Содержание

[Введение 3](#_Toc453787782)

[1 Объектно-ориентированный анализ и проектирование системы 5](#_Toc453787783)

[1.1 Сущность задачи 5](#_Toc453787784)

[2 Вычислительная система 8](#_Toc453787785)

[2.1 Требования к аппаратным и операционным ресурсам 8](#_Toc453787786)

[2.2 Инструменты разработки 8](#_Toc453787787)

[3 Проектирование задачи 10](#_Toc453787788)

[3.1 Требования к приложению 10](#_Toc453787789)

[3.2 Концептуальный прототип 10](#_Toc453787790)

[3.3 Организация данных 11](#_Toc453787791)

[3.4 Функции: логическая и физическая организация 13](#_Toc453787792)

[3.5 Проектирование справочной системы приложения 14](#_Toc453787793)

[4 Описание программного средства 16](#_Toc453787794)

[4.1 Общее сведения 16](#_Toc453787795)

[4.2 Функциональное назначение 16](#_Toc453787796)

[4.3 Входные и выходные данные 16](#_Toc453787797)

[5 Методика испытаний 18](#_Toc453787798)

[5.1 Технические требования 18](#_Toc453787799)

[5.2 Порядок проведения испытаний 18](#_Toc453787800)

[5.2.1 Функциональное тестирование 18](#_Toc453787801)

[5.2.2 Полное тестирование 22](#_Toc453787802)

[6 Применение 24](#_Toc453787803)

[6.1 Назначение программы 24](#_Toc453787804)

[6.2 Условия применения 24](#_Toc453787805)

[6.3 Справочная система 24](#_Toc453787806)

[Заключение 26](#_Toc453787807)

[Список используемых источников 27](#_Toc453787808)

[Приложение А 28](#_Toc453787809)

# Введение

Изм.

Лист

№ докум.

Подпись

Дата

Лист

КП Т.691015 ПЗ

Разраб.

Кочмарик Н.С.

Провер.

Багласова Е.В.

Т. Контр.

Н. Контр.

Утверд.

Систепа поддержки бронирования и заселения номеров гостиницы

Лит.

Листов

49

КБП

Black sun – небольшой двухэтажный отель. В работе отля важно грамотно использовать доступные номера для заселения и зачастую это не простая задачи. Для упрощения этой задачи было разработано одноименное программное средство. Программное средство способно гибко обрабатывать входной потоко заявок и контролировать заселение и бронирование номеров. В программном средстве также имеется режим редактирования данных чтобы името возсожность подкорректировать данные или заселить человека вручную.

Пояснительная записка к курсовому проекту состоит из шести разделов, содержащих необходимую информацию по организации эксплуатации программного приложения.

В первом разделе «Постановка задачи» раскрывается организационная сущность задачи, описывается предметная область и круг задач, которые должны быть автоматизированы. Описывается задача, перечисляются основные функции программы. Строится информационная модель, отражающая сущности задачи, их свойства и взаимосвязи.

Во втором разделе «Вычислительная система» перечисляются требования к аппаратному обеспечению и конфигурации компьютера, проводится характеристика операционной системы, обоснование выбранной среды для разработки приложения. Описываются новые возможности программы, а также ее отличия от предыдущих версий.

В третьем разделе «Проектирование задачи» проводится объектно-ориентированный анализ задачи, строится концептуальный прототип системы меню, диалоговых окон и элементов управления.

В четвертом разделе «Описание программного средства» представлены общие сведения о программном средстве и его функциональном назначении.

В пятом разделе «Методика испытаний» описываются требования к техническим средствам для проведения испытаний, требования к характеристикам программы применительно к условиям эксплуатации, требования к информационной и программной совместимости. Представляются результаты функционального и полного тестирования.

Шестой раздел «Применение» предназначен для описания сведений о назначении программного средства и области его применения. В этом разделе приводится структура справочной системы, а также методика ее использования.

В заключении будет проанализировано созданное программное приложение, определена степень соответствия поставленной задачи и выполненной работы.

Приложение будет содержать текст программы.

В графической части будут представлены диаграммы вариантов использования, классов, деятельности, последовательности и компонентов.

# 1 Объектно-ориентированный анализ и проектирование системы

# 1.1 Сущность задачи

Задачей данной курсовой работы является создание программного средства, предоставляющего возможность пользователю контролировать зачселение и бронирование номер в отеле, просмотр текущих заказов, клиентов, и статус каждого номера. Рассмотрим предметную область задачи:

* + номер – помещение, в котором проживают клиенты;
  + клиент – человек, который купил время проживания в определенном номере.

Программное средство должно обеспечивать следующие функции:

* + создание и ведение базы данных;
  + обработка базы данных;
  + вывод информации на экран;
  + симулировать работу;
  + формировать отчетную документацию;
  + редактировать данные.

1.2 Проектирование модели

Цель моделирования данных состоит в обеспечении разработчика информационной системы концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных.

Наиболее распространенным средством моделирования данных являются диаграммы «сущность-связь» (ERD). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных. Нотация ERD была впервые введена П. Ченном и получила дальнейшее развитие в работах Баркера.

Исходя из исследования предметной области, можно выделить следующие сущности разработки: заказ, клиент, комната, состояние, тип комнаты.

Для сущности «заказ» можно выделить следующие атрибуты:

* + номер;
  + дата регистрации;
  + дата выезда;
  + полная стоймость;
  + активен.

Для сущности «клиент» можно выделить следующие атрибуты:

* + номер паспорта;
  + Фамиля;
  + Имя;
  + Отчество;
  + Дата рождения;
  + Возраст;
  + Активен.

Для сущности «Комната» можно выделить следующие атрибуты:

* + Номер комнаты;

Для сущности «Состояние» можно выделить следующие атрибуты:

* + название.

Для сущности «Тип комнаты» можно выделить следующие атрибуты:

* + название;
  + цена.

Диаграмма «сущность-связь» представлена на рисунке 1.1.

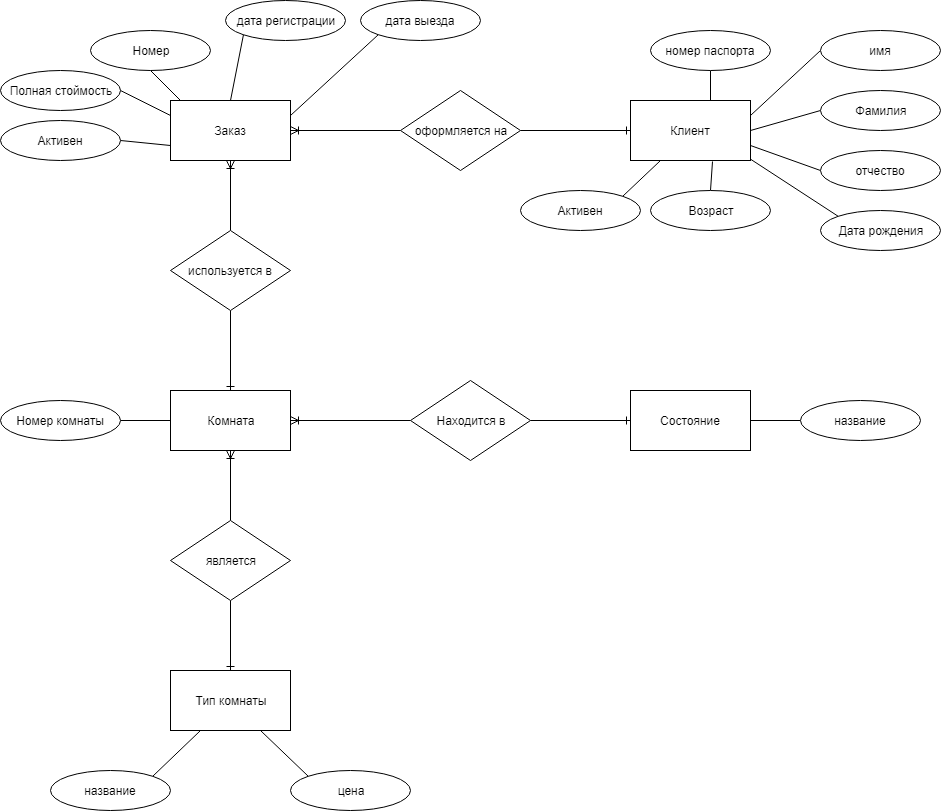


Рисунок 1.1 – Диаграмма «Сущность-связь»

Суть диаграммы вариантов использования состоит в том, что проектируемая система представляется в виде множества сущностей или актёров, взаимодействующих с системой с помощью, так называемых, вариантов использования.

Варианты использования описывают не только взаимодействия между пользователями и сущностью, но также реакции сущности на получение отдельных сообщений от пользователей и восприятие этих сообщений за пределами сущности. Варианты использования могут включать в себя описание особенностей способов реализации сервиса и различных исключительных ситуаций, таких как корректная обработка ошибок системы. Множество вариантов использования в целом должно определять все возможные стороны ожидаемого поведения системы.

Актёр представляет собой внешнюю по отношению к моделируемой системе сущность, которая взаимодействует с системой и использует её функциональные возможности для достижения определённых целей или решения частных задач. При этом актёры служат для обозначения согласованного множества ролей, которые могут играть пользователи в процессе взаимодействия с проектируемой системой. Каждый актёр может рассматриваться как некоторая отдельная роль относительно конкретного варианта использования.

Данное программное средство имеет следующие основные (Include) функции:

* вывод результата на печать;
* ручное редактирование;
* добавление в БД;
* удаление из БД;
* созданеие клиентов;
* создание заказов;
* отображение данных.

К вспомогательным (Extend) функциям, расширяющим возможности системы, относится функция симуляции.

Диаграмма вариантов использования представлена в графической части на листе 1.

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. На данной диаграмме не указывается информация о временных аспектах функционирования системы. С этой точки зрения диаграмма классов является дальнейшим развитием концептуальной модели проектируемой системы.

В данном курсовом проекте будут реализованы классы и их методы. При запуске приложеи ябудет открыта форма авторизации, при вводе названия базы данных будет осуществлен пререход на главную форму. На главной форме отражены 2 кнопки «Simulation» и «Edit Data». Конопка «Simulation» запустит режим симуляции. В режимк симуляции будет открыта форма отображения действия симуляции. Кнопка «Edit Data» откроет форму редактирования данных, на которой можно отредактировать данные в БД.

При моделировании поведения проектируемой или анализируемой системы возникает необходимость детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности. Каждое состояние на диаграмме деятельности соответствует выполнению некоторой элементарной операции, переход в следующее состояние срабатывает только при завершении этой операции. Графически диаграмма деятельности представляется в форме графа, вершинами которого являются состояния действия, а дугами - переходы от одного состояния действия к другому.

Основная цель использования диаграмм деятельности - визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения.

Диаграмма деятельности для функции добавления зказа представлена в графической части на листе 3.

Для моделирования взаимодействия объектов в UML используются соответствующие диаграммы взаимодействия. Если рассматривать взаимодействия объектов во времени, тогда для представления временных особенностей передачи и приема сообщений между объектами используется диаграмма последовательности.

Временной аспект поведения имеет существенное значение при моделировании синхронных процессов, описывающих взаимодействия объектов. Именно для этой цели и используются диаграммы последовательности, в которых ключевым моментом является динамика взаимодействия объектов во времени. При этом диаграмма последовательности имеет как бы два измерения: одно - слева направо в виде вертикальных линий, каждая из которых изображает линию жизни отдельного объекта, участвующего во взаимодействии; второе - вертикальная временная ось, направленная сверху вниз, на которой начальному моменту времени соответствует самая верхняя часть диаграммы.

Диаграмма последовательности для функции добавления клиента представлена в графической части на листе 4.

Рассмотренные ранее диаграммы отражали концептуальные аспекты построения модели системы и относились к логическому уровню представления. Особенность логического представления заключается в том, что оно оперирует понятиями, которые не имеют самостоятельного материального воплощения. Другими словами, различные элементы логического представления, такие как классы, ассоциации, состояния, сообщения, не существуют материально или физически. Они лишь отражают наше понимание структуры физической системы или аспекты ее поведения.

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Однако для создания конкретной физической системы необходимо, некоторым образом, реализовать все элементы логического представления в конкретные материальные сущности. Для описания таких реальных сущностей предназначен другой аспект модельного представления, а именно физическое представление модели.

Диаграмма компонентов описывает объекты реального мира – компоненты программного обеспечения. Эта диаграмма позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами.

Вид диаграммы компонентов для данной проектируемой системы представлен в графической части на листе 5 и содержит следующие компоненты:

* файлы базы данных «Hotel.sql» и «Hotel\_log.mdf»;
* файлы отчетов «\*.docx»;
* файл ресурсов «\*.resx»;
* файлы форм «\*.cs»;
* библиотеки «\*.dll».

# 2 Вычислительная система

# 2.1 Требования к аппаратным и операционным ресурсам

Основными минимальными требованиями, выдвигаемыми к аппаратному обеспечению персонального компьютера, являются:

* процессор 800 МГц и выше;
* оперативная память 128 Мбайт и более;
* свободное место на диске 100 Мбайт;
* интегрированная видеокарта на 512 Мбайт и более;
* монитор;
* мышь, клавиатура;
* принтер.

Компьютер должен работать под управлением операционной системы, начиная с Windows 7 и выше. Наиболее удобной операционной системой для проведения испытаний является Windows 10, так как она ориентированна на максимальное использование всех возможностей ПК, сетевых ресурсов и обеспечение комфортных условий работы.

# 2.2 Инструменты разработки

Инструментами разработки для будущего программного приложения будут являться:

* операционная система Widows 10;
* программная среда разработки Microsoft Visual Studio 2017;
* язык программирования C#;
* система управления базами данных MS SQL Server Management Studio 2018;
* пакет MaterialSkin для создания графической части.

Для разработки данного проекта была использована операционная система Microsoft Windows 10 Максимальная.

Что касается среды разработки, то был выбран так же продукт компании Microsoft. Microsoft Visual Studio — линейка бесплатных интегрированных сред разработки, облегчённая версия Microsoft Visual Studio.

C# — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как основной язык разработки приложений для платформы Microsoft .NET. Компилятор с C# входит в стандартную установку самой .NET, поэтому программы на нём можно создавать и компилировать даже без инструментальных средств, вроде Visual Studio [2].

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, делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

C# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов C#, 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, представляющим собой расширения языка, не опирающиеся на расширения платформы .NET.) CLR предоставляет C#, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования.

Согласно корпорации Майкрософт, SQL Server 2018 является безопасной, масштабируемой и управляемой платформой, обеспечивающей работу ответственных бизнес-приложений. Он включает массу новых возможностей по работе с базами данных, в том числе регулятор ресурсов, сжатие данных, прозрачное шифрование, комплексный аудит и улучшенное зеркалирование баз данных. SQL Server 2018 повышает производительность труда администраторов баз данных, позволяя им управлять многими БД с помощью политик. Он также способствует разработке приложений, управляемых данными, за счет более тесной интеграции с Visual Studio и .NET Framework, а также поддержки всех типов данных.

# 3 Проектирование задачи

# 3.1 Требования к приложению

Разрабатываемое приложение должно иметь понятный и удобный в использовании интерфейс, чтобы взаимодействие между программой и пользователем было максимально упрощено.

Кроме этого, при разработке форм необходимо соблюдать определённые требования: формы в приложении должны иметь стандартные элементы управления данными, не должны быть перегружены информацией, понятны простому пользователю. При конструировании форм в необходимых случаях нужно предусмотреть возможность защиты данных от изменения, установить ограничения на некорректный ввод данных.

* средства защиты;
* накладываемые ограничения;
* требования к интерфейсу программного средства.

# 3.2 Концептуальный прототип

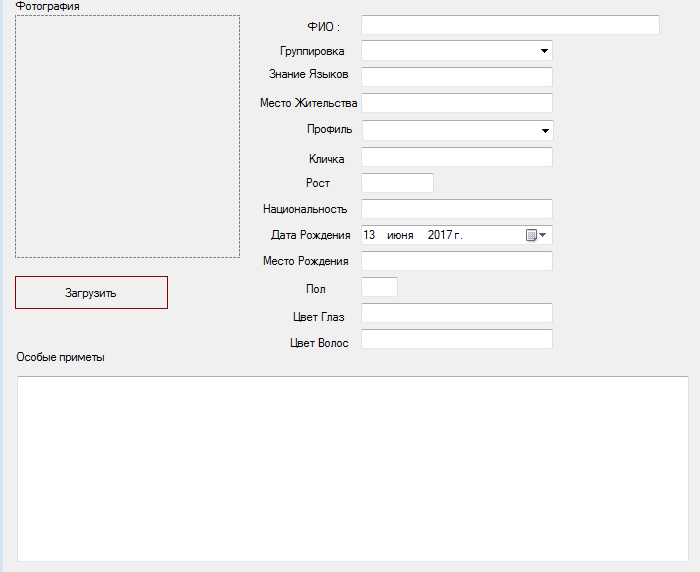
Концептуальный прототип представляет собой описание внешнего пользовательского интерфейса – систему меню и форм.

При запуске приложения будет открыта форма авторизации Рисунок 3.1. После входа открывается главная форма Рисунок 3.2.

Все действия можно будет выполнить с помощью главной формы. Главная форма будет иметь кнопочное меню программы позволяющее вызывать остальные формы программы, осуществлять различные манипуляции с данными, а также получать справочную информацию о работе с приложением. Используя меню, пользователь должен иметь возможность завершить работу всего приложения с сохранением сделанных изменений.

С помощью кнопок на главной форме можно получить доступ к другим формам к редактированию преступников и группировок, просмотру информации и справки.

Для добавления нового преступника будет разработана специальная форма ввода, представленная на рисунке 3.1.



Рисунак 3.1 – Проект формы для добавления нового преступника

Панель инструментов для организации поиска представлена на рисунке 3.2.

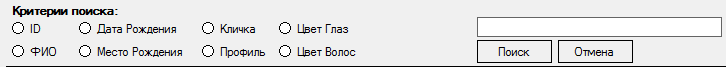


Рисунок 3.2 – Проект панели инструментов

# 3.3 Организация данных

Организация данных подразумевает создание модели данных, главными элементами которой являются сущности и их связи.

Реляционная модель основана на математическом понятии отношения, представлением которого является таблица. В реляционной модели отношения используются для хранения информации об объектах, представленных в базе данных. Отношение имеет вид двухмерной таблицы, в которой строки соответствуют записям, а столбцы - атрибутам. Каждая запись должна однозначно характеризоваться в таблице. Для этого используют первичные и вторичные ключи. Достоинством реляционной модели является простота и удобство физической реализации.

Реляционная модель базы данных подразумевает нормализацию всех таблиц данных. Нормализация - это формальный метод анализа отношений на основе их первичного ключа и функциональных зависимостей, существующих между их атрибутами.

В разделе 1.2 на рисунке 1.1 отображены главные сущности задачи: «Администратор», «Преступник», «Группировка». Каждой сущности ставится в соответствие таблица базы данных.

Структура базы данных разрабатываемого программного средства включает три таблицы. Их краткое описание приводится в таблицах 3.1-3.3.

Таблица «Преступник» хранит информацию о преступниках, ее структура представлена в таблице 3.1.

Таблица 3.1 – Структура таблицы «Преступник»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  поля | Размер поля | Описание поля |
| id\_criminals | Int | 4 | Код преступника |
| full\_name | Varchar | 120 | Фамилия, имя, отчество |
| monkiy | Varchar | 20 | Кличка |
| hiegh | Int | 4 | Рост |
| nationality | Varchar | 20 | Национальность |
| sex | Varchar | 1 | Пол |
| photo | Image |  | Фотография |
| colour\_of\_eyes | Varchar | 15 | Цвет глаз |
| colour\_of\_hairs | Varchar | 15 | Цвет волос |
| id\_profile\_criminals | Int | 4 | Код профиля |
| friends\_id | Int | 4 | Код соучастника |
| Продолжение таблицы 3.1 |  |  |  |
| Имя поля | Тип  Поля | Размер поля | Описание поля |
| special\_details | Varchar | 100 | Особые приметы |
| Language\_spoken | Varchar | 100 | Знание языков |
| date\_of\_birth | Date |  | Дата рождения |
| place\_of\_birth | Varchar | 50 | Место рождения |
| place\_life | Varchar | 50 | Место проживания |
| Archiv | Bit | 1 | Перенос в архив |

Таблица «Группировка» хранит информацию о группировках, она представлена в таблице 3.2

Таблица 3.2 – Структура таблицы «Группировка»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  Поля | Размер поля | Описание поля |
| id\_bands | Int | 4 | Код группировки |
| name | Varchar | 30 | Название |
| number\_of\_members | Int | 4 | Количество участников |
| lider | Int | 4 | Код лидера |

Таблица «Администратор» хранит информацию о пользователях, она представлена в таблице 3.3.

Таблица 3.3 – Структура таблицы «Администратор»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип  Поля | Размер поля | Описание поля |
| id\_user | Int | 4 | Код администратора |
| login | Varchar | 20 | Логин |
| password | Varchar | 20 | Пароль |
| type\_of\_access | bit | 1 | Тип доступа |

Структура базы данных представлена на схеме на рисунке 3.3.

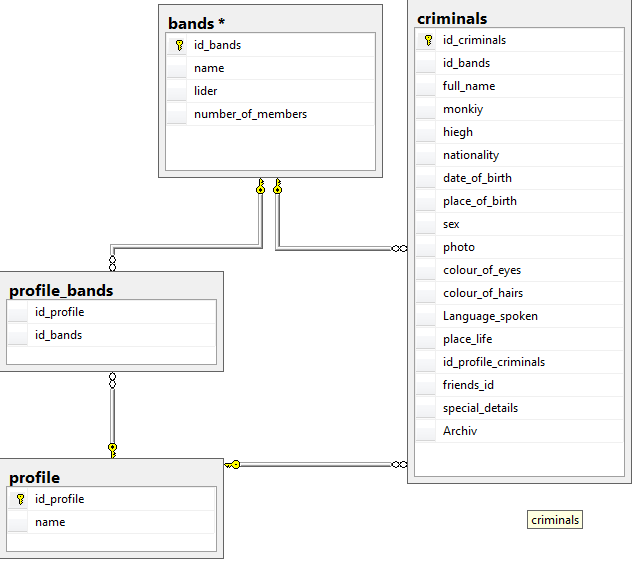


Рисунок 3.3 – Схема данных

# 3.4 Функции: логическая и физическая организация

Рассмотрим основные функции программы.

При нажатии на кнопку «Добавить» и отвечает метод «TB\_ADD\_NEW\_CRIMINALS\_Click», позволяющий добавить нового преступника в базу данных. Программный код реализации данного метода представлен ниже.

private void TB\_ADD\_NEW\_CRIMINALS\_Click(object sender, EventArgs e)

{

try

{

if (PB\_PHOTO.Image == null)

{

if (TB\_FIO.Text != string.Empty)

{

crim.ADD\_NEW\_CRIM(Convert.ToInt32(CB\_Profile.SelectedValue), TB\_FIO.Text, TB\_MONKIE.Text, Convert.ToInt32(TB\_HIEGH.Text), TB\_NATIONALITY.Text, Convert.ToString(dateTimePicker1.Value.ToString()), TB\_PLACE\_OF\_BIRTH.Text, TB\_SEX.Text, TB\_EYES.Text, TB\_HAIR.Text, TB\_LANGUAGE.Text, Convert.ToInt32(CB\_Prof.SelectedValue), TB\_SPECIAL.Text, TB\_PLACE\_LIFE.Text);

MessageBox.Show("Преступник успешно добавлен!", "Успешно");

this.Close();

}

}

else

{

PB\_PHOTO.Image.Save(ms, PB\_PHOTO.Image.RawFormat);

byte[] image = ms.ToArray();

crim.AddCriminals(Convert.ToInt32(CB\_Profile.SelectedValue), TB\_FIO.Text, TB\_MONKIE.Text, Convert.ToInt32(TB\_HIEGH.Text), TB\_NATIONALITY.Text, Convert.ToString(dateTimePicker1.Value.ToString()), TB\_PLACE\_OF\_BIRTH.Text, TB\_SEX.Text, image, TB\_EYES.Text, TB\_HAIR.Text, TB\_LANGUAGE.Text, TB\_PLACE\_LIFE.Text, Convert.ToInt32(CB\_Prof.SelectedValue), TB\_SPECIAL.Text);

MessageBox.Show("Преступник успешно добавлен!", "Успешно");

this.Close();

}

}

catch

{

MessageBox.Show("Ой, что-то пошло не так, проверьте все поля!", "Ошибка");

}

}

При нажатии на кнопку «Удалить» будет вызван метод «BT\_DEL\_Click». Программный код реализации данного метода представлен ниже.

private void BT\_DEL\_Click(object sender, EventArgs e)

{

if (TB\_ID.Text != string.Empty)

{

if (MessageBox.Show("Вы точно желаете удалить этого пользователя?", "Удаление", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

crim.deleteCriminals(Convert.ToInt32(TB\_ID.Text));

MessageBox.Show("Пользователь удачно удален!", "Удаление", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Close();

}

}

}

Весь текст программы представлен в приложении А.

# 3.5 Проектирование справочной системы приложения

Для работы с приложением начинающего пользователя необходимо обеспечить качественной справочной системой, в которой должны быть приведены методы и приемы работы с приложением, включающие данные о том, что произойдет после нажатия на определенную кнопку или при выборе пункта меню; сведения о том, какую информацию и в каком виде следует вводить в соответствующие поля, каким образом можно вносить изменения в базу данных.

Пользователь должен иметь доступ к справочной информации на протяжении всей работы с программой для этого пользователь может воспользоваться выбором в главном меню подпункта «Справка», после чего пользователю будет предоставлена справочная система, состоящая из нескольких разделов, и предоставляющая полную информацию о функциях приложения, а также на главной форме приложения.

Справочная система необходима для ознакомления с программой. В ней должна присутствовать информация, которая может пригодиться пользователю: о правилах пользования приложением, о его возможностях и о создателе программы.

Система справки данного программного средства будет содержать следующие разделы:

* «Главная форма»;
* «Преступники»;
* «Пользователи»;
* «Группировки»;
* «Архив».

Справочная система будет создана в программе Help & Manual Pro v6.2.3.

# 4 Описание программного средства

# 4.1 Общее сведения

Из всех задач, которые будет решать разрабатываемое программное средство, можно выделить ряд основных:

* предоставление личных дел преступников;
* добавление, удаление, редактирование информации о преступнике;
* вывод информации в файлы Microsoft Office;
* осуществление поиска по выбранному критерию.

Разрабатываемое программное средство предназначено для отображения информации по зарегистрированным преступникам. Приложение «Картотека Интерпола» позволит пользователю получить сведения по каждому преступнику.

Для применения данного программного средства необходимы следующие технические требования:

* процессор Pentium 800 или выше;
* минимальный объем оперативной памяти — 512 Мбайт;
* операционная система Windows7 и выше;
* рекомендуется монитор типа VGA или с лучшей разрешающей способностью;
* клавиатура;
* мышь;
* среда SQL Server Management Studio.

Программное средство создано в среде разработки Visual Studio 2012 на языке программирования C# в операционной системе Windows. Программное средство может работать в средах операционных систем семейства Microsoft Windows начиная с Windows 7. Программа не требовательна к системным ресурсам, также проста в использовании и не требует специальных навыков при работе.

Инсталляция программного средства не требуется, достаточно только скопировать готовый проект на ПК и запустить.

# 4.2 Функциональное назначение

Предметной областью решаемой задачи является предоставление сведений о преступниках, которые хранятся в базе данных.

Основными задачами приложения является обеспечение пользователя полной информацией о преступниках.

Программа использует стандартные элементы управления, такие как кнопки, меню, списка, поля ввода, что обеспечивает единство интерфейса системы и программного средства.

Работа с приложением осуществляется на локальном компьютере, сетевая поддержка присутствует.

Таким образом, программа может применяться в реальных условиях, представляя собой достаточно удобную картотеку преступников.

# 4.3 Входные и выходные данные

Входными данными при добавлении преступника в базу данных являются поля всех таблиц, а именно:

* Ф.И.О.;
* кличка;
* рост;
* цвет волос и глаз;
* особые приметы;
* гражданство;
* место и дата рождения;
* место жительства;
* знание языков;
* преступная профессия;
* группировка;
* фотография.

Выходные данные – выходными данными является сформированный отчет о выбранном преступнике.

# 5 Методика испытаний

# 5.1 Технические требования

Минимальными требованиями для оптимальной работы программного средства является персональный компьютер (ПК) со следующими характеристиками:

* процессор 800 МГц и выше;
* оперативная память 128 Мбайт и более;
* свободное место на диске 100 Мбайт;
* интегрированная видеокарта на 512 Мбайт и более;
* среда SQL Server Management Studio.

Компьютер должен работать под управлением операционной системы, начиная с Windows7 и выше. Наиболее удобной операционной системой для проведения испытаний является Windows 7, так как она ориентированна на максимальное использование всех возможностей ПК, сетевых ресурсов и обеспечение комфортных условий работы.

# 5.2 Порядок проведения испытаний

# 5.2.1 Функциональное тестирование

В процессе написания программного средства необходимо производить тестирование на правильность работы приложения. Одной из основных задач тестирования является устранение ошибок, происходящих при вводе данных.

Функциональное тестирование – это тестирование функций приложения на соответствие требованиям. Оценка производится в соответствии с ожидаемыми и полученными результатами (на основании функциональной спецификации), при условии, что функции отрабатывали на различных значениях.

Тестирование программы будет производиться последовательно, переходя из одной части программы в другую. Во время теста будут проверяться все действия с программой, навигация пунктам меню, которые может произвести пользователь. После чего, все собранные и найденные ошибки будут исправлены.

При запуске программы появляется окно для проверки подключения к SQL серверу, вид которого приведен на рисунке 5.1, из этого окна при нажатии на кнопки «Проверить подключение» происходит переход на форму авторизации.



Рисунок 5.1 – Окно проверки подключения

В программном средстве предусмотрен доступ, к базе данных с помощью логина и пароля пользователя, для защиты информации предусмотрено разграничение прав пользователей. На рисунке 5.2 представлена форма авторизации.

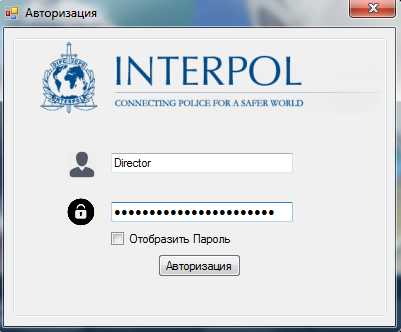


Рисунок 5.2 – Окно авторизации

При правильном вводе логина и пароля, отображается окно главной формы. На рисунке 5.3 предоставлена главная форма.

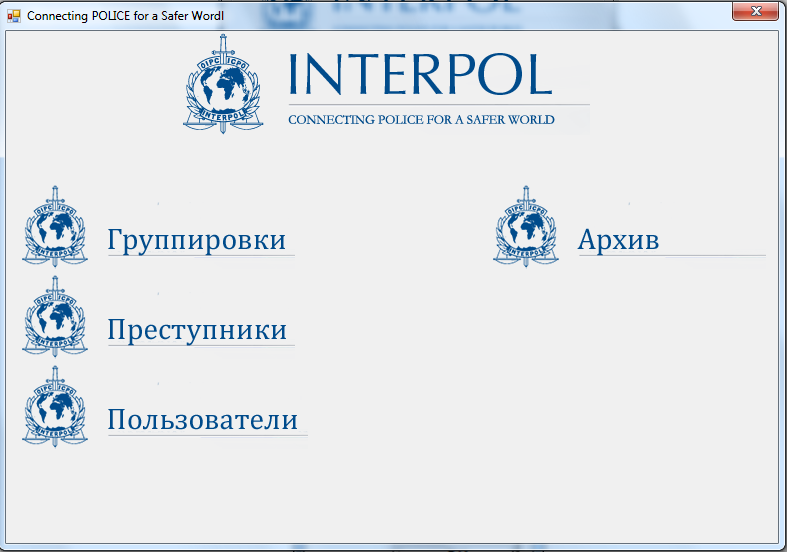


Рисунок 5.3 – Главная форма

При неправильном вводе логина и пароля, отображается сообщение о неверном пароле. На рисунке 5.4 предоставлено окно с сообщением об ошибке.

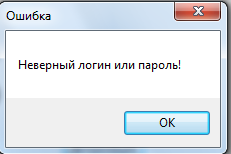


Рисунок 5.4 – Окно с сообщением о неверном пароле

При нажатии на кнопку «Преступники», отображается окно с информацией о преступниках, хранящихся в базе данных. Результат представлен на рисунке 5.5.



Рисунок 5.5 – Окно редактирование базы данных

При нажатии на кнопку «Добавить», на форме «Преступники» отображается форма с возможностью добавления нового преступника в базу данных. На рисунке 5.6 предоставлен результат работы программы.

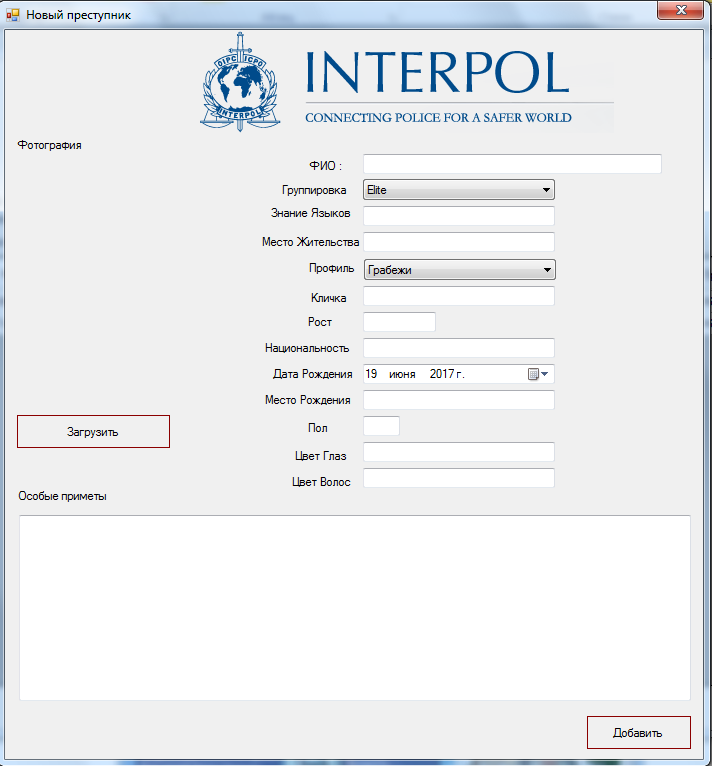


Рисунок 5.6 – Окно добавления нового преступника

На форме «Преступники» при выборе любого преступника отобразится окно с информацией о нем. Результат представлен на рисунке 5.7.

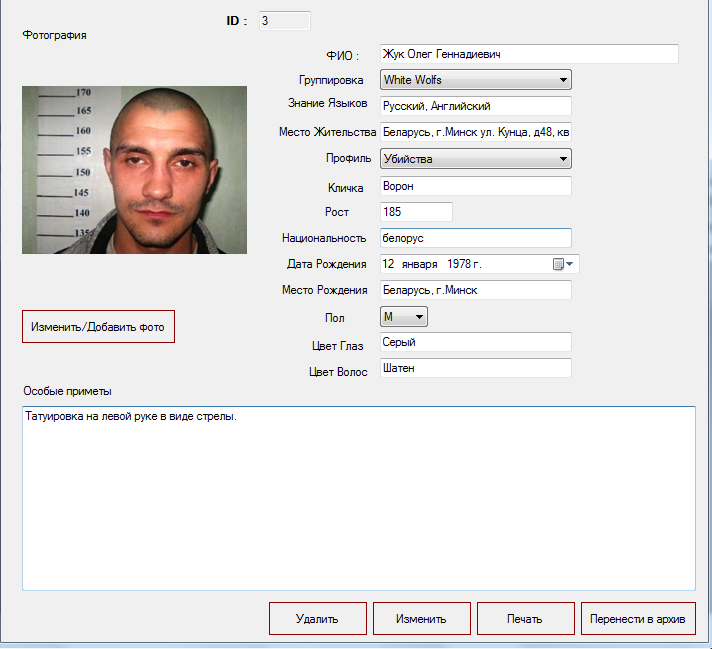


Рисунок 5.7 – Окно с информацией о преступнике

# 5.2.2 Полное тестирование

Полное тестирование предполагает моделирование всех действий администратора при работе с программой, так же будет представлен тестовый пример с конкретными входными данными и результатами.

При нажатии на кнопку «Архив» на «Главная форма» отобразится окно со сведениями о завязавших преступниках. Результат предоставлен на рисунке 5.8.

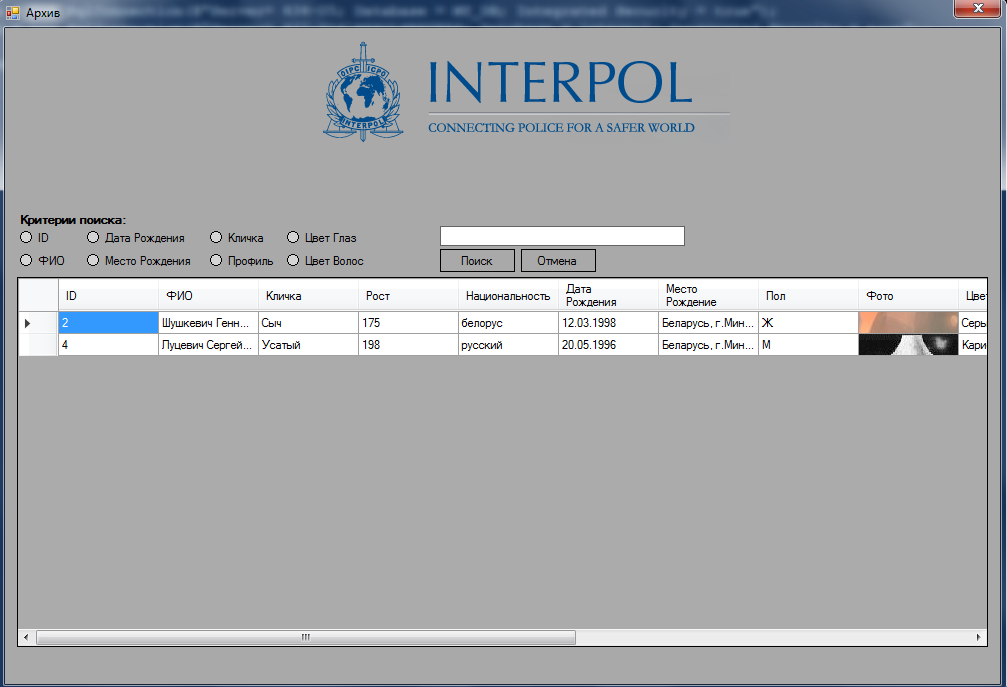


Рисунок 5.8 – Окно формы архив

При выборе завязавшего преступника появляется возможность вывести информацию на печать. Результат предоставлен на рисунке 5.9.

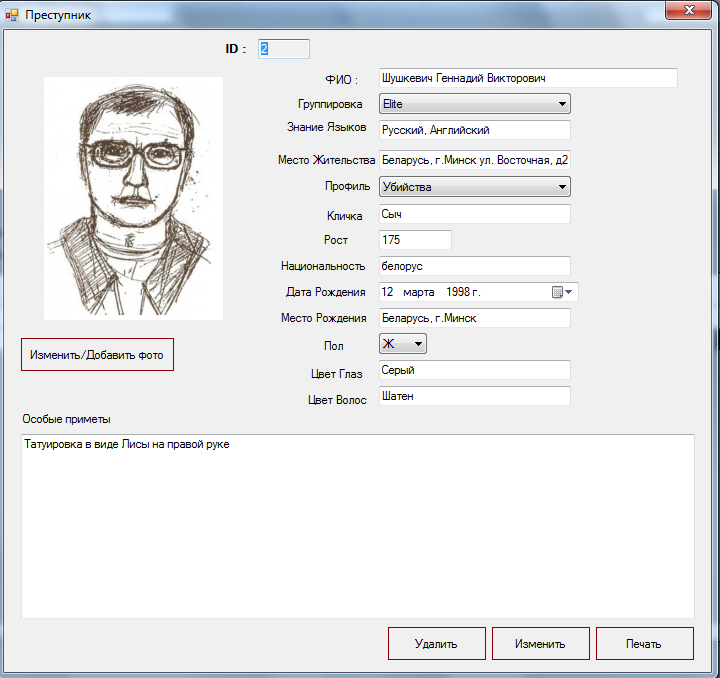


Рисунок 5.9 – Окно с информацией о преступнике

Форма выходной отчетности представлена на рисунке 6.1

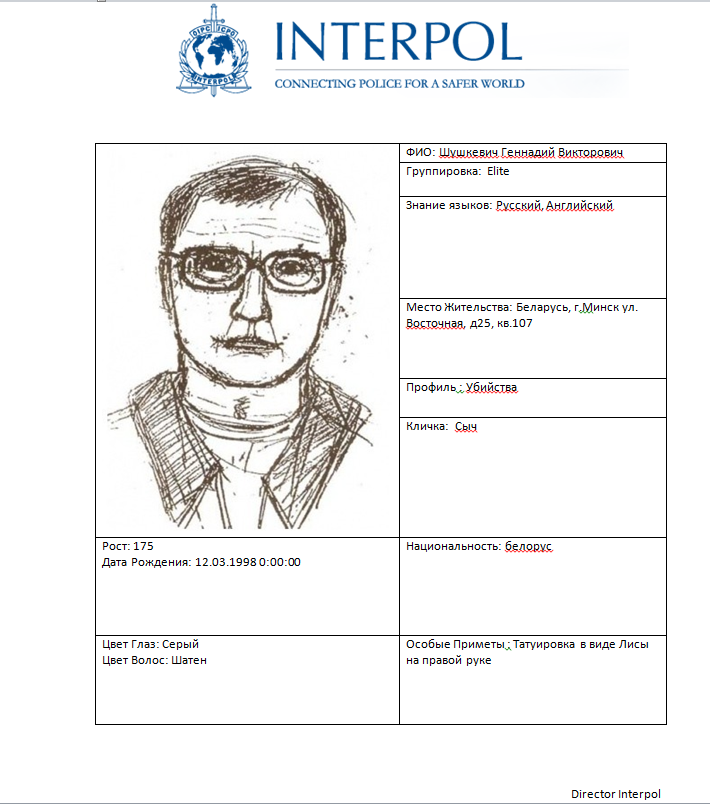


Рисунок 6.1 – Форма выходной отчетности

# 6 Применение

# 6.1 Назначение программы

Главным назначением программы является получения сведений о преступниках, хранящихся в базе данных, в зависимости от выбранных пользователем критериев.

Разработанное приложение имеет понятный интерфейс, довольно просто в применении, но в тоже время позволяет уменьшить затраты времени, используемые при вычислении за счет автоматизации вычислительных процессов

# 6.2 Условия применения

Для работы с приложением требуется установка дополнительных компонентов и специальных технических и программных ресурсов.

Для применения данного программного средства необходимы следующие технически требования:

* процессор Intel Core 2 Duo или выше;
* минимальный объем оперативной памяти — 512 Мбайт;
* операционная система Windows 7 и выше;
* Framework v4.5;
* рекомендуется монитор типа VGA или с лучшей разрешающей способностью;
* устройство для чтения дисков;
* клавиатура;
* мышь;
* среда SQL Server Management Studio.

# 6.3 Справочная система

Справочная система необходима для ознакомления с программным средством. В ней будет изложена информация, которая может пригодиться пользователю. Содержание справочной системы должно быть просто и ясно изложено.

Справочная система разработана с помощью программы Help & Manual Pro v6.2.3, предназначенной для компиляции html-документов в единый .chm файл.

Справочная система программного средства представляет собой отдельный файл «help.chm» с полным описанием основных функций программы в формате \*.chm. В справочной системе даны ответы на типичные вопросы, возникающие при работе с приложением, что, несомненно, должно помочь при освоении программного средства.

Справочная система данного программного средства содержит следующие разделы:

* «Главная форма»;
* «Преступники»;
* «Пользователи»;
* «Группировки»;
* «Архив».

Справочная система по работе с программным средством будет представлена при нажатии кнопки «Справка» в главном окне.

Структура справочной системы представлена на рисунке 6.2.

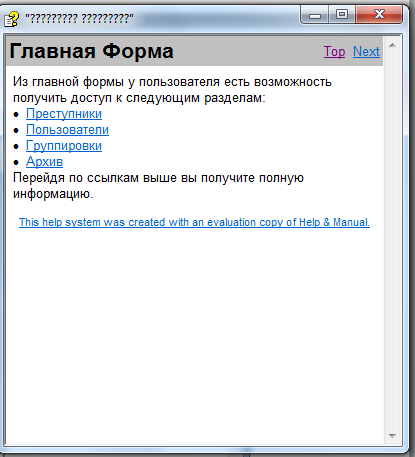


Рисунок 6.2 – Справочная система

# Заключение

В рамках курсового проектирования по предмету «Конструирование программ и языки программирования» было разработано программное средство, предоставляющее возможность редактирования, добавления и удаления информации, просмотр и печать информации.

Во время курсового проектирования удалось накопить неоценимый опыт в разработке приложений в интегрированной среде визуального программирования Microsoft Visual Studio 2012, развить умения самостоятельно находить оптимальный метод решения задачи. Кроме того, изучить принципы организации и технологии реализации программного обеспечения, развить умения разработки программ в соответствии с требованиями технического задания, обеспечивая высокий уровень качества программного обеспечения и экономической эффективности. Получить навыки оформления комплекта документации на созданное программное обеспечение.

Целями курсового проектирования по предмету «Конструирование программ и языки программирования» являлось:

- закрепление знаний, связанных с технологией обработки информации;

- приобретение навыков разработки программ с использованием современных средств обработки экономической и деловой информации;

- выработка умений применять средства стандартных библиотек в практических задачах.

Задачами курсового проектирования являлось:

- углубленное изучение принципов организации программного обеспечения и технологии его проектирования;

- владение методами надежного программирования;

- умение разрабатывать программы в соответствии с промышленными требованиями, обеспечивая высокий уровень качества программного обеспечения и экономической эффективности;

- изучение способов определения экономической себестоимости и эффективности разработки программного обеспечения;

- оформление комплекта документации на созданное программное обеспечение.

При создании программных приложений, входящих к курсовому проектированию, были отработаны навыки владения методами надежного программирования и изучены способы определения экономической себестоимости и эффективности разработки программного обеспечения.

# Список информационных источников

1. Багласова, Т.Г. Методические указания по выполнению дипломного проекта для учащихся по специальности 2-40 01 01 «Программное обеспечение технологий» / Т.Г. Багласова. – Минск : КБП, 2013. – 24 c.
2. Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск : КБП, 2013. – 29 c.
3. Вендров, А.М. Проектирование программного обеспечения экономических информационных систем: Учебник / А.М. Вендров. – 2-е изд., перераб. и доп.– М. : Финансы и статистика, 2006. — 544 с.
4. Мацяшек, Лешек А. Анализ и проектирование информационных систем с помощью UML 2.0 / Лешек А. Мацяшек. – 3-е изд. – М. : ООО «И.Д. Вильямс», 2008. — 816 с.

# Приложение А

(обязательное)

Текст программы

// главная форма

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace INTERPOL

{

public partial class Main\_Form : Form

{

public Main\_Form()

{

InitializeComponent();

}

private void PB\_USER\_Click(object sender, EventArgs e)

{

this.Hide();

USER\_FORM FH = new USER\_FORM();

FH.Closed += (s, args) => this.Show();

FH.Show();

}

private void PB\_GROUP\_Click(object sender, EventArgs e)

{

this.Hide();

BAND\_FORM FH = new BAND\_FORM();

FH.Closed += (s, args) => this.Show();

FH.Show();

}

private void PB\_PRISON\_Click(object sender, EventArgs e)

{

this.Hide();

PRISON\_FORM FH = new PRISON\_FORM();

FH.Closed += (s, args) => this.Show();

FH.Show();

}

private void PB\_Archiv\_Click(object sender, EventArgs e)

{

this.Hide();

Archiv\_Form FH = new Archiv\_Form();

FH.Closed += (s, args) => this.Show();

FH.Show();

}

}

}// подключение справки

private void button8\_Click(object sender, RoutedEventArgs e)

{

Process SysInfo = new Process();

SysInfo.StartInfo.ErrorDialog = true;

SysInfo.StartInfo.FileName = "Справка.chm";

SysInfo.Start();

}

}

}

// класс Преступник

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

namespace INTERPOL

{

class Criminal

{

public DataTable getCriminal()

{

ConnectBD db = new ConnectBD();

DataTable tab = new DataTable();

tab = db.getData("spr\_getCriminals", null);

db.closeConnection();

return tab;

}

public void deleteCriminals(int userId)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[1];

parameters[0] = new SqlParameter("@uid", SqlDbType.Int);

parameters[0].Value = userId;

db.setData("spr\_delete\_criminal", parameters);

db.closeConnection();

}

public void updateCriminals(int uid,int band, string name, string monkie, int hiegh, string nationality, string place, string sex, string colourE, string colourH, string lang, string life, int prof, string special,DateTime Date)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[15];

parameters[0] = new SqlParameter("@full\_name", SqlDbType.VarChar, 120);

parameters[0].Value = name;

parameters[1] = new SqlParameter("@monkie", SqlDbType.VarChar, 20);

parameters[1].Value = monkie;

parameters[2] = new SqlParameter("@hiegh", SqlDbType.Int);

parameters[2].Value = hiegh;

parameters[3] = new SqlParameter("@nationality", SqlDbType.VarChar, 20);

parameters[3].Value = nationality;

parameters[4] = new SqlParameter("@place", SqlDbType.VarChar, 50);

parameters[4].Value = place;

parameters[5] = new SqlParameter("@sex", SqlDbType.VarChar, 1);

parameters[5].Value = sex;

parameters[6] = new SqlParameter("@coloureyes", SqlDbType.VarChar, 15);

parameters[6].Value = colourE;

parameters[7] = new SqlParameter("@colourhair", SqlDbType.VarChar, 15);

parameters[7].Value = colourH;

parameters[8] = new SqlParameter("@language", SqlDbType.VarChar, 100);

parameters[8].Value = lang;

parameters[9] = new SqlParameter("@place\_life", SqlDbType.VarChar, 50);

parameters[9].Value = life;

parameters[10] = new SqlParameter("@prolife", SqlDbType.Int);

parameters[10].Value = prof;

parameters[11] = new SqlParameter("@special", SqlDbType.VarChar, 100);

parameters[11].Value = special;

parameters[12] = new SqlParameter("@id\_band", SqlDbType.Int);

parameters[12].Value = band;

parameters[13] = new SqlParameter("@id",SqlDbType.Int);

parameters[13].Value = uid;

parameters[14] = new SqlParameter("@date",SqlDbType.Date);

parameters[14].Value = Date;

db.setData("spr\_update\_criminals", parameters);

db.closeConnection();

}

public void AddCriminals(int band, string name, string monkie, int hiegh, string nationality, string date, string place, string sex, byte[] image, string colourE, string colourH, string lang, string life, int prof, string special)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[15];

parameters[0] = new SqlParameter("@full\_name", SqlDbType.VarChar, 120);

parameters[0].Value = name;

parameters[1] = new SqlParameter("@monkie", SqlDbType.VarChar, 20);

parameters[1].Value = monkie;

parameters[2] = new SqlParameter("@hiegh", SqlDbType.Int);

parameters[2].Value = hiegh;

parameters[3] = new SqlParameter("@nationality", SqlDbType.VarChar, 20);

parameters[3].Value = nationality;

parameters[4] = new SqlParameter("@date", SqlDbType.Date);

parameters[4].Value = date;

parameters[5] = new SqlParameter("@place", SqlDbType.VarChar, 50);

parameters[5].Value = place;

parameters[6] = new SqlParameter("@sex", SqlDbType.VarChar, 1);

parameters[6].Value = sex;

parameters[7] = new SqlParameter("@img", SqlDbType.Image);

parameters[7].Value = image;

parameters[8] = new SqlParameter("@coloureyes", SqlDbType.VarChar, 15);

parameters[8].Value = colourE;

parameters[9] = new SqlParameter("@colourhair", SqlDbType.VarChar, 15);

parameters[9].Value = colourH;

parameters[10] = new SqlParameter("@language", SqlDbType.VarChar, 100);

parameters[10].Value = lang;

parameters[11] = new SqlParameter("@place\_life", SqlDbType.VarChar, 50);

parameters[11].Value = life;

parameters[12] = new SqlParameter("@prolife", SqlDbType.Int);

parameters[12].Value = prof;

parameters[13] = new SqlParameter("@special", SqlDbType.VarChar, 100);

parameters[13].Value = special;

parameters[14] = new SqlParameter("@id\_band", SqlDbType.Int);

parameters[14].Value = band;

db.setData("spr\_add\_criminals", parameters);

db.closeConnection();

}

public void updateCriminalsPhoto(int uid, byte[] img)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[2];

parameters[0] = new SqlParameter("@id", SqlDbType.Int);

parameters[0].Value = uid;

parameters[1] = new SqlParameter("@imag", SqlDbType.Image);

parameters[1].Value = img;

db.setData("spr\_update\_criminals\_Photo", parameters);

db.closeConnection();

}

public DataTable CriminalsInBands(int ID)

{

ConnectBD db = new ConnectBD();

DataTable table = new DataTable();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[1];

parameters[0] = new SqlParameter("@cid", SqlDbType.Int);

parameters[0].Value = ID;

table = db.getData("spr\_crim\_in\_bands", parameters);

db.closeConnection();

return table;

}

public void ADD\_NEW\_CRIM(int bands, string name, string monkey, int heigh, string nation,string date, string place,

string sex, string colorE,string colorH, string language, int profile, string special , string life)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[14];

parameters[0] = new SqlParameter("@full\_name", SqlDbType.VarChar, 120);

parameters[0].Value = name;

parameters[1] = new SqlParameter("@monkie", SqlDbType.VarChar,20);

parameters[1].Value = monkey;

parameters[2] = new SqlParameter("@hiegh", SqlDbType.Int);

parameters[2].Value = heigh;

parameters[3] = new SqlParameter("@nationality", SqlDbType.VarChar,20);

parameters[3].Value = nation;

parameters[4] = new SqlParameter("@date", SqlDbType.Date);

parameters[4].Value = date;

parameters[5] = new SqlParameter("@place", SqlDbType.VarChar,50);

parameters[5].Value = place;

parameters[6] = new SqlParameter("@sex", SqlDbType.VarChar,1);

parameters[6].Value = sex;

parameters[7] = new SqlParameter("@coloureyes",SqlDbType.VarChar,15);

parameters[7].Value = colorE;

parameters[8] = new SqlParameter("@colourhair",SqlDbType.VarChar,15);

parameters[8].Value = colorH;

parameters[9] = new SqlParameter("@language",SqlDbType.VarChar,100);

parameters[9].Value = language;

parameters[10] = new SqlParameter("@place\_life",SqlDbType.VarChar,50);

parameters[10].Value = life;

parameters[11] = new SqlParameter("@prolife",SqlDbType.Int);

parameters[11].Value = profile;

parameters[12] = new SqlParameter("@special",SqlDbType.VarChar,100);

parameters[12].Value = special;

parameters[13] = new SqlParameter("@bands", SqlDbType.Int);

parameters[13].Value = bands;

db.setData("spr\_add\_new\_crim\_",parameters);

db.closeConnection();

}

}

}

// класс Подключение к базе данных

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

using System.Xml.Serialization;

using System.Data.SqlClient;

using System.Data;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace INTERPOL

{

class ConnectBD

{

SqlConnection connection;

public ConnectBD()

{

connection = new SqlConnection(@"Server= " + Server.server + " ; Database = interpol; Integrated Security = true");

}

public void openConnection()

{

if (connection.State != ConnectionState.Open)

{

connection.Open();

}

}

public void closeConnection()

{

if (connection.State != ConnectionState.Closed)

{

connection.Close();

}

}

public DataTable getData(string procedureName, SqlParameter[] procedureParams)

{

try

{

SqlCommand command = new SqlCommand();

command.CommandType = CommandType.StoredProcedure;

command.CommandText = procedureName;

command.Connection = connection;

if (procedureParams != null)

{

for (int i = 0; i < procedureParams.Length; i++)

{

command.Parameters.Add(procedureParams[i]);

}

}

SqlDataAdapter adapter = new SqlDataAdapter(command);

DataTable table = new DataTable();

adapter.Fill(table);

return table;

}

catch

{

MessageBox.Show("Ошибка получения информации","Ошибка");

return null;

}

}

public void setData(string procedureName, SqlParameter[] procedureParams)

{

SqlCommand command = new SqlCommand();

command.CommandType = CommandType.StoredProcedure;

command.CommandText = procedureName;

command.Connection = connection;

if (procedureParams != null)

{

command.Parameters.AddRange(procedureParams);

}

command.ExecuteNonQuery();

}

}

}

// класс Форма с преступниками

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using Word = Microsoft.Office.Interop.Word;

namespace INTERPOL

{

public partial class VIEW\_CRIMINAL\_FORM : Form

{

Criminal crim = new Criminal();

private readonly string TemplateWord = @"F:\Курсовой Проект C#\INTERPOL\INTERPOL\bin\Debug\Template.docx";

public VIEW\_CRIMINAL\_FORM()

{

InitializeComponent();

CB\_Prof.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

CB\_Profile.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

CB\_SEX.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

}

private void BT\_DEL\_Click(object sender, EventArgs e)

{

if (TB\_ID.Text != string.Empty)

{

if (MessageBox.Show("Вы точно желаете удалить этого пользователя?", "Удаление", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

crim.deleteCriminals(Convert.ToInt32(TB\_ID.Text));

MessageBox.Show("Пользователь удачно удален!", "Удаление", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Close();

}

}

}

private void BT\_EDDIT\_Click(object sender, EventArgs e)

{

if (TB\_FIO.Text == string.Empty)

{

MessageBox.Show("Введите ФИО!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

if (TB\_FIO.Text != string.Empty && TB\_EYES.Text != string.Empty && TB\_HAIR.Text != string.Empty && TB\_HIEGH.Text != string.Empty && TB\_LANGUAGE.Text != string.Empty &&

TB\_MONKIE.Text != string.Empty && TB\_NATIONALITY.Text != string.Empty && TB\_PLACE\_LIFE.Text != string.Empty && TB\_PLACE\_OF\_BIRTH.Text != string.Empty &&

DT\_DATE.Value != null)

{

if (MessageBox.Show("Вы точно желаете внести изменения в базу данных?", "Изменить преступника", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

crim.updateCriminals(Convert.ToInt32(TB\_ID.Text), Convert.ToInt32(CB\_Profile.SelectedValue), TB\_FIO.Text, TB\_MONKIE.Text, Convert.ToInt32(TB\_HIEGH.Text), TB\_NATIONALITY.Text, TB\_PLACE\_OF\_BIRTH.Text, CB\_SEX.Text, TB\_EYES.Text, TB\_HAIR.Text, TB\_LANGUAGE.Text, TB\_PLACE\_LIFE.Text, Convert.ToInt32(CB\_Prof.SelectedValue), LB\_SPECIAL.Text, DT\_DATE.Value);

MessageBox.Show("Данные успешно изменены!", "Изменить", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Close();

}

}

}

private void BT\_PHOTO\_Click(object sender, EventArgs e)

{

OpenFileDialog ImagefileDialog = new OpenFileDialog();

ImagefileDialog.Filter = "Image |\*.JPG; \*.PNG; \*.GIF";

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

{

PB\_PHOTO.Image = Image.FromFile(ImagefileDialog.FileName);

MemoryStream ms = new MemoryStream();

PB\_PHOTO.Image.Save(ms, PB\_PHOTO.Image.RawFormat);

byte[] image = ms.ToArray();

crim.updateCriminalsPhoto(Convert.ToInt32(TB\_ID.Text), image);

MessageBox.Show("Фотография загруженна!", "Новая фотография", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Close();

}

}

private void VIEW\_CRIMINAL\_FORM\_Load(object sender, EventArgs e)

{

PRISON\_FORM main = this.Owner as PRISON\_FORM;

if (main != null)

{

TB\_ID.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[0].Value.ToString();

TB\_FIO.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[1].Value.ToString();

TB\_MONKIE.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[2].Value.ToString();

TB\_HIEGH.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[3].Value.ToString();

TB\_NATIONALITY.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[4].Value.ToString();

DT\_DATE.Value = (DateTime)main.DGV\_CRIMINAL.CurrentRow.Cells[5].Value;

TB\_PLACE\_OF\_BIRTH.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[6].Value.ToString();

TB\_EYES.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[9].Value.ToString();

TB\_HAIR.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[10].Value.ToString();

TB\_LANGUAGE.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[11].Value.ToString();

TB\_PLACE\_LIFE.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[12].Value.ToString();

LB\_SPECIAL.Text= main.DGV\_CRIMINAL.CurrentRow.Cells[14].Value.ToString();

CB\_SEX.Text = main.DGV\_CRIMINAL.CurrentRow.Cells[7].Value.ToString();

try

{

//string ss =

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand ss = con.CreateCommand();

ss.CommandText = @"select id\_bands, name from bands";

SqlDataAdapter com = new SqlDataAdapter();

DataSet comm = new DataSet();

com.SelectCommand = ss;

com.Fill(comm, "bands");

CB\_Profile.DataSource = comm.Tables["bands"];

CB\_Profile.DisplayMember = "name";

CB\_Profile.ValueMember = "id\_bands";

}

catch

{

MessageBox.Show("Ошибка загрузки банд из Базы Данных!\nПроверьте БД");

}

try

{

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand ss = con.CreateCommand();

ss.CommandText = @"select id\_profile, name from profile";

SqlDataAdapter com = new SqlDataAdapter();

DataSet comm = new DataSet();

com.SelectCommand = ss;

com.Fill(comm, "profile");

CB\_Prof.DataSource = comm.Tables["profile"];

CB\_Prof.DisplayMember = "name";

CB\_Prof.ValueMember = "id\_profile";

}

catch

{

MessageBox.Show("Ошибка загрузки профиля преступника из Базы Данных!\nПроверьте подключение к БД");

}

try

{

string s = @"select bands.name from criminals,bands where bands.id\_bands = criminals.id\_bands and id\_criminals=" + Convert.ToInt32(TB\_ID.Text);

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

CB\_Profile.Text = (string)comm.ExecuteScalar();

s = @"select profile.name from criminals, profile where profile.id\_profile = criminals.id\_profile\_criminals and criminals.id\_criminals =" + Convert.ToInt32(TB\_ID.Text);

comm = new SqlCommand(s, con);

CB\_Prof.Text = (string)comm.ExecuteScalar();

con.Close();

if (main.DGV\_CRIMINAL.CurrentRow.Cells[8].Value.ToString() != null)

{

string Query = @"select photo from criminals where id\_criminals=" + Convert.ToInt32(TB\_ID.Text);

con.Open();

comm = new SqlCommand(Query, con);

byte[] byteArray = new byte[0];

byteArray = (byte[])comm.ExecuteScalar();

MemoryStream ms = new MemoryStream(byteArray);

PB\_PHOTO.Image = Image.FromStream(ms);

PB\_PHOTO.SizeMode = PictureBoxSizeMode.Zoom;

con.Close();

ms.Close();

}

}

catch

{

MessageBox.Show("Извините, но фотографии преступника в базе нет!","Ошибка");

}

}// if :1

} // Close method

private void BT\_PRINT\_Click(object sender, EventArgs e)

{

SaveFileDialog savedialog = new SaveFileDialog();

savedialog.Title = "Сохранить Отчет по преступнику как...";

savedialog.OverwritePrompt = true;

savedialog.CheckPathExists = true;

savedialog.Filter = "Word Files(\*.Docx)|";

savedialog.ShowHelp = true;

if (savedialog.ShowDialog() == DialogResult.OK) //если в диалоговом окне нажата кнопка "ОК"

{

try

{

var WordReport = new Word.Application();

WordReport.Visible = false;

var FIO = TB\_FIO.Text;

var WordDocument = WordReport.Documents.Open(TemplateWord); // Object usefull

ReplaceWordStub("{F}", FIO, WordDocument);

ReplaceWordStub("{group}", CB\_Profile.Text, WordDocument);

ReplaceWordStub("{place}", TB\_PLACE\_LIFE.Text, WordDocument);

ReplaceWordStub("{Lang}", TB\_LANGUAGE.Text, WordDocument);

ReplaceWordStub("{prof}", CB\_Prof.Text, WordDocument);

ReplaceWordStub("{monkie}", TB\_MONKIE.Text, WordDocument);

ReplaceWordStub("{hieg}", TB\_HIEGH.Text, WordDocument);

ReplaceWordStub("{national}", TB\_NATIONALITY.Text, WordDocument);

ReplaceWordStub("{date}", DT\_DATE.Value.ToString(), WordDocument);

ReplaceWordStub("{eyes}", TB\_EYES.Text, WordDocument);

ReplaceWordStub("{hair}", TB\_HAIR.Text, WordDocument);

ReplaceWordStub("{special}", LB\_SPECIAL.Text, WordDocument);

if (PB\_PHOTO.Image != null)

{

Clipboard.SetImage(PB\_PHOTO.Image);

WordDocument.Bookmarks.get\_Item("photo").Range.Paste();

}

WordDocument.SaveAs(savedialog.FileName);

WordReport.Visible = true;

}

catch

{

MessageBox.Show("Невозможно сохранить отчет", "Ошибка",

MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void ReplaceWordStub(string StubtoReplace, string text, Word.Document WordDocument)

{

var Range = WordDocument.Content;

Range.Find.ClearFormatting();

Range.Find.Execute(FindText: StubtoReplace, ReplaceWith: text);

}

private void BT\_ARCHIV\_Click(object sender, EventArgs e)

{

try

{

Archiv archiv = new Archiv();

archiv.AddInArchiv(TB\_ID.Text);

MessageBox.Show("Преступник перенесен в архив!","Успешно!");

this.Close();

}

catch

{

MessageBox.Show("Упс... Что-то пошло не так, проверьте все поля!","Ошибка");

}

}

private void TB\_HIEGH\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar != '\b')

{

e.Handled = !char.IsNumber(e.KeyChar);

}

}

}

}

// класс Форма с преступниками

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace INTERPOL

{

public partial class PRISON\_FORM : Form

{

Criminal crim = new Criminal();

public PRISON\_FORM()

{

InitializeComponent();

DGV\_CRIMINAL.DataSource = crim.getCriminal();

}

private void BT\_EDIT\_Click(object sender, EventArgs e)

{

}

private void BT\_ADD\_CRIMINAL\_Click(object sender, EventArgs e)

{

ADD\_NEW\_CRIMINALS FH = new ADD\_NEW\_CRIMINALS();

FH.ShowDialog();

DGV\_CRIMINAL.DataSource = crim.getCriminal();

}

private void BT\_START\_SEARCH\_Click(object sender, EventArgs e)

{

Search search = new Search();

if (TB\_Search.Text != string.Empty)

{

if (RB\_FIO.Checked == true)

{

DGV\_CRIMINAL.DataSource = search.FIO(TB\_Search.Text);

}

if (RB\_Colour\_EYES.Checked == true)

{

DGV\_CRIMINAL.DataSource = search.Colour\_EYES(TB\_Search.Text);

}

if (RB\_COLOUR\_HAIRS.Checked == true)

{

DGV\_CRIMINAL.DataSource = search.Colour\_HAIRS(TB\_Search.Text);

}

if (RB\_ID.Checked == true)

{

DGV\_CRIMINAL.DataSource = search.ID(TB\_Search.Text);

}

if (RB\_Profile.Checked == true)

{

DGV\_CRIMINAL.DataSource = search.PROFILE(TB\_Search.Text);

}

if (RB\_Place\_Of\_Life.Checked == true)

{

DGV\_CRIMINAL.DataSource = search.BORN\_Place(TB\_Search.Text);

}

if (RB\_Monkie.Checked == true)

{

DGV\_CRIMINAL.DataSource = search.Monkey(TB\_Search.Text);

}

if (RB\_DATE.Checked)

{

DGV\_CRIMINAL.DataSource = search.DateOfBirthday(TB\_Search.Text);

}

}

else

{

MessageBox.Show("Введите критерий для поиска!","Пустое поле ПОИСКА");

}

}

private void BT\_SOURCE\_DATE\_Click(object sender, EventArgs e)

{

TB\_Search.Text = "";

DGV\_CRIMINAL.DataSource = crim.getCriminal();

}

private void DGV\_CRIMINAL\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

try

{

VIEW\_CRIMINAL\_FORM FH = new VIEW\_CRIMINAL\_FORM();

FH.Owner = this;

FH.ShowDialog();

DGV\_CRIMINAL.DataSource = crim.getCriminal();

}

catch

{

MessageBox.Show("Ой, что-то пошло не так!\nПроверьте подключение в БД");

}

}

}

}

// класс Поиска

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.SqlClient;

using System.Data;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace INTERPOL

{

class Search

{

public DataTable FIO(string Name)

{

string s = @"SELECT \* From get\_Criminals Where ФИО LIKE '%"+Name+"%'";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

public DataTable ID(string id)

{

try

{

string s = @"SELECT \* From get\_Criminals Where ID =" + Convert.ToInt32(id) + "";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

catch

{

MessageBox.Show("Введите цифры!");

return null;

}

}

public DataTable Colour\_EYES(string Colour)

{

string s = @"SELECT \* From get\_Criminals Where [Цвет глаз] ='"+Colour+"'";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

public DataTable Colour\_HAIRS(string Colour)

{

string s = @"SELECT \* From get\_Criminals Where [Цвет Волос] ='" + Colour + "'";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

public DataTable PROFILE(string profile)

{

string s = @"SELECT \* From get\_Criminals Where Профиль ='" + profile + "'";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

public DataTable BORN\_Place(string Place)

{

string s = @"SELECT \* From get\_Criminals Where [Место Рождение] ='" + Place + "'";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

public DataTable DateOfBirthday(string date)

{

try

{

string s = @"SELECT \* From get\_Criminals Where [Дата Рождения] ='" + Convert.ToDateTime(date) + "'";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

catch

{

MessageBox.Show("Введите корректную дату!");

return null;

}

}

public DataTable Monkey(string monkey)

{

string s = @"SELECT \* From get\_Criminals Where Кличка ='" + monkey + "'";

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand comm = new SqlCommand(s, con);

SqlDataAdapter adapter = new SqlDataAdapter(comm);

DataTable table = new DataTable();

adapter.Fill(table);

con.Close();

return table;

}

}

}

// класс Авторизации

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.SqlClient;

using System.Data;

using System.Threading.Tasks;

namespace INTERPOL

{

class Login

{

public DataTable login(string username, string password)

{

ConnectBD db = new ConnectBD();

SqlParameter[] param = new SqlParameter[2];

param[0] = new SqlParameter("@Login", SqlDbType.VarChar, 20);

param[0].Value = username;

param[1] = new SqlParameter("@pass", SqlDbType.VarChar, 20);

param[1].Value = password;

DataTable tab = new DataTable();

tab = db.getData("spr\_login", param);

db.closeConnection();

return tab;

}

}

}

// класс форма добавления нового преступника

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Data.SqlClient;

namespace INTERPOL

{

public partial class ADD\_NEW\_CRIMINALS : Form

{

Criminal crim = new Criminal();

public ADD\_NEW\_CRIMINALS()

{

InitializeComponent();

CB\_Prof.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

CB\_Profile.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

}

string server = @"433-004"; // RAINDASH-ПК\SQLEXPRESS Сервер !!!

public static MemoryStream ms = new MemoryStream();

private void TB\_ADD\_NEW\_CRIMINALS\_Click(object sender, EventArgs e)

{

try

{

if (PB\_PHOTO.Image == null)

{

if (TB\_FIO.Text != string.Empty)

{

crim.ADD\_NEW\_CRIM(Convert.ToInt32(CB\_Profile.SelectedValue), TB\_FIO.Text, TB\_MONKIE.Text, Convert.ToInt32(TB\_HIEGH.Text), TB\_NATIONALITY.Text, Convert.ToString(dateTimePicker1.Value.ToString()), TB\_PLACE\_OF\_BIRTH.Text, TB\_SEX.Text, TB\_EYES.Text, TB\_HAIR.Text, TB\_LANGUAGE.Text, Convert.ToInt32(CB\_Prof.SelectedValue), TB\_SPECIAL.Text, TB\_PLACE\_LIFE.Text);

MessageBox.Show("Преступник успешно добавлен!", "Успешно");

this.Close();

}

}

else

{

PB\_PHOTO.Image.Save(ms, PB\_PHOTO.Image.RawFormat);

byte[] image = ms.ToArray();

crim.AddCriminals(Convert.ToInt32(CB\_Profile.SelectedValue), TB\_FIO.Text, TB\_MONKIE.Text, Convert.ToInt32(TB\_HIEGH.Text), TB\_NATIONALITY.Text, Convert.ToString(dateTimePicker1.Value.ToString()), TB\_PLACE\_OF\_BIRTH.Text, TB\_SEX.Text, image, TB\_EYES.Text, TB\_HAIR.Text, TB\_LANGUAGE.Text, TB\_PLACE\_LIFE.Text, Convert.ToInt32(CB\_Prof.SelectedValue), TB\_SPECIAL.Text);

MessageBox.Show("Преступник успешно добавлен!", "Успешно");

this.Close();

}

}

catch

{

MessageBox.Show("Ой, что-то пошло не так, проверьте все поля!", "Ошибка");

}

}

private void BT\_PHOTO\_Click(object sender, EventArgs e)

{

OpenFileDialog ImagefileDialog = new OpenFileDialog();

ImagefileDialog.Filter = "Image |\*.JPG; \*.PNG; \*.GIF";

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

{

PB\_PHOTO.Image = Image.FromFile(ImagefileDialog.FileName);

MessageBox.Show("Фотография загруженна!", "Новая фотография", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

private void ADD\_NEW\_CRIMINALS\_Load(object sender, EventArgs e)

{

try

{

SqlConnection con = new SqlConnection(@"Server= " + Server.server + "; Database = interpol; Integrated Security = true");

con.Open();

SqlCommand ss = con.CreateCommand();

ss.CommandText = @"select id\_bands, name from bands";

SqlDataAdapter com = new SqlDataAdapter();

DataSet comm = new DataSet();

com.SelectCommand = ss;

com.Fill(comm, "bands");

CB\_Profile.DataSource = comm.Tables["bands"];

CB\_Profile.DisplayMember = "name";

CB\_Profile.ValueMember = "id\_bands";

ss.CommandText = @"select id\_profile, name from profile";

com.SelectCommand = ss;

com.Fill(comm, "profile");

CB\_Prof.DataSource = comm.Tables["profile"];

CB\_Prof.DisplayMember = "name";

CB\_Prof.ValueMember = "id\_profile";

}

catch

{

MessageBox.Show("Ошибка! Проверьте данные");

}

}

}

}

// форма добавлении нового пользователя

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace INTERPOL

{

public partial class ADD\_USER\_FORM : Form

{

User user = new User();

public ADD\_USER\_FORM()

{

InitializeComponent();

this.ActiveControl = TB\_LOGIN;

CB\_TYPE\_OF\_ACCESS.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

CB\_TYPE\_OF\_ACCESS.SelectedIndex = 1;

}

private void BT\_Create\_Click(object sender, EventArgs e)

{

if (TB\_LOGIN.Text == string.Empty || TB\_PASSWORD.Text == string.Empty)

{

MessageBox.Show("Заполните Все Поля!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

int TypeOfAccess;

if (CB\_TYPE\_OF\_ACCESS.SelectedIndex == 1)

{

TypeOfAccess = 0;

}

else

{

TypeOfAccess = 1;

}

user.insertUser(TB\_LOGIN.Text, TB\_PASSWORD.Text, TypeOfAccess.ToString());

MessageBox.Show("Новый пользователь успешно добавлен", "Новый Пользователь", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Close();

}

}

}

}

// класс банды

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Data;

namespace INTERPOL

{

class Band

{

public DataTable getBand()

{

ConnectBD db = new ConnectBD();

DataTable tab = new DataTable();

tab = db.getData("get\_Bandss", null);

db.closeConnection();

return tab;

}

public void insertBand(string name, int count, int lider, byte[] image)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[4];

parameters[0] = new SqlParameter("@name", SqlDbType.VarChar,50);

parameters[0].Value = name;

parameters[1] = new SqlParameter("@Count", SqlDbType.Int);

parameters[1].Value = count;

parameters[2] = new SqlParameter("@lider", SqlDbType.Int);

parameters[2].Value = lider;

parameters[3] = new SqlParameter("@img", SqlDbType.Image);

parameters[3].Value = image;

db.setData("spr\_inser\_bands", parameters);

db.closeConnection();

}

public void deleteBand(int id)

{

ConnectBD db = new ConnectBD();

DataTable table = new DataTable();

SqlParameter[] parameters = new SqlParameter[1];

parameters[0] = new SqlParameter("@cid", SqlDbType.Int);

parameters[0].Value = id;

db.openConnection();

db.setData("spr\_delete\_band", parameters);

db.closeConnection();

}

public void updateBand(string name,int count,string lider,string profile,int id)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[5];

parameters[0] = new SqlParameter("@c\_id", SqlDbType.Int);

parameters[0].Value = id;

parameters[1] = new SqlParameter("@c\_name", SqlDbType.VarChar, 50);

parameters[1].Value = name;

parameters[2] = new SqlParameter("@count", SqlDbType.Int);

parameters[2].Value = count;

parameters[3] = new SqlParameter("@lider", SqlDbType.VarChar, 120);

parameters[3].Value = lider;

parameters[4] = new SqlParameter("@prof", SqlDbType.VarChar, 120);

parameters[4].Value = profile;

db.setData("spr\_update\_band", parameters);

db.closeConnection();

}

}

}

//Класс пользователь

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Data;

namespace INTERPOL

{

class User

{

public DataTable getUsers()

{

ConnectBD db = new ConnectBD();

DataTable tab = new DataTable();

tab = db.getData("spr\_get\_user", null);

db.closeConnection();

return tab;

}

public void insertUser(string login, string password, string type)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[3];

parameters[0] = new SqlParameter("@login", SqlDbType.VarChar, 50);

parameters[0].Value = login;

parameters[1] = new SqlParameter("@Password", SqlDbType.VarChar, 50);

parameters[1].Value = password;

parameters[2] = new SqlParameter("@type", SqlDbType.VarChar, 50);

parameters[2].Value = type;

db.setData("spr\_Insert\_User", parameters);

db.closeConnection();

}

public void updateUser(string login, string password,string access,int uid)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[4];

parameters[0] = new SqlParameter("@Login", SqlDbType.VarChar, 20);

parameters[0].Value = login;

parameters[1] = new SqlParameter("@PASS", SqlDbType.VarChar, 100);

parameters[1].Value = password;

parameters[2] = new SqlParameter("@Type", SqlDbType.VarChar, 20);

parameters[2].Value = access;

parameters[3] = new SqlParameter("@id", SqlDbType.Int);

parameters[3].Value = uid;

db.setData("spr\_update\_user", parameters);

db.closeConnection();

}

public void deleteUser(int userId)

{

ConnectBD db = new ConnectBD();

db.openConnection();

SqlParameter[] parameters = new SqlParameter[1];

parameters[0] = new SqlParameter("@uid", SqlDbType.Int);

parameters[0].Value = userId;

db.setData("spr\_delete\_user", parameters);

db.closeConnection();

}

}

}