МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационные системы и технологии

Специальность 1-98 01 03 «Программное обеспечение информационной безопасности мобильных ситсем»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОЙ РАБОТЫ**

по дисциплине «Объектно-ориентированное программирование»

Тема «Фитнес-тренер»

Исполнитель:

студент 2 курса группы 2 \_\_\_\_\_\_\_\_\_\_ С.А.Юревич

подпись, дата

Руководитель:

ассистент \_\_\_\_\_\_\_\_\_\_\_ E.А.Чевжик

подпись, дата

Курсовая работа защищена с оценкой

Руководитель Е.А. Чевжик

подпись

Минск 2019

**Реферат**

Пояснительная записка курсового проекта содержит 0 страниц, 0 рисунков, 0 источников литературы, 1 приложение.

Основной целью курсового проекта является: разработка программного средства для спортсменов – «Fitness Coach».

Пояснительная записка состоит из введения, четырёх разделов, заключения.

Во введении представлена общая информация, дающая представление о предстоящей работе, определены цели.

В первом разделе рассматривается алгоритм решения, постановка задачи и обзор прототипов.

Во втором разделе рассматривается архитектура базы данных и программного средства.

В третьем разделе представлен процесс и результаты разработки итогового приложения.

В четвёртом разделе представлено руководство пользователя.

В заключении представлены итоги курсового проектирования и задачи, которые были решены в ходе разработки программного средства.

**Содержание**

[**1.Постановка задачи** 5](#_Toc10027564)

[**1.1 Алгоритмы решения** 5](#_Toc10027565)

[**1.2 Обзор прототипов** 6](#_Toc10027566)

[**2. Разработка архитектуры проекта** 8](#_Toc10027567)

[**2.1 Обобщённая структура** 8](#_Toc10027568)

[**2.2 Диаграммы UML, взаимосвязь всех компонентов** 10](#_Toc10027569)

[**3. Разработка функциональной модели и модели данных ПС** 13](#_Toc10027570)

[**3.1 Модель базы данных** 13](#_Toc10027571)

[**3.2 Выполняемые функции** 14](#_Toc10027572)

[**3.2.1 Регистрация и авторизация** 14](#_Toc10027573)

[**3.2.2 Реализация главного окна** 14](#_Toc10027574)

[**4. Руководство пользователя** 15](#_Toc10027575)

[**Заключение** 20](#_Toc10027576)

[**Список используемых источников** 21](#_Toc10027577)

[**Приложение А** 22](#_Toc10027578)

[**Приложение Б** 28](#_Toc10027579)

**Введение**

Приложение ComicsMaster – программное средство для чтения комиксов. С помощью данного инструмента можно весело проводить свой досуг занимаясь чтением комиксов и их обсуждением.

Сейчас в связи с популярностью 2D анимированных мультипликационных фильмов, многие всё чаще начинают читать комиксы с целью углубиться в сюжетную линию мультипликации, или же современных фильмов Компаний Marvel и DC, ведь именно они продвинули в прошлом истории о супергероях с красочными иллюстрациями, такими будто смотришь мультфильм, но только по кадрам. И данное приложение помогает в считанные минуты занять себя чтением на несколько часов.

Прогресс ушел далеко вперед, и теперь не обязательно покупать комиксы. Люди, будучи не в магазине, могут свободно читать комиксы где угодно.

Целью курсового проекта является проектирование и реализация программного средства «ComicsMaster».

**1.Постановка задачи**

Основной задачей курсового проекта является разработка программного средства, позволяющего пользователю хранить информацию в базе данных.

Функционально ПС должно выполнять следующие задачи:

* выполнять авторизацию и регистрацию пользователя;
* выдавать комикс пользователям для чтения;
* просматривать историю посещения;
* Просматривать последние добавленные комиксы;
* Выбор комиксов по категориям;

**1.1 Алгоритмы решения**

В разработке приложения были использованы нижеперечисленные технологии:

* ADO.NET;
* Microsoft SQL Server;
* WPF.

ADO.NET (ActiveX Data Object для .NET) – технология, предоставляющая доступ к данным для приложений, основанных на Microsoft .NET. Является не развитием более ранней технологии ADO, а самостоятельной технологией, частью фреймворка .NET. В отличие от классической ADO, которая была в основном предназначена для тесно связанных клиент-серверных систем, ADO.NET больше нацелена на автономную работу с помощью объектов DataSet. Эти типы представляют локальные копии любого количества взаимосвязанных таблиц данных, каждая из которых содержит набор строк и столбцов. Объекты DataSet позволяют вызывающей сборке (наподобие веб-страницы или программы, выполняющейся на настольном компьютере) работать с содержимым DataSet, изменять его, не требуя подключения к источнику данных, и отправлять обратно блоки измененных данных для обработки с помощью соответствующего адаптера данных. Фундаментальное различие между классической ADO и ADO.NET состоит в том, что ADO.NET является управляемой кодовой библиотекой, и, значит, подчиняется тем же правилам, что и любая управляемая библиотека. Типы, составляющие ADO.NET, используют протокол управления памятью CLR, принадлежат к той же системе типов (классы, интерфейсы, перечисления, структуры и делегаты), и доступ к ним возможен с помощью любого языка .NET. Классы ADO.NET находятся в сборке System.Data.dll

Microsoft SQL Server – система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз, данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

Microsoft SQL Server в качестве языка запросов использует версию SQL, получившую название Transact-SQL (сокращённо T-SQL), являющуюся реализацией SQL-92 (стандарт ISO для SQL) с множественными расширениями. T-SQL позволяет использовать дополнительный синтаксис для хранимых процедур и обеспечивает поддержку транзакций (взаимодействие базы данных с управляющим приложением). Microsoft SQL Server и Sybase ASE для взаимодействия с сетью используют протокол уровня приложения под названием Tabular Data Stream (TDS, протокол передачи табличных данных). Протокол TDS также был реализован в проекте FreeTDS с целью обеспечить различным приложениям возможность взаимодействия с базами данных Microsoft SQL Server и Sybase.

Windows Presentation Foundation (WPF) – система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление.

Графической технологией, лежащей в основе WPF, является DirectX, в отличие от Windows Forms, где используется GDI/GDI+. Производительность WPF выше, чем у GDI+ за счёт использования аппаратного ускорения графики через DirectX.

**1.2 Обзор прототипов**

Программное средство “Comics”.

Приложение Comics призвано разукрасить пользовательский досуг чтением комиксов. Пользователи скачав и установив приложение получают доступ к более чем ста тысячам комиксов для бесплатного чтения, также в приложении есть и платные комиксы, в свою очередь пользователь приобретает их отдельно. Также в приложении есть сортировка комиксов по категориям. Внешний вид приложения представлен на рисунке 1.1.

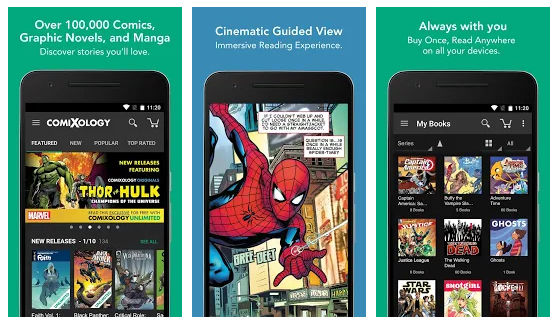


Рисунок 1.1 – приложение Comics!

Пользователи могут читать комиксы на свои смартфонах, могут ставить им отметки и просматривать купленные комиксы на различных устройствах.

Из недостатков можно выделить необходимость постоянного подключения к интернету.

**2. Разработка архитектуры проекта**

**2.1 Обобщённая структура**

Решение ComicsMaster представлено одним проектом ComicsMaster, имеющим структуру, представленную на рисунке 2.1.

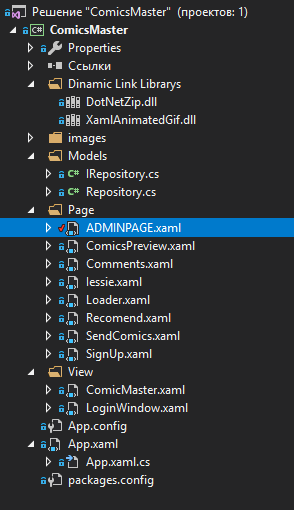


Рисунок 2.1 – Структура проекта

Все пояснения о функционале файлов описаны ниже в Таблице 2.1.

Таблица 2.1 – Пояснения значения файлов

|  |  |
| --- | --- |
| Properties | Свойства проекта, содержит информацию о сборке, используемых ресурсах и настройках |
| Ссылки | Перечень сборок, используемых в проекте |
| Properties | Папка со свойствами проекта |
| App.config | Файл с параметрами проекта |
| App.xaml | Класс Application |
| Dinamic Link Librarys | Папка, содержащая используемые в проекте библиотеки |
| images | Папка с картинками |
| Models | Папка, содержащая классы для работы с БД |
| Page | Папка содержащая пользовательские элементы управления |
| View | Папка, содержащая окна приложения |
| ADMINPAGE.xaml | Страница админа для добавления комиксов |
| ComicsPreview.xaml | Шаблон для построения комикса |
| Comments.xaml | Пользовательский элемент управления, являющийся шаблоном для построения комментариев |
| Iessie.xaml | Пользовательский элемент управления, являющийся шаблонном для построения глав комикса |
| Loader.xaml | Прелоадер |
| Recomend.xaml | Пользовательский элемент управления |
| SendComics.xaml | Пользовательский элемент управления, являющейся шаблоном для построения Сообщения в комментарии |
| SignUp.xaml | Пользовательский элемент управления, необходимый для создания новых пользователей |
| LoginWindow.xaml | Окно авторизации |
| ComicMaster.xaml | Окно с комиксами |
| DotNetZip.dll | Библиотека для работы с архивами |
| XamlAnimatedGif.dll | Библиотека для работы с изображениями в gif формате |

**2.2 Диаграммы UML, взаимосвязь всех компонентов**

Диаграмма UML – это графическое представление набора элементов, изображаемое чаще всего в виде графа с вершинами (сущностями) и ребрами (отношениями). Диаграммы рисуют для визуализации. Основная цель диаграмм – визуализация разрабатываемой системы с разных точек зрения.

На диаграммах 2.2 и 2.3 представлены принципы работы приложения с точки зрения пользователя и админа соответственно.

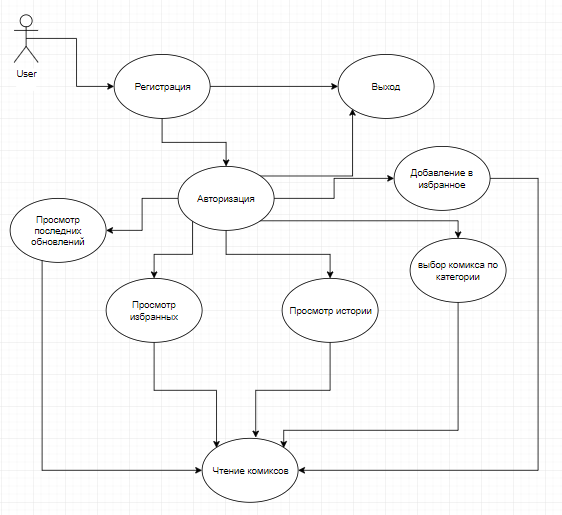


Рисунок 2.2 – Принцип работы спортсмена

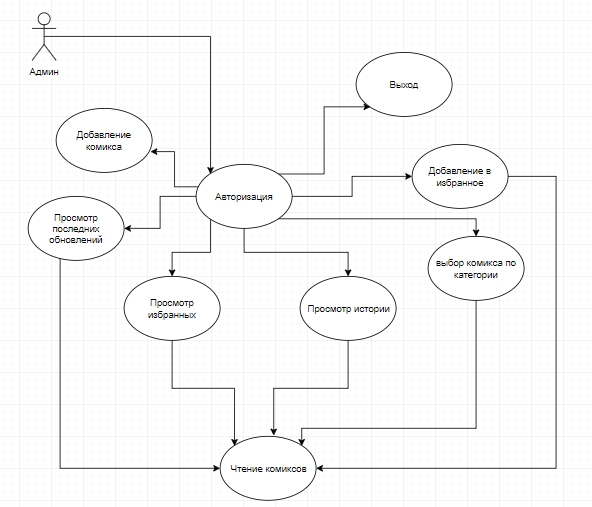


Рисунок 2.3 – Принцип работы админа

На рисунке 2.4 представлена диаграмма классов

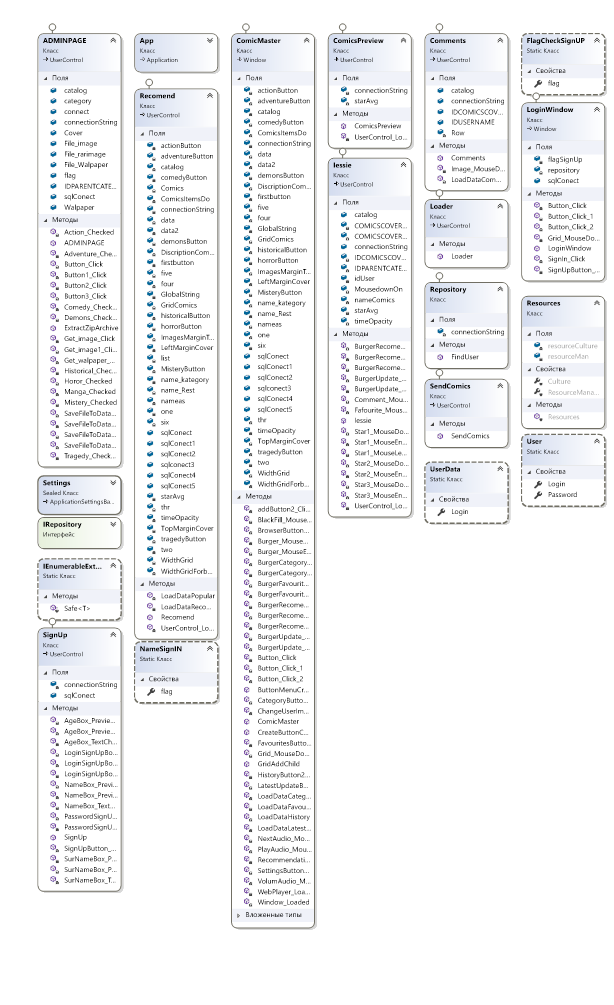


Рисунок 2.4 – диаграмма классов

**3. Разработка функциональной модели и модели данных ПС**

**3.1 Модель базы данных**

Для реализации поставленной задачи была создана база данных ComicsKursach. Для её создания использовалась система управления реляционными базами данных Microsoft SQL Server 2012.

База данных состоит из 8 таблиц, перечисленных ниже (рисунок 3.1). Скрипты для работы с базой данных представлены в Приложении Б.

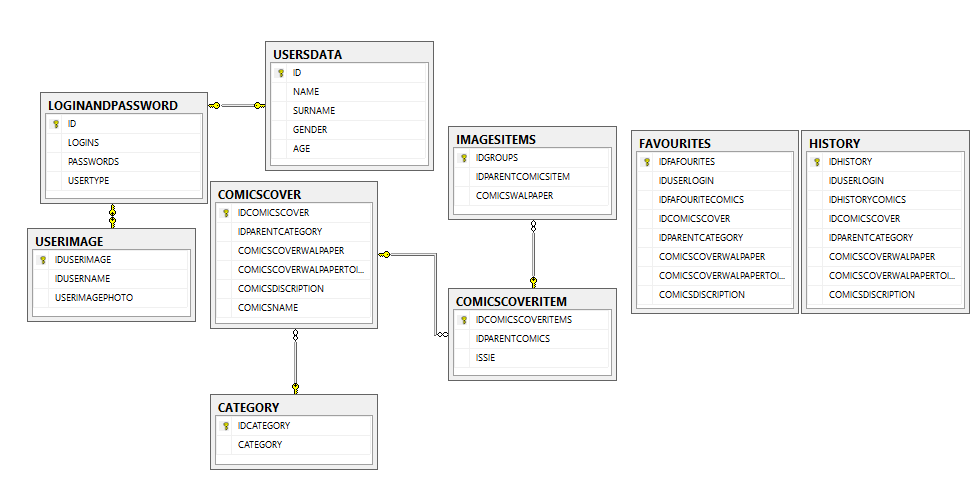


Рисунок 3.1 – Взаимосвязь таблиц базы данных

Таблица «USERDATA», содержит информацию о пользователе.

Таблица «LOGINANDPASSWORD»,

Таблица «USERIMAGE», содержит информацию о картинки пользователя.

Таблица «COMICSCOVER», содержит информацию о комиксе.

Таблица «CATEGORY», содержит информацию о категориях.

Таблица «COMICSCOVERITEMS», содержит номер комикса и его Главы.

Таблица «IMAGESITEMS», содержит ссылки на картинки комиксов.

Таблица «FAVOURITE», содержит информацию о комиксах, которые пользователь добавил себе в избранное.

Таблица «HISTORY», содержит информацию о посещённых пользователем комиксах.

**3.2 Выполняемые функции**

Данное приложение предоставляет минимальные требования к выполнению функций приложения чтения комиксов и выполняет следующие функции:

* выполнять авторизацию и регистрацию пользователя;
* выдавать комикс пользователям для чтения;
* просматривать историю посещения;
* Просматривать последние добавленные комиксы;
* Выбор комиксов по категориям;

**3.2.1 Регистрация и авторизация**

Регистрация и(или) авторизация пользователя происходит в окне LoginWindow в пользовательском элементе управления SignUp, имеющем следующую структуру, отображенную на рисунке 3.7.

Листинг наиболее важных методов представлен в приложении A.

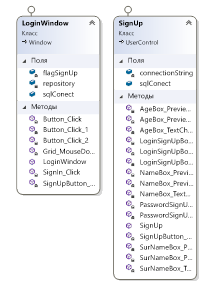


Рисунок 3.7 – Структура классов SignUp и LoginWindow

**3.2.2 Реализация главного окна**

Главное окно – окно в котором происходит вся навигация в приложении.

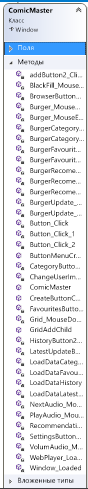
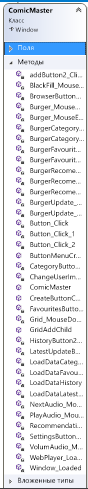
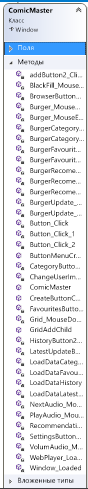
  

Рисунок 3.11 – Структура класса ComicMaster

**4. Руководство пользователя**

При запуске приложения открывается окно авторизации (рисунок 4.1), на котором имеются поля для инициализации пользователя и кнопка для регистрации.

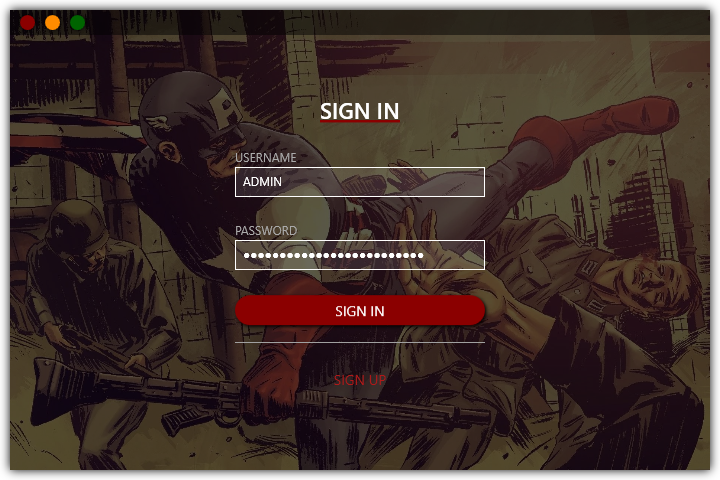


Рисунок 4.1 – Окно авторизации

Если у пользователя нет аккаунта, то он должен пройти регистрацию, нажав на кнопку «SignUp». Затем откроется окно регистрации, где пользователю будет предложено заполнить необходимую информацию для дальнейшего использования программного обеспечения (рисунок 4.2).

После успешного прохождения регистрации и авторизации откроется главное окно приложения (Рисунок 4.3). На главном окне расположены кнопки навигации: Главная, Последние обновления, Избранное, Открыть боковое меню; кнопки Радио плеера: Включить/выключить звук, Следующий трек, Старт и пауза.

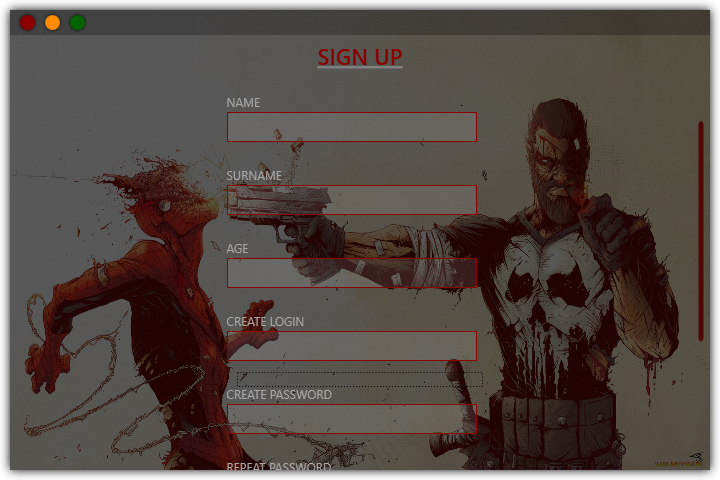


Рисунок 4.2 – Окно регистрации

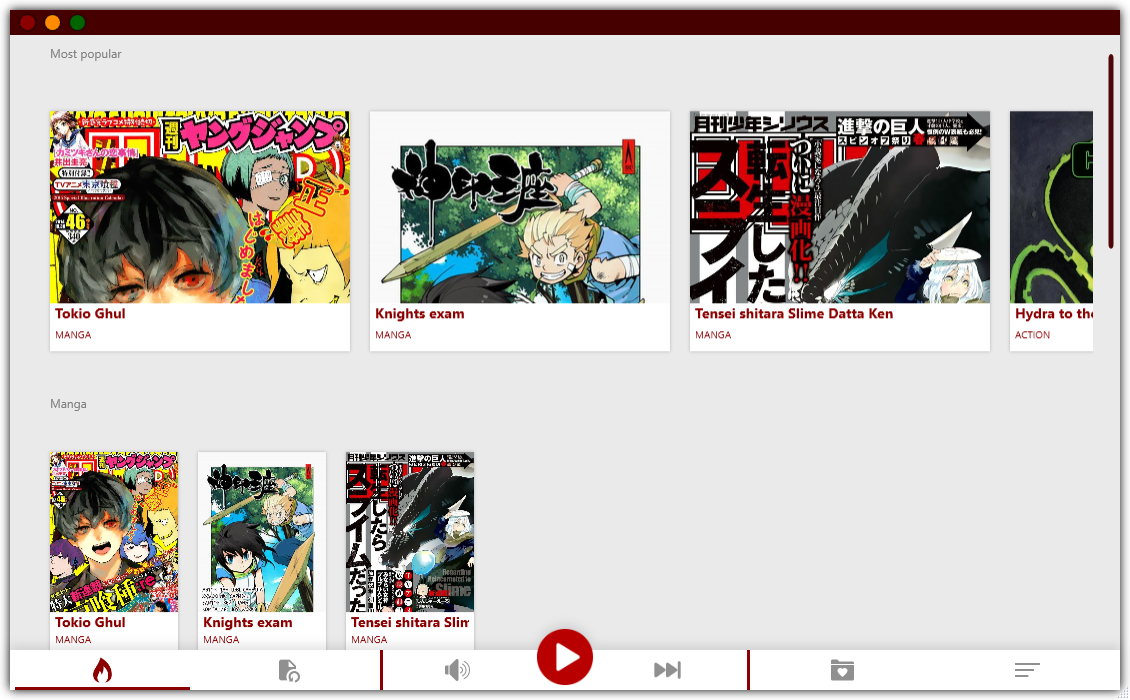


Рисунок 4.3 – Личный кабинет пользователя

Нажав на кнопку открыть боковое меню пользователь получает доступ к доп. Функционалу. Кнопки: Категории, Рекомендации, Избранное. История, Последние обновления. У админа же есть кнопка добавить   
комикс (Рисунок4.4).

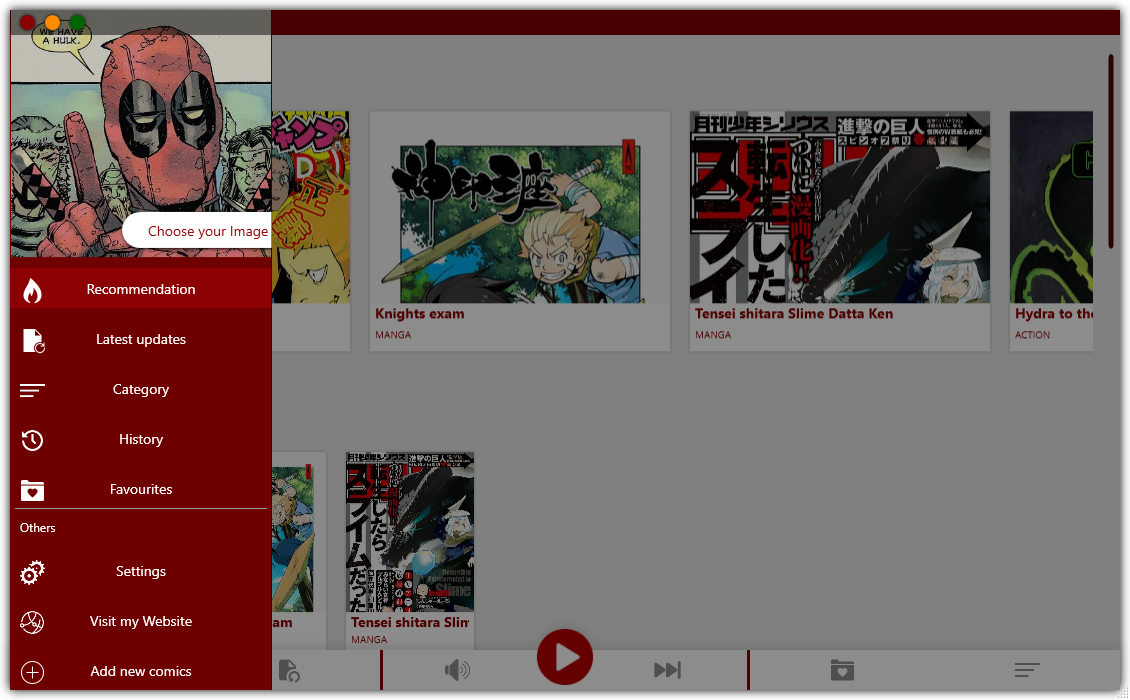


Рисунок 4.4 – Нажатие ан кнопку «Открыть боковое меню»

Нажатие на кнопку Последние обновл. и избранное (Рисунок 4.5 и 4.6).

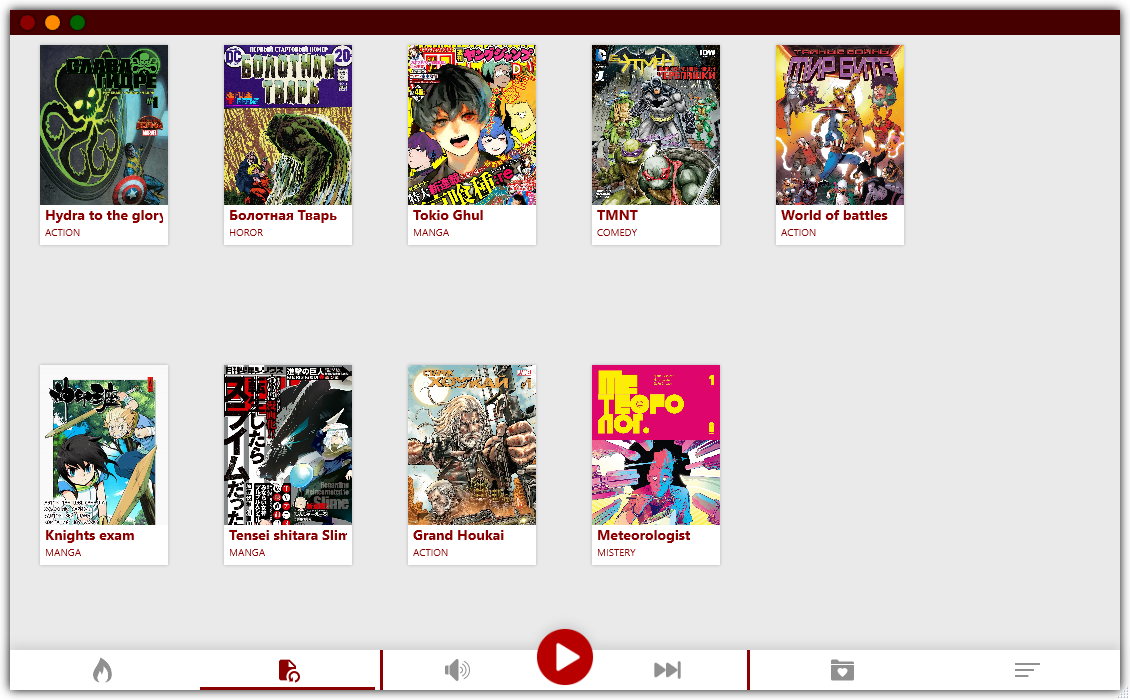


Рисунок 4.5 – Нажатие на кнопку «Последнее обновление»



Рисунок 4.6 – Нажатие на кнопку «Избранное»

Страница главы комикса Содержит в себе кнопки для открытия определенной главы комикса. Кнопки отметок, кнопку чтения, добавления в избранное и комментарии (с права в верху)

Страница Главы и комментариев приведены соответственно на Рисунке 4.7 и 4.8.

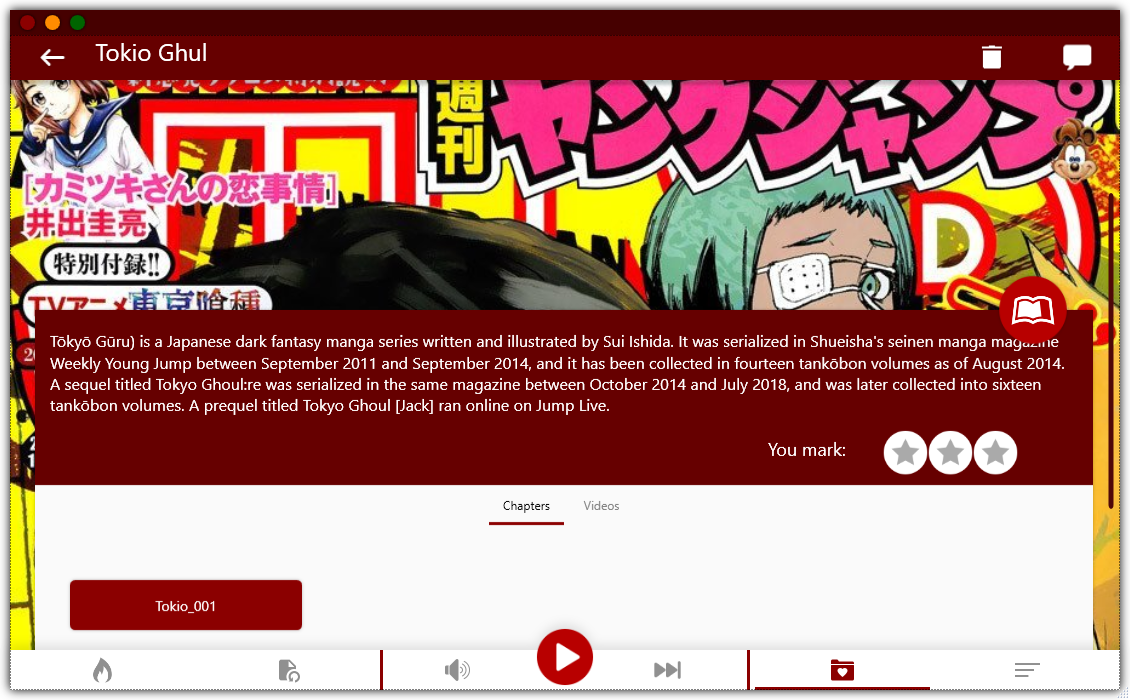


Рисунок 4.7 – страница Глав

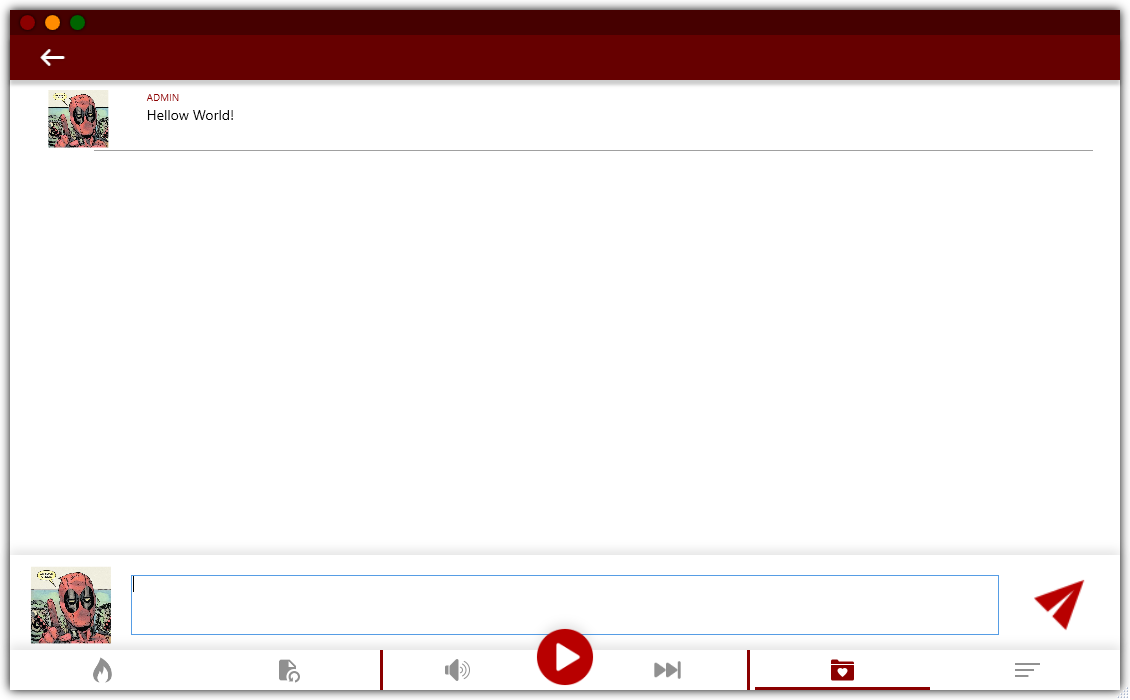


Рисунок 4.8 – Нажатие на кнопку «Комментарии»

Все введенные пользователем данные сохраняются в базу данных и при последующем входе в личный кабинет – автоматически подгружаются в личный кабинет, что несомненно очень удобно для пользователя.

**Заключение**

В курсовом проекте описана и реализована программа, написанная на языке C#, с использованием технологий ADO.NET, WPF.

В ходе выполнения курсового проекта было создано приложение для чтения, которое выполняет следующие функции:

* выполнять авторизацию и регистрацию пользователя;
* выдавать комикс пользователям для чтения;
* просматривать историю посещения;
* Просматривать последние добавленные комиксы;
* Выбор комиксов по категориям;

Данное программное средство имеет удобный и понятный интерфейс, реагирует на ошибочный ввод данных (например, ввод данных не того типа, проверка на пустоту вводимых значений).

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает верно, а требования технического задания выполнены в полном объеме.

**Список используемых источников**

1. MSDN сеть разработчиков в Microsoft [Электронный ресурс] – <http://msdn.microsoft.com/library/rus/>.
2. Язык C# и .NET Framework [Электронный ресурс] – <http://professorweb.ru/my/csharp/charp_theory/level1/infonet.php>.
3. Руководство по ADO.NET и работе с базами данных [Электронный ресурс] – <https://metanit.com/sharp/adonet/>. Дата доступа: 10.05.2017

4 Форум для программистов или разработчиков [Электронный ресурс] – https://stackoverflow.com/

**Приложение А**

Листинг методов регистрации:

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

{

FlagCheckSignUP.flag = false;

sqlConect = new SqlConnection(connectionString);

sqlConect.Open();

string sq1 = string.Format("Insert Into USERSDATA" + "(NAME,SURNAME,GENDER,AGE) Values (@NAME,@SURNAME,@GENDER,@AGE); Insert Into LOGINANDPASSWORD" + "(LOGINS,PASSWORDS) Values (@LOGINS,@PASSWORDS)");

using (SqlCommand cmd = new SqlCommand(sq1, this.sqlConect))

{

cmd.Parameters.AddWithValue("@NAME", NameBox.Text);

cmd.Parameters.AddWithValue("@SURNAME", SurNameBox.Text);

cmd.Parameters.AddWithValue("@GENDER", "INCOGNITO");

cmd.Parameters.AddWithValue("AGE", AgeBox.Text);

cmd.Parameters.AddWithValue("@LOGINS", LoginSignUpBox.Text);

cmd.Parameters.AddWithValue("@PASSWORDS", PasswordSignUpBox.Password.GetHashCode());

cmd.ExecuteNonQuery();

FlagCheckSignUP.flag = true;

User.Login = LoginSignUpBox.Text;

}

sqlConect.Close();

if (FlagCheckSignUP.flag == true)

{

DoubleAnimation ShowSignUP = new DoubleAnimation();

ShowSignUP.From = this.ActualHeight;

ShowSignUP.To = 0;

ShowSignUP.Duration = TimeSpan.FromSeconds(0.15);

this.BeginAnimation(UserControl.HeightProperty, ShowSignUP);

}

}

private void PasswordSignUpBox\_PasswordChanged(object sender, RoutedEventArgs e)

{

if (PasswordSignUpBox.Password.Length <= 4)

{

PasswordSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(255, 0, 0));

BlockPas.Text = "PASSWORD IS SMALL";

WarningPassword.Visibility = Visibility.Visible;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

else

{

PasswordSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(0, 0, 0));

BlockPas.Text = "";

WarningPassword.Visibility = Visibility.Hidden;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

}

private void LoginSignUpBox\_TextChanged(object sender, TextChangedEventArgs e)

{

sqlConect = new SqlConnection(connectionString);

int sqlrezult;

sqlConect.Open();

string sq1 = string.Format("SELECT COUNT(\*) from LOGINANDPASSWORD WHERE" + "( LOGINS=" + "@LOG)");

using (SqlCommand cmd = new SqlCommand(sq1, this.sqlConect))

{

cmd.Parameters.AddWithValue("@LOG", LoginSignUpBox.Text);

sqlrezult = (int)cmd.ExecuteScalar();

}

sqlConect.Close();

if (!Regex.IsMatch(LoginSignUpBox.Text, "[A-zА-я]"))

{

LoginSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(255, 0, 0));

ShortLogin.Text = "ADD ALPHABETIC CHARACTERS";

WarningLogin.Visibility = Visibility.Visible;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

else if (LoginSignUpBox.Text.Length < 5)

{

LoginSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(255, 0, 0));

ShortLogin.Text = "LOGIN IS SMALL";

WarningLogin.Visibility = Visibility.Visible;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

else if (sqlrezult==1)

{

LoginSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(255, 0, 0));

ShortLogin.Text = "LOGIN IS BUSY";

WarningLogin.Visibility = Visibility.Visible;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

else if (LoginSignUpBox.Text.Length > 95)

{

LoginSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(255, 0, 0));

ShortLogin.Text = "LOGIN IS SO BIG";

WarningLogin.Visibility = Visibility.Visible;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

else if (LoginSignUpBox.Text.Length ==0)

{

LoginSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(255, 0, 0));

ShortLogin.Text = "LOGIN IS EMPTY";

WarningLogin.Visibility = Visibility.Visible;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

else

{

LoginSignUpBox.BorderBrush = new SolidColorBrush(Color.FromRgb(0, 0, 0));

ShortLogin.Text = "";

WarningLogin.Visibility = Visibility.Hidden;

SignUpButton.Click -= new System.Windows.RoutedEventHandler(this.SignUpButton\_Click);

}

}

Листинг методов Главного окна:

private async Task LoadDataLatestUpdate()

{

Loader loader = new Loader();

Canvas.SetZIndex(loader, (int)98);

loader.Margin = new Thickness(0, 0, -40, 0);

Content.Children.Add(loader);

sqlConect = new SqlConnection(connectionString);

await sqlConect.OpenAsync();

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

string sql = "SELECT \* FROM COMICSCOVER";

SqlCommand command = new SqlCommand(sql, connection);

SqlDataReader reader = await command.ExecuteReaderAsync();

while (reader.Read())

{

nameas = reader.GetValue(0).ToString();

name\_kategory = reader.GetString(1);

data = reader.GetString(2);

data2 = reader.GetString(3);

DiscriptionComics = reader.GetString(4);

name\_Rest = reader.GetString(5);

ComicsPreview comicsPreview = new ComicsPreview();

Iessie iessie = new Iessie(Convert.ToInt32(nameas), name\_kategory, data, data2);

iessie.Margin = new Thickness(0, 0, 0, 0);

iessie.ComicsDiscription.Text = DiscriptionComics;

iessie.ComicsName.Text = name\_Rest;

//iessie.Content.HorizontalAlignment = HorizontalAlignment.Center;

comicsPreview.ComicsWalpeper.Source = new BitmapImage(new Uri(catalog+data));

comicsPreview.NameComix.Text = name\_Rest;

comicsPreview.NameGlavaComix.Text = name\_kategory;

#region IESSIE\_ADD\_BUTTON

int buttonMarginLeft = 0,

buttonMarginTop = 0;

sqlConect1 = new SqlConnection(connectionString);

await sqlConect1.OpenAsync();

var image = new Image { Source = new BitmapImage(new Uri(catalog + data2)) };

using (SqlConnection connection1 = new SqlConnection(connectionString))

{

connection1.Open();

string sql1 = "SELECT \* FROM COMICSCOVERITEM";

SqlCommand command1 = new SqlCommand(sql1, connection1);

SqlDataReader reader1 = await command1.ExecuteReaderAsync();

while (reader1.Read())

{

string nameqw = reader1.GetValue(1).ToString();

string name\_Rest1 = reader1.GetString(2);

GlobalString = name\_Rest1;

if (nameqw == nameas)

{

Button button = new Button();

button.Content = name\_Rest1;

button.Margin = new Thickness(10, 5, 10, 5);

button.Height = 50;

if (buttonMarginLeft < 4)

{

if (firstbutton) {

Grid.SetRow(button, buttonMarginTop);

Grid.SetColumn(button, buttonMarginLeft);

WidthGridForbutton++;

buttonMarginLeft++;

button.Background = Brushes.DarkRed;

button.Foreground = Brushes.White;

firstbutton = false;

button.Style = (Style)button.FindResource("ButtonComicsIssieStylefirst");

}

else {

button.Style = (Style)button.FindResource("ButtonComicsIssieStyle");

Grid.SetRow(button, buttonMarginTop);

Grid.SetColumn(button, buttonMarginLeft);

WidthGridForbutton++;

buttonMarginLeft++;

}

}

else

{

button.Style = (Style)button.FindResource("ButtonComicsIssieStyle");

WidthGridForbutton = 0;

iessie.ContentForButton.RowDefinitions.Add(new RowDefinition { Height = GridLength.Auto });

buttonMarginLeft = 0; buttonMarginTop++;

Grid.SetRow(button, buttonMarginTop);

Grid.SetColumn(button, buttonMarginLeft);

buttonMarginLeft++;

}

button.Click += (sender, e) =>

{

Content.Children.Clear();

ContentToIessie.Children.Clear();

BackgroundIessie.Children.Clear();

Loader loaderItems = new Loader();

Canvas.SetZIndex(loaderItems, (int)98);

Content.Children.Add(loaderItems);

BackgroundIessie.Children.Remove(image);

Content.Children.Add(GridAddChild(name\_Rest1));

Content.Children.Remove(loaderItems);

ImagesMarginTop = 0;

};

iessie.ContentForButton.Children.Add(button);

}

};

connection1.Close();

}

buttonMarginTop = 0; buttonMarginLeft = 0; firstbutton = true;

#endregion

iessie.backImage.MouseDown += (sen, t) => { Content.Children.Clear(); ContentToIessie.Children.Clear(); BackgroundIessie.Children.Clear(); LoadDataLatestUpdate().GetAwaiter(); };

comicsPreview.ButtonGrid.MouseDown += (source, e) => {

Content.Children.Clear();

ContentToIessie.Children.Clear();

ContentToIessie.Children.Add(iessie);

BackgroundIessie.Children.Add(image);

try

{

var str = NameSignIN.flag;

var sqlConectHistory = new SqlConnection(connectionString);

int sqlrezult;

sqlConectHistory.Open();

string sq1 = string.Format("SELECT COUNT(\*) from HISTORY WHERE" + "( IDHISTORYCOMICS=" + "@LOG)");

using (SqlCommand cmd = new SqlCommand(sq1, sqlConectHistory))

{

cmd.Parameters.AddWithValue("@LOG", name\_Rest);

sqlrezult = (int)cmd.ExecuteScalar();

}

sqlConect.Close();

if (sqlrezult < 1)

{

var connectHistory = new SqlConnection(connectionString);

connectHistory.Open();

string sqlQ = string.Format("Insert Into HISTORY(IDUSERLOGIN,IDHISTORYCOMICS,IDCOMICSCOVER,IDPARENTCATEGORY,COMICSCOVERWALPAPER,COMICSCOVERWALPAPERTOIESSIE,COMICSDISCRIPTION) Values(@IDUSERLOGIN,@IDHISTORYCOMICS,@IDCOMICSCOVER,@IDPARENTCATEGORY,@COMICSCOVERWALPAPER,@COMICSCOVERWALPAPERTOIESSIE,@COMICSDISCRIPTION)");

using (SqlCommand cmd = new SqlCommand(sqlQ, connectHistory))

{

cmd.Parameters.AddWithValue("@IDUSERLOGIN", str);

cmd.Parameters.AddWithValue("@IDHISTORYCOMICS", name\_Rest);

cmd.Parameters.AddWithValue("@IDCOMICSCOVER", nameas);

cmd.Parameters.AddWithValue("@IDPARENTCATEGORY", name\_kategory);

cmd.Parameters.AddWithValue("@COMICSCOVERWALPAPER", data);

cmd.Parameters.AddWithValue("@COMICSCOVERWALPAPERTOIESSIE", data2);

cmd.Parameters.AddWithValue("@COMICSDISCRIPTION", iessie.ComicsDiscription.Text);

cmd.ExecuteNonQuery();

}

connectHistory.Close();

}

}

catch

{

MessageBox.Show("Error");

}

};

if (WidthGrid < (int)Content.ActualWidth)

{

comicsPreview.Margin = new Thickness(LeftMarginCover \* 184, 320 \* TopMarginCover, 0, 0);

LeftMarginCover++;

WidthGrid = LeftMarginCover \* 184;

}

if (WidthGrid >= (int)Content.ActualWidth)

{

LeftMarginCover = 0; TopMarginCover++;

comicsPreview.Margin = new Thickness(LeftMarginCover \* 184, 320 \* TopMarginCover, 0, 0);

LeftMarginCover++;

WidthGrid = 0;

}

Content.Children.Add(comicsPreview);

};

connection.Close();

LeftMarginCover = 0;

TopMarginCover = 0;

WidthGrid = 0;

}

Content.Children.Remove(loader);

}

**Приложение Б**

Листинг SQL для создания/удаления таблиц

CREATE DATABASE ComicsKursach

USE ComicsKursach;

CREATE TABLE USERSDATA(

ID BIGINT IDENTITY(1,1) PRIMARY KEY,

NAME NVARCHAR(MAX),

SURNAME NVARCHAR(MAX),

GENDER NVARCHAR(MAX),

AGE NVARCHAR(MAX));

CREATE TABLE LOGINANDPASSWORD(

ID BIGINT IDENTITY(1,1) PRIMARY KEY,

LOGINS NVARCHAR(255) UNIQUE,

PASSWORDS NVARCHAR(MAX),

USERTYPE NVARCHAR(MAX) DEFAULT('USER'), FOREIGN KEY (ID) REFERENCES USERSDATA(ID));

INSERT INTO USERSDATA(NAME,SURNAME,GENDER,AGE) VALUES ('ADMIN','ADMIN','MALE','0');

INSERT INTO LOGINANDPASSWORD(LOGINS,PASSWORDS,USERTYPE) VALUES ('ADMIN','-1470019936','ADMIN');

CREATE TABLE USERIMAGE(

IDUSERIMAGE BIGINT IDENTITY(1,1) PRIMARY KEY,

IDUSERNAME BIGINT UNIQUE, FOREIGN KEY (IDUSERNAME) REFERENCES LOGINANDPASSWORD(ID),

USERIMAGEPHOTO NVARCHAR(MAX));

DROP TABLE LOGINANDPASSWORD;

DROP TABLE USERSDATA;

DROP TABLE IMAGESITEMS;

DROP TABLE COMICSCOVERITEM;

DROP TABLE HISTORY;

DROP TABLE COMICSCOVER;

DROP TABLE CATEGORY;

DROP TABLE FAVOURITES;

DROP TABLE USERIMAGE;

DROP TABLE COMICSCOMMENT;

drop table MARK;

CREATE TABLE CATEGORY(

IDCATEGORY BIGINT IDENTITY(1,1) PRIMARY KEY,

CATEGORY NVARCHAR(255) UNIQUE);

CREATE TABLE COMICSCOVER(

IDCOMICSCOVER BIGINT IDENTITY(1,1) PRIMARY KEY,

IDPARENTCATEGORY NVARCHAR(255),

COMICSCOVERWALPAPER NVARCHAR(MAX),

COMICSCOVERWALPAPERTOIESSIE NVARCHAR(MAX),

COMICSDISCRIPTION NVARCHAR(MAX),

COMICSNAME NVARCHAR(255) UNIQUE, FOREIGN KEY (IDPARENTCATEGORY) REFERENCES CATEGORY(CATEGORY));

CREATE TABLE HISTORY(

IDHISTORY BIGINT IDENTITY(1,1) PRIMARY KEY,

IDUSERLOGIN NVARCHAR(255),

IDHISTORYCOMICS NVARCHAR(255),

IDCOMICSCOVER BIGINT,

IDPARENTCATEGORY NVARCHAR(255),

COMICSCOVERWALPAPER NVARCHAR(MAX),

COMICSCOVERWALPAPERTOIESSIE NVARCHAR(MAX),

COMICSDISCRIPTION NVARCHAR(MAX));

CREATE TABLE FAVOURITES(

IDFAFOURITES BIGINT IDENTITY(1,1) PRIMARY KEY,

IDUSERLOGIN NVARCHAR(255),

IDFAFOURITECOMICS NVARCHAR(255),

IDCOMICSCOVER BIGINT,

IDPARENTCATEGORY NVARCHAR(255),

COMICSCOVERWALPAPER NVARCHAR(MAX),

COMICSCOVERWALPAPERTOIESSIE NVARCHAR(MAX),

COMICSDISCRIPTION NVARCHAR(MAX));

CREATE TABLE COMICSCOMMENT(

IDCOMICSCOMMENT BIGINT IDENTITY(1,1) PRIMARY KEY,

IDCOMICSCOVER NVARCHAR(255),

USERNAME NVARCHAR(255),

COMMENTMESSAGE NVARCHAR(MAX));

CREATE TABLE COMICSCOVERITEM(

IDCOMICSCOVERITEMS BIGINT IDENTITY(1,1) PRIMARY KEY,

IDPARENTCOMICS BIGINT,

ISSIE NVARCHAR(255) UNIQUE, FOREIGN KEY (IDPARENTCOMICS) REFERENCES COMICSCOVER(IDCOMICSCOVER));

CREATE TABLE IMAGESITEMS(

IDGROUPS BIGINT IDENTITY(1,1) PRIMARY KEY,

IDPARENTCOMICSITEM NVARCHAR(255),

COMICSWALPAPER NVARCHAR(MAX), FOREIGN KEY (IDPARENTCOMICSITEM) REFERENCES COMICSCOVERITEM(ISSIE));

CREATE TABLE MARK(

IDMARK BIGINT IDENTITY(1,1) PRIMARY KEY,

IDPARENTCOMICS NVARCHAR(255),

IDNAMECOMICS NVARCHAR(255),

IDUSERNAME NVARCHAR(255),

STAR Float);

Select \* from LOGINANDPASSWORD;

Select \* from USERSDATA;

Select \* from IMAGESITEMS;

Select \* from COMICSCOVERITEM;

Select \* from HISTORY;

Select \* from COMICSCOVER;

Select \* from CATEGORY;

Select \* from FAVOURITES;

Select \* from USERIMAGE;

Select \* from COMICSCOMMENT;

Select \* from MARK;