****

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное**

**учреждение Московской области**

**«Люберецкий техникум имени Героя Советского Союза,**

**лётчика–космонавта Ю.А. Гагарина»**

|  |
| --- |
|  |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**(ДИПЛОМНАЯ РАБОТА)**

На тему: Разработка WPF приложения «Проверка соискателя на соответствие выдвигаемым требованиям»

по специальности: Информационные системы и программирование

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнил |  | Д.В. Карабчиков |  |
| Руководитель |  | Ю.В. Витальевна |  |

ДОПУЩЕН К ЗАЩИТЕ

Заместитель директора по информационным технологиям

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /А.В.Корнеев /

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023г.



**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное**

**учреждение Московской области**

**«Люберецкий техникум имени Героя Советского Союза,**

**лётчика-космонавта Ю.А. Гагарина»**

|  |  |  |
| --- | --- | --- |
| РАССМОТРЕНО  Предметно-цикловой комиссией  Технологического профиля  Протокол № \_\_ от «\_\_» \_\_\_\_\_\_\_\_\_\_\_2023 г.  Председатель МЦК\_\_\_\_\_\_\_\_Ю.В.Жирнова |  | УТВЕРЖДАЮ  Заместитель директора по учебной работе  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.В.Карабчиков  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

**ЗАДАНИЕ НА ДИПЛОМНЫЙ ПРОЕКТ**

Студент: Карабчиков Денис Викторович

Группа № 195-ИС специальность 09.02.07 Информационные системы и программирование

Тема работы: «Разработка WPF приложения «Проверка соискателя на соответствии выдвигаемым требованиям»

Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов по индивидуальной теме):

Введение

1.Глава 1. Проектирование информационной системы

1.1.1 Описание бизнес-процессов

1.1.2 Описание используемого ПО в организации

1.1.3 Анализ рынка существующих программных средств

1.2. Информационная модель

1.2.1 Моделирование информационных процессов

1.2.2. Моделирование логики приложения

2. Глава 2. Реализация информационной системы

2.1 Информационно-логическая модель

2.2. Реализация БД

2.2 Реализация интерфейса

Заключение

Список используемых источников

Руководитель ВКР \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Ж.В.Жирнова/

Задание выдано \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Д.В.Карабчиков/

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023г.

Срок предоставления работы до «10» июня 2023г.

**СОДЕРЖАНИЕ**

[Введение 4](#_Toc8289)

[1. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 6](#_Toc3314)

[1.1. Описание бизнес–процессов 6](#_Toc22817)

[1.1.1. Описание предметной области 8](#_Toc15457)

[1.1.2. Описание используемого ПО в организации 11](#_Toc743)

[1.1.3. Анализ рынка существующих программных средств 13](#_Toc18294)

[1.2. Информационная модель 16](#_Toc13978)

[1.2.1. Моделирование информационных процессов 16](#_Toc25384)

[1.2.2. Моделирование логики приложения 19](#_Toc7738)

[2. РЕАЛИЗАЦИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ 16](#_Toc24256)

[2.1 Информационно–логическая модель 16](#_Toc27692)

[2.2 Реализация БД 23](#_Toc10846)

[2.3 Реализация интерфейса 27](#_Toc10846)

Приложение……………………………………………………………….25

**ВВЕДЕНИЕ**

Дипломная работа была разработана для создания WPF приложения "Проверка соискателя на соответствие выдвигаемым требованиям". Тема имеет наибольшую актуальность из–за большого количества людей, желающих получить профессию в сфере IT. В данной информационной системе очень высокая проходимость данных, как личные данные людей, сдающих тест, так и количества тестов и вопросов в них.

Таким образом, актуальность темы дипломной работы заключается в возможности создать доступную и удобную систему для устройства на работу в сфере IT.

Цель дипломной работы: Разработка WPF приложения «Проверка соискателя на соответствие выдвигаемым требованиям».

Для достижения поставленной цели необходимо решить следующие задачи:

– Проанализировать предметную область;

– Подготовить проект информационной системы (Разработка WPF приложения «Проверка соискателя на соответствие выдвигаемым требованиям»);

– Создать информационную систему согласно проекту и провести ее тестирование.

Дипломная работа состоит из введения, трех основных глав, заключения и списка использованных источников.

Задачами дипломного проекта в связи с указанной целью являются:

1. Анализ предметной области;
2. Проектирование и разработка БД;
3. Проектирование и разработка ПО;
4. Тестирование и отладка ПО;
5. Создание документации ПО;

Методы исследования, использованные в данном дипломном проекте:

* Анализ литературы;
* Анализ документов;
* Обобщение.

**ГЛАВА 1. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ**

**1.1.1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

Предметная область " Проверка соискателя на соответствие выдвигаемым требованиям " включает в себя следующие аспекты:

1. Разработка программного обеспечения для автоматизации процесса проведения тестов.
2. Создание базы данных для хранения информации о каждом тестируемым и его результатах. Это создание электронной базы данных, где будет храниться информация о каждом его результате и других важных параметрах.
3. Разработка системы проверки результатов на соответствие верного решения. Это разработка программного обеспечения, которое будет автоматически проверять тест на соответствие верного решения.
4. Тестируемый – являются люди, которые сдают тесты.

С учетом предметной области была разработана диаграмма use–case, которая отражает функциональные возможности системы и ее взаимодействие с актерами, что позволяет более полно и профессионально описать процессы, связанные с использованием системы.

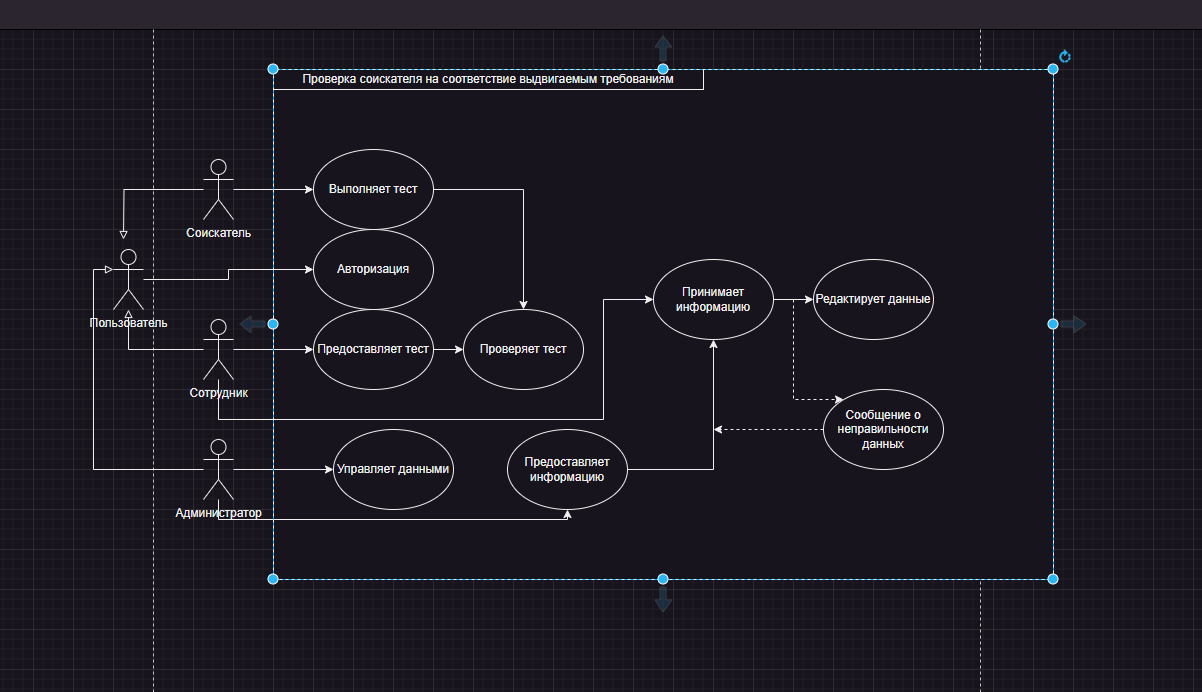


Рисунок 1 – Use case

В автоматизируемой предметной области можно выделить следующие бизнес–процессы.

1. Процесс подачи заявки для получения возможности тестирования. В рамках этого процесса тестируемый подаёт документы, которые требуются для тестов.

2. Процесс входа в систему и создания тестируемого, заполняя информацию о нём и его документах.

3. Процесс проверки документов на соответствие требованиям предприятия. Этот процесс включает в себя проверку корректности данных, заполняемые сотрудниками предприятия. В случае ошибки выведет сообщение о некорректности ведёных данных.

**1.1.2. Описание используемого ПО в организации**

В организации используется бумажный архив для хранения данных о тестируемых, однако данная информационная система по автоматизации проверки соискателя на соответствие выдвигаемым требованиям предлагает более продвинутые и удобные возможности для управления данными. Эта информационная система позволяет автоматизировать процесс зачисления тестируемых. Кроме того, данная информационная система обеспечивает более высокую точность и надежность данных, что может быть особенно важно для бизнес–процессов, связанных с личными данными тестируемых. В результате, разрабатываемая информационная система предлагает более эффективный и удобный способ управления данными, что может привести к повышению производительности и улучшению качества работы предприятия.

**Администратор** будет использовать автоматизированную систему для ряда процессов, связанных с обработкой документов и хранением информации. С помощью ИС сократиться время на ведение учета документов.

Когда экзаменуемый обращается в отдел экзаменов, его данные и документы внесутся в ИС. Система автоматически добавит данные и проверит на наличие ошибок, сформировав приказ, согласно требованиям.

Кроме того, ИС может помочь с контролем за хранением документов и их сроками хранения. Данные, подлежащие уничтожению, будут автоматически удалены из системы. Это упрощает процесс управления документами и обеспечивает соблюдение требований по их хранению и удалению.

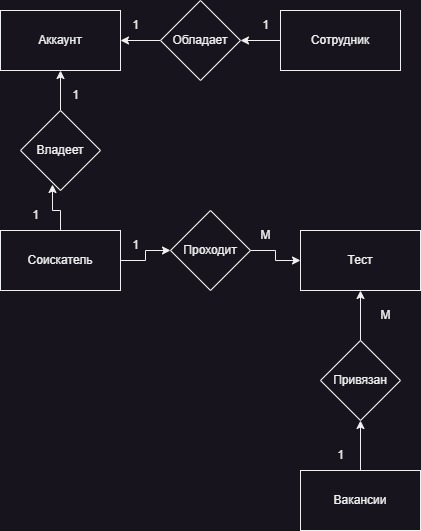


Рисунок. 2 – Диаграмма сущность–связь

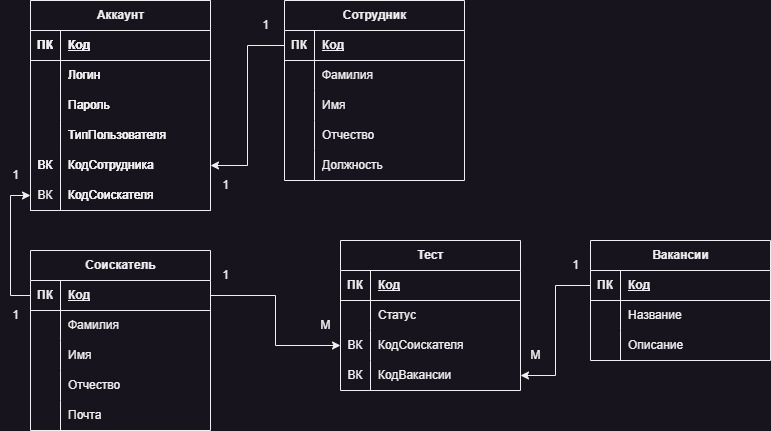


Рисунок 3 – Даталогическая модель

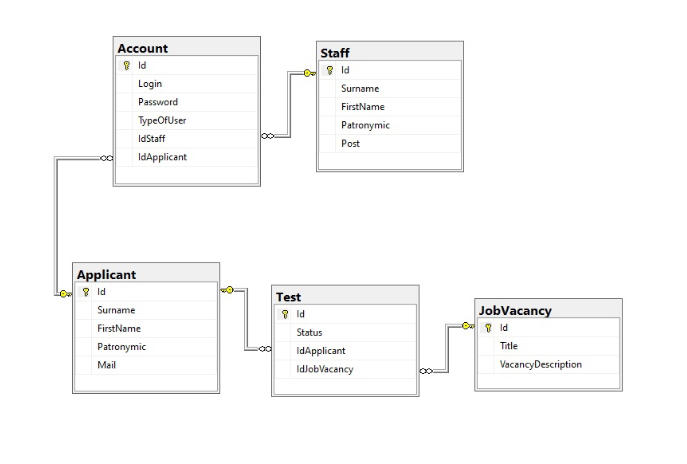


Рисунок 4 – Реляционная модель

**1.2.2. Моделирование логики приложения**

На диаграмме присутствует один актер – Сотрудник. Сотрудник входит в приложение и попадает в навигационное меню. Из навигационного меню можно совершить переход в разделы Тестируемый, Тест. У сотрудника есть варианты исполнения работы – добавить, удалить, изменить. После заполнения данных сохраняем изменения ИС. Система проверяет данные, если они верны, система уведомляет об успешной работе. По окончанию работы система перейдет в главное меню навигации и сотрудник сможет завершить сеанс.

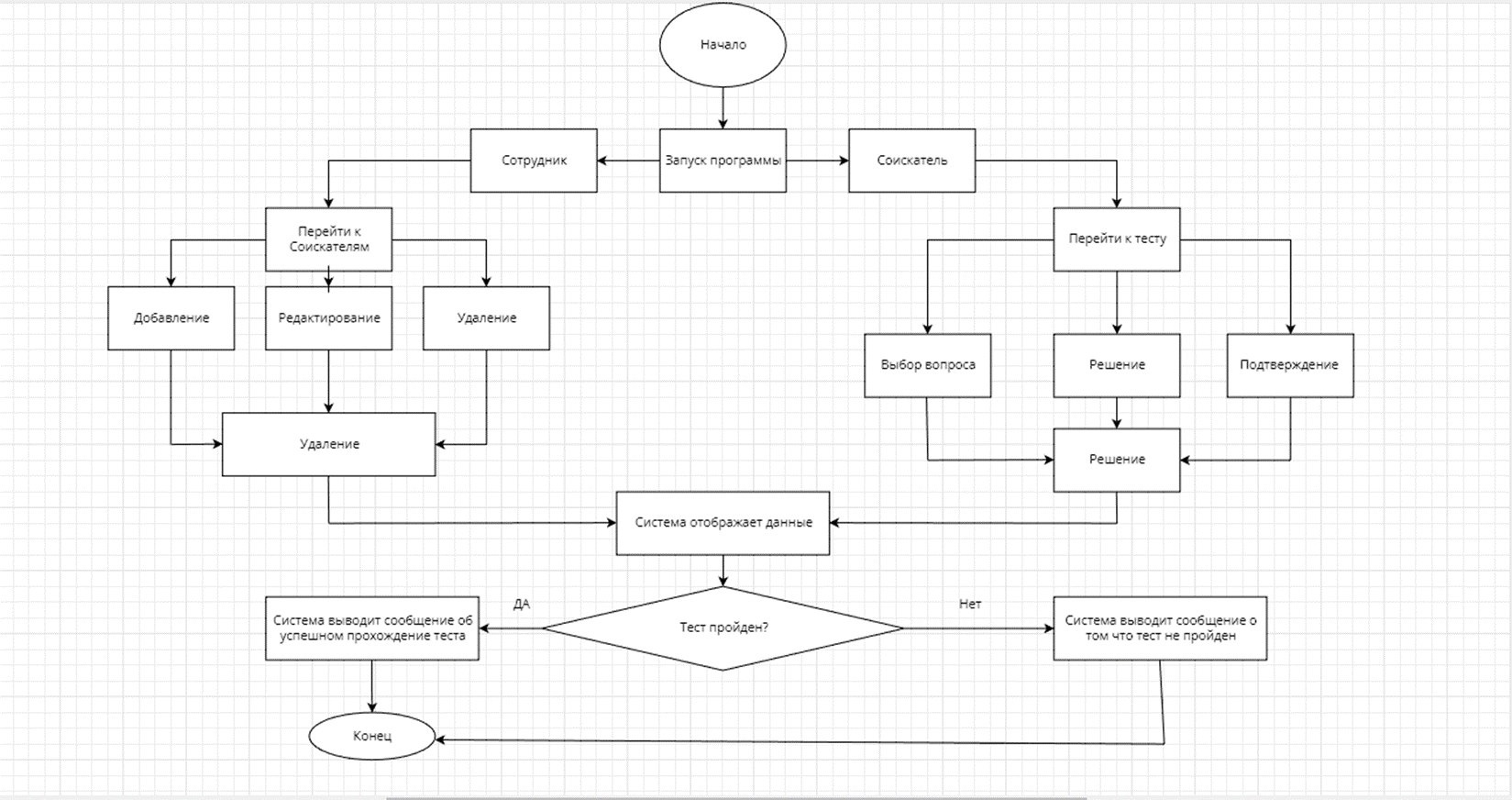


Рисунок 5 – Диаграмма UserFlow

**1.1.3. Анализ рынка существующих программных средств**

HRM Experium — это комплексное программное обеспечение для управления ресурсами человека, которое предназначено для оптимизации и автоматизации процессов управления персоналом в компаниях.

**Функционал:**

Управление кадровыми данными

Учет рабочего времени и оплаты труда

Планирование персонала

Найм и обучение

Оценка производительности

Интеграция

**Преимущества:**

Оптимизация управления персоналом

Улучшение производительности сотрудников

Снижение затрат

Автоматизация процессов

Удобство использования

Безопасность и конфиденциальность данных

Интеграция с другими системами

**Недостатки:**

Сложность внедрения

Высокая стоимость

Ограниченность функционала

Зависимость от интернет-соединения

Необходимсть обучения персонала

Ограничения настройки

Описание

**FriendWork**

FriendWork — сервис для ведения базы резюме, вакансий и заказчиков, в котором удобно работать и анализировать все этапы подбора

FriendWork — сервис для ведения базы резюме, вакансий и заказчиков, в котором удобно работать и анализировать все этапы подбора.

**Особенности FriendWork**

FriendWork автоматизирует рутинные действия на каждом этапе подбора. Вам остается перемещать кандидатов по этапам воронки, которую мы настраиваем под ваш бизнес-процесс, и общаться со всеми участниками подбора в режиме 1 окна. Вся история общения с кандидатами и заказчиками сохраняется. Работайте с любого устройства без установки программ.

**Преимущества FriendWork**

Гибкий поиск. Гибкий поиск резюме по вашей базе в FriendWork по 23 критериям. Можно использовать специальные условия поиска как при Boolean Search.

Мощная аналитика. Умная и удобная аналитика по проделанной работе в виде понятных диаграмм от воронки кандидатов до эффективности источников подбора. Отслеживайте любые данные, FriendWork легко подстроится под ваш запрос.

Кастомизация под клиента. Сервис легко настраивается под ваш бизнес-процесс. При необходимости можно заказать кастомную доработку, которая поможет решить вашу бизнес-задачу.

Возможности FriendWork

Цепочка напоминаний о событиях для рекрутера, заказчика и кандидата.

Инеграция с Telegram и другими мессенджерами.

Кастомные поля в вакансии — настраивается любое поле под заказчика и в него подтягиваются любые необходимые данные.

Кастомные поля в профиле кандидата — настраивается любое поле под заказчика и в него подтягиваются любые необходимые данные.

Строгая последовательность статусов — рекрутер может переводить кандидата по воронке в строго запланированном порядк.

Массовые действия с кандидатами — смена статусов, отправка писем, отправка резюме заказчику.

Kanban-доска для ведения кандидатов.

Когортный анализ – например, сортировка кандидатов в аналитике по дате выхода.

Возможность для Заказчика самому управлять разрешенными статусами.

Встраивание в форму отклика на карьерном портале прохождение теста.

Форма обратной связи для руководителей.

Распределение ролей и прав доступа для разных аккаунтов.

Интеграция с конструктором сайтов Тильда.

API для интеграции с карьерным сайтом заказчика.

Интеграция с BI-системами.

Англоязычная версия сервиса.

**ГЛАВА 2. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ**

## 2.1 Выбор программных средств для реализации ИС

Данная ИС использует системное программное обеспечение, необходимое для её функционирования, т.е. операционную систему Windows. Программа написана на языке высокого уровня в среде программирования Visual Studio C#.

Для создания программы был выбран язык программирования (ЯП) C#.Этот ЯП имеет много плюсов и причин использовать именно его. Вот некоторые из них:

* Приоритетная ориентированность на платформу Windows;
* Типы данных имеют фиксированный размер, что повышает «мобильность» языка и упрощает программирование;
* Большое количество специальных конструкций, разработанных для понимания и написания кода.;
* Наличие большого количества библиотек и шаблонов, позволяющих не тратить время на изобретение своего собственного шаблона;
* Много рабочих мест с требованием знания этого ЯП;
* Низкий порог вхождения. Синтаксис C# имеет много схожего с другими языками программирования. Язык C# часто признают наиболее понятным и подходящим для новичков;

В качестве Интегрированной среды разработки (IDE) была выбрана Visual Studio 2022. Данная IDE имеет ряд плюсов:

* Гибкая и мощная среда разработки;
* подсветка синтаксиса, авто дополнение кода;
* подсветка ошибок;
* статический анализ кода и указание на логические ошибки;
* широкие возможности отладки и профилирования кода;
* средства для работы с базами данных;
* возможность расширения функционала с помощью системы плагинов;
* все необходимые средства для разработки современных Web/Mobile/Desktop приложений и многое другое.

Так же, корпорации Microsoft не безразлично качество своего продукта и поэтому тщательно следит за качеством выпускаемого ПО и таким образом за всю историю своего развития смогла вывести Visual Studio на лидирующие позиции среди сред программирования, используемых в мире. Функциональная структура среды также включает в себя:

* Редактор исходного кода, который включает множество дополнительных функций, как автодополнение IntelliSense, рефакторинг кода и т. д.;
* редактор форм, предназначенный для упрощённого конструирования графических интерфейсов;
* отладчик кода;
* дизайнер классов;
* дизайнер схем баз данных

**2.2 Реализация БД**

На основании задач, определённых ранее, была создана структура данных из 5 таблиц.

Таблица 1 – «Аккаунт»

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Примечание** |
| Код | int | Номер |
| Логин | Nvarhar(50) | Логин пользователя |
| Пароль | Nvarhar(50) | Пароль пользователя |
| Тип пользователя | Nvarhar(50) | Тип пользователя |
| Код сотрудника | Nvarhar(50) | Внешний ключ с таблицей «Сотрудник» |
| Код соискателя | Nvarhar(50) | Внешний ключ с таблицей «Соискателей» |

Таблица 2 – «Сотрудник»

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Примечание** |
| Код | int | Код сотрудника |
| Фамилия | Nvarchar(50) | Фамилия сотрудника |
| Имя | Nvarchar(50) | Имя сотрудника |
| Отчество | Nvarchar(50) | Отчество сотрудника |
| Должность | Nvarchar(50) | Должность сотрудника |

Таблица 3 – «Соискатель»

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Примечание** |
| Код | int | Код соискателя |
| Фамилия | Nvarchar(50) | Фамилия соискателя |
| Имя | Nvarchar(50) | Имя соискателя |
| Отчество | Nvarchar(50) | Отчество соискателя |
| Почта | Nvarchar(50) | Почта соискателя |

Таблица 4 – «Тест»

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Примечание** |
| Код | int | Код теста |
| Статус | Nvarchar(50) | Статус теста |
| Код Соискателя | int | Тест соискателя |
| Код Вакансии | int | Тест вакансии |

Таблица 5 – «Вакансии»

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Примечание** |
| Код | int | Код вакансии |
| Название | Nvarchar(50) | Название вакансии |
| Описание | Nvarchar(max) | Описание вакансии |

## 2.3 Реализация интерфейса

Для реализации интерфейса был выбран шаблон «Приложение WPF». В этом шаблоне при создании интерфейса используется язык XAML с помощью которого легко создать интерфейс программы. В WPF существует элемент «страница» которая позволяет заменять данные в элементе «окно» на своё содержимое.

При запуске программы открывается форма с навигационным меню, на которой пользователю можно перенаправляться с формы на форму.

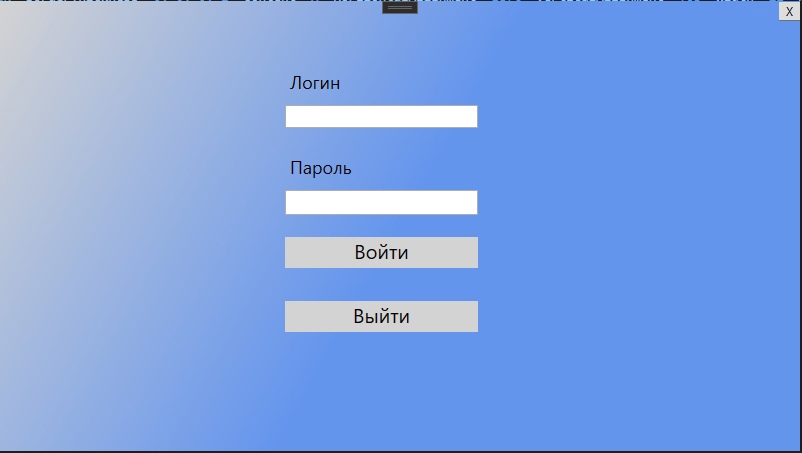


Рисунок 6 «Авторизации»

Во вкладке «Тесты» пользователь может добавлять/изменять и удалять данные.

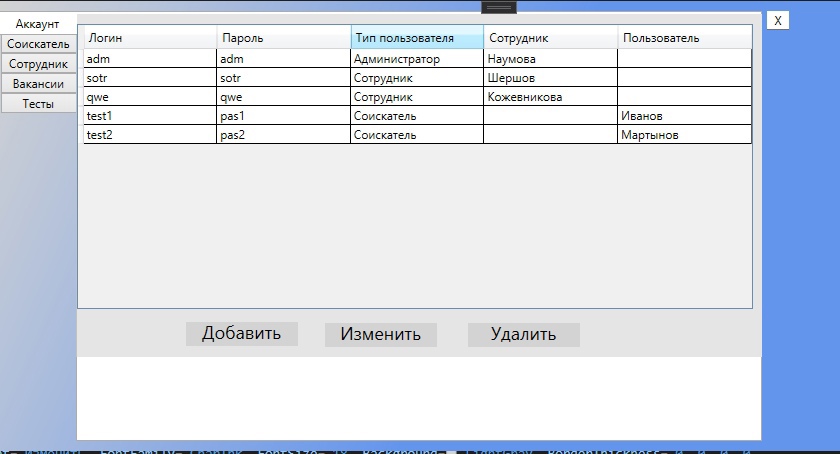


Рисунок 7 «Главное меню»

Нажав на кнопку «Добавить» или двойным нажатием по строке с данными, пользователь перейдет форму изменения/добавления данных.

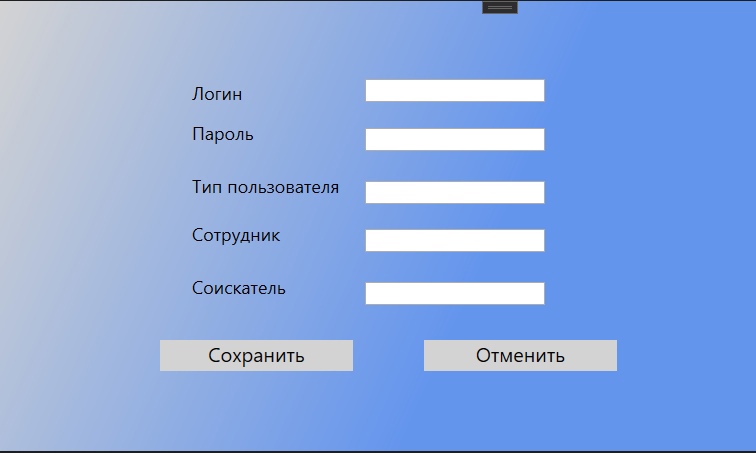


Рисунок 8 «Редактирование»

Если пользователь оставит пустые поля, то приложение выведет ошибку с просьбой заполнить строки.

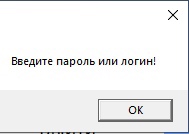


Рисунок 9 «Уведомление об ошибке»

Если вошли правильно в программу, то Вам выдаст окно с выбором профессии, который Вы хотите получить входе выполнения теста

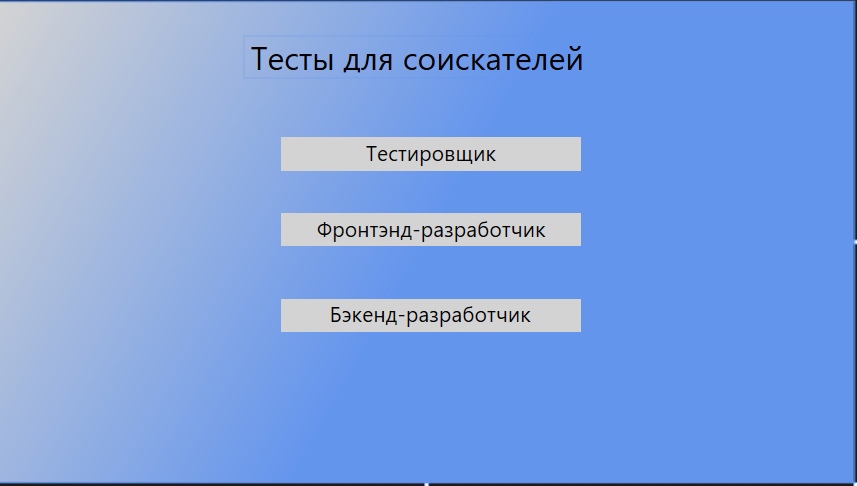


Рисунок 10 «Выбор профессии»

После выбора профессии Вас переключит на тест по желанной профессии.

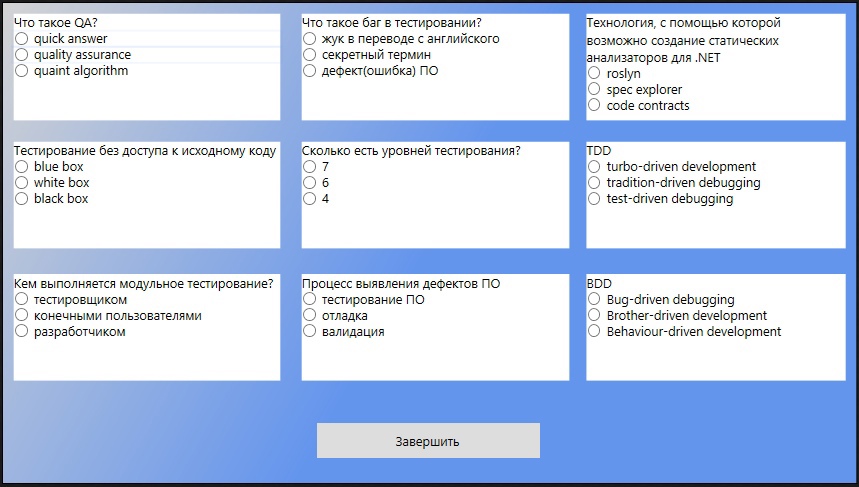


Рисунок 11 «Прохождение теста»

После прохождения теста, Вам выдаст результат положительный или отрицательный, чтобы сдать тест нужно ответить на все вопросы правильно

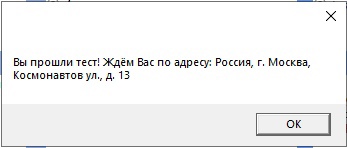


Рисунок 11 «Положительный результат теста»

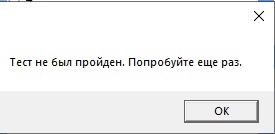


Рисунок 12«Отрицательный результат теста»

Программа была разработана на C#. Приоритетами этой программы были удобствами и легко осваивание во время работы. Хочу немного Вас проинформировать про этот язык программирования, чтоб Вы тоже смогли открыть для себя. Объектно-ориентированный язык программирования общего назначения. Разработан в 1998-2001 группой инженеров компании Microsoft под руководством Андреса Хейлсбрега и Скотта Вильтаумота, как язык разработки приложений для платформы Microsoft.NET Framework и NET core. Впоследствии был стандартизирован как ECMA -334 и ISO/IEC23270.

**ПРИЛОЖЕНИЕ**

класс Worker

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

namespace ApplicantVerification.Classes

{

class Worker

{

public static Frame MainFrame { get; set; }

}

}

КЛАСС Auth

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ApplicantVerification.Classes

{

class Auth

{

public static int rights { get; set; }

}

}

MainWindow.xaml.cs

using ApplicantVerification.Classes;

using ApplicantVerification.Pages;

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;

namespace ApplicantVerification

{

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

MainFrame.Navigate(new Pages.LoginPage());

Worker.MainFrame = MainFrame;

}

}

}

MainWindow.xaml

<Window x:Class="ApplicantVerification.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:ApplicantVerification"

mc:Ignorable="d"

Title="Проверка соискателя на соответствие выдвигаемым требованиям"

Height="450" Width="800"

WindowStartupLocation="CenterScreen"

ResizeMode="NoResize"

WindowStyle="None">

<Grid>

<Frame x:Name="MainFrame" Background="White" NavigationUIVisibility="Hidden"/>

</Grid>

</Window>

LoginPage.xaml

<Page x:Class="ApplicantVerification.Pages.LoginPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:ApplicantVerification.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="LoginPage">

<Border>

<Border.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<GradientStop Color="LightGray" Offset="0"/>

<GradientStop Color="CornflowerBlue" Offset="0.5"/>

</LinearGradientBrush>

</Border.Background>

<Grid>

<Frame x:Name="LoginFrame"/>

<Button x:Name="btnLogin" Content="Войти"

HorizontalAlignment="Left" VerticalAlignment="Top"

Width="193" Margin="285,236,0,0" Height="31"

Click="btnLogin\_Click"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<Button x:Name="btnExit" Content="Выйти"

HorizontalAlignment="Left" VerticalAlignment="Top"

Width="193" Margin="285,300,0,0" Height="31"

Click="btnExit\_Click"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<TextBox x:Name="txtLogin" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="193" Margin="285,104,0,0"/>

<PasswordBox x:Name="txtPassword" HorizontalAlignment="Left" VerticalAlignment="Top" Width="193" Margin="285,189,0,0" Height="25"/>

<Label Content="Логин" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="285,67,0,0"/>

<Label Content="Пароль" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="285,152,0,0"/>

<Button x:Name="btnClose" Click="btnClose\_Click" BorderThickness="0, 0, 0, 1" Content="X" HorizontalAlignment="Left" VerticalAlignment="Top" Width="21" Margin="779,1,0,0"/>

<Label Content="Авторизация" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="285,10,0,0" Width="201" Height="52" FontFamily="CharInk" FontSize="30"/>

</Grid>

</Border>

</Page>

LoginPage.xaml.cs

using ApplicantVerification.Classes;

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 ApplicantVerification.Pages;

namespace ApplicantVerification.Pages

{

public partial class LoginPage : Page

{

public LoginPage()

{

InitializeComponent();

}

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

{

AppVerDBEntities db = new AppVerDBEntities();

if (string.IsNullOrEmpty(txtLogin.Text) || string.IsNullOrWhiteSpace(txtPassword.Password))

{

MessageBox.Show("Введите пароль или логин!");

return;

}

var account = db.Account

.AsNoTracking()

.FirstOrDefault(a => a.Login == txtLogin.Text && a.Password == txtPassword.Password);

Application.Current.MainWindow.Width = 1000;

if (account == null)

{

MessageBox.Show("Пользователь с такими данными не найден!");

return;

}

if (account.TypeOfUser == "Сотрудник" || account.TypeOfUser == "Администратор")

{

MessageBox.Show("Вы зашли как сотрудник");

NavigationService.Navigate(new MainPage());

}

if (account.TypeOfUser == "Соискатель")

{

MessageBox.Show("Вы зашли как пользователь");

NavigationService.Navigate(new TestsPage());

}

}

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

{

MessageBoxResult result = MessageBox.Show("Закрыть приложение?", "Окончание работы.", MessageBoxButton.YesNo);

if (result == MessageBoxResult.Yes)

{

Application.Current.Shutdown();

}

}

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

{

MessageBoxResult result = MessageBox.Show("Закрыть приложение?", "Окончание работы.", MessageBoxButton.YesNo);

if (result == MessageBoxResult.Yes)

{

Application.Current.Shutdown();

}

}

}

}

Experience the Power of Windows 11 OS, Computers, &amp; Apps

Experience the Power of Windows 11 OS, Computers, & Apps

www.microsoft.com

MainPage.xaml

<Page x:Class="ApplicantVerification.Pages.MainPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:ApplicantVerification.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="MainPage" IsVisibleChanged="Page\_IsVisibleChanged">

<Border>

<Border.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<GradientStop Color="LightGray" Offset="0"/>

<GradientStop Color="CornflowerBlue" Offset="0.5"/>

</LinearGradientBrush>

</Border.Background>

<Grid>

<TabControl TabStripPlacement="Left" HorizontalAlignment="Left" Height="430" VerticalAlignment="Top" Width="763" Margin="0,10,0,0">

<TabItem Header="Аккаунт">

<Grid Background="#FFE5E5E5" Margin="-2,0,-4,81">

<DataGrid x:Name="gridAccount" HorizontalAlignment="Left" Height="285" VerticalAlignment="Top" Width="676"

AutoGenerateColumns="False" CanUserAddRows="False" CanUserDeleteRows="False" IsReadOnly="True" Margin="0,10,0,0">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding Id}" Header="Код" Width="\*" Visibility="Hidden"/>

<DataGridTextColumn Binding="{Binding Login}" Header="Логин" Width="\*"/>

<DataGridTextColumn Binding="{Binding Password}" Header="Пароль" Width="\*"/>

<DataGridTextColumn Binding="{Binding TypeOfUser}" Header="Тип пользователя" Width="\*"/>

<DataGridTextColumn Binding="{Binding Staff.Surname}" Header="Сотрудник" Width="\*"/>

<DataGridTextColumn Binding="{Binding Applicant.Surname}" Header="Пользователь" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnAddAcc" Content="Добавить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnAddAcc\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="109,308,0,0"/>

<Button x:Name="btnEditAcc" Content="Изменить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnEditAcc\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="248,309,0,0"/>

<Button x:Name="btnDeleteAcc" Content="Удалить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnDeleteAcc\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="391,309,0,0"/>

</Grid>

</TabItem>

<TabItem Header="Соискатель">

<Grid Background="#FFE5E5E5">

<DataGrid x:Name="gridApplicant" HorizontalAlignment="Left" Height="262" VerticalAlignment="Top" Width="718"

AutoGenerateColumns="False" CanUserAddRows="False" CanUserDeleteRows="False" IsReadOnly="True" Margin="0,0,-11,0">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding Id}" Header="Код" Width="\*" Visibility="Hidden"/>

<DataGridTextColumn Binding="{Binding Surname}" Header="Фамилия" Width="\*"/>

<DataGridTextColumn Binding="{Binding FirstName}" Header="Имя" Width="\*"/>

<DataGridTextColumn Binding="{Binding Patronymic}" Header="Отчество" Width="\*"/>

<DataGridTextColumn Binding="{Binding Mail}" Header="Почта" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnAdd" Content="Добавить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnAdd\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="109,308,0,0"/>

<Button x:Name="btnEdit" Content="Изменить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnEdit\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="248,309,0,0"/>

<Button x:Name="btnDelete" Content="Удалить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnDelete\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="391,309,0,0"/>

</Grid>

</TabItem>

<TabItem Header="Сотрудник">

<Grid

Background="#FFE5E5E5">

<DataGrid x:Name="gridStaff" HorizontalAlignment="Left" Height="263" VerticalAlignment="Top" Width="718"

AutoGenerateColumns="False" CanUserAddRows="False" CanUserDeleteRows="False" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding Id}" Header="Код" Width="\*" Visibility="Hidden"/>

<DataGridTextColumn Binding="{Binding Surname}" Header="Фамилия" Width="\*"/>

<DataGridTextColumn Binding="{Binding FirstName}" Header="Имя" Width="\*"/>

<DataGridTextColumn Binding="{Binding Patronymic}" Header="Отчество" Width="\*"/>

<DataGridTextColumn Binding="{Binding Post}" Header="Должность" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnAddStaff" Content="Добавить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnAddStaff\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="109,308,0,0"/>

<Button x:Name="btnEditStaff" Content="Изменить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnEditStaff\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="248,309,0,0"/>

<Button x:Name="btnDeleteStaff" Content="Удалить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnDeleteStaff\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="391,309,0,0"/>

</Grid>

</TabItem>

<TabItem Header="Вакансии">

<Grid Background="#FFE5E5E5">

<DataGrid x:Name="gridJobVacancy" HorizontalAlignment="Left" Height="263" VerticalAlignment="Top" Width="718"

AutoGenerateColumns="False" CanUserAddRows="False" CanUserDeleteRows="False" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding Id}" Header="Код" Width="\*" Visibility="Hidden"/>

<DataGridTextColumn Binding="{Binding Title}" Header="Название" Width="\*"/>

<DataGridTextColumn Binding="{Binding VacancyDescription}" Header="Описание" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button Visibility="Hidden" x:Name="btnAddVacancy" Content="Добавить" Click="btnAddVacancy\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Margin="166,318,0,0"/>

<Button Visibility="Hidden" x:Name="btnEditVacancy" Content="Изменить" Click="btnEditVacancy\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Margin="283,318,0,0"/>

<Button Visibility="Hidden" x:Name="btnDeleteVacancy" Content="Удалить" Click="btnDeleteVacancy\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Margin="407,318,0,0"/>

</Grid>

</TabItem>

<TabItem Header="Тесты">

<Grid Background="#FFE5E5E5">

<DataGrid x:Name="gridTests" HorizontalAlignment="Left" Height="263" VerticalAlignment="Top" Width="718"

AutoGenerateColumns="False" CanUserAddRows="False" CanUserDeleteRows="False" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Binding="{Binding Id}" Header="Код" Width="\*" Visibility="Hidden"/>

<DataGridTextColumn Binding="{Binding Status}" Header="Статус" Width="\*"/>

<DataGridTextColumn Binding="{Binding Applicant.Surname}" Header="Соискатель" Width="\*"/>

<DataGridTextColumn Binding="{Binding JobVacancy.Title}" Header="Вакансия" Width="\*"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnAddTest" Content="Добавить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnAddTest\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="109,308,0,0"/>

<Button x:Name="btnEditTest" Content="Изменить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnEditTest\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="248,309,0,0"/>

<Button x:Name="btnDeleteTest" Content="Удалить" FontFamily="CharInk" FontSize="18" Background="LightGray" BorderThickness="0, 0, 0, 0"

Click="btnDeleteTest\_Click" HorizontalAlignment="Left" VerticalAlignment="Top" Width="112" Margin="391,309,0,0"/>

</Grid>

</TabItem>

</TabControl>

<Button x:Name="btnClose" Click="btnClose\_Click" Background="White" BorderThickness="0, 0, 0, 1" Content="X" HorizontalAlignment="Left" VerticalAlignment="Top" Width="22" Margin="768,10,0,0"/>

</Grid>

</Border>

</Page>

MainPage.xaml.cs

using ApplicantVerification.Classes;

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;

namespace ApplicantVerification.Pages

{

public partial class MainPage : Page

{

AppVerDBEntities db = new AppVerDBEntities();

public MainPage()

{

InitializeComponent();

}

private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

if (Visibility == Visibility.Visible)

{

AppVerDBEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());

gridAccount.ItemsSource = AppVerDBEntities.GetContext().Account.ToList();

gridApplicant.ItemsSource = AppVerDBEntities.GetContext().Applicant.ToList();

gridStaff.ItemsSource = AppVerDBEntities.GetContext().Staff.ToList();

gridJobVacancy.ItemsSource = AppVerDBEntities.GetContext().JobVacancy.ToList();

gridTests.ItemsSource = AppVerDBEntities.GetContext().Test.ToList();

}

}

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

{

AppVerDBEntities change = new AppVerDBEntities();

Worker.MainFrame.Navigate(new EditApplicantPage(null));

}

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

{

var selectedItem = gridApplicant.SelectedItem as Applicant;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для изменения данных!");

return;

}

else

{

int empId = (gridApplicant.SelectedItem as Applicant).Id;

Applicant applicant = (from r in db.Applicant where r.Id == empId select r).SingleOrDefault();

Worker.MainFrame.Navigate(new EditApplicantPage(gridApplicant.SelectedItem as Applicant));

}

}

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

{

var selectedItem = gridApplicant.SelectedItem as Applicant;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для удаления данных!");

return;

}

else

{

var clientForRemoving = gridApplicant.SelectedItems.Cast<Applicant>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить элемент?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

int empId = (gridApplicant.SelectedItem as Applicant).Id;

Applicant applicant = (from r in db.Applicant where r.Id == empId select r).SingleOrDefault();

db.Applicant.Remove(applicant);

db.SaveChanges();

MessageBox.Show("Данные удалены!");

gridApplicant.ItemsSource = AppVerDBEntities.GetContext().Applicant.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

}

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

{

AppVerDBEntities change = new AppVerDBEntities();

Worker.MainFrame.Navigate(new EditTestPage(null));

}

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

{

var selectedItem = gridTests.SelectedItem as Test;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для изменения данных!");

return;

}

else

{

int empId = (gridTests.SelectedItem as Test).Id;

Test test = (from r in db.Test where r.Id == empId select r).SingleOrDefault();

Worker.MainFrame.Navigate(new EditTestPage(gridTests.SelectedItem as Test));

}

}

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

{

var selectedItem = gridTests.SelectedItem as Test;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для удаления данных!");

return;

}

else

{

var clientForRemoving = gridTests.SelectedItems.Cast<Test>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить элемент?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

int empId = (gridTests.SelectedItem as Test).Id;

Test test = (from r in db.Test where r.Id == empId select

Experience the Power of Windows 11 OS, Computers, &amp; Apps

Experience the Power of Windows 11 OS, Computers, & Apps

www.microsoft.com

r).SingleOrDefault();

db.Test.Remove(test);

db.SaveChanges();

MessageBox.Show("Данные удалены!");

gridTests.ItemsSource = AppVerDBEntities.GetContext().Test.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

}

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

{

/\*AppVerDBEntities change = new AppVerDBEntities();

Worker.MainFrame.Navigate(new EditVacancyPage(null));\*/

}

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

{

/\*var selectedItem = gridJobVacancy.SelectedItem as JobVacancy;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для изменения данных!");

return;

}

else

{

int empId = (gridJobVacancy.SelectedItem as JobVacancy).Id;

JobVacancy job = (from r in db.JobVacancy where r.Id == empId select r).SingleOrDefault();

Worker.MainFrame.Navigate(new EditVacancyPage(gridJobVacancy.SelectedItem as JobVacancy));

}\*/

}

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

{

/\*var selectedItem = gridJobVacancy.SelectedItem as JobVacancy;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для удаления данных!");

return;

}

else

{

var clientForRemoving = gridJobVacancy.SelectedItems.Cast<JobVacancy>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить элемент?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

int empId = (gridJobVacancy.SelectedItem as JobVacancy).Id;

JobVacancy job = (from r in db.JobVacancy where r.Id == empId select r).SingleOrDefault();

db.JobVacancy.Remove(job);

db.SaveChanges();

MessageBox.Show("Данные удалены!");

gridJobVacancy.ItemsSource = AppVerDBEntities.GetContext().JobVacancy.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}\*/

}

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

{

AppVerDBEntities change = new AppVerDBEntities();

Worker.MainFrame.Navigate(new EditStaffPage(null));

}

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

{

var selectedItem = gridStaff.SelectedItem as Staff;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для изменения данных!");

return;

}

else

{

int empId = (gridStaff.SelectedItem as Staff).Id;

Staff staff = (from r in db.Staff where r.Id == empId select r).SingleOrDefault();

Worker.MainFrame.Navigate(new EditStaffPage(gridStaff.SelectedItem as Staff));

}

}

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

{

var selectedItem = gridStaff.SelectedItem as Staff;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для удаления данных!");

return;

}

else

{

var clientForRemoving = gridStaff.SelectedItems.Cast<Staff>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить элемент?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

int empId = (gridStaff.SelectedItem as Staff).Id;

Staff staff = (from r in db.Staff where r.Id == empId select r).SingleOrDefault();

db.Staff.Remove(staff);

db.SaveChanges();

MessageBox.Show("Данные удалены!");

gridStaff.ItemsSource = AppVerDBEntities.GetContext().Staff.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

}

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

{

AppVerDBEntities change = new AppVerDBEntities();

Worker.MainFrame.Navigate(new EditAccPage(null));

}

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

{

var selectedItem = gridAccount.SelectedItem as Account;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для изменения данных!");

return;

}

else

{

int empId = (gridAccount.SelectedItem as Account).Id;

Account account = (from r in db.Account where r.Id == empId select r).SingleOrDefault();

Worker.MainFrame.Navigate(new EditAccPage(gridAccount.SelectedItem as Account));

}

}

private void

btnDeleteAcc\_Click(object sender, RoutedEventArgs e)

{

var selectedItem = gridAccount.SelectedItem as Account;

if (selectedItem == null)

{

MessageBox.Show("Выберите строку для удаления данных!");

return;

}

else

{

var clientForRemoving = gridAccount.SelectedItems.Cast<Account>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить элемент?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

int empId = (gridAccount.SelectedItem as Account).Id;

Account account = (from r in db.Account where r.Id == empId select r).SingleOrDefault();

db.Account.Remove(account);

db.SaveChanges();

MessageBox.Show("Данные удалены!");

gridAccount.ItemsSource = AppVerDBEntities.GetContext().Account.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

}

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

{

MessageBoxResult result = MessageBox.Show("Закрыть приложение?", "Окончание работы.", MessageBoxButton.YesNo);

if (result == MessageBoxResult.Yes)

{

Application.Current.Shutdown();

}

}

}

}

EditAccPage.xaml

<Page x:Class="ApplicantVerification.Pages.EditAccPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:ApplicantVerification.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="EditAccPage">

<Border>

<Border.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<GradientStop Color="LightGray" Offset="0"/>

<GradientStop Color="CornflowerBlue" Offset="0.5"/>

</LinearGradientBrush>

</Border.Background>

<Grid>

<Button x:Name="btnSave" Click="btnSave\_Click" Content="Сохранить"

HorizontalAlignment="Left" VerticalAlignment="Top" Width="193" Margin="160,339,0,0" Height="31"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<Button x:Name="btnCancel" Click="btnCancel\_Click" Content="Отменить"

HorizontalAlignment="Left" VerticalAlignment="Top" Width="193" Margin="424,339,0,0" Height="31"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<TextBox Text="{Binding Login}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="365,78,0,0"/>

<TextBox Text="{Binding Password}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="365,127,0,0"/>

<TextBox Text="{Binding TypeOfUser}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="365,180,0,0"/>

<TextBox Text="{Binding IdStaff}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="365,228,0,0"/>

<TextBox Text="{Binding IdApplicant}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="365,281,0,0"/>

<Label Content="Логин" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,78,0,0"/>

<Label Content="Пароль" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,118,0,0"/>

<Label Content="Тип пользователя" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,171,0,0"/>

<Label Content="Код сотрудника" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,219,0,0"/>

<Label Content="Код соискателя" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,272,0,0"/>

<Label Content="Редактирование аккаунтов" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,10,0,0" Width="396" Height="51" FontFamily="CharInk" FontSize="30"/>

</Grid>

</Border>

</Page>

EditAccPage.xaml.cs

using ApplicantVerification.Classes;

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;

namespace ApplicantVerification.Pages

{

public partial class EditAccPage : Page

{

private Account \_acc = new Account();

public EditAccPage(Account selectedAcc)

{

InitializeComponent();

if (selectedAcc != null)

\_acc = selectedAcc;

DataContext = \_acc;

AppVerDBEntities db = new AppVerDBEntities();

}

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

{

StringBuilder errors = new StringBuilder();

if (string.IsNullOrWhiteSpace(\_acc.Login))

errors.AppendLine("Укажите фамилию соискателя");

if (string.IsNullOrWhiteSpace(\_acc.Password))

errors.AppendLine("Укажите имя соискателя");

if (string.IsNullOrWhiteSpace(\_acc.TypeOfUser))

errors.AppendLine("Укажите отчество соискателя");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

errors.Clear();

return;

}

if (\_acc.Id == 0)

AppVerDBEntities.GetContext().Account.Add(\_acc);

AppVerDBEntities.GetContext().SaveChanges();

MessageBox.Show("Информация сохранена!");

Worker.MainFrame.Navigate(new MainPage());

}

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

{

Worker.MainFrame.Navigate(new MainPage());

}

}

}

Experience the Power of Windows 11 OS, Computers, &amp; Apps

Experience the Power of Windows 11 OS, Computers, & Apps

www.microsoft.com

EditApplicantPage.xaml

<Page x:Class="ApplicantVerification.Pages.EditApplicantPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:ApplicantVerification.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="EditApplicantPage">

<Border>

<Border.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<GradientStop Color="LightGray" Offset="0"/>

<GradientStop Color="CornflowerBlue" Offset="0.5"/>

</LinearGradientBrush>

</Border.Background>

<Grid>

<Button x:Name="btnSave" Click="btnSave\_Click" Content="Сохранить"

HorizontalAlignment="Left" VerticalAlignment="Top" Width="193" Margin="160,339,0,0" Height="31"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<Button x:Name="btnCancel" Click="btnCancel\_Click" Content="Отменить"

HorizontalAlignment="Left" VerticalAlignment="Top" Width="193" Margin="424,339,0,0" Height="31"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<TextBox Text="{Binding Surname}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="390,78,0,0"/>

<TextBox Text="{Binding FirstName}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="390,127,0,0"/>

<TextBox Text="{Binding Patronymic}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="390,180,0,0"/>

<TextBox Text="{Binding Mail}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="390,228,0,0"/>

<Label Content="Фамилия соискателя" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,78,0,0"/>

<Label Content="Имя соискателя" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,118,0,0"/>

<Label Content="Отчество соискателя" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,171,0,0"/>

<Label Content="Почта" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,219,0,0"/>

<Label Content="Редактирование соискателей" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,10,0,0" Width="435" Height="39" FontFamily="CharInk" FontSize="30"/>

</Grid>

</Border>

</Page>

EditApplicantPage.xaml.cs

using ApplicantVerification.Classes;

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;

namespace ApplicantVerification.Pages

{

public partial class EditApplicantPage : Page

{

private Applicant \_app = new Applicant();

public EditApplicantPage(Applicant selectedApplicant)

{

InitializeComponent();

if (selectedApplicant != null)

\_app = selectedApplicant;

DataContext = \_app;

AppVerDBEntities db = new AppVerDBEntities();

}

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

{

StringBuilder errors = new StringBuilder();

if (string.IsNullOrWhiteSpace(\_app.Surname))

errors.AppendLine("Укажите фамилию соискателя");

if (string.IsNullOrWhiteSpace(\_app.FirstName))

errors.AppendLine("Укажите имя соискателя");

if (string.IsNullOrWhiteSpace(\_app.Patronymic))

errors.AppendLine("Укажите отчество соискателя");

if (string.IsNullOrWhiteSpace(\_app.Mail))

errors.AppendLine("Укажите почту соискателя");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

errors.Clear();

return;

}

if (\_app.Id == 0)

AppVerDBEntities.GetContext().Applicant.Add(\_app);

AppVerDBEntities.GetContext().SaveChanges();

MessageBox.Show("Информация сохранена!");

Worker.MainFrame.Navigate(new MainPage());

}

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

{

Worker.MainFrame.Navigate(new MainPage());

}

}

}

Experience the Power of Windows 11 OS, Computers, &amp; Apps

Experience the Power of Windows 11 OS, Computers, & Apps

www.microsoft.com

EditStaffPage.xaml

<Page x:Class="ApplicantVerification.Pages.EditStaffPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:ApplicantVerification.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="EditStaffPage">

<Border>

<Border.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<GradientStop Color="LightGray" Offset="0"/>

<GradientStop Color="CornflowerBlue" Offset="0.5"/>

</LinearGradientBrush>

</Border.Background>

<Grid>

<Button x:Name="btnSave" Click="btnSave\_Click" Content="Сохранить"

HorizontalAlignment="Left" VerticalAlignment="Top" Width="193" Margin="160,339,0,0" Height="31"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<Button x:Name="btnCancel" Click="btnCancel\_Click" Content="Отменить"

HorizontalAlignment="Left" VerticalAlignment="Top" Width="193" Margin="424,339,0,0" Height="31"

FontFamily="CharInk" FontSize="20" Background="LightGray" BorderThickness="0, 0, 0, 0"/>

<TextBox Text="{Binding Surname}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="424,78,0,0"/>

<TextBox Text="{Binding FirstName}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="424,127,0,0"/>

<TextBox Text="{Binding Patronymic}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="424,180,0,0"/>

<TextBox Text="{Binding Post}" HorizontalAlignment="Left" Height="23" TextWrapping="Wrap" VerticalAlignment="Top" Width="180" Margin="424,228,0,0"/>

<Label Content="Фамилия сотрудника" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,78,0,0"/>

<Label Content="Имя сотрудника" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,118,0,0"/>

<Label Content="Отчество сотрудника" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,171,0,0"/>

<Label Content="Должность" FontFamily="CharInk" FontSize="18" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="187,219,0,0"/>

<Label Content="Редактирование сотрудников" HorizontalAlignment="Left" VerticalAlignment="Top" Margin="10,10,0,0" Width="430" Height="47" FontFamily="CharInk" FontSize="30"/>

</Grid>

</Border>

</Page>

EditStaffPage.xaml.cs

using ApplicantVerification.Classes;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

**ЗАКЛЮЧЕНИЕ**

В результате дипломной работы спроектирована информационная система «Тест проверка соискателя на соответствии выдвигаемым требованиям» для хранения информации о Тестах, готовая к внедрению.

В ходе подготовки дипломной работы решены следующие задачи:

– проанализирована автоматизируемая предметная область «Тест проверка соискателя на соответствии выдвигаемым требованиям», построена модель ее бизнес–процессов;

– сформулированы функциональные требования к ИС;

– разработан проект ИС, включающий информационно–логическую и даталогическую модели;

– создан программный продукт – информационная система «Тест проверка соискателя на соответствии выдвигаемым требованиям», проведено ее тестирование.

По итогам дипломной работы можно сделать следующие выводы:

1) Проектирование информационной системы «Тест проверка соискателя на соответствии выдвигаемым требованиям» является задачей, для которой можно предложить автоматизированное решение, учитывающее требования организации–заказчика;

2) информационная система «Тест проверка соискателя на соответствии выдвигаемым требованиям» соответствует всем заявленным функциональным требованиям, корректно формирует выходную информацию и может быть использована сотрудниками медицинского центра анализов.

**Список использованных источников**

Список литературы

1. Грин Дженнифер и Стиллмен Эндрю «Head First. Изучаем C#. 4–е изд.» [Книга]

Электронные ресурсы

1. Онлайн–сервис Diagrams.net [Электронный ресурс] – Режим доступа: <https://app.diagrams.net/>.
2. Автоматизация бизнеса: начинаем разбираться [Электронный ресурс] – Режим доступа: <https://habr.com/ru/company/regionsoft/blog/520242/>.
3. Автоматизация процессов: кому нужна, кто её проводит и какие системы для неё использовать [Электронный ресурс] – Режим доступа: [https://skillbox.ru/media/management/avtomatizatsiya–protsessov–komu–nuzhna–kto–eye–provodit–i–kakie–sistemy–dlya–neye–ispolzovat/](https://skillbox.ru/media/management/avtomatizatsiya-protsessov-komu-nuzhna-kto-eye-provodit-i-kakie-sistemy-dlya-neye-ispolzovat/).