Государственное бюджетное профессиональное образовательное учреждение

«Пермский техникум промышленных и информационных

технологий им. Б.Г. Изгагина»

**ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА**

**ИНФОРМАЦИОННЫХ СИСТЕМ**

Разработка информационной системы «Музыкальная картотека»

по МДК.05.02 Разработка кода информационных систем

МДК.05.03 Тестирование информационных систем

МДК.05.04 Программные решения для бизнеса

09.02.07.21.3.1.21

Разработал студент Нефедов А. В.

Проверил преподаватель Филимонов А. Ю.

Пермь, 2023

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ…………………………………………………………….3

РАЗДЕЛ 1. ПРОЕКТИРОВАНИЕ……………………………………4

РАЗДЕЛ 2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ…………………………….5

РАЗДЕЛ 3. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ...6

РАЗДЕЛ 4. UNIT ТЕСТИРОВАНИЕ………………………………11

ЗАКЛЮЧЕНИЕ………………………………………………………13

**ВВЕДЕНИЕ**

На данный момент существует множество программ для совершенно разных нужд. Разнообразие в выборе настолько большое что даже программы предназначенные для одни и тех же целей постоянно появляются улучшаются и борются за внимание пользователя. Продвинутые пользователи ПК зачастую используют пользовательские вариации даже для стандартных приложений какой-либо операционной системы. Например, многие пользователи ОС Windows находят стандартные медиапроигрыватели, или обозреватели файлов хуже, чем альтернативные решения от разработчиков по самым разным причинам. Кому-то может не нравится конкретно внешний вид и интерфейс, а кто-то может негодовать от предоставляемого функционала. В нашем проекте представлена информационная система музыкальной картотеки, представляющая собой обозреватель файлов с функциями менеджера.

**РАЗДЕЛ 1**. **ПРОЕКТИРОВАНИЕ**

Используемая среда разработки – Visual Studio, качестве языка программирования будет использован язык WPF (Windows Presentation Form), который содержит инструменты для размещения объектов в окне с помощью специального интерфейса, он упрощает процесс создания программы. Также в проекте будет реализована связь приложения с базой данной, поэтому также необходимо выбрать СУБД. В качестве используемой СУБД будет использована Microsoft SQL Server Management.



Рисунок 1. Use-Case Диаграмма

**РАЗДЕЛ 2.** **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**Цель проекта**:

Создание информационной системы по теме «Музыкальная картотека»

**Задачи проекта**:

1. Разработать приложение с возможностью авторизации и регистрации
2. Создать базу данных для подключения к приложению с помощью
3. Связать базу данных с разработанным приложением

Приложение представляет себя обозреватель данных, предназначенный для музыкальных записей, в нём можно будет записывать данные о композициях и просматривать их, эти данные будут сохраняться. Для реализации подобного функционала (авторизации, регистрации и сохранения данных) необходимо изучить средства работы с базами данных.

В проекте будет реализован изменяющийся интерфейс в зависимости от текущего пользователя. Если пользователь имеет свой профиль, то ему будет доступен интерфейс со всеми функциями приложения, иначе, если пользователь не зарегистрирован, он может войти в приложение как гость, в таком режиме будет доступен обзор на функции приложения, но воспользоваться предоставляемыми функциями в таком режиме нельзя.

**РАЗДЕЛ 3. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

Разработка начинается с того, что нужно создать шаблон на WPF, а конкретно страницу авторизации, она должна иметь в себе поля ввода, кнопки входа в приложение, а также кнопку регистрации, если пользователь не имеет профиля для входа. По итогу получается страница авторизации со следующими элементами:

* Поле ввода (логин)
* Поле ввода (пароль)
* Кнопка входа (авторизация)
* Кнопка входа (войти как гость)
* Кнопка регистрации (перейти к окну регистрации)

Так выглядят элементы расположенные на странице авторизации (рис. 2)

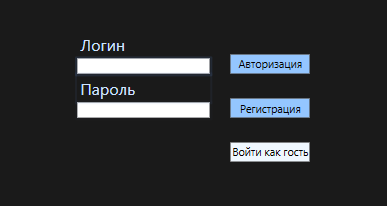


Рисунок 2. Элементы на странице авторизации

Кроме страницы авторизации, нужно подготовить несколько страниц которые будут отображаться в окне, в них входят:

* Страница с интерфейсом авторизованного пользователя
* Страница с интерфейсом пользователя-гостя
* Страница с регистрацией пользователя

На странице с интерфейсом авторизованного пользователя будет располагаться основные функции приложения, можно будет добавлять композиции в формате: исполнитель, название, также будет возможность просматривать записанные композиции (рис. 3).

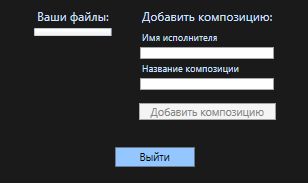


Рисунок 3. Элементы на странице авторизованного пользователя

Просмотр файлов на странице авторизованного пользователя реализован с помощью элемента DataGrid который представляет собой элемент таблицы, данные в таблице будет импортированы из базы данных, в базе данных, в свою очередь, будет находится таблица, которая отдаёт данные в DataGrid. Код, который выполняет присоединение DataGrid к базе данных можно увидеть в листинге 1

Листинг 1 –Привязка данных DataGrid (dg1) к базе данных (MUEntities1)

|  |
| --- |
| public MainPage()  {InitializeComponent();  var q = from song in dataent.Song select new {song.Name, song.Author };  dg1.ItemsSource = q.ToList();} |

Страница с интерфейсом гостя представляет собой такой же как у авторизованного пользователя только без доступа к предоставленным функциям и подсказкой об авторизации для получения доступа.

Страница с регистрацией пользователяпредставляет собой модифицированную страницу авторизации, только с двумя полями ввода для пароля, где второе окно создано в целях подтверждения ранее введённого пароля (рис. 4)

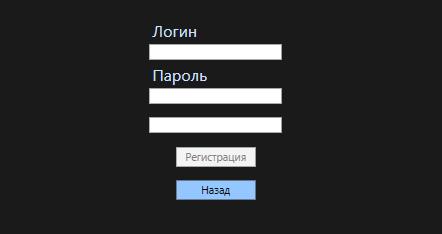


Рисунок 4. Страница регистрации пользователя

Когда внешний вид приложения готов осталось привязать функции кнопок и полей ввода с разных страниц к базам данных.

В базе данных будет таблица User, которая будет содержать данные о пользователях, их логины и пароли она будет привязана к странице авторизации в которой данные будут проверятся на достоверность, если пользователь существует то авторизация будет пройдена и доступ к приложению будет получен, если же проверка не пройдётся то будет показано сообщение либо об ошибке либо об неправильности/отсутствии введённых данных. Код, который будет проверять правильность введённых данных отображён в листинге 1.2

Листинг 2 – Проверка введённых данных на достоверность

|  |
| --- |
| private void log(object sender, RoutedEventArgs e)  {  try  {  var userObj = AppConnection.model0db.MUUser. FirstOrDefault (x => x. Login == alog. Text && x. Password == apass.Password);  if (userObj == null)  {  MessageBox.Show("Такого пользователя нет", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);  }  else  {  switch (userObj.Role)  {  case 1: |

Листинг 2.1 – Проверка введённых данных на достоверность

|  |
| --- |
| MessageBox.Show("Вы авторизованы!" , "Уведомление", MessageBoxButton.OK, MessageBoxImage.Information);  AppFrame.frameMain. Navigate (new authpage. MainPage ());  break;  default:  MessageBox.Show("Данные не обнаружены", "Уведомление", MessageBoxButton.OK, MessageBoxImage.Warning);  break;}}}  catch (Exception Ex) {  MessageBox.Show("Ошибка" + Ex. Message.ToString() + "Критическая работа приложения!", "Уведомление", MessageBoxButton.OK, MessageBoxImage.Warning);}} |

После того как были сделаны все основные функции приложения и привязка к базе данных, можно приступать к внешнему виду приложения. Интерфейс будет выполнен в тёмных оттенках с использованием светлых шрифтов, в верхней части окна будет отображено название текущей страницы приложения, а в нижней текущая страница. После использования стилей, конечный внешний вид приложения выглядит так (рис. 5,6)

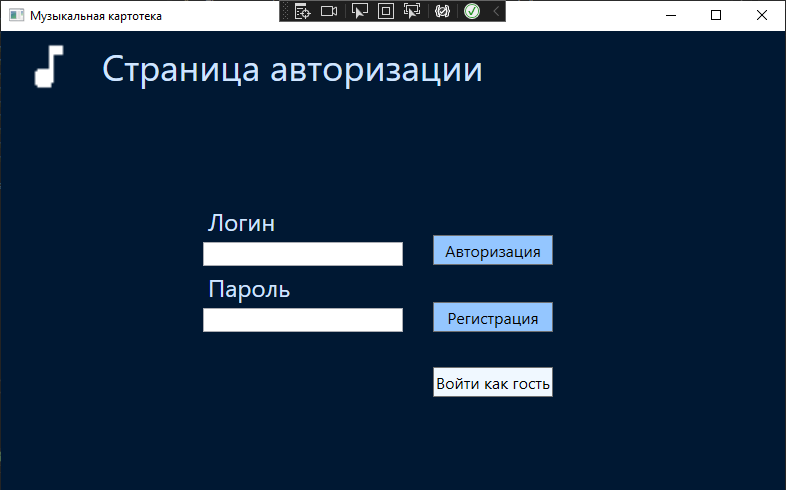


Рисунок 5. Конечная страница авторизации

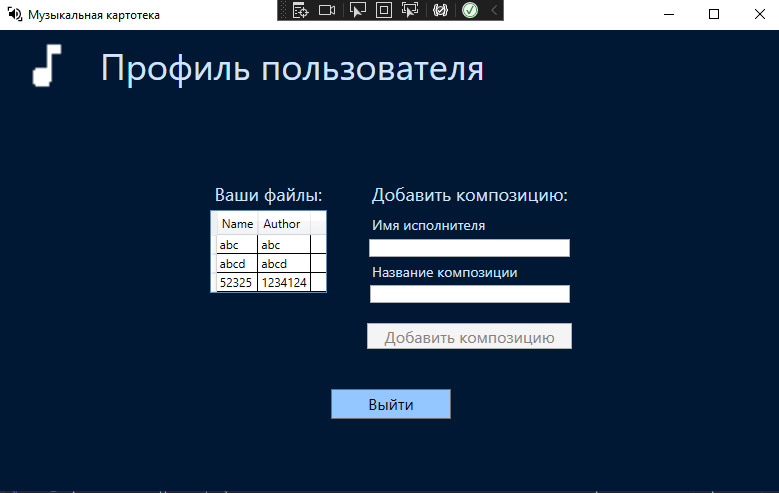


Рисунок 6. Финальный вид интерфейса пользователя

**РАЗДЕЛ 4. UNIT ТЕСТИРОВАНИЕ**

Так как в приложении присутствует возможность регистрации пользователя и в программе существует функция подсчёта для выделяемого ID для пользователя, в качестве теста я решил проверить правильно ли работает функция подсчёта ID, код для регистрации пользователя приведён в листинге 2

Листинг 2 – Код описывающий процесс регистрации нового пользователя в базе данных

|  |
| --- |
| Song songObj = new Song()  {  Id = AppConnection.model0db.Song.Count() + 1,  Name = compname.Text,  Author = authorname.Text  }; |

Был создан проект модульного теста, к которому подключили базу данных. После был написан код, который можно видеть в листинге 2.2

Листинг 3 – Проверка введённых данных на достоверность

|  |
| --- |
| public class UnitTest1  {  public static MU\_Entities\_2 model;  public static MU\_Entities\_3 model1;  [TestMethod]  public void TestMethod1()  {  model1 = new MU\_Entities\_3() ;  model = new MU\_Entities\_2();  int counter = 1;  int excepted = model1.MUUser.Count() + 1;  int actual = model.MUUser.Count();  Assert.AreEqual(excepted, actual + counter) ;  }  } |

Данный код сравнивает счёт из базы данных основного проекта с счётом с нуля в проекте модульного теста, после успешно выполненного теста (рис.7) было установлено, что подсчёт ID работает без ошибок и регистрация нового пользователя происходит по правилам.

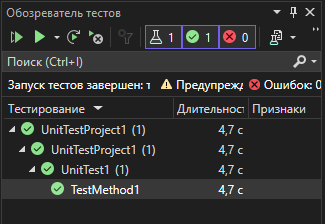


Рисунок 7. Успешно пройденное UNIT-Тестирование

**ЗАКЛЮЧЕНИЕ**

В рамках проекта было разработано приложение на платформе WPF, имеющее возможность авторизации и регистрации пользователя в базе данных. Также была создана база данных, в которой хранятся данные о пользователях программы. Были изучены средства и методы связи базы данных с приложением, чтобы данные как брались из базы данных для изменения вида приложения, так и отправлялись в базу данных.

*Приложение 1*

[**Электронный ресурсы**]

Ссылка на приложение

URL: https://github.com/4njess/bd-proekt