Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ***ИУК «Информатика и управление»***

КАФЕДРА ***ИУК4 «Программное обеспечение ЭВМ, информационные технологии»***

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему:

Разработка картографического сервиса для управления сельским хозяйством

по дисциплине Компьютерные сети

Студент гр. ИУК4-71Б _____ (Котенко Н.А.)
(подпись) (Ф.И.О.)

Руководитель _____ (Кручинин И.И.)
(подпись) (Ф.И.О.)

Оценка руководителя _____ баллов _____
30-50 (дата)

Оценка защиты _____ баллов _____
30-50 (дата)

Оценка работы _____ баллов _____
(оценка по пятибалльной шкале)

Комиссия: _____ (Красавин Е.В.)
(подпись) (Ф.И.О.)

_____ (Белов Ю.С.)
(подпись) (Ф.И.О.)

_____ (Гагарин Ю.Е.)
(подпись) (Ф.И.О.)

Калуга, 2022

Калужский филиал
федерального государственного бюджетного образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой ИУК4
_____ (Гагарин Ю.Е.)
« 09 » сентября 2022 г.

ЗАДАНИЕ
на выполнение курсового проекта

по дисциплине Компьютерные сети

Студент Котенко Н.А., ИУК4-71Б
(фамилия, инициалы, индекс группы)

Руководитель Кручинин И.И.
(фамилия, инициалы)

График выполнения проекта: 25% к 4 нед., 50% к 7 нед., 75% к 10 нед., 100% к 14 нед.

1. Тема курсового проекта

Разработка картографического сервиса для управления сельским хозяйством

2. Техническое задание

Разработать клиент-серверное приложение для помощи в управлении сельским хозяйством в области планирования, наблюдения и контроля над процессом с помощью интерактивной карты и панели управления

3. Оформление курсового проекта

3.1. Расчетно-пояснительная записка на 40 листах формата А4.
3.2. Перечень графического материала КП (плакаты, схемы, чертежи и т.п.)

1. Демонстрационный чертеж
2. Схема клиент-серверного взаимодействия
3. ER-диаграмма

Дата выдачи задания « 09 » сентября 2022 г.

Руководитель курсового проекта _____ / Кручинин И.И.
(подпись) (Ф.И.О.)

Задание получил _____ / Котенко Н.А. / « 09 » сентября 2022 г.
(подпись) (Ф.И.О.)

Примечание:
Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

РЕФЕРАТ

Расчетно-пояснительная записка 40 с., 33 рисунка, 16 источников.

Разработка картографического сервиса для управления сельским хозяйством.

Объектом разработки является система, упрощающая наблюдение и управление над сельскохозяйственными работами организации.

Цель проекта – разработка картографического сервиса для упрощения управления сельскохозяйственными работами за счет наблюдения и ведения статистики.

Поставленные задачи решаются путем проектирования и разработки клиент-серверного взаимодействия серверного и клиентского приложений с помощью выбранных инструментов и технологий.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность темы курсового проекта обусловлена тем, что отрасль сельского хозяйства всегда являлась важнейшим источником дохода для любого государства или нации. Особенно в последнее время, когда во многих странах наступает период голода, вследствие отсутствия средств для ведения сельскохозяйственной деятельности, поэтому государства с развитой отраслью сельского хозяйства экспортируют продукцию в другие страны и этим спасают множество людей. Но всегда эффективность любой отрасли можно повысить за счет цифровизации и введения систем, помогающих агрегировать статистику и управлять процессом работы.

Поэтому актуальным является создание картографического сервиса, состоящего из компонентов, которые в совокупности будут помогать пользователям наблюдать за сельскохозяйственными работами и контролировать их.

Объектом курсовой работы является разработка клиент-серверного веб-приложения.

Предметом исследования курсовой работы является отрасль сельского хозяйства.

Целью работы является разработка картографического сервиса для управления сельским хозяйством.

Для достижения поставленной цели решаются следующие задачи:

1. Выполнить анализ предметной области;
2. Провести сравнительный анализ существующих аналогов;
3. Определить оптимальную структуру системы;
4. Осуществить выбор средств реализации программного продукта, соответствующего выбранной структуре;
5. Реализовать программные компоненты системы;
6. Осуществить тестирование компонентов;
7. Разработать сопроводительную документацию.

1. МЕТОДЫ И ИНСТРУМЕНТЫ ПРОГРАММНОЙ ИНЖЕНЕРИИ

1.1 Техническое задание

Требования к функциям, выполняемым системой

Система должна предоставлять пользователям следующие функции:

- Просмотр карты с геозонами и техникой;
- Удобное управление картой (масштабирование, поиск, переключение подложек, измерение расстояний, печать, сохранение участка карты и возможность делиться ссылкой на карту для «гостей»);
- Редактирование и добавление геозон, сельскохозяйственных культур, техники и различных дополнительных агрегатов, сотрудников, ежегодных планов, подразделений в организации и дополнительной информации для контролирования процесса;
- Просмотр всей доступной информации и статистики с помощью таблиц и графиков;
- Экспорт данных в формате CSV и геозон в формате GeoJSON;
- Аутентификация и авторизация пользователей по ролям: гость может смотреть только карту; наблюдатель может просматривать и карту, и всю информацию, но не редактировать ее; модератор имеет полный доступ к управлению сельскохозяйственными работами организации; администратор – то же самое, что и модератор, только имеет право создавать организации и регистрировать пользователей;
- Работа не только со «статическими» данными, но и с данными, непрерывно поступающими со стороны сервера или других API;
- Возможность принимать участие в управлении процессом для множества пользователей.

Требования к клиентской части системы

Клиентская часть системы должна предоставлять:

- Адаптивный и при этом удобный пользовательский интерфейс;
- Обработку всех ошибок со стороны клиента и оповещение об ошибках на сервере;
- Полную совместимость со внешним интерфейсом серверного приложения;
- Безопасность и защиту от существующих уязвимостей;
- Хорошую производительность карт при отрисовке, поиске и редактировании множества объектов.

Требования к серверной части системы

Серверная часть системы должна иметь:

- Удобный для использования внешний интерфейс в формате REST API;
- Внутреннюю структуру, построенную на основе микросервисной архитектуры;
- Кеширование данных для более быстрой обработки запросов и облегчения работы для СУБД;
- Обработку всех исключительных ситуаций и оповещение клиентов об ошибках;
- Быструю обработку запросов со стороны клиента;
- Кроссплатформенную основу для возможности размещения разработанного сервиса на сервере как под операционной системой Linux, так и под ОС Windows Server.

1.2 Анализ существующих аналогов и прототипов

В настоящее время существует ограниченное количество аналогов, предоставляющих подобный функционал. Однако данные системы часто явля-

ются узконаправленными либо имеют высокую стоимость внедрения в бизнес и обслуживания.

АгроСигнал

Единственное известное приложение, реализующее функционал наблюдения и контроля в области сельского хозяйства (Рис. 1.1).

Из функций системы можно отметить:

- Ведение оперативных планов и графиков работ;
- Мониторинг техники в режиме реального времени;
- Контроль здоровья полей и растений;
- Ведение отчетности и аналитика;
- Кадастровый учет;
- Отображение всей необходимой информации на карте в реальном времени.

Данная система имеет крайне высокую стоимость обслуживания, в том числе из-за использования множества датчиков для получения данных (метеостанции, трекеры техники и т.д.), хотя периодически можно обойтись сторонними API для отображения нужной информации, например, о погоде.

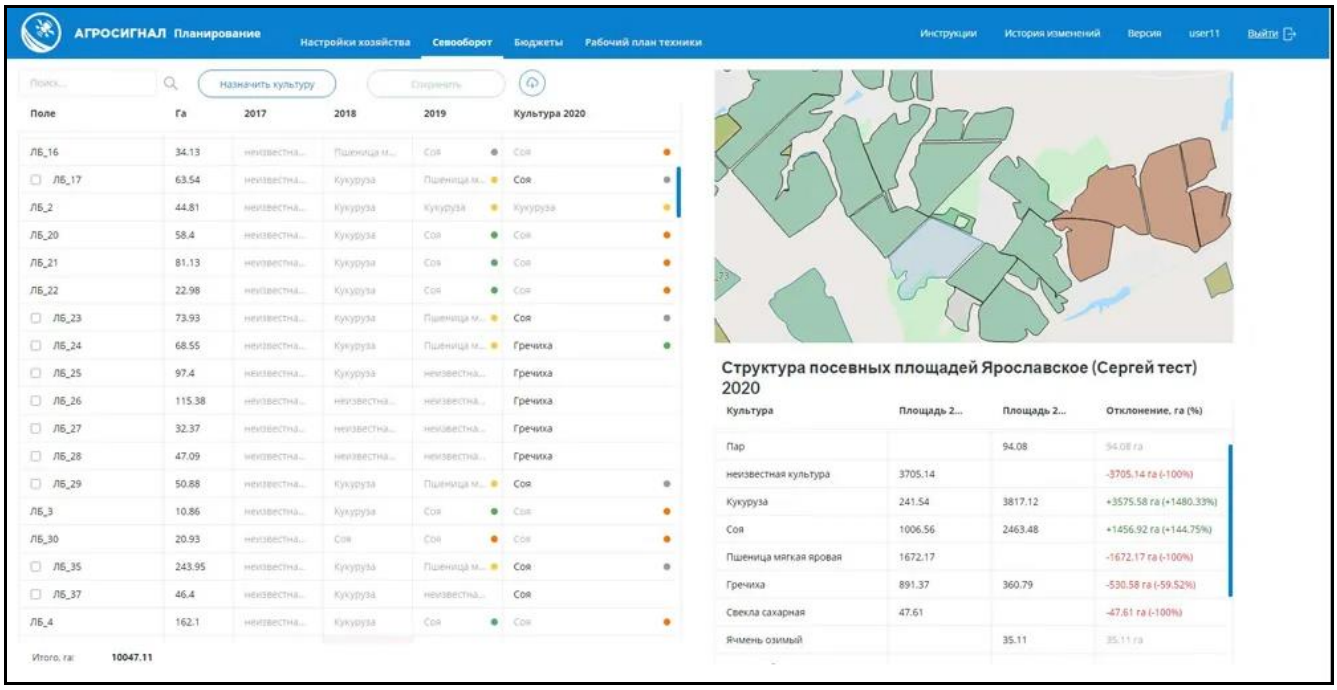

Рис. 1.1 Система «АгроСигнал»

ORBISmap

Геоинформационная платформа для визуализации, хранения и управления пространственными данными в сети Интернет (Рис. 1.2).

Из функций системы можно отметить:

- Быстрая публикация пространственных данных различного типа;
- Отображение аналитических и статистических данных на картах с выбором различных видов и алгоритмов визуализации;
- Оперативная работа с большими объемами векторных и растровых геоданных;
- Подробная карта РФ с административно-территориальным делением;
- Поддержка внешних картографических сервисов, геопривязка данных, широкие возможности интеграции.

Данная система представляет собой API и конструктор для создания интерактивных карт. Хоть на ее основе возможно создать интерактивный картографический сервис, но разработать систему для контроля над сельскохозяйственными работами уже не выйдет.

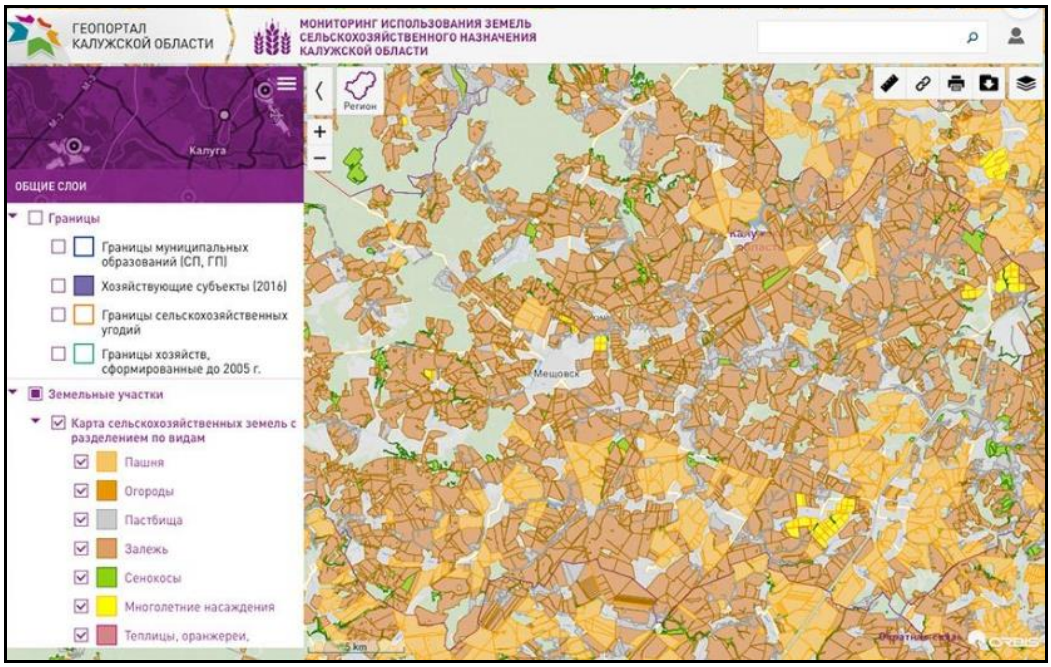

Рис. 1.2 Пример проекта, созданного с использованием ORBISmap

1.3 Обоснование выбора инструментов и платформы для разработки

Обоснование выбора картографической библиотеки

Leaflet – это JavaScript библиотека с открытым исходным кодом для создания адаптивных интерактивных карт. Из особенностей можно приметить систему плагинов, которая позволяет уменьшить размер основного пакета и добавлять необходимый функционал за счет сторонних модулей.

В настоящее время у библиотеки есть существенный недостаток в виде ее блокировки на территории РФ.

Mapbox – многофункциональная JavaScript библиотека. Она предоставляет простой в использовании набор инструментов для создания карт (работа с источниками геоданных, слоями, географической привязкой, проекциями и т.д.). Отличительной особенностью является возможность стилизации карты мира и использование своих стилей при разработке интерактивной карты.

Платформа Mapbox предоставляет текстуры, иллюстрации, пользовательские маркеры, статичные карты, геокодирование и многое другое.

Из недостатков платформы Mapbox можно отметить условно-бесплатную основу: ограниченное обращение к API Mapbox и излишняя высокоуровневость предоставляемого API.

OpenLayers – высокопроизводительная JavaScript библиотека с открытым исходным кодом, предназначенный для создания интерактивных карт при помощи различных сервисов. По сравнению с вышеуказанными аналогами имеет низкоуровневый API, предоставляющий разработчику больше контроля над инструментарием (например, встроенные средства для обработки геоданных с помощью WebGL). Одной из особенностей библиотеки OpenLayers является использование проекционной системы координат, где используются метры, в отличие от географической системы координат, основанной на долготе и широте.

Документация библиотеки выполнена на высоком уровне: имеется более двухсот примеров работы с геометрией, разными форматами геоданных и т.д.

OpenLayers подходит для рендеринга векторных данных из GeoJSON, TopoJSON, KML, GML и других географических форматов данных.

Вместе с JavaScript библиотекой turf.js для работы с геометрией в формате GeoJSON OpenLayers является отличной заменой для MapboxGL js, предоставляющей больше низкоуровневого функционала и свободы в работе с геометрией и слоями геоданных.

Обоснование выбора формата хранения геоданных

Shapefiles – один из самых распространенных форматов хранения векторных данных в геоинформационных системах, разработанный компанией Esri. Отличительной особенностью данного формата является отделение геометрической информации от атрибутивной за счет использования файлов расширения .shp, .dbf и .shx (геометрическая, атрибутивная информации и связь между ними соответственно).

KML – язык разметки геоданных на основе XML, разработанный корпорацией Google. Отличительными особенностями данного формата являются возможности представления трехмерных геоданных и общая простота восприятия информации, приведенной в формат, основанном на XML.

CSV – распространенный текстовый формат. Данные представляются в виде столбцов, разделенных запятыми. Формат CSV используется не только в сфере веб-картографии, но и в других областях, где может потребоваться хранение информации. Поэтому отличительной особенностью данного формата от вышеперечисленных является свобода определения структуры данных (название и порядок колонок).

GeoJSON – текстовый формат, основанный на JSON, из-за чего очень распространен в разработке интерактивных карт с помощью картографических API, использующих язык программирования JavaScript. Также стоит упомянуть TopoJSON – более эффективное расширение формата GeoJSON, избавляющее его от лишней избыточности при описании геометрии объектов. Одним из преимуществ данного формата является его поддержка расширением для СУБД

PostgreSQL под названием PostGIS, что позволяет удобно хранить геоданные в базе данных и с помощью SQL запросов осуществлять обработку данных.

Из-за удобства использования JSON файлов в JavaScript и их парсинга, а также приемлемой удобочитаемости для пользователя, было решено выбрать GeoJSON как основной формат для представления геоданных в разрабатываемой системе.

Обоснование выбора СУБД

Разрабатываемая система предполагает хранение и обработку данных, следовательно, необходимо средство для эффективного и удобного выполнения этих задач. Наилучшим образом для этого подходят базы данных и системы управления базами данных.

СУБД PostgreSQL - Свободно распространяемая СУБД, относится к объектно-реляционному типу. PostgreSQL отличается тем, что предоставляет объектно-ориентированный функционал.

Одной из особенностей данной СУБД является поддержка расширения PostGIS для работы с геоданными, что особенно важно в рамках разрабатываемого картографического сервиса.

Обоснование выбора языков программирования и технологий

Для серверной части

C# - объектно-ориентированный язык программирования для платформы .NET. Язык основан на строгой компонентной архитектуре и реализует передовые механизмы обеспечения безопасности кода.

C# относится к семье языков с C-подобным синтаксисом. Поддерживает черты объектно-ориентированного программирования. Является надежным и устойчивым языком, за счет использования «сборки мусора», безопасности типов и обработки исключений.

ASP.NET Core является кроссплатформенной, высокопроизводительной средой с открытым исходным кодом для создания современных облачных приложений, подключенных к Интернету.

ASP.NET Core предоставляет следующие преимущества:

— Единое решение для создания пользовательского веб-интерфейса и веб-API;

— Возможность разработки и запуска в ОС Windows, macOS и Linux;

— Открытый исходный код и ориентация на сообщество;

— Встроенное введение зависимостей;

— Возможность использования технологии SignalR;

— Упрощенный высокопроизводительный модульный конвейер HTTP-запросов.

Для клиентской части

Выбор языков разметки и языков программирования для разработки клиентской части веб-приложения на данный момент однозначен: HTML, CSS и JavaScript, но есть различные варианты улучшить опыт программирования клиентской части за счет использования разных популярных библиотек и расширений для данных языков.

При разработке планируется использовать препроцессор SASS/SCSS**,** который расширяет возможности CSS упрощает описание внешнего вида сайта за счет введения вложенности правил, более удобных переменных, условий, миксинов и т.д.

При разработке на JavaScript разработчику часто затруднительно «держать в уме» типы всех объектов и состояний программы, так что для введения статической типизации, обработки ошибок, связанных с преобразованиями типов, и облегчения работы для IDE применяется язык Typescript, расширяющий возможности Javascript.

React – библиотека Javascript, упрощающая создание пользовательских веб-интерфейсов за счет использования расширения JSX/TSX для синтаксиса

языка Javascript/Typescript, позволяющего объединять разметку и скрипты в единые переиспользуемые компоненты.

Выбор средства для взаимодействия с базой данных

Entity Framework Core — это простая, кроссплатформенная и расширяемая технология доступа к данным с открытым исходным кодом.

EF Core используется в качестве объектно-реляционного модуля сопоставления (ORM), который:

— Позволяет разработчикам .NET работать с базой данных с помощью объектов .NET, что абстрагирует слой приложения от конкретной используемой СУБД;

— Устраняет необходимость в большей части кода для доступа к данным, который обычно приходится писать;

— EF Core поддерживает PostgreSQL и ее расширение PostGIS с помощью пакета NetTopologySuite.

Форма взаимодействия между клиентом и сервером, компонентами системы

В качестве взаимодействия сервера и клиент был была выбрана архитектура REST, так что сервер будет представлять собой REST API, обращающийся к базе данных за информацией и отправляющий ее клиенту по HTTP протоколу. Система считается спроектированной по REST, если:

1. Система имеет явное разделение на клиент и сервер;

2. Сервер хранит какой-либо информации о клиентах. В запросе должна храниться вся необходимая информация для обработки запроса;

3. Используется кеширование данных;

4. Используется единый интерфейс между клиентом и сервером;

5. Система разделена на несколько малосвязанных между собой слоев.

Там, где необходимо непрерывное обновление данных для пользователей, используется библиотека SignalR от Microsoft для .NET Core, работающая на основе WebSockets или Long Polling (в зависимости от используемого брау-

зера) и позволяющая в режиме реального времени оповещать пользователей об изменениях данных на сервере. Данная технология отлично подходит для реализации уведомлений, беспрерывного получения информации с сервера и т.д.

В том случае если необходимо поделить серверное приложение на микросервисы, для их взаимодействия будет использоваться фреймворк YARP, позволяющий организовать API шлюз для объединения сервисов в единое целое.

Сами сервисы представляют собой отдельные несвязанные проекты, построенные на REST API и использующие технологии SignalR и MassTransit.

Схема взаимодействия компонентов системы

В процессе выбора средств клиент-серверного взаимодействия, была спроектирована предполагаемая схема (рис. 1.3), при соблюдении которой в процессе создания серверной и клиентской частей приложения ожидается эффективное использование выбранных инструментов разработки:

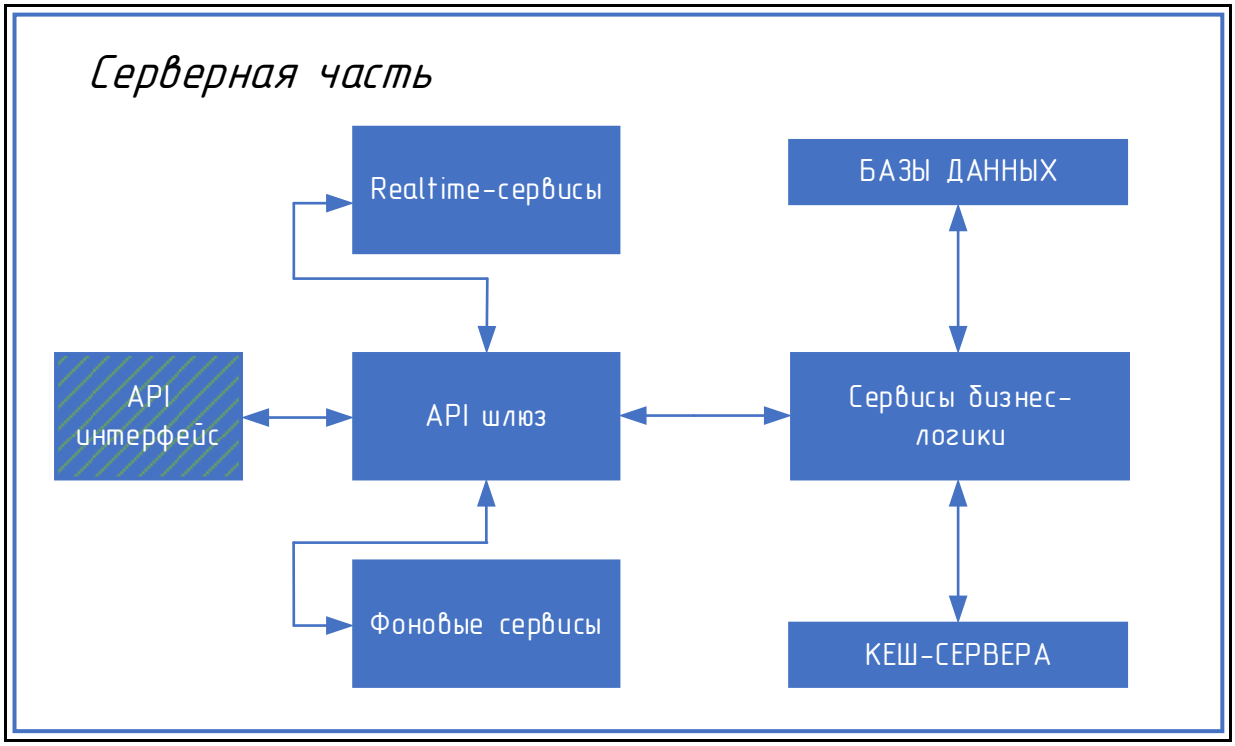

Рис. 1.3 Предполагаемая схема взаимодействия компонентов серверного приложения

2. ПРОЕКТИРОВАНИЕ КОМПОНЕНТОВ ПРОГРАММНОГО ПРОДУКТА

2.1 Разработка структуры системы

Разрабатываемая система состоит из следующих компонентов:

База данных PostgreSQL – хранит всю необходимую информацию о технике, геозонах, сотрудниках, планах и т.д. Для компактного хранения геоданных в СУБД используется расширение PostGIS.

Кеш-сервер Redis – кеширует информацию для «разгрузки» базы данных и улучшения скорости работы системы. Также кеш-сервер используется для хранения refresh-токенов, необходимых для получения access-токенов, с помощью которых пользователь аутентифицируется и авторизуется в системе.

RabbitMQ-сервер – брокер сообщений для обеспечения связи между микросервисами и слоями внутри данных сервисов. Данная технология работает по принципам AMQP – протокола для передачи сообщений между компонентами системы. Использование данной технологии позволяет уменьшить связанность микросервисов и их внутренних компонентов.

Серверная часть – представляет собой Web API на платформе ASP.NET Core набором сервисов, служащих для добавления, получения и редактирования имеющейся информации в базе данных по геозонам, сотрудникам, технике и т.д.

Клиентская часть – SPA-приложение на базе React для просмотра информации из базы данных, регистрации пользователей с использованием стандарта JWT и просмотра статистики и других данных в виде таблиц, схем и графиков.

2.2 Разработка клиент-серверной архитектуры системы

Схема взаимодействия компонентов системы

В процессе разработки и проектирования соблюдался подход, описанный в ранее приведенной схеме (рис. 1.3).

При разработке системы по этой схеме была спроектирована серверная и клиентская части и способы взаимодействие между ними и их внутренними компонентами (рис. 2.1):

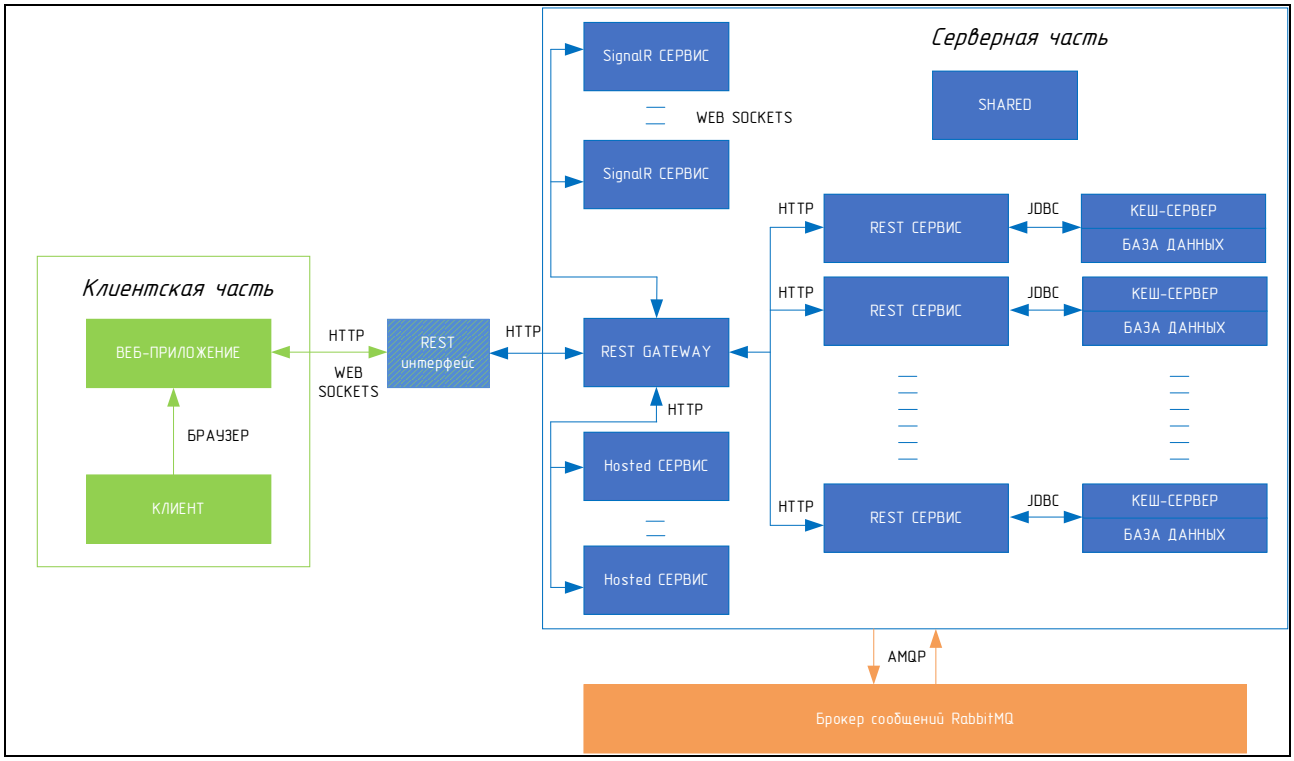

Рис. 2.1 Схема клиент-серверного взаимодействия компонентов разработанной системы

1. REST-интерфейс обеспечивает передачу данных между веб-приложением (с ним взаимодействует клиент) и сервером (система, работающая с базой данных и сторонними сервисами) согласно архитектуре REST. Клиент взаимодействует с этим интерфейсом посредством протокола HTTP и WebSockets.

2. REST-Gateway обеспечивает объединение имеющихся сервисов в кластеры с определенными маршрутами и адресами. Представляет собой про-

межуточное звено между сервером и клиентом и фактически формирует внешний REST-интерфейс.

3. Внутри серверного приложения REST-Gateway работает с микросервисами системы. К примеру, Hosted-сервисы смогут напрямую обращаться к нужному REST-сервису через REST шлюз. Данный компонент представляет собой нечто похожее на «посредника», перенаправляющего поступающие запросы в нужное место.

4. Hosted-сервисы служат для беспрерывной и контролируемой отправки запросов на клиентскую часть. В текущий момент разработки системы представляют собой имитацию работы датчиков-трекеров техники или непрерывно поступаемой информации. Располагаются по одному адресу, но с разными портами. Взаимодействуют с другими сервисами посредством протокола HTTP или AMQP в случае использования механизма событий или посредника из пакета MassTransit и RabbitMQ.

5. SignalR-сервисы служат для обновления и передачи данных в REST-интерфейс или иные внутренние сервисы в режиме реального времени. Располагаются по одному адресу, но с разными портами. Взаимодействие с другими сервисами преимущественно идет с помощью протокола AMQP.

6. REST-сервисы работают в качестве слоев бизнес-логики, которые обрабатывают, поступающие к ним запросы, работая с базами данных и распределенным кешированием. Располагаются по разным адресам. Взаимодействуют со сторонними сервисами посредством протоколов HTTP и AMQP и курирует подключение к СУБД с помощью стандарта JDBC.

7. СУБД и кеш-сервера обеспечивают механизм хранения, кеширования и обработки данных. Располагаются на отдельном адресе с разными портами.

8. Shared – отдельный конфигурационный проект, содержащий общие настройки для сервисов и абстракции, используемые для связи между объектами запросов микросервисов.

9. Все сервисы внутри серверной части представляют собой монолитные проекты, построенные на основе «чистой архитектуры».

10. Между микросервисами системы и их внутренними слоями налажена связь посредством брокера сообщений RabbitMQ, который представляет собой дополнительный уровень абстракции в общении между сервисами.

2.3 Разработка архитектуры сервисов серверной части

Для разработки Web API была выбрана «чистая», или многослойная, архитектура. Смысл данной архитектуры заключается в разделении системы на несколько слоев, которые имеют ограниченную ответственность, отвечают за строго определенные действия и слабо связаны с другими слоями.

В процессе разработки были выделены следующие слои приложения:

— Слой представления – отвечает за процесс конфигурации Web API (содержит настройки доступа к БД, конфигурацию JWT и т.д.), запуска и является отправной точкой обработки HTTP-запроса с помощью контроллеров и последующей отправки запроса в слой бизнес-логики.

— Слой абстракций – отвечает за хранение интерфейсов и сущностей базы данных. Интерфейсы используются при работе механизма внедрения зависимостей и реализации принципа инверсии зависимостей. Сущности используются в отображении объектов языка программирования на таблицы внутри СУБД с помощью ORM.

— Слой базы данных, или инфраструктуры. Данный слой отвечает за работу со внешними источниками информации, к примеру, с базой данных или файловой системой сервера. Данный слой использует абстракции из предыдущего слоя для реализации классов-репозиториев, облегчающих работу с БД. Также данный слой отвечает за конфигурацию БД, и при идеальной структуре лишь слой базы данных должен иметь зависимости от конкретной СУБД или ORM-технологии.

— Слой бизнес-логики – отвечает за обработку информации, поступающей из базы данных. Малая связанность со слоем инфраструктуры достигается за счет введения слоя абстракций. Также слой бизнес-логики содержит классы запросов и ответов, по которым идет распределение поступающих со слоя представления запросов по соответствующим обработчикам. Данный механизм работает на основе паттерна проектирования Mediator и применяется в совместности с брокером сообщений RabbitMQ.

Схема компонентов слоев и взаимодействия между ними представлена на рисунке 2.2.

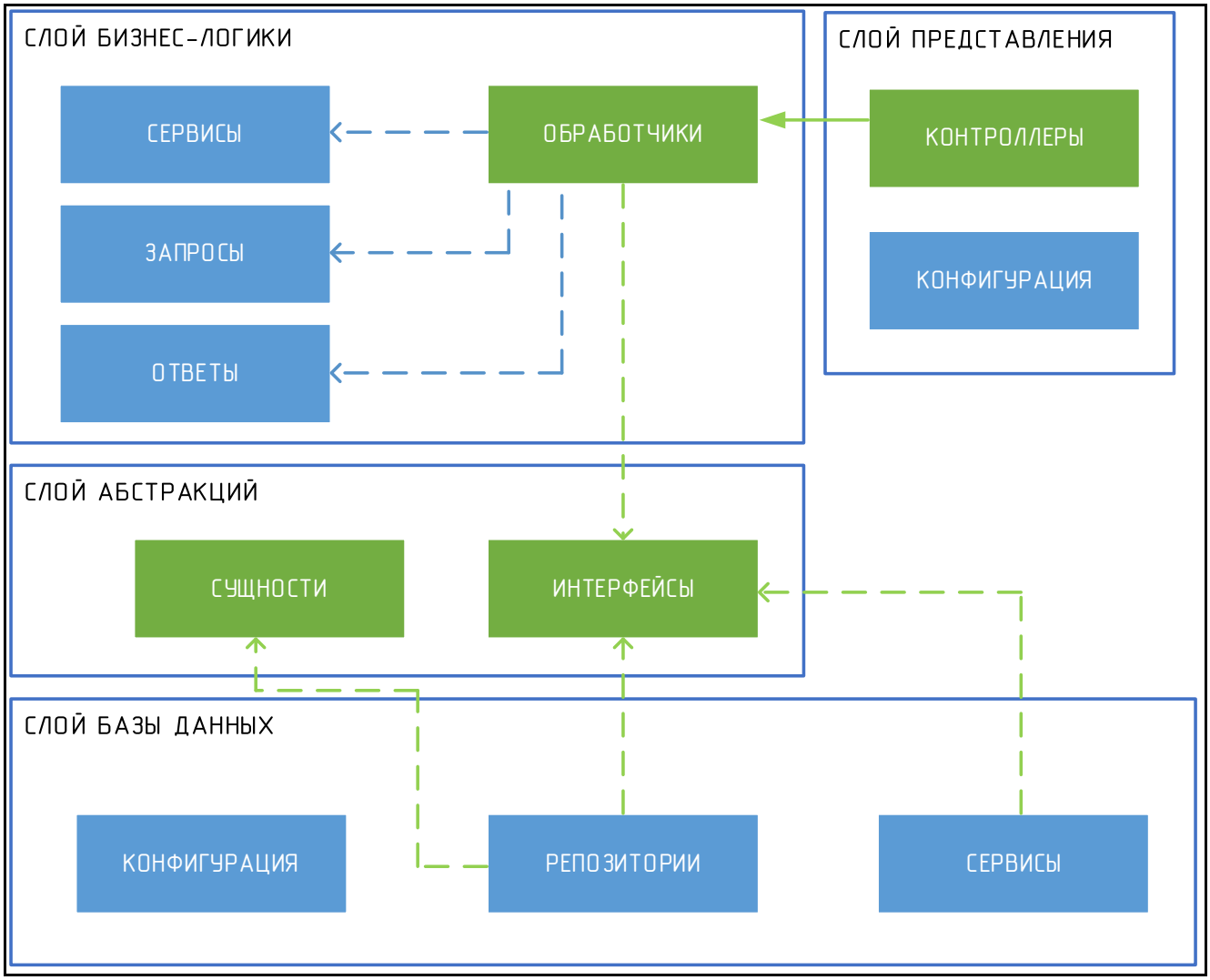

Рис. 2.2 Схема взаимодействия слоев микросервиса

2.4 Разработка структуры базы данных

База данных состоит из следующих таблиц:

— Техника – хранит в себе информацию по всей технике, к примеру, по тракторам, автомобилям, заправщикам, грузовикам и т.д. Может иметь один из 5 статусов:

1. Работает – выполняет работу на поле или выдвигается на поручение;

2. Стоянка – находится на геозоне, отвечающей за базу или склад;

3. Простой – какое-то время и не работает и не на стоянке;

4. Нет связи – непреднамеренный обрыв связи между системой и техникой;

5. Выключено – преднамеренное отключение техники от системы.

— Прицепы – хранит в себе информацию по всем прицепам, которые могут быть использованы совместно с техникой;

— Навесные агрегаты – хранит в себе информацию по всем навесным агрегатам, которые служат для высчитывания объемов проделанной работы техникой;

— Скоростные режимы – хранит в себе информацию по всем скоростным режимам для техники, на основании которых выдаются уведомления модераторам о нарушениях;

— Культуры – хранит в себе информацию по всем культурам, которые могут быть использованы при создании геозон-полей;

— Геозоны – хранит в себе информацию по всем геозонам, которые могут быть как полями, так и складами или заправками;

— Ежегодные планы – хранит в себе информацию по всем планируемым объемам выращивания культур;

— Сотрудники – хранит в себе информацию по всем сотрудникам в организации, которые могут быть модераторами системы или водителями техники;

— Должности – хранит в себе информацию по должностям сотрудников в организации;

— Пользователи – хранит в себе информацию о пользователях системы и их ролях:

1. Гость – имеет возможность просмотра карты с проводимыми сельскохозяйственными работами. Не требует регистрации в системе. Так как система спроектирована в первую очередь для организаций, которые заинтересованы в подобного рода приложении, а не на «случайных» пользователей, то предусмотрена передача ссылки «гостям» на карту организации;
2. Наблюдатель – имеет возможность просмотра карты с проводимыми сельскохозяйственными работами и всей информации в панели управления. Требует регистрации в системе;
3. Модератор – имеет контроль над всеми функциями системы внутри своей организации. Требует регистрации в системе.
4. Гость – имеет контроль над всеми функциями системы среди всех организаций. Требует регистрации в системе.

— Refresh-токены – хранит в себе информацию о долгоживущих токенах, служащих для получение короткоживущих access-токенов для JWT-аутентификации;

— Подразделения – хранит в себе информацию о подразделениях внутри организации;

— Организации – хранит в себе информацию о всех организациях, зарегистрированных в системе.

ER-диаграмма структуры базы данных представлена на рисунке 2.3.

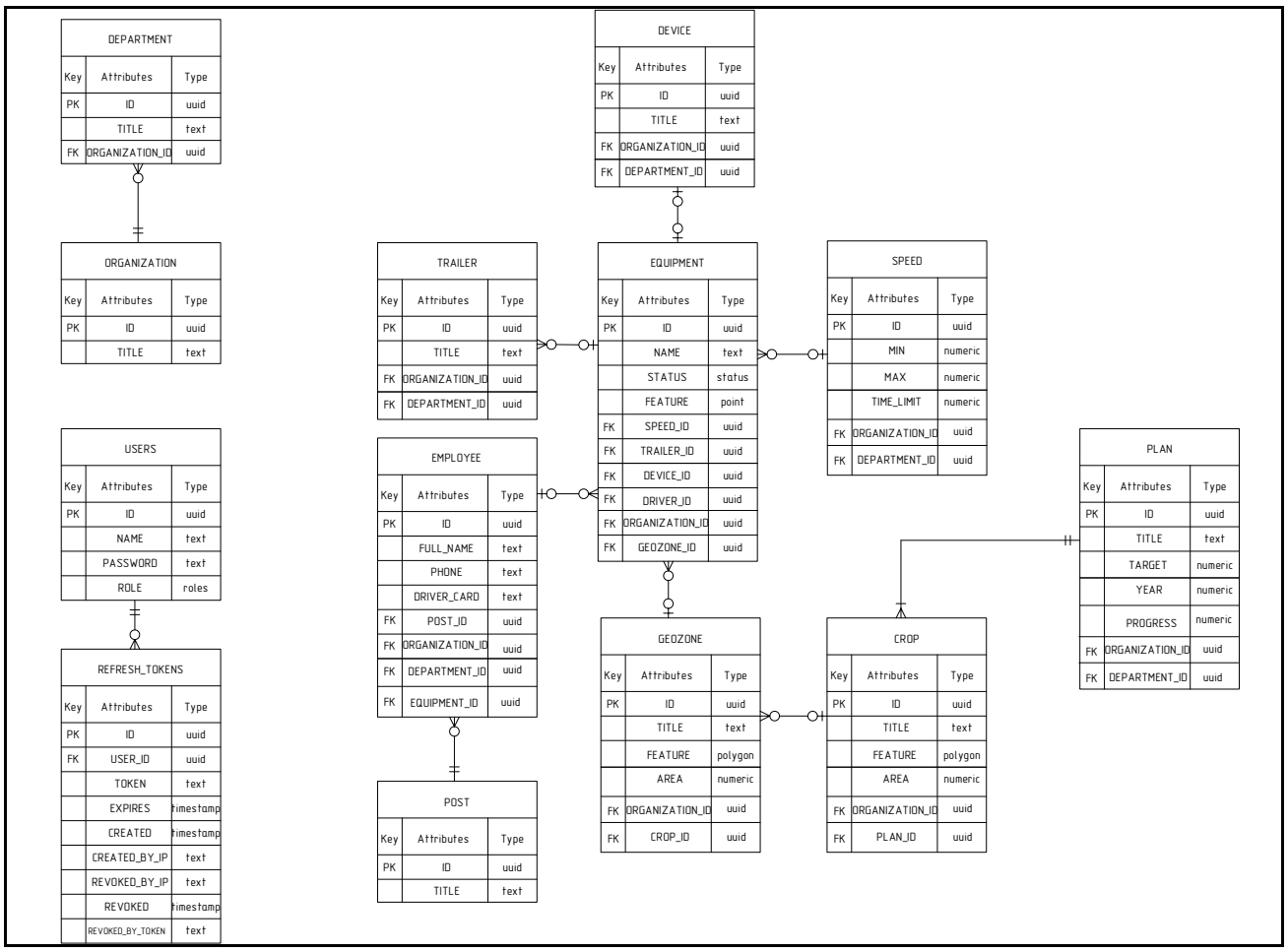

Рис. 2.3 ER-диаграмма структуры базы данных

3. ТЕСТИРОВАНИЕ И ИНТЕГРАЦИЯ КОМПОНЕНТОВ ПРОГРАММНОГО ПРОДУКТА

3.1 Тестирование системы

Тестирование системы производилось с использованием персонального компьютера с ОС Windows 11, веб-сервером Kestrel для обработки запросов, поступающих к серверу, веб-сервером Webpack dev-server для сборки и работы клиентского приложения в браузере, кеш-сервером Redis для распределенного кеширования данных на сервере, функционирующем на WSL2 с дистрибутивом Linux Ubuntu 20.04, СУБД PostgreSQL версии 15.1 и сервером брокера сообщений RabbitMQ.

Тестирование серверной части включало в себя проверку запросов к базе данных, устойчивости к возникновению нежелательных исключений, обработки статических файлов клиентской части приложения и общее тестирование разработанного API через Postman.

Клиентская часть проверялась на адаптивность интерфейса под разные устройства (мобильные и десктопные), скорость получения и отправки запросов к серверу и защиту информации при регистрации пользователей.

3.2 Руководство администратора

Для запуска и работы системы на операционной системе Windows 10/11 (Windows Server) необходимо:

1. Установить пакет .NET 6.0;
2. Установить npm и среды разработки Visual Studio и Visual Studio Code (JetBrains Rider и WebStorm) для запуска клиентского и серверного приложений;
3. Установить систему контроля версий git;

4. Установить и настроить СУБД PostgreSQL 15 с менеджером пакетов для установки расширения PostGIS;

5. Установить произвольный дистрибутив Linux на WSL2 или установить и настроить сеть в Oracle VM VirtualBox;

6. Установить кеш-сервер Redis на настроенную систему Linux;

7. Установить сервер брокера сообщений RabbitMQ. Для работы этого сервера нужно установить язык программирования Erlang;

8. Дальше необходимо пройти по следующей ссылке на репозиторий https://github.com/UnusualMessage/cartographic-application/tree/main/back и клонировать его в произвольную папку.

9. Прописать порты и адреса запущенных серверов в файле appsettings.json в сервисах серверного приложения и опционально прописать свой proxy в настройках webpack папки config клиентского веб-приложения.

10. Опционально можно переместить проект с клиентской частью в папку wwwroot проекта Entry.API. Тогда сервер будет обрабатывать всю статику клиентского приложения и для запуска потребуется запустить лишь все проекты из серверного решения. Данное решение полезно при размещении системы на виртуальном хостинге.

11. Запустить клиентское приложение командой в терминале «npm start» и собрать и запустить все проекты на сервере и тестировать систему или хостить.

3.3 Руководство пользователя

Для запуска веб-приложения необходимо авторизоваться в системе. Для этого необходимо получить у администратора логин и пароль.

Дальше перейти в любой доступный браузер и перейти на адрес аутентификации, ввести свои логин и пароль. Если регистрация прошла успешно, то на экране появится главная страница приложения, которая будет зависеть от вашей роли.

При авторизации в качестве администратора или модератора на экране появится панель управления:

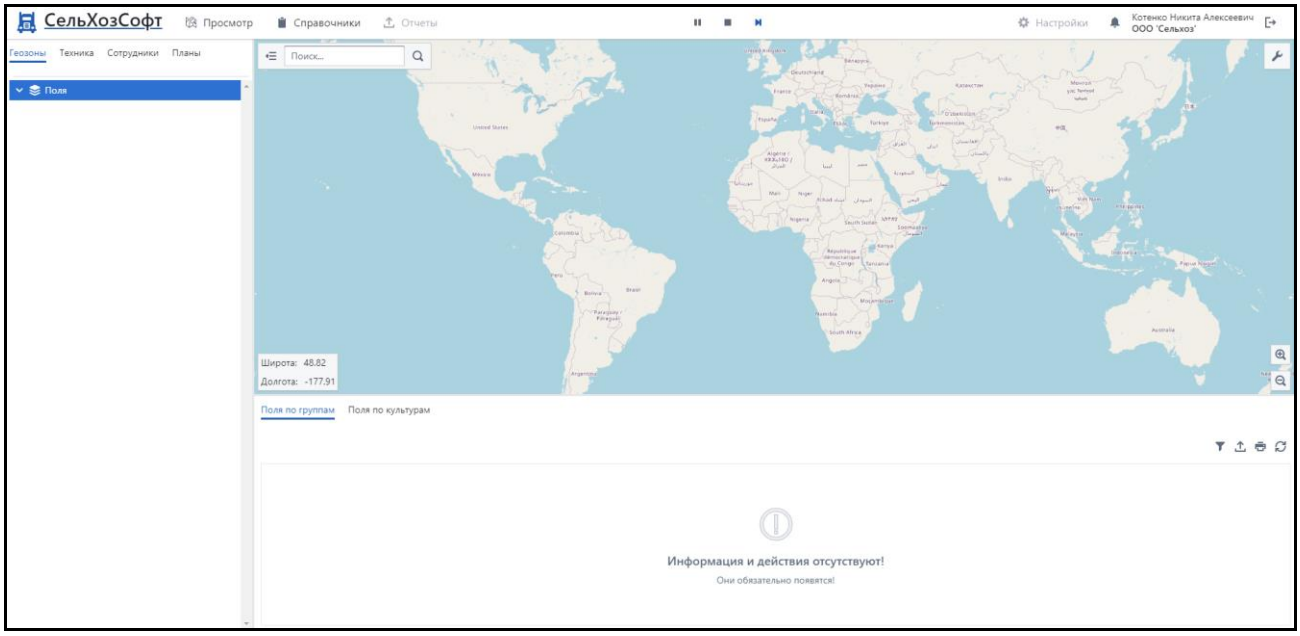

Рис. 3.1 Панель управления системой

Панель состоит из нескольких компонентов:

1. Верхняя панель (рис. 3.2) – служит для навигации между страницами панели управления (главная страница, справочники, в будущем – отчеты) и для контроля передачей данных в систему с сервера.

Рис. 3.2 Верхняя панель

2. Дерево объектов (рис. 3.3) – служит для просмотра главных объектов системы (геозоны, техника, сотрудники и планы) с помощью древовидной структуры.

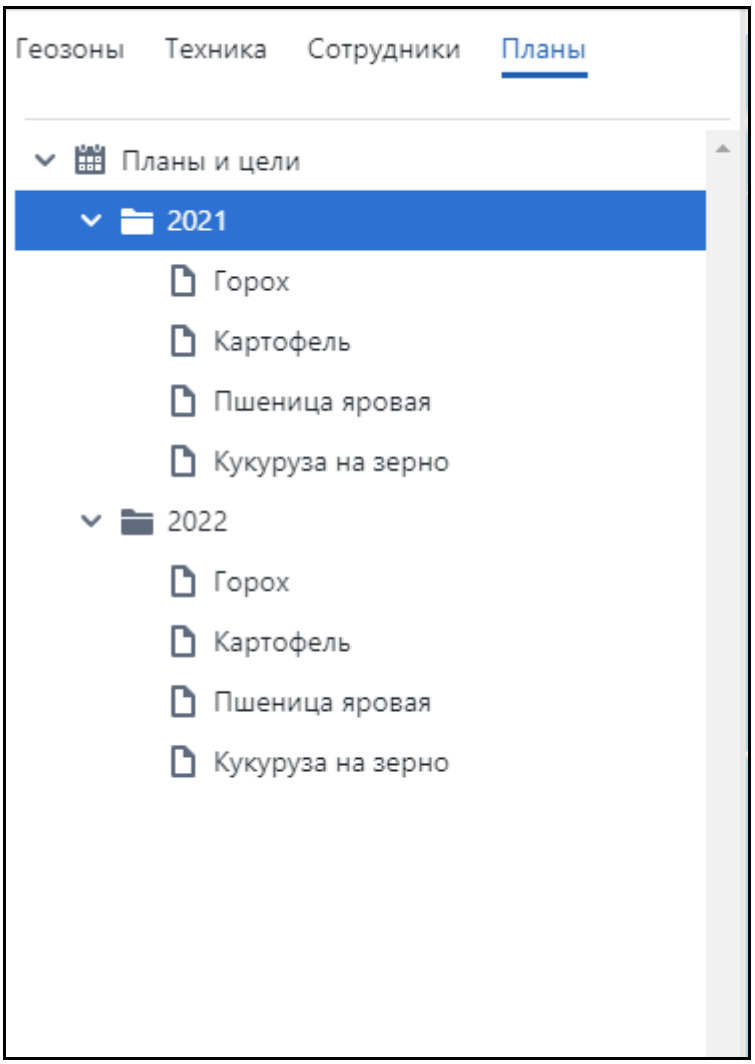

Рис. 3.3 Дерево объектов

3. Карта (рис. 3.4) – служит для просмотра и редактирования геозон, техники и ведущихся работ. Имеет множество опций для рисования, редактирования геозон (множественное копирование, трансформация, удаление, перемещение) и управления видом карты (переключение базовых слоев, печать, масштабирование, измерение расстояний и площадей и т.д.).

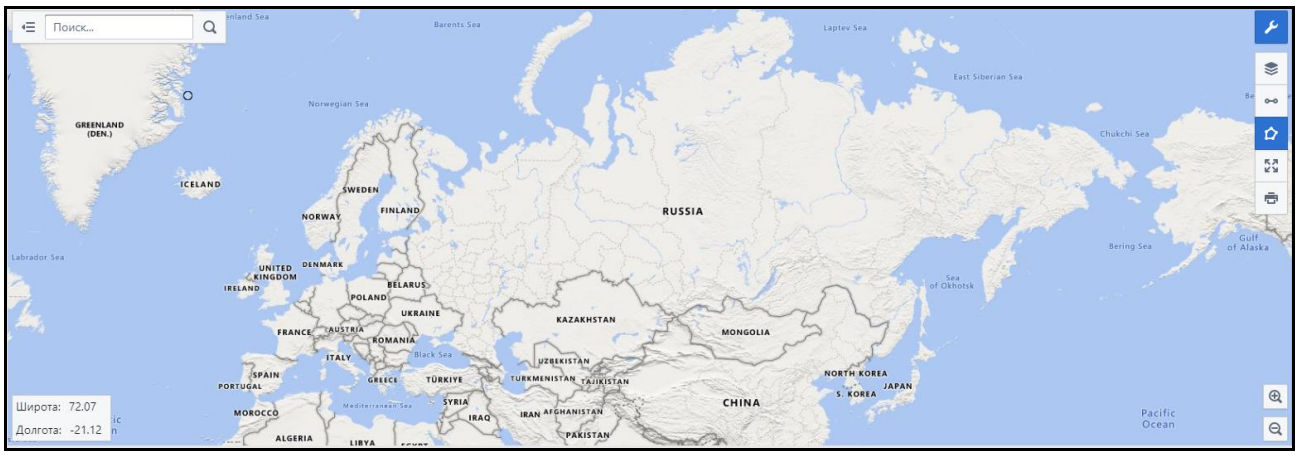

Рис. 3.4. Карта

4. Нижняя панель (рис. 3.5) – служит для просмотра информации и графиков для выбранных геозон, сотрудников, планов и т.д. В случае отсутствия информации о выбранном объекте выводится заглушка. Панель имеет несколько дополнительных опций: обновление данных с сервера, экспорт таблиц в формате CSV, печать таблиц.

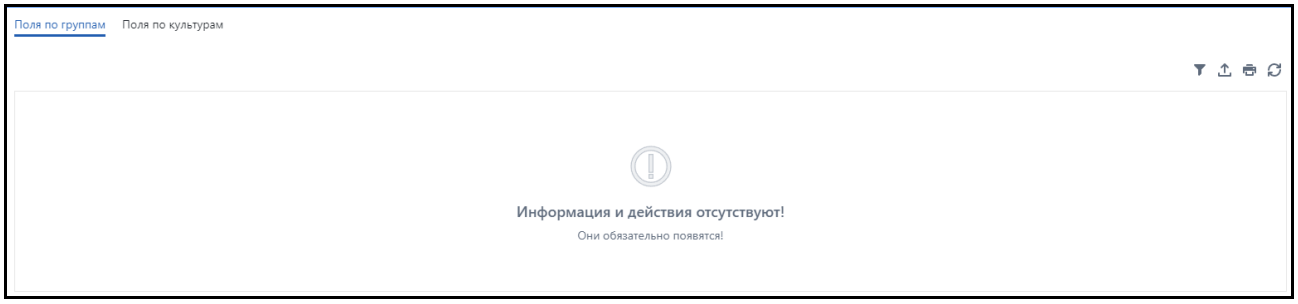

Рис. 3.5. Нижняя панель

Из дополнительных опций панели управления можно отметить возможность сворачивания дерева объектов и нижней панели для более удобного просмотра параллельно карты и другой информации (рис. 3.6):

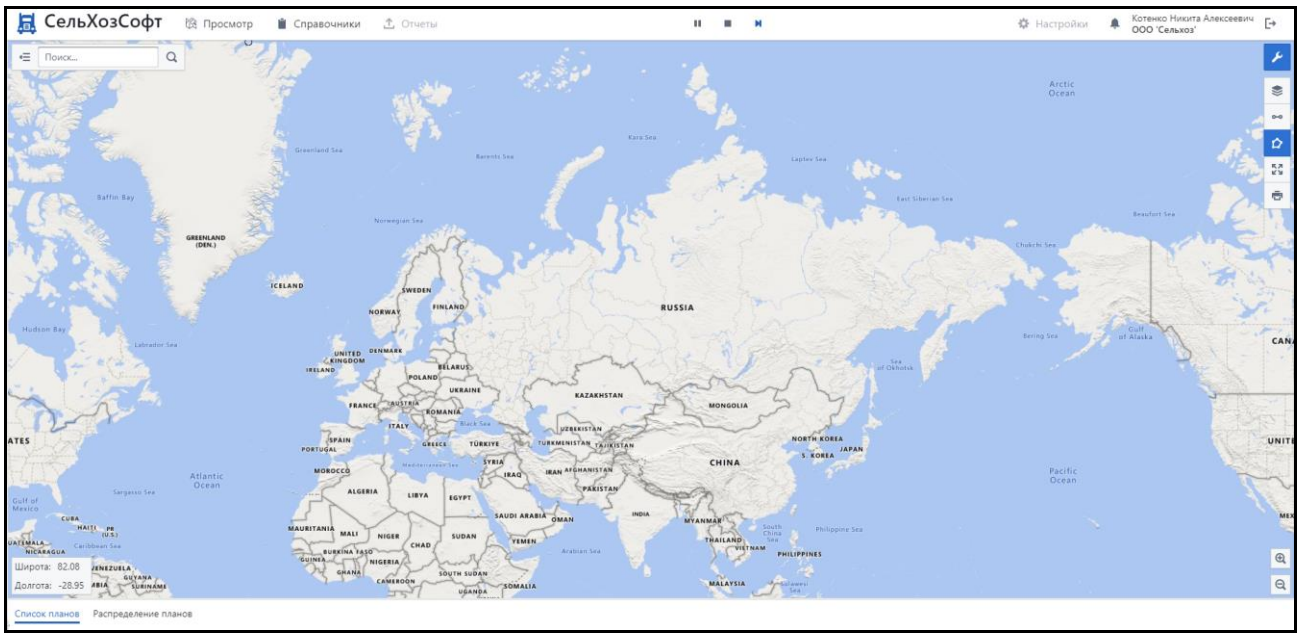

Рис. 3.6 Пример сворачивания панелей

При нажатии на кнопку «Справочники» на верхней панели откроется панель для редактирования и просмотра таблиц с данными (рис. 3.7):

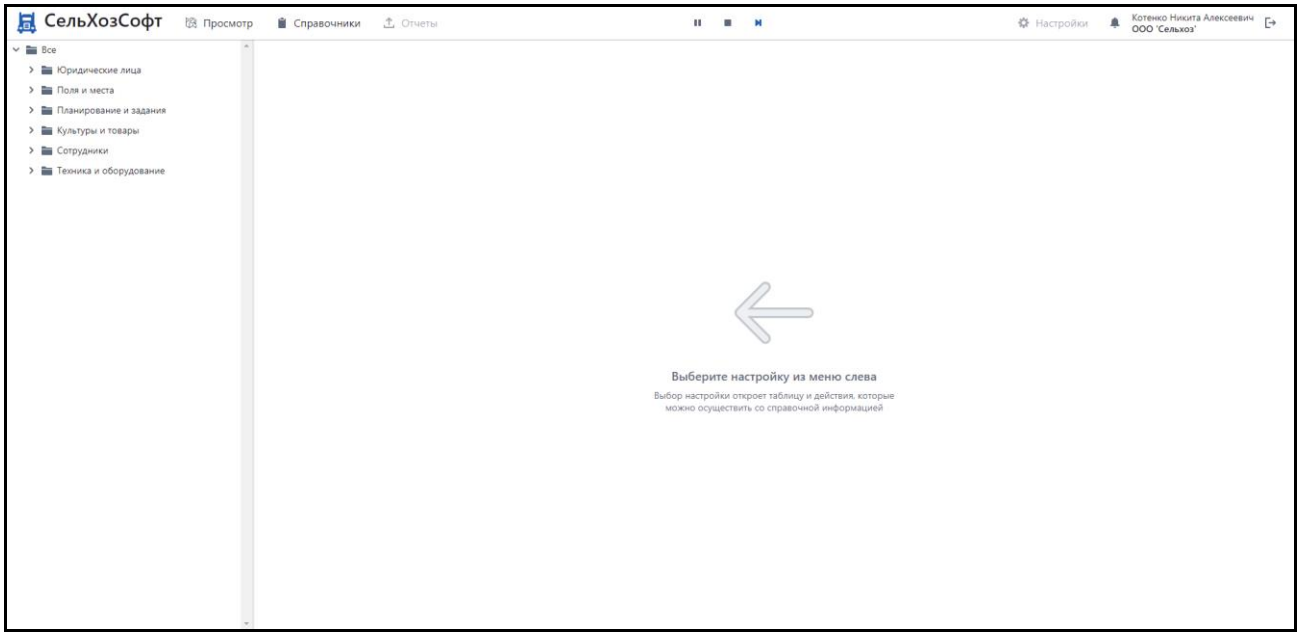

Рис. 3.7 Страница со справочниками

При выборе одной из настроек откроется таблица (рис. 3.8) с четырьмя возможными опциями (добавление, удаление, редактирование и дублирование):

Рис. 3.8 Пример таблицы для справочника

1. Чтобы добавить новую запись, необходимо нажать на кнопку «Создать» на нижней панели. Тогда откроется форма добавления новой записи (рис. 3.9). В формах поддерживается валидация полей (обязательно для заполнения, минимальное и максимальное значения, по типу данных). В случае, если заполняемая форма некорректна, то подтвердить операцию не получится, и появится предупреждение с текстом ошибки под неверным полем. При нажатии на кнопку «подтвердить» и условии, что форма корректна, в таблицу добавится новая запись (рис. 3.10).

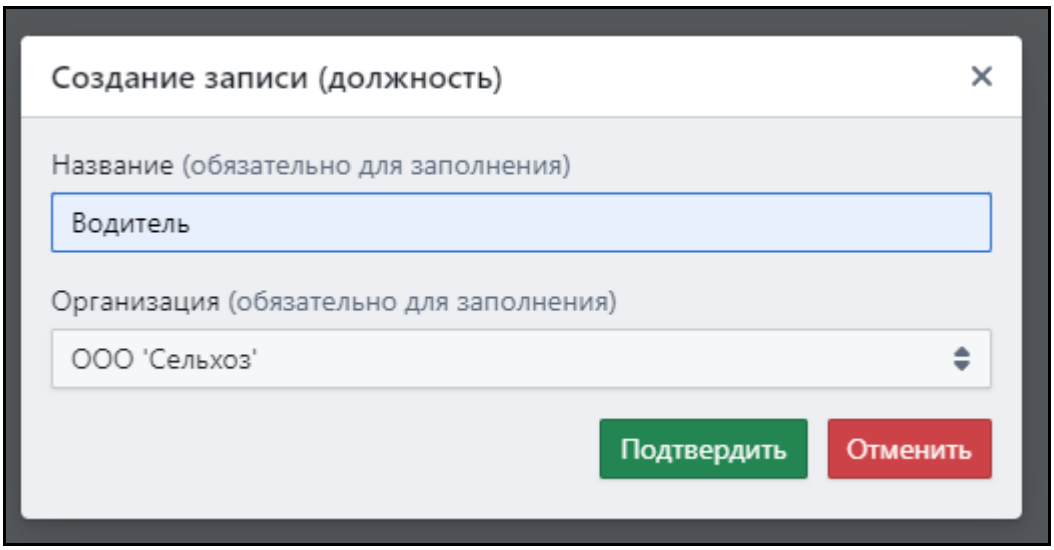

Рис. 3.9 Форма создания записи

	Номер	Название	Организация
1	0855495488482...	Водитель	ООО 'Сельхоз'

Рис. 3.10 Добавленная запись

2. Чтобы обновить запись, необходимо выбрать нужную запись в таблице нажатием кнопки по любой ячейке строки записи и нажать на кнопку «Редактировать» на нижней панели. Тогда откроется форма обновления записи (рис. 3.11). Форма работает аналогично форме добавления новой записи, за исключением того, что все поле уже заполнены в соответствие с выбранной записью. При нажатии на кнопку «подтвердить» и условии, что форма корректна, выбранная запись примет указанные в полях значения (рис. 3.12).

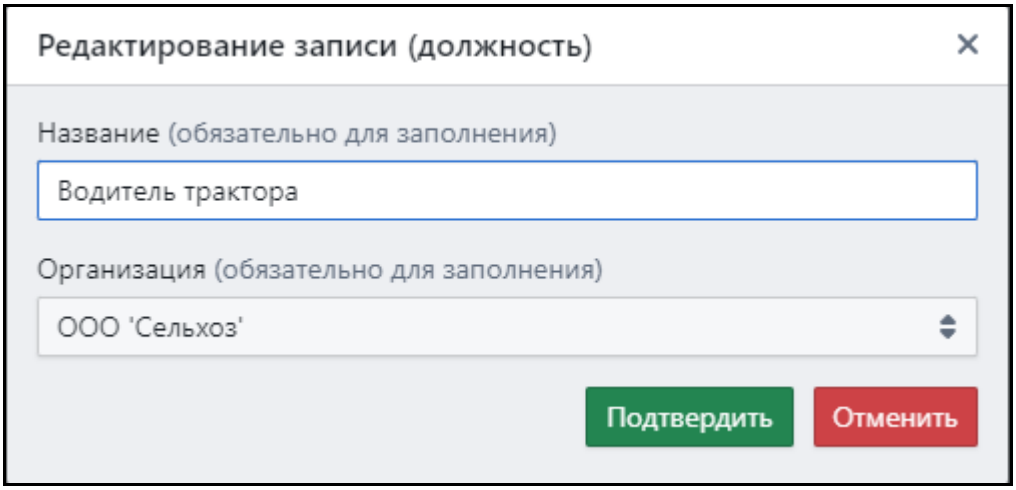

Рис. 3.11 Форма редактирования записи

	Номер	Название	Организация
1	0855495488482964086	Водитель трактора	ООО 'Сельхоз'

Рис. 3.12 Отредактированная запись

3. Чтобы дублировать запись, необходимо выбрать нужную запись в таблице нажатием кнопки по любой ячейке строки записи и нажать на кнопку «Дублировать» на нижней панели. Тогда откроется окно подтверждения

действия (рис. 3.13) и при нажатии на зеленую кнопку запись продублируется уже с новым идентификационным номером (рис. 3.14).

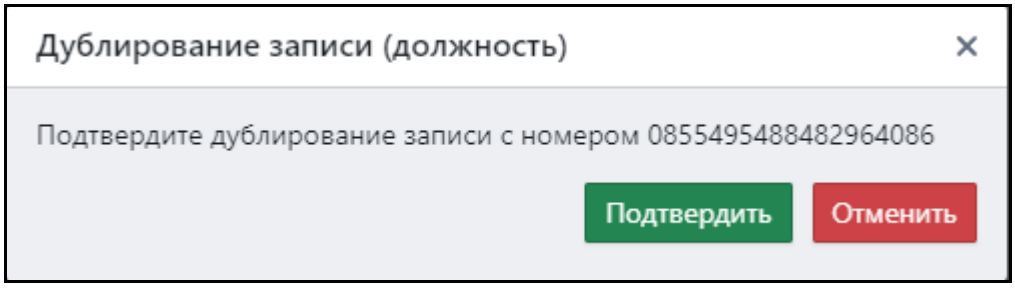

Рис. 3.13. Окно подтверждения дублирования записи

	Номер	Название	Организация
1	0855495488482964086	Водитель трактора	ООО 'Сельхоз'
2	4681990496180307672852	Водитель трактора	ООО 'Сельхоз'

Рис. 3.14. Дублированная запись

4. Чтобы удалить запись, необходимо выбрать нужную запись в таблице нажатием кнопки по любой ячейке строки записи и нажать на кнопку «Удалить» на нижней панели. Тогда откроется окно подтверждения действия (рис. 3.15) и при нажатии на зеленую кнопку запись удалится из таблицы (рис. 3.16).

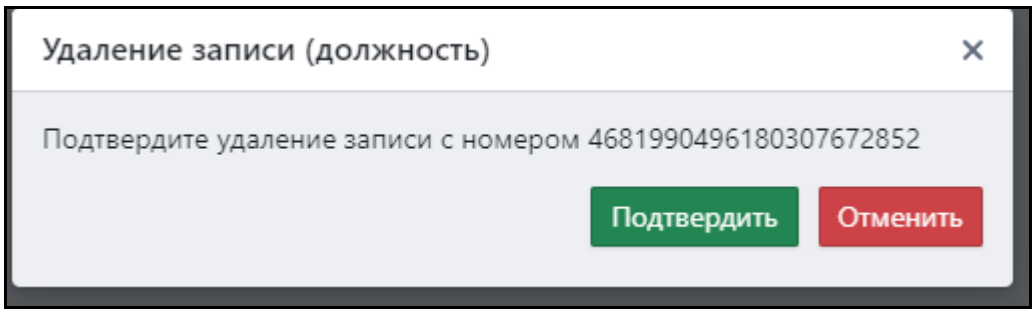

Рис. 3.15. Окно подтверждения удаления записи

	Номер	Название	Организация
1	0855495488482964086	Водитель трактора	ООО 'Сельхоз'

Рис. 3.16. Обновленная таблица без удаленной записи

Панель управления картой (рис. 3.17) имеет множество действий:

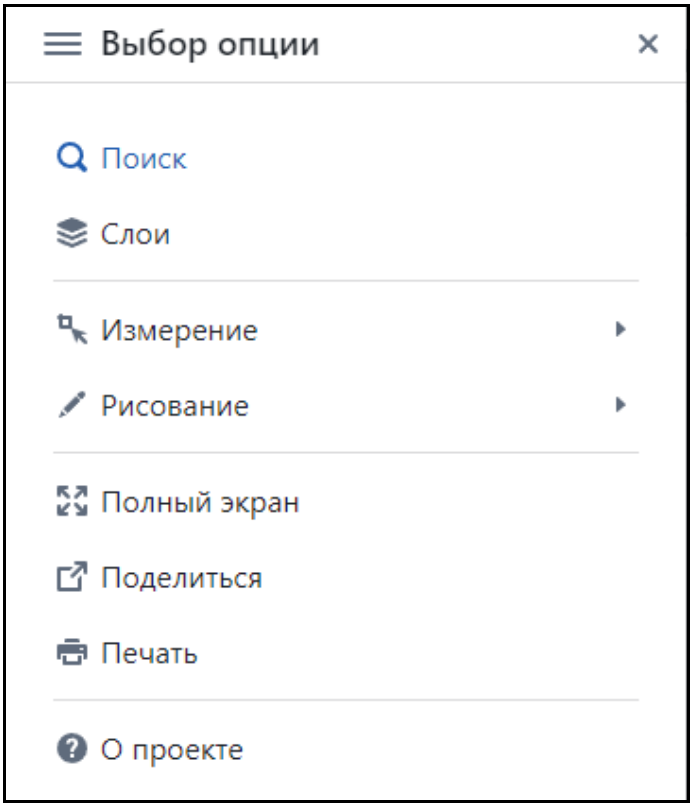

Рис. 3.17 Меню карты

1. Опция «Поиск» - позволяет использовать сторонний геокодер от сервиса Mapbox для поиска улиц, городов, стран по карте мира (рис. 3.18).

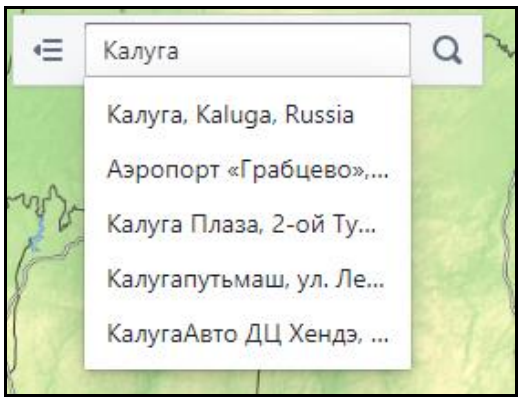

Рис. 3.18 Поиск по карте мира

2. Опция «Слои» - открывает панель управления растровыми и векторными слоями карты (рис. 3.19).

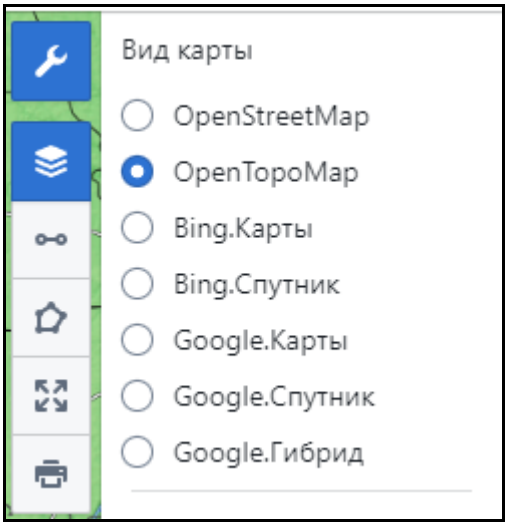

Рис. 3.19 Управление слоями карты

3. Опция «Измерение» - позволяет выбрать измерение расстояния между точками на карте (рис. 3.20) или площадь выделенной территории (рис. 3.21).

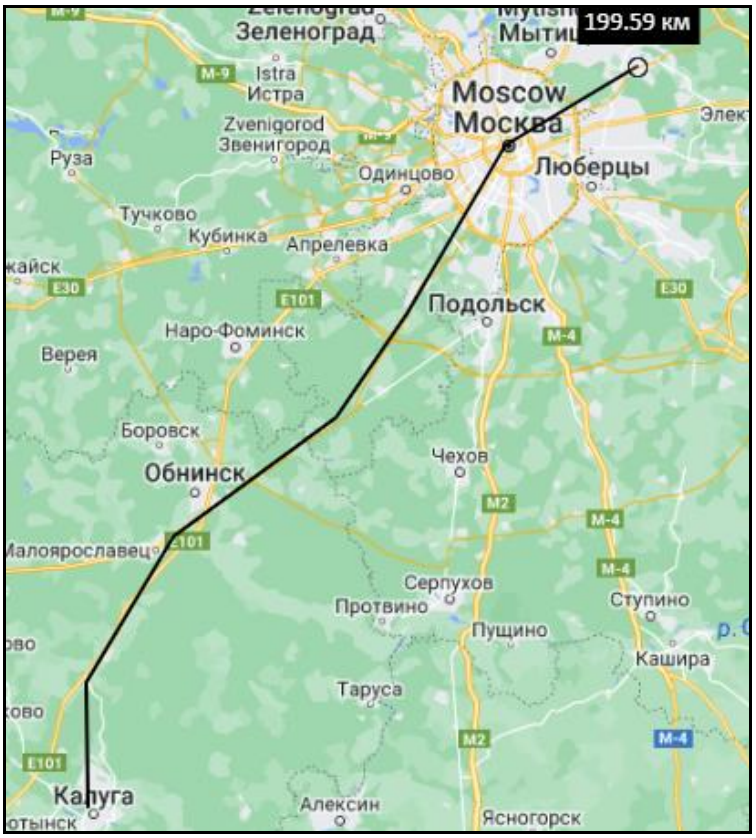

Рис. 3.20 Измерение расстояния

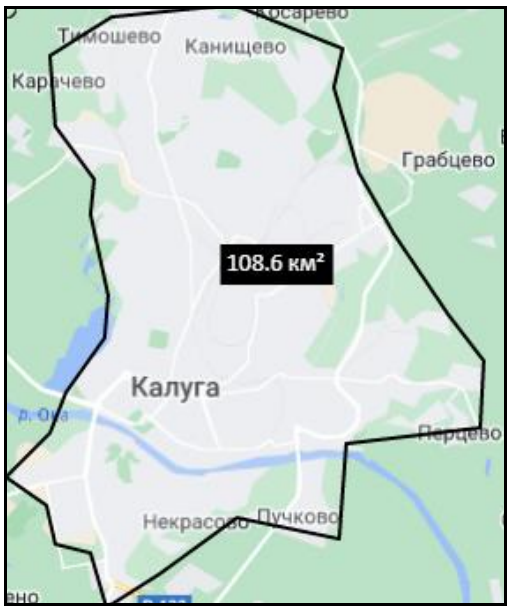

Рис. 3.21 Измерение площади

4. Опция «Рисование» - позволяет выбрать режим рисования векторных объектов на карте: многоугольник (рис. 3.22) служит для представления геозон, точка служит для представления техники, курсор позволяет перемещать и модифицировать геозоны (рис. 3.23).

Рис. 3.22 Пример отрисованных геозон

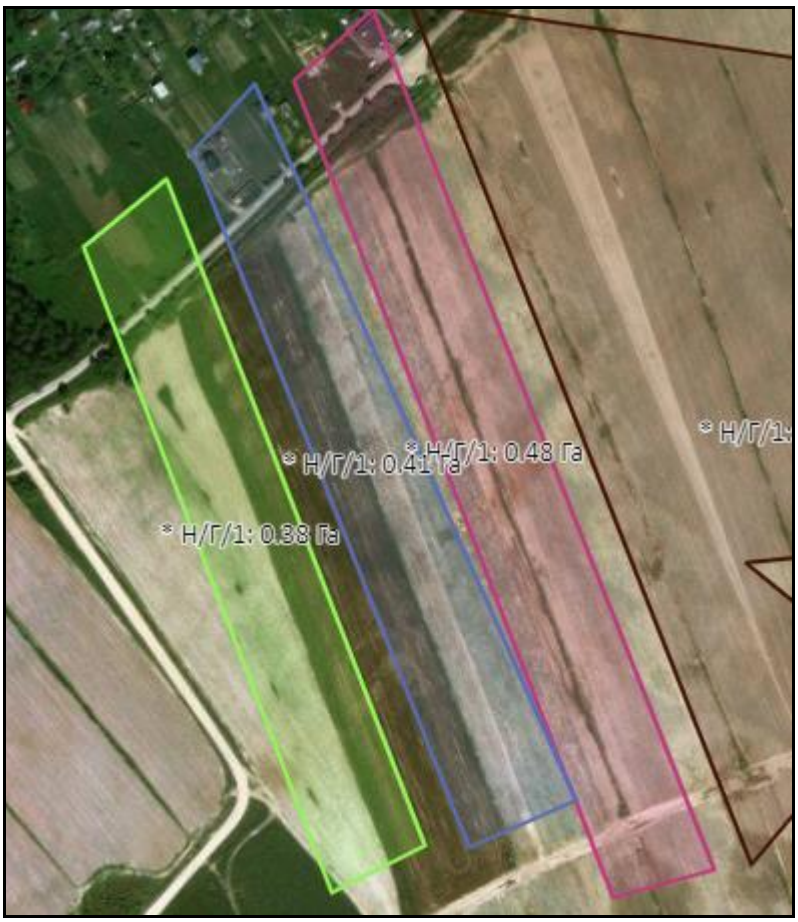

Рис. 3.23 Результат перемещения и изменение геозон

4. Опция «Полный экран» - позволяет развернуть карту в браузере в полный экран для демонстрации или работы без лишней информации.

5. Опция «Поделиться» - позволяет сгенерировать ссылку на карту для других пользователей. Если пользователь еще не регистрировался в системе, то он будет видеть только карту без возможности редактирования. Вид карты будет схож с отображением при включенном «полном экране».

6. Опция «Печать» - позволяет сгенерировать PDF-файл участка карты для последующей печати.

Также в системе имеется опция для экспорта геозоны в формате GeoJSON. Для этого необходимо нажать правой кнопкой по геозоне в дереве объектов и нажать на опцию «Экспорт». Дальше в отдельном окне появится представление геозоны в формате GeoJSON (рис. 3.24), которое можно скопировать в буфер обмена.

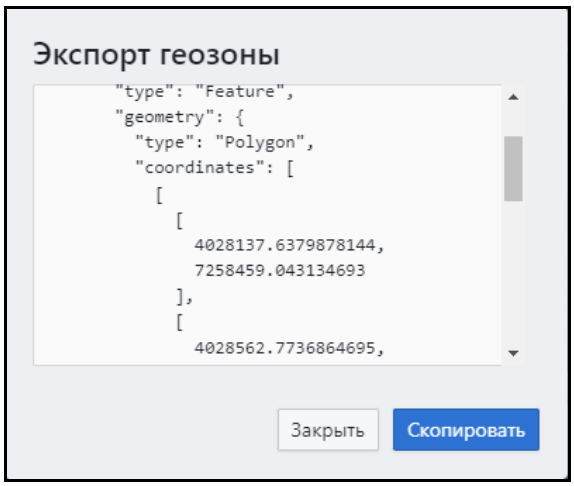

Рис. 3.24 Экспортированная геозона с проекцией координат EPSG-3857

Примеры графиков и статистики системы (рис. 3.26, 3.27, 3.28):

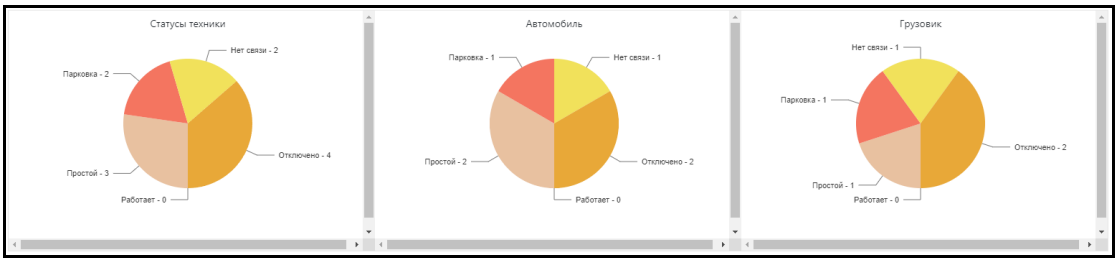

Рис. 3.25 Статистика статусов техники

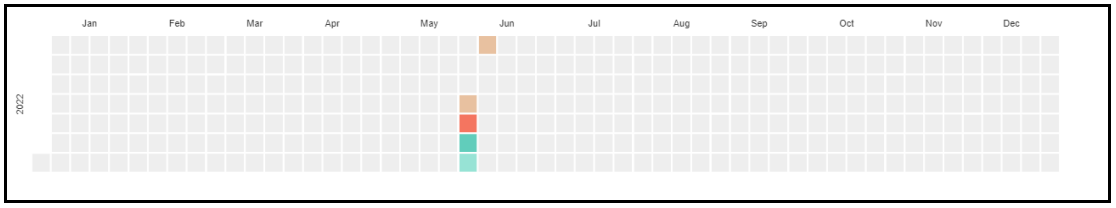

Рис. 3.26 Статистика выполненных работ по дням для техники или сотрудника

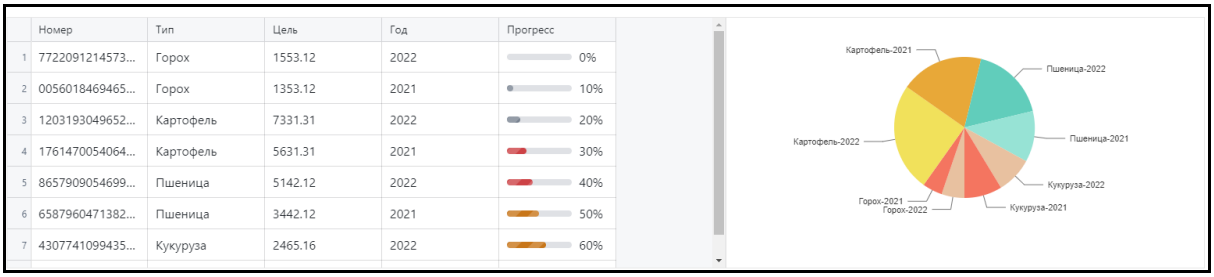

Рис. 3.27 Статистика по текущим планам

ЗАКЛЮЧЕНИЕ

В результате выполнения курсового проекта был спроектирован и разработан картографический сервис для управления сельским хозяйством.

Результат соответствует всем заранее определенным требованиям технического задания.

Разработанное приложение обладает простым и понятным пользовательским интерфейсом, что позволит использовать веб-приложение любому пользователю.

Благодаря удачно спроектированной архитектуре клиентского и серверного приложений, при необходимости можно улучшить и расширить разработанную систему путем добавления дополнительного функционала, к примеру:

— Возможность генерации пользовательских отчетов с выбором таблиц для отчета, полей, графиков;

— Расширение области применения приложения от отдельных организаций до муниципальных образований;

— Введение более глубоких настроек планирования;

— Продвинутая архивация данных с возможностью ведения статистики и механизма «подсказок» и предсказаний для более продвинутого управления сельским хозяйством.

— Возможность управления не только культурами, техникой, сотрудниками, но и областью животноводства.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Основная литература

1. Сергеев, А.Н. Основы локальных компьютерных сетей [Электронный ресурс]: учебное пособие / А.Н. Сергеев. — Санкт-Петербург : Лань, 2016. — 184 с. – Режим доступа: URL: https://e.lanbook.com/book/87591

2. Топорков, С.С. Компьютерные сети для продвинутых пользователей [Электронный ресурс]: учебное пособие / С.С. Топорков. — Москва : ДМК Пресс, 2009. — 192 с. – Режим доступа: URL: https://e.lanbook.com/book/1170

Дополнительная литература

3. Арно Лоре, Проектирование веб-API / Пер. с англ. Д. А. Беликова.– М.: ДМК Пресс, 2020.– 440 с. ISBN 978-5-97060-861-6

4. Ачилов, Р.Н. Построение защищенных корпоративных сетей [Электронный ресурс]: учебное пособие / Р.Н. Ачилов. — Москва : ДМК Пресс, 2013. — 250 с. — Режим доступа: URL: https://e.lanbook.com/book/66472

5. Ибе, О. Компьютерные сети и службы удаленного доступа [Электронный ресурс]: справочник / О. Ибе. — Москва : ДМК Пресс, 2007. — 336 с. — Режим доступа: URL: https://e.lanbook.com/book/1169

6. Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения. — СПб.: Питер, 2018. — 352 с.: ил. — (Серия «Библиотека программиста»). ISBN 978-5-4461-0772-8

7. Мартин Р. Чистый код: создание, анализ и рефакторинг. Библиотека программиста. – СПб.: Питер, 2013. – 464 с.

8. Мейер, Б. Объектно-ориентированное программирование и программная инженерия [Электронный ресурс] / Б. Мейер. — 3-е изд. — Электрон. текстовые данные. — М.: Интернет-Университет Информационных Технологий

(ИНТУИТ), Ай Пи Эр Медиа, 2019. — 285 c. — 978-5-4486-0513-0. — Режим доступа: http://www.iprbookshop.ru/79706.html

9. Моргунов, Е. П. PostgreSQL. Основы языка SQL: учеб. пособие / Е. П. Моргунов; под ред. Е. В. Рогова, П. В. Лузанова. — СПб.: БХВ-Петербург, 2018. — 336 с.: ил. ISBN 978-5-9775-4022-3 (БХВ-Петербург)

10. Поллард Б. HTTP/2 в действии / пер. с анг. П. М. Бомбаковой. – М.: ДМК Пресс, 2021. – 424 с.: ил. ISBN 978-5-97060-920-0

11. Ричардсон Крис, Микросервисы. Паттерны разработки и рефакторинга. — СПб.: Питер, 2019. — 544 с.: ил. — (Серия «Библиотека программиста»). ISBN 978-5-4461-0996-8

12. Саранча Михаил Александрович, Якимова Светлана Леонидовна Проблемы использования современного инструментария для создания интерактивных туристских веб-карт и геопорталов // Сервис в России и за рубежом. 2020. №1 (88).

13. Троелсен, Э. Язык программирования C# 7 и платформы .NET и .NET Core / Э. Троелсен, Ф. Джепикс. – 8-е изд. – Москва: Санкт-Петербург : Диалектика, 2019. – 1328 с.

14. Файн Яков, Моисеев Антон TypeScript быстро. — СПб.: Питер, 2021. — 540 с.: ил. — (Серия «Для профессионалов»). ISBN 978-5-4461-1725-3

15. Чамберс Джеймс, Пэкетт Дэвид, Тиммс Саймон, ASP.NET Core. Разработка приложений. — СПб.: Питер, 2018. — 464 с.: ил. — (Серия «Для профессионалов»). ISBN 978-5-496-03071-7

16. Шениг Г.-Ю. Ш47 PostgreSQL-11. Мастерство разработки / пер. с анг. А. А. Слинкина. – М.: ДМК Пресс, 2019. – 352 с.: ил.