**ПРОФЕСИОНАЛНА ГИМНАЗИЯ ПО МЕХАНОТЕХНИКА, ЕЛЕКТРОТЕХНИКА, ТЕЛЕКОМУНИКАЦИИ **

**И ТРАНСПОРТ – ШУМЕН**

**ДИПЛОМНА РАБОТА**

**НА**

**Селим Сеферов 12a КЛАС**

**СПЕЦИАЛНОСТ : ПРИЛОЖЕН ПРОГРАМИСТ**

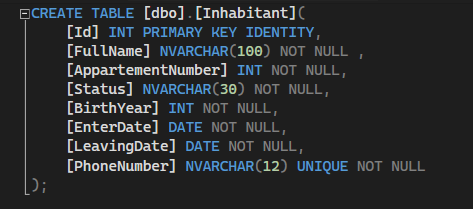
**ТЕМА:**

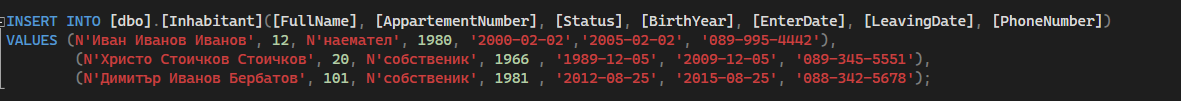
**Разработка на информационна система „Домоуправител“**

**Изговил: ................... Ръководител: ...................**

**/Селим Сеферов 12А/ /инж. Васил Бояджиев/**

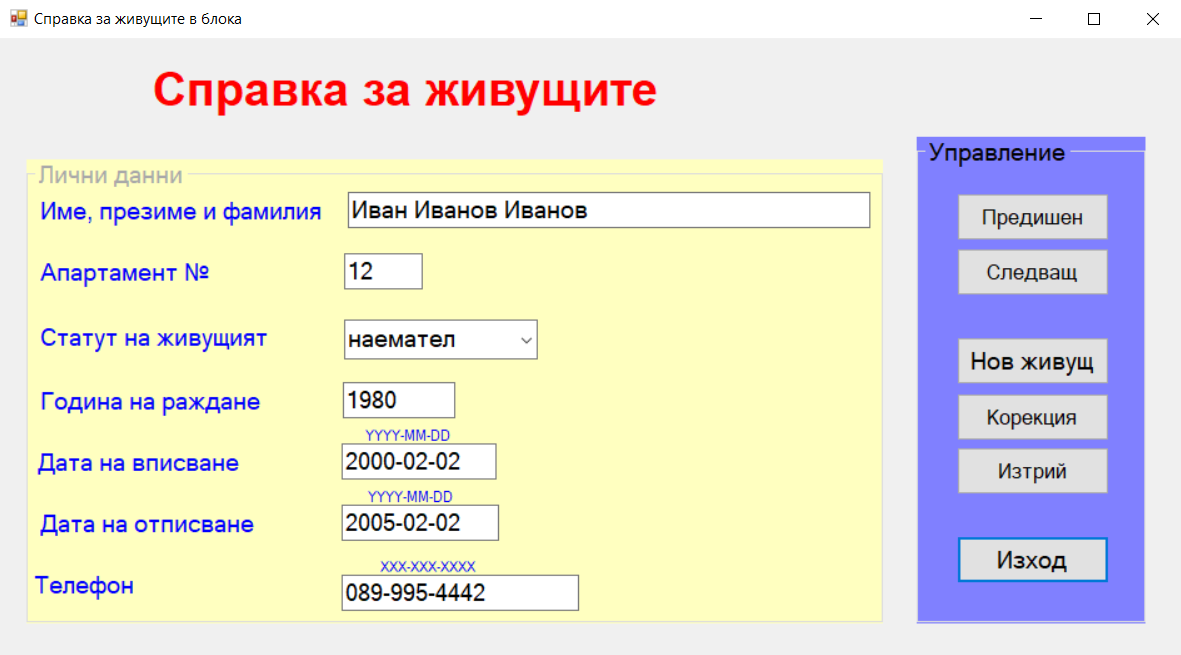
* **Преглед на информационната система „Домоуправител“**
  + В този раздел ще представим общ преглед на ключовите компоненти на информационната система "Домоуправител", като ще разгледаме нейната цел, функционалности и архитектура.
* **Цел на системата**
  + Информационната система "Домоуправител" е проектирана и разработена с цел подпомагане на управлението на апартаментни сгради и финансовите аспекти на такава дейност. Целта й е да предостави на домоуправителите инструменти за ефективно и удобно управление на данните за жителите, апартаментите и финансовите транзакции.
* **Функционалности**
  + Информационната система "Домоуправител" предоставя разнообразни функционалности, които обхващат различни аспекти на управлението на апартаментните сгради:
  + **Управление на жители**: Потребителите могат да добавят, редактират и изтриват данни за жителите, включително пълно име, номер на апартамент, статус, година на раждане и дати на настаняване и напускане.
  + **Управление на апартаментите**: Системата позволява асоцииране на жители към апартаменти, както и въвеждане на данни за статус и дати на настаняване и напускане на апартаментите.
  + **Административни функционалности**: Потребителите имат възможност да генерират месечни фактури за такси, да следят плащанията на жителите и да управляват финансовите транзакции.
  + **Генериране на доклади и статистики**: Системата предоставя възможност за създаване на разнообразни доклади и статистики, които да помогнат за анализ и вземане на решения.
* **Архитектура**
  + Информационната система "Домоуправител" е базирана на клиент-сървър архитектурен модел. Тя включва следните компоненти:
  + **База данни**: Използва се MS SQL база данни за съхранение на информацията за жителите, апартаментите и финансовите транзакции.
  + **Бизнес логика**: Бизнес логиката на системата управлява операциите, свързани със създаване, редакция и изтриване на данни, както и генерирането на доклади и статистики.
  + **Потребителски интерфейс**: Потребителският интерфейс е базиран на Windows Forms технологията. Той предоставя на потребителите възможност да въвеждат и преглеждат данните чрез интуитивни форми и контроли.
  + **Обобщение**
    - Информационната система "Домоуправител" е създадена с цел да облекчи управлението на апартаментни сгради и да осигури лесен и ефективен начин за въвеждане, анализ и обработка на данните. В следващите раздели ще разгледаме по-подробно архитектурата, процеса на разработка и реализацията на системата.
* **Преглед на информационната система „Домоуправител“**
  + В този раздел ще разгледаме подробно архитектурата и техническите детайли на информационната система "Домоуправител". Ще разгледаме какви технологии и инструменти са използвани за създаването й и как те се взаимодействат помежду си.
* **Архитектурен модел**
  + Информационната система "Домоуправител" е проектирана със силно акцентиране върху ролята на администратора. Този модел акцентира на управлението и контрола на данните и операциите чрез администраторската роля.
* **Технически компоненти**
  + Информационната система "Домоуправител" използва следните технологии и инструменти:
  + **.NET Framework 4.7.2**: Системата е разработена с използването на .NET Framework 4.7.2, който предоставя основни библиотеки и инструменти за разработка на Windows приложения.
  + **Windows Forms**: За създаване на потребителския интерфейс са използвани Windows Forms. Този инструмент предоставя графични елементи като форми, бутони и текстови полета, които позволяват на администратора да управлява системата.
  + **.NET Framework Data Provider for SQL Server**: За връзка с базата данни се използва .NET Framework Data Provider за SQL Server. Този компонент позволява създаването на връзка със SQL база данни, изпращане на заявки и получаване на резултатите.
  + **MS SQL база данни**: За съхранение на данните се използва MS SQL база данни. Тя съхранява информацията за апартаментите, жителите и финансовите транзакции.
* **Потребителски интерфейс**
  + Потребителският интерфейс на системата "Домоуправител" е създаден чрез Windows Forms. Той предоставя на администратора лесен и интуитивен начин за управление на данните. Интерфейсът включва различни форми и контроли, като текстови полета, бутони и таблица, които улесняват администратора при навигацията и въвеждането на информация.
  + **Обобщение**
    - Информационната система "Домоуправител" се фокусира върху ролята на администратора и управлението на данните. Техническият компонент на системата използва .NET Framework 4.7.2 и Windows Forms за създаване на административния интерфейс. Взаимодействието с базата данни става чрез .NET Framework Data Provider за SQL Server. В следващите раздели ще разгледаме процеса на разработка и внедряване на системата.
* **Архитектура на системата**
  + В този раздел ще разгледаме архитектурните аспекти на информационната система "Домоуправител". Ще разгледаме високо ниво на архитектурния обзор, компонентите и модулите, както и диаграми на потока на данни в системата.
* **Високо ниво на архитектурен обзор**
  + Информационната система "Домоуправител" е разработена като клиент-сървър приложение, където клиентският компонент е базиран на Windows Forms, а сървърният компонент е базата данни на MS SQL.
* **Компоненти и модули**
  + Системата е разделена на няколко ключови компонента и модули:
  + **Потребителски интерфейс (Windows Forms)**: Този компонент предоставя визуален интерфейс за администратора, който може да въвежда, редактира и преглежда информация за апартаментите, жителите и финансовите транзакции.
  + **Бизнес логика**: Този модул съдържа логиката за обработка на данните.
  + **Данни и база данни**: Този модул съхранява данните, необходими за функционирането на системата. Тук се включват таблиците с информация за апартаментите, жителите и т.н.
  + **Диаграми на потока на данни**
    - Диаграмите на потока на данни са полезни инструменти за представяне на как данните се движат през системата. В случая на информационната система "Домоуправител", можем да представим следните диаграми на потока на данни:
  + **Диаграма на потока на данни за добавяне на нов жител**:
    - Администраторът използва потребителския интерфейс, за да въведе информация за нов жител.
    - Информацията се изпраща към бизнес логиката.
    - Бизнес логиката валидира информацията и я обработва.
    - Данните се записват в базата данни.
  + **Диаграма на потока на данни за преглед на финансови транзакции**:
    - Администраторът избира опцията за преглед на финансовите транзакции.
    - Заявката се изпраща към бизнес логиката.
    - Бизнес логиката извлича информацията от базата данни.
    - Информацията се връща обратно на администраторския интерфейс и се показва на екрана.
  + **Обобщение**
    - Архитектурата на информационната система "Домоуправител" следва модела клиент-сървър и включва компоненти като потребителски интерфейс, бизнес логика и база данни. Диаграмите на потока на данни ни дават представа как данните се движат през системата. В следващите раздели ще разгледаме дълбоките детайли на всяка компонента и модул, както и тяхната функционалност.
* **Дизайн на базата данни**
  + В информационната система "Домоуправител" е изграден добре структуриран дизайн на базата данни, който осигурява ефективно съхранение и управление на информацията. Тук ще разгледаме ключовите аспекти на дизайна на базата данни.
* **Диаграма на съотношение на сущности (ERD)**
  + Дизайнът на базата данни започва със създаването на диаграма на съотношение на сущности (Entity-Relationship Diagram, ERD). Тази диаграма представя връзките между различните сущности в системата и тяхната структура. В случая на системата "Домоуправител", диаграмата на съотношение на сущности описва връзките между таблиците и техните атрибути, като например връзката между таблицата "Inhabitant" и нейните характеристики.
* **Схема на базата данни и таблици**
  + Базата данни на системата се състои от различни таблици, които позволяват съхранение на информация за жителите, техните апартаменти, статуси и контактни данни. Всяка таблица представлява отделна сущност и има свои атрибути. Например, таблицата "Inhabitant" съдържа информация за жителите, включително техния идентификационен номер, имена, номер на апартамент, статус и други.
* **Данни за цялостност и ограничения**
  + За да се гарантира коректността и надеждността на данните, в базата данни са приложени различни ограничения и правила за цялостност. Например, полето "Id" в таблицата "Inhabitant" е зададено като PRIMARY KEY, което гарантира уникалност на идентификационните номера на жителите. Полето "PhoneNumber" се налага като UNIQUE, за да се избегне съхранението на повтарящи се телефонни номера.
  + Този дизайн на базата данни осигурява ефективно съхранение, бърз достъп и правилно управление на информацията в системата "Домоуправител".
* **Използвани технологии и инструменти**
  + При разработката на информационната система "Домоуправител" бяха използвани различни технологии и инструменти, които съчетаваха се в интегрирана среда за разработка. Този раздел ще представи основните технологии и инструменти, използвани по време на създаването на системата.
  + **.NET Framework и C#**
    - .NET Framework е платформа за разработка на софтуер, предоставяща широка гама от библиотеки и инструменти, които подпомагат разработката на разнообразни приложения. В случая на информационната система "Домоуправител", .NET Framework бе използван като основна платформа, а програмният език C# бе избран за разработката на приложението.
  + **SQL Server и ADO.NET**
    - За съхранение и управление на данните в информационната система беше използван Microsoft SQL Server - релационна база данни, предоставяща надежден и ефективен начин за съхранение на информация. За взаимодействие с базата данни бе използван ADO.NET (Active Data Objects .NET) - технология за достъп и манипулация на данни в .NET приложения.
  + **Windows Forms**
    - За създаване на графичния интерфейс на приложението беше използвана технологията Windows Forms. Тази технология предоставя гъвкави инструменти за създаване на потребителски интерфейси с помощта на визуален дизайнер. Този подход позволи на разработчиците да създават интуитивни и функционални потребителски интерфейси.
  + **Интегрирана среда за разработка (IDE)**
    - За създаване на приложението бе използвана интегрираната среда за разработка Visual Studio. Тази среда предоставя инструменти за създаване, редакция и отстраняване на грешки в кода, както и интуитивни визуални дизайнери за създаване на потребителски интерфейси.
  + **Графичен дизайн и документация**
    - За създаване на графични елементи, икони и други графични ресурси бе използван софтуер за графичен дизайн, като например Adobe Photoshop. Също така, за съставянето на този документ бе използван Microsoft Word, като бяха използвани различни форматиране и стилове за създаване на четлив и професионален документ.
    - Използването на тези технологии и инструменти позволи на екипа по разработка да създаде функционална, стабилна и удобна за употреба информационна система "Домоуправител", която успешно отговаряше на изискванията на потребителите.
* **Дизайн на потребителския интерфейс**
  + Потребителският интерфейс е ключов аспект за успешната употреба на информационната система "Домоуправител". В този раздел ще разгледаме дизайна на потребителския интерфейс, включително оформлението на екраните, навигацията, използваните форми и контроли, както и усилията за подобряване на потребителския опит.
  + **Оформление на екраните и навигацията**
    - При проектирането на потребителския интерфейс бе обърнато внимание на четливостта, ефективността и интуитивното навигиране. Всяка основна функционалност на системата бе представена чрез отделни екрани, които бяха съчетани в логични групи.
    - Добавени бутони за бърз достъп до често използвани опции.
  + **Форми и контроли, използвани**
    - В приложението бе използвана палитра от стандартни и специализирани контроли на Windows Forms, за да се осигури комфорт на потребителите при взаимодействието с системата. Някои от използваните контроли включваха:
  + **Текстови полета и числови полета:** За въвеждане на текстова информация и числови стойности, например имена, номера на апартаменти и телефонни номера.
  + **Дата и време контроли:** За избор на дати и времена, като например дати на влизане и напускане на жилище.
  + **Таблични изгледи:** За представяне на списъци с данни, като например списък с жители и техните данни.
  + **Бутони и командни елементи:** За изпълнение на операции като добавяне, редактиране и изтриване на данни.
  + **Застъпване на потребителския опит**
    - При проектирането на потребителския интерфейс бе обърнато внимание и на оптимизацията на потребителския опит. За целта бяха предприети следните стъпки:
  + **Ергономичен дизайн:** Екрани и контроли бяха разположени така, че да осигурят лесен достъп и интуитивно използване.
  + **Валидация на данни:** При въвеждане на информация бяха приложени валидации, за да се гарантира правилното въвеждане на данни и предотвратяване на грешки.
  + **Съобщения за потребителя:** Потребителите бяха информирани със съобщения за успешно изпълнение на операции или при необходимост от корекции.
  + **Интуитивно оформление:** Използването на цветове, икони и лесно разпознаваеми елементи допълнително подобри усещането при работа с приложението.
  + Дизайнът на потребителския интерфейс играе ролята на мост между потребителите и функционалностите на системата, затова голямото внимание, обърнато към него, осигури гладко и удовлетворително използване на информационната система "Домоуправител".
* **Подробности за функционалността**
  + Информационната система "Домоуправител" предоставя разнообразни функционалности, които улесняват ежедневната работа на администратора и оптимизират управлението на сградата и жителите. В този раздел ще разгледаме подробности за основните функционалности на системата.
  + **Управление на жителите**
    - Системата позволява на администратора да въвежда, редактира и изтрива данни за жителите. За всеки жител могат да се въведат следните данни:
    - Име и фамилия
    - Номер на апартамент
    - Статус (наемател, собственик и т.н.)
    - Година на раждане
    - Дата на влизане и напускане на апартамента
    - Телефонен номер
    - Тази функционалност позволява администратора лесно да следи информацията за всеки жител и да извършва необходимите обновления.
  + **Управление на апартаментите**
    - Администраторът може да въвежда и управлява информацията за апартаментите в сградата. Това включва:
    - Номер на апартамент
    - Брой жители в апартамента
    - Площ на апартамента
    - Описание на апартамента (брой стаи, етаж и т.н.)
    - Този модул осигурява организиране и проследяване на данни за всяко жилище в сградата.
  + **Фактуриране и плащания**
    - Системата позволява създаване на фактури за различни видове разходи, свързани с обслужването на сградата. Това включва разходи за поддръжка, електричество, вода и други. Администраторът може да генерира фактури за определен период и да ги изпраща на жителите. Потребителите могат да въведат информация за извършените плащания, което облекчава процеса на фактуриране и следене на плащанията.
    - **Доклади и анализи**
    - Системата предоставя възможност за генериране на различни доклади и анализи. Това включва:
  + **Справка за жителите:** Списък с всички жители и техните данни.
  + **Фактури за плащане:** Списък с издадените фактури за определен период.
  + **Анализ на разходите:** Отчет за разходите по различни категории за определен период.
    - Тези доклади и анализи предоставят ценни данни за администратора и могат да бъдат използвани за вземане на решения за оптимизиране на управлението на сградата.
    - Информационната система "Домоуправител" съчетава тези функционалности, за да предостави пълен инструментариум за управление на сградата и улесняване на административните задачи на администратора.
* **Достъп до данни и бизнес логика**
  + В раздела тук ще разгледаме как системата за управление на жилищни сгради "Домоуправител" осигурява достъп до данни и какви механизми за бизнес логика се използват за обработка на информацията.
  + **Слой за достъп до данни (DAL)**
    - Слойът за достъп до данни (DAL) е отговорен за комуникацията с базата данни. В случая на системата "Домоуправител", този слой използва технологията ADO.NET за установяване на връзка с базата данни на SQL Server. Компонентите в този слой са отговорни за извличане, въвеждане и обновление на данни в базата данни.
  + **Бизнес логика (BLL)**
    - Бизнес логиката (BLL) представлява сърцевината на системата, където се извършва обработката на данни и взаимодействието със слоя за достъп до данни. В този слой се прилагат правилата и логиката, свързани с операциите на системата. Например, при създаване на нов фактура, бизнес логиката проверява целостността на данните, изчислява общата сума за плащане и определя статуса на фактурата.
  + **Обработка на изключения и регистриране**
    - Всяка сериозна система трябва да предвиди обработката на грешки и изключения. В системата "Домоуправител" се използва обработка на изключения, за да се предпази приложението от сривове и да се осигури плавно функциониране. Грешките и изключенията се регистрират и логират за последващ анализ и отстраняване.
    - Тези компоненти работят заедно, за да осигурят надежден и ефективен достъп до данните, като в същото време се прилага правилната бизнес логика за обработка на информацията. Обработката на изключенията и регистрирането гарантират, че администраторите и потребителите са информирани за възможни проблеми и грешки в системата.
  + **Сигурност и удостоверяване на самоличността**
    - В този раздел ще разгледаме как системата за управление на жилищни сгради "Домоуправител" осигурява сигурността на данните и какви механизми се използват за удостоверяване на самоличността на потребителите.
  + **Потребителско удостоверяване и разрешаване**
    - Системата "Домоуправител" използва потребителско удостоверяване, което позволява на потребителите да се регистрират и влизат в системата със свои уникални потребителски идентификатори (потребителско име и парола). След успешно влизане в системата, потребителите имат достъп до различни функционалности в зависимост от техните роли.
  + **Контрол на достъпа на база на роли**
    - Системата има различни роли, като например администратор, жител или служител на управление. В зависимост от ролята, потребителите имат различни права и достъп до определени функционалности. Например, администраторът има пълен достъп и може да управлява всички аспекти на системата, докато жителите могат само да преглеждат своите данни и сметки.
  + **Криптиране на данни и най-добри практики за сигурност**
    - За да се осигури сигурност на данните, системата използва криптиране на чувствителна информация като пароли и лични данни. Това предпазва данните от неразрешени достъпи и злоупотреби. Освен това, се прилагат най-добрите практики за сигурност, включително редовни актуализации на системата, защита срещу атаки и мониторинг на активностите.
    - Тези механизми осигуряват, че данните на потребителите са сигурни и достъпът до тях е строго контролиран, за да се предотвратят възможни инциденти на нарушаване на сигурността.
  + **Тестване и качество на софтуера**
    - В този раздел ще разгледаме как системата за управление на жилищни сгради "Домоуправител" се тества, за да се гарантира качеството и коректността на софтуера.
  + **Тестване на модули**
    - Системата се тества на ниво модули, където отделните компоненти и функционалности се изпитват изолирано. За всяка функционалност се създават тестови случаи, които включват различни входни данни и сценарии. Тестовете се изпълняват автоматично, като се проверява дали изходът от системата съвпада с очаквания резултат.
  + **Тестване на интеграцията**
    - След тестването на модулите се извършва тестване на интеграцията, което включва проверка на взаимодействието между различните компоненти и модули на системата. Този тип тестове открива евентуални проблеми при комбиниране на компонентите и уверява се, че те работят ефективно заедно.
  + **Тестване от страна на потребителя**
    - След като модулите и интеграцията са тествани вътрешно, системата се предоставя за тестване и от страна на потребителите. Това може да включва бета тестове или пилотни използвания, при които потребителите използват системата в реални условия и предоставят обратна връзка. Това помага за идентифициране на евентуални проблеми и подобрения, които да бъдат внесени преди финалния продукт.
    - Тестването играе ключова роля за гарантиране на качеството на софтуера и за осигуряване, че системата функционира коректно и отговаря на изискванията на потребителите.
  + **Разгръщане и хостинг**
    - В този раздел ще разгледаме процеса на разгръщане и хостинг на информационната система "Домоуправител".
  + **Стратегии за разгръщане**
    - За да се осигури гладък процес на разгръщане на системата, се използват стратегии за разгръщане. Системата се разгръща последователно и стъпково, като се тестват отделните компоненти и модули преди да бъдат пуснати в продукционна среда. Това помага да се избегнат проблеми и нарушения в работата на системата след разгръщането.
  + **Изисквания към сървъра**
    - За успешното разгръщане и хостинг на системата са нужни специфични изисквания към сървъра, на който ще бъде инсталирана. Сървърът трябва да поддържа .NET Framework и C#, както и да има инсталиран SQL Server със съответната база данни за системата. Сървърът трябва да бъде настроен за поддръжка на Windows Forms приложения.
  + **Инсталация и конфигурация**
    - Инсталацията на системата се извършва на сървъра със специален инсталационен пакет. При инсталацията се извършва конфигурация на връзката с базата данни, както и на други настройки, необходими за правилната работа на системата. След успешната инсталация и конфигурация системата е готова да бъде използвана от потребителите.
    - Разгръщането и хостингът на системата са ключови фази в жизнения цикъл на софтуера. Те гарантират, че системата е достъпна и функционираща в продукционна среда, като същевременно се поддържа нейната стабилност и сигурност.
* **Поддръжка и поддържане**
  + В този раздел ще разгледаме важни аспекти на поддръжката и поддържането на информационната система "Домоуправител".
  + **План за непрекъснато поддържане**
    - За да се гарантира непрекъснатата работа на системата, е важно да се изготви план за поддръжка. Този план включва редица стъпки и процедури, които ще се използват за поддържане на системата в работоспособно състояние. Това включва редовна проверка на системата, актуализации и оптимизации.
  + **Следене на грешки и решаване на проблеми**
    - Процесът на следене и отстраняване на грешки е важен за поддържането на системата. Използването на инструменти за проследяване на грешки позволява на екипа за поддръжка да идентифицира и следи проблемите. Когато се открие грешка, се предприемат действия за нейното решаване, което може да включва поправка на кода и публикуване на актуализация.
  + **Потребителска поддръжка и обучение**
    - Потребителската поддръжка е от съществено значение, за да се гарантира, че потребителите на системата имат нужната помощ и ресурси за използване на системата ефективно. Това включва предоставяне на документация, обучение и поддръжка на потребителите. Обучението на нови потребители е важна част от процеса, за да се гарантира, че системата се използва правилно и оптимално.
    - Поддръжката и поддържането на системата "Домоуправител" осигуряват нейната дългосрочна функционалност и стабилност. Редовната проверка, подновяване и поддръжка са от съществено значение за удовлетворението на потребителите и успешното използване на системата.
* **Оптимизация на производителността**
  + В този раздел ще разгледаме важни аспекти на оптимизацията на производителността на информационната система "Домоуправител".
  + **Идентифициране на забавяния**
    - За да се подобри производителността на системата, е важно да се идентифицират забавянията и точките на бутълнек. Това може да се осъществи чрез измерване на времето за зареждане на страници, изпълнение на заявки към базата данни и други операции. Идентифицирането на забавянията позволява на екипа за разработка да се фокусира върху оптимизацията на конкретни части от системата.
  + **Тестване и оптимизация на производителността**
    - Редовното тестване на производителността е ключов процес за оптимизацията на системата. Тестовете на производителността могат да бъдат използвани за оценка на времето за отговор на системата при натоварване. Това позволява на екипа за разработка да идентифицира проблемни зони и да предприеме необходимите стъпки за подобрение.
  + **Обмисляне на скалируемостта**
    - Скалируемостта е важен аспект на производителността, особено ако се очаква нарастване на броя на потребителите. Проектирането на системата с мисъл за скалируемост позволява лесно добавяне на ресурси и инфраструктура, когато се налага. Това може да включва използването на облачни услуги и разпределени архитектурни подходи.
    - Оптимизацията на производителността на системата "Домоуправител" е важна за осигуряване на бърз и ефективен достъп до данни и функционалности. Идентифицирането на забавяния, тестването на производителността и обмислянето на скалируемостта са ключови стъпки за подобрение на работата на системата.
    - Управление на проекта
    - В раздела за управление на проекта беше фокусирано върху индивидуалния характер на процеса по създаване на информационната система "Домоуправител". Тъй като проектът беше разработван от един човек, не беше налице формален екип. Този сценарий отразява личните усилия и ангажимент на един индивидуален разработчик.
    - Проектният график беше стратегически изграден с установяване на милници, които служеха като ключови точки за напредъка на проекта. Този подход позволи на разработчика да следи и оценява развитието на системата в рамките на поставените срокове и цели. Милниците допълнително допринесоха за управлението на процеса на разработка и създадоха възможност за контрол и анализ на напредъка.
    - Въпреки липсата на екип и индивидуалния характер на работата, усилията на един човек доведоха до текущото състояние на системата. Ограничените ресурси, включително времето и знанията, представляваха предизвикателство, което изискваше адаптация и решителност за постигане на поставените цели.
    - Този опит предостави ценни поуки за ефективното управление на индивидуални проекти и показва как дори в случаи на самостоятелна разработка може да бъде постигнат значителен напредък. Това също така подчертава важността на стратегическият подход към управлението на проекта и използването на милници за проследяване на развитието му.
* **Литература**
  + <https://www.syncfusion.com/forums/145624/winforms-controls-in-net-framework-4-7-and-4-8>
  + <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/overview/?view=netdesktop-7.0>
  + <https://developercommunity.visualstudio.com/t/cant-use-mdf-file-or-logldf-that-was-created-on-an/1547664>
  + <https://docs.oracle.com/en-us/iaas/mysql-database/doc/getting-started-mysql-database-service.html#GUID-3C1DA346-CDA2-442E-9D22-A7C244B9DF04>
  + <https://dobritesasedi.bg/resources/domoupravitel-narachnik.pdf>
  + <https://delarbg.com/d48/wp-content/uploads/2018/04/NUES-Web.pdf>
* **Приложения**
  + Примерни SQL заявки
  + Снимки на потребителския интерфейс
  + Откъси от кода
* SQL заявки

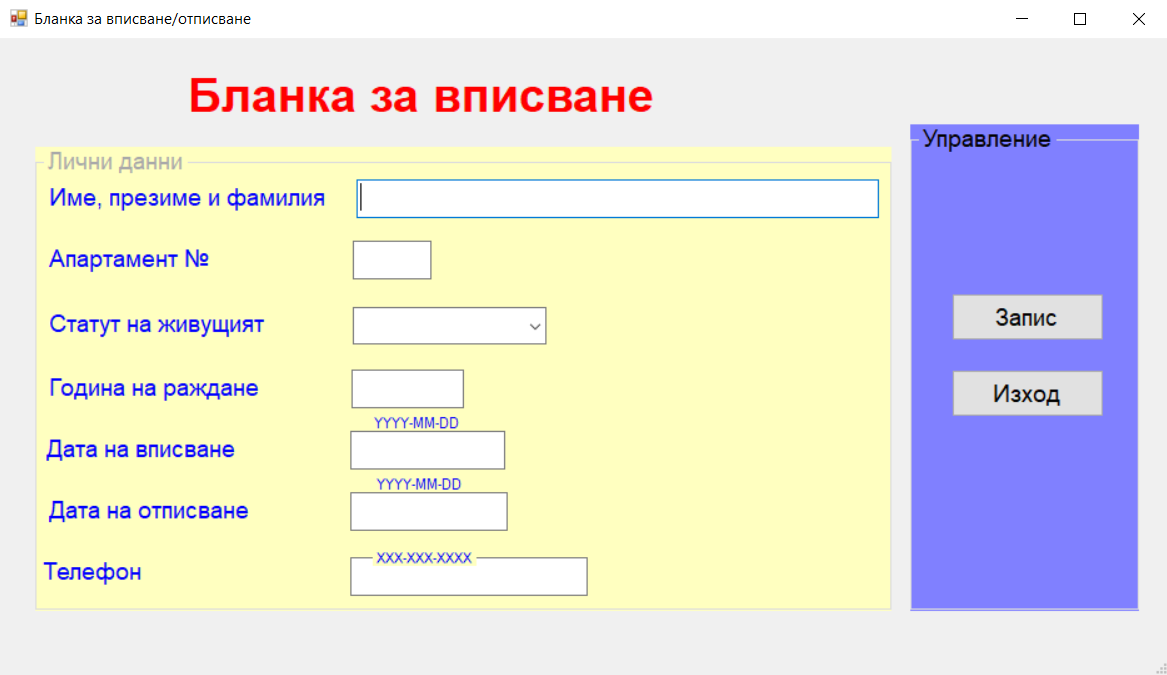
****

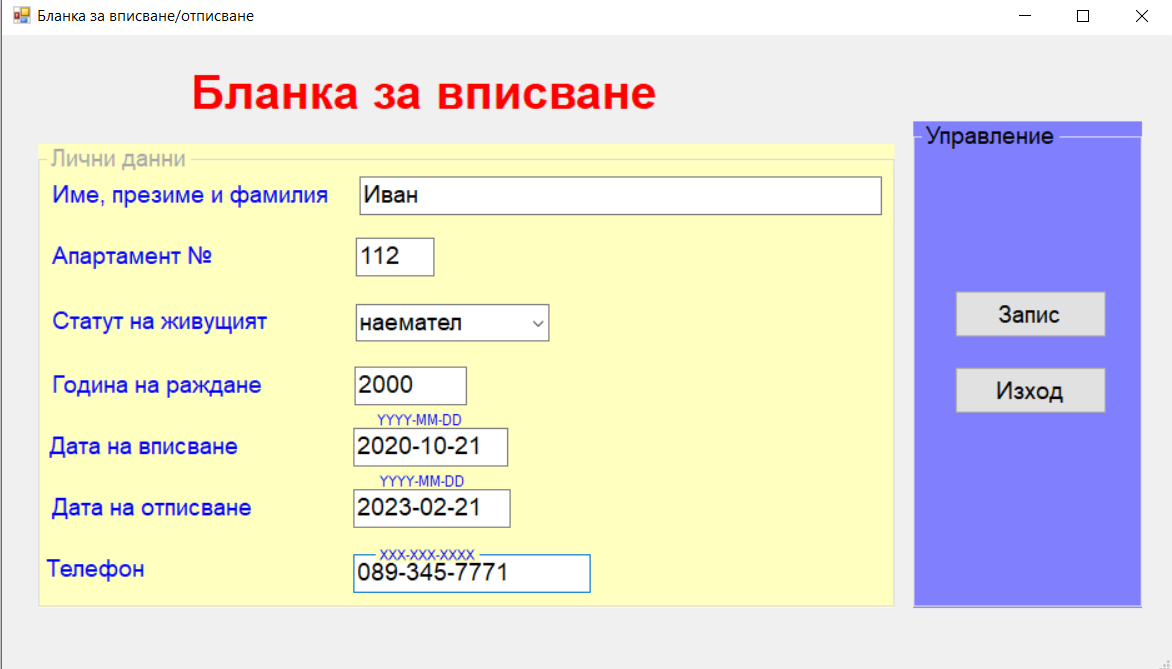
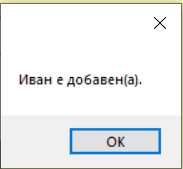
****

****

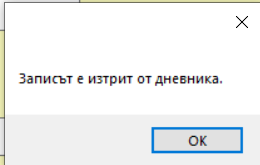
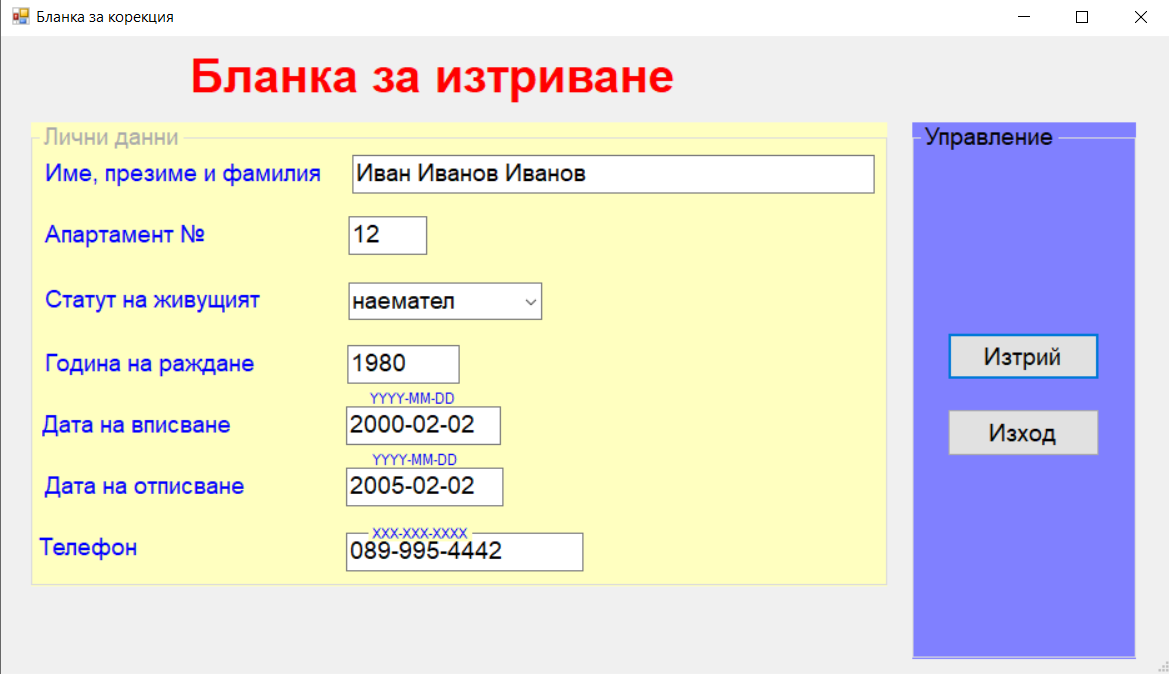
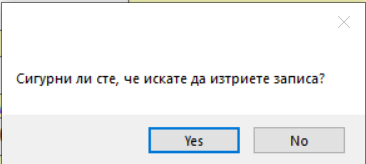
* **Потребителския интерфейс**

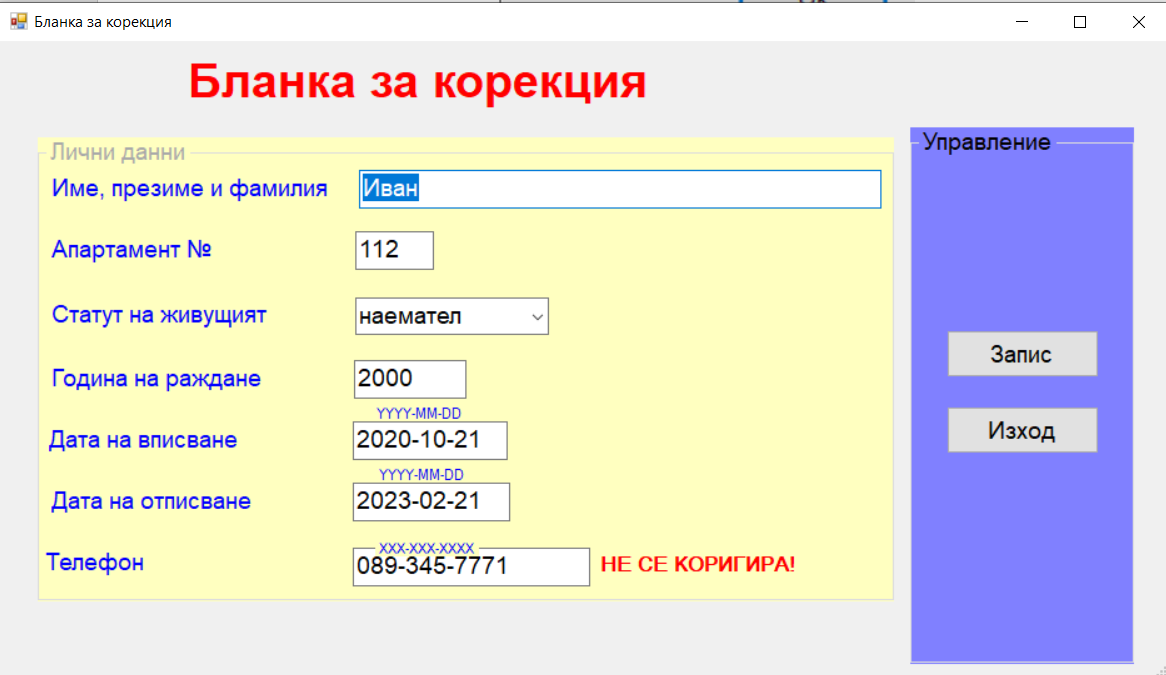
****

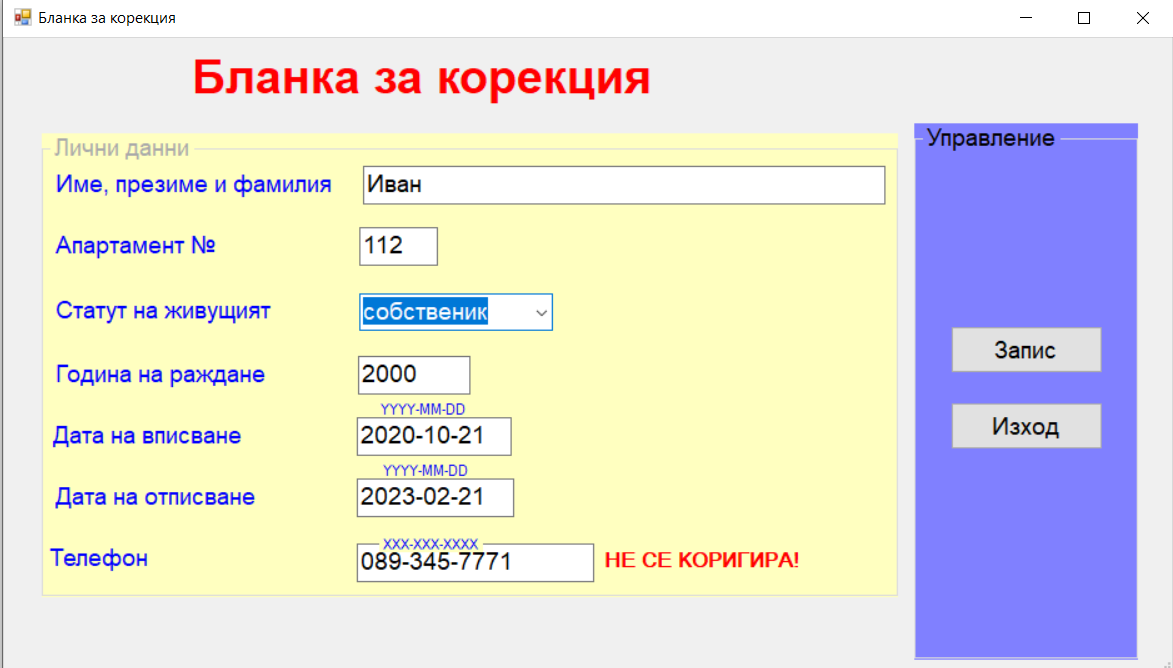
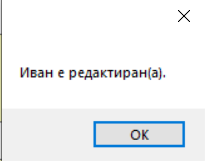
* + **Добавяне**

****

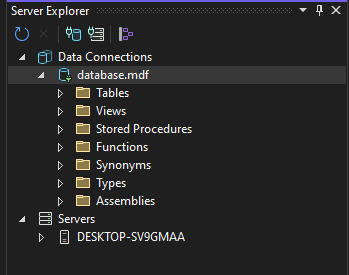
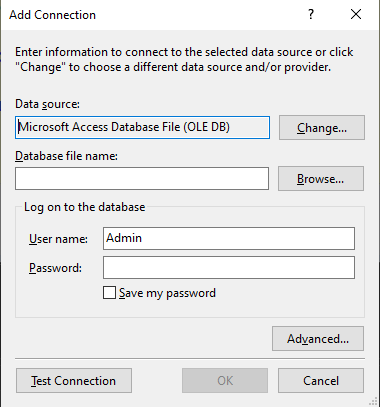
* + **Изтриване**

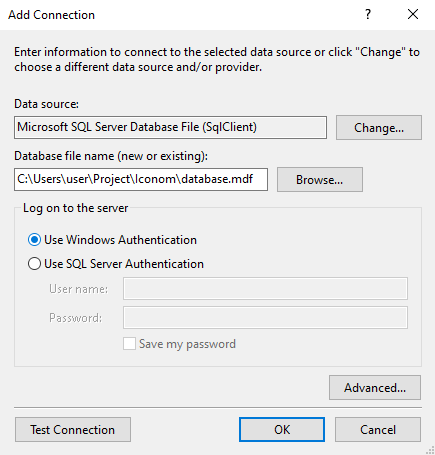
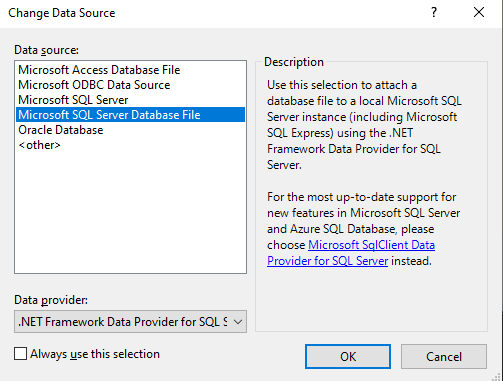
****

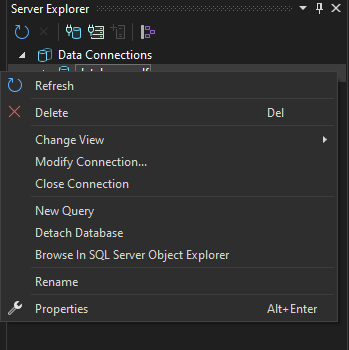
* + **Корекция**

****

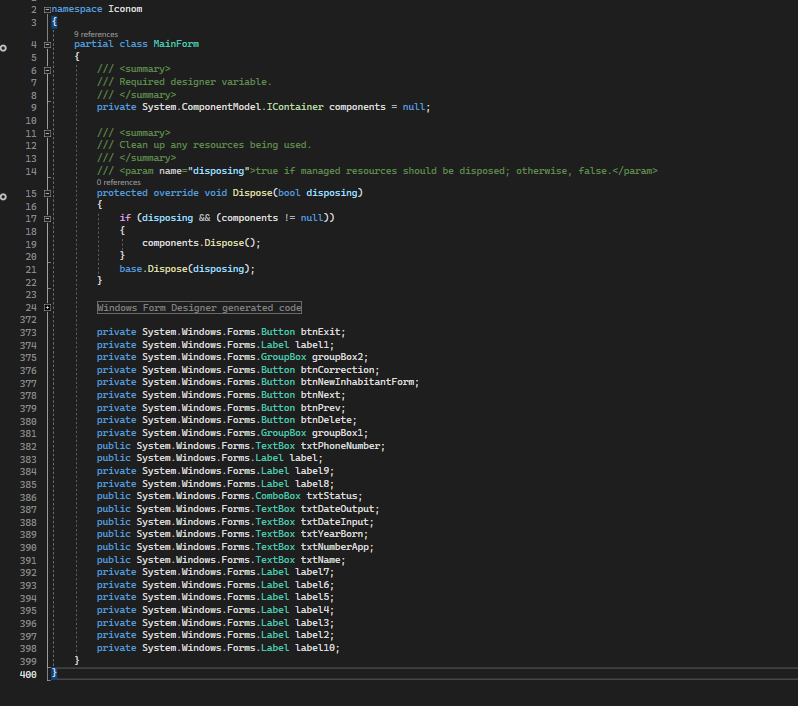
* **Свързване към базата (използвайки .mdf файл)**

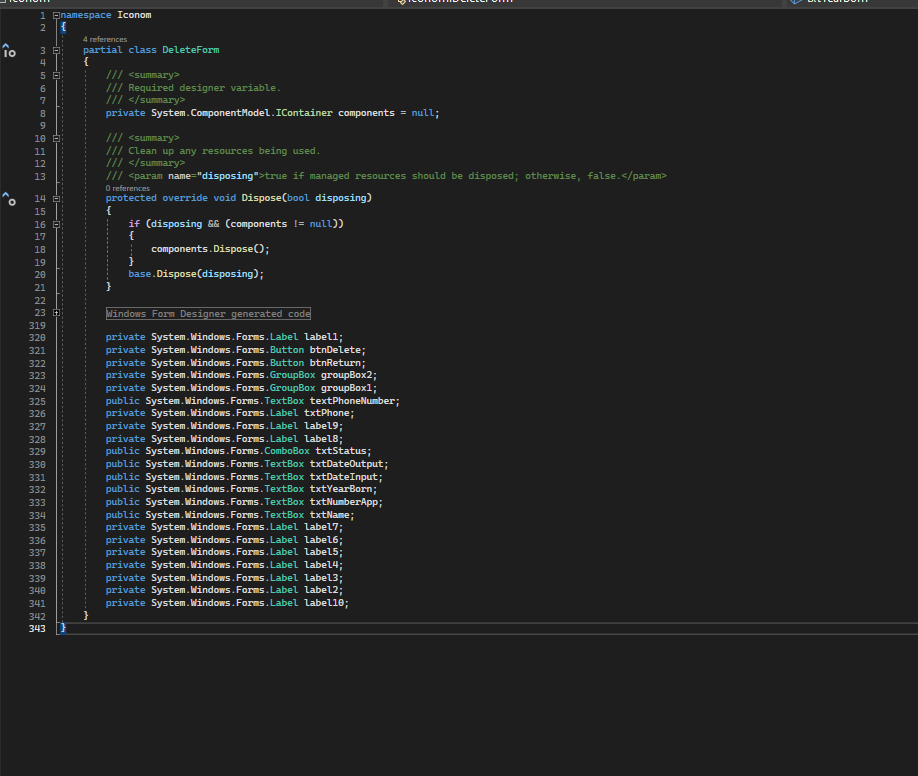
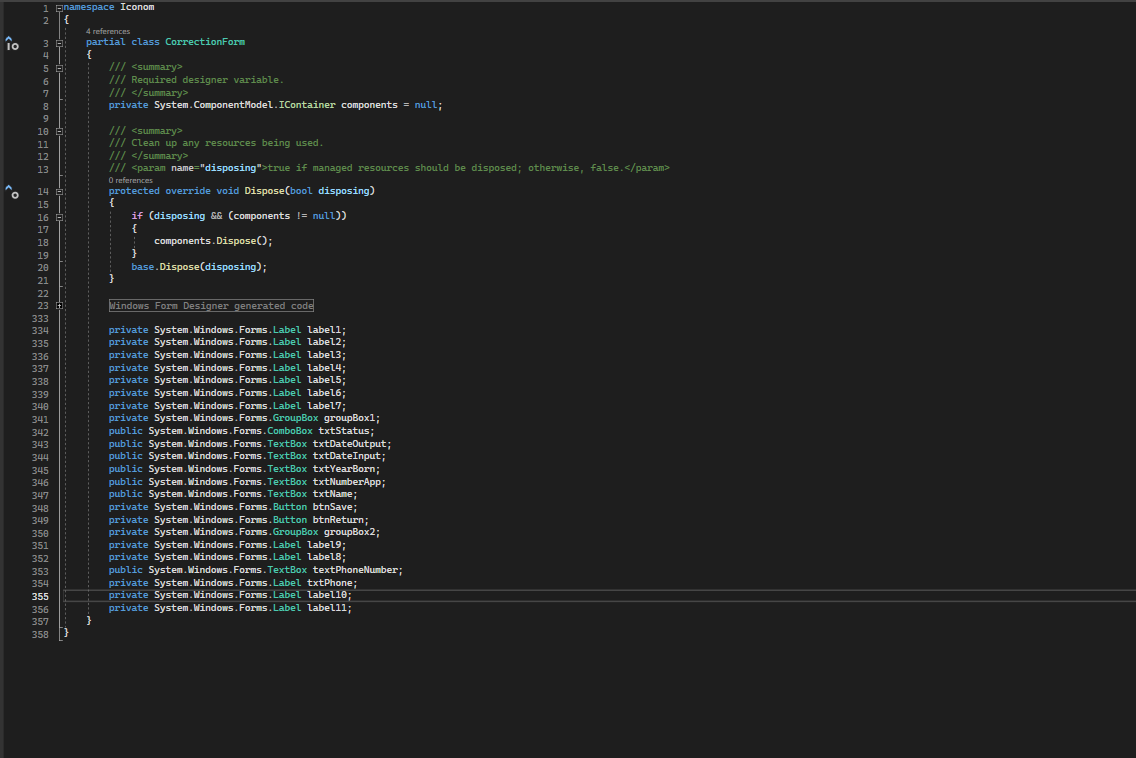
****

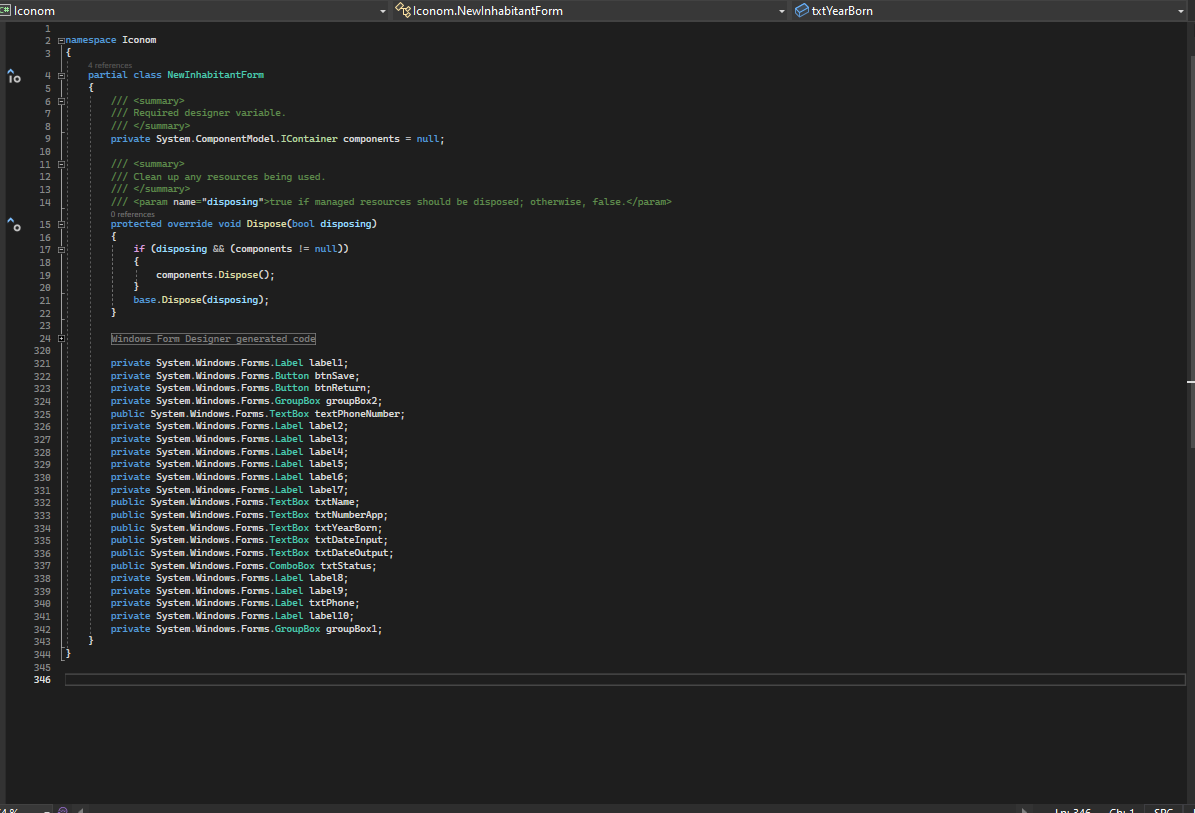
* + **Сменяме default настройките, за да може да използваме .mdf файл и избираме локацията на файлаДобавяне на SQL QUERY**

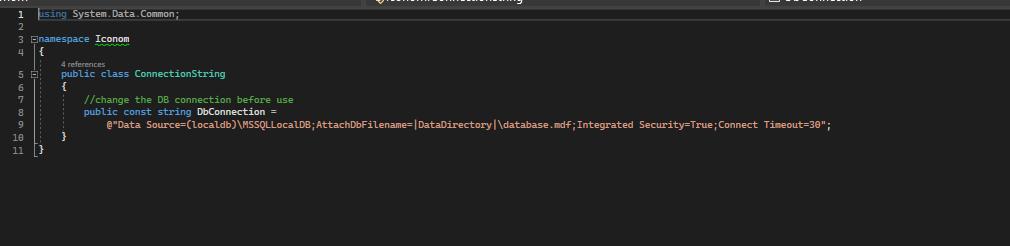
****

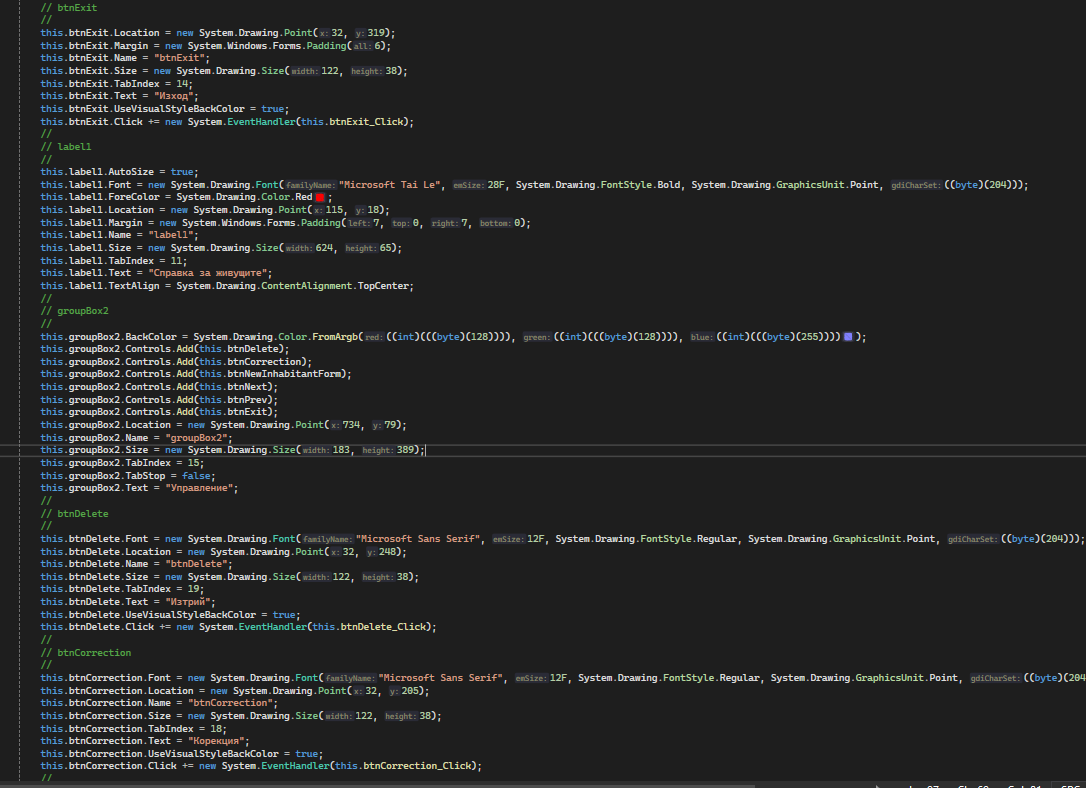
* **Код**

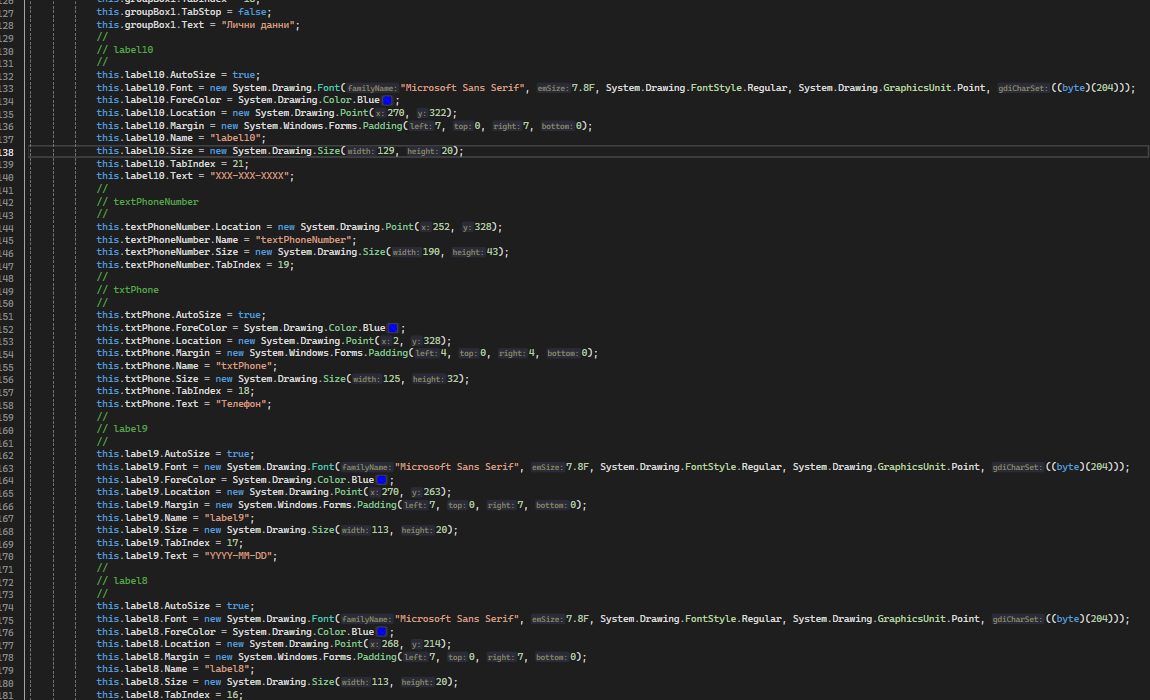
****

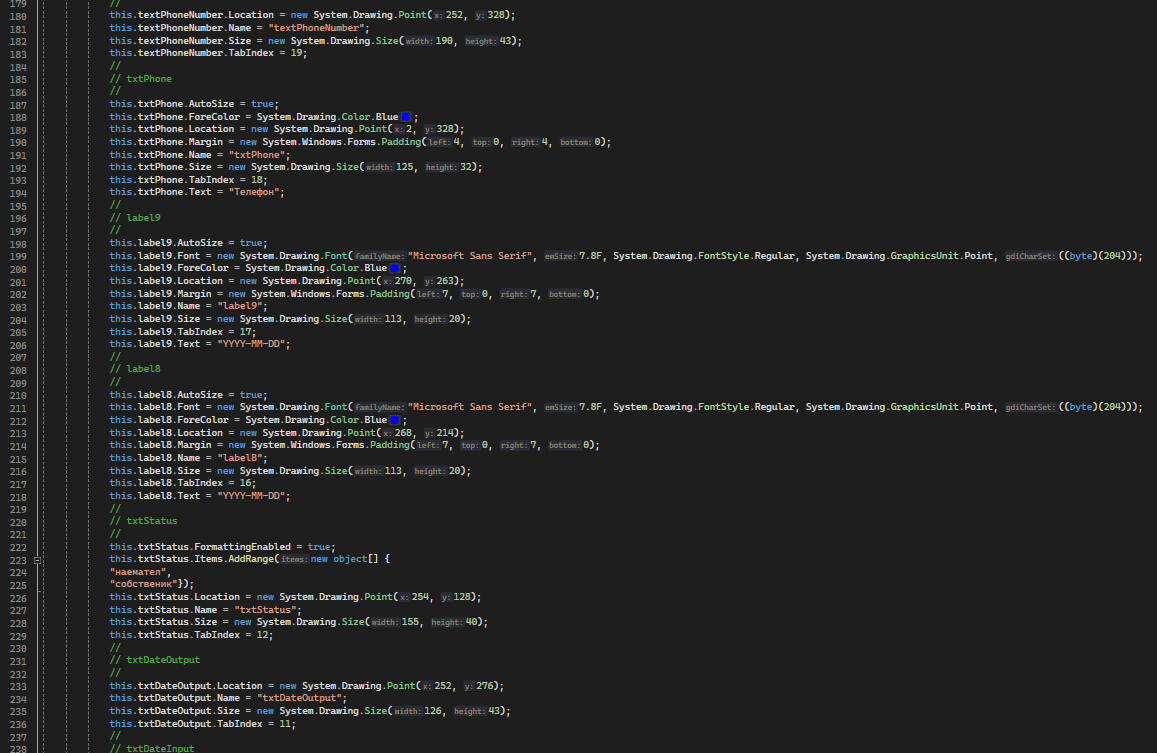
* + - ****
* ****

****

* **Connection string**

****

****

****