ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

КП.083.

Министерство цифрового развития, связи и массовых коммуникаций РФ

Уральский технический институт связи и информатики (филиал)

ФГБОУ ВО «Сибирский государственный университет

телекоммуникаций и информатики» в г. Екатеринбурге

(УрТИСИ СибГУТИ)

КУРСОВОЙ ПРОЕКТ

по междисциплинарному курсу: «Технология разработки программного обеспечения»

на тему: Разработка приложения по автоматизации деятельности АН «Азбука Недвижимости»

Выполнил:

студент группы 083

Морева А. Н.

Руководитель:

Ермоленко О. М.

Екатеринбург 2023

Отзыв руководителя

Министерство цифрового развития, связи и массовых коммуникаций РФ

Уральский технический институт связи и информатики (филиал)

ФГБОУ ВО «Сибирский государственный университет

телекоммуникаций и информатики» в г. Екатеринбурге

(УрТИСИ СибГУТИ)

Цикловая комиссия ИТ и АСУ кафедры ИСТ

ЗАДАНИЕ ДЛЯ КУРСОВОГО ПРОЕКТИРОВАНИЯ

По междисциплинарному курсу Технология разработки программного обеспечения

Студента 4 курса 083 группы

Моревой Анны Николаевной

ТЕМА ЗАДАНИЯ Разработка приложения по автоматизации деятельности АН «Азбука Недвижимости»

ИСХОДНЫЕ ДАННЫЕ

Рассмотреть теоретические вопросы, связанные с деятельностью АН «Азбука Недвижимости».

Составить программу, которая должна включать в себя: базу данных, сортировку, добавление недвижимости и авторизацию пользователя.

При выполнении курсового проекта на указанную тему должны быть представлены:

1. Пояснительная записка

Введение; Теоретические основы разработки приложения автоматизации; Анализ предметной области; Аналогичные программные продукты; Аналогичные программные продукты; Разработка программного продукта; Проектирование базы данных; Инструкция по работе с программой Заключение; Библиография; Приложение А

2. Графическая часть

Дата выдачи Срок окончания

Председатель ЦК ИТиАСУ кафедры ИСТ Ермоленко О.М.

Преподаватель – руководитель Ермоленко О.М.

Студент Морева А.Н.

Содержание

Введение 6

1 Теоретические основы разработки приложения автоматизации 7

1.1 Анализ предметной области 7

1.2 Аналогичные программные продукты 9

1.3 Требования к приложениям по автоматизации деятельности 10

# 1.4 Выбор средств для реализации приложения 11

2 Разработка программного продукта 13

2.1 Разработка блок-схемы программы 13

2.2 Проектирование базы данных 14

2.3 Макеты основных форм приложения 15

2.4 Интерфейс приложения 16

2.5 Инструкция по работе с программой 17

Заключение 20

Библиография 21

Приложение А 22

# Введение

Использование автоматизированной системы управления предприятием на сегодняшний день является ключевым фактором конкурентной борьбы организаций. Информационная система позволяет повысить производительность труда сотрудников, ускорить информационный обмен между подразделениями организации, снизить издержки, контролировать в режиме реально времени все процессы, протекающие на предприятии. Актуальность данной работы обусловлена тем, что автоматизация все го спектра бизнес-процессов организации позволяет существенно повысить конкурентоспособность организации, улучшить качество управления предприятием, снизить издержки, быстро обрабатывать большие объемы информации.

Комплекс задач по автоматизации деятельности агентства недвижимость включает в себя следующие задачи: хранение информации о клиентах; хранение информации об объектах недвижимости и их владельцах, осуществление учет сделок и создание отчетов по параметрам запросов.

Проектирование информационных систем – сложная и длительная работа, требующая высокого профессионализма. В процессе создания и работы системы потребности пользователей всегда изменяются или уточняются, что усложняет и удлиняет процесс разработки таких систем.

Большинство существующих CASE-средств основано на методах структурного или объектно-ориентированного анализа и проектирования, использующих спецификации в виде диаграмм или текстов для описания внешних требований, связей между моделями системы, динамики поведения системы и архитектуры программных средств.

Объект работы – агентство недвижимости;

Предмет работы – учет клиентов, сделок, объектов недвижимости.

Целью курсового проекта является разработка приложения по автоматизации деятельности АН «Азбука Недвижимости».

Для достижения поставленной цели необходимо решить следующие задачи:

1. изучение литературы по данной теме.
2. анализ существующих программных разработок для агентства недвижимости.
3. выбор средств разработки программного приложения.
4. разработка структуры системы и модели базы данных системы.

1 Теоретические основы разработки приложения автоматизации

* 1. Анализ предметной области

АН "Азбука Недвижимости" является агентством недвижимости, осуществляющим различные виды услуг в сфере недвижимости. Основной целью деятельности агентства является предоставление клиентам профессиональной помощи в покупке, продаже, аренде или аренде с правом выкупа недвижимости.

АН "Азбука Недвижимости" предлагает широкий спектр услуг, включающий:

1. консультации и помощь в выборе недвижимости. Клиентам предоставляется профессиональное сопровождение на каждом этапе сделки, начиная с оценки потребностей и пожеланий клиента, до подбора подходящих объектов недвижимости и составления договора.
2. размещение объявлений о продаже и аренде недвижимости. Агентство предоставляет возможность продавцам и арендодателям размещать объявления о продаже или аренде своей недвижимости на сайте агентства, что позволяет значительно увеличить шансы на успешное завершение сделки.
3. правовая поддержка сделок с недвижимостью. Агентство предоставляет услуги юристов, специализирующихся на сделках с недвижимостью, которые могут помочь клиентам в оформлении необходимой документации, проверке юридической чистоты объекта недвижимости и всем юридическим вопросам, связанным с сделкой.
4. оценка стоимости недвижимости. Агентство проводит оценку рыночной стоимости недвижимости, что позволяет клиентам более точно определить цену при продаже или аренде объекта.
5. управление недвижимостью. Агентство может оказывать услуги по управлению недвижимостью, включая сдачу в аренду, взыскание арендной платы, проведение текущего и капитального ремонта и другие операции, связанные с эксплуатацией недвижимости.

АН "Азбука Недвижимости" имеет следующую структуру:

1. руководство: включает в себя директора или генерального директора, который отвечает за общее управление агентством, принятие стратегических решений и контроль за выполнением поставленных целей.
2. отдел продаж: занимается поиском покупателей или арендаторов для объектов недвижимости, проведением показов, переговорами и заключением договоров.
3. отдел аренды: отвечает за поиск арендаторов для объектов недвижимости, подбор подходящих арендных предложений, составление договоров аренды и контроль за своевременным взысканием арендной платы.
4. отдел юриспруденции: включает в себя юристов, которые занимаются юридическим сопровождением сделок с недвижимостью, оформлением необходимой документации и предоставлением правовой консультации клиентам.
5. отдел маркетинга и рекламы: отвечает за разработку маркетинговых стратегий, рекламные кампании, поддержку сайта агентства, размещение объявлений и привлечение клиентов.
6. отдел оценки недвижимости: занимается проведением оценки рыночной стоимости объектов недвижимости, составлением экспертных заключений и консультацией клиентов по вопросам ценообразования.
7. отдел управления недвижимостью: выполняет функции по управлению недвижимостью, включая сдачу в аренду, взыскание арендной платы, обслуживание и ремонт объектов.

Все отделы тесно взаимодействуют друг с другом для достижения общей цели - предоставление клиентам профессиональных услуг в сфере недвижимости.

* 1. Аналогичные программные продукты

Онлайн CRM система. База объектов и клиентов, прием входящих заявок и автоматический подбор для них объектов, ведение сделки по стадиям и формы для быстрого создания документов, выгрузка объектов недвижимости на сайты объявлений, возможность sms, email рассылок, звонки непосредственно из браузера через встроенную voip телефонию, интеграция с социальными сетями. Готовое отраслевое решение для агентств недвижимости.

Вид программного продукта представлен рисунком ниже (Рисунок 1.1):

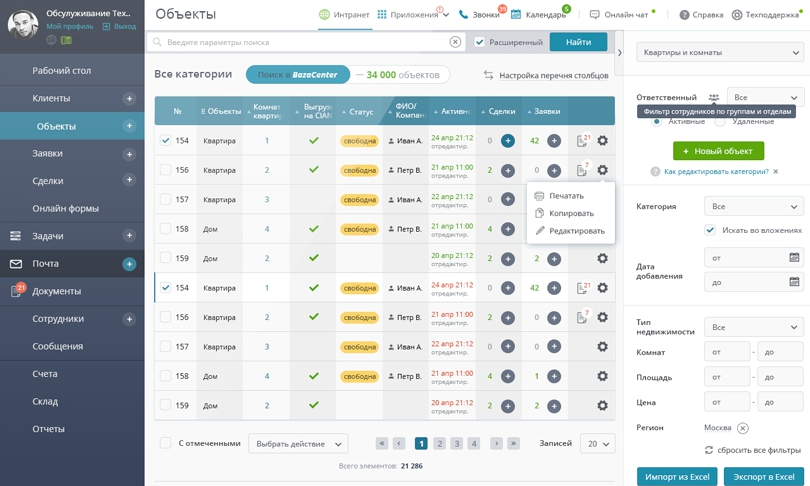


Рисунок 1.1 - Intrum CRM

Профессиональная CRM для агентств недвижимости. Полная статистика по разным рекламным источникам, что поможет избежать лишних затрат при их низкой эффективности. Выгрузка объявления на рекламную площадку в несколько кликов. Автоматическая запись поступающих в CRM звонков и напоминания о необходимости перезвонить клиенту. Все объекты, которые ведет агентство, отображаются на карте внутри интерфейса CRM. Оперативная техподдержка, быстро решающая текущие вопросы.

Вид программного продукта представлен рисунком ниже (Рисунок 1.2):

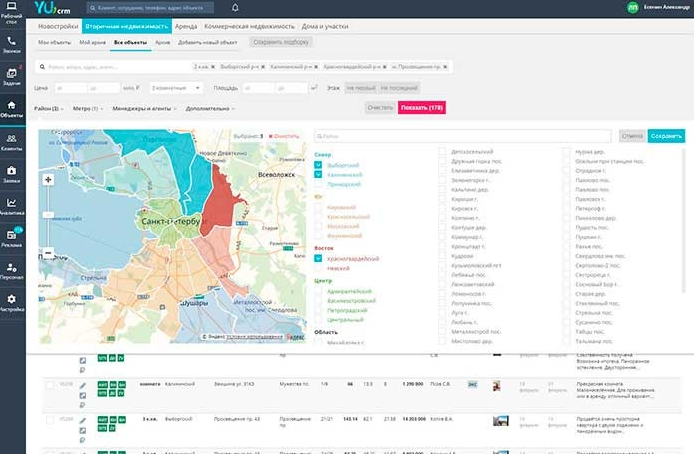


Рисунок 1.2 – Yucrm

Универсальная программа с автоматизацией всех процессов аренды недвижимости. Для учета и контроля всех процессов по аренде объектов недвижимости — таких как боксы на складах индивидуального хранения, офисы, магазины и т.д. Система позволяет в режиме онлайн видеть занятость арендуемых площадей, стоимость и другие характеристики каждого объекта.

Вид программного продукта представлен рисунком ниже (Рисунок 1.3):

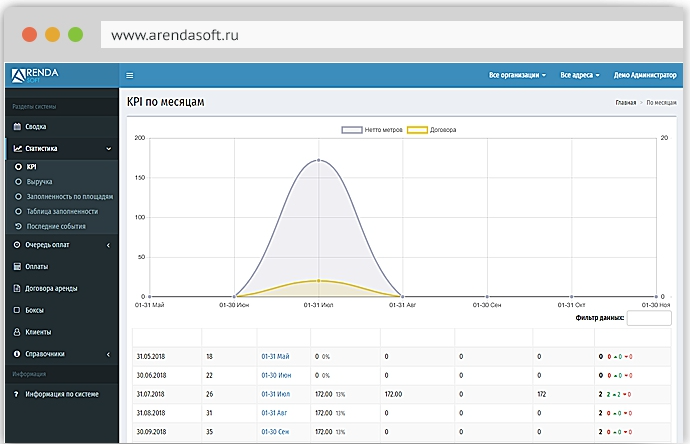


Рисунок 1.3 - АрендаСофт

1.3 Требования к приложениям по автоматизации деятельности

1.3.1 Функциональные требования

Приложение должно обеспечивать выполнение следующих функций:

1. инициализацию системы.
2. ввод, редактирование и хранение информации о клиентах, арендодателях и агентов по недвижимости. Хранение журнала сделок.
3. обработка запросов пользователей и выдача информации, соответствующей запросу.
4. разделение уровней доступа пользователей системы.
5. формирование отчетов и документов.

1.3.2 Нефункциональные требования

Нефункциональные требования - требования, которые определяют, как хорошо должно работать приложение, какие характеристики качества и производительности оно должно обладать. Например:

1. контроль вводимой информации;
2. резервное копирование данных;
3. разграничение прав доступа;
4. блокирование некорректных действий;
5. обработка исключительных ситуаций;
6. длительное хранение информации (не менее 5 лет).

1.3.3 Требования к составу и параметрам технических средств

1. система должна работать на IBM совместимых компьютерах;
2. тактовая частота процессора не менее 1,5 МГц.
3. объем оперативного запоминающего устройства – 256 Мб и более;

1.4 Выбор средств для реализации приложения

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки (IDE) программного обеспечения и ряд других инструментов. Данные продукты позволяют разрабатывать как консольные приложения, так и игры и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, UWP, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, .NET Core, .NET, MAUI, Xbox, Windows Phone .NET Compact Framework и Silverlight. После покупки компании Xamarin корпорацией Microsoft появилась возможность разработки IOS и Android программ.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

Microsoft Office Access или просто Microsoft Access — реляционная система управления базами данных (СУБД) корпорации Microsoft. Входит в состав пакета Microsoft Office. Имеет широкий спектр функций, включая связанные запросы, связь с внешними таблицами и базами данных. Благодаря встроенному языку VBA в самом Access можно писать приложения, работающие с базами данных.

C# — объектно-ориентированный язык программирования общего назначения. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework и .NET Core. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

С#‎ разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов С#‎, которая отражает BCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем (однако, эта закономерность была нарушена с выходом C# 3.0, представляющего собой расширения языка, не опирающиеся на расширения платформы .NET). CLR предоставляет С#‎, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#‎, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на VB.NET, J#.

Преимущества:

1. широкое применение. C# используется для разработки разнообразных программ и приложений, включая десктопные, веб-приложения, мобильные приложения, игры и т.д.

2. интеграция с платформой NET. C# входит в состав платформы .NET, что позволяет разработчикам легко использовать ее функциональность, такую как управление памятью, работа с базами данных, многопоточность и другие возможности.

3. объектно-ориентированность. C# является языком с полной поддержкой объектно-ориентированного программирования, позволяющего разрабатывать структурированный и модульный код.

4. большое сообщество разработчиков: C# имеет огромное сообщество разработчиков и обширную документацию, что упрощает получение помощи и решение возникающих проблем.

Недостатки:

1. ограничения платформы. C# основан на платформе .NET, что может быть ограничением в некоторых ситуациях, особенно при разработке для операционных систем, не являющихся Windows.

2. недостаточная производительность. В некоторых случаях C# может иметь худшую производительность по сравнению с низкоуровневыми языками, такими как C++.

3. более высокая стоимость разработки. Поскольку C# является одним из коммерческих языков программирования, его использование может требовать определенных затрат на лицензии и инструменты разработки.

В целом, C# является отличным выбором для разработки различных программных решений и приложений, обеспечивая надежность, удобство и эффективность кода.

2 Разработка программного продукта

2.1 Разработка блок-схемы программы

Алгоритм рекомендованного поведения пользователя и то, как пользоваться программой можно увидеть на блок-схеме ниже (Рисунок 2.1).

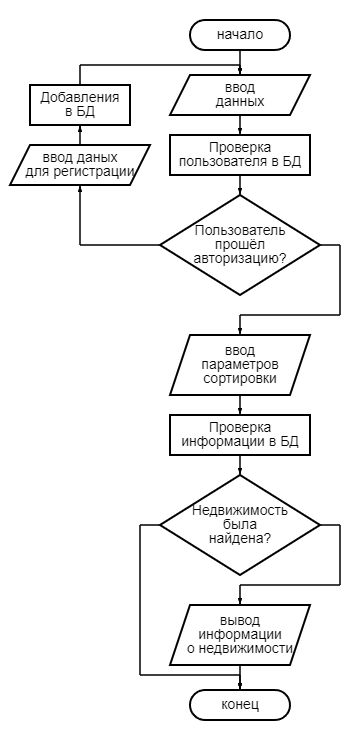


Рисунок 2.1 - Блок-схема программы

2.2 Проектирование базы данных

Проанализировав деятельность агентства недвижимости, можно приступить к проектированию базы данных. Для проектирования используется программа «Microsoft Access».

Для решения задач учета клиентов, объектов недвижимости и сделок с ними, деятельности сотрудников, учета владельцев объектов недвижимости создается база данных. Она состоит из следующих таблиц:

* владельцы;
* клиенты;
* сотрудники;
* сделки;
* типы объектов;
* информации о клиенте.

База данных приведена к третьей нормальной форме. Логическая модель базы (Рисунок 2.2).

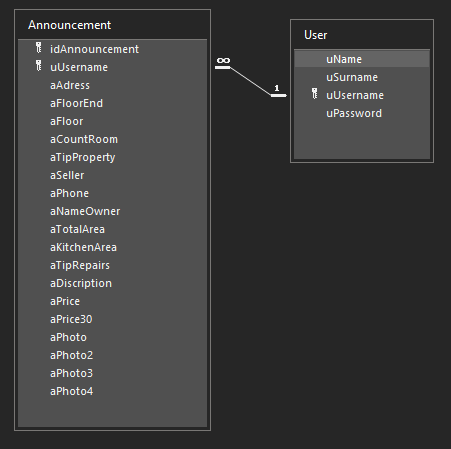


Рисунок 2.2 – Логическая структура базы данных

2.3 Макеты основных форм приложения

Для будущей программы был нарисован макет форм в растровом графическом редакторе Microsoft Paint (Рисунок 2.3, 2.4, 2.5).

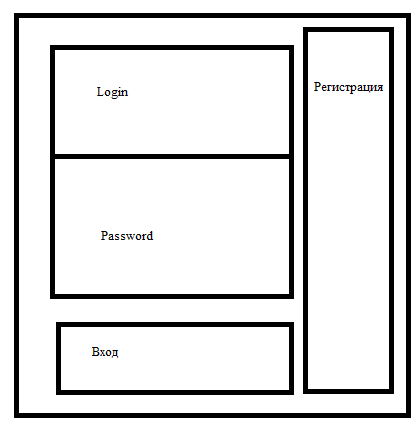


Рисунок 2.3 – Макет окна авторизации

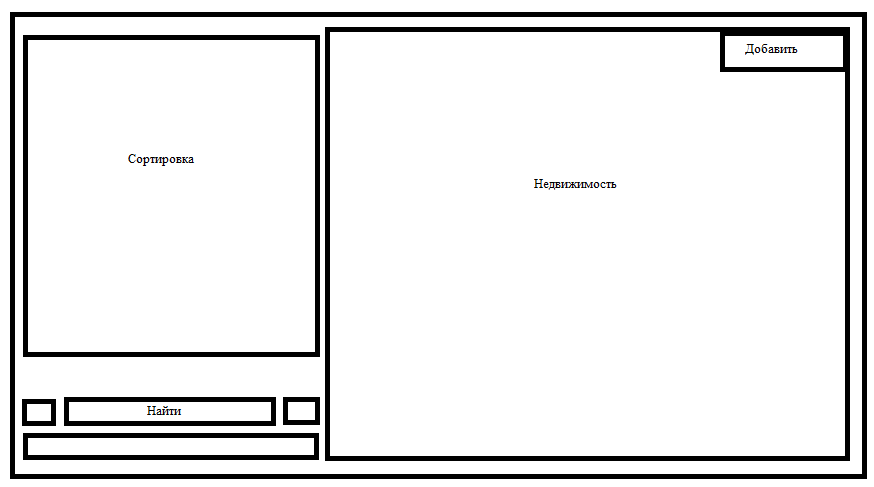


Рисунок 2.4 – Макет окна приложения

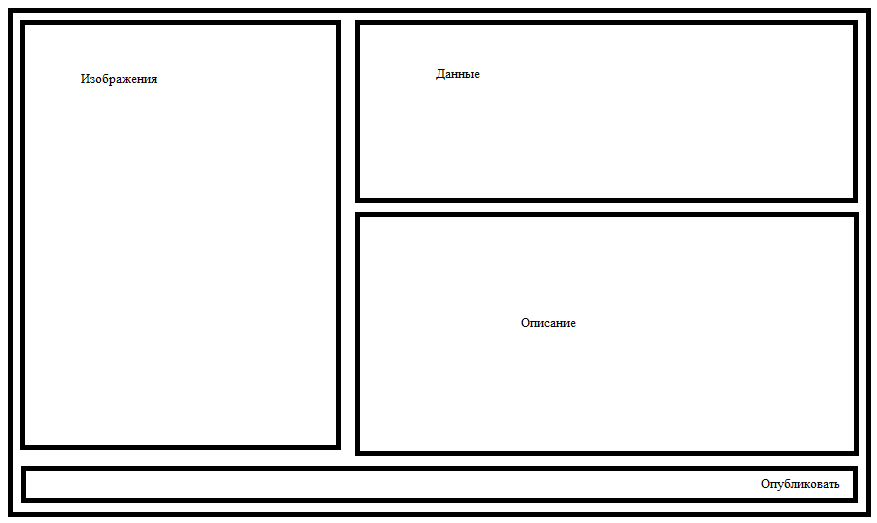


Рисунок 2.5 – Макет окна добавления недвижимости

2.4 Интерфейс приложения

Интерфейс разрабатываемого приложения по автоматизации деятельности АН «Азбука Недвижимости» предлагает удобный и интуитивно понятный пользовательский опыт для эффективного выполнения задач в сфере недвижимости.

Главная цель интерфейса данного приложения - облегчить работу сотрудников АН «Азбука Недвижимости», упростить процессы и повысить производительность бизнеса. Приложение предлагает широкий спектр функций и возможностей, которые охватывают деятельность агентства по недвижимости.

Основные особенности интерфейса включают:

* панель управления.
* Каталог недвижимости.
* Управление запросами.
* Расписание просмотров.
* Отчеты и аналитика.

Общий дизайн интерфейса зависит от индивидуальных предпочтений АН «Азбука Недвижимости», однако, наиболее эффективным подходом может быть использование интуитивно понятных символов, простого и легкодоступного меню навигации, а также контрастных цветовых схем для удобства восприятия информации.

Окружение в приложении должно быть гибким и персонализируемым, позволяющим пользователям настраивать интерфейс в соответствии с их особенностями работы и предпочтениями. В целом, интерфейс разработанного приложения по автоматизации деятельности АН «Азбука Недвижимости» должен быть интуитивно понятным, эффективным и помогать сотрудникам максимально эффективно выполнять свои задачи в сфере недвижимости.

2.5 Инструкция по работе с программой

На рисунке 2.9 продемонстрирована авторизация в приложении.

1) Вводится логин с паролем, а далее кнопка «SIGN IN» (Рисунок 2.6).

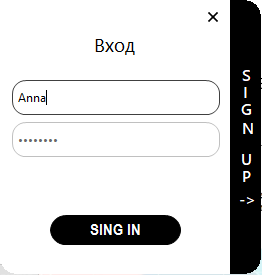


Рисунок 2.6 – Авторизация

2) В главном окне заполняется панель сортировки, а после выводится список объявлений (Рисунок 2.7).

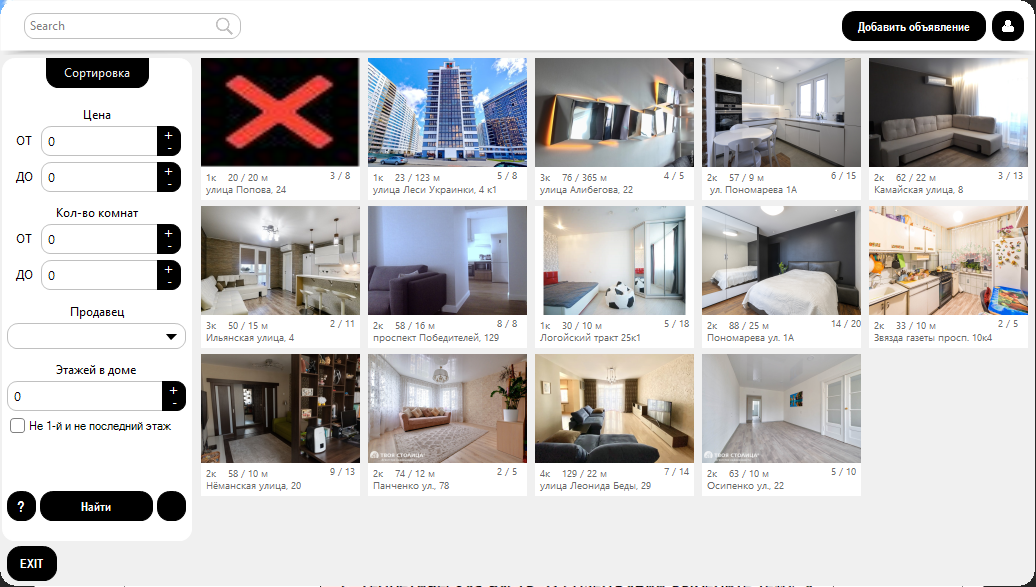


Рисунок 2.7 – Главное окно

3) Для добавления объявления необходимо нажать на кнопку «Добавить объявление» и заполнить все параметры объявления (Рисунок 2.8, 2.9).



Рисунок 2.8 – Кнопка «Добавить объявление»

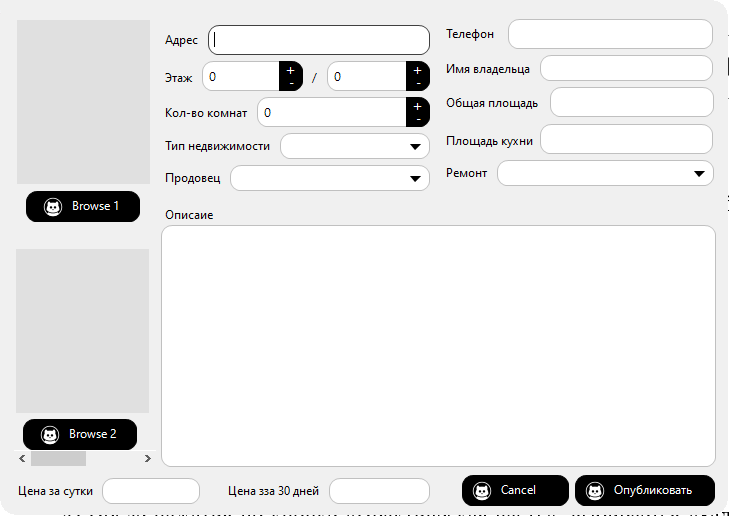


Рисунок 2.9 – Параметры объявления

4) Далее все данные будут загружаться в базу данных Microsoft Access (Рисунок 2.10).

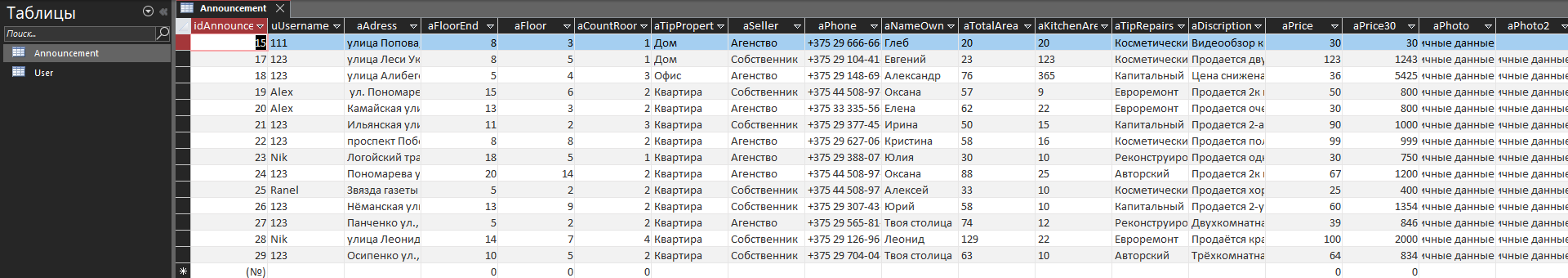


Рисунок 2.10 – Таблица «Недвижимость»

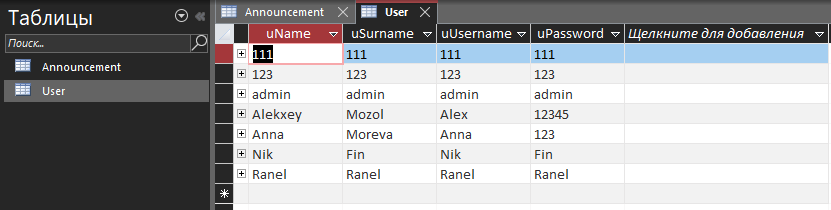


Рисунок 2.11 – Таблица «Пользователей»

Заключение

В данной работе было разработано приложение по автоматизации деятельности АН «Азбука Недвижимости».

Успешная работа компании во многом зависит от уровня ее технического оснащения и эффективной автоматизации процессов управления в условиях увеличивающегося с каждым днем объема информации.

Появление приложений является одной из первоначальных задач усовершенствования деятельности управления.

В результате внедрения приложения удается достичь:

* повышения продуктивности работы отдельных сотрудников и подразделений в целом.
* сокращения времени исполнения заданий сотрудниками.
* повышения оперативности получения необходимой информации.
* качественного улучшения контроля деятельности учреждения.
* повышения оперативности и качества принятия управленческих решений за счет более адекватного отражения реальной ситуации на предприятии.

В рамках данной курсовой работы было проведено изучение деятельности агентства недвижимости, создана база данных организации.

Созданная БД позволяет хранить информацию о клиентах, сотрудниках, объектах недвижимости, собственниках недвижимости, а также вести учет сделок.

Поставленная цель работы достигнута, а именно разработано приложение автоматизации и обработки информации о клиентах, объектах недвижимости, собственниках недвижимости.

В дальнейшем планируется усовершенствовать существующую БД, а также создать эргономичный программный интерфейс для облегчения работы с системой, добавить аналитических возможностей для поддержки принятия решений руководителем агентства.

Библиография

1 Автоматизированные информационные технологии в экономике [Электронный ресурс] – Режим доступа: https://www.studmed.ru/emelyanova-nz-partyka-tl-popov-ii-osnovy-postroeniya-avtomatizirovannyh-informacionnyh-sistem\_be59ee12b74.html;

2 Проектирование и разработка корпоративных информационных систем [Электронный ресурс] – Режим доступа: https://infourok.ru/lekciya-sostav-i-struktura-asu-po-predmetu-osnovi-postroeniya-avtomatizirovannih-informacionnih-sistem-2936571.html;

3 Проектирование экономических информационных систем [Электронный ресурс] – Режим доступа: https://infourok.ru/lekciya-sostav-i-struktura-asu-po-predmetu-osnovi-postroeniya-avtomatizirovannih-informacionnih-sistem-2936571.html;

4 Информационные системы в экономике и управлении [Электронный ресурс] – Режим доступа: http://ptca.narod.ru/lec/lec4\_4.html;

5 Автоматизированные системы управления [Электронный ресурс] – Режим доступа: https://clck.ru/RGGss;

6 Проектирование информационных систем [Электронный ресурс] – Режим доступа: http://lib.usue.ru/resource/free/10/MelnikovAlgebra3/00FncBoo1.pdf;

7 Анализ и оценка приносящей доход недвижимости [Электронный ресурс] – Режим доступа: http://scask.ru/p\_book\_loga.php?id=15.

# 

# Приложение А

(справочное)

Листинг программы

Ниже представлен листинг программы:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.OleDb;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace TheProperty

{

public partial class Form1 : Form

{

OleDbConnection connection = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=Database.mdb;");

public Form1()

{

InitializeComponent();

SearchPanel.Location = new Point(21, 11);

this.ActiveControl = UserName;

PrintAnnouncement("SELECT \* FROM Announcement");

\_obj = this;

}

static Form1 \_obj;

public static Form1 Instance

{

get

{

if (\_obj == null)

{

\_obj = new Form1();

}

return \_obj;

}

}

public Guna.UI.WinForms.GunaPanel Search

{

get { return SearchPanel; }

set { SearchPanel = value; }

}

public TableLayoutPanel tableLayout

{

get { return tableLayoutPanel1; }

set { tableLayoutPanel1 = value; }

}

public Guna.UI.WinForms.GunaPanel Info

{

get { return InfoPanel; }

set { InfoPanel = value; }

}

Point lastPoint;

private void TopPanel\_MouseDown(object sender, MouseEventArgs e)

{

this.ActiveControl = UserName;

lastPoint = new Point(e.X, e.Y);

}

private void TopPanel\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

this.Left += e.X - lastPoint.X;

this.Top += e.Y - lastPoint.Y;

}

}

private void gunaButton5\_Click(object sender, EventArgs e)

{

ReferenceSystem system = new ReferenceSystem();

system.Show();

}

private void btnEXIT\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void btnSignIN\_Click(object sender, EventArgs e)

{

this.Hide();

SIGN\_IN sIGN\_IN = new SIGN\_IN();

sIGN\_IN.Show();

}

private void btnSignUP\_Click(object sender, EventArgs e)

{

this.Hide();

SIGN\_UP sIGN\_UP = new SIGN\_UP();

sIGN\_UP.Show();

}

private void gunaButton6\_Click(object sender, EventArgs e)

{

this.Hide();

Add add = new Add();

add.gunaLabel5.Text = UserName.Text;

add.Show();

}

private void gunaButton7\_Click(object sender, EventArgs e)

{

tableLayoutPanel1.Visible = false;

SearchPanel.Location = new Point(60, 11);

AdminPanel Item = new AdminPanel();

Item.Dock = DockStyle.Fill;

Info.Controls.Add(Item);

PrintCount(Item);

PrintUser(Item);

PrintProperty(Item);

\_obj = this;

}

public void PrintUser(AdminPanel admin)

{

OleDbCommand command = new OleDbCommand("SELECT uUsername, COUNT(\*) FROM Announcement Group BY uUsername", connection);

connection.Open();

OleDbDataReader reader = command.ExecuteReader();

admin.flowLayoutPanel3.Controls.Clear();

Random random = new Random();

while (reader.Read())

{

ElUser Item = new ElUser();

Item.gunaLabel4.Text += reader[0].ToString();

Item.gunaLabel2.Text += reader[1].ToString();

admin.flowLayoutPanel3.Controls.Add(Item);

}

reader.Close();

connection.Close();

}

public void PrintProperty(AdminPanel admin)

{

OleDbCommand command = new OleDbCommand("SELECT \* FROM Announcement", connection);

connection.Open();

OleDbDataReader reader = command.ExecuteReader();

admin.LayoutPanel1.Controls.Clear();

while (reader.Read())

{

ElProperty Item = new ElProperty();

byte[] Photo = (byte[])(reader[16]);

MemoryStream memoryStream = new MemoryStream(Photo);

Item.gunaPictureBox2.Image = Image.FromStream(memoryStream);

Item.gunaLabel1.Text = $"{reader[5]}к {reader[10]} / {reader[11]} м";

Item.gunaLabel2.Text = $"{reader[4]} / {reader[3]}";

Item.gunaLabel3.Text = reader[2].ToString();

Item.IDLabel.Text = reader[0].ToString();

admin.LayoutPanel1.Controls.Add(Item);

}

reader.Close();

connection.Close();

}

public void PrintCount(AdminPanel admin)

{

OleDbCommand command = new OleDbCommand("SELECT COUNT(\*) FROM Announcement", connection);

connection.Open();

admin.CountAds.Text = "0" + command.ExecuteScalar().ToString();

connection.Close();

OleDbCommand command1 = new OleDbCommand("SELECT COUNT(\*) FROM [User]", connection);

connection.Open();

admin.CountUsers.Text = "0" + command1.ExecuteScalar().ToString();

connection.Close();

Random random = new Random();

admin.CountBuyAds.Text = "0" + random.Next(20, 40);

admin.DataTime.Text = DateTime.Now.Day.ToString() + "." + DateTime.Now.Month.ToString() + "." + DateTime.Now.Year.ToString();

}

private void btnALL\_Click(object sender, EventArgs e)

{

PrintAnnouncement("SELECT \* FROM Announcement");

if(InfoPanel.Controls.Count >= 2)

InfoPanel.Controls.RemoveAt(1);

SearchPanel.Location = new Point(21, 11);

tableLayoutPanel1.Visible = true;

btnUser.Visible = true;

btnALL.Visible = false;

}

private void btnUser\_Click(object sender, EventArgs e)

{

PrintAnnouncement($"SELECT \* FROM Announcement Where uUsername='{UserName.Text}'");

if (InfoPanel.Controls.Count >= 2)

InfoPanel.Controls.RemoveAt(1);

SearchPanel.Location = new Point(21, 11);

tableLayoutPanel1.Visible = true;

btnUser.Visible = false;

btnALL.Visible = true;

}

public void PrintAnnouncement(string inquiry)

{

OleDbCommand command = new OleDbCommand(inquiry, connection);

connection.Open();

OleDbDataReader reader = command.ExecuteReader();

tableLayoutPanel1.Controls.Clear();

while (reader.Read())

{

ElProperty Item = new ElProperty();

byte[] Photo = (byte[])(reader[16]);

MemoryStream memoryStream = new MemoryStream(Photo);

Item.gunaPictureBox2.Image = Image.FromStream(memoryStream);

Item.gunaLabel1.Text = $"{reader[5]}к {reader[10]} / {reader[11]} м";

Item.gunaLabel2.Text = $"{reader[4]} / {reader[3]}";

Item.gunaLabel3.Text = reader[2].ToString();

Item.IDLabel.Text = reader[0].ToString();

tableLayoutPanel1.Controls.Add(Item);

}

reader.Close();

connection.Close();

}

private void TextBoxSearch\_Enter(object sender, EventArgs e)

{

if (TextBoxSearch.Text == "Search")

TextBoxSearch.Text = "";

}

private void TextBoxSearch\_Leave(object sender, EventArgs e)

{

if (TextBoxSearch.Text == "")

TextBoxSearch.Text = "Search";

}

private void btnBack\_Click(object sender, EventArgs e)

{

InfoPanel.Controls.RemoveAt(1);

SearchPanel.Location = new Point(21, 11);

tableLayoutPanel1.Visible = true;

}

private void gunaButton4\_Click(object sender, EventArgs e)

{

PrintAnnouncement($"SELECT \* FROM Announcement Where aPrice>={gunaNumeric1.Value} and aPrice <= {gunaNumeric2.Value} or " +

$"aCountRoom>={gunaNumeric4.Value} and aCountRoom <= {gunaNumeric3.Value} or " +

$"aSeller='{ComboBoxSeller.Text}' or " +

$"aFloorEnd={gunaNumeric5.Value}");

}

private void gunaButton1\_Click(object sender, EventArgs e)

{

gunaNumeric1.Value = 0;

gunaNumeric2.Value = 0;

gunaNumeric4.Value = 0;

gunaNumeric3.Value = 0;

ComboBoxSeller.Text = ComboBoxSeller.Items[0].ToString();

gunaNumeric5.Value = 0;

gunaCheckBox1.Checked = false;

PrintAnnouncement("SELECT \* FROM Announcement");

}

private void TextBoxSearch\_TextChanged(object sender, EventArgs e)

{

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.OleDb;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace TheProperty

{

public partial class Add : Form

{

OleDbConnection connection = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=Database.mdb;");

public Add()

{

InitializeComponent();

}

public void ElVisible()

{

Form1 mainWindow = new Form1();

mainWindow.UserName.Text = gunaLabel5.Text;

mainWindow.btnIN.Visible = false;

mainWindow.btnUP.Visible = false;

mainWindow.btnAdd.Visible = true;

mainWindow.btnUser.Visible = true;

if (gunaLabel5.Text == "admin") { mainWindow.btnAdmPanel.Visible = true; }

mainWindow.Show();

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Hide();

ElVisible();

}

Point lastPoint;

private void Add\_MouseDown(object sender, MouseEventArgs e)

{

lastPoint = new Point(e.X, e.Y);

}

private void Add\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

this.Left += e.X - lastPoint.X;

this.Top += e.Y - lastPoint.Y;

}

}

byte[] Photo1;

byte[] Photo2;

byte[] Photo3;

byte[] Photo4;

public byte[] PhotoBD(Guna.UI.WinForms.GunaPictureBox PictureBox)

{

MemoryStream memoryStream = new MemoryStream();

PictureBox.Image.Save(memoryStream, System.Drawing.Imaging.ImageFormat.Jpeg);

byte[] Photo = new byte[memoryStream.Length];

memoryStream.Position = 0;

memoryStream.Read(Photo, 0, Photo.Length);

return Photo;

}

private void btnAdd\_Click(object sender, EventArgs e)

{

try

{

Photo1 = PhotoBD(PictureBox);

Photo2 = PhotoBD(PictureBox2);

Photo3 = PhotoBD(PictureBox3);

Photo4 = PhotoBD(PictureBox4);

OleDbCommand command = new OleDbCommand($"INSERT INTO Announcement (uUsername, aAdress, aFloorEnd, aFloor, aCountRoom, aTipProperty, aSeller, aPhone, aNameOwner, aTotalArea, aKitchenArea, aTipRepairs, aDiscription, aPrice, aPrice30, aPhoto, aPhoto2, aPhoto3, aPhoto4) " +

$"VALUES ({gunaLabel5.Text}, '{TextBoxAdress.Text}', {gunaNumeric3.Value}, {gunaNumeric1.Value}, {gunaNumeric2.Value}, '{ComboBoxTio.Text}', '{ComboBoxSeller.Text}', '{TextBoxPhone.Text}', '{TextBoxNameSell.Text}', '{TextBox1.Text}', '{TextBox2.Text}', '{ComboBox1.Text}', '{TextBoxDiscription.Text}', {int.Parse(TextBoxPrice.Text)}, {int.Parse(TextBoxPrice30.Text)}, @photo, @photo2, @photo3, @photo4)", connection);

command.Parameters.AddWithValue("@photo", Photo1);

command.Parameters.AddWithValue("@photo2", Photo2);

command.Parameters.AddWithValue("@photo3", Photo3);

command.Parameters.AddWithValue("@photo4", Photo4);

connection.Open();

command.ExecuteNonQuery();

connection.Close();

ElVisible();

this.Close();

}

catch(NullReferenceException nex)

{

MessageBox.Show(nex.Message, "Messenge", MessageBoxButtons.OK, MessageBoxIcon.Error);

connection.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Messenge", MessageBoxButtons.OK, MessageBoxIcon.Error);

connection.Close();

}

}

private void btnBrowse\_Click(object sender, EventArgs e)

{

try

{

OpenFileDialog efd = new OpenFileDialog();

if (efd.ShowDialog() == DialogResult.OK)

PictureBox.Image = Image.FromFile(efd.FileName);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Message", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void gunaAdvenceButton1\_Click(object sender, EventArgs e)

{

try

{

OpenFileDialog efd = new OpenFileDialog();

if (efd.ShowDialog() == DialogResult.OK)

PictureBox2.Image = Image.FromFile(efd.FileName);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Message", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void gunaAdvenceButton2\_Click(object sender, EventArgs e)

{

try

{

OpenFileDialog efd = new OpenFileDialog();

if (efd.ShowDialog() == DialogResult.OK)

PictureBox3.Image = Image.FromFile(efd.FileName);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Message", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void gunaAdvenceButton3\_Click(object sender, EventArgs e)

{

try

{

OpenFileDialog efd = new OpenFileDialog();

if (efd.ShowDialog() == DialogResult.OK)

PictureBox4.Image = Image.FromFile(efd.FileName);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Message", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void gunaLabel9\_Click(object sender, EventArgs e)

{

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.OleDb;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace TheProperty

{

public partial class SIGN\_IN : Form

{

OleDbConnection connection = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=Database.mdb;");

public SIGN\_IN()

{

InitializeComponent();

this.ActiveControl = Label;

}

private void gunaLabel8\_Click(object sender, EventArgs e)

{

this.Hide();

SIGN\_UP sIGN\_UP = new SIGN\_UP();

sIGN\_UP.Show();

}

private void btnClose\_Click(object sender, EventArgs e)

{

this.Hide();

Form1 mainWindow = new Form1();

mainWindow.UserName.Text = TextBoxUser.Text;

mainWindow.Show();

}

private void btnSignIN\_Click(object sender, EventArgs e)

{

try

{

Form1 mainWindow = new Form1();

if (TextBoxUser.Text != "Username" && TextBoxPassword.Text != "Password")

{

OleDbDataAdapter oleDbDataAdapter = new OleDbDataAdapter("SELECT \* FROM [User] WHERE uUsername='" + TextBoxUser.Text + "' AND uPassword='" + TextBoxPassword.Text + "'", connection);

DataTable dataTable = new DataTable();

oleDbDataAdapter.Fill(dataTable);

if (dataTable.Rows.Count != 1)

{

ErrorPanel.Visible = true;

Messenge.ForeColor = Color.Red;

Messenge.Text = "Пользователь не найден";

}

else

{

mainWindow.UserName.Text = TextBoxUser.Text;

mainWindow.btnAdd.Visible = true;

mainWindow.btnIN.Visible = false;

mainWindow.btnUP.Visible = false;

mainWindow.btnUser.Visible = true;

if (TextBoxUser.Text == "admin") { mainWindow.btnAdmPanel.Visible = true; }

mainWindow.Show();

this.Hide();

}

}

else

{

if (TextBoxPassword.Text == "Password")

TextBoxPassword.BorderColor = Color.Red;

if (TextBoxUser.Text == "Username")

TextBoxUser.BorderColor = Color.Red;

ErrorPanel.Visible = true;

Messenge.ForeColor = Color.Red;

Messenge.Text = "Есть пустые поля";

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Messenge", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

Point lastPoint;

private void SIGN\_IN\_MouseDown(object sender, MouseEventArgs e)

{

lastPoint = new Point(e.X, e.Y);

}

private void SIGN\_IN\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

this.Left += e.X - lastPoint.X;

this.Top += e.Y - lastPoint.Y;

}

}

private void TextBoxUser\_Enter(object sender, EventArgs e)

{

if (TextBoxUser.Text == "Username")

{

TextBoxUser.Text = "";

TextBoxUser.ForeColor = Color.Black;

}

}

private void TextBoxUser\_Leave(object sender, EventArgs e)

{

if (TextBoxUser.Text == "")

{

TextBoxUser.Text = "Username";

TextBoxUser.ForeColor = Color.Gray;

}

}

private void TextBoxPassword\_Enter(object sender, EventArgs e)

{

if (TextBoxPassword.Text == "Password")

{

TextBoxPassword.Text = "";

TextBoxPassword.ForeColor = Color.Black;

}

}

private void TextBoxPassword\_Leave(object sender, EventArgs e)

{

if (TextBoxPassword.Text == "")

{

TextBoxPassword.Text = "Password";

TextBoxPassword.ForeColor = Color.Gray;

}

}

}

}