СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc34383751)

[1.Техническое задание 5](#_Toc34383752)

[2.План-график проектных работ 12](#_Toc34383770)

[3.Обоснование выбора инструментальной среды разработки программного средства 13](#_Toc34383771)

[4.Структурная схема приложения 15](#_Toc34383775)

[5.Макет экранных форм 20](#_Toc34383776)

[6.Руководство программиста 24](#_Toc34383777)

[ЗАКЛЮЧЕНИЕ 26](#_Toc34383782)

[СПИСОК ЛИТЕРАТУРЫ 27](#_Toc34383783)

[ПРИЛОЖЕНИЕ А 28](#_Toc34383784)

[ПРИЛОЖЕНИЕ Б 38](#_Toc34383794)

## ВВЕДЕНИЕ

Практика – это один из важнейших элементов профессиональной деятельности. Без неё невозможно закрепить знания и навыки в любой сфере деятельности. Практика позволяет научиться новым профессиональным навыкам и умениям, закрепить уже имеющиеся и проверить их, развить технические и аналитические способности. Учебная практика направлена на усвоение и перенос новых знаний из теории в практику. Практика должна помочь студентам научиться управлять своим временем и распределять задачи по приоритету.

Учебная практика является неотъемлемой частью учебного процесса и направлена на развитие и дальнейшее формирование у студентов практических профессиональных умений и приобретение первоначального практического опыта по профессии ПМ 01 «Эксплуатация и модификация информационных систем». Практика проходит в вычислительном центре ХПЭТ с 17 февраля 2020 г. до 7 марта 2020 г. и её объём составляет 108 часов.

Основной задачей практики является создание программного продукта и поэтому на ней создавалась система для тестирования «ТеСи», которая является ИС, состоящей из программы и БД.

Тестирование – это очень старый метод проверки знаний, но в последнее время вместо бумажных тестов всё чаще встречаются электронные варианты. Тестирующие программы и системы всё чаще встречаются по всему миру. Им нашлось применение в школах, техникумах, вузах, предприятиях – везде, где требуется быстрая и надёжная проверка знаний. Поэтому популярность электронных тестов только растёт.

ТеСи позволяет проводить не какой-то один определённый тест, но любой возможный. Данная система представляет платформу для создания, хранения и прохождения тестов по любым темам любой сложности и размера. Это гораздо эффективнее и удобнее создания множества разрозненных, несвязанных между собой тестов.

Цель данной системы: проверка знаний пользователя по любой выбранной им теме из доступных для прохождения и при желании сравнения с результатами других пользователей. Доступно прохождения как одного, так и нескольких тестов для получения более полной картины по выбранной теме.

Практикант во время прохождения практики закрепляет профессиональные компетенции, приобретаемые в результате освоения теории, и помогает приобрести практический опыт выполнения профессиональных заданий.

## 1.Техническое задание

## **1.1.Термины и принятые обозначения**

1. БД – база данных
2. СУБД – система управления базами данных
3. ОС – операционная система
4. ИС – информационная система
5. Клиент – программа-тестер
6. CE - Community Edition
7. VS – Visual Studio
8. IDE – интегрированная среда разработки
9. MVS – Microsoft Visual Studio Community Edition
10. ЯП – язык программирования
11. ИСР – инструментальное средство разработки
12. Git – система контроля версий
13. SQL – структурированный язык запросов

## **1.2.Общие сведения**

Полное наименование ИС: ИС «ТеСи».

Условное обозначение: ТеСи.

## **1.3.Назначения и цели создания системы**

Вид автоматизируемой деятельности – добавление, хранение и проведение тестов по различным темам, их оценка, сохранение и сравнение полученных результатов. Следовательно, назначение ИС заключается в осуществлении следующих функций:

1. Добавление тестов в систему и их редактирование
2. Хранение тестов в БД с возможностью создания их резервных копий, полных или частичных
3. Проведение тестов с перемешиванием вопросов
4. Оценка результатов тестов
5. Сохранение и просмотр результатов в БД

## **1.4.Общие требования к системе**

## **1.4.1.Требования к подсистемам**

ИС состоит из нескольких взаимосвязанных подсистем, реализующих различные функции системы:

1. Хранилище данных ( БД ) предназначено для хранения всей необходимой информации, получения и сохранения данных. БД должна быть защищена от несанкционированного проникновения и доступна большую часть времени.
2. Административная подсистема, которая обеспечивает добавление и изменение тестов с помощью встроенного в программу редактора.
3. Подсистема логов клиента, которая обеспечивает создание и сохранения логов всех действий, производимых программой и пользователем.
4. Подсистема тестирования клиента, которая собственно и проводит тестирование пользователя, проверку и сохранение результатов.
5. Подсистема авторизации и регистрации, которая обеспечивает разграничение прав доступа и является отправной точкой для административной и тестирующей подсистем.

## **1.4.2.Требования к аппаратному обеспечению**

Таблица 1 – системные требования

|  |  |  |
| --- | --- | --- |
| Компонент | минимальные | Рекомендуемые |
| Процессор | Intel или AMD с тактовой частотой не менее 1.5 ГГц | Intel или AMD с тактовой частотой 3 ГГц или выше |
| ОЗУ | 2 ГБ или больше | 16 ГБ или больше |
| Диск | Не менее 10МБ свободного места для установки и работы приложения  Не менее 1ГБ для СУБД и БД | Не менее 2ГБ свободного места или больше |
| Экран | Монитор с разрешением 1024 \* 768 или выше | Монитор с разрешением 1920 \* 1080 или выше |

## **1.4.3.Требования к программному обеспечению**

Необходимое ПО для серверной части:

1. СУБД MySQL

Необходимое ПО для клиентской части:

1. Операционная система Windows 7 или старше
2. Microsoft .NET Framework 4.8 или выше

## **1.4.4.Требования к техническому обеспечению**

Особых требований к техническому обеспечению не предъявляется. Поддержка спецоборудования отсутствует.

## **1.4.5.Требования к персоналу**

Для эксплуатации системы определены следующие роли:

1. Администратор – имеет доступ ко всем функциям редактора, а также к БД напрямую и исходному коду программы. Имеет право удаления тестов и вопросов, редактирования БД и программы.
2. Редактор – имеет доступ ко всем функциям пользователя, а также к добавлению и редактированию тестов, требуется авторизация для подтверждения личности.
3. Пользователь – имеет доступ к прохождениям тестов и просмотру лучших результатов других пользователей, авторизация и регистрация не обязательны, но возможны.

## **1.4.6.Требования к архитектуре системы**

     Система должна быть построена на основе клиент-серверной архитектуры с централизованной БД и поддержкой большого количества одновременно работающих удалённых пользователей. Передача данных должна происходить по сети Интернет. Программа может быть доступна любому пользователю для прохождения тестов. Обновления программы не встроены в систему и могут требовать полной переустановки.

## **1.5.Требования к подсистемам**

## **1.5.1.Хранилище данных**

     БД ( хранилище данных ) будет работать под управлением СУБД MySQL 8.0 CE. Это бесплатная многопользовательская реляционная СУБД, что даст возможность системе

работать с большим количеством пользователей одновременно. БД будет состоять только из таблиц и не будет иметь каких-либо триггеров и процедур. Основные планируемые запросы к БД – select и insert.

## **1.5.1.1.Инфологоческая схема данных**

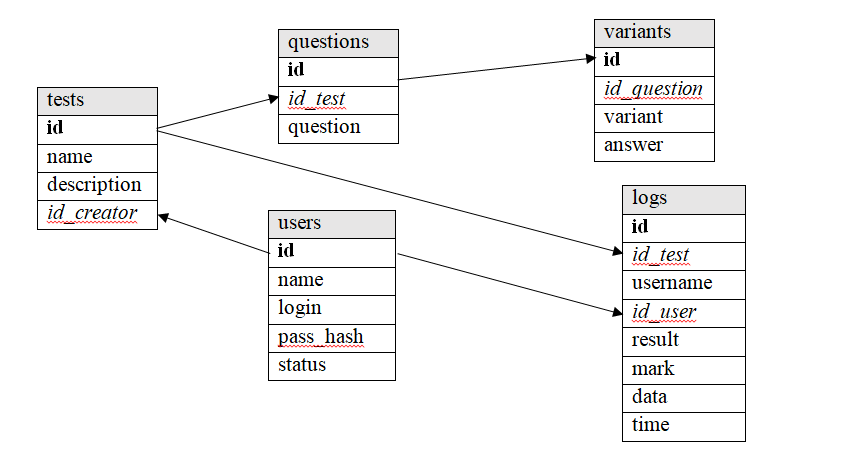


Рисунок 1 – инфологическая схема данных

Таблица 2 – tests

     Хранит в себе все доступные к прохождению тесты системы, их названия и описания. Является самой главной таблицей, от которой зависят все остальные.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***№*** | *Название поля* | *Тип поля* | *Атрибуты* | *Описание поля* |
| **1** | **id** | int | pk, auto\_increment | Уникальный номер теста |
| **2** | name | varchar(50) | not null | Название теста |
| **3** | description | tinytext | not null | Описание теста |
| **4** | id\_creator | int | fk, not null | Id создателя теста |

     Таблица 3 – questions

   Хранит в себе все вопросы всех тестов. Зависит от таблицы tests. Является главной для таблицы variants.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***№*** | *Название поля* | *Тип поля* | *Атрибуты* | *Описание поля* |
| **1** | **id** | int | pk, auto\_increment | Уникальный номер вопроса |
| **2** | *id\_test* | int | fk, not null, default “1” | Номер теста |
| **3** | question | tinytext | not\_null | Собственно вопрос |

     Таблица 4 – variants

     Хранит в себе все варианты ответов всех тестов, а также ключи правильных ответов. Зависит от таблицы questions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***№*** | *Название поля* | *Тип поля* | *Атрибуты* | *Описание поля* |
| **1** | **id** | int | pk, auto\_increment | Уникальный номер варианта ответа |
| **2** | *id\_question* | int | fk, default “1” | Номер вопроса |
| **3** | variant | varchar(60) | not null | Собственно вариант ответа |
| **4** | answer | char(1) | not null, default “f” | Правильность варианта ответа  t – значит «верно»  f – значит «неверно» |

Таблица 5 – logs

     Хранит в себе лог прохождения тестов пользователями, а также их результаты. Зависит от таблицы tests.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***№*** | *Название поля* | *Тип поля* | *Атрибуты* | *Описание поля* |
| **1** | **id** | int | pk, auto\_increment | Уникальный номер записи лога |
| **2** | *id\_test* | int | fk, not null, default “1” | Номер теста |
| **3** | username | varchar(50) | null | Имя пользователя, проходящего тест  ( если не указан id\_user ) |
| **4** | *id\_user* | int | fk, null | Id пользователя, проходящего тест  ( если не указан username ) |
| **5** | result | int | not null, default “0” | Результат теста, количество правильных ответов |
| **6** | mark | int | not null, default “0” | Оценка на тест |
| **7** | data | datatime | current\_timestamp | Время добавления в БД |
| **8** | time | time | not null | Время, затраченное на тест |

     Таблица 6 – users

     Хранит в себе данные о пользователях и их роли. В зависимости от роли пользователю будут выделяться разные права – простого пользователя или администратора.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***№*** | *Название поля* | *Тип поля* | *Атрибуты* | *Описание поля* |
| **1** | **id** | int | pk, auto\_increment | Уникальный номер пользователя |
| **2** | name | varchar(60) | not null | Имя пользователя |
| **3** | login | varchar(50) | not null | Логин пользователя |
| **4** | pass\_hash | char(60) | not null | Хэш пароля |
| **5** | status | char(1) | not null, default “u” | Статус пользователя  u – значит «пользователь»  a – значит «администратор» |

## **1.5.2.Административная подсистема**

     Данная подсистема позволяет администратору системы создавать новые тесты и вопросы, а также редактировать старые. Важный критерий подсистемы – это разграничение прав доступа, чтобы кто угодно не мог редактировать тесты. Подсистема включает в себя форму редактирования тестов, их вопросов и ответов. Редактор должен позволять добавлять новые тесты, просматривать уже существующие, добавлять и полностью изменять вопросы и ответы.

## **1.5.3.Подсистема логов**

     Подсистема логов обеспечивает протоколирование всех действий пользователя и ошибок программы в виде файлов специального формата. При работе, в главной директории программы должна создаваться папка «logs», в ней – папки по дням, в формате ГГГГ-ММ-ДД. В этих папках содержаться файлы логов 4х разных уровней:

1. debug.log – файл самого нижнего уровня. В него подробно пишутся все действия программы и пользователя.
2. info.log – файл основного уровня. В него пишутся только самые главные действия, такие как успешное прохождение теста, авторизация и регистрация, изменение теста.
3. warning.log – файл уровня предупреждений. В него пишутся предупреждения о возможных ошибках программы.
4. error.log – файл уровня ошибок. В него пишутся ошибки программы и их возможные причины. Все вероятные ошибки и исключения должны быть обработаны и записаны, а программа должна продолжать работу даже при их возникновении. По необходимости, пользователь может быть уведомлён о возникших проблемах.

Собственно формат протоколирования должен быть следующим:

время | уровень | отправитель | сообщение

Например:

2020-02-25 11:19:24.6399 | DEBUG | Test.Program | Инициализация глобальных переменных

     Сообщения могут писаться как на русском, так и английском языке, а также иметь структуру вложенности и разделения на блоки. Допустимы выделения отдельных сообщений для их подчёркивания и / или разделения лога на блоки. Протоколироваться должна большая часть действий, производимых программой и пользователем. Также в логии могут писаться данные, если их открытая запись не противоречит общей безопасности системы.

## **1.5.4.Подсистема тестирования**

     Данная подсистема проводит собственно тестирование пользователя. Она включает в себя выбор теста, вывод вопросов и ответов, их проверку, вывод и сохранение результата в базу данных. Она включает в себя несколько форм пользовательского интерфейса:

1. форма с выбором тестов
2. форма с собственно тестом
3. форма с итоговыми результатами

## **1.5.5.Подсистема авторизации и регистрации**

     Данная подсистема обеспечивает авторизацию и регистрацию пользователей, а также разграничение прав доступа. Она включает в себя две формы:

1. форма авторизации
2. форма регистрации

Авторизация и регистрация должна происходить через базу данных, таблицу users. Должна быть возможность локального «запоминания» логина и пароля в зашифрованном виде.

## 2.План-график проектных работ

Таблица 7 – распределение бюджета времени по этапам практики

|  |  |  |
| --- | --- | --- |
| Содержание учебного материала | Дата | Часов |
| Инструкция по ТБ и ОТ. Изучение должностных инструкций программиста. | 17.02.2020 | 6 |
| Постановка целей и задач проектных работ | 18.02.2020 | 6 |
| Составление плана-графика проектных работ | 19.02.2020 | 6 |
| Изучение и разработка технического задания | 20.02.2020 | 6 |
| Изучение и разработка технического задания | 21.02.2020 | 6 |
| Составление структурной схемы приложения | 22.02.2020 | 6 |
| Использование графического режима для создания интерфейса пользователя. Оформление дизайна проекта. Использование текстового режима для создания интерфейса пользователя. | 25.02.2020 | 8 |
| Разработка макета стартовой формы приложения. | 26.02.2020 | 8 |
| Добавление анимационных эффектов. | 27.02.2020 | 8 |
| Создание правил «дружественного» диалога интуитивно-понятного интерфейса. Макет формы. | 28.02.2020 | 6 |
| Разработка формы вопроса в выбранной среде программирования. | 29.02.2020 | 6 |
| Создание стандартных компонентов для ввода-вывода данных. | 2.03.2020 | 6 |
| Ввод данных из файла. | 3.03.2020 | 6 |
| Разработка управляющих структур. | 4.03.2020 | 6 |
| Разработка формы для вывода результатов тестирования. | 5.03.2020 | 6 |
| **Разработка руководства пользователя и программиста.** | 6.03.2020 | 6 |
| **Экспериментальное тестирование приложения.** | 7.03.2020 | 6 |
| ИТОГО: | 17 дней | 108 |

## 3.Обоснование выбора инструментальной среды разработки программного средства

     В качестве ИСР ПО были выбраны MVS и СУБД MySQL. Используемый ЯП – C#, версия – 8.0, платформа - .NF 4.8. Используемая технология – Windows Forms.

## **3.1.C#**

C# – это современный объектно-ориентированный ЯП со статической типизацией данных. Относится к семейству языков C и обладает С – подобным синтаксисом. Поддерживает полиморфизм, перегрузку операторов, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, исключения и комментарии в формате XML. Вдохновлялся такими языками как С++, Delphi и Java. Данный язык является довольно гибким и многие его особенности зависят от платформы, на которой он используется. Таким образом, C# используется и в играх, и в программах, и в сайтах. Также С# предлагает следующие возможности:

1. сборка мусора – периодическая очистка памяти от ненужных и неиспользуемых элементов, которая позволяет повысить производительность и надёжность программы
2. обработка исключений – позволяет обрабатывать возникающие ошибки и предпринимать какие-либо действия, что позволяет избежать крашей и вылетов программы, а также уменьшить общее количество ошибок
3. единая и типобезопасная система типов, которая позволяет работать со всеми переменными схожим образом и избежать многих ошибок

Однако главная особенность С# - подстраивание под используемую платформу. В данном случае используемая платформа – .NF 4.8 с технологией Windows Forms. Она позволяют быстро и легко создавать высокопроизводительные программы для ОС Windows с собственным интерфейсом и дизайном. В частности, предоставляются следующие возможности:

1. создание оконных форм при помощи графического интерфейса
2. возможность лёгкого добавления, удаления и изменения элементов формы при помощи кода
3. возможность лёгкого и быстрого подключения внешних модулей и библиотек для их последующего использования в проекте через менеджер пакетов

## **3.2.MySQL**

MySQL - свободная реляционная система управления базами данных. CE версия продукта распространяется под GNU General Public License, что позволяет бесплатно её использовать в некоммерческих проектах с открытым исходным кодом. Данная СУБД также имеет коннекторы ( драйверы ) подключения к различным ЯП, в том числе и С#.

Основные особенности:

1. простота развёртывания и использования
2. возможность очень гибких настроек
3. легковесность и низкие системные требования
4. высокая производительность и эффективность
5. возможность изменения параметров выполнения SQL - запросов для большей гибкости

## **3.3.Дополнительно**

Помимо ЯП С# и СУБД MySQL для создания ТеСи используется дополнительное ПО:

1. MVS – мощная IDE для разработки на различных языках, в данном случае – на C#. Поддерживает много функций, таких как переименование, поддержка Git, автодополнение, автоформатирование, графические редакторы, средства отладки и многое другое
2. HeidiSQL – редактор для различных СУБД, в т.ч. и для MySQL. Позволяет графически создавать и редактировать таблицы, выполнять запросы, вводить и просматривать данные, создавать и выполнять SQL – дампы и многое другое.
3. ODBC драйвер для MySQL – является драйвером для С - подобных языков.
4. Драйвер MySQL для VS – позволяет просматривать БД напрямую из VS, а также использовать связку элементов с источником данных
5. Пакет NLog для C# - обеспечивает простой интерфейс для создания логов в C# с большим количеством возможностей, например, распределение на различные уровни и файлы.

## 4.Структурная схема приложения

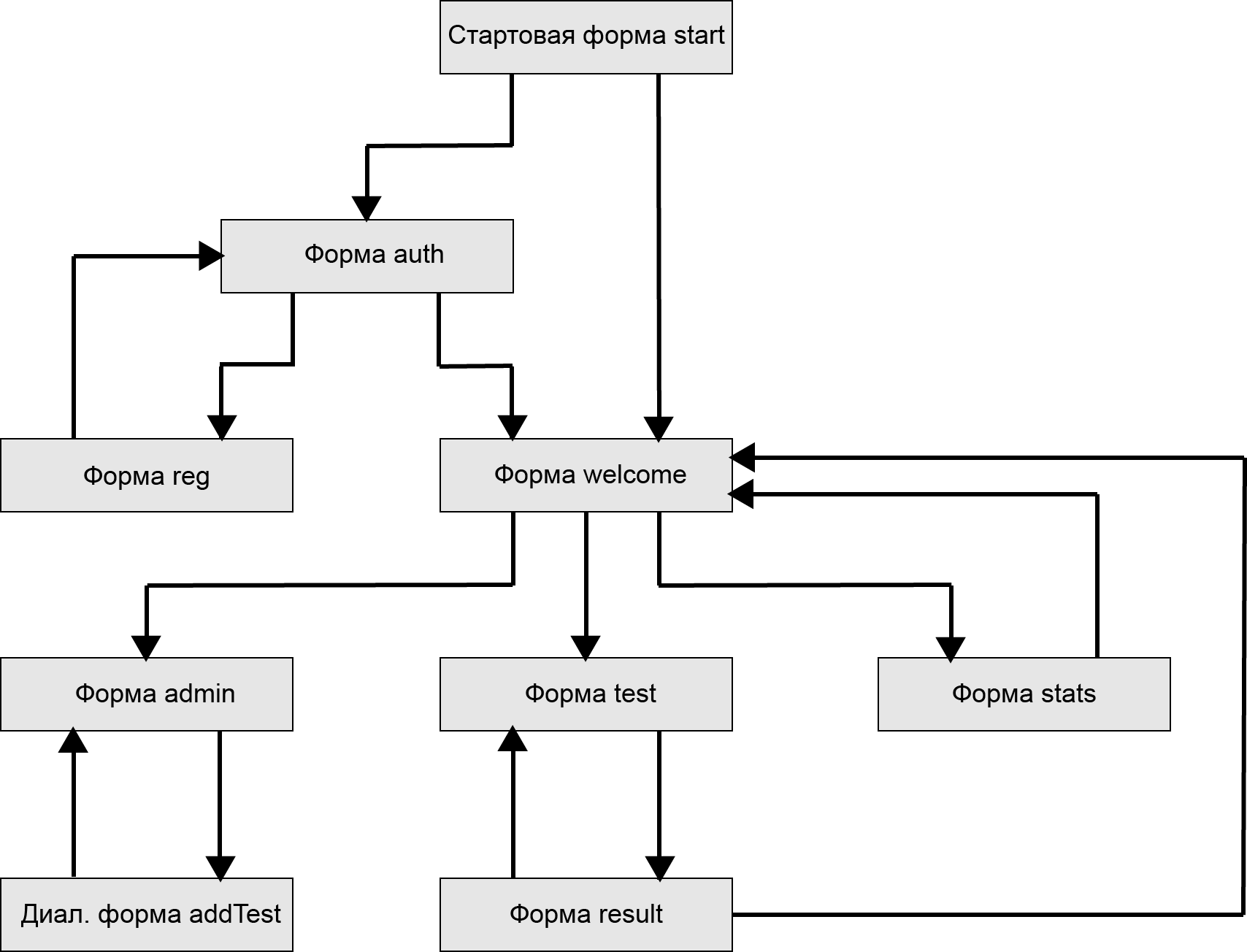


Рисунок 1 – структурная схема приложения

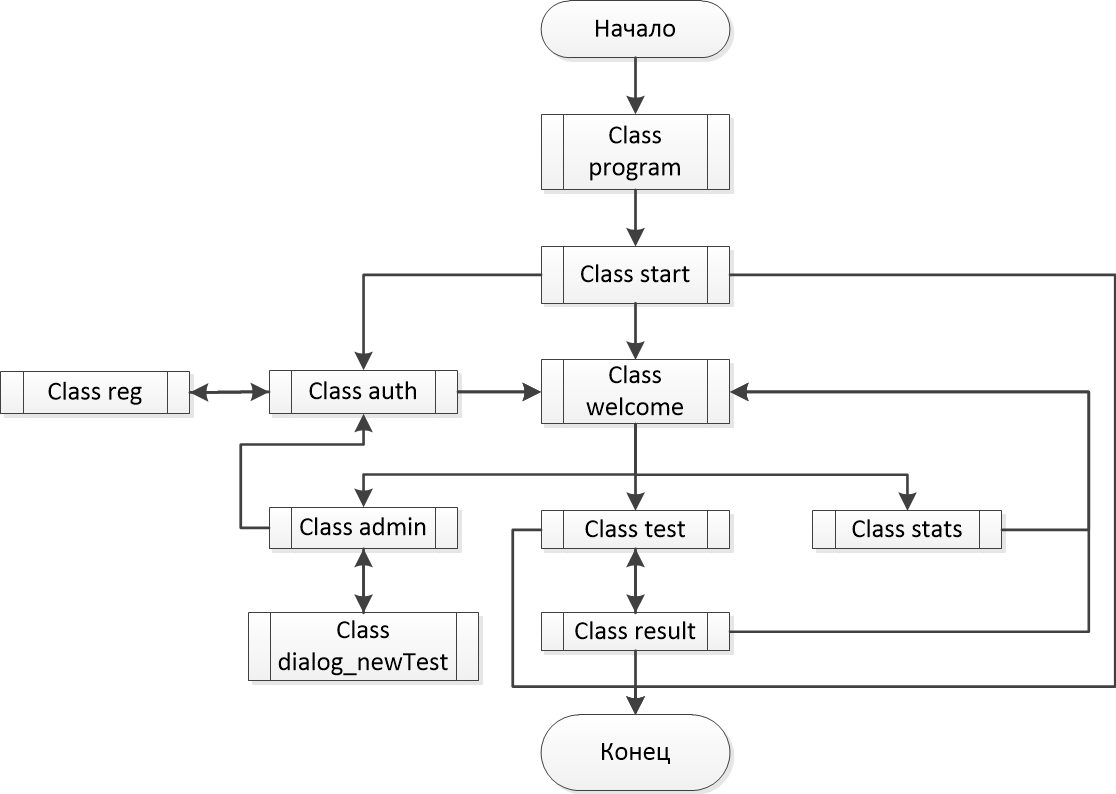


Рисунок 2 – общий вид блок схемы

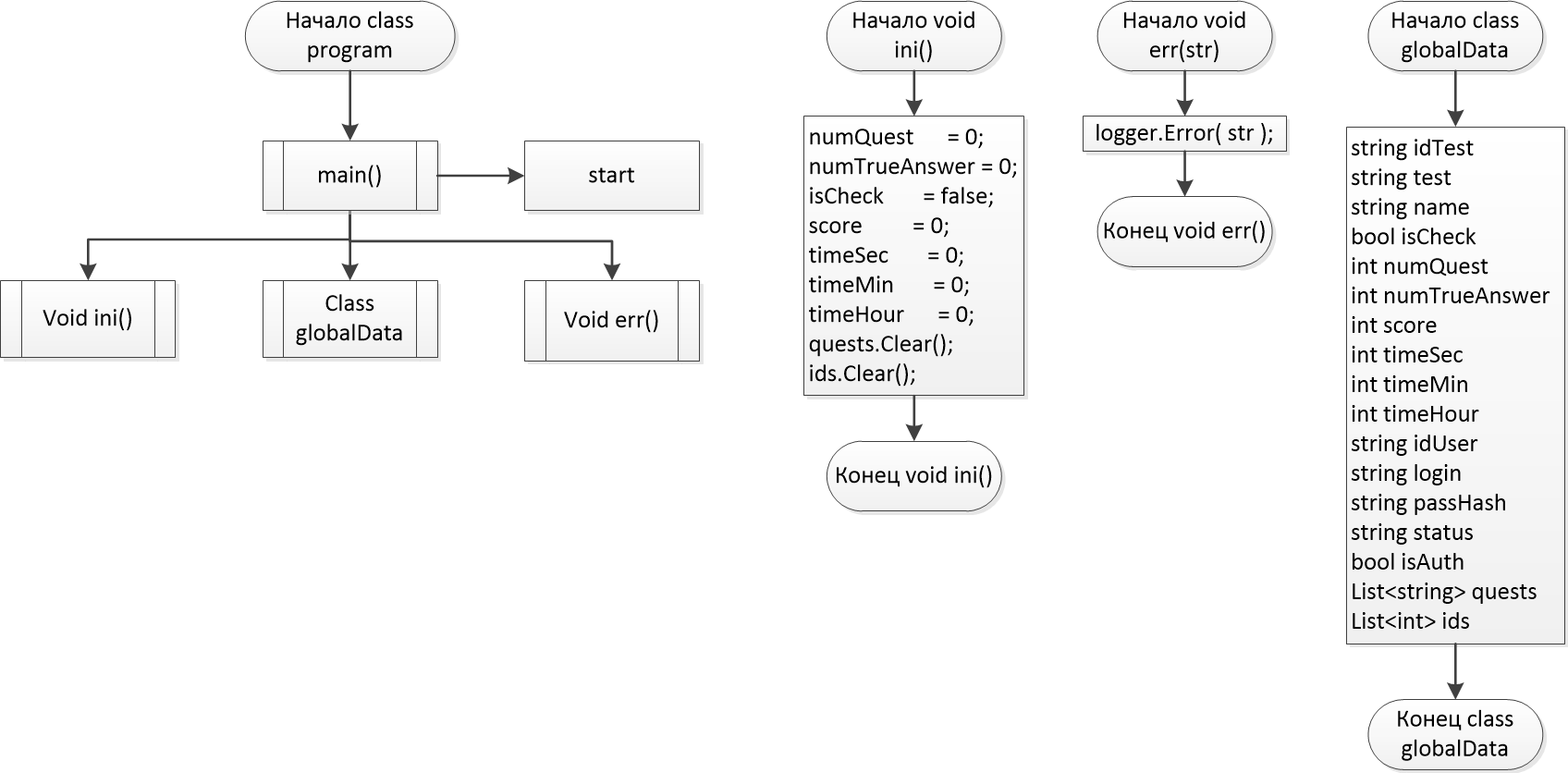


Рисунок 3 – блок схема формы program

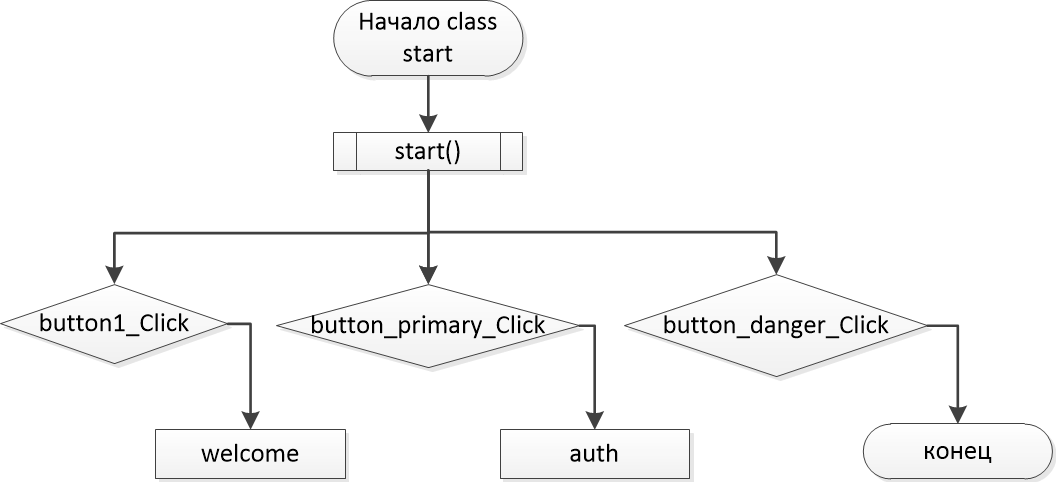


Рисунок 4 – блок схема формы start

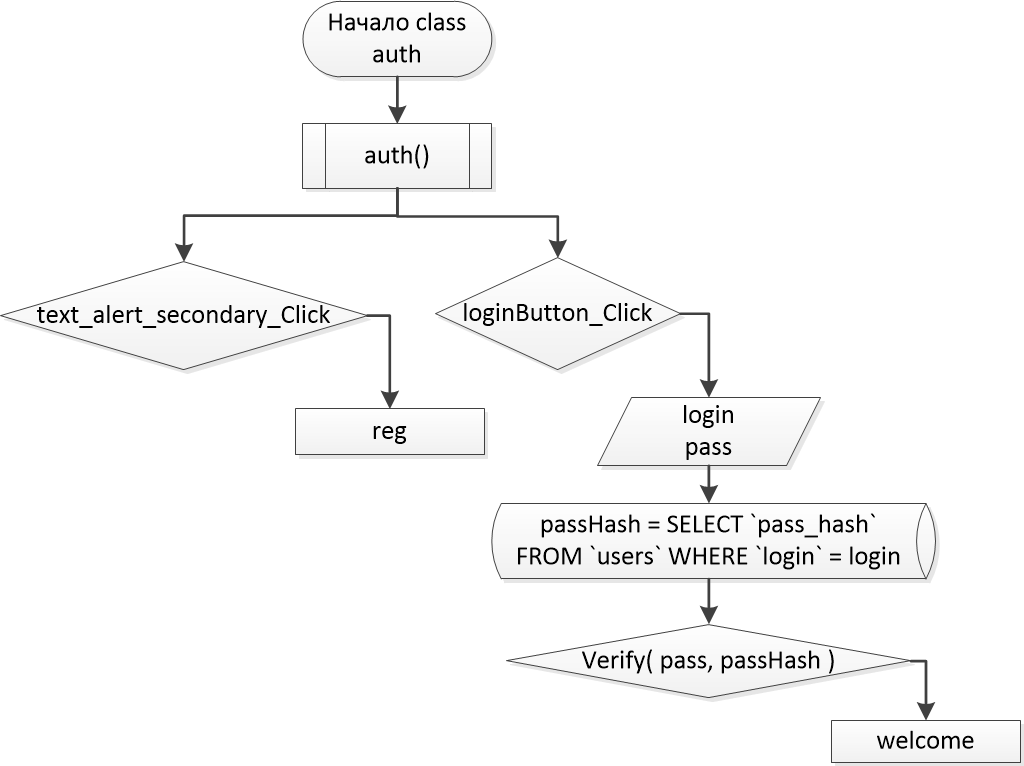


Рисунок 5 – блок схема формы auth

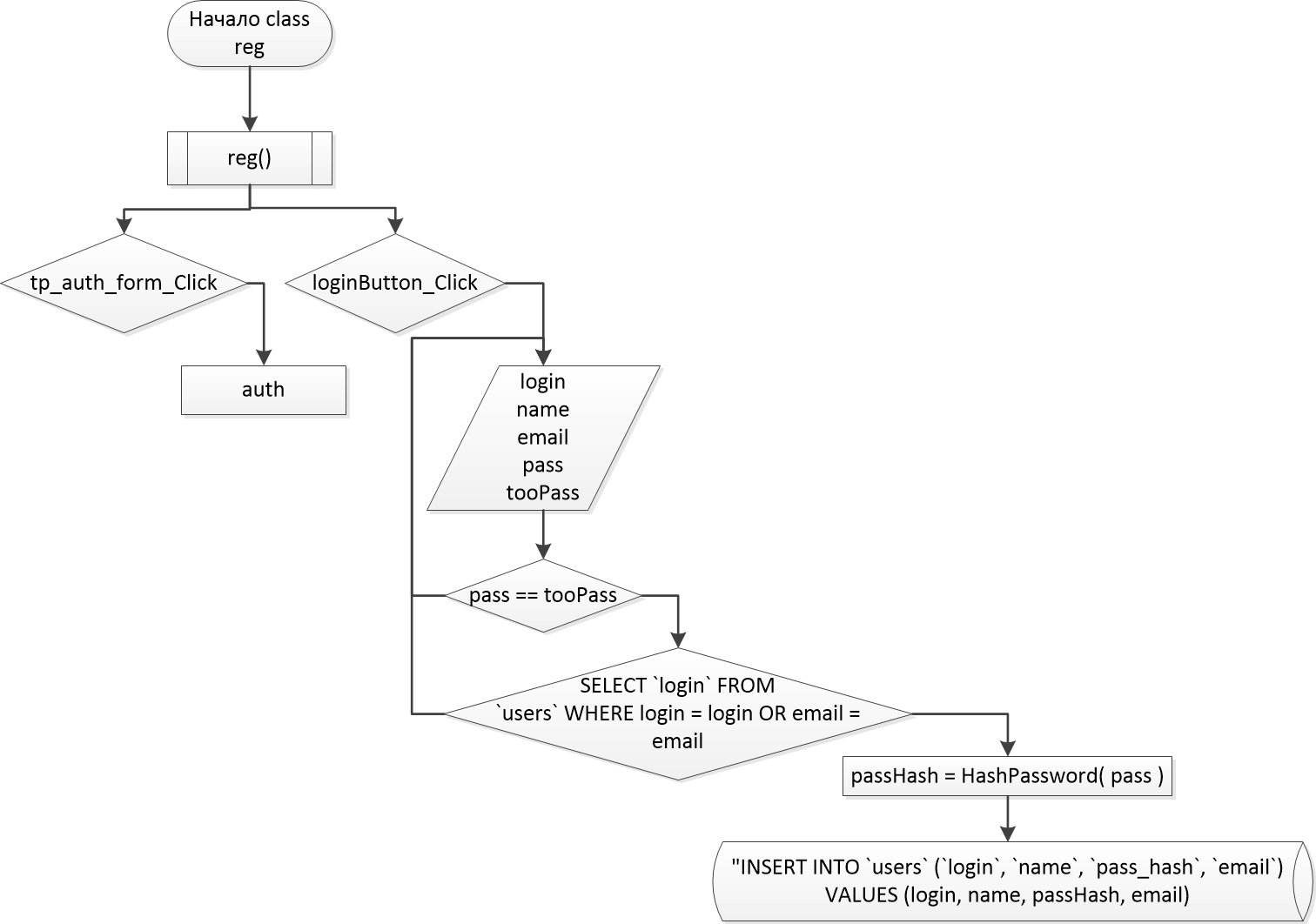


Рисунок 6 – блок схема формы reg

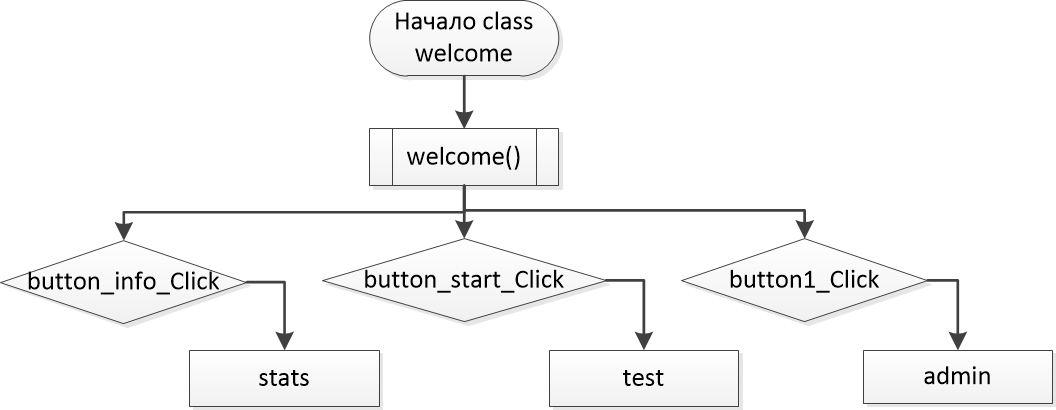
****

Рисунок 7 – блок схема формы welcome

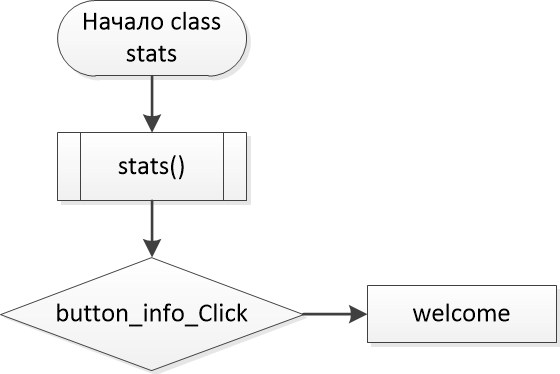


Рисунок 8 – блок схема формы stats

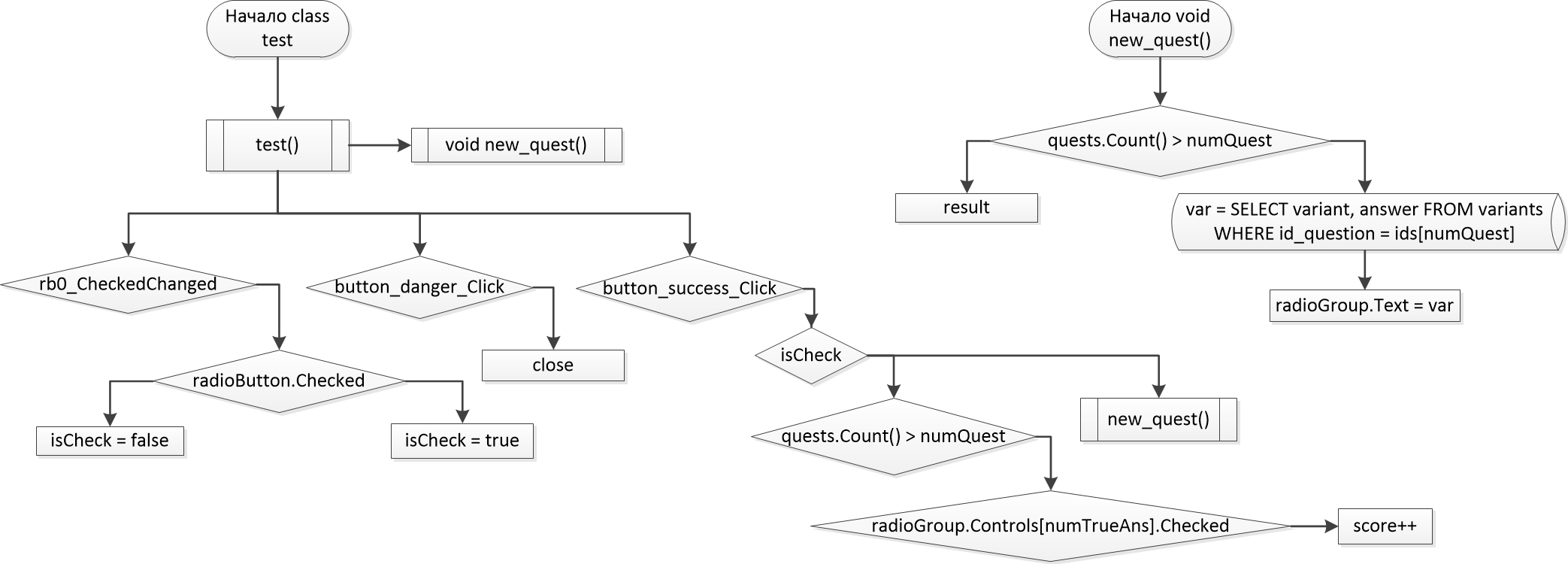


Рисунок 9 – блок схема формы test

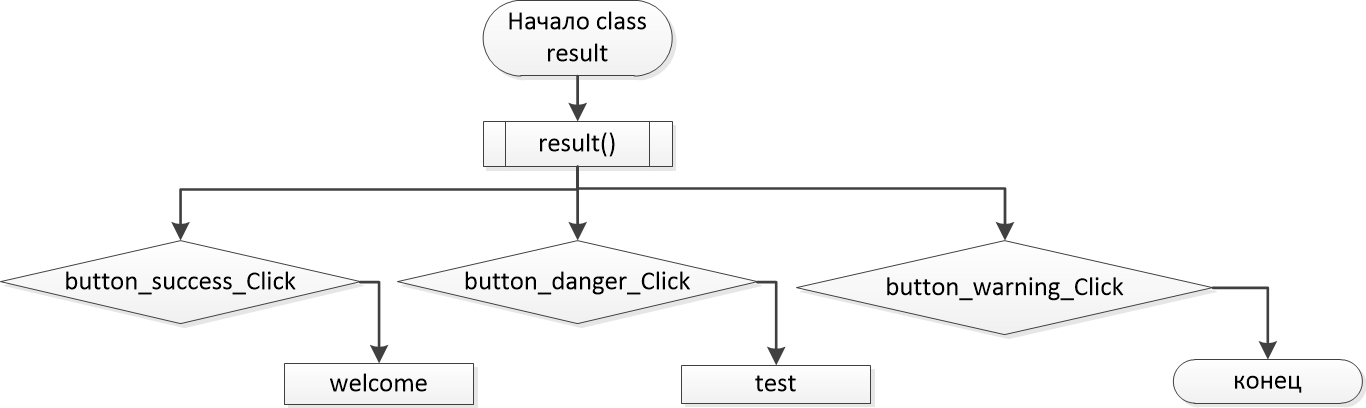


Рисунок 10 – блок схема формы result

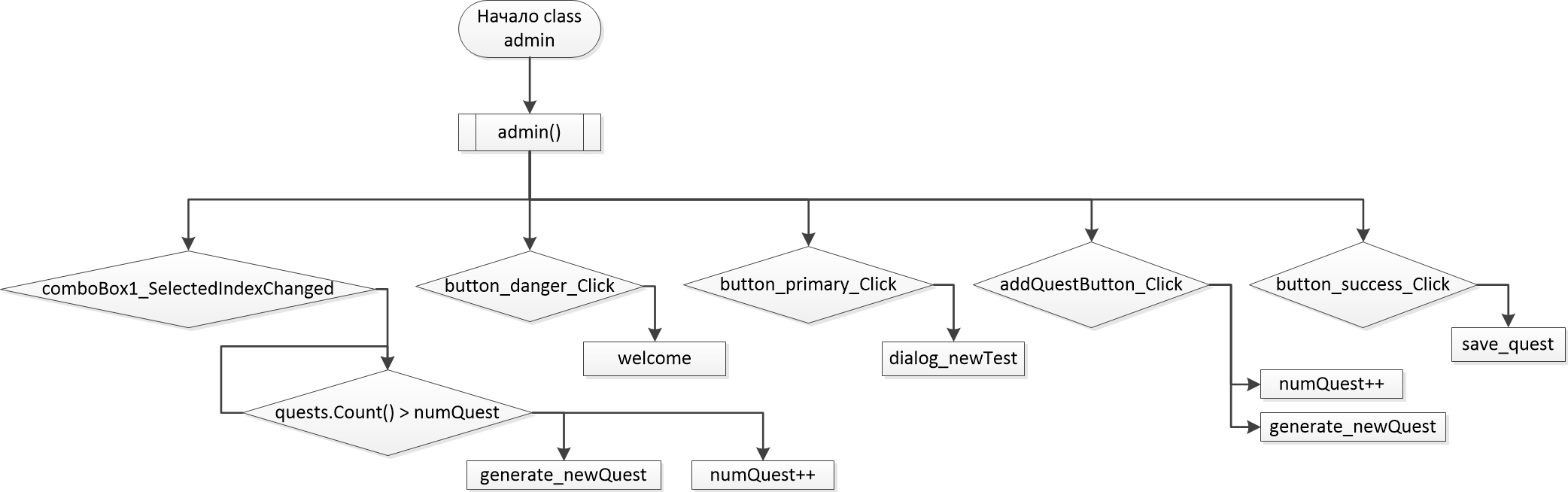


Рисунок 11 – блок схема формы admin

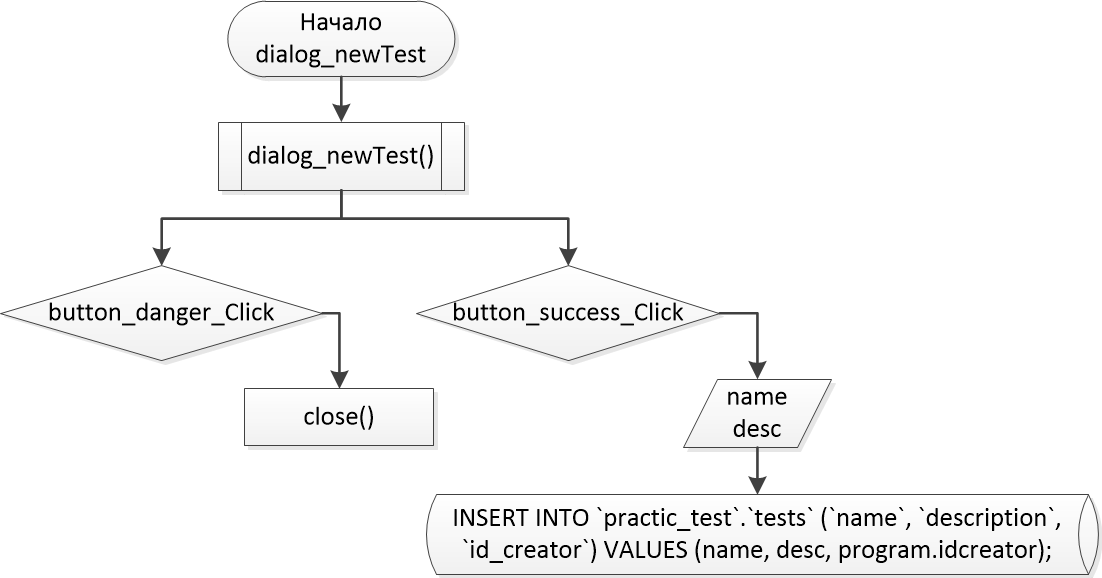


Рисунок 12 – блок схема формы dialog\_newTest

## 5.Макет экранных форм

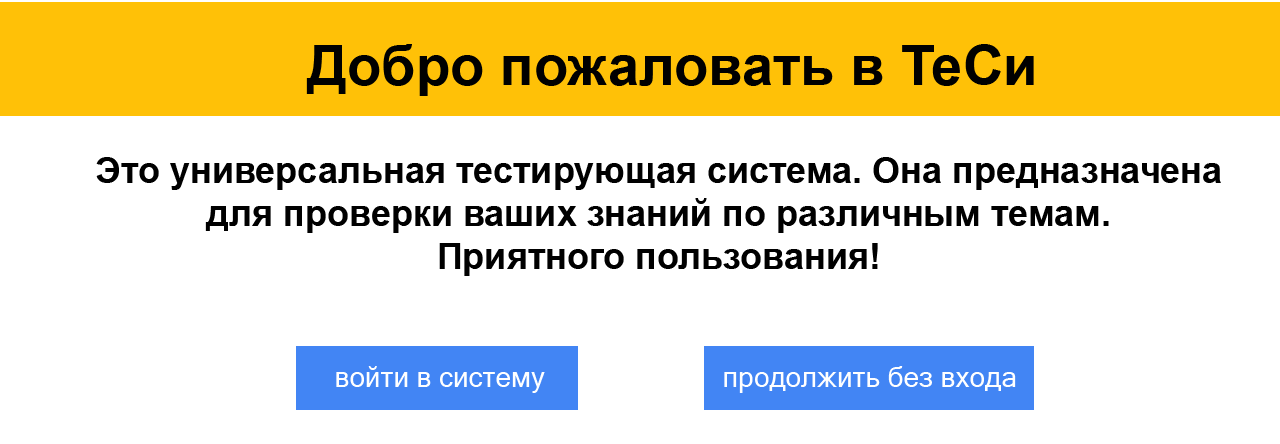


Рисунок 13 – макет формы start

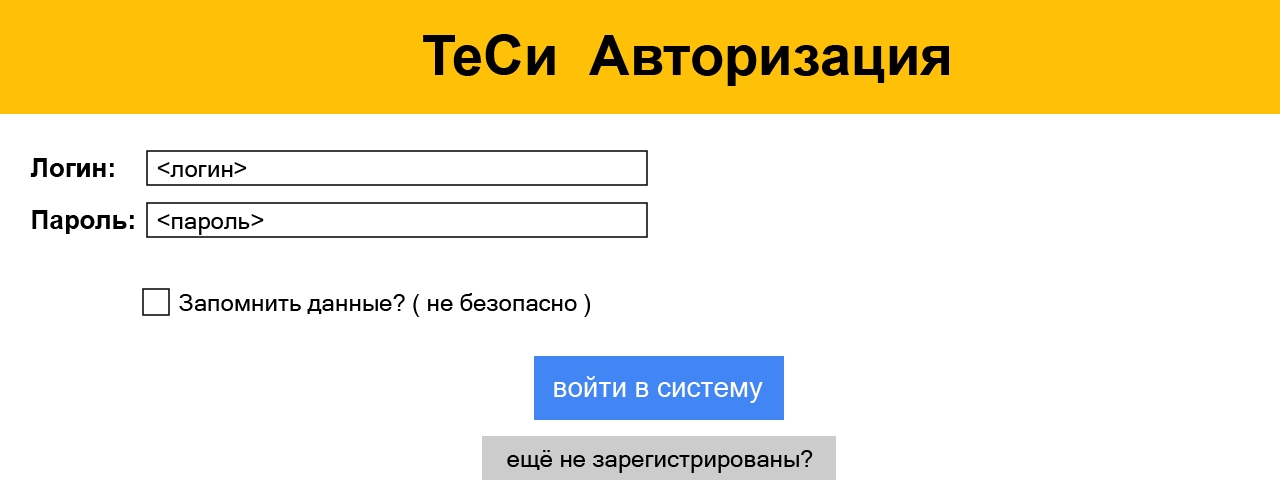


Рисунок 14 – макет формы auth

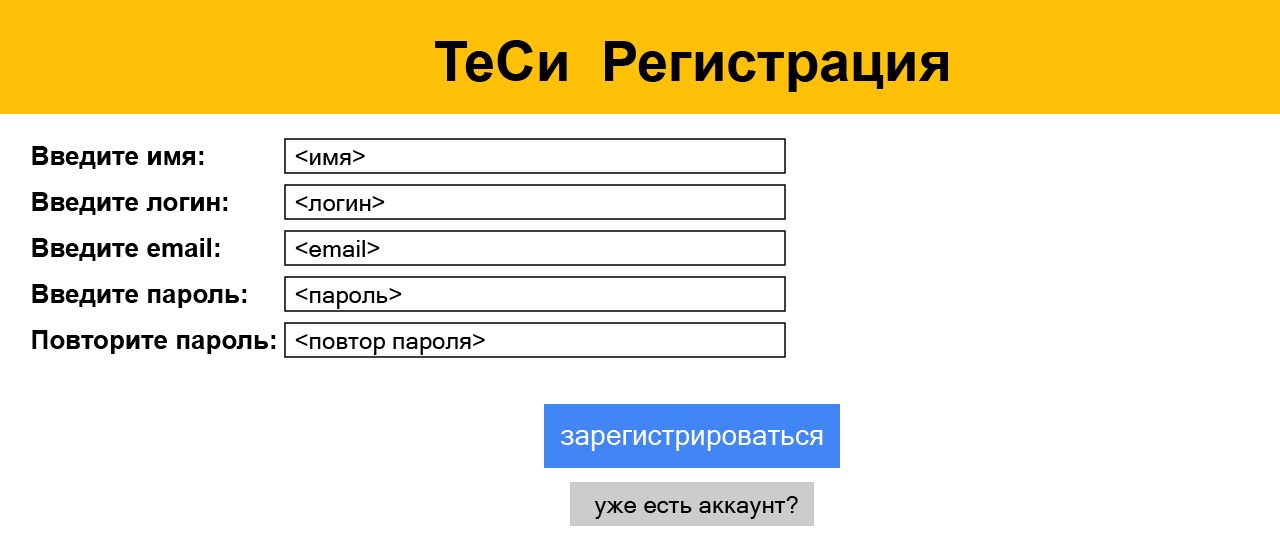


Рисунок 15 – макет формы reg

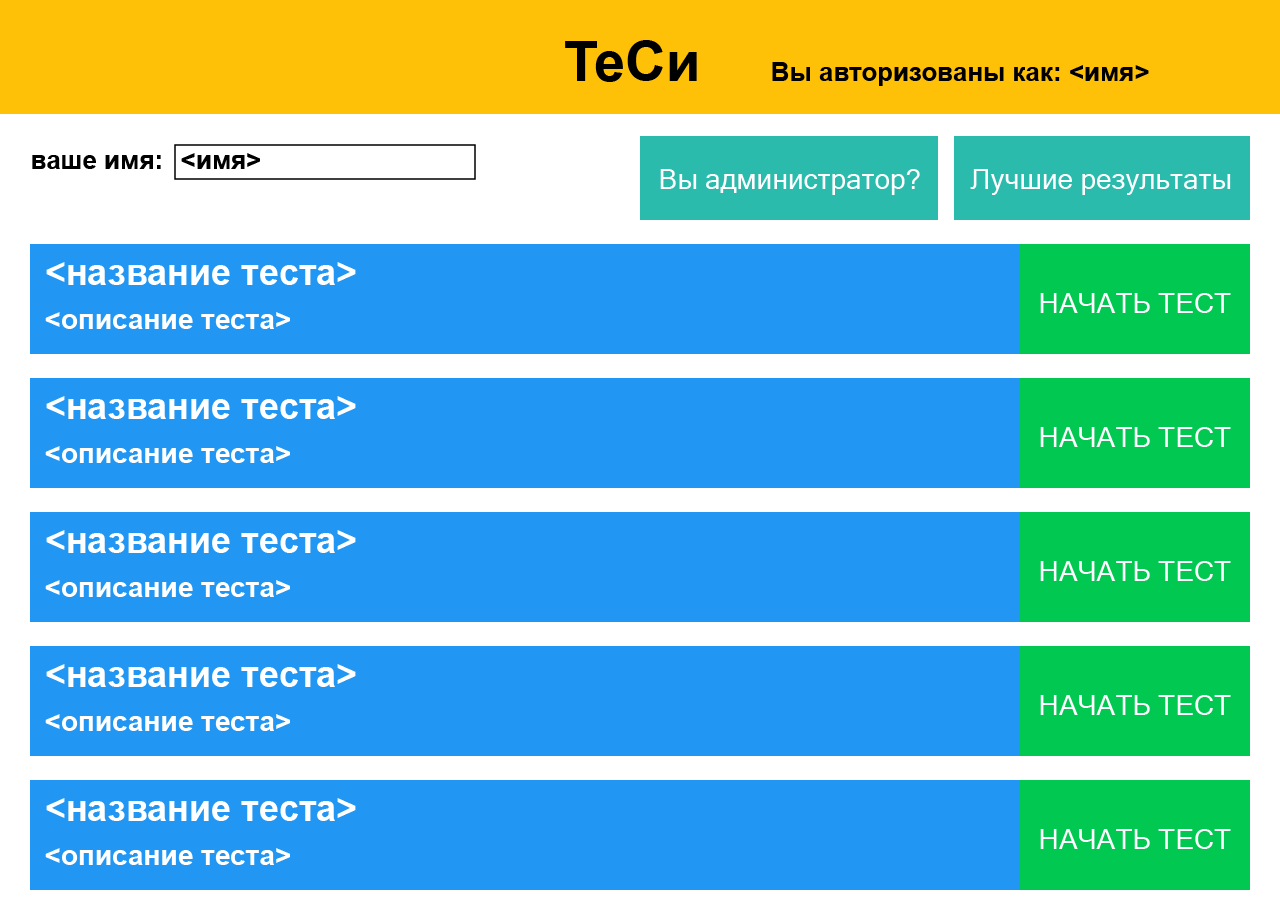


Рисунок 16 – макет формы welcome

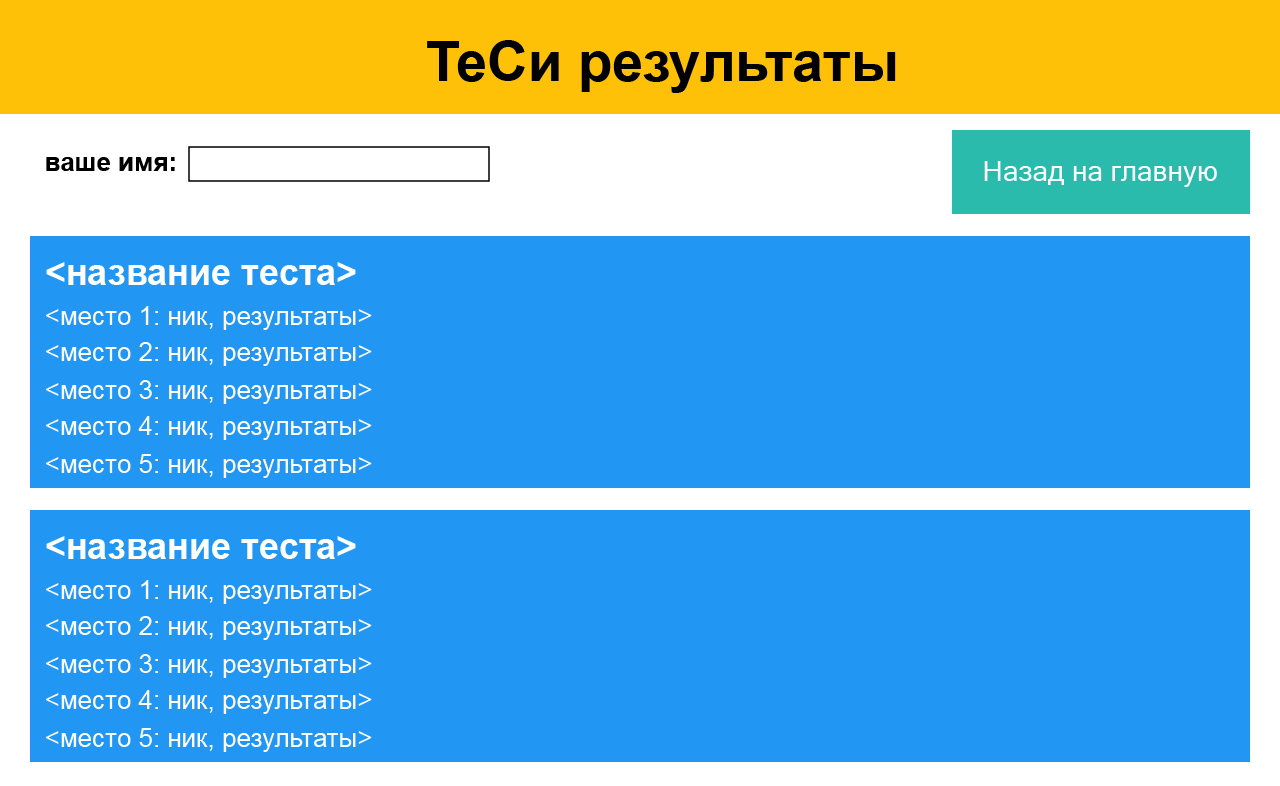


Рисунок 17 – макет формы stats

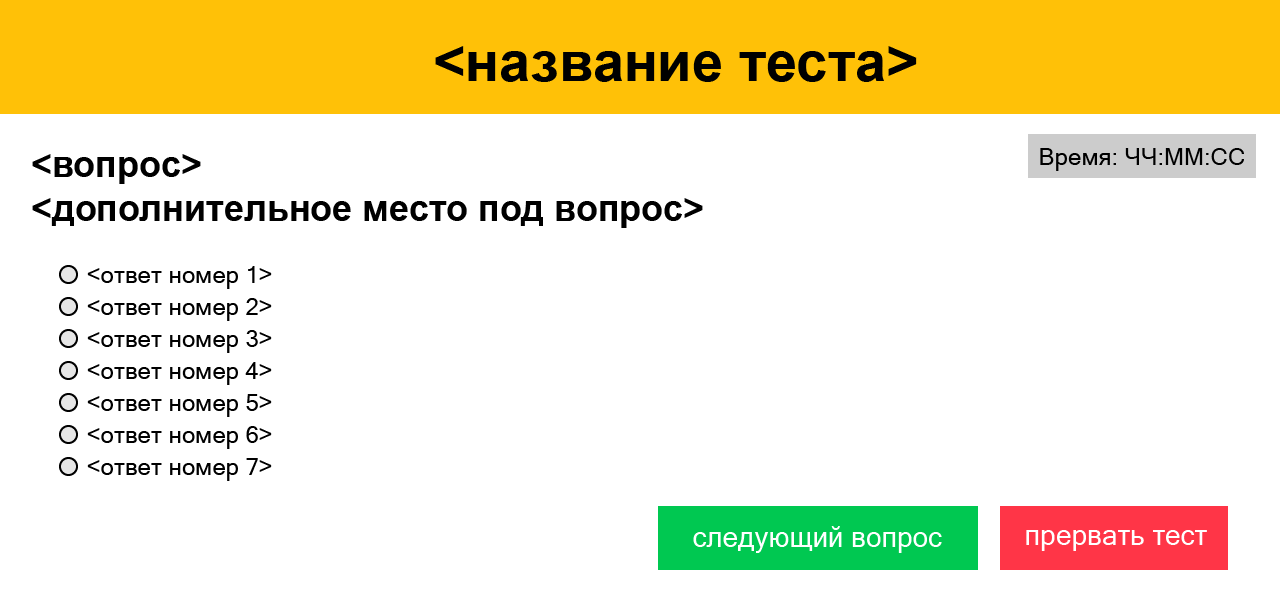


Рисунок 18 – макет формы test

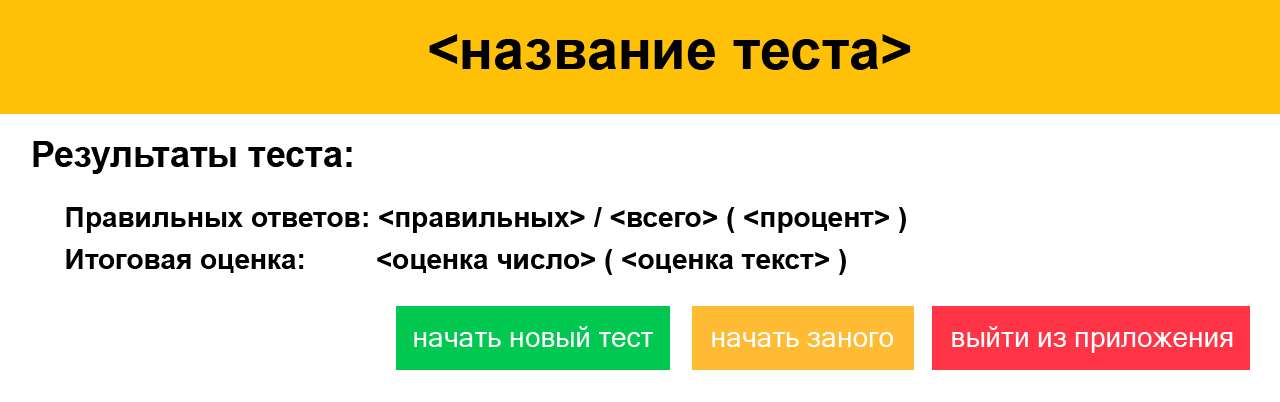


Рисунок 19 – макет формы result

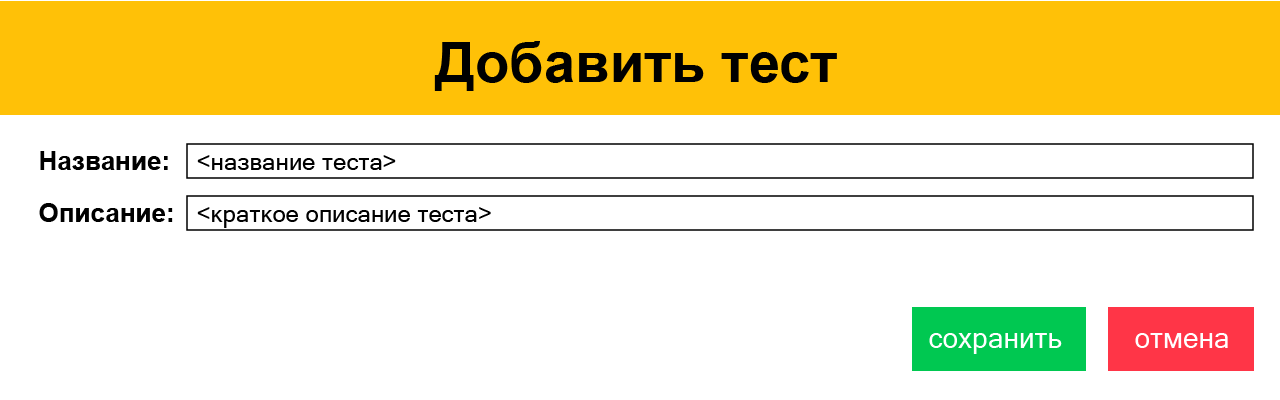


Рисунок 20 – макет формы dialog\_newTest

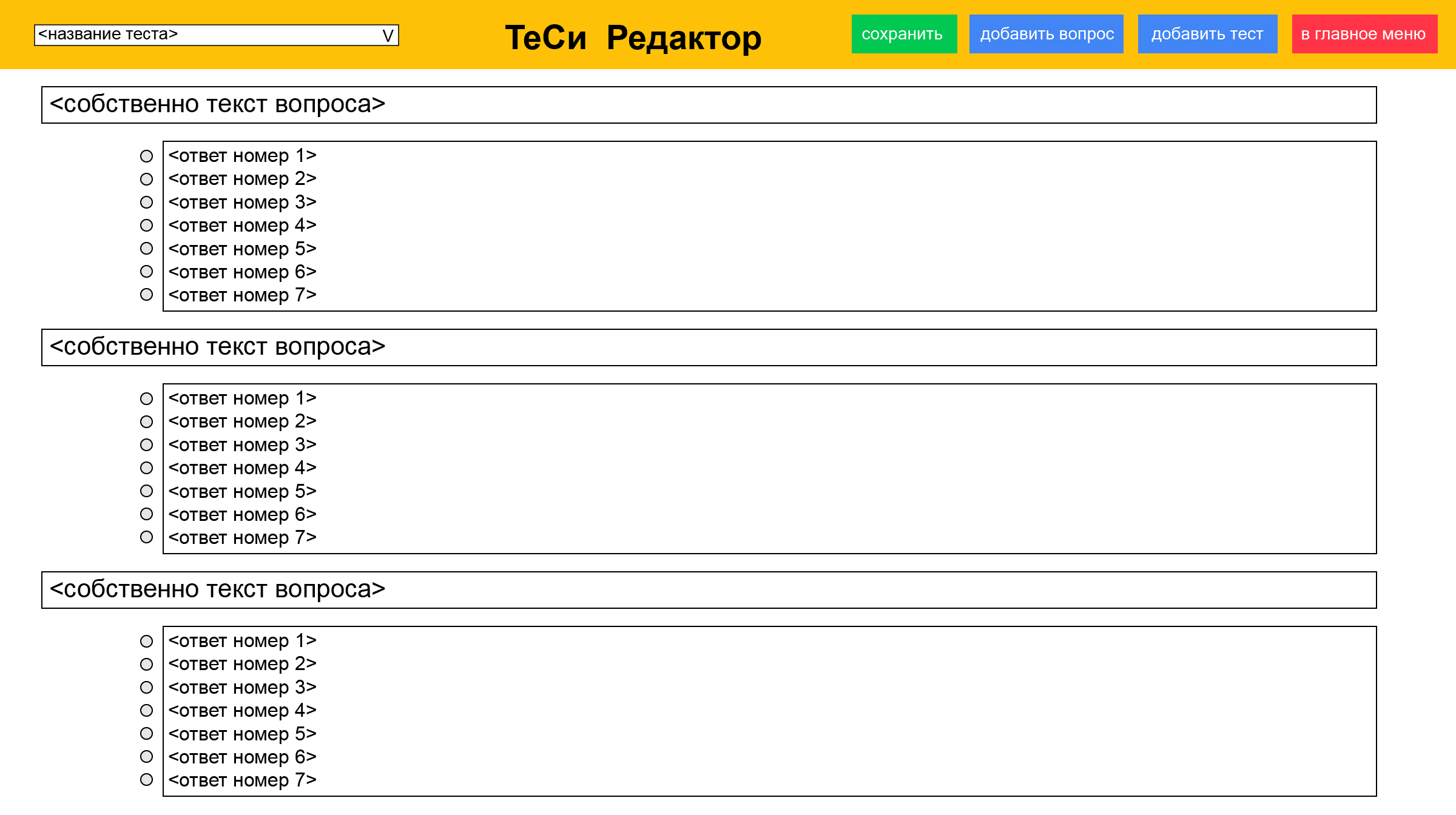


Рисунок 21 – макет формы editor

## 6.Руководство программиста

## **6.1.Общая структура**

Так как C# - это объектно-ориентированный язык, каждый его файл имеет единую строгую структуру:

// подключение пакетов и модулей

using NLog;

using System;

using System.Windows.Forms;

// пространство имён ( совпадает с названием проекта )

namespace Test {

// объявление класса ( совпадает с названием файла)

public partial class start : Form {

// объявление переменной

readonly Logger logger = LogManager.GetCurrentClassLogger();

// объявление функции

public start() {

// вызов функции

InitializeComponent();

}

}

}

Эта структура распространяется на все файлы проекта.

## **6.2.Обработка событий**

Обработка событий – заключается в создании функций, которые вызываются при срабатывании какого-либо события. Например, следующая функция выполниться при нажатии кнопки:

private void button\_primary\_Click( object sender, EventArgs e ) {

logger.Debug( "" );

logger.Debug( "Переход на форму auth" );

logger.Debug( "" );

start.ActiveForm.Hide();

auth auth = new auth();

auth.ShowDialog();

Close();

}

## **6.3.Связь с БД**

Связь с БД осуществляется при помощи подключенного ODBC – драйвера. Также для облегчения данной задачи был создан отдельный класс с функциями подключения и отключения.

Каждый запрос к БД отправляется и обрабатывается следующим образом:

1. создание SQL запроса, если необходимо использовать данные, то вместо них создаётся «заглушка», например, id = @id
2. добавление параметров в запрос ( по необходимости ), т.е. замена «заглушек» на данные, необходимые в данном запросе
3. открытие соединения, если соединение не может быть открыто – вызывается исключение
4. отправка запроса к БД
5. получение ответа от БД, если запрос не может быть выполнен – вызывается исключение
6. обработка ответа и полученных данных
7. закрытие соединения – необходимо для уменьшения лишней нагрузки на сервер БД
8. дополнительная обработка полученных данных ( по необходимости )

В коде это выглядит следующем образом:

// подключение драйвера MySQL

using MySql.Data.MySqlClient;

// подключение класса для работы с MySQL

readonly dbMySQL db = new dbMySQL();

// создание запроса

MySqlCommand sql = new MySqlCommand("SELECT \* FROM tbl WHERE id = @id", db.getConn());

// добавление параметров запроса

sql.Parameters.Add( "@id", MySqlDbType.Int32 ).Value = id;

// открытие соединения

db.openConnection();

// выполнение запроса и получение данных

MySqlDataReader readerSql = sql.ExecuteReader()

// обработка данных

while ( readerSql.Read() ) {

logger.Debug( readerSql[0].ToString() );

}

// закрытие соединения

db.closeConnection();

## **6.4.Обработка исключений**

Исключение – это событие, которое вызывается, когда в коде возникает ошибка. Их можно обрабатывать для исправления последствий этих ошибок либо их логгирования.

В коде обработка исключений обеспечивается следующим блоком кода:

try {

// код, который может вызвать ошибку

}

catch {

// код, вызываемый при ошибке

}

finally {

// код, вызываемый после 2х предыдущих блоков в любом случае, не обязателен

}

## ЗАКЛЮЧЕНИЕ

Практика проходила в вычислительном центре ХПЭТ с 17 февраля 2020 г. до 7 марта 2020 г. и её объём составил 108 часов. Центр был оснащён всеми необходимыми средствами вычислительной техники и учебного оборудования, необходимых для прохождения практики.

По прохождению учебной практики были успешно выполнены все цели и задачи по закреплению теоретических и формированию практических знаний, а также по формированию профессионально необходимых качеств по профессии «Эксплуатация и модификация информационных систем». Основной целью было создание тестирующий системы «ТеСи», всех её модулей и компонентов, для дальнейшего использования во всех сферах деятельности где может требоваться проверка знаний. В итоге тестирующая система была создана в полном объёме со всеми необходимыми для её работы модулями и протестирована на других студентах из группы практикантов.

Во время практики были закреплены следующие теоретические знания и практические навыки:

1. Должностные инструкции программиста, т.е. необходимые для соблюдения правила на время практики. Должностные инструкции были освоены и полностью соблюдены.
2. Составление целей и задач, которые нужно выполнить для того, чтобы успешно создать тестирующую систему. Весь список поставленных целей и задач был успешно выполнен.
3. Составление плана-графика на время практики. Выполнение всех задач произошло точно по плану.
4. Разработка технического задания на создание тестирующей системы. По итогу техническое задание было выполнено в полном объёме.
5. Профессиональная работа с ЯП С#, IDE MVS, платформой .NET Framework и технологией Windows Forms.
6. Профессиональная работа с СУБД MySQL и её связь с ЯП C#.

В итоге прохождения практики все поставленные цели и задачи были выполнены, теоретические знания получены и освоены, а практические навыки закреплены.

## СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 2.104 ЕСКД (форма 2, форма ) для всех специальностей.
2. ГОСТ 34.601-90 Межгосударственный стандарт. Информационная технология.
3. ГОСТ 7.1-2003 Система стандартов по информации.
4. ГОСТ 19.781-90 «Единая система программной документации. Программное обеспечение систем обраб0отки информации. Термины и определения».
5. ГОСТ 34.321-96 Информационные технологии.
6. Стиллмен Э., Грин Дж. Изучаем C#. 3-е издание.: Издательство Питер СПб.
7. Албахари Дж., Албахари Б. C# 7.0. Карманный справочник. Скорая помощь для программистов на C# 7.0.: Издательство Альфа-Книга
8. Рихтер Д. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд.: Издательство Питер
9. Бэрон Шварц, Петр Зайцев, Вадим Ткаченко, Джереми Заводны, Арьен Ленц, Дерек Боллинг MySQL. Оптимизация производительности.: Издательство Символ-Плюс
10. Документация по .NET [Электронный ресурс]. Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/>

## ПРИЛОЖЕНИЕ А

Данный листинг приводит только классы с бизнес-логикой и не затрагивает классы с от рисовкой графических элементов форм, а также классы внешних модулей и библиотек, в частности для логгирования и работы с базой данных. Также он не включает дополнительные файлы, такие как ресурсы, конфигурации и динамически загружаемые библиотеки. Классы приведены полностью, без сокращений и вырезанных блоков кода.

## **1.Class program**

using NLog;

using System;

using System.Collections.Generic;

using System.Windows.Forms;

namespace Test {

static class Program {

[STAThread]

static void Main() {

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new start());

}

public static void ini() {

data.numQuest=0;

data.numTrueAnswer=0;

data.isCheck=false;

data.score=0;

data.timeSec=0;

data.timeMin=0;

data.timeHour=0;

data.quests.Clear();

data.ids.Clear();

}

public static class data {

public static string idTest { get; set; } // id теста

public static string test { get; set; } // название теста

public static string name { get; set; } // имя пользователя

public static bool isCheck { get; set; } // выбран ли ответ?

public static int numQuest { get; set; } // текущий номер вопроса

public static int numTrueAnswer { get; set; } // правильный ответ на текущий вопрос

public static int score { get; set; } // счёт ответов

public static int timeSec { get; set; } // секунды

public static int timeMin { get; set; } // минуты

public static int timeHour { get; set; } // часы

public static string idUser { get; set; } // id пользователя

public static string login { get; set; } // ник

public static string passHash { get; set; } // хэш пароля

public static string status { get; set; } // статус пользователя

public static bool isAuth { get; set; } // аутентифицирован ли пользователь?

public static List<string> quests=new List<string>(); // список вопросов

public static List<int> ids=new List<int>(); // список id вопросов

}

}

}

## **2.Class start**

using NLog;

using System;

using System.Windows.Forms;

namespace Test {

public partial class start : Form {

public start() {

InitializeComponent();

}

private void button\_primary\_Click(object sender,EventArgs e) {

start.ActiveForm.Hide();

auth auth=new auth();

auth.ShowDialog();

Close();

}

private void button1\_Click(object sender,EventArgs e) {

start.ActiveForm.Hide();

welcome welcome=new welcome();

welcome.ShowDialog();

Close();

}

private void button\_danger\_Click(object sender,EventArgs e) {

Close();

}

}

}

## **3.Class auth**

using MySql.Data.MySqlClient;

using NLog;

using System;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

namespace Test {

public partial class auth : Form {

public auth() {

InitializeComponent();

string currentPath=AppDomain.CurrentDomain.BaseDirectory;

if (!Directory.Exists(Path.Combine(currentPath,"userData"))) {

Directory.CreateDirectory(Path.Combine(currentPath,"userData"));

}

try {

string[] arrStr= File.ReadAllLines(currentPath+"/userData/data.txt");

loginField.Text=cryptoPass.EasyDecryptPassword(arrStr[0]);

passField.Text=cryptoPass.EasyDecryptPassword(arrStr[1]);

checkSavePass.Checked=true;

}

catch {

File.Create(currentPath+"/userData/data.txt");

}

}

private void loginButton\_Click(object sender,EventArgs e) {

string login=loginField.Text;

string pass=passField.Text;

dbMySQL db=new dbMySQL();

MySqlCommand command=new MySqlCommand("SELECT `pass\_hash`,`name`,`id`,`status` FROM `users` WHERE `login`=@login",db.getConnection());

command.Parameters.Add("@login",MySqlDbType.VarChar).Value=login;

db.openConnection();

try {

using (MySqlDataReader readerCommand=command.ExecuteReader()) {

if (readerCommand.Read()) {

if (BCrypt.Net.BCrypt.Verify(pass,readerCommand[0].ToString())) {

if (checkSavePass.Checked == true) {

string currentPath=AppDomain.CurrentDomain.BaseDirectory;

FileStream fs=File.Create(currentPath+"/userData/data.txt");

StreamWriter writer=new StreamWriter(fs);

string localLoginHash=cryptoPass.EasyEncryptPassword(login);

string localPassHash=cryptoPass.EasyEncryptPassword(pass);

writer.Write(localLoginHash+"\n"+localPassHash);

writer.Close();

}

string passHash=BCrypt.Net.BCrypt.HashPassword(pass);

string name =readerCommand[1].ToString();

string id =readerCommand[2].ToString();

string status =readerCommand[3].ToString();

Program.data.idUser=id;

Program.data.login=login;

Program.data.passHash=passHash;

Program.data.name=name;

Program.data.status=status;

Program.data.isAuth=true;

auth.ActiveForm.Hide();

welcome welcome=new welcome();

welcome.ShowDialog();

Close();

}

else {

return;

}

}

else {

return;

}

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

db.closeConnection();

}

private void text\_alert\_secondary\_Click(object sender,EventArgs e) {

auth.ActiveForm.Hide();

reg reg=new reg();

reg.ShowDialog();

Close();

}

}

}

## **4.Class reg**

using MySql.Data.MySqlClient;

using NLog;

using System;

using System.Drawing;

using System.Windows.Forms;

namespace Test {

public partial class reg : Form {

public reg() {

InitializeComponent();

}

private void loginButton\_Click(object sender,EventArgs e) {

string name=nameField.Text;

string login=loginField.Text;

string email=emailField.Text;

string pass=passField.Text;

string pass2=pass2Field.Text;

string passHash=BCrypt.Net.BCrypt.HashPassword(pass);

dbMySQL db=new dbMySQL();

db.openConnection();

MySqlCommand checkUser=new MySqlCommand("SELECT `login` FROM `users` WHERE login=@login OR email=@email",db.getConnection());

checkUser.Parameters.Add("@login",MySqlDbType.VarChar).Value=login;

checkUser.Parameters.Add("@email",MySqlDbType.VarChar).Value=email;

try {

using (MySqlDataReader reader=checkUser.ExecuteReader()) {

if (reader.Read()) {

return;

}

else {

reader.Close();

MySqlCommand insertUser=new MySqlCommand("INSERT INTO `users` (`login`,`name`,`pass\_hash`,`email`) VALUES (@login,@name,@passHash,@email)",db.getConnection());

insertUser.Parameters.Add("@login",MySqlDbType.VarChar).Value=login;

insertUser.Parameters.Add("@name",MySqlDbType.VarChar).Value=name;

insertUser.Parameters.Add("@passHash",MySqlDbType.VarChar).Value=passHash;

insertUser.Parameters.Add("@email",MySqlDbType.VarChar).Value=email;

}

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

db.closeConnection();

}

private void tp\_auth\_form\_Click(object sender,EventArgs e) {

reg.ActiveForm.Hide();

auth auth=new auth();

auth.ShowDialog();

Close();

}

}

}

## **5.Class welcome**

using MySql.Data.MySqlClient;

using NLog;

using System;

using System.Drawing;

using System.Windows.Forms;

namespace Test {

public partial class welcome : Form {

public welcome() {

InitializeComponent();

dbMySQL db=new dbMySQL();

Program.ini();

MySqlCommand command=new MySqlCommand("SELECT id,name,description FROM tests LIMIT 10",db.getConnection());

db.openConnection();

logger.Debug(" "+res.db\_executeQuery);

try {

using (MySqlDataReader readerCommand=command.ExecuteReader()) {

int i=0;

while (readerCommand.Read()) {

Label H2=new Label();

Button button\_start=new Button();

Label H4=new Label();

Panel panel=new Panel();

Controls.Add(panel);

i++;

}

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

db.closeConnection();

if (Program.data.isAuth == true) {

authNameLabel.Visible=true;

authNameField.Visible=true;

authNameField.Text=Program.data.name;

}

else {

nameLabel.Visible=true;

nameField.Visible=true;

}

if (Program.data.status == "a") {

adminButton.Visible=true;

}

}

private void button\_start\_Click(object sender,EventArgs e) {

if (Program.data.isAuth == false) {

if (nameField.Text == "") {

nameField.BackColor=Color.Red;

return;

}

else {

Program.data.name=nameField.Text;

}

}

Program.data.idTest=(sender as Button).Parent.Name;

Program.data.test=((sender as Button).Parent.Controls[2] as Label).Text;

welcome.ActiveForm.Hide();

test Test=new test();

Test.ShowDialog();

Close();

}

private void button\_info\_Click(object sender,EventArgs e) {

welcome.ActiveForm.Hide();

stats stats=new stats();

stats.ShowDialog();

Close();

}

private void button1\_Click(object sender,EventArgs e) {

welcome.ActiveForm.Hide();

admin admin=new admin();

admin.ShowDialog();

Close();

}

}

}

## **6.Class stats**

using MySql.Data.MySqlClient;

using NLog;

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

namespace Test {

public partial class stats : Form {

public stats() {

InitializeComponent();

dbMySQL db=new dbMySQL();

MySqlCommand testsCom=new MySqlCommand("SELECT `id\_test`,COUNT(\*) FROM `logs` GROUP BY `id\_test` HAVING COUNT(\*) >= 5",db.getConnection());

db.openConnection();

List<int> tests=new List<int>();

using (MySqlDataReader readerTestsCom=testsCom.ExecuteReader()) {

while (readerTestsCom.Read()) {

tests.Add(int.Parse(readerTestsCom[0].ToString()));

}

}

for (int i=0; i < tests.Count; i++) {

Label label0=new Label();

Label label1=new Label();

Label label2=new Label();

Label label3=new Label();

Label label4=new Label();

Label label5=new Label();

Panel panel=new Panel();

panel.Controls.Add(label0);

panel.Controls.Add(label1);

panel.Controls.Add(label2);

panel.Controls.Add(label3);

panel.Controls.Add(label4);

panel.Controls.Add(label5);

MySqlCommand stats=new MySqlCommand("SELECT `id\_test`,`name`,`username`,`result`,`mark` FROM `logs`,`tests` WHERE `id\_test`=`tests`.`id` AND `id\_test`=@id ORDER BY `result` DESC LIMIT 5 ",db.getConnection());

stats.Parameters.Add("@id",MySqlDbType.Int32).Value=tests[i];

try {

using (MySqlDataReader readerStats=stats.ExecuteReader()) {

int j=1;

while (readerStats.Read()) {

(panel.Controls[j] as Label).Text=readerStats[2].ToString()+" - "+readerStats[3].ToString()+" баллов ("+readerStats[4].ToString()+")";

logger.Debug(" - "+readerStats[2].ToString()+" - "+readerStats[3].ToString()+" баллов ("+readerStats[4].ToString()+")");

j++;

}

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

Controls.Add(panel);

}

db.closeConnection();

}

private void button\_info\_Click(object sender,EventArgs e) {

stats.ActiveForm.Hide();

welcome welcome=new welcome();

welcome.ShowDialog();

Close();

}

}

}

## **7.Class test**

using MySql.Data.MySqlClient;

using NLog;

using System;

using System.Linq;

using System.Windows.Forms;

namespace Test {

public partial class test : Form {

readonly dbMySQL db=new dbMySQL();

public test() {

InitializeComponent();

H1.Text=Program.data.test;

timer.Start();

Program.ini();

MySqlCommand quests=new MySqlCommand("SELECT id,question FROM questions WHERE id\_test=@id",db.getConnection());

quests.Parameters.Add("@id",MySqlDbType.Int32).Value=Program.data.idTest;

db.openConnection();

logger.Debug(res.db\_executeQueryFirst);

try {

using (MySqlDataReader readerQuests=quests.ExecuteReader()) {

while (readerQuests.Read()) {

Program.data.ids.Add(int.Parse(readerQuests[0].ToString()));

Program.data.quests.Add(readerQuests[1].ToString());

}

Program.data.numQuest=0;

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

Random random=new Random();

for (int i=Program.data.ids.Count - 1; i >= 1; i--) {

int j=random.Next(i+1);

var temp=Program.data.ids[j];

Program.data.ids[j]=Program.data.ids[i];

Program.data.ids[i]=temp;

var temp2=Program.data.quests[j];

Program.data.quests[j]=Program.data.quests[i];

Program.data.quests[i]=temp2;

}

new\_quest();

}

private void button\_success\_Click(object sender,EventArgs e) {

if (Program.data.isCheck) {

if (Program.data.quests.Count() >= Program.data.numQuest) {

if ((radioGroup.Controls[Program.data.numTrueAnswer] as RadioButton).Checked == true) {

Program.data.score++;

}

rb\_unvisible();

new\_quest();

}

}

}

private void rb0\_CheckedChanged(object sender,EventArgs e) {

RadioButton radioButton=(RadioButton)sender;

if (radioButton.Checked) {

Program.data.isCheck=true;

}

else {

Program.data.isCheck=false;

}

}

public void new\_quest() {

if (Program.data.quests.Count() > Program.data.numQuest) {

MySqlCommand answers=new MySqlCommand("SELECT variant,answer FROM variants WHERE id\_question=@id\_quest",db.getConnection());

answers.Parameters.Add("@id\_quest",MySqlDbType.Int32).Value=Program.data.ids[Program.data.numQuest];

H2.Text=Program.data.quests[Program.data.numQuest];

try {

using (MySqlDataReader readerAnswers=answers.ExecuteReader()) {

int i=0;

while (readerAnswers.Read()) {

(radioGroup.Controls[i] as RadioButton).Text=readerAnswers[0].ToString();

(radioGroup.Controls[i] as RadioButton).Visible=true;

if (readerAnswers[1].ToString() == "t") {

Program.data.numTrueAnswer=i;

}

i++;

}

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

Program.data.numQuest++;

}

else {

timer.Stop();

test.ActiveForm.Hide();

result result=new result();

result.ShowDialog();

Close();

}

}

private void button\_danger\_Click(object sender,EventArgs e) {

timer.Stop();

Close();

}

private void timer\_Tick(object sender,EventArgs e) {

Program.data.timeSec++;

if (Program.data.timeSec > 59) {

Program.data.timeSec=0;

Program.data.timeMin++;

}

if (Program.data.timeMin > 59) {

Program.data.timeMin=0;

Program.data.timeHour++;

}

timerSec.Text=Program.data.timeSec.ToString();

timerMin.Text=Program.data.timeMin.ToString();

timerHour.Text=Program.data.timeHour.ToString();

}

}

}

## **8.Class result**

using MySql.Data.MySqlClient;

using NLog;

using System;

using System.Linq;

using System.Windows.Forms;

namespace Test {

public partial class result : Form {

public result() {

InitializeComponent();

H1.Text=Program.data.test;

dbMySQL db=new dbMySQL();

int pers=(int)((float)Program.data.score / Program.data.quests.Count() \* 100);

resultLabel.Text=Program.data.score+" / "+Program.data.quests.Count()+" ("+pers+"%)";

TimeSpan time=new TimeSpan(0,Program.data.timeMin,Program.data.timeSec);

int mark;

if (pers > 90) {

markLabel.Text=res.mark\_5;

}

else if ((pers > 75) && (pers <= 90)) {

markLabel.Text=res.mark\_4;

}

else if ((pers > 50) && (pers <= 75)) {

markLabel.Text=res.mark\_3;

}

else {

markLabel.Text=res.mark\_2;

}

string sql;

if (Program.data.isAuth == true) {

sql="INSERT INTO `practic\_test`.`logs` (`id\_test`,`id\_user`,`result`,`mark`,`time`) VALUES (@id,@iduser,@score,@mark,@time);";

}

else {

sql="INSERT INTO `practic\_test`.`logs` (`id\_test`,`username`,`result`,`mark`,`time`) VALUES (@id,@name,@score,@mark,@time);";

}

MySqlCommand addLog=new MySqlCommand(sql,db.getConnection());

addLog.Parameters.Add("@id",MySqlDbType.Int32).Value=Program.data.idTest;

addLog.Parameters.Add("@score",MySqlDbType.Int32).Value=Program.data.score;

addLog.Parameters.Add("@mark",MySqlDbType.Int32).Value=mark;

addLog.Parameters.Add("@time",MySqlDbType.Time).Value=time;

if (Program.data.isAuth == true) {

addLog.Parameters.Add("@iduser",MySqlDbType.Int32).Value=Program.data.idUser;

}

else {

addLog.Parameters.Add("@name",MySqlDbType.VarChar).Value=Program.data.name;

}

db.openConnection();

try {

addLog.ExecuteNonQuery();

}

catch {

Program.err(res.err\_dbNotQuery);

}

db.closeConnection();

}

private void button\_success\_Click(object sender,EventArgs e) {

result.ActiveForm.Hide();

welcome welcome=new welcome();

welcome.ShowDialog();

Close();

}

private void button\_danger\_Click(object sender,EventArgs e) {

Close();

}

private void button\_warning\_Click(object sender,EventArgs e) {

result.ActiveForm.Hide();

test test=new test();

test.ShowDialog();

Close();

}

}

}

## **9.Class admin**

using MySql.Data.MySqlClient;

using NLog;

using System;

using System.Drawing;

using System.Linq;

using System.Windows.Forms;

namespace Test {

public partial class admin : Form {

readonly dbMySQL db=new dbMySQL();

public admin() {

InitializeComponent();

Program.ini();

}

private void admin\_Load(object sender,EventArgs e) {

this.testsTableAdapter.Fill(this.practic\_testDataSet.tests);

}

private void comboBox1\_SelectedIndexChanged(object sender,EventArgs e) {

panelMain.Controls.Clear();

Program.ini();

db.openConnection();

logger.Debug(res.db\_createQuery);

MySqlCommand quests=new MySqlCommand("SELECT id,question FROM questions WHERE id\_test=@idtest",db.getConnection());

quests.Parameters.Add("@idtest",MySqlDbType.VarChar).Value=comboBox1.SelectedValue;

try {

using (MySqlDataReader readerQuests=quests.ExecuteReader()) {

while (readerQuests.Read()) {

Program.data.ids.Add(int.Parse(readerQuests[0].ToString()));

Program.data.quests.Add(readerQuests[1].ToString());

}

Program.data.numQuest=0;

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

while (Program.data.quests.Count() > Program.data.numQuest) {

MySqlCommand answers=new MySqlCommand("SELECT variant,answer FROM variants WHERE id\_question=@id\_quest",db.getConnection());

answers.Parameters.Add("@id\_quest",MySqlDbType.Int32).Value=Program.data.ids[Program.data.numQuest];

TextBox answersLabel=new TextBox();

TextBox questLabel =new TextBox();

Panel panel1 =new Panel();

RadioButton radioButton1=new RadioButton();

RadioButton radioButton2=new RadioButton();

RadioButton radioButton3=new RadioButton();

RadioButton radioButton4=new RadioButton();

RadioButton radioButton5=new RadioButton();

RadioButton radioButton6=new RadioButton();

RadioButton radioButton7=new RadioButton();

questLabel.Text=Program.data.quests[Program.data.numQuest];

try {

using (MySqlDataReader readerAnswers=answers.ExecuteReader()) {

int i=0;

while (readerAnswers.Read()) {

answersLabel.Text += readerAnswers[0].ToString()+Environment.NewLine;

if (readerAnswers[1].ToString() == "t") {

(panel1.Controls[i] as RadioButton).Checked=true;

}

i++;

}

answersLabel.Text=answersLabel.Text.Remove(answersLabel.Text.Length - 2);

}

}

catch {

Program.err(res.err\_dbNotQuery);

}

panelMain.Controls.Add(answersLabel);

panelMain.Controls.Add(questLabel);

panelMain.Controls.Add(panel1);

Program.data.numQuest++;

}

db.closeConnection();

}

private void answersLabel\_TextChanged(object sender,EventArgs e) {

TextBox TextBox=(TextBox)sender;

foreach (string line in TextBox.Lines) {

if (line.Length > 60) {

TextBox.Undo();

}

}

TextBox.ClearUndo();

}

private void button\_danger\_Click(object sender,EventArgs e) {

start.ActiveForm.Hide();

welcome welcome=new welcome();

welcome.ShowDialog();

Close();

}

private void button\_primary\_Click(object sender,EventArgs e) {

dialog\_newTest dialog\_newTest=new dialog\_newTest();

dialog\_newTest.ShowDialog();

this.testsTableAdapter.Fill(this.practic\_testDataSet.tests);

}

private void button\_success\_Click(object sender,EventArgs e) {

db.openConnection();

Program.data.idTest=comboBox1.SelectedValue.ToString();

MySqlCommand delTest=new MySqlCommand("DELETE FROM `questions` WHERE id\_test=@idtest;",db.getConnection());

delTest.Parameters.Add("@idtest",MySqlDbType.Int32).Value=Program.data.idTest;

try {

delTest.ExecuteNonQuery();

}

catch {

Program.err(res.err\_dbNotQuery);

}

int i=0;

int numQuest=1;

int numQuestDB=0;

char ans;

while (i < panelMain.Controls.Count) {

MySqlCommand insertQuest=new MySqlCommand("INSERT INTO `questions` (`id\_test`,`question`) VALUES (@idtest,@quest);",db.getConnection());

insertQuest.Parameters.Add("@idtest",MySqlDbType.Int32).Value=Program.data.idTest;

insertQuest.Parameters.Add("@quest",MySqlDbType.TinyText).Value=(panelMain.Controls[i+1] as TextBox).Text;

try {

insertQuest.ExecuteNonQuery();

}

catch {

Program.err(res.err\_dbNotQuery);

}

MySqlCommand lastId=new MySqlCommand("SELECT LAST\_INSERT\_ID();",db.getConnection());

try {

using (MySqlDataReader readerLastId=lastId.ExecuteReader()) {

if (readerLastId.Read()) {

numQuestDB=int.Parse(readerLastId[0].ToString());

}

}

for (int a=0; a < ((panelMain.Controls[i] as TextBox).Lines.Length); a++) {

MySqlCommand addAnswer=new MySqlCommand("INSERT INTO `variants` (`id\_question`,`variant`,`answer`) VALUES (@idquest,@variant,@answer);",db.getConnection());

addAnswer.Parameters.Add("@idquest",MySqlDbType.Int32).Value=numQuestDB;

addAnswer.Parameters.Add("@variant",MySqlDbType.VarChar).Value=(panelMain.Controls[i] as TextBox).Lines[a].ToString();

if (((panelMain.Controls[i+2] as Panel).Controls[a] as RadioButton).Checked) {

addAnswer.Parameters.Add("@answer",MySqlDbType.VarChar).Value="t";

ans='+';

}

else {

addAnswer.Parameters.Add("@answer",MySqlDbType.VarChar).Value="f";

ans=' ';

}

try {

addAnswer.ExecuteNonQuery();

}

catch (Exception ex) {

Program.err(ex.Message);

Program.err(res.err\_dbNotQuery);

}

}

}

catch (Exception ex) {

Program.err(res.err\_dbNotQuery);

}

i += 3;

numQuest++;

}

db.closeConnection();

}

}

}

## ПРИЛОЖЕНИЕ Б

|  |  |
| --- | --- |
| C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд1.png | C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд2.png |
| C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд4.png | C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд3.png |
| C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд5.png | C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд6.png |

|  |  |
| --- | --- |
| C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд7.png | C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд8.png |
| C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд9.png | C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд10.png |
| C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд11.png | C:\Users\Student\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Слайд12.png |