Государственное бюджетное профессиональное образовательное учреждение

Самарской области

«Тольяттинский социально-экономический колледж»

**КУРСОВАЯ РАБОТА**

**РАЗРАБОТКА И ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ «РАСЧЕТ СТОИМОСТИ УСЛУГ, ПРЕДОСТАВЛЯЕМЫХ КЛИЕНТАМ»**

**ПМ.05 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННЫХ СИСТЕМ**

**МДК 05.02 Разработка кода информационных систем**

**09.02.07 Информационные системы и программирование**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  | **/** | Д.А. Сардов |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |
| Оценка выполнения и защиты курсовой работы | | |  |
|  | | |  |
| Руководитель |  | **/** | Е.В. Плюснина |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |

Тольятти, 2022

Государственное бюджетное профессиональное образовательное учреждение  
Самарской области  
«Тольяттинский социально-экономический колледж»

|  |  |  |
| --- | --- | --- |
|  |  | Утверждаю:  Заместитель директора по УР  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.С. Киронова  *« » 202 г.* |

**ЗАДАНИЕ**

на курсовую работу

**по ПМ.05 Проектирование и разработка информационных систем модуля, выполняемой в рамках МДК.05.02 Разработка кода информационных систем**

студента группы ИСП-31

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Сардова Даниила Александровича\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Фамилия Имя отчество студента*

Тема курсовой работы: «Разработка информационной системы **«**Расчет стоимости услуг, предоставляемых клиентам»

1. Содержание задания:

1.1 Разработать физическую структуру информационной системы.

1.2 Разработать запросы к информационной системе.

1.3 Разработать приложение.

1. Исходные данные:

Исходные данные для практической реализации автоматизированной информационной системы (АИС) берутся из различных информационных источников (Интернет-ресурсы, печатные издания, периодика и др.).

1. Содержание курсовой работы

Введение

1 Разработка базы данных информационной системы (название)

1.1 Разработка физической структуры БД

1.2 Разработка запросов ИС

2 Разработка интерфейса информационной системы (название)

2.1 Разработка интерфейса

2.2 Разработка руководства пользователя

Заключение

Список использованных источников

Приложения

Приложение А Листинг программы

Дата выдачи задания: «13» января 2022 г.

Дата сдачи работы на отделение: «\_\_»\_\_\_\_\_\_\_\_2022 г.

Руководитель курсового(ой) проекта(работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Е.В. Плюснина

подпись расшифровка подписи

**КАЛЕНДАРНЫЙ ПЛАН**

выполнения курсовой работы

Студентом 3 курса группы ИСП-31

По теме Разработка и проектирование ИС «Расчет стоимости услуг, предоставляемых клиентам»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №  этапа  работы | Содержание этапов работы | Плановый срок выполнения этапа | Планируемый объем выполнения  этапа, % | Отметка  о  выполнении  этапа |
| 1 | Выбор, обоснование темы и объекта исследования | Январь 2022 | 5% |  |
| 2 | Утверждение темы, согласование плана. Введение, библиография | Январь 2022 | 10% |  |
| 3 | Изучение и анализ информационных материалов по теме | Февраль 2022 | 15% |  |
| 4 | Обоснование актуальности выбранной темы применительно к профессиональной деятельности (введение) | Февраль 2022 | 20% |  |
| 5 | Изложение материала основной части по теме курсовой работы | Февраль 2022 | 20% |  |
| 6 | Подведение итогов проведенного анализа, формулировка выводов УИР применительно к профессиональной деятельности (заключение) | Март 2022 | 20% |  |
| 7 | Оформление работы и сдача на проверку | Март 2022 | 10% |  |
| 8 | Защита работы |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  | **/** | Д.А. Сардов |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |
| Руководитель |  | **/** | Е.В. Плюснина |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#__RefHeading___Toc94035240)

[1 РАЗРАБОТКА БАЗЫ ДАННЫХ ИНФОРМАЦИОННОЙ СИСТЕМЫ РАСЧЕТ СТОИМОСТИ УСЛУГ, ПРЕДОСТАВЛЯЕМЫХ КЛИЕНТАМ 7](#__RefHeading___Toc94035241)

[1.1 Разработка физической структуры БД 7](#__RefHeading___Toc94035242)

[1.2 Разработка запросов ИС 10](#__RefHeading___Toc94035243)

[2 РАЗРАБОТКА ИНТЕРФЕЙСА ИНФОРМАЦИОННОЙ СИСТЕМЫ РАСЧЕТ СТОИМОСТИ УСЛУГ, ПРЕДОСТАВЛЯЕМЫХ КЛИЕНТАМ 12](#__RefHeading___Toc94035244)

2.1 Разработка интерфейса ……………………………………………………. 12

2.2 Разработка руководства пользователя ………………………….……….... 21

[ЗАКЛЮЧЕНИЕ 23](#__RefHeading___Toc94035246)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 24](#__RefHeading___Toc94035247)

[ПРИЛОЖЕНИЕ А.](#__RefHeading___Toc94035248) [ЛИСТИНГ ПРОГРАММЫ 26](#__RefHeading___Toc94035249)

# ВВЕДЕНИЕ

# В современном мире все больше и больше набирает популярность такое направление качественного звучания музыки в автомобиле, как автозвук. Истинные автолюбители, которые любят музыку и неравнодушны к качественному ее воспроизведению, не могут обойтись без хорошей акустической системы в своем авто. А также не стоит упускать тот факт, что автозвук стал видом спорта, по которому проходят соревнования.

# Соревнования по автозвуку — это мероприятия, на которых собираются как новички автозвука, так и профессионалы и между собой выясняют, чья система музыкальнее или громче. Соревнования традиционно подразделяют на три вида, которые учитывают конкретные параметры качества, мощности и работоспособности соревнующейся. При проведении каждого вида соревнования учитываются определенные нюансы и специфические правила. Так, на соревнованиях по качеству звука, судьями, в основном, учитывается точность и правильность музыкального воспроизведения стандартных треков, утверждённых ассоциациями, а также стандарты, в соответствии с которыми была установлена данная аудиосистема.

# Таким образом, сейчас развиваются автозвуковые студии, где предоставляют клиентам услуги по установке акустических систем. Им, разумеется, никак не обойтись без базы данных. Существует множество СУБД для их редактирования, но хотелось бы создать такое приложение, занимающее небольшой объем, простое в использовании и позволяющее редактировать базу данных без установленной СУБД, а также выполняющее расчет стоимости услуг. Именно этому и посвящена данная курсовая работа.

# При изучении предметной области, выяснилось, что информационная система «Расчет стоимости услуг, предоставляемых клиентам» на первый взгляд простая, но для ее реализации необходимо обратится к некоторым разделам, таким как создание базы данных и создание приложения для взаимодействия с этой базой данных.

# Объектом исследования курсовой работы является процесс расчета стоимости услуг, предоставляемых клиенту на выбор, в студии автозвука «Sound Studio».

# Предметом исследования процесса будет ИС «Расчет стоимости услуг, предоставляемых клиенту».

# Выше изложенное в целом на теоретико-методологическом уровне определило проблему настоящего исследования: разработка программного модуля «Расчет стоимости» в программе MS Visual Studio 2019.

# Целью курсовой работы является разработка и проектирование информационной системы «Расчет стоимости услуг, предоставляемых клиенту».

# Для достижения поставленной цели были определены следующие задачи:

# разработать базы данных информационной системы;

# разработать физическую структуру базы данных;

# разработать запросы ИС;

# разработать интерфейс ИС;

# разработать руководство пользователя.

# Для разработки информационной системы будут применяться методы: анализ деятельности предприятия с выявлением его функции, а также функциональное моделирование системы.

# Практическая значимость заключается в использовании разработанной системы на различных предприятиях по «Расчету стоимости услуг».

# Структура работы соответствует логике исследования и включает в себя введение, две главы, заключение, список использованной литературы, приложение.

# 1 РАЗРАБОТКА БАЗЫ ДАННЫХ ИНФОРМАЦИОННОЙ СИСТЕМЫ РАСЧЕТ СТОИМОСТИ УСЛУГ, ПРЕДОСТАВЛЯЕМЫХ КЛИЕНТАМ

## 1.1 Разработка физической структуры БД

Чтобы создать новую базу данных, откройте SQL Server Management Studio 18. В обозревателе объектов подключитесь к экземпляру компонента Компонент SQL Server Database Engine и разверните его. Щелкните правой кнопкой мыши узел Базы данных и выберите команду создать базу данных. В поле Новая база данных введите имя базы данных. Чтобы создать базу данных, приняв все значения по умолчанию, нажмите кнопку ОК.

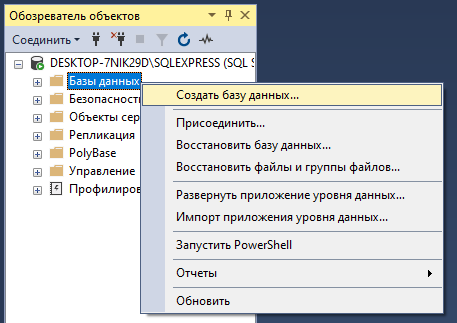


Рисунок 1.1 – Окно обозреватель объектов

В обозревателе объектов открываем контейнер «Базы данных», затем открываем нужную базу данных и щелкаем правой кнопкой мыши по пункту «Таблицы», после «Создать» и выбираем «Таблица…».

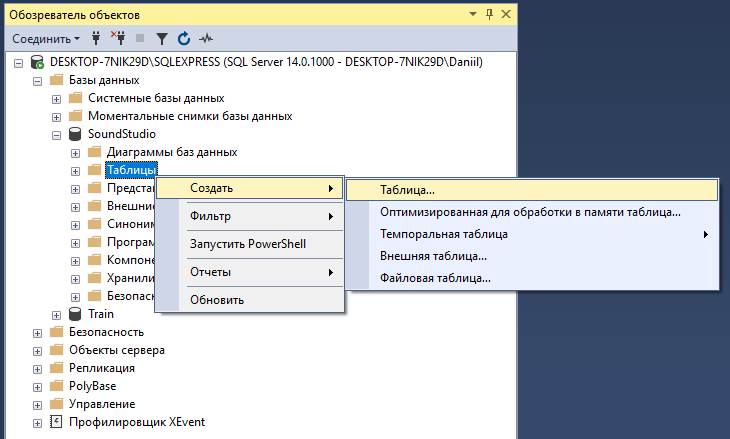


Рисунок 1.2 – Окно создания таблицы

Введите имена столбцов, выберите типы данных и определите для каждого столбца, могут ли в нем присутствовать значения NULL, как показано на следующей иллюстрации:

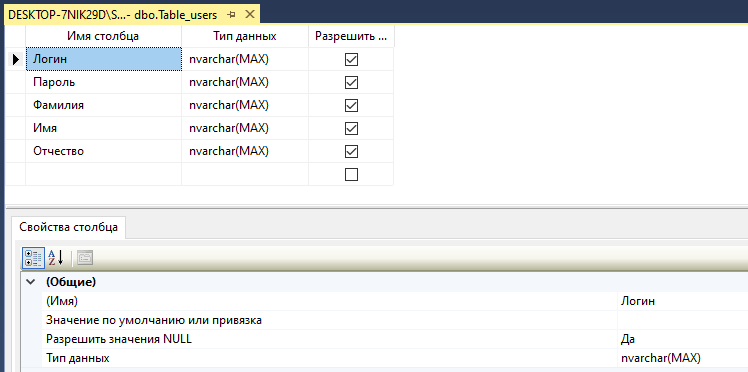


Рисунок 1.3 – Таблица «Пользователи»

Вы также можете задать другие свойства столбца, например, является ли этот столбец столбцом идентификаторов или вычисляемым столбцом. Для этого щелкните столбец на вкладке свойств столбцов. Чтобы указать, что столбец является столбцом первичного ключа, щелкните его правой кнопкой мыши и выберите «Задать первичный ключ». Чтобы создать связи по внешнему ключу, проверочные ограничения или индексы, щелкните правой кнопкой мыши панель конструктора таблиц и выберите в списке объект, как показано на следующей иллюстрации:

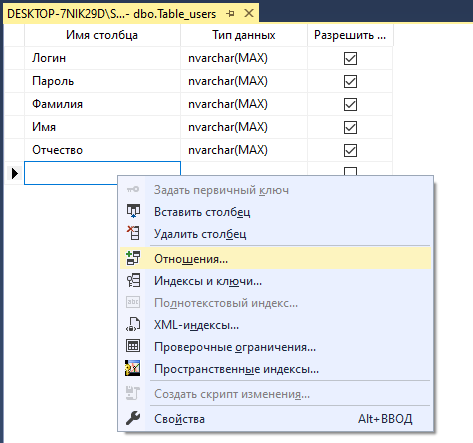


Рисунок 1.4 – Окно создания связей по внешнему ключу

По умолчанию таблица содержится в схеме dbo. Чтобы указать другую схему для таблицы, щелкните правой кнопкой мыши панель конструктора таблиц и выберите «Свойства», как показано на следующей иллюстрации:

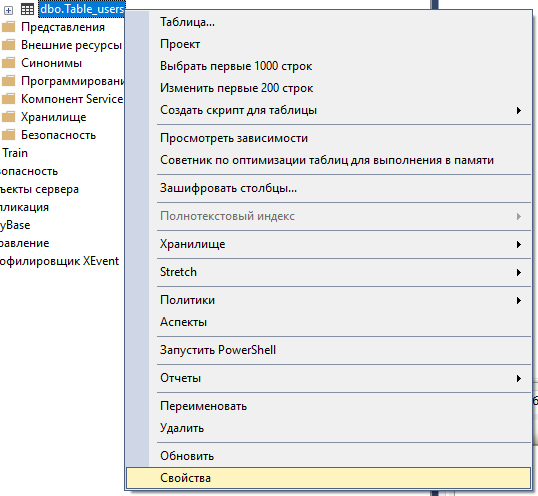


Рисунок 1.5 – Окно выбора свойств таблицы «Пользователи»

Выберите нужную схему из раскрывающегося списка «Схема».

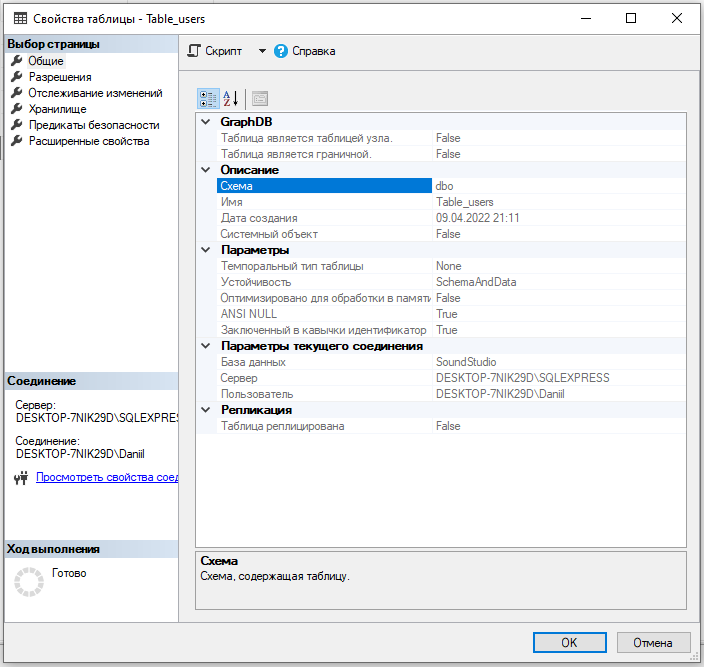


Рисунок 1.6 – Окно свойств таблицы «Пользователи»

В меню Файл выберите команду «Сохранить» и введите название таблицы – «имя\_таблицы».

Чтобы просмотреть новую таблицу, в обозревателе объектов разверните узел Таблицы, а затем нажмите клавишу F5, чтобы обновить список объектов. Новая таблица будет отображена в списке таблиц.

Создание остальных таблиц базы данных для информационной системы происходит аналогично примеру, описанному выше.

## 1.2 Разработка запросов ИС

Чтобы создать представление, в обозревателе объектов открываем контейнер «Базы данных», затем открываем нужную базу данных и щелкаем правой кнопкой мыши по пункту «Представления», после «Создать представление…».

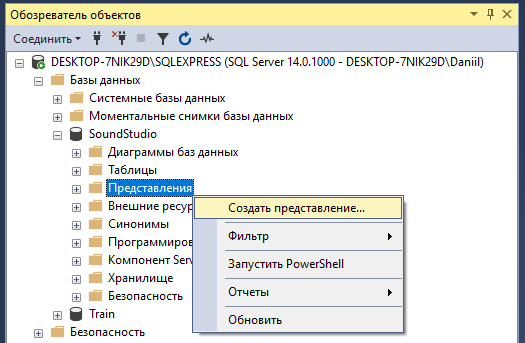


Рисунок 1.7 – Окно создания представления

В диалоговом окне «Добавление таблицы» выберите один или несколько элементов, которые необходимо включить в новое представление, на одной из следующих вкладок: «Таблицы», «Представления», «Функции» и «Синонимы». Щелкните «Добавить», а затем выберите Закрыть.

На Панели диаграмм выберите столбцы или другие элементы для включения в новое представление. На Панели критериев выберите дополнительные условия сортировки или фильтрации для столбцов.

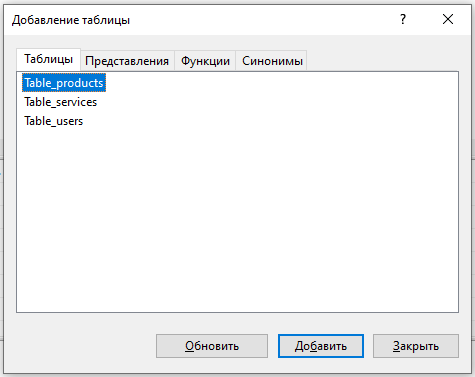


Рисунок 1.8 – Диалоговое окно добавление таблицы

В меню Файл выберите пункт «Сохранить» и введите название представления – «имя\_представления».

Затем создадим запрос для информационной системы путем выбора столбцов или других элементов представления. На панели критериев выберите дополнительные условия сортировки или фильтрации для столбцов.

Общая структура запроса выглядит следующим образом:

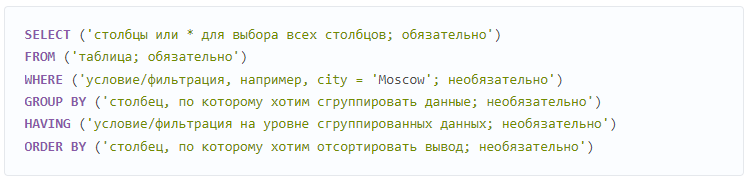


Рисунок 1.9 – Общая структура запроса

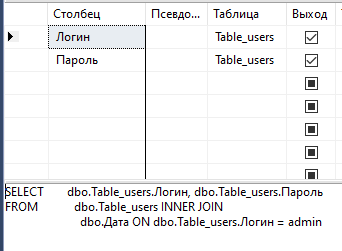


Рисунок 1.10 – Запрос Select для информационной системы

# Создание остальных запросов базы данных для информационной системы происходит аналогично примеру, описанному выше.

# 2 РАЗРАБОТКА ИНТЕРФЕЙСА ИНФОРМАЦИОННОЙ СИСТЕМЫ РАСЧЕТ СТОИМОСТИ УСЛУГ, ПРЕДОСТАВЛЯЕМЫХ КЛИЕНТАМ

## 2.1 Разработка интерфейса

Первым шагом является создание инфраструктуры приложения. Создайте новый проект приложения WPF в Visual Basic или Visual C# и введите название. Откройте Visual Studio и выберите создать новый проект в меню начало работы. Откроется диалоговое окно «Создание нового проекта». В раскрывающемся списке язык выберите C#. Выберите шаблон приложения WPF (платформа .NET Framework) и нажмите кнопку далее.

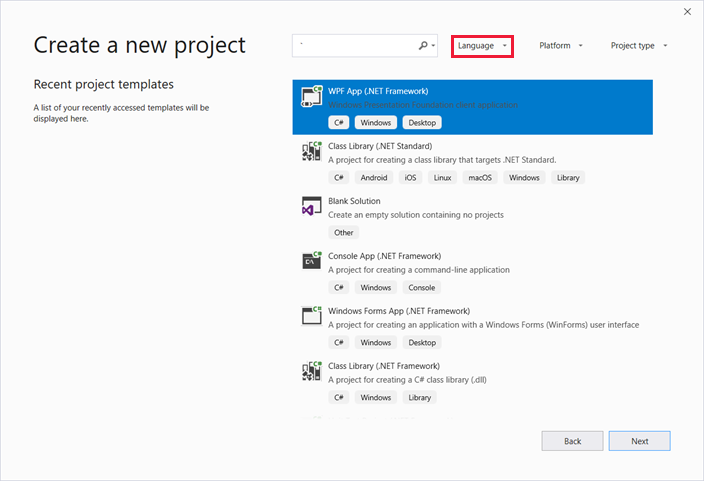


Рисунок 2.1 – Окно создания приложения WPF

Откроется диалоговое окно «Настройка нового проекта». Введите имя проекта и нажмите кнопку создать.

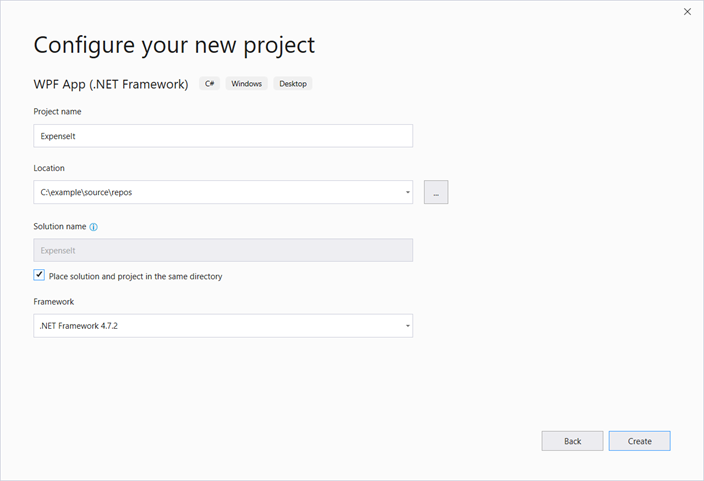


Рисунок 2.2 – Окно создания приложения WPF

Visual Studio создает проект и открывает конструктор для окна приложения по умолчанию с именем MainWindow.xaml.

Чтобы как-то взаимодействовать с пользователем, получать от пользователя ввод с клавиатуры или мыши и использовать введенные данные в программе, нам нужны элементы управления.

Создаем интерфейс окна авторизации, используя элементы управления WPF – Grid, Label, Border, Image для визуального восприятия информации; TextBox, PasswordBox для ввода логина и пароля; CheckBox для просмотра пароля; Button для закрытия окна и для входа в систему.

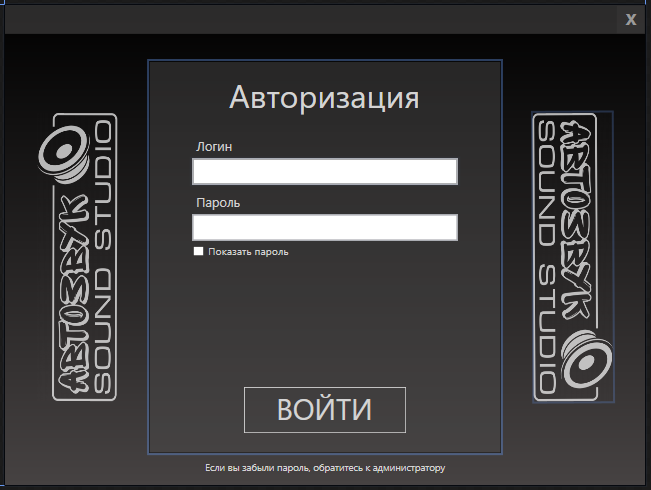


Рисунок 2.3 – Интерфейс окна «Авторизация»

Затем откроем App.xaml (C#). Этот XAML-файл определяет приложение WPF и все ресурсы приложения. Этот файл также используется для указания пользовательского интерфейса, в данном случае окна авторизации, которое автоматически отображается при запуске приложения.

Код XAML для окна авторизации выглядит следующим образом:

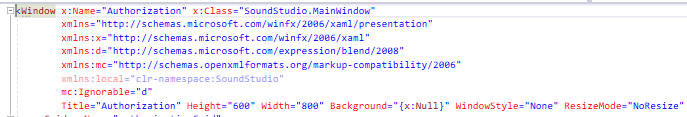


Рисунок 2.4 – XAML-код окна авторизации

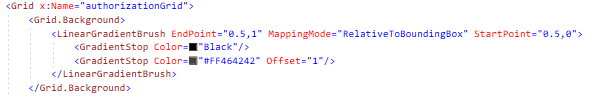


Рисунок 2.5 – XAML-код Grid

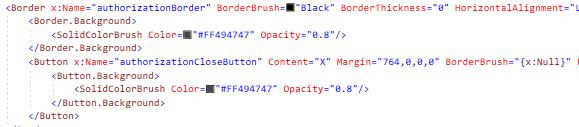


Рисунок 2.6 – XAML-код Border с кнопками

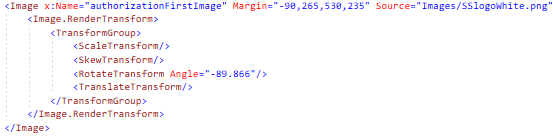


Рисунок 2.7 – XAML-код Image

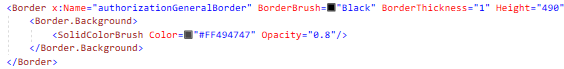


Рисунок 2.8 – XAML-код Border

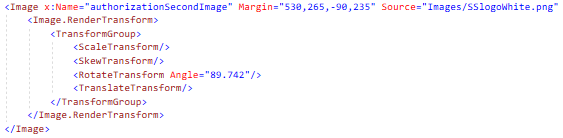


Рисунок 2.9 – XAML-код Image

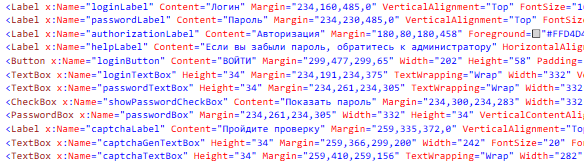


Рисунок 2.10 – XAML-код формы ввода данных

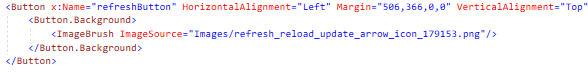


Рисунок 2.11 – XAML-код Button

После этого добавим проверку (Captcha) в случае неверного ввода логина или пароля в окно авторизации. Интерфейс окна авторизации с проверкой будет выглядеть следующим образом:

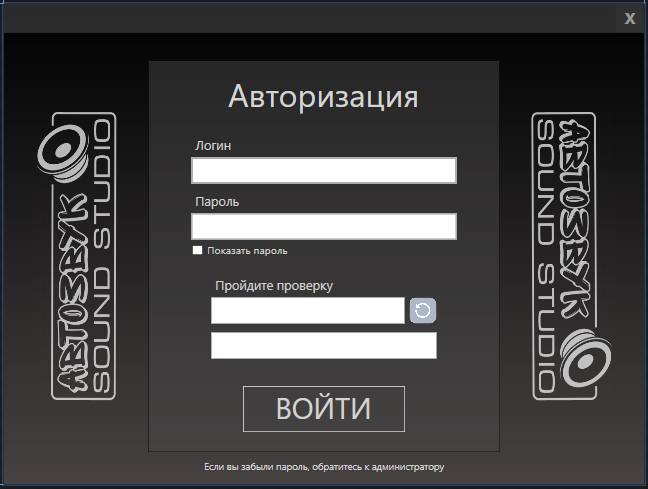


Рисунок 2.12 – Интерфейс окна «Авторизация» с проверкой ввода логина или пароля

Следом перейдем к созданию кода на языке C# для окна авторизации. В обозревателе решений нажмем правой кнопкой мыши по проекту, выберем пункт «Добавить» - «Создать элемент».

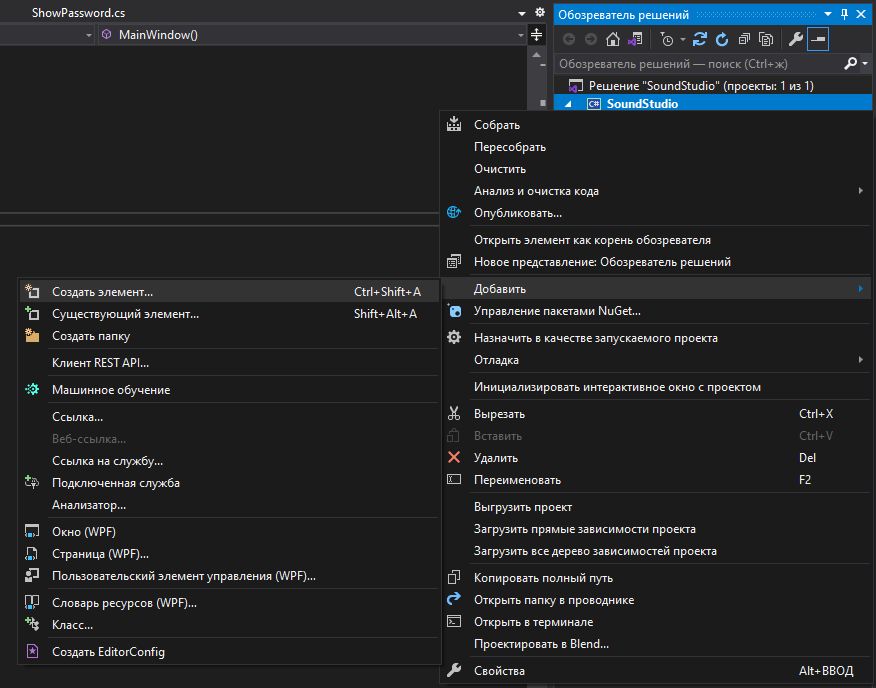


Рисунок 2.13 – Окно обозреватель решений

В открывшемся окне «Добавление нового элемента» выберем элемент «Класс Visual C#» и дадим ему название. Нажмем кнопку «Добавить» и перейдем к нему.

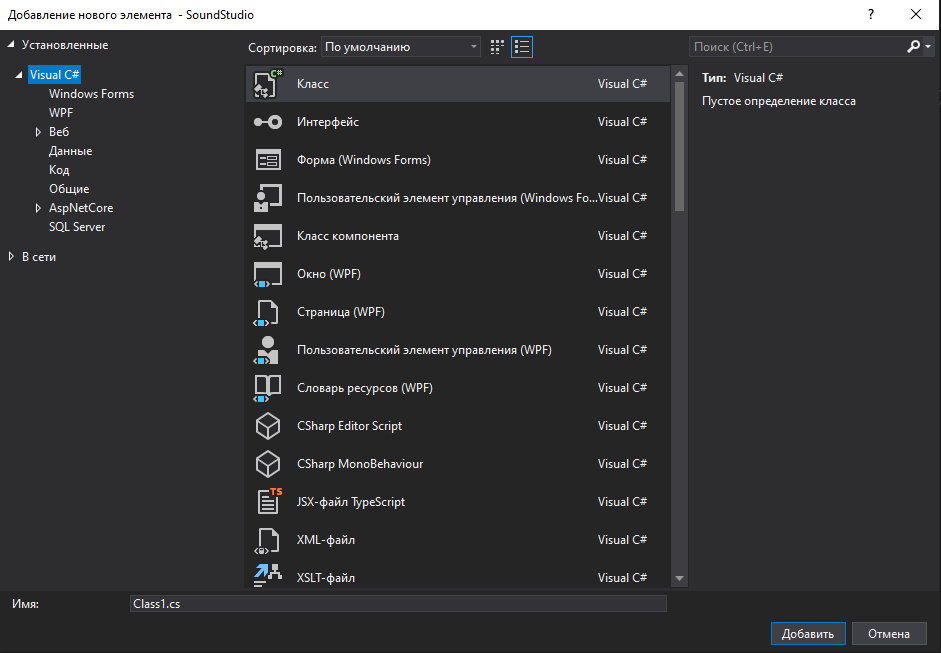


Рисунок 2.14 – Окно добавления нового класса

Создадим класс Connection, который будет брать данные о пользователе из sql-базы данных, используя метод Select.

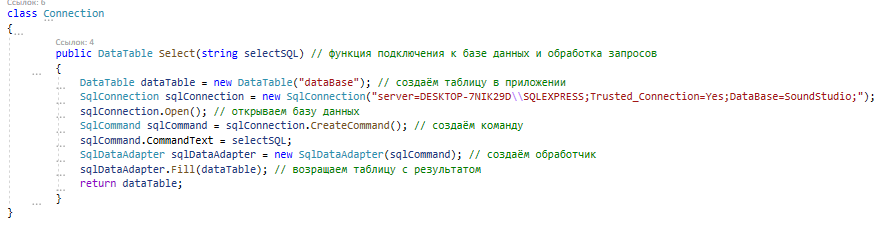


Рисунок 2.15 – Код класса Connection

Затем сформируем класс Login, отвечающий за вход в систему. В этом классе будет обращение к классу Connection для получения информации из sql-базы данных о пользователе. Поставим условия входа – при удачном вводе логина и пароля, пользователь попадает на главное окно. При ошибке ввода добавляется проверка (Captcha) с возможность обновления.



Рисунок 2.16 – Код класса Login с условием удачного ввода логина и пароля

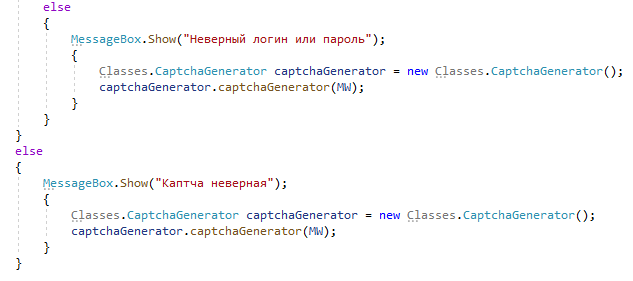


Рисунок 2.17 – Код класса Login с условием неудачного ввода логина и пароля

Для показа введенного пароля на окне авторизации создадим класс ShowPassword.

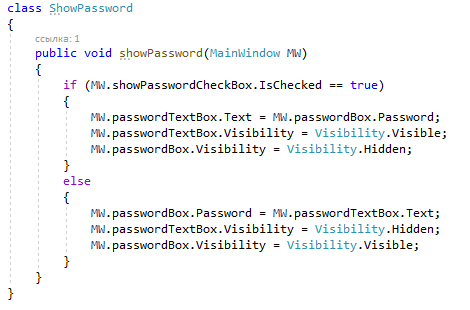


Рисунок 2.18 – Код класса ShowPassword

Для проверки неверного ввода логина или пароля создадим класс CaptchaGenerator, который будет генерировать проверку (Captcha).

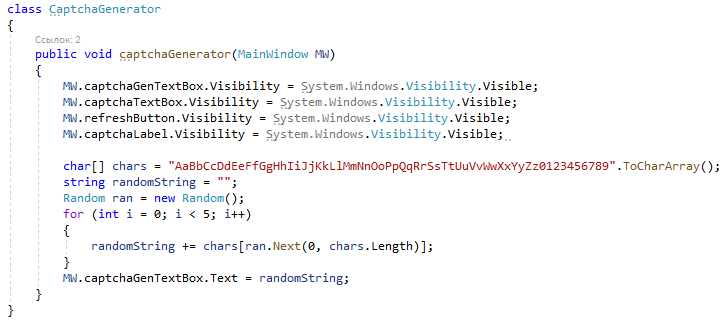


Рисунок 2.19 – Код класса CaptchaGenerator

Чтобы была возможность обновить проверку (Captcha), нам понадобится класс Refresh, приведенный на рисунке 2.20.

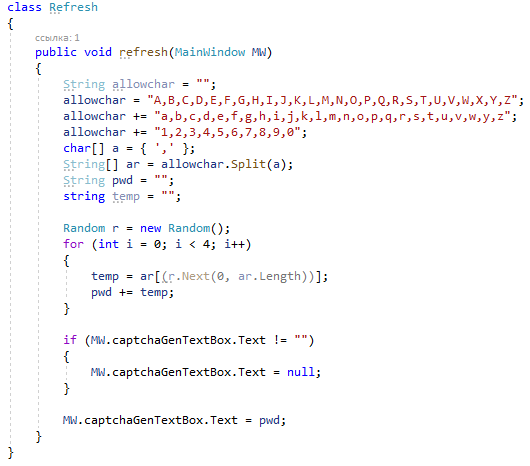


Рисунок 2.20 – Код класса Refresh

Конечный код для окна авторизации, в котором будут вызываться все созданные классы, будет выглядеть следующим образом:



Рисунок 2.21 – Код окна авторизации

## Следом добавим главное окно проекта. Чтобы осуществить открытие нового окна в WPF приложении, для начала нужно создать окно. Для этого правой кнопкой мыши (ПКМ) нажимаем на название проекта в «Обозреватель решений» — пункт «Добавить» — «Окно».

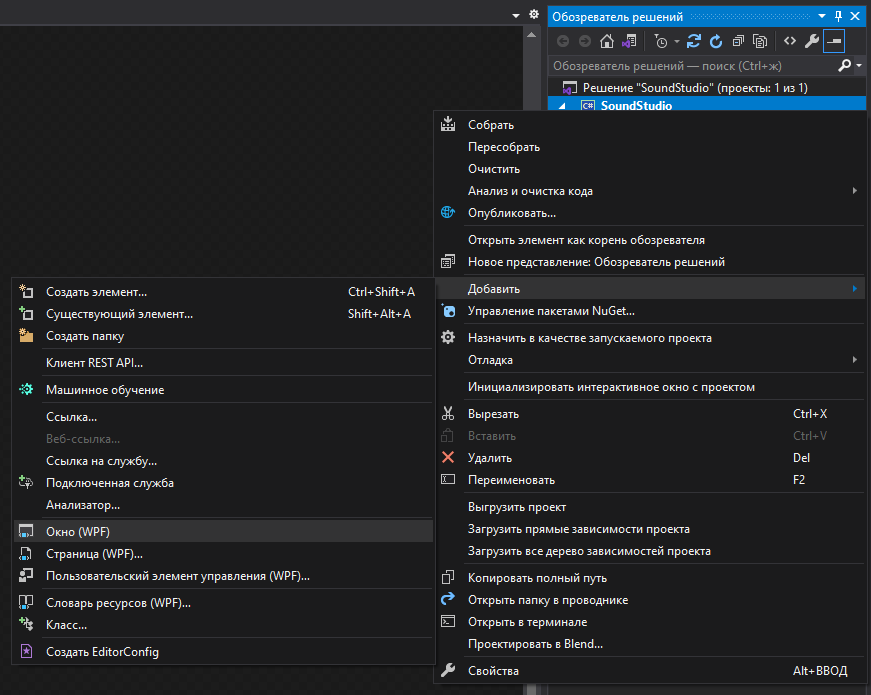


Рисунок 2.22 – Добавление главного окна проекта

После нужно указать название и нажать кнопку «Далее».

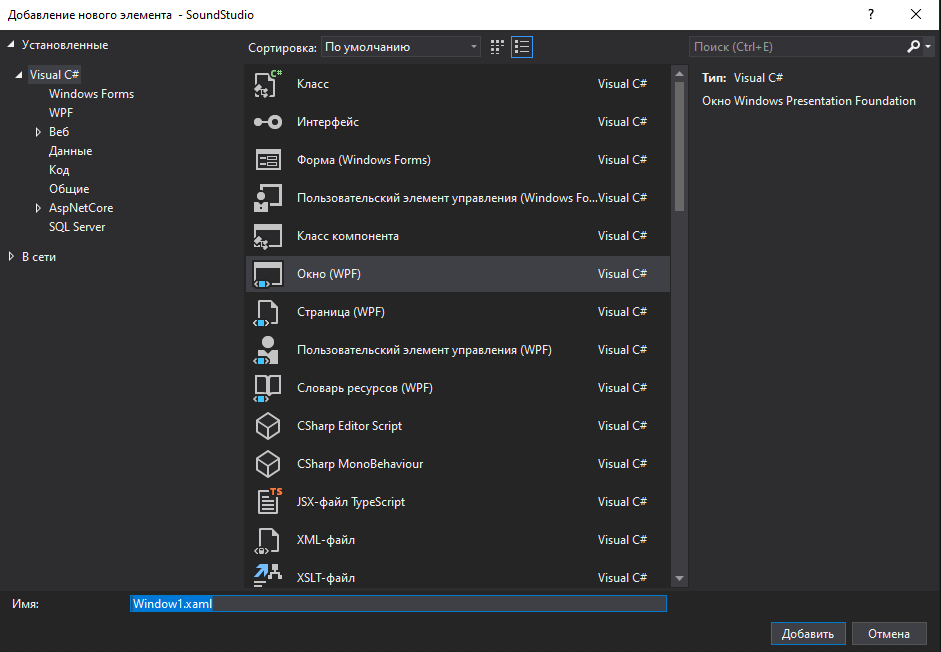


Рисунок 2.23 – Добавление имени главного окна проекта

Интерфейс главного окна и программный код разрабатываются аналогично описанному выше примеру.

## 2.2 Разработка руководства пользователя

Информационная система «Sound Studio» представляет собой программный комплекс для ознакомления заказчиков и потенциальных клиентов с деятельностью и юридической информацией об организации ООО «Sound Studio».

Данное руководство предназначено для пользователей приложения, установленного на устройствах, работающих под управлением операционной системы Windows 10.

Для установки приложения компьютер конечного пользователя должен удовлетворять требованиям.

Минимальная конфигурация сервера:

* + - тип процессора – Pentium IV и выше или совместимый с ним;
    - объем оперативного запоминающего устройства 4Гб и более;
    - жесткий диск 80 Гб;
    - модем, для выхода в Интернет;
    - монитор, клавиатура, мышь.

Требования, предъявляемые к конфигурации клиентских станций:

* + - процессор, с тактовой частотой не менее 1200 MHz,
    - 2 GB оперативной памяти;
    - монитор – SVGA;
    - клавиатура - 101/102 клавиши;
    - манипулятор типа «мышь».

Для реализации данной системы требуется установка следующего ПО: Microsoft Visual Studio 2021 и СУБД Microsoft SQL Server 2018 Express Edition.

Минимальная конфигурация:

* + - процессор x86 (Intel, AMD) с частотой как минимум 1,8 ГГц;
    - 2 ГБ ОЗУ; рекомендуется 4 ГБ ОЗУ;
    - доступное место на жестком диске объемом от 2 до 10 ГБ.

Для авторизации введите логин и пароль в открывшемся окне при входе в приложение, затем нажмите кнопку «ВОЙТИ». Для просмотра введенного вами пароля поставьте галочку рядом с «Показать пароль».

Открытие главного окна обозначит успешный вход в приложение.

Диалоговое окно «Неправильный логин или пароль» обозначит ошибку ввода логина или пароля. Нажмите «Ок» и попробуйте ввести данные заново, дополнительно пройдя проверку (Captcha). Если вам сложно различить символы в предложенной автоматически проверке, вы можете обновить ее, нажав рядом кнопку «Обновить».

Для закрытия окна авторизации нажмите кнопку «Х» в правом верхнем углу экрана.

На главном окне находится вступительная информация о компании «Sound Studio», в верхнем левом углу расположена информация о пользователе.

В боковом меню, расположенном на главном окне слева, представлены разделы:

* + - Главная (содержит главную страницу приложения);
    - Товары (представлены товары студии);
    - Услуги (представлен перечень услуг, предоставляемых клиентам);
    - Корзина (содержит выбранные клиентом услуги и их стоимость, а также штрих-код);
    - О нас (представлена информация о студии);
    - Контакты (представлена контактная информация студии).

Для возврата из главного окна на окно авторизации нажмите кнопку «Выйти», расположенную в правом верхнем углу.

Для закрытия главного окна нажмите кнопку «X» в правом верхнем углу экрана.

# ЗАКЛЮЧЕНИЕ

В первой части курсовой работы была произведена разработка базы данных информационной системы «Расчет стоимости услуг, предоставляемых клиентам». Была разработана физическая структура базы данных для информационной системы. Также, был рассмотрен процесс разработки запросов ИС «Расчет стоимости услуг, предоставляемых клиентам». В дальнейшем создана база данных системы в программной среде MS SQL Server Management Studio 18.

Во второй части курсовой работы на основе рассмотренных процессов отдела «Расчет стоимости услуг, предоставляемых клиентам» был разработан и подробно описан интерфейс информационной системы в программе MS Visual Studio 19, а также было разработано руководство пользователя.

Затем был разработан программный модуль автоматизированной информационной системы «Расчет стоимости услуг, предоставляемых клиентам» с собственным интерфейсом и запросами при помощи программ Microsoft Visual Studio 2019 и MS SQL Server Management Studio 18.

Подробно описан и проиллюстрирован функционал и структура самого модуля, а также проведено тестирование готового программного продукта. В последнюю очередь была составлена сопутствующая документация.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 34.601 – 90. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.
2. ГОСТ 34.602 - 2020. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы.
3. ГОСТ 19.201 - 78 ЕСПД. Техническое задание. Требования к содержанию и оформлению.
4. ГОСТ 19.202 - 78 ЕСПД. Спецификация. Требования к содержанию и оформлению.
5. ГОСТ Р ИСО/МЭК 12207. Процессы жизненного цикла программных средств.
6. Золотов С.Ю. Проектирование информационных систем: Учебно-методическое пособие. Методические рекомендации для выполнения курсового проекта, лабораторных работ и практических занятий по дисциплине «Проектирование информационных систем» - Томск: ТУСУР, 2013. - 34 с.
7. Шнайдер, Роберт Microsoft SQL Server 6.5. Проектирование высокопроизводительных баз данных; М.: Лори, 2010. - 361 c
8. Петкович, Душан Microsoft SQL Server 2012. Руководство для начинающих / Душан Петкович. - М.: БХВ-Петербург, 2012. - 460 c.
9. Тейлор, Аллен SQL для чайников / Аллен Тейлор. - М.: Вильямс, 2014. - 416 c.
10. Браст, Э.Дж. Разработка приложений на основе Microsoft SQL Server 2008 / Э.Дж. Браст. - М.: Русская Редакция, 2010. - 751 c.
11. Хетагуров, Я. А. Проектирование автоматизированных систем обработки информации и управления (АСОИУ). Учебник / Я.А. Хетагуров. - М.: Бином. Лаборатория знаний, 2015. - 240 c.
12. Кристофер, Д. Маннинг Введение в информационный поиск / Кристофер Д. Маннинг, ПрабхакарРагхаван ,ХайнрихШютце. - М.: Вильямс, 2014. - 528 c.
13. Бишоп Дж. С# в кратком изложении; Бином. Лаборатория знаний - М., 2015. - 234 c.
14. ГриффитсИэн Программирование на C# 5.0; Эксмо - М., 2014. - 580 c.
15. Гуриков С. Р. Введение в программирование на языке Visual C#; ИЛ - Москва, 2013. - 448 c.
16. Шилдт Герберт C# 4.0. Полное руководство; Вильямс - М., 2015. - 291 c.
17. Эндрю Троелсен Язык программирования C# 5.0 и платформа .NET 4.5; Диалектика / Вильямс - М., 2015. - 126 c.

# ПРИЛОЖЕНИЕ А

# ЛИСТИНГ ПРОГРАММЫ

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using System.Windows.Threading;

using System.Data;

using System.Data.SqlClient;

namespace SoundStudio

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

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

{

base.OnClosed(e);

App.Current.Shutdown();

}

private void authorizationBorder\_MouseDown(object sender, MouseButtonEventArgs e)

{

this.DragMove();

}

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

{

Classes.ShowPassword showPassword = new Classes.ShowPassword();

showPassword.showPassword(this);

}

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

{

Classes.Refresh refresh = new Classes.Refresh();

refresh.refresh(this);

}

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

{

Classes.Login login = new Classes.Login();

login.login(this);

}

}

}

using System;

using System.Windows;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SoundStudio.Classes

{

class ShowPassword

{

public void showPassword(MainWindow MW)

{

if (MW.showPasswordCheckBox.IsChecked == true)

{

MW.passwordTextBox.Text = MW.passwordBox.Password;

MW.passwordTextBox.Visibility = Visibility.Visible;

MW.passwordBox.Visibility = Visibility.Hidden;

}

else

{

MW.passwordBox.Password = MW.passwordTextBox.Text;

MW.passwordTextBox.Visibility = Visibility.Hidden;

MW.passwordBox.Visibility = Visibility.Visible;

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SoundStudio.Classes

{

class Refresh

{

public void refresh(MainWindow MW)

{

String allowchar = "";

allowchar = "A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z";

allowchar += "a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,y,z";

allowchar += "1,2,3,4,5,6,7,8,9,0";

char[] a = { ',' };

String[] ar = allowchar.Split(a);

String pwd = "";

string temp = "";

Random r = new Random();

for (int i = 0; i < 4; i++)

{

temp = ar[(r.Next(0, ar.Length))];

pwd += temp;

}

if (MW.captchaGenTextBox.Text != "")

{

MW.captchaGenTextBox.Text = null;

}

MW.captchaGenTextBox.Text = pwd;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

namespace SoundStudio.Classes

{

class Connection

{

public DataTable Select(string selectSQL) // функция подключения к базе данных и обработка запросов

{

DataTable dataTable = new DataTable("dataBase"); // создаём таблицу в приложении

SqlConnection sqlConnection = new SqlConnection("server=DESKTOP-7NIK29D\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=SoundStudio;");

sqlConnection.Open(); // открываем базу данных

SqlCommand sqlCommand = sqlConnection.CreateCommand(); // создаём команду

sqlCommand.CommandText = selectSQL;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable); // возращаем таблицу с результатом

return dataTable;

}

}

}

using System;

using System.Windows;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SoundStudio.Classes

{

class CaptchaGenerator

{

public void captchaGenerator(MainWindow MW)

{

MW.captchaGenTextBox.Visibility = System.Windows.Visibility.Visible;

MW.captchaTextBox.Visibility = System.Windows.Visibility.Visible;

MW.refreshButton.Visibility = System.Windows.Visibility.Visible;

MW.captchaLabel.Visibility = System.Windows.Visibility.Visible;

char[] chars = "AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz0123456789".ToCharArray();

string randomString = "";

Random ran = new Random();

for (int i = 0; i < 5; i++)

{

randomString += chars[ran.Next(0, chars.Length)];

}

MW.captchaGenTextBox.Text = randomString;

}

}

}

using System;

using System.Windows;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

namespace SoundStudio.Classes

{

class Login

{

string log;

int index;

public void login(MainWindow MW)

{

if (MW.loginTextBox.Text.Length > 0)

{

if (MW.passwordBox.Password.Length > 0)

{

if (MW.captchaTextBox.Text == MW.captchaGenTextBox.Text || MW.captchaGenTextBox.Text.Length == 0)

{

Classes.Connection connection = new Classes.Connection();

DataTable dt\_SoundStudio = connection.Select("SELECT \* FROM [dbo].[Table\_users] WHERE [Логин] = '" + MW.loginTextBox.Text + "' AND [Пароль] = '" + MW.passwordBox.Password + "'");

if (dt\_SoundStudio.Rows.Count > 0)

{

dt\_SoundStudio = connection.Select("SELECT \* FROM [dbo].[Table\_users]"); // данные из БД

for (int i = 0; i < dt\_SoundStudio.Rows.Count; i++) // перебираем данные

{

log = dt\_SoundStudio.Rows[i][0].ToString();

if (log == MW.loginTextBox.Text.ToString())

{

index = i;

}

}

SoundStudioWindow sswin = new SoundStudioWindow();

sswin.Owner = MW;

sswin.userLabel.Content = dt\_SoundStudio.Rows[index][2].ToString() + " " + dt\_SoundStudio.Rows[index][3].ToString() + " " + dt\_SoundStudio.Rows[index][4].ToString();

sswin.Show();

MW.Hide();

}

else

{

MessageBox.Show("Неверный логин или пароль");

{

Classes.CaptchaGenerator captchaGenerator = new Classes.CaptchaGenerator();

captchaGenerator.captchaGenerator(MW);

}

}

}

else

{

MessageBox.Show("Каптча неверная");

{

Classes.CaptchaGenerator captchaGenerator = new Classes.CaptchaGenerator();

captchaGenerator.captchaGenerator(MW);

}

}

}

}

}

}

}

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Windows.Threading;

using BarcodeLib;

using System.Drawing.Imaging;

namespace SoundStudio

{

/// <summary>

/// Логика взаимодействия для SoundStudioWindow.xaml

/// </summary>

public partial class SoundStudioWindow : Window

{

public SoundStudioWindow()

{

InitializeComponent();

homePageBorder.Visibility = Visibility.Visible;

homePageImage.Visibility = Visibility.Visible;

}

private void soundStudioBorder\_MouseDown(object sender, MouseButtonEventArgs e)

{

this.DragMove();

}

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

{

base.OnClosed(e);

App.Current.Shutdown();

}

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

{

MainWindow mwin = new MainWindow();

mwin.Show();

this.Close();

}

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

{

contactsBorder.Visibility = Visibility.Hidden;

contactsLabelOne.Visibility = Visibility.Hidden;

contactsLabelTwo.Visibility = Visibility.Hidden;

contactsLabelThree.Visibility = Visibility.Hidden;

contactsLabelFour.Visibility = Visibility.Hidden;

instImage.Visibility = Visibility.Hidden;

vkImage.Visibility = Visibility.Hidden;

contactsLabelImage.Visibility = Visibility.Hidden;

homePageBorder.Visibility = Visibility.Hidden;

homePageImage.Visibility = Visibility.Hidden;

productsListView.Visibility = Visibility.Hidden;

productsInfoLabel.Visibility = Visibility.Hidden;

servicesListView.Visibility = Visibility.Hidden;

inBasketButton.Visibility = Visibility.Hidden;

costBasketListView.Visibility = Visibility.Hidden;

inBasketDeleteButton.Visibility = Visibility.Hidden;

costLabel.Visibility = Visibility.Hidden;

paymentReceiptButton.Visibility = Visibility.Hidden;

aboutUsBorder.Visibility = Visibility.Visible;

aboutUsLabelOne.Visibility = Visibility.Visible;

aboutUsLabelTwo.Visibility = Visibility.Visible;

aboutUsLabelThree.Visibility = Visibility.Visible;

aboutUsLabelFour.Visibility = Visibility.Visible;

aboutUsLabelFive.Visibility = Visibility.Visible;

aboutUsLabelSix.Visibility = Visibility.Visible;

aboutUsLabelSeven.Visibility = Visibility.Visible;

aboutUsLabelEight.Visibility = Visibility.Visible;

aboutUsLabelNine.Visibility = Visibility.Visible;

}

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

{

aboutUsBorder.Visibility = Visibility.Hidden;

aboutUsLabelOne.Visibility = Visibility.Hidden;

aboutUsLabelTwo.Visibility = Visibility.Hidden;

aboutUsLabelThree.Visibility = Visibility.Hidden;

aboutUsLabelFour.Visibility = Visibility.Hidden;

aboutUsLabelFive.Visibility = Visibility.Hidden;

aboutUsLabelSix.Visibility = Visibility.Hidden;

aboutUsLabelSeven.Visibility = Visibility.Hidden;

aboutUsLabelEight.Visibility = Visibility.Hidden;

aboutUsLabelNine.Visibility = Visibility.Hidden;

homePageBorder.Visibility = Visibility.Hidden;

homePageImage.Visibility = Visibility.Hidden;

productsListView.Visibility = Visibility.Hidden;

productsInfoLabel.Visibility = Visibility.Hidden;

servicesListView.Visibility = Visibility.Hidden;

inBasketButton.Visibility = Visibility.Hidden;

costBasketListView.Visibility = Visibility.Hidden;

inBasketDeleteButton.Visibility = Visibility.Hidden;

costLabel.Visibility = Visibility.Hidden;

paymentReceiptButton.Visibility = Visibility.Hidden;

contactsBorder.Visibility = Visibility.Visible;

contactsLabelOne.Visibility = Visibility.Visible;

contactsLabelTwo.Visibility = Visibility.Visible;

contactsLabelThree.Visibility = Visibility.Visible;

contactsLabelFour.Visibility = Visibility.Visible;

instImage.Visibility = Visibility.Visible;

vkImage.Visibility = Visibility.Visible;

contactsLabelImage.Visibility = Visibility.Visible;

}

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

{

aboutUsBorder.Visibility = Visibility.Hidden;

aboutUsLabelOne.Visibility = Visibility.Hidden;

aboutUsLabelTwo.Visibility = Visibility.Hidden;

aboutUsLabelThree.Visibility = Visibility.Hidden;

aboutUsLabelFour.Visibility = Visibility.Hidden;

aboutUsLabelFive.Visibility = Visibility.Hidden;

aboutUsLabelSix.Visibility = Visibility.Hidden;

aboutUsLabelSeven.Visibility = Visibility.Hidden;

aboutUsLabelEight.Visibility = Visibility.Hidden;

aboutUsLabelNine.Visibility = Visibility.Hidden;

contactsBorder.Visibility = Visibility.Hidden;

contactsLabelOne.Visibility = Visibility.Hidden;

contactsLabelTwo.Visibility = Visibility.Hidden;

contactsLabelThree.Visibility = Visibility.Hidden;

contactsLabelFour.Visibility = Visibility.Hidden;

instImage.Visibility = Visibility.Hidden;

vkImage.Visibility = Visibility.Hidden;

contactsLabelImage.Visibility = Visibility.Hidden;

productsListView.Visibility = Visibility.Hidden;

productsInfoLabel.Visibility = Visibility.Hidden;

servicesListView.Visibility = Visibility.Hidden;

inBasketButton.Visibility = Visibility.Hidden;

costBasketListView.Visibility = Visibility.Hidden;

inBasketDeleteButton.Visibility = Visibility.Hidden;

costLabel.Visibility = Visibility.Hidden;

paymentReceiptButton.Visibility = Visibility.Hidden;

homePageBorder.Visibility = Visibility.Visible;

homePageImage.Visibility = Visibility.Visible;

}

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

{

homePageBorder.Visibility = Visibility.Hidden;

homePageImage.Visibility = Visibility.Hidden;

aboutUsBorder.Visibility = Visibility.Hidden;

aboutUsLabelOne.Visibility = Visibility.Hidden;

aboutUsLabelTwo.Visibility = Visibility.Hidden;

aboutUsLabelThree.Visibility = Visibility.Hidden;

aboutUsLabelFour.Visibility = Visibility.Hidden;

aboutUsLabelFive.Visibility = Visibility.Hidden;

aboutUsLabelSix.Visibility = Visibility.Hidden;

aboutUsLabelSeven.Visibility = Visibility.Hidden;

aboutUsLabelEight.Visibility = Visibility.Hidden;

aboutUsLabelNine.Visibility = Visibility.Hidden;

contactsBorder.Visibility = Visibility.Hidden;

contactsLabelOne.Visibility = Visibility.Hidden;

contactsLabelTwo.Visibility = Visibility.Hidden;

contactsLabelThree.Visibility = Visibility.Hidden;

contactsLabelFour.Visibility = Visibility.Hidden;

instImage.Visibility = Visibility.Hidden;

vkImage.Visibility = Visibility.Hidden;

contactsLabelImage.Visibility = Visibility.Hidden;

servicesListView.Visibility = Visibility.Hidden;

inBasketButton.Visibility = Visibility.Hidden;

costBasketListView.Visibility = Visibility.Hidden;

inBasketDeleteButton.Visibility = Visibility.Hidden;

costLabel.Visibility = Visibility.Hidden;

paymentReceiptButton.Visibility = Visibility.Hidden;

productsListView.Visibility = Visibility.Visible;

productsInfoLabel.Visibility = Visibility.Visible;

productsListView.Items.Clear();

Classes.LoadProducts loadProducts = new Classes.LoadProducts();

loadProducts.loadProducts(this);

}

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

{

homePageBorder.Visibility = Visibility.Hidden;

homePageImage.Visibility = Visibility.Hidden;

aboutUsBorder.Visibility = Visibility.Hidden;

aboutUsLabelOne.Visibility = Visibility.Hidden;

aboutUsLabelTwo.Visibility = Visibility.Hidden;

aboutUsLabelThree.Visibility = Visibility.Hidden;

aboutUsLabelFour.Visibility = Visibility.Hidden;

aboutUsLabelFive.Visibility = Visibility.Hidden;

aboutUsLabelSix.Visibility = Visibility.Hidden;

aboutUsLabelSeven.Visibility = Visibility.Hidden;

aboutUsLabelEight.Visibility = Visibility.Hidden;

aboutUsLabelNine.Visibility = Visibility.Hidden;

contactsBorder.Visibility = Visibility.Hidden;

contactsLabelOne.Visibility = Visibility.Hidden;

contactsLabelTwo.Visibility = Visibility.Hidden;

contactsLabelThree.Visibility = Visibility.Hidden;

contactsLabelFour.Visibility = Visibility.Hidden;

instImage.Visibility = Visibility.Hidden;

vkImage.Visibility = Visibility.Hidden;

contactsLabelImage.Visibility = Visibility.Hidden;

productsListView.Visibility = Visibility.Hidden;

productsInfoLabel.Visibility = Visibility.Hidden;

costBasketListView.Visibility = Visibility.Hidden;

inBasketDeleteButton.Visibility = Visibility.Hidden;

costLabel.Visibility = Visibility.Hidden;

paymentReceiptButton.Visibility = Visibility.Hidden;

servicesListView.Visibility = Visibility.Visible;

inBasketButton.Visibility = Visibility.Visible;

servicesListView.Items.Clear();

Classes.LoadServices loadServices = new Classes.LoadServices();

loadServices.loadServices(this);

}

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

{

homePageBorder.Visibility = Visibility.Hidden;

homePageImage.Visibility = Visibility.Hidden;

aboutUsBorder.Visibility = Visibility.Hidden;

aboutUsLabelOne.Visibility = Visibility.Hidden;

aboutUsLabelTwo.Visibility = Visibility.Hidden;

aboutUsLabelThree.Visibility = Visibility.Hidden;

aboutUsLabelFour.Visibility = Visibility.Hidden;

aboutUsLabelFive.Visibility = Visibility.Hidden;

aboutUsLabelSix.Visibility = Visibility.Hidden;

aboutUsLabelSeven.Visibility = Visibility.Hidden;

aboutUsLabelEight.Visibility = Visibility.Hidden;

aboutUsLabelNine.Visibility = Visibility.Hidden;

contactsBorder.Visibility = Visibility.Hidden;

contactsLabelOne.Visibility = Visibility.Hidden;

contactsLabelTwo.Visibility = Visibility.Hidden;

contactsLabelThree.Visibility = Visibility.Hidden;

contactsLabelFour.Visibility = Visibility.Hidden;

instImage.Visibility = Visibility.Hidden;

vkImage.Visibility = Visibility.Hidden;

contactsLabelImage.Visibility = Visibility.Hidden;

productsListView.Visibility = Visibility.Hidden;

productsInfoLabel.Visibility = Visibility.Hidden;

servicesListView.Visibility = Visibility.Hidden;

inBasketButton.Visibility = Visibility.Hidden;

costBasketListView.Visibility = Visibility.Visible;

inBasketDeleteButton.Visibility = Visibility.Visible;

costLabel.Visibility = Visibility.Visible;

paymentReceiptButton.Visibility = Visibility.Visible;

}

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

{

Classes.Сalculation сalculation = new Classes.Сalculation();

сalculation.calculationCostServices(this);

}

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

{

Classes.Сalculation сalculation = new Classes.Сalculation();

сalculation.calculationCostDelete(this);

}

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

{

Barcode barcodeWin = new Barcode();

barcodeWin.Owner = SoundStudioWin;

barcodeWin.Show();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

namespace SoundStudio.Classes

{

class Connection

{

public DataTable Select(string selectSQL) // функция подключения к базе данных и обработка запросов

{

DataTable dataTable = new DataTable("dataBase"); // создаём таблицу в приложении

SqlConnection sqlConnection = new SqlConnection("server=DESKTOP-7NIK29D\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=SoundStudio;");

sqlConnection.Open(); // открываем базу данных

SqlCommand sqlCommand = sqlConnection.CreateCommand(); // создаём команду

sqlCommand.CommandText = selectSQL;

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable); // возращаем таблицу с результатом

return dataTable;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

namespace SoundStudio.Classes

{

class LoadProducts

{

public class products

{

public string title { get; set; }

public string description { get; set; }

public string cost { get; set; }

}

public void loadProducts(SoundStudioWindow SSW)

{

Classes.Connection connection = new Classes.Connection();

DataTable dt\_SoundStudioProducts = connection.Select("SELECT \* FROM [dbo].[Table\_products]"); // данные из БД

for (int i = 0; i < dt\_SoundStudioProducts.Rows.Count; i++) // перебираем данные

{

products dataProducts = new products() // создаём экземпляр класса

{

title = dt\_SoundStudioProducts.Rows[i][0].ToString(),

description = dt\_SoundStudioProducts.Rows[i][1].ToString(),

cost = dt\_SoundStudioProducts.Rows[i][2].ToString(),

};

SSW.productsListView.Items.Add(dataProducts); // выводим строку в список

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

namespace SoundStudio.Classes

{

class LoadServices

{

public class services

{

public string titleServices { get; set; }

public string descriptionServices { get; set; }

public string costServices { get; set; }

}

public void loadServices(SoundStudioWindow SSW)

{

Classes.Connection connection = new Classes.Connection();

DataTable dt\_SoundStudioServices = connection.Select("SELECT \* FROM [dbo].[Table\_services]"); // данные из БД

for (int i = 0; i < dt\_SoundStudioServices.Rows.Count; i++) // перебираем данные

{

services dataServices = new services() // создаём экземпляр класса

{

titleServices = dt\_SoundStudioServices.Rows[i][0].ToString(),

descriptionServices = dt\_SoundStudioServices.Rows[i][1].ToString(),

costServices = dt\_SoundStudioServices.Rows[i][2].ToString(),

};

SSW.servicesListView.Items.Add(dataServices); // выводим строку в список

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

namespace SoundStudio.Classes

{

class Сalculation

{

int sum = 0;

public void calculationCostServices(SoundStudioWindow SSW)

{

int iz = 0;

iz = SSW.servicesListView.SelectedIndex;

if (iz >= 0)

{

var selectedItem = (dynamic)SSW.servicesListView.Items[iz];

SSW.costBasketListView.Items.Add(selectedItem);

}

for (int i = 0; i < SSW.costBasketListView.Items.Count; i++)

{

var selectedItem = (dynamic)SSW.costBasketListView.Items[i];

sum += Convert.ToInt32(selectedItem.costServices);

SSW.costLabel.Content = "Итого к оплате: " + sum.ToString();

}

}

public void calculationCostDelete(SoundStudioWindow SSW)

{

if (SSW.costBasketListView.SelectedItem != null)

{ SSW.costBasketListView.Items.RemoveAt(SSW.costBasketListView.Items.IndexOf(SSW.costBasketListView.SelectedItem));

}

for (int i = 0; i < SSW.costBasketListView.Items.Count; i++)

{

var selectedItem = (dynamic)SSW.costBasketListView.Items[i];

sum += Convert.ToInt32(selectedItem.costServices);

SSW.costLabel.Content = "Итого к оплате: " + sum.ToString();

}

if (sum == 0)

{

SSW.costLabel.Content = "Итого к оплате: 0";

}

}

}

}

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Windows.Threading;

using BarcodeLib;

using System.Drawing.Imaging;

namespace SoundStudio.Classes

{

class BarcodeGeneration

{

DateTime date = DateTime.Now;

public void barcodeGeneration(Barcode BW)

{

if (BW.barcodeTextBox.Text.Length == 0 || BW.barcodeTextBoxTwo.Text.Length == 0)

{

MessageBox.Show("Данные не введены!");

}

else

{

string gener = "51" + date.Day + date.Month + date.Year + "38";

BarcodeLib.Barcode b = new BarcodeLib.Barcode();

System.Drawing.Image img = b.Encode(BarcodeLib.TYPE.UPCA, gener, System.Drawing.Color.Black, System.Drawing.Color.White, 290, 120);

MemoryStream ms = new MemoryStream();

img.Save(ms, ImageFormat.Bmp);//save image in memory

byte[] buffer = ms.GetBuffer();

MemoryStream bufferPasser = new MemoryStream(buffer);

BitmapImage bitmap = new BitmapImage();

bitmap.BeginInit();

bitmap.StreamSource = bufferPasser;

bitmap.EndInit();

BW.barcodeImage.Source = bitmap;

}

}

}

}