МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационные системы и технологии

Специальность 1-40 01 01 «Программное обеспечение информационных технологий»

Специализация Программирование интернет-приложений

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

«eLearning» – учим английский вместе!

Выполнил студент Септилко Анастасия Антоновна

(Ф.И.О.)

Руководитель проекта Бондарчик Е.Н.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Заведующий кафедрой к.т.н., доц. Смелов В.В.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Минск 2019

**Реферат**

ФИО

Подпись

Дата

Лист

1

КП 00.01.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

Реферат

Лит.

Листов

1

БГТУ 71172586, 2019

У

Пояснительная записка курсового проекта содержит 42 страниц, 24 рисунок, 9 источников литературы, 3 приложения.

C#, WPF, MATERIAL DESIGN, MICROSOFT SQL SERVER , MD5, ADO.NET

В современном мире интернет, мобильная связь играют огромную роль в жизни человечества. Пару-тройку десятков лет назад нельзя было и подумать, что для того, чтобы пообщаться или увидеться со своими лучшими друзьями или родственниками, проживающими в другой стране или даже городе, не придется к ним ехать. В настоящее время благодаря Интернету можно встречаться, решать сложнейшие задачи посредством доступа во всемирную паутину, где для решение различных задач есть вся необходимая информация.

Люди каждый день проводят время со своим компьютером или смартфоном удовлетворяя свои потребности, но лишь единицы используют его для саморазвития. Свое свободное время можно потратить с пользой, например, изучать иностранные языки с помощью специальных приложений.

Темой курсового проекта является разработка программного средства для обучения английскому языку. Пользователю предоставлена возможность изучать английский язык и проходить соответствующее тестирование, администратору добавлять тесты, удалять и следить за пользователями. Пояснительная записка состоит из введения, пяти разделов, заключения.

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

В первом разделе обзор технических решений и литературных источников.

Во втором разделе рассматривается проектирование проекта.

В третьем разделе представлена реализация программного средства.

В четвертом разделе представлено руководство пользователя.

В пятом разделе представлено тестирование приложения.

В заключении представлены результаты выполнения поставленных задач и целей курсового проекта, которые были решены в ходе разработки программного средства.

Содержание

ФИО

Подпись

Дата

Лист

1

КП 00.02.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

Содержание

Лит.

Листов

1

БГТУ 71172586, 2019

У

[Введение 4](#_Toc27124528)

[1. Обзор технических решений и литературных источников 5](#_Toc27124529)

[1.2 Обзор средств разработки 7](#_Toc27124530)

[2. Проектирование 9](#_Toc27124531)

[2.1 Диаграмма использования приложения 9](#_Toc27124532)

[2.2 Описание технологии 10](#_Toc27124533)

[2.3. Разработка модели базы данных 11](#_Toc27124534)

[2.4. Используемые объекты базы данных 14](#_Toc27124535)

[2.4.1. Таблицы 14](#_Toc27124536)

[2.4.2. Пользователи 14](#_Toc27124537)

[2.4.3. Хранимые процедуры 14](#_Toc27124538)

[2.4.4. Триггеры 15](#_Toc27124539)

[3. Реализация программного средства 16](#_Toc27124540)

[3.1 Постановка задачи 16](#_Toc27124541)

[3.2 Обобщенная структура проекта 16](#_Toc27124542)

[3.3 Описание процедур импорта и экспорта 18](#_Toc27124543)

[3.3.1. Описание процедуры экспорта 18](#_Toc27124544)

[3.3.2. Описание процедуры импорта 18](#_Toc27124545)

[3.3. Регистрация и авторизация 19](#_Toc27124546)

[3.4. Выбор нужного раздела 20](#_Toc27124547)

[4. Руководство пользователя 23](#_Toc27124548)

[5. Тестирование производительности 28](#_Toc27124549)

[Заключение 31](#_Toc27124550)

[Список используемых источников 32](#_Toc27124551)

[Приложение A 33](#_Toc27124552)

[Приложение Б 41](#_Toc27124553)

[Приложение В 42](#_Toc27124554)

# Введение

ФИО

Подпись

Дата

Лист

1

КП 00.03.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

Введение

Лит.

Листов

1

БГТУ 71172586, 2019

У

В данной записке приведено описание реляционной базы данных, разработанной в соответствии с заданием на курсовое проектирование по теме «eLearning» по дисциплине «Базы данных». Целью курсового проекта является разработка программного средства для обучения английскому языку и прохождение тестов для проверки уровня знания английского языка.

База данных — представленная в объективной форме совокупность самостоятельных материалов (статей, расчётов, нормативных актов и иных подобных материалов), систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины. Реляционная база данных — база данных, основанная на реляционной модели данных.

Для разработки и управления базой данных курсового проекта использовалась объектно-реляционная система управления базами данных «MS SQL Server».

Система управления базами данных (СУБД) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Основные функции СУБД:

* определение структуры создаваемой базы данных, ее инициализация и проведение начальной загрузки;
* предоставление пользователям возможности манипулирования данными (выборка необходимых данных, выполнение вычислений, разработка интерфейса ввода/вывода, визуализация);
* обеспечение логической и физической независимости данных;
* защита логической целостности базы данных;
* защита физической целостности;
* управление полномочиями пользователей на доступ к базе данных;
* синхронизация работы нескольких пользователей;
* управление ресурсами среды хранения;
* поддержка деятельности системного персонала.

Примеры СУБД: Oracle, MS SQL Server, Microsoft Access, MySql, PostgreSQL, MongoDB и так далее.

# Обзор технических решений и литературных источников

ФИО

Подпись

Дата

Лист

1

КП 01.04.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

1. Обзор технических решений и литературных источников

Лит.

Листов

4

БГТУ 71172586, 2019

«Слова Бегом» – это быстрый способ заговорить на английском языке. С помощью методик приложения, запоминание иностранных слов и фраз происходит в 5 раз быстрее. Приложение подойдет тем, кто учит язык с нуля, и тем, кому необходимо существенно расширить словарный запас, перейти на новый уровень знаний. Приложение активно используется в связи со своими эффективными результатами (рисунок 1.1).

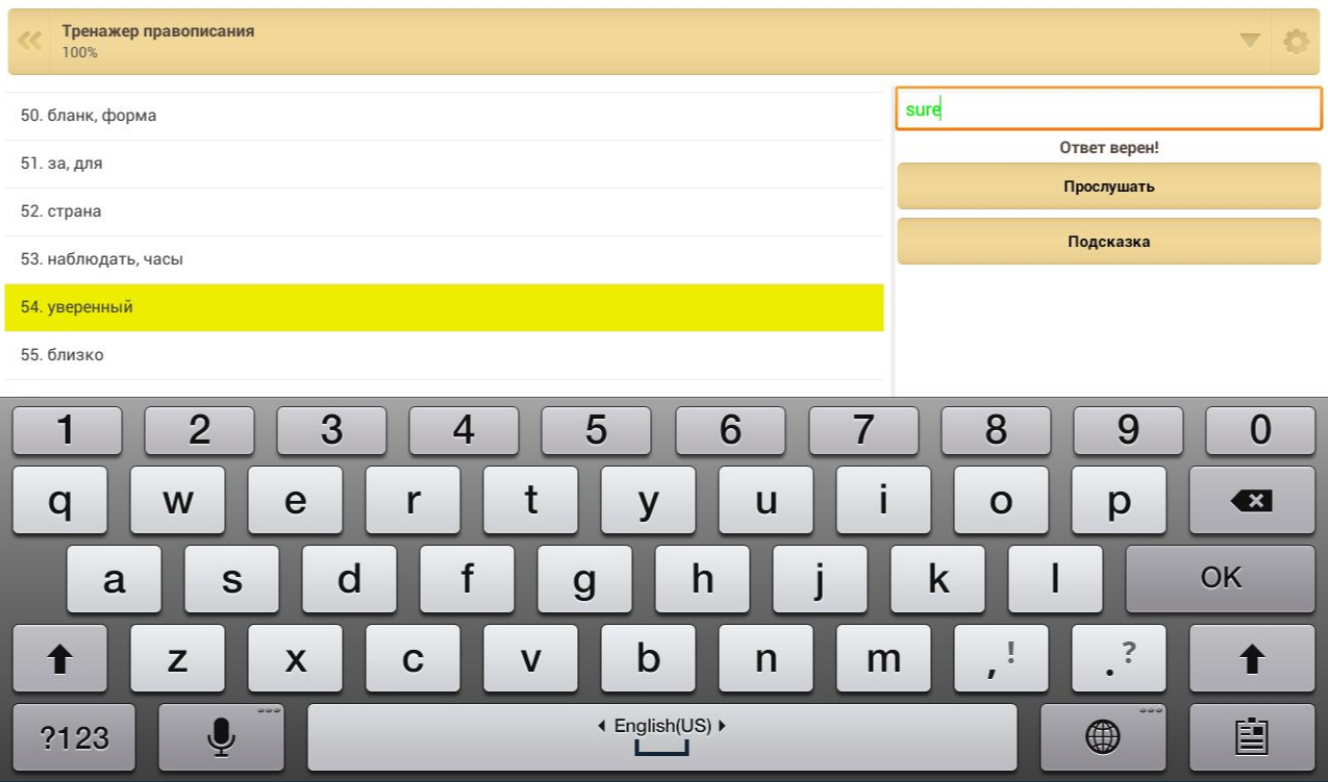


Рисунок 1.1 – «Слова Бегом»

«Слова Бегом» – это:

* профессиональное студийное озвучивание;
* тренажер произношения;
* легкая, эффективная проверка знаний;
* наборы слов и популярных фраз;
* изучение грамматики и фонетики;
* персональный словарь освоенных слов и фраз.

*«*Lingualeo*»* – полезное приложение, которое позволяет улучшь навыки английского языка как новичку, так и профессионалу. Данное приложение позволит выбрать индивидуальный план развития для каждого пользователя. Здесь пользователю предоставлен интерфейс с необходимыми упражнениями для выполнения на текущий день. Приложение доступно для различных платформ, таких как IOS, Android и Windows Phone (рисунок 1.2).

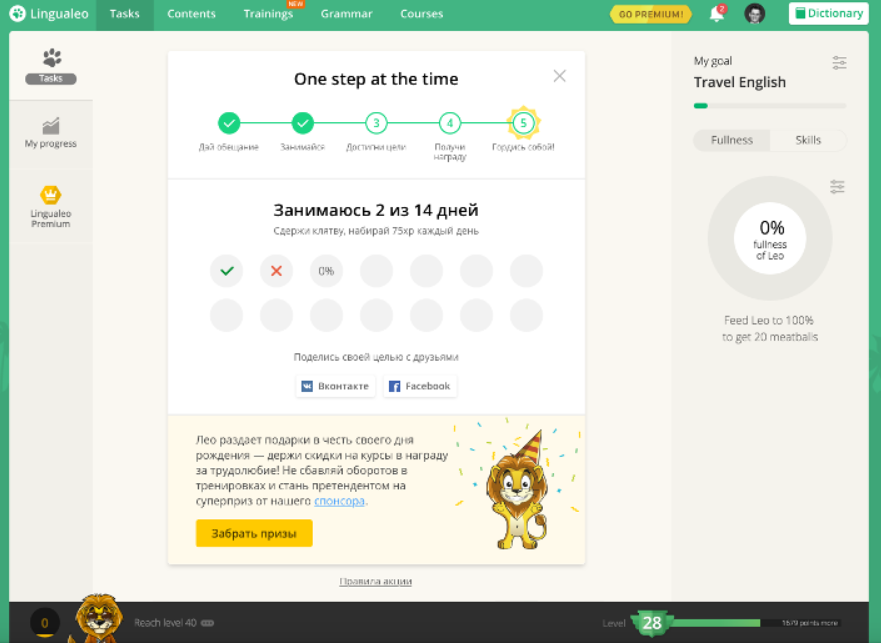


Рисунок 1.2 – «Lingualeo»

Программное средство выполняет следующие задачи:

* побор индивидуальных упражнений;
* добавление выбранных слов в индивидуальный словарь;
* выполняет перевод текста;
* отображает статистику и навыки пользователя.

## [1.2 Обзор средств разработки](#_Toc485109377)

C# — объектно-ориентированный язык программирования. Разработан в 1998 — 2001 под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework.C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML. Переняв многое от своих предшественников — языков C++, Pascal, Модула, Smalltalk и, в особенности, Java — С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов) [3]

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. [XAML](https://ru.wikipedia.org/wiki/XAML) представляет собой язык декларативного описания интерфейса, основанный на [XML](https://ru.wikipedia.org/wiki/XML). Также реализована модель разделения кода и дизайна, позволяющая кооперироваться программисту и дизайнеру. Кроме того, есть встроенная поддержка стилей элементов, а сами элементы легко разделить на [элементы управления](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F) второго уровня, которые, в свою очередь, разделяются до уровня векторных фигур и свойств/действий. Это позволяет легко задать стиль для любого элемента, например, [Button](https://ru.wikipedia.org/wiki/%D0%9A%D0%BD%D0%BE%D0%BF%D0%BA%D0%B0_(%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC)) (кнопка). [4]

Material Design (рус. Материальный дизайн) — дизайн программного обеспечения и приложений операционных систем Android и Chrome OS от компании Google. Впервые представлен на конференции Google I/O 25 июня 2014 года. Идея дизайна заключается в приложениях, которые открываются и сворачиваются как карточки, используя эффекты теней. По идее дизайнеров Google, у приложений не должно быть острых углов, карточки должны переключаться между собой плавно и практически незаметно.[5]

ADO.NET (ActiveX Data Object для .NET) – технология, предоставляющая доступ к данным для приложений, основанных на Microsoft. 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.[6]

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.[1]

XML ( [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) eXtensible Markup Language) —расширяемый [язык разметки](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%82%D0%BA%D0%B8). Рекомендован [Консорциумом Всемирной паутины](https://ru.wikipedia.org/wiki/W3C) (W3C). Спецификация XML описывает XML-документы и частично описывает поведение XML-процессоров (программ, читающих XML-документы и обеспечивающих доступ к их содержимому). XML разрабатывался как язык с простым формальным [синтаксисом](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), удобный для создания и обработки документов программами и одновременно удобный для чтения и создания документов человеком, с подчёркиванием нацеленности на использование в Интернете. Расширение XML — это конкретная [грамматика](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0), созданная на базе XML и представленная словарём тегов и их атрибутов, а также набором правил, определяющих какие атрибуты и элементы могут входить в состав других элементов. Сочетание простого формального синтаксиса, удобства для человека, расширяемости, а также базирование на кодировках [Юникод](https://ru.wikipedia.org/wiki/%D0%AE%D0%BD%D0%B8%D0%BA%D0%BE%D0%B4) для представления содержания документов привело к широкому использованию как собственно XML, так и множества производных специализированных языков на базе XML в самых разнообразных программных средствах.[2]

MD5 (Message Digest Algorithm 5) был разработан в RSA Laboratories и работает быстрее, чем SHA1. Алгоритм SHA1 является более устойчивым к атакам с применением методов грубой силы. Алгоритмы SHA разрабатывались в NSA (National Security Agency — Агентство национальной безопасности). В MD5 размер хеш-значения составляет 128 бит, а в SHA1 — 160 бит. В других алгоритмах SHA размер хеш-значения указан прямо в названии. [7]

# 2. Проектирование

ФИО

Подпись

Дата

Лист

1

КП 02.05.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

2. Проектирование

Лит.

Листов

7

БГТУ 71172586, 2019

У

## 2.1 Диаграмма использования приложения

Визуальное моделирование в UML можно представить как некоторый процесс поуровневого спуска от наиболее обшей и абстрактной концептуальной модели исходной системы к логической, а затем и к физической модели соответствующей программной системы. Для достижения этих целей вначале строится модель в форме так называемой диаграммы вариантов использования (use case diagram), которая описывает функциональное назначение системы или, другими словами, то, что система будет делать в процессе своего функционирования. Диаграмма вариантов использования является исходным концептуальным представлением или концептуальной моделью системы в процессе ее проектирования и разработки.

ДиаграммаUML–это графическое представление набора элементов,изображаемое в виде связанного графа с вершинами (сущностями) и ребрами (отношениями).

Диаграмма использования представлена на рисунке 2.1.

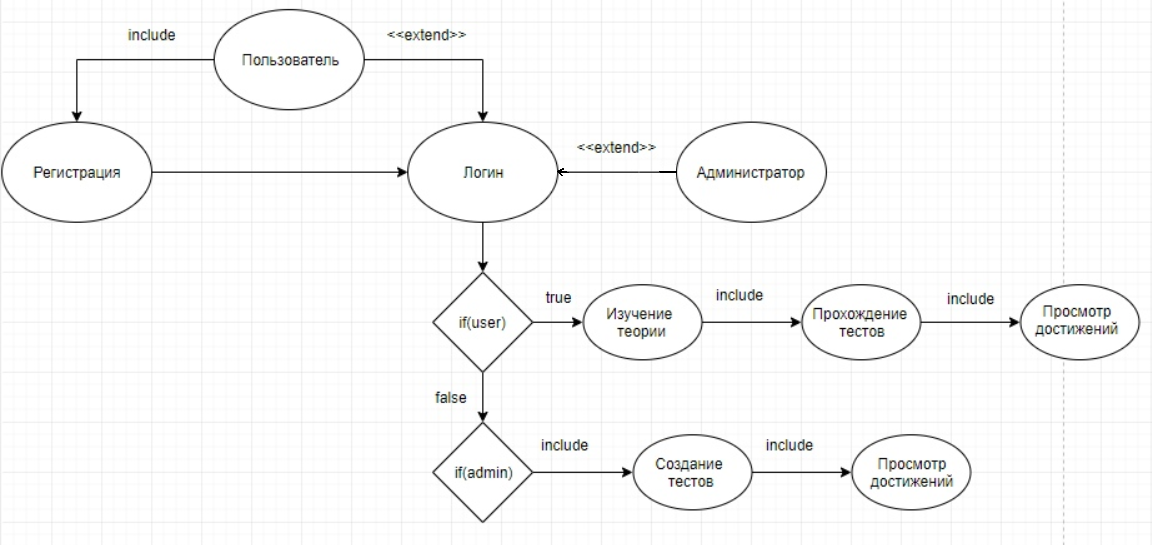


Рисунок 2.1 – Диаграмма использования

## 2.2 Описание технологии

Все мы знакомы с таким понятием как регистрация на сайтах с последующим прохождением авторизации/аутентификации.

Конфиденциальные данные должны обязательно защищаться, чтобы их не могли прочитать пользователи, не обладающие необходимыми правами. Это касается как данных, которые пересылаются по сети, так и хранимых данных.

Действительно, как минимум, при регистрации в любой мало-мальски защищенной системе следует указывать логин и пароль.

Естественно, логин и пароль могут храниться, как и в базе данных, так и в текстовом файле, где угодно, не имеет значения, в нашем случае это база данных MS SQL Server.

Проблема заключается в том, что если к такой БД получат доступ сторонние особы, то все логины/пароли пользователей будут «как на ладоне».

В этом случае целесообразно шифровать пароли. То есть, при регистрации нового пользователя, в БД заносится предварительно зашифрованный пароль пользователя, так сказать в закрытом виде.

Шифровать можно по своему собственному алгоритму, например, после каждого символа в пароле дописывать некоторый предварительно сформированный набор символов, или последовательность символов пароля, записанных наоборот, или еще что-нибудь, что вы придумаете сами.

Но такой подход чреват тем, что все-таки имеется возможность обнаружить ключ к расшифровыванию. Это способны достаточно быстро сделать специальные программы. Поэтому, в таких случаях следует отдавать предпочтение хеш-представлению пароля.

Для такой задачи существует алгоритм MD5. По данному алгоритму, возвращается значение в виде 32-разрядной шестнадцатеричной строки. Одним словом, имея любой длины набор символов (строку) и применив к ней алгоритм шифрования MD5, мы получим строку в 32 символа, причем в 16-ричном представлении.

Следует знать, что обратного алгоритма не существует. Расшифровке зашифрованные по MD5 данные не поддаются, а потому проверять пароль зарегистрированного пользователя при попытке входа следует также приводить вводимый пароль к хеш-значению и только после этого сравнивать с находящимся в БД.[7]

## 2.3. Разработка модели базы данных

Для базы данных eLearning было разработано 11 таблиц. В графической части 1 представлена диаграмма таблиц. База данных была разработана в СУБД MS SQL Server 2012.Описание всех таблиц приведено ниже.

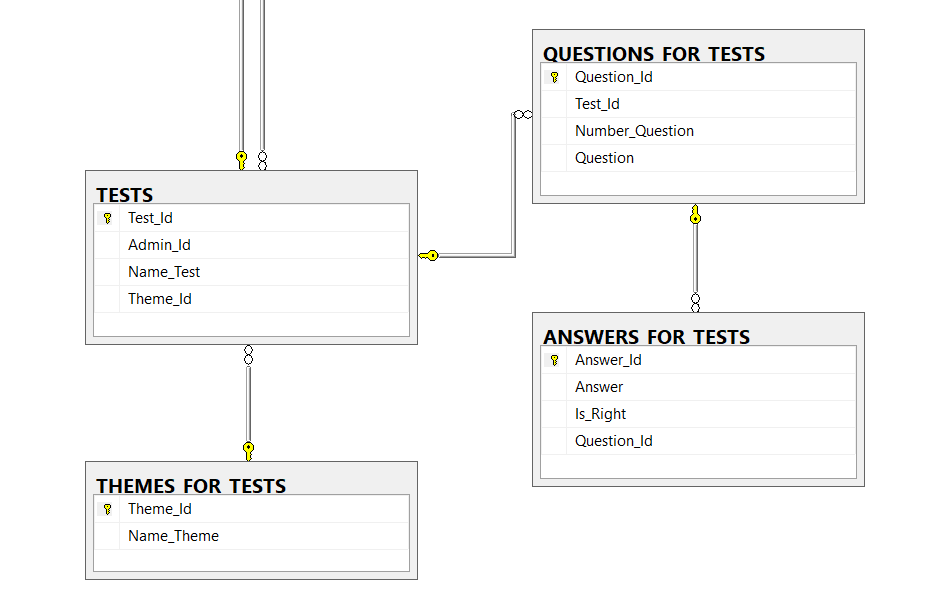


Рисунок 2.3.1 – Таблицы Tests, Questions\_For\_Tests, Themes\_For\_Tests, Answers\_For\_Tests

Таблица **Tests** содержит в себе тесты по определенным темам:

* Test\_Id (первичный ключ) – идентификатор теста, int, not null;
* Admin\_Id(вторичный ключ) – идентификатор администратора, int, null;
* Name\_Test – название теста, nvarchar(50), null;
* Theme\_Id – идентификатор темы, nvarchar(50), null.

Таблица **Questions\_For\_Tests** представляет собой набор вопросов к определенному тесту:

– Question\_Id (первичный ключ) – идентификатор вопроса, int, not null;

* Test\_Id (вторичный ключ) – идентификатор теста, int, null;
* Number\_Question – номер вопроса, int, null;
* Question – название вопроса, nvarchar(100), null.

Таблица **Themes\_For\_Tests** представляет собой перечень тем для создания теста:

* Theme\_Id (первичный ключ) – идентификатор темы, int, not null;
* Name\_Theme – название темы, nvarchar(100), null.

Таблица **Answers\_For**\_**Tests** содержит все ответы на вопросы:

* Answer\_Id (первичный ключ) – идентификатор ответов, int, not null;
* Answer – название ответов, nvarchar(50), null;
* Is\_Right – тип ответа(положительный-1, отрицательный-0), bit, null;
* Question\_Id (вторичный ключ) – идентификатор вопроса, int, null.

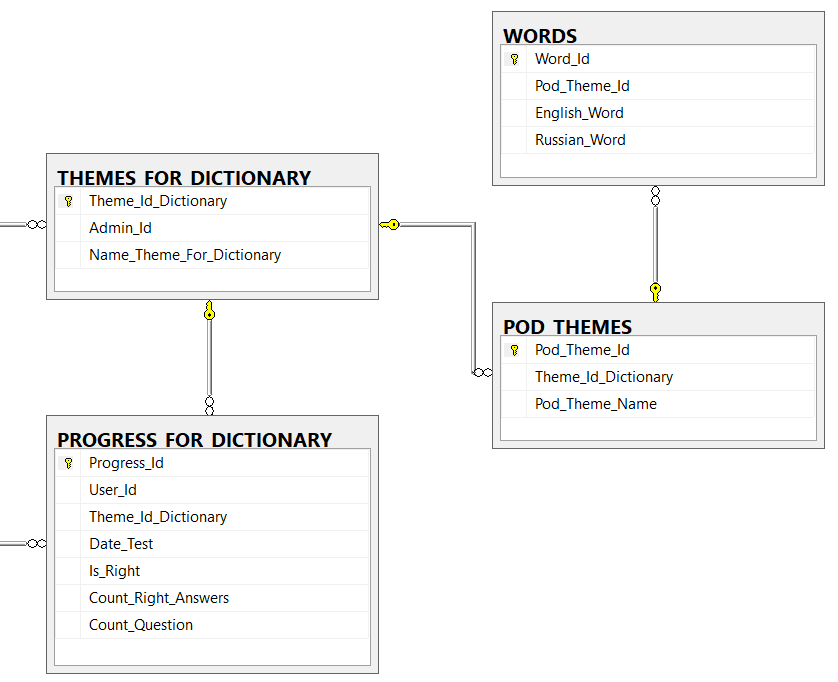


Рисунок 3.2 – Таблицы Themes\_For\_Dictionary, Progress\_For\_Dictionary, Pod\_Themes, Words

Таблица **Themes\_For\_Dictionary** представляет собой перечень тем для словаря:

* Theme\_Id\_Dictionary (первичный ключ) – идентификатор темы по словарю, int, not null;
* Admin\_Id (вторичный ключ) – идентификатор администратора, int, null;
* Name\_Theme\_For\_Dictionary – название темы для словаря, nvarchar(50), null.

Таблица **Pod\_Themes** содержит в тебе под темы к определенной теме словаря:

* Pod\_Theme\_Id (первичный ключ) – идентификатор под темы, int, not null;
* Theme\_Id\_Dictionary (вторичный ключ) – идентификатор темы по словарю, int, null;
* Pod\_Theme\_Name – название под темы, nvarchar(50), null.

Таблица **Words** содержит в себе слова на английском языке с переводом на русский язык :

* Word\_Id (первичный ключ) – идентификатор слов, int, not null;
* Pod\_Theme\_Id (вторичный ключ) – идентификатор под темы, int, null;
* English\_Word – слово на английском языке, nvarchar(30), null;
* Russian\_Word – слово на русском языке, nvarchar(30), null.

Таблица **Progress\_For\_Dictionary**содержит в себе все пройденные и не пройденные тесты на перевод слов:

* Progress\_Id (первичный ключ) – идентификатор, int, not null;
* User\_Id (вторичный ключ) – идентификатор пользователя, int, null;
* Theme\_Id\_Dictionary (вторичный ключ) – идентификатор темы по словарю, int, null;
* Date\_Test – дата прохождения теста, nvarchar(50), null;
* Is\_Right – тип ответа(положительный-1, отрицательный-0), bit, null;
* Count\_Right\_Answers – количество правильных ответов, int, null;
* Count\_Question – количество вопросов, int, null.

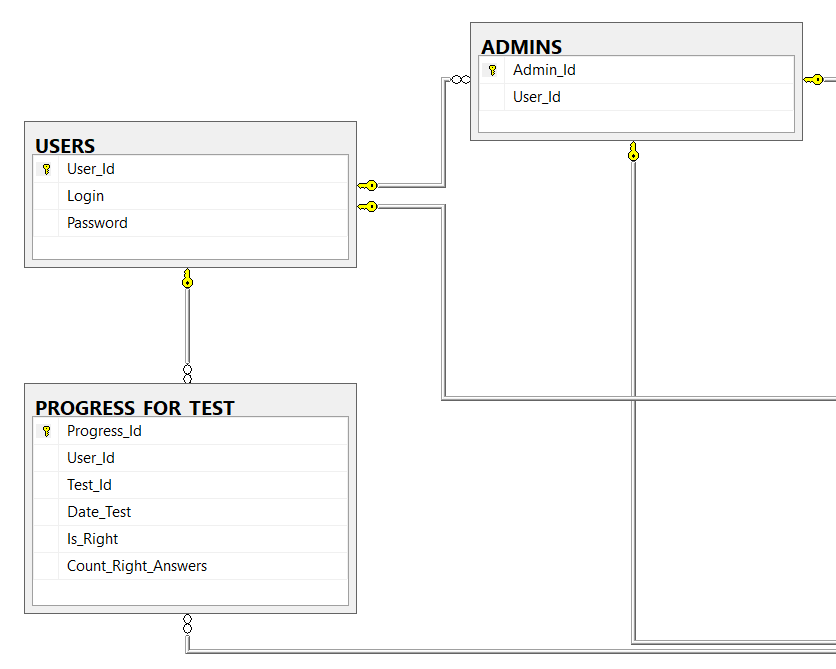


Рисунок 3.3 – Таблицы Progress\_For\_Test, Users, Admins

Таблица **Progress\_For\_Test** содержит в себе все пройденные и не пройденные тесты, которые создал администратор на определенные темы:

* Progress\_Id (первичный ключ) – идентификатор, int, not null;
* User\_Id (вторичный ключ) – идентификатор пользователя, int, null;
* Test\_Id (вторичный ключ) – идентификатор теста, int, null;
* Date\_Test – дата прохождения теста, nvarchar(50), null;
* Is\_Right – тип ответа(положительный-1, отрицательный-0), bit, null;
* Count\_Right\_Answers – количество правильных ответов, int, null.

Таблица **Users** хранит список всех пользователей:

* User\_Id (первичный ключ) – идентификатор пользователя, int, not null;
* Login – имя пользователя, nvarchar(30), not null;
* Password – пароль пользователя, nvarchar(15), not null.

Таблица **Admins** представляет собой список администраторов:

* Admin\_Id (первичный ключ) – идентификатор администратора, int, not null;
* User\_Id (вторичный ключ) – идентификатор пользователя, int, null.

Скрипты для создания всех таблиц базы данных представлены в Приложении А листинг для создания таблиц базы данных.

## 2.4. Используемые объекты базы данных

При разработке базы данных курсового проекта понадобилось создать следующие объекты:

* Таблицы
* Пользователи
* Процедуры
* Триггер

### **2.4.1. Т**а**блицы**

Ключевым объектом в базе данных являются таблицы. Таблицы состоят из строк и столбцов. Столбцы определяют тип информации, которая хранится, а строки содержат значения для этих столбцов. Именно в них хранится вся информация. При проектировании базы данных было создано 11 таблиц, которые подробно описаны выше, а SQL-скрипты для их создания находятся в приложении А.

### **2.4.2. Пользователи**

Пользователь базы данных – это учётная запись, которая имеет доступ к базе данных и пользуется услугами информационной системы для получения необходимой информации. В базе данных данного курсового проекта имеется 3 пользователя.

* Default\_User –пользователь, привилегии которого ограничены ролью User \_Role;
* Admin\_User – администратор базы данных, привилегии которого ограничены ролью Admin\_Role, владеющий всеми необходимыми привилегиями.

Код создания пользователей и ролей приведен в приложении А.

### **2.4.3. Хранимые процедуры**

Нередко операция с данными представляет набор инструкций, которые необходимо выполнить в определенной последовательности. И в этом случае более оптимально будет инкапсулировать все эти действия в один объект - хранимую процедуру (stored procedure).

Хранимая процедура – объект базы данных, представляющий собой набор SQL–инструкций, который компилируется один раз и хранится на сервере.  В хранимых процедурах могут выполняться стандартные операции с базами данных (как [DDL](https://ru.wikipedia.org/wiki/DDL), так и [DML](https://ru.wikipedia.org/wiki/DML)). Кроме того, в хранимых процедурах возможны циклы и ветвления, то есть в них могут использоваться инструкции управления процессом исполнения. В данном курсовом проекте они используются для предоставления доступа пользователю к данным. Код хранимых процедур находится ниже в приложении А. [8]

### **2.4.4. Триггеры**

Три́ггер ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *trigger*) — [хранимая процедура](https://ru.wikipedia.org/wiki/%D0%A5%D1%80%D0%B0%D0%BD%D0%B8%D0%BC%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0) особого типа, которую пользователь не вызывает непосредственно, а исполнение которой обусловлено действием по модификации данных: добавлением INSERT, удалением DELETE строки в заданной таблице, или изменением UPDATE данных в определённом столбце заданной таблицы [реляционной базы данных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Триггеры применяются для обеспечения целостности данных и реализации сложной [бизнес-логики](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B7%D0%BD%D0%B5%D1%81-%D0%BB%D0%BE%D0%B3%D0%B8%D0%BA%D0%B0). Триггер запускается сервером автоматически при попытке изменения данных в таблице, с которой он связан. Все производимые им модификации данных рассматриваются как выполняемые в [транзакции](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D1%8F_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), в которой выполнено действие, вызвавшее срабатывание триггера. Соответственно, в случае обнаружения ошибки или нарушения целостности данных может произойти откат этой транзакции.

В данной курсовой работе был реализован триггер для возможности удалять тест в котором нет ни одного вопроса. Администратор имеет возможность удалять вопросы из нужного ему теста и когда в этом тесте не будет вопросов он удалится.

Код триггера предоставлен на листинге 2.4.4.

Листинг 2.4.4 – Реализация триггера Question\_Removed

|  |
| --- |
| CREATE TRIGGER Question\_Removed  ON QUESTIONS\_FOR\_TESTS AFTER DELETE  AS  BEGIN  DELETE p FROM PROGRESS\_FOR\_TEST p  WHERE 0 = (SELECT COUNT(\*) FROM QUESTIONS\_FOR\_TESTS q WHERE q.Test\_Id = p.Test\_Id);  DELETE t FROM TESTS t  WHERE 0 = (SELECT COUNT(\*) FROM QUESTIONS\_FOR\_TESTS q WHERE q.Test\_Id = t.Test\_Id);  END |

# 3. Реализация программного средства

ФИО

Подпись

Дата

Лист

1

КП 03.06.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

3. Реализация программного средства

Лит.

Листов

7

БГТУ 71172586, 2019

У

## 3.1 Постановка задачи

Основной задачей курсового проекта является разработка десктопного приложения, позволяющего пользователю изучать язык и проходить тестирования по различным темам и администратору добавлять новые тесты, следить за пользователями, так же удалять вопросы. В данном курсовом проекте требовалось реализовать следующие задачи:

* Сохранять рабочую информацию в базе данных;
* Выбирать информацию из базы данных;
* Регистрация и авторизация пользователей;
* Прохождение тестов;
* Создание тестов, вопросов, ответов;
* Удаление вопросов и теста, в котором нет вопросов;
* Запись информации в xml-файл;
* Считывание информации из xml-файла;
* Разделение ролей пользователей;
* Шифрование данных клиента.

## 3.2 Обобщенная структура проекта

Решение представлено одним проектом eLearning, имеющим структуру, представленную на рисунке 3.2.

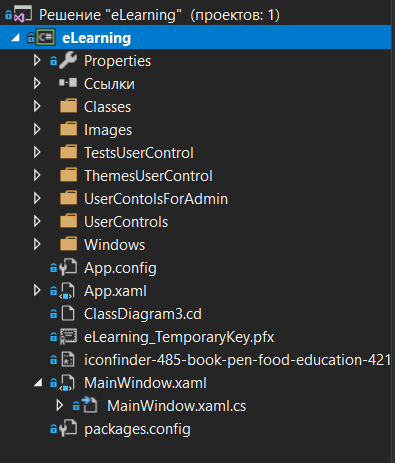


Рисунок 3.2 – Обобщенная структура проекта

Описание обобщенной структуры проекта eLearning представлено в таблице 3.2.

Таблица 3.2 – Обобщенная структура проекта

|  |  |
| --- | --- |
| Объект | Описание |
| Properties | Свойства проекта, содержит информацию о сборке, используемых ресурсах и настройках |
| Ссылки | Перечень сборок, используемых в проекте |
| Папка Classes | Содержит в себе основные классы |
| Папка Images | Содержит в себе картинки, используемые в проекте |
| Папка TestsUserControl | Содержит два xalm файла: TestArticles и TestDictionary. Они представляют собой динамическую форму для тесто |
| Папка ThemesUserControl | Содержит 4 пользовательских элемента: Adjctive, Articles, Noun, Tences. Представляют описание определенной темы |
| Папка UserControlsForAdmin | Содержит в себе 3 пользовательских элемента (CreatorTests, DialogWindow, Information) и 1 класс (Theme), используемый для создания темы |
| Папка UserControls | Состоит из 4 основных элементов интерфейса пользователя: Dictionary, Progress, Themes, Topics |
| Папка Windows | Содержит в себе окна регистрации и авторизации |
| App.config | Файл с параметрами проекта |
| App | Класс Application |
| ClassDiagram3 | Диаграмма классов |
| MainWindow | Главное окно проекта |

## 3.3 Описание процедур импорта и экспорта

Импорт и экспорт реализованы для таблицы Tests.

### **3.3.1. Описание процедуры экспорта**

Экспорт данных – выгрузка цифровой информации из базы данных с преобразованием формата. В нашем случае это формат XML.

В данной курсовой работе для того, чтобы провести экспорт данных из таблиц Tests необходимо выполнить SELECT запрос и использовать FOR XML PATH (‘Test’), ROOT (‘Tests’) для генерации XML–документа, более удобного для восприятия человеком.

Код процедур экспорта данных представлен в приложении Б.

### **3.3.2. Описание процедуры импорта**

XML - это формат данных, используемый для обмена данными в форме, которая может быть легко использована и распространена. Часто возникает необходимость импортировать XML-файлы в SQL Server, в данной курсовой работе используется только команды T-SQL.

Для того, чтобы использовать импорт из XML, в курсовом проекте были использованы ключевые функции.

Для начала нужно сделать чтобы SQL Server прочитал файл XML и импортировал данные с помощью [функции OPENROWSET](https://docs.microsoft.com/en-us/sql/t-sql/functions/openrowset-transact-sql?view=sql-server-2017) . Эта функция является родной для T-SQL и позволяет нам читать данные из разных типов файлов с помощью функции импорта BULK, которая позволяет импортировать из множества типов файлов, таких как XML. Столбцы в SELECT извлекаются из созданного псевдонима MY\_XML, и запрашиваем каждый элемент узла Test. Предложение FROM получено с помощью операции OPENROWSET с использованием опции BULK и опции SINGLE\_BLOB, чтобы данные возвращались из файла XML в один столбец и строку. Функция nodes() вместе с CROSS APPLY позволяет перемещаться по элементам XML, чтобы правильно инкапсулировать все объекты.

Код процедур импорта данных представлен в приложении Б. Сам XML–документ, полученный в результате выполнения соответствующей процедуры представлен в приложении В.

## 3.3. Регистрация и авторизация

Регистрация пользователя происходит в окне Registration. При регистрации предусмотрена валидация данных. Логин и пароль пользователя уникальны. Каждый пользователь сначала должен зарегистрироваться, после чего он сможет авторизоваться.

Для регистрации было разработано две процедуры GET\_USERS (получение всех пользователей) и ADD\_USERS (добавление новых пользователей в таблицу USERS). Код этих процедур приведен в листинге 3.3.

Листинг 3.3 – Реализация процедур GET\_USERS и ADD\_USERS

|  |
| --- |
| GO  CREATE PROCEDURE GET\_USERS AS  SELECT \* FROM USERS u LEFT JOIN ADMINS a ON a.User\_Id = u.User\_Id;  EXEC GET\_USERS;  GO  CREATE PROCEDURE ADD\_USERS  @login NVARCHAR(30),  @password NVARCHAR(15)  AS  BEGIN  INSERT INTO USERS([Login], [Password]) VALUES (@login, @password);  END |

Авторизация пользователя происходит в окне Login. Также предусмотрена валидация данных. Используется процедура GET\_USERS которая предоставлена выше в листинге 3.3.

## 3.4. Выбор нужного раздела

После регистрации или авторизации пользователь непосредственно попадает на главную страницу. В зависимости от роли, открывается необходимая страница. Более подробно про разделение на пользователей и их роли описано в разделе 2.4.2.

Интерфейс для пользователя (рисунок 3.4.1) содержит одно окно, состоящее из пользовательских классов: Topics, Dictionary, Themes и Progress. В свою очередь Topics содержит переход к классу TestArticles, Dictionary содержит переход к TestDictionary, Themes – Adjctive, Articles, Noun и Tences.

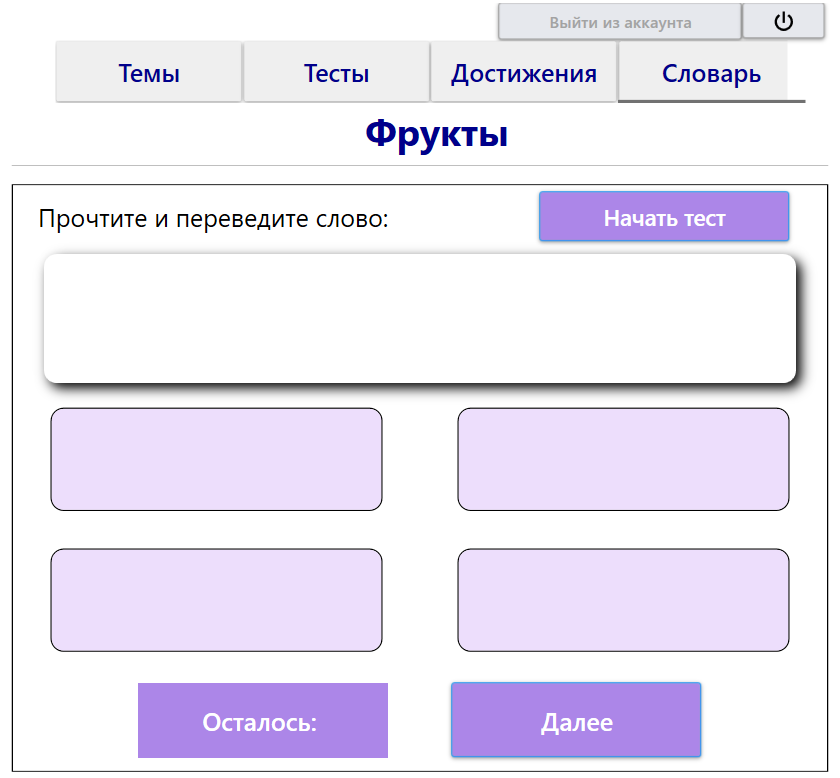


Рисунок 3.4.1 – Пример вкладки «Словарь»

Интерфейс для администратора представлен в том же окне, но содержит иные пользовательские классы. Создание тестов (рисунок 3.4.2) происходит с помощью класса CreatorTests.

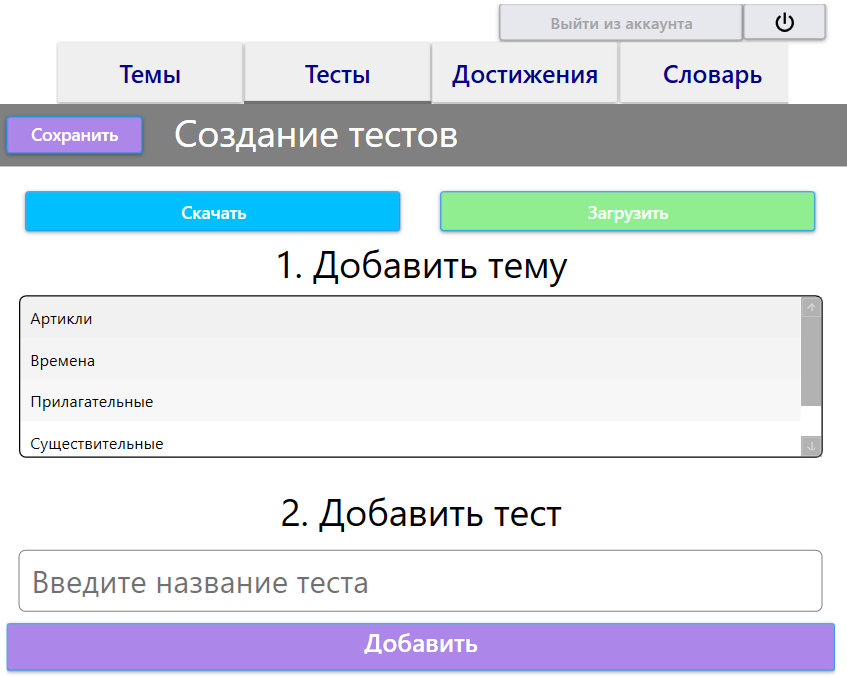


Рисунок 3.4.2 – Пример вкладки «Тесты»

В вкладке «Тесты» администратор имеет возможность создавать тесты для пользователя. Это происходит примерно так: выбирает определенную тему, создает тест, либо выбирает его из уже существующих, создает вопрос и ответы на данный вопрос, нажимает на кнопку сохранить.

Для просмотра информации реализует класс – Information. Наглядно представленно на рисунке 3.4.3.

Во вкладке «Достижения» администратор может посмотреть всех пользователей и перечень созданных тестов с названием вопроса.

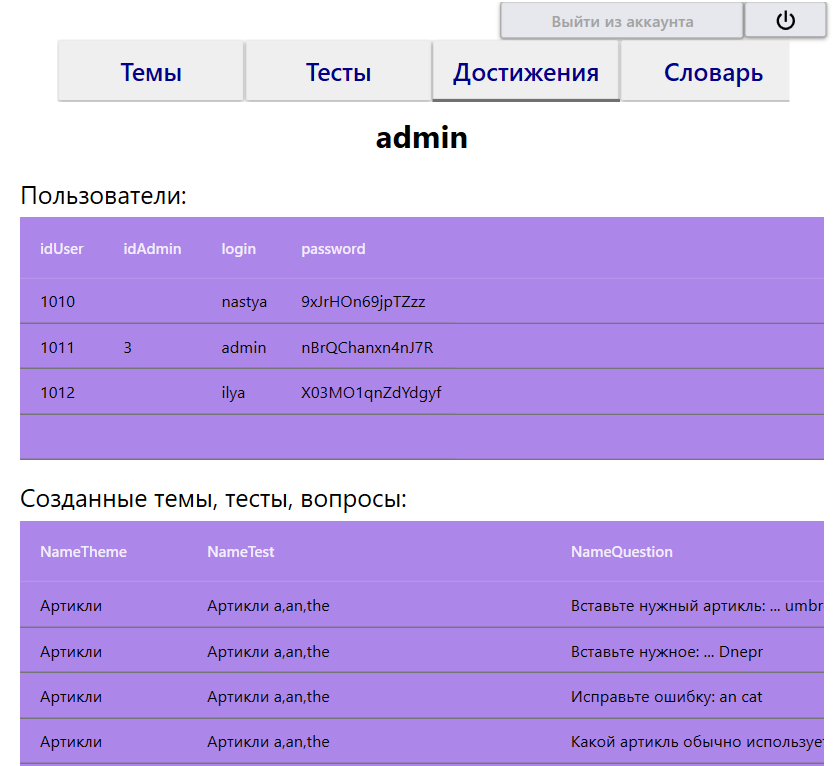


Рисунок 3.4.3 – Пример вкладки «Достижения»

Все необходимые процедуры, которые были созданы находятся в приложении А ниже.

# 4. Руководство пользователя

ФИО

Подпись

Дата

Лист

1

КП 04.07.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

4. Руководство пользователя

Лит.

Листов

5

БГТУ 71172586, 2019

У

При запуске приложения пользователю предоставлена возможность зарегистрироваться либо войти в систему (рисунок 4.1). При этом вход должен быть защищен паролем. Но не стоит забывать, что функция восстановления пароля в приложении не предусмотрена, поэтому к выбору пароля стоит относиться весьма внимательно.

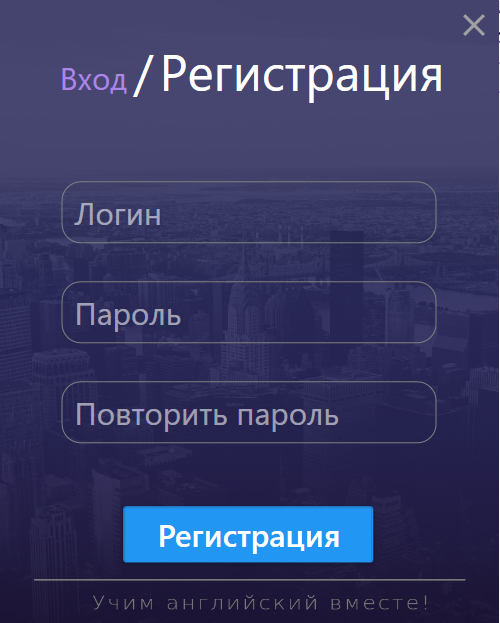


Рисунок 4.1 – Регистрация

Также в окне регистрации предусмотрен переход в окно авторизации пользователя (рисунок 4.2).

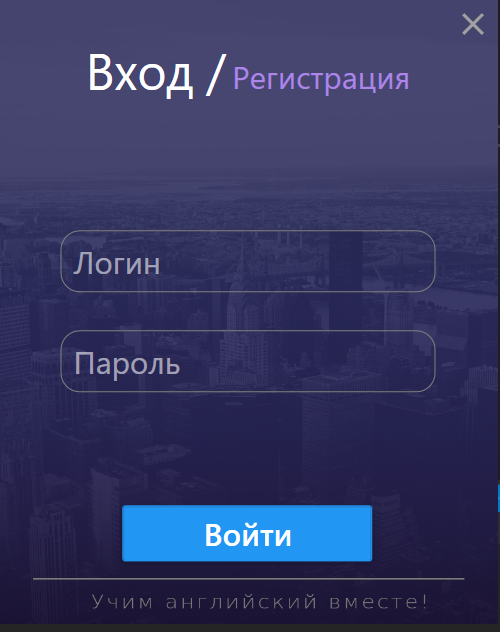


Рисунок 4.2 – Авторизация

После того, как пользователь создаст аккаунт и войдет в систему под своим именем, ему будем доступна работа приложения. Открывается новое окно, называемое рабочей страницей (рисунок 4.3).



Рисунок 4.3 – Основное окно

На данной странице пользователь может просматривать существующие темы (рисунок 4.4).

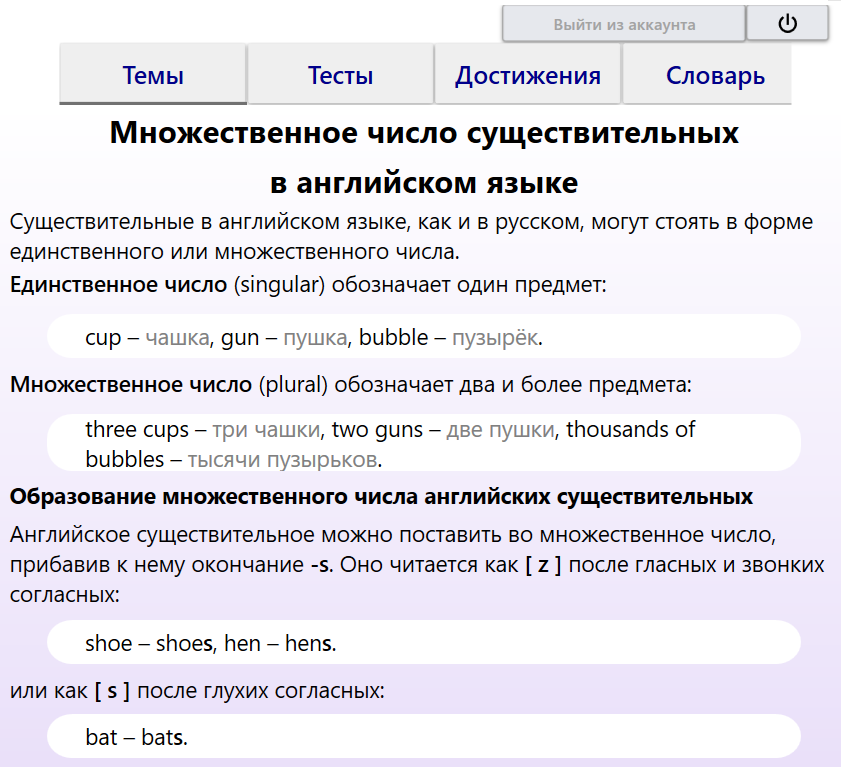


Рисунок 4.4 – Темы

При нажатии на вкладку «Тесты» пользователю станут доступны для прохождения тестов созданные ранее администратором (рисунок. 4.5).

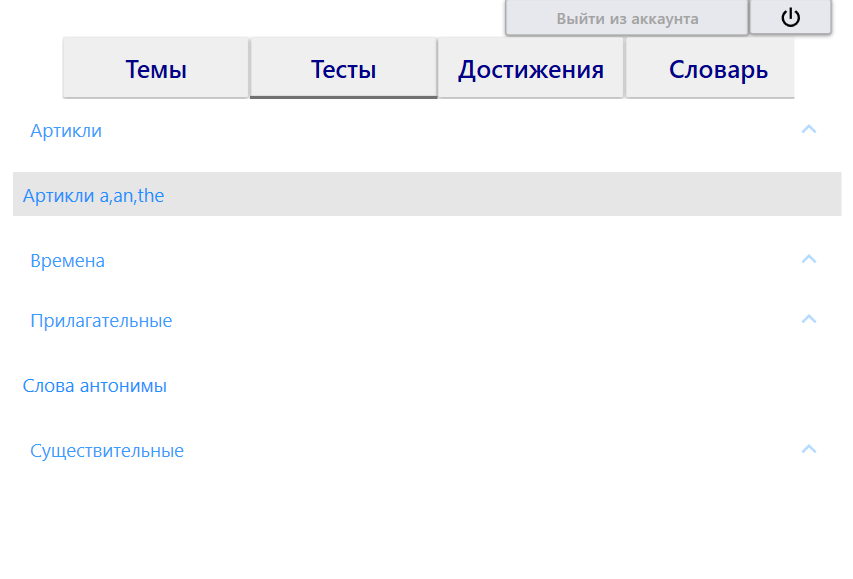


Рисунок 4.5 – Тесты

При переходе на вкладку «Словарь» пользователь доступ к словарю. Здесь он может изучать и слушать английские слова, а также проходить тесты на закрепление знаний (рисунок. 4.6).

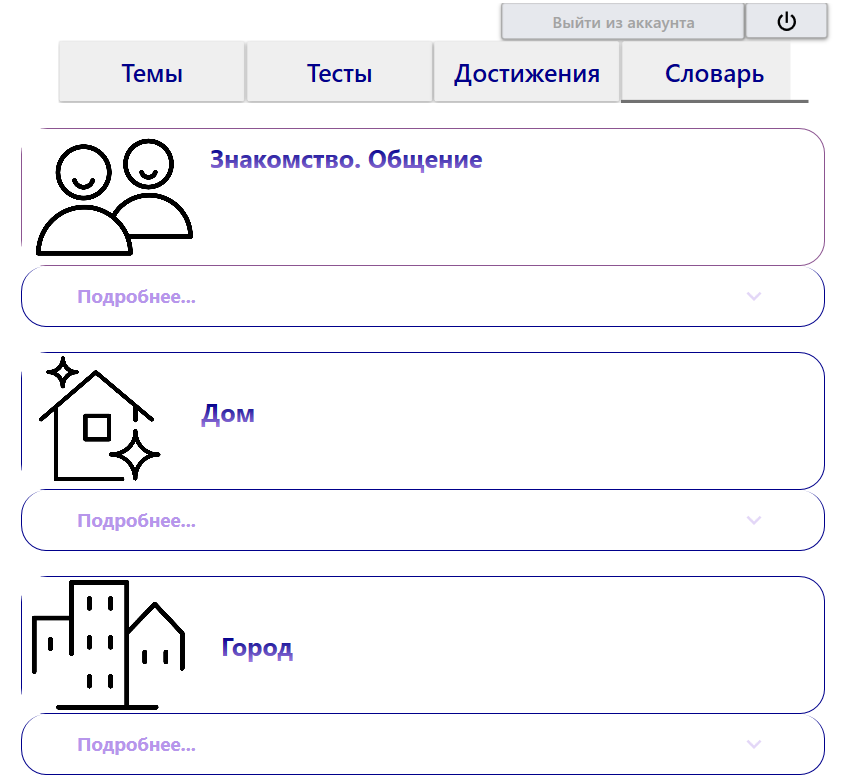


Рисунок 4.6 – Словарь

В каждой из тем есть свои под темы, при выборе определенной под темы будут предоставлены слова на английском языке с переводом на русский язык. Можно воспользоваться такой функцией как прослушивание слова. После ознакомления с новыми словами можно проверить себя пройдя тест на перевод этих слов. Данный тест представлен на рисунке 4.7.

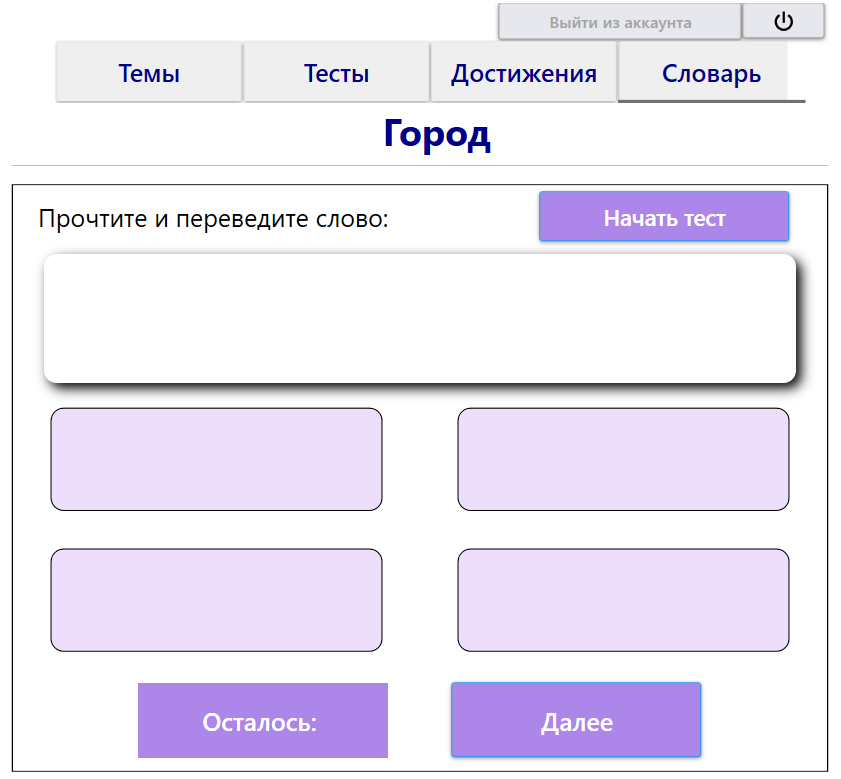


Рисунок 4.7 – Тест на перевод слов

На вкладке «Достижения» пользователь может видеть результаты пройденных тестов. Тест считается пройденным, если на каждый вопрос был дан правильный ответ (рисунок 4.8).

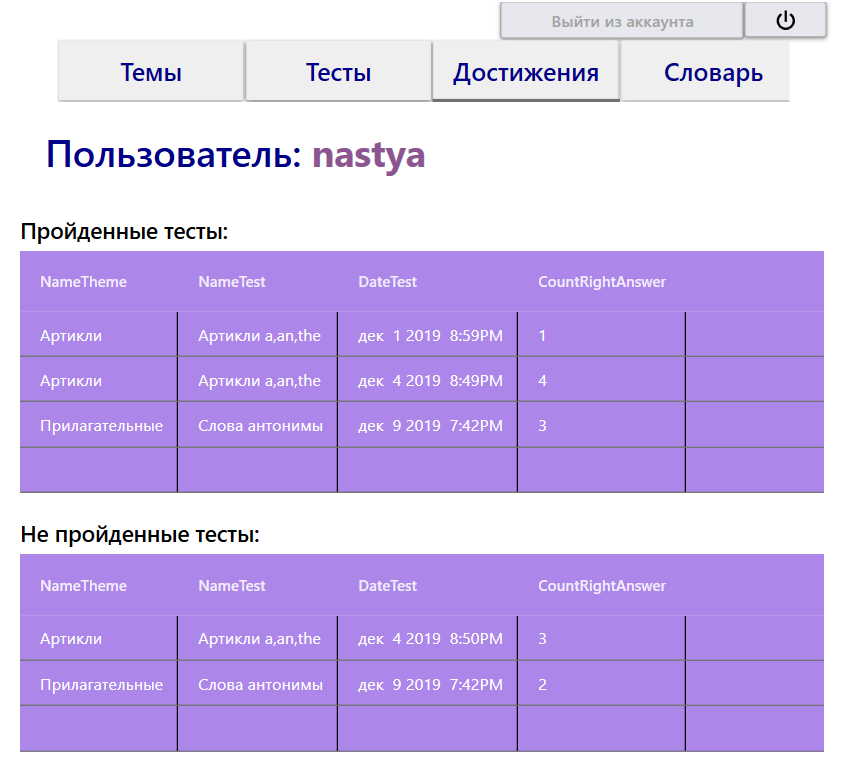


Рисунок 4.8 – Достижения

# 5. Тестирование производительности

ФИО

Подпись

Дата

Лист

1

КП 05.08.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

5. Тестирование производительности

Лит.

Листов

3

БГТУ 71172586, 2019

У

Для обеспечения корректности работы программы, обрабатываются различные ошибки, возникающие в процессе работы. Данное программное средство использует подключение к базе данных, следовательно, неправильно введенные данные или же их отсутствие может повлечь за собой неработоспособность приложения.

На рисунке 5.1 представлен результат обработки исключения при регистрации. Поле «Пароль» должно состоять минимум из 6 символов.

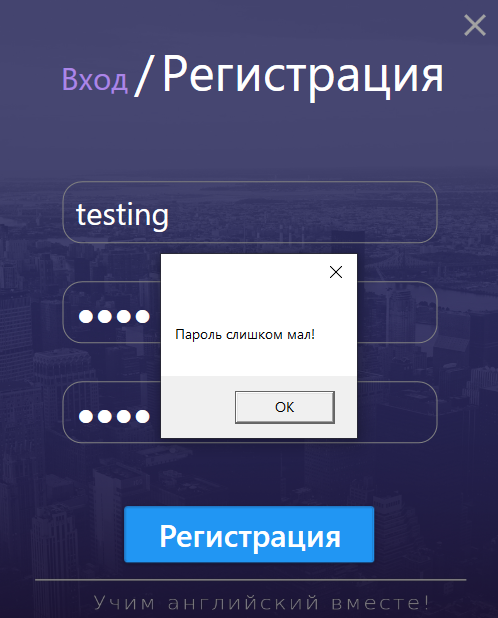


Рисунок 5.1 – Некорректный пароль

Так как поле «Логин» уникально, в базе данных не могут храниться пользователи с одинаковыми логинами (рисунок 5.2).

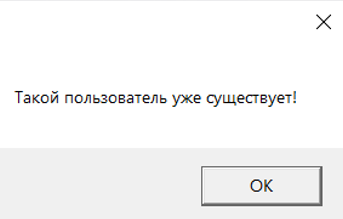


Рисунок 5.2 – Некорректный логин

На рисунке 5.3 представлен результат обработки исключения при входе в систему не указывая логин и пароль.

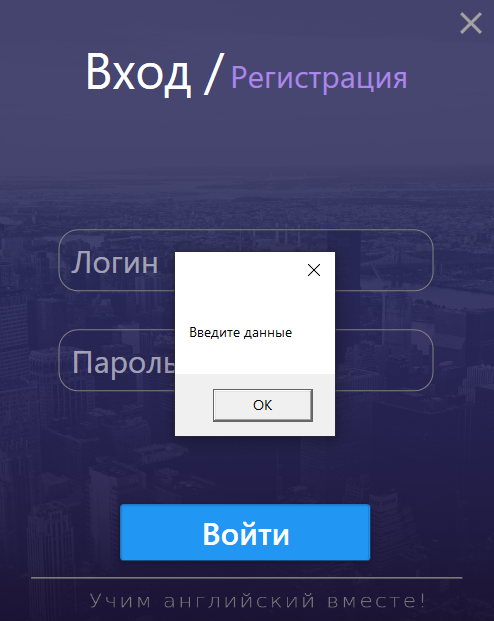


Рисунок 5.3 – Не введены данные

Также возникают ошибки, если пользователь пытается ввести неверные данные в поле логин или пароль (рисунок 5.4).

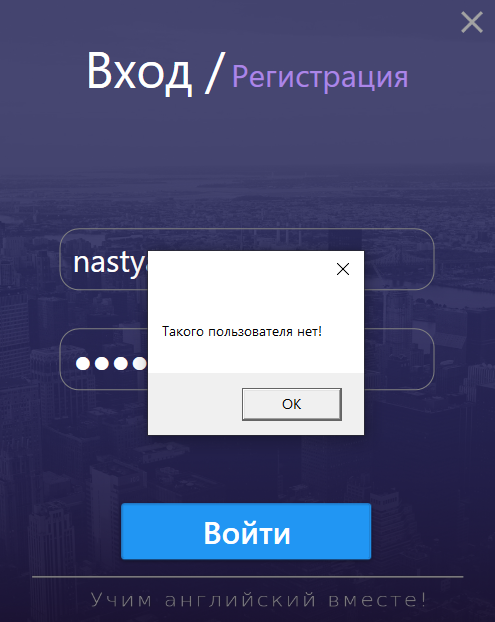


Рисунок 5.4 – Неверно введены данные

При создании тестов администратор должен помнить что поля «Тема», «Тест», «Вопрос» и «Ответы» должны быть заполнены (рисунок 5.5).

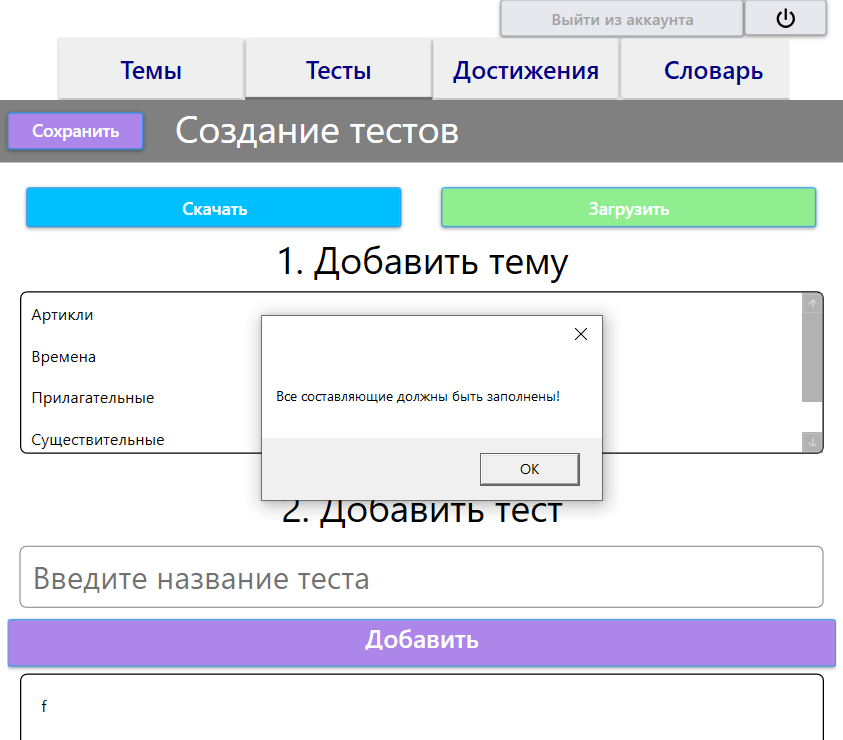


Рисунок 5.5 – Не введены данные

При добавлении ответов должен быть выбран вопрос к которому относятся ответы (рисунок 6.6).

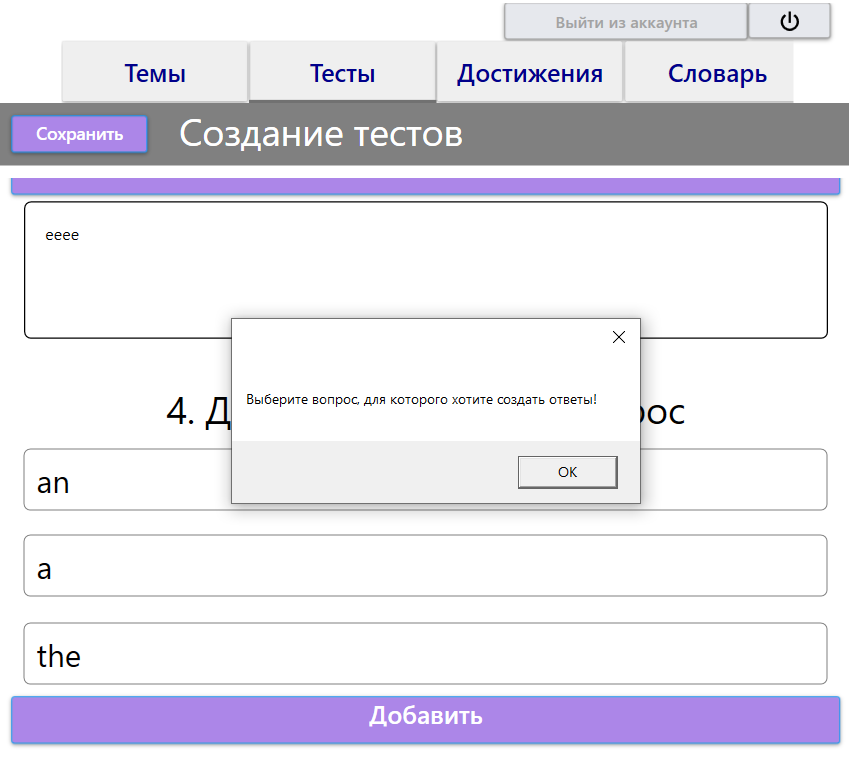


Рисунок 5.6 – Не указан вопрос

Большинство возникающих ошибок в данном приложении были откорректированы, однако все остальные действия обернуты в try .. catch, что даёт возможность уведомить пользователя о непредвиденной ошибки.

# Заключение

ФИО

Подпись

Дата

Лист

1

КП 00.09.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

Заключение

Лит.

Листов

1

БГТУ 71172586, 2019

У

В данном курсовом проекте была разработана база данных «eLearning», хранимые процедуры, триггер, а также настроены пользователь и администратор и их права на выполнение тех или иных процедур.

Были разработаны процедуры для реализации следующих целей:

* Прохождение тестов;
* Создание тестов, вопросов, ответов;
* Удаление вопросов и ответов;
* Просмотр необходимой информации;
* Запись информации в xml-файл;
* Считывание информации из xml-файла.

Доступ к данным, а также манипулирование ими осуществляется посредством процедур.

В результате выполнения курсового проекта было получено рабочее десктопное приложение «eLearning – Учим английский вместе!».

В программном средстве были выполнены все пункты из указанного списка основного функционала приложения, включающие в себя роли обычного пользователя и администратора, разработка тестов по английскому языку, прохождение тестов и просмотр достижений в изучении языка.

# Список используемых источников

ФИО

Подпись

Дата

Лист

1

КП 00.10.ПЗ

Разраб.

Септилко А.А.

Провер.

Бондарчик Е.Н.

Н. контр.

Бондарчик Е.Н.

Утверд.

Бондарчик Е.Н.

Список используемых источников

Лит.

Листов

1

БГТУ 71172586, 2019

У

1. Понятие и функции системы управления базами данных [Электронный ресурс]. — Режим доступа: <https://vuzlit.ru/998046/ponyatie_funktsii_sistemy_upravleniya_bazami_dannyh>. Дата доступа: 13.12.2019.
2. Простой способ импортировать данные XML в SQL Server с помощью T-SQL[Электронный ресурс]. — Режим доступа: [https://www.mssqltips.com/sqlservertip/5707/simple-way-to-import-xml-data-into-sql-server-with-tsql/. Дата](https://www.mssqltips.com/sqlservertip/5707/simple-way-to-import-xml-data-into-sql-server-with-tsql/.%20Дата) доступа: 14.12.2019.
3. Полное руководство по языку программирования С# 7.0 и платформе .NET 4.7 [Электронный ресурс] <https://metanit.com/sharp/tutorial/> Дата доступа:12.10.2019
4. WPF: Windows Presentation Foundation в .NET 4.0 с примерами на C# / М. Макдональд. – Вильямс, 2011. – 1024 с.
5. Design Com WPF – C# WPF UI TabMenu [Электронный ресурс] <https://www.youtube.com/watch?v=Jma5VYwEQcY> Дата доступа:21.11.2019
6. Руководство по ADO.NET и работе с базами данных [Электронный ресурс] – <https://metanit.com/sharp/adonet/>. Дата доступа: 16.10.2019
7. Шифрование на c# (хеш-код строки по алгоритму MD5)[Электронный ресурс] <http://skillcoding.com/Default.aspx?id=165> Дата доступа:11.12.2019
8. Хранимые процедуры. Создание и выполнение процедур [Электронный ресурс] <https://metanit.com/sql/sqlserver/11.1.php> Дата доступа:12.12.2019

# Приложение A

Листинг создания таблиц базы данных eLearning:

CREATE TABLE USERS(

[User\_Id] int PRIMARY KEY IDENTITY,

[Login] nvarchar(30) NOT NULL,

[Password] nvarchar(15) NOT NULL

)

CREATE TABLE ADMINS(

[Admin\_Id] int PRIMARY KEY IDENTITY,

[User\_Id] int references USERS (User\_Id)

)

CREATE TABLE PROGRESS\_FOR\_TEST

(

[Progress\_Id] int PRIMARY KEY IDENTITY,

[User\_Id] int references USERS (User\_Id),

[Test\_Id] int references TESTS (Test\_Id),

[Date\_Test] nvarchar(50),

[Is\_Right] bit,

[Count\_Right\_Answers] int

)

CREATE TABLE TESTS

(

[Test\_Id] int PRIMARY KEY IDENTITY,

[Admin\_Id] int references ADMINS (Admin\_Id),

[Name\_Test] nvarchar(50) UNIQUE,

[Theme\_Id] int references THEMES\_FOR\_TESTS (Theme\_Id)

)

CREATE TABLE THEMES\_FOR\_TESTS

(

[Theme\_Id] int PRIMARY KEY IDENTITY,

[Name\_Theme] nvarchar(100) UNIQUE

)

CREATE TABLE QUESTIONS\_FOR\_TESTS

(

[Question\_Id] int PRIMARY KEY IDENTITY,

[Test\_Id] int references TESTS (Test\_Id),

[Number\_Question] int,

[Question] nvarchar(100) UNIQUE

)

CREATE TABLE ANSWERS\_FOR\_TESTS

(

[Answer\_Id] int PRIMARY KEY IDENTITY,

[Answer] nvarchar(50),

[Is\_Right] bit,

[Question\_Id] int references QUESTIONS\_FOR\_TESTS (Question\_Id) ,

)

CREATE TABLE PROGRESS\_FOR\_DICTIONARY

(

[Progress\_Id] int PRIMARY KEY IDENTITY,

[User\_Id] int references USERS (User\_Id),

[Theme\_Id\_Dictionary] int references THEMES\_FOR\_DICTIONARY (Theme\_Id\_Dictionary),

[Date\_Test] nvarchar(50),

[Is\_Right] bit,

[Count\_Right\_Answers] int,

[Count\_Question] int

)

CREATE TABLE THEMES\_FOR\_DICTIONARY

(

[Theme\_Id\_Dictionary] int PRIMARY KEY,

[Admin\_Id] int references ADMINS (Admin\_Id),

[Name\_Theme\_For\_Dictionary] nvarchar(50)

)

CREATE TABLE POD\_THEMES

(

[Pod\_Theme\_Id] int PRIMARY KEY,

[Theme\_Id\_Dictionary] int references THEMES\_FOR\_DICTIONARY (Theme\_Id\_Dictionary),

[Pod\_Theme\_Name] nvarchar(50)

CREATE TABLE WORDS

(

[Word\_Id] int PRIMARY KEY,

[Pod\_Theme\_Id] int references POD\_THEMES (Pod\_Theme\_Id),

[English\_Word] nvarchar(30),

[Russian\_Word] nvarchar(30)

)

Листинг создания хранимых процедур:

GO

CREATE PROCEDURE GET\_USERS AS

SELECT \* FROM USERS u LEFT JOIN ADMINS a ON a.User\_Id = u.User\_Id;

EXEC GET\_USERS;

GO

CREATE PROCEDURE ADD\_USERS

@login NVARCHAR(30),

@password NVARCHAR(15)

AS

BEGIN

INSERT INTO USERS([Login], [Password]) VALUES (@login, @password);

END

GO

CREATE PROCEDURE NO\_PASSED\_FOR\_TEST

@user\_Id int

AS

BEGIN

SELECT THEMES\_FOR\_TESTS.Name\_Theme, TESTS.Name\_Test, PROGRESS\_FOR\_TEST.Date\_Test, PROGRESS\_FOR\_TEST.Count\_Right\_Answers

FROM PROGRESS\_FOR\_TEST

JOIN TESTS ON PROGRESS\_FOR\_TEST.Test\_Id = TESTS.Test\_Id

JOIN THEMES\_FOR\_TESTS ON TESTS.Theme\_Id = THEMES\_FOR\_TESTS.Theme\_Id

WHERE PROGRESS\_FOR\_TEST.Is\_Right = 0 AND User\_Id = @user\_Id;

END

GO

CREATE PROCEDURE PASSED\_FOR\_TEST

@user\_Id int

AS

BEGIN

SELECT THEMES\_FOR\_TESTS.Name\_Theme, TESTS.Name\_Test, PROGRESS\_FOR\_TEST.Date\_Test, PROGRESS\_FOR\_TEST.Count\_Right\_Answers

FROM PROGRESS\_FOR\_TEST

JOIN TESTS ON PROGRESS\_FOR\_TEST.Test\_Id = TESTS.Test\_Id

JOIN THEMES\_FOR\_TESTS ON TESTS.Theme\_Id = THEMES\_FOR\_TESTS.Theme\_Id

WHERE PROGRESS\_FOR\_TEST.Is\_Right = 1 AND User\_Id = @user\_Id;

END

GO

CREATE PROCEDURE NO\_PASSED\_FOR\_DICTIONARY

@user\_Id int

AS

BEGIN

SELECT \* FROM PROGRESS\_FOR\_DICTIONARY pr

JOIN THEMES\_FOR\_DICTIONARY tm ON tm.Theme\_Id\_Dictionary = pr.Theme\_Id\_Dictionary

WHERE User\_Id = @user\_Id AND Is\_Right = 0;

END

GO

CREATE PROCEDURE PASSED\_FOR\_DICTIONARY

@user\_Id int

AS

BEGIN

SELECT \* FROM PROGRESS\_FOR\_DICTIONARY pr

JOIN THEMES\_FOR\_DICTIONARY tm ON tm.Theme\_Id\_Dictionary = pr.Theme\_Id\_Dictionary

WHERE User\_Id = @user\_Id AND Is\_Right = 1;

END

GO

CREATE PROCEDURE GET\_THEME\_FOR\_TEST AS

SELECT \* FROM THEMES\_FOR\_TESTS

EXEC GET\_THEME\_FOR\_TEST ;

GO

CREATE PROCEDURE GET\_ADMINS AS

SELECT \* FROM ADMINS

EXEC GET\_ADMINS;

GO

CREATE PROCEDURE GET\_TESTS\_TEST AS

SELECT \* FROM TESTS;

EXEC GET\_TESTS\_TEST ;

GO

CREATE PROCEDURE GET\_QUESTIONS AS

SELECT \* FROM QUESTIONS\_FOR\_TESTS

EXEC GET\_QUESTIONS;

GO

CREATE PROCEDURE GET\_TESTS\_FOR\_TEST

@theme\_Id int

AS

BEGIN

SELECT \* FROM TESTS WHERE Theme\_Id = @theme\_Id

END

GO

CREATE PROCEDURE Add\_TESTS

@name\_test NVARCHAR(50),

@admin\_id int,

@theme\_Id int

AS

BEGIN

INSERT INTO TESTS([Name\_Test], [Admin\_Id], [Theme\_Id]) VALUES (@name\_test, @admin\_id, @theme\_Id);

END

GO

CREATE PROCEDURE Add\_QUESTIONS

@test\_Id int,

@number\_question int,

@question NVARCHAR(100)

AS

BEGIN

INSERT INTO QUESTIONS\_FOR\_TESTS([Test\_Id], [Number\_Question], [Question]) VALUES (@test\_Id, @number\_question, @question);

END

GO

CREATE PROCEDURE Add\_ANSWER

@answer NVARCHAR(50),

@is\_right bit,

@question\_Id int

AS

BEGIN

INSERT INTO ANSWERS\_FOR\_TESTS([Answer], [Is\_Right], [Question\_Id]) VALUES (@answer, @is\_right, @question\_Id);

END

GO

CREATE PROCEDURE JOIN\_QUESTION

@name\_test NVARCHAR(50)

AS

BEGIN

SELECT QUESTIONS\_FOR\_TESTS.Question FROM QUESTIONS\_FOR\_TESTS

JOIN TESTS ON QUESTIONS\_FOR\_TESTS.Test\_Id = TESTS.Test\_Id

WHERE TESTS.Name\_Test = @name\_test;

END

GO

CREATE PROCEDURE GET\_INFORMATION AS

BEGIN

SELECT THEMES\_FOR\_TESTS.Name\_Theme, TESTS.Name\_Test, QUESTIONS\_FOR\_TESTS.Question FROM THEMES\_FOR\_TESTS

JOIN TESTS ON THEMES\_FOR\_TESTS.Theme\_Id = TESTS.Theme\_Id

LEFT JOIN QUESTIONS\_FOR\_TESTS ON TESTS.Test\_Id = QUESTIONS\_FOR\_TESTS.Test\_Id

END

EXEC GET\_INFORMATION;

GO

CREATE PROCEDURE DELETE\_ANSWER

@question nvarchar (100)

AS

BEGIN

DELETE a FROM ANSWERS\_FOR\_TESTS a INNER JOIN

QUESTIONS\_FOR\_TESTS q ON a.Question\_Id = q.Question\_Id

WHERE q.Question = @question

END

GO

CREATE PROCEDURE DELETE\_QUESTIONS

@question nvarchar (100)

AS

BEGIN

DELETE FROM QUESTIONS\_FOR\_TESTS WHERE Question = @question;

END

GO

CREATE PROCEDURE TEST\_ARTICLES\_TESTS

@test\_Id int

AS

BEGIN

SELECT TESTS.Name\_Test, QUESTIONS\_FOR\_TESTS.Question\_Id, QUESTIONS\_FOR\_TESTS.Number\_Question, QUESTIONS\_FOR\_TESTS.Question FROM QUESTIONS\_FOR\_TESTS

join TESTS ON QUESTIONS\_FOR\_TESTS.Test\_Id = TESTS.Test\_Id

where TESTS.Test\_Id = @test\_Id;

END

GO

CREATE PROCEDURE TEST\_ARTICLES\_ANSWERS

@question NVARCHAR(100)

AS

BEGIN

SELECT ANSWERS\_FOR\_TESTS.Answer FROM ANSWERS\_FOR\_TESTS

JOIN QUESTIONS\_FOR\_TESTS on ANSWERS\_FOR\_TESTS.Question\_Id = QUESTIONS\_FOR\_TESTS.Question\_Id

where QUESTIONS\_FOR\_TESTS.Question = @question;

END

GO

CREATE PROCEDURE TEST\_ARTICLES\_QUESTIONS

@name\_test NVARCHAR(50)

AS

BEGIN

SELECT QUESTIONS\_FOR\_TESTS.Number\_Question, QUESTIONS\_FOR\_TESTS.Question, ANSWERS\_FOR\_TESTS.Answer FROM QUESTIONS\_FOR\_TESTS

JOIN ANSWERS\_FOR\_TESTS ON QUESTIONS\_FOR\_TESTS.Question\_Id = ANSWERS\_FOR\_TESTS.Question\_Id

JOIN TESTS ON QUESTIONS\_FOR\_TESTS.Test\_Id = TESTS.Test\_Id

WHERE ANSWERS\_FOR\_TESTS.Is\_Right = 1 AND TESTS.Name\_Test = @name\_test

END

GO

CREATE PROCEDURE ADD\_PROGRESS\_FOR\_TESTS

@user\_Id int,

@test\_Id int,

@date\_test nvarchar(50),

@is\_right bit,

@count\_right\_answer int

AS

BEGIN

INSERT INTO PROGRESS\_FOR\_TEST(User\_Id, Test\_Id, Date\_Test, Is\_Right, Count\_Right\_Answers)

VALUES (@user\_Id, @test\_Id, @date\_test, 1, @count\_right\_answer);

END

GO

CREATE PROCEDURE NO\_ADD\_PROGRESS\_FOR\_TESTS

@user\_Id int,

@test\_Id int,

@date\_test nvarchar(50),

@is\_right bit,

@count\_right\_answer int

AS

BEGIN

INSERT INTO PROGRESS\_FOR\_TEST(User\_Id, Test\_Id, Date\_Test, Is\_Right, Count\_Right\_Answers)

VALUES (@user\_Id, @test\_Id, @date\_test, 0, @count\_right\_answer);

END

GO

CREATE PROCEDURE GET\_THEME\_FOR\_DICTIONARY AS

BEGIN

SELECT \* FROM THEMES\_FOR\_DICTIONARY;

END

EXEC GET\_THEME\_FOR\_DICTIONARY;

GO

CREATE PROCEDURE GET\_POD\_THEMES AS

BEGIN

SELECT \* FROM POD\_THEMES;

END

EXEC GET\_POD\_THEMES;

GO

CREATE PROCEDURE TEST\_DICTIONARY

@theme\_id int,

@podtheme\_id int

AS

BEGIN

SELECT Words.Word\_Id, Words.English\_Word, Words.Russian\_Word

FROM Words

JOIN POD\_THEMES ON Words.Pod\_Theme\_Id = POD\_THEMES.Pod\_Theme\_Id

LEFT JOIN THEMES\_FOR\_DICTIONARY ON THEMES\_FOR\_DICTIONARY.Theme\_Id\_Dictionary = Pod\_Themes.Theme\_Id\_Dictionary

WHERE THEMES\_FOR\_DICTIONARY.Theme\_Id\_Dictionary = @theme\_id

AND POD\_THEMES.Pod\_Theme\_Id = @podtheme\_id;

END

GO

CREATE PROCEDURE ADD\_PROGRESS\_FOR\_DICTIONARY

@user\_Id int,

@theme\_Id\_Dictionary int,

@date\_test NVARCHAR(50),

@is\_right bit,

@count\_right\_answer int,

@count\_question int

AS

BEGIN

INSERT INTO PROGRESS\_FOR\_DICTIONARY(User\_Id, Theme\_Id\_Dictionary ,Date\_Test, Is\_Right, Count\_Right\_Answers, Count\_Question)

VALUES (@user\_Id,@theme\_Id\_Dictionary , @date\_test, @is\_right, @count\_right\_answer, @count\_question );

END

Добавление информации в базу данных для словаря:

INSERT INTO [THEMES\_FOR\_TESTS]

VALUES ('Артикли'),

('Времена'),

('Прилагательные'),

('Существительные');

INSERT INTO [POD\_THEMES] ([Pod\_Theme\_Id], [Theme\_Id\_Dictionary], [Pod\_Theme\_Name])

VALUES (1, 1, 'Приветсвия и прощания'),

(2, 1, 'Формулы вежливости'),

(3, 2, 'Дом, квартира'),

(4, 2, 'Мебель, интерьер'),

(5, 3, 'Город'),

(6, 3, 'Транспорт'),

(7, 4, 'Овощи'),

(8, 4, 'Фрукты');

INSERT INTO [Words] ([Word\_Id], [Pod\_Theme\_Id], [English\_Word], [Russian\_Word])

VALUES

-----DICTIONARY-1-1---------------------------------

(1, 1, 'Hello', 'Здравствуйте'),

(2, 1, 'Hi', 'Привет'),

(3, 1, 'Goodbye', 'До свидания'),

(4, 1, 'Bye', 'Пока'),

(5, 1, 'Good morning', 'Доброе утро'),

(6, 1, 'See you later', 'До встречи'),

(7, 1, 'Good night', 'Спокойной ночи'),

-----DICTIONARY-1-2---------------------------------

(8, 2, 'Thank you', 'Спасибо'),

(9, 2, 'Thanks a lot', 'Большое спасибо'),

(10, 2, 'You are welcome', 'Пожалуйста'),

(11, 2, 'Not at all', 'Не за что'),

(12, 2, 'No problem', 'Нет проблем'),

(13, 2, 'Excuse me', 'Простите'),

(14, 2, 'Thats OK', 'Всё нормально'),

-----DICTIONARY-2-1---------------------------------

(15, 3, 'house', 'дом'),

(16, 3, 'flat', 'квартира'),

(17, 3, 'door', 'дверь'),

(18, 3, 'floor', 'пол'),

(19, 3, 'wall', 'стена'),

(20, 3, 'window', 'окно'),

(21, 3, 'bedroom', 'спальня'),

-----DICTIONARY-2-2---------------------------------

(22, 4, 'bed', 'кровать'),

(23, 4, 'chair', 'стул'),

(24, 4, 'table', 'стол'),

(25, 4, 'lamp', 'лампа'),

(26, 4, 'picture', 'картина'),

(27, 4, 'carpet', 'ковёр'),

(28, 4, 'mirror', 'зеркало'),

-----DICTIONARY-3-1---------------------------------

(29, 5, 'address', 'адрес'),

(30, 5, 'bank', 'банк'),

(31, 5, 'bridge', 'мост'),

(32, 5, 'bus station', 'автовокзал'),

(33, 5, 'city', 'город'),

(34, 5, 'park', 'парк'),

(35, 5, 'monument', 'памятник'),

-----DICTIONARY-3-2---------------------------------

(36, 6, 'bus', 'автобус'),

(37, 6, 'car', 'машина'),

(38, 6, 'taxi', 'такси'),

(39, 6, 'tram', 'трамвай'),

(40, 6, 'subway', 'метро'),

(41, 6, 'trolleybus', 'троллейбус'),

(42, 6, 'driver', 'водитель'),

-----DICTIONARY-4-1---------------------------------

(43, 7, 'vegetables', 'овощи'),

(44, 7, 'cabbage', 'капуста'),

(45, 7, 'potatoes', 'помидор'),

(46, 7, 'radish', 'редис'),

(47, 7, 'corn', 'попкорн'),

(48, 7, 'pepper', 'перец'),

(49, 7, 'beans', 'бобы'),

-----DICTIONARY-4-2---------------------------------

(50, 8, 'fruit', 'фрукты'),

(51, 8, 'berries', 'ягоды'),

(52, 8, 'apple', 'яблоко'),

(53, 8, 'banana', 'банан'),

(54, 8, 'cherry', 'вишня'),

(55, 8, 'lemon', 'лимон'),

(56, 8, 'peach', 'персик');

Листинг создания пользователей и их ролей, раздача необходимых привилегий:

CREATE LOGIN [Admin\_User]

WITH PASSWORD=N'password',

DEFAULT\_DATABASE=[eLEARNING],

DEFAULT\_LANGUAGE=[русский],

CHECK\_EXPIRATION=OFF,

CHECK\_POLICY=ON;

CREATE LOGIN [Default\_User]

WITH PASSWORD=N'password',

DEFAULT\_DATABASE=[eLEARNING],

DEFAULT\_LANGUAGE=[русский],

CHECK\_EXPIRATION=OFF,

CHECK\_POLICY=ON;

GO

CREATE USER [Admin\_User] FOR LOGIN [Admin\_User] WITH DEFAULT\_SCHEMA=[dbo];

CREATE USER [Default\_User] FOR LOGIN [Default\_User] WITH DEFAULT\_SCHEMA=[dbo];

GO

CREATE ROLE ADMIN\_ROLE AUTHORIZATION db\_owner;

CREATE ROLE USER\_ROLE AUTHORIZATION db\_owner;

GRANT EXECUTE TO [ADMIN\_ROLE];

GRANT SELECT TO [ADMIN\_ROLE];

GRANT DELETE TO [ADMIN\_ROLE];

GRANT INSERT TO [ADMIN\_ROLE];

GRANT ALTER TO [ADMIN\_ROLE];

GRANT UPDATE TO [ADMIN\_ROLE];

GRANT ADMINISTER BULK OPERATIONS TO [Admin\_User];

GRANT EXECUTE TO [USER\_ROLE];

GRANT SELECT TO [USER\_ROLE];

ALTER ROLE ADMIN\_ROLE ADD MEMBER [Admin\_User];

ALTER ROLE USER\_ROLE ADD MEMBER [Default\_User];

EXEC xp\_instance\_regwrite N'HKEY\_LOCAL\_MACHINE',

N'Software\Microsoft\MSSQLServer\MSSQLServer',

N'LoginMode', REG\_DWORD, 1

GO

# Приложение Б

Листинг импорта и экспорта данных в формат xml:

GO

CREATE PROCEDURE Import\_To\_XML

AS

BEGIN

INSERT INTO TESTS(Admin\_Id, Name\_Test, Theme\_Id)

SELECT

MY\_XML.Test.query('Admin\_Id').value('.', 'int'),

MY\_XML.Test.query('Name\_Test').value('.', 'NVARCHAR(50)'),

MY\_XML.Test.query('Theme\_Id').value('.', 'int')

FROM (SELECT CAST(MY\_XML AS xml)

FROM OPENROWSET(BULK 'D:\Учёба\3 curs\DB\КП\Tests.xml', SINGLE\_BLOB) AS T(MY\_XML)) AS T(MY\_XML)

CROSS APPLY MY\_XML.nodes('Tests/Test') AS MY\_XML (Test);

END

EXEC Import\_To\_XML;

GO

CREATE PROCEDURE Export\_To\_XML

AS

BEGIN

SELECT \* FROM TESTS

for XML PATH('Test'), Root('Tests')

END

EXEC Export\_To\_XML;

# 

# Приложение В

<?xml version="1.0" encoding="utf-8"?>

<Tests>

<Test>

<Test\_Id>25</Test\_Id>

<Admin\_Id>3</Admin\_Id>

<Name\_Test>Артикли a,an,the</Name\_Test>

<Theme\_Id>1</Theme\_Id>

</Test>

<Test>

<Test\_Id>30</Test\_Id>

<Admin\_Id>3</Admin\_Id>

<Name\_Test>Слова антонимы</Name\_Test>

<Theme\_Id>4</Theme\_Id>

</Test>

<Test>

<Test\_Id>33</Test\_Id>

<Admin\_Id>3</Admin\_Id>

<Name\_Test>Существительные во множественном числе</Name\_Test>

<Theme\_Id>3</Theme\_Id>

</Test>

<Test>

<Test\_Id>36</Test\_Id>

<Admin\_Id>3</Admin\_Id>

<Name\_Test>Past Simple</Name\_Test>

<Theme\_Id>3</Theme\_Id>

</Test>

<Test>

<Test\_Id>37</Test\_Id>

<Admin\_Id>3</Admin\_Id>

<Name\_Test>Past Continues</Name\_Test>

<Theme\_Id>3</Theme\_Id>

</Test>

<Test>

<Test\_Id>38</Test\_Id>

<Admin\_Id>3</Admin\_Id>

<Name\_Test>Past Perfect</Name\_Test>

<Theme\_Id>3</Theme\_Id>

</Test>

</Tests>