МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ». Кафедра АВТОМАТИЗИРОВАННЫХ СИСТЕМ УПРАВЛЕНИЯ

Пояснительная записка

к курсовой работе

по дисциплине БАЗЫ ДАННЫХ

на тему: Разработка фрагмента информационной системы рекламного агентства

Студентки 3 курса АП-227 группы

Евдокимовой Анастасии Алексеевны

Руководитель

Эстрайх Игорь Викторович

старший преподаватель кафедры АСУ

Новосибирск 2024

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новосибирский государственный технический университет»

Кафедра автоматизированных систем управления

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)

Студент Евдокимова Анастасия Алексеевна АП-227

Тема курсовой работы

Разработка фрагмента информационной системы рекламного агентства

Утверждена на заседании кафедры от

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_протокол № \_\_\_\_\_

Срок защиты работы

Краткая аннотация задания: Разработка фрагмента информационной системы рекламного агентства с использованием языка MySql

Научный руководитель Эстрайх Игорь Викторович

Задание приняла к исполнению А.А. Евдокимова

Дата 13.09.2024

Курсовая работа по дисциплине «Программирование» на тему «Разработка фрагмента информационной системы рекламного агентства» студентки третьего курса Евдокимовой А.А.

Количество страниц – 54, рисунков – 20, количество используемых источников – 9, количество приложений – 1.

Ключевые слова, идентифицирующие предметную область – база данных, SQL, проектирование базы данных, управление услугами, клиентская база, безопасность данных.

Объект исследования – процесс управления рекламной деятельностью агентства, включающий организацию рекламных кампаний, учет клиентов и заказов, аренду рекламных площадей, управление рекламными площадями.

Предмет исследования – автоматизация процессов управления рекламной деятельностью агентства с использованием базы данных и программного обеспечения для учета заказов, аренды рекламных площадей. Разработка базы данных для хранения информации о клиентах, заказах и рекламных площадях. Механизмы управления заказами на рекламу, включая сроки размещения, стоимость и оплату. Разработка пользовательского интерфейса для взаимодействия с системой.

Цель работы – разработка и внедрение программного обеспечения для автоматизации учета и управления рекламной деятельностью агентства, что позволит повысить эффективность работы, минимизировать ошибки при ведении документации и улучшить контроль над процессами аренды рекламных площадей и заказов клиентов.

Задачи работы

1. Анализ предметной области – изучение структуры работы рекламного агентства, его основных бизнес-процессов и требований к автоматизированной системе.
2. Проектирование базы данных – определение сущностей, связей и структуры хранения данных о клиентах, рекламных площадях и заказах.
3. Разработка функционала управления заказами – учет клиентов, оформление заказов, контроль сроков размещения и оплат.
4. Создание модуля управления рекламными площадями – учет доступных мест размещения, их занятости и сроков аренды.
5. Разработка пользовательского интерфейса – создание удобного интерфейса для сотрудников агентства.
6. Тестирование и отладка программы – проверка работоспособности, выявление и исправление ошибок.

Методы исследования (используемые инструменты) – Анализ предметной области, методы проектирования и разработки ПО, методы программирования, методы тестирования.

Краткая характеристика полученных результатов

В ходе выполнения курсовой работы была разработана информационная система для автоматизации процессов рекламного агентства, что позволило оптимизировать управление рекламными площадями и заказами клиентов.

Разработана и внедрена база данных. Определены ключевые сущности (клиенты, рекламные площадки, заказы, платежи). Создана нормализованная реляционная модель базы данных. Реализованы SQL-запросы для управления данными. Разработан модуль управления заказами и клиентами. Реализован учет заказчиков с возможностью хранения их данных. Внедрен механизм оформления заказов, контроля сроков размещения и оплаты. Создана система управления рекламными площадями. Разработан механизм аренды рекламных щитов и экранов. Добавлена проверка доступности рекламных мест. Встроена система отчетности по клиентам, заказам и платежам. Разработан удобный пользовательский интерфейс. Реализованы функции добавления, редактирования и поиска данных. Проведено тестирование и отладка системы. Проверена корректность работы основных функций. Исправлены выявленные ошибки, проведена оптимизация кода.

Разработанная программа позволила автоматизировать ключевые процессы рекламного агентства, что привело к повышению скорости обработки заказов, снижению ошибок при ведении учета и улучшению контроля над арендой рекламных площадей. Система готова к дальнейшему расширению и внедрению новых функций.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 7](#_Toc193363454)

[1 Анализ предметной области 9](#_Toc193363455)

[1.1 Описание предметной области 9](#_Toc193363456)

[1.2 Описание процессов (процедур) 10](#_Toc193363457)

[1.3 Перечень входных и выходных данных (документов) 11](#_Toc193363458)

[1.4 Список пользователей (ролей) 13](#_Toc193363459)

[2. Проектирование базы данных 16](#_Toc193363460)

[2.1 Концептуальная модель (ER-диаграмма) 16](#_Toc193363461)

[3 Выбор и обоснование инструментальных средств для разработки программного обеспечения 22](#_Toc193363462)

[3.1 Выбор системы управления базами данных 22](#_Toc193363463)

[3.2 Выбор языка программирования 31](#_Toc193363464)

[3.3 Выбор интегрированной среды разработки 33](#_Toc193363465)

[4 Программная реализация информационной системы 35](#_Toc193363466)

[5 Руководство пользователя 42](#_Toc193363467)

[5.1 Назначение и условия применения 42](#_Toc193363468)

[5.2 Подготовка к работе 42](#_Toc193363469)

[5.3 Описание операций 42](#_Toc193363470)

[5.4 Аварийные ситуации 43](#_Toc193363471)

[6 Руководство системного программиста 44](#_Toc193363472)

[6.1 Общие сведения 44](#_Toc193363473)

[6.2 Структура программы 44](#_Toc193363474)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 46](#_Toc193363475)

[Приложение А. Руководство пользователя 47](#_Toc193363476)

# ВВЕДЕНИЕ

В современном мире рекламная индустрия играет ключевую роль в продвижении товаров и услуг. Рекламные агентства обслуживают множество клиентов, управляют рекламными кампаниями, арендуют рекламные площадки и анализируют эффективность размещенной рекламы. Однако традиционные методы управления (таблицы, бумажный учет, ручной контроль заказов) уже не соответствуют требованиям динамичного рынка, приводя к ошибкам, задержкам и неэффективному использованию ресурсов.

В связи с этим актуальность данной работы заключается в необходимости разработки автоматизированной системы управления рекламным агентством, которая позволит:

* Оптимизировать учет клиентов и заказов за счет централизованного хранения данных.
* Автоматизировать процесс аренды рекламных площадей и контроля сроков размещения.
* Обеспечить учет финансовых операций (оплата, статистика).
* Сократить временные и трудозатратные расходы сотрудников агентства.

Таким образом, разработка информационной системы для управления рекламными процессами является востребованной и актуальной задачей, так как позволит повысить производительность, минимизировать ошибки и создать удобный инструмент для работы рекламных агентств.

Целью работы является разработка программы для рекламного агентства.

Для реализации поставленной цели необходимо решить ряд задач:

– изучить методику работы базы данных «My SQL»;

– изучить способы разработки алгоритма ПО;

– проанализировать алгоритмы и описать их работу;

– разработать программу для хранения и обработки информации;

– создать пользовательский интерфейс.

В результате выполнения работы возможно использование полученного программного обеспечения для хранения и обработки сведений о клиентах.

Теоретической основой написания курсовой работы явились работы следующих авторов: Гниденко И.Г.[1], Гордеев С.И.[2], Зыков С.В[3], Кудрина Е.В.[4], Малявко А.А.[5].

Практической основой написания курсовой работы явились учебные пособия Маркин А.В [6], Нортроп Т.[7], Советов Б.Я.[8], Трофимов В.В.[9], Фридман.А.Л[10].

Курсовая работа написана на языке программирования C# с использованием свободной IDE – Visual Studio 2022. Также в программе используется база данных MySQL, которая обеспечивает хранение информации о пользователях и предоставленных им услугах.

# Анализ предметной области

## Описание предметной области

Разработать базу данных для рекламного агентства. В собственности агентства находятся рекламные щиты, расположенные в разных местах города. На этих щитах может размещаться реклама по заказу любого арендатора, причем один щит является неделимой рекламной единицей (т.е. его нельзя одновременно предоставлять нескольким арендаторам). Срок и стоимость аренды щита, а также стоимость изготовления самой рекламы оговариваются в условиях договора с агентством. По одному договору заказчик может арендовать сразу несколько рекламных щитов.

Для каждого рекламного щита в БД должна присутствовать следующая информация: регистрационный номер; район города; точный адрес расположения; описание местоположения; полезная площадь. При обращении в агентство по каждому арендатору необходимо фиксировать следующие данные: название арендатора и его статус (ТОО, ЗАО, ИЧП и др.); юридический адрес; ФИО и телефон руководителя; ответственное лицо и его телефон; банк арендатора и номер счета в этом банке; ИНН арендатора.

Регистрация в БД договоров на аренду должна осуществляться с учетом следующих реквизитов по каждому договору: номер договора и дата его подписания; ответственный от агентства; общая стоимость договора; вид оплаты (еженедельная, ежемесячная и др.); дополнительные условия; перечень арендуемых рекламных щитов с указанием по каждому из них: номера щита, срока аренды (начало и конец), стоимости аренды, фотографии щита с рекламой. В случае необходимости изготовления рекламы оформляется дополнительный перечень работ с указанием стоимости по каждой позиции.

* + 1. Программа предназначена для хранения и редактирования информации о пользователях, рекламных щитах и арендаторах. Также программа выводит результат работы на экран и в текстовый файл.
    2. Программа используется сотрудниками для получения, хранения и изменения информации клиентов рекламного агентства.
    3. Периодичность решения задачи по запросу пользователя.
    4. Прекращение автоматизированного решения задачи происходит при отключении источника электропитания ЭВМ.
    5. Связь с другими задачами отсутствует.
    6. Специальных ограничений на временные характеристики решения задачи не налагается.
    7. Специальных требования на уровень подготовки пользователя не налагаются. Но лицо, работающее с программой, должно иметь минимальное представление о компьютере (знание необходимых операций).

## 1.2 Описание процессов (процедур)

Основные процессы в системе

1. Учет клиентов
   * Регистрация новых клиентов.
   * Хранение информации о клиентах (данные компании, контактные данные, банковские данные).
2. Управление арендой
   * Заключение договора на аренду рекламных щитов.
   * Проверка занятости рекламных щитов.
   * Отмена или перенос аренды.
3. Учет услуг и заказов
   * Список доступных услуг.
   * Формирование договора на аренду.
   * Расчет стоимости аренды.
4. Оптимизация процессов работы рекламного агентства
   * Автоматизация учета свободных рекламных щитов.
   * Генерация отчетов об аренде рекламных щитов.

Основные алгоритмы и процедуры

1. Алгоритм регистрации сотрудников
   * Проверка существующего сотрудника в базе.
   * Добавление новой записи при необходимости.
   * Подтверждение регистрации.
2. Алгоритм заключения договора на аренду
   * Выбор рекламного щита, срока аренды, начала аренды.
   * Проверка доступных рекламных щитов в назначенный срок
   * Подтверждение аренды и заключение договора.
3. Алгоритм расчета стоимости аренды
   * Определение стоимости выбранных услуг.
   * Финальный расчет суммы заказа.
4. Алгоритм генерации отчетов
   * Анализ выполненных заказов за определенный период.
   * Формирование итоговых статистических данных.

Цифровизация и автоматизация

* Введение базы данных для удобного хранения информации.
* Оптимизация рабочих процессов для уменьшения времени обслуживания клиентов.
* Использование программного обеспечения для повышения эффективности управления рекламным агентством.

## **1.3 Перечень входных и выходных данных (документов)**

Входные данные (вводимые пользователем):

1. Данные о клиентах:
   * ФИО / название компании
   * Контактная информация (телефон, адрес)
   * Банковские реквизиты
2. Информация о заказах:
   * Клиент (идентификатор в системе)
   * Рекламный щит
   * Даты начала и окончания размещения
   * Стоимость заказа
3. Учет рекламных площадей:
   * Местоположение (адрес)
   * Полезная площадь
   * Доступность (занято, свободно, в резерве)
   * Стоимость аренды
   * Фото или схема размещения
4. Финансовые операции:
   * Оплата заказов (сумма, дата, вид оплаты)
   * Выставленные счета
   * История транзакций
5. Аналитические данные:
   * Количество заказов по клиентам
   * Доходность рекламных площадок

Выходные данные (результаты работы системы):

1. Отчеты по заказам:
   * Список заказов
   * История заказов по клиентам
   * Финансовый статус (вид оплаты)
2. Отчеты по аренде рекламных площадей:
   * Занятость рекламных площадок
   * Доходность конкретных рекламных носителей
3. Финансовые отчеты:
   * Выставленные счета и платежи
4. Экспорт документов:
   * Генерация отчетов в Word, Excel
   * Выгрузка списка клиентов, заказов и платежей

## 1.4 Список пользователей (ролей)

Конструктор класса mainForm с параметрами login и role

* Конструктор принимает логин и роль пользователя, авторизованного в системе.
* InitializeComponent() – стандартный метод, генерируемый Visual Studio, который инициализирует визуальные элементы формы (загружает компоненты, созданные в дизайнере).
* InitializeRoleSpecificControls() – вызывается сразу после инициализации формы, чтобы настроить элементы управления в зависимости от роли пользователя.

Метод InitializeRoleSpecificControls()

Этот метод определяет, какие вкладки и элементы управления должны быть видны или доступны для текущего пользователя.

Блок if (\_role == "admin")

1. Проверка: если роль равна «admin», значит пользователь – администратор.
2. Создаётся новый объект employeesControl (UserControl, который, показывает список сотрудников).

Таким образом, администратор видит вкладку «Сотрудники» и может управлять списком сотрудников.

Блок else (для пользователей без роли «admin»)

1. Если роль не «admin», то для уточнения уровня доступа выполняется запрос к базе данных MySQL.
2. Подключение создаётся на основании строки "server=localhost;port=3306;username=root;password=root;database=advertising\_agency":

* server=localhost;port=3306 – локальный сервер MySQL на порту 3306;
* username=root;password=root – учётные данные для подключения;
* database=advertising\_agency – целевая база данных.

1. connection.Open() – устанавливает соединение.
2. Формируется MySqlCommand с запросом:

SELECT permissions

FROM user\_registration

WHERE login = @login

* Параметр @login связывается со значением \_login, чтобы получить строку permissions для данного логина.

1. command.ExecuteScalar() возвращает первое поле первой строки результата запроса или null, если ничего не найдено.
2. Если permissions = "0000", выполняется EmployeesPage.Enabled = false;, то есть вкладка «Сотрудники» (если она была добавлена ранее) становится недоступной (пользователь не может в неё войти).

Логика ограничения прав (role, permissions)

1. role – это более общее понятие. Значения – «admin», «user».
2. permissions – это детализированные биты или коды, определяющие, что конкретно может/не может делать пользователь. Например, «0000» выдается по умолчанию, что означает «нет прав на просмотр» вкладки «Сотрудники».

Таким образом, когда пользователь не является администратором, система проверяет конкретный уровень его полномочий в базе данных. Если permissions = «0000», вкладка для управления сотрудниками становится неактивной.

Класс DirectoriesControl — это пользовательский элемент управления (UserControl), позволяющий:

1. Отображать список доступных таблиц в выпадающем списке (ComboBox).
2. Загружать и отображать данные выбранной таблицы в DataGridView.
3. Учитывать уровень доступа (permissions) пользователя (чтение, добавление, редактирование, удаление).
4. Сохранять изменения в базе данных (insert/update/delete) при помощи MySqlDataAdapter.

Проверка прав доступа: CheckPermissions()

1. Открывает соединение с базой данных MySQL по заданной строке.
2. Выполняет запрос: SELECT permissions FROM user\_registration WHERE login = @login.
3. Считывает значение поля permissions. По коду:
   * **"0000"** – никаких прав (контрол и всё его содержимое отключается).
   * **"1000"** – только чтение (запрещены кнопки «Сохранить» и «Удалить», dataGridView1 в режиме ReadOnly = true).
   * **"1100"** – чтение и добавление (можно сохранять новые записи, но нельзя удалять).
   * **"1110"** – чтение + добавление + изменение.
   * **"1111"** – полный доступ (разрешено всё, включая удаление строк).

Таким образом, на базе хранимого в БД permissions контрол динамически изменяет доступность элементов.

LoadTableList()

1. Если пользователь не «admin», удаляет из списка пункты «user\_registration» и «employee», то есть эти таблицы будут недоступны в списке для обычных пользователей.
2. Затем устанавливается DataSource для comboBoxTableList — наш tableMapping.
3. DisplayMember и ValueMember указывают, какие поля TableItem привязываются к отображаемому и программному значению.
4. Наконец, устанавливается SelectedIndex = 0, чтобы по умолчанию выбрать первую таблицу (если элементы есть).

# 2. Проектирование базы данных

## 2.1 Концептуальная модель (ER-диаграмма)

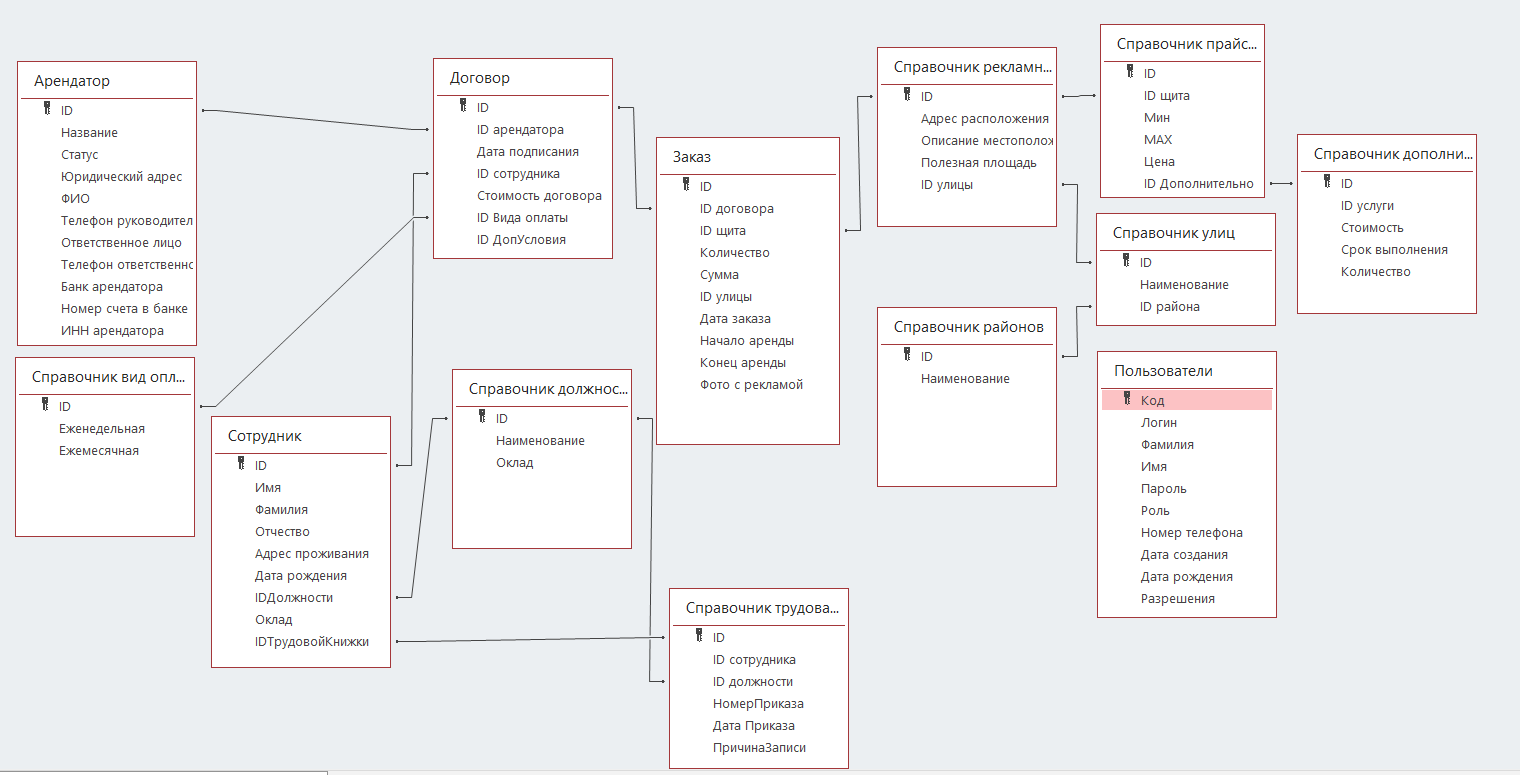


Рисунок 2.1 – Концептуальная модель

Основные сущности и их атрибуты:

1. Арендатор – хранит данные о клиентах, арендующих рекламные щиты.
   * ID – уникальный идентификатор.
   * Название – наименование арендатора.
   * Статус – текущий статус (активный, неактивный и т. д.).
   * Юридический адрес, ФИО, Телефон руководителя, Ответственное лицо – контактная информация.
   * Банк арендатора, Номер счета в банке – банковские реквизиты.
   * ИНН арендатора – идентификационный номер налогоплательщика.
2. Сотрудник – содержит информацию о работниках агентства.
   * ID – уникальный идентификатор сотрудника.
   * Имя, Фамилия, Отчество.
   * Адрес проживания, Дата рождения.
   * Должность, Оклад.
   * IDТрудовойКнижки – связь с трудовой книжкой.
3. Договор – фиксирует информацию о соглашениях с арендаторами.
   * ID – идентификатор договора.
   * ID арендатора – связь с таблицей Арендатор.
   * Дата подписания.
   * ID сотрудника – связь с таблицей Сотрудник (кто оформил договор).
   * Стоимость договора, ID вида оплаты, Доп. условия.
4. Заказ – хранит данные о заказах на размещение рекламы.
   * ID – уникальный номер заказа.
   * ID договора – связь с таблицей Договор.
   * ID щита – связь с таблицей рекламных площадей.
   * Количество, Сумма.
   * Дата заказа, Начало аренды, Конец аренды.
   * Фото с рекламой.
5. Пользователи – учетные записи сотрудников системы.
   * Код – уникальный идентификатор пользователя.
   * Логин, Фамилия, Имя, Пароль.
   * Роль – права доступа (администратор, менеджер и т. д.).
   * Номер телефона, Дата создания, Дата рождения, Разрешения.

Справочники (нормализация данных)

Эти таблицы хранят фиксированные значения, что позволяет избегать дублирования данных:

1. Справочник видов оплаты (ID, Ежедневная, Ежемесячная).
2. Справочник рекламных щитов (ID, Адрес, Описание, ID улицы).
3. Справочник цен (прайс-лист) (ID, ID щита, ММХ, Площадь, Цена).
4. Справочник улиц (ID, Наименование, ID района).
5. Справочник районов (ID, Наименование).
6. Справочник дополнительных услуг (ID, ID услуги, Стоимость, Срок выполнения, Количество).
7. Справочник должностей (ID, Наименование, Оклад).
8. Справочник трудовых книжек (ID сотрудника, ID должности, Номер приказа, Дата приказа, Причина записи).

Взаимосвязи между таблицами:

* Арендатор ↔ Договор – каждый договор привязан к конкретному арендатору.
* Договор ↔ Заказ – заказ оформляется в рамках определенного договора.
* Заказ ↔ Рекламные щиты – указывает, на каком рекламном щите размещена реклама.
* Сотрудник ↔ Договор – фиксирует, кто заключил договор.
* Сотрудник ↔ Справочник должностей – содержит данные о должности и зарплате.
* Рекламные щиты ↔ Улицы ↔ Районы – помогает определять местоположение рекламных площадей.
* Пользователи – связаны с системой для управления доступом.

Rental

Хранит информацию о клиентах, заключающих договоры на размещение рекламы (таблица 1).

Таблица 1. Описание таблицы Rental

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор арендатора (первичный ключ). |
| name | VARCHAR(100) | Название компании или ФИО частного лица. |
| status | VARCHAR(100) | Статус арендатора (ООО, ИП и др.) |
| office | VARCHAR(100) | Юридический/фактический адрес. |
| FIO | VARCHAR(100) | ФИО руководителя. |
| phonenumber1 | BIGINT(15) | Основной контактный номер телефона. |
| responsible | VARCHAR(100) | ФИО ответственного лица. |
| phonenumber2 | BIGINT(15) | Контактный номер телефона ответственного лица |
| bank | VARCHAR(100) | Название банка арендатора. |
| bank\_account | INT(11) | Банковские реквизиты |
| INN | INT(12) | Идентификационный номер налогоплательщика. |

Typepay

Содержит информацию о вариантах оплаты рекламных услуг.

Таблица 2. Описание таблицы Typepay

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор. |
| type | VARCHAR(100) | |  | | --- | |  |  |  | | --- | | Тип оплаты (ежемесячная, разовая и т. д.). | |

Contract

Хранит информацию о договорах между агентством и арендаторами.

Таблица 3. Описание таблицы Contract

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор договора. |
| id renter | INT(11) | Ссылка на арендатора (rental). |
| dateorder | TIMESTAMP | |  | | --- | |  |  |  | | --- | | Дата заключения договора. | |
| id employee | INT(11) | Ссылка на сотрудника (employee), ответственного за договор. |
| cost | INT(11) | |  | | --- | |  |  |  | | --- | | Общая стоимость договора. | |
| id typepay | INT(11) | Ссылка на тип оплаты (typepay). |
| id optional | INT(11) | |  | | --- | |  |  |  | | --- | | Ссылка на дополнительные услуги (optional). | |

Order

Хранит информацию о рекламных заказах.

Таблица 4. Описание таблицы Order

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) UNSIGNED | Уникальный идентификатор заказа. |
| contract\_id | INT(11) | Ссылка на договор (contract). |
| billdoard\_id | INT(11) | Ссылка на рекламный щит (billboards). |
| count | INT(11) | Количество заказанных мест. |
| cost | INT(11) | Стоимость заказа. |
| street\_id | INT(11) | Ссылка на улицу (street). |
| date\_order | TIMESTAMP | |  | | --- | |  |  |  | | --- | | Дата размещения заказа. | |
| startdate | DATE | Дата начала аренды. |
| enddate | DATE | |  | | --- | |  |  |  | | --- | | Дата завершения аренды. | |
| pictures | VARCHAR(250) | Ссылка на изображение рекламы. |

Billboards

Хранит информацию о местах размещения рекламы.

Таблица 5. Описание таблицы Billboards

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор щита. |
| address | VARCHAR(100) | Адрес расположения. |
| location description | VARCHAR(100) | Краткое описание местоположения. |
| usable area | FLOAT | Доступная площадь для размещения. |
| id street | INT(11) | Ссылка на улицу (street). |

Pricelist

Содержит информацию о стоимости аренды рекламных мест.

Таблица 6. Описание таблицы Pricelist

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор. |
| id billboard | INT(11) | |  | | --- | |  |  |  | | --- | | Ссылка на рекламный щит (billboards). | |
| cost | INT(11) | Стоимость аренды. |
| id optional | INT(11) | |  | | --- | |  |  |  | | --- | | Ссылка на дополнительные услуги (optional). | |

Optional

Содержит информацию о дополнительных услугах агентства.

Таблица 7. Описание таблицы Optional

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор. |
| service | VARCHAR(100) | Наименование услуги. |
| cost | INT(11) | Стоимость услуги. |
| term | DATE | Срок окончания выполнения услуги. |
| count | INT(11) | |  | | --- | |  |  |  | | --- | | Количество единиц услуги. | |

Employee

Содержит данные о сотрудниках рекламного агентства.

Таблица 8. Описание таблицы Employee

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор. |
| name | VARCHAR(100) | Имя. |
| surname | VARCHAR(100) | Фамилия. |
| residental address | VARCHAR(100) | Адрес проживания. |
| datebirth | DATE | Дата рождения. |
| id post | INT(11) | Должность (из post). |
| salary | INT(11) | Оклад. |
| id work record | INT(11) | Ссылка на трудовую запись (work record). |

Post

Хранит данные о должностях сотрудников.

Таблица 9. Описание таблицы Post

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT(11) | Уникальный идентификатор. |
| title | VARCHAR(100) | |  | | --- | |  |  |  | | --- | | Название должности. | |
| salary | INT(11) | Оклад. |

User\_registration

Управляет аутентификацией пользователей в системе.

Таблица 10. Описание таблицы User\_registration

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | |  | | --- | |  |  |  | | --- | | INT(11) UNSIGNED | | Уникальный идентификатор. |
| login | VARCHAR(20) | Логин |
| surname | VARCHAR(20) | Фамилия |
| username | VARCHAR(50) | Имя |
| password\_hash | VARCHAR(255) | Хеш пароля |
| role | ENUM('admin','user') | Роль пользователя |
| number | BIGINT(15) | Номер телефона |
| created\_at | TIMESTAMP | Дата регистрации |
| date of birth | |  | | --- | |  |  |  | | --- | | DATE | | Дата рождения |
| permissions | VARCHAR(4) | Права доступа |

# 3 Выбор и обоснование инструментальных средств для разработки программного обеспечения

## 3.1 Выбор системы управления базами данных

Сравнение СУБД представлено на таблица 11.

Таблица 11. Сравнение систем управления базами данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерии | MySQL | PostgreSQL | Microsoft SQL Server | SQLite | MongoDB |
| Модель данных | Реляционная, мульти-модель | Реляционная, мульти-модель | Реляционная, мульти-модель | Реляционная | Документная, мульти-модель |
| Поддержка ACID | Частичная (InnoDB) | Полная | Полная | Частичная | Частичная (начиная с версии 4.0, поддержка многодокументных транзакций) |
| Типизация данных | Строгая | Строгая | Строгая | Гибкая | Гибкая |
| Производительность | Высокая для операций чтения | Высокая для чтения и записи, поддержка сложных транзакций | Высокая, особенно в корпоративных средах | Высокая для небольших объемов данных | Высокая для неструктурированных данных |
| Масштабируемость | Вертикальная, ограниченная горизонтальная | Вертикальная и горизонтальная | Вертикальная | Не предназначена для масштабирования | Горизонтальная, разработана для распределенных систем |
| Лицензия | GPL для Community Edition; проприетарная для Enterprise | Открытая (PostgreSQL License) Проприетарная (Microsoft) | Проприетарная (Microsoft) | Открытая (Public Domain) | Открытая (SSPL для серверного ПО) |
| Поддержка ОС | До 64 ТБ | Ограничений нет (зависит от конфигурации системы) | Зависит от лицензии (от 10 ГБ до нескольких ПБ) | До 140 ТБ в одном файле | Ограничений нет (зависит от объема диска и конфигурации) |
| Конкурентность | Хорошо работает с операциями чтения; возможны проблемы при высокой конкуренции на запись | Эффективно обрабатывает одновременные транзакции | Хорошо работает с одновременными запросами | Подходит для приложений с низкой конкуренцией | Хорошо справляется с конкурентной нагрузкой |
| Управление пользователями | Есть (роль и доступ) | Есть (гибкие настройки ролей и прав доступа) | Расширенные возможности управления доступом | Нет встроенного управления пользователями | Есть (гибкая настройка доступа к коллекциям) |
| Сложность настройки | Простая установка и настройка | Сложнее, требует знаний для оптимизации | |  | | --- | | Сложная настройка, рекомендуется для профессионалов |  |  | | --- | |  | | Очень простая | Средняя, требует понимания документной модели данных |
| Поддержка JSON | Частичная (JSON и JSONB, начиная с версии 5.7) | Полная (JSONB для индексации и быстродействия) | Полная (JSON) | Отсутствует | Основной формат хранения данных |
| Основное назначение | Веб-приложения, CMS, форумы | Финансовые системы, аналитика, научные приложения | Корпоративные приложения, аналитика, сложные транзакции | Встроенные системы, мобильные приложения | Системы управления контентом, аналитика данных, приложения с большим объемом данных |
| Поддержка сообщества | Широкая | Активное сообщество разработчиков и экспертов | Профессиональная поддержка от Microsoft | Ограниченная, но доступная документация | Активное сообщество, ориентированное на разработчиков |
| Пример использования | Веб-сайты (WordPress, Joomla), CRM | Банковские системы, системы учета | Бизнес-приложения, решения для больших данных | Локальные мобильные приложения, тестовые базы данных | Аналитические системы, платформы обработки данных в реальном времени |

MySQL: Подходит для приложений, где приоритет отдается чтению данных, таких как веб-сайты, системы управления контентом. Простота настройки делает его популярным выбором для небольших и средних проектов.

PostgreSQL: Идеален для сложных транзакционных систем, требующих надежности и поддержки ACID. Подходит для финансовых приложений и систем, где важна масштабируемость.

Microsoft SQL Server: Лучший выбор для корпоративных клиентов, уже использующих продукты Microsoft. Отлично справляется с высоконагруженными транзакциями.

SQLite: Простой и легковесный выбор для мобильных приложений, встроенных систем или небольших проектов. Не рекомендуется для высоконагруженных систем.

MongoDB: Оптимален для приложений с неструктурированными данными, систем управления контентом и аналитики, где важна гибкость данных и горизонтальная масштабируемость.

Достоинства и недостатки баз данных

MySQL

Преимущества:

* Широкое распространение и поддержка: MySQL является одной из самых популярных реляционных СУБД с обширным сообществом, документацией и множеством сторонних инструментов.
* Высокая производительность: Оптимизирована для операций чтения, что делает ее идеальной для приложений, где чтение данных преобладает над записью.
* Масштабируемость: Поддерживает горизонтальное масштабирование через шардинг.
* Интеграция: Легко интегрируется с различными платформами, включая PHP, WordPress и популярные веб-хостинги.
* Бесплатность: Community-версия MySQL доступна бесплатно.

Недостатки:

* Ограниченная поддержка продвинутых функций: Некоторые расширенные функции, такие как полная поддержка ACID в определенных конфигурациях, могут быть ограничены по сравнению с PostgreSQL.
* Лицензирование: Некоторые расширенные функции доступны только в коммерческой версии.
* Проблемы с конкурентностью: Может испытывать трудности при высоком уровне одновременных записей, требуя настройки или выбора другой СУБД для приложений с интенсивной записью.

PostgreSQL

Преимущества:

* Расширенные функции: Полная поддержка ACID, продвинутая индексация, полнотекстовый поиск и поддержка пользовательских типов данных.
* Расширяемость: Возможность добавления пользовательских функций, типов данных и даже языков.
* Соответствие стандартам: Строгое соответствие стандартам SQL, обеспечивающее совместимость с широким спектром приложений и инструментов.
* Управление конкурентностью: Эффективно обрабатывает высокий уровень одновременных транзакций, что делает ее подходящей для сложных приложений с интенсивной записью.
* Открытый исходный код: Полностью открытая без каких-либо проприетарных лицензий, все функции доступны бесплатно.

Недостатки:

* Производительность: Может требовать больше ресурсов и настройки для достижения оптимальной производительности по сравнению с MySQL, особенно для простых операций чтения.
* Сложность: Расширенные функции и гибкость могут усложнить настройку и обслуживание, особенно для начинающих.

Microsoft SQL Server

Преимущества:

* Интеграция с продуктами Microsoft: Бесшовная интеграция с другими продуктами Microsoft, такими как .NET и Azure.
* Производительность: Высокая производительность для сложных запросов и транзакционных рабочих нагрузок.
* Безопасность: Расширенные функции безопасности, такие как прозрачное шифрование данных и безопасность на уровне строк.
* Аналитические возможности: Встроенные функции для аналитики и бизнес-анализа.

Недостатки:

* Лицензирование: Коммерческая лицензия может быть дорогой, особенно для небольших предприятий.
* Платформенная зависимость: Основная поддержка для Windows, ограниченная поддержка для других операционных систем.

SQLite

Преимущества:

* Легковесность и простота: Сервер не требуется, что делает ее идеальной для встроенных приложений и небольших проектов.
* Портативность: Вся база данных хранится в одном файле, что упрощает перенос и резервное копирование.
* Быстрота для небольших приложений: Оптимизирована для небольших наборов данных, обеспечивая высокую скорость чтения и записи.
* Отсутствие необходимости в сервере: Уменьшает накладные расходы для небольших приложений.

Недостатки:

* Ограниченная масштабируемость: Не предназначена для высококонкурентных или крупных приложений; лучше всего подходит для небольших и средних проектов с легкой нагрузкой.
* Ограниченная функциональность: Отсутствие некоторых расширенных функций, таких как хранимые процедуры и управление пользователями.
* Проблемы с конкурентностью: Обрабатывает операции записи последовательно, что может привести к узким местам в приложениях с интенсивной записью.

MongoDB

Преимущества:

* Гибкость схемы: Как NoSQL база данных, позволяет гибкие, бессхемные модели данных, что делает ее идеальной для неструктурированных или полуструктурированных данных.
* Масштабируемость: Разработана для горизонтального масштабирования, что упрощает распределение данных по нескольким серверам.
* Документы, похожие на JSON: Использует формат BSON (Binary JSON), что интуитивно понятно для разработчиков, работающих с JSON в своих приложениях.
* Высокая производительность для определенных случаев использования: Хорошо работает в сценариях, требующих быстрых, масштабируемых операций чтения и записи, таких как аналитика в реальном времени и системы управления контентом.

Недостатки:

* Отсутствие ACID-транзакций: Хотя поддерживает многодокументные транзакции, они не так надежны, как в реляционных базах данных, таких как PostgreSQL, особенно в сложных транзакциях.
* Консистентность данных: Гибкость может привести к несоответствиям, если не управлять ими тщательно, особенно в больших распределенных системах.
* Использование памяти: Может быть ресурсоемкой, особенно при работе с большими наборами данных, так как пытается держать рабочие наборы в памяти

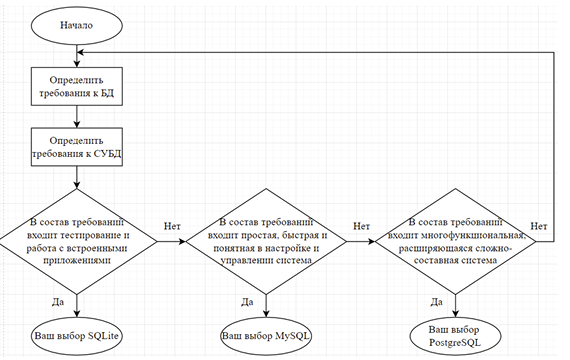
****

Рисунок 3.1 – Алгоритм выбора СУБД

MySQL

1. Простота использования и настройки

MySQL проста в установке и настройке, что позволяет быстро запустить проект. Большое количество руководств, статей и форумов помогает легко найти ответы на возникающие вопросы.

2. Высокая производительность

Рекламное агентство часто работает с отчетами, аналитикой и презентацией данных клиентам. MySQL обеспечивает отличную производительность для операций чтения. MySQL поддерживает InnoDB, который предоставляет высокую производительность для транзакций и гарантирует соблюдение ACID.

3. Широкая поддержка экосистемы

MySQL легко интегрируется с веб-приложениями, CRM-системами и BI-инструментами, которые часто используются в рекламной отрасли. MySQL поддерживает все популярные языки разработки (PHP, Python, Java, .NET), что упрощает разработку приложений для управления рекламной деятельностью.

4. Широкое сообщество и доступность специалистов

Благодаря широкому распространению, найти специалистов, знающих MySQL, значительно проще, чем для других СУБД. Доступность учебных материалов позволяет быстро обучить сотрудников, если это потребуется.

5. Масштабируемость и производительность

MySQL поддерживает горизонтальное масштабирование через репликацию, что позволяет системе расти с увеличением объема данных. MySQL отлично справляется с базами данных небольшого и среднего размера, что идеально подходит для рекламного агентства.

6. Экономичность

Community Edition предоставляет большинство необходимых функций бесплатно, что снижает затраты на лицензирование. Благодаря низким системным требованиям, MySQL позволяет сэкономить на серверном оборудовании.

7. Функции, полезные для рекламного агентства

MySQL поддерживает индексацию, что ускоряет запросы, важные для рекламных отчетов. Инструменты для расчетов и аналитики позволяют быстро подготавливать данные для отчетов о рекламных кампаниях. Начиная с версии 5.7, MySQL поддерживает хранение JSON-данных, что полезно для работы с метаданными рекламных кампаний или настроек.

MySQL подойдет, если:

* Основная нагрузка приходится на чтение данных (например, анализ и отчеты по кампаниям).
* Требуется быстрое внедрение и простое управление базой данных.
* Объем данных и количество пользователей находятся в пределах средней нагрузки.
* Если рекламное агентство планирует масштабировать свои операции или работать с большими объемами структурированных данных, MySQL может быть удачным выбором как стабильное, производительное и экономичное решение.

Для разрабатываемой информационной системы рекламного агентства оптимальным выбором является MySQL по следующим причинам:

1. Производительность – MySQL работает быстрее при обработке простых и средне сложных запросов, что важно при управлении заказами, клиентами и рекламными площадями.
2. Простота интеграции – MySQL легко интегрируется с языками программирования, такими как Python (Django, Flask), Java (Spring Boot), PHP (Laravel).
3. Поддержка транзакций – Использование InnoDB позволяет гарантировать целостность данных при операциях с договорами и платежами.
4. Масштабируемость – СУБД поддерживает репликацию и кластеризацию, что позволяет расширять систему при увеличении нагрузки.
5. Широкая распространенность – Доступно множество обучающих материалов и документации, что упрощает разработку и поддержку системы.

Таким образом, MySQL является оптимальным выбором для хранения и управления данными в системе рекламного агентства.

## 3.2 Выбор языка программирования

При выборе языка программирования для разработки программного обеспечения необходимо учитывать такие факторы, как производительность, удобство разработки, поддержка баз данных, безопасность и масштабируемость. Среди доступных вариантов **C#** является одним из наиболее подходящих языков для разработки корпоративных приложений, в том числе систем управления рекламными агентствами.

Преимущества использования C# для разработки данной системы

Высокая производительность

* C# работает быстрее, чем интерпретируемые языки (например, Python или PHP), благодаря компиляции в промежуточный код (MSIL), который затем выполняется средой .NET.
* Это особенно важно при обработке больших объемов данных, управлении заказами и аналитике.

Интеграция с Microsoft SQL Server и другими СУБД

* C# идеально подходит для работы с Microsoft SQL Server (MS SQL), который обеспечивает высокую надежность, безопасность и поддержку транзакций.
* Поддержка других популярных СУБД, включая MySQL и PostgreSQL, через Entity Framework (EF Core).

Использование мощного фреймворка .NET Core / .NET 6+

* Разработка на .NET Core (кроссплатформенный) или .NET 6/7 (универсальная среда разработки).
* Возможность разработки веб-приложений (ASP.NET Core MVC, Blazor, Web API) и настольных приложений (Windows Forms, WPF).
* Поддержка асинхронного программирования (async/await) для обработки большого количества запросов.

Широкие возможности для веб-разработки

* C# в сочетании с ASP.NET Core позволяет легко создавать многопользовательские веб-системы с поддержкой REST API.
* Легкая интеграция с JavaScript (React, Angular, Vue.js) на клиентской стороне.

Безопасность

* Встроенные механизмы защиты, такие как контроль типов, управление памятью, защита от SQL-инъекций при использовании ORM.
* Интеграция с Active Directory для управления пользователями в корпоративной среде.

Масштабируемость и поддержка облачных технологий

* Легкая интеграция с Microsoft Azure, что позволяет развертывать облачные решения.
* Возможность работы в контейнерах (Docker, Kubernetes) для обеспечения отказоустойчивости.

Развитая экосистема инструментов

* Visual Studio / Rider – мощные IDE для разработки.
* Entity Framework (EF Core) – удобный ORM для работы с базой данных.
* SignalR – поддержка реального времени (например, отслеживание изменений заказов, уведомления).

Таким образом, использование C# в сочетании с .NET Core / ASP.NET Core и Entity Framework обеспечивает надежную, безопасную и производительную систему управления рекламными процессами.

## 3.3 Выбор интегрированной среды разработки

При разработке программного обеспечения для рекламного агентства на языке C# важно выбрать интегрированную среду разработки (IDE), которая обеспечит удобство кодирования, отладку, поддержку баз данных и инструментов управления проектами.(Таблица 12)

Таблица 12. Анализ популярных IDE для C#

|  |  |  |
| --- | --- | --- |
| **IDE** | **Преимущества** | **Недостатки** |
| **Visual Studio (VS)** | Полная поддержка C# и .NET,  Встроенные инструменты для работы с базами данных (SQL Server),  Отличный встроенный отладчик,  Поддержка ASP.NET Core, Blazor, WinForms и WPF,  Интеграция с Azure | Высокие системные требования,  Полная версия требует лицензии |
| **JetBrains Rider** | Высокая производительность,  Отличные инструменты рефакторинга кода,  Кроссплатформенность (Windows, Linux, macOS),  Интеграция с Docker, Kubernetes, GitHub | Не имеет встроенного GUI-дизайнера для WinForms/WPF,  Ограниченные возможности для больших проектов |
| **Visual Studio Code (VS Code)** | Легкость и быстродействие,  Поддержка C# через расширение Omnisharp,  Поддержка Git, Docker,  Бесплатна и кроссплатформенна | Не имеет встроенного GUI-дизайнера для WinForms/WPF,  Ограниченные возможности для больших проектов |
| **MonoDevelop** | Подходит для кроссплатформенной разработки (Linux, macOS),  Бесплатная | Ограниченный функционал,  Не так активно обновляется, как Visual Studio или Rider |

Обоснование выбора IDE

Для разработки программного обеспечения рекламного агентства на C# лучшими вариантами являются Visual Studio и JetBrains Rider.

Если проект ориентирован на Windows и Microsoft-технологии (SQL Server, ASP.NET, Blazor, WinForms, WPF):  
Visual Studio – оптимальный выбор. Он обладает встроенной поддержкой всех инструментов разработки на C#, лучшим отладчиком и инструментами профилирования.

Если разработка кроссплатформенная и требуется высокая производительность:  
JetBrains Rider – мощная IDE с отличными инструментами анализа кода и поддержки работы с Docker, Kubernetes и облачными сервисами.

Если важна легкость и бесплатность:  
Visual Studio Code – хороший вариант для веб-разработки на ASP.NET Core + Blazor, но менее удобен для работы с WinForms/WPF.

Оптимальным выбором для данного проекта является Visual Studio (предпочтительно Visual Studio 2022 Community – бесплатная версия для индивидуальных разработчиков и малого бизнеса). Эта среда поддерживает все инструменты для создания веб- и настольных приложений, интеграцию с базами данных, облачными сервисами и API, что делает её лучшим решением для разработки программного обеспечения рекламного агентства.

# 4 Программная реализация информационной системы

1. Файл change\_pass.cs

Класс change\_pass отвечает за изменение пароля текущего пользователя.

* private void new\_pass\_TextChanged(object sender, EventArgs e)

Обработчик события изменения текста в поле для нового пароля. Может использоваться для валидации вводимых данных.

* private void new\_pass\_Enter(object sender, EventArgs e) / private void new\_pass\_Leave(object sender, EventArgs e)

Обработчики событий при получении/потере фокуса полем для пароля. Используются для вывода подсказок или очистки плейсхолдера.

* private void ChangeButton\_Click(object sender, EventArgs e)

Обработчик нажатия на кнопку «Изменить пароль». Внутри происходит:

* 1. Создание объекта для работы с БД (new DB()).
  2. Формирование MySqlCommand с SQL-запросом на обновление пароля в таблице user\_registration.
  3. Переход к форме mainForm() при успешном обновлении.
* private void backbutton\_Click(object sender, EventArgs e)

Обработчик кнопки «Назад». Возвращает пользователя на предыдущую форму.

2. Файл DataBaza.cs

Класс служит для подключения к базе данных.

* new MySqlConnection(...)

Создаёт подключение к MySQL (server=localhost;port=3306;...).

* public void openConnection()

Открывает текущее подключение к БД.

* public void closeConnection()

Закрывает текущее соединение.

* public MySqlConnection getConnection()

Возвращает объект MySqlConnection, чтобы другие классы могли работать с ним (выполнять запросы, и т. д.)

3. Файл DirectoriesControl.cs

Отвечает за работу с «справочниками» (directories) — то есть таблицами БД, доступными для просмотра и редактирования. Есть методы, связанные с отображением списка таблиц, загрузкой данных, сохранением изменений.

* public TableItem(string tableName, string displayName)  
  Предназначен для хранения пары (имя таблицы в БД, название для отображения в ComboBox).
* public DirectoriesControl(string login, string role)  
  Конструктор пользовательского элемента управления (Control) для работы со справочными таблицами. Сохраняет логин и роль текущего пользователя. На основе роли (admin/user) изменяются права.
* private void InitializeTableMapping()

Создаёт список соответствий (tableName → DisplayName), чтобы в ComboBox пользователь видел русские названия вместо английских.

* private void LoadTableList()

Загружает перечень таблиц для ComboBox. Если роль не «admin», может скрывать некоторые таблицы.

* private void ComboBoxTableList\_SelectedIndexChanged(object sender, EventArgs e)

Реакция на выбор пользователем конкретного справочника в ComboBox. Вызывает метод загрузки данных.

* private void LoadTableData(string tableName)

Подгружает (через MySqlDataAdapter) строки из выбранной таблицы в DataGridView. Создаёт MySqlCommandBuilder для обеспечения возможности редактирования.

* private void btnSaveChanges\_Click(object sender, EventArgs e)

Сохраняет изменения, внесённые в DataGridView, обратно в базу данных (INSERT/UPDATE/DELETE).

4. Файл DocumentsControl.cs

Используется для формирования и экспорта некоторых документов на основе данных из БД.

* private void LoadData()

Загружает данные из таблиц (contract, order и т.п.) и отображает их. Через подключение к БД, MySqlCommand, MySqlDataAdapter и DataTable.

* private void (поле)\_Enter(...) / (поле)\_Leave(...)

Набор обработчиков для полей ввода (rentalText, employeeText и т. д.), Служат для подсказок/очистки плейсхолдера.

* private void buttonExport\_Click(object sender, EventArgs e)

По нажатию «Экспорт» формируется новая запись в contract и создаётся «чек» (документ Word). Логика:

* 1. Вставка новой строки в таблицу contract.
  2. Подсчёт общей стоимости (на базе заказов).
  3. Обновление поля cost в таблице contract.
  4. Генерация Word-файла (чека).
* private int InsertContract()

Создаёт новую запись в таблице contract и возвращает ID (через SELECT LAST\_INSERT\_ID()).

* private int GetOrderCost(int contractId), private string GetRenterName(int renterId), private string GetEmployeeName(int employeeId), и т.д.  
  Сбор сервисных методов для поиска/получения различных значений из базы (стоимость заказа, имя арендатора, имя сотрудника, цена доп. услуг и пр.).
* private bool UpdateContractCost(int contractId, int totalCost)

Обновляет общую стоимость в записи contract.

* private string GetOrderInfoForContract(int contractId)

Составляет детальное текстовое описание всех заказов, связанных с конкретным контрактом.

* private void ExportReceiptToWord(...)

Генерирует Word-документ (чек) и предлагает пользователю сохранить файл через SaveFileDialog.

5. Файл EmployeesControl.cs

Отвечает за отображение списка сотрудников и за присвоение прав пользователю.

* private void LoadData()

Подключается к базе, выполняет SELECT-запрос (с алиасами), заполняет DataSet, привязывает к DataGridView.

* private void SavePermissions\_Click(object sender, EventArgs e)

Обработчик нажатия на кнопку «Сохранить права доступа». Проверяет, что в dataGridView1 выбрана ровно одна строка (один пользователь). Если нет — выводит сообщение «Выберите пользователя». Из выбранной строки берёт значение столбца «Логин» (ключ пользователя). Формирует строку permissions на основе четырех флажков (чтение, добавление, изменение, удаление). Каждый флажок добавляет символ '1' (если включен) или '0' (если выключен). Например, «1010» может обозначать включённое чтение и изменение, но выключенные добавление и удаление. Открывает соединение с БД advertising\_agency, выполняет UPDATE user\_registration SET permissions = @permissions WHERE login = @login. Параметры @permissions и @login подставляются из собранной строки и выбранного логина. При успешном обновлении выводит «Права доступа обновлены», при ошибке — сообщение с текстом исключения.

6. Файл LoginForm.cs

Форма авторизации.

* static string HashPassword(string password)

Хеширует строку пароля с помощью SHA256.

* private void AddRegistration\_Click(object sender, EventArgs e)

Обработчик кнопки «Войти». Логика:

* 1. Создаём объект DB().
  2. Проверяем логин и пароль в таблице user\_registration.
  3. Если авторизация успешна, открывается mainForm(...).
* private string GetUserRole(string login)

Запрашивает из БД роль пользователя (admin или user).

* private void RegisterLabel\_Click(object sender, EventArgs e)

Переход к форме регистрации (RegisterForm()).

* private void ExitLabel\_Click(object sender, EventArgs e)

Закрытие приложения или возврат на предыдущий экран (в зависимости от логики).

7. Файл mainForm.cs

Главное окно приложения после авторизации.

* public mainForm(string login, string role)  
  Конструктор, принимает логин и роль. Внутри добавляются нужные вкладки (DirectoriesControl, EmployeesControl и т. д.) в зависимости от роли.
* private void ExitLabel\_Click(object sender, EventArgs e)

Выход из приложения.

8. Файл OrdersControl.cs

Управление заказами (строки таблицы order).

* private void LoadData()

Загружает заказы из таблицы order с помощью MySqlDataAdapter → DataTable.

* private bool Checker()

Проверка корректности заполнения полей (что введён номер договора, верная дата и т.д.).

* private void add\_button\_Click(object sender, EventArgs e)

Обработка нажатия на кнопку «Добавить».

* 1. Вызывает Checker();
  2. Проверяет, свободен ли рекламный щит (CheckBillboardBusy);
  3. Добавляет запись в таблицу order (INSERT).
* private bool CheckBillboardBusy(int billboardId, DateTime start, DateTime end)  
  Проверяет, не занят ли рекламный щит в указанный период. Делает запрос к БД SELECT COUNT(\*).
* private void deleteButton\_Click(object sender, EventArgs e)

Удаляет выбранную строку из таблицы order.

* private void change\_Click(object sender, EventArgs e)

Редактирует выбранную запись в таблице order. Снова вызывается проверка занятости.

* private void button\_word\_Click(object sender, EventArgs e)

Экспорт заказов в текстовый файл через SaveFileDialog. Сначала пишет заголовки столбцов, потом все строки из DataGridView.

9. Файл OtherControl.cs

Вспомогательный контрол, содержащий кнопки смены пароля и т. п.

* private void ChangePass\_Click(object sender, EventArgs e)

Открывает форму change\_pass (смена пароля).

* private void button2\_Click(...) / private void button1\_Click(...)

Дополнительные кнопки, возможно, для каких-то справочных функций или настроек (логика не детализирована в названии).

10. Файл RegisterForm.cs

Форма регистрации новых пользователей.

* static string HashPassword(string Password)

Хеширование введённого пароля (SHA256).

* private void Username\_TextChanged(...) / Login\_Enter(...) / Login\_Leave(...) и т.д.

Набор обработчиков для полей ввода (логин, пароль, номер телефона, дата рождения).

Используются для валидации, плейсхолдеров, вывода подсказок.

* private bool Check()

Проверяет, заполнены ли обязательные поля корректно.

* private void ButtonRegistration\_Click(object sender, EventArgs e)

Основная логика регистрации:

* 1. Создаёт объект для работы с БД.
  2. Формирует SQL-запрос INSERT INTO user\_registration ....
  3. Хеширует пароль перед записью.
  4. Переходит на форму авторизации.
* public Boolean CheckUser()

Проверяет, нет ли уже такого же логина или номера телефона в таблице user\_registration.

* private void ResetFormFields()

Сбрасывает поля формы после успешной регистрации.

* private void Labelfinish\_Click(...)

Возврат к форме LoginForm().

# Руководство пользователя

## Назначение и условия применения

Программа предназначена для хранения и обработки информации о клиентах.

Решение данной задачи призвано обеспечить:

– быстрый и удобный доступ для работы со списком клиентов;

– создание файла с расширение .txt, содержащего всю необходимую информацию о клиентах.

* + 1. Программа реализована на языке C# с использованием свободной IDE – Visual studio 2022. Работает на операционной системе Windows.

## Подготовка к работе

* + 1. Система состоит из исполнительного файла «kursach\_2\_0.exe.». Все файлы программы хранятся в папке «kursach\_2\_0». Данная система использует следующие файлы:

\*.txt – список информации о всех клиентах.

* + 1. Запуск программы

Программа поставляется на любом носителе.

Для установки программы нужно запустить инсталлятор kursach\_2\_0.exe. После чего разрешить установку всех компонентов. После завершения установки программа будет расположена в указанной папке и, если было выбрано создание ярлыка на рабочем столе, то там создастся файл «Рекламное агентство».

Запуск программы осуществляется двойным щелчком по ярлыку.

## Описание операций

* + 1. Введение

Основные функции системы соответствуют основным алгоритмам, используемых в программе. Для удобства пользователя и более легкого изучения системы большинство форм и диалогов имеют идентичный интерфейс. Далее описаны все функции системы, а также формы и диалоги для ввода информации. Для каждой формы приведены основные компоненты и их назначение.

## 5.4 Аварийные ситуации

К аварийным ситуациям относятся: потеря соединения с базой данных. В данном случае работа программы протекает неверно. Так же при возникновении вирусов на компьютере возможны сбои в работе программы.

# Руководство системного программиста

## Общие сведения

Программа «kursach\_2\_0.exe.» предназначена для хранения и обработки информации о клиентах. Также программа выводит результат работы на экран и в текстовый файл.

Входные и выходные данные подробно описаны в п.1 «Постановка задачи».

Результатом работы программы являются видеограммы и файлы.

Программа реализована на языке C# с использованием свободной IDE – Visual Studio 2022. Работает на операционной системе Windows. Также в программе используется база данных MySQL, которая обеспечивает хранение информации о пользователях и созданных ими абонентах.

Периодичность решения задача по запросу пользователя. Ответ на запрос осуществляется в диалоговом режиме.

Перед эксплуатацией системы необходимости в установке дополнительных пакетов программного обеспечения отсутствует.

## Структура программы

Программа реализована на языке C# с использованием свободной IDE – Visual studio 2022, основанном на визуальном построении приложений (помещение компонентов на формы и изменение их свойств и методов), так же через реализацию базы данных MySQL, поэтому некоторые функции формирования окон и отчетов невозможно описать в списке функций и листинге программы.

Руководство пользователя представлено в приложении А.

##### ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы были приобретены навыки по программированию на языке C# и разработке приложений с графическим интерфейсом при помощи Windows Forms. В результате была создана программа «kursach\_2\_0.exe.», которая позволяет хранить и обрабатывать данные о клиентах, обращавшихся в рекламное агентство. В процессе выполнения задания были изучены методики работы с базой данных MySQL, которая обеспечивает хранение информации о пользователях и созданных ими абонентах.

Для использования программа подойдет для любых пользователей, имеющих начальные знания в работе с компьютером. Данная программа защищена авторизацией от использования алгоритмов сторонними лицами и предназначена для автоматизации процесса хранения и обработки больших объемов данных, позволяет удобный формат хранения и редактирования клиентов рекламного агентства.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Гниденко, И. Г. Технология разработки программного обеспечения : учеб. пособие для СПО / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — М. : Издательство Юрайт, 2017. — 235 с.
2. Гордеев, С. И. Организация баз данных в 2 ч. Часть 2 : учебник для вузов / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2019. — 501 с.
3. Зыков, С. В. Программирование. Объектно-ориентированный подход : учебник и практикум для академического бакалавриата / С. В. Зыков. — М. : Издательство Юрайт, 2019. — 155 с.
4. Кудрина, Е. В. Основы алгоритмизации и программирования на языке c# : учеб. пособие для бакалавриата и специалитета / Е. В. Кудрина, М. В. Огнева. — М. : Издательство Юрайт, 2019. — 322 с.
5. Малявко, А. А. Формальные языки и компиляторы : учеб. пособие для вузов / А. А. Малявко. — М. : Издательство Юрайт, 2018. — 429 с.
6. Маркин, А. В. Программирование на sql в 2 ч. Часть 2 : учебник и практикум для бакалавриата и магистратуры / А. В. Маркин. — М. : Издательство Юрайт, 2019. — 292 с.
7. Нортроп, Т. Основы разработки приложений на платформе Microsoft .NET Framework. Учебный курс Microsoft. Перевод с англ./ Т. Нортроп, Ш. Уилдермьюс, Б. Райан. - М.: «Русская редакция», 2007. - 864 с.
8. Советов, Б. Я. Базы данных : учебник для прикладного бакалавриата / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. — 3-е изд., перераб. и доп. — М. : Издательство Юрайт, 2019. — 420 с.
9. Трофимов, В. В. Основы алгоритмизации и программирования : учебник для СПО / В. В. Трофимов, Т. А. Павловская ; под ред. В. В. Трофимова. — М. : Издательство Юрайт, 2019. — 137 с.

# Приложение А. Руководство пользователя

7.1 Запуск программы

7.1.1 После запуска программы на экран выводится главное окно программы, в котором требуется авторизация. Вид окна представлен на рисунке 7.1.

|  |
| --- |
| Рисунок 7.1 – Главное окно программы |

* + 1. Для авторизации необходимо ввести логин, пароль, номер телефона и нажать функциональную клавишу “Войти”.
    2. Если введен неверный логин или пароль или номер телефона, то программа выдаст соответствующую ошибку (рисунок 7.2). Потребуется повторный ввод пароля.

|  |
| --- |
| Рисунок 7.2 – Окно ошибки «Неуспешная авторизация» |

* + 1. Если у пользователя нет аккаунта, он может его зарегистрировать, перейдя на форму регистрации. Вид окна регистрации представлен на рисунке 7.3.

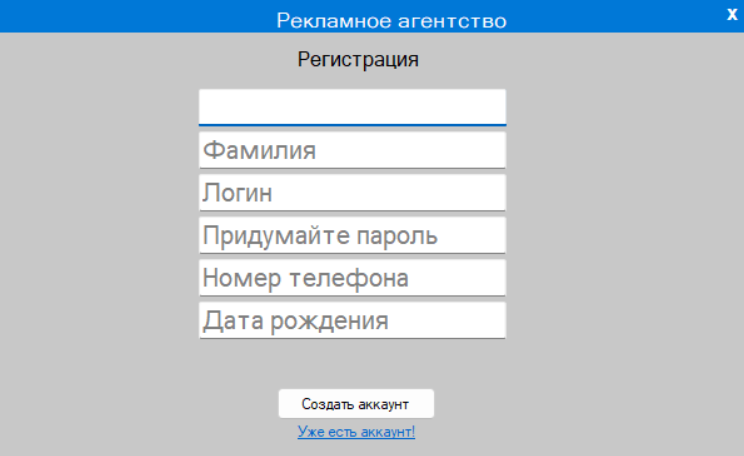


Рисунок 7.3 – Окно регистрации

* + 1. После успешной регистрации и авторизации откроется новое окно, в котором представлен функционал работы с базой данных для рекламного агентства. Вид основного меню представлен на рисунке 7.4.

|  |
| --- |
| Рисунок 7.4 – Окно основного меню |

* + 1. При заполнении полей данными и нажатии на кнопку «Добавить» пользователь добавляется в список абонентов.
    2. При заполнении всех полей необходимой информацией и нажатии на кнопку “Изменить”, предварительно выбрав нужную строку, данные клиента изменятся на данные, введенные пользователем.
    3. При выборе определенной строки и нажатии кнопки “Удалить” появится диалоговое окно с подтверждением действия. Вид диалогового окна основного меню после нажатия кнопки “Удалить” представлен на рисунке 7.5.

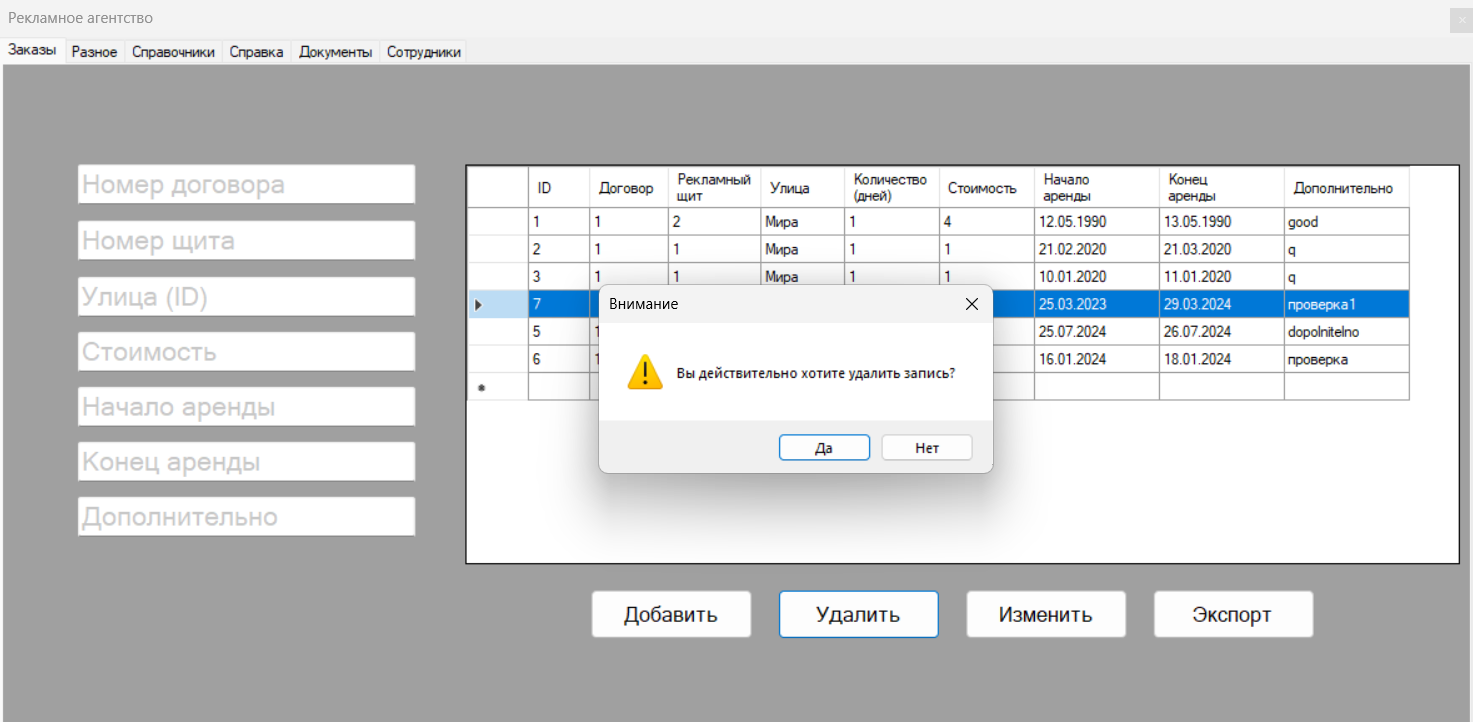


Рисунок 7.5 – Диалоговое окно подтверждающее удаление

* + 1. Далее, если на диалоговом окне нажать “Да” произойдет удаление, выбранного ранее пользователя. Вид окна основного меню после удаления пользователя на рисунках 7.6. и 7.7

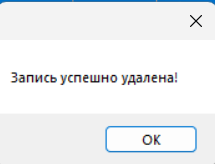
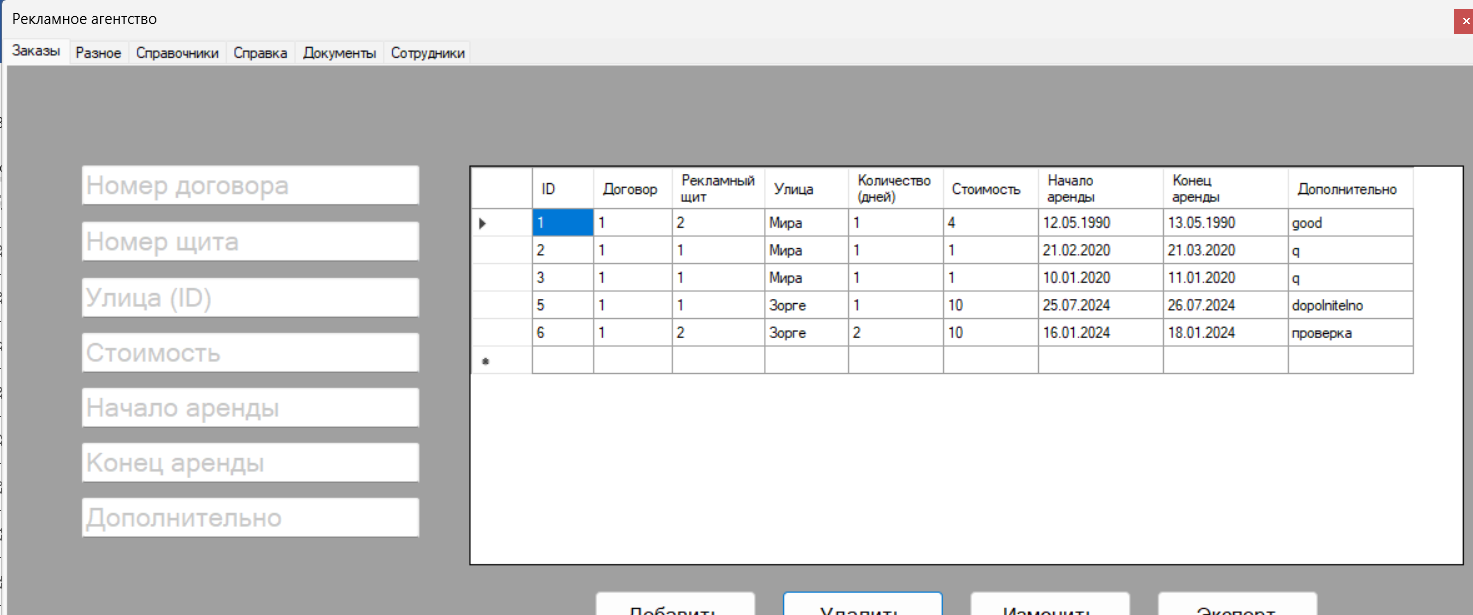


Рисунок 7.6 – Диалоговое окно подтверждающее удаление



|  |
| --- |
| Рисунок 7.7 – Вид окна основного меню после удаления |

* + 1. При нажатии на кнопку экспорт вызовется диалоговое окно, в котором можно либо создать новый текстовый файл, либо заменить существующий. Вид диалогового окна на рисунке 7.8.

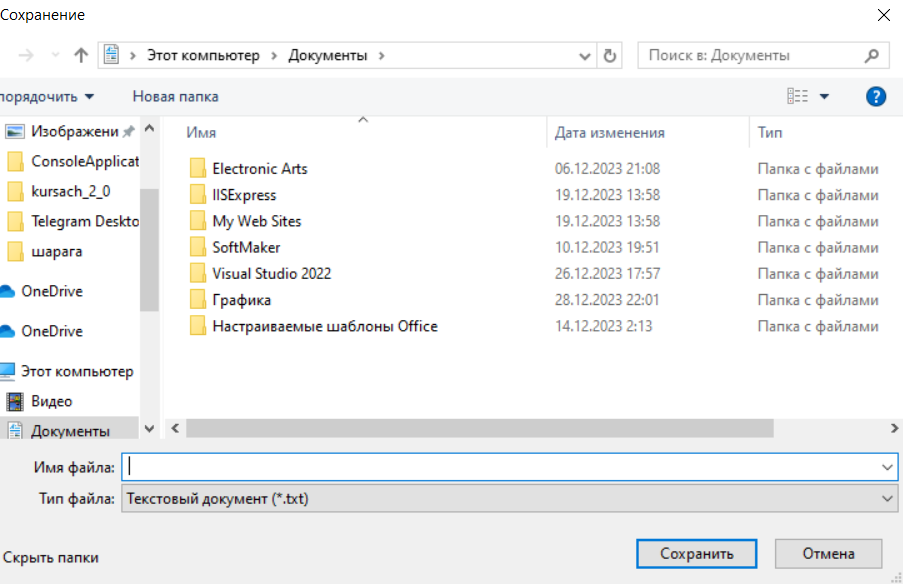


Рисунок 7.8 – Вид диалогового окна для экспорта данных

7.3.7 После выбора или создания файла и нажатия на кнопку сохранить, сформируется текстовый документ со всеми клиентами. Вид диалогового окна на рисунке 7.9.

|  |
| --- |
| Рисунок 7.9 – Окно сформированного текстового документа |

* + 1. Для выхода из программы необходимо нажать на кнопку  в правом верхнем углу.
    2. При необходимости, пользователь может сменить пароль, выбрав вкладку «Разное». Вид вкладки представлен на рисунке 7.10



Рисунок 7.10 – Вид вкладки «Разное»

* + 1. После нажатия на кнопку «Сменить пароль» открывается дополнительное окно для смены пароля. (Рисунок 7.11)

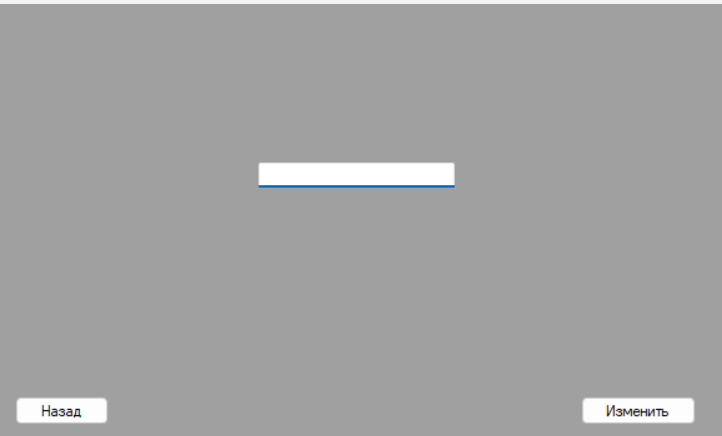


Рисунок 5.11 – Вид окна для смены пароля

7.4 Вид вкладок для администратора представлен на рисунке 7.12. Содержание справочников для администратора.

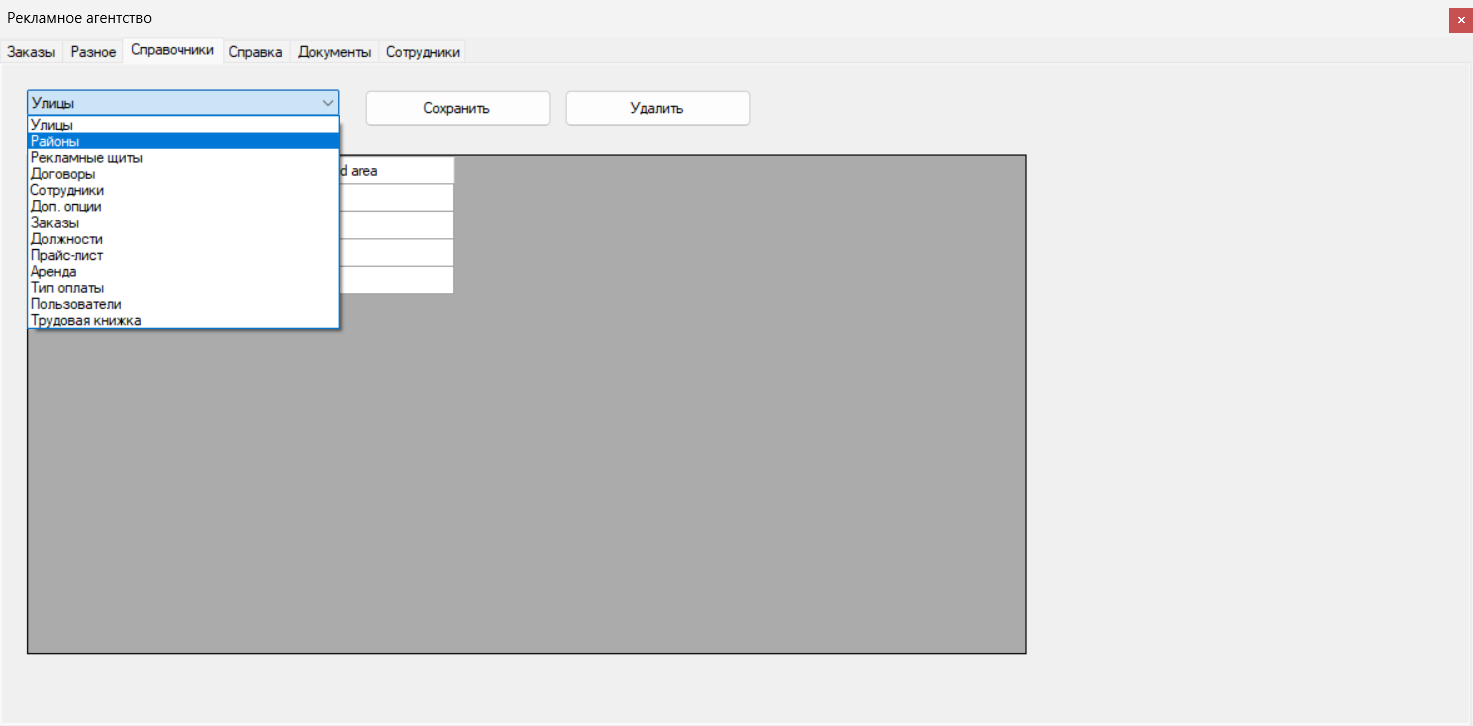


Рисунок 7.12 – Вид вкладок и справочников для администратора

7.5 Для обычного пользователя ограничен список доступных справочников и вкладок (рисунок 7.13).

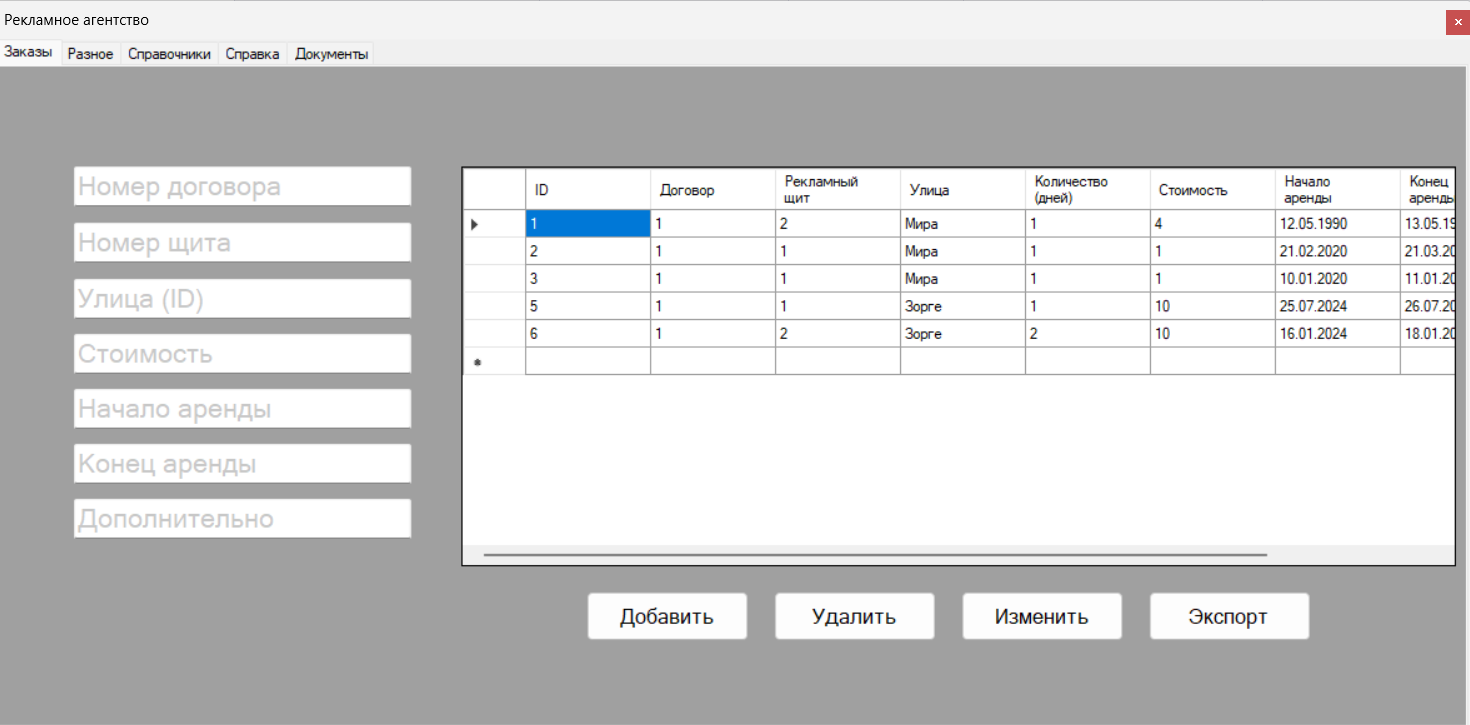


Рисунок 7.13 – Вид вкладок для обычного пользователя

7.6 Для пользователя c ограниченным доступом кнопки становятся недоступными в зависимости от полученных прав. (рисунок 7.14)

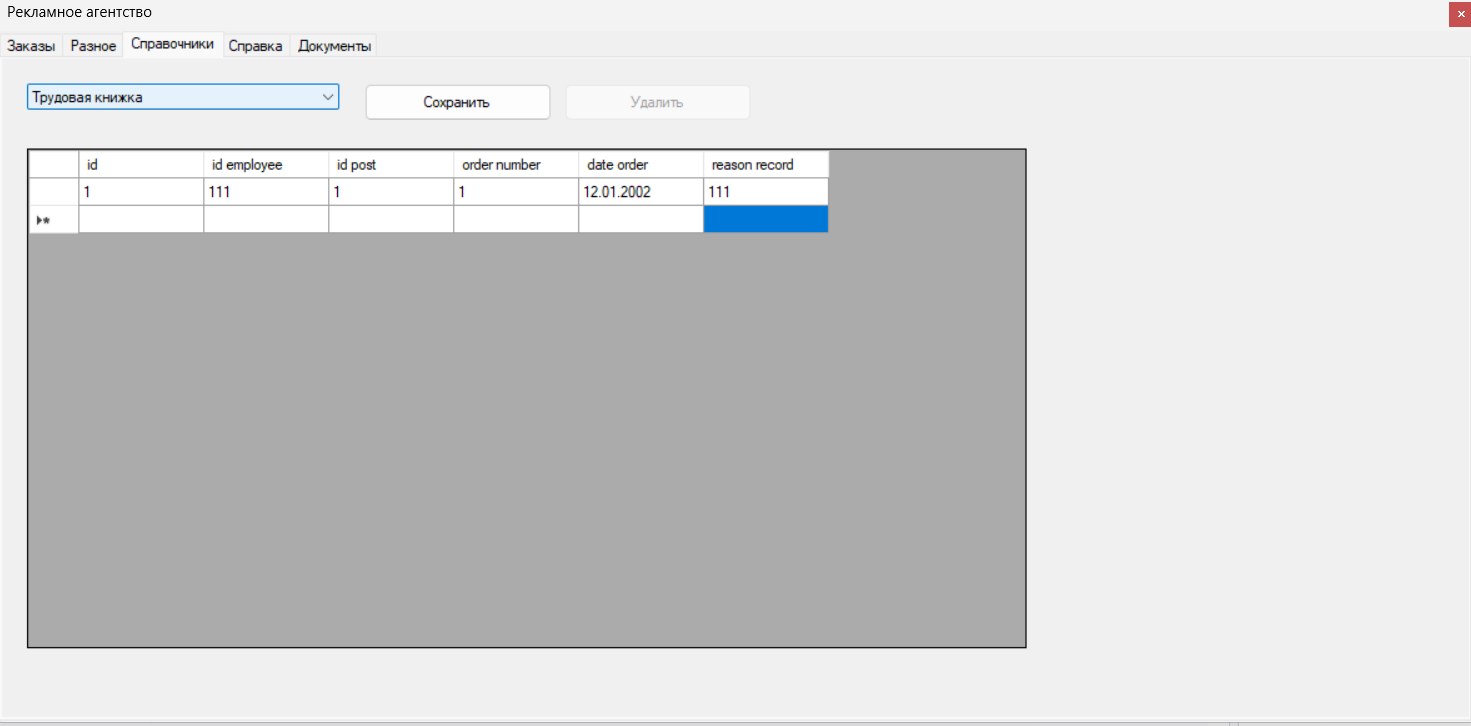


Рисунок 7.14 – Вид окна «Справочники» для пользователя с ограниченным доступом

7.6 Вид вкладки «Справка» с разделами «Содержание» и «О программе» представлен на рисунке 7.15

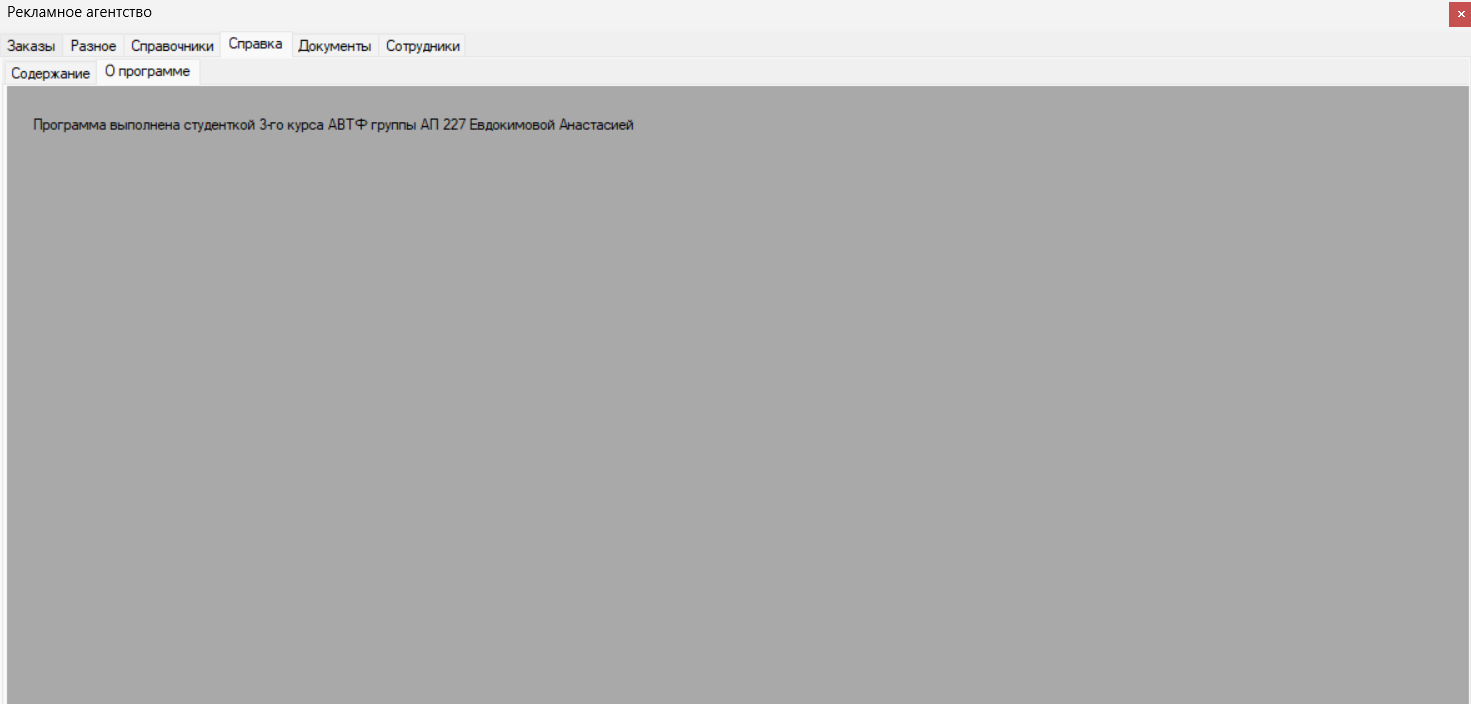


Рисунок 7.15 – Вид вкладки «Справка»

7.7 Вкладка «Документы» доступна для всех пользователей. Вид представлен на рисунке 7.16.

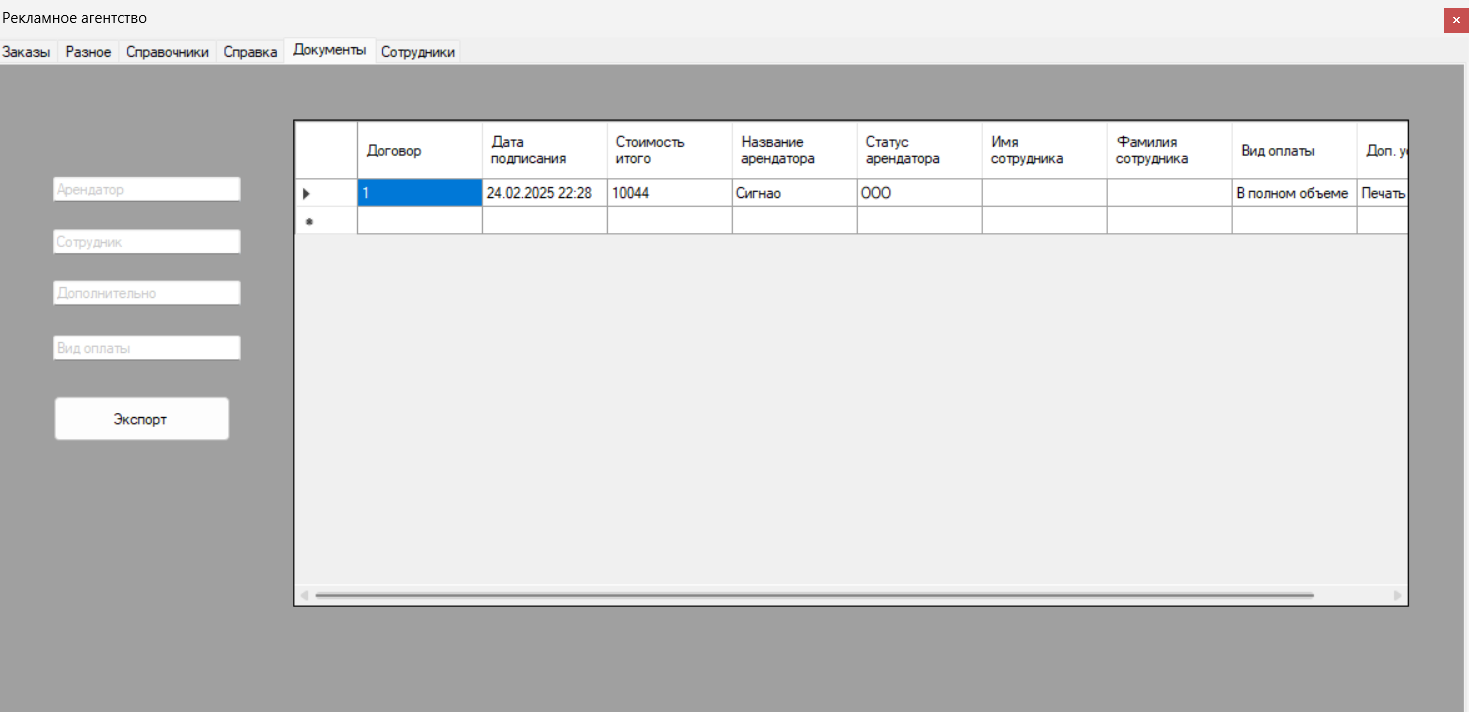


Рисунок 7.16 – Вид вкладки «Документы»

7.8 Вкладка «Сотрудники» недоступна обычному пользователю. Суперпользователь может выдавать права зарегистрированным пользователям. (Рисунок 7.17, рисунок 7.18)

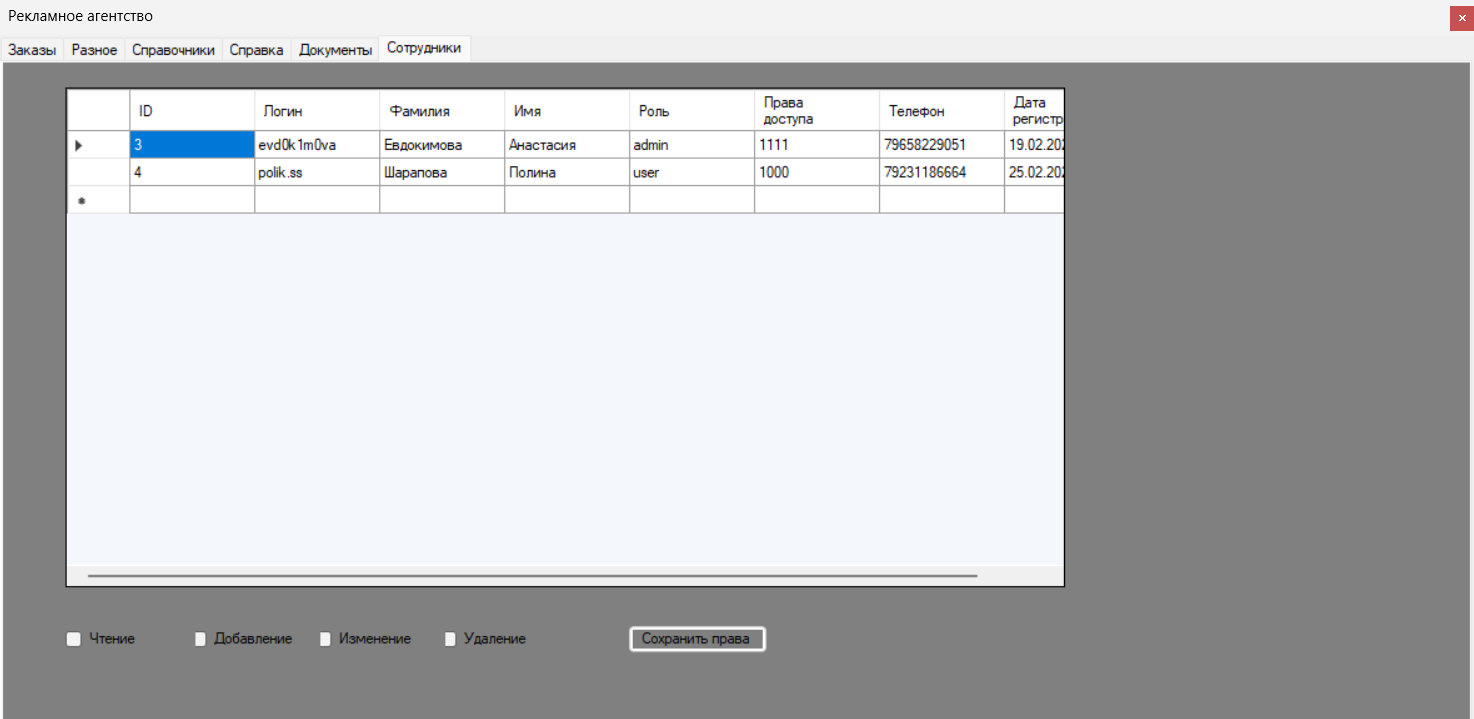


Рисунок 7.17 - Вид вкладки «Сотрудники»

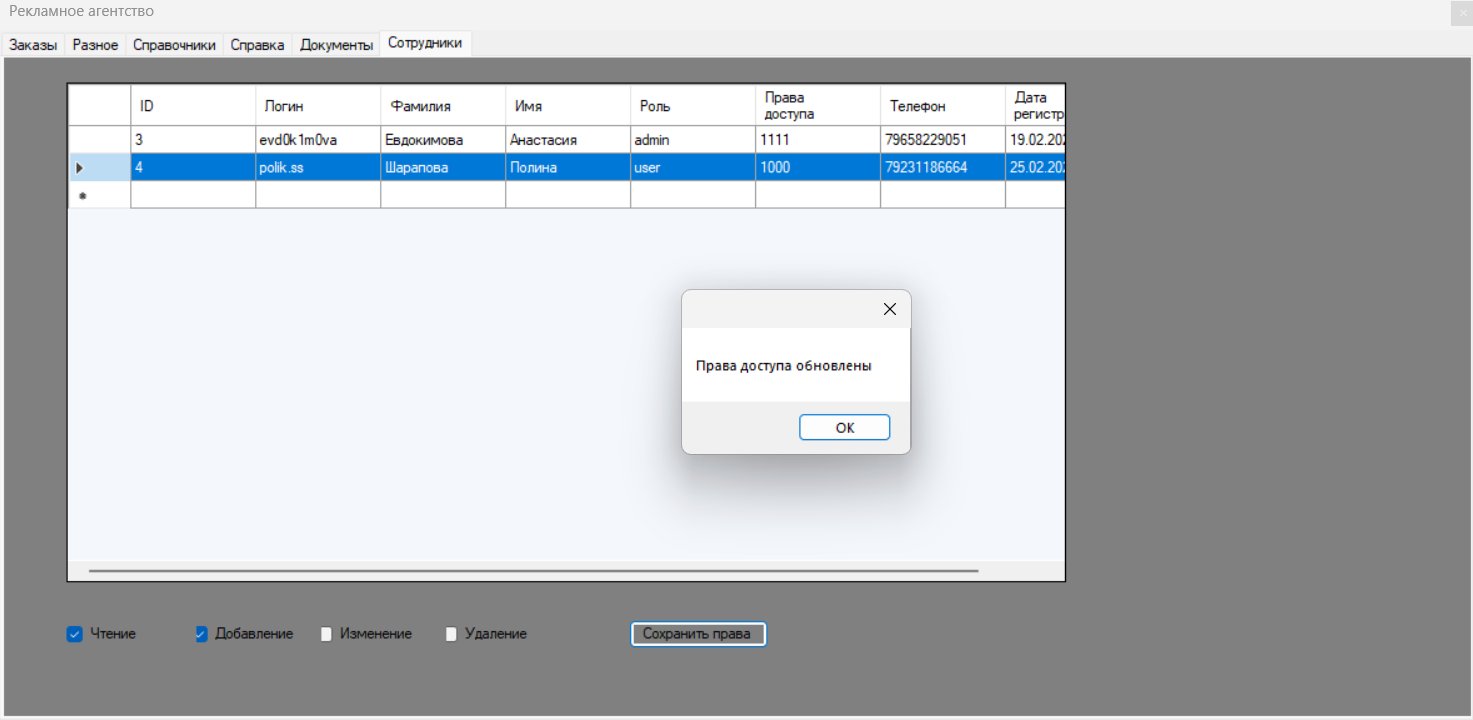


Рисунок 7.18 – Обновление прав доступа для сотрудника