**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Владимирский государственный университет**

**имени Александра Григорьевича и Николая Григорьевича Столетовых»**

**(ВлГУ)**

кафедра физики и прикладной математики

**ОТЧЕТ**

**ПО ПРЕДДИПЛОМНОЙ ПРАКТИКЕ**

#### Студента Лаврухина Максима Дмитриевича

#### Колледжа инновационных технологий и предпринимательства

Специальности 09.02.07 Информационные системы и программирование

Группы ИПсп-121

Время прохождения практики

с «20» апреля 2025 г.

по «17» мая 2025 г.

*Руководитель от ВлГУ*:

преподаватель КИТП  
(кафедра ФиПМ)

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*М.Ю. Кабанова

Владимир, 2025

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ |
|  | Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | «19» апреля 2025 г. |

З А Д А Н И Е

на преддипломную практику

студента ***Лаврухина Максима Дмитриевича***

4 курса, группы ИПсп-121 специальности 09.02.07 Информационные системы и программирование

Место прохождения практики **кафедра ФиПМ ВлГУ**

Тема ***Разработка веб приложения учёта и контроля заявок пользователей интернет-провайдера***

Последовательность прохождение практики получить задание, пройти инструктаж по технике безопасности и охране труда, выполнить все задания, оформить отчет согласно требованиям.

За время прохождения практики необходимо:

1. Проанализировать современные системы управления заявками и выбрать лучшие практики

2. Составить требования и спроектировать архитектуру приложения

3. Реализовать модули для работы с заявками

4. Провести тестирование системы

5. Подготовить отчет по результатам практики

6. Задание по стандартизации: оформить отчет по результатам практики согласно требованиям к оформлению документации

7. Задание по охране труда, технике безопасности и охране окружающей среды

изучить правила техники безопасности при работе за компьютером, вопросы охраны труда на месте прохождения практики

Отчет по практике составить к 17.05.2025

Задание выдал:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ преподаватель КИТП (кафедра ФиПМ) М.Ю. Кабанова

Задание получил:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.Д. Лаврухин

Согласовано: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ преподаватель КИТП А.В. Лоханов

19.04.2025

СОДЕРЖАНИЕ

[Введение 5](#_Toc198306940)

[1 Анализ предметной области 6](#_Toc198306941)

[1.1 Описание предметной области 6](#_Toc198306942)

[1.2 Анализ существующих решений 8](#_Toc198306943)

[1.3 Потребности пользователей и заинтересованных лиц 14](#_Toc198306944)

[2 Постановка задачи 16](#_Toc198306945)

[2.1 Требования к разрабатываемой системе 16](#_Toc198306946)

[2.1.1 Функциональные требования 16](#_Toc198306947)

[2.2 Обоснование выбора инструментальных средств 18](#_Toc198306948)

[3 Проектирование приложения 20](#_Toc198306949)

[3.1 Архитектурное проектирование 20](#_Toc198306950)

[3.2 Проектирование базы данных 21](#_Toc198306951)

[3.3 Проектирование интерфейса пользователя 22](#_Toc198306952)

[3.5 Диаграммы UML 25](#_Toc198306953)

[3.5.1 Диаграмма классов 25](#_Toc198306954)

[3.5.2 Диаграмма прецедентов 26](#_Toc198306955)

[3.5.3 Диаграммы деятельности 28](#_Toc198306956)

[4 Разработка приложения 31](#_Toc198306957)

[4.1 Реализация архитектуры 31](#_Toc198306958)

[4.2 Реализация базы данных 32](#_Toc198306959)

[4.3 Реализация пользовательского интерфейса 34](#_Toc198306960)

[5 Тестирование 39](#_Toc198306961)

[5.1 Unit-тестирование 39](#_Toc198306962)

[5.2 Результаты тестирования 39](#_Toc198306963)

[Список использованных источников 41](#_Toc198306964)

[Приложение А 42](#_Toc198306965)

# Введение

Целью прохождения преддипломной практики является создание веб-приложения учета и контроля заявок пользователей интернет-провайдера, обеспечивающего автоматизированное распределение задач между исполнителями. Для достижения поставленной цели необходимо решить следующие задачи:

* Исследовать современные системы и платформы для управления заявками, выделив их преимущества.
* Составить перечень функциональных и технических требований к приложению.
* Разработать модульную структуру приложения.
* Разработать удобный интерфейс с адаптивным дизайном, учитывающий особенности работы разных категорий пользователей.
* Реализовать ключевые подсистемы, включающие прием и обработку заявок, распределение задач, механизмы формирования отчетности.
* Провести всестороннюю проверку системы.

В связи с ростом числа цифровых услуг и увеличения объема заявок на подключение и техническое обслуживание интернет-провайдеры сталкиваются со сложностями в контроле работы кадров. Разрабатываемое веб-приложение решает эту проблему за счет автоматизации процессов обработки заявок, распределения задач между инсталляторами и встроенного модуля отчетности. Это позволит повысить скорость и качество обслуживания клиентов, снизить операционные затраты и улучшить управление ресурсами компании. Таким образом, проект актуален как для интернет-провайдеров, стремящихся оптимизировать свою работу, так и для абонентов, заинтересованных в быстром и профессиональном сервисе.

# 1 Анализ предметной области

## 1.1 Описание предметной области

Темой дипломного проекта является разработка веб-приложения учёта и контроля заявок пользователей интернет-провайдера. Приложение является онлайн-платформой, предназначенной для эффективного распределения заявок от новых клиентов на подключение цифровых услуг, а также запросов от текущих абонентов на устранение неполадок, связанных с использованием интернета, интерактивного телевидения, мобильной связи и видеонаблюдения.

Заявка может быть создана и сохранена в базе данных через личный кабинет/официальный сайт клиентом или через систему сервиса дистрибуции сотрудником группы продаж или технического блока.

В приложении предусмотрены три роли пользователей: диспетчер, инсталлятор и руководитель.

Диспетчер связывается с клиентом по номеру телефона, указанному в форме обратной связи, чтобы согласовать удобную дату и время визита инсталлятора. При необходимости уточняются детали заявки, после чего она подтверждается.

Подтвержденные заявки поступают в систему учета, где распределяются между инсталляторами, формируя их рабочий график на день вперед, при этом в целях сокращения времени, затрачиваемого на транспортировку от адреса к адресу, система выделяет для сотрудников набор запросов, поступивших в радиусе одного района.

Руководители имеют расширенный доступ к управлению персоналом и аналитике. Они могут добавлять новых сотрудников, настраивать их права доступа, а также отслеживать статистику по выполненным заявкам, оценивая эффективность работы подчиненных.

Кроме прочих, задачей сервиса является предоставление статистики о характере проблем, возникающих у пользователей.

1. Функционал для инсталляторов:
   1. Просмотр полученных заявок на подключение услуг и устранение неполадок.
   2. Фильтрация заявок по статусу, времени, месту и приоритету.
   3. Доступ к своему рабочему графику, сформированному автоматически.
   4. Отправка уведомлений о предстоящих выездах для поддержания расписания.
   5. Возможность фиксировать результаты работы: проблема устранена, требуется повторный выезд, клиент подключен, отказ клиента и т.д.
   6. Заполнение отчетов об исполнении работ после каждого визита.
   7. Возможность оставлять комментарии и замечания по каждой заявке для других сотрудников.
2. Функционал для диспетчеров:
   1. Сохранение заявок в базе данных.
   2. Доступ к номеру телефона для связи с клиентом.
   3. Возможность установки более высокого приоритета при необходимости срочно устранить проблему.
3. Функционал для руководителей:
   1. Добавление, редактирование, удаление сотрудников.
   2. Отслеживание статистики по работе подчиненных.

Таким образом, веб-приложение служит рабочим инструментом как для диспетчеров, так и для инсталляторов, которые фиксируют свои результаты работы: проблема устранена или требуется повторный выезд, абонент подключен к услуге(ам) или отказался по определенным причинам.

## 1.2 Анализ существующих решений

МТС Координатор [1]

Приложение «МТС Координатор» позволяет легко организовать эффективную работу разъездного персонала – торговых представителей, курьеров водителей, сервисных инженеров и др.

В возможности пользователя личного кабинета входит:

* Планирование рабочего дня с помощью назначения задач в автоматическом и ручном режимах,
* Передача требуемой информации в задаче (адрес, телефон, комментарий), а также её кастомизация с помощью дополнительных пользовательских полей,
* Контроль выполнения задач в режиме реального времени или с использованием настраиваемой отчетности по факту выполнения задачи или посещения объекта,
* Онлайн нотификация (E-mail, SMS, Telegram) по настраиваемым событиям работы приложения,
* Построение маршрутов и контроль их соблюдения между задачами для минимизации пробега и времени,
* Контроль местоположения и истории перемещений сотрудников,
* Контроль посещений рабочих объектов и гео-зон

Интерфейс «МТС Координатора» представлен на рисунке 1.1.

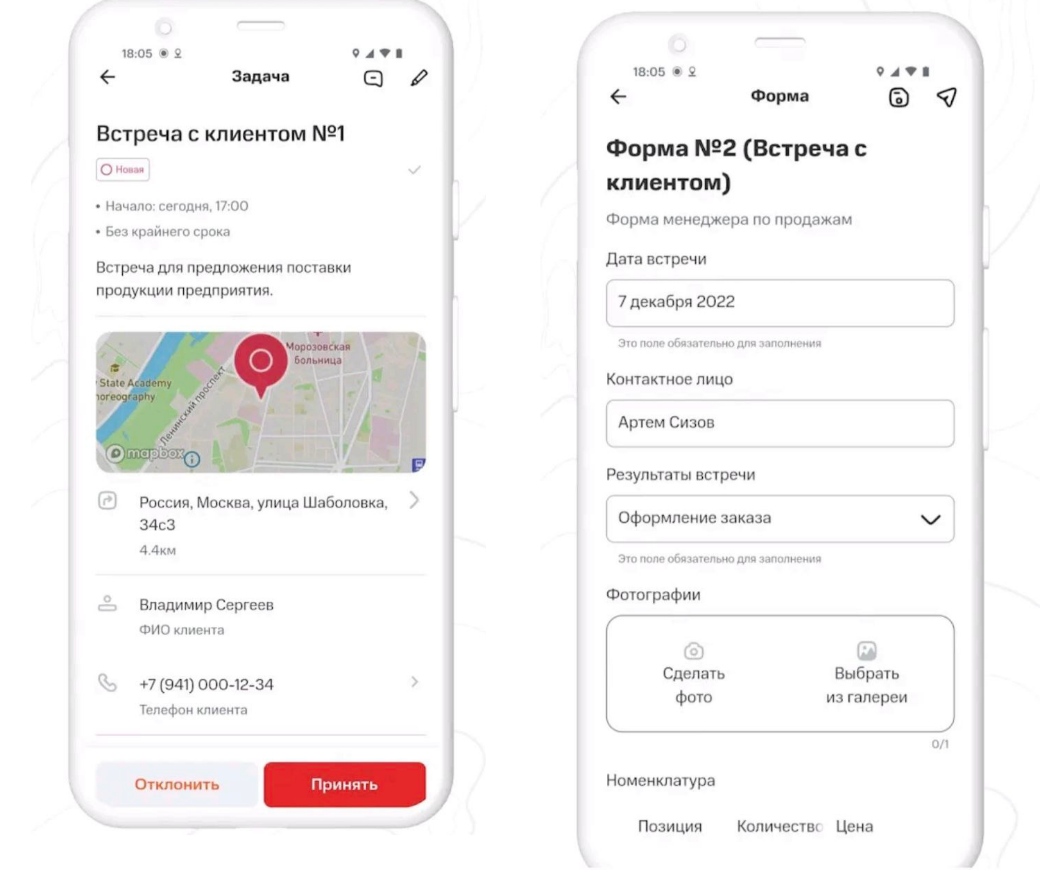


Рисунок 1.1. МТС Координатор

Мобильный агент [2]

Приложение предназначено для внутреннего использования сотрудниками компании ПАО «Ростелеком».

Среди функционала можно выделить следующее:

* Просмотр маршрутного листа и выбор домов для обхода,
* Создание заявок на подключение,
* Отслеживание статуса принятых в работу заявок,
* Просмотр статистики агента

Интерфейс «Мобильного агента Ростелеком» представлен на рисунке 1.2.

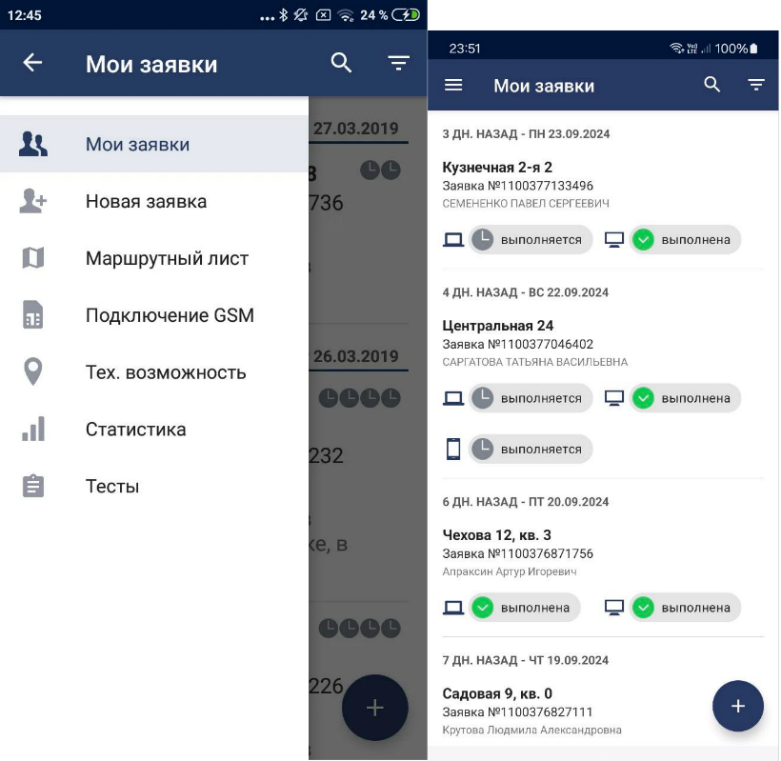


Рисунок 1.2. Мобильный агент (Ростелеком)

Дом.ру [3]

Для специалистов компании Эр-Телеком Холдинг. Приложение создано для автономной работы сотрудников, объединяет в себе инструменты АРМа, EQM и сервис геолокации для выполнения заказов

В функционал входит:

* Интеграция с АРМ (автоматизированное рабочее место)
* Доступ к системе EQM (управление качеством обслуживания),
* Сервис геолокации и навигации к месту выполнения заказа,
* Просмотр и прием заявок на подключение/ремонт,
* Ввод результатов выполненных работ,
* Доступ к технической документации и регламентам,
* Оффлайн-режим работы

Интерфейс «Дом.ру» представлен на рисунке 1.3.

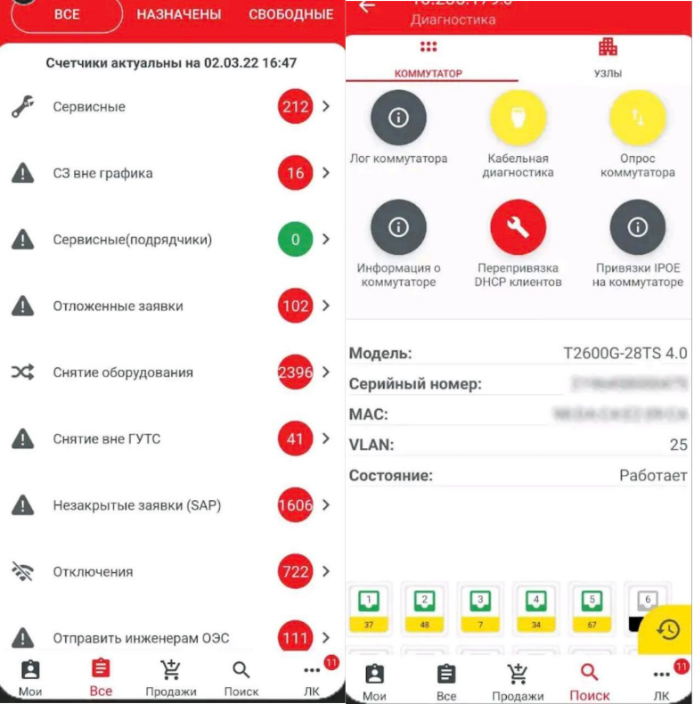


Рисунок 1.3. Рабочее место специалиста (Дом.ру)

BeeTrack [4]

Предназначен для корпоративных клиентов Билайна. BeeTrack помогает сотрудникам организовать свой рабочий день и работать с задачами, назначенными диспетчером или руководителем. Функционал:

* Просмотр списка задач,
* Установка статуса работы,
* Обмен сообщений с диспетчером,
* Заполнение отчетов по выполненной работе

Интерфейс «BeeTrack» представлен на рисунке 1.4.

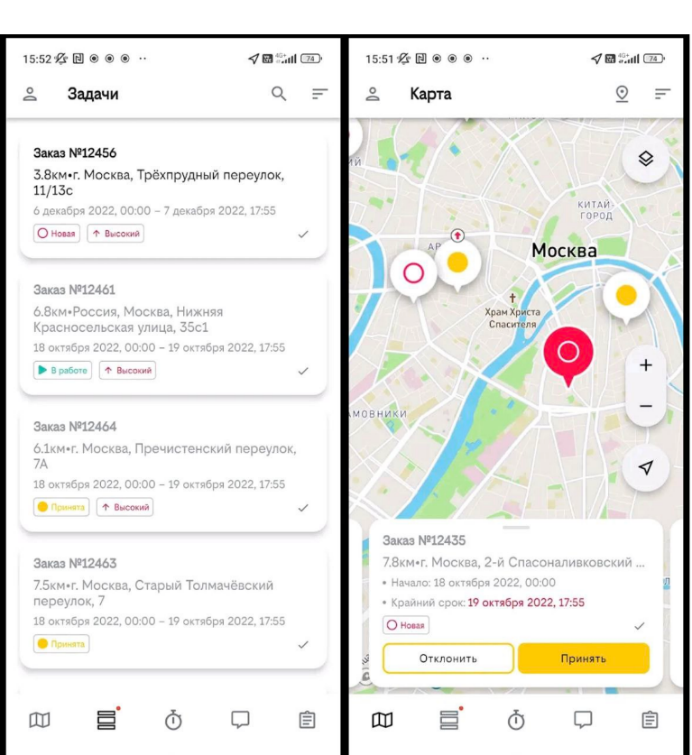


Рисунок 1.4. BeeTrack (Билайн)

ГПБМ [5]

Приложение предназначено специально для услуги «Мониторинг кадров и транспорта», которая доступна для корпоративных клиентов ГПБ Мобайл. Приложение позволяет работать с задачами от диспетчера или руководителя и помогает разъездному сотруднику спланировать свой рабочий день.

Обладает следующим функционалом:

* Прием и выполнение задач от диспетчера/руководителя,
* Планирование рабочего дня и маршрутов,
* Отметки о выполнении задач,
* Функции временного учета,
* Интеграция с системой мониторинга транспорта,
* Возможность оперативной связи с диспетчером

Интерфейс «ГПБМ» представлен на рисунке 1.5.

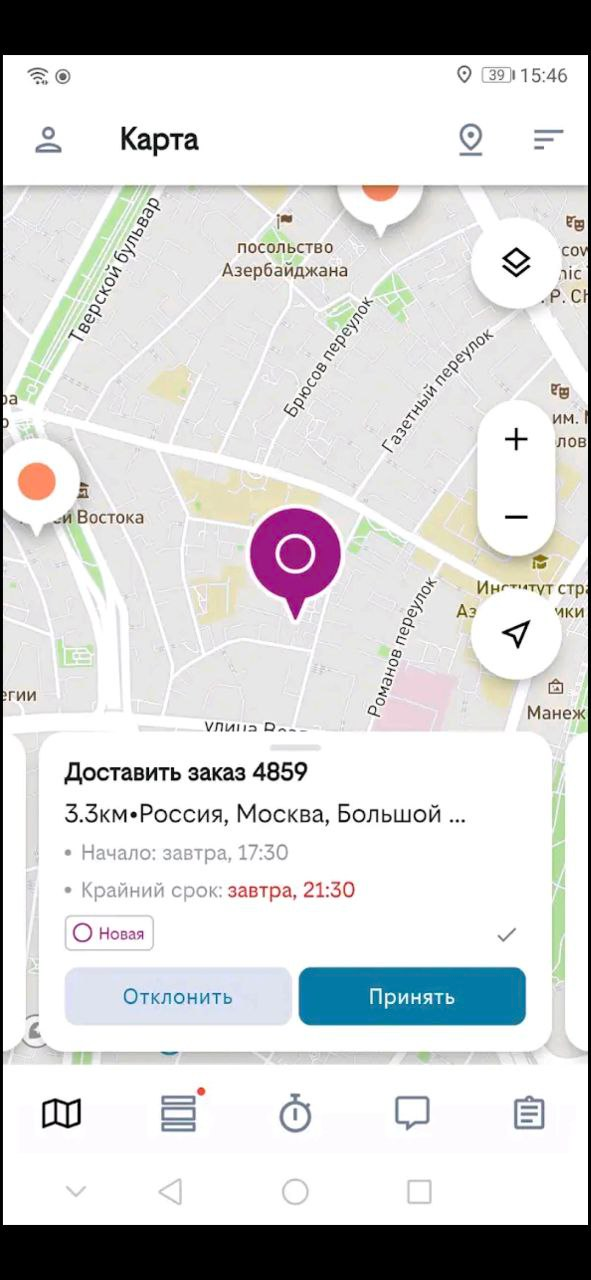


Рисунок 1.5. Газпромбанк Мобайл

Яндекс Еда [6]

Приложение "Яндекс Еда" для курьеров позволяет доставщикам эффективно управлять своим рабочим процессом. Курьеры могут получать заказы в режиме реального времени, видеть расположение ресторанов и адреса доставки, а также оптимизировать маршрут.

Интуитивно понятный интерфейс помогает быстро принимать заказы и отслеживать их статус, а функция обратной связи позволяет пользователям оставлять оценки и комментарии. Кроме того, приложение предлагает информацию о заработке и статистику по заказам, что способствует улучшению качества обслуживания и увеличению дохода.

Особенности «Яндекс Еды»:

* Прием заказов в реальном времени,
* Карта с маршрутом и точками доставки,
* Навигация к ресторанам и клиентам,
* Отслеживание статуса заказов,
* Система рейтингов и обратной связи,
* Статистика выполненных заказов,
* Просмотр информации о заработке,
* Чат с поддержкой и клиентами,
* История выполненных заказов

Интерфейс «Яндекс Еды» представлен на рисунке 1.6.

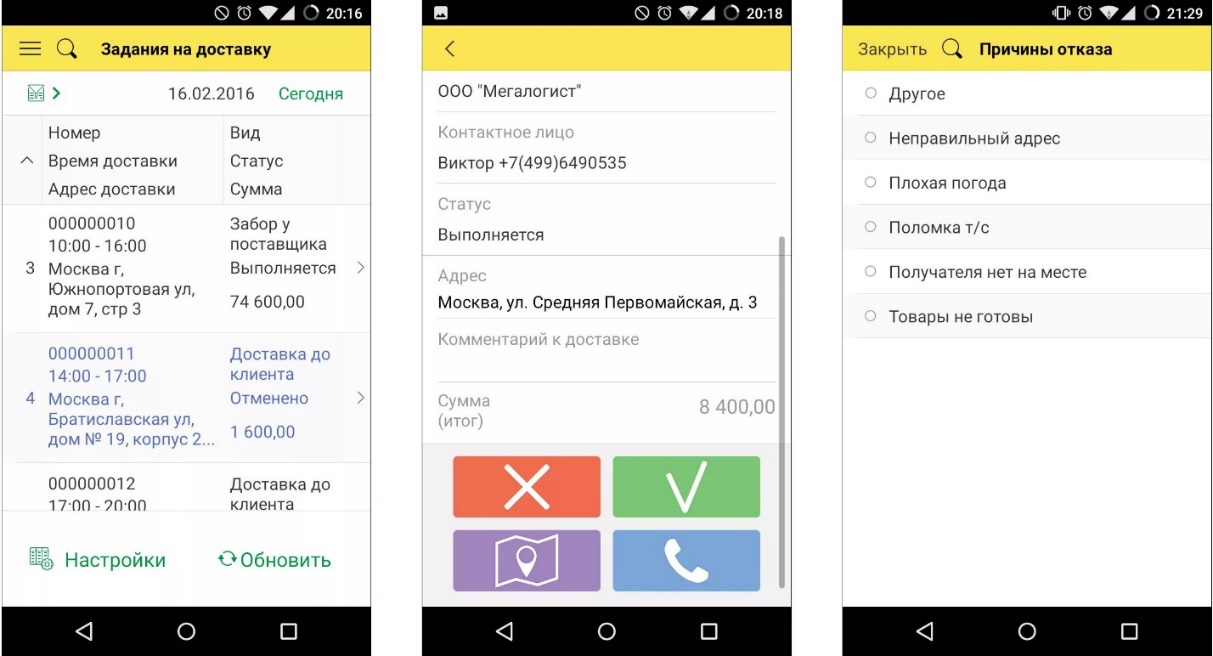


Рисунок 1.6. Яндекс Еда

Рассмотренные аналоги позволили выделить ключевые подходы к реализации подобных веб-приложений на C#.

## 1.3 Потребности пользователей и заинтересованных лиц

Пользователями разрабатываемого веб-приложения являются сотрудники интернет-провайдера, включая:

Инсталляторов (технических специалистов) – монтажников и сервисных инженеров, которым требуется:

* четкое распределение задач с учетом их местоположения
* мобильный доступ к информации о заявках
* простой способ фиксации результатов работы

Диспетчеров – сотрудников call-центра и технической поддержки, которым важно:

* оперативно обрабатывать входящие запросы
* контролировать очередь заявок
* оперативно связываться с клиентами

Руководителей подразделений , заинтересованных в:

* анализе загруженности сотрудников
* контроле качества обслуживания
* получении статистики по типам и частоте обращений

Ключевые потребности всех категорий пользователей:

* Простота и интуитивность интерфейса
* Минимальный, но достаточный функционал для выполнения задач
* Возможность быстрого обучения работе с системой
* Доступ к актуальной информации в режиме реального времени

Система должна обеспечивать удобное взаимодействие между всеми участниками процесса – от момента создания заявки клиентом до ее выполнения техническим специалистом и последующего анализа руководством.

# 2 Постановка задачи

## 2.1 Требования к разрабатываемой системе

### 2.1.1 Функциональные требования

Системные характеристики.

* + СХ-1: Приложение является веб-приложением, с использованием фреймворка Blazor Server.
  + СХ-2: Приложение написано на высокоуровневом языке программирования C#.
  + СХ-3: Приложение взаимодействует с базой данных SQLite.
  + СХ-4: Работа с базой данных осуществляется с использованием LINQ-запросов.

Пользовательские требования.

* ПТ-1: Запуск и выход из приложения.
* ПТ-1.1: Запуск приложения осуществляется через браузер по указанному URL.
* ПТ-1.2: Выход из приложения осуществляется закрытием вкладки браузера или завершением сессии.

ПТ-2: Интерфейс.

* ПТ-2.1: Главное меню представляет собой горизонтальную панель с кнопками для навигации между разделами приложения.
* ПТ-2.2: Данные отображаются в виде таблиц или списков, в зависимости от контекста.
* ПТ-2.3: Интерфейс адаптируется под разные разрешения экранов (десктоп и мобильные устройства).

БП-1: Работа с заявками.

* БП-1.1: Абонент может создавать заявки на устранение технических проблем, смену тарифа или подключение новых функций.
* БП-1.2: Инсталлятор может просматривать список назначенных заявок и обновлять их статус.
* БП-1.3: Руководитель может переназначать заявки между инсталляторами.

БП-2: Отчеты инсталляторов.

* БП-2.1: Инсталлятор должен создать отчет по выполненной заявке, указав результат и комментарий (опционально).

Атрибуты качества.

* АК-1: Производительность.
* АК-1.1: Запуск приложения должен занимать не более 5 секунд на конфигурации: i3-6100, 4 ГБ ОЗУ, Intel HD 530.
* АК-1.2: Выполнение LINQ-запросов к базе данных должно занимать не более 2 секунд на той же конфигурации.

АК-2: Интерфейс.

* АК-2.1: Интерфейс должен быть интуитивно понятным и удобным для пользователей.
* АК-2.2: Основные функции должны быть доступны в 1-2 клика от главного меню.
* АК-2.3: Интерфейс должен корректно отображаться на устройствах с разными разрешениями экрана.

АК-3: Целостность данных

* АК-3.1: Каждая запись в базе данных должна иметь уникальный идентификатор (первичный ключ).

АК-4: Вычислительные механизмы.

* АК-4.1: Статус заявки должен автоматически обновляться при создании отчета инсталлятором.

Детальные спецификации.

* ДС-1: Язык C#.
* ДС-1.1: Минимальная версия .NET — 8.0.
* ДС-1.2: Устанавливаемые пакеты NuGet должны быть совместимы с .NET 8.0.

ДС-2: Сообщения об ошибках.

* ДС-2.1: Сообщение о незаполненных обязательных полях: «Заполните все обязательные поля!».
* ДС-2.2: Сообщение о некорректном вводе данных: «Номер телефона должен содержать 11/12 символов и начинаться с +7/8».
* ДС-2.3: Сообщение о некорректном пароле: «Пароль должен содержать не менее 8 символов».

ДС-3: Отображаемые статусы заявок.

* ДС-3.1: Перечень допустимых статусов заявок: «Новая», «В процессе выполнена», «Выполнена», «Требуется повторный выезд».

ДС-4: Работа с базой данных.

* ДС-4.1: Все запросы к базе данных выполняются через LINQ.
* ДС-4.2: Данные в интерфейсе обновляются автоматически при изменении данных в базе.

## 2.2 Обоснование выбора инструментальных средств

Средой разработки приложения выбрана Microsoft Visual Studio – интегрированная среда разработки, предоставляющая удобный отладчик, систему контроля версий, богатую систему расширений и функции IntelliSense для автодополнения кода.

Приложение разрабатывается на платформе Blazor Server, которая обеспечивает тесную интеграцию с инфраструктурой .NET. Компонентная архитектура фреймворка способствует созданию решений, где критически важная логика безопасно выполняется на стороне сервера.

Серверная часть приложения будет написана на языке программирования C# – современном объекто-ориентированном языке, который сочетает высокую производительность с кроссплатформенностью. Для реализации бизнес-логики служит богатая библиотека .NET и экосистема пакетов NuGet.

Для работы с данными используется СУБД SQLite с инструментом DB Browser for SQLite. Эта система управления базами данных была выбрана благодаря своей компактности, надежности и простоте развертывания. DB Browser for SQLite предоставляет интуитивно понятный графический интерфейс для администрирования, а возможность хранения всей базы данных в одном файле значительно упрощает перенос и развертывание приложения.

В процессе разработки проекта используется система контроля версий GitHub - современная облачная платформа для хостинга проектов и организации командной работы. GitHub обеспечивает надежное хранение исходного кода с полным сохранением истории всех изменений, что позволяет в любой момент вернуться к предыдущим версиям проекта.

# 3 Проектирование приложения

## 3.1 Архитектурное проектирование

Разрабатываемая система представляет собой веб-приложение, построенное на архитектурном шаблоне Model-View-ViewModel с использованием Entity Framework Core для взаимодействия с базой данных..

Структура организована следующим образом:

* Модели – классы, описывающие основные сущности системы (пользователи, заявки, услуги, тарифы и др.) и их взаимосвязи.
* Представления – компоненты пользовательского интерфейса, отвечающие за отображение данных и взаимодействие с пользователем. Реализованы с использованием Razor Pages..
* Модели представления – промежуточные классы, которые адаптируют данные моделей для отображения в представлениях, содержат логику преобразования данных и команды для обработки пользовательских действий.

Таким образом, бизнес-логика приложения распределена между моделями и моделями представления, что обеспечивает удобство дальнейшего развития системы без необходимости кардинального изменения архитектуры.

## 3.2 Проектирование базы данных

Разработанная ER – диаграмма в нотации Мартина представлена на рисунке 3.1.

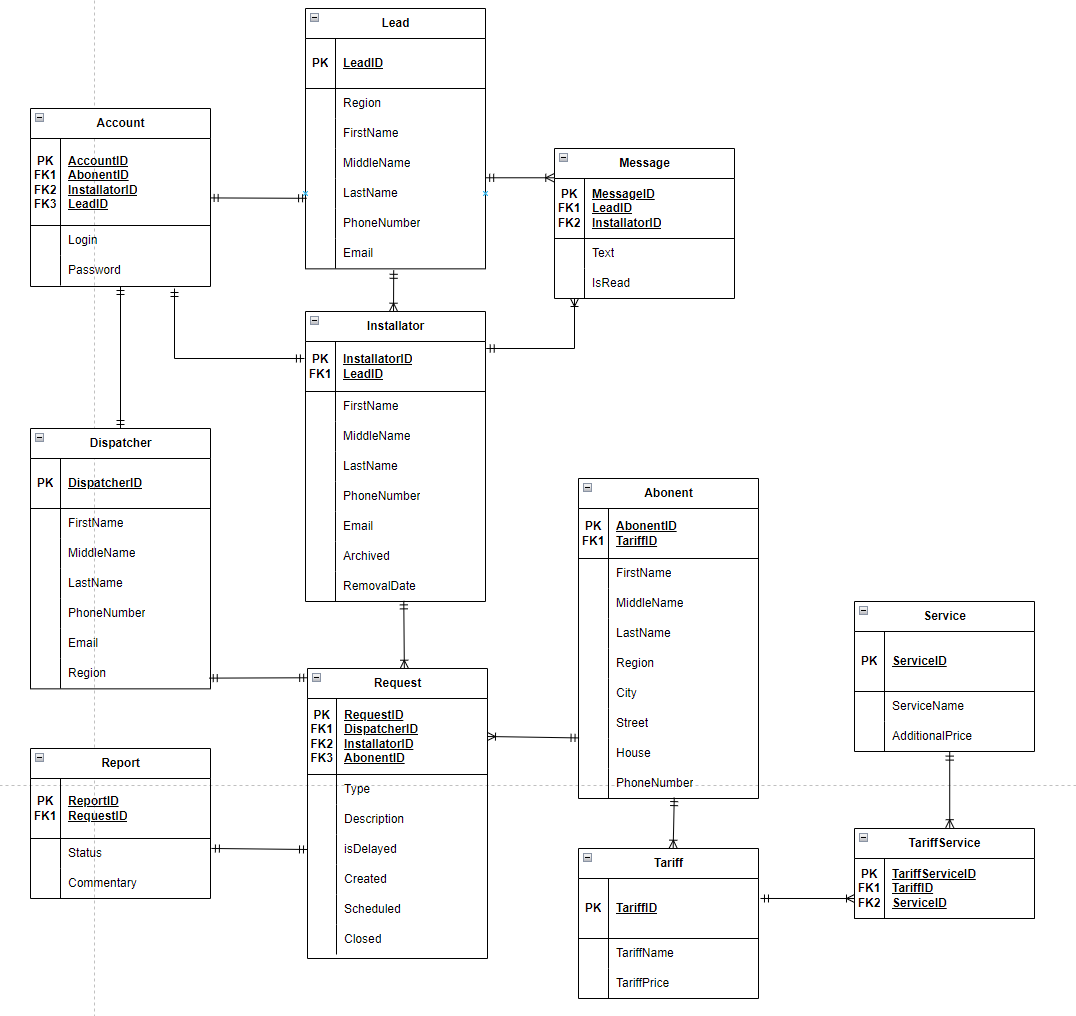


Рисунок 3.1 − ER − диаграмма в нотации Мартина

Созданная ER-диаграмма четко демонстрирует объекты предметной области, их характеристики и взаимосвязи.

## 3.3 Проектирование интерфейса пользователя

Перед началом разработки приложения были спроектированы макеты пользовательского интерфейса.

Макет страницы авторизации представлен на рисунке 3.2 Пользователю необходимо ввести электронную почту или номер телефона и пароль. Приложение предназначено только для сотрудников, поэтому функция регистрации недоступна. В случае утери пароля предусмотрено его восстановление через письмо, отправленное на электронную почту.

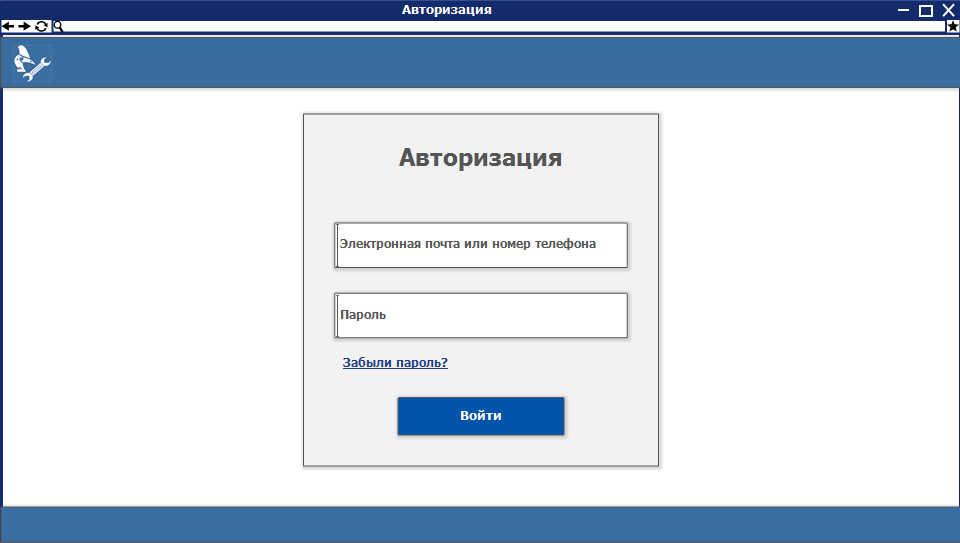


Рисунок 3.2 – Макет страницы авторизации

Макет страницы с расписанием выездов представлен на рисунке 3.3. На этой странице отображается список вызовов, назначенных сотруднику.

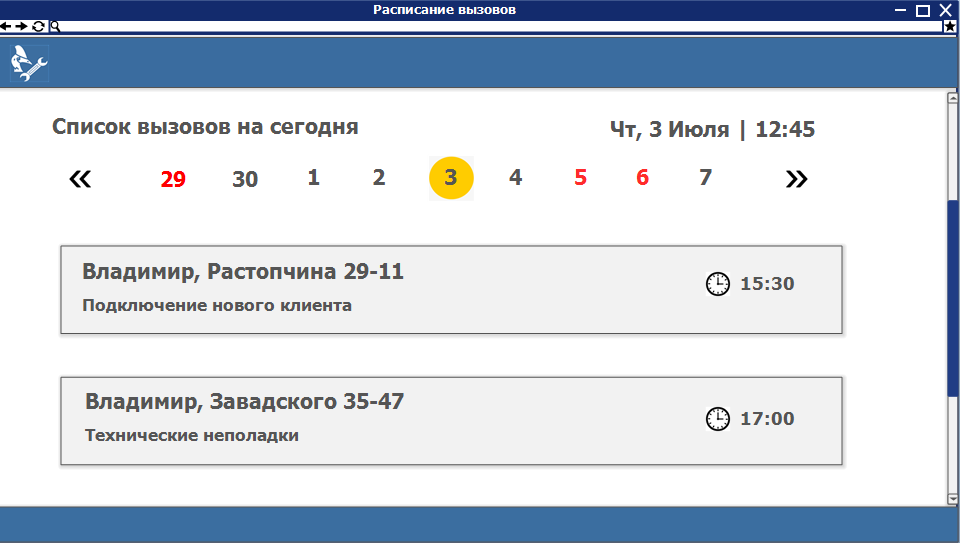


Рисунок 3.3 – Макет страницы с расписанием выездов

Макет страницы с информацией о выезде представлен на рисунке 3.4. Здесь представлены детали заявки и виджет с картой для построения маршрута.

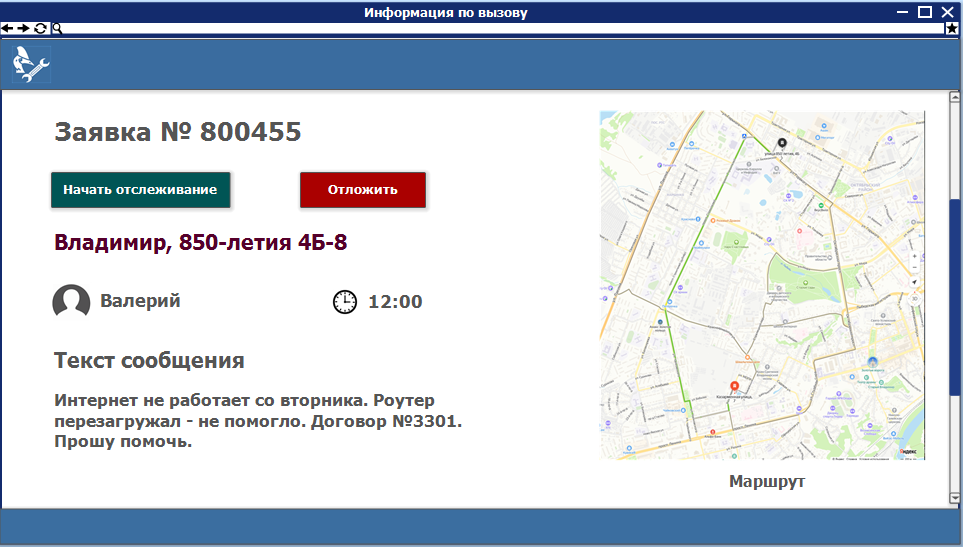


Рисунок 3.3 – Макет страницы с информацией о выезде

Макет страницы руководителя представлен на рисунке 3.5. В этом разделе отображен список сотрудников отдела.

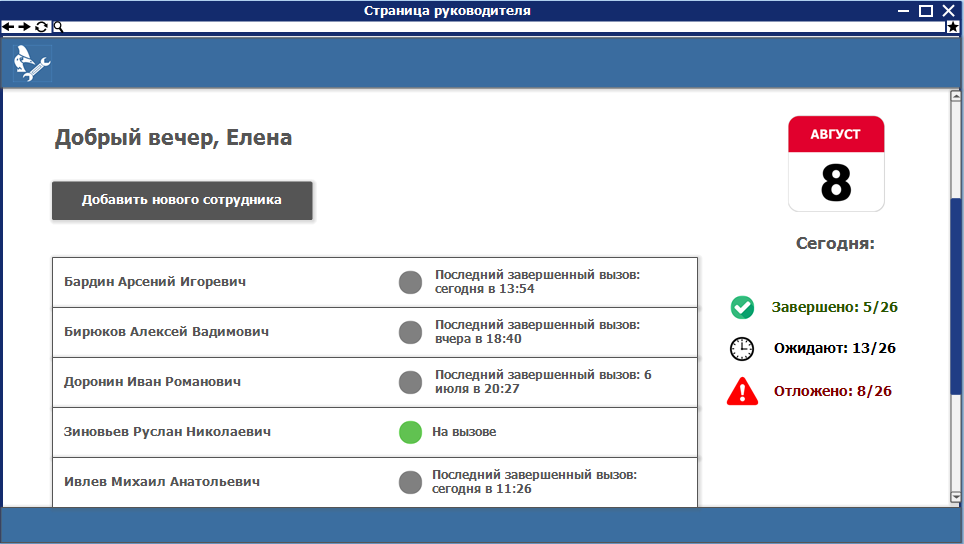


Рисунок 3.5 – Макет страницы руководителя

Макет страницы с сотрудником представлен на рисунке 3.6. Здесь указаны контактные данные подчиненного и его график работы.

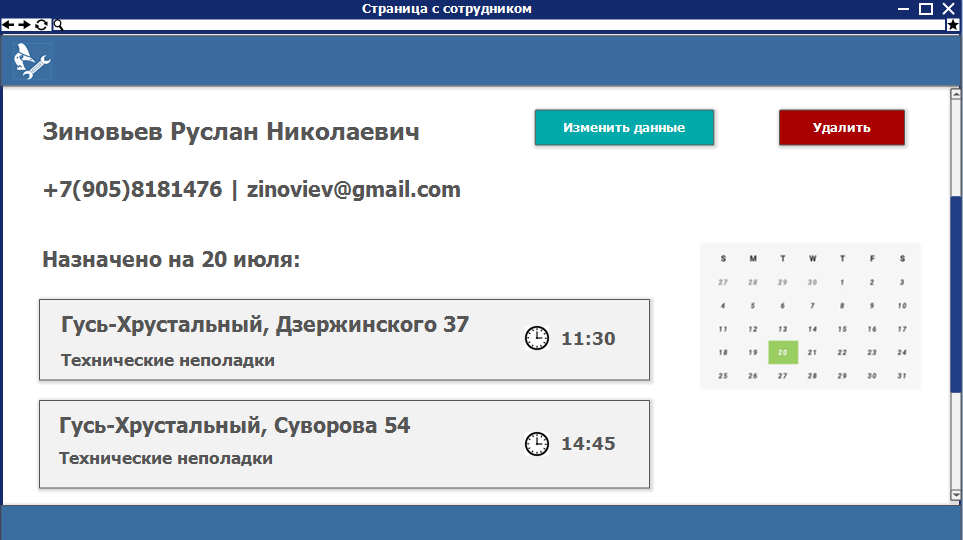


Рисунок 3.6 – Макет страницы с сотрудником

За каждым сотрудником ведется учет отработанных часов. Макет страницы со статистикой представлен на рисунке 3.7.

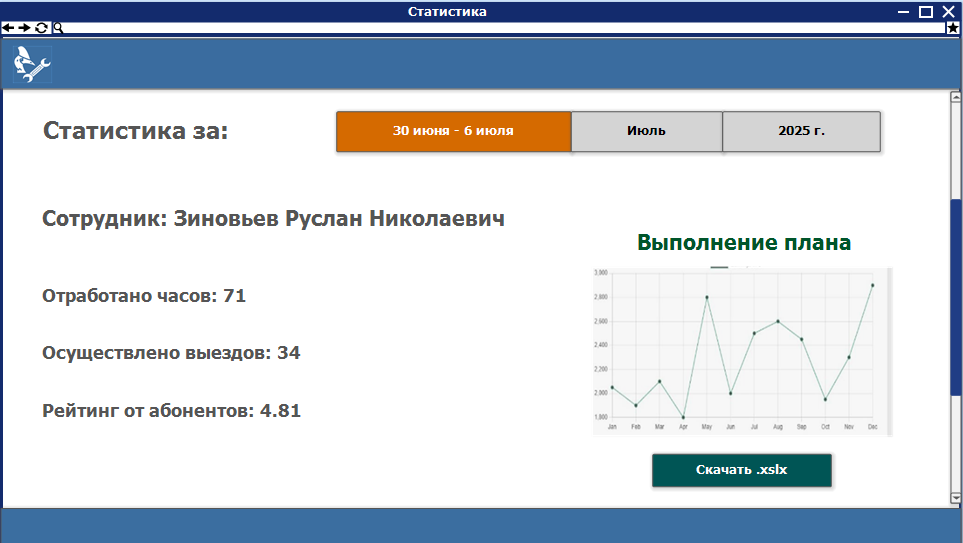


Рисунок 3.7 – Макет страницы со статистикой

В итоге был разработан прототип пользовательского интерфейса, который иллюстрирует ключевые функциональные возможности и сценарии использования приложения

## 3.5 Диаграммы UML

### 3.5.1 Диаграмма классов

На рисунке 3.8 представлена диаграмма классов

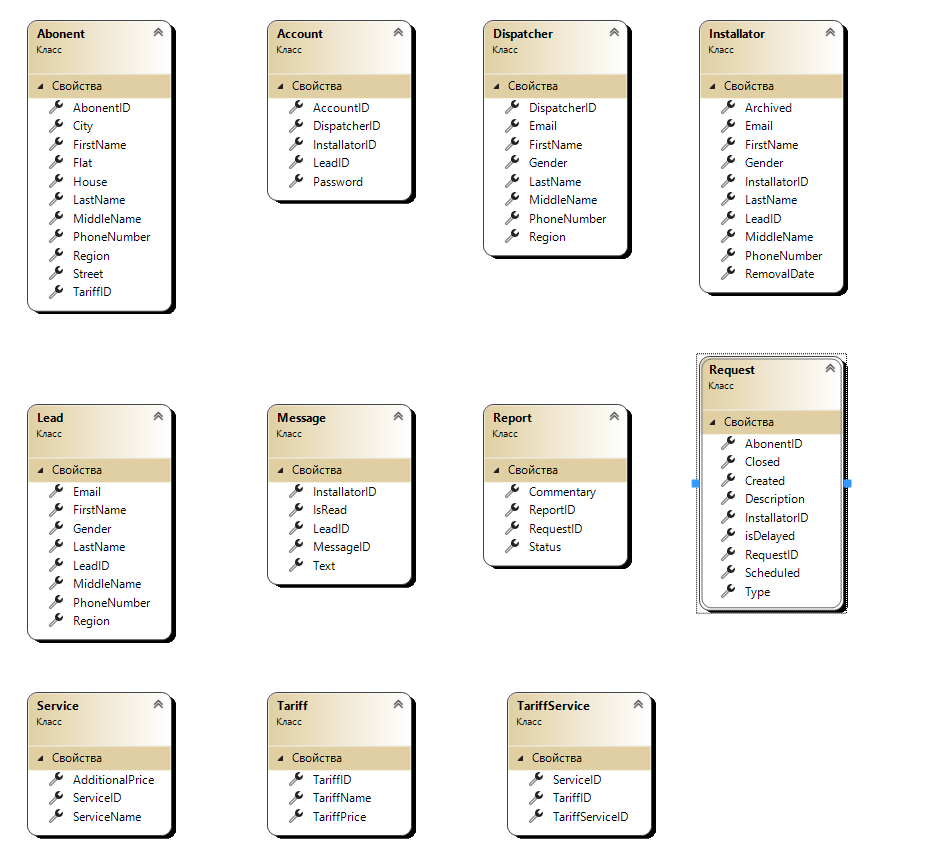


Рисунок 3.8 – Диаграмма классов

Таким образом диаграмма классов предоставляет четкое представление о структуре системы и взаимосвязях между её компонентами.

### 3.5.2 Диаграмма прецедентов

Диаграмма прецедентов представлена на рисунке 3.9.

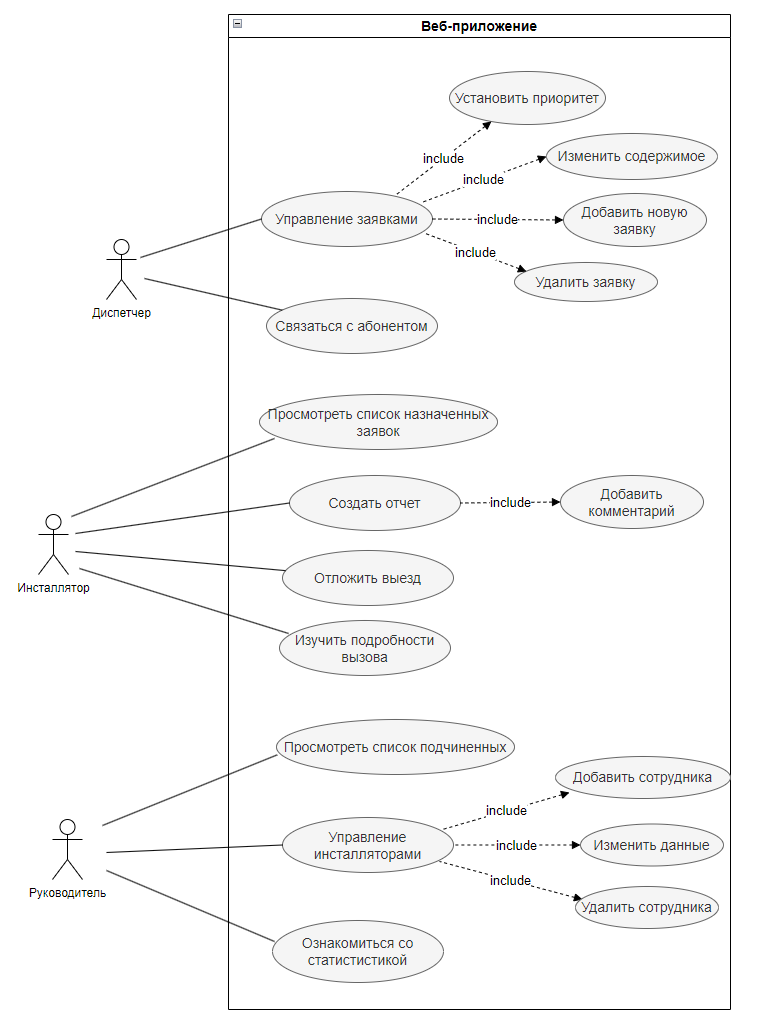


Рисунок 3.9 – Диаграмма прецедентов

В диаграмме прецедентов представлены три актера: инсталлятор, диспетчер, руководитель. Инсталлятор смотрит расписание, фиксирует результаты и заполняет отчёты. Диспетчер добавляет новые заявки. Руководитель управляет сотрудниками и отслеживает статистику.

### 3.5.3 Диаграммы деятельности

Диаграмма деятельности для обработки заявки представлена на рисунке 3.10.

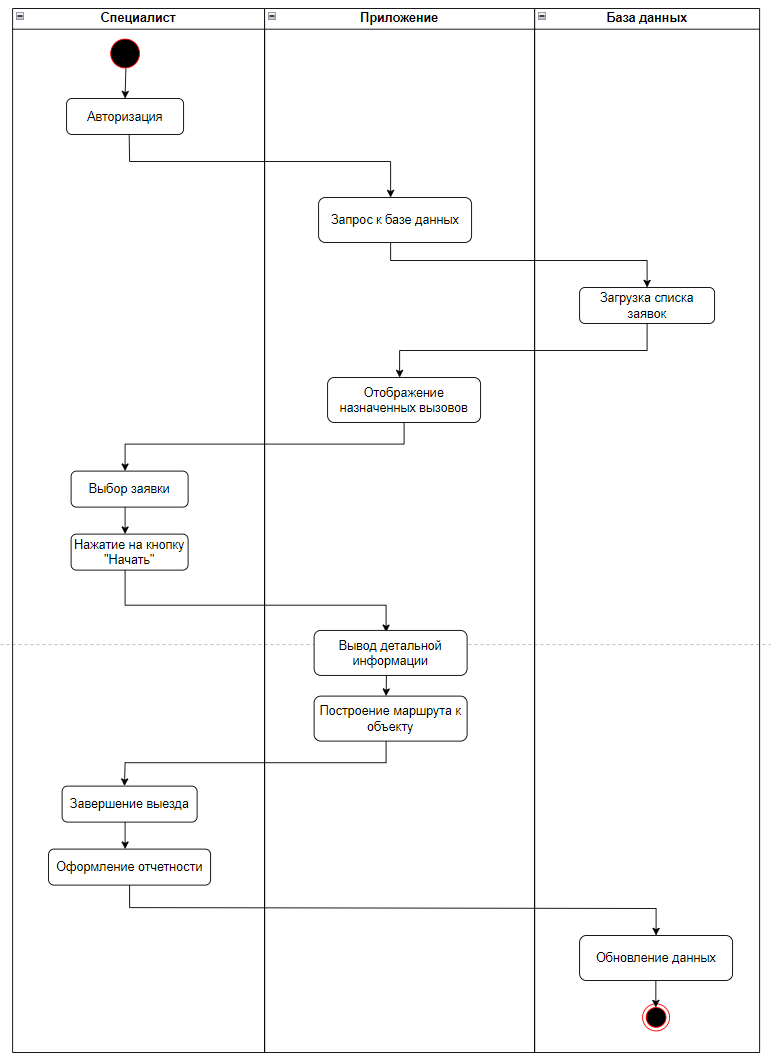


Рисунок 3.10 – Диаграмма активности. Обработка заявки

Перед завершением выезда инсталлятор указывает причину проблемы и способ ее устранения во всплывающем окне. Для заявок на подключение достаточно нажать «Завершить».

Диаграмма деятельности для добавления нового сотрудника на рисунке 3.11.

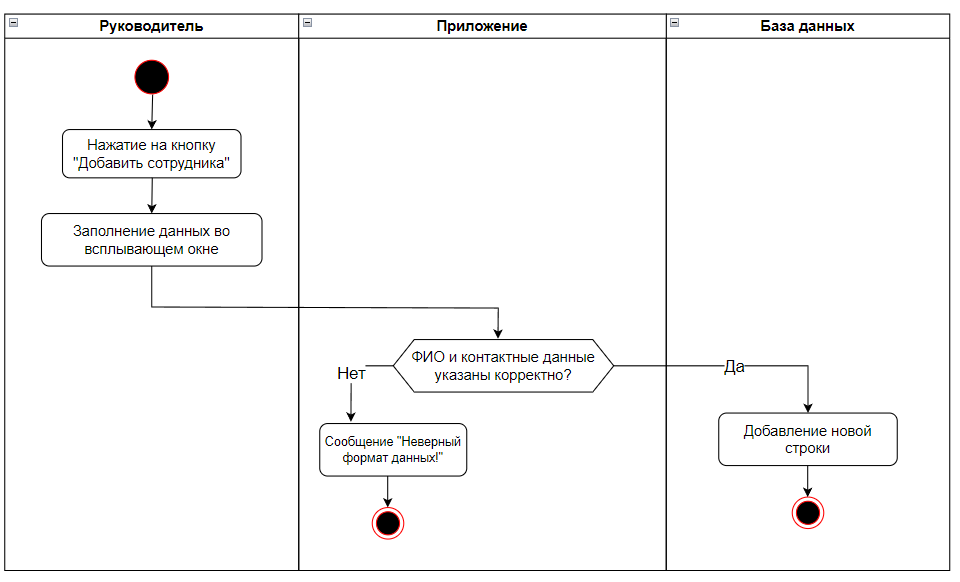


Рисунок 3.11 – Диаграмма активности. Добавление нового сотрудника

Приложение проверяет корректность вводимых данных. Фамилия и имя не должны содержать цифр, номер телефона должен начинаться с +7 или 8.

Диаграмма деятельности для передачи заявки другому лицу представлена на рисунке 3.12.

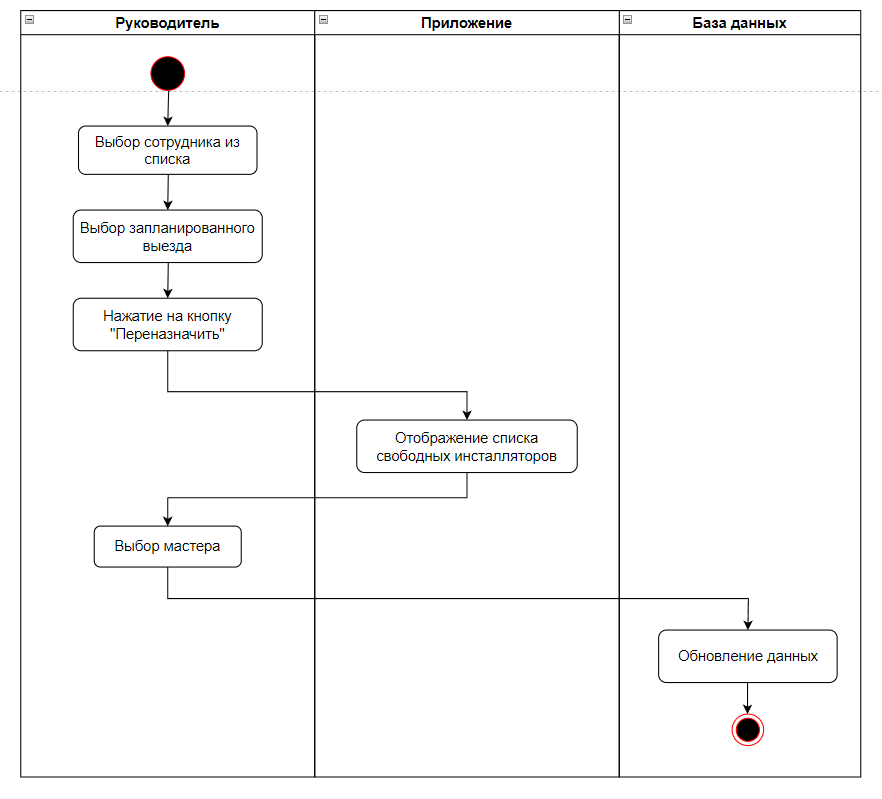


Рисунок 3.12 – Диаграмма активности. Переназначение заявки другому специалисту

Приложение учитывает запланированное время выезда и выбирает только тех специалистов, у которых в расписании в этот момент еще есть свободное время.

# 4 Разработка приложения

## 4.1 Реализация архитектуры

Для реализации функционала работы с данными в системе было создано восемь моделей, описывающих основные сущности приложения:

* Abonent
* Account
* Dispatcher
* Installator
* Lead
* Message
* Report
* Request
* Service
* Tariff
* TariffService

Листинг кода реализованных моделей представлен в приложении А.

Классы моделей и класс AppDbContext.cs представлены на рисунке 4.1.

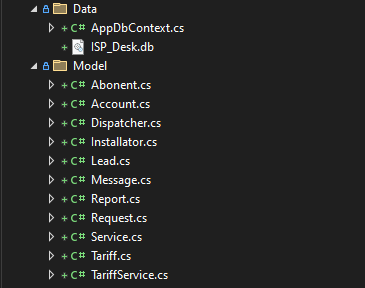


Рисунок 4.1 – Модели данных

Таким образом, в системе определены одиннадцать моделей данных, охватывающих все сущности приложения.

## 4.2 Реализация базы данных

Реализованная в SQLite база данных имеет структуру, соответствующую разработанной ER-диаграмме. Разработанная база данных представлена на рисунках 4.2 – 4.3.

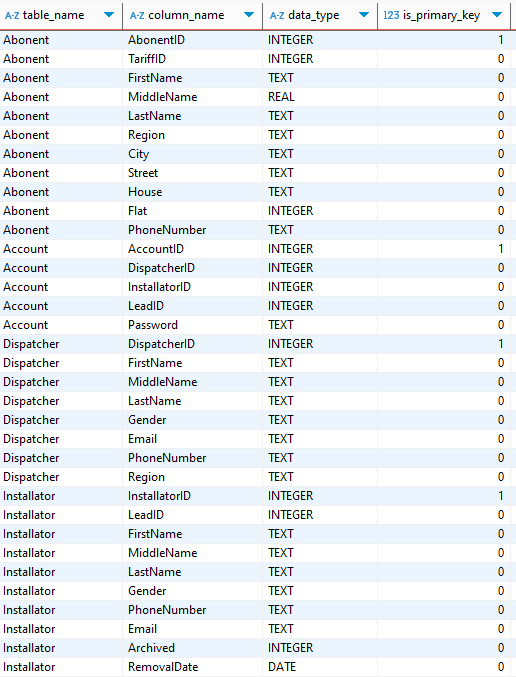


Рисунок 4.2 – Структура базы данных

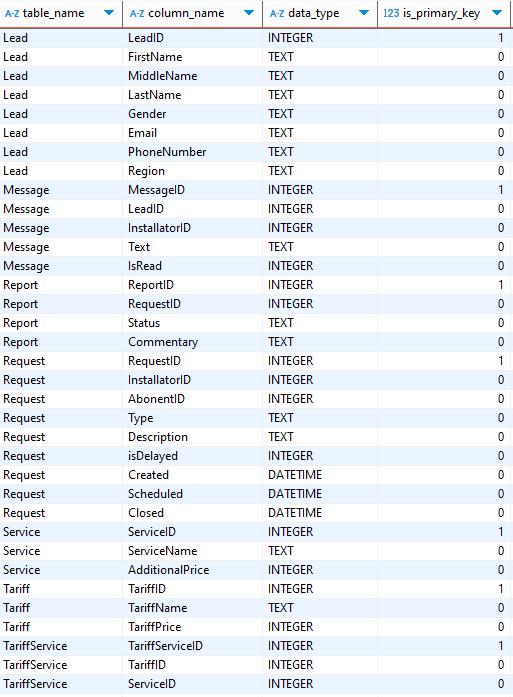


Рисунок 4.3 – Структура базы данных

Таким образом, все сущности, их атрибуты и взаимосвязи, представленные на диаграмме, нашли свое точное отражение в структуре созданной базы данных.

## 4.3 Реализация пользовательского интерфейса

Запуск приложения можно осуществить либо с помощью ярлыка на рабочем столе, либо введя URL в адресную строку браузера. Страница авторизации изображена на рисунке 4.4.

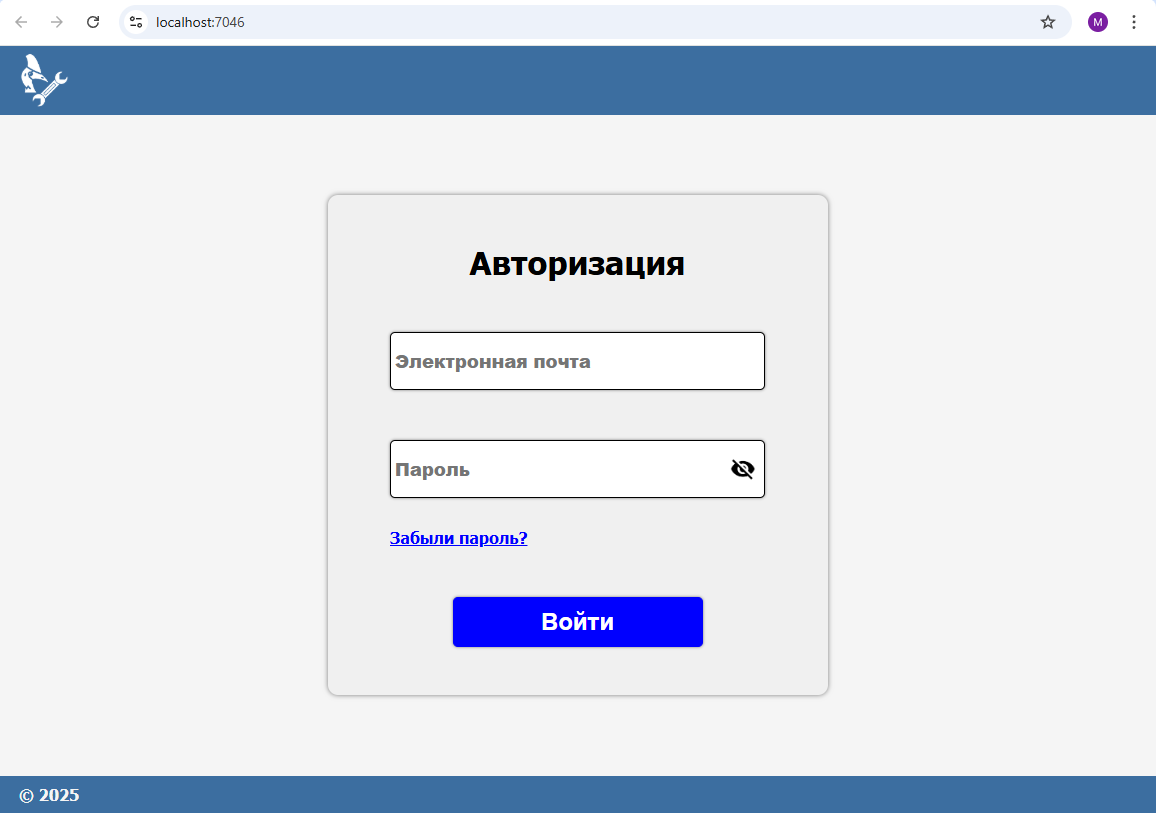


Рисунок 4.4 – Страница авторизации

Страница, на которую попадает инсталлятор после авторизации, представлена на рисунке 4.5.

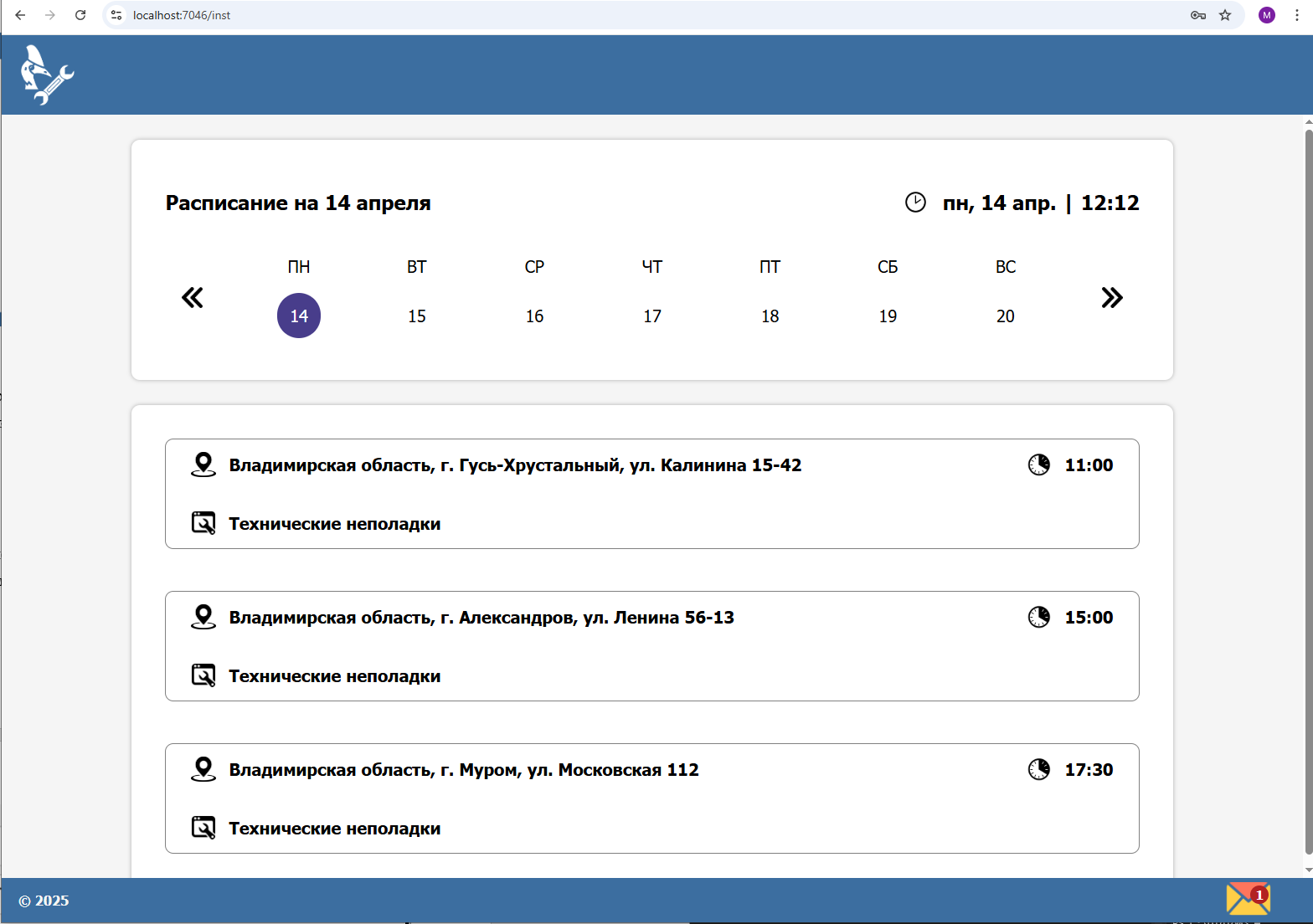


Рисунок 4.5 – Страница инсталлятора

Главная страница руководителя изображена на рисунке 4.6.

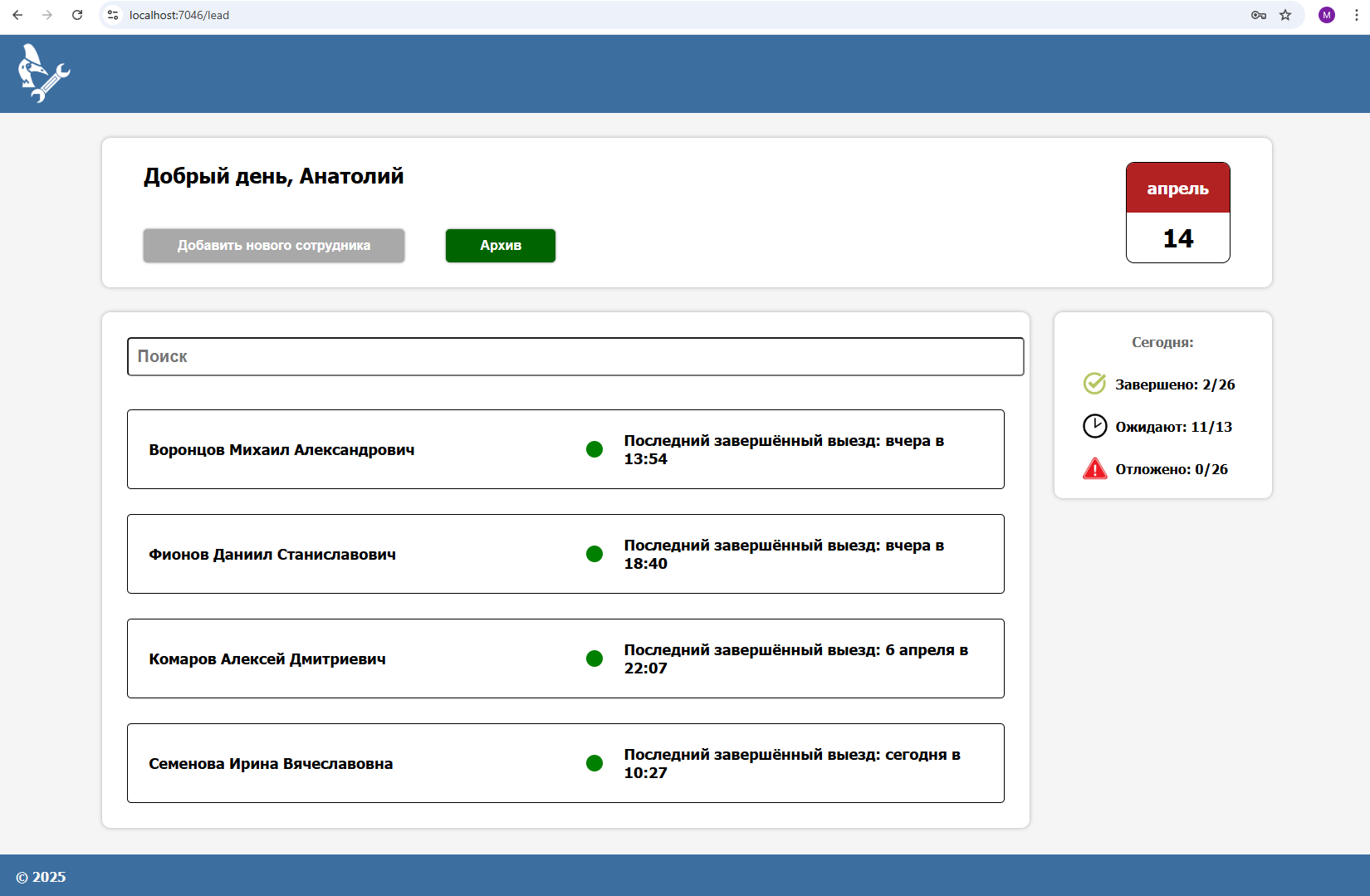


Рисунок 4.6 – Страница руководителя

Страница со списком сотрудников представлена на рисунке 4.7.

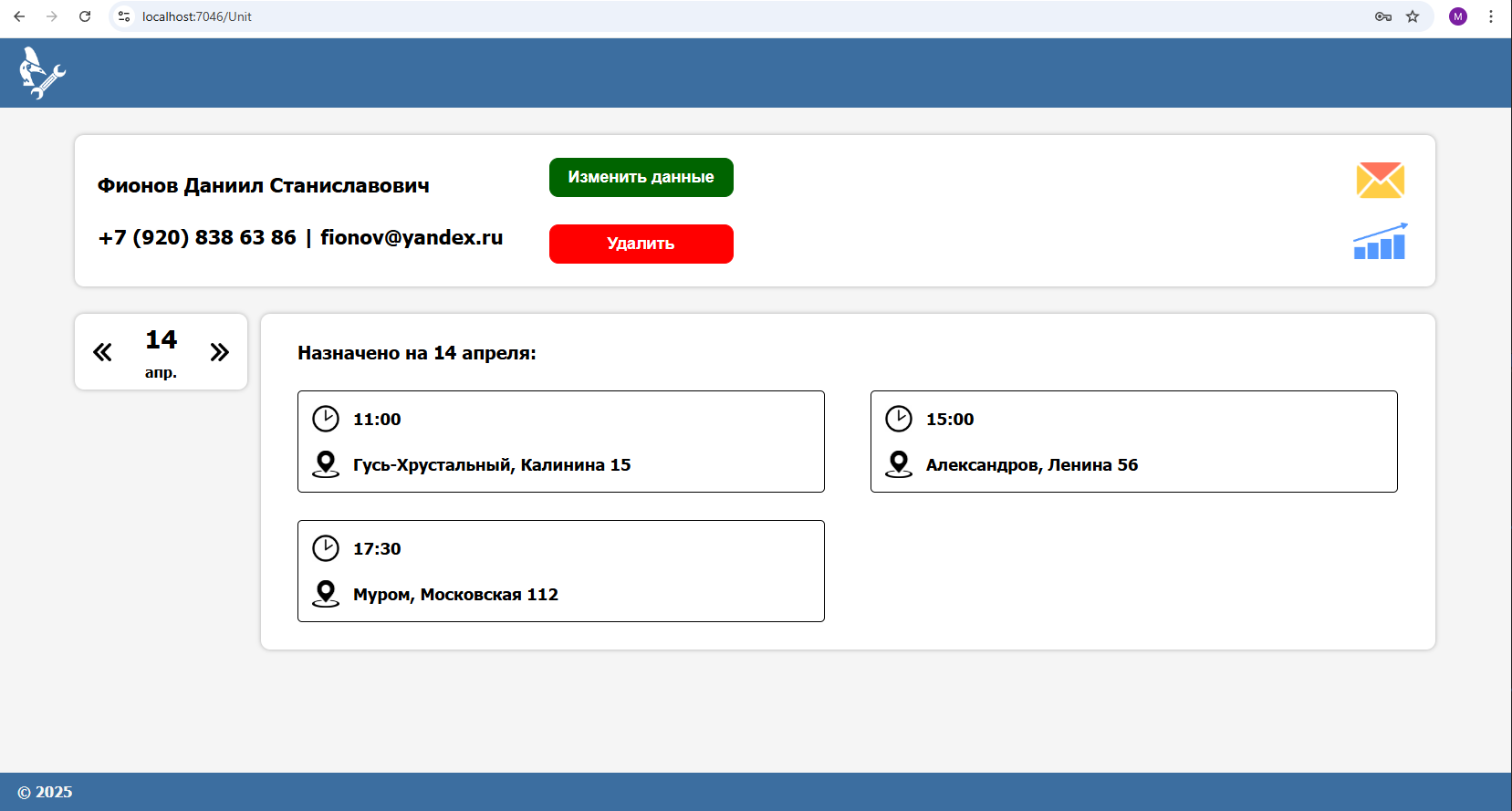


Рисунок 4.7 – Сотрудник

Всплывающее окно для добавления нового специалиста изображен на рисунке 4.8.

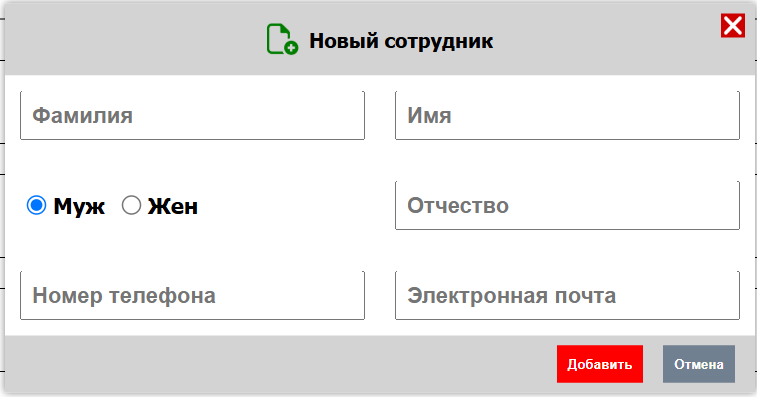


Рисунок 4.8 – Попап для добавления

Архив с удаленными специалистами представлен на рисунке 4.9.

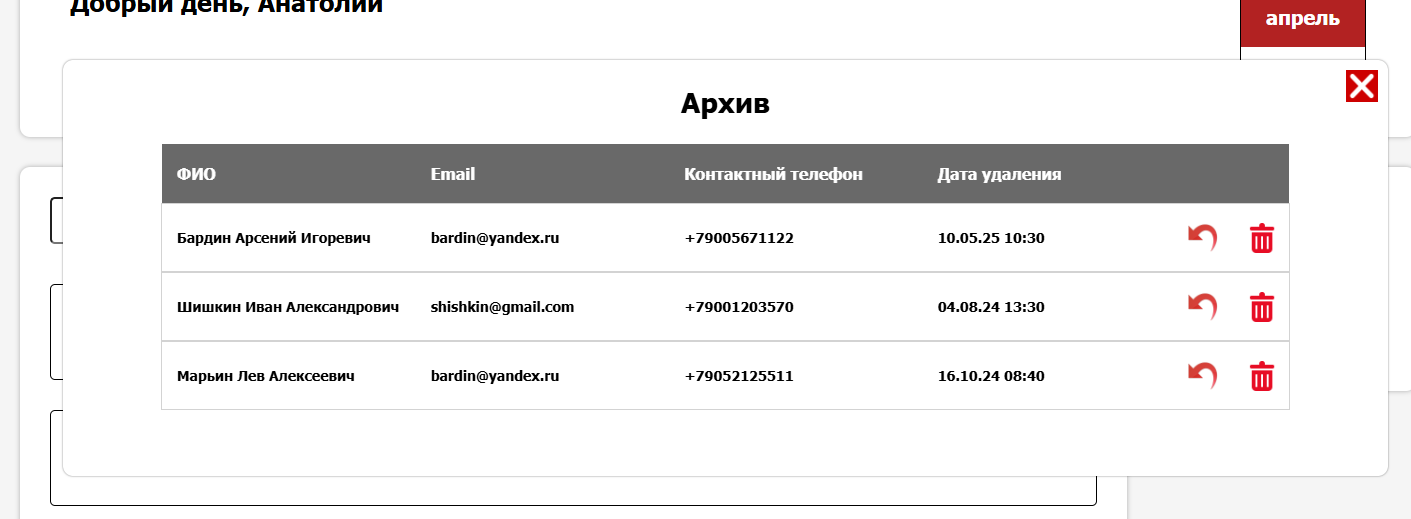


Рисунок 4.9 – Архив

Всплывающие окна для отправки и получения сообщений представлены на рисунках 4.10 – 4.11.

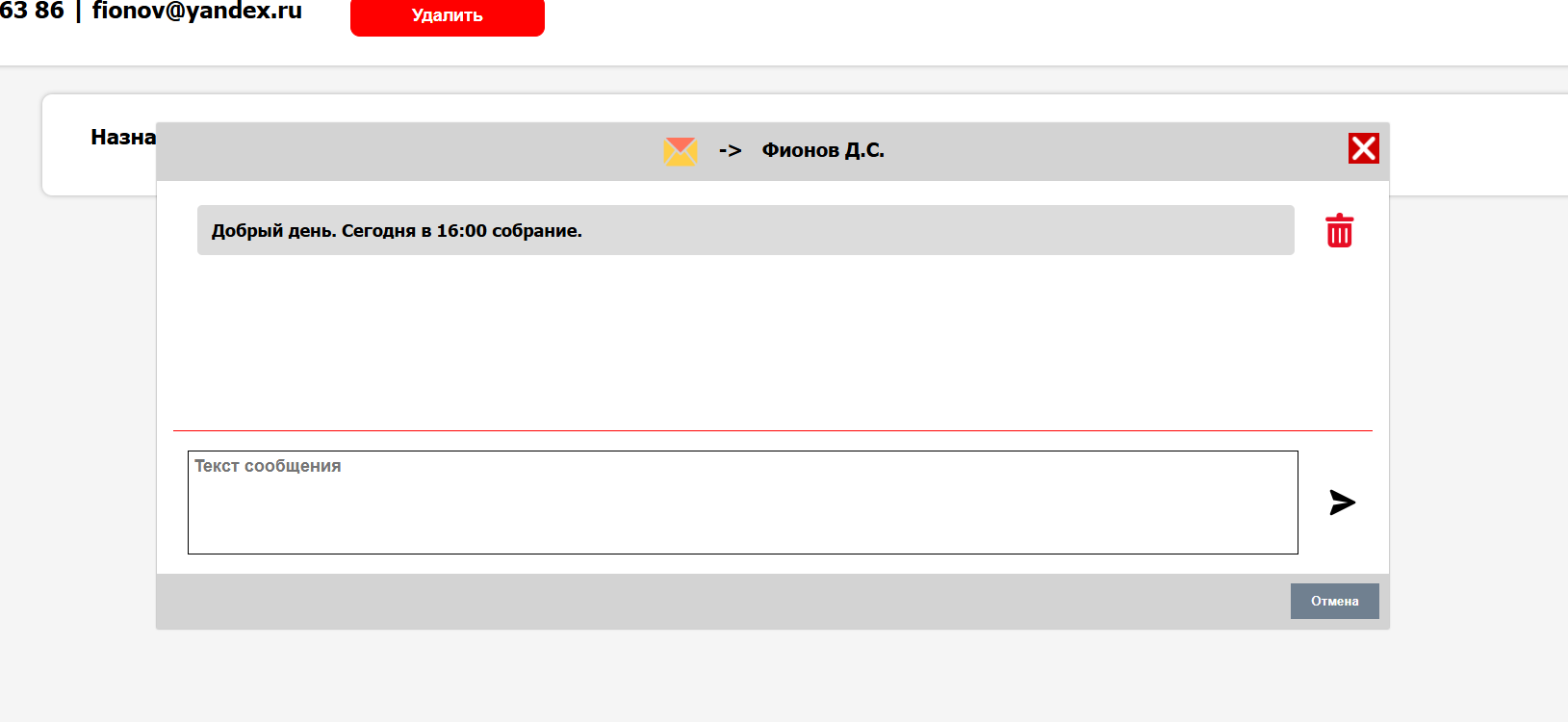


Рисунок 4.10 – Отправленные

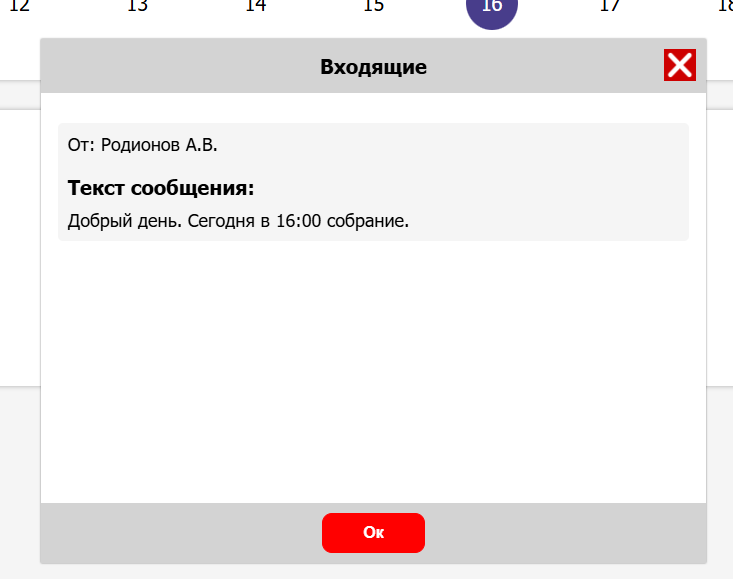


Рисунок 4.11 – Входящие

В данном разделе был представлен спроектированный пользовательский интерфейс.

# 5 Тестирование

# Список использованных источников

1. МТС Координатор [Электронный ресурс] — URL: https://www.mts-ru.com (дата обращения: 14.05.25)
2. Мобильный агент [Электронный ресурс] — URL: https://www.rt.ru (дата обращения: 14.05.25)
3. Дом.ру [Электронный ресурс] — URL: https://www.dom.ru (дата обращения: 14.05.25)
4. BeeTrack [Электронный ресурс] — URL: https://www.beeline-r.ru (дата обращения: 14.05.25)
5. ГПБМ [Электронный ресурс] — URL:https://www.gazprom.ru (дата обращения: 14.05.25)
6. Яндекс Еда [Электронный ресурс] — URL: https://www.eda.yandex.ru (дата обращения: 14.05.25)
7. Сэйнти К. Blazor в действии. – М.: ДМК Пресс, 2022. – 320 c. – ISBN: 978-5-93700-179-5.
8. Троелсен Э. Язык программирования C# 10 и платформа .NET 6. – СПб.:Питер, 2022. – 1088 c. – ISBN: 978-5-4461-1849-6.
9. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework на языке C# (4-е изд.). – СПб.: Питер, 2021. – ISBN: 978-5-4461-1375-0.
10. Фримен А. Pro ASP.NET Core MVC 2. – Apress, 2022. – ISBN: 978-1-4842-8174-5.

# Приложение А

Листинг класса Abonent:

public class Abonent

{

public int AbonentID { get; set; }

public int TariffID { get; set; }

public string FirstName { get; set; }

public string MiddleName { get; set; }

public string LastName { get; set; }

public string Region { get; set; }

public string City { get; set; }

public string Street { get; set; }

public string House { get; set; }

public int? Flat { get; set; }

public string PhoneNumber { get; set; }

}

Листинг класса Account:

public class Account

{

public int AccountID { get; set; }

public int? DispatcherID { get; set; }

public int? InstallatorID { get; set; }

public int? LeadID { get; set; }

public string Password { get; set; }

}

Листинг класса Dispatcher:

public class Dispatcher

{

public int DispatcherID { get; set; }

public string FirstName { get; set; }

public string MiddleName { get; set; }

public string LastName { get; set; }

public string Gender { get; set; }

public string PhoneNumber { get; set; }

public string Email { get; set; }

public string Region { get; set; }

}

Листинг класса Installator:

public class Installator

{

public int InstallatorID { get; set; }

public int LeadID { get; set; }

public string FirstName { get; set; }

public string MiddleName { get; set; }

public string LastName { get; set; }

public string Gender { get; set; }

public string PhoneNumber { get; set; }

public string Email { get; set; }

public int Archived { get; set; }

public DateOnly? RemovalDate { get; set; }

}

Листинг класса Lead:

public class Lead

{

public int LeadID { get; set; }

public string FirstName { get; set; }

public string MiddleName { get; set; }

public string LastName { get; set; }

public string Gender { get; set; }

public string PhoneNumber { get; set; }

public string Email { get; set; }

public string Region { get; set; }

}

Листинг класса Message:

public class Message

{

public int MessageID { get; set; }

public int LeadID { get; set; }

public int InstallatorID { get; set; }

public string Text { get; set; }

public int IsRead { get; set; }

}

Листинг класса Report:

public class Report

{

public int ReportID { get; set; }

public int RequestID { get; set; }

public string Status { get; set; }

public string Commentary { get; set; }

}

Листинг класса Request:

public class Request

{

public int RequestID { get; set; }

public int InstallatorID { get; set; }

public int AbonentID { get; set; }

public string Type { get; set; }

public string? Description { get; set; }

public int isDelayed { get; set; }

public DateTime Created { get; set; }

public DateTime Scheduled { get; set; }

public DateTime? Closed { get; set; }

}

Листинг класса Service:

public class Service

{

public int ServiceID { get; set; }

public string ServiceName { get; set; }

public int AdditionalPrice { get; set; }

}

Листинг класса Tariff:

public class Tariff

{

public int TariffID { get; set; }

public string TariffName { get; set; }

public int TariffPrice { get; set; }

}

Листинг класса TariffService:

public class TariffService

{

public int TariffServiceID { get; set; }

public int TariffID { get; set; }

public int ServiceID { get; set; }

}