Компьютерная академия «ШАГ»

Одесский филиал

Кафедра Разработки программного обеспечения

КУРСОВОЙ ПРОЕКТ ПО ADO.NET

«Тестирующая система»

Студента группы ВПУ-1311

Ковалева Дениса

Руководитель курсового проекта:

Полянский Виталий Владилинович

Одесса 2015

АННОТАЦИЯ

Приложение «Тестирующая система» предназначено для проверки знаний. Система позволяет быстро, с помощью встроенного редактора вопросов и тестов, организовать проверку знаний учащихся или персонала организации. С помощью приложения «Тестирующая система» легко организовать проверку знаний учащихся по различным предметам, проведение экзаменов, проверку знаний и аттестацию персонала на повышение квалификации, проверку знаний рабочих и служащих при приёме на работу, психологические тесты на соответствие должности, проверку знаний иностранных языков, аттестацию по охране труда, аттестацию по электробезопасности и многое другое.

В системе есть свой встроенный редактор вопросов с простым и интуитивно понятным интерфейсом, который позволяет ввести вопрос любого вида и с любым содержанием.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Компьютерное тестирование знаний это эффективный способ проверки, который находит в образовании все большее применение. Одним из его достоинств компьютерного тестирования является минимум временных затрат на получение надежных итогов контроля, и получение результатов практически сразу по завершении контролирующего теста. От традиционных оценок и контроля знаний - тесты отличаются объективностью измерения результатов обучения, поскольку они ориентируются не на субъективное мнение преподавателей, а на объективные критерии.

Применение программ тестирования и компьютерных тестов при проверке знаний является экономически выгодным и обеспечивает повышение эффективности учебного процесса, объективности оценки уровня знаний и является рациональным дополнением к другим методам проверки знаний. Немаловажную роль, при использовании компьютерного тестирования, играет выбор программ для тестирования знаний, составление и создания тестов..

1. **ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

1.1 Постановка задачи

В рамках курсовой работы по ADO.NET необходимо разработать приложение, которое бы позволяло пользователю пройти выбранный тест и увидеть его результаты. Программа также посзволяет редактировать
тесты, но только пользователю с определенными правами.
Требования:

- Тест может иметь сколько угодно вопросов, каждый вопрос может иметь сколько угодно вариантов ответа. При этом один из вариантов ответа помечается как правильный.
- База данных содержит двух пользователей (admin, student).
- Пользователю teacher доступны окна для редактирования вопросов и добавления новых тестов.
- Пользователю student доступны окна для сдачи теста.
- Программа должна выводить список всех доступных тестов (под пользователем student).
- После прохождения теста система выдает результат.
- Для отображения вопросов выбирается дисциплина, все вопросы выводятся на панель (номер, вопрос, варианты ответа) при этом под каждым две кнопки (удалить, добавить). Такие же кнопки под каждым вариантом ответа (под пользователем teacher).
- Добавление и редактирование выполняется в отдельных окнах.

1.2 Список технологий и инструментальных средств

- Microsoft SQL Server 2012 Express;
- Microsoft Visual Studio 2012;
- Язык программирования C#;
- WPF;
- Entity Framework 5.0.0;

1.3 Требования к программным и техническим характеристикам системы

- Операционные системы – Windows XP , Windows 7;
- Платформа .NET FRAMEWORK 3.5 или выше;
- Microsoft SQL Server 2012 Express;
- Процессор с тактовой частотой 2 ГГц;
- Оперативная память 1 Гбайт;
- 1 Гбайт свободного места на жёстком диске.

2. ВЫБОР ТЕХНОЛОГИИ ДЛЯ РЕАЛИЗАЦИИ ПРОЕКТА

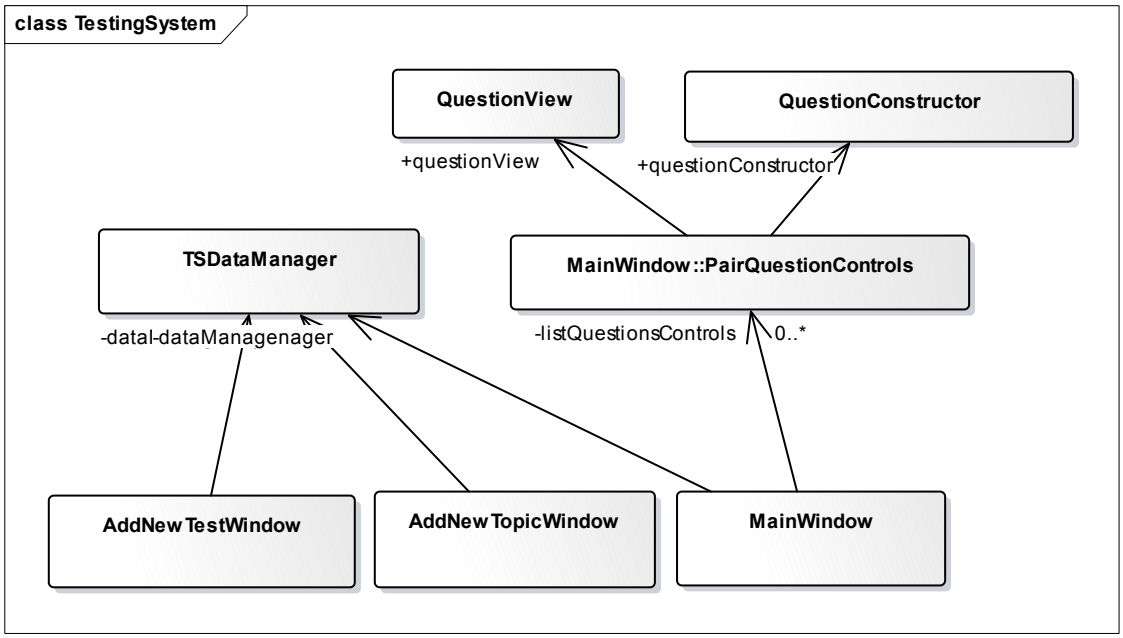
Для реализации проекта выбрана технология .NET Framework . Программа для .NET Framework, написанная на любом поддерживаемом языке программирования, сначала переводится компилятором в единый для .NET промежуточный байт-код Common Intermediate Language (CIL) (ранее назывался Microsoft Intermediate Language, MSIL). В терминах .NET получается *сборка*, англ. *assembly*. Затем код либо исполняется виртуальной машиной Common Language Runtime (CLR), либо транслируется утилитой NGen.exe в исполняемый код для конкретного целевого процессора. Использование виртуальной машины предпочтительно, так как избавляет разработчиков от необходимости заботиться об особенностях аппаратной части. В случае использования виртуальной машины CLR встроенный в неё JIT-компилятор «на лету» (just in time) преобразует промежуточный байт-код в машинные коды нужного процессора. Современная технология динамической компиляции позволяет достигнуть высокого уровня быстродействия. Виртуальная машина CLR также сама заботится о базовой безопасности, управлении памятью и системе исключений, избавляя разработчика от части работы.

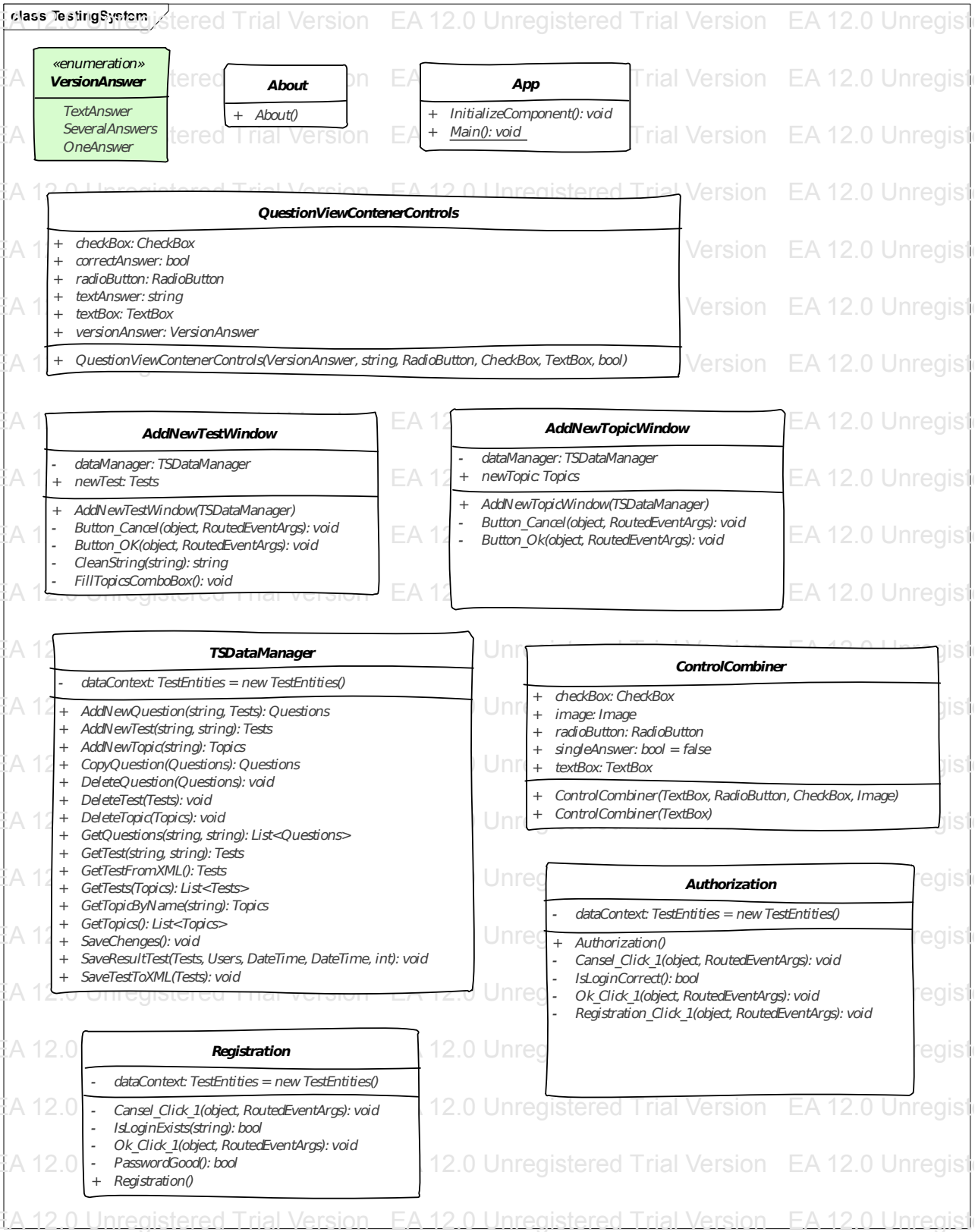
Для работы с базой данных MS SQL выбрана технология Entity Framework (EF). Это объектно-реляционный модуль сопоставления, позволяющий разработчикам .NET работать с реляционными данными с помощью объектов, специализированных для доменов. Это устраняет необходимость в написания большей части кода для доступа к данным, который обычно требуется разработчикам.

3. РАЗРАБОТКА СТРУКТУРЫ СИСТЕМЫ

3.1. Диаграмма классов

Для представления модели статической структуры программной системы используется диаграмма классов (рис. 3.1.).





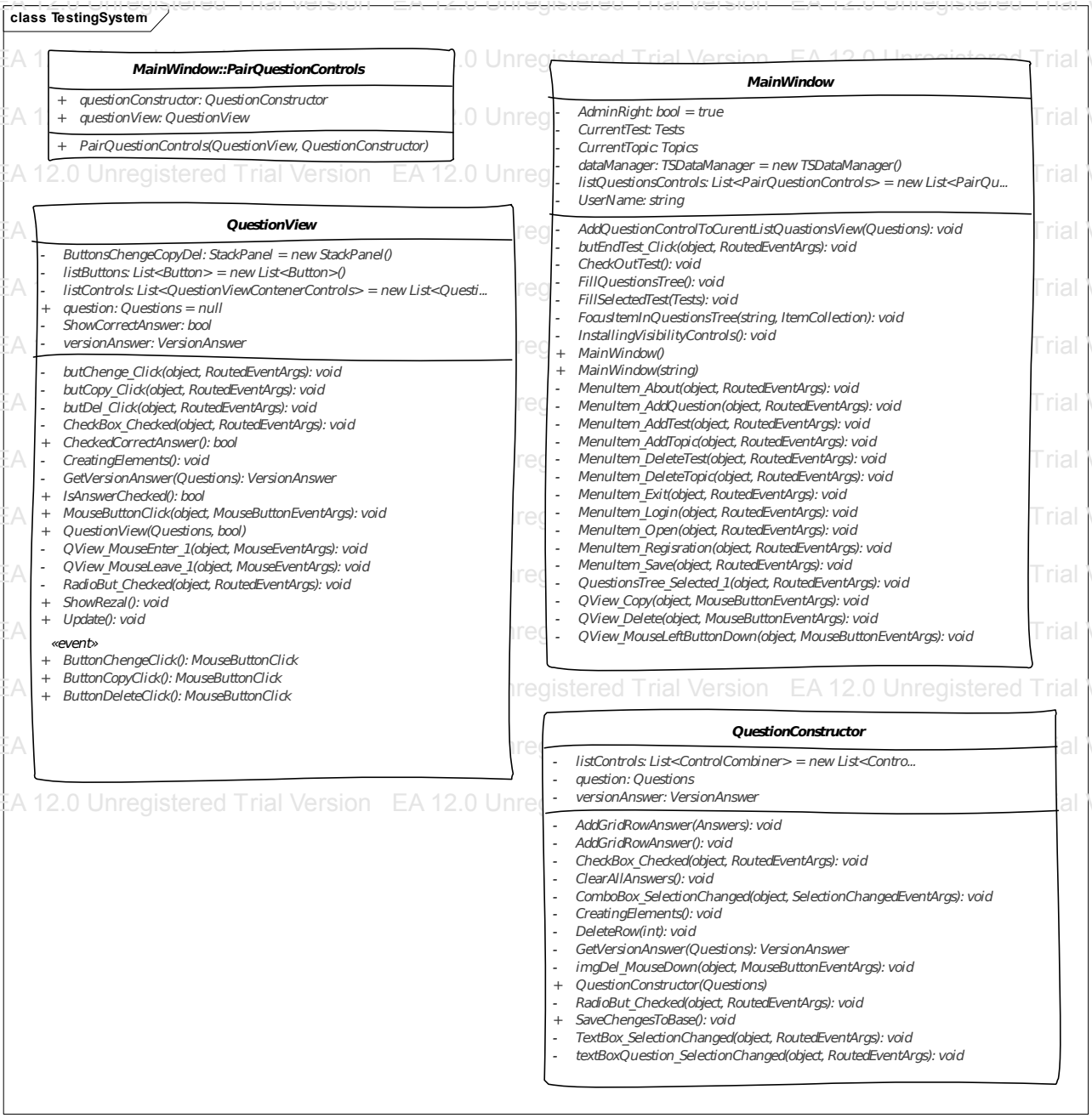


Рис. 3.1 Диаграммы классов

3.2 Описание классов

Перечисление **VersionAnswer** – содержит возможные виды ответов на вопрос (один правильный ответ, несколько правильных ответов, ответ текстом)

Класс **App** – окно входа в систему. С него запускается приложение. Переадресует запуск на окно Authorization

Класс **Authorization** – окно входа пользователя. Содержит поля ввода логина и пароля. Если данные авторизации верны, открывается окно MainWindow.

Класс **MainWindow** – главное окно программы «Тестирующая система». Содержит в себе список доступных тестов разбитых на темы. И окна где можно эти тесты редактировать или проходить.

Класс **About** – окно содержащее описание программы.

Класс **QuestionViewContenerControls** – класс предназначен для группировки всех необходимых элементов управления необходимых для отображения вопроса. Используется только в классе QuestionView.

Класс **QuestionView** – новый пользовательский элемент управления, созданный для отображения вопроса.

Класс **AddNewTestWindow** – окно, в котором можно добавить новый вопрос в текущем тесте.

Класс **AddNewTopicWindow** – окно, в котором можно добавить новую тему вопросов.

Класс **TSDataManager** – класс содержит в себе необходимый набор функций по работе с базой данных. Содержит в себе ссылку на контекст базы данных entity framework.

Класс **ControlCombiner** – класс предназначен для группировки всех необходимых элементов управления необходимых для редактирования вопроса. Используется только в классе QuestionConstructor.

Класс **QuestionConstructor** – новый пользовательский элемент управления, созданный для редактирования вопроса. А именно редактировать текст вопроса, изменять правильные и неправильные ответы.

Класс **Registration** – окно для регистрации новых пользователей.

Класс **PairQuestionControls** – класс содержит в себе пару QuestionConstructor и QuestionView. Служит для интерактивной смены элементов управления вопроса из режима просмотра в режим редактирования и обратно. Используется только в MainWindow.

4. РАЗРАБОТКА АЛГОРИТМОВ ФУНКЦИОНИРОВАНИЯ СИСТЕМЫ

4.1 Диаграммы деятельности и взаимодействия

Алгоритм функционирования системы «Тестирующая система» представлен на диаграмме деятельности (рис. 4.1.) и диаграмме последовательностей (рис.4.2.).

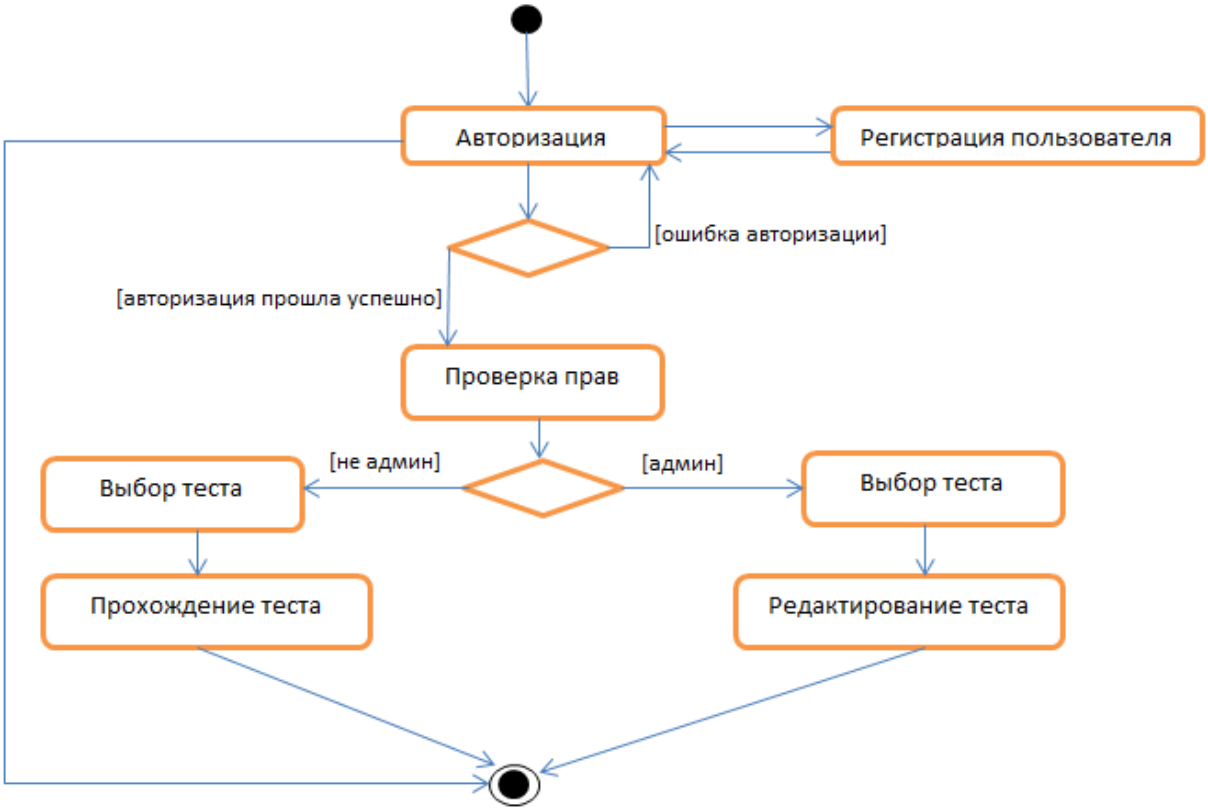


Рис. 4.1. Диаграмма деятельности системы «Тестирующая система»

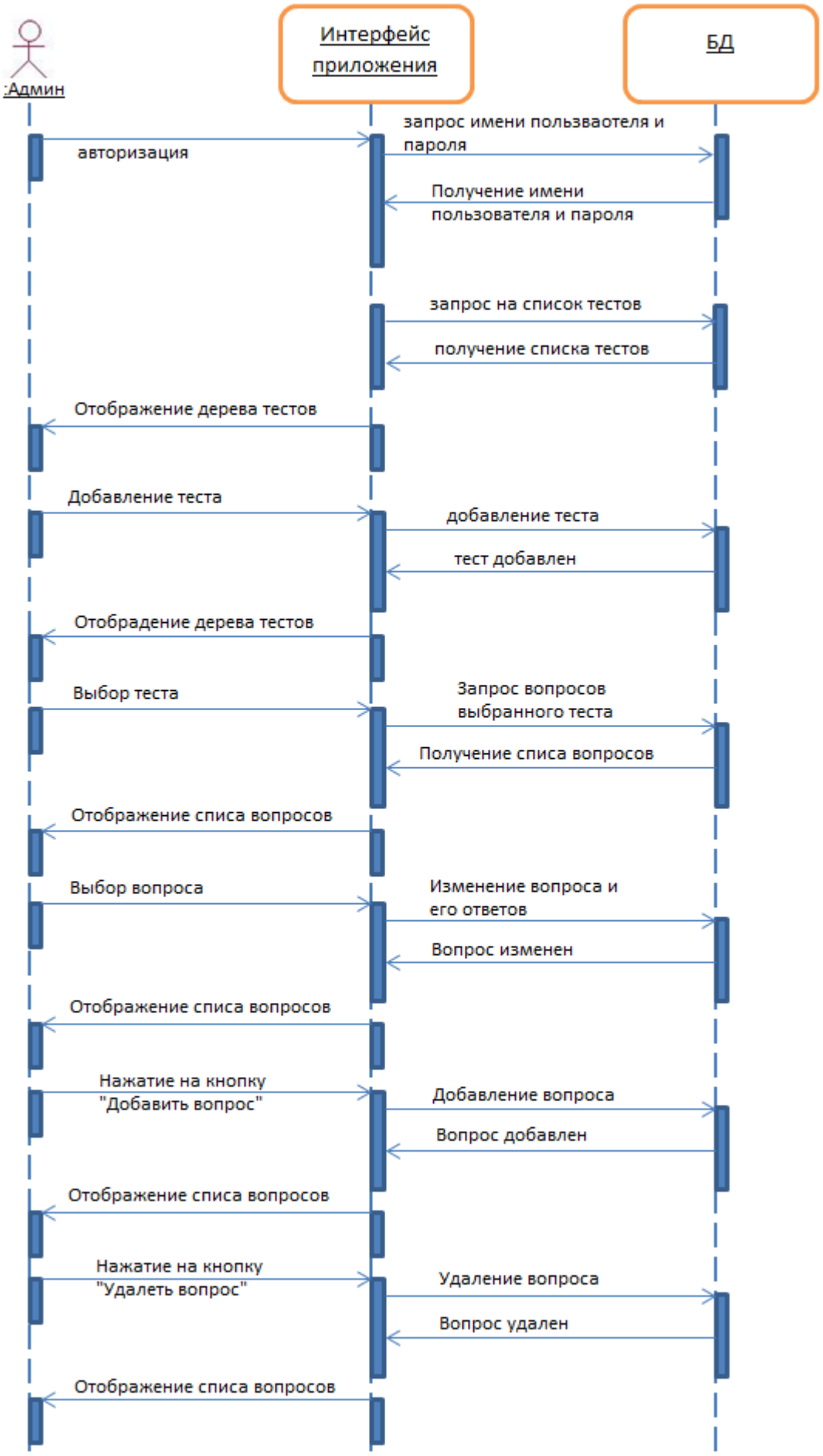


Рис. 4.4. Диаграмма последовательностей программы «Тестирующая система»

5. РАЗРАБОТКА БАЗЫ ДАННЫХ ДЛЯ СИСТЕМЫ

5.1 Диаграмма базы данных

Диаграмма базы данных, представленная на рисунке 5.1, описывает структуру базы данных приложения. Скрипт создания базы данных приводится в Приложении 2.

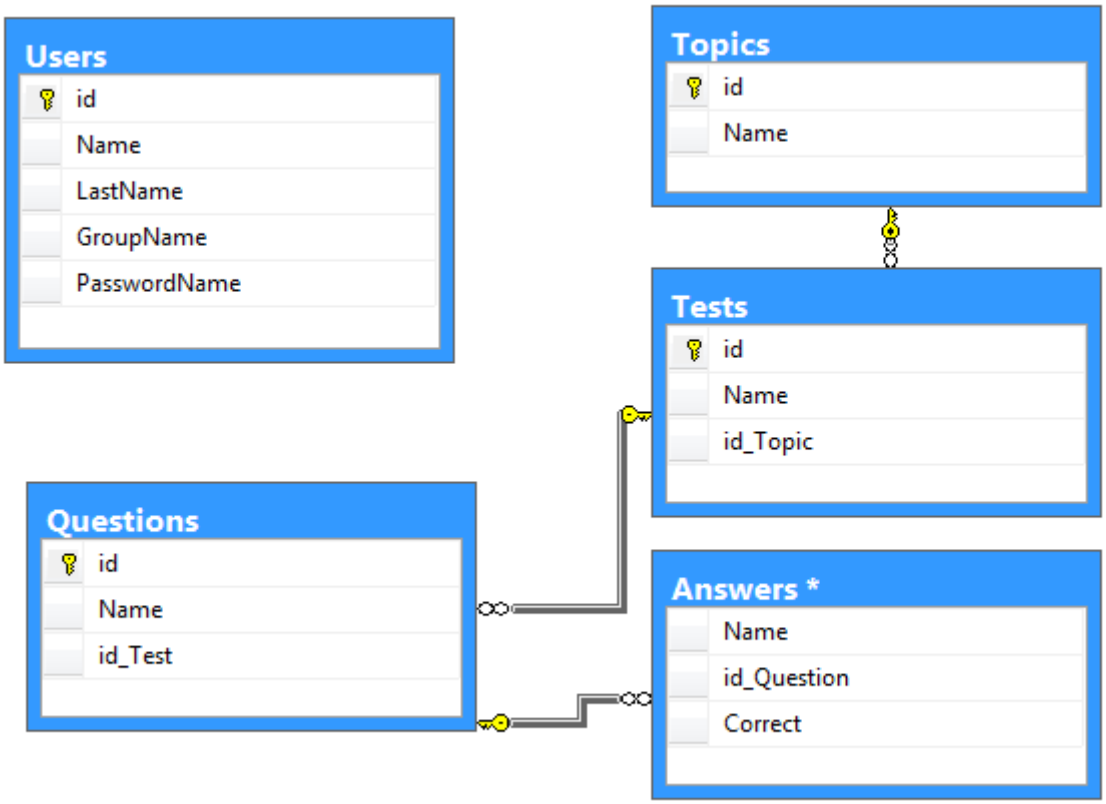


Рис. 5.1. Диаграмма базы данных

5.2 Описание таблиц базы данных

1) Users – таблица с данными о пользователях и их паролей.
2) Topics – таблица с названиями тем вопросов.
3) Tests – таблица с данными о тестах.
4) Questions – таблица с вопросами.
5) Answers – таблица с ответами вопросов.

6. РАЗРАБОТКА ИНТЕРФЕЙСА СИСТЕМЫ

Приложение «Тестирующая система» предназначено проверки знаний на выбранные темы.

Приложение написано на языке C#. Для создания интерфейса использовалась технология WPF. Windows Presentation Foundation (WPF) — это система следующего поколения для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем.

В основе WPF лежит векторная система отрисовки, не зависящая от разрешения и созданная с расчетом на возможности современного графического оборудования. WPF расширяет базовую систему полным набором функций разработки приложений, в том числе Язык XAML (Extensible Application Markup Language), элементами управления, привязкой данных, макетом, двухмерный и трехмерный графикой, анимацией, стилями, шаблонами, документами, мультимедиа, текстом и оформлением. WPF входит в состав Microsoft .NET Framework и позволяет создавать приложения, включающие другие элементы библиотеки классов .NET Framework.

Для облегчения работы с базой данных Microsoft SQL Server 2012 Express была выбрана технология Entity Framework 5.0.0. Это объектно-ориентированная технология доступа к данным, является object-relational mapping (ORM) решением для .NET Framework от Microsoft. Предоставляет возможность взаимодействия с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL.

7. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Для начала работы с приложением ввести логин пароль

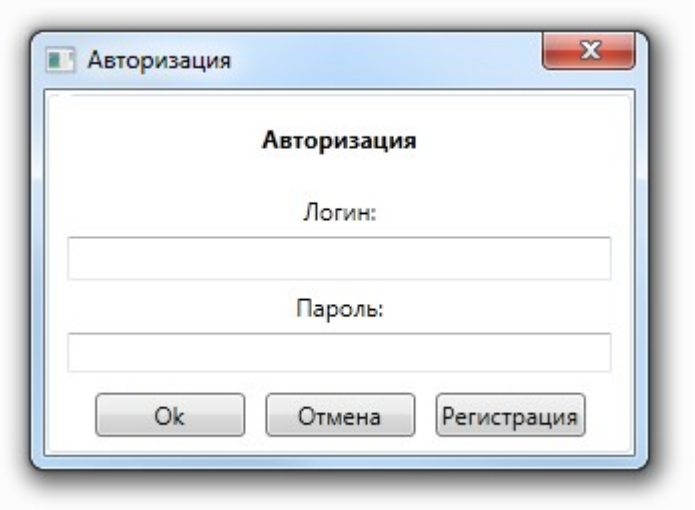


Рис. 7.1. Авторизация

Для создания новой учетной записи предназначена кнопка «Регистрация», при нажатии на которую, открывается окно регистрации (рис 7.2.).

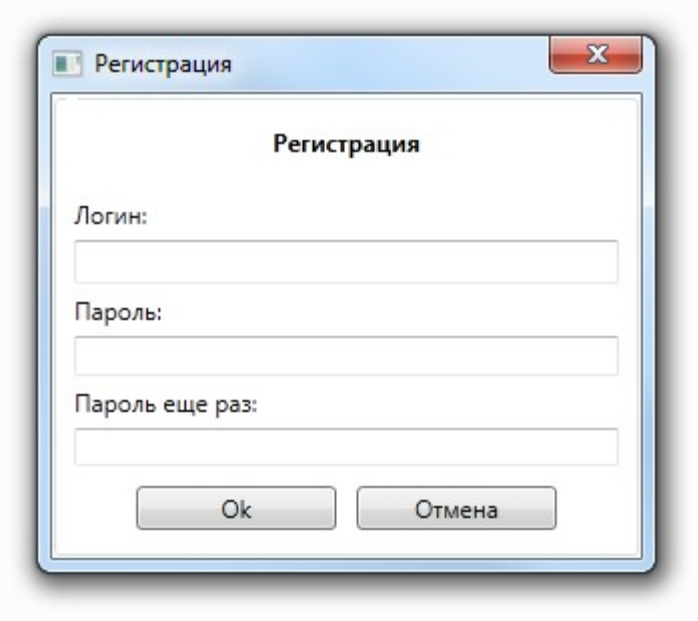


Рис. 7.2. Окно регистрации

Пароль должен быть не менее 8 символов. При успешной регистрации нового пользователя, управление возвращается окну авторизации. При успешной авторизации открывается главное окно системы (рис 7.3.)

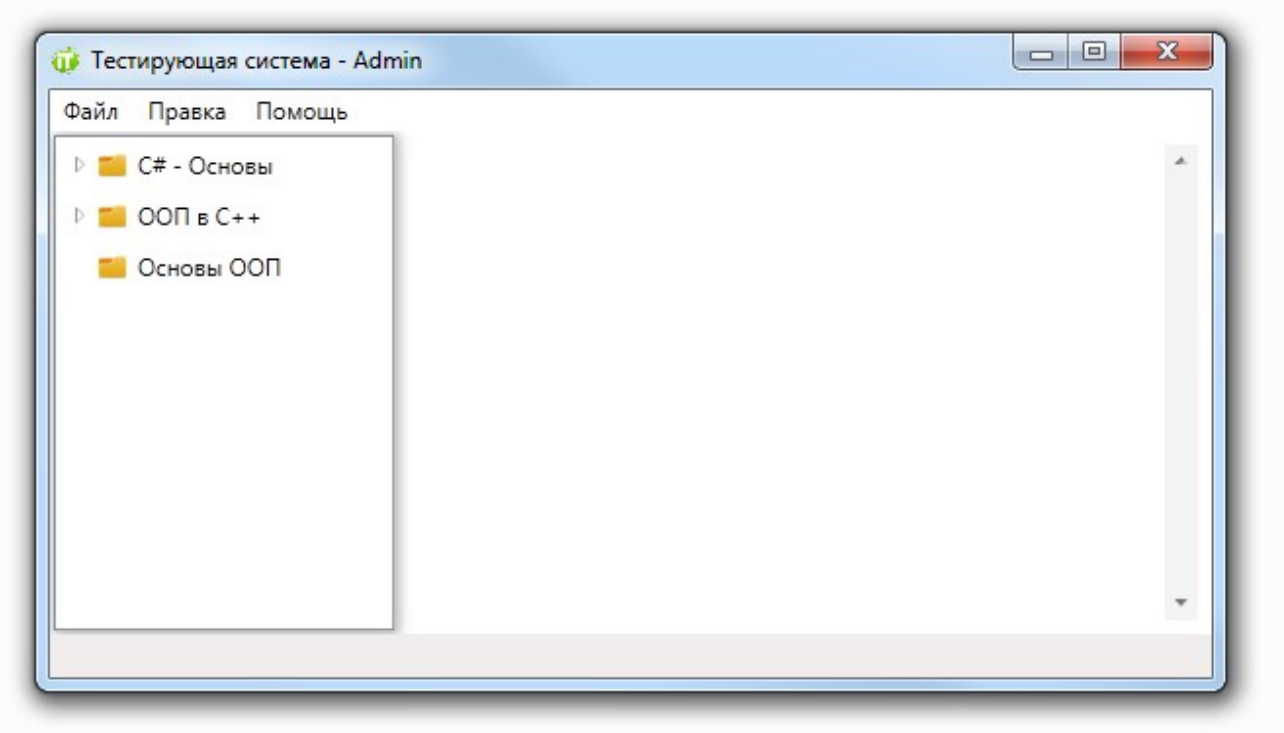


Рис. 7.3. Главное окно программы «Тестирующая система»

Подменю «Файл» содержит команды (Рис.7.4)

- Сохранить – сохраняет выбранный тест в xml файл
- Открыть – открывает ранее сохраненный тест из выбранного файла

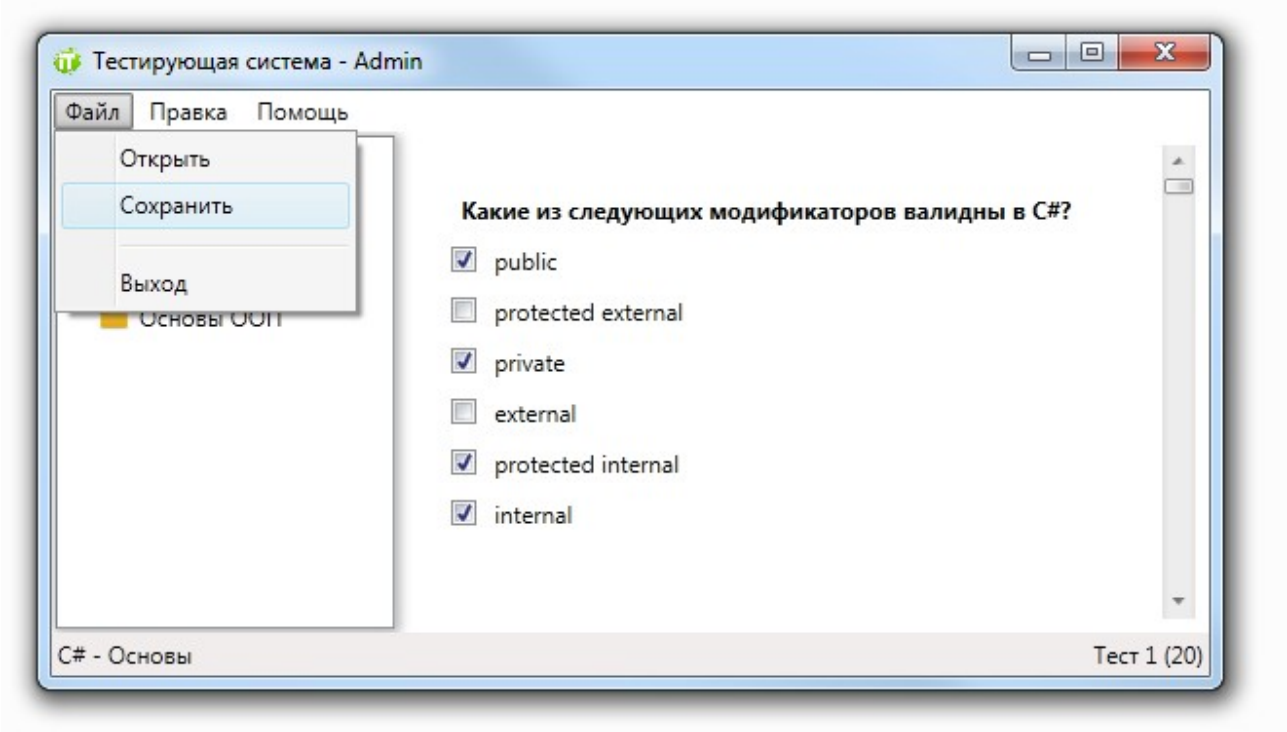


Рис. 7.4. Подменю «Файл»

Подменю «Правка» содержит команды, доступные для администратора (рис 7.5.)

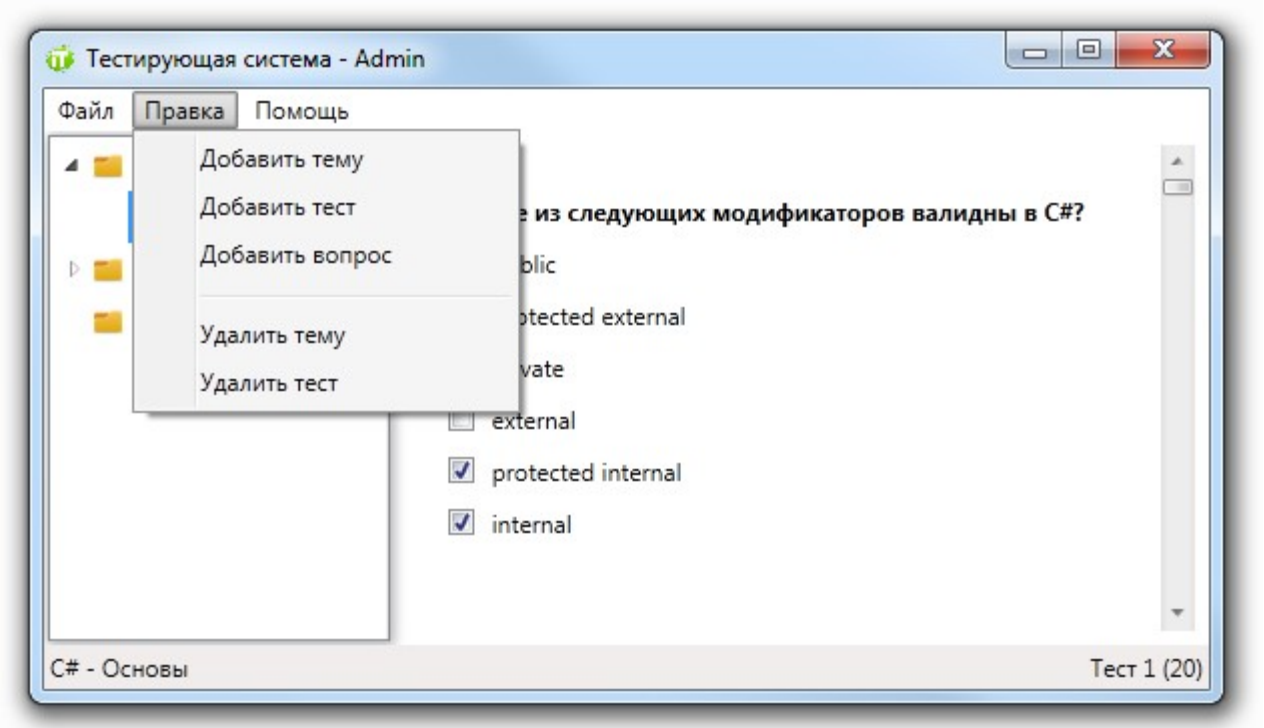


Рис. 7.5. Подменю «Правка»

В режиме администратора доступны инструменты по редактированию, добавлению и удалению вопросов. Так же возможно изменять ответы вопроса (Рис. 7.6.). Доступно три вида ответов: один верный ответ, несколько верных ответов и текст, когда ответ вводится пользователем с клавиатуры (ответ считается верным, если все символы введение с клавиатуры пользователем совпадают с ответом)

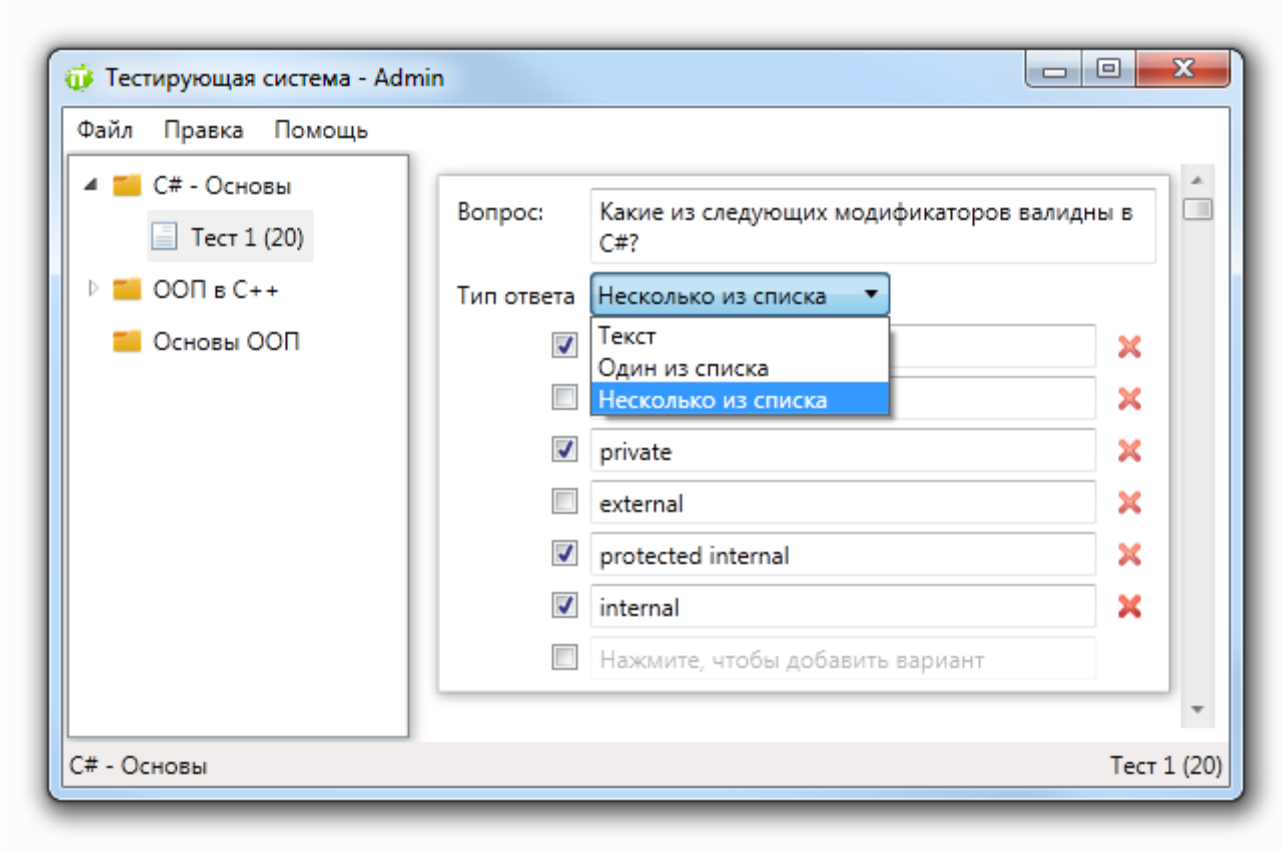


Рис. 7.6. Редактор вопроса

В подменю «Помощь» возможно посмотреть информацию о программе и перевойти в систему или добавить нового пользователя (Рис.7.7.).

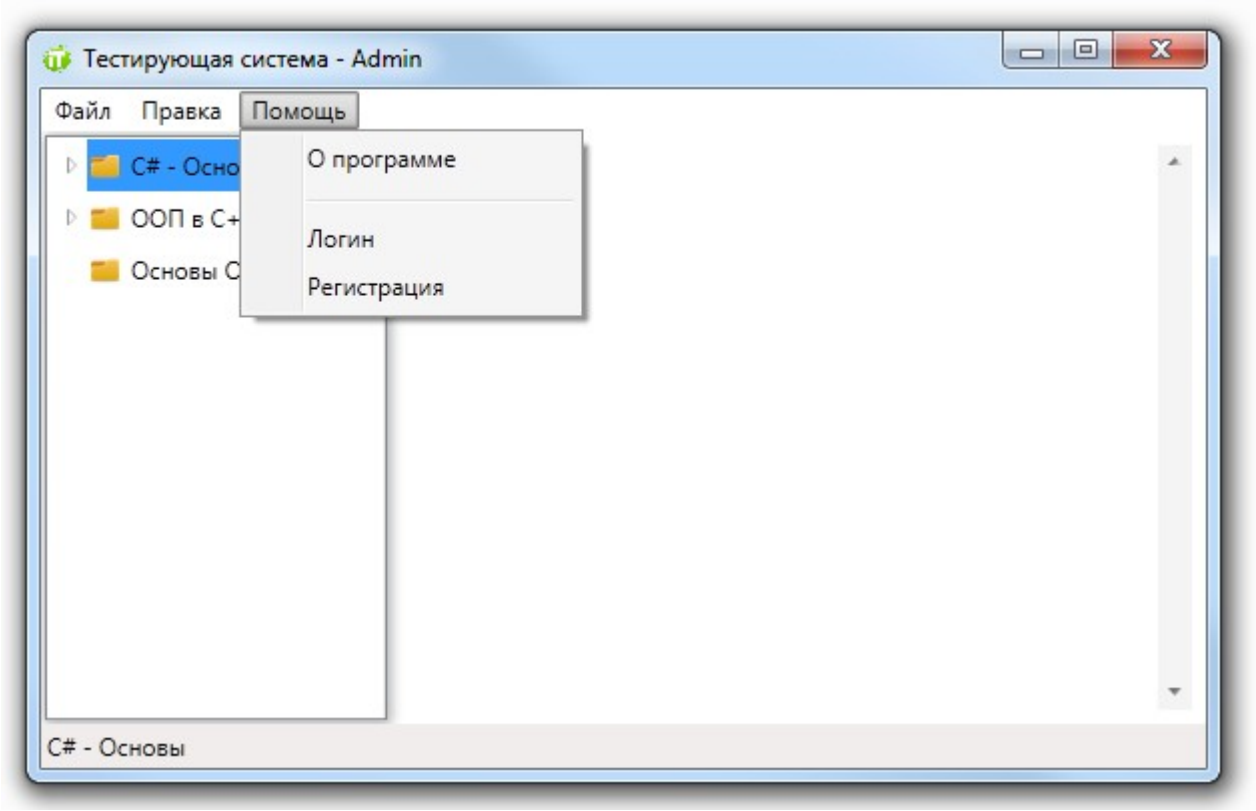


Рис. 7.7. Подменю «Помощь»

При входе в систему под учетной записью, не являющуюся администратором, подменю «Правка» и все остальные инструмент по редактированию тестов не доступны. Пользователю представлены вопросы с возможными ответами, которые ему предстоит выбрать. (Рис. 7.8.)

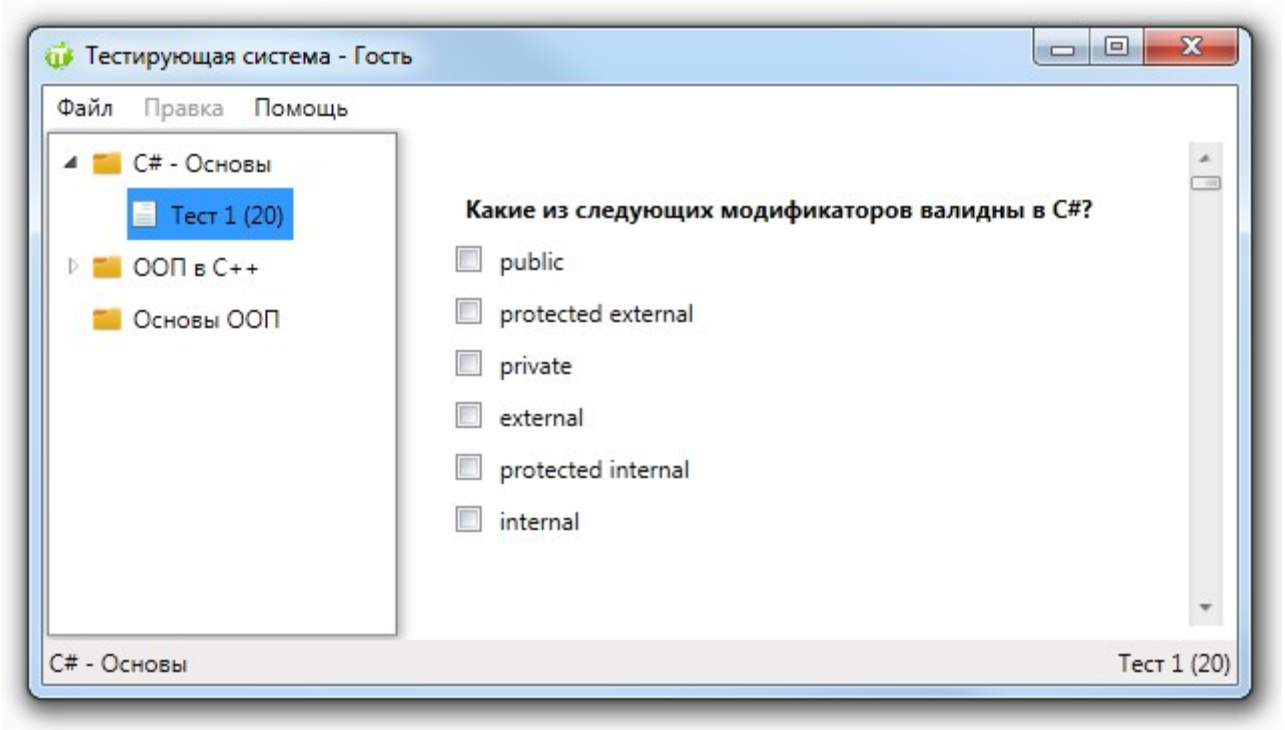


Рис. 7.8. Прохождение теста

При завершении теста, нужно нажать кнопку «Завершить тест» (Рис. 7.9.). Тест не будет завершён пока останется ходьбы один не отвеченный вопрос. По окончании выводится результат теста, в виде подсчета количества верных и неверных ответов.

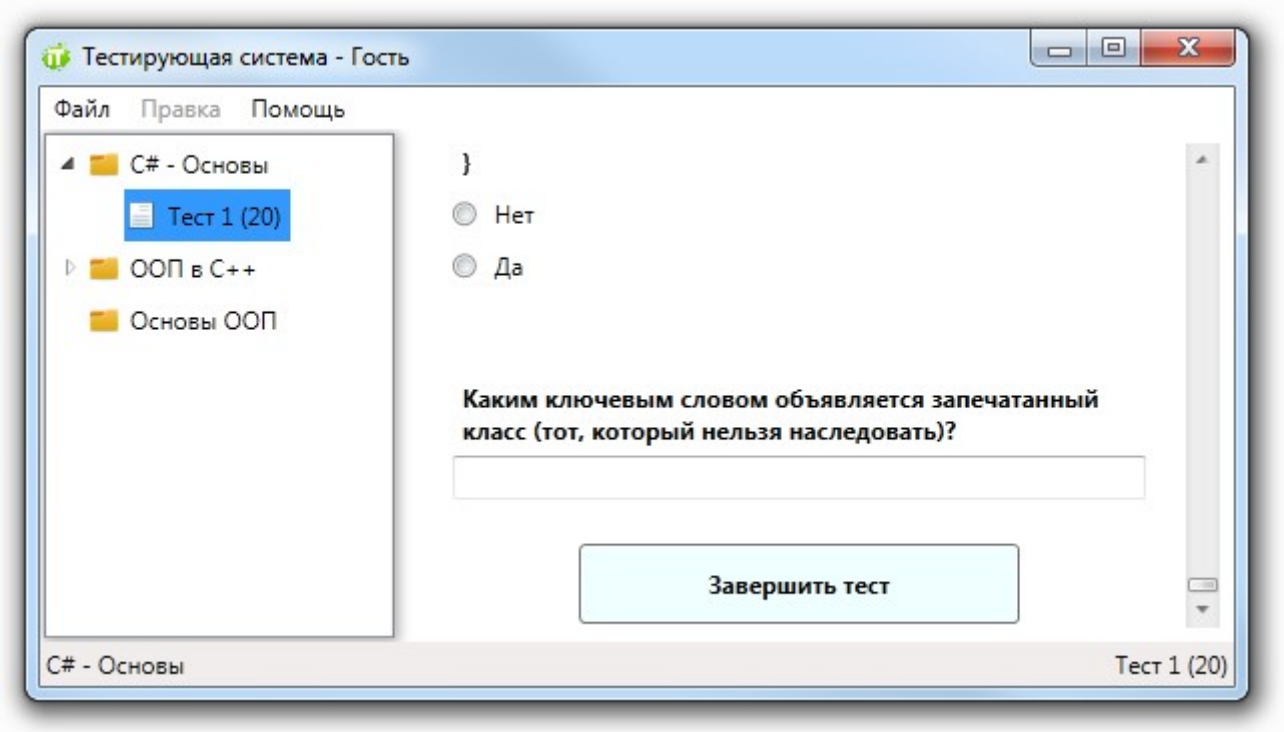


Рис. 7.9. Завершение теста

ВЫВОДЫ

Программа является инструментом для проверки знаний. Позволяет легко и быстро создавать тесты любой сложности и объема. При прохождении теста система сразу сообщит результат проверки знаний.

Предоставлена возможность экспорта тестов из базы в XML. Также возможно открыть внешний тест и пройти его в программе.

При разработке приложения были использованы платформа .NET Framework, язык программирования C#, технология взаимодействия с БД Entity Framework.

Простой и понятный интерфейс позволяет пользователю комфортно работать с приложением даже с минимальными знаниями, многие действия интуитивно понятны.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1) Julia Lerman - Programming Entity Framework, 2nd Edition 2010
2) Кристиан Нейгел - C# 2008 и платформа NET 3.5 для профессионалов, И.Д. Вильямс
3) Э. Троельсэн - Язык программирования C# 2008 и платформа .NET 3.5
4) Джесс Либерти - Programming C# / Программирование на C#, 2-е издание, Символ-Плюс
5) Рихтер Дж. - CLR via C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C# (Мастер-класс) – 2012
6) Мак-Дональд Мэтью - Windows Presentation Foundation в .NET 4 -2011
7) http://msdn.microsoft.com/ru-ru/library/ms123401.aspx
8) https://msdn.microsoft.com
9) www.codeproject.com
10) http://www.firststeps.ru/

ПРИЛОЖЕНИЕ 1. ЛИСТИНГ ПРОГРАММЫ

TSDataManager.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using TestingSystem.Data;
using System.Windows.Forms;
using System.Xml;

namespace TestingSystem
{
  public enum VersionAnswer { TextAnswer, SeveralAnswers, OneAnswer };


  public class TSDataManager
  {
    TestEntities dataContext = new TestEntities();

    public List<Topics> GetTopics()
    {
      List<Topics> listTopics = new List<Topics>();

      IQueryable<Topics> quary = dataContext.Topics.Select(t => t);
      foreach (Topics t in quary)
      {
        listTopics.Add(t);
      }

      return listTopics;
    }

    public List<Tests> GetTests(Topics topic)
    {
      List<Tests> listTests = new List<Tests>();

      IQueryable<Tests> quary = dataContext.Tests.Where(t => t.id_Topic == topic.id);
      foreach (Tests t in quary)
      {
        listTests.Add(t);
      }

      return listTests;
    }

    public List<Questions> GetQuestions(string topicName, string testName)
    {
      Topics topic = GetTopicByName(topicName);
      Tests test = null;
      IQueryable<Tests> quaryT = dataContext.Tests.Where(t => t.Name == testName && t.id_Topic == topic.id);
      foreach (Tests t in quaryT)
      {
        test = t;
        break;
      }

      List<Questions> listQuestions = new List<Questions>();
      IQueryable<Questions> quaryQ = dataContext.Questions.Where(q => q.id_Test == test.id);
      foreach (Questions question in quaryQ)
      {
        listQuestions.Add(question);
      }

      return listQuestions;
    }

    public Tests GetTest(string topicName, string testName)
    {
      Topics topic = GetTopicByName(topicName);
      Tests test = null;
      IQueryable<Tests> quaryT = dataContext.Tests.Where(t => t.Name == testName && t.id_Topic == topic.id);
      foreach (Tests t in quaryT)
      {
        test = t;
        break;
      }
```

```
    return test;
}

public Topics GetTopicByName(string TopicName)
{
    IQueryable<Topics> quary = dataContext.Topics.Where(t => t.Name == TopicName);
    foreach (Topics t in quary)
    {
        return t;
    }

    return null;
}

public Questions AddNewQuestion(string qName, Tests test)
{
    Questions newQuestion = new Questions();
    newQuestion.Name = qName;
    newQuestion.id_Test = test.id;

    dataContext.Questions.Add(newQuestion);
    dataContext.SaveChanges();

    return newQuestion;
}

public Topics AddNewTopic(string tName)
{
    if (tName == "")
        return null;

    IQueryable<Topics> quary = dataContext.Topics.Where(t => t.Name == tName);
    foreach (Topics t in quary)
    {
        return t;
    }

    Topics topic = new Topics();
    topic.Name = tName;
    dataContext.Topics.Add(topic);
    dataContext.SaveChanges();
    return topic;
}

public Tests AddNewTest(string testName, string topicName)
{
    if (testName == "" || topicName == "")
        return null;

    Topics topic = null;
    IQueryable<Topics> quaryTopic = dataContext.Topics.Where(t => t.Name == topicName);
    foreach (Topics t in quaryTopic)
    {
        topic = t;
        break;
    }

    IQueryable<Tests> quaryTest = dataContext.Tests.Where(t => t.Name == topicName && t.id_Topic == topic.id);
    foreach (Tests t in quaryTest)
    {
        return t;
    }

    Tests test = new Tests();
    test.Name = testName;
    test.id_Topic = topic.id;
    dataContext.Tests.Add(test);
    dataContext.SaveChanges();

    return test;
}

public void SaveChenges()
{
    dataContext.SaveChanges();
}

public Questions CopyQuestion(Questions question)
{
    Questions newQuestion = new Questions();
    newQuestion.id_Test = question.id_Test;
    newQuestion.Name = question.Name;
```

```
    dataContext.Questions.Add(newQuestion);
    dataContext.SaveChanges();

    foreach (Answers answer in question.Answers)
    {
        Answers newAnswer = new Answers();
        newAnswer.Correct     = answer.Correct;
        newAnswer.id_Question = newQuestion.id;
        newAnswer.Name        = answer.Name;
        dataContext.Answers.Add(newAnswer);
    }
    dataContext.SaveChanges();

    return newQuestion;
}

public void DeleteQuestion(Questions question)
{
    question.Answers.Clear();
    dataContext.Questions.Remove(question);
    dataContext.SaveChanges();
}

public void DeleteTopic(Topics topic)
{
    dataContext.Topics.Remove(topic);
    dataContext.SaveChanges();
}

public void DeleteTest(Tests test)
{
   while( test.Questions.Count != 0)
       DeleteQuestion(test.Questions.ElementAt(0));

    dataContext.SaveChanges();

    dataContext.Tests.Remove(test);
    dataContext.SaveChanges();
}

public void SaveTestToXML(Tests test)
{
    SaveFileDialog saveDlg = new SaveFileDialog();
    saveDlg.FileName = "Test";
    saveDlg.DefaultExt = "xml";
    saveDlg.Filter = "Файлы XML (*.xml)|*.xml";
    if (saveDlg.ShowDialog() != DialogResult.OK)
        return;

    XmlTextWriter writer = null;

    try
    {
        writer = new XmlTextWriter(saveDlg.FileName, System.Text.Encoding.Unicode);

        writer.WriteStartDocument();
        writer.WriteStartElement("Test");
        writer.WriteAttributeString("TestName", test.Name);
        writer.WriteAttributeString("TopicName", test.Topic.Name);

        writer.WriteStartElement("Questions");
        foreach (Questions q in test.Questions)
        {
            writer.WriteStartElement("Question");
            writer.WriteAttributeString("QuestionName", q.Name);
            foreach (Answers answer in q.Answers)
            {
                writer.WriteStartElement("Answer");
                writer.WriteAttributeString("AnswerName", answer.Name);
                writer.WriteAttributeString("Correct", answer.Correct.ToString());
                writer.WriteEndElement();
            }

            writer.WriteEndElement();
        }
        writer.WriteEndElement(); //Questions
        writer.WriteEndElement(); //Test
        writer.WriteEndDocument();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Ошибка: " + ex.Message);
    }
```

```
            finally
            {
                if (writer != null)
                    writer.Close();
            }
        }

        public Tests GetTestFromXML()
        {

            OpenFileDialog dlg = new OpenFileDialog();
            dlg.Filter = "Файлы XML (*.xml)|*.xml";
            if (dlg.ShowDialog() != DialogResult.OK)
                return null;

            XmlTextReader reader = null;
            Tests Test = new Tests();

            try
            {
                reader = new XmlTextReader(dlg.FileName);
                reader.WhitespaceHandling = WhitespaceHandling.None; // пропускаем пустые узлы

                while (reader.Read())
                {
                    if (reader.NodeType == XmlNodeType.Element)
                    {
                        if (reader.Name == "Questions")
                        {
                            while (reader.Read() && reader.Name == "Question")
                            {
                                Questions question = new Questions();
                                question.Name = reader.GetAttribute("QuestionName");

                                while (reader.Read() && reader.Name == "Answer")
                                {
                                    Answers answer = new Answers();
                                    answer.Name = reader.GetAttribute("AnswerName");
                                    answer.Correct = reader.GetAttribute("Correct") == "True" ? true : false;
                                    question.Answers.Add(answer);
                                }
                                Test.Questions.Add(question);
                            }
                        }
                    }
                }


            }
            catch (Exception ex)
            {
                MessageBox.Show("Ошибка: " + ex.Message);
            }
            finally
            {
                if (reader != null)
                    reader.Close();
            }

            return Test;
        }

        public void SaveResultTest(Tests test, Users user, DateTime tStart, DateTime tEnd, int rezalt)
        {
            Results rez = new Results();
            rez.DateTimeStart = tStart;
            rez.DateTimeEnd = tEnd;
            rez.id_Test = test.id;
            rez.id_User = user.id;
            rez.Rezalt = rezalt;

            dataContext.Results.Add(rez);
            dataContext.SaveChanges();
        }
    }
}
```

QuestionConstructor.xaml.cs

```
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 TestingSystem.Data;

namespace TestingSystem
{
  class ControlCombiner
  {
    public TextBox textBox;
    public RadioButton radioButton;
    public CheckBox checkBox;
    public Image image;
    public bool singleAnswer = false;

    public ControlCombiner(TextBox tBox, RadioButton rButton, CheckBox cBox, Image im)
    {
      textBox     = tBox;
      radioButton = rButton;
      checkBox    = cBox;
      image       = im;
    }

    public ControlCombiner(TextBox tBox)
    {
      textBox        = tBox;
      radioButton    = null;
      checkBox       = null;
      singleAnswer   = true;
    }

  }

  public partial class QuestionConstructor : UserControl
  {

    #region Поля
    Questions question;
    VersionAnswer versionAnswer;
    List<ControlCombiner> listControls         = new List<ControlCombiner>();
    #endregion


    public QuestionConstructor(Questions q)
    {
      question = q;
      versionAnswer = GetVersionAnswer(q);
      InitializeComponent();
      CreatingElements();
    }


    #region Общие функции

    void CreatingElements()
    {

      Thickness margin_2_2_2_2 = new Thickness();
      margin_2_2_2_2.Right = 2;
      margin_2_2_2_2.Left = 2;
      margin_2_2_2_2.Top = 2;
      margin_2_2_2_2.Bottom = 2;

      // Define the Columns
      ColumnDefinition colDef1 = new ColumnDefinition();
      colDef1.Width = GridLength.Auto;
      ColumnDefinition colDef2 = new ColumnDefinition();
      ColumnDefinition colDef3 = new ColumnDefinition();
      colDef3.MinWidth = 30;
      colDef3.Width = GridLength.Auto;
      QGrid.ColumnDefinitions.Add(colDef1);
      QGrid.ColumnDefinitions.Add(colDef2);
      QGrid.ColumnDefinitions.Add(colDef3);
```

```
    // Первая строка
    RowDefinition rowDef1 = new RowDefinition();
    rowDef1.Height = GridLength.Auto;
    QGrid.RowDefinitions.Add(rowDef1);

    Label label1 = new Label();
    label1.Content = "Вопрос:";
    //label1.FontWeight = FontWeights.Bold;
    Grid.SetRow(label1, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(label1, 0);
    QGrid.Children.Add(label1);

    TextBox textBox1 = new TextBox();
    textBox1.TextWrapping = TextWrapping.Wrap;
    textBox1.Margin = margin_2_2_2_2;
    textBox1.AcceptsReturn = true;
    textBox1.Text = question.Name;
    textBox1.SelectionChanged += textBoxQuestion_SelectionChanged;
    Grid.SetRow(textBox1, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(textBox1, 1);
    Grid.SetColumnSpan(textBox1, 2);
    QGrid.Children.Add(textBox1);


    // Вторая строка
    RowDefinition rowDef2 = new RowDefinition();
    rowDef2.Height = GridLength.Auto;
    QGrid.RowDefinitions.Add(rowDef2);

    Label label2 = new Label();
    label2.Content = "Тип ответа";
    //label2.FontWeight = FontWeights.Bold;
    Grid.SetRow(label2, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(label2, 0);
    QGrid.Children.Add(label2);


    ComboBox comboBox1 = new ComboBox();
    comboBox1.Margin = margin_2_2_2_2;
    comboBox1.MinWidth = 150;
    comboBox1.Items.Add("Текст");
    comboBox1.Items.Add("Один из списка");
    comboBox1.Items.Add("Несколько из списка");
    if (versionAnswer == VersionAnswer.TextAnswer)
        comboBox1.SelectedIndex = 0;
    else if(versionAnswer == VersionAnswer.OneAnswer)
        comboBox1.SelectedIndex = 1;
    else if (versionAnswer == VersionAnswer.SeveralAnswers)
        comboBox1.SelectedIndex = 2;

    comboBox1.SelectionChanged += ComboBox_SelectionChanged;
    WrapPanel wrap = new WrapPanel();
    wrap.Children.Add(comboBox1);
    Grid.SetRow(wrap, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(wrap, 1);
    QGrid.Children.Add(wrap);

    foreach(Answers answer in question.Answers)
        AddGridRowAnswer(answer);

    AddGridRowAnswer();
}

void AddGridRowAnswer(Answers answer)
{
    Thickness margin_2_2_2_2 = new Thickness();
    margin_2_2_2_2.Right = 2;
    margin_2_2_2_2.Left = 2;
    margin_2_2_2_2.Top = 2;
    margin_2_2_2_2.Bottom = 2;

    if (versionAnswer == VersionAnswer.TextAnswer)
    {
        // Третья строка
        RowDefinition rowDef3 = new RowDefinition();
        rowDef3.Height = GridLength.Auto;
        QGrid.RowDefinitions.Add(rowDef3);

        TextBox SingleAnswerTextBox = new TextBox();
        SingleAnswerTextBox.TextWrapping = TextWrapping.Wrap;
        SingleAnswerTextBox.Margin = margin_2_2_2_2;
        SingleAnswerTextBox.AcceptsReturn = true;
        SingleAnswerTextBox.Text = answer.Name;
```

```
    SingleAnswerTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
    Grid.SetRow(SingleAnswerTextBox, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(SingleAnswerTextBox, 1);
    QGrid.Children.Add(SingleAnswerTextBox);

    ControlCombiner controlCombiner = new ControlCombiner(SingleAnswerTextBox);
    listControls.Add(controlCombiner);
}
else if (versionAnswer == VersionAnswer.OneAnswer)
{

    // Третья строка
    RowDefinition rowDef3 = new RowDefinition();
    rowDef3.Height = GridLength.Auto;
    QGrid.RowDefinitions.Add(rowDef3);



    RadioButton NewRadioBut = new RadioButton();
    Thickness marginRadioButton = new Thickness();
    marginRadioButton.Right = 4;
    NewRadioBut.Margin = marginRadioButton;
    NewRadioBut.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
    NewRadioBut.VerticalAlignment = System.Windows.VerticalAlignment.Center;
    NewRadioBut.IsChecked = answer.Correct;
    NewRadioBut.Checked += RadioBut_Checked;
    Grid.SetRow(NewRadioBut, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewRadioBut, 0);
    QGrid.Children.Add(NewRadioBut);

    TextBox NewTextBox = new TextBox();
    NewTextBox.TextWrapping = TextWrapping.Wrap;
    NewTextBox.Margin = margin_2_2_2_2;
    NewTextBox.Text = answer.Name;
    NewTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
    Grid.SetRow(NewTextBox, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewTextBox, 1);
    QGrid.Children.Add(NewTextBox);

    Image imgDel = new Image();
    imgDel.Source = new BitmapImage(new Uri("../images/Удалить.png", UriKind.Relative));
    imgDel.MaxWidth = 16;
    imgDel.MaxHeight = 16;
    imgDel.MouseDown += imgDel_MouseDown;
    Grid.SetRow(imgDel, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(imgDel, 2);
    QGrid.Children.Add(imgDel);

    ControlCombiner controlCombiner = new ControlCombiner(NewTextBox, NewRadioBut, null, imgDel);
    listControls.Add(controlCombiner);
}
else if (versionAnswer == VersionAnswer.SeveralAnswers)
{
    // Третья строка
    RowDefinition rowDef3 = new RowDefinition();
    rowDef3.Height = GridLength.Auto;
    QGrid.RowDefinitions.Add(rowDef3);


    CheckBox NewCheckBox = new CheckBox();
    Thickness marginRadioButton = new Thickness();
    marginRadioButton.Right = 4;
    NewCheckBox.Margin = marginRadioButton;
    NewCheckBox.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
    NewCheckBox.VerticalAlignment = System.Windows.VerticalAlignment.Center;
    NewCheckBox.IsChecked = answer.Correct;
    NewCheckBox.Checked += CheckBox_Checked;
    Grid.SetRow(NewCheckBox, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewCheckBox, 0);
    QGrid.Children.Add(NewCheckBox);

    TextBox NewTextBox = new TextBox();
    NewTextBox.TextWrapping = TextWrapping.Wrap;
    NewTextBox.Margin = margin_2_2_2_2;
    NewTextBox.Text = answer.Name;
    NewTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
    Grid.SetRow(NewTextBox, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewTextBox, 1);
    QGrid.Children.Add(NewTextBox);

    Image imgDel = new Image();
    imgDel.Source = new BitmapImage(new Uri("../images/Удалить.png", UriKind.Relative));
    imgDel.MaxWidth = 16;
```

```
            imgDel.MaxHeight = 16;
            imgDel.MouseDown += imgDel_MouseDown;
            Grid.SetRow(imgDel, QGrid.RowDefinitions.Count - 1);
            Grid.SetColumn(imgDel, 2);
            QGrid.Children.Add(imgDel);

            ControlCombiner controlCombiner = new ControlCombiner(NewTextBox, null, NewCheckBox, imgDel);
            listControls.Add(controlCombiner);
        }

        Grid.SetRowSpan(ShadowBorder, QGrid.RowDefinitions.Count);
        Grid.SetColumnSpan(ShadowBorder, QGrid.RowDefinitions.Count);
    }

    void AddGridRowAnswer()
    {
        string AnswerName = "Вариант 1";
        Thickness margin_2_2_2_2 = new Thickness();
        margin_2_2_2_2.Right = 2;
        margin_2_2_2_2.Left = 2;
        margin_2_2_2_2.Top = 2;
        margin_2_2_2_2.Bottom = 2;

        if (versionAnswer == VersionAnswer.TextAnswer)
        {
            if (QGrid.RowDefinitions.Count == 2)
            {
                // Третья строка
                RowDefinition rowDef3 = new RowDefinition();
                rowDef3.Height = GridLength.Auto;
                QGrid.RowDefinitions.Add(rowDef3);

                TextBox SingleAnswerTextBox = new TextBox();
                SingleAnswerTextBox.TextWrapping = TextWrapping.Wrap;
                SingleAnswerTextBox.Margin = margin_2_2_2_2;
                SingleAnswerTextBox.AcceptsReturn = true;
                SingleAnswerTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
                Grid.SetRow(SingleAnswerTextBox, QGrid.RowDefinitions.Count - 1);
                Grid.SetColumn(SingleAnswerTextBox, 1);
                QGrid.Children.Add(SingleAnswerTextBox);

                ControlCombiner controlCombiner = new ControlCombiner(SingleAnswerTextBox);
                listControls.Add(controlCombiner);
            }
        }
        else if (versionAnswer == VersionAnswer.OneAnswer)
        {

            if (QGrid.RowDefinitions.Count == 2)
            {
                // Третья строка
                RowDefinition rowDef3 = new RowDefinition();
                rowDef3.Height = GridLength.Auto;
                QGrid.RowDefinitions.Add(rowDef3);


                RadioButton NewRadioBut = new RadioButton();
                Thickness marginRadioButton = new Thickness();
                marginRadioButton.Right = 4;
                NewRadioBut.Margin = marginRadioButton;
                NewRadioBut.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
                NewRadioBut.VerticalAlignment = System.Windows.VerticalAlignment.Center;
                NewRadioBut.Checked += RadioBut_Checked;
                Grid.SetRow(NewRadioBut, QGrid.RowDefinitions.Count - 1);
                Grid.SetColumn(NewRadioBut, 0);
                QGrid.Children.Add(NewRadioBut);

                TextBox NewTextBox = new TextBox();
                NewTextBox.TextWrapping = TextWrapping.Wrap;
                NewTextBox.Margin = margin_2_2_2_2;
                NewTextBox.Text = AnswerName;
                NewTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
                Grid.SetRow(NewTextBox, QGrid.RowDefinitions.Count - 1);
                Grid.SetColumn(NewTextBox, 1);
                QGrid.Children.Add(NewTextBox);

                Image imgDel = new Image();
                imgDel.Source = new BitmapImage(new Uri("../images/Удалить.png", UriKind.Relative));
                imgDel.MaxWidth = 16;
                imgDel.MaxHeight = 16;
                imgDel.MouseDown += imgDel_MouseDown;
                Grid.SetRow(imgDel, QGrid.RowDefinitions.Count - 1);
                Grid.SetColumn(imgDel, 2);
```

```
            QGrid.Children.Add(imgDel);

            ControlCombiner controlCombiner = new ControlCombiner(NewTextBox, NewRadioBut, null, imgDel);
            listControls.Add(controlCombiner);

            // Четвертая строка
            RowDefinition rowDef4 = new RowDefinition();
            rowDef4.Height = GridLength.Auto;
            QGrid.RowDefinitions.Add(rowDef4);

            RadioButton NewRadioBut2 = new RadioButton();
            Thickness marginRadioButton2 = new Thickness();
            marginRadioButton2.Right = 4;
            NewRadioBut2.Margin = marginRadioButton;
            NewRadioBut2.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
            NewRadioBut2.VerticalAlignment = System.Windows.VerticalAlignment.Center;
            NewRadioBut2.Tag = "Last";
            NewRadioBut2.Checked += RadioBut_Checked;
            Grid.SetRow(NewRadioBut2, QGrid.RowDefinitions.Count - 1);
            Grid.SetColumn(NewRadioBut2, 0);
            QGrid.Children.Add(NewRadioBut2);

            TextBox NewTextBox2 = new TextBox();
            NewTextBox2.TextWrapping = TextWrapping.Wrap;
            NewTextBox2.Margin = margin_2_2_2_2;
            NewTextBox2.Opacity = 0.3;
            NewTextBox2.Text = "Нажмите, чтобы добавить вариант";
            NewTextBox2.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
            Grid.SetRow(NewTextBox2, QGrid.RowDefinitions.Count - 1);
            Grid.SetColumn(NewTextBox2, 1);
            QGrid.Children.Add(NewTextBox2);

            listControls.Add(new ControlCombiner(NewTextBox2, NewRadioBut2, null, null));
        }
        else
        {
            Image imgDel = new Image();
            imgDel.Source = new BitmapImage(new Uri("../images/Удалить.png", UriKind.Relative));
            imgDel.MaxWidth = 16;
            imgDel.MaxHeight = 16;
            imgDel.MouseDown += imgDel_MouseDown;
            Grid.SetRow(imgDel, QGrid.RowDefinitions.Count - 1);
            Grid.SetColumn(imgDel, 2);
            QGrid.Children.Add(imgDel);

            RowDefinition NewRow = new RowDefinition();
            NewRow.Height = GridLength.Auto;
            QGrid.RowDefinitions.Add(NewRow);
            int LastIndexRow = QGrid.RowDefinitions.Count - 1;

            RadioButton _NewRadioBut = new RadioButton();
             Thickness _marginRadioButton = new Thickness();
            _marginRadioButton.Right = 4;
            _NewRadioBut.Margin = _marginRadioButton;
            _NewRadioBut.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
            _NewRadioBut.VerticalAlignment = System.Windows.VerticalAlignment.Center;
            _NewRadioBut.Tag = "Last";
            _NewRadioBut.Checked += RadioBut_Checked;
            Grid.SetRow(_NewRadioBut, LastIndexRow);
            Grid.SetColumn(_NewRadioBut, 0);
            QGrid.Children.Add(_NewRadioBut);


            TextBox _NewTextBox = new TextBox();
            Thickness marginText = new Thickness();
            marginText.Right = 2;
            marginText.Left = 2;
            marginText.Top = 2;
            marginText.Bottom = 2;
            _NewTextBox.TextWrapping = TextWrapping.Wrap;
            _NewTextBox.Margin = marginText;
            _NewTextBox.Opacity = 0.3;
            _NewTextBox.Text = "Нажмите, чтобы добавить вариант";
            _NewTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
            Grid.SetRow(_NewTextBox, LastIndexRow);
            Grid.SetColumn(_NewTextBox, 1);
            QGrid.Children.Add(_NewTextBox);

            listControls.Add(new ControlCombiner(_NewTextBox, _NewRadioBut, null, imgDel));
        }
    }
    else if (versionAnswer == VersionAnswer.SeveralAnswers)
    {
```

```
if (QGrid.RowDefinitions.Count == 2)
{
    // Третья строка
    RowDefinition rowDef3 = new RowDefinition();
    rowDef3.Height = GridLength.Auto;
    QGrid.RowDefinitions.Add(rowDef3);


    CheckBox NewCheckBox = new CheckBox();
    Thickness marginRadioButton = new Thickness();
    marginRadioButton.Right = 4;
    NewCheckBox.Margin = marginRadioButton;
    NewCheckBox.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
    NewCheckBox.VerticalAlignment = System.Windows.VerticalAlignment.Center;
    NewCheckBox.Checked += CheckBox_Checked;
    Grid.SetRow(NewCheckBox, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewCheckBox, 0);
    QGrid.Children.Add(NewCheckBox);

    TextBox NewTextBox = new TextBox();
    NewTextBox.TextWrapping = TextWrapping.Wrap;
    NewTextBox.Margin = margin_2_2_2_2;
    NewTextBox.Text = AnswerName;
    NewTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
    Grid.SetRow(NewTextBox, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewTextBox, 1);
    QGrid.Children.Add(NewTextBox);

    Image imgDel = new Image();
    imgDel.Source = new BitmapImage(new Uri("../images/Удалить.png", UriKind.Relative));
    imgDel.MaxWidth = 16;
    imgDel.MaxHeight = 16;
    imgDel.MouseDown += imgDel_MouseDown;
    Grid.SetRow(imgDel, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(imgDel, 2);
    QGrid.Children.Add(imgDel);

    listControls.Add(new ControlCombiner(NewTextBox, null, NewCheckBox, imgDel));


    // Четвертая строка
    RowDefinition rowDef4 = new RowDefinition();
    rowDef4.Height = GridLength.Auto;
    QGrid.RowDefinitions.Add(rowDef4);

    CheckBox NewCheckBox2 = new CheckBox();
    Thickness marginRadioButton2 = new Thickness();
    marginRadioButton2.Right = 4;
    NewCheckBox2.Margin = marginRadioButton;
    NewCheckBox2.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
    NewCheckBox2.VerticalAlignment = System.Windows.VerticalAlignment.Center;
    NewCheckBox2.Tag = "Last";
    NewCheckBox2.Checked += CheckBox_Checked;
    Grid.SetRow(NewCheckBox2, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewCheckBox2, 0);
    QGrid.Children.Add(NewCheckBox2);

    TextBox NewTextBox2 = new TextBox();
    NewTextBox2.TextWrapping = TextWrapping.Wrap;
    NewTextBox2.Margin = margin_2_2_2_2;
    NewTextBox2.Opacity = 0.3;
    NewTextBox2.Text = "Нажмите, чтобы добавить вариант";
    NewTextBox2.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
    Grid.SetRow(NewTextBox2, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(NewTextBox2, 1);
    QGrid.Children.Add(NewTextBox2);

    listControls.Add(new ControlCombiner(NewTextBox2, null, NewCheckBox2, null));

}
else
{
    Image imgDel = new Image();
    imgDel.Source = new BitmapImage(new Uri("../images/Удалить.png", UriKind.Relative));
    imgDel.MaxWidth = 16;
    imgDel.MaxHeight = 16;
    imgDel.MouseDown += imgDel_MouseDown;
    Grid.SetRow(imgDel, QGrid.RowDefinitions.Count - 1);
    Grid.SetColumn(imgDel, 2);
    QGrid.Children.Add(imgDel);

    RowDefinition NewRow = new RowDefinition();
```

```
            NewRow.Height = GridLength.Auto;
            QGrid.RowDefinitions.Add(NewRow);
            int LastIndexRow = QGrid.RowDefinitions.Count - 1;

            CheckBox _NewCheckBox = new CheckBox();
            Thickness _marginRadioButton = new Thickness();
            _marginRadioButton.Right = 4;
            _NewCheckBox.Margin = _marginRadioButton;
            _NewCheckBox.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
            _NewCheckBox.VerticalAlignment = System.Windows.VerticalAlignment.Center;
            _NewCheckBox.Tag = "Last";
            _NewCheckBox.Checked += CheckBox_Checked;
            Grid.SetRow(_NewCheckBox, LastIndexRow);
            Grid.SetColumn(_NewCheckBox, 0);
            QGrid.Children.Add(_NewCheckBox);


            TextBox _NewTextBox = new TextBox();
            Thickness marginText = new Thickness();
            marginText.Right = 2;
            marginText.Left = 2;
            marginText.Top = 2;
            marginText.Bottom = 2;
            _NewTextBox.TextWrapping = TextWrapping.Wrap;
            _NewTextBox.Margin = marginText;
            _NewTextBox.Opacity = 0.3;
            _NewTextBox.Text = "Нажмите, чтобы добавить вариант";
            _NewTextBox.SelectionChanged += new RoutedEventHandler(TextBox_SelectionChanged);
            Grid.SetRow(_NewTextBox, LastIndexRow);
            Grid.SetColumn(_NewTextBox, 1);
            QGrid.Children.Add(_NewTextBox);

            listControls.Add(new ControlCombiner(_NewTextBox, null, _NewCheckBox, imgDel));
        }
    }

    Grid.SetRowSpan(ShadowBorder, QGrid.RowDefinitions.Count);
    Grid.SetColumnSpan(ShadowBorder, QGrid.RowDefinitions.Count);

}

void DeleteRow(int indexRow)
{
    if (QGrid.RowDefinitions.Count > 4)
     {

        try
        {
            // Удалим элементы
            ControlCombiner cc = listControls.Where(t => Grid.GetRow(t.textBox) == indexRow).First();
            QGrid.Children.Remove(cc.textBox);
            QGrid.Children.Remove(cc.radioButton);
            QGrid.Children.Remove(cc.image);
            QGrid.Children.Remove(cc.checkBox);
            listControls.Remove(cc);

            // переопределим строки
            var query = listControls.Where(t => Grid.GetRow(t.textBox) > indexRow);
            foreach (ControlCombiner container in query)
            {
                if (container.textBox != null)
                    Grid.SetRow(container.textBox, Grid.GetRow(container.textBox) - 1);

                if (container.radioButton != null)
                    Grid.SetRow(container.radioButton, Grid.GetRow(container.radioButton) - 1);

                if (container.image != null)
                    Grid.SetRow(container.image, Grid.GetRow(container.image) - 1);

                if (container.checkBox != null)
                    Grid.SetRow(container.checkBox, Grid.GetRow(container.checkBox) - 1);
            }

        }
        catch { }

       QGrid.RowDefinitions.RemoveAt(indexRow);
      }
}

void ClearAllAnswers()
{
```

```
    foreach (ControlCombiner cc in listControls)
    {
        QGrid.Children.Remove(cc.textBox);
        QGrid.Children.Remove(cc.radioButton);
        QGrid.Children.Remove(cc.image);
        QGrid.Children.Remove(cc.checkBox);
    }

    listControls.Clear();

    while (QGrid.RowDefinitions.Count > 2)
        QGrid.RowDefinitions.RemoveAt(QGrid.RowDefinitions.Count - 1);

}

VersionAnswer GetVersionAnswer(Questions q)
{
    if (question.Answers.Count == 1)
        return VersionAnswer.TextAnswer;

    int NumberCorrectAnswers = q.Answers.Where(an => an.Correct == true).Count();
    if (NumberCorrectAnswers == 1)
        return VersionAnswer.OneAnswer;
    else
        return VersionAnswer.SeveralAnswers;
}

public void SaveChengesToBase()
{
    question.Answers.Clear();
    if (versionAnswer == VersionAnswer.TextAnswer)
    {
        Answers answer = new Answers();
        answer.Name = listControls.First().textBox.Text;
        answer.Correct = true;
        question.Answers.Add(answer);
    }
    else if(versionAnswer == VersionAnswer.OneAnswer)
    {
        foreach(ControlCombiner cc in listControls)
        {
            if (Grid.GetRow(cc.textBox) != QGrid.RowDefinitions.Count && cc.textBox.Text != "Нажмите, чтобы добавить вариант")
            {
                Answers answer = new Answers();
                answer.Name = cc.textBox.Text;
                answer.Correct = cc.radioButton.IsChecked;
                question.Answers.Add(answer);
            }
        }

    }
    else if (versionAnswer == VersionAnswer.SeveralAnswers)
    {
        foreach (ControlCombiner cc in listControls)
        {
            if (Grid.GetRow(cc.textBox) != QGrid.RowDefinitions.Count && cc.textBox.Text != "Нажмите, чтобы добавить вариант")
            {
                Answers answer = new Answers();
                answer.Name = cc.textBox.Text;
                answer.Correct = cc.checkBox.IsChecked;
                question.Answers.Add(answer);
            }
        }
    }
}

#endregion


#region Обработчики событий

void textBoxQuestion_SelectionChanged(object sender, RoutedEventArgs e)
{
    foreach (ControlCombiner t in listControls)
    {
        t.textBox.BorderThickness = new Thickness(1);
        t.textBox.BorderBrush = null;
    }

}

void ComboBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
```

```
            string SelectedTypeOfResponse = ((ComboBox)sender).SelectedItem.ToString();
            if (SelectedTypeOfResponse == "Текст")
                versionAnswer = VersionAnswer.TextAnswer;
            else if (SelectedTypeOfResponse == "Один из списка")
                versionAnswer = VersionAnswer.OneAnswer;
            else if (SelectedTypeOfResponse == "Несколько из списка")
                versionAnswer = VersionAnswer.SeveralAnswers;

            ClearAllAnswers();
            AddGridRowAnswer();
        }

        void TextBox_SelectionChanged(object sender, RoutedEventArgs e)
        {
            TextBox senderTextBox = ((TextBox)sender);
            if (senderTextBox.Text == "Нажмите, чтобы добавить вариант" && Grid.GetRow(senderTextBox) == QGrid.RowDefinitions.Count - 1)
            {
                int NumAnswer = Grid.GetRow(senderTextBox) - 1;
                senderTextBox.Text = "Вариант " + NumAnswer.ToString();
                senderTextBox.Opacity = 1;

                if (versionAnswer == VersionAnswer.OneAnswer)
                {
                    listControls.Last().radioButton.Tag = "";
                    listControls.Last().radioButton.IsChecked = false;
                }
                else if (versionAnswer == VersionAnswer.SeveralAnswers)
                {
                    listControls.Last().checkBox.Tag = "";
                    listControls.Last().checkBox.IsChecked = false;
                }

                AddGridRowAnswer();
            }
            foreach (ControlCombiner t in listControls)
            {
                t.textBox.BorderThickness = new Thickness(1);
                t.textBox.BorderBrush = null;
            }

            senderTextBox.BorderBrush = Brushes.Gray;
            senderTextBox.BorderThickness = new Thickness(2);
        }

        void RadioBut_Checked(object sender, RoutedEventArgs e)
        {
            RadioButton senderRadioButton = ((RadioButton)sender);
            if (Grid.GetRow(senderRadioButton) == QGrid.RowDefinitions.Count - 1)
            {
                senderRadioButton.Tag = "";
                senderRadioButton.IsChecked = false;
                listControls.Last().textBox.Text = "Вариант " + listControls.Count.ToString();
                listControls.Last().textBox.Opacity = 1;

                AddGridRowAnswer();
            }
        }

        void CheckBox_Checked(object sender, RoutedEventArgs e)
        {
            CheckBox senderCheckBox = ((CheckBox)sender);
            if (Grid.GetRow(senderCheckBox) == QGrid.RowDefinitions.Count - 1)
            {
                senderCheckBox.Tag = "";
                senderCheckBox.IsChecked = false;
                listControls.Last().textBox.Text = "Вариант " + listControls.Count.ToString();
                listControls.Last().textBox.Opacity = 1;

                AddGridRowAnswer();
            }
        }

        void imgDel_MouseDown(object sender, MouseButtonEventArgs e)
        {
            int index = Grid.GetRow(((Image)sender));
            int CurentRow = Grid.GetRow((Image)sender);

            DeleteRow(CurentRow);
        }

        #endregion

    }
```

```
}
```

QuestionView.xaml.cs

```
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 TestingSystem.Data;

namespace TestingSystem
{

    public class QuestionViewContenerControls
    {
        public string textAnswer;
        public RadioButton radioButton;
        public CheckBox checkBox;
        public TextBox textBox;
        public bool correctAnswer;
        public VersionAnswer versionAnswer;

        public QuestionViewContenerControls(VersionAnswer vAnswer, string tAnswer, RadioButton rButton, CheckBox cBox, TextBox tbAnswer,
bool cAnswer)
        {
            versionAnswer      = vAnswer;
            textAnswer         = tAnswer;
            radioButton        = rButton;
            checkBox           = cBox;
            correctAnswer      = cAnswer;
            textBox            = tbAnswer;
        }

    }

    public partial class QuestionView : UserControl
    {
        public Questions question = null;
        VersionAnswer versionAnswer;
        List<Button> listButtons = new List<Button>();
        StackPanel ButtonsChengeCopyDel = new StackPanel();
        List<QuestionViewContenerControls> listControls = new List<QuestionViewContenerControls>();
        bool ShowCorrectAnswer;

        public QuestionView(Questions q, bool showCorrectAnswer = false)
        {
            question = q;
            ShowCorrectAnswer = showCorrectAnswer;
            versionAnswer = GetVersionAnswer(question);
            InitializeComponent();
            CreatingElements();

        }


        #region Общие функции

        private void CreatingElements()
        {

            Button butChenge = new Button();
            Image imgChenge = new Image();
            imgChenge.Source = new BitmapImage(new Uri("../images/Изменить.png", UriKind.Relative));
            butChenge.Content = imgChenge;
            butChenge.Width = 20;
            butChenge.Height = 20;
            butChenge.Margin = new Thickness() { Top = 2, Bottom = 2, Left = 2, Right = 2 };
            butChenge.Click += butChenge_Click;
            listButtons.Add(butChenge);

            Button butCopy = new Button();
            Image imgCopy = new Image();
            imgCopy.Source = new BitmapImage(new Uri("../images/СкопироватьОбъект.png", UriKind.Relative));
            butCopy.Content = imgCopy;
```

```
butCopy.Width = 20;
butCopy.Height = 20;
butCopy.Margin = new Thickness() { Top = 2, Bottom = 2, Left = 2, Right = 2 };
butCopy.Click += butCopy_Click;
listButtons.Add(butCopy);

Button butDel = new Button();
Image imgDel = new Image();
imgDel.Source = new BitmapImage(new Uri("../images/Удалить.png", UriKind.Relative));
butDel.Content = imgDel;
butDel.Width = 20;
butDel.Height = 20;
butDel.Margin = new Thickness() { Top = 2, Bottom = 2, Left = 2, Right = 2 };
butDel.Click += butDel_Click;
listButtons.Add(butDel);

ButtonsChengeCopyDel.Orientation = Orientation.Horizontal;
ButtonsChengeCopyDel.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
ButtonsChengeCopyDel.VerticalAlignment = System.Windows.VerticalAlignment.Top;
//ButtonsChengeCopyDel.Opacity = 0.9;
foreach (Button d in listButtons)
    ButtonsChengeCopyDel.Children.Add(d);

Grid.SetRow(ButtonsChengeCopyDel, 0);
Grid.SetColumn(ButtonsChengeCopyDel, 2);


// Define the Columns
ColumnDefinition colDef1 = new ColumnDefinition();
colDef1.Width = GridLength.Auto;
ColumnDefinition colDef2 = new ColumnDefinition();
colDef2.Width = GridLength.Auto;
ColumnDefinition colDef3 = new ColumnDefinition();

QView.ColumnDefinitions.Add(colDef1);
QView.ColumnDefinitions.Add(colDef2);
QView.ColumnDefinitions.Add(colDef3);



// Первая строка
RowDefinition rowDef1 = new RowDefinition();
rowDef1.Height = GridLength.Auto;
QView.RowDefinitions.Add(rowDef1);

Label label1 = new Label();
TextBlock TB = new TextBlock();
TB.TextWrapping = TextWrapping.Wrap;
TB.Text = question.Name;
TB.FontWeight = FontWeights.Bold;
label1.Content = TB;

Grid.SetRow(label1, QView.RowDefinitions.Count - 1);
Grid.SetColumn(label1, 1);
Grid.SetColumnSpan(label1, 2);
QView.Children.Add(label1);

foreach (Answers answer in question.Answers)
{


    RowDefinition rowDef2 = new RowDefinition();
    rowDef2.Height = GridLength.Auto;
    QView.RowDefinitions.Add(rowDef2);

    if (versionAnswer == VersionAnswer.OneAnswer)
    {
        RadioButton NewRadioBut = new RadioButton();
        Thickness marginRadioButton = new Thickness();
        marginRadioButton.Right = 4;
        NewRadioBut.Margin = marginRadioButton;
        NewRadioBut.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
        NewRadioBut.VerticalAlignment = System.Windows.VerticalAlignment.Center;
        NewRadioBut.IsChecked = ShowCorrectAnswer ? answer.Correct : false;
        NewRadioBut.Checked += RadioBut_Checked;
        Grid.SetRow(NewRadioBut, QView.RowDefinitions.Count - 1);
        Grid.SetColumn(NewRadioBut, 1);
        QView.Children.Add(NewRadioBut);

        Label label2 = new Label();
        TextBlock TB2 = new TextBlock();
        TB2.TextWrapping = TextWrapping.Wrap;
```

```
                TB2.Text = answer.Name;
                label2.Content = TB2;

                Grid.SetRow(label2, QView.RowDefinitions.Count - 1);
                Grid.SetColumn(label2, 2);
                QView.Children.Add(label2);

                QuestionViewContenerControls CControl = new QuestionViewContenerControls(versionAnswer, answer.Name, NewRadioBut, null,
null, Convert.ToBoolean(answer.Correct));
                listControls.Add(CControl);



            }
            else if (versionAnswer == VersionAnswer.SeveralAnswers)
            {
                CheckBox NewCheckBox = new CheckBox();
                Thickness marginRadioButton = new Thickness();
                marginRadioButton.Right = 4;
                NewCheckBox.Margin = marginRadioButton;
                NewCheckBox.HorizontalAlignment = System.Windows.HorizontalAlignment.Right;
                NewCheckBox.VerticalAlignment = System.Windows.VerticalAlignment.Center;
                NewCheckBox.IsChecked = ShowCorrectAnswer ? answer.Correct : false;
                NewCheckBox.Checked += CheckBox_Checked;
                Grid.SetRow(NewCheckBox, QView.RowDefinitions.Count - 1);
                Grid.SetColumn(NewCheckBox, 1);
                QView.Children.Add(NewCheckBox);

                Label label2 = new Label();
                TextBlock TB2 = new TextBlock();
                TB2.TextWrapping = TextWrapping.Wrap;
                TB2.Text = answer.Name;
                label2.Content = TB2;

                Grid.SetRow(label2, QView.RowDefinitions.Count - 1);
                Grid.SetColumn(label2, 2);
                QView.Children.Add(label2);

                QuestionViewContenerControls CControl = new QuestionViewContenerControls(versionAnswer, answer.Name, null, NewCheckBox,
null, Convert.ToBoolean(answer.Correct));
                listControls.Add(CControl);
            }

            else if (versionAnswer == VersionAnswer.TextAnswer)
            {

                TextBox TB2 = new TextBox();
                TB2.TextWrapping = TextWrapping.Wrap;
                TB2.Text = ShowCorrectAnswer ? answer.Name : "";

                Grid.SetRow(TB2, QView.RowDefinitions.Count - 1);
                Grid.SetColumnSpan(TB2, 2);
                Grid.SetColumn(TB2, 1);
                QView.Children.Add(TB2);

                QuestionViewContenerControls CControl = new QuestionViewContenerControls(versionAnswer, answer.Name, null, null, TB2, true);
                listControls.Add(CControl);
            }
        }
    }

    private VersionAnswer GetVersionAnswer(Questions q)
    {
        if (question.Answers.Count == 1)
            return VersionAnswer.TextAnswer;

        int NumberCorrectAnswers = q.Answers.Where(an => an.Correct == true).Count();
        if (NumberCorrectAnswers == 1)
            return VersionAnswer.OneAnswer;
        else
            return VersionAnswer.SeveralAnswers;
    }

    public bool CheckedCorrectAnswer()
    {
        bool rezalt = true;

        foreach (QuestionViewContenerControls cc in listControls)
        {
            if (versionAnswer == VersionAnswer.TextAnswer)
            {
                rezalt = (cc.textAnswer == cc.textBox.Text);
            }
```

```
            else if (versionAnswer == VersionAnswer.OneAnswer)
            {
                rezalt = (cc.radioButton.IsChecked == cc.correctAnswer);
            }
            else if (versionAnswer == VersionAnswer.SeveralAnswers)
            {
                rezalt = (cc.checkBox.IsChecked == cc.correctAnswer);
            }

            if (rezalt == false)
                return rezalt;
        }

        return rezalt;
    }

    public bool IsAnswerChecked()
    {
        bool rezalt = false;

        foreach (QuestionViewContenerControls cc in listControls)
        {
            if (versionAnswer == VersionAnswer.TextAnswer)
            {
                rezalt = cc.textBox.Text.Length > 0;
            }
            else if (versionAnswer == VersionAnswer.OneAnswer)
            {
                rezalt = Convert.ToBoolean(cc.radioButton.IsChecked);
            }
            else if (versionAnswer == VersionAnswer.SeveralAnswers)
            {
                rezalt = Convert.ToBoolean(cc.checkBox.IsChecked);
            }

            if (rezalt)
                return rezalt;
        }

        return rezalt;
    }

    public void Update()
    {
        QView.Children.Clear();
        QView.RowDefinitions.Clear();
        QView.ColumnDefinitions.Clear();
        listButtons.Clear();
        listControls.Clear();
        ButtonsChengeCopyDel.Children.Clear();
        versionAnswer = GetVersionAnswer(question);
        CreatingElements();
    }

    public void ShowRezal()
    {

        Image img = new Image();
        if (CheckedCorrectAnswer())
            img.Source = new BitmapImage(new Uri("../images/SymbolCheck.png", UriKind.Relative));
        else
            img.Source = new BitmapImage(new Uri("../images/SymbolError.png", UriKind.Relative));

        img.Width = 32;
        img.Height = 32;
        img.VerticalAlignment = System.Windows.VerticalAlignment.Top;
        Grid.SetRow(img, 0);
        Grid.SetColumn(img, 0);
        QView.Children.Add(img);

        foreach (QuestionViewContenerControls cc in listControls)
        {
            Image imgC = new Image();
            int NumberRow = 0;

            if (versionAnswer == VersionAnswer.TextAnswer)
            {
                cc.textBox.IsEnabled = false;
                NumberRow = Grid.GetRow(cc.textBox);
                if(cc.textAnswer == cc.textBox.Text)
                    imgC.Source = new BitmapImage(new Uri("../images/Check16.png", UriKind.Relative));
                //else
                //    imgC.Source = new BitmapImage(new Uri("../images/Error16.png", UriKind.Relative));
```

```
            RowDefinition rowDef = new RowDefinition();
            rowDef.Height = GridLength.Auto;
            QView.RowDefinitions.Add(rowDef);

            Label label2 = new Label();
            TextBlock TB2 = new TextBlock();
            TB2.TextWrapping = TextWrapping.Wrap;
            TB2.Text = "Ответ: " + cc.textAnswer;
            label2.Content = TB2;

            Grid.SetRow(label2, QView.RowDefinitions.Count - 1);
            Grid.SetColumn(label2, 2);
            QView.Children.Add(label2);
        }
        else if (versionAnswer == VersionAnswer.OneAnswer)
        {
            cc.radioButton.IsEnabled = false;
            NumberRow = Grid.GetRow(cc.radioButton);
            if(cc.correctAnswer)
                imgC.Source = new BitmapImage(new Uri("../images/Check16.png", UriKind.Relative));
            //else
                //imgC.Source = new BitmapImage(new Uri("../images/Error16.png", UriKind.Relative));
        }
        else if (versionAnswer == VersionAnswer.SeveralAnswers)
        {
            cc.checkBox.IsEnabled = false;
            NumberRow = Grid.GetRow(cc.checkBox);
            if(cc.correctAnswer)
                imgC.Source = new BitmapImage(new Uri("../images/Check16.png", UriKind.Relative));
            //else
                //imgC.Source = new BitmapImage(new Uri("../images/Error16.png", UriKind.Relative));
        }

        imgC.Width = 16;
        imgC.Height = 16;
        Grid.SetRow(imgC, NumberRow);
        Grid.SetColumn(imgC, 0);
        QView.Children.Add(imgC);
    }
}

#endregion

#region События

public delegate void MouseButtonClick(object sender, MouseButtonEventArgs e);
public event MouseButtonClick ButtonChengeClick;
public event MouseButtonClick ButtonCopyClick;
public event MouseButtonClick ButtonDeleteClick;

private void butDel_Click(object sender, RoutedEventArgs e)
{
    ButtonDeleteClick(this, null);
}

private void butCopy_Click(object sender, RoutedEventArgs e)
{
    ButtonCopyClick(this, null);
}

private void butChenge_Click(object sender, RoutedEventArgs e)
{
    ButtonChengeClick(this, null);
}

private void RadioBut_Checked(object sender, RoutedEventArgs e)
{
    if (ShowCorrectAnswer)
        ((RadioButton)sender).IsChecked = false;
}

private void CheckBox_Checked(object sender, RoutedEventArgs e)
{
    if (ShowCorrectAnswer)
        ((CheckBox)sender).IsChecked = false;
}

private void QView_MouseEnter_1(object sender, MouseEventArgs e)
{
    Background = new SolidColorBrush(Color.FromArgb(255, (byte)245, (byte)245, (byte)245));
    if (ShowCorrectAnswer)
        QView.Children.Add(ButtonsChengeCopyDel);
```

```
        }

        private void QView_MouseLeave_1(object sender, MouseEventArgs e)
        {
            Background = Brushes.White;
            if (ShowCorrectAnswer)
                QView.Children.Remove(ButtonsChengeCopyDel);
        }

        #endregion

    }
}
```

AddNewTestWindow.xaml.cs

```
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.Shapes;
using TestingSystem.Data;
using System.Text.RegularExpressions;

namespace TestingSystem
{

    public partial class AddNewTestWindow : Window
    {
        TSDataManager dataManager;
        public Tests newTest;

        public AddNewTestWindow(TSDataManager dManager)
        {
            dataManager = dManager;
            InitializeComponent();
            FillTopicsComboBox();
        }

        void FillTopicsComboBox()
        {
            foreach (Topics t in dataManager.GetTopics())
            {
                TopicsComboBox.Items.Add(t.Name);
            }
        }

        private void Button_OK(object sender, RoutedEventArgs e)
        {
            QuestionName.Text = CleanString(QuestionName.Text);
            newTest = dataManager.AddNewTest(QuestionName.Text, TopicsComboBox.SelectedItem.ToString());
            Close();
        }

        string CleanString(string strIn)
        {
            try
            {
                return Regex.Replace(strIn, @"[^\w\.@-]", "",
                                     RegexOptions.None, TimeSpan.FromSeconds(1.5));
            }
            catch (RegexMatchTimeoutException)
            {
                return String.Empty;
            }
        }

        private void Button_Cancel(object sender, RoutedEventArgs e)
        {
            Close();
        }
    }
}
```

AddNewTopicWindow.xaml.cs

```
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.Shapes;
using TestingSystem.Data;

namespace TestingSystem
{

   public partial class AddNewTopicWindow : Window
   {
      TSDataManager dataManager;
      public Topics newTopic;

      public AddNewTopicWindow(TSDataManager tm)
      {
         dataManager = tm;
         InitializeComponent();
      }

      private void Button_Ok(object sender, RoutedEventArgs e)
      {
         newTopic = dataManager.AddNewTopic(NameNewTopic.Text);
         Close();
      }

      private void Button_Cancel(object sender, RoutedEventArgs e)
      {
         Close();
      }
   }
}
```

Authorization.xaml.cs

```
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.Shapes;
using TestingSystem.Data;

namespace TestingSystem
{

   public partial class Authorization : Window
   {
      TestEntities dataContext = new TestEntities();

      public Authorization()
      {
         InitializeComponent();
      }

      private void Ok_Click_1(object sender, RoutedEventArgs e)
      {
         if (IsLoginCorrect())
         {
            MainWindow mainWindow = new MainWindow(Login.Text);
            mainWindow.Show();
```

```
            }

            else
            {
                System.Windows.MessageBox.Show("Ошибка авторизации. Не верные данные");
            }
            Close();
        }

        private void Cansel_Click_1(object sender, RoutedEventArgs e)
        {
            Close();
        }

        private void Registration_Click_1(object sender, RoutedEventArgs e)
        {
            Registration RegWindow = new Registration();
            RegWindow.Show();
        }

        private bool IsLoginCorrect()
        {
            IQueryable<Users> query = dataContext.Users.Where(l => l.Name == Login.Text);
            foreach (Users user in query)
            {
                return (user.PasswordName == Password.Password);
            }
            return false;
        }


    }
}
```

Registration.xaml.cs

```
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.Shapes;
using TestingSystem.Data;

namespace TestingSystem
{

    public partial class Registration : Window
    {
        TestEntities dataContext = new TestEntities();

        public Registration()
        {
            InitializeComponent();
        }

        private void Ok_Click_1(object sender, RoutedEventArgs e)
        {
            string ErrorText = "";
            if (Login.Text == "")
                ErrorText += "\nЗаполните логин";
            if (Password.Password == "")
                ErrorText += "\nЗаполните пароль";
            if (PasswordAgain.Password == "")
                ErrorText += "\nЗаполните пароль повтор";

            if (ErrorText != "")
            {
                System.Windows.MessageBox.Show(ErrorText);
            }
            else
            {
                if (PasswordGood())
                {
                    if (IsLoginExists(Login.Text))
                        System.Windows.MessageBox.Show("Логин не уникален");
```

```
                else
                {
                    Users user = new Users();
                    user.Name = Login.Text;
                    user.PasswordName = Password.Password;
                    dataContext.Users.Add(user);
                    dataContext.SaveChanges();
                    Close();
                }
            }
        }
    }

    private void Cansel_Click_1(object sender, RoutedEventArgs e)
    {
        Close();
    }

    private bool IsLoginExists(string log)
    {
        IQueryable<Users> query = dataContext.Users.Where(u => u.Name == log);
        foreach (Users user in query)
        {
            return true;
        }
        return false;
    }

    private bool PasswordGood()
    {
        if (Password.Password.Length < 8)
        {
            MessageBox.Show("Пароль должен быть не менее 8 символов");
            return false;
        }

        if (Password.Password != PasswordAgain.Password)
        {
            System.Windows.MessageBox.Show("Пароли не совпадают");
            return false;
        }

        return true;
    }
  }
}
```

MainWindow.xaml.cs

```
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 TestingSystem.Data;


namespace TestingSystem
{

  public partial class MainWindow : Window
  {
    public class PairQuestionControls
    {
        public QuestionView questionView;
        public QuestionConstructor questionConstructor;

        public PairQuestionControls(QuestionView qView, QuestionConstructor qConstructor)
        {
            questionView = qView;
            questionConstructor = qConstructor;
        }
    }
```

```
TSDataManager dataManager = new TSDataManager();
List<PairQuestionControls> listQuestionsControls = new List<PairQuestionControls>();
bool AdminRight = true;
string UserName;
Tests CurrentTest;
Topics CurrentTopic;

public MainWindow()
{
   InitializeComponent();
   FillQuestionsTree();
   UserName = "";
   InstallingVisibilityControls();
}

public MainWindow(string userName)
{
   InitializeComponent();
   FillQuestionsTree();
   UserName = userName;
   AdminRight = (userName == "Admin");
   Title = Title + ((userName != "") ? " - "+userName : "");
   InstallingVisibilityControls();
}

private void InstallingVisibilityControls()
{
   MenuEdit.IsEnabled             = AdminRight;
   MenuItemDeleteTest.IsEnabled    = CurrentTest == null ? false : true;
   MenuItemDeleteTopic.IsEnabled   = CurrentTopic == null ? false : true;
   MenuItemAddTest.IsEnabled       = CurrentTopic == null ? false : true;
   MenuItemAddQuestion.IsEnabled   = CurrentTest == null ? false : true;
   MenuItemSave.IsEnabled          = CurrentTest == null ? false : true;
}

private void FillQuestionsTree()
{
   QuestionsTree.Items.Clear();
   List<Topics> listTopics = dataManager.GetTopics();

   foreach (Topics topic in listTopics)
   {
      TreeViewItem newItem = new TreeViewItem();
      StackPanel stackTopic = new StackPanel();
      stackTopic.Orientation = Orientation.Horizontal;
      Label labelTopic = new Label();
      labelTopic.Name = "LabelHeader";
      labelTopic.Content = topic.Name;
      Image imageTopic = new Image();
      imageTopic.Source = new BitmapImage(new Uri("../images/folder.png", UriKind.Relative));
      stackTopic.Children.Add(imageTopic);
      stackTopic.Children.Add(labelTopic);
      newItem.Header = stackTopic;

      foreach (Tests test in dataManager.GetTests(topic))
      {
         TreeViewItem newItemTest = new TreeViewItem();
         StackPanel stackTest = new StackPanel();
         stackTest.Orientation = Orientation.Horizontal;
         Label labelTest = new Label();
         labelTest.Name = "LabelHeader";
         labelTest.Content = test.Name;
         Image imageTest = new Image();
         imageTest.Source = new BitmapImage(new Uri("../images/Документ.png", UriKind.Relative));
         stackTest.Children.Add(imageTest);
         stackTest.Children.Add(labelTest);

         newItemTest.Header = stackTest;
         newItem.Items.Add(newItemTest);
      }
      QuestionsTree.Items.Add(newItem);
   }
}

private void FillSelectedTest(Tests cTest)
{
   CurentListQuastionsView.Children.Clear();
   listQuestionsControls.Clear();

   if (cTest == null)
      return;

   foreach (Questions q in cTest.Questions)
```

```
            AddQuestionControlToCurentListQuastionsView(q);

        if (!AdminRight)
        {
            Button butEndTest = new Button();
            butEndTest.Name = "ButtonEndTest";
            butEndTest.Content = "Завершить тест";
            butEndTest.FontWeight = FontWeights.Bold;
            butEndTest.Background = Brushes.Azure;
            butEndTest.Margin = new Thickness() { Top = 2, Bottom = 2, Left = 2, Right = 2 };
            butEndTest.Click += butEndTest_Click;
            butEndTest.MaxWidth = 220;
            butEndTest.MinWidth = 220;
            butEndTest.MinHeight = 40;
            butEndTest.HorizontalAlignment = System.Windows.HorizontalAlignment.Center;
            CurentListQuastionsView.Children.Add(butEndTest);
        }
    }

    private void AddQuestionControlToCurentListQuastionsView(Questions q)
    {
        QuestionView QView = new QuestionView(q, AdminRight);
        if (AdminRight)
        {
            QView.MouseLeftButtonDown   += QView_MouseLeftButtonDown;
            QView.ButtonChengeClick     += QView_MouseLeftButtonDown;
            QView.ButtonCopyClick       += QView_Copy;
            QView.ButtonDeleteClick     += QView_Delete;
        }
        QuestionConstructor QConstructor = new QuestionConstructor(q);

        PairQuestionControls pair = new PairQuestionControls(QView, QConstructor);
        listQuestionsControls.Add(pair);

        StackPanel panel = new StackPanel();
        panel.Children.Add(QView);

        CurentListQuastionsView.Children.Add(panel);
    }

    private void FocusItemInQuestionsTree(string nameItem, ItemCollection Items)
    {
        if (Items == null || nameItem == "")
            return;

        foreach (TreeViewItem item in Items)
        {
            string CurrentItemName = ((item.Header as StackPanel).Children[1] as Label).Content.ToString();
            if (CurrentItemName == nameItem)
            {
                item.IsExpanded = true;
                item.IsSelected = true;
                TreeViewItem parent = item.Parent as TreeViewItem;
                if (parent != null)
                    parent.IsExpanded = true;

                return;
            }
            else if (item.Items.Count > 0)
                FocusItemInQuestionsTree(nameItem, item.Items);
        }
    }

    private void CheckOutTest()
    {
        int numberCorrectAnswers = 0;
        int numberFalseAnswers = 0;

        foreach (PairQuestionControls p in listQuestionsControls)
        {
            if (!p.questionView.IsAnswerChecked())
                p.questionView.Background = Brushes.Red;

            if (p.questionView.CheckedCorrectAnswer())
                numberCorrectAnswers++;
            else
                numberFalseAnswers++;

            p.questionView.ShowRezal();
        }
        string rezalt = String.Format("Правильных ответов: {0} , неправильных ответов: {1}", numberCorrectAnswers, numberFalseAnswers);
        MessageBox.Show(rezalt);
```

```
}

#region Обработчики событий

private void butEndTest_Click(object sender, RoutedEventArgs e)
{
    Button but = sender as Button;
    if (but.Content == "Повторить тест")
    {
        FillSelectedTest(CurrentTest);
        but.Content = "Завершить тест";
    }
    else if (but.Content == "Завершить тест")
    {
        foreach (PairQuestionControls p in listQuestionsControls)
        {
            if (!p.questionView.IsAnswerChecked())
            {
                MessageBox.Show("Проверьте тест. Есть пропущенные вопросы!");
                return;
            }
        }

        CheckOutTest();
        but.Content = "Повторить тест";
    }

}

private void QView_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)
{
    foreach (StackPanel panel in CurentListQuastionsView.Children)
    {
        if (panel.Children[0] is QuestionConstructor)
        {
            QuestionConstructor QC = panel.Children[0] as QuestionConstructor;
            QC.SaveChengesToBase();

            QuestionView QV = listQuestionsControls.Where(p => p.questionConstructor == panel.Children[0]).First().questionView;
            QV.Update();
            panel.Children.Clear();
            panel.Children.Add(QV);
        }
    }

    StackPanel CurrentPanel = null;
    foreach (StackPanel panel in CurentListQuastionsView.Children)
    {
        if (panel.Children.Contains((QuestionView)sender))
        {
            CurrentPanel = panel;
            break;
        }
    }

    CurrentPanel.Children.Clear();
    CurrentPanel.Children.Add(listQuestionsControls.Where(el => el.questionView == (QuestionView)sender).First().questionConstructor);
}

private void QView_Copy(object sender, MouseButtonEventArgs e)
{
    QuestionView QV = (sender as QuestionView);
    Questions CopyQuestion = dataManager.CopyQuestion(QV.question);
    AddQuestionControlToCurentListQuastionsView(CopyQuestion);
}

private void QView_Delete(object sender, MouseButtonEventArgs e)
{
    QuestionView QV = (sender as QuestionView);
    dataManager.DeleteQuestion(QV.question);

    PairQuestionControls PairFind = listQuestionsControls.Where(p => p.questionView == QV).First();
    listQuestionsControls.Remove(PairFind);

    StackPanel PanelFind = null;
    foreach (StackPanel panel in CurentListQuastionsView.Children)
    {
        if (panel.Children[0] is QuestionView && (QuestionView)panel.Children[0] == QV)
        {
            PanelFind = panel;
            break;
        }
```

```
            }
            CurentListQuastionsView.Children.Remove(PanelFind);
        }

        private void QuestionsTree_Selected_1(object sender, RoutedEventArgs e)
        {
            TreeViewItem test = e.OriginalSource as TreeViewItem;
            TreeViewItem topic = test.Parent as TreeViewItem;

            //Сохраним последние изменения в тесте
            if (AdminRight)
            {
                foreach (StackPanel panel in CurentListQuastionsView.Children)
                {
                    if (panel.Children[0] is QuestionConstructor)
                    {
                        QuestionConstructor QC = panel.Children[0] as QuestionConstructor;
                        QC.SaveChengesToBase();
                    }
                }
            }

            CurentListQuastionsView.Children.Clear();

            if (topic == null && test != null)
            {
                string tName = ((test.Header as StackPanel).Children[1] as Label).Content.ToString();
                CurrentTopic = dataManager.GetTopicByName(tName);
                CurrentTest = null;
                InstallingVisibilityControls();
            }

            StatusBarLeftSide.Text = CurrentTopic == null ? "" :CurrentTopic.Name;
            StatusBarRightSide.Text = "";
            if (topic == null || test == null)
                return;

            string topicName = ((topic.Header as StackPanel).Children[1] as Label).Content.ToString();
            string testName = ((test.Header as StackPanel).Children[1] as Label).Content.ToString();


            StatusBarLeftSide.Text = topicName;
            StatusBarRightSide.Text = (topicName == testName) ? "..." : testName;

            if (test != topic && testName.Length > 0 && topicName.Length > 0)
            {
                CurrentTest = dataManager.GetTest(topicName, testName);
                CurrentTopic = dataManager.GetTopicByName(topicName);
                FillSelectedTest(CurrentTest);
            }

            InstallingVisibilityControls();
        }

        #endregion


        #region Кнопки меню

        private void MenuItem_About(object sender, RoutedEventArgs e)
        {
            About aboutWindow = new About();
            aboutWindow.Show();
        }

        private void MenuItem_Login(object sender, RoutedEventArgs e)
        {
            MessageBoxResult result = MessageBox.Show("Все не сохраненные данные будут утеряны. Продолжить?", "Внимание",
MessageBoxButton.YesNo, MessageBoxImage.Question);
            if (result == MessageBoxResult.Yes)
            {
                Authorization authorizationWindow = new Authorization();
                authorizationWindow.Show();
                Close();
            }
        }

        private void MenuItem_Regisration(object sender, RoutedEventArgs e)
        {
            Registration regWindow = new Registration();
            regWindow.Show();
        }
```

```
        private void MenuItem_AddTopic(object sender, RoutedEventArgs e)
        {
            AddNewTopicWindow newTopicWindow = new AddNewTopicWindow(dataManager);
            newTopicWindow.ShowDialog();
            FillQuestionsTree();
            if (newTopicWindow.newTopic == null)
                return;

            FocusItemInQuestionsTree(newTopicWindow.newTopic.Name, QuestionsTree.Items);
        }

        private void MenuItem_AddTest(object sender, RoutedEventArgs e)
        {
            AddNewTestWindow newTestWindow = new AddNewTestWindow(dataManager);
            newTestWindow.ShowDialog();
            FillQuestionsTree();
            if (newTestWindow.newTest == null)
                return;

            FocusItemInQuestionsTree(newTestWindow.newTest.Name, QuestionsTree.Items);
        }

        private void MenuItem_AddQuestion(object sender, RoutedEventArgs e)
        {
            string questionName = "Вопрос " + (CurentListQuastionsView.Children.Count + 1).ToString();
            Questions newQuestion = dataManager.AddNewQuestion(questionName, CurrentTest);

            AddQuestionControlToCurentListQuastionsView(newQuestion);
            ScrollCurentListQuastionsView.ScrollToEnd();
            PairQuestionControls pair = listQuestionsControls[listQuestionsControls.Count - 1];
            QView_MouseLeftButtonDown(pair.questionView, null);
            pair.questionView.Focus();
        }

        private void MenuItem_Exit(object sender, RoutedEventArgs e)
        {
            MessageBoxResult result = MessageBox.Show("Все не сохраненные данные будут утеряны. Продолжить?", "Выход из системы",
MessageBoxButton.YesNo, MessageBoxImage.Question);
            if (result == MessageBoxResult.Yes)
            {
                Close();
            }
        }

        private void MenuItem_DeleteTopic(object sender, RoutedEventArgs e)
        {
            if (CurrentTopic != null)
            {
                MessageBoxResult result = MessageBox.Show("Желаете удалить тему " + CurrentTopic.Name + " со всем его содержимым? Вы
уверены?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question);
                if (result == MessageBoxResult.Yes)
                {
                    dataManager.DeleteTopic(CurrentTopic);
                    CurrentTopic = null;
                    FillQuestionsTree();
                }
            }
        }

        private void MenuItem_DeleteTest(object sender, RoutedEventArgs e)
        {
            if (CurrentTest != null)
            {
                MessageBoxResult result = MessageBox.Show("Желаете удалить тест " + CurrentTest.Name + "? Вы уверены?", "Внимание",
MessageBoxButton.YesNo, MessageBoxImage.Question);
                if (result == MessageBoxResult.Yes)
                {
                    dataManager.DeleteTest(CurrentTest);
                    CurrentTest = null;
                    FillQuestionsTree();
                    FillSelectedTest(CurrentTest);
                }
            }
        }

        private void MenuItem_Save(object sender, RoutedEventArgs e)
        {
            if(CurrentTest != null)
                dataManager.SaveTestToXML(CurrentTest);
        }

        private void MenuItem_Open(object sender, RoutedEventArgs e)
        {
```

```
            Tests TestXML = dataManager.GetTestFromXML();
            if (TestXML == null)
                return;

            CurrentTest = TestXML;
            FillSelectedTest(CurrentTest);
        }

        #endregion

    }
}
```

ПРИЛОЖЕНИЕ 2. СКРИПТ БАЗЫ ДАННЫХ

```
create table Topics(
	id 				INT not null PRIMARY KEY IDENTITY(1,1),
	Name 				VARCHAR (255) not null UNIQUE
)
go

create table Tests(
	id 				INT not null PRIMARY KEY IDENTITY(1,1),
	Name 				VARCHAR (255) not null,
	id_Topic 			INT REFERENCES Topics (id),
	UNIQUE (id, id_Topic)
)
go

create table Questions(
	id 				INT not null PRIMARY KEY IDENTITY(1,1),
	Name 				TEXT not null,
	id_Test 		INT REFERENCES Tests (id) not null
)
go

create table Answers(
	id 				INT not null PRIMARY KEY IDENTITY(1,1),
	Name 				TEXT not null,
	id_Question 			INT REFERENCES Questions (id),
	Correct 		bit DEFAULT 0
)
go

create table Users(
	id 				INT not null PRIMARY KEY IDENTITY(1,1),
	Name 				VARCHAR (255) not null,
	LastName 			VARCHAR (255),
	GroupName 			VARCHAR (255),
	PasswordName 			VARCHAR (10)
)
go
```