСТЬ

2.1. Формирование перечня параметров эксплуатации ЭЗС

В методологической части данной работы стоит обратить внимание на конкретные условия поставленной задачи бронирования электронных зарядных станций. В первую очередь существует необходимость в рассмотрении параметров и разновидностей ЭЗС, чтобы сформировать перечень критериев, которые определяют совместимость процесса их эксплуатации для отдельных технических характеристик электротранспортного средства владельца. Другими словами, необходимо сформировать массив параметров зарядных станций с целью фильтрации подходящих из них для конкретного агента, ищущего возможность в её использовании.

Таким образом, обратимся к опыту одного из ведущих российских разработчиков и производителей электротехнической преобразовательной техники «Парус электро» [15], который играет большую роль в развитии инфраструктуры зарядных станций в городе Москве. Стоит изучить каталог предлагаемых им услуг в виде установки зарядных станций под ключ [16], так как это позволит составить перечень существующих параметров данной продукции. При рассмотрении характеристик представляемых агрегатов, главным образом, есть необходимость в рассмотрении выходных параметров зарядных станций, так как именно они определяют совместимость технических характеристик электрокаров и требований их владельцев с конкретным объектом зарядной инфраструктуры, в то время как входные параметры – область внимания команд проектировщиков. Таким образом, у ЭЗС имеются следующие выходные параметры:

- тип коннектора, {Type2, CHAdeMO, ...};
- тип электрического тока, {AC, DC};
- максимальная выходная мощность, кВт;
- число выходных кабелей, шт.;

- стоимость потребления 1 кВт, руб.

Рассмотрим параметр ЭЗС, описывающий её тип коннектора. Данная характеристика определяет возможность электрокара с подходящим типом разъёма или наличием у его владельца подходящего переходящего адаптера быть подключённым к зарядной станции для потребления электроэнергии. Другими словами, этот параметр является фильтрационным механизмом для ЭЗС, не совместимых с разъёмом данной модели электрокара или имеющимся у водителя адаптером. Для определения примерного перечня возможных типов коннекторов, которые могут быть найдены в России, можно обратиться к опыту российского производителя зарядных станций для электромобилей и программного обеспечения для работы зарядной инфраструктуры «TOUCH» [17], который предоставляет справочную информацию такого рода на своём официальном веб-сайте (*Рисунок 9*):

	Япония	Европа	США	Китай	Tesla
AC (переменный ток)					
DC (постоянный ток)					

Рисунок 9. Разъёмы автомобилей, встречающихся в России [17]

Параметр, характеризующий тип электрического тока, может принимать только два значения: переменный (AC) и постоянный (DC) [18]. Мощность, поступающая из электрической сети, всегда является переменным током, но аккумуляторная батарея электромобиля может принимать только постоянный

ток. Основное различие между зарядной станцией с переменным и постоянным током заключается в том, где происходит преобразование переменного тока. Его можно преобразовать как снаружи (преобразователь станции), так и внутри автомобиля (бортовое зарядное устройство электромобиля). ЭЗС с постоянным током обычно больше по габаритам, так как ранее упомянутый преобразователь находится внутри них, и стоят они дороже, а их установка отличается большей стоимостью, потому что требуются подключение к электрической сети с высокой мощностью и внедрение дополнительных элементов активного охлаждения, что, главным образом, сказывается на конечной стоимости потребляемой электроэнергии для потребителя в связи с большим периодом окупаемости. Однако главным преимуществом зарядных станций с постоянным током является скорость зарядки: она больше, чем у устройств с переменным током, так как преобразование тока происходит в зарядной станции, минуя бортовое зарядное устройство автомобиля, то есть он сразу достигает аккумулятора электротранспортного средства, что экономит временные ресурсы. Помимо этого, стоит отметить, что также между переменным и постоянным током существуют отличия в форме кривой зарядки (**Рисунок 10**):

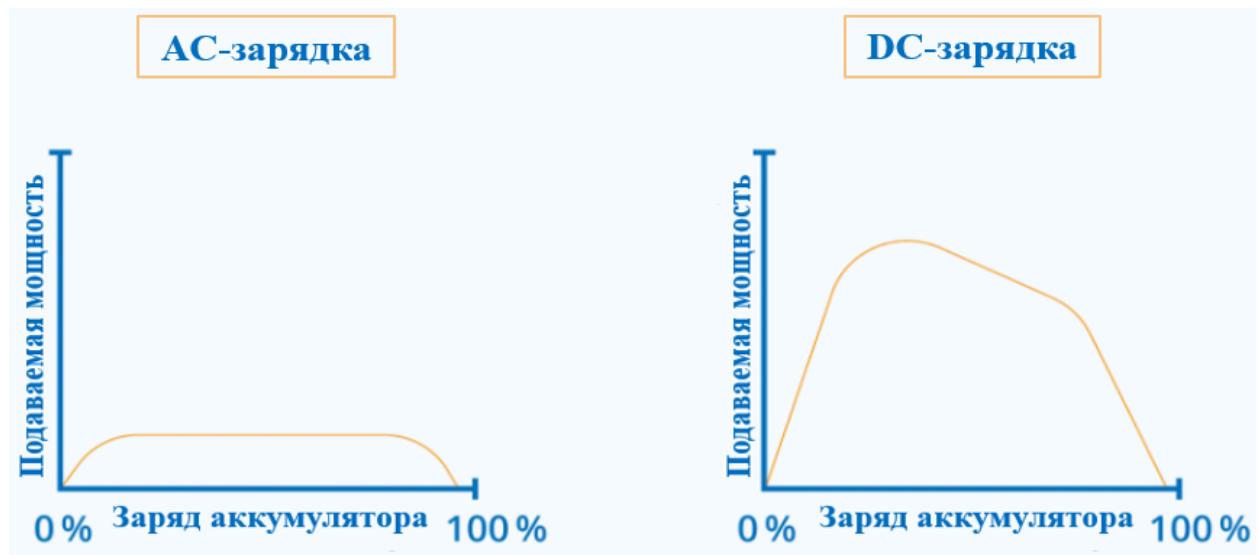


Рисунок 10. Кривые зарядки при разных типах электрического тока [18]

Пологая форма графика при зарядке с переменным током обусловлена малым размером бортового зарядного устройства и, соответственно, его

ограниченной мощностью. Горбообразная кривая при выполнении процедуры зарядки с помощью постоянного тока объясняется аккумуляторной батареей электромобиля, которая сначала принимает более быстрый поток электроэнергии, но постепенно потребляет меньше электроэнергии, когда она достигает максимальной ёмкости.

Приступим к рассмотрению параметра, выражающего выходную мощность зарядной станции. Хотелось бы отметить, что, помимо этой характеристики, также указываются такие величины как сила тока и напряжения, но их произведение составляет мощность тока, поэтому достаточно учёта одного заявленного параметра [19]:

$$P = U * I, \text{ где} \quad (2.1)$$

P – мощность тока, Вт;

U – электрическое напряжение на участке цепи, В;

I – сила тока, А.

Его значение напрямую зависит от прошлого параметра: зарядные станции с переменным током обладают меньшей мощностью, в то время как с постоянным – большей. Стоит упомянуть, что чем больше мощность ЭЗС, тем быстрее осуществляется процесс зарядки, однако, износ аккумулятора при таких режимах эксплуатации возрастает [20]:

$$t = E/P, \text{ где} \quad (2.2)$$

t – продолжительность зарядки, ч;

E – объём полученной электроэнергии аккумулятором, кВт·ч;

P – мощность тока, кВт.

Зарядные станции с переменным током и пониженной мощностью, как правило, устанавливают в пределах городской черты, где есть возможность оставить электротранспортное средство на зарядке на больший период времени и заняться каким-либо досугом. ЭЗС с постоянным типом тока и повышенными показателями мощности чаще всего внедряют на трассах, где существует необходимость в оперативной зарядке электрокара для

продолжения следования по длительному маршруту.

Рассмотрим параметр ЭЗС, характеризующий число выходящих из неё кабелей. Этот критерий также можно назвать «пропускной способностью» зарядной станции, так как он определяет максимальное число электротранспортных средств, которые могут подвергаться процедуре зарядки от неё одновременно. Стоит упомянуть, что, как правило, в ЭЗС с несколькими выходными кабелями, так же называемыми «слотами», используются разные виды штекеров для разных типов разъёмов с целью увеличения спектра возможных моделей электрокаров, способных к подключению к ней.

Значения, которые принимает последний параметр, стоимость электроэнергии, напрямую зависит от всех ранее упомянутых критериев. На формирование тарифов влияют местоположение зарядной станции, сложность её интеграции с локальным участником инфраструктуры, число имеющихся у неё слотов, её мощность, вид используемого тока, перечень имеющихся коннекторов, а также и другие факторы, связанные с амортизацией оборудования и ожиданиями её владельца по величине доходности. Стоит отметить, что, как правило, цена указывается за потребление 1 кВт·ч, что облегчает задачу пользователю по расчёту конечной стоимости зарядки при знании текущего уровня зарядки электрокара, а также учитывает временные ресурсы, сопряжённые с процедурой потребления электроэнергии электротранспортным средством.

Таким образом, в данном пункте металогической части работы удалось сформировать перечень критериев, которые определяют совместимость процесса эксплуатации зарядных станций для отдельных технических характеристик электротранспортного средства владельца. Полученная информация позволит выстроить процесс поиска ЭЗС наиболее точным образом с фокусировкой внимания исключительно на тех объектах, которые удовлетворяют техническим характеристикам электрокара и требованиям его водителя, что позволит существенно сократить объём привлекаемых для этого временных и вычислительных ресурсов.

2.2. Формирование перечня внутренних и внешних факторов влияния на эксплуатацию электрокаров

В данном пункте методологической части сместим фокус внимания с зарядных станций на электротранспортные средства. Некоторые параметры электрокаров также оказывают влияние на выбор оптимальной ЭЗС и механику построения маршрутов, что делает их обязательными для учёта при решении задачи бронирования зарядных станций. Другими словами, необходимо сформировать массив параметров для электротранспортных средств с целью установки зависимости изменения величин переменных эксплуатации электрокаров от сопутствующих условий для будущей математической модели представляемого метода решения поставленной задачи.

Факторы влияния на использование электрокаров можно разделить на два вида: внутренние и внешние. К внутреннему типу относятся те параметры, которые обусловлены текущими техническими характеристиками электротранспортного средства, в то время как к внешнему – окружающей средой, состояние которой не зависит от деталей эксплуатационного процесса электрокара.

Таким образом, к внутренним факторам влияния на эксплуатацию электротранспортного средства можно отнести:

- тип разъёма гнезда для зарядки, {Type2, CHAdeMO, ...};
- расход электроэнергии на 100 км пробега, кВт·ч/км;
- текущий уровень заряда аккумулятора, кВт·ч;
- ёмкость аккумулятора, кВт·ч.

Про первый параметр, тип разъёма гнезда для зарядки, можно заявить, что он тесно связан с первой характеристикой ЭЗС, типом коннектора. Другими словами, их обобщенная совместимость играет решающую роль в возможности электротранспортного средства в получении электроэнергии от конкретной ЭЗС. Однако стоит заметить, что технические характеристики электрокара не всегда определяют его коммуникабельность с зарядной

станцией по той причине, что его владелец может обладать дополнительным адаптером, служащим переходным звеном между разными коннектором и зарядным гнездом. Именно по этой причине следует предоставить выбор разъёма гнезда для зарядки самому пользователю, нежели полагаться на заводские параметры модели электротранспортного средства.

Обратим внимание на второй фактор, который выражает расход электроэнергии на 100 км пробега и является аналогом привычного показателя у традиционных автомобилей с двигателем внутреннего сгорания под названием «расход топлива на 100 км». Другими словами, данная характеристика отражает, сколько заряда аккумулятора необходимо потратить, чтобы преодолеть дистанцию, длинною в 100 км, без остановок. Данное заявление можно выразить простой формулой:

$$Q = (E * 100) / S, \text{ где} \quad (2.3)$$

Q – расход электроэнергии на 100 км пробега, кВт·ч/км;

E – объём затрачиваемой электроэнергии для S , кВт·ч;

S – преодолеваемое расстояние, км.

Этот параметр помогает в определении величины предельно допустимой протяженности пути при текущем фиксируемом уровне заряда аккумулятора, то есть он характеризует физическую возможность достижения предполагаемой точки электротранспортным средством с имеющимся у него на это ресурсами. Эта информация крайне полезна в процессе построения маршрута с включением в него какой-либо ЭЗС в связи с тем, что она может определить возможность её достижения. Более того, стоит упомянуть, что двигатели современных электрокаров обладают КПД равным от 85–95%, в то время как автомобили с двигателями внутреннего сгорания только 40% выделяемой электроэнергии преобразуют в полезное механическое движение, а остальное – в тепло [21].

Третий фактор, выражающий текущий уровень заряда аккумулятора, является решающим. Он определяет возможность продвижения

электротранспортного средства на конкретную величину расстояния в зависимости от расхода электроэнергии на 100 км пробега и внешних факторов, которые будут рассмотрены позже:

$$d = (C * 100)/Q, \text{ где} \quad (2.4)$$

d – предельное преодолимое расстояние, км;

C – текущий уровень зарядки аккумулятора, кВт·ч;

Q – расход электроэнергии на 100 км пробега, кВт·ч/км;

Данный параметр является аналогом датчиков уровня топлива в традиционных автомобилях с двигателями внутреннего сгорания. Единицей измерения заряда аккумулятора электротранспортного средства является «Киловатт-час», которая выражает количество электроэнергии, произведённое электрическим устройством мощностью 1 киловатт за 1 час своей работы. Текущий уровень зарядки аккумулятора является важным параметром в связи с тем, что при решении задачи бронирования зарядной станции важны текущие количественные величины факторов.

Последним фактором является ёмкость аккумулятора электротранспортного средства, которая диктуется его заводскими техническими характеристиками. Данный параметр может выражаться как в единице измерения «ампер-час», так и в уже привычной «киловатт-час». Он необходим для расчёта запаса хода после осуществления процедуры полной зарядки на ЭЗС. Батареи электротранспортных средств являются литий-ионными, что делает их восприимчивыми к показателям температуры внешней среды. Помимо этого, их ёмкости условно принимаются как постоянные при нормальной эксплуатации и разнятся от модели к модели электрокаров, оснащёнными ими.

К внешним факторам влияния на эксплуатацию электротранспортного средства можно отнести:

- дорожно-транспортная обстановка;
- метеорологические условия;

- стиль вождения владельца (дополнительно).

Касаемо фактора, отражающего состояние дорожного трафика на местности, по которой проходит маршрут электротранспортного средства, можно заявить, что этот критерий также заслуживает внимания при решении поставленной в этой работе задачи. В качестве составляющих, влияющих на дорожно-транспортную остановку, можно назвать пробки, аварии, ремонт дорожного покрытия, гололёд и другие дисфункции, которые нарушают протекание привычного процесса движения трафика. В свою очередь, они способны повлечь за собой увеличение времени прохождения маршрута и объёма потребляемой для его преодоления электроэнергии, что будет иметь непосредственное влияние на выбор подходящих ЭЗС и механику построения пути исходя из ранее перечисленных условий. Фиксировать такие факторы и рассчитывать эффект от их существования в таком показателе, как время прохождения маршрута, могут распространённые картографические сервисы. Например, наиболее популярными примерами могут служить «Google Карты» [22] и «Яндекс Карты» [23] со встроенными функциями по отслеживанию пробок, ремонта дорог и аварий, имеющих место быть на дорожно-транспортной инфраструктуре в переделах конкретной местности. Всё сказанное говорит о том, что возникает необходимость в обращении к дополнительным инструментам для расчёта количественной оценки такого фактора внешней среды, как дорожно-транспортная обстановка, который может быть воспроизведен с помощью стороннего программного решения.

Вторым фактором влияния являются метеорологические условия, которые оказывают непосредственный эффект на работу аккумулятора. Они способны изменять продолжительность процедуры зарядки, а также увеличивать расход электроэнергии, тем самым уменьшая максимально допустимую протяжённость маршрута при текущих показателях уровня заряда аккумулятора. Именно по этим причинам важность этого фактора внешней среды не может быть приуменьшена. Стоит отметить, что, согласно большому числу исследований [24] [25], оптимальной температурой для

эксплуатации электротранспортного средства, при которой литий-ионный аккумулятор демонстрирует оптимальные показатели эффективности, является температура 20 ± 1.5 °С. Логично предположить, что такая температура воздуха преобладает далеко не во всех климатических поясах, поэтому следует рассмотреть количественные зависимости параметров функционирования аккумуляторов электрокаров в погодных условиях, отличных от оптимальных.

Увеличение потребления электроэнергии электротранспортными средствами при отклонениях в обоих направлениях от оптимальных показателей температуры воздуха происходит из-за необходимости систем электрокара в выделении отдельного объёма электричества для охлаждения или прогревания салона и аккумулятора с целью обеспечения продолжения движения [26] [27]. С использованием регрессионного анализа для построения уравнения зависимости температуры окружающей среды от величины расхода электроэнергии можно выявить следующий график для разных моделей электрокаров (*Рисунок 11*):

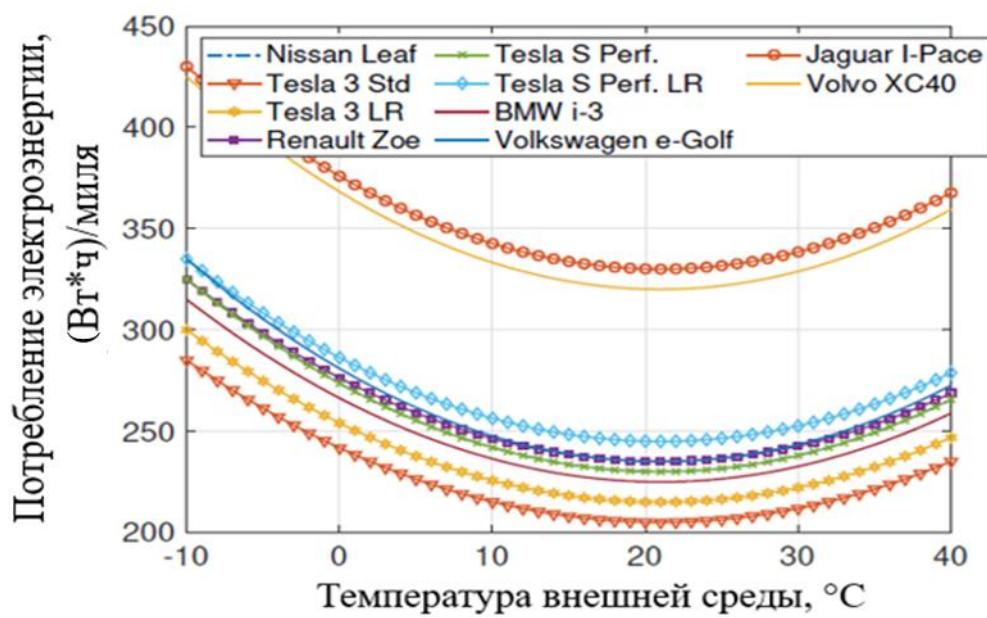


Рисунок 11. Зависимость расхода электроэнергии и температуры окружающей среды [24]

Можно заметить, что паттерн одинаков для всех рассматриваемых моделей электротранспортных средств. При снижении температуры к -10 °С расход электроэнергии электротранспортными средствами увеличивается более, чем

на одну треть от их оптимального значения. Данная зависимость для единицы измерения «Вт·ч/миля» будет так же справедлива и для «кВт·ч/км». С помощью метода наименьших квадратов можно аппроксимировать уравнение параболы, устанавливающей зависимость потребления объёма электричества аккумулятором от температуры окружающей среды:

$$Q_m = 5.6 * 10^{-4} Q_{opt} (T^2 - 40T + 2200), \text{ где} \quad (2.5)$$

Q_m – потребление электроэнергии на 100 км, кВт·ч/км;

Q_{opt} – оптимальное потребление электроэнергии на 100 км, указанное в

технических характеристиках электрокара (при $T = 20^\circ\text{C}$), кВт·ч/км;

T – температура окружающей среды, $^\circ\text{C}$;

Помимо увеличения потребления объёма электроэнергии при движении по маршруту, метеорологические условия также оказывают влияние на процесс зарядки электротранспортного средства [24] [28]. Во время неблагоприятной температуры воздуха система управления аккумулятором в электрокарах снижает темп приёма электроэнергии, чтобы избежать нанесения урона батарее во время процедуры зарядки, что влечёт за собой увеличение продолжительности цикла процесса зарядки. Агрегация показателей сессий зарядок электротранспортных средств разных моделей позволяет построить следующий график (**Рисунок 12**):

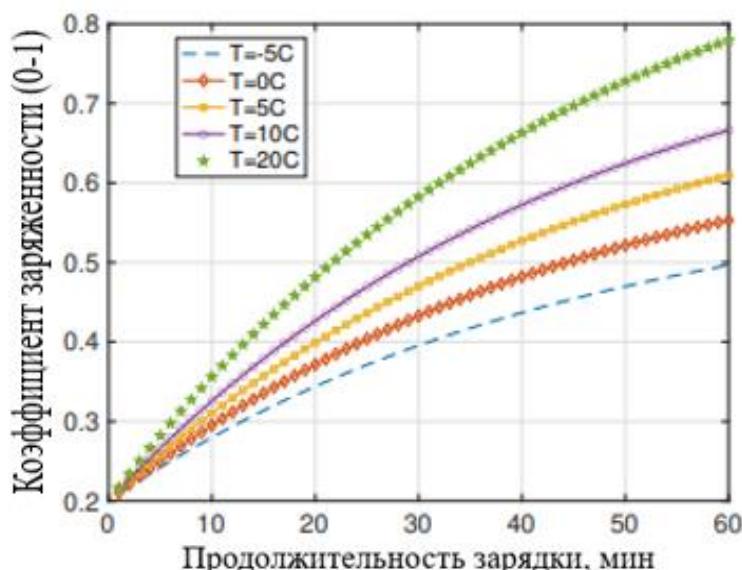


Рисунок 12. Зависимость продолжительности сессии зарядки и погодных условий [24]

Можно заметить, что при уменьшении температуры окружающего воздуха время, требуемое для полной зарядки аккумулятора, увеличивается. Причём, чем дольше длится сессия зарядки, тем больше разрыв между степенями заряженности аккумулятора при разных погодных условиях, что напоминает паттерн экспоненциальной функции. Авторы исследования также предоставляют уравнение, объясняющее рассмотренную зависимость, однако стоит заметить, что оно учитывает влияние температур только ниже оптимального значения, равного примерно 20°C [24]. Сформируем систему уравнений с учётом температурных периодов, относительно которых они справедливы:

$$\begin{cases} \begin{cases} CH(t) = \left(CH_0 - \frac{0.015+0.00034T}{0.022} \right) e^{-0.022t} + \frac{0.015+0.00034T}{0.022} \\ T < 20 \end{cases} \\ \begin{cases} CH(t) = (CH_0 - 1)e^{-0.022t} + 1 \\ T \geq 20 \end{cases} \end{cases}, \text{где } (2.6)$$

$CH(t)$ – коэффициент заряженности, набранный за время t сессии зарядки;

$CH_0 = \frac{C}{C_{max}}$ – начальный коэффициент заряженности за время перед

сессией зарядки;

t – продолжительность сессии зарядки, мин;

T – температура окружающей среды, °С.

Продолжительность времени зарядки может быть переведена с минут в часы, тогда уравнение примет вид:

$$\begin{cases} \begin{cases} CH(t) = \left(\frac{C}{C_{max}} - \frac{0.015+0.00034T}{0.022} \right) e^{-1.32t} + \frac{0.015+0.00034T}{0.022} \\ T < 20 \end{cases} \\ \begin{cases} CH(t) = \left(\frac{C}{C_{max}} - 1 \right) e^{-1.32t} + 1 \\ T \geq 20 \end{cases} \end{cases}, \text{где } (2.6.1)$$

C – уровень зарядки аккумулятора электрокара перед процедурой зарядки, кВт·ч;

C_{max} – предельная ёмкость аккумулятора, установленная техническими

характеристиками электрокара, кВт·ч;

В связи с этим формула для определения количества часов, необходимых для полного заряда аккумулятора ($\lim_{t \rightarrow \infty} (CH(t)) = 1$) выглядит следующим образом:

$$\begin{cases} \left\{ \begin{array}{l} t = 0.76 \log \left(\frac{\frac{C}{C_{max}} - 17T + 750}{17T + 339} \right) \\ T < 20 \end{array} \right. \\ \left\{ \begin{array}{l} t = 0.76 \log \left(100 \left(1 - \frac{C}{C_{max}} \right) \right) \\ T \geq 20 \end{array} \right. \end{cases} \quad (2.6.2)$$

Таким образом, можно заключить, что погодные условия оказывают весомое влияние на эксплуатацию аккумулятора электротранспортного средства. Все вышеперечисленные уравнения должны быть учтены в будущей математической модели разрабатываемого решения.

Последним внешним фактором, имеющим эффект на показатели работы электрокара, является стиль вождения его владельца. Например, данный тезис подтверждается практическим исследованием, занимающимся установлением взаимосвязи между стилем вождения электротранспортного средства и объёмом потребления имеющейся у него электроэнергии во втором по населённости городе Великобритании – Бирмингеме [29]. Исходя из величины параметра, характеризующего число мгновенных ускорений электрокара на всём протяжении заданного маршрута, удается выделить 3 категории наблюдаемого стиля вождения: пассивный, умеренный и агрессивный. Сопоставление потребляемой единицы объёма электроэнергии на километр и категорий стиля вождения позволяют построить следующую столбчатую диаграмму (**Рисунок 13**):

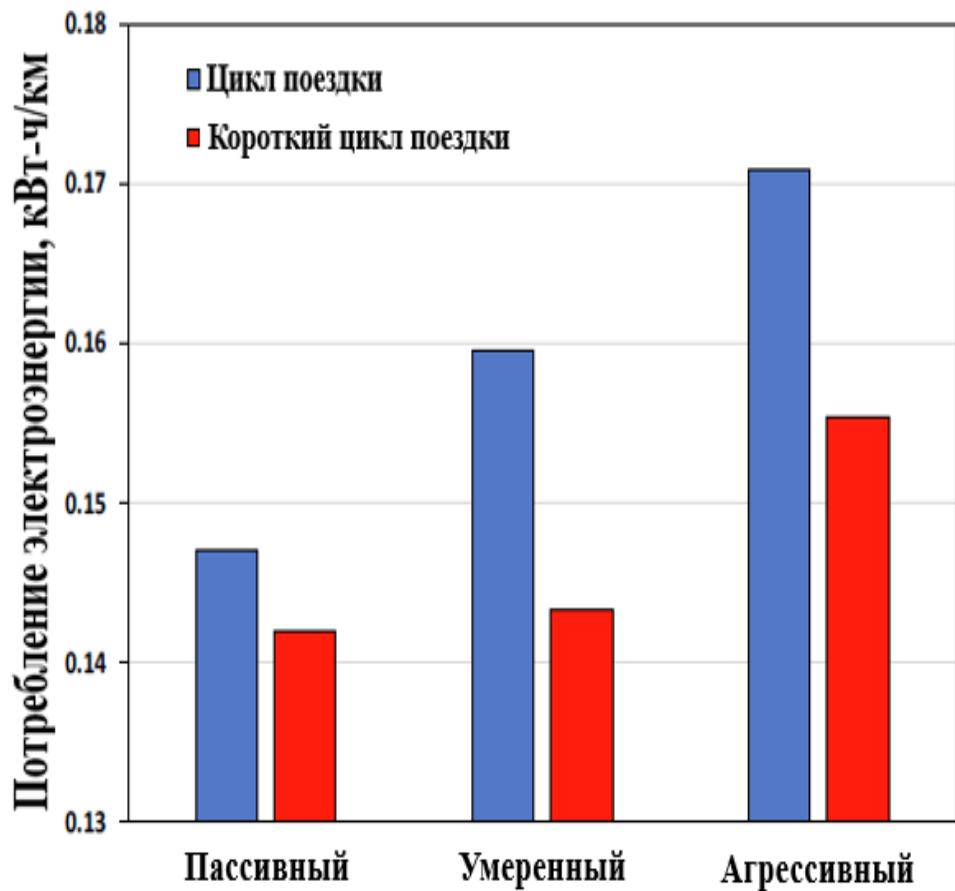


Рисунок 13. Зависимость расхода электроэнергии и стиля вождения [29]

Можно заметить, что с увеличением числа мгновенных ускорений как на коротких маршрутах, так и на средних и длинных наблюдаются увеличения показателя расхода электроэнергии на километр. Более того, на коротких поездках эта зависимость выражается более остро. Пусть увеличение потребления электроэнергии и колеблется в ограниченном диапазоне от 5–10% при смене стиля вождения от умеренного к агрессивному, но этот фактор по-прежнему заслуживает упоминания и внимания.

Несмотря на то, что характер управления электротранспортным средством его водителем влияет на показатели работы электрокара, этот параметр не представляется возможным к измерению в рамках инструментария, используемого в данной работе, так как он требует привлечение дополнительных технических средств. В связи с этим, математическая модель будет упрощена и не будет учитывать стиль вождения владельца электротранспортного средства.

Таким образом, в данном пункте нам удалось рассмотреть как внутренние, так и внешние факторы влияния на эксплуатацию электрокара. Это позволит определить величину изменения параметров его использования в зависимости от сторонних факторов, действующих на него. Все перечисленные аспекты должны быть затронуты при составлении математической модели.

2.3. Адаптация метода интеллектуального управления эксплуатацией ЭЗС к текущим условиям задачи

Итогом методологической части данной работы должен являться специально разработанный метод, который способен решить поставленные задачи. В связи с этим в этом пункте нам предстоит составить математическую модель, которая будет строиться на исследованном опыте в теоретической части и учитывать выявленные условия функционирования системы в методологической части. Другими словами, следует консолидировать все разобранные аспекты для составления решения поставленной задачи воедино путём введения объективной математической модели, позволяющей успешно отражать объективную реальность рассматриваемой проблемы.

Таким образом, исходя из ценного практического опыта индийских и китайских специалистов [11] [12] можно сделать вывод о том, что оптимизация совокупной стоимости всех процессов, протекающих при осуществлении прохождения электрокара по маршруту с остановками на подзарядку, может считаться фундаментальной метрикой, к минимизации которой нужно стремиться. Более того, представление совокупности ЭЗС, начальной и конечной точек маршрута в качестве вершин полного ориентированного графа, поможет сделать процесс моделирования дискретным. В связи с этим попробуется разбить процесс над подпроцессы с использованием блок-схемы, чтобы получить понимание того, стоимость чего стоит включать многоцелевую функцию (*Рисунок 14*):

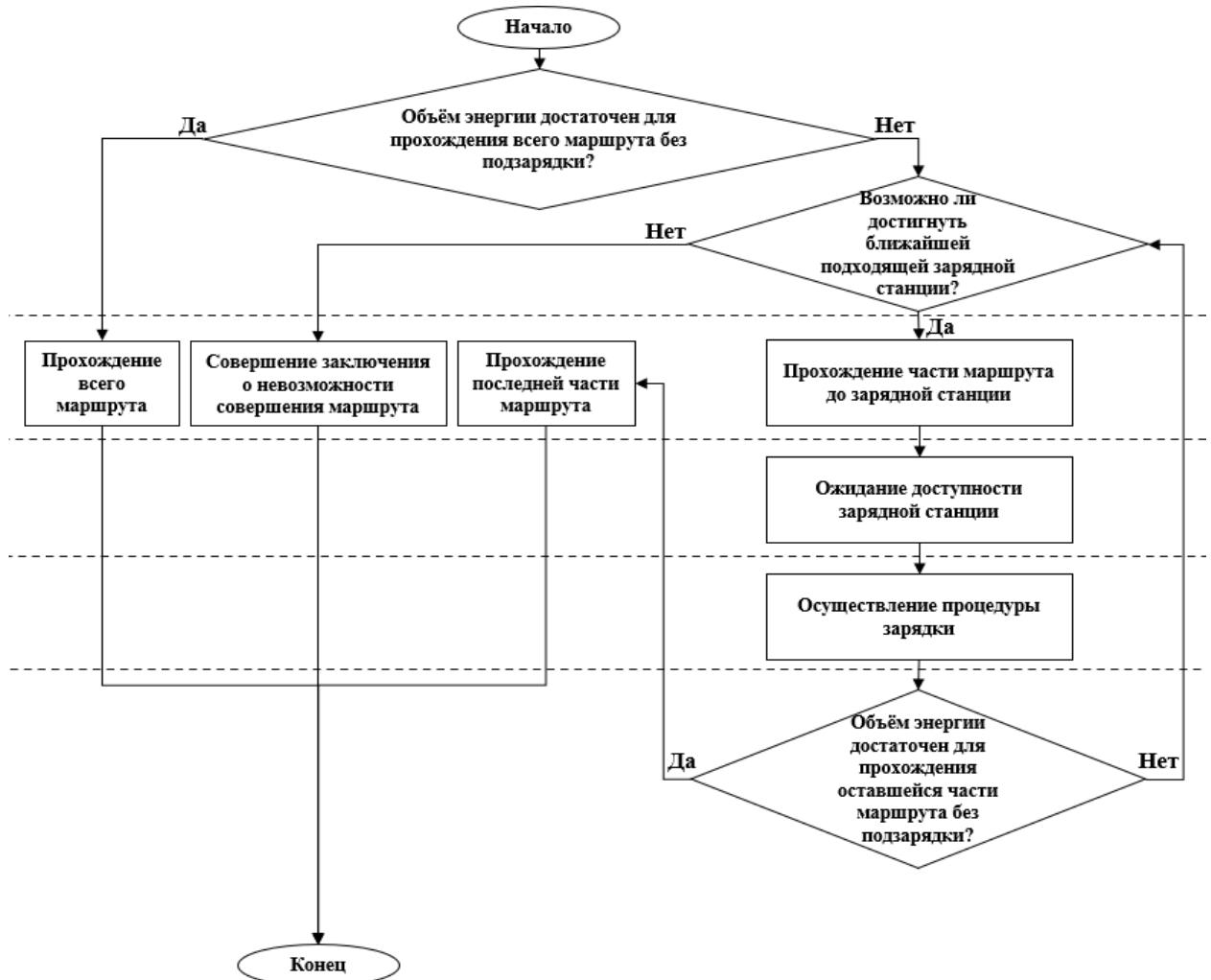


Рисунок 14. Блок-схема процессов цикла поездки

Таким образом, можно выделить 3 основных повторяющихся во времени процесса, стоимостные издержки протекания которых должны быть минимизированы:

- прохождение частей маршрута от начальной точки/текущей зарядной станции до конечной точки/следующей подходящей зарядной станции;
- ожидание доступности зарядной станции;
- осуществление процедуры зарядки.

Попробуем представить их с помощью самого верхнего уровня целевой функции:

$$Z = Z_s + Z_w + Z_{ch}, \text{ где} \quad (2.7)$$

Z – совокупная стоимость цикла поездки, руб.;

Z_s – стоимость поездок от начальной точки/текущей зарядной станции до

конечной точки/следующей подходящей зарядной станции, руб.;

Z_w – стоимостное выражение потерянного времени при ожидании

доступности подходящей зарядной станции, руб.;

Z_{ch} – стоимость процедуры зарядки для электромобиля, руб.

Используя все рассмотренные в методологической части этой работы, внутренние и внешние факторы, сформируем следующий уровень декомпозиции разрабатываемой целевой функции. Стоит отметить, что согласно ранее упомянутому практическому опыту китайских специалистов [13] конечной единицей моделирования будет служить не сама зарядная станция, а слоты в ней. Более того, стоимостное выражение времени, затрачиваемого во время прохождения частей маршрута с учётом дорожно-транспортных условий, ожидания освобождения ЭЗС и зарядки на ней, будет рассчитано путём его отождествления с упущенными расстоянием, которое могло бы быть преодолено электрокаром в этот временной промежуток, выраженным в количестве энергии для осуществления этого с переводом в конечную стоимость согласно усреднённому тарифу. По этой причине стоит ввести константы: средняя скорость легкового транспортного средства в современном мегаполисе равняется 45 км/ч [30], а средний тариф за потребление 1 кВт·ч электротранспортным средством на зарядной станции – 15 рублей [31]. Помимо этого, представление совокупности возможных частей маршрута удобно сделать дискретным, то есть представить начальную и конечную пункты, а также ЭЗС как вершины полного ориентированного графа [11]. В связи с вышеизложенным следующий уровень декомпозиции целевой функции выглядит следующим образом:

$$Z = \sum_i^{i\epsilon[1;N]} \sum_j^{j\epsilon[1;N]} x_{i,j} \left(\frac{Qr}{100} (S_{i,j} + V t_{i,j}^{dist}) + \right. \\ \left. + \sum_m^{m\epsilon[1;M]} x_{j,m} \left(\frac{QrV}{100} (t_{j,m}^{wait} + t_{j,m}^{charge}) + r_{j,m} (C_{max} - C_j) \right) \right),$$

где (2.7.1)

$x_{i,j}$ – бинарная переменная выбора поездки из вершины i в j , {0, 1};

N – общее число вершин в построенном ориентированном графе;

Q – расход электроэнергии на 100 км пробега, кВт·ч/км;

$r = 15$ – средний тариф за потребление 1 кВт·ч на зарядных станциях в мегаполисе, руб.;

$S_{i,j}$ – длина ребра с вершинами i, j (части маршрута), км;

$V = 45$ – средняя скорость движения легкового транспортного средства в мегаполисе, км/ч;

$t_{i,j}^{dist}$ – время, затрачиваемое электрокаром для прохождения ребра с вершинами i, j (части маршрута), ч;

$x_{j,m}$ – бинарная переменная выбора слота m зарядной станции в вершине j , $\{0, 1\}$;

M – общее число подходящих слотов ЭЗС;

$t_{j,m}^{wait}$ – время, затрачиваемое водителем на ожидание доступности слота m зарядной станции в вершине j , ч;

$t_{j,m}^{charge}$ – время, затрачиваемое водителем на потребление электроэнергии на слоте m зарядной станции в вершине j , ч;

$r_{j,m}$ – тариф за потребление 1 кВт·ч, установленный для слота m зарядной станции в вершине j , руб.;

C_{max} – предельная ёмкость аккумулятора, установленная техническими характеристиками электрокара, кВт·ч;

C_j – уровень зарядки аккумулятора электрокара перед процедурой зарядки на зарядной станции в вершине j , кВт·ч;

Как было изложено ранее в методологической части, метеорологические условия являются внешними факторами эксплуатации электрокаров [24] [25], [26], [27], [28]. В связи с этим стоит осуществить разложение переменных Q (2.5.1) и $t_{j,m}^{charge}$ (2.6.2), величины которых поддаются влиянию температур окружающей среды. С учётом этого фактора финальная декомпозиция целевой

функции будет выглядеть следующим образом:

$$Z = \begin{cases} \sum_i^{i \in [1;N]} \sum_j^{j \in [1;N]} x_{i,j} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)r}{100} (S_{i,j} + V t_{i,j}^{dist}) + \sum_m^{m \in [1;M]} x_{j,m} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)rV}{100} (t_{j,m}^{wait} + 0.76 \log \left(\frac{1100 \frac{c}{c_{max}} - 17T + 750}{17T + 339} \right)) + r_{j,m} (C_{max} - C_j) \right) \right) \\ T < 20 \end{cases}, \text{ где}$$

$$Z = \begin{cases} \sum_i^{i \in [1;N]} \sum_j^{j \in [1;N]} x_{i,j} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)r}{100} (S_{i,j} + V t_{i,j}^{dist}) + \sum_m^{m \in [1;M]} x_{j,m} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)rV}{100} (t_{j,m}^{wait} + 0.76 \log \left(100(1 - \frac{c}{c_{max}}) \right)) + r_{j,m} (C_{max} - C_j) \right) \right) \\ T \geq 20 \end{cases} \quad (2.7.2)$$

Q_{opt} – оптимальный расход электроэнергии на пробег 100 км, указанный в технических характеристиках электрокара, кВт·ч/км;
 T – текущая температура окружающего воздуха, °С.

Несмотря на введённую целевую функцию (2.7.2), минимизация которой должна предоставить наилучшую комбинацию частей маршрута с точки зрения совокупной стоимости их прохождения, стоит обратить внимание на некоторые ограничения для неё, необходимые для исключения тех вариантов, которые физически или технически не осуществимы. Согласно **Рисунок 14**, после прохождения каждой части пути нужно проверять достаточность объёма имеющейся электроэнергии для определения возможности достижения какого-либо из следующих пунктов маршрута для приближения к конечному пункту назначения. Помимо этого, необходимо исключить логические ошибки, связанные с теорией графов, по типу движения из одной вершины в ту же самую, возвращения к стартовой вершине и продолжения

движения из финальной вершины. Таким образом, с учётом всего ранее сказанного ведённая целевая функция с сопутствующими ей ограничениями, составляющими математическую модель вводимого метода, является математической моделью, которая будет выглядеть следующим образом:

$$\left\{
 \begin{array}{l}
 Z = \sum_i^{i \in [1;N]} \sum_j^{j \in [1;N]} x_{i,j} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)r}{100} (S_{i,j} + \right. \\
 \left. + V t_{i,j}^{dist}) + \sum_m^{m \in [1;M]} x_{j,m} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)rV}{100} (t_{j,m}^{wait} + \right. \right. \\
 \left. \left. + 0.76 \log \left(\frac{1100 \frac{C}{C_{max}} - 17T + 750}{17T + 339} \right) \right) + r_{j,m} (C_{max} - C_j) \right) \right) \\
 | \quad T < 20 \\
 | \quad i \neq j \\
 | \quad i \neq N \\
 | \quad j \neq 1 \\
 | \quad C_i \geq S_{i,j} \frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)}{100} \\
 | \\
 | \quad T \geq 20 \\
 | \quad i \neq j \\
 | \quad i \neq N \\
 | \quad j \neq 1 \\
 | \quad C_i \geq S_{i,j} \frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)}{100}
 \end{array}
 \right. , \text{ где}$$

$$\left\{
 \begin{array}{l}
 Z = \sum_i^{i \in [1;N]} \sum_j^{j \in [1;N]} x_{i,j} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)r}{100} (S_{i,j} + \right. \\
 \left. + V t_{i,j}^{dist}) + \sum_m^{m \in [1;M]} x_{j,m} \left(\frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)rV}{100} (t_{j,m}^{wait} + \right. \right. \\
 \left. \left. + 0.76 \log \left(100 \left(1 - \frac{C}{C_{max}} \right) \right) \right) + r_{j,m} (C_{max} - C_j) \right) \right) \\
 | \quad T \geq 20 \\
 | \quad i \neq j \\
 | \quad i \neq N \\
 | \quad j \neq 1 \\
 | \quad C_i \geq S_{i,j} \frac{5.6 \cdot 10^{-4} Q_{opt}(T^2 - 40T + 2200)}{100}
 \end{array}
 \right. \quad (2.8)$$

C_i – уровень зарядки аккумулятора электрокара перед процедурой зарядки на зарядной станции в вершине i , кВт·ч.

Следует определить наиболее рациональный метод составления комбинаций граней (частей маршрута) для формирования оптимального пути с учётом всех имеющихся условий. Если практический опыт индийских исследователей ограничивается местностью в один город [11], и грубая сила в виде банального перебора работала эффективно, то в текущих особенностях решаемой задачи масштаб принимает размеры федерального значения. По этой причине перебор всех возможных частей маршрута потребует большое количество вычислительных ресурсов, а также займёт большой объём времени. Работа китайских специалистов акцентирует внимание на использовании эволюционных алгоритмов [12], которые, однако, так же являются некоторой формой перебора, но с более направленной природой, которая устанавливается с помощью функции приспособленности. В связи с этим применение такого метода допустимо в рамках рассматриваемой проблемы, потому что он будет требовать гораздо меньшее количество ресурсов для своей реализации, нежели предыдущий подход.

Более того, данная транспортная задача может быть решена также с помощью специальных алгоритмов, которые полагаются не на перебор, а на выбор оптимальной из альтернатив частей пути в каждый из моментов времени следования по маршруту. Примером метода, обладающим описанными свойствами, является алгоритм Дейкстры, который обладает хорошей репутацией и долгой историей использования в теории графов, сфере управления цепями поставок, а также даже в решении задач маршрутизации для электротранспортных средств [32] [33]. Его принципом является поэтапное формирование такого пути, веса вершин на протяжении которого обладают наименьшим совокупным весом из всех возможных альтернатив в графе. Таким образом, адаптация эволюционных алгоритмов под условия заявленной проблемы будет основываться на введённой математической

модели (2.8), где функцией приспособленности будет служить входящая в неё целевая функция (2.7.2), в то время как адаптация алгоритма Дейкстры под рассматриваемую задачу будет выражаться в формировании комбинаций маршрутов согласно введённой математической модели (2.8), где весами вершин будут являться значения составляющей её целевой функции (2.7.2).

В связи с вышеизложенным хотелось бы заявить, что в рамках предстоящей программной разработки будут реализованы два метода построения маршрута: эволюционные алгоритмы и алгоритм Дейкстры, а также будет проведён их сравнительный анализ. Подход, показывающий лучшие показатели целевой функции и скорость работы, будет отобран как основной.

Стоит разобраться в вопросе в систематизации и хранении данных, необходимых для моделирования вышеописанных сценариев. В рамках решения поставленной задачи задействовано большое число реально существующих зарядных станций на территории Российской Федерации, каждая из которых обладает несколькими слотами со своими свойствами, участвующими в проведении процедуры фильтрации ЭЗС для отдельной модели электротранспортного средства. Следует выделить параметры слотов ЭЗС, которые стоит фиксировать:

1. идентификационный ключ (id);
2. координаты широты местоположения;
3. координаты долготы местоположения;
4. адрес местоположения;
5. название владеющего физического или юридического лица;
6. ценовой тариф;
7. тип коннектора;
8. мощность электрического тока;
9. вид электрического тока;
10. статус работоспособности.

Описанный выше перечень обладает наименьшим числом пунктов, необходимых для решения рассматриваемой задачи. Пункт №1 позволяет сформировать уникальный ключ слота зарядной станции, пункты №2 и №3 носят пространственный характер, пункты №4 и №5 - описательно идентификационный, в то время как остальные пункты – экономический и технический. Последняя выделенная группа параметров предоставляет возможность в фильтрации ЭЗС согласно их совместимости с техническими характеристиками электротранспортного средства и желаниями его владельца. Стоит отметить, что в связи с конечно определённым числом полей, которые будут использоваться для описания слотов зарядных станций, и возможности в однозначном определении типа данных, содержащихся в каждом из них, стоит обратить внимание на реляционную базу данных как инструмент добавления, хранения и редактирования информации по слотам ЭЗС. По этой причине все необходимые сведения о них можно представить в таблице с фиксированным числом столбцов, количество которых будет тождественно количеству перечисленных ранее полей.

Следует обратить внимание на процесс регистрации брони слота зарядной станции и фиксации факта её совершения. Прежде всего, обратимся к аналитической работе немецких экспертов, целью которой является выделение эффективных практик организации общественной зарядной инфраструктуры с помощью проведения личных опросов и систематизации имеющегося опыта [34]. Из статьи можно сделать вывод о том, что наиболее допустимым периодом времени, в течение которого пользователь может планировать бронирование слотов зарядных станций под свои нужды равен 2-ум неделям. Помимо этого, самый подходящий срок до которого можно отменить зарегистрированную резервацию равен 30-ти минутам до предполагаемого начала процедуры зарядки. Данные сведения были получены путём проведения опросов и анализа прошлого практического опыта. Для удобства наименьший период, доступный для бронирования слота зарядной станции, можно принять равным 15 минутам с добавлением 5 дополнительных

минут для проведения мероприятий, имеющих место быть после завершения процедуры зарядки [13].

В связи с вышеизложенным возникает возможность предложить метод для систематизации и хранения данных, необходимых для моделирования расписания бронирования. По причине того, что доступных окон для брони будет 1008 штук: [текущая дата, 00:00-00:20; текущая дата + 13 дней, 23:30-00:00], а число идентификационных ключей ЭЗС, участвующих в процессе бронирования, так же будет отличаться своим большим размером, то представление данных в виде реляционной базы данных не является рациональным. Тем не менее, стоит обратить внимание на нереляционные методы хранения информации по типу JSON-документа, который более легко редактируем, чем таблица с огромным количеством строк и столбцов. Таким образом, можно определить структуру JSON-документа, являющейся базой данных для расписания броней (*Рисунок 15*):

1. идентификационный ключ (id):
 - └ 2. логины пользователей, совершивших резервацию;
 - └ 3. временные окна брони.

Рисунок 15. Структура JSON-документа для документирования резерваций ЭЗС

Несложно догадаться, что с течением времени записи со старыми бронями можно настроить на автоматическое удаление, чтобы уменьшить размер базы данных, а возможность выбора доступных альтернатив для резервации ограничить во время программной реализации решения. Динамическая структура JSON-документа позволит без препятствий часто обращаться к нему и сужать или расширять его структуру.

Следует обратить внимание на ситуации, когда сформированный оптимальный маршрут не подходит для пользователя, так как зарядные станции на его протяжении не обладают свободными подходящими для него временными окнами. В таком случае необходимо предложить

субоптимальный маршрут. Для достижения этой цели нужно разработать механизм для фиксации идентификационных номеров ЭЗС, используемых в неподходящих маршрутах, чтобы поочерёдно исключать их из сети зарядных станций с целью построения другого альтернативного маршрута. Таким образом, для построения альтернативного субоптимального маршрута подойдёт метод, заключающийся в поочерёдном исключении зарядных станций из сети по возрастанию числа доступных у них временных окон для брони с целью обеспечения сохранения большого выбора периодов для резервации.

В данном пункте методологической части с опорой на теоретические основы и практический опыт нам удалось сформировать собственный метод для решения поставленной задачи. Все его аспекты должны быть подвергнуты программной реализации в следующей части этой работы.

Выводы по главе 2

Рассматриваемая проблема обладает конкретными условиями, которые необходимо учитывать, чтобы найти к ней оптимальное решение. Они могут быть разделены на две группы: параметры эксплуатации ЭЗС, а также внутренние и внешние факторы влияния на эксплуатацию электротранспортных средств. Параметры зарядных станций, главным образом, определяют совместимость конкретной ЭЗС с техническими характеристиками используемого электрокара. Внутренние и внешние характеристики эксплуатации электротранспортного средства характеризуют поведение величин его технических характеристик в текущих условиях выполнения задачи. Все эти параметры позволяют сформировать переменные для функционирования введённой математической модели, которая лежит в основе потенциальной программной реализации.

Знания из теоретической и методологической частей данной работы

предоставляют возможность в формировании собственного метода для поиска оптимальных маршрутов для электрокаров с захватом ЭЗС и предоставлением временных окон для резервации зарядных станций на их протяжении. Основные аспекты предлагаемого подхода могут быть выражены в следующей таблице (*Таблица 2*):

Таблица 2. Реализация основных аспектов предлагаемого метода

№	Аспект метода	Реализация
1	Интерфейс пользователя	WEB-страница
2	Целевая функция	(2.7.2)
3	Математическая модель	(2.8)
4	Построение маршрута	Алгоритм Дейкстры или эволюционные алгоритмы
5	Добавление, хранение и редактирование данных о ЭЗС	Реляционная база данных
6	Добавление, хранение и редактирование расписания броней ЭЗС	Нереляционная база данных
7	Построение субоптимальных маршрутов по требованию	Поочерёдное исключение ЭЗС из сети в порядке возрастания числа доступных альтернатив для брони

ГЛАВА 3. ПРАКТИЧЕСКАЯ ЧАСТЬ

3.1. Моделирование основного бизнес-процесса

После формализации предлагаемого метода решения поставленной задачи стоит обратиться к нотациям моделирования бизнес-процессов с целью представления вводимого подхода посредством его графической интерпретации. В рамках построения его модели будет использоваться нотация «EPC» (*Рисунок 16*) [35] [36]. Причина в таком выборе лежит в возможности исчерпывающего представления предлагаемого метода только с помощью бизнес-процессов верхнего уровня и его линейности, отражающей чёткую последовательность действий.



Рисунок 16. Базовая структура EPC-диаграммы [36]

В связи с этим взаимосвязанные бизнес-процессы в нотации «EPC» вводимого метода решения задачи построения оптимальных маршрутов для электротранспортных средств с включением в них зарядных станций, временные окна использования которых бронируются пользователями под свои нужды, выглядят следующим образом (*Рисунок 17, Рисунок 18, Рисунок 19*):

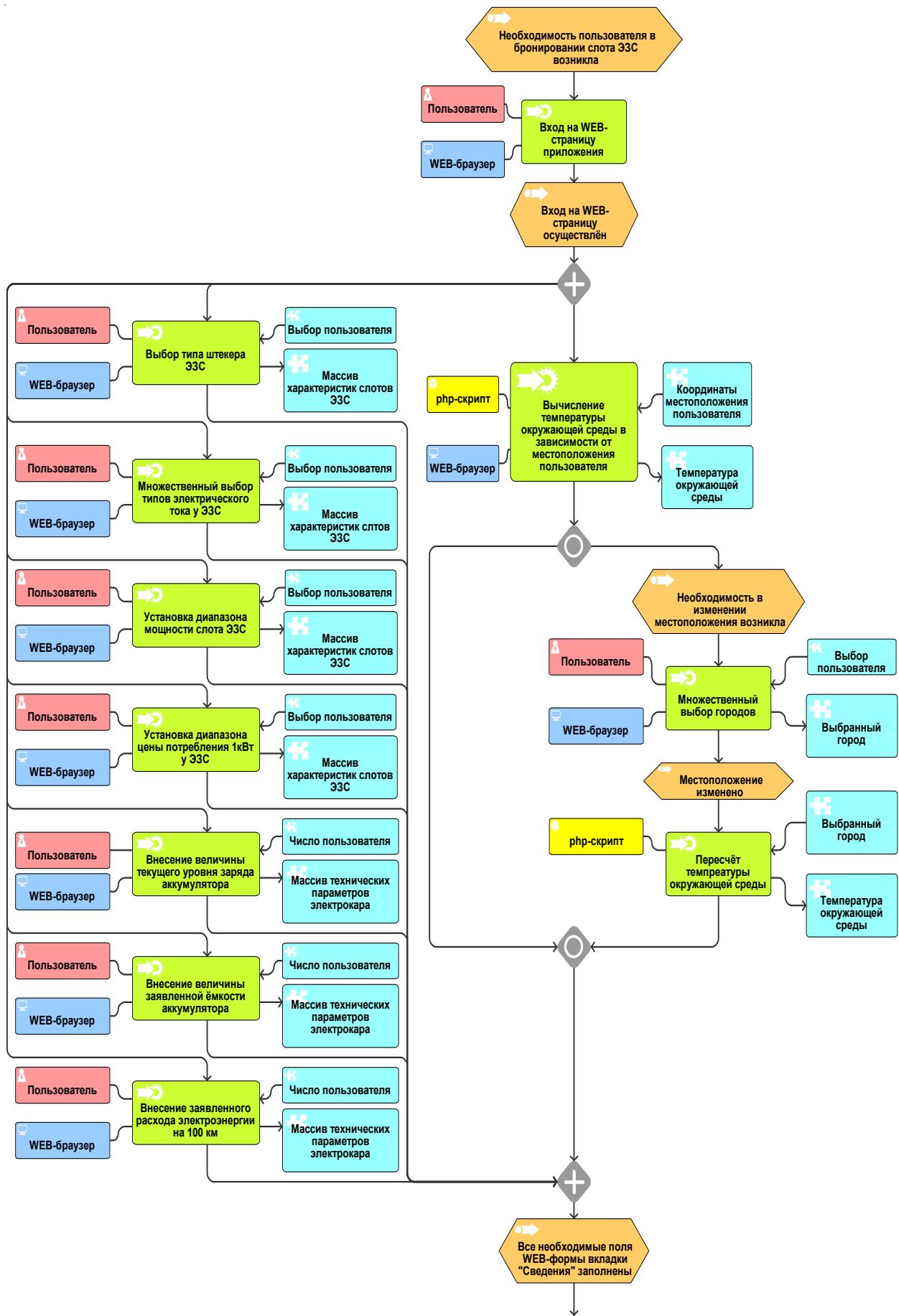


Рисунок 17. Модель бизнес-процессов предлагаемого решения (1)

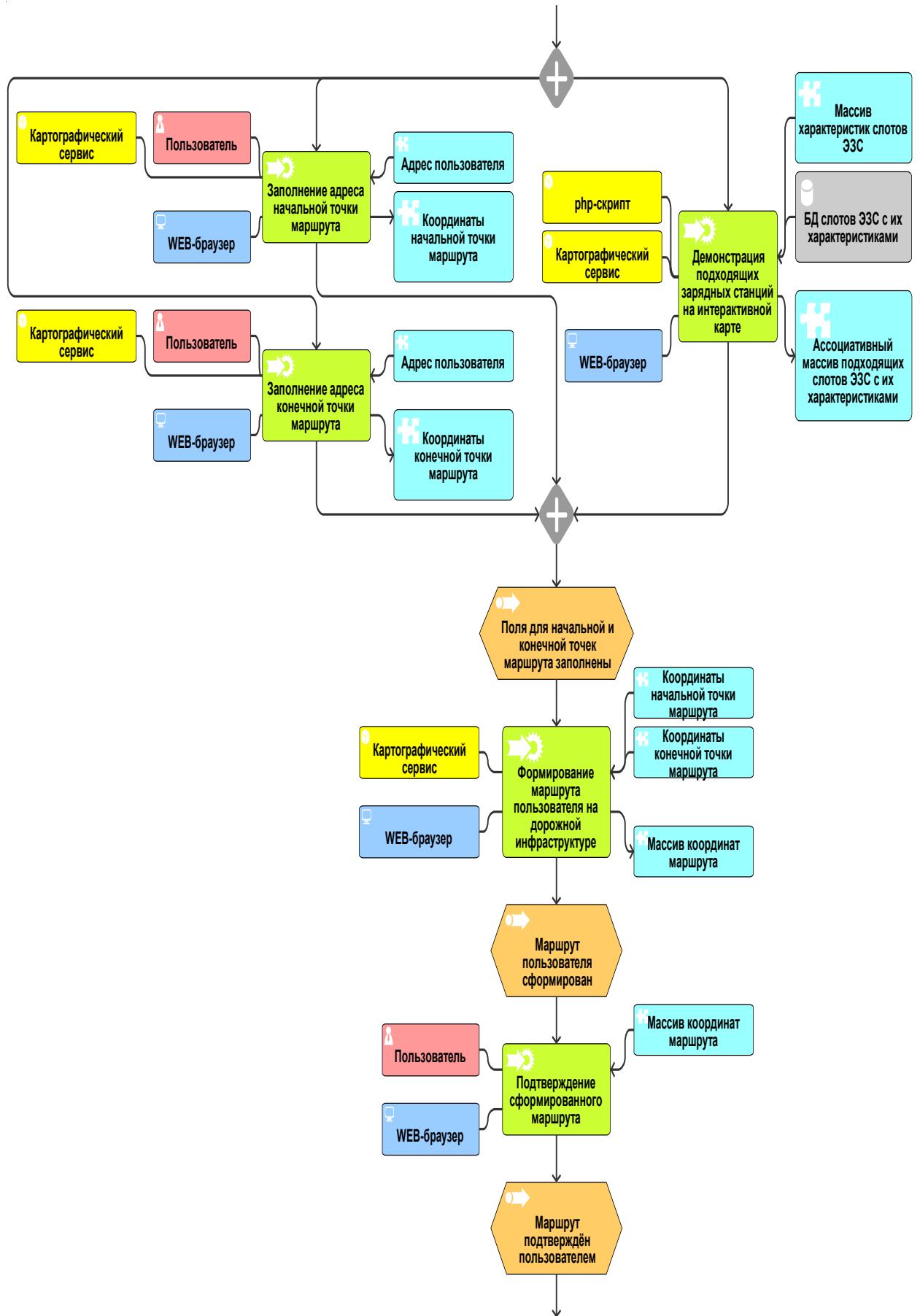


Рисунок 18. Модель бизнес-процессов предлагаемого решения (2)

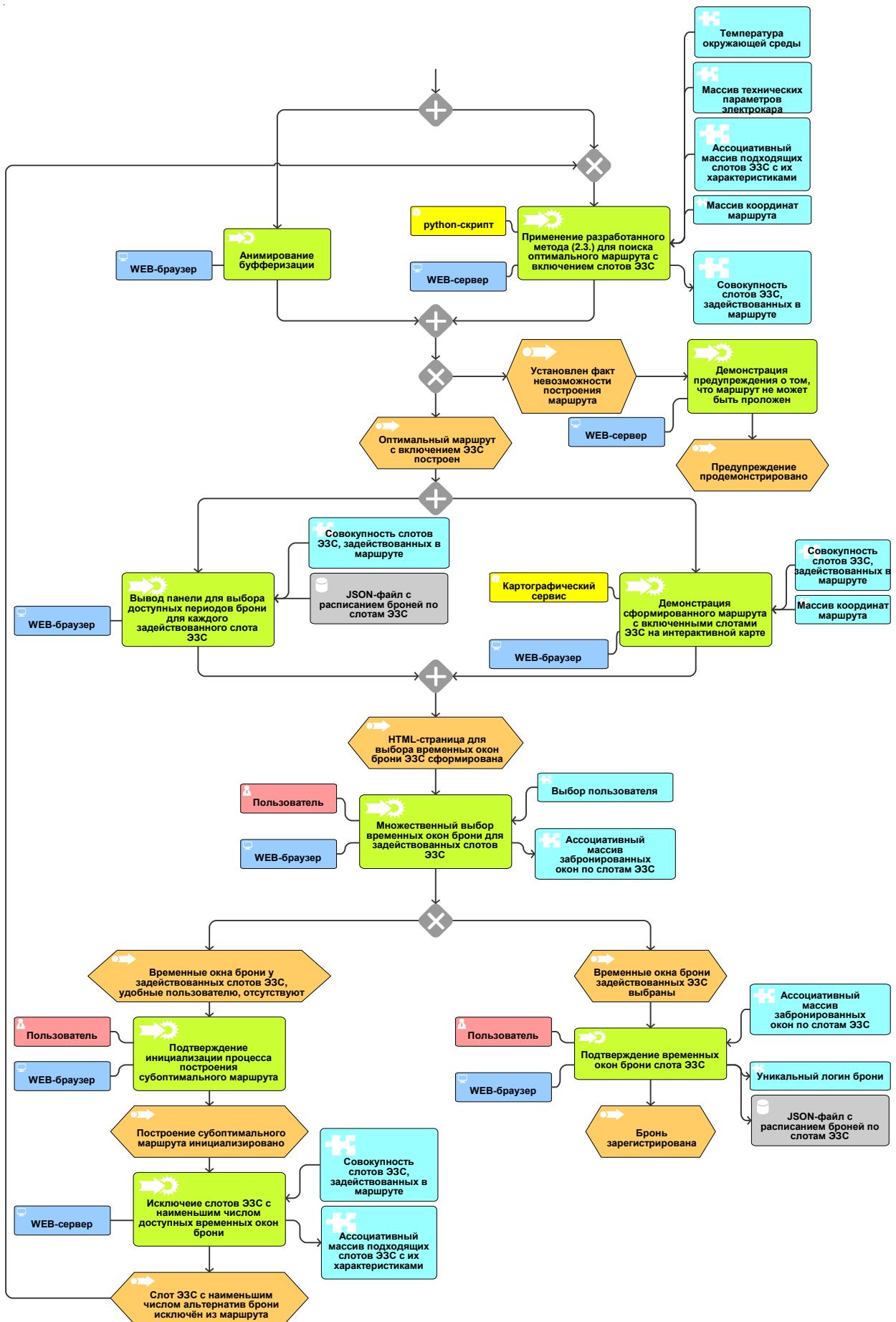


Рисунок 19. Модель бизнес-процессов предлагаемого решения (3)

Таким образом, прогрессию бизнес-процессов, составляющих механику вводимого метода, можно разделить на три этапа:

1) Заполнение вкладки «Сведения» (*Рисунок 17*);

На данном этапе у пользователя возникает желание в бронировании временного окна эксплуатации зарядной станции, и онходит на WEB-страницу приложения. Там он заполняет все необходимые поля, связанные с характеристиками ЭЗС и техническими параметрами используемого электрокара. Во время его действий специально разработанная PHP-функция определяет координаты местоположения пользователя и отправляет curl-запрос на онлайн-сервис «Яндекс.Погода», ответом которого является целое число, выражающее температуру окружающей среды на определённой местности и входящее в соответствующее поле на вкладке WEB-страницы. Помимо этого, пользователь может изменить автоматически определённый город, в котором он находится, через выпадающий список с городами, если он желает зарегистрировать бронь для электротранспортного средства, эксплуатирующегося в другой части страны. В таком случае ранее упомянутая PHP-функция снова отправит запрос, но уже с изменёнными параметрами.

2) Заполнение вкладки «Маршрутизация и бронирование» (*Рисунок 18*);

После перехода на следующую вкладку потребителю услуги становится доступна интерактивная карта от картографического сервиса «Яндекс.Карты», на которой из базы данных MySQL посредством PHP-функции загружаются метки с подходящими зарядными станциями согласно ранее заданным фильтрам на первом этапе, при нажатии на которых появляются всплывающие подсказки, содержащие их характеристики. На данной вкладке пользователь вносит в поля начальный и конечный адреса маршрута путём их написания или выбора их точек на карте. После автоматической визуализации маршрута на карте он нажимает кнопку его подтверждения, и содержимое всех полей, заполненных на двух вкладках, с ассоциативным массивом ранее

отфильтрованных слотов ЭЗС с их характеристиками записываются в JSON-документ, который принимается бэкэнд-алгоритмом, написанном на языке программирования Python.

3) Регистрация брони (*Рисунок 19*).

На данном этапе если нахождение решения возможно, то формируется новая вкладка: на интерактивной карте строится маршрут с использованием ранее установленных начальной и конечной точек планируемой поездки пользователя, а также массива задействованных в оптимальном маршруте id слотов ЭЗС, определённым бэкэнд-алгоритмом, а если нет, то выводится всплывающее окно с сообщением об этом факте. На панели управления бронями демонстрируется список id слотов ЭЗС, участвующих в сформированном маршруте, с прилагающимися доступными временными окнами для брони к каждому из них. При отсутствии удобного окна для брони у конкретного слота ЭЗС пользователь может запустить процесс поиска субоптимального маршрута, что повторит работу бэкэнд-алгоритма с исключением одного из элементов из ассоциативного массива подходящих слотов ЭЗС с их характеристиками по ранее изложенному принципу. Если регистрация брони может быть воспроизведена сразу, то выбор потребителя услуги подтверждается и он получает уникальный логин для идентификации брони, а его выбор фиксируется в базе данных расписания.

Таким образом, в данном пункте практической части удалось построить модель бизнес-процессов предлагаемого решения задачи в нотации «EPC» и предоставить описательные комментарии к ней. Результат проведённого моделирования будет являться фундаментом, на который будут опираться следующие разделы этой работы.

3.2. Проектирование архитектуры интеллектуального сервиса

Прогрессия бизнес-процессов во времени является необходимой, но недостаточной деталью полного понимания механики функционирования

предполагаемого веб-приложения. В связи с этим стоит уделить внимание взаимодействию отдельных сущностей между собой в контуре единой системы, представляющей из себя разрабатываемое решение. Взаимосвязь агентов интеллектуального сервиса может быть продемонстрирована с помощью проектирования его архитектуры. Таким образом, ИТ-архитектура планируемого веб-приложения принимает следующий вид (**Рисунок 20**):

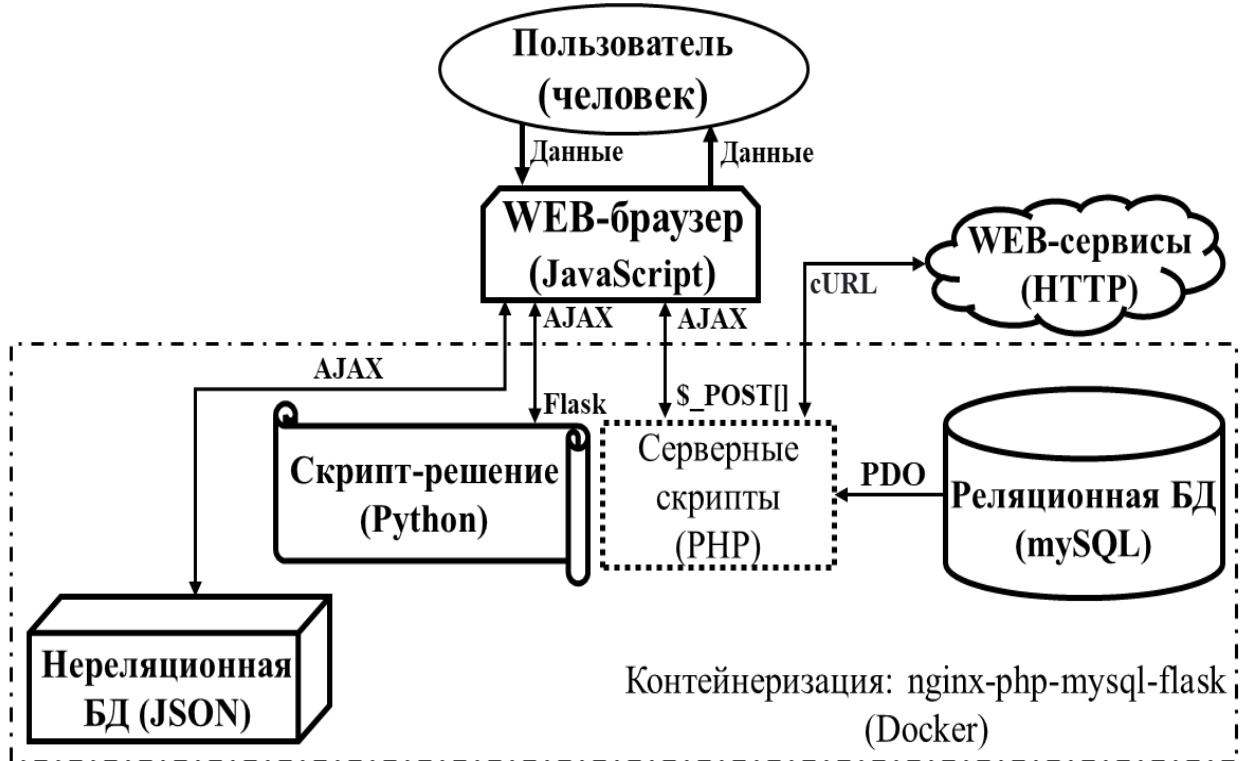


Рисунок 20. Архитектура интеллектуального сервиса

Из вышеприведённой схемы можно выделить два типа объектов: основной и вспомогательный. Основные объекты непосредственно участвуют в формировании решения к поставленной задачи, формируя необходимые переменные и проводя математические операции с ними с целью достижения поставленной цели, в то время как вспомогательные – обеспечивают нужные условия для их функционирования между собой, не формируя сущностей, используемых в процессе нахождения оптимального решения. Согласно этому определению, основные объекты составляют:

- 1) WEB-браузер;
- 2) WEB-сервер;
- 3) скрипт-решение;

- 4) реляционная БД;
- 5) нереляционная БД;
- 6) WEB-сервисы (внешние источники).

Основным элементам фронт-энда и бэк-энда и их взаимодействию между собой уделялось уже много внимания в теоретической части этой работы (*Рисунок 3*), поэтому стоит сфокусироваться на вспомогательных агентах, которые включают в себя:

- 1) Серверные скрипты;

Данные программы позволяют основным объектам фронт-энда и бэк-энда обмениваться между собой данными через веб-сервер. В рассматриваемом случае они написаны на языке программирования PHP, так как он поддерживается веб-сервером «nginx» [37]. Главная роль таких скриптов в данной системе – это обеспечение формирования JSON-файлов и их последующая дистрибуция между основными объектами архитектуры интеллектуального сервиса, которые, в свою очередь, с помощью использования собственных протоколов взаимодействуют с этими файлами нужным для достижения своих локальных целей образом.

- 2) Контейнеризация.

Данный метод представляет из себя построение единого контура, получаемого путём развёртывания программного обеспечения, объединяющего код приложения со всеми файлами, библиотеками и базами данных, необходимыми для возможности его запуска в любой инфраструктуре. Основными преимуществами такого подхода к эксплуатации приложений являются портативность, позволяющая использовать программы на любых операционных системах и при её любых спецификах, а также отказоустойчивость, объясняющаяся изолированностью функционирования отдельных контейнеров с приложениями друг от друга. Таким образом, в рассматриваемом случае используется платформа «Docker» [38], предоставляющая возможность в формировании контейнеров, вмещающих в себя веб-сервер «nginx», SQL-

сервер, язык программирования PHP и веб-фреймворк «flask», работа которых предоставляет возможность на любой инфраструктуре без установки для этого отдельных программ и библиотек на жёсткий диск компьютера.

Помимо основных и вспомогательных агентов разрабатываемого интеллектуального сервиса, стоит обратить внимание на протоколы и функции, использующиеся ими для обмена информацией:

- AJAX;

Аббревиатура данного протокола расшифровывается как «Asynchronous JavaScript and XML». Этот инструмент позволяет веб-браузеру и веб-серверу в фоновом режиме, то есть без перезагрузки HTML-страницы, взаимодействовать друг с другом. В данном случае, с помощью AJAX веб-браузер принимает JSON-файлы, возвращаемые PHP-функциями после коммуникации с другими элементами системы. После получения JSON-файла по AJAX данные, содержащиеся в нём, обрабатываются и впоследствии разносятся по элементам интерфейса с помощью языка программирования JavaScript.

- cURL;

Название данного метода расшифровывается как «Client URL», и он представляет из себя программное обеспечение, которое предоставляет библиотеку API к «libcurl» и инструмент командной строки «curl». Главным образом, этот инструмент используется с целью передачи данных с сервера на сервер при помощи различных протоколов. В рассматриваемой системе данный метод применяется для получения данных из внешних веб-сервисов, таких как «Яндекс.Карты» и «Яндекс.Погода» в рамках эксплуатации функциональных возможностей сторонних инструментов, необходимых для решения поставленной задачи. Запросы типа «curl» к внешним сервисам формируются в PHP-функциях, которые впоследствии обмениваются переданными данными с веб-браузером посредством ранее упомянутому AJAX.

- PDO;

Этот инструмент представляет из себя внедрённый интерфейс для доступа к базам данных SQL в PHP. В рассматриваемой системе с помощью него PHP-функция получает данные из таблицы, находящейся в базе данных MySQL в формате JSON-файла, который впоследствии передаётся веб-браузеру посредством AJAX для дополнения элементов фронт-энда.

- flask;

Эта библиотека представляет собой небольшой и лёгкий веб-фреймворк, предлагающий полезные инструменты и функции для облегчения процесса создания веб-приложений с использованием Python. Он обеспечивает гибкость и является более доступным фреймворком для новых разработчиков, так как позволяет создать веб-приложение быстро, используя только один файл Python.

- \$_POST[].

Стандартная функция языка программирования PHP для чтения ассоциативных массивов данных, переданных в функцию посредством HTTP-метода.

Таким образом, в этом пункте нам удалось построить архитектуру интеллектуального сервиса, изучить роли его основных и вспомогательных элементов, а также познакомиться с инструментами, посредством которых они обмениваются информацией. Полученная модель обеспечивает хорошее понимание механики функционирования всего веб-приложения, упрощая процесс его разработки.

3.3. Создание пользовательского интерфейса сервиса

В рамках процесса создания фронтэнд-части веб-приложения используется язык программирования JavaScript с его фреймворком «Jquery», позволяющим значительно уменьшать объём написанного кода за счёт упрощения синтаксиса стандартных методов, использующегося в классической версии языка. Помимо этого, применяются языки разметки HTML

и метаязык SCSS, упрощающий взаимодействие с языком декорирования внешнего вида документа CSS. Главную роль, которую в данном случае играет пользовательский интерфейс – это предоставление пользователю возможности во взаимодействии с бэкэнд-алгоритмом наиболее понятным и комфортным образом. Все составляющие части фронт-энда отличаются минимализмом и спроектированы в унифицированном стиле, чтобы максимально способствовать увеличению степени восприятия информации клиентом.

Рассматривать элементы пользовательского интерфейса лучше всего в порядке прогрессии бизнес-процессов, протекающих в веб-приложении, согласно пункту №3.2 этой работы. Таким образом, первая вкладка под названием «Сведения» выглядит следующим образом (*Рисунок 21*):

Рисунок 21. Интерфейс вкладки "Сведения"

Город пребывания пользователя выбирается автоматически посредством запроса его геолокации. Тем не менее, при возникновении желания у клиента в его изменении он может выбрать другой посредством выпадающего списка в шапке вкладки, после этого температура окружающей среды и фокус на интерактивной карте в следующей вкладке изменятся согласно его выбору.

Более того, пользовательский интерфейс алгоритмически построен таким образом, что все поля для ввода на вкладке «Сведения» в любом случае не будут оставленными пустыми – это необходимо для того, чтобы предотвратить передачу бэкэнд-алгоритму пустых переменных.

Выпадающий список для изменения целевого города для моделирования маршрута и загрузки данных о погодных условиях выглядит следующим образом (*Рисунок 22*):

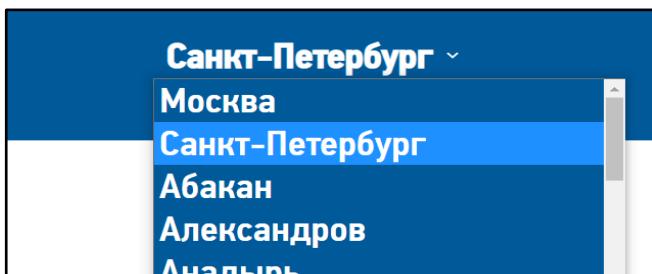


Рисунок 22. Селектор городов

Адаптивный пользовательский интерфейс для устройств с маленькой диагональю экрана или мобильных гаджетов выглядит следующим образом (*Рисунок 23*):

A screenshot of the ChargeBook adaptive interface for the "Сведения" tab. The interface includes the following sections:

- Характеристики электрозарядной станции:**
 - Тип штекера:** Options include T2 and CHAdeMO, with T2 selected.
 - Тип электрического тока:** Options include AC (переменный ток) and DC (постоянный ток), with DC selected.
 - Диапазон мощности (кВт):** A slider ranging from 10 to 50, currently set at 50.
 - Диапазон цены за потребление 1 кВт (руб.):** A slider ranging from 0 to 60, currently set at 60.
- Технические параметры электрокара:**
 - Текущий уровень заряда аккумулятора:** 50 кВт·ч
 - Заявленная ёмкость аккумулятора:** 75 кВт·ч

Рисунок 23. Адаптивный интерфейс вкладки "Сведения"

После этого пользователь переходит на вкладку «Маршрутизация и бронирование». Данные, внесённые клиентом в части блока, посвящённого желаемым характеристикам зарядных станций, используются в алгоритмах, заложенных в логике работы этой вкладки. Её интерфейс также обладает интерактивными элементами, цветной подсветкой для кнопок и выглядит следующим образом (*Рисунок 24*):

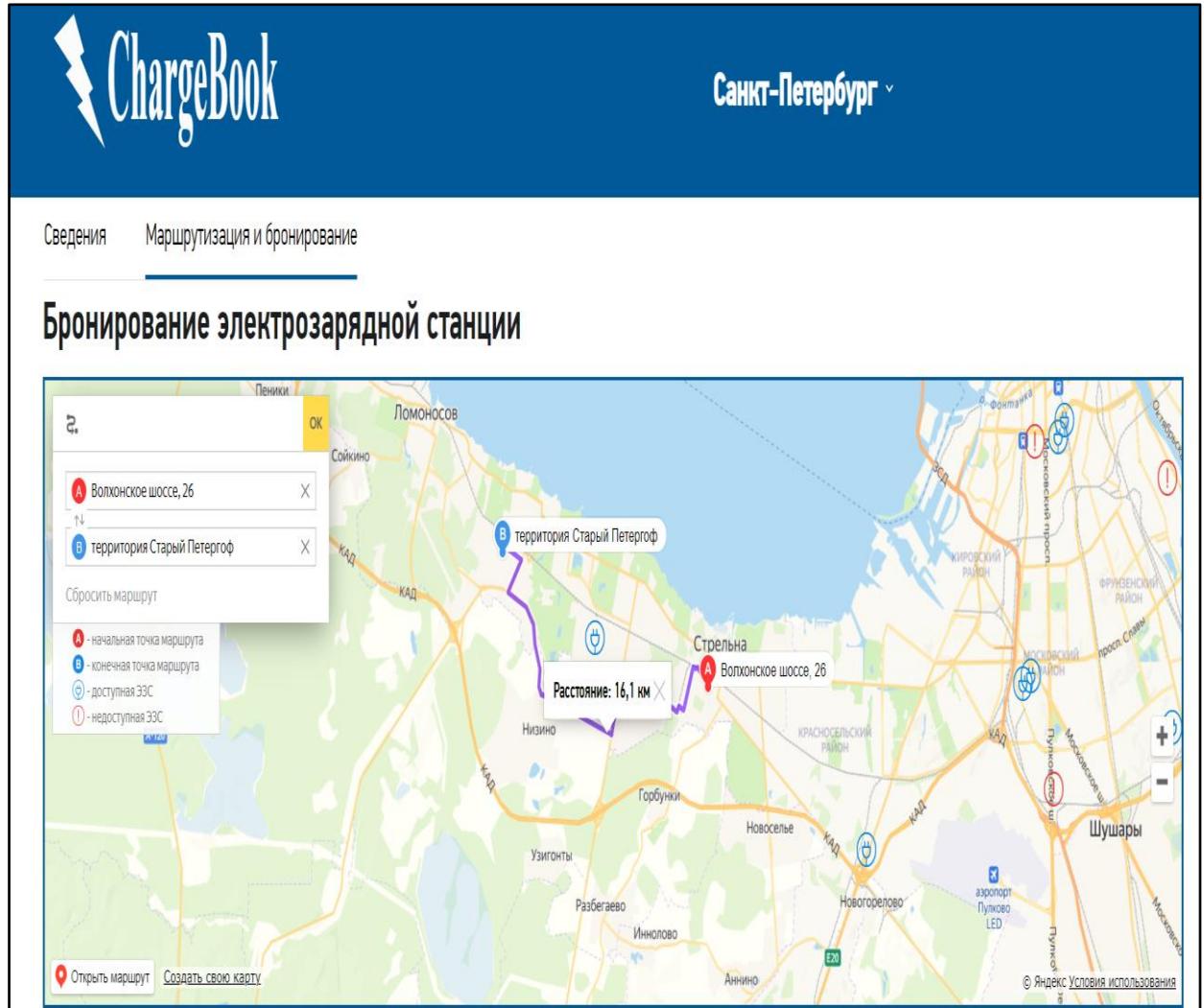


Рисунок 24. Интерфейс вкладки "Маршрутизация и бронирование"

В данном случае пользователю предоставляется интерактивная карта от веб-сервиса «Яндекс.Карты», где он строит свой маршрут путём занесения адресов в поля или же отметки их точек на карте. После составления маршрута автоматически появляется всплывающая подсказка с длинной пути между начальной и конечной точками составленного маршрута. Помимо этого, согласно заполненным данным из прошлой вкладки на карте из базы данных

генерируются подходящие слоты ЭЗС с их характеристиками во всплывающих подсказках, появляющихся после клика на их иконки (*Рисунок 25*):

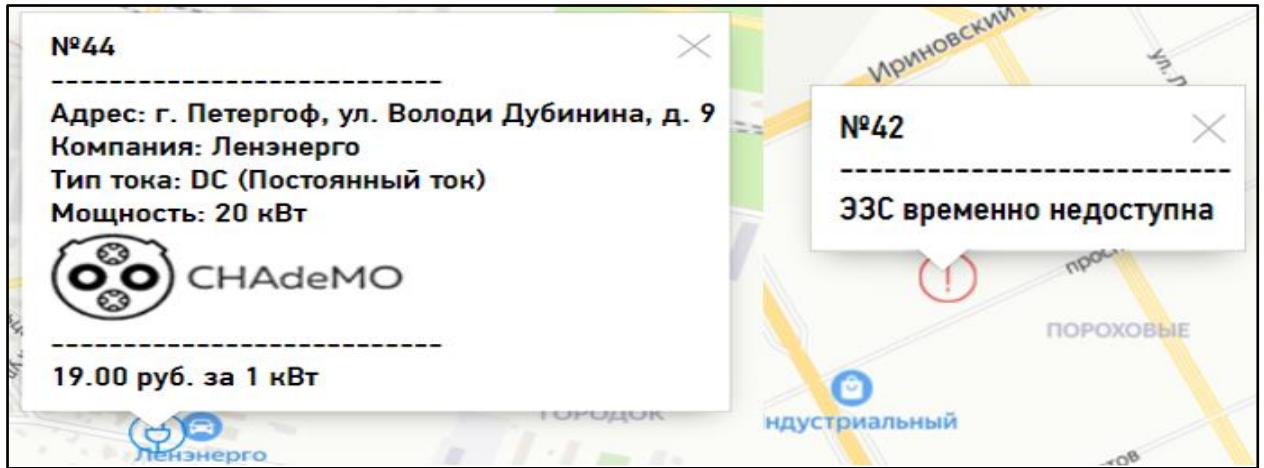


Рисунок 25. Дизайн всплывающих подсказок меток

Помимо этого, в рассматриваемой вкладке так же предусмотрена защита, не позволяющая отправить формы с массивом координат маршрута, если все или один из его пунктов остались незаполненными (*Рисунок 26*):

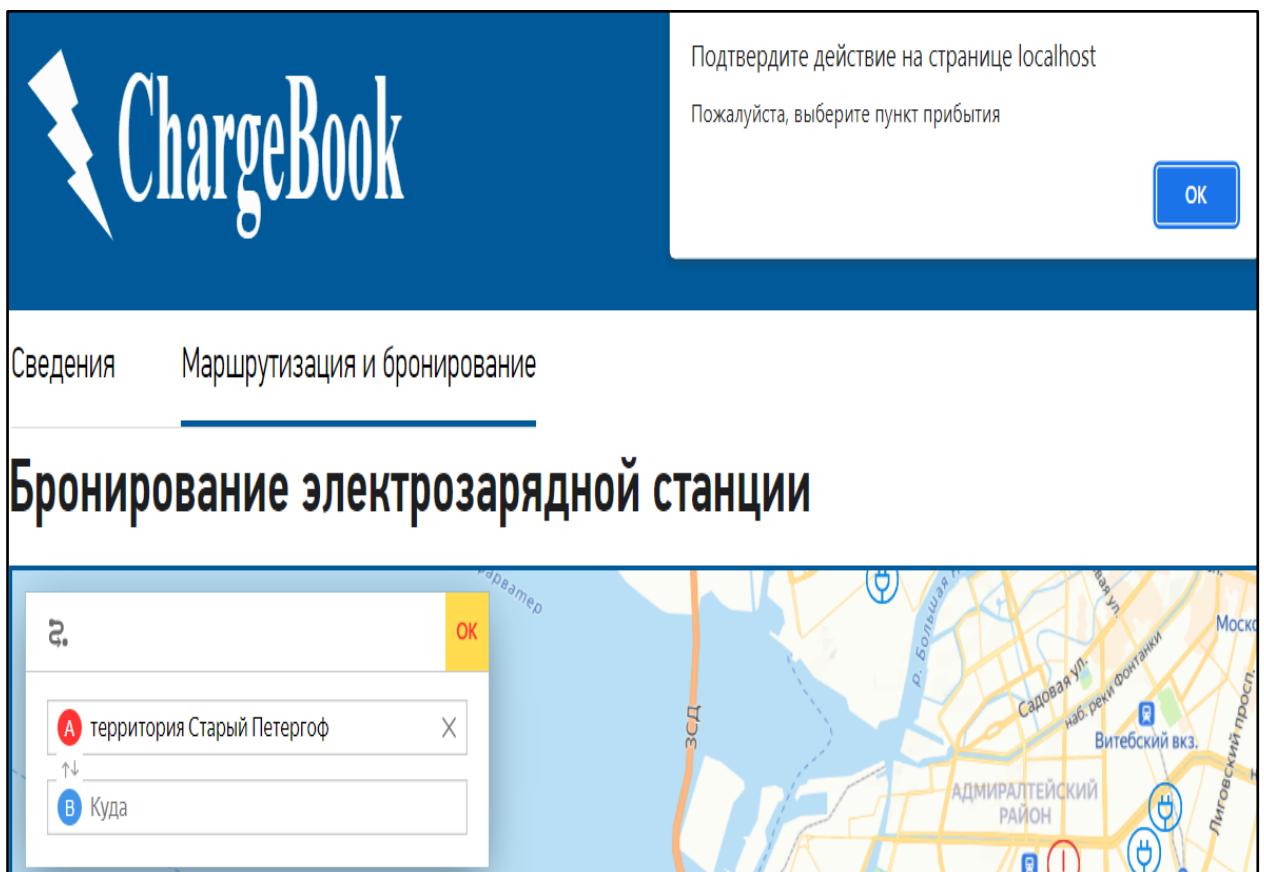


Рисунок 26. Дизайн предупреждения об ошибке

Адаптивный пользовательский интерфейс для устройств с маленькой диагональю экрана или мобильных гаджетов выглядит следующим образом (*Рисунок 27*):

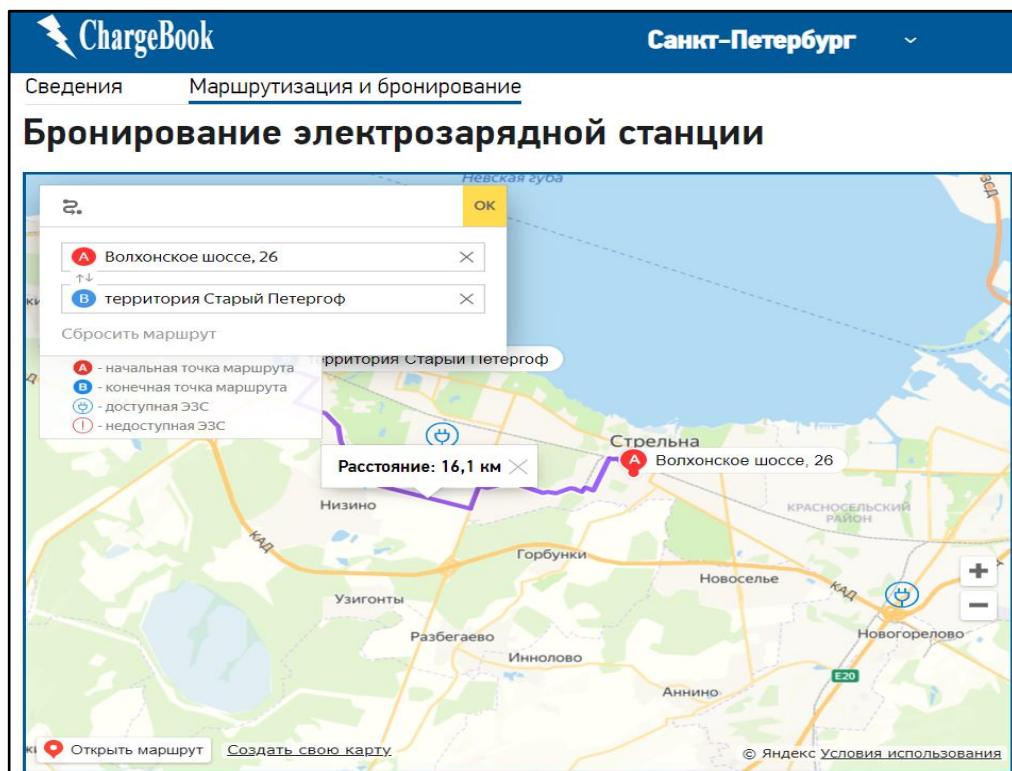


Рисунок 27. Адаптивный интерфейс вкладки "Маршрутизация и бронирование"

После отправки формы с содержимым заполненных полей из двух вкладок бэкэнд-алгоритм принимает её и начинает обработку полученных переменных. На протяжении всей работы скрипта-решения пользователь может увидеть анимацию прелоудера, которая свидетельствует о том, что задача обрабатывается. Она выглядит следующим образом (*Рисунок 28*):

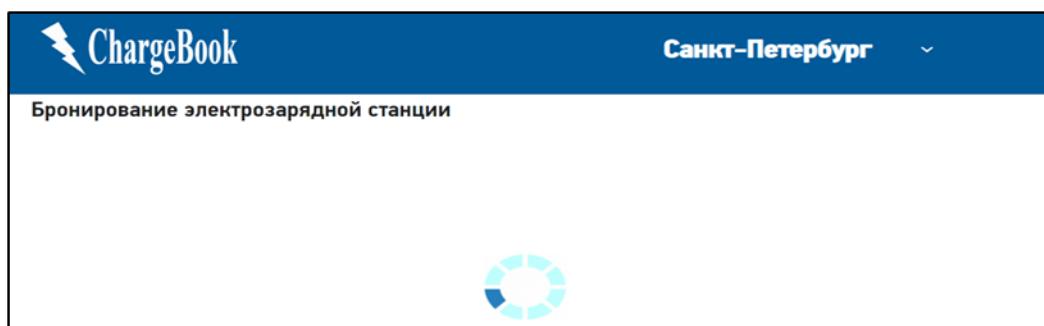


Рисунок 28. Дизайн прелоудера

Если выполнение поставленной задачи невозможно, то пользователь получит всплывающее окно с сообщением об этом факте, после закрытия

которого HTML-страница полностью обновится. Выглядит оно следующим образом (*Рисунок 29*):

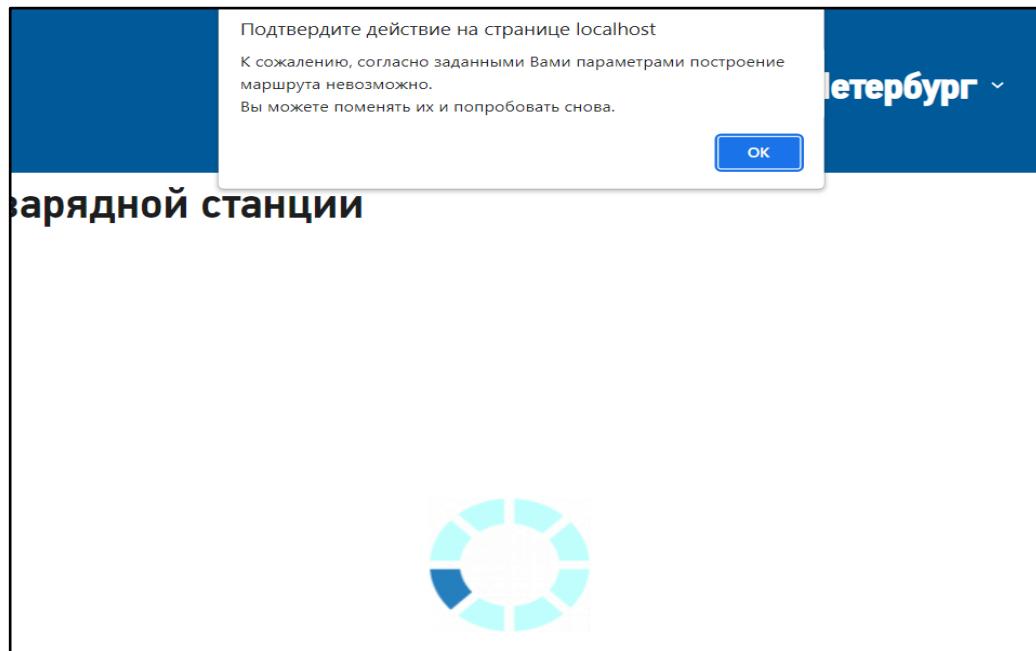


Рисунок 29. Предупреждение об невозможности выполнения задачи

Если для выполнения маршрута не требуется включение в него ЭЗС, то строится такой маршрут с последующей всплывающей подсказкой, повествующем о данном факте. Выглядит она таким образом (*Рисунок 30*):

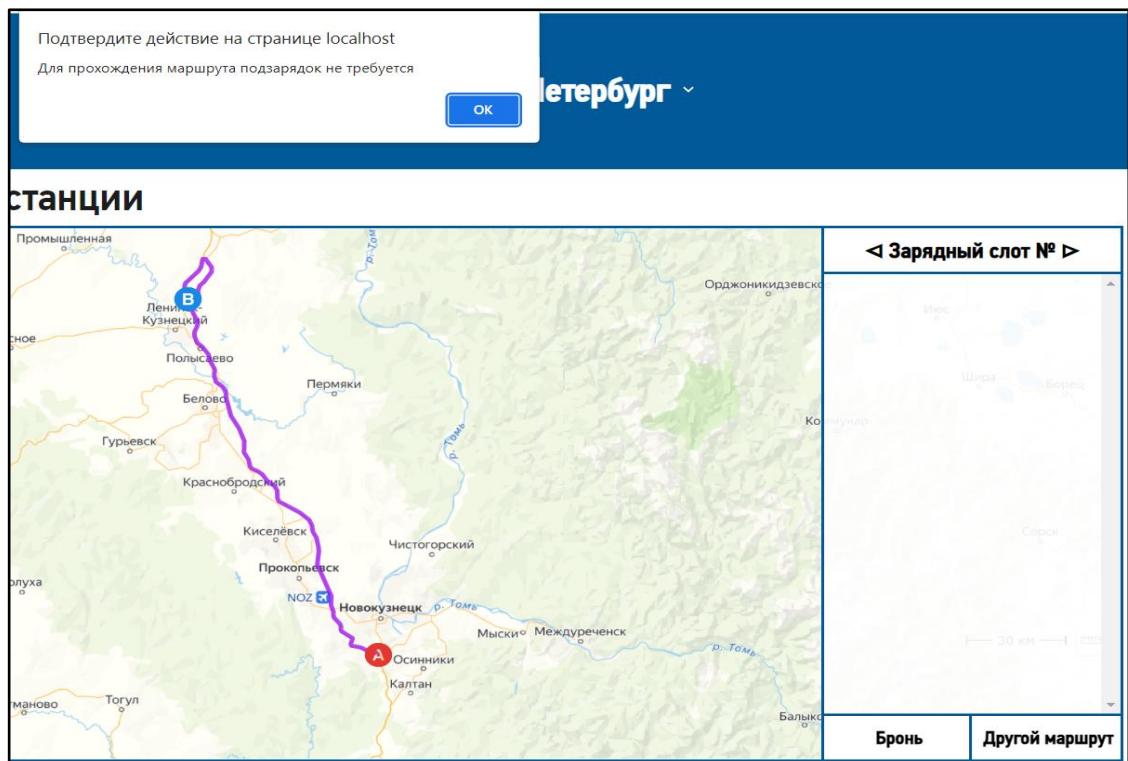


Рисунок 30. Сообщение об отсутствии необходимости включения в маршрут ЭЗС

Последняя вкладка также обладает интерактивными элементами и подсвечивающейся кнопками. Она принимает результаты работы скрипта-решения и визуализирует построенный оптимальный маршрут, предоставляя пользователю возможность в его подробном изучении. Она выглядит следующим образом (

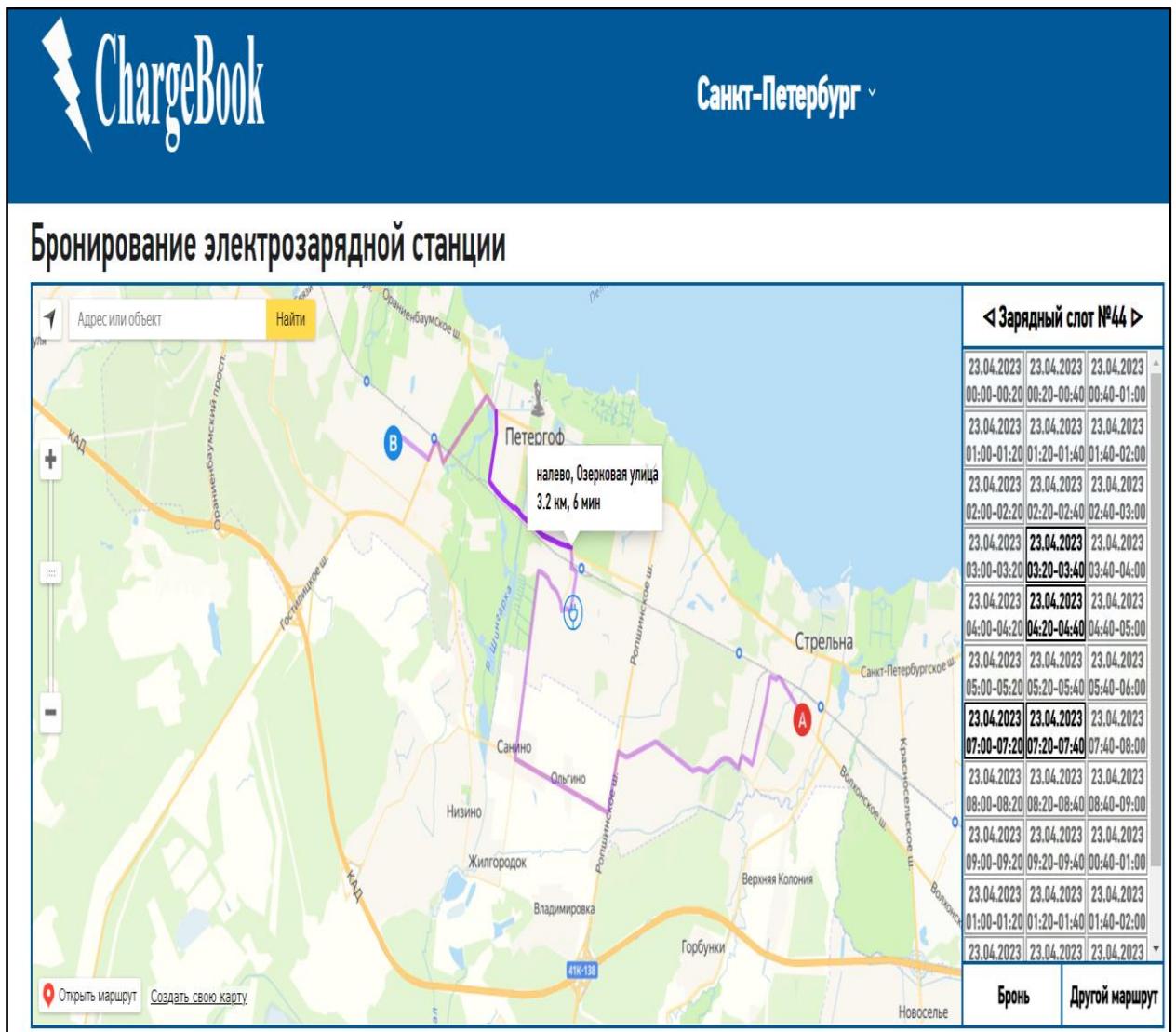


Рисунок 31):

Бронирование электрозарядной станции



Рисунок 31. Интерфейс вкладки бронирования

На данной вкладке можно получать всплывающие подсказки о частях построенного маршрута, взаимодействовать с метками на карте образом, аналогичным тому, который был реализован в предыдущей вкладке. Помимо этого, можно выбирать id задействованных слотов ЭЗС в оптимальном маршруте и получать для каждого из них список доступных для брони временных окон. Более того, здесь так же реализована проверка на ошибку, препятствующая пользователю в отправлении им пустой формы (

Подтвердите действие на странице localhost

Выберите хотя бы одно окно брони

OK

Санкт-Петербург

Станции

< Зарядный слот №44 >		
23.04.2023 00:00-00:20	23.04.2023 00:20-00:40	23.04.2023 00:40-01:00
23.04.2023 01:00-01:20	23.04.2023 01:20-01:40	23.04.2023 01:40-02:00
23.04.2023 02:00-02:20	23.04.2023 02:20-02:40	23.04.2023 02:40-03:00
23.04.2023 03:00-03:20	23.04.2023 03:20-03:40	23.04.2023 03:40-04:00
23.04.2023 04:00-04:20	23.04.2023 04:20-04:40	23.04.2023 04:40-05:00
23.04.2023 05:00-05:20	23.04.2023 05:20-05:40	23.04.2023 05:40-06:00
23.04.2023 07:00-07:20	23.04.2023 07:20-07:40	23.04.2023 07:40-08:00
23.04.2023 08:00-08:20	23.04.2023 08:20-08:40	23.04.2023 08:40-09:00
23.04.2023 09:00-09:20	23.04.2023 09:20-09:40	23.04.2023 00:40-01:00
23.04.2023 01:00-01:20	23.04.2023 01:20-01:40	23.04.2023 01:40-02:00
23.04.2023	23.04.2023	23.04.2023

Бронь **Другой маршрут**

Рисунок 32):

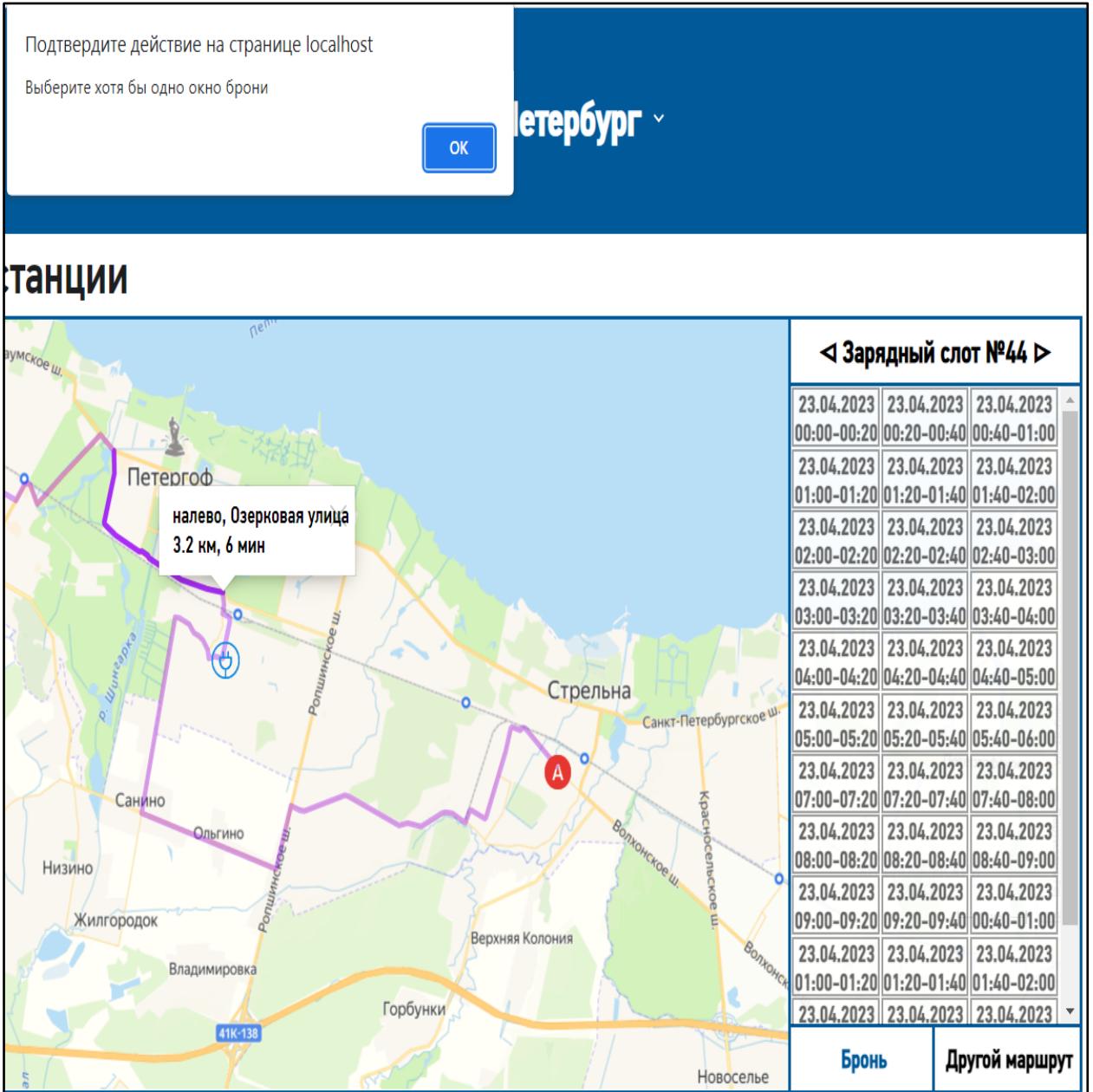


Рисунок 32. Дизайн предупреждения об ошибке

При нажатии на кнопку «Другой маршрут» выполняется алгоритм исключения из массива id подходящих слотов ЭЗС одного из пунктов построенного маршрута по ранее описанному методу и все действия, начиная с анимации прелоудера, повторятся снова. При успешном выборе временных окон брони и нажатии кнопки «Бронь» выйдет всплывающее окно со сгенерированным логином для подтверждения брони (

Подтвердите действие на странице localhost

Ваш уникальный логин брони: eRkT1543

OK

Петербург

танции

№ 44

Адрес: г. Петергоф, ул. Володи Дубинина, д. 9
Компания: Ленэнерго
Тип тока: AC (Переменный ток)
Мощность: 22 кВт

T2

19.00 руб. за 1 кВт

▷ Зарядный слот №44 ▷		
23.04.2023 00:00-00:20	23.04.2023 00:20-00:40	23.04.2023 00:40-01:00
23.04.2023 01:00-01:20	23.04.2023 01:20-01:40	23.04.2023 01:40-02:00
23.04.2023 02:00-02:20	23.04.2023 02:20-02:40	23.04.2023 02:40-03:00
23.04.2023 03:00-03:20	23.04.2023 03:20-03:40	23.04.2023 03:40-04:00
23.04.2023 04:00-04:20	23.04.2023 04:20-04:40	23.04.2023 04:40-05:00
23.04.2023 05:00-05:20	23.04.2023 05:20-05:40	23.04.2023 05:40-06:00
23.04.2023 07:00-07:20	23.04.2023 07:20-07:40	23.04.2023 07:40-08:00
23.04.2023 08:00-08:20	23.04.2023 08:20-08:40	23.04.2023 08:40-09:00
23.04.2023 09:00-09:20	23.04.2023 09:20-09:40	23.04.2023 00:40-01:00
23.04.2023 01:00-01:20	23.04.2023 01:20-01:40	23.04.2023 01:40-02:00
23.04.2023	23.04.2023	23.04.2023
Бронь		Другой маршрут

Рисунок 33):

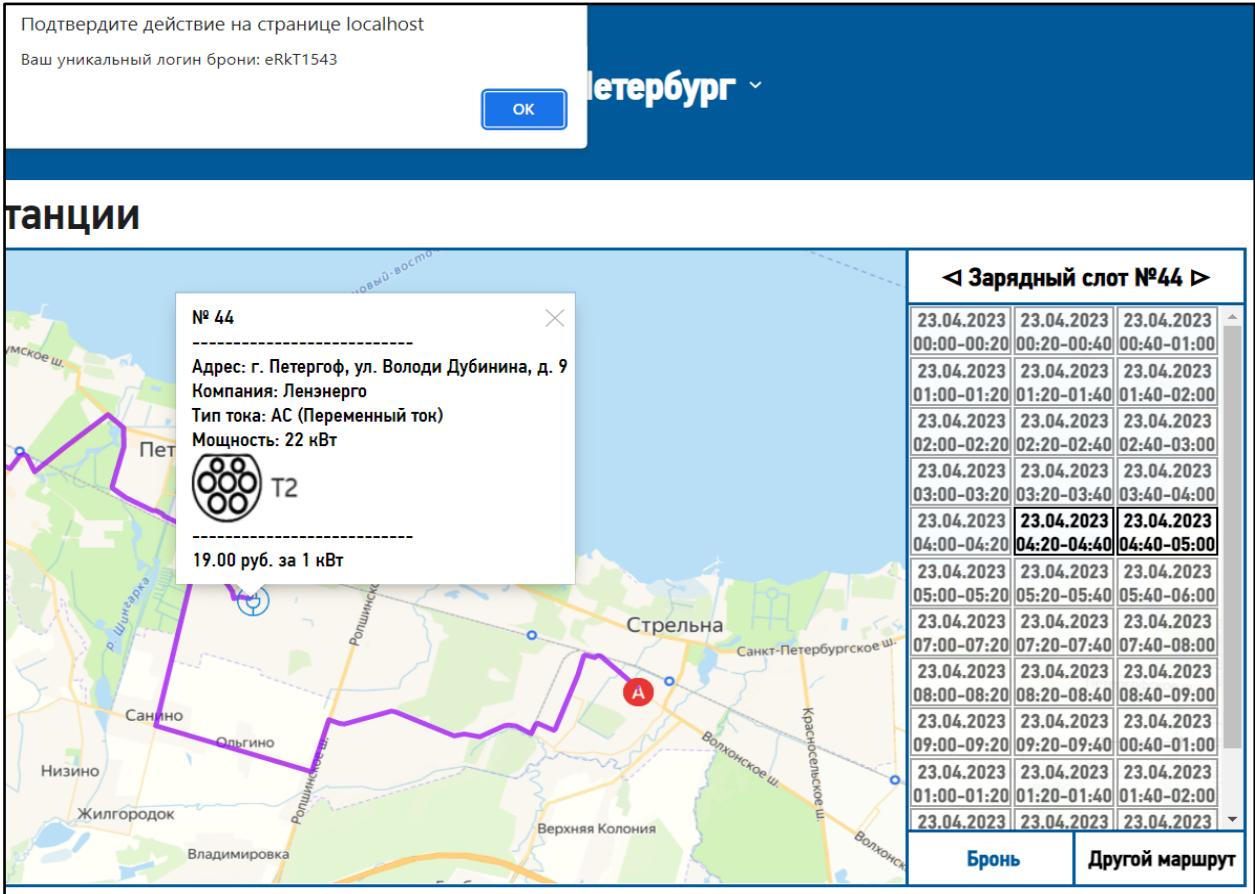


Рисунок 33. Дизайн сообщения о генерации логина подтверждения брони

Адаптивный пользовательский интерфейс для устройств с маленькой диагональю экрана или мобильных гаджетов выглядит вот так (*Рисунок 34*):

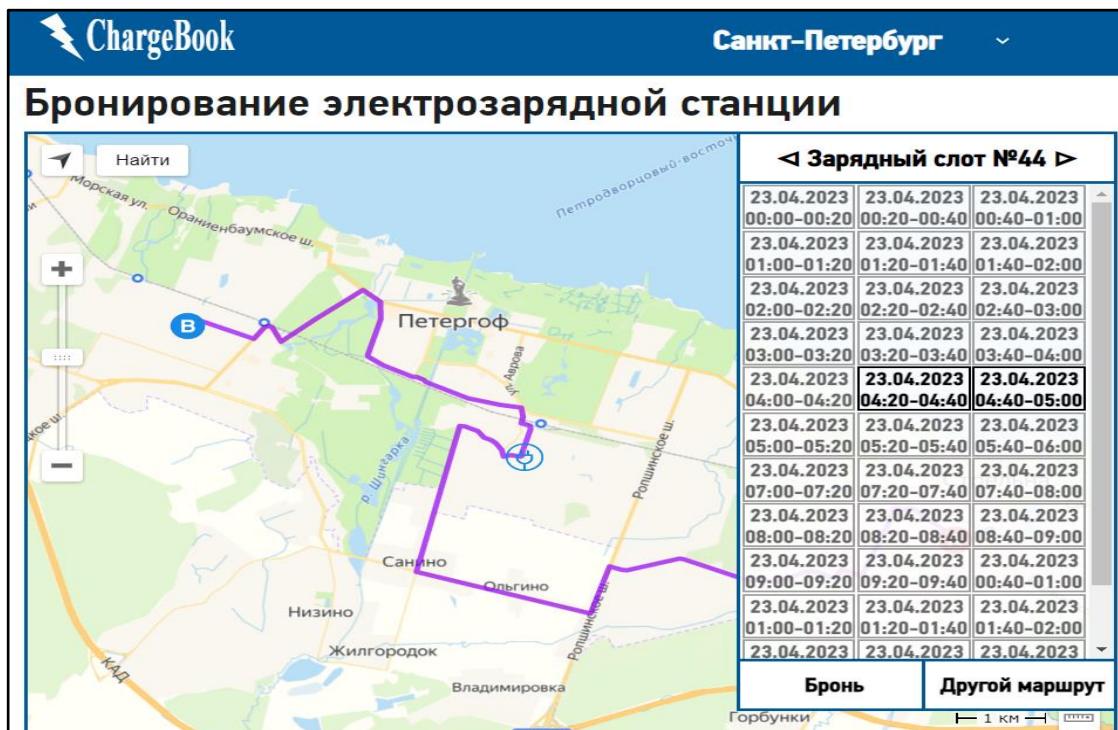


Рисунок 34. Адаптивный интерфейс последней вкладки

Таким образом, в данном пункте главы этой работы нам удалось ознакомиться с пользовательским интерфейсом рассматриваемого веб-приложения. Он полностью следует прогрессии бизнес-процессов, что делает его комфортным и интуитивно понятным. Представленный пользовательский интерфейс играет огромную роль в веб-приложении, позволяя его пользователям коммуницировать с его скрытыми алгоритмами наиболее быстрым способом и без необходимости в знании ими языков программирования .

3.4. Программная реализация выбранного метода интеллектуального управления эксплуатацией ЭЗС

После разбора фронтэнд-части стоит обратить внимание на работу бэкэнд-алгоритмов разрабатываемого веб-приложения. Именно данная инкапсулированная часть рассматриваемого веб-сервиса отвечает за движение данных, их последующую обработку и составления решения для поставленных пользователем задач. Необходимо отметить, что бэк-энд этого веб-приложения реализован с помощью двух языков программирования: PHP и Python. На первом языке сформирован контроллер маршрутов запросов и привязанные к ним функции для обращения к внешнему веб-сервису для сбора температуры окружающей среды, получения JSON-документа из MySQL базы данных слотов ЭЗС и их характеристиками, отфильтрованными согласно переданным параметрам, и добавления новых записей в JSON-документ с расписанием броней слотов ЭЗС. В то время как на втором языке программирования построен сам алгоритм, воплощающий логику ранее введённой математической модели метода (2.8). Все бэкэнд-скрипты вместе с общим каталогом веб-приложения представлены в разделах «ПРИЛОЖЕНИЕ А», «ПРИЛОЖЕНИЕ Б» и «ПРИЛОЖЕНИЕ В».

В рамках получения информации о погодных условиях в точке текущей геолокации или выбранном городе используется GET-запрос на веб-сервис

«Яндекс.Погода»: `$url='http://export.yandex.ru/bar/reginfo.xml?region=.'.$city.'.xml';`, где текстовая переменная «\$city» содержит полное название города или ближайшего крупного населённого пункта области, где находится или которые выбрал пользователь. После получения JSON-документа в ответе внешнего источника он подвергается процедуре декомпозиции: `$weather['temp'] = $xml -> weather -> day -> day_part[0] -> temperature;`, после которой другой JSON-документ записывается в переменную \$weather с единственным ключом «temp», идентифицирующим целое число, равное количеству градусов по Цельсию. Оно передаётся в поле ввода на пользовательском интерфейсе. Это зафиксированное в поле ввода число играет большую роль в реализации алгоритма-решения на Python.

Для работы с массивом слотов ЭЗС и их характеристиками используется реляционная база данных «MySQL», в которой есть одна таблица под названием «Stations». Список полей этой таблицы полностью совпадает с установленным перечнем параметров слотов ЭЗС, определённым в пункте №2.3 методологической части данной работы. Согласно ему запрос на создание таблицы должен выглядеть следующим образом:

```
'CREATE TABLE ELECTRO.Stations(id SMALLINT, address NVARCHAR(120), lat DECIMAL(10,6), lon DECIMAL(10,6), price NUMERIC(4,2), company NVARCHAR(25), plug NVARCHAR(25), power SMALLINT, plug_type NVARCHAR(2), status SMALLINT);'.
```

В рамках использования моделирования объектов ЭЗС используются официально зарегистрированные сертифицированные зарядные станции из базы данных ПАО «РОССЕТИ» [39], так как данная корпорация является основным поставщиком агрегатов такого рода в Российскую Федерацию, в связи с чем она ведёт тщательный учёт их параметров и работоспособности [40]. Посредством использования запросов по типу:

```
'INSERT INTO ELECTRO.Stations VALUES(2, "г. Выборг, Ленинградское ш., д. 46", 60.700894, 28.786423, 20, "Ленэнерго", "type2", 22, "ac", 1);'
```

заполняется целевая таблица, по отношению которой воспроизводятся SQL-запросы. Выглядит она следующим образом (*Рисунок 35*):

id	address	lat	lon	price	company	plug	power	plug_type	status
116	Московская обл., сельское поселение Борис...	55.463881	35.828770	25.00	Энергоце...	chademo	60	dc	1
117	Московская обл., сельское поселение Борис...	55.463881	35.828770	25.00	Энергоце...	gtbdc	120	dc	1
118	г. Руза, ул. Федеративная, д. 43	55.709459	36.204792	25.00	Энергоце...	ccscombo2	120	dc	1
119	г. Руза, ул. Федеративная, д. 43	55.709459	36.204792	25.00	Энергоце...	chademo	60	dc	1
120	г. Руза, ул. Федеративная, д. 43	55.709459	36.204792	25.00	Энергоце...	gtbdc	120	dc	1
121	Московская обл., рабочий посёлок Шаховск...	56.037322	35.485816	25.00	Энергоце...	ccscombo2	120	dc	1
122	Московская обл., рабочий посёлок Шаховск...	56.037322	35.485816	25.00	Энергоце...	chademo	60	dc	1
123	Московская обл., рабочий посёлок Шаховск...	56.037322	35.485816	25.00	Энергоце...	gtbdc	120	dc	1
124	г. Волокаламск, Рижское шоссе, 4Б	56.023978	35.956857	25.00	Энергоце...	ccscombo2	120	dc	1
125	г. Волокаламск, Рижское шоссе, 4Б	56.023978	35.956857	25.00	Энергоце...	chademo	60	dc	1
126	г. Волокаламск, Рижское шоссе, 4Б	56.023978	35.956857	25.00	Энергоце...	gtbdc	120	dc	1
127	Московская обл., осёлок Новолотошино, Тве...	56.250412	35.647028	25.00	Энергоце...	ccscombo2	120	dc	1
128	Московская обл., осёлок Новолотошино, Тве...	56.250412	35.647028	25.00	Энергоце...	chademo	60	dc	1
129	Московская обл., осёлок Новолотошино, Тве...	56.250412	35.647028	25.00	Энергоце...	gtbdc	120	dc	1
130	Московская обл., городской округ Истра	55.853989	36.781560	NULL	NULL	NULL	NULL	NULL	0
131	Московская обл., городской округ Клин, М-1...	56.411982	36.556707	25.00	Энергоце...	saej1772	7	ac	1
132	Московская обл., городской округ Клин, М-1...	56.411982	36.556707	25.00	Энергоце...	type2	22	ac	1
133	Московская обл., городской округ Клин, М-1...	56.411982	36.556707	25.00	Энергоце...	chademo	60	dc	1
134	Московская обл., городской округ Клин, М-1...	56.411982	36.556707	25.00	Энергоце...	ccscombo2	60	dc	1
135	Московская обл., городской округ Клин, М-1...	56.412236	36.557550	25.00	Энергоце...	saej1772	7	ac	1

Рисунок 35. Фрагмент реляционной базы данных со слотами ЭЗС

Фильтрация подходящих слотов и их упаковка в JSON-документ, используемый для отображения меток со всплывающими подсказками на интерактивной карте и Python-скриптом для выполнения рассчитывающих алгоритмов, происходит путём SQL-запроса к БД в функции, написанной на языке программирования PHP:

```
$statement = $con->prepare('SELECT * FROM
Stations WHERE (plug="'.(string)$plug.'" OR plug IS NULL) AND
(plug_type IN '.(string)$ac_dc.' OR plug_type IS NULL) AND
((power>='.(string)$from_power.' AND power<='.(string)$to_power.') OR
power IS NULL) AND ((price>='.(string)$from_price.' AND
price<='.(string)$to_price.') OR price IS NULL)'). В приведённом SQL-запросе текстовые переменные $plug, $ac_dc, $from_power, $to_power, $from_price, $to_price содержат характеристики подходящего слота зарядной станции, установленные или изменённые пользователем.
```

Функция перезаписи нереляционной базы данных броней слотов ЭЗС опирается на команду: `file_put_contents($path, JSON_encode($resp));`, где текстовая переменная \$path содержит путь из корневой папки к JSON-документу с зарегистрированными бронями, а переменная \$resp – обновлённый JSON-объект с добавленными в него зарезервированными временными окнами. Работа функции может быть продемонстрирована следующим образом (*Рисунок 36*):

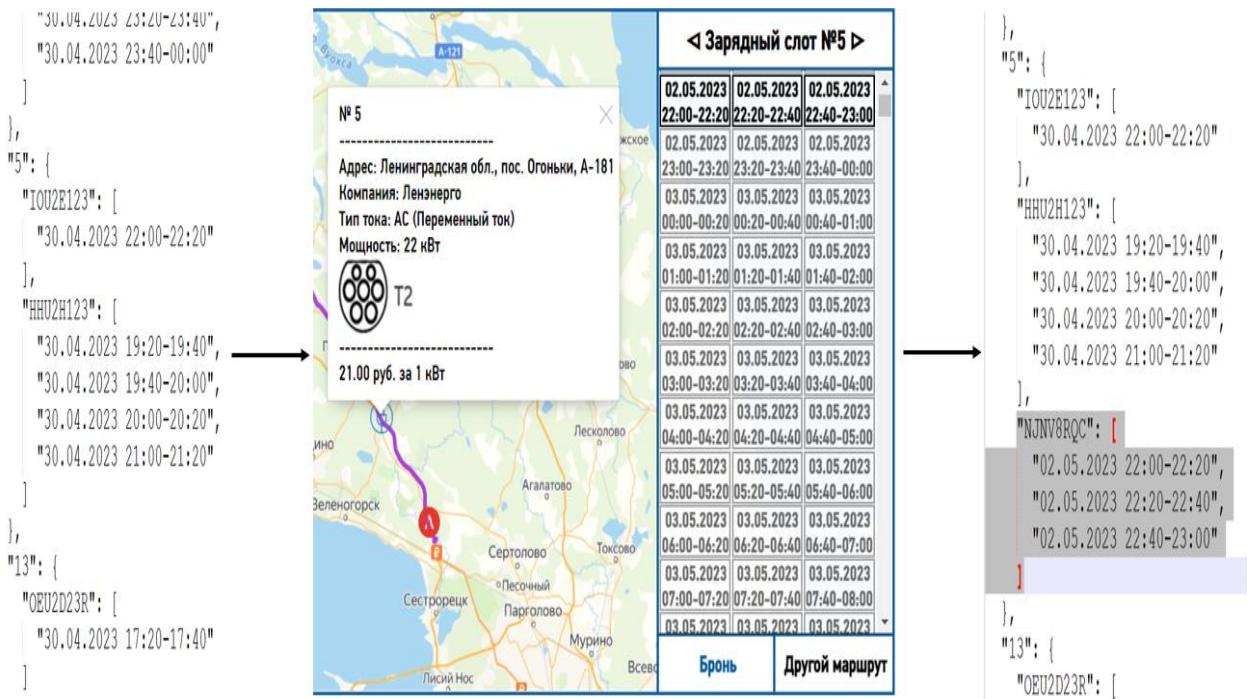


Рисунок 36. Результат работы функции перезаписи расписания

Также стоит обратить внимание на основной бэкэнд-скрипт, написанный на языке программирования Python и отвечающий за реализацию алгоритма решения поставленной пользователем задачи. Как и заявляется в пункте №2.3 методологической части этой работы в нём представлены два подхода: алгоритм Дейкстры и генетические алгоритмы, сравнительный анализ которых так же реализован в коде программы.

Прежде всего стоит разобрать библиотеки, которые используются в бэкэнд-алгоритме:

1) flask [41];

Ранее упомянутый веб-фреймворк, предлагающий полезные инструменты и функции для облегчения процесса создания веб-приложений с использованием языка программирования Python.

2) flask_cors [42];

Дополнение к библиотеке «flask», позволяющее последней реализовывать стандарт «Cross-Origin Resource Sharing» (совместного использования ресурсов разных источников). Он предоставляет возможность в приёме данных, переданных посредством AJAX-протокола.

- 3) openrouteservice [43];

Библиотека, предоставляющая пользовательский API для использования глобальных пространственных сервисов, работа которых основана на платформе «OpenStreetMap» [44]. В рамках рассматриваемого проекта она используется для определения расстояния оптимального маршрута и времени его прохождения в конкретный период времени с учётом текущей дорожно-транспортной ситуации. Основным преимуществом данного инструмента является тот факт, что этот API полностью бесплатный, тем не менее, в качестве недостатка можно отметить ограничение по числу запросов на получение массивов параметров маршрутов в секунду.

- 4) math [45];

Стандартная библиотека, вмещающая в себя множество математических функций. Предоставляемый ею метод вычисления натурального логарифма применяется при расчёте введённой в методологической части этой работы целевой функции (2.7.2).

- 5) random [46];

Стандартная библиотека, позволяющая использовать функции, специализирующиеся на генерации псевдослучайных чисел, а также на работе с различными распределениями вероятностей. В рамках рассматриваемого проекта её инструментарий используется для моделирования стохастических процессов при формировании метода решения задачи посредством генетических алгоритмов.

- 6) string [47].

Стандартная библиотека, позволяющая проводить операции с символами кодировки «ASCII». В рамках рассматриваемого проекта её функционал используется для генерации уникальных логинов для последующей привязки броней пользователей к ним.

В связи с тем, что бэкэнд-скрипт включает в себя два метода, то логично предположить, что он оперирует двумя классами (*Рисунок 37*):

```

# алгоритм Дейкстры
class Dijkstra:
    def __init__(self, jsc, params):
        # получение матрицы смежности из json-документа
    def matrix(self):
        # реализация алгоритма Дейкстры
    def solve(self, parameter):
        ...

# генетические алгоритмы
class Genetic(Dijkstra):
    def __init__(self, jsc, params):
        super().__init__(jsc, params)
        # получение матрицы смежности из json-документа
    def matrix(self):
        return super().matrix()
        # реализация генетических алгоритмов
    def solve(self, parameter):
        ...

```

Рисунок 37. Классы бэкэнд-алгоритма

Можно заметить, что, помимо функции присвоения значений классовым переменным, у обоих классов есть ещё один общий метод формирования матрицы смежности, который наследуется классом генетических алгоритмов от класса-родителя алгоритма Дейкстры согласно парадигме объектно-ориентированного программирования. Также у обоих классов есть методы реализации своих алгоритмов, которые, несмотря на одинаковое название, согласно принципу полиморфизма объектно-ориентированного программирования исполняются по абсолютно разной логике, присущей характеру класса, к которому они принадлежат.

Стоит отметить, что матрица смежности представляет из себя вид представления графа в виде матрицы, где пересечение столбцов и строк, то есть его вершин, формирует рёбра графа между ними. В данном формате пересечениям (рёбрам) присваиваются свойственные им веса, рассчитываемые согласно утверждённой логике решения поставленной задачи и необходимые для последующего приложения к ним инструментов поиска оптимального пути в ориентированном графе. В рамках текущего проекта матрица смежности представляется в виде ассоциативного массива (словаря),

так как данный способ формирования ориентированного графа даёт возможность в более простом применении по отношению к нему вложенных циклов при совершении операций над составляющими его рёбрами. Алгоритм конструирования ассоциативного массива (матрицы смежности) в рассматриваемом подходе к решению задачи выглядит следующим образом:

- 1) фильтрация нерабочих зарядных станций в полученном массиве подходящих слотов ЭЗС для уменьшения размера выборки. Она включает в себя только функционирующих агентов моделирования после исполнения команды: `nodes = [obj for obj in self.stations if obj['plug_type'] != None]`.
- 2) последующая фильтрация слотов ЭЗС, находящихся в области допустимости совершения маршрута. Она осуществляется с помощью специально разработанного подхода, заключающегося в поиске середины отрезка между начальной и конечной точками маршрута и заключении в область фильтрации тех слотов ЭЗС, чьё евклидово расстояние между их координатами местоположения и координатами середины отрезка между начальной и конечной точками маршрута не больше 120% евклидового расстояния половины отрезка, разделённого ранее упомянутой срединной точкой. Графическая интерпретация такого подхода выглядит следующим образом (**Рисунок 38**):

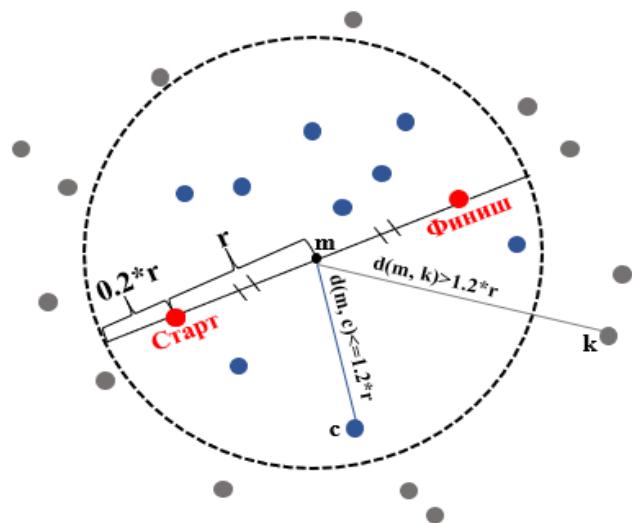


Рисунок 38. Графическая интерпретация определения области допущения маршрута

3) формирование маршрутов между оставшимися вершинами и назначение им весов согласно логике математической модели (2.8) для построения матрицы смежности, представленной в виде ассоциативного массива. Ассоциативный массив обладает следующей структурой (вершина «0» – начальная точка маршрута, а вершина «81» – конечная) (*Рисунок 39*):

```
{"0": {"72": 1255.51, "80": 895.62},
"72": {"80": 1028.07, "81": 746.54},
"76": {"72": 1124.84, "80": 1002.21, "81": 742.52},
"80": {"76": 1232.92, "81": 683.4}}
```

Рисунок 39. Структура ассоциативного массива взвешенных рёбер

Функция реализации алгоритма Дейкстры в соответствующем классе представляет из себя обычную интерпретацию такого метода, нацеленного на минимизацию совокупного веса пути в ориентированном графе. Хорошим примером работы алгоритма может служить схема ориентированного графа, демонстрирующая, как он постепенно ищет путь к каждой вершине с наименьшим совокупным весом, требуемым для её достижения до того момента, как будет затронута финальная целевая вершина (*Рисунок 40*) [48]:

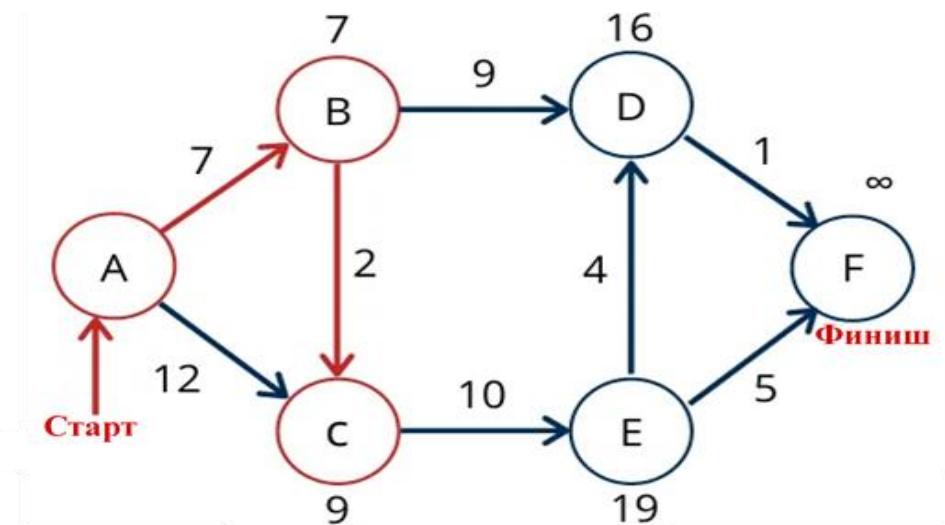


Рисунок 40. Схематичное представление работы алгоритма Дейкстры

Функция реализации генетических алгоритмов в соответствующем классе представляет из себя применение случайной совокупности операторов мутации и кроссовера в случайном порядке с целью достижения необходимой

совокупности вершин, состоящей из бинарной последовательности, где 1 – вершина посещается, а 0 – игнорируется. Она должна обеспечивать построение пути от начальной до конечной вершин, который при этом будет обладать наименьшим совокупным весом. Работу эволюционных операторов в данном алгоритме можно представить следующим образом (*Рисунок 41*):



Рисунок 41. Механика эволюционных операторов

Эволюционные операторы вызываются до тех пор, пока не произойдёт удовлетворение одного из условий, являющегося ограничительным параметром работы всего алгоритма. После этого отбирается тот получившийся путь, который продемонстрировал наименьший совокупный вес за всё время работы функции. В рамках рассматриваемого проекта значения параметров могут быть представлены следующим списком:

- `success_rate = 10 #число хромосом, прошедших естественный отбор (формирующих путь);`
- `rep_rate = 100000 #предельное количество поколений (итераций).`

Стоит отметить, что уже было заявлено, что в этой работе будет проведён сравнительный анализ двух ранее описанных подходов с целью выбора оптимального из них в качестве ядра решающего алгоритма. В связи с этим на одних и тех же выборках были проведены тесты, которые позволяют свидетельствовать о превосходстве одного метода над другим. Код для проведения экспериментов закомментирован, а результаты проведённого тестирования задокументированы и зафиксированы в таблице (*Таблица 3*):

Таблица 3. Результаты сравнительного анализа

Метрика	Число вершин, шт.	Число итераций, шт.	Алгоритм Дейкстры	Генетические алгоритмы
Средняя продолжительность выполнения, сек.	20	5	0.0001	1.382111
	15		9.8e-05	1.225703
	10		4.8e-05	1.095966
	5		0.0...	0.908882
Средняя условная стоимость маршрута, руб.	20	5	1067.08	1067.08
	15		1067.08	1067.08
	10		1067.08	1067.08
	5		2282.67	2282.67

Исходя из результатов тестирования можно сделать вывод о том, что оба подхода демонстрируют одинаковую эффективность, так как каждый из них пришёл к минимальным условным стоимостям маршрутов в условиях каждого эксперимента. Тем не менее, отчётливо видно, что алгоритм Дейкстры работает в разы быстрее. В связи с этим можно сделать вывод о том, что алгоритм Дейкстры представляется оптимальным.

Ввиду доказанного превосходства алгоритма Дейкстры над генетическими алгоритмами он будет служить ядром решающего алгоритма. В связи с этим блок решения поставленной задачи будет выглядеть следующим образом (*Рисунок 42*):

```

# решение задачи

tasks = Dijkstra(jsc, params)

login = ''.join(random.choice(string.ascii_uppercase +
                               string.digits) for _ in range(8))

solution = {login: tasks.solve('path')}

```

```

# формирование массива оптимального пути
res['path'][0] = '|-->'
res['path'][-1] = '-->|'
if res['path'] == ['-->|']:
    error = 5 / 0
return res[parameter]
except:
    res = {'path': 'impossible',
           'cost': 'impossible'}
return res[parameter]

```

{'MVR1CVNM': ['|-->', '164', '-->|']}

Рисунок 42. Блок решения задачи

Можно заметить, что функции реализации алгоритма класса Дейкстры передаётся текстовый параметр «path», который в свою очередь служит ключом для возвращаемого ею ассоциативного массива, предоставляющим доступ к списку id слотов ЭЗС, участвующих в построении оптимального маршрута. Более того, функция класса работает с конструкцией «try-except», которая при возникновении ошибки в блоке «try» не прерывает выполнение программы, а переходит к командам в блоке «except». Это позволяет формировать сигнал о невозможности построения маршрута для фронтэнд-части без инициализации ошибок подключения. Возвращаемый JSON-документ с ключом, представляющим из себя сгенерированный уникальный логин для регистрации брони, и привязанным к нему значением в виде массива id слотов зарядных станций маршрута, впоследствии используется для визуализации результатов на интерактивной карте и проведения процедуры резервации временных окон.

Таким образом, в данном пункте практического раздела нам в полной мере удалось познакомиться со спецификами программной реализации предложенного метода. Основные аспекты бэкэнд-алгоритмов на нескольких языках программирования были рассмотрены и проанализированы, что привело к результату полного понимания работы инкапсулированной части веб-приложения, отвечающей за реализацию решения поставленной в данной работе задачи.

3.5. Экономическое обоснование разработанного веб-приложения

Рассчитаем экономический эффект, оказываемый предлагаемым веб-приложением, на финансовое состояние компании на основе расчёта одного из важнейших показателей, демонстрирующего эффективность внедряемого ИТ-решения на каком-либо предприятии: доходности собственного капитала (ROE). Он определяет, какой объём чистой прибыли приносит компания на вложенный капитал. Для этого будет использоваться популярная модель стратегической прибыли от консалтинговой компании «DuPont» [49], основывающаяся на данных из бухгалтерского баланса, а также отчёта о прибылях и убытках от 2022 года ПАО «РОССЕТИ» (ПРИЛОЖЕНИЕ Г) [50], которое является поставщиком данных для данного проекта. Этот инструмент предоставит возможность в наглядной презентации взаимозависимостей между финансовыми показателями предприятия и влияния изменений их значений в связи с внедрением разработанного веб-приложения на ранее указанный важный индикатор.

Прежде всего стоит вспомнить о том, что на данный момент в фронт-энд-алгоритме и бэкэнд-алгоритме используются бесплатные версии API сервисов «Яндекс.Карты» и «openrouteservice» для взаимодействия с интерактивной картой и построения маршрутов соответственно. Они функционируют с ограничением в числе отправляемых запросов в секунду, что создаёт большие сложности при выпуске веб-приложения в

промышленную эксплуатацию с большим числом пользователей, одновременно обращающихся к этому веб-сервису. По этой причине в случае внедрения следует рассмотреть платные варианты API для маршрутизации, способные беспрепятственно обрабатывать большой объём запросов за короткий период времени. В связи с этим обратимся к платному варианту API сервиса «Яндекс.Карты», позволяющему как взаимодействовать с интерактивной картой в пользовательском интерфейсе, так и строить маршруты в инкапсулированной части приложения. Изучим его тарифную сетку, которая выглядит следующим образом (**Таблица 4**):

Таблица 4. Тарифная сетка доступа к API "Яндекс.Карты" [51]

Лимит запросов в сутки, шт.	Стоимость в год, руб.	Стоимость тысячи запросов сверх лимита, руб.
1000	120000	240
10000	360000	120
25000	680000	90
50000	1000000	90
100000	1200000	90

Также обратимся к анализу фактора распространённости использования электротранспорта в России (**Рисунок 43**):

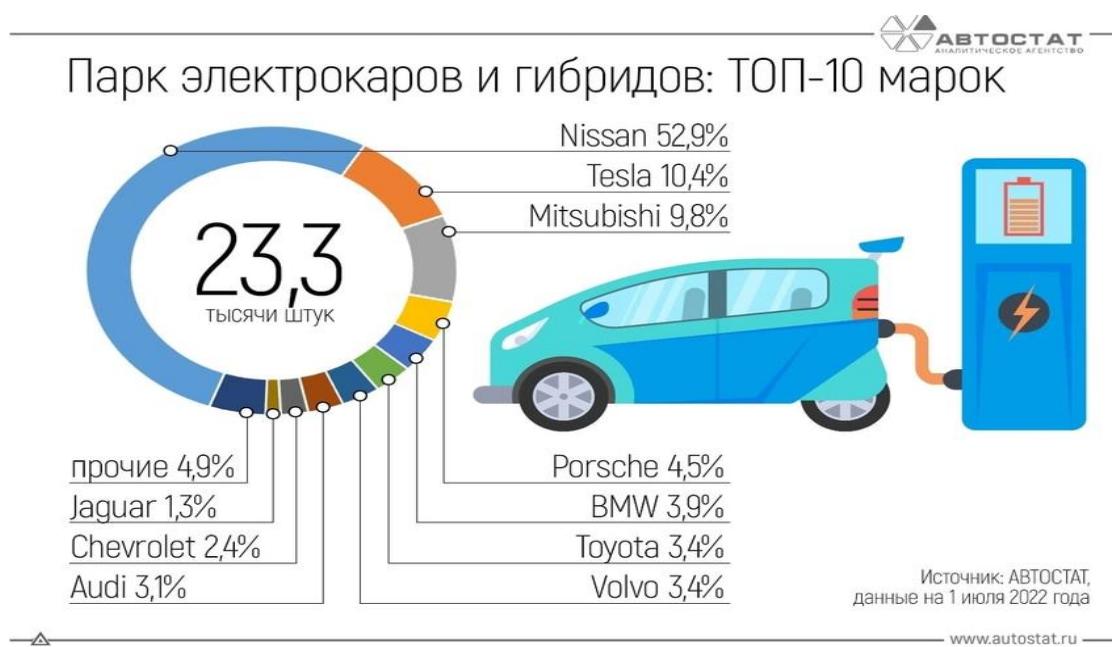


Рисунок 43. Статистика распространённости электромобилей в России [52]

В связи со всеми вышеизложенными данными можно прийти к выводу о том, что лицензия с примерной стоимостью в 360000 рублей на лимит запросов в количестве 10000 штук за сутки вполне удовлетворяет выдвинутым условиям. Помимо этого, что в веб-приложении также используются результаты работы сервиса «Яндекс.Погода», при использовании которых для коммерческих целей так же потребуется платный вариант API. Изучим его тарифную сетку, которая выглядит следующим образом (**Таблица 5**):

Таблица 5. Тарифная сетка доступа к API "Яндекс.Погода" [53]

Тариф	Стоимость	Лимит запросов	Прогноз
«Погода на вашем сайте»	Не тарифицируется	50 в сутки	Текущая погода и 2 последующих периода
«Тестовый»	Не тарифицируется 30 дней	5000 в сутки	7 дней
«Коммерческий»	От 30 000 ₽ в месяц	От 1 500 000 в месяц	До 10 дней
«Коммерческий расширенный»	От 60 000 ₽ в месяц	От 1 500 000 в месяц	Без ограничений

В связи со всеми вышеизложенными данными можно прийти к выводу о том, что лицензия с примерной стоимостью в 30000 рублей в месяц на лимит запросов в количестве от 1500000 штук в месяц вполне удовлетворяет выдвинутым условиям.

Помимо затрат, стоит рассмотреть потенциальные выгоды от внедрения этого веб-приложения в компанию, выражющиеся в дополнительных доходах для неё. Согласно информации по распространённости использования электромобилей в России [52] можно предположить, что внедрение такого веб-сервиса на официальном вебсайте этого крупного холдинга исходя из пессимистичного прогноза привлечёт к его использованию хотя бы половину владельцев электрокаров в стране, то есть около 11650 отдельных пользователей в течение месяца. При использовании ими предлагаемого веб-

приложения для бронирования зарядных станций хотя бы 1 раз в неделю число обращений к ИТ-решению будет равняться 605800 в год, который в среднем содержит в себе 52 недели. В качестве стоимости регистрации одной брони от пользователя на эксплуатацию слота ЭЗС можно принять всего лишь 50 рублей, так как эта цифра не является большой дополнительной тратой для владельца электрокара, который при среднем тарифе потребления 1 кВт·ч на ЭЗС, равному 15 рублям [31], и средней ёмкости аккумулятора электромобиля, равной 40 кВт·ч [54], будет дополнительно тратить на резервацию временного периода использования зарядной станции всего лишь 8.3% от общей стоимости полной зарядки батареи, равной 600 рублям.

Годовые инвестиции для внедрения и поддержки функциональности рассматриваемого веб-приложения будут равны 7720000 рублей, то есть сумме утверждённого бюджета проекта согласно внутренней документации компании, равного 7000000 рублей, и совокупности стоимостей годовых лицензий использования пользовательских API сервисов «Яндекс.Карты» и «Яндекс.Погода». Потенциальные доходы, в свою очередь, в связи с предполагаемым числом пользователей и установленной стоимостью регистрации брони будут равны 30290000 рублей. Параметры фрагмента такого бизнес-плана могут быть зафиксированы следующим образом (**Таблица 6**):

Таблица 6. Фрагмент бизнес-плана внедрения веб-сервиса

№	Объект моделирования	Период	Значение	Порядок вычисления
1	Доступ к API "Яндекс.Карты", руб.	1 год	360000	[51]
2	Доступ к API "Яндекс.Погода", руб.	1 месяц	30000	[53]
2.1	Доступ к API "Яндекс.Погода", руб.	1 год	360000	№1 * 12

3.2	Число владельцев электромобилей в стране, чел.	1 месяц	23300	[52]
3.3	Прогноз привлечённого числа пользователей, чел.	1 месяц	11650	№3.2 / 2 (прогноз)
3.4	Прогноз привлечённого числа пользователей, чел.	1 год	11650	№3.3
4.1	Частота обращений к веб-сервису одного пользователя, шт.	1 неделя	1	прогноз
4.2	Частота обращений к веб-сервису одного пользователя, шт.	1 год	52	№4.1 * 52
5	Совокупное число обращений привлечённых пользователей	1 год	605800	№3.4 * №4.2
6	Средняя стоимость потребления 1 кВт-ч, руб.	-	15	[31]
7	Средняя ёмкость аккумулятора электромобиля, кВт-ч	-	40	[54]
8	Средняя стоимость полной зарядки аккумулятора, руб.	-	600	№6 * №7
9	Тариф стоимости регистрации брони от средней стоимости полной зарядки аккумулятора, %	-	8.3	прогноз
10	Стоимости регистрации брони, руб.	-	50	№8 * №9
11	Стоимость внедрения веб-сервиса, руб.	1 год	7000000	бюджет проекта по внутренней документации
12	Предполагаемые расходы, руб.	1 год	7720000	№1 + №2.1 + №11
13	Предполагаемые доходы, руб.	1 год	30290000	№5 * №10

Данные затраты и доходы будут отражены в модели стратегической прибыли, реализация которой для предлагаемого ИТ-решения выглядит следующим образом (*Рисунок 44*):

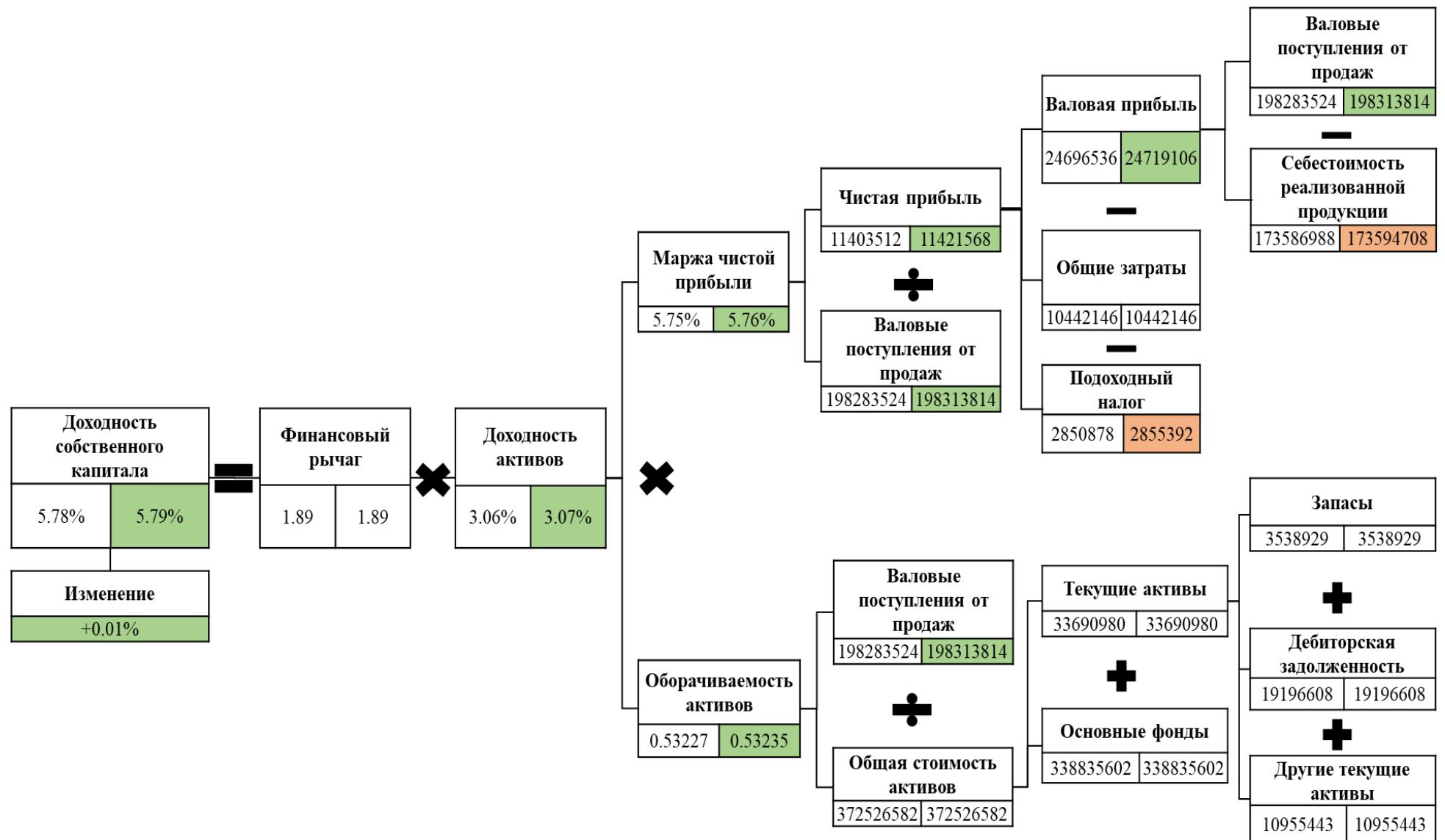


Рисунок 44. Модель стратегической прибыли «DuPont» (в тыс. руб.)

Из предложенной модели можно сделать вывод, что себестоимость реализованной продукции увеличивается на 7720000 рублей в связи со всеми упомянутыми инвестициями,ложенными в веб-приложение, в то время как валовые поступления от продаж увеличивается на 30290000 рублей ввиду предложения новой услуги по бронированию слотов зарядных станций. В связи с этими изменениями в показателях происходит увеличение валовой прибыли на 22570000 рублей, и при неизменных величинах общих затратах из-за отсутствия влияния на структуру постоянных и переменных издержек она увеличивает значение подоходного налога на 4514000 рублей. Данная цепочка изменений влечёт за собой увеличение чистой прибыли на 18056000 рубля, что увеличивает маржу чистой прибыли на примерно 0.01%. В связи с отсутствием влияния внедрения веб-приложения на структуру баланса холдинга и увеличением валовых поступлений от продаж значение коэффициента оборачиваемости активов увеличивается на примерно 0.00008. Ввиду данной цепочки изменений величина доходности активов возрастает на примерно 0.01%, что при неизменном финансовом рычаге из-за отсутствия изменений в структуре баланса холдинга приводит к увеличению доходности собственного капитала на примерно 0.01%. Таким образом, целевой показатель модели действительно увеличивается после внедрения рассматриваемого веб-приложения, что доказывает приносимый им положительный экономический эффект для компаний.

Таким образом, с резюме применения такого инструмента, как модель стратегической прибыли Дюпона, можно заявить, что предложенный интеллектуальный веб-сервис произвёл благоприятное влияние на один из важнейших финансовых показателей: доходность собственного капитала. Его увеличение означает, что эффективность работы собственных средств собственников компании, инвестированных в предприятие, возросла. Другими словами, каждый рубль владельцев,ложенный в рассматриваемый холдинг, теперь приносит им большую сумму доходов.

Выводы по главе 3

Таким образом, в практической части данной работы нам удалось разобрать непосредственный процесс реализации адаптированного метода для решения поставленных задач. В ней было затронуто моделирование бизнес-процессов в нотации EPC для составления логики работы веб-приложения и получения понимания всей прогрессии действий, происходящих в нём за его полный цикл работы. Результаты моделирования также позволили нам сформировать необходимую модель ИТ-инфраструктуры, объясняющую взаимосвязь элементов веб-сервиса и принципы их коммуникации между друг другом. Всё это предоставило возможность в разработке интуитивно понятного пользовательского интерфейса, полностью повторяющего логику прогрессии бизнес-процессов и удовлетворяющего требованиям предложенной модели ИТ-инфраструктуры. Данная подготовка позволила создать каркас для последующей программной реализации адаптированного для решения поставленной задачи метода на нескольких языках программирования, которая включала в себя использование двух подходов, эффективность работы которых была подвергнута сравнительному анализу с последующим выбором ядра алгоритма-решения.

Более того, в практической части данной работы удалось количественно ценить экономический эффект, который принесёт внедрение веб-приложения в рассматриваемый холдинг, являющимся поставщиком данных для этого проекта. Моделирование частичного бизнес-плана эксплуатации рассматриваемого ИТ-решения и использование модели стратегической прибыли «DuPont» позволило объективно показать, насколько может быть полезен данный интеллектуальный веб-сервис бронирования зарядных станций для финансового положения компании.

ЗАКЛЮЧЕНИЕ

Таким образом, цель данной работы можно считать достигнутой, а её задачи выполненными, так как в ней удалось успешно разобрать и реализовать разработку веб-приложения, представляющего собой интеллектуальный сервис по бронированию электронных зарядных станций для владельцев электрокаров и обладающего понятным интерфейсом, позволяющим любому пользователю без знаний этапов протекания бизнес-процесса и основ программирования получить качественную услугу. В рамках данного проекта были разобраны как теоретические аспекты, необходимые для определения фундамента, на котором строился целевой продукт проекта, так и методологические особенности, позволяющие конкретизировать индивидуальный метод решения поставленной задачи, отвечающий требованиям её конкретных условий и окружения. Погружение в эти области исследования проблемы позволило успешно осуществить практическую часть проекта, заключающуюся в подготовке к программной реализации вводимого метода и непосредственно самом её процессе для получения готового ИТ-решения, полностью удовлетворяющего определённые для него критерии. Более того, экономический эффект от внедрения разработанного веб-сервиса был количественно обоснован на примере реальной компании с помощью конкретных моделей финансового консалтинга.

Результаты, достигнутые в процессе выполнения данной работы, позволили подробно изучить проблему интеллектуального управления инфраструктурой зарядных станций. Коллaborация полученных знаний и сформированных на их основе практических навыков стали результатом успешной разработки ИТ-продукта, играющего роль инструмента для её решения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Немецкая компания, специализирующаяся на сборе рыночных и потребительских данных, «Statista» [Электронный ресурс]//Режим доступа: <https://www.statista.com/outlook/mmo/electric-vehicles/russia#unit-sales> (дата обращения: 06.02.2023);
2. Официальный интернет-ресурс для информирования о социально-экономической ситуации в России «Объясняем.рф» [Электронный ресурс]//Режим доступа:
https://xn--90a1vcdt6dxbc.xn--p1ai/articles/questions/transport-tourism/rabota_transporta/skolko_v_rossii_zaryadnykh_stantsiy_dlya_elektroavtomobiley (дата обращения: 06.02.2023);
3. Devashish Gupta, Dr. Sunil K. Singh (2022). Evolution of the WEB 3.0: History and the Future, Insights2Techinfo, pp.1;
4. Manishkumar R Solanki, Abhijit Dongaonkar (2016). A Journey of Human Comfort: WEB 1.0 to WEB 4.0, International Journal of Research and Scientific Innovation (IJRSI), Volume III, Issue IX, pp. 75-78;
5. Организация, разрабатывающая и внедряющая технологические стандарты, «Консорциум Всемирной паутины» (W3C) [Электронный ресурс]//Режим доступа: <https://www.w3.org/TR/rdf11-primer/> (дата обращения: 08.02.2023);
6. Dr. Adamko, Attila (2014). Internet Tools and Services, University of Debrecen, Faculty of Informatics, chapter 3;
7. Международная компания, занимающаяся продажей услуг в сфере информационно-коммуникационных технологий, «Delaware» [Электронный доступ]//Режим доступа: <https://www.delaware.pro/en/be/solutions/intelligent-apps> (дата обращения: 13.02.2023);
8. Американская компания, занимающаяся разработкой программной платформой типа «low-code», «Mendix» [Электронный доступ]//Режим доступа: <https://www.mendix.com/smart-apps> (дата обращения: 13.02.2023);

9. Американская компания, специализирующаяся на разработке программных продуктов и веб-приложений, «LITSLINK» [Электронный ресурс]//Режим доступа: <https://litslink.com/blog/web-application-architecture> (дата обращения: 13.02.2023);
10. Международная компания, проводящая опросы по использованию технологий сети Интернет, «W3Techs» [Электронный ресурс]//Режим доступа: <https://w3techs.com/technologies/details/pl-php> (дата обращения: 16.02.2023);
11. Ashwani Kumar, Ravinder Kumar, Ashutosh Aggarwal (2022). A multi-objective route planning and charging slot reservation approach for electric vehicles considering state of traffic and charging station, Journal of King Saud University - Computer and Information Sciences (34:5), pp. 2192-2206;
12. Yong Wang, Jingxin Zhou, Yaoyao Sun, Jianxin Fan, Zheng Wang, Haizhong Wang (2023). Collaborative multidepot electric vehicle routing problem with time windows and shared charging stations, Expert Systems with Applications (219), pp. 1-29;
13. Radu Flocea, Andrei Hîncu, Andrei Robu, Stelian Senocico, Andrei Traciu, Baltariu Marian Remus, Maria Simona Raboaca, Constantin Filote (2022). Electric Vehicle Smart Charging Reservation Algorithm, Sensors (22, 2834), pp. 1–16;
14. Yisheng An, Yuxin Gao, Naiqi Wub, Jiawei Zhu, Hongzhang Li, Jinhui Yang (2023). Optimal scheduling of electric vehicle charging operations considering real-time traffic condition and travel distance, Expert Systems With Applications 213, pp. 1–18;
15. Российский разработчик и производитель электротехнической преобразовательной техники «Парус Электро» [Электронный ресурс]//Режим доступа: <https://parus-electro.ru/company/> (дата обращения: 22.02.2023);
16. Каталог услуг по установке зарядных станций от российского разработчика и производителя электротехнической преобразовательной

- техники «Парус Электро» [Электронный ресурс]//Режим доступа: https://parus-electro.ru/presentations/09_ev_charger_station.pdf (дата обращения: 22.02.2023);
17. Российский производитель зарядных станций для электромобилей и программного обеспечения для работы зарядной инфраструктуры «TOUCH» [Электронный ресурс]//Режим доступа: <https://touch-station.com/blog/howtocharge> (дата обращения: 22.02.2023);
18. Австрийский производитель зарядных станций переменного тока для электромобилей «го-е» [Электронный ресурс]//Режим доступа: <https://go-e.com/ru/zhurnal/zaryadka-peremennym-i-postoyannym-tokom> (дата обращения: 22.02.2023);
19. Iliassov F.N (2022). What is the "current strength" and what does the "ammeter" measure, Moscow: IC Orion, pp. 1-9;
20. Shahjalal M., Shams T., Tasnim M. N., Ahmed M.R., Ahsan M., Haider J. (2022). A Critical Review on Charging Technologies of Electric Vehicles, Energies (15), pp. 8239;
21. Ассоциация, посвящённая производству электроэнергии и электричества в Австралии, «Australian Energy Council» [Электронный ресурс]//Режим доступа: <https://www.energycouncil.com.au/analysis/evs-are-they-really-more-efficient> (дата обращения: 01.03.2023);
22. Набор приложений, построенных на основе бесплатного картографического сервиса и технологий, предоставляемых компанией «Google» [Электронный ресурс]//Режим доступа: https://www.google.com/intl/ru_RU/maps/about/#/ (дата обращения: 01.03.2023);
23. Поисково-информационная картографическая служба Яндекса [Электронный ресурс]//Режим доступа: <https://yandex.ru/support/maps/> (дата обращения: 01.03.2023);
24. Safak Bayram (2021). Impacts of Electric Vehicle Charging under Cold Weather on Power Networks, IEEE, pp. 1–6;

25. Paolo Iora, Laura Tribioli (2019). Effect of Ambient Temperature on Electric Vehicles' Energy Consumption and Range: Model Definition, and Sensitivity Analysis Based on Nissan Leaf Data, World Electric Vehicle Journal (10, 2), pp. 1–15;
26. D. Ramsey, A. Bouscayrol, L. Boulon, and A. Vaudrey (2020). Simulation of an electric vehicle to study the impact of cabin heating on the driving range, IEEE 91st Vehicular Technology Conference (VTC2020-Spring), pp. 1–5;
27. J. R. M. D. Reyes, R. V. Parsons, and R. Hoemsen (2016). Winter happens: The effect of ambient temperature on the travel range of electric vehicles, IEEE Transactions on Vehicular Technology, vol. 65, no. 6, pp. 4016–4022;
28. Hamza Mediouni, Amal Ezzouhri, Zakaria Charouh, Khadija El Harouri, Soumia El Hani, Mounir Ghogho (2022). Energy Consumption Prediction and Analysis for Electric Vehicles: A Hybrid Approach, Energies (15, 6490), pp. 1–17;
29. Yazan Al-Wreikat, Clara Serrano, Jos'e Ricardo Sodr'e (2021). Driving behaviour and trip condition effects on the energy consumption of an electric vehicle under real-world driving, Applied Energy (297), pp. 1–6;
30. Официальный сайт Мэра Москвы [Электронный ресурс]//Режим доступа:
<https://www.mos.ru/mayor/themes/2299/9032050/>
(дата обращения: 27.03.2023);
31. Российская компания «Electro.cars», специализирующаяся на установке и ремонте зарядных станций в Москве [Электронный ресурс]//Режим доступа:
<https://electro.cars/tpost/hz28153111-ckolko-stoit-zaryadit-elektromobil-v-202>
(дата обращения: 27.03.2023);
32. Kostas Kollias, Sreenivas Gollapudi (2021). Addressing Range Anxiety with Smart Electric Vehicle Routing, Google Research;
33. Shixiong Jiang, Weijing Pan (2020). Dynamic-Area-Based Shortest-Path Algorithm for Intelligent Charging Guidance of Electric Vehicles, Sustainability 12(18):7343, pp. 1–20;

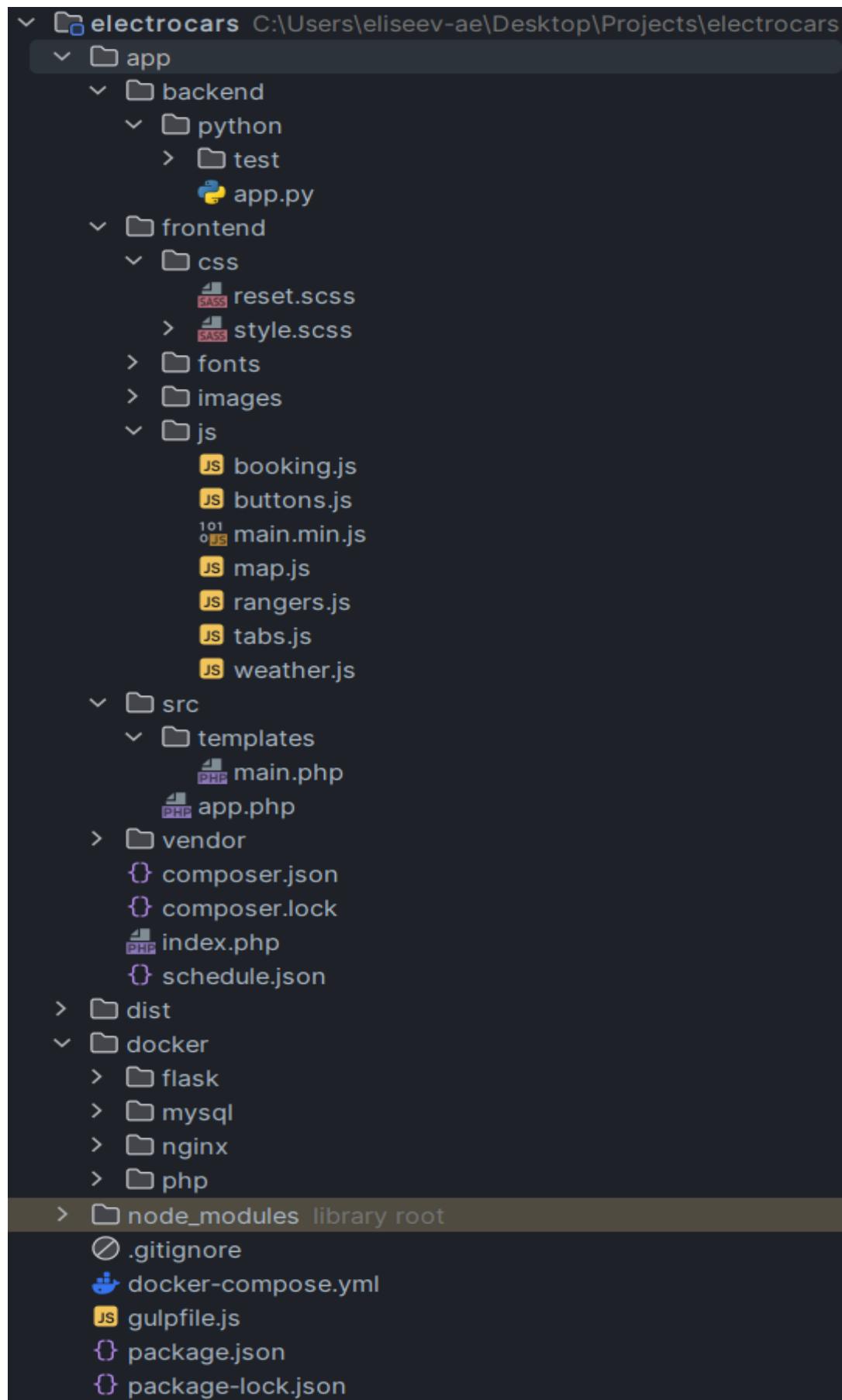
34. Hardingham M., Anderson J.E., Nobis C., Stark K., Vladova G. (2022). Booking Public Charging: User Preferences and Behavior towards Public Charging Infrastructure with a Reservation Option, *Electronics* 2022, 11, 2476, pp. 1–18;
35. Anam A., Farooque A., Muhammed W. A., Wasi H. B., Muhhamed R. (2018). Event-Driven Process Chain for Modeling and Verification of Business Requirements—A Systematic Literature Review, *Digital Object Identifier* 10.1109/ACCESS.2018.2791666, pp. 1–22;
36. Образовательный портал, посвящённый бизнес-моделированию, «Business Studio» [Электронный ресурс]//Режим доступа:
https://www.businessstudio.ru/wiki/docs/v4/doku.php/ru/csdesign/bpmmodeling/epc_notation (дата обращения: 06.04.2023);
37. Официальный вебсайт веб-сервера «nginx» с открытым кодом [Электронный ресурс]//Режим доступа:
<https://www.nginx.com/resources/glossary/nginx/>
(дата обращения: 06.04.2023);
38. Официальный вебсайт платформы контейнеризации приложений «Docker» [Электронный ресурс]//Режим доступа:
<https://docs.docker.com/get-started/overview/> (дата обращения: 06.04.2023);
39. Официальный вебсайт одного из крупнейших в мире электросетевых холдингов ПАО «РОССЕТИ» [Электронные ресурсы]//Режим доступа:
<https://www.rosseti.ru/company/> (дата обращения: 23.04.2023);
40. Справочник зарядной инфраструктуры ПАО «Россети Ленэнерго» [Электронный ресурс]//Режим доступа: <https://rosseti-lenenergo.ru/ev/> (дата обращения: 23.04.2023);
41. Официальный вебсайт документации библиотеки «flask» для языка программирования Python [Электронный ресурс]//Режим доступа:
<https://flask.palletsprojects.com/en/2.3.x/> (дата обращения: 30.04.2023);
42. Официальный вебсайт документации библиотеки «flask_cors» для языка программирования Python [Электронный ресурс]//Режим доступа:
<https://flask-cors.readthedocs.io/en/latest/> (дата обращения: 30.04.2023);

43. Официальный вебсайт для получения API и документации библиотеки «openrouteservice» для языка программирования Python [Электронный ресурс]//Режим доступа: <https://openrouteservice.org/> (дата обращения: 30.04.2023);
44. Официальный вебсайт географического веб-сервиса «OpenStreetMap», созданный сообществом картографов [Электронный ресурс]//Режим доступа: <https://www.openstreetmap.org/about> (дата обращения: 30.04.2023);
45. Официальный вебсайт документации стандартной библиотеки «math» для языка программирования Python [Электронный ресурс]//Режим доступа: <https://docs.Python.org/3/library/math.html> (дата обращения: 30.04.2023);
46. Официальный вебсайт документации стандартной библиотеки «random» для языка программирования Python [Электронный ресурс]//Режим доступа: <https://docs.Python.org/3/library/random.html> (дата обращения: 30.04.2023);
47. Официальный вебсайт документации стандартной библиотеки «string» для языка программирования Python [Электронный ресурс]//Режим доступа: <https://docs.Python.org/3/library/string.html> (дата обращения: 30.04.2023);
48. Jason, Melvin Siever, Alvin Valentino, Kristien Margi Suryaningrum, Rezki Yunanda (2023). Dijkstra's algorithm to find the nearest vaccine location, Procedia Computer Science (216), pp. 5-12;
49. Hak-Seon Kim (2016). A Study of Financial Performance using DuPont Analysis in Food Distribution Market, Culinary Science & Hospitality Research (22(6):52-60), pp. 1 – 9;
50. Раздел официального вебсайта одного из крупнейших в мире электросетевых холдингов ПАО «РОССЕТИ», посвящённый к финансовым отчётностям [Электронный ресурс]//Режим доступа: https://rossetimr.ru/invest_news/otchetnost/otchet_rsby/#tab15-2022 (дата обращения: 04.05.2023);

51. Раздел официального вебсайта сервиса «Яндекс.Карты», посвящённый тарифной сетке доступа к API [Электронный ресурс]//Режим доступа: <https://yandex.ru/dev/maps/commercial/> (дата обращения: 04.05.2023);
52. Раздел официального вебсайта делового портала с бизнес-статистикой и бизнес-аналитикой «TAdviser.ru», посвящённый распространённости электромобилей в России [Электронный ресурс]//Режим доступа: [https://www.tadviser.ru/index.php/Статья:Электромобили_\(рынок_России\)](https://www.tadviser.ru/index.php/Статья:Электромобили_(рынок_России)) (дата обращения: 04.05.2023);
53. Раздел официального вебсайта сервиса «Яндекс.Погода», посвящённый тарифной сетке доступа к API [Электронный ресурс]//Режим доступа: <https://yandex.ru/dev/weather/doc/dg/concepts/pricing.html> (дата обращения: 04.05.2023);
54. Официальный вебсайт ведущего индийского производителя аккумуляторов «Microtex» [Электронный ресурс]//Режим доступа: <https://microtexindia.com/ru/электромобили/> (дата обращения: 04.05.2023).

ПРИЛОЖЕНИЕ А

СТРУКТУРА КАТАЛОГА ФАЙЛОВ ВЕБ-СЕРВИСА



ПРИЛОЖЕНИЕ Б
app.php

```
<?php
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing;

function render_template($request)
{
    extract($request->attributes->all(), EXTR_SKIP);
    ob_start();
    include sprintf(__DIR__ . "/templates/main.php", $_route);

    return new Response(ob_get_clean());
}

function getWeatherDataXml($cache_life, $city)
{
    $weather = [];
    $cache_file = $_SERVER["DOCUMENT_ROOT"] . "/backend/weather.txt";
    $url = "http://export.yandex.ru/bar/reginfo.xml?region=" . $city . ".xml";
    if (time() - @filemtime($cache_file) >= $cache_life) {
        $ch = curl_init($url);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        $data = curl_exec($ch);
        curl_close($ch);
        file_put_contents($cache_file, $data);
        $buf = file_get_contents($url);
        if ($buf) {
            file_put_contents($cache_file, $buf);
        }
    }
    $xml = simplexml_load_file($cache_file);
    $weather["temp"] = $xml->weather->day->day_part[0]->temperature;
    if (substr($weather["temp"], 0, 1) == "+") {
        $weather["temp"] = ltrim($weather["temp"], "+");
    }
    return $weather["temp"];
}

function quer($con, $plug, $ac_dc, $from_power,
             $to_power, $from_price, $to_price) {
    $statement = $con->prepare(
        'SELECT * FROM Stations
         WHERE (plug="'.(string)$plug.'" OR plug IS NULL) AND
               (plug_type IN "'.(string)$ac_dc.' OR plug_type IS NULL) AND
               ((power>="'.(string)$from_power.'" AND power<="'.(string)$to_power.'')) OR power IS NULL) AND ((price>="'.(string)$from_price.'" AND price<="'.(string)$to_price.'"))
               OR price ISNULL)');
    $statement->execute();
    $results = $statement->fetchAll(PDO::FETCH_ASSOC);
    $json = json_encode($results, JSON_UNESCAPED_UNICODE);
    return $json;
}

function overwrite($resp)
{
    $path = $_SERVER["DOCUMENT_ROOT"] . "/schedule.json";
    file_put_contents($path, json_encode($resp));
}
```

```

$routes = new Routing\RouteCollection();
$routes->add(
    "main",
    new Routing\Route("/chargebook", [
        "_controller" => "render_template",
    ])
);

$routes->add(
    "weather",
    new Routing\Route("/weather", [
        "_controller" => function ($request) {
            $request = json_decode($request->getContent(), true);
            $weather = getWeatherDataXml(3600, (string) $request["city"]);
            $path = $_SERVER["DOCUMENT_ROOT"] . "/backend/weather.txt";
            unlink($path);
            return new Response($weather);
        },
    ])
);

$routes->add(
    "stations",
    new Routing\Route("/stations", [
        "_controller" => function ($request) {
            $request = json_decode($request->getContent(), true);
            $pdo = new PDO('mysql:host=mysql',
                           dbname=ELECTRO,
                           charset=UTF8',
                           "arsel",
                           "minidelphi"
            );
            $jsc = quer($pdo, $request["plug"], $request["ac_dc"],
                       $request["from_power"], $request["to_power"],
                       $request["from_price"], $request["to_price"]
            );
            return new Response($jsc);
        },
    ])
);

$routes->add(
    "schedule",
    new Routing\Route("/schedule", [
        "_controller" => function ($request) {
            $request = json_decode($request->getContent(), true);
            $resp = $request["resp"];
            overwrite($resp);
            $success = array('result' => 'success');
            $res = json_encode($success, True);
            return new Response($res);
        },
    ])
);

return $routes;

```

ПРИЛОЖЕНИЕ В

app.py

```
# библиотеки
from flask import Flask, request
from flask_cors import CORS
import openrouteservice
import math
import random
import string

# веб-приложение
app = Flask(__name__)
CORS(app, support_credentials=True)
@app.route("/", methods=['POST', 'GET'])

def task():
    resp = {}
    # приём JSON
    if request.method == 'POST':
        resp = request.get_json()
    params = {'start': [resp['start'].split(',')][0],
              resp['start'].split(',')[1]],
              'finish': [resp['finish'].split(',')][0],
              resp['finish'].split(',')[1]],
              'grad': float(resp['grad']),
              'acc': float(resp['acc']),
              'maxacc': float(resp['maxacc']),
              'spend': float(resp['spend'])}
    jsc = resp['jsc']

    # алгоритм Дейкстры
    class Dijkstra:
        def __init__(self, jsc, params):
            self.stations = jsc
            self.params = params
            self.api = '5b3ce3597851110' \
                      '001cf6248af0110' \
                      'f113d94b52baff8' \
                      'd35814621e4'
            self.id_start = '0'
            self.id_finish = ''

        # получение матрицы смежности из JSON-документа
        def matrix(self):
            # фильтрация рабочих ЭЗС
            nodes = [obj for obj in self.stations
                     if obj['plug_type'] != None]
            # объявление параметров
            spend_opt = self.params['spend']
            grad = self.params['grad']
            acc = self.params['acc']
            maxacc = self.params['maxacc']
            spend = round((spend_opt * 0.00056 * (grad ** 2 - \
                40 * grad + 2200)) / 100, 3)
            r = 15
            v = 45

            # преобразование координат
            def coordinates(nodes, id):
                for ind_obj, obj in enumerate(nodes):
```

```

        if obj['id'] == id:
            coord_ind = ind_obj;
        return [(float(nodes[coord_ind]['lon']),
                  float(nodes[coord_ind]['lat'])),
                  coord_ind]

# нахождение расстояния маршрута между вершинами графа
def route(api, coord1, coord2):
    client = openrouteservice.Client(key=api)
    geo = client.directions((coord1, coord2),
                             instructions=False,
                             geometry=False)
    km = geo['routes'][0]['summary']['distance'] / 1000
    ch = geo['routes'][0]['summary']['duration'] / 3600
    return [round(km, 3), round(ch, 3)]

# нахождение веса вершины графа
def cost(id, id_start, grad, spend, r,
         dist, v, t_dist, acc, maxacc,
         r_station):
    if id == id_start:
        acc_lvl = acc
    else:
        acc_lvl = maxacc
    if grad >= 20.0:
        weight = spend * r * (dist + v * t_dist) + \
                 spend * r * v * (float(random.randint(0, 5) / \
                 60) + 0.76 * math.log(100 * (1 - ((acc_lvl - \
                 spend * dist) / maxacc))) + r_station * \
                 (maxacc - (acc_lvl - spend * dist)))
    else:
        weight = spend * r * (dist + v * t_dist) + \
                 spend * r * v * (float(random.randint(0, 5) / \
                 60) + 0.76 * math.log(abs((1100 * ((acc_lvl - \
                 spend * dist) / maxacc) - 17 * grad + 750) / \
                 (17 * grad + 339))) + r_station * (maxacc - \
                 (acc_lvl - spend * dist)))
    return round(weight, 2)

# формирование списка вершин графа
coord_start = [float(params['start'][0]),
               float(params['start'][1])]
coord_finish = [float(params['finish'][0]),
                float(params['finish'][1])]
diameter = math.dist(coord_start, coord_finish) * 1.2
middle = [sum(x) / 2 for x in zip(*[coord_start, coord_finish])]
nodes_arr = []
for node in nodes:
    stat_coord = [float(node['lat']),
                  float(node['lon'])]
    stat_dist = math.dist(middle, stat_coord)
    if stat_dist <= diameter / 2:
        nodes_arr.append(node['id'])
nodes_arr.append(self.id_start)
nodes_arr.sort(key=lambda x: int(x))
self.id_finish = str(int(nodes_arr[-1]) + 1)
nodes_arr.append(self.id_finish)
# добавление старта и финиша к графу
add = {self.id_start: 'start',
       self.id_finish: 'finish'}
for attr in list(add.keys()):
    nodes.append({'id': attr,

```

```

        'lon': params[add[attr]][1],
        'lat': params[add[attr]][0],
        'price': '0.0'})
    nodes.sort(key=lambda x: int(x['id']))
    # формирование смежной матрицы весов
    table = {}
    for ind_id, id in enumerate(nodes_arr):
        if id != self.id_finish:
            table[id] = {}
            coord_1 = coordinates(nodes, id)
            coord1 = coord_1[0]
            for ind_rest, rest in enumerate(nodes_arr):
                if (id != rest) & (rest != self.id_start):
                    # расчёт расстояния
                    # расчёт продолжительности поездки
                    # расчёт стоимости 1кВт·ч на слоте ЭЗС
                    coord_2 = coordinates(nodes, rest)
                    coord2 = coord_2[0]
                    dist = route(self.api, coord1, coord2)[0]
                    t_dist = route(self.api, coord1, coord2)[1]
                    r_station = float(nodes[coord_2[1]]['price'])
                    ves = 0
                    # соединение физически достижимых вершин
                    # назначение вершинам весов
                    if (id == self.id_start) & \
                        (acc >= spend * dist):
                        ves = cost(id, self.id_start, grad, spend, r,
                                   dist, v, t_dist, acc, maxacc,
                                   r_station)
                    elif (id != self.id_start) & \
                        (maxacc >= spend * dist):
                        ves = cost(id, self.id_start, grad, spend, r,
                                   dist, v, t_dist, acc, maxacc,
                                   r_station)
                    if ves > 0:
                        table[id][rest] = ves
                    else:
                        pass
    return table

# реализация алгоритма Дейкстры
def solve(self, parameter):
    try:
        # объявление переменных
        graph = self.matrix()
        print(graph)
        start = self.id_start
        finish = self.id_finish
        route = {}
        connect_node = {}
        queue = []
        res = {}
        graph[finish] = {}
        # выполнение алгоритма Дейкстры
        for node in graph:
            route[node] = float("inf")
            connect_node[node] = None
            queue.append(node)
        # перебор вершин формируемого пути
        route[start] = 0
        while queue:
            key_min = queue[0]

```

```

        val_min = route[key_min]
        for n in range(1, len(queue)):
            if route[queue[n]] < val_min:
                key_min = queue[n]
                val_min = route[key_min]
        now = key_min
        queue.remove(now)
        # включение вершины в оптимальный путь
        for i in graph[now]:
            other = graph[now][i] + route[now]
            try:
                if route[i] > other:
                    route[i] = other
                    connect_node[i] = now
            except:
                pass
        # формирование результата алгоритма
        res['path'] = []
        res['path'].append(finish)
        while True:
            finish = connect_node[finish]
            if finish is None:
                break
            res['path'].append(finish)
        res['path'].reverse()
        # расчёт стоимости оптимального пути
        suma = 0
        opt = res['path']
        for index, node in enumerate(opt):
            if index != len(opt) - 1:
                suma += graph[node][opt[index + 1]]
        res['cost'] = suma
        # формирование массива оптимального пути
        res['path'][0] = '|-->'
        res['path'][-1] = '-->|'
        if res['path'] == ['-->|']:
            error = 5 / 0
        return res[parameter]
    except:
        res = {'path': 'impossible',
               'cost': 'impossible'}
        return res[parameter]

# генетические алгоритмы
class Genetic(Dijkstra):
    def __init__(self, jsc, params):
        super().__init__(jsc, params)

    def matrix(self):
        return super().matrix()

    def solve(self, parameter):
        try:
            graph = self.matrix()

            # список ключей:
            def key(dictionary):
                return list(dictionary.keys())

            # формирование пустой хромосомы
            parents = []
            dna = {}

```

```

for node in key(graph):
    dna[node] = 0
    dna[self.id_start] = 1
    dna[self.id_finish] = 1

# оператор мутации
def mutation(dna):
    n = len(key(dna))
    ind_gene = random.randint(1, n - 2)
    gene = key(dna)[ind_gene]
    dna[gene] = 1 - dna[gene]
    parents.append(dna)
    return {'dna': dna,
            'parents': parents}

# оператор кроссовера
def crossover(dna):
    if len(parents) < 3:
        return mutation(dna)
    else:
        n = len(parents)
        ind_parent1 = random.randint(0, n - 1)
        ind_parent2 = random.randint(0, n - 1)
        parent1 = parents[ind_parent1]
        parent2 = parents[ind_parent2]
        genes = key(dna)
        new_dna = {}
        for gene in genes:
            if (gene != self.id_start) & \
                (gene != self.id_finish):
                new_dna[gene] = parent1[gene] + \
                    parent2[gene]
            else:
                new_dna[gene] = 1
            if new_dna[gene] == 2:
                new_dna[gene] == random.randint(0, 1)
        parents.append(new_dna)
    return {'dna': new_dna,
            'parents': parents}

# выполнение процесса эволюции
best_fit = float('inf')
rep = -1
success = 0
success_rate = 10
rep_rate = 100000
while success < success_rate:
    rep += 1
    # выход из цикла
    if rep == rep_rate:
        if success == 0:
            res = {'path': 'impossible',
                   'cost': 'impossible'}
        success = success_rate
    fit = 0
    fail = 0
    rand = random.randint(0, 3)
    if rand == 0:
        dna = mutation(dna)[ 'dna' ]
    elif rand == 1:
        dna = crossover(dna)[ 'dna' ]
    elif rand == 2:

```

```

        dna = mutation(dna)[ 'dna' ]
        dna = crossover(dna)[ 'dna' ]
    else:
        dna = crossover(dna)[ 'dna' ]
        dna = mutation(dna)[ 'dna' ]
    # проверка приспособленности хромосомы
    path = [gene for gene in key(dna) \
            if dna[gene] == 1]
    for ind_node, node in enumerate(path):
        if ind_node != len(path) - 1:
            next_node = path[ind_node + 1]
            if next_node in key(graph[node]):
                fit += graph[node][next_node]
            else:
                fail += 1
                break
        if fail == 0:
            if fit < best_fit:
                best_fit = fit
                best_dna = path
                res = {'path': best_dna,
                       'cost': best_fit}
                success += 1
    # формирование массива пути
    if res['path'] != 'impossible':
        res['path'][0] = '|-->'
        res['path'][-1] = '-->|'
    return res[parameter]
except:
    res = {'path': 'impossible',
           'cost': 'impossible'}
return res[parameter]

# решение задачи
tasks = Dijkstra(jsc, params)
login = ''.join(random.choice(string.ascii_uppercase +
                               string.digits) for _ in range(8))
solution = {login: tasks.solve('path')}

# сравнение методов
"""
import datetime
def curr_time():
    date = datetime.datetime.now()
    return date
def attempt(tool, test, step, itera):
    cum_sum, cum_dur = 0, 0
    n_exp = itera
    for n in range(itera):
        if tool == 'Алгоритм Дейкстры':
            approach = Dijkstra(jsc, params)
        else:
            approach = Genetic(jsc, params)
        now = curr_time()
        price = approach.solve(test, 'cost')
        cum_dur += (curr_time() - now).total_seconds()
        if price != 'impossible':
            cum_sum += price
        else:
            n_exp -= 1
    test_res[tool]['Условная стоимость']\
        [str(step)] = round(cum_sum/n_exp, 2)
"""

```

```

    test_res[tool]['Длительность']\
        [str(step)]= round(cum_dur/n_exp, 6)
#чтение тестовой матрицы
doc = 'test_matrix.JSON'
with open(doc, encoding='utf-8', mode='r') as file:
    test = JSON.load(file)
#тестирование
id_last = list(test.keys())[-1]
test_arr = list(test.keys())[:-5]
test_arr.append(id_last)
test_res = {'Алгоритм Дейкстры':
            {'Условная стоимость': {},
             'Длительность': {}},
            'Генетические алгоритмы':
            {'Условная стоимость': {},
             'Длительность': {}}}
for exp in range(5):
    step = len(test_arr) - 5*exp
    itera = 10
    if step != 0:
        sample = test_arr[:step-1]
        sample.append(id_last)
        test = {key: test[key] for key in sample}
        attempt('Алгоритм Дейкстры',
                test, step, itera)
        attempt('Генетические алгоритмы',
                test, step, itera)
    test_res
"""
print(solution)
return solution

if __name__ == '__main__':
    app.run(debug = True)

```

ПРИЛОЖЕНИЕ Г

ФИНАНСОВАЯ ОТЧЁТНОСТЬ ПАО «РОССЕТИ»

Приложение 1

Бухгалтерский баланс				
на 31 декабря 20 22		г.		
Организация	ПАО "РОССЕТИ МОСКОВСКИЙ РЕГИОН"	Форма по ОКУД	Коды	
Идентификационный номер налогоплательщика		Дата (число, месяц, год)	0710001	
Вид экономической деятельности	Передача электроэнергии и технологическое присоединение к распределительным электросетям	по ОКПО	31 12 2022	75273098
Организационно-правовая форма/форма собственности	Публичное акционерное общество/частная собственность	по ИНН	5036065113	
Единица измерения: тыс. руб.		по ОКВЭД 2	35.12	
Местонахождение (адрес)	115114, г. Москва, 2-й Павелецкий проезд д. 3, стр. 2	по ОКПО/ОКФС	12247/16	
Бухгалтерская отчетность подлежит обязательному аудиту		по ОКЕИ	384	
		V ДА	НЕТ	
Наименование аудиторской организации/фамилия, имя, отчество (при наличии) индивидуального аудитора		ООО "ЦАРП-аудиторские услуги" (лидер коллективного участника и АО Аудиторская компания «ДЕЛОВОЙ ПРОФИЛЬ», член коллективного участника)		
Идентификационный номер налогоплательщика в аудиторской организации/индивидуального аудитора		ИНН	7709383532	
Основной государственный регистрационный номер аудиторской организации/индивидуального аудитора		ОГРН / ОГРНП	1027739707203	
Пояснения	Наименование показателя	Код строки	На 31 декабря 2022 (1)	На 31 декабря 2021 (2)
			На 31 декабря 2020 (3)	
	АКТИВ			
	I. ВНЕОБОРТОНЫЕ АКТИВЫ			
5.1.1.-5.2.2.	Нематериальные активы	1110	5 300 352	4 889 107
5.2.2.	в т.ч. незаконченные операции по приобретению нематериальных активов	1111	1 106 934	1 194 806
5.2.1.-5.2.2.	Результаты исследований и разработок	1120	111 972	94 681
5.2.2.	в т.ч. затраты по незаконченным исследованиям и разработкам	1121	93 691	21 888
	Нематериальные поисковые активы	1130		
	Материальные поисковые активы	1140		
5.3.1.-5.3.7.	Основные средства	1150	338 895 602	344 892 087
	земельные участки и объекты природопользования	1151	115 227	113 447
	здания, машины и оборудование, сооружения	1152	280 791 704	290 972 280
	другие виды основных средств	1153	4 135 750	3 912 404
5.3.7	Право пользования активом	11531	1 793 996	2 006 311
5.3.5.	незавершенное строительство	1154	51 407 054	47 156 870
5.3.6.	авансы, выданные под капитальное строительство и приобретение основных средств сырья и материалов, предназначенные для использования при создании основных средств*	1155	118 808	127 288
		1156	533 061	603 487
5.3.1.	Доходные вложения в материальные ценности	1160		
5.4.1.-5.4.3.	Финансовые вложения	1170	2 888 916	2 247 440
	инвестиции в дочерние общества	1171	2 888 916	2 247 440
	инвестиции в зависимые общества	1172		
	инвестиции в другие организации	1173		
	займы, предоставленные организациям на срок более 12 месяцев	1174		
	финансовые вложения	1175		
5.7.2.	Отложенные налоговые активы	1180	8 680 662	8 081 814
	Прочие внеоборотные активы	1190	499 719	473 084
	Итого по разделу I	1100	356 377 223	360 678 213
	II. ОБОРОТНЫЕ АКТИВЫ			
5.5.1-5.5.2.	Запасы	1210	3 538 929	2 907 614
	сырье, материалы и другие аналогичные ценности	1211	3 244 164	2 860 840
	затраты в незавершенном производстве	1212	294 765	46 774
	готовая продукция и товары для перепродажи	1213		
	товары отгруженные	1214		
	прочие запасы и затраты	1215		
	Налог на добавленную стоимость по приобретенным ценностям	1220	420 041	148 549
5.6.1.-5.6.4.	Дебиторская задолженность	1230	19 196 608	17 840 996
	Платежи по которой ожидаются более чем через 12 месяцев после отчетной даты	1231	178 313	303 738
	покупатели и заказчики	123101		
	векселя к получению	123102		
	авансы выданные	123103		
	прочая дебиторская задолженность	123104	178 313	303 738
	Платежи по которой ожидаются в течение 12 месяцев после отчетной даты	1232	19 018 295	17 537 258
	покупатели и заказчики	123201	15 477 689	13 789 521
	векселя к получению	123202		
	задолженность дочерних и зависимых обществ по дивидендам	123203		
	задолженность участников (учредителей) по взносам в уставный капитал	123204		
	авансы выданные	123205	2 424 880	2 003 258
	прочая дебиторская задолженность	123206	1 115 726	1 744 479
5.4.1-5.4.3.	Финансовые вложения (за исключением денежных эквивалентов)	1240	-	-
	займы, предоставленные организациям на срок менее 12 месяцев	1241		
	прочие краткосрочные финансовые вложения	1242		
Ф.4	Денежные средства и денежные эквиваленты	1250	15 191 396	10 416 263
	касса	1251		
	расчетные счета	1252	991 394	5 324 969
	валютные счета	1253		
	прочие денежные средства	1254	14 200 002	5 091 294
Прим. 12	Прочие оборотные активы	1260	10 535 402	7 672 967
	Итого по разделу II	1200	48 882 376	38 986 389
	БАЛАНС	1600	405 259 599	399 664 602
				369 911 204

Пояснения	Наименование показателя	Код строки	На 31 декабря 2022 (1)	На 31 декабря 2021 (2)	На 31 декабря 2020 (3)
ПАССИВ					
III. КАПИТАЛ И РЕЗЕРВЫ					
3.1.	Уставный капитал (складочный капитал, уставный фонд, вклады товарищей)	1310	24 353 546	24 353 546	24 353 546
3.1.	Капитал (до регистрации изменений)	1311			
3.1.	Собственные акции, выкупленные у акционеров	1320			
5.3.1.5.1.1.	Переоценка внеоборотных активов	1340		-	46 637 218
3.1.	Добавочный капитал (без переоценки)	1350	21 680 990	21 680 990	21 680 990
3.1.	Резервный капитал	1360	1 217 678	1 217 678	1 217 678
3.1.	Нераспределенная прибыль (непокрытый убыток) прошлых лет	1370	150 002 785	147 344 835	93 462 476
	отчетного периода	1371	138 599 273	147 344 835	93 462 476
	Итого по разделу III	1300	197 254 999	194 597 049	187 351 908
IV. ДОЛГОСРОЧНЫЕ ОБЯЗАТЕЛЬСТВА					
5.6.7-5.6.8.	Заемные средства кредиты банков, подлежащие погашению более, чем через 12 месяцев после отчетной даты	1410	25 000 000	62 880 000	80 600 000
	займы, подлежащие погашению более чем через 12 месяцев после отчетной даты	1411	20 000 000	42 880 000	42 600 000
5.7.2.	Отложенные налоговые обязательства	1420	21 565 168	24 201 495	22 799 103
5.7.1.	Оценочные обязательства	1430			
5.6.5-5.6.6.	Прочие обязательства в т.ч. расчеты по обязательствам по аренде	1450	21 143 726	18 059 714	11 384 032
	Итого по разделу IV	1400	67 708 892	105 141 209	114 783 135
V. КРАТКОСРОЧНЫЕ ОБЯЗАТЕЛЬСТВА					
5.6.7-5.6.8.	Заемные средства кредиты банков, подлежащие погашению в течение 12 месяцев после отчетной даты	1510	38 233 344	18 602 044	595 470
	займы, подлежащие погашению в течение 12 месяцев после отчетной даты	1511	22 915 244	35 244	36 410
5.6.5-5.6.6.	Кредиторская задолженность поставщики и подрядчики	1520	77 622 113	55 260 909	46 278 390
	векселя к уплате	1521	23 491 064	23 486 303	18 382 328
	задолженность по оплате труда перед персоналом	1522			
	задолженность перед государственными внебюджетными фондами	1523			
	задолженность по налогам и сборам	1524	1 768 455	393 399	315 063
	авансы полученные	1525	3 526 954	459 700	1 306 975
	задолженность участникам (учредителям) по выплате доходов	1526	43 555 462	29 719 830	25 627 107
	прочая кредиторская задолженность	1527	4 197 597	42 525	32 153
	расчеты по обязательствам по аренде	1528	479 388	609 799	614 764
Прим. 1530	Доходы будущих периодов	1530	16 710 360	10 266 035	8 698 513
5.7.1.	Оценочные обязательства	1540	7 589 020	15 655 875	11 905 927
	Прочие обязательства	1550	140 871	141 481	297 861
	Итого по разделу V	1500	140 295 708	99 926 344	67 776 161
	БАЛАНС	1700	405 259 699	399 664 602	369 911 204

Генеральный
директор

(подпись)

П.А. Синютин
(расшифровка подписи)

(подпись)

В.В. Витинский

* 21 " марта 20 23 г.



Отчет о финансовых результатах

за 12 месяцев 20 22 г.

Организация ПАО "РОССЕТИ МОСКОВСКИЙ РЕГИОН"

Идентификационный номер налогоплательщика

Вид экономической деятельности Передача электроэнергии и технологическое присоединение к распределительным электросетям

Организационно-правовая форма/форма собственности Публичное акционерное общество/частная собственность

Единица измерения: тыс. руб.

		Коды
Форма по ОКУД		0710002
Дата (число, месяц, год)		0710002
по ОКПО	31 12 22	
ИИН		75273098
по ОКВЭД 2		5036065113
по ОКОПФ/ОКФС		35.12
по ОКЕИ		12247 16
		384

Пояснение	Наименование показателя	Код	За 12 месяцев	
			2022 г. (1)	2021 г. (2)
1	2	3	4	5
	Выручка	2110	198 283 524	181 968 704
	в том числе			
	выручка от передачи электроэнергии	2111	179 421 639	168 998 978
	выручка от техприсоединения	2112	13 032 446	9 276 122
	выручка от организации функционирования и развитию ЕЭС России в части распределительного электросетевого комплекса	2113		
	выручка от передпродажи электроэнергии и мощности	2114		979 151
	доходы от участия в других организациях	2115		
	доходы от аренды	2116	279 064	341 633
	выручка от продажи прочей продукции, товаров, работ, услуг промышленного характера	2117	5 550 375	2 372 820
	выручка от продажи прочей продукции, товаров, работ, услуг непромышленного характера	2118		
2.1.	Себестоимость продаж	2120	(173 586 988)	(158 287 537)
	в том числе			
	себестоимость передачи электроэнергии	2121	(167 747 926)	(154 831 246)
	себестоимость техприсоединения	2122	(1 348 385)	(1 097 320)
	себестоимость организации функционирования и развитию ЕЭС России в части распределительного электросетевого комплекса	2123		
	себестоимость передпродажи электроэнергии и мощности	2124		(829 805)
	себестоимость участия в других организациях	2125		
	себестоимость услуг аренды	2126	(7 613)	(35 978)
	себестоимость прочей продукции, товаров, работ, услуг промышленного характера	2127	(4 483 064)	(1 493 188)
	себестоимость прочей продукции, товаров, работ, услуг непромышленного характера	2128		
	Валовая прибыль (убыток)	2100	24 696 536	23 681 167
2.1.	Коммерческие расходы	2210		(61 366)
2.1.	Управленческие расходы	2220	(110 035)	(456 931)
	Прибыль (убыток) от продаж	2200	24 586 501	23 162 870
	Доходы от участия в других организациях	2310	42 913	
	Проценты к получению	2320	1 313 266	494 258
	Проценты к уплате	2330	(3 061 149)	(3 327 555)
5.11.	Прочие доходы	2340	21 506 005	7 899 404
5.11.	Прочие расходы	2350	(30 033 521)	(13 025 555)
	Прибыль (убыток) до налогообложения	2300	14 354 015	15 203 422
2.3.	Налог на прибыль	2410	(2 624 682)	(3 526 395)
2.3.	в т.ч. текущий налог на прибыль	2411	(6 306 586)	(3 868 530)
2.3.	отложенный налог на прибыль	2412	3 681 904	342 135
2.3.	Прочее	2460	(325 821)	(35 988)
	Чистая прибыль (убыток)	2400	11 403 512	11 641 039

Пояснение	Наименование показателя	Код	За 12 месяцев	За 12 месяцев
			2022 г. (1)	2021 г. (2)
5.1.1. 5.3.1.	СПРАВОЧНО Результат от переоценки внеоборотных активов, не включаемый в чистую прибыль (убыток) периода	2510		
	Результат от прочих операций, не включаемый в чистую прибыль (убыток) периода	2520		
	Налог на прибыль от операций, результат которых не включается в чистую прибыль (убыток) периода	2530		
	Совокупный финансовый результат периода	2500	11 403 512	11 641 039
2.2.	Базовая прибыль (убыток) на акцию	2900	0.00023	0.00024
2.2.	Разводненная прибыль (убыток) на акцию	2910		

Генеральный
директор

[подпись]

(подпись)

П.А. Синютко, Москва, г. Нагорный, бухгалтер



[подпись]

B.B. Витинский

" 21 " марта 20 23 г.