Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет издательского дела и полиграфии

Кафедра информационных систем и технологий

Специальность 1-40 01 02 «информационные системы и технологии»

Специализация «издательско-полиграфический комплекс»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОЙ РАБОТЫ**

по дисциплине «Объектно-ориентированное программирование»

Тема «Программное средство “Школа - электронный дневник”»

Исполнитель

студент 2 курса группы 8 \_\_\_\_\_\_\_\_\_\_ Э.Б Келин

подпись, дата

Руководитель

доцент, к.т.н. \_\_\_\_\_\_\_\_\_\_\_ Н.В. Пацей

подпись, дата

Курсовая работа защищена с оценкой

Руководитель Н.В. Пацей

подпись

Минск 2014

Оглавление

[Введение.......................................................................................................................3](#_Toc386922654)

[1.Теоретическое введение.......................................................................................4](#_Toc386922655)-[5](#_Toc386922656)

[1.2 С# WinForm............................................................................................................6](#_Toc386922657)

1.3 Microsoft SQL Server………………………………………………………….....7

1.4 Среда Visual Studio.NET………………………………………………………...8

[2. Проектирование проект-схемы программы и диаграммы классов...............](#_Toc386922659)9-10

2.1 Проектирование и создание базы данных…….……………………................11

[3. Интерфейс программы.....................................................................................12-14](#_Toc386922660)

[4.Тестирование.....................................................................................................15-17](#_Toc386922661)

[Заключение.................................................................................................................18](#_Toc386922662)

[Список используемой литературы...........................................................................](#_Toc386922663)19

# Введение

С точки зрения программирования объектный подход можно рассматривать как развитие понятия типов данных. Тип данного определяет множество значений, которые оно может принимать, и набор операций, которые могут быть выполнены над данными этого конкретного типа. В языках программирования со строгой типизацией каждое данное принадлежит только одному типу, и разрешаются только явные преобразования данных одного типа в другой. Строгая типизация повышает надежность разрабатываемых программ, позволяет выявить многие ошибки еще на этапе отладки программы. В языках программирования предусматривается и возможность построения новых типов данных, определяемых программистом, но для таких типов, как правило, отсутствует перечень допустимых операций. Объектно-ориентированные языки программирования позволяют распространить требования строгой типизации на типы данных, определяемых программистом.

Систематическое применение объектно-ориентированного подхода позволяет разрабатывать хорошо структурированные, надежные в эксплуатации, достаточно просто модифицируемые программные системы. Этим объясняется интерес программистов к объектно-ориентированному подходу и объектно-ориентированным языкам программирования. Объектно-ориентированный подход является одним из наиболее интенсивно развивающихся направлений теоретического и прикладного программирования.

За последние несколько лет объектно-ориентированная технология проникла в различные разделы компьютерных наук. К ней относятся как к средству преодоления сложности, присущей многим реальным системам. Объектная модель показала себя мощной объединяющей концепцией.

Составными частями любой информационной системы являются базы данных и приложение для обработки данных.

## 1.Теоретическое введение

# 1.1 Объектно-ориентированное программирование

# ООП возникло в результате развития идеологии процедурного программирования, где данные и подпрограммы (процедуры, функции) их обработки формально не связаны. Для дальнейшего развития объектно-ориентированного программирования часто большое значение имеют понятия события (так называемое событийно-ориентированное программирование) и компонента (компонентное программирование, КОП).

# Взаимодействие объектов происходит посредством сообщений. Результатом дальнейшего развития ООП, по-видимому, будет агентно-ориентированое программирование, где агенты — независимые части кода на уровне выполнения. Взаимодействие агентов происходит посредством изменения среды, в которой они находятся.

# Языковые конструкции, конструктивно не относящиеся непосредственно к объектам, но сопутствующие им для их безопасной (исключительные ситуации, проверки) и эффективной работы, инкапсулируются от них в аспекты (в аспектно-ориентированном программировании). Субъектно-ориентированное программирование расширяет понятие объекта посредством обеспечения более унифицированного и независимого взаимодействия объектов. Может являться переходной стадией между ООП и агентным программированием в части самостоятельного их взаимодействия.Первым языком программирования, в котором были предложены принципы объектной ориентированности, была Симула. В момент его появления в 1967 году в нём были предложены революционные идеи: объекты, классы, виртуальные методы и др., однако это всё не было воспринято современниками как нечто грандиозное. Тем не менее, большинство концепций были развиты Аланом Кэем и Дэном Ингаллсом в языке Smalltalk. Именно он стал первым широко распространённым объектно-ориентированным языком программирования.

# В настоящее время количество прикладных языков программирования (список языков), реализующих объектно-ориентированную парадигму, является наибольшим по отношению к другим парадигмам. В области системного программирования до сих пор применяется парадигма процедурного программирования, и общепринятым языком программирования является Cи. При взаимодействии системного и прикладного уровней операционных систем заметное влияние стали оказывать языки объектно-ориентированного программирования. Например, одной из наиболее распространённых библиотек мультиплатформенного программирования является объектно-ориентированная библиотека Qt, написанная на языке.

# Сущность в адресном пространстве вычислительной системы, появляющаяся при создании экземпляра класса или копирования прототипа (например, после запуска результатов компиляции и связывания исходного кода на выполнение).

# Прототип — это объект-образец, по образу и подобию которого создаются другие объекты. Объекты-копии могут сохранять связь с родительским объектом, автоматически наследуя изменения в прототипе; эта особенность определяется в рамках конкретного языка.

# 1.2. C# WinForm C# (произносится си-шарп) — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как основной язык разработки приложений для платформы Microsoft .NET. Компилятор с C# входит в стандартную установку самой .NET, поэтому программы на нём можно создавать и компилировать даже без инструментальных средств, вроде Visual Studio. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеетстатическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы. Переняв многое от своих предшественников — языков C++, Java, Delphi, Модула и Smalltalk — С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем: так, C# не поддерживает множественное наследование классов (в отличие от C++). C# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов C#, которая отражает BCL. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем. (Однако эта закономерность была нарушена с выходом C# 3.0, представляющим собой расширения языка, не опирающиеся на расширения платформы .NET.) CLR предоставляет C#, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на VB.NET, J# и др.

# 

# 1.3. Microsoft SQL Server

Для управления своими базами данных я предпочёл использовать Microsoft SQL Server.

SQL Server Management Studio — утилита из [Microsoft SQL Server](http://ru.wikipedia.org/wiki/Microsoft_SQL_Server) 2005 и более поздних версий для конфигурирования, управления и [администрирования](http://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B5_%D0%B0%D0%B4%D0%BC%D0%B8%D0%BD%D0%B8%D1%81%D1%82%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) всех компонентов Microsoft SQL Server. Утилита включает [скриптовый](http://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82) редактор и графическую программу, которая работает с объектами и настройками сервера.

Главным инструментом SQL Server Management Studio является Object Explorer, который позволяет пользователю просматривать, извлекать объекты сервера, а также полностью ими управлять.

SQL является прежде всего [информационно-логическим языком](http://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA), предназначенным для описания, изменения и извлечения данных, хранимых в [реляционных базах данных](http://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). SQL можно назвать [языком программирования](http://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%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%D1%8F), при этом он не является [тьюринг-полным](http://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D0%BE%D1%82%D0%B0_%D0%BF%D0%BE_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3%D1%83), но вместе с тем стандарт языка спецификацией [SQL/PSM](http://ru.wikipedia.org/wiki/SQL/PSM) предусматривает возможность его [процедурных расширений](http://ru.wikipedia.org/wiki/SQL#.D0.9F.D1.80.D0.BE.D1.86.D0.B5.D0.B4.D1.83.D1.80.D0.BD.D1.8B.D0.B5_.D1.80.D0.B0.D1.81.D1.88.D0.B8.D1.80.D0.B5.D0.BD.D0.B8.D1.8F).

Изначально SQL был основным способом работы пользователя с [базой данных](http://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) и позволял выполнять следующий набор операций:

* создание в базе данных новой таблицы;
* добавление в таблицу новых записей;
* изменение записей;
* удаление записей;
* выборка записей из одной или нескольких таблиц (в соответствии с заданным условием);
* изменение структур таблиц.

При всех своих изменениях SQL остаётся единственным механизмом связи между прикладным программным обеспечением и базой данных. В то же время современные СУБД, а также информационные системы, использующие СУБД, предоставляют пользователю развитые средства визуального построения запросов.

Таким образом использование SQL сводится, по сути, к формированию всевозможных выборок строк и совершению операций над всеми записями, входящими в набор.

# 1.4. Среда Visual Studio.NET

Имена нынешнего поколения продуктов от Microsoft сопровождаются окончанием .Net (читается Dot Net), отражающим видение Microsoft современного коммуникативного мира.

За именем .Net в программных продуктах стоит вполне конкретное содержание, которое предполагает в частности, наличие открытых стандартов коммуникации, переход от создания монолитных приложений к созданию компонентов, допускающих повторное использование в разных средах и приложениях. Всё это непременные атрибуты новых технологий, в которых важную роль играет язык XML, ставший стандартом обмена сообщениями в сети.

Visual Studio.NET – это открытая среда разработки, в которой ярко выражены 2 идеи:

* Открытость для языков программирования;
* Принципиально новый подход к построению каркаса среды – Framework .Net.

Открытость.

Это означает что на ряду с языками программирования, включенными в среду фирмой Microsoft – Visual C++ .Net (c управляемыми расширениями) , Visual C# .Net, Visual Basic .Net, в среду могут добавляться любые языки программирования, компиляторы которых создаются другими фирмами-производителями.

Открытость среды не означает полной свободы. Все разработчики компиляторов при включении нового языка в среду разработки должны следовать определённым ограничениям. Одним из главных таких ограничений можно считать то, что все языки, включаемые в среду разработки Visual Studio .Net, должны использовать единый каркас -- Framework .Net. Благодаря, которому ,достигаются такие свойства, как легкость использования компонентов, разработанных на различных языках; возможность написать класс на одном языке, а его потомков – на других языках и т.д..Преодоление языкового барьера является одной из главных задач нашего современного мира. Но благодаря единому каркасу, Visual Studio .Net в определенной мере решает эту задачу в мире программистов.

В каркасе Framework .Net можно выделить два основных компонента:

* Статический (библиотека классов каркаса);
* Динамический (общеязыковая исполнительная среда).

# 2. Проектирование проект-схемы программы и диаграммы классов

Необходимые нам формы – это форма главного окна, в которой будут отображаться необходимая администратору информация. Далее, нужны формы добавления диска, добавления клиента, выдачи диска, и форма возврата диска. Таким образом, мы обеспечим нашему приложению удобный и простой интерфейс.

На рисунке 2.1 мы видим проект-схему программы.

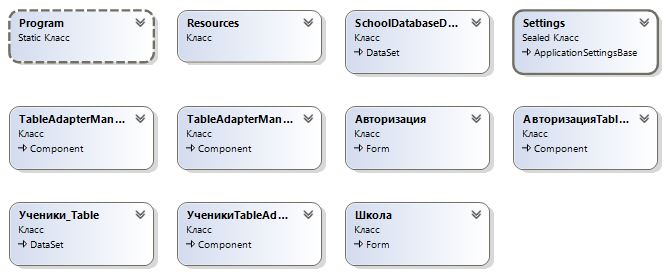


Рисунок 2.1 – проект-схема программы.

На рисунке 2.2 представлена диаграмма классов и их функции.

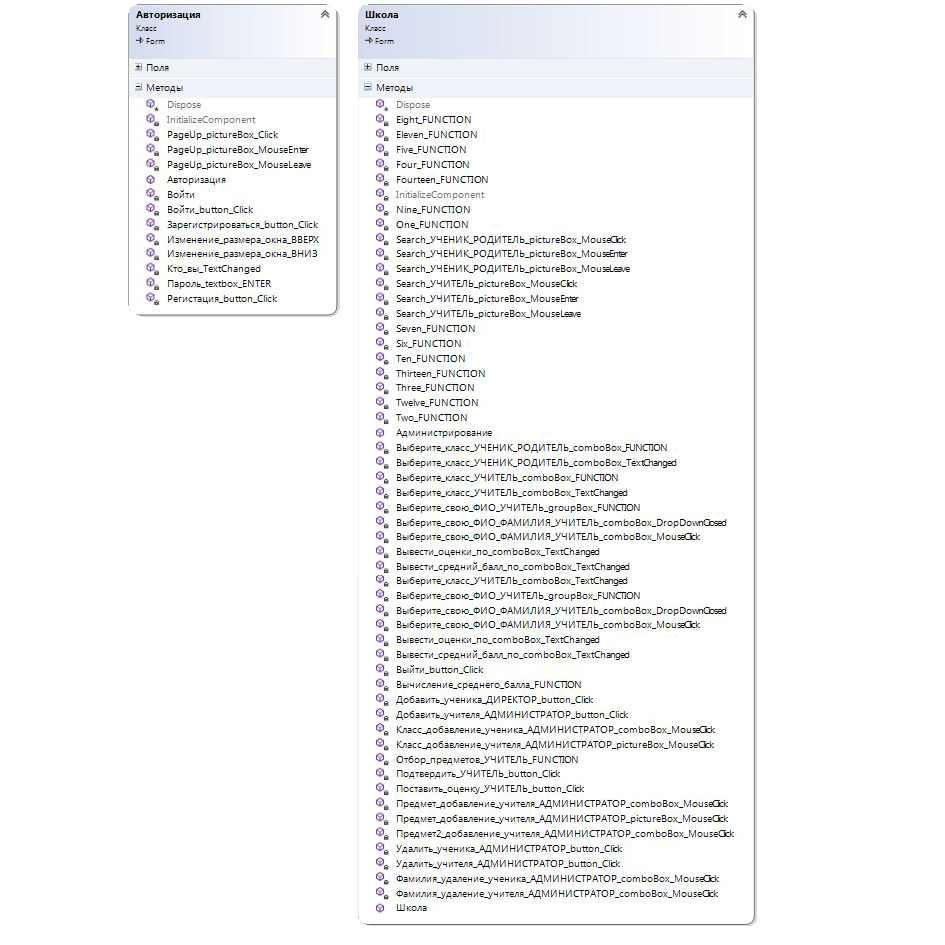


Рисунок 2.2 – диаграмма классов и функций.

**2.1. Проектирование и создание базы данных**

Выполним физическое проектирование в среде СУБД Microsoft SQL Server. Зададим имена таблиц и полей, определим типы данных и размерность полей таблиц. В таблицах выберем первичные ключи и индексированные поля. Так же для поля определим его основные характеристики – является ли это поле внешним или первичным ключом, создан ли индекс по этому полю, задано ли для поля значение по умолчанию, какие ограничения заданы для данного поля (уникальность значений, маска ввода). Вся эта информация представлена на рисунке 2.3.

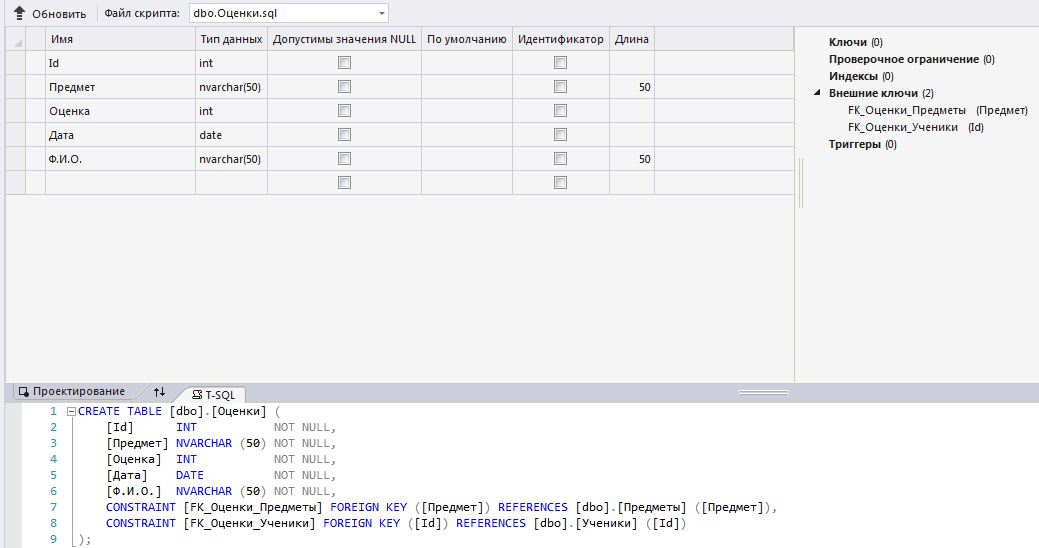


Рисунок 2.3 – Структура таблиц базы данных «Школа - электронный дневник»

А следующая диаграмма иллюстрирует таблицу в базе.

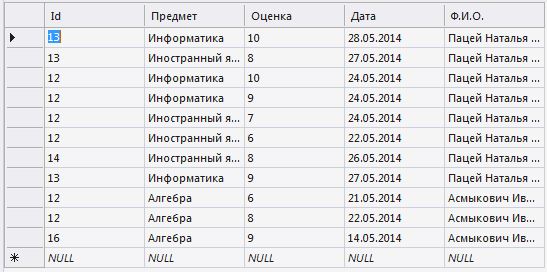


Рисунок 2.4 – таблица в базе данных

В таблице представлены следующие столбцы: id — осуществление доступа к любому из учеников осуществляется через него, задается автоматически. Предмет, оценка, дата и Ф.И.О. учителя - основные составляющие этой таблицы, так как на их основе можно узнать какой учитель поставил оценку, по какому предмету поставил оценку, дату выставления оценки и собственно саму оценку.

**3. Интерфейс программы**

Интерфейс — совокупность возможностей взаимодействия двух систем, устройств или программ, определённая их характеристиками, характеристиками соединения, сигналов обмена и т. п..

Оконный интерфейс — самый распространенный (по крайней мере, на сегодняшний день) способ организации взаимодействия пользователя и компьютера. Несмотря на наличие большого количества программ, их интерфейсы стандартизированы и содержат такие элементы как заголовок, меню, панель инструментов, рабочую область, строку состояния. Каждая более-менее сложная пользовательская программа имеет те или иные специфические особенности в организации своего интерфейса. Зачастую интерфейс любой программы настраиваем, и может отличаться от стандартного. Современные операционные системы позволяют запускать множество программ одновременно, а следовательно отрывать далеко не одно окно. Поэтому необходимо уметь организовывать удобное для работы расположение окон на экране.

Первое что открывается при запуске программы это окно авторизации с возможностью регистрации (Рисунок 3.1)

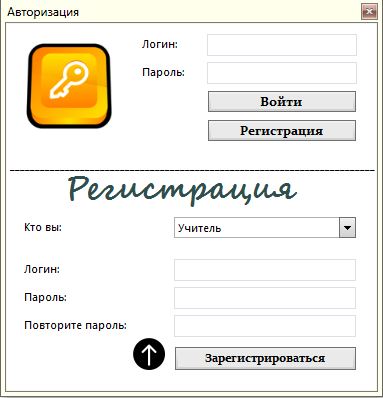
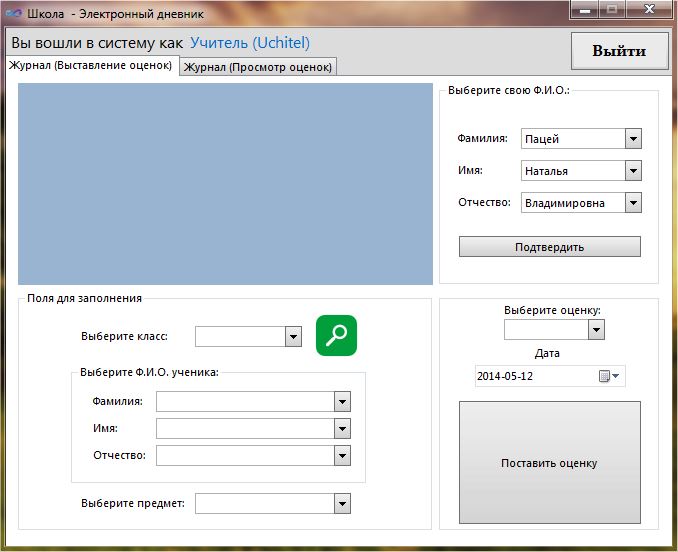


Рисунок 3.1 – окно авторизации и регистрации

# После успешной авторизации у вас появится главное окно для работы с электронным дневником.

Главное окно программы для учителя выглядит состоит из двух страниц: страницы для выставления оценок и страницы для просмотра оценок. Скриншот страницы с выставлением оценок выглядит следующим образом (Рисунок 3.2), а страницу для просмотра оценок можно увидеть на рисунке 3.3.

Рисунок 3.2 – страница с выставлением оценок для учителя

Чтобы поставить оценку ученику учителю необходимо прежде всего выбрать свою Ф.И.О. из списка. Затем появятся дополнительные поля для выбора ученика в классе с возможностью просмотра учеников в выбранном вами классе, затем выбрать предмет по которому вы хотите поставить оценку, собственно саму оценку и дату. После заполнения всех перечисленных полей нажать кнопку « Поставить оценку ».

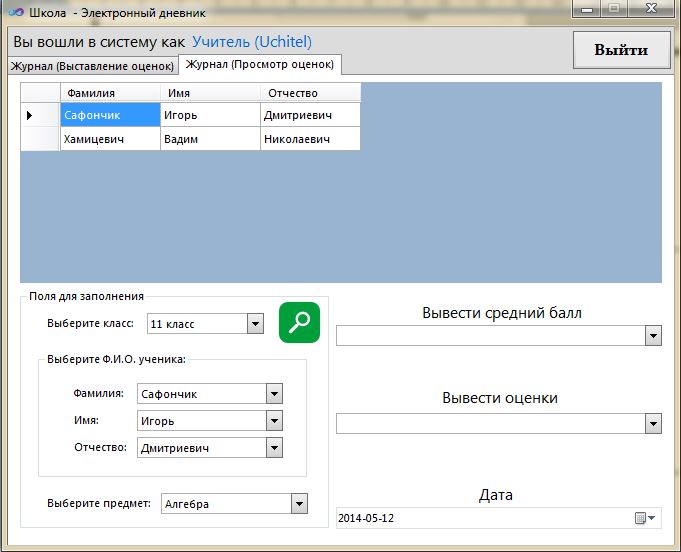


Рисунок 3.3 – с выставлением оценок для учителя

Эта страница позволяет просмотреть оценки любого ученика по определённым критериям.

# 4. Тестирование

Данное программное средство тестировалось вручную, были выполнены все технические моменты предусмотренные реализацией. Программное средство тестировалось на разных компьютерах.

1. В случае, если вы ввели при регистрации уже существующий логин вы увидите следующую ошибку, смотрите рисунок 4.1

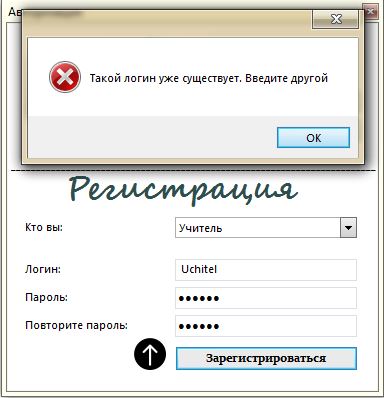


Рисунок 4.1 – Исключение, такой логин уже существует.

1. При попытке ввести при регистрации логин с количеством символов больше 50, то у вас появится такая ошибка, рисунок 4.2.

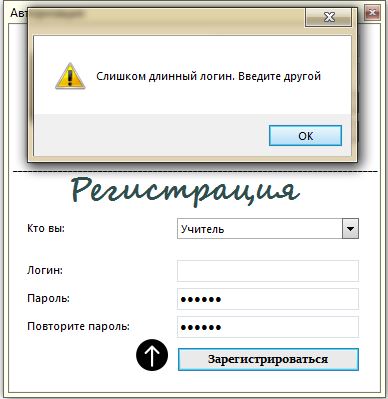
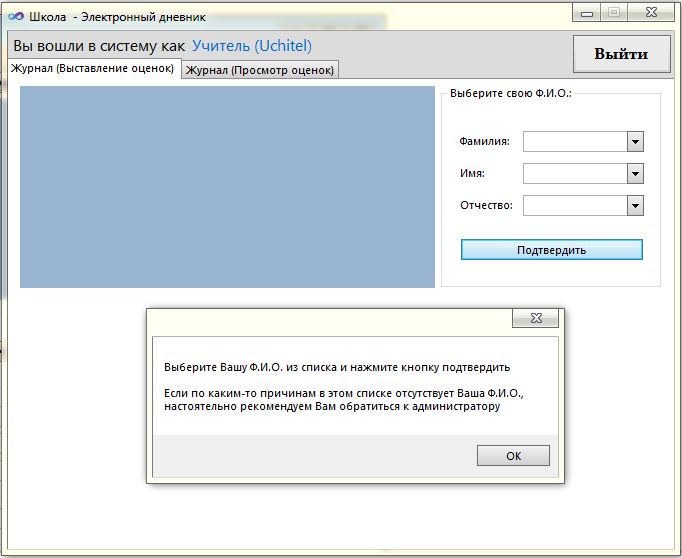


Рисунок 4.2 – Исключение, слишком длинный логин.

1. Если Вы вошли под пользователем «Учитель», то первое, что необходимо вам сделать это выбрать Вашу Ф.И.О. и нажать кнопку «Подтвердить». В случае, если вы не выбрали Ф.И.О. из поиска и нажали кнопку «Подтвердить», то у вас появится ошибка, показанная на рисунке 4.3.



# Рисунок 4.3 – Исключение, выберите Вашу Ф.И.О. из списка.

# Заключение

В данном курсовом проекте было разработано программное средство “Школа - электронный дневник” при помощи C# Windows Form и Microsoft SQL Server. При разработке данного программного средства мы не могли обойтись без базы данных SQL Server. Это дало возможность разработки простого, но в тоже время интуитивно понятного интерфейса и использования БД, в котором создавалось огромное количество запросов, таких как: просмотр успеваемости учеников в школе, вычисление среднего балла конкретного ученика, подсчёт среднего балла школы, просмотр оценок полученных в конкретный день по конкретному предмету и др.

Данное приложение соответствует поставленной задаче и отвечает всем требованиям, необходимым для пользования данным программным средством “Школа - электронный дневник”.

# Список используемой литературы

1. Албахари Дж., Албахари Б. С# [Текст]: Карманный справочник/ Албахари Дж., Албахари Б. — СПб.: Питер, 2013. — 598 с.
2. Уотсон К., Нейгел К.Visual C# 2010. Полный курс/ Уотсон К., Нейгел К. — СПб.: Питер, 201. — 653 с.
3. Зиборов В.В. Visual С# 2012 на примерах/ Зиборов В.В. — СПб.: Питер, 2012. — 723 с.